TMS320C672x DSP
Universal Host Port Interface (UHPI)

Reference Guide

TEXAS INSTRUMENTS

Literature Number: SPRU719A

December 2005–Revised March 2009

Contents

List of Figures

List of Tables

Read This First

About This Manual

This document provides an overview and describes the common operation of the universal host port interface (UHPI).

Notational Conventions

This document uses the following conventions:

- Hexadecimal numbers are shown with the suffix h. For example, the following number is 40 hexadecimal (decimal 64): 40h.
- Registers in this document are shown in chapter and described in tables
  - Each register figure shows a rectangle divided into fields that represent the fields of the register. Each field is labeled with its bit name, its beginning and ending bit numbers above, and its read/write properties below. A legend explains the notation used for the properties.
  - Reserved bits in a register figure designate a bit that is used for future device expansion.

Related Documentation From Texas Instruments

The following documents describe the C6000™ devices and related support tools. Copies of these documents are available on the Internet at www.ti.com. *Tip:* Enter the literature number in the search box provided at www.ti.com.

TMS320C672x DSP Peripherals Overview Reference Guide (literature number SPRU723) describes peripherals available on the TMS320C672x™ DSPs.

TMS320C6000 Technical Brief (literature number SPRU197) gives an introduction to the TMS320C62x™ and TMS320C67x™ DSPs, development tools, and third-party support.

TMS320c672x DSP CPU and Instruction Set Reference Guide (literature number SPRU733) describes the TMS320C672x™ CPU architecture, instruction set, pipeline, and interrupts for these digital signal processors.

TMS320C6000 Code Composer Studio Tutorial (literature number SPRU301) introduces the Code Composer Studio™ integrated development environment and software tools.

TMS320C6000 Programmer's Guide (literature number SPRU198) describes ways to optimize C and assembly code for the TMS320C6000 DSPs and includes application program examples.

Code Composer Studio Application Programming Interface Reference Guide (literature number SPRU321) describes the Code Composer Studio™ application programming interface (API), which allows you to program custom plug-ins for Code Composer.

Trademarks

C6000, TMS320C672x, TMS320C62x, TMS320C67x, Code Composer Studio are trademarks of Texas Instruments.

Reference Guide
SPRU719A–December 2005–Revised March 2009

TMS320C672x DSP Universal Host Port Interface (UHPI)

1 C672x UHPI Overview

1.1 Introduction

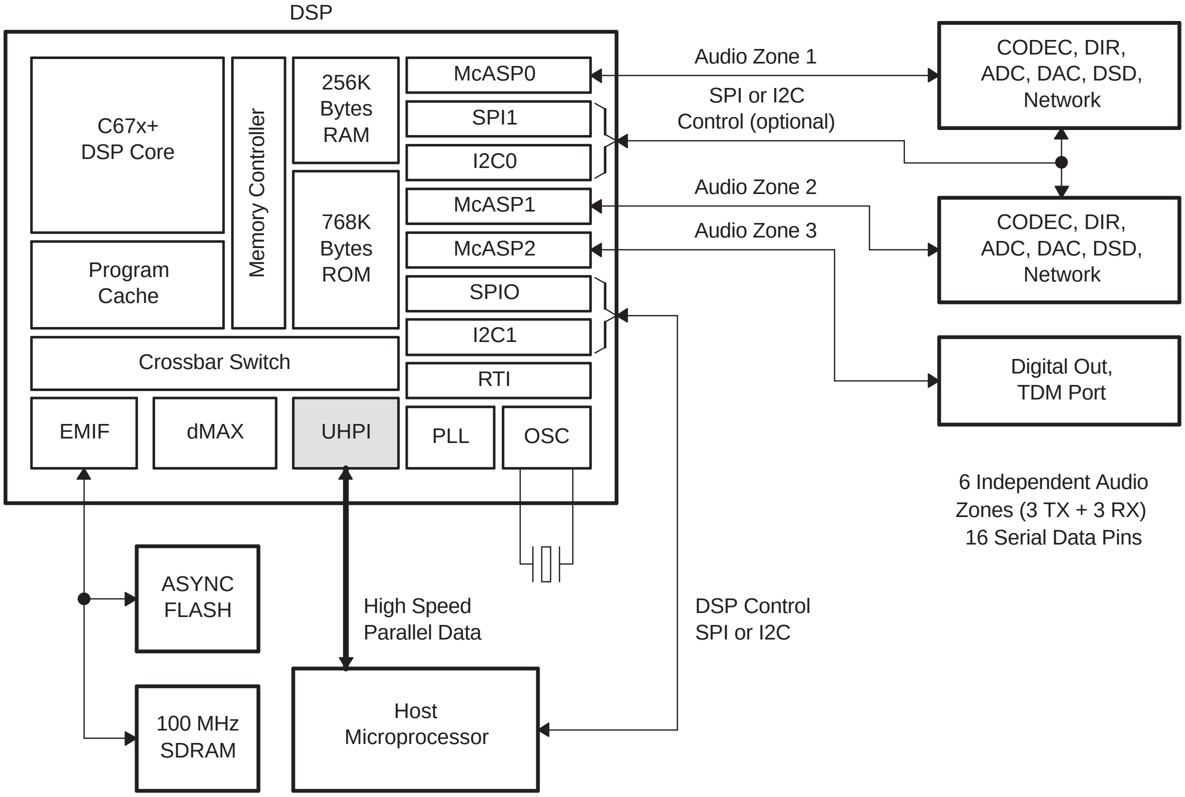
The UHPI is a high-speed parallel interface which is used to allow an external host microprocessor to communicate with the TMS320C672x DSP device. An example of a typical C672x system appears in Figure 1.

In this example, the host microprocessor connects to the UHPI using an asynchronous memory interface. Several UHPI modes are available to accommodate hosts with different types of busses.

The features that the UHPI provides are:

- The host may access the DSP memories (RAM, ROM, and external devices on the EMIF)
- The DSP may restrict the host to a single 64K byte page of memory
- Or, the DSP may disable host accesses completely
- The host may bootload the DSP through the UHPI
- The DSP and host may signal each other through UHPI interrupts

Figure 1. UHPI in a Typical C6727 System

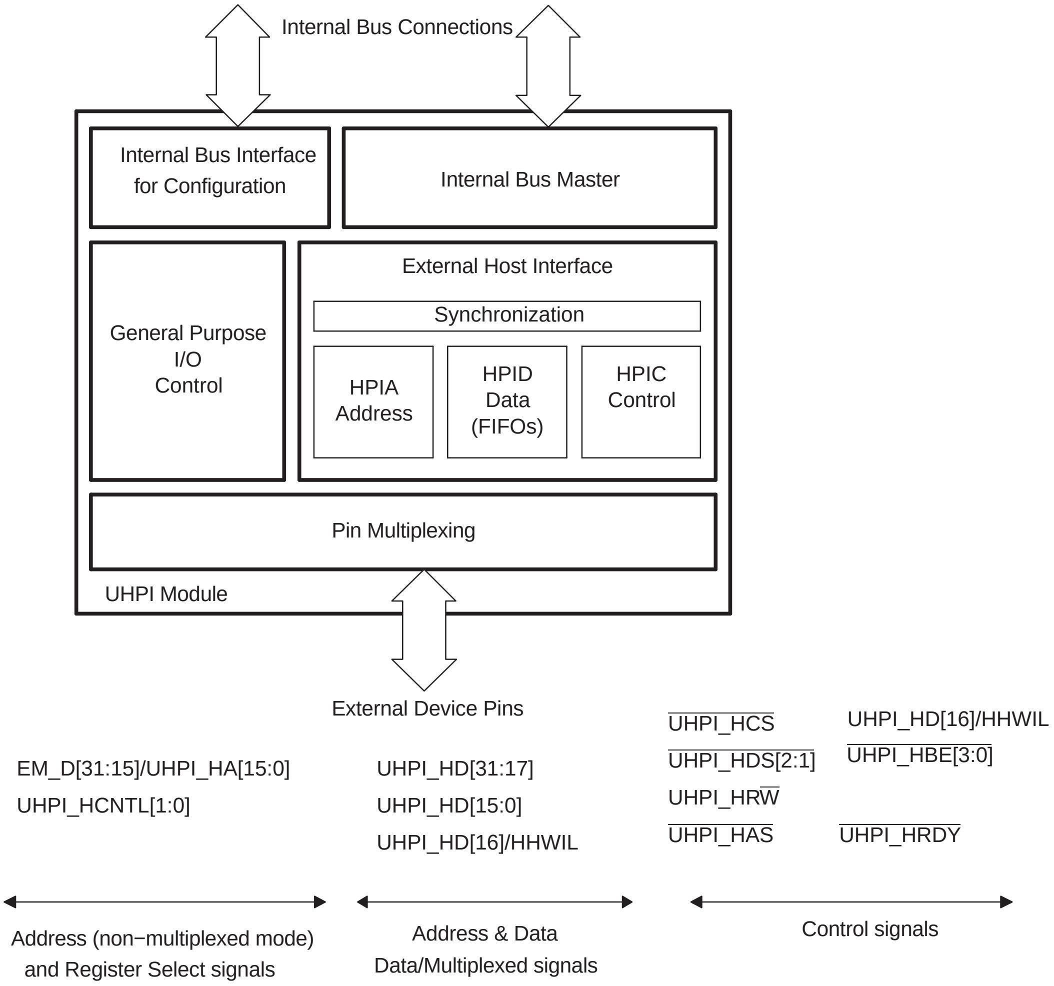


1.2 C672x UHPI Block Diagram

Figure 2 contains a block diagram of the C672x UHPI peripheral and includes the following major functional blocks:

- Pin multiplexing block: Multiplexes UHPI functions with general-purpose I/O functions.
- External host interface: Allows the external host to issue requests to the C672x internal bus master. Includes FIFOs.
- Internal bus master: Takes commands from the external host interface, and creates accesses to the C672x internal bus structure to execute the commands.
- Internal bus interface for configuration: Allows the DSP CPU to initialize the UHPI module, as well as access UHPI registers shared with the external host.

Figure 2. UHPI Block Diagram



1.3 UHPI Pins

An overview of the UHPI pins is given in Table 1

Table 1. UHPI Pins

Signal Name	Description
UHPI_HD[15:0]	Lower 16 bits of UHPI data bus. In dual halfword mode, these are the only data pins required.
UHPI_HD[16]/HHWIL	In dual halfword mode, this pin is the HHWIL input which is used to distinguish between the first halfword (HHWIL = 0) and the second halfword (HHWIL = 1) always in this order. In fullword mode, this pin functions as data bit 16.
UHPI_HD[31:17]	Not used in dual halfword mode (can be GPIO). In fullword mode these pins function as the upper portion of the UHPI data bus.
$\overline{\text{UHPI\_HBE[3:0]}}$	UHPI Byte Enables (Optional). In dual halfword mode, only $\overline{\text{UHPI\_BE[1:0]}}$ are used. Byte enables are applicable to single HPID accesses. All byte enables must be active during HPID with post increment (burst) UHPI accesses.
UHPI_HCNTL[1:0]	UHPI Control Inputs selecting between HPIA, HPIC, HPID, and HPID with post increment accesses.
$\overline{\text{UHPI\_HAS}}$	Optional host address strobe input. Can be used to latch UHPI_HCNTL[1:0], UHPI_HR$\overline{\text{W}}$, and HHWIL in multiplexed modes for hosts that have multiplexed address and data busses. Not used in non-multiplexed host address/data mode.
UHPI_HR/$\overline{\text{W}}$	Indicates whether the HPI access is a read or a write
$\overline{\text{UHPI\_HCS}}$, $\overline{\text{UHPI\_HDS[2:1]}}$	These three signals combine to generate the internal $\overline{\text{HSTROBE}}$ signal, which is the fundamental strobe signal from the host to the UHPI.
$\overline{\text{UHPI\_HRDY}}$	UHPI Ready output. Must be used to extended host wait states while this signal is deasserted.
EM_D[31:16]/UHPI_HA[15:0]	These pins are shared between the EMIF upper 16 data bits and the UHPI host address inputs. The UHPI host address inputs are used in non-multiplexed host address/data mode to provide a simpler asynchronous SRAM type interface to an external host. The UHPI host address inputs are not used in multiplexed modes of the UHPI.

1.4 Comparison of UHPI Major Operating Modes

A more detailed comparison between the modes is given in Table 2

Table 2. Comparison of UHPI Major Operating Modes

Feature	Multiplexed Host Address/Data Fullword	Multiplexed Host Address/ Data Dual Halfword	Non-Multiplexed Host Address/ Data Fullword
Host Data Bus	x32	x16	x32
Host Address Bus	Multiplexed with Host Data	Multiplexed with Host Data	16 Separate Address Lines
Restricted Addressing through **CFGHPIAMSB**/**CFGHPIAUMB**	Optional	Optional	Required
HPIA Register Access	Yes	Yes	No
HPID Register Access	Yes	Yes	Yes
HPID Register Access + Post Increment of **HPIA**	Yes	Yes	No
HPIC Register Access	Yes	Yes	Yes
HPID FIFO Depth	8 Words	8 Words	1 Word
Handshake signal UHPI_HRDY required	Yes	Yes	Yes
DSP and Host Interrupts Available	Yes	Yes	Yes
Required UHPI Signals	UHPI_HD[31:0], UHPI_HCS, UHPI_HDS[2:1], UHPI_HRDY	UHPI_HD[15:0], UHPI_HD[16]/HHWIL (as HHWIL), UHPI_HCS, UHPI_HDS[2:1], UHPI_HRDY	EM_D[31:16]/UHPI_HA[15:0] (as UHPI_HA[15:0]), UHPI_HD[31:0], UHPI_HCS, UHPI_HDS[2:1], UHPI_HRDY
Optional UHPI Signals[1]	UHPI_HBE[3:0], UHPI_HAS, UHPI_HINT,	UHPI_HBE[1:0], UHPI_HAS, UHPI_HINT	UHPI_HBE[3:0], UHPI_HINT
Unused UHPI Signals[1]		UHPI_HD[31:17], UHPI_HBE[3:2]	UHPI_HAS
Effect on C672x EMIF	None	None	EMIF is limited to 16-bit memories since EM_D[31:16]/UHPI_HA[15:0] pins function as host address inputs.

[1] In general, unused and optional UHPI signals may be used as general purpose I/O. However, the EM_D[31:6]/UHPI_HA[15:0] pins are not available for use as general purpose I/O even in multiplexed UHPI modes.

1.5 Multiplexed Host Address/Data versus Non-Multiplexed Host Address/Data Modes

The external host interface of the UHPI supports three major operating modes:

- Multiplexed host address/data fullword mode
- Multiplexed host address/data dual halfword mode
- Non-multiplexed host address/data fullword mode

1.5.1 Multiplexed Host Address/Data Mode Advantages

In the two multiplexed modes, the UHPI interfaces to the external host through data and control signals only. The host writes addresses into the HPIA register before reading or writing the HPID register to trigger an access on the C672x device. Advantages of operating the UHPI in multiplexed host address and data modes are:

- Requires fewer pins than non-multiplexed host address/data mode
- Does not affect 672x EMIF width
- Internal 8 Deep FIFOs improve performance for applications with large numbers of sequential accesses

1.5.2 Non-Multiplexed Host Address/Data Mode Advantages

In the non-multiplexed mode, the external host accesses the UHPI in much the same way as it would an asynchronous SRAM, except that the UHPI must be able to assert wait states through UHPI_HRDY. Advantages of operating the UHPI in non-multiplexed host address/data mode are:

- Simple interface - like an asynchronous SRAM
- Separate address and data pins may improve performance for applications with large amounts of random access through the UHPI

1.6 Dual HPIA and Single HPIA Options

The dual and single HPIA options are the two major options that affect UHPI operation for the multiplexed host address/data modes.

1.6.1 Dual HPIA Option and Advantages

With the dual HPIA option, there are separate copies of the current read and write addresses. In addition, the FIFO acts as two separate FIFOs, each eight words deep. One FIFO is dedicated to reads while the other is dedicated to writes. The advantage of choosing this mode is that the UHPI operation can be more efficient. For example, while the read FIFO is occupied with prefetching data from internal memory, the host can be writing data into the write FIFO. However, this mode allows coherency problems in the sense that it is possible to load old data from a memory location which is about to be replaced by data already written into the write FIFO but not yet copied to memory.

1.6.2 Single HPIA Option and Advantages

With the single HPIA option, the two HPIA registers and two FIFOs behave as if there is only a single address register and single FIFO used for both reads and writes. The advantage of choosing this mode is that any data in the write FIFO which has not yet been written to memory will be flushed (written to memory) before a read occurs. This ensures that the external host never reads data that is 'stale' due to its own write operations not having completed.

1.7 Fullword and Dual Halfword Operation Options

The UHPI supports both fullword and dual halfword options for multiplexed host/address and data modes. The control bit responsible for this choice is CFGHPI.FULL. Each of these options is discussed in this section. For non-multiplexed address/data mode, only fullword operation is supported.

1.7.1 Fullword Operation

If the fullword option is chosen (CFGHPI.FULL=1), then each external host access is done in one step and all 32 host data pins are used (UHPI_HD[31:0]). The multiplexed pin UHPI_HD[16]/HHWIL is used as HD[16].

During fullword operation, the HPIC.HWOB bit has no effect and the HHWIL input is not required.

1.7.2 Dual Halfword Operation

If the dual halfword option is chosen (CFGHPI.FULL=0), then each external host access requires two steps. The two halfwords are demultiplexed internally and presented to the external host interface as a single 32-bit word.

In this case, only UHPI_HD[15:0] are used to carry data and UHPI_HD[16]/HHWIL is used as the HHWIL signal.

Dual halfword operation is not available when non-multiplexed address/data mode is enabled.

1.7.3 HHWIL Pin during Dual Halfword Operation

When using the UHPI with the dual halfword option, the UHPI_HD[16]/HHWIL pin functions as the HHWIL input.

In this mode, the HPI operation is broken into two steps, and a halfword is transferred during each step. The HHWIL pin must always be driven low during the first halfword transfer and high during the second halfword transfer, in this order.

Note: The HHWIL pin does not signify whether the halfword is most significant or least significant. This is controlled by HPIC.HWOB. HHWIL simply distinguishes between the first half and the second half of an operation.

1.7.4 HPIC.HWOB during Dual Halfword Operation

To support hosts with different endianness, the HPIC supports a programmable halfword order. The HPIC.HWOB bit controls this and must be initialized by the external host.

