TMS320VC5510 DSP
Host Port Interface (HPI)
Reference Guide

Literature Number: SPRU588A
April 2003
Revised February 2004

Texas Instruments

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
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
		Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

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

Preface

Read This First

About This Manual

This manual describes the features and operation of the host port interface (HPI) that is available on the TMS320VC5510 digital signal processors (DSPs) in the TMS320C55x™ (C55x™) DSP generation.

Notational Conventions

This document uses the following conventions:

- In most cases, hexadecimal numbers are shown with the suffix h. For example, the following number is a hexadecimal 40 (decimal 64):

  40h

  Similarly, binary numbers often are shown with the suffix b. For example, the following number is the decimal number 4 shown in binary form:

  0100b

- If a signal or pin is active low, it has an overbar. For example, the $\overline{\text{RESET}}$ signal is active low.

- Bits and signals are sometimes referenced with the following notations:

Notation	**Description**	**Example**
Register(n−m)	Bits n through m of Register	R(15−0) represents the 16 least significant bits of register R.
Bus[n:m]	Signals n through m of Bus	A[21:1] represents signals 21 through 1 of bus A.

- The following terms are used to name portions of data:

Term	**Description**	**Example**
LSB	Least significant bit	In R(15−0), bit 0 is the LSB of register R.
MSB	Most significant bit	In R(15−0), bit 15 is the MSB of register R.

Related Documentation From Texas Instruments

The following documents describe the C55x 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.

TMS320VC5510 Fixed-Point Digital Signal Processor Data Manual (literature number SPRS076) describes the features of the TMS320VC5510 fixed-point DSP and provides signal descriptions, pinouts, electrical specifications, and timings for the device.

TMS320C55x Technical Overview (literature number SPRU393). This overview is an introduction to the TMS320C55x DSPs, the latest generation of fixed-point DSPs in the TMS320C5000™ DSP platform. Like the previous generations, this processor is optimized for high performance and low-power operation. This book describes the CPU architecture, low-power enhancements, and embedded emulation features.

TMS320C55x DSP CPU Reference Guide (literature number SPRU371) describes the architecture, registers, and operation of the CPU for the TMS320C55x DSPs.

TMS320C55x DSP Peripherals Overview Reference Guide (literature number SPRU317) introduces the peripherals, interfaces, and related hardware that are available on TMS320C55x DSPs.

TMS320C55x DSP Algebraic Instruction Set Reference Guide (literature number SPRU375) describes the TMS320C55x DSP algebraic instructions individually. Also includes a summary of the instruction set, a list of the instruction opcodes, and a cross-reference to the mnemonic instruction set.

TMS320C55x DSP Mnemonic Instruction Set Reference Guide (literature number SPRU374) describes the TMS320C55x DSP mnemonic instructions individually. Also includes a summary of the instruction set, a list of the instruction opcodes, and a cross-reference to the algebraic instruction set.

TMS320C55x Optimizing C/C++ Compiler User’s Guide (literature number SPRU281) describes the TMS320C55x™ C/C++ Compiler. This C/C++ compiler accepts ISO standard C and C++ source code and produces assembly language source code for TMS320C55x devices.

TMS320C55x Assembly Language Tools User’s Guide (literature number SPRU280) describes the assembly language tools (assembler, linker, and other tools used to develop assembly language code), assembler directives, macros, common object file format, and symbolic debugging directives for TMS320C55x devices.

TMS320C55x DSP Programmer’s Guide (literature number SPRU376) describes ways to optimize C and assembly code for the TMS320C55x DSPs and explains how to write code that uses special features and instructions of the DSPs.

Trademarks

TMS320, TMS320C5000, TMS320C55x, and C55x are trademarks of Texas Instruments.

Trademarks are the property of their respective owners.

Contents

Figures

Tables

Host Port Interface (HPI)

This chapter describes the features and operation of the host port interface (HPI) on TMS320VC5510 devices in the TMS320C55x™ (C55x™) DSP generation. An HPI enables an external host processor (host) to directly access a portion of the memory in the memory map of the C55x DSP.

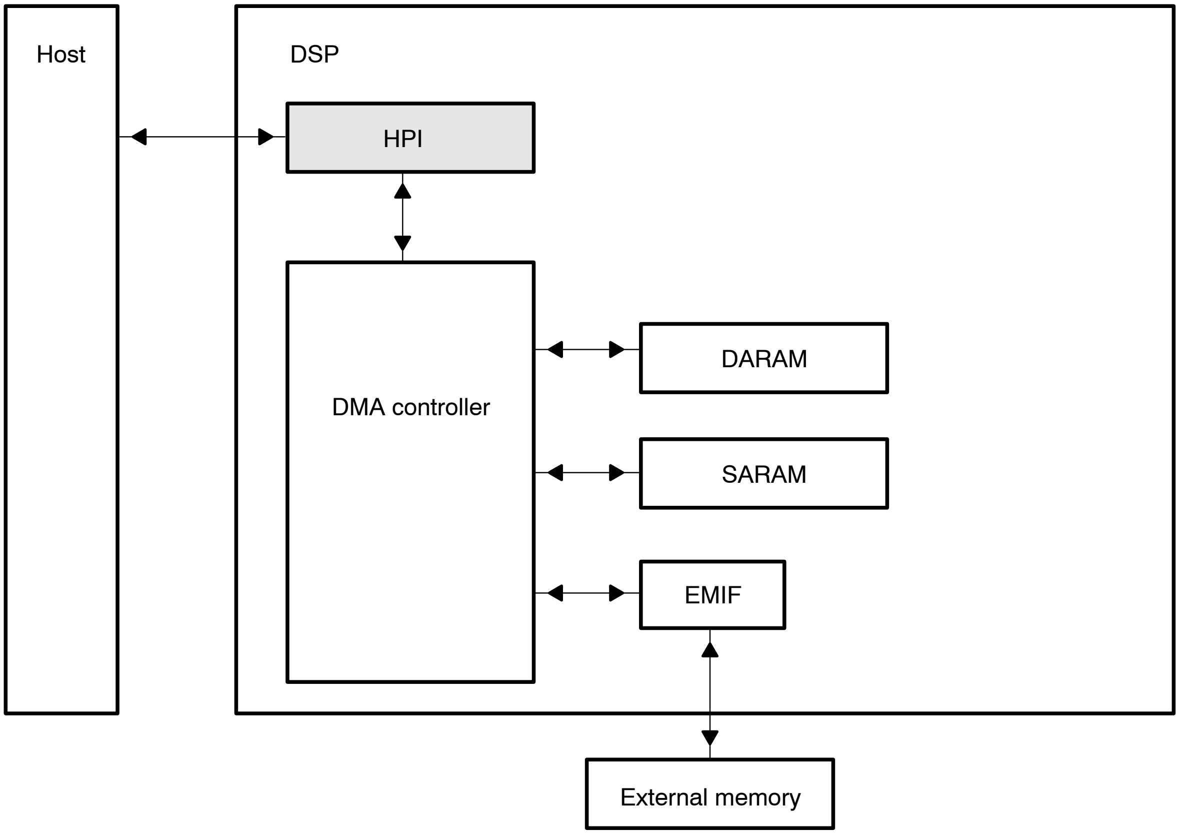
Note:

This document supports 5510 silicon revision 2.1 or later. For 5510 silicon earlier than revision 2.1, see the application report *Using the TMS320VC5509/5510 Enhanced HPI* (SPRA741).

