Using the TMS320VC5509/5510 Enhanced HPI

Greg Shurtz — *C5000 Hardware Applications Team*

ABSTRACT

The enhanced host port interface (EHPI) on the TMS320VC5509 and TMS320VC5510 DSPs provides a 16-bit port through which a host device may access the internal and external memory space of the DSP. The flexibility of the EHPI interface allows various host devices to interface with the DSP using minimal or no external interface logic. This document describes various possible EHPI configurations and explains how to properly interface the EHPI to a host device. The document also explains the proper sequence for loading boot code to the DSP via EHPI. Summaries of EHPI signal descriptions, register definitions, and EHPI memory maps are included in Appendixes A, B, and C.

Contents

List of Figures

List of Tables

1 Introduction

The enhanced host port interface (EHPI) on the TMS320VC5509 and TMS320VC5510 DSPs provides a host device access to a portion of DSP memory space. The EHPI interface acts as a slave port through which the host processor (usually a microprocessor) can read or write to the internal or external memory of the DSP. Bi-directional communication is made possible through the use of host-to-DSP and DSP-to-host interrupts or through polling of memory locations by the DSP and host. The EHPI interface is extremely flexible, providing use of either a multiplexed address/data bus or separate address and data busses. Control and strobe logic is also flexible, allowing different host devices to interface with the EHPI using minimal or no external interface logic. The fundamental details of EHPI signaling and operation can be found in the *TMS320C55x Peripherals Reference Guide* (SPRU317) and the appropriate device data sheets.

The purpose of this document is to illustrate several possible hardware configurations of the EHPI interface and describe how to properly use the HRDY signal in implementing such an interface. This document also provides a description of the proper start-up sequence used to load boot code to the DSP via EHPI. Summaries of EHPI signal descriptions, register definitions, and EHPI memory maps are included in Appendixes A, B, and C.

NOTE: Throughout this document, the VC5509 and VC5510 are differentiated through several key features and requirements. Some early silicon revisions of the VC5510 (1.0 and 1.0A) possess the VC5509 EHPI characteristics detailed in this document as opposed to the VC5510 EHPI characteristics. Please refer to the VC5510 device data sheet and errata to determine which VC5510 device you are using. This note relates only to HRDY and strobe characteristics and not to memory maps or reset vector addresses. A table summarizing these differences can be found in Appendix D.

2 EHPI Hardware Configurations

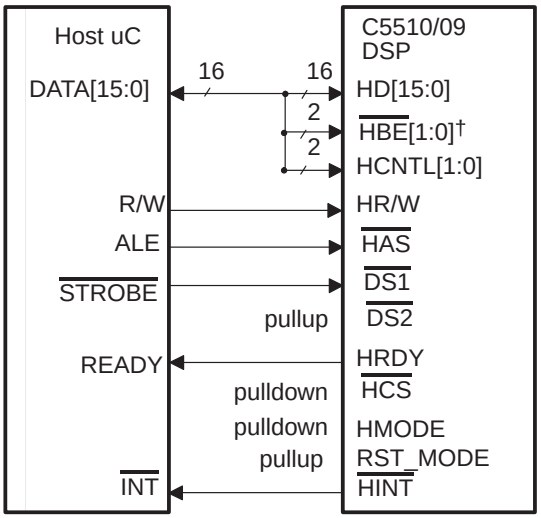
Figure 1 through Figure 5 illustrate several hardware configurations of theVC5509/5510 EHPI interface. In each configuration, the byte enable $\overline{\text{HBE}}$[1:0] can be optionally tied active low to permanently enable only word writes and reads. If byte enable outputs are available on the host device, these may also be connected to $\overline{\text{HBE}}$[1:0]. The $\overline{\text{HINT}}$ output can be optionally used to drive an interrupt to the host. The RST_MODE input in each example can be tied high to enable RESET in the HPIC register, or tied low to disable the bit (see section 4 for more details about the RESET bit). In each example, the $\overline{\text{HDS1}}$ input is used as an active low data strobe. There are several other strobe options that can be used in all the examples. These options are explained in section 2.6.

NOTE: The HBE signals are not supported on the 5510/5510A and must always be tied low.

2.1 Multiplexed Mode Using $\overline{\text{HAS}}$

If a host processor multiplexes its address and data information on the same bus, multiplexed mode and the $\overline{\text{HAS}}$ signal must be used. Multiplexed mode is selected by driving the HMODE input low. Such a host processor usually has an address strobe or address latch enable output (ALE). The address strobe is connected to the HAS input on the EHPI. In this configuration, the control signals (HCNTL[1:0] and $\overline{\text{HBE}}$[1:0]) are provided by the same host bus on which the data is transferred. Control information is latched on the falling edge of $\overline{\text{HAS}}$ and data information is latched on the rising edge of $\overline{\text{HDS1}}$ (see the appropriate device data sheet for timing requirements). The host can then access the HPIA, HPIC, or HPID registers by performing access cycles to addresses, which cause HCNTL[1:0] lines to be driven to the appropriate values during falling $\overline{\text{HAS}}$.

On some devices, when the EHPI is configured for multiplexed mode, internal pull-up resistors may be enabled on the unused address inputs (HA[19:0]). See the appropriate device data sheet for details.



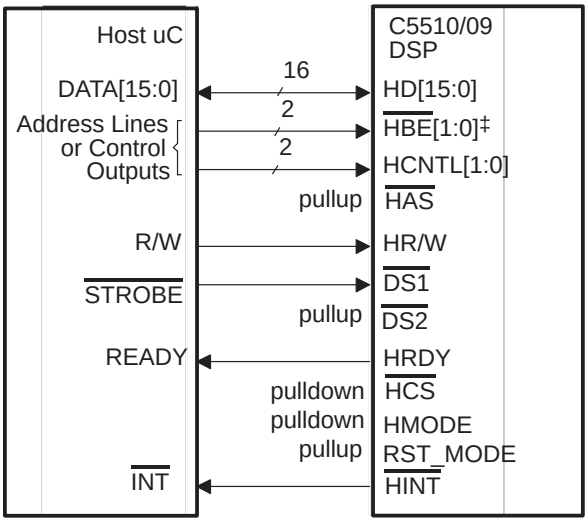
† The HBE signals are not supported on the 5510/5510A and must always be tied low.

Figure 1. EHPI Multiplexed Mode Using $\overline{\text{HAS}}$

2.2 Multiplexed Mode Without $\overline{\text{HAS}}$

Even if a host has separate address and data busses, multiplexed mode may be used to minimize the number of signal connections between the host and the DSP. In this case, $\overline{\text{HAS}}$ can be tied high since it is not used. Host address lines may be connected to the control signal HCNTL[1:0]. Control information is latched on the falling edge of $\overline{\text{HDS1}}$ and data information is latched on the rising edge of $\overline{\text{HDS1}}$ (see the appropriate device data sheet for timing requirements). The host can access the HPIA, HPIC, or HPID registers by performing access cycles to addresses, which cause the HCNTL[1:0] lines to be driven to the appropriate values during falling $\overline{\text{HDS1}}$. Likewise, byte-writes and reads may be performed (if desired) by connecting address lines to the $\overline{\text{HBE}}$[1:0] input. However, if you do not wish to use address lines, you can use the general-purpose I/O or another control logic to drive all of these control inputs.

