Using TI FIFOs to Interface High-Speed Data Converters With TI TMS320™ DSPs

Robert Finger — *Automotive and Military Products*

ABSTRACT

Most high-speed data converters cannot be connected directly to a digital signal processor (DSP). The required transfer rates would tie up most of the DSP's I/O bandwidth. A FIFO is an appropriate solution for this problem because it can buffer a large block of data, and the DSP can read data from the FIFO in a burst mode. This is much more efficient compared to single reads for every sampled value.

The new generation of TI FIFOs makes the connection between a DSP and a data converter very easy because there is no need for additional glue or control logic.

This application report shows how to use two SN74V2x5 FIFOs as an interface between TMS320C6201 and TMS320C6203 DSPs and a THS1031 analog-to-digital converter or THS5661A digital-to-analog converter. This method can also be used for other data converters.

Contents

Figures

Tables

1 Introduction

In a digital signal-processor (DSP) system, there is often the need for high-speed data acquisition. Connecting fast data converters to a DSP can be a demanding task. For example, if a 200-MHz DSP must service a 50-MHz analog-to-digital converter (ADC), the DSP must read one value every four clock cycles. This high data rate cannot be achieved if interrupts are used to read data from the converter. Interrupt latencies prevent the DSP from reacting fast enough. Only a direct memory access (DMA) controller can handle such data rates. But a fast data converter, connected directly to the DSP, ties up the whole I/O bandwidth of the DSP. A FIFO is an ideal solution to buffer some of the data. When a whole block of data has been sampled, the data can be transferred in a read burst from the FIFO to the DSP.

This application report describes how a FIFO can be used for buffering the data between a DSP and analog front ends (AFEs). The figures in this application report show how a THS1031 ADC and a THS5661A digital-to-analog converter (DAC) can be connected to a TMS320C6000™ DSP, using SN74V2x5 FIFOs as buffers.

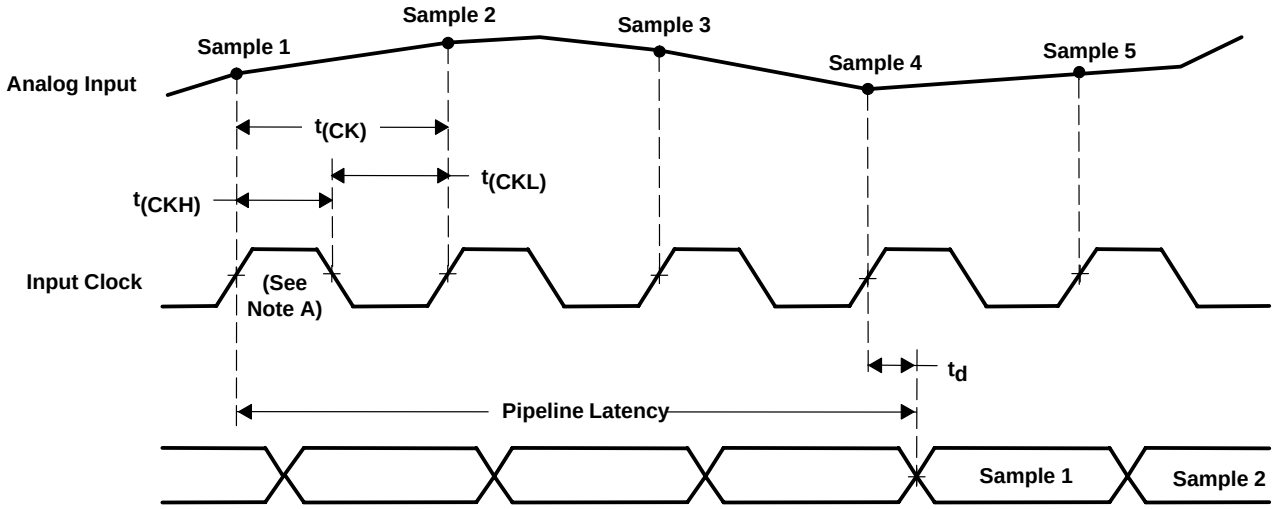
A generic discussion on connecting FIFOs to TMS320C6000 DSPs can be found in two application reports: *TMS320C6000 EMIF to External FIFO Interface* (SPRA543) and *TMS320C6000 Expansion Bus to External Synchronous FIFO Interface* (SPRA547).

2 Parts Description

2.1 THS1031 (ADC)

The THS1031 is a low-power, 10-bit, 30-MSPS ADC that operates with a 2.7-V to 3.3-V supply voltage. An out-of-range output is used to monitor any out-of-range condition in the THS1031 input range. The speed, resolution, and single-supply operation of the THS1031 are suited for set-top-box, video, multimedia, imaging, high-speed acquisition, and communications applications.

The timing diagram for the output (see Figure 1) shows why the ADC cannot be connected directly to a TMS320™ DSP.



NOTE A: All timing measurements are based on 50% of edge transition.

Figure 1. THS1031 Digital-Output Timing

The digital output of the ADC is always on, therefore, the output has no high-impedance state. If the converter is connected directly to the DSP bus, no other peripheral can be used. Furthermore, one sample must be read by the DSP every cycle of the conversion clock. This is nearly impossible to realize, even if using a DMA channel for the transfer, because other external memory transfers also can take place on the bus.

A simple latch could solve the output-always-on problem. But, if the converter were running at 30 MHz, the DSP still would have to read a new sample every 33 μs. This would tie up a significant amount of the DSP's I/O bandwidth. If the DSP is busy and can't read the next sample within 33 μs, data is lost.

A better solution is to use a FIFO as a buffer. The ADC writes the sampled values synchronous to the conversion speed into the FIFO. If a certain block of data has been written, a signal is sent to the DSP. The DSP now reads the whole block of data in a burst mode from the FIFO. This is much more efficient than performing single read commands for every sample.

2.2 THS5661A (DAC)

The THS5661A is a 12-bit-resolution DAC specifically optimized for digital data transmission in wired and wireless communication systems. The 12-bit DAC is a member of the CommsDAC series of high-speed, low-power CMOS DACs. The CommsDAC family consists of pin-compatible 14-, 12-, 10-, and 8-bit DACs. All devices offer identical interface options, small-outline packages, and pinouts. The THS5661A offers superior ac and dc performance, while supporting update rates up to 125 MSPS.

A very simple interface is provided on the digital side of the DAC. The interface consists only of the data lines and the conversion clock. The digital write timing is similar to the read timing of the THS1031 (see Figure 2).

