Texas Instruments

Application Report
SPRA632A - August 2001

TMS320C6000 DSP Expansion Bus: Multiple DSP Connection Using Asynchronous Host Mode

Kyle Castille *DSP Applications*

ABSTRACT

You can easily interface multiple TMS320C6000™ (C6000™) DSP devices to each other by way of a direct connection of the expansion bus (XBus). Each device's XBus I/O port masters the other's XBus host port configured for asynchronous slave mode.

This application report highlights the XBus asynchronous host mode and XBus asynchronous I/O mode and explains the interface used to connect two C6000s together using these two interfaces.

This application report contains:

- Full example of a two-DSP interface via the asynchronous I/O mode and the asynchronous host mode at the XBus
- Timing diagrams illustrating the interface functionality

Contents

List of Figures

List of Tables

1 Interface Overview

The XBus asynchronous I/O port provides a flexible means to interface to asynchronous peripheral busses, while the XBus asynchronous host port provides a flexible means for asynchronous hosts to directly access the memory map of the C6000. The combination of these two interfaces externally provides a simple solution for interfacing two, or more, C6000s together using the XBus. This scheme offers the advantage of isolating the XBus accesses from the faster high-speed memory accesses of the external memory interface (EMIF).

Figure 1 shows a high-level block diagram of the Async/Async interface between two C6000s. From a high level, the interface is symmetric, allowing either device to master the bus (using the Async I/O port) and access the slave device (using the Async Host port). Since the data busses are shared, arbitration takes place to determine which device is the master at any given time.

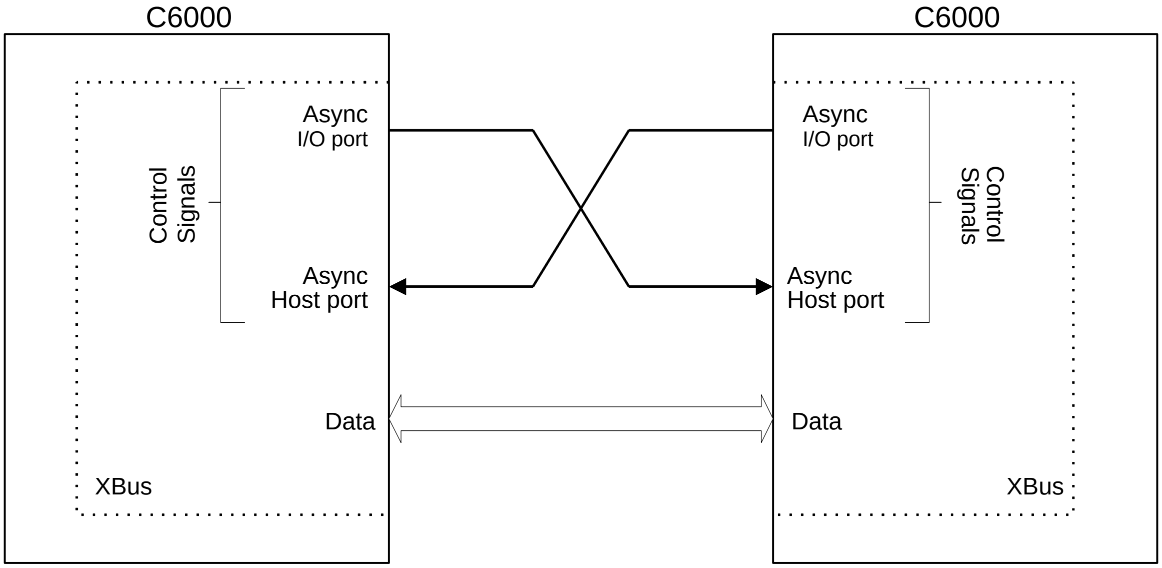

Figure 1. Figure 1. High–Level Block Diagram

2 Interface Details

This section discusses how to interface two C6000s together using the expansion bus with a minimal amount of external logic. The asynchronous I/O port of each DSP interfaces to the asynchronous host port of the other DSP, providing a bidirectional communication path. This requires only a small amount of logic, which can easily fit in a PAL or other type of programmable logic device.

Figure 2 shows the interface between the I/O port and host port of two C6000s. Note that the internal bus arbiter of DSP1 is enabled (XARB=1) and the internal bus arbiter of DSP2 is disabled (XARB=0). A direct connection between the Hold/HoldA signals is possible since DSP2 requests the bus using its Hold output. DSP1 responds to this Hold input by asserting its HoldA output, indicating that DSP2 is free to access the bus.