On some devices, when the EHPI is configured for multiplexed mode, internal pull-up resistors may be enabled on the unused address inputs (HA[19:0]). See the appropriate device data sheet for details.



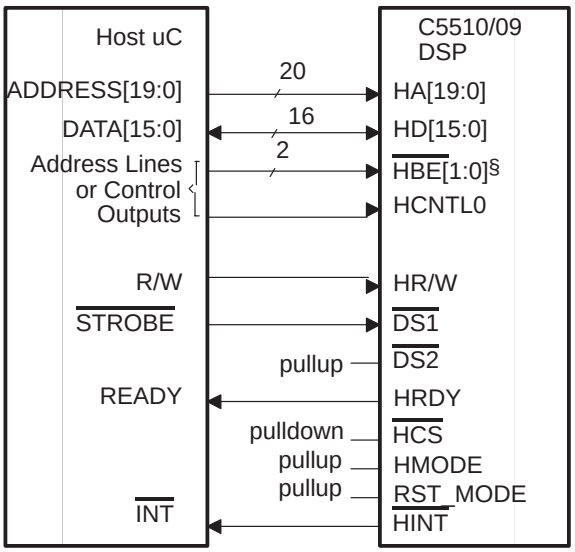
‡ The HBE signals are not supported on the 5510/5510A and must always be tied low.

Figure 2. EHPI Multiplexed Mode Without $\overline{\text{HAS}}$

2.3 Non-Multiplexed Mode

The EHPI has a 20-bit address input bus that allows a 16-bit host to access DSP memory through word addressing without the use of a multiplexed address/data bus. In the non-multiplexed mode of operation, the host address bus must be connected directly to the EHPI address inputs (HA[19:0]) of the DSP. This mode is selected by driving the HMODE input high. Since the DSP memory is addressed directly via the address lines, the HPIA register is not used. The HPIC register is still accessible for reset and interrupt control. The HPIC register is accessed by driving the HCNTL0 input low during the cycle using an address line or another control signal that the host may have available. For data (HPID) cycles, HCNTL0 must be driven high.

On the VC5510, when the EHPI is configured in non-multiplexed mode, the physical pins used for HCNTL1 and $\overline{\text{HAS}}$ become the HA1 and HA2 inputs respectively. These pins are reused because HCNTL1 and $\overline{\text{HAS}}$ are not required in this mode.



§ The HBE signals are not supported on the 5510/5510A and must always be tied low.

Figure 3. EHPI Non-Multiplexed Mode

2.4 EHPI Interfacing With 8-Bit Host in Non-Multiplexed Mode (Not Supported on 5510/5510A)

The active low byte enable inputs can be used to do byte reads and byte writes of DSP memory by enabling only the upper 8 bits or the lower 8 bits of the host data bus. During byte reads of the HPID, the disabled byte of the EHPI data bus is tri-stated. This allows the interfacing of an 8-bit host to the EHPI by connecting the 8 bits of the host data bus to both the upper (HD[15:8]) and lower (HD[7:0]) bytes of the EHPI data bus. The byte enable inputs ($\overline{\text{HBE}}$[1:0]) can be driven by address lines or other control logic. In this case, it is imperative that $\overline{\text{HBE}}$[1] and $\overline{\text{HBE}}$[0] always be mutually exclusive. If both of these byte enables are driven active low simultaneously, bus contention occurs on the EHPI data bus. To ensure mutual exclusion, it is highly recommended that $\overline{\text{HBE}}$[1] be driven with the inversion of the control signal or address line that is driving $\overline{\text{HBE}}$[0]. This forces mutual exclusion in the hardware. If the mutual exclusion of the byte enables is enforced in software alone, the danger of bus contention exists, which could result in damage to the device if the software were to fail.

This method only works in non-multiplexed mode with the HCNTL0 input tied to logic 1 with a pull-up resistor. This ensures that no HPIC or HPIA accesses are possible. HPIC and HPIA register accesses do not utilize the byte enable inputs. A write to these registers always updates all 16-bits of the registers regardless of the $\overline{\text{HBE}}$[1:0] settings. Likewise, a read from these registers always drives all 16 outputs of the data bus (HD[15:0]). The $\overline{\text{HBE}}$[1:0] inputs only affect HPID register accesses. Therefore, the upper and lower bytes can never be bussed together when using multiplexed mode or when accessing the HPIC register. Using an 8-bit host in these modes requires external logic to demultiplex the 8-bit host bus into 16-bit data.

Figure 4 illustrates an example of interfacing an 8-bit host to EHPI. In this case, the host address lines AD[20:1] are connected to the EHPI address inputs HA[19:0]. The mapping of the host address lines to the EHPI address lines is shifted by one bit (AD[1]→HA[0], AD[2]→HA[1], etc.). The least significant host address line and its inversion are used to drive the EHPI byte enables (AD[0]→$\overline{\text{HBE}}$[0] and ~AD[0]→$\overline{\text{HBE}}$[1]). This allows the host to access DSP memory using byte addressing. Note that RST_MODE is tied low to disable the RESET bit in the HPIC. This is because the HPIC is not accessible in this configuration. If RST_MODE was tied high in this configuration, the DSP would stay in permanent reset since the RESET bit is inaccessible (see section 4 for details on the RESET bit).

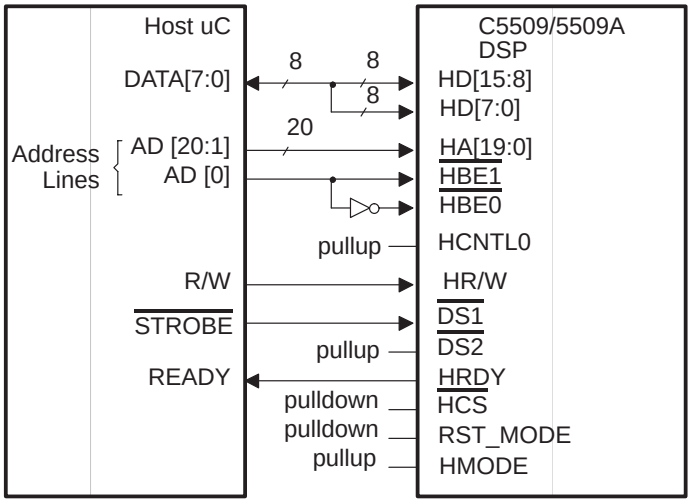


Figure 4. EHPI Interfacing With 8-Bit Host in Non-Multiplexed Mode

Table 1 illustrates a sequence of host writes utilizing the configuration shown in Figure 4. Note that the $\overline{\text{HBE}}$[1:0] values are both derived from the host address output AD[0].