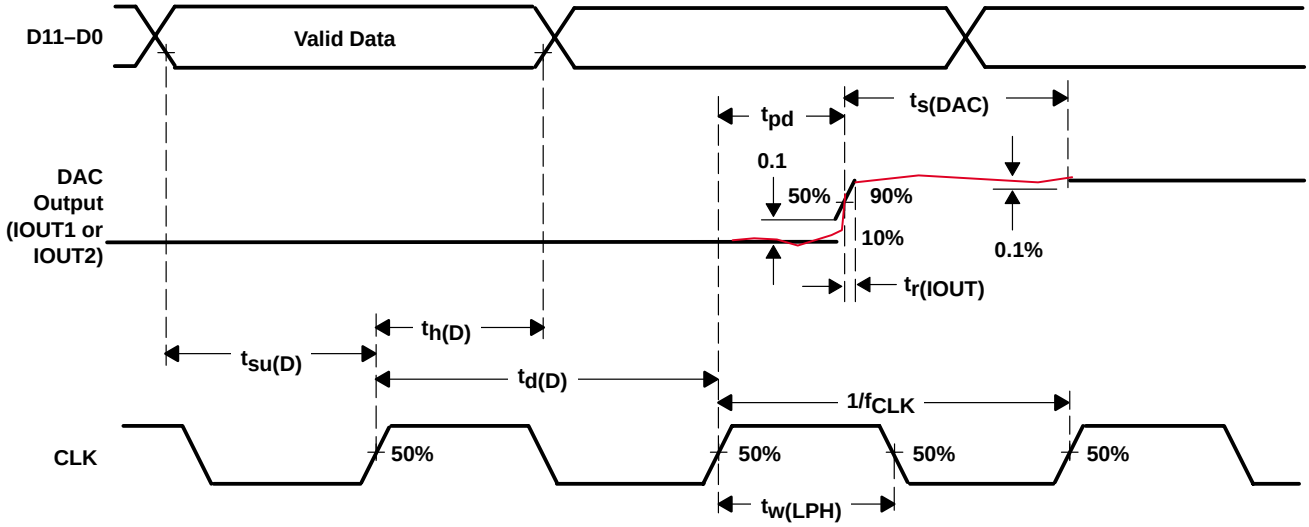


Figure 2. THS5661A Timing

A direct DSP connection is not possible because the data writes must occur synchronously to the free-running conversion clock. Using a FIFO as a buffer is the best solution for connecting the data converter to the DSP.

2.3 SN74V2x5 (Synchronous FIFO)

TI's high-performance FIFOs provide the necessary data rate to allow DSPs and AFEs to operate at peak efficiency. The SN74V2x5 FIFOs are very high-speed, low-power CMOS, clocked FIFO memories. These are synchronous FIFOs, which means each port employs a synchronous interface. The clocks for each port are independent from each other and can be asynchronous or coincident. This enables the two ports of the FIFO to run at different speeds.

Traditional FIFOs often required additional control logic to initiate the data transfer. The new generation of SN74V2x5 devices allows a glueless connection between the FIFO and the DSP. In certain applications, it is possible to avoid using glue logic between the FIFO and the data converter.

Two timing modes of operation are possible with these devices: first-word fall-through (FWFT) mode and standard mode. In FWFT mode, the first word written to an empty FIFO is clocked directly to the data output lines after three transitions of the RCLK signal. In standard mode, the first word written to an empty FIFO does not appear on the data output lines unless a specific read operation is performed.

The inputs are 5-V tolerant. Therefore, it is easy to connect 5-V data converters to a 3.3-V I/O DSP, if a FIFO is used as a buffer.

3 FIFO Connection to External Memory Interface (EMIF)

The EMIF is available on all TMS320C6000 series DSPs. This interface can be used to connect different memory types, such as asynchronous SRAM, SDRAM, or SBSRAM, to the same address and data bus. If the FIFO also is interfaced to the EMIF, it shares bandwidth with the other devices on the bus.

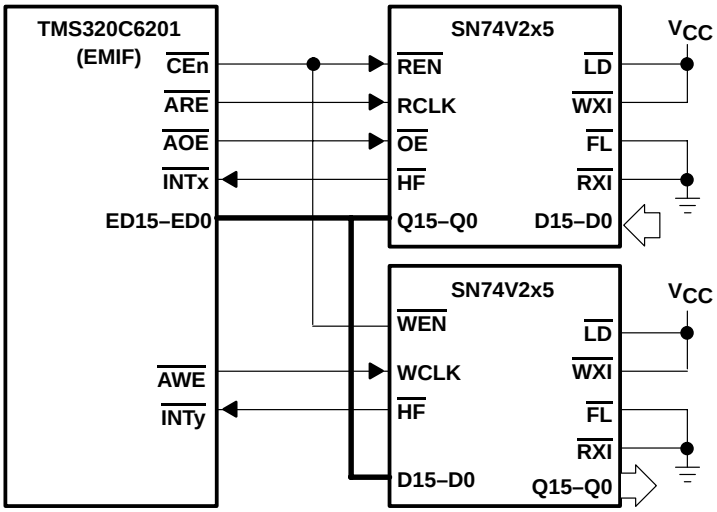
FWFT mode is used for the following examples because it offers an easier interface to the TMS320C6000 series DSP EMIF. Additional glue logic would be needed for the FIFO to operate in standard mode.

A detailed description of how to interface FIFOs to the TMS320C6000 series DSP EMIF can be found in the *TMS320C6000 EMIF to External FIFO Interface* application report (SPRA543).

3.1 Hardware Realization of EMIF-to-FIFO Interface

The family of SN74V2x5 FIFOs offers a glueless DSP interface (see Figure 3). This glueless EMIF interface can be realized by using the FIFO as an output buffer. If used as an input buffer, the FIFO should be the only asynchronous device on the EMIF. If other asynchronous devices also are connected, simple glue logic is required to control the output-enable ($\overline{OE}$) pin of the FIFO to prevent bus contention. Glue logic for $\overline{OE}$ always is required on devices with shared control pins for different types of external devices, such as the TMS320C6211/TMS320C6711 DSPs.

Because the EMIF does not support the synchronous FIFO mode, the EMIF must be configured to operate in asynchronous mode. The input pins for operating-mode selection are set up for FWFT mode.



NOTE A: This configuration can be used only if no other devices are using the asynchronous control signals or if the FIFO is used only as an output buffer, not for input. The DSP must offer separate control lines for the asynchronous and synchronous memory types (C6x0x).

Figure 3. Glueless SN74V2x5-to-EMIF Interface

If other peripherals also use the asynchronous port of the TMS320C6000 series DSP, a simple OR gate is required to generate the output enable ($\overline{OE}$) for the FIFO. Additional logic is necessary to provide address decoding if other devices are connected to the same CE space (see Figure 4).

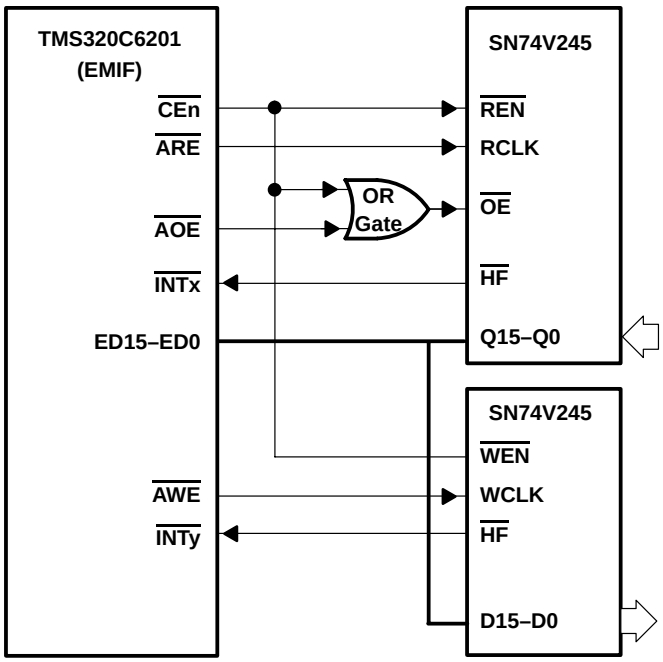


Figure 4. SN74V2x5-to-EMIF Interface With Glue Logic

The half-full flag ($\overline{HF}$) is (in both examples) used as an input signal for the DSP. For the input FIFO, $\overline{HF}$ tells the DSP that valid data is available in the FIFO (at least half the FIFO size). Either an interrupt should be generated or a DMA transfer started to read the block of data from the FIFO. On the output side, $\overline{HF}$ indicates that there is free space in the FIFO and the DSP can write a block of data.