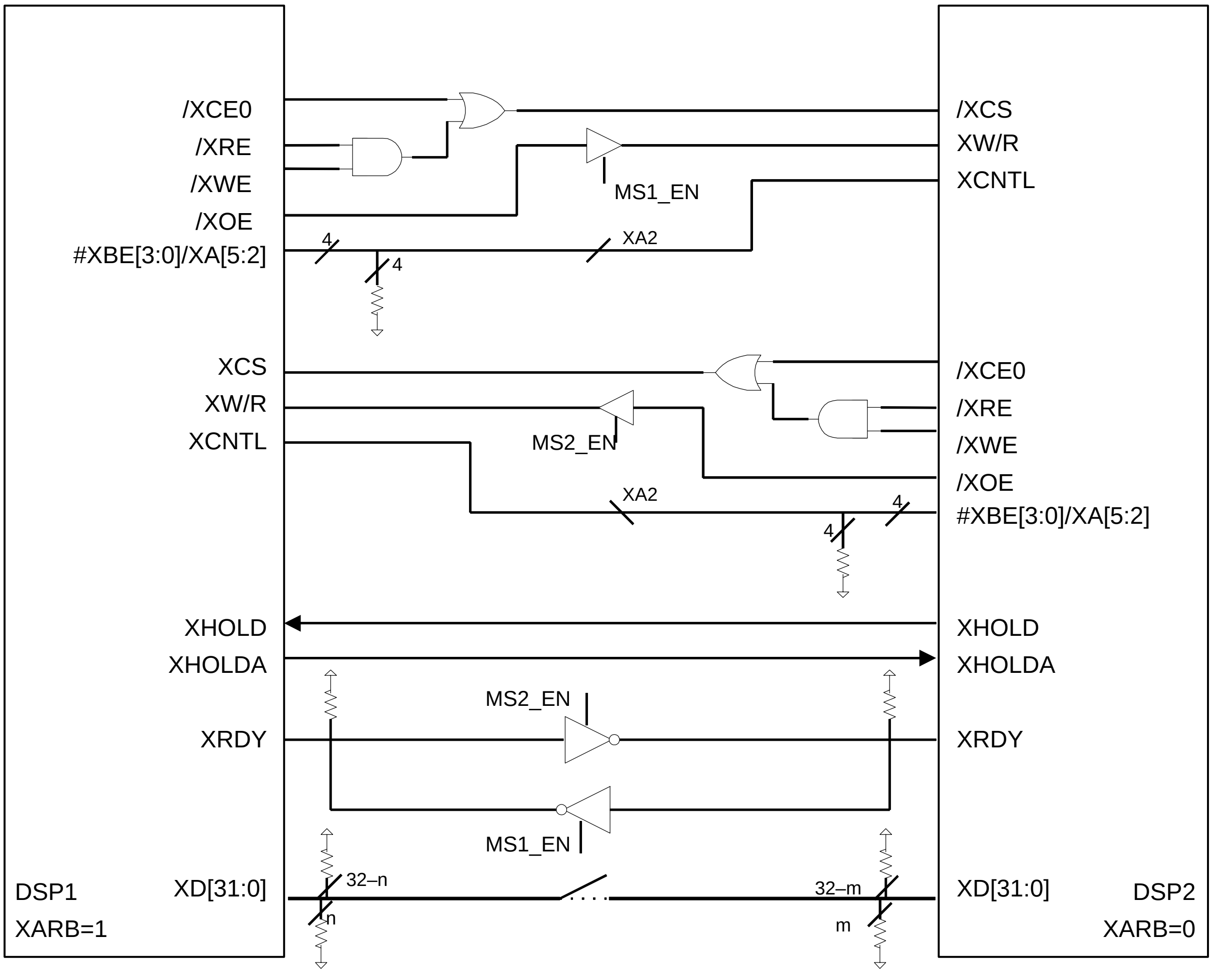

Figure 2. Async I/O to Async Host Interface Schematic

Table 1. Signal List

Master Signal	Slave Signal	Description
/XCE0 /XRE /XWE	/XCS	/XCE0, /XRE, and /XWE generate the /XCS signal of the slave DSP.
/XOE	XW/R	/XOE generates the XW/R signal. With the logic shown, XW/R should be set as an active low read signal at reset via pull–down resistor on XD[12].
XA2	XCNTL	Used to control whether XBISA or XBD is selected on the slave DSP.
XRDY	XRDY	This master input/slave output allows the slave DSP to hold off the master until it is ready to accept (write) or present (read) data.
XD [31:0]	XD [31:0]	The XD lines are separated at Reset in order to configure each of the DSPs separately via pull–up/down resistors. A general purpose output of either DSP can be used to close a bus switch after exiting reset.