1 Introduction to the HPI

The host port interface (HPI) provides a 16-bit-wide parallel port through which an external host processor (host) can directly access the memory of the DSP. The host and the DSP can exchange information via memory internal or external to the DSP and within the address reach of the HPI. The HPI uses 20-bit addresses, where each address is assigned to a 16-bit word in memory. Figure 1 is a conceptual diagram of the connections between the HPI and other components of a host-DSP system.

Figure 1. The Position of the HPI in a Host-DSP System



The DMA controller handles all HPI accesses. Through the DMA controller, one of two HPI access configurations can be chosen. In one configuration, the HPI shares internal memory with the DMA channels. In the other configuration, the HPI has exclusive access to the internal memory.

The HPI cannot directly access other peripherals' registers. If the host requires data from other peripherals, that data must be moved to memory first, either by the CPU or by activity in one of the six DMA channels. Likewise, data from the host must be transferred to memory before being transferred to other peripherals.

To provide flexibility in the choice of a host, the HPI allows two modes for passing data and addresses. The nonmultiplexed mode (see section 5, page 19) provides the host processor with separate address and data buses. The multiplexed mode (see section 6, page 22) provides a single bus to transport address and data information. The different modes require some different connections to HPI signals. There are three HPI registers for data, addresses, and control information (see section 10, page 30).

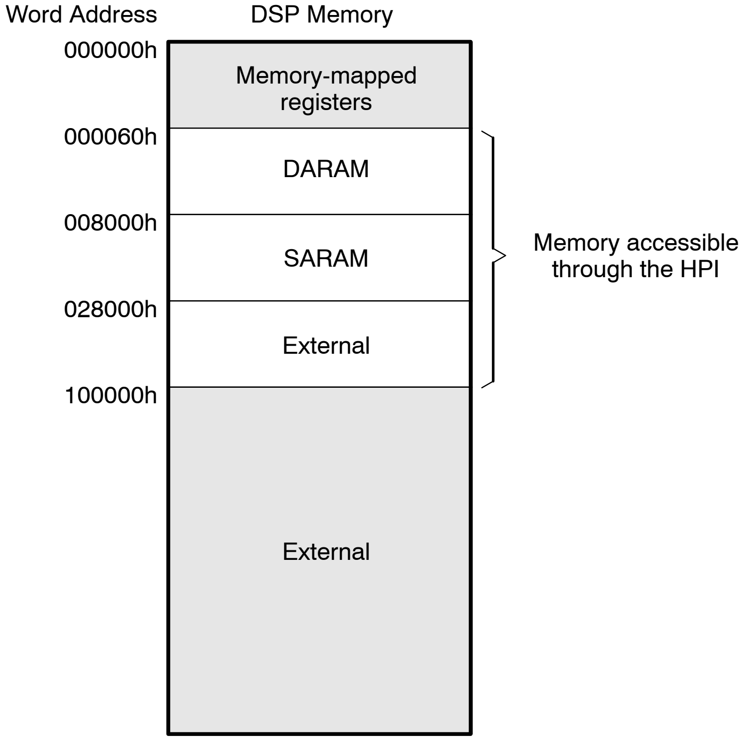
2 DSP Memory Accessible Through the HPI

Addresses driven by a host on the external address lines of the HPI are treated as word addresses, not byte addresses. Each HPI address corresponds to a 16-bit word in data space.

Figure 2 highlights the portion of the DSP data memory map that is accessible to a host through the HPI. The shaded areas in the memory map are not accessible through the HPI.

The 20 address lines of the HPI enable the host to access internal and external memory at addresses 000060h-0FFFFFh. Addresses 000000h-00005Fh are reserved for the memory-mapped registers (MMRs) of the CPU and are not accessible through the HPI.

Figure 2. DSP Memory Accessible Through the HPI



Note: The shaded areas of the memory map are not accessible through the HPI.

3 HPI-DMA Interaction

The HPI uses the DMA controller to move data into and out of DSP memory. The DMA controller services the HPI (via a dedicated port) and six programmable DMA channels. Activity in the channels is controlled by such factors as their priority, the DMA port resources they use, and whether they are triggered by synchronization events. The HPI and the six channels are serviced by the DMA controller in a round-robin manner. Because of this structure, the activity in the enabled channels will affect the latency of HPI transactions and vice versa.

Two programmable options can affect the latencies of HPI-DMA interaction:

- The EHPIPRIO bit in the DMA global control register (DMAGCR) controls the priority of the HPI requests in the DMA service chain. When EHPIPRIO = 0, HPI requests are considered low priority and are serviced after all high priority channels. When EHPIPRIO = 1, HPI requests are considered high priority and are serviced before low priority channels. If the HPI and any channels are at the same priority level, they are serviced in a round-robin manner.

- The EHPIEXCL bit in DMAGCR gives the HPI exclusive access to the internal memory inside the DSP (SARAM and DARAM). When EHPIEXCL = 0, the HPI can access any internal or external memory within its address reach, and the DMA channels can use any DMA port. When EHPIEXCL = 1, the HPI can access only the internal memory, and the DMA channels can access only the EMIF and Peripheral ports. Any channels configured to use internal memory will be suspended until the HPI-exclusive condition is released. This capability provides the host a minimum latency operation condition at the expense of suspending the channels.

For detailed information on the service chain and the DMA global control register, see the *TMS320VC5509/5510 DSP Direct Memory Access (DMA) Controller Reference Guide* (SPRU587).

4 HPI Signals

Table 1 provides a summary of the signals. In the Type column, Z refers to the high impedance state. There are some differences in the signal connections between the two modes of the HPI: Nonmultiplexed mode (see section 5, page 19) and multiplexed mode (see section 6, page 22). For timing information on the HPI signals, see the *TMS320VC5510 Fixed-Point Digital Signal Processor Data Manual* (SPRS076).

