TCM320C6201/6701 DSP Host Port Interface (HPI) Performance

APPLICATION REPORT: SPRA449

Zoran Nikolic

Digital Signal Processing Solutions
April 1998

TEXAS INSTRUMENTS

IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Certain application using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards should be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.

TRADEMARKS

TI is a trademark of Texas Instruments Incorporated.

Other brands and names are the property of their respective owners.

CONTACT INFORMATION

US TMS320 HOTLINE	(281) 274-2320
US TMS320 FAX	(281) 274-2324
US TMS320 BBS	(281) 274-2323
US TMS320 email	dsph@ti.com

Contents

Figures

Tables

TCM320C6201/6701 DSP Host Port Interface (HPI) Performance

Abstract

This document describes how to determine the number of CPU cycles required to transfer data between the host CPU and the Texas Instruments (TI™) TMS320C6201/6701 digital signal processor (DSP) using the Host Port Interface (HPI).

The host has direct access to the processor's data memory via the 16-bit HPI. The number of cycles required for a data transfer between the host CPU and the TMS320C6201/6701 depends on the transfer type (read and write, with and without auto-increment, and internal and external memory type).

This document provides equations to calculate the time required to read and write a data word to the TMS320C6201/6701 DSP memory. Timing diagrams are included to illustrate signal relationships for memory reads and writes.

Product Support

Related Documentation

The following list specifies product names, part numbers, and literature numbers of corresponding TI documentation.

- *TMS320C6201 Digital Signal Processor* Data Sheet, March 1998, Literature number SPRS051C
- *TMS320C6201/C6701 Peripherals Reference Guide*, March 1998, Literature number SPRA190C

World Wide Web

Our World Wide Web site at **www.ti.com** contains the most up to date product information, revisions, and additions. Users registering with TI&ME can build custom information pages and receive new product updates automatically via email.

Email

For technical issues or clarification on switching products, please send a detailed email to **dsph@ti.com**. Questions receive prompt attention and are usually answered within one business day.

Design Problem

How many CPU cycles does it take to transfer data between the host CPU and TMS320C6201/6701 DSP using the Host Port Interface (HPI)?

Solution

A host processor can directly access TMS320C6201/6701 memory through the HPI. The number of cycles required for a data transfer between the host CPU and the TMS320C6201/6701 depends on the transfer type and the memory type to which or from which, the transfer is taking place. The transfer types are:

- Read with auto-increment
- Read without auto-increment
- Write with auto-increment
- Write without auto-increment

The HPI provides 32-bit data to the CPU with a 16-bit external asynchronous interface. This is accomplished by automatically combining two successive half-word transfers. The HPI ready pin (HRDY_) allows insertion of host wait states and therefore affects data transfer speed. HRDY_ provides a convenient way to automatically adjust the host access rate to the rate of data delivery from the DMA auxiliary channel. When low, HRDY_ indicates that the HPI is ready for a transfer.

Since the HPI asserts (HRDY_) high to insert a delay during a transfer, HRDY_ is used to characterize data transfer speed through the HPI.

An exemption to the HPI transfers previously described are transactions to or from the Host Port Interface Address register (HPIA) or the Host Port Interface Control register (HPIC). Transfers to or from the HPIA and HPIC do not involve the HRDY_ signal. Data is immediately stored/read from these registers without involving the auxiliary DMA channel.

In the cases presented here, the TMS320C6201/6701 DSP is in the idle mode. In this case the DMA controller is dedicated to the HPI, since the only requests to the DMA are the ones from the HPI. This results in the fastest HPI access to memory.

Read without Auto-Increment

When the host performs a read without auto-increment from the HPI data register (HPID), the HPI sends the read request to the DMA auxiliary channel, and HRDY_ goes high. This event occurs on the first falling edge of HSTROBE_. HRDY_ remains high until the DMA auxiliary channel loads the requested data into the HPID. At the beginning of the second read access the data is already present in the HPID (DMA auxiliary channel performs word reads). Thus, the second half-word HPID read will never encounter a not ready condition and HRDY_ will remain low.