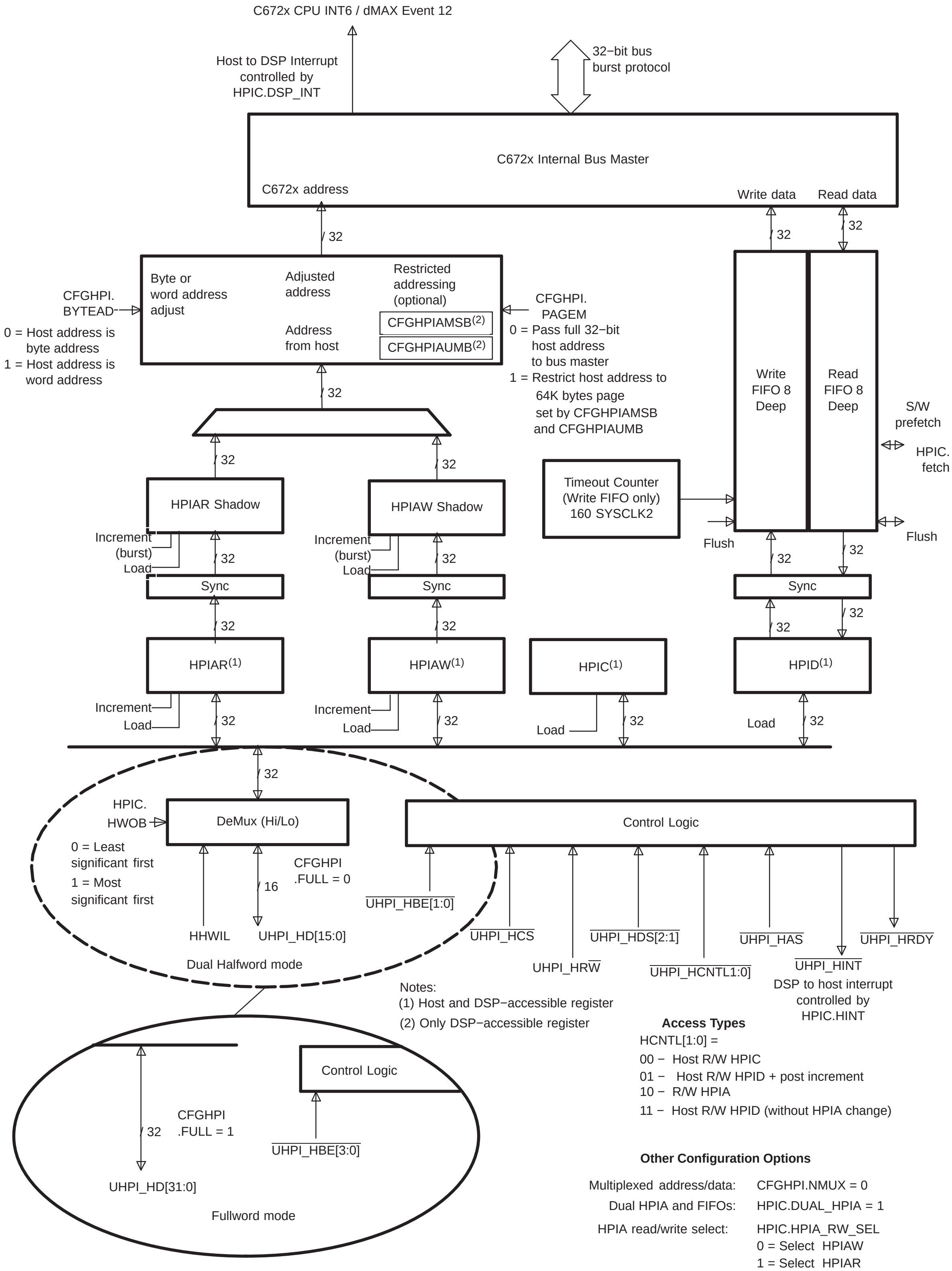
When **HPIC.HWOB**=0, the UHPI assembles 32-bit values by interpreting the first halfword transferred (HHWIL=0) as bits [15:0] and the second halfword transferred (HHWIL=1) as bits [31:16].

Likewise, when **HPIC.HWOB**=1, the UHPI assembles 32-bit values by interpreting the first halfword transferred (HHWIL=0) as bits [31:16] and the second halfword transferred (HHWIL=1) as bits [15:0].

1.8 Multiplexed Host Address/Data Mode with Dual HPIA Option

Figure 3 shows the UHPI in multiplexed modes with a dual **HPIA** view.

Figure 3. Multiplexed Host Address/Data with Dual HPIA Option



Notes:
(1) Host and DSP−accessible register
(2) Only DSP−accessible register

Access Types

HCNTL[1:0] =
00 − Host R/W HPIC
01 − Host R/W HPID + post increment
10 − R/W HPIA
11 − Host R/W HPID (without HPIA change)

Other Configuration Options

Multiplexed address/data:	CFGHPI.NMUX = 0
Dual HPIA and FIFOs:	HPIC.DUAL_HPIA = 1
HPIA read/write select:	HPIC.HPIA_RW_SEL 0 = Select HPIAW 1 = Select HPIAR

1.8.1 Supported Access Types

In multiplexed host address/data mode with dual **HPIA** registers, the UHPI supports the following access types, which are selected based upon the UHPI_HCNTL[1:0] and UHPI_HRW pins as well as the **HPIC.HPIA_RW_SEL** bit.

Table 3. UHPI Access Types for Multiplexed Host Address/Data with Dual HPIA

Access Type	**UHPI_HCNTL[1:0]**	**UHPI_HRW**	**HPIC.HPIA_RW_SEL bit**
Read **HPIC**	"00"	1	Don't Care
Write **HPIC**	"00"	0	Don't Care
Read **HPID** + **HPIAR** Post Increment	"01"	1	Don't Care
Write **HPID** + **HPIAW** Post Increment	"01"	0	Don't Care
Read **HPIAW**	"10"	1	0
Write **HPIAW**	"10"	0	0
Read **HPIAR**	"10"	1	1
Write **HPIAR**	"10"	0	1
Read **HPID** (no post increment)	"11"	1	Don't Care
Write **HPID** (no post increment)	"11"	0	Don't Care

1.8.2 Fullword and Dual Halfword Options Supported

In multiplexed host address/data mode, both fullword and dual halfword options are supported. See Section 1.7.1 for an explanation of the fullword option and Section 1.7.2 for an explanation of the dual halfword option.

1.8.3 HPIAR and HPIAW Register Operation

In the dual **HPIA** mode, separate registers exist for read accesses and write accesses.

The **HPIAR** register contains the address used for the next **HPID** read.

The **HPIAW** register contains the address used for the next **HPID** write.

When writing to these registers be sure to set **HPIC.HPIA_RW_SEL** for the desired register before performing the **HPIA** write operation.

1.8.3.1 Byte or Word Addressing Options

The UHPI supports external hosts which provide either a byte or a word aligned address in the **HPIAW** and **HPIAR** registers. The UHPI configuration bit **CFGHPI.BYTEAD** selects between the two options.

By selecting **CFGHPI.BYTEAD** = 0, the UHPI interprets the **HPIAW** and **HPIAR** registers as word aligned addresses. For example, if **HPIAW** were to contain the value 0x0000_0001 then this would be interpreted as byte address 0x0000_0004 in the DSP memory space.

By selecting **CFGHPI.BYTEAD** = 1, the UHPI interprets the **HPIAW** and **HPIAR** registers as byte aligned addresses and no shifting is required.

1.8.3.2 Restricted Addressing Option

The UHPI supports an option to override the upper 16 bits of the host address replacing these with the 16 bits stored in the **CFGHPIAMSB** and **CFGHPIAUMB** registers. This allows the DSP to restrict the external host to a 64K-Byte page in the DSP's memory space. The page can be changed, but only by the DSP.

An example of how this feature might be used is to allow the external host high-speed access to an on chip data RAM area, while preventing the external host from reading proprietary code stored in ROM.

To enable this feature, the upper 16 bits of the page address should be written to the **CFGHPIAMSB** (holds address [31:24]) and **CFGHPIAUMB** (holds address [23:16]). Then the control bit **CFGHPI.PAGEM** should be set to '1'.

Note that the address override occurs **after** the address shift is made if **CFGHPI.BYTEAD** = '0'. In other words, **CFGHPIAMSB** and **CFGHPIAUMB** always override the internal address range [31:16] regardless of whether the host is configured for byte addressing or word addressing.

1.8.3.3 Initializing HPIAR, HPIAW before accessing HPID

Before performing any **HPID** accesses, the correct **HPIA** register (**HPIAR** or **HPIAW**) must be initialized by performing a write to that register.

1.8.3.4 Shadowing of Address Registers

The **HPIAR** and **HPIAW** registers contain the next address from the perspective of the host; and do not account for burst effects. This set of registers is visible to the external host as well as to the DSP CPU.

A separate set of shadow address registers keeps track of the address needed for the next burst operation by the internal bus master. This set of registers is hidden from view and is not useful.

This means that when reading back **HPIAR** and **HPIAW** after performing **HPID** with postincrement transactions, the value returned will reflect the initial base address plus the number of post-increments actually performed by the host.

The effect of the FIFO is hidden from the host. For example, if two words have been written to the FIFO with **HPID** + postincrement accesses by the host, and then the host reads back the **HPIAW** register, it will see the original address has been incremented by 2 words. However, the data would still be in the FIFO, and the FIFO will pass the original address (stored in the shadow register) to the internal bus master as the base address for the burst access when it is actually time to issue the burst write to the memory system.

1.8.4 HPID Register Operation

When **HPIC.DUAL_HPIA**=1, the read FIFO operates independently from the write FIFO. In this mode, **HPID** reads operate on the read FIFO, and **HPID** writes operate on the write FIFO. This section describes the operation of the read and write FIFOs.

1.8.4.1 Read FIFO Operation

The read FIFO typically begins in an empty state (see Section 1.8.4.2).

There are two ways for the host to begin filling the read FIFO:

- Write to the **HPIC.FETCH** bit to initiate a prefetch
- Execute an **HPID** with post increment read access

If the read FIFO is empty, the UHPI will request a burst of eight words from the DSP internal memory system. The burst will begin at the address specified in the **HPIAR** register [after adjustments described in Section 1.8.3.1 and Section 1.8.3.2].

The host must wait until the FIFO is ready before completing the initial **HPID** read. The $\overline{\text{UHPI\_HRDY}}$ pin remains inactive (high) until the FIFO is ready. The host must extend its strobes to the UHPI until $\overline{\text{UHPI\_HRDY}}$ returns to an active (low) state.

Subsequent **HPID** with postincrement reads will continue to pull data from the read FIFO. When the read FIFO has space again for four more words, the UHPI internal bus interface generates a burst request for four more words in an attempt to keep the read FIFO from being emptied. The host can continue to read sequentially by issuing additional **HPID** with postincrement reads, until a read FIFO flush occurs.

1.8.4.2 Read FIFO Flush

When the read FIFO is flushed, any data currently in the FIFO is discarded and the FIFO becomes empty.

Actions which cause the read FIFO to flush are:

- Writing to the **HPIAR** register

- Reading from **HPID** without postincrement

1.8.4.3 *Write FIFO Operation*

The write FIFO typically begins in an empty state (after being flushed).

Data is written to the write FIFO using an **HPID** with postincrement write.

As soon as there are four words in the write FIFO, the UHPI issues a four word write burst to the DSP memory system in order to maintain the write FIFO near the empty level.

1.8.4.4 *Write FIFO Timeout*

The write FIFO normally waits until there are at least four words in the FIFO before it issues a burst write of four words to the DSP memory system.

However, the condition may arise where the FIFO contains less than four words. In this case, to prevent the data from sitting in the FIFO for an indefinite amount of time, the write FIFO includes a timeout counter. The timeout counter is reset every time new data is written to the FIFO, and expires after 160 SYSCLK2 cycles have passed.

When the timeout counter expires, if there are one, two, or three words still in the write FIFO then a burst write to the DSP memory system will be issued of the appropriate size, and the FIFO will be flushed. This mechanism ensures that write data reaches the DSP memory system in a timely manner.

1.8.4.5 *Write FIFO Flush*

When the write FIFO is flushed, any data currently in the FIFO is first ***written*** to memory. After all the data has been written to memory, the write FIFO becomes empty.

Actions which cause the write FIFO to flush are:

- Writing to the **HPIAW** register
- Writing to **HPID** without postincrement
- Expiration of the Write FIFO Timeout Counter

1.8.4.6 *FIFO Reset*

The read and write FIFOs are reset and all data in the FIFOs is discarded whenever:

- The DSP itself is reset.
- The DSP clears **CFGHPI.ENA**
- The DSP sets **HPIC.HPI_RST** = 1

When the HPI is reset, the UHPI_HRDY pin is asserted if there is a host access in progress. This allows any host accesses currently in progress to complete, although the data being transferred will be lost or invalid. On subsequent host accesses the UHPI_HRDY pin will be deasserted as long as the FIFOs remain in reset.

1.8.5 HPIC Register Operation

The **HPIC** register may be read and written at any time.

Accessing the **HPIC** register does not create wait states for the host (UHPI_HRDY will not be deasserted).

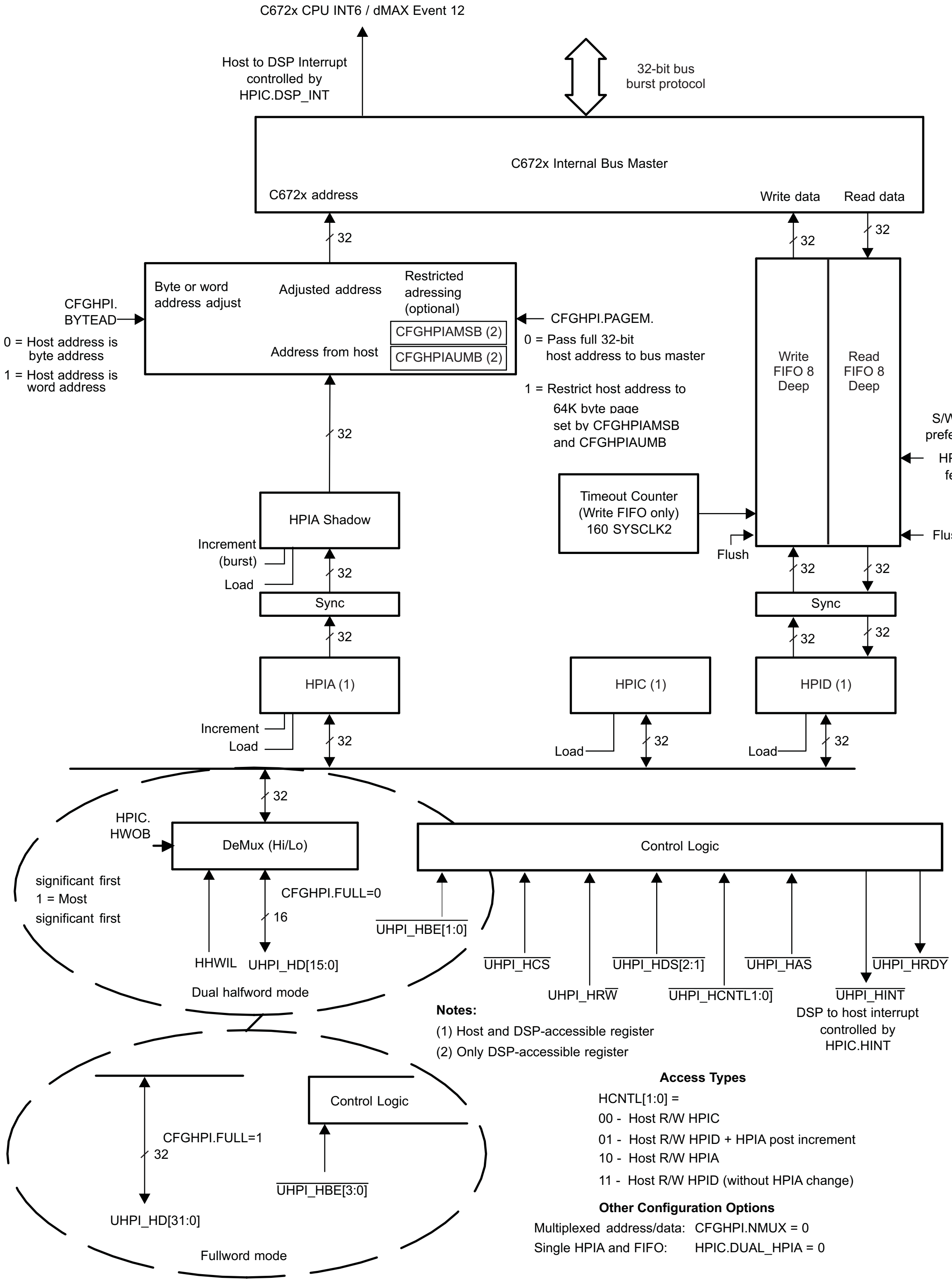
Simply accessing the **HPIC** register does not have any effect on the **HPIAR**/**HPIAW** registers or the read/write FIFOs. However, there are some bits in the **HPIC** register which may have a side effect on the **HPIAR**/**HPIAW** register and read/write FIFOs when set. The DSP writing to **HPIC.HPI_RST** will reset the FIFOs.

Also if the host modifies **HPIC.DUAL_HPIA** then it must explicitly flush the FIFO(s) before attempting an access.

1.9 Multiplexed Host Address/Data Mode with Single HPIA Option

The following diagram, Figure 4, shows the UHPI in multiplexed modes with a single **HPIA** view.

Figure 4. Multiplexed Host Address/Data with Single HPIA Option



Notes:

(1) Host and DSP-accessible register

(2) Only DSP-accessible register

Access Types

HCNTL[1:0] =

00 - Host R/W HPIC

01 - Host R/W HPID + HPIA post increment

10 - Host R/W HPIA

11 - Host R/W HPID (without HPIA change)

Other Configuration Options

Multiplexed address/data:	CFGHPI.NMUX = 0
Single HPIA and FIFO:	HPIC.DUAL_HPIA = 0

1.9.1 Supported Access Types

The multiplexed host address/data mode with single **HPIA** register option supports the following access types, which are selected based upon the UHPI_HCNTL[1:0] and UHPI_HR$\overline{W}$ pins. The **HPIC.HPIA_RW_SEL** bit has no effect when the single **HPIA** option is selected.

Table 4. UHPI Access Types for Multiplexed Host Address/Data with Dual HPIA

Access Type	**UHPI_HCNTL[1:0]**	**UHPI_HR$\overline{W}$**	**HPIC.HPIA_RW_SEL bit**
Read **HPIC**	"00"	1	Don't Care
Write **HPIC**	"00"	0	Don't Care
Read **HPID** + **HPIA** Post Increment	"01"	1	Don't Care
Write **HPID** + **HPIA** Post Increment	"01"	0	Don't Care
Read **HPIA**	"10"	1	Don't Care
Write **HPIA**	"10"	0	Don't Care
Read **HPID** (no post increment)	"11"	1	Don't Care
Write **HPID** (no post increment)	"11"	0	Don't Care

1.9.2 Fullword and Dual Halfword Options Supported

In multiplexed host address/data mode, both fullword and dual halfword options are supported. See Section 1.7.1 for an explanation of the fullword option and Section 1.7.2 for an explanation of the dual halfword option.

1.9.3 HPIA Register Operation

In the single **HPIA** mode, the **HPIAR** and **HPIAW** registers are combined. Also, the read and write FIFOs act as a single combined eight-word FIFO.