The final hardware and software implementation is application dependent. Other flags, such as $\overline{PAE}$ or $\overline{PAF}$, can be used instead of $\overline{HF}$. The DSP can use the flags as synchronization events for the DMA or use a polling scheme to check the status of the signal and manually start a DMA transfer.

Care must be taken concerning the interrupt polarity if $\overline{HF}$ is used to generate an interrupt. By default, a low-to-high transition on $\overline{INTx}$ generates an interrupt. The FIFO is more than half full when $\overline{HF}$ is low. Therefore, the default polarity can be used for the output FIFO. The connected data converter takes data out of the FIFO and a low-to-high transition of $\overline{HF}$ occurs when the half-full boundary is crossed. This signals the DSP that space is available and the DSP can write new data to the FIFO.

The interrupt polarity must be changed for the input FIFO. An interrupt should be generated when enough data is available in the FIFO. In this case, a high-to-low transition of $\overline{HF}$ is used as the signal for the DSP. The polarity can be programmed via the External Interrupt Polarity register [see the *TMS320C6000 Peripherals Reference Guide* (SPRU190) for details].

3.2 Timing Constraints for Read FIFO Connected to EMIF

This section discusses all relevant timing parameters for an EMIF-to-FIFO connection. A ’C6201 running at 200 MHz is used as an example. The example is based on a connection without external glue logic. If glue logic is used, additional delays occur.

There are three programmable timing parameters that can be configured by software: setup, strobe, and hold. All parameters are programmable in terms of CPU clock cycles. For setup and strobe, a minimum value of 1 can be used; hold allows a minimum value of 0. This leads to a maximum read or write frequency of 100 MHz on a 200-MHz ’C6201 device.

Figure 5 shows the important timing parameters for a single asynchronous read command on the ’C6201 DSP, running at 200 MHz. Timings are based on the TMS320C6201 DSP data sheet (SPRS051).

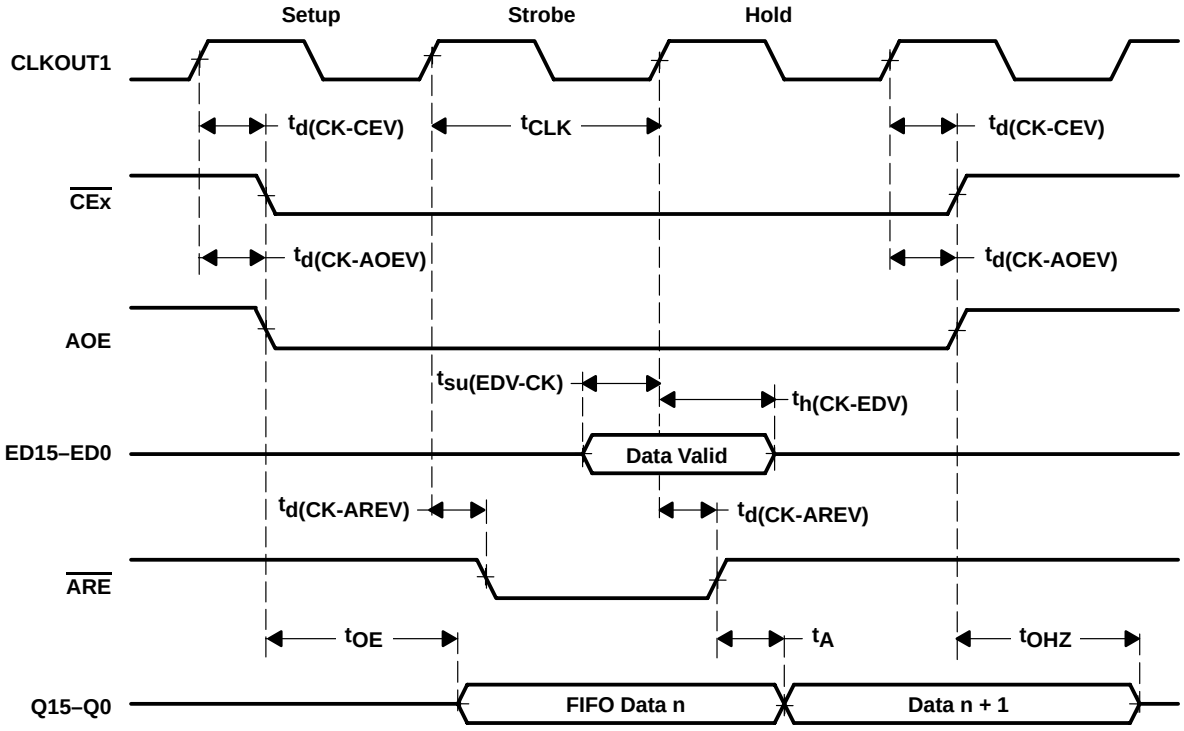


Figure 5. ’C6201 DSP Asynchronous Read Timing (Single Read)

In Figure 5, a value of 1 has been used for setup, hold and strobe. Because the asynchronous read-enable ($\overline{\text{ARE}}$) signal of the DSP is used as the read clock for the FIFO, a rising edge of $\overline{\text{ARE}}$ tells the FIFO to put the next data value on the bus. The data-valid section of the ED bus symbolizes the time when the ’C6201 expects valid data on the bus. The exact timing values for the ’C6201 and SN74V2x5 FIFO (taken from the respective data sheets) are given in Table 1 and Table 2, respectively.

Table 1. ’C6201 DSP Asynchronous Read Timing Parameters

PARAMETER		MIN	MAX	UNIT
$t_{d(CK-CEV)}$	Delay time, CLKOUT1 high to $\overline{CEx}$ valid	–0.2	4	ns
$t_{d(CK-AOEV)}$	Delay time, CLKOUT1 high to $\overline{AOE}$ valid	–0.2	4	ns
$t_{su(EDV-CK)}$	Setup time, read EDx valid before CLKOUT1 high	4		ns
$t_{h(CK-EDV)}$	Hold time, read EDx valid after CLKOUT1 high	0.8		ns
$t_{d(CK-AREV)}$	Delay time, CLKOUT1 high to $\overline{ARE}$ valid	–0.2	4	ns

Table 2. SN74V2x5 FIFO Read Timing Requirements

PARAMETER		SN74V2x5-10		SN74V2x5-7		UNIT
		MIN	MAX	MIN	MAX	
t_A	Data access time	2	6.5	2	5	ns
t_{CLK}	Clock cycle time	10		7.5		ns
t_{CLKH}	Clock high time	4.5		3.5		ns
t_{CLKL}	Clock low time	4.5		3.5		ns
t_{OE}	Output enable to output valid		6		6	ns
t_{OHZ}	Output enable to output in high Z	1	6	1	6	ns

To fulfill the input setup time of the ’C6201, the following condition must be true for the first access to the FIFO in read burst (or for a single read):

$$\text{Setup} + \text{Strobe} \geq t_{d(CK-AOEV)}\ \text{max} + t_{OE}\ \text{max} + t_{SU(EDV-CK)}\ \text{min} \quad (1)$$

In a worst-case scenario, the setup and strobe phase has to be longer than 14 ns (4 ns + 6 ns + 4 ns). If the DSP is running at 200 MHz, one CLKOUT1 cycle is 5 ns. If a value of 1 is used for setup and strobe, this leads to 10 ns and would not fulfill the timing requirements. Therefore, only one of the two programmable parameters can be 1; the other one must be ≥2. Nevertheless, to fulfill the first requirement, a value of 1 for setup and strobe can be used because the ’C6201 setup period for the first access always is a minimum count of 2.