Table 1. 8-Bit Host Writes in Non-Multiplexed Mode

Host Cycle	Host Data Bus [7:0]	Host Address Output AD[20:0]	HA[19:0]	$\overline{\text{HBE}}$[1:0]
Host write cycle	10h	010000h	8000h	10
Host write cycle	32h	010001h	8000h	01
Host write cycle	54h	010002h	8001h	10
Host write cycle	76h	010003h	8001h	01
Host write cycle	98h	010004h	8002h	10
Host write cycle	BAh	010005h	8002h	01
Host write cycle	DCh	010006h	8003h	10
Host write cycle	FEh	010007h	8003h	01

The sequence shown in Table 1 results in the following data values being stored in DSP memory (see Table 2).

Table 2. Data Values Stored in DSP Memory

Word Address	Data in Memory
8000h	3210h
8001h	7654h
8002h	BA98h
8003h	FEDCh

2.5 Interfacing a Host to Multiple DSPs

The EHPI data bus (HD[15:0]) outputs are tri-stated when $\overline{\text{HCS}}$ is inactive-high. This allows the EHPI interfaces of several DSP devices to be bussed together. Each $\overline{\text{HCS}}$ must be driven by a mutually exclusive chip select signal provided by the host. If the $\overline{\text{HCS}}$ input of more than one DSP is driven active low at the same time, contention may occur and device damage is possible. Therefore, it is crucial that all $\overline{\text{HCS}}$ sources are mutually exclusive.

On the VC5510, the HRDY output is also tri-stated and may be bussed and connected to the READY input of the host device. On the VC5509, the HRDY output is not tri-stated, but is always driven. Multiple HRDY outputs must never be tied together on VC5509 devices. In this case, the HRDY signals must be multiplexed or bussed by external logic, or the HRDY must be ignored. In the worst case, the HRDY timing must be assumed (see section 3.2). The VC5510 HRDY output has an internal pull-up. Therefore, an external pull-up is not necessary.

Using this configuration, a single host may access memory on multiple DSP devices by driving the $\overline{\text{HCS}}$ input low on the desired DSP and performing the appropriate host cycle.

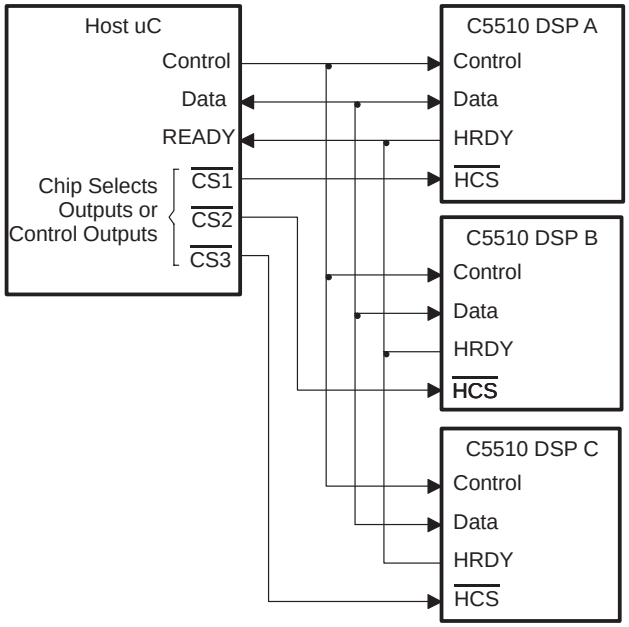


Figure 5. EHPI Using Multiple DSPs (VC5510)

The host can initiate a new cycle on another DSP even if the internal portion of the previous cycle has not yet completed. This applies to consecutive cycles that are multiplexed or non-multiplexed writes, or multiplexed reads with auto-increment. Figure 6 and Figure 7 illustrate this concept. It is more apparent in Figure 7 as HRDY_A is still low when DSP_B completes its first write cycle. In Figure 6, this is less apparent because the HRDY output of DSP_A is tri-stated when $\overline{\text{HCS}}$_A goes high. Even though the HRDY bus is allowed to go high after $\overline{\text{HCS}}$_A goes high, internal write activity to memory is still occurring in DSP_A.

Notice in Figure 6 that when the host performs the second write cycle to DSP_B, the HRDY is still low from the previous cycle. The host is able to initiate another write cycle to DSP_B, even though the HRDY was driven low. This is only possible on the VC5510 DSP. On the VC5509 (Figure 7), the host must wait for HRDY to be driven high by the DSP before initiating a new cycle with falling strobe ($\overline{\text{HDS}}$). In either case, any current host cycle cannot complete until the HRDY output of the DSP currently being accessed is driven high.

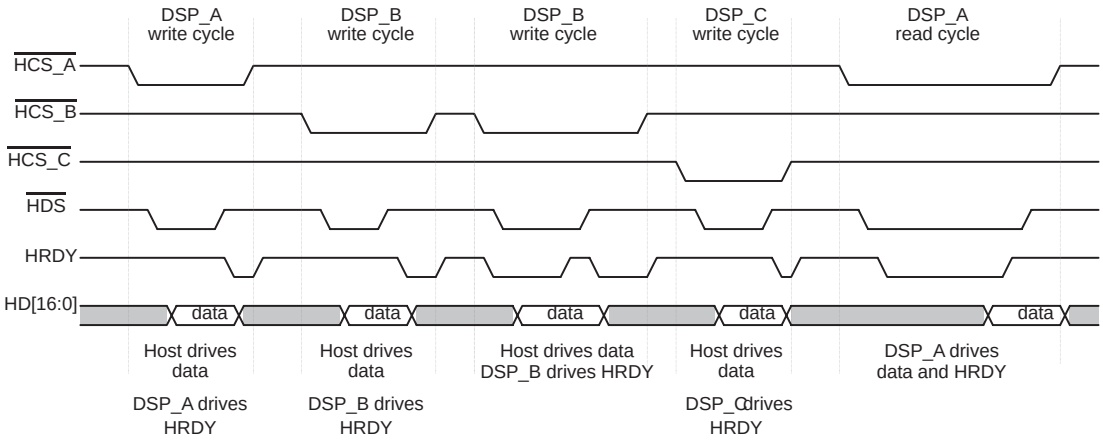


Figure 6. Multiple DSPs With HRDYs Bussed Together (VC5510)