The timing diagram for a read without auto-increment is shown in Figure 1. The duration of the delay inserted by HRDY_ high is shown for a read from:

- Internal data memory
- Internal program memory
- SBSRAM (1/2 x)
- SBSRAM (1/x)
- SDRAM
- Internal Peripheral Bus

NOTE:
For clarity the only control signal presented is the internal control signal HSTROBE_.

Figure 1. Read without Auto-Increment Timing Diagram

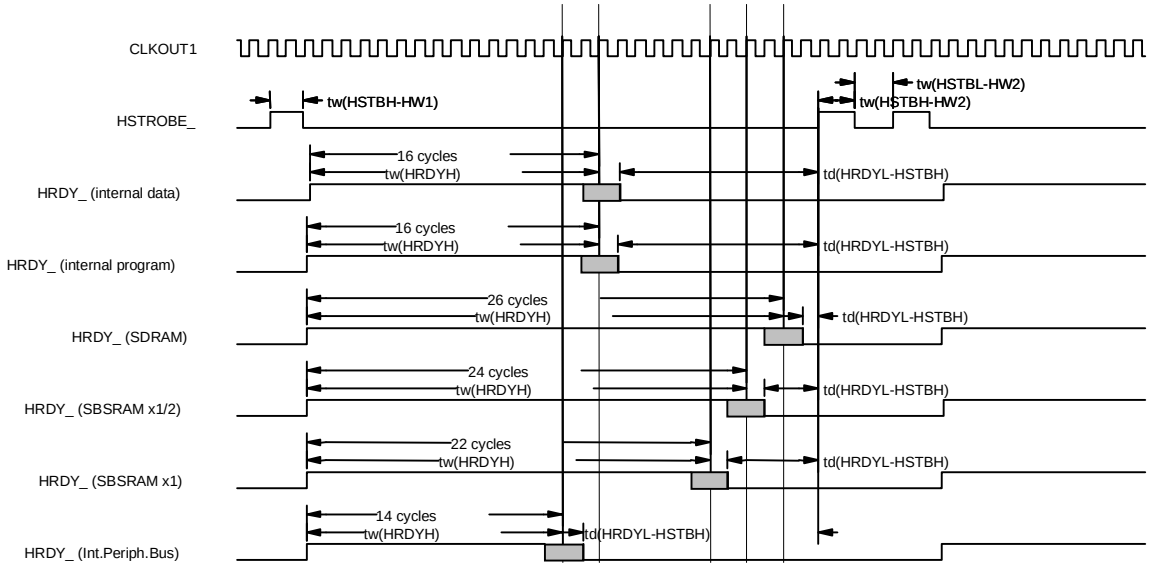


Table 1 shows the duration of HRDY_ high, $t_w(HRDYH)$, when the host CPU performs a read without auto-increment from TMS320C6201/6701 memory using the HPI.

Table 1. Duration of HRDY_ High without Auto-Increment

Read without Auto-increment	Duration of HRDY_ high (in clock cycles)
Internal Data Memory	15 – 17
Internal Program Memory	15 – 17
SDRAM	25 – 27
SDRAM (new page accessed)	37 – 39
SBSRAM (1/2 x)	23 – 25
SBSRAM (1/x)	21 – 23
Internal Peripheral Bus	13 – 15

The following formula calculates the time required to read one word of data without auto-increment:

$$t_{CYC}(READ) = t_W(HSTBH - HW1) + t_W(HRDYH) + \\ + t_D(HRDYL - HSTBH) + t_W(HSTBH - HW2) \\ + t_W(HSTBL - HW2).$$

Where:

$t_{CYC}(READ)$ = Number of clock cycles required for a word read without auto-increment

$t_W(HSTBH - HW1)$ = Number of clock cycles HSTROBE_ stays high before requesting the first half-word. The external host determines this value.