1.9.3.1 Byte or Word Addressing Options

The same byte and word addressing options are available as in dual **HPIA** mode. See Section 1.8.3.1

1.9.3.2 Restricted Addressing Option

The same restricted addressing options are available as in dual **HPIA** mode. See Section 1.8.3.2

1.9.3.3 Initializing HPIAR, HPIAW before accessing HPID

Before performing any **HPID** accesses, the correct **HPIA** (**HPIAR** or **HPIAW**) must be initialized by performing a write to that register.

1.9.3.4 Shadowing of Address Registers

In the single **HPIA** mode, the **HPIA** register is shadowed in much the same way that the **HPIAR** and **HPIAW** registers are shadowed in the dual **HPIA** mode. The shadow **HPIA** register is initialized to the same value as the **HPIA** register whenever a write to the **HPIA** register occurs. However, the **HPIA** register updates as the host reads or writes each word to the UHPI FIFOs. The shadow address only updates whenever a burst to the DSP memory system occurs; and in this case it updates by the correct amount as dictated by the burst size. The shadow address register is not accessible directly by either the host or the DSP.

1.9.4 HPID Register Operation

When **HPIC.DUAL_HPIA**=0, the read FIFO and write FIFO operate together as a single bidirectional FIFO. In this mode, **HPID** reads and **HPID** writes operate on this single bidirectional FIFO. This section describes the FIFO operation.

1.9.4.1 FIFO Operation

When **HPIC.DUAL_HPIA**=0, the read and write FIFOs operate as if they are a single bidirectional FIFO.

The behavior of the combined FIFO is the same as described in Section 1.8.4.1 (for reads) and Section 1.8.4.3 (for writes). However, because the read and write FIFOs are combined, the FIFO flush conditions are also combined.

1.9.4.2 FIFO Flush

When the FIFO is flushed, any read data remaining in the FIFO is discarded. Any write data remaining in the FIFO will be written to the DSP memory before the UHPI will become ready.

Actions which cause the FIFO to flush are:

- Writing to the **HPIA** register
- Reading from **HPID** without postincrement
- Writing to **HPID** without postincrement
- Expiration of the Write FIFO Timeout Counter
- Writing to **HPID** with postincrement while read data remains in the FIFO.
- Reading **HPID** with postincrement while write data remains in the FIFO.

1.9.4.3 Write FIFO Timeout

In single **HPIA** mode, the write FIFO timeout function behaves the same as it does in dual **HPIA** mode. See Section 1.8.4.4.

1.9.4.4 FIFO Reset

The same reset conditions affect the FIFO whether in single or dual HPIA mode. See Section 1.8.4.6 for a description of these conditions.

1.9.5 HPIC Register Operation

The **HPIC** register may be read and written at any time.

Accessing the **HPIC** register does not create wait states for the host ($\overline{\text{UHPI\_HRDY}}$ will not be deasserted).

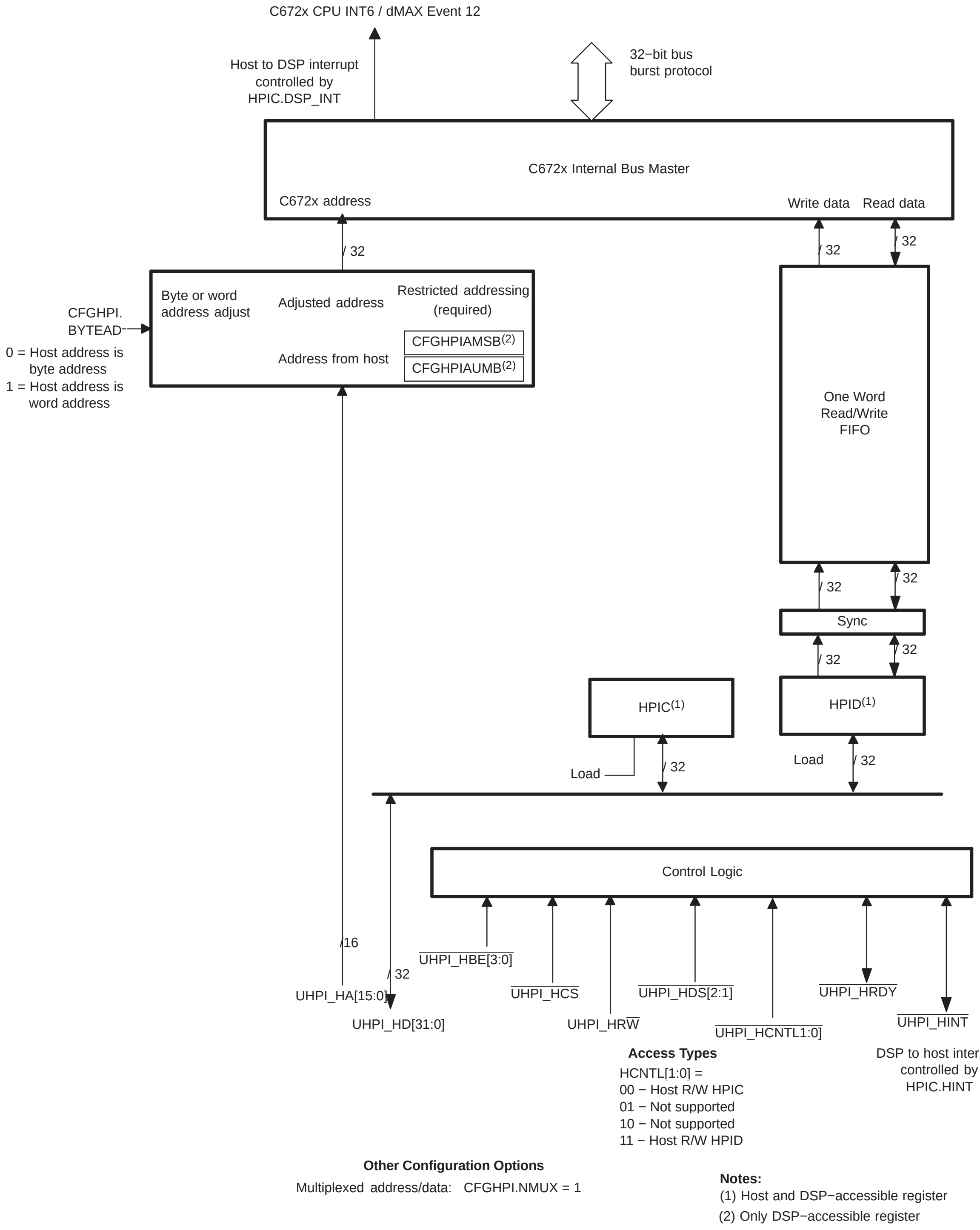
Simply accessing the **HPIC** register does not have any effect on the **HPIA** register or the FIFO. However, there are some bits in the **HPIC** register which may have a side effect on the **HPIA** register and FIFO when written. The DSP writing to **HPIC.HPI_RST** will reset the FIFO.

Also if the host modifies **HPIC.DUAL_HPIA**, then it should also ensure that the **HPIA** registers (**HPIAR** and **HPIAW**) are also initialized with the desired address values prior to attempting an access. A write to **HPIA** register (**HPIAR** or **HPIAW**) will result in flushing the corresponding FIFO, which is transparent to the host and user.

1.10 Non-Multiplexed Host Address/Data Mode

Figure 5 illustrates the functionality of the UHPI in the non-multiplexed host address/data host address/data mode.

Figure 5. Non-Multiplexed Host Address/Data Fullword Mode


C672x CPU INT6 / dMAX Event 12
Host to DSP interrupt controlled by HPIC.DSP_INT
32−bit bus burst protocol
C672x Internal Bus Master
C672x address
Write data
Read data
32
32
32
Byte or word address adjust
Adjusted address
Restricted addressing (required)
CFGHPI. BYTEAD
0 = Host address is byte address
1 = Host address is word address
Address from host
CFGHPIAMSB(2)
CFGHPIAUMB(2)
One Word Read/Write FIFO
32
32
Sync
32
32
HPIC(1)
HPID(1)
Load
32
Load
32
Control Logic
16
32
UHPI_HA[15:0]
UHPI_HD[31:0]
UHPI_HBE[3:0]
UHPI_HCS
UHPI_HRW
UHPI_HDS[2:1]
UHPI_HCNTL1:0]
UHPI_HRDY
UHPI_HINT
DSP to host interrupt controlled by HPIC.HINT

Access Types
HCNTL[1:0] =
00 − Host R/W HPIC
01 − Not supported
10 − Not supported
11 − Host R/W HPID

Other Configuration Options
Multiplexed address/data: CFGHPI.NMUX = 1

Notes:
(1) Host and DSP−accessible register
(2) Only DSP−accessible register

1.10.1 Supported Access Types

The non-multiplexed host address/data mode supports the following access types, which are selected based upon the UHPI_HCNTL[1:0] and UHPI_HRW pins. The **HPIC.HPIA_RW_SEL** bit has no effect.

Table 5. UHPI Access Types for Multiplexed Host Address/Data with Dual HPIA

Access Type	**UHPI_HCNTL[1:0]**	**UHPI_HRW**	**HPIC.HPIA_RW_SEL bit**
Read **HPIC**	"00"	1	Don't Care
Write **HPIC**	"00"	0	Don't Care
Not Supported	"01"	1	Don't Care
Not Supported	"01"	0	Don't Care
Not Supported	"10"	1	Don't Care
Not Supported	"10"	0	Don't Care
Read **HPID** (no post increment)	"11"	1	Don't Care
Write **HPID** (no post increment)	"11"	0	Don't Care

1.10.1.1 Only Fullword Supported

In non-multiplexed host address/data mode, only the fullword accesses are supported.

1.10.2 Addressing (UHPI_HA[15:0] used instead of HPIA Register)

In non-multiplexed address/data mode, the **HPIA** register is not used.

Instead, the UHPI_HA[15:0] inputs select the least significant portion of the address. **The host must provide an address along with data on every access**.

Byte and word addressing options are supported, but only apply to UHPI_HA[15:0].

The upper 16 bits of the DSP address are always configured by the DSP through the **CFGHPIAMSB** and **CFGHPIAUMB** registers. This restriction is due to the limited number of external host address input pins.

1.10.3 HPID Register Operation

In the non-multiplexed host address/data mode, only single **HPID** accesses without postincrement are supported.

1.10.4 HPIC Register Operation

The **HPIC** register may be read and written at any time.

Accessing the **HPIC** register does not create wait states for the host (UHPI_HRDY will not be deasserted).

1.11 Pin Multiplexing and General Purpose I/O Control Blocks

The UHPI supports general purpose I/O capability on all pins. All UHPI pins may be enabled for GPIO mode when the UHPI is disabled (CFGHPI.HPIENA=0).

When the UHPI is enabled, the pins not being used for host accesses may be configured as general purpose I/O. These pins are listed by mode in Table 2 (in the rows listing optional and unused pins).

1.11.1 Treatment of Optional Pins when Configured as General Purpose I/O

Certain pins are optional but if used to interface to the external host are inputs to the UHPI. For example the UHPI_HBE[3:0] pins fall into this category. In this case, for the purpose of host accesses, the UHPI treats these pins as if they were driven to the values listed in Table 6

Table 6. Treatment of Optional Pins when Configured as General Purpose I/O

Pin	GPIO Enable Bit(s)	When Enabled as GPIO, Treated as Driven:
UHPI_HCNTL[1]	GPIO_EN1	1
UHPI_HCNTL[0]	GPIO_EN1	1
UHPI_HAS	GPIO_EN2	1
UHPI_HBE[3]	GPIO_EN3	0
UHPI_HBE[2]	GPIO_EN3	0
UHPI_HBE[1]	GPIO_EN3	0
UHPI_HBE[0]	GPIO_EN3	0
HD[31:0]	GPIO_EN10 - GPIO_EN7	0
HA[15:0]	GPIO_EN12 - GPIO_EN11	0

1.11.2 General Purpose I/O Programmer's Model

For each UHPI pin, there are three bits that control this pin as general purpose I/O:

- Enable: GPIO_EN.GPIO_EN[xx]
- Direction: GPIO_DIRn.DIR[yy]
- Data: GPIO_DAT[n].DIR[yy]

For example, the UHPI_HAS pin is enabled with bit GPIO_EN.GPIO_EN2. In the default setting, this enable bit is '0' and therefore the UHPI_HAS pin functions as the host address strobe.

Enabling this pin for GPIO GPIO_EN.GPIO_EN2 to '1' does two things:

- Transfers control of the UHPI_HAS pin to GPIO direction and data bits
- Drives a '1' into the UHPI_HAS input of the external host interface block (regardless of the actual pin value).

Once enabled as general purpose I/O, the direction of the UHPI_HAS is controlled by the bit GPIO_DIR2.DIR0. If this bit is set to '1', the pin will be driven as an output. If this bit is set to '0', the pin will be an input.

Once the direction of the UHPI_HAS pin is set, the data value is either written to or read from the GPIO_DAT2.DAT0 bit. If the pin was configured as an output, then writing to this bit determines the value to drive out the pin. Reading from this bit will return the value written.

On the other hand, when GPIO_DIR2.DIR0 is set to '0' configuring UHPI_HAS as an input, writing to GPIO_DAT2.DAT0 has no effect. Reading from this bit will return the value on the UHPI_HAS.

Note: Note that you cannot preload a value into the 'DAT' field before configuring the pin as an output. This means that when switching the pin from input to output, the pin will initially drive the last value input back out the pin. Then the DAT bit can be written to change the value on the pin. If the intermediate value between writing to DIR and writing to DAT will cause a problem at the system level, it is suggested to use another general purpose I/O pin on the C672x device (for example from one of the McASP peripherals).

1.11.3 GPIO Interrupt Capability ($\overline{\text{UHPI\_HAS}}$ pin only)

The $\overline{\text{UHPI\_HAS}}$ pin has interrupt capability when used as a general purpose input.

It can generate events on:

- DSP CPU Interrupt INT6
- dMAX Event 12

This capability is controlled by the GPINT_CTRL register.

The GPINT_CTRL.GPINT_EN2 bit controls whether the value at the $\overline{\text{UHPI\_HAS}}$ pin is routed to the dMAX and CPU.

The signal may be inverted before being sent to the dMAX and CPU by setting the GPINT_CTRL.GPINT_INV2 bit.

The DSP CPU requires a rising edge on INT6 to generate an event.

The dMAX can accept either edge on Event 12 (programmable). Therefore it would be possible to trigger a dMAX event and CPU interrupt on opposite edges of the $\overline{\text{UHPI\_HAS}}$ pin.

2 C672x UHPI Operation

2.1 Initialization

Before using the UHPI on C672x devices, the peripheral must be enabled and configured by the DSP. Additionally, some configuration must be done by the external host. This section describes the initialization steps.

2.1.1 Enabling and Configuring the UHPI (DSP)

Initially after a reset, the C672x UHPI is held in a disabled state until configured and then enabled by the DSP.

The steps which the DSP must follow to enable and configure the UHPI are:

1. The **CFGHPI** register should be written in order to configure the major operating modes of the DSP. First write to only the **CFGHPI.BYTEAD**, **CFGHPI.FULL**, **CFGHPI.NMUX**, and **CFGHPI.PAGEM** bits leaving the **CFGHPI.ENA** bit as '0'. For a description of these control bits see Section 3.2.
2. If either **CFGHPI.PAGEM** or **CFGHPI.NMUX** are set in step 1, then the **CFGHPIAMSB** and **CFGHPIAUMB** registers must be initialized with 64K Byte page (upper 16 address bits) to which host accesses will be restricted.
3. Next, the HPI should be enabled by setting **CFGHPI.ENA**='1' (without changing the values of the other bits which were set in step 1).
4. The DSP should then write to the **HPIC.HPI_RST** bit setting this bit to '0'. At this point, the UHPI will respond to accesses from the external host, however the UHPI still requires further initialization before the host should attempt to access it.
5. If certain pins (for example, the $\overline{\text{UHPI\_HBE[3:0]}}$ or $\overline{\text{UHPI\_HAS}}$ pins) are not going to be used by the host, these pins should be configured as general purpose I/O so the host interface will use the default value and not the actual pin value for these signals.

After completing the steps above, the UHPI is now ready for configuration by the external host.

2.1.2 Configuring the UHPI (External Host)

The external host should write to the **HPIC** register to configure the UHPI before beginning any other accesses.

The host should correctly configure the **HPIC.HPIA_RW_SEL**, **CFGHPI.DUAL_HPIA**, and **HPIC.HWOB** bits. See Section 3.15 for a description of these bits.

After configuring the **HPIC** register, the host must initialize the **HPIA** register(s) to a valid address before attempting any **HPID** accesses.

2.2 Host Access Types

This section includes diagrams of the different types of UHPI accesses available to an external host.

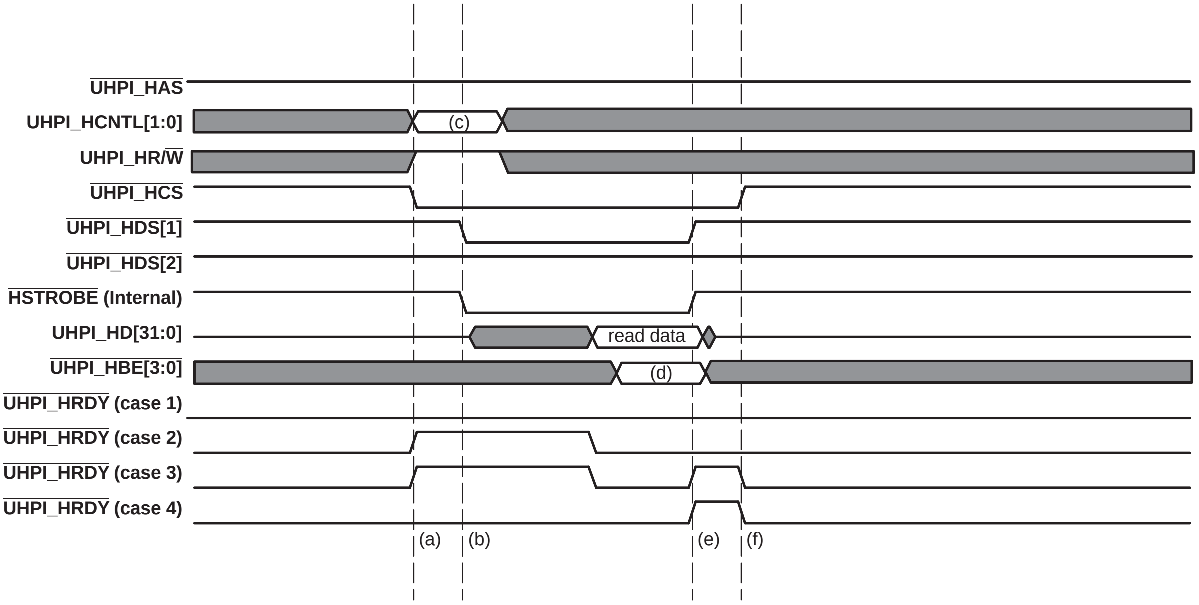
2.2.1 Multiplexed Host Address/Data Mode, Fullword Option

This section describes host accesses in the multiplexed host address/data mode with fullword option.

