Application Report
SPRACR2–March 2020

Enabling Peripheral Expansion Applications Using the HIC

Praveen Kumar Ravichandran

ABSTRACT

C2000™ microcontrollers offer several differentiated peripherals, enabling real-time control intensive industrial and digital power applications. The Host Interface Controller (HIC) allows an external application controller to take advantage of the C2000's differentiated peripherals and processing capabilities through a commonly supported Asynchronous interface. For instance, the Fast Serial Interface (FSI) is capable of supporting low-latency and robust high-speed communication across isolation in a system. Using the HIC, any application processor can integrate FSI on to their system to perform high speed communication over isolation. This application report introduces example HIC applications and helps to select the right configuration of the HIC peripheral as per the application requirements.

You are expected to be familiar with the *HIC* chapter of an applicable C2000 Device's Technical Reference Manual (TRM), which covers the complete feature set and register definitions.

Contents

List of Figures

List of Tables

Trademarks

1 Introduction

The Host Interface Controller (HIC) provides an asynchronous interface through which an external Host processor can access most of the C2000 device's peripherals.

NOTE: This document includes HIC usage examples using the F28002x Device, but the concepts also apply to other C2000 devices that support the HIC. Throughout the document, the designation of “Host” will refer to the external Host processor, while the designation of “Device” will refer to the Device with support for the HIC.

Through the HIC, the Host can use the Device as a peripheral expander, by taking advantage of the C2000’s differentiated features. For example, Figure 1 shows how the Host with an Asynchronous RAM interface can use the Device as a peripheral bridge for high-speed communication across isolation using the FSI.

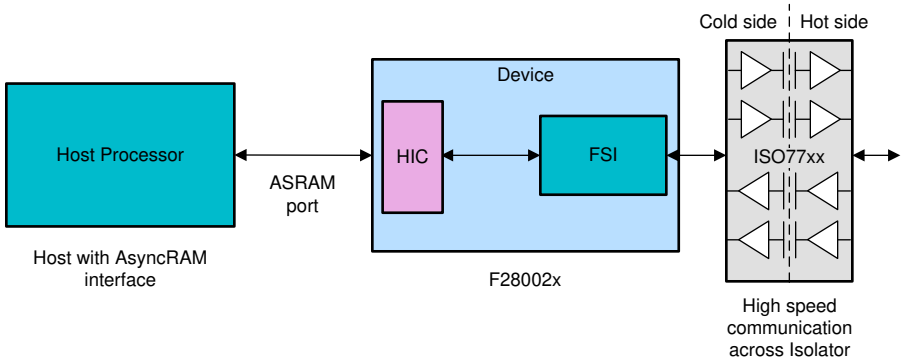

Figure 1. HIC Bridge for FSI Applications

Another application of the HIC could be to take advantage of the Configurable Logic Block (CLB) peripheral that is available with the C2000 family of devices. The CLB enables custom logic implementation and augments the existing C2000 peripheral set, thereby eliminating or reducing the need for FPGA, CPLD, or external logic components to achieve the same results. There are numerous solutions that can be implemented with the CLB, one of which is the absolute encoder protocol implementation to interface with the position sensors in an industrial drive control system. For the encoder protocols that are supported by the CLB, see the Position manager technology. The HIC can be used as a bridge for the Host to interface with the position sensors as shown in Figure 2.

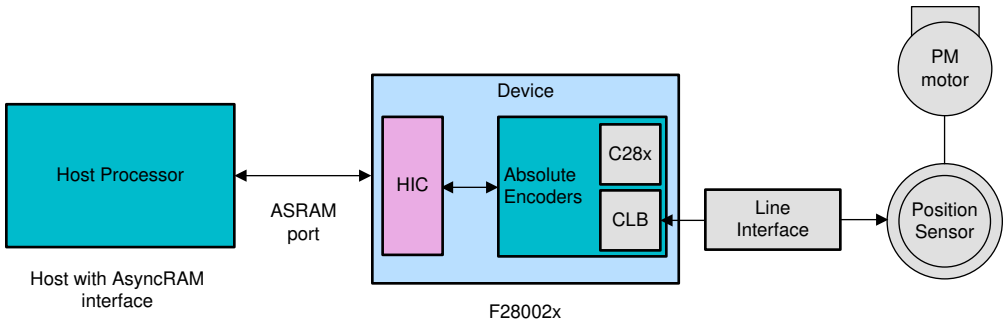

Figure 2. HIC Bridge for Position Encoder Applications

In a similar fashion, F28002x can be used as a motor controller by the Host taking advantage of the peripherals like PWMs, ADCs and CMPSS. Position reference data can be sent from the Host to the F28002x and the feedback data like motor current, present position can be accessed via ADC.

