EPSON
EXCEED YOUR VISION

S1R72V05 Technical Manual

SEIKO EPSON CORPORATION

Rev.1.3

General Rules

Scope of Application

This specification applies to the USB2.0 Controller “S1R72V05**” manufactured by the Semiconductor Operations Division of Seiko Epson Corporation.

Table of Contents

1. Overview

The S1R72V05** is a USB host/device controller LSI that supports USB2.0-compliant high-speed mode. This LSI incorporates host ports and device ports independently, so that when control is switched over it can operate as a USB host or as a USB device.

In addition, this LSI incorporates an IDE interface, making it suitable for portable equipment that has a built-in hard disk.

2. Features

<< USB2.0 device functions >>

- Supports HS <480 Mbps) and FS (12 Mbps) transfers.
- Includes FS/HS termination (external circuit unnecessary).
- Includes VBUS 5V interface (external circuit unnecessary).
- Supports control, bulk, and interrupt transfers.
- Supports two endpoints for Bulk transfer, one endpoint for Interrupt transfer, and Endpoint 0.

<< USB2.0 host functions >>

- Supports HS (480 Mbps), FS (12 Mbps), and LS (1.5 Mbps) transfers.
- Includes pull-down resistors for downstream ports (external circuit unnecessary).
- Includes HS termination (external circuit unnecessary).
- Supports control, bulk, and interrupt transfers.
  Channel structure
  One (1) channel used exclusively for Control transfer
  One (1) channel used exclusively for Bulk transfer
  Four (4) channels used for Bulk and Interrupt transfers
- USB power switch interface.

<< Media data transfer functions >>

- Capable of data transfer between IDE and CPU while USB function is turned off.
- Includes a dedicated FIFO (independently of the FIFO used for USB).

<< CPU interface >>

- Accepts 16-bit or 8-bit wide general-purpose CPU interfaces.
- Incorporates two DMA channels (Multiword transfer).
- Big Endian (incorporating a bus swap function for Little Endian CPUs).
- Changeable interface voltages (3.3 V or 1.8 V).

<< IDE interface >>

- Supports ATA/ATAPI6
  PIO modes 0–4, Multi-word DMA, and UDMA modes 0–5

<< Other >>

- Accepts a 12 MHz/24 MHz crystal resonator for clock input. (built-in Oscillator circuit and 1M Ωfeedback resistor)
- Triple-power supply system: 3.3 V, 1.8 V and variable CPU interface power
- Package type: QFP15-128, PFBGA8UX121, PFBGA10UX121
- Guaranteed operation temperature range: –40°C to 110°C

* This LSI is not designed to resist radiation.

3. Block Diagram

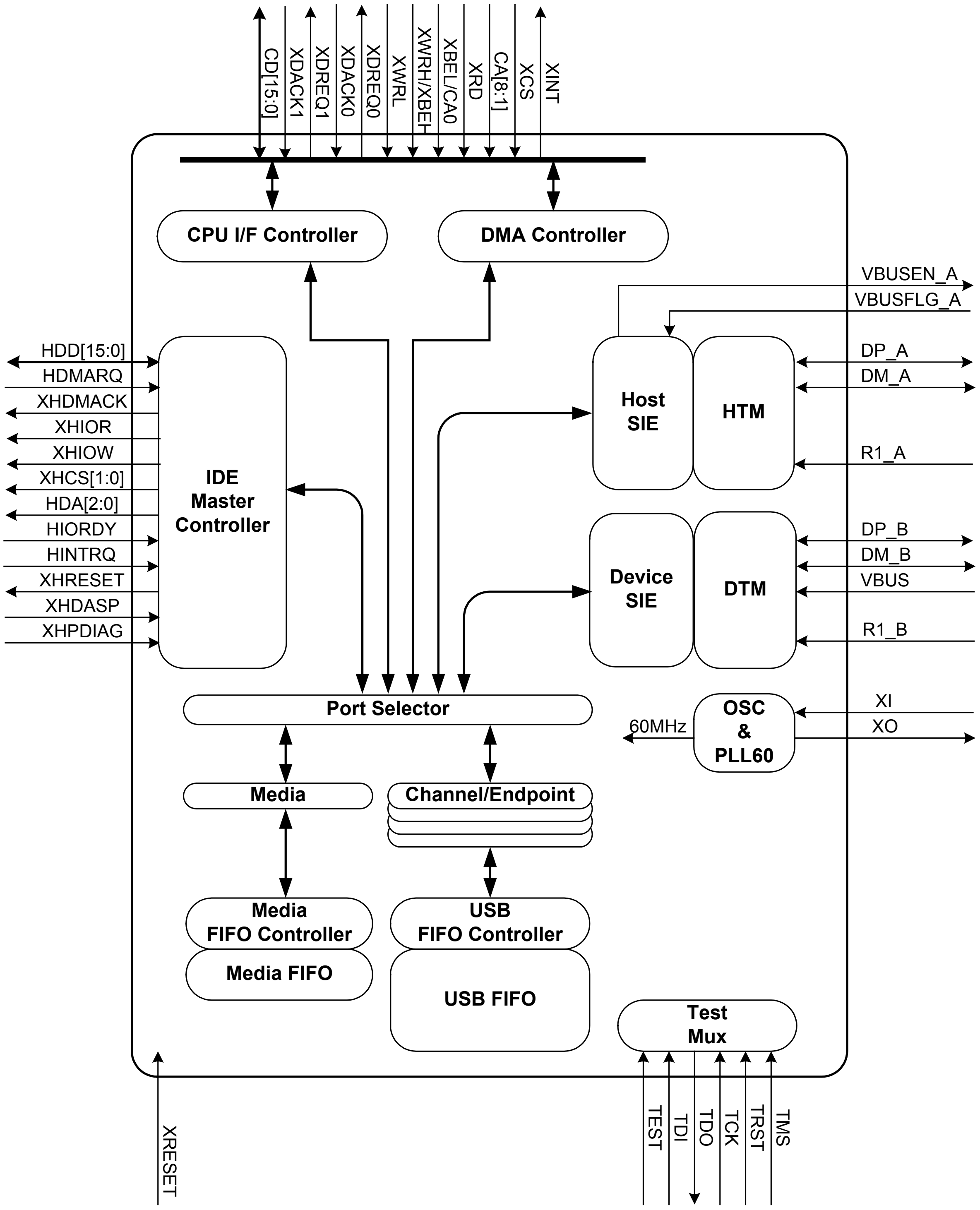


Fig. 3.1 General block diagram

3.1 Device Transceiver Macro (DTM)

This is a USB2.0 transceiver macro compliant with UTMI1.03 specification. Incorporating an analog circuit and high-speed logic circuit, it supports HS mode (480 Mbps) and FS mode (12 Mbps).

In addition to the transmitter, receiver, termination, and squelch circuit for HS/FS support, it incorporates an elasticity buffer, serial-parallel/parallel-serial conversion circuit, bit stuff/unstuff circuit, SYNC/EOP add/remove circuit, etc., which together comprise a USB interface.

Furthermore, it has a built-in PLL that generates a 480 MHz clock needed for HS transfer. Internal oscillator is the clock source of the PLL.

3.2 Host Transceiver Macro (HTM)

This is a USB2.0 transceiver macro provided for use by the host. Incorporating an analog circuit and high-speed logic circuit, it supports HS mode (480 Mbps), FS mode (12 Mbps), and LS mode (1.5 Mbps).

In addition to the transmitter, receiver, termination, squelch circuit, cutoff envelope detection circuit for HS/FS support, it incorporates an elasticity buffer, serial-parallel/parallel-serial conversion circuit, bit stuff/unstuff circuit, SYNC/EOP add/remove circuit, etc., which together comprise a USB interface.

Furthermore, it has a built-in PLL that generates a 480 MHz clock needed for HS transfer.

3.3 Oscillator & PLL60

With an oscillator circuit built-in, this block generates a 600 MHz clock needed for operation of the internal logic. The input clock for the oscillator circuit can be supplied from a 12 MHz/24 MHz crystal resonator.

3.4 Device Serial Interface Engine (Device SIE)

This block manages transactions and generates packets. Furthermore, it controls bus events such as suspend, resume, and reset.

3.5 Host Serial Interface Engine (Host SIE)

This block schedules transactions, manages transactions, and generates packets.

Furthermore, it controls bus events such as suspend, resume, and reset.

It also detects connect/disconnect status and controls the VBUS (in cooperation with an external USB power switch).

3.6 Port Selector

This block switches connections of the CPU(DMA), IDE, DeviceSIE, or HostSIE ports with the USB FIFO or Media FIFO.

3.7 USB FIFO and USB FIFO Controller

These blocks comprise a channel/endpoint buffer.

3.8 Media FIFO and Media FIFO Controller

These blocks comprise a Media buffer.

3.9 CPU I/F Controller

Controls the CPU interface timing, allowing registers to be accessed properly.

3.10 DMA Controller

Controls the DMA timing of the CPU interface, allowing access to FIFO. It incorporates two DMA channels.

3.11 IDE Master Controller

Provides a UATA100-compatible IDE interface.

3.12 Test MUX

This is a test circuit.

4. Pin Layout Diagram

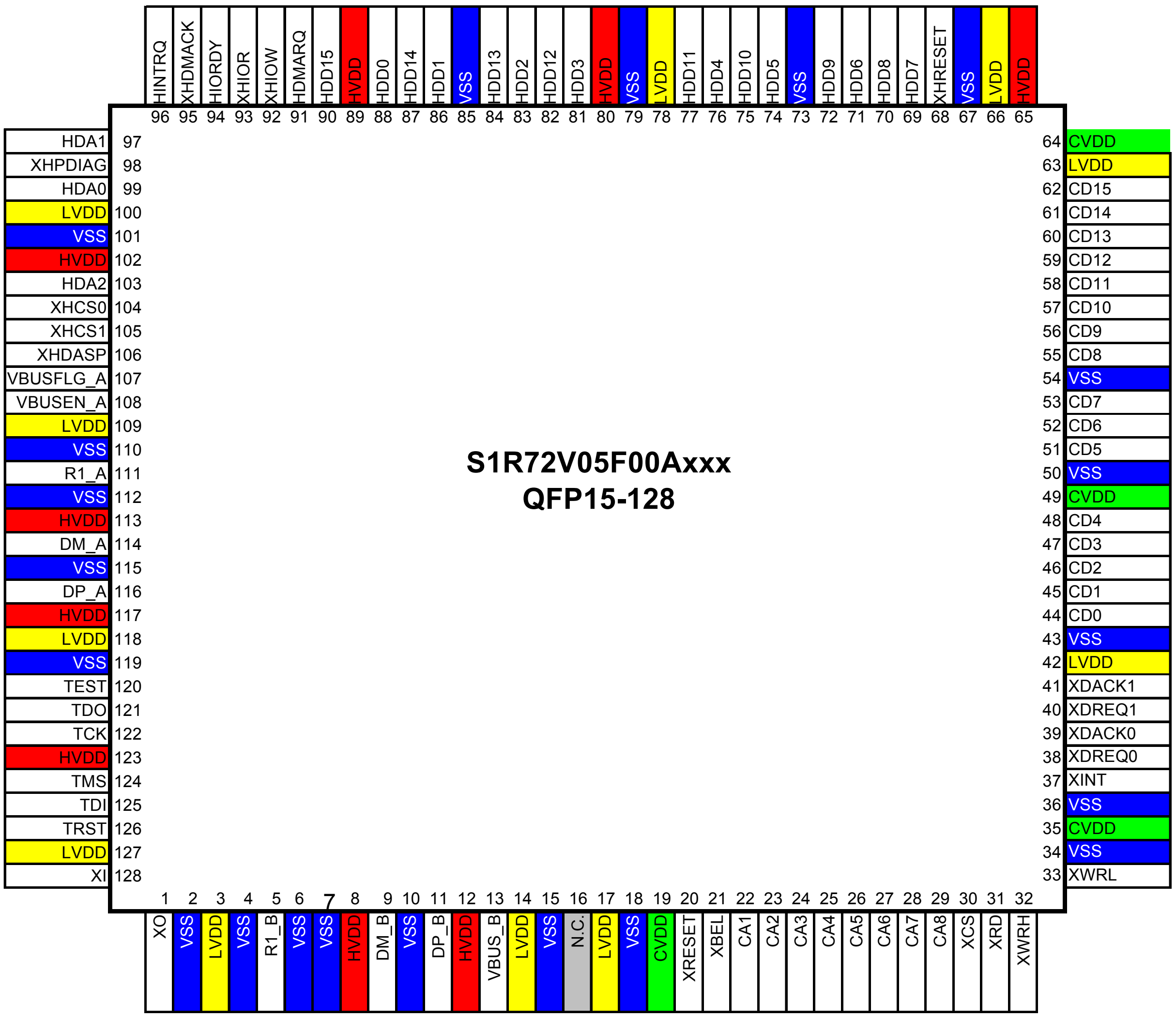


Fig. 4.1 Pin Layout Diagram of the QFP package

S1R72V05B00A2XX/PFBGA8UX121
S1R72V05B00B3XX/PFBGA10UX121
TOP View

1	2	3	4	5	6	7	8	9	10	11	
NC	XI	LVDD	LVDD	DP_A	DM_A	HVDD	R1_A	LVDD	HDA0	NC	A
XO	VSS	TRST	VSS	HVDD	VSS	VBUSEN_A	VSS	VSS	HDA2	XHPDIAG	B
LVDD	VSS	TDI	TCK	TEST	XHCS0	VBUSFLG_A	VSS	XHCS1	HDA1	HINTRQ	C
R1_B	VSS	TDO	XHDASP	HVDD	XHDMACK	HIORDY	XHIOW	XHIOR	HDD0	HDMARQ	D
HVDD	TMS	VSS	LVDD	VSS	HDD14	HDD15	HDD12	VSS	HDD2	HDD13	E
DM_B	VSS	VSS	CA2	VSS	LVDD	HDD3	VSS	HDD1	VSS	HVDD	F
DP_B	HVDD	VBUS_B	CA3	XINT	XDACK1	HVDD	HDD11	HDD5	HDD10	HDD4	G
LVDD	VSS	CVDD	CA4	XDACK0	CD3	CD6	CVDD	CD13	HDD8	HDD9	H
LVDD	XRESET	CA1	XBEL	XDREQ1	CD0	CD4	CD7	CD10	HDD6	HDD7	J
CA8	XCS	CA5	CA6	CA7	CD1	CD5	CD9	CD12	CD14	XHRESET	K
NC	XRD	XWRH	XWRL	XDREQ0	CD2	CVDD	CD8	CD11	CD15	NC	L
1	2	3	4	5	6	7	8	9	10	11	

Fig. 4.2 Pin Layout Diagram of the BGA package

5. Pin Description

OSC

Pin	Ball	Name	I/O	RESET	Pin Type	Pin Description
128	A2	XI	IN	-	Analog	Input for the internal oscillator circuit 12 MHz/24 MHz
1	B1	XO	OUT	-	Analog	Output for the internal oscillator circuit

TEST

Pin	Ball	Name	I/O	RESET	Pin Type	Pin Description
120	C5	TEST	IN	-		Test pin (fixed low)
121	D3	TDO	OUT	Hi-Z	2mA	JTAG TDO pin
122	C4	TCK	IN	-		JTAG TCK pin
124	E2	TMS	IN	-		JTAG TMS pin
125	C3	TDI	IN	-		JTAG TDI pin
126	B3	TRST	IN	-		JTAG TRST pin

If the JTAG function is to be left unused, then process pins TEST, TICK, TMS, TDI, TRST each as LOW and pin TDO as OPEN.

PD: Pull Down, PU: Pull Up

USB

Pin	Ball	Name	I/O	RESET	Pin Type	Pin Description
111	A8	R1_A	IN	-	Analog	Internal operation setup pin 6.2 kΩ±1% resistor connected between this pin and VSS
116	A5	DP_A	BI	Hi-Z	Analog	USB host data line, Data+
114	A6	DM_A	BI	Hi-Z	Analog	USB host data line, Data–
107	C7	VBUSFLG_A	IN	-	PU Schmitt	USB power switch fault detection signal 1: Normal; 0: Erratic
108	B7	VBUSEN_A	OUT	Lo	2mA	USB power switch control signal
5	D1	R1_B	IN	-	Analog	Internal operation setup pin 6.2 kΩ±11% resistor connected between this pin and VSS
11	G1	DP_B	BI	Hi-Z	Analog	USB device data line, Data+
9	F1	DM_B	BI	Hi-Z	Analog	USB device data line, Data–
13	G3	VBUS_B	IN	(PD)	(PD)	USB device bus detection signal

PD: Pull Down

PU: Pull Up

CPU I/F

Pin	Ball	Name	I/O	RESET	Pin Type	Pin Description		
					Bus Mode ⇒	16bit Strobe mode	16bit BE mode	8bit mode
20	J2	XRESET	IN	-	-	Reset signal		
31	L2	XRD	IN	-	-	Read strobe		
33	L4	XWRL (XWR)	IN	-	-	Write strobe, lower	Write strobe	
32	L3	XWRH (XBEH)	IN	-	-	Write strobe, upper	High byte enable	Fixed high
30	K2	XCS	IN	-	-	Chip select signal		
37	G5	XINT	OUT	High	2mA (Tri-state)	Interrupt output signal		
38	L5	XDREQ0	OUT	High	2mA	DMA0 request		
39	H5	XDACK0	IN	-	-	DMA0 acknowledge		
40	J5	XDREQ1	OUT	High	2mA	DMA1 request		
41	G6	XDACK1	IN	-	-	DMA1 acknowledge		
21	J4	XBEL (CA0)	IN	-	-	Fixed high or low	Low byte enable	Address 0
22	J3	CA1	IN	-	-	CPU bus address		
23	F4	CA2	IN	-	-			
24	G4	CA3	IN	-	-			
25	H4	CA4	IN	-	-			
26	K3	CA5	IN	-	-			
27	K4	CA6	IN	-	-			
28	K5	CA7	IN	-	-			
29	K1	CA8	IN	-	-			
44	J6	CD0	BI	Hi-Z	2mA	CPU data bus		CPU data bus
45	K6	CD1	BI	Hi-Z	2mA			
46	L6	CD2	BI	Hi-Z	2mA			
47	H6	CD3	BI	Hi-Z	2mA			
48	J7	CD4	BI	Hi-Z	2mA			
51	K7	CD5	BI	Hi-Z	2mA			
52	H7	CD6	BI	Hi-Z	2mA			
53	J8	CD7	BI	Hi-Z	2mA			
55	L8	CD8	BI	Hi-Z	2mA			Pull Up or Pull Down
56	K8	CD9	BI	Hi-Z	2mA			
57	J9	CD10	BI	Hi-Z	2mA			
58	L9	CD11	BI	Hi-Z	2mA			
59	K9	CD12	BI	Hi-Z	2mA			

60	H9	CD13	BI	Hi-Z	2mA		
61	K10	CD14	BI	Hi-Z	2mA		
62	L10	CD15	BI	Hi-Z	2mA		

PD: Pull Down

PU: Pull Up

IDE I/F

Pin	Ball	Name	I/O	RESET	Pin Type	Pin Description
103	B10	HDA2	OUT	Hi-Z	4mA	IDE register address
97	C10	HDA1	OUT	Hi-Z	4mA	
99	A10	HDA0	OUT	Hi-Z	4mA	
105	C9	XHCS1	OUT	Hi-Z	4mA	Chip select for accessing control registers
104	C6	XHCS0	OUT	Hi-Z	4mA	Chip select for accessing command block registers
93	D9	XHIOR	OUT	Hi-Z	4mA	IDE read strobe
92	D8	XHIOW	OUT	Hi-Z	4mA	IDE write strobe
91	D11	HDMARQ	IN	(PD)	(PD)	DMA transfer request
95	D6	XHDMACK	OUT	Hi-Z	4mA	DMA transfer enable
94	D7	HIORDY	IN	(PU)	(PU)	IDE register ready signal
96	C11	HINTRQ	IN	(PD)	(PD)	IDE interrupt request
68	K11	XHRESET	OUT	Hi-Z	4mA	IDE bus reset
106	D4	XHDASP	IN	(PU)	(PU)	Driver enable/slave drive available
98	B11	XHPDIAG	IN	(PU)	(PU)	Diagnostic sequence end signal
90	E7	HDD15	BI	Hi-Z	4mA(PU)	IDE data bus
87	E6	HDD14	BI	Hi-Z	4mA(PU)	
84	E11	HDD13	BI	Hi-Z	4mA(PU)	
82	E8	HDD12	BI	Hi-Z	4mA(PU)	
77	G8	HDD11	BI	Hi-Z	4mA(PU)	
75	G10	HDD10	BI	Hi-Z	4mA(PU)	
72	H11	HDD9	BI	Hi-Z	4mA(PU)	
70	H10	HDD8	BI	Hi-Z	4mA(PU)	
69	J11	HDD7	BI	(PD)	4mA(PD)	
71	J10	HDD6	BI	Hi-Z	4mA(PU)	
74	G9	HDD5	BI	Hi-Z	4mA(PU)	
76	G11	HDD4	BI	Hi-Z	4mA(PU)	
81	F7	HDD3	BI	Hi-Z	4mA(PU)	
83	E10	HDD2	BI	Hi-Z	4mA(PU)	
86	F9	HDD1	BI	Hi-Z	4mA(PU)	
88	D10	HDD0	BI	Hi-Z	4mA(PU)	

PU and PD can be turned on or off by setting the appropriate register.

PD: Pull Down

PU: Pull Up

Note: All IDE I/F pins are 5 V tolerant.

POWER

Pin	Ball	Name	Voltage	Pin Description
8, 12, 65, 80, 89, 102 113, 117, 123	G7, D5, F11, E1, G2, B5, A7	HVDD	3.3V	Power supply for the IDE, USB, and TEST I/O
14, 17, 42, 63, 66, 78, 100, 118, 127, 3, 109	J1, E4, F6, H1, A3, A4, C1, A9	LVDD	1.8V	Internal power supply and power supply for the OSC I/O
19, 35, 49, 64	H3, L7, H8	CVDD	3.3-1.8V	Power supply for the CPU I/O
2, 6, 7, 10, 15, 18, 34, 36, 43, 50, 54, 67, 73, 79, 85, 101, 112, 115, 119, 4, 110	F3, E3, E5, F5, C8, F8, E9, F10, H2, F2, B2, B4, B6, B8, D2, C2, B9	VSS	0V	GND
16	A1, L1, A11, L11	N.C.	0V	N.C. (Connect this pin to GND)

6. Functional Description

This section describes the operation of the LSI.

In the explanation below, the registers are described according to the following naming conventions.

- Names indicating a register comprising one address
  Register name + register
  Example: "MainIntStat register"
- Names indicating individual register bits
  Register name.bit name + bit, or bit name + bit
  Example: "MainIntStat.CPU_IntStat bit"
- Registers provided for each endpoint
  Described as D_EPx{x=0, a-c}register, etc.
  Example: "D_EPx{x=0, a-c}IntStat register"
- Registers provided for each channel
  Described as H_CHx{x=0, a-e}register, etc.
  Example: "H_CHx{x=0, a-c}IntStat register"
- Registers provided for each DMA channel
  Described as DMAx{x=0, 1}register, etc.
  Example: "DMAx{x=0, 1}Config register"

6.1 Selection of Device/Host Register Maps

The S1R72V05 LSI permits either a device or a host register map to be selected when using USB.

When a device register map is selected (hereinafter referred to as "device mode"), the register bits of both device/host shared registers and device registers are enabled.

When a host register map is selected (hereinafter referred to as "host mode"), the register bits of both device/host shared registers and host registers are enabled.

6.1.1 Method for Selecting a Register Map

A device or a host register map is selected by setting the appropriate register bit.

Either register map can be selected in all of Sleep, Snooze, Active60, ActDevice, and ActHost states (see the section on Power Management). Table 6.1 shows the item to be set when selecting a device/host register map.

Table 6.1 Setup Item for Selecting Device/Host Mode

Item	Register/Bit	Description
Device/host register map selection	HostDeviceSel. HOSTxDEVICE	Selects either a device or a host register map, so that the selected registers can be used.

6.1.2 Port State Change Detection Status

The S1R72V05 LSI has the function to detect port status.

This function can be used in all of Sleep, Snooze, Active60, ActDevice, and ActHost states (see the section on Power Management). Since this function can be used in either device or host mode, it allows to detect a change of the state of the Downstream port (host port) during device mode or that of the Upstream port (device port) during host mode.

A change of the host port or device port state can be used as the right timing for mode selection.

Table 6.2 shows the set items and statuses of port state change detection.

6.1.2.1 Example Usage of Port State Change Detection Status

The following shows an example usage of VBUS_B pin change status and signal line change status.

6.1.2.1.1 VBUS_B Pin Change Status

This status indicates that VBUS_B pin of the device port has changed state.

Table 6.2 shows the register associated with the VBUS_B pin change status.

Table 6.2 Resisters Associated with VBUS_B Pin Change Status

Item	Register/Bit	Description
VBUS_B pin change status	DeviceIntStat. VBUS_Changed	Indicates that the VBUS_B pin of the device port has changed state.
VBUS_B pin change status enable	DeviceIntEnb. EnVBUS_Changed	Enable/disables assertion of the MainIntStat.DeviceIntStat bit by the DeviceIntStat.VBUS_Changed bit.
Device port VBUS_B state	D_USB_Status. VBUS	Indicates the VBUS_B pin state of the device port.

When using the VBUS_B pin change status, the firmware should perform processing (1) , (2), and (4) to (7).

(1) Clear the VBUS_B pin change status.

(2) Set to VBUS_B pin change status enable.

(3) When the host port has a device connected, the a VBUS_B pin change status is issued.

(4) Check the VBUS_B pin change status.

(5) Clear the VBUS_B pin change status.

(6) Clear the VBUS_B pin change status enable.

(7) Check the device port VBUS_B status. If the device port VBUS_B status is "1", determine that the device port has the VBUS (host or hub) connected to it.

6.1.2.1.2 Signal Line Change Status

This status indicates that the DP and DM pins of the host port have changed state from SE0.

Table 6.3 shows the registers associated with the signal line change status.

Table 6.3 Registers Associated with Signal Line Change Status

Item	Register/Bit	Description
Signal line change status	HostIntStat. LineStateChanged	Indicate that the state of DP and/or DM pin of the host port has changed from SE0.
Signal line change status enable	HostIntEnb. EnLineStateChanged	Enable/disables assertion of the MainIntStat.HostIntStat bit by the HostIntStat.LineStateChanged bit.
VBUS error detection status	HostIntStat. VBUS_Err	Indicates that an error occurred in VBUS.
VBUS error detection status enable	HostIntEnb. EnVBUS_Err	Enable/disables assertion of the MainIntStat.HostIntStat bit by the HostIntStat.VBUS_Err bit.
Operation mode	H_XcvrControl. OpMode[1:0]	Sets operation mode of the HTM.
VBUS enable	H_USB_Control. VBUS_Enb	Enables an external USB power switch.
Host port VBUS state	H_USB_Status. VBUS_State	Indicates the VBUS state of the host port (normal or erratic).
Host port signal line state	H_USB_Status. LineState[1:0]	Indicates the signal state of USB cable.
Host state transition execution	H_NegoControl_1. AutoMode[3:0]	Sets the host state that is required of a transition.

Unlike the connection detection status normally used while the LSI is operating as a USB host, this status is provided as a special status to detect a change of the host port signal line while the USB host function is not in use, i.e., during Sleep, Snooze, Active60, or ActDevice state (see the section on Power Management). This status is asserted frequently if the HostIntEnb. EnLineStateChanged is enabled while the LSI is operating as a USB host. Therefore, use of this status should be limited to detection of change in the signal line state here, and it should be disabled when the USB host is operating.

Furthermore, if a VBUS error detection status is detected while the USB host is not operating, immediately turn the VBUS enable off to stop VBUS drive.

When using the signal line change status, the firmware should perform processing (1) to (4) and (6) to (9).

(1) Clear the signal line change status and VBUS error detection status.

(2) Set the signal line change status enable and VBUS error detection status enable.

(3) Set H_XcvrControl register to 0x93.

(4) Turn the VBUS enable on.

(5) When the host port has a device connected, the line state changes from SE0 and a signal line change status is issued.

(6) Check the signal line change status.

(7) Clear the signal line change status.

(8) Clear the signal line change status enable.

(9) Check the line state of host port. If the line state is “01” or “10”, determine that the host port has a device connected to it.

When host mode is selected after that, the H_XcvrControl register and the VBUS enable retain the values that have been set here. Once the firmware sets appropriate code in the host state transition execution register, these settings are automatically performed in hardware according to the set host state. For details, refer to the relevant section on host state management support function.

6.2 USB Device Control

The following describes the USB device functions.

6.2.1 Endpoints

The LSI stipulated herein has the endpoint for control transfer (EP0) and three general-purpose endpoints (EPa, EPb, and EPc). The endpoints EPa, EPb, and EPc can respectively be used as the endpoint for bulk or interrupt transfer.

The hardware of the LSI provides endpoints for the purpose of transaction management. However, it does not provide management functions for the interfaces defined in the USB standard (hereinafter referred to as the "USB-defined interface"). The USB-defined interface should be implemented by the user firmware. Set up and combine endpoints as appropriate for the descriptor definitions specific to the device, to configure the USB-defined interface.

Each endpoint has fixed basic setup items determined by the USB-defined interface and the variable control items and status to be controlled for each transfer. The basic setup items should be set up when initializing the chip or switching USB-defined interfaces from one to another.

Table 6.4 lists the basic setup items for the endpoint EP0 (default control pipe).

The endpoint EP0 shares the register set and FIFO area for transfers in IN and OUT directions. The direction of data transaction should be set by the firmware as appropriate for the execution of the data and status stages at the endpoint EP0.

Table 6.4 Basic Setup Items of the Endpoint EP0

Item	Register/Bit	Description
Max. packet size	D_EP0MaxSize	Sets the Max. packet size to 8, 16, 32, or 64 during FS operation, or to 64 during HS operation. The endpoint EP0 has allocated to it a 64-byte area beginning with the FIFO address 0.

Table 6.5 lists the basic setup items for the general-purpose endpoints (EPa, EPb, and EPc). Since the endpoints EPa, EPb, and EPc permit the transaction direction and endpoint number to be set as desired, up to three independent endpoints can be used. Set up endpoints as appropriate for the contents of definitions of the USB-defined interface and enable the set endpoints as necessary, to configure the USB-defined interface.

The FIFO areas for the endpoints EPa, EPb, and EPc are set by the start and the end address of each area.

Table 6.5 Basic Setup Items of the General.purpose Endpoints

Item	Register/Bit	Description
Transaction direction	D_EPx{x=a-c}Config_0.INxOUT	Sets the direction of transfer at each endpoint.
Max. packet size	D_EPx{x=b-c}MaxSize_H, D_EPx{x=a-c}MaxSize_L	Sets the Max. packet size for each endpoint to 8, 16, 32, 64, or 512 bytes. For the endpoints at which bulk transfer is to be performed, however, set the Max. packet size to 8, 16, 32, or 64 bytes during FS mode, or to 512 bytes during HS mode. (Note: The Max. packet size that can be set for EPa is limited to 64 bytes.)
Endpoint number	D_EPx{x=a-c}Config_0.EndpointNumber	Sets the endpoint number for each endpoint to any value between 0x1 to 0xF.
Toggle mode	D_EPx{x=a-c}Config_0.IntEP_Mode	Sets the operation mode of interrupt transfer. For the endpoints at which bulk transfer is to be performed, always set this register bit to 0 irrespective of the transaction direction. For the IN direction endpoints, set the mode of toggle sequence. For the OUT direction endpoints, if interrupt transfer is to be performed, always set this register bit to 0.
Endpoint enable	D_EPx{x=a-c}Config_0.EnEndpoint	Enables each endpoint. Make this setting for any endpoint when the USB-defined interface that will use that endpoint is enabled.
FIFO area	D_EPx{x=a-c}StartAdrs_H, D_EPx{x=a-c}StartAdrs_L, D_EPcEndAdrs_H, D_EPcEndAdrs_L	Sets the area to be allocated to each endpoint by FIFO address. Make sure the allocated FIFO area is equal to or greater than the Max. packet size on each channel. The size of the FIFO area affects the throughput of data transfers. For details on the allocation of FIFO areas, refer to the relevant section in Chapter 6 on FIFOs.

6.2.2 Transactions

The LSI provides transaction execution functions in hardware and provides the firmware with the interfaces necessary to execute transactions. The interfaces for the firmware are implemented as control and status registers and the interrupt signals that are asserted by a status. For details on setting interrupt assertion by status, refer to the relevant section on registers.

The LSI issues a status to the firmware for each transaction performed. However, the firmware does not always need to manage each individual transaction. When responding to a transaction request, the LSI inspects the FIFO to find its data quantity or free space to determine whether a data transfer can be performed, and then performs the transaction automatically.

For an OUT endpoint, for example, the firmware can read data out of the FIFO through the CPU interface (DMA read or register read) or the IDE interface (IDE write) to create a free space in the FIFO, thereby allowing OUT transactions to be automatically executed in succession. For an IN endpoint also, the firmware can write data to the FIFO through the CPU interface (DMA write or register write) or the IDE interface (IDE read) to create valid data in the FIFO, thereby allowing IN transactions to be automatically executed in succession.

Table 6.6 lists the control items and status relating to the transaction control for the endpoint EP0.

Table 6.6 Endpoint EP0 Control Items and Status

Item	Register/Bit	Description
Transaction direction	D_EP0Control.INxOUT	Sets the direction of transfer in the data and status stages.
Descriptor reply enable	D_EP0Control.ReplyDescriptor	Invokes automatic descriptor response.
Descriptor reply address	D_DescAdrs_H, DescAdrs_L	Specifies the start address in FIFO of the data to be returned by an automatic descriptor response.
Descriptor size	D_DescSize_H, DescSize_L	Specifies the data quantity to be returned by an automatic descriptor response.
Control protect	D_SETUP_Control.ProtectEP0	When this bit is set, the ForceNAK and ForceSTALL bits in the EP0ControlIN and EP0ControlOUT registers are protected against access. This bit is set in hardware by the LSI when a RcvEP0SETUP status is flagged, and can be cleared by a register access by the CPU.
Short packet transmit enable	D_EP0ControlIN.EnShortPkt	Enables transmission of short packets less than Max. packet size. This bit is cleared when the IN transaction that transmitted a short packet is completed.
Toggle sequence bit	D_EP0ControlIN.ToggleStat, D_EP0ControlOUT.ToggleStat	Indicates the status of the toggle sequence bits. These bits are automatically initialized by a SETUP stage.
Toggle set	D_EP0ControlIN.ToggleSet, D_EP0ControlOUT.ToggleSet	Sets the toggle sequence bits.
Toggle clear	D_EP0ControlIN.ToggleClr, D_EP0ControlOUT.ToggleClr	Clears the toggle sequence bits.

Forced NAK response	D_EP0ControlIN.ForceNAK, D_EP0ControlOUT.ForceNAK	Always responds with NAK for IN or OUT (including PING) transactions irrespective of the data quantity and free space in the FIFO.
STALL response	D_EP0ControlIN.ForceSTALL, D_EP0ControlOUT.ForceSTALL	Responds with STALL for IN or OUT (including PING) transactions.
Automatic ForceNAK set	D_EP0ControlOUT.AutoForceNAK	Sets the EP0ControlOUT.ForceNAK bit each time an OUT transaction is completed.
SETUP receive status	DeviceIntStat.RcvEP0SETUP	Indicates that a SETUP transaction has been executed.
Transaction status	D_EP0IntStat.OUT_ShortACK, D_EP0IntStat.IN_TranACK, D_EP0IntStat.OUT_TranACK, D_EP0IntStat.IN_TranNAK, D_EP0IntStat.OUT_TranNAK, D_EP0IntStat.IN_TranErr, D_EP0IntStat.OUT_TranErr	Indicates the result of a transaction.
Descriptor reply data stage end status	D_FIFO_IntStat.DescriptorCmp	Indicates that the data stage of an automatic descriptor response has ended.

Table 6.7 lists the control items and status relating to the transaction processing for the general-purpose endpoints EPa, EPb, and EPc.

Table 6.7 General.purpose Endpoint Control Items and Status

Item	Register/Bit	Description
Automatic ForceNAK set	D_EPx{x=a-c}Control.AutoForceNAK	Sets the D_EPx{x=a-c}Control.ForceNAK bit for an endpoint each time an OUT transaction at that endpoint is completed.
Short packet transmit enable	D_EPx{x=a-c}Control.EnShortPkt	Enables transmission of short packets less than Max. packet size for an IN transaction. This bit is cleared when the IN transaction that transmitted a short packet is completed.
Automatic ForceNAK set by short packet reception disable	D_EPx{x=a-c}Control. DisAF_NAK_Short	Disables the function(*) to automatically set the D_EPx{x=a-c}Control.ForceNAK bit for an endpoint when a short packet is received for that endpoint in an OUT transaction. *: This function remains enabled unless it is disabled by this bit.
Toggle sequence bit	D_EPx{x=a-c}Control.ToggleStat	Indicates the status of the toggle sequence bit.
Toggle set	D_EPx{x=a-c}Control.ToggleSet	Sets the toggle sequence bit.
Toggle clear	D_EPx{x=a-c}Control.ToggleClr	Clears the toggle sequence bit.
Forced NAK response	D_EPx{x=a-c}Control.ForceNAK	Always responds with NAK for transactions irrespective of the data quantity and free space in the FIFO.
STALL response	D_EPx{x=a-c}Control.ForceSTALL	Responds with STALL for transactions.
Transaction status	D_EPx{x=a-c}IntStat.OUT_ShortACK, D_EPx{x=a-c}IntStat.IN_TranACK, D_EPx{x=a-c}IntStat.OUT_TranACK, D_EPx{x=a-c}IntStat.IN_TranNAK, D_EPx{x=a-c}IntStat.OUT_TranNAK, D_EPx{x=a-c}IntStat.IN_TranErr, D_EPx{x=a-c}I~tStat.OUT_TranErr	Indicates the result of a transaction.

6.2.2.1 SETUP Transactions

SETUP transactions addressed to the endpoint EP0 of the local node are unconditionally executed. (The USB functions must be enabled by the D_NegoControl.ActiveUSB bit before this can occur.)

When a SETUP transaction is issued, the LSI stores the entire content of the data packet (8 bytes) in the D_EP0SETUP_0 through D_EP0SETUP_7 registers and then returns an ACK response. Furthermore, except for SetAddress() requests, the LSI issues a RcvEP0SETUP status to the firmware.

If an error occurs during the SETUP transaction, the LSI does not respond, nor does it issue a status.

When the SETUP transaction is completed, the LSI sets the ForceNAK bit and clears the ForceSTALL bit in the D_EP0ControlIN and D_EP0ControlOUT registers. It also sets the ToggleStat bit. Furthermore, it sets the D_SETUP_Control.ProtectEP0 bit. When the firmware has finished setting up the endpoint EP0 and is ready to go to the next stage, it should clear the SETUP_Control.ProtectEP0 bit and then the ForceNAK bit in the D_EP0ControlIN or D_EP0ControlOUT register for the direction concerned.

Fig. 6.1 shows how a SETUP transaction will be performed. In (a), the host issues a SETUP token addressed to the endpoint EP0 of this node. In (b), the host continues to send an 8 bytes long data packet. The LSI writes this data to the D_EP0SETUP_0 to D_EP0SETUP_7 registers. In (c), the LSI automatically returns an ACK response. Furthermore, it sets up the registers to be automatically set and issues a status to the firmware.

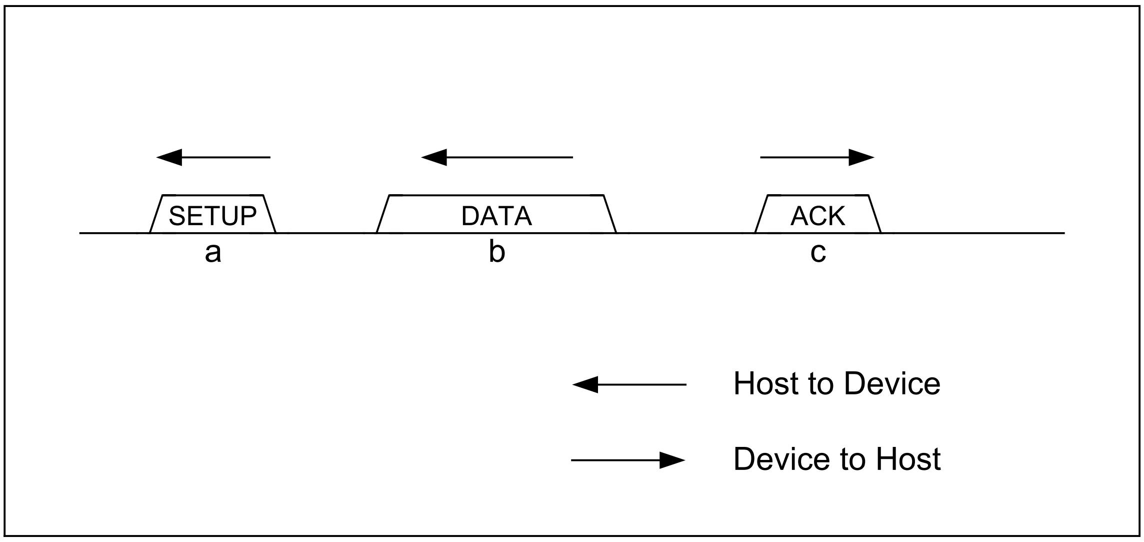


Fig. 6.1 SETUP transaction

6.2.2.2 Bulk/Interrupt OUT Transactions

In a bulk/interrupt OUT transaction, the LSI starts receiving data if the FIFO has a free space equal to or greater than Max. packet size.

When all bytes of data are received correctly in a bulk/interrupt OUT transaction, the LSI completes the transaction and returns an ACK or a NYET response. It then issues an OUT_TranACK status for the corresponding endpoint (D_EPx{x=0, a-c}IntStat.OUT_TranACK bit) to the firmware. It also updates the FIFO and assuming that data has all been received, reserves a storage area.

Furthermore, when all data bytes of a short packet have been received in a bulk/interrupt OUT transaction, the LSI issues an OUT_ShortACK status (D_EPx{x=0, a-c}IntStat.OUT_ShortACK bit), in addition to the transaction-complete processing described above. Furthermore, if the D_EPx{x=0, a-c}Control.DisAF_NAK_Short bit is cleared, the LSI sets the D_EPx{x=a-c}ForceNAK bit for the endpoint.

If a toggle mismatch occurs in a bulk/interrupt OUT transaction, the LSI responds with ACK for the transaction but does not issue a status. The FIFO is not updated.

If an error occurs in a bulk/interrupt OUT transaction, the LSI does not respond for the transaction. In this case, it issues an OUT_TranErr status (D_EPx{x=0, a-c}IntStat.OUT_TranErr bit). The FIFO is not updated.

If all bytes of data could not be received in a bulk/interrupt OUT transaction, the LSI responds with NAK for the transaction. It also issues an OUT_TranNAK status (D_EPx{x=0, a-c}IntStat.OUT_TranNAK bit). The FIFO is not updated.

Fig. 6.2 shows how a bulk or interrupt OUT transaction will be performed in cases when the transaction is completed normally. In (a), the host issues an OUT token addressed to the OUT-direction endpoint present in this node. In (b), the host continues to send a data packet within Max. packet size. The LSI writes this data to the FIFO for the corresponding endpoint. In (c), the LSI automatically returns an ACK response when it successfully received the data. It also sets up the registers to be automatically set and issues a status to the firmware.

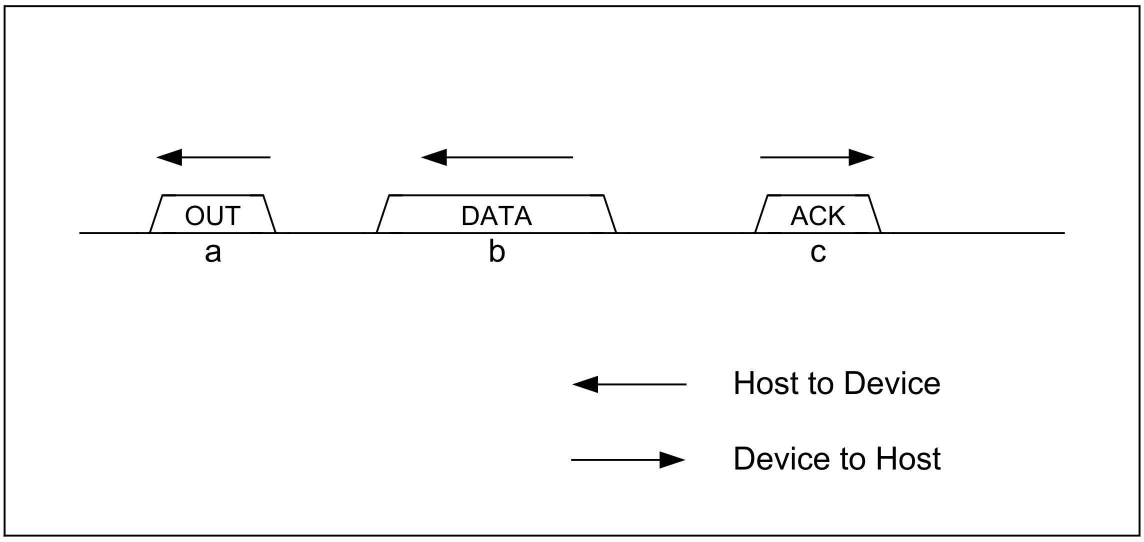


Fig. 6.2 OUT transaction

6.2.2.3 Bulk/Interrupt IN Transactions

When the FIFO for an IN-direction bulk/interrupt endpoint has a quantity of data equivalent to Max. packet size, or short packet transmission for that endpoint has been enabled by the firmware, the LSI sends back a data packet in response to the IN transaction.

Transmission of short packets (including packets with data length of 0) is enabled by setting the D_EP0ControlIN.EnShortPkt bit or D_EPx{x=a-c}Control.EnShortPkt bit. When transmitting short packets, make sure that no new data will be written to the FIFO for the endpoint concerned before the transaction is completed after packet transmission has been enabled.

For the endpoint EP0, when the IN transaction to transmit a short packet is completed, the D_EP0ControlIN.ForceNAK bit is set.

When ACK is received in the IN transaction by which data was sent back to the host, the LSI completes the transaction and issues an IN_TranACK status (D_EPx{x=0, a-c}IntStat.IN_TranACK bit) to the firmware. It also updates the FIFO and assuming the transmitted data to have been transmitted, frees the storage area.

If ACK is not received in the IN transaction by which data was sent back to the host, the LSI assumes that the transaction has failed and issues an IN_TranErr status (D_EPx{x=0, a-c}IntStat.IN_TranErr bit) to the firmware. It does not update the FIFO, nor does it free the storage area.

When the FIFO does not have a quantity of data equivalent to the Max. packet size for an IN-direction bulk/interrupt endpoint, or short packet transmission for that endpoint has not been enabled by the firmware, the LSI responds with NAK for the IN transaction and issues IN_TranNAK status (EPx{x=0, a-c}IntStat.IN_TranNAK bit) to the firmware. It does not update the FIFO, and it does not free the storage area.

Fig. 6.3 shows how a bulk or interrupt IN transaction will be performed in cases when the transaction is completed normally. In (a), the host issues an IN token addressed to the IN-direction endpoint present in this node. In (b), if the LSI can respond to this IN transaction, it transmits a data packet within Max. packet size. In (c), the host responds with ACK. When the LSI receives ACK, it sets up the registers to be automatically set and issues a status to the firmware.

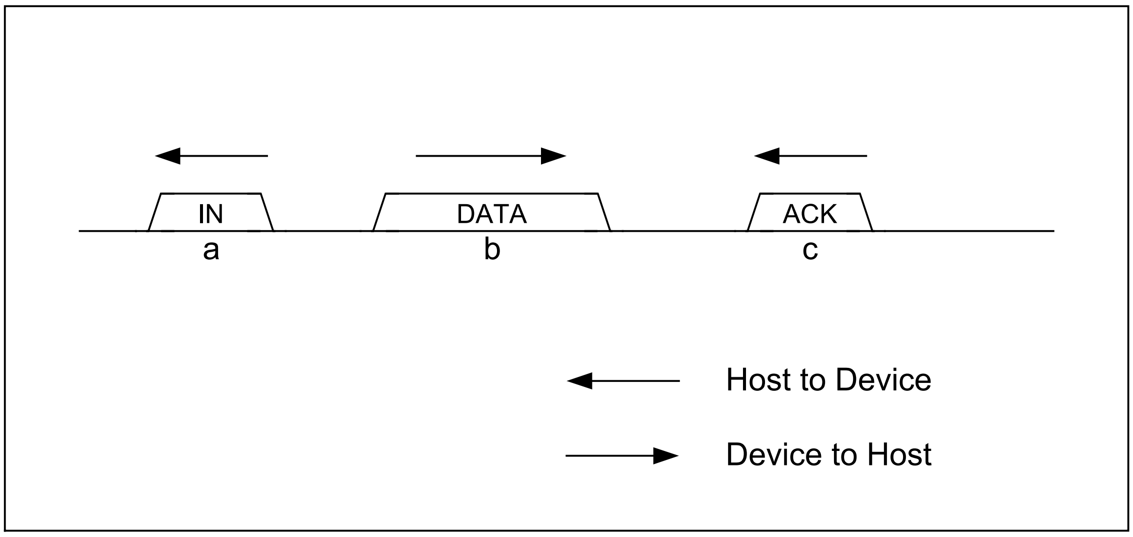


Fig. 6.3 IN transaction

6.2.2.4 PING Transactions

At bulk OUT-direction endpoints, a PING transaction is executed when operating in HS mode.

If the FIFO for the corresponding endpoint has a free space equal to or greater than Max. packet size, the LSI responds with ACK for the PING transaction. It does not issue a status to the firmware, however.

If the FIFO for the corresponding endpoint has a free space less than Max. packet size, the LSI responds with NAK for the PING transaction. It also issues an OUT_TranNAK status (D_EPx{x=0, a-c}IntStat.OUT_TranNAK bit) to the firmware.

In no case will the FIFO be updated in PING transactions.

Fig. 6.4 shows how a PING transmission will be performed when responded with ACK. In (a), the host issues a PING token addressed to the OUT-direction endpoint present in this node. In (b), if the FIFO has a free space equivalent to Max. packet size, the LSI responds with ACK for the PING transaction. It also issues a status to the firmware.

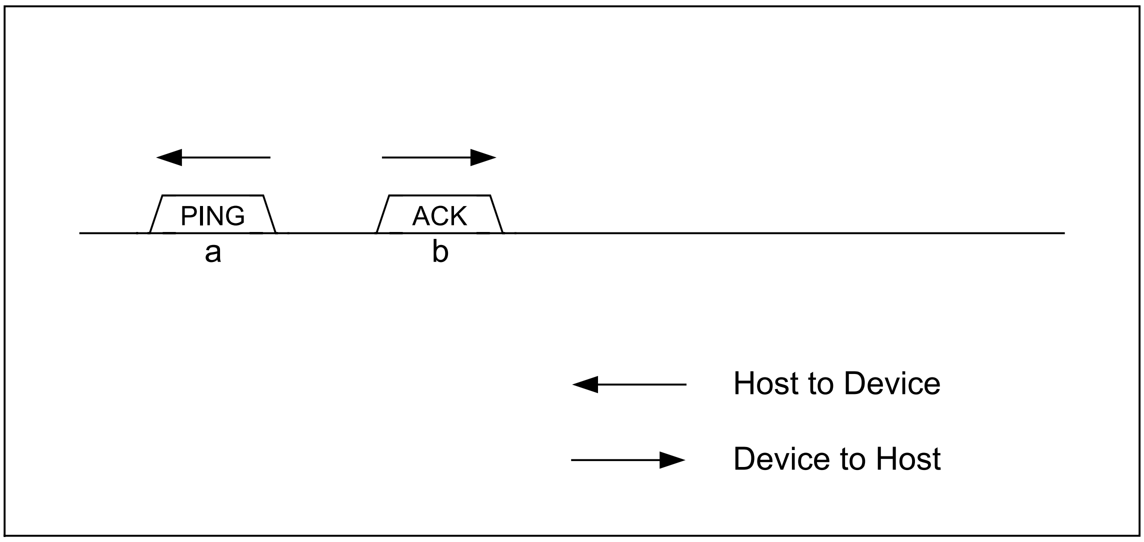


Fig. 6.4 PING transaction

6.2.3 Control Transfers

Control transfers at the endpoint EP0, except for SetAddress() requests, are controlled as a combination of individual transactions. SetAddress() requests are automatically processed using the automatic address setup function that will be described later.

Fig. 6.5 shows how a control transfer will be performed in cases when the data stage is directed for OUT. In (a), the host starts a control transfer via a SETUP transaction. The firmware of the device analyzes the content of the request to get prepared for responding to a data stage. In (b), the host issues an OUT transaction to execute a data stage, and the device receives data. In (c), the host issues an IN transaction to execute a status stage, and the device sends a packet in data length of zero back to the host.

For control transfers without data stages, the operation is executed without performing the data stage described in this example.

Transition to the status stage is accomplished by issuing a transaction for the direction opposite to the data stage from the host. The firmware monitors the IN_TranNAK status (D_EP0IntStat.IN_TranNAK bit) to seize a chance for transition from the data stage to the status stage.

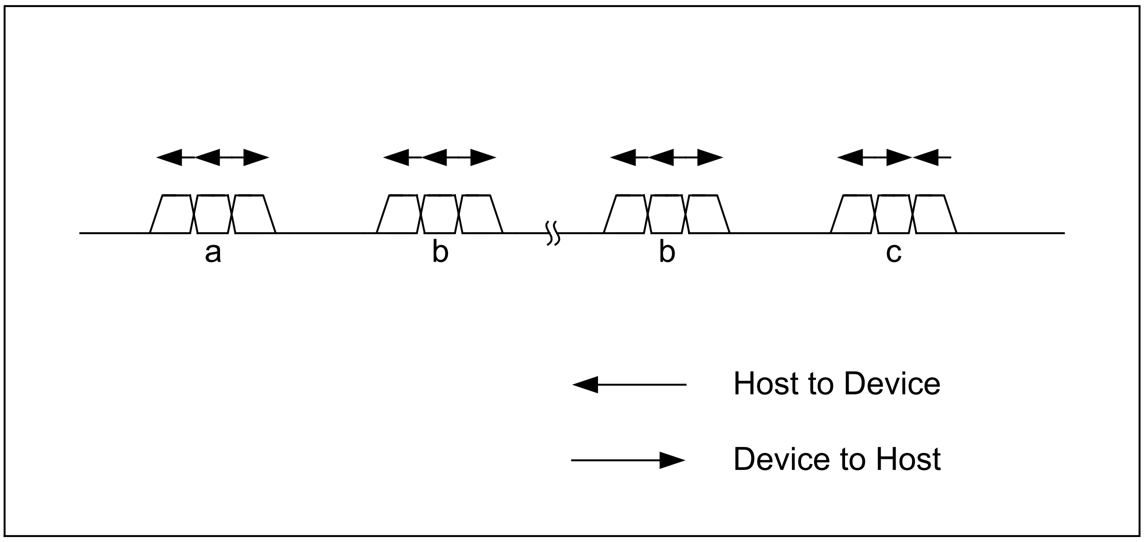


Fig. 6.5 Control transfer when the data stage is directed for OUT

Fig. 6.6 shows how a control transfer will be performed when the data stage is directed for IN. In (a), the host starts a control transfer by means of a SETUP transaction. The firmware of the device analyzes the content of the request to get ready to respond to a data stage. In (b), the host issues an IN transaction to execute a data stage, and the device transmits data. In (c), the host issues an OUT transaction to execute a status stage, and the device responds to it with ACK.

Transition to the status stage is accomplished by issuing a transaction for the direction opposite to the data stage from the host. The firmware monitors the OUT_TranNAK status (D_EP0IntStat.OUT_TranNAK bit) to seize a chance for transition from the data stage to the status stage.

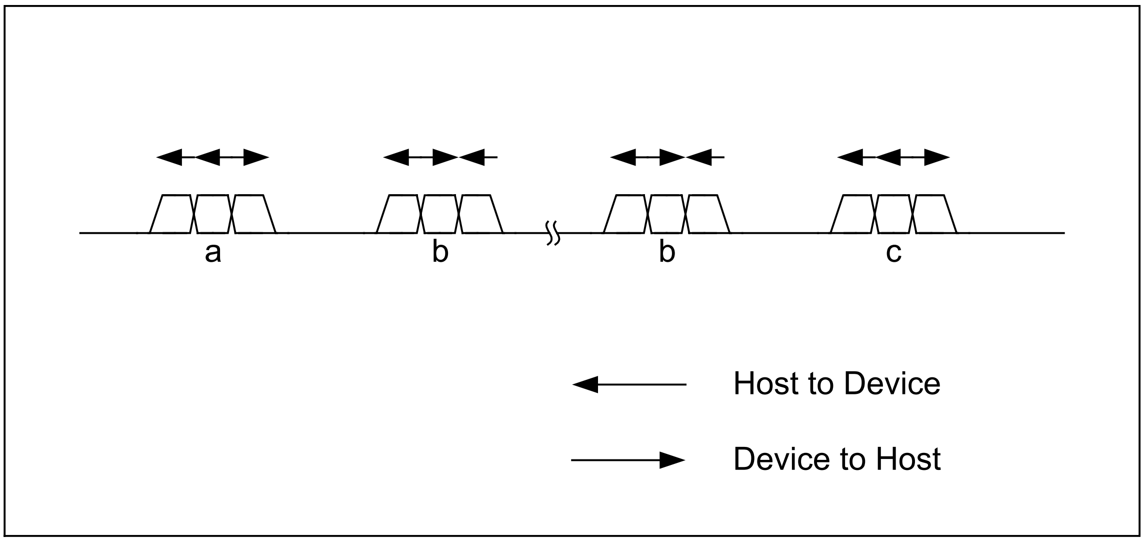


Fig. 6.6 Control transfer when the data stage is directed for IN

For the data and status stages of control transfers, a flow control by NAK is enabled, because ordinary OUT and IN transactions are performed in those stages. The device is allowed to get ready to respond within a designated time.

6.2.3.1 Setup Stage

When a SETUP token addressed to the local node is received, the LSI automatically executes a setup transaction.

The firmware monitors the RcvEP0SETUP status and analyze the request by reading out the D_EP0SETUP_0 through D_EP0SETUP_7 registers to control the control transfer.

If the received request is for a control transfer with an OUT-direction data stage involved, clear the INxOUT bit in the D_EP0Control register to direct the endpoint EP0 for OUT to permit a transition to the data stage.

If the received request is for a control transfer with an IN-direction data stage involved, set the INxOUT bit in the D_EP0Control register to direct the endpoint EP0 to permit a transition to the data stage.

If the received request is for a control transfer without a data stage involved, set the INxOUT bit in the D_EP0Control register to direct the endpoint EP0 for IN to permit a transition to the status stage.

6.2.3.2 Data Stage and Status Stage

Go to the next stage according to the content of a request analyzed by reading out the D_EP0SETUP_0 through D_EP0SETUP_7 registers.

If that stage is for the OUT direction, clear the INxOUT bit in the D_EP0Control register to direct it for OUT, and then set up the D_EP0ControlOUT register as appropriate to control the stage. By the time when the SETUP stage has finished, the ForceNAK bit must be set. Similarly, the D_SETUP_Control.ProtectEP0 bit must be set also.

If that stage is for the IN direction, set the INxOUT bit in the D_EP0Control register to direct it for IN, and then set up the D_EP0ControlIN register as appropriate to control the stage. By the time when the SETUP stage has finished, the ForceNAK bit must be set. Similarly, the D_SETUP_Control.ProtectEP0 bit must be set also.

6.2.3.3 Automatic Address Setup Function

The LSI stipulated herein has a function to automate the processing of SetAddress() requests in control transfers at the endpoint EP0.

The LSI checks the content of a request by reading out the D_EP0SETUP_0 through D_EP0SETUP_7 registers in hardware. If the request is found to be a valid SetAddress() request, the LSI shifts to processing of the status stage for that request without notifying the firmware. When the status stage is completed, the LSI sets the relevant address in the USB_Address register and issues a SetAddressCmp status (D_SIE_IntStat.SetAddressCmp bit) to the firmware.

The firmware monitors the SetAddressCmp status, so that when the status is issued, it can confirm the address by reading out the USB_Address register.

6.2.3.4 Descriptor Reply Function

The LSI stipulated herein has a descriptor reply function which is effective for requests in control transfers at the endpoint EP0 such as GetDescriptor() that requests data that has been issued a number of times.

For requests where the data stage is for IN transfers, the firmware can make use of this function.

Before starting a response to the data stage by clearing the D_EP0ControlIN.ForceNAK bit, set the start address of the internal data of the FIFO’s descriptor area to be returned in the D_DescAdrs_H and _L registers or the total number of bytes of the data to be returned in the D_DescSize_H and _L registers, and then set the D_EP0Control.ReplyDescriptor bit.

The descriptor reply function executes an IN transaction by sending back data packets in response to the IN transaction of the data stage until the set bytes of data have all been transmitted. If an IN transaction is issued after the set bytes of data have all been transmitted, the function responds to it with NAK. If odd data less than Max. packet size exits, the descriptor reply function sets the D_EP0ControlIN.EnShortPkt bit to allow for the IN transaction to be responded until all bytes of data are sent back.

When transition to the status stage is detected by receiving an OUT token, the function clears the D_EP0Control.ReplyDescriptor bit and issues a DescriptorCmp status (D_FIFO_IntStat.DescriptorCmp bit) to the firmware. When a DescriptorCmp status is detected, the firmware should execute the status stage.

For details about the descriptor area, refer to the relevant section in Chapter 6 on FIFOs.

6.2.4 Bulk Transfer and Interrupt Transfer

Bulk and interrupt transfers at the general-purpose endpoints EPa, EPb, and EPc can be controlled as a data flow (see 6.2.5) or as successive individual transactions (see 6.2.2).

6.2.5 Data Flow

The following describes general data flow control of OUT and IN transfers.

6.2.5.1 OUT Transfer

The data received by an OUT transfer is written into the FIFO for each corresponding endpoint. To read data out of the FIFO, there are several methods available: a register read through the CPU interface, a DMA read through the CPU interface, or a write transfer to the IDE.

To read data from the FIFO via a register read through the CPU interface, select only one endpoint using the D_EPx{x=0, a-c}Join.JoinCPU_Rd bit. The FIFO for the selected endpoint can be read out in the order the data was received by using the FIFO_Rd or FIFO_ByteRd register. The bytes of data that can be read out of the FIFO can be determined from the FIFO_RdRemain_H and FIFO_RdRemain_L registers. Since empty FIFOs cannot be read out, be sure to check the FIFO_RdRemain_H and FIFO_RdRemain_L registers to determine the bytes of data in the FIFO, and make sure those bytes of data will not be exceeded when data is read from the FIFO.

To read data from the FIFO by means of a DMA read through the CPU interface, select only one endpoint for each DMA channel using the D_EPx{x=0, a-c}Join.JoinDMAx{x=0,1} bit and set the DMAx{x=0,1}_Control.Dir bit to 1. The FIFO for the selected endpoint can be read out in the order the data was received by executing a DMA procedure in the CPU interface. The remaining bytes of data in the FIFO can be determined from the DMAx{x=0,1}_Remain_H and DMAx{x=0,1}_Remain_L registers. When the FIFO is emptied, the CPU interface automatically causes the DMA to pause for flow control.

To read data from the FIFO via a write transfer through the IDE interface, select only one OUT endpoint using the D_EPx{x=0, a-c}Join.JoinIDE bit and set the IDE_Control.Dir bit to 1. The FIFO for the selected endpoint can be read out in the order the data was received by executing an IDE transfer by IDE_Control.IDE_Go. When the FIFO is emptied, the IDE interface automatically causes the write transfer to pause for flow control.

If the FIFO has a sufficient free space to receive data packets, data can be received by automatically responding to an OUT transaction. Therefore, OUT transfers can be performed without the need for control of individual transactions by the firmware. However, if short packets (including packets in data length of zero) are received while the D_EPx{x=a-c}Control.DisAF_NAK_Short bit is cleared (default), the D_EPx{x=a-c}Control.ForceNAK bit for the corresponding endpoint is set. Therefore, when you have prepared for the next data transfer, be sure to clear the D_EPx{x=a-c}Control.ForceNAK bit.

6.2.5.2 IN Transfer

Write the data to be sent by an IN transfer into the FIFO for each corresponding endpoint. To write data into the FIFO, there are several methods available: a register write through the CPU interface, a DMA write through the CPU interface, or a read transfer from the IDE.

To write data into the FIFO via a register write through the CPU interface, select only one endpoint using the D_EPx{x=0, a-c}Join.JoinCPU_Wr bit. The FIFO for the selected endpoint can be written to by using the FIFO_Wr register, and the data is transmitted in packets in the order written. The amount of free space in the FIFO can be determined by inspecting the FIFO_WrRemain_H and _L registers. Full FIFOs cannot be written to. Always be sure to check the FIFO_WrRemain_H and _L registers to know the free bytes in the FIFO, and make sure those bytes will not be exceeded when data is written to the FIFO.

To write data into the FIFO via a DMA write through the CPU interface, select only one endpoint for each DMA channel by using the D_EPx{x=0, a-c}Join.JoinDMAx{x=0,1} bit. The FIFO for the selected endpoint can be written to by executing a DMA procedure in the CPU interface, and the data is transmitted in packets in the order written. When the FIFO is full, the CPU interface automatically causes the DMA to pause for flow control.

To write data into the FIFO via a read transfer through the IDE interface, select only one endpoint by using the D_EPx{x=0, a-c}Join.JoinIDE bit and set the IDE_Control.Dir bit to 0. The FIFO for the selected endpoint can be written to in the order read from the IDE by executing an IDE transfer by IDE_Control.IDE_Go, and the data is transmitted in packets in the order written. When the FIFO is full, the IDE interface automatically causes the read transfer to pause for flow control.

If the FIFO contains data equal to or larger than Max. packet size, the data can be transmitted by automatically responding to an IN transaction. Therefore, IN transfers can be performed without the need for control of individual transactions by the firmware. However, if a short packet needs to be transmitted at the end of data transfer, set the EnShortPkt bit. This bit is cleared when the IN transaction that transmitted the short packet is completed. The bit can be set at the time the LSI has finished writing data to the FIFO. Furthermore, if the FIFO contains odd data smaller than Max. packet size when a DMA write in the CPU interface has finished while the DMAx{x=0,1}_FIFO_Control.AutoEnShort bit is being set, the EnShortPkt bit for the corresponding endpoint is automatically set.

6.2.6 Bulk Only Support

The LSI stipulated herein has a bulk-only support function which in bulk transfers at the endpoints EPa, EPb, and EPc, provides support for Command Block Wrapper (CBW) receptions and Command Status Wrapper (CSW) transmissions specific to the USB Mass Storage Class (BulkOnly Transport Protocol).

Setting the BulkOnlyConfig.EPx{x=b,c}BulkOnly bit enables the bulk-only support function for the target endpoint.

While CBW support or CSW support of the bulk-only support function is being executed, the LSI uses the area reserved as the CBW or CSW area, and not the FIFOs normally reserved for endpoints, as it performs packet reception (CBW) or transmission (CSW).

6.2.6.1 CBW Support

The firmware can use CBW support when it performs a command transport of the BulkOnly Transport Protocol. When the BulkOnlyConfig.EPx{x=b,c}BulkOnly bit is set, CBW support for the corresponding OUT endpoint is enabled. Control should be exercised in such a way that CBW support is enabled for only one endpoint at a time. Setting the BulkOnlyControl.GoCBW_Mode bit while CBW support is active causes CBW support to be executed, so that the data received in an OUT transaction at the target endpoint is handled as CBW.

If the data packet is 31 bytes long, or the data length expected as CBW, the LSI saves the data in the CBW area and issues a CBW-complete status (D_BulkIntStat.CWB_Cmp bit) to the firmware. It also automatically clears the C_BulkOnlyControl.GoCBW_Mode bit to terminate execution of CBW support. Furthermore, if the BulkOnlyControl.GoCSW_Mode bit remains set at this point, the LSI clears it too.

If the data packet is less than or greater than 31 bytes in data length, the LSI issues a CBW data length error status (D_BulkIntStat.CBW_LengthErr bit) to the firmware. It also automatically clears the D_BulkOnlyControl.GoCBW_Mode bit to terminate execution of CBW support. Furthermore, if the D_BulkOnlyControl.GoCSW_Mode bit remains set at this point, the LSI clears it too. If a CBW_Err status is issued, it means that a phase mismatch has occurred in the BulkOnly Transport Protocol. Therefore, the firmware should restore communication by, for example, STALL'ing the endpoint.

If D_EPx{x=b,c}Control.ForceSTALL is set at the target endpoint and an OUT transaction is responded with STALL, the LSI issues a CBW error status (D_BulkIntStat.CBW_Err bit) to the firmware and clears the D_BulkOnlyControl.GoCBW_Mode bit to terminate execution of CBW support. If the D_BulkOnlyControl.GoCSW_Mode bit remains set at this point, the LSI clears it too.

If a CRC error or other transaction error occurs in an OUT transaction, the LSI issues a CBW transaction error status (D_BulkIntStat.CBW_TranErr bit) to the firmware without receiving data. In this case, the D_BulkOnlyControl.GoCBW_Mode bit is not cleared and execution of CBW support is continued. Even if the D_BulkOnlyControl.GoCSW_Mode bit remains set at this point, it is not cleared either.

The data received in the CBW area can be read out by using the RAM_Rd function.

6.2.6.2 CSW Support

The firmware can use CSW support when it performs a status transport of the BulkOnly Transport Protocol. When the D_BulkOnlyConfig.EPx{x=b,c}BulkOnly bit is set, CSW support for the corresponding IN endpoint is enabled. Control should be exercised in such a way that CSW support is enabled for only one endpoint at a time. Setting the D_BulkOnlyControl.GoCSW_Mode bit while CSW support is active causes CSW support to be executed, so that the data to be transmitted in an IN transaction at the target endpoint is handled as CSW.

If in an IN transaction, ACK is received from the host after 13 bytes of CSW data was sent back to the host and the transaction is thereby completed, the LSI issues a CSW-complete status (D_BulkIntStat.CSW_Cmp bit) to the firmware. It also automatically clears the D_BulkOnlyControl.GoCSW_Mode bit to terminate execution of CSW support. At the same time, it sets the D_BulkOnlyControl.GoCBW_Mode bit to initiate execution of CBW support.

If in an IN transaction, ACK cannot be received from the host after 13 bytes of data was sent back to the host, the LSI issues a CSW error status (D_BulkIntStat.CSW_Err bit) to the firmware. At this point, the LSI does not clear the D_BulkOnlyControl.GoCSW_Mode bit and continues execution of CSW support. At the same time, it sets the D_BulkOnlyControl.GoCBW_Mode bit in hardware to initiate execution of CBW support. In this case, therefore, execution of CSW support and execution of CBW support are exercised at the same time. If the host could not receive CSW and the transaction resulted in an error, CSW will be retried, but because CSW support is being executed, a response can be returned. Furthermore, if the device could not receive ACK and the transaction resulted in an error, the next CBW will be performed, but because CBW support is being executed, a response can be returned. Execution of CSW support is terminated by the CBW support thus executed.

Data can be written to the CSW area by using the RAM_WrDoor function.

6.2.7 Auto Negotiation Function

Suspend Detection, Reset Detection, HS Detection Handshaking, Resume Detection, and Restore Execution are automatically performed while checking the USB bus state each time. What has actually been executed can be confirmed by checking the respective interrupts (DetectRESET, DetectSUSPEND, ChirpCmp, or RestoreCmp).

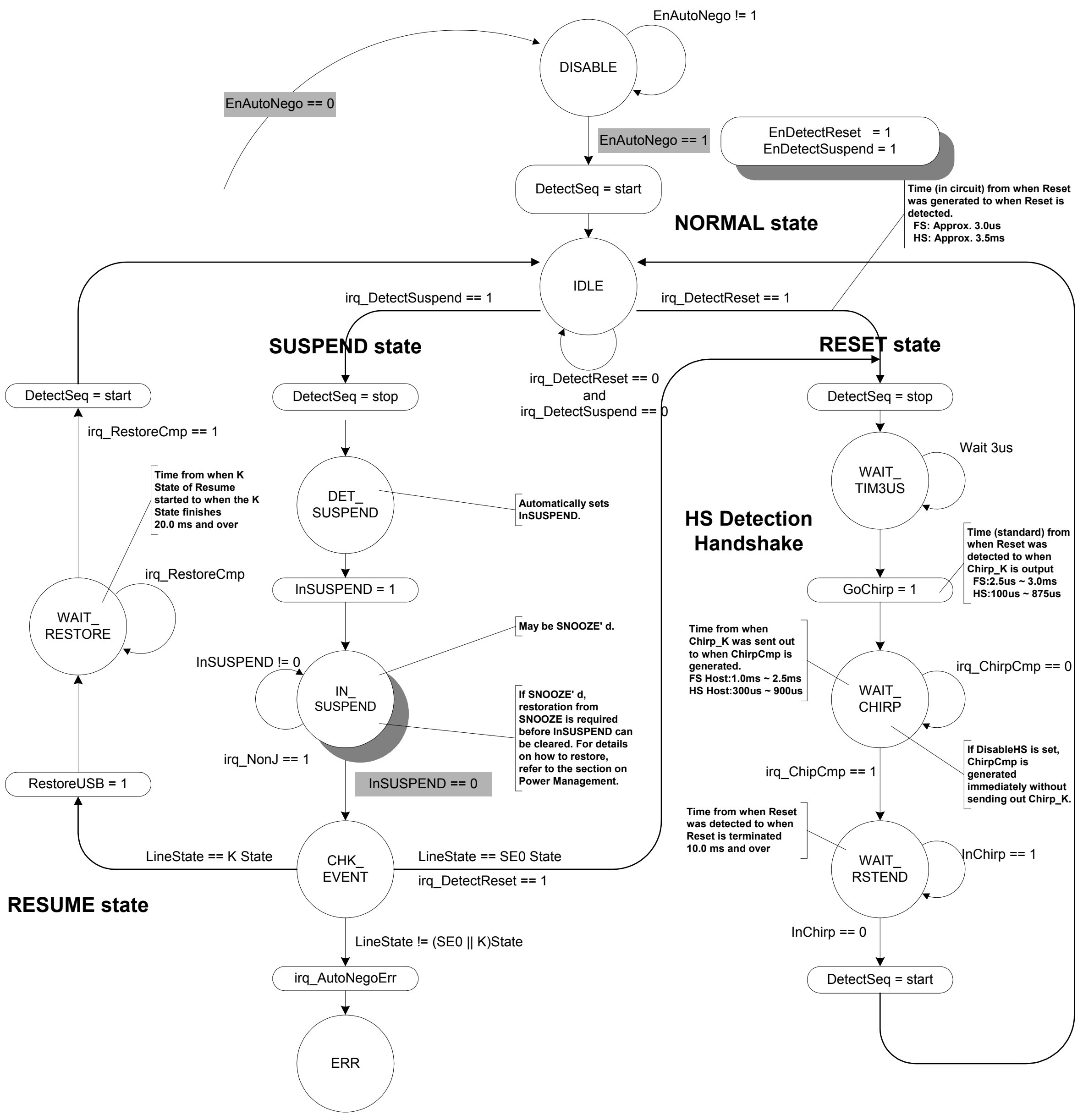


Fig. 6.7 Auto Negotiator

6.2.7.1 DISABLE

The state shifts to this state when the D_NegoControl.EnAutoNego bit is cleared.

When enabling the auto negotiation function, set the reset detection interrupt enable bit (D_SIE_IntEnb.EnDetectRESET) and the suspend detection interrupt enable bit (D_SIE_IntEnb.EnDetectSUSPEND) to enable both event detection interrupts before setting the D_NegoControl.EnAutoNego bit.

When the auto negotiation function is enabled, the internal event detection function is enabled automatically. Do not set the D_NegoControl.DisBusDetect bit while the auto negotiation function is enabled.

6.2.7.2 IDLE

This is the waiting state for Reset Detection or Suspend Detection.

When the current USB speed is HS, if no activity on the USB bus is detected for 3 ms or more, the FS termination is temporarily enabled and then Suspend is assumed if FS-J is detected, or Reset is assumed if SE0 is detected. When the current USB speed is FS, Reset is assumed if SE0 in duration of 2.5 μs or more is detected, or Suspend is assumed if no bus activity is detected for 3 ms or more. A reset detection or suspend detection interrupt is generated at the same time the above judgment is made, and the D_SIE_IntStat.DetectRESET or D_SIE_IntStat.DetectSUSPEND bit is set.

When Suspend is assumed, the event detection function is temporarily turned off and the LSI is shifted to the DET_SUSPEND state.

When Reset is assumed, the event detection function is temporarily turned off and the LSI is shifted to the WAIT_TIM3US state.

6.2.7.3 WAIT_TIM3US

This state is provided for adjusting the time before HS Detection Handshaking is executed after reset has been detected. The WAIT_CHIRP enters the state after certain predetermined time has elapsed (approx. 3 μs later).

6.2.7.4 WAIT_CHIRP

HS Detection Handshaking is executed by automatically setting the D_NegoControl.GoChirp bit. When HS Detection Handshaking finishes, the Chirp-complete interrupt status (D_SIE_IntStat.ChirpCmp) is set and the LSI is shifted to the WAIT_RSTEND state. For details about HS Detection Handshaking, refer to Section 6.2.7.11.5.

Furthermore, while the D_NegoControl.DisableHS bit remains set, the Chirp-complete interrupt status (D_SIE_IntStat.ChirpCmp) is set and the state shifts to the WAIT_RSTEND state without executing HS Detection Handshaking.

Note that after this state terminates, the device operates at the transfer speed that is set in the D_USB_Status.FSxHS bit. If it is necessary to detect that the transfer speed has changed, set the D_SIE_IntEnb.EnChirpCmp bit to enable the Chirp-complete interrupt described above.

6.2.7.5 WAIT_RSTEND

The device waits in this state until the reset period terminates. During HS, the reset period is determined as finished when the Chirp transmission from the host (or reception for the LSI) is complete. During FS, the same is assumed when a transition from SE0 to J occurred.

After the reset period is determined as finished, the event detection function is enabled and the LSI re-enters the IDLE state.

6.2.7.6 DET_SUSPEND

When Suspend is assumed, the D_NegoControl.InSUSPEND bit is automatically set and the LSI is shifted to the IN_SUSPEND state. This D_NegoControl.InSUSPEND bit enables the function to detect a bus transition from FS-J to another, making it possible to detect Resume or Reset from the host.

Whether the reduction in the current consumption amount in the chip actually takes place during Suspend would depend on the application. The LSI stipulated herein incorporates measures to reduce the current consumption in two stages (Snooze and Sleep). For details about these measures and on how to control, refer to Section 6.5, “Power Management Function.”

To ensure that Resume (FS-K), or the instruction to terminate Suspend, can be detected, the D_SIE_IntEnb.EnNonJ bit should set by the firmware to enable the NonJ interrupt.

6.2.7.7 IN_SUSPEND

When the NonJ interrupt status (D_SIE_IntStat.NonJ) is set, an instruction to return from Suspend is assumed, so that when the D_NegoControl.InSUSPEND bit is cleared by the firmware, the LSI is shifted to the CHK_EVENT state.

If spontaneous return from Suspend is desired in an application with remote wakeup function enabled, set the D_NegoControl.SendWakeup bit in this state and output FS-K for a period of 1 ms or more but not exceeding 15 ms.

6.2.7.8 CHK_EVENT

Events on the USB cable are checked, and if FS-K is detected, Resume is assumed, or if SE0 is detected, Reset is assumed. When Resume is assumed, the D_NegoControl.RestoreUSB bit is set, and the transfer speed before Suspend (which depends on the D_USB_Status.FSxHS value) is restored. When Reset is assumed, the event detection function is temporarily turned off as for a transition from the IDLE state, and the LSI is shifted to the WAIT_TIM3US state.

If a state that is neither FS-K nor SE0 is detected, the auto negotiation error interrupt status (D_SIE_IntStat.AutoNegoErr) bit is set and the ERR enters the state.

6.2.7.9 WAIT_RESTORE

When the D_SIE_IntStat.RestoreCmp bit is set, the event detection function is enabled and the LSI is shifted to the IDLE state.

6.2.7.10 ERR

Once the LSI is shifted to this state, it will not exit this state unless the Auto Negotiation function is turned off. This state is nonexistent in the USB standard.

Note that in whichever state, no determination is made with regard to the removal of the USB cable. In the event the USB cable is removed, therefore, the application should turn off the Auto Negotiation function.

6.2.7.11 Individual Description of Each Negotiation Function

6.2.7.11.1 Suspend Detection (HS Mode)

If while the LSI stipulated herein is operating in HS mode, no transmit/receive events are detected for 3 ms or more (T1), the function is shifted to the FS mode automatically. (The HS termination is disabled and the FS termination (Rpu) is enabled.) As a result, DP shifts to ‘high' , and the “J” state can be confirmed by checking the D_USB_Status.LineState [1:0] bits. (Be aware that when SE0 is detected, Reset is assumed as noted later.) Beyond this point, if “J” is still detected at T2, the D_SIE_IntStat.DetectSUSPEND bit is set.

At this point, if the D_SIE_IntEnb.EnDetectSUSPEND and DeviceIntEnb.EnD_SIE_IntStat bits have both been set and the MainIntEnb.EnDeviceIntStat bit has also been set, the XINT signal will be asserted simultaneously with the above, so that the Suspend state of USB is assumed. Shown in the diagram below is a device operation when Snooze' d.

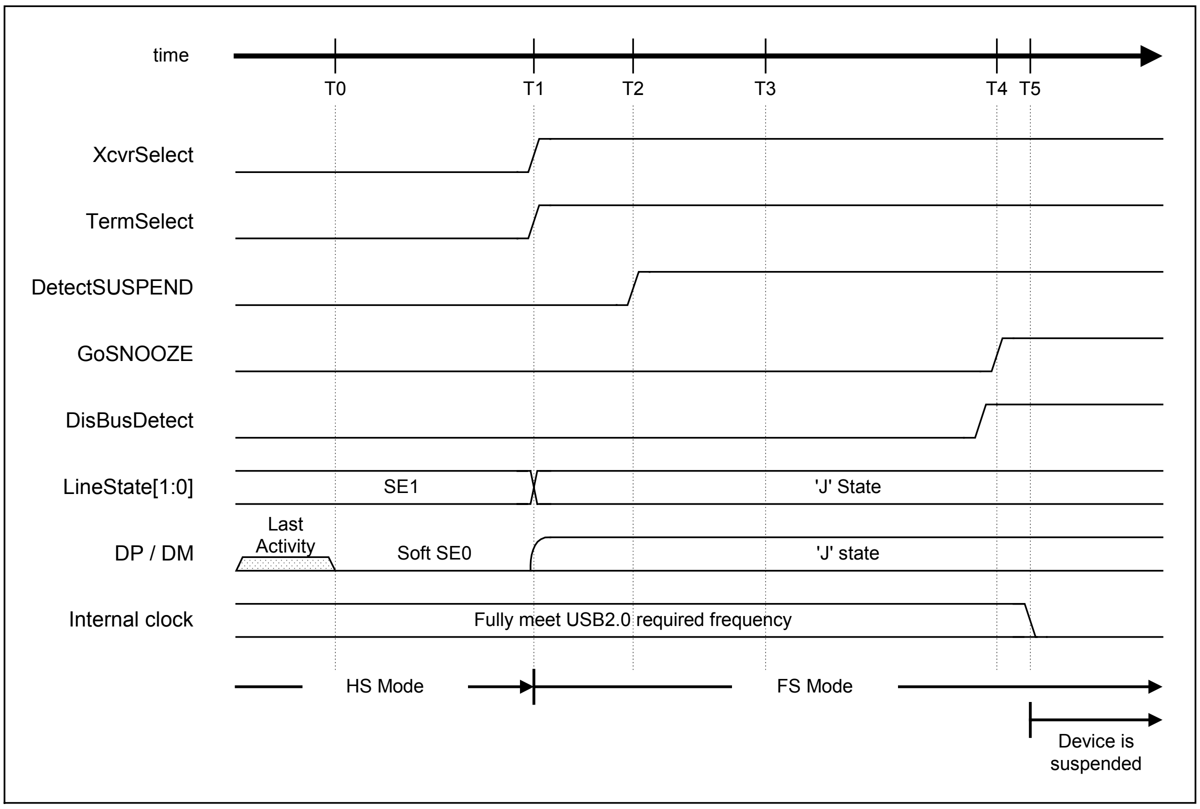


Fig. 6.8 Suspend timing (HS mode)

Table 6.8 Suspend Timing Values (HS Mode)

Timing Parameter	Description	Value
T0	Most recent bus activity.	0 (reference)
T1	If no bus activities are still detected at this point, set XcvrSelect and TermSelect to 1 to change modes from HS to FS.	HS Reset T0 + 3.0ms < T1 {T_{WTREV}} < HS Reset T0 + 3.125ms
T2	Sample LineState [1:0]. If “J” is detected at this point, DetectSUSPEND is set to 1, so that the Suspend state of USB should be assumed.	T1 + 100us < T2 {$T_{WTWRSTHS}$} < T1 + 875us
T3	RESUME cannot be issued prior to this state.	HS Reset T0 + 5ms {T_{WTRSM}}
T4	Set GoSNOOZE to 1, to place the device fully into a Snooze state. Beyond this point, no suspend currents greater than stipulated in USB standard can be drawn from VBUS. (Set DisBusDetect to 1 before entering Snooze state.)	HS Reset T0 + 10ms {T_{2SUSP}}
T5	The internal clock is completely turned off. (Snooze current = 8 mA, typ.)	T5 < T4 + 10us

Note: Names stipulated in the USB2.0 standard are shown in { } .

6.2.7.11.2 Suspend Detection (FS Mode)

If while the LSI stipulated herein is operating in FS mode, no transmit/receive events are detected for 3 ms or more, or “J” in the D_USB_Status.LineState [1:0] bits is detected continuously (T1) and is still detected at T2, the Suspend state of USB is assumed and the SIE_IntStat.DetectSUSPEND bit is set.

At this point, if the D_SIE_IntEnb.EnDetectSUSPEND and DeviceIntEnb.EnD_SIE_IntStat bits have both been set and the MainIntEnb.EnDeviceIntStat bit has also been set, the XINT signal will be asserted simultaneously with the above. Shown in the diagram below is a device operation when it is Snooze' d.

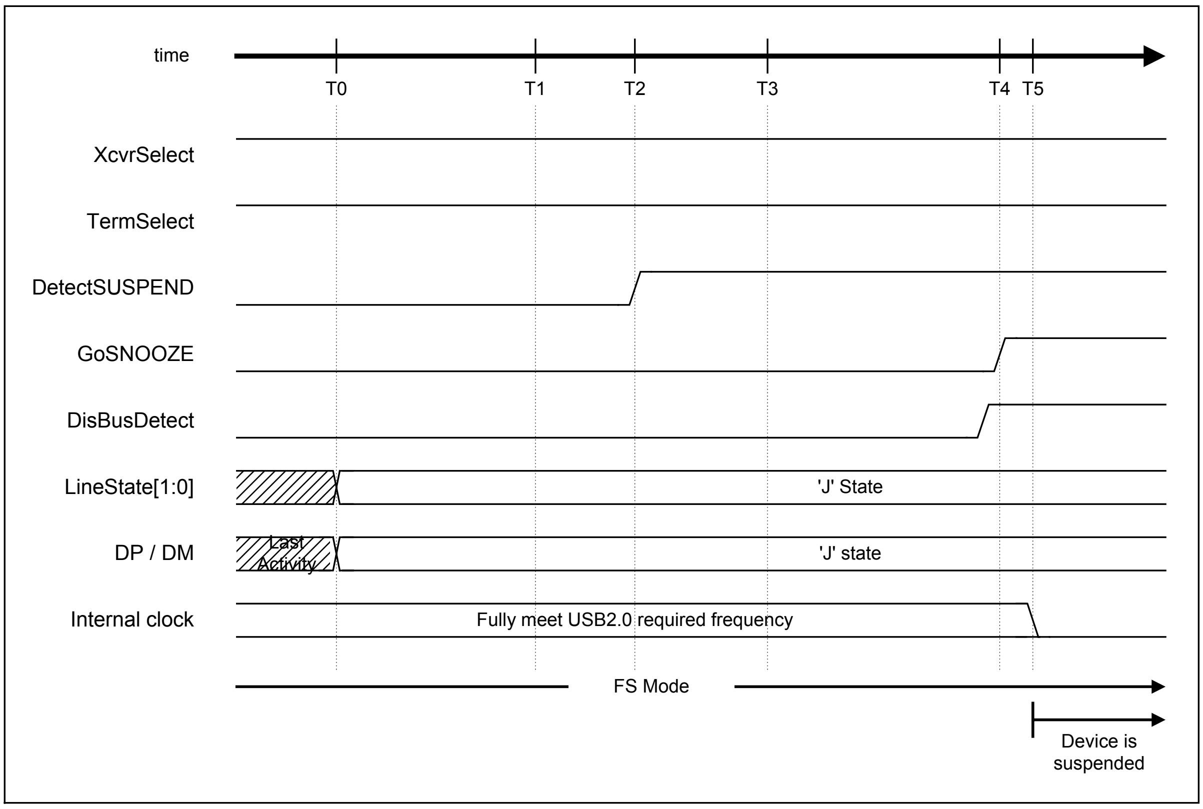


Fig. 6.9 Suspend timing (FS mode)

Table 6.9 Suspend Timing Values (FS Mode)

Timing Parameter	Description	Value
T0	Most recent bus activity.	0 (reference)
T1	No bus activities are still detected at this point.	T0 + 3.0ms < T1 {T_{WTREV}} < T0 + 3.125ms
T2	Sample LineState [1:0]. If "J" is detected at this point, DetectSUSPEND is set to 1, so that the Suspend state of USB should be assumed.	T1 + 100us < T2 {$T_{WTWRSTHS}$} < T1 + 875us
T3	RESUME cannot be issued prior to this state.	T0 + 5ms {T_{WTRSM}}
T4	Set GoSNOOZE to 1, to place the device fully into a Snooze state. Beyond this point, no suspend currents greater than stipulated in USB standard can be drawn from VBUS. (Set DisBusDetect to 1 before entering Snooze state.)	T0 + 10ms {T_{2SUSP}}
T5	The internal clock is completely turned off. (Snooze current = 8 mA, typ.)	T5 < T4 + 10us

Note: Names stipulated in the USB2.0 standard are shown in { } .

6.2.7.11.3 Reset Detection (HS Mode)

If while the LSI stipulated herein is operating in HS mode, no transmit/receive events are detected for 3 ms or more, the function is shifted to the FS mode automatically. (The HS termination is disabled and the FS termination (Rpu) is enabled.) Even when this operation is performed, the DP line remains low, and consequently "SE0" can be detected in the D_USB_Status.LineState [1:0] bits. If "SE0" is still detected at T2, the D_SIE_IntStat.DetectRESET bit is set.

At this point, if the D_SIE_IntEnb.EnDetectRESET and DeviceIntEnb.EnD_SIE_IntStat bits have both been set and the MainIntEnb.EnDeviceIntStat bit has also been set, the XINT signal will be asserted simultaneously with the above. Therefore, assuming that this is an instruction for Reset, execute HS Detection Handshaking (described later) after setting the D_NegoControl.DisBusDetect bit.

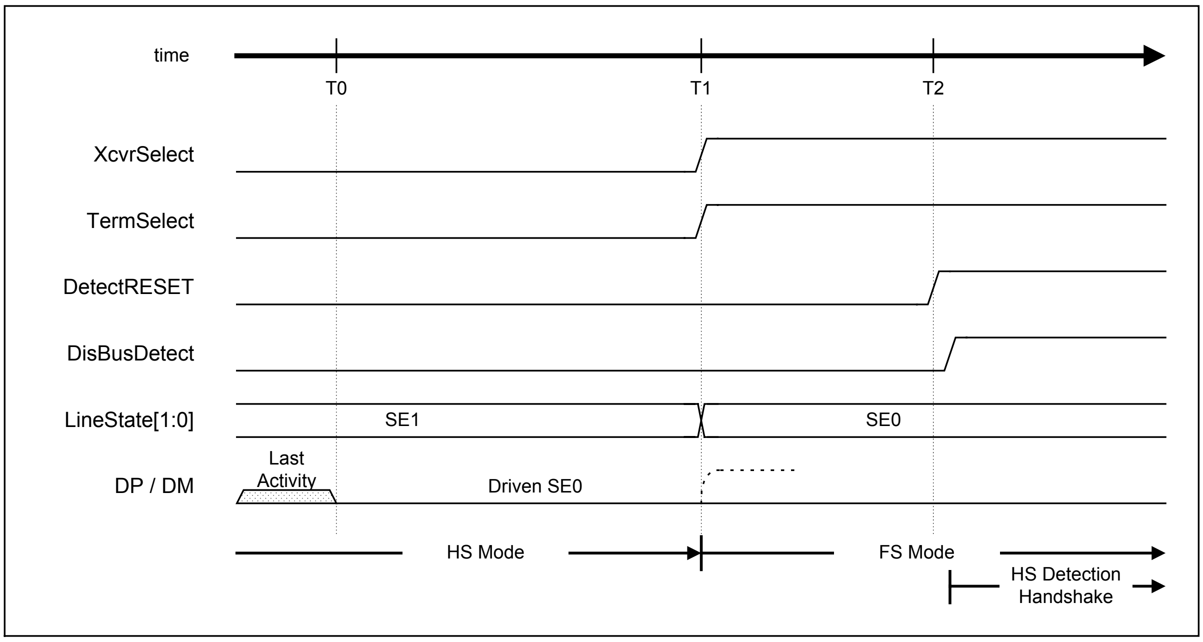


Fig. 6.10 Reset timing (HS mode)

Table 6.10 Reset Timing Values (HS Mode)

Timing Parameter	Description	Value
T0	Most recent bus activity.	0 (reference)
T1	If no bus activities are still detected at this point, set XcvrSelect and TermSelect to 1 to change modes from HS to FS.	HS Reset T0 + 3.0ms < T1 {T_{WTREV}} < HS Reset T0 + 3.125ms
T2	Sample LineState [1:0]. If "SE0" is detected at this point, DetectRESET is set to 1, so that a transition to Reset should be assumed. After detecting the Reset instruction, set DisBusDetect to 1 and then execute HS Detection Handshaking.	T1 + 100us < T2 {$T_{WTWRSTHS}$} < T1 + 875us

Note: Names stipulated in the USB2.0 standard are shown in { } .

6.2.7.11.4 Reset Detection (FS Mode)

If while the LSI stipulated herein is operating in FS mode, “SE0” in the D_USB_Status.LineState [1:0] bits is detected continuously for 2.5 μs or more (T1), the D_SIE_IntStat.DetectRESET bit is set.

At this point, if the D_SIE_IntEnb.EnDetectRESET and DeviceIntEnb.EnD_SIE_IntStat bits have both been set and the MainIntEnb.EnDeviceIntStat bit has also been set, the XINT signal will be asserted simultaneously with the above. Therefore, assuming that this is an instruction for Reset, execute HS Detection Handshaking (described later) after setting the D_NegoControl.DisBusDetect bit.

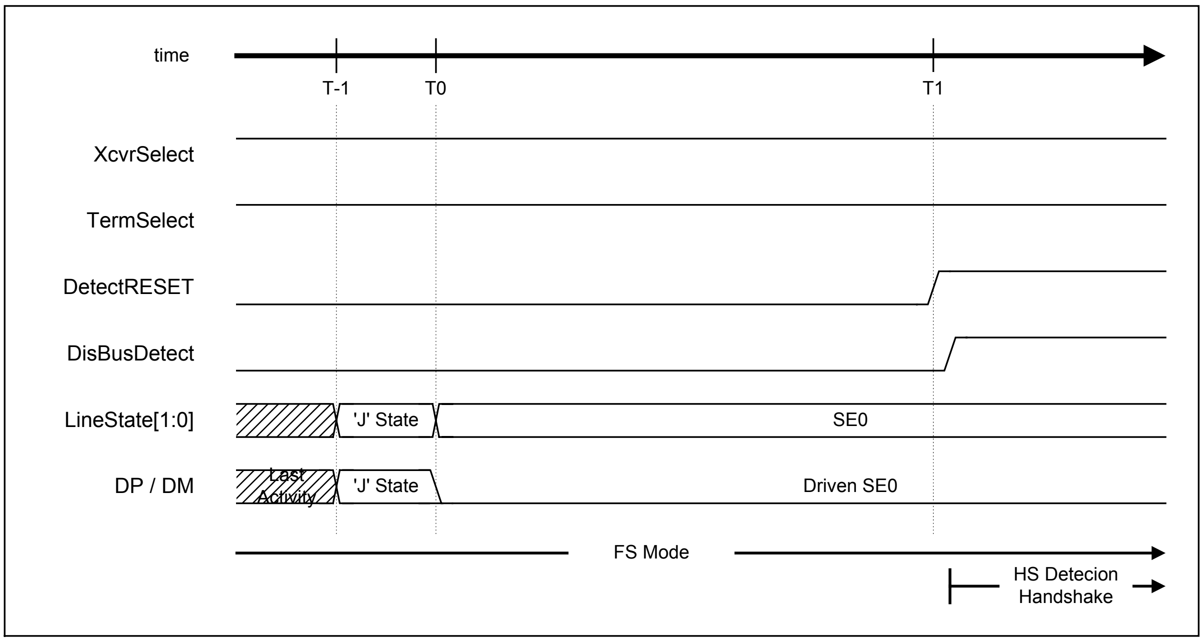


Fig. 6.11 Reset timing (FS mode)

Table 6.11 Reset Timing Values (FS Mode)

Timing Parameter	Description	Value
T-1	Most recent bus activity.	
T0	A reset instruction from the downstream port is initiated.	0 (reference)
T1	If “SE0” continues, DetectRESET is set to 1, so that a transition to Reset should be assumed. After detecting the Reset instruction, set DisBusDetect 1 and then execute HS Detection Handshaking.	HS Reset T0 + 2.5us < T1 {T_{WTREV}}

Note: Names stipulated in the USB2.0 standard are shown in { } .

6.2.7.11.5 HS Detection Handshaking

HS Detection Handshaking is initiated from one of three states—Suspend, FS operation, or HS operation—by the assertion of “SE0” from the downstream port (when Reset is initiated from the above state). For details, refer to the USB2.0 standard.

The following describes how to go to HS Detection Handshaking from the above three states.

While the LSI stipulated herein is in a Suspend state, go to HS Detection Handshaking immediately after detecting “SE0” on the bus.

While the LSI stipulated herein is operating in FS mode, go to HS Detection Handshaking after detecting “SE0” in duration of 2.5 µs or more.

While the LSI stipulated herein is operating in HS mode, temporarily change modes to FS after detecting “SE0” for duration of 3.0 ms or more because it is necessary to determine whether the USB state is Suspend or Reset before going to HS Detection Handshaking. To do this, change the D_XcvrControl.XcvrSelect and D_XcvrControl.TermSelect bits both to FS mode, disable the HS termination, and enable the FS termination. This mode change operation must be performed within 3.125 ms. Within a period of 100 µs to 875 µs after the mode change, check the D_USB_Status.LineState [1:0] bits, and if the bits indicate “J,” the Suspend state of USB should be assumed; if “SE0,” the Reset of USB should be assumed. If Reset is assumed at this point, go to HS Detection Handshaking after that.

In either case, a reset of at least 10 ms exists, but the exact timing differs slightly depending on the state (HS or FS) prior to the transition. Here, the time at which Reset was initiated is defined as “HS Reset T0,” and the explanation below refers to the operation after “HS Reset T0.”

Although there will be no problem during the LSI' s operation because the internal clock has been sufficiently tuned in, the internal clock will become inactive when Reset is detected if the LSI is placed in a sleep or Snooze state during Suspend. Therefore, the PM_Control_0.GoActDevice bit must be set to 1 to activate the internal clock before HS Detection Handshaking can be performed. For details about this operation, refer to Section 6.5, “Power Management Function.”

6.2.7.11.5.1 When Connected to an FS Downstream Port

This section describes the operation of the LSI stipulated herein when it is connected to a downstream port that does not support HS. At the time HS Detection Handshaking is initiated (T0), the D_XcvrControl.XcvrSelect and D_XcvrControl.TermSelect bits must both be in FS mode (with the FS termination, i.e., DP pullup resistor (Rpu) enabled and the HS termination disabled).

First, set the D_NegoControl.GoChirp bit. The XcvrControl.OpMode [1:0] bits will thereby be set to “Disable Bit Stuffing and NRZI encoding,” and data fully populated with 0s will be prepared (T1). This is used to send “HS K” (chirp) on to the bus. If the D_XcvrControl.XcvrSelect bit is set to HS mode and transmission is enabled simultaneously with the above, “HS K” (chirp) is sent out to the downstream port. After the send activity, the LSI waits for “chirp” from the downstream (T2). Usually, if the downstream port supports HS, “HS K” and “HS J” will be sent out successively from the downstream port beginning with T3 (as will be described later). If the downstream port does not support HS (as in the present case), however, “chirp” will not be sent out from the downstream port even at T4. Therefore, the D_XcvrControl.XcvrSelect bit is automatically changed to FS mode, and while the D_USB_Status.FSxHS bit is set simultaneously as the D_NegoControl.GoChirp bit is cleared, along with which the D_SIE_IntStat.ChirpCmp bit is also set.

At this point, if the D_SIE_IntEnb.EnChirpCmp and DeviceIntEnb.EnD_SIE_IntStat bits have both been set and the MainIntEnb.EnDeviceIntStat bit has also been set, the XINT signal will be asserted simultaneously with the above, so that HS Detection Handshaking should be assumed to have finished.

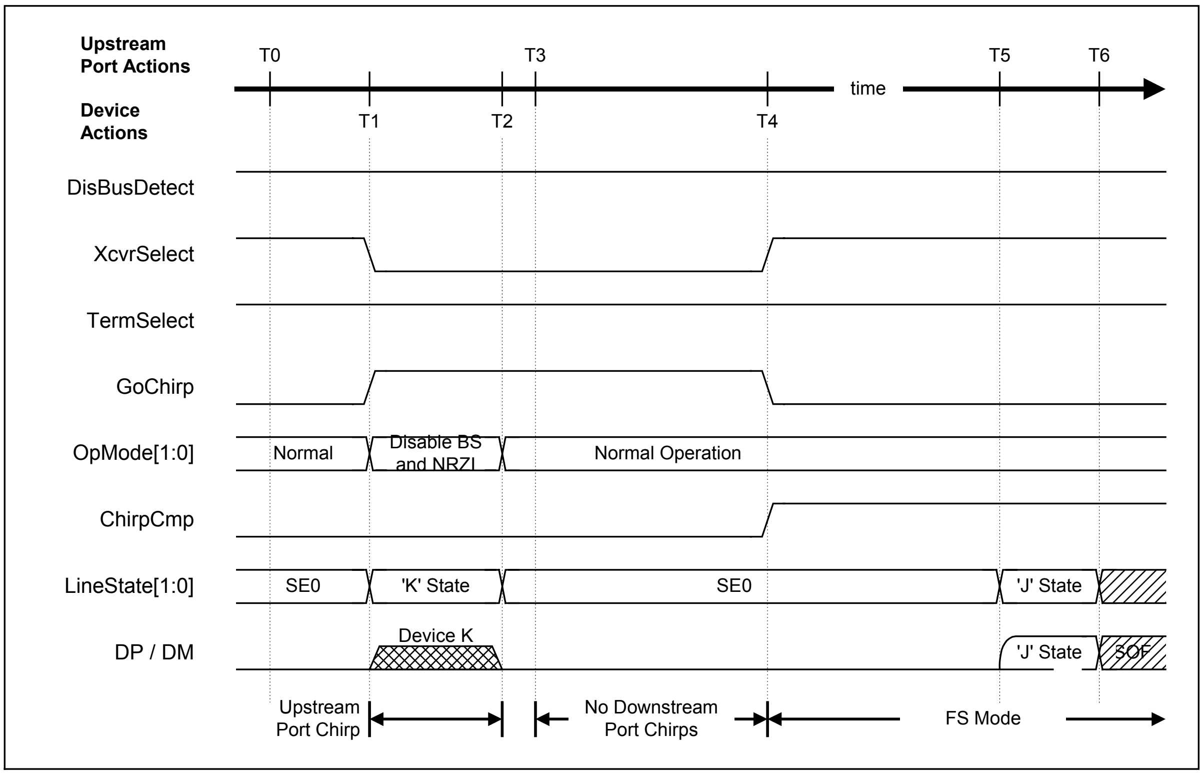


Fig. 6.12 HS Detection Handshake timing (FS mode)

Table 6.12 HS Detection Handshake Timing Values (FS Mode)

Timing Parameter	Description	Value
T0	HS Detection Handshaking is initiated.	0 (reference)
T1	HSEnable the HS transceiver and set GoChirp to 1 to start sending out Chirp K.	T0 < T1 < HS Reset T0 + 6.0ms
T2	Finish sending out Chirp K. This signal must be sent out for at least 1 ms.	T1 + 1.0ms {T_{UCH}} < T2 < HS Reset T0 + 7.0ms {T_{UCHEND}}
T3	If the downstream port supports HS, start sending out Chirp K from here.	T2 < T3 < T2 + 100us {T_{WTDCH}}
T4	If Chirp cannot be detected, return to FS mode at this point and wait until ChirpCmp is set to 1 and the reset sequence finishes.	T2 + 1.0ms < T4 {T_{WTFS}} < T2 + 2.5ms
T5	The reset sequence finishes.	HS Reset T0 + 10ms {T_{DRST} (Min)}
T6	Normal operation in FS mode.	T6

Note: Names stipulated in the USB2.0 standard are shown in { } .

Note: To generate Chirp K of at least 1 ms, determine the duration by 66,000 cycles (internal clock = 60 MHz).

6.2.7.11.5.2 When Connected to an HS Downstream Port

This section describes the operation of the LSI stipulated herein when it is connected to a downstream port that supports HS. At the time HS Detection Handshaking is initiated (T0), the D_XcvrControl.XcvrSelect and D_XcvrControl.TermSelect bits must both be in FS mode (with the FS termination, i.e., DP pullup resistor (Rpu) enabled and the HS termination disabled).

First, set the D_NegoControl.GoChirp bit. The D_XcvrControl.OpMode [1:0] bits will thereby be set to "Disable Bit Stuffing and NRZI encoding," and data fully populated with 0s will be prepared (T1). This is used to send "HS K" (chirp) on to the bus. If the D_XcvrControl.XcvrSelect bit is set to HS mode and transmission is enabled simultaneously with the above, "HS K" (chirp) is sent out to the downstream port. After the LSI has finished sending out, it waits for "chirp" from the downstream (T2). Since the downstream port supports HS in the present case, "HS K" (Chirp K) and "HS J" (Chirp J) are alternately sent out from the downstream port successively (T3). When this state is detected at least six times as Chirp K-J-K-J-K-J by the D_USB_Status.LineState [1:0] bits (T6), the D_XcvrControl.TermSelect bit is automatically changed to HS mode (T7), by which the LSI is placed completely into HS mode. At the same time, the D_NegoControl.GoChirp bit is cleared as is the D_NegoStatus.FSxHS bit, and the D_SIE_IntStat.ChirpCmp bit is set.

At this point, if the D_SIE_IntEnb.EnChirpCmp and DeviceIntEnb.EnD_SIE_IntStat bits have both been set and the MainIntEnb.EnDeviceIntStat bit has also been set, the XINT signal will be asserted simultaneously with the above, so that HS Detection Handshaking should be assumed to have finished.

Chirp K and Chirp J from this downstream port should be recognized as bus activities, and must not be misinterpreted as the Suspend state of USB. To ensure this, when in HS mode, Chirp K and Chirp J are latched into the internal Suspend Timer as they are detected successively.

Note that the D_USB_Status.LineState [1:0] bits are used to detect Chirp K-J-K-J-K-J. Unlike ordinary HS packets, Chirp K and Chirp J are very slow, and this is the reason that the D_USB_Status.LineState [1:0] bits can be used for said purpose. However, if bus signals are superimposed on the D_USB_Status.LineState [1:0] bits, the bus signals become extremely noisy, so that when the D_XcvrControl.TermSelect bit is in HS mode, the D_USB_Status.LineState [1:0] bits will output "J" if presence of bus activity is assumed, or "SE0" if absence of bus activity is assumed.

In the diagram below, the change of the Chirp level beginning at the point of T6 indicates that the HS termination on the device side has been enabled by the D_XcvrControl.TermSelect bit. Normally, when D_XcvrControl.TermSelect is in FS mode, Chirp is approximately 800 mV, and when D_XcvrControl.TermSelect is in HS mode, Chirp (as for ordinary transmit/receive packets in HS) is approximately 400 mV.

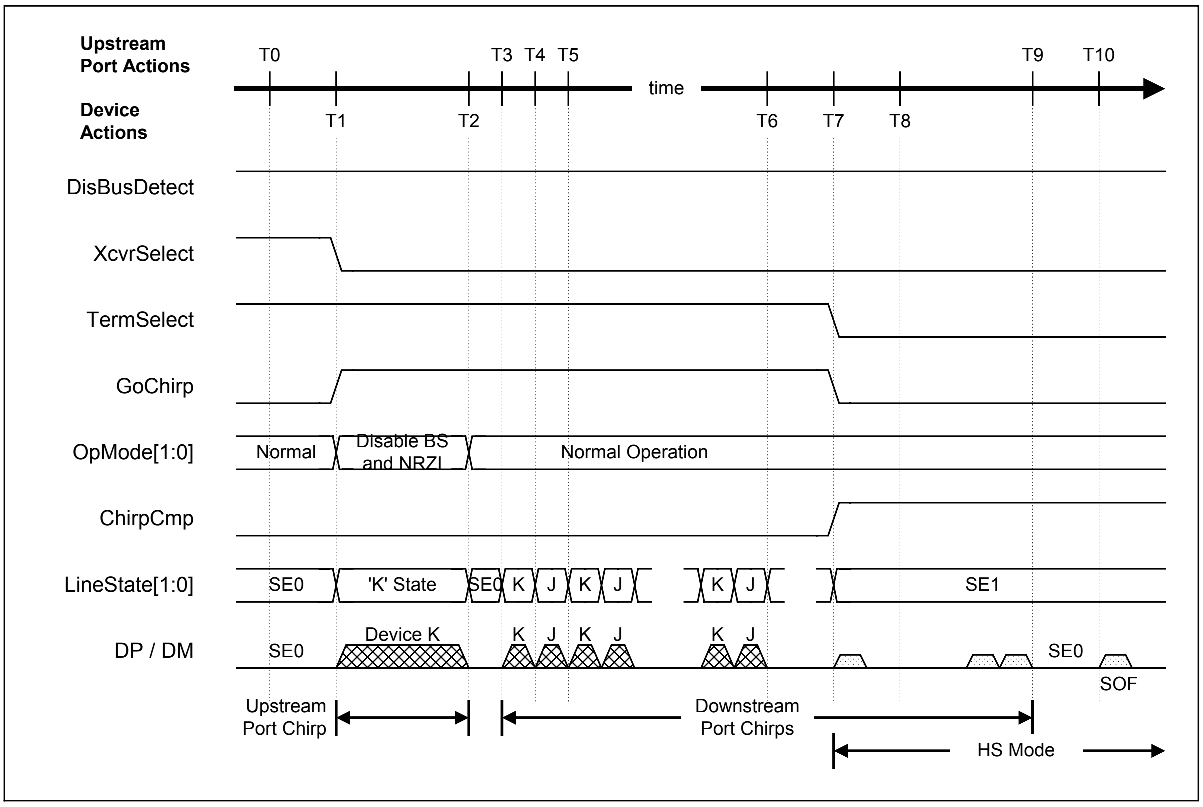


Fig. 6.13 HS Detection Handshake timing (HS mode)

Table 6.13 HS Detection Handshake Timing Values (HS Mode)

Timing Parameter	Description	Value
T0	HS Detection Handshaking is initiated.	0 (reference)
T1	Enable the HS transceiver and set GoChirp to 1 to start sending out Chirp K.	T0 < T1 < HS Reset T0 + 6.0ms
T2	Finish sending out Chirp K. This signal must be sent out for at least 1 ms.	T1 + 1.0ms {T_{UCH}} < T2 < HS Reset T0 + 7.0ms {T_{UCHEND}}
T3	The downstream port sends the first Chirp K on to the bus.	T2 < T3 < T2 + 100us {T_{WTDCH}}
T4	The downstream port stops sending Chirp K and sends out Chirp J instead.	T3 + 40us {T_{DCHBIT} (Min)} < T4 < T3 + 60us {T_{DCHBIT} (Max)}
T5	The downstream port stops sending Chirp J and sends out Chirp K instead.	T4 + 40us {T_{DCHBIT} (Min)} < T5 < T4 + 60us {T_{DCHBIT} (Max)}
T6	Chirp K-J-K-J-K-J are detected.	T6
T7	Pursuant to the detection of Chirp K-J-K-J-K-J, disable the FS termination and enable the HS termination. ChirpCmp is set to 1. Then, wait until Reset finishes.	T6 < T7 < T6 + 500us
T8	Recognized as bus activity due to Chirp K and Chirp J. However, since SYNC cannot be detected, this is not recognized as a packet reception in progress.	T8
T9	Transmission of Chirp K and Chirp J from the downstream port finishes.	T10 - 500us {T_{DCHSE0} (Max)} < T9 < T10 - 100us{T_{DCHSE0} (Min)}
T10	The reset sequence finishes.	HS Reset T0 + 10ms {T_{DRST} (Min)}

Note: Names stipulated in the USB2.0 standard are shown in { } .

Note: To generate Chirp K of at least 1 ms, determine the duration by 66,000 cycles (internal clock = 60 MHz).

6.2.7.11.5.3 When Reset during Snooze

The LSI stipulated herein does not have its internal clocks output during a Snooze state. The following describes operation of the LSI assuming that its oscillator circuit is active (in a Snooze state, and not in a Sleep state).

If Reset is detected during a Snooze state (T0), the D_SIE_IntStat.NonJ bit is set. Furthermore, if the D_SIE_IntEnb.EnNonJ and DeviceIntEnb.EnD_SIE_IntStat bits have both been set and the MainIntEnb.EnDeviceIntStat bit has also been set, the XINT signal will be asserted simultaneously with the above. In this case, to allow the LSI to immediately return from the Snooze state and go to a reset sequence, set the PM_Control.GoActDevice bit to 1 (T1). After the PLL power-up time (T2) has elapsed, PM_Control_1.PM_State [2:0] becomes "ACT_DEVICE," at which time the internal clock starts oscillating. After that, execute HS Detection Handshaking (described above).

At this point, unless the oscillator circuit has been turned off (unless returning from a Sleep state), the internal clock is output with the frequency accuracy conforming to the USB2.0 standard.

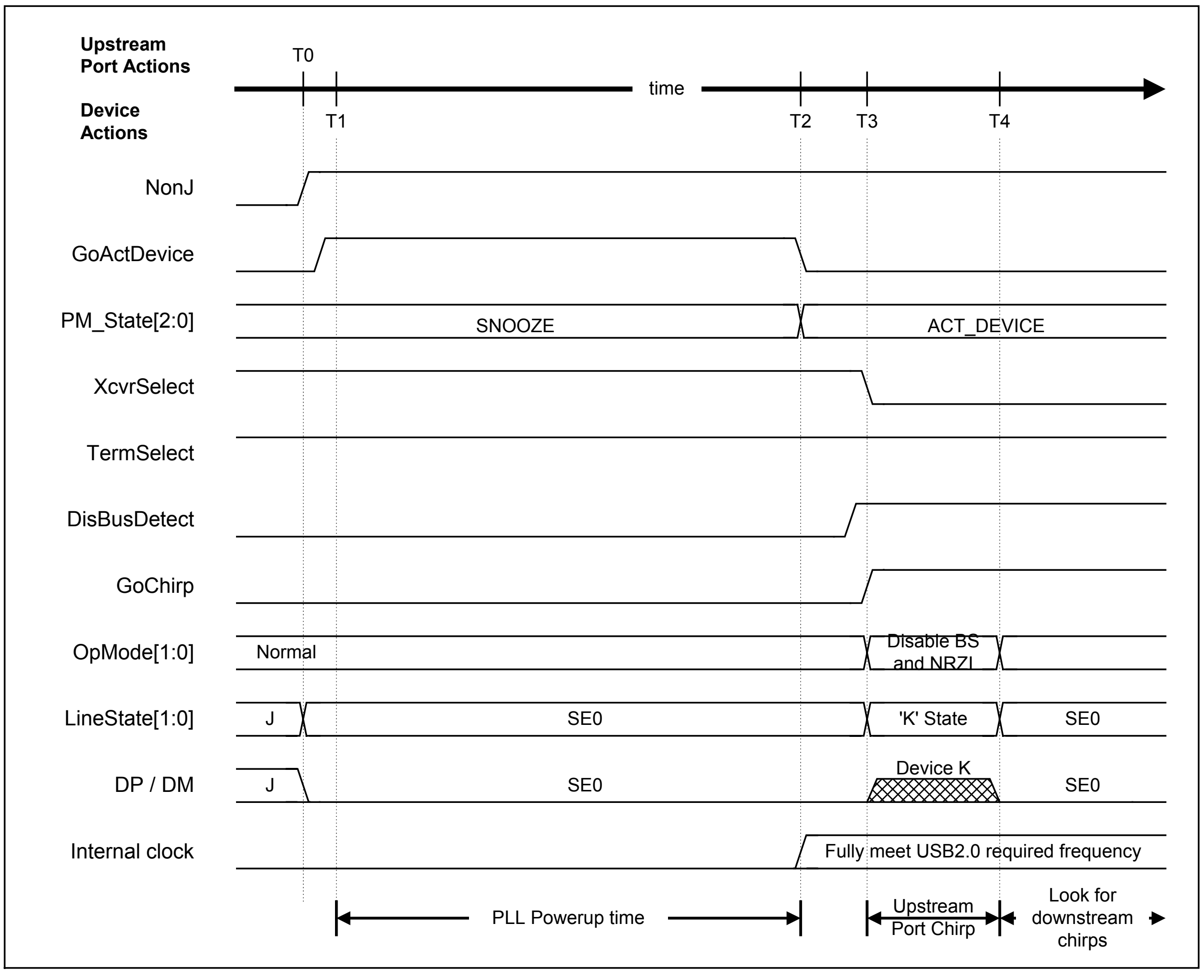


Fig. 6.14 HS Detection Handshake Timing from Suspend

Table 6.14 HS Detection Handshake Timing Values from Suspend

Timing Parameter	Description	Value
T0	When NonJ is set to 1 and “SE0” is confirmed by LineState [1:0], a reset during snooze is detected.	0 (HS Reset T0)
T1	After detection of Reset, set GoActDevice to 1.	T1
T2	PM_State becomes “ACT_DEVICE.” Internal clock output stabilizes.	T1 + 250us < T2
T3	Set GoChirp to 1 to send Chirp K on to the bus. (Set DisBusDetect to 1 before sending out Chirp K.)	T2 < T3 < HS Reset T0 + 5.8ms
T4	Finish sending out Chirp K.	T3 + 1.0ms {T_{UCH}} < T4 < HS Reset T0 + 7.0ms {T_{UCHEND}}

Note: Names stipulated in the USB2.0 standard are shown in { } .

Note: To generate Chirp K of at least 1 ms, determine the duration by 66,000 cycles (internal clock = 60 MHz).

Note: If the oscillator circuit has also been turned off (Sleep state), an OSC power-up time as well as the PLL power-up time described later are required.

6.2.7.11.6 Issuance of Resume

When remote wakeup is supported and this remote wakeup function is enabled from the host, there may be a case when the device needs to resume by itself for certain reason. This section describes how to resume in such a case. Note, however, that at least 5 ms must elapse after the bus became idle before remote wakeup can be executed. Furthermore, no currents in a state prior to shift to the Suspend state of USB can be drawn from VBUS before the passage of 10 ms after the Resume signal was output.

For remote wakeup to be executed, the device must first be restored from sleep/snooze. Clear the D_SIE_IntEnb.EnNonJ bit and set the PM_Control.GoActDevice bit (T0), and when the PM_Control_1.PM_State [2:0] bits are set to “ACT_DEVICE.” after the PLL power-up time (T1) has elapsed, the internal clock starts oscillating. At this point, unless the oscillator circuit has been turned off, the internal clock is output with the frequency accuracy conforming to the USB2.0 standard.

Following this, set the D_NegoControl.SendWakeup bit to send out a Resume signal (T2). At this point, internally in the device, the D_XcvrControl.OpMode [1:0] bits are set to “Disable Bit Stuffing and NRZI encoding” and data consisting of 0s are prepared as the transmit data, and after a packet transmit enters the state, “K” (Resume signal) is sent out. Upon detecting this resume signal, the downstream port returns “K” (resume signal) on to the bus (T3).

The resume signal being sent out to the bus is stopped by clearing the D_NegoControl.SendWakeup bit about 1 ms after the device started sending out the resume signal (T4). At this point, however, the downstream port still holds the resume signal on the bus.

Therefore, set the D_NegoControl.RestoreUSB bit. After a certain predetermined time has elapsed, the downstream port stops sending back the resume signal (T5) and instead sends out a 2-bit LS-EOP (2*SE0) to switch the speed mode to that of prior to Suspend of USB. When this mode change (no longer “K”) is detected, the D_XcvrControl.XcvrSelect and D_XcvrControl.TermSelect bits are both changed to the desired mode (in the present case, HS mode), and the D_NegoControl.RestoreUSB bit is cleared and the D_SIE_IntStat.RestoreCmp bit is set simultaneously with it. At this point, if the D_SIE_IntEnb.EnRestoreCmp and DeviceIntEnb.EnD_SIE_IntStat bits have both been set and the MainIntEnb.EnDeviceIntStat bit has also been set, the XINT signal will be asserted simultaneously with the above.

When Suspend of USB is initiated, the speed mode (HS or FS) is saved in the USB_Status.FSxHS bit, so that when restored by Resume, the device returns to the mode indicated by this USB_Status.FSxHS bit. At this point, HS Detection Handshaking does not need to be executed for each Resume attempted. Please note that the explanation made here refers to only the case where the speed mode prior to Suspend of USB was HS. Actually, when in FS mode, the states following T5 become FS mode and there is no significant difference in the sequence.

When the LSI stipulated herein is in a Snooze state (PM_Control_1.PM_State [2:0] bits = SNOOZE), it does not have its internal clocks output. The device operation explained here assumes that the oscillator circuit is active (in a Snooze state, and not in a Sleep state).

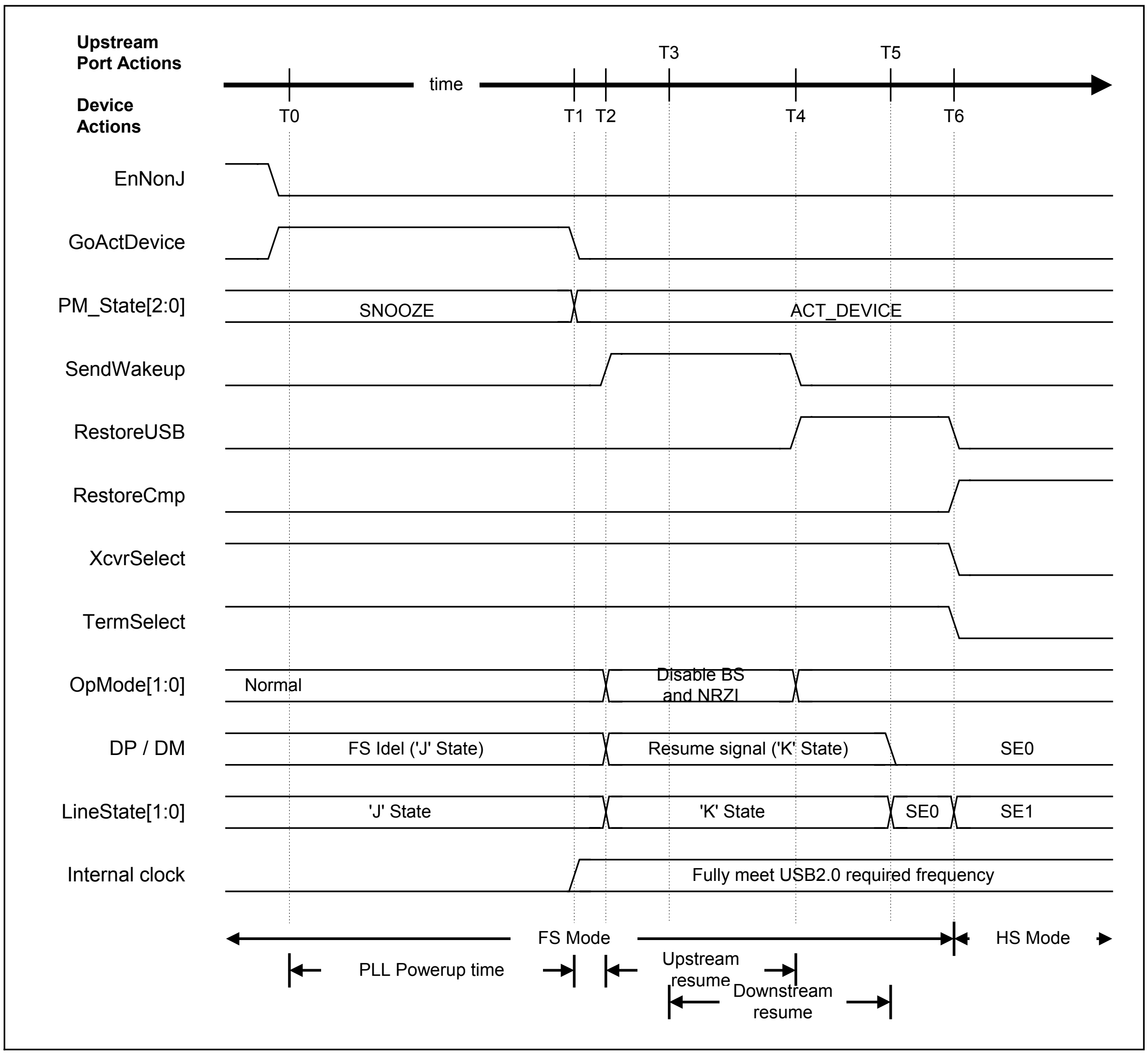


Fig. 6.15 Assert Resume timing (HS mode)

Table 6.15 Assert Resume Timing Values (HS Mode)

Timing Parameter	Description	Value
T0	Resume is initiated. Set GoActDevice to 1. (EnNonJ must be cleared to 0 before Resume is initiated.)	0 (reference)
T1	PM_State becomes “ACT_DEVICE.” Internal clock output stabilizes.	T0 + 250us < T1
T2	Set SendWakeup to 1 to start sending out “K” for FS. Here, no currents in a state prior to Suspend of USB can be drawn from VBUS within 10 ms after that.	T0 < T2 < T0 + 10ms
T3	The downstream port returns “K” for FS.	T2 < T3 < T2 + 1.0ms
T4	Clear SendWakeup to 0 to finish sending out “K” for FS. After confirming “K” by LineState [1:0], set RestoreUSB to 1.	T2 + 1.0ms {T_{DRSMUP} (Min)} < T4 < T2 + 15ms {T_{DRSMUP} (Max)}
T5	The downstream port finishes sending “K” for FS.	T2 + 20ms {T_{DRSMDN}}
T6	RestoreCmp is set to 1. If the mode prior to Suspend of USB was HS, the device automatically shifts to HS mode.	T5 + 1.33us {2 Low-speed bit times}

Note: Names stipulated in the USB2.0 standard are shown in { }.

6.2.7.11.7 Detection of Resume

While the LSI stipulated herein is snoozing, “J” (D_USB_Status.LineState [1:0] = J) will be observed on the bus. If “K” is observed on the bus, it means that an instruction for wakeup (instruction for Resume) from the downstream port has been received (T0). At this point, unless the oscillator circuit is turned off (not in a Sleep state), the SIE_IntStat.NonJ bit is set. If the D_SIE_IntEnb.EnNonJ and DeviceIntEnv.EnD_SIE_IntStat bits have both been set and the MainIntEnb.EnDeviceIntStat bit has also been set at this point, the XINT signal will be asserted simultaneously with the above.

First, set the PM_Control.GoActDevice bit to 1 (T1), and when the PM_Control_1.PM_State [2:0] bits are set to “ACT_DEVICE” after the PLL power-up time (T2) has elapsed, the internal clock starts oscillating. At this point, unless the oscillator circuit has been turned off, the internal clock is output with the frequency accuracy conforming to the USB2.0 standard.

Therefore, set the D_NegoControl.RestoreUSB bit. After certain predetermined time has elapsed, the downstream port will stop sending the resume signal (T3) to switch the speed mode to that of prior to Suspend of USB. When this mode change (no longer “K”) is detected, the D_XcvrControl.XcvrSelect and D_XcvrControl.TermSelect bits both are changed to the desired mode (in the present case, HS mode), and the D_NegoControl.RestoreUSB bit is cleared and the D_SIE_IntStat.RestoreCmp bit is set simultaneously with it. At this point, if the D_SIE_IntEnb.EnRestoreCmp and DeviceIntEnb.EnD_SIE_IntStat bits have both been set and the MainIntEnb.EnDeviceIntStat bit has also been set, the XINT signal will be asserted simultaneously with the above.

The device operation explained here assumes that the oscillator circuit is active (in a Snooze state, and not in a Sleep state).

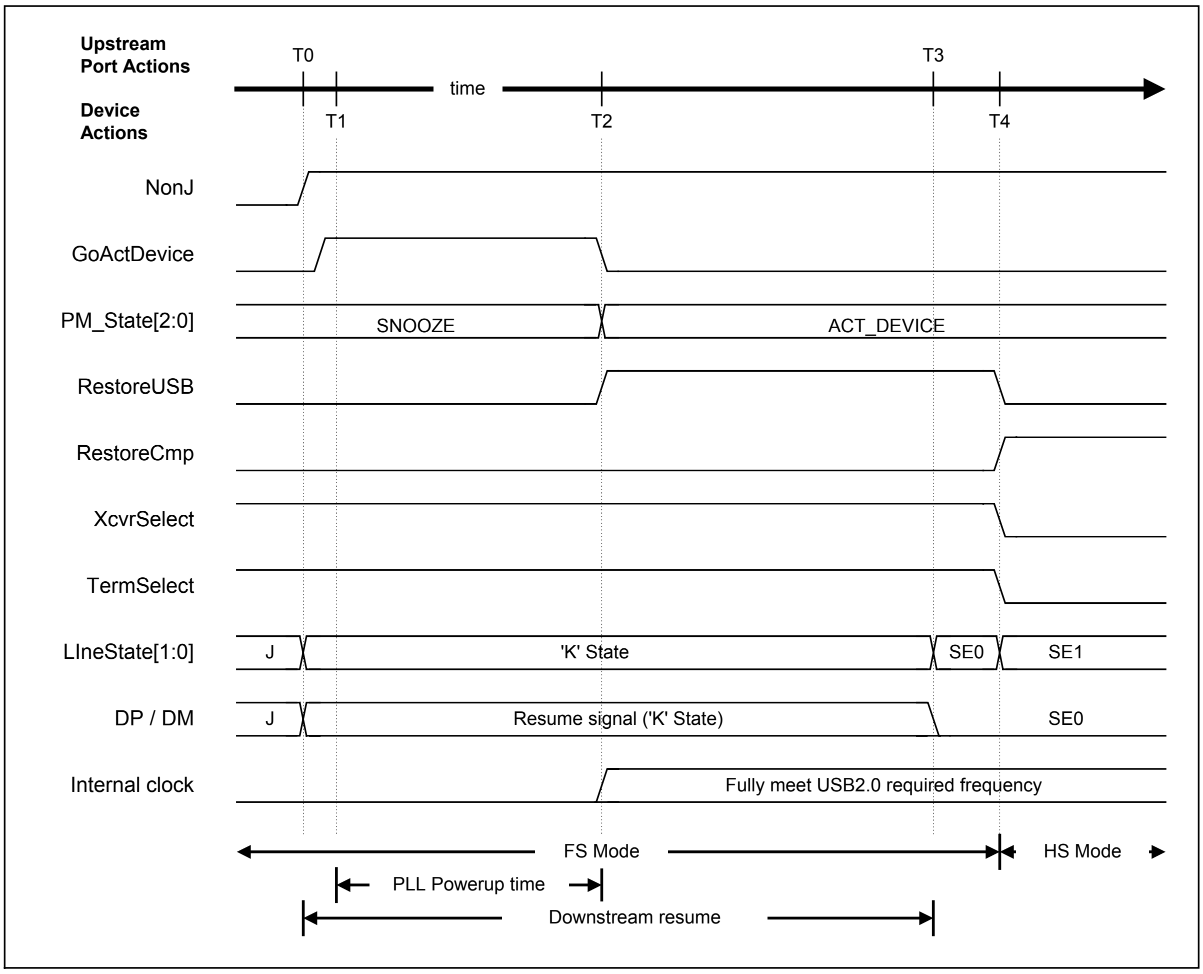


Fig. 6.16 Detect Resume timing (HS mode)

Table 6.16 Detect Resume Timing Values (HS Mode)

Timing Parameter	Description	Value
T0	The downstream port sends out “K” for FS. NonJ is set to 1.	0 (reference)
T1	Set GoActDevice to 1.	T1
T2	PM_State is shifted to “ACT_DEVICE.” Internal clock output stabilizes. After confirming “K” with LineState [1:0], set RestoreUSB to 1.	T1 + 250us < T2
T3	The downstream port finishes sending out “K” for FS. At the same time, it shifts to HS mode in which it was prior to Suspend of USB.	T2 + 20ms {T_{DRSMDN}}
T4	If the mode prior to Suspend of USB was HS, the device automatically shifts to HS mode.	T5 + 1.33us {2 Low-speed bit times}

Note: Names stipulated in the USB2.0 standard are shown in { }.

6.2.7.11.8 Insertion of USB Cable

This section describes the case where the device is connected to a hub or the host, i.e., it has had USB cable inserted.

When cable is removed or intentionally maintained in a disconnected state, make sure the D_XcvrControl.XcvrSelect and D_XcvrControl.TermSelect bits default to FS mode and HS mode, respectively.

When cable is connected while no cable has been connected (T0), VBUS shifts to ‘high' and at the same time, the D_USB_Status.VBUS bit is set (T1). If the device has been placed in a Snooze state at this point, set the PM_Control.GoActDevice bit to 1 (T2), and when the PM_Control_1.PM_State [2:0] becomes “ACT_DEVICE.” after the PLL power-up time (T3) has elapsed, the internal clock will start oscillating simultaneously with it. After this, since the function must shift to the FS mode temporarily in order to pretend that the FS device has been connected, set the D_XcvrControl.TermSelect bit to FS mode (T4).

Beyond this point, the downstream port sends out Reset (T5), from which HS Detection Handshaking is initiated.