$t_W(HRDYH)$ = Number of clock cycles the HPI indicates busy status (HRDY_ stays high)

$t_D(HRDYL - HSTBH)$ = Number of clock cycles between the time HRDY_ goes low and the time HSTROBE_ goes high. The external host determines this value.

$t_W(HSTBH - HW2)$ = Number of clock cycles HSTROBE_ stays high before requesting the second half-word transfer. The external host determines this value.

$t_W(HSTBL - HW2)$ = Number of clock cycles HSTROBE_ is low during the transfer of the second half-word. The external host determines this value.

Table 2 lists the maximum HPI transfer rates possible for a read without auto-increment.

NOTE:
The transfer rates shown in Table 2 assume that the host can drive the interface at the rates shown.

Table 2. Maximum HPI Transfer Rates for a Read without Auto-Increment

Read without Auto-increment	Transfer Rate (Mbytes/s)		Transfer Rate (MBits/s)	
	'C6201 200MHz	'C6701 167MHz	'C6201 200MHz	'C6701 167MHz
Internal Data Memory	34.7	28.97	277.6	231.79
Internal Program Memory	34.7	28.97	277.6	231.79
SDRAM	24.2	20.2	193.6	161.65
SBSRAM (1/2 x)	25.8	21.54	206.4	172.34
SBSRAM (1/x)	27.5	22.96	220	183.7
Internal Peripheral Bus	38.0	31.73	304	253.84

Read with Auto-Increment

The auto-increment feature provides efficient sequential host accesses. For both HPID read and write accesses, this feature eliminates the need for the host to load incremented the address into the HPIA (HPI address register). For read accesses, the data pointed to by the next address is fetched immediately upon completion of the current read. Therefore, after the second half-word transfer of the current read (with the second rising edge of HSTROBE_), HRDY_ goes high. When high HRDY_ indicates that the HPI is busy pre-fetching data.

Therefore, on the first half-word transfer when HCS_ becomes valid, HSTROBE_ is high and HRDY_ indicates that HPI is busy completing the internal portion of a previous HPI request. There are two options, assuming that HCS_ stays low:

- The host can wait for HRDY_ to go low then assert HSTROBE_ low (see Figure 2).
- The host can assert HSTROBE_ low before HRDY_ goes low (see Figure 3).

The timing diagram for a read with auto-increment is shown in Figure 2. The duration of the HRDY_ high is shown for a read from:

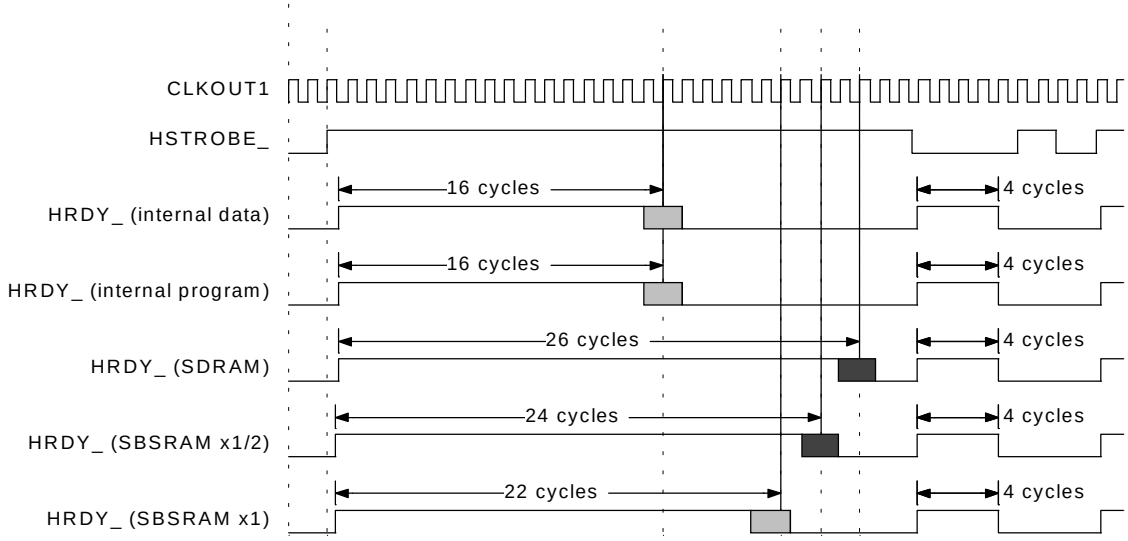
- Internal data memory
- Internal program memory