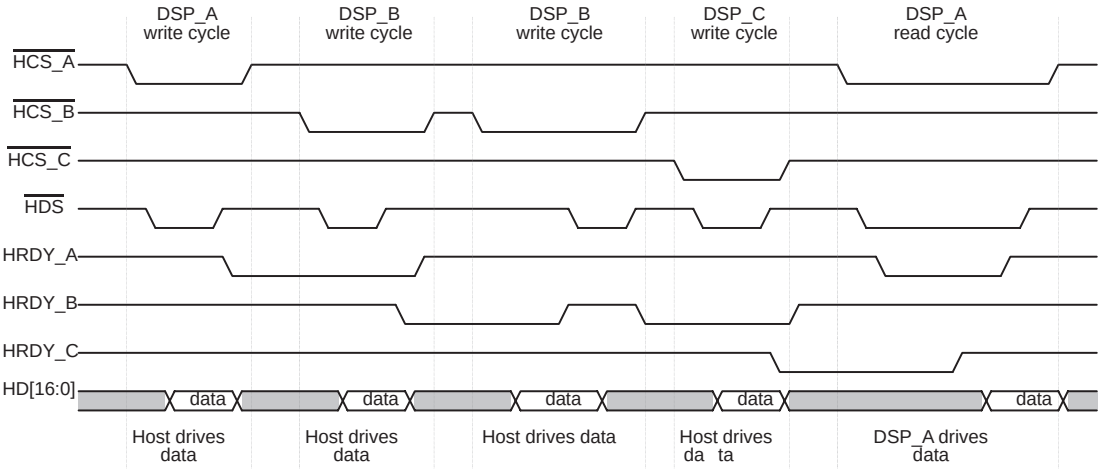


Figure 7. Multiple DSPs With HRDYs Sampled Separately (VC5509)

NOTE: When bussing output signals together, it is important to ensure that there is no case where bus contention occurs. Such a design needs to take into account the delay times from $\overline{HCS}$ going inactive to the HD[15:0] and HRDY outputs being tri-stated. See the appropriate device data sheets for all device-switching characteristics.

2.6 Strobe Options

The internal strobe that actually controls the EHPI accesses is derived from the logical exclusive NOR of the strobe inputs, HDS1 and HDS2. The strobe is qualified by the chip-select signal $\overline{HCS}$. The EHPI strobe logic allows the host to utilize a number of different strobe options. Table 3 illustrates these configurations.

Table 3. Strobe Options

Host Strobe Configuration	Connection to HDS1	Connection to HDS2
Single active high strobe	Pulldown resistor	Host strobe
	Host strobe	Pulldown resistor
Single active low strobe	Pullup resistor	Host strobe
	Host strobe	Pullup resistor
Active high read and write strobes†	Host read strobe	Host write strobe
	Host write strobe	Host read strobe
Active low read and write strobes†	Host read strobe	Host write strobe
	Host write strobe	Host read strobe

† Note: If a host has a read and a write strobe and there is no read/write (R/W) control signal available, the HR/W input of the EHPI interface must be connected to an address line, another control logic, or the general-purpose I/O from the host. The host must then control whether each access is a read or a write by using the correct address, or correctly driving the control logic connected to HR/W. The read or write strobes themselves cannot be connected to the HR/W input since there is a setup time that must be enforced from the HR/W input to the active edge of the strobe. See the appropriate device data sheets for all timing requirements.

3 HRDY Latency

Unlike host port interfaces on the TMS320C54x generation of devices such as HPI-16, host accesses do not automatically have precedence over DMA channel accesses. Therefore the HRDY latency varies according to the number of active DMA channels and the priority settings of those channels. The EHPI is treated like any other DMA channel in terms of arbitration. A host cycle cannot complete until HRDY is high. HRDY is held low until any current DMA channels with higher priority complete their accesses. If the EHPI has the same priority as any active DMA channels, the accesses take place in round robin fashion. Because of this behavior, the maximum HRDY latency is only specified for the case in which there is no DMA activity. See the appropriate device data sheet for specific timing details.

It is suggested that the EHPI priority be set equal to or higher than any DMA channels that might be active during host accesses. This prevents the host from being starved out by the DMA (see section 3.2 for details).

3.1 Using HRDY to Extend Cycles

Because of the nature of the arbitration between EHPI and DMA channels, it is highly recommended that you use the HRDY output of the EHPI to extend host cycles when necessary. This method helps to achieve maximum possible throughput. The HRDY output must be connected to the ready input of the host. When HRDY is low during a host cycle, the host must extend the cycle until HRDY is high before driving the strobe inactive to end the cycle. On the VC5510, the host can begin a new cycle while HRDY is still low (falling strobe), but cannot end a cycle while HRDY is low (rising strobe). On the VC5509, the host HRDY signal must be high before the host can begin or end a cycle (falling or rising strobe). EHPI throughput is maximized when using HRDY since each cycle completes as soon as possible.

Figure 8 and Figure 9 illustrate a series of host writes to memory (either multiplexed or non-multiplexed). The host initiates each write cycle by driving the strobe low with the control lines valid designating an HPID write. Each rising strobe edge driven by the host causes the EHPI to kick off an internal DMA cycle to retrieve the data from memory. In both examples, the host can only drive $\overline{\text{HDS}}$ high when the HRDY output from the DSP is high, indicating that the previous internal cycle is complete and the EHPI is ready for another cycle.

In the case of theVC5510, notice that the host is allowed to initiate each subsequent cycle while HRDY is still low. On the VC5509, however, HRDY must be high before a host cycle can be initiated by a falling strobe. In both cases, each host cycle completes rapidly since the host uses the HRDY signal to extend the cycles no more than is necessary. This is the most efficient usage of the EHPI.

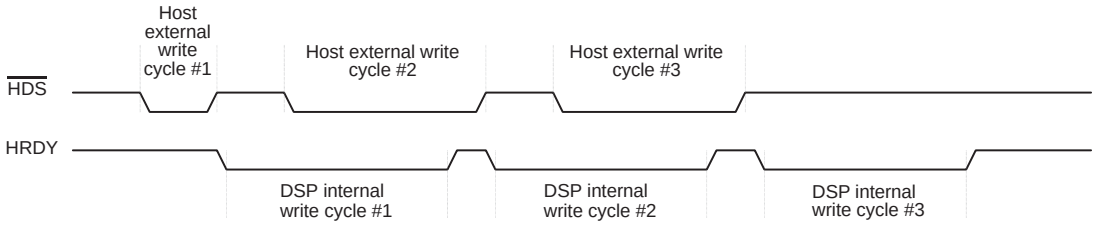


Figure 8. HRDY During Host Writes (VC5510)

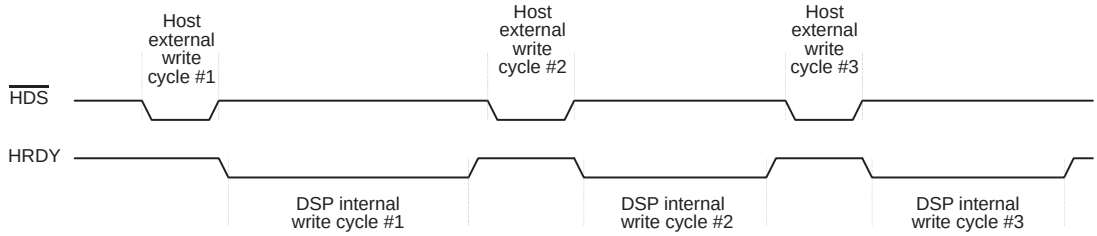


Figure 9. HRDY During Host Writes (VC5509)

Figure 10 and Figure 11 illustrate a series of host reads from memory in multiplexed mode using auto-increment. During the initial host read cycle (first read after HPIA is modified), the falling host strobe causes the EHPI to kick off an internal read cycle, driving HRDY low. When HRDY goes high and the read data becomes valid, the host can complete the cycle by driving $\overline{\text{HDS}}$ high and latching the read data. There may be a small delay before the read data is valid after HRDY goes high (see the appropriate device data sheet for switching characteristics). Because this is an auto-increment cycle, the rising strobe also causes the HPIA address register to increment and kicks off another internal read cycle to memory in anticipation of another host read. HRDY goes low again until the internal read cycle is complete. The host can initiate a subsequent read cycle by driving the strobe low. For the VC5510, this can be done while HRDY is still low, as shown in Figure 10. In the case of the VC5509, shown in Figure 11, the host must wait for HRDY to be driven high before driving the strobe low to initiate a new cycle. In all cases, the host must wait for HRDY to be driven high, before ending the read cycle with a rising strobe.

