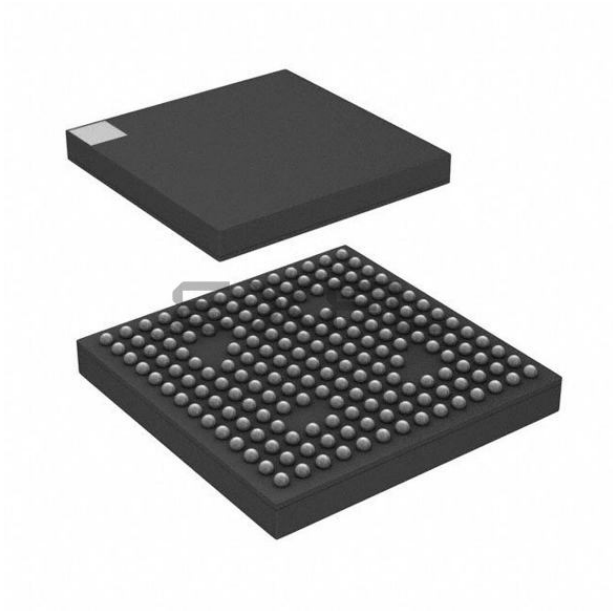
Analog Devices Inc. - ADSP-BF701KBCZ-1 Datasheet



Welcome to E-XFL.COM

Understanding Embedded - DSP (Digital Signal Processors)

Embedded - DSP (Digital Signal Processors) are specialized microprocessors designed to perform complex mathematical computations on digital signals in real-time. Unlike general-purpose processors, DSPs are optimized for high-speed numeric processing tasks, making them ideal for applications that require efficient and precise manipulation of digital data. These processors are fundamental in converting and processing signals in various forms, including audio, video, and communication signals, ensuring that data is accurately interpreted and utilized in embedded systems.

Applications of Embedded - DSP (Digital Signal Processors)

Details	
Product Status	Active
Type	Blackfin+
Interface	CAN, DSPI, EBI/EMI, I²C, PPI, QSPI, SD/SDIO, SPI, SPORT, UART/USART, USB OTG
Clock Rate	100MHz
Non-Volatile Memory	ROM (512kB)
On-Chip RAM	128kB
Voltage - I/O	1.8V, 3.3V
Voltage - Core	1.10V
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	184-LFBGA, CSPBGA
Supplier Device Package	184-CSPBGA (12x12)
Purchase URL	https://www.e-xfl.com/product-detail/analog-devices/adsp-bf701kbcz-1

The program sequencer controls the flow of instruction execution, including instruction alignment and decoding. For program flow control, the sequencer supports PC relative and indirect conditional jumps (with dynamic branch prediction), and subroutine calls. Hardware supports zero-overhead looping. The architecture is fully interlocked, meaning that the programmer need not manage the pipeline when executing instructions with data dependencies.

The address arithmetic unit provides two addresses for simultaneous dual fetches from memory. It contains a multiported register file consisting of four sets of 32-bit index, modify, length, and base registers (for circular buffering), and eight additional 32-bit pointer registers (for C-style indexed stack manipulation).

The Blackfin processor supports a modified Harvard architecture in combination with a hierarchical memory structure. Level 1 (L1) memories are those that typically operate at the full processor speed with little or no latency. At the L1 level, the instruction memory holds instructions only. The data memory holds data, and a dedicated scratchpad data memory stores stack and local variable information.

In addition, multiple L1 memory blocks are provided, offering a configurable mix of SRAM and cache. The memory management unit (MMU) provides memory protection for individual tasks that may be operating on the core and can protect system registers from unintended access.

The architecture provides three modes of operation: user mode, supervisor mode, and emulation mode. User mode has restricted access to certain system resources, thus providing a protected software environment, while supervisor mode has unrestricted access to the system and core resources.

INSTRUCTION SET DESCRIPTION

The Blackfin processor instruction set has been optimized so that 16-bit opcodes represent the most frequently used instructions, resulting in excellent compiled code density. Complex DSP instructions are encoded into 32-bit opcodes, representing fully featured multifunction instructions. The Blackfin processor supports a limited multi-issue capability, where a 32-bit instruction can be issued in parallel with two 16-bit instructions, allowing the programmer to use many of the core resources in a single instruction cycle.

The Blackfin processor family assembly language instruction set employs an algebraic syntax designed for ease of coding and readability. The instructions have been specifically tuned to provide a flexible, densely encoded instruction set that compiles to a very small final memory size. The instruction set also provides fully featured multifunction instructions that allow the programmer to use many of the processor core resources in a single instruction. Coupled with many features more often seen on microcontrollers, this instruction set is very efficient when compiling C and C++ source code. In addition, the architecture supports both user (algorithm/application code) and supervisor (O/S kernel, device drivers, debuggers, ISRs) modes of operation, allowing multiple levels of access to core processor resources.

The assembly language, which takes advantage of the processor's unique architecture, offers the following advantages:

- Seamlessly integrated DSP/MCU features are optimized for both 8-bit and 16-bit operations.
- A multi-issue load/store modified-Harvard architecture, which supports two 16-bit MAC or four 8-bit ALU + two load/store + two pointer updates per cycle.
- All registers, I/O, and memory are mapped into a unified 4G byte memory space, providing a simplified programming model.
- Control of all asynchronous and synchronous events to the processor is handled by two subsystems: the core event controller (CEC) and the system event controller (SEC).
- Microcontroller features, such as arbitrary bit and bit-field manipulation, insertion, and extraction; integer operations on 8-, 16-, and 32-bit data-types; and separate user and supervisor stack pointers.
- Code density enhancements, which include intermixing of 16-bit and 32-bit instructions (no mode switching, no code segregation). Frequently used instructions are encoded in 16 bits.

PROCESSOR INFRASTRUCTURE

The following sections provide information on the primary infrastructure components of the ADSP-BF70x processor.

DMA Controllers

The processor uses direct memory access (DMA) to transfer data within memory spaces or between a memory space and a peripheral. The processor can specify data transfer operations and return to normal processing while the fully integrated DMA controller carries out the data transfers independent of processor activity.