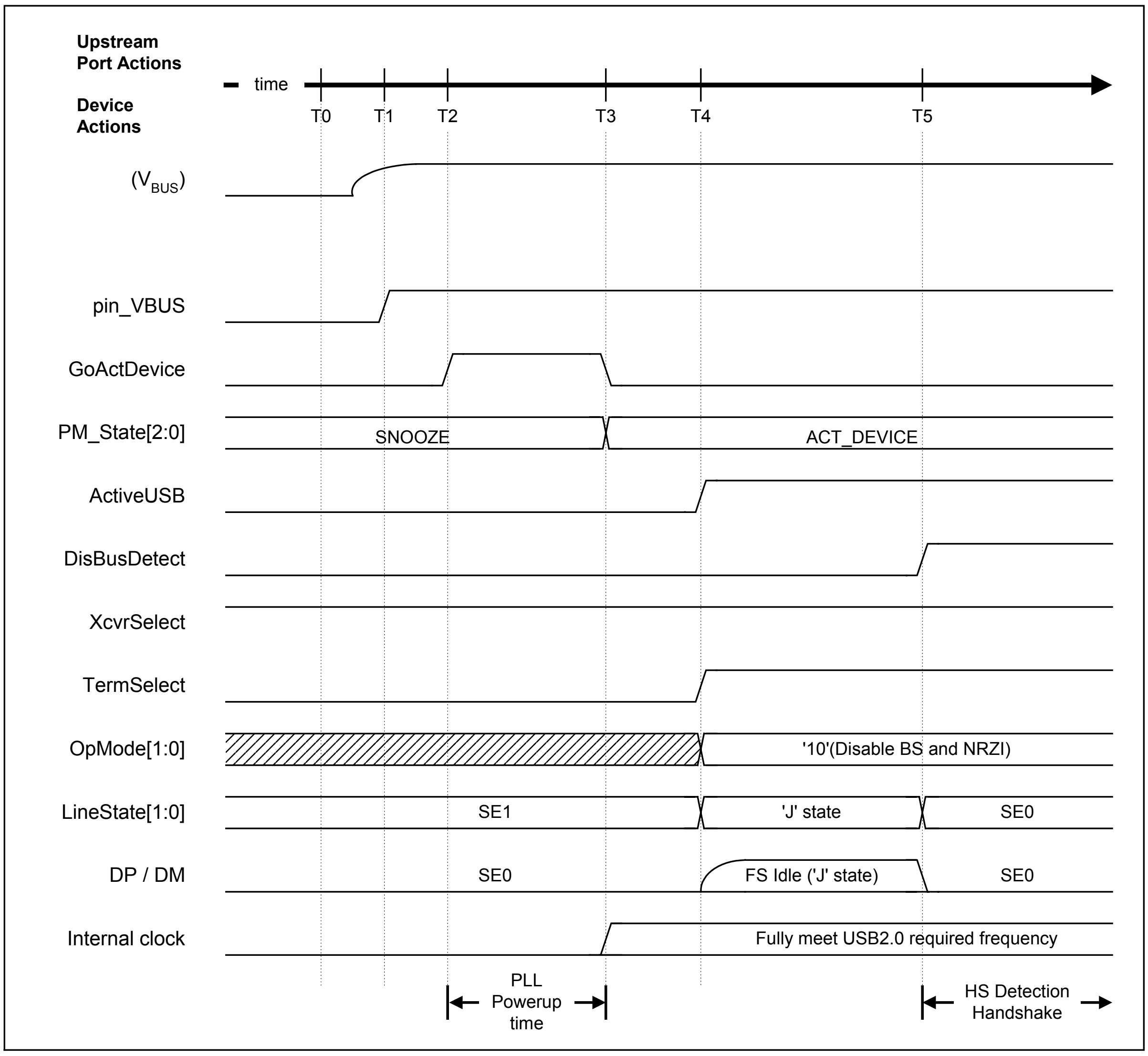


Fig. 6.17 Device Attach timing

Table 6.17 Device Attach Timing Values

Timing Parameter	Description	Value
T0	No cable is inserted.	0 (reference)
T1	Cable is inserted, and the input pin VBUS_B shifts to 'high' .	T1
T2	Set GoActDevice to 1.	T2
T3	PM_State becomes "ACT_DEVICE." Internal clock output stabilizes.	T2 + 250us < T3
T4	Set ActiveUSB to 1. Set TermSelect to 1. Set OpMode [1:0] to '00.' The function is shifted to FS mode. FS termination is enabled.	T1 + 100ms {T_{SIGATT}} < T4
T5	Reset is sent from the downstream port. Set DisBusDetect to 1.	T4 + 100ms {T_{ATTDB}} < T5

Note: Names stipulated in the USB2.0 standard are shown in { } .

6.3 USB Host Control

6.3.1 Channels

6.3.1.1 Channel Overview

The LSI stipulated herein has the host-side buffers corresponding one for one to the pipe and various setup registers used for transfers performed via those buffers, collectively referred to as "channels."

Transfer information is set in units of IRP (I/O Request Packet) for a channel. The channel divides the IRP into multiple transactions based on the set information as it executes transfer. Since a channel can switch over settings in IRP units, one channel can handle transfers for multiple endpoints.

Fig. 6.18 schematically shows a channel.

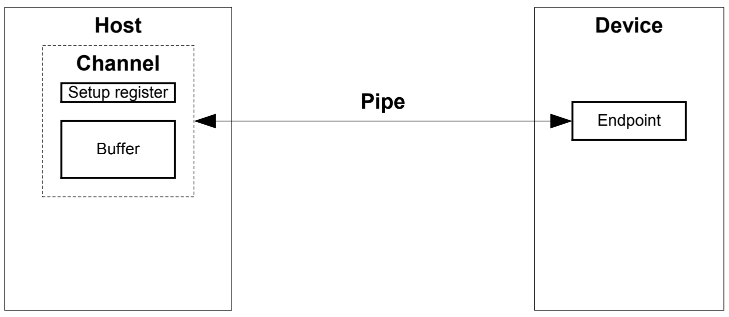


Fig. 6.18 Conceptual diagram of a channel

The firmware sets buffers and transfer information first and then sets transfer execution. After setting transfer execution, the firmware performs several processes by writing data to the buffer (for OUT transfer) or reading data from the buffer (for IN transfer) until processing for bytes of IRP data finishes.

On the other hand, the hardware (channel) automatically divides a IRP into multiple transactions as it executes transfer. When the transfer finishes, it notifies the firmware to that effect via an interrupt.

The buffer on each channel can be allocated to any desired area in the internal RAM of the LSI by the firmware, and can each operate as an independent FIFO.

Fig. 6.19 shows the basic flow of operation for a transfer to be performed.

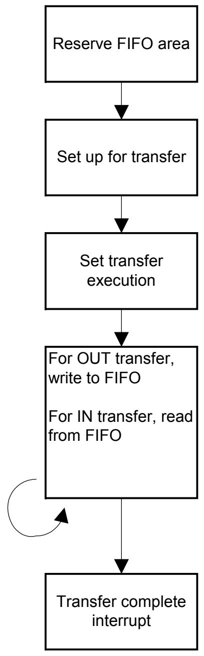

Reserve FIFO area
Set up for transfer
Set transfer execution
For OUT transfer, write to FIFO
For IN transfer, read from FIFO
Transfer complete interrupt

Fig. 6.19 Basic procedure for transfer on a channel

The LSI stipulated herein has a total of six channels, consisting of a channel that performs only a control transfer (CH0), a channel that performs only a bulk transfer (CHa), and channels that perform bulk/interrupt transfers (CHb, CHc, CHd, and CHe). Here, channel CH0 is referred to as a control-only channel, while channels CHa, CHb, CHc, CHd, and CHe are referred to as general-purpose channels.

Each channel has fixed basic setup items determined by the USB-defined interface and variable control items and status used for control of each transfer performed.The basic setup items should be set when, for example, the chip is initialized or USB-defined interfaces are switched over.

Note that the maximum number of interrupt transfers that can be set at the same time is 4.

Table 6.18 lists the transfer types that can be handled by each channel.

Table 6.18 Available Transfer Types

Channel	Available transfer type	Remark
CH0	Control transfer	Control transfer support function (described later) may be used.
CHa	Bulk transfer	Bulk Only support function (described later) may be used.
CHb, CHc CHd, CHe	Bulk transfer Interrupt transfer	

6.3.1.2 Control-only Channel

The LSI stipulated herein uses the control-only channel (CH0) to perform control transfers. Therefore, when performing a control transfer to or from multiple endpoints, it time-multiplexes the channel CH0 so that multiple packets can be sent on a single channel separated only in time.

Table 6.19 lists the basic setup items of the control-only channel (CH0).

Table 6.19 Basic Setup Items of the Control.only Channel

Item	Register/Bit	Description
Transfer rate	H_CH0Config_0.SpeedMode	Sets the transfer rate (HS, FS or LS) of the endpoint corresponding to channel CH0.
Toggle sequence bit	H_CH0Config_0.Toggle	Sets the initial value of a toggle sequence bit with which a transaction is initiated. While the transaction is underway and after the transaction is completed, it indicates the state of the toggle sequence bit.
Transaction type	H_CH0Config_1.PID	Sets the transaction type (SETUP, IN or OUT) issued on channel CH0.
Max. packet size	H_CH0MaxPktSize	Sets the Max. packet size to 8 for operation in LS mode, or to 8, 16, 32, or 64 for operation in FS mode. Or set it to 64 for operation in HS mode.
USB address	H_CH0FuncAdrs.FuncAdrs	Sets the USB address of the function (including an endpoint) managed by channel CH0 to any value between 0x0 and 0xF.
Endpoint number	H_CH0FuncAdrs.EP_Number	Sets the endpoint number of the endpoint corresponding to channel CH0 to any value between 0x0 and 0xF.
Hub address	H_CH0HubAdrs.HubAdrs	Sets the USB address of the hub that performs split transactions.
Port number	H_CH0HubAdrs.Port	Sets the IRP data quantity on channel CH0 in byte units.
Number of IRP data bytes	H_CH0TotalSize_H, H_CH0TotalSize _L	Sets the area allocated to channel CH0 by a FIFO address.
FIFO area	H_CH0StartAdrs_H, H_CH0StartAdrs _L, H_CH0EndAdrs_H, H_CH0EndAdrs _L	Make sure the allocated FIFO is equal to or greater than Max. packet size of channel CH0. Also be aware that the size of the FIFO area affects the throughput of data transfer. For details on how to allocate the FIFO area, refer to the relevant section in Chapter 6 on FIFOs.
Setup data	H_CH0SETUP_x(x=0-7)	Sets 8 bytes of data to be transmitted by a setup transaction.

6.3.1.3 General-purpose Channels

The general-purpose channels permit the transaction direction, USB address, and endpoint number to be set as desired, so that channels can be corresponded one for one to a maximum of five endpoints at the same time. As for the control-only channel, these channels can be time-multiplexed in IRP units, making it possible to perform transfers to or from many more endpoints than the maximum five.

Each channel has fixed basic setup items determined by the USB-defined interface and variable control items and status used for control of each transfer performed. The basic setup items should be set when, for example, the chip is initialized or USB-defined interfaces are switched over.

Table 6.20 lists the basic setup items of the general-purpose channels. Set up these times as appropriate for the contents of definitions of the USB-defined interface. Also, enable the settings made, to configure a USB-defined interface.

Table 6.20 Basic Setup Items of the General.purpose Channels

Item	Register/Bit	Description
Transfer rate	H_CHx{x=a-e}Config_0.SpeedMode	Sets the transfer rate (HS, FS or LS) of the endpoint corresponding to each channel.
Toggle sequence bit	H_CHx{x=a-e}Config_0.Toggle	Sets the initial value of a toggle sequence bit with which a transaction is initiated. While the transaction is underway and after the transaction is completed, it indicates the state of the toggle sequence bit.
Transaction type	H_CHx{x=a-e}Config_1.PID	Sets the transaction type (IN or OUT) issued on each channel.
Transfer type	H_CHx{x=b-e}Config_1.TranType	Sets the transfer type (bulk or interrupt) on each channel.
Toggle mode	H_CHx{x=b-e}Config_1.ToggleMode	Sets the operation mode of the toggle bit in interrupt transfer.
Max. packet size	H_CHx{x=a-e}MaxPktSize_H, H_CHx{x=a-e}MaxPktSize_L	Sets the Max. packet size on each channel to any value between 1 byte to 512 bytes.
Hub address	H_CHx{x=a-e}HubAdrs.HubAdrs	Sets the USB address of the hub that performs split transactions.
Port number	H_CHx{x=a-e}HubAdrs.Port	Sets the port number of the hub that performs split transactions.
USB address	H_CHx{x=a-e}FuncAdrs.FuncAdrs	Sets the USB address of the function (including an endpoint) managed by each channel to any value between 0x0 and 0xF.
Endpoint number	H_CHx{x=a-e}FuncAdrs.EP_Number	Sets the endpoint number of the endpoint corresponding to each channel to any value between 0x0 and 0xF.
Number of IRP data bytes	H_CHx{x=a-e}TotalSize_HH, H_CHx{x=a-e}TotalSize_HL, H_CHx{x=a-e}TotalSize_LH, H_CHx{x=a-e}TotalSize_LL	Sets the IRP data quantity on each channel in bytes.
Token issuance interval	H_CHx{x=b-e}Interval_H, H_CHx{x=b-e}Interval_L	Sets the interrupt (period) at which tokens are issued in interrupt transfer.
FIFO area	H_CHx{x=a-e}StartAdrs_H, H_CHx{x=a-e}StartAdrs_L, H_CHx{x=a-e}EndAdrs_H, H_CHx{x=a-e}EndAdrs_L	Sets the area allocated to each channel by a FIFO address. Make sure the allocated FIFO is equal to or greater than Max. packet size of each channel. Also be aware that the size of the FIFO area affects the throughput of data transfer. For details on how to allocate the FIFO area, refer to the relevant section in Chapter 6 on FIFOs.

6.3.1.4 Example for Using Channels

6.3.1.4.1 For One Storage Device Connected

Fig. 6.20 shows an example for using channels for the case where the system has a storage device compatible with USB Mass Storage Class (BulkOnly Transport Protocol) connected to it (e.g., a hard disk).

The bulk IN and bulk OUT transfers used in this class can be successively processed.

The IRP for control transfer uses CH0.

On the other hand, the IRP for bulk IN transfer and the IRP for bulk OUT transfer use CHa one after the other.

CHa has the function to automatically manage a series of Mass Storage Class (BulkOnly Transport Protocol) transports (see 6.3.8.), such as command transport (CBW), data transport, and status transport (CSW). This function helps to reduce the transfer processing load of the CPU and increase the efficiency of transfer.

Note that if the bulk-only support function does not need to be used, the IRP for bulk IN transfer and the IRP for bulk OUT transfer can be assigned separately to other general-purpose channels (e.g., CHb and CHc).

The hardware schedules transfers for the IRPs set on channels (see 6.3.2) and executes transactions.

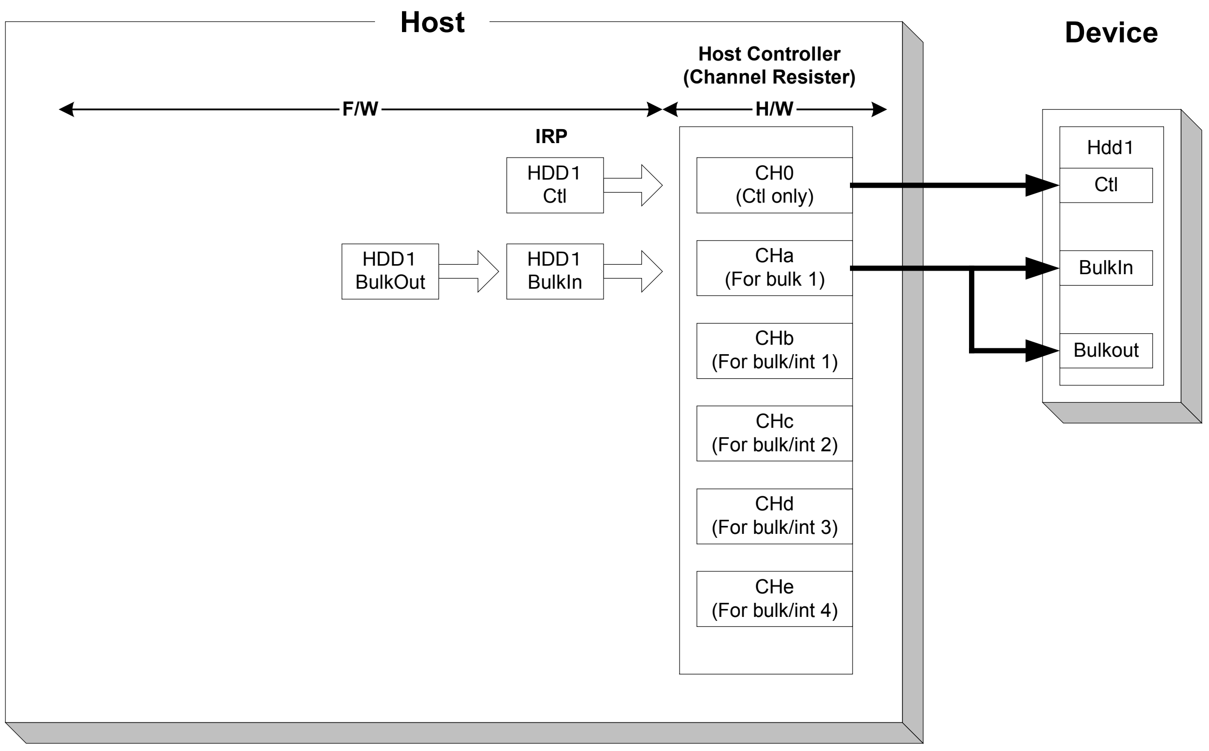


Fig. 6.20 Example for using channels (when the system has one storage device connected)

6.3.1.4.2 For One Communication Device Connected

Fig. 6.21 shows an example for using channels for the case where the system has a communication device compatible with USB Communication Device Class connected to it (e.g., a wireless LAN adapter).

The bulk IN and bulk OUT transfers used in this class can be processed in parallel.

The IRP for control transfer uses CH0.

The IRP for bulk IN transfer, the IRP for bulk OUT transfer, and the IRP for interrupt IN transfer use general-purpose channels (e.g., CHb, CHc, and CHd) that are assigned individually to each.

The hardware schedules transfers for the IRPs set on channels and executes transactions (see 6.3.2).

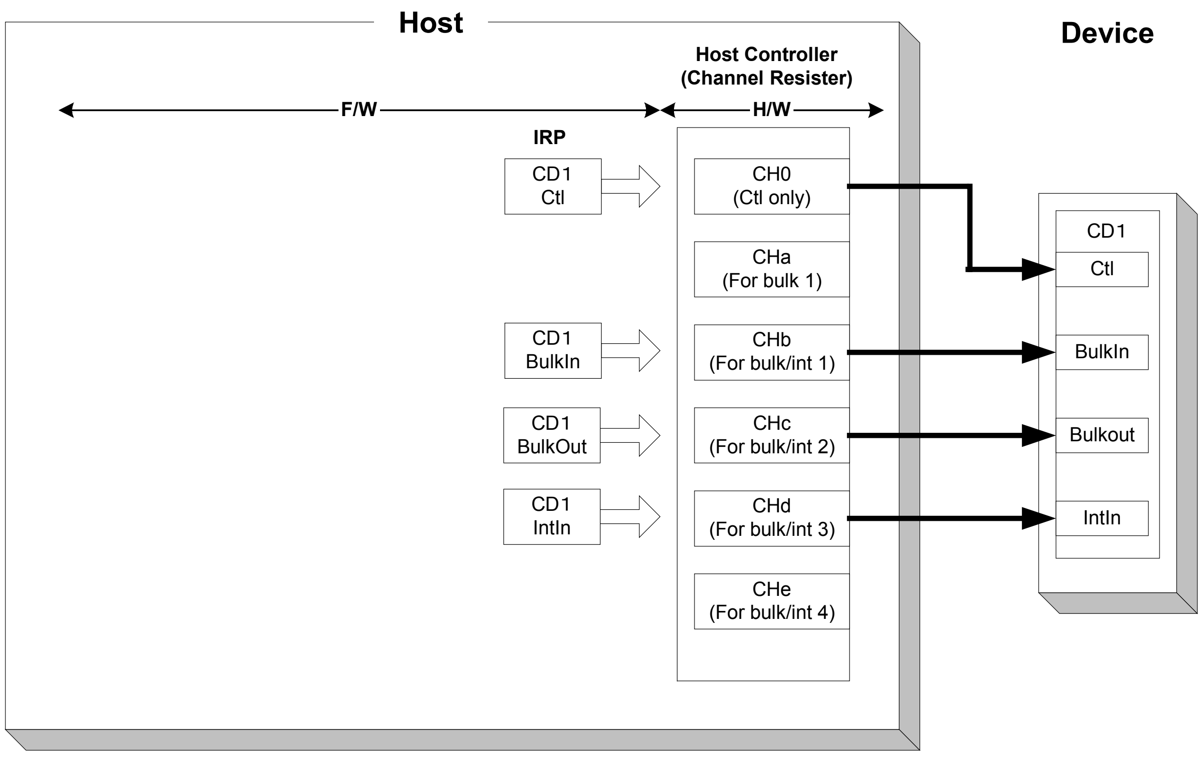


Fig. 6.21 Example for using channels
(when system has one communication device connected)

6.3.1.4.3 For One Human Interface Device Connected

Fig. 6.22 shows an example for using channels for the case where the system has a human interface device compatible with USB Human Interface Device Class connected to it (e.g., a mouse).

The IRP for control transfer uses CH0.

The IRP for interrupt IN transfer uses a general-purpose channel that is assigned to it (e.g., CHd).

The hardware schedules a transfer for the IRP set on a channel and executes a transaction (see 6.3.2).

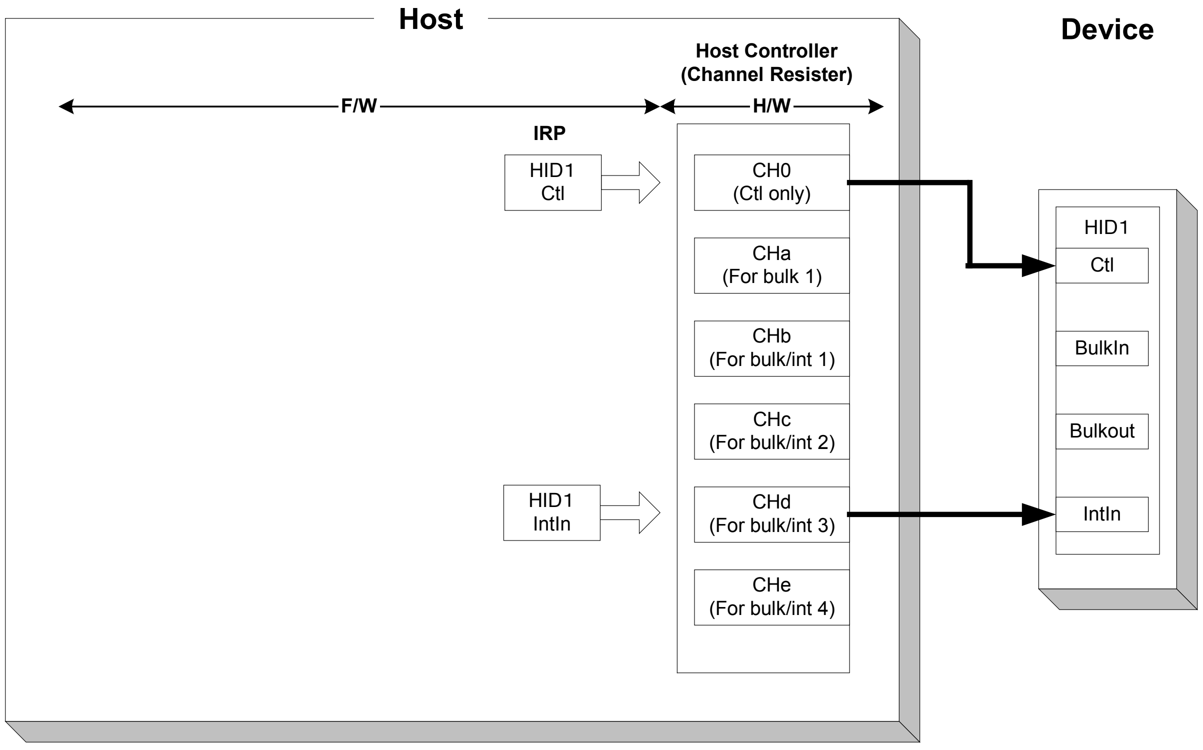


Fig. 6.22 Example for using channels (when system has one human interface device connected)

6.3.1.4.4 For Two Storage Devices Connected via Hub

Fig. 6.23 shows an example for using channels for the case where the system has two storage devices compatible with USB Mass Storage Class (BulkOnly Transport Protocol) connected to it via a hub (e.g., a hard disk and USB memory).

The bulk IN and bulk OUT transfers used in this class can be successively processed.

The IRP for control transfer uses CH0.

The IRP for interrupt IN transfer uses general-purpose channels that are assigned to it (e.g., CHd and CHe).

On the other hand, the IRP for bulk IN transfer and the IRP for bulk OUT transfer use CHa one after the other.

CHa has the function to automatically manage a series of Mass Storage Class (BulkOnly Transport Protocol) transports (see 6.3.8.), such as command transport (CBW), data transport, and status transport (CSW). This function helps to reduce the transfer processing load of the CPU and increase the efficiency of transfer.

Note that if the bulk-only support function need not to be used, the IRP for bulk IN transfer and the IRP for bulk OUT transfer can be assigned separately to other general-purpose channels (e.g., CHb and CHc).

The hardware schedules transfers for the IRPs set on channels and executes transactions (see 6.3.2).

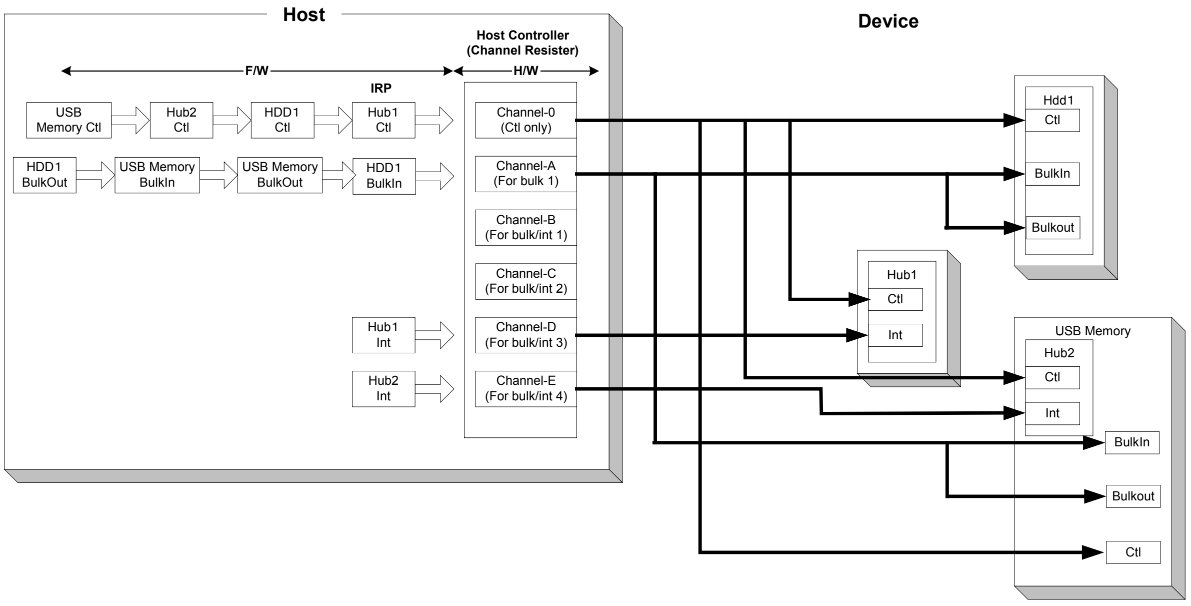


Fig. 6.23 Example for using channels (when system has two storage devices connected via a hub)

6.3.2 Scheduling

The hardware selects one of the channels which have had transfer executions set (H_CHx{x=0,a-e}Config_0.TranGo), and determines whether the transfer set for that channel can be executed. When found executable, it executes one transaction according to settings made. When the transaction finishes, the hardware selects another channel, determines whether execution is possible, and executes a transaction in the same way.

That way, by selecting a channel, determining whether execution is possible, and executing a transaction repeatedly, the hardware performs transfers to or from multiple endpoints.

Table 6.21 lists the control items relating to scheduling control on channel CH0.

Table 6.21 Setup Items for Scheduling on Channel CH0

Item	Register/Bit	Description
Transfer execution	H_CH0Config_0.TranGo	Sets transfer execution on channel CH0. Performs a transfer according to settings made on channel CH0.

Table 6.22 lists the control items relating to scheduling control on general-purpose channels (CHa, CHb, CHc, CHd, and CHe).

Table 6.22 Setup Items for Scheduling on General.purpose Channels

Item	Register/Bit	Description
Transfer execution	H_CHx{x=a-e}Config_0.TranGo	Sets transfer execution on each channel. Performs a transfer according to settings made on each channel.

6.3.3 Transactions

The LSI provides transaction execution functions in hardware and provides the firmware with the interfaces necessary to execute transactions. The interfaces for the firmware are implemented as control and status registers and the interrupt signals that are asserted by a status. For details about settings necessary to assert an interrupt by status, refer to the relevant section on registers.

The LSI issues a status to the firmware for each transaction performed. However, the firmware does not always need to manage each individual transaction.

For an IN channel, for example, the firmware can read data from the FIFO through the CPU interface (DMA read or register read) to create a free space in the FIFO, thereby allowing IN transactions to be automatically executed in succession.

For an OUT channel also, the firmware can write data to the FIFO through the CPU interface (DMA write or register write) to create valid data in the FIFO, thereby allowing OUT transactions to be automatically executed in succession.

Table 6.23 lists the control items and status relating to the transaction control on channel CH0.

Table 6.23 Control Items and status for Channel CH0

Item	Register/Bit	Description
Transaction status	H_CH0IntStat. TotalSizeCmp, H_CH0IntStat.TranACK, H_CH0IntStat.TranErr, H_CH0IntStat.ChangeCondition	Shows the result of transaction.
Transaction condition code	H_CH0ConditionCode	Shows the result of transaction in detail.

Table 6.24 lists the control items and status relating to the transaction control on general-purpose channels (CHa, CHb, CHc, CHd, and CHe).

Table 6.24 Control Items and status for General.purpose Channels

Item	Register/Bit	Description
Transaction status	H_CHx{x=a-e}IntStat.TotalSizeCmp, H_CHx{x=a-e}IntStat.TranACK, H_CHx{x=a-e}IntStat.TranErr, H_CHx{x=a-e}IntStat.ChangeCondition	Shows the result of transaction.
Transaction condition code	H_CHx{x=a-e}ConditionCode	Shows the result of transaction in detail.

6.3.3.1 SETUP Transactions

In the CH0 basic setup register, set the transaction type (H_CH0Config_1.PID) to SETUP. Set other basic setup items as accordingly, write setup data (8 bytes) to the H_CH0SETUP_0–7 registers, and set transfer execution (H_CH0Config_0.TranGo). The channel will thereby be recognized as the target for which USB transfer is to be scheduled by the hardware, so that when this channel is selected, the hardware determines the remaining frame time and executes a SETUP transaction.

In SETUP transactions, the data in the H_CH0_SETUP_0–7 registers are used, where the data packets are 8 bytes in length.

When ACK was received for a SETUP transaction, the hardware issues an ACK status (H_CH0IntStat.TranACK bit) to the firmware.

If correct response was not received for a SETUP transaction, the hardware performs a retry process. If the error continues three times consecutively, the hardware terminates the transfer by automatically clearing H_CH0Control.TranGo and sets condition code (H_CH0ConditionCode) accordingly. It then issues a ChangeCondition status (H-CH0IntStat.ChangeCondition bit) to the firmware.

Fig. 6.24 shows how a SETUP transaction will be performed. In (a), the LSI issues a SETUP token addressed to the endpoint 0 that resides in the local node. In (b), the LSI proceeds to send a data packet of 8 bytes in length. In (c), upon receiving ACK, the LSI automatically sets the relevant register and issues a status to the firmware.

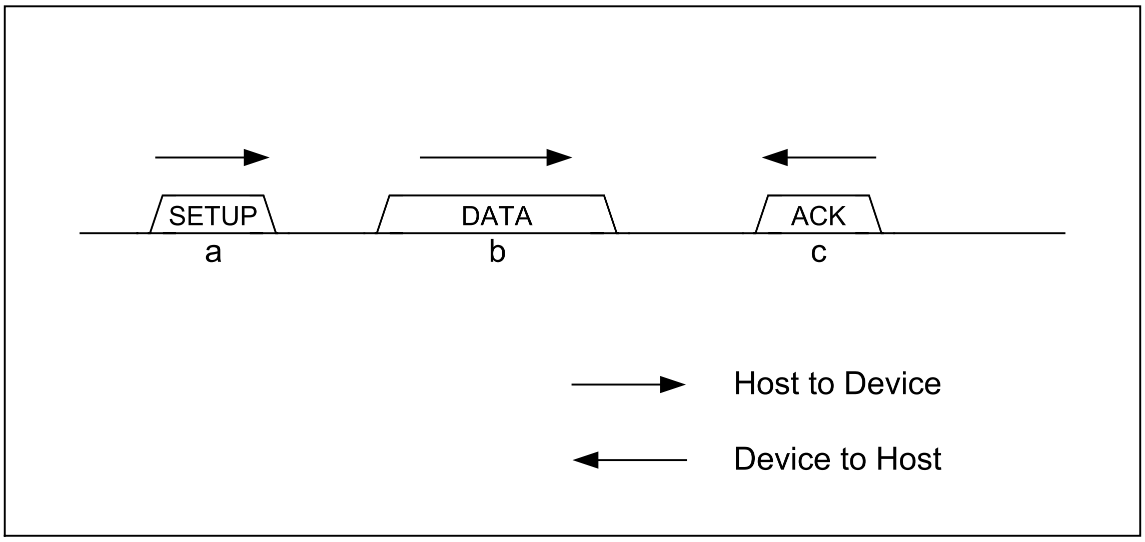


Fig. 6.24 SETUP transaction

6.3.3.2 Bulk OUT Transaction

In the CHx basic setup register, set the transfer type (H_CHx{x=b-e}Config_1.TranType) to Bulk and the transaction type (H_CHx{x=a-e}Config_1.TID) to OUT. Set other basic setup items as accordingly, and set transfer execution (H_CHx{x=a-e}Config_0.TranGo). The channel will thereby be recognized as the target for which USB transfer is to be scheduled by the hardware, so that when this channel is selected, the hardware determines the number of valid data bytes in the FIFO and the remaining frame time and thereby executes a bulk OUT transaction.

The data length of a data packet is H_CHx{x=a-e}MaxPacketSize_H,L or H_CHx{x=a-e}TotalSize_HH,HL,LH,LL whichever is smaller.

When ACK was received for a bulk OUT transaction, the hardware issues an ACK status (H_CHx{x=a-e}IntStat.TranACK bit) to the firmware. It also updates the FIFO, and assuming the transmitted data as having been transmitted, frees the reserved area.

If NAK was received for a bulk OUT transaction, the hardware does not update the FIFO, nor does it free the reserved area. Therefore, if the channel is selected again, the hardware executes the same transaction.

If STALL was received for a bulk OUT transaction, the hardware terminates the transfer by automatically clearing H_CHx{x=0,a-e}Config_0.TranGo and sets the condition code (H_CHx{x=a-e}ConditionCode) to STALL. It then issues a ChangeCondition status (H-CHx{x=a-e}IntStat.ChangeCondition bit) to the firmware. The FIFO is not updated, nor is the reserved area freed.

If correct response was not received for a bulk OUT transaction, the hardware neither updates the FIFO nor frees the reserved area, sets the condition code (H_CHx{x=a-e}ConditionCode) to RetryError, and issues a TranErr status (H-CHx{x=a-e}IntStat.TranErr bit) to the firmware. It then performs a retry process. If the error continues three times consecutively, the hardware terminates the transfer by automatically clearing H_CHx{x=a-e}Control.TranGo and sets condition code

(H_CHx{x=a-e}ConditionCode) as accordingly. It then issues a ChangeCondition status (H-CHx{x=a-e}IntStat.ChangeCondition bit) to the firmware.

Fig. 6.25 shows how a bulk OUT transaction will be performed for the case where the transaction finishes without errors. In (a), the LSI issues an OUT token addressed to the OUT-direction endpoint that resides in the local node. In (b), the LSI proceeds to send a data packet within Max. packet size. In (c), upon receiving ACK, the LSI automatically sets the relevant register and issues a status to the firmware.

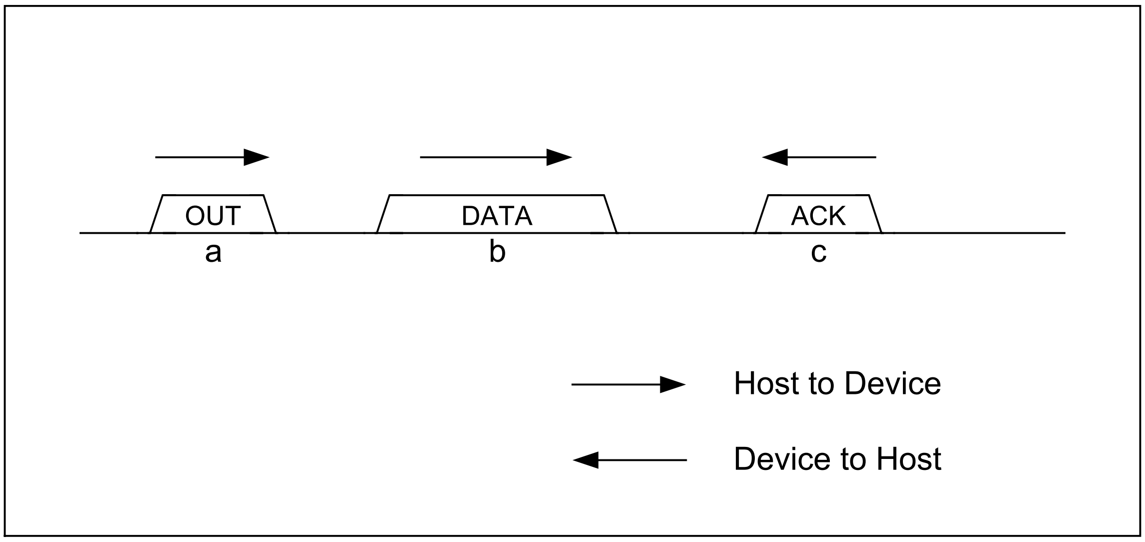


Fig. 6.25 OUT transaction

6.3.3.3 Interrupt OUT Transaction

In the CHx basic setup register, set the transfer type (H_CHx{x=b-e}Config_1.TranType) to Interrupt and the tokoen type (H_CHx{x=b-e}Config_1.TID) to OUT. Furthermore, set a token issuance interval (H_CHx{x=b-e}Interval_H,L), and after setting other basic setup items as accordingly, set transfer execution (H_CHx{x=b-e}Config_0.TranGo). The channel will thereby be recognized as the target for which USB transfer is to be scheduled by the hardware, so that when this channel is selected, the hardware determines the token issuance interval (H_CHx{x=b-e}Interval_H,L), the number of valid data bytes in the FIFO, and the remaining frame time and thereby executes an interrupt OUT transaction.

The data length of a data packet is H_CHx{x=b-e}MaxPacketSize_H,L or H_CHx{x=b-e}TotalSize_HH,HL,LH,LL whichever is smaller.

When ACK is received for an interrupt OUT transaction, the hardware issues an ACK status (H_CHx{x=b-e}IntStat.TranACK bit) to the firmware. It also updates the FIFO, and assuming the transmitted data as having been transmitted, frees the reserved area.

If NAK is received for an interrupt OUT transaction, the hardware does not update the FIFO, nor does it free the reserved area. Therefore, if the channel is selected again, the hardware executes the same transaction.

If STALL is received for an interrupt OUT transaction, the hardware terminates the transfer by automatically clearing H_CHx{x=b-e}Config_0.TranGo and sets the condition code (H_CHx{x=b-e}ConditionCode) to STALL. It then issues a ChangeCondition status (H-CHx{x=b-e}IntStat.ChangeCondition bit) to the firmware. The FIFO is not updated, nor is the reserved area freed.

If correct response was not received for an interrupt OUT transaction, the hardware neither updates the FIFO nor frees the reserved area, sets the condition code (H_CHx{x=a-e}ConditionCode) to RetryError, and issues a TranErr status (H-CHx{x=b-e}IntStat.TranErr bit) to the firmware. It then performs a retry process. If the error continues three times consecutively, the hardware terminates the transfer by automatically clearing H_CHx{x=b-e}Control.TranGo and sets condition code (H_CHx{x=b-e}ConditionCode) as accordingly. It then issues a ChangeCondition status (H-CHx{x=b-e}IntStat.ChangeCondition bit) to the firmware.

6.3.3.4 Bulk IN Transaction

In the CHx basic setup register, set the transfer type (H_CHx{x=b-e}Config_1.TranType) to Bulk and the transaction type (H_CHx{x=a-e}Config_1.TID) to IN. Set other basic setup items as accordingly, and set transfer execution (H_CHx{x=a-e}Config_0.TranGo). The channel will thereby be recognized as the target for which USB transfer is to be scheduled by the hardware, so that when this channel is selected, the hardware determines the free space in the FIFO and the remaining frame time and thereby executes a bulk IN transaction.

The expected data length of the data packet to be received is H_CHx{x=a-e}MaxPacketSize_H,L or H_CHx{x=a-e}TotalSize_HH,HL,LH,LL whichever is smaller.

When all data bytes are received correctly in a bulk IN transaction, the hardware responds with ACK to complete the transaction. It also issues an ACK status (H_CHx{x=a-e}IntStat.TranACK bit) to the firmware. It further updates the FIFO, and assuming the data as having been received, reserves an area.

If the received data length in a bulk IN transaction is less than the expected data length, the hardware terminates the transfer by automatically clearing H_CHx{x=a-e}Config_0.TranGo and responds with ACK. It also sets the condition code (H_CHx{x=a-e}ConditionCode) to DataUnderrun and then issues a ChangeCondition status (H_CHx{x=a-e}IntStat.ChangeCondition bit) to the firmware. It further updates the FIFO, and assuming the data has been received, reserves an area.

If NAK is received in a bulk IN transaction, the hardware does not issue a status. Nor does it update the FIFO.

If STALL is received in a bulk IN transaction, the hardware terminates the transfer by automatically clearing H_CHx{x=a-e}Config_0.TranGo and sets the condition code (H_CHx{x=a-e}ConditionCode) to STALL. It then issues a ChangeCondition status (H_CHx{x=a-e}IntStat.ChangeCondition bit) to the firmware. The FIFO is not updated.

If the received data length in a bulk IN transaction is larger than the expected data length, the hardware terminates the transfer by automatically clearing H_CHx{x=a-e}Config_0.TranGo. It does not respond. It also sets the condition code (H_CHx{x=a-e}ConditionCode) to DataOverrun, and issues a ChangeCondition status (H_CHx{x=a-e}IntStat.ChangeCondition bit) to the firmware. The FIFO is not updated.

When a toggle mismatch occurs in a bulk IN transaction, the hardware responds with ACK. It also sets the condition code (H_CHx{x=a-e}ConditionCode) to RetryError, and issues a TranErr status (H_CHx{x=a-e}IntStat.TranErr bit) to the firmware. The FIFO is not updated.

When a time-out error, CRC error, bit stuffing error, or PID error (including unexpected PID) occurs in a bulk IN transaction, the hardware does not respond. It sets the condition code (H_CHx{x=a-e}ConditionCode) to RetryError, and issues a TranErr status (H_CHx{x=a-e}IntStat.TranErr bit) to the firmware. The FIFO is not updated.

When an error that would cause the condition code (H_CHx{x=a-e}ConditionCode) to be set to RetryError occurs, the hardware performs a retry process. If the error continues three times consecutively, the hardware terminates the transfer by automatically clearing H_CHx{x=a-e}Control.TranGo and issues a ChangeCondition status (H_CHx{x=a-e}IntStat.ChangeCondition bit) to the firmware.

Fig. 6.26 shows how a bulk IN transaction will be performed for the case where the transaction finishes without errorswithout errors. In (a), the LSI issues an IN token addressed to the IN-direction endpoint that resides in the local node. In (b), if the endpoint can respond to this IN transaction, it sends a data packet within Max. packet size. In (c), the LSI responds with ACK. It then automatically sets the relevant register and issues a status to the firmware.

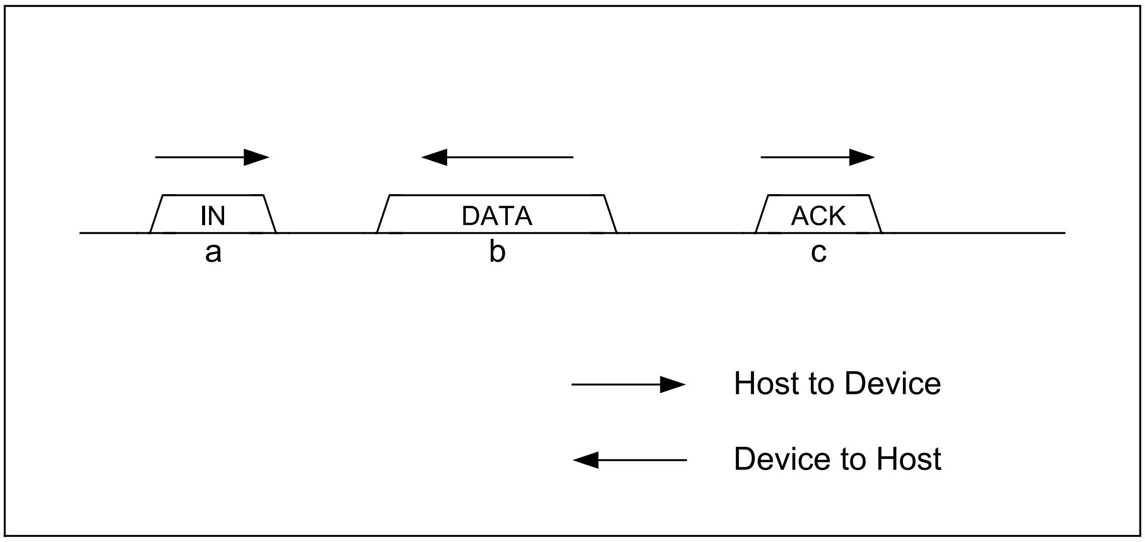


Fig. 6.26 IN transaction

6.3.3.5 Interrupt IN Transaction

In the CHx basic setup register, set the transfer type (H_CHx{x=b-e}Config_1.TranType) to Interrupt and the transaction type (H_CHx{x=b-e}Config_1.TID) to IN. In addition, set a token issuance interval (H_CHx{x=b-e}Interval_H,L), and after setting other basic setup items as accordingly, set transfer execution (H_CHx{x=b-e}Config_0.TranGo). The channel will thereby be recognized as the target for which USB transfer is to be scheduled by the hardware, so that when this channel is selected, the hardware determines the token issuance interval (H_CHx{x=b-e}Interval_H,L), the free space in the FIFO, and the remaining frame time and thereby executes an interrupt IN transaction.

The expected data length of the data packet to be received is H_CHx{x=b-e}MaxPacketSize_H,L or H_CHx{x=b-e}TotalSize_HH,HL,LH,LL whichever is smaller.

When all data bytes are received correctly in an interrupt IN transaction, the hardware responds with ACK to complete the transaction. It also issues an ACK status (H_CHx{x=b-e}IntStat.TranACK bit) to the firmware. It further updates the FIFO, and assuming the data has been received, reserves an area.

If the received data length in an interrupt IN transaction is larger than the expected data length, the hardware terminates the transfer by automatically clearing H_CHx{x=b-e}Config_0.TranGo and responds with ACK. It also sets the condition code (H_CHx{x=b-e}ConditionCode) to DataUnderrun and then issues a ChangeCondition status (H_CHx{x=b-e}IntStat.ChangeCondition bit) to the firmware. It further updates the FIFO, and assuming the data has been received, reserves an area.

If NAK is received in an interrupt IN transaction, the hardware does not issue a status. Nor does it update the FIFO. The next transaction is performed in the next cycle.

If STALL is received in an interrupt IN transaction, the hardware terminates the transfer by automatically clearing H_CHx{x=b-e}Config_0.TranGo and sets the condition code (H_CHx{x=b-e}ConditionCode) to STALL. It then issues a ChangeCondition status (H_CHx{x=b-e}IntStat.ChangeCondition bit) to the firmware. The FIFO is not updated.

If the received data length in an interrupt IN transaction is larger than the expected data length, the hardware terminates the transfer by automatically clearing H_CHx{x=b-e}Config_0.TranGo. It does not respond. It also sets the condition code (H_CHx{x=b-e}ConditionCode) to DataOverrun, and issues a ChangeCondition status (H_CHx{x=b-e}IntStat.ChangeCondition bit) to the firmware. The FIFO is not updated.

If a toggle mismatch occurrs in an interrupt IN transaction, the hardware responds with ACK. It also sets the condition code (H_CHx{x=b-e}ConditionCode) to RetryError, and issues a TranErr status (H_CHx{x=b-e}IntStat.TranErr bit) to the firmware. The FIFO is not updated.

If a time-out error, CRC error, bit stuffing error, or PID error (including unexpected PID) occurrs in an interrupt IN transaction, the hardware does not respond. It sets the condition code (H_CHx{x=b-e}ConditionCode) to RetryError, and issues a TranErr status (H_CHx{x=b-e}IntStat.TranErr bit) to the firmware. The FIFO is not updated.

If an error occurrs that causes the condition code (H_CHx{x=b-e}ConditionCode) to be set to RetryError, the hardware performs a retry process in the next cycle. If the error continues three times consecutively, the hardware terminates the transfer by automatically clearing H_CHx{x=b-e}Control.TranGo and issues a ChangeCondition status (H_CHx{x=b-e}IntStat.ChangeCondition bit) to the firmware.

6.3.3.6 PING Transaction

On channels in which bulk OUT transactions or control OUT transactions are performed, the LSI executes a PING transaction when it is operating in HS mode.

If NYET or NAK or no response is received for an OUT transaction, the hardware shifts to a state in which it can execute a PING transaction.

If ACK is received for a PING transaction, the hardware returns to a state in which it can execute an OUT transaction. It does not issue a status.

If NAK is received for a PING transaction, the hardware remains in a state in which it can execute a PING transaction. It does not issue a status to the firmware.

If STALL is received for a PING transaction, the hardware terminates the transfer by automatically clearing H_CHx{x=0,a-e}Config_0. TranGo and sets the condition code (H_CHx{x=0,a-e}ConditionCode) to STALL. It then issues a ChangeCondition status (H_CHx{x=0,a-e}IntStat.ChangeCondition bit) to the firmware.

If correct response is not received for a PING transaction, the hardware sets the condition code (H_CHx{x=0,a-e}ConditionCode) to RetryError, and issues a TranErr status (H_CHx{x=0,a-e}IntStat.TranErr bit) to the firmware. In this case, it performs a retry process. If the error continues three times consecutively, the hardware terminates the transfer by automatically clearing H_CHx{x=0,a-e}Control.TranGo and issues a ChangeCondition status (H_CHx{x=0,a-e}IntStat.ChangeCondition bit) to the firmware.

In no event will the FIFO be updated in a PING transaction.

Fig. 6.27 shows how a PING transaction will be responded with ACK. In (a), the LSI issues a PING token addressed to the OUT-direction endpoint that resides in the local node. In (b), if the endpoint has a free space equivalent to Max. packet size, the device responds to this PING transaction with ACK.

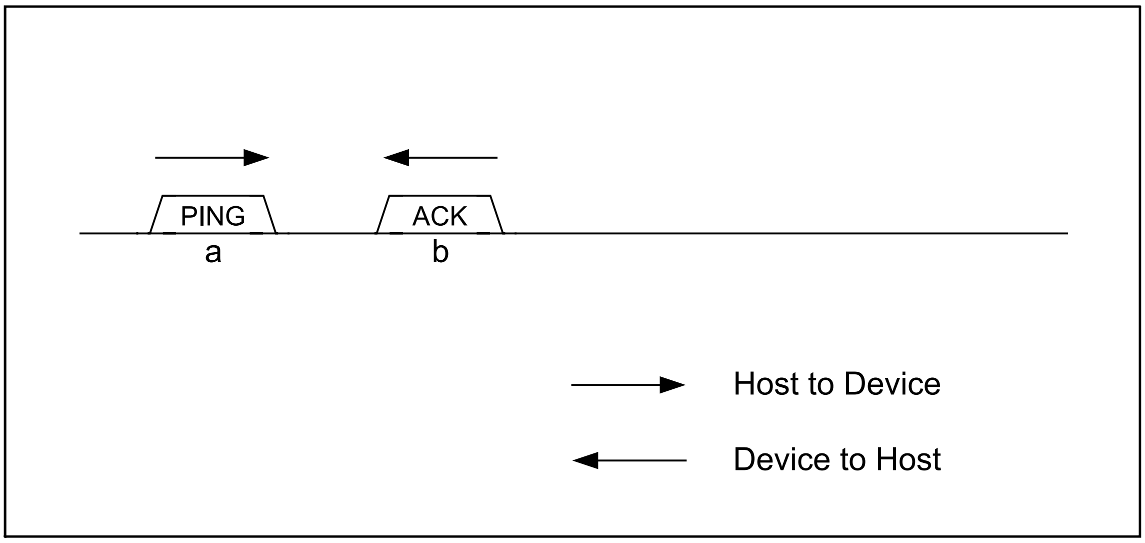


Fig. 6.27 PING transaction

6.3.3.7 Low-Speed (LS) Transaction

Transfers to or from LS devices are accomplished using control or interrupt transfers.

If the downstream port has an LS device connected, the host operates in LS mode. The host sets the transfer rate for the channel used (H_CHx{x=0,a-e}Config_0.SpeedMode) to LS, and thereby executes a transaction with LS bit time.

On the other hand, if the downstream port has a full-speed (FS) hub connected and the downstream port of the hub has a LS device connected, the host operates in FS mode. The host sets the transfer rate for the channel used (H_CHx{x=0,a-e}Config_0.SpeedMode) to LS, and thereby transmits downstream packets to the corresponding endpoint after attaching a preamble at the beginning of all packets. The preamble is transmitted with FS bit time, and the downstream packets following it are transmitted with LS bit time.

Fig. 6.28 shows how an interrupt OUT transaction will be performed for the case in which the transaction finishes without errorswithout errors. In (a), the LSI issues an OUT token addressed to the OUT-direction endpoint that resides in the local node after attaching a preamble at the beginning of it. In (b), the LSI transmits a data packet within Max. packet size after attaching a preamble at the beginning of it. In (c), upon receiving ACK, the LSI automatically sets the relevant register and issues a status to the firmware.

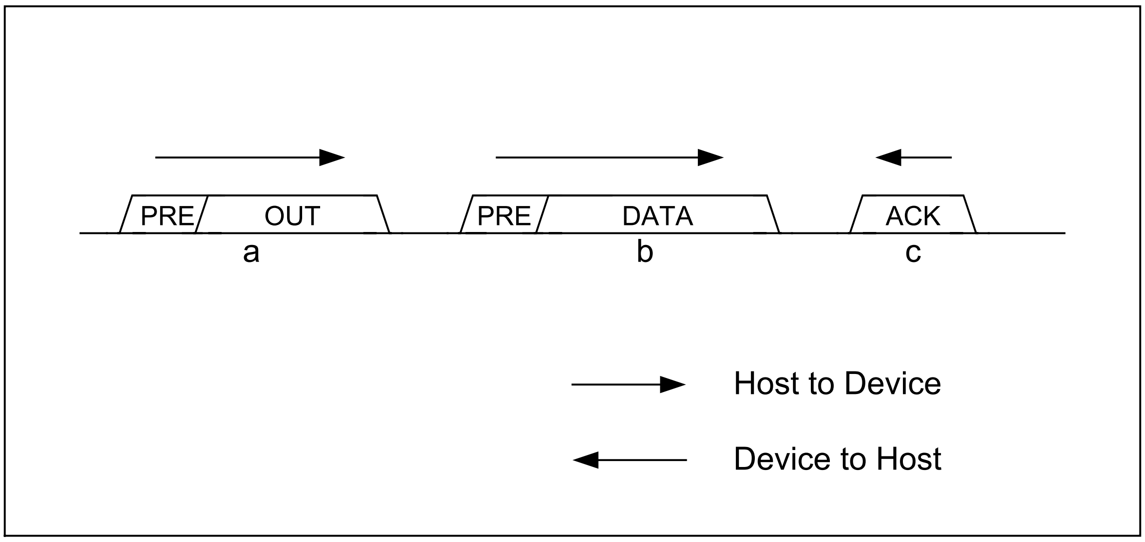


Fig. 6.28 OUT transaction with preamble attached

Fig. 6.29 shows how an interrupt IN transaction will be performed for the case where the transaction finishes without errors. In (a), the LSI issues an IN token addressed to the IN-direction endpoint that resides in the local node after attaching a preamble at the beginning of it. In (b), the device sends a data packet within Max. packet size. The LSI writes this data to the FIFO of the relevant channel. In (c), the LSI returns ACK for response after attaching a preamble at the beginning of it. It then automatically sets the relevant register and issues a status to the firmware.

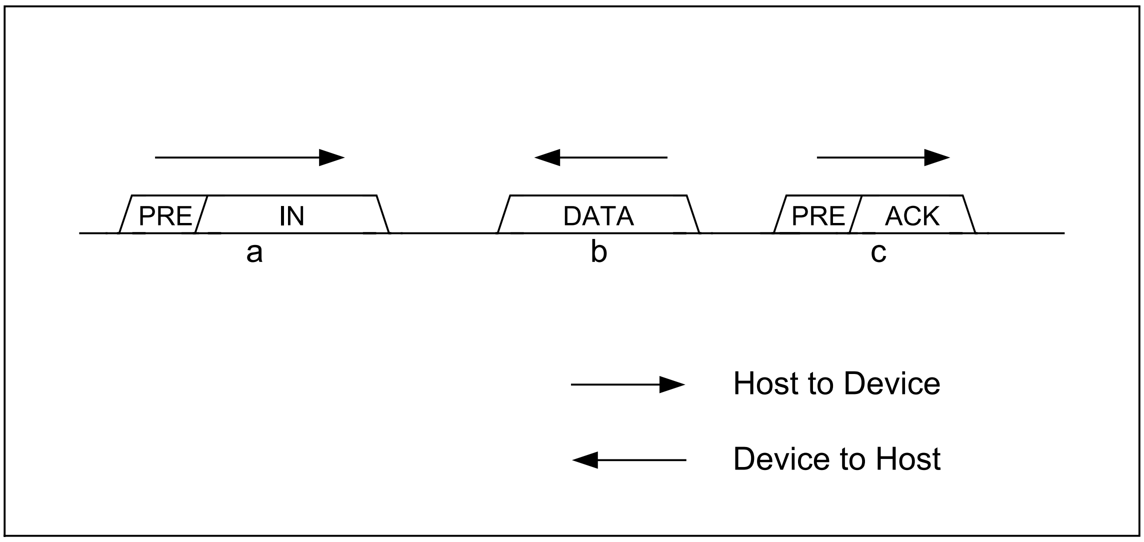


Fig. 6.29 IN transaction with preamble attached

6.3.3.8 Split Transactions

If the downstream port has a high-speed (HS) hub connected and the downstream port of the hub has an FS or LS device connected, the host operates in HS mode. The host sets the transfer rate for the channel used (H_CHx{x=0,a-e}Config_0.SpeedMode) to FS or LS, and thereby executes a transaction for the corresponding endpoint in split transactions to or from the hub.

For the relevant channel, set the hub address (H_CHx{x=0,a-e}HubAdrs.HubAdrs) and port number (H_CHx{x=0,a-e}HubAdrs.Port) to appropriate values.

The sequence of start split transaction through complete split transactions in split transactions is controlled by the hardware. The individual transactions in split transactions will not affect the firmware.

If the last complete split transaction in the sequence of start split transaction through complete split transactions finishes without errors, the hardware issues an ACK status (H_CHx{x=0,a-e}IntStat.TranACK bit) to the firmware and updates the FIFO.

On the other hand, for individual transactions other than the last complete split transaction, no status is issued to the firmware for their successful completion.

If an error occurrs in any individual transaction in the sequence of start split transaction through complete split transactions, the hardware sets the condition code (H_CHx{x=0,a-e}ConditionCode) to RetryError and issues a TranErr status (H_CHx{x=0,a-e}IntStat.TranErr bit) to the firmware. In this case, the hardware does not update the FIFO, and performs a retry process. If the error continues three times consecutively, the hardware terminates the transfer by automatically clearing H_CHx{x=0,a-e}Control.TranGo and issues a ChangeCondition status (H_CHx{x=0,a-e}IntStat.ChangeCondition bit) to the firmware.

6.3.4 Control Transfer

Each stage of the transfer in a control transfer is controlled as an individual transaction.

Fig. 6.30 shows how a control transfer is controlled. The firmware sets transactions “a”, “b”, and “c” as accordingly, thereby allowing the hardware to execute a control transfer.

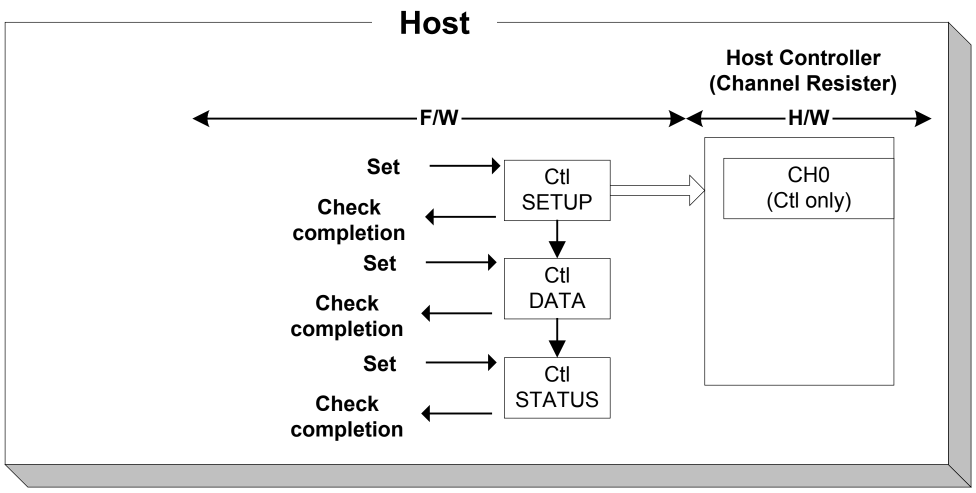


Fig. 6.30 Controlling a control transfer

Fig. 6.31 shows how a control transfer is performed when the data stage is directed for OUT. In (a), the host starts a control transfer via a SETUP transaction. In (b), the host issues an OUT transaction to execute a data stage. In (c), the host issues an IN transaction to execute a status stage.

For control transfers without data stages, the operation is executed without performing the data stage described in this example.

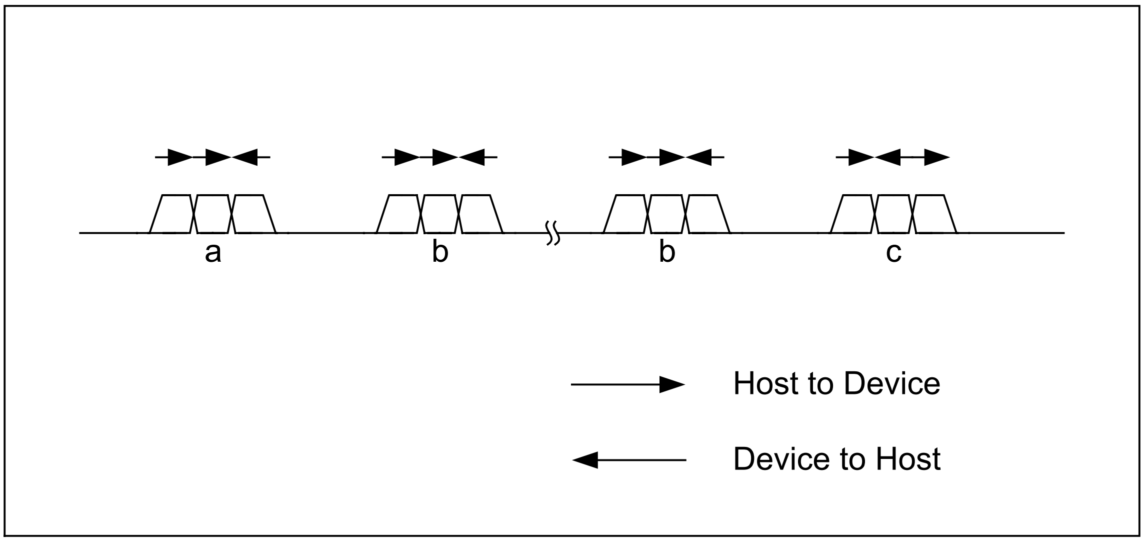


Fig. 6.31 Control transfer when the data stage is directed for OUT

Fig. 6.32 shows how a control transfer is performed when the data stage is directed for IN. In (a), the host starts a control transfer via a SETUP transaction. In (b), the host issues an IN transaction to execute a data stage. In (c), the host issues an OUT transaction to execute a status stage.

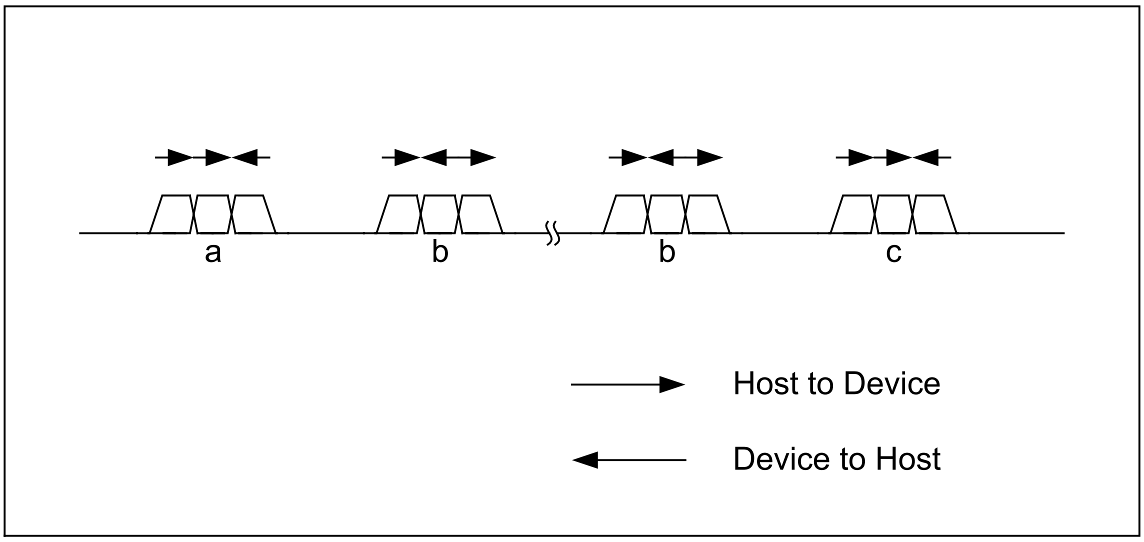


Fig. 6.32 Control transfer when the data stage is directed for IN

6.3.4.1 Setup Stage

The setup stage is executed by a setup transaction. For details, refer to the relevant section in Chapter 6 on setup transactions.

6.3.4.2 Data Stage and Status Stage

After the setup stage is complete, go to the next stage.

If the next stage is for an IN direction, set the transaction type (H_CH0Config_1.PID) to IN, and then set other basic setup registers as accordingly to execute a transaction.

On the other hand, if the next stage is for an OUT direction, set the transaction type (H_CH0Config_1.PID) to OUT, and then set other basic setup registers as accordingly to execute a transaction.

Note that if a status stage is that is executed, set the number of IRP data bytes (H_CH0TotalSize_H,L) to 0 before executing a transaction.

6.3.4.3 Control Transfer Support Function

The LSI stipulated herein has the function to automatically manage a series of stages executed in a control transfer. Use of this function relieves the firmware from the burden of managing each stage as an individual transaction.

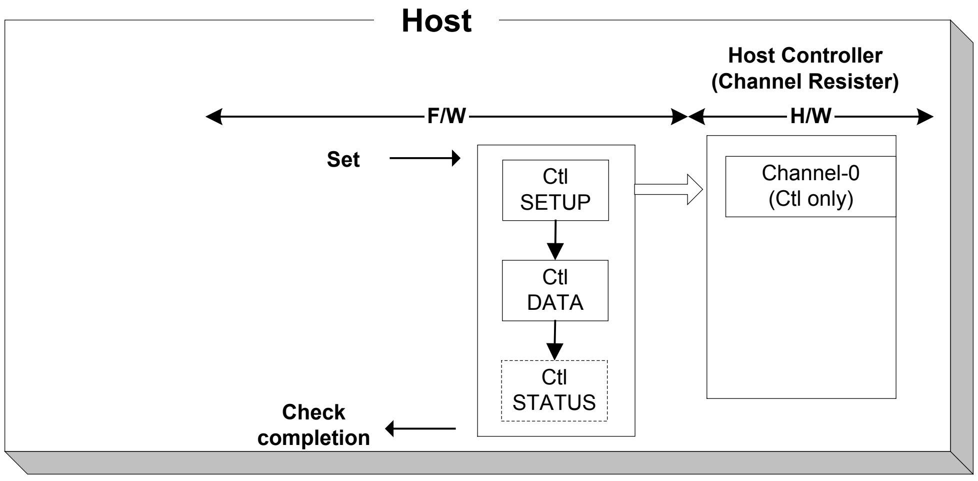


Fig. 6.33 Control by the control transfer support function

The control transfer support function is available only for channel CH0. When this function is used, a control transfer is performed in the manner shown below. Note that the firmware performs processes (1) to (4), and (7).

(1) Set the following basic setup registers for channel CH0 as accordingly.
Transfer rate (H_CH0Config_0.SpeedMode), max packet size (H_CH0MaxPktSize), USB address (H_CH0FuncAdrs.FuncAdrs), endpoint number (H_CH0FuncAdrs.EP_Number), FIFO area (H_CH0StartAdrs,H_CH0EndAdrs)

(2) Write setup data (8 bytes) to the setup registers (H_CH0SETUP_0–7).

(3) If the data stage is directed for OUT, write the transmit data to the CH0 area of the FIFO. If the data stage is directed for IN, create a free space equal to greater than MaxPacketSize in the CH0 area of the FIFO.

(4) Set the control transfer support execution (H_CTL_SupportControl.CTL_SupportGo).
At this point, set the control transfer stage (H_CTL_SupportControl.CTL_SupportState) to Idle (00b) by writing that value to the H_CTL_SupportControl register.

(5) A SETUP transaction (setup stage) is executed using the data (8 bytes) in the SETUP register.

(6) A data stage is executed based on the SETUP data.

- If bmRequestType in bit 7 of the SETUP data = 0, the data present in the channel CH0 area of the FIFO is transmitted by a bulk OUT transaction (OUT-direction data stage).
- If bmRequestType in bit 7 of the SETUP data = 1, a bulk IN transaction is issued and the received data is written to the channel CH0 area of the FIFO (IN-direction data stage).
- A data stage is executed by performing transactions for the number of data bytes indicated by wLength of the SETUP data.
- A data stage when it is directed for IN is complete even when the data size of the received data is less than that specified by wLength of the SETUP data.
- A data stage in which wLength of the SETUP data = 0x0000 is not executed.

(7) If the CH0 area of the FIFO is smaller than the value indicated by wLength of the SETUP data, the firmware should divide the data before processing at a data stage.

- If the data stage is of the OUT-direction, no transactions are issued when no more transmit data exists in the FIFO. The firmware should check for free space in the FIFO when it writes remaining transmit data sequentially to the FIFO.
- If the data stage is of the IN-direction, no transactions are issued if no more free space exists in the FIFO. The firmware should check the valid data bytes in the FIFO as it reads received data sequentially from the FIFO to create FIFO free space.

(8) A status stage is executed based on the SETUP data.

- If the status stage is directed for OUT, a bulk IN transaction is issued (IN-direction status stage).
- If the status stage is directed for IN, a bulk OUT transaction with packet length = zero is issued (OUT-direction status stage).

(9) When the control transfer is completed without errors, the control transfer support execution (H_CTL_SupportControl.CTL_SupportGo) is automatically cleared and a control transfer complete status (H_CH0IntStat.CTL_SupportCmp) is issued.

(10) If a transaction error is detected in the middle of a control transfer, the control transfer support execution (H_CTL_SupportControl.CTL_SupportGo) is automatically cleared to abort the control transfer and a control transfer stopped status (H_CH0IntStat.CTL_SupportStop) is issued. The control transfer stage (H_CTL_SupportControl.CTL_SupportState) is flagged to indicate the stage in which the error occurred. In addition, the condition code (H_CH0ConditionCode) is set to a valid value and a ChangeCondition status (H_CH0IntStat.ChangeCondition bit) is issued.

To abort a control transfer, clear the control transfer support execution (H_CTL_SupportControl.CTL_SupportGo). A status will be issued when processing of abortion for the control transfer is complete.

At this point, if the control transfer is completed upto the status stage by the time abortion processing is complete, a control transfer complete status (H_CH0IntStat.CTL_SupportCmp) is issued.

If the control transfer is not completed by the time abortion processing is complete, a control transfer stopped status (H_CH0IntStat.CTL_SupportStop) is issued.

The stage in which the control transfer was aborted is indicated by the control transfer stage (H_CTL_SupportControl.CTL_SupportState).

To resume a control transfer from the stage in which it was aborted, set the control transfer stage (H_CTL_SupportControl.CTL_SupportState) to a stage from which the operation is to be resumed (i.e., the aborted stage) and set the control transfer support execution (H_CTL_SupportControl.CTL_SupportGo).

On the other hand, to perform a new control transfer, set the control transfer stage (H_CTL_SupportControl.CTL_SupportState) to Idle (00b) and then set the control transfer support execution (H_CTL_SupportControl.CTL_SupportGo).

While the control transfer support function is being executed, the transfer execution bit (H_CH0Config_0.TranGo), toggle sequence bit (H_CH0Config_0.Toggle), transaction type (H_CH0Config_1.PID), and IRP data size (H_CH0TotalSize_H,L) are set and updated by the hardware. Therefore, do not write to these register bits during that time.

For details about transaction errors, refer to the relevant sections in Chapter 6 on individual transactions.

Table 6.25 lists the setup items and status of the control transfer support function.

Table 6.25 Control Items and Status of the Control Transfer Support Function

Item	Register/Bit	Description
Control transfer support execution	H_CTL_SupportControl.CTL_SupportGo	Automatically executes control transfer stages. For details, refer to the section in Chapter 6 on control transfer support function.
Control transfer stage	H_CTL_SupportControl.CTL_SupportState	Indicates the stage being executed by the control transfer support function. Or if a control transfer was aborted for an error, it indicates the stage in which the error occurred.
Control transfer execution result	H_CH0IntStat.CTL_SupportCmp H_CH0IntStat.CTL_SupportStop	Indicates the result of a control transfer executed by the control transfer support function.
Transaction status	H_CH0IntStat.TotalSizeCmp, H_CH0IntStat.TranACK, H_CH0IntStat.TranErr, H_CH0IntStat.ChangeCondition	Indicates the result of a transaction.
Transaction condition code	H_CH0ConditionCode.ConditionCode	Indicates details of transaction result.

6.3.5 Bulk and Interrupt Transfers

Bulk transfers on CHa, as well as bulk and interrupt transfers on CHb, CHc, CHd, and CHe can be controlled either as a data flow (see 6.3.6) or as successive individual transactions (see 6.3.3).

6.3.6 Data Flow

This section describes control of a general data flow in OUT and IN transfers.

6.3.6.1 OUT Transfer

Set the total number of OUT transfer data bytes in H_CH0TotalSize_H,L or CHx{x=a-e}TotalSize_HH,HL,LH,LL and write the data to be transmitted by an OUT transfer to the FIFO of each channel. There are several methods for writing data to the FIFO: a register write through the CPU interface, a DMA write through the CPU interface, and a read transfer from IDE.

To write data to the FIFO by a register write through the CPU interface, select only one channel using the H_CHx{x=0,a-e}Join.JoinCPU_Wr bit. The FIFO of the selected channel can be written to by a FIFO_Wr register, and the data is transmitted in data packets in the order in which it was written. A free space in the FIFO can be inspected by FIFO_Remain_H,L registers. FIFOs in full state cannot be written to. Always be sure to inspect the FIFO_Remain_H,L registers to confirm the available size (in bytes) in the FIFO and make sure data is not written exceeding that size.

To write data to the FIFO by a DMA write through the CPU interface, select only one channel on either DMA channel using the H_CHx{x=0,a-e}Join.JoinDMAx{x=0,1} bit and set the DMAx{x=0,1}_Control.Dir bit to 0. The FIFO of the selected channel is written to by executing a DMA procedure in the CPU interface, and the data is transmitted in data packets in the order in which it was written. When the FIFO is full, the CPU interface automatically pause the DMA for flow control.

To write data to the FIFO by a read transfer of the IDE interface, select only one channel using the H_CHx{x=a-e}Join.JoinIDE bit and set the IDE_Control.Dir bit to 0. The FIFO of the selected channel is written to in the order read from IDE by executing an IDE transfer by IDE_Control.IDE_Go and the data is transmitted in data packets in the order in which it was written. When the FIFO is full, the IDE interface automatically pause the read transfer for flow control.

The size of data packets transmitted by an OUT transaction is H_CH0TotalSize_H,L or smaller one of H_CHx{x=a-e}TotalSize_HH,HL,LH,LL and H_CHx{x=0,a-e}MaxPktSize_H,L.

If data equal to or greater than the data size of data packet is present in the FIFO, an OUT transaction is executed to transmit the data. Also, H_CH0TotalSize_H,L or H_CHx{x=a-e}TotalSize_HH,HL,LH,LL is decremented by an amount equal to the transmitted data size. When TotalSize is reduced to 0, H_CHx{x=0,a-h}Config_0.TranGo is automatically cleared to terminate the transfer, and a TotalSizeCmp status (H_CHx{x=0,a-e}IntStat.TotalSizeCmp bit) is issued to the firmware.

That way, OUT transfers can be performed without the burden of controlling individual transactions by the firmware.

6.3.6.2 IN Transfer

Set the total number of IN transfer data bytes in H_CH0TotalSize_H,L or CHx{x=a-e}TotalSize_HH,HL,LH,LL.

The expected data length of the data packet to be received in an IN transaction is H_CH0TotalSize_H,L or smaller one of H_CHx{x=a-e}TotalSize_HH,HL,LH,LL and H_CHx{x=0,a-e}MaxPktSize_H,L. If the FIFO has a free space equal to or greater than MaxPacketSize, an IN transaction is executed to receive data. Also, the H_CH0TotalSize_H,L or H_CHx{x=0,a-e}TotalSize_HH,HL,LH,LL is decremented by an amount equal to the received data size. When TotalSize is reduced to 0, H_CHx{x=0,a-e}Config_0.TranGo is automatically cleared to terminate the transfer, and a TotalSizeCmp status (H_CHx{x=0,a-e}IntStat.TotalSizeCmp bit) is issued to the firmware.

If the received data size is larger than the expected size of data packet, H_CHx{x=0,a-e}Config_0.TranGo is automatically cleared to terminate the transfer. No response is returned. The condition code (H_CHx{x=0,a-e}ConditionCode) is set to DataOverrun, and a ChangeCondition status (H_CHx{x=0,a-e}IntStat.ChangeCondition bit) is issued to the firmware. The FIFO is not updated.

If the received data size is less than the expected size of data packet, H_CHx{x=0,a-e}Config_0.TranGo is automatically cleared to terminate the transfer. The condition code (H_CHx{x=0,a-e}ConditionCode) is set to DataUnderrun, and a ChangeCondition status (H_CHx{x=0,a-e}IntStat.ChangeCondition bit) is issued to the firmware. In addition, the FIFO is updated, and assuming data has been received, an area is reserved.

That way, IN transfers can be performed without the burden of controlling individual transactions by the firmware.

The data received by an IN transaction is written to the FIFO of each channel. There are several methods for reading the data from the FIFO: a register read through the CPU interface, a DMA read through the CPU interface, and a write transfer to IDE.

To read data from the FIFO by a register read through the CPU interface, select only one channel using the H_CHx{x=0,a-e}Join.JoinCPU_Rd bit. The FIFO of the selected channel can be read out in the order in which data was received by using the FIFO_Rd or FIFO_ByteRd register. The number of data bytes in the FIFO that can be read out is indicated by the FIFO_RrRemain_H,L registers. Since empty FIFOs cannot be read out, always be sure to inspect the FIFO_RrRemain_H,L registers to confirm the number of readable data bytes and make sure the FIFO is not accessed for read exceeding those bytes.

To read data from the FIFO by a DMA read through the CPU interface, select only one channel on either DMA channel using the H_CHx{x=0,a-e}Join.JoinDMAx{x=0,1} bit and set the DMAx{x=0,1}_Control.Dir bit to 1. The FIFO of the selected channel is read out in the order in which data was received by executing a DMA procedure in the CPU interface. The number of remaining data bytes in the FIFO can be known by the DMAx{x=0,1}_Remain_H,L registers. When the FIFO is emptied, the CPU interface automatically pause the DMA for flow control.

To read data from the FIFO by a write transfer to IDE, select only one channel using the H_CHx{x=a-e}Join.Join.IDE bit and set the IDE_Control.Dir bit to 1. The FIFO of the selected channel is read out in the order in which data was received by executing an IDE transfer by IDE_Control.IDE_Go. When the FIFO is emptied, the IDE interface automatically pause the write transfer for flow control.

6.3.7 Zero-length Packet Auto Issue Function

A function to automatically issue a zero-length packet is enabled by setting the H_CHx{x=a-e}Config_1.AutoZeroLen bit on a channel where an OUT transfer is performed.

Table 6.26 lists the setup items of the zero-length packet automatic issue function.

Table 6.26 Control Items of the Zero.length Packet Automatic Issue Function

Item	Register/Bit	Description
Zero-length packet automatic issue	H_CHx{x=a-e}Config_1.AutoZeroLen	Enables the zero-length packet automatic issue function. This bit is effective for only OUT transfers.

6.3.7.1 Zero-length Packet Auto Issue Function in Bulk/Interrupt OUT Transfers

On a channel where a bulk/interrupt OUT transfer is being executed, even when a transfer for the data size set by the H_CHx{x=a-e}TotalSize_HH,HL,LH,LL register is completed exactly with MaxPacketSize, H_CHx{x=a-e}Config_0.TranGo is not automatically cleared and the transfer is continued. Then, when the channel is scheduled again, an OUT transaction is executed with zero-length packet. When this transaction is completed without errors, H_CHx{x=a-e}Config_0.TranGo is automatically cleared to terminate the transfer, and a TotalSizeCmp status (H_CHx{x=a-e}IntStat.TotalSizeCmp bit) is issued to the firmware.

6.3.8 Bulk Only Support Function

The LSI stipulated herein has a function to automatically manage a series of transport operations for command transport (CBW), data transport, and status transport (CSW) of the USB Mass Storage Class (BulkOnly Transport Protocol). Use of this function relieves the firmware of the burden of controlling each transport individually. The diagram below shows an example where transports are controlled by the bulk-only support function and an example where each transport is controlled as an individual transaction without using this function.

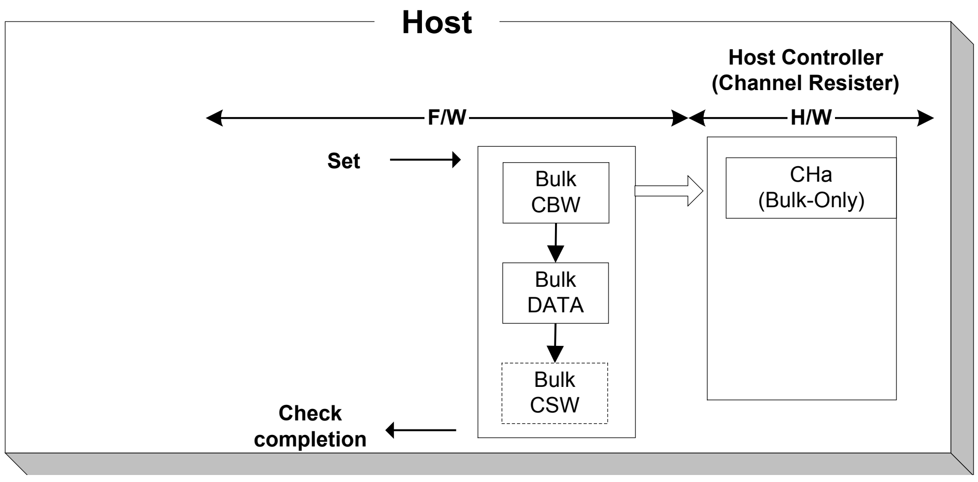


Fig. 6.34 Control of transports by the bulk.only support function

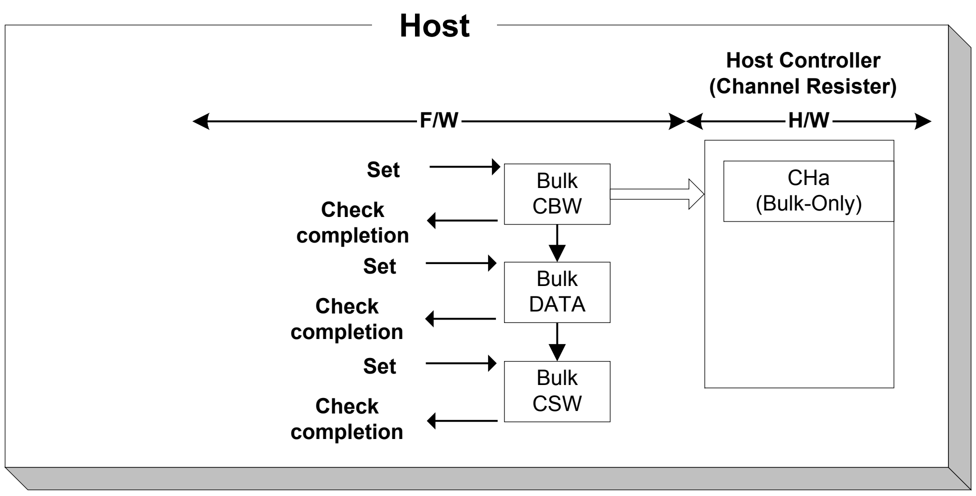


Fig. 6.35 Control of transports without using the bulk.only support function (reference)

The bulk-only support function is available only for channel CHa. Transport processing with the use of this function is performed in the manner shown below. Note that the firmware performs processes (1) to (5).

(1) Set the following basic setup registers for channel CHa as accordingly.
Transfer rate (H_CHaConfig_0.SpeedMode), max packet size (H_CHaMaxPktSize), USB address (H_CHaFuncAdrs.FuncAdrs), FIFO area (H_CHaStartAdrs,CHaEndAdrs)

(2) Set the following control registers for the bulk-only support function as accordingly.
OUT endpoint toggle sequence (H_OUT_EP_Control.OUT_Toggle), OUT endpoint number (H_OUT_EP_Control.OUT_EP_Number), IN endpoint toggle sequence (H_IN_EP_Control.IN_Toggle), IN endpoint number (H_IN_EP_Control.IN_EP_Number)

(3) Write CBW data (31 bytes) to the CBW area of the FIFO.

(4) Set up DMA or IDE in the CHa area of the FIFO.

(5) Set bulk-only support execution (H_BO_SupportControl.BO_SupportGo).
At this point, set the transport state (H_BO_SupportControl.BO_TransportState) to Idle (00b) by writing that value to the H_BO_SupportControl register.

(6) The data (31 bytes) present in the CBW area is transmitted to the OUT-direction endpoint indicated by the OUT endpoint number (H_OUT_EP_Control.OUT_EP_Number) via a bulk OUT transaction (command transport).

(7) A data transport is executed based on the CBW data.

- If bmCBWFlags in bit 7 of the CBW data = 0, the data present in the channel CHa area of the FIFO is transmitted to the OUT-direction endpoint indicated by the OUT endpoint number (H_OUT_EP_Control.OUT_EP_Number) via a bulk OUT transaction (OUT-direction data transport).
- If bmCBWFlags in bit 7 of the CBW data = 1, a bulk IN transaction is issued to the IN-direction endpoint indicated by the IN endpoint number

(H_IN_EP_Control.IN_EP_Number) and the received data is written to the channel CHa area of the FIFO (IN-direction data transport).

- A data transport is executed by performing transactions for the number of data bytes indicated by dCBWDataTransferLength of the CBW data.
- A data transport when it is directed for IN is complete even when the data size of the received data is less than that specified by dCBWDataTransferLength of the CBW data.
- A data transport in which dCBWDataTransferLength of the CBW data = 0x00000000 is not executed.

(8) A bulk IN transaction is issued to the IN-direction endpoint indicated by the IN endpoint number (H_IN_EP_Control.IN_EP_Number) and the received data is written to the CSW area of the FIFO (status transport). The number of data bytes received in a status transport is reflected in the status transport received data size (H_CSW_RcvDataSize.CSW_RcvDataSize).

- While the data transport is of the OUT-direction, if a transaction is completed for the data bytes indicated by dCBWDataTransferLength of the CBW data, the ensuing state will permit a status transport.
- While the data transport is of the IN-direction, if a transaction is completed for the data bytes indicated by dCBWDataTransferLength of the CBW data or all received data in the FIFO is read out and the FIFO is thereby emptied while no more IN transactions are issued due to short packet reception, the ensuring state will permit a status transport.

(9) The CSW received in a status transport is checked. This check is designed to confirm the following:

- The received data length of CSW is 13 bytes.
- dCSWSignature of CSW is 0x53425355.
- dCSWTag of CSW matches dCBWTag of CBW.
- If any one of the above is not true, the bulk-only support execution (H_BO_SupportControl.BO_SupportGo) is automatically cleared to turn the bulk-only support function off. In addition a bulk-only support stopped status (H_CHaIntStat.BO_SupportStop) is issued. The data received in the CSW area can be read out using the RAM_Monitor function.

(10) When a status transport is completed without errors, the bulk-only support execution (H_BO_SupportControl.BO_SupportGo) is automatically cleared, and a bulk-only support complete status (H_CHaIntStat.BO_SupportCmp) is issued.

(11) When a transaction error is detected during any of the transport processes, the bulk-only support execution (H_BO_SupportControl.BO_SupportGo) is automatically cleared to turn the bulk-only support function off, and a bulk-only support stopped status (H_CHaIntStat.BO_SupportStop) is issued. Then the transport state (H_BO_SupportControl.BO_TransportState) is flagged to indicate the transport in which the error occurred. In addition, the condition code (H_CHaConditionCode) is set to a valid value, and a ChangeCondition status (H-CHaIntStat.ChangeCondition bit) is issued.

To abort the bulk-only support function, clear the bulk-only support execution (H_BO_SupportControl.BO_SupportGo). A status is issued when abortion processing for the bulk-only support function is complete.

At this point, if the transport had been completed way up to a status transport by the time abortion processing has finished, a bulk-only support complete status (H_CHaIntStat.BO_SupportCmp) is issued.

At this point, if the transport is not completed upto the status transport by the time abortion processing is complete, a bulk-only support status (H_CHaIntStat.BO_SupportStop) is issued.

The aborted transport is indicated by the transport state (H_BO_SupportControl.BO_TransportState).

To resume a bulk-only support function from the aborted transport, set the transport state (H_BO_SupportControl.BO_TransportState) to a transport from which the operation is to be resumed (i.e., the aborted transport), and set the bulk-only support execution (H_BO_SupportControl.BO_SupportGo).

On the other hand, to execute the bulk-only support function anew, set the transport state (H_BO_SupportControl.BO_TransportState) to Idle (00b) and set the bulk-only support execution (H_BO_SupportControl.BO_SupportGo).

While the bulk-only support execution is being executed, the transfer execution bit (H_CHaConfig_0.TranGo), toggle sequence bit (H_CHaConfig_0.Toggle), transaction type (H_CHaConfig_1.PID), total size free bit (H_CHaConfig_1.TotalSizeFree), endpoint number (H_CHaFuncAdrs.EP_Number), and IRP data size (H_CHaTotalSize_HH,HL,LH,LL) are set and updated by the hardware. Therefore, do not write to these register bits during that time.

For details about transaction errors, refer to the relevant sections in Chapter 6 on individual transactions.

For details about the CBW and CSW areas of the FIFO, refer to the relevant section in Chapter 6 on FIFOs.

For details about the DMA, refer to the relevant section in Chapter 6 on DMAs.

For details about the IDE, refer to the relevant section in Chapter 6 on IDEs.

Table 6.27 lists the setup items and status of the bulk-only support function.

Table 6.27 Setup Items and Status of the Bulk Only Support Function

Item	Register/Bit	Description
Bulk Only support execution	H_BO_SupportControl.BO_SupportGo	Executes the bulk-only support function. For details, refer to the relevant section in Chapter 6 on bulk-only support function.
OUT endpoint toggle sequence	H_OUT_EP_Control.OUT_Toggle	Sets the initial value of the OUT endpoint toggle sequence bit. When a transaction is being executed or a transaction is completed, it indicates the state of the OUT endpoint toggle sequence bit.
OUT endpoint number	H_OUT_EP_Control.OUT_EP_Number	Sets the endpoint number of the OUT endpoint to any value between 0x0 and 0xF.
IN endpoint toggle sequence	H_IN_EP_Control.IN_Toggle	Sets the initial value of the IN endpoint toggle sequence bit. When a transaction is being executed or a transaction is completed, it indicates the state of the IN endpoint toggle sequence bit.
IN endpoint number	H_IN_EP_Control.IN_EP_Number	Sets the endpoint number of the IN endpoint to any value between 0x0 and 0xF.
Result of bulk-only support execution	H_CHaIntStat.BO_SupportCmp H_CHaIntStat.BO_SupportStop	Indicates the result of the bulk-only support executed.
Transaction status	H_CHaIntStat.TotalSizeCmp, H_CHaIntStat.TranACK, H_ CHaIntStat.TranErr, H_CHaIntStat.ChangeCondition	Indicates the result of a transaction.
Transaction condition code	H_CHaConditionCode	Indicates details of transaction result.
Transport status	H_BO_SupportControl. BO_TransportState	Indicates the transport being executed under control of the bulk-only support function. If any transport was aborted for an error, it indicates the transport in which the error occurred.
Status transport received data size	H_CSW_RcvDataSize	Indicates the number of data bytes received in a status transport.

6.3.9 Host State Management Support Function

6.3.9.1 Host States

The host must have its state changed according to a request from a high-order system and the bus state. Therefore, the host states are managed by the firmware. The hardware supports various settings and negotiations in each state.

Fig. 6.36 shows a transition of the host states.

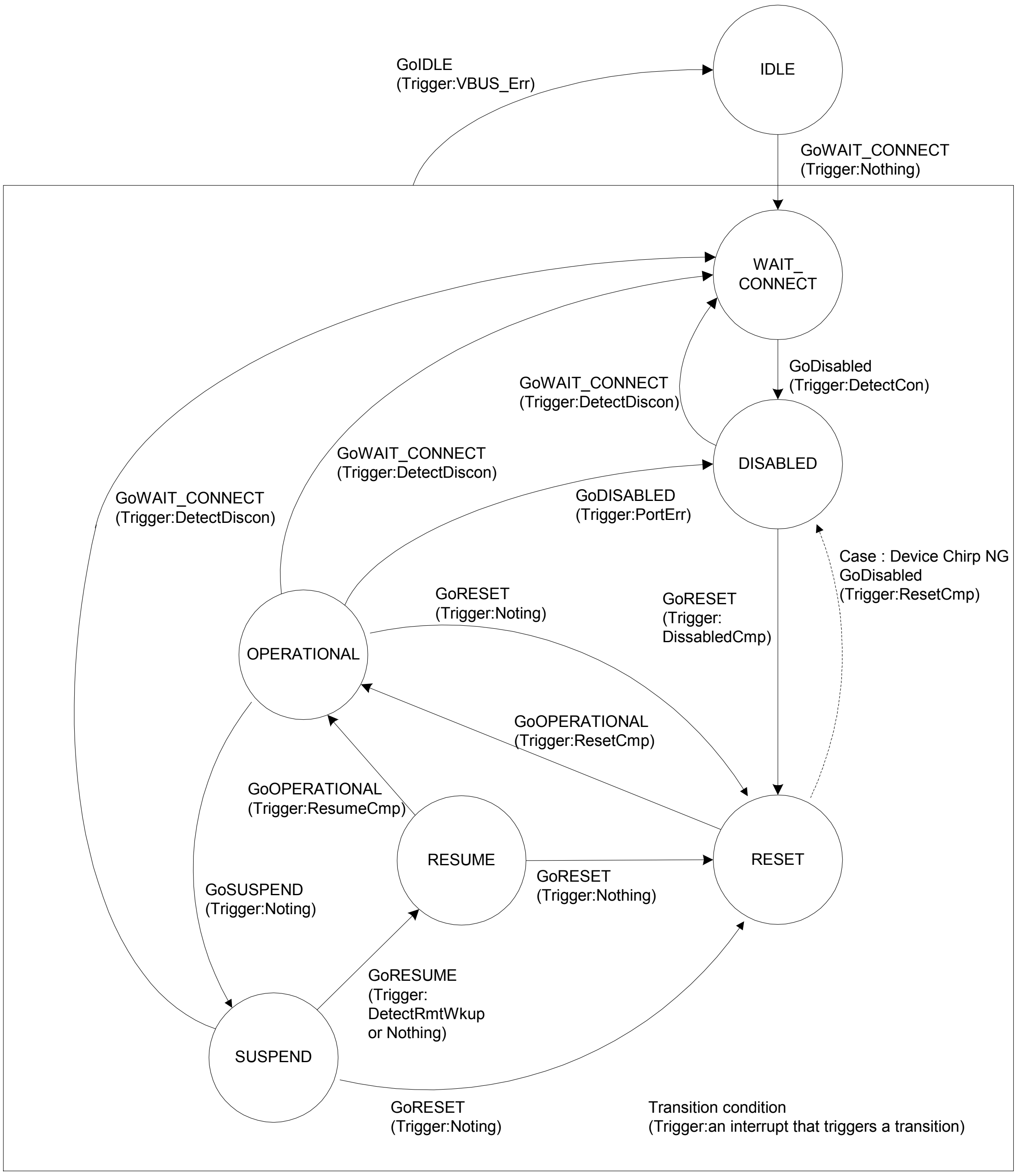


Fig. 6.36 Host state transition diagram

Table 6.28 lists the setup items and status of the host state management support function.

Table 6.28 Setup Items and Status of the Host State Management Support Function

Item	Register/Bit	Description
Host state transition execution	H_NegoControl_0. AutoMode	Sets the host state to which to go. The state that is set here is one of the following: GoIDLE GoWAIT_CONNECT GoDISABLED GoRESET GoOPERATIONAL GoSUSPEND GoRESUME GoWAIT_CONNECTtoDIS GoWAIT_CONNECTtoOP GoRESETtoOP GoRESUMEtoOP GoSUSPENDtoOP
Host state transition execution cancel	H_NegoControl_0. AutoModeCancel	Stops processing of the current host state and ceases processing at that state.
Host state monitor	H_NegoControl_0. HostState	Indicates the current host state (shown below). IDLE WAIT_CONNECT DISABLED RESET OPERATIONAL SUSPEND RESUME
VBUS state monitor	H_USB_Status. VBUS_State	Indicates the VBUS state (normal/erratic).
Remote wakeup acceptance enable	H_NegoControl_1. RmtWkupDetEnb	Enables remote wakeup acceptance.
Chirp complete disable	H_NegoControl_1.DisChirpFinish	Sets operation mode to be assumed when device chirp is not completed within a specified time.
VBUS error detection status	H_SIE_IntStat_0. VBUS_Err	Indicates that an error occurred in VBUS.
Connection detection status	H_SIE_IntStat_0. DetectCon	Indicates that a device has been connected to the downstream port.
Disconnection detection status	H_SIE_IntStat_0. DetectDisCon	Indicates that a device has been disconnected from the downstream port.
Remote wakeup detection status	H_SIE_IntStat_0. DetectRmtWkup	Indicates that a remote wakeup signal from the device has been detected.
Device chirp complete detection status	H_SIE_IntStat_0. DetectDevChirpOK	Indicates that a chirp signal from the device is normal.
Device chirp error detection status	H_SIE_IntStat_0. DetectDevChirpNG	Indicates that a chirp signal from the device is abnormal.
Reset complete status	H_SIE_IntStat_1. ResetCmp	Indicates that USB reset is completed without errors.
Suspend transition complete status	H_SIE_IntStat_1. SuspendCmp	Indicates that a transition to Suspend is completed without errors.
Resume completion status	H_SIE_IntStat_1. ResumeCmp	Indicates that Resume is completed without errors.
Port speed	H_NegoControl_1. PortSpeed	Indicates the operating speed (HS, FS or LS) of the downstream port, i.e., operation of the device connected to the port.

Line state	H_USB_Status. LineState	Indicates the signal state of USB cable.
Transceiver selection	H_XcvrControl. XcvrSelect	Selects and enables one of HS, FS or LS transceivers.
Terminal selection	H_XcvrControl. TermSelect	Selects and enables one of HS, FS or LS terminals.
Operation mode	H_XcvrControl. OpMode	Sets operation mode of the HTM.

6.3.9.1.1 IDLE

This is the state where the USB host function is initialized, and is the default state that is assumed when the USB host function is enabled.

The host must be moved to this state when VBUS_Err is detected in any other state.

To execute a transition, write 0x80 to the H_NegoControl_0 register (H_NegoControl_0.AutoModeCancel = 1 and H_NegoControl_0.AutoMode = 0x0) to cause operation of the state under execution to stop. The H_NegoControl_0.AutoModeCancel bit is cleared to 0 by a completion of the stop processing (6 cycles required when operating with 60 MHz clock). After confirming that the H_NegoControl_0.AutoModeCancel bit has been cleared to 0, write 0x01 to the same register (which sets the host state transition execution (H_NegoControl_0.AutoMode) to GoIDLE). The LSI thereby enters this state.

In this state, the following settings are automatically executed:

- Transaction execution function of the USB host put to immediate stop
- Port set to FS mode and to NonDriving
- VBUSEN_A turned off
- All of the connection detection, disconnection detection, remote wakeup detection, and device chirp detection functions turned off

6.3.9.1.2 WAIT_CONNECT

This is the state where the host waits until a device is connected to the downstream port.

When a device disconnection is detected in OPERATIONAL and RESET state due to request from a higher bit in the IDLE state, the host must be temporarily moved to this state and wait until the other party is connected.

The LSI shifts to this state when the host state transition execution (H_NegoControl_0.AutoMode) is set to GoWAIT_CONNECT.

In this state, the following initial settings are automatically executed:

- Transaction execution function of the USB host put to immediate stop
- Port set to FS mode and to PowerDown
- VBUSEN_A turned on
- All of the connection detection, disconnection detection, remote wakeup detection, and device chirp detection functions turned on

Next, after internal power supply stabilization time for the bus power device has elapsed, the hardware automatically turns the connection detection function on, and waits until a device is connected. Note that the duration of time from when VBUSEN_A turned on to when a device connection is detected is not controlled by the hardware. This time should be managed by the firmware as necessary (by, for example, placing the host in IDLE state by AutoModeCancel when a connection cannot be detected within a specified time after it shifts to this state).

If FS or HS device is connected, it can be referenced as line state "J." If an LS device is connected, it can be referenced as line state "K." When either state continues for 2.5 µs or more, a device connection is detected. If the connected device is an LS device, the port is set to LS mode.

When a connection is detected, the disconnection detection function is automatically turned on.

Thereafter, if a disconnection is not detected during the debounce interval period, a connection detected status (H_SIE_IntStat_0.DetectCon) is issued to the firmware and the connection detection function as well as disconnection detection function is automatically turned off. Conversely, if a disconnection is detected, the disconnection detection function is automatically turned off and the process restarts from the connection detection phase.

6.3.9.1.3 DISABLED

This is the state where the downstream port has a device connected, with no signals sent or received on the bus.

The LSI shifts to this state when a connection is detected in the WAIT_CONNECT state, Chirp from an erratic device is detected in the RESET state, or a port error is detected in the OPERATIONAL state.

The LSI shifts to this state when the host state transition execution (H_NegoControl_0.AutoMode) is set to GoDISALBED.

In this state, the following initial settings are automatically executed.

- Transaction execution function of the USB host put to immediate stop
- Port set to FS mode (in HS mode before the LSI shifted to this state) or retain current mode (in FS or LS mode)
- Port set to PowerDown

Next, the following processes are automatically executed after the disconnection detection disabled period expires:

- Disconnection detection function is turned on
- Disabled transition complete status (H_SIE_IntStat_1.DisableCmp) is issued

6.3.9.1.4 RESET

This is the state where a USB reset is issued to the downstream port.

When a disabled transition complete status is issued in the DISABLED state, the LSI shifts to this state and then issues a USB reset.

In addition, if a request is made from a high-order system, the LSI can shift to this state from whichever USB mode (OPERATIONAL, SUSPEND, or RESUME).

The LSI shifts to this state when the host state transition execution (H_NegoControl_0.AutoMode) is set to GoRESET.

In this state, the following initial settings are automatically executed.

- Transaction execution function of the USB host put to immediate stop
- Port set to HS mode and to NormalOperation (reset signal SE0 is driven onto the USB cable)
- Connection detection, disconnection detection, and remote wakeup detection functions turned on
- Device chirp detection function turned on

Chirp from a device is detected as “HS K” from the downstream port. It is detected by a line state “K” that continues for 2.5 μs or more, and when the line state “K” is terminated within a specified time after a USB reset is issued, it is detected as normal Chirp. If the state is not terminated within a specified time, it is detected as erratic Chirp.

The following processes are automatically executed according to the detection result.

(1) When Chirp from a normal device is detected:
Pursuant to completion of Chirp from the device, “HS K” (Chirp K) and “HS J” (Chirp J) are alternately set out in succession from the host. When the host completes sending out Chirp, a reset complete status (H_SIE_IntStat_1.ResetCmp) is issued to the firmware.
The port remains in HS mode.

(2) When Chirp from an erratic device is detected:
After certain specified time elapses a device chirp error detected status (H_SIE_IntStat_0.DetectDevChirpNG) is issued to the firmware.
Ensuing operation can be selected from two operation modes by setting the chirp complete disable (H_NegoControl_1.DisChirpFinish). For details, refer to Section 6.3.9.3.4.2, “Detection of Chirp from an Erratic Device.”

(3) When Chirp from a device is not detected and the other connected party is FS:
The port is set to FS mode after issuing a USB reset of a designated duration.
A reset complete status (H_SIE_IntStat_0.ResetCmp) is issued to the firmware.

(4) When the other connected party is LS:
The port is set to LS mode after issuing a USB reset of a designated duration.
A reset complete status (H_SIE_IntStat_0.ResetCmp) is issued to the firmware.

6.3.9.1.5 OPERATIONAL

This is the state where a USB transaction is executed.

The LSI shifts to this state after completion of RESET or RESUME and then executes a transaction.

The LSI shifts to this state when the host state transition execution (H_NegoControl_0.AutoMode) is set to GoOPERATIONAL.

In this state, the following settings are automatically executed:

- Port set to NormalOperation
- Transaction execution function of the USB host enabled
- Disconnection detection function turned on

Note: To ensure a correct transition from RESET or RESUME to OPERATIONAL, the designated time in USB 2.0 standard for the interval before the first uSOF(HS), SOF(FS), or KeepAlive(LS) is issued must be observed. To perform a transition without regard for this designated interval, we recommend using a command for successive transitions, i.e., GoRESETtoOP, GoSUSPENDtoOP, or GoRESUMEtoOP. For more information on designated intervals, refer to T11 in 6.3.9.3.4.1, T5 in 6.3.9.3.4.3, T5 in 6.3.9.3.4.4, T5 in 6.3.9.3.7.1, T4 in 6.3.9.3.7.2, and T4 in 6.3.9.3.7.3.

6.3.9.1.6 SUSPEND

This is the state where USB is suspended.

The LSI shifts to this state from OPERATIONAL when it is necessary to stop the use of the USB bus.

The LSI shifts to this state when the host state transition execution (H_NegoControl_0.AutoMode) is set to GoSUSPEND.

In this state, the following initial settings are automatically executed:

- Disconnection detection and remote wakeup detection functions turned off
- Transaction execution function of the USB host disabled after waiting for completion of the current transaction
- Port set to FS mode (in HS mode), or retain current mode (in FS or LS mode)
- Port set to PowerDown.

Next, the following processes are automatically executed after the disconnection and remote wakeup detection disable period has expired.

- Disconnection detection function turned on.
- Remote wakeup detection turned on if remote wakeup acceptance enable (H_NegoControl_1.RmtWkupDetEnb) is enabled.
- Suspend transition complete status (H_SIE_IntStat_1.SuspendCmp) issued

In addition, if while the remote wakeup acceptance enable (H_NegoControl_1.RmtWkupDetEnb) is enabled, a remote wakeup signal (consecutive “K” for 2.5 μs or more) is detected, a remote wakeup detected status (H_SIE_IntStat_0.DetectRmtWkup) is issued to the firmware.

6.3.9.1.7 RESUME

This is the state where a USB resume signal is issued to the downstream port.

The LSI shifts to this state from SUSPEND in order to restore the USB device from a suspend state.

The LSI shifts to this state when the host state transition execution (H_NegoControl_0.AutoMode) is set to GoRESUME.

In this state, the disconnection and remote wakeup detection functions are automatically turned off, and a resume signal (K) of a designated duration is issued.

When issuance of the resume signal is complete, the port is set back to its previous mode before shifting to SUSPEND, and is set to NormalOperation.

In addition, a suspend complete status (H_SIE_IntStat_0.ResumeCmp) is issued to the firmware.

6.3.9.2 Detection Functions

6.3.9.2.1 VBUS Error Detection

A VBUS error is detected by a level change (from high to low) at the VBUSFLG_A input pin. The procedure that is executed when a VBUS error is detected is described below. Procedure step (2) below is automatically executed by the LSI’s hardware.

(1) Pull the VBUSFLG_A (error flag for external USB power switch) input pin low (error occurred) (T0).

(2) Issue a VBUS error detected status (HostIntStat.VBUS_Err) to the firmware (T0).

Note that when the host detects a VBUS error, the VBUS must be turned off immediately. Therefore, the firmware should write 0x80 to the H_NegoControl_0 register (H_NegoControl_0.AutoModeCancel = 1 and H_NegoControl_0.AutoMode = 0x0) to cause operation of the state under execution to stop. The H_NegoControl_0.AutoModeCancel bit is cleared to 0 by a completion of the stop process (6 cycles required when operating with 60 MHz clock). After confirming that the H_NegoControl_0.AutoModeCancel bit is cleared to 0, write 0x01 to the same register (which sets the host state transition execution

(H_NegoControl_0.AutoMode) to GoIDLE). This will cause the host to shift to IDLE state and the VBUSEN_A pin logic to be disabled, with the result of VBUS being turned off.

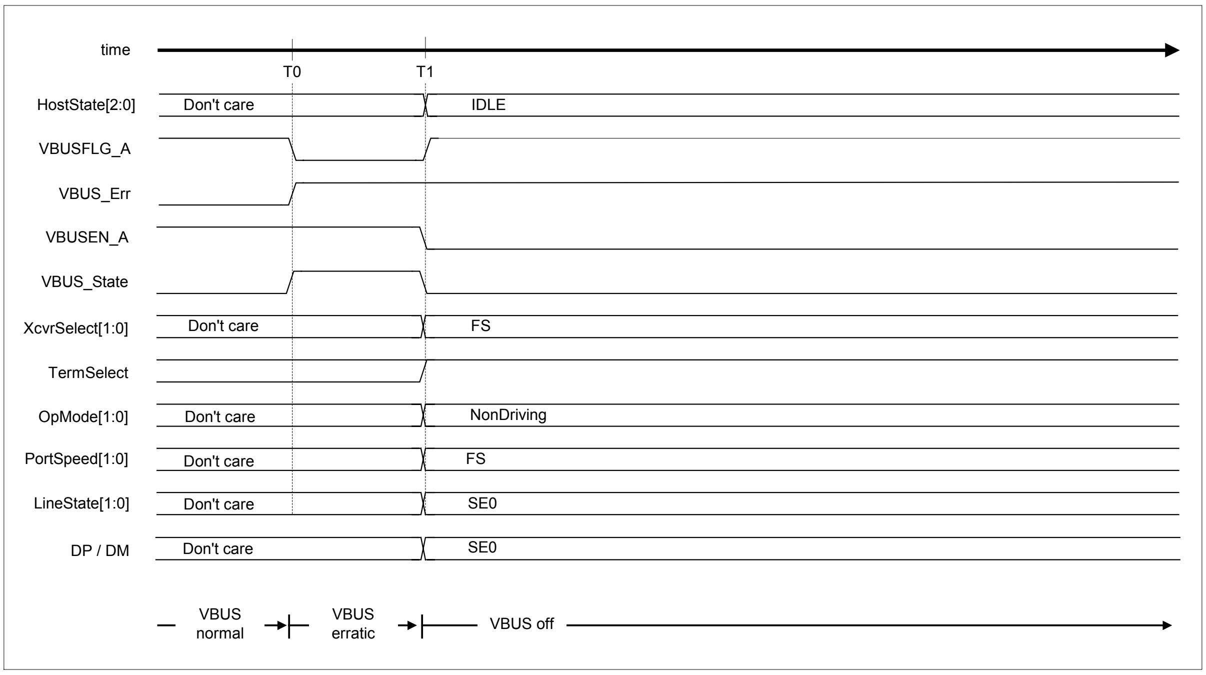


Fig. 6.37 VBUS error detection timing

Table 6.29 VBUS Error Detection Timing Values

Timing Parameter	Description	Value
T0	The VBUSFLG_A (error flag for external USB power switch) input pin goes low (error occurred). A VBUS error detected status (HostIntStat.VBUS_Err) is issued (by hardware).	0 (reference)
T1 (reference)	Transition to IDLE state is executed by first writing 0x80 and then writing 0x01 to H_NegoControl_0 (by firmware).	T1

6.3.9.2.2 Disconnection Detection

Disconnection of a device is detected in DISABLED, OPERATIONAL, and SUSPEND state.

To restart from the connection detection phase without turning VBUS off when a disconnection is detected, be sure to execute transition of the host state to WAIT_CONNECT. On the other hand, if the VBUS needs to be turned off, then execute transition of the host state to IDLE.

6.3.9.2.2.1 When HS Device is Disconnected

Disconnection of an HS device is detected in the OPERATIONAL state.

The procedure that is executed when an HS device is disconnected is described below. Procedures (2) to (3) are automatically executed by the LSI's hardware.

(1) A device is disconnected (T0).

(2) A disconnection is detected in the EOP period of uSOF(HS_SOF) after the device was disconnected (T1).

(3) A disconnection detected status (H_SIE_IntStat_0.DetectDiscon) is issued to the firmware (T1).

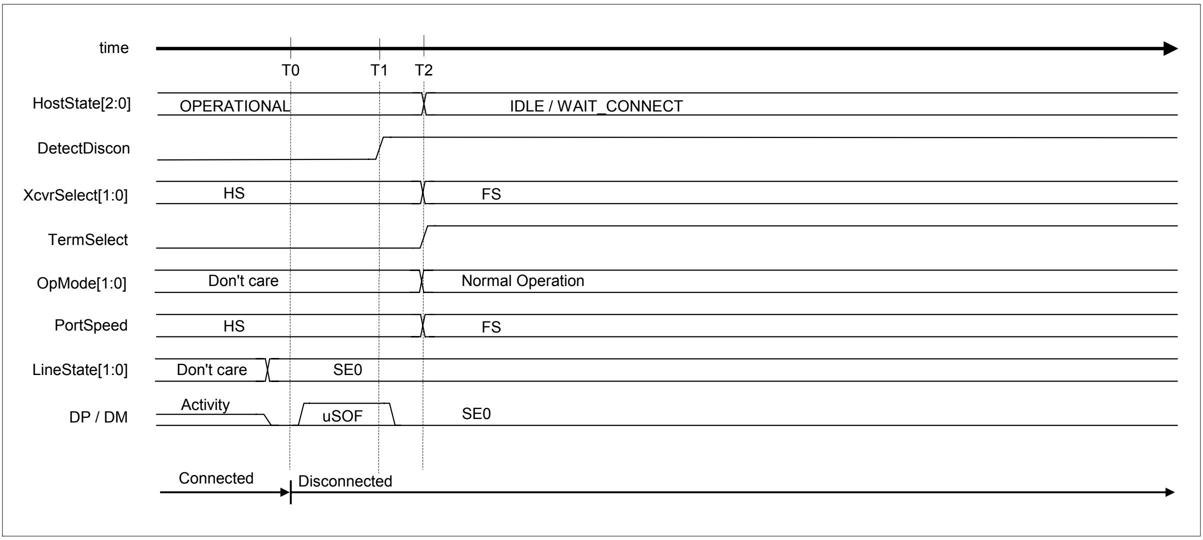


Fig. 6.38 Disconnection detection timing (HS mode)

Table 6.30 Disconnection Detection Timing Values (HS Mode)

Timing Parameter	Description	Value
T0	A device is disconnected.	0 (reference)
T1	A disconnection detected status (H_SIE_IntStat_0.DetectDiscon) is issued (by hardware).	T1
T2 (reference)	Host state transition execution (H_NegoControl_0.AutoMode) is set to GoWAIT_CONNECT (by firmware).	T2

6.3.9.2.2.2 When FS or LS Device is Disconnected

Disconnection of an FS or LS device is detected in the states WAIT_CONNECT, DISABLED, OPERATIONAL, and SUSPEND.

The procedure that is executed when an FS or LS device is disconnected is described below. Procedures (2) to (3) are automatically executed by the LSI's hardware.

(1) A device is disconnected (T0).

(2) A disconnection is detected from the signal line state.

(3) A disconnection detected status (H_SIE_IntStat_0.DetectDiscon) is issued to the firmware (T1).

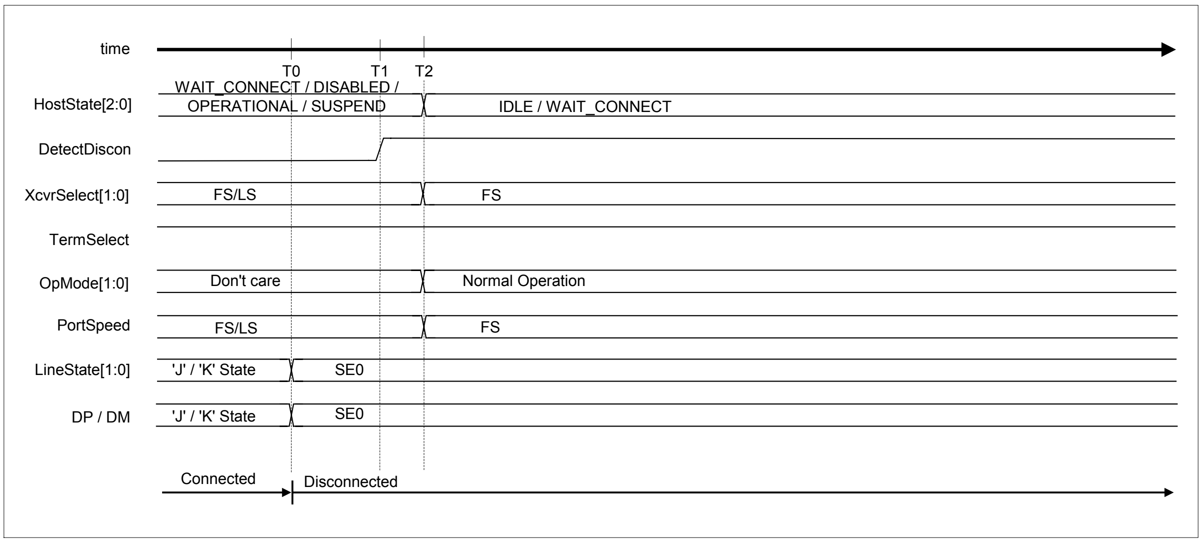


Fig. 6.39 Disconnection detection timing (FS or LS mode)

Table 6.31 Disconnection Detection Timing Values (FS or LS Mode)

Timing Parameter	Description	Value
T0	A device is disconnected.	0 (reference)
T1	A disconnection detected status (H_SIE_IntStat_0.DetectDiscon) is issued (by hardware).	T0 + 2.5us< T1 {T_{URLSE0}}
T2 (reference)	Host state transition execution (H_NegoControl_0.AutoMode) is set to GoWAIT_CONNECT (by firmware).	Not stipulated

6.3.9.2.3 Remote Wakeup Detection

If the remote wakeup acceptance enable (H_NegoControl_1.RmtWkupDetEnb) is enabled, remote wakeup detection is performed in the SUSPEND state.

6.3.9.2.3.1 When HS Device is Connected

The procedure that is executed for the case where an HS device is connected is described below. Procedures (2) to (3) are automatically executed by the LSI's hardware.

(1) The device starts sending out a remote wakeup signal (K) (T0).

(2) The host detects the remote wakeup signal (K) (T1).

(3) A remote wakeup detected status (H_SIE_IntStat_0.DetectRmtWkup) is issued to the firmware (T1).

Note that the host must issue a resume signal (K) within 1 ms after detecting a remote wakeup from the device. Therefore, the firmware should immediately set the host state transition execution (H_NegoControl_0.AutoMode) to GoRESUME within 900 µs after recognizing the remote wakeup detected status.

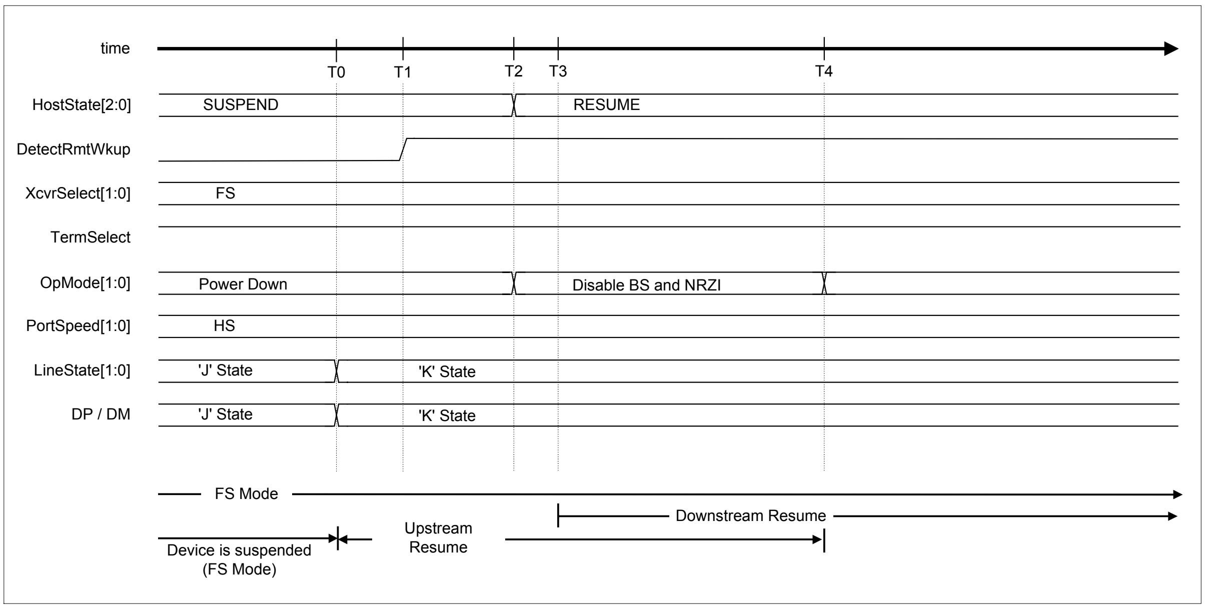


Fig. 6.40 Remote wakeup timing (HS mode)

Table 6.32 Remote Wakeup Timing Values (HS Mode)

Timing Parameter	Description	Value
T0	The device starts sending out a remote wakeup signal (K).	0 (reference)
T1	The remote wakeup signal (K) is detected. A remote wakeup detected status is issued (by hardware).	T0 + 2.5us{T_{URLK}} < T1
T2 (reference)	Host state transition execution (H_NegoControl_0.AutoMode) is set to GoRESUME (by firmware).	T2< T1 + 900us
T3 (reference)	The host starts issuing a resume signal (K) (in hardware).	T3< T0 + 1ms{T_{URSM}}

Note: Names stipulated in the USB2.0 Standard are shown in { } .

6.3.9.2.3.2 When FS Device is Connected

The procedure to be executed when an FS device is connected is the same as for the case when HS device is connected.

For details, refer to the preceding section.

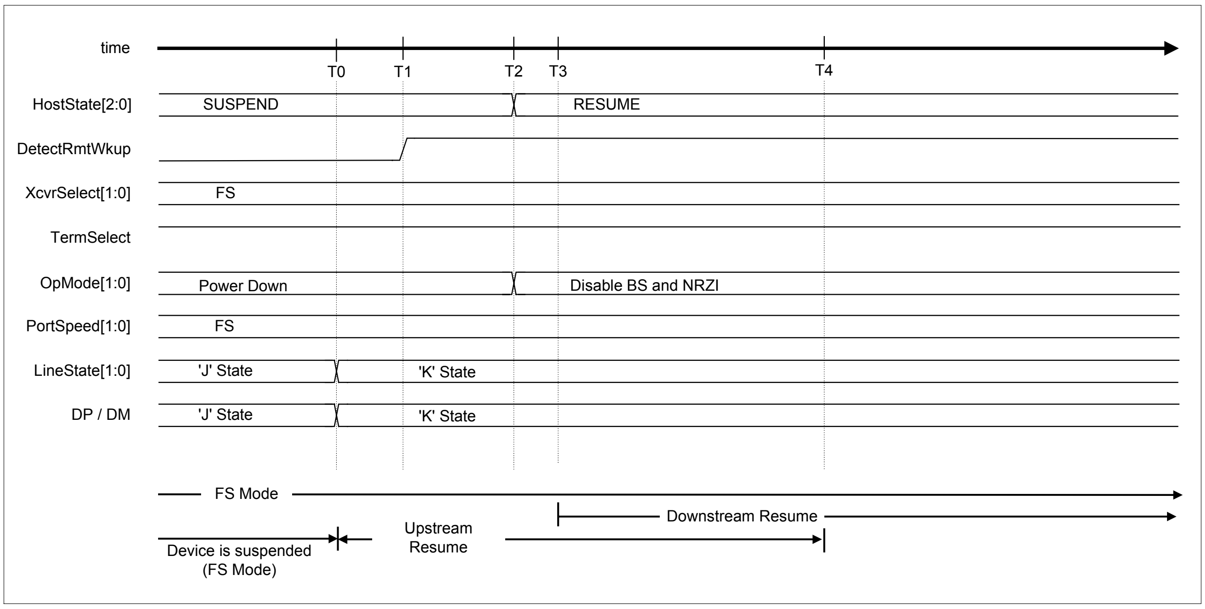


Fig. 6.41 Remote wakeup timing (FS mode)

Table 6.33 Remote Wakeup Timing Values (FS Mode)

Timing Parameter	Description	Value
T0	The device starts sending out a remote wakeup signal (K).	0 (reference)
T1	The remote wakeup signal (K) is detected. A remote wakeup detected status is issued (by hardware).	T0 + 2.5us< T1 {T_{URLK}}
T2 (reference)	Host state transition execution (H_NegoControl_0.AutoMode) is set to GoRESUME (by firmware).	T2< T1 + 900us
T3 (reference)	The host starts issuing a resume signal (K) (in hardware).	T3< T0 + 1ms{T_{URSM}}

Note: Names stipulated in the USB2.0 Standard are shown in { }.

6.3.9.2.3.3 When LS Device is Connected

The procedure to be executed when an LS device is connected is the same as for the case when an HS device is connected.

For details, refer to the preceding section.

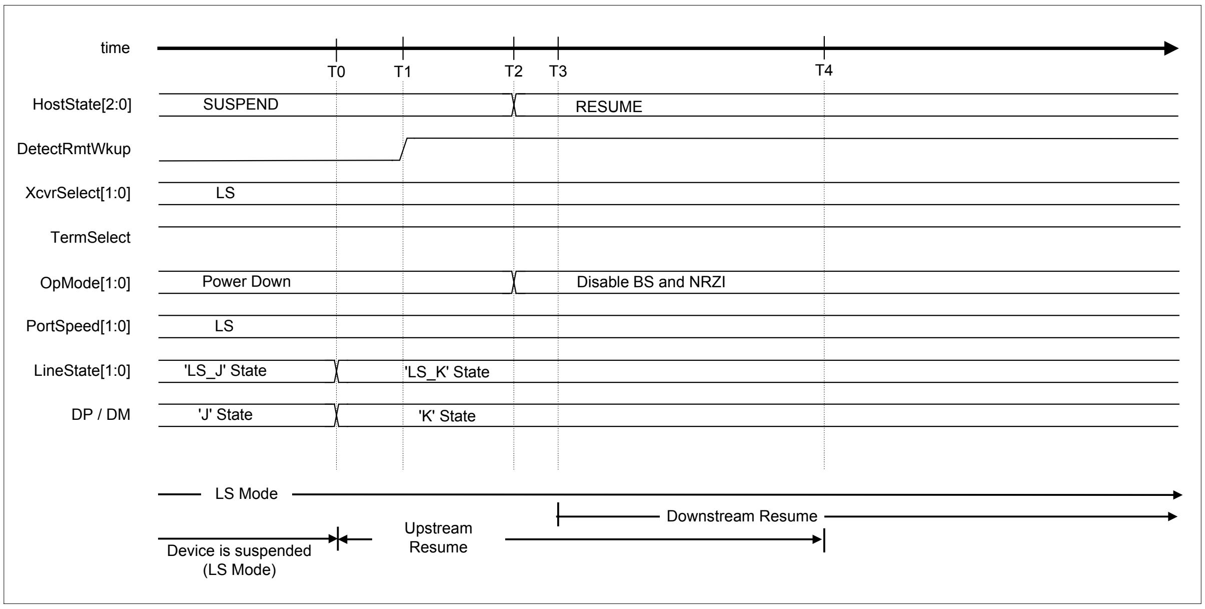


Fig. 6.42 Remote wakeup timing (LS mode)

Table 6.34 Remote Wakeup Timing Values (LS Mode)

Timing Parameter	Description	Value
T0	The device starts sending out a remote wakeup signal (K).	0 (reference)
T1	The remote wakeup signal (K) is detected. A remote wakeup detected status is issued (by hardware).	T0 + 2.5us< T1 {T_{URLK}}
T2 (reference)	Host state transition execution (H_NegoControl_0.AutoMode) is set to GoRESUME (by firmware).	T2< T1 + 900us
T3 (reference)	The host starts issuing a resume signal (K) (in hardware).	T3< T0 + 1ms{T_{URSM}}

Note: Names stipulated in the USB2.0 Standard are shown in { }.

6.3.9.2.4 Device Chirp Detection Function

A device Chirp is detected.

The device chirp detection function is turned on in the RESET state.

6.3.9.2.4.1 When a Correct Device Chirp is Detected

The procedure for detecting a device Chirp is as described below:

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoRESET (T0).

(2) The device chirp detection function is turned on (T0).

(3) The device sends out a Chirp (T1).

(4) A device Chirp is recognized by a line state “K” (indicated by USB_Host_Status.LineState[1:0]) that continues for a specified time or more (T2).

(5) When the device Chirp is determined to have ended within a specified time after a reset started (i.e., the line state (USB_Host_Status.LineState[1:0]) has changed to ‘SE0’), a device chirp normal detection status (H_SIE_IntStat_0.DetectDevChirpOK) is issued (T3).

(6) When a device Chirp is detected, the device chirp detection function is turned off (T3).

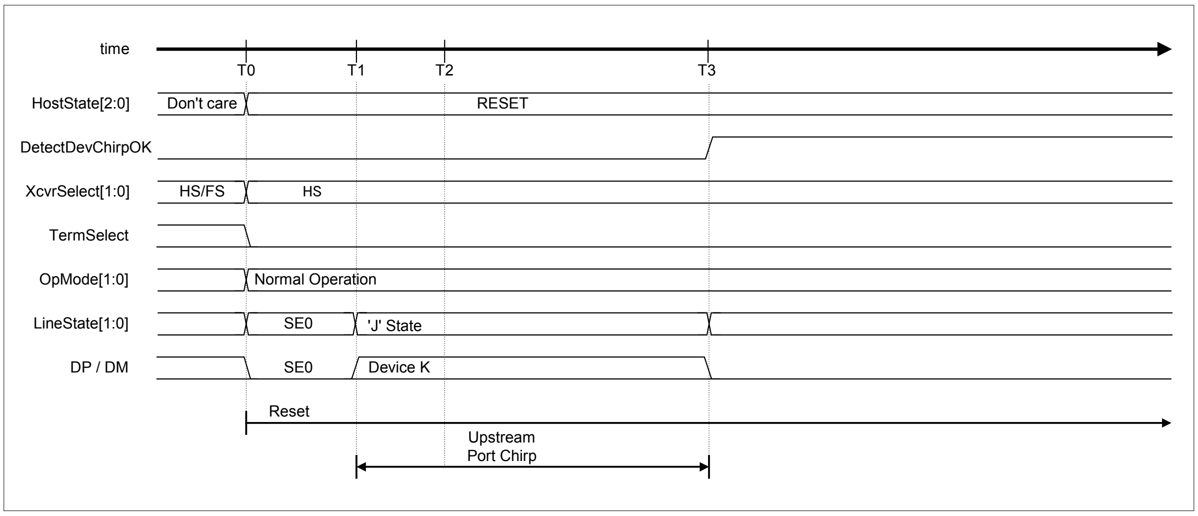


Fig. 6.43 Device chirp timing

Table 6.35 Device Chirp Timing Values

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoRESET (by firmware).	0 (reference)
T1	The device starts a Chirp.	T0 < T1 < T0 + 6.0ms
T2	The device Chirp is recognized (by hardware).	T1 + 2.5us {T_{FILT}} < T2
T3	The device completes the Chirp. The device chirp detection function is turned off. A device chirp normal detection status (DetectDevChirpOK) is issued (by hardware).	T1 + 1.0ms {T_{UCH}} < T3 < T0 + 7.0ms {T_{UCHEND}}

Note: Names stipulated in the USB2.0 Standard are shown in { }.

6.3.9.2.4.2 When an Erratic Device Chirp is Detected

The device chirp detection function assumes an error when a device Chirp does not end within a specified time, and issues a status accordingly.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoRESET (T0).

(2) The device chirp detection function is turned on (T0).

(3) The device sends out a Chirp (T1).

(4) A device Chirp is recognized by a line state “K” (indicated by USB_Host_Status.LineState[1:0]) that continues for a specified time or more (T2).

(5) Because the device Chirp does not end within a specified time after a reset started, an error is assumed and a device chirp abnormal detection status (H_SIE_IntStat_0.DetectDevChirpNG) is issued (T3).

(6) When a device Chirp is detected, the device chirp detection function is turned off (T3).