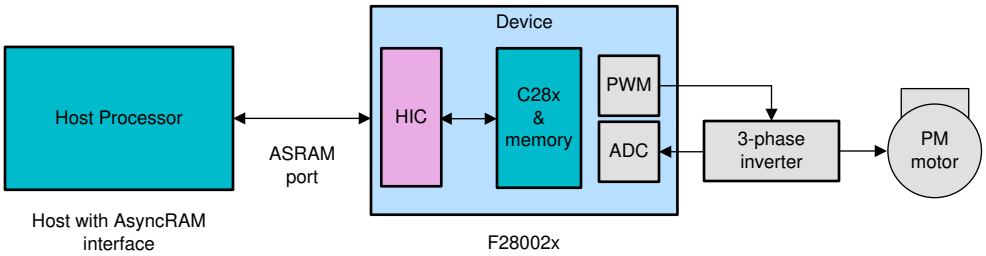

Figure 3. HIC Bridge for Motor Control Applications

HIC applications are not just limited to peripheral expansion; the Host can offload math intensive computations to the C2000 Device whose enhanced instruction set can be leveraged to increase the system performance in real-time applications. To know about different performance enhanced instruction sets and accelerators supported by C2000 family of devices, see the *Accelerators: Enhancing the Capabilities of the C2000 MCU Family Technical Brief*.

2 HIC Configurations Overview

The HIC supports a variety of configurations with which the Host can be connected. Each configuration has certain advantages and implications with respect to latency, input/output (I/O) pin overheads and so on. For a full overview of the HIC features, see the *HIC* chapter of the device-specific technical reference manual. This section deals with the various configurations and serves as a guide in selecting the right configuration according to the system requirements.

2.1 Access Modes

The HIC supports two access modes through which the Host can access the Device peripherals:

- Mailbox access mode
- Direct access mode

For the configuration details and the programming sequences for each of these usage models, see the *HIC* chapter of the device-specific Technical Reference Manual.

In the direct access mode, the Host CPU can directly access the memory map of the accessible peripheral in the Device; the Host access to the peripheral does not need Device CPU intervention once configured.

In Mailbox access type, the Host access is limited to the HIC's registers and any access to the peripheral should be facilitated via the Device's CPU or DMA to transfer data between the peripheral registers and HIC mailboxes.

Mailbox accesses have the advantage of fixed and lower latency when compared to the direct accesses to the peripherals. This is due to the fact that Host accesses in direct access mode are routed through the Device Bus-Interconnects which may arbitrate with CPU and DMA within the device.

2.2 Data Width Selection

Support for 8-bit and 16-bit HIC data width modes is controlled via the HICMODECR.DW_MODE register field. In order to perform a 32-bit access, the Host has to initiate four accesses in 8-bit mode, while two accesses are sufficient in the case of 16-bit mode as shown in Table 1. This gives 16-bit data width mode an advantage over the 8-bit mode in terms of raw throughput at the cost of incurring 8 additional I/O pins for data.

Table 1. Host Transaction for Different Data-Width Modes

Intended Write by the Host	Data Width Mode	Write cycle1	Write cycle2	Write cycle3	Write cycle4
Address = 0x5000 Data = 0x12345678	16 bits	HICDBADDR0 = 0x5000 *HIC*_A[7:0] (1) = 0x0 *HIC*_D[15:0] = 0x5678	HICDBADDR0 = 0x5000 *HIC*.A[7:0] = 0x1 *HIC*.D[15:0] = 0x1234	-	-
	8 bits	HICDBADDR0 = 0x5000 *HIC*.A[7:0] = 0x0 *HIC*.D[15:0] = 0x78	HICDBADDR0 = 0x5000 *HIC*.A[7:0] = 0x1 *HIC*.D[15:0] = 0x56	HICDBADDR0 = 0x5000 *HIC*.A[7:0] = 0x2 *HIC*.D[15:0] = 0x34	HICDBADDR0=0x5000 *HIC*.A[7:0] = 0x3 *HIC*.D[15:0] = 0x12

(1) Fields in italics are I/O pins.

The address translation for different data width modes is covered in detail in Appendix A.

2.3 Base Address Selection

In the direct access mode, the addressable space inside the Device is much larger while the HIC interface has a limited number of address lines. In order to efficiently address a larger space, the HIC interface breaks down the addressable space into regions.

For example, to access an address 0x8000_8040 of the Device, the region base address of 0x8000_80 is programmed in the HICBASEADDRx register and the offset value of 0x40 is extracted from the incoming 8-bit address lines.