DMA transfers can occur between memory and a peripheral or between one memory and another memory. Each memory-to-memory DMA stream uses two channels, where one channel is the source channel, and the second is the destination channel.

All DMAs can transport data to and from all on-chip and off-chip memories. Programs can use two types of DMA transfers, descriptor-based or register-based. Register-based DMA allows the processor to directly program DMA control registers to initiate a DMA transfer. On completion, the control registers may be automatically updated with their original setup values for continuous transfer. Descriptor-based DMA transfers require a set of parameters stored within memory to initiate a DMA sequence. Descriptor-based DMA transfers allow multiple DMA sequences to be chained together and a DMA channel can be programmed to automatically set up and start another DMA transfer after the current sequence completes.

The DMA controller supports the following DMA operations.

- A single linear buffer that stops on completion.
- A linear buffer with negative, positive, or zero stride length.
- A circular, auto-refreshing buffer that interrupts when each buffer becomes full.

output enable and the input enable of a GPIO pin are both active, the data signal before the pad driver is looped back to the receive path for the same GPIO pin.

MEMORY ARCHITECTURE

The processor views memory as a single unified 4G byte address space, using 32-bit addresses. All resources, including internal memory, external memory, and I/O control registers, occupy separate sections of this common address space. The memory portions of this address space are arranged in a hierarchical structure to provide a good cost/performance balance of some very fast, low-latency core-accessible memory as cache or SRAM, and larger, lower-cost and performance interface-accessible memory systems. See Figure 3.

Internal (Core-Accessible) Memory

The L1 memory system is the highest-performance memory available to the Blackfin+ processor core.

The core has its own private L1 memory. The modified Harvard architecture supports two concurrent 32-bit data accesses along with an instruction fetch at full processor speed which provides high-bandwidth processor performance. In the core, a 64K byte block of data memory partners with an 64K byte memory block for instruction storage. Each data block is multibanked for efficient data exchange through DMA and can be configured as SRAM. Alternatively, 16K bytes of each block can be configured in L1 cache mode. The four-way set-associative instruction cache and the 2 two-way set-associative data caches greatly accelerate memory access performance, especially when accessing external memories.

The L1 memory domain also features a 8K byte data SRAM block which is ideal for storing local variables and the software stack. All L1 memory is protected by a multi-parity-bit concept, regardless of whether the memory is operating in SRAM or cache mode.

Outside of the L1 domain, L2 and L3 memories are arranged using a Von Neumann topology. The L2 memory domain is a unified instruction and data memory and can hold any mixture of code and data required by the system design. The L2 memory domain is accessible by the Blackfin+ core through a dedicated 64-bit interface. It operates at SYSCLK frequency.

The processor features up to 1M byte of L2 SRAM, which is ECC-protected and organized in eight banks. Individual banks can be made private to any system master. There is also a 512K byte single-bank ROM in the L2 domain. It contains boot code, security code, and general-purpose ROM space.

OTP Memory

The processor features 4 kB of one-time-programmable (OTP) memory which is memory-map accessible. This memory stores a unique chip identification and is used to support secure-boot and secure operation.

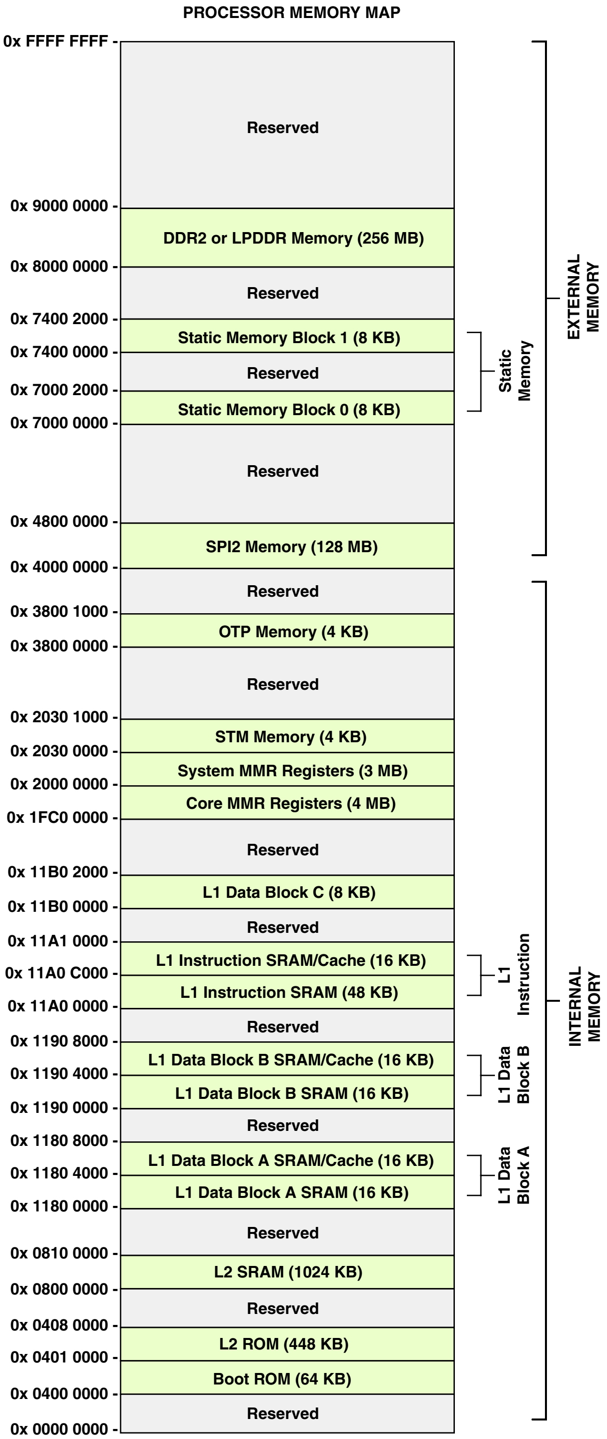


Figure 3. ADSP-BF706/ADSP-BF707 Internal/External Memory Map

The CAN controller offers the following features:

- 32 mailboxes (8 receive only, 8 transmit only, 16 configurable for receive or transmit)
- Dedicated acceptance masks for each mailbox
- Additional data filtering on first two bytes
- Support for both the standard (11-bit) and extended (29-bit) identifier (ID) message formats
- Support for remote frames
- Active or passive network support
- CAN wake-up from hibernation mode (lowest static power consumption mode)
- Interrupts, including: TX complete, RX complete, error and global

An additional crystal is not required to supply the CAN clock, as the CAN clock is derived from a system clock through a programmable divider.