Note that the /XCS line is generated based upon /XCE0, /XRE, and /XWE. This is done to delay the /XCS signal so that adequate input setup time is provided for XW/R and XCNTL relative to the /XCS falling edge. This is illustrated in the Timing Verification section. If multiple DSPs were used, address decode could be included in the /XCS generation so that only the target DSP is selected based on some predetermined address range.

3 Preventing Bus Contention

Table 2 and Figure 3 show the details of the arbitration for a two DSP system. The timing diagram shows the key signals that potentially have contention problems, and which have been buffered in the schematic shown in Figure 2. Figure 3 shows the output signals directly from each of the DSPs, not the buffered version.

Table 2. Arbitration Between Two TMS320C6000 Devices

Signal	XARB=1 (DSP1)	XARB=0 (DSP2)
XHOLD	Input	Output
XHOLDA	Output	Input

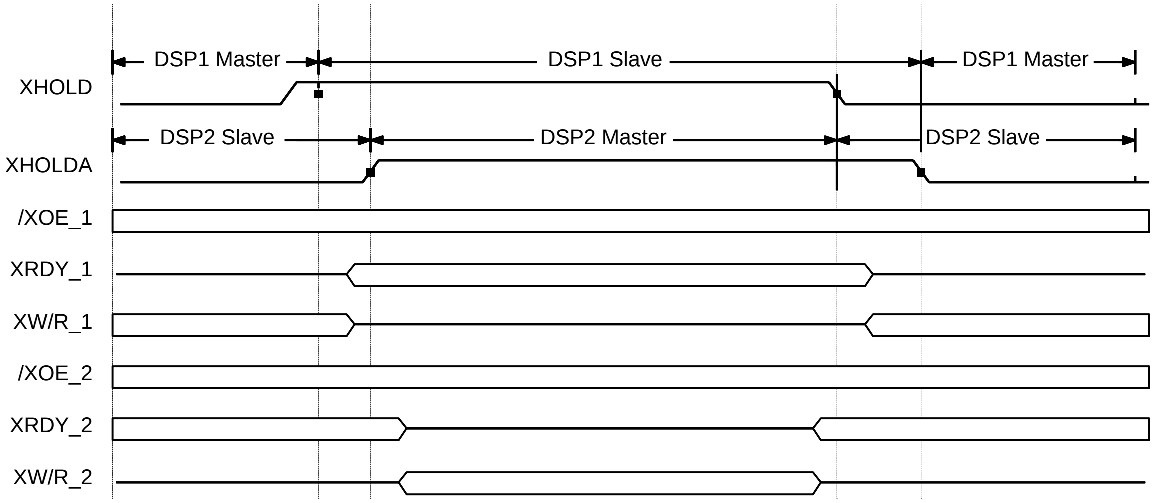

Figure 3. Arbitration Timing

3.1 /XOE vs XW/R

In this interface, buffers are used to prevent contention between /XOE and XW/R. As Figure 3 shows, /XOE (and other I/O port control signals) is always driven, regardless of whether the given DSP owns the bus. In addition, XW/R is driven when a DSP is bus master. So, when either DSP gets the bus to perform a transaction using the I/O port, the host port signals are also driven. In order to prevent the master XW/R contending with the /XOE signal of the slave (always driven), a buffer is inserted to tri-state the /XOE output signal of each DSP when 1) It is not the master and 2) It is not performing transactions. The second requirement is included to avoid the situation where both DSPs are slaves on the bus. As seen in Figure 3, this condition occurs during the transition from Master to Slave and from Slave to Master for either DSP.

3.2 XRDY vs XRDY

XRDY is always driven when a DSP is in slave mode. Although this interface is designed such that one DSP is the master and one DSP is the slave, there is a short interval of time during arbitration in which both DSPs are a slave. During this time, it is necessary to turn off both XRDY outputs so that contention does not occur, as shown in Figure 3. A pull-up resistor is included on each XRDY signal to ensure that XRDY stays in a valid state in case other accesses are going on.