The HIC interface provides 8 base address registers(HICDBADDRx) to configure the desired region base addresses. The HIC can quickly access different peripheral regions efficiently by selecting the corresponding HICDBADDRx. This region selection can be done in either of two ways:

- BASE_SELECT field in the HICBASESEL register
- HIC_BASESEL[2:0] I/O pins from the Host

This selection is controlled by the HICHOSTCR.PAGESEL register field.

The I/O based region selection (done via HIC_BASESEL[2:0] pins) enables the Host to access locations from different regions without the additional latency of configuring the HICBASESEL register between accesses. However, if the application is pin constrained, the register selection method can be used. This method requires an additional Host-Register write cycle (accesses shown in violet in Figure 4) in between for cross-region accesses.

Figure 4 and Figure 5 show the dynamic region crossing across SRAM, CAN, EPWM1 peripherals for both modes. These diagrams assume that HICDBADDR1, HICDBADDR3 and HICDBADDR6 are programmed to SRAM, CAN, EPWM1 base addresses.

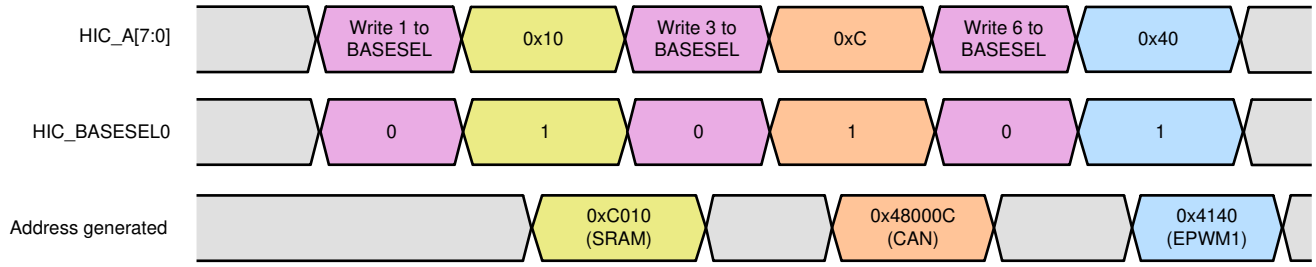

Figure 4. Cross-Region Access With Register BASESEL Select

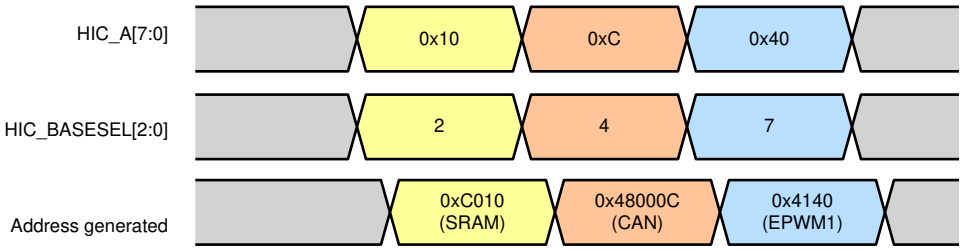

Figure 5. Cross-Region Access With I/O BASESEL Select

Note that the HIC_BASESEL[2:0] pins should be driven with a value of 2 to select HICDBADDR1 in the I/O pin selection mode.

2.4 Read/write I/O Configuration

The HIC supports dual pin and single pin mode for read and write configuration which is controlled by the HICMODECR.RnW register field. In case of dual pin mode, the HIC_nCS acts as the enable pin, whereas, the HIC_nOE for reads and HIC_nWE for writes acts as the strobe/select for each access. Figure 6 shows the timing diagram of the read and write access in dual pin mode.