USB 2.0 On-the-Go Dual-Role Device Controller

The USB 2.0 on-the-go (OTG) dual-role device controller provides a low-cost connectivity solution for the growing adoption of this bus standard in industrial applications, as well as consumer mobile devices such as cell phones, digital still cameras, and MP3 players. The USB 2.0 controller allows these devices to transfer data using a point-to-point USB connection without the need for a PC host. The module can operate in a traditional USB peripheral-only mode as well as the host mode presented in the OTG supplement to the USB 2.0 specification.

The USB clock is provided through a dedicated external crystal or crystal oscillator.

The USB OTG dual-role device controller includes a phase locked loop with programmable multipliers to generate the necessary internal clocking frequency for USB.

Housekeeping ADC (HADC)

The HADC provides a general-purpose, multichannel successive approximation analog-to-digital converter. It supports the following features:

- 12-bit ADC core (10-bit accuracy) with built-in sample and hold
- 4 single-ended input channels
- Throughput rates up to 1 MSPS
- Single external reference with analog inputs between 0 V and 3.3 V
- Selectable ADC clock frequency including the ability to program a prescaler
- Adaptable conversion type: allows single or continuous conversion with option of autoscan
- Auto sequencing capability with up to 4 autoconversions in a single session. Each conversion can be programmed to select any input channel.
- Four data registers (individually addressable) to store conversion values

System Crossbars (SCB)

The system crossbars (SCB) are the fundamental building blocks of a switch-fabric style for (on-chip) system bus interconnection. The SCBs connect system bus masters to system bus slaves, providing concurrent data transfer between multiple bus masters and multiple bus slaves. A hierarchical model—built from multiple SCBs—provides a power and area efficient system interconnect, which satisfies the performance and flexibility requirements of a specific system.

The SCBs provide the following features:

- Highly efficient, pipelined bus transfer protocol for sustained throughput
- Full-duplex bus operation for flexibility and reduced latency
- Concurrent bus transfer support to allow multiple bus masters to access bus slaves simultaneously
- Protection model (privileged/secure) support for selective bus interconnect protection

POWER AND CLOCK MANAGEMENT

The processor provides three operating modes, each with a different performance/power profile. Control of clocking to each of the processor peripherals also reduces power consumption. See Table 5 for a summary of the power settings for each mode.

System Crystal Oscillator and USB Crystal Oscillator

The processor can be clocked by an external crystal (see Figure 4), a sine wave input, or a buffered, shaped clock derived from an external clock oscillator. If an external clock is used, it should be a TTL compatible signal and must not be halted, changed, or operated below the specified frequency during normal operation. This signal is connected to the SYS_CLKIN pin of the processor. When an external clock is used, the SYS_XTAL pin must be left unconnected. Alternatively, because the processor includes an on-chip oscillator circuit, an external crystal may be used.

For fundamental frequency operation, use the circuit shown in Figure 4. A parallel-resonant, fundamental frequency, microprocessor grade crystal is connected across the SYS_CLKIN and SYS_XTAL pins. The on-chip resistance between SYS_CLKIN and the SYS_XTAL pin is in the 500 kΩ range. Further parallel resistors are typically not recommended.

The two capacitors and the series resistor shown in Figure 4 fine-tune phase and amplitude of the sine frequency. The capacitor and resistor values shown in Figure 4 are typical values only. The capacitor values are dependent upon the load capacitance recommendations of the crystal manufacturer and the PCB physical layout. The resistor value depends on the drive

Table 6. ADSP-BF70x Detailed Signal Descriptions (Continued)

Port Name	Direction	Description
SPT_BCLK	I/O	**Channel B Clock.** Data and Frame Sync are driven/sampled with respect to this clock. This signal can be either internally or externally generated.
SPT_BD0	I/O	**Channel B Data 0.** Primary bidirectional data I/O. This signal can be configured as an output to transmit serial data, or as an input to receive serial data.
SPT_BD1	I/O	**Channel B Data 1.** Secondary bidirectional data I/O. This signal can be configured as an output to transmit serial data, or as an input to receive serial data.
SPT_BFS	I/O	**Channel B Frame Sync.** The frame sync pulse initiates shifting of serial data. This signal is either generated internally or externally.
SPT_BTDV	Output	**Channel B Transmit Data Valid.** This signal is optional and only active when SPORT is configured in multi-channel transmit mode. It is asserted during enabled slots.
SYS_BMODEn	Input	**Boot Mode Control n.** Selects the boot mode of the processor.
SYS_CLKIN	Input	**Clock/Crystal Input.** Connect to an external clock source or crystal.
SYS_CLKOUT	Output	**Processor Clock Output.** Outputs internal clocks. Clocks may be divided down. See the CGU chapter of the HRM for more details.
SYS_EXTWAKE	Output	**External Wake Control.** Drives low during hibernate and high all other times. Typically connected to the enable input of the voltage regulator controlling the VDD_INT supply.
$\overline{\text{SYS\_FAULT}}$	I/O	**Active-Low Fault Output.** Indicates internal faults or senses external faults depending on the operating mode.
$\overline{\text{SYS\_HWRST}}$	Input	**Processor Hardware Reset Control.** Resets the device when asserted.
$\overline{\text{SYS\_NMI}}$	Input	**Non-maskable Interrupt.** See the processor hardware and programming references for more details.
$\overline{\text{SYS\_RESOUT}}$	Output	**Reset Output.** Indicates that the device is in the reset or hibernate state.
SYS_WAKEn	Input	**Power Saving Mode Wakeup n.** Wake-up source input for deep sleep and/or hibernate mode.
SYS_XTAL	Output	**Crystal Output.** Drives an external crystal. Must be left unconnected if an external clock is driving CLKIN.
JTG_SWCLK	I/O	**Serial Wire Clock.** Clocks data into and out of the target during debug.
JTG_SWDIO	I/O	**Serial Wire DIO.** Sends and receives serial data to and from the target during debug.
JTG_SWO	Output	**Serial Wire Out.** Provides trace data to the emulator.
JTG_TCK	Input	**JTAG Clock.** JTAG test access port clock.
JTG_TDI	Input	**JTAG Serial Data In.** JTAG test access port data input.
JTG_TDO	Output	**JTAG Serial Data Out.** JTAG test access port data output.
JTG_TMS	Input	**JTAG Mode Select.** JTAG test access port mode select.
$\overline{\text{JTG\_TRST}}$	Input	**JTAG Reset.** JTAG test access port reset.
TM_ACIn	Input	**Alternate Capture Input n.** Provides an additional input for WIDCAP, WATCHDOG, and PININT modes.
TM_ACLKn	Input	**Alternate Clock n.** Provides an additional time base for use by an individual timer.
TM_CLK	Input	**Clock.** Provides an additional global time base for use by all the GP timers.
TM_TMRn	I/O	**Timer n.** The main input/output signal for each timer.
TRACE_CLK	Output	**Trace Clock.** Clock output.
TRACE_Dnn	Output	**Trace Data n.** Unidirectional data bus.
TWI_SCL	I/O	**Serial Clock.** Clock output when master, clock input when slave.
TWI_SDA	I/O	**Serial Data.** Receives or transmits data.
$\overline{\text{UART\_CTS}}$	Input	**Clear to Send.** Flow control signal.
$\overline{\text{UART\_RTS}}$	Output	**Request to Send.** Flow control signal.
$\overline{\text{UART\_RX}}$	Input	**Receive.** Receive input. Typically connects to a transceiver that meets the electrical requirements of the device being communicated with.
$\overline{\text{UART\_TX}}$	Output	**Transmit.** Transmit output. Typically connects to a transceiver that meets the electrical requirements of the device being communicated with.
USB_CLKIN	Input	**Clock/Crystal Input.** This clock input is multiplied by a PLL to form the USB clock. See data sheet specifications for frequency/tolerance information.