Table 1. Signals of the HPI

Signal(s)	Type	Description
HD[15:0]	Input/Output/Z	HPI data bus. HD is a parallel, bidirectional, 3-state bus. In the nonmultiplexed mode: These 16 signals are used to carry data only. In the multiplexed mode: These 16 signals are used to carry both addresses and data. Between data transfers: The HPI does not drive HD. If the bus holders are enabled, HD retains the last driven state. If the bus holders are disabled, HD enters the high-impedance state. For information about the bus holders, see section 4.3 (page 18).
HA[19:0]	Input	HPI address bus. In the nonmultiplexed mode: HA is a parallel, unidirectional address bus that carries 20-bit addresses from the host processor to the HPI. The 20 lines of this bus allow the addressing of 1024K words of the DSP memory. In the multiplexed mode: Pin HA[1] is used for the signal HCNTL1. Pin HA[2] is available for the signal $\overline{\text{HAS}}$. The other pins are not used.
$\overline{\text{HCS}}$	Input	HPI chip-select signal. $\overline{\text{HCS}}$ serves as the enable input of the HPI, and must be low during an access. For the relationships among the signals $\overline{\text{HDS1}}$, $\overline{\text{HDS2}}$, $\overline{\text{HCS}}$, and HRDY, see section 4.1 (page 17).
HR/$\overline{\text{W}}$	Input	HPI read/write signal. This input indicates the direction of the host access. When high, HR/$\overline{\text{W}}$ indicates a read from the DSP memory. When low, HR/$\overline{\text{W}}$ indicates a write to the DSP memory.

Table 1. Signals of the HPI (Continued)

<table>
<tr><th>Signal(s)</th><th>Type</th><th>Description</th></tr>
<tr><td>$\overline{\mathrm{HDS1}}$,
$\overline{\mathrm{HDS2}}$</td><td>Input</td><td>HPI data strobe signals. The exclusive-NOR of $\overline{\mathrm{HDS1}}$ and $\overline{\mathrm{HDS2}}$ forms a strobe signal for controlling data transfers during host-access cycles. For the relationships among the signals $\overline{\mathrm{HDS1}}$, $\overline{\mathrm{HDS2}}$, $\overline{\mathrm{HCS}}$, and HRDY, see section 4.1 (page 17). Connections to $\overline{\mathrm{HDS1}}$ and $\overline{\mathrm{HDS2}}$ depend on the host's strobe signal(s):
<table>
<tr><th>Available Host Data Strobe Pins</th><th>Connections to HPI Data Strobe Pins</th></tr>
<tr><td>Host has separate read and write strobe pins, both active-low</td><td>Connect one strobe to $\overline{\mathrm{HDS1}}$ and the other to $\overline{\mathrm{HDS2}}$.</td></tr>
<tr><td>Host has one active-low strobe pin</td><td>Connect the strobe to $\overline{\mathrm{HDS1}}$ or $\overline{\mathrm{HDS2}}$, and connect the other $\overline{\mathrm{HDS}}$ pin to logic level 1.</td></tr>
<tr><td>Host has one active-high strobe pin</td><td>Connect the strobe to $\overline{\mathrm{HDS1}}$ or $\overline{\mathrm{HDS2}}$ and connect the other $\overline{\mathrm{HDS}}$ pin to logic level 0.</td></tr>
</table>
</td></tr>
<tr><td>HRDY</td><td>Output</td><td>HPI-ready signal. This signal tells the host whether the HPI is ready for an access. When low, HRDY indicates that the HPI is busy and the host should extend the current transfer cycle. When high, HRDY indicates that the HPI has completed the data transfer and is ready for the host to continue.
When $\overline{\mathrm{HCS}}$ goes high (inactive), HRDY is always driven high, regardless of the internal status of the HPI. For the relationships among the signals $\overline{\mathrm{HDS1}}$, $\overline{\mathrm{HDS2}}$, $\overline{\mathrm{HCS}}$, and HRDY, see section 4.1 (page 17).</td></tr>
</table>

Table 1. Signals of the HPI (Continued)

Signal(s)	Type	Description
HCNTL0, HCNTL1	Input	HPI access control signals. In the nonmultiplexed mode: HCNTL0 determines whether the HPI accesses the control register (HPIC) or the data register (HPID), as shown in the following table (0 = low, 1 = high). HCNTL1 is not used. **HCNTL0** **Access Type (Nonmultiplexed Mode)** 0 HPIC read/write 1 HPID read/write In the multiplexed mode: HCNTL1 and HCNTL0 together select the type of register access, as shown in the following table (0 = low, 1 = high). HCNTL1 is multiplexed with line 1 of the address bus, HA[1], but this creates no conflict because HA is not used in the multiplexed mode. **HCNTL[1:0]** **Access Type (Multiplexed Mode)** 00 HPIC read/write 01 HPID read/write with address auto-increment by 1 10 HPIA read/write 11 HPID read/write without address auto-increment
$\overline{\text{HAS}}$	Input	Address strobe signal. This signal is used only in the multiplexed mode. This address strobe signal allows HCNTL[1:0] and HR/$\overline{\text{W}}$ to be removed earlier in an access cycle, which allows more time to switch bus states from address to data information. $\overline{\text{HAS}}$ facilitates interfacing to multiplexed address and data type buses. Typically, an address latch enable (ALE) signal of a host is connected to $\overline{\text{HAS}}$. $\overline{\text{HAS}}$ must be kept high if it is not used. $\overline{\text{HAS}}$ is multiplexed with line 2 of the address bus, HA[2], but this creates no conflict because HA is not used in the multiplexed mode.
HMODE	Input	HPI mode signal. When held high, HMODE selects the nonmultiplexed mode (separate address and data buses). When held low, HMODE selects the multiplexed mode (data bus carries addresses and data).
$\overline{\text{HINT}}$	Output	DSP-to-host interrupt signal. $\overline{\text{HINT}}$ enables the DSP to send an interrupt pulse to the host processor. The signal level is controlled by the HINT bit in status register ST3_55 of the C55x CPU (HINT = 0 means $\overline{\text{HINT}}$ low; HINT = 1 means $\overline{\text{HINT}}$ high).