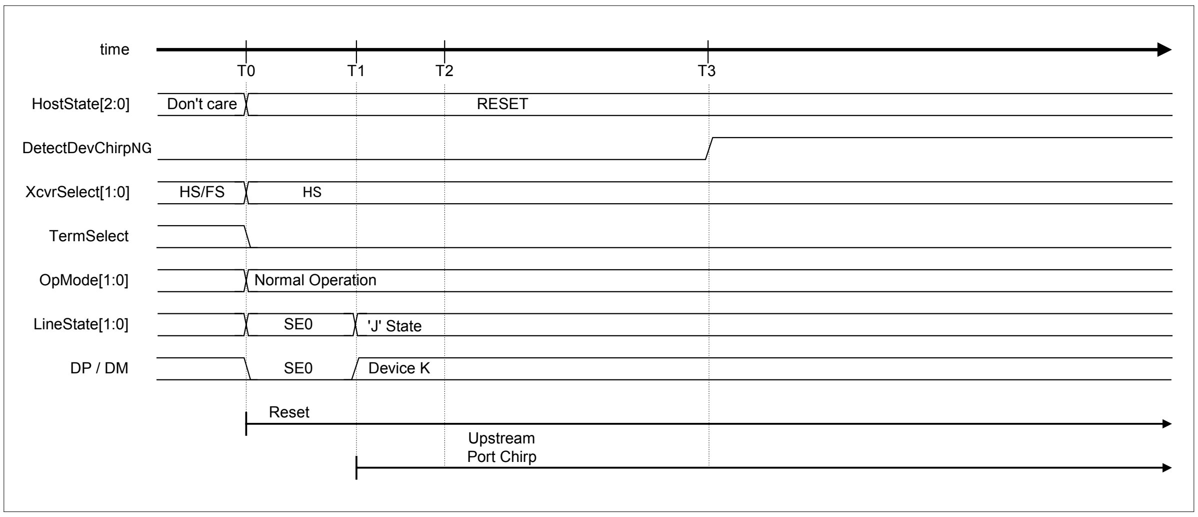


Fig. 6.44 Device chirp timing (NG)

Table 6.36 Device Chirp Timing (NG) Values

Timing Parameter	Description	Value
T0	USB_Control_0.AutoMode is set to GoRESET (by firmware).	0 (reference)
T1	The device starts a Chirp.	T0 < T1 < T0 + 6.0ms
T2	The device Chirp is recognized (by hardware).	T1 + 2.5us {T_{FILT}} < T2
T3	A device chirp abnormal detection status (DetectDevChirpNG) is issued (by hardware).	T0 + 7ms{T_{UCHEND}} < T3

Note: Names stipulated in the USB2.0 Standard are shown in { }.

6.3.9.2.5 Port Error Detection

In the OPERATIONAL state, port errors are detected.

A port error is raised when the LSI cannot detect the EOP even at the endpoint of the (micro) frame during a packet reception.

Once a port error is detected by the host, a port error detection status (H_FrameIntStat.PortErr) is issued to the firmware while transaction is stopped immediately. Thereafter, no transaction (including SOF) will be issued.

When a port error occurs, set the firmware as follows:

(1) Set H_NegoControl_0.AutoMode to GoDISABLED.

(2) Set H_NegoControl_0.ResetHTM to 1 and reset the host transceiver macro.

(3) After more than 3 cycles have elapsed with 60MHz clock, set H_NegoControl_0.ResetHTM to 0 and then cancel the reset on host transceiver macro.

6.3.9.3 Description of Individual Host State Management Support Function

6.3.9.3.1 GoIDLE

Write 0x80 to the H_NegoControl_0 register (H_NegoControl_0.AutoModeCancel = 1 and H_NegoControl_0.AutoMode = 0x0) to cause operation of the state under execution to stop. The H_NegoControl_0.AutoModeCancel bit is cleared to 0 by a completion of the stop process (6 cycles required when operating with 60 MHz clock). After confirming that the H_NegoControl_0.AutoModeCancel bit is cleared to 0, write 0x01 to the same register (which sets the host state transition execution (H_NegoControl_0.AutoMode) to GoIDLE). This way, the processes required for a transition to IDLE will be automatically executed by the LSI's hardware.

Processes (3) to (8) are automatically executed by the LSI's hardware.

(1) Write 0x80 to the H_NegoControl_0 register (H_NegoControl_0.AutoModeCancel = 1 and H_NegoControl_0.AutoMode = 0x0) (T0).

(2) After confirming that the H_NegoControl_0.AutoModeCancel bit is cleared to 0, write 0x01 to the H_NegoControl_0 register (H_NegoControl_0.AutoMode = 0x1) (T1).

(3) The host state monitor (H_NegoControl_0.HostState) is set to IDLE (T1).

(4) VBUSEN_A is turned off (T1).

(5) Transceiver selection (H_XcvrControl.XcvrSelect) and terminal selection (H_XcvrControl.TermSelect) are set to FS mode (T1).

(6) Operation mode (H_XcvrControl.OpMode[1:0]) is set to NonDriving (T1).

(7) The transaction execution function of the USB host is immediately turned off (T1).

(8) All of the connection detection, disconnection detection, remote wakeup detection, and device chirp detection functions are turned off (T1).

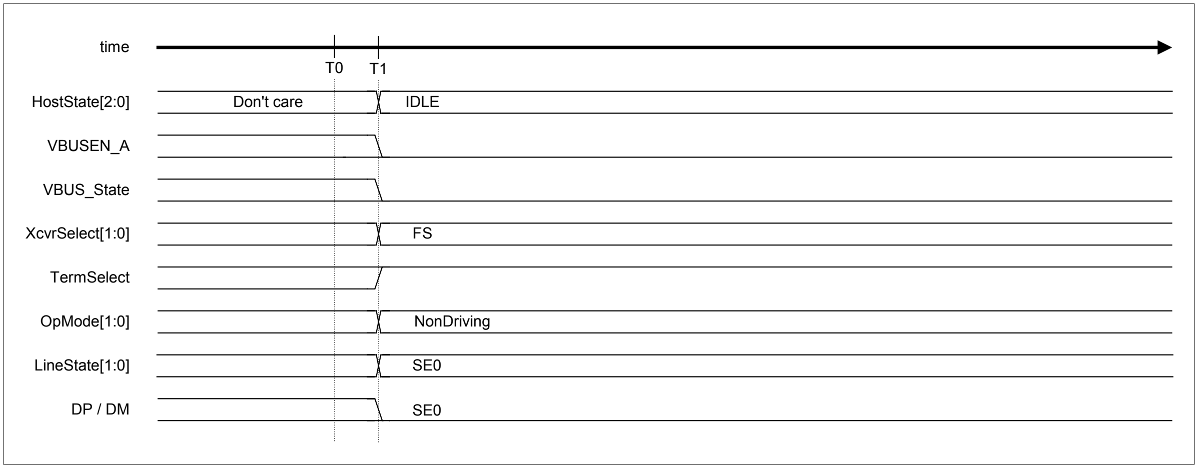


Fig. 6.45 GoIDLE timing

Table 6.37 GoIDLE Timing Values

Timing Parameter	Description	Value
T0	Host state transition execution is cancelled (by firmware).	0 (reference)
T1	After confirmation that the host state transition execution cancel bit is cleared to 0, the register is set to GoIDLE (by firmware). VBUSEN_A is turned off. Transceiver selection is set to FS mode. Terminal selection is set to FS mode. Operation mode is set to NonDriving. The transaction execution function is immediately turned off. The connection detection, disconnection detection, remote wakeup detection, and device chirp detection functions are turned off (by hardware).	T0 + 5cycle(60MHz) < T1

6.3.9.3.2 GoWAIT_CONNECT

When the host state transition execution (H_NegoControl_0.AutoMode) is set to GoWAIT_CONNECT, the process required for transition to WAIT_CONNECT is automatically executed by the LSI's hardware.

Note that at this point, the HS device is connected as an FS device. It is made to operate as an HS device by HS Detection Handshaking that is performed in the subsequent reset operation.

6.3.9.3.2.1 When FS Device is Connected

The procedure that is executed when an FS device is connected is described below. Processes (2) to (12) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoWAIT_CONNECT (T0).

(2) The host state monitor (H_NegoControl_0.HostState) is set to WAIT_CONNECT (T0).

(3) VBUSEN_A is turned on (T0).

(4) Transceiver selection (H_XcvrControl.XcvrSelect) and terminal selection (H_XcvrControl.TermSelect) are set to FS mode (T0).

(5) Operation mode (H_XcvrControl.OpMode[1:0]) is set to PowerDown (T0).

(6) The port speed (H_NegoControl_1.PortSpeed[1:0]) is set to FS (T0).

(7) After waiting a while for the internal device power supply to stabilize, the connection detection function is turned on (T1).

(8) When an FS device is connected, “J” appears in the line state (H_USB_Status.LineState[1:0]) (T2).

(9) A connection of an FS device is assumed by a fact that the line state “J” (indicated by H_USB_Status.LineState[1:0]) continues for 2.5 μs or more (T3).

(10) The disconnection detection function is turned on (T3).

(11) If a disconnection is not detected during the debounce interval period, a connection detected status (SIE_IntStat_0.DetectCon) is issued (T4).

(12) The disconnection detection function is turned off (T4).

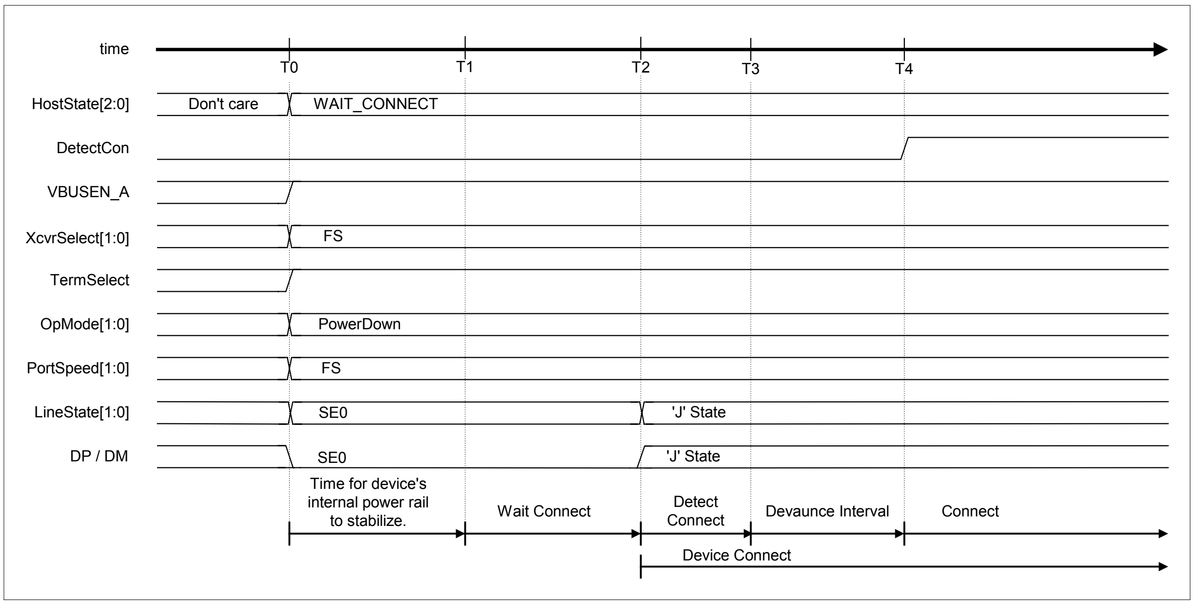


Fig. 6.46 Device attach timing (FS mode)

Table 6.38 Device Attach Timing Values (FS Mode)

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoWAIT_CONNECT (by firmware).	0 (reference)
T1	The connection detection function is turned on (by hardware).	T0+100ms{T_{SIGATT}} < T1
T2	A device is connected.	T2
T3	The connection detection function is turned off and the disconnection detection function is turned on (by hardware).	T2+2.5us{T_{DCNN}} < T3
T4	A connection detected status (DetectCon) is issued (by hardware). The disconnection detection function is turned off (by hardware).	T3+100ms{T_{ATTDB}} <T4

Note: Names stipulated in the USB2.0 Standard are shown in { }.

6.3.9.3.2.2 When LS Device is Connected

The procedure that is executed when an LS device is connected is described below. Processes (2) to (14) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoWAIT_CONNECT (T0).

(2) The host state monitor (H_NegoControl_0.HostState) is set to WAIT_CONNECT (T0).

(3) VBUSEN_A is turned on (T0).

(4) Transceiver selection (H_XcvrControl.XcvrSelect) and terminal selection (H_XcvrControl.TermSelect) are set to FS mode (T0).

(5) Operation mode (H_XcvrControl.OpMode[1:0]) is set to PowerDown (T0).

(6) The port speed (H_NegoControl_1.PortSpeed[1:0]) is set to FS (T0).

(7) After waiting 100 ms for the internal device power supply to stabilize, the connection detection function is turned on (T1).

(8) When an LS device is connected, "K" appears in the line state (H_USB_Status.LineState[1:0]) (T2).

(9) A connection of an LS device is assumed by a fact that the line state "K" (indicated by H_USB_Status.LineState[1:0]) continues for 2.5 μs or more (T3).

(10) Transceiver selection (H_XcvrControl.XcvrSelect) is set to LS (T3). As a result, the polarity of the line state (H_USB_Status.LineState[1:0]) changes to LS, and "J" appears in the line state (H_USB_Status.LineState[1:0]).

(11) The port speed (H_NegoControl_1.PortSpeed[1:0]) is set to LS (T3).

(12) The disconnection detection function is turned on (T3).

(13) If a disconnection is not detected during the debounce interval period, a connection detected status (SIE_IntStat_0.DetectCon) is issued (T4).

(14) The disconnection detection function is turned off (T4).

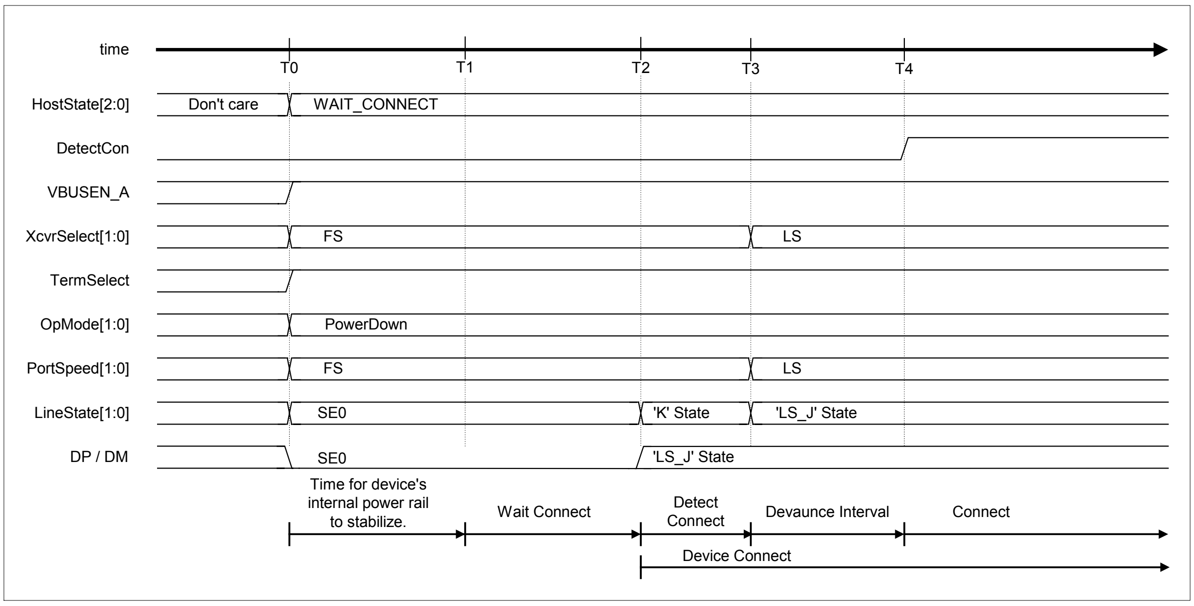


Fig. 6.47 Device attach timing (LS mode)

Table 6.39 Device Attach Timing Values (LS Mode)

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoWAIT_CONNECT (by firmware).	0 (reference)
T1	The connection detection function is turned on (by hardware).	T0+100ms{T_{SIGATT}} < T1
T2	A device is connected.	T2
T3	The connection detection function is turned off and the disconnection detection function is turned on (by hardware).	T2+2.5us{T_{DCNN}} < T3
T4	A connection detected status (DetectCon) is issued (by hardware). The disconnection detection function is turned off (by hardware).	T3+100ms{T_{ATTDB}} <T4

Note: Names stipulated in the USB2.0 Standard are shown in { }.

6.3.9.3.3 GoDISABLED

When the host state transition execution (H_NegoControl_0.AutoMode) is set to GoDISABLED, the processes required for a transition to DISABLED is automatically executed by the LSI's hardware.

The LSI shifts to this state when a connection is detected in the WAIT−CONNECT state, a Chirp from an erratic device is detected in the RESET state, or a port error is detected in the OPERATIONAL state.

6.3.9.3.3.1 When HS Device is Connected

The procedure that is executed when an HS device is connected is described below. Processes (2) to (6) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoDISABLED (T0).

(2) The host state monitor (H_NegoControl_0.HostState) is set to DISABLED (T0).

(3) The disconnection detection function is turned off (T0).

(4) After waiting for the currently executed transaction to complete, transceiver selection (H_XcvrControl.XcvrSelect) and terminal selection (H_XcvrControl.TermSelect) and the port speed (H_NegoControl_1.PortSpeed[1:0]) all are set to FS mode, and operation mode (H_XcvrControl.OpMode[1:0]) is set to PowerDown (T1).

(5) The disconnection detection function is turned on (T3).

(6) A transition-to-disabled complete status (H_SIE_IntStat_1.DisabledCmp) is issued to the firmware (T3).

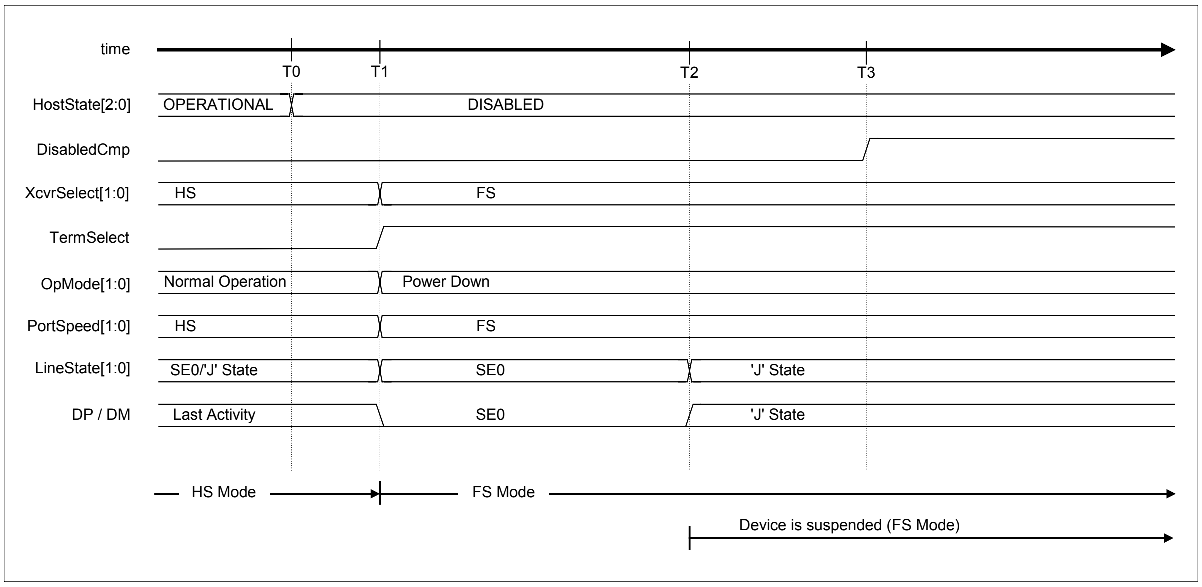


Fig. 6.48 Disabled timing (HS mode)

Table 6.40 Disabled Timing Values (HS Mode)

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoDISABLED (by firmware). The host state monitor (H_NegoControl_0.HostState) is set to DISABLED. The disconnection detection function is turned off (by hardware).	0 (reference)
T1	After last bus activity is complete, transceiver selection and terminal selection and port speed are set to FS mode, and operation mode (XcvrControl.OpMode[1:0]) is set to PowerDown (by hardware).	T1
T2	The device detects Suspend and shifts to FS mode.	T1 + 3.0ms < T2 {T_{WTREV}} < T1 + 3.125ms
T3	The disconnection detection function is turned on (by hardware). A transition-to-disabled complete status is issued (by hardware).	T1 + 4ms < T3

Note: Names stipulated in the USB2.0 Standard are shown in { }.

6.3.9.3.3.2 When FS Device is Connected

The procedure that is executed when an FS device is connected is described below. Processes (2) to (6) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoDISABLED (T0).

(2) The host state monitor (H_NegoControl_0.HostState) is set to DISABLED (T0).

(3) The disconnection detection function is turned off (T0).

(4) After waiting for the currently executed transaction to complete, operation mode (H_XcvrControl.OpMode[1:0]) is set to PowerDown (T1).

(5) The disconnection detection function is turned on (T3).

(6) A transition-to-disabled complete status (H_SIE_IntStat_1.DisabledCmp) is issued to the firmware (T3).

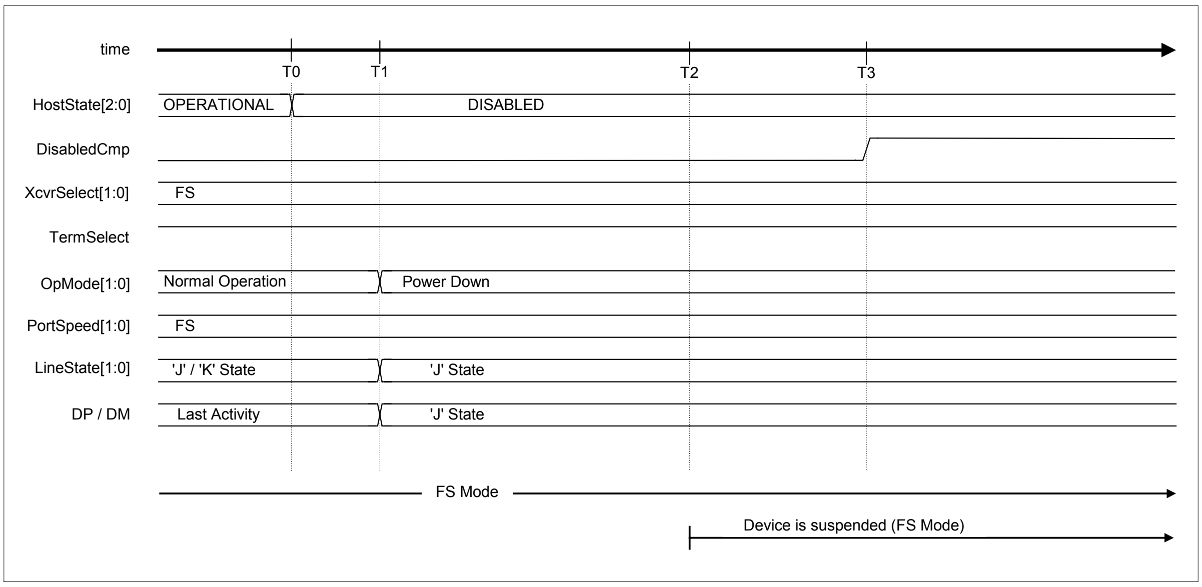


Fig. 6.49 Disabled timing (FS mode)

Table 6.41 Disabled Timing Values (FS Mode)

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoDISABLED (by firmware). The host state monitor (H_NegoControl_0.HostState) is set to SUSPEND. The disconnection detection function is turned off (by hardware).	0 (reference)
T1	After last bus activity is complete, operation mode (XcvrControl.OpMode[1:0]) is set to PowerDown (by hardware).	T1
T2	The device detects Suspend.	T1 + 3.0ms < T2 {T_{WTREV}} < T1 + 3.125ms
T3	The disconnection detection function is turned on (by hardware). A transition-to-disabled complete status is issued (by hardware).	T1 + 4ms < T3

Note: Names stipulated in the USB2.0 Standard are shown in { }.

6.3.9.3.3.3 When LS Device is Connected

The procedure that is executed when an LS device is connected is described below. Processes (2) to (6) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoDISABLED (T0).

(2) The host state monitor (H_NegoControl_0.HostState) is set to DISABLED (T0).

(3) The disconnection detection function is turned off (T0).

(4) After waiting for the currently executed transaction to complete, operation mode (H_XcvrControl.OpMode[1:0]) is set to PowerDown (T1).

(5) The disconnection detection function is turned on (T3).

(6) A transition-to-disabled complete status (H_SIE_IntStat_1.DisabledCmp) is issued to the firmware (T3).

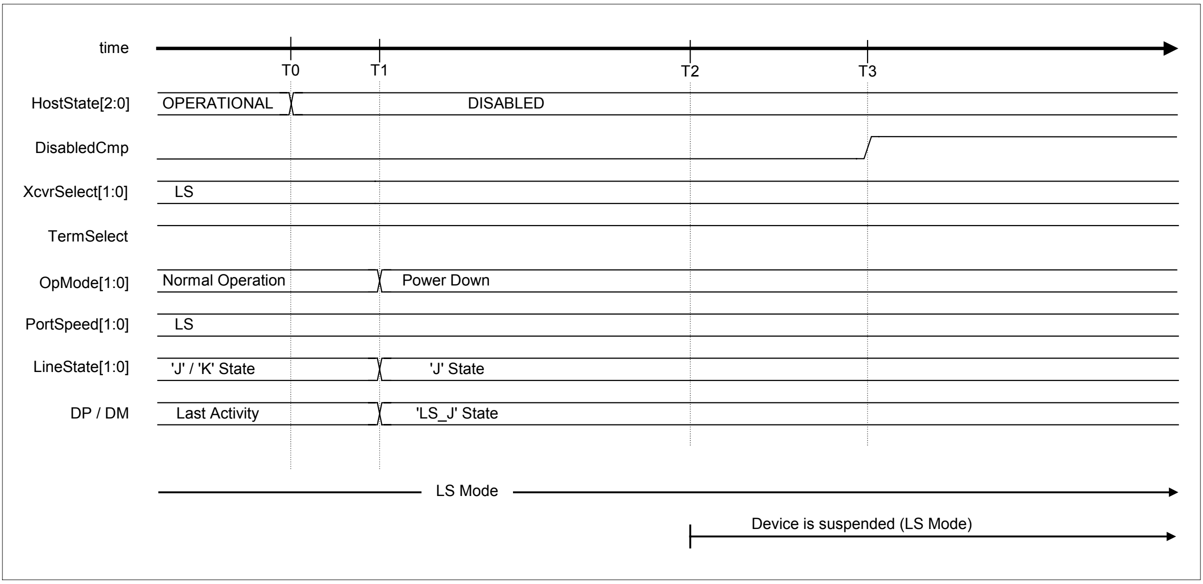


Fig. 6.50 Disabled timing (LS mode)

Table 6.42 Disabled Timing Values (LS Mode)

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoDISABLED (by firmware). The host state monitor (USB_Control_0.HostState) is set to DISABLED. The disconnection detection function is turned off (by hardware).	0 (reference)
T1	After last bus activity is complete, operation mode (XcvrControl.OpMode[1:0]) is set to PowerDown (by hardware).	T1
T2	The device detects Suspend.	T1 + 3.0ms < T2 {T_{WTREV}} < T1 + 3.125ms
T3	The disconnection detection function is turned on (by hardware). A transition-to-disabled complete status is issued (by hardware).	T1 + 4ms < T3

Note: Names stipulated in the USB2.0 Standard are shown in { }.

6.3.9.3.4 GoRESET

When the host state transition execution (H_NegoControl_0.AutoMode) is set to GoRESET, the processes required for a transition to RESET is automatically executed by the LSI's hardware. When transition to this state is to be executed from the OPERATIONAL state, the LSI will wait for the completion of the transaction under execution by the hardware and then start the RESET processing.

6.3.9.3.4.1 Reset for an HS Device

The procedure that is executed when an HS device is to be reset is described below. Processes (2) to (14) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoRESET (T0).

(2) The host state monitor (H_NegoControl_0.HostState) is set to RESET (T0).

(3) Transceiver selection (H_XcvrControl.XcvrSelect) and terminal selection (H_XcvrControl.TermSelect) are set to HS mode (T0).

(4) Operation mode (H_XcvrControl.OpMode[1:0]) is set to Normal (T0).

(5) The device chirp detection function is turned on (T0).

(6) A device Chirp is recognized by a line state (H_USB_Status.LineState[1:0]) that indicates presence of activity (seen as 'J' state) continuously for 2.5 μs or more. Then, when the device Chirp is determined to have ended within a specified time after a reset started (i.e., the line state (USB_Host_Status.LineState[1:0]) has changed to 'SE0'), a device chirp normal detection status (H_SIE_IntStat_0.DetectDevChirpOK) is issued (T2).

(7) The device chirp detection function is turned off (T3).

(8) After the device Chirp is complete, the host starts the output of Chirp K (T3).

(9) The host switches its output from Chirp K to Chirp J (T4)

(10) The host switches its output from Chirp J to Chirp K (T5). Thereafter, the host outputs Chirp K and Chirp J sequences alternately.

(11) Upon detecting a host Chirp, the device shifts to HS mode (T6). The fact that the Chirp level changed beginning with T7 means that the HS termination on the device side is enabled. Normally, Chirp is approximately 800 mV when the device is in FS mode, and approximately 400 mV when the device is in HS mode.

(12) The host completes Chirp (T8).

(13) Reset is complete (T9).

(14) A reset complete status (H_SIE_IntStat_1.ResetCmp) is issued (T9).

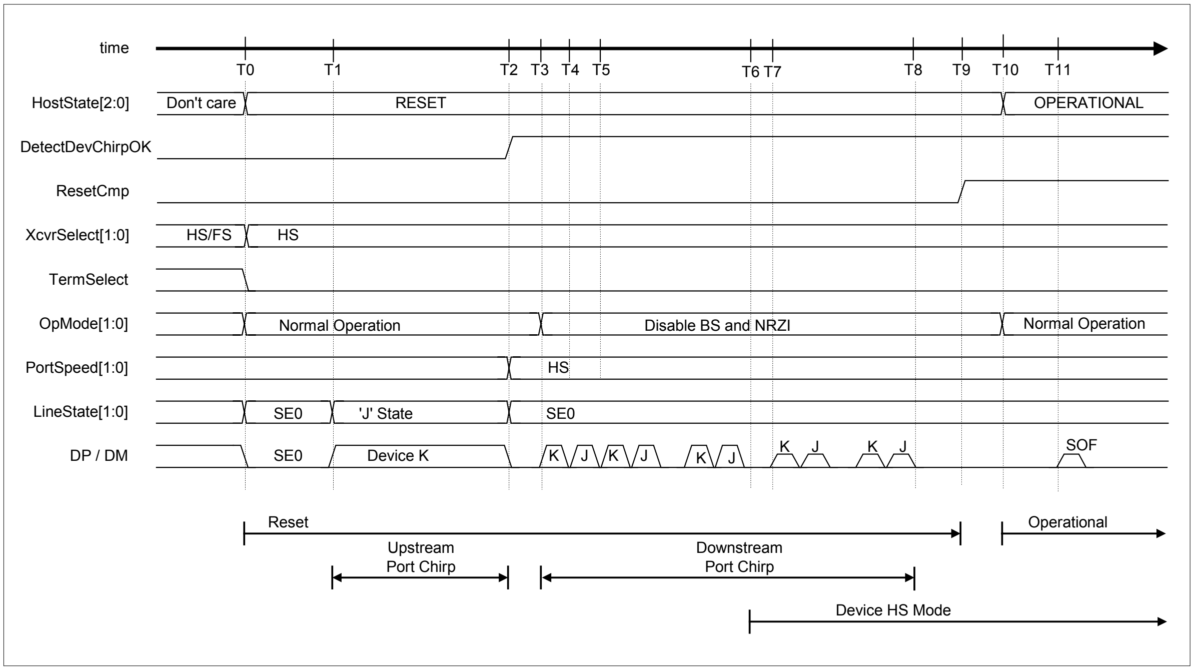


Fig. 6.51 Reset timing (HS mode)

Table 6.43 Reset Timing Values (HS Mode)

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoRESET (by firmware). The device chirp detection function is turned on (by hardware).	0 (reference)
T1	The device starts a Chirp.	T0 < T1 < T0 + 6.0ms
T2	The device completes a Chirp. The port speed is set to HS. The device chirp detection function is turned off. A device chirp normal detection status (DetectDevChirpOK) is issued (by hardware).	T1 + 1.0ms {T_{UCH}} < T2 < T0 + 7.0ms {T_{UCHEND}}
T3	The host outputs the first Chirp (Chirp K) (in hardware).	T2 < T3 < T2 + 100us {T_{WTDCH}}
T4	The host switches its output from Chirp K to Chirp J (in hardware).	T3 + 40us {T_{DCHBIT}} < T4 < T3 + 60us {T_{DCHBIT}}
T5	The host switches its output from Chirp J to Chirp K (in hardware).	T4 + 40us {T_{DCHBIT}} < T5 < T4 + 60us {T_{DCHBIT}}
T6	The device detects a host Chirp.	T6
T7	The device shifts to HS mode.	T6 < T7 < T6 + 500us
T8	The host completes Chirp (in hardware).	T3+ 50ms{T_{DRSTR}} < T8
T9	After reset, a reset completed status (ResetCmp) is issued (by hardware).	T8 < T9 <T8 + 150us
T10 (reference)	H_NegoControl_0.AutoMode is set to GoOPERATIONAL (by firmware).	T10 < T9 + 200us
T11 (reference)	First SOF is sent out (by hardware).	T10 + 120us < T11 < T10 + 130us T8 + 100us{T_{DCHSE0}} < T11 < T8 + 500us {T_{DCHSE0}}

Note: Names stipulated in the USB2.0 Standard are shown in { }.

6.3.9.3.4.2 Erratic Device Chirp Detected

Behavior of the LSI when a device Chirp is found erratic in HS Detection Handshaking is shown below. Two operation modes are available to choose from depending on how the chirp complete disable (H_NegoControl_1.DisChirpFinish) is set.

6.3.9.3.4.2.1 When Chirp Complete Disable (H_NegoControl_1.DisChirpFinish) = 0

A host Chirp is not performed after an error is detected. If a device chirp abnormal detection status is issued, the firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoDISABLED after waiting for a reset complete status (H_SIE_IntStat_1.ResetCmp) to be issued, thereby shifting the host into the DISABLED state. Processes (2) to (9) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoRESET (T0).

(2) The host state monitor (H_NegoControl_0.HostState) is set to RESET (T0).

(3) Transceiver selection (H_XcvrControl.XcvrSelect) and terminal selection (H_XcvrControl.TermSelect) are set to HS mode (T0).

(4) Operation mode (H_XcvrControl.OpMode[1:0]) is set to Normal (T0).

(5) The device chirp detection function is turned on (T0).

(6) A device Chirp is recognized by a line state (H_USB_Status.LineState[1:0]) that indicates presence of activity (seen as 'J' state) continuously for 2.5 μs or more. However, because the device Chirp does not end within a specified time after a reset started, an error is assumed and a device chirp abnormal detection status (H_SIE_IntStat_0.DetectDevChirpNG) is issued (T2).

(7) The device chirp detection function is turned off (T2).

(8) Reset is complete (T3).

(9) A reset completed status (H_SIE_IntStat_1.ResetCmp) is issued (T3).

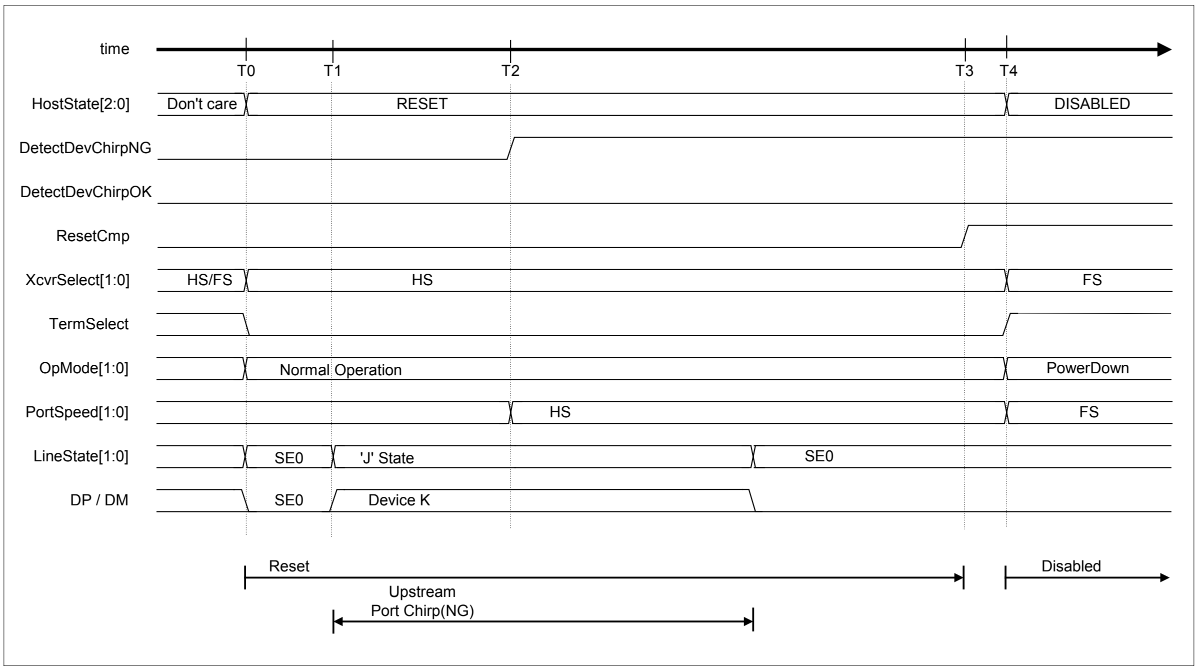


Fig. 6.52 Detect device chirp NG timing (DisChirpFinish = 0)

Table 6.44 Detect Device Chirp NG Timing Values (DisChirpFinish = 0)

Timing Parameter	Description	Value
T0	USB_Control_0. AutoMode is set to GoRESET (by firmware). The device chirp detection function is turned on (by hardware).	0 (reference)
T1	The device starts a Chirp.	T0 < T1 < T0 + 6.0ms
T2	A device chirp abnormal detection status (DetectDevChirpNG) is issued. The device chirp detection function is turned off (by hardware).	T0 + 7ms{T_{UCHEND}} < T2
T3	Reset is complete. A reset complete status (ResetCmp) is issued (by hardware).	T2 + 50ms{T_{DRSTR} }< T3
T4 (reference)	H_NegoControl_0.AutoMode is set to GoDISABLED (by firmware).	

Note: Names stipulated in the USB2.0 Standard are shown in { }.

6.3.9.3.4.2.2 When Chirp Complete Disable (H_NegoControl_1.DisChirpFinish) = 1

A host Chirp is performed after an error is detected.

When using this mode and executing transition of the host state to DISABLED without waiting for the reset complete status (H_SIE_IntStat_1.ResetCmp) to be issued, write 0x80 to the H_NegoControl_0 register (H_NegoControl_0.AutoModeCancel=1 and H_NegoControl_0.AutoMode=0x0) to cause operation of the state under execution to stop. The H_NegoControl_0.AutoModeCancel bit is cleared to 0 by a completion of the stop processing (6 cycles required when operating with 60 MHz clock). After confirming that the H_NegoControl_0.AutoModeCancel bit has been cleared to 0, write 0x03 to the same register (which sets the host state transition execution (H_NegoControl_0.AutoMode) to GoDISABLED).

Processes (2) to (15) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoRESET (T0).

(2) The host state monitor (H_NegoControl_0.HostState) is set to RESET (T0).

(3) Transceiver selection (H_XcvrControl.XcvrSelect) and terminal selection (H_XcvrControl.TermSelect) are set to HS mode (T0).

(4) Operation mode (H_XcvrControl.OpMode[1:0]) is set to Normal (T0).

(5) The device chirp detection function is turned on (T0).

(6) A device Chirp is recognized by a line state (H_USB_Status.LineState[1:0]) that indicates presence of activity (shown as 'J' state) continuously for 2.5 μs or more. However, because the device Chirp does not end within a specified time after a reset started, an error is assumed and a device chirp abnormal detection status (H_SIE_IntStat_0.DetectDevChirpNG) is issued (T2).

(7) The device chirp detection function is turned off (T2).

(8) When the device Chirp is determined to have ended by a line state (USB_Host_Status.LineState[1:0]) that indicates absence of activity (shown as 'SE0'), a device chirp normal detection status (H_SIE_IntStat_0.DetectDevChirpOK) is issued (T3).

(9) After the device Chirp is complete, the host starts the output of Chirp K (T4).

(10) The host switches its output from Chirp K to Chirp J (T5).

(11) The host switches its output from Chirp J to Chirp K (T6). Thereafter, the host outputs Chirp K and Chirp J sequences alternately.

(12) Upon detecting a host Chirp, the device shifts to HS mode (T7). The fact that the Chirp level changed beginning with T8 means that the HS termination on the device side is enabled. Normally, Chirp is approximately 800 mV when the device is in FS mode, and approximately 400 mV when the device is in HS mode.

(13) The host completes Chirp (T9).

(14) Reset is complete (T10).

(15) A reset complete status (H_SIE_IntStat_1.ResetCmp) is issued (T10).

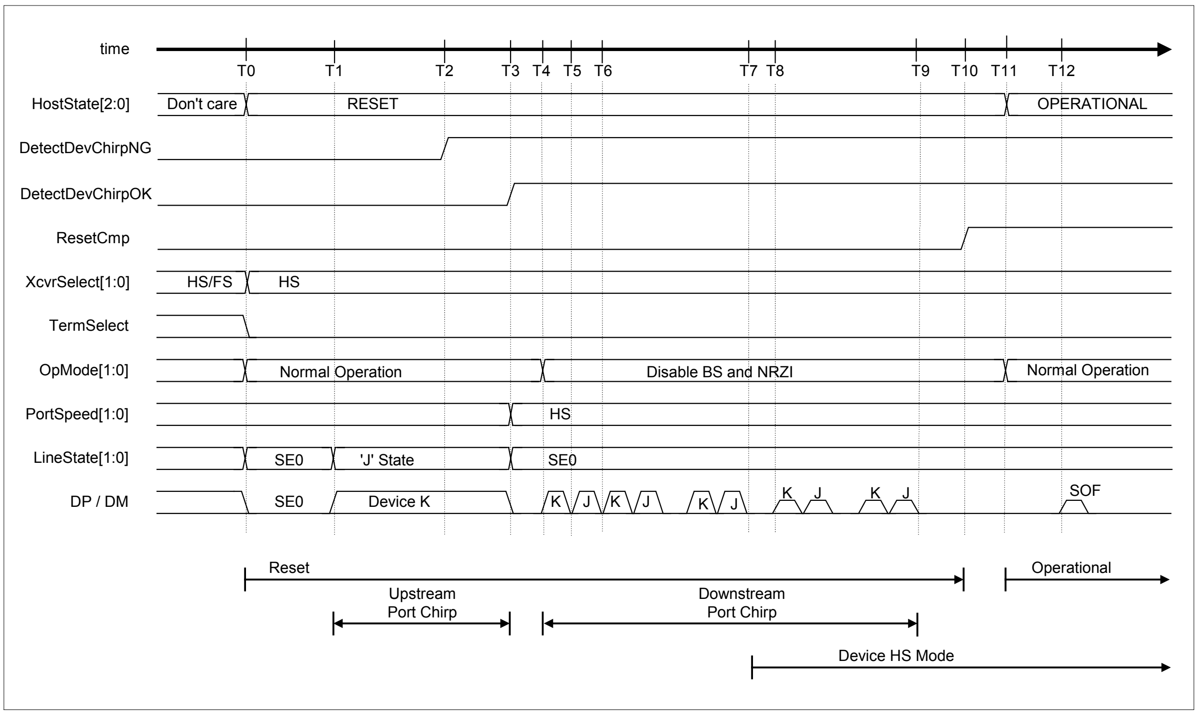


Fig. 6.53 Detect device chirp NG timing (DisChirpFinish = 1)

Table 6.45 Detect Device Chirp NG Timing Values (DisChirpFinish = 1)

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoRESET (by firmware). The device chirp detection function is turned on (by hardware).	0 (reference)
T1	The device starts a Chirp.	T0 < T1 < T0 + 6.0ms
T2	A device chirp abnormal detection status (DetectDevChirpNG) is issued. The device chirp detection function is turned off (by hardware).	T0 + 7ms{T_{UCHEND}} < T2
T3	The device completes a Chirp. The port speed is set to HS. A device chirp normal detection status (DetectDevChirpOK) is issued (by hardware).	T3
T4	The host outputs the first Chirp (Chirp K) (in hardware).	T3< T4 < T3 + 100us {T_{WTDCH}}
T5	The host switches its output from Chirp K to Chirp J (in hardware).	T4 + 40us {T_{DCHBIT}} < T45< T4 + 60us {T_{DCHBIT}}
T6	The host switches its output from Chirp J to Chirp K (in hardware).	T5 + 40us {T_{DCHBIT}} < T6 < T5 + 60us {T_{DCHBIT}}
T7	The device detects a host Chirp.	T7
T8	The device shifts to HS mode.	T7 < T8 < T6 + 500us
T9	The host completes Chirp (in hardware).	T4 + 50ms{T_{DRSTR}} < T9
T10	Reset is complete. A reset complete status (ResetCmp) is issued (by hardware).	T9 < T10 <T9 + 150us
T11 (reference)	H_NegoControl_0.AutoMode is set to GoOPERATIONAL (by firmware).	T11< T10 + 200us
T12 (reference)	First SOF is sent out (by hardware).	T11 + 120us < T12 < T11 + 130us T9+ 100us{T_{DCHSE0}} < T12 < T9 + 500us {T_{DCHSE0}}

Note: Names stipulated in the USB2.0 Standard are shown in { }.

6.3.9.3.4.3 Reset for an FS Device

The procedure that is executed when an FS device is to be reset is described below. Processes (2) to (9) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoRESET (T0).

(2) The host state monitor (H_NegoControl_0.HostState) is set to RESET (T0).

(3) Transceiver selection (H_XcvrControl.XcvrSelect) and terminal selection (H_XcvrControl.TermSelect) are set to HS mode (T0).

(4) Operation mode (H_XcvrControl.OpMode[1:0]) is set to Normal (T0).

(5) The device chirp detection function is turned on (T0).

(6) Because no device Chirp is detected and the port speed (H_NegoControl_1.PortSpeed[1:0]) = HS/FS, the other party device is assumed to be an FS device, setting the transceiver selection (H_XcvrControl.XcvrSelect) and port speed (H_NegoControl_1.PortSpeed[1:0]) to FS (T1).

(7) The device chirp detection function is turned off (T1).

(8) Terminal selection (H_XcvrControl.TermSelect) is set to FS (T2).

(9) A reset complete status (H_SIE_IntStat_1.ResetCmp) is issued (T3).

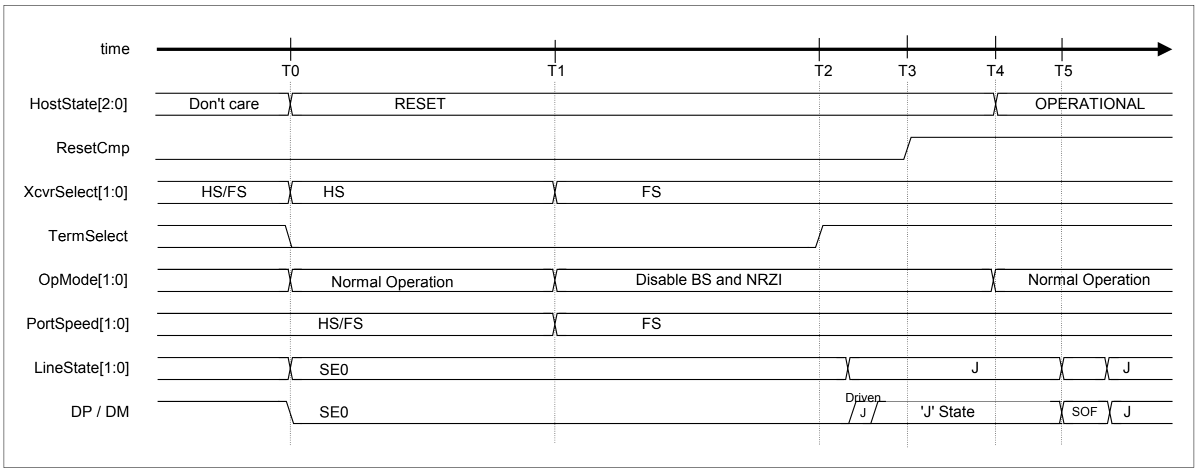


Fig. 6.54 Reset timing (FS mode)

Table 6.46 Reset Timing Values (FS mode)

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoRESET (by firmware). The device chirp detection function is turned on (by hardware).	0 (reference)
T1	Transceiver selection is set to FS. Port speed is set to FS. The device chirp detection function is turned off (by hardware).	T0 + 7.0ms{T_{UCHEND}} < T1
T2	Terminal selection is set to FS (by hardware).	T0 + 50ms{T_{DRSTR}} < T2
T3	A reset complete status is issued (by hardware).	T2 + 150us < T3
T4 (reference)	USB_Control_0.AutoMode is set to GoOPERATIONAL (by firmware).	T4
T5 (reference)	First SOF is sent out (by hardware).	T4 + 0.9ms < T5 < T4 + 1.1ms (T5 < T2 + 3ms)

Note: Names stipulated in the USB2.0 Standard are shown in { } .

6.3.9.3.4.4 Reset for an LS Device

The procedure that is executed when an LS device is to be reset is described below. Processes (2) to (7) below are automatically executed by the LSI’s hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoRESET (T0).

(2) The host state monitor (H_NegoControl_0.HostState) is set to RESET (T0).

(3) Transceiver selection (H_XcvrControl.XcvrSelect) and terminal selection (H_XcvrControl.TermSelect) are set to HS mode (T0).

(4) Operation mode (H_XcvrControl.OpMode[1:0]) is set to Normal (T0).

(5) Because the port speed (H_NegoControl_1.PortSpeed[1:0]) = LS, the other party device is assumed to be an LS device, so that transceiver selection (H_XcvrControl.XcvrSelect) is set to LS (T1).

(6) Terminal selection (H_XcvrControl.TermSelect) is set to FS (T2).

(7) A reset complete status (H_SIE_IntStat_1.ResetCmp) is issued (T3).

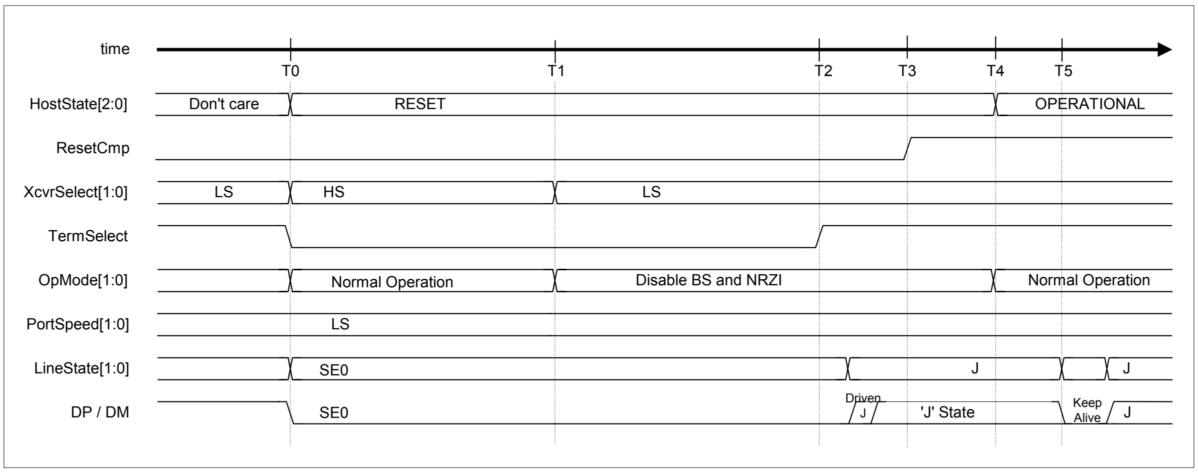


Fig. 6.55 Reset timing (LS mode)

Table 6.47 Reset Timing Values (LS Mode)

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoRESET (by firmware).	0 (reference)
T1	Transceiver selection is set to LS (by hardware).	0 (reference)
T2	Terminal selection is set to FS (by hardware).	T0 + 7.0ms{T_{UCHEND}} < T1
T3	A reset complete status is issued (by hardware).	T0 + 50ms{T_{DRSTR}} < T2
T4 (reference)	USB_Control_0. AutoMode is set to GoOPERATIONAL (by firmware).	T4
T5 (reference)	First KeepAlive is sent out (by hardware).	T3 + 1ms < T4

Note: Names stipulated in the USB2.0 Standard are shown in { } .

6.3.9.3.5 GoOPERATIONAL

When the host state transition execution (H_NegoControl_0.AutoMode) is set to GoOPERATIONAL, the processes required for a transition to OPERATIONAL is automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoOPERATIONAL (T0).

(2) The host state monitor (H_NegoControl_0.HostState) is set to OPERATIONAL (T0).

(3) Operation mode (H_XcvrControl.OpMode[1:0]) is set to Normal and a state is thereby entered in which USB transactions can be executed (T0).

(4) The disconnection detection function is turned on (T0).

(5) If the port speed (H_NegoControl_1.PortSpeed[1:0]) is set to HS or FS, the first SOF is issued; if the port speed is set to LS, the first KeepAlive is issued (T1). Thereafter, transfers are performed according to channel settings.

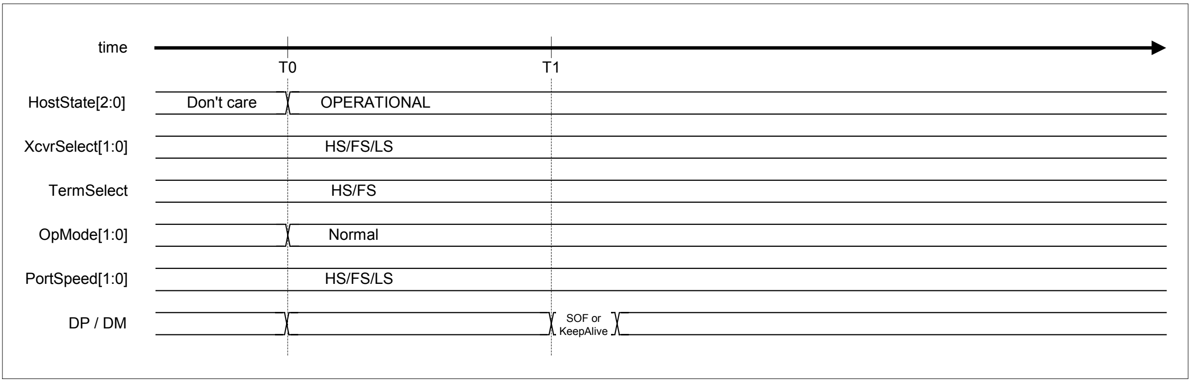


Fig. 6.56 GoOPERATIONAL timing

Table 6.48 GoOPERATIONAL Timing Values

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoOPERATIONAL (by firmware).	0 (reference)
T1	First SOF (HS/FS) or first KeepAlive (LS) is issued.	T0+120us < T1(HS) < T0 + 130us T0+0.9ms <T1(FS,LS) < T0 + 1.1ms

6.3.9.3.6 GoSUSPEND

When the host state transition execution (H_NegoControl_0.AutoMode) is set to GoSUSPEND, the processes required for a transition to SUSPEND is automatically executed by the LSI's hardware.

6.3.9.3.6.1 When HS Device is Connected

The procedure that is executed when an HS device is connected is described below. Processes (2) to (7) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoSUSPEND (T0).

(2) The host state monitor (H_NegoControl_0.HostState) is set to SUSPEND (T0).

(3) The disconnection detection and remote wakeup detection functions are turned off (T0).

(4) After waiting for the currently executed transaction to complete, transceiver selection (H_XcvrControl.XcvrSelect) and terminal selection (H_XcvrControl.TermSelect) are set to FS mode, and operation mode (H_XcvrControl.OpMode[1:0]) is set to PowerDown (T1).

(5) The disconnection detection function is turned on (T3).

(6) If the remote wakeup acceptance enable (H_NegoControl_1.RmtWkupDetEnb) is enabled, the remote wakeup detection function is turned on (T3).

(7) A transition-to-suspend complete status (H_SIE_IntStat_1.SuspendCmp) is issued to the firmware (T3).

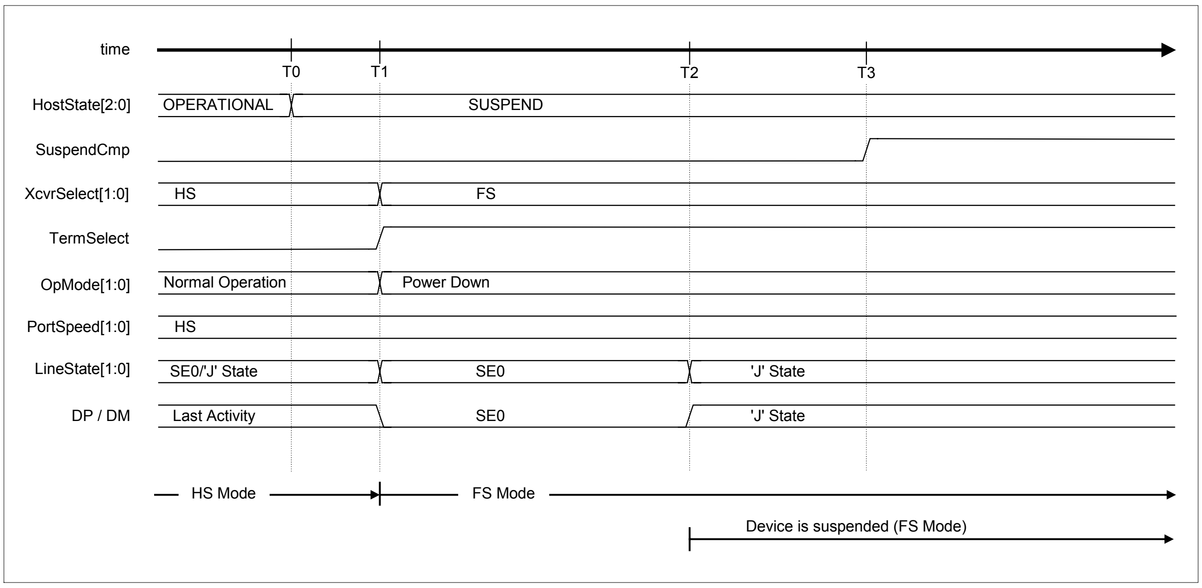


Fig. 6.57 Suspend timing (HS mode)

Table 6.49 Suspend Timing Values (HS Mode)

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoSUSPEND (by firmware). The host state monitor (H_NegoControl_0.HostState) is set to SUSPEND. The disconnection detection and remote wakeup detection functions are turned off (by hardware).	0 (reference)
T1	After last bus activity is complete, transceiver selection and terminal selection are set to FS mode, and operation mode (XcvrControl.OpMode[1:0]) is set to PowerDown (by hardware).	T1
T2	The device detects Suspend and shifts to FS mode.	T1 + 3.0ms < T2 {T_{WTREV}} < T1 + 3.125ms
T3	The disconnection detection function is turned on. If the remote wakeup acceptance enable is enabled, the remote wakeup detection function is turned on. A transition-to-suspend complete status is issued (by hardware).	T1 + 5ms {T_{WTRSM}} < T3

Note: Names stipulated in the USB2.0 Standard are shown in { }.

6.3.9.3.6.2 When FS Device is Connected

The procedure that is executed when an FS device is connected is described below. Processes (2) to (7) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoSUSPEND (T0).

(2) The host state monitor (H_NegoControl_0.HostState) is set to SUSPEND (T0).

(3) The disconnection detection and remote wakeup detection functions are turned off (T0).

(4) After waiting for the currently executed transaction to complete, operation mode (H_XcvrControl.OpMode[1:0]) is set to PowerDown (T1).

(5) The disconnection detection function is turned on (T3).

(6) If the remote wakeup acceptance enable (H_NegoControl_1.RmtWkupDetEnb) is enabled, the remote wakeup detection function is turned on (T3).

(7) A transition-to-suspend complete status (H_SIE_IntStat_1.SuspendCmp) is issued to the firmware (T3).

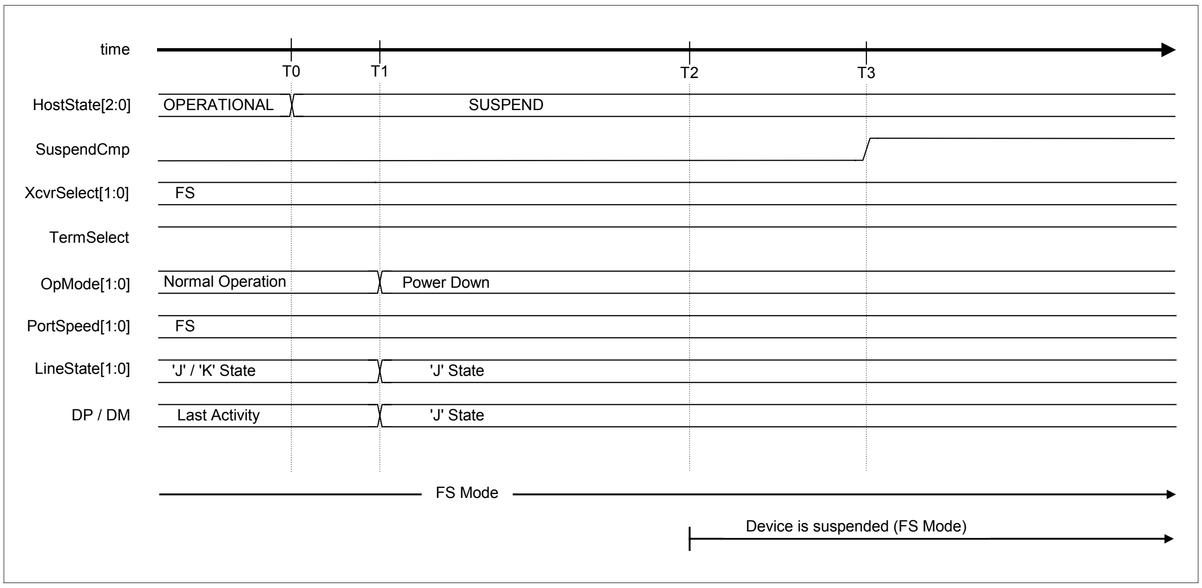


Fig. 6.58 Suspend timing (FS mode)

Table 6.50 Suspend Timing Values (FS Mode)

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoSUSPEND (by firmware). The host state monitor (H_NegoControl_0.HostState) is set to SUSPEND. The disconnection detection and remote wakeup detection functions are turned off (by hardware).	0 (reference)
T1	After last bus activity is complete, operation mode (XcvrControl.OpMode[1:0]) is set to PowerDown (by hardware).	T1
T2	The device detects Suspend.	T1 + 3.0ms < T2 {T_{WTREV}} < T1 + 3.125ms
T3	The disconnection detection function is turned on. If the remote wakeup acceptance enable is enabled, the remote wakeup detection function is turned on. A transition-to-suspend complete status is issued (by hardware).	T1+ 5ms {T_{WTRSM}} < T3

Note: Names stipulated in the USB2.0 Standard are shown in { }.

6.3.9.3.6.3 When LS Device is Connected

The procedure that is executed when an LS device is connected is described below. Processes (2) to (7) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoSUSPEND (T0).

(2) The host state monitor (H_NegoControl_0.HostState) is set to SUSPEND (T0).

(3) The disconnection detection and remote wakeup detection functions are turned off (T0).

(4) After waiting for the currently executed transaction to complete, operation mode (H_XcvrControl.OpMode[1:0]) is set to PowerDown (T1).

(5) The disconnection detection function is turned on (T3).

(6) If the remote wakeup acceptance enable (H_NegoControl_1.RmtWkupDetEnb) is enabled, the remote wakeup detection function is turned on (T3).

(7) A transition-to-suspend complete status (H_SIE_IntStat_1.SuspendCmp) is issued to the firmware (T3).

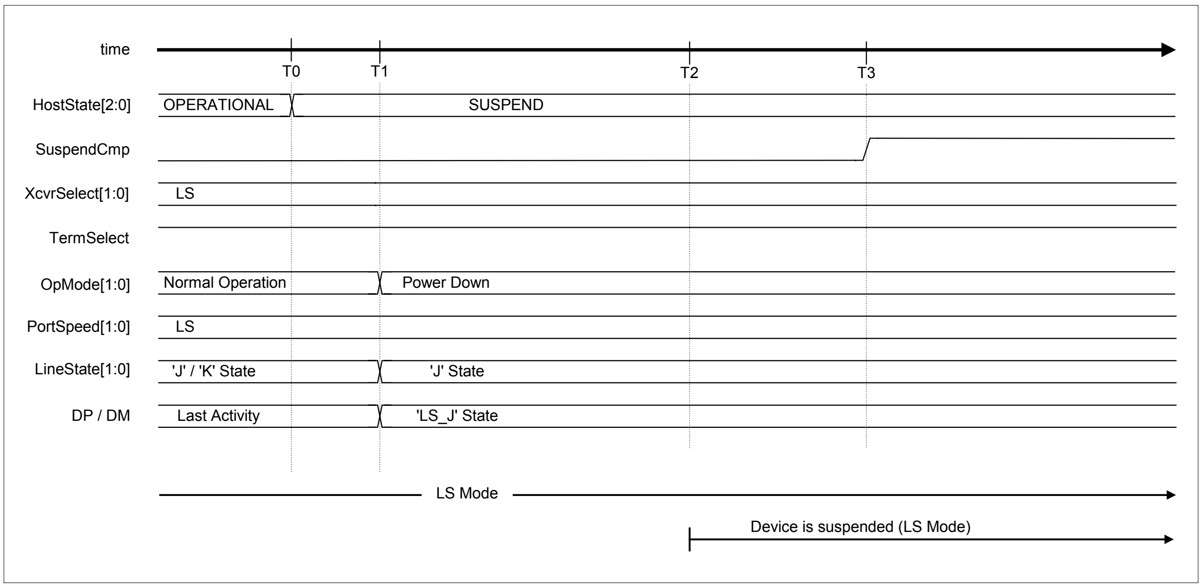


Fig. 6.59 Suspend timing (LS mode)

Table 6.51 Suspend Timing Values (LS Mode)

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoSUSPEND (by firmware). The host state monitor (H_NegoControl_0.HostState) is set to SUSPEND. The disconnection detection and remote wakeup detection functions are turned off (by hardware).	0 (reference)
T1	After last bus activity is complete, operation mode (XcvrControl.OpMode[1:0]) is set to PowerDown (by hardware).	T1
T2	The device detects Suspend.	T1 + 3.0ms < T2 {T_{WTREV}} < T1 + 3.125ms
T3	The disconnection detection function is turned on. If the remote wakeup acceptance enable is enabled, the remote wakeup detection function is turned on. A transition-to-suspend complete status is issued (by hardware).	T1+ 5ms {T_{WTRSM}} < T3

Note: Names stipulated in the USB2.0 Standard are shown in { }.

6.3.9.3.7 GoRESUME

When the host state transition execution (H_NegoControl_0.AutoMode) is set to GoRESUME, the processes required for a transition to RESUME is automatically executed by the LSI's hardware.

6.3.9.3.7.1 When HS Device is Connected

The procedure that is executed when an HS device is connected is described below. Processes (2) to (8) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoRESUME (T0).

(2) The host state monitor (H_NegoControl_0.HostState) is set to RESUME (T0).

(3) The disconnection detection and remote wakeup detection functions are turned off (T0).

(4) Operation mode (H_XcvrControl.OpMode) is set to Disable BS and NRZI, and the hardware thereby starts issuing a resume "K" signal (T0).

(5) The hardware finishes issuing the resume "K" signal (T1).

(6) Terminal selection (H_XcvrControl.TermSelect) is set to HS (T2).

(7) Transceiver selection (H_XcvrControl.XcvrSelect) is set to HS (T3).

(8) A resume complete status (SIE_IntStat_1.ResumeCmp) is issued to the firmware (T3).

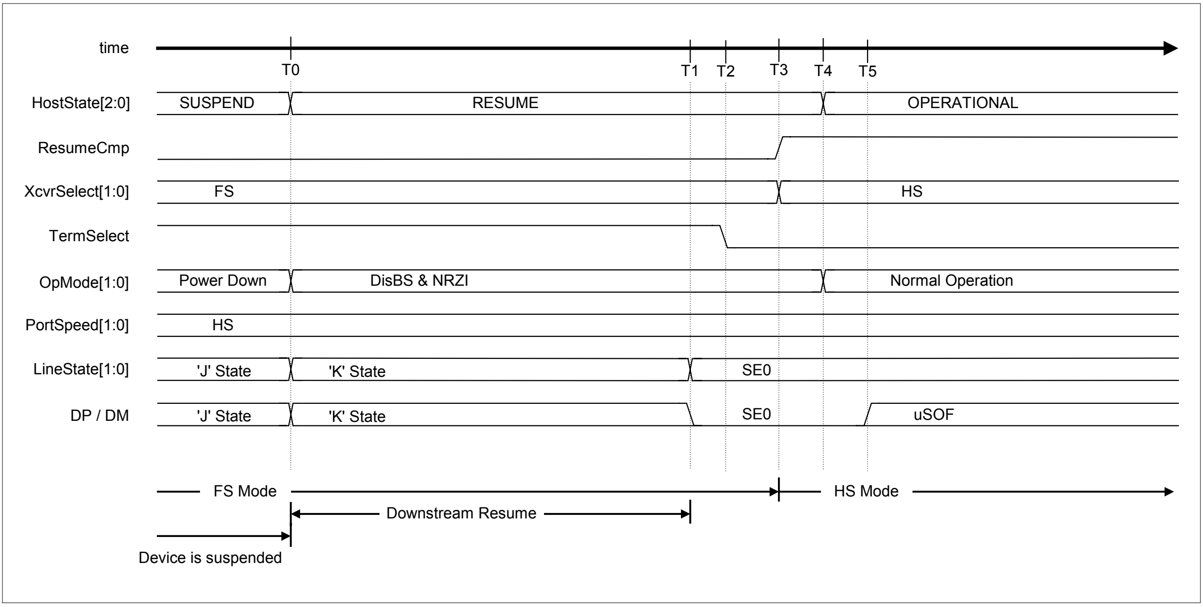


Fig. 6.60 Resume timing (HS mode)

Table 6.52 Resume Timing Values (HS Mode)

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoSUSPEND (by firmware). The host state monitor (USB_Control_0.HostState) is set to RESUME. The disconnection detection and remote wakeup detection functions are turned off. Operation mode is set to Disable BS and NRZI and the hardware starts issuing a resume "K" signal.	0 (reference)
T1	The hardware finishes issuing the resume "K" signal. Terminal selection is set to HS (by hardware).	T0 + 20ms{T_{DRSMDN}}< T1
T2	Transceiver selection is set to HS (by hardware).	T1 + 100us < T2 < T1 + 2.0us
T3	A resume complete status (SIE_IntStat_1.ResumeCmp) is issued (by hardware).	T1 + 90us < T3 < T1 + 110us
T4 (reference)	H_NegoControl_0.AutoMode is set to GoOPERATIONAL (by firmware) Operation mode is set to NormalOperation (by hardware).	T4
T5 (reference)	First micro SOF is issued (by hardware).	T5 < T1 + 3ms T4 + 120us < T5 < T4 + 130us

Note: Names stipulated in the USB2.0 Standard are shown in { }.

6.3.9.3.7.2 When FS Device is Connected

The procedure that is executed when an FS device is connected is described below. Processes (2) to (6) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoRESUME (T0).

(2) The host state monitor (H_NegoControl_0.HostState) is set to RESUME (T0).

(3) The disconnection detection and remote wakeup detection functions are turned off (T0).

(4) Operation mode (H_XcvrControl.OpMode) is set to Disable BS and NRZI, and the hardware thereby starts issuing a resume "K" signal (T0).

(5) The hardware finishes issuing the resume "K" signal (T1), with EOP in LS bit time appended at the end.

(6) A resume complete status (SIE_IntStat_1.ResumeCmp) is issued to the firmware (T2).

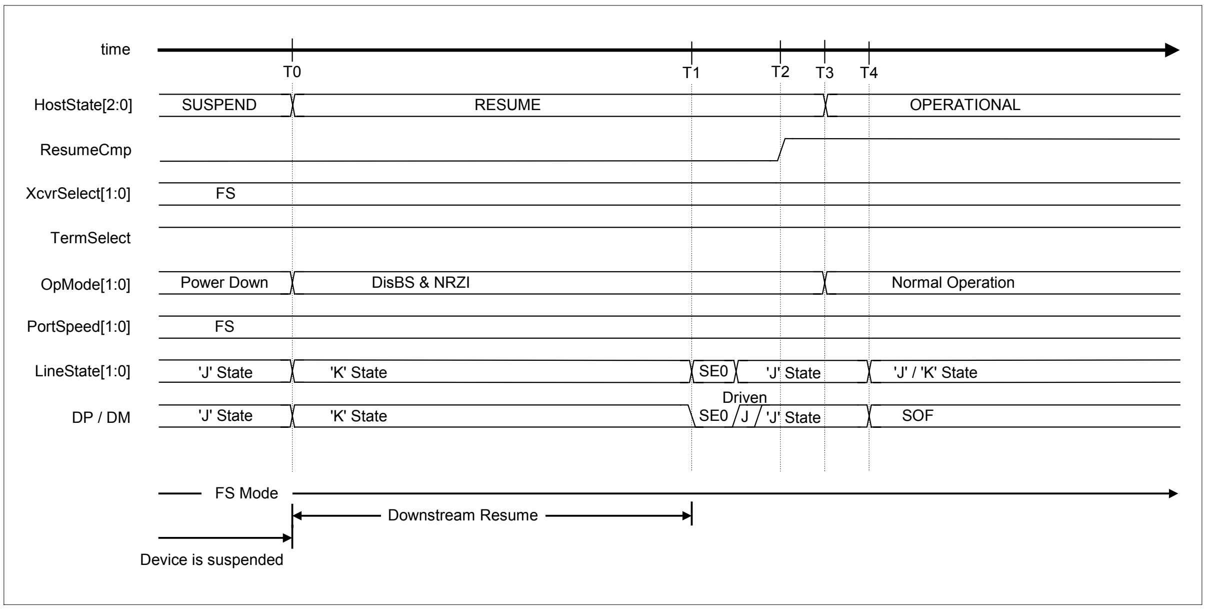


Fig. 6.61 Resume timing (FS mode)

Table 6.53 Resume Timing Values (FS Mode)

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoRESUME (by firmware). The host state monitor (H_NegoControl_0.HostState) is set to RESUME. The disconnection detection and remote wakeup detection functions are turned off. Operation mode is set to Disable BS and NRZI, and the hardware starts issuing a resume "K" signal.	0 (reference)
T1	The hardware finishes issuing the resume "K" signal, with EOP in LS bit time appended at the end.	T0 + 20ms {T_{DRSMDN}} < T1
T2	A resume complete status (SIE_IntStat_1.ResumeCmp) is issued (by hardware).	T1 + 90us < T2 < T1 + 110us
T3 (reference)	H_NegoControl_0.AutoMode is set to GoOPERATIONAL (by firmware) Operation mode is set to NormalOperation (by hardware).	T3
T4 (reference)	First SOF is issued (by hardware).	T4 < T1 + 3ms T3 + 0.9ms < T4 < T3 + 1.1ms

Note: Names stipulated in the USB2.0 Standard are shown in { }.

6.3.9.3.7.3 When LS Device is Connected

The procedure that is executed when an LS device is connected is described below. Processes (2) to (6) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoRESUME (T0).

(2) The host state monitor (H_NegoControl_0.HostState) is set to RESUME (T0).

(3) The disconnection detection and remote wakeup detection functions are turned off (T0).

(4) Operation mode (H_XcvrControl.OpMode) is set to Disable BS and NRZI, and the hardware thereby starts issuing a resume "K" signal (T0).

(5) The hardware finishes issuing the resume "K" signal (T1), with EOP in LS bit time appended at the end.

(6) A resume complete status (SIE_IntStat_1.ResumeCmp) is issued to the firmware (T2).

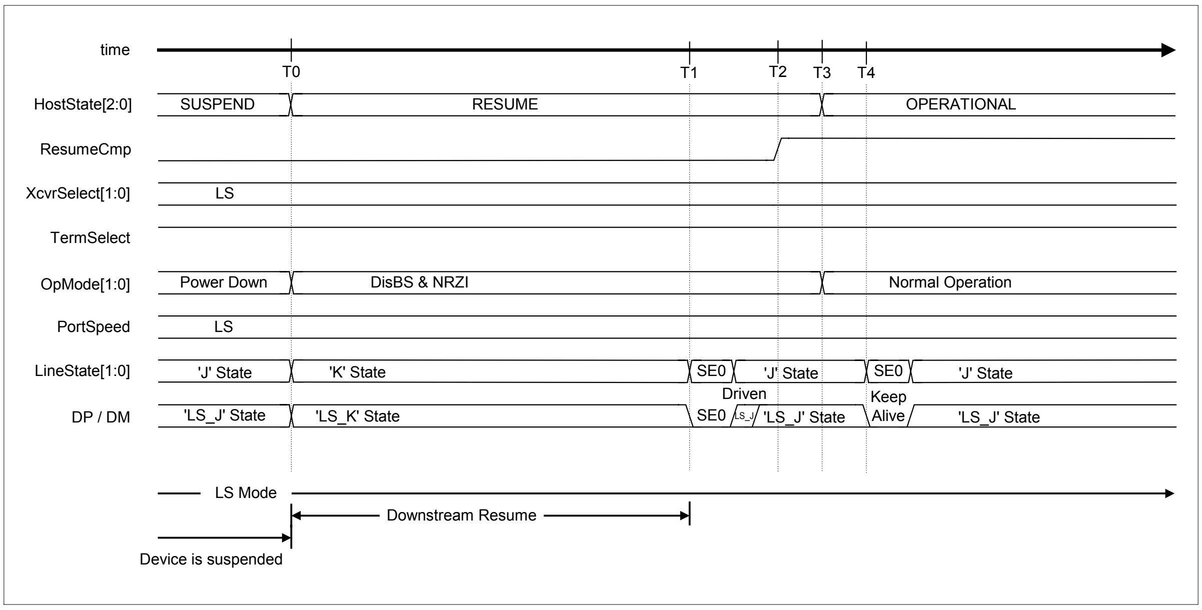


Fig. 6.62 Resume timing (LS mode)

Table 6.54 Resume Timing Values (FS Mode)

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoSUSPEND (by firmware). The host state monitor (H_NegoControl_0.HostState) is set to RESUME. The disconnection detection and remote wakeup detection functions are turned off. Operation mode is set to Disable BS and NRZI, and the hardware starts issuing a resume "K" signal.	0 (reference)
T1	The hardware finishes issuing the resume "K" signal, with EOP in LS bit time appended at the end.	T0 + 20ms {T_{DRSMDN}} < T1
T2	A resume complete status (SIE_IntStat_1.ResumeCmp) is issued (by hardware).	T1 + 90us < T2 < T1 + 110us
T3 (reference)	H_NegoControl_0.AutoMode is set to GoOPERATIONAL (by firmware) Operation mode is set to NormalOperation (by hardware).	T3
T4 (reference)	First KeepAlive is issued (by hardware).	T4 < T1 + 3ms T3 + 0.9ms < T4 < T3 + 1.1ms

Note: Names stipulated in the USB2.0 Standard are shown in { }.

6.3.9.3.8 GoWAIT_CONNECTtoDIS

When the host state transition execution (H_NegoControl_0.AutoMode) is set to GoWAIT_CONNECTtoDIS, the processes required for a transition from WAIT_CONNECT state to DISABLED state is automatically executed by the LSI's hardware.

The execution procedure is described below. Processes (2) to (5) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoWAIT_CONNECTtoDIS (T0).

(2) Process equivalent to GoWAIT_CONNECT is executed (T0).

(3) A connection detection is performed and a connection detected status (H_SIE_IntStat_0.DetectCon) is issued (T1).

(4) Process equivalent to GoDISABLED is executed (T1).

(5) A disabled complete status (H_SIE_IntStat_1.DisabledCmp) is issued (T2).

Note that the timing in each state here is the same as when GoWAIT_CONNECT and GoDISABLED are executed. For details about the timing, refer to the relevant sections on GoWAIT_CONNECT and GoDISABLED.

For the execution procedure and timing in cases when an error (disconnection or VBUS error) is detected during the process, refer to the relevant sections on disconnection detection and VBUS error.

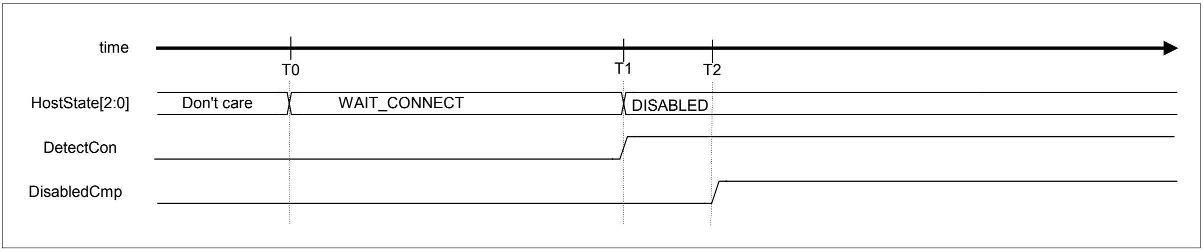


Fig. 6.63 GoWAIT_CONNECTtoDIS timing (HS mode)

Table 6.55 GoWAIT_CONNECTtoDIS Timing Values (HS Mode)

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoWAIT_CONNECTtoDIS (by firmware). Process equivalent to GoWAIT_CONNECT is executed (by hardware).	0 (reference)
T1	A connection detection is performed and a connection detected status is issued. Process equivalent to GoDISABLED is executed (by hardware).	T1
T2	A disabled complete status is issued (by hardware).	T2

6.3.9.3.9 GoWAIT_CONNECTtoOP

When the host state transition execution (H_NegoControl_0.AutoMode) is set to GoWAIT_CONNECTtoOP, the processes required for a transition from WAIT_CONNECT state to OPERATIONAL state are automatically executed by the LSI's hardware.

6.3.9.3.9.1 When HS Device is Connected

The procedure that is executed when an HS device is connected is described below. Processes (2) to (9) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoWAIT_CONNECTtoOP (T0).

(2) A process equivalent to GoWAIT_CONNECT is executed (T0).

(3) A connection detection is performed and a connection detected status (H_SIE_IntStat_0.DetectCon) is issued (T1).

(4) Process equivalent to GoDISABLED is executed (T1).

(5) A disabled complete status (H_SIE_IntStat_1.DisabledCmp) is issued (T2).

(6) Process equivalent to GoRESET is executed (T2).

(7) A device Chirp is detected, and a device chirp normal detection status (H_SIE_IntStat_0.DetectDevChirpOK) is issued (T3).

(8) A reset complete status (H_SIE_IntStat_1.ResetCmp) is issued (T4).

(9) Process equivalent to GoOPERATIONAL is executed (T4).

Note that the timing in each state here is the same as when GoWAIT_CONNECT, GoDISABLED, GoRESET, and GoOPERATIONAL are executed. For details about the timing, refer to the relevant sections on GoWAIT_CONNECT, GoDISABLED, GoRESET, and GoOPERATIONAL.

For the execution procedure and timing in cases when an error (disconnection, VBUS error, or device chirp error) is detected during the process, refer to the relevant sections on disconnection detection, VBUS error, and GoRESET.

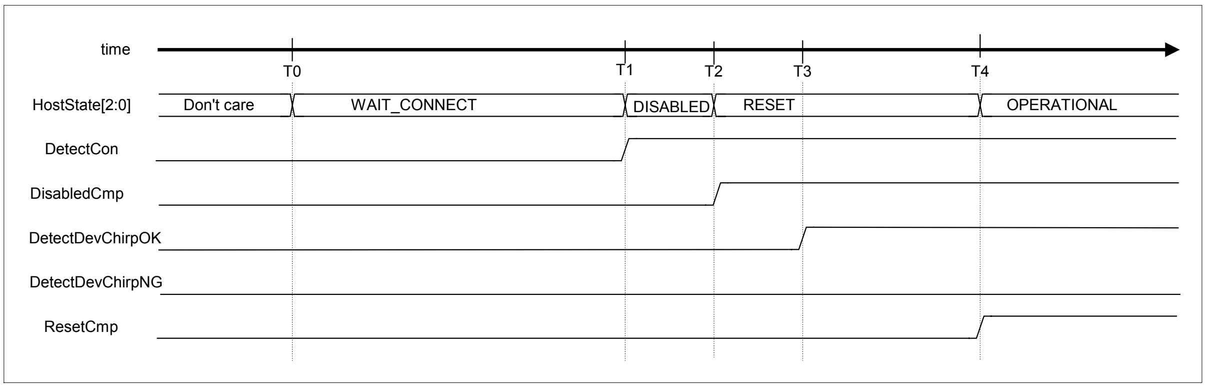


Fig. 6.64 GoWAIT_CONNECTtoOP timing (HS mode)

Table 6.56 GoWAIT_CONNECTtoOP Timing Values (HS Mode)

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoWAIT_CONNECTtoOP (by firmware). A process equivalent to GoWAIT_CONNECT is executed (by hardware).	0 (reference)
T1	A connection detection is performed and a connection detected status is issued. A process equivalent to GoDISABLED is executed (by hardware).	T1
T2	A disabled complete status is issued. A process equivalent to GoRESET is executed (by hardware).	T2
T3	A device Chirp is detected, and a device chirp normal detection status is issued (by hardware).	T3
T4	A reset complete status (H_SIE_IntStat_1.ResetCmp) is issued. A process equivalent to GoOPERATIONAL is executed (by hardware).	T4

6.3.9.3.9.2 When FS or LS Device is Connected

The procedure that is executed when an FS or LS device is connected is described below. Processes (2) to (9) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoWAIT_CONNECTtoOP (T0).

(2) A process equivalent to GoWAIT_CONNECT is executed (T0).

(3) A connection detection is performed and a connection detected status (H_SIE_IntStat_0.DetectCon) is issued (T1).

(4) A process equivalent to GoDISABLED is executed (T1).

(5) A disabled complete status (H_SIE_IntStat_1.DisabledCmp) is issued (T2).

(6) A process equivalent to GoRESET is executed (T1).

(7) Because a device Chirp is not detected, a device chirp normal/abnormal detection (H_SIE_IntStat_0.DetectDevChirpOK/NG) is not issued (T3).

(8) A reset complete status (H_SIE_IntStat_1.ResetCmp) is issued (T4).

(9) A process equivalent to GoOPERATIONAL is executed (T4).

Note that the timing in each state here is the same as when GoWAIT_CONNECT, GoDISABLED, GoRESET, and GoOPERATIONAL are executed. For details about the timing, refer to the relevant sections on GoWAIT_CONNECT, GoDISABLED, GoRESET, and GoOPERATIONAL.

For the execution procedure and timing in cases when an error (disconnection or VBUS error) is detected during the process, refer to the relevant sections on disconnection detection and VBUS error.

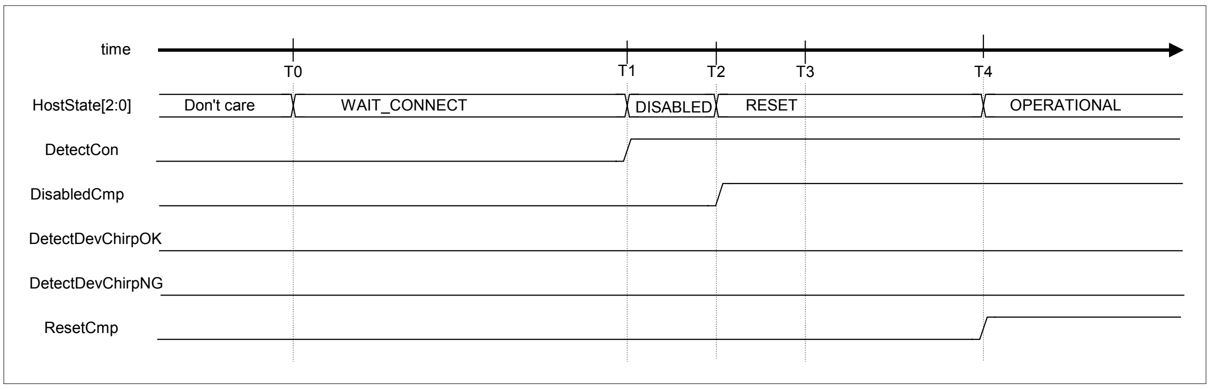


Fig. 6.65 GoWAIT_CONNECTtoOP timing (FS or LS mode)

Table 6.57 GoWAIT_CONNECTtoOP Timing Values (FS or LS Mode)

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoWAIT_CONNECTtoOP (by firmware). A process equivalent to GoWAIT_CONNECT is executed (by hardware).	0 (reference)
T1	A connection detection is performed and a connection detected status is issued. A process equivalent to GoDISABLED is executed (by hardware).	T1
T2	A disabled complete status is issued. A process equivalent to GoRESET is executed (by hardware).	T2
T3	Because a device Chirp is not detected, a device chirp normal/abnormal detection is not issued (by hardware).	T3
T4	A reset complete status (H_SIE_IntStat_1.ResetCmp) is issued. A process equivalent to GoOPERATIONAL is executed (by hardware).	T4

6.3.9.3.10 GoRESETtoOP

When the host state transition execution (H_NegoControl_0.AutoMode) is set to GoRESETtoOP, the processes required for a transition from RESET state to OPERATIONAL state are automatically executed by the LSI's hardware.

6.3.9.3.10.1 When HS Device is Connected

The procedure that is executed when an HS device is connected is described below. Processes (2) to (5) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoRESETtoOP (T0).

(2) A process equivalent to GoRESET is executed (T0).

(3) A device Chirp is detected, and a device chirp normal detection status (H_SIE_IntStat_0.DetectDevChirpOK) is issued (T1).

(4) A reset complete status (H_SIE_IntStat_1.ResetCmp) is issued (T2).

(5) A process equivalent to GoOPERATIONAL is executed (T2).

Note that the timing in each state here is the same as when GoRESET and GoOPERATIONAL are executed. For details about the timing, refer to the relevant sections on GoRESET and GoOPERATIONAL.

For the execution procedure and timing in cases when an error (VBUS error or device chirp error) is detected during the process, refer to the relevant sections on VBUS error and GoRESET.

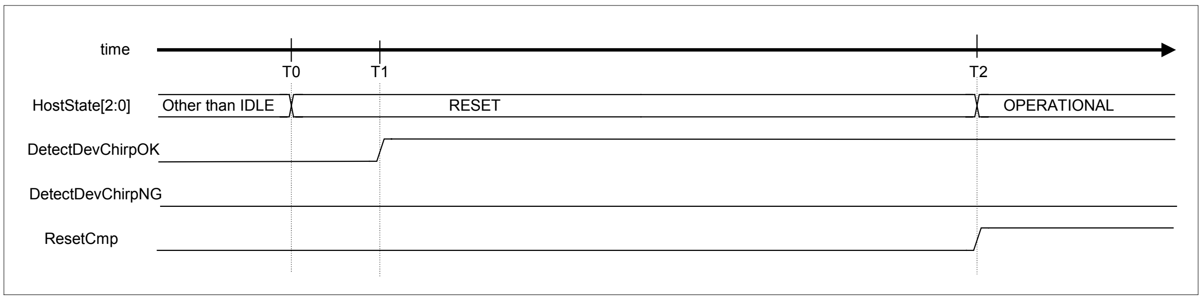


Fig. 6.66 GoRESETtoOP timing (HS mode)

Table 6.58 GoRESETtoOP Timing Values (HS Mode)

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoRESETtoOP (by firmware). A process equivalent to GoRESET is executed (by hardware).	0 (reference)
T1	A device Chirp is detected, and a device chirp normal detection status is issued (by hardware).	T1
T2	A reset complete status is issued. A process equivalent to GoOPERATIONAL is executed (by hardware).	T2

6.3.9.3.10.2 When FS or LS Device is Connected

The procedure that is executed when an FS or LS device is connected is described below. Processes (2) to (5) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoRESETtoOP (T0).

(2) A process equivalent to GoRESET is executed (T0).

(3) Because a device Chirp is not detected, a device chirp normal/abnormal detection (H_SIE_IntStat_0.DetectDevChirpOK) is not issued (T1).

(4) A reset complete status (H_SIE_IntStat_1.ResetCmp) is issued (T2).

(5) A process equivalent to GoOPERATIONAL is executed (T2).

Note that the timing in each state here is the same as when GoRESET and GoOPERATIONAL are executed. For details about the timing, refer to the relevant sections on GoRESET and GoOPERATIONAL.

For the execution procedure and timing in cases when an error (VBUS error) is detected during the process, refer to the relevant section on VBUS error.

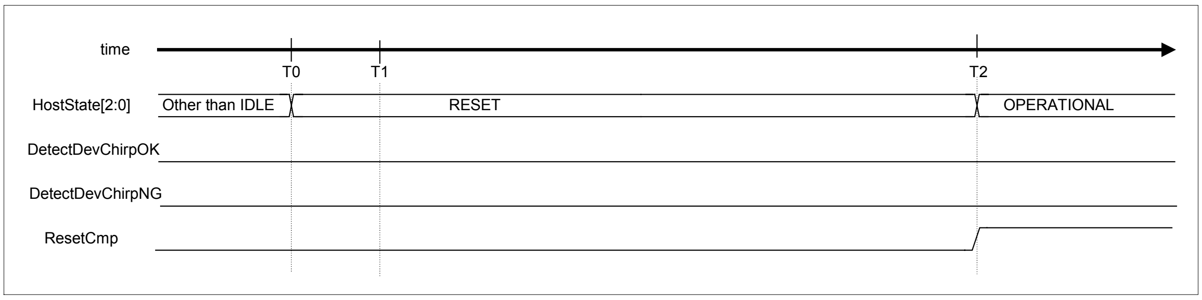


Fig. 6.67 GoRESETtoOP timing (FS or LS mode)

Table 6.59 GoRESETtoOP Timing Values (FS or LS Mode)

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoRESETtoOP (by firmware). A process equivalent to GoRESET is executed (by hardware).	0 (reference)
T1	Because a device Chirp is not detected, a device chirp normal/abnormal detection is not issued (by hardware).	T1
T2	A reset complete status is issued. A process equivalent to GoOPERATIONAL is executed (by hardware).	T2

6.3.9.3.11 GoSUSPENDtoOP

When the host state transition execution (H_NegoControl_0.AutoMode) is set to GoSUSPENDtoOP, the processes required for a transition from SUSPEND state to OPERATIONAL state is automatically executed by the LSI's hardware.

When GoSUSPENDtoOP is set, the remote wakeup detection function is automatically turned on/off. Note, however, that since this on/off is not reflected in the remote wakeup acceptance enable (H_NegoControl_1.RmtWkupDetEnb), the firmware is not required to manipulate the remote wakeup acceptance enable (H_NegoControl_1.RmtWkupDetEnb).

When this setting is used, do not use the power management function.

The procedure that is executed for this setting is described below. Processes (2) to (7) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoSUSPENDtoOP (T0).

(2) A process equivalent to GoSUSPEND is executed (T0).

(3) A transition-to-suspend complete status (H_SIE_IntStat_1.SuspendCmp) is issued (T1).

(4) A remote wakeup is detected, and a remote wakeup detected status (H_SIE_IntStat_0.DetectRmtWkup) is issued (T2).

(5) A process equivalent to GoRESUME is executed (T2).

(6) A resume complete status (H_SIE_IntStat_1.ResumeCmp) is issued (T3).

(7) A process equivalent to GoOPERATIONAL is executed (T3).

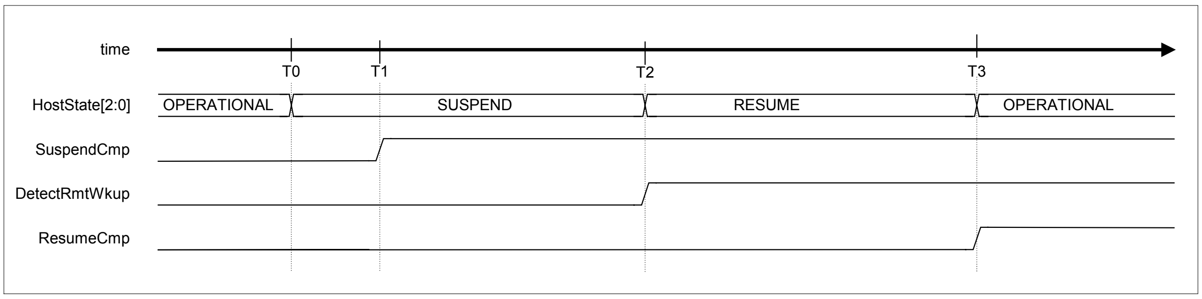


Fig. 6.68 GoSUSPENDtoOP timing

Table 6.60 GoSUSPENDtoOP Timing Values

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoSUSPENDtoOP (by firmware). A process equivalent to GoSUSPEND is executed (by hardware).	0 (reference)
T1	A transition-to-suspend complete status is issued (by hardware).	T1
T2	A remote wakeup is detected, and a remote wakeup detected status is issued. A process equivalent to GoRESUME is executed (by hardware).	T2
T3	A resume complete status is issued. A process equivalent to GoOPERATIONAL is executed (by hardware).	T3

6.3.9.3.12 GoRESUMEtoOP

When the host state transition execution (H_NegoControl_0.AutoMode) is set to GoRESUMEtoOP, the processes required for a transition from RESUME state to OPERATIONAL state is automatically executed by the LSI's hardware.

The procedure that is executed for this setting is described below. Processes (2) to (4) below are automatically executed by the LSI's hardware.

(1) The firmware sets the host state transition execution (H_NegoControl_0.AutoMode) to GoRESUMEtoOP (T0).

(2) A process equivalent to GoRESUME is executed (T0).

(3) A resume complete status (H_SIE_IntStat_1.ResumeCmp) is issued (T1).

(4) A process equivalent to GoOPERATIONAL is executed (T1).

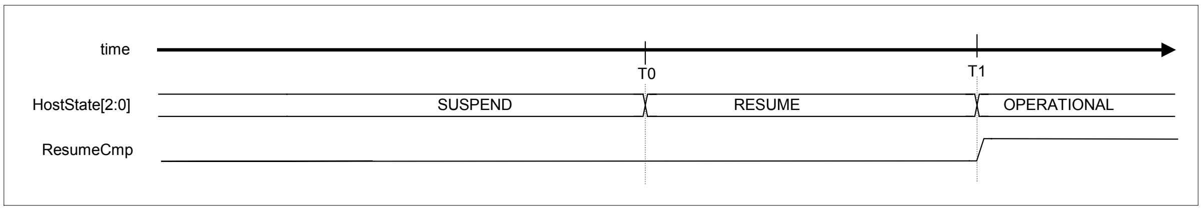


Fig. 6.69 GoRESUMEtoOP timing

Table 6.61 GoRESUMEtoOP Timing Values

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to GoRESUMEtoOP (by firmware). A process equivalent to GoRESUME is executed (by hardware).	0 (reference)
T1	A resume complete status is issued. A process equivalent to GoOPERATIONAL is executed (by hardware).	T1

6.4 Media Transfer Function

6.4.1 Media Data

The types of data handled by equipment vary widely with the application concerned. For example, these include music data and video data, which collectively in this specification are referred to as "media data."

6.4.2 Transfer of Media Data

The LSI stipulated herein has a 64-byte FIFO (media FIFO) used to transfer media data, separately from the FIFOs used for USB transfers. Use of this FIFO helps to realize an easy transfer of media data between a HDD and memory.

The function of the LSI to perform transfers between a HDD and memory via the media FIFO is referred to in this specification as the "media data transfer function."

The diagram below shows how a media data transfer is performed during playback (HDD to memory).

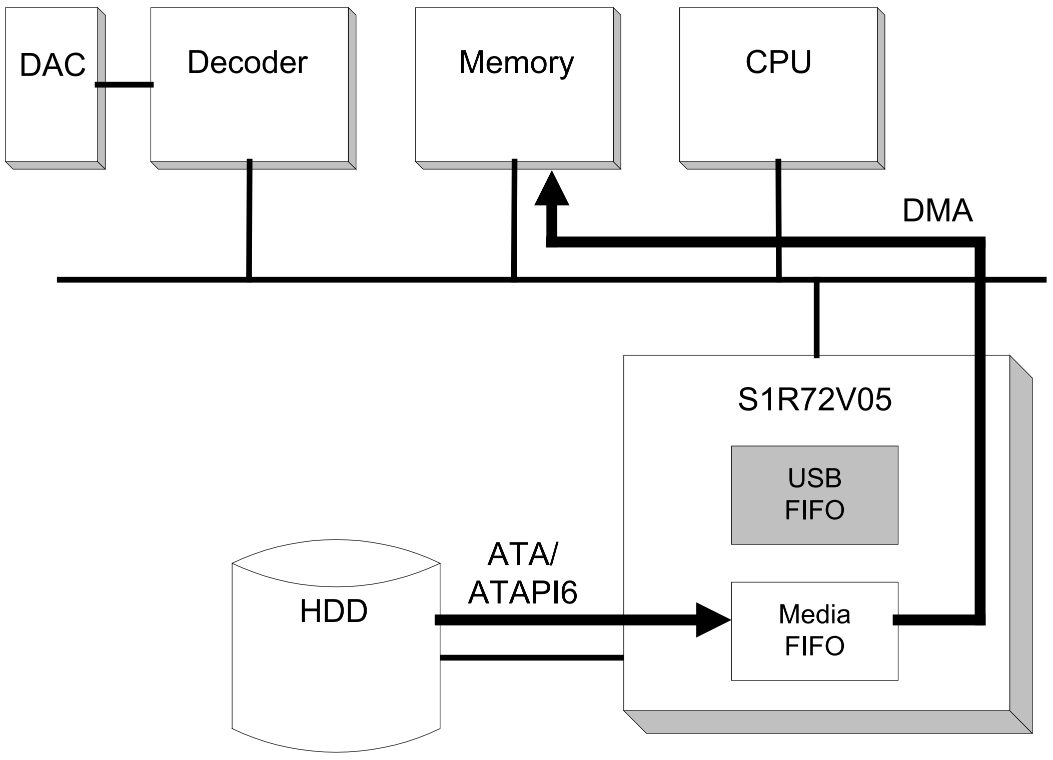


Fig. 6.70 Example of a media data transfer during playback

Access to the media FIFO is established by specifying the IDE port and another port (DMA0/1, CPU_Rd, or CPU_Wr) using the Media FIFO_Join register.

For details on how to access the media FIFO, refer to Section 6.6.3, "Media FIFO Management."

6.4.3 Reducing Power Consumption

The media data transfer function can be used in one of the power mode states of the LSI: ACTIVE60, ACT_DEVICE, or ACT_HOST. Of these, the ACTIVE60 state allows a significant reduction in the amount of power consumed in the chip, because the USB device and the USB host functions are turned off when in this state.

For details about the power management function, refer to Section 6.5, “Power Management Function.”

The diagrams below show the data flow for the case where USB transfer and media data transfer are performed, and for the case where only a media data transfer is performed.

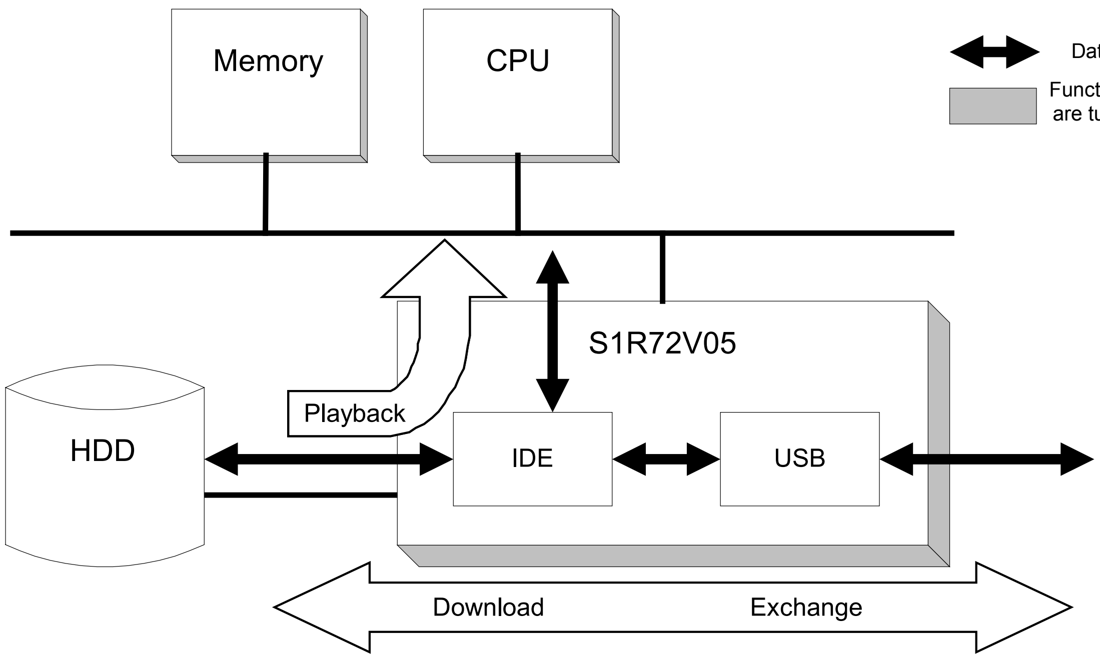


Fig. 6.71 USB transfer and media data transfer (example during playback)

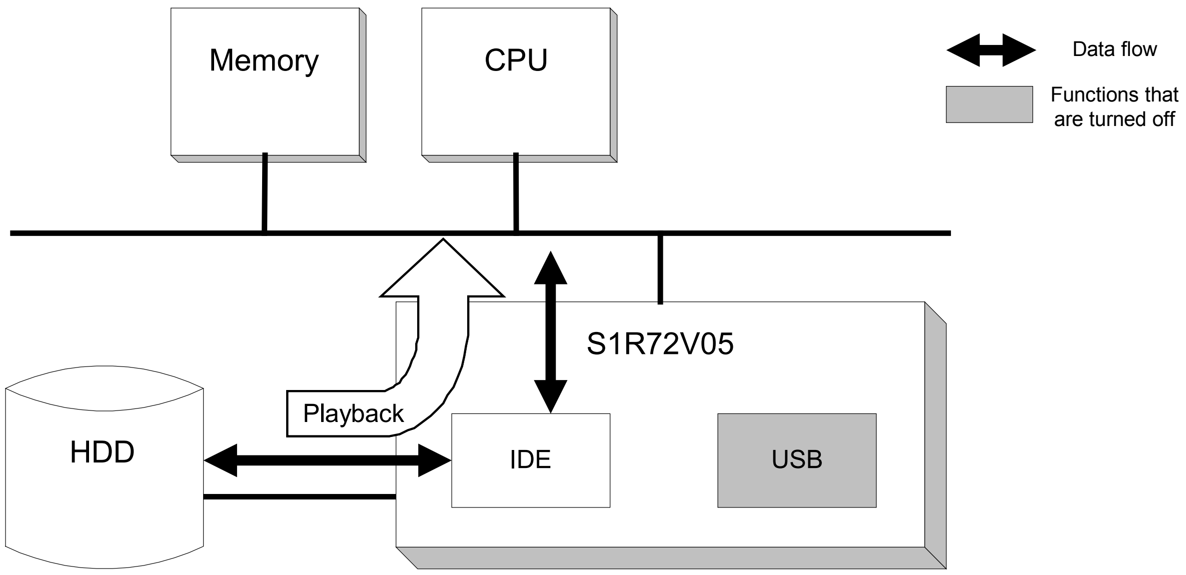
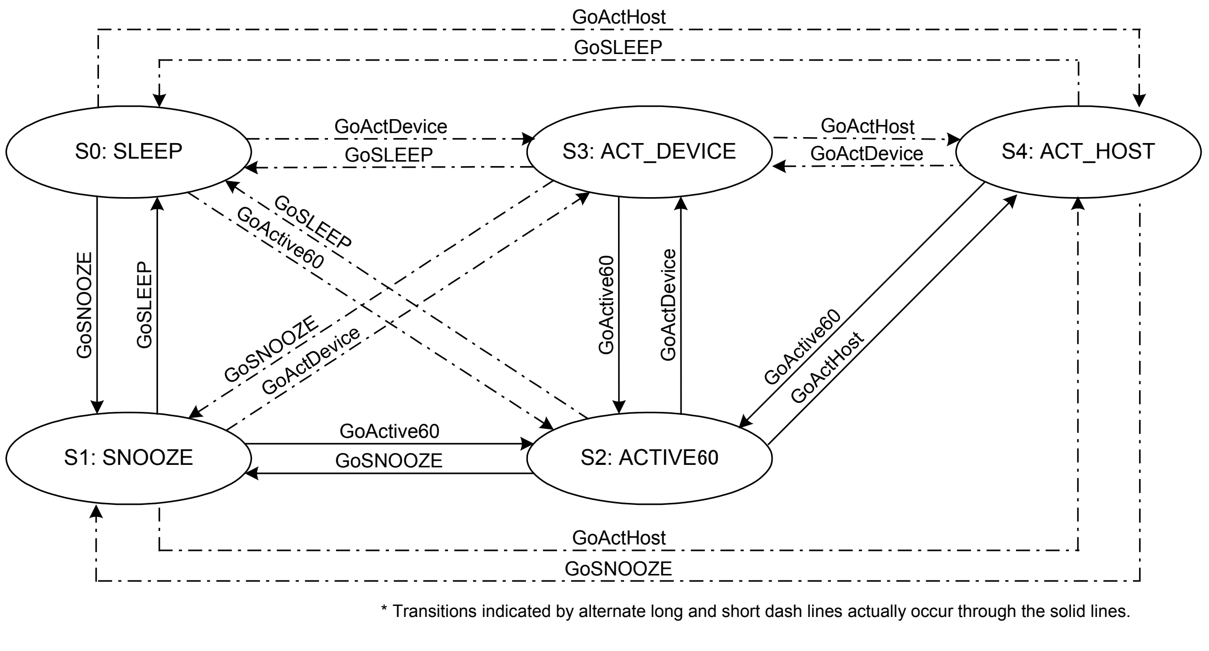


Fig. 6.72 Media data transfer only (example during playback)

6.5 Power Management Function

The power management function controls the operation of the oscillator and the PLLs (DevicePLL480, HostPLL480, and PLL60), manipulating a transition between five states: Sleep, Snooze, Active60, ActDevice, and ActHost. To shift to other states, set the PM_Control_0.GoSLEEP, PM_Control_0.GoSNOOZE, PM_Control_0.GoActive60, PM_Control_0.GoActDevice, or PM_Control_0.GoActHost bit, and a state transition will start. This state transition finishes after given processes are performed. To confirm the LSI's current state, check the PM_Control_1.PM_State [3:0]. A MainIntStat.FinishedPM event is generated after a transition is complete. At this time, if the MainIntEnb.EnFinishedPM and MainIntEnb.EnSIE_IntStat bits have both been set, an XINT interrupt is generated.

A state can be entered from all other states, so that if the PM_Control.GoSLEEP bit is set during the ActDevice or ActHost state, the LSI shifts to the Sleep state via the Active60 and Snooze states. When the state transition is fully complete, a SIE_IntStat.FinishedPM event is generated. Also, if the PM_Control.GoActDevice or PM_Control.GoActHost bit is set during the Sleep state, the LSI shifts to the ActDevice or ActHost state via the Snooze and Active60 states. When the state transition is fully complete, a MainIntStat.FinishedPM event is generated. Similarly, if the PM_Control_0.GoSLEEP bit is set during the Active60 state, the LSI shifts to the Sleep state via the Snooze state. When the state transition is fully complete, a MainIntStat.FinishedPM event is generated. Furthermore, if the PM_Control_0.GoActHost bit is set during the ActDevice state, the LSI shifts to the ActHost state via the Active60 state. When the state transition is fully complete, a MainIntStat.FinishedPM event is generated. Also similarly, if the PM_Control_0.GoActDevice bit is set during the ActHost state, the LSI shifts to the ActDevice state via the Active60 state. When the state transition is fully complete, a MainIntStat.FinishedPM event is generated.



* Transitions indicated by alternate long and short dash lines actually occur through the solid lines.

Fig. 6.73 Power management

6.5.1 SLEEP (Sleep)

This is the state when the oscillator is not oscillating. In this state, therefore, the PLLs are not oscillating either.

When the LSI shifts to the Sleep sate by setting the PM_Control_0.GoSLEEP bit during the Snooze, Active60, ActDevice, or ActHost state, the operating PLLs and OSC are turned off in order of DevicePLL480 or HostPLL480, and PLL60, and finally the OSCCLK output is turned off before the clock oscillation stops.

Conversely, if the LSI shifts to the Snooze state after leaving Sleep by setting the PM_Control_0.GoSNOOZE, PM_Control_0.GoActive60, PM_Control.GoActDevice, and PM_Control_0.GoActHost bits during the Sleep state, OSCCLK is gated for an oscillation stabilization time to ensure that it will not be supplied to the internal circuit until after the oscillator‘s oscillation is stablized. Since the oscillation stabilization time varies with the oscillator cell, resonator, peripheral circuits, and the board involved, use the WakeUpTim_H, L registers to set it.

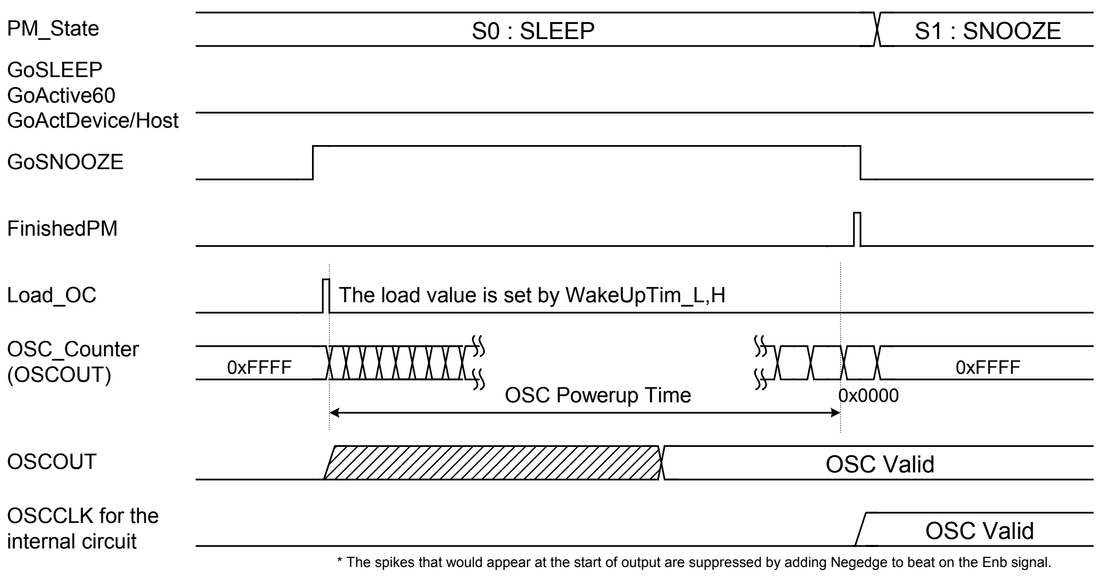


Fig. 6.74 Leaving SLEEP state (during GoSNOOZE)

6.5.2 SNOOZE (Snooze)

This is the state in which the oscillator is oscillating but the PLLs are not.

When the LSI shifts to the Snooze state by setting the PM_Control_0.GoSNOOZE bit during the Active60, ActDevice, or ActHost state, clock being output are halted and then the PLLs are turned off in order of DevicePLL480 or HostPLL480 ,and PLL60.

However, if the LSI shifts to the Active state after leaving Snooze by setting the PM_Control_0.GoActDevice, PM_Control.GoActHost, and PM_Control_0.GoActive60 bits during the Snooze state, SCLK is gated for a PLL stabilization time (approx. 250 us) to ensure that it will not be supplied to the internal circuit until after the PLL's oscillation is stablized.

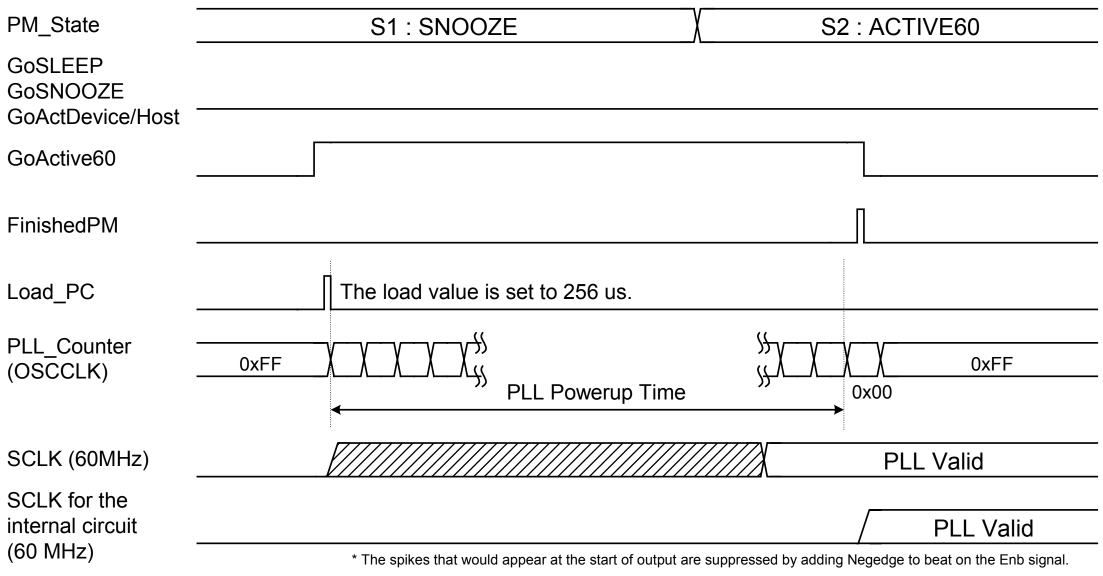

PM_State
S1 : SNOOZE
S2 : ACTIVE60
GoSLEEP
GoSNOOZE
GoActDevice/Host
GoActive60
FinishedPM
Load_PC
The load value is set to 256 us.
PLL_Counter (OSCCLK)
0xFF
0xFF
0x00
PLL Powerup Time
SCLK (60MHz)
PLL Valid
SCLK for the internal circuit (60 MHz)
PLL Valid
* The spikes that would appear at the start of output are suppressed by adding Negedge to beat on the Enb signal.

Fig. 6.75 Leaving SNOOZE state (during GoActive60)

6.5.3 ACTIVE60 (Active60)

In this state, the oscillator and PLL60 are active and the DevicePLL480 and HostPLL480 are inactive. The registers and bits shown in bold-face italic in the register map can be read from and written to in Snooze and Sleep states. The other registers can be read or written to in the Active60 state, subject to limitations. The registers in the device/host common register map (7.1) can both be read from and written to.(Note) Those in the device register map (7.2) and host register map (7.3) can only be read. Note that since the USB circuit requires an SCLK480, USB device functionality is available only in the ActDevice state. The USB host functionality works only in the ActHost state. The IDE and CPU circuits do not require SCLK480 and can therefore operate in any of the Active60, ActDevice, and ActHost states. Since the 480 MHz PLL consumes more current than the 60 MHz PLL, you can shift to the Ative60 state to reduce chip power consumption if USB is not needed.

Note: The registers at addresses 0x40 to 0x47 and 0x50 to 0x6F cannot be read from or written to in the Active60 state.

6.5.4 ACT_DEVICE (ActDevice)

This is the state in which the oscillator, PLL60, and DevicePLL480 are operating. The registers and bits shown in ***bold face italic*** in the register map can be read or written to in Snooze or Sleep state. All other registers cannot be read or written to unless in the Active60, ActDevice, or ActHost state. When these registers are read in states other than these states, the read values will be 0.

Note also that since the USB device circuit requires SCLK480, it operates only in the ActDevice state. The IDE and CPU circuits do not require SCLK480, allowing their operation in any of the Active60, ActDevice, and ActHost states. Since the 480 MHz PLL consumes more current than the 60 MHz PLL, if the USB is unnecessary, the amount of power consumed in the chip can be reduced by shifting to the Active60 state.

6.5.5 ACT_HOST (ActHost)

This is the state in which the oscillator, PLL60, and HostPLL480 are operating. The registers and bits shown in ***bold face italic*** in the register map can be read or written to in Snooze or Sleep states. All other registers cannot be read or written to unless in the Active60, ActDevice, or ActHost state. When these registers are read in states other than these states, the read values will be 0.

Note also that since the USB host circuit requires SCLK480, it operates only in the ActHost state. The IDE and CPU circuits do not require SCLK480, allowing their operation in any of the Active60, ActDevice, and ActHost states. Since the 480 MHz PLL consumes more current than the 60 MHz PLL, if the USB is unnecessary, the amount of power consumed in the chip can be reduced by shifting to the Active60 state.

6.6 FIFO Management

6.6.1 Device FIFO Management

This section describes FIFO management.

6.6.1.1 FIFO Memory Map

The following shows the FIFO memory map.

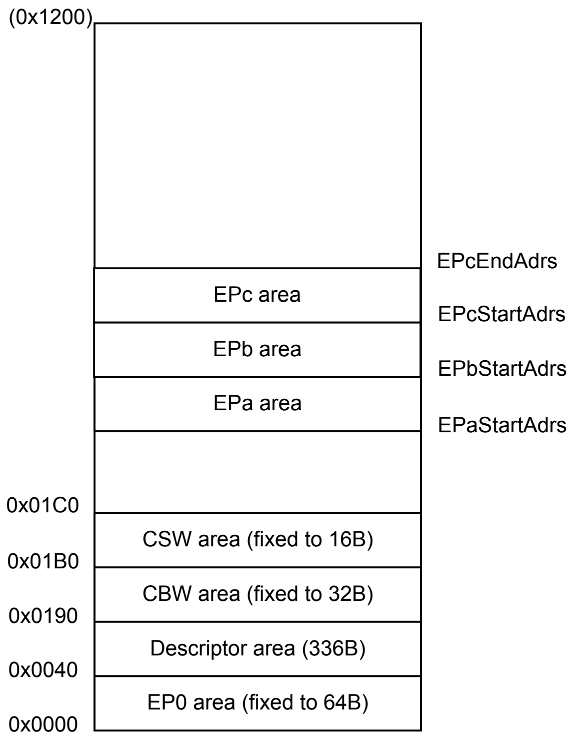


Fig. 6.76 Device FIFO memory map

The FIFO memory can be divided for use into a maximum of seven areas—the EP0 area, descriptor area, CBW area, CSW area, EPa area, EPb area, and EPc area. Of these areas, EP0, descriptor, CBW, and CSW areas have a fixed amount of storage allocated to each, as shown in Fig. 6.76. The other EPx{x=a-c} areas can have any amount of storage allocated by setting the FIFO area setup registers (D_EPx{x=a-c}StartAdrs_H,L and D_EPcEndAdrs_H,L) as desired.

The EP0 area is provided for endpoint 0 that is indispensable for the USB, and is used for operation in both IN and OUT directions. This area has 64 bytes of storage reserved for it, of which a finite amount of storage equal to the max packet size of endpoint 0 beginning with the address 0x000 can be used. Therefore, endpoint 0 is always a single buffer.

The descriptor area is provided for use by the descriptor reply function. This area has 336 bytes of storage reserved for it, which can be used from any address in it. The actual method for using this area is described later in Section 6.6.1.2. Actually, any FIFO area can be set for use by the descriptor reply function, but to avoid conflict we recommend that the area shown above be used as the descriptor area.

The CBW area is used for CBW support of the bulk-only support function. This area has 32 bytes of storage reserved for it, of which 31 bytes of storage beginning with the address 0x190 is used. The actual method for using this area is described later in Section 6.6.1.3.

The CSW area is used for CSW support of the bulk-only support function. This area has 16 bytes of storage reserved for it, of which 13 bytes of storage beginning with the address 0x1B0 is used. The actual method for using this area is described later in Section 6.6.1.4.

The EPa, EPb, and EPc areas are the general-purpose endpoint areas whose endpoint numbers and IN or OUT directions can be set anyway as desired. The EPa area can be used for bulk transfers in FS mode or interrupt transfers in HS/FS mode. The EPb and EPc areas can be used for bulk transfers in HS mode, in addition to the above.

The EP0, EPa, EPb, and EPc areas are controlled as FIFO, so that the number of data bytes stored in each is retained. To clear this retained status, set the D_EPnControl.AllFIFO_Clr or D_EPnControl.EP0FIFO_Clr bit or the D_EPrFIFO_Clr.EPx{x=a-c}FIFO_Clr bits that are provided for each area.

This status clearing operation only initializes the data retention information, and does not write or clear data to or from the area. Therefore, in no case will the data in RAM be cleared by these bits, so that the information recorded in the descriptor area will never be lost and there is no need to write data back again after clearing the status.

6.6.1.2 Method for Using the Descriptor Area

The descriptor area is provided for use by the descriptor reply function. The descriptor reply function can be used when the data stage is executed in IN transfer at endpoint 0.

To execute a data stage in the IN direction, set the start address of the data written into this area and the data size to be returned and then execute the descriptor reply function. The data stage will be automatically executed.

This area may be used to write the content of uniquely determined equipment data as for a device descriptor. Once such data is written into this area during initialization after power-on, for example, it is possible to instruct that the data in this area be returned when an request is accepted. That way, requests can be responded promptly because there is no need to write data into the EP0 area for each request.

6.6.1.2.1 Writing Data into the Descriptor Area

To write data into the descriptor area, use the RAM_WrDoor function. Set the write start address in the RAM_WrArs_H,L registers and then write data in the RAM_WrDoor_0,1 registers. The RAM_WrArs_H,L register values are updated every write data bytes each time the registers are written to, so that if data needs to be written to contiguous addresses, the data can be written to the RAM_WrDoor_0,1 registers successively.

Note that the RAM_WrDoor_0,1 registers are write-only registers.

6.6.1.2.2 Executing a Data Stage (IN) in the Descriptor Area

To use the written data in the descriptor reply function, set the start address of the data to be transmitted to the data stage in the DescAdrs_H,L registers and the data size to be returned in the DescSize_H,L registers, and then set the D_EP0Control.ReplyDescriptor bit to 1. Furthermore, set the D_EP0Control.INxOUT bit to 1 to permit an IN transaction to be performed. To ensure that data packets will be sent back to an IN transaction of the data stage, make sure the D_SETUP_Control.ProtectEP0 is cleared before the D_EP0Control.IN.ForceNAK bit is cleared.

After settings are made, data packets are sent back to the host in response to an IN transaction from the host while being automatically divided into the max packet size (set by D_EP0MaxSize) until the number of data bytes set in the D_DescSize_H,L registers is reached. If the D_DescSize_H,L register value is less than the max packet size or the remaining data bytes after being divided are less than the max packet size, the data is automatically transmitted as a short packet.

When an OUT transaction is issued from the host, the D_EP0Control.ReplyDescriptor bit is cleared and the D_FIFO_IntStat.DescriptorCmp is set. The firmware should go to processing of the status stage.

6.6.1.3 Method for Using the CBW Area

The CBW area is used for CBW support of the bulk-only support function. When a command transport of Bulk Only Transport Protocol is to be performed at the Bulk Only endpoint (endpoint EPb or EPc), data can be received in this area. That way, control of IDE or DMA transfers can be facilitated by allowing only data reception in the endpoint FIFO.

6.6.1.3.1 Receiving in the CBW Area

If when CBW support is being executed, an OUT transaction is performed at the target endpoint and the data size is 31 bytes, the data is received in the CBW area. If the data is more than 31 bytes in size, an error status is issued and the data is discarded.

6.6.1.3.2 Reading Data from the CBW Area

To read the data received in the CBW area, use the RAM_Rd function. When the RAM_RdControl.RAM_GoRdCBW_CSW bit is set, data is read from the CBW area and copied to the RAM_Rd_00 through RAM_Rd_1E registers, at end of which a completion status (CPU_IntStat.RAM_RdCmp bit) is issued.

6.6.1.4 Method for Using the CSW Area

The CSW area is used for CSW support of the bulk-only support function. When a command transport of Bulk Only Transport Protocol is to be performed at the Bulk IN endpoint (endpoint EPb or EPc), data can be transmitted from this area. That way, control of IDE or DMA transfers can be facilitated by allowing only data reception in the endpoint FIFO.

6.6.1.4.1 Transmitting from the CSW Area

If when CSW support is being executed, an IN transaction is performed at the target endpoint, 13 bytes of data is transmitted from the CSW area as data packet.

6.6.1.4.2 Writing Data into the CSW Area

To write data into the CSW area, use the RAM_WrDoor function, write the start address of the CSW area (0x1B0) in the RAM_WrAdr_H,L registers and write 13 bytes of valid data via the RAM_WrDoor_0,1 registers. Since the CSW area has a 16 bytes of storage reserved for it, there is no possibility of affecting other areas even when 14 bytes of data are written into this area wordwise.

6.6.1.5 Method for Accessing the FIFO

There are several methods to access the FIFO, including the CPU (registers), CPU (DMA), IDE, and USB.

6.6.1.5.1 Method for Accessing the FIFO (RAM_Rd)

To access the FIFO for read via the RAM_Rd register of the CPUIF, set the start address of the FIFO area from which to read and the data size in the RAM_RdAdr_H,L registers and RAM_RdCount register, respectively, and then set the RAM_RdControl.RAM_GoRd bit. When the data of the specified FIFO area is ready for read from the RAM_Rd register, the CPU_IntStat.RAM_RdCmp bit is set to 1. After confirming the RAM_RdCmp bit, read data from the RAM_Rd registers. The data in the RAM_Rd registers are stored in order of register numbers beginning with RAM_Rd_00. If the data size set in the RAM_RdCount register is smaller than 32 bytes, the data bytes in the RAM_Rd registers exceeding the set size are ignored.

FIFO data read via the RAM_Rd registers can be performed regardless of the settings for FIFO area of a relevant channel.

The RAM_Adrs_H,L and RAM_Count register values are updated one by one while the RAM_Rd function is in operation. Once the RAM_Rd function is activated, do not access these registers until the CPU_IntStat.RAM_RdCmp bit is set. If these registers are accessed for read while the RAM_Rd function is in operation, the read values cannot be guaranteed. Writing to these registers while in operation will cause the LSI to operate erratically.

6.6.1.5.2 Method for Accessing the FIFO (RAM_WrDoor)

To access the FIFO for write via the RAM_WrDoor_0,1 registers of the CPUIF, set the write start address in the RAM_WrAdr_H,L registers and write data via the RAM_WrDoor_0,1 registers. The RAM_WrAdr_0,1 registers are automatically incremented by an amount equal to written bytes each time the FIFO is accessed for write, so that if data needs to be written to contiguous addresses, the data can be written to the RAM_WrDoor_0,1 registers successively.

Write to the FIFO via the RAM_WrDoor_0,1 registers can be performed regardless of the settings for FIFO area of a relevant channel.

6.6.1.5.3 Method for Accessing the FIFO (Register Access)

To access the FIFO for read via a register access of the CPU, set D_EPx{x=0,a-c}Join.JoinCPU_Rd for any one of the channels to 1 and read data via the FIFO_Rd_0,1 or FIFO_ByteRd register.

To access the FIFO for write via a register access of the CPU, set D_EPx{x=0,a-c}Join.JoinCPU_Wr for any one of the channels to 1 and write data in the FIFO_Wr_0,1 registers.

The FIFO_RdRemain_H,L registers indicate the remaining number of data bytes that can be read from the FIFO for only one channel that is set by D_EPx{x=0,a-c}Join.JoinCPU_Rd. Similarly, the FIFO_WrRemain_H,L registers indicate the remaining number of bytes in the FIFO area to which data can be written for only one channel that is set by D_EPx{x=0,a-c}Join.JoinCPU_Wr.

Be aware that in cases where registers are to be dumped when debugging the firmware using an ICE or other tool, if any D_EPx{x=0,a-c}Join.JoinCPU_Rd register is set, data may inadvertently be read from the FIFO when registers are dumped.

6.6.1.5.4 Method for Accessing the FIFO (DMA)

To access the FIFO for read via a DMA access of the CPU, select only one endpoint on either DMA channel using the D_EPx{x=0,a-c}Join.JoinDMAx{x=0,1} bit and set the DMAx{x=0,1}_Control.Dir bit to 1, and then execute a DMA procedure to read data.

To access the FIFO for write via a DMA access of the CPU, select only one endpoint on either DMA channel using the D_EPx{x=0,a-c}Join.JoinDMAx{x=0,1} bit and set the DMAx{x=0,1}_Control.Dir bit to 0, and then execute a DMA procedure to write data.

The DMAx{x=0,1}_Remain_H,L registers indicate the remaining number of data bytes that can be read from the FIFO for only one endpoint on either DMA channel that is selected by the D_EPx{x=0,a-c}Join.JoinDMAx{x=0,1} bit. Similarly, they indicate the remaining number of bytes in the FIFO area to which data can be written for only one endpoint on either DMA channel that is selected by the D_EPx{x=0,a-c}Join.JoinDMAx{x=0,1} bit.

6.6.1.5.5 Method for Accessing the FIFO (IDE)

For the FIFO to be accessed by the IDE, select only one endpoint on any one of channels by using the D_EPx{x=0,a-c}Join.JoinIDE bit and execute an IDE procedure to transfer data. The direction of IDE transfer is determined by the IDE_Control.Dir bit.

6.6.1.5.6 Limitations on FIFO Access

The FIFO of the LSI stipulated herein is designed in such a way that transmission/reception with the USB, register or DMA read/write from the CPU bus, and transmission/reception with the IDE are performed at the same time. Furthermore, read from the CPU bus is accomplished by look-ahead processing.

For the above reasons, the setup method (Join) for access to the FIFO on respective Endpoints is subject basically to the following exclusion rules:

- For one endpoint, only one of writing factors can be set.
- For one endpoint, only one of reading factors can be set.
- For one endpoint, only one of JoinCPU_Wr, JoinCPU_Rd, or JoinDMAx{x=0,1} can be set.
- JoinIDE, JoinCPU_Wr, JoinCPU_Rd, and JoinDMAx{x=0,1} can respectively be set for only one endpoint at the same time.
- JoinIDE, JoinCPU_Wr, JoinCPU_Rd and JoinDMAx{x=0,1} cannot be set for FIFO and MediaFIFO at the same time.
- Only one of JoinCPU_Wr or JoinCPU_Rd can be set for FIFO at any given time.

There is an exception for the rule that says only one writing/reading factors can be set to one endpoint. For example, although it is possible to write to an OUT-endpoint FIFO area after setting JoinCPU_Wr, there must be no OUT transactions being performed when the JoinCPU_Wr is set. Similarly, although it is possible to read from an IN-endpoint FIFO area after setting JoinCPU_Rd, there must be no IN transactions being performed when the JoinCPU_Rd is set. The situation where no transactions are being performed can be confirmed by the fact that the ActiveUSB bit is cleared, EnEndpoint for the endpoint concerned is cleared, or ForceNAK is set.

6.6.2 Host FIFO Management

This section describes FIFO management in the USB host function.

6.6.2.1 FIFO Memory Map

The following shows the FIFO memory map.