Table 6. ADSP-BF70x Detailed Signal Descriptions (Continued)

Port Name	Direction	Description
USB_DM	I/O	**Data –.** Bidirectional differential data line.
USB_DP	I/O	**Data +.** Bidirectional differential data line.
USB_ID	Input	**OTG ID.** Senses whether the controller is a host or device. This signal is pulled low when an A-type plug is sensed (signifying that the USB controller is the A device), but the input is high when a B-type plug is sensed (signifying that the USB controller is the B device).
USB_VBC	Output	**VBUS Control.** Controls an external voltage source to supply VBUS when in host mode. May be configured as open-drain. Polarity is configurable as well.
USB_VBUS	I/O	**Bus Voltage.** Connects to bus voltage in host and device modes.
USB_XTAL	Output	**Crystal.** Drives an external crystal. Must be left unconnected if an external clock is driving USB_CLKIN.

Table 7. ADSP-BF70x 184-Ball CSP_BGA Signal Descriptions (Continued)

Signal Name	Description	Port	Pin Name
SYS_FAULT	Active-Low Fault Output	Not Muxed	SYS_FAULT
SYS_HWRST	Processor Hardware Reset Control	Not Muxed	SYS_HWRST
SYS_NMI	Nonmaskable Interrupt	Not Muxed	SYS_NMI
SYS_RESOUT	Reset Output	Not Muxed	SYS_RESOUT
SYS_WAKE0	Power Saving Mode Wake-up 0	B	PB_07
SYS_WAKE1	Power Saving Mode Wake-up 1	B	PB_08
SYS_WAKE2	Power Saving Mode Wake-up 2	B	PB_12
SYS_WAKE3	Power Saving Mode Wake-up 3	C	PC_02
SYS_WAKE4	Power Saving Mode Wake-up 4	A	PA_12
SYS_XTAL	Crystal Output	Not Muxed	SYS_XTAL
TM0_ACI0	TIMER0 Alternate Capture Input 0	C	PC_03
TM0_ACI1	TIMER0 Alternate Capture Input 1	B	PB_01
TM0_ACI2	TIMER0 Alternate Capture Input 2	C	PC_07
TM0_ACI3	TIMER0 Alternate Capture Input 3	B	PB_09
TM0_ACI4	TIMER0 Alternate Capture Input 4	C	PC_01
TM0_ACI5	TIMER0 Alternate Capture Input 5	C	PC_02
TM0_ACI6	TIMER0 Alternate Capture Input 6	A	PA_12
TM0_ACLK0	TIMER0 Alternate Clock 0	C	PC_04
TM0_ACLK1	TIMER0 Alternate Clock 1	C	PC_10
TM0_ACLK2	TIMER0 Alternate Clock 2	C	PC_09
TM0_ACLK3	TIMER0 Alternate Clock 3	B	PB_00
TM0_ACLK4	TIMER0 Alternate Clock 4	B	PB_10
TM0_ACLK5	TIMER0 Alternate Clock 5	A	PA_14
TM0_ACLK6	TIMER0 Alternate Clock 6	B	PB_04
TM0_CLK	TIMER0 Clock	B	PB_06
TM0_TMR0	TIMER0 Timer 0	A	PA_05
TM0_TMR1	TIMER0 Timer 1	A	PA_06
TM0_TMR2	TIMER0 Timer 2	A	PA_07
TM0_TMR3	TIMER0 Timer 3	C	PC_05
TM0_TMR4	TIMER0 Timer 4	A	PA_09
TM0_TMR5	TIMER0 Timer 5	A	PA_10
TM0_TMR6	TIMER0 Timer 6	A	PA_11
TM0_TMR7	TIMER0 Timer 7	A	PA_04
TRACE0_CLK	TPIU0 Trace Clock	B	PB_10
TRACE0_D00	TPIU0 Trace Data 0	B	PB_15
TRACE0_D01	TPIU0 Trace Data 1	B	PB_14
TRACE0_D02	TPIU0 Trace Data 2	B	PB_13
TRACE0_D03	TPIU0 Trace Data 3	B	PB_12
TRACE0_D04	TPIU0 Trace Data 4	B	PB_11
TRACE0_D05	TPIU0 Trace Data 5	A	PA_02
TRACE0_D06	TPIU0 Trace Data 6	A	PA_01
TRACE0_D07	TPIU0 Trace Data 7	A	PA_00
TWI0_SCL	TWI0 Serial Clock	Not Muxed	TWI0_SCL
TWI0_SDA	TWI0 Serial Data	Not Muxed	TWI0_SDA
UART0_CTS	UART0 Clear to Send	C	PC_03
UART0_RTS	UART0 Request to Send	C	PC_02