For the following accesses, $\overline{OE}$ is already active. Now, the access time of the FIFO is important.

$$\text{Hold} + \text{Setup} + \text{Strobe} \geq t_A\ \text{max} + t_{SU(EDV-CK)}\ \text{min} \quad (2)$$

One full access must be longer than 10.5 ns (t_A max + $t_{SU(EDV-CK)}$ min = 6.5 ns + 4 ns) if a SN74V2x5-10 (100 MHz) FIFO is used. This cannot be ensured if the minimum values (1 for setup and strobe and 0 for hold) are used. This leads to the conclusion that the 100-MHz FIFO cannot operate at its maximum frequency with a 200-MHz ’C6201. To use the DSP maximum asynchronous EMIF data rate, the SN74V2x5-7 (133 MHz) FIFO is needed. With this device, one access has to be longer than 9 ns (5 ns + 4 ns).

To fulfill the hold time of the ’C6201, two conditions must be true:

$$t_{d(CK-AREV)}\ \text{min} + t_A\ \text{min} \geq t_{h(CK-EDV)}\ \text{min} \quad (3)$$

$$\text{Hold} \geq t_{h(CK-EDV)}\ \text{min} - t_{d(CK-AOEV)}\ \text{min} - t_{OHZ}\ \text{min} \quad (4)$$

If the condition in equation 3 is true, –0.2 + 2 ≥ 0.8. For the minimum hold phase, we get 0 ns [0.8 ns – (–0.2 ns) – 1 ns]. This means the hold phase could be programmed to be 0.

The last two conditions ensured are the clock high and low times. Because the clock signal for the FIFO is generated directly by $\overline{ARE}$, this signal must stay low longer than the minimum clock time of the FIFO.

$$\text{Strobe} \geq t_{CLKL} \text{ min} \quad (5)$$

$$\text{Hold} + \text{Setup} \geq t_{CLKH} \text{ min} \quad (6)$$

$$\text{Setup} + \text{Strobe} + \text{Hold} \geq t_{CLK} \text{ min} \quad (7)$$

On a 200-MHz ’C6201, the minimum strobe phase of 1 results in a 5-ns low time for $\overline{ARE}$. This is enough to fulfill the FIFO requirements.

In conclusion, the following settings for the ’C6201 programmable EMIF parameters can be used:

$$\text{Setup} = 1, \text{Strobe} = 1, \text{Hold} = 0 \quad (8)$$

This leads to a maximum data transfer rate of 100 MHz between the FIFO and the DSP. Additional information on timing constraints, examples, and discussions about how glue logic delays affect the timing can be found in the *TMS320C6000 EMIF to External FIFO Interface* application report (SPRA543).

3.3 Timing Constraints for Write FIFO Connected to EMIF

The timing for the write FIFO is similar to the timing for the read FIFO. In this case, the DSP provides data to the FIFO and must satisfy the FIFO’s setup and hold times (see Figure 6).

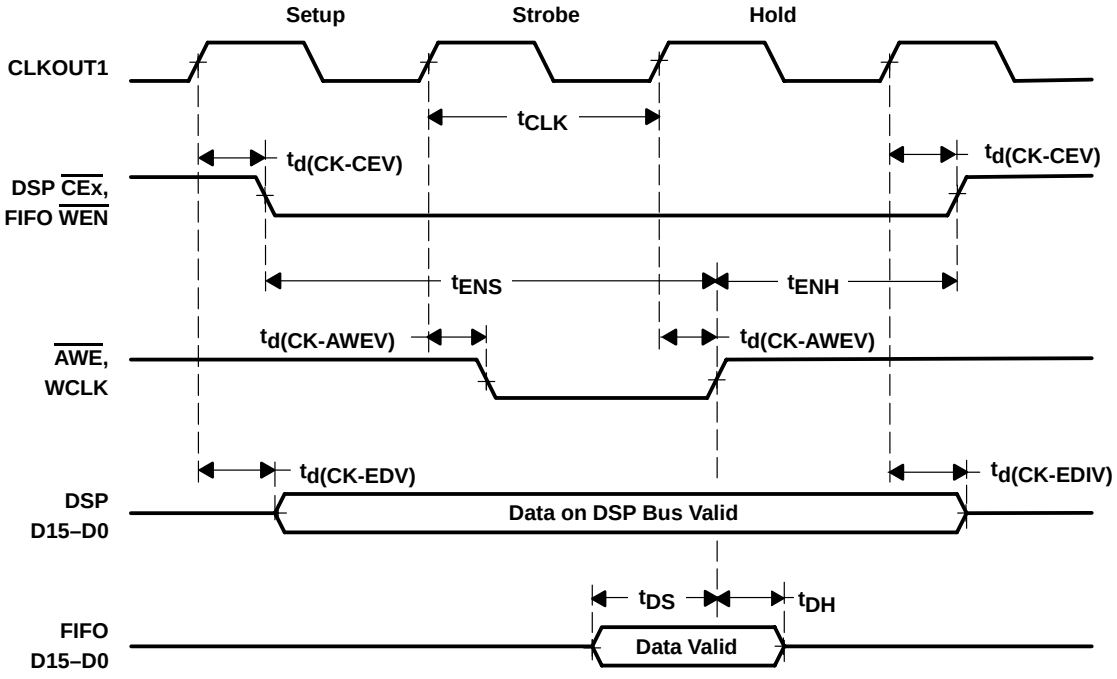


Figure 6. ’C6201 DSP Asynchronous Write Timing (Single Write)

The DSP D15–D0 signal shows the time the data on the DSP data bus is valid, whereas, FIFO D15–D0 is the time during which the FIFO needs valid data on its input data bus. Table 3 gives the timing parameters of the ’C6201 for an asynchronous write access. Table 4 gives write timing requirements of the SN74V245-10 FIFO.

Table 3. ’C6201 DSP Asynchronous Write Timing Parameters

PARAMETER		MIN	MAX	UNIT
$t_{d(CK-CEV)}$	Delay time, CLKOUT1 high to $\overline{CEx}$ valid	–0.2	4	ns
$t_{d(CK-EDV)}$	Delay time, CLKOUT1 high to EDx valid		4	ns
$t_{d(CK-EDIV)}$	Delay time, CLKOUT1 high to EDx invalid	–0.2		ns
$t_{d(CK-AWEV)}$	Delay time, CLKOUT1 high to $\overline{AWE}$ valid	–0.2	4	ns

Table 4. SN74V2x5-10 FIFO Write Timing Requirements

PARAMETER		MIN	MAX	UNIT
t_{CLK}	Clock cycle time	10		ns
t_{CLKH}	Clock high time	4.5		ns
t_{CLKL}	Clock low time	4.5		ns
t_{DS}	Data setup time	3		ns
t_{DH}	Data hold time	0.5		ns
t_{ENS}	Enable setup time	3		ns
t_{ENH}	Enable hold time	0.5		ns

Data on the bus must be valid at least t_{DS} before the rising edge of $\overline{AWE}$. This can be accomplished by:

$$\text{Setup + Strobe} \geq t_{DS}\ \text{min} + t_{d(CK–EDV)}\ \text{max} – t_{d(CK–AWEV)}\ \text{min} \quad (9)$$

Setup and strobe must be greater than 7.2 ns [3 ns + 4 ns – (–0.2 ns)]. It is possible to use the minimum of 1 for both parameters. This leads to 10-ns setup and strobe phases and leaves a margin of 2.8 ns for signal delays.