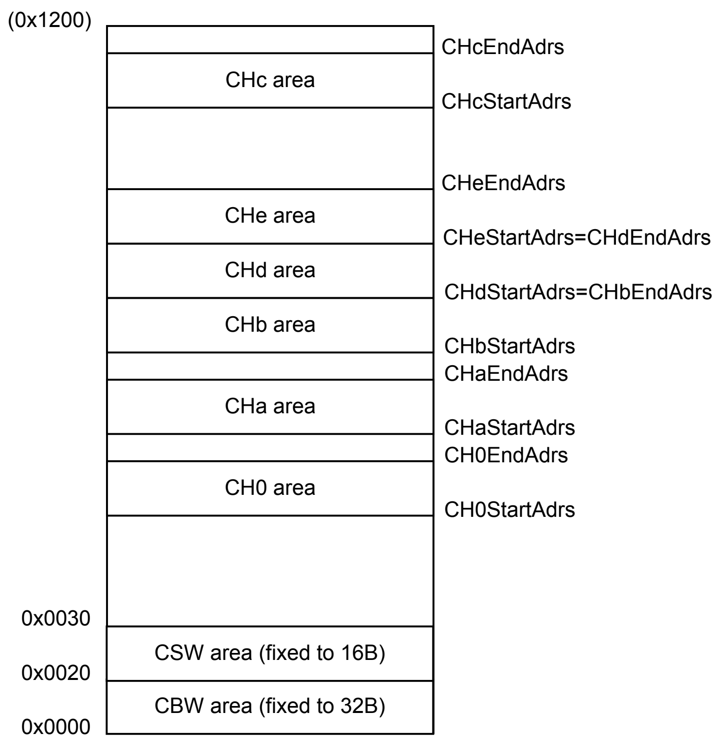


Fig. 6.77 Host FIFO memory map

The FIFO memory can be divided for use into a maximum of eight areas—CBW area, CSW area, CH0 area, CHa area, CHb area, CHc area, CHd area, and CHe area. Of these areas, the CBW area and CSW area have a fixed amount of storage allocated to each, as shown in Fig. 6.77. The other CHx{x=0,a-e} areas can have any amount of storage allocated by setting the FIFO area setup registers (H_CHx{x=0,a-e}StartAdrs_H,L and H_CHx{x=0,a-e}EndAdrs_H,L) as desired.

The CBW area is a 32-byte area beginning with 0x0000. The actual method for using this area is described later in Section 6.6.2.2.

The CSW area is a 16-byte area beginning with 0x0020. The actual method for using this area is described later in Section 6.6.2.3.

The CH0 area is used exclusively for control transfers.

The CHa area is used for bulk transfers.

The CHb, CHc, CHd, and CHe areas are used for bulk and interrupt transfers.

The CH0, CHa, CHb, CHc, CHd, and CHe areas are controlled as FIFO, so that the number of data bytes stored in each is retained. To clear this retained status, set the H_CHnControl.AllFIFO_Clr or

D_CHnControl.CH0FIFO_Clr bit or the H_CHrFIFO_Clr.CHx{x=a-c}FIFO_Clr bits that are provided for each area.

This status clearing operation only initializes the data retention information, and does not write or clear data to or from the area. Therefore, in no case will the data in RAM be cleared by these bits.

6.6.2.2 Method for Using the CBW Area

The CBW area is used for the bulk-only support function. When a command transport of Bulk Only Transport Protocol is performed on channel CHa, CBW data is transmitted from this area as data packet.

Prepare CBW data (31 bytes) in the CBW area beginning with 0x0000.

6.6.2.2.1 Transmitting from the CBW Area

When a command transport is performed on channel CHa while bulk-only support is active, CBW data is transmitted from this area as data packet.

6.6.2.2.2 Writing Data to the CBW Area

Use the RAM_WrDoor function to write data to the CBW area. Write the start address of the CBW area (0x0000) in the RAM_WrAdrs_H,L registers, and write 31 bytes of valid data via the RAM_WrDoor_0,1 registers. Since the CBW area has a 32 bytes of storage reserved for it, there is no possibility of affecting other areas even when 32 bytes of data are written into this area wordwise.

6.6.2.3 Method for Using the CSW Area

The CSW area is used for the bulk-only support function. When a status transport of Bulk Only Transport Protocol is performed on channel CHa, CSW data is received in this area.

6.6.2.3.1 Receiving in the CSW Area

When bulk-only support is active, a status transport is performed on channel CHa to receive CSW data.

6.6.2.3.2 Reading Data from the CSW Area

Use the RAM_Rd function to read the data that was received in the CSW area. When the RAM_RdControl.RAM_GoRdCBW_CSW bit is set, data (13 bytes) is read from the CSW area and copied to the RAM_Rd_00 through RAM_Rd_0C registers, at end of which a completion status (CPU_IntStat.RAM_RdCmp bit) is issued.

6.6.2.4 Method for Accessing the FIFO

There are several methods to access the FIFO, including the CPU (registers), CPU (DMA), IDE, and USB.

6.6.2.4.1 Method for Accessing the FIFO (RAM_Rd)

To access the FIFO for read via the RAM_Rd register of the CPUIF, set the start address of the FIFO area from which to read and the data size in the RAM_RdAdr_H,L registers and RAM_RdCount register, respectively, and then set the RAM_RdControl.RAM_GoRd bit. When the data of the specified FIFO area is ready for read from the RAM_Rd register, the CPU_IntStat.RAM_RdCmp bit is set to 1. After confirming the RAM_RdCmp bit, read data from the RAM_Rd registers. The data in the RAM_Rd registers are stored in order of register numbers beginning with RAM_Rd_00. If the data size set in the RAM_RdCount register is smaller than 32 bytes, the data bytes in the RAM_Rd registers exceeding the set size are ignored.

FIFO data read via the RAM_Rd registers can be performed regardless of the settings for FIFO area of a relevant channel.

The RAM_Adrs_H,L and RAM_Count register values are updated one by one while the RAM_Rd function is in operation. Once the RAM_Rd function is activated, do not access these registers until the CPU_IntStat.RAM_RdCmp bit is set. If these registers are accessed for read while the RAM_Rd function is in operation, the read values cannot be guaranteed. Writing to these registers while in operation will cause the LSI to operate erratically.

6.6.2.4.2 Method for Accessing the FIFO (RAM_WrDoor)

To access the FIFO for write via the RAM_WrDoor_0,1 registers of the CPUIF, set the write start address in the RAM_WrAdr_H,L registers and write data via the RAM_WrDoor_0,1 registers. The RAM_WrAdr_H,L registers are automatically incremented by an amount equal to written bytes each time the FIFO is accessed for write, so that if data needs to be written to contiguous addresses, the data can be written to the RAM_WrDoor_0,1 registers successively.

Write to the FIFO via the RAM_WrDoor_0,1 registers can be performed regardless of the settings for FIFO area of a relevant channel.

6.6.2.4.3 Method for Accessing the FIFO (Register Access)

To access the FIFO for read via a register access of the CPU, set H_CHx{x=0,a-c}Join.JoinCPU_Rd for any one of channels to 1 and read data via the FIFO_Rd_0,1 or FIFO_ByteRd register.

To access the FIFO for write via a register access of the CPU, set H_CHx{x=0,a-c}Join.JoinCPU_Wr for any one of channels to 1 and write data in the FIFO_Wr_0,1 registers.

The FIFO_RdRemain_H,L registers indicate the remaining number of data bytes that can be read from the FIFO for only one channel that is set by H_CHx{x=0,a-c}Join.JoinCPU_Rd. Similarly, the FIFO_WrRemain_H,L registers indicate the remaining number of bytes in the

FIFO area to which data can be written for only one channel that is set by H_CHx{x=0,a-c}Join.JoinCPU_Wr.

Be aware that in cases where registers are to be dumped when debugging the firmware using an ICE or other tool, if any H_CHx{x=0,a-c}Join.JoinCPU_Rd register is set, data may inadvertently be read from the FIFO when registers are dumped.

6.6.2.4.4 Method for Accessing the FIFO (DMA)

To access the FIFO for read via a DMA access of the CPU, select only one channel on either DMA channel using the H_CHx{x=0,a-h}Join.JoinDMAx{x=0,1} bit and set the DMAx{x=0,1}_Control.Dir bit to 1, and then execute a DMA procedure to read data.

To access the FIFO for write via a DMA access of the CPU, select only one channel on either DMA channel using the H_CHx{x=0,a-h}Join.JoinDMAx{x=0,1} bit and set the DMAx{x=0,1}_Control.Dir bit to 0, and then execute a DMA procedure to write data.

The DMAx{x=0,1}_Remain_H,L registers indicate the remaining number of data bytes that can be read from the FIFO for only one channel on either DMA channel that is selected by the H_CHx{x=0,a-h}Join.JoinDMAx{x=0,1} bit. Similarly, they indicate the remaining number of bytes in the FIFO area to which data can be written for only one channel on either DMA channel that is selected by the H_CHx{x=0,a-h}Join.JoinDMAx{x=0,1} bit.

6.6.2.4.5 Method for Accessing the FIFO (IDE)

For the FIFO to be accessed by the IDE, select only one of channels by using the H_CHx{x=0,a-h}Join.JoinIDE bit and execute an IDE procedure to transfer data. The direction of IDE transfer is determined by the IDE_Control.Dir bit.

6.6.2.4.6 Limitations on FIFO Access

The FIFO of the LSI stipulated herein is designed in such a way that transmission/reception with the USB, register or DMA read/write from the CPU bus, and transmission/reception with the IDE are performed at the same time. Furthermore, read from the CPU bus is accomplished by look-ahead processing.

For the above reasons, the setup method (Join) for access to the FIFO on respective channels is subject basically to the following exclusion rules:

- For one channel, only one writing factors can be set.
- For one channel, only one reading factors can be set.
- For one channel, only one of JoinCPU_Wr, JoinCPU_Rd, or JoinDMAx{x=0,1} can be set.
- JoinIDE, JoinCPU_Wr, JoinCPU_Rd, and JoinDMAx{x=0,1} can respectively be set for only one channel at the same time.
- JoinIDE, JoinCPU_Wr, JoinCPU_Rd and JoinDMAx{x=0,1} cannot be set for FIFO and MediaFIFO at the same time.

- Only one of JoinCPU_Wr or JoinCPU_Rd can be set for FIFO at any given time.

There is an exception for the rule that says only one writing/reading factors can be set to one endpoint. For example, although it is possible to write to an IN-channel FIFO area after setting JoinCPU_Wr, there must be no IN transactions being performed when the JoinCPU_Wr is set. Similarly, although it is possible to read from an OUT-channel FIFO area after setting JoinCPU_Rd, there must be no OUT transactions being performed when the JoinCPU_Rd is set.

6.6.3 Media FIFO Management

This section describes FIFO management in the media data transfer function.

6.6.3.1 Media FIFO

The media FIFO is used for transfers of media data between IDE and memory.

The media FIFO is independent from the USB FIFO, and has 64 bytes of storage automatically allocated to it as its fixed usable area. Therefore, the firmware does not need to set an area for the media FIFO.

The used area is controlled as FIFO, so that the number of data bytes stored in it is retained. To clear this retained status, set MediaFIFO_Control.MediaFIFO_Clr.

6.6.3.2 Method for Accessing the FIFO

There are several methods to access the FIFO, including the CPU (registers), CPU (DMA), IDE, and USB.

6.6.3.2.1 Method for Accessing the FIFO (Register Access)

To access the FIFO for read via a register access of the CPU, set MediaFIFOJoin.JoinCPU_Rd to 1 and read data via the FIFO_Rd_0,1 or FIFO_ByteRd register.

To access the FIFO for write via a register access of the CPU, set MediaFIFOJoin.JoinCPU_Wr for any one of channels to 1 and write data to the FIFO_Wr_0,1 registers.

The FIFO_RdRemain_H,L registers indicate the remaining number of data bytes that can be read from the FIFO when a read is set by MediaFIFOJoin.JoinCPU_Rd. Similarly, the FIFO_WrRemain_H,L registers indicate the remaining number of bytes in the FIFO area to which data can be written to when a write is set by MediaFIFOJoin.JoinCPU_Wr.

Be aware that in cases where registers are to be dumped when debugging the firmware using an ICE or other tool, if the MediaFIFOJoin.JoinCPU_Rd register is set, data may inadvertently be read from the FIFO when registers are dumped.

6.6.3.2.2 Method for Accessing the FIFO (DMA)

To access the FIFO for read via a DMA access of the CPU, select only one channel on either DMA channel using the MediaFIFOJoin.JoinDMAx{x=0,1} bit and set the DMAx{x=0,1}_Control.Dir bit to 1, and then execute a DMA procedure to read data.

To access the FIFO for write via a DMA access of the CPU, select only one channel on either DMA channel using the MediaFIFOJoin.JoinDMAx{x=0,1} bit and set the DMAx{x=0,1}_Control.Dir bit to 0, and then execute a DMA procedure to write data.

The DMAx{x=0,1}_Remain_H,L registers indicate the remaining number of data bytes that can be read from the FIFO for only one channel on either DMA channel that is selected by the MediaFIFOJoin.JoinDMAx{x=0,1} bit. Similarly, they indicate the remaining number of bytes in the FIFO area to which data can be written for only one channel on either DMA channel that is selected by the MediaFIFOJoin.JoinDMAx{x=0,1} bit.

6.6.3.2.3 Method for Accessing the FIFO (IDE)

For the FIFO to be accessed by the IDE, set the MediaFIFOJoin.JoinIDE bit to 1 and execute an IDE procedure to transfer data. The direction of IDE transfer is determined by the IDE_Control.Dir bit.

6.6.3.2.4 Limitations on FIFO Access

The FIFO of the LSI stipulated herein is designed in such a way that register or DMA read/write from the CPU bus and transmission/reception with the IDE are performed at the same time. Furthermore, read from the CPU bus is accomplished by look-ahead processing.

For the above reasons, the setup method (Join) for access to the Meaia FIFO is subject to the following exclusion rules:

- For one area, only one of writing factors can be set.
- For one area, only one of reading factors can be set.
- For the media FIFO, only one of JoinCPU_Wr, JoinCPU_Rd, or JoinDMAx{x=0,1} can be set.
- JoinIDE, JoinCPU_Wr, JoinCPU_Rd and JoinDMAx{x=0,1} cannot be set for FIFO and MediaFIFO at the time.
- Only one of JoinCPU_Wr or JoinCPU_Rd can be set for MediaFIFO at any given time.

6.7 CPUIF

6.7.1 Mode Switching

The CPUIF of the S1R72V05 accommodates asynchronous CPUs, and has the following three operation modes.

Table 6.62 CPUIF Operation Mode Settings

Operation Mode	BusMode	Bus8x16	Remark
16bit Strobe mode	0	0	Default
16bit BE mode	1	*	BusMode bit settings have priority
8bit mode	0	1	

Switching between these operation modes is accomplished by setting the BusMode and Bus8x16 bits in the ChipConfig register. The value of the ChipConfig register can be protected against erroneous writes by setting the ModeProtect register.

In actual use, first set the ChipConfig register immediately after power-on to determine operation mode. Then set the ModeProtect register to write protect it.

In addition, the CPUIF of the S1R72V05 has a bus swap function. To use this function, set the ChipConfig.CPU_Endian bit when initially setting up the ChipConfig register. The Swap function is enabled by reading the address B9h after setting the CPU_Endian bit. Furthermore, it is possible to set the XINT logic level and output mode, the logic levels of XDREQ0,1 and XDACK0,1, and the CS_Mode of DMA0,1 during initial setting of the ChipConfig register.

In the description below, explanations are made based on default settings (16-bit Strobe mode, no Bus Swap) unless otherwise noted.

6.7.2 Notes on Mode Switching

The S1R72V05 allows the CPU bus-operating mode suitable for the CPU to be set using the ChipConfig register. In its initial state, the chip operates in 16-bit Strobe mode. You must observe the following precautions if you switch operating modes to 16-bit BE or 8-bit modes.

6.7.2.1 When Using 16-bit BE Mode

To use the 16-bit BE mode for your CPU, first set the ChipConfig register as described in Section 6.7.1. When setting this register, be sure to write data in bytes to address B7h, as shown in Fig. 6.78. Since the S1R72V05 operates in its initial state of 16-bit Strobe mode, if the chip select and byte mask high signals of the CPU (XCS and XWRH) have a skew like the one shown below, the LSI may misidentify it as a valid write period and proceed on that assumption. Although the S1R72V05 incorporates a filter circuit (min: 1 ns) to eliminate such skews, you should still confirm the AC characteristics of the CPU used and process the board as necessary to prevent such skews.

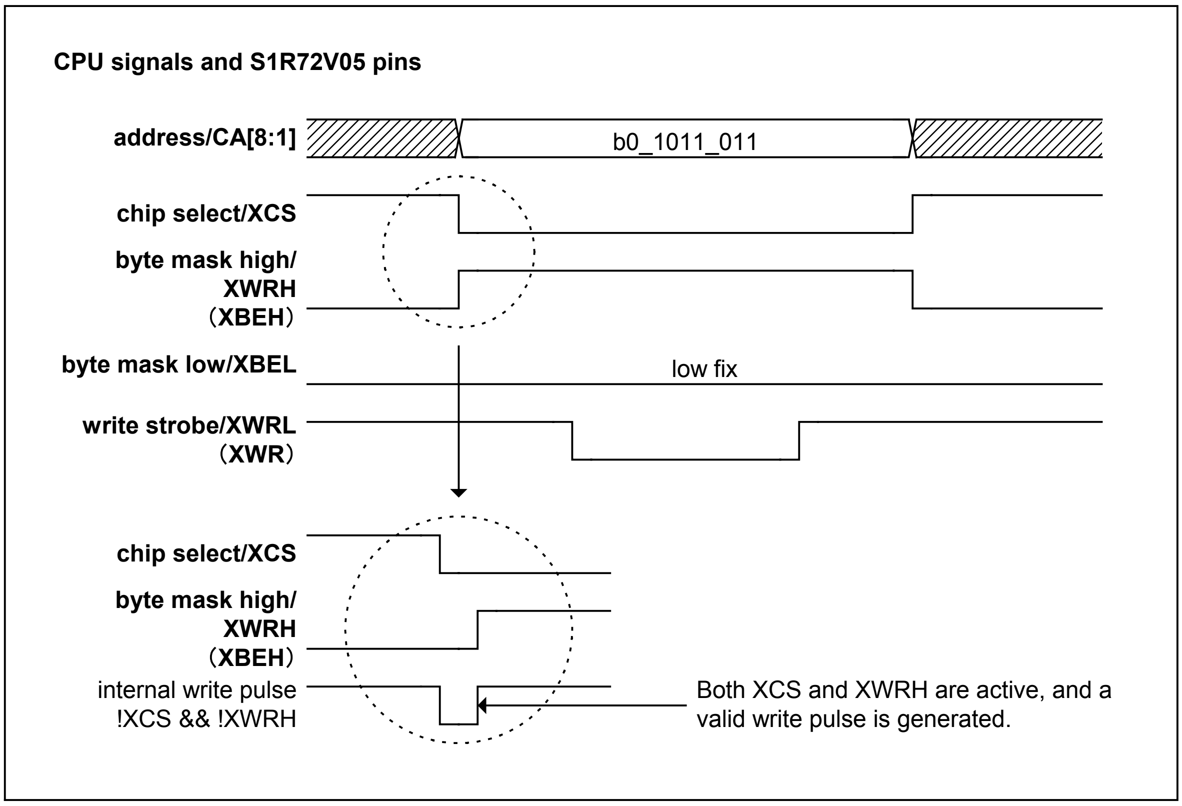

CPU signals and S1R72V05 pins
address/CA[8:1]
b0_1011_011
chip select/XCS
byte mask high/ XWRH (XBEH)
byte mask low/XBEL
low fix
write strobe/XWRL (XWR)
chip select/XCS
byte mask high/ XWRH (XBEH)
internal write pulse !XCS && !XWRH
Both XCS and XWRH are active, and a valid write pulse is generated.

Fig. 6.78 Initializing the ChipConfig register

This limitation no longer applies once operating mode settings are completed, since internal write pulse generation conditions are updated for 16-bit BE mode.

If the S1R72V05 is accessed for reads before the ChipConfig register is set, an internal problem may occur in the chip in which read and write operations are simultaneously performed, as shown in Fig. 6.79. LSI operations cannot be guaranteed in such cases. Be sure to set the ChipConfig register first.

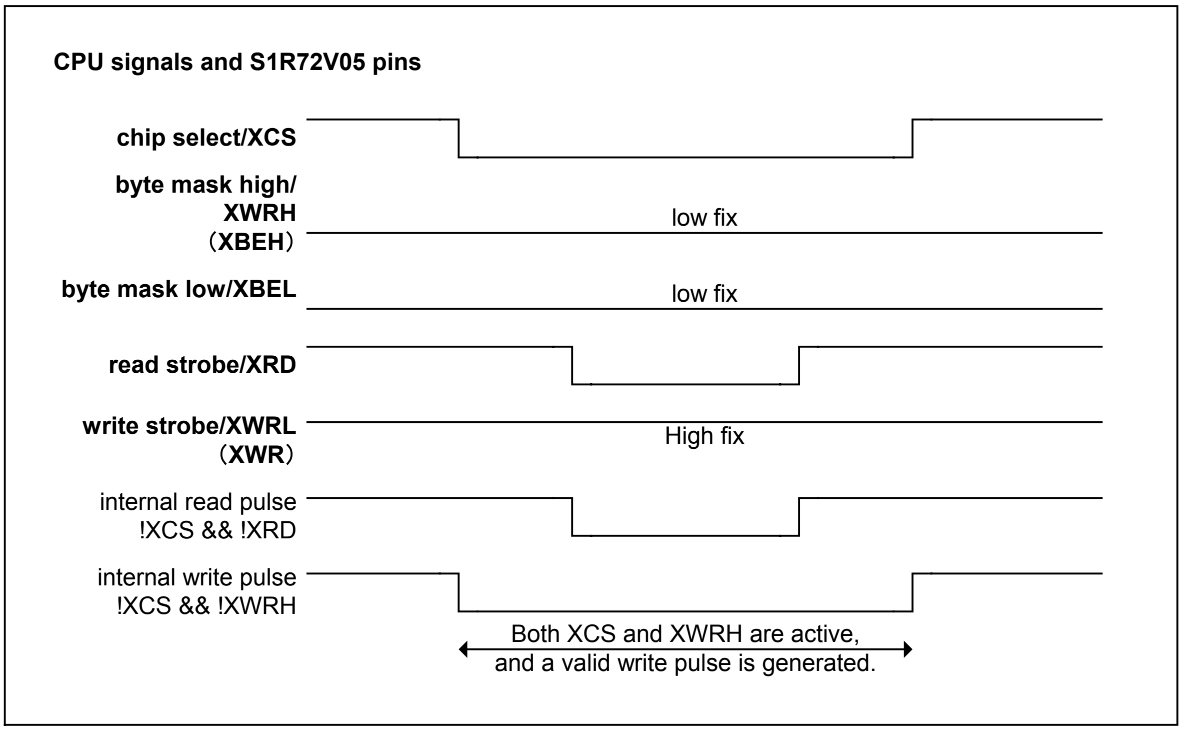

CPU signals and S1R72V05 pins
chip select/XCS
byte mask high/ XWRH (XBEH)
low fix
byte mask low/XBEL
low fix
read strobe/XRD
write strobe/XWRL (XWR)
High fix
internal read pulse !XCS && !XRD
internal write pulse !XCS && !XWRH
Both XCS and XWRH are active, and a valid write pulse is generated.

Fig. 6.79 Read access before the ChipConfig register is initialized

6.7.2.2. When Using 8-bit Mode

If the CPU used is to run in 8-bit mode, first set the ChipConfig register as described in Section 6.7.1. If the S1R72V05 is accessed for reads before the ChipConfig register is set, all of the CD[15:0] pins will be placed in an output state, since the S1R72V05 operates in its initial state of 16-bit Strobe mode. No specific problems will result if CD[15:8] pins are internally pulled high or low through the resistors, but if these pins are connected directly to VDD or GND, chip current consumption will increase significantly. <u>Be sure to set the ChipConfig register first to avoid such increase</u>.

6.7.3 Block Configuration

The block configuration of the CPUIF of the S1R72V05 (hereafter referred to as "CPUIF") is shown in Fig. 6.78.

The CPUIF is comprised of three blocks—REG, DMA0, and DMA1.

- REG: Controls access to the S1R72V05 register area
- DMA0: DMA channel 0
- DMA1: DMA channel 1

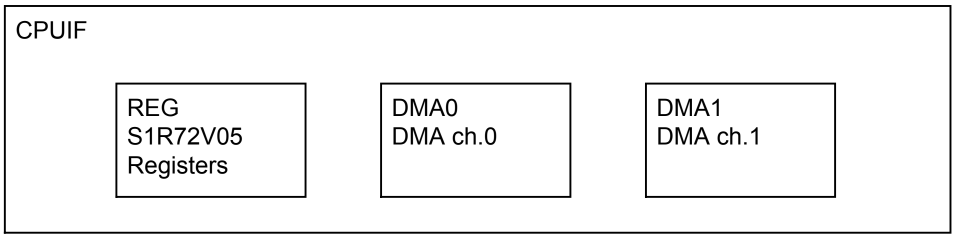


Fig. 6.80 Block Configuration

6.7.3.1 REG (S1R72V05 Registers)

Controls access to the S1R72V05 register area. This includes the following access functions:

- Synchronous register access
- FIFO access
- RAM_Rd access
- Asynchronous register access

6.7.3.1.1 Synchronous Register Access (Write)

In this access, external bus data is written to the registers synchronously with the internal clock.

6.7.3.1.2 Synchronous Register Access (Read)

In this access, the register data is output to the external bus during a read period in which XCS and XRD both are asserted as an output enable period.

In a register read operation, the registers that accommodate 3 bytes or more as meaningful data as in the case of a count value (for 8-bit mode, 2 bytes or more) require caution, because it is possible that an erroneous count value will be read due to a carry of the count that may occur during the access cycle. To avoid this problem, the lower-byte register value is latched when the most significant byte is read, and the latched value is output to the external bus when the lower bytes are read.

6.7.3.1.3 FIFO Access (Write)

The FIFO write access refers to writing data to the FIFO_Wr_0,1 and RAM_WrDoor_0,1 registers. When operating in 8-bit mode, either one of the FIFO_Wr_0,1 registers can be accessed for write to the FIFO without causing any problem. The same applies to the RAM_WrDoor_0,1 registers.

The FIFO access (write) is subject to the following limitations:

- After setting either one of the D_EPx{x=0,a-c}Join.JoinCPU_Wr, H_CHx{x=0,a-e}Join.JoinCPU_Wr, or MediaFIFO_Join.JoinCPU_Wr bit, inspect the FIFO_WrRemain_H,L registers to confirm the number of writable data bytes before accessing the FIFO. This limitation does not apply to the RAM_WrDoor_0,1 registers.
- When using a 16-bit CPU, the FIFO must basically be accessed by word (in 2-byte units). For writes of odd bytes, take the byte boundaries of the FIFO into consideration when controlling the strobe signal. For details, refer to Section 6.7.2.1.5, "Processing Odd Bytes in FIFO Access."
- The FIFO_WrRemain_H,L registers, if inspected immediately after writing to the FIFO_Wr_0,1 registers, will not show the exact amount of free space in the FIFO. Be sure to insert an interval equal to 1 CPU cycle or more before reading the FIFO_WrRemain_H,L registers.
- The RAM_WrDoorAdr_H,L registers, if inspected immediately after writing to the RAM_WrDoor_0,1 registers, will not show the exact address. Be sure to insert an interval equal to 1 CPU cycle or more before reading the RAM_WrDoorAdr_H,L registers.

6.7.3.1.4 FIFO Access (Read)

The FIFO read access refers to reading out data from the FIFO_Rd_0,1 or FIFO_ByteRd registers. When operating in 8-bit mode, either FIFO_Rd_0 or 1 or FIFO_ByteRd register can be accessed for read from the FIFO without causing any problem.

The FIFO read access is subject to the following limitations:

- After setting either one of the D_EPx{x=0,a-c}Join.JoinCPU_Wr, H_CHx{x=0,a-e}Join.JoinCPU_Wr, or MediaFIFO_Join.JoinCPU_Wr bit, inspect the FIFO_RdRemain_H,L registers to confirm the number of readable data bytes and also check the RdRemainValid bit before accessing the FIFO.
- When operating in 16-bit mode, use the FIFO_Rd_0,1 registers to read data by word. Use the FIFO_ByteRd register to read data bytewise. If byte boundaries exist, read data bytewise. In this case, if data is read by word using the FIFO_Rd_0,1 registers, valid data is output only one side of the registers. For details, refer to Section 6.7.2.1.5, "Processing Odd Bytes in FIFO Access."

6.7.3.1.5 Processing Odd Bytes in FIFO Access

This section describes the relationship between the manner of how data is stored in the FIFO and the FIFO access made when handling odd bytes of data. Although the actual FIFO is 4 bytes in width, the FIFO in the explanation below is referred to as being 2 bytes in width for simplicity purpose. There are no operational differences between 4 bytes and 2 bytes.

[Write operation]

Basically, we recommend that a write operation be performed from a byte boundary nonexistent state.

If odd data is found present after data was written wordwise from a byte boundary nonexistent state by setting the D_EPnControl.EP0FIFO_Clr bit, D_EPrFIFO_Clr.EPx{x=a-c}FIFO_Clr bit (during device) or H_CHnControl.CH0FIFO_Clr bit (during host) or the MediaFIFO_Control.MediaFIFO_Clr bit (during media data transfer), etc. as necessary, make sure that only the last byte of consecutive data (i.e., data Z) is written to the High side. This state is shown in (1) of Fig. 6.79. The data is output from the USB, etc. in order of A, B, C, D, ... X, Y, and Z.

To write data while the FIFO has a byte boundary in it, first write data to the Low side (write of data K) to eliminate the byte boundary and then write data wordwise (data L and M). This state is shown in (2) of Fig. 6.79.

Described above are the normal write operations.

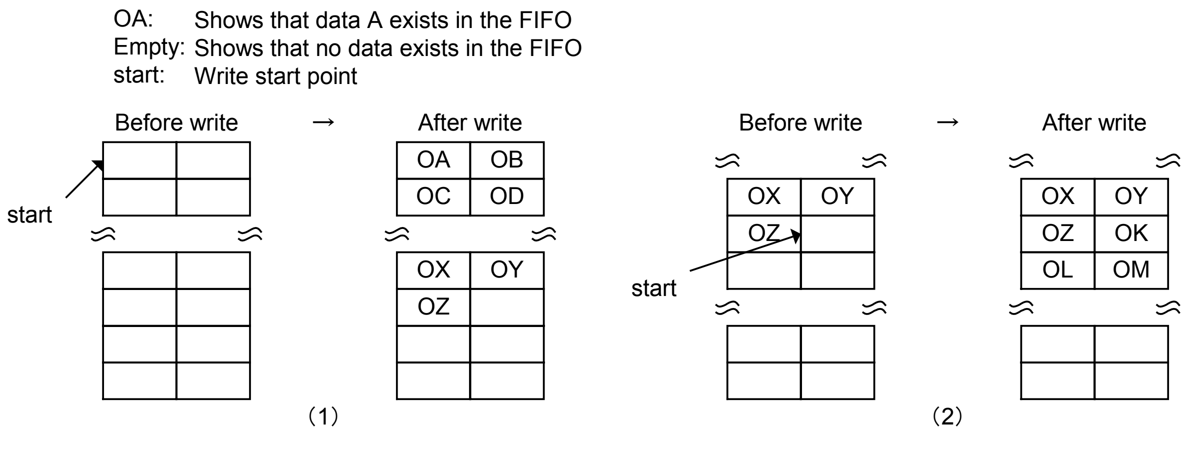


Fig. 6.81 FIFO write processing (normal operations)

Described below are the write operations that require caution.

If data is written wordwise while the FIFO has a byte boundary in it, a write to the High side is ignored, and data is written to only the Low side ((3) in Fig. 6.82). This is the same as writing data to the Low side bytewise. Furthermore, if data is written to only the High side while the FIFO has a byte boundary in it, the write operation performed is ignored ((4) in Fig. 6.82).

If data is written to only the Low side while the FIFO has no byte boundary in it, the write operation performed is ignored ((5) in Fig. 6.82). Furthermore, if data is written wordwise while the FIFO has no byte boundary in it and the number of writable bytes is 1, a write to the Low side is ignored and data is written to only the High side ((6) in Fig. 6.82). This is the same as writing data to the High side bytewise.

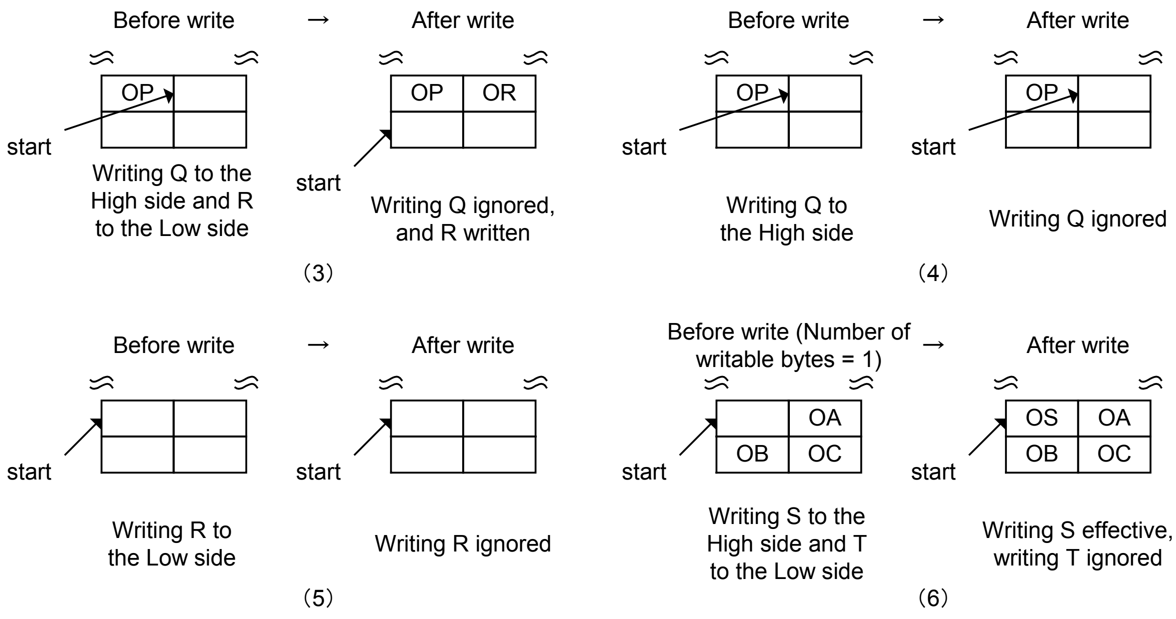

Before write
→
After write
OP
OP
OR
start
start
Writing Q to the High side and R to the Low side
Writing Q ignored, and R written
(3)
Before write
→
After write
OP
OP
start
start
Writing Q to the High side
Writing Q ignored
(4)
Before write
→
After write
start
start
Writing R to the Low side
Writing R ignored
(5)
Before write (Number of writable bytes = 1)
→
After write
OA
OB
OC
OS
OA
OB
OC
start
start
Writing S to the High side and T to the Low side
Writing S effective, writing T ignored
(6)

Fig. 6.82 FIFO write processing (operations that require caution)

[Read operation]

If no byte boundaries exist, data can be read wordwise using the FIFO_Rd_0,1 registers or can be read bytewise using the FIFO_ByteRd register without causing any problem. If any byte boundary exists, data must be read bytewise using the FIFO_ByteRd register. Once the byte boundary is eliminated, data can be read either wordwise or bytewise without causing any problem.

The manner of how data is read wordwise when no byte boundaries exist is shown in (1) of Fig. 6.83. Data A,B and then data C,D are read each time the FIFO is accessed. The manner of how data is read bytewise is shown in (2) of Fig. 6.83. Data A, data B, data C, and data D are read each time the FIFO is accessed. Described above are the normal read operations.

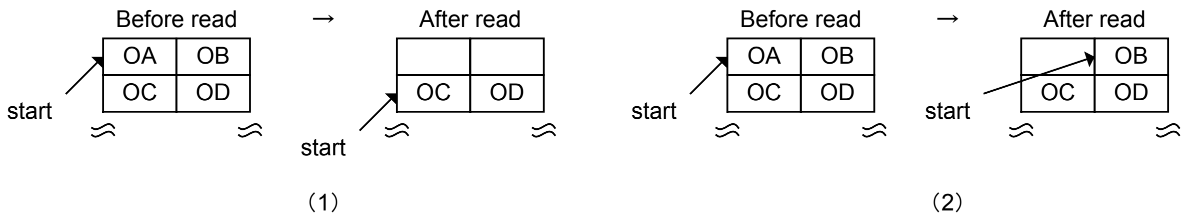


Fig. 6.83 FIFO read processing (normal operations)

Described below are the read operations that require caution.

Shown in (3) of Fig. 6.84 is an operation in which data is read wordwise using the FIFO_Rd_0,1 registers while a byte boundary exists. Indeterminate data is output to the High side, and data J is output to the Low side. The read pointer increments for only 1 byte of data. Shown in (4) of Fig. 6.84 is an operation in which data is read wordwise using the FIFO_Rd_0,1 registers while no byte boundary exists but the remaining bytes of data = 1. Data X is output to the High side, and indeterminate data is output to the Low side. The read pointer increments for only 1 byte of data.

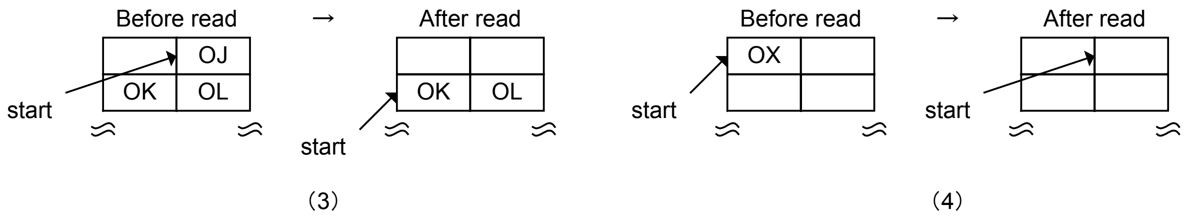


Fig. 6.84 FIFO read processing (operations that require caution)

Based on the above, the following shows an example read operation in odd bytes processing.

1) To read 64 bytes of data sent from the USB, first 31 bytes and then 33 bytes

   (1) The CPUIF latches Ready for 64 bytes to start a series of read sequences.

   (2) The 30 bytes of data are read wordwise using the FIFO_Rd_0,1 registers or read bytewise using the FIFO_ByteRd register.

   (3) The 31st byte of data is read bytewise using the FIFO_ByteRd register. ->A byte boundary is created.

   (4) The 32nd byte of data is read bytewise. In this case, it is recommended that the FIFO_ByteRd register be used for byte read. If the data is read wordwise using the FIFO_Rd_0,1 registers, the read data is output to the Low side. ->The byte boundary is eliminated.

   (5) The remaining 32 bytes of data are read wordwise using the FIFO_Rd_H,L registers or read bytewise using the FIFO_ByteRd register.

2) To read 64 bytes of data sent separately 31 bytes and then 33 bytes from the USB while JoinCPU_Rd is set, all wordwise by using the FIFO_Rd_0,1 registers

   (1) When 31 bytes of data is received from the USB, the CPUIF latches Ready for 31 bytes to start a series of operation sequences.

   (2) The 30 bytes of data are read wordwise.

   (3) To eliminate the cached 31st byte of data (byte boundary), Join is temporarily disconnected.

   (4) After 33 bytes of data have been sent from the USB, Join is reconnected. (1 + 33 bytes)

   (5) The CPUIF latches Ready for 34 bytes to start a series of operation sequences.

   (6) The 34 bytes of data are read wordwise.

6.7.3.1.6 RAM_Rd Access

As with synchronous register read, data is output to the external bus during a read period in XCS and XRD both are asserted as an output enable period. For details, refer to Section 6.6.1.5.1 or 6.6.2.4.1, “Method for Accessing the FIFO (RAM_Rd).”

6.7.3.1.7 Asynchronous Register Access (Write)

After creating a write pulse from the external write signals (XCS and XWRL,H), external bus data is written to the registers.

6.7.3.1.8 Asynchronous Register Access (Read)

As with synchronous register read, the register data is output to the external bus during a read period in which XCS and XRD both are asserted as an output enable period.

6.7.3.2 DMA0/DMA1 (DMA Channels 0/1)

6.7.3.2.1 Basic Functionality

The basic operations of the DMA are described below.

[Write operation]

If the FIFO has a writable free space, XDREQ is asserted to enable DMA transfers to be performed.

[Read operation]

When the FIFO has valid readable data and is readable, XDREQ is asserted to enable DMA transfers to be performed.

The DMA has two operation modes and one operation option.

- Count mode
  DMA transfers are performed a number of times equal to the counts set.
  When the internal FIFO has a writable free space or valid readable data and there is a remaining count in the DMAx{x=0,1}_Count_HH,HL,LH,LL registers, XDREQ is asserted to enable DMA transfers to be performed.
- Free-running mode
  When the internal FIFO has a writable free space or valid readable data, XDREQ is asserted to enable DMA transfers to be performed.

- REQ assert count option
  This option is provided for burst read/write by the CPU. This option can be used in either count mode or free-running mode. If the FIFO has a writable free space or valid readable data more than the assert counts set by the DMAx{x=0,1}_Config.ReqAssertCount [1:0] bits, XDREQ is asserted to enable DMA transfers to be performed. This means that once XDREQ is asserted, transfers for bytes equal to the set assert counts are guaranteed. However, even when the free space or data in the FIFO is less than the set assert count, if count mode is selected and the said free space or data in the FIFO is greater than the remaining count, then XDREQ is asserted. In this case, the guaranteed transfer bytes equal the remaining count.

In 16-bit mode, DMA basically is data processed in word units. Data processing in byte units can be performed only when DMA is operating in count mode and the remaining count = 1. The table below lists the relationship between XDREQ assert conditions and the number of transfers performed in each operation mode with the option used or unused.

Table 6.63 Operation Modes and Option vs. Transfer Start Conditions

Count mode with the ReqAssertCount option used (when operating in 16-bit or 8-bit mode)

Condition	Count mode (Count > 0)			
	Count≧Ready		Count < Req	
	Ready≧Req	Ready < Req	Ready≧Count	Ready < Count
XDREQ	Asserted	Negated	Asserted	Negated
Transfers performed	Req	-	Count	-

Free-running mode with the ReqAssertCount option used (when operating in 16-bit or 8-bit mode)

Condition	Free-running mode	
	-	
	Ready≧Req	Ready < Req
XDREQ	Asserted	Negated
Transfers performed	Req	-

Count mode with the ReqAssertCount option unused (when operating in 16-bit mode)

Condition	Count mode (Count > 0)		
	Count≧Ready		Count < Ready
	Ready≧2	Ready < 2	Ready≧Count
XDREQ	Asserted	Negated	Asserted
Transfers performed	Ready (if Ready is an odd number, Ready - 1)	-	Count

Free-running mode with the ReqAssertCount option unused (when operating in 16-bit mode)

Condition	Free-running mode	
	-	
	Ready≧2	Ready < 2
XDREQ	Asserted	Negated
Transfers performed	Ready (if Ready is an odd number, Ready - 1)	-

Count mode with the ReqAssertCount option unused (when operating in 8-bit mode)

Condition	Count mode (Count > 0)		
	Count≧Ready		Count < Ready
	Ready≧1	Ready < 1	Ready≧Count
XDREQ	Asserted	Negated	Asserted
Transfers performed	Ready	-	Count

Free-running mode with the ReqAssertCount option unused (when operating in 8-bit mode)

Condition	Free-running mode	
	-	
	Ready≧1	Ready < 1
XDREQ	Asserted	Negated
Transfers performed	Ready	-

* In the above table, Req indicates the set value of DMAx{x=0,1}_Config.ReqAssertCount, Ready indicates the free space/data bytes in the FIFO, and Count indicates the value of DMAx{x=0,1}_Count_HH,HL,LH,LL.

6.7.3.2.2 Pin Settings

It is possible to set the DREQx{x=1,0} and XDACKx{x=1,0} logic levels each by setting up the ChipConfig register. In the explanation below, XDREQ and XDACK both are described as being active-low (negative logic) unless otherwise noted.

6.7.3.2.3 Count Mode (Write)

[Starting operation]

After setting a count value in the DMAx{x=0,1}_Count_HH,HL,LH,LL registers, set the DMAx{x=0,1}_Control_DMA_Go bit to 1. If the internal FIFO has 2 bytes or more of writable free space (DMA_Ready) (for 8-bit mode, 1 byte or more) and has any remaining count, XDREQ is asserted to enable DMA transfers to be performed. If the free space available in the FIFO is only 1 byte, count mode is selected, in which case XDREQ is asserted only when the remaining count = 1.

If a byte boundary is created in the FIFO as a result of odd bytes written into it, clear the FIFO after data is transferred from the USB, etc. to eliminate the byte boundary before starting the next write operation. To transfer data from the USB 31 bytes each time after writing data from the DMA 31 bytes each time, for example, (1) set the DMA count value to 31 and write 31 bytes of data, (2) wait until 31 bytes of data are transferred to the USB, (3) after confirming that 31 bytes of data have been transferred from the USB, clear the FIFO. Repeat the above operations.

Until the operation stops, the DMAx{x=0,1}_Control.DMA_Running bit reads “1.”

[Stopping operation]

There are two conditions to stop the operation:

- DMA transfers equal to the counts set in the DMAx{x=0,1}_Count_HH,HL,LH,LL registers are completed.
- The DMAx{x=0,1}_Control.DMA_Stop bit is set by writing 1 in software.

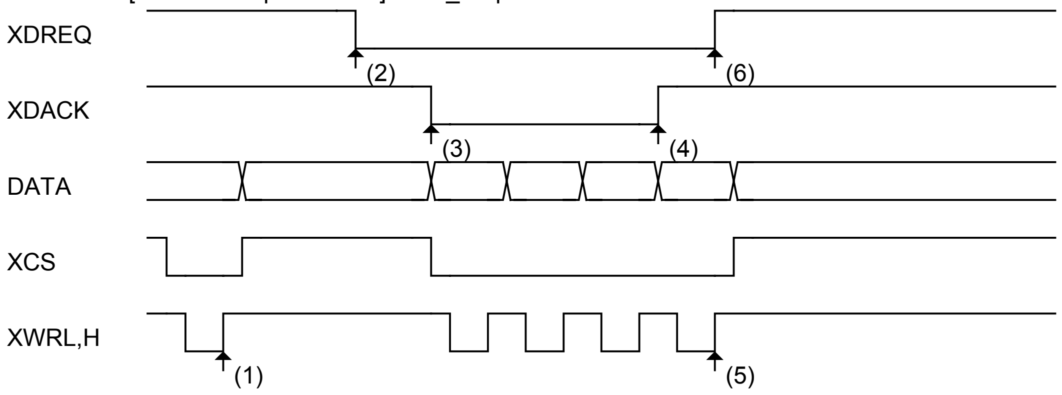
When the DMA operation stops, the CPU_IntStat.DMAx{x=0,1}_Cmp bit is set.

When the transfer stops due to expiration of counts set in the DMAx{x=0,1}_Count_HH,HL,LH,LL registers, XDREQ is negated during the strobe assert period of the last access.

When the transfer stops due to DMA_Stop bit, the chip’s internal operation is halted in synch with the write timing of a synchronous register access and then XDREQ is negated. To have the DMA stopped by the DMA_Stop bit, stop the DMAC (master) on the CPU side first.

Fig. 6.85 shows an operation timing for the case where a transfer is started in count mode and the transfer is stopped by setting the DMAx{x=0,1}_Control.DMA_Stop bit before transfers for the set count are completed.

Example 1: [Transfer start condition] Count (8 bytes) < free space in FIFO (16 bytes)
[Transfer stop condition] DMA_Stop

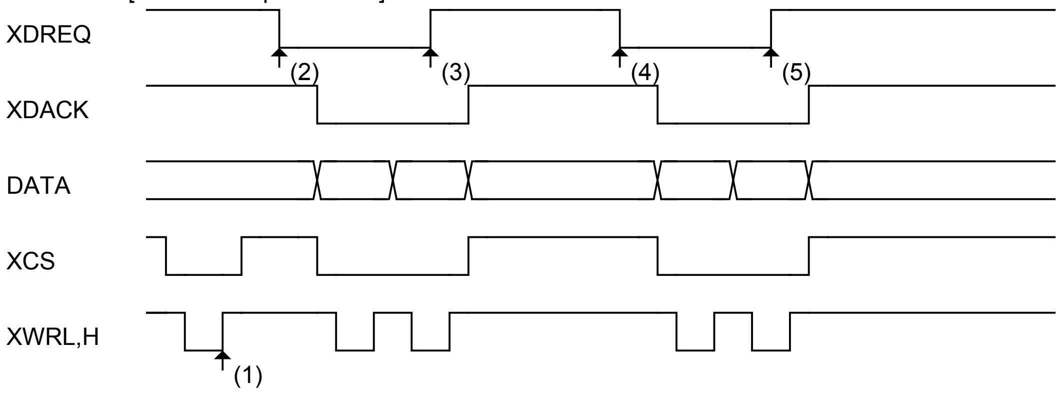


(1) The DMA circuit is activated by writing 1 to the DMA_Control.DMA_Go bit.
(2) A free space is created in the FIFO (DMA_Ready) due to transfer of data from the USB, etc., and XDREQ is asserted in response to DMA_Ready.
(3) XDACK is asserted causing a DMA transfer to start.
(4) The master is stopped and XDACK is negated before transfers in count mode are completed.
(5) The DMA circuit is deactivated by writing 1 to the DMA_Control.DMA_Stop bit.
(6) XDREQ is negated in response to deactivation of the DMA circuit.

Fig. 6.85 Count mode write timing 1

Fig. 6.86 shows an operation timing for the case where a transfer is started in count mode and when the DMA transfer finishes due to completion of transfers for the set count.

Example 2: [Transfer start condition] Count (8 bytes) < free space in FIFO (4 bytes)
[Transfer stop condition] Count 0



(1) The DMA circuit is activated by writing 1 to the DMA_Control.DMA_Go bit.
(2) A free space is created in the FIFO (DMA_Ready) due to the transfer of data from the USB, etc., and XDREQ is asserted in response to DMA_Ready.
(3) XDREQ is negated in synch with the disappearance of DMA_Ready.
(4) A free space is created in the FIFO (DMA_Ready) due to the transfer of data from the USB, etc., and XDREQ is asserted in response to DMA_Ready.
(5) XDREQ is negated in synch with the last data timing of DMA_Count. The DMA circuit stops due to completion of transfers equal to DMA_Count.

Fig. 6.86 Count mode write timing 2

6.7.3.2.4 Count Mode (Read)

[Starting operation]

After setting a count value in the DMAx{x=0,1}_Count_HH,HL,LH,LL registers, set the DMAx{x=0,1}_Control_DMA_Go bit to 1. If the internal FIFO has 2 bytes or more of valid readable data (for 8-bit mode, 1 byte or more) and has any remaining count, and while in that state, the FIFO is prepared for read from an external device, XDREQ is asserted to enable DMA transfers to be performed. If the remaining data available in the FIFO is only 1 byte, count mode is selected, in which case XDREQ is asserted only when the remaining count = 1.

For read in count mode during USB device operation for example, when bytes of data more than the counts set in the DMAx{x=0,1}_Count_HH,HL,LH,LL registers have accumulated in the FIFO for the endpoint to which the present DMA is connected, the ForceNAK bit is automatically set to 1 to return a NAK response. Furthermore, even when a short packet is received from the USB, unless the DisAF_NAK_Short is set, the ForceNAK bit for the relevant endpoint is automatically set to 1 to return a NAK response.

If a byte boundary is created in the FIFO as a result of odd bytes read from it, clear the FIFO to eliminate the byte boundary before performing the next transfer. To read data from the DMA 31 bytes each time after data is transferred from the USB 31 bytes each time, for example, (1) receive 31 bytes of data from the USB (at this point, the ForceNAK is set and the relevant endpoint returns a NAK response), (2) read 31 bytes of data from the DMA, (3) clear the FIFO and then the ForceNAK to allow for transfers from the USB to be received. Repeat the above operations.

Until the operation stops, the DMAx{x=0,1}_Control.DMA_Running bit reads “1.”

[Stopping operation]

There are two conditions to stop the operation:

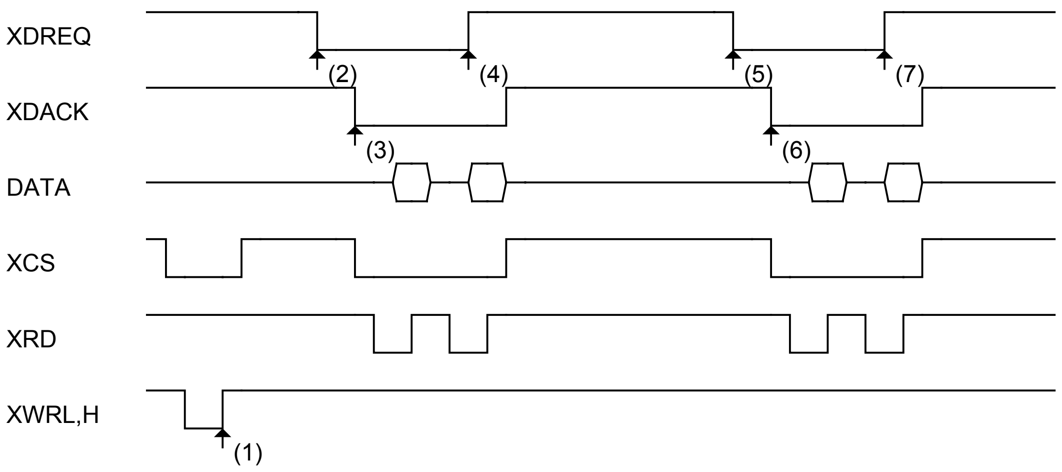
- DMA transfers equal to the counts set in the DMAx{x=0,1}_Count_HH,HL,LH,LL registers are completed.
- The DMAx{x=0,1}_Control.DMA_Stop bit is set by writing 1 in software.

When the transfer stops due to expiration of counts set in the DMAx{x=0,1}_Count_HH,HL,LH,LL registers, XDREQ is negated during the strobe signal assert period of the last access.

When the transfer stops due to the DMA_Stop bit, the chip’s internal operation is halted simultaneously with the write timing of a synchronous register access and then XDREQ is negated. To have the DMA stopped by the DMA_Stop bit, stop the DMAC (master) on the CPU side first.

Fig. 6.87 shows an operation timing for the case where a transfer is started in count mode, and when the DMA transfer finishes due to completion of transfers for the set count.

Example: [Transfer start condition] Count (8 bytes) < data in FIFO (4 bytes)
[Transfer stop condition] Count 0



(1) The DMA circuit is activated by writing 1 to the DMA_Control.DMA_Go bit.
(2) When data is written to the FIFO from the USB, etc., and the FIFO is readable from an external device, XDREQ is asserted.
(3) XDACK is asserted causing a DMA transfer to start.
(4) XDREQ is negated in synch with the timing at which the FIFO is emptied.
(5) When data is written into the FIFO from the USB, etc., and the FIFO is readable from an external device, XDREQ is asserted.
(6) XDACK is asserted causing a DMA transfer to start.
(7) XDREQ is negated in synch with the last data timing of DMA_Count.

Fig. 6.87 Count mode read timing

6.7.3.2.5 Free-running Mode (Write)

[Starting operation]

After setting the DMAx{x=0,1}_Config.FreeRun bit, set the DMAx{x=0,1}_Control_DMA_Go bit by writing 1. If the internal FIFO has 2 bytes or more of writable free space (DMA_Ready) (for 8-bit mode, 1 byte or more), XDREQ is asserted to enable DMA transfers to be performed. If the free space available in the FIFO is only 1 byte, XDREQ is not asserted when in free-running mode. If transfers need to be performed, refer to the explanation of count mode in the preceding section.

Until the operation stops, the DMAx{x=0,1}_Control.DMA_Running bit reads “1.”

[Stopping operation]

There is one condition to stop the operation:

- The DMAx{x=0,1}_Control.DMA_Stop bit is set by writing 1 in software.

When the transfer stops due to the DMA_Stop bit, the chip’s internal operation is halted in synch with the write timing of a synchronous register access and then XDREQ is negated. To have the DMA stopped by the DMA_Stop bit, stop the DMAC (master) on the CPU side first.

If the DMAx{x=0,1}_Count_HH,HL,LH,LL register value overflows after reaching terminal count during a DMA transfer in free-running mode, the

CPU_IntStat.DMAx{x=0,1}_Countup bit is set. Even in this case, the DMA transfer is continued and the count in the DMAx{x=0,1}_Count_HH,HL,LH,LL restarts incrementing.

The operation timing in free-running mode is the same as in count mode, except that the limitations due to DMAx{x=0,1}_Count_HH,HL,LH,LL are nonexistent.

6.7.3.2.6 Free-running Mode (Read)

[Starting operation]

After setting the DMAx{x=0,1}_Config.FreeRun bit, set the DMAx{x=0,1}_Control_DMA_Go bit by writing 1. If the internal FIFO has 2 bytes or more of valid readable data (for 8-bit mode, 1 byte or more) and is readable from an external device, XDREQ is asserted. If the remaining valid data in the FIFO is only 1 byte, DMA transfer is not started. If transfers need to be performed, refer to the explanation of count mode in the preceding section.

Until the operation stops, the DMAx{x=0,1}_Control.DMA_Running bit reads "1."

[Stopping operation]

There is one condition to stop the operation:

- The DMAx{x=0,1}_Control.DMA_Stop bit is set by writing 1 in software.

When the transfer stops due to the DMA_Stop bit, the chip's internal operation is halted in synch with the write timing of a synchronous register access and then XDREQ is negated. To have the DMA stopped by the DMA_Stop bit, stop the DMAC (master) on the CPU side first.

If the DMAx{x=0,1}_Count_HH,HL,LH,LL register value overflows after reaching terminal count during a DMA transfer in free-running mode, the CPU_IntStat.DMAx{x=0,1}_Countup bit is set. Even in this case, the DMA transfer is continued and the count in the DMAx{x=0,1}_Count_HH,HL,LH,LL restarts incrementing.

The operation timing in free-running mode is the same as in count mode, except that the limitations due to DMAx{x=0,1}_Count_HH,HL,LH,LL are nonexistent.

6.7.3.2.7 REQ Assert Count Option (Write)

[Starting operation]

After setting an assert count with the DMAx{x=0,1}_Config.ReqAssertCount [1:0] bits, set the DMAx{x=0,1}_Control_DMA_Go bit to 1. If the internal FIFO has more bytes of writable free space than the set assert count, XDREQ is asserted to enable DMA transfers to be performed. This means that once XDREQ is asserted, transfers for bytes equal to the set assert count are guaranteed. However, even when the free space in the FIFO is less than the set assert count, if count mode is selected and the said free space in the FIFO is greater than the remaining count, then XDREQ is asserted. In this case, the guaranteed transfer bytes equal the remaining count.

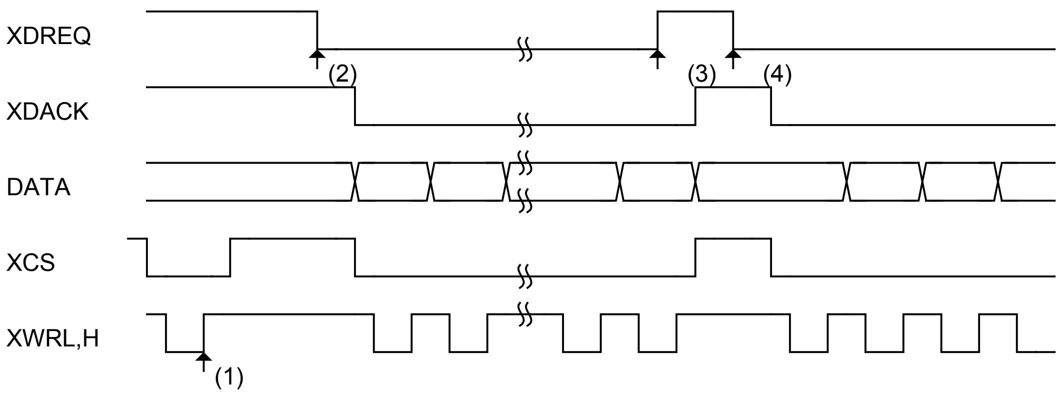
In this mode, XDREQ is temporarily negated every transfer bytes set in the ReqAssertCount [1:0] bits.

Until the operation stops, the DMAx{x=0,1}_Control.DMA_Running bit reads "1."

[Stopping operation]

For condition(s) that stop the operation, refer to the explanation of count mode and free-running mode in the preceding sections.

Example: [Transfer start condition] REQ assert count (8-beat: 16 bytes)



(1) The DMA circuit is activated by writing 1 to the DMA_Control.DMA_Go bit.
Since the DMA_Ready value is less than required for successive transfers, XDREQ is not asserted.
(2) When data is transferred from the USB, etc., and a valid free space greater than required for successive transfers is created in the FIFO (DMA_Ready).
XDREQ is asserted in response to DMA_Ready.
(3) XDREQ is negated in synch upon completion of successive transfers (REQ assert count).
(4) When the first round of successive transfers is complete, a free space for the next successive transfers is available (DMA_Ready).
XDREQ is asserted in response to DMA_Ready.

Fig. 6.88 REQ assert count option write timing

6.7.3.2.8 REQ Assert Count Option (Read)

[Starting operation]

After setting an assert count with the DMAx{x=0,1}_Config.ReqAssertCount [1:0] bits, set the DMAx{x=0,1}_Control_DMA_Go bit to 1. If the internal FIFO has more bytes of valid readable data than the set assert count and is readable from an external device, XDREQ is asserted to enable DMA transfers to be performed. This means that once XDREQ is asserted, transfers for bytes equal to the set assert count are guaranteed. However, even when the data in the FIFO is less than the REQ assert count, if count mode is selected and the said data in the FIFO is greater than the remaining count, then XDREQ is asserted. In this case, the guaranteed transfer bytes equal the remaining count.

In this mode, XDREQ is temporarily negated every transfer bytes set in the ReqAssertCount [1:0] bits.

Until the operation stops, the DMAx{x=0,1}_Control.DMA_Running bit reads "1."

[Stopping operation]

For the condition(s) that stop the operation, refer to the explanation of count mode and free-running mode in the preceding sections.

For the operation timing, refer to Fig. 6.87 and Fig. 6.88.

6.7.3.2.9 FIFO Access Odd Bytes Processing in DMA

Refer to Section 6.7.3.1.5, “Processing Odd Bytes in FIFO Access.” Note that the DMA has no entries for byte read.

6.8 IDE I/F

This section describes the functions of the IDE I/F.

Before the IDE I/F functions can be used, the IDE_Config_1.ActiveIDE bit must be set by writing 1 in software.

6.8.1 Access to the IDE Task File Registers

This section describes the method for accessing the IDE task file registers.

The firmware can access the IDE task file registers via a range of registers from IDE_RegAdr to IDE_RegConfig. Since all read/write sequences on the IDE bus are performed in hardware by the LSI until the operation finishes, the firmware does not need to access the IDE task file registers until it is notified of a termination of the sequence by polling or an interrupt after it sets a command.

When the IDE task file registers are accessed, if both XHCS0 and HDA[2:0] of the IDE bus = 0, the set value of the IDE_Tmod register is used for transfers performed; otherwise, the set value of the IDE_Rmod register is used. Therefore, an appropriate value must be set in the IDE_Tmod or IDE_Rmod register according to IDE transfer mode before transfers start.

6.8.1.1 Read from the IDE Task File Registers

A read from the IDE task file registers can be performed by setting the address to access in the IDE_RegAdrs.IDE_RegAddress[3:0] bits in advance or at the time of access and then writing 1 to the IDE_ReqAdrs.IDE_RdReg bit.

The read operation is complete when the IDE_ReqAdrs.IDE_RdReg bit is cleared to 0 and the IDE_IntStat.IDE_RegCmp bit is set to 1. The data read from the IDE task file registers is stored in the IDE_RdRegValue register.

6.8.1.2 Write to the IDE Task File Registers

A write to the IDE task file registers can be performed by setting the data to be written in the IDE_WrRegValue register in advance, setting the address to access in the IDE_RegAdrs.IDE_RegAddress[3:0] bits in advance or at the time of access, and then writing 1 to the IDE_ReqAdrs.IDE_WrReg bit.

The write operation is complete when the IDE_ReqAdrs.IDE_WrReg bit is cleared to 0 and the IDE_IntStat.IDE_RegCmp bit is set to 1.

6.8.1.3 Sequential Write to the IDE Task File Registers

By using the IDE_SeqWrRegControl register, it is possible to write to the IDE task file registers sequentially for up to a 16 pairs of address and data that have been set in advance.

Set the address at which to write in the IDE_SeqWrRegAdrs.IDE_SeqWrRegAddress[3:0] bits and the byte data to be written in the IDE_SeqWrRegValue register for up to 16 pairs in advance. Since access is made wordwise when XHCS0 = 0 and HDA[2:0] = 0, data need to be written twice in order of the upper and the lower byte, so that two address and data pairs are used. After that, a sequential write operation can be initiated by writing 1 to the IDE_SeqWrReqControl.IDE_SeqWrReg bit. The sequential write operation is complete when the IDE_SeqWrReqControl.IDE_SeqWrReg bit is cleared to 0 and the IDE_IntStat.IDE_SeqWrRegCmp bit is set to 1.

If the address and data that was set in advance becomes unnecessary before a sequential write operation starts, those address and data can be discarded by writing 1 to the IDE_SeqWrReqControl.IDE_SeqWrRegClr bit.

If the firmware performs an ordinary write or read to or from the IDE task file registers during a sequential write operation, the IDE_IntStat.IDE_RegErr bit is set to 1 and the attempted write or read is ignored.

6.8.1.4 Auto Status Register Read from the IDE Task File Registers

If HINTRQ of the IDE bus is set while the IDE_ReqConfig.EnAutoStsRd bit is set by writing 1 in software, the LSI automatically reads the IDE status register (XHCS0 = 0, HDA[2:0] = 7), and stores the read data in the IDE_RdRegValue register. Thereafter, the IDE_IntStat.IDE_CompleteINTRQ bit is set to 1.

If the firmware performs an ordinary write or read to or from the IDE task file registers before it reads the IDE_RdRegValue_1 register after performing an auto status register read operation, the IDE_IntStat.IDE_RegErr bit is set to 1 and the attempted write or read is ignored.

6.8.2 PIO Access

This section describes the DMA functions in PIO mode.

The DMA in PIO mode uses the value set in the IDE_Tmod register for the DMA transfers performed. Therefore, an appropriate value must be set in the IDE_Tmod register according to IDE transfer mode before transfers start.

6.8.2.1 PIO Read DMA

The PIO read DMA operates following the procedure described below.

Write 0 to both the IDE_Config_0.DMA and IDE_Config_0.Ultra bits.

Set the number of transfer bytes in the IDE_CountH/M/L register.

Write 1 to the IDE_Control.IDE_Clr bit to initialize the IDE circuit. (This step may not always be required.)

Write 1 to the IDE_Control.IDE_Go bit to get a DMA operation started. The content of the IDE_CountH/M/L register decrements as the transfer proceeds. When the count reaches 0 and the DMA operation of the IDE and writing the read data from the IDE into the internal FIFO are all complete, the IDE_Control.IDE_Go bit is cleared to 0 and the IDE_IntStat.IDE_Cmp bit is set to 1, causing the DMA operation to end.

If the IDE_Control.IDE_Go bit is cleared by writing 0 while a DMA transfer is underway, the DMA transfer is aborted and the DMA is terminated. In this case, although the data that is transferred into the internal FIFO by that time is valid, the data that has accumulated in the intermediate buffer of the circuit is discarded, so that the number of transferred bytes of the IDE cannot be managed exactly.

6.8.2.2 PIO Write DMA

The PIO write DMA operates following the procedure described below.

Write 0 to both the IDE_Config_0.DMA and IDE_Config_0.Ultra bits.

Set the number of transfer bytes in the IDE_CountH/M/L register.

Write 1 to the IDE_Control.IDE_Clr bit to initialize the IDE circuit. (This step may not always be required.)

Write 1 to the IDE_Control.IDE_Go bit to get a DMA operation started. The content of the IDE_CountH/M/L register decrements as the transfer proceeds. When the count reaches 0 and the DMA write to the IDE are all complete the IDE_Control.IDE_Go bit is cleared to 0 and the IDE_IntStat.IDE_Cmp bit is set to 1, causing the DMA operation to end.

If the IDE_Control.IDE_Go bit is cleared by writing 0 while a DMA transfer is underway, the DMA transfer is aborted and the DMA is terminated. In this case, although the data that is written to the IDE by that time is valid, the data that has accumulated in the intermediate buffer of the circuit is discarded, so that the number of transferred bytes of the IDE cannot be managed exactly.

6.8.3 Multi-Word DMA

This section describes the DMA functions in Multi-Word DMA mode.

The DMA in Multi-Word DMA mode uses the value set in the IDE_Tmod register for the DMA transfers performed. Therefore, an appropriate value must be set in the IDE_Tmod register according to IDE transfer mode before transfers start.

6.8.3.1 Multi-Word DMA Read

The Multi-Word DMA read operates following the procedure described below.

Write 1 to the IDE_Config_0.DMA bit and 0 to the IDE_Config_0.Ultra bit.

Write appropriate values to the IDE_Config_1.DelayStrobe and IDE_Config_1.InterLock bits.

Set the number of transfer bytes in the IDE_CountH/M/L register.

Write 1 to the IDE_Control.IDE_Clr bit to initialize the IDE circuit. (This step may not always be required.)

Write 1 to the IDE_Control.IDE_Go bit to get a DMA operation started. The content of the IDE_CountH/M/L register decrements as the transfer proceeds. When the count reaches 0 and the DMA operation of the IDE and writing the read data from the IDE into the internal FIFO are all complete, the IDE_Control.IDE_Go bit is cleared to 0 and the IDE_IntStat.IDE_Cmp bit is set to 1, causing the DMA operation to end.

If the IDE_Control.IDE_Go bit is cleared by writing 0 while a DMA transfer is underway, the DMA transfer is aborted and the DMA is terminated. In this case, although the data that is transferred to the internal FIFO by that time is valid, the data that has accumulated in the intermediate buffer of the circuit is discarded, so that the number of transferred bytes of the IDE cannot be managed exactly.

6.8.3.2 Multi-Word DMA Write

The Multi-Word DMA write operates following the procedure described below.

Write 1 to the IDE_Config_0.DMA bit and 0 to the IDE_Config_0.Ultra bit.

Write appropriate values to the IDE_Config_1.DelayStrobe and IDE_Config_1.InterLock bits.

Set the number of transfer bytes in the IDE_CountH/M/L register.

Write 1 to the IDE_Control.IDE_Clr bit to initialize the IDE circuit. (This step may not always be required.)

Write 1 to the IDE_Control.IDE_Go bit to get a DMA operation started. The content of the IDE_CountH/M/L register decrements as the transfer proceeds. When the count reaches 0 and the DMA write to the IDE are all complete, the IDE_Control.IDE_Go bit is cleared to 0 and the IDE_IntStat.IDE_Cmp bit is set to 1, causing the DMA operation to end.

If the IDE_Control.IDE_Go bit is cleared by writing 0 while a DMA transfer is underway, the DMA transfer is aborted and the DMA is terminated. In this case, although the data that is written to the IDE by that time is valid, the data that has accumulated in the intermediate buffer of the circuit is discarded, so that the number of transferred bytes of the IDE cannot be managed exactly.

6.8.4 Ultra DMA

This section describes the DMA functions in Ultra DMA mode.

The DMA in Ultra DMA mode uses the value set in the IDE_Umod register for the DMA transfers performed. Therefore, an appropriate value must be set in the IDE_Umod register according to IDE transfer mode before transfers start.

6.8.4.1 Ultra DMA Read

The Ultra DMA read operates following the procedure described below.

Write 1 to the IDE_Config_0.DMA and IDE_Config_0.Ultra bits.

Set the number of transfer bytes in the IDE_CountH/M/L register.

Write 1 to the IDE_Control.IDE_Clr bit to initialize the IDE circuit. (This step may not always be required.)

Write 1 to the IDE_Control.IDE_Go bit to get a DMA operation started. The content of the IDE_CountH/M/L register decrements as the transfer proceeds. When the count reaches 0 and the

DMA operation of the IDE and writing the read data from the IDE into the internal FIFO are all complete, the IDE_Control.IDE_Go bit is cleared to 0 and the IDE_IntStat.IDE_Cmp bit is set to 1, causing the DMA operation to end.

If the IDE_Control.IDE_Go bit is cleared by writing 0 while a DMA transfer is underway, the DMA transfer is aborted and the DMA is terminated. In this case, although the data that is transferred to the internal FIFO by that time is valid, the data that has accumulated in the intermediate buffer of the circuit is discarded, so that the number of transferred bytes of the IDE cannot be managed exactly. In this case, furthermore, the IDE device connected to the system is notified of an event “host terminate.”

If a “device terminate” event occurs during a DMA transfer, the IDE_IntStat.DetectTerm bit is set to 1 and the connected IDE device stops the transfer, but the activated DMA does not end automatically. Therefore, the DMA must be terminated by writing 0 to the IDE_Control.IDE_Go bit.

6.8.4.2 Ultra DMA Write

The Ultra DMA write operates following the procedure described below.

Write 1 to the IDE_Config_0.DMA and IDE_Config_0.Ultra bits.

Set the number of transfer bytes in the IDE_CountH/M/L register.

Write 1 to the IDE_Control.IDE_Clr bit to initialize the IDE circuit. (This step may not always be required.)

Write 1 to the IDE_Control.IDE_Go bit to get a DMA operation started. The content of the IDE_CountH/M/L register decrements as the transfer proceeds. When the count reaches 0 and the DMA write to the IDE are all complete, the IDE_Control.IDE_Go bit is cleared to 0 and the IDE_IntStat.IDE_Cmp bit is set to 1, causing the DMA operation to end.

If the IDE_Control.IDE_Go bit is cleared by writing 0 while a DMA transfer is underway, the DMA transfer is aborted and the DMA is terminated. In this case, although the data that is written to the IDE by that time is valid, the data that has accumulated in the intermediate buffer of the circuit is discarded, so that the number of transferred bytes of the IDE cannot be managed exactly. Further more, in this case the IDE device connected to the system is notified of an event “host terminate.”

If a “device terminate” event occurs during a DMA transfer, the IDE_IntStat.DetectTerm bit is set to 1 and the connected IDE device stops the transfer, but the activated DMA does not end automatically. Therefore, the DMA must be terminated by writing 0 to the IDE_Control.IDE_Go bit.

6.8.5 IDE Transfer Mode Settings

The tables below list the register values to be set for each mode of IDE transfer.

- Register Mode (For access to the IDE task file registers by firmware, when other than XHCS = 0 and HDA = 0)

Mode	Register set value (required)		Register set value (recommended)
	IDE_Config_0.Ultra	IDE_Config_0.DMA	IDE_Rmod
0	No effect	No effect	FFh
1	No effect	No effect	F1h
2	No effect	No effect	F0h
3	No effect	No effect	22h
4	No effect	No effect	10h

- PIO Mode (For access to the IDE task file registers by firmware, when XHCS = 0 and HDA = 0)

Mode	Register set value (required)		Register set value (recommended)
	IDE_Config_0.Ultra	IDE_Config_0.DMA	IDE_Tmod
0	No effect	No effect	FFh
1	No effect	No effect	88h
2	No effect	No effect	44h
3	No effect	No effect	22h
4	No effect	No effect	10h

- PIO Mode (For DMA transfer)

Mode	Register set value (required)		Register set value (recommended)
	IDE_Config_0.Ultra	IDE_Config_0.DMA	IDE_Tmod
0	0	0	FFh
1	0	0	88h
2	0	0	44h
3	0	0	22h
4	0	0	10h

- Multi-word DMA Mode (For DMA transfer)

Mode	Register set value (required)		Register set value (recommended)
	IDE_Config_0.Ultra	IDE_Config_0.DMA	IDE_Tmod
0	0	1	BBh
1	0	1	20h
2	0	1	10h

- Ultra Mode (For DMA transfer, during DATA-OUT)

Mode	Register set value (required)		Register set value (recommended)
	IDE_Config_0.Ultra	IDE_Config_0.DMA	IDE_Umod
0	1	1	06h
1	1	1	04h
2	1	1	03h
3	1	1	02h
4	1	1	01h
5	1	1	00h

Note: During DATA-IN in Ultra mode where data is input from the IDE bus, data can be received in any modes regardless of how the IDE_Umode register is set.

6.9 Boundary Scan (JTAG)

Boundary Scan (JTAG) can be used when the TEST pin is held low (default). The boundary scan consists of the BSR (Boundary Scan Register) compliant with the JTAG (IEEE1149.1) specification, a scan path to connect it, and a TAP controller. Information on boundary scan connections will be supplied to users in BSDL format.

6.9.1 Supported Instructions

The JTAG instructions supported by the LSI stipulated herein are 4 bits in width. These correspond to the JTAG instructions listed in the table below.

Table 6.64 JTAG Instruction Codes

Instruction	Description	Code
SAMPLE/PRELOAD	Latches internal LSI status into BSR and sets data	0010
BYPASS	Bypasses scan path configured by BSR	1111
EXTEST	Checks physical device connection	0000
CLAMP	Bypasses scan path while retaining output value	0011
HIGHZ	Fixes all outputs to the Hi-Z state	0100
IDCODE	Outputs designated DEVICE_CODE	0001

6.9.2 Regarding DEVICE_CODE

The elements that comprise DEVICE_CODE for the IDCODE instruction are listed in the table below.

Table 6.65 DEVICE_CODE

Version	1
Part Number	000F
Manufacturer	0x0BE

Therefore, the DEVICE_CODE response for the IDCODE instruction is as follows:

0001_0000000000001111_00010111110_1

6.9.3 Boundary Scan Exclusion Pins

Of the pins mounted on the LSI stipulated herein, DP_A, DM_A, DP_B, DM_B, XI, XO, VBUS_B, R1_A, R1_B and TEST are excluded from the scan because they do not have boundary-scan cells inserted.

7. Registers

The registers in the S1R72V05 are classified into three groups: device/host shared registers, device registers, and host registers. The register maps between the device and host registers are switched over by the HostDeviceSel.HOSTxDEVICE bit. The device register map is selected when this bit = 0, and the host register map is selected when this bit = 1. The set values in either register map are not cleared by changing the setting of this bit.

Do not write 1 to the reserved register bits.

7.1 Device/Host Shared Register Map

The registers that can be both read and written even during Sleep/Snooze are shown in ***bold face italic***.

Note: The registers at addresses 0x40 to 0x47 and 0x50 to 0x6F cannot be read from or written to in the Active60 state.

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x00	***MainIntStat***	R/(W)	0x00	***DeviceIntStat***	***HostIntStat***	CPU_IntStat	IDE_IntStat	MediaFIFO_Intstat			***FinishedPM***
0x01	***DeviceIntStat***	R/(W)	0x00	***VBUS_Changed***		***D_SIE_IntStat***	D_BulkIntStat	RcvEP0SETUP	D_FIFO_IntStat	D_EP0IntStat	D_EPrIntStat
0x02	***HostIntStat***	R/(W)	0x00	***VBUS_Err***	***LineStateChanged***	H_SIE_IntStat_1	H_SIE_IntStat_0	H_FrameIntStat	H_FIFO_IntStat	H_CH0IntStat	H_CHrIntStat
0x03	CPU_IntStat	R/(W)	0x00	RAM_RdCmp				DMA1_Countup	DMA1_Cmp	DMA0_Countup	DMA0_Cmp
0x04	IDE_IntStat	R/(W)	0x00	IDE_RegCmp	IDE_RegErr	IDE_SeqWrRegCmp	CompleteINTRQ		IDE_Cmp	DetectINTRQ	DetectTerm
0x05	MediaFIFO_IntStat	R/(W)	0x00		MediaIDE_Cmp				FIFO_NotEmpty	FIFO_Full	FIFO_Empty
0x06			0xXX								
0x07			0xXX								
0x08			0xXX								
0x09			0xXX								
0x0A			0xXX								
0x0B			0xXX								
0x0C			0xXX								
0x0D			0xXX								
0x0E			0xXX								
0x0F			0xXX								

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x10	***MainIntEnb***	R/W	0x00	***EnDeviceIntStat***	***EnHostIntStat***	EnCPU_IntStat	EnIDE_IntStat	EnMediaFIFO_IntStat			***EnFinishedPM***
0x11	***DeviceIntEnb***	R/W	0x00	***EnVBUS_Changed***		***EnD_SIE_IntStat***	EnD_BulkIntStat	EnRcvEP0SETUP	EnD_FIFO_IntStat	EnD_EP0IntStat	EnD_EPrIntStat
0x12	***HostIntEnb***	R/W	0x00	***EnVBUS_Err***	***EnLineStateChanged***	EnH_SIE_IntStat_1	EnH_SIE_IntStat_0	EnH_FrameIntStat	EnH_FIFO_IntStat	EnH_CH0IntStat	EnH_CHrIntStat
0x13	CPU_IntEnb	R/W	0x00	EnRAM_RdCmp				EnDMA1_Countup	EnDMA1_Cmp	EnDMA0_Countup	EnDMA0_Cmp
0x14	IDE_IntEnb	R/W	0x00	EnIDE_RegCmp	EnIDE_RegErr	EnIDE_SeqWrRegCmp	EnComplete INTRQ		EnIDE_Cmp	EnDetectINTRQ	EnDetectTerm
0x15	MediaFIFOIntEnb	R/W	0x00		EnMediaIDE_Cmp				FIFO_NotEmpty	FIFO_Full	FIFO_Empty
0x16			0xXX								
0x17			0xXX								
0x18			0xXX								
0x19			0xXX								
0x1A			0xXX								
0x1B			0xXX								
0x1C			0xXX								
0x1D			0xXX								
0x1E			0xXX								
0x1F			0xXX								

The registers that can be both read and written even during Sleep/Snooze are shown in ***bold face italic***.

The other registers can be read and written to during Active60, Act_Device, and Act_Host.
Note: The registers at addresses 0x40 to 0x47 and 0x50 to 0x6F cannot be read from or written to in the Active60 state.

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x20	***RevisionNum***	R	0x50	***RevisionNumer***							
0x21	***ChipReset***	R/W	0x00								***AllReset***
0x22	***PM_Control_0***	R/W	0x00	***GoSLEEP***	***GoSNOOZE***	***GoActive60***	***GoActDevice***	***GoActHost***			
0x23	***PM_Control_1***	R	0x00					PM_State[3:0]			
0x24	***WakeupTim_H***	R/W	0x00	WakeupTim [15:8]							
0x25	***WakeupTim_L***	R/W	0x00	WakeupTim [7:0]							
0x26	***H_USB_Control***	R/W	0x00	***VBUS_Enb***							
0x27	***H_XcvrControl***	R/W	0x11	***TermSelect***	***RemoveRPD***	***XcvrSelect[1:0]***				***OpMode[1:0]***	
0x28	***D_USB_Status***	R/W	0xXX	***VBUS***	***FSxHS***					***LineState[1:0]***	
0x29	***H_USB_Status***	R	0xXX	VBUS_State						***LineState[1:0]***	
0x2A			0xXX								
0x2B			0xXX								
0x2C			0xXX								
0x2D			0xXX								
0x2E			0xXX								
0x2F			0xXX								

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x30	FIFO_Rd_0	R	0xXX	FIFO_Rd_0[7:0]							
0x31	FIFO_Rd_1	R	0xXX	FIFO_Rd_1[7:0]							
0x32	FIFO_Wr_0	W	0xXX	FIFO_Wr_0[7:0]							
0x33	FIFO_Wr_1	W	0xXX	FIFO_Wr_1[7:0]							
0x34	FIFO_RdRemain_H	R	0x00	RdRemainValid			RdRemain[12:8]				
0x35	FIFO_RdRemain_L	R	0x00	RdRemain[7:0]							
0x36	FIFO_WrRemain_H	R	0x00				WrRemain[12:8]				
0x37	FIFO_WrRemain_L	R	0x00	WrRemain[7:0]							
0x38	FIFO_ByteRd	R	0xXX	FIFO_ByteRd[7:0]							
0x39											
0x3A											
0x3B											
0x3C											
0x3D											
0x3E											
0x3F											

The registers that can be both read and written even during Sleep/Snooze are shown in ***bold face italic***.

The other registers can be read and written to during Active60, Act_Device, and Act_Host.
Note: The registers at addresses 0x40 to 0x47 and 0x50 to 0x6F cannot be read from or written to in the Active60 state.

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x40	RAM_RdAdrs_H	R/W	0x00					RAM_RdAdrs[12:8]			
0x41	RAM_RdAdrs_L	R/W	0x00	RAM_RdAdrs[7:2]							
0x42	RAM_RdControl	R/W	0x00	RAM_GoRdCBW_CSW	RAM_GoRd						
0x43	RAM_RdCount	R/W	0x00			RAM_RdCount[5:2]					
0x44	RAM_WrAdrs_H	R/W	0x00					RAM_Adrs[12:8]			
0x45	RAM_WrAdrs_L	R/W	0x00	RAM_Adrs[7:0]							
0x46	RAM_WrDoor_0	W	0xXX	RAM_Door_0[7:0]							
0x47	RAM_WrDoor_1	W	0xXX	RAM_Door_1[7:0]							
0x48	MediaFIFO_Control	W	0xXX								MediaFIFO_Clr
0x49			0xXX								
0x4A	MediaFIFO_Join	R/W	0x00	JoinIDE				JoinDMA1	JoinDMA0	JoinCPU_Rd	JoinCPU_Wr
0x4B	ClrAllMediaFIFO_Join	W	0xXX	ClrJoinIDE				ClrJoinDMA1	ClrJoinDMA0	ClrJoinCPU_Rd	ClrJoinCPU_Wr
0x4C			0xXX								
0x4D			0xXX								
0x4E			0xXX								
0x4F			0xXX								

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x50	RAM_Rd_00	R	0x00	RAM_Rd_00[7:0]							
0x51	RAM_Rd_01	R	0x00	RAM_Rd_01[7:0]							
0x52	RAM_Rd_02	R	0x00	RAM_Rd_02[7:0]							
0x53	RAM_Rd_03	R	0x00	RAM_Rd_03[7:0]							
0x54	RAM_Rd_04	R	0x00	RAM_Rd_04[7:0]							
0x55	RAM_Rd_05	R	0x00	RAM_Rd_05[7:0]							
0x56	RAM_Rd_06	R	0x00	RAM_Rd_06[7:0]							
0x57	RAM_Rd_07	R	0x00	RAM_Rd_07[7:0]							
0x58	RAM_Rd_08	R	0x00	RAM_Rd_08[7:0]							
0x59	RAM_Rd_09	R	0x00	RAM_Rd_09[7:0]							
0x5A	RAM_Rd_0A	R	0x00	RAM_Rd_0A[7:0]							
0x5B	RAM_Rd_0B	R	0x00	RAM_Rd_0B[7:0]							
0x5C	RAM_Rd_0C	R	0x00	RAM_Rd_0C[7:0]							
0x5D	RAM_Rd_0D	R	0x00	RAM_Rd_0D[7:0]							
0x5E	RAM_Rd_0E	R	0x00	RAM_Rd_0E[7:0]							
0x5F	RAM_Rd_0F	R	0x00	RAM_Rd_0F[7:0]							

The registers that can be both read and written even during Sleep/Snooze are shown in ***bold face italic***.

The other registers can be read and written to during Active60, Act_Device, and Act_Host.
Note: The registers at addresses 0x40 to 0x47 and 0x50 to 0x6F cannot be read from or written to in the Active60 state.

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x60	RAM_Rd_10	R	0x00	RAM_Rd_10[7:0]							
0x61	RAM_Rd_11	R	0x00	RAM_Rd_11[7:0]							
0x62	RAM_Rd_12	R	0x00	RAM_Rd_12[7:0]							
0x63	RAM_Rd_13	R	0x00	RAM_Rd_13[7:0]							
0x64	RAM_Rd_14	R	0x00	RAM_Rd_14[7:0]							
0x65	RAM_Rd_15	R	0x00	RAM_Rd_15[7:0]							
0x66	RAM_Rd_16	R	0x00	RAM_Rd_16[7:0]							
0x67	RAM_Rd_17	R	0x00	RAM_Rd_17[7:0]							
0x68	RAM_Rd_18	R	0x00	RAM_Rd_18[7:0]							
0x69	RAM_Rd_19	R	0x00	RAM_Rd_19[7:0]							
0x6A	RAM_Rd_1A	R	0x00	RAM_Rd_1A[7:0]							
0x6B	RAM_Rd_1B	R	0x00	RAM_Rd_1B[7:0]							
0x6C	RAM_Rd_1C	R	0x00	RAM_Rd_1C[7:0]							
0x6D	RAM_Rd_1D	R	0x00	RAM_Rd_1D[7:0]							
0x6E	RAM_Rd_1E	R	0x00	RAM_Rd_1E[7:0]							
0x6F	RAM_Rd_1F	R	0x00	RAM_Rd_1F[7:0]							

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x70			0xXX								
0x71	DMA0_Config	R/W	0x00	FreeRun	DMA_Mode			ActiveDMA		ReqAssertCount [1:0]	
0x72	DMA0_Control	R/W	0x00	DMA_Running			CounterClr	Dir		DMA_Stop	DMA_Go
0x73			0xXX								
0x74	DMA0_Remain_H	R	0x00				DMA_Remain [12:8]				
0x75	DMA0_Remain_L	R	0x00	DMA_Remain [7:0]							
0x76			0xXX								
0x77			0xXX								
0x78	DMA0_Count_HH	R/W	0x00	DMA_Count [31:24]							
0x79	DMA0_Count_HL	R/W	0x00	DMA_Count [23:16]							
0x7A	DMA0_Count_LH	R/W	0x00	DMA_Count [15:8]							
0x7B	DMA0_Count_LL	R/W	0x00	DMA_Count [7:0]							
0x7C	DMA0_RdData_0	R	0xXX	DMA_RdData_0[7:0]							
0x7D	DMA0_RdData_1	R	0xXX	DMA_RdData_1[7:0]							
0x7E	DMA0_WrData_0	W	0xXX	DMA_WrData_0[7:0]							
0x7F	DMA0_WrData_1	W	0xXX	DMA_WrData_1[7:0]							

The registers that can be both read and written even during Sleep/Snooze are shown in ***bold face italic***.

The other registers can be read and written to during Active60, Act_Device, and Act_Host.
Note: The registers at addresses 0x40 to 0x47 and 0x50 to 0x6F cannot be read from or written to in the Active60 state.

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x80			0xXX								
0x81	DMA1_Config	R/W	0x00	FreeRun	DMA_Mode			ActiveDMA		ReqAssertCount [1:0]	
0x82	DMA1_Control	R/W	0x00	DMA_Running			CounterClr	Dir		DMA_Stop	DMA_Go
0x83			0xXX								
0x84	DMA1_Remain_H	R	0x00				DMA_Remain [12:8]				
0x85	DMA1_Remain_L	R	0x00	DMA_Remain [7:0]							
0x86			0xXX								
0x87			0xXX								
0x88	DMA1_Count_HH	R/W	0x00	DMA_Count [31:24]							
0x89	DMA1_Count_HL	R/W	0x00	DMA_Count [23:16]							
0x8A	DMA1_Count_LH	R/W	0x00	DMA_Count [15:8]							
0x8B	DMA1_Count_LL	R/W	0x00	DMA_Count [7:0]							
0x8C	DMA1_RdData_0	R	0xXX	DMA_RdData[15:8]							
0x8D	DMA1_RdData_1	R	0xXX	DMA_RdData[7:0]							
0x8E	DMA1_WrData_0	W	0xXX	DMA_WrData[15:8]							
0x8F	DMA1_WrData_1	W	0xXX	DMA_WrData[7:0]							

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x90	IDE_Status	R	0x00	DMARQ	DMACK	INTRQ	IORDY			PDIAG	DASP
0x91	IDE_Control	R/W	0x00		IDE_Clr			Dir			IDE_Go
0x92	IDE_Comfig_0	R/W	0x00	IDE_BusReset	IDE_LongBusReset					Ultra	DMA
0x93	IDE_Config_1	R/W	0x04	ActiveIDE	DelayStrobe		InterLock		Swap		
0x94	IDE_Rmod	R/W	0x00	RegisterAssertPulseWidth[3:0]				RegisterNegatePulseWidth[3:0]			
0x95	IDE_Tmod	R/W	0x00	TransferAssertPulseWidth[3:0]				TransferNegatePulseWidth[3:0]			
0x96	IDE_Umod	R/W	0x00					UltraDMA_Cycle[3:0]			
0x97			0xXX								
0x98			0xXX								
0x99			0xXX								
0x9A	IDE_CRC_H	R/W	0x00	IDE_CRC[15:8]							
0x9B	IDE_CRC_L	R/W	0x00	IDE_CRC[7:0]							
0x9C			0xXX								
0x9D	IDE_Count_H	R/W	0x00	IDE_Count[23:16]							
0x9E	IDE_Count_M	R/W	0x00	IDE_Count[15:8]							
0x9F	IDE_Count_L	R/W	0x00	IDE_Count[7:1]							

The registers that can be both read and written even during Sleep/Snooze are shown in ***bold face italic***.

The other registers can be read and written to during Active60, Act_Device, and Act_Host.
Note: The registers at addresses 0x40 to 0x47 and 0x50 to 0x6F cannot be read from or written to in the Active60 state.

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0xA0	IDE_RegAdrs	R/W	0x00	IDE_WrReg	IDE_RdReg			IDE_RegAddress[3:0]			
0xA1			0xXX								
0xA2	IDE_RdRegValue_0	R	0x00	IDE_RdRegValue_0[7:0]							
0xA3	IDE_RdRegValue_1	R	0x00	IDE_RdRegValue_1[7:0]							
0xA4	IDE_WrRegValue_0	R/W	0x00	IDE_WrRegValue_0[7:0]							
0xA5	IDE_WrRegValue_1	R/W	0x00	IDE_WrRegValue_1[7:0]							
0xA6	IDE_SeqWrRegControl	R/W	0x00	IDE_SeqWrReg	IDE_SeqWrRegClr						
0xA7	IDE_SeqWrRegCnt	R	0x00				IDE_SeqWrRegCnt[4:0]				
0xA8	IDE_SeqWrRegAdrs	W	0xXX					IDE_SeqRegAddress[3:0]			
0xA9	IDE_SeqWrRegValue	W	0xXX	IDE_SeqWrRegValue[7:0]							
0xAA			0xXX								
0xAB			0xXX								
0xAC	IDE_RegConfig	R/W	0x00	EnAutoStsRd							
0xAD			0xXX								
0xAE			0xXX								
0xAF			0xXX								

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0xB0			0xXX								
0xB1	***HostDeviceSel***	R/W	0x00								***HOSTxDEVICE***
0xB2			0xXX								
0xB3	***ModeProtect***	R/W	0x56	***ModeProtect[7:0] (writing other than 56 enables protect; 0x56 disables protect)***							
0xB4			0xXX								
0xB5	***ClkSelect***	R/W	0x41	xActIDE_Term	***XActIDE_DD_Term***					***Port1x2***	***ClkSelect***
0xB6			0xXX								
0xB7	***ChipConfig***	R/W	0x00	***IntLevel***	***IntMode***	***DREQ_Level***	***DACK_Level***	***CS_Mode***	***CPU_Endian***	***BusMode***	***Bus8x16***
0xB8			0xXX								
0xB9	***CPU_ChgEndian***	R	0xXX	***The endian set by ChipConfig.CPU_Endian is enabled by accessing this register for a dummy read.***							
0xBA			0xXX								
0xBB			0xXX								
0xBC			0xXX								
0xBD			0xXX								
0xBE			0xXX								
0xBF			0xXX								

0xC0–0xDF are reserved.

The registers that can be both read and written even during Sleep/Snooze are shown in ***bold face italic***.

The other registers can be read during Active60, Act_Device, and Act_Host, and can be written to during Act_Device.

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
0xE0	***D_SIE_IntStat***	R/(W)	0x00		***NonJ***	RcvSOF	DetectRESET	DetectSUSPEND	ChirpCmp	RestoreCmp	SetAddressCmp
0xE1			0xXX								
0xE2	D_FIFO_IntSat	R/(W)	0x00	DescriptorCmp	FIFO_IDE_Cmp	FIFO1_Cmp	FIFO0_Cmp		FIFO_NotEmpty	FIFO_Full	FIFO_Empty
0xE3	D_BulkIntStat	R/(W)	0x00	CBW_Cmp	CBW_LengthErr	CBW_Err		CSW_Cmp	CSW_Err		
0xE4	D_EPrIntStat	R	0x00						EPcIntStat	EPbIntStat	EPaIntStat
0xE5	D_EP0IntStat	R/(W)	0x00		OUT_ShortACK	IN_TranACK	OUT_TranACK	IN_TranNAK	OUT_TranNAK	IN_TranErr	OUT_TranErr
0xE6	D_EPaIntStat	R/(W)	0x00		OUT_ShortACK	IN_TranACK	OUT_TranACK	IN_TranNAK	OUT_TranNAK	IN_TranErr	OUT_TranErr
0xE7	D_EPbIntStat	R/(W)	0x00		OUT_ShortACK	IN_TranACK	OUT_TranACK	IN_TranNAK	OUT_TranNAK	IN_TranErr	OUT_TranErr
0xE8	D_EPcIntStat	R/(W)	0x00		OUT_ShortACK	IN_TranACK	OUT_TranACK	IN_TranNAK	OUT_TranNAK	IN_TranErr	OUT_TranErr
0xE9			0xXX								
0xEA			0xXX								
0xEB			0xXX								
0xEC			0xXX								
0xED			0xXX								
0xEE			0xXX								
0xEF			0xXX								

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
0xF0	***D_SIE_IntEnb***	R/W	0x00		***EnNonJ***	EnRcvSOF	EnDetectRESET	EnDetectSUSPEND	EnChirpCmp	EnRestoreCmp	EnSetAddressCmp
0xF1		R/W	0x00								
0xF2	D_FIFO_IntEnb	R/W	0x00	EnDescriptorCmp	EnFIFO_IDE_Cmp	EnFIFO1_Cmp	EnFIFO0_Cmp		EnFIFO_NotEmpty	EnFIFO_Full	EnFIFO_Empty
0xF3	D_BulkIntEnb	R/W	0x00	EnCBW_Cmp	EnCBW_LengthErr	EnCBW_Err		EnCSW_Cmp	EnCSW_Err		
0xF4	D_EPrIntEnb	R/W	0x00						EnEPcIntStat	EnEPbIntStat	EnEPaIntStat
0xF5	D_EP0IntEnb	R/W	0x00		EnOUT_ShortACK	EnIN_TranACK	EnOUT_TranACK	EnIN_TranNAK	EnOUT_TranNAK	EnIN_TranErr	EnOUT_TranErr
0xF6	D_EPaIntEnb	R/W	0x00		EnOUT_ShortACK	EnIN_TranACK	EnOUT_TranACK	EnIN_TranNAK	EnOUT_TranNAK	EnIN_TranErr	EnOUT_TranErr
0xF7	D_EPbIntEnb	R/W	0x00		EnOUT_ShortACK	EnIN_TranACK	EnOUT_TranACK	EnIN_TranNAK	EnOUT_TranNAK	EnIN_TranErr	EnOUT_TranErr
0xF8	D_EPcIntEnb	R/W	0x00		EnOUT_ShortACK	EnIN_TranACK	EnOUT_TranACK	EnIN_TranNAK	EnOUT_TranNAK	EnIN_TranErr	EnOUT_TranErr
0xF9			0xXX								
0xFA			0xXX								
0xFB			0xXX								
0xFC			0xXX								
0xFD			0xXX								
0xFE			0xXX								
0xFF			0xXX								

7.2 Device Register Map

The registers that can be both read and written even during Sleep/Snooze are shown in ***bold face italic***.

The other registers can be read during Active60, Act_Device, and Act_Host, and can be written to during Act_Device.

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x100	***D_Reset***	R/W	0x00								***ResetDTM***
0x101			0xXX								
0x102	D_NegoControl	R/W	0x00	DisBusDetect	EnAutoNego	InSUSPEND	DisableHS	SendWakeup	RestoreUSB	GoChirp	ActiveUSB
0x103			0xXX								
0x104	D_ClrAllEPnJoin	W	0xXX	ClrJoinIDE	ClrJoinFIFO_Stat			ClrJoinDMA1	ClrJoinDMA0	ClrJoinCPU_Rd	ClrJoinCPU_Wr
0x105	D_XcvrControl	R/W	0x41	TermSelect	XcvrSelect					OpMode [1:0]	
0x106	D_USB_Test	R/W	0x00	EnHS_Test				Test_SE0_NAK	Test_J	Test_K	Test_Packet
0x107			0xXX								
0x108	D_EPnControl	W	0xXX	AllForceNAK	EPrForceSTALL	AllFIFO_Clr					EP0FIFO_Clr
0x109	D_EPrFIFO_Clr	W	0xXX						EPcFIFO_Clr	EPbFIFO_Clr	EPaFIFO_Clr
0x10A	D_BulkOnlyControl	R/W	0x00	AutoForceNAK_CBW					GoCBW_Mode	GoCSW_Mode	
0x10B	D_BulkOnlyConfig	R/W	0x00						EPcBulkOnly	EPbBulkOnly	
0x10C			0xXX								
0x10D			0xXX								
0x10E			0xXX								
0x10F			0xXX								

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x110	D_EP0SETUP_0	R	0x00	SETUP 0							
0x111	D_EP0SETUP_1	R	0x00	SETUP 1							
0x112	D_EP0SETUP_2	R	0x00	SETUP 2							
0x113	D_EP0SETUP_3	R	0x00	SETUP 3							
0x114	D_EP0SETUP_4	R	0x00	SETUP 4							
0x115	D_EP0SETUP_5	R	0x00	SETUP 5							
0x116	D_EP0SETUP_6	R	0x00	SETUP 6							
0x117	D_EP0SETUP_7	R	0x00	SETUP 7							
0x118	D_USB_Address	R(/W)	0x00	SetAddress	USB_Address [6:0]						
0x119			0xXX								
0x11A	D_SETUP_Control	R/W	0x00								ProtectEP0
0x11B			0xXX								
0x11C			0xXX								
0x11D			0xXX								
0x11E	D_FrameNumber_H	R	0x80	FN_Invalid					FrameNumber [10:8]		
0x11F	D_FrameNumber_L	R	0x00	FrameNumber [7:0]							

The registers that can be both read and written even during Sleep/Snooze are shown in ***bold face italic***.

The other registers can be read during Active60, Act_Device, and Act_Host, and can be written to during Act_Device.

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x120	D_EP0MaxSize	R/W	0x40		EP0MaxSize[6:3]						
0x121	D_EP0Control	R/W	0x00	INxOUT							ReplyDescriptor
0x122	D_EP0ControlIN	R/W	0x00		EnShortPkt		ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x123	D_EP0ControlOUT	R/W	0x00	AutoForceNAK			ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x124			0xXX								
0x125	D_EP0Join	R/W	0x00		JoinFIFO_Stat			JoinDMA1	JoinDMA0	JoinCPU_Rd	JoinCPU_Wr
0x126			0xXX								
0x127			0xXX								
0x128			0xXX								
0x129			0xXX								
0x12A			0xXX								
0x12B			0xXX								
0x12C			0xXX								
0x12D			0xXX								
0x12E			0xXX								
0x12F			0xXX								

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x130	D_EPaMaxSize_H	R/W	0x00							EPaMaxSize[9:8]	
0x131	D_EPaMaxSize_L	R/W	0x00	EPaMaxSize [7:3]							
0x132	D_EPaConfig_0	R/W	0x00	INxOUT	IntEP_Mode	EnEndpoint		EndpointNumber[3:0]			
0x133			0xXX								
0x134	D_EPaControl	R/W	0x00	AutoForceNAK	EnShortPkt	DisAF_NAK_Short	ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x135	D_EPaJoin	R/W	0x00	JoinIDE	JoinFIFO_Stat			JoinDMA1	JoinDMA0	JoinCPU_Rd	JoinCPU_Wr
0x136			0xXX								
0x137			0xXX								
0x138			0xXX								
0x139			0xXX								
0x13A			0xXX								
0x13B			0xXX								
0x13C			0xXX								
0x13D			0xXX								
0x13E			0xXX								
0x13F			0xXX								

The registers that can be both read and written even during Sleep/Snooze are shown in ***bold face italic***.

The other registers can be read during Active60, Act_Device, and Act_Host, and can be written to during Act_Device.

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x140	D_EPbMaxSize_H	R/W	0x00							EPbMaxSize[9:8]	
0x141	D_EPbMaxSize_L	R/W	0x00	EPbMaxSize[7:3]							
0x142	D_EPbConfig_0	R/W	0x00	INxOUT	IntEP_Mode	EnEndpoint		EndpointNumber[3:0]			
0x143			0xXX								
0x144	D_EPbControl	R/W	0x00	AutoForceNAK	EnShortPkt	DisAF_NAK_Short	ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x145	D_EPbJoin	R/W	0x00	JoinIDE	JoinFIFO_Stat			JoinDMA1	JoinDMA0	JoinCPU_Rd	JoinCPU_Wr
0x146			0xXX								
0x147			0xXX								
0x148			0xXX								
0x149			0xXX								
0x14A			0xXX								
0x14B			0xXX								
0x14C			0xXX								
0x14D			0xXX								
0x14E			0xXX								
0x14F			0xXX								

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x150	D_EPcMaxSize_H	R/W	0x00							EPcMaxSize[9:8]	
0x151	D_EPcMaxSize_L	R/W	0x00	EPcMaxSize[7:3]							
0x152	D_EPcConfig_0	R/W	0x00	INxOUT	IntEP_Mode	EnEndpoint		EndpointNumber[3:0]			
0x153			0xXX								
0x154	D_EPcControl	R/W	0x00	AutoForceNAK	EnShortPkt	DisAF_NAK_Short	ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x155	D_EPcJoin	R/W	0x00	JoinIDE	JoinFIFO_Stat			JoinDMA1	JoinDMA0	JoinCPU_Rd	JoinCPU_Wr
0x156			0xXX								
0x157			0xXX								
0x158			0xXX								
0x159			0xXX								
0x15A			0xXX								
0x15B			0xXX								
0x15C			0xXX								
0x15D			0xXX								
0x15E			0xXX								
0x15F			0xXX								

The registers that can be both read and written even during Sleep/Snooze are shown in ***bold face italic***.

The other registers can be read during Active60, Act_Device, and Act_Host, and can be written to during Act_Device.

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x160	D_DescAdrs_H	R/W	0x00					DescAdrs[11:8]			
0x161	D_DescAdrs_L	R/W	0x00	DescAdrs [7:0]							
0x162	D_DescSize_H	R/W	0x00					DescSize [9:8]			
0x163	D_DescSize_L	R/W	0x00	DescSize [7:0]							
0x164			0xXX								
0x165			0xXX								
0x166			0xXX								
0x167			0xXX								
0x168			0xXX								
0x169			0xXX								
0x16A			0xXX								
0x16B			0xXX								
0x16C			0xXX								
0x16D			0xXX								
0x16E			0xXX								
0x16F			0xXX								

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x170	DMA0_FIFO_Control	R/W	0x00	FIFO_Running	AutoEnShort						
0x171			0xXX								
0x172	DMA1_FIFO_Control	R/W	0x00	FIFO_Running	AutoEnShort						
0x173			0xXX								
0x174			0xXX								
0x175			0xXX								
0x176			0xXX								
0x177			0xXX								
0x178			0xXX								
0x179			0xXX								
0x17A			0xXX								
0x17B			0xXX								
0x17C			0xXX								
0x17D			0xXX								
0x17E			0xXX								
0x17F			0xXX								

The registers that can be both read and written even during Sleep/Snooze are shown in ***bold face italic***.

The other registers can be read during Active60, Act_Device, and Act_Host, and can be written to during Act_Device.

Byte Addr.	Register Name	R/W	Reset	bit15 / bit7	bit14 / bit6	bit13 / bit5	bit12 / bit4	bit11 / bit3	bit10 / bit2	bit9 / bit1	bit8 / bit0
0x180			0xXX								
0x181			0xXX								
0x182			0xXX								
0x183			0xXX								
0x184	D_EPaStartAdrs_H	R/W	0x00				StartAdrs[12:8]				
0x185	D_EPaStartAdrs_L	R/W	0x00	StartAdrs[7:2]							
0x186			0xXX								
0x187			0xXX								
0x188	D_EPbStartAdrs_H	R/W	0x00				StartAdrs[12:8]				
0x189	D_EPbStartAdrs_L	R/W	0x00	StartAdrs[7:2]							
0x18A			0xXX								
0x18B			0xXX								
0x18C	D_EPcStartAdrs_H	R/W	0x00				StartAdrs[12:8]				
0x18D	D_EPcStartAdrs_L	R/W	0x00	StartAdrs[7:2]							
0x18E	D_EPcEndAdrs_H	R/W	0x00				EndAdrs[12:8]				
0x18F	D_EPcEndAdrs_L	R/W	0x00	EndAdrs[7:2]							

0x190–0x1DF are reserved.

For detailed information on the registers listed below, refer to Appendix D.

Byte Addr.	Register Name	R/W	Reset	bit15 / bit7	bit14 / bit6	bit13 / bit5	bit12 / bit4	bit11 / bit3	bit10 / bit2	bit9 / bit1	bit8 / bit0
0x1E0	(Reserved)		0xXX								
0x1E1	D_ModeControl	W	0xXX	(Reserved)	(Reserved)	(Reserved)	SetAddressMode	(Reserved)	(Reserved)	(Reserved)	(Reserved)

0x1E2–0x1FF are reserved.

7.3. Host Register Map

The registers that can be both read and written even during Sleep/Snooze are shown in ***bold face italic***.

The other registers can be read during Active60, Act_Device, and Act_Host, and can be written to during Act_Device.

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0xE0	H_SIE_IntStat_0	R/(W)	0x00				DetectCon	DetectDiscon	DetectRmtWkup	DetectDevChirpOK	DetectDevChirpNG
0xE1	H_SIE_IntStat_1	R/(W)	0x00					DisabledCmp	ResumeCmp	SuspendCmp	ResetCmp
0xE2	H_FIFO_IntSat	R/(W)	0x00		FIFO_IDE_Cmp	FIFO1_Cmp	FIFO0_Cmp		FIFO_NotEmpty	FIFO_Full	FIFO_Empty
0xE3	H_FrameIntStat	R/(W)	0x00						PortErr	FrameNumOver	SOF
0xE4	H_CHrIntStat	R	0x00				H_CHeIntStat	H_CHdIntStat	H_CHcIntStat	H_CHbIntStat	H_CHaIntStat
0xE5	H_CH0IntStat	R/(W)	0x00	TotalSizeCmp	TranACK	TranErr	ChangeCondition			CTL_SupportCmp	CTL_SupportStop
0xE6	H_CHaIntStat	R/(W)	0x00	TotalSizeCmp	TranACK	TranErr	ChangeCondition			BO_SupportCmp	BO_SupportStop
0xE7	H_CHbIntStat	R/(W)	0x00	TotalSizeCmp	TranACK	TranErr	ChangeCondition				
0xE8	H_CHcIntStat	R/(W)	0x00	TotalSizeCmp	TranACK	TranErr	ChangeCondition				
0xE9	H_CHdIntStat	R/(W)	0x00	TotalSizeCmp	TranACK	TranErr	ChangeCondition				
0xEA	H_CHeIntStat	R/(W)	0x00	TotalSizeCmp	TranACK	TranErr	ChangeCondition				
0xEB			0xXX								
0xEC			0xXX								
0xED			0xXX								
0xEE			0xXX								
0xEF			0xXX								

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0xF0	H_SIE_IntEnb_0	R/W	0x00				EnDetectCon	EnDetectDiscon	EnDetectRmtWkup	EnDetectDevChirpOK	EnDetectDevChirpNG
0xF1	H_SIE_IntEnb_1	R/W	0x00					EnDisabledCmp	EnResumeCmp	EnSuspendCmp	EnResetCmp
0xF2	H_FIFO_IntEnb	R/W	0x00		EnFIFO_IDE_Cmp	EnFIFO1_Cmp	EnFIFO0_Cmp		EmFIFO_NotEmpty	EnFIFO_Full	EnFIFO_Empty
0xF3	H_FrameIntEnb	R/W	0x00						EnPortErr	EnFrameNumOver	EnSOF
0xF4	H_CHrIntEnb	R/W	0x00				EnCHeIntStat	EnCHdIntStat	EnCHcIntStat	EnCHbIntStat	EnH_CHaIntStat
0xF5	H_CH0IntEnb	R/W	0x00	EnTotalSizeCmp	EnTranACK	EnTranErr	EnChangeCondition			EnCTL_SupportCmp	EnCTL_SupportStop
0xF6	H_CHaIntEnb	R/W	0x00	EnTotalSizeCmp	EnTranACK	EnTranErr	EnChangeCondition			EnBO_SupportCmp	EnBO_SupportStop
0xF7	H_CHbIntEnb	R/W	0x00	EnTotalSizeCmp	EnTranACK	EnTranErr	EnChangeCondition				
0xF8	H_CHcIntEnb	R/W	0x00	EnTotalSizeCmp	EnTranACK	EnTranErr	EnChangeCondition				
0xF9	H_CHdIntEnb	R/W	0x00	EnTotalSizeCmp	EnTranACK	EnTranErr	EnChangeCondition				
0xFA	H_CHeIntEnb	R/W	0x00	EnTotalSizeCmp	EnTranACK	EnTranErr	EnChangeCondition				
0xFB			0xXX								
0xFC			0xXX								
0xFD			0xXX								
0xFE			0xXX								
0xFF			0xXX								

The registers that can be both read and written even during Sleep/Snooze are shown in ***bold face italic***.

The other registers can be read during Active60, Act_Device, and Act_Host, and can be written to during Act_Device.

Note: The Reset values for H_NegoControl_1, H_FrameNumber_H, and H_FrameNumber_L indicate the values read during ACT_HOST. In other states, the Reset values are read as 0x00.

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x100	***H_Reset***	R/W	0x01								***ResetHTM***
0x101			0xXX								
0x102	H_NegoControl_0	R/W	0x1X	AutoModeCancel	HostState[2:0]			AutoMode[3:0]			
0x103											
0x104	H_NegoControl_1	R/W	0x10			PortSpeed[1:0]				DisChirpFinish	RmtWkupDetEnb
0x105			0xXX								
0x106	H_USB_Test	R/W	0x00	EnHS_Test			Test_Force_Enable	Test_SE0_NAK	Test_J	Test_K	Test_Packet
0x107			0xXX								
0x108	H_CHnControl	W	0xXX			AllFIFO_Clr					CH0FIFO_Clr
0x109	H_CHrFIFO_Clr	W	0xXX				CHeFIFO_Clr	CHdFIFO_Clr	CHcFIFO_Clr	CHbFIFO_Clr	CHaFIFO_Clr
0x10A	H_ClrAllCHnJoin	W	0xXX	ClrJoinIDE	ClrJoinFIFO_Stat			ClrJoinDMA1	ClrJoinDMA0	ClrJoinCPU_Rd	ClrJoinCPU_Wr
0x10B			0xXX								
0x10C			0xXX								
0x10D			0xXX								
0x10E			0xXX								
0x10F			0xXX								

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x110	H_CH0SETUP_0	R/W	0x00	SETUP_0[7:0]							
0x111	H_CH0SETUP_1	R/W	0x00	SETUP_1[7:0]							
0x112	H_CH0SETUP_2	R/W	0x00	SETUP_2[7:0]							
0x113	H_CH0SETUP_3	R/W	0x00	SETUP_3[7:0]							
0x114	H_CH0SETUP_4	R/W	0x00	SETUP_4[7:0]							
0x115	H_CH0SETUP_5	R/W	0x00	SETUP_5[7:0]							
0x116	H_CH0SETUP_6	R/W	0x00	SETUP_6[7:0]							
0x117	H_CH0SETUP_7	R/W	0x00	SETUP_7[7:0]							
0x118			0xXX								
0x119			0xXX								
0x11A			0xXX								
0x11B			0xXX								
0x11C			0xXX								
0x11D			0xXX								
0x11E	H_FrameNumber_H	R	0x07						FrameNumber[10:8]		
0x11F	H_FrameNumber_L	R	0xFF	FrameNumber[7:0]							

The registers that can be both read and written even during Sleep/Snooze are shown in ***bold face italic***.

The other registers can be read during Active60, Act_Device, and Act_Host, and can be written to during Act_Device.

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x120	H_CH0Config_0	R/W	0x00	ACK_Cnt[3:0]				SpeedMode[1:0]		Toggle	TranGo
0x121	H_CH0Config_1	R/W	0x00	TID[1:0]							
0x122			0xXX								
0x123	H_CH0MaxPktSize	R/W	0x00		MaxPktSize[6:0]						
0x124			0xXX								
0x125			0xXX								
0x126	H_CH0TotalSize_H	R/W	0x00	TotalSize[15:8]							
0x127	H_CH0TotalSize_L	R/W	0x00	TotalSize[7:0]							
0x128	H_CH0HubAdrs	R/W	0x00	HubAdrs[3:0]					Port[2:0]		
0x129	H_CH0FuncAdrs	R/W	0x00	FuncAdrs[3:0]				EP_Number[3:0]			
0x12A			0xXX								
0x12B	H_CTL_SupportControl	R/W	0x00			CTL_SupportState[1::0]					CTL_SupportGo
0x12C			0xXX								
0x12D			0xXX								
0x12E	H_CH0ConditionCode	R	0x00		ConditonCode[2:0]						
0x12F	H_CH0Join	R/W	0x00		JoinFIFO_Stat			JoinDMA1	JoinDMA0	JoinCPU_Rd	JoinCPU_Wr

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x130	H_CHaConfig_0	R/W	0x00	ACK_Cnt[3:0]				SpeedMode[1:0]		Toggle	TranGo
0x131	H_CHaConfig_1	R/W	0x00	TID[1:0]				AutoZerolen			TotalSizeFree
0x132	H_CHaMaxPktSize_H	R/W	0x00							MaxPktSize[9:8]	
0x133	H_CHaMaxPktSize_L	R/W	0x00	MaxPktSize[7:0]							
0x134	H_CHaTotalSize_HH	R/W	0x00	TotalSize[31:24]							
0x135	H_CHaTotalSize_HL	R/W	0x00	TotalSize[23:16]							
0x136	H_CHaTotalSize_LH	R/W	0x00	TotalSize[15:8]							
0x137	H_CHaTotalSize_LL	R/W	0x00	TotalSize[7:0]							
0x138	H_CHaHubAdrs	R/W	0x00	HubAdrs[3:0]					Port[2:0]		
0x139	H_CHaFuncAdrs	R/W	0x00	FuncAdrs[3:0]				EP_Number[3:0]			
0x13A	H_BO_SupportControl	R/W	0x00			BO_TransportState[1::0]					BO_SupportGo
0x13B	H_CSW_RcvDataSize	R	0x00					CSW_RcvDataSize[3:0]			
0x13C	H_OUT_EP_Control	R/W	0x00				OUT_Toggle	OUT_EP_Number[3:0]			
0x13D	H_IN_EP_Control	R/W	0x00				IN_Toggle	IN_EP_Number[3:0]			
0x13E	H_CHaConditionCode	R	0x00		ConditonCode[2::0]						
0x13F	H_CHaJoin	R/W	0x00	JoinIDE	JoinFIFO_Stat			JoinDMA1	JoinDMA0	JoinCPU_Rd	JoinCPU_Wr

The registers that can be both read and written even during Sleep/Snooze are shown in ***bold face italic***.

The other registers can be read during Active60, Act_Device, and Act_Host, and can be written to during Act_Device.

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x140	H_CHbConfig_0	R/W	0x00	ACK_Cnt[3:0]				SpeedMode[1:0]		Toggle	TranGo
0x141	H_CHbConfig_1	R/W	0x00	TID[1:0]		TranType[1:0]		AutoZerolen			TotalSizeFree
0x142	H_CHbMaxPktSize_H	R/W	0x00							MaxPktSize[9:8]	
0x143	H_CHbMaxPktSize_L	R/W	0x00	MaxPktSize[7:0]							
0x144	H_CHbTotalSize_HH	R/W	0x00	TotalSize[31:24]							
0x145	H_CHbTotalSize_HL	R/W	0x00	TotalSize[23:16]							
0x146	H_CHbTotalSize_LH	R/W	0x00	TotalSize[15:8]							
0x147	H_CHbTotalSize_LL	R/W	0x00	TotalSize[7:0]							
0x148	H_CHbHubAdrs	R/W	0x00	HubAdrs[3:0]					Port[2:0]		
0x149	H_CHbFuncAdrs	R/W	0x00	FuncAdrs[3:0]				EP_Number[3:0]			
0x14A	H_CHbInterval_H	R/W	0x00						Interval[10:8]		
0x14B	H_CHdbnterval_L	R/W	0x00	Interval[7::0]							
0x14C			0xXX								
0x14D			0xXX								
0x14E	H_CHbConditionCode	R	0x00		ConditonCode[2::0]						
0x14F	H_CHbJoin	R/W	0x00	JoinIDE	JoinFIFO_Stat			JoinDMA1	JoinDMA0	JoinCPU_Rd	JoinCPU_Wr

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x150	H_CHcConfig_0	R/W	0x00	ACK_Cnt[3:0]				SpeedMode[1:0]		Toggle	TranGo
0x151	H_CHcConfig_1	R/W	0x00	TID[1:0]		TranType[1:0]		AutoZerolen			TotalSizeFree
0x152	H_CHcMaxPktSize_H	R/W	0x00							MaxPktSize[9:8]	
0x153	H_CHcMaxPktSize_L	R/W	0x00	MaxPktSize[7:0]							
0x154	H_CHcTotalSize_HH	R/W	0x00	TotalSize[31:24]							
0x155	H_CHcTotalSize_HL	R/W	0x00	TotalSize[23:16]							
0x156	H_CHcTotalSize_LH	R/W	0x00	TotalSize[15:8]							
0x157	H_HcTotalSize_LL	R/W	0x00	TotalSize[7:0]							
0x158	H_CHcHubAdrs	R/W	0x00	HubAdrs[3:0]					Port[2:0]		
0x159	H_CHcFuncAdrs	R/W	0x00	FuncAdrs[3:0]				EP_Number[3:0]			
0x15A	H_CHcInterval_H	R/W	0x00						Interval[10:8]		
0x15B	H_CHdcnterval_L	R/W	0x00	Interval[7: 0]							
0x15C			0xXX								
0x15D			0xXX								
0x15E	H_CHcConditionCode	R	0x00		ConditonCode[2::0]						
0x15F	H_CHcJoin	R/W	0x00	JoinIDE	JoinFIFO_Stat			JoinDMA1	JoinDMA0	JoinCPU_Rd	JoinCPU_Wr

The registers that can be both read and written even during Sleep/Snooze are shown in ***bold face italic***.

The other registers can be read during Active60, Act_Device, and Act_Host, and can be written to during Act_Device.

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x160	H_CHdConfig_0	R/W	0x00	ACK_Cnt[3:0]				SpeedMode[1:0]		Toggle	TranGo
0x161	H_CHdConfig_1	R/W	0x00	TID[1:0]		TranType[1:0]		AutoZerolen			TotalSizeFree
0x162	H_CHdMaxPktSize_H	R/W	0x00							MaxPktSize[9:8]	
0x163	H_CHdMaxPktSize_L	R/W	0x00	MaxPktSize[7:0]							
0x164	H_CHdTotalSize_HH	R/W	0x00	TotalSize[31:24]							
0x165	H_CHdTotalSize_HL	R/W	0x00	TotalSize[23:16]							
0x166	H_CHdTotalSize_LH	R/W	0x00	TotalSize[15:8]							
0x167	H_CHdTotalSize_LL	R/W	0x00	TotalSize[7:0]							
0x168	H_CHdHubAdrs	R/W	0x00	HubAdrs[3:0]					Port[2:0]		
0x169	H_CHdFuncAdrs	R/W	0x00	FuncAdrs[3:0]				EP_Number[3:0]			
0x16A	H_CHdInterval_H	R/W	0x00						Interval[10:8]		
0x16B	H_CHdInterval_L	R/W	0x00	Interval[7: 0]							
0x16C			0xXX								
0x16D			0xXX								
0x16E	H_CHdConditionCode	R	0x00		ConditonCode[2::0]						
0x16F	H_HdJoin	R/W	0x00	JoinIDE	JoinFIFO_Stat			JoinDMA1	JoinDMA0	JoinCPU_Rd	JoinCPU_Wr

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x170	H_CHeConfig_0	R/W	0x00	ACK_Cnt[3:0]				SpeedMode[1:0]		Toggle	TranGo
0x171	H_CHeConfig_1	R/W	0x00	TID[1:0]		TranType[1:0]		AutoZerolen			TotalSizeFree
0x172	H_CHeMaxPktSize_H	R/W	0x00							MaxPktSize[9:8]	
0x173	H_CHeMaxPktSize_L	R/W	0x00	MaxPktSize[7:0]							
0x174	H_CHeHubAdrs	R/W	0x00	TotalSize[31:24]							
0x175	H_CHeFuncAdrs	R/W	0x00	TotalSize[23:16]							
0x176	H_CHeTotalSize_HH	R/W	0x00	TotalSize[15:8]							
0x177	H_CHeTotalSize_HL	R/W	0x00	TotalSize[7:0]							
0x178	H_CHeTotalSize_LH	R/W	0x00	HubAdrs[3:0]					Port[2:0]		
0x179	H_CHeTotalSize_LL	R/W	0x00	FuncAdrs[3:0]				EP_Number[3:0]			
0x17A	H_CHeInterval_H	R/W	0x00						Interval[10:8]		
0x17B	H_CHeInterval_L	R/W	0x00	Interval[7:0]							
0x17C			0xXX								
0x17D			0xXX								
0x17E	H_CHeConditionCode	R	0x00		ConditonCode[2::0]						
0x17F	H_CHeJoin	R/W	0x00	JoinIDE	JoinFIFO_Stat			JoinDMA1	JoinDMA0	JoinCPU_Rd	JoinCPU_Wr

The registers that can be both read and written even during Sleep/Snooze are shown in ***bold face italic***.

The other registers can be read during Active60, Act_Device, and Act_Host, and can be written to during Act_Device.

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x180	H_CH0StartAdrs_H	R/W	0x00				StartAdrs[12:8]				
0x181	H_CH0StartAdrs_L	R/W	0x00	StartAdrs[7:2]							
0x182	H_CH0EndAdrs_H	R/W	0x00				EndAdrs[12:8]				
0x183	H_CH0EndAdrs_L	R/W	0x00	EndAdrs[7:2]							
0x184	H_CHaStartAdrs_H	R/W	0x00				StartAdrs[12:8]				
0x185	H_CHaStartAdrs_L	R/W	0x00	StartAdrs[7:2]							
0x186	H_CHaEndAdrs_H	R/W	0x00				EndAdrs[12:8]				
0x187	H_CHaEndAdrs_L	R/W	0x00	EndAdrs[7:2]							
0x188	H_CHbStartAdrs_H	R/W	0x00				StartAdrs[12:8]				
0x189	H_CHbStartAdrs_L	R/W	0x00	StartAdrs[7:2]							
0x18A	H_CHbEndAdrs_H	R/W	0x00				EndAdrs[12:8]				
0x18B	H_CHbEndAdrs_L	R/W	0x00	EndAdrs[7:2]							
0x18C	H_CHcStartAdrs_H	R/W	0x00				StartAdrs[12:8]				
0x18D	H_CHcStartAdrs_L	R/W	0x00	StartAdrs[7:2]							
0x18E	H_CHcEndAdrs_H	R/W	0x00				EndAdrs[12:8]				
0x18F	H_CHcEndAdrs_L	R/W	0x00	EndAdrs[7:2]							

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit12	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x190	H_CHdStartAdrs_H	R/W	0x00				StartAdrs[12:8]				
0x191	H_CHdStartAdrs_L	R/W	0x00	StartAdrs[7:2]							
0x192	H_CHdEndAdrs_H	R/W	0x00				EndAdrs[12:8]				
0x193	H_CHdEndAdrs_L	R/W	0x00	EndAdrs[7:2]							
0x194	H_CHeStartAdrs_H	R/W	0x00				StartAdrs[12:8]				
0x195	H_CHeStartAdrs_L	R/W	0x00	StartAdrs[7:2]							
0x196	H_CHeEndAdrs_H	R/W	0x00				EndAdrs[12:8]				
0x197	H_CHeEndAdrs_L	R/W	0x00	EndAdrs[7:2]							
0x198			0xXX								
0x199			0xXX								
0x19A			0xXX								
0x19B			0xXX								
0x19C			0xXX								
0x19D			0xXX								
0x19E			0xXX								
0x19F			0xXX								

0x1A0–0x1FF are reserved. (Except 0x1F5 and 0x1F6)

See Appendix D for more information on the registers listed below.

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit12	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x1F4			0xXX								
0x1F5	H_Protect	R/W	0x00					PortSpeedWrEnb		TranEnb[1:0]	
0x1F6	H_Monitor	R	0x00								TranRunning
0x1F7			0xXX								

7.4 Detailed Description of Device/Host Shared Registers

7.4.1 00h *MainIntStat (Main Interrupt Status)*

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	00h	***MainIntStat***	R	7: ***DeviceIntStat***	0: None	1: Device Interrupts	00h
			R	6: ***HostIntStat***	0: None	1: Host Interrupts	
			R	5: CPU_IntStat	0: None	1: CPU Interrupts	
			R	4: IDE_IntStat	0: None	1: IDE Interrupts	
			R	3: MediaFIFO_IntStat	0: None	1: MediaFIFO Interrupts	
				2:	0:	1:	
				1:	0:	1:	
			R (W)	0: ***FinishedPM***	0: None	1:Detect FinishedPM	

This register shows the causes of interrupts generated in the LSI stipulated herein.

Some bits in this register indicate the causes of interrupts indirectly, i.e., they indicate the 'source' from which an interrupt was generated. Other bits indicate the causes of interrupts directly, i.e., they indicate the 'reason' for which an interrupt was generated. If the cause of an interrupt is indicated indirectly by a bit in this register, read the interrupt status register corresponding to that bit. That way, it is possible to trace back to the bit that indicates the reason that caused the interrupt to occur. The bits indicating the causes of interrupts indirectly are read-only, and are automatically cleared by clearing the bit in another register that indicates the cause of the interrupt directly. On the other hand, the bits indicating the causes of interrupts directly are writable, so that the cause of an interrupt indicated by a bit can be cleared by writing 1 to that bit. If the interrupts corresponding to each bit in an interrupt status register was enabled by the MainIntEnb register and any bit in that register is set to 1, the XINT pin is asserted to generate an interrupt to the CPU. When all causes of interrupts in the status register are cleared, the XINT pin is negated.

Bit7 DeviceIntStat

This bit indicates the cause of interrupt indirectly.

If the DeviceIntStat register has the cause of interrupt and the bit corresponding to that interrupt cause in the DeviceIntEnb register is enabled, this bit is set to 1. This bit is effective even during Sleep/Snooze.

Bit6 HostIntStat

This bit indicates the cause of interrupt indirectly.

If the HostIntStat register has the cause of interrupt and the bit corresponding to that interrupt cause in the HostIntEnb register is enabled, this bit is set to 1. This bit is effective even during Sleep/Snooze.

Bit5 CPU_IntStat

This bit indicates the cause of interrupt indirectly.

If the CPU_IntStat register has the cause of interrupt and the bit corresponding to that interrupt cause in the CPU_IntEnb register is enabled, this bit is set to 1.

Bit4 **IDE_IntStat**

This bit indicates the cause of interrupt indirectly.

If the IDE_IntStat register has the cause of interrupt and the bit corresponding to that interrupt cause in the IDE_IntEnb register is enabled, this bit is set to 1.

Bit3 **MediaFIFO_IntStat**

This bit indicates the cause of interrupt indirectly.

If the has the cause of interrupt MediaFIFO_IntStat register and the bit corresponding to that interrupt cause in the MediaFIFO_IntEnb register is enabled, this bit is set to 1.

Bits2-1 **Reserved**

Bit0 **FinishedPM**

This bit indicates the cause of interrupt directly.

If GoSLEEP, GoSNOOZE, GoActive60, GoActDevice, or GoActHost is set by the PM_Control_0 register and the instructed state is reached, this bit is set to 1. This bit is effective even during Sleep/Snooze.

7.4.2 01h *DeviceIntStat (Device Interrupt Status)*

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	01h	***DeviceIntStat***	R (W)	7: ***VBUS_Changed***	0: None	1: VBUS is Changed	00h
				6:	0:	1:	
			R	5: ***D_SIE_IntStat***	0: None	1: SIE Interrupts	
			R	4: D_BulkIntStat	0: None	1: Bulk Interrupts	
			R (W)	3: RcvEP0SETUP	0: None	1: Receive EP0 SETUP	
			R	2: D_FIFO_IntStat	0: None	1: FIFO Interrupts	
			R	1: D_EP0IntStat	0: None	1: EP0 Interrupts	
			R	0: D_EPrIntStat	0: None	1: EPr Interrupts	

This register shows the device-related interrupts.

Some bits in this register indicate the causes of interrupts indirectly, i.e., they indicate the ‘source’ from which an interrupt was generated. Other bits indicate the causes of interrupts directly, i.e., they indicate the ‘reason’ for which an interrupt was generated. If the cause of an interrupt is indicated indirectly by a bit in this register, read the interrupt status register corresponding to that bit. That way, it is possible to trace back to the bit that indicates the reason that caused the interrupt to occur. The bits indicating the causes of interrupts indirectly are read-only, and are automatically cleared by clearing the bit in another register that indicates the cause of the interrupt directly. On the other hand, the bits indicating the causes of interrupts directly are writable, so that the cause of an interrupt indicated by a bit can be cleared by writing 1 to that bit.

Bit7 VBUS_Changed

This bit indicates the cause of interrupt directly.

This bit is set to 1 when the status of the VBUS bit is changed.

Check the VBUS bit in the D_USB_Status register to confirm the VBUS_B status. If VBUS = 0, it means that the cable is removed. This bit is effective even during Sleep/Snooze.

Bit6 Reserved

Bit5 D_SIE_IntStat

This bit indicates the cause of interrupt indirectly.

If the D_SIE_IntStat register has the cause of interrupt and the bit corresponding to that interrupt cause in the D_SIE_IntEnb register is enabled, this bit is set to 1. This bit is effective even during Sleep/Snooze.

Bit4 D_BulkIntStat

This bit indicates the cause of interrupt indirectly.

If the D_Bulk_IntStat register has the cause of interrupt and the bit corresponding to that interrupt cause in the D_Bulk_IntEnb register is enabled, this bit is set to 1.

This bit is available only when the LSI is operating in device mode.

Bit3 RcvEP0SETUP

This bit indicates the cause of interrupt directly.

When the setup stage for a control transfer is complete and the received data is stored in the D_EP0Setup_0 through D_EP0Setup_7 registers, this bit is set to 1. At the same time, the ForceSTALL bit in the D_EP0ControlIN, D_EP0ControlOUT registers is cleared to 0, and the ForceNAK and ToggleStat bits in the D_EP0ControlIN, D_EP0ControlOUT registers and the ProtectEP0 bit in the D_SETUP Control register are set to 1, all automatically. SetAddress() requests are automatically responded by the AutoSetAddress function, and this status is not set.

This bit is available only when the LSI is operating in device mode.

Bit2 D_FIFO_IntStat

This bit indicates the cause of interrupt indirectly.

If the D_FIFO_IntStat register has the cause of interrupt and the bit corresponding to that interrupt cause in the D_FIFO_IntEnb register is enabled, this bit is set to 1.

This bit is available only when the LSI is operating in device mode.

Bit1 D_EP0IntStat

This bit indicates the cause of interrupt indirectly.

If the D_EP0IntStat register has the cause of interrupt and the bit corresponding to that interrupt cause in the D_EP0IntEnb register is enabled, this bit is set to 1.

This bit is available only when the LSI is operating in device mode.

Bit0 D_EPrIntStat

This bit indicates the cause of interrupt indirectly.

If the D_EPrIntStat register has the cause of interrupt and the bit corresponding to that interrupt cause in the D_EPaIntEnb register is enabled, this bit is set to 1.

This bit is available only when the LSI is operating in device mode.

7.4.3 02h *HostIntStat (Host Interrupt Status)*

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	02h	***HostIntStat***	R (W)	7: ***VBUS_Err***	0: None	1: VBUS Error	00h
			R (W)	6: ***LineStatusChanged***	0: None	1: Line Status Changed	
				5: (Reserved)	0:	1:	
				4: (Reserved)	0:	1:	
				3: (Reserved)	0:	1:	
				2: (Reserved)	0:	1:	
				1: (Reserved)	0:	1:	
				0: (Reserved)	0:	1:	

This register shows the host-related interrupts.