Table 15. ADSP-BF70x Designer Quick Reference (Continued)

Signal Name	Type	Driver Type	Int Term	Reset Term	Reset Drive	Hiber Term	Hiber Drive	Power Domain	Description and Notes
PC_05	I/O	A	none	none	none	none	none	VDD_EXT	Desc: SPT0 Channel A Frame Sync | TM0 Timer 3 | MSI0 Command Notes: An external pull-up may be required for MSI modes, see the MSI chapter in the hardware reference for details.
PC_06	I/O	A	none	none	none	none	none	VDD_EXT	Desc: SPT0 Channel B Data 0 | SPI0 Master In, Slave Out | MSI0 Data 3 Notes: An external pull-up may be required for MSI modes, see the MSI chapter in the hardware reference for details.
PC_07	I/O	A	none	none	none	none	none	VDD_EXT	Desc: SPT0 Channel B Frame Sync | SPI0 Master Out, Slave In | MSI0 Data 2 | TM0 Alternate Capture Input 2 Notes: An external pull-up may be required for MSI modes, see the MSI chapter in the hardware reference for details.
PC_08	I/O	A	none	none	none	none	none	VDD_EXT	Desc: SPT0 Channel A Data 0 | SPI0 Data 2 | MSI0 Data 0 Notes: An external pull-up may be required for MSI modes, see the MSI chapter in the hardware reference for details.
PC_09	I/O	A	none	none	none	none	none	VDD_EXT	Desc: SPT0 Channel A Clock | SPI0 Data 3 | MSI0 Clock | TM0 Alternate Clock 2 Notes: No notes.
PC_10	I/O	A	none	none	none	none	none	VDD_EXT	Desc: SPT1 Channel B Clock | MSI0 Data 4 | SPI1 Slave Select Output 3 | TM0 Alternate Clock 1 Notes: An external pull-up may be required for MSI modes, see the MSI chapter in the hardware reference for details. SPI slave select outputs require a pull-up when used.
PC_11	I/O	A	none	none	none	none	none	VDD_EXT	Desc: SPT1 Channel B Frame Sync | MSI0 Data 5 | SPI0 Slave Select Output 3 Notes: An external pull-up may be required for MSI modes, see the MSI chapter in the hardware reference for details. SPI slave select outputs require a pull-up when used.
PC_12	I/O	A	none	none	none	none	none	VDD_EXT	Desc: SPT1 Channel B Data 0 | MSI0 Data 6 Notes: An external pull-up may be required for MSI modes, see the MSI chapter in the hardware reference for details.

Table 15. ADSP-BF70x Designer Quick Reference (Continued)

Signal Name	Type	Driver Type	Int Term	Reset Term	Reset Drive	Hiber Term	Hiber Drive	Power Domain	Description and Notes
VDD_OTP	s	na	none	none	none	none	none	na	Desc: VDD for OTP Notes: Must be powered.
VDD_RTC	s	na	none	none	none	none	none	na	Desc: VDD for RTC Notes: If RTC is not used, connect to ground.
VDD_USB	s	na	none	none	none	none	none	na	Desc: VDD for USB Notes: If USB is not used, connect to VDD_EXT.