$$\text{Hold} \geq t_{DH}\ \text{min} + t_{d(CK–AWEV)}\ \text{min} – t_{d(CK–EDIV)}\ \text{min} \quad (10)$$

Equation 10 results in a hold phase that must be at least 0.5 ns [0.5 ns + (–0.2 ns) – (–0.2 ns)]. In contrast to the read interface, a value of 0 for the hold phase cannot be used. The hold phase must be at least 1. With setup and strobe also at least 1, the maximum operating frequency is 66 MHz.

$$\text{Setup + Strobe} \geq t_{D(CK–CEV)}\ \text{max} – t_{d(CK–AWEV)}\ \text{min} + t_{ENS}\ \text{min} \quad (11)$$

$$\text{Hold} \geq t_{d(CK–AWEV)}\ \text{max} – t_{d(CK–CEV)}\ \text{min} + t_{ENH}\ \text{min} \quad (12)$$

The result of equation 3 is 7.2 ns [4 ns – (–0.2 ns) + 3 ns], which is ensured by the minimum values for setup and strobe. From equation 2 there is a hold phase of 1 (5 ns). That is more than the 4.7 ns [4 ns – (–0.2 ns) + 0.5 ns] required by equation 4.

The requirements for the write clock ($\overline{AWE}$) are the same as for the read FIFO:

Strobe $\geq t_{CLKL}$ min (13)

Hold + Setup $\geq t_{CLKH}$ min (14)

Setup + Strobe + Hold $\geq t_{CLK}$ min (15)

With a 100-MHz FIFO, all conditions are true, which shows that a 66-MHz FIFO cannot be used. The SN74V2x5-15 data sheet gives a minimum clock low time of 6 ns, which requires the strobe phase of the 200-MHz ’C6201 to be 2. With setup and hold being 1, we get four cycles minimum per access, which means a maximum rate of 50 MHz. Therefore, a -10 speed-grade FIFO is needed to operate with 66 MHz on the TMS320C6000 EMIF.

For the write FIFO in this example, the EMIF parameters for asynchronous writing are:

Setup = 1, Strobe = 1, Hold = 1 (16)

4 FIFO Connected to Expansion Bus (XBUS)

The XBUS is ideally suited for connecting FIFOs because it provides a special glueless interface for synchronous FIFOs. Furthermore, the XBUS is independent from the EMIF and does not share bandwidth with devices connected to the EMIF. The XBUS only supports FIFOs operating in standard mode; FWFT mode is not supported without special attention to the first-word read from the FIFO.

A general description of how to interface FIFOs to the TMS320C6000 XBUS is in the *TMS320C6000 Expansion Bus to External Synchronous FIFO Interface* application report (SPRA547).

4.1 Hardware Interface – XBUS to FIFO

The XBUS can interface up to four write FIFOs without additional logic (one per XCE space) or three write FIFOs and a single read FIFO (in $\overline{XCE3}$ only). However, with a minimal amount of glue, up to 16 read and write FIFOs can be used per XCE space.

The example (with one read and one write FIFO as a buffer for an ADC and a DAC) could, therefore, be realized without any glue logic, if the XCE3 space is used for the read FIFO (see Figure 7). In this case, no other input peripherals can be used on the XBUS.

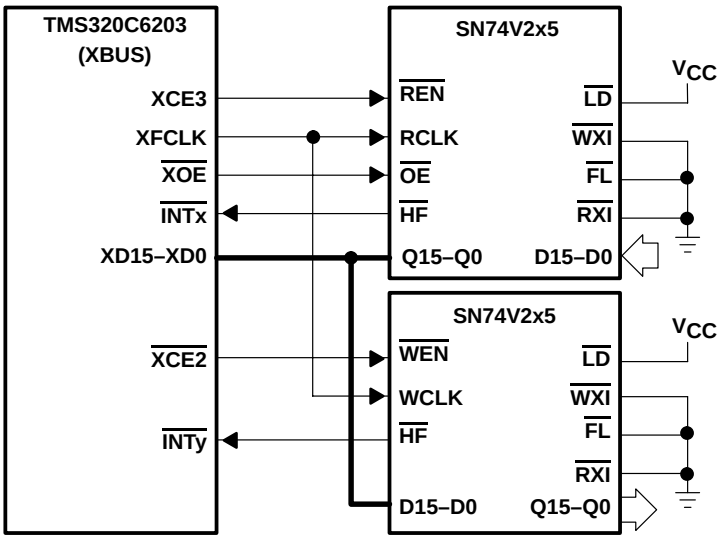


Figure 7. Glueless SN74V2x5-to-XBUS Interface

4.2 Timing Constraints for Read FIFO Connected to XBUS

The frequency of the FIFO clock (XFCLK) signal can be programmed via the FIFO clock rate (XFRAT) bits in the expansion bus global control (XBGC) register. Valid frequencies are 1/2, 1/4, 1/6, or 1/8 of the CPU clock, which leads to the possibilities for FIFO frequencies in Table 5.

Table 5. Achievable FIFO Frequencies on the XBUS

CLOCK RATE	DSP FREQUENCY		
	200 MHz	250 MHz	300 MHz
1/2 CPU clock rate	100-MHz FIFO clock SN74V2x5-10 SN74V2x3-10†	125-MHz FIFO clock SN74V2x5-7 SN74V2x3-7†	150-MHz FIFO clock SN74V2x3-6†
1/4 CPU clock rate	50-MHz FIFO clock SN74V2x5-20 SN74V2x3-15†	62.5-MHz FIFO clock SN74V2x5-15 SN74V2x3-15†	75-MHz FIFO clock SN74V2x5-10 SN74V2x3-10†
1/6 CPU clock rate	33.3-MHz FIFO clock SN74V2x5-20 SN74V2x3-15†	41.67-MHz FIFO clock SN74V2x5-20 SN74V2x3-15†	50-MHz FIFO clock SN74V2x5-20 SN74V2x3-15†
1/8 CPU clock rate	25-MHz FIFO clock SN74V2x5-20 SN74V2x3-15†	31.25-MHz FIFO clock SN74V2x5-20 SN74V2x3-15†	37.5-MHz FIFO clock SN74V2x5-20 SN74V2x3-15†

† Product preview

For a 300-MHz DSP, FIFO frequencies of 150, 75, 50, or 37.5 MHz are possible. A 250-MHz ’C6000 DSP can achieve FIFO operating speeds of 125, 62.5, 41.67, or 31.25 MHz. To bring the desired XCE space of the XBUS into FIFO mode, the MTYPE field in the associated XBUS XCEx space-control register must be set to 101b. No additional timing parameters need to be set if the synchronous FIFO mode of the XBUS is used. Alternatively, the XBUS also can operate in asynchronous mode. In this case, the same timing comparisons used for the EMIF connection are necessary.

A detailed description of the XBUS, its configuration, and examples of how to transfer data from and to the XBUS via DMA can be found in the *TMS320C6000 Peripherals Reference Guide* (SPRU190). Additional information and timing examples are shown in the *TMS320C6000 Expansion Bus to External Synchronous FIFO Interface* application report (SPRA547).

5 Interface Between AFE and FIFO

Examples in the following paragraphs show the simplest possible connection between FIFOs and AFEs. Data is transferred as long as the conversion clock is running. The DSP or an additional clock circuit can generate this clock.

5.1 ADC – THS1031 ADC-to-SN74V2x5 FIFO Interface

To enable the input FIFO, the write enable ($\overline{WEN}$) must be tied low (see Figure 8). The data bus of the THS1031 ADC is connected directly to the FIFO. Data is stored in the FIFO as soon as the data converter clock (AD_CLK) starts running.