Some bits in this register indicate the causes of interrupts indirectly, i.e., they indicate the ‘source’ from which an interrupt was generated. Other bits indicate the causes of interrupts directly, i.e., they indicate the ‘reason’ for which an interrupt was generated. If the cause of an interrupt is indicated indirectly by a bit in this register, read the interrupt status register corresponding to that bit. That way, it is possible to trace back to the bit that indicates the reason that caused the interrupt to occur. The bits indicating the causes of interrupts indirectly are read-only, and are automatically cleared by clearing the bit in another register that indicates the cause of the interrupt directly. On the other hand, the bits indicating the causes of interrupts directly are writable, so that the cause of an interrupt indicated by a bit can be cleared by writing 1 to that bit.

Bit7 ***VBUS_Err***

This bit indicates the cause of interrupt directly.

When a VBUS error signal (high to low-going edge) is input from an external VBUS power switch connected to the VBUSFLG_A pin externally to the chip, this bit is set to 1.

Check the VBUS_State bit in the H_USB_Status register to confirm the VBUSFLG_A pin state.

The above-mentioned error signal differs with specifications of an externally connected power switch, so consult specifications of the power switch used in your system.

This bit is effective even during Sleep/Snooze.

Bit6 ***LineStateChanged***

This bit indicates the cause of interrupt directly.

It indicates that the DP and DM pins of the host port have changed state from SE0.

This bit is used to detect a change of signal lines at the host port when the USB host function is not in use. If HostIntEnb.EnLineStateChanged is enabled during USB host operation, this bit will be frequently asserted. Therefore, make sure HostIntEnb.EnLineStateChanged is disabled during USB host operation.

Bit5 H_SIE_IntStat_1

This bit indicates the cause of interrupt indirectly.

If the H_SIE_IntStat_1 register has the cause of interrupt and the bit corresponding to that interrupt cause in the H_SIE_IntEnb1 register is enabled, this bit is set to 1.

This bit is available only when the LSI is operating in device mode.

Bit4 H_SIE_IntStat_0

This bit indicates the cause of interrupt indirectly.

If the H_SIE_IntStat_0 register has the cause of interrupt and the bit corresponding to that interrupt cause in the H_SIE_IntEnb0 register is enabled, this bit is set to 1.

This bit is available only when the LSI is operating in device mode.

Bit3 H_FrameIntStat

This bit indicates the cause of interrupt indirectly.

If the H_FrameIntStat register has the cause of interrupt and the bit corresponding to that interrupt cause in the H_FrameIntEnb register is enabled, this bit is set to 1.

This bit is available only when the LSI is operating in device mode.

Bit2 H_FIFO_IntStat

This bit indicates the cause of interrupt indirectly.

If the H_FIFO_IntStat register has the cause of interrupt and the bit corresponding to that interrupt cause in the H_FIFO_IntEnb register is enabled, this bit is set to 1.

This bit is available only when the LSI is operating in device mode.

Bit1 H_CH0_IntStat

This bit indicates the cause of interrupt indirectly.

If the has the cause of interrupt H_CH0_IntStat register and the bit corresponding to that interrupt cause in the H_CH0_IntEnb register is enabled, this bit is set to 1.

This bit is available only when the LSI is operating in device mode.

Bit0 H_CHr_IntStat

This bit indicates the cause of interrupt indirectly.

If the H_CHr_IntStat register has the cause of interrupt and the bit corresponding to that interrupt cause in the H_CHr_IntEnb register is enabled, this bit is set to 1.

This bit is available only when the LSI is operating in device mode.

7.4.4 03h CPU_IntStat (CPU Interrupt Status)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	03h	CPU_IntStat	R (W)	7: RAM_RdCmp	0: None	1: RAM Read Complete	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
			R (W)	3: DMA1_Countup	0: None	1: DMA1 Counter Overflow	
			R (W)	2: DMA1_Cmp	0: None	1: DMA1 Complete	
			R (W)	1: DMA0_CountUp	0: None	1: DMA0 Counter Overflow	
			R (W)	0: DMA0_Cmp	0: None	1: DMA0 Complete	

This register shows the interrupts associated with the CPU interface.

The cause of interrupt indicated by any bit in this register can be cleared by writing 1 to that bit.

Bit7 RAM_RdCmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when the data placed in the RAM_Rd_XX after being read from the RAM is prepared in the RAM_Rd function.

Bits6-4 Reserved

Bit3 DMA1_CountUp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when the value of the DMA1_Count_HH,HL,LH,LL has overflowed while the LSI is operating in free-running mode of transfer. The value of the DMA1_Count_HH,HL,LH,LL recycles to 0, with the DMA operation continued.

Bit2 DMA1_Cmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when the DMA transfer in progress is stopped or when processing for termination of transfer is complete after completion of a specified number of transfers.

Bit1 DMA0_CountUp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when the value of the DMA0_Count_HH,HL,LH,LL has overflowed while the LSI is operating in free-running mode of transfer. The value of the DMA0_Count_HH,HL,LH,LL recycles to 0, with the DMA operation continued.

Bit0 DMA0_Cmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when the DMA transfer in progress is stopped or when processing for termination of transfer is complete after completion of a specified number of transfers.

7.4.5 04h IDE_IntStat (IDE Interrupt Status)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	04h	IDE_IntStat	R (W)	7: IDE_RegCmp	0: None	1: Register Access Cmplete	00h
			R (W)	6: IDE_RegErr	0: None	1: Register Access Error	
			R (W)	5: IDE_SeqWrRegCmp	0: None	1: Sequence Write Complete	
			R (W)	4: CompleteINTRQ	0: None	1: Auto Status Read Compl.	
				3:	0:	1:	
			R (W)	2: IDE_Cmp	0: None	1: DMA Complete	
			R (W)	1: DetectINTRQ	0: None	1: Detected Interrupt	
			R (W)	0: DetectTerm	0: None	1: Detected Device terminate	

This register shows the interrupts associated with the IDE.

The cause of interrupt indicated by any bit in this register can be cleared by writing 1 to that bit.

Bit7 IDE_RegCmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a read/write access to the IDE register via the IDE_RegAdr register is complete.

Bit6 IDE_RegErr

This bit indicates the cause of interrupt directly.

This bit is set to 1 in the following cases:

1) A read/write access to the IDE register via the IDE_RegAdr register is attempted during the operation sequence of an auto status read by the IDE_RegConfig register.

2) A read/write access to the IDE register via the IDE_RegAdr register is attempted during the operation sequence of a sequential write by the IDE_SeqWrRegControl register.

Bit5 IDE_SeqWrRegCmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a sequential write operation by the IDE_SeqWrRegControl register is complete.

Bit4 CompleteINTRQ

This bit indicates the cause of interrupt directly.

This bit is set to 1 when an auto status read operation by the IDE_RegConfig register is complete.

Bit3 Reserved

Bit2 IDE_Cmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a DMA operation by the IDE_Control register is complete.

Bit1 DetectINTRQ

This bit indicates the cause of interrupt directly.

This bit is set to 1 when while the EnAutoStsRd bit in the IDE_RegConfig register has not been set, a rising edge of the HINTRQ signal of the IDE is detected.

Bit0 DetectTerm

This bit indicates the cause of interrupt directly.

This bit is set to 1 when while DMA operation of the IDE by the IDE_Control register is underway in transfer mode of Ultra-DMA, a “device terminate” event is detected.

7.4.6 05h MediaFIFO_IntStat (Media FIFO Interrupt Status)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	05h	MediaFIFO_IntStat		7:	0:	1:	00h
			R (W)	6: MediaIDE_Cmp	0: None	1: Media FIFO IDE Complete	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R (W)	2: FIFO_NotEmpty	0: None	1: FIFO NotEmpty	
			R (W)	1: FIFO_Full	0: None	1: FIFO Full	
			R (W)	0: FIFO_Empty	0: None	1: FIFO Empty	

This register shows the interrupts associated with the Media FIFO.

The cause of interrupt indicated by any bit in this register can be cleared by writing 1 to that bit.

Bit7 Reserved

Bit6 MediaDE_Cmp

This bit indicates the cause of interrupt directly.

If the IDE is joined and Dir = 0, this bit is set to 1 when the Media FIFO is emptied after an IDE transfer is completed. If Dir = 1, this bit is set to 1 when an IDE transfer is completed.

Bits5-3 Reserved

Bit2 FIFO_NotEmpty

This bit indicates the cause of interrupt directly.

This bit is set to 1 if any data is present in the Media FIFO (NotEmpty).

Bit1 FIFO_Full

This bit indicates the cause of interrupt directly.

This bit is set to 1 when the Media FIFO is full.

Bit0 FIFO_Empty

This bit indicates the cause of interrupt directly.

This bit is set to 1 when the Media FIFO is empty.

7.4.7 06h~0Fh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	06h -0Fh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.8 10h *MainIntEnb (Main Interrupt Enable)*

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	10h	***MainIntEnb***	R / W	7: ***EnDeviceIntStat***	0: Disable	1: Enable	00h
			R / W	6: ***EnHostIntStat***	0: Disable	1: Enable	
			R / W	5: EnCPU_IntStat	0: Disable	1: Enable	
			R / W	4: En IDE_IntStat	0: Disable	1: Enable	
			R / W	3: EnMediaFIFO_IntStat	0: Disable	1: Enable	
				2:	0:	1:	
				1:	0:	1	
			R / W	0: ***EnFinishedPM***	0: Disable	1: Enable	

This register is used to enable or disable the assertion of the interrupt signal (XINT) for the interrupt causes accommodated in the MainIntStat register.

These interrupt causes can be enabled for interrupt generation by setting the corresponding bits in this register to 1.

The EnDevice_IntStat, EnHost_IntStat, and EnFinishedPM bits are effective even during Sleep/Snooze.

7.4.9 11h *DeviceIntEnb (Device Interrupt Enable)*

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	11h	***DeviceIntEnb***	R / W	7: ***EnVBUS_Changed***	0: Disable	1: Enable	00h
				6:	0:	1:	
			R / W	5: ***EnD_SIE_IntStat***	0: Disable	1: Enable	
			R / W	4: EnD_BulkIntStat	0: Disable	1: Enable	
			R / W	3: EnRcvEP0SETUP	0: Disable	1: Enable	
			R / W	2: EnD_FIFO_IntStat	0: Disable	1: Enable	
			R / W	1: EnD_EP0IntStat	0: Disable	1: Enable	
			R / W	0: EnD_EPrIntStat	0: Disable	1: Enable	

This register is used to enable or disable the assertion of the DeviceIntStat bit in the MainIntStat register for the interrupt causes accommodated in the DeviceIntStat register.

The EnVBUS_Changed and EnD_SIE_IntStat bits are effective even during Sleep/Snooze.

7.4.10 12h *HostIntEnb (Host Interrupt Enable)*

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	12h	***HostIntEnb***	R / W	7: ***EnVBUS_Err***	0: Disable	1: Enable	00h
			R / W	6: ***EnLineStatusChanged***	0: Disable	1: Enable	
			R / W	5: EnH_SIE_IntStat_1	0: Disable	1: Enable	
			R / W	4: EnH_SIE_IntStat_0	0: Disable	1: Enable	
			R / W	3: EnH_FrameIntStat	0: Disable	1: Enable	
			R / W	2: EnH_FIFOIntStat	0: Disable	1: Enable	
			R / W	1: EnH_CH0IntStat	0: Disable	1: Enable	
			R / W	0: EnH_CHrIntStat	0: Disable	1: Enable	

This register is used to enable or disable the assertion of the HostIntStat bit in the MainIntStat register for the interrupt causes accommodated in the HostIntStat register.

The EnVBUS_Err and EnLineStateChanged bits are effective even during Sleep/Snooze.

7.4.11 13h CPU_IntEnb (CPU Interrupt Enable)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	13h	CPU_IntEnb	R / W	7: EnRAM_RdCmp	0: Disable	1: Enable	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
			R / W	3: EnDMA1_Countup	0: Disable	1: Enable	
			R / W	2: EnDMA1_Cmp	0: Disable	1: Enable	
			R / W	1: EnDMA0_CountUp	0: Disable	1: Enable	
			R / W	0: EnDMA0_Cmp	0: Disable	1: Enable	

This register is used to enable or disable the assertion of the CPU_IntStat bit in the MainIntStat register for the interrupt causes accommodated in the CPU_IntStat register.

7.4.12 14h IDE_IntEnb (IDE Interrupt Enable)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	14h	IDE_IntEnb	R / W	7: EnIDE_RegCmp	0: Disable	1: Enable	00h
			R / W	6: EnIDE_RegErr	0: Disable	1: Enable	
			R / W	5: En_SeqWrRegCmp	0: Disable	1: Enable	
			R / W	4: EnCompleteINTRQ	0: Disable	1: Enable	
				3:	0:	1:	
			R / W	2: EnIDE_Cmp	0: Disable	1: Enable	
			R / W	1: EnDetectINTRQ	0: Disable	1: Enable	
			R / W	0: EnDetectTerm	0: Disable	1: Enable	

This register is used to enable or disable the assertion of the IDE_IntStat bit in the MainIntStat register for the interrupt causes accommodated in the IDE_IntStat register.

7.4.13 15h MediaFIFO_IntEnb (Media FIFO Interrupt Enable)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	15h	MediaFIFO_IntEnb		7:	0:	1:	00h
			R / W	6: EnMediaIDE_Cmp	0: Disable	1: Enable	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R / W	2: EnFIFO_NotEmpty	0: Disable	1: Enable	
			R / W	1: EnFIFO_Full	0: Disable	1: Enable	
			R / W	0: Empty	0: Disable	1: Enable	

This register is used to enable or disable the assertion of the MediaFIFO_IntStat bit in the MainIntStat register for the interrupt causes accommodated in the MediaFIFO_IntStat register.

7.4.14 16h~1Fh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
	16h -1Fh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.15 20h *RevisionNum (Revision Number)*

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	20h	***RevisionNum***	R	7: ***RevisionNum [7]***	Revision Number	50h
				6: ***RevisionNum [6]***		
				5: ***RevisionNum [5]***		
				4: ***RevisionNum [4]***		
				3: ***RevisionNum [3]***		
				2: ***RevisionNum [2]***		
				1: ***RevisionNum [1]***		
				0: ***RevisionNum [0]***		

This register indicates the revision of the LSI stipulated herein. This register can be accessed even during Sleep/Snooze.

The revision number relating to the current specifications of the LSI is 0x50.

7.4.16 21h *ChipReset (Chip Reset)*

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	21h	***ChipReset***		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
			W	0: ***AllReset***	0: None	1: Reset	

This register is used to reset the LSI stipulated herein.

This register can be accessed even during Sleep/Snooze.

Bits7-1 Reserved

Bit0 AllReset

This bit resets the entire circuit of the LSI. It works the same way as the external reset pin (XRST).

Do not write to this register unless the LSI needs to be reset.

Be aware that if a write to this register is attempted for other than resetting the LSI in violation of A.C. characteristics, the LSI may operate erratically.

7.4.17 22h *PM_Control_0 (Power Management Control 0)*

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	22h	***PM_Control_0***	R / W	7: ***GoSLEEP***	0: Do nothing	1: Go to SLEEP	0XXh
			R / W	6: ***GoSNOOZE***	0: Do nothing	1: Go to SNOOZE	
			R / W	5: ***GoActive60***	0: Do nothing	1: Go to ACTIVE60	
			R / W	4: ***GoActDevice***	0: Do nothing	1: Go to ACT_DEVICE	
			R / W	3: ***GoActHost***	0: Do nothing	1: Go to ACT_HOST	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register is used to set the operations relating to the power management of the LSI stipulated herein.

This register can be accessed even during Sleep/Snooze.

Bit7 GoSLEEP

This bit causes the LSI to start shifting to Sleep state from any state other than that.

When this bit is set to 1 during the Snooze state, the LSI turns off the oscillator, thereby shifting to Sleep state.

When this bit is set to 1 during the Active60 state, the LSI first turns off the PLL60 and then the oscillator, thereby shifting to Sleep state.

When this bit is set to 1 during the ActDevice state, the LSI first turns off the DevicePLL480 and then the PLL60 and finally the oscillator, thereby shifting to Sleep state.

When this bit is set to 1 during the ActHost state, the LSI first turns off the HostPLL480 and then the PLL60 and finally the oscillator, thereby shifting to Sleep state.

Upon completion of a state transition no matter which state it has occurred from, this bit is automatically cleared and at the same time, the MainIntStat.FinishedPM bit is set.

Bit6 GoSNOOZE

This bit causes the LSI to start shifting to Snooze state from any state other than that.

When this bit is set to 1 during the Sleep state, the LSI turns on the oscillator and after the oscillator's oscillation stabilization time has elapsed (set by WakeupTim_H,L), shifts to the Snooze state.

When this bit is set to 1 during the Active60 state, the LSI turns off the PLL60, thereby shifting to Snooze state.

When this bit is set to 1 during the ActDevice state, the LSI first turns off the DevicePLL480 and then the PLL60, thereby shifting to Snooze state.

When this bit is set to 1 during the ActHost state, the LSI first turns off the HostPLL480 and then the PLL60, thereby shifting to Snooze state.

Upon completion of a state transition no matter which state it has occurred from, this bit is automatically cleared and at the same time, the MainIntStat.FinishedPM bit is set.

Bit5 **GoActive60**

This bit causes the LSI to start shifting to Active60 state from any state other than that.

When this bit is set to 1 during the Sleep state, the LSI turns on the oscillator and after the oscillator's oscillation stabilization time (set by WakeupTim_H,L) has elapsed, turns on the PLL60 and after the PLL60 oscillation stabilization time (approx. 250 μs) has elapsed, shifts to the Active60 state.

When this bit is set to 1 during the Snooze state, the LSI turns on the PLL60 and after the PLL60 oscillation stabilization time (approx. 250 μs) has elapsed, shifts to the Active60 state.

When this bit is set to 1 during the ActDevice state, the LSI turns off the DevicePLL480, thereby shifting to Active60 state.

When this bit is set to 1 during the ActHost state, the LSI turns off the HostPLL480, thereby shifting to Active60 state.

Upon completion of a state transition no matter which state it has occurred from, this bit is automatically cleared and at the same time, the MainIntStat.FinishedPM bit is set.

Bit4 **GoActDevice**

This bit causes the LSI to start shifting to ActDevice state from any state other than that.

When this bit is set to 1 during the Sleep state, the LSI turns on the oscillator and after the oscillator's oscillation stabilization time has elapsed (set by WakeupTim_H,L), turns on the PLL60 and after the PLL60 oscillation stabilization time (approx. 250 μs) has elapsed, turns on the DevicePLL480 and after the PLL480 oscillation stabilization time (approx. 250 μs) has elapsed, shifts to the ActDevice state.

When this bit is set to 1 during the Snooze state, the LSI turns on the PLL60 and after the PLL60 oscillation stabilization time (approx. 250 μs) has elapsed, turns on the DevicePLL480 and after the PLL480 oscillation stabilization time (approx. 250 μs) has elapsed, shifts to the ActDevice state.

When this bit is set to 1 during the Active60 state, the LSI starts the oscillation of DevicePLL480 and after the PLL480 oscillation stabilization time (approx. 250 μs) has elapsed, shifts to the ActiveDevice state.

When this bit is set to 1 during the ActHost state, the LSI turns off the HostPLL480, thereby shifting to Active60 state. It then turns on the DevicePLL480 and after the PLL480 oscillation stabilization time (approx. 250 μs) has elapsed, shifts to the ActDevice state.

Upon completion of a state transition no matter which state it has occurred from, this bit is automatically cleared and at the same time, the MainIntStat.FinishedPM bit is set.

Bit3 GoActHost

This bit causes the LSI to start shifting to ActHost state from any state other than that.

When this bit is set to 1 during the Sleep state, the LSI turns on the oscillator and after the oscillator's oscillation stabilization time has elapsed (set by WakeupTim_H,L), turns on the PLL60 and after the PLL60 oscillation stabilization time (approx. 250 μs) has elapsed, turns on the HostPLL480 and after the PLL480 oscillation stabilization time (approx. 250 μs) has elapsed, shifts to the ActHost state.

When this bit is set to 1 during the Snooze state, the LSI turns on the PLL60 and after the PLL60 oscillation stabilization time (approx. 250 μs) has elapsed, turns on the HostPLL480 and after the PLL480 oscillation stabilization time (approx. 250 μs) has elapsed, shifts to the ActHost state.

When this bit is set to 1 during the Active60 state, the LSI turns on the HostPLL480 and after the PLL480 oscillation stabilization time (approx. 250 μs) has elapsed, shifts to the ActHost state.

When this bit is set to 1 during the ActDevice state, the LSI turns on the HostPLL480, thereby shifting to Active60 state. It then turns on the HostPLL480 and after the PLL480 oscillation stabilization time (approx. 250 μs) has elapsed, shifts to the ActHost state.

Upon completion of a state transition no matter which state it has occurred from, this bit is automatically cleared and at the same time, the MainIntStat.FinishedPM bit is set.

* The LSI stipulated herein has the XINT signal masked not to be asserted during Snooze by an interrupt status that cannot be accessed during Sleep/Snooze (hereafter referred to as a "synchronous status"). However, to ensure that the XINT pin will not be asserted at the same time the LSI has exited Snooze, the firmware should execute the processing described below.

<Before entering Sleep/Snooze>

Process the synchronous status to clear it (~IntStat).

Disable the synchronous status (~IntEnb).

<After exiting Sleep/Snooze>

Clear the synchronous status (~IntStat).

Re-enable the synchronous status (~IntEnb).

Bits2-0 Reserved

7.4.18 23h *PM_Control_1 (Power Management Control 1)*

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	23h	***PM_Control_1***		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
			R	3: ***PM_State [3]***	PM_State [3:0]		
				2: ***PM_State [2]***			
				1: ***PM_State [1]***			
				0: ***PM_State [0]***			

This register is used to set the operations relating to the power management of the LSI stipulated herein.

This register can be accessed even during Sleep/Snooze.

Bits7-4 Reserved

Bits3-0 PM_State [3:0]

Indicates the state of power mode.

0000: Sleep state	(OSC off, PLL60 off, DevicePLL480 off, HostPLL480 off)
0001: Snooze state	(OSC on, PLL60 off, DevicePLL480 off, HostPLL480 off)
0011: Active60 state	(OSC on, PLL60 on, DevicePLL480 off, HostPLL480 off)
0111: ActDevice state	(OSC on, PLL60 on, DevicePLL480 on, HostPLL480 off)
1011: ActHost state	(OSC on, PLL60 on, DevicePLL480 off, HostPLL480 on)
Other: Unused	

Note that these power mode states are unstable for a transitory period from when PM_Control_0.GoXXXX is set till when the MainIntStat.FinishedPM interrupt status is set and PM_Control_0.GoXXXX is cleared. Therefore, do not refer to these states during that time.

7.4.19 24h *WakeupTim_H (Wakeup Time High)*

7.4.20 25h *WakeupTim_L* (Wakeup Time Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	24h -25h	***WakeupTim_H*** ***WakeupTim_L***	R / W	***WakeupTim [15:0]***	Wakeup Time [15:0]	0000h

These registers are used to set the oscillator's oscillation stabilization time to be waited for when the LSI returns from the Sleep state to the Snooze state. These registers can be accessed even during Sleep.

When the PM_Control_0.GoActDevice, PM_Control_0.GoActHost, PM_Control_0.GoActive60, or PM_Control_0.GoSNOOZE bit is set by writing 1 during the Sleep state, the oscillator cell is enabled, causing the oscillator to start oscillating. At this time, the counter is loaded with the set value of these WakeupTim_H,L registers and starts counting down synchronously with the rising edge of the OSC. When the counter is complete counting down, the gate for the internal OSCCLK is opened, allowing CLK to be sent out to the PLL and other circuits.

This oscillation stabilization time varies with the resonator, oscillator cell, circuit board, and load capacitance. If the LSI needs to be dropped into the Sleep state during Suspend of the USB, the internal SCLK must be stabilized to 60 MHz ±10% within 5.1 ms after Reset of the USB is detected.

Therefore, the sum total of the following must be 5.1 ms or less:
Oscillator's oscillation stabilization time + PLL60 stabilization time (less than 250 μs) + PLL480 stabilization time (less than 250 μs)

7.4.21 26h *H_USB_Control (Host USB Control)*

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	26h	***H_USB_Control_1***	R / W	7: ***VBUS_Enb***	0: Disable	1: Enable	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register is used to set the operations associated with the host.

Bit7 ***VBUS_Enb***

This bit sets the VBUSEN_A pin (output) state. The default output state of this pin is low.

This bit is effective even during Sleep/Snooze.

Bits6-0 Reserved

7.4.22 27h *H_XcvrControl (Host Xcvr Control)*

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	27h	***H_XcvrControl***	R / W	7: ***TermSelect***	0: HS Termination	1: FS Termination	91h
				6: ***RemovedRPD***	0: RPD ON	1: RPD OFF	
			R / W	4: ***XcvrSelect[1]***	XcvrSelect[1:0]		
			R / W	4: ***XcvrSelect[0]***			
				3:	0:	1:	
				2:	0:	1:	
			R / W	1: ***OpMode [1]***	OpMode [1:0]		
				0: ***OpMode [0]***			

This register is used to make settings relating to the host transceiver macro.

Bit7 TermSelect

This bit selects FS or HS termination to enable it.

Do not set this bit manually because the operation mode is automatically set to H_NegoControl_0.AutoMode by the hardware.

Bit6 RemovedRPD

This bit turns on/off the internal pulldown resistors for DP_A aand DM_A that are host data lines.

0: RPD is turned on

1: RPD is turned off

Normally, use this bit as “0” (on). In particular, always use "0” (on) during USB host operation (including SUSPEND). If you use values other than “0” the charcteristics of the host data line changes and it may cause the USB to operate erratically.

Bits5-4 XcvrSelect[1:0]

These bits select the FS, HS, or LS transceiver to enable it.

00: High Speed transceiver

01: Full Speed transceiver

10: Reserved

11: Low Speed transceiver

Do not set this bit manually because the operation mode is automatically set to H_NegoControl_0.AutoMode by the hardware.

Bits3-2 Reserved

Bits1-0 OpMode

These bits set the operation mode of the HTM.

Do not set this bit manually because the operation mode is automatically set to H_NegoControl_0.AutoMode by the hardware.

However, if the signal line change status of the host port is to be detected by state other than the ACT_HOST state, refer to the section on signal line change status (6.1.2.1.2) and then set this bit.

OpMode		
00	"Normal Operation"	Normal operating state
01	"Non-Driving"	Unused state
10	"Disable Bitstuffing and NRZI encoding"	Bitstuffing and NRZI encoding function disabled state in normal operating state
11	"Power-Down"	State where only single-end receiver is used

7.4.23 28h *D_USB_Status (Device USB Status)*

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device /Host	28h	***D_USB_Status***	R	7: ***VBUS***	0: VBUS = L	1: VBUS = H	XXh
			R / W	6: FSxHS	0: HS mode	1: FS mode	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R	1: ***LineState [1]***	Line State [1:0]		
				0: ***LineState [0]***			

This register indicates the device-related status.

Bit7 ***VBUS***

This bit indicates the status of the VBUS_B pin. This bit is effective even during Sleep/Snooze.

Bit6 **FSxHS**

This bit indicates the current operation mode. This bit is automatically set when HS Detection Handshaking is executed via the USB_Control.GoChirp bit (see “Functional Description”). Although operation mode can be forcibly changed by writing to this bit, it is recommended that this bit be manipulated only when operation mode needs to be changed without performing HS Detection Handshaking during a simulation, etc.

This bit should be set to 1 (= FS mode) when cable is attached.

This bit can be read during Active60, Act_Device, and Act_Host, and can be written to during Act_Device.

Bits5-2 **Reserved**

Bits1-0 ***LineState [1:0]***

These bits indicate the signal status on the USB cable. These bits are effective even during Sleep/Snooze.

If the XcvrSelect bit = 1 (FS transceiver selected) when the D_XcvrControl register’s TermSelect bit = 1 (FS termination selected), these bits indicate the received value of the FS receiver of the DP/DM. If the XcvrSelect bit = 0 (HS transceiver selected), these bits indicate the received value of the HS receiver.

When TermSelect = 0, these bits indicate a USB bus activity.

LineState		
TermSelect	**DP / DM**	**LineState [1:0]**
0	Don’t Care	Bus activity
1	SE0	0b00
1	J	0b01
1	K	0b10
1	SE1	0b11

7.4.24 29h *H_USB_Status (Host USB Status)*

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	29h	***H_USB_Status***	R	7: ***VBUS_State***	0: VBUSFLG = InActive	1: VBUSFLG = Active	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R	1: ***LineState [1]***	Line State [1:0]		
				0: ***LineState [0]***			

This register indicates the host-related status.

Bit7 ***VBUS_State***

This bit indicates the status of the VBUSFLG_A pin. This bit is effective even during Sleep/Snooze.

Bits6-2 Reserved

Bits1-0 ***LineState [1:0]***

These bits indicate the signal status on the USB cable. These bits are effective even during Sleep/Snooze.

If the D_XcvrControl register's XcvrSelect[1:0] = 01 (FS transceiver selected), these bits indicate the received value of the FS receiver of the DP/DM. If XcvrSelect[1:0] = 11 (LS transceiver selected), these bits indicate the received value of the LS receiver.

When XcvrSelect[1:0] = 00 (HS transceiver selected), these bits indicate a USB bus activity.

LineState		
XcvrSelect[1:0]	**DP / DM**	**LineState [1:0]**
00	Don't Care	Bus activity Bus activity detected: 0b01 No bus activity detected: 0b00
01 or 11	SE0	0b00
01 or 11	J	0b01
01 or 11	K	0b10
01 or 11	SE1	0b11

Note: XcvrSelect[1:0] = 10 is reserved, so that when this code is set, operation of the LSI cannot be guaranteed.

7.4.25 2A~2Fh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	2Ah -2Fh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.26 30h FIFO_Rd_0 (FIFO Read 0)

7.4.27 31h FIFO_Rd_1 (FIFO Read 1)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	30h	FIFO_Rd_0	R	7: FIFO_Rd_0 [7]	Endpoint n / Channel n / Media FIFO Read	XXh
				6: FIFO_Rd_0 [6]		
				5: FIFO_Rd_0 [5]		
				4: FIFO_Rd_0 [4]		
				3: FIFO_Rd_0 [3]		
				2: FIFO_Rd_0 [2]		
				1: FIFO_Rd_0 [1]		
				0: FIFO_Rd_0 [0]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	31h	FIFO_Rd_1	R	7: FIFO_Rd_1 [7]	Endpoint n / Channel n / Media FIFO Read	XXh
				6: FIFO_Rd_1 [6]		
				5: FIFO_Rd_1 [5]		
				4: FIFO_Rd_1 [4]		
				3: FIFO_Rd_1 [3]		
				2: FIFO_Rd_1 [2]		
				1: FIFO_Rd_1 [1]		
				0: FIFO_Rd_1 [0]		

30h.Bit7-0, 31h.Bit7-0 FIFO_Rd_0 [7:0], FIFO_Rd_1 [7:0]

These registers allow data to be read from the endpoint FIFO with the D_EPx{x=0,a-c}Join.JoinCPU_Rd bit, the channel FIFO with the H_CHx[x=0,a-e]Join.JoinCPU_Rd bit set, or the media FIFO with the MediaFIFOJoin.JoinCPU_Rd bit set.

During 8-bit mode, the FIFO data can be read by accessing either register, FIFO_Rd_0 or 1.

During 16-bit mode, if these registers are accessed for read while a byte boundary exists in the FIFO, valid data will be output to only one side of the registers. For details, refer to Section 6.7.2.1.5, “Processing Odd Bytes in FIFO Access.”

To read FIFO data using these registers, always be sure to read the FIFO_RdRemain_H,L registers first to confirm the number of data bytes that can be read.

7.4.28 32h FIFO_Wr_0(FIFO Write 0)

7.4.29 33h FIFO_Wr_1(FIFO Write 1)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	32h	FIFO_Wr_0	W	7: FIFO_Wr_0 [7]	Endpoint n / Channel n / Media FIFO Write	XXh
				6: FIFO_Wr_0 [6]		
				5: FIFO_Wr_0 [5]		
				4: FIFO_Wr_0 [4]		
				3: FIFO_Wr_0 [3]		
				2: FIFO_Wr_0 [2]		
				1: FIFO_Wr_0 [1]		
				0: FIFO_Wr_0 [0]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	33h	FIFO_Wr_1	W	7: FIFO_Wr_1 [7]	Endpoint n / Channel n / Media FIFO Write	XXh
				6: FIFO_Wr_1 [6]		
				5: FIFO_Wr_1 [5]		
				4: FIFO_Wr_1 [4]		
				3: FIFO_Wr_1 [3]		
				2: FIFO_Wr_1 [2]		
				1: FIFO_Wr_1 [1]		
				0: FIFO_Wr_1 [0]		

32h.Bit7-0, 33h.Bit7-0 FIFO_Wr_0 [7:0], FIFO_Wr_1 [7:0]

These registers allow data to be written into the endpoint FIFO with the D_EPx{x=0,a-c}Join.JoinCPU_Wr bit set, the channel FIFO with the H_CHx[x=0,a-e]Join.JoinCPU_Wr bit set, or the media FIFO with the MediaFIFOJoin.JoinCPU_Wr bit set.

During 8-bit mode, data can be written to the FIFO by accessing either register, FIFO_Wr_0 or 1.

During 16-bit mode, if these registers are accessed for write while a byte boundary exists in the FIFO, the data will be written to only one side of the registers. For details, refer to Section 6.7.2.1.5, "Processing Odd Bytes in FIFO Access."

To write data into the FIFO using these registers, always be sure to read the FIFO_WrRemain_H,L registers first to confirm the number of data bytes that can be written to.

7.4.30 34h FIFO_RdRemain_H (FIFO Read Remain High)

7.4.31 35h FIFO_RdRemain_L (FIFO Read Remain Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	34h	FIFO_RdRemain_H	R	7: RdRemainValid	0:None	1: Read Remain Valid	00h
				6:	0:	1:	
				5:	0:	1:	
			R	4: RdRemain [12]	Endpoint n / Channel n / Media FIFO Read Remain High		
				3: RdRemain [11]			
				2: RdRemain [10]			
				1: nRdRemain [9]			
				0: RdRemain [8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	35h	FIFO_RdRemain_L	R	7: RdRemain [7]	Endpoint n / Channel n / Media FIFO Read Remain Low	00h
				6: RdRemain [6]		
				5: RdRemain [5]		
				4: RdRemain [4]		
				3: RdRemain [3]		
				2: RdRemain [2]		
				1: RdRemain [1]		
				0: RdRemain [0]		

34h.Bit7 RdRemainValid

This bit is set to 1 when an endpoint, channel, or media is joined to the CPU I/F by the D_EPx{x=0,a-c}Join.JoinCPU_Rr, H_CHx[x=0,a-e]Join.JoinCPU_Rd, or MediaFIFOJoin.JoinCPU_Rd bit and the value of FIFO_RdRemain is valid. If this bit = 0, the value of RdRemain has no effect.

34h.Bit6-5 Reserved

34h.Bit4-0, 35h.Bit7-0 RdRemain [12:0]

These bits indicate the number of readable data bytes in the FIFO for the endpoint, channel, or media is joined to the CPU I/F by the D_EPx{x=0,a-c}Join.JoinCPU_Rr, H_CHx[x=0,a-e]Join.JoinCPU_Rd, or MediaFIFOJoin.JoinCPU_Rd bit. To get the number of readable data bytes in the FIFO, it is necessary to access the FIFO_RdRemain_H and FIFO_RdRemain_L registers in pairs. Be sure to access the FIFO_RdRemain_H register first.

7.4.32 36h FIFO_WrRemain_H (FIFO Write Remain High)

7.4.33 37h FIFO_WrRemain_L (FIFO Write Remain Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	36h	WrRemain_H		7:	0:	1:	00h
/ Host				6:	0:	1:	
				5:	0:	1:	
			R	4: WrRemain [12]	Endpoint n / Channel n / Media FIFO Write Remain High		
				3: WrRemain [11]			
				2: WrRemain [10]			
				1: WrRemain [9]			
				0: WrRemain [8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	37h	WrRemain_L	R	7: nWrRemain [7]	Endpoint n / Channel n / Media FIFO Write Remain Low	00h
				6: WrRemain [6]		
				5: WrRemain [5]		
				4: WrRemain [4]		
				3: WrRemain [3]		
				2: WrRemain [2]		
				1: WrRemain [1]		
				0: WrRemain [0]		

36h.Bit7-5 Reserved

36h.Bit4-0, 37h.Bit7-0 WrRemain [12:0]

These bits indicate the amount of free space in the FIFO for the endpoint that is connected to the CPU I/F by the D_EPx{x=0,a-c}Join.JoinCPU_Wr bit or for the channel that is connected to the CPU I/F by the H_CHx[x=0,a-e]Join.JoinCPU_Wr or MediaFIFOJoin.JoinCPU_Wr bit. An exact amount of free space in the FIFO cannot be known immediately after a write to the FIFO. Insert an interval of at least 1 CPU cycle before checking the amount of free space in the FIFO. To get the amount of free space in the FIFO, it is necessary to access the FIFO_WrRemain_H and FIFO_WrRemain_L registers in pairs. Be sure to access the FIFO_WrRemain_H register first.

7.4.34 38h FIFO_ByteRd(FIFO Byte Read)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	38h	FIFO_ByteRd	R	7: FIFO_ByteRd [7]	Endpoint n / Channel n / Media FIFO Byte Read	XXh
				6: FIFO_ByteRd [6]		
				5: FIFO_ByteRd [5]		
				4: FIFO_ByteRd [4]		
				3: FIFO_ByteRd [3]		
				2: FIFO_ByteRd [2]		
				1: FIFO_ByteRd [1]		
				0: FIFO_ByteRd [0]		

Bits7-0 FIFO_ByteRd [7:0]

These bits allow data to be read in bytes from the endpoint FIFO with the EPx{x=0,a-c}Join.JoinCPU_Rd bit set or the channel FIFO with the H_CHx[x=0,a-e]Join.JoinCPU_Rd or MediaFIFOJoin.JoinCPU_Rd bit set. To read FIFO data using this register, always be sure to read the FIFO_RdRemain_H,L registers first to confirm the number of data bytes that can be read.

7.4.35 39~3Fh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	39h -3Fh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.36 40h RAM_RdAdrs_H (RAM Read Address High)

7.4.37 41h RAM_RdAdrs_L (RAM Read Address Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	40h	RAM_RdAdrs_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: RAM Read Address [12]	RAM Read Address		
				3: RAM Read Address [11]			
				2: RAM Read Address [10]			
				1: RAM Read Address [9]			
				0: RAM Read Address [8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	41h	RAM_RdAdrs_L	R / W	7: RAM Read Address [7]	RAM Read Address	00h
				6: RAM Read Address [6]		
				5: RAM Read Address [5]		
				4: RAM Read Address [4]		
				3: RAM Read Address [3]		
				2: RAM Read Address [2]		
				1:		
				0:		

40h.Bit7-5 Reserved

40h.Bit4-0, 41h.Bit7-2 RAM_RdAdrs[12:2]

These registers are used to set the start address for RAM_Rd to be performed. After setting these registers, set the RAM_RdCount register and then the relevant bit in the RAM_RdControl register. The RAM_Rd function will be activated. While the RAM_Rd function is active, the value set in these registers changes according to the internal operation in the chip. Therefore, once the RAM_Rd function is activated by setting the relevant bit in the RAM_RdControl register, these registers should not be accessed for read until the CPU_IntStat.RAM_RdCmp bit is set. If these registers are accessed for read while the RAM_Rd function is active, the read value cannot be guaranteed. Note also that if data is written to these registers while the RAM_Rd function is active, such an operation will cause the LSI to operate erratically.

While operating in device mode, do not write 1 to RAM_RdAdrs[12]. Only RAM_RdAdrs[11:2] bits are usable.

During host mode, all of RAM_RdAdrs[12:2] are effective.

41h.Bit1-0 Reserved

7.4.38 42h RAM_RdControl (RAM Read Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	42h	RAM_RdControl	R / W	7: RAM_GoRdCBW_CSW	0: Do nothing	1: RAM Read CBW_CSW start	00h
			R / W	6: RAM_GoRd	0: Do nothing	1: RAM Read start	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

Bit7 RAM_GoRdCBW_CSW

This bit activates the RAM_Rd function to read the data that is received in the CBW area during USB device operation (ActDevice state) or the data that is received in the CSW area during USB host operation (ActHost state).

When this bit is set by writing 1 during USB device operation, the RAM_Rd function is activated, reading data from the CBW area. When the data stored in the RAM_Rd_00 through RAM_Rd_1E registers become valid, the CPU_IntStat.RAM_RdCmp bit is set to 1, at which time this bit is automatically cleared.

When this bit is set by writing 1 during USB host operation, the RAM_Rd function is activated, reading data from the CSW area. When the data stored in the RAM_Rd_00 through RAM_Rd_0C registers become valid, the CPU_IntStat.RAM_RdCmp bit is set to 1, at which time this bit is automatically cleared.

In either case, setting the RAM_RdAdrs_H,L registers and RAM_RdCount register is not necessary.

If this bit is set simultaneously with the RAM_GoRd bit, the function of this bit is given priority.

Bit6 RAM_GoRd

This bit activates the RAM_Rd function.

After setting the start address for RAM_Rd to be performed in the RAM_RdAdrs_H,L registers, set the RAM_RdCount register and write 1 to this bit to activate the RAM_Rd function. Bytes of data equal to the specified count are read beginning with the specified start address, and when the data stored in the RAM_Rd_xx{xx=00–1F} registers become valid, the CPU_IntStat.RAM_RdCmp bit is set to 1, at which time this bit is automatically cleared.

If this bit is set simultaneously with the RAM_GoRdCBW_CSW bit, the function of the RAM_GoRdCBW_CSW bit is given priority.

Bits5-0 Reserved

7.4.39 43h RAM_RdCount (RAM Read Counter)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	43h	RAM_RdCount	R / W	7:	RAM Read Counter	00h
				6:		
				5: RAM_RdCount [5]		
				4: RAM_RdCount [4]		
				3: RAM_RdCount [3]		
				2: RAM_RdCount [2]		
				1:		
				0:		

Bits7-0 RAM_RdCount [5:2]

These bits are used to set the number of data bytes to be read into the RAM_Rd_xx{xx=00–1F} registers using the RAM_Rd function. After setting the RAM_RdAdrs_H,L registers, set this register and then the relevant bit in the RAM_RdControl register to activate the RAM_Rd function. While the RAM_Rd function is active, the value of this register changes according to the internal operation in the chip. Therefore, once the RAM_Rd function is activated by setting the relevant bit in the RAM_RdControl register, this register should not be accessed for read until the CPU_IntStat.RAM_RdCmp bit is set. If this register is accessed for read while the RAM_Rd function is active, the read value cannot be guaranteed. Note also that if data is written to this register while the RAM_Rd function is active, such an operation will cause the LSI to operate erratically.

The maximum value that can be set in this register is 32 bytes. Be aware that if any number of data bytes exceeding this limit is set, the LSI will operate erratically.

7.4.40 44h RAM_WrAdrs_H (RAM Write Address High)

7.4.41 45h RAM_WrAdrs_L (RAM Write Address Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	44h	RAM_WrAdrs_H		7:		00h
				6:		
				5:		
			R / W	4: RAM_WrAdrs [12]	RAM Write Address High	
				3: RAM_WrAdrs [11]		
				2: RAM_WrAdrs [10]		
				1: RAM_WrAdrs [9]		
				0: RAM_WrAdrs [8]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	45h	RAM_WrAdrs_L	R / W	7: RAM_WrAdrs [7]	RAM Write Address Low	00h
				6: RAM_WrAdrs [6]		
				5: RAM_WrAdrs [5]		
				4: RAM_WrAdrs [4]		
				3: RAM_WrAdrs [3]		
				2: RAM_WrAdrs [2]		
				1: RAM_WrAdrs [1]		
				0: RAM_WrAdrs [0]		

These registers specify a RAM address when data is written to the RAM via the RAM_WrDoorH,L registers.

44h.Bit7-5 Reserved

44h.Bit4-0, 45h.Bit7-0 RAM_WrAdrs[12:0]

These bits specify a RAM address when data is written to the RAM. The address is incremented according to the number of bytes written to the RAM_WrDoor0,1 registers. Since exact RAM_WrAdrs cannot be known immediately after a write to the RAM_WrDoor0,1 registers, insert an interval of at least 1 CPU cycle before checking RAM_WrAdrs. For details on how to write data, refer to the section on RAM_WrDoor0,1 registers.

To inspect RAM_WrAdrs for confirmation, access the registers in order of RAM_WrAdrs_H and RAM_WrAdrs_L.

7.4.42 46h RAM_WrDoor_0 (RAM Write Door 0)

7.4.43 47h RAM_WrDoor_1 (RAM Write Door 1)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	46h	RAM_WrDoor_0	W	7: RAM_WrDoor_0 [7]	RAM Write Door 0	XXh
				6: RAM_WrDoor_0 [6]		
				5: RAM_WrDoor_0 [5]		
				4: RAM_WrDoor_0 [4]		
				3: RAM_WrDoor_0 [3]		
				2: RAM_WrDoor_0 [2]		
				1: RAM_WrDoor_0 [1]		
				0: RAM_WrDoor_0 [0]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	47h	RAM_WrDoor_1	W	7: RAM_WrDoor_1 [7]	RAM Write Door 1	XXh
				6: RAM_WrDoor_1 [6]		
				5: RAM_WrDoor_1 [5]		
				4: RAM_WrDoor_1 [4]		
				3: RAM_WrDoor_1 [3]		
				2: RAM_WrDoor_1 [2]		
				1: RAM_WrDoor_1 [1]		
				0: RAM_WrDoor_1 [0]		

46h.Bit7-0, 47h.Bit7-0 RAM_WrDoor_0 [7:0], RAM_WrDoor_1 [7:0]

These registers are a write-only register, which is used to write the data to be written to the RAM.

Before writing data to these registers, set the start address of the RAM to be written to in the RAM_WrAdrs_H,L registers. Then, when data is written to the RAM_WrDoor_0,1 registers, RAM_WrAdrs_H,L is automatically incremented according to the number of written bytes, allowing data to be written to successively.

During USB device mode, the RAM_WrDoor_0,1 registers may be used to write data for the descriptor or CSW area. The data written to the descriptor area via these registers can be used as much as necessary by the ReplyDescriptor function. In other words, this data is neither erased nor overwritten by the descriptor reply function. However, if the area to which descriptor data is written overlaps any area reserved for another endpoint, the data in it may be overwritten.

During USB host mode, the RAM_WrDoor_0,1 registers may be used to write data for the CBW area.

7.4.44 48h MediaFIFO_Control (Media FIFO Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	48h	MediaFIFO_Control		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
			W	0: MediaFIFO_Clr	0: Do nothing	1: Clear Media FIFO	

This register is used to set the operation of the Media FIFO. This is a write-only register.

Bits7-1 Reserved

Bit0 MediaFIFO_Clr

This bits clears the Media FIFO.

When this bit is set to 1, it only clears the FIFO and the value set in it is not retained.

Do not set this bit to 1 when DMAx{x=0,1} is joined to the Media FIFO and the relevant DMA is active (DMA_Running bit = 1).

7.4.45 49h ClrAllMediaFIFO_Join (Clear All Media FIFO Join)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	49h	ClrAllMediaFIFO_Join	W	7:ClrJoinIDE	0: Do nothing	1: Clear Join IDE	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
			W	3:ClrJoinDMA1	0: Do nothing	1: Clear Join DMA1	
			W	2:ClrJoinDMA0	0: Do nothing	1: Clear Join DMA0	
			W	1:ClrJoinCPU_Rd	0: Do nothing	1: Clear Join CPU_Rd	
			W	0:ClrJoinCPU_Wr	0: Do nothing	1: Clear Join CPU_Wr	

A write-only register used to clear a connection between the port selected by a bit in it and the Media FIFO.

The bits in this register area automatically cleared to 0 after a connection is cleared.

Do not set any bit in this register to 1 while the Media FIFO is connected to a port (Media FIFO_Join register's relevant bit = 1) and the port is active.

7.4.46 4Ah MediaFIFO_Join (Media FIFO Join)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	4Ah	MediaFIFO_Join	R / W	7:JoinIDE	0: Do nothing	1: Join MediaFIFO to IDE	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
			R / W	3: JoinDMA1	0: Do nothing	1: Join MediaFIFO to DMA1	
			R / W	2: JoinDMA0	0: Do nothing	1: Join MediaFIFO to DMA0	
			R / W	1: JoinCPU_Rd	0: Do nothing	1: Join MediaFIFO to CPU_Rd	
			R / W	0: JoinCPU_Wr	0: Do nothing	1: Join MediaFIFO to CPU_Wr	

This register specifies the port through which data is to be transferred to or from the Media FIFO.

Bit7 JoinIDE

Setting this bit to 1 causes a DMA1 transfer with the Media FIFO to be performed. The direction of transfer is determined by the IDE_Control.Dir bit.

Bits6-4 Reserved

Bit3 JoinDMA1

Setting this bit to 1 causes a DMA1 transfer with the Media FIFO to be performed. The direction of transfer is determined by the DMA1_Control.Dir bit.

Bit2 JoinDMA0

Setting this bit to 1 causes a DMA0 transfer with the Media FIFO to be performed. The direction of transfer is determined by the DMA0_Control.Dir bit.

Bit1 JoinCPU_Rd

Setting this bit to 1 causes a read transfer for CPU register access with the Media FIFO to be performed. In other words, when a read from the FIFO_Rd_0,1 registers or FIFO_ByteRd register is performed, data is read from the FIFO on this channel.

Bit0 JoinCPU_Wr

Setting this bit to 1 causes a write transfer for CPU register access with the Media FIFO to be performed. In other words, when a write to the FIFO_Wr_0,1 registers is performed, data is written to the FIFO at this endpoint.

If the JoinDMAx{x=0,} is set, it is possible to know the number of remaining data bytes when DMA0_Control.Dir bit = 1 or a free space when DMA0_Control.Dir bit = 0 by checking the DMAx{x=0,1}_Remain_H,L registers.

If the JoinCPU_Rd or JoinCPU_Wr bit is set, data can be read or written to or from the FIFO_Rd_0,1, FIFO_ByteRd, or FIFO_Wr_0,1 registers after checking FIFO_RdRemain_H,L or FIFO_WrRemain_H,L.

The JoinDMA1, JoinDMA0, JoinCPU_Rd, and JoinCPU_Wr bits can be set to 1 only one bit at a time. If an attempt is made to set two or more of these bits by writing 1 at the same time, only the highest bit is set.

7.4.47 4Bh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	4Bh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	

7.4.48 4C~4Fh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	4Ch -4Fh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.49 50h RAM_Rd_00 (RAM Read 00)

7.4.50 51h RAM_Rd_01 (RAM Read 01)

7.4.51 52h RAM_Rd_02 (RAM Read 02)

7.4.52 53h RAM_Rd_03 (RAM Read 03)

7.4.53 54h RAM_Rd_04 (RAM Read 04)

7.4.54 55h RAM_Rd_05 (RAM Read 05)

7.4.55 56h RAM_Rd_06 (RAM Read 06)

7.4.56 57h RAM_Rd_07 (RAM Read 07)

7.4.57 58h RAM_Rd_08 (RAM Read 08)

7.4.58 59h RAM_Rd_09 (RAM Read 09)

7.4.59 5Ah RAM_Rd_0A (RAM Read 0A)

7.4.60 5Bh RAM_Rd_0B (RAM Read 0B)

7.4.61 5Ch RAM_Rd_0C (RAM Read 0C)

7.4.62 5Dh RAM_Rd_0D (RAM Read 0D)

7.4.63 5Eh RAM_Rd_0E (RAM Read 0E)

7.4.64 5Fh RAM_Rd_0F (RAM Read 0F)

7.4.65 60h RAM_Rd_10 (RAM Read 10)

7.4.66 61h RAM_Rd_11 (RAM Read 11)

7.4.67 62h RAM_Rd_12 (RAM Read 12)

7.4.68 63h RAM_Rd_13 (RAM Read 13)

7.4.69 64h RAM_Rd_14 (RAM Read 14)

7.4.70 65h RAM_Rd_15 (RAM Read 15)

7.4.71 66h RAM_Rd_16 (RAM Read 16)

7.4.72 67h RAM_Rd_17 (RAM Read 17)

7.4.73 68h RAM_Rd_18 (RAM Read 18)

7.4.74 69h RAM_Rd_19 (RAM Read 19)

7.4.75 6Ah RAM_Rd_1A (RAM Read 1A)

7.4.76 6Bh RAM_Rd_1B (RAM Read 1B)

7.4.77 6Ch RAM_Rd_1C (RAM Read 1C)

7.4.78 6Dh RAM_Rd_1D (RAM Read 1D)

7.4.79 6Eh RAM_Rd_1E (RAM Read 1E)

7.4.80 6Fh RAM_Rd_1F (RAM Read 1F)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	50h -6Fh	RAM_Rd_00 ~ RAM_Rd_1F	R	7: RAM Read_xx [7]	RAM Read	00h
				6: RAM Read_xx [6]		
				5: RAM Read_xx [5]		
				4: RAM Read_xx [4]		
				3: RAM Read_xx [3]		
				2: RAM Read_xx [2]		
				1: RAM Read_xx [1]		
				0: RAM Read_xx [0]		

50h-6Fh.Bit7-0 RAM_Rd_xx [7:0]

These registers are used to store the data that is read from the RAM using the RAM_Rd function. Set the RAM_RdAdrs_H,L registers and the RAM_RdCount register, and then activate the RAM_Rd function using the relevant bit in the RAM_RdControl register. When the data in these registers becomes valid, the FIFO_IntStat.RAM_RdCmp bit is set to 1. If the value set in the RAM_RdCount register is less than 32 bytes, the data read from the RAM is stored in these registers sequentially beginning with RAM_Rd_00. The data stored in the registers exceeding the count of data bytes set in the RAM_RdCount register (e.g., if the count = 16, those stored in RAM_Rd_10 through RAM_Rd_1F) are ignored.

7.4.81 70h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	70h	Reserved		7:	0:	1	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.82 71h DMA0_Config (DMA0 Config)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	71h	DMA0_Config	R / W	7: FreeRun	0: Count mode	1: FreeRun mode	00h
			R / W	6: DMA_Mode	0: Normal mode	1: Address Decode mode	
				5:	0:	1:	
				4:	0:	1:	
			R / W	3: ActiveDMA	0: DMA0 Active	1: DMA0 Inactive	
				2:	0:	1:	
			R / W	1: ReqAssertCount [1]	Request Assert Count		
				0: ReqAssertCount [0]			

This register is used to set the operation mode of DMA0.

Bit7 FreeRun

This bit sets DMA0 mode.

0: Count mode

1: Free-running mode

Bit6 DMA_Mode

This bit sets DMA0 mode.

0: The DMA operates in response to XDACK from the host as acknowledge.

1: The DMA operates in response to an access to the DMA0_RdData/DMA0_WrData register from the host as acknowledge.

Bits5-4 Reserved

Bit3 ActiveDMA

This bit enables DMA0.

0: Enables DMA0.

1: Disables DMA0.

Bit2 Reserved

Bits1-0 ReqAssertCount [1:0]

These bits set the REQ Assert Count option provided to support burst read/writes by the CPU.

Set an assert count for XDREQ0 (number of transfer bytes). If the FIFO has a writable free space or readable valid data greater than the set assert count, XDREQ0 will be asserted.

When DMA transfers for bytes equal to the set assert count is complete, XDREQ0 is temporarily negated, and when the FIFO is confirmed to have a writable free space or readable valid data greater than the set assert count again, XDREQ0 is reasserted.

This means that when XDREQ0 is asserted once, transfers for bytes equal to the set assert count are guaranteed.

However, if DMA0 is set to count mode and the count of remaining bytes in DMA0_Count_HH,HL,LH,LL is smaller than the set assert count, the remaining count in DMA0_Count_HH,HL,LH,LL has priority, so that XDREQ0 is asserted when the FIFO has a writable free space or readable valid data greater than the remaining count in DMA0_Count_HH,HL,LH,LL.

The table below shows the relationship between DMA0_Count_HH,HL,LH,LL (represented by Count in the table), ReqAssertCount (represented by Req in the table), and the free space/readable data in the FIFO (represented by Ready in the table) vs. the XDREQ0 signal and the number of transferable bytes.

The remaining count in DMA0_Count_HH,HL,LH,LL must be greater than or equal to 1 for the REQ Assert Count option to be able to work.

	Count>=Req		**Count<Req**	
	Ready>=Req	**Ready<Req**	**Ready>=Count**	**Ready<Count**
XDREQ0	Asserted	Negated	Asserted	Negated
Number of transferable bytes	Req	-	Req	-

ReqAssertCount [1:0]	**Mode**	
	16bit mode	**8bit mode**
0b00	Normal	Normal
0b01	16Byte(8Count)	16Byte(16Count)
0b10	32Byte(16Count)	32Byte(32Count)
0b11	64Byte(32Count)	64Byte(64Count)

When set to 00 (= Normal), the REQ Assert Count option is unused.

7.4.83 72h DMA0_Control (DMA0 Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	72h	DMA0_Control	R	7: DMA_Running	0: DMA is not running	1: DMA is running	00h
				6:	0:	1:	
				5:	0:	1:	
			W	4: CounterClr	0: Do nothing	1: Clear DMA counter	
			R / W	3: Dir	0: CPU-IF -> FIFO RAM	1: CPU-IF <- FIFO RAM	
				2:	0:	1:	
			W	1: DMA_Stop	0: Do nothing	1: Finish DMA	
			W	0: DMA_Go	0: Do nothing	1: Start DMA	

This register controls DMA0 and shows the status of DMA0.

Bit7 DMA_Running

This bit is set to 1 and remains set while a transfer on DMA0 is underway. While this bit remains set, the EPx{x=0,a-c}Join.JoinDMA0, CHx{x=0,a-e}Join.JoinDMA0, and MediaFIFO_JoinJoinDMA0 bits cannot be overwritten.

Bits6-5 Reserved

Bit4 CounterClr

Setting this bit to 1 clears the DMA0_Count_HH,HL,LH,LL registers to 0x00. A write to this bit is ignored while the DMA_Running bit = 1.

Bit3 Dir

This bit sets the transfer direction of DMA0.

0: CPU IF to FIFO RAM (DMA write)

1: FIFO RAM to CPU IF (DMA read)

Bit2 Reserved

Bit1 DMA_Stop

Setting this bit to 1 stops the transfer on DMA0. When the transfer on DMA0 stops, the DMA_Running bit is cleared to 0. Furthermore, the DMA0_Cmp bit in the CPU_IntStat register is set to 1. To restart a transfer on DMA0, check the DMA_Running or the DMA0_Cmp bit to confirm the status and wait until the DMA terminates.

Bit0 DMA_Go

Setting this bit to 1 starts a transfer on DMA0.

7.4.84 73h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	73h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.85 74h DMA0_Remain_H (DMA0 FIFO Remain High)

7.4.86 75h DMA0_Remain_L (DMA0 FIFO Remain Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	74h	DMA0_Remain_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R	4: DMA_Remain [12]	DMA FIFO Remain High		
				3: DMA_Remain [11]			
				2: DMA_Remain [10]			
				1: DMA_Remain [9]			
				0: DMA_Remain [8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	75h	DMA0_Remain_L	R	7: DMA_Remain [7]	DMA FIFO Remain Low	00h
				6: DMA_Remain [6]		
				5: DMA_Remain [5]		
				4: DMA_Remain [4]		
				3: DMA_Remain [3]		
				2: DMA_Remain [2]		
				1: DMA_Remain [1]		
				0: DMA_Remain [0]		

74h.Bit7-5 Reserved

74h.Bit4-0, 75h.Bit7-0 DMA_Remain [12:0]

For read, these bits indicate the number of data bytes remaining in the FIFO for the endpoint connected to the DMA by the EPx{x=0,a-c}Join.JoinDMA0 bit, the channel connected to the DMA by the CHx{x=0,a-e}Join.JoinDMA0 bit, or the Media FIFO connected to the DMA by the MediaFIFO_Join.JoinDMA0 bit.

For write, these bits indicate the amount of free space remaining in the FIFO for the endpoint connected to the DMA by the EPx{x=0,a-c}Join.JoinDMA0 bit, the channel connected to the DMA by the CHx{x=0,a-e}Join.JoinDMA0 bit, or the Media FIFO connected to the DMA by the MediaFIFO_Join.JoinDMA0 bit. Since an exact amount of free space in the FIFO cannot be known through these registers immediately after a write to the DMA, insert an interval of at least 1 CPU cycle before checking the amount of free space in the FIFO.

To read these registers, access them in order of DMA0_Remain_H and L.

7.4.87 76h~77h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	76h -77h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.88 78h DMA0_Count_HH (DMA0 Transfer Byte Counter High/High)

7.4.89 79h DMA0_Count_HL (DMA0 Transfer Byte Counter High/Low)

7.4.90 7Ah DMA0_Count_LH (DMA0 Transfer Byte Counter Low/High)

7.4.91 7Bh DMA0_Count_LL (DMA0 Transfer Byte Counter Low/Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	78h	DMA0_Count_HH	R / W	7: DMA_Count [31]	DMA Transfer Byte Counter High-HIgh	00h
				6: DMA_Count [30]		
				5: DMA_Count [29]		
				4: DMA_Count [28]		
				3: DMA_Count [27]		
				2: DMA_Count [26]		
				1: DMA_Count [25]		
				0: DMA_Count [24]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	79h	DMA0_Count_HL	R / W	7: DMA_Count [23]	DMA Transfer Byte Counter High-Low	00h
				6: DMA_Count [22]		
				5: DMA_Count [21]		
				4: DMA_Count [20]		
				3: DMA_Count [19]		
				2: DMA_Count [18]		
				1: DMA_Count [17]		
				0: DMA_Count [16]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	7Ah	DMA0_Count_LH	R / W	7: DMA_Count [15]	DMA Transfer Byte Counter Low-High	00h
				6: DMA_Count [14]		
				5: DMA_Count [13]		
				4: DMA_Count [12]		
				3: DMA_Count [11]		
				2: DMA_Count [10]		
				1: DMA_Count [9]		
				0: DMA_Count [8]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	7Bh	DMA0_Count_LL	R / W	7: DMA_Count [7]	DMA Transfer Byte Counter Low-Low	00h
				6: DMA_Count [6]		
				5: DMA_Count [5]		
				4: DMA_Count [4]		
				3: DMA_Count [3]		
				2: DMA_Count [2]		
				1: DMA_Count [1]		
				0: DMA_Count [0]		

These counter registers are used to set the data length in bytes for a transfer on DMA0 during count mode. The data length can be set for up to 0xFFFF_FFFF bytes. The counter starts counting down from the set value. After setting transfer bytes in these registers, set the DMA0_Control.DMA_Go bit to 1 to start a DMA transfer. When transfers for the transfer bytes set in these registers is complete, the DMA transfer is terminated.

During free-running mode, the counter counts up from the set value. When the value in the DMA0_Count_HH,HL,LH,LL registers overflows, the DMA0_CountUp bit in the CPU_IntStat register is set to 1. The counter continues counting even after an overflow. In this mode, the number of bytes that have been DMA transferred can be inspected.

Since an exact byte count cannot be known through these registers immediately after a write to the DMA, insert an interval of at least 1 CPU cycle before checking the byte count. To read these registers, access them in order of DMA0_Count_HH, HL, LH, and LL.

7.4.92 7Ch DMA0_RdData_0 (DMA0 Read Data 0)

7.4.93 7Dh DMA0_RdData_1 (DMA0 Read Data 1)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	7Ch	DMA0_RdData_0	R	7: DMA_RdData_0 [7]	DMA Read Data 0	XXh
				6: DMA_RdData_0 [6]		
				5: DMA_RdData_0 [5]		
				4: DMA_RdData_0 [4]		
				3: DMA_RdData_0 [3]		
				2: DMA_RdData_0 [2]		
				1: DMA_RdData_0 [1]		
				0: DMA_RdData_0 [0]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	7Dh	DMA0_RdData_1	R	7: DMA_RdData_1 [7]	DMA Read Data 1	XXh
				6: DMA_RdData_1 [6]		
				5: DMA_RdData_1 [5]		
				4: DMA_RdData_1 [4]		
				3: DMA_RdData_1 [3]		
				2: DMA_RdData_1 [2]		
				1: DMA_RdData_1 [1]		
				0: DMA_RdData_1 [0]		

7Ch.Bit7-0, 7Dh.Bit7-0 DMA_RdData_0 [7:0], DMA_RdData_1 [7:0]

By accessing these registers when the DMA0_Config.DMA_Mode bit = 1, it is possible to read data from the FIFO for the endpoint, channel, or Media FIFO connected to the DMA by the EPx{x=0,a-c}Join.JoinDMA0, CHx{x=0,a-e}Join.JoinDMA0, or MediaFIFO_Join.JoinDMA0 bit. Before this operation can be performed, the DMA0_Control.Dir bit must be set for DMA read.

When operating in 8-bit mode, this DMA access can be achieved by accessing either DMA0_RdData_0 or DMA0_RdData_1.

7.4.94 7Eh DMA0_WrData_0 (DMA0 Write Data 0)

7.4.95 7Fh DMA0_WrData_1 (DMA0 Write Data 1)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	7Eh	DMA0_WrData_0	W	7: DMA_WrData_0 [7]	DMA Write Data 0	XXh
				6: DMA_WrData_0 [6]		
				5: DMA_WrData_0 [5]		
				4: DMA_WrData_0 [4]		
				3: DMA_WrData_0 [3]		
				2: DMA_WrData_0 [2]		
				1: DMA_WrData_0 [1]		
				0: DMA_WrData_0 [0]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	7Fh	DMA0_WrData_1	W	7: DMA_WrData_1 [7]	DMA Write Data 1	XXh
				6: DMA_WrData_1 [6]		
				5: DMA_WrData_1 [5]		
				4: DMA_WrData_1 [4]		
				3: DMA_WrData_1 [3]		
				2: DMA_WrData_1 [2]		
				1: DMA_WrData_1 [1]		
				0: DMA_WrData_1 [0]		

7Eh.Bit7-0, 7Fh.Bit7-0 DMA_WrData_0 [7:0], DMA_WrData_1 [7:0]

By accessing these registers when the DMA0_Config.DMA_Mode bit = 1, it is possible to write data into the FIFO for the endpoint, channel, or Media FIFO connected to the DMA by the EPx{x=0,a-c}Join.JoinDMA0, CHx{x=0,a-e}Join.JoinDMA0, or MediaFIFO_Join.JoinDMA0 bit. Before this operation can be performed, the DMA0_Control.Dir bit must be set for DMA write.

When operating in 8-bit mode, this DMA access can be achieved by accessing either DMA0_WrData_0 or DMA0_WrData_1.

7.4.96 80h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	80h	reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.97 81h DMA1_Config (DMA0 Config)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	81h	DMA1_Config	R / W	7: FreeRun	0: Count mode	1: FreeRun mode	00h
			R / W	6: DMA_Mode	0: Normal mode	1: Address Decode mode	
				5:	0:	1:	
				4:	0:	1:	
			R / W	3: ActiveDMA	0: DMA1 Active	1: DMA1 Inactive	
				2:	0:	1:	
			R / W	1: ReqAssertCount [1]	Request Assert Count		
				0: ReqAssertCount [0]			

This register is used to set the operation mode of DMA1.

Bit7 FreeRun

This bit sets the operation mode of DMA1.

0: Count mode

1: Free-running mode

Bit6 DMA_Mode

This bit sets DMA1 mode.

0: The DMA operates in response to XDACK from the host as acknowledge.

1: The DMA operates in response to an access to the DMA0_RdData/DMA0_WrData register from the host as acknowledge.

Bits5-4 Reserved

Bit3 ActiveDMA

This bit enables DMA1.

0: Enables DMA1.

1: Disables DMA1.

Bit2 Reserved

Bits1-0 ReqAssertCount [1:0]

These bits set the REQ Assert Count option provided to support burst read/writes by the CPU.

Set an assert count for XDREQ0 (number of transfer bytes). If the FIFO has a writable free space or readable valid data greater than the set assert count, XDREQ0 will be asserted.

When DMA transfers for bytes equal to the set assert count is complete, XDREQ0 is temporarily negated, and when the FIFO is confirmed to have a writable free space or readable valid data greater than the set assert count again, XDREQ0 is reasserted.

This means that when XDREQ0 is asserted once, transfers for bytes equal to the set assert count are guaranteed.

However, if DMA1 is set to count mode and the count of remaining bytes in DMA1_Count_HH,HL,LH,LL is smaller than the set assert count, the remaining count in DMA1_Count_HH,HL,LH,LL will take priority, so that XDREQ0 is asserted when the FIFO has a writable free space or readable valid data greater than the remaining count in DMA1_Count_HH,HL,LH,LL.

The table below shows the relationship between DMA1_Count_HH,HL,LH,LL (represented by Count in the table), ReqAssertCount (represented by Req in the table), and the free space/readable data in the FIFO (represented by Ready in the table) vs. the XDREQ0 signal and the number of transferable bytes.

The remaining count in DMA1_Count_HH,HL,LH,LL must be greater than or equal to 1 for the REQ Assert Count option to be able to work.

	Count>=Req		**Count<Req**	
	Ready>=Req	**Ready<Req**	**Ready>=Count**	**Ready<Count**
XDREQ1	Asserted	Negated	Asserted	Negated
Number of transferable bytes	Req	-	Req	-

ReqAssertCount [1:0]	**Mode**	
	16bit mode	**8bit mode**
0b00	Normal	Normal
0b01	16Byte(8Count)	16Byte(16Count)
0b10	32Byte(16Count)	32Byte(32Count)
0b11	64Byte(32Count)	64Byte(64Count)

When set to 00 (= Normal), the REQ Assert Count option is unused.

7.4.98 82h DMA1_Control (DMA1 Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	82h	DMA1_Control	R	7: DMA_Running	0: DMA is not running	1: DMA is running	00h
				6:	0:	1:	
				5:	0:	1:	
			W	4: CounterClr	0: Do nothing	1: Clear DMA counter	
			R / W	3: Dir	0: CPU-IF -> FIFO RAM	1: CPU-IF <- FIFO RAM	
				2:	0:	1:	
			W	1: DMA_Stop	0: Do nothing	1: Finish DMA	
			W	0: DMA_Go	0: Do nothing	1: Start DMA	

This register controls DMA1 and shows the status of DMA1.

Bit7 DMA_Running

This bit is set to 1 and remains set while a transfer on DMA1 is underway. While this bit remains set, the EPx{x=0,a-c}Join.JoinDMA1, CHx{x=0,a-e}Join.JoinDMA1, and MediaFIFO_JoinJoinDMA1 bits cannot be rewritten.

Bits6-5 Reserved

Bit4 CounterClr

Setting this bit to 1 clears the DMA1_Count_HH,HL,LH,LL registers to 0x00. A write to this bit is ignored while the DMA_Running bit = 1.

Bit3 Dir

This bit sets the transfer direction of DMA1.

0: CPU IF to FIFO RAM (DMA write)

1: FIFO RAM to CPU IF (DMA read)

Bit2 Reserved

Bit1 DMA_Stop

Setting this bit to 1 stops the transfer on DMA1. When the transfer on DMA1 stops, the DMA_Running bit is cleared to 0. In addition, the DMA1_Cmp bit in the CPU_IntStat register is set to 1. To restart a transfer on DMA1, check the DMA_Running or the DMA1_Cmp bit to confirm the status and wait until the DMA terminates.

Bit0 DMA_Go

Setting this bit to 1 starts a transfer on DMA1.

7.4.99 83h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	83h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.100 84h DMA1_Remain_H (DMA1 FIFO Remain High)

7.4.101 85h DMA1_Remain_L (DMA1 FIFO Remain Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	84h	DMA1_Remain_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R	4: DMA_Remain [12]	DMA FIFO Remain High		
				3: DMA_Remain [11]			
				2: DMA_Remain [10]			
				1: DMA_Remain [9]			
				0: DMA_Remain [8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	85h	DMA1_Remain_L	R	7: DMA_Remain [7]	DMA FIFO Remain Low	00h
				6: DMA_Remain [6]		
				5: DMA_Remain [5]		
				4: DMA_Remain [4]		
				3: DMA_Remain [3]		
				2: DMA_Remain [2]		
				1: DMA_Remain [1]		
				0: DMA_Remain [0]		

84h.Bit7-5 Reserved

84h.Bit4-0, 85h.Bit7-0 DMA_Remain [12:0]

For read, these bits indicate the number of data bytes remaining in the FIFO for the endpoint connected to the DMA by the EPx{x=0,a-c}Join.JoinDMA1 bit, the channel connected to the DMA by the CHx{x=0,a-e}Join.JoinDMA1 bit, or the Media FIFO connected to the DMA by the MediaFIFO_Join.JoinDMA1 bit.

For write, these bits indicate the amount of free space remaining in the FIFO for the endpoint connected to the DMA by the EPx{x=0,a-c}Join.JoinDMA1 bit, the channel connected to the DMA by the CHx{x=0,a-e}Join.JoinDMA1 bit, or the Media FIFO connected to the DMA by the MediaFIFO_Join.JoinDMA1 bit. Since an exact amount of free space in the FIFO cannot be known through these registers immediately after a write to the DMA, insert an interval of at least 1 CPU cycle before checking the amount of free space in the FIFO.

To read these registers, access them in order of DMA1_Remain_H and L.

7.4.102 86h~87h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	86h -87h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.103 88h DMA1_Count_HH (DMA1 Transfer Byte Counter High/High)

7.4.104 89h DMA1_Count_HL (DMA1 Transfer Byte Counter High/Low)

7.4.105 8Ah DMA1_Count_LH (DMA1 Transfer Byte Counter Low/High)

7.4.106 8Bh DMA1_Count_LL (DMA1 Transfer Byte Counter Low/Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	88h	DMA1_Count_HH	R / W	7: DMA_Count [31]	DMA Transfer Byte Counter High-High	00h
				6: DMA_Count [30]		
				5: DMA_Count [29]		
				4: DMA_Count [28]		
				3: DMA_Count [27]		
				2: DMA_Count [26]		
				1: DMA_Count [25]		
				0: DMA_Count [24]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	89h	DMA1_Count_HL	R / W	7: DMA_Count [23]	DMA Transfer Byte Counter High-Low	00h
				6: DMA_Count [22]		
				5: DMA_Count [21]		
				4: DMA_Count [20]		
				3: DMA_Count [19]		
				2: DMA_Count [18]		
				1: DMA_Count [17]		
				0: DMA_Count [16]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	8Ah	DMA1_Count_LH	R / W	7: DMA_Count [15]	DMA Transfer Byte Counter Low-High	00h
				6: DMA_Count [14]		
				5: DMA_Count [13]		
				4: DMA_Count [12]		
				3: DMA_Count [11]		
				2: DMA_Count [10]		
				1: DMA_Count [9]		
				0: DMA_Count [8]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	8Bh	DMA1_Count_LL	R / W	7: DMA_Count [7]	DMA Transfer Byte Counter Low-Low	00h
				6: DMA_Count [6]		
				5: DMA_Count [5]		
				4: DMA_Count [4]		
				3: DMA_Count [3]		
				2: DMA_Count [2]		
				1: DMA_Count [1]		
				0: DMA_Count [0]		

88h-8Bh.Bit7-0 DMA_Count [31:0]

These counter registers are used to set the data length in bytes for a transfer on DMA1 during count mode. The data length can be set for up to 0xFFFF_FFFF bytes. The counter starts counting down from the set value. After setting transfer bytes in these registers, set the DMA1_Control.DMA_Go bit to 1 to start a DMA transfer. When transfers for the transfer bytes set in these registers is complete, the DMA transfer is terminated.

During free-running mode, the counter counts up from the set value. When the value in the DMA1_Count_HH,HL,LH,LL registers overflows, the DMA1_CountUp bit in the CPU_IntStat register is set to 1. The counter continues counting even after an overflow. In this mode, the number of bytes that have been DMA transferred can be inspected.

Since an exact byte count cannot be known through these registers immediately after a write to the DMA, insert an interval of at least 1 CPU cycle before checking the byte count. To read these registers, access them in order of DMA1_Count_HH, HL, LH, and LL.

7.4.107 8Ch DMA1_RdData_0 (DMA1 Read Data 0)

7.4.108 8Dh DMA1_RdData_1 (DMA1 Read Data 1)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	8Ch	DMA1_RdData_0	R	7: DMA_RdData_0 [7]	DMA Read Data 0	XXh
				6: DMA_RdData_0 [6]		
				5: DMA_RdData_0 [5]		
				4: DMA_RdData_0 [4]		
				3: DMA_RdData_0 [3]		
				2: DMA_RdData_0 [2]		
				1: DMA_RdData_0 [1]		
				0: DMA_RdData_0 [0]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	8Dh	DMA1_RdData_1	R	7: DMA_RdData_1 [7]	DMA Read Data 1	XXh
				6: DMA_RdData_1 [6]		
				5: DMA_RdData_1 [5]		
				4: DMA_RdData_1 [4]		
				3: DMA_RdData_1 [3]		
				2: DMA_RdData_1 [2]		
				1: DMA_RdData_1 [1]		
				0: DMA_RdData_1 [0]		

8Ch.Bit7-0, 8Dh.Bit7-0 DMA_RdData_0 [7:0], DMA_RdData_1 [7:0]

By accessing these registers when the DMA1_Config.DMA_Mode bit = 1, it is possible to read data from the FIFO for the endpoint, channel, or Media FIFO connected to the DMA by the EPx{x=0,a-c}Join.JoinDMA1, CHx{x=0,a-e}Join.JoinDMA1, or MediaFIFO_Join.JoinDMA1 bit. Before this operation can be performed, the DMA1_Control.Dir bit must be set for DMA read.

When operating in 8-bit mode, this DMA access can be achieved by accessing either DMA1_RdData_0 or DMA1_RdData_1.

7.4.109 8Eh DMA1_WrData_0 (DMA1 Write Data 0)

7.4.110 8Fh DMA1_WrData_1 (DMA1 Write Data 1)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	8Eh	DMA1_WrData_0	W	7: DMA_WrData_0 [7]	DMA Write Data 0	XXh
				6: DMA_WrData_0 [6]		
				5: DMA_WrData_0 [5]		
				4: DMA_WrData_0 [4]		
				3: DMA_WrData_0 [3]		
				2: DMA_WrData_0 [2]		
				1: DMA_WrData_0 [1]		
				0: DMA_WrData_0 [0]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	8Fh	DMA1_WrData_1	W	7: DMA_WrData_1 [7]	DMA Write Data 1	XXh
				6: DMA_WrData_1 [6]		
				5: DMA_WrData_1 [5]		
				4: DMA_WrData_1 [4]		
				3: DMA_WrData_1 [3]		
				2: DMA_WrData_1 [2]		
				1: DMA_WrData_1 [1]		
				0: DMA_WrData_1 [0]		

8Eh.Bit7-0, 8Fh.Bit7-0 DMA_WrData_0 [7:0], DMA_WrData_1 [7:0]

By accessing these registers when the DMA1_Config.DMA_Mode bit = 1, it is possible to write data into the FIFO for the endpoint, channel, or Media FIFO connected to the DMA by the EPx{x=0,a-c}Join.JoinDMA1, CHx{x=0,a-e}Join.JoinDMA1, or MediaFIFO_Join.JoinDMA1 bit. Before this operation can be performed, the DMA1_Control.Dir bit must be set for DMA write.

When operating in 8-bit mode, this DMA access can be achieved by accessing either DMA1_WrData_0 or DMA1_WrData_1.

7.4.111 90h IDE_Status (IDE Status)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	90h	IDE_Status	R	7: DMARQ	0:HDMARQ Not Asserted	1:HDMARQ Asserted	00h
			R	6: DMACK	0:XHDMACK Not Asserted	1:XHDMACK Asserted	
			R	5: INTRQ	0:HINTRQ Not Asserted	1:HINTRQ Asserted	
			R	4: IORDY	0:HIORDY Not Asserted	1:HIORDY Asserted	
				3:	0:	1:	
				2:	0:	1:	
			R	1: PDIAG	0:xHPDIAG Not Asserted	1:xHPDIAG Asserted	
			R	0: DASP	0:xHDASP Not Asserted	1:xHDASP Asserted	

This register shows the signal status of the IDE bus. When any signal is asserted, the corresponding bit in this register reads “1.”

Be aware that for the active-low signals XHDMACK, XHPDIAG, and XHDASP, each corresponding bit in this register reads “1” when the voltage level is 0.

Bits 3-2 is reserved and always reads “0”.

7.4.112 91h IDE_Control (IDE Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	91h	IDE_Control		7:	0:	1:	00h
			W	6: IDE_Clr	0: None	1: Clear IDE Circuit	
				5:	0:	1:	
				4:	0:	1:	
			R/W	3: Dir	0: IDE -> FIFO RAM	1: IDE <- FIFO RAM	
				2:	0:	1:	
				1:	0:	1:	
			R/W	0: IDE_Go	0: None	1: IDE DMA Go	

This register controls the DMA for the IDE.

Bit7 Reserved

Bit6 IDE_Clr

Setting this bit initializes the IDE circuit. The contents of the IDE registers that have been set do not change. Do not set this bit while a DMA operation or a register access sequence for the IDE is underway.

Bits5-4 Reserved

Bit3 Dir

This bit sets the direction of IDE transfer.

0: IDE to FIFO RAM (IDE read)

1: FIFO RAM to IDE (IDE write)

Bits2-1 Reserved

Bit0 IDE_Go

Setting this bit to 1 starts a DMA operation for the IDE. When the DMA operation is complete, the IDE_Cmp bit in the IDE_IntStat register is set to 1.

This bit is set to 1 and remains set during a DMA transfer, and is reset to 0 when the transfer is complete.

If this bit is cleared by writing 0 while it is set to 1, the DMA transfer underway is aborted and thereby terminated, in which case the IDE_Cmp bit in the IDE_IntStat register is not set.

7.4.113 92h IDE_Config_0 (IDE Configuration 0)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	92h	IDE_Config_0	R/W	7: IDE_BusReset	0: None	1: XHRESET Asserted	00h
			R/W	6: IDE_LongBusReset	0: None	1: XHRESET Asserted	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R/W	1: Ultra	0: Non Ultra mode	1: Ultra mode	
			R/W	0: DMA	0: Non DMA mode	1: DMA mode	

This register controls the DMA for the IDE.

Bit7 IDE_BusReset

Setting this bit to 1 causes the xHRESET signal of the IDE to be asserted for 50 µs. If this bit is set again by writing 1 while it shows the value 1, the XHRESET signal of the IDE is asserted for another period of 50 µs from that point in time. When the XHRESET signal is asserted by either bit 7 or bit 6, this bit reads “1.”

Bit6 IDE_LongBusReset

Setting this bit to 1 causes the XHRESET signal of the IDE to be asserted for 400 µs. If this bit is set again by writing 1 while it shows the value 1, the XHRESET signal of the IDE is asserted for another period of 400 µs from that point in time. When the XHRESET signal is asserted by either bit 7 or bit 6, this bit reads “1.”

Bits5-2 Reserved

Bit1 Ultra

When this bit is set to 1 at the same time with bit 0 (= DMA), the DMA for the IDE activated by the IDE_Control register is placed in Ultra mode.

This bit cannot be rewritten while a DMA transfer is underway. The table below lists DMA transfer modes of the IDE that are set by a combination of bits 1–0.

Bit0 DMA

When this bit is set to 1, the DMA for the IDE activated by the IDE_Control register is placed in Multiword DMA mode.

This bit cannot be rewritten while a DMA transfer is underway. The table below lists DMA transfer modes of the IDE that are set by a combination of bits 1–0.

Bit1-0	“00”	“01”	“10”	“11”
	PIO	Multiword DMA	Settings prohibited	Ultra

7.4.114 93h IDE_Config_1 (IDE Configuration 1)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	93h	IDE_Config_1	R/W	7: ActiveIDE	0: InActivated IDE Bus	1: Activate IDE Bus	04h
			R/W	6: DelayStrobe	0: Not Delay Strobe Signal	1: Delay Strobe Signal	
				5:	0:	1:	
			R/W	4: InterLock	0: None	1: DMA InterLock	
				3:	0:	1:	
			R/W	2:	0: None	1: Data Swap	
				1:	0:	1:	
				0:	0:	1:	

This register controls the status of the IDE bus.

Bit7 ActiveIDE

Setting this bit to 1 enables the output signals of the IDE. Before a register read/write command for the IDE bus can be issued or an IDE-DMA can be executed, this bit must first be set to 1. When this bit = 0, all IDE signals are set for input.

Bit6 DelayStrobe

If this bit is set to 1, a setup time is held on for 2 system clock periods (approx. 33 ns) before the XHIOR/XHIOW strobe signal is asserted after the assertion of XHDMACK during a Multiword DMA transfer of IDE-DMA. If this bit = 0, the XHIOR/XHIOW strobe signal is asserted simultaneously with the assertion of XHDMACK (approx. 0 ns) during a Multiword DMA transfer of IDE-DMA.

Bit5 Reserved

Bit4 InterLock

If this bit is set to 1, XHDMACK is not negated for reasons that the LSI internally became unable to transfer data during a Multiword DMA transfer of IDE-DMA; instead, the IDE bus is retained active until data is prepared internally in the LSI. When this bit = 0, XHDMACK is temporarily released unless data cannot be prepared internally.

Bit3 Reserved

Bit2 Reserved

Bits1-0 Reserved

7.4.115 94h IDE_Rmod (IDE Register Mode)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	94h	IDE_Rmod	R/W	7: RegisterAssertPulseWidth [3]	Register Assert Pulse Width	00h
				6: RegisterAssertPulseWidth [2]		
				5: RegisterAssertPulseWidth [1]		
				4: RegisterAssertPulseWidth [0]		
			R/W	3: RegisterNegatePulseWidth [3]	Register Negate Pulse Width	
				2: RegisterNegatePulseWidth [2]		
				1: RegisterNegatePulseWidth [1]		
				0: RegisterNegatePulseWidth [0]		

This register is used to set the strobe width in which time XHIOR/XHIOW are asserted or negated when accessing the IDE bus in register mode via the CPU.

The appropriate value suitable for transfer mode of the IDE must be selected.

Bits7-4 RegisterAssertPulseWidth [3:0]

The system clock (60 MHz) period multiplied by "RegisterAssertPulseWidth + 4" comprises the strobe width.

ex 0000: $4 \times 16.67\text{nS} = 67\text{nS}$

0001: $5 \times 16.67\text{nS} = 83\text{nS}$

Bits3-0 RegisterNegatePulseWidth [3:0]

The system clock (60 MHz) period multiplied by "RegisterNegatePulseWidth + 4" comprises the strobe width.

ex 0000: $4 \times 16.67\text{nS} = 67\text{nS}$

0001: $5 \times 16.67\text{nS} = 83\text{nS}$

7.4.116 95h IDE_Tmod (IDE Transfer Mode)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	95h	IDE_Tmod	R/W	7: TransferAssertPulseWidth [3]	Transfer Assert Pulse Width	00h
				6: TransferAssertPulseWidth [2]		
				5: TransferAssertPulseWidth [1]		
				4: TransferAssertPulseWidth [0]		
			R/W	3: TransferNegatePulseWidth [3]	Transfer Negate Pulse Width	
				2: TransferNegatePulseWidth [2]		
				1: TransferNegatePulseWidth [1]		
				0: TransferNegatePulseWidth [0]		

This register is used to set the strobe width in which time XHIOR/XHIOW are asserted or negated when accessing the IDE bus in PIO mode during CPU and DMA transfers.

The appropriate value suitable for transfer mode of the IDE must be selected.

Bits7-4 TransferAssertPulseWidth [3:0]

The system clock (60 MHz) period multiplied by "TransferAssertPulseWidth + 4" comprises the strobe width.

ex 0000: $4 \times 16.67\text{nS} = 67\text{nS}$

0001: $5 \times 16.67\text{nS} = 83\text{nS}$

Bits3-0 TransferNegatePulseWidth [3:0]

The system clock (60 MHz) period multiplied by "TransferNegatePulseWidth + 4" comprises the strobe width.

ex 0000: $4 \times 16.67\text{nS} = 67\text{nS}$

0001: $5 \times 16.67\text{nS} = 83\text{nS}$

7.4.117 96h IDE_Umod (IDE Ultra-DMA Transfer Mode)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	96h	IDE_Umod		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
			R/W	3: UltraDMA_Cycle [3]	UltraDMA_Cycle		
				2: UltraDMA_Cycle [2]			
				1: UltraDMA_Cycle [1]			
				0: UltraDMA_Cycle [0]			

This register is used to set the access cycle width in which time access to the IDE bus in Ultra mode is performed during a DMA transfer.

The appropriate value suitable for transfer mode of the IDE must be selected.

Bits7-4 Reserved

Bits3-0 UltraDMA_Cycle [3:0]

The system clock (60 MHz) period multiplied by “UltraDMA_Cycle + 4” comprises the access cycle width.

ex 0000: $2 \times 16.67\text{nS} = 33\text{nS}$

0001: $3 \times 16.67\text{nS} = 50\text{nS}$

7.4.118 97h~99h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	97h -99h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.119 9Ah IDE_CRC_H (IDE CRC High)

7.4.120 9Bh IDE_CRC_L (IDE CRC Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	9Ah	IDE_CRC_H	R/W	7: IDE_CRC [15]	IDE_CRC [15:8]	00h
				6: IDE_CRC [14]		
				5: IDE_CRC [13]		
				4: IDE_CRC [12]		
				3: IDE_CRC [11]		
				2: IDE_CRC [10]		
				1: IDE_CRC [9]		
				0: IDE_CRC [8]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	9Bh	IDE_CRC_L	R/W	7: IDE_CRC [7]	IDE_CRC [7:0]	00h
				6: IDE_CRC [6]		
				5: IDE_CRC [5]		
				4: IDE_CRC [4]		
				3: IDE_CRC [3]		
				2: IDE_CRC [2]		
				1: IDE_CRC [1]		
				0: IDE_CRC [0]		

These registers show the CRC calculation result during a DMA transfer in Ultra mode of the IDE each time CRC is calculated. To read the CRC calculation result, the IDE_CRC_H and IDE_CRC_L registers must be accessed in pairs. At that time, be sure to access the IDE_CRC_H register first.

7.4.121 9Ch Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	9Ch	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.122 9Dh IDE_Count_H (IDE Transfer Byte Counter High)

7.4.123 9Eh IDE_Count_M (IDE Transfer Byte Counter Middle)

7.4.124 9Fh IDE_Count_L (IDE Transfer Byte Counter Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	9Dh	IDE_Count_H	R/W	7: IDE_Count [23]	IDE_Count [23:16]	00h
				6: IDE_Count [22]		
				5: IDE_Count [21]		
				4: IDE_Count [20]		
				3: IDE_Count [19]		
				2: IDE_Count [18]		
				1: IDE_Count [17]		
				0: IDE_Count [16]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	9Eh	IDE_Count_M	R/W	7: IDE_Count [15]	IDE_Count [15:8]	00h
				6: IDE_Count [14]		
				5: IDE_Count [13]		
				4: IDE_Count [12]		
				3: IDE_Count [11]		
				2: IDE_Count [10]		
				1: IDE_Count [9]		
				0: IDE_Count [8]		

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	9Fh	IDE_Count_L	R/W	7: IDE_Count [7]	IDE_Count [7:1]		00h
				6: IDE_Count [6]			
				5: IDE_Count [5]			
				4: IDE_Count [4]			
				3: IDE_Count [3]			
				2: IDE_Count [2]			
				1: IDE_Count [1]			
				0:	0:	1:	

These registers are used to set the number of transfer bytes for a DMA transfer of the IDE. If an attempt is made to start the DMA while 0 bytes are set in these registers, the attempted startup is ignored. To read transfer bytes, be sure to access IDE_Count_H and IDE_Count_L in pairs. At that time, access IDE_Count_H first. Note that the lowest bit in the IDE_Count_L register always reads "0."

7.4.125 A0h IDE_RegAdrs (IDE Register Address)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	A0h	IDE_RegAdrs	R/W	7: IDE_WrReg	0: None	1: IDE Register Write Go	00h
			R/W	6: IDE_RdReg	0: None	1: IDE Register Read Go	
				5:	0:	1:	
				4:	0:	1:	
			R/W	3: IDE_RegAddress [3]	IDE_RegAddress [3:0]		
				2: IDE_RegAddress [2]			
				1: IDE_RegAddress [1]			
				0: IDE_RegAddress [0]			

This register controls register accesses to the IDE bus by the CPU.

Bit7 IDE_WrReg

When this bit is set to 1, the LSI writes to the IDE task file registers in PIO or register mode for write to the IDE bus using the contents preset in the IDE_WrRegValue_H/L registers. During operation this bit reads "1" and when the operation is complete, the IDE_RegCmp bit in the IDE_IntStat register is set and this bit is reset to 0. The address for the IDE bus must be set in IDE_RegAddress in advance or simultaneously with a write. Note that the IDE_Rmod and IDE_Tmod registers must be set to the appropriate mode in advance.

Bit6 IDE_RdReg

When this bit is set to 1, the LSI reads the IDE task file registers in PIO or register mode for read from the IDE bus and sets the read value in the IDE_RdRegValue_H/L registers. During operation this bit reads "1" and when the operation is complete, the IDE_RegCmp bit in the IDE_IntStat register is set and this bit is reset to 0. The address for the IDE bus must be set in IDE_RegAddress in advance or simultaneously with a read. Furthermore, the IDE_Rmod and IDE_Tmod registers must be set to the appropriate mode in advance.

Bits5-4 Reserved

Bits3-0 IDE_RegAddress [3:0]

These bits are used to set the address for a register access to the IDE bus by the CPU that is initiated by setting the IDE_WrReg or IDE_RdReg bit. The table below shows the relationship between these bits and the addresses output to the IDE bus.

IDE_RegAddress [3]	0:XHCS0=0	1:XHCS1=0
IDE_RegAddress [2]	0:HDA2=0	1:HDA2=1
IDE_RegAddress [1]	0:HDA1=0	1:HDA1=1
IDE_RegAddress [0]	0:HDA0=0	1:HDA0=1

7.4.126 A1h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	A1h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.127 A2h IDE_RdRegValue_0 (IDE Register Read Value 0)

7.4.128 A3h IDE_RdRegValue_1 (IDE Register Read Value 1)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	A2h	IDE_RdRegValue_0	R	7: IDE_RdRegValue_0 [7]	IDE_RdRegValue 0	00h
				6: IDE_RdRegValue_0 [6]		
				5: IDE_RdRegValue_0 [5]		
				4: IDE_RdRegValue_0 [4]		
				3: IDE_RdRegValue_0 [3]		
				2: IDE_RdRegValue_0 [2]		
				1: IDE_RdRegValue_0 [1]		
				0: IDE_RdRegValue_0 [0]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	A3h	IDE_RdRegValue_1	R	7: IDE_RdRegValue_1 [7]	IDE_RdRegValue 1	00h
				6: IDE_RdRegValue_1 [6]		
				5: IDE_RdRegValue_1 [5]		
				4: IDE_RdRegValue_1 [4]		
				3: IDE_RdRegValue_1 [3]		
				2: IDE_RdRegValue_1 [2]		
				1: IDE_RdRegValue_1 [1]		
				0: IDE_RdRegValue_1 [0]		

These registers are used to store the value that is read from the IDE bus via an IDE register read by the CPU that was initiated by setting the IDE_RdReg bit in the IDE_RegAdrs register. Furthermore, the value read by an auto status register read that is initiated by setting the IDE_RegConfig register is stored in these registers. To read these registers, be sure to access IDE_RdRegValue_0 and IDE_RdRegValue_1 in pairs. At that time, access IDE_RdRegValue_0 first.

7.4.129 A4h IDE_WrRegValue_0 (IDE Register Write Value 0)

7.4.130 A5h IDE_WrRegValue_1 (IDE Register Write Value 1)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	A4h	IDE_WrRegValue_0	R/W	7: IDE_WrRegValue_0 [7]	IDE_WrRegValue 0	00h
				6: IDE_WrRegValue_0 [6]		
				5: IDE_WrRegValue_0 [5]		
				4: IDE_WrRegValue_0 [4]		
				3: IDE_WrRegValue_0 [3]		
				2: IDE_WrRegValue_0 [2]		
				1: IDE_WrRegValue_0 [1]		
				0: IDE_WrRegValue_0 [0]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	A5h	IDE_WrRegValue_1	R/W	7: IDE_WrRegValue_1 [7]	IDE_WrRegValue 1	00h
				6: IDE_WrRegValue_1 [6]		
				5: IDE_WrRegValue_1 [5]		
				4: IDE_WrRegValue_1 [4]		
				3: IDE_WrRegValue_1 [3]		
				2: IDE_WrRegValue_1 [2]		
				1: IDE_WrRegValue_1 [1]		
				0: IDE_WrRegValue_1 [0]		

These registers are used to set in advance the data to be written to the IDE bus via an IDE register write by the CPU that is initiated by enabling the IDE_WrReg bit in the IDE_RegAdrs register.

7.4.131 A6h IDE_SeqWrRegControl (IDE Sequential Register Write Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	A6h	IDE_SeqWrRegControl	R/W	7: IDE_SeqWrReg	0:	1: IDE Sequence Write Go	00h
			W	6: IDE_SeqWrRegClr	0:	1: Clear IDE Sequence Write	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register controls a sequential register write to the IDE bus by the CPU.

Bit7 IDE_SeqWrReg

When this bit is set to 1, up to 16 pairs of the addresses and data preset in IDE_SeqWrRegAdrs and IDE_SeqWrRegValue are written to the IDE bus via IDE task file registers in the order in which they were set. When the operation is complete, the IDE_SeqWrRegCmp bit in the IDE_IntStat register is set to 1.

This bit remains set to 1 during a sequential write operation, and is reset to 0 when the operation is complete.

Bit6 IDE_SeqWrRegClr

When this bit is set to 1, up to 16 pairs of the addresses and data preset in IDE_SeqWrRegAdrs and IDE_SeqWrRegValue are discarded and the registers can thereby be initialized. This bit cannot be set during a sequential write operation.

Bits5-0 Reserved

7.4.132 A7h IDE_SeqWrRegCnt (IDE Sequential Register Write Counter)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	A7h	IDE_SeqWrRegCnt		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R	4: IDE_SeqWrRegCnt [4]	IDE_SeqWrRegCnt [4:0]		
				3: IDE_SeqWrRegCnt [3]			
				2: IDE_SeqWrRegCnt [2]			
				1: IDE_SeqWrRegCnt [1]			
				0: IDE_SeqWrRegCnt [0]			

This register shows the number of data bytes written in the IDE_SeqWrRegValue register. A value of up to 10h is indicated. The value in this register is decremented as a sequential write to the IDE bus proceeds, and when all bytes of data written in the IDE_SeqWrRegValue register are written to the IDE bus, it is reset to 0. The value is also reset to 0 by writing 1 to the IDE_SeqWrRegClr bit in the IDE_SeqWrRegControl register.

7.4.133 A8h IDE_SeqWrRegAdrs (IDE Sequential Register Write Address FIFO)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	A8h	IDE_SeqWrRegAdrs		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
			W	3: IDE_SeqRegAddress [3]	IDE_SeqRegAddress [3:0]		
				2: IDE_SeqRegAddress [2]			
				1: IDE_SeqRegAddress [1]			
				0: IDE_SeqRegAddress [0]			

This register is used to set the address to be output to the IDE bus during a sequential write to the IDE bus that is initiated by setting the IDE_SeqWrRegControl register. This address must be set in pairs with the data that is set in the IDE_SeqWrRegValue register. If the addresses to be set successively in this register are the same, there is no need to set the same address again once it is set. The relationship between the addresses output to the IDE bus and the address bits in this register is the same as for the IDE_RegAddress bits in the IDE_RegAdrs register.

7.4.134 A9h IDE_SeqWrRegValue (IDE Sequential Register Write Value FIFO)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	A9h	IDE_SeqWrRegValue	W	7: IDE_SeqWrRegValue [7]	IDE_SeqWrRegValue [7:0]	XXh
				6: IDE_SeqWrRegValue [6]		
				5: IDE_SeqWrRegValue [5]		
				4: IDE_SeqWrRegValue [4]		
				3: IDE_SeqWrRegValue [3]		
				2: IDE_SeqWrRegValue [2]		
				1: IDE_SeqWrRegValue [1]		
				0: IDE_SeqWrRegValue [0]		

This register is used to set the data to be output to the IDE bus during a sequential write to the IDE bus that is initiated by setting the IDE_SeqWrRegControl register. This data must be set in pairs with the address that is set in the IDE_SeqWrRegAdrs register. Up to 16 pairs of addresses and data can be set, and write operations exceeding that maximum are ignored. When IDE_SeqWrRegAdrs = 0 (write to the data port whose XHCS = 0 and HDA = 0), the IDE is accessed in 16 bits and this register must therefore be set twice (in order of the lower byte and the upper byte of data). In that case, two pairs of addresses and data out of a maximum of 16 pairs are used. For other addresses, the IDE is accessed in 8 bits, so register needs to be set once for each write to the IDE.

7.4.135 AAh~ABh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	AAh -ABh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.136 ACh IDE_RegConfig (IDE Register Configuration)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	ACh	IDE_RegConfig	R/W	7: EnAutoStsRd	0: None	1: Auto Status Read Enable	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register controls the auto status read operation in a HINTRQ interrupt of the IDE bus.

Bit7 EnAutoStsRd

When this bit is set to 1, the LSI automatically shifts to read the status register of the IDE bus (XHCS0 = 0, HDA = 7) when a HINTRQ interrupt of the IDE bus occurs. Upon completion of the operation, the LSI sets the read value in the IDE_RdRegValue register, at which time the CompleteINTRQ bit in the IDE_IntStat register is set to 1. This bit remains set even after the operation is complete.

Bits6-0 Reserved

7.4.137 ADh~AFh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	ADh -AFh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.138 B0h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	B0h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.139 B1h *HostDeviceSel (Host Device Select)*

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	B1h	***HostDeviceSel***		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
			R / W	0: ***HOSTxDEVICE***	0: Device mode	1: Host mode	

This register is used to set device or host mode of the LSI.

Bits7-1 Reserved

Bit0 HOSTxDEVICE

This bit sets device or host mode of the register map.

Even when the setting of this bit is changed, the values in all other registers of the register map are not cleared.

This bit can be accessed even during Sleep/Snooze.

7.4.140 B2h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	B2h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.141 B3h *ModeProtect(Mode Protection)*

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	B3h	***ModeProtect***	R / W	7: ***ModeProtect [7]***	Mode Protection	56h
				6: ***ModeProtect [6]***		
				5***: ModeProtect [5]***		
				4: ***ModeProtect [4]***		
				3: ***ModeProtect [3]***		
				2: ***ModeProtect [2]***		
				1: ***ModeProtect [1]***		
				0: ***ModeProtect [0]***		

Bits7-0 ModeProtect [7:0]

This register protects the values of the ChipConfig register and ClkSelect.ClkSelect bit. Writing "56h" to this register removes protection, allowing the ChipConfig register and ClkSelect.ClkSelect bit to be accessed for write.

During normal use, after setting the ChipConfig register and ClkSelect.ClkSelect bit to any value, write other than "56h" (e.g., 00h) to this register to protect the ChipConfig register and ClkSelect.ClkSelect bit.

7.4.142 B4h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	B4h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.143 B5h *ClkSelect (Clock Select)*

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	B5h	***ClkSelect***	R / W	7: ***xActIDE_Term***	0: IDE Termination ON	1: IDE Termination OFF	41h
			R / W	6: ***xActIDE_DD_Term***	0: Termination ON	1: Termination OFF	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R / W	1: ***PORT1x2***	0: 2 Port	1: 1 Port	
			R / W	0: ***ClkSelect***	0: 12MHz	1: 24MHz	

Bit7 ***xActIDE_Term***

Of the IDE port pins, this bit switches termination on or off for HDMARQ, HIORDY, HINTRQ, XHDASP, XHPDIAG, and HDD7.

This bit can be accessed even in SLEEP/SNOOZE.

0: Termination turned on

1: Termination turned off

Bit6 ***xActIDE_DD_Term***

Of the IDE port pins, this bit switches termination on or off for HDD15–HDD8 and HDD6–HDD0.

This bit can be accessed even in SLEEP/SNOOZE.

0: Termination turned on

1: Termination turned off

Bits5-2 **Reserved**

Bit1 ***PORT1x2***

This bit selects the USB port mode used in the LSI.

If the 2 Port is selected, ports A and B are assigned USB Host and USB Device, respectively. Either functionality may be selected. If the 1 Port is selected, only port B can be used, in which case either USB Host or USB Device functionality may be selected for that port B.

For more information on 1-port mode, refer to Appendix C.

This can be accessed even in SLEEP/SNOOZE.

0: 2 Port

1: 1 Port

Bit0 ***ClkSelect***

This bit selects the clock to be used for the LSI. This bit can be accessed even during Sleep/Snooze.

0: 12 MHz

1: 24 MHz

7.4.144 B6h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	B6h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.145 B7h *ChipConfig (Chip Configuration)*

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	B7h	ChipConfig	R / W	7: ***IntLevel***	0: Low Active	1: High Active	00h
			R / W	6: ***IntMode***	0: 1/0 mode	1: Hi-z/0 mode	
			R / W	5: ***DREQ_Level***	0: Low Active	1: High Active	
			R / W	4: ***DACK_Level***	0: Low Active	1: High Active	
			R / W	3: ***CS_Mode***	0: DACK mode	1: CS mode	
			R / W	2: ***CPU_Endian***	0: Do nothing	1: Bus Swap	
			R / W	1: ***BusMode***	0: XWRH/L mode	1: XBEH/L mode	
			R / W	0: ***Bus8x16***	0: 16bit mode	1: 8bit mode	

This register is used to set the operation mode of the LSI.

Bit7 ***IntLevel***

This bit sets the logic level of XINT. This bit can be accessed even during Sleep/Snooze.

0: Active low

1: Active high

Bit6 ***IntMode***

This bit sets the output mode of XINT. This bit can be accessed even during Sleep/Snooze.

0: 1/0 mode

1: Hi-z/0 mode

Bit5 **DREQ_Level**

This bit sets the logic levels of XDREQ0,1. This bit can be accessed even during Sleep/Snooze.

0: Active low

1: Active high

Bit4 ***DACK_Level***

This bit sets the logic levels of XDACK0,1. This bit can be accessed even during Sleep/Snooze.

0: Active low

1: Active high

Bit3 **CS_Mode**

This bit sets the operation mode of DMA0,1. This bit can be accessed even during Sleep/Snooze.

0: Operates as a DMA access that is enabled by the assertion of XDACK0,1

1: Operates as a DMA access that is enabled by the assertion of XCS and XDACK0,1

Bit2 CPU_Endian

This bit sets the CPU bus when operating in 16-bit mode. Setting of this bit has no effect during 8-bit mode. This bit can be accessed even during Sleep/Snooze.

0: The even and the odd addresses of the bus are used for the upper and the lower bytes, respectively.

1: The even and the odd addresses of the bus are used for the lower and the upper bytes, respectively.

Setting of this bit is made effective by reading the address B9h after writing to the register. If the circuit is reset by the ChipReset.ResetAll bit, the register value is initialized, in which case the register's set value takes effect after the address B9h is read, as in the above case.

Bits1-0 BusMode, Bus8x16

These bits set the operation mode of the CPU. These bits can be accessed even during Sleep/Snooze.

Operation mode	bit1.BusMode	bit0.Bus8x16
16bit Strobe mode	0	0
16bit BE mode	1	*
8bit mode	0	1

7.4.146 B8h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	B8h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.4.147 B9h *CPU_ChgEndian (CPU Change Endian)*

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device / Host	E9h	***CPU_ChgGndian***	R	7: ***CPU_ChgEndian[7]***	CPU Change Endian	XXh
				6: ***CPU_ChgEndian[6]***		
				5: ***CPU_ChgEndian[5]***		
				4: ***CPU_ChgEndian[4]***		
				3: ***CPU_ChgEndian[3]***		
				2: ***CPU_ChgEndian[2]***		
				1: ***CPU_ChgEndian[1]***		
				0: ***CPU_ChgEndian[0]***		

Bits7-0 Reserved

The endian set by ChipConfig.CPU_Endian is enabled by accessing this register for a dummy read.

This register can be accessed even during SLEEP/SNOOZE.

7.4.148 BAh~DFh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	BAh ~DFh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5 Detailed Description of Device Registers

7.5.1 E0h *D_SIE_IntStat (Device SIE Interrupt Status)*

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	E0h	***D_SIE_IntStat***		7:	0:	1:	00h
			R (W)	6: ***NonJ***	0: None	1: Detect Non J state	
			R (W)	5: RcvSOF	0: None	1: Received SOF	
			R (W)	4: DetectReset	0: None	1: Detect USB Reset	
			R (W)	3: DetectSuspend	0: None	1: Detect USB Suspend	
			R (W)	2: ChirpCmp	0: None	1: Chirp Complete	
			R (W)	1: RestoreCmp	0: None	1: Restore Complete	
			R (W)	0: SetAddressCmp	0: None	1: AutoSetAddress Complete	

This register shows the interrupts associated with the device SIE.

The cause of interrupt indicated by any bit in this register can be cleared by writing 1 to that bit.

Bit7 Reserved

Bit6 NonJ

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a state other than J is detected on the USB bus. This bit is effective when the LSI is in Snooze mode (PM_Control register’s InSnooze bit = 1) and when the InSUSPEND bit in the USB_Control register remains set to 1 while the AutoNegotiation function is in use.

Bit5 RcvSOF

This bit indicates the cause of interrupt directly.

This bit is set to 1 when an SOF token is received.

Bit4 DetectReset

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a reset state of the USB is detected. While this bit remains set, the suspend state of the USB cannot be detected (DetectSUSPEND is not set).

This reset detection is effective when the ActiveUSB bit in the D_NegoControl register remains set.

During HS operation mode, when no bus activity is detected for a predetermined time, the FS termination is automatically set in order to detect a reset/suspend of the USB; when SE0 is detected, a reset is assumed and this bit is set to 1.

When not using the AutoNegotiation function, it is necessary to prevent erroneous detection of subsequent resets after this bit is set to 1. Therefore, set the DisBusDetect bit in the D_NegoControl register to 1 to disable the detection of reset/suspend states of the USB. After processing for the reset is complete, clear the DisBusDetect bit to 0 to reenable the detection of reset/suspend states of the USB.

Bit3 DetectSuspend

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a suspend state of the USB is detected. While this bit remains set, the reset state of the USB cannot be detected (DetectRESET is not set).

During HS operation mode, when no bus activity is detected for a predetermined time, FS operation mode is automatically entered in order to detect a reset/suspend of the USB. After a suspend state of the USB is detected, the LSI can be placed in Snooze mode (internal PLLs turned off) by setting the GoSnooze bit in the PM_Control_0 register to 1.

Bit2 ChirpCmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when HS Detection Handshaking that was started by the GoChirp bit in the D_NegoControl register is complete.

The current operation mode (FS or HS) can be determined by reading the FSxHS bit in the D_USB_Status register after the interrupt occurred.

Bit1 RestoreCmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when the Restore processing that was started by RestoreUSB bit in the D_NegoControl register is complete. When this bit is set to 1, operation mode (FS or HS) returns to the state in which the USB was prior to Suspend.

Bit0 SetAddressCmp

This bit indicates the cause of interrupt directly.

When a SetAddress() request is received, the AutoSetAddress function (see “USB_Address register”) automatically performs processing for the control transfer needed. Then, when the control transfer for the SetAddress() request is completed by executing a status stage, this status bit is set to 1. At the same time, the address is set in the D_USB_Address register.

The synchronous bits (bits 5–0) can be read out, but cannot be written to (for clearing the cause of interrupt) during the Active60 and Act_Host state.

To exit the Act_Device state, the processing described below should be executed in firmware to ensure that the interrupt signal XINT will not inadvertently be asserted inadvertently by these interrupt statuses.

<When shifting out from the Act_Device state>

1) Process the interrupt status and then clear (D_SIE_IntStat.Bit5–0)

2) Disable the interrupt status (D_SIE_IntEnb.Bit5–0)

<When shifting to the Act_Device state>

3) Clear the interrupt status (D_SIE_IntStat.Bit5–0)

4) Reenable the interrupt status (D_SIE_IntEnb.Bit5–0)

7.5.2 E1h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device / Host	E1h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.3 E2h D_FIFO_IntStat (Device FIFO Interrupt Status)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	E2h	D_FIFO_IntStat	R (W)	7: DescriptorCmp	0: None	1: Descriptor Complete	00h
			R (W)	6: FIFO_IDE_Cmp	0: None	1: FIFO-IDE Complete	
			R (W)	5: FIFO1_Cmp	0: None	1: FIFO1 Complete	
			R (W)	4: FIFO0_Cmp	0: None	1: FIFO0 Complete	
				3:	0:	1:	
			R (W)	2: FIFO_NotEmpty	0: None	1: FIFO NotEmpty	
			R (W)	1: FIFO_Full	0: None	1: FIFO Full	
			R (W)	0: FIFO_Empty	0: None	1: FIFO Empty	

This register shows the interrupts associated with the device FIFO.

The cause of an interrupt indicated by any bit in this register can be cleared by writing 1 to that bit.

Bit7 DescriptorCmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when in the Descriptor reply function, the LSI is complete sending a number of data bytes indicated by the DescriptorSize register back to the host.

If a transition to the status stage occurs (OUT token received) before the data bytes indicated by the DescriptorSize register have all been sent back, the OUT_TranNAK bit in the D_EP0IntStat register is set to 1 along with this status bit.

Bit6 FIFO_IDE_Cmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when while the endpoint joined to the IDE is directed for IN, the FIFO is emptied after IDE transfer is completed. If the endpoint joined to the IDE is directed for OUT, this bit is set to 1 when the IDE transfer is completed.

Bit5 FIFO1_Cmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when while the endpoint joined to DMA1 is directed for IN, the FIFO is emptied after DMA1 transfer is completed. If the endpoint joined to DMA1 is directed for OUT, this bit is set to 1 when the DMA1 transfer is completed.

Bit4 FIFO0_Cmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when while the endpoint joined to DMA0 is directed for IN, the FIFO is emptied after DMA0 transfer is completed. If the endpoint joined to DMA0 is directed for OUT, this bit is set to 1 when the DMA0 transfer is completed.

Bit3 Reserved

Bit2 FIFO_NotEmpty

This bit indicates the cause of interrupt directly.

This bit is set to 1 if the FIFO area for the corresponding endpoint contains any data (NotEmpty) while the EPx{x=0,a-c}Join.JoinFIFO_Stat bit is set to 1.

Bit1 FIFO_Full

This bit indicates the cause of interrupt directly.

This bit is set to 1 when while EPx{x=0,a-c}Join.JoinFIFO_Stat bit = 1, the FIFO area for the corresponding endpoint is filled.

Bit0 FIFO_Empty

This bit indicates the cause of interrupt directly.

This bit is set to 1 when while EPx{x=0,a-c}Join.JoinFIFO_Stat bit = 1, the FIFO area for the corresponding endpoint is emptied.

7.5.4 E3h D_BulkIntStat (Device Bulk Interrupt Status)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	E3h	D_BulkIntStat	R (W)	7: CBW_Cmp	0: None	1: CBW Complete	00h
			R (W)	6: CBW_LengthErr	0: None	1: CBW Length Error	
			R (W)	5: CBW_Err	0: None	1: CBW Transaction Error	
			R (W)	4:	0:	1:	
			R (W)	3: CSW_Cmp	0: None	1: CSW Complete	
			R (W)	2: CSW_Err	0: None	1: CSW Error	
				1:	0:	1:	
				0:	0:	1:	

This register shows the interrupts associated with the Bulk transfer function. The cause of an interrupt indicated by any bit in this register can be cleared by writing 1 to that bit.

Bit7 CBW_Comp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when 31 bytes of CBW is received without errors.

Bit6 CBW_LengthErr

This bit indicates the cause of interrupt directly.

This bit is set to 1 when the received CBW packet is not 31 bytes in length.

Bit5 CBW_Err

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a CRC error or other transaction error is detected in the received CBW.

Bit4 Reserved

Bit3 CSW_Cmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when 31 bytes of CSW is received without errors.

Bit2 CSW_Err

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a CSW transmit error (not responded by ACK) occurrs.

Bits1-0 Reserved

7.5.5 E4h D_EPrIntStat (Device EPr Interrupt Status)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	E4h	D_EPrIntStat		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R	2: D_EPcIntStat	0: None	1: EPc Interrupt	
			R	1: D_EPbIntStat	0: None	1: EPb Interrupt	
			R	0: D_EPaIntStat	0: None	1: EPa Interrupt	

This register shows the interrupts of the endpoint EPr.

Bits7-3 Reserved

Bit2 D_EPcIntStat

This bit indicates the cause of interrupt indirectly.

If the D_EPcIntStat register has the cause of interrupt and the bit corresponding to that interrupt cause in the D_EPcIntEnb register is enabled, this bit is set to 1.

Bit1 D_EPbIntStat

This bit indicates the cause of interrupt indirectly.

If the D_EPbIntStat register has the cause of interrupt and the bit corresponding to that interrupt cause in the D_EPbIntEnb register is enabled, this bit is set to 1.

Bit0 D_EPaIntStat

This bit indicates the cause of interrupt indirectly.

If the D_EPaIntStat register has the cause of interrupt and the bit corresponding to that interrupt cause in the D_EPaIntEnb register is enabled, this bit is set to 1.

7.5.6 E5h D_EP0IntStat (Device EP0 Interrupt Status)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	E5h	D_EP0IntStat		7:	0:	1:	00h
			R (W)	6: OUT_ShortACK	0: None	1: OUT Short-Packet ACK	
			R (W)	5: IN_TranACK	0: None	1: IN Transaction ACK	
			R (W)	4: OUT_TranACK	0: None	1: OUT Transaction ACK	
			R (W)	3: IN_TranNAK	0: None	1: IN Transaction NAK	
			R (W)	2: OUT_TranNAK	0: None	1: OUT Transaction NAK	
			R (W)	1: IN_TranErr	0: None	1: IN Transaction Error	
			R (W)	0: OUT_TranErr	0: None	1: OUT Transaction Error	

This register shows the interrupt statuses of the endpoint EP0. The cause of an interrupt indicated by any bit in this register can be cleared by writing 1 to that bit.

Bit7 Reserved

Bit6 OUT_ShortACK

This bit indicates the cause of interrupt directly.

This bit is set to 1 at the same time OUT_TranACK is set when ACK is returned for the short packet received in an OUT transaction.

Bit5 IN_TranACK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when ACK is received in an IN transaction.

Bit4 OUT_TranACK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when ACK is returned to the host in an OUT transaction.

Bit3 IN_TranNAK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when NAK is returned to the host in an IN transaction.

Bit2 OUT_TranNAK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when NAK is returned to the host for the OUT or PING transaction attempted.

Bit1 IN_TranErr

This bit indicates the cause of interrupt directly.

This bit is set to 1 when in an IN transaction, STALL is returned, an error in the packet is found, or handshaking has timed out.

Bit0 OUT_TranErr

This bit indicates the cause of interrupt directly.

This bit is set to 1 when in an OUT transaction, STALL is returned or an error in the packet is found.

7.5.7 E6h D_EPaIntStat (Device EPa Interrupt Status)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	E6h	D_EPaIntStat		7:	0:	1:	00h
			R (W)	6: OUT_ShortACK	0: None	1: OUT Short Packet ACK	
			R (W)	5: IN_TranACK	0: None	1: IN Transaction ACK	
			R (W)	4: OUT_TranACK	0: None	1: OUT Transaction ACK	
			R (W)	3: IN_TranNAK	0: None	1: IN Transaction NAK	
			R (W)	2: OUT_TranNAK	0: None	1: OUT Transaction NAK	
			R (W)	1: IN_TranErr	0: None	1: IN Transaction Error	
			R (W)	0: OUT_TranErr	0: None	1: OUT Transaction Error	

This register shows the interrupt statuses of the endpoint EPa. The cause of an interrupt indicated by any bit in this register can be cleared by writing 1 to that bit.

Bit7 Reserved

Bit6 OUT_ShortACK

This bit indicates the cause of interrupt directly.

This bit is set to 1 at the same time OUT_TranACK is set when ACK is returned for the short packet received in an OUT transaction.

Bit5 IN_TranACK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when ACK is received in an IN transaction.

Bit4 OUT_TranACK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when ACK is returned to the host in an OUT transaction.

Bit3 IN_TranNAK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when NAK is returned to the host in an IN transaction.

Bit2 OUT_TranNAK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when NAK is returned to the host for the OUT or PING transaction attempted.

Bit1 IN_TranErr

This bit indicates the cause of interrupt directly.

This bit is set to 1 when in an IN transaction, STALL is returned, an error in the packet is found, or handshaking has timed out.

Bit0 **OUT_TranErr**

This bit indicates the cause of interrupt directly.

This bit is set to 1 when in an OUT transaction, STALL is returned or an error in the packet is found.

7.5.8 E7h D_EPbIntStat (Device EPb Interrupt Status)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	E7h	D_EPbIntStat		7:	0:	1:	00h
			R (W)	6: OUT_ShortACK	0: None	1: OUT Short Packet ACK	
			R (W)	5: IN_TranACK	0: None	1: IN Transaction ACK	
			R (W)	4: OUT_TranACK	0: None	1: OUT Transaction ACK	
			R (W)	3: IN_TranNAK	0: None	1: IN Transaction NAK	
			R (W)	2: OUT_TranNAK	0: None	1: OUT Transaction NAK	
			R (W)	1: IN_TranErr	0: None	1: IN Transaction Error	
			R (W)	0: OUT_TranErr	0: None	1: OUT Transaction Error	

This register shows the interrupt status of the endpoint EPb. The cause of an interrupt indicated by any bit in this register can be cleared by writing 1 to that bit.

Bit7 Reserved

Bit6 OUT_ShortACK

This bit indicates the cause of interrupt directly.

This bit is set to 1 at the same time OUT_TranACK is set when ACK is returned for the short packet received in an OUT transaction.

Bit5 IN_TranACK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when ACK is received in an IN transaction.

Bit4 OUT_TranACK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when ACK is returned to the host in an OUT transaction.

Bit3 IN_TranNAK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when NAK is returned to the host in an IN transaction.

Bit2 OUT_TranNAK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when NAK is returned to the host for the OUT or PING transaction attempted.

Bit1 IN_TranErr

This bit indicates the cause of interrupt directly.

This bit is set to 1 when in an IN transaction, STALL is returned, an error in the packet is found, or handshaking has timed out.

Bit0 OUT_TranErr

This bit indicates the cause of interrupt directly.

This bit is set to 1 when in an OUT transaction, STALL is returned or an error in the packet is found.

7.5.9 E8h D_EPcIntStat (D_EPc Interrupt Status)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	E8h	D_EPcIntStat		7:	0:	1:	00h
			R (W)	6: OUT_ShortACK	0: None	1: OUT Short Packet ACK	
			R (W)	5: IN_TranACK	0: None	1: IN Transaction ACK	
			R (W)	4: OUT_TranACK	0: None	1: OUT Transaction ACK	
			R (W)	3: IN_TranNAK	0: None	1: IN Transaction NAK	
			R (W)	2: OUT_TranNAK	0: None	1: OUT Transaction NAK	
			R (W)	1: IN_TranErr	0: None	1: IN Transaction Error	
			R (W)	0: OUT_TranErr	0: None	1: OUT Transaction Error	

This register shows the interrupt status of the endpoint EPc. The cause of an interrupt indicated by any bit in this register can be cleared by writing 1 to that bit.

Bit7 Reserved

Bit6 OUT_ShortACK

This bit indicates the cause of interrupt directly.

This bit is set to 1 at the same time OUT_TranACK is set when ACK is returned for the short packet received in an OUT transaction.

Bit5 IN_TranACK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when ACK is received in an IN transaction.

Bit4 OUT_TranACK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when ACK is returned to the host in an OUT transaction.

Bit3 IN_TranNAK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when NAK is returned to the host in an IN transaction.

Bit2 OUT_TranNAK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when NAK is returned to the host for the OUT or PING transaction attempted.

Bit1 IN_TranErr

This bit indicates the cause of interrupt directly.

This bit is set to 1 when in an IN transaction, STALL is returned, an error in the packet is found, or handshaking has timed out.

Bit0 OUT_TranErr

This bit indicates the cause of interrupt directly.

This bit is set to 1 when in an OUT transaction, STALL is returned or an error in the packet is found.

7.5.10 E9h~EFh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	E9h -EFh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.11 F0h *D_SIE_IntEnb (Device SIE Interrupt Enable)*

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	F0h	***D_SIE_DevIntEnb***		7:	0:	1:	00h
			R / W	6: ***EnNonJ***	0: Disable	1: Enable	
			R / W	5: EnRcvSOF	0: Disable	1: Enable	
			R / W	4: EnDetectRESET	0: Disable	1: Enable	
			R / W	3: EnDetectSUSPEND	0: Disable	1: Enable	
			R / W	2: EnChirpCmp	0: Disable	1: Enable	
			R / W	1: EnRestoreCmp	0: Disable	1: Enable	
			R / W	0: EnSetAddressCmp	0: Disable	1: Enable	

This register is used to enable or disable the assertion of the D_SIE_IntStat bit in the DeviceIntStat register for the interrupt causes accommodated in the D_SIE_IntStat register.

The EnNonJ bit in this register is effective even during Sleep/Snooze.

The synchronous bits (bits 5–0) can be read out, but cannot be written to during the Active60 and Act_Host state. For the processing to be executed on these synchronous bits when exiting theshifting out from the Act_Device state, refer to the description of the D_SIE_IntStat register.

7.5.12 F1h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	F1h	Reserved		7:	0:	1:	00h
				6:	0:	1	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.13 F2h D_FIFO_IntEnb (Device FIFO Interrupt Enable)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	F2h	D_FIFO_IntEnb	R / W	7: EnDescriptorCmp	0: Disable	1: Enable	00h
			R / W	6: EnFIFO_IDE_Cmp	0: Disable	1: Enable	
			R / W	5: EnFIFO1_Cmp	0: Disable	1: Enable	
			R / W	4: EnFIFO0_Cmp	0: Disable	1: Enable	
				3:	0:	1:	
			R / W	2: EnFIFO_NotEmpty	0: Disable	1: Enable	
			R / W	1: EnFIFO_Full	0: Disable	1: Enable	
			R / W	0: EnFIFO_Empty	0: Disable	1: Enable	

This register is used to enable or disable the assertion of the D_FIFO_IntStat bit in the DeviceIntStat register for the interrupt causes accommodated in the D_FIFO_IntStat register.

7.5.14 F3h D_BulkIntEnb (Device Bulk Interrupt Enable)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	F3h	D_BulkIntEnb	R / W	7: EnCBW_Cmp	0: Disable	1: Enable	00h
			R / W	6: EnCBW_LengthErr	0: Disable	1: Enable	
			R / W	5: EnCBW_Err	0: Disable	1: Enable	
				4:	0:	1:	
			R / W	3: EnCSW_Cmp	0: Disable	1: Enable	
			R / W	2: EnCSW_Err	0: Disable	1: Enable	
				1:	0:	1:	
				0:	0:	1:	

This register is used to enable or disable the assertion of the D_BulkIntStat bit in the DeviceIntStat register for the interrupt causes accommodated in the D_BulkIntStat register.

7.5.15 F4h D_EPrIntEnb (Device EPr Interrupt Enable)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	F4h	D_EPrIntEnb		7: (Reserved)	0:	1:	00h
				6: (Reserved)	0:	1:	
				5: (Reserved)	0:	1:	
				4: (Reserved)	0:	1:	
				3: (Reserved)	0:	1:	
			R / W	2: EnD_EnEPcIntStat	0: Disable	1: Enable	
			R / W	1: EnD_EnEPbIntStat	0: Disable	1: Enable	
			R / W	0: EnD_EnEPaIntStat	0: Disable	1: Enable	

This register is used to enable or disable the assertion of the D_EPrIntStat bit in the DeviceIntStat register for the interrupt causes accommodated in the D_EPrIntStat register.

7.5.16 F5h D_EP0IntEnb (Device EP0 Interrupt Enable)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	F5h	D_EP0IntEnb		7:	0:	1:	00h
			R / W	6: EnOUT_ShortACK	0: Disable	1: Enable	
			R / W	5: EnIN_TranACK	0: Disable	1: Enable	
			R / W	4: EnOUT_TranACK	0: Disable	1: Enable	
			R / W	3: EnIN_TranNAK	0: Disable	1: Enable	
			R / W	2: EnOUT_TranNAK	0: Disable	1: Enable	
			R / W	1: EnIN_TranErr	0: Disable	1: Enable	
			R / W	0: EnOUT_TranErr	0: Disable	1: Enable	

This register is used to enable or disable the assertion of the D_EP0IntStat bit in the DeviceIntStat register for the interrupt causes accommodated in the D_EP0IntStat register.

7.5.17 F6h D_EPaIntEnb (DeviceEPa Interrupt Enable)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	F6h	D_EPaIntEnb		7:	0:	1:	00h
			R / W	6: EnOUT_ShortACK	0: Disable	1: Enable	
			R / W	5: EnIN_TranACK	0: Disable	1: Enable	
			R / W	4: EnOUT_TranACK	0: Disable	1: Enable	
			R / W	3: EnIN_TranNAK	0: Disable	1: Enable	
			R / W	2: EnOUT_TranNAK	0: Disable	1: Enable	
			R / W	1: EnIN_TranErr	0: Disable	1: Enable	
			R / W	0: EnOUT_TranErr	0: Disable	1: Enable	

This register is used to enable or disable the assertion of the EPaIntStat bit in the D_EPrIntStat register for the interrupt causes accommodated in the D_EPaIntStat register.

7.5.18 F7h D_EPbIntEnb (Device EPb Interrupt Enable)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	F7h	D_EPbIntEnb		7:	0:	1:	00h
			R / W	6: EnOUT_ShortACK	0: Disable	1: Enable	
			R / W	5: EnIN_TranACK	0: Disable	1: Enable	
			R / W	4: EnOUT_TranACK	0: Disable	1: Enable	
			R / W	3: EnIN_TranNAK	0: Disable	1: Enable	
			R / W	2: EnOUT_TranNAK	0: Disable	1: Enable	
			R / W	1: EnIN_TranErr	0: Disable	1: Enable	
			R / W	0: EnOUT_TranErr	0: Disable	1: Enable	

This register is used to enable or disable the assertion of the EPbIntStat bit in the D_EPrIntStat register for the interrupt causes accommodated in the D_EPbIntStat register.

7.5.19 F8h D_EPcIntEnb (Device EPc Interrupt Enable)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	F8h	D_EPcIntEnb		7:	0:	1:	00h
			R / W	6: EnOUT_ShortACK	0: Disable	1: Enable	
			R / W	5: EnIN_TranACK	0: Disable	1: Enable	
			R / W	4: EnOUT_TranACK	0: Disable	1: Enable	
			R / W	3: EnIN_TranNAK	0: Disable	1: Enable	
			R / W	2: EnOUT_TranNAK	0: Disable	1: Enable	
			R / W	1: EnIN_TranErr	0: Disable	1: Enable	
			R / W	0: EnOUT_TranErr	0: Disable	1: Enable	

This register is used to enable or disable the assertion of the EPcIntStat bit in the D_EPrIntStat register for the interrupt causes accommodated in the D_EPcIntStat register.

7.5.20 F9h~FFh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	F9h -FFh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.21 100h *D_Reset (Device Reset)*

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	100h	***D_Reset***		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
			R / W	0: ***ResetDTM***	0: Reset DTM	1: Do nothing	

This register is used to reset the device transceiver macro.

This register can be accessed even during Sleep/Snooze.

Bits7-1 Reserved

Bit0 ResetDTM

Writing 0 to this bit initializes the device transceiver macro of the LSI.

To negate the reset, write 1 to this bit.

7.5.22 101h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	101h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.23 102h D_NegoControl (Device Nego Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	102h	D_NegoControl	R / W	7: DisBusDetect	0: Enable BusDetect	1: Disable BusDetect	00h
			R / W	6: EnAutoNego	0: Disable AutoNegotiation	1: Enable AutoNegotiation	
			R / W	5: InSUSPEND	0: Do nothing	1: Monitor NonJ	
			R / W	4: DisableHS	0: HS mode	1: Disable HS mode	
			R / W	3: SendWakeup	0: Do nothing	1: Send Remotewakeup Signal	
			R / W	2: RestoreUSB	0: Do nothing	1: Restore operation mode	
			R / W	1: GoChirp	0: Do nothing	1: Do Chirp sequence	
			R / W	0: ActiveUSB	0: Disactivate USB	1: Activate USB	

This register is used to set the operations associated with device negotiation.

Bit7 DisBusDetect

Setting this bit to 1 disables the automatic detection of the reset/suspend states of the USB. If this bit remains cleared to 0, bus activities on the USB bus are monitored in order to detect the reset/suspend states of the USB.

During HS mode, if no bus activities are detected for a 3 ms period, operation mode is automatically switched to FS mode and then determination is made to see if the USB is in a reset or suspend state, according to which the relevant cause of interrupt (DetectReset or DetectSuspend) is set. During FS mode, if no bus activities are detected for a 3 ms period, a suspend state of the USB is assumed. Furthermore, if SE0 in duration of 2.5 μs or more is detected, a reset is assumed, and the relevant cause of interrupt is set.

When the DetectReset or DetectSuspend bit is set, set the DisBusDetect bit to 1 to disable the detection of states while the reset or suspend state of the USB continues. When using the AutoNegotiation function, do not set this bit to 1.

Bit6 EnAutoNego

This bit enables the AutoNegotiation function. This function automates a series of sequences during reset detection until speed mode is determined after a speed negotiation is complete. For details about the AutoNegotiation function, refer to the relevant section in Chapter 6, "Functional Description."

Bit5 InSUSPEND

When while using the AutoNegotiation function a suspend state of the USB is detected, this bit is automatically set to 1 to enable a function to detect the NonJ state. Clear this bit to 0 when returning from a suspend state of the USB.

For a detailed explanation about the AutoNegotiation function, refer to "Auto Negotiation Function" in Chapter 6, "Functional Description."

Bit4 DisableHS

If this bit is set to 1 when GoChirp is set to 1, the LSI is forcibly placed in FS mode without sending out DeviceChirp and generates a ChirpCmp interrupt.

Bit3 SendWakeup

When this bit is set to 1, a RemoteWakeup signal (K) is output to the USB port.

After 1 ms or more but within 15 ms have elapsed after the LSI started sending out the RemoteWakeup signal, clear this bit to 0 to stop the transmission.

Bit2 RestoreUSB

If this bit is set to 1 when resuming the USB from a suspend state, operation mode (FS or HS) is automatically switched to the one in which the USB was in prior to Suspend and the relevant cause of interrupt (RestoreCmp) is set.

This bit is automatically cleared to 0 after the operation is complete.

When using the AutoNegotiation function, do not set or clear this bit because the function of this bit is automatically controlled during that time.

Bit1 GoChirp

If this bit is set to 1 while the USB bus is in a reset status, HS Detection Handshaking is executed between the host and hub, and the TermSelect and XcvrSelect bits in the XcvrControl register and the FSxHS bit in the USB_Status register are automatically set. The cause of interrupt (ChirpCmp) is set at the same time the operation is complete.

This bit is automatically cleared to 0 after the operation is complete. The result of HS Detection Handshaking can be confirmed by inspecting the FSxHS bit in the USB_Status register.

When using the AutoNegotiation function, do not set or clear this bit because the function of this bit is automatically controlled during that time.

Bit0 ActiveUSB

Since this bit remains cleared to 0 after the LSI is reset in hardware, the entire function of the USB is disabled. Therefore, after setting up the LSI, set this bit to 1 to enable the USB.