3.3 Logic Equations for Output Buffers

As seen in Figure 2, buffers are used to isolate signals that can potentially cause bus contention. The buffers are controlled with MS1_EN and MS2_EN. MS1_EN should be driven active low when DSP1 is the bus master and MS2_EN should be driven active low when DSP2 is the bus master. Both MS1_EN and MS2_EN should be inactive when neither DSP is the master (during arbitration transitions). The following equations describe the enable control for these buffers:

MS1_EN = NOT (/CE0_1) * NOT (XHOLDA)
MS2_EN = NOT(/CE0_2) * XHOLDA

4 Hardware Configuration at Reset

The interface previously described requires several registers and hardware parameters to be set correctly, including:

Hardware configuration using pull-up/down resistors on the XBus Data (XD) lines

- The Host port must be configured to operate in Asynchronous host mode.
- The XBus CE space used for this interface must be configured as asynchronous.
- The XW/R signal should be configured as an active-low read enable.

Since the XARB mode must be set differently for each of the DSPs using pull-up/down resistors on the same data line, the XD lines must be physically separated at reset. In Figure 2 this is shown as a switch and can be implemented with an SN74CBTLV3245 bus switch or similar device. Only the XD lines that are configured differently must be separated by a bus switch. Since the XBus configuration is set at reset, when the device comes out of reset, one of the DSPs can close the switch using an unused general-purpose output (such as timer output pins, DMAC outputs, serial port outputs, etc…)

Another possible solution is to configure one of the DSPs with pull-up/down resistors, and release this DSP from reset first. The DSP that is released from reset can then drive the appropriate value on the data bus for the second DSP, which is released from reset while the data lines are driven to the desired configuration value. This is a more complicated solution, since the reset timing must be accurately controlled relative to the data being driven by the first DSP.

Note that an emulation reset will cause the device configuration information to be latched from the XBus data lines. Care should be taken to ensure that the data lines are configured correctly during an emulation reset. The bus switch solution meets this requirement.

This example setup assumes that XCE0 is used when one C6000 masters the other. This implies that the XCE0 space should be set to operate in asynchronous mode. Note that all the XCE spaces are set to operate in asynchronous mode as well, since a valid combination must be used. Figure 4 illustrates the values used in this example. Cells shown in gray may be set differently depending on your system (For example, LEND = 1 specifies Little Endian. This should be cleared to 0 if Big Endian is desired). Cells in white should be set as specified here for the interface described here to work correctly.

31	30 28	27	26 24	23	22 20	19	18 16
rsv	MTYPE XCE3	rsv	MTYPE XCE2	rsv	MTYPE XCE1	rsv	MTYPE XCE0
0	010	0	010	0	010	0	010

15 14	13	12	11	10	9	8	7 5	4 0
reserved	BLPOL	RWPOL	HMOD	XARB	FMOD	LEND	reserved	BootMode
00	0	0	0	0/1	0	1	000	00101

Figure 4. XD Pull-Up/Pull-Down Configuration Resistors

MTYPE = 010 selects XCE0 as an Asynchronous interface

RWPOL = 0 selects read as active low for XR/#W

HMOD = 0 selects asynchronous host mode

XARB=1 for DSP1 enables internal arbiter. XARB = 0 for DSP2 disables internal arbiter.

FMOD = 0 allows /XOE to be used in all XCE spaces.

5 Software Configuration

The cycle timings of the Asynchronous memory cycles must be set appropriately in the correct XCE space control register. The values calculated for the Asynchronous cycle timings are based on the TMS320C6000 and will be proven in the next section.

31 28	27 22	21 20	19 16
WRITE SETUP	WRITE STROBE	WRITE HOLD	READ SETUP
0010	000110	11	0010

15 14	13 8	7 4	3 2	1 0
reserved	READ STROBE*	MTYPE	reserved	RDHLD
00	000110	0010	00	11

* READ STROBE = 000111 for C6204.

Figure 5. Expansion Bus XCE(0/1/2/3) Space Control Register Diagram

6 Timing Verification

The following timing diagrams illustrate a read and a write cycle and are based on timing information for the following devices:

- TMS320C6000 (C6202(B), C6204)
- Lattice GAL22LV10

Timing analysis should always be verified in the context of the specific system environment and using the most current datasheets. This example does not take into account all system constraints. Instead, the board designer can use this example as a framework for independent verification.