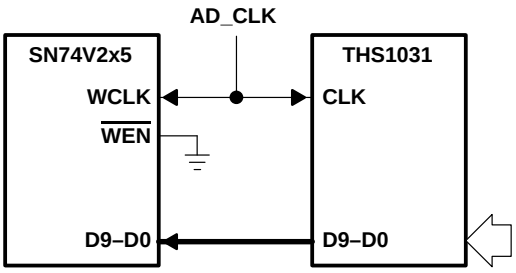


Figure 8. Simple ADC-to-FIFO Interface

The timing requirements for both devices shows that a direct connection is possible; both devices are supplied with the same clock signal. The data sheet of the THS1031 ADC states that it takes a maximum of 25 ns between the rising edge of CLK and a valid output signal on the data bus (see Figure 1). There is no note about the minimum time.

On the FIFO side, the most important parameters are the data setup time and the data hold time. Depending on the speed grade, the setup time is in the range of 2.5 ns to 5 ns, and the hold time is in the range of 0.5 ns to 1 ns. If we take a -15 speed-grade FIFO as an example, the input at the data bus must be valid 4 ns before and 1 ns after the rising edge of the write clock (WCLK).

If the THS1031 ADC is running at its maximum sample rate of 30 MHz, the cycle time is 33.3 ns. Therefore, it is ensured that the output on the data lines of the ADC is valid at least 8.3 ns before the rising edge of the clock signal (33.3 ns – 25 ns = 8.3 ns). In this case, there are no problems with the setup time of the FIFO inputs.

The data sheet for the THS1031 device says nothing about the minimum time after the rising clock edge, therefore, an exact margin for the hold time cannot be calculated. Measurements show an access time of 5 ns to 10 ns for the outputs of the THS1031. Even the slower speed grades of the SN74V2x5 FIFOs require a hold time of only 1 ns. Therefore, there should not be any problem meeting the input timing requirements of the FIFO.

Figure 9 shows the timing between the ADC and the FIFO.

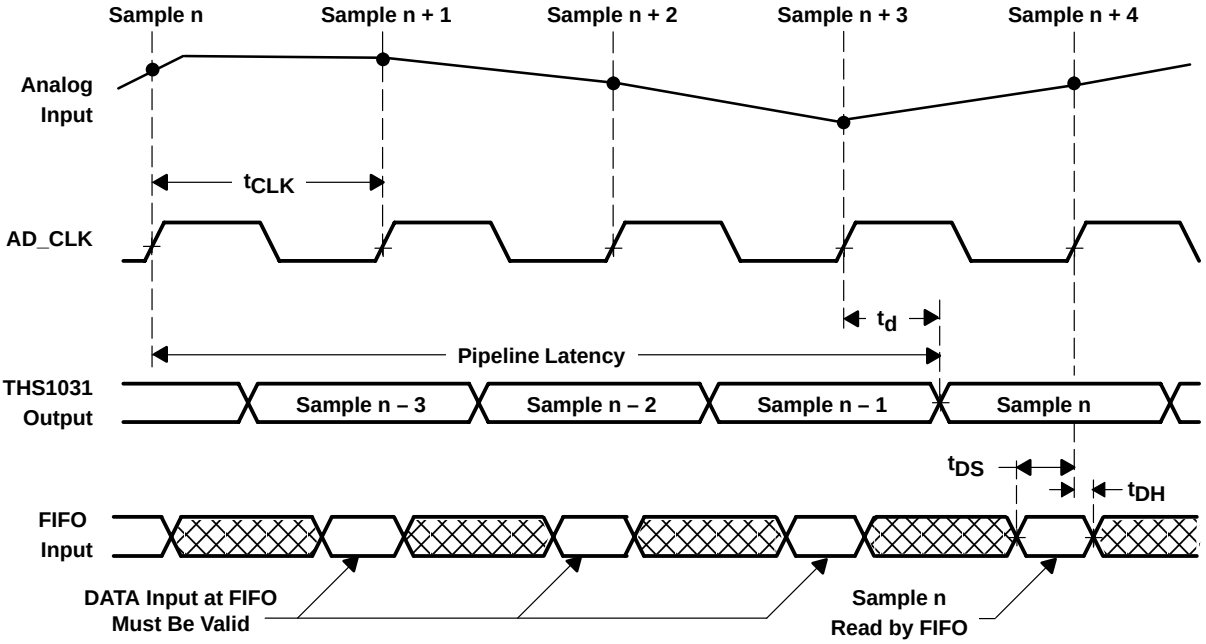


Figure 9. ADC-to-FIFO Timing

The THS1031 output signal shows the output of the ADC as it appears on the data pins of the THS1031 ADC. On the FIFO side, the shaded areas in FIFO input symbolize the moments when the data input does not care. The input has to be valid only in the white areas. To prevent timing violations, no change in the THS1031 ADC output signal must occur during these periods.

5.2 DAC – SN74V2x5 FIFO-to-THS5661A DAC Interface

The interface between the THS5661A DAC and the SN74V2x5 FIFO (see Figure 10) is similar to the ADC-to-FIFO interface. Additionally, the read enable ($\overline{\text{REN}}$) and $\overline{\text{OE}}$ of the FIFO must be tied low. Therefore, the outputs always are active.

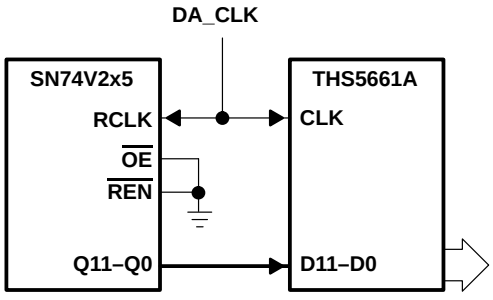


Figure 10. Simple DAC-to-FIFO Interface

The THS5661A DAC expects the input data to be valid 1 ns before and 1 ns after the rising edge of the clock signal (see Figure 11). Because the $\overline{OE}$ signal of the FIFO always is active, the output data bus of the FIFO does not go back into high-impedance mode. In this case, only the access time, t_A, of the FIFO is important.

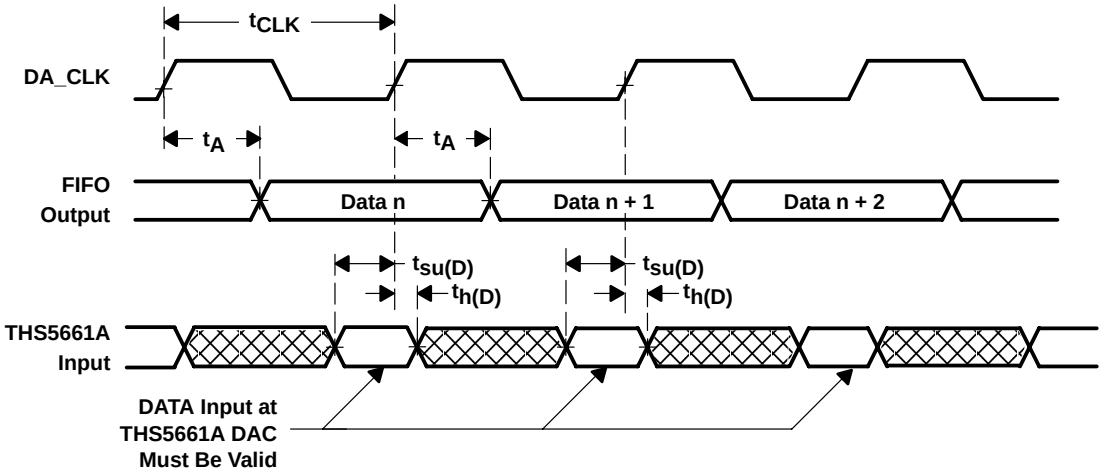


Figure 11. DAC-to-FIFO Timing

The minimum access time, t_A, for all speed grades of the SN742x5 FIFO is 2 ns (see Table 6). This ensures that the input hold time of the DAC is not violated. The following two conditions must be true to prevent timing violations on the FIFO/DAC interface:

$$t_A < t_{CLK} - t_{su\_DAC} \quad (17)$$

$$t_A > t_{hold\_DAC} \quad (18)$$

Table 6. DAC-to-FIFO Interface Timing Parameters

PARAMETER		MIN	MAX	UNIT
t_A	FIFO data access time	2	5 to 12 (depending on speed grade)	ns
t_{su_DAC}	THS5661A data setup time	1		ns
t_{hold_DAC}	THS5661A data hold time	1		ns