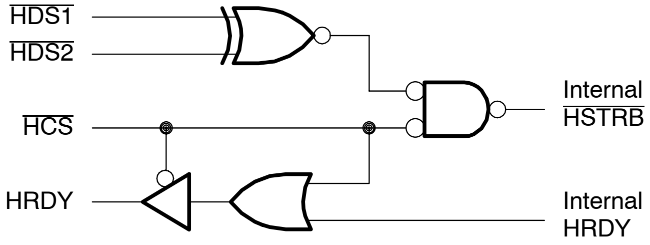
4.1 $\overline{\text{HDS2}}$, $\overline{\text{HDS1}}$, and $\overline{\text{HCS}}$: Data Strobing and Chip Selection

As illustrated in Figure 3, the strobing logic is a function of three key inputs: the chip select pin ($\overline{\text{HCS}}$) and two data strobe signals ($\overline{\text{HDS1}}$ and $\overline{\text{HDS2}}$). The internal strobe signal, $\overline{\text{HSTRB}}$, functions as the actual strobe signal inside the HPI. $\overline{\text{HCS}}$ must be low (HPI selected) during strobe activity on the $\overline{\text{HDS}}$ pins. If $\overline{\text{HCS}}$ remains high (HPI not selected), activity on the $\overline{\text{HDS}}$ pins is ignored.

Strobe connections between the host and the HPI depend in part on the number and types of strobe pins available on the host. Table 1 (preceding this section) describes some options for connecting to the $\overline{\text{HDS}}$ pins.

The $\overline{\text{HCS}}$ input and one $\overline{\text{HDS}}$ strobe input can be tied together and driven with a single strobe signal from the host. This technique selects the HPI and provides the strobe simultaneously. However, since HRDY is also gated by $\overline{\text{HCS}}$ (see Figure 3) using $\overline{\text{HCS}}$ as a strobe will limit the ability to use HRDY. If $\overline{\text{HCS}}$ goes high (HPI not selected), HRDY is placed in an high-impedance state, regardless of whether the current transfer is completed.

Figure 3. HPI Strobe and Select Logic



4.2 Unused HPI Control Signals and Internal Pull-Ups

Unused HPI inputs signals should be controlled either externally or by using the internal software-controlled pull-ups provided on the TMS320VC5510. The internal pull-ups are controlled by the HPE bit in the system register (SYSR). When HPE is set, internal pull-ups are enabled on HA[19:0], $\overline{\text{HCS}}$, $\overline{\text{HAS}}$, $\overline{\text{HDS1}}$, $\overline{\text{HDS2}}$, HR/$\overline{\text{W}}$, HMODE, HCNTL0 and HCNTL1. When HPE is cleared, all of these pull-ups are disabled. For more information about SYSR, see the *TMS320VC5510 Fixed-Point Digital Signal Processor Data Manual* (SPRS076).

4.3 HPI Data Bus Holders

The HPI data bus has software-controlled bus holders that can be used to hold the bus state when the bus is not being driven. Controlling the bus eliminates additional power consumption due to a floating bus. The HPI data bus holders are controlled by the HBH bit in the system register (SYSR). When HBH = 1, the bus holders are enabled on the HPI data bus. The bus holders are weak drivers that will maintain the last driven state of the bus. The bus holders do not prevent the bus being driven by the host. When HBH = 0, the bus holders are disabled. In multiprocessing structures where multiple DSPs are present with the HPI data buses connected in parallel, it may be desirable to disable the bus holders on some or all of the DSPs.

For more information on SYSR register functions see the *TMS320VC5510 Fixed-Point Digital Signal Processor Data Manual* (SPRS076).

5 Nonmultiplexed Mode

To select the nonmultiplexed mode, hold the HMODE signal high. In this mode:

- ❑ The HPI uses **separate buses for addresses and data**.
- ❑ The HPI receives 20-bit addresses from the host via the address bus (HA). For each data transfer, an address must be driven on HA. The HPI address register (HPIA) is not used.
- ❑ The HPI data register (HPID) acts as a temporary holding place for data to be transferred through the HPI. If the current access is a read, HPID contains the data that was read from the DSP memory. If the current access is a write, HPID contains the data that will be written to the DSP memory. The DSP CPU cannot access HPID.
- ❑ HPIC provides the host with important options for controlling data transfers. By writing to HPIC, interrupt requests can be sent to the DSP and the HPI bootload process can be controlled. For detailed information on boot loading through the HPI, see the application report *Using the TMS320VC5510 Bootloader* (SPRA763). The DSP CPU cannot access HPIC. More details about HPIC are in section 10 (page 30).

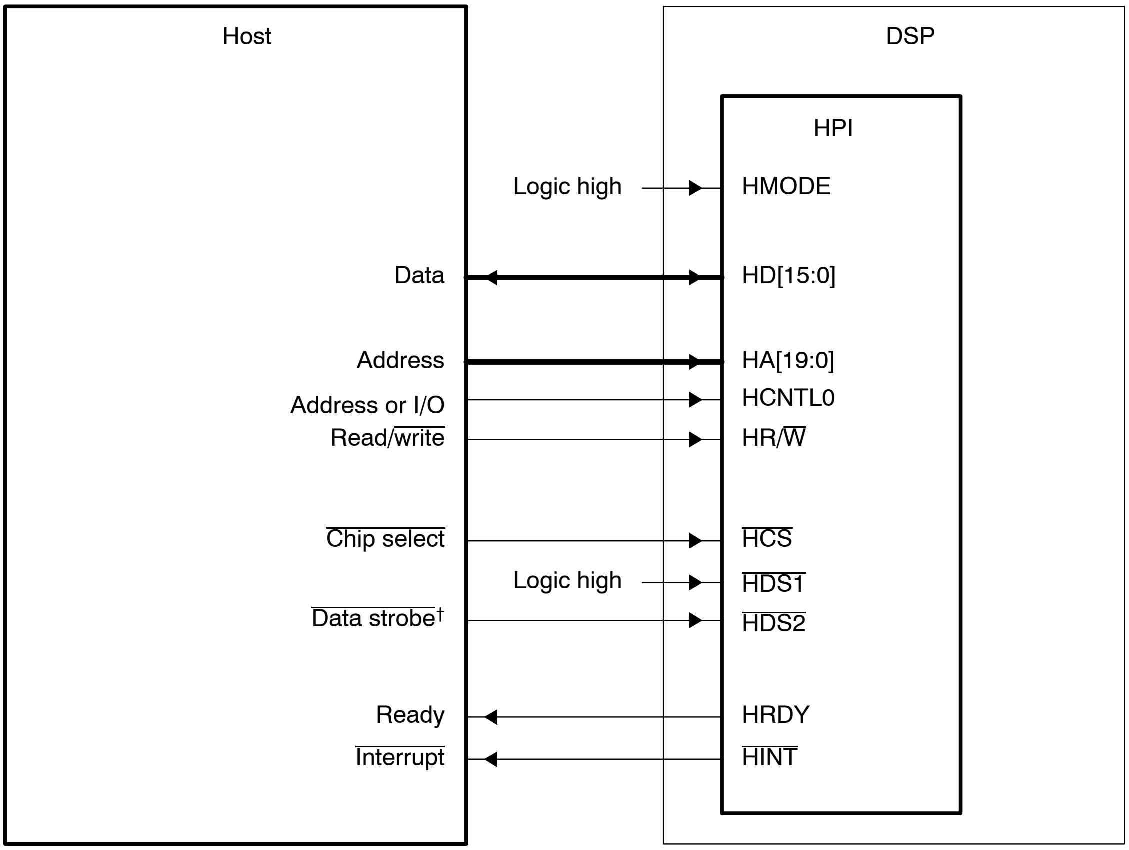
Section 5.1 provides an example of signal connections. Section 5.2 explains how the host must differentiate various accesses to HPID and HPIC. For timing diagrams of host-HPI activity, see the device-specific data manual.