- SBSRAM (1/2 x)
- SBSRAM (1/x)
- SDRAM

NOTES:

1) For clarity, only the HSTROBE_ internal control signal is shown.
2) In Figure 2 HSTROBE_ goes low (falling edge) after HRDY_ goes low.

Figure 2. Read with Auto-Increment Timing Diagram



The fastest auto-incremented read shown in Figure 2 is performed from internal memory of the TMS320C6201/6701. The HPI is busy for 15-17 cycles after the rising edge of HSTROBE_, during a second half-word transfer. The host waits for HRDY_ to go low and then asserts HSTROBE_ low. It requires only 4 cycles to retrieve the previously fetched data from the HPID register (which is initiated by falling edge of HSTROBE_).

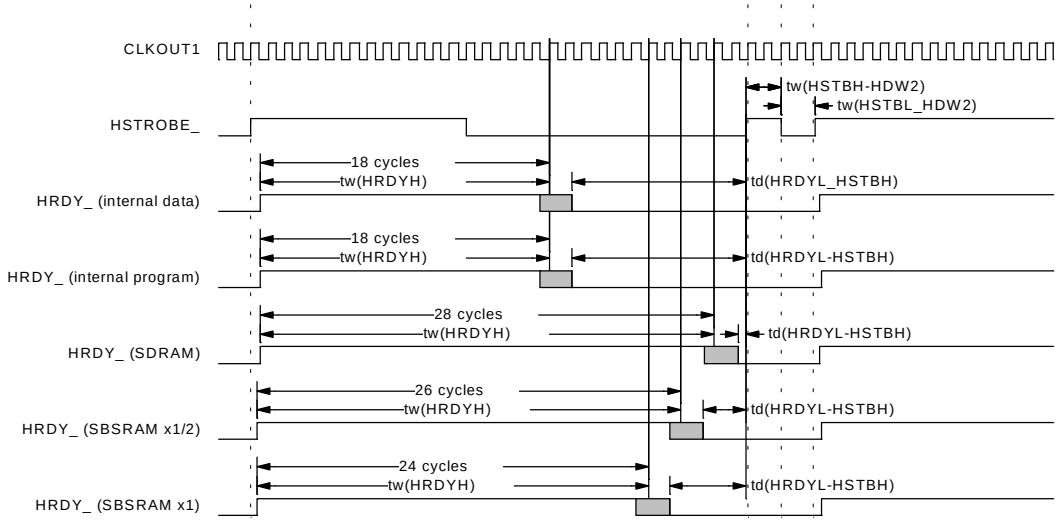
The timing diagram for a read with auto-increment is shown in Figure 3. The duration of HRDY_ high is shown for a read from:

- Internal data memory
- Internal program memory
- SBSRAM (1/2 x)
- SBSRAM (1/x)
- SDRAM

NOTES:

1) For clarity, only the HSTROBE_ internal control signal is shown.
2) Signal HSTROBE_ goes low (falling edge) before HRDY_ goes low.

Figure 3. Read with Auto-Increment Timing Diagram



The number of clock cycles HRDY_ stays high, $t_w(HRDYH)$, when the host CPU performs a read with auto-increment from TMS320C6201/6701 memory using the HPI is listed in Table 3.