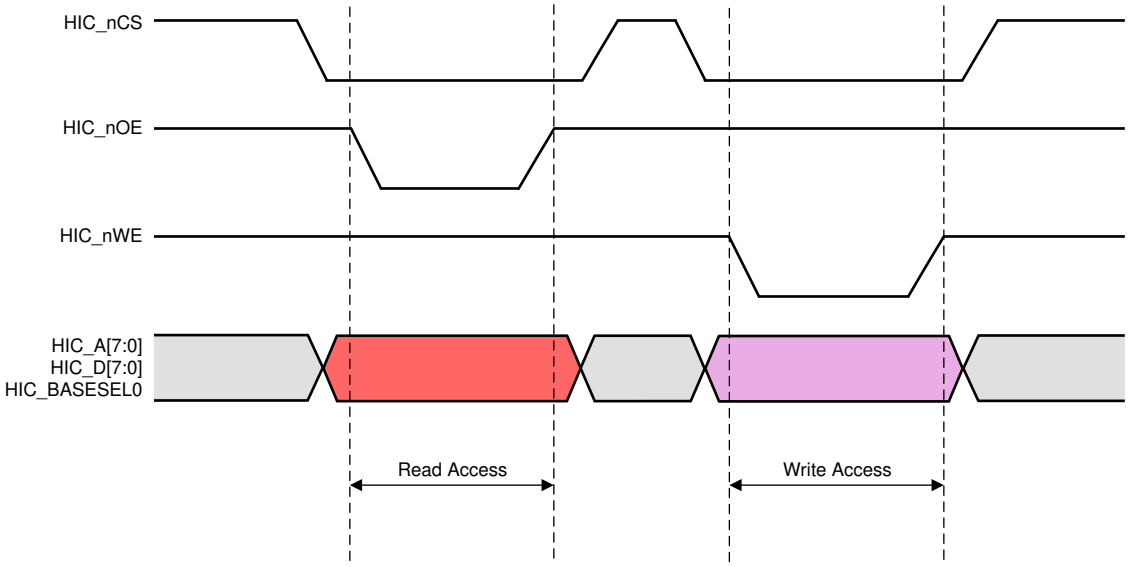

Figure 6. Read and Write Access in Dual Pin Mode

In the single pin mode, the HIC_nCS pin acts as the strobe/select for each access and the HIC_nOE pin acts as a level pin to select read or write. It indicates a read acess when set to 1 and a write access when set to 0. Figure 7 shows the timing diagram of the read and write access in single pin mode.

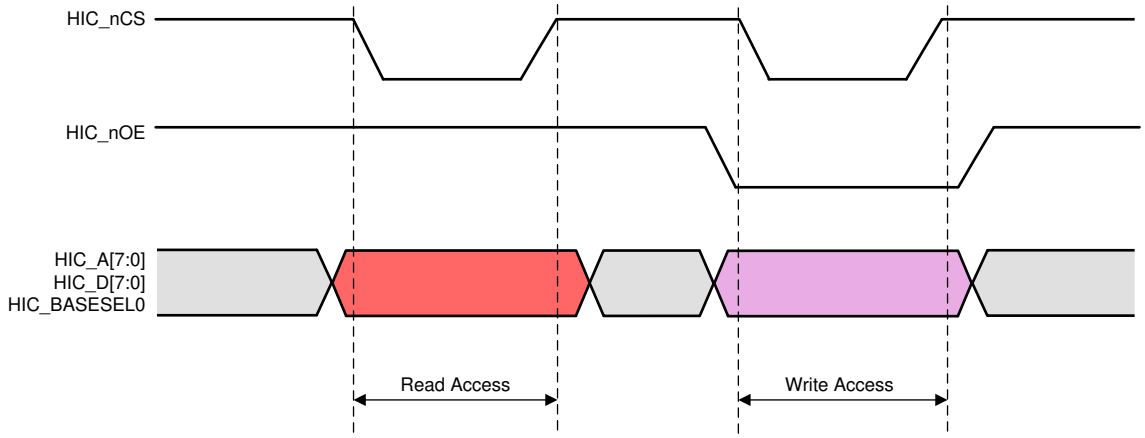

Figure 7. Read and Write Access in Single Pin Mode

If the Host memory controller supports strobe timing control of HIC_nCS for each access, the application can take advantage of the single pin mode and free up one pin for other requirements. One example of this mode is the select strobe mode supported by the C2000 External Memory Interface (EMIF) controller.

2.5 Device to Host Interrupts

The HIC module supports Host interrupt capability by using the HIC_INT Device output pin. Any general-purpose input I/O pin of the Host with the interrupt capability can be connected to the HIC_INT pin. There are two ways of generating interrupts from the Device through HIC_INT as described in the subsequent sections.

2.5.1 Device Internal Events

The HIC_INT port can be directly triggered through the EVENTRIG bus by supported peripherals as listed in the *Event Trigger Sources* table under the *HIC* chapter in the device-specific technical reference manual. For example, the FSI receiver can be configured to trigger an interrupt event (FSIRX_INT1) on a data tag match and this event can be enabled to generate a HIC_INT to the Host by enabling the corresponding HICD2HINTEN.EVTRIG bit. If the HIC_INT line is connected to an interrupt capable input pin of the Host, the FSI receiver can generate a direct interrupt to the Host CPU whenever there is a data tag match on the received data, thereby eliminating any intervention from the Device CPU.