5.1 Signal Connections in the Nonmultiplexed Mode

Figure 4 shows an example of signal connections for the nonmultiplexed mode. Details about all the HPI signals are in section 4 (page 14). Here are key points that are specific to the nonmultiplexed mode:

- ❑ The HMODE pin must be held high to select the nonmultiplexed mode.
- ❑ Data and addresses travel on separate buses (HD and HA, respectively).
- ❑ The host indicates the cycle type with the HCNTL0 and HR/$\overline{\text{W}}$ signals as described in section 5.2.

Figure 4. Example of Host-DSP Signal Connections in the Nonmultiplexed Mode



† Data strobing options are given in Table 1, page 14.

5.2 Indicating the Cycle Type in the Nonmultiplexed Mode

The host uses the HCNTL0 and HR/$\overline{\text{W}}$ pins of the HPI to indicate the cycle type. The cycle type consists of:

- ❑ The access type that the host selects by driving the appropriate level on the HCNTL0 pin. Table 2 describes the available access types for the nonmultiplexed mode. In this mode, HPIA is not used and, therefore, only HPIC accesses and HPID accesses without auto-incrementing are valid. Any HPI access with HCNTL0 driven low is an HPIC access; the value on the address bus is ignored.
- ❑ The transfer direction that the host selects with the HR/$\overline{\text{W}}$ pin. The host must drive the HR/$\overline{\text{W}}$ signal high (read) or low (write).

A summary of cycle types is in Table 3. The HPI samples the HCNTL0 and HR/$\overline{\text{W}}$ levels at the falling edge of the internal strobe signal, $\overline{\text{HSTRB}}$.

Table 2. Access Types Selectable With the HCNTL0 Signal in the Nonmultiplexed Mode

HCNTL0	Access Type
0	HPIC access The host requests to access the control register (HPIC).
1	HPID access The host requests to access the data register (HPID).

Table 3. Cycle Types Selectable With the HCNTL0 and HR/$\overline{W}$ Signals in the Nonmultiplexed Mode

HCNTL0	HR/$\overline{W}$	Cycle Type
0	0	HPIC write cycle
0	1	HPIC read cycle
1	0	HPID write cycle
1	1	HPID read cycle

6 Multiplexed Mode

To select the multiplexed mode, hold the HMODE signal low. In this mode:

- **Addresses and data are carried on the same bus** (the HPI data bus, HD[15:0]). Thus, an address register (HPIA) is needed to store an address while the bus is carrying data. HPIA is 20 bits wide in order to support up to 1024K words of accessible memory. The multiplexing of addresses and data means that the host must load HPIA *before* performing reads and writes to the DSP memory. As HPIA is a 20 bit register, it must be loaded with two HPI accesses (see section 6.3, page 26)

- When reading from or writing to the DSP memory, the HPI uses its data register (HPID) as a temporary holding place for the data. HPID contains the data that was read from the DSP memory (for a host read operation) or the data that will be written to the DSP memory (for a host write operation). The DSP CPU cannot access HPID.

- HPIC provides the host with important options for controlling data transfers. Writing to HPIC selects which portion of HPIA is being loaded (lower portion or upper portion), send interrupt requests to the DSP, and control the HPI bootload process. The DSP CPU cannot access HPIC. See section 10 (page 30) for more details about HPIC.

Section 6.1 provides examples of signal connections. Section 6.2 explains how the host must differentiate various accesses to HPIA, HPID, and HPIC. Section 6.3 gives a procedure for the host to follow when loading an address to HPIA, and section 6.4 describes how the address can be automatically incremented between data transfers. For timing diagrams of host-HPI activity, see the device-specific data manual.

6.1 Signal Connections in the Multiplexed Mode

Figure 5 and Figure 6 show examples of signal connections for the multiplexed mode. In Figure 5, the address strobe signal ($\overline{\text{HAS}}$) signal is used. In Figure 6, $\overline{\text{HAS}}$ is tied high (not used).

Details about all the HPI signals are in section 4 (page 14). The following are key points about signals used in the multiplexed mode:

- The HMODE pin must be held low to select the multiplexed mode.

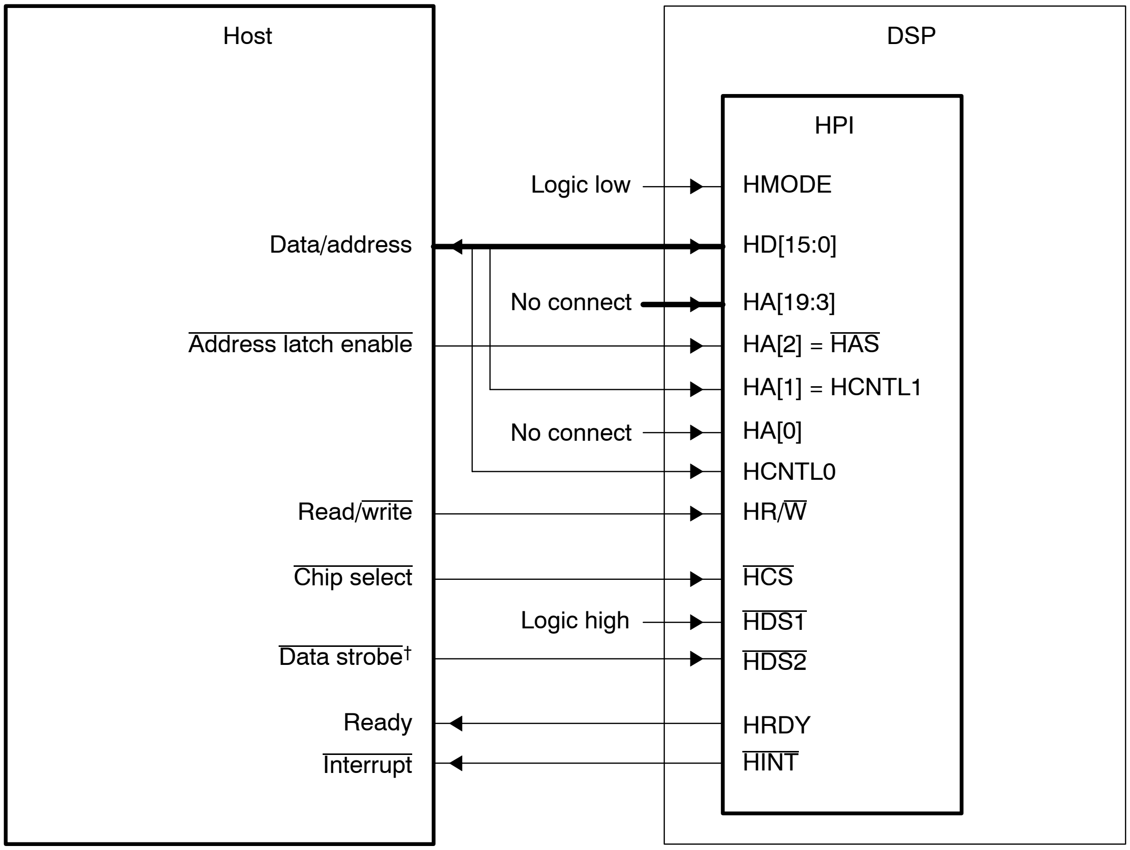
- Addresses must share the HD lines with data.