The following notation has been adopted:

M -> Master

S -> Slave

I -> Input

O -> Output

Therefore, MO indicates Master Output, SI indicates Slave Input, etc. The following diagrams represent the read and write timing characteristics for the C6000. Since the timing parameters for the C6000 devices are similar, these two examples can be used as guides for all devices.

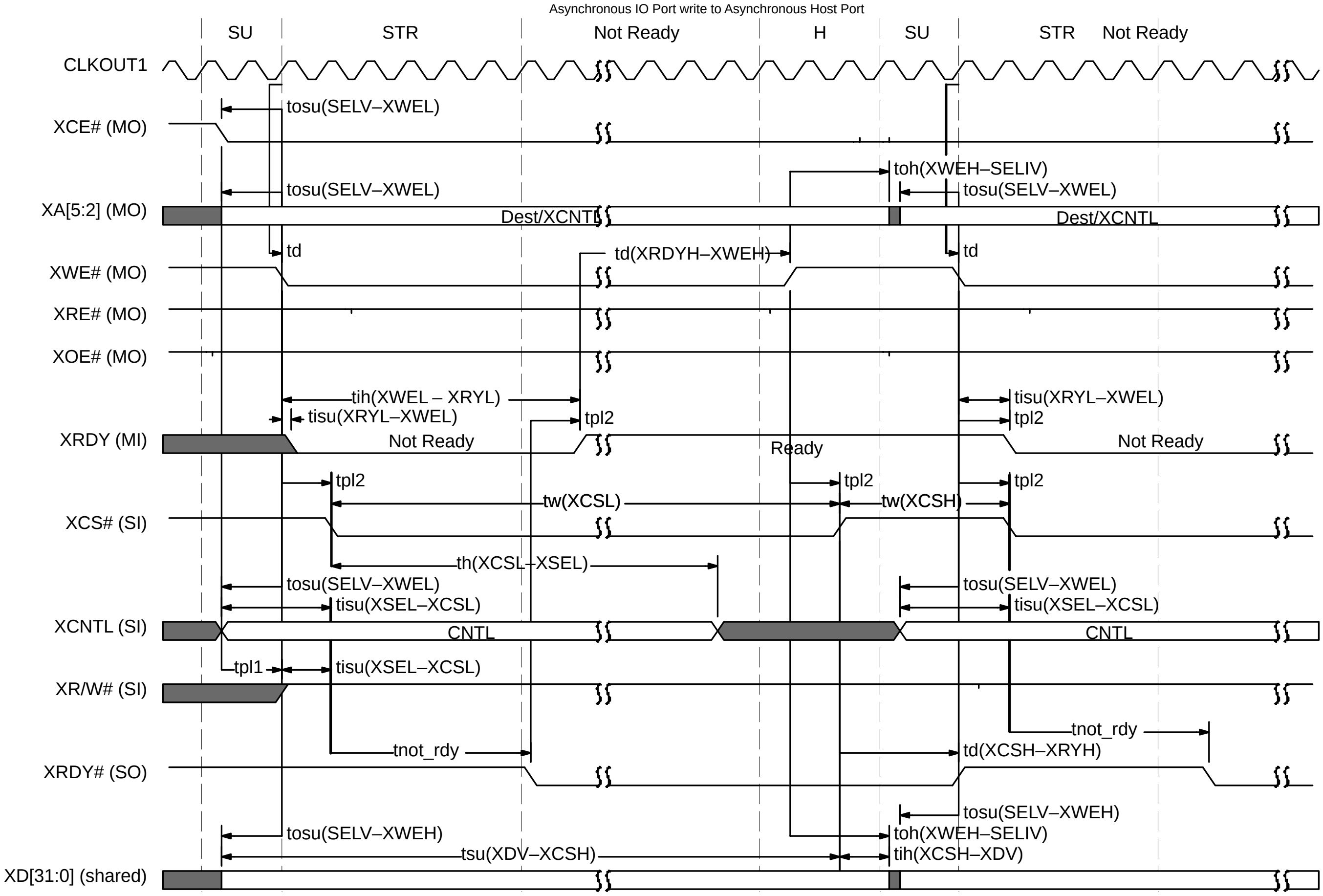

Figure 6. Burst Write Example