5.3 Additional Points to Consider

In this simple configuration, there are some important things to consider. On the input side, the THS1031 ADC starts converting samples as soon as AD_CLK starts running. To prevent the FIFO from being filled with data during the initialization phase of the DSP, an output pin of the DSP should be used to keep the FIFO in reset. If the DSP is ready, it releases reset from the FIFO and the samples are written into the FIFO. Another possibility would be to use a clock signal generated by the DSP and start clocking after the initialization of the DSP and other hardware.

The output side is more critical because the FIFO is empty at the beginning. The DSP would have to write a block of data into the FIFO before the THS5661A starts to read values from the FIFO. $\overline{\text{REN}}$ cannot be controlled by the DSP. Changing $\overline{\text{REN}}$ by the DSP would be asynchronous to RCLK and could put the FIFO in an undefined state. Holding the FIFO in reset also is not a possible solution because the DSP cannot write data into the FIFO if the FIFO is in the reset state. Having DA_CLK generated by the DSP works if the DSP starts DA_CLK after the first block of data has been written into the FIFO.

Another solution would be to ensure that, via software, at the first access, the FIFO would be filled completely. After initialization and starting the conversion clock, a block of dummy data could be written to the FIFO. The FIFO transfers this data to the DAC, and the DSP has until the FIFO is half empty to generate valid data for the FIFO.

6 Software Control

6.1 Data Transfer Flow

Depending on the application, there are numerous possibilities for software control. In most applications, different data blocks are used; one or more blocks for the CPU to work on, and other blocks that are used for the data transfer between the EDMA/DMA controller and the FIFOs. It is possible for the TMS320C6000 DSP to read in one data block from the FIFO and to write out another block in the same time frame. In this case, the DMA would automatically switch between the two transfer channels. But, this causes delays every time the EMIF switches from input to output mode. Therefore, for best performance, only one block transfer (read or write) should be active.

The following example shows a simple data-transfer method for an application in which the ADC and DAC run at the same clock speed. Only two data blocks are used in this example, data block A and data block B (see Figure 12).

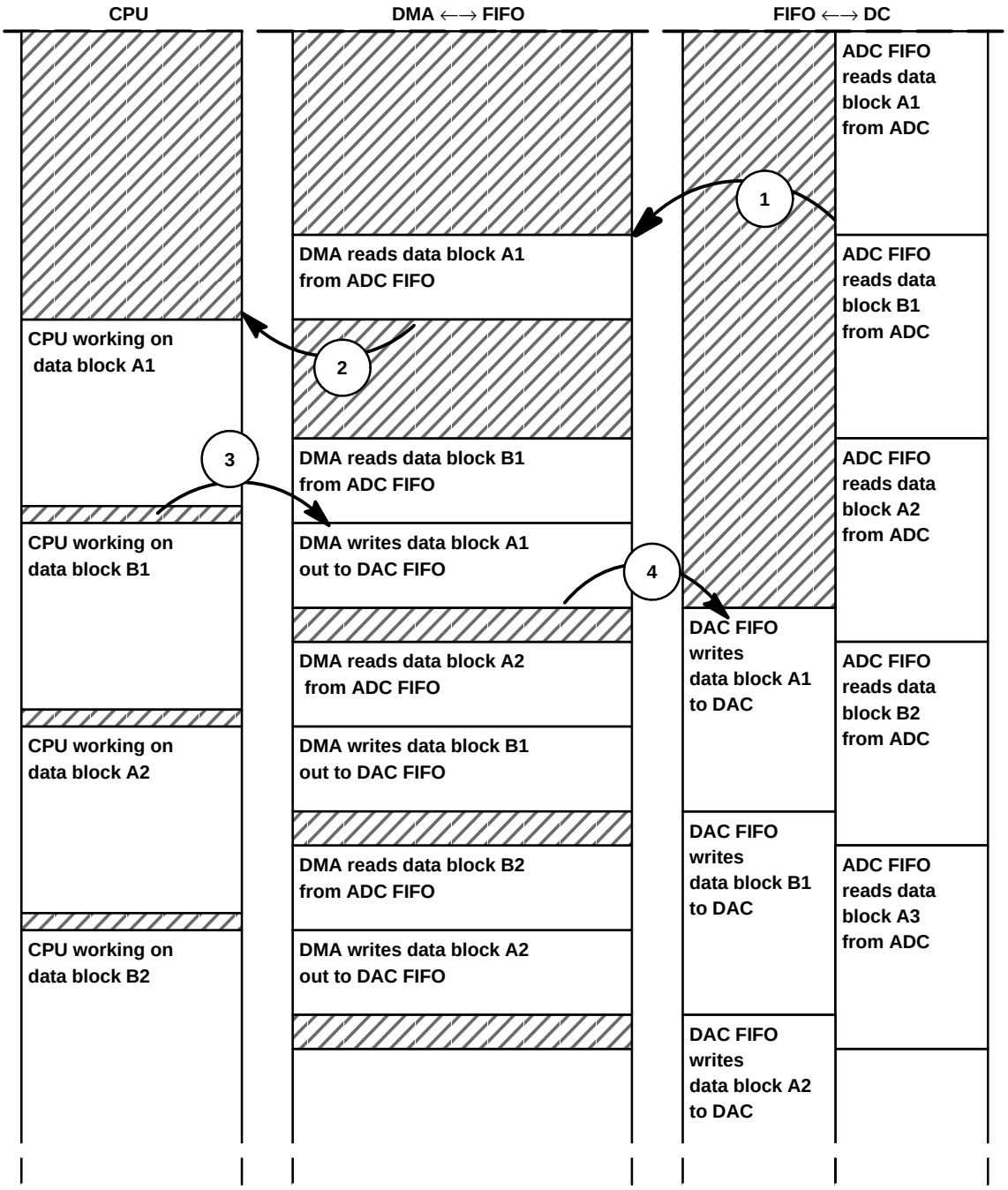


Figure 12. Software Flow in a Simple Application

After reset, the CPU must do all necessary initializations and release reset from the ADC FIFO. This FIFO now starts to read values from the THS1031 ADC. At this moment, the CPU has no data to work on. After the ADC is half full, an interrupt is generated (1, Figure 12). In the ISR, the DMA is set up to transfer the sampled values from the FIFO into data block A. When the transfer is finished, a DMA-completion interrupt is generated. This ISR sets a flag that signals the CPU that data block A now contains valid data (2, Figure 12).