7.5.24 103h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	103h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.25 104h D_ClrAllEPnJoin (Device Clear All EPn Join)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	104h	D_ClrAllEPnJoin	W	7:ClrJoinIDE	0: Do nothing	1: Clear Join IDE	XXh
			W	6:ClrJoinFIFO_Stat	0: Do nothing	1: Clear Join FIFO_Stat	
				5:	0:	1:	
				4:	0:	1:	
			W	3:ClrJoinDMA1	0: Do nothing	1: Clear Join DMA1	
			W	2:ClrJoinDMA0	0: Do nothing	1: Clear Join DMA0	
			W	1:ClrJoinCPU_Rd	0: Do nothing	1: Clear Join CPU_Rd	
			W	0:ClrJoinCPU_Wr	0: Do nothing	1: Clear Join CPU_Wr	

This register is used to clear a connection between the port selected by a bit in it and an endpoint.

The bits in this register are automatically cleared to 0 after a connection is cleared.

Do not set any bit in this register to 1 while an endpoint is connected to a port (EPx[x=0,a-c]Join register's relevant bit = 1) and the port is active.

7.5.26 105h D_XcvrControl (Device Xcvr Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	105h	D_XcvrControl	R / W	7: TermSelect	0: HS Termination	1: FS Termination	41h
			R / W	6: XcvrSelect	0: HS Transceiver	1: FS Transceiver	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R / W	1: OpMode [1]	OpMode [1:0]		
				0: OpMode [0]			

This register is used to make settings relating to the device transceiver macro.

Bit7 TermSelect

This bit selects FS or HS termination to enable it. This bit is automatically set when HS Detection Handshaking is executed via the GoChirp bit in the USB_Control register or when the AutoNegotiation function is executed after the EnAutoNego bit in the D_NegoControl register is set.

Bit6 XcvrSelect

This bit selects the FS or HS transceiver to enable it. This bit is automatically set when HS Detection Handshaking is executed via the GoChirp bit in the USB_Control register or when the AutoNegotiation function is executed after the EnAutoNego bit in the D_NegoControl register is set.

Bits5-2 Reserved

Bits1-0 OpMode

These bits set the operation mode of the UTM.

These bits are normally not required to be set unless the USB cable is removed (*), or unless the USB is in a suspend state or during test mode.

OpMode		
00	"Normal Operation"	Normal operating state
01	"Non-Driving"	Set the bits to this state when the USB cable is removed.
10	"Disable Bitstuffing and NRZI encoding"	Set the bits to this state when the USB is in test mode.
11	"Power-Down"	Set the bits to this state when the USB is suspended.

* It is recommended to set this register to “41h” if the USB cable is not connected.

7.5.27 106h D_USB_Test (Device USB_Test)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	106h	D_USB_Test	R / W	7: EnHS_Test	0: Do nothing	1: EnHS_Test	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
			R / W	3: Test_SE0_NAK	0: Do nothing	1: Test_SE0_NAK	
			R / W	2: Test_J	0: Do nothing	1: Test_J	
			R / W	1: Test_K	0: Do nothing	1: Test_K	
			R / W	0: Test_Packet	0: Do nothing	1: Test_Packett	

This register is used to set the operations relating to USB2.0 test mode during USB device operation. To perform any test mode defined in the USB2.0 standard, set the bit corresponding to the test mode specified in a SetFeature request and after the status stage is complete, set the EnHS_Test bit to 1.

Bit7 EnHS_Test

If when this bit is set to 1, any of the 4 low-order bits in the USB_Test register is set to 1, the test mode corresponding to that bit is entered. Before test mode can be performed, the D_NegoControl register's DisBusDetect bit must be set to 1 to disable the detection of suspend/reset states of the USB. In addition, disable the AutoNegotiation function by clearing the D_NegoControl register's EnAutoNego bit to 0.

Also make sure that shift to the test mode is made after the status stage in a SetFeature request is complete.

Bits6-4 Reserved

Bit3 Test_SE0_NAK

The LSI can be shifted to the Test_SE0_NAK test mode by setting this bit to 1 and then the EnHS_Test bit to 1.

Bit2 TEST_J

The LSI can be shifted to the Test_J test mode by setting this bit to 1 and then the EnHS_Test bit to 1. This test mode requires that TermSelect and XcvrSelect in the XcvrControl register be set according to the speed mode before the EnHS_Test bit is set to 1, and that OpMode be set to "10" (= Disable Bitstuffing and NRZI encoding).

Bit1 TEST_K

The LSI can be shifted to the Test_K test mode by setting this bit to 1 and then the EnHS_Test bit to 1. This test mode requires that TermSelect and XcvrSelect in the XcvrControl register be set according to the speed mode before the EnHS_Test bit is set to 1, and that OpMode be set to "10" (= Disable Bitstuffing and NRZI encoding).

Bit0 Test_Packet

The LSI can be shifted to the Test_Packet test mode by setting this bit to 1 and then the EnHS_Test bit to 1.

Since this test mode can be used at any endpoint other than EP0, following settings need to be made:

1) Set MaxPacketSize and the direction of transfer for the endpoint EPx{x=a-c} to 64 or more and IN, respectively, and then set EndpointNumber to "0xF" to make the endpoint usable. Remember to allocate 64 bytes or more of storage to the FIFO of the endpoint EPx{x=a-c}.
2) Make sure that other endpoint settings do not overlap the above EPx{x=a-c} setting. Or clear the EPx{x=a-c}Config.EnEndpoint bit.
3) Clear the FIFO for EPx{x=a-c} and write the data for test packet shown below into the FIFO. Clear the IN_TranErr bit in the EPx{x=a-c}IntStat register to 0.
4) Each time test packet transmission is complete, the IN_TranErr status is set to 1.

The following 53 bytes are the data to be written to the FIFO during packet transmission test mode:

00h, 00h, 00h, 00h, 00h, 00h, 00h, 00h,

00h, Aah, Aah, Aah, Aah, Aah, Aah, Aah,

Aah, Eeh, Eeh, Eeh, Eeh, Eeh, Eeh, Eeh,

Eeh, Feh, FFh, FFh, FFh, FFh, FFh, FFh,

FFh, FFh, FFh, FFh, FFh, 7Fh, BFh, DFh,

EFh, F7h, FBh, FDh, FCh, 7Eh, BFh, DFh,

EFh, F7h, FBh, FDh, 7Eh

Since the SIE adds PID and CRC to a test packet when it is transmitted, the data to be written into the FIFO should consist of only a range of the test packet data stipulated in USB Standard Rev. 2.0 from the data next to DATA0 PID to those that follow but not including CRC16.

7.5.28 107h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	107h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.29 108h D_EPnControl (Device Endpoint Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	108h	D_EPnControl	W	7: AllForceNAK	0: Do nothing	1: Set All ForceNAK	XXh
			W	6: EPrForceSTALL	0: Do nothing	1: Set EP's ForceSTALL	
			W	5: AllFIFO_Clr	0: Do nothing	1: Clear All FIFO	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
			W	0: EP0FIFO_Clr	0: Do nothing	1: Clear EP0 FIFO	

This register is used to set the endpoint operations. This is a write-only register.

Bit7 AllForceNAK

This bit sets the ForceANK bits for all endpoints to 1.

Bit6 EPrForceSTALL

This bit sets the ForceSTALL bits for the endpoints EPa, EPb, and EPc to 1.

Bit5 AllFIFO_Clr

This bit clears the FIFOs for all endpoints. When areas for the respective endpoints are set, temporarily set this bit to 1 to clear the FIFOs for all endpoints after the area settings is complete. This bit is automatically cleared after the FIFOs have successfully been cleared.

When DMAx{x=0,1} is joined to any endpoint and the relevant DMA is active (when DMA_Running bit = 1), do not set the bit for that endpoint to 1.

Bits4-1 Reserved

Bit0 EP0FIFO_Clr

This bit clears the FIFO for the endpoint EP0.

When set to 1, this bit only clears the FIFO and does not hold the value set in it.

When DMAx{x=0,1} is joined to the endpoint EP0 and the relevant DMA is active (when DMA_Running bit = 1), do not set this bit to 1.

7.5.30 109h D_EPrFIFO_Clr (Device Endpoint FIFO Clear)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	109h	EPrFIFO_Clr		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			W	2: EPcFIFO_Clr	0: Do nothing	1: Clear EPc FIFO	
			W	1: EPbFIFO_Clr	0: Do nothing	1: Clear EPb FIFO	
			W	0: EPaFIFO_Clr	0: Do nothing	1: Clear EPa FIFO	

This register is used to clear the FIFO for the relevant endpoint. This is a write-only register.

When set to 1, all bits in this register only clear the FIFO and do not retain the value set in it.

When DMAx{x=0,1} is joined to any endpoint and the relevant DMA is active (when DMA_Running bit = 1), do not set the bit for that endpoint to 1.

7.5.31 10Ah D_BulkOnlyControl (Device BulkOnly Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	10Ah	D_BulkOnlyControl	R / W	7:AutoForceNAK_CBW	0: None	1: AutoForceNAK after CBW	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R / W	2: GoCBW_Mode	0: None	1: Begin CBW Mode	
			R / W	1: GoCSW_Mode	0: None	1: Begin CSW Mode	
				0:	0:	1:	

This register controls the Bulk Only Support function.

Bit7 AutoForceNAK_CBW

If this bit is set to 1, the ForceNAK bit for the relevant endpoint is set to 1 when the OUT transaction in which a CBW is to be received for CBW support is complete.

Bits6-3 Reserved

Bit2 GoCBW_Mode

When this bit is set to 1, CBW support is executed at the relevant endpoint. For the endpoints at which CBW support will be executed, refer to the section on BulkOnlyConfig register.

Bit1 GoCSW_Mode

When this bit is set to 1, CSW support is executed at the relevant endpoint. For the endpoints at which CSW support will be executed, refer to the section on BulkOnlyConfig register.

Bit0 Reserved

7.5.32 10Bh D_BulkOnlyConfig (Device BulkOnly Configuration)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	10Bh	D_BulkOnlyConfig		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R / W	2: EPcBulkOnly	0: None	1: Enable BulkOnly on EPc	
			R / W	1: EPbBulkOnly	0: None	1: Enable BulkOnly on EPb	
				0:	0:	1:	

This register enables the Bulk Only Support function.

Bits7-3 Reserved

Bit2 EPcBulkOnly

Setting this bit to 1 enables the Bulk Only Support function for the endpoint EPc. If the endpoint EPc is an OUT endpoint when the Bulk Only Support function is enabled this way, CBW support can be executed by setting the BulkOnlyControl.GoCBW_Mode bit. Or, if the endpoint EPc is an IN endpoint, CSW support can be executed by setting the BulkOnlyControl.GoCSW_Mode bit.

Do not enable the Bulk Only Support function for two or more OUT endpoints at the same time.

Bit1 EPbBulkOnly

Setting this bit to 1 enables the Bulk Only Support function for the endpoint EPb. If the endpoint EPb is an OUT endpoint when the Bulk Only Support function is enabled this way, CBW support can be executed by setting the BulkOnlyControl.GoCBW_Mode bit. Or, if the endpoint EPb is an IN endpoint, CSW support can be executed by setting the BulkOnlyControl.GoCSW_Mode bit.

Do not enable the Bulk Only Support function for two or more OUT endpoints at the same time.

Bit0 Reserved

7.5.33 10C~10Fh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	10Ch -10Fh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.34 110h D_EP0SETUP_0 (Device EP0 SETUP 0)
7.5.35 111h D_EP0SETUP_1 (Device EP0 SETUP 1)
7.5.36 112h D_EP0SETUP_2 (Device EP0 SETUP 2)
7.5.37 113h D_EP0SETUP_3 (Device EP0 SETUP 3)
7.5.38 114h D_EP0SETUP_4 (Device EP0 SETUP 4)
7.5.39 115h D_EP0SETUP_5 (Device EP0 SETUP 5)
7.5.40 116h D_EP0SETUP_6 (Device EP0 SETUP 6)
7.5.41 117h D_EP0SETUP_7 (Device EP0 SETUP 7)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device	110h -117h	D_EP0SETUP_0 -D_EP0SETUP_7	R	7: EP0SETUP_n [7]	Endpoint 0 SETUP Data 0 -Endpoint 0 SETUP Data 7	00h
				6: EP0SETUP_n [6]		
				5: EP0SETUP_n [5]		
				4: EP0SETUP_n [4]		
				3: EP0SETUP_n [3]		
				2: EP0SETUP_n [2]		
				1: EP0SETUP_n [1]		
				0: EP0SETUP_n [0]		

The 8 bytes of data received in the setup stage of the endpoint EP0 are stored in these registers sequentially beginning with EP0SETUP_0.

EP0SETUP_0

BmRequestType is set in this register.

EP0SETUP_1

BRequest is set in this register.

EP0SETUP_2

The 8 low-order bits of Wvalue are set in this register.

EP0SETUP_3

The 8 high-order bits of Wvalue are set in this register.

EP0SETUP_4

The 8 low-order bits of WIndex are set in this register.

EP0SETUP_5

The 8 high-order bits of WIndex are set in this register.

EP0SETUP_6

The 8 low-order bits of WLength are set in this register.

EP0SETUP_7

The 8 high-order bits of WLength are set in this register.

7.5.42 118h D_USB_Address (Device USB Address)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	118h	D_USB_Address	R / W	7: SetAddress	0: none	1: Set USB Address	00h
			R (W)	6: USB_Address [6]	USB Address		
				5: USB_Address [5]			
				4: USB_Address [4]			
				3: USB_Address [3]			
				2: USB_Address [2]			
				1: USB_Address [1]			
				0: USB_Address [0]			

This register is used by the AutoSetAddress function to set a USB address.

When a SetAddress() request is received, the AutoSetAddress function automatically performs a control transfer for it. When the status stage for the control transfer associated with the SetAddress() request is complete, the AutoSetAddress function sets USB_Address and then issues a SetAddressCmp status.

Bit7 SetAddress

If this bit is set when a SetAddress request is received, USB_Address is automatically set when a status stage of the request is completed. Automatic address setup mode must be disabled for this bit setting to have any effect.

Bits6-0 USB_Address

These bits are used to set a USB address.

The address is automatically written to by the AutoSetAddress function.

Any value can be written to these bits in software, but when a SetAddress() request is received, the value will be automatically rewritten.

7.5.43 119h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	119h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.44 11Ah D_SETUP_Control(Device SETUP Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	11Ah	D_SETUP_Control		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
			R / W	0: ProtectEP0	0: None	1: Protect EP0	

This register is used for settings associated with control transfers.

Bits7-1 Reserved

Bit0 ProtectEP0

This bit is set to 1 when after the setup stage of a control transfer is complete, the received data is stored in the registers EP0SETUP_0 through EP0SETUP_7.

At the same time, the ForceSTALL bits in the D_EP0ControlIN and D_EP0ControlOUT registers are cleared to 0 and the ForceNAK and ToggleStat bits in those registers both are set to 1, all automatically.

The ProtectEP0 bit is set when a SETUP transaction is performed. Therefore, it is set for the SetAddres() request received.

The ForceNAK and ForceSTALL bits for EP0 cannot have their settings altered while this bit remains set.

7.5.45 11Bh~11Dh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	11Bh -11Dh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.46 11Eh D_FrameNumber_H (Device FrameNumber High)

7.5.47 11Fh D_FrameNumber_L (Device FrameNumber Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	11Eh	D_FrameNumber _H	R	7: FnInvalid	0: Frame number is valid	1: Frame number is not valid	80h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R	2: FrameNumber [10]	Frame Number High		
				1: FrameNumber [9]			
				0: FrameNumber [8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device	11Fh	D_FrameNumber _L	R	7: FrameNumber [7]	Frame Number Low	00h
				6: FrameNumber [6]		
				5: FrameNumber [5]		
				4: FrameNumber [4]		
				3: FrameNumber [3]		
				2: FrameNumber [2]		
				1: FrameNumber [1]		
				0: FrameNumber [0]		

These registers show the frame number of the USB that is updated for each SOF token received. To get a frame number, the FrameNumber_H and FrameNumber_L registers must be accessed in pairs. In that case, be sure to access the FrameNumber_H register first.

11Eh.Bit7 FnInvalid

This bit is set to 1 when an error occurs in the received SOF packet.

11Eh.Bit6-3 Reserved

11Eh.Bit2-0, 11Fh.Bit7-0 FrameNumber [10:0]

These bits show the FrameNumber of the received SOF packet.

7.5.48 120h D_EP0MaxSize (Device EP0 Max Packet Size)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	120h	D_EP0MaxSize		7:	0:	1:	40h
			R / W	6: EP0MaxSize [6]	Endpoint [0] Max Packet Size		
				5: EP0MaxSize [5]			
				4: EP0MaxSize [4]			
				3: EP0MaxSize [3]			
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register is used to set the endpoint EP0.

Bit7 Reserved

Bits6-3 EP0MaxSize [6:3]

These bits set the MaxPacketSize of the endpoint EP0.

Any size can be specified from those listed below for use with this endpoint:

During FS	8, 16, 32, or 64 bytes
During HS	64 bytes

Bits2-0 Reserved

7.5.49 121h D_EP0Control (Device EP0 Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	121h	D_EP0Control	R / W	7: INxOUT	0: OUT	1: IN	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
			R / W	0: ReplyDescriptor	0: Do nothing	1: Reply Descriptor	

This register sets the endpoint EP0.

Bit7 INxOUT

This bit sets the direction of transfer for the endpoint EP0.

Consider the request received in the setup stage when setting a value in this bit.

If a data stage is involved, set the direction of transfer at the data stage in this bit. When a setup stage is complete, the ForceNAK bits in the D_EP0ControlIN and D_EP0ControlOUT registers are set, so be sure to clear these bits before executing the data and status stages.

When the data stage is complete, reset this bit as suited to the direction of the status stage. If the direction of the data stage is IN, the status stage is directed for OUT, so set this bit to 0. Conversely, if the direction of the data stage is OUT or a data stage is not involved, the status stage is directed for IN, so clear the FIFO for the endpoint EP0 and set this bit to 1.

An IN or OUT transaction in the direction opposite to the set value of this bit is attempted, a NAK response is returned. However, if the ForceSTALL bit in the D_EP0ControlIN or D_EP0ControlOUT register for that transaction is set, the transaction is responded with STALL.

Bits6-1 Reserved

Bit0 ReplyDescriptor

This bit executes the descriptor reply function.

When this bit is set to 1, bytes of descriptor data equal to MaxPacketSize are sent back to the host from the FIFO in response to an IN transaction at the endpoint EP0. The descriptor data here refers to the data that starts from the address set in the DescAdr_H,L registers and whose size is set in the DescSize_H,L registers. Since these set values are updated during execution of the descriptor reply function, they need to be set each time the ReplyDescriptor bit is set.

The DescAdr_H,L registers are incremented for each transaction performed by an amount equal to the transmitted data bytes, and the DescSize_H,L registers are decremented by an amount equal to the transmitted data bytes.

When the transmit operation is complete after the bytes of data set by DescSizeH,L have been transmitted, or when a transaction other than an IN transaction has been performed, the descriptor reply function is terminated and the ReplyDescriptor bit is cleared to 0. At the same time, the DescriptorCmp bit in the D_FIFO_IntStat register and the IN_TranACK bit in the D_EP0IntStat register both are set to 1.

For more detailed explanations, refer to Chapter 6, "Functional Description."

7.5.50 122h D_EP0ControlIN (Device EP0 Control IN)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	122h	D_EP0ControlIN		7:	0:	1:	00h
			R / W	6: EnShortPkt	0: Do nothing	1: Enable short Packet	
				5:	0:	1:	
			R	4: ToggleStat	Toggle sequence bit		
			W	3: ToggleSet	0: Do nothing	1: Set Toggle sequence bit	
			W	2: ToggleClr	0: Do nothing	1: Clear Toggle sequence bit	
			R / W	1: ForceNAK	0: Do nothing	1: Force NAK	
			R / W	0: ForceSTALL	0: Do nothing	1: Force STALL	

This register sets the operations relating to IN transactions at the endpoint EP0 and shows the status of those operations.

Bit7 Reserved

Bit6 EnShortPkt

Setting this bit to 1 allows the data in the FIFO less than MaxPacketSize to be transmitted as a short packet for an IN transaction at the endpoint EP0. When the IN transaction in which a short packet has been transmitted is complete, this bit is automatically cleared to 0. If a packet equal to MaxPacketSize has been transmitted, this bit is not cleared.

If this bit is set to 1 while no data exists in the FIFO, a packet of zero-length can be transmitted in response to an IN token from the host. If while a packet is being transmitted, data is written to the relevant FIFO after this bit was set, the written data may inadvertently be included in the transmitted packet depending on the write timing. Therefore, do not write data to the FIFO until after a packet transmission is complete and this bit is cleared.

Bit5 Reserved

Bit4 ToggleStat

This bit shows the status of the toggle sequence bit in an IN transaction at the endpoint EP0.

Bit3 ToggleSet

This bit sets the toggle sequence bit in an IN transaction at the endpoint EP0 to 1. If this bit is set at the same time the ToggleClr bit is set, the function of the ToggleClr bit has priority.

Bit2 ToggleClr

This bit clears the toggle sequence bit in an IN transaction at the endpoint EP0 to 0. If this bit is set at the same time the ToggleSet bit is set, the function of this bit has priority.

Bit1 ForceNAK

Setting this bit to 1 returns the NAK response for an IN transaction at the endpoint EP0 irrespective of the number of data bytes in the FIFO.

When the RcvEP0SETUP bit in the MainIntStat register is set to 1 upon completion of a setup stage, this bit is set to 1 and while the RcvEP0SETUP bit remains set, cannot be cleared to 0. Furthermore, this bit is set to 1 when an IN transaction in which a short packet was transmitted is complete.

If a transaction has already been underway when an attempt was made to set this bit to 1, the bit is not set until the transaction is complete, and is set to 1 upon completion. If no transactions are under execution, the bit is set to 1 immediately.

Bit0 ForceSTALL

Setting this bit to 1 returns STALL as response for an IN transaction at the endpoint EP0. This bit has priority over the ForceNAK bit.

When the RcvEP0SETUP bit in the DeviceIntStat register is set to 1 upon completion of a setup stage, this bit is cleared to 0 and while the RcvEP0SETUP bit remains set, cannot be set to 1.

If there is any transaction currently executed and this bit is set a certain time after the transaction has started, setting of this bit takes effect beginning with the next transaction.

7.5.51 123h D_EP0ControlOUT (Device EP0 Control OUT)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	123h	D_EP0ControlOUT	R / W	7: AutoForceNAK	0: Do nothing	1: Auto Force NAK	00h
				6:	0:	1:	
				5:	0:	1	
			R	4: ToggleStat	Toggle sequence bit		
			W	3: ToggleSet	0: Do nothing	1: Set Toggle sequence bit	
			W	2: ToggleClr	0: Do nothing	1: Clear Toggle sequence bit	
			R / W	1: ForceNAK	0: Do nothing	1: Force NAK	
			R / W	0: ForceSTALL	0: Do nothing	1: Force STALL	

This register sets the operations relating to OUT transactions at the endpoint EP0 and shows the status of those operations.

Bit7 AutoForceNAK

When while this bit is set an OUT transaction at the endpoint EP0 is completed without errors, the ForceNAK bit in this register is set to 1.

Bits6-5 Reserved

Bit4 ToggleStat

This bit shows the status of the toggle sequence bit in an OUT transaction at the endpoint EP0.

Bit3 ToggleSet

This bit sets the toggle sequence bit in an OUT transaction at the endpoint EP0 to 1. If this bit is set at the same time the ToggleClr bit is set, the function of the ToggleClr bit has priority.

Bit2 ToggleClr

This bit clears the toggle sequence bit in an OUT transaction at the endpoint EP0 to 0. If this bit is set at the same time the ToggleSet bit is set, the function of this bit has priority.

Bit1 ForceNAK

Setting this bit to 1 returns NAK as response for an OUT transaction at the endpoint EP0 irrespective of the amount of free space in the FIFO.

When the RcvEP0SETUP bit in the MainIntStat register is set to 1 upon completion of a setup stage, this bit is set to 1 and while the RcvEP0SETUP bit remains set, cannot be cleared to 0.

If a transaction has already been underway when an attempt was made to set this bit to 1, the bit is not set until the transaction is complete, and is set to 1 upon completion of it. If no transactions are under execution, the bit is set to 1 immediately.

Bit0 ForceSTALL

Setting this bit to 1 returns STALL as response for an OUT transaction at the endpoint EP0. This bit has priority over the ForceNAK bit.

When the RcvEP0SETUP bit in the DeviceIntStat register is set to 1 upon completion of a setup stage, this bit is cleared to 0 and while the RcvEP0SETUP bit remains set, cannot be set to 1.

If there is any transaction currently executed and this bit is set a certain time after the transaction has started, setting of this bit takes effect beginning with the next transaction.

7.5.52 124h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	124h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.53 125h D_EP0Join (Device EndPoint0 Join)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	125h	D_EP0Join		7:	0:	1:	00h
			R / W	6: JoinFIFO_Stat	0: Do nothing	1 Join EP0 to show Status	
				5:	0:	1:	
				4:	0:	1	
			R / W	3: JoinDMA1	0: Do nothing	1: Join EP0 to DMA1	
			R / W	2: JoinDMA0	0: Do nothing	1: Join EP0 to DMA0	
			R / W	1: JoinCPU_Rd	0: Do nothing	1: Join EP0 to CPU_Rd	
			R / W	0: JoinCPU_Wr	0: Do nothing	1: Join EP0 to CPU_Wr	

This register specifies the port with which a data transfer to or from the endpoint 0 is to be performed.

Bit7 Reserved

Bit6 JoinFIFO_Stat

Allows the Full/Empty/NotEmpty status of the FIFO of endpoint EP0 to be monitored via D_FIFO_IntStat.FIFO_full, D_FIFO_IntStat.FIFO_Empty, and D_FIFO_IntStat.FIFO_NotEmpty.

Bits5-4 Reserved

Bit3 JoinDMA1

Performs a data transfer for the DMA1 with the FIFO of endpoint EP0. The transfer direction is determined by the DMA1_Control.Dir bit.

Bit2 JoinDMA0

Performs a data transfer for the DMA0 with the FIFO of endpoint EP0. The transfer direction is determined by the DMA0_Control.Dir bit.

Bit1 JoinCPU_Rd

Performs a read transfer of CPU register access with the FIFO of endpoint EP0. Namely, when a read of the FIFO_Rd_0,1 registers or FIFO_ByteRd register is performed, data is read from the FIFO of this endpoint.

Bit0 JoinCPU_Wr

Performs a write transfer of CPU register access with the FIFO of endpoint EP0. Namely, when a write to the FIFO_Wr_0,1 registers is performed, data is written to the FIFO of this endpoint

If the JoinDMAx{x=0,1} bit is set, it is possible to refer to the remaining bytes of data in the FIFO when the DMA0_Control.Dir bit = 1 or the amount of free space in the FIFO when the DMA0_Control.Dir bit = 0 by inspecting the DMAx{x=0,1}_Remain_H,L registers.

If the JoinCPU_Rd or JoinCPU_Wr bit is set, it is possible to write or read data to or from the FIFO_Rd_0,1, FIFO_ByteRd, or FIFO_Wr_0,1 registers after inspecting the FIFO_RdRemain_H,L or FIFO_WrRemain_H,L registers.

The JoinDMA1, JoinDMA0, JoinCPU_Rd, and JoinCPU_Wr bits can be set to 1 only one bit at a time. If an attempt is made to set two or more of these bits by writing 1 at the same time, only the higher bit is set.

7.5.54 126h~12Fh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	126h -12Fh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.55 130h D_EPaMaxSize_H (Device EPa Max Packet Size High)

7.5.56 131h D_EPaMaxSize_L (Device EPa Max Packet Size Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	130h	D_EPaMaxSize_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R / W	1: MaxSize [9]			
				0: MaxSize [8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	131h	D_EPaMaxSize_L	R / W	7: MaxSize [7]	Endpoint [a] Max Packet Size		00h
				6: MaxSize [6]			
				5: MaxSize [5]			
				4: MaxSize [4]			
				3: MaxSize [3]			
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

These registers set MaxPacketSize.

130h.Bit7-2 Reserved

131h.Bit1-0, 131h.Bit7-3 EPaMaxSize [9:3]

These bits set MaxPacketSize for the endpoint EPa.

To use this endpoint for bulk transfers, set MaxSize to any of the following:

During FS 8, 16, 32, or 64 bytes

During HS 512 bytes

To use this endpoint for interrupt transfers, set MaxSize to any desired transfer bytes within the limits given below:

During FS Up to 64 bytes

During HS Up to 512 bytes

131h.Bit2-0 Reserved

7.5.57 132h D_EPaConfig_0 (Device EPa Configuration 0)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	132h	D_EPaConfig_0	R / W	7: INxOUT	0: OUT	1: IN	00h
			R / W	6: IntEP_Mode	0: Normal Toggle (IN) 0: Bulk OUT (OUT)	1: Always Toggle (IN) 1: Interrupt OUT (OUT)	
			R / W	5: EnEndpoint	0: Disable Endpoint	1: Enable Endpoint	
				4:	0:	1:	
			R / W	3: EndpointNumber [3]	Endpoint Number		
				2: EndpointNumber [2]			
				1: EndpointNumber [1]			
				0: EndpointNumber [0]			

This register sets the endpoint EPa.

Make sure that the combination of EndpointNumber and INxOUT set for this endpoint does not overlap those for other endpoints.

Bit7 INxOUT

This bit sets the direction of transfer at the endpoint.

Bit6 IntEP_Mode

This bit sets interrupt transfer mode.

Do not set this bit to 1 for a Bulk endpoint.

Setting of this bit varies depending on the endpoint direction (IN or OUT). (The endpoint direction is set by INxOUT in bit 7.)

For the IN direction (INxOUT = 1), set the operation mode of the toggle sequence bit. The toggle sequence operation mode depends on the application. Select either operation mode for an Interrupt IN endpoint.

0: Normal toggle —	Performs a normal toggle sequence.
1: Always toggle —	Always toggles for each transaction performed. For details about this mode, refer to Section 5.7.5 in the USB2.0 Standard.

For the OUT direction (INxOUT = 0), set whether to perform PING flow control at this endpoint. For an Interrupt OUT endpoint, set this bit to 1.

0: Bulk OUT —	Set this for a Bulk OUT endpoint.
1: Interrupt OUT —	Set this for an Interrupt OUT endpoint.

Bit5 EnEndpoint

Setting this bit to 1 enables the endpoint EPa.

When this bit is 0, accesses to the endpoint are ignored.

This bit must be set following the SetConfiguration request from the host.

Bit4 Reserved

Bits3-0 EndpointNumber

Set any endpoint number in the range of 0x1 to 0xF.

7.5.58 133h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	133h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.59 134h D_EPaControl (Device EPa Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	134h	D_EPaControl	R / W	7: AutoForceNAK	0: Do nothing	1: Auto Force NAK	00h
			R / W	6: EnShortPkt	0: Do nothing	1: Enable Short Packet	
			R / W	5: DisAF_NAK_Short	0: Auto Force NAK Short	1 Disable Auto Force	
			R	4: ToggleStat	Toggle sequence bit		
			W	3: ToggleSet	0: Do nothing	1: Set Toggle sequence bit	
			W	2: ToggleClr	0: Do nothing	1: Clear Toggle sequence bit	
			R / W	1: ForceNAK	0: Do nothing	1: Force NAK	
			R / W	0: ForceSTALL	0: Do nothing	1: Force STALL	

This register sets the operation of the endpoint EPa.

Bit7 AutoForceNAK

When transaction at the endpoint EPa is completed without errors, the ForceNAK bit in this register is set to 1.

Bit6 EnShortPkt

Setting this bit to 1 allows the data in the FIFO less than MaxPacketSize to be transmitted as a short packet for an IN transaction at the endpoint EPa. When the IN transaction in which a short packet has been transmitted is complete, this bit is automatically cleared to 0. If a packet equal to MaxPacketSize has been transmitted, this bit is not cleared.

If this bit is set to 1 while no data exists in the FIFO, a packet with zero-length can be transmitted in response to an IN token from the host. If while a packet is being transmitted, data is written to the relevant FIFO after this bit was set, the written data may inadvertently be included in the transmitted packet depending on the timing of the write. Therefore, do not write data to the FIFO until after a packet transmission is completed and this bit is cleared.

Bit5 DisAF_NAK_Short

This bit enable/disables the Auto Force NAK Short function (hereonafter referred to as "AF_NAK_Short"*).

* This function automatically sets the ForceNAK bit to 1 if the packet received in an OUT transaction that completed without errors is a short packet.

The AF_NAK_Short function is by default enabled.

Setting this bit to 1 disables the AF_NAK_Short function.

If the AutoForceNAK bit is set when this bit is set, the AutoForceNAK bit has priority.

Bit4 ToggleStat

This bit shows the status of the toggle sequence bit for the endpoint EPa.

Bit3 ToggleSet

This bit sets the toggle sequence bit for the endpoint EPa to 1. If this bit is set at the same time the ToggleClr bit is set, the function of the ToggleClr bit has priority.

Bit2 ToggleClr

This bit clears the toggle sequence bit for the endpoint EPa to 0. If this bit is set at the same time the ToggleSet bit is set, the function of this bit has priority.

Bit1 ForceNAK

Setting this bit to 1 returns NAK as response for a transaction at the endpoint EPa irrespective of the number of data bytes or the amount of free space in the FIFO.

If a transaction has already been underway when an attempt was made to set this bit to 1, the bit is not set until the transaction is complete, and is set to 1 upon completion of it. If no transactions are under execution, the bit is set to 1 immediately.

Bit0 ForceSTALL

Setting this bit to 1 returns STALL as response for a transaction at the endpoint EPa. This bit has priority over the ForceNAK bit.

If there is any transaction currently executed and this bit is set a certain time after the transaction has started, setting of this bit takes effect beginning with the next transaction.

7.5.60 135h D_EPaJoin (Device End Point a Join)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	135h	D_EPaJoin	R / W	7: JoinIDE	0: Do nothing	1: Join EPa to IDE	00h
			R / W	6: JoinFIFO_Stat	0: Do nothing	1 Join EPa to show Status	
				5:	0:	1:	
				4:	0:	1	
			R / W	3: JoinDMA1	0: Do nothing	1: Join EPa to DMA1	
			R / W	2: JoinDMA0	0: Do nothing	1: Join EPa to DMA0	
			R / W	1: JoinCPU_Rd	0: Do nothing	1: Join EPa to CPU_Rd	
			R / W	0: JoinCPU_Wr	0: Do nothing	1: Join EPa to CPU_Wr	

This register specifies the port with which a data transfer to or from the endpoint EPa is to be performed.

Bit7 JoinIDE

Performs a data transfer for the IDE with the FIFO of endpoint EPa. The transfer direction is determined by the IDE_Control.Dir bit.

Bit6 JoinFIFO_Stat

Allows the Full/Empty/NotEmpty status of the FIFO of endpoint EPa to be monitored via D_FIFO_IntStat.FIFO_Full, D_FIFO_IntStat.FIFO_Empty, and D_FIFO_IntStat.FIFO_NotEmpty.

Bits5-4 Reserved

Bit3 JoinDMA1

Performs a DMA1 transfer with the FIFO of endpoint EPa. The transfer direction is determined by the DMA1_Control.Dir bit.

Bit2 JoinDMA0

Performs a DMA0 transfer with the FIFO of endpoint EPa. The transfer direction is determined by the DMA0_Control.Dir bit.

Bit1 JoinCPU_Rd

Performs a read transfer of CPU register access with the FIFO of endpoint EPa. Namely, when a read of the FIFO_Rd_0,1 registers or FIFO_ByteRd register is performed, data is read from the FIFO of this endpoint.

Bit0 JoinCPU_Wr

Performs a write transfer of CPU register access with the FIFO of endpoint EPa. Namely, when a write to the FIFO_Wr_0,1 registers is performed, data is written to the FIFO of this endpoint.

If the JoinDMAx{x=0,1} bit is set, it is possible to refer to the remaining bytes of data in the FIFO when the DMA0_Control.Dir bit = 1 or the amount of free space in the FIFO when the DMA0_Control.Dir bit = 0 by inspecting the DMAx{x=0,1}_Remain_H,L registers.

If the JoinCPU_Rd or JoinCPU_Wr bit is set, it is possible to write or read data to or from the FIFO_Rd_0,1, FIFO_ByteRd, or FIFO_Wr_0,1 registers after inspecting the FIFO_RdRemain_H,L or FIFO_WrRemain_H,L registers.

The JoinDMA1, JoinDMA0, JoinCPU_Rd, and JoinCPU_Wr bits can be set to 1 only one bit at a time. If an attempt is made to set two or more of these bits by writing 1 at the same time, only the higher bit is set.

7.5.61 136h~13Fh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	136h -13Fh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.62 140h D_EPbMaxSize_H (Device EPb Max Packet Size High)

7.5.63 141h D_EPbMaxSize_L (Device EPb Max Packet Size Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	140h	D_EPbMaxSize_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R / W	1: MaxSize [9]	Endpoint [b] Max Packet Size		
				0: MaxSize [8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	141h	D_EPbMaxSize_L	R / W	7: MaxSize [7]	Endpoint [b] Max Packet Size		00h
				6: MaxSize [6]			
				5: MaxSize [5]			
				4: MaxSize [4]			
				3: MaxSize [3]			
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

These registers set MaxPacketSize.

140h.Bit7-2 Reserved

141h.Bit1-0, 141h.Bit7-3 EPbMaxSize [9:3]

These bits set MaxPacketSize for the endpoint EPb.

To use this endpoint for bulk transfers, set MaxSize to any of the following:

During FS 8, 16, 32, or 64 bytes

During HS 512 bytes

To use this endpoint for interrupt transfers, set MaxSize to any desired transfer bytes within the limits given below:

During FS	Up to 64 bytes
During HS	Up to 512 bytes

141h.Bit-0 Reserved

7.5.64 142h D_EPbConfig_0 (Devie EPb Configuration 0)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	142h	D_EPbConfig_0	R / W	7: INxOUT	0: OUT	1: IN	00h
			R / W	6: IntEP_Mode	0: Normal Toggle (IN) 0: Bulk OUT (OUT)	1: Always Toggle (IN) 1: Interrupt OUT (OUT)	
			R / W	5: EnEndpoint	0: Disable Endpoint	1: Enable Endpoint	
				4:	0:	1:	
			R / W	3: EndpointNumber [3]	Endpoint Number		
				2: EndpointNumber [2]			
				1: EndpointNumber [1]			
				0: EndpointNumber [0]			

This register sets the endpoint EPb.

Make sure that the combination of EndpointNumber and INxOUT set for this endpoint does not overlap those for other endpoints.

Bit7 INxOUT

This bit sets the direction of transfer at the endpoint.

Bit6 IntEP_Mode

This bit sets interrupt transfer mode.

Do not set this bit to 1 for a Bulk endpoint.

Setting of this bit varies with the endpoint direction (IN or OUT). (The endpoint direction is set by INxOUT in bit 7.)

For the IN direction (INxOUT = 1), set the operation mode of the toggle sequence bit. The toggle sequence operation mode depends on the application. Select either operation mode for an Interrupt IN endpoint.

0: Normal toggle —	Performs a normal toggle sequence.
1: Always toggle —	Always toggles for each transaction performed. For details about this mode, refer to Section 5.7.5 in the USB2.0 Standard.

For the OUT direction (INxOUT = 0), set whether to perform PING flow control at this endpoint. For an Interrupt OUT endpoint, set this bit to 1.

0: Bulk OUT —	Set this for a Bulk OUT endpoint.
1: Interrupt OUT —	Set this for an Interrupt OUT endpoint.

Bit5 EnEndpoint

Setting this bit to 1 enables the endpoint EPb.

When this bit is 0, accesses to the endpoint are ignored.

This bit must be set following the SetConfiguration request from the host.

Bit4 Reserved

Bits3-0 EndpointNumber

Set any endpoint number in the range of 0x1 to 0xF.

7.5.65 143h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	143h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.66 144h D_EPbControl (Device EPb Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	144h	D_EPbControl	R / W	7: AutoForceNAK	0: Do nothing	1: Auto Force NAK	00h
			R / W	6: EnShortPkt	0: Do nothing	1: Enable Short Packet	
			R / W	5: DisAF_NAK_Short	0: Auto Force NAK Short	1 Disable Auto Force	
			R	4: ToggleStat	Toggle sequence bit		
			W	3: ToggleSet	0: Do nothing	1: Set Toggle sequence bit	
			W	2: ToggleClr	0: Do nothing	1: Clear Toggle sequence bit	
			R / W	1: ForceNAK	0: Do nothing	1: Force NAK	
			R / W	0: ForceSTALL	0: Do nothing	1: Force STALL	

This register sets the operation of the endpoint EPb.

Bit7 AutoForceNAK

When while this bit is set a transaction at the endpoint EPb is completed without errors, the ForceNAK bit in this register is set to 1.

Bit6 EnShortPkt

Setting this bit to 1 allows the data in the FIFO less than MaxPacketSize to be transmitted as a short packet for an IN transaction at the endpoint EPb. When the IN transaction in which a short packet has been transmitted is complete, this bit is automatically cleared to 0. If a packet equal to MaxPacketSize has been transmitted, this bit is not cleared.

If this bit is set to 1 while no data exists in the FIFO, a zero-length packet can be transmitted in response to an IN token from the host. If while a packet is being transmitted, data is written to the relevant FIFO after this bit was set, the written data may inadvertently be included in the transmitted packet depending on the write timing. Therefore, do not write data to the FIFO until after a packet transmission is completed and this bit is cleared.

Bit5 DisAF_NAK_Short

This bit enable/disables the Auto Force NAK Short function (hereonafter referred to as "AF_NAK_Short"*).

* This function automatically sets the ForceNAK bit to 1 if the packet received in an OUT transaction that completed without errors is a short packet.

The AF_NAK_Short function is by default enabled.

Setting this bit to 1 disables the AF_NAK_Short function.

If the AutoForceNAK bit is set when this bit is set, the AutoForceNAK bit has priority.

Bit4 ToggleStat

This bit shows the status of the toggle sequence bit for the endpoint EPb.

Bit3 ToggleSet

This bit sets the toggle sequence bit for the endpoint EPb to 1. If this bit is set at the same time the ToggleClr bit is set, the function of the ToggleClr bit has priority.

Bit2 ToggleClr

This bit clears the toggle sequence bit for the endpoint EPb to 0. If this bit is set at the same time the ToggleSet bit is set, the function of this bit has priority.

Bit1 ForceNAK

Setting this bit to 1 returns NAK as response for a transaction at the endpoint EPb irrespective of the number of data bytes or the amount of free space in the FIFO.

If a transaction has already been underway when an attempt was made to set this bit to 1, the bit is not set until the transaction is complete, and is set to 1 upon completion of it. If no transactions are under execution, the bit is set to 1 immediately.

Bit0 ForceSTALL

Setting this bit to 1 returns STALL as response for a transaction at the endpoint EPb. This bit has priority over the ForceNAK bit.

If there is any transaction currently executed and this bit is set a certain time after the transaction has started, setting of this bit takes effect beginning with the next transaction.

7.5.67 145h D_EPbJoin (Device End Point b Join)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	145h	D_EPbJoin	R / W	7: JoinIDE	0: Do nothing	1: Join EPb to IDE	00h
			R / W	6: JoinFIFO_Stat	0: Do nothing	1 Join EPb to show Status	
				5:	0:	1:	
				4:	0:	1	
			R / W	3: JoinDMA1	0: Do nothing	1: Join EPb to DMA1	
			R / W	2: JoinDMA0	0: Do nothing	1: Join EPb to DMA0	
			R / W	1: JoinCPU_Rd	0: Do nothing	1: Join EPb to CPU_Rd	
			R / W	0: JoinCPU_Wr	0: Do nothing	1: Join EPb to CPU_Wr	

This register specifies the port with which a data transfer to or from the endpoint EPb is to be performed.

Bit7 JoinIDE

Performs a data transfer for the IDE with the FIFO of endpoint EPb. The transfer direction is determined by the IDE_Control.Dir bit.

Bit6 JoinFIFO_Stat

Allows the Full/Empty/NotEmpty status of the FIFO of endpoint EPb to be monitored via D_FIFO_IntStat.FIFO_Full, D_FIFO_IntStat.FIFO_Empty, and D_FIFO_IntStat.FIFO_NotEmpty.

Bits5-4 Reserved

Bit3 JoinDMA1

Performs a DMA1 transfer with the FIFO of endpoint EPb. The transfer direction is determined by the DMA1_Control.Dir bit.

Bit2 JoinDMA0

Performs a DMA0 transfer with the FIFO of endpoint EPb. The transfer direction is determined by the DMA0_Control.Dir bit.

Bit1 JoinCPU_Rd

Performs a read transfer of CPU register access with the FIFO of endpoint EPb. Namely, when a read of the FIFO_Rd_0,1 registers or FIFO_ByteRd register is performed, data is read from the FIFO of this endpoint.

Bit0 JoinCPU_Wr

Performs a write transfer of CPU register access with the FIFO of endpoint EPb. Namely, when a write to the FIFO_Wr_0,1 registers is performed, data is written to the FIFO of this endpoint.

If the JoinDMAx{x=0,1} bit is set, it is possible to refer to the remaining bytes of data in the FIFO when the DMA0_Control.Dir bit = 1 or the amount of free space in the FIFO when the DMA0_Control.Dir bit = 0 by inspecting the DMAx{x=0,1}_Remain_H,L registers.

If the JoinCPU_Rd or JoinCPU_Wr bit is set, it is possible to write or read data to or from the FIFO_Rd_0,1, FIFO_ByteRd, or FIFO_Wr_0,1 registers after inspecting the FIFO_RdRemain_H,L or FIFO_WrRemain_H,L registers.

The JoinDMA1, JoinDMA0, JoinCPU_Rd, and JoinCPU_Wr bits can be set to 1 only one bit at a time. If an attempt is made to set two or more of these bits by writing 1 at the same time, only the higher bit is set.

7.5.68 146h~14Fh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	146h -14Fh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.69 150h D_EPcMaxSize_H (Device EPc Max Packet Size High)

7.5.70 151h D_EPcMaxSize_L (Device EPc Max Packet Size Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	150h	D_EPcMaxSize_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R / W	1: MaxSize [9]	Endpoint [c] Max Packet Size		
				0: MaxSize [8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	151h	D_EPcMaxSize_L	R / W	7: MaxSize [7]	Endpoint [c] Max Packet Size		00h
				6: MaxSize [6]			
				5: MaxSize [5]			
				4: MaxSize [4]			
				3: MaxSize [3]			
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

These registers set MaxPacketSize.

150h.Bit7-2 Reserved

150h.Bit1-0, 151h.Bit7-3 EPcMaxSize [9:3]

These bits set MaxPacketSize for the endpoint EPc.

To use this endpoint for bulk transfers, set MaxSize to any of the following:

During FS 8, 16, 32, or 64 bytes

During HS 512 bytes

To use this endpoint for interrupt transfers, set MaxSize to any desired transfer bytes within the limits given below:

During FS Up to 64 bytes

During HS Up to 512 bytes

151h.Bit2-0 Reserved

7.5.71 152h D_EPcConfig_0 (Device EPc Configuration 0)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	152h	D_EPcConfig_0	R / W	7: INxOUT	0: OUT	1: IN	00h
			R / W	6: IntEP_Mode	0: Normal Toggle (IN) 0: Bulk OUT (OUT)	1: Always Toggle (IN) 1: Interrupt OUT (OUT)	
			R / W	5: EnEndpoint	0: Disable Endpoint	1: Enable Endpoint	
				4:	0:	1:	
			R / W	3: EndpointNumber [3]	Endpoint Number		
				2: EndpointNumber [2]			
				1: EndpointNumber [1]			
				0: EndpointNumber [0]			

This register sets the endpoint EPc.

Make sure that the combination of EndpointNumber and INxOUT set for this endpoint does not overlap those for other endpoints.

Bit7 INxOUT

This bit sets the direction of transfer at the endpoint.

Bit6 IntEP_Mode

This bit sets interrupt transfer mode.

Do not set this bit to 1 for a Bulk endpoint.

Setting of this bit differs with the endpoint direction (IN or OUT). (The endpoint direction is set by INxOUT in bit 7.)

For the IN direction (INxOUT = 1), set the operation mode of the toggle sequence bit. The toggle sequence operation mode depends on the application. Select either operation mode for an Interrupt IN endpoint.

0: Normal toggle —	Performs a normal toggle sequence.
1: Always toggle —	Always toggles for each transaction performed. For details about this mode, refer to Section 5.7.5 in the USB2.0 Standard.

For the OUT direction (INxOUT = 0), set whether to perform PING flow control at this endpoint. For an Interrupt OUT endpoint, set this bit to 1.

0: Bulk OUT —	Set this for a Bulk OUT endpoint.
1: Interrupt OUT —	Set this for an Interrupt OUT endpoint.

Bit5 EnEndpoint

Setting this bit to 1 enables the endpoint EPc.

When this bit is 0, accesses to the endpoint are ignored.

This bit must be set following the SetConfiguration request from the host.

Bit4 Reserved

Bits3-0 EndpointNumber

Set any endpoint number in the range of 0x1 to 0xF.

7.5.72 153h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	153h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.73 154h D_EPcControl (Device EPc Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	154h	D_EPcControl	R / W	7: AutoForceNAK	0: Do nothing	1: Auto Force NAK	00h
			R / W	6: EnShortPkt	0: Do nothing	1: Enable Short Packet	
			R / W	5: DisAF_NAK_Short	0: Auto Force NAK Short	1 Disable Auto Force	
			R	4: ToggleStat	Toggle sequence bit		
			W	3: ToggleSet	0: Do nothing	1: Set Toggle sequence bit	
			W	2: ToggleClr	0: Do nothing	1: Clear Toggle sequence bit	
			R / W	1: ForceNAK	0: Do nothing	1: Force NAK	
			R / W	0: ForceSTALL	0: Do nothing	1: Force STALL	

This register sets the operation of the endpoint EPc.

Bit7 AutoForceNAK

When while this bit is set and transaction at the endpoint EPc is completed without errors, the ForceNAK bit in this register is set to 1.

Bit6 EnShortPkt

Setting this bit to 1 allows the data in the FIFO less than MaxPacketSize to be transmitted as a short packet for an IN transaction at the endpoint EPc. When the IN transaction in which a short packet has been transmitted is complete, this bit is automatically cleared to 0. If a packet equal to MaxPacketSize has been transmitted, this bit is not cleared.

If this bit is set to 1 while no data exists in the FIFO, a zero-length packet can be transmitted in response to an IN token from the host. If while a packet is being transmitted, data is written to the relevant FIFO after this bit was set, the written data may inadvertently be included in the transmitted packet depending on the write timing. Therefore, do not write data to the FIFO until after a packet transmission is completed and this bit is cleared.

Bit5 DisAF_NAK_Short

This bit enable/disables the Auto Force NAK Short function (hereonafter referred to as "AF_NAK_Short"*).

* This function automatically sets the ForceNAK bit to 1 if the packet received in an OUT transaction that completed without errors is a short packet.

The AF_NAK_Short function is by default enabled.

Setting this bit to 1 disables the AF_NAK_Short function.

If the AutoForceNAK bit is set when this bit is set, the AutoForceNAK bit has priority.

Bit4 ToggleStat

This bit shows the status of the toggle sequence bit for the endpoint EPc.

Bit3 ToggleSet

This bit sets the toggle sequence bit for the endpoint EPc to 1. If this bit is set at the same time the ToggleClr bit is set, the function of the ToggleClr bit has priority.

Bit2 ToggleClr

This bit clears the toggle sequence bit for the endpoint EPc to 0. If this bit is set at the same time the ToggleSet bit is set, the function of this bit has priority.

Bit1 ForceNAK

Setting this bit to 1 returns NAK as response for a transaction at the endpoint EPc irrespective of the number of data bytes or the amount of free space in the FIFO.

If a transaction has already been underway when an attempt was made to set this bit to 1, the bit is not set until the transaction is complete, and is set to 1 upon completion of it. If no transactions are under execution, the bit is set to 1 immediately.

Bit0 ForceSTALL

Setting this bit to 1 returns STALL as response for a transaction at the endpoint EPc. This bit has priority over the ForceNAK bit.

If there is any transaction currently executed and this bit is set a certain time after the transaction has started, setting of this bit takes effect beginning with the next transaction.

7.5.74 155h D_EPcJoin (Device End Point c Join)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	155h	D_EPcJoin	R / W	7: JoinIDE	0: Do nothing	1: Join EPc to IDE	00h
			R / W	6: JoinFIFO_Stat	0: Do nothing	1 Join EPc to show Status	
				5:	0:	1:	
				4:	0:	1	
			R / W	3: JoinDMA1	0: Do nothing	1: Join EPc to DMA1	
			R / W	2: JoinDMA0	0: Do nothing	1: Join EPc to DMA0	
			R / W	1: JoinCPU_Rd	0: Do nothing	1: Join EPc to CPU_Rd	
			R / W	0: JoinCPU_Wr	0: Do nothing	1: Join EPc to CPU_Wr	

This register specifies the port with which a data transfer to or from the endpoint EPc is to be performed.

Bit7 JoinIDE

Performs a data transfer for the IDE with the FIFO of endpoint EPc. The transfer direction is determined by the IDE_Control.Dir bit.

Bit6 JoinFIFO_Stat

Allows the Full/Empty/NotEmpty status of the FIFO of endpoint EPc to be monitored via D_FIFO_IntStat.FIFO_Full, D_FIFO_IntStat.FIFO_Empty, and D_FIFO_IntStat.FIFO_NotEmpty.

Bits5-4 Reserved

Bit3 JoinDMA1

Performs a DMA1 transfer with the FIFO of endpoint EPc. The transfer direction is determined by the DMA1_Control.Dir bit.

Bit2 JoinDMA0

Performs a DMA0 transfer with the FIFO of endpoint EPc. The transfer direction is determined by the DMA0_Control.Dir bit.

Bit1 JoinCPU_Rd

Performs a read transfer of CPU register access with the FIFO of endpoint EPc. Namely, when a read of the FIFO_Rd_0,1 registers or FIFO_ByteRd register is performed, data is read from the FIFO of this endpoint.

Bit0 JoinCPU_Wr

Performs a write transfer of CPU register access with the FIFO of endpoint EPc. Namely, when a write to the FIFO_Wr_0,1 registers is performed, data is written to the FIFO of this endpoint.

If the JoinDMAx{x=0,1} bit is set, it is possible to refer to the remaining bytes of data in the FIFO when the DMA0_Control.Dir bit = 1 or the amount of free space in the FIFO when the DMA0_Control.Dir bit = 0 by inspecting the DMAx{x=0,1}_Remain_H,L registers.

If the JoinCPU_Rd or JoinCPU_Wr bit is set, it is possible to write or read data to or from the FIFO_Rd_0,1, FIFO_ByteRd, or FIFO_Wr_0,1 registers after inspecting the FIFO_RdRemain_H,L or FIFO_WrRemain_H,L registers.

The JoinDMA1, JoinDMA0, JoinCPU_Rd, and JoinCPU_Wr bits can be set to 1 only one bit at a time. If an attempt is made to set two or more of these bits by writing 1 at the same time, only the higher bits are set.

7.5.75 156h~15Fh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	156h -15Fh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.76 160h D_DescAdrs_H (Device Descriptor Address High)

7.5.77 161h D_DescAdrs_L (Device Descriptor Address Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	160h	D_DescAdrs_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
			R / W	3: DescAdrs [11]	Descriptor Address		
				2: DescAdrs [10]			
				1: DescAdrs [9]			
				0: DescAdrs [8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device	161h	D_DescAdrs_L	R / W	7: DescAdrs [7]	Descriptor Address	00h
				6: DescAdrs [6]		
				5: DescAdrs [5]		
				4: DescAdrs [4]		
				3: DescAdrs [3]		
				2: DescAdrs [2]		
				1: DescAdrs [1]		
				0: DescAdrs [0]		

These registers specify the Descriptor Address.

160h.Bit7-4 Reserved

160h.Bit3-0, 161h.Bit7-0 DescAdrs [11:0]

These bits specify the start address of the FIFO from which a descriptor reply operation of the Descriptor Reply function is to start.

Descriptor Address is not intended to allocate a FIFO area to the Descriptor Reply function. No matter how FIFO areas have been set, any address in the entire FIFO area from 0x000 to 0x9FF (2.5 kbytes) can be specified for Descriptor Address.

During descriptor reply, DescAdrs is updated by an amount equal to the transmitted bytes of data each time an IN transaction at the endpoint EP0 is complete. For details about the Descriptor Reply function, refer to the section on ReplyDescriptor of the D_EP0Control register.

Since the FIFO area used for the Descriptor Reply function is not explicitly allocated, make sure this FIFO does not overlap those of other endpoints by setting up the D_DescAdrs_H,L and the D_DescSize_H,L registers properly. Any location between the end address of the reserved area for the endpoint EP0 (0x040) to the start address of the CBW area (0x190) is appropriate.

To inspect Descriptor Address, read D_DescAdrs_H and D_DescAdrs_L in that order.

7.5.78 162h D_DescSize_H (Device Descriptor Size High)

7.5.79 163h D_DescSize_L (Device Descriptor Size Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	162h	D_DescSize_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R / W	1: DescSize [9]	Descriptor Size		
				0: DescSize [8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device	163h	D_DescSize_L	R / W	7: DescSize [7]	DescriptorSize	00h
				6: DescSize [6]		
				5: DescSize [5]		
				4: DescSize [4]		
				3: DescSize [3]		
				2: DescSize [2]		
				1: DescSize [1]		
				0: DescSize [0]		

These registers specify the Descriptor Size.

162h.Bit7-2 Reserved

162h.Bit1-0, 163h.Bit7-0 DescSize [9:0]

For Descriptor Size, specify the total number of data bytes to be sent back to the host in the Descriptor Reply function. For details about the Descriptor Reply function, refer to the section on ReplyDescriptor of the EP0Control register.

No matter how FIFO areas have been set, any size from 0x000 to 0x3FF can be specified for Descriptor Size. During descriptor reply, DescSize is updated by an amount equal to the transmitted bytes of data each time an IN transaction at the endpoint EP0 is complete.

Since the FIFO area used for the Descriptor Reply function is not explicitly allocated, make sure this FIFO does not overlap those of other endpoints by setting up the DescAdrs_H,L and the DescSize_H,L registers properly. Any location between the end address of the reserved area for the endpoint EP0 (0x040) to the start address of the CBW area (0x190) is appropriate.

To inspect Descriptor Size, read DescSize_H and DescSize_L in that order.

7.5.80 164h~16Fh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	164h -16Fh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.81 170h D_DMA0_FIFO_Control (Device DMA0 FIFO Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	170h	D_	R	7: FIFO_Running	0: FIFO is not running	1: FIFO is running	00h
		DMA0_FIFO_Control	R / W	6: AutoEnShort	0: Do nothing	1: Auto Enable Short Packet	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register shows and sets the FIFO status during DMA0 transfer.

Bit7 FIFO_Running

This bit indicates that the FIFO for the endpoint connected to DMA0 is operating now. This bit is set to 1 when DMA0 is activated, and is cleared to 0 when the FIFO is emptied after DMA0 is complete.

Bit6 AutoEnShort

If while this bit is set some bytes of data less than MaxPacketSize remains in the FIFO for any endpoint when DMA0 is complete, the EnShortPkt bit for that endpoint is set to 1.

This bit is effective when the endpoint connected to DMA0 is directed for IN.

Bits5-0 Reserved

7.5.82 171h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	171h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.83 172h D_DMA1_FIFO_Control (Device DMA1 FIFO Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	172h	D_	R	7: FIFO_Running	0: FIFO is not running	1: FIFO is running	00h
		DMA1_FIFO_Control	R / W	6: AutoEnShort	0: Do nothing	1: Auto Enable Short Packet	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register shows and sets the FIFO status during DMA1 transfer.

Bit7 FIFO_Running

This bit indicates that the FIFO for the endpoint connected to DMA1 is operating now. This bit is set to 1 when DMA1 is activated, and is cleared to 0 when the FIFO is emptied after DMA1 is complete.

Bit6 AutoEnShort

If while this bit is set some bytes of data less than MaxPacketSize remains in the FIFO for any endpoint when DMA1 is complete, the EnShortPkt bit for that endpoint is set to 1.

This bit is effective when the endpoint connected to DMA1 is directed for IN.

Bits5-0 Reserved

7.5.84 173h~17Fh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	173h -17Fh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.85 180h~183h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	180h -183h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.86 184h D_EPaStartAdrs_H (Device Endpoint a Start Address High)

7.5.87 185h D_EPaStartAdrs_L (Device Endpoint a Start Address Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	184h	D_EPaStartAdrs_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: StartAdrs[12]	EPa Start Address High		
				3: StartAdrs[11]			
				2: StartAdrs[10]			
				1: StartAdrs[9]			
				0: StartAdrs[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device	185h	D_EPaStartAdrs_L	R / W	7: StartAdrs[7]	EPa Start Address Low	00h
				6: StartAdrs[6]		
				5: StartAdrs[5]		
				4: StartAdrs[4]		
				3: StartAdrs[3]		
				2: StartAdrs[2]		
				1:		
				0:		

These registers set a FIFO area that is used for channel EPa during USB device operation.

184h.Bit7-5 Reserved

184h.Bit4-0, 185h.Bit7-2 StartAdrs[12:2]

These bits set the start address of the FIFO that is allocated to the endpoint EPa.

Since an address value is set with the high-order bits 12–2, address specification needs to be made in 4-byte units.

The area allocated to the channel EPa ranges from a given address to a location preceding 1 byte the address set by EPbStartAdrs.

After setting EPaStartAdrs and EPbStartAdrs, always be sure to set the EPaFIFO_Clr bit in the D_EPrFIFO_Clr register to 1 to clear the FIFO of endpoint EPa.

If MaxSize of EPa is greater than the area set here, the LSI may not operate correctly.

Make sure the sum total of the FIFO areas reserved for all endpoints, the EP0 area, descriptor area, CBW area, and CSW area does not exceed the total size of the internal RAM.

185h.Bit1-0 Reserved

7.5.88 186h~187h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	186h -187h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.89 188h D_EPbStartAdrs_H (Device Endpoint b Start Address High)

7.5.90 189h D_EPbStartAdrs_L (Device Endpoint b Start Address Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	188h	D_EPbStartAdrs_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: StartAdrs[12]	EPb Start Address High		
				3: StartAdrs[11]			
				2: StartAdrs[10]			
				1: StartAdrs[9]			
				0: StartAdrs[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device	189h	D_EPbStartAdrs_L	R / W	7: StartAdrs[7]	EPb Start Address Low	00h
				6: StartAdrs[6]		
				5: StartAdrs[5]		
				4: StartAdrs[4]		
				3: StartAdrs[3]		
				2: StartAdrs[2]		
				1:		
				0:		

These registers set a FIFO area that is used for channel EPb during USB device operation.

188h.Bit7-5 Reserved

188h.Bit4-0, 189h.Bit7-2 StartAdrs[12:2]

These bits set the start address of the FIFO that is allocated to the endpoint EPb.

Since an address value is set with the high-order bits 12–2, address specification needs to be made in 4-byte units.

The area allocated to the channel EPb ranges from a given address to a location preceding 1 byte the address set by EPcStartAdrs.

After setting EPbStartAdrs and EPcStartAdrs, always be sure to set the EPbFIFO_Clr bit in the D_EPrFIFO_Clr register to 1 to clear the FIFO of endpoint EPb.

If MaxSize of EPb is greater than the area set here, the LSI may not operate correctly.

Make sure the sum total of the FIFO areas reserved for all endpoints, the EP0 area, descriptor area, CBW area, and CSW area does not exceed the total size of the internal RAM.

189h.Bit1-0 Reserved

7.5.91 18Ah~18Bh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	18Ah -18Bh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.5.92 18Ch D_EPcStartAdrs_H (Device Endpoint c Start Address High)

7.5.93 18Dh D_EPcStartAdrs_L (Device Endpoint c Start Address Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	18Ch	D_EPcStartAdrs_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: StartAdrs[12]	EPc Start Address High		
				3: StartAdrs[11]			
				2: StartAdrs[10]			
				1: StartAdrs[9]			
				0: StartAdrs[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device	18Dh	D_EPcStartAdrs_L	R / W	7: StartAdrs[7]	EPc Start Address Low	00h
				6: StartAdrs[6]		
				5: StartAdrs[5]		
				4: StartAdrs[4]		
				3: StartAdrs[3]		
				2: StartAdrs[2]		
				1:		
				0:		

These registers set a FIFO area that is used for channel EPc during USB device operation.

18Ch.Bit7-5 Reserved

18Ch.Bit4-0, 18Dh.Bit7-2 StartAdrs[12:2]

These bits set the start address of the FIFO that is allocated to the endpoint EPc.

Since an address value is set with the high-order bits 12–2, address specification needs to be made in 4-byte units.

18Dh.Bit1-0 Reserved

7.5.94 18Eh D_EPcEndAdrs_H (Device Endpoint c End Address High)

7.5.95 18Fh D_EPcEndAdrs_L (Device Endpoint c End Address Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	18Ch	D_EPcEndAdrs_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: EndAdrs[12]	EPc End Address High		
				3: EndAdrs[11]			
				2: EndAdrs[10]			
				1: EndAdrs[9]			
				0: EndAdrs[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Device	18Dh	D_EPcEndAdrs_L	R / W	7: EndAdrs[7]	EPc End Address Low	00h
				6: EndAdrs[6]		
				5: EndAdrs[5]		
				4: EndAdrs[4]		
				3: EndAdrs[3]		
				2: EndAdrs[2]		
				1:		
				0:		

These registers set a FIFO area that is used for channel EPc during USB device operation.

18Ch.Bit7-5 **Reserved**

18Ch.Bit4-0, 18Dh.Bit7-2 **EndAdrs[12:2]**

These bits set the end address of the FIFO that is allocated to the endpoint EPc.

Since an address value is set with the high-order bits 12–2, address specification needs to be made in 64-byte units.

The area allocated to the channel EPc ranges from the address set by EPcStartAdrs to a location preceding 1 byte the address set by EPcEndAdrs.

After setting EPcStartAdrs and EPcEndAdrs, always be sure to set the EPcFIFO_Clr bit in the D_EPrFIFO_Clr register to 1 to clear the FIFO of endpoint EPc.

If MaxSize of EPc is greater than the area set here, the LSI may not operate correctly.

Make sure the sum total of the FIFO areas reserved for all endpoints, the EP0 area, descriptor area, CBW area, and CSW area does not exceed the total size of the internal RAM.

18Dh.Bit1-0 **Reserved**

7.5.96 190h~1FFh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	190h -1FFh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.6 Detailed Description of Host Registers

7.6.1 E0h H_SIE_IntStat_0(Host SIE Interrupt Status 0)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	E0h	H_SIE_IntStat_0		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R (W)	4: DetectCon	0: None	1: Detect Connect	
			R (W)	3: DetectDiscon	0: None	1: Detect Disconnect	
			R (W)	2: DetectRmtWkup	0: None	1: Detect Remote WakeUp	
			R (W)	1: DetectDevChirpOK	0: None	1: Detect Device Chirp OK	
			R (W)	0: DetectDevChirpNG	0: None	1: Detect Device Chirp NG	

This register shows the SIE-related interrupts of the host.

The cause of interrupt indicated by any bit in this register can be cleared by writing 1 to that bit.

Bits7-5 Reserved

Bit4 DetectCon

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a connection of USB cable is detected.

Bit3 DetectDiscon

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a disconnection of USB cable is detected.

This detection function is disabled when H_NegoControl_1.PortSpeed == “HS” and H_NegoControl_0.HostState == “SUSPEND.”

Bit2 DetectRmtWkup

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a Remote Wakeup signal from the USB device is detected during a Suspend state.

Bit1 DetectDevChirpOK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when the chirp signal from the USB device is normal.

Bit0 DetectDevChirpNG

This bit indicates the cause of interrupt directly.

This bit is set to 1 when the chirp signal from the USB device is erratic.

The synchronous bits (bits 4–0) can be read out, but cannot be written to (for clearing the cause of interrupt) during the Active60 and Act_Device state.

To exit the Act_Host state, the processing described below should be executed in firmware to ensure that the interrupt signal XINT will not be asserted inadvertently by these interrupt statuses:.

<When shifting out from the Act_Host state>

1) Process the interrupt status and then clear (H_SIE_IntStat_0.Bit4–0)

2) Disable the interrupt status (H_SIE_IntEnb_0.Bit4–0)

<When shifting to the Act_Host state>

3) Clear the interrupt status (H_SIE_IntStat_0.Bit4–0)

4) Reenable the interrupt status (H_SIE_IntEnb_0.Bit4–0)

7.6.2 E1h H_SIE_IntStat_1(SIE Host Interrupt Status 1)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	E1h	H_SIE_IntStat_1		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1	
				4:	0:	1:	
				3:	0:	1:	
			R (W)	2: ResumeCmp	0: None	1: Resume Complete	
			R (W)	1: SuspendCmp	0: None	1: Suspend Cmplete	
			R (W)	0: ResetCmp	0: None	1: Reset Cmplete	

This register shows the SIE-related interrupts of the host. The cause of interrupt indicated by any bit in this register can be cleared by writing 1 to that bit.

Bits7-4 Reserved

Bit 3 DisabledCmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 if a transition to DISABLED state is completed without errors when the state management function is executed after setting Go_DISABLED in H_NegoControl_0.AutoMode[3:0].

Bit2 ResumeCmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 if a Resume is completed without errors when the state management function is executed after setting Go_RESUME in H_NegoControl_0.AutoMode[3:0].

Bit1 SuspendCmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 if a transition to Suspend is completed without errors when the state management function is executed after setting Go_SUSPEND in H_NegoControl_0.AutoMode[3:0].

Bit0 ResetCmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 if a USB reset is completed without errors when the state management function is executed after setting Go_RESET in H_NegoControl_0.AutoMode[3:0].

The synchronous bits (bits 3–0) can be read out, but cannot be written to (for clearing the cause of interrupt) during the Active60 and Act_Device state.

To exit the Act_Host state, the processing described below should be executed in firmware to ensure that the interrupt signal XINT will not be asserted inadvertently by these interrupt statuses:

When shifting out from the ACT_HOST state:

1) Process the interrupt status and then clear (H_SIE_IntStat_1.Bit3-0).

2) Disable the interrupt status (H_SIE_IntEnb_1.Bit3-0).

When shifting in to the ACT_HOST state:

3) Clear the interrupt status (H_SIE_IntStat_1.Bit3-0).

Enable the interrupt status (H_SIE_IntEnb_1.Bit3-0)

7.6.3 E2h H_FIFO_IntStat(Host FIFO Interrupt Status)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	E2h	H_FIFO_IntStat		7:	0:	1:	00h
			R (W)	6: FIFO_IDE_Cmp	0: None	1: FIFO-IDE Complete	
			R (W)	5: FIFO1_Cmp	0: None	1: FIFO1 Complete	
			R (W)	4: FIFO0_Cmp	0: None	1: FIFO0 Complete	
				3:	0:	1:	
			R (W)	2: FIFO_NotEmpty	0: None	1: FIFO NotEmpty	
			R (W)	1: FIFO_Full	0: None	1: FIFO Full	
			R (W)	0: FIFO_Empty	0: None	1: FIFO Empty	

This register shows the FIFO-related interrupts of the host.

The cause of interrupt indicated by any bit in this register can be cleared by writing 1 to that bit.

Bit7 Reserved

Bit6 FIFO_IDE_Cmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when the FIFO is emptied after IDE transfer is completed while the channel joined to the IDE is directed for IN. If the channel joined to the IDE is directed for OUT, this bit is set to 1 when the IDE transfer is complete.

Bit5 FIFO1_Cmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when the FIFO is emptied after DMA1 transfer is complete while the channel joined to DMA1 is directed for IN. If the channel joined to DMA1 is directed for OUT, this bit is set to 1 when the DMA1 transfer is complete.

Bit4 FIFO0_Cmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when the FIFO is emptied after DMA0 transfer is complete while the channel joined to DMA0 is directed for IN. If the channel joined to DMA0 is directed for OUT, this bit is set to 1 when the DMA0 transfer is complete.

Bit3 Reserved

Bit2 FIFO_NotEmpty

This bit indicates the cause of interrupt directly.

This bit is set to 1 if the FIFO area for the corresponding endpoint contains any data (NotEmpty) while the EPx{x=0,a-e}Join.JoinFIFO_Stat bit is set to 1.

Bit1 FIFO_Full

This bit indicates the cause of interrupt directly.

This bit is set to 1 when the FIFO for the corresponding channel is filled while the H_CHx{x=0,a-e}Join.JoinFIFO_Stat bit = 1.

Bit0 FIFO_Empty

This bit indicates the cause of interrupt directly.

This bit is set to 1 when the FIFO for the corresponding channel is emptied while the H_CHx{x=0,a-e}Join.JoinFIFO_Stat bit = 1.

7.6.4 E3h H_FrameIntStat(Host Frame Interrupt Status)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	E3h	H_FrameIntStat		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R (W)	2: PortErr	0: None	1: Port Error	
			R (W)	1: FrameNumOver	0: None	1: Frame Number Over	
			R (W)	0: SOF	0: None	1: SOF	

This register shows the frame-related interrupts of the host.

The cause of interrupt indicated by any bit in this register can be cleared by writing 1 to that bit.

Bits7-3 Reserved

Bit2 PortErr

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a port is detected during USB host operation.

Bit1 FrameNumOver

This bit indicates the cause of interrupt directly.

This bit is set to 1 when the frame number counter is overflowed (FrameNumber_H register's MSB (bit 2) changed state from 1 to 0). If the FrameNumber_H,L registers are insufficient for the necessary digits of counts, the deficiency can be compensated for by counting occurrences of this interrupt.

Bit0 SOF

This bit indicates the cause of interrupt directly.

This bit is set to 1 in the following cases depending on the transfer speed:

HS: When the host controller transmitted an SOF token in microframe 0

FS: When the host controller transmitted an SOF token

LS: When the host controller transmitted Keepalive

7.6.5 E4h H_CHrIntStat (Host CHr Interrupt Status)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	E4h	H_CHrIntStat		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R	4: H_CHeIntStat	0: None	1: CHe Interrupt	
			R	3: H_CHdIntStat	0: None	1: CHd Interrupt	
			R	2: H_CHcIntStat	0: None	1: CHc Interrupt	
			R	1: H_CHbIntStat	0: None	1: CHb Interrupt	
			R	0: H_CHaIntStat	0: None	1: CHa Interrupt	

This register shows the interrupts of channel CHr.

Bits7-5 Reserved

Bit4 H_CHeIntStat

This bit indicates the cause of interrupt indirectly.

This bit is set to 1 when any bit in the H_CHeIntEnb register corresponding to an interrupt cause in the H_CHeIntStat register is enabled.

Bit3 H_CHdIntStat

This bit indicates the cause of interrupt indirectly.

This bit is set to 1 when any bit in the H_CHdIntEnb register corresponding to an interrupt cause in the H_CHdIntStat register is enabled.

Bit2 H_CHcIntStat

This bit indicates the cause of interrupt indirectly.

This bit is set to 1 when any bit in the H_CHcIntEnb register corresponding to an interrupt cause in the H_CHcIntStat register is enabled.

Bit1 H_CHbIntStat

This bit indicates the cause of interrupt indirectly.

This bit is set to 1 when any bit in the H_CHbIntEnb register corresponding to an interrupt cause in the H_CHbIntStat register is enabled.

Bit0 H_CHaIntStat

This bit indicates the cause of interrupt indirectly.

This bit is set to 1 when any bit in the H_CHaIntEnb register corresponding to an interrupt cause in the H_CHaIntStat register is enabled.

7.6.6 E5h H_CH0IntStat (Host CH0 Interrupt Status)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	E5h	H_CH0IntStat	R (W)	7: TotalSizeCmp	0: None	1: TotalSize Complete	00h
			R (W)	6: TranACK	0: None	1: Tran ACK	
			R (W)	5: TranErr	0: None	1: Tran Error	
			R (W)	4: ChangeCondition	0: None	1: Change Condition	
				3:	0:	1:	
				2:	0:	1:	
			R (W)	1: CTL_SupportCmp	0: None	1: CTL_Support Cmplete	
			R (W)	0: CTL_SupportStop	0: None	1: CTL_Support Stop	

This register shows the interrupt statuses of channel CH0.

The cause of interrupt indicated by any bit in this register can be cleared by writing 1 to that bit.

Bit7 TotalSizeCmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a transfer in a packet transfer in IRP units is completed without errors.

When the control transfer support function is active, this bit is set to 1 when the setup stage, data stage, and status stage each has finished normallyis completed without errors.

Bit6 TranACK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when transfers for the number of individual transactions set in the H_CH0Config_0.ACK_Cnt bits end normally.

When the control transfer support function is active, this bit is set to 1 at only the data stage.

Bit5 TranErr

This bit indicates the cause of interrupt directly.

This bit is set to 1 when an individual transaction has terminated in an error such as a retry error (i.e, time-out error), CRC error, bit stuffing error, PID error (including unexpected PID), or toggle mismatch error.

Bit4 ChangeCondition

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a condition code STALL, Data Overrun or Data Underrun or three consecutive retry errors occurred in a transaction.

This bit is also set to 1 when the H_CH0Config0.TranGo bit was cleared by the firmware. In that case, ConditionCode indicates the result of the last transaction.

Code	Meaning	Description
000	NoError	The transaction is completed without error.
001	Stall	The endpoint has returned stall PID.
010	DataOverrun	• A data packet exceeding MaxPacketSize was received The amount of data received exceeds the maximum allowed size of a data packet (MaxPacketSize). * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error. • Bytes of data exceeding IRP (TotalSize) were received. * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error. * If the received data packet is equal to or less than MaxPacketSize and the data toggle included in it does not match the expected value, this is processed as a toggle mismatch, and not as a data overrun.
011	DataUnderrun	• A data packet smaller than MaxPacketSize was received and the bytes of data in it were less than IRP (TotalSize)The endpoint has returned bytes of data which are less than MaxPacketSize and are insufficient to fill the IRP (TotalSize). * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error.
100	RetryError	• The device does not respond to a token within a designated time (IN) or does not transmit a handshaking signal within a designated time (OUT). • The data packet from the endpoint contains a CRC error. • The data packet from the endpoint contains a bit stuffing error. • The PID inspection bit from the endpoint failed in data PID (IN) or handshaking (OUT). • The received PID is invalid or has no PID values defined. • The data toggle included in the data packet from the endpoint does not match the expected value (toggle mismatch).
Other	Reserved	

Bits3-2 Reserved

Bit1 CTL_SupportCmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when all stages of a control transfer initiated by the control transfer function are completed without errors.

Furthermore, this bit is set to 1 when in a process to deactivate the control transfer support function by clearing the CTL_SupportGo bit in the H_CTL_SupportControl register, the status stage has finished normallyis completed without errors, resulting in a termination of the deactivation process.

Bit0 CTL_SupportStop

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a control transfer initiated by the control transfer function is abnormally terminated during the process.

Furthermore, this bit is set to 1 when in a process to deactivate the control transfer support function by clearing the CTL_SupportGo bit in the H_CTL_SupportControl register, the deactivation process has terminated at other than the status stage or the transaction has terminated in error at the status stage.

7.6.7 E6h H_CHaIntStat (Host CHa Interrupt Status)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	E6h	H_CHaIntStat	R (W)	7: TotalSizeCmp	0: None	1: Tran Complete	00h
			R (W)	6: TranACK	0: None	1: Tran ACK	
			R (W)	5: TranErr	0: None	1: Tran Error	
			R (W)	4: ChangeCondition	0: None	1: Change Condition	
				3:	0:	1:	
				2:	0:	1:	
			R (W)	1: BO_SupportCmp	0: None	1: BO Support Cmplete	
			R (W)	0: BO_SupportStop	0: None	1: BO Support Stop	

This register shows the interrupt statuses of channel CHa.

The cause of interrupt indicated by any bit in this register can be cleared by writing 1 to that bit.

Bit7 TotalSizeCmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a transfer in a packet transfer in IRP units is completed without errors.

When the bulk-only support function is active, this bit is set to 1 when a CBW transport, data transport, or CSW transport has finished normallyis completed without errors.

Bit6 TranACK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when transfers for the number of individual transactions set in the H_CHaConfig_0.ACK_Cnt bits end normally.

When the bulk-only support function is active, this bit is set to 1, only in only a data transport.

Bit5 TranErr

This bit indicates the cause of interrupt directly.

This bit is set to 1 when an individual transaction has terminated in an error such as a retry error (i.e, time-out error), CRC error, bit stuffing error, PID error (including unexpected PID), or toggle mismatch error.

Bit4 ChangeCondition

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a condition code STALL, Data Overrun or Data Underrun or three consecutive retry errors occurred in a transaction.

This bit is also set to 1 when the H_CHaConfig0.TranGo bit is cleared by the firmware. In that case, ConditionCode indicates the result of the last transaction.

Code	Meaning	Description
000	NoError	The transaction is completed without error.
001	Stall	The endpoint has returned stall PID.
010	DataOverrun	• A data packet exceeding MaxPacketSize was received The amount of data received exceeds the maximum allowed size of a data packet (MaxPacketSize). * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error. • Bytes of data exceeding IRP (TotalSize) were received. * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error. * If the received data packet is equal to or less than MaxPacketSize and the data toggle included in it does not match the expected value, this is processed as a toggle mismatch, and not as a data overrun.
011	DataUnderrun	• A data packet smaller than MaxPacketSize was received and the bytes of data in it were less than IRP (TotalSize)The endpoint has returned bytes of data which are less than MaxPacketSize and are insufficient to fill the IRP (TotalSize). * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error.
100	RetryError	• The device does not respond to a token within a designated time (IN) or does not transmit a handshaking signal within a designated time (OUT). • The data packet from the endpoint contains a CRC error. • The data packet from the endpoint contains a bit stuffing error. • The PID inspection bit from the endpoint failed in data PID (IN) or handshaking (OUT). • The received PID is invalid or has no PID values defined. • The data toggle included in the data packet from the endpoint does not match the expected value (toggle mismatch).
Other	Reserved	

Bits3-2 Reserved

Bit1 BO_SupportCmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a status transport in transfers initiated by the bulk-only support function is completed without errors.

Furthermore, this bit is set to 1 when in a process to deactivate the bulk-only support function by clearing the BO_SupportGo bit in the H_BO_SupportControl register, a CSW transport has finished normallyis completed without errors, resulting in a termination of the deactivation process.

Bit0 BO_SupportStop1

This bit indicates the cause of interrupt directly.

This bit is set to 1 when any transfer in transfers initiated by the bulk-only support function is abnormally terminated.

Furthermore, this bit is set to 1 when in a process to deactivate the bulk-only support function by clearing the BO_SupportGo bit in the H_BO_SupportControl register, the deactivation process has terminated in other than a CSW transport or an error was detected in a CSW transport.

7.6.8 E7h H_CHbIntStat (Host CHb Interrupt Status)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	E7h	H_ChbIntStat	R (W)	7: TotalSizeCmp	0: None	1: Tran Complete	00h
			R (W)	6: TranACK	0: None	1: Tran ACK	
			R (W)	5: TranErr	0: None	1: Tran Error	
			R (W)	4: ChangeCondition	0: None	1: Change Condition	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register shows the interrupt statuses of channel CHb.

The cause of interrupt indicated by any bit in this register can be cleared by writing 1 to that bit.

Bit7 TotalSizeCmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a transfer in a packet transfer in IRP units is completed without errors.

Bit6 TranACK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when transfers for the number of individual transactions set in the H_CHbConfig_0.ACK_Cnt bits end normally.

Bit5 TranErr

This bit indicates the cause of interrupt directly.

This bit is set to 1 when an individual transaction has terminated in an error such as a retry error (i.e, time-out error), CRC error, bit stuffing error, PID error (including unexpected PID), or toggle mismatch error.

Bit4 ChangeCondition

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a condition code STALL, Data Overrun or Data Underrun or three consecutive retry errors occurred in a transaction.

This bit is also set to 1 when the H_CHbConfig0.TranGo bit is cleared by the firmware. In that case, ConditionCode indicates the result of the last transaction.

Code	Meaning	Description
000	NoError	The transaction is completed without error.
001	Stall	The endpoint has returned stall PID.
010	DataOverrun	• The amount of data received exceeds the maximum allowed size of a data packet (MaxPacketSize). * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error. • The amount of data received exceeds the maximum allowed IRP (TotalSize) * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error. * If the received data packet is equal to or less than MaxPacketSize and the data toggle included in it does not match the expected value, this is processed as a toggle mismatch, and not as a data overrun.
011	DataUnderrun	• The endpoint has returned bytes of data which are less than MaxPacketSize and are insufficient to fill the IRP (TotalSize). * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error.
100	RetryError	• The device does not respond to a token within a designated time (IN) or does not transmit a handshaking signal within a designated time (OUT). • The data packet from the endpoint contains a CRC error. • The data packet from the endpoint contains a bit stuffing error. • The PID inspection bit from the endpoint failed in data PID (IN) or handshaking (OUT). • The received PID is invalid or has no PID values defined. • An ERR handshaking signal was received in a split transaction of interrupt transfer. • Three NYET handshaking signals were received consecutively in a split transaction of interrupt transfer. • The data toggle included in the data packet from the endpoint does not match the expected value (toggle mismatch).
Other	Reserved	

Bits3-0 Reserved

7.6.9 E8h H_CHcIntStat (Host CHc Interrupt Status)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	E8h	H_CHcIntStat	R (W)	7: TotalSizeCmp	0: None	1: Tran Complete	00h
			R (W)	6: TranACK	0: None	1: Tran ACK	
			R (W)	5: TranErr	0: None	1: Tran Error	
			R (W)	4: ChangeCondition	0: None	1: Change Condition	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register shows the interrupt statuses of channel CHc.

The cause of interrupt indicated by any bit in this register can be cleared by writing 1 to that bit.

Bit7 TotalSizeCmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a transfer in a packet transfer in IRP units is completed without errors.

Bit6 TranACK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when transfers for the number of individual transactions set in the H_CHcConfig_0.ACK_Cnt bits end normally.

Bit5 TranErr

This bit indicates the cause of interrupt directly.

This bit is set to 1 when an individual transaction has terminated in an error such as a retry error (i.e, time-out error), CRC error, bit stuffing error, PID error (including unexpected PID), or toggle mismatch error.

Bit4 ChangeCondition

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a condition code STALL, Data Overrun or Data Underrun or three consecutive retry errors occurred in a transaction.

This bit is also set to 1 when the H_CHcConfig0.TranGo bit is cleared by the firmware. In that case, ConditionCode indicates the result of the last transaction.

Code	Meaning	Description
000	NoError	The transaction is completed without error.
001	Stall	The endpoint has returned stall PID.
010	DataOverrun	• The amount of data received exceeds the maximum allowed size of a data packet (MaxPacketSize). * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error. • The amount of data received exceeds the maximum allowed IRP (TotalSize) * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error. * If the received data packet is equal to or less than MaxPacketSize and the data toggle included in it does not match the expected value, this is processed as a toggle mismatch, and not as a data overrun.
011	DataUnderrun	• The endpoint has returned bytes of data which are less than MaxPacketSize and are insufficient to fill the IRP (TotalSize). * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error.
100	RetryError	• The device does not respond to a token within a designated time (IN) or does not transmit a handshaking signal within a designated time (OUT). • The data packet from the endpoint contains a CRC error. • The data packet from the endpoint contains a bit stuffing error. • The PID inspection bit from the endpoint failed in data PID (IN) or handshaking (OUT). • The received PID is invalid or has no PID values defined. • An ERR handshaking signal was received in a split transaction of interrupt transfer. • Three NYET handshaking signals were received consecutively in a split transaction of interrupt transfer. • The data toggle included in the data packet from the endpoint does not match the expected value (toggle mismatch).
Other	Reserved	

Bits3-0 Reserved

7.6.10 E9h H_CHdIntStat (Host CHd Interrupt Status)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	E9h	H_CHdIntStat	R (W)	7: TotalSizeCmp	0: None	1: Tran Complete	00h
			R (W)	6: TranACK	0: None	1: Tran ACK	
			R (W)	5: TranErr	0: None	1: Tran Error	
			R (W)	4: ChangeCondition	0: None	1: Change Condition	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register shows the interrupt statuses of channel CHd. The cause of interrupt indicated by any bit in this register can be cleared by writing 1 to that bit.

Bit7 TotalSizeCmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a transfer in a packet transfer in IRP units is completed without errors.

Bit6 TranACK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when transfers for the number of individual transactions set in the H_CHdConfig_0.ACK_Cnt bits end normally.

Bit5 TranErr

This bit indicates the cause of interrupt directly.

This bit is set to 1 when an individual transaction has terminated in an error such as a retry error (i.e, time-out error), CRC error, bit stuffing error, PID error (including unexpected PID), or toggle mismatch error.

Bit4 ChangeCondition

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a condition code STALL, Data Overrun or Data Underrun or three consecutive retry errors occurred in a transaction.

This bit is also set to 1 when the H_CHdConfig0.TranGo bit is cleared by the firmware. In that case, ConditionCode indicates the result of the last transaction.

Code	Meaning	Description
000	NoError	The transaction is completed without error.
001	Stall	The endpoint has returned stall PID.
010	DataOverrun	• The amount of data received exceeds the maximum allowed size of a data packet (MaxPacketSize). * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error. • The amount of data received exceeds the maximum allowed IRP (TotalSize) * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error. * If the received data packet is equal to or less than MaxPacketSize and the data toggle included in it does not match the expected value, this is processed as a toggle mismatch, and not as a data overrun.
011	DataUnderrun	• The endpoint has returned bytes of data which are less than MaxPacketSize and are insufficient to fill the IRP (TotalSize). * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error.
100	RetryError	• The device does not respond to a token within a designated time (IN) or does not transmit a handshaking signal within a designated time (OUT). • The data packet from the endpoint contains a CRC error. • The data packet from the endpoint contains a bit stuffing error. • The PID inspection bit from the endpoint failed in data PID (IN) or handshaking (OUT). • The received PID is invalid or has no PID values defined. • An ERR handshaking signal was received in a split transaction of interrupt transfer. • Three NYET handshaking signals were received consecutively in a split transaction of interrupt transfer. • The data toggle included in the data packet from the endpoint does not match the expected value (toggle mismatch).
Other	Reserved	

Bits3-0 Reserved

7.6.11 EAh H_CHeIntStat (Host CHe Interrupt Status)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	EAh	H_CHeIntStat	R (W)	7: TotalSizeCmp	0: None	1: Tran Complete	00h
			R (W)	6: TranACK	0: None	1: Tran ACK	
			R (W)	5: TranErr	0: None	1: Tran Error	
			R (W)	4: ChangeCondition	0: None	1: Change Condition	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register shows the interrupt statuses of channel CHe.

The cause of interrupt indicated by any bit in this register can be cleared by writing 1 to that bit.

Bit7 TotalSizeCmp

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a transfer in a packet transfer in IRP units is completed without errors.

Bit6 TranACK

This bit indicates the cause of interrupt directly.

This bit is set to 1 when transfers for the number of individual transactions set in the H_CHeConfig_0.ACK_Cnt bits end normally.

Bit5 TranErr

This bit indicates the cause of interrupt directly.

This bit is set to 1 when an individual transaction has terminated in an error such as a retry error (i.e, time-out error), CRC error, bit stuffing error, PID error (including unexpected PID), or toggle mismatch error.

Bit4 ChangeCondition

This bit indicates the cause of interrupt directly.

This bit is set to 1 when a condition code STALL, Data Overrun or Data Underrun or three consecutive retry errors occurred in a transaction.

This bit is also set to 1 when the H_CHeConfig0.TranGo bit is cleared by the firmware. In that case, ConditionCode indicates the result of the last transaction.

Code	Meaning	Description
000	NoError	The transaction is completed without error.
001	Stall	The endpoint has returned stall PID.
010	DataOverrun	• The amount of data received exceeds the maximum allowed size of a data packet (MaxPacketSize). * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error. • The amount of data received exceeds the maximum allowed IRP (TotalSize) * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error. * If the received data packet is equal to or less than MaxPacketSize and the data toggle included in it does not match the expected value, this is processed as a toggle mismatch, and not as a data overrun.
011	DataUnderrun	• The endpoint has returned bytes of data which are less than MaxPacketSize and are insufficient to fill the IRP (TotalSize). * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error.
100	RetryError	• The device does not respond to a token within a designated time (IN) or does not transmit a handshaking signal within a designated time (OUT). • The data packet from the endpoint contains a CRC error. • The data packet from the endpoint contains a bit stuffing error. • The PID inspection bit from the endpoint failed in data PID (IN) or handshaking (OUT). • The received PID is invalid or has no PID values defined. • An ERR handshaking signal was received in a split transaction of interrupt transfer. • Three NYET handshaking signals were received consecutively in a split transaction of interrupt transfer. • The data toggle included in the data packet from the endpoint does not match the expected value (toggle mismatch).
Other	Reserved	

Bits3-0 Reserved

7.6.12 Ebh~EFh Reserved()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	EBh -EFh	Reserved		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.6.13 F0h H_SIE_IntEnb0 (Host SIE Interrupt Enable)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	F0h	H_SIE_IntEnb0		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: EnDetectCon	0: Disable	1: Enable	
			R / W	3: EnDetectDiscon	0: Disable	1: Enable	
			R / W	2: EnDetectRmtWkup	0: Disable	1: Enable	
			R / W	1: EnDetectDevChirpOK	0: Disable	1: Enable	
			R / W	0: EnDetectDevChirpNG	0: Disable	1: Enable	

This register is used to enable or disable the assertion of the H_SIE_IntStat_0 bit in the HostIntStat register for the interrupt causes accommodated in the H_SIE_IntStat_0 register. The synchronous bits (bits 3–0) can be read out, but cannot be written to during the Active60 and Act_Device state. For the processing to be executed on these synchronous bits when shifting out from the Act_Host state, refer to the description of the H_SIE_IntStat_0 register.

7.6.14 F1h H_SIE_IntEnb_1(SIE Host Interrupt Enable 1)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	F1h	H_SIE_IntEnb1		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1	
				4:	0:	1: F	
			R / W	3: EnDisabledCmp	0: Disable	1: Enable	
			R / W	2: EnResumeCmp	0: Disable	1: Enable	
			R / W	1: EnSuspendCmp	0: Disable	1: Enable	
			R / W	0: EnResetCmp	0: Disable	1: Enable	

This register is used to enable or disable the assertion of the H_SIE_IntStat_1 bit in the HostIntStat register for the interrupt causes accommodated in the H_SIE_IntStat_1 register.

The synchronous bits (bits 3–0) can be read out, but cannot be written to during the Active60 and Act_Device state. For the processing to be executed on these synchronous bits when shifting out from the Act_Host state, refer to the description of the H_SIE_IntStat_1 register.

7.6.15 F2h H_FIFO_IntEnb(Host FIFO Interrupt Enable)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	F2h	H_FIFO_IntEnb		7:	0:	1:	00h
			R / W	6: EnFIFO_IDE_Cmp	0: Disable	1: Enable	
			R / W	5: EnFIFO1_Cmp	0: Disable	1: Enable	
			R / W	4: EnFIFO0_Cmp	0: Disable	1: Enable	
				3:	0:	1:	
			R / W	2: EnFIFO_NotEmpty	0: Disable	1: Enable	
			R / W	1: EnFIFO_Full	0: Disable	1: Enable	
			R / W	0: EnFIFO_Empty	0: Disable	1: Enable	

This register is used to enable or disable the assertion of the H_FIFO_IntStat bit in the HostIntStat register for the interrupt causes accommodated in the H_FIFO_IntStat register.

7.6.16 F3h H_FrameIntEnb(Host Frame Interrupt Enable)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	F3h	H_FrameIntEnb		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R / W	2: EnPortErr	0: Disable	1: Enable	
			R / W	1: EnFrameNumOver	0: Disable	1: Enable	
			R / W	0: EnSOF	0: Disable	1: Enable	

This register is used to enable or disable the assertion of the H_FrameIntStat bit in the HostIntStat register for the interrupt causes accommodated in the H_FrameIntStat register.

7.6.17 F4h H_CHrIntEnb(Host CHr Interrupt Enable)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	F4h	H_CHrIntEnb		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: EnH_EnCHeIntStat	0: Disable	1: Enable	
			R / W	3: EnH_EnCHdIntStat	0: Disable	1: Enable	
			R / W	2: EnH_EnCHcIntStat	0: Disable	1: Enable	
			R / W	1: EnH_EnCHbIntStat	0: Disable	1: Enable	
			R / W	0: EnH_EnCHaIntStat	0: Disable	1: Enable	

This register is used to enable or disable the assertion of the H_CHrIntStat bit in the HostIntStat register for the interrupt causes accommodated in the H_CHrIntStat register.

7.6.18 F5h H_CH0IntEnb(Host CH0 Interrupt Enable)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	F5h	H_CH0IntEnb	R / W	7: EnTotalSizeCmp	0: Disable	1: Enable	00h
			R / W	6: EnTranACK	0: Disable	1: Enable	
			R / W	5: EnTranErr	0: Disable	1: Enable	
			R / W	4: EnChangeCondition	0: Disable	1: Enable	
				3:	0:	1:	
				2:	0:	1:	
			R / W	1: EnCTL_SupportCmp	0: Disable	1: Enable	
			R / W	0: EnCTL_SupportStop	0: Disable	1: Enable	

This register is used to enable or disable the assertion of the H_CH0IntStat bit in the HostIntStat register for the interrupt causes accommodated in the H_CH0IntStat register.

7.6.19 F6h H_CHaIntEnb (Host CHa Interrupt Enable)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	F6h	H_CHaIntEnb	R / W	7: EnTotalSizeCmp	0: Disable	1: Enable	00h
			R / W	6: EnTranACK	0: Disable	1: Enable	
			R / W	5: EnTranErr	0: Disable	1: Enable	
			R / W	4: EnChangeCondition	0: Disable	1: Enable	
				3:	0:	1:	
				2:	0: Disable	1:	
			R / W	1: EnBO_Support_Cmp	0: Disable	1: Enable	
			R / W	0: EnBO_Support_Stop	0: Disable	1: Enable	

This register is used to enable or disable the assertion of the CHaIntStat bit in the H_CHrIntStat register for the interrupt causes accommodated in the H_CHaIntStat register.

7.6.20 F7h H_CHbIntEnb (Host CHb Interrupt Enable)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	F7h	H_CHbIntEnb	R / W	7: EnTotalSizeCmp	0: Disable	1: Enable	00h
			R / W	6: EnTranACK	0: Disable	1: Enable	
			R / W	5: EnTranErr	0: Disable	1: Enable	
			R / W	4: EnChangeCondition	0: Disable	1: Enable	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register is used to enable or disable the assertion of the CHbIntStat bit in the H_CHrIntStat register for the interrupt causes accommodated in the H_CHbIntStat register.

7.6.21 F8h H_CHcIntEnb (Host CHc Interrupt Enable)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	F8h	H_CHcIntEnb	R / W	7: EnTotalSizeCmp	0: Disable	1: Enable	00h
			R / W	6: EnTranACK	0: Disable	1: Enable	
			R / W	5: EnTranErr	0: Disable	1: Enable	
			R / W	4: EnChangeCondition	0: Disable	1: Enable	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register is used to enable or disable the assertion of the CHcIntStat bit in the H_CHrIntStat register for the interrupt causes accommodated in the H_CHcIntStat register.

7.6.22 F9h H_CHdIntEnb (Host CHd Interrupt Enable)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	F9h	H_CHdIntEnb	R / W	7: EnTotalSizeCmp	0: Disable	1: Enable	00h
			R / W	6: EnTranACK	0: Disable	1: Enable	
			R / W	5: EnTranErr	0: Disable	1: Enable	
			R / W	4: EnChangeCondition	0: Disable	1: Enable	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register is used to enable or disable the assertion of the CHdIntStat bit in the H_CHrIntStat register for the interrupt causes accommodated in the H_CHdIntStat register.

7.6.23 FAh H_CHeIntEnb (Host CHe Interrupt Enable)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	FAh	H_CHeIntEnb	R / W	7: EnTotalSizeCmp	0: Disable	1: Enable	00h
			R / W	6: EnTranACK	0: Disable	1: Enable	
			R / W	5: EnTranErr	0: Disable	1: Enable	
			R / W	4: EnChangeCondition	0: Disable	1: Enable	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register is used to enable or disable the assertion of the CHeIntStat bit in the H_CHrIntStat register for the interrupt causes accommodated in the H_CHeIntStat register.

7.6.24 FBh~FFh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	FEh -FFh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.6.25 100h *H_Reset (Host Reset)*

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	100h	***H_Reset***		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
			R / W	0: ***ResetHTM***	0: Reset DTM	1: Do nothing	

This register resets the host transceiver macro.

This register can be accessed even during Sleep/Snooze.

Bits7-1 Reserved

Bit0 ResetHTM

Setting this bit to 1 initializes the host transceiver macro of the LSI.

To deassert the reset, clear this bit to 0.

7.6.26 101h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	101h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.6.27 102h H_NegoControl_0 (Host NegoControl 0)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	102h	H_NegoControl_0	R / W	7: AutoModeCancel	0: None	1: Cancel	1Xh
			R	6: HostState[2]	HostState[2:0]		
			R	5: HostState[1]			
			R	4: HostState[0]			
			W	3: AutoMode[3]	AutoMode[3:0]		
			W	2: AutoMode[2]			
			W	1: AutoMode[1]			
			W	0: AutoMode[0]			

This register sets the operations associated with host negotiation.

Bit7 AutoModeCancel

Setting this bit to 1 causes execution of the host state management support function to stop, with thewhile the LSI kept in thatretains that state (settings of H_NegoControl_0.AutoMode and H_XcvrControl are retained, the signal line state is retained, the internal timer is deactivated, and the connect/disconnect, device Chirp, and remote wakeup detection functions each are turned off).

Before the following operations can be performed, execution of the host state management support function must be stopped by setting this bit.

- To change the host state to the IDLE state
- To change the host state to the DISABLED state without waiting for a reset complete status (H_SIE_IntStat_1.ResetCmp) to be issued after detecting an error in Chirp from the device
- To execute test mode by setting H_USB_Test.EnHS_Test

Setting this bit to 1 causes execution of the host state management support function to stop, and this bit is cleared to 0 upon completion of the stopping process (approximately 6 clock cycles required when operating at 60 MHz). In the above case, be sure to confirm that this bit has beenis cleared to 0 before setting GoIDLE or GoDISABLE in H_NegoControl_0.AutoMode or setting H_USB_Test.EnHS_Test.

Bits6-4 HostState[2:0]

These bits indicate the current host state while the host state management support function is under execution.

The host state is one of the following:

000: Reserved

001: IDLE

010: WAIT_CONNECT

011: DISABLED

100: USB_RESET

101: USB_OPERATIONAL

110: USB_SUSPEND

111: USB_RESUME

Bits3-0 AutoMode[3:0]

These bits set a host state to which the LSI is to be placed in by execution of the host state management support function.

One of the following host states can be set:

0001: GoIDLE (transitions to IDLE state)

0010: GoWAIT_CONNECT (transitions to WAIT_CONNECT state)

0011: GoDISABLED (transitions to DISABLED state)

0100: GoRESET (transitions to RESET state)

0101: GoOPERATIONAL (transitions to OPERATIONAL state)

0110: GoSUSPEND (transitions to SUSPEND state)

0111: GoRESUME (transitions to RESUME state)

1001: GoWAIT_CONNECTtoDIS (transitions from WAIT_CONNECT to DISABLED state consecutively)

1010: GoWAIT_CONNECTtoOP (transitions from WAIT_CONNECT to OPERATIONAL state consecutively)

1100: GoRESETtoOP (transitions from RESET to OPERATIONAL state consecutively)

1110: GoSUSPENDtoOP (transitions from SUSPEND to OPERATIONAL state consecutively)

1111: GoRESUMEtoOP (transitions from RESUME to OPERATIONAL state consecutively)

Other than above: Reserved

To place the LSI from a given state into the IDLE state (by executing GoIDLE), execute the procedure described below:

- Write 0x80 to the H_NegoControl register
- Check to see that the H_NegoControl_0.AutoModeCancel bit is cleared to 0
- Write 0x01 to the H_NegoControl register

7.6.28 103h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	103h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.6.29 104h H_NegoControl_1 (Host NegoControl 1)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	104h	H_NegoControl_1		7:	0:	1:	10h
				6:	0:	1:	
			R / W	5: PortSpeed[1]	PortSpeed[1:0]		
			R / W	4: PortSpeed[0]			
				3:	0:	1:	
				2:	0:	1:	
			R / W	1: DisChirpFinish	0: Normal	1: DisableChirpFinish	
			R / W	0: RmtWkupDetEnb	0: Disable	1: Enable	

This register sets the operations associated with host negotiation.

Note: The Reset value for this register indicates the value read during ACT_HOST. In other states, the Reset value is read as 00h.

Bits7-6 Reserved

Bits5-4 PortSpeed[1:0]

These bits indicate the transfer speed.

00: High Speed

01: Full Speed

10: Reserved

11: Low Speed

Bits3-2 Reserved

Bit1 DisChirpFinish

This bit sets an operation mode to be assumed when a device Chirp is not completed in a designated time.

0: After flagging a device Chirp error status, the LSI sends out USB Reset for a designated duration to complete USB Reset.

1: After flagging a device Chirp error status, the LSI keeps waiting for the device Chirp to complete, and upon completion of it, finishes USB Reset after executing a host Chirp.

Bit0 RmtWkupDetEnb

This bit enable/disables the remote wakeup detection function.

7.6.30 105h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	105h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.6.31 106h H_USB_Test (Host USB_Test)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	106h	H_USB_Test	R / W	7: EnHS_Test	0: Do nothing	1: EnHS_Test	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: Test_Force_Enable	0: Do nothing	1: TestForceEnable	
			R / W	3: Test_SE0_NAK	0: Do nothing	1: Test_SE0_NAK	
			R / W	2: Test_J	0: Do nothing	1: Test_J	
			R / W	1: Test_K	0: Do nothing	1: Test_K	
			R / W	0: Test_Packet	0: Do nothing	1: Test_Packett	

This register sets the operations relating to USB2.0 test mode during USB host operation.

Test mode can be executed in any of WAIT_CONNECT, DISABLED, or SUSPEND states.

Before the LSI can be shifted from these states into test mode, processing in either state must be terminated. To shift to test mode, execute the procedure described below.

- Set the TranGo bit for all channels (H_CHx{x=0,a-e}Config_0.TranGo), H_CTL_SupportControl.CTL_SupportGo, and H_BO_SupportControl.BOSupportGo all to 0.
- Write 0x80 to the H_NegoControl register.
- Check to see that the H_NegoControl_0.AutoModeCancel bit is cleared to 0.
- Set any of the five low-order bits in this register and EnHS_Test to 1 concurrently.

Furthermore, to switch from one test mode to another or terminate a test mode, write 0x00 to this register. Test mode will be terminated, and the host state will shift to IDLE.

Bit7 EnHS_Test

When this bit and any of the five low-order bits in the H_USB_Test register are set to 1 concurrently, the LSI enters the test mode corresponding to that low-order bit.

Bits6-5 Reserved

Bit4 TestForceEnable

The LSI can shift to the TestForceEnable test mode, by setting this bit and the En_HS_Test bit to 1 concurrently. In this test mode, the host port can be disconnected by sending out an SOF in HS mode.

Bit3 Test_SE0_NAK

The LSI shifts to the Test_SE0_NAK test mode, by setting this bit and the En_HS_Test bit to 1 concurrently. In this test mode, the host port becomes ready to receive in HS mode.

Bit2 TEST_J

The LSI shifts to the Test_J test mode, by setting this bit and the En_HS_Test bit to 1 concurrently. In this test mode, the host port sends out "J" in HS mode.

Bit1 TEST_K

The LSI shifts to the Test_K test mode, by setting this bit and the En_HS_Test bit to 1 concurrently. In this test mode, the host port sends out “K” in HS mode.

Bit0 Test_Packet

The LSI shifts to the Test_Packet test mode by setting this bit to 1 and then the EnHS_Test bit to 1. This test mode can only be used on channel CH0. Before entering this test mode, set the FIFO area for CH0 to 64 bytes, clear the FIFO, and write the following data for a test packet to the FIFO. There are no other settings required for CH0.

The following 53 bytes are the data to be written to the FIFO during packet transmission test mode:

00h, 00h, 00h, 00h, 00h, 00h, 00h, 00h,

00h, AAh, AAh, AAh, AAh, AAh, AAh, AAh,

AAh, EEh, EEh, EEh, EEh, EEh, EEh, EEh,

EEh, FEh, FFh, FFh, FFh, FFh, FFh, FFh,

FFh, FFh, FFh, FFh, FFh, 7Fh, BFh, DFh,

EFh, F7h, FBh, FDh, FCh, 7Eh, BFh, DFh,

EFh, F7h, FBh, FDh, 7Eh

Since the SIE adds PID and CRC to a test packet when it is transmitted, the data to be written to the FIFO should consist of only a range of the test packet data stipulated in USB standard Rev. 2.0 from the data next to DATA0 PID to those that follow but not including CRC16.

7.6.32 107h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	107h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.6.33 108h H_CHnControl (Host CHr FIFO Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	108h	H_CHnFIFOControl		7:	0:	1:	XXh
				6:	0:	1:	
			W	5: AllFIFO_Clr	0: Do nothing	1: Clear All FIFO	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
			W	0: CH0FIFO_Clr	0: Do nothing	1: Clear EP0 FIFO	

This register sets the channel operations. This is a write-only register.

Bits7-6 Reserved

Bit5 AllFIFO_Clr

This bit clears the FIFOs for all channels. When areas for the respective channels are set, temporarily set this bit to 1 to clear the FIFOs for all channels after completing the area settings. This bit is automatically cleared when the FIFOs are cleared.

When DMAx{x=0,1} has been joined to any channel and the relevant DMA is active (when DMA_Running bit = 1), do not set the bit for that channel to 1.

Bits4-1 Reserved

Bit0 CH0FIFO_Clr

This bit clears the FIFO for the channel CH0.

When set to 1, this bit only clears the FIFO and does not hold the value set in it.

When DMAx{x=0,1} are joined to the channel CH0 and the relevant DMA is active (when DMA_Running bit = 1), do not set this bit to 1.

7.6.34 109h H_CHrFIFO_Clr (Host CHr FIFO Clear)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	109h	H_CHrFIFO_Clr		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
			W	4: CHeFIFO_Clr	0: Do nothing	1: Clear CHe FIFO	
			W	3: CHdFIFO_Clr	0: Do nothing	1: Clear CHd FIFO	
			W	2: CHcFIFO_Clr	0: Do nothing	1: Clear CHc FIFO	
			W	1: CHbFIFO_Clr	0: Do nothing	1: Clear CHb FIFO	
			W	0: CHaFIFO_Clr	0: Do nothing	1: Clear CHa FIFO	

This register clears the FIFO for the relevant channel. This is a write-only register.

When set to 1, each bit in this register only clears the FIFO and does not hold the value set in it.

When DMAx{x=0,1} are joined to any channel and the relevant DMA is active (when DMA_Running bit = 1), do not set the bit for that channel to 1.

7.6.35 10Ah H_ClrAllCHnJoin (Host Clear All CHn Join)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	10Ah	H_ClrAllCHnJoin	W	7:ClrJoinIDE	0: Do nothing	1: Clear Join IDE	XXh
			W	6:ClrJoinFIFO_Stat	0: Do nothing	1: Clear Join FIFO_Stat	
				5:	0:	1:	
				4:	0:	1:	
			W	3:ClrJoinDMA1	0: Do nothing	1: Clear Join DMA1	
			W	2:ClrJoinDMA0	0: Do nothing	1: Clear Join DMA0	
			W	1:ClrJoinCPU_Rd	0: Do nothing	1: Clear Join CPU_Rd	
			W	0:ClrJoinCPU_Wr	0: Do nothing	1: Clear Join CPU_Wr	

This register clears a connection between the port selected by a bit in it and a channel.

The bits in this register area automatically cleared to 0 after a connection is cleared.

Do not set any bit in this register to 1 while a channel is connected to a port (CHx{x=0,a-e}Join register's relevant bit = 1) and the port is active. It may cause erratic behavior.

7.6.36 10B~10Fh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	10Bh -10Fh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.6.37 110h H_CH0SETUP_0 (Host CH0 SETUP 0)

7.6.38 111h H_CH0SETUP_1 (Host CH0 SETUP 1)

7.6.39 112h H_CH0SETUP_2 (Host CH0 SETUP 2)

7.6.40 113h H_CH0SETUP_3 (Host CH0 SETUP 3)

7.6.41 114h H_CH0SETUP_4 (Host CH0 SETUP 4)

7.6.42 115h H_CH0SETUP_5 (Host CH0 SETUP 5)

7.6.43 116h H_CH0SETUP_6 (Host CH0 SETUP 6)

7.6.44 117h H_CH0SETUP_7 (Host CH0 SETUP 7)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	110h -117h	H_CH0SETUP_0 -H_CH0SETUP_7	R/W	7: CH0SETUP_n [7]	Channel 0 SETUP Data 0 - Channel 0 SETUP Data 7	00h
				6: CH0SETUP_n [6]		
				5: CH0SETUP _n [5]		
				4: CH0SETUP _n [4]		
				3: CH0SETUP _n [3]		
				2: CH0SETUP _n [2]		
				1: CH0SETUP _n [1]		
				0: CH0SETUP _n [0]		

The 8 bytes of data received in the setup stage of channel CH0 are stored in these registers sequentially beginning with CH0SETUP_0.

CH0SETUP_0

BmRequestType is set in this register.

CH0SETUP_1

BRequest is set in this register.

CH0SETUP_2

The 8 low-order bits of Wvalue are set in this register.

CH0SETUP_3

The 8 high-order bits of Wvalue are set in this register.

CH0SETUP_4

The 8 low-order bits of WIndex are set in this register.

CH0SETUP_5

The 8 high-order bits of WIndex are set in this register.

CH0SETUP_6

The 8 low-order bits of WLength are set in this register.

CH0SETUP_7

The 8 high-order bits of WLength are set in this register.

7.6.45 118h~11Dh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	118h -11Dh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.6.46 11Eh H_FrameNumber_H (Host FrameNumber High)

7.6.47 11Fh H_FrameNumber_L (Host FrameNumber Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	11Eh	H_FrameNumber_H		7:	0:	1:	07h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R/W	2: FrameNumber [10]	Frame Number High		
				1: FrameNumber [9]			
				0: FrameNumber [8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	11Fh	H_FrameNumber_L	R/W	7: FrameNumber [7]	Frame Number Low	FFh
				6: FrameNumber [6]		
				5: FrameNumber [5]		
				4: FrameNumber [4]		
				3: FrameNumber [3]		
				2: FrameNumber [2]		
				1: FrameNumber [1]		
				0: FrameNumber [0]		

These registers show the frame number of the USB that is updated for each SOF token transmitted. To get a frame number, the FrameNumber_H and FrameNumber_L registers must be accessed in pairs. At that time, be sure to access the FrameNumber_H register first.

Note: The Reset value for this register indicates the value read during ACT_HOST. In other states, the Reset value is read as 00h.

11Eh.Bit7-3 **Reserved**

11Eh.Bit2-0, 11Fh.Bit7-0 **FrameNumber [10:0]**

These bits show the FrameNumber of the transmitted SOF packet.

7.6.48 120h H_CH0Config_0(Host Channel 0 Configuration0)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	120h	H_CH0Config_0	R / W	7: ACK_Cnt[3]	Channel 0 ACK Count		00h
				6: ACK_Cnt[2]			
				5: ACK_Cnt[1]			
				4: ACK_Cnt[0]			
			R / W	3: SpeedMode[1]	Channel 0 Speed Mode		
				2: SpeedMode[0]			
			R / W	1: Toggle	0: Toggle0	1: Toggle1	
			R / W	0: TranGo	0: Stand by	1: Transaction Start	

This register is used to make basic settings of channel CH0 during USB host operation.

Bit7-4 ACK_Cnt [3:0]

These bits set the number of ACK count in a transfer performed on channel CH0.

When ACK count reaches the set value, the TranACK bit in the H_CH0IntStat register is set.

0000: ACK is counted up to 16 times.

0001–1111: ACK is counted anywhere between 1 to 15 times.

During the execution of the control transfer support function, only the transactions in the data stage are counted and transactions from setup stage and status stage are not counted.

Bits3-2 SpeedMode [1:0]

These bits set the operation mode of the device that performs a transfer on channel CH0.

00: HS mode — Set this for HS devices.

01: FS mode — Set this for FS devices.

10: Reserved — Use of this value is prohibited.

11: LS mode — Set this for FS devices.

Bit1 Toggle

This bit sets the initial value of the toggle sequence bit with which a transaction is to be started. While a transaction is under execution and after a transaction is complete, this bit indicates the status of the toggle sequence bit.

0: Toggle 0

1: Toggle 1

Bit0 **TranGo**

Setting this bit to 1 causes a transaction on channel CH0 to start. Transaction process can be stopped by clearing this bit to 0 after a transaction started. This bit also serves as a status indicating whether a transaction on channel CH0 is under execution or not.

0: Stops a transaction (transaction stopped)

1: Starts a transaction (transaction under execution)

When a transfer for the number of bytes set by the H_CH0TotalSize_H–L registers is complete, the TotalSizeCmp bit in the H-CH0IntStat register is set to 1, at which time this bit is automatically reset to 0. Furthermore, this bit is reset to 0 when the ChangeCondition bit in the H_CH0IntStat register is set. In that case, inspect the H_CH0ConditionCode register to find a reason for which the condition bit was set.

If a transaction is stopped by clearing this bit, the ChangeCondition bit in the H_CH0IntStat register is set when the transaction being processed is complete. Even when a transaction is stopped, the data present in the FIFO, the (remaining) total size, and channel-related settings are left intact. Therefore, a transaction can be resumed from where it stopped by setting this bit to 1 again. (To perform a new transaction, clear the FIFO and set channel information over again.)

This setting is not required when using the control transfer support function.

7.6.49 121h H_CH0Config_1(Host Channel 0 Configuration1)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	121h	H_CH0Config_1	R / W	7: TID[1]	Channel 0 Transaction ID		00h
				6: TD[0]			
				5:	0:		
				4:	0:		
				3:	0:		
				2:	0:		
				1:	0:		
				0:	0:		

This register is used to make basic settings of channel CH0 during USB host operation.

Bits7-6 TID[1:0]

These bits set the type of token (SETUP, OUT, or IN) that is issued on channel CH0. Settings of these bits have no effect when a transaction is started by setting the CTL_SupportGo bit in the CTL_SupportControl register to 1.

00: SETUP — Issues a SETUP token.

01: OUT — Issues an OUT token.

10: IN — Issues an IN token.

11: Reserved — Use of this value is prohibited.

This setting is not required when using the control transfer support function.

Bits5-0 Reserved

7.6.50 122h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	122h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.6.51 123h H_CH0MaxPktSize (Host Channel 0 Max Packet Size)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	123h	H_CH0MaxPktSize		7:	0:	1:	00h
			R / W	6: MaxPktSize[6]	Channel 0 Max Packet Size		
				5: MaxPktSize[5]			
				4: MaxPktSize[4]			
				3: MaxPktSize[3]			
				2: MaxPktSize[2]			
				1: MaxPktSize[1]			
				0: MaxPktSize[0]			

This register sets MaxPacketSize of channel CH0 during USB host operation.

Bit7 Reserved

Bits6-0 MaxPktSize[6:0]

These bits set the MaxPacketSize of channel CH0.

During LS 8 bytes

During FS 8, 16, 32, or 64 bytes

During HS 64 bytes

Set one of the above transfer sizes.

Setting any other transfer size is prohibited.

7.6.52 124h~125h Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	124h -125h	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.6.53 125h H_CH0FuncAdrs (Host Channel 0 Function Address)

7.6.54. 127h H_CH0TotalSize_L (Host Channel 0 Total Size Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	126h	H_CH0TotalSize_H	R / W	7: TotalSize[15]	Channel 0 Total Size High	00h
				6: TotalSize[14]		
				5: TotalSize[13]		
				4: TotalSize[12]		
				3: TotalSize[11]		
				2: TotalSize[10]		
				1: TotalSize[9]		
				0: TotalSize[8]		

This register sets the Total Size of the data to be transferred on channel CH0 during USB host operation.

118h.Bit7-0, 119h.Bit7-0 TotalSize[15:0]

These bits set a total number of data bytes to be transferred on channel CH0 (maximum 65,535 bytes: approx. 64 Kbytes).

Once a transaction is started by the TranGo bit in the H_CH0Config_0 register, the remaining number of bytes to be transferred can be read through this register.

When bits 15–8 (H_CH0TotalSize_H register) are read, the value of bits 7–0 (H_CH0TotalSize_L register) is fixed. (Even when the read value is fixed, the internal counter of the LSI continues counting.)

Therefore, to read the remaining number of transfer bytes through 8-bit register accesses, read H_CH0TotalSize_H and H_CH0TotalSize_L in that order.

Note also that if an OUT transaction is executed with TotalSize = 0, a zero-length packet is transmitted.

The setting of this register is not required when executing a SETUP transaction or using the control transfer support function.

7.6.55 128h H_CH0HubAdrs (Host Channel 0 Hub Address)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	128h	H_CH0HubAdrs	R / W	7: HubAdrs[3]	Channel 0 Hub Address		00h
				6: HubAdrs[2]			
				5: HubAdrs[1]			
				4: HubAdrs[0]			
				3:	0:	1:	
			R / W	2: Port[2]	Channel 0 Port Number		
				1: Port[1]			
				0: Port[0]			

This register sets the hub that is connected to channel CH0 during USB host operation.

Bits7-4 HubAdrs[3:0]

These bits set the USB address of the hub to which a function is connected that performs a transfer on channel CH0. Any value in the range of 0 to 15 can be set.

Bit3 Reserved

Bits2-0 Port[2:0]

These bits set the port number of the hub to which a function is connected that performs a transfer on channel CH0. Any value in the range of 0 to 7 can be set.

7.6.56 129h H_CH0FuncAdrs (Host Channel 0 Function Address)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	129h	H_CH0FuncAdrs	R / W	7: FuncAdrs[3]	Channel 0 Function Address	00h
				6: FuncAdrs[2]		
				5: FuncAdrs[1]		
				4: FuncAdrs[0]		
			R / W	3: EP_Number[3]	Channel 0 Endpoint Number	
				2: EP_Number[2]		
				1: EP_Number[1]		
				0: EP_Number[0]		

This register sets the address of a function that performs a transfer on channel CH0 during USB host operation.

Bits7-4 FuncAdrs[3:0]

These bits set the USB address of the function that includes the endpoint managed by channel CH0. Any value in the range of 0 to 15 can be set.

Bits3-0 EP_Number[3:0]

This bit sets the endpoint number that performs a transfer on channel CH0. Any value in the range of 0 to 15 can be set.

7.6.57 12Ah Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	12Ah	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.6.58 12Bh CTL_SupportControl (Host ControlTransfer Support Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	12Bh	H_CTL_Support-Control		7:	0:	1:	XXh
				6:	0:	1:	
			R	5: CTL_SupportState[1]	ControlTransfer Support State		
				4: CTL_SupportState[0]			
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
			R / W	0: CTL_SupportGo	0: Stand by	1: Control Transfer Go	

This register sets the support functions used for control transfers on channel CH0 during USB host operation.

Bits7-6 Reserved

Bits5-4 CTL_SupportState[1:0]

These bits indicate which stage is under execution when a transfer is performed using the control transfer support function after setting the CTL_SupportGo bit to 1.

00: Idle —	Indicates that transfer is not executed yet or a transfer has completed without errors.
01: Setup Stage —	Indicates that a setup stage is under execution.
10: Data Stage —	Indicates that a data stage is under execution.
11: Status Stage —	Indicates that a status stage is under execution.

Bits3-1 Reserved

Bit0 CTL_SupportGo

Setting this bit to 1 enables the control transfer support function so that a control transfer on channel CH0 is automatically performed, ranging from the setup stage to the status stage with or without a data stage included.

In the setup stage, a SETUP token is automatically sent out, and the requests set in H_CH0SETUP_0 through 7 are transmitted.

Next, if a data stage is involved, a transaction is automatically executed in a specified direction with a specified size.

Finally in the status stage, an appropriate PID token is issued depending on whether a data stage is involved and the direction of a data stage, and a zero-length packet is transmitted/received between the host and device.

When the above transition and stage sequence is completed without errors, the CTL_SupportCmp bit in the H_CH0IntStat register is set. If a packet error was detected during the sequence, the CTL_SupportStop bit in the H_CH0IntStat register is set, causing the transaction to stop. In that case, inspect the ConditionCode register to find the cause of the error.

7.6.59 12Ch~12Dh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	12Ch -12Dh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.6.60 12Eh H_CH0ConditionCode (Host Channel 0 Condition Code)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	11Eh	H_CH0ConditionCode		7:	0:	1:	00h
			R	6: ConditionCode[2]	Channel 0 Condition Code		
				5: ConditionCode[1]			
				4: ConditionCode[0]			
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register indicates the result of a transfer that was completed on channel CH0 during USB host operation.

Bit7 Reserved

Bits6-4 ConditionCode[2:0]

These bits indicate the result of a transfer that was completed on channel CH0.

Code	Meaning	Description
000	NoError	The transaction is completed without error.
001	Stall	The endpoint has returned stall PID.
010	DataOverrun	• The amount of data received exceeds the maximum allowed size of a data packet (MaxPacketSize). * In the event the CRC error and the bit stuffing error is detected simultaneously, the errors will be processed as retry errors. • The amount of data received exceeds the maximum allowed IRP (TotalSize) * In the event the CRC error and the bit stuffing error is detected simultaneously, the errors will be processed as retry errors. * If the size of the received data packet is less than the MaxPacketSize and the data toggle included in the data packet does not match the expected value, the error will be treated as Toggle Mismatch Error and not Data Overrun.
011	DataUnderrun	• The endpoint has returned bytes of data which are less than MaxPacketSize and are insufficient to fill the IRP (TotalSize). * In the event the CRC error and the bit stuffing error is detected simultaneously, the errors will be processed as retry errors.

Code	Meaning	Description
100	RETRYERROR	• The device does not respond to a token within a designated time (IN) or does not transmit a handshaking signal within a designated time (OUT). • The data packet from the endpoint contains a CRC error. • The data packet from the endpoint contains a bit stuffing error. • The PID inspection bit from the endpoint failed in data PID (IN) or handshaking (OUT). • The received PID is invalid or has no PID values defined. • The data toggle included in the data packet from the endpoint does not match the expected value. (toggle mismatch).
Other	Reserved	

Bits3-0 Reserved

7.6.61 12Fh H_CH0Join (Host Channel 0 Join)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	12Fh	H_CH0Join		7:	0:	1:	00h
			R / W	6: JoinFIFO_Stat	0: Do nothing	1 Join CH0 to show Status	
				5:	0:	1:	
				4:	0	1:	
			R / W	3: JoinDMA0	0: Do nothing	1: Join CH0 to DMA0	
			R / W	2: JoinDMA1	0: Do nothing	1: Join CH0 to DMA1	
			R / W	1: JoinCPU_Rd	0: Do nothing	1: Join CH0 to CPU_Rd	
			R / W	0: JoinCPU_Wr	0: Do nothing	1: Join CH0 to CPU_Wr	

This register specifies the port with which data transfers with channel CH0 are to be performed during USB host operation.

Bit7 Reserved

Bit6 JoinFIFO_Stat

This bit permits the Full, Empty or NotEmpty status of the FIFO of channel CH0 to be monitored with H_FIFO_IntStat.FIFO_Full, H_FIFO_IntStat.FIFO_Empty, and H_FIFO_IntStat.FIFO_NotEmpty.

Bits5-4 Reserved

Bit3 JoinDMA1

Performs a DMA1 transfer with the FIFO of channel CH0. The transfer direction is determined by the DMA1_Control.Dir bit.

Bit2 JoinDMA0

Performs a DMA0 transfer with the FIFO of channel CH0. The transfer direction is determined by the DMA0_Control.Dir bit.

Bit1 JoinCPU_Rd

Performs a read transfer of CPU register access with the FIFO of channel CH0. Namely, when a read of the FIFO_Rd_0,1 registers or FIFO_ByteRd register is performed, data is read from the FIFO of this channel.

Bit0 JoinCPU_Wr

Performs a write transfer of CPU register access with the FIFO of channel CH0. Namely, when a write to the FIFO_Wr_0,1 registers is performed, data is written to the FIFO of this channel.

If the JoinDMAx{x=0,1} bit is set, it is possible to refer to the remaining bytes of data in the FIFO when the DMA0_Control.Dir bit = 1 or the amount of free space in the FIFO when the DMA0_Control.Dir bit = 0 by inspecting the DMAx{x=0,1}_Remain_H,L registers.

If the JoinCPU_Rd or JoinCPU_Wr bit is set, it is possible to write or read data to or from the FIFO_Rd_0,1, FIFO_ByteRd, or FIFO_Wr_0,1 registers after inspecting the FIFO_RdRemain_H,L or FIFO_WrRemain_H,L registers.

The JoinDMA1, JoinDMA0, JoinCPU_Rd, and JoinCPU_Wr bits can be set to 1 only one bit at a time. If an attempt is made to set two or more of these bits by writing 1 at the same time, only the higher bit is set.

7.6.62 130h H_CHaConfig_0(Host Channel a Configuration0)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	130h	H_CHaConfig_0	R / W	7: ACK_Cnt[3]	Channel [a] ACK Count		10h
				6: ACK_Cnt[2]			
				5: ACK_Cnt[1]			
				4: ACK_Cnt[0]			
			R / W	3: SpeedMode[1]	Channel [a] Speed Mode		
				2: SpeedMode[0]			
			R / W	1: Toggle	0: Toggle0	1: Toggle1	
			R / W	0: TranGo	0: Stand by	1: Transaction Start	

This register is used to make basic settings of channel CHa during USB host operation.

Bits7-4 ACK_Cnt [3:0]

These bits set the number of ACK count in a transfer performed on channel CHa.

When ACK count reaches the set value, the TranACK bit in the H_CHaIntStat register is set.

0000: Reserved — Use of this value is prohibited.

0001–1111: ACK is counted anywhere between 1 to 15 times.

During the execution of the Bulk Only Transfer Support function, only the data transport transactions are counted and CBW/CSW transactions are not counted.

Bits3-2 SpeedMode [1:0]

These bits set the operation mode of the device that performs a transfer on channel CHa.

00: HS mode — Set this for HS devices.

01: FS mode — Set this for FS devices.

10–11: Reserved — Use of this value is prohibited.

Bit1 Toggle

This bit sets the initial value of the toggle sequence bit with which a transaction is to be started. While a transaction is under execution and after a transaction is complete, this bit indicates the status of the toggle sequence bit.

0: Toggle 0

1: Toggle 1

The setting of this bit is not required when using the Bulk Only Support function.

Bit0 TranGo

Setting this bit to 1 causes a transaction on channel CHa to start. Transaction processing can be stopped by clearing this bit to 0 after a transaction started. This bit also serves as a status indicating whether a transaction on channel CHa is under execution or not.

0: Stops a transaction (transaction stopped)

1: Starts a transaction (transaction under execution)

When a transfer for the number of bytes set by the H_CHaTotalSize_HH–LL registers is complete, the TranCmp bit in the H-CHaIntStat register is set to 1, at which time this bit is automatically reset to 0. Furthermore, this bit is reset to 0 when the ChangeCondition bit in the H_CHaIntStat register is set. In that case, inspect the H_CHaConditionCode register to find a reason for which the condition bit was set.

If a transaction is stopped by clearing this bit, the ChangeCondition bit in the H_CHaIntStat register is set when the transaction being processed is complete. Even when a transaction is stopped, the data present in the FIFO, the (remaining) total size, and channel-related settings are left intact. Therefore, a transaction can be resumed from where it stopped by setting this bit to 1 again. (To perform a new transaction, clear the FIFO and set channel information over again.)

The setting of this bit is not required when using the Bulk Only Support function.

7.6.63 131h H_CHaConfig_1(Host Channel a Configuration1)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	131h	H_CHaConfig_1	R / W	7: PID[1]	Channel a PID		00h
				6: PID[0]			
				5:	0:	1:	
				4:	0:	1:	
			R / W	3: AutoZerolen	0: Do nothing	1: Add Zerolen	
				2:	0:	1:	
				1:	0:	1:	
			R / W	0: TotalSizeFree	0: Do nothing	1: Total Size Free	

This register is used to make basic settings of channel CHa during USB host operation.

Bits7-6 PID[1:0]

These bits set the type of token (OUT or IN) that is issued on channel CHa. Settings of these bits have no effect when a transaction is started by setting the CTL_SupportGo bit in the CTL_SupportControl register to 1.

00: Reserved —	Use of this value is prohibited.
01: OUT —	Issues an OUT token.
10: IN —	Issues an IN token.
11: Reserved —	Use of this value is prohibited.

The setting of this bit is not required when using the Bulk Only support function

Bits5-4 Reserved

Bit3 AutoZerolen

Setting this bit to 1 enables the auto zero-length function, so that when a transfer for the data bytes set by H_CHaTotalSizeHH–LL registers is completed exactly with MaxPacketSize, a zero-length packet is automatically attached at the end of the transfer. This bit is effective for only OUT transfers.

Bits2-1 Reserved

Bit0 TotalSizeFree

Setting this bit to 1 makes the transfer size infinite regardless of the value set in the H_CHaTotalSizeHH–LL registers.

7.6.64 132h H_CHaMaxPktSize_H (Host Channel a Max Packet Size High)

7.6.65 133h H_CHaMaxPktSize_L (Host Channel a Max Packet Size Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	132h	H_CHaMaxPktSize_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R / W	1: MaxPktSize[9]	Channel a Max Packet Size High		
				0: MaxPktSize[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	133h	H_CHaMaxPktSize_L	R / W	7: MaxPktSize[7]	Channel a Max Packet Size Low	00h
				6: MaxPktSize[6]		
				5: MaxPktSize[5]		
				4: MaxPktSize[4]		
				3: MaxPktSize[3]		
				2: MaxPktSize[2]		
				1: MaxPktSize[1]		
				0: MaxPktSize[0]		

This register sets MaxPacketSize of channel CHa during USB host operation.

132h.Bit7-2 Reserved

Do not write 1 to the reserved bits.

132h.Bit1-0, 133h.Bit7-0 MaxPktSize[9:0]

These bits set the MaxPacketSize of channel CHa.

During FS 8, 16, 32, or 64 bytes

During HS 512 bytes

Set one of the above transfer sizes.

Setting any other transfer size is prohibited.

7.6.66 134h H_CHaHubAdrs (Host Channel a Hub Address)

7.6.67. 135h H_CHaTotalSize_HL (Host Channel a Total Size High-Low)

7.6.68. 136h H_CHaTotalSize_LH (Host Channel a Total Size Low-High)

7.6.69. 137h H_CHaTotalSize_LL (Host Channel a Total Size Low-Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	134h	H_CHaTotalSize_HH	R / W	7: TotalSize[31]	Channel a Total Size High-High	00h
				6: TotalSize[30]		
				5: TotalSize[29]		
				4: TotalSize[28]		
				3: TotalSize[27]		
				2: TotalSize[26]		
				1: TotalSize[25]		
				0: TotalSize[24]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	135h	H_CHaTotalSize_HL	R / W	7: TotalSize[23]	Channel a Total Size High-Low	00h
				6: TotalSize[22]		
				5: TotalSize[21]		
				4: TotalSize[20]		
				3: TotalSize[19]		
				2: TotalSize[18]		
				1: TotalSize[17]		
				0: TotalSize[16]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	136h	H_CHaTotalSize_LH	R / W	7: TotalSize[15]	Channel a Total Size Low-High	00h
				6: TotalSize[14]		
				5: TotalSize[13]		
				4: TotalSize[12]		
				3: TotalSize[11]		
				2: TotalSize[10]		
				1: TotalSize[9]		
				0: TotalSize[8]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	137h	H_CHaTotalSize_LL	R / W	7: TotalSize[7]	Channel a Total Size Low-Low	00h
				6: TotalSize[6]		
				5: TotalSize[5]		
				4: TotalSize[4]		
				3: TotalSize[3]		
				2: TotalSize[2]		
				1: TotalSize[1]		
				0: TotalSize[0]		

These registers set the Total Size of the data to be transferred on channel CHa during USB host operation.