- The host indicates the cycle type with the HCNTL[1:0] and HR/$\overline{\text{W}}$ signals as described in section 6.2.

- $\overline{\mathrm{HAS}}$ allows HCNTL[1:0] and HR/$\overline{\mathrm{W}}$ to be removed earlier in an access cycle, which allows more time to switch bus states from address to data information. $\overline{\mathrm{HAS}}$ is an optional signal available for hosts that carry both data and addresses on a single bus. The HPI can be used without $\overline{\mathrm{HAS}}$. If $\overline{\mathrm{HAS}}$ is not used, it should be driven high externally, or pulled up using the internal pull-ups controlled by the HPE bit in the system register (SYSR). For more information about SYSR, see the *TMS320VC5510 Fixed-Point Digital Signal Processor Data Manual* (SPRS076).

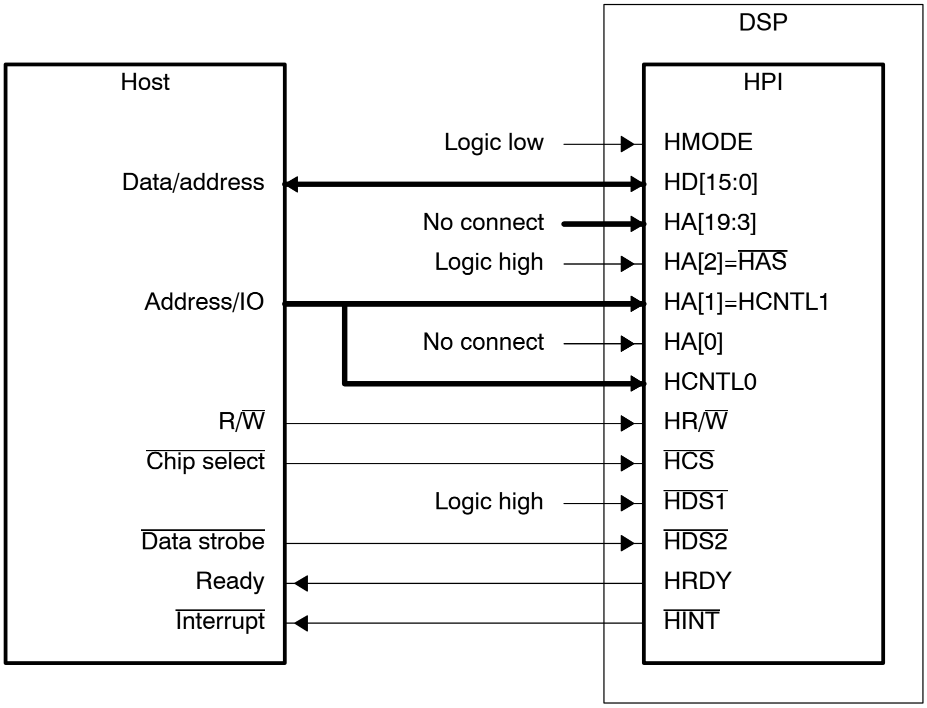
- HCNTL1 and $\overline{\mathrm{HAS}}$ each use one of the lines of the address bus (HA), which is not otherwise used in the multiplexed mode. The other lines of HA can be pulled up internally, as mentioned in section 4.2 (page 17).

Figure 5. Example of Host-DSP Signal Connections When Using the $\overline{\mathrm{HAS}}$ Signal in the Multiplexed Mode


Host
DSP
HPI
Logic low
HMODE
Data/address
HD[15:0]
No connect
HA[19:3]
$\overline{\mathrm{Address\ latch\ enable}}$
HA[2] = $\overline{\mathrm{HAS}}$
HA[1] = HCNTL1
No connect
HA[0]
HCNTL0
Read/$\overline{\mathrm{write}}$
HR/$\overline{\mathrm{W}}$
$\overline{\mathrm{Chip\ select}}$
$\overline{\mathrm{HCS}}$
Logic high
$\overline{\mathrm{HDS1}}$
$\overline{\mathrm{Data\ strobe}}$†
$\overline{\mathrm{HDS2}}$
Ready
HRDY
$\overline{\mathrm{Interrupt}}$
$\overline{\mathrm{HINT}}$

† Data strobing options are given in Table 1, page 14.

Figure 6. Example of Host-DSP Signal Connections When the $\overline{\text{HAS}}$ Signal is Tied High in the Multiplexed Mode



† Data strobing options are given in Table 1, page 14.

6.2 Indicating the Cycle Type in the Multiplexed Mode

The host uses the HCNTL[1:0] and HR/$\overline{\text{W}}$ pins of the HPI to indicate the cycle type. The cycle type consists of:

- ❑ The access type that the host selects by driving the appropriate levels on the HCNTL[1:0] pins. Table 4 describes the available access types for the multiplexed mode.
- ❑ The transfer direction that the host selects with the HR/$\overline{\text{W}}$ pin. The host must drive the HR/$\overline{\text{W}}$ signal high (read) or low (write).