Table 3. Duration of HRDY_ High During a Read with Auto-Increment

Read with Auto-increment	Duration of HRDY_ High (in clock cycles)
Internal Data Memory	17 – 19
Internal Program Memory	17 – 19
SDRAM	27 – 29
SBSRAM (1/2 x)	25 – 27
SBSRAM (1/x)	23 – 25

In this case (Figure 3), the host asserts HSTROBE_ low before HRDY_ goes low. When a read is performed from the internal memory, the HPI is busy for 17-19 cycles, after the rising edge of HSTROBE_ during a second half-word transfer. Therefore, a read in case 2 (Figure 3) is faster than a read in case 1 (Figure 2).

The following formula can be used to calculate the time required to complete an auto-incremented read (see Figure 3):

$$t_{CYC}(AC\_READ) = t_W(HRDYH) + t_D(HRDYL - HSTBH) + t_W(HSTBH - HW2) + t_W(HSTBL - HW2).$$

Where:

$t_{CYC}(AC\_READ)$ = Number of clock cycles required for an auto-incremented word read

$t_W(HRDYH)$ = Number of clock cycles the HPI indicates busy status (HRDY_ stays high)

$t_D(HRDYL - HSTBH)$ = Number of clock cycles between the time HRDY_ goes low and the time HSTROBE_ goes high. The external host determines this value.

$t_W(HSTBH - HDW2)$ = Number of clock cycles HSTROBE_ stays high before requesting the second half-word transfer. The external host determines this value.

$t_W(HSTBL - HDW2)$ = Number of clock cycles that HSTROBE_ is low while a transfer of the second half-word takes place. The external host determines this value.

The maximum HPI transfer rates possible for a read with auto-increment are given in Table 4.

NOTE:
The transfer rates shown Table 4 assumes the host can drive the interface at the rates shown.

Table 4. Maximum HPI Transfer Rate for an Auto-Incremented Read

Read without Auto-increment	Transfer Rate (Mbytes/s)		Transfer Rate (MBits/s)	
	'C6201 200MHz	'C6701 167MHz	'C6201 200MHz	'C6701 167MHz
Internal Data Memory	34.7	28.97	277.6	231.79
Internal Program Memory	34.7	28.97	277.6	231.79
SDRAM	24.2	20.2	193.6	161.65
SBSRAM (1/2 x)	25.8	21.54	206.4	172.34
SBSRAM (1/x)	27.5	22.96	220	183.7

Write with and without Auto-Increment

During an HPID write access two half-word portions of the HPID are transferred from the host. At the end of this write access HRDY_ goes high on with the second rising edge of HSTROBE_ and HPID is transferred as a 32-bit word to the address specified by HPIA. A write and an auto-incremented write require the same number of cycles.