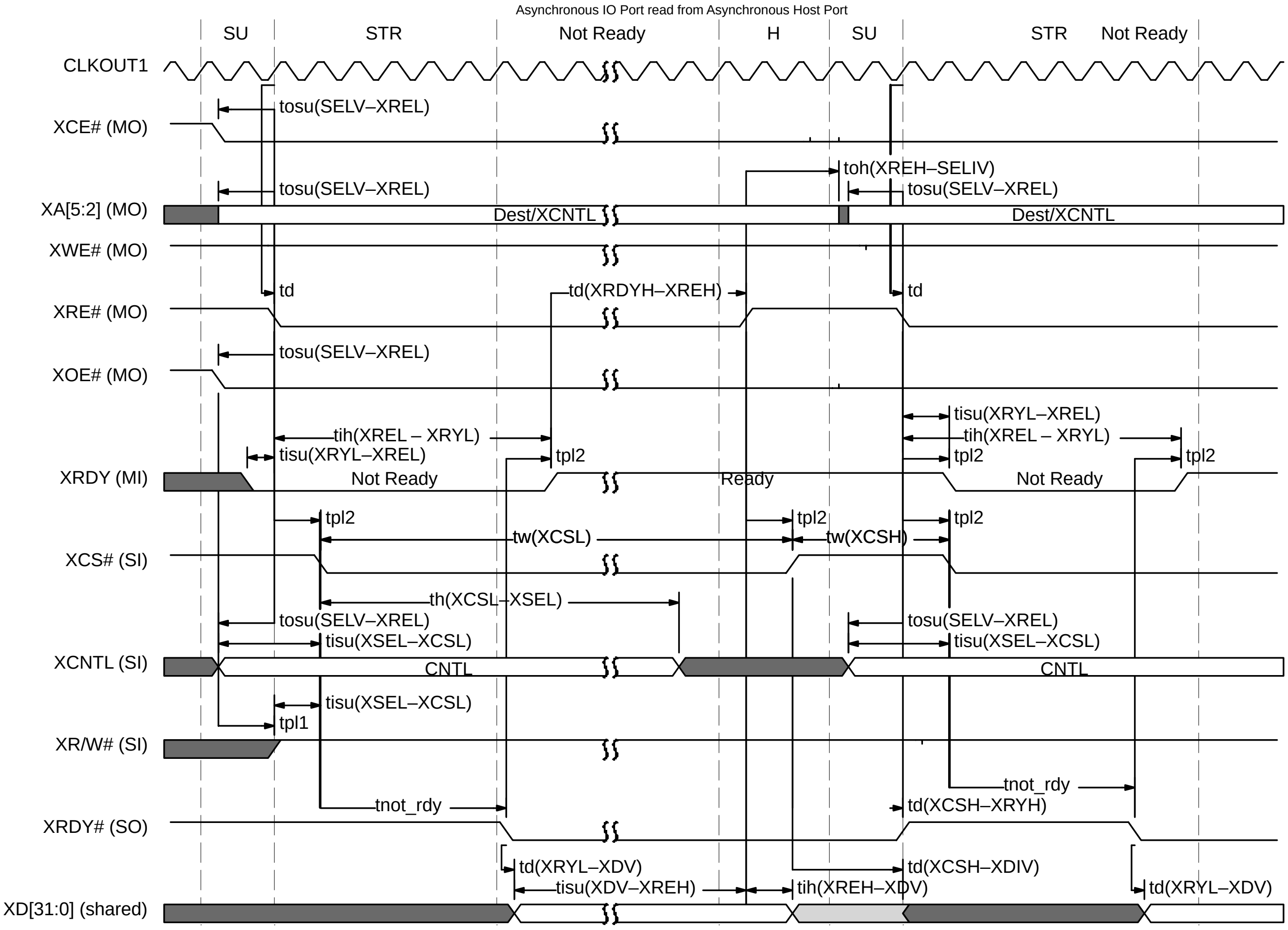

Figure 7. Burst Read Example

Table 3 through Table 10 show the calculations required to prove that the previous interface meets the timing requirements for both DSPs. The following notational convention is used:

- P = CPU Clock Period = 4 ns
- WSU = Write Setup = 2 * Tcyc
- WST = Write Strobe = 6 * Tcyc
- WH = Write Hold = 3 * Tcyc
- RSU = Read Setup = 2 * Tcyc
- RST = Read Strobe = 6‡ * Tcyc
- RH = Read Hold = 3 * Tcyc