A summary of cycle types is in Table 5. The HPI samples the HCNTL levels either at the falling edge of $\overline{\text{HAS}}$ (if $\overline{\text{HAS}}$ is used in the multiplexed mode) or at the falling edge of the internal strobe signal, $\overline{\text{HSTRB}}$ (if $\overline{\text{HAS}}$ is not used and is tied high).

Table 4. Access Types Selectable With the HCNTL[1:0] Signals in the Multiplexed Mode

HCNTL1	HCNTL0	Access Type
0	0	HPIC access The host requests to access the control register (HPIC).
0	1	HPID access with auto-incrementing The host requests to access the data register (HPID) and to have the memory address automatically incremented by 1 after the access.
1	0	HPIA access The host requests to access the address register (HPIA).
1	1	HPID access without auto-incrementing The host requests to access the data register (HPID) but requests no automatic post-increment of the memory address.

Table 5. Cycle Types Selectable With the HCNTL[1:0] and HR/$\overline{\text{W}}$ Signals in the Multiplexed Mode

HCNTL1	HCNTL0	HR/$\overline{\text{W}}$	Cycle Type
0	0	0	HPIC write cycle
0	0	1	HPIC read cycle
0	1	0	HPID write cycle with auto-incrementing
0	1	1	HPID read cycle with auto-incrementing
1	0	0	HPIA write cycle
1	0	1	HPIA read cycle
1	1	0	HPID write cycle without auto-incrementing
1	1	1	HPID read cycle without auto-incrementing

6.3 Loading HPIA With an Address

Because the data bus has only 16 bits, the 20-bit address must be loaded into HPIA with two HPIA write cycles. Before each HPIA write cycle, the host must indicate whether HPIA(19-16) or HPIA(15-0) is to be accessed. The host does this by toggling the extended address enable (XADD) bit in HPIC, as shown in the following procedure. For details about HPIC, see section 10 (page 30).

1) Host performs HPIC write access to set XADD=1.
   a) Drive HCNTL1, HCNTL0, and HR/W low and assert HDSx or HAS to indicate HPIC write.
   b) Drive data such that XADD (bit 5) is high and deassert HDSx to latch the data.
   c) Now any HPIA writes will go to HPIA[19:16].
2) Host performs HPIA write access to initialize HPIA[19:16].
   a) Drive HCNTL1 high, and HCNTL0 and HR/W low, and assert HDSx or HAS to indicate HPIA write.
   b) Drive the four most significant bits (MSBs) of the memory address onto HD[3:0] and deassert HDSx to latch data (unused data bits HD[15:4] should be driven to 0).
   c) Because of the XADD setting, HD[3:0] gets written to the four MSBs of HPIA.
3) Host performs HPIC write access to clear XADD=0.
   a) Same as steps 1a and 1b, only with XADD (bit 5) low.
   b) Now any HPIA write will go to HPIA[15:0].
4) Host performs HPIA write access to initialize HPIA[15:0].
   a) Same as steps 2a and 2b, only the HD[15:0] lines should be driven with the 16 LSBs of the memory address.
   b) Now HPIA contains the full 20-bit address.

Notes:

1) If a series of accesses within a 64K-word address range is being made, HPIA(19-16) must be loaded only one time, before the first access. For each subsequent access within that page, a change to HPIA(15-0) is sufficient.
2) After the initial address is set, XADD should be left cleared (0) for proper operation during HPID accesses with auto-incrementing.

6.4 Auto-Increment Option: Automatic Address Increment Between Transfers

If the host is reading and/or writing at random addresses, it must write to HPIA before each data transfer. However, if the host performs accesses at sequential addresses and the HPI is in its multiplexed mode, it can reduce the required number of cycles by using the address auto-increment option: Drive HCNTL1 low and HCNTL0 high for auto-incremented data accesses.

When auto-incrementing is used, the host needs to write only the start address to HPIA; for each subsequent HPID access, the address in HPIA is automatically incremented by 1. Although the internal address is automatically incremented, a host read of HPIA will always return the last address written to HPIA by the host.

Notes:

When crossing a 64K-word boundary with auto-incremented data accesses, the upper portion of the HPIA address (HPIA[19:16]) does not increment. In this case, the host must reinitialize the entire HPIA register using the procedure detailed in section 6.3, *Loading HPIA with an Address*, page 26.

7 Interrupts Between the Host and the DSP

By modifying special interrupt bits, the host and the DSP can send interrupt requests to each other.

7.1 Sending an Interrupt Request From the Host to the DSP

To have the host send an interrupt request to the DSP, perform the following:

1) Make sure the HPI is configured to write to the HPI control register (HPIC).

   In the nonmultiplexed mode of the HPI, the HCNTL0 signal must be held low to select HPIC. In the multiplexed mode of the HPI, HCNTL1 and HCNTL0 must both be held low to select HPIC.

2) Write a 1 to bit 1 (DSPINT) of HPIC.

   Setting this bit causes the DSP to set the DSPINT flag bit in the CPU. If this maskable interrupt is properly enabled in the CPU, the CPU will fetch the DSPINT interrupt vector and branch to the corresponding interrupt service routine.

The host does not have to clear the DSPINT bit of HPIC. Only one interrupt is generated each time the host writes 1 to DSPINT.

Whenever DSPINT is read, 0 is returned.

7.2 Sending an Interrupt Request From the DSP to the Host

The DSP can send an interrupt request to the host by clearing and then setting the HINT bit in status register ST3_55 of the CPU. A change to the HINT bit changes the level of the $\overline{\text{HINT}}$ output signal of the HPI. If the DSP writes a 0 to the HINT bit, $\overline{\text{HINT}}$ goes low (active). If the DSP writes a 1 to the HINT bit, $\overline{\text{HINT}}$ goes high (inactive). Thus, the interrupt pulse width is managed by software.

Although the $\overline{\text{HINT}}$ signal can be used to interrupt the host, there is no direct acknowledgement path from the host back to the DSP. If desired, the host can acknowledge the interrupt by writing to a memory location that is common to the host and the DSP.

During a DSP reset, the CPU sets the HINT bit and $\overline{\text{HINT}}$ goes high (inactive).

8 Boot Loading With the HPI

The HPI can be used to load application code to the internal memory of the DSP. After reset, the host can load the desired application code into the memory space of the DSP through the HPI. After the code has been loaded, the host can cause the DSP to begin execution of the loaded code. For details, see the application report *Using the TMS320VC5510 Bootloader* (SPRA763).