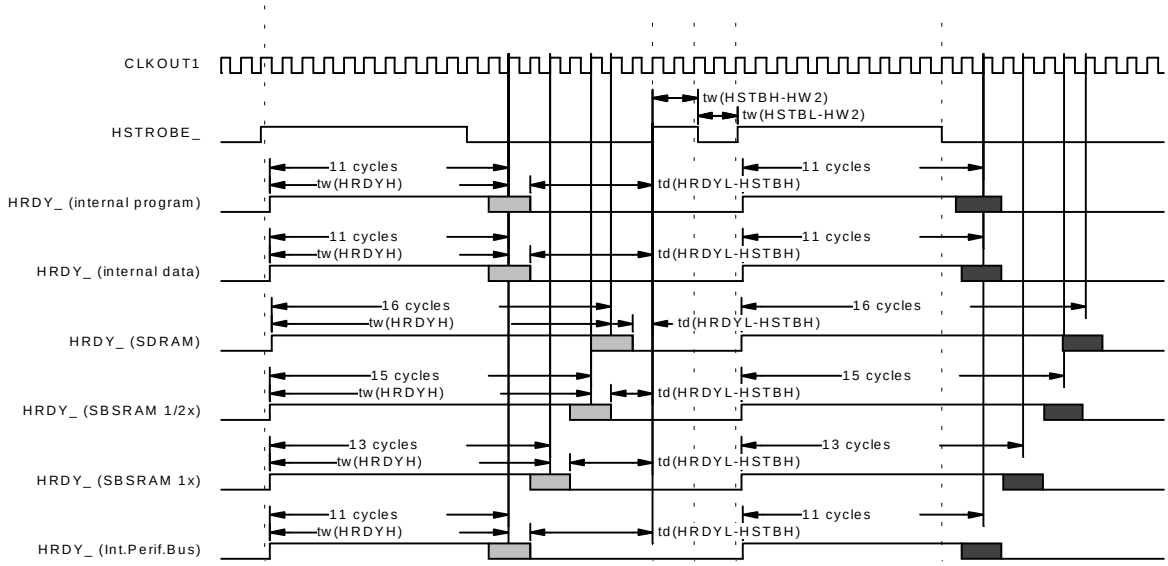
The timing diagram for a write transfer is shown in Figure 4. The duration of the HRDY_ high is shown for a write to:

- Internal data memory
- Internal program memory
- SBSRAM
- SBSRAM (1/2 x)
- SDRAM
- Internal Peripheral Bus

NOTES:

1) For clarity only the HSTROBE_ internal control signal is shown.
2) Signal HSTROBE_ goes low (falling edge) before HRDY_ goes low.

Figure 4. Write Timing Diagram



The number of clock cycles HRDY_ stays high, t_w*(HRDYH),* when the host CPU performs a write to TMS320C6201/6701 DSP memory using the HPI listed in Table 5.

Table 5. Duration of HRDY_ High During a Write

Write	Duration of HRDY_ High (in clock cycles)
Internal Data Memory	10 – 12
Internal Program Memory	10 – 12
SDRAM	15 – 17
SDRAM (new page accessed)	27 – 29
SBSRAM (1/2 x)	14 – 16
SBSRAM (1/x)	12 – 14
Internal Peripheral Bus	10 – 12

The following formula calculates the time required to write a word of data:

$$t_{CYC}(WRITE) = t_W(HRDYH) + t_D(HRDYL - HSTBH) + t_W(HSTBH - HW2) + t_W(HSTBL - HW2).$$

Where:

$t_{CYC}(WRITE)$ = Number of clock cycles required to complete a word write

$t_W(HRDYH)$ = Number of clock cycles the HPI indicates busy status (HRDY_ stays high)

$t_D(HRDYL - HSTBH)$ = Number of clock cycles between the time HRDY_ goes low and the time HSTROBE_ goes high. The external host determines this value.

$t_W(HSTBH - HW2)$ = Number of clock cycles HSTROBE_ stays high before requesting the second half word transfer. The external host determines this value.

$t_W(HSTBL - HW2)$ = Number of clock cycles that HSTROBE_ is low during the transfer of the second half-word. The external host determines this value.

The maximum HPI transfer rates possible for a write are listed in Table 6.

NOTE:
The transfer rates shown in Table 6 assume the host can drive the interface at the rates listed.

Table 6. Maximum HPI Transfer Rates for a Write

Read without Auto-increment	Transfer Rate (Mbytes/s)		Transfer Rate (MBits/s)	
	‘C6201 200MHz	‘C6701 167MHz	‘C6201 200MHz	‘C6701 167MHz
Internal Data Memory	50	41.75	400	334
Internal Program Memory	50	41.75	400	334
SDRAM	38	31.73	304	253.84
SBSRAM (1/2 x)	40	33.4	320	267.2
SBSRAM (1/x)	44.4	37.07	355.2	296.59
Internal Peripheral Bus	50	41.75	400	334