- Tpl1 = Logic Propagation #1 (Buffer turn on time)
  - Tpl1max = 6 ns (This is met with Lattice part number GAL22LV10)
  - Tpl1min = 1 ns (This is met with Lattice part number GAL22LV10)
- Tpl2 = Logic Propagation #2 (Input to output valid time)
  - Tpl2max= 5 ns (This is met with Lattice part number GAL22LV10)

  - Tpl2min = 1 ns (This is met with Lattice part number GAL22LV10)
- Tplskew = Skew between Tpl1 and Tpl2, assumed to be Tpl1max – Tpl2max
  - Tplskew ~ 1.0 ns

‡ RST = 7 * Tcyc for C6204.

Table 3. Timing Requirement for Async Host Port (C6203, C6203B)

Host Port Timing Requirement (A)	Value (A)	System Timing Characteristics (B)	Value (B)	Criteria	Pass?

Tw(XCSL)	4*P = 16 ns	Tw(XWEL)
		OR
		Tw(XREL)
			WST * P = 24 ns
			OR
			RST * P = 24 ns
				B >= A	Yes

Tw(XCSH)	4*P = 16 ns	Tw(XWEH)
		OR
		Tw(XREH)
			WH * P + WSU *P = 20 ns
			OR
			RH * P + RSU *P = 20 ns
				B >= A	Yes

Tisu(XSEL–XCSL)	1 ns	Tosu(SELV–XWEL) - Tplskew OR Tosu(SELV–XREL) - Tplskew	(WSU*P – 2) - 1 = 5 ns OR (RSU*P – 2) - 1 = 5 ns	B >= A	Yes
Tisu(XDV–XCSH)	1 ns	Tosu(SELV–XWEL) + Tw(XWEL) + tpl2min	(WSU*P – 2) + WST * P + 1 = 31 ns	B >= A	Yes
Tih(XCSH–XDIV)	3 ns	Toh(XWEH–SELIV) - tpl2max)	(WH * P – 2) - 5 = 5 ns	B >= A	Yes

Table 4. Timing Requirements for Async I/O Port (C6203, C6203B)

I/O Port Timing Requirement (A)	Value (A)	System Timing Characteristic (B)	Value (B)	Criteria	Pass?

Tisu(XRYL-XWEL)
OR
Tisu(XRYL-XREL)
	-((WST-3)*P-6) = -2 ns
	OR
	-((RST-3)*P-6) = -6 ns
		(Tw(XWEH) - (2* tpl2max + Td(XCSH-XRYH))
		OR
		(Tw(XREH) - (2* tpl2max + Td(XCSH-XRYH))
			(WH * P + WSU * P) - (10 + 12) = -2 ns
			OR
			(RH * P + RSU * P) - (10 + 12) = -2 ns
				B >= A	Yes

Tisu(XDV-XREH)	4.5 ns	Tpl2min + td(XRDYH-XREH)min - td(XRYL - XDV)	1 + 3P – 1 = 12 ns	B >= A	Yes

Table 5. Timing Requirements for Async Host Port (C6203)

Host Port Timing Requirement (A)	Value (A)	System Timing Characteristics (B)	Value (B)	Criteria	Pass?

Tw(XCSL)	4*P = 16 ns	Tw(XWEL)
		OR
		Tw(XREL)
			WST * P = 24 ns
			OR
			RST * P = 24 ns
				B >= A	Yes

Tw(XCSH)	4*P = 16 ns	Tw(XWEH)
		OR
		Tw(XREH)
			WH * P + WSU *P = 20 ns
			OR
			RH * P + RSU *P = 20 ns
				B >= A	Yes

Tisu(XSEL-XCSL)	1 ns	Tosu(SELV-XWEL) - Tplskew OR Tosu(SELV-XREL) - Tplskew	(WSU*P – 3) - 1 = 4 ns OR (RSU*P – 2) - 1 = 5 ns	B >= A	Yes
Tisu(XDV-XCSH)	1 ns	Tosu(SELV-XWEL) + Tw(XWEL) + tpl2min	(WSU*P – 3) + WST * P + 1 = 30 ns	B >= A	Yes
Tih(XCSH-XDIV)	3 ns	Toh(XWEH-SELIV) - tpl2max)	(WH * P – 2) - 5 = 5 ns	B >= A	Yes

Table 6. Timing Requirements for Async I/O Port (C6203)

I/O Port Timing Requirement (A)	Value (A)	System Timing Characteristic (B)	Value (B)	Criteria	Pass?