Again, this is the most efficient usage of the EHPI because each host cycle completes rapidly by using HRDY to extend the cycles no more than is necessary.

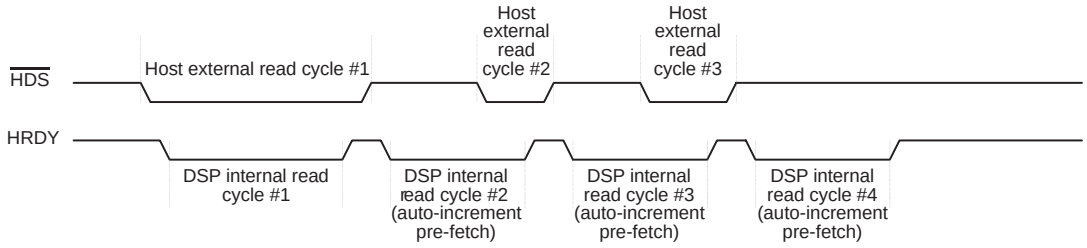


Figure 10. HRDY During Multiplexed Reads With Auto-Increment (VC5510)

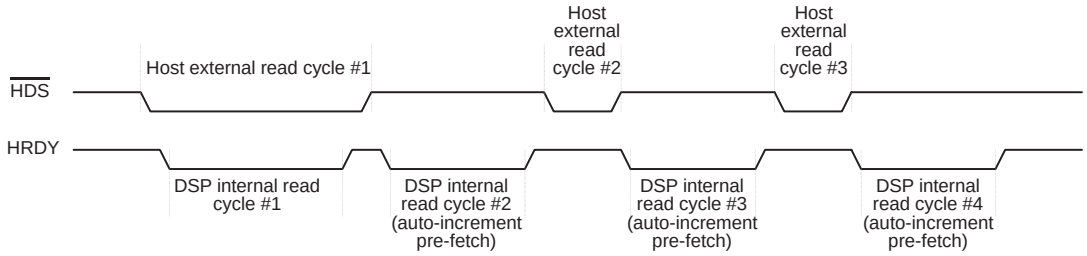


Figure 11. HRDY During Multiplexed Reads With Auto-Increment (VC5509)

Figure 12 illustrates a series of host reads from memory in non-multiplexed mode. In this mode, there is no pre-fetching of data since there is no HPIA register and no auto-incrementing. Therefore, each falling strobe kicks off a new internal memory read cycle from the address provided on the HA[19:0] address lines. For each cycle, the host must wait for HRDY to be driven high before completing the read cycle.

Although the host does not begin a new cycle before the previous internal cycle is complete, using HRDY in this manner still provides the most efficient usage of EHPI in this mode since each cycle completes as soon as possible.

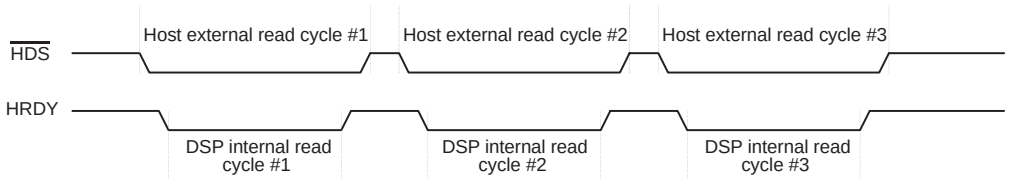


Figure 12. HRDY During Non-Multiplexed Reads (VC5509/VC5510)

3.2 Ignoring HRDY and Assuming Worst Case

If it is impossible to use the HRDY signal to insert wait states during host cycles, the host must then always assume the worst-case HRDY latency for each situation.

In previous HPI interfaces such as HPI-16 on TMS320C54x devices, the HRDY latency of all HPI accesses were limited to a finite maximum number of CPU cycles. This is not the case with the EHPI interface. Because of this fundamental difference, it becomes more difficult to design a system in which the HRDY output is ignored. In such a system, the worst-case HRDY latency is assumed for all cycles. Basically, to accomplish this, the system design must be characterized by measuring HRDY delay during all conditions of system operation to determine the longest HRDY latency for that system. Several CPU cycles of margin must be added to the measured maximum to arrive at a worst-case HRDY latency. The host must always assume, that every cycle has an HRDY latency equal to this worst-case number.

In designing such a system, it is suggested that the HPI priority always be set equal to or higher than the priorities of all active DMA channels. If the EHPI priority is 0 and an active DMA is set to 1, it is possible for the DMA to starve out the EHPI indefinitely with HRDY low for microseconds at a time (as shown in Table 4). Care must be taken when designing a DSP system so that such situations are avoided or do not cause real-time deadlines to be missed.

Setting the HPI priority to 1 and all DMA priorities to 0 minimizes the worst-case HRDY latency in the system. In all cases, if the HRDY output is not sampled by the host, it is necessary to characterize the system to determine worst-case HRDY latency. This is because the HRDY latency can vary greatly depending upon the following factors:

- Number of active DMA channels
- DMA and HPI priority settings
- Which memory blocks are being accessed by the DMA channels (SARAM/DARAM/external)
- Which memory blocks are being accessed by the host device (SARAM/DARAM/external)
- Wait states and external memory timing if external memory space is being accessed by the host via EHPI or by other DMA channels.
- Execution of DSP instructions that may cause accesses to memory

Table 4 illustrates how different combinations of these factors cause differing results in terms of maximum HRDY latency. All active DMA channels in the following examples are memory-to-memory DMAs. These numbers are measured in specific systems and are not guaranteed maximums for the associated configurations.

Table 4. HRDY Latencies

Maximum HRDY Latency	HPI Priority	DMA Priorities	System Configuration
15 CPU cycles	1	0	5 DMA channels active. All DMA source addresses are in DARAM and all DMA destination addresses are in SARAM. Host is accessing SARAM.
22 CPU cycles	0	0	5 DMA channels active. All DMA source addresses are in DARAM and all DMA destination addresses are in SARAM. Host is accessing SARAM.
17 CPU cycles	1	0	6 DMA channels active. All DMA source and destination addresses are in SARAM. Host is accessing SARAM. No DSP activity.
19 CPU cycles	1	0	6 DMA channels active. All DMA source and destination addresses are in SARAM. Host is accessing SARAM. DSP instructions are causing SARAM accesses.
~80 microseconds	0	1	1 DMA channel active doing memory-to-memory copy. Host is accessing SARAM.