ELECTRICAL CHARACTERISTICS

Parameter		Test Conditions/Comments	Min	Typ	Max	Unit
V_{OH}[1]	High Level Output Voltage	V_{DD_EXT} = 1.7 V, I_{OH} = –1.0 mA	0.8 × V_{DD_EXT}			V
V_{OH}[1]	High Level Output Voltage	V_{DD_EXT} = 3.13 V, I_{OH} = –2.0 mA	0.9 × V_{DD_EXT}			V
V_{OH_DDR2}[2]	High Level Output Voltage, DDR2, Programmed Impedance = 34 Ω	V_{DD_DMC} = 1.70 V, I_{OH} = –7.1 mA	V_{DD_DMC} – 0.320			V
V_{OH_DDR2}[2]	High Level Output Voltage, DDR2, Programmed Impedance = 40 Ω	V_{DD_DMC} = 1.70 V, I_{OH} = –5.8 mA	V_{DD_DMC} – 0.320			V
V_{OH_DDR2}[2]	High Level Output Voltage, DDR2, Programmed Impedance = 50 Ω	V_{DD_DMC} = 1.70 V, I_{OH} = –4.1 mA	V_{DD_DMC} – 0.320			V
V_{OH_DDR2}[2]	High Level Output Voltage, DDR2, Programmed Impedance = 60 Ω	V_{DD_DMC} = 1.70 V, I_{OH} = –3.4 mA	V_{DD_DMC} – 0.320			V
V_{OH_LPDDR}[2]	High Level Output Voltage, LPDDR	V_{DD_DMC} = 1.70 V, I_{OH} = –2.0 mA	V_{DD_DMC} – 0.320			V
V_{OL}[3]	Low Level Output Voltage	V_{DD_EXT} = 1.7 V, I_{OL} = 1.0 mA			0.400	V
V_{OL}[3]	Low Level Output Voltage	V_{DD_EXT} = 3.13 V, I_{OL} = 2.0 mA			0.400	V
V_{OL_DDR2}[2]	Low Level Output Voltage, DDR2, Programmed Impedance = 34 Ω	V_{DD_DMC} = 1.70 V, I_{OL} = 7.1 mA			0.320	V
V_{OL_DDR2}[2]	Low Level Output Voltage, DDR2, Programmed Impedance = 40 Ω	V_{DD_DMC} = 1.70 V, I_{OL} = 5.8 mA			0.320	V
V_{OL_DDR2}[2]	Low Level Output Voltage, DDR2, Programmed Impedance = 50 Ω	V_{DD_DMC} = 1.70 V, I_{OL} = 4.1 mA			0.320	V
V_{OL_DDR2}[2]	Low Level Output Voltage, DDR2, Programmed Impedance = 60 Ω	V_{DD_DMC} = 1.70 V, I_{OL} = 3.4 mA			0.320	V
V_{OL_LPDDR}[2]	Low Level Output Voltage, LPDDR	V_{DD_DMC} = 1.70 V, I_{OL} = 2.0 mA			0.320	V
I_{IH}[4]	High Level Input Current	V_{DD_EXT} = 3.47 V, V_{DD_DMC} = 1.9 V, V_{DD_USB} = 3.47 V, V_{IN} = 3.47 V			10	μA
$I_{IH\_DMC0\_VREF}$[5]	High Level Input Current	V_{DD_EXT} = 3.47 V, V_{DD_DMC} = 1.9 V, V_{DD_USB} = 3.47 V, V_{IN} = 3.47 V			1	μA
I_{IH_PD}[6]	High Level Input Current with Pull-down Resistor	V_{DD_EXT} = 3.47 V, V_{DD_DMC} = 1.9 V, V_{DD_USB} = 3.47 V, V_{IN} = 3.47 V			100	μA
R_{PD}[6]	Internal Pull-down Resistance	V_{DD_EXT} = 3.47 V, V_{DD_DMC} = 1.9 V, V_{DD_USB} = 3.47 V, V_{IN} = 3.47 V	57		130	kΩ
I_{IL}[7]	Low Level Input Current	V_{DD_EXT} = 3.47 V, V_{DD_DMC} = 1.9 V, V_{DD_USB} = 3.47 V, V_{IN} = 0 V			10	μA
$I_{IL\_DMC0\_VREF}$[5]	Low Level Input Current	V_{DD_EXT} = 3.47 V, V_{DD_DMC} = 1.9 V, V_{DD_USB} = 3.47 V, V_{IN} = 0 V			1	μA
I_{IL_PU}[8]	Low Level Input Current with Pull-up Resistor	V_{DD_EXT} = 3.47 V, V_{DD_DMC} = 1.9 V, V_{DD_USB} = 3.47 V, V_{IN} = 0 V			100	μA
R_{PU}[8]	Internal Pull-up Resistance	V_{DD_EXT} = 3.47 V, V_{DD_DMC} = 1.9 V, V_{DD_USB} = 3.47 V, V_{IN} = 0 V	53		129	kΩ
I_{IH_USB0}[9]	High Level Input Current	V_{DD_EXT} = 3.47 V, V_{DD_DMC} = 1.9 V, V_{DD_USB} = 3.47 V, V_{IN} = 3.47 V			10	μA
I_{IL_USB0}[9]	Low Level Input Current	V_{DD_EXT} = 3.47 V, V_{DD_DMC} = 1.9 V, V_{DD_USB} = 3.47 V, V_{IN} = 0 V			10	μA
I_{OZH}[10]	Three-State Leakage Current	V_{DD_EXT} = 3.47 V, V_{DD_DMC} = 1.9 V, V_{DD_USB} = 3.47 V, V_{IN} = 3.47 V			10	μA
I_{OZH}[11]	Three-State Leakage Current	V_{DD_EXT} = 3.47 V, V_{DD_DMC} = 1.9 V, V_{DD_USB} = 3.47 V, V_{IN} = 1.9 V			10	μA
I_{OZL}[12]	Three-State Leakage Current	V_{DD_EXT} = 3.47 V, V_{DD_DMC} = 1.9 V, V_{DD_USB} = 3.47 V, V_{IN} = 0 V			10	μA
I_{OZH_PD}[13]	Three-State Leakage Current	V_{DD_EXT} = 3.47 V, V_{DD_DMC} = 1.9 V, V_{DD_USB} = 3.47 V, V_{IN} = 3.47 V			100	μA