9 Power, Emulation, and Reset Considerations

9.1 HPI Affected by the States of Certain Idle Domains

The DSP is divided into idle domains that can be programmed to be idle or active. To reduce power consumption, certain idle domains can be turned off. The HPI does not belong to any of the idle domains, but it is affected by the clock generator idle domain and the DMA idle domain. If either of these domains are idle, the host cannot access the DSP memory.

9.2 HPI Emulation Modes

The HPI relies on the operation of the DMA controller to move data to/from the memory of the DSP. As a result, when emulation activity affects the DMA controller, it also affects the ability of the HPI to access memory. The FREE bit of the DMA controller determines how the DMA controller reacts to an emulation breakpoint or other emulation halt:

- If FREE = 0 (the reset value), a breakpoint or other emulation halt suspends DMA transfers.
- If FREE = 1, DMA transfers are not interrupted by a breakpoint or other emulation halt.

The FREE bit is bit 2 of the DMA global control register. This register is described in the *TMS320VC5509/5510 DSP Direct Memory Access (DMA) Controller Reference Guide* (SPRU587).

9.3 Effects of a DSP Reset on the HPI

If the DSP reset signal is driven low, the DSP undergoes a reset. The control register (HPIC) is forced to its default value (see Figure 9 in section 10). The address register (HPIA) and the data register (HPID) are not initialized by a DSP reset.

10 HPI Registers

The host port interface (HPI) contains three registers (see the following sections) that a host can use to access the memory of the DSP. The registers share a data bus; therefore, the host must drive the signals HCNTL1 and/or HCNTL0 to the appropriate levels to indicate which HPI register host is to access. The DSP can neither read from nor write to these registers.

10.1 Data Register (HPID)

As shown in Figure 7, HPID is a 16-bit register. This register acts as a temporary holding place for data to be transferred through the HPI. HPID contains the data that was read from the DSP memory if the current access is a read, or the data that will be written to the DSP memory if the current access is a write.

Figure 7. Data Register (HPID)

15	0
Data	
HR/W-x	

Legend: HR/W = Host read/write access; -x = Value not initialized after DSP reset
(The DSP cannot access HPID)

10.2 Address Register (HPIA)

As shown in Figure 8, HPIA is a 20-bit register. In the multiplexed mode of the HPI (see section 6, page 22), this register acts as a temporary holding place for a 20-bit address for a read or write operation. In the nonmultiplexed mode of the HPI (see section 5, page 19), HPIA is not needed because the address is directly available on input signals HA[19:0].

Notes:

As HPIA is a 20 bit register, it requires two 16-bit host cycles to access (see section 6.3, page 26.) Also, the upper four bits do not increment during auto-increment cycles (see section 6.4, page 27.)

Figure 8. Address Register (HPIA)

19	0
Address	
HR/W-x	

Legend: HR/W = Host read/write access; -x = Value not initialized after DSP reset
(The DSP cannot access HPIA)

10.3 Control Register (HPIC)

HPIC provides the host with important options for controlling data transfers. The fields of HPIC are shown in Figure 9 and described in Table 6.

Figure 9. Control Register (HPIC)

15–8
Reserved†
HR/W-0

7–6	5	4–2	1	0
Reserved†	XADD	Reserved†	DSPINT	RESET
HR/W-0	HR/W-0	HR/W-0	HW-0	HW-0

Legend: HR/W = Host read/write access; HW = Host write-only access; -*n* = Value after DSP reset
(The DSP cannot access HPIC)

† Always write 0s to the reserved bits. The read states of the reserved bits are not defined.

Table 6. Control Register (HPIC) Field Descriptions

Bit	Field	Value	Description
15-6	Reserved	0	Always write 0s to these bits. The read states of these bits are not defined.
5	XADD		Extended address enable bit. In the multiplexed mode of the HPI, the HPI address register (HPIA) is loaded via the 16-bit data bus, HD[15:0]. When a 20-bit address is used, HPIA must be loaded with two transfers across HD[15:0]. To load bits 15-0 of HPIA, clear XADD beforehand. To load bits 19-16, set XADD beforehand. In the nonmultiplexed mode of the HPI, XADD is ignored because HPIA is not used; instead, the address comes directly from the 20-bit address bus, HA[19:0]. XADD should be left cleared after the address has been written to HPIA.
		0	Values written to HPIA go to HPIA(15-0). Value comes from HD[15:0].
		1	Values written to HPIA go to HPIA(19-16). Value comes from HD[3:0].
4-2	Reserved	0	Always write 0s to these bits. The read states of these bits are not defined.
1	DSPINT		Host-to-DSP interrupt request bit. The host can send a maskable interrupt request to the DSP CPU by writing a 1 to DSPINT. If the interrupt is properly enabled, the CPU will respond to the interrupt request; otherwise, the CPU will ignore it.
		0	Writing 0 to DSPINT has no effect. This bit always reads as 0.
		1	Writing 1 to DSPINT causes the HPI to send an interrupt request to the DSP CPU.

Table 6. Control Register (HPIC) Field Descriptions (Continued)

Bit	Field	Value	Description
0	RESET		Reset bit. This bit is cleared by a DSP reset. During HPI boot, the CPU will wait while the RESET bit is 0 before continuing code execution. When the RESET bit is set, the CPU will execute the loaded code.
		0	This bit is cleared by a DSP reset and will cause the CPU to wait for the code to be loaded during HPI boot mode.
		1	After HPI boot, set this bit to 1 to indicate that the boot is complete.
			Note: For detailed information on HPI boot, see the application report *Using the TMS320VC5510 Bootloader* (SPRA763).

Revision History

This document was revised to SPRU588A from SPRU588, which was released in April 2003.

Table 7 lists the changes made since the previous version of this document.

Table 7. Document Revision History

Page	Additions/Modifications/Deletions
Global	Added section numbers to page number references.
Global	Added overbars where necessary.
14	Updated bit $\overline{\text{HAS}}$ in Table 1, *Signals of the HPI*.
17	Updated section 4.1, clarified that HRDY is placed in a high−impedance state if $\overline{\text{HCS}}$ goes high and HPI is not selected.
17	Updated Figure 3, *HPI Strobe and Select Logic*, to include connection between $\overline{\text{HCS}}$ and HRDY.
22	Added sentence describing HPIA to first bullet in section 6, *Multiplexed Mode*.
24	Added Address/IO connections to Figure 6.
26	Replaced entire procedure list in section 6.3, *Loading HPIA with an Address*.
27	Added note about incrementation of HPIA under certain conditions to section 6.4, *Auto−Increment Option*.
28	Added note about clearing the DSPINT bit to section 7.1, *Sending an Interrupt Request from the Host to the DSP*.
30	Added note about host cycles to section 10.2, *Address Register (HPIA)*.

Index