There are several ways to implement a host interface that assumes worst-case HRDY latency. The active strobe pulse width can be fixed to be longer than the worst-case HRDY latency, or the host can ensure a fixed delay between cycles that is greater than the worst-case HRDY latency.

Figure 13 illustrates a series of host writes to memory (either multiplexed or non-multiplexed). Since the host does not use the HRDY signal, it has no visibility to know whether or not the previous internal cycle is complete. Therefore we must always assume a worst-case delay for the internal cycles. In this example, the host strobe width is fixed to always be greater than the maximum HRDY delay in the system. This ensures that the host never drives the strobe high to end the cycle while HRDY is still low. As seen in the example, ignoring HRDY is less efficient since the host cannot take advantage of the cases where the EHPI internal cycle delay is less than the worst case. There is also an efficiency hit on the very first write access since the EHPI is ready immediately, even though the host extends the cycle because of the fixed strobe width. This implementation cannot be used on the VC5509 because cycles cannot be initiated while HRDY is low on the VC5509.

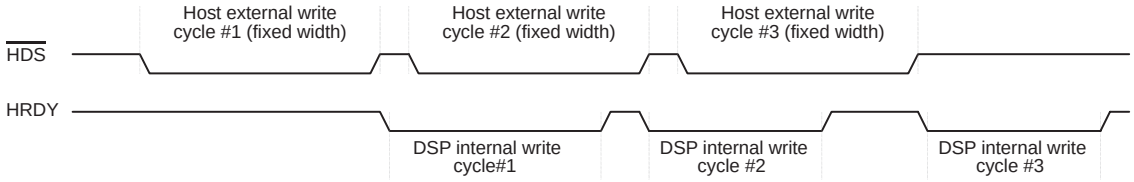


Figure 13. Fixing Strobe Width During EHPI Writes (VC5510)

Figure 14 illustrates a series of host writes to memory (either multiplexed or non-multiplexed). In this example, instead of extending each host cycle, the same result is achieved by forcing a fixed delay between each host cycle. Again, this ensures that the host never drives the strobe high to end the cycle while HRDY is still low. On the VC5509, the delay inserted between host cycles must ensure that falling $\overline{\text{HDS}}$ never occurs while HRDY is low. On the VC5510, the delays must ensure that rising $\overline{\text{HDS}}$ never occurs while HRDY is low. However, since it is recommended that several DSP clock periods of margin be allowed for the worst-case HRDY delay value, the VC5510 implementation can ensure that HRDY is high before falling $\overline{\text{HDS}}$ occurs. The host can perform other operations (except for the EHPI operations) during this delay time, as long as the delay between cycles is always greater than the worst-case HRDY latency.

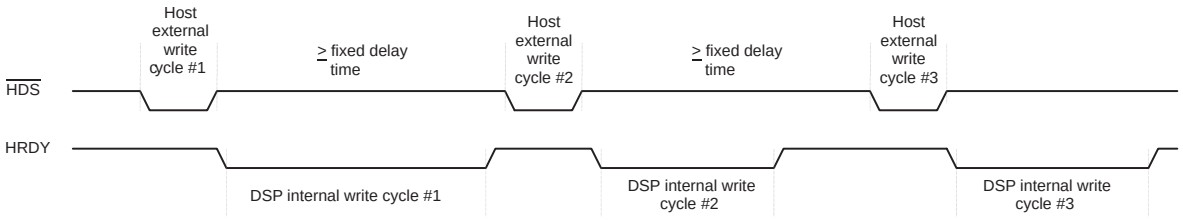


Figure 14. Forcing Delay Between Cycles for EHPI Writes (VC5509/VC5510)

Figure 15 illustrates a series of host reads from memory in multiplexed mode using auto-increment. As in Figure 13, the active strobe width is fixed to be greater than the worst-case HRDY delay. This ensures that the host never drives the strobe high to end the cycle while HRDY is still low. This implementation cannot be used on the VC5509 because cycles cannot be initiated while HRDY is low on the VC5509.

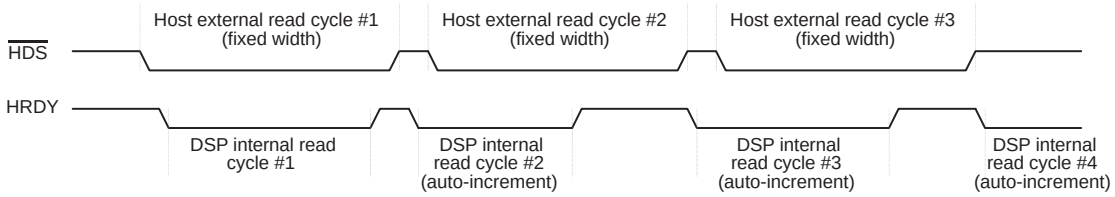


Figure 15. Fixing Strobe Width During Multiplexed Reads (VC5510)

As in Figure 15, Figure 16 illustrates a series of host reads from memory in multiplexed mode using auto-increment. In this case, the active strobe width of the initial read cycle is fixed to be greater than the worst-case delay. All subsequent read cycles, however, have short active strobes, but a fixed delay is forced between each cycle. This achieves the same basic result as the previous example. This implementation can be used with the VC5509 or the VC5510. On the VC5509, the delay inserted between host cycles must ensure that falling $\overline{\text{HDS}}$ never occurs while HRDY is low. On the VC5510, the delays must ensure that rising $\overline{\text{HDS}}$ never occurs while HRDY is low. However, since it is recommended that several DSP clock periods of margin be allowed for the worst-case HRDY delay value, the VC5510 implementation can guarantee that HRDY is high before falling $\overline{\text{HDS}}$ occurs.

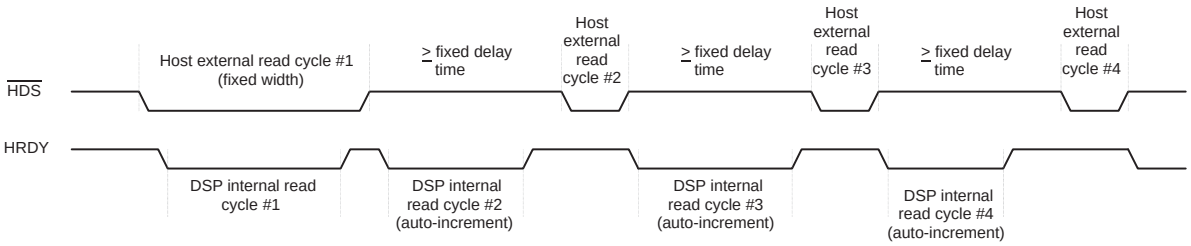


Figure 16. Forcing Delay Between Cycles for Multiplexed Reads (VC5509/VC5510)

The only option during non-multiplexed reads is to fix the $\overline{\text{HDS}}$ low pulse width to always be greater than the maximum HRDY delay in the system. This is because the falling $\overline{\text{HDS}}$ signal of each host read causes an internal DSP cycle that must complete before the cycle is terminated with rising $\overline{\text{HDS}}$. This implementation is valid for both the VC5510 and VC5509 and is illustrated in Figure 17.