134h.Bit7-0, 135h.Bit7-0, 136h.Bit7-0, 137h.Bit7-0 TotalSize[31:0]

These bits set the total number of data bytes to be transferred on channel CHa (maximum 4,294,967,295 bytes: approx. 4 Gbytes).

Once a transaction is started by the TranGo bit in the H_CHaConfig_0 register, the remaining number of bytes to be transferred can be read through this register.

When bits 31–24 (H_CHaTotalSize_HH register) are read, the values of bits 23–16 (H_CHaTotalSize_HL register), bits 15–8 (H_CHaTotalSize_LH register), and bits 7–0 (H_CHaTotalSize_LL register) all are fixed. (Even when the read value is fixed, the internal counter of the LSI continues counting.)

Therefore, to read the remaining number of transfer bytes through 8-bit register accesses, read H_CHaTotalSize_HH, H_CHaTotalSize_HL, H_CHaTotalSize_LH, and H_CHaTotalSize_LL in that order.

Note also that if an OUT transaction is executed with TotalSize = 0, a zero-length packet is transmitted.

The setting of this register is not required when using the Bulk Only Support function.

7.6.70. 138h H_CHaHubAdrs (Host Channel a Hub Address)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	138h	H_CHaHubAdrs	R / W	7: HubAdrs[3]	Channel a Hub Address		00h
				6: HubAdrs[2]			
				5: HubAdrs[1]			
				4: HubAdrs[0]			
				3:	0:	1:	
			R / W	2: Port[2]	Channel a Port Number		
				1: Port[1]			
				0: Port[0]			

This register sets the hub that is connected to channel CHa during USB host operation.

Bits7-4 HubAdrs[3:0]

These bits set sthe USB address of the hub to which a function is connected that performs a transfer on channel CHa.

Any value in the range of 0 to 15 can be set.

Bit3 Reserved

Bits2-0 Port[2:0]

These bits set the port number of the hub to which a function is connected that performs a transfer on channel CHa.

Any value in the range of 0 to 7 can be set.

7.6.71 139h H_CHaFuncAdrs (Host Channel a Function Address)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	139h	H_CHaFuncAdrs	R / W	7: FuncAdrs[3]	Channel a Function Address	00h
				6: FuncAdrs[2]		
				5: FuncAdrs[1]		
				4: FuncAdrs[0]		
			R / W	3: EP_Number[3]	Channel a Endpoint Number	
				2: EP_Number[2]		
				1: EP_Number[1]		
				0: EP_Number[0]		

This register sets the address of a function that performs a transfer on channel CHa during USB host operation.

Bits7-4 FuncAdrs[3:0]

These bits set the USB address of the function that includes the endpoint managed by channel CHa. Any value in the range of 0 to 15 can be set.

Bits3-0 EP_Number[3:0]

These bits set the endpoint number that performs a transfer on channel CHa. Any value in the range of 0 to 15 can be set.

The setting of this bit is not required when using the Bulk Only support function

7.6.72 13Ah H_BO_SupporotControl (Host Bulk Only Transfer Supporot Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	13Ah	H_BO_Supporot-Control		7:	0:	1:	00h
				6:	0:	1:	
			R / W	5: BO_TransportState[1]	Bulk Only Transfer Transport State		
				4: BO_TransportState[0]			
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
			R / W	0: BO_SupportGo	0: Stand by	1: BO Transfer Go	

This register sets the Bulk Only transfer support functions on channel CHa during USB host operation.

Bits7-6 Reserved

Bits5-4 BO_TransportState[1:0]

These bits indicate which transport is under execution when a transfer is performed using the Bulk Only transfer support function after setting the BO_SupportGo bit to 1.

00: Idle —	Indicates that transfer is not executed yet or a transfer is completed without errors.
01: CBW Transport —	Indicates that a CBW transport is under execution.
10: Data Transport —	Indicates that a data transport is under execution.
11: CSW Transport —	Indicates that a CSW transport is under execution.

Bits3-1 Reserved

Bit0 BO_SupportGo

Setting this bit to 1 enables the Bulk Only transfer support function so that a Bulk Only transfer on channel CHa is automatically performed, ranging from the CBW transport to CSW transport with or without a data transport included.

In the CBW transport, an OUT token is automatically sent out, and the data set in the CBW area of the FIFO is transmitted.

Next, if a data transport is involved, a data transport is automatically executed in a specified direction with a specified size.

Finally in the CSW transport, an IN token is automatically sent out, and data is received in the CSW area of the FIFO.

When the above transports are completed without errors, the BO_SupportCmp bit in the H_BO_SupportIntStat register is set. If a packet error is detected during the transport or the CSW value is found inappropriate, the BO_SupportStop bit in the H_CHaIntStat register is set, causing the transaction to stop. In that case, inspect the H_CHaConditionCode register to find the cause of the error. If the value of ConditionCode = 000 when the BO_SupportStop bit is set to 1, it means that the CSW value is inappropriate. Inspect the stopped transport by shifting to the BO_TranportState.

7.6.73 13Bh H_CSW_RcvDataSize(Host CSW Receive Data Size)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	13Bh	H_CSW_RcvDataSize		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
			R	3: CSW_RcvDataSize[3]	CSW Resceive Data Size		
				2: CSW_RcvDataSize[2]			
				1: CSW_RcvDataSize[1]			
				0: CSW_RcvDataSize[0]			

This register indicates the number of data bytes received during execution of the CSW transport function when using the Bulk Only transfer support function on channel CHa during USB host operation.

Bits7-4 Reserved

Bits3-0 CSW_RcvDataSize[3:0]

These bits indicate the number of transmitted data bytes of CSW.

If less than 13 bytes of data is received in a CSW transport, this register will show the number of trasmitted data bytes.

If a hanshake is received in a CSW transport, the values on this register will have no significance in cases other than CSW transport.

7.6.74 13Ch H_OUT_EP_Control(Host OUT Endpoint Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	13Ch	H_OUT_EP_Control		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: OUT_Toggle	0: Toggle0	1: Toggle1	
			R / W	3: OUT_EP_Number[3]	OUT EP Number		
				2: OUT_EP_Number[2]			
				1: OUT_EP_Number[1]			
				0: OUT_EP_Number[0]			

This register sets the Bulk Only transfer support functions used on channel CHa during USB host operation.

Bits7-5 Reserved

Bit4 OUT_Toggle

This bit sets the initial value of the toggle sequence bit for an OUT direction transfer (CBW transport or Data OUT transport) to be performed using the Bulk Only transfer support function after setting the BO_SupportGo bit in the H_CBW_Control register to 1.

0: Toggle 0

1: Toggle 1

In addition, when an OUT-direction transport is completed without errors, this bit will retain the toggle sequence bit automatically.

Bits3-0 OUT_EP_Number[3:0]

These bits set the endpoint number of the destination device for an OUT direction transfer (CBW transport or Data OUT transport) to be performed using the Bulk Only transfer support function after setting the BO_SupportGo bit in the H_CBW_Control register to 1.

Any number in the range of 0 to 15 can be set.

7.6.75 13Dh H_IN_EP_Control(Host IN Endpoint Control)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	13Dh	H_IN_EP_Control		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: IN_Toggle	0: Toggle0	1: Toggle1	
			R / W	3: IN_EP_Number[3]	IN EP Number		
				2: IN_EP_Number[2]			
				1: IN_EP_Number[1]			
				0: IN_EP_Number[0]			

This register sets the Bulk Only transfer support functions used on channel CHa during USB host operation.

Bits7-5 Reserved

Bit4 IN_Toggle

This bit sets the initial value of the toggle sequence bit for an IN direction transfer (CBW transport or Data IN transport) to be performed using the Bulk Only transfer support function after setting the BO_SupportGo bit in the H_CBW_Control register to 1.

0: Toggle 0

1: Toggle 1

In addition, when an IN-direction transport is completed without errors, this bit will retain the toggle sequence bit automatically.

Bits3-0 IN_EP_Number[3:0]

These bits set the endpoint number of the destination device for an IN direction transfer (CBW transport or Data IN transport) to be performed using the Bulk Only transfer support function after setting the BO_SupportGo bit in the H_CBW_Control register to 1.

Any number in the range of 0 to 15 can be set.

7.6.76 13Eh H_CHaConditionCode (Channel a Condition Code)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	13Eh	H_CHaConditionCode		7:	0:	1:	00h
			R	6: ConditionCode[2]	Channel a Condition Code		
			R	5: ConditionCode[1]	Channel a Condition Code		
			R	4: ConditionCode[0]	Channel a Condition Code		
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register indicates the result of a transfer that was completed on channel CHa during USB host operation.

Bit7 Reserved

Bits6-4 ConditionCode[2:0]

These bits indicate the result of a transfer that was completed on channel CHa.

Code	Meaning	Description
000	NoError	The transaction is completed without error.
001	Stall	The endpoint has returned stall PID.
010	DataOverrun	• The amount of data received exceeds the maximum allowed size of a data packet (MaxPacketSize). * In the event the CRC error and the bit stuffing error is detected simultaneously, the errors will be processed as retry errors. • The amount of data received exceeds the maximum allowed IRP (TotalSize) * In the event the CRC error and the bit stuffing error is detected simultaneously, the errors will be processed as retry errors. * If the size of the received data packet is less than the MaxPacketSize and the data toggle included in the data packet does not match the expected value, the error will be treated as Toggle Mismatch Error and not Data Overrun.
011	DataUnderrun	• The endpoint has returned bytes of data which are less than MaxPacketSize and are insufficient to fill the IRP (TotalSize). * In the event the CRC error and the bit stuffing error is detected simultaneously, the errors will be processed as retry errors.

Code	Meaning	Description
100	RetryError	• The device does not respond to a token within a designated time (IN) or does not transmit a handshaking signal within a designated time (OUT). • The data packet from the endpoint contains a CRC error. • The data packet from the endpoint contains a bit stuffing error. • The PID inspection bit from the endpoint failed in data PID (IN) or handshaking (OUT). • The received PID is invalid or has no PID values defined. • The data toggle included in the data packet from the endpoint does not match the expected value (toggle mismatch).
Other	Reserved	

Bits3-0 Reserved

7.6.77 13Fh H_CHaJoin (Host Channel a Join)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	13Fh	H_CHaJoin	R / W	7: JoinIDE	0: Do nothing	1: Join CHa to IDE	00h
			R / W	6: JoinFIFO_Stat	0: Do nothing	1 Join CHa to show Status	
				5:	0:	1:	
				4:	0	1:	
			R / W	3: JoinDMA1	0: Do nothing	1: Join CHa to DMA1	
			R / W	2: JoinDMA0	0: Do nothing	1: Join CHa to DMA0	
			R / W	1: JoinCPU_Rd	0: Do nothing	1: Join CHa to CPU_Rd	
			R / W	0: JoinCPU_Wr	0: Do nothing	1: Join CHa to CPU_Wr	

This register specifies the port with which data transfers with channel CHa are to be performed during USB host operation.

Bit7 JoinIDE

Performs an IDE data transfer with the FIFO of channel CHa. The transfer direction is determined by the IDE_Control.Dir bit.

Bit6 JoinFIFO_Stat

This bit permits the Full, Empty, or NotEmpty status of the FIFO of channel CHa to be monitored with H_FIFO_IntStat.FIFO_Full, H_FIFO_IntStat.FIFO_Empty, and H_FIFO_IntStatFIFO_NotEmpty.

Bits5-4 Reserved

Bit3 JoinDMA1

Performs a DMA1 transfer with the FIFO of channel CHa. The transfer direction is determined by the DMA1_Control.Dir bit.

Bit2 JoinDMA0

Performs a DMA0 transfer with the FIFO of channel CHa. The transfer direction is determined by the DMA0_Control.Dir bit.

Bit1 JoinCPU_Rd

Performs a read transfer of CPU register access with the FIFO of channel CHa. Namely, when a read of the FIFO_Rd_0,1 registers or FIFO_ByteRd register is performed, data is read from the FIFO of this channel.

Bit0 JoinCPU_Wr

Performs a write transfer of CPU register access with the FIFO of channel CHa. Namely, when a write to the FIFO_Wr_0,1 registers is performed, data is written to the FIFO of this channel.

If the JoinDMAx{x=0,1} bit is set, it is possible to refer to the remaining bytes of data in the FIFO when the DMA0_Control.Dir bit = 1 or the amount of free space in the FIFO when the DMA0_Control.Dir bit = 0 by inspecting the DMAx{x=0,1}_Remain_H,L registers.

If the JoinCPU_Rd or JoinCPU_Wr bit is set, it is possible to write or read data to or from the FIFO_Rd_0,1, FIFO_ByteRd, or FIFO_Wr_0,1 registers after inspecting the FIFO_RdRemain_H,L or FIFO_WrRemain_H,L registers.

The JoinDMA1, JoinDMA0, JoinCPU_Rd, and JoinCPU_Wr bits can be set to 1 only one bit at a time. If an attempt is made to set two or more of these bits by writing 1 at the same time, only the higher bit is set.

7.6.78 140h H_CHbConfig_0(Host Channel b Configuration0)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	140h	CHbConfig_0	R / W	7: ACK_Cnt[3]	Channel b ACK Count		00h
				6: ACK_Cnt[2]			
				5: ACK_Cnt[1]			
				4: ACK_Cnt[1]			
			R / W	3: SpeedMode[1]	Channel b Speed Mode		
				2: SpeedMode[0]			
			R / W	1: Toggle	0: Toggle0	1: Toggle1	
			R / W	0: TranGo	0: Stand by	1: Transaction Start	

This register is used to make basic settings of channel CHb during USB host operation.

Bits7-4 ACK_Cnt [3:0]

These bits set a number of ACK counts in a transfer performed on channel CHb.

When ACK count reaches the set value, the TranACK bit in the H_CHbIntStat register is set.

0000: ACK is counted up to 16 times.

0001–1111: ACK is counted anywhere between 1 to 15 times.

Bits3-2 SpeedMode [1:0]

These bits set the operation mode of the device that performs a transfer on channel CHb.

00: HS mode — Set this for HS devices.

01: FS mode — Set this for FS devices.

10: Reserved — Use of this value is prohibited.

11: LS mode — Set this for LS devices.

Bit1 Toggle

This bit sets the initial value of the toggle sequence bit with which a transaction is to be started. While a transaction is under execution and after a transaction is complete, this bit indicates the status of the toggle sequence bit.

0: Toggle 0

1: Toggle 1

Bit0 TranGo

Setting this bit to 1 causes a transaction on channel CHb to start. Transaction processing can be stopped by clearing this bit to 0 after a transaction started. This bit also serves as a status indicating whether a transaction on channel CHb is under execution or not.

0: Stops a transaction (transaction stopped)

1: Starts a transaction (transaction under execution)

When a transfer for the number of bytes set by the H_CHbTotalSize_HH–LL registers is complete, the TranCmp bit in the H-CHbIntStat register is set to 1, at which time this bit is automatically reset to 0. Furthermore, this bit is reset to 0 when the ChangeCondition bit in the H_CHbIntStat register is set. In that case, inspect the H_CHbConditionCode register to find a reason for which the condition bit was set.

If a transaction is stopped by clearing this bit, the ChangeCondition bit in the H_CHbIntStat register is set when the transaction being processed is complete. Even when a transaction is stopped, the data present in the FIFO, the (remaining) total size, and channel-related settings are left intact. Therefore, a transaction can be resumed from where it stopped by setting this bit to 1 again. (To perform a new transaction, clear the FIFO and set channel information over again.)

7.6.79 141h H_CHbConfig_1(Host Channel b Configuration1)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	141h	H_CHbConfig_1	R / W	7: TID[1]	Channel b Transaction ID		00h
				6: TID[0]			
			R / W	5: TranType[1]	Channel b Transfer Type		
				4: TranType[0]			
			R / W	3: AutoZerolen	0: Do nothing	1: Add Zerolen	
				2:	0:	1:	
				1:	0:	1:	
			R / W	0: TotalSizeFree	0: Do nothing	1: Total Size Free	

This register is used to make basic settings of channel CHb during USB host operation.

Bits7-6 TID[1:0]

These bits set the type of token (OUT or IN) that is issued on channel CHb.

00: Reserved — Use of this value is prohibited.

01: OUT — Issues an OUT token.

10: IN — Issues an IN token.

11: Reserved — Use of this value is prohibited.

Bits5-4 TranType[1:0]

These bits set the type of transfer that is performed on channel CHb.

00: Reserved — Use of this value is prohibited.

01: Reserved — Use of this value is prohibited.

10: Bulk — Performs a bulk transfer.

11: Interrupt — Performs an interrupt transfer.

Bit3 AutoZerolen

Setting this bit to 1 enables the auto zero-length function, so that when a transfer for the data bytes set by H_CHbTotalSizeHH–LL registers is completed exactly with MaxPacketSize, a zero-length packet is automatically attached at the end of the transfer. This bit is effective for only OUT transfers.

Bits2-1 Reserved

Bit0 TotalSizeFree

Setting this bit to 1 makes the transfer size infinite regardless of the value set in the H_CHbTotalSizeHH–LL registers.

7.6.80 142h H_CHbMaxPktSize_H (Host Channel b Max Packet Size High)

7.6.81 143h H_CHbMaxPktSize_L (Host Channel b Max Packet Size Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	142h	H_CHbMaxPktSize_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R / W	1: MaxPktSize[9]	Channel b Max Packet Size High		
				0: MaxPktSize[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	143h	H_CHbMaxPktSize_L	R / W	7: MaxPktSize[7]	Channel b Max Packet Size Low	00h
				6: MaxPktSize[6]		
				5: MaxPktSize[5]		
				4: MaxPktSize[4]		
				3: MaxPktSize[3]		
				2: MaxPktSize[2]		
				1: MaxPktSize[1]		
				0: MaxPktSize[0]		

This register sets MaxPacketSize of channel CHb during USB host operation.

142h.Bit7-2 Reserved

Do not write 1 to the reserved bits.

142h.Bit1-0, 143h.Bits7-0 MaxPktSize[9:0]

These bits set the MaxPacketSize of channel CHb.

When using this channel for bulk transfers, set one of the following transfer sizes.

During FS 8, 16, 32, or 64 bytes

During HS 512 bytes

When using this channel for interrupt transfers, set any desired transfer size within the limits given below.

During LS Up to 8 bytes

During FS Up to 64 bytes

During HS Up to 512 bytes

Setting any other transfer size is prohibited.

7.6.82 144h H_CHbHubAdrs (Host Channel b Hub Address)

7.6.83. 145h H_CHbTotalSize_HL (Host Channel b Total Size High-Low)

7.6.84. 146h H_CHbTotalSize_LH (Host Channel b Total Size Low-High)

7.6.85. 147h H_CHbTotalSize_LL (Host Channel b Total Size Low-Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	144h	H_CHbTotalSize_HH	R / W	7: TotalSize[31]	Channel b Total Size High-High	00h
				6: TotalSize[30]		
				5: TotalSize[29]		
				4: TotalSize[28]		
				3: TotalSize[27]		
				2: TotalSize[26]		
				1: TotalSize[25]		
				0: TotalSize[24]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	145h	H_CHbTotalSize_HL	R / W	7: TotalSize[23]	Channel b Total Size High-Low	00h
				6: TotalSize[22]		
				5: TotalSize[21]		
				4: TotalSize[20]		
				3: TotalSize[19]		
				2: TotalSize[18]		
				1: TotalSize[17]		
				0: TotalSize[16]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	146h	H_CHbTotalSize_LH	R / W	7: TotalSize[15]	Channel b Total Size Low-High	00h
				6: TotalSize[14]		
				5: TotalSize[13]		
				4: TotalSize[12]		
				3: TotalSize[11]		
				2: TotalSize[10]		
				1: TotalSize[9]		
				0: TotalSize[8]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	147h	H_CHbTotalSize_LL	R / W	7: TotalSize[7]	Channel b Total Size Low-Low	00h
				6: TotalSize[6]		
				5: TotalSize[5]		
				4: TotalSize[4]		
				3: TotalSize[3]		
				2: TotalSize[2]		
				1: TotalSize[1]		
				0: TotalSize[0]		

These registers set the Total Size of the data to be transferred on channel CHb during USB host operation.

144h.Bit7-0, 145h.Bit7-0, 146h.Bit7-0, 147h.Bit7-0 TotalSize[31:0]

These bits set a total number of data bytes to be transferred on channel CHb (maximum 4,294,967,295 bytes: approx. 4 Gbytes).

Once a transaction is started by the TranGo bit in the H_CHbConfig_0 register, the remaining number of bytes to be transferred can be read through this register.

When bits 31–24 (H_CHbTotalSize_HH register) are read, the values of bits 23–16 (H_CHbTotalSize_HL register), bits 15–8 (H_CHbTotalSize_LH register), and bits 7–0 (H_CHbTotalSize_LL register) all are fixed. (Even when the read value is fixed, the internal counter of the LSI continues counting.)

Therefore, to read the remaining number of transfer bytes through 8-bit register accesses, read H_CHaTotalSize_HH, H_CHaTotalSize_HL, H_CHaTotalSize_LH, and H_CHaTotalSize_LL in that order.

Note also that if an OUT transaction is executed with TotalSize = 0, a zero-length packet is transmitted.

7.6.86. 148h H_CHbHubAdrs (Host Channel b Hub Address)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	148h	H_CHbHubAdrs	R / W	7: HubAdrs[3]	Channel b Hub Address		00h
				6: HubAdrs[2]			
				5: HubAdrs[1]			
				4: HubAdrs[0]			
				3:	0:	1:	
			R / W	2: Port[2]	Channel b Port Number		
				1: Port[1]			
				0: Port[0]			

This register sets the hub that is connected to channel CHb during USB host operation.

Bits7-4 HubAdrs[3:0]

These bits set the USB address of the hub to which a function is connected that performs a transfer on channel CHb.

Any value in the range of 0 to 15 can be set.

Bit3 Reserved

Bits2-0 Port[2:0]

These bits set the port number of the hub to which a function is connected that performs a transfer on channel CHb.

Any value in the range of 0 to 7 can be set.

7.6.87 149h H_CHbFuncAdrs (Host Channel b Function Address)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	149h	H_CHbFuncAdrs	R / W	7: FuncAdrs[3]	Channel b Function Address	00h
				6: FuncAdrs[2]		
				5: FuncAdrs[1]		
				4: FuncAdrs[0]		
			R / W	3: EP_Number[3]	Channel b Endpoint Number	
				2: EP_Number[2]		
				1: EP_Number[1]		
				0: EP_Number[0]		

This register sets the address of a function that performs a transfer on channel CHb during USB host operation.

Bits7-4 FuncAdrs[3:0]

These bits set the USB address of the function that includes the endpoint managed by channel CHb.

Any value in the range of 0 to 15 can be set.

Bits3-0 EP_Number[3:0]

These bits set the endpoint number that performs a transfer on channel CHb.

Any value in the range of 0 to 15 can be set.

7.6.88 14Ah CHbInterval_H(Channel b Interval High)

7.6.89 14Bh CHbInterval_L(Channel b Interval Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	14Ah	H_CHbInterval_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R / W	2: Interval[10]	Channel b Interrupt Transfer Interval High		
				1: Interval[9]			
				0: Interval[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	14Bh	H_CHdInterval_L	R / W	7: Interval[7]	Channel b Interrupt Transfer Interval Low	00h
				6: Interval[6]		
				5: Interval[5]		
				4: Interval[4]		
				3: Interval[3]		
				2: Interval[2]		
				1: Interval[1]		
				0: Interval[0]		

This register sets an interval value for the interrupt transfers to be performed on channel CHb during USB host operation.

14Ah.Bit7-3 **Reserved**

14Ah.Bit2-0, 14Bh.Bit7-0 **Interval[10:0]**

These bits set the interval (period) at which intervals tokens for interrupt transfers will be issued. The 3 low-order bits specify the interval time in microframes (125 μs), and the 7 high-order bits specify the interval time in frames (ms). Settings of this register are effective only when the H_CHbConfig1 register's TranType bits = 11 (interrupt transfer). Setting the value "0d" in this register has no effect.

The interval time set in this register is also used when transactions are retransmitted.

Interval[2:0] μFrame —	Specifies an interval time in 125 μs. Set this interval time to 1, 2, or 4 microframes. Setting any other value is prohibited. When setting these bits, make sure Interval[10:3] are all set to 0s.
Interval[10:3] Frame —	Specifies an interval time in ms. This interval time can be set to any value within the range 1 to 255 frames. When setting these bits, make sure Interval[2:0] are all set to 0s.

7.6.90 14Ch~14Dh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	14Ch -14Dh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.6.91 14Eh H_CHbConditionCode (Host Channel b Condition Code)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	14Eh	H_CHbConditionCode		7:	0:	1:	00h
			R	6: ConditionCode[2]	Channel b Condition Code		
				5: ConditionCode[1]			
				4: ConditionCode[0]			
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register indicates the result of a transfer that was completed on channel CHb during USB host operation.

Bit7 Reserved

Bits6-4 ConditionCode[2:0]

These bits indicate the result of a transfer that was completed on channel CHb.

Code	Meaning	Description
000	NoError	The transaction is completed without error.
001	Stall	The endpoint has returned stall PID.
010	DataOverrun	• The amount of data received exceeds the maximum allowed size of a data packet (MaxPacketSize). * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error. • The amount of data received exceeds the maximum allowed IRP (TotalSize) * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error. * If the received data packet is equal to or less than MaxPacketSize and the data toggle included in it does not match the expected value, this is processed as a toggle mismatch, and not as a data overrun.
011	DataUnderrun	• The endpoint has returned bytes of data which are less than MaxPacketSize and are insufficient to fill the IRP (TotalSize). * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error.

Code	Meaning	Description
100	RETRYERROR	• The device does not respond to a token within a designated time (IN) or does not transmit a handshaking signal within a designated time (OUT). • The data packet from the endpoint contains a CRC error. • The data packet from the endpoint contains a bit stuffing error. • The PID inspection bit from the endpoint failed in data PID (IN) or handshaking (OUT). • The received PID is invalid or has no PID values defined. • An ERR handshaking signal was received in a split transaction of interrupt transfer. • Three NYET handshaking signals were received consecutively in a split transaction of interrupt transfer. • The data toggle included in the data packet from the endpoint does not match the expected value (toggle mismatch).
Other	Reserved	

Bits3-0 Reserved

7.6.92 14Fh H_CHbJoin (Host Channel b Join)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	14Fh	H_CHbJoin	R / W	7: JoinIDE	0: Do nothing	1: Join CHb to IDE	00h
			R / W	6: JoinFIFO_Stat	0: Do nothing	1 Join CHb to show Status	
				5:	0:	1:	
				4:	0	1:	
			R / W	3: JoinDMA1	0: Do nothing	1: Join CHb to DMA1	
			R / W	2: JoinDMA0	0: Do nothing	1: Join CHb to DMA0	
			R / W	1: JoinCPU_Rd	0: Do nothing	1: Join CHb to CPU_Rd	
			R / W	0: JoinCPU_Wr	0: Do nothing	1: Join CHb to CPU_Wr	

This register specifies the port with which data transfers with channel CHb are to be performed during USB host operation.

Bit7 JoinIDE

Performs an IDE data transfer with the FIFO of channel CHb. The transfer direction is determined by the IDE_Control.Dir bit.

Bit6 JoinFIFO_Stat

This bit permits the Full, Empty or NotEmpty status of the FIFO of channel CHb to be monitored with H_FIFO_IntStat.FIFO_Full, H_FIFO_IntStat.FIFO_Empty, and H_FIFO_IntStat.FIFO_NotEmpty.

Bits5-4 Reserved

Bit3 JoinDMA1

Performs a DMA1 transfer with the FIFO of channel CHb. The transfer direction is determined by the DMA1_Control.Dir bit.

Bit2 JoinDMA0

Performs a DMA0 transfer with the FIFO of channel CHb. The transfer direction is determined by the DMA0_Control.Dir bit.

Bit1 JoinCPU_Rd

Performs a read transfer of CPU register access with the FIFO of channel CHb. Namely, when a read of the FIFO_Rd_0,1 registers or FIFO_ByteRd register is performed, data is read from the FIFO of this channel.

Bit0 JoinCPU_Wr

Performs a write transfer of CPU register access with the FIFO of channel CHb. Namely, when a write to the FIFO_Wr_0,1 registers is performed, data is written to the FIFO of this channel.

If the JoinDMAx{x=0,1} bit is set, it is possible to refer to the remaining bytes of data in the FIFO when the DMA0_Control.Dir bit = 1 or the amount of free space in the FIFO when the DMA0_Control.Dir bit = 0 by inspecting the DMAx{x=0,1}_Remain_H,L registers.

If the JoinCPU_Rd or JoinCPU_Wr bit is set, it is possible to write or read data to or from the FIFO_Rd_0,1, FIFO_ByteRd, or FIFO_Wr_0,1 registers after inspecting the FIFO_RdRemain_H,L or FIFO_WrRemain_H,L registers.

The JoinDMA1, JoinDMA0, JoinCPU_Rd, and JoinCPU_Wr bits can be set to 1 only one bit at a time. If an attempt is made to set two or more of these bits by writing 1 at the same time, only the higher bit is set.

7.6.93 150h H_CHcConfig_0(Host Channel c Configuration0)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	150h	CHcConfig_0	R / W	7: ACK_Cnt[3]	Channel c ACK Count		00h
				6: ACK_Cnt[2]			
				5: ACK_Cnt[1]			
				4: ACK_Cnt[0]			
			R / W	3: SpeedMode[1]	Channel c Speed Mode		
				2: SpeedMode[0]			
			R / W	1: Toggle	0: Toggle0	1: Toggle1	
			R / W	0: TranGo	0: Stand by	1: Transaction Start	

This register is used to make basic settings of channel CHc during USB host operation.

Bits7-4 ACK_Cnt [3:0]

These bits set the number of ACK count in a transfer performed on channel CHc.

When ACK count reaches the set value, the TranACK bit in the H_CHcIntStat register is set.

0000: ACK is counted up to 16 times.

0001–1111: ACK is counted anywhere between 1 to 15 times.

Bits3-2 SpeedMode [1:0]

These bits set the operation mode of the device that performs a transfer on channel CHc.

00: HS mode — Set this for HS devices.

01: FS mode — Set this for FS devices.

10: Reserved — Use of this value is prohibited.

11: LS mode — Set this for LS devices.

Bit1 Toggle

This bit sets the initial value of the toggle sequence bit with which a transaction is to be started. While a transaction is under execution and after a transaction is complete, this bit indicates the status of the toggle sequence bit.

0: Toggle 0

1: Toggle 1

Bit0 TranGo

Setting this bit to 1 causes a transaction on channel CHc to start. Transaction processing can be stopped by clearing this bit to 0 after a transaction started. This bit also serves as a status indicating whether a transaction on channel CHc is under execution or not.

0: Stops a transaction (transaction stopped)

1: Starts a transaction (transaction under execution)

When a transfer for the number of bytes set by the H_CHcTotalSize_HH–LL registers is complete, the TranCmp bit in the H-CHcIntStat register is set to 1, at which time this bit is automatically reset to 0. Furthermore, this bit is reset to 0 when the ChangeCondition bit in the H_CHcIntStat register is set. In that case, inspect the H_CHcConditionCode register to find a reason for which the condition bit was set.

If a transaction is stopped by clearing this bit, the ChangeCondition bit in the H_CHcIntStat register is set when the transaction being processed is complete. Even when a transaction is stopped, the data present in the FIFO, the (remaining) total size, and channel-related settings are left intact. Therefore, a transaction can be resumed from where it stopped by setting this bit to 1 again. (To perform a new transaction, clear the FIFO and set channel information over again.)

7.6.94 151h H_CHcConfig_1(Host Channel c Configuration1)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	151h	H_CHcConfig_1	R / W	7: PID[1]	Channel c PID		00h
				6: PID[0]			
			R / W	5: TranType[1]	Channel c Transfer Type		
				4: TranType[0]			
			R / W	3: AutoZerolen	0: Do nothing	1: Add Zerolen	
				2:	0:	1:	
				1:	0:	1:	
			R / W	0: TotalSizeFree	0: Do nothing	1: Total Size Free	

This register is used to make basic settings of channel CHc during USB host operation.

Bits7-6 TID[1:0]

These bits set the type of token (OUT or IN) that is issued on channel CHc.

00: Reserved — Use of this value is prohibited.

01: OUT — Issues an OUT token.

10: IN — Issues an IN token.

11: Reserved — Use of this value is prohibited.

Bits5-4 TranType[1:0]

These bits set the type of transfer that is performed on channel CHc.

00: Reserved — Use of this value is prohibited.

01: Reserved — Use of this value is prohibited.

10: Bulk — Performs a bulk transfer.

11: Interrupt — Performs an interrupt transfer.

Bit3 AutoZerolen

Setting this bit to 1 enables the auto zero-length function, so that when a transfer for the data bytes set by H_CHcTotalSizeHH–LL registers is completed exactly with MaxPacketSize, a zero-length packet is automatically attached at the end of the transfer. This bit is effective for only OUT transfers.

Bits2-1 Reserved

Bit 0 TotalSizeFree

Setting this bit to 1 makes the transfer size infinite regardless of the value set in the H_CHcTotalSizeHH–LL registers.

7.6.95 152h H_CHcMaxPktSize_H (Host Channel c Max Packet Size High)

7.6.96 153h H_CHcMaxPktSize_L (Host Channel c Max Packet Size Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	152h	H_CHcMaxPktSize_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R / W	1: MaxPktSize[9]	Channel c Max Packet Size High		
				0: MaxPktSize[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	153h	H_CHcMaxPktSize_L	R / W	7: MaxPktSize[7]	Channel c Max Packet Size Low	00h
				6: MaxPktSize[6]		
				5: MaxPktSize[5]		
				4: MaxPktSize[4]		
				3: MaxPktSize[3]		
				2: MaxPktSize[2]		
				1: MaxPktSize[1]		
				0: MaxPktSize[0]		

This register sets MaxPacketSize of channel CHc during USB host operation.

152h.Bit7-2 Reserved

Do not write 1 to the reserved bits.

152h.Bit1-0, 153h.Bit7-0 MaxPktSize[9:0]

These bits set the MaxPacketSize of channel CHc.

When using this channel for bulk transfers, set one of the following transfer sizes.

During FS	8, 16, 32, or 64 bytes
During HS	512 bytes

When using this channel for interrupt transfers, set any desired transfer size within the limits given below.

During LS	Up to 8 bytes
During FS	Up to 64 bytes
During HS	Up to 512 bytes

Setting any other transfer size is prohibited.

7.6.97 154h H_CHcTotalSize_HH (Host Channel c Total Size High-High)

7.6.98 155h H_CHcTotalSize_HL (Host Channel c Total Size High-Low)

7.6.99 156h H_CHcTotalSize_LH (Host Channel c Total Size Low-High)

7.6.100 157h H_CHcTotalSize_LL (Host Channel c Total Size Low-Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	154h	H_CHcTotalSize_HH	R / W	7: TotalSize[31]	Channel c Total Size High-High	00h
				6: TotalSize[30]		
				5: TotalSize[29]		
				4: TotalSize[28]		
				3: TotalSize[27]		
				2: TotalSize[26]		
				1: TotalSize[25]		
				0: TotalSize[24]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	155h	H_CHcTotalSize_HL	R / W	7: TotalSize[23]	Channel c Total Size High-Low	00h
				6: TotalSize[22]		
				5: TotalSize[21]		
				4: TotalSize[20]		
				3: TotalSize[19]		
				2: TotalSize[18]		
				1: TotalSize[17]		
				0: TotalSize[16]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	156h	H_CHcTotalSize_LH	R / W	7: TotalSize[15]	Channel c Total Size Low-High	00h
				6: TotalSize[14]		
				5: TotalSize[13]		
				4: TotalSize[12]		
				3: TotalSize[11]		
				2: TotalSize[10]		
				1: TotalSize[9]		
				0: TotalSize[8]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	157h	H_CHcTotalSize_LL	R / W	7: TotalSize[7]	Channel c Total Size Low-Low	00h
				6: TotalSize[6]		
				5: TotalSize[5]		
				4: TotalSize[4]		
				3: TotalSize[3]		
				2: TotalSize[2]		
				1: TotalSize[1]		
				0: TotalSize[0]		

These registers set the Total Size of the data to be transferred on channel CHc during USB host operation.

154h.Bit7-0, 155h.Bit7-0, 156h.Bit7-0, 157h.Bit7-0 TotalSize[31:0]

These bits set a total number of data bytes to be transferred on channel CHc (maximum 4,294,967,295 bytes: approx. 4 Gbytes).

Once a transaction is started by the TranGo bit in the H_CHcConfig_0 register, the remaining number of bytes to be transferred can be read through this register.

When bits 31–24 (H_CHcTotalSize_HH register) are read, the values of bits 23–16 (H_CHcTotalSize_HL register), bits 15–8 (H_CHcTotalSize_LH register), and bits 7–0 (H_CHcTotalSize_LL register) all are fixed. (Even when the read value is fixed, the internal counter of the LSI continues counting.)

Therefore, to read the remaining number of transfer bytes through 8-bit register accesses, read H_CHaTotalSize_HH, H_CHaTotalSize_HL, H_CHaTotalSize_LH, and H_CHaTotalSize_LL in that order.

Note also that if an OUT transaction is executed with TotalSize = 0, a zero-length packet is transmitted.

7.6.101 158h H_CHcHubAdrs (Host Channel c Hub Address)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	148h	H_CHcHubAdrs	R / W	7: HubAdrs[3]	Channel c Hub Address	00h
				6: HubAdrs[2]		
				5: HubAdrs[1]		
				4: HubAdrs[0]		
				3:	0: 1:	
			R / W	2: Port[2]	Channel c Port Number	
				1: Port[1]		
				0: Port[0]		

This register sets the hub that is connected to channel CHc during USB host operation.

Bits7-4 HubAdrs[3:0]

These bits set the USB address of the hub to which a function is connected that performs a transfer on channel CHc.

Any value in the range of 0 to 15 can be set.

Bit3 Reserved

Bits2-0 Port[2:0]

These bits set the port number of the hub to which a function is connected that performs a transfer on channel CHc.

Any value in the range of 0 to 7 can be set.

7.6.102 159h H_CHcFuncAdrs (Host Channel c Function Address)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	159h	H_CHcFuncAdrs	R / W	7: FuncAdrs[3]	Channel c Function Address	00h
				6: FuncAdrs[2]		
				5: FuncAdrs[1]		
				4: FuncAdrs[0]		
			R / W	3: EP_Number[3]	Channel c Endpoint Number	
				2: EP_Number[2]		
				1: EP_Number[1]		
				0: EP_Number[0]		

This register sets the address of a function that performs a transfer on channel CHc during USB host operation.

Bits7-4 FuncAdrs[3:0]

These bits set the USB address of the function that includes the endpoint managed by channel CHc.

Any value in the range of 0 to 15 can be set.

Bits3-0 EP_Number[3:0]

These bits set the endpoint number that performs a transfer on channel CHc.

Any value in the range of 0 to 15 can be set.

7.6.103 15Ah H_CHcInterval_H(Host Channel c Interval High)

7.6.104 15Bh H_CHcInterval_L(Host Channel c Interval Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	15Ah	H_CHcInterval_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R / W	2: Interval[10]	Channel c Interrupt Transfer Interval High		
				1: Interval[9]			
				0: Interval[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	15Bh	H_CHcInterval_L	R / W	7: Interval[7]	Channel c Interrupt Transfer Interval Low	00h
				6: Interval[6]		
				5: Interval[5]		
				4: Interval[4]		
				3: Interval[3]		
				2: Interval[2]		
				1: Interval[1]		
				0: Interval[0]		

This register sets an interval value for the interrupt transfers to be performed on channel CHc during USB host operation.

15Ah.Bit7-3 **Reserved**

15Ah.Bit2-0, 15Bh.Bit7-0 **Interval[10:0]**

These bits set the interval (period) at which intervals tokens for interrupt transfers will be issued. The 3 low-order bits specify the interval time in microframes (125 μs), and the 7 high-order bits specify the interval time in frames (ms). Settings of this register are effective only when the H_CHcConfig1 register's TranType bits = 11 (interrupt transfer). Setting the value "0d" in this register has no effect.

The interval time set in this register is also used when transactions are retransmitted.

Interval[2:0] μFrame — Specifies an interval time in 125 μs units. Set this interval time to 1, 2, or 4 microframes. Setting any other value is prohibited. When setting these bits, make sure Interval[10:3] are all set to 0s.

Interval[10:3] Frame — Specifies an interval time in ms units. This interval time can be set to any value within the range 1 to 255 frames. When setting these bits, make sure Interval[2:0] are all set to 0s.

7.6.105 15Ch~15Dh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	15Ch -15Dh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.6.106 15Eh H_CHcConditionCode (Host Channel c Condition Code)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	15Eh	H_CHcConditionCode		7:	0:	1:	00h
			R	6: ConditionCode[2]	Channel c Condition Code		
			R	5: ConditionCode[1]	Channel c Condition Code		
			R	4: ConditionCode[0]	Channel c Condition Code		
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register indicates the result of a transfer that was completed on channel CHc during USB host operation.

Bit7 Reserved

Bits6-4 ConditionCode[2:0]

These bits indicate the result of a transfer that was completed on channel CHc.

Code	Meaning	Description
000	NOERROR	The transaction is completed without error.
001	STALL	The endpoint has returned stall PID.
010	DATAOVERRUN	• The amount of data received exceeds the maximum allowed size of a data packet (MaxPacketSize). * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error. • The amount of data received exceeds the maximum allowed IRP (TotalSize) * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error. * If the received data packet is equal to or less than MaxPacketSize and the data toggle included in it does not match the expected value, this is processed as a toggle mismatch, and not as a data overrun.
011	DATAUNDERRUN	• The endpoint has returned bytes of data which are less than MaxPacketSize and are insufficient to fill the IRP (TotalSize). * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error.

Code	Meaning	Description
100	RetryError	• The device does not respond to a token within a designated time (IN) or does not transmit a handshaking signal within a designated time (OUT). • The data packet from the endpoint contains a CRC error. • The data packet from the endpoint contains a bit stuffing error. • The PID inspection bit from the endpoint failed in data PID (IN) or handshaking (OUT). • The received PID is invalid or has no PID values defined. • An ERR handshaking signal was received in a split transaction of interrupt transfer. • Three NYET handshaking signals were received consecutively in a split transaction of interrupt transfer. • The data toggle included in the data packet from the endpoint does not match the expected value (toggle mismatch).
Other	Reserved	

Bits3-0 Reserved

7.6.107 15Fh H_CHcJoin (Host Channel c Join)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	15Fh	H_CHcJoin	R / W	7: JoinIDE	0: Do nothing	1: Join CHc to IDE	00h
			R / W	6: JoinFIFO_Stat	0: Do nothing	1 Join CHc to show Status	
				5:	0:	1:	
				4:	0	1:	
			R / W	3: JoinDMA1	0: Do nothing	1: Join CHc to DMA1	
			R / W	2: JoinDMA0	0: Do nothing	1: Join CHc to DMA0	
			R / W	1: JoinCPU_Rd	0: Do nothing	1: Join CHc to CPU_Rd	
			R / W	0: JoinCPU_Wr	0: Do nothing	1: Join CHc to CPU_Wr	

This register specifies the port with which data transfers with channel CHc are to be performed during USB host operation.

Bit7 JoinIDE

Performs an IDE data transfer with the FIFO of channel CHc. The transfer direction is determined by the IDE_Control.Dir bit.

Bit6 JoinFIFO_Stat

This bit permits the Full, Empty, or NotEmpty status of the FIFO of channel CHc to be monitored with H_FIFO_IntStat.FIFO_Full, H_FIFO_IntStat.FIFO_Empty, and H_FIFO_IntStat.FIFO_NotEmpty.

Bits5-4 Reserved

Bit3 JoinDMA1

Performs a DMA1 transfer with the FIFO of channel CHc. The transfer direction is determined by the DMA1_Control.Dir bit.

Bit2 JoinDMA0

Performs a DMA0 transfer with the FIFO of channel CHc. The transfer direction is determined by the DMA0_Control.Dir bit.

Bit1 JoinCPU_Rd

Performs a read transfer of CPU register access with the FIFO of channel CHc. Namely, when a read of the FIFO_Rd_0,1 registers or FIFO_ByteRd register is performed, data is read from the FIFO of this channel.

Bit0 JoinCPU_Wr

Performs a write transfer of CPU register access with the FIFO of channel CHc. Namely, when a write to the FIFO_Wr_0,1 registers is performed, data is written to the FIFO of this channel.

If the JoinDMAx{x=0,1} bit is set, it is possible to refer to the remaining bytes of data in the FIFO when the DMA0_Control.Dir bit = 1 or the amount of free space in the FIFO when the DMA0_Control.Dir bit = 0 by inspecting the DMAx{x=0,1}_Remain_H,L registers.

If the JoinCPU_Rd or JoinCPU_Wr bit is set, it is possible to write or read data to or from the FIFO_Rd_0,1, FIFO_ByteRd, or FIFO_Wr_0,1 registers after inspecting the FIFO_RdRemain_H,L or FIFO_WrRemain_H,L registers.

The JoinDMA1, JoinDMA0, JoinCPU_Rd, and JoinCPU_Wr bits can be set to 1 only one bit at a time. If an attempt is made to set two or more of these bits by writing 1 at the same time, only the higher bit is set.

7.6.108 160h H_CHdConfig_0(Host Channel d Configuration0)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	160h	CHdConfig_0	R / W	7: ACK_Cnt[3]	Channel d ACK Count		00h
				6: ACK_Cnt[2]			
				5: ACK_Cnt[1]			
				4: ACK_Cnt[0]			
			R / W	3: SpeedMode[1]	Channel d Speed Mode		
				2: SpeedMode[0]			
			R / W	1: Toggle	0: Toggle0	1: Toggle1	
			R / W	0: TranGo	0: Stand by	1: Transaction Start	

This register is used to make basic settings of channel CHd during USB host operation.

Bits7-4 ACK_Cnt [3:0]

These bits set the number of ACK count in a transfer performed on channel CHd.

When ACK count reaches the set value, the TranACK bit in the H_CHdIntStat register is set.

0000: ACK is counted up to 16 times.

0001–1111: ACK is counted anywhere between 1 to 15 times.

Bits3-2 SpeedMode [1:0]

These bits set the operation mode of the device that performs a transfer on channel CHd.

00: HS mode — Set this for HS devices.

01: FS mode — Set this for FS devices.

10: Reserved — Use of this value is prohibited.

11: LS mode — Set this for LS devices.

Bit1 Toggle

This bit sets the initial value of the toggle sequence bit with which a transaction is to be started. While a transaction is under execution and after a transaction is complete, this bit indicates the status of the toggle sequence bit.

0: Toggle 0

1: Toggle 1

Bit0 TranGo

Setting this bit to 1 causes a transaction on channel CHd to start. Transaction processing can be stopped by clearing this bit to 0 after a transaction started. This bit also serves as a status indicating whether a transaction on channel CHd is under execution or not.

0: Stops a transaction (transaction stopped)

1: Starts a transaction (transaction under execution)

When a transfer for the number of bytes set by the H_CHdTotalSize_HH–LL registers is complete, the TranCmp bit in the H-CHdIntStat register is set to 1, at which time this bit is automatically reset to 0. Furthermore, this bit is reset to 0 when the ChangeCondition bit in the H_CHdIntStat register is set. In that case, inspect the H_CHdConditionCode register to find a reason for which the condition bit was set.

If a transaction is stopped by clearing this bit, the ChangeCondition bit in the H_CHdIntStat register is set when the transaction being processed is complete. Even when a transaction is stopped, the data present in the FIFO, the (remaining) total size, and channel-related settings are left intact. Therefore, a transaction can be resumed from where it stopped by setting this bit to 1 again. (To perform a new transaction, clear the FIFO and set channel information over again.)

7.6.109 161h H_CHdConfig_1(Host Channel d Configuration1)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	161h	H_CHdConfig_1	R / W	7: TID[1]	Channel d Transaction ID		00h
				6: TID[0]			
			R / W	5: TranType[1]	Channel d Transfer Type		
				4: TranType[0]			
			R / W	3: AutoZerolen	0: Do nothing	1: Add Zerolen	
				2:	0:	1:	
				1:	0:	1:	
			R / W	0: TotalSizeFree	0: Do nothing	1: Total Size Free	

This register is used to make basic settings of channel CHd during USB host operation.

Bits7-6 TID[1:0]

These bits set the type of token (OUT or IN) that is issued on channel CHd.

00: Reserved — Use of this value is prohibited.

01: OUT — Issues an OUT token.

10: IN — Issues an IN token.

11: Reserved — Use of this value is prohibited.

Bits5-4 TranType[1:0]

These bits set the type of transfer that is performed on channel CHd.

00: Reserved — Use of this value is prohibited.

01: Reserved — Use of this value is prohibited.

10: Bulk — Performs a bulk transfer.

11: Interrupt — Performs an interrupt transfer.

Bit3 AutoZerolen

Setting this bit to 1 enables the auto zero-length function, so that when a transfer for the data bytes set by H_CHdTotalSizeHH–LL registers is completed exactly with MaxPacketSize, a zero-length packet is automatically attached at the end of the transfer. This bit is effective for only OUT transfers.

Bits2-1 Reserved

Bit0 TotalSizeFree

Setting this bit to 1 makes the transfer size infinite regardless of the value set in the H_CHdTotalSizeHH–LL registers.

7.6.110 162h H_CHdMaxPktSize_H (Host Channel d Max Packet Size High)

7.6.111 163h H_CHdMaxPktSize_L (Host Channel d Max Packet Size Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	162h	H_CHdMaxPktSize_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R / W	1: MaxPktSize[9]	Channel d Max Packet Size High		
				0: MaxPktSize[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	163h	H_CHdMaxPktSize_L	R / W	7: MaxPktSize[7]	Channel d Max Packet Size Low	00h
				6: MaxPktSize[6]		
				5: MaxPktSize[5]		
				4: MaxPktSize[4]		
				3: MaxPktSize[3]		
				2: MaxPktSize[2]		
				1: MaxPktSize[1]		
				0: MaxPktSize[0]		

This register sets MaxPacketSize of channel CHd during USB host operation.

162h.Bit7-2 Reserved

Do not write 1 to the reserved bits.

162h.Bit1-0, 163h.Bit7-0 MaxPktSize[9:0]

These bits set the MaxPacketSize of channel CHd.

When using this channel for bulk transfers, set one of the following transfer sizes.

During FS	8, 16, 32, or 64 bytes
During HS	512 bytes

When using this channel for interrupt transfers, set any desired transfer size within the limits given below.

During LS	Up to 8 bytes
During FS	Up to 64 bytes
During HS	Up to 512 bytes

Setting any other transfer size is prohibited.

7.6.112 164h H_CHdTotalSize_HH (Host Channel d Total Size High-High)

7.6.113 165h H_CHdTotalSize_HL (Host Channel d Total Size High-Low)

7.6.114 166h H_CHdTotalSize_LH (Host Channel d Total Size Low-High)

7.6.115 167h H_CHdTotalSize_LL (Host Channel d Total Size Low-Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	164h	H_CHdTotalSize_HH	R / W	7: TotalSize[31]	Channel d Total Size High-High	00h
				6: TotalSize[30]		
				5: TotalSize[29]		
				4: TotalSize[28]		
				3: TotalSize[27]		
				2: TotalSize[26]		
				1: TotalSize[25]		
				0: TotalSize[24]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	165h	H_CHdTotalSize_HL	R / W	7: TotalSize[23]	Channel d Total Size High-Low	00h
				6: TotalSize[22]		
				5: TotalSize[21]		
				4: TotalSize[20]		
				3: TotalSize[19]		
				2: TotalSize[18]		
				1: TotalSize[17]		
				0: TotalSize[16]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	166h	H_CHdTotalSize_LH	R / W	7: TotalSize[15]	Channel d Total Size Low-High	00h
				6: TotalSize[14]		
				5: TotalSize[13]		
				4: TotalSize[12]		
				3: TotalSize[11]		
				2: TotalSize[10]		
				1: TotalSize[9]		
				0: TotalSize[8]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	167h	H_CHdTotalSize_LL	R / W	7: TotalSize[7]	Channel d Total Size Low-Low	00h
				6: TotalSize[6]		
				5: TotalSize[5]		
				4: TotalSize[4]		
				3: TotalSize[3]		
				2: TotalSize[2]		
				1: TotalSize[1]		
				0: TotalSize[0]		

These registers set the Total Size of the data to be transferred on channel CHd during USB host operation.

166h.Bit7-0, 167h.Bit7-0, 168h.Bit7-0, 169h.Bit7-0 TotalSize[31:0]

These bits set a total number of data bytes to be transferred on channel CHd (maximum 4,294,967,295 bytes: approx. 4 Gbytes).

Once a transaction is started by the TranGo bit in the H_CHdConfig_0 register, the remaining number of bytes to be transferred can be read through this register.

When bits 31–24 (H_CHdTotalSize_HH register) are read, the values of bits 23–16 (H_CHdTotalSize_HL register), bits 15–8 (H_CHdTotalSize_LH register), and bits 7–0 (H_CHdTotalSize_LL register) all are fixed. (Even when the read value is fixed, the internal counter of the LSI continues counting.)

Therefore, to read the remaining number of transfer bytes through 8-bit register accesses, read H_CHaTotalSize_HH, H_CHaTotalSize_HL, H_CHaTotalSize_LH, and H_CHaTotalSize_LL in that order.

Note also that if an OUT transaction is executed with TotalSize = 0, a zero-length packet is transmitted.

7.6.116 168h H_CHdHubAdrs (Host Channel d Hub Address)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	168h	H_CHdHubAdrs	R / W	7: HubAdrs[3]	Channel d Hub Address		00h
				6: HubAdrs[2]			
				5: HubAdrs[1]			
				4: HubAdrs[0]			
				3:	0:	1:	
			R / W	2: Port[2]	Channel d Port Number		
				1: Port[1]			
				0: Port[0]			

This register sets the hub that is connected to channel CHd during USB host operation.

Bits7-4 HubAdrs[3:0]

These bits set the USB address of the hub to which a function is connected that performs a transfer on channel CHd.

Any value in the range of 0 to 15 can be set.

Bit3 Reserved

Bits2-0 Port[2:0]

These bits set the port number of the hub to which a function is connected that performs a transfer on channel CHd.

Any value in the range of 0 to 7 can be set.

7.6.117 169h H_CHdFuncAdrs (Host Channel d Function Address)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	169h	H_CHdFuncAdrs	R / W	7: FuncAdrs[3]	Channel d Function Address	00h
				6: FuncAdrs[2]		
				5: FuncAdrs[1]		
				4: FuncAdrs[0]		
			R / W	3: EP_Number[3]	Channel d Endpoint Number	
				2: EP_Number[2]		
				1: EP_Number[1]		
				0: EP_Number[0]		

This register sets the address of a function that performs a transfer on channel CHd during USB host operation.

Bits7-4 FuncAdrs[3:0]

These bits set the USB address of the function that includes the endpoint managed by channel CHd.

Any value in the range of 0 to 15 can be set.

Bits3-0 EP_Number[3:0]

These bits set the endpoint number that performs a transfer on channel CHd.

Any value in the range of 0 to 15 can be set.

7.6.118 16Ah H_CHdInterval_H(Host Channel d Interval High)

7.6.119 16Bh H_CHdInterval_L(Host Channel d Interval Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	16Ah	H_CHdInterval_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R / W	2: Interval[10]	Channel d Interrupt Transfer Interval High		
				1: Interval[9]			
				0: Interval[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	16Bh	H_CHdInterval_L	R / W	7: Interval[7]	Channel d Interrupt Transfer Interval Low	00h
				6: Interval[6]		
				5: Interval[5]		
				4: Interval[4]		
				3: Interval[3]		
				2: Interval[2]		
				1: Interval[1]		
				0: Interval[0]		

This register sets an interval value for the interrupt transfers to be performed on channel CHd during USB host operation.

16Ah.Bit7-3 Reserved

16Ah.Bit2-0, 16Bh.Bit7-0 Interval[10:0]

These bits set the interval (period) at which intervals tokens for interrupt transfers will be issued. The 3 low-order bits specify the interval time in microframes (125 µs), and the 7 high-order bits specify the interval time in frames (ms). Settings of this register are effective only when the H_CHdConfig1 register's TranType bits = 11 (interrupt transfer). Setting the value "0d" in this register has no effect.

The interval time set in this register is also used when transactions are retransmitted.

Interval[2:0] µFrame — Specifies an interval time in 125 µs. Set this interval time to 1, 2, or 4 microframes. Setting any other value is prohibited. When setting these bits, make sure Interval[10:3] are all set to 0s.

Interval[10:3] Frame — Specifies an interval time in ms units. This interval time can be set to any value within the range of 1 to 255 frames. When setting these bits, make sure Interval[2:0] are all set to 0s.

7.6.120 16Ch~16Dh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	16Ch -16Dh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.6.121 16Eh H_CHdConditionCode (Host Channel d Condition Code)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	16Eh	H_CHdConditionCode		7:	0:	1:	00h
			R	6: ConditionCode[2]	Channel d Condition Code		
				5: ConditionCode[1]			
				4: ConditionCode[0]			
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register indicates the result of a transfer that was completed on channel CHd during USB host operation.

Bit7 Reserved

Bits6-4 ConditionCode[2:0]

These bits indicate the result of a transfer that was completed on channel CHd.

Code	Meaning	Description
000	NOERROR	The transaction is completed without error.
001	STALL	The endpoint has returned stall PID.
010	DATAOVERRUN	• The amount of data received exceeds the maximum allowed size of a data packet (MaxPacketSize). * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error. • The amount of data received exceeds the maximum allowed IRP (TotalSize) * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error. * If the received data packet is equal to or less than MaxPacketSize and the data toggle included in it does not match the expected value, this is processed as a toggle mismatch, and not as a data overrun.
011	DATAUNDERRUN	• The endpoint has returned bytes of data which are less than MaxPacketSize and are insufficient to fill the IRP (TotalSize). * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error.

Code	Meaning	Description
100	RETRYERROR	• The device does not respond to a token within a designated time (IN) or does not transmit a handshaking signal within a designated time (OUT). • The data packet from the endpoint contains a CRC error. • The data packet from the endpoint contains a bit stuffing error. • The PID inspection bit from the endpoint failed in data PID (IN) or handshaking (OUT). • The received PID is invalid or has no PID values defined. • An ERR handshaking signal was received in a split transaction of interrupt transfer. • Three NYET handshaking signals were received consecutively in a split transaction of interrupt transfer. • The data toggle included in the data packet from the endpoint does not match the expected value (toggle mismatch).
Other	Reserved	

Bits3-0 Reserved

7.6.122 16Fh H_CHdJoin (Host Channel d Join)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	16Fh	H_CHdJoin	R / W	7: JoinIDE	0: Do nothing	1: Join CHd to IDE	00h
			R / W	6: JoinFIFO_Stat	0: Do nothing	1 Join CHd to show Status	
				5:	0:	1:	
				4:	0	1:	
			R / W	3: JoinDMA1	0: Do nothing	1: Join CHd to DMA1	
			R / W	2: JoinDMA0	0: Do nothing	1: Join CHd to DMA0	
			R / W	1: JoinCPU_Rd	0: Do nothing	1: Join CHd to CPU_Rd	
			R / W	0: JoinCPU_Wr	0: Do nothing	1: Join CHd to CPU_Wr	

This register specifies the port with which data transfers with channel CHd are to be performed during USB host operation.

Bit7 JoinIDE

Performs an IDE data transfer with the FIFO of channel CHd. The transfer direction is determined by the IDE_Control.Dir bit.

Bit6 JoinFIFO_Stat

This bit permits the Full, Empty, or NotEmpty status of the FIFO of channel CHd to be monitored with H_FIFO_IntStat.FIFO_Full, H_FIFO_IntStat.FIFO_Empty, and H_FIFO_IntStat.FIFO_NotEmpty.

Bits5-4 Reserved

Bit3 JoinDMA1

Performs a DMA1 transfer with the FIFO of channel CHd. The transfer direction is determined by the DMA1_Control.Dir bit.

Bit2 JoinDMA0

Performs a DMA0 transfer with the FIFO of channel CHd. The transfer direction is determined by the DMA0_Control.Dir bit.

Bit1 JoinCPU_Rd

Performs a read transfer of CPU register access with the FIFO of channel CHd. Namely, when a read of the FIFO_Rd_0,1 registers or FIFO_ByteRd register is performed, data is read from the FIFO of this channel.

Bit0 JoinCPU_Wr

Performs a write transfer of CPU register access with the FIFO of channel CHd. Namely, when a write to the FIFO_Wr_0,1 registers is performed, data is written to the FIFO of this channel.

If the JoinDMAx{x=0,1} bit is set, it is possible to refer to the remaining bytes of data in the FIFO when the DMA0_Control.Dir bit = 1 or the amount of free space in the FIFO when the DMA0_Control.Dir bit = 0 by inspecting the DMAx{x=0,1}_Remain_H,L registers.

If the JoinCPU_Rd or JoinCPU_Wr bit is set, it is possible to write or read data to or from the FIFO_Rd_0,1, FIFO_ByteRd, or FIFO_Wr_0,1 registers after inspecting the FIFO_RdRemain_H,L or FIFO_WrRemain_H,L registers.

The JoinDMA1, JoinDMA0, JoinCPU_Rd, and JoinCPU_Wr bits can be set to 1 only one bit at a time. If an attempt is made to set two or more of these bits by writing 1 at the same time, only the higher bit is set.

7.6.123 170h H_CHeConfig_0(Host Channel e Configuration0)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	170h	CHeConfig_0	R / W	7: ACK_Cnt[3]	Channel e ACK Count		00h
				6: ACK_Cnt[2]			
				5: ACK_Cnt[1]			
				4: ACK_Cnt[0]			
			R / W	3: SpeedMode[1]	Channel e Speed Mode		
				2: SpeedMode[0]			
			R / W	1: Toggle	0: Toggle0	1: Toggle1	
			R / W	0: TranGo	0: Stand by	1: Transaction Start	

This register is used to make basic settings of channel CHe during USB host operation.

Bits7-4 ACK_Cnt [3:0]

These bits set the number of ACK count in a transfer performed on channel CHe.

When ACK count reaches the set value, the TranACK bit in the H_CHeIntStat register is set.

0000: ACK is counted up to 16 times.

0001–1111: ACK is counted anywhere between 1 to 15 times.

Bits3-2 SpeedMode [1:0]

These bits set the operation mode of the device that performs a transfer on channel CHe.

00: HS mode — Set this for HS devices.

01: FS mode — Set this for FS devices.

10: Reserved — Use of this value is prohibited.

11: LS mode — Set this for LS devices.

Bit1 Toggle

This bit sets the initial value of the toggle sequence bit with which a transaction is to be started. While a transaction is under execution and after a transaction is complete, this bit indicates the status of the toggle sequence bit.

0: Toggle 0

1: Toggle 1

Bit0 TranGo

Setting this bit to 1 causes a transaction on channel CHe to start. Transaction processing can be stopped by clearing this bit to 0 after a transaction started. This bit also serves as a status indicating whether a transaction on channel CHe is under execution or not.

0: Stops a transaction (transaction stopped)

1: Starts a transaction (transaction under execution)

When a transfer for the number of bytes set by the H_CHeTotalSize_HH–LL registers is complete, the TranCmp bit in the H-CHeIntStat register is set to 1, at which time this bit is automatically reset to 0. Furthermore, this bit is reset to 0 when the ChangeCondition bit in the H_CHeIntStat register is set. In that case, inspect the H_CHeConditionCode register to find a reason for which the condition bit was set.

If a transaction is stopped by clearing this bit, the ChangeCondition bit in the H_CHeIntStat register is set when the transaction being processed is complete. Even when a transaction is stopped, the data present in the FIFO, the (remaining) total size, and channel-related settings are left intact. Therefore, a transaction can be resumed from where it is stopped by setting this bit to 1 again. (To perform a new transaction, clear the FIFO and set channel information over again.)

7.6.124 171h H_CHeConfig_1(Host Channel e Configuration1)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	171h	CHeConfig_1	R / W	7: TID[1]	Channel e Transaction ID		00h
				6: TID[0]			
			R / W	5: TranType[1]	Channel e Transfer Type		
				4: TranType[0]			
			R / W	3: AutoZerolen	0: Do nothing	1: Add Zerolen	
				2:	0:	1:	
				1:	0:	1:	
			R / W	0: TotalSizeFree	0: Do nothing	1: Total Size Free	

This register is used to make basic settings of channel CHe during USB host operation.

Bits7-6 PID[1:0]

These bits set the type of token (OUT or IN) that is issued on channel CHe.

00: Reserved — Use of this value is prohibited.

01: OUT — Issues an OUT token.

10: IN — Issues an IN token.

11: Reserved — Use of this value is prohibited.

Bits5-4 TranType[1:0]

These bits set the type of transfer that is performed on channel CHe.

00: Reserved — Use of this value is prohibited.

01: Reserved — Use of this value is prohibited.

10: Bulk — Performs a bulk transfer.

11: Interrupt — Performs an interrupt transfer.

Bit3 AutoZerolen

Setting this bit to 1 enables the auto zero-length function, so that when a transfer for the data bytes set by H_CHeTotalSizeHH–LL registers is completed exactly with MaxPacketSize, a zero-length packet is automatically attached at the end of the transfer. This bit is effective for only OUT transfers.

Bits2-1 Reserved

Bit0 TotalSizeFree

Setting this bit to 1 makes the transfer size infinite regardless of the value set in the H_CHeTotalSizeHH–LL registers.

7.6.125 172h H_CHeMaxPktSize_H (Host Channel e Max Packet Size High)

7.6.126 173h H_CHeMaxPktSize_L (Host Channel e Max Packet Size Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	172h	H_CHeMaxPktSize_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R / W	1: MaxPktSize[9]	Channel e Max Packet Size High		
				0: MaxPktSize[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	173h	H_CHeMaxPktSize_L	R / W	7: MaxPktSize[7]	Channel e Max Packet Size Low	00h
				6: MaxPktSize[6]		
				5: MaxPktSize[5]		
				4: MaxPktSize[4]		
				3: MaxPktSize[3]		
				2: MaxPktSize[2]		
				1: MaxPktSize[1]		
				0: MaxPktSize[0]		

This register sets MaxPacketSize of channel CHe during USB host operation.

172h.Bit7-2 Reserved

Do not write 1 to the reserved bits.

172h.Bit1-0, 173h.Bit7-0 MaxPktSize[9:0]

These bits set the MaxPacketSize of channel CHe.

When using this channel for bulk transfers, set one of the following transfer sizes.

During FS	8, 16, 32, or 64 bytes
During HS	512 bytes

When using this channel for interrupt transfers, set any desired transfer size within the limits given below.

During LS	Up to 8 bytes
During FS	Up to 64 bytes
During HS	Up to 512 bytes

Setting any other transfer size is prohibited.

7.6.127 174h H_CHeTotalSize_HH (Host Channel e Total Size High-High)

7.6.128 175h H_CHeTotalSize_HL (Host Channel e Total Size High-Low)

7.6.129 176h H_CHeTotalSize_LH (Host Channel e Total Size Low-High)

7.6.130 177h H_CHeTotalSize_LL (Host Channel e Total Size Low-Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	174h	H_CHeTotalSize_HH	R / W	7: TotalSize[31]	Channel e Total Size High-High	00h
				6: TotalSize[30]		
				5: TotalSize[29]		
				4: TotalSize[28]		
				3: TotalSize[27]		
				2: TotalSize[26]		
				1: TotalSize[25]		
				0: TotalSize[24]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	175h	H_CHeTotalSize_HL	R / W	7: TotalSize[23]	Channel e Total Size High-Low	00h
				6: TotalSize[22]		
				5: TotalSize[21]		
				4: TotalSize[20]		
				3: TotalSize[19]		
				2: TotalSize[18]		
				1: TotalSize[17]		
				0: TotalSize[16]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	176h	H_CHeTotalSize_LH	R / W	7: TotalSize[15]	Channel e Total Size Low-High	00h
				6: TotalSize[14]		
				5: TotalSize[13]		
				4: TotalSize[12]		
				3: TotalSize[11]		
				2: TotalSize[10]		
				1: TotalSize[9]		
				0: TotalSize[8]		

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	177h	H_CHeTotalSize_LL	R / W	7: TotalSize[7]	Channel e Total Size Low-Low	00h
				6: TotalSize[6]		
				5: TotalSize[5]		
				4: TotalSize[4]		
				3: TotalSize[3]		
				2: TotalSize[2]		
				1: TotalSize[1]		
				0: TotalSize[0]		

These registers set the Total Size of the data to be transferred on channel CHe during USB host operation.

174h.Bit7-0, 175h.Bit7-0, 176h.Bit7-0, 177h.Bit7-0 TotalSize[31:0]

These bits set a total number of data bytes to be transferred on channel CHe (maximum 4,294,967,295 bytes: approx. 4 Gbytes).

Once a transaction is started by the TranGo bit in the H_CHeConfig_0 register, the remaining number of bytes to be transferred can be read through this register.

When bits 31–24 (H_CHeTotalSize_HH register) are read, the values of bits 23–16 (H_CHeTotalSize_HL register), bits 15–8 (H_CHeTotalSize_LH register), and bits 7–0 (H_CHeTotalSize_LL register) all are fixed. (Even when the read value is fixed, the internal counter of the LSI continues counting.)

Therefore, to read the remaining number of transfer bytes through 8-bit register accesses, read H_CHaTotalSize_HH, H_CHaTotalSize_HL, H_CHaTotalSize_LH, and H_CHaTotalSize_LL in that order.

Note also that if an OUT transaction is executed with TotalSize = 0, a zero-length packet is transmitted.

7.6.131 178h H_CHeHubAdrs (Host Channel e Hub Address)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	178h	H_CHeHubAdrs	R / W	7: HubAdrs[3]	Channel e Hub Address		00h
				6: HubAdrs[2]			
				5: HubAdrs[1]			
				4: HubAdrs[0]			
				3:	0:	1:	
			R / W	2: Port[2]	Channel e Port Number		
				1: Port[1]			
				0: Port[0]			

This register sets the hub that is connected to channel CHe during USB host operation.

Bits7-4 HubAdrs[3:0]

These bits set the USB address of the hub to which a function is connected that performs a transfer on channel CHe.

Any value in the range of 0 to 15 can be set.

Bit3 Reserved

Bits2-0 Port[2:0]

These bits set the port number of the hub to which a function is connected that performs a transfer on channel CHe.

Any value in the range of 0 to 7 can be set.

7.6.132 179h H_CHeFuncAdrs (Host Channel e Function Address)

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	179h	H_CHeFuncAdrs	R / W	7: FuncAdrs[3]	Channel e Function Address	00h
				6: FuncAdrs[2]		
				5: FuncAdrs[1]		
				4: FuncAdrs[0]		
			R / W	3: EP_Number[3]	Channel e Endpoint Number	
				2: EP_Number[2]		
				1: EP_Number[1]		
				0: EP_Number[0]		

This register sets the address of a function that performs a transfer on channel CHe during USB host operation.

Bits7-4 FuncAdrs[3:0]

These bits set the USB address of the function that includes the endpoint managed by channel CHe.

Any value in the range of 0 to 15 can be set.

Bits3-0 EP_Number[3:0]

These bits set the endpoint number that performs a transfer on channel CHe.

Any value in the range of 0 to 15 can be set.

7.6.133 17Ah H_CHeInterval_H(Host Channel e Interval High)

7.6.134 17Bh H_CHeInterval_L(Host Channel e Interval Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	17Ah	H_CHeInterval_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R / W	2: Interval[10]	Channel e Interrupt Transfer Interval High		
				1: Interval[9]			
				0: Interval[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	17Bh	H_CHeInterval_L	R / W	7: Interval[7]	Channel e Interrupt Transfer Interval Low	00h
				6: Interval[6]		
				5: Interval[5]		
				4: Interval[4]		
				3: Interval[3]		
				2: Interval[2]		
				1: Interval[1]		
				0: Interval[0]		

This register sets an interval value for the interrupt transfers to be performed on channel CHe during USB host operation.

17Ah.Bit7-3 Reserved

17Ah.Bit2-0, 17Bh.Bit7-0 Interval[10:0]

These bits set the interval (period) at which intervals tokens for interrupt transfers will be issued. The 3 low-order bits specify the interval time in microframes (125 μs), and the 7 high-order bits specify the interval time in frames (ms). Settings of this register are effective only when the H_CHeConfig1 register's TranType bits = 11 (interrupt transfer). Setting the value "0d" in this register has no effect.

The interval time set in this register is also used when transactions are retransmitted.

Interval[2:0] μFrame — Specifies an interval time in 125 μs units. Set this interval time to 1, 2, or 4 microframes. Setting any other value is prohibited. When setting these bits, make sure Interval[10:3] are all set to 0s.

Interval[10:3] Frame — Specifies an interval time in ms units. This interval time can be set to any value within the range 1 to 255 frames. When setting these bits, make sure Interval[2:0] are all set to 0s.

7.6.135 17Ch~17Dh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	17Ch -17Dh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

7.6.136 17Eh H_CHeConditionCode (Host Channel e Condition Code)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	17Eh	H_CHeConditionCode		7:	0:	1:	00h
			R	6: ConditionCode[2]	Channel e Condition Code		
			R	5: ConditionCode[1]			
			R	4: ConditionCode[0]			
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This register indicates the result of a transfer that was completed on channel CHe during USB host operation.

Bit7 Reserved

Bits6-4 ConditionCode[2:0]

These bits indicate the result of a transfer that was completed on channel CHe.

Code	Meaning	Description
000	NOERROR	The transaction is completed without error.
001	STALL	The endpoint has returned stall PID.
010	DATAOVERRUN	• The amount of data received exceeds the maximum allowed size of a data packet (MaxPacketSize). * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error. • The amount of data received exceeds the maximum allowed IRP (TotalSize) * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error. * If the received data packet is equal to or less than MaxPacketSize and the data toggle included in it does not match the expected value, this is processed as a toggle mismatch, and not as a data overrun.
011	DATAUNDERRUN	• The endpoint has returned bytes of data which are less than MaxPacketSize and are insufficient to fill the IRP (TotalSize). * If a CRC error or bit stuffing error was detected at the same time, this is processed as a retry error.

Code	Meaning	Description
100	RETRYERROR	• The device does not respond to a token within a designated time (IN) or does not transmit a handshaking signal within a designated time (OUT). • The data packet from the endpoint contains a CRC error. • The data packet from the endpoint contains a bit stuffing error. • The PID inspection bit from the endpoint failed in data PID (IN) or handshaking (OUT). • The received PID is invalid or has no PID values defined. • An ERR handshaking signal was received in a split transaction of interrupt transfer. • Three NYET handshaking signals were received consecutively in a split transaction of interrupt transfer. • The data toggle included in the data packet from the endpoint does not match the expected value (toggle mismatch).
Other	Reserved	

Bits3-0 Reserved

7.6.137 17Fh H_CHeJoin (Host Channel b Join)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	17Fh	H_CHeJoin	R / W	7: JoinIDE	0: Do nothing	1: Join CHe to IDE	00h
			R / W	6: JoinFIFO_Stat	0: Do nothing	1 Join CHe to show Status	
				5:	0:	1:	
				4:	0	1:	
			R / W	3: JoinDMA1	0: Do nothing	1: Join CHe to DMA1	
			R / W	2: JoinDMA0	0: Do nothing	1: Join CHe to DMA0	
			R / W	1: JoinCPU_Rd	0: Do nothing	1: Join CHe to CPU_Rd	
			R / W	0: JoinCPU_Wr	0: Do nothing	1: Join CHe to CPU_Wr	

This register specifies the port with which data transfers with channel CHe are to be performed during USB host operation.

Bit7 JoinIDE

Performs an IDE data transfer with the FIFO of channel CHe. The transfer direction is determined by the IDE_Control.Dir bit.

Bit6 JoinFIFO_Stat

This bit permits the Full, Empty, or NotEmpty status of the FIFO of channel CHe to be monitored with H_FIFO_IntStat.FIFO_Full, H_FIFO_IntStat.FIFO_Empty, and H_FIFO_IntStat.FIFO_NotEmpty.

Bits5-4 Reserved

Bit3 JoinDMA1

Performs a DMA1 transfer with the FIFO of channel CHe. The transfer direction is determined by the DMA1_Control.Dir bit.

Bit2 JoinDMA0

Performs a DMA0 transfer with the FIFO of channel CHe. The transfer direction is determined by the DMA0_Control.Dir bit.

Bit1 JoinCPU_Rd

Performs a read transfer of CPU register access with the FIFO of channel CHe. Namely, when a read of the FIFO_Rd_0,1 registers or FIFO_ByteRd register is performed, data is read from the FIFO of this channel.

Bit0 JoinCPU_Wr

Performs a write transfer of CPU register access with the FIFO of channel CHe. Namely, when a write to the FIFO_Wr_0,1 registers is performed, data is written to the FIFO of this channel.

If the JoinDMAx{x=0,1} bit is set, it is possible to refer to the remaining bytes of data in the FIFO when the DMA0_Control.Dir bit = 1 or the amount of free space in the FIFO when the DMA0_Control.Dir bit = 0 by inspecting the DMAx{x=0,1}_Remain_H,L registers.

If the JoinCPU_Rd or JoinCPU_Wr bit is set, it is possible to write or read data to or from the FIFO_Rd_0,1, FIFO_ByteRd, or FIFO_Wr_0,1 registers after inspecting the FIFO_RdRemain_H,L or FIFO_WrRemain_H,L registers.

The JoinDMA1, JoinDMA0, JoinCPU_Rd, and JoinCPU_Wr bits can be set to 1 only one bit at a time. If an attempt is made to set two or more of these bits by writing 1 at the same time, only the higher bit is set.

7.6.138 180h H_CH0StartAdrs_H (Host Channel 0 Start Address High)

7.6.139 181h H_CH0StartAdrs_L (Host Channel 0 Start Address Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	180h	H_CH0StartAdrs_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: StartAdrs[12]	Channel 0 Start Address High		
				3: StartAdrs[11]			
				2: StartAdrs[10]			
				1: StartAdrs[9]			
				0: StartAdrs[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	181h	H_CH0StartAdrs_L	R / W	7: StartAdrs[7]	Channel 0 Start Address Low	00h
				6: StartAdrs[6]		
				5: StartAdrs[5]		
				4: StartAdrs[4]		
				3: StartAdrs[3]		
				2: StartAdrs[2]		
				1:		
				0:		

These registers set a FIFO area that is used for channel CH0 during USB host operation.

180h.Bit7-5 Reserved

180h.Bit4-0, 181h.Bit7-2 StartAdrs[12:2]

These bits set the start address of the FIFO that is allocated to the channel CH0.

Since address values are set with the high-order bits 12–2, address specification needs to be made in 4-byte units.

The area allocated to the channel CH0 ranges from the address set by H_CH0StartAdrs to a location preceding 1 byte the address set by H_CH0EndAdrs.

After setting H_CH0StartAdrs and H_CH0EndAdrs, always be sure to set the CH0FIFO_Clr bit in the H_CHnControl register to 1 to clear the FIFO of channel CH0.

If MaxPktSize of channel CH0 is greater than the area set here, the LSI may not operate correctly.

Make sure the sum total of the FIFO areas reserved for all channels, CBW area, and CSW area does not exceed the total size of the internal RAM.

181h.Bit1-0 Reserved

7.6.140 182h H_CH0EndAdrs_H (Host Channel 0 End Address High)

7.6.141 183h H_CH0EndAdrs _L (Host Channel 0 End Address Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	182h	H_CH0EndAdrs _H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: EndAdrs[12]	Channel 0 EndAddress High		
				3: EndAdrs[11]			
				2: EndAdrs[10]			
				1: EndAdrs[9]			
				0: EndAdrs[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	183h	H_CH0EndAdrs _L	R / W	7: EndAdrs[7]	Channel 0 End Address Low	00h
				6: EndAdrs[6]		
				5: EndAdrs[5]		
				4: EndAdrs[4]		
				3: EndAdrs[3]		
				2: EndAdrs[2]		
				1:		
				0:		

These registers set a FIFO area that is used for channel CH0 during USB host operation.

182h.Bit7-5 Reserved

182h.Bit4-0, 183h.Bit7-2 EndAdrs[12:2]

These bits set the end address of the FIFO that is allocated to the channel CH0.

Since an address value is set with the high-order bits 12–2, address specification needs to be made in 4-byte units.

1A3h.Bit1-0 Reserved

7.6.142 184h H_CHaStartAdrs_H (Host Channel a Start Address High)

7.6.143 185h H_CHaStartAdrs_L (Host Channel a Start Address Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	184h	H_CHaStartAdrs_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: StartAdrs[12]	Channel a Start Address High		
				3: StartAdrs[11]			
				2: StartAdrs[10]			
				1: StartAdrs[9]			
				0: StartAdrs[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	185h	H_CHaStartAdrs_L	R / W	7: StartAdrs[7]	Channel a Start Address Low	00h
				6: StartAdrs[6]		
				5: StartAdrs[5]		
				4: StartAdrs[4]		
				3: StartAdrs[3]		
				2: StartAdrs[2]		
				1:		
				0:		

These registers set a FIFO area that is used for channel CHa during USB host operation.

184h.Bit7-5 Reserved

184h.Bit4-0, 185h.Bit7-2 StartAdrs[12:2]

These bits set the start address of the FIFO that is allocated to the channel CHa.

Since address values are set with the high-order bits 12–2, address specification needs to be made in 4-byte units.

The area allocated to the channel CHa ranges from the address set by H_CHaStartAdrs to a location preceding 1 byte the address set by H_CHaEndAdrs.

After setting H_CHaStartAdrs and H_CHaEndAdrs, always be sure to set the CHaFIFO_Clr bit in the H_CHnControl register to 1 to clear the FIFO of channel CHa.

If MaxPktSize of channel CHa is greater than the area set here, the LSI may not operate correctly.

Make sure the sum total of the FIFO areas reserved for all channels, CBW area, and CSW area does not exceed the total size of the internal RAM.

185h.Bit1-0 Reserved

7.6.144 186h H_CHaEndAdrs_H (Host Channel a End Address High)

7.6.145 187h H_CHaEndAdrs _L (Host Channel a End Address Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	186h	H_CHaEndAdrs _H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: EndAdrs[12]	Channel a EndAddress High		
				3: EndAdrs[11]			
				2: EndAdrs[10]			
				1: EndAdrs[9]			
				0: EndAdrs[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	187h	H_CHaEndAdrs _L	R / W	7: EndAdrs[7]	Channel a End Address Low	00h
				6: EndAdrs[6]		
				5: EndAdrs[5]		
				4: EndAdrs[4]		
				3: EndAdrs[3]		
				2: EndAdrs[2]		
				1:		
				0:		

These registers set a FIFO area that is used for channel CHa during USB host operation.

186h.Bit7-5 Reserved

186h.Bit4-0, 187h.Bit7-2 EndAdrs[12:2]

These bits set the end address of the FIFO that is allocated to the channel CHa.

Since an address value is set with the high-order bits 12–2, address specification needs to be made in 4-byte units.

187h.Bit1-0 Reserved

7.6.146 188h H_CHbStartAdrs_H (Host Channel b Start Address High)

7.6.147 189h H_CHbStartAdrs_L (Host Channel b Start Address Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	188h	H_CHbStartAdrs_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: StartAdrs[12]	Channel b Start Address High		
				3: StartAdrs[11]			
				2: StartAdrs[10]			
				1: StartAdrs[9]			
				0: StartAdrs[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	189h	H_CHbStartAdrs_L	R / W	7: StartAdrs[7]	Channel b Start Address Low	00h
				6: StartAdrs[6]		
				5: StartAdrs[5]		
				4: StartAdrs[4]		
				3: StartAdrs[3]		
				2: StartAdrs[2]		
				1:		
				0:		

These registers set a FIFO area that is used for channel CHb during USB host operation.

188h.Bit7-5 Reserved

188h.Bit4-0, 189h.Bit7-2 StartAdrs[12:2]

These bits set the start address of the FIFO that is allocated to the channel CHb.

Since address values are set with the high-order bits 12–2, address specification needs to be made in 4-byte units.

The area allocated to the channel CHb ranges from the address set by H_CHbStartAdrs to a location preceding 1 byte the address set by H_CHbEndAdrs.

After setting H_CHbStartAdrs and H_CHbEndAdrs, always be sure to set the CHbFIFO_Clr bit in the H_CHrFIFO_Clr register to 1 to clear the FIFO of channel CHb.

If MaxPktSiz+e of channel CHb is greater than the area set here, the LSI may not operate correctly.

Make sure the sum total of the FIFO areas reserved for all channels, CBW area, and CSW area does not exceed the total size of the internal RAM.

189h.Bit1-0 Reserved

7.6.148 18Ah H_CHbEndAdrs_H (Host Channel b End Address High)

7.6.149 18Bh H_CHbEndAdrs _L (Host Channel b End Address Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	18Ah	H_CHbEndAdrs _H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: EndAdrs[12]	Channel b EndAddress High		
				3: EndAdrs[11]			
				2: EndAdrs[10]			
				1: EndAdrs[9]			
				0: EndAdrs[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	18Bh	H_CHbEndAdrs _L	R / W	7: EndAdrs[7]	Channel b End Address Low	00h
				6: EndAdrs[6]		
				5: EndAdrs[5]		
				4: EndAdrs[4]		
				3: EndAdrs[3]		
				2: EndAdrs[2]		
				1:		
				0:		

These registers set a FIFO area that is used for channel CHb during USB host operation.

18Ah.Bit7-5 Reserved

18Ah.Bit4-0, 18Bh.Bit7-2 EndAdrs[12:2]

These bits set the end address of the FIFO that is allocated to the channel CHb.

Since an address value is set with the high-order bits 12–2, address specification needs to be made in 4-byte units.

18Bh.Bit1-0 Reserved

7.6.150 18Ch H_CHcStartAdrs_H (Host Channel c Start Address High)

7.6.151 18Dh H_CHcStartAdrs_L (Host Channel c Start Address Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	18Ch	H_CHcStartAdrs_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: StartAdrs[12]	Channel c Start Address High		
				3: StartAdrs[11]			
				2: StartAdrs[10]			
				1: StartAdrs[9]			
				0: StartAdrs[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	18Dh	H_CHcStartAdrs_L	R / W	7: StartAdrs[7]	Channel c Start Address Low	00h
				6: StartAdrs[6]		
				5: StartAdrs[5]		
				4: StartAdrs[4]		
				3: StartAdrs[3]		
				2: StartAdrs[2]		
				1:		
				0:		

These registers set a FIFO area that is used for channel CHc during USB host operation.

18Ch.Bit7-5 Reserved

18Ch.Bit4-0, 18Dh.Bit7-2 StartAdrs[12:2]

These bits set the start address of the FIFO that is allocated to the channel CHc.

Since address values are set with the high-order bits 12–2, address specification needs to be made in 64-byte units.

The area allocated to the channel CHc ranges from the address set by H_CHcStartAdrs to a location preceding 1 byte the address set by H_CHcEndAdrs.

After setting H_CHcStartAdrs and H_CHcEndAdrs, always be sure to set the CHcFIFO_Clr bit in the H_CHrFIFO_Clr register to 1 to clear the FIFO of channel CHc.

If MaxPktSize of channel CHc is greater than the area set here, the LSI may not operate correctly.

Make sure the sum total of the FIFO areas reserved for all channels, CBW area, and CSW area does not exceed the total size of the internal RAM.

18Dh.Bit1-0 Reserved

7.6.152 18Eh H_CHcEndAdrs_H (Host Channel c End Address High)

7.6.153 18Fh H_CHcEndAdrs _L (Host Channel c End Address Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	18Eh	H_CHcEndAdrs _H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: EndAdrs[12]	Channel c EndAddress High		
				3: EndAdrs[11]			
				2: EndAdrs[10]			
				1: EndAdrs[9]			
				0: EndAdrs[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	18Fh	H_CHcEndAdrs _L	R / W	7: EndAdrs[7]	Channel c End Address Low	00h
				6: EndAdrs[6]		
				5: EndAdrs[5]		
				4: EndAdrs[4]		
				3: EndAdrs[3]		
				2: EndAdrs[2]		
				1:		
				0:		

These registers set a FIFO area that is used for channel CHc during USB host operation.

18Eh.Bit7-5 Reserved

18Eh.Bit4-0, 18Fh.Bit7-2 EndAdrs[12:2]

These bits set the end address of the FIFO that is allocated to the channel CHc.

Since an address value is set with the high-order bits 12–2, address specification needs to be made in 4-byte units.

18Fh.Bit1-0Reserved

7.6.154 190h H_CHdStartAdrs_H (Host Channel d Start Address High)

7.6.155 191h H_CHdStartAdrs_L (Host Channel d Start Address Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	190h	H_CHdStartAdrs_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: StartAdrs[12]	Channel d Start Address High		
				3: StartAdrs[11]			
				2: StartAdrs[10]			
				1: StartAdrs[9]			
				0: StartAdrs[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	191h	H_CHdStartAdrs_L	R / W	7: StartAdrs[7]	Channel d Start Address Low	00h
				6: StartAdrs[6]		
				5: StartAdrs[6]		
				4: StartAdrs[5]		
				3: StartAdrs[4]		
				2: StartAdrs[3]		
				1:		
				0:		

These registers set a FIFO area that is used for channel CHd during USB host operation.

190h.Bit7-5 Reserved

190h.Bit4-0, 191h.Bit7-2 StartAdrs[12:2]

These bits set the start address of the FIFO that is allocated to the channel CHd.

Since address values are set with the high-order bits 12–2, address specification needs to be made in 4-byte units.

The area allocated to the channel CHd ranges from the address set by H_CHdStartAdrs to a location preceding 1 byte the address set by H_CHdEndAdrs.

After setting H_CHdStartAdrs and H_CHdEndAdrs, always be sure to set the CHdFIFO_Clr bit in the H_CHrFIFO_Clr register to 1 to clear the FIFO of channel CHd.

If MaxPktSize of channel CHd is greater than the area set here, the LSI may not operate correctly.

Make sure the sum total of the FIFO areas reserved for all channels, CBW area, and CSW area does not exceed the total size of the internal RAM.

191h.Bit1-0 Reserved

7.6.156 192h H_CHdEndAdrs_H (Host Channel d End Address High)

7.6.157 193h H_CHdEndAdrs _L (Host Channel d End Address Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	192h	H_CHdEndAdrs _H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: EndAdrs[12]	Channel d EndAddress High		
				3: EndAdrs[11]			
				2: EndAdrs[10]			
				1: EndAdrs[9]			
				0: EndAdrs[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	193h	H_CHdEndAdrs _L	R / W	7: EndAdrs[7]	Channel d End Address Low	00h
				6: EndAdrs[6]		
				5: EndAdrs[5]		
				4: EndAdrs[4]		
				3: EndAdrs[3]		
				2: EndAdrs[2]		
				1:		
				0:		

These registers set a FIFO area that is used for channel CHd during USB host operation.

192h.Bit7-2 Reserved

192h.Bit4-0, 193h.Bit7-2 EndAdrs[12:2]

These bits set the end address of the FIFO that is allocated to the channel CHd.

Since an address value is set with the high-order bits 12–2, address specification needs to be made in 4-byte units.

193h.Bit1-0 Reserved

7.6.158 194h H_CHeStartAdrs_H (Host Channel e Start Address High)

7.6.159 195h H_CHeStartAdrs_L (Host Channel e Start Address Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	194h	H_CHeStartAdrs_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: StartAdrs[12]	Channel e Start Address High		
				3: StartAdrs[11]			
				2: StartAdrs[10]			
				1: StartAdrs[9]			
				0: StartAdrs[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	195h	H_CHeStartAdrs_L	R / W	7: StartAdrs[7]	Channel e Start Address Low	00h
				6: StartAdrs[6]		
				5: StartAdrs[5]		
				4: StartAdrs[4]		
				3: StartAdrs[3]		
				2: StartAdrs[2]		
				1:		
				0:		

These registers set a FIFO area that is used for channel CHe during USB host operation.

194h.Bit7-5 Reserved

194h.Bit4-0, 195h.Bit7-2 StartAdrs[12:2]

These bits set the start address of the FIFO that is allocated to the channel CHe.

Since address values are set with the high-order bits 12–2, address specification needs to be made in 4-byte units.

The area allocated to the channel CHe ranges from the address set by H_CHeStartAdrs to a location preceding 1 byte the address set by H_CHeEndAdrs.

After setting H_CHeStartAdrs and H_CHeEndAdrs, always be sure to set the CHeFIFO_Clr bit in the H_CHrFIFO_Clr register to 1 to clear the FIFO of channel CHe.

If MaxPktSize of channel CHe is greater than the area set here, the LSI may not operate correctly.

Make sure the sum total of the FIFO areas reserved for all channels, CBW area, and CSW area does not exceed the total size of the internal RAM.

195h.Bit1-0 Reserved

7.6.160 196h H_CHeEndAdrs_H (Host Channel e End Address High)

7.6.161 197h H_CHeEndAdrs _L (Host Channel e End Address Low)

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	196h	H_CHeEndAdrs _H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R / W	4: EndAdrs[12]	Channel e EndAddress High		
				3: EndAdrs[11]			
				2: EndAdrs[10]			
				1: EndAdrs[9]			
				0: EndAdrs[8]			

Mode	Address	Register Name	R / W	Bit Symbol	Description	Reset
Host	197h	H_CHeEndAdrs _L	R / W	7: EndAdrs[7]	Channel e End Address Low	00h
				6: EndAdrs[6]		
				5: EndAdrs[5]		
				4: EndAdrs[4]		
				3: EndAdrs[3]		
				2: EndAdrs[2]		
				1:		
				0:		

These registers set a FIFO area that is used for channel CHe during USB host operation.

196h.Bit7-5 Reserved

196h.Bit4-0, 197h.Bit7-2 EndAdrs[12:2]

These bits set the end address of the FIFO that is allocated to the channel CHe.

Since an address value is set with the high-order bits 12–2, address specification needs to be made in 4-byte units.

197h.Bit1-0 Reserved

7.6.162 198h~1FFh Reserved ()

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	198h -1FFh	Reserved		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

Registers at addresses 0x1F5 and 0x1F6 have been added. See Appendix D for more information.

8. Electrical Characteristics

8.1 Absolute Maximum Ratings

(Vss=0V)

Parameter	Symbol	Rated Value	Unit
Supply Voltage	HVDD	VSS-0.3 - 4.0	V
	CVDD	VSS-0.3 - 4.0	V
	LVDD	VSS-0.3 - 2.5	V
Input Voltage	HVI	VSS-0.3 - HVDD+0.5	V
	CVI *1	VSS-0.3 - CVDD+0.5	V
	IVI *2	VSS-0.3 - 5.5	V
	VVI *3	VSS-0.3 - 6.0	V
	LVI *4	VSS-0.3 - LVDD+0.5	V
Output Voltage	HVO	VSS-0.3 - HVDD+0.5	V
	CVO *1	VSS-0.3 - CVDD+0.5	V
Output Current per Pin	IOUT	±10	mA
Storage Temperature	Tstg	-65 - 150	°C

*1 CPU I/F
*2 IDE I/F
*3 VBUS_B
*4 XI

8.2 Recommended Operating Conditions

Parameter	Symbol	MIN	TYP	MAX	Unit
Supply Voltage	HVDD	3.00	3.30	3.60	V
	CVDD	1.65	-	3.60	V
	LVDD	1.65	1.80	1.95	V
Input Voltage	HVI	-0.3	-	HVDD+0.3	V
	CVI *1	-0.3	-	CVDD+0.3	V
	IVI *2	-0.3	-	5.5	V
	VVI *3	-0.3		6.0	V
	LVI *4	-0.3	-	LVDD+0.3	V
Ambient Temperature	Ta	-40	25	110	°C

*1 CPU I/F
*2 IDE I/F
*3 VBUS_B
*4 XI

Engage power following the procedure below:

LVDD, PVDD (int.) --> HVDD,CVDD (I/O)

Power to the IC must be turned off in the following sequence:

HVDD, CVDD (I/O) -> LVDD (int.)

NOTE: Due to the chip's limited reliability, do not apply HVDD,CVDD continuously (for over 1 sec) while the LVDD, PVDD is disconnected.

8.3 D.C. Characteristics

Input Characteristics in the D.C. State (under Recommended Operating Conditions)

Parameter		Symbol	Condition	MIN	TYP	MAX	Unit
Power Supply Current *1							
	Supply Current	IDDH	HVDD = 3.3V(typ), 3.6V(max) *1	-	41	65	mA
		IDDCH	CVDD = 3.3V(typ), 3.6V(max) *1	-	1	4	mA
		IDDCL	CVDD = 1.8V(typ), 1.95V(max) *1	-	0.7	2	mA
		IDDL	LVDD = 1.8V(typ), 1.95V(max)	-	75	120	mA
Quiescent Current *2							
	Supply Current	IDDS	VIN = HVDD, CVDD, LVDD or VSS HVDD = 3.6V CVDD = 3.6V LVDD = 1.95V	-	-	80	μA
Input Leakage							
	Input Leakage Current	IL	HVDD = 3.6V CVDD = 3.6V LVDD = 1.95V HVIH = HVDD CVIH = CVDD LVIH = LVDD VIL = VSS	-5	-	5	μA
Input Leakage							
	Input Leakage Current (5V tolerant)	ILIF	HVDD = 3.0V CVDD = 1.65V LVDD = 1.65V HVIH = 5.5V	-10	-	10	μA

*1: Typical values are measured current values when operating 30MB/s IDE-USB transfer at Ta=2
Maximum values are approximate current values based on typical values.
*2: Quiescent current for the case in which Ta = 25°C and the bidirectional pins are set for input.

Under our test environment, the power consumption measurement value in each power management state (Ta = 25°C)

Parameter	Condition	MIN	TYP	MAX	Unit
SLEEP (At the CPU bus has activities *1) *2					
Power Comsumption	HVDD=3.3V CVDD=3.3V LVDD=1.8V	-	0.23	-	mW
SNOOZE (At the CPU bus has activities *1) *2					
Power Comsumption	HVDD=3.3V CVDD=3.3V LVDD=1.8V	-	1.8	-	mW
ACTIVE60 (Operates IDE to/from CPU transfer) *3					
Power Comsumption	HVDD=3.3V CVDD=3.3V LVDD=1.8V	-	41	-	mW
ACT_DEVICE (Operates IDE to/from USB transfer) *4					
Power Comsumption	HVDD=3.3V CVDD=3.3V LVDD=1.8V		131		mW
ACT_HOST (Operates IDE to/from USB transfer at COPY mode) *5					
Power Comsumption	HVDD=3.3V CVDD=3.3V LVDD=1.8V		134		mW
ACT_HOST (Operates IDE to/from USB transfer at At Direct COPY mode) *6					
Power Comsumption	HVDD=3.3V CVDD=3.3V LVDD=1.8V	-	273	-	mW

*1: The condition where the CPU is accessing to the memory (the SRAM and ROM etc.) that is connected on the CPU bus.

*2: At it acts as USB device, it is excepted the current consumption value (about 200uA) by VBUS at the DP pull-up resistance that S1R72V05 is containing.

*3: It is condition where is transfering data between IDE-HDD and CPU(real transfer rate 4MB/s).

*4: It connects to the PC as USB device, and it is condition where is transfering data between IDE-HDD and PC(real transfer rate 25MB/s).

*5: It connects to the USB-HDD as USB host, and it is condition where is transfering data between IDE-HDD and USB-HDD(real transfer rate 5.3MB/s).

*6: It connects to the USB-HDD as USB host, and it is condition where is transfering data between IDE-HDD and USB-HDD(real transfer rate 30MB/s).

Input Characteristics in the D.C. State (under Recommended Operating Conditions) Cont'd from the preceding page

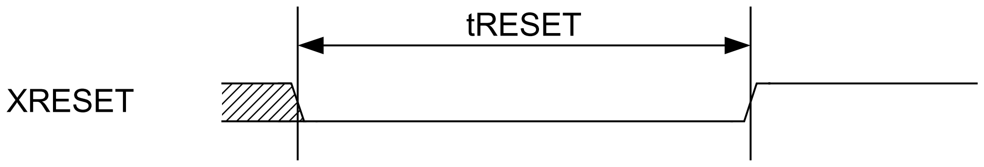
Parameter	Symbol	Condition	MIN	TYP	MAX	Unit
Input Characteristics (LVCMOS)	Pin name:	TEST, TDI, TCK ,TRST, TMS				
High Level Input Voltage	VIH1	HVDD = 3.6V	2.2	-	-	V
Low Level Input Voltage	VIL1	HVDD = 3.0V	-	-	0.8	V
Input Characteristics (LVCMOS)	Pin name:	CA[8:1], CD[15:0], XCS, XRD, XWRL, XWRH, XBEL, XDACK0, XDACK1, XRESET				
High Level Input Voltage	VIH2	CVDD = 3.6V	2.2	-	-	V
Low Level Input Voltage	VIL2	CVDD = 3.0V	-	-	0.8	V
High Level Input Voltage	VIH3	CVDD = 1.95V	1.27	-	-	V
Low Level Input Voltage	VIL3	CVDD = 1.65V	-	-	0.57	V
Input Characteristics (LVTTL)	Pin name:	HDD[15:0], HDMARQ, HIORDY, HINTRQ, XHDASP, XHPDIAG				
High Level Input Voltage	VIH4	HVDD = 3.6V	2.2	-	-	V
Low Level Input Voltage	VIL4	HVDD = 3.0V	-	-	0.8	V
Schmitt Input Characteristics	Pin name:	VBUSFLG_A				
High Level Trigger Voltage	VT+	HVDD = 3.6V	1.4	-	2.7	V
Low Level Trigger Voltage	VT-	HVDD = 3.0V	0.6	-	1.8	V
Hysteresis Voltage	ΔV	HVDD = 3.0V	0.3	-	-	V
Schmitt Input Characteristics (USB:FS)	Pin name:	DP_A, DM_A, DP_B, DM_B				
High Level Trigger Voltage	VT+(USB)	HVDD = 3.6V	1.1	-	1.8	V
Low Level Trigger Voltage	VT-(USB)	HVDD = 3.0V	1.0	-	1.5	V
Hysteresis Voltage	ΔV(USB)	HVDD = 3.0V	0.1	-	-	V
Input Characteristics (USB:FS Differential Input)	Pin name:	DP_A and DM_A, DP_B and DM_B in pairs				
Sensitivity of Differential Input	VDS(USB)	HVDD = 3.0V Differential input voltage 0.8V ～ 2.5V	-	-	0.2	V
Input Characteristics (VBUS)	Pin name:	VBUS_B				
High Level Trigger Voltage	VT+(VBUS)	HVDD = 3.6V	1.86	-	2.85	V
Low Level Trigger Voltage	VT-(VBUS)	HVDD = 3.0V	1.48	-	2.23	V
Hysteresis Voltage	ΔV(VBUS)	HVDD = 3.0V	0.31	-	0.64	V
Pulldown Resistance	RPLDV	VI=5.0V	110	125	150	kΩ
Input Characteristics	Pin name:	HDD[15:8], HDD[6:0], HIORDY, XHDASP, XHPDIAG, VBUSFLG_A				
Pullup Resistance	RPLU	VI = VSS	50	100	240	kΩ
Input Characteristics	Pin name:	HDD7, HDMARQ, HINTRQ				
Pulldown Resistance	RPLD	VI = HVDD	50	100	240	kΩ

Parameter	Symbol	Condition	MIN	TYP	MAX	Unit
Input Characteristics	Pin name:	CD[15:0], XDREQ0, XDREQ1, XINT				
High Level Output Voltage	VOH1	CVDD = 3.0V IOH = -2mA	CVDD-0.4	-	-	V
Low Level Output Voltage	VOL1	CVDD = 3.0V IOL = 2mA	-	-	VSS+0.4	V
High Level Output Voltage	VOH2	CVDD = 1.65V IOH = -1mA	CVDD-0.4	-	-	V
Low Level Output Voltage	VOL2	CVDD = 1.65V IOL = 1mA	-	-	VSS+0.4	V
Output Characteristics	Pin name:	HDD[15:0], HDA[2:0], XHCS1, XHCS0, XHIOR, XHIOW, XHDMACK, XHRESET				
High Level Output Voltage	VOH3	HVDD = 3.0V IOH = -4mA	HVDD-1.0	-	-	V
Low Level Output Voltage	VOL3	HVDD = 3.0V IOL = 4mA	-	-	VSS+0.4	V
Output Characteristics	Pin name:	TDO, VBUSEN_A				
High Level Output Voltage	VOH4	HVDD = 3.0V IOH = -2mA	HVDD-0.4	-	-	V
Low Level Output Voltage	VOL4	HVDD = 3.0V IOL = 2mA	-	-	VSS+0.4	V
Output Characteristics (USB:FS)	Pin name:	DP_A, DM_A, DP_B, DM_B				
High Level Output Voltage	VOH(USB)	HVDD = 3.0V	2.8	-	-	V
Low Level Output Voltage	VOL(USB)	HVDD = 3.6V	-	-	0.3	V
Output Characteristics (USB:HS)	Pin name:	DP_A, DM_A, DP_B, DM_B				
High Level Input Voltage	VHSOH (USB)	HVDD = 3.0V	360	-	-	mV
Low Level Output Voltage	VHSOL (USB)	HVDD = 3.6V	-	-	10.0	mV
Output Characteristics	Pin name:	CD[15:0], XINT				
OFF-State Leakage Current	IOZ	CVDD = 3.6 LVDD = 1.95V CVOH = CVDD LVOH = LVDD VOL = VSS	-5	-	5	μA
Output Characteristics	Pin name:	HDD[15:0], HDA[2:0], XHCS1, XHCS0, XHIOR, XHIOW, XHDMACK, XHRESET				
OFF-State Leakage Current (5V tolerant)	IOZHF	HVDD = 3.0 HVOH = 5.5V	-10	-	10	μA

Parameter	Symbol	Condition	MIN	TYP	MAX	Unit
Pin Capacitance	Pin name:	All input pins				
Input Pin Capacitance	CI	f = 10MHz HVDD=CVDD=LVDD=VSS	-	-	10	pF
Pin Capacitance	Pin name:	All input pins				
Output Pin Capacitance	CO	f = 10MHz HVDD=CVDD=LVDD=VSS	-	-	10	pF
Pin Capacitance	Pin name:	All input/output pins (not including DP_A, DM_A, DP_B, and DM_B)				
Input/Output Pin Capacitance 1	CIO1	f = 10MHz HVDD=CVDD=LVDD=VSS	-	-	10	pF
Pin Capacitance	Pin name:	DP_A, DM_A, DP_B, DM_B				
Input/Output Pin Capacitance 2	CIO2	f = 10MHz HVDD=CVDD=LVDD=VSS	-	-	11	pF

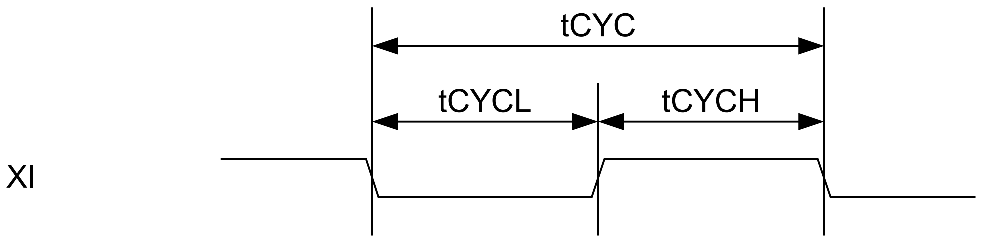
8.4 A.C. Characteristics

8.4.1 RESET Timing



Symbol	Description	min	typ	max	Unit
tRESET	Reset pulse width	40	-	-	ns

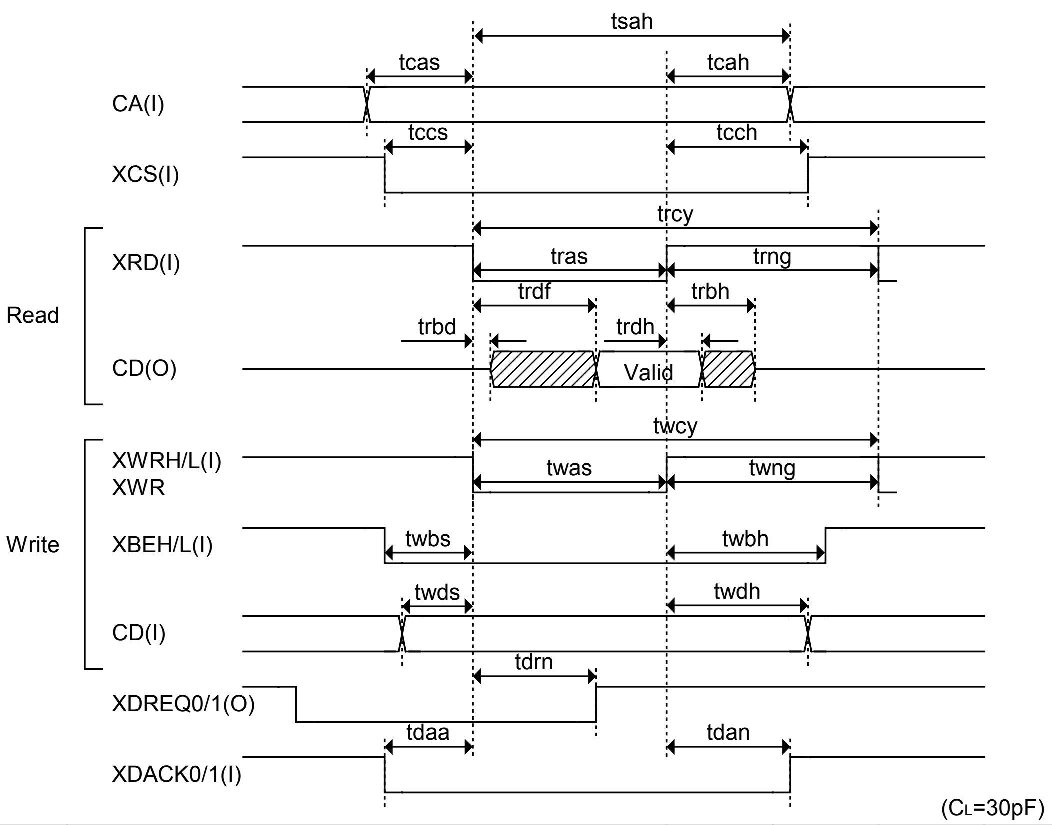
8.4.2. Clock Timing



Symbol	Description	min	typ	max	Unit
tCYC	Clock cycle(ClkSelect=0)	11.9988	12	12.0012	MHz
tCYC	Clock cycle(ClkSelect=1)	23.9976	24	24.0024	MHz
tCYCH tCYCL	Clock duty	45	-	55	%

8.4.3 CPU and DMA I/F Access Timing

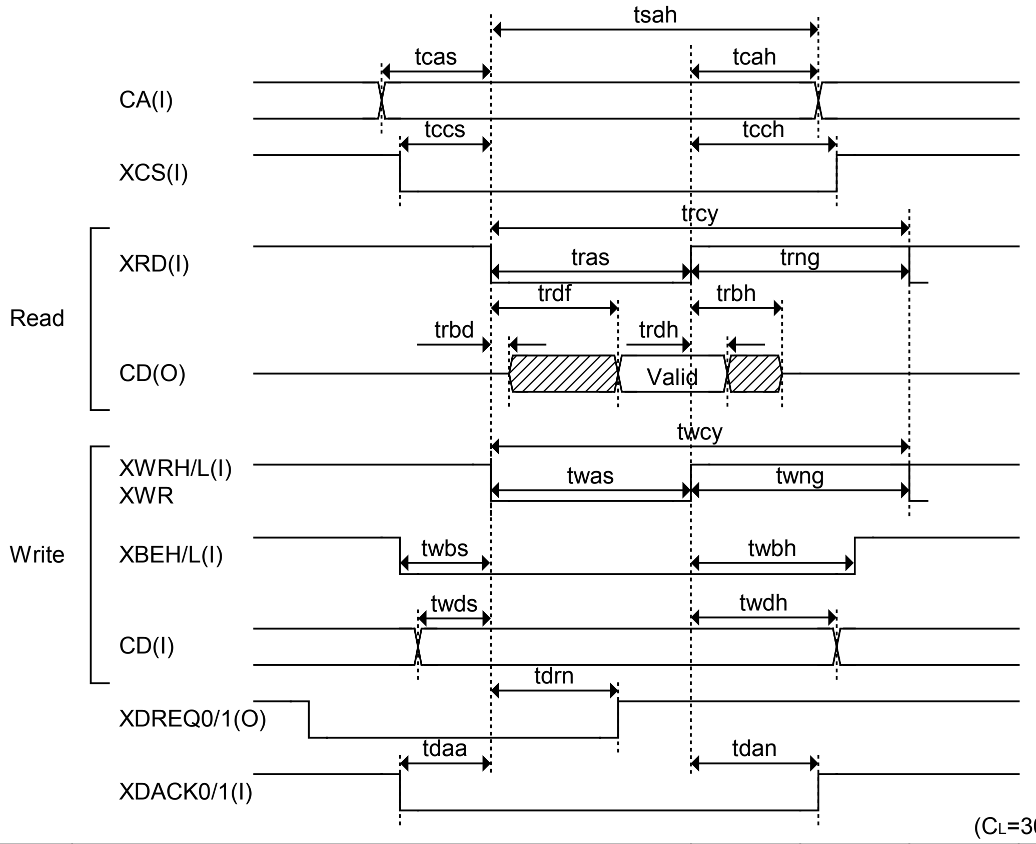
8.4.3.1 For the condition that CVDD(typ)=1.8V-3.3V range



(CL=30pF)

Symbol	Description	min	typ	max	unit
tcas	Address setup time	6	-	-	ns
tcah	Adress hold time from nagation of strobe	6	-	-	ns
tsah	Adress hold time from assertion of strobe	55	-	-	ns
tccs	XCS setup time	6	-	-	ns
tcch	XCS hold time	6	-	-	ns
trcy	Read cycle	80	-	-	ns
tras	Read strobe assert time	40	-	-	ns
trng	Read strobe negate time	25	-	-	ns
trbd	Read data output start time	1	-	-	ns
trdf	Read data valid time	-	-	35	ns
trdh	Read data hold time	3	-	-	ns
trbh	Read data output delay time	-	-	10	ns
twcy	Write cycle	80	-	-	ns
twas	Write strobe assert time	40	-	-	ns
twng	Write strobe negate time	25	-	-	ns
twbs	Write byte enable setup time	6	-	-	ns
twbh	Write byte enable hold time	6	-	-	ns
twds	Write data setup time	0	-	-	ns
twdh	Write data hold time	0	-	-	ns
tdrn	XDREQ0/1 negate delay time	-	-	35	ns
tdaa	XDACK0/1 setup time	6	-	-	ns
tdan	XDACK0/1 hold time	6	-	-	ns

8.4.3.2 For the condition that CVDD(typ) is limited to 1.8V(alleviate some listrictions)

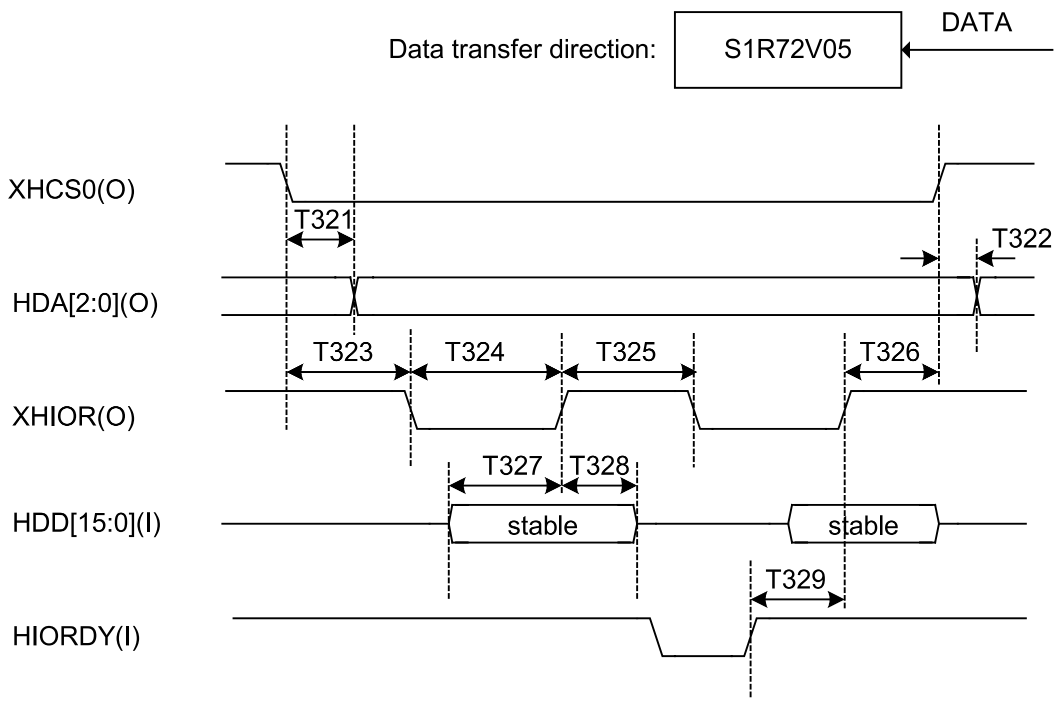


(CL=30pF)

Symbol	Description	min	typ	max	unit
tcas	Address setup time	6	-	-	ns
tcah	Adress hold time from nagation of strobe	6	-	-	ns
tsah	Adress hold time from assertion of strobe	55	-	-	ns
tccs	XCS setup time	6	-	-	ns
tcch	XCS hold time	6	-	-	ns
trcy	Read cycle	75	-	-	ns
tras	Read strobe assert time	37	-	-	ns
trng	Read strobe negate time	25	-	-	ns
trbd	Read data output start time	1	-	-	ns
trdf	Read data valid time	-	-	30	ns
trdh	Read data hold time	3	-	-	ns
trbh	Read data output delay time	-	-	10	ns
twcy	Write cycle	75	-	-	ns
twas	Write strobe assert time	37	-	-	ns
twng	Write strobe negate time	25	-	-	ns
twbs	Write byte enable setup time	6	-	-	ns
twbh	Write byte enable hold time	6	-	-	ns
twds	Write data setup time	0	-	-	ns
twdh	Write data hold time	0	-	-	ns
tdrn	XDREQ0/1 negate delay time	-	-	30	ns
tdaa	XDACK0/1 setup time	6	-	-	ns
tdan	XDACK0/1 hold time	6	-	-	ns

8.4.4 IDE I/F Timing

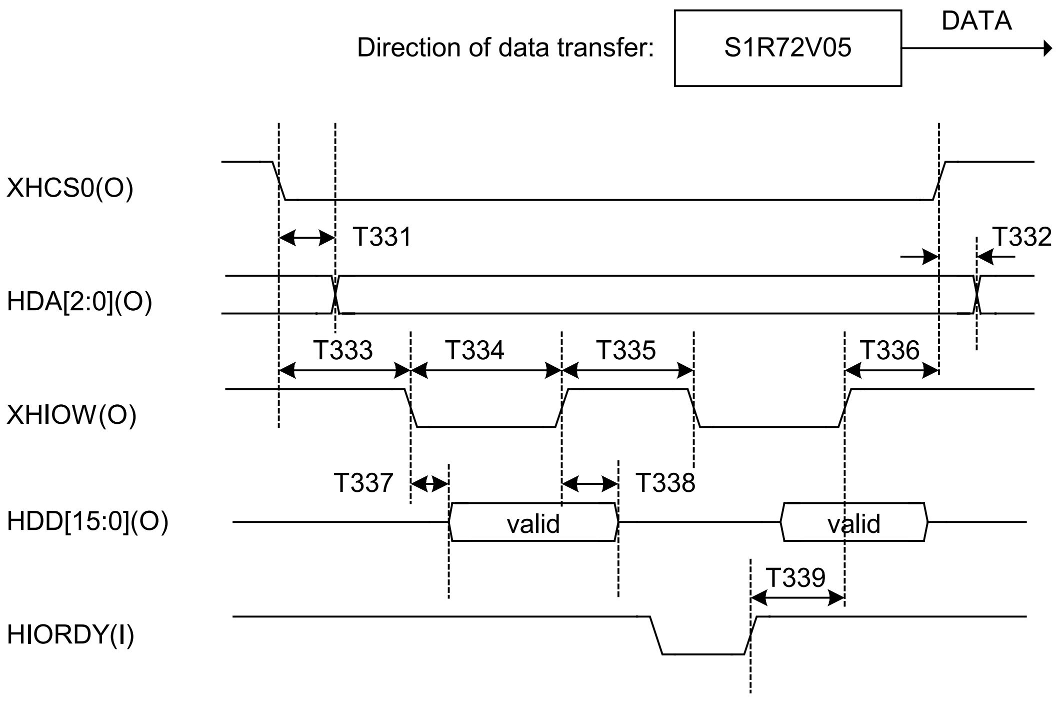
8.4.4.1 PIO Read Timing



Symbol	Description	min	typ	max	Unit
T321	XHCS0↓ → HDA HDA output delay time	-	0	-	ns
T322	XHCS0↑ → HDA HDA hold time	-	0	-	ns
T323	XHCS0↓ → XHIOR↓ XHCS0 setup time	80	-	-	ns
T324	XHIOR↓ → XHIOR↑ XHIOR assert pulse width	-	(AP+4) * 16.7 - 3	-	ns
T325	XHIOR↑ → XHIOR↓ XHIOR negate pulse width	-	(NP+4) * 16.7 + 3	-	ns
T326	XHIOR↑ → XHCS0↑ XHCS0 hold time	50	-	-	ns
T327	HDD → XHIOR↑ Data setup time	10	-	-	ns
T328	XHIOR↑ → HDD Data hold time	0	-	-	ns
T329	HIORDY assert → XHIOR↑ XHIOR output delay time	-	-	25	ns

*1: For details about AP=IDE_Tmod.AssertPulseWidth and NP=IDE_Tmod.NegatePulseWidth, refer to "IDE Transfer Mode" in the register description.

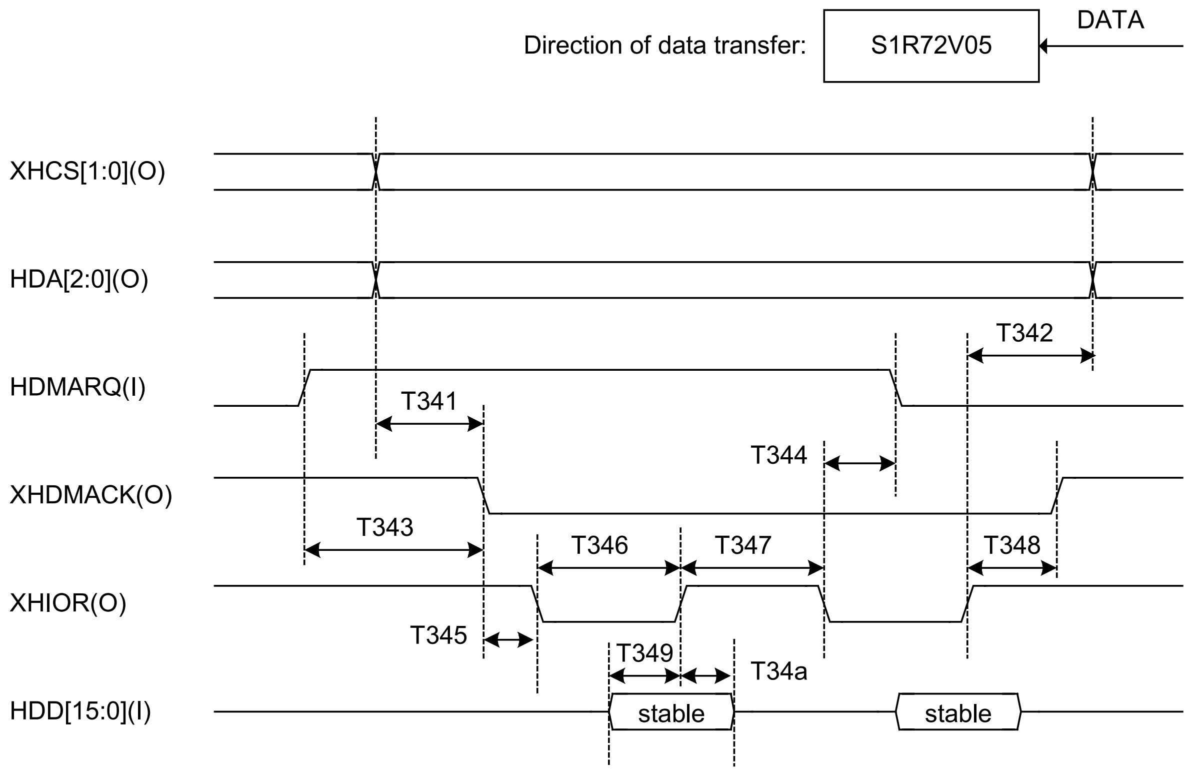
8.4.4.2 PIO Write Timing



Symbol	Description	min	typ	max	Unit
T331	XHCS0 ↓ → HDA HDA output delay time	-	0	-	ns
T332	XHCS0 ↑ → HDA HDA hold time	-	0	-	ns
T333	XHCS0 ↓ → XHIOW ↓ XHCS0 setup time	80	-	-	ns
T334	XHIOW ↓ → XHIOW ↑ XHIOW assert pulse width	-	(AP+4) * 16.7 - 3	-	ns
T335	XHIOW ↑ → XHIOW ↓ XHIOW negate pulse width	-	(NP+4) * 16.7 + 3	-	ns
T336	XHIOW ↑ → XHCS0 ↑ XHCS0 hold time	50	-	-	ns
T337	XHIOW ↓ → HDD Data output delay time	0	-	10	ns
T338	XHIOW ↑ → HDD Data bus negate time	33	-	45	ns
T339	HIORDY assert → XHIOW ↑ XHIOW output delay time	-	-	25	ns

*1: For details about AP=IDE_Tmod.AssertPulseWidth and NP=IDE_Tmod.NegatePulseWidth, refer to "IDE Transfer Mode" in the register description.

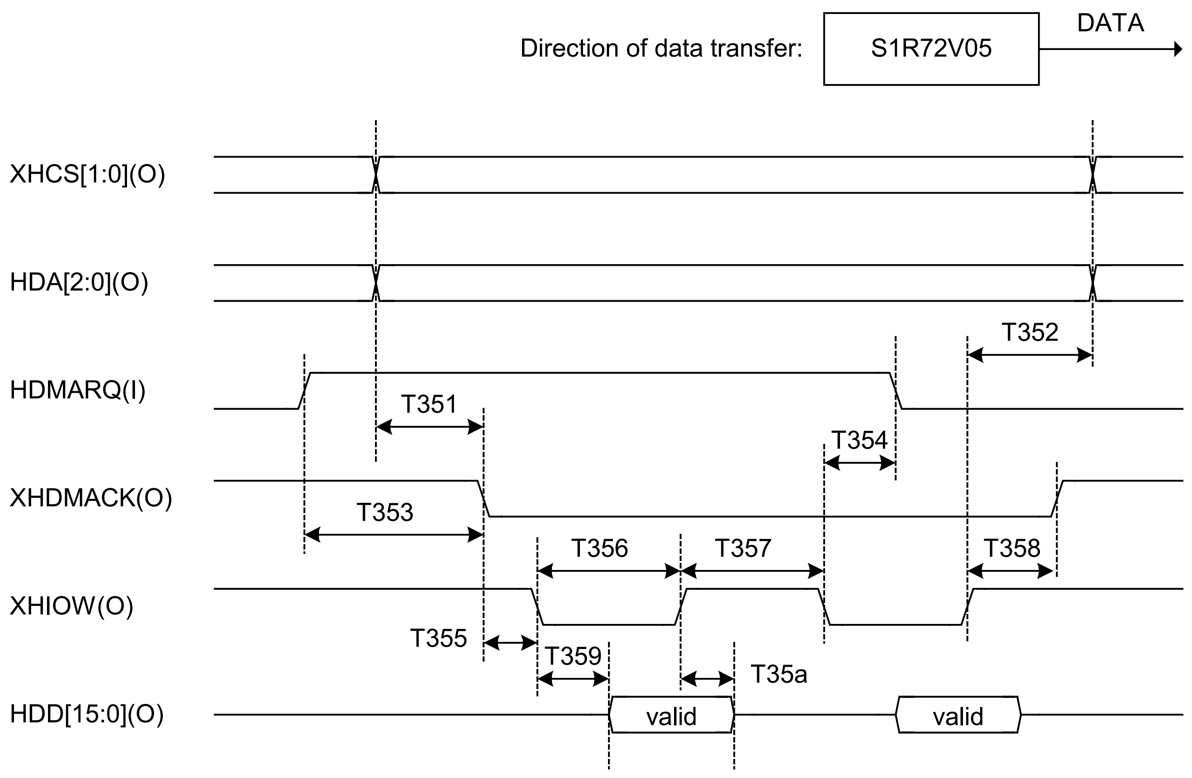
8.4.4.3 DMA Read Timing



Symbol	Description	min	typ	max	Unit
T341	XHCS ↑ , HDA → XHDMACK ↓ Address setup time	70	-	-	ns
T342	XHIOR ↑ → XHCS ↑ , HDA Address hold time	50	-	-	ns
T343	HDMARQ ↑ → XHDMACK ↓ XHDMACK response time	17	-	-	ns
T344	XHIOR ↓ → HDMARQ negate HDMARQ hold time	0	-	-	ns
T345	XHDMACK ↓ → XHIOR ↓ XHDMACK setup time	0	-	-	ns
T346	XHIOR ↓ → XHIOR ↑ XHIOR assert pulse width	-	(AP+4) * 16.7 - 3	-	ns
T347	XHIOR ↑ → XHIOR ↓ XHIOR negate pulse width	-	(NP+4) * 16.7 + 3	-	ns
T348	XHIOR ↑ → XHDMACK ↑ XHDMACK hold time	30	-	90	ns
T349	HDD → XHIOR ↑ Data setup time	10	-	-	ns
T34a	XHIOR ↑ → HDD Data bus hold time	0	-	-	ns

*1: For details about AP=IDE_Tmod.AssertPulseWidth and NP=IDE_Tmod.NegatePulseWidth, refer to "IDE Transfer Mode" in the register description.

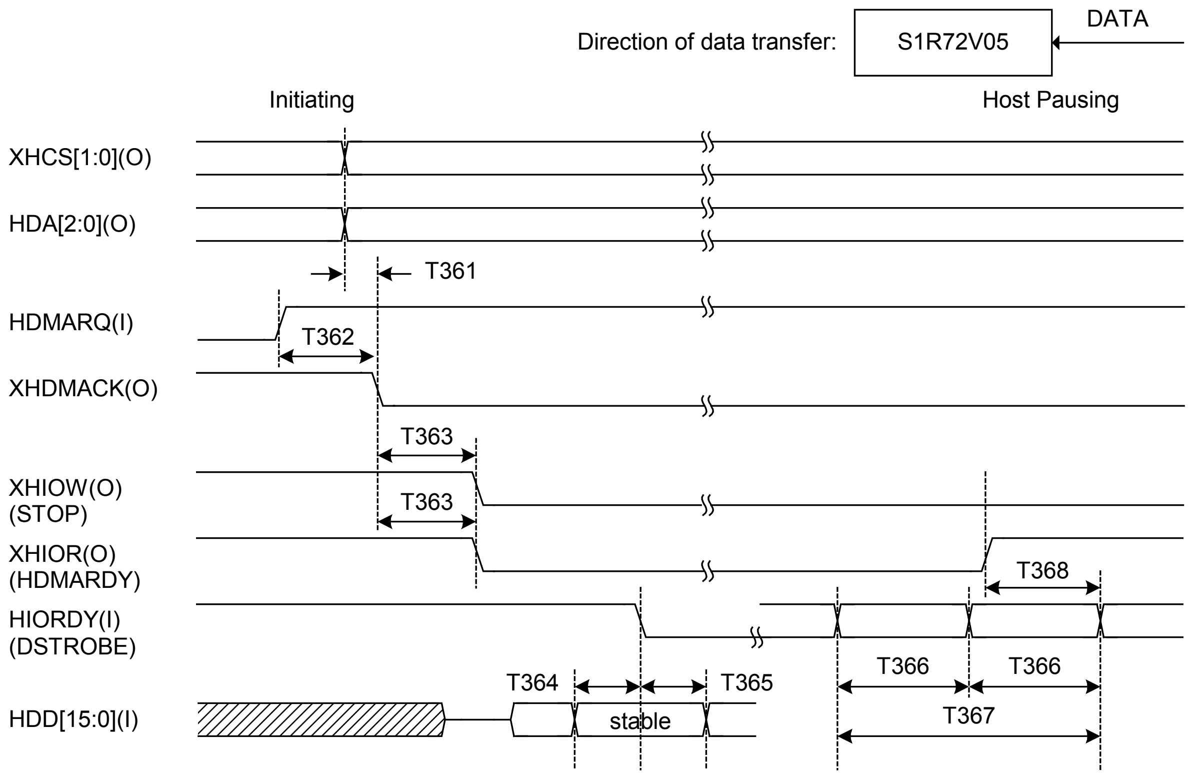
8.4.4.4 DMA Write Timing



Symbol	Description	min	typ	max	Unit
T351	XHCS ↑ , HDA → XHDMACK ↓ Address setup time	70	-	-	ns
T352	XHIOW ↑ → XHCS ↑ , HDA Address hold time	50	-	-	ns
T353	HDMARQ ↑ → XHDMACK ↓ XHDMACK response time	17	-	-	ns
T354	XHIOW ↓ → HDMARQ negate HDMARQ hold time	0	-	-	ns
T355	XHDMACK ↓ → XHIOW ↓ XHDMACK setup time	0	-	-	ns
T356	XHIOW ↓ → XHIOW ↑ XHIOW assert pulse width	-	(AP+4) * 16.7 - 3	-	ns
T357	XHIOW ↑ → XHIOW ↓ XHIOW negate pulse width	-	(NP+4) * 16.7 + 3	-	ns
T358	XHIOW ↑ → XHDMACK ↑ XHDMACK hold time	30	-	90	ns
T359	XHIOW ↓ → HDD Data output delay time	0	-	10	ns
T35a	XHIOW ↑ → HDD Data bus negate time	33	-	45	ns

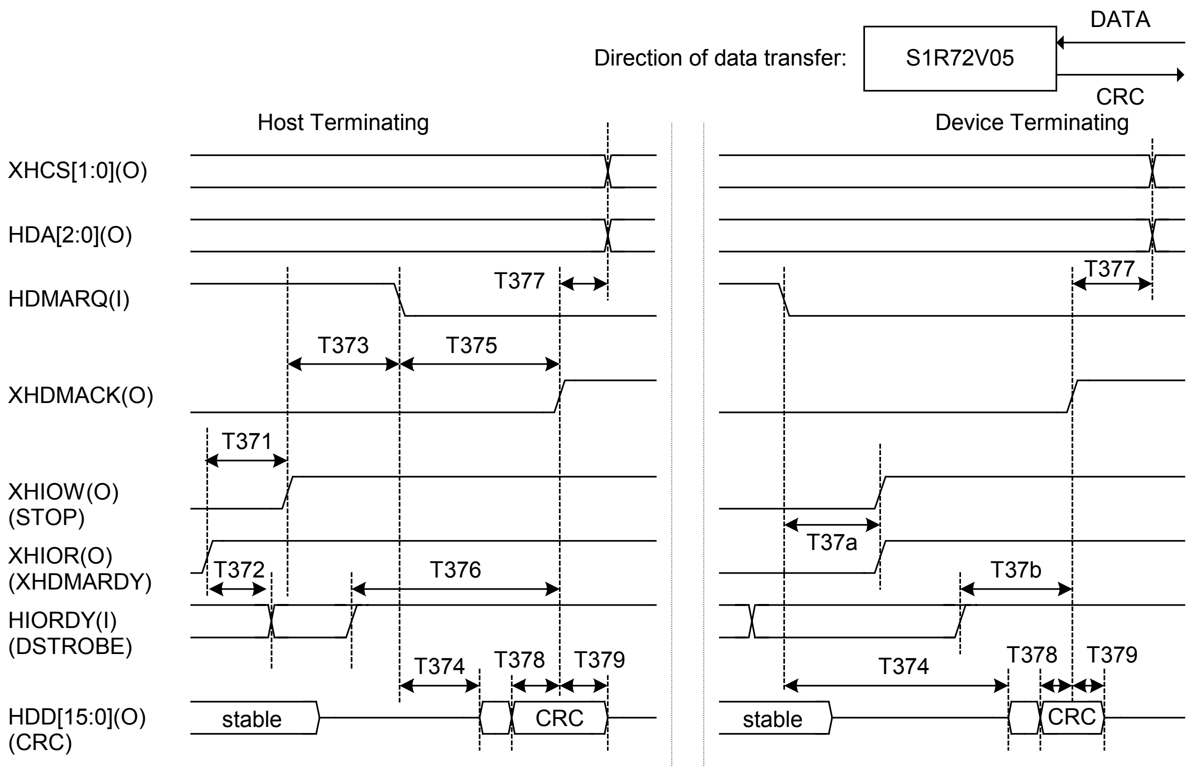
*1: For details about AP=IDE_Tmod.AssertPulseWidth and NP=IDE_Tmod.NegatePulseWidth, refer to "IDE Transfer Mode" in the register description.