Using the Device internal interrupt mapping capability and the direct Device access mode, the Host can access the peripheral of interest (that are accessible through the HIC) as if it is available on its own memory map with interrupts mapped directly to its CPU. This is depicted in Figure 8.

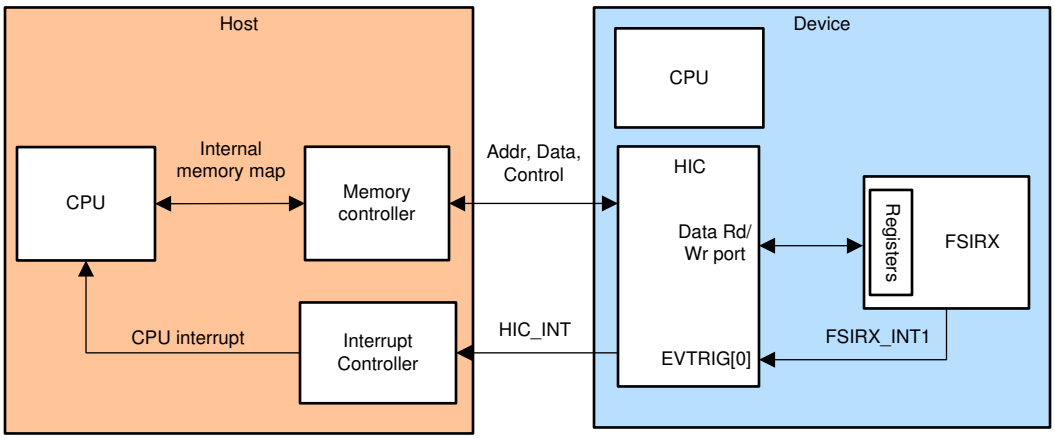

Figure 8. Device Internal Event Usage Model Using FSI

2.5.2 Software Interrupts

The HIC provides a software method to generate an interrupt to the Host. Any write into the HICD2HTOKEN register will trigger an interrupt on the HIC_INT pin. This method enables the extension of Host interrupt capability to Device peripherals that do not have direct interrupts mapped to the EVTRIG bus. For example, Figure 9 shows the Host-Device sequence to interrupt the Host CPU on a Device ADC End Of Conversion event.

Custom data exchange protocols can be implemented using the mailbox registers (D2H_BUFn and H2D_BUFn) and the token registers. For example, the Device can fill the D2H_BUFn registers with data to be exchanged and notify the Host by writing HICD2HTOKEN register with the number of words available in the D2H_BUFn registers.