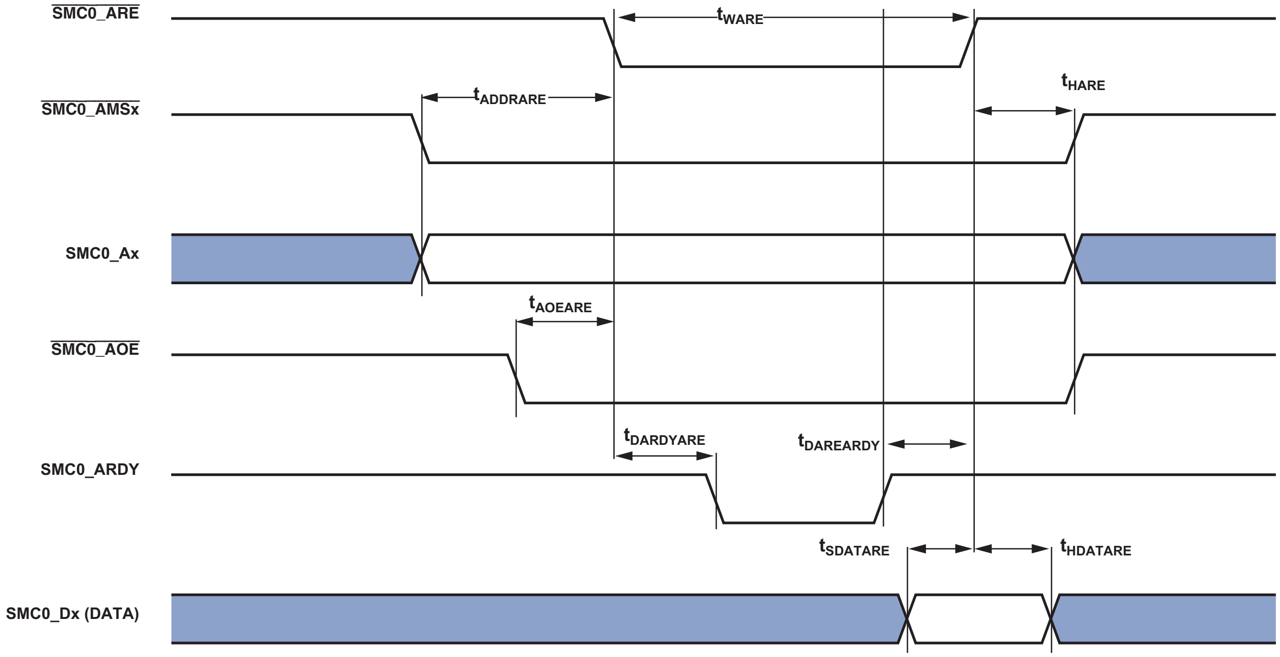


Figure 10. Asynchronous Read

SMC Read Cycle Timing With Reference to SYS_CLKOUT

The following SMC specifications with respect to SYS_CLKOUT are given to accommodate the connection of the SMC to programmable logic devices. These specifications assume that SYS_CLKOUT is outputting a buffered version of SCLK0 by setting CGU_CLKOUTSEL.CLKOUTSEL = 0x3. However, SCLK0 must not run faster than the maximum f_{OCLK} specification. For this example, RST = 0x2, RAT = 0x4, and RHT = 0x1.

Table 32. SMC Read Cycle Timing With Reference to SYS_CLKOUT (BxMODE = b#00)

		V_{DD_EXT} 1.8 V Nominal		V_{DD_EXT} 3.3 V Nominal		
Parameter		**Min**	**Max**	**Min**	**Max**	**Unit**
Timing Requirements						
t_{SDAT}	SMC0_Dx Setup Before SYS_CLKOUT	5.3		4.3		ns
t_{HDAT}	SMC0_Dx Hold After SYS_CLKOUT	1.5		1.5		ns
t_{SARDY}	SMC0_ARDY Setup Before SYS_CLKOUT	16.6		14.4		ns
t_{HARDY}	SMC0_ARDY Hold After SYS_CLKOUT	0.7		0.7		ns
Switching Characteristics						
t_{DO}	Output Delay After SYS_CLKOUT[1]		7		7	ns
t_{HO}	Output Hold After SYS_CLKOUT [1]	–2.5		–2.5		ns

[1] Output signals are SMC0_Ax, $\overline{\text{SMC0\_AMSx}}$, $\overline{\text{SMC0\_AOE}}$, and $\overline{\text{SMC0\_ABEx}}$.

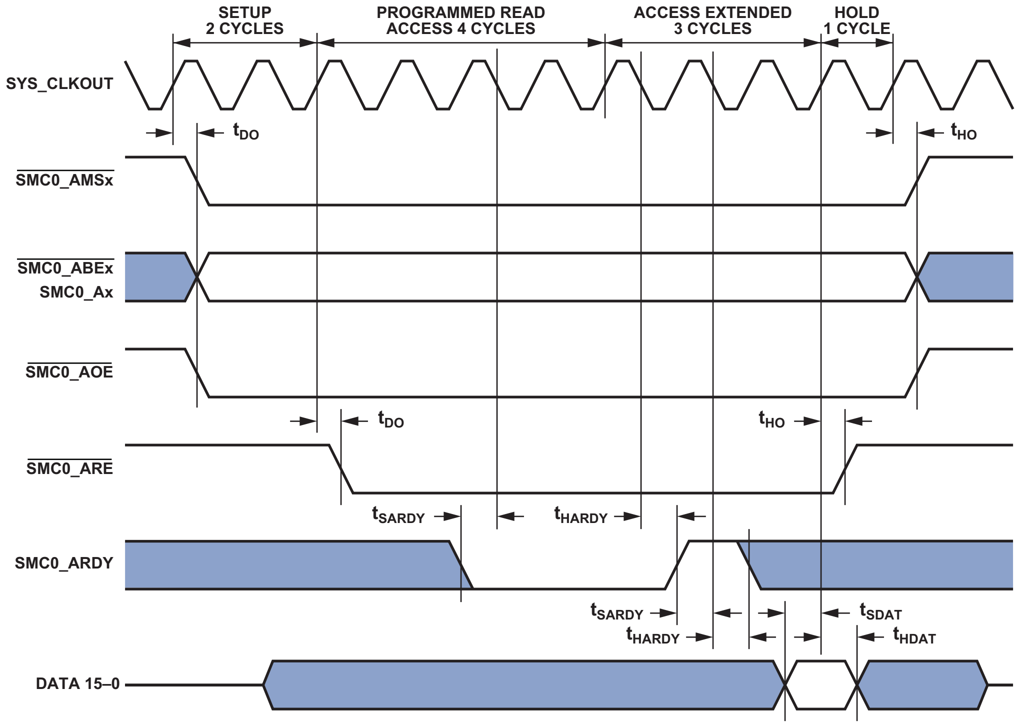


Figure 11. Asynchronous Memory Read Cycle Timing

Asynchronous Write

Table 35 and Figure 14 show asynchronous memory write timing, related to the static memory controller (SMC).

Table 35. Asynchronous Memory Write (BxMODE = b#00)

		V_{DD_EXT} 1.8 V Nominal		V_{DD_EXT} 3.3 V Nominal		
Parameter		**Min**	**Max**	**Min**	**Max**	**Unit**
Timing Requirement						
$t_{DARDYAWE}$[1]	SMC0_ARDY Valid After $\overline{\text{SMC0\_AWE}}$ Low[2]		(WAT – 2.5) × t_{SCLK0} – 17.5		(WAT – 2.5) × t_{SCLK0} – 17.5	ns
Switching Characteristics						
t_{ENDAT}	DATA Enable After $\overline{\text{SMC0\_AMSx}}$ Assertion	–3		–2		ns
t_{DDAT}	DATA Disable After $\overline{\text{SMC0\_AMSx}}$ Deassertion		4.5		4	ns
t_{AMSAWE}	SMC0_Ax/$\overline{\text{SMC0\_AMSx}}$ Assertion Before $\overline{\text{SMC0\_AWE}}$ Low[3]	(PREST + WST + PREAT) × t_{SCLK0} – 2		(PREST + WST + PREAT) × t_{SCLK0} – 4		ns
t_{HAWE}	Output[4] Hold After $\overline{\text{SMC0\_AWE}}$ High[5]	WHT × t_{SCLK0}		WHT × t_{SCLK0}		ns
t_{WAWE}[6]	$\overline{\text{SMC0\_AWE}}$ Active Low Width[6]	WAT × t_{SCLK0} – 2		WAT × t_{SCLK0} – 2		ns
$t_{DAWEARDY}$[1]	$\overline{\text{SMC0\_AWE}}$ High Delay After SMC0_ARDY Assertion		3.5 × t_{SCLK0} + 17.5		3.5 × t_{SCLK0} + 17.5	ns

[1] SMC_BxCTL.ARDYEN bit = 1.