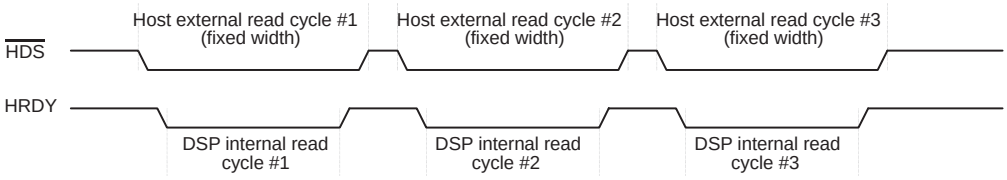


Figure 17. Fixing Strobe Width During Non-Multiplexed Reads (VC5509/VC5510)

NOTE: HPIC and HPIA accesses do not cause HRDY to be deasserted, since these accesses do not cause an internal DMA access to occur. HRDY is only a concern during HPID cycles. In all of the examples in sections 3.1 and 3.2, minimum strobe widths (high and low) must be satisfied. See the appropriate device data sheet for specific timing requirements.

4 RESET Behavior

The RESET bit in the HPIC register allows the host to control the time at which the DSP comes out of reset and fetches its reset vector to begin code execution. Upon device power up, if the DSP is configured for EHPI boot mode, the DSP remains in reset even after the DSP device reset input pin is released. The DSP is held in reset until the host writes a 1 to the RESET bit position in the HPIC register. The RESET bit defaults to 0 upon DSP power up.

While the RESET bit is 0, the host can load code into DSP memory. Setting the RESET bit to 1 causes the DSP to fetch its reset vector and begin executing code. If the DSP is configured in EHPI boot mode, it then branches to the correct reset interrupt vector address and begins executing code. In the case of the VC5510, the DSP branches to the first SARAM location (word address 8000h, byte address 10000h). See the *TMS320C55x Peripherals Reference Guide* (SPRU317) and device data sheet for complete boot mode options and reset vector addresses. The process for loading boot code via EHPI on the VC5510 is as follows:

1. DSP is powered up in EHPI boot mode with RST_MODE high.
2. DSP hardware reset input is deasserted.
3. Host writes boot code to SARAM via EHPI beginning at word address 8000h.

4. Host writes to HPIC with 1 in RESET bit position.
5. DSP fetches reset vector and begins executing code at program address 10000h (word address 8000h).

On the VC5510, the RST_MODE input is used to enable the function of the RESET bit in the HPIC. If the RST_MODE input is tied low, the RESET bit in the HPIC is ignored. In this case, the DSP immediately fetches its reset vector when the DSP reset input pin is deasserted. The host can then access the entire DARAM and SARAM portions of the DSP memory, and a portion of the external memory. If RST_MODE is tied high, the RESET bit is enabled, which means that the DSP is not able to fetch its reset vector until RESET is set to 1 by the host. In this case, host access is limited to the on-chip SARAM until RESET is set to 1. See Appendix C for complete EHPI memory maps.

The RESET bit is write-only. The host always reads a 0 in this bit position. After the DSP is running, all HPIC writes must have a 1 in the RESET bit position; otherwise, the DSP will be put in reset. If the RESET bit is written to 0, the DSP again is held in reset until the host sets the RESET bit to 1, at which point, the DSP branches to the appropriate reset vector. This applies only if the RESET function is enabled with RST_MODE tied high. If RST_MODE is tied low, then the RESET bit has no effect on DSP operation. RST_MODE is a static configuration input that is not expected to change during DSP operation.

NOTE: In some early versions of the TMS320CC55x™ (C55x™) documentation, RST_MODE is referred to as EHPIENA.

5 References

1. *TMS320C55x Peripherals Reference Guide* (SPRU317)
2. *TMS320VC5510 Fixed-Point DSP* (SPRS076)
3. *TMS320VC5509 Fixed-Point DSP* (SPRS163)

Appendix A EHPI Signal Descriptions

Table A−1. EHPI Signal Descriptions

Name	Type	Description
HD[15:0]	In/Out/Z	Host data bus. Carries read and write data (and address information in multiplexed mode). These signals are always tri-stated when $\overline{\text{HCS}}$ is inactive (high) or when the corresponding byte-enable is inactive (high) during HPID cycles. These pins are not tri-stated during HPIA and HPIC cycles regardless of byte enable settings.
HA[19:0]	In	Host address bus. Host drives address onto these lines in non-multiplexed mode.
$\overline{\text{HBE}}$[1:0]	In	Host byte enables. Active low byte enables. When $\overline{\text{HBE1}}$ = 0, enables MSB is enabled. When $\overline{\text{HBE0}}$ = 0, LSB is enabled.
$\overline{\text{HCS}}$	In	Chip select. When driven inactive (high), HD[15:0] outputs are tri-stated. On the VC5510, the HRDY output is also tri-stated when $\overline{\text{HCS}}$ is inactive.
HR/$\overline{\text{W}}$	In	Read/write signal. Indicates whether the current cycle is a read or a write cycle.
$\overline{\text{HDS1}}$, $\overline{\text{HDS2}}$	In	Data strobes. The logical exclusive NOR of these two inputs creates the active low internal strobe controlling data transfers during EHPI cycles.
HRDY	Out/Z	Ready output. Indicates that the current host cycle may be completed.
HCNTL[1:0]	In	Control inputs. These inputs allow the host to select the access type of the current cycle (HPIA access, HPIC access, HPID access, HPID w/auto-increment).
$\overline{\text{HAS}}$	In	Address strobe. This input may be used to latch control information during host cycles (available in multiplexed mode only).
HMODE	In	Mode. Selects between multiplexed (low) and non-multiplexed (high) modes.
RST_MODE†	In	Enables RESET bit in HPIC. When RST_MODE=1, RESET is enabled.
$\overline{\text{HINT}}$	Out	Interrupt output. Allows the DSP to interrupt the host processor.

† The RST_MODE input is referred to as the EHPIENA input pin in some early versions of the C55x documentation.