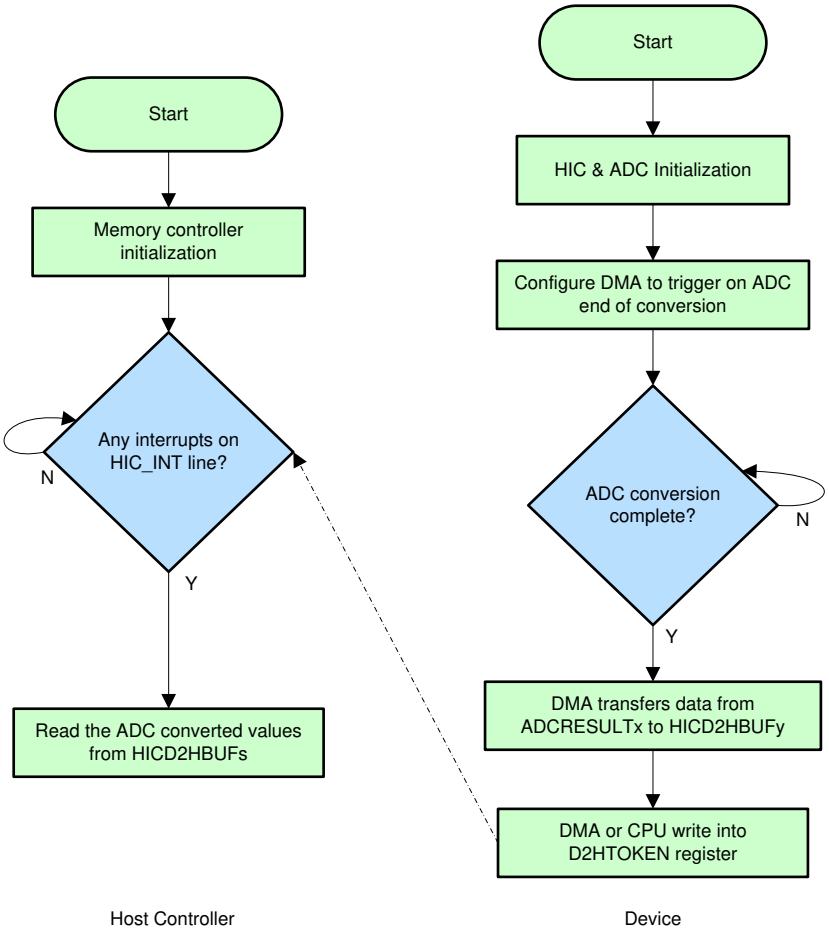

Figure 9. Software Interrupt Usage Model

3 Hardware Considerations

3.1 Common Signal Names

The HIC pins are designed to be connected directly to any Host controller which supports asynchronous memory. For the Host to Device pin mapping, see the *Connections* section under the *HIC* chapter in the device-specific technical reference manual. The signal naming convention could vary between different Host controllers. Some common Host signal name variants are covered in Table 2.

Table 2. HIC Pins and Their Common Name Variants

HIC Pin	Common Signal Name Variant
nCS	$\overline{CE}$
Dx	DQx, IOx
nBEx	DQMx, BHE/BLE, be0/1n
nOE	$\overline{OE}$, OEN, ren
nWE	$\overline{WE}$, WEN
nRDY	xWAIT

The polarity of each control pin can be configured based on the Host controller's pinout using the HICPINPOLCR register.

3.2 Address Pin Mapping

The address pin mapping between the Host and Device could vary based on the data width modes and the addressing scheme of the Host memory controllers.

A Host controller that uses the address bit 0 to represent the least significant bit of the 8-bit word address can be directly connected to the HIC address bus for 8-bit data width mode. While in case of the 16-bit data width mode, HIC_A0 has to be connected to the address bit 1 of the Host controller, since the Host address bit 1 represents a unique 16-bit word.

An example of this type of controller would be the General Purpose Memory Controller (GPMC) of the Sitara™ AM57x processor family. Table 3 shows the address translation between the GPMC and HIC for different data width modes

Table 3. Address Pin Mapping for the Sitara GPMC Controller With HIC

Device Signal	Host Signal	
	Data Width = 16	Data Width = 8
HIC_A0	GPMC_A1	GPMC_A0
HIC_A1	GPMC_A2	GPMC_A1
HIC_A2	GPMC_A3	GPMC_A2
HIC_A3	GPMC_A4	GPMC_A3
HIC_A4	GPMC_A5	GPMC_A4
HIC_A5	GPMC_A6	GPMC_A5
HIC_A6	GPMC_A7	GPMC_A6
HIC_A7	GPMC_A8	GPMC_A7

Some of the Host memory controllers (for example, the EMIF controller of the C2000 Device family) uses an address shifting scheme to maintain uniform memory capacity for all word sizes by supplementing the address pins with additional pins like EMxBAy. For these controllers, the Host address bit 0 represents a unique 32-bit word and, hence, the address pins of such controllers have to be connected as described in Table 4.

Table 4. Address Pin Mapping for the C2000 EMIF Controller With HIC

Device Signal	Host Signal	
	Data Width = 16	Data Width = 8
HIC_A0	EMxBA1	EMxBA0
HIC_A1	EMxA0	EMxBA1
HIC_A2	EMxA1	EMxA0
HIC_A3	EMxA2	EMxA1
HIC_A4	EMxA3	EMxA2
HIC_A5	EMxA4	EMxA3
HIC_A6	EMxA5	EMxA4
HIC_A7	EMxA6	EMxA5

3.3 BASESEL Pin Mapping

As discussed in the Section 2.1, the Mailbox access mode is restricted to the HIC registers. In the Direct access mode, the Host can access both the HIC registers and the Device peripheral registers. These accesses are differentiated by the HIC_BASESEL0 pin. If an access is initiated with the HIC_BASESEL0 set to 0, it will be treated as Mailbox access operation, and if the pin is set to 1, the access will be treated as a Direct access operation.

For Direct access mode, the higher unused address pins of the Host memory controller can be connected to the HIC_BASESEL[2:0] pins and the corresponding addresses in the Host memory map can be used as the Device access offset. The Table 5 shows the BASESEL pin mapping for different access types for 16-bit data width.

Table 5. BASESEL Pin Mapping With C2000 EMIF Host

Device Signal	Host EMIF Signal		
	Mailbox Access	Direct Access	
		HOSTCR.PAGESEL=0	HOSTCR.PAGESEL=1
HIC_BASESEL0	X (1)	EMxA7	EMxA7
HIC_BASESEL1	X (1)	X (1)	EMxA8
HIC_BASESEL2	X (1)	X (1)	EMxA9

(1) X = IO pins are not to be selected for HIC function.