Tisu(XRYL-XWEL)
OR
Tisu(XRYL-XREL)
	-((WST-3)*P-6) = -2 ns
	OR
	-((RST-3)*P-6) = -6 ns
		(Tw(XWEH) - (2* tpl2max + Td(XCSH-XRYH))
		OR
		(Tw(XREH) - (2* tpl2max + Td(XCSH-XRYH))
			(WH * P + WSU * P) - (10 + 12) = -2 ns
			OR
			(RH * P + RSU * P) - (10 + 12) = -2 ns
				B >= A	Yes

Tisu(XDV-XREH)	4.5 ns	Tpl2min + td(XRDYH-XREH)min - td(XRYL-XDV)	1 + 3P - 1 = 12 ns	B >= A	Yes

Table 7. Timing Requirements for Async Host Port (C6204)

Host Port Timing Requirement (A)	Value (A)	System Timing Characteristics (B)	Value (B)	Criteria	Pass?

Tw(XCSL)	4*P = 16 ns	Tw(XWEL)
		OR
		Tw(XREL)
			WST * P = 24 ns
			OR
			RST * P = 28 ns
				B >= A	Yes

Tw(XCSH)	4*P = 16 ns	Tw(XWEH)
		OR
		Tw(XREH)
			WH * P + WSU *P = 20 ns
			OR
			RH * P + RSU *P = 20 ns
				B >= A	Yes

Tisu(XSEL-XCSL)	1 ns	Tosu(SELV - XWEL) - Tplskew OR Tosu(SELV - XREL) - Tplskew	(WSU*P - 2) - 1 = 5 ns OR (RSU*P - 2) - 1 = 5 ns	B >= A	Yes
Tisu(XDV-XCSH)	1 ns	Tosu(SELV - XWEL) + Tw(XWEL) + tpl2min	(WSU*P - 2) + WST * P + 1 = 31 ns	B >= A	Yes
Tih(XCSH-XDIV)	3 ns	Toh(XWEH - SELIV) - tpl2max)	(WH * P - 2) - 5 = 5 ns	B >= A	Yes

Table 8. Timing Requirements for Async I/O Port (C6204)

I/O Port Timing Requirement (A)	Value (A)	System Timing Characteristic (B)	Value (B)	Criteria	Pass?

Tisu(XRYL-XWEL)
OR
Tisu(XRYL-XREL)
	-((WST-3)*P-6)
	= -2 ns
	OR
	-((RST-3)*P-6)
	= -6 ns
		(Tw(XWEH)
		- (2* tpl2max
		+ Td(XCSH-XRYH))
		OR
		(Tw(XREH)
		- (2* tpl2max
		+ Td(XCSH-XRYH))
			(WH * P + WSU * P)
			- (10 + 12) = -2 ns
			OR
			(RH * P + RSU * P)
			- (10 + 12) = -2 ns
				B >= A	Yes

Tisu(XDV-XREH)	8.5 ns	Tpl2min + td(XRDYH-XREH)min - td(XRYL-XDV)	1 + 3P - 1 = 12 ns	B >= A	Yes

IMPORTANT NOTICE

Texas Instruments and its subsidiaries (TI) reserve the right to make changes to their products or to discontinue any product or service without notice, and advise customers to obtain the latest version of relevant information to verify, before placing orders, that information being relied on is current and complete. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgment, including those pertaining to warranty, patent infringement, and limitation of liability.

TI warrants performance of its products 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.

Customers are responsible for their applications using TI components.

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

TI assumes no liability for applications assistance or customer product design. TI does not 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 products or services might be or are used. TI's publication of information regarding any third party's products or services does not constitute TI's approval, license, warranty or endorsement thereof.

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. Representation or reproduction of this information with alteration voids all warranties provided for an associated TI product or service, is an unfair and deceptive business practice, and TI is not responsible nor liable for any such use.

Resale of TI's 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, is an unfair and deceptive business practice, and TI is not responsible nor liable for any such use.

Also see: Standard Terms and Conditions of Sale for Semiconductor Products. www.ti.com/sc/docs/stdterms.htm

Mailing Address:

Texas Instruments
Post Office Box 655303
Dallas, Texas 75265