Appendix B Registers and Descriptions

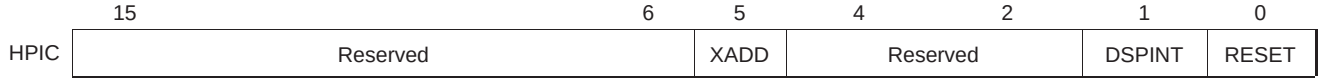


Figure B−1. HPIC Register

Table B−1. HPIC Register Description

Name	Bit Position	Reset Value	Description
Reserved	[15:6]		These bits are reserved and must always be written to 0 during all HPIC writes.
XADD	5	0	This bit selects between the HPIA register and the extended HPIA register during HPIA read/write cycles. XADD = 0 selects the HPIA register or address bits 15:0. XADD = 1 selects the extended HPIA register or address bits 19:16.
Reserved	[4:2]		These bits are reserved and must always be written to 0 during all HPIC writes.
DSPINT	1	0	A host write of 1 to this bit position causes an interrupt condition to the DSP. The host always reads a 0 in this bit position and writes of 0 have no effect.
RESET	0	0	When RESET = 0 and RST_MODE = 1, the DSP is held in reset and does not jump to its reset interrupt vector location. A host write of 1 to this bit position releases the DSP from reset allowing it to jump to its reset vector. When RST_MODE = 0, this bit has no effect. The host always reads a 0 in this bit position.

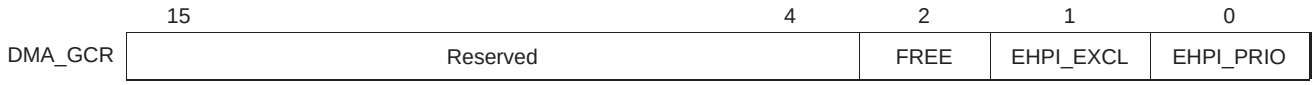


Figure B−2. DMA Global Control Register (DMA_GCR)

Table B−2. DMA Global Control Register Description

Name	Bit Position	Reset Value	Description
FREE	2	0	This bit allows the DMA to continue functioning after the CPU is halted via the emulation port. It must be set to 1 to allow this functionality. When set to 0, host cycles to memory while the CPU is halted causes HRDY to go low until the CPU is allowed to run.
EHPI_EXCL	1	0	When this bit is set to 1, only the EHPI can access the memory via DMA. DMA channels cannot access the CPU's memory map unless this bit is set to 0.
EHPI_PRIO	0	0	This bit sets the priority of the EHPI port's channel to the DMA. 1 is high priority and 0 is low priority. Each DMA channel can also be set to high or low priority.

Appendix C EHPI Memory Maps

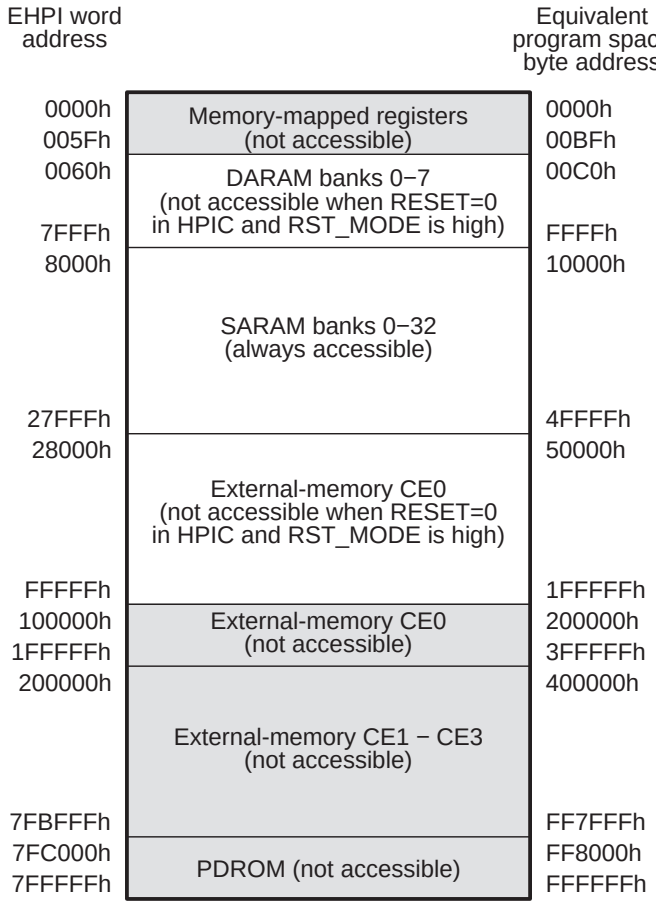


Figure C−1. TMS320VC5510 EHPI Memory Map

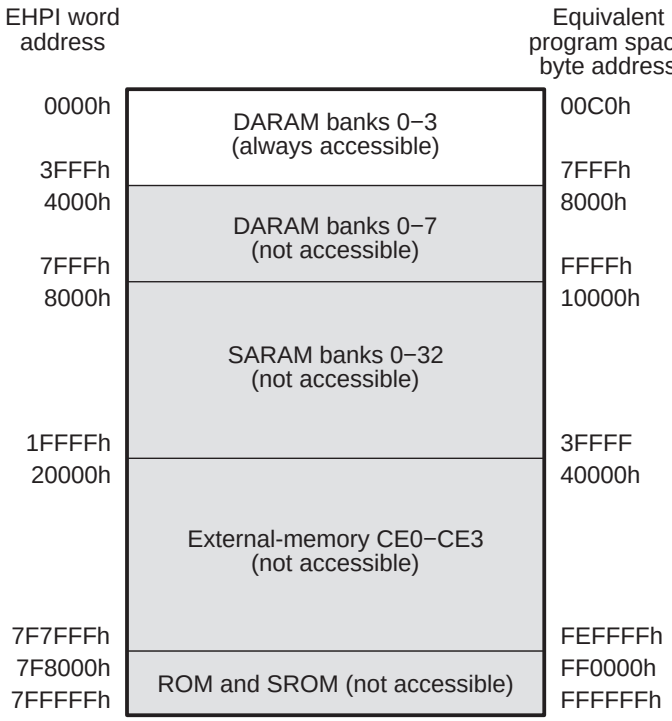


Figure C−2. TMS320VC5509 EHPI Memory Map

NOTE: Host accesses to DSP memory via the EHPI interface utilize word addressing. Each address held by the HPIA register or driven onto the HA[19:0] inputs corresponds to a 16-bit word in DSP memory. On the VC5510, some portions of memory are only accessible if the RESET bit in the HPIC is high or if the RST_MODE input is tied low.

Appendix D Differences Between the VC5509 and VC5510

Table D−1. VC5509 and VC5510 Differences

Characteristic	VC5509	VC5510 rev 1.0 and 1.0A	VC5510 rev 2.0 and up
HRDY is tri-stated when $\overline{\text{HCS}}$ = 1; HRDY outputs may be bussed together			√
HRDY is always driven regardless of $\overline{\text{HCS}}$	√	√	
Host may initiate a cycle with falling $\overline{\text{HDS}}$ while HRDY is low			√
Host must wait until HRDY is high before initiating a cycle with falling $\overline{\text{HDS}}$	√	√	
Host has access to on-chip DARAM, SARAM, and portion of external memory		√	√
Host only has access to portion of DARAM	√		

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

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