4 Example Configuration for Pin Constrained Applications

This section provides the HIC configuration that uses the least number of I/O pins connected to C2000 EMIF controller. A Host side and Device side example for this configuration is provided in the C2000ware Package. The Host memory controller used in this example is the F2838x EMIF connected to a F28002x HIC device.

The example projects can be accessed here:

- Host side:<C2000Ware InstallDir>\driverlib\f2838x\examples\c28x\emif\CCS\emif_ex8_8bit_asram_hic_adc
- Device side: <C2000Ware Install Dir>\driverlib\f28002x\examples\hic\CCS\hic_ex2_config_8bit_adc

4.1 Test Setup

Table 6 captures the HIC and EMIF configuration used in the example. The total number of pins required for this configuration is 19.

Table 6. Device and Host Configurations for Pin Constrained Applications

HIC Configuration	EMIF Configuration
Access type: Mailbox mode Data width: 8-bit mode Read/write: Single pin	Access Type: Select strobe ASIZE: 8-bit data bus

Figure 10 shows the hardware connections required for the example. The access type used in this example is mailbox mode and hence the HIC_nRDY and HIC_BASESEL pins are not connected. Also, since the data width mode is set to 8, the byte enables are not required.

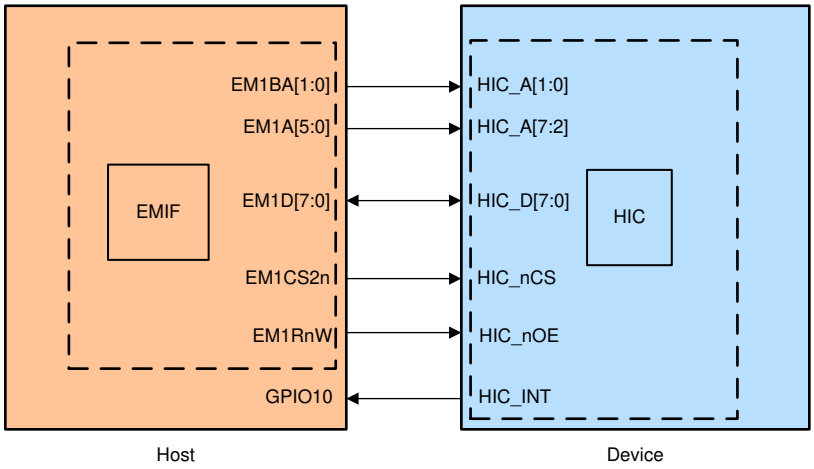

Figure 10. Hardware Setup for the Example hic_ex2_config_8bit_adc

4.2 Test Description

- This example uses mailbox access mode on F28002x side to access the ADC Result on the Device side.
- The Device side sets up the HIC, DMA, ADC SOCs for conversion and sends one Mailbox D2HTOKEN to the host.
- The Host triggers ADC conversion by passing a H2DTOKEN to the Device.
- On reception of the token, the Device triggers a Timer that periodically triggers the ADC conversion.
- On each ADC End of Conversion, the DMA gets triggered which copies the ADC result to the mailbox buffers and notifies the Host with a D2HTOKEN.
- D2HTOKEN reception is notified to the Host using the HIC_INT interrupt.

5 Example Configuration for Performance-Critical Applications

This section provides the HIC configuration that provides better performance when connected to C2000 EMIF controller. A Host side and Device side example is provided in the C2000ware Package. The Host memory controller used in this example is the F2838x EMIF connected to a F28002x HIC device.

The example projects can be accessed here:

- Host side: <C2000Ware InstallDir>\driverlib\f2838x\examples\c28x\emif\CCS\emif_ex7_16bit_asram_hic_fsi
- Device side: <C2000Ware Install Dir>\driverlib\f28002x\examples\hic\CCS\hic_ex2_config_16bit_fsi

5.1 Test Setup

Table 7 captures the HIC and EMIF configuration used in the example. The total number of pins required for this configuration is 32.

Table 7. Device and Host Configurations for Performance-Critical applications

HIC Configuration	EMIF Configuration
Access type: Direct access mode	Access Type: Normal mode
Data width: 16-bit mode	ASIZE: 16-bit data bus
Read/write: Dual pin	
Pagesel type: Register selection	

Figure 11 shows the hardware connections required for the example. The access type used in this example is direct access mode and hence the HIC_nRDY and HIC_BASEL0 pins are required to be connected. Also, the byte enables are connected to perform 8-bit accesses in the 16-bit data width mode. Note that dual pin read/write mode is chosen in the example to cover the variety of configurations and is not a mandate mode for higher performance.