2.2.1.1 Read Accesses

Figure 6 illustrates a host read from the UHPI in multiplexed host address/data, fullword mode using the falling edge of the internal HSTROBE signal to latch in the UHPI register address. Figure 7 illustrates the same access using the optional UHPI_HAS signal to latch in the register address.

Figure 6. Multiplexed Host Address/Data Fullword Read (without UHPI_HAS)



a In cases where the UHPI is initially not ready,UHPI_HRDY will be deasserted only while UHPI_HCS is asserted.

b UHPI latches UHPI_HCNTL[1:0] and UHPI_HRW on the falling edge of HSTROBE.

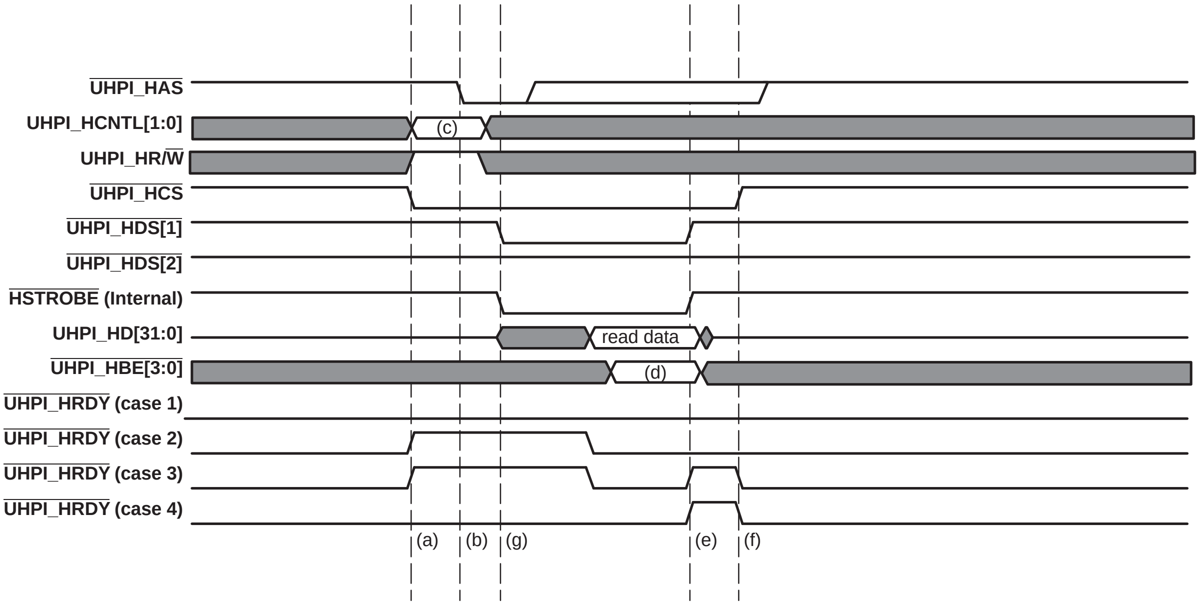
c UHPI_HCNTL[1:0] selects register address: 00=**HPIC**, 01=**HPID** with post increment, 10=**HPIA**, 11=**HPID** without postincrement

d UHPI_HBE[3:0] are ignored by the UHPI during reads

e Host latches UHPI read data on rising edge of HSTROBE

f In cases where the UHPI is not ready after the access, UHPI_HRDY remains deasserted until UHPI_HCS is deasserted.

Figure 7. Multiplexed Host Address/Data Fullword Read (using UHPI_HAS)



a In cases where the UHPI is initially not ready,UHPI_HRDY will be deasserted only while UHPI_HCS is asserted.

b UHPI latches UHPI_HCNTL[1:0] and UHPI_HRW on the falling edge of UHPI_HAS.

c UHPI_HCNTL[1:0] selects register address: 00=**HPIC**, 01=**HPID** with postincrement, 10=**HPIA**, 11=**HPID** without postincrement

d UHPI_HBE[3:0] are ignored by the UHPI during reads

e Host latches UHPI read data on rising edge of HSTROBE

f In cases where the UHPI is not ready after the access, UHPI_HRDY remains deasserted until UHPI_HCS is deasserted.

g UHPI_HAS can be released any after the falling edge of HSTROBEis captured by the UHPI.

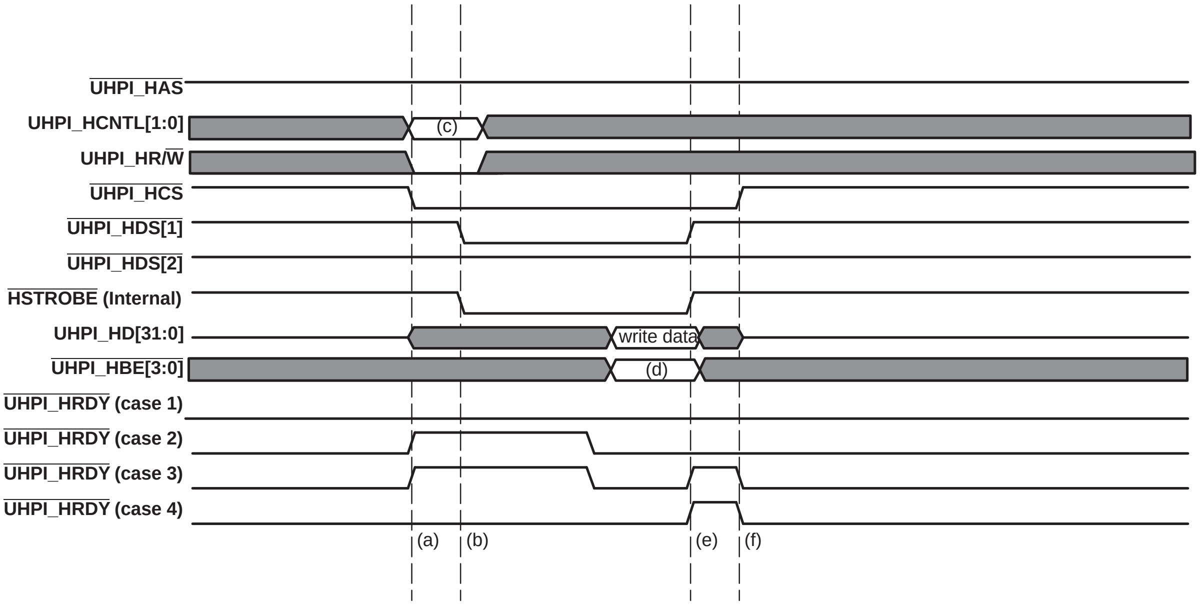
In Figure 6 and Figure 7 there are four different cases for the UHPI_HRDY signal, depending upon the UHPI state when the read is issued. The cases are:

- Case 1: UHPI_HRDY is never deasserted. This case occurs if either **HPIA** or **HPIC** is read. This case also occurs if an **HPID** with post increment read is issued, and the read FIFO is not empty.
- Case 2: UHPI_HRDY is initially deasserted. This case occurs if an **HPID** read without post increment is issued. It also occurs if an **HPID** read with post increment is issued but the read FIFO is empty or being flushed.
- Case 3: UHPI_HRDY is initially deasserted, and then is deasserted again after the read completes. This case occurs if the FIFO is empty or being flushed when an **HPID** read with postincrement is issued, and is empty again after the read completes.
- Case 4: UHPI_HRDY is initially asserted, but is deasserted after the read completes. This occurs when the read access empties the FIFO.

2.2.1.2 Write Accesses

Figure 8 illustrates a host write to the UHPI in multiplexed host address/data, fullword mode using the falling edge of the internal HSTROBE signal to latch in the UHPI register address. Figure 9 illustrates the same access using the optional UHPI_HAS signal to latch in the register address.

Figure 8. Multiplexed Host Address/Data Fullword Write (without UHPI_HAS)



a In cases where the UHPI is initially not ready,UHPI_HRDY will be deasserted only while UHPI_HCS is asserted.

b UHPI latches UHPI_HCNTL[1:0] and UHPI_HRW on the falling edge of HSTROBE.

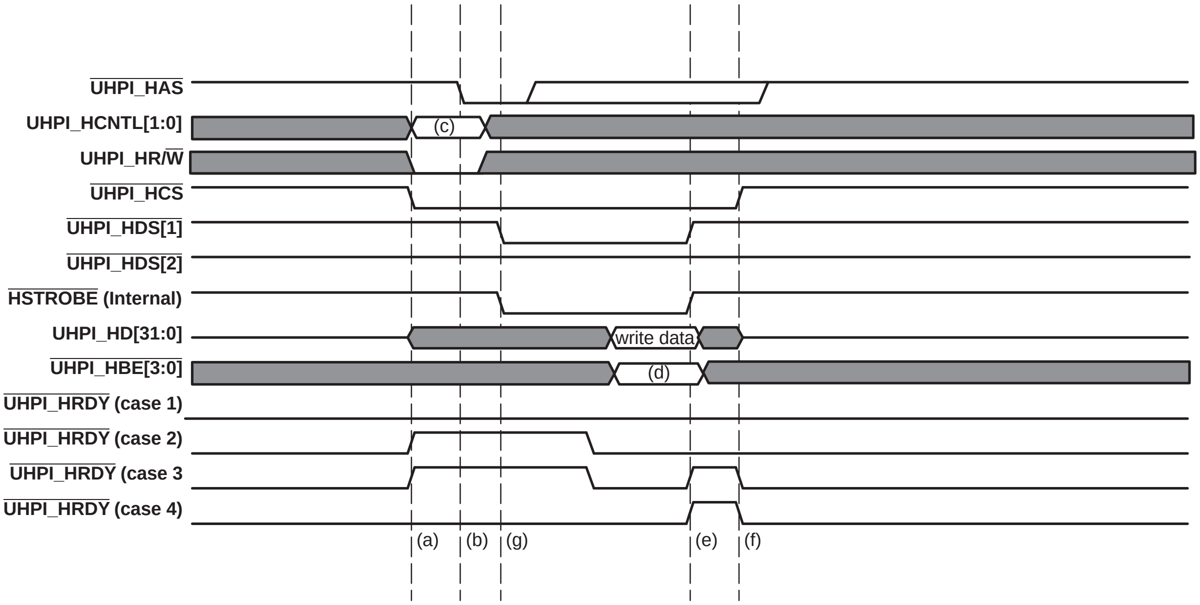
c UHPI_HCNTL[1:0] selects register address: 00=**HPIC**, 01=**HPID** with postincrement, 10=**HPIA**, 11=**HPID** without postincrement

d Byte writes are only supported for **HPID without** post increment accesses. For all other accesses, all four of the UHPI_HBE[3:0] must be asserted.

e UHPI latches in write data and byte enables on rising edge of HSTROBE

f In cases where the UHPI is not ready after the access, UHPI_HRDY remains deasserted until UHPI_HCS is deasserted.

Figure 9. Multiplexed Host Address/Data, Fullword Write (using UHPI_HAS)



a In cases where the UHPI is initially not ready,UHPI_HRDY will be deasserted only while UHPI_HCS is asserted.

b UHPI latches UHPI_HCNTL[1:0] and UHPI_HRW on the falling edge of UHPI_HAS.

c UHPI_HCNTL[1:0] selects register address: 00=**HPIC**, 01=**HPID** with postincrement, 10=**HPIA**, 11=**HPID** without postincrement

d Byte writes are only supported for **HPID without** post increment accesses. For all other accesses, all four of the UHPI_HBE[3:0] must be asserted.

e UHPI latches in write data and byte enables on rising edge of HSTROBE

f In cases where the UHPI is not ready after the access, UHPI_HRDY remains deasserted until UHPI_HCS is deasserted.

g UHPI_HAS can be released any after the falling edge of HSTROBEis captured by the UHPI.

In Figure 8 and Figure 9 there are four different cases for the UHPI_HRDY signal, depending upon the UHPI state when the write is issued. The cases are:

- Case 1: UHPI_HRDY is never deasserted. This case will occur if **HPIC** is written, and in most cases if **HPIA** is written. This case also occurs if the write is to **HPID** with post increment and the write FIFO is not full.
- Case 2: UHPI_HRDY is initially deasserted. This case occurs during an **HPIA** write if a previous **HPIA** write has not yet been synchronized into the DSP SYSCLK2 domain. This case also occurs if an **HPID** write with post increment is issued but the write FIFO is full or being flushed.
- Case 3: UHPI_HRDY is initially deasserted, and then deasserted again after the write completes. The case occurs if an **HPID** write without post increment is issued, and the write FIFO is not empty or being flushed. This case can also occur if an **HPID** write with post increment is issued, and the FIFO is full both before and after the write.
- Case 4: UHPI_HRDY is initially asserted, but becomes deasserted after the write completes. This case occurs when the write FIFO is initially empty or being flushed and there is an **HPID** write without post increment. This case also occurs when the write FIFO is not full initially, but becomes full as a result of the write.

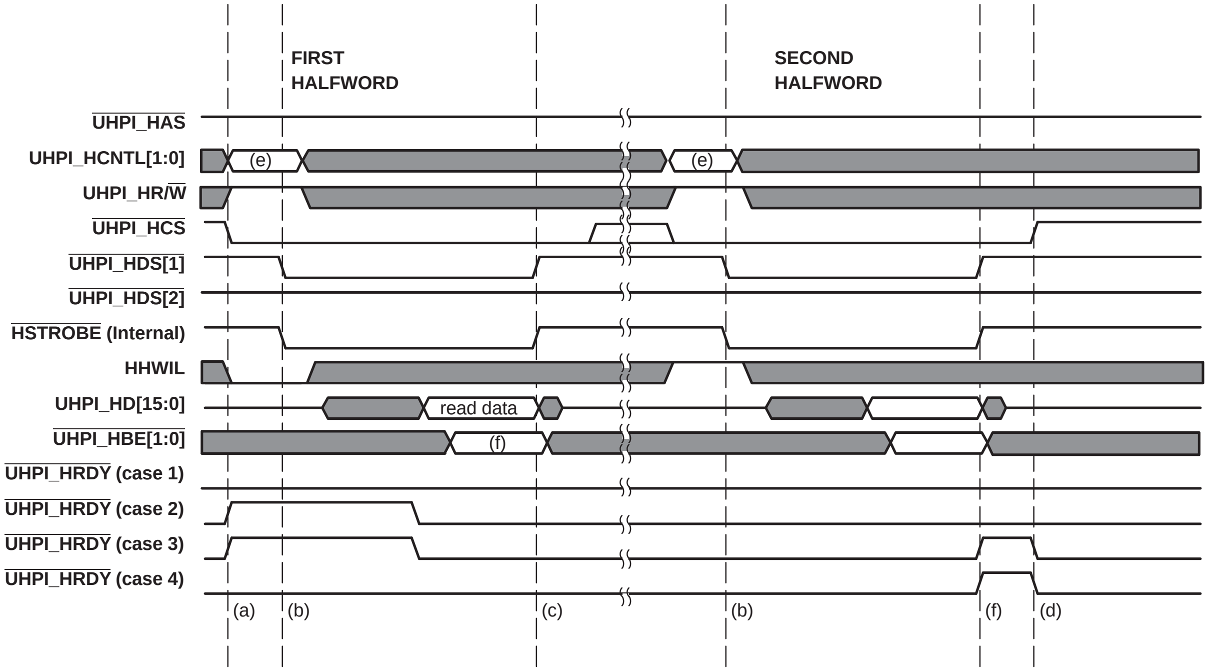
2.2.2 Multiplexed Host Address/Data Mode, Dual Halfword Option

This section describes host accesses in the multiplexed host address/data mode with dual halfword option.

2.2.2.1 Read Accesses

Figure 10 illustrates a host read from the UHPI in Multiplexed Host Address/Data, dual halfword mode using the falling edge of the internal HSTROBE signal to latch in the UHPI register address. Figure 11 illustrates the same access using the optional UHPI_HAS signal to latch in the register address.

Figure 10. Multiplexed Host Address/Data, Dual Halfword Read (without UHPI_HAS)



a In cases where the UHPI is initially not ready,UHPI_HRDY will be deasserted only while UHPI_HCS is asserted.

b UHPI latches UHPI_HCNTL[1:0], HHWIL, and UHPI_HRW on the falling edge of HSTROBE.

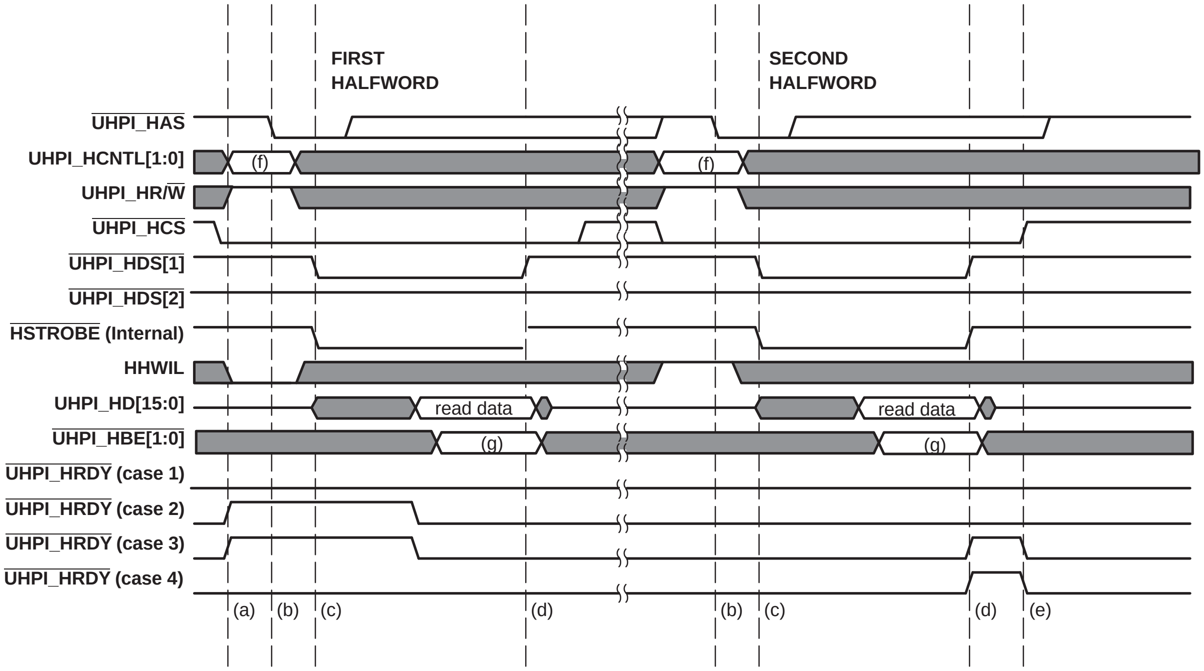
c Host latches UHPI read data on rising edge of HSTROBE

d In cases where the UHPI is not ready after the access, UHPI_HRDY remains deasserted until UHPI_HCS is deasserted.

e UHPI_HCNTL[1:0] selects register address: 00=**HPIC**, 01=**HPID** with postincrement, 10=**HPIA**, 11=**HPID** without postincrement

f UHPI_HBE[3:0] are ignored by the UHPI during reads

Figure 11. Multiplexed Host Address/Data, Dual Halfword Read (using UHPI_HAS)



a In cases where the UHPI is initially not ready,UHPI_HRDY will be deasserted only while UHPI_HCS is asserted.

b UHPI latches UHPI_HCNTL[1:0], HHWIL, and UHPI_HRW on the falling edge of UHPI_HAS.

c UHPI_HAS can be released anytime after the falling edge of HSTROBE is captured by the UHPI.

d Host latches UHPI read data on rising edge of HSTROBE.

e In cases where the UHPI is not ready after the access, UHPI_HRDY remains deasserted until UHPI_HCS is deasserted.

f UHPI_HCNTL[1:0] selects register address: 00=**HPIC**, 01=**HPID** with postincrement, 10=**HPIA**, 11=**HPID** without postincrement.

g UHPI_HBE[3:0] are ignored by the UHPI during reads.