8.4.4.5 Ultra DMA Read Timing



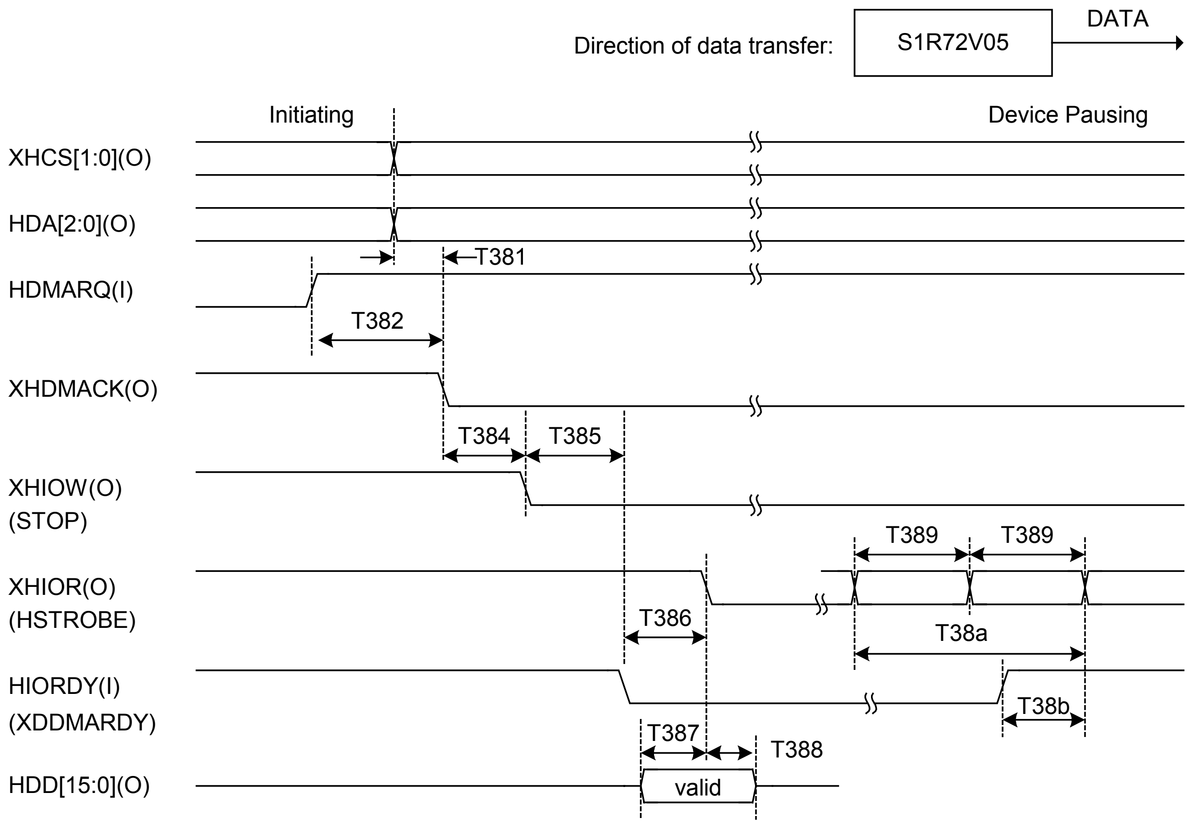
Symbol	Description	min	typ	max	Unit
T361	XHCS ↑ , HDA → XHDMACK ↓ Address setup time	80	-	-	ns
T362	HDMARQ ↑ → XHDMACK ↓ XHDMACK response time	65	-	-	ns
T363	XHDMACK ↓ → XHIOR(W) ↓ Envelope time	28	-	40	ns
T364	HDD → HIORDY Data setup time	4	-	-	ns
T365	HIORDY → HDD Data hold time	4	-	-	ns
T366	HIORDY → HIORDY HIORDY cycle time	15	-	-	ns
T367	HIORDY → HIORDY HIORDY cycle time x 2	30	-	-	ns
T368	XHIOR ↑ → HIORDY Last STROBE time	-	-	IDE standard t_{RFS}	ns

Ultra DMA Read Timing (Cont'd from the preceding page)



Symbol	Description	min	typ	max	Unit
T371	XHIOR↑ → XHIOW↑ Time until STOP asserted	180	-	-	ns
T372	XHIOR↑ → HIORDY Last STROBE time	-	-	IDE standard t_{RFS}	ns
T373	XHIOW↑ → HDMARQ↓ Limited interlock time	-	-	IDE standard t_{LI}	ns
T374	HDMARQ↓ → HDD Output delay time	70	-	-	ns
T375	HDMARQ↓ → XHDMACK↑ Minimum interlock time	160	-	-	ns
T376	HIORDY → XHDMACK↑ Minimum interlock time	110	-	-	ns
T377	XHDMACK↑ → XHCS0,1 XHCS0,1 hold time	35	-	-	ns
T378	HDD(CRC) → XHDMACK↑ CRC data setup time	75	-	-	ns
T379	XHDMACK↑ → HDD(CRC) CRC data hold time	12	-	-	ns
T37a	HDMARQ↓ → XHIOR↑ Limited interlock time	20	-	38	ns
T37b	HIORDY → XHDMACK↑ Minimum interlock time	110	-	-	ns

8.4.4.6 Ultra DMA Write Timing



Symbol	Description	min	typ	max	Unit
T381	XHCS↑, HDA → XHDMACK↓ Address setup time	80	-	-	ns
T382	HDMARQ↑ → XHDMACK↓ XHDMACK response time	65	-	-	ns
T384	XHDMACK↓ → XHIOW↓ Envelope time	28	-	40	ns
T385	XHIOW↓ → HIORDY↓ Limited interlock time	IDE standard t_{LI}	-	IDE standard t_{LI}	ns
T386	HIORDY↓ → XHIOR↓ Non-limited interlock time	20	-	-	ns
T387	HDD → XHIOR↓ Data setup time	-	(cyc+1) * 16.7	-	ns
T388	XHIOR↓ → HDD Data hold time	-	(cyc+1) * 16.7	-	ns
T389	XHIOR → XHIOR XHIOR cycle time	-	(cyc+2) * 16.7	-	ns
T38a	XHIOR → XHIOR XHIOR cycle time x 2	-	T389 * 2	-	ns
T38b	HIORDY↑ → XHIOR Last STROBE time	20	-	38	ns

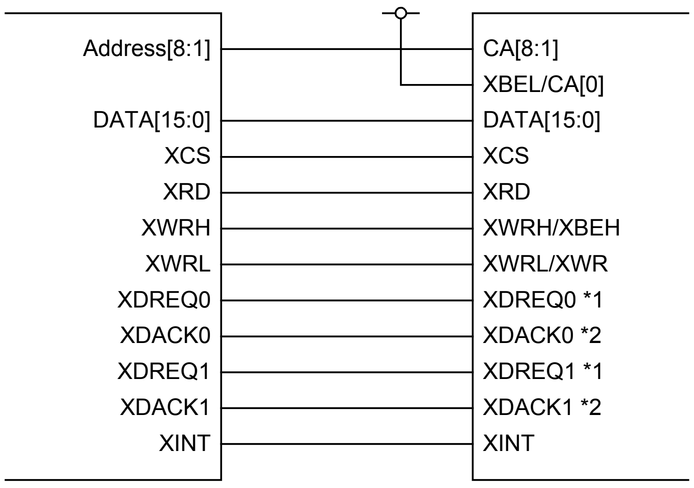
*1:cyc=UltraDMAcycle
For details, refer to "IDE Ultra-DMA Transfer Mode" in the register description.

8.4.5 USB I/F Timing

The USB I/F timing conforms to the USB 2.0 Standard.

9. Connection Examples

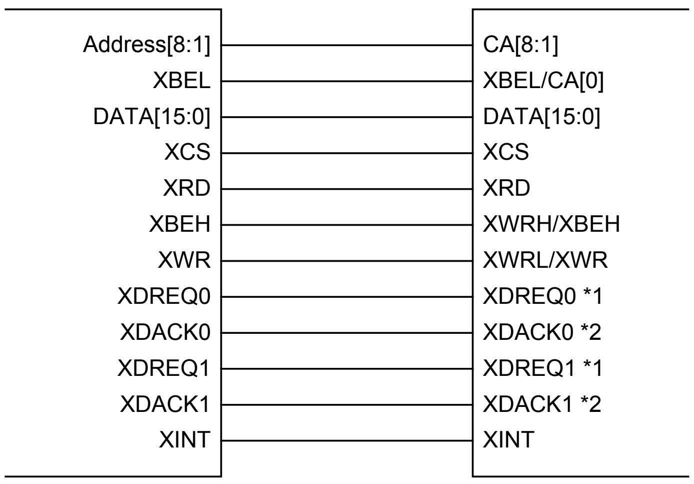
9.1 CPU I/F Connection Example



*1: Leave these pins open when not using DMA.

*2: Fix these pins either high or low when not using DMA.

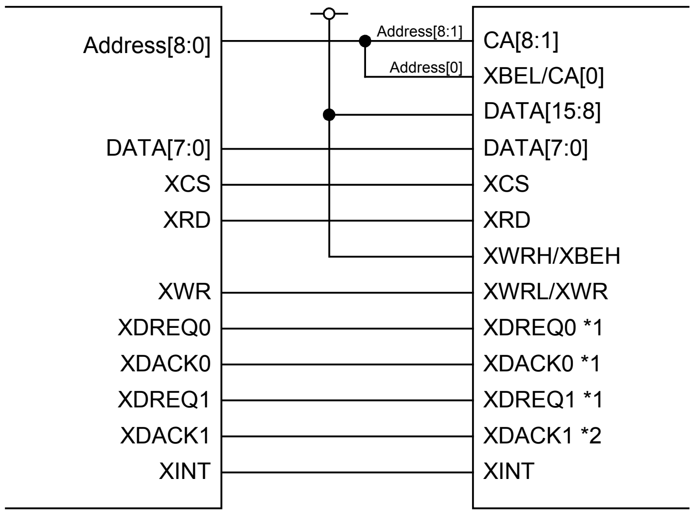
Connection example for a 16bit CPU(XWRH/XWRL)



*1: Leave these pins open when not using DMA.

*2: Fix these pins either high or low when not using DMA.

Connection example for a 16bit CPU(XBEH/XBEL)



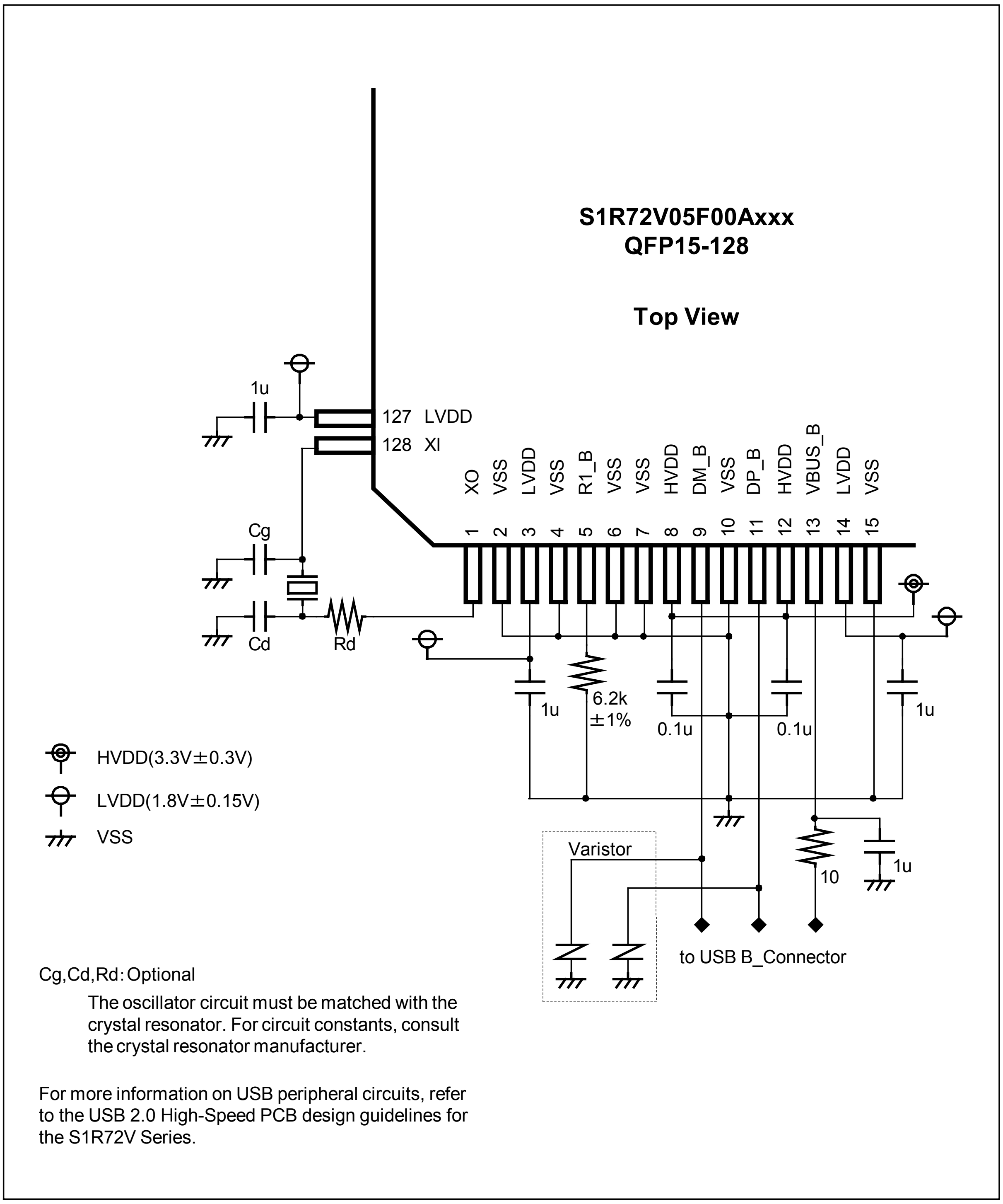
*1 : Leave these pins open when not using DMA.

*2 : Fix these pins either high or low when not using DMA.

Connection example for an 8-bit CPU

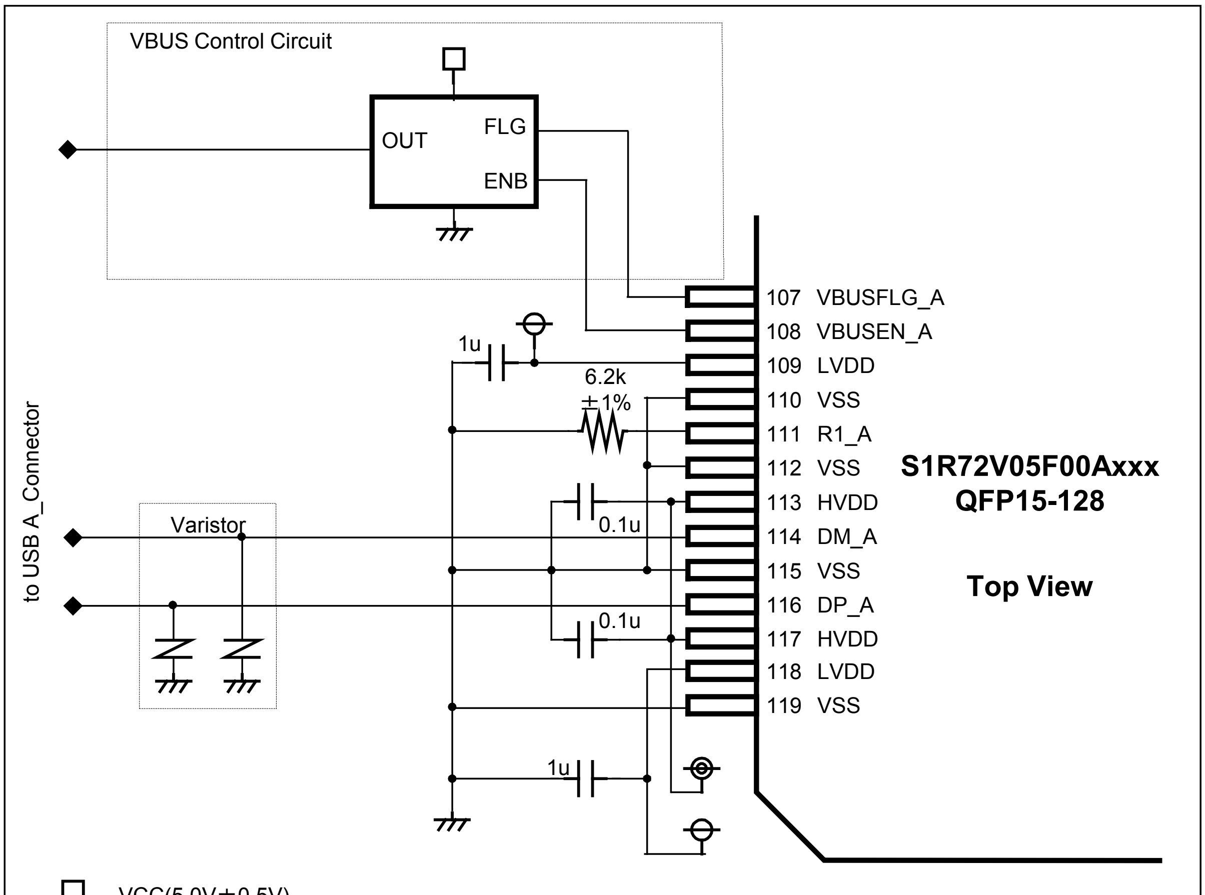
9.2 USB I/F Connection Example

9.2.1 For QFP15-128 (USB device peripheral)


S1R72V05F00Axxx
QFP15-128
Top View
127 LVDD
128 XI
1u
Cg
Cd
Rd
1 XO
2 VSS
3 LVDD
4 VSS
5 R1_B
6 VSS
7 VSS
8 HVDD
9 DM_B
10 VSS
11 DP_B
12 HVDD
13 VBUS_B
14 LVDD
15 VSS
1u
6.2k ±1%
0.1u
0.1u
1u
HVDD(3.3V±0.3V)
LVDD(1.8V±0.15V)
VSS
Varistor
10
1u
to USB B_Connector
Cg,Cd,Rd: Optional
The oscillator circuit must be matched with the crystal resonator. For circuit constants, consult the crystal resonator manufacturer.
For more information on USB peripheral circuits, refer to the USB 2.0 High-Speed PCB design guidelines for the S1R72V Series.

Take caution in selecting the power element device because its performance affects the quality of the USB signal waveform.

9.2.2 For QFP15-128 (USB host peripheral)



□ VCC(5.0V±0.5V)

HVDD(3.3V±0.3V)

LVDD(1.8V±0.15V)

VSS

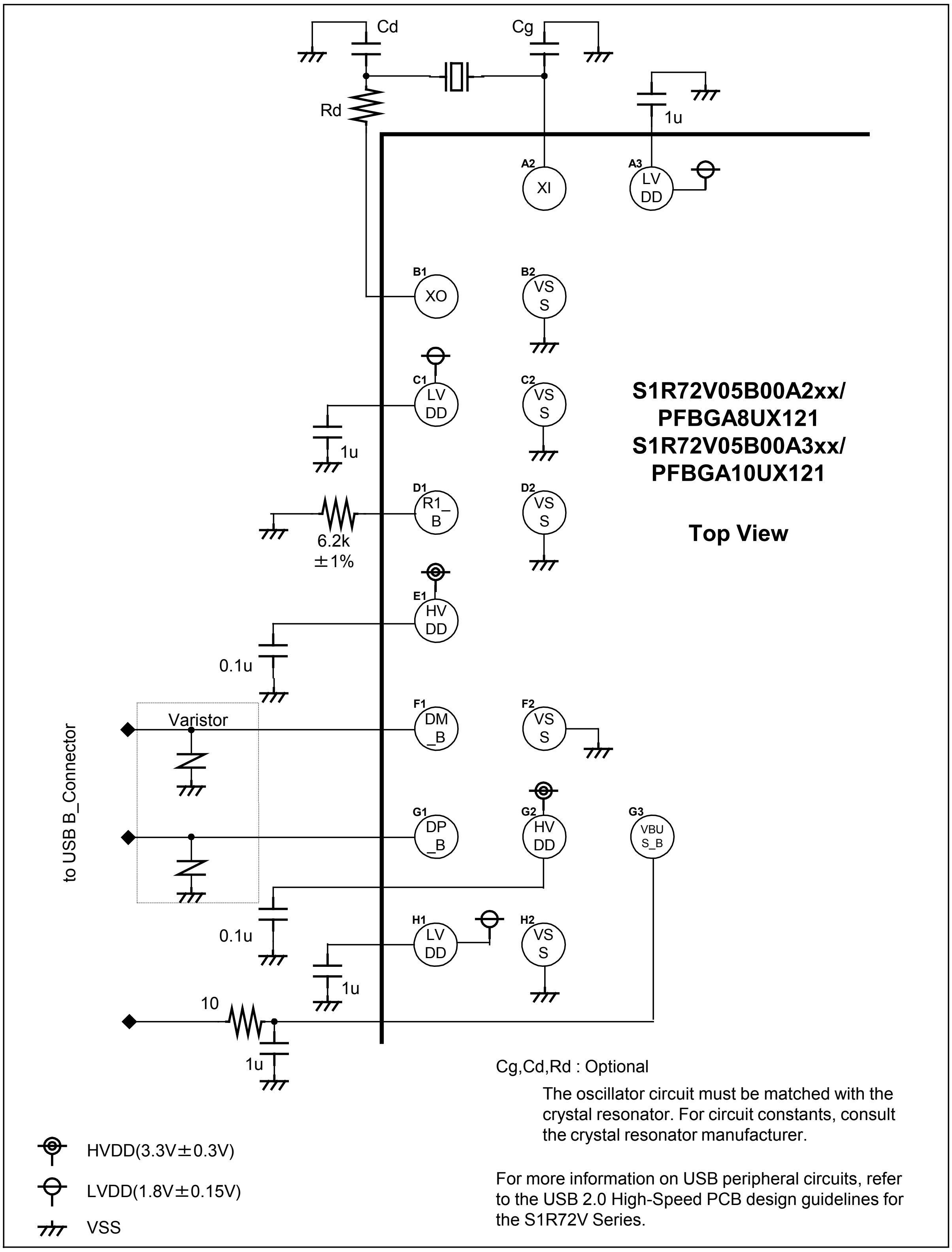
For more information on USB peripheral circuits, refer to the USB 2.0 High-Speed PCB design guidelines for the S1R72V Series.

Leave VBUSFLG_A and VBUSEN_A pin open if VBUS control circuit is not implemented.

The VBUS control circuit is shown solely for reference purposes. It does not represent a recommended circuit configuration. Choose the components and circuit design suitable for your system.
Be careful when using a component incorporating an FET switch. If the OUT pin voltage in this component exceeds the IN pin voltage, the parasitic diode between the source and drain may cause current to flow from the OUT pin into the IN pin, whether the component is enabled or disabled.

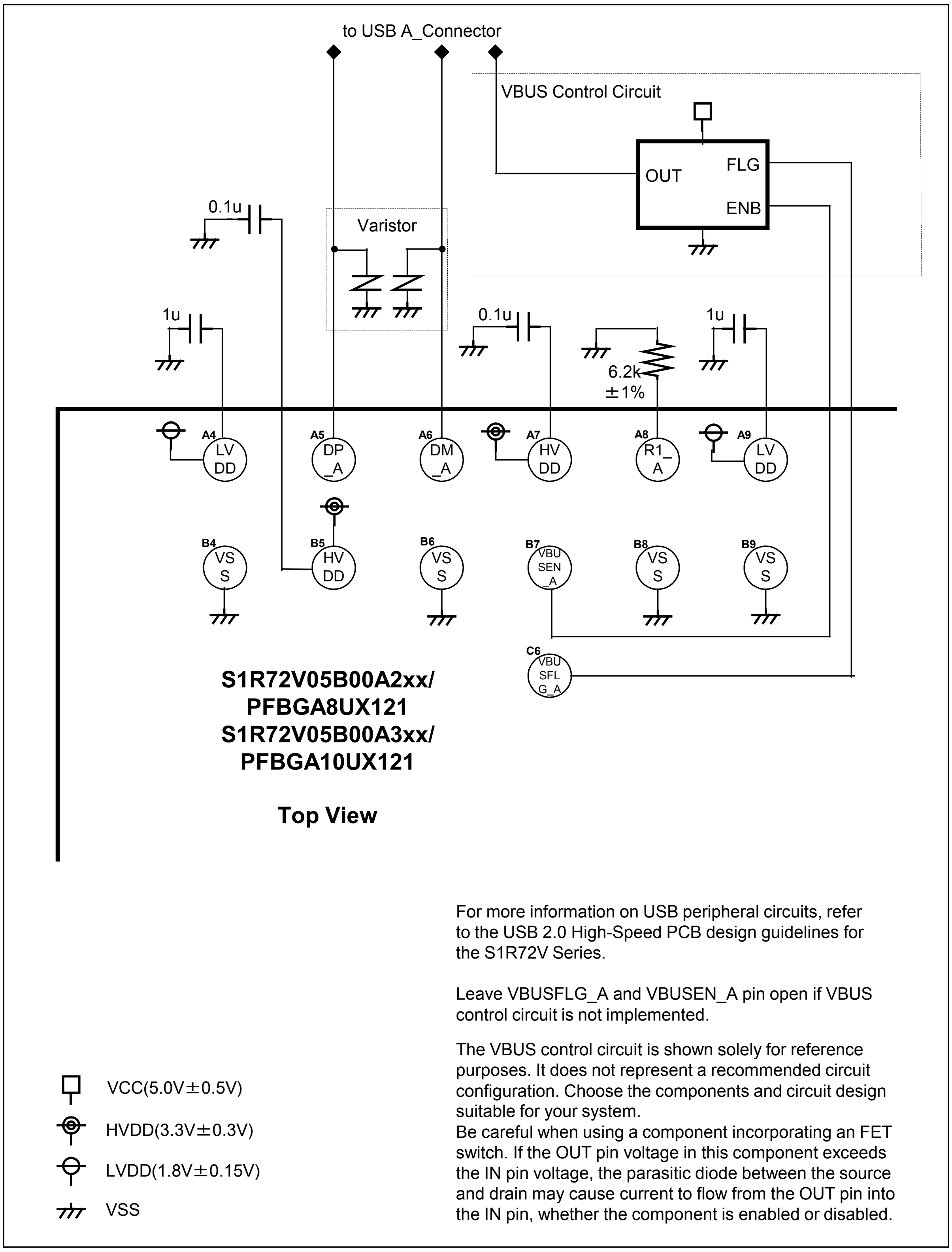
Take caution in selecting the power element device because its performance affects the quality of the USB signal waveform.

9.2.3 For PFBGA8UX121/PFBGA10UX121 (USB device peripheral)



Take caution in selecting the power element device because its performance affects the quality of the USB signal waveform.

9.2.4 For PFBGA8UX121/PFBGA10UX121 (USB host peripheral)



For more information on USB peripheral circuits, refer to the USB 2.0 High-Speed PCB design guidelines for the S1R72V Series.

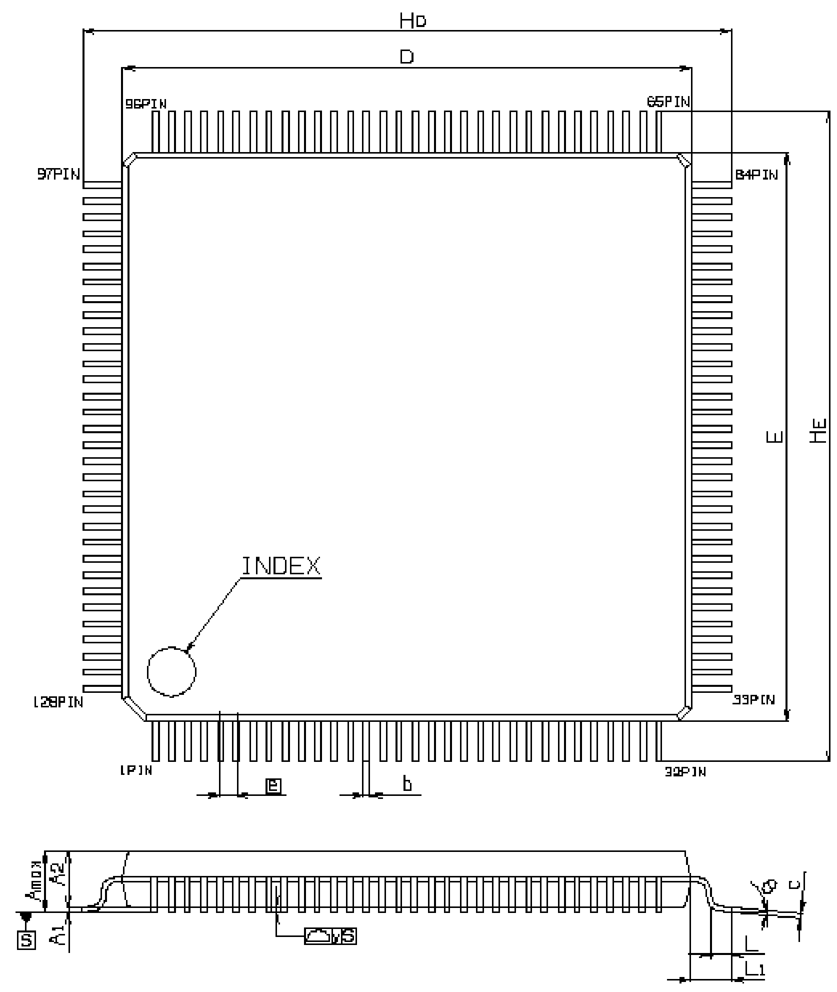
Leave VBUSFLG_A and VBUSEN_A pin open if VBUS control circuit is not implemented.

The VBUS control circuit is shown solely for reference purposes. It does not represent a recommended circuit configuration. Choose the components and circuit design suitable for your system.
Be careful when using a component incorporating an FET switch. If the OUT pin voltage in this component exceeds the IN pin voltage, the parasitic diode between the source and drain may cause current to flow from the OUT pin into the IN pin, whether the component is enabled or disabled.

Take caution in selecting the power element device because its performance affects the quality of the USB signal waveform.

10. Package Dimensions

10.1 QFP Package



Symbol	Dimension in Millimeters		
	Min	Nom	Max
E	-	14	-
D	-	14	-
A_{max}	-	-	1.7
A_1	-	0.1	-
A_2	-	1.4	-
$\boxed{e}$	-	0.4	-
b	0.13	-	0.23
c	0.09	-	0.2
θ	0°	-	10°
L	0.3	-	0.75
L_1	-	1	-
H_E	-	16	-
H_D	-	16	-
y	-	-	0.08

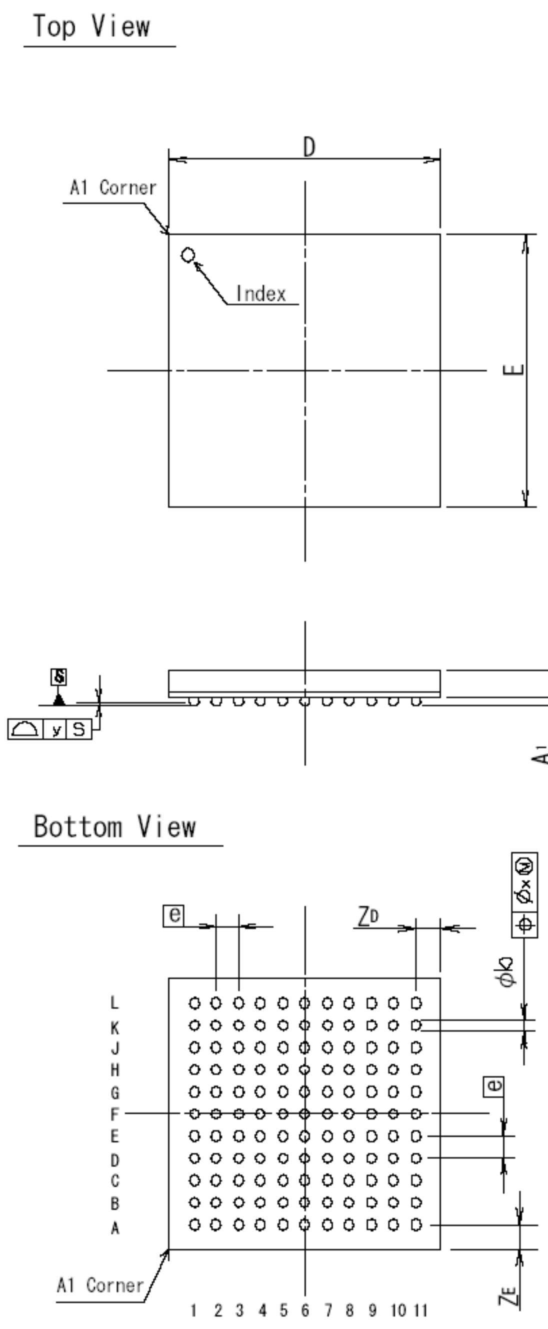
1 = 1mm

5.0	2004.08.11	C.SAITO	M.SONE	T.OTSUKI	-
REV.	DATE	DRAW	CHECKED	APPROVED	CORRECTION ARTICLE

TITLE P-LQFP128-1414-0.40(QFP15-128PIN)		-	
	CAD FILE	-	
SEIKO EPSON CORP.	DWG No.	4900-0057	
	SCALE	Free	SHEET 1 of 1

2990-002-01(Rev.1.1)

10.2 BGA Package(PFBGA8UX121)



Symbol	Dimension in Millimeters		
	Min	Nom	Max
D	–	8	–
E	–	8	–
A	–	–	1.2
A1	–	0.22	–
e	–	0.65	–
b	0.27	–	0.37
x	–	–	0.08
y	–	–	0.1
ZD	–	0.75	–
ZE	–	0.75	–

1 = 1mm

1.0	2005.03.25	O. SHOJI	M. SONE	T. OTSUKI	–
REV.	DATE	DRAW	CHECKED	APPROVED	CORRECTION ARTICLE

TITLE P-TFBGA-121-0808-0.65(PFBGA8U-121) –

CAD FILE –

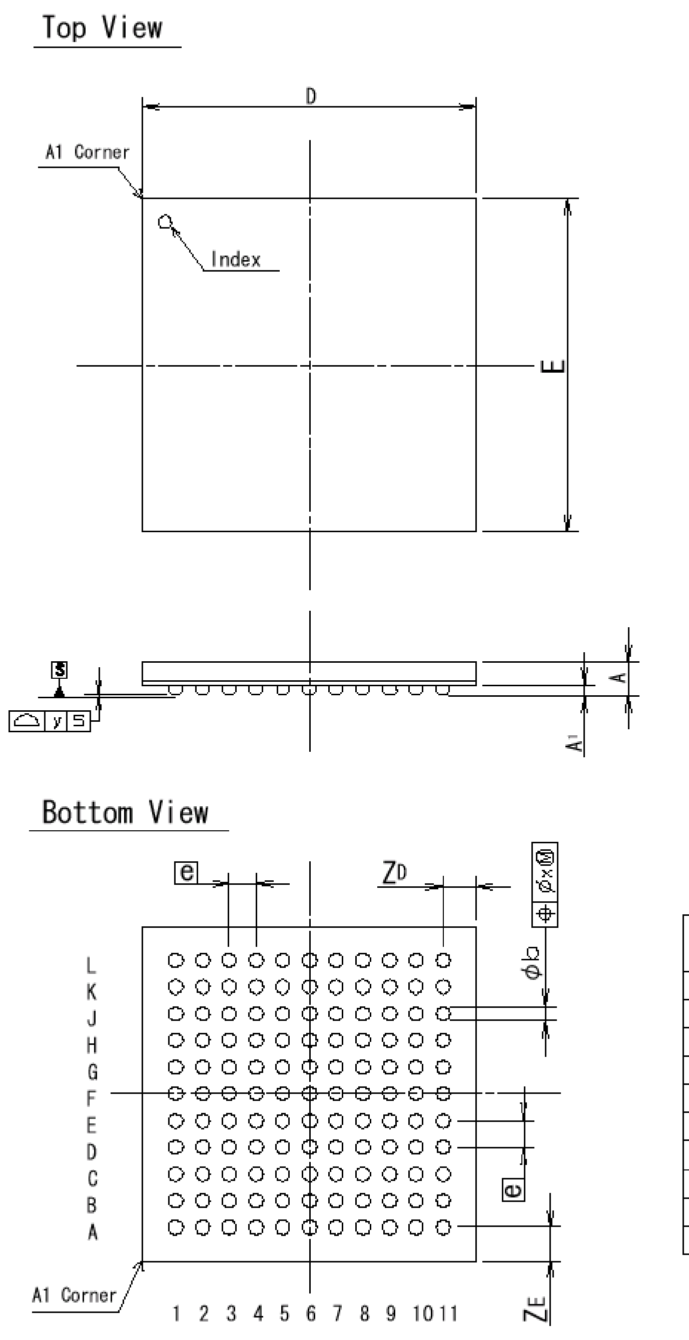
SEIKO EPSON CORP.

DWG No. 4900-0395

SCALE Free SHEET 1 of 1

2900-0002-01(Rev.1.1)

10.3 BGA Package(PFBGA10UX121)



Symbol	Dimension in Millimeters		
	Min	Nom	Max
D	–	10	–
E	–	10	–
A	–	–	1.2
A_1	–	0.3	–
$\boxed{e}$	–	0.8	–
b	0.38	–	0.48
x	–	–	0.08
y	–	–	0.1
Z_D	–	1	–
Z_E	–	1	–

1 = 1mm

2.0	2004.10.18	C. SHOJI	M. SONE	T. OTSUKI	—
REV.	DATE	DRAW	CHECKED	APPROVED	CORRECTION ARTICLE
TITLE	P-TFBGA-121-1010-0.80(PFBGA10U-121)				—
					CAD FILE —
SEIKO EPSON CORP.					DWG No. 4900-0350
					SCALE Free SHEET 1 of 1

2900-0002-01(Rev.1.1)

Appendix A IDE_Config_1.Swap Bit Settings

The internal buses of the S1R72V05 are configured in big-endian format, with [15:8] pins comprising the first byte. In contrast, the IDE I/F is configured in little-endian format, with [7:0] pins comprising the first byte. The S1R72V05 permits switch of the data bus connections between its internal buses and the IDE I/F via the IDE_Config_1.Swap bit.

Described below are chip hardware operations that vary depending on the IDE_Config_1.Swap bit setting.

Note that A -> B in the tables below indicates that value A is reflected in B.

- DMA transfer of data by the IDE_Control.IDE_Go bit

Swap	HDD[15:0]	
	IDE read	IDE write
0	HDD[15:0] -> Internal bus	Internal bus[15:0] -> HDD[15:0]
1	HDD[15:0] -> { Internal bus[7:0], Internal bus[15:8]}	Internal bus[15:0] -> {HDD[7:0], HDD[15:8]}

- IDE data register access

Swap	IDE data register HDD[15:0]			
	IDE read		IDE write	
	IDE_RegAdrs. IDE_RdReg	IDE_RegConfig. EnAutoStsRd	IDE_RegAdrs. IDE_WrReg	IDE_SeqWrRegControl. IDE_SeqWrReg

0	HDD[15:8] ->
	IDE_RdRegValue_0
	HDD[7:0] ->
	IDE_RdRegValue_1
		None	IDE_WrRegValue_0
			-> HDD[15:8]
			IDE_WrRegValue_1
			-> HDD[7:0]
				IDE_SeqWrRegValue(1st)
				-> HDD[7:0]
				IDE_SeqWrRegValue(2nd)
				-> HDD[15:8]

1	HDD[15:8] ->
	IDE_RdRegValue_1
	HDD[7:0] ->
	IDE_RdRegValue_0
		None	IDE_WrRegValue_0
			-> HDD[7:0]
			IDE_WrRegValue_1
			-> HDD[15:8]
				IDE_SeqWrRegValue(1st)
				-> HDD[15:8]
				IDE_SeqWrRegValue(2nd)
				-> HDD[7:0]

- IDE task file register access

Swap	IDE task file register HDD[7:0]			
	IDE read		IDE write	
	IDE_RegAdrs. IDE_RdReg	IDE_RegConfig. EnAutoStsRd	IDE_RegAdrs. IDE_WrReg	IDE_SeqWrRegControl. IDE_SeqWrReg
0	HDD[7:0] -> IDE_RdRegValue_0 HDD[7:0] -> IDE_RdRegValue_1	Same as shown to left	IDE_WrRegValue_1 -> HDD[7:0]	IDE_SeqWrRegValue -> HDD[7:0]
1	Same as shown above	Same as shown above	IDE_WrRegValue_0 -> HDD[7:0]	Same as shown above

Appendix B Connecting to Little Endian CPUs

The S1R72V05 internal buses are configured in big-endian format, with even and odd addresses, respectively, comprising the upper and lower bytes. To use the S1R72V05 with a little-endian CPU, connect to the little-endian CPU as described below.

<Circuit board>

Connect the little-endian CPU pins and the S1R72V05 pins one-for-one for data bus and control signals, as indicated by pin names. Connect CD15-CD8 of the S1R72V05 to data bus bits 15-8, or the upper byte of the CPU, and connect CD7-CD0 of the S1R72V05 to data bus bits 7-0, or the lower byte of the CPU. Similarly, you can also connect write signals directly: High for high and low for low.

However, keep in mind that write signal specifications for a specific CPU may differ from those for other CPUs.

<Firmware>

Use the procedure given below if the S1R72V05 must be operated in a little-endian CPU.

(1) Set the ChipConfig.CPU_Endian bit to 1.
Although this register in the S1R72V05 is mapped to address 0xB7, when operating in a little-endian CPU, it behaves as if this register is mapped to address 0xB6 until the operation in (2) is performed. This is because the S1R72V05 in its initial state is a big-endian device, meaning that the upper and lower bytes of write signals are reversed.

(2) Read address 0xB9.
This read operation causes the S1R72V05 to reverse the upper and lower bytes of its CPU buses. Keep in mind that performing (1) alone will not change the byte order of the CPU buses. Once this read operation is performed, all registers are precisely mapped to the addresses shown in the register map in the next pages.

Once these settings are made, all internal registers can be accessed in Char or Short. No problems should occur even when registers are accessed using the DMAC of the CPU (see the table below).

Example: Access to the FIFO_Rd_0/1 register when data is received from USB sequentially in order of 01, 02, 03, 04, 05, and 06

Accessed in Short	CPU access method			
	Big endian		Little endian	
	CD[15:8]	CD[7:0]	CD[15:8]	CD[7:0]
1st	01	02	02	01
2nd	03	04	04	03
3rd	05	06	06	05

However, use registers larger than Short by accessing separately in Short units and casting types in CPU memory.

<Register map>

Big Endian | **Little Endian**

Registers whose upper and lower bytes change during little-endian

Device/host common registers

	Big Endian		Little Endian
0x00	***MainIntStat***	0x00	***MainIntStat***
0x01	***DeviceIntStat***	0x01	***DeviceIntStat***
0x02	***HostIntStat***	0x02	***HostIntStat***
0x03	CPU_IntStat	0x03	CPU_IntStat
0x04	IDE_IntStat	0x04	IDE_IntStat
0x05	MediaFIFO_IntStat	0x05	MediaFIFO_IntStat
0x06		0x06	
0x07		0x07	
0x08		0x08	
0x09		0x09	
0x0A		0x0A	
0x0B		0x0B	
0x0C		0x0C	
0x0D		0x0D	
0x0E		0x0E	
0x0F		0x0F	

	Big Endian		Little Endian
0x10	***MainIntEnb***	0x10	***MainIntEnb***
0x11	***DeviceIntEnb***	0x11	***DeviceIntEnb***
0x12	***HostIntEnb***	0x12	***HostIntEnb***
0x13	CPU_IntEnb	0x13	CPU_IntEnb
0x14	IDE_IntEnb	0x14	IDE_IntEnb
0x15	MediaFIFO_IntEnb	0x15	MediaFIFO_IntEnb
0x16		0x16	
0x17		0x17	
0x18		0x18	
0x19		0x19	
0x1A		0x1A	
0x1B		0x1B	
0x1C		0x1C	
0x1D		0x1D	
0x1E		0x1E	
0x1F		0x1F	

	Big Endian		Little Endian
0x20	*RevisionNum*	0x20	*RevisionNum*
0x21	***ChipReset***	0x21	***ChipReset***
0x22	***PM_Control_0***	0x22	***PM_Control_0***
0x23	***PM_Control_1***	0x23	***PM_Control_1***
0x24	***WakeupTim_H***	0x24	***WakeupTim_L***
0x25	***WakeupTim_L***	0x25	***WakeupTim_H***
0x26	***H_USB_Control***	0x26	***H_USB_Control***
0x27	***H_XcvrControl***	0x27	***H_XcvrControl***
0x28	***D_USB_Status***	0x28	***D_USB_Status***
0x29	***H_USB_Status***	0x29	***H_USB_Status***
0x2A		0x2A	
0x2B		0x2B	
0x2C		0x2C	
0x2D		0x2D	
0x2E		0x2E	
0x2F		0x2F	

	Big Endian		Little Endian
0x30	FIFO_Rd_0	0x30	FIFO_Rd_0
0x31	FIFO_Rd_1	0x31	FIFO_Rd_1
0x32	FIFO_Wr_0	0x32	FIFO_Wr_0
0x33	FIFO_Wr_1	0x33	FIFO_Wr_1
0x34	FIFO_RdRemain_H	0x34	FIFO_RdRemain_L
0x35	FIFO_RdRemain_L	0x35	FIFO_RdRemain_H
0x36	FIFO_WrRemain_H	0x36	FIFO_WrRemain_L
0x37	FIFO_WrRemain_L	0x37	FIFO_WrRemain_H
0x38	FIFO_ByteRd	0x38	FIFO_ByteRd
0x39		0x39	
0x3A		0x3A	
0x3B		0x3B	
0x3C		0x3C	
0x3D		0x3D	
0x3E		0x3E	
0x3F		0x3F	

Big Endian

Device/host common registers

0x40	RAM_RdAdrs_H
0x41	RAM_RdAdrs_L
0x42	RAM_RdControl
0x43	RAM_RdCount
0x44	RAM_WrAdrs_H
0x45	RAM_WrAdrs_L
0x46	RAM_WrDoor_0
0x47	RAM_WrDoor_1
0x48	MediaFIFO_Control
0x49	ClrAllMediaFIFO_Join
0x4A	MediaFIFO_Join
0x4B	
0x4C	
0x4D	
0x4E	
0x4F	

0x50	RAM_Rd_00
0x51	RAM_Rd_01
0x52	RAM_Rd_02
0x53	RAM_Rd_03
0x54	RAM_Rd_04
0x55	RAM_Rd_05
0x56	RAM_Rd_06
0x57	RAM_Rd_07
0x58	RAM_Rd_08
0x59	RAM_Rd_09
0x510	RAM_Rd_0A
0x511	RAM_Rd_0B
0x512	RAM_Rd_0C
0x513	RAM_Rd_0D
0x514	RAM_Rd_0E
0x515	RAM_Rd_0F

0x60	RAM_Rd_10
0x61	RAM_Rd_11
0x62	RAM_Rd_12
0x63	RAM_Rd_13
0x64	RAM_Rd_14
0x65	RAM_Rd_15
0x66	RAM_Rd_16
0x67	RAM_Rd_17
0x68	RAM_Rd_18
0x69	RAM_Rd_19
0x6A	RAM_Rd_1A
0x6B	RAM_Rd_1B
0x6C	RAM_Rd_1C
0x6D	RAM_Rd_1D
0x6E	RAM_Rd_1E
0x6F	RAM_Rd_1F

0x70	
0x71	DMA0_Config
0x72	DMA0_Control
0x73	
0x74	DMA0_Remain_H
0x75	DMA0_Remain_L
0x76	
0x77	
0x78	DMA0_Count_HH
0x79	DMA0_Count_HL
0x7A	DMA0_Count_LH
0x7B	DMA0_Count_LL
0x7C	DMA0_RdData_0
0x7D	DMA0_RdData_1
0x7E	DMA0_WrData_0
0x7F	DMA0_WrData_1

Little Endian

Registers whose upper and lower bytes change during little-endian

Device/host common registers

0x40	RAM_RdAdrs_L
0x41	RAM_RdAdrs_H
0x42	RAM_RdControl
0x43	RAM_RdCount
0x44	RAM_WrAdrs_L
0x45	RAM_WrAdrs_H
0x46	RAM_WrDoor_0
0x47	RAM_WrDoor_1
0x48	MediaFIFO_Control
0x49	ClrAllMediaFIFO_Join
0x4A	MediaFIFO_Join
0x4B	
0x4C	
0x4D	
0x4E	
0x4F	

0x50	RAM_Rd_00
0x51	RAM_Rd_01
0x52	RAM_Rd_02
0x53	RAM_Rd_03
0x54	RAM_Rd_04
0x55	RAM_Rd_05
0x56	RAM_Rd_06
0x57	RAM_Rd_07
0x58	RAM_Rd_08
0x59	RAM_Rd_09
0x5A	RAM_Rd_0A
0x5B	RAM_Rd_0B
0x5C	RAM_Rd_0C
0x5D	RAM_Rd_0D
0x5E	RAM_Rd_0E
0x5F	RAM_Rd_0F

0x60	RAM_Rd_10
0x61	RAM_Rd_11
0x62	RAM_Rd_12
0x63	RAM_Rd_13
0x64	RAM_Rd_14
0x65	RAM_Rd_15
0x66	RAM_Rd_16
0x67	RAM_Rd_17
0x68	RAM_Rd_18
0x69	RAM_Rd_19
0x6A	RAM_Rd_1A
0x6B	RAM_Rd_1B
0x6C	RAM_Rd_1C
0x6D	RAM_Rd_1D
0x6E	RAM_Rd_1E
0x6F	RAM_Rd_1F

0x70	
0x71	DMA0_Config
0x72	DMA0_Control
0x73	
0x74	DMA0_Remain_L
0x75	DMA0_Remain_H
0x76	
0x77	
0x78	DMA0_Count_HL
0x79	DMA0_Count_HH
0x7A	DMA0_Count_LL
0x7B	DMA0_Count_LH
0x7C	DMA0_RdData_0
0x7D	DMA0_RdData_1
0x7E	DMA0_WrData_0
0x7F	DMA0_WrData_1

Big Endian

Device/host common registers

0x80	
0x81	DMA1_Config
0x82	DMA1_Control
0x83	
0x84	DMA1_Remain_H
0x85	DMA1_Remain_L
0x86	
0x87	
0x88	DMA1_Count_HH
0x89	DMA1_Count_HL
0x8A	DMA1_Count_LH
0x8B	DMA1_Count_LL
0x8C	DMA1_RdData_0
0x8D	DMA1_RdData_1
0x8E	DMA1_WrData_0
0x8F	DMA1_WrData_1

0x90	IDE_Status
0x91	IDE_Control
0x92	IDE_Config_0
0x93	IDE_Config_1
0x94	IDE_Rmod
0x95	IDE_Tmod
0x96	IDE_Umod
0x97	
0x98	
0x99	
0x9A	IDE_CRC_H
0x9B	IDE_CRC_L
0x9C	
0x9D	IDE_Count_H
0x9E	IDE_Count_M
0x9F	IDE_Count_L

0xA0	IDE_RegAdrs
0xA1	
0xA2	IDE_RdRegValue_0
0xA3	IDE_RdRegValue_1
0xA4	IDE_WrRegValue_0
0xA5	IDE_WrRegValue_1
0xA6	IDE_SeqWrRegControl
0xA7	IDE_SeqWrRegCnt
0xA8	IDE_SeqWrRegAdrs
0xA9	IDE_SeqWrRegValue
0xAA	
0xAB	
0xAC	IDE_RegConfig
0xAD	
0xAE	
0xAF	

0xB0	
0xB1	***HostDeviceSel***
0xB2	
0xB3	***ModeProtect***
0xB4	
0xB5	***ClkSelect***
0xB6	
0xB7	***ChipConfig***
0xB8	
0xB9	***CPU_ChgEndian***
0xBA	
0xBB	
0xBC	
0xBD	
0xBE	
0xBF	

0xC0～0xDF:Reserved

Little Endian

Registers whose upper and lower bytes change during little-endian

Device/host common registers

0x80	
0x81	DMA1_Config
0x82	DMA1_Control
0x83	
0x84	DMA1_Remain_L
0x85	DMA1_Remain_H
0x86	
0x87	
0x88	DMA1_Count_HL
0x89	DMA1_Count_HH
0x8A	DMA1_Count_LL
0x8B	DMA1_Count_LH
0x8C	DMA1_RdData_0
0x8D	DMA1_RdData_1
0x8E	DMA1_WrData_0
0x8F	DMA1_WrData_1

0x90	IDE_Status
0x91	IDE_Control
0x92	IDE_Config_0
0x93	IDE_Config_1
0x94	IDE_Rmod
0x95	IDE_Tmod
0x96	IDE_Umod
0x97	
0x98	
0x99	
0x9A	IDE_CRC_L
0x9B	IDE_CRC_H
0x9C	IDE_Count_H
0x9D	
0x9E	IDE_Count_L
0x9F	IDE_Count_M

0xA0	IDE_RegAdrs
0xA1	
0xA2	IDE_RdRegValue_0
0xA3	IDE_RdRegValue_1
0xA4	IDE_WrRegValue_0
0xA5	IDE_WrRegValue_1
0xA6	IDE_SeqWrRegControl
0xA7	IDE_SeqWrRegCnt
0xA8	IDE_SeqWrRegAdrs
0xA9	IDE_SeqWrRegValue
0xAA	
0xAB	
0xAC	IDE_RegConfig
0xAD	
0xAE	
0xAF	

0xB0	
0xB1	***HostDeviceSel***
0xB2	
0xB3	***ModeProtect***
0xB4	
0xB5	***ClkSelect***
0xB6	
0xB7	***ChipConfig***
0xB8	
0xB9	***CPU_ChgEndian***
0xBA	
0xBB	
0xBC	
0xBD	
0xBE	
0xBF	

0xC0～0xDF:Reserved

Device registers
Big Endian

0xE0	***D_SIE_IntStat***
0xE1	
0xE2	D_FIFO_IntStat
0xE3	D_BulkIntStat
0xE4	D_EPrIntStat
0xE5	D_EP0IntStat
0xE6	D_EPaIntStat
0xE7	D_EPbIntStat
0xE8	D_EPcIntStat
0xE9	
0xEA	
0xEB	
0xEC	
0xED	
0xEE	
0xEF	

0xF0	***D_SIE_IntEnb***
0xF1	
0xF2	D_FIFO_IntEnb
0xF3	D_BulkIntEnb
0xF4	D_EPrIntEnb
0xF5	D_EP0IntEnb
0xF6	D_EPaIntEnb
0xF7	D_EPbIntEnb
0xF8	D_EPcIntEnb
0xF9	
0xFA	
0xFB	
0xFC	
0xFD	
0xFE	
0xFF	

0x100	***D_Reset***
0x101	
0x102	D_NegoControl
0x103	
0x104	D_ClrAllEPnJoin
0x105	D_XcvrControl
0x106	D_USB_Test
0x107	
0x108	D_EPnControl
0x109	D_EPrFIFO_Clr
0x10A	D_BulkOnlyControl
0x10B	D_BulkOnlyConfig
0x10C	
0x10D	
0x10E	
0x10F	

0x110	D_EP0SETUP_0
0x111	D_EP0SETUP_1
0x112	D_EP0SETUP_2
0x113	D_EP0SETUP_3
0x114	D_EP0SETUP_4
0x115	D_EP0SETUP_5
0x116	D_EP0SETUP_6
0x117	D_EP0SETUP_7
0x118	D_USB_Address
0x119	
0x11A	D_SETUP_Control
0x11B	
0x11C	
0x11D	
0x11E	D_FrameNumber_H
0x11F	D_FrameNumber_L

Device registers
Little Endian

Registers whose upper and lower bytes change during little-endian

0xE0	***D_SIE_IntStat***
0xE1	
0xE2	D_FIFO_IntStat
0xE3	D_BulkIntStat
0xE4	D_EPrIntStat
0xE5	D_EP0IntStat
0xE6	D_EPaIntStat
0xE7	D_EPbIntStat
0xE8	D_EPcIntStat
0xE9	
0xEA	
0xEB	
0xEC	
0xED	
0xEE	
0xEF	

0xF0	***D_SIE_IntEnb***
0xF1	
0xF2	D_FIFO_IntEnb
0xF3	D_BulkIntEnb
0xF4	D_EPrIntEnb
0xF5	D_EP0IntEnb
0xF6	D_EPaIntEnb
0xF7	D_EPbIntEnb
0xF8	D_EPcIntEnb
0xF9	
0xFA	
0xFB	
0xFC	
0xFD	
0xFE	
0xFF	

0x100	***D_Reset***
0x101	
0x102	D_NegoControl
0x103	
0x104	D_ClrAllEPnJoin
0x105	D_XcvrControl
0x106	D_USB_Test
0x107	
0x108	D_EPnControl
0x109	D_EPrFIFO_Clr
0x10A	D_BulkOnlyControl
0x10B	D_BulkOnlyConfig
0x10C	
0x10D	
0x10E	
0x10F	

0x110	D_EP0SETUP_0
0x111	D_EP0SETUP_1
0x112	D_EP0SETUP_2
0x113	D_EP0SETUP_3
0x114	D_EP0SETUP_4
0x115	D_EP0SETUP_5
0x116	D_EP0SETUP_6
0x117	D_EP0SETUP_7
0x118	D_USB_Address
0x119	
0x11A	D_SETUP_Control
0x11B	
0x11C	
0x11D	
0x11E	D_FrameNumber_L
0x11F	D_FrameNumber_H

Device registers
Big Endian

0x120	D_EP0MaxSize
0x121	D_EP0Control
0x122	D_EP0ControlIN
0x123	D_EP0ControlOUT
0x124	
0x125	D_EP0Join
0x126	
0x127	
0x128	
0x129	
0x12A	
0x12B	
0x12C	
0x12D	
0x12E	
0x12F	

0x130	D_EPaMaxSize_H
0x131	D_EPaMaxSize_L
0x132	D_EPaConfig_0
0x133	
0x134	D_EPaControl
0x135	D_EPaJoin
0x136	
0x137	
0x138	
0x139	
0x13A	
0x13B	
0x13C	
0x13D	
0x13E	
0x13F	

0x140	D_EPbMaxSize_H
0x141	D_EPbMaxSize_L
0x142	D_EPbConfig_0
0x143	
0x144	D_EPbControl
0x145	D_EPbJoin
0x146	
0x147	
0x148	
0x149	
0x14A	
0x14B	
0x14C	
0x14D	
0x14E	
0x14F	

0x150	D_EPcMaxSize_H
0x151	D_EPcMaxSize_L
0x152	D_EPcConfig_0
0x153	
0x154	D_EPcControl
0x155	D_EPcJoin
0x156	
0x157	
0x158	
0x159	
0x15A	
0x15B	
0x15C	
0x15D	
0x15E	
0x15F	

Device registers
Little Endian

Registers whose upper and lower bytes change during little-endian

0x120	D_EP0MaxSize
0x121	D_EP0Control
0x122	D_EP0ControlIN
0x123	D_EP0ControlOUT
0x124	
0x125	D_EP0Join
0x126	
0x127	
0x128	
0x129	
0x12A	
0x12B	
0x12C	
0x12D	
0x12E	
0x12F	

0x130	D_EPaMaxSize_L
0x131	D_EPaMaxSize_H
0x132	D_EPaConfig_0
0x133	
0x134	D_EPaControl
0x135	D_EPaJoin
0x136	
0x137	
0x138	
0x139	
0x13A	
0x13B	
0x13C	
0x13D	
0x13E	
0x13F	

0x140	D_EPbMaxSize_L
0x141	D_EPbMaxSize_H
0x142	D_EPbConfig_0
0x143	
0x144	D_EPbControl
0x145	D_EPbJoin
0x146	
0x147	
0x148	
0x149	
0x14A	
0x14B	
0x14C	
0x14D	
0x14E	
0x14F	

0x150	D_EPcMaxSize_L
0x151	D_EPcMaxSize_H
0x152	D_EPcConfig_0
0x153	
0x154	D_EPcControl
0x155	D_EPcJoin
0x156	
0x157	
0x158	
0x159	
0x15A	
0x15B	
0x15C	
0x15D	
0x15E	
0x15F	

Device registers
Big Endian

0x160	D_DescAdrs_H
0x161	D_DescAdrs_L
0x162	D_DescSize_H
0x163	D_DescSize_L
0x164	
0x165	
0x166	
0x167	
0x168	
0x169	
0x16A	
0x16B	
0x16C	
0x16D	
0x16E	
0x16F	

0x170	D_DMA0_FIFO_Control
0x171	
0x172	D_DMA1_FIFO_Control
0x173	
0x174	
0x175	
0x176	
0x177	
0x178	
0x179	
0x17A	
0x17B	
0x17C	
0x17D	
0x17E	
0x17F	

0x180	
0x181	
0x182	
0x183	
0x184	D_EPaStartAdrs_H
0x185	D_EPaStartAdrs_L
0x186	
0x187	
0x188	D_EPbStartAdrs_H
0x189	D_EPbStartAdrs_L
0x18A	
0x18B	
0x18C	D_EPcStartAdrs_H
0x18D	D_EPcStartAdrs_L
0x18E	D_EPcEndAdrs_H
0x18F	D_EPcEndAdrs_L

0x190~0x1DF:Reserved

0x1E0	(Reserved)
0x1E1	D_ModeControl

0x1E2~0x1FF:Reserved

Device registers
Little Endian

Registers whose upper and lower bytes change during little-endian

0x160	D_DescAdrs_L
0x161	D_DescAdrs_H
0x162	D_DescSize_L
0x163	D_DescSize_H
0x164	
0x165	
0x166	
0x167	
0x168	
0x169	
0x16A	
0x16B	
0x16C	
0x16D	
0x16E	
0x16F	

0x170	D_DMA0_FIFO_Control
0x171	
0x172	D_DMA1_FIFO_Control
0x173	
0x174	
0x175	
0x176	
0x177	
0x178	
0x179	
0x17A	
0x17B	
0x17C	
0x17D	
0x17E	
0x17F	

0x180	
0x181	
0x182	
0x183	
0x184	D_EPaStartAdrs_L
0x185	D_EPaStartAdrs_H
0x186	
0x187	
0x188	D_EPbStartAdrs_L
0x189	D_EPbStartAdrs_H
0x18A	
0x18B	
0x18C	D_EPcStartAdrs_L
0x18D	D_EPcStartAdrs_H
0x18E	D_EPcEndAdrs_L
0x18F	D_EPcEndAdrs_H

0x190~0x1DF:Reserved

0x1E0	(Reserved)
0x1E1	D_ModeControl

'0x1E2~0x1FF:Reserved

Device registers
Big Endian

0xE0	H_SIE_IntStat_0
0xE1	H_SIE_IntStat_1
0xE2	H_FIFO_IntStat
0xE3	H_FrameIntStat
0xE4	H_CHrIntStat
0xE5	H_CH0IntStat
0xE6	H_CHaIntStat
0xE7	H_CHbIntStat
0xE8	H_CHcIntStat
0xE9	H_CHdIntStat
0xEA	H_CHeIntStat
0xEB	
0xEC	
0xED	
0xEE	
0xEF	

0xF0	H_SIE_IntEnb_0
0xF1	H_SIE_IntEnb_1
0xF2	H_FIFO_IntEnb
0xF3	H_FrameIntEnb
0xF4	H_CHrIntEnb
0xF5	H_CH0IntEnb
0xF6	H_CHaIntEnb
0xF7	H_CHbIntEnb
0xF8	H_CHcIntEnb
0xF9	H_CHdIntEnb
0xFA	H_CHeIntEnb
0xFB	
0xFC	
0xFD	
0xFE	
0xFF	

0x100	***H_Reset***
0x101	
0x102	H_NegoControl_0
0x103	
0x104	H_NegoControl_1
0x105	
0x106	H_USB_Test
0x107	
0x108	H_CHnControl
0x109	H_CHrFIFO_Clr
0x10A	H_ClrAllCHnJoin
0x10B	
0x10C	
0x10D	
0x10E	
0x10F	

0x110	H_CH0SETUP_0
0x111	H_CH0SETUP_1
0x112	H_CH0SETUP_2
0x113	H_CH0SETUP_3
0x114	H_CH0SETUP_4
0x115	H_CH0SETUP_5
0x116	H_CH0SETUP_6
0x117	H_CH0SETUP_7
0x118	
0x119	
0x11A	
0x11B	
0x11C	
0x11D	
0x11E	H_FrameNumber_H
0x11F	H_FrameNumber_L

Device registers
Little Endian
Registers whose upper and lower bytes change during little-endian

0xE0	H_SIE_IntStat_0
0xE1	H_SIE_IntStat_1
0xE2	H_FIFO_IntStat
0xE3	H_FrameIntStat
0xE4	H_CHrIntStat
0xE5	H_CH0IntStat
0xE6	H_CHaIntStat
0xE7	H_CHbIntStat
0xE8	H_CHcIntStat
0xE9	H_CHdIntStat
0xEA	H_CHeIntStat
0xEB	
0xEC	
0xED	
0xEE	
0xEF	

0xF0	H_SIE_IntEnb_0
0xF1	H_SIE_IntEnb_1
0xF2	H_FIFO_IntEnb
0xF3	H_FrameIntEnb
0xF4	H_CHrIntEnb
0xF5	H_CH0IntEnb
0xF6	H_CHaIntEnb
0xF7	H_CHbIntEnb
0xF8	H_CHcIntEnb
0xF9	H_CHdIntEnb
0xFA	H_CHeIntEnb
0xFB	
0xFC	
0xFD	
0xFE	
0xFF	

0x100	***H_Reset***
0x101	
0x102	H_NegoControl_0
0x103	
0x104	H_NegoControl_1
0x105	
0x106	H_USB_Test
0x107	
0x108	H_CHnControl
0x109	H_CHrFIFO_Clr
0x10A	H_ClrAllCHnJoin
0x10B	
0x10C	
0x10D	
0x10E	
0x10F	

0x110	H_CH0SETUP_0
0x111	H_CH0SETUP_1
0x112	H_CH0SETUP_2
0x113	H_CH0SETUP_3
0x114	H_CH0SETUP_4
0x115	H_CH0SETUP_5
0x116	H_CH0SETUP_6
0x117	H_CH0SETUP_7
0x118	
0x119	
0x11A	
0x11B	
0x11C	
0x11D	
0x11E	H_FrameNumber_L
0x11F	H_FrameNumber_H

Device registers
Big Endian

0x120	H_CH0Config_0
0x121	H_CH0Config_1
0x122	
0x123	H_CH0MaxPktSize
0x124	
0x125	
0x126	H_CH0TotalSize_H
0x127	H_CH0TotalSize_L
0x128	H_CH0HubAdrs
0x129	H_CH0FuncAdrs
0x12A	
0x12B	H_CTL_SupportControl
0x12C	
0x12D	
0x12E	H_CH0ConditionCode
0x12F	H_CH0Join

0x130	H_CHaConfig_0
0x131	H_CHaConfig_1
0x132	H_CHaMaxPktSize_H
0x133	H_CHaMaxPktSize_L
0x134	H_CHaTotalSize_HH
0x135	H_CHaTotalSize_HL
0x136	H_CHaTotalSize_LH
0x137	H_CHaTotalSize_LL
0x138	H_CHaHubAdrs
0x139	H_CHaFuncAdrs
0x13A	H_BO_SupportControl
0x13B	H_CSW_RcvDataSize
0x13C	H_OUT_EP_Control
0x13D	H_IN_EP_Control
0x13E	H_CHaConditionCode
0x13F	H_CHaJoin

0x140	H_CHbConfig_0
0x141	H_CHbConfig_1
0x142	H_CHbMaxPktSize_H
0x143	H_CHbMaxPktSize_L
0x144	H_CHbTotalSize_HH
0x145	H_CHbTotalSize_HL
0x146	H_CHbTotalSize_LH
0x147	H_CHbTotalSize_LL
0x148	H_CHbHubAdrs
0x149	H_CHbFuncAdrs
0x14A	H_CHbInterval_H
0x14B	H_CHbInterval_L
0x14C	
0x14D	
0x14E	H_CHbConditionCode
0x14F	H_CHbJoin

0x150	H_CHcConfig_0
0x151	H_CHcConfig_1
0x152	H_CHcMaxPktSize_H
0x153	H_CHcMaxPktSize_L
0x154	H_CHcTotalSize_HH
0x155	H_CHcTotalSize_HL
0x156	H_CHcTotalSize_LH
0x157	H_CHcTotalSize_LL
0x158	H_CHcHubAdrs
0x159	H_CHcFuncAdrs
0x15A	H_CHcInterval_H
0x15B	H_CHcInterval_L
0x15C	
0x15D	
0x15E	H_CHcConditionCode
0x15F	H_CHcJoin

Device registers
Little Endian

Registers whose upper and lower bytes change during little-endian

0x120	H_CH0Config_0
0x121	H_CH0Config_1
0x122	H_CH0MaxPktSize
0x123	
0x124	
0x125	
0x126	H_CH0TotalSize_L
0x127	H_CH0TotalSize_H
0x128	H_CH0HubAdrs
0x129	H_CH0FuncAdrs
0x12A	
0x12B	H_CTL_SupportControl
0x12C	
0x12D	
0x12E	H_CH0ConditionCode
0x12F	H_CH0Join

0x130	H_CHaConfig_0
0x131	H_CHaConfig_1
0x132	H_CHaMaxPktSize_L
0x133	H_CHaMaxPktSize_H
0x134	H_CHaTotalSize_HL
0x135	H_CHaTotalSize_HH
0x136	H_CHaTotalSize_LL
0x137	H_CHaTotalSize_LH
0x138	H_CHaHubAdrs
0x139	H_CHaFuncAdrs
0x13A	H_BO_SupportControl
0x13B	H_CSW_RcvDataSize
0x13C	H_OUT_EP_Control
0x13D	H_IN_EP_Control
0x13E	H_CHaConditionCode
0x13F	H_CHaJoin

0x140	H_CHbConfig_0
0x141	H_CHbConfig_1
0x142	H_CHbMaxPktSize_L
0x143	H_CHbMaxPktSize_H
0x144	H_CHbTotalSize_HL
0x145	H_CHbTotalSize_HH
0x146	H_CHbTotalSize_LL
0x147	H_CHbTotalSize_LH
0x148	H_CHbHubAdrs
0x149	H_CHbFuncAdrs
0x14A	H_CHbInterval_L
0x14B	H_CHbInterval_H
0x14C	
0x14D	
0x14E	H_CHbConditionCode
0x14F	H_CHbJoin

0x150	H_CHcConfig_0
0x151	H_CHcConfig_1
0x152	H_CHcMaxPktSize_L
0x153	H_CHcMaxPktSize_H
0x154	H_CHcTotalSize_HL
0x155	H_CHcTotalSize_HH
0x156	H_CHcTotalSize_LL
0x157	H_CHcTotalSize_LH
0x158	H_CHcHubAdrs
0x159	H_CHcFuncAdrs
0x15A	H_CHcInterval_L
0x15B	H_CHcInterval_H
0x15C	
0x15D	
0x15E	H_CHcConditionCode
0x15F	H_CHcJoin

Host registers
Big Endian

0x160	H_CHdConfig_0
0x161	H_CHdConfig_1
0x162	H_CHdMaxPktSize_H
0x163	H_CHdMaxPktSize_L
0x164	H_CHdTotalSize_HH
0x165	H_CHdTotalSize_HL
0x166	H_CHdTotalSize_LH
0x167	H_CHdTotalSize_LL
0x168	H_CHdHubAdrs
0x169	H_CHdFuncAdrs
0x16A	H_CHdInterval_H
0x16B	H_CHdInterval_L
0x16C	
0x16D	
0x16E	H_CHdConditionCode
0x16F	H_CHdJoin

0x170	H_CHeConfig_0
0x171	H_CHeConfig_1
0x172	H_CHeMaxPktSize_H
0x173	H_CHeMaxPktSize_L
0x174	H_CHeTotalSize_HH
0x175	H_CHeTotalSize_HL
0x176	H_CHeTotalSize_LH
0x177	H_CHeTotalSize_LL
0x178	H_CHeHubAdrs
0x179	H_CHeFuncAdrs
0x17A	H_CHeInterval_H
0x17B	H_CHeInterval_L
0x17C	
0x17D	
0x17E	H_CHeConditionCode
0x17F	H_CHeJoin

0x180	H_CH0StartAdrs_H
0x181	H_CH0StartAdrs_L
0x182	H_CH0EndAdrs_H
0x183	H_CH0EndAdrs_L
0x184	H_CHaStartAdrs_H
0x185	H_CHaStartAdrs_L
0x186	H_CHaEndAdrs_H
0x187	H_CHaEndAdrs_L
0x188	H_CHbStartAdrs_H
0x189	H_CHbStartAdrs_L
0x18A	H_CHbEndAdrs_H
0x18B	H_CHbEndAdrs_L
0x18C	H_CHcStartAdrs_H
0x18D	H_CHcStartAdrs_L
0x18E	H_CHcEndAdrs_H
0x18F	H_CHcEndAdrs_L

0x190	H_CHdStartAdrs_H
0x191	H_CHdStartAdrs_L
0x192	H_CHdEndAdrs_H
0x193	H_CHdEndAdrs_L
0x194	H_CHeStartAdrs_H
0x195	H_CHeStartAdrs_L
0x196	H_CHeEndAdrs_H
0x197	H_CHeEndAdrs_L

0x198～0x1F3: Reserved

0x1F4	(Reserved)
0x1F5	H_Protect
0x1F6	H_Monitor
0x1F7	(Reserved)

0x1F8～0x1FF:Reserved

Host registers
Little Endian

Registers whose upper and lower bytes change during little-endian

0x160	H_CHdConfig_0
0x161	H_CHdConfig_1
0x162	H_CHdMaxPktSize_L
0x163	H_CHdMaxPktSize_H
0x164	H_CHdTotalSize_HL
0x165	H_CHdTotalSize_HH
0x166	H_CHdTotalSize_LL
0x167	H_CHdTotalSize_LH
0x168	H_CHdHubAdrs
0x169	H_CHdFuncAdrs
0x16A	H_CHdInterval_L
0x16B	H_CHdInterval_H
0x16C	
0x16D	
0x16E	H_CHdConditionCode
0x16F	H_CHdJoin

0x170	H_CHeConfig_0
0x171	H_CHeConfig_1
0x172	H_CHeMaxPktSize_L
0x173	H_CHeMaxPktSize_H
0x174	H_CHeTotalSize_HL
0x175	H_CHeTotalSize_HH
0x176	H_CHeTotalSize_LL
0x177	H_CHeTotalSize_LH
0x178	H_CHeHubAdrs
0x179	H_CHeFuncAdrs
0x17A	H_CHeInterval_L
0x17B	H_CHeInterval_H
0x17C	
0x17D	
0x17E	H_CHeConditionCode
0x17F	H_CHeJoin

0x180	H_CH0StartAdrs_L
0x181	H_CH0StartAdrs_H
0x182	H_CH0EndAdrs_L
0x183	H_CH0EndAdrs_H
0x184	H_CHaStartAdrs_L
0x185	H_CHaStartAdrs_H
0x186	H_CHaEndAdrs_L
0x187	H_CHaEndAdrs_H
0x188	H_CHbStartAdrs_L
0x189	H_CHbStartAdrs_H
0x18A	H_CHbEndAdrs_L
0x18B	H_CHbEndAdrs_H
0x18C	H_CHcStartAdrs_L
0x18D	H_CHcStartAdrs_H
0x18E	H_CHcEndAdrs_L
0x18F	H_CHcEndAdrs_H

0x190	H_CHdStartAdrs_L
0x191	H_CHdStartAdrs_H
0x192	H_CHdEndAdrs_L
0x193	H_CHdEndAdrs_H
0x194	H_CHeStartAdrs_L
0x195	H_CHeStartAdrs_H
0x196	H_CHeEndAdrs_L
0x197	H_CHeEndAdrs_H

0x198～0x1F3: Reserved

0x1F4	(Reserved)
0x1F5	H_Protect
0x1F6	H_Monitor
0x1F7	(Reserved)

0x1F8～0x1FF:Reserved

Appendix C 1-Port Mode

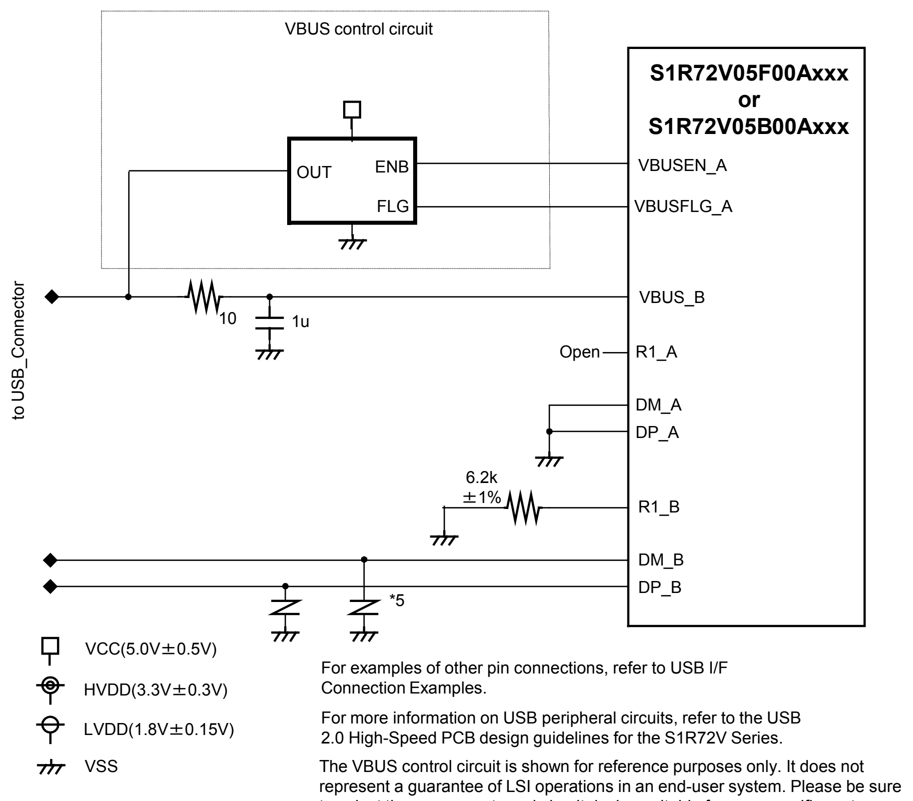
Set the ClkSelectPORT1x2 bit during initialization to set the S1R72V05 to 2-port (ClkSelectPORT1x2 = 0) or 1-port mode (ClkSelectPORT1x2 = 1). In 2-port mode, host and device functions are implemented with ports A and B, respectively. In 1-port mode, on the other hand, the host and the device functions are both implemented with port B only.

If you select 1-port mode, you must process port A or the free port by the procedure given below. Additionally, with the device operating in host mode, use the VBUS control signals (VBUSEN_A and VBUSFLG_A) in the VBUS control circuit regardless of suffix "_A."

If the S1R72V05 is used in 1-port mode, make sure the PM_Control_0.GoActHost bit is not set. This blocks use of the ACT_HOST status of the power management function. If the S1R72V05 is used in 1-port mode, use the host or device function, whichever is executed, with the ACT_DEVICE status of the power management function.

Also confirm that both the D_Reset.ResetDTM and H_Reset.ResetHTM bits are set to 0.

<Connection Example>



For examples of other pin connections, refer to USB I/F Connection Examples.

For more information on USB peripheral circuits, refer to the USB 2.0 High-Speed PCB design guidelines for the S1R72V Series.

The VBUS control circuit is shown for reference purposes only. It does not represent a guarantee of LSI operations in an end-user system. Please be sure to select the components and circuit design suitable for your specific system.

Leave VBUSFLG_A and VBUSEN_A open if VBUS control circuit is not implemented.

*5: This is a static protective device (varistor).

Appendix D SUSPEND during HOST High-Speed Operations

The disconnect detection function is disabled when the USB bus is in SUSPEND state with the S1R72V05's USB Host Port operating in HS mode. In this case, no H_SIE_IntStat.DetectDiscon interrupt status is issued, even when the cable is disconnected.

* Here, the USB Host Port refers to the following:

USB Port A when ClkSelect.PORT1x2 == "0"

USB Port B when ClkSelect.PORT1x2 == "1" and HostDeviceSel.HOSTxDEVICE == "1"

* The condition that the USB Host Port operates in HS mode applies in the following case:

H_NegoControl_1.PortSpeed == "HS (0b00)"

This register is automatically set by hardware when the auto-negotiation function executes.

* The condition that the USB bus operates in SUSPEND state applies to the following case:

H_NegoControl_0.HostState == "USB_SUSPEND (0b110)"

This register is automatically set by hardware when the auto-negotiation function executes.

Take one of the following corrective measures:

1. Avoid the SUSPEND state.

   Avoiding SUSPEND generally means keeping the bus active. However, for the "embedded" host, forcing the LSI into a disconnect state in software rather than keeping it connected in a SUSPEND state may improve system power savings and control. The S1R72V05 is designed so that the LSI can be placed into power-save mode (e.g., SNOOZE or SLEEP) if disconnected in software.

   If the LSI needs to be placed into a disconnect state in software by the host, this can be done by turning off power for the VBUS. Execute a state transition to H_NegoControl_0.HostState = "IDLE." This negates output at the VBUSEN pin.

   To enter the IDLE state, follow the procedure given below.

   - Set H_NegoControl_0.AutoModeCancel = 1.

   To do this, write "0x80" to H_NegoControl_0.

   - Wait until H_NegoControl_0.AutoModeCancel = 0.

   This change may take up to 100 ns.

   - Set H_NegoControl_0.AutoMode = "GoIDLE (0b0001)".

2. Go to a SUSPEND state after forcing the LSI into FS mode in firmware.

The disconnect detection function can be enabled for use during a SUSPEND state by rewriting the value of H_NegoControl_1.PortSpeed to FS before placing the USB bus into SUSPEND state.

When doing this, however, keep in mind that supporting Remote Wakeup from a USB device means subjecting firmware processing to time constraints. When Remote Wakeup is detected, the hardware issues a RESUME signal. PortSpeed must be rewritten to "HS (0b00)" during this hardware processing.

Do the following to enter a SUSPEND state:

- Terminate issuance of all transactions.

Do not issue any new transactions.

- Set H_Protect.TranEnb = "STOP (0b01)."

This stops issuance of SOF transactions. Write "0x01" to H_Protect.

- Wait until H_Monitor.TranRunning = "0."

Wait until issuance of SOF transactions stops. This may take up to 1 us.

- Set H_Protect.PortSpeedWrEnb = "1."

This permits PortSpeed to be rewritten. Access this register in read-modify-write mode to avoid rewriting the other bits.

- Set H_NegoControl_1.PortSpeed = "FS (0b01)."
- Clear H_Protect.PortSpeedWrEnb = "0."

Access this register in read-modify-write mode to avoid rewriting the other bits.

- Set H_NegoControl_0.AutoMode = "GoSUSPENDtoOP (0b1110)."

Disconnect detection is performed as described below.

- A device-disconnected interrupt (H_SIE_IntStat.DetectDiscon) is generated.
- Set H_NegoControl_0.AutoModeCancel = "1."

This clears settings for GoSUSPENDtoOP. Write "0x80" to H_NegoControl_0.

- Wait until H_NegoControl_0.AutoModeCancel = "0."

This change may take up to 100 ns.

- Set H_NegoControl_0.AutoMode = "GoWAIT_CONNECTtoDIS (0b1001)."

The LSI waits until a connection is detected.

To perform RESUME, do the following:

- Set H_Protect.PortSpeedWrEnb = "1."

Do this in read-modify-write mode.

- Set the operation speed in H_NegoControl_1.PortSpeed.
- Clear H_Protect.PortSpeedWrEnb = “0.”

Do this in read-modify-write mode.

- Set H_NegoControl_0.AutoModeCancel = “1.”

This clears settings for GoSUSPENDtoOP. Write “0x80” to NegoControl_0.

- Wait until H_NegoControl_0.AutoModeCancel = “0.”

This change may take up to 100 ns.

- Set H_NegoControl_0.AutoMode = “GoRESUMEtoOP (0b1111).”

To respond to Remote Wakeup, do the following:。

- A Remote Wakeup detected interrupt (H_SIE_IntStat_0.DetectRmWkup) is generated.

A Remote Wakeup is detected.

- Set H_Protect.PortSpeedWrEnb = “1.”

Do this in read-modify-write mode.

- Set the operation speed in H_NegoControl_1.PortSpeed.
- Clear H_Protect.PortSpeedWrEnb = “0.”

Do this in read-modify-write mode.

When responding to Remote Wakeup, make sure the above processing is executed within 20 ms after Remote Wakeup detection.

To perform RESET during RESUME, do the following:

- Set H_NegoControl_0.AutoModeCancel = “1.”

This clears settings for GoRESUMEtoOP. Write “0x80” to NegoControl_0.

- Wait until H_NegoControl_0.AutoModeCancel = “0.”

This change may take up to 100 ns.

- Set H_NegoControl_0.AutoMode = “GoRESETtoOP (0b1100).”

The rest of the procedure is identical to the conventional procedure.

3. Monitor LineState in firmware.

   Disconnections can be detected by polling H_USB_Status.LineState.

   The following items are monitored if the USB device remains connected during SUSPEND:

   - H_USB_Status.LineState == "J (0b01)"

   The following items are monitored if the USB device is disconnected during SUSPEND:

   - H_USB_Status.LineState == "SE0 (0b00)"

Of the registers included in the above procedures, note that the following are defined for the first time in Technical Manual Rev 1.10.

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	1F5h	H_Protect		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
			R / W	3: PortSpeedWrEnb	Enable to replace PortSpeed		
				2:	0:	1:	
			R / W	1: TranEnb[1]	Transaction Control		
			R / W	0: TranEnb[0]			

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Host	1F6h	H_Monitor		7:			00h
				6:			
				5:	0:		
				4:	0:		
				3:	0:		
				2:	0:		
				1:	0:		
			R	0: TranRunning	Monitor transaction		

1F5h.Bit7-4Reserved

1F5h.Bit3 PortSpeedWrEnb

This bit enables writing to H_NegoControl_1.PortSpeed.

1F5h.Bit2 Reserved

1F5h.Bit1-0TranEnb

This bit causes SOF transmission to stop. Set this bit before writing to H_NegoControl_1.PortSpeed.

1F6h.Bit7-1Reserved

1F6h.Bit0 TranRunning

This bit monitors whether SOF transmission is underway or not.

Appendix E About Responses to a SetAddress Request

If a request in which bmRequestType = not 0 (standard request) and bRequest = 0x05 is received, no RcvEP0SETUP interrupt status is issued.

This problem is attributable to the fact that since the automatic address setup function automatically processes a SetAddress request (bmRequestType==0, bRequest==0x05), the RcvEP0SETUP interrupt status is masked with the bRequest value.

To resolve this problem, do one of the following.

1. Limit vendor and class requests.
   Unless a vendor or a class request in which bRequest==0x05 is used, no particular measures need to be taken.
2. Disable the automatic address setup function.
   This problem can be solved by disabling the automatic address setup function. In this case, the function for automatically executing a status stage after receiving a SetAddress request is disabled, so that the status stage of a received SetAddress request must be executed in firmware as for other requests. However, part of the automatic address setup function may be used to automate D_USB_Address register setup.
   The following shows how to disable the automatic address setup function and describes a control sequence for cases when the automatic address setup function is disabled. For comparison, the control sequence is described for cases in which the automatic address setup function is enabled.

<Process for disabling the automatic address setup function>

Event/process	Automatic address setup function = enabled	Automatic address setup function = disabled
(1) Disabling the automatic address setup function	-	The firmware sets D_ModeControl.SetAddressMode = 1.

(1) Disabling the automatic address setup function
Set D_ModeControl.SetAddressMode = 1.
Once this bit is set after the chip reset, it does not need to be set again thereafter.

<Processing a SetAddress request>

Event/process	Automatic address setup function = enabled	Automatic address setup function = disabled
(1) SetAddress request received	-	The hardware issues an RcvEP0SETUP interrupt status.
(2) Checking the request	-	The firmware checks EP0SETUP0 and EP0SETUP1 for confirmation.
(3) Instructing address setup	-	The firmware makes the setting USB_Address.SetAddress = 1.

(4) Preparing a status stage response	-	The firmware sets the following:
		D_SETUP_Control.ProtectEP0 = 0
		D_EP0Control.INxOUT = 1
		D_EP0ControlIN = 0x40*
		* ForceNAK = 0, EnShortPkt = 1

(5) Status stage executed	The hardware issues a SetAddressCmp interrupt status.	The hardware issues a SetAddressCmp interrupt status.

(1) SetAddress request received

Upon receiving a SetAddress request, the hardware issues an RcvEP0SETUP interrupt status. Since the automatic address setup function is disabled, this status indicates the receipt of a SETUP transaction for even a SetAddress request, just as for other requests.

(2) Checking the request

The firmware checks the contents of D_EP0SETUP0 and 1 registers* to determine the values of bmRequestType and bRequest. If bmRequestType==0 and bRequest==0x05, a SetAddress request is assumed.

* In register definitions of the S1R72V03, this is “RcvEP0SETUP.”

(3) Instructing address setup

The firmware sets USB_Address.SetAddress = 1.

When a completed status stage is executed after this register setting, the hardware writes the address indicated in the SetAddress request to the USB_Address register over the existing address. It also indicates the completion of this operation by means of a SetAddressCmp interrupt status.

(4) Preparing a status stage response

The firmware executes a process for returning a zero-length packet as in an IN-direction status stage for other requests.

- D_SETUP_Control.ProtectEP0="0"
- D_EP0Control.INxOUT="1"
- D_EP0ControlIN="0x40"(ForceNAK="0", EnShortPkt="1")

(5) Status stage executed

When a status stage (IN transaction) is executed, the hardware issues a SetAddressCmp interrupt status.

Of the registers included in the above procedure, the “D_ModeControl” register shown below has been additionally defined in Development Specifications Rev. 1.20. Bit 7 of the D_USB_Address register is also defined in Development Specifications Rev. 1.20.

Mode	Address	Register Name	R / W	Bit Symbol	Description		Reset
Device	1E1h	D_ModeControl	W	7: (Reserved)	Don't set "1"		XXh
			W	6: (Reserved)	Don't set "1"		
			W	5: (Reserved)	Don't set "1"		
			W	4: SetAddressMode	0: Auto mode	1: Manual mode	
			W	3: (Reserved)	Don't set "1"		
			W	2: (Reserved)	Don't set "1"		
			W	1: (Reserved)	Don't set "1"		
			W	0: (Reserved)	Don't set "1"		

Bit7-5 Reserved

Bit4 SetAddressMode

Disables the automatic address setup function.

Bit3-0 Reserved

Revision History

Date	Content of Revision			
	Rev.	Page or section	Category	Content
04/11/30	0.79	All pages	New	Newly created.
05/02/28	0.80	General	Correction	Register names changed.
		Chapter 2	Correction	Number of channels available for the USB host function reduced (by 6) and transfer types changed (isochronous transfer unsupported).
			Addition	Media data transfer function added. Package types stipulated.
		Chapter 3	Correction	Port Selector function changed.
			Addition	Media FIFO and Media FIFO Controller added.
		Chapter 4	Addition	QFP package pin assignment diagram stipulated.
		Chapter 5	Addition	Pin numbers stipulated. 5 V tolerant notes added.
		Section 6.3	Correction	USB host control general; isochronous transfer related items deleted.
		Section 6.3.1.1	Correction	Number of channels reduced (by 6) and transfer types changed (isochronous transfer unsupported).
		Section 6.3.1.4	Correction	Example channel usage changed.
		Section 6.3.3	Correction	Transaction status changed.
		Section 6.3.4.3	Correction	Specifications of the control transfer support function changed.
		Section 6.3.8	Correction	Specifications of the bulk only support function changed.
		Section 6.3.9.1	Correction	DISABLED state added to the host state transition diagram.
		Section 6.3.9.1.1 Section 6.3.9.3.1	Correction	IDLE state transition execution procedure changed.
		Section 6.3.9.1.3 Section 6.3.9.3.3	Addition	DISABLED state added.
		Section 6.3.9.1.4 Section 6.3.9.3.4.2	Addition	Operation mode added which is to be assumed when Chirp from an erratic device is detected.
		Section 6.3.9.3.11	Correction	Behavior and processing of the RmtWkupDetEnb bit in case of GoSUSPENDtoOP explicitly stated.
		Section 6.3.9.3.2.1 Section 6.3.9.3.2.2	Correction	Error in writing of the PortSpeed change point corrected.
		Section 6.3.9.3.8	Addition	GoWAIT_CONNECTtoDIS added.
		Section 6.4	Addition	Media data transfer function added.

		Section 6.6.1.1	Correction	Device FIFO management; FIFO area settings for EPa-c changed to be variable.
		Section 6.6.2.1	Correction	Host FIFO management; host FIFO memory map changed (fixed areas are only CBW and CSW) and PingPong mode removed.
		Section 6.6.3	Addition	Media FIFO management added.
		Chapter 7	Correction	Register names and register mapping entirely changed.
		Section 8.3	Addition	DC characteristics stipulated (partly TBD).
		Section 10.1	Addition	QFP package dimensions stipulated.
05/04/28	0.90	Chapter 4	Correction	QFP pin layout diagram corrected.
			Addition	BGA pin layout diagram stipulated.
		Chapter 5	Correction	Pin names for QFP and BGA corrected. Explanation for pin processing while JTAG function is left unused is stated.
		Section 6.1 Section 6.1.1 Section 6.1.2 Section 6.1.2.1.1 Section 6.1.2.1.2	Correction	Description corrected. Items for Table 6-2 moved to Tables 6-3 and 6-4.
		All Section 6.3	Correction	Token type changed to Transaction type. Register bit PID changed to TID.
		Section 6.3.4	Correction	Figure 6-30 corrected.
		Section 6.3.4.3	Correction	Figure 6-33 corrected. TranRetry in Table 6-26 corrected to TranErr.
		Section 6.3.8	Correction	Figures 6-34, 6-35 corrected. Misplaced description in (10) corrected.
		Section 6.3.9.1	Correction	State diagram changed.
		6.3.9.1.2 6.3.9.3.2.1 6.3.9.3.2.2	Correction	Connection detection procedure for WAIT_CONNECT state changed.
		Section 6.3.9.1.4	Correction	Procedure for stopping transaction upon entry to RESET state changed.
		Section 6.3.9.2.1	Correction	Description for T1 changed.
		Section 6.3.9.2.2 Section 6.3.9.2.2.2	Correction	WAIT_CONNECT state excluded from status for disconnection detection.
		Section 6.3.9.2.3.1 Section 6.3.9.2.3.2 Section 6.3.9.2.3.3	Correction	T4 deleted.
		Section 6.3.9.3.4	Correction	Description for the process involving transition from OPERATIONAL to RESET added.
		Section 6.3.9.3.4.2.2	Correction	Procedure to transition to DISABLED state prior to issuance of the ResetCmp status added.
		Section 6.3.9.3.4.3.2 Section 6.3.9.3.4.3.3	Correction	Misplaced Value items in the table corrected.
		Section 6.3.9.3.6.1 Section 6.3.9.3.6.2 Section 6.3.9.3.6.3	Correction	Misplaced Value items in the table corrected.

		Section 6.3.9.3.7.1 Section 6.3.9.3.7.2 Section 6.3.9.3.7.3	Correction	Misplaced Value items in the table corrected.
		Section 6.3.9.2.5	Insertion	Section on port error detection added.
		Section 6.3.9.1.1 Section 6.3.9.2.1 Section 6.3.9.3.1 Section 6.3.9.3.4.2.2	Correction	Description of time required for stopping process of AutoModeCancel corrected.
		Section 6.8.4.1 Section 6.8.4.2	Correction	Description stating writing the initial values to IDE_CRC_L/H register deleted.
		Section 6.9.1	Addition	CLAMP code value 0011 stated. HIGHZ code value 0100 stated.
		Section 6.9.2	Correction	PartNumber value corrected from 000E to 000F.
		Section 6.9.3	Correction	Boundary scan exclusion pins DP, DM, and VBUS corrected toDP_A, DM_A, DP_B, DM_B, and VBUS_B.
		Chapter 7	Correction	Description of explanation corrected.
		Section 7.1 Section 7.2	Correction	7.1. Device Register Map and 7.2. Host Register Map changed to 7.1. Device/Host Shared Register Map, 7.2. Device Register Map, and 7.3. Host Register Map. Names of registers and register bits corrected. Correction made in reference to addition and deletion of register and register bits.
		Section 7.3.2 Section 7.3.3 Section 7.3.9 Section 7.3.10	Correction	Description of Device mode and Host mode in registers 01h, 02h, 11h, 12h each fixed to Host/Device mode.
		Section 7.3.3	Correction	Cautionary remark added to LineStatusChanged in the 02h register.
		Section 7.3.6	Addition	MediaIDE_Cmp added to 05h register.
		Section 7.3.13	Addition	EnMediaIDE_Cmp added to 15h register.
		Section 7.3.18	Correction	Cautionary remark added to 23h register concerning reference of PM_State[3:0].
		Section 7.3.22	Addition	RemoveRPD added to Bit6 of the 27h register.
			Correction	Reset value for 27h register corrected from 11h to 91h.
		Section 7.3.23	Correction	Remark added to FSxHS in the 28h register concerning cable attachment. Description on readable and writable states added.
		Section 7.3.30~ Section 7.3.33 Section 7.3.36 Section 7.3.37	Correction	Bit names misplaced in registers 34h-37h, 40h, 41h corrected.
		Section 7.3.34	Correction	Misplaced register name in 38h register corrected.
		Section 7.3.39	Correction	Limitation in number of bits set added to description of 43h register.
		Section 7.3.45 Section 7.3.47	Correction	ClrAllMediaFIFO_Join register moved from 4Bh to 49h and 4Bh changed toReserved.
		Section 7.3.49~ Section 7.3.80	Correction	Bit names misplaced in registers 50h-4Fh corrected.
		Section 7.3.82 Section 7.3.97	Correction	Logic and description for the ActiveDMA of registers 71h and 81h corrected.
		Section 7.3.114	Correction	SWAP in Bit2 of the 93h register deleted.

		Section 7.3.141	Correction	ClkSelect.ClkSelect added as a register to be protected by ModeProtect in the B3h register.
		Section 7.3.143	Correction	xActIDE_Term of the B5h register corrected to asynchronous and description added.
		Section 7.3.145	Correction	Description of CPU_Swap bit in the B7h register corrected.
		Section 7.4.1	Correction	Interrupt status processing upon exiting the ACT_DEVICE explained.
		Section 7.4.11	Correction	Interrupt status processing upon exiting the ACT_DEVICE explained.
		Section 7.4.12	Correction	Logic for ResetDTM in the 100h register changed to “1: ResetDTM.”
		Section 7.4.5 Section 7.4.15	Correction	Bit names for registers E4h and F4h corrected.
		Section 7.4.46 Section 7.4.47	Correction	Reset values in registers 11Eh and 11Fh corrected.
		Section 7.4.55	Correction	Reset values in 130h register corrected.
		Section 7.5.1	Correction	Register name for E0h register corrected. Interrupt status processing upon exiting the ACT_HOST explained.
		Section 7.5.13	Correction	Interrupt status processing upon exiting the ACT_HOST of the F0h register explained.
		Section 7.5.2	Correction	Register name for E1h register corrected. Interrupt status processing upon exiting the ACT_HOST explained.
			Addition	DisabledCmp added to Bit3.
		Section 7.5.14	Correction	Register name for F1h register corrected. Bit0 EnWaitConCmp deleted. EnResetCmp moved from Bit1 to Bit0. EnSuspendCmp moved from Bit2 to Bit1. EnResumeCmp moved from Bit3 to Bit2. Interrupt status processing upon exiting the ACT_HOST explained.
			Addition	EnDisabledCmp added to Bit3.
		Section 7.5.5	Correction	Bit name for E4h register corrected.
		Section 7.5.6	Correction	Description for TotalSizeCmp in the E5h register corrected. Description for TranAck corrected. Description within the ConditionCode table of ChangeCondition corrected. Description for CTL_SupportCmp corrected. Description for CTL_SupportStop corrected.
		Section 7.5.7~ Section 7.5.11	Correction	Bit name for Bit7 of registers E6h-EAh corrected from TranCmp to TotalSizeCmp.
		Section 7.5.7	Correction	Description for TotalSizeCmp in the E6h register corrected. Description for TranAck corrected. Description for BO_SupportCmp corrected. Description for BO_SupportStop corrected.
			Addition	ConditionCode table added for description of ChangeCondition.
		Section 7.5.8~ Section 7.5.11	Addition	ConditionCode table added to the description of ChangeCondition.
		Section 7.5.17	Correction	Bit name for F4h register corrected.

		Section 7.5.19~ Section 7.5.23	Correction	Bit name for Bit7 of registers F6h-EAh corrected from EnTranCmp to EnTotalSizeCmp.
		Section 7.5.25	Correction	Misplaced name of ResetDTM of the 100h register corrected to ResetHTM. Logic of ResetHTM changed to “1:ResetHTM.” Misplaced Device Transceiver Macro corrected to Host Transceiver Macro.
		Section 7.5.27	Correction	Register name for 102h register corrected. Description for AutoModeCancel corrected. Description for AutoMode corrected. Reset values corrected.
		Section 7.5.28	Correction	Register name for 103h register corrected. Description for DisChirpFinish corrected.
		Section 7.5.31	Correction	Bit name for Bit4 of 106h register corrected from TestForceEnable to Test_Force_Enable.
		Section 7.5.33	Correction	Register name for 108h register corrected.
		Section 7.5.46 Section 7.5.60	Correction	Reset values in registers 120h and 130h corrected. Description for ACK_Cnt[3:0] corrected. Description for Toggle corrected. Description for TranGo corrected.
		Section 7.5.76 Section 7.5.91 Section 7.5.106 Section 7.5.121	Correction	Reset values in registers 140h, 150h, 160h, 170h corrected. Description for ACK_Cnt bit corrected.
		Section 7.5.47 Section 7.5.61 Section 7.5.77 Section 7.5.92 Section 7.5.107 Section 7.5.122	Correction	Bit names in Bits7-6 of registers 121h, 131h, 141h, 151h, 161h, 171h corrected from PID bit to TID bit. Description for the above bits corrected.
		Section 7.5.53 Section 7.5.54	Correction	Description for TotalSize[15:0] of registers 128h and 1291h corrected.
		Section 7.5.50~ Section 7.5.54	Correction	Address for registers 124h-129h changed.
		Section 7.5.64~ Section 7.5.69	Correction	Address for registers 134h-139h changed.
		Section 7.5.80~ Section 7.5.84	Correction	Address for registers 144h-149h changed.
		Section 7.5.95~ Section 7.5.100	Correction	Address for registers 154h-159h changed.
		Section 7.5.110~ Section 7.5.115	Correction	Address for registers 164h-169h changed.
		Section 7.5.125~ Section 7.5.130	Correction	Address for registers 174h-179h changed.
		Section 7.5.56	Correction	Reset values in the 12Bh register corrected. CTL_SupportState corrected from R register to R/W register. Description for CTL_SupportGo corrected.
		Section 7.5.58 Section 7.5.74 Section 7.5.89 Section 7.5.104 Section 7.5.119 Section 7.5.134	Correction	Description within table of ConditionCode[2:0] for registers 12Eh, 13Eh, 14Eh, 15Eh, 16Eh, 17Eh corrected.
		Section 7.5.62 Section 7.5.63	Correction	Description for MaxPktSize[9:0] of registers 132h and 133h corrected.

		Section 7.5.65	Correction	Description for EP_Number in the 135h register corrected.
		Section 7.5.66~ Section 7.5.69	Correction	Description for TotalSize[31:0] of registers 136h-139h corrected.
		Section 7.5.70	Correction	Register description for 13Ah register corrected. Description for BO_TransportState[1:0] corrected. Description for BO_SupportGo corrected.
		Section 7.5.71	Correction	Register description for 13Bh register corrected. Description for CSW_RcvDataSize[3:0] corrected.
		Section 7.5.72	Correction	Register description for 13Ch register corrected. Description for OUT_Toggle corrected. Bit width and description for the OUT_EP_Number register corrected.
		Section 7.5.73	Correction	Register description for 13Dh register corrected. Description for IN_Toggle corrected. Bit width and description for the IN_EP_Number register corrected.
		Section 7.5.77 Section 7.5.92 Section 7.5.107 Section 7.5.171	Correction	ToggleMode in registers 141h, 151h, 161h, 171h deleted.
		Section 7.6.86	Correction	Register name of 14Ah register corrected.
		Section 7.6.87	Correction	Register name and misplaced address of 14Bh register corrected.
		Section 8.2	Correction	Power engagement procedure stated.
		Section 8.3	Correction	Input leakage stated.
		Section 8.4.1	Correction	Numerical value stated for CPU/DMA I/F access timing (Preliminary).
		Section 8.4.1 Section 8.4.2	Insertion	RESET Timing, Clock Timing added. Succeeding section numbers adjusted due to insertion of a section.
		Section 9.2	Addition	USB I/F Connection Example stated through 9.2.1-9.2.4 (Preliminary).
		Section 10.2	Addition	BGA Package diagram stipulated.
05/12/02	0.91	Section 3	Correction	TDO direction changed from input to output.
		Section 4, Section 5, Section 8.1, Section 8.2, Section 8.3, Section 9.2	Correction	PIN names changed from PVDD to LVDD and PVSS to VSS
		Section 5	Correction	XRESET pin category changed from GENERAL to CPU I/F
			Addition	Notes stipulated regarding tristate of the XINT pin
			Correction	Termination of the HINTRQ pin changed from (PU) to (PD)
			Correction	Termination of the HDD7 pin changed from (PU) to (PD)
			Correction	RESET values stipulated for the IN and HDD7 pins of the IDE I/F
			Addition	Note on PU and PD of the IDE I/F pins added
			Correction	5 V tolerant pins limited to the IDE I/F.
			Correction	PU and PD described in the I/O column moved to the Pin Type column.

		Section 6.3~6.3.7	Correction	Overall, corrected written errors involving channels specified by CHx (0,a-e).
		Section 6.3.4.3	Correction	Corrections made due to changes in control method for the control transfer support function.
		Section 6.3.8	Correction	Corrections made due to changes in the control method of the bulk-only support function.
		Section 6.5	Correction	PM_Control1 -> PM_Control_1
		Section 6.5 Section 6.5.1 Section 6.5.2 Section 7.5.1	Correction	PM_Control0 -> PM_Control_0
		Section 6.6.1.2.1 Section 6.6.1.4.2 Section 6.6.1.5.2 Section 6.6.2.2.2 Section 6.6.2.4.2 Section 6.7.2.1.3 Section 7.4.40 Section 7.4.41	Correction	RAM_WrDoor_H,L -> RAM_WrDoor_0,1
		Section 6.6.1.5.3 Section 6.6.2.4.3 Section 6.6.3.2.1 Section 6.7.2.1.4 Section 6.7.2.1.5 Section 7.4.46 Section 7.5.53 Section 7.5.60 Section 7.5.67 Section 7.5.74 Section 7.6.61 Section 7.6.77 Section 7.6.92 Section 7.6.107 Section 7.6.122 Section 7.6.137	Correction	FIFO_Rd_H,L -> FIFO_Rd_0,1
		Section 6.6.1.5.3 Section 6.6.2.4.3 Section 6.6.3.2.1 Section 6.7.2.1.3 Section 7.4.46 Section 7.5.53 Section 7.5.60 Section 7.5.67 Section 7.5.74 Section 7.6.61 Section 7.6.77 Section 7.6.92 Section 7.6.107 Section 7.6.122 Section 7.6.137	Correction	FIFO_Wr_H,L -> FIFO_Wr_0,1
		Section 6.8.1.4	Correction	IDE_RdRegValue_L -> IDE_RdRegValue_1
		Section 7.1	Correction	H_FIFOI_ntStat -> H_FIFO_IntStat
		Section 7.1 Section 7.4.6	Correction	Bit positions and bit names changed for the MediaFIFO_IntStat register: bit5 Full -> bit1FIFO_Full, bit4 Empty -> bit0 FIFO_Empty
			Addition	FIFO_NotEmpty bit added to the MediaFIFO_IntStat register

		Section 7.1 Section 7.4.13	Correction	Bit positions and bit names changed for the MediaFIFO_IntEnb register: bit5 EnFull -> bit1 EnFIFO_Full, bit4 EnEmpty -> bit0 EnFIFO_Empty
			Addition	EnFIFO_NotEmpty bit added to the MediaFIFO_IntStat register
		Section 7.1 Section 7.4.26 Section 7.4.27	Correction	Register names changed: FIFO_Rd_H -> FIFO_Rd_0 FIFO_Rd_L -> FIFO_Rd_1
		Section 7.1 Section 7.4.28 Section 7.4.29	Correction	Register names changed: FIFO_Wr_H -> FIFO_Wr_0 FIFO_Wr_L -> FIFO_Wr_1
		Section 7.1 Section 7.4.42 Section 7.4.43	Correction	Register names changed: RAM_WrDoor_H -> RAM_WrDoor_0 RAM_WrDoor_L -> RAM_WrDoor_1
		Section 7.1 Section 7.4.92 Section 7.4.93	Correction	Register names changed: DMA0_RdData_H -> DMA0_RdData_0 DMA0_RdData_L -> DMA0_RdData_1
		Section 7.1 Section 7.4.94 Section 7.4.95	Correction	DMA0_WrData_H -> DMA0_WrData_0 DMA0_WrData_L -> DMA0_WrData_1
		Section 7.1 Section 7.4.107 Section 7.4.108	Correction	Register names changed: DMA1_RdData_H -> DMA1_RdData_0 DMA1_RdData_L -> DMA1_RdData_1
		Section 7.1 Section 7.4.109 Section 7.4.110	Correction	Register names changed: DMA1_WrData_H -> DMA1_WrData_0 DMA1_WrData_L -> DMA1_WrData_1
		Section 7.1 Section 7.4.127 Section 7.4.128	Correction	Register names changed: IDE_RdRegValue_H -> IDE_RdRegValue_0 IDE_RdRegValue_L -> IDE_RdRegValue_1
		Section 7.1 Section 7.4.129 Section 7.4.130	Correction	Register names changed: IDE_WrRegValue_H -> IDE_WrRegValue_0 IDE_WrRegValue_L -> IDE_WrRegValue_1
		Section 7.1 Section 7.4.114	Addition	Swap bit added to IDE_Config_1 register
			Correction	Reset value of IDE_Config_1 changed: 0x00 -> 0x04
		Section 7.1 Section 7.4.143	Addition	xActIDE_DD_Term and PORT1x2 bits added to ClkSelect register
			Correction	Reset value of ClkSelect register changed: 0x01 -> to 0x41
		Section 7.1 Section 7.4.145	Correction	Bit name of ChipConfig register changed: CPU_Swap -> CPU_Endian
		Section 7.2 Section 7.5.3	Addition	FIFO_NotEmpty bit added to D_FIFO_IntStat register
		Section 7.2 Section 7.5.13	Addition	EnFIFO_NotEmpty bit added to D_FIFO_IntEnb register
		Section 7.2 Section 7.5.53	Addition	Join_FIFO_Stat bit added to D_EP0Join register
		Section 7.3	Correction	Initial value of the H_NegoControl_0 register changed: 0x00 -> 0x1X
		Section 7.3 Section 7.6.3	Addition	FIFO_NotEmpty bit added to H_FIFO_IntStat register
		Section 7.3 Section 7.6.15	Addition	EnFIFO_NotEmpty bit added to H_FIFO_IntEnb register
		Section 7.3 Section 7.6.28	Correction	0x103 register changed to Reserved following address change for H_NegoControl_1 register.

		Section 7.3 Section 7.6.29	Correction	Address changed for H_NegoControl_1 register: 0x103 -> 0x104
		Section 7.3 Section 7.6.29 Section 7.6.46 Section 7.6.47	Addition	Notes added on Reset values for the H_NegoControl_1, H_FrameNumber_H, and H_FrameNumber_L registers.
		Section 7.3 Section 7.6.35	Correction	Address changed for H_ClrAllHnJoin register: 0x104 -> 0x10A
		Section 7.3 Section 7.6.61	Addition	Join_FIFO_Stat bit added to H_CH0Join register
		Section 7.4.1	Correction	SIE Interrupts -> Device Interrupts
			Correction	EPr Interrupts -> Host Interrupts
			Correction	Device_IntStat -> DeviceIntStat
			Correction	PM_Control register -> PM_Control_0 register
		Section 7.4.1 Section 7.4.3	Correction	Host_IntStat -> HostIntStat
		Section 7.4.125	Correction	IDE_RdRegValue_H,L -> IDE_RdRegValue_0,1 IDE_WrRegValue_H,L -> IDE_WrRegValue_0,1
		Section 7.5.53	Correction	Bit5 JoinCPU_Rd -> Bit1 JoinCPU_Rd Bit4 JoinCPU_Wr -> Bit0 JoinCPU_Wr
		Section 7.5.60	Correction	EPnCHnFIFO_Wr_H,L -> FIFO_Wr_0,1
		Section 7.5.60 Section 7.5.67	Correction	EPnCHnFIFO_ByteRd -> FIFO_ByteRd
		Section 7.5.60 Section 7.5.67 Section 7.5.74	Addition	NotEmpty was added to the description of JoinFIFO_Stat bit.
			Correction	Bit5 JoinCPU_Rd -> Bit1 JoinCPU_Rd Bit4 JoinCPU_Wr -> Bit0 JoinCPU_Wr
		Section 7.6.3	Correction	CH0 was added to the description of FIFO_Full and FIFO_Empty bits.
		Section 7.6.4	Correction	Bit2 FrameNumOver -> Bit1 FrameNumOver
		Section 7.6.15	Addition	EnFIFO_NotEmpty bit
		Section 7.6.36	Addition	Reserved (10Bh -Fh)
		Section 7.6.46 Section 7.6.47	Correction	Reset value changed: 0800h -> 07FFh
		Section 7.6.48 Section 7.6.62 Section 7.6.93 Section 7.6.108 Section 7.6.123	Correction	4:ACK_Cnt[1] -> 4:ACK_Cnt[0]
		Section 7.6.55 Section 7.6.70 Section 7.6.86 Section 7.6.101 Section 7.6.116 Section 7.6.131	Correction	Bit7 HubAdrs[3:0] -> Bit7-4 HubAdrs[3:0]
		Section 7.6.56 Section 7.6.71 Section 7.6.87 Section 7.6.102 Section 7.6.117 Section 7.6.132	Correction	6:FuncAdrs[3], 5:FuncAdrs[3], 4:FuncAdrs[3] -> 6:FuncAdrs[2], 5:FuncAdrs[1], 4:FuncAdrs[0]
			Correction	Bit5 EP_Number[3:0] -> Bit3-0 EP_Number[3:0]

		Section 7.6.75 Section 7.6.91 Section 7.6.106 Section 7.6.121 Section 7.6.136	Correction	Bit7-4 ConditionCode[2:0] -> Bit7 Reserved Bit6-4 ConditionCode[2:0]
		Section 7.6.77 Section 7.6.92 Section 7.6.107 Section 7.6.122 Section 7.6.137	Addition	NotEmpty was added to the description of JoinFIFO_Stat bit.
		Section 7.6.143	Correction	185h.Bit2-0 -> 185h.Bit1-0
		Section 7.6.149	Correction	18Bh.Bit2-0 -> 18Bh.Bit1-0
		Section 7.6.153	Correction	18Fh.Bit2-0 -> 18Fh.Bit1-0
		Section 7.6.157	Correction	193h.Bit2-0 -> 193h.Bit1-0
		Section 7.6.159	Correction	195h.Bit2-0 -> 195h.Bit1-0
		Section 7.6.161	Correction	197h.Bit2-0 -> 197h.Bit1-0
			Correction	EndAdra[5], EndAdrs[4], EndAdrs[3], and EndAdrs[2] stipulated in the table.
		Section 8.1	Deletion	UVI, UVO
			Addition	CVI, IVI, CVO
		Section 8.2	Correction	CVDD
			Deletion	PVDD
			Addition	CVI, UVI
		Section 8.3	Correction	Tables renewed overall.
		Section 8.4.3	Correction	AC specifications stipulated (trcy, tras, trdf, trdh, trbh, twcy, twas, and tdm each corrected).
		Section 9.2.1~9.2.4	Correction	Figures changed overall.
		Appendix A	Addition	IDE_Config_1.Swap bit settings
		Appendix B	Addition	Connecting to little endian CPUs
		Appendix C	Addition	1-Port Mode
06/02/24	1.0	Section 5	Correction	GENERAL -> OSC
			Correction	CA1: I/O, RESET, and Pin Type stipulated.
			Correction	In Pin Description of LVDD, power supply for OSC I/O added.
		Section 6.2.7.11.5 Section 6.2.7.11.5.3 Section 6.2.7.11.6 Section 6.2.7.11.7 Section 6.2.7.11.8	Correction	Errors corrected: PM_Control.GoActiveALL -> PM_Control_0.GoActDevice
		Section 6.2.7.11.5.3 Section 6.2.7.11.6 Section 6.2.7.11.7 Section 6.2.7.11.8	Correction	Errors corrected: ACTIVEALL -> ACT_DEVICE GoActiveALL -> GoActDevice PM_Control.PM_State[2:0] -> PM_Control_1.PM_State[2:0]
		Section 6.3.9.1.5	Addition	Note added on recommended command for successive transition.
		Section 6.5.3 Section 6.5.4 Section 6.5.5	Correction	Errors corrected involving register access for registers other than those shown in italics.
		Section 6.5.3 Section 7.1	Addition	Note added on registers 0x40-0x47 and 0x50-0x6F.
		Section 6.7.1	Correction	Error corrected: CPU_BusSwap -> CPU_Endian

			Addition	Added “The Swap function is enabled by reading the address B9h after setting the CPU_Endian bit.”
		Section 6.7.2	Addition	Added “Notes on Mode Switching.”
		Section 6.7.2.1	Addition	Added “When Using 16-bit BE Mode.”
		Section 6.7.2.2	Addition	Added “When Using 8-bit Mode.”
		Section 7.1	Correction	Error corrected: In R/W field of the 0x90 IDE_Status, R/E changed to R.
		Section 7.1 Section 7.4.26 Section 7.4.27	Correction	Bit symbol names changed: FIFO_Rd[15:8] -> FIFO_Rd_0[7:0] FIFO_Rd[7:0] -> FIFO_Rd_1[7:0]
		Section 7.1 Section 7.4.28 Section 7.4.29	Correction	Bit symbol names changed: FIFO_Wr[15:8] -> FIFO_Wr_0[7:0] FIFO_Wr[7:0] -> FIFO_Wr_1[7:0]
		Section 7.1 Section 7.4.42 Section 7.4.43	Correction	Bit symbol names changed: RAM_WrDoor[15:8] -> RAM_WrDoor_0[7:0] RAM_WrDoor[7:0] -> RAM_WrDoor_1[7:0]
		Section 7.1 Section 7.4.92 Section 7.4.93 Section 7.4.107 Section 7.4.108	Correction	Bit symbol names changed: DMA_RdData[15:8] -> DMA_RdData_0[7:0] DMA_RdData[7:0] -> DMA_RdData_1[7:0]
		Section 7.1 Section 7.4.94 Section 7.4.95 Section 7.4.109 Section 7.4.110	Correction	Bit symbol names changed: DMA_WrData[15:8] -> DMA_WrData_0[7:0] DMA_WrData[7:0] -> DMA_WrData_1[7:0]
		Section 7.1 Section 7.4.127 Section 7.4.128	Correction	Bit symbol names changed: IDE_RdRegValue[15:8] -> IDE_RdRegValue_0[7:0] IDE_RdRegValue[7:0] -> IDE_RdRegValue_1[7:0]
		Section 7.1 Section 7.4.129 Section 7.4.130	Correction	Bit symbol names changed: IDE_WrRegValue[15:8] -> IDE_WrRegValue_0[7:0] IDE_WrRegValue[7:0] -> IDE_WrRegValue_1[7:0]
		Section 7.1 Section 7.4.147 Appendix B	Addition	Error corrected: In Reset field of the 0x12B H_CTL_SupportControl, 0xXX changed to 0x00.
		Section 7.3	Correction	Error corrected: In Reset field of the 0x12B H_CTL_SupportControl, 0xXX changed to 0x00.
			Correction	Error corrected: Bit position error corrected: For 0x13A H_CL_SupportControl.BO_TransporState[1:0], bit 13 changed to bits 13-12.
			Correction	Error corrected: Bit position error corrected: For 0x13A H_CL_SupportControl.BO_BO_SupportGo, bit 9 changed to bit 8.
			Correction	Error corrected: In R/W field of the 0x13B H_CSW_RcvDataSze, R/W changed to R.
			Correction	Error corrected: In 0x171 H_CheConfig_1, Toggle Mode deleted.

		Section 7.4.3	Correction	Errors corrected: H_SIE_IntStat1 -> H_SIE_IntStat_1, H_SIE_IntStat0 -> H_SIE_IntStat_0 H_SIE_IntEnb1 -> H_SIE_IntEnb_1, H_SIE_IntEnb0 -> H_SIE_IntEnb_0
		Section 7.4.10	Correction	Errors corrected: EnH_SIE_IntStat1 -> EnH_SIE_IntStat_1, EnH_SIE_IntStat0 -> EnH_SIE_IntStat_0
		Section 7.4.139	Correction	Error corrected: B0h -> B1h
		Section 7.4.143	Correction	Omission of XHPDIAG corrected for the xActIDE_Term controlled pins.
		Section 7.4.146 Section 7.4.148	Correction	Due to changes by which B9h became register-defined, B8h–DFh Reserved() divided into B8h Reserved() and BAh–DFh Reserved().
		Section 7.5.11 Section 7.5.13 Section 7.5.14 Section 7.5.15 Section 7.5.16	Correction	Error corrected: MainIntStat -> DeviceIntStat
		Section 7.6.13 Section 7.6.14 Section 7.6.15 Section 7.6.16 Section 7.6.17 Section 7.6.18	Correction	Error corrected: MainIntStat -> HostIntStat
		Section 7.6.48	Correction	Sentence deleted: "Bits3-2 This setting is not required when using the control transfer support function."
		Section 8.1 Section 8.2	Addition	LVI added to input voltage. *4 added to comments outside the table.
			Correction	Error corrected: *3 VBUS -> *3 VBUS_B
		Section 8.2	Correction	Maximum value for ambient temperature changed from 85 to 110.
		Section 8.3	Correction	IDDC divided into IDDCH and IDDCL in the power supply current. TYP value stipulated.
			Correction	Maximum value stipulated for the static current field.
			Correction	LVIH = LVDD added to the condition of input leakage.
			Correction	Error corrected for the input leakage (5V tolerant) from HVOH to HVIH.
			Correction	Omission of a letter in a pin name corrected for the LVCMOS input characteristics field: XHPDIAG.
		Section 8.3	Correction	Omission of a letter in a pin name corrected for the input characteristics field: VBUSFLG_A.
			Correction	Error corrected: For HDD[15:0]-XHRESET output characteristics, CVDD changed to HVDD.
			Correction	Error corrected: For TOD,VBUSEN_A output characteristics, VOH2 changed to VOH4, and VOL2 to VOL4.
			Correction	Error corrected: For CD[15:0],XINT off-state leakage, HVDD changed to CVDD, and HVOH to CVOH.
			Correction	Error corrected: For HDD[15:0]-XHRESET off-state leakage, LVDD = 1.65V deleted.

			Correction	Maximum value stipulated for pin capacitance.
		Section 9.1	Correction	In 8-bit CPU connection examples, corrected written errors involving DATA[15:8] and DATA[7:0].
		Section 9.2.1 Section 9.2.2 Section 9.2.3 Section 9.2.4	Correction	10μ condenser omitted.
		Appendix C	Addition	D_Reset.ResetDTM and H_Reset.ResetHTM settings stipulated during 1-port mode.
06/04/10	1.1	3.9	Deletion	"Furthermore, it incorporates two DMA channels" was deleted to avoid repeating a description.
		6.3.9.3.4.3 6.3.9.3.4.4 6.3.9.3.5	Correction	Correct) H_USB_Status.PortSpeed[1:0] Incorrect) H_NegoControl_1.PortSpeed[1:0]
		7.1	Correction	0x12 Incorrect) EnH_SIE_IntEnb_1 Correct) EnH_SIE_IntStat_1 Incorrect) En_FrameIntStat Correct) EnH_FrameIntStat 0x14 Incorrect) EnompleteINTRQ Correct) EnCompleteINTRQ
		7.4.30 7.4.31	Deletion	The following passage was deleted due to content errors: "If when the D_EPx{x=0,a-c}Join.JoinCPU_Rr, H_CHx[x=0,a-e]Join.JoinCPU_Rd, or MediaFIFOJoin.JoinCPU_Rd bit is set, no data exists for the relevant endpoint, channel, or media, this bit is not set until data is written to the endpoint, channel, or media from the USB, etc. (not RdRemainValid = 0, however)."
		7.6.1	Addition	Limitations added to discussion of Bit3 DetectDiscon.
		7.6.31	Correction	Incorrect) Set one of the 5 low-order bits in this register to 1 and then EnHS_Test to 1. Correct) Set any of the five low-order bits in this register and EnHS_Test to 1 concurrently. Discussion of the EnHS_Test bit Incorrect) If when this bit is set to 1, any of the 5 low-order bits in the H_USB_Test register is set to 1, the LSI shifts to the test mode corresponding to that bit. Correct) When this bit and any of the five low-order bits in the H_USB_Test register are set to 1 concurrently, the LSI enters the test mode corresponding to that low-order bit. Explanation of the Test_ForceEnable, Test_SE0_NAK, TEST_J, TEST_K, and Test_Packet bits Incorrect) ... by setting this bit to 1 and then the EnHS_Test bit to 1. Correct) ... by setting this bit and the EnHS_Test bit to 1 concurrently.
		7.6.6 7.6.7 7.6.8 7.6.9 7.6.10 7.6.11	Addition	Relationship with H_CHx{x=0,a-e}Config_0.ACK_Cnt bit added to the description of Bit6 TranACK bit.

		9.2.1	Correction	Varistor for DP/DM line protection added.
		9.2.2	Correction	Varistor for DP/DM line protection added.
		9.2.3	Correction	Varistor for DP/DM line protection added.
		9.2.4	Correction	Name for Ball A9 Incorrect) PVDD Correct) LVDD Name of Ball B9 Incorrect) PVSS Correct) VSS Varistor for DP/DM line protection added.
		Appendix C	Correction	In the connection examples, the VBUS control circuit was changed to one using a USB power switch IC with Reverse Current Protection function (the corresponding notes were also revised accordingly). Varistor for DP/DM line protection added.
		7.3 7.6.162 Appendix D	Addition	0x1F5 H_Protect register added. 0x1F6 H_Monitor register added.
06/07/14	1.2	2	Correction	Incorrect) Dual-power supply system Correc) Triple-power supply system
		2	Addition	Package type PFBGA10UX121 added
		3.1	Addition	Description for the clock source of PLL added
		4(Fig.4-2)	Addition	Package type PFBGA10UX121 added Product name S1R72V05B00A2XX added
		5	Correction	Incorrect) Reset value and Pin type of VBUS are not specified Correct) Reset value and Pin type of VBUS are specified as PD.
		6.2.2.2	Correction	Incorrect) return an ACK response Correct) return an ACK or NYET response
		6.6.1.5.6 6.6.2.4.6	Correction	Incorrect) No description for exclusive condition regarding FIFO and Media FIFO Correct) Description for exclusive condition regarding FIFO and Media FIFO added
		6.9.2	Correction	Incorrect) Manufacturer code is specified as 17D Correct) Manufacturer code is specified as 0x0BE
		6.9.3	Correction	Incorret) R1_A and R1_B are not specified as un-scanned Correct) R1_A and R1_B are specified as un-scanned
		6.1.2.1 6.1.2.1.1 6.2.7.11.8 7.4.2 7.4.23	Correction	Incorrect) Pin name "VBUS" Correct) Pin name "VBUS_B"
		7.2 7.5.42	Correction	Incorrect) USB_Address.SetAddress bit is not specified Correct) USB_Address.SetAddress bit is specified on bit 7 of 0x118
		7.2 Appendix B Appendix E	Correction	Incorrect) D_ModeControl is not specified Correct) D_ModeControl is specified on 0x1E1
		7.4.1, 7.4.2, 7.4.3, 7.4.8, 7.4.9, 7.4.10, 7.4.15, 7.4.16, 7.4.18, 7.4.19, 7.4.20, 7.4.21, 7.4.22, 7.4.141, 7.4.143, 7.4.145, 7.4.147, 7.5.1,	Correction	show register and bit names in bold face italic

		7.4.45	Correction	Incorrect) Address 4Ah Correct) Address 49h
		7.4.9 7.4.10	Deletion	Delete one of duplicated tables
		7.6.29	Correction	Incorrect) R/W type of bit PortSpeed[1:0] are specified as R Correct) R/W type of bit PortSpeed[1:0] are specified as R/W
		8.2	Addition	Description of power-off procedure added
		8.3	Correction	Power Supply Current value and condition corrected Quiescent Current value and condition corrected
		8.3	Addition	Adds actual power consumption measurement value from examination
		8.3	Addition	Input Characteristics for Pulldown Resistance on VBUS pin is added
		8.4.2	Deletion	Unspecified marker "*" on tCYC is deleted
		8.4.3	Addition	Adds a timing restriction "tsah"
		8.4.3	Addition	Adds 8.4.3.2 Offer alleviation of some timing ristrictions by limiting I/F voltage
		9.2.2, 9.2.4, Appendix C	Correction	VBUS feed parts symbolized
		9.2.3, 9.2.4	Addition	Show packege types
		10.2	Addition	Show a packege type
		10.3	Addition	Add a packege type
		Appendix B	Addition	H_Protect, H_Monitor register added
		Appendix D	Deletion	Duplicated description deleted
		Appendix E	Addition	Appendix E added
07/11/5	1.3	2	Correction	Operation temperature corrected Incorrect) –40°C to 85°C Correct) –40°C to 110°C
		5	Correction	Pin list corrected Incorrect) Pin67 LVDD, Pin66 VSS Correct) Pin66 LVDD, Pin67, VSS
		5	Correction	CVDD voltage corrected Incorrect) 3.3/1.8V Correct) 3.3-1.8V
		6.2.7.11.1 6.2.7.11.2	Correction	Typo corrected Incorrect) SnoozeUTM Correct) GoSNOOZE
		6.6.1.5.6 6.6.2.4.6 6.6.3.2.4	Addition	Description of Limitations on FIFO Access added
		6.9.2	Correction	Typo corrected Incorrect) INCODE Correct) IDCODE
		8.1	Correction	Typo corrected Incorrect) LVI *3 Correct) LVI *4
		8.3	Correction	Typo corrected Incorrect) ACTIVE60 (Operates IDE to/from USB transfer) Correct) ACTIVE60 (Operates IDE to/from CPU transfer)

		8.3	Correction	Typo corrected Incorrect) HVDD=CVDD=LVDD=V Correct) HVDD=CVDD=LVDD=VSS
		8.4.2	Correction	tCYC values are eased
		8.4.3.2	Correction	trdf, tdrn values corrected Incorrect) max.35ns Correct) max 30ns

EPSON

International Sales Operations

AMERICA

EPSON ELECTRONICS AMERICA, INC.
HEADQUARTERS
2580 Orchard Parkway
San Jose , CA 95131,USA
Phone: +1-800-228-3964 FAX: +1-408-922-0238

SALES OFFICES
Northeast
301 Edgewater Place, Suite 210
Wakefield, MA 01880, U.S.A.
Phone: +1-800-922-7667 FAX: +1-781-246-5443

EUROPE

EPSON EUROPE ELECTRONICS GmbH
HEADQUARTERS
Riesstrasse 15
80992 Munich, GERMANY
Phone: +49-89-14005-0 FAX: +49-89-14005-110

ASIA

EPSON (CHINA) CO., LTD.
23F, Beijing Silver Tower 2# North RD DongSanHuan
ChaoYang District, Beijing, CHINA
Phone: +86-10-6410-6655 FAX: +86-10-6410-7320

SHANGHAI BRANCH
7F, High-Tech Bldg., 900, Yishan Road,
Shanghai 200233, CHINA
Phone: +86-21-5423-5522 FAX: +86-21-5423-5512

EPSON HONG KONG LTD.
20/F., Harbour Centre, 25 Harbour Road
Wanchai, Hong Kong
Phone: +852-2585-4600 FAX: +852-2827-4346
Telex: 65542 EPSCO HX

EPSON Electronic Technology Development (Shenzhen) LTD.
12/F, Dawning Mansion, Keji South 12th Road,
Hi- Tech Park, Shenzhen
Phone: +86-755-2699-3828 FAX: +86-755-2699-3838

EPSON TAIWAN TECHNOLOGY & TRADING LTD.
14F, No. 7, Song Ren Road,
Taipei 110
Phone: +886-2-8786-6688 FAX: +886-2-8786-6660

EPSON SINGAPORE PTE., LTD.
1 HarbourFront Place,
#03-02 HarbourFront Tower One, Singapore 098633
Phone: +65-6586-5500 FAX: +65-6271-3182

SEIKO EPSON CORPORATION
KOREA OFFICE
50F, KLI 63 Bldg., 60 Yoido-dong
Youngdeungpo-Ku, Seoul, 150-763, KOREA
Phone: +82-2-784-6027 FAX: +82-2-767-3677

GUMI OFFICE
2F, Grand B/D, 457-4 Songjeong-dong,
Gumi-City, KOREA
Phone: +82-54-454-6027 FAX: +82-54-454-6093

SEIKO EPSON CORPORATION
SEMICONDUCTOR OPERATIONS DIVISION

IC Sales Dept.
IC International Sales Group
421-8, Hino, Hino-shi, Tokyo 191-8501, JAPAN
Phone: +81-42-587-5814 FAX: +81-42-587-5117

Document Code: 410213504
First Issue November 2007
Printed December 2007 in JAPAN ©