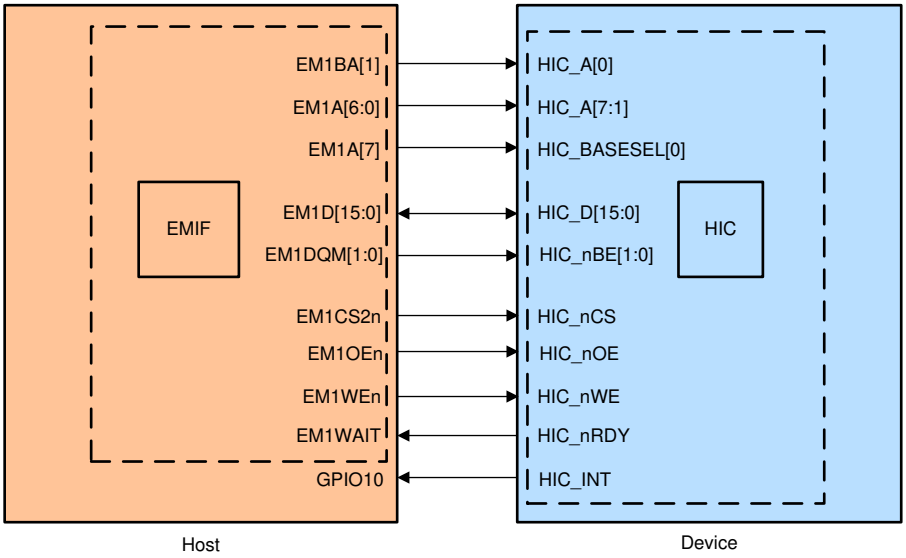

Figure 11. Hardware Setup for the Example hic_ex2_config_16bit_fsi

5.2 Test Description

- This uses direct access mode on F28002x side. The Host sends the data over FSI TX and reads the data from FSI RX.
- The Device side sets up the HIC, configures one region for FSI Base address and sends one token D2H to Host after the FSI module is configured in Loopback mode.
- The Host programs the FSI TX Buffer through direct access mode and starts the transmission
- The Received FSI RX interrupt is directly notified to Host via HIC Event Trigger Bus through HIC_INT.
- On the reception HIC_INT the Host reads the FSI RX Buffer through direct access mode.

6 Handling Device Reset and Low-Power Conditions

While the Device is undergoing a reset or if the Device is under the low power mode, the Host accesses could fail without any specific signature (writes might be ignored and reads might return undefined values). If such conditions are anticipated during the application run, the application layer must implement additional data integrity checks along with the normal accesses to ensure the Device is ready for accesses.

The data integrity checks could be implemented in the following ways:

- A simple “Interface-Active” status check, where the Host would initiate a dummy write to a H2DBUF register and read-back for correctness. If the data that is read back is the same as the written data, the Device side of the interface is active and data exchanges can be performed.
- The data payload that is being exchanged through the mailbox registers can be padded with checksum bytes and checksum verification can be performed by the receiving node.

7 References

- Texas Instruments: *TMS320F28002x Microcontrollers Data Sheet Technical Reference Manual*
- Texas Instruments: *TMS320F28002x Microcontrollers Data Sheet*
- Texas Instruments: *Design and Usage Guidelines for the C2000 External Memory Interface (EMIF)*
- Texas Instruments: *AM574x, AM576x Sitara™ Processors Silicon Revision 1.0 Technical Reference Manual*
- Texas Instruments: *Accelerators: Enhancing the Capabilities of the C2000 MCU Family Technical Brief*
- Position manager technology
- Pinmux Tool

Address Translation for Different Data Width Modes

A.1 Base Address and Offset Address Configuration

The base address configuration and the offset address to be generated by the Host varies according to the data width modes. For example, to address a region inside the Device starting from Addr[x:0], the following translation has to be used:

- Addr [x:n] should be programmed into the HICDBADDRx register
- Addr[n-1:0] should be used to generate address on the I/O HIC_A[7:0]

Where:

- n=8 for 16-bit data width mode
- n=7 for 8-bit data width mode

Table 8 captures various HICDBADDRx and I/O address offset combinations to be generated from the Host for different data width modes.

Table 8. Address Translation for Different Data Width Modes

Address to be Accessed	Data Width = 16		Data Width = 8	
	HICDBADDRx	I/O Address Offset	HICDBADDRx	I/O Address Offset
0x6604	0x6600	0x4	0x6600	0x4
0x6688	0x6600	0x88	0x6680	0x8
0xFFFF0	0xFF00	0xF0	0xFF80	0x7F

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale (www.ti.com/legal/termsofsale.html) or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

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