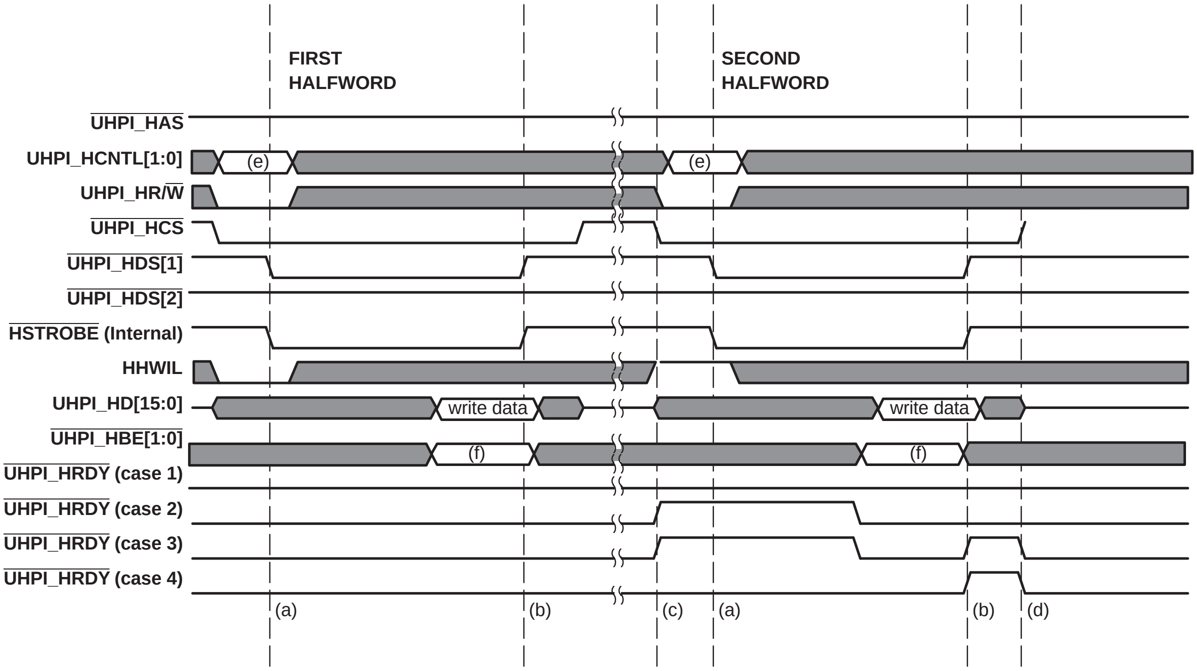
In Figure 10 and Figure 11 there are four different cases for the UHPI_HRDY signal, depending upon the UHPI state when the read is issued. The cases are:

- Case 1: UHPI_HRDY is never deasserted. This case occurs if either **HPIA** or **HPIC** is read. This case also occurs if an **HPID** with post increment read is issued, and the read FIFO is not empty.
- Case 2: UHPI_HRDY is initially deasserted. This case occurs if an **HPID** read without post increment is issued. It also occurs if an **HPID** read with post increment is issued but the read FIFO is empty or being flushed.
- Case 3: UHPI_HRDY is initially deasserted, and then is deasserted again after the read completes. This case occurs if the FIFO is empty or being flushed when an **HPID** read with postincrement is issued, and is empty again after the read completes.
- Case 4: UHPI_HRDY is initially asserted, but is deasserted after the read completes. This occurs when the read access empties the FIFO.

2.2.2.2 Write Accesses

Figure 12 illustrates a host write to the UHPI in multiplexed host address/data, dual halfword mode using the falling edge of the internal HSTROBE signal to latch in the UHPI register address. Figure 13 illustrates the same access using the optional UHPI_HAS signal to latch in the register address.

Figure 12. Multiplexed Host Address/Data Dual Halfword Write (without UHPI_HAS)



a UHPI latches UHPI_HCNTL[1:0], HHWIL, and UHPI_HRW on the falling edge of HSTROBE.

b UHPI latches in write data and byte enables on rising edge of HSTROBE

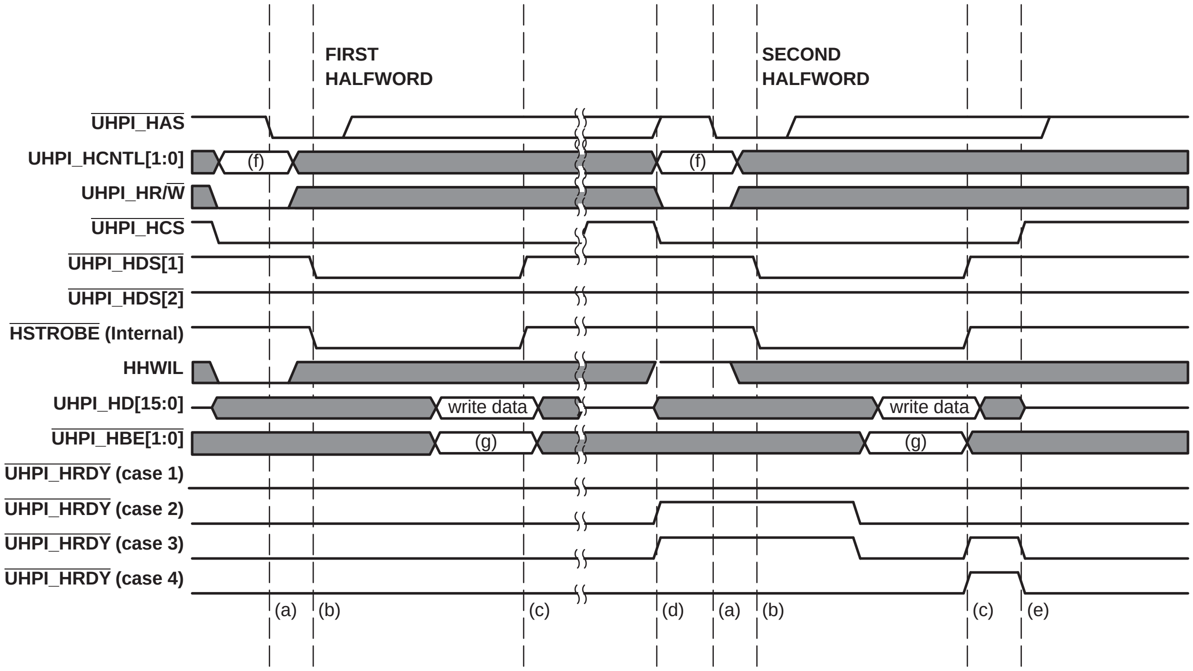
c In cases where the UHPI is initially not ready,UHPI_HRDY will be deasserted only while UHPI_HCS is asserted. This happens on the second halfword cycle for writes.

d In cases where the UHPI is not ready after the access, UHPI_HRDY remains deasserted until UHPI_HCS is deasserted.

e UHPI_HCNTL[1:0] selects register address: 00=**HPIC**, 01=**HPID** with postincrement, 10=**HPIA**, 11=**HPID** without postincrement

f Byte writes are only supported for **HPID without** post increment accesses. For all other accesses, all four of the UHPI_HBE[3:0] must be asserted. Two of the byte enables are latched in during each halfword.

Figure 13. Multiplexed Host Address/Data Dual Halfword Write (using UHPI_HAS)



a UHPI latches UHPI_HCNTL[1:0], HHWIL, and UHPI_HRW on the falling edge of UHPI_HAS.

b UHPI_HAS can be released any after the falling edge of HSTROBEis captured by the UHPI.

c UHPI latches in write data and byte enables on rising edge of HSTROBE

d In cases where the UHPI is initially not ready,UHPI_HRDY will be deasserted only while UHPI_HCS is asserted. This happens on the second halfword cycle for writes.

e In cases where the UHPI is not ready after the access, UHPI_HRDY remains deasserted until UHPI_HCS is deasserted.

f UHPI_HCNTL[1:0] selects register address: 00=**HPIC**, 01=**HPID** with post increment, 10=**HPIA**, 11=**HPID** without postincrement

g Byte writes are only supported for **HPID without** post increment accesses. For all other accesses, all four of the UHPI_HBE[3:0] must be asserted. Two of the byte enables are latched in during each halfword.

In Figure 12 and Figure 13 there are four different cases for the UHPI_HRDY signal, depending upon the UHPI state when the write is issued. The cases are:

- Case 1: UHPI_HRDY is never deasserted. This case will occur if **HPIC** is written, and in most cases if **HPIA** is written. This case also occurs if the write is to **HPID** with post increment and the write FIFO is not full.
- Case 2: UHPI_HRDY is initially deasserted. This case occurs during an **HPIA** write if a previous **HPIA** write has not yet been synchronized into the DSP SYSCLK2 domain. This case also occurs if an **HPID** write with post increment is issued but the write FIFO is full or being flushed.
- Case 3: UHPI_HRDY is initially deasserted, and then deasserted again after the write completes. The case occurs if an **HPID** write without post increment is issued, and the write FIFO is not empty or being flushed. This case can also occur if an **HPID** write with post increment is issued, and the FIFO is full both before and after the write.
- Case 4: UHPI_HRDY is initially asserted, but becomes deasserted after the write completes. This case occurs when the write FIFO is initially empty or being flushed and there is an **HPID** write without post increment. This case also occurs when the write FIFO is not full initially, but becomes full as a result of the write.

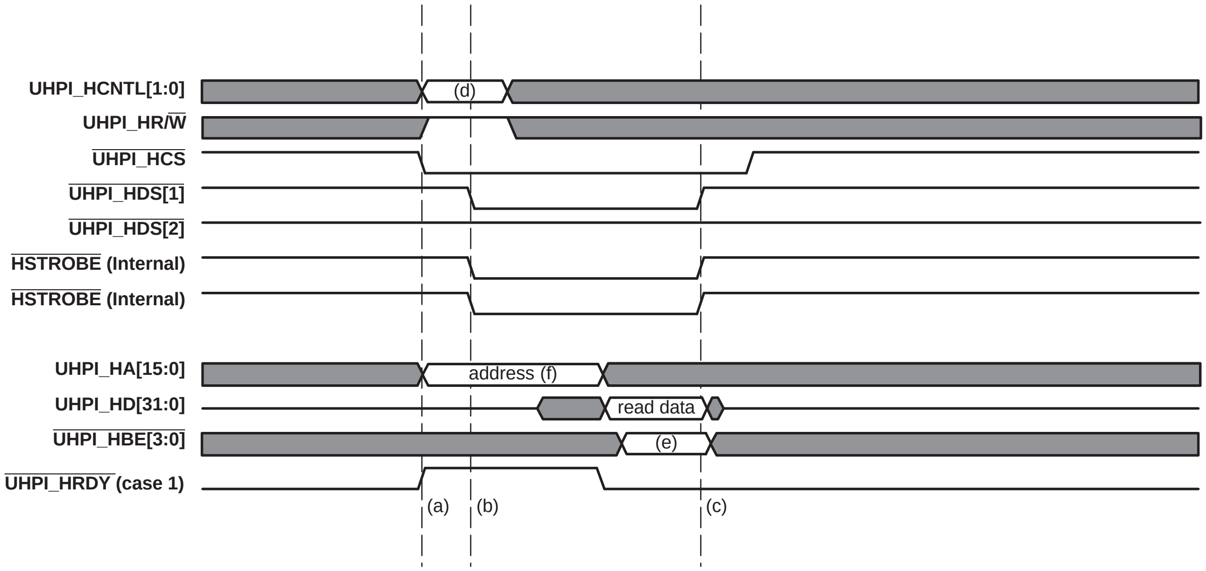
2.2.3 Non-Multiplexed Host Address/Data Mode, Fullword Option

This section describes host accesses in the multiplexed host address/data mode with dual halfword option.

2.2.3.1 Read Access

Figure 14 illustrates a host read from the UHPI in non-multiplexed host address/data, fullword mode using the falling edge of the internal HSTROBE signal to latch in the UHPI register address. The UHPI_HAS signal is not supported in non-multiplexed host address/data mode.

Figure 14. Non-Multiplexed Host Address/Data Fullword Read



a UHPI_HRDY will be deasserted initially when UHPI_HCS is asserted.

b UHPI latches UHPI_HCNTL[1:0], UHPI_HRW, and UHPI_HA[15:0] on the falling edge of HSTROBE.

c Host latches UHPI read data on rising edge of HSTROBE

d UHPI_HCNTL[1:0] selects register address: 00=**HPIC**, 01=reserved, 10=reserved, 11=**HPID** without postincrement

e UHPI_HBE[3:0] are ignored by the UHPI during reads

f The upper 16 bits of the Host Address are set by the DSP using the **CFGHPIAMSB**, **CFGHPIAUMB** registers

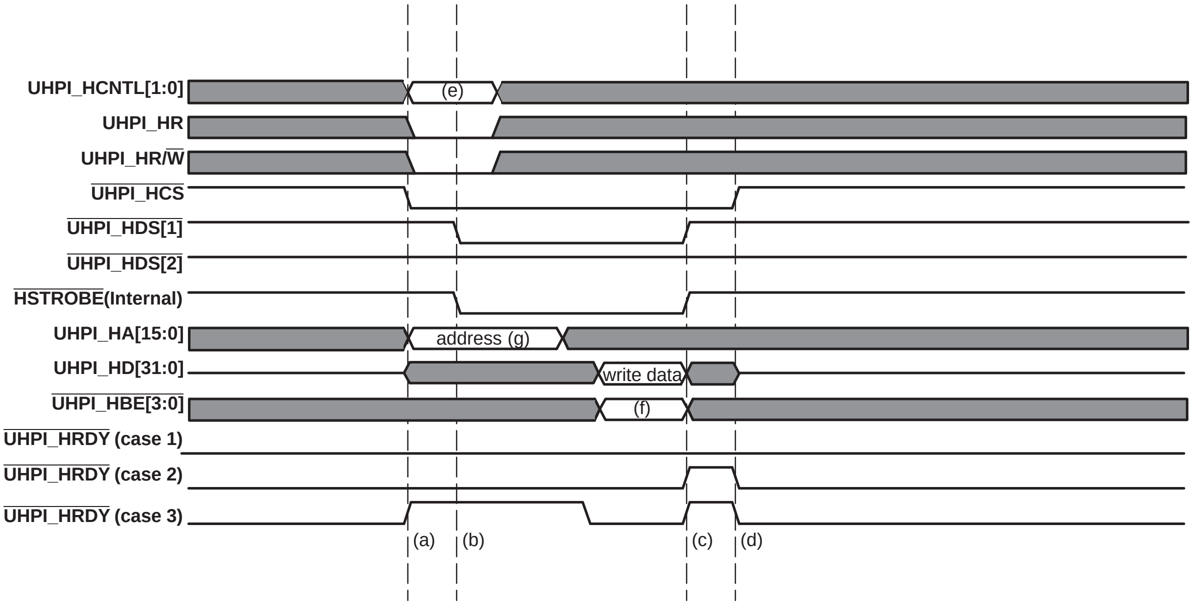
In Figure 14 there is only one case for the UHPI_HRDY signal:

- Case 1: UHPI_HRDY is never deasserted. This case occurs if either **HPIA** or **HPIC** is read. This case also occurs if an **HPID** with post increment read is issued, and the read FIFO is not empty.

2.2.3.2 Write Accesses

Figure 15 illustrates a host write to the UHPI in non-multiplexed host address/data, fullword mode using the falling edge of the internal HSTROBE signal to latch in the UHPI register address. The UHPI_HAS signal is not supported in non-multiplexed host address/data mode.

Figure 15. Non-Multiplexed Host Address/Data Fullword Write



a In cases where the UHPI is initially not ready, UHPI_HRDY will be deasserted only while UHPI_HCS is asserted.

b UHPI latches UHPI_HCNTL[1:0], UHPI_HRW, and UHPI_HA[15:0] on the falling edge of HSTROBE.

c UHPI latches in write data and byte enables on rising edge of HSTROBE.

d In cases where the UHPI is not ready after the access, UHPI_HRDY remains deasserted until UHPI_HCS is deasserted.

e UHPI_HCNTL[1:0] selects register address: 00=**HPIC**, 01=reserved 10=reserved, 11=**HPID** without postincrement.

f Byte writes are only supported for **HPID** accesses. For all other accesses, all four of the UHPI_HBE[3:0] must be asserted.

g The upper 16 bits of the Host Address are set by the DSP using the **CFGHPIAMSB**, **CFGHPIAUMB** registers.