Now, the CPU can start working on the sampled data. In this simple example, with only two data blocks used, the sampled data is replaced by the result generated by the CPU. During processing, a new ADC interrupt can occur. The ISR now sets up the DMA to fill data block B with sampled data from the FIFO. Because this transfer runs independently from the CPU, the CPU can continue to process the data in block A. After the CPU is finished with block A, it waits until valid data is available in block B (DMA read finished). Now, the CPU starts a DMA transfer from data block A to the output FIFO (3, Figure 12) and sets the data-valid flag for block A to invalid.

There are different possibilities for controlling the DAC FIFO. The DAC clock could be started immediately after reset of the DSP, if the DAC FIFO is held in reset. In this case, reset must be released immediately before the DMA first starts to transfer data from the DSP's internal memory to the FIFO (3, Figure 12), or reset can be released immediately after the DSP starts working, if the DAC clock is not running. Here the DSP can start the DAC clock any time before the DAC FIFO is filled completely.

The timing is very simple and no problems occur if the following conditions are fulfilled:

- The time to process one block by the CPU is shorter than the time it takes the FIFO to read one block from the ADC.
- The DMA transfer is fast enough to read one block from the ADC FIFO and transfer another block to the DAC FIFO in the same time the ADC FIFO is reading one block of samples from the ADC.

6.2 Debugging

Debugging a DSP system with FIFOs is not an easy task, especially if the DSP is halted for emulation. Depending on the system, the data converter clock might continue to run and, therefore, fill the input FIFO or empty the output FIFO. This could put the whole software flow out of synchronization if execution should continue after some time.

If the conversion clock is generated by the DSP, the clock signals could be turned off in certain code sections and breakpoints would be allowed only in these sections. But, this leads to a clock signal that is not running continuously, which is not acceptable in many cases.

The safest, but also most complex way, is to monitor the flags of the FIFO by software to recognize any FIFO overflow or underflow. Here the FIFO flags are connected not only to interrupt signals, but also to general-purpose I/O pins. The external interrupt pins on the TMS320C6000 devices are edge sensitive. Therefore, they cannot be used to verify the status of a FIFO flag at a certain time. For example, if the DSP is halted, but the input FIFO continues to read values from the ADC, the DSP does not recognize the change of $\overline{HF}$ and the FIFO is filled completely. When the DSP continues its work, it might wait forever for $\overline{HF}$ to change. A periodic check of the flag via a general-purpose I/O pin detects such overflows and the software can try to resynchronize.

Care must be taken when using watch windows during debugging. Data from the FIFO is read if a watch window or a memory view window accesses the FIFO's address range. In this case, the read data is lost for the DSP program when it continues to run. Unless the DSP program takes care of such situations, a malfunction in the program can occur.

6.3 Programmable Flags

Besides $\overline{HF}$, the SN74V2x5 FIFOs also offer $\overline{PAF}$ and $\overline{PAE}$ flags. These flags become active if the FIFO is nearly full or empty. If the default values are sufficient, these flags also could be used as a signal to the DSP for starting a new transfer to or from the FIFO. This might help make better use of the FIFO's internal memory size.

Full-flag and empty-flag offset values are user programmable. The SN74V2x5 FIFOs have internal registers for these offsets. The default values are given in the data sheet. To program the offset, load ($\overline{LD}$) and $\overline{WEN}$ must be held low. A low-to-high transition of WCLK transfers the value at the D data bus into the empty offset register; the next WCLK pulse programs the full offset register. This can be achieved easily for the output FIFO, because the D bus (data input) of the FIFO is connected directly to the DSP. $\overline{LD}$ can be controlled via a general-purpose I/O pin of the DSP.

Programming is more complicated on the input FIFO. Here the D bus is connected to the ADC. Additional logic is required to load values in the offset registers.

6.4 Bit Assignment

The TMS320C6000 DSP family is capable of loading 8-, 16-, or 32-bit values, but the THS1031 ADC offers only ten bits on its data bus. These ten bits could be connected to the upper or lower data-bus bits of the DSP. If the DPS is configured for a 16-bit read, six bits would be left unconnected. Pulldown resistors should be used to tie these bits to a defined level. Otherwise, these bits can float and lead to unwanted results. Figure 13 shows an example with the THS1031 ADC connected to the lower ten data bits.

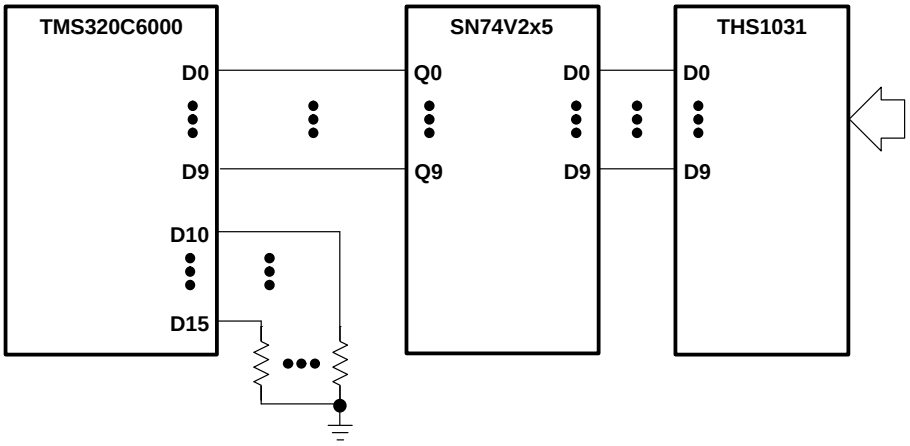


Figure 13. Pulldown Resistor for Unused Input Data Pins

Figure 14 shows an example of the THS5661A connected to the upper 12 bits in a 16-bit data-bus configuration. This bit assignment could be useful in systems where the Q15 format is used.

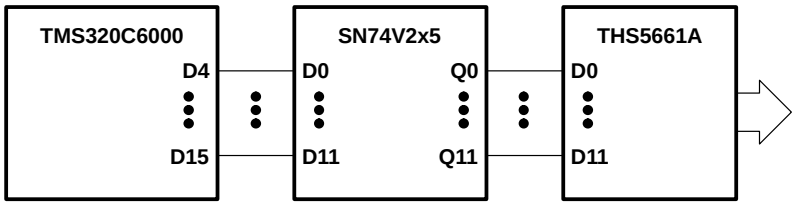


Figure 14. Bit-Assignment Example

6.5 Maximizing Data Throughput in Multichannel Systems

Different addresses for each FIFO could be used if more than one input or output channel is needed. In this case, one data access must be made for each of the FIFOs. To maximize data throughput, two FIFOs can be connected to the 32-bit data bus of the TMS320C6000 series DSP. One FIFO uses the lower 16 bits; the other FIFO uses the upper 16 bits. In this case, only one read command is needed to read both FIFOs.

7 References

1. *TMS320C6000 EMIF to External FIFO Interface* application report(SPRA543)
2. *TMS320C6000 Expansion Bus to External Synchronous FIFO Interface* application report (SPRA547)
3. *TMS320C6000 Peripherals Reference Guide* (SPRU190)
4. SN74V2x5, 3.3-V CMOS First-In, First-Out Memories data sheet (SCAS636)
5. TMS320C6201 Fixed-Point Digital Signal Processor Data Sheet (SPRS051)
6. THS1031, 10-Bit, 30-MSPS CMOS Analog-to-Digital Converter data sheet (SLAS242)
7. THS5661A, 12-Bit, 125-MSPS Digital-to-Analog Converter data sheet (SLAS247)

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