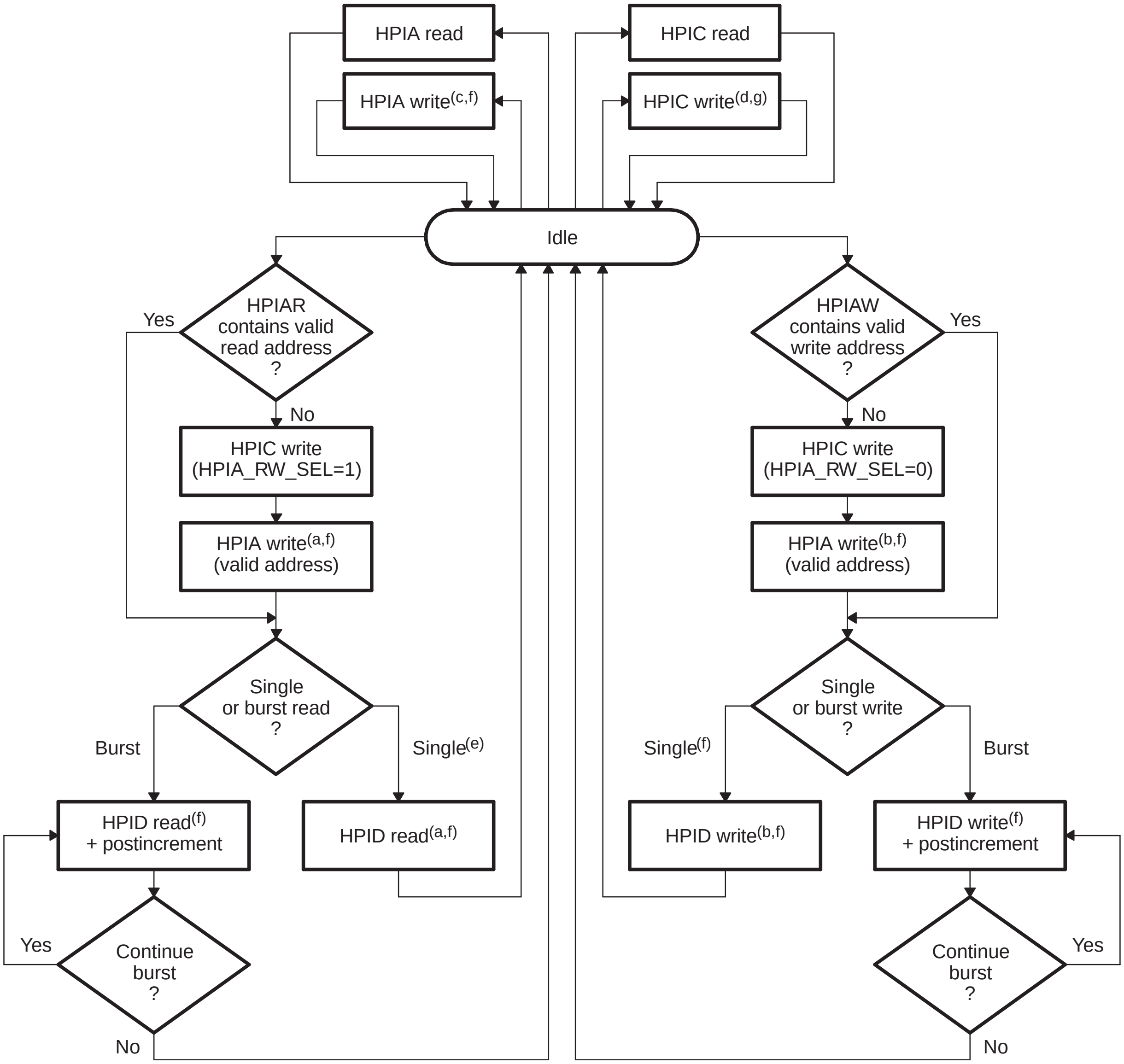
In Figure 15 there are three different cases for the UHPI_HRDY signal, depending upon the UHPI state when the write is issued. The cases are:

- Case 1: UHPI_HRDY is never deasserted. This case will occur if **HPIC** is written.
- Case 2: UHPI_HRDY is initially asserted, but is deasserted after the write completes. This case occurs when the write FIFO (only 1 word deep in this mode) is initially empty.
- Case 3: UHPI_HRDY is initially deasserted, and then deasserted again after the write completes. This case occurs when the write FIFO (only 1 word deep in this mode) is initially full.

2.3 State Diagrams of UHPI Operation

This section shows state diagrams for the expected use of the UHPI in both dual and single HPIA modes.

Figure 16. UHPI State Diagram for Dual HPIA Mode


HPIA read
HPIA write(c,f)
HPIC read
HPIC write(d,g)
Idle
HPIAR contains valid read address ?
Yes
No
HPIC write (HPIA_RW_SEL=1)
HPIA write(a,f) (valid address)
HPIAW contains valid write address ?
Yes
No
HPIC write (HPIA_RW_SEL=0)
HPIA write(b,f) (valid address)
Single or burst read ?
Burst
Single(e)
Single or burst write ?
Single(f)
Burst
HPID read(f) + postincrement
HPID read(a,f)
HPID write(b,f)
HPID write(f) + postincrement
Continue burst ?
Yes
No
Continue burst ?
Yes
No

a Causes FIFO Flush.

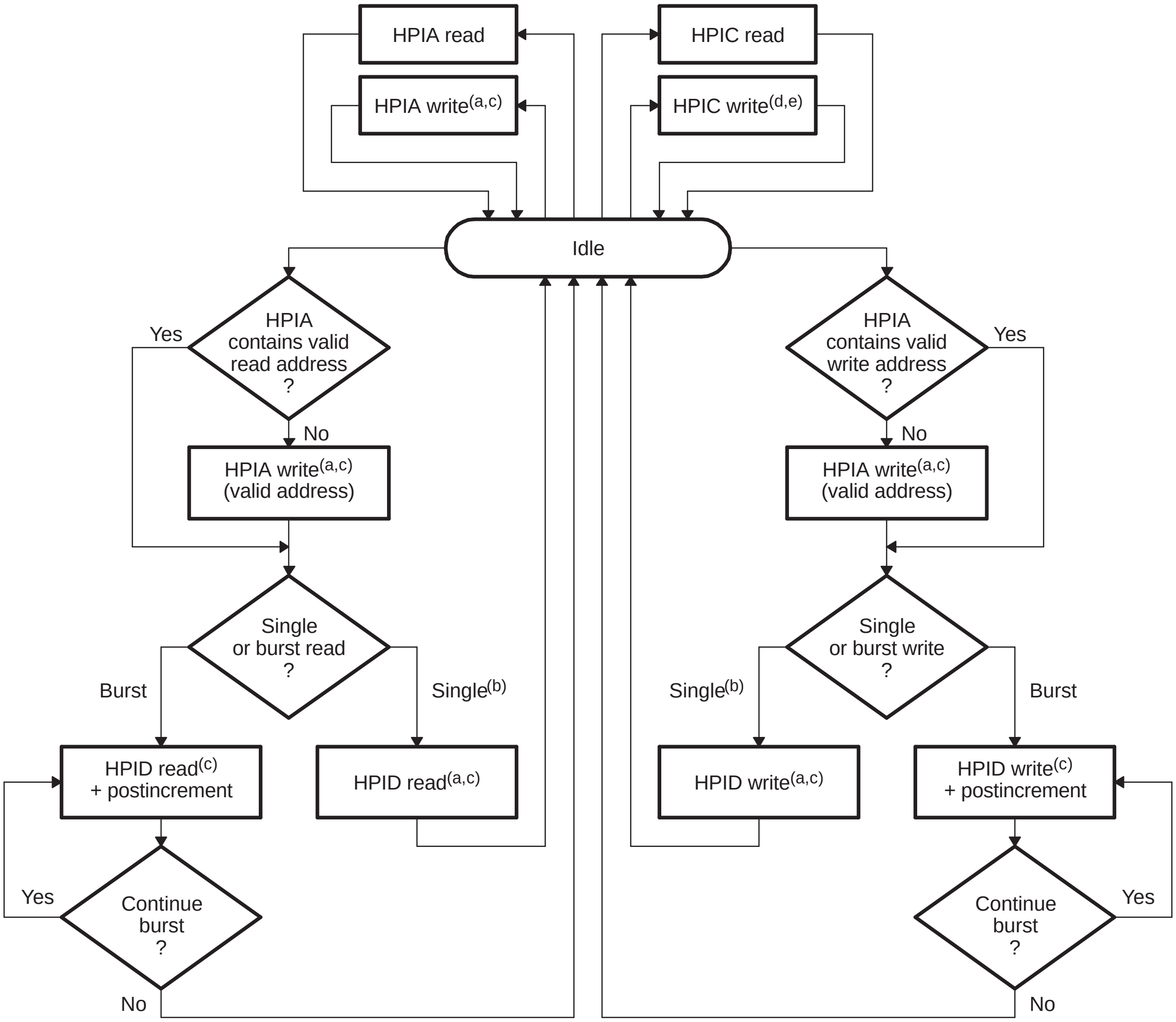
b Should be preceded by a write to HPIA unless the intent is to access the same address again. Must be preceded by an HPIA write if preceded by an HPID with post increment access.

c UHPI may stretch this operation by deasserting the $\overline{\text{UHPI\_HRDY}}$ pin.

d Writing to this register will place the FIFO in reset if the HPI_RST bit is set.

e Write to this register with the FETCH bit set to '1' must be preceded with either an HPIA write or an HPID without postincrement read.

Figure 17. UHPI State Diagram for Single HPIA Mode



a Causes a read FIFO Flush.

b Causes a write FIFO Flush.

c Causes a flush of the FIFO currently selected by the **HPIC.HPIA_RW_SEL** bit.

d Writing to this register will place the FIFO in reset if the HPI_RST bit is set.

e Should be preceded by a write to HPIA unless the intent is to access the same address again. Must be preceded by an HPIA write if preceded by an HPID with post increment access.

f UHPI may stretch this operation by deasserting the $\overline{\text{UHPI\_HRDY}}$ pin.

g Write to this register with the FETCH bit set to '1' must be preceded with either an HPIA write or an HPID without postincrement read.

2.4 UHPI Host Access Sequences

This section contains some examples of different UHPI access sequences and shows how the UHPI state changes during each access. In all of these examples, it is assumed that **CFGHPI.BYTEAD**=0 and **CFGHPI.PAGEM**=0.

2.4.1 Initialization Examples

Table 7 shows an example initialization for Multiplexed Host Address/Data Mode with dual HPIA and dual halfword options, in which the **HPIC**, **HPIAW**, and **HPIAR** registers are initialized, when **HPIC.HWOB**=0.

Table 7. Example Initialization 1

	Value During Access					Value After Access		
Event	**HD[15:0]**	**$\overline{\text{HBE}}$[1:0]**	**HR/$\overline{\text{W}}$**	**HCNTL[1:0]**	**HHWIL**	**HPIC**	**HPIAR**	**HPIAW**
Host Writes HPIC 1st HW	0200	xx	0	00	0	02000200	????????	????????
Host Writes HPIC 2nd HW	0200	xx	0	00	1	02000200	????????	????????
Host Writes HPIAW 1st HW	8000	xx	0	10	0	02000200	????????	8000????
Host Writes HPIAW 2nd HW	0000	xx	0	10	1	02000200	????????	80000000
Host Writes HPIC 1st HW	0A00	xx	0	00	0	0A000A00	????????	80000000
Host Writes HPIC 2nd HW	0A00	xx	0	00	1	0A000A00	????????	80000000
Host Writes HPIAR 1st HW	9000	xx	0	10	0	0A000A00	9000????	80000000
Host Writes HPIAR 2nd HW	0000	xx	0	10	1	0A000A00	90000000	80000000

Table 8 shows the same example as Table 7, except with HWOB=1.

Table 8. Example Initialization 2

	Value During Access					Value After Access		
Event	**HD[15:0]**	**$\overline{\text{HBE}}$[1:0]**	**HR/$\overline{\text{W}}$**	**HCNTL[1:0]**	**HHWIL**	**HPIC**	**HPIAR**	**HPIAW**
Host Writes HPIC 1st HW	0201	xx	0	00	0	03010301	????????	????????
Host Writes HPIC 2nd HW	0201	xx	0	00	1	03010301	????????	????????
Host Writes HPIAW 1st HW	0000	xx	0	10	0	03010301	????????	????0000
Host Writes HPIAW 2nd HW	8000	xx	0	10	1	03010301	????????	80000000
Host Writes HPIC 1st HW	0A01	xx	0	00	0	0B010B01	????????	80000000
Host Writes HPIC 2nd HW	0A01	xx	0	00	1	0B010B01	????????	80000000
Host Writes HPIAR 1st HW	0000	xx	0	10	0	0B010B01	????0000	80000000
Host Writes HPIAR 2nd HW	9000	xx	0	10	1	0B010B01	90000000	80000000

Table 9 shows the same example as Table 8, except this time the single HPIA option is chosen.

Table 9. Example Initialization 3

	Value During Access					Value After Access		
Event	HD[15:0]	$\overline{\text{HBE}}$[1:0]	HR/$\overline{\text{W}}$	HCNTL[1:0]	HHWIL	HPIC	HPIAR	HPIAW
Host Writes HPIC 1st HW	0001	xx	0	00	0	01010101	????????	????????
Host Writes HPIC 2nd HW	0001	xx	0	00	1	01010101	????????	????????
Host Writes HPIA 1st HW	0000	xx	0	10	0	01010101	????0000	????0000
Host Writes HPIA 2nd HW	8000	xx	0	10	1	01010101	80000000	80000000

Table 10 shows the same example as Table 9, except this time the fullword mode is used.

Table 10. Example Initialization 4

	Value During Access				Value After Access		
Event	HD[31:0]	$\overline{\text{HBE}}$[1:0]	HR/$\overline{\text{W}}$	HCNTL[1:0]	HPIC	HPIAR	HPIAW
Host Writes HPIC	00010001	xx	0	00	01010101	????????	????????
Host Writes HPIA	80000000	xx	0	10	01010101	80000000	80000000

2.4.2 Data Transfer Examples

Table 11 shows an example initialization for Multiplexed Host Address/Data Mode with dual HPIA and dual halfword options, in which the **HPIC**, **HPIAW**, and **HPIAR** registers are initialized, with **HPIC.HWOB**=0. Also, memory is filled such that each word contains the inverted value of its address. For example, at address 0x90000000 the data value would be 0x6FFFFFFF.

Table 11. Data Transfer Example - Dual HPIA Option

	Value During Access					Value After Access		
Event	HD[15:0]	$\overline{\text{HBE}}$[1:0]	HR/$\overline{\text{W}}$	HCNTL[1:0]	HHWIL	HPIC	HPIAR	HPIAW
Host Writes HPIC 1st HW	0200	xx	0	00	0	02000200	????????	????????
Host Writes HPIC 2nd HW	0200	xx	0	00	1	02000200	????????	????????
Host Writes HPIAW 1st HW	8000	xx	0	10	0	02000200	????????	8000????
Host Writes HPIAW 2nd HW[1]	0000	xx	0	10	1	02000200	????????	80000000
Host Writes HPIC 1st HW	0A00	xx	0	00	0	0A000A00	????????	80000000
Host Writes HPIC 2nd HW	0A00	xx	0	00	1	0A000A00	????????	80000000
Host Writes HPIAR 1st HW	9000	xx	0	10	0	0A000A00	9000????	80000000
Host Writes HPIAR 2nd HW[2]	0000	xx	0	10	1	0A000A00	90000000	80000000
Host Reads HPID with post inc. 1st HW [3]	6FFF	xx	1	01	0	0A000A00	90000000	80000000

[1] This access causes the write FIFO to be flushed.

[2] This access causes the read FIFO to be flushed

[3] This access causes the UHPI to create a burst request for 8 words of data beginning at address 0x90000000. The host will be wait stated until the FIFO has been filled.

Table 11. Data Transfer Example - Dual HPIA Option (continued)

	Value During Access					Value After Access		
Event	HD[15:0]	$\overline{\text{HBE[1:0]}}$	HR/$\overline{\text{W}}$	HCNTL[1:0]	HHWIL	HPIC	HPIAR	HPIAW
Host Reads HPID with post inc. 2nd HW	FFFF	xx	1	01	1	0A000A00	90000004	80000000
Host Reads HPID with post inc. 1st HW	FFFB	xx	1	01	0	0A000A00	90000004	80000000
Host Reads HPID with post inc. 2nd HW	6FFF	xx	1	01	1	0A000A00	90000008	80000000
Host Writes HPID with post inc. 1st HW	1234	00	0	01	0	0A000A00	90000008	80000000
Host Writes HPID with post inc. 2nd HW	5378	00	0	01	1	0A000A00	90000008	80000004
Host Writes HPID with post inc.t 1st HW	9ABC	00	0	01	0	0A000A00	90000008	80000004
Host Writes HPID with post inc. 2nd HW[4]	DEF0	00	0	01	1	0A000A00	90000008	80000008
Host Reads HPID with post inc. 1st HW	FFF7	xx	1	01	0	0A000A00	90000008	80000008
Host Reads HPID with post inc. 2nd HW	6FFF	xx	1	01	1	0A000A00	9000000C	80000008
Host Reads HPID with post inc. 1st HW	FFF3	xx	1	01	0	0A000A00	90000008	80000008
Host Reads HPID with post inc. 2nd HW[5]	6FFF	xx	1	01	1	0A000A00	90000010	80000008
Host Writes HPID with post inc. 1st HW	5555	00	0	01	0	0A000A00	90000010	80000008
Host Writes HPID with post inc. 2nd HW	5555	00	0	01	1	0A000A00	90000010	8000000C
Host Writes HPID with post inc.t 1st HW	AAAA	00	0	01	0	0A000A00	90000010	8000000C
Host Writes HPID with post inc. 2nd HW[6]	AAAA	00	0	01	1	0A000A00	90000010	80000010

[4] Write FIFO is queued up to write 0x12345678 to address 0x80000000, and 0x9ABCDEF0 to address 0x80000004. But the write doesn't happen until either the host writes two additional words to the FIFO, the FIFO is flushed, or the write FIFO timeout counter expires.

[5] This read results in the read FIFO having four empty slots, the threshold for issuing another read request. The UHPI issues a request for a burst of four more words starting from address 0x90000020 (remember, data from 0x90000010 - 0x9000001F was already fetched in the initial 8 word burst.

[6] This write results in four words having been written to the FIFO by the host. Assuming that the write FIFO timeout counter has not already expired, this write will cause the write FIFO to issue a bus request to write a burst of the four data words 0x12345678, 0x9ABCDEF0, 0x55555555, 0xAAAAAAAA to addresses 0x80000000 through 0x8000000F. Note that if the host had tried to read from addresses 0x80000000 through 0x8000000F prior to this burst completing, the previous values would have been returned by the read and *not the values queued up in the FIFO*.

Table 12 shows how the same operations would be performed in the multiplexed host address/data mode with single HPIA and dual halfword options in which the **HPIC**, **HPIAW**, and **HPIAR** registers are initialized, with **HPIC.HWOB**=0. Also, memory is filled such that each word contains the inverted value of its address. For example, at address 0x90000000 the data value would be 0x6FFFFFFF.

Table 12. Data Transfer Example - Single HPIA Option

	Value During Access					Value After Access		
Event	**HD[15:0]**	**$\overline{\text{HBE}}$[1:0]**	**HR/$\overline{\text{W}}$**	**HCNTL[1:0]**	**HHWIL**	**HPIC**	**HPIAR**	**HPIAW**
Host Writes HPIC 1st HW	0200	xx	0	00	0	02000200	????????	????????
Host Writes HPIC 2nd HW	0200	xx	0	00	1	02000200	????????	????????
Host Writes HPIA 1st HW	9000	xx	0	10	0	02000200	9000????	9000????
Host Writes HPIA 2nd HW[1]	0000	xx	0	10	1	02000200	90000000	90000000
Host Reads HPID with post inc. 1st HW [2]	6FFF	xx	1	01	0	02000200	90000000	90000000
Host Reads HPID with post inc. 2nd HW	FFFF	xx	1	01	1	02000200	90000004	90000004
Host Reads HPID with post inc. 1st HW	FFFB	xx	1	01	0	02000200	90000004	90000004
Host Reads HPID with post inc. 2nd HW	6FFF	xx	1	01	1	02000200	90000008	90000008
Host Writes HPIA 1st HW	8000	xx	0	10	0	02000200	8000????	8000????
Host Writes HPIA 2nd HW[3]	0000	xx	0	10	1	02000200	80000000	80000000
Host Writes HPID with post inc. 1st HW	1234	00	0	01	0	02000200	80000000	80000000
Host Writes HPID with post inc. 2nd HW	5678	00	0	01	1	02000200	80000004	80000004
Host Writes HPID with post inc.t 1st HW	9ABC	00	0	01	0	02000200	80000004	80000004
Host Writes HPID with post inc. 2nd HW[4]	DEF0	00	0	01	1	02000200	80000008	80000008
Host Writes HPIA 1st HW	9000	xx	0	10	0	02000200	9000????	9000????
Host Writes HPIA 2nd HW[5]	0008	xx	0	10	1	02000200	90000008	90000008

[1] This access causes the read and write FIFOs to be flushed.

[2] This access causes the UHPI to create a burst request for 8 words of data beginning at address 0x90000000. The host will be wait stated until the FIFO has been filled.

[3] This access causes the read and write FIFOs to be flushed. The data in the read FIFO from addresses 0x90000008 through 0x9000001F is discarded.

[4] Write FIFO is queued up to write 0x12345678 to address 0x80000000, and 0x9ABCDEF0 to address 0x80000004. But the write doesn't happen until either the host writes two additional words to the FIFO, the write FIFO is flushed, or the write FIFO timeout counter expires.

[5] This access causes the read and write FIFOs to be flushed, and the UHPI creates a burst write of two words to move data already in the FIFO to memory.

Table 12. Data Transfer Example - Single HPIA Option (continued)

Event	Value During Access					Value After Access		
	HD[15:0]	$\overline{\text{HBE}[1:0]}$	HR/$\overline{\text{W}}$	HCNTL[1:0]	HHWIL	HPIC	HPIAR	HPIAW
Host Reads HPID with post inc. 1st HW[6]	FFF7	xx	1	01	0	02000200	90000008	90000008
Host Reads HPID with post inc. 2nd HW	6FFF	xx	1	01	1	02000200	9000000C	9000000C
Host Reads HPID with post inc. 1st HW	FFF3	xx	1	01	0	02000200	9000000C	9000000C
Host Reads HPID with post inc. 2nd HW	6FFF	xx	1	01	1	02000200	90000010	90000010
Host Writes HPIA 1st HW	8000	xx	0	10	0	02000200	8000????	8000????
Host Writes HPIA 2nd HW[7]	0008	xx	0	10	1	02000200	80000008	80000008
Host Writes HPID with post inc. 1st HW	5555	00	0	01	0	02000200	80000008	80000008
Host Writes HPID with post inc. 2nd HW	5555	00	0	01	1	02000200	8000000C	8000000C
Host Writes HPID with post inc.t 1st HW	AAAA	00	0	01	0	02000200	8000000C	8000000C
Host Writes HPID with post inc. 2nd HW[8]	AAAA	00	0	01	1	02000200	80000010	80000010

[6] This access causes the UHPI to create a burst request for 8 words of data beginning at address 0x90000008. The host will be wait stated until the FIFO has been filled.

[7] This access causes the read and write FIFOs to be flushed. The data in the read FIFO from addresses 0x90000010 through 0x90000027 is discarded.

[8] Write FIFO is queued up to write 0x55555555 to address 0x80000008, and 0xAAAAAAAA to address 0x8000000C. But the write doesn't happen until either the host writes two additional words to the FIFO, the write FIFO is flushed, or the write FIFO timeout counter expires.

2.5 UHPI Access to C672x Resources

All host accesses to the **HPID** register are routed to the UHPI internal bus master logic. This results in a bus access to the crossbar switch which allows the UHPI to access other resources on the C672x device. The UHPI is bus master M5 in the C672x memory system. As illustrated in Figure 18, the external host can access (through the UHPI):

- RAM and ROM through the CSP port of the C672x device
- Memory and/or other devices on the 672x EMIF

The external host does not have access to:

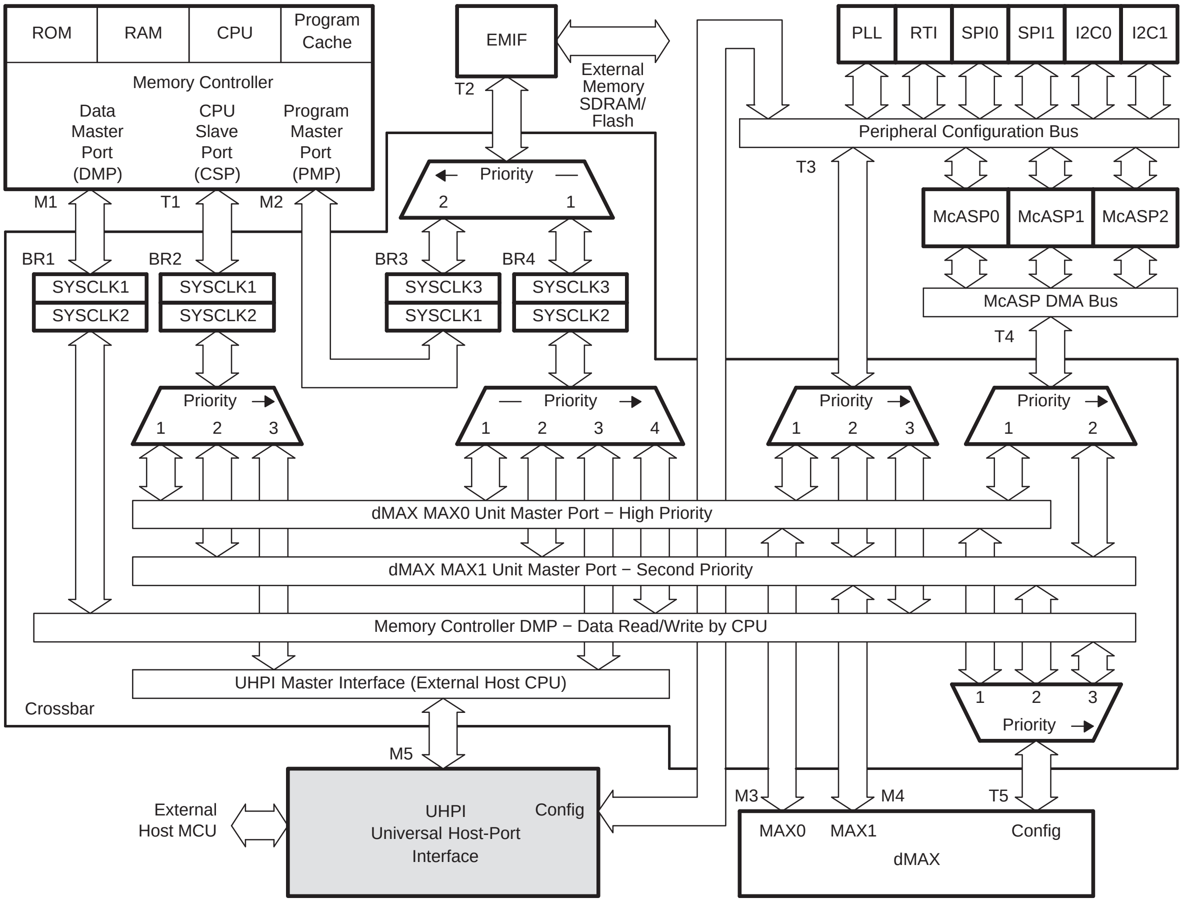
- dMAX configuration
- Internal peripheral configuration
- McASP DMA ports

The priority of external host accesses relative to other bus masters (the dMAX, the DSP CPU) are also shown in Figure 18. This priority only comes into play when two bus masters attempt to access the same resources.

When accessing internal ROM and RAM through the CSP port, the external host has third priority after the two dMAX bus masters. The external host has higher priority than the DSP CPU to the internal RAM/ROM by the nature of the CSP port definition.

For EMIF accesses, the UHPI has third priority again. Accesses originating from the two dMAX bus masters take higher priority than the UHPI. Likewise, the UHPI takes priority over the CPU and program cache when arbitrating for the EMIF.

Figure 18. UHPI Position in C672x Crossbar Switch Memory System



2.6 Interrupts between the DSP or dMAX and the External Host

Both host to DSP or dMAX and DSP or dMAX to host interrupts are available through the UHPI. This section describes the interrupt functionality of the UHPI.

2.6.1 DSP or dMAX to Host Interrupts

The DSP/dMAX can generate an interrupt to the external host by writing a '1' to the **HPIC.HINT** bit. This causes the UHPI_HINT bit to be driven active (low).

The external host must acknowledge the interrupt and write a '1' again to the **HPIC.HINT** bit to clear the bit and drive the UHPI_HINT back to an inactive level (high).

Writing a '0' to the **HPIC.HINT** has no effect.

Before writing a '1' to the **HPIC.HINT** bit, the DSP/dMAX should read the bit to determine if the external host has acknowledged the previous interrupt. If the **HPIC.HINT** bit is still set, then the DSP/dMAX should wait until the host clears the interrupt bit before attempting to set it again. Otherwise an interrupt may be lost.

2.6.2 Host to DSP or dMAX Interrupts

The host can generate an interrupt to the DSP/dMAX by writing a '1' to the **HPIC.DSP_INT** bit. This causes the DSP interrupt INT6 to be set. It also causes a high going pulse to dMAX event input 12.

The DSP or dMAX must acknowledge the interrupt and write a '1' again to the **HPIC.DSP_INT** bit to clear the bit ***after*** performing the ISR or dMAX function and clearing the local interrupt / event status.

Writing a '0' to the **HPIC.DSP_INT** has no effect.

Before writing a '1' to the **HPIC.DSP_INT** bit, the host should read the bit to determine if the DSP or dMAX has acknowledged the previous interrupt. If the **HPIC.DSP_INT** bit is still set, then the host should wait until the DSP or dMAX clears the interrupt bit before attempting to set it again. Otherwise an interrupt may be lost.

3 Register Map

3.1 Register Table

The UHPI registers on C672x devices are listed in Table 13. Some of the registers are located inside the UHPI peripheral, while others are located at the device level but still affect the operation of the UHPI peripheral on the C672x DSP.

Table 13. UHPI Configuration Registers

Byte Address	Register Name	Description
Device-Level Configuration Registers Controlling UHPI		
0x4000 0008	CFGHPI	UHPI Configuration Register
0x4000 000C	CFGHPIAMSB	Most Significant Byte of UHPI Address
0x4000 0010	CFGHPIAUMB	Upper Middle Byte of UHPI Address
UHPI Internal Registers		
0x4300 0000	PID	Peripheral ID Register
0x4300 0004	PWREMU	Power and Emulation Management Register
0x4300 0008	GPIOINT	General Purpose I/O Interrupt Control Register
0x4300 000C	GPIOEN	General Purpose I/O Enable Register
0x4300 0010	GPIODIR1	General Purpose I/O Direction Register 1
0x4300 0014	GPIODAT1	General Purpose I/O Data Register 1
0x4300 0018	GPIODIR2	General Purpose I/O Direction Register 2
0x4300 001C	GPIODAT2	General Purpose I/O Data Register 2
0x4300 0020	GPIODIR3	General Purpose I/O Direction Register 3
0x4300 0024	GPIODAT3	General Purpose I/O Data Register 3
0x4300 0028	Reserved	Reserved
0x4300 002C	Reserved	Reserved
0x4300 0030	HPIC	Control Register
0x4300 0034	HPIAW	Write Address Register
0x4300 0038	HPIAR	Read Address Register
0x4300 003C	Reserved	Reserved
0x4300 0040	Reserved	Reserved

3.2 C672x UHPI Configuration Register (CFGHPI) (0x4000 0008)

This register is used to select the UHPI operating mode and then to enable the UHPI. By default the UHPI is disabled after a device reset.

Figure 19. CFGHPI Register Bit Layout (0x4000 0008)

31 … 8
Reserved

7 … 5	4	3	2	1	0
Reserved	BYTEAD	FULL	NMUX	PAGEM	ENA
	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 14. CFGHPI Register Bit Field Description (0x4000 0008)

Bit No.	Name	Reset Value	Read Write	Description
31:5	Reserved	N/A	N/A	Reads are indeterminate. Only 0s should be written to these bits.
4	BYTEAD	0	R/W	UHPI Host Address Type 0 = Host Address is a word address 1 = Host Address is a byte address
3	FULL	0	R/W	UHPI Multiplexing Mode (when NMUX = 0) 0 = Half-Word (16-bit data) Multiplexed Host Address and Data Mode 1 = Fullword (32-bit data) Multiplexed Host Address and Data Mode
2	NMUX	0	R/W	UHPI Non-Multiplexed Mode Enable 0 = Multiplexed Host Address and Data Mode 1 = Non-Multiplexed Host Address and Data Mode (utilizes optional UHPI_HA[15:0] pins). Host data bus is 32 bits in Non-Multiplexed mode.
1	PAGEM	0	R/W	UHPI Page Mode Enable (Only for Multiplexed Host Address and Data Mode). 0 = Full 32-bit DSP address specified through host port. 1 = Only lower 16 bits of DSP address are specified through host port. Upper 16 bits are restricted to the page selected by **CFGHPIAMSB** and **CFGHPIAUMB** registers.
0	ENA	0	R/W	UHPI Enable 0 = UHPI is disabled 1 = UHPI is enabled. Set this bit to '1' only after configuring the other bits in this register.

3.3 C672x UHPI Address Most Significant Byte Register (CFGHPIAMSB) (0x4000 000C)

Figure 20. CFGHPIAMSB Register Bit Layout (0x4000 000C)

31	8
Reserved	

7	0
HPIAMSB	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 15. CFGHPIAMSB Register Bit Field Description (0x4000 000C)

Bit No.	Name	Reset Value	Read Write	Description
31:8	Reserved	N/A	N/A	Reads are indeterminate. Only 0s should be written to these bits.
7:0	HPIAMSB	0	R/W	UHPI most significant byte of DSP address to access in non-multiplexed host address/data mode and in multiplexed host address and data mode when **CFGHPI.PAGEM** = 1. Sets bits [31:24] of the DSP internal address as accessed through UHPI.

3.4 C672x UHPI Address Upper Middle Byte Register (CFGHPIAUMB) (0x4000 0010)

Figure 21. CFGHPIAUMB Register Bit Layout (0x4000 0010)

31	8
Reserved	

7	0
HPIAUMB	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 16. CFGHPIAUMB Register Bit Field Description (0x4000 0010)

Bit No.	Name	Reset Value	Read Write	Description
31:8	Reserved	N/A	N/A	Reads are indeterminate. Only 0s should be written to these bits.
7:0	HPIAUMB	0	R/W	UHPI upper middle byte of DSP address to access in non-multiplexed host address/data mode and in multiplexed host address and data mode when **CFGHPI.PAGEM** = 1. Sets bits [23:16] of the DSP internal address as accessed through UHPI.

3.5 Peripheral Identification Register (PID) (0x4300 0000)

This register identifies the module as a UHPI peripheral, and contains a module revision code.

Figure 22. UHPI PID Register Bit Layout (0x4300 0000)

31	24	23	16
Reserved		Type	
R		R-0001	

15	8	7	0
Class		Rev	
R-0010		R-yyyy	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 17. UHPI PID Register Bit Field Description (0x4300 0000)

Bit No.	Name	Reset Value	Read Write	Description
31:24	Reserved	N/A	N/A	Reserved
23:16	Type	0001	R	Indicates the peripheral type is UHPI (0x1)
15:8	Class	0010	R	Indicates the class of the peripheral as UHPI (0x02)
7:0	Revision	yyyy	R	Indicates the Revision of the UHPI Module is yyyy (Silicon Specific)

3.6 Power and Emulation Management Register (PWREMU) (0x4300 0004)

This register controls the behavior of the UHPI peripheral when emulation suspend is asserted. Note that emulation suspend features are not supported on the C672x device, therefore soft-stop mode is not supported.

Figure 23. UHPI PWREMU Register Bit Layout (0x4300 0004)

31	8
Reserved	
R-0	

7	2	1	0
Reserved		SOFT	FREE
R-0		R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 18. UHPI PWREMU Register Bit Field Description (0x4300 0004)

Bit No.	Name	Reset Value	Read Write	Description
31:2	Reserved	N/A	N/A	Reserved

1	SOFT	0	R/W	SOFT = 0: UHPI Continues to Operate During Emulation Suspend.
				SOFT = 1: NOT SUPPORTED ON C672x Devices.

0	FREE	0	R/W	FREE=0: The SOFT bit takes effect. FREE=1: Upon Emulation Suspend the peripheral operation is not affected (regardless of SOFT bit setting).

3.7 General Purpose I/O Interrupt Control Register (GPIOINT) (0x4300 0008)

This register controls the ability of UHPI GPIO pins to generate an interrupt. Note that the dMAX and CPU also need to be configured to receive an interrupt from the UHPI before a UHPI interrupt will occur.

Figure 24. UHPI GPIOINT Register Bit Layout (0x4300 0008)

31	19	18	17	16
Reserved		GPINT_INV2	GPINT_INV1	GPINT_INV0
R		R/W, 0	R/W, 0	R/W, 0

15	3	2	1	0
Reserved		GPINT_EN2	GPINT_EN1	GPINT_EN0
R		R/W, 0	R/W, 0	R/W, 0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 19. UHPI GPIOINT Register Bit Field Description (0x4300 0008)

Bit No.	Name	Reset Value	Read Write	Description
31:19	Reserved	N/A	N/A	
18	GPINT_INV2	0	R/W	GPINT_INV2=0: The $\overline{\text{UHPI\_HAS}}$ pin value drives dMAX Event 12 and CPU INT6. GPINT_INV2=1: The $\overline{\text{UHPI\_HAS}}$ inverted pin value drives dMAX Event 12 and CPU INT6.
17	GPINT_INV1	0	R/W	GPINT_EN1 is not supported on C672x Devices and '0' should be written to this bit.
16	GPINT_INV0	0	R/W	GPINT_EN0 is not supported on C672x Devices and '0' should be written to this bit.
15:3	Reserved	N/A	N/A	
2	GPINT_EN2	0	R/W	GPINT_EN2=0: Interrupts from the $\overline{\text{UHPI\_HAS}}$ pin are disabled GPINT_EN2=1: Interrupts from the $\overline{\text{UHPI\_HAS}}$ pin are enabled to generate dMAX Event 12 and CPU INT6
1	GPINT_EN1	0	R/W	GPINT_EN1 is not supported on C672x Devices and '0' should be written to this bit.
0	GPINT_EN0	0	R/W	GPINT_EN0 is not supported on C672x Devices and '0' should be written to this bit.

3.8 General Purpose I/O Enable Register (GPIOEN) (0x4300 000C)

This register enables UHPI pins for General Purpose I/O functionality.

Figure 25. UHPI GPIOEN Register1 Bit Layout (0x4300 00C)

31 … 17	16
Reserved	GPIO_EN16
R	R/W-0

15	14	13	12	11	10	9	8
GPIO_EN15	GPIO_EN14	GPIO_EN13	GPIO_EN12	GPIO_EN11	GPIO_EN10	GPIO_EN9	GPIO_EN8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
GPIO_EN7	GPIO_EN6	GPIO_EN5	GPIO_EN4	GPIO_EN3	GPIO_EN2	GPIO_EN1	GPIO_EN0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 20. UHPI GPIOEN Register Bit Field Description (0x4300 000C)

Bit No.	Name	Reset Value	Read Write	Description
31:17	Reserved	N/A	N/A	
16:13	GPIO_EN16 : GPIO_EN13	0000	0000	HA[31:28], HA[27:24], HA[23:20] and HA[19:16] are **unsupported** on the C672x. Write "0000" to these bits.

12	GPIO_EN12	0	0	GPIO_EN12=0: UHPI_HA[15:8] pins function as UHPI pins.
				GPIO_EN12=1: UHPI_HA[15:8] pins function as GPIO pins.

11	GPIO_EN11	0	0	GPIO_EN11=0: UHPI_HA[7:0] pins function as UHPI pins.
				GPIO_EN11=1: UHPI_HA[7:0] pins function as GPIO pins.

10	GPIO_EN10	0	0	GPIO_EN10=0: UHPI_HD[31:24] pins function as UHPI pins.
				GPIO_EN10=1: UHPI_HD[31:24] pins function as GPIO pins.

9	GPIO_EN9	0	0	GPIO_EN9=0: UHPI_HD[23:16] pins function as UHPI pins.
				GPIO_EN9=1: UHPI_HD[23:16] pins function as GPIO pins.

8	GPIO_EN8	0	0	GPIO_EN8=0: UHPI_HD[15:8] pins function as UHPI pins.
				GPIO_EN8=1: UHPI_HD[15:8] pins function as GPIO pins.

7	GPIO_EN7	0	0	GPIO_EN7=0: UHPI_HD[7:0] pins function as UHPI pins.
				GPIO_EN7=1: UHPI_HD[7:0] pins function as GPIO pins.

6	GPIO_EN6	0	0	GPIO_EN6=0: UHPI_HINT pin functions as a UHPI pin. GPIO_EN6=1: UHPI_HINT pin functions as a GPIO pin.
5	GPIO_EN5	0	0	GPIO_EN5=0: UHPI_HRDY pin functions as a UHPI pin. GPIO_EN5=1: UHPI_HRDY pin functions as a GPIO pin.
4	GPIO_EN4	0	0	UHPI_HHWIL pin **is not supported as GPIO** on C672x DSP devices through this bit. Write '0' to this register bit.

3	GPIO_EN3	0	0	GPIO_EN3=0: UHPI_HBE[3:0] pins function as UHPI pins.
				GPIO_EN3=1: UHPI_HBE[3:0] pins function as GPIO pins.

2	GPIO_EN2	0	0	GPIO_EN2=0: UHPI_HAS pin functions as a UHPI pin. GPIO_EN2=1: UHPI_HAS pin functions as a GPIO pin.

1	GPIO_EN1	0	0	GPIO_EN1=0: UHPI_HCNTL[1:0] pins function as UHPI pins.
				GPIO_EN1=1: UHPI_HCNTL[1:0] pins function as GPIO pins.

0	GPIO_EN0	0	0	GPIO_EN0=0: UHPI_HCS, UHPI_HDS[2:1], and UHPI_HR/Wfunction as UHPI pins.
				GPIO_EN0=1: UHPI_HCS, UHPI_HDS[2:1], and UHPI_HR/W function as GPIO pins.

3.9 General Purpose I/O Data Direction Register 1 (GPIODIR1) (0x4300 0010)

This register controls whether UHPI host data pins (UHPI_HD[31:0]) are general purpose inputs or outputs when configured as GPIO through the GPIOEN register Section 3.8.

Figure 26. UHPI GPIODIR1 Register Bit Layout (0x4300 0010)

31	0
DIR1[31:0]	
R/W-00000000000000000000000000000000	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 21. UHPI GPIODIR1 Register Bit Field Description (0x4300 0010)

Bit No.	Name	Reset Value	Read Write	Description
31:0	DIR1[31:0]	0	R/W	This register controls whether each of the UHPI_HD[31:0] pins are configured as general purpose input or general purpose output. The value of this register only takes effect when the pins are enabled as GPIO through the GPIOEN register Section 3.8. DIR1[n] = 0: UHPI_HD[n] is a general purpose input when enabled as GPIO through the GPIOEN register. DIR1[n] = 1: UHPI_HD[n] is a general purpose output when enabled as GPIO through the GPIOEN register.

3.10 General Purpose I/O Data Register 1 (GPIODAT1) (0x4300 0014)

This register controls the data on the UHPI_HD[31:0] pins if these pins are configured as general purpose outputs. If these pins are configured as general purpose inputs, then reading this register returns the value present on the pin.

Figure 27. UHPI GPIODAT1 Register Bit Layout (0x4300 0014)

31	0
DIR1[31:0]	
R/W-00000000000000000000000000000000	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 22. UHPI GPIODAT1 Register Bit Field Description (0x4300 0014)

Bit No.	Name	Reset Value	Read Write	Description
31:0	DAT1[31:0]	0	R/W	When the UHPI_HD[n] pin is configured as a general purpose input pin: (See Section 3.8 and Section 3.9), the value read from DAT1[n] represents the value input on the pin. Writing to DAT1[n] has no effect. When the UHPI_HD[n] pin is configured as a general purpose output pin, the value written to DAT1[n] will be driven out the UHPI_HD[n] pin. This value may be read back by reading the DAT1[n] register bit.

3.11 General Purpose I/O Data Direction Register 2 (GPIODIR2) (0x4300 0018)

This register controls whether UHPI control pins are general purpose inputs or outputs when configured as GPIO through the GPIOEN register Section 3.8.

Figure 28. UHPI GPIODIR2 Register Bit Layout (0x4300 0018)

31–16
Reserved
R

15	14	13	12	11	10	9	8
Reserved	DIR2[14]	DIR2[13]	DIR2[12]	DIR2[11]	DIR2[10]	Reserved	DIR2[8]
R	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
DIR2[7]	DIR2[6]	DIR2[5]	DIR2[4]	DIR2[3]	DIR2[2]	DIR2[1]	DIR2[0]
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 23. UHPI GPIODIR2 Register Bit Field Description (0x4300 0018)

Bit No.	Name	Reset Value	Read Write	Description
31:15	Reserved	N/A	R	
14	DIR2[14]	0	R/W	DIR2 bit for $\overline{\text{UHPI\_HBE[3]}}$
13	DIR2[13]	0	R/W	DIR2 bit for $\overline{\text{UHPI\_HBE[2]}}$
12	DIR2[12]	0	R/W	DIR2 bit for $\overline{\text{UHPI\_HBE[1]}}$
11	DIR2[11]	0	R/W	DIR2 bit for $\overline{\text{UHPI\_HBE[0]}}$
10	DIR2[10]	0	R/W	DIR2 bit for $\overline{\text{UHPI\_HRDY}}$
9	Reserved	0	R/W	This bit should always be written as '0'.
8	DIR2[8]	0	R/W	DIR2 bit for $\overline{\text{UHPI\_HINT}}$
7	DIR2[7]	0	R/W	DIR2 bit for UHPI_HCNTL[0]
6	DIR2[6]	0	R/W	DIR2 bit for UHPI_HCNTL[1]
5	DIR2[5]	0	R/W	DIR2 bit for UHPI_HHWIL
4	DIR2[4]	0	R/W	DIR2 bit for UHPI_HR/$\overline{\text{W}}$
3	DIR2[3]	0	R/W	DIR2 bit for $\overline{\text{UHPI\_HDS[2]}}$
2	DIR2[2]	0	R/W	DIR2 bit for $\overline{\text{UHPI\_HDS[1]}}$
1	DIR2[1]	0	R/W	DIR2 bit for $\overline{\text{UHPI\_HCS}}$
0	DIR2[0]	0	R/W	DIR2 bit for $\overline{\text{UHPI\_HAS}}$ When the UHPI control pin is configured as a general purpose input pin: (See Section 3.8 and Section 3.9), the value read from DAT2[n] represents the value input on the pin. Writing to DAT2[n] has no effect. When the UHPI control pin is configured as a general purpose output pin, the value written to DAT2[n] will be driven out the corresponding pin. This value may be read back by reading the DAT2[n] register bit.

3.12 General Purpose I/O Data Direction Register 2 (GPIODAT2) (0x4300 001C)

This register controls the data on the UHPI control pins if these pins are configured as general purpose outputs. If these pins are configured as general purpose inputs, then reading this register returns the value present on the pin.

Figure 29. UHPI GPIODAT2 Register Bit Layout (0x4300 001C)

31–16
Reserved
R

15	14	13	12	11	10	9	8
Reserved	DAT2[14]	DAT2[13]	DAT2[12]	DAT2[11]	DAT2[10]	Reserved	DAT2[8]
R	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
DAT2[7]	DAT2[6]	DAT2[5]	DAT2[4]	DAT2[3]	DAT2[2]	DAT2[1]	DAT2[0]
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 24. UHPI GPIODAT2 Register Bit Field Description (0x4300 001C)

Bit No.	Name	Reset Value	Read Write	Description
31:15	Reserved	N/A	R	
14	DAT2[14]	0	R/W	DAT2 bit for UHPI_HBE[3]
13	DAT2[13]	0	R/W	DAT2 bit for UHPI_HBE[2]
12	DAT2[12]	0	R/W	DAT2 bit for UHPI_HBE[1]
11	DAT2[11]	0	R/W	DAT2 bit for UHPI_HBE[0]
10	DAT2[10]	0	R/W	DAT2 bit for UHPI_HRDY
9	Reserved	0	R/W	This bit should always be written as '0'.
8	DAT2[8]	0	R/W	DAT2 bit for UHPI_HINT
7	DAT2[7]	0	R/W	DAT2 bit for UHPI_HCNTL[0]
6	DAT2[6]	0	R/W	DAT2 bit for UHPI_HCNTL[1]
5	DAT[5]	0	R/W	DAT2 bit for UHPI_HHWIL
4	DAT2[4]	0	R/W	DAT2 bit for UHPI_HR/W
3	DAT2[3]	0	R/W	DAT2 bit for UHPI_HDS[2]
2	DAT2[2]	0	R/W	DAT2 bit for UHPI_HDS[1]
1	DAT2[1]	0	R/W	DAT2 bit for UHPI_HCS
0	DAT2[0]	0	R/W	DAT2 bit for UHPI_HAS

3.13 General Purpose I/O Data Direction Register 3 (GPIODIR3) (0x4300 0020)

This register controls whether UHPI host address pins (UHPI_HA[15:0]) are general purpose inputs or outputs when configured as GPIO through the GPIOEN register Section 3.8.

Figure 30. UHPI GPIODIR3 Register Bit Layout (0x4300 0020)

31 ... 0
DIR3[31:0]
R/W-00000000000000000000000000000000

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 25. UHPI GPIODIR3 Register Bit Field Description (0x4300 0020)

Bit No.	Name	Reset Value	Read Write	Description
31:16	DIR3[31:16]	0	R/W	UHPI_HA[31:16] are not supported on C672x devices. A value of '0' should be written to these register bits.
15:0	DIR3[15:0]	0	R/W	This register controls whether each of the UHPI_HA[15:0] pins are configured as general purpose input or general purpose output. The value of this register only takes effect when the pins are enabled as GPIO through the GPIOEN register Section 3.8. DIR3[n] = 0: UHPI_HA[n] is a general purpose input when enabled as GPIO through the GPIOEN register. DIR3[n] = 1: UHPI_HA[n] is a general purpose output when enabled as GPIO through the GPIOEN register.

3.14 General Purpose I/O Data Direction Register 3 (GPIODAT3) (0x4300 0024)

This register controls the data on the UHPI_HD[31:0] pins if these pins are configured as general purpose outputs. If these pins are configured as general purpose inputs, then reading this register returns the value present on the pin.

Figure 31. UHPI GPIODAT3 Register Bit Layout (0x4300 0024)

31 ... 0
DAT3[31:0]
R/W-00000000000000000000000000000000

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 26. UHPI GPIODAT3 Register Bit Field Description (0x4300 0024)

Bit No.	Name	Reset Value	Read Write	Description
31:16	DAT3[31:16]	0	R/W	UHPI_HA[31:16] are not supported on C672x devices. A value of '0' should be written to these register bits.
15:0	DAT3[15:0]	0	R/W	When the UHPI_HA[n] pin is configured as a general purpose input pin: (See Section 3.8 and Section 3.9), the value read from DAT3[n] represents the value input on the pin. Writing to DAT3[n] has no effect. When the UHPI_HA[n] pin is configured as a general purpose output pin, the value written to DAT3[n] will be driven out the UHPI_HA[n] pin. This value may be read back by reading the DAT3[n] register bit.

3.15 UHPI Control Register (HPIC) (0x4300 0030)

This register is shared between the host and DSP. The address given is the address from the DSP perspective. The host selects this register by driving the UHPI.HCNTL[1:0] pins to "00" during the access.

Figure 32. UHPI HPIC Register Bit Layout (0x4300 0030)

31	12	11	10	9	8
Reserved		HPIA_RW_SEL	LB_MODE	DUAL_HPIA	HWOB_STAT
R		R/W, 0	R, 0	R/W, 0	R, 0

7	6	5	4	3	2	1	0
HPI_RST	RESET	XHPIA	FETCH	HRDY	HINT	DSP_INT	HWOB
R/W, 1	R/W, 0	R/W, 0	W,0	R, 0	R/W, 0	R/W,0	R/W,0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 27. UHPI HPIC Register Bit Field Description (0x4300 0030)

Bit No.	Name	Reset Value	Read Write Host	Read Write DSP	Description
31:12	Reserved	-	-	-	Reserved
11	HPIA_RW_SEL	0	R/W	R	When **HPIC.DUAL_HPIA** = '1', this bit selects whether **HPIA** cycles address **HPIAR** if **HPIC.HPIA_RW_SEL** = '1' or **HPIAW** if **HPIC.HPIA_RW_SEL** = '0'.
10	LB_MODE	0	R	R	Loopback mode is **unsupported** on C672x devices.
9	DUAL_HPIA	0	R/W	R	Writing '1' to this bit selects dual (**HPIAR**, **HPIAW**) mode. Writing '0' to this bit selects a single (**HPIA**) view of the registers.
8	HWOB_STAT	0	R	R	Read only duplicate of **HPIC.HWOB**.
7	HPI_RST	1	-	R/W	UHPI is held in reset by default, and must be released by the DSP writing '0' to this bit.
6	RESET	0	R/W	R	Host resetting the C672x through UHPI is **unsupported**. Write '0' to this bit.
5	XHPIA	0	R/W	R	**XHPIA** mode is **unsupported** on C672x devices. Write '0' to this bit.
4	FETCH	0	W-1	-	Writing '1' to this bit causes the UHPI to prefetch read data. Reads return '0'.
3	HRDY[1][2]	0	R	-	This bit is not fully supported on the C672x DSP. The HPIC.HRDY option should not be used. Instead, the host must use the UHPI_HRDY pin to extend accesses whenever the UHPI is not ready.
2	HINT[3]	0	R/W	R/W	DSP to Host Interrupt. Writing a '1' to this bit generates an interrupt through the AMUTE2/HINT pin.
1	DSP_INT	0	R/W	R/W	Host to DSP Interrupt. Writing a '1' to this bit will generate an interrupt to the DSP.
0	HWOB	0	R/W	R	Affects Multiplexed Host Address/Data dual halfword Mode Only. Write '0' when the Host accesses the most significant halfword first and the least significant second. Write '1' when the host accesses the least significant halfword first.

[1] Internal HRDY and UHPI_HRDY are opposite polarity.

[2] The logic level of the internal HRDY signal appears in this field. The intent of this bit field is for cases where H/W HRDY is not implemented as one of the UHPI interface signals. In theory, the Host can read this bit by performing software polling. The issue with this scenario is that it is impossible for the Host to check the status of the HPI while it is in a middle of a transaction to go off and read the HPIC register in order to determine the HPI module ready/not-ready state. Since access to HPIC in a middle of other transfer is not possible or allowed, the use of this bit field is not always useful. However, the use of this bit field could be useful when using FETCH bit to perform read accesses when having no H/W HRDY signal not implemented.

[3] Internal HINT bit and AMUTE2/UHPI_HINTare opposite polarity.

3.16 UHPI Write Address Register (HPIAW) (0x4300 0034)

This register is shared between the host and DSP. The address given is the address from the DSP perspective. The host selects this register by driving the UHPI.HCNTL[1:0] pins to "10" during the access. In addition, if the **HPIC.DUAL_HPIA** bit is set, then the host must set the **HPIC.HPIA_RW_SEL** bit to '0' in order to access this register. The DSP always has access to the **HPIAW** register even if **HPIC.DUAL_HPIA** is not set.

Figure 33. UHPI HPIAW Register Bit Layout (0x4300 0034)

31 ... 0
HPIAW[31:0]
R/W, 00000000000000000000000000000000

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 28. UHPI HPIAW Register Bit Field Description (0x4300 0034)

Bit No.	Name	Reset Value	Read Write Host	Read Write DSP	Description
31:0	HPIAW	0	R/W	R	HPI Address for Writes

3.17 UHPI Read Address Register (HPIAR) (0x4300 0038)

This register is shared between the host and DSP. The address given is the address from the DSP perspective. The host selects this register by driving the UHPI.HCNTL[1:0] pins to "10" during the access. In addition, if the **HPIC.DUAL_HPIA** bit is set, then the host must set the **HPIC.HPIA_RW_SEL** bit to '1' in order to access this register. The DSP always has access to the **HPIAW** register even if **HPIC.DUAL_HPIA** is not set.

Figure 34. UHPI HPIAR Register Bit Layout (0x4300 0038)

31	0
HPIAR[31:0]	
R/W, 00000000000000000000000000000000	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 29. UHPI HPIAR Register Bit Field Description (0x4300 0038)

Bit No.	Name	Reset Value	Read Write Host	Read Write DSP	Description
31:0	HPIAR	0	R/W	R	HPI Address for Reads

4 Revision History

Table 30 lists the changes made since the previous version of this document.

Table 30. Document Revision History

Reference	Additions/Modifications/Deletions
Section 1.6.2	Changed 1st sentence.
Section 1.7	Changed 1st sentence.
Figure 4	Changed figure.
Section 1.9.3	Changed 2nd sentence.
Section 1.9.3.3	Changed 1st sentence.
Section 1.9.5	Changed 4th paragraph.
Section 1.10.4	Deleted 3rd and 4th paragraphs.
Table 6	Changed last 2 rows.
Section 1.11.2	Changed 2nd sentence in 2nd paragraph.
Figure 11	Changed figure note c.
Table 7	Corrected table.
Table 8	Corrected table.
Table 9	Corrected table.
Table 10	Corrected table.
Table 11	Corrected table.
Table 12	Corrected table.
Table 18	Changed table description.
Table 27	Changed bit description and added table note #2.

IMPORTANT NOTICE

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DLP® Products	www.dlp.com	Broadband	www.ti.com/broadband
DSP	dsp.ti.com	Digital Control	www.ti.com/digitalcontrol
Clocks and Timers	www.ti.com/clocks	Medical	www.ti.com/medical
Interface	interface.ti.com	Military	www.ti.com/military
Logic	logic.ti.com	Optical Networking	www.ti.com/opticalnetwork
Power Mgmt	power.ti.com	Security	www.ti.com/security
Microcontrollers	microcontroller.ti.com	Telephony	www.ti.com/telephony
RFID	www.ti-rfid.com	Video & Imaging	www.ti.com/video
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265