[2] WAT value set using the SMC_BxTIM.WAT bits.

[3] PREST, WST, PREAT values set using the SMC_BxETIM.PREST bits, SMC_BxTIM.WST bits, SMC_BxETIM.PREAT bits, and the SMC_BxTIM.RAT bits.

[4] Output signals are DATA, SMC0_Ax, $\overline{\text{SMC0\_AMSx}}$, $\overline{\text{SMC0\_ABEx}}$.

[5] WHT value set using the SMC_BxTIM.WHT bits.

[6] SMC_BxCTL.ARDYEN bit = 0.

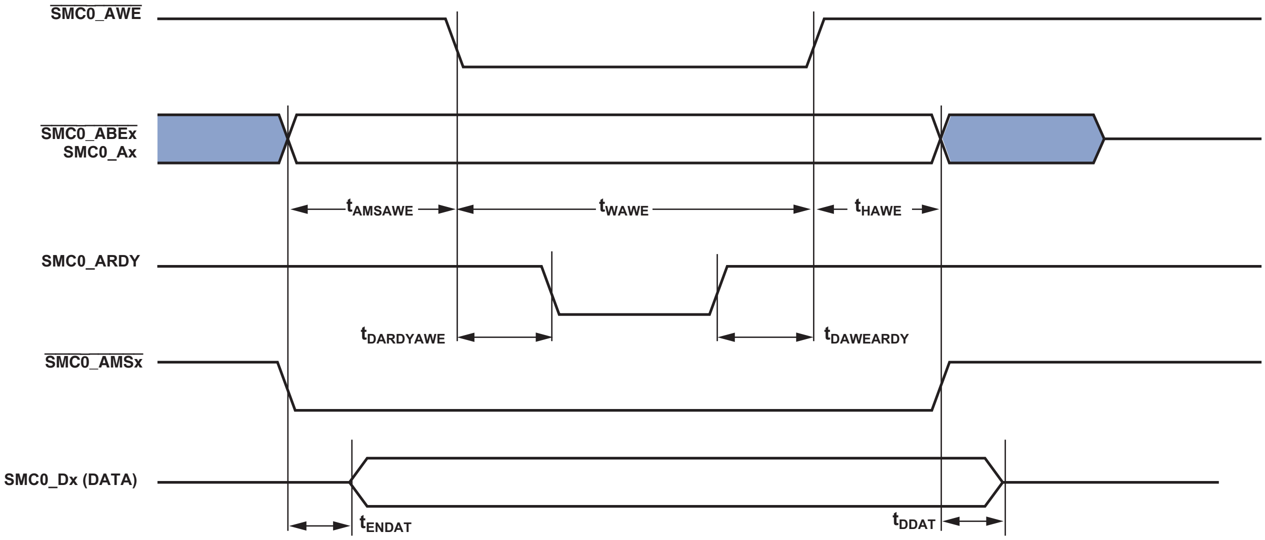


Figure 14. Asynchronous Write

Debug Interface (JTAG Emulation Port) Timing

Table 48 and Figure 26 provide I/O timing, related to the debug interface (JTAG emulator port).

Table 48. JTAG Port Timing

		V_{DD_EXT} 1.8 V Nominal		V_{DD_EXT} 3.3 V Nominal		
Parameter		**Min**	**Max**	**Min**	**Max**	**Unit**
Timing Requirements						
t_{TCK}	JTG_TCK Period	20		20		ns
t_{STAP}	JTG_TDI, JTG_TMS Setup Before JTG_TCK High	5		4		ns
t_{HTAP}	JTG_TDI, JTG_TMS Hold After JTG_TCK High	4		4		ns
t_{SSYS}	System Inputs Setup Before JTG_TCK High[1]	4		4		ns
t_{HSYS}	System Inputs Hold After JTG_TCK High[1]	4		4		ns
t_{TRSTW}	$\overline{\text{JTG\_TRST}}$ Pulse Width (Measured in JTG_TCK Cycles)[2]	4		4		t_{TCK}
Switching Characteristics						
t_{DTDO}	JTG_TDO Delay From JTG_TCK Low		16.5		14.5	ns
t_{DSYS}	System Outputs Delay After JTG_TCK Low[3]		18		16.5	ns
t_{DTMS}	TMS Delay After TCK High in SWD Mode	3.5	16.5	3.5	14.5	ns

[1] System inputs = DMC0_DQxx, DMC0_LDQS, $\overline{\text{DMC0\_LDQS}}$, DMC0_UDQS, $\overline{\text{DMC0\_UDQS}}$, PA_xx, PB_xx, PC_xx, SYS_BMODEx, $\overline{\text{SYS\_HWRST}}$, $\overline{\text{SYS\_FAULT}}$, $\overline{\text{SYS\_NMI}}$, TWI0_SCL, TWI0_SDA, and SYS_EXTWAKE.

[2] 50 MHz maximum.

[3] System outputs = DMC0_Axx, DMC0_BAx, $\overline{\text{DMC0\_CAS}}$, DMC0_CK, $\overline{\text{DMC0\_CK}}$, DMC0_CKE, $\overline{\text{DMC0\_CS0}}$, DMC0_DQxx, DMC0_LDM, DMC0_LDQS, $\overline{\text{DMC0\_LDQS}}$, DMC0_ODT, $\overline{\text{DMC0\_RAS}}$, DMC0_UDM, DMC0_UDQS, $\overline{\text{DMC0\_UDQS}}$, $\overline{\text{DMC0\_WE}}$, PA_xx, PB_xx, PC_xx, SYS_CLKOUT, $\overline{\text{SYS\_FAULT}}$, $\overline{\text{SYS\_RESOUT}}$, and $\overline{\text{SYS\_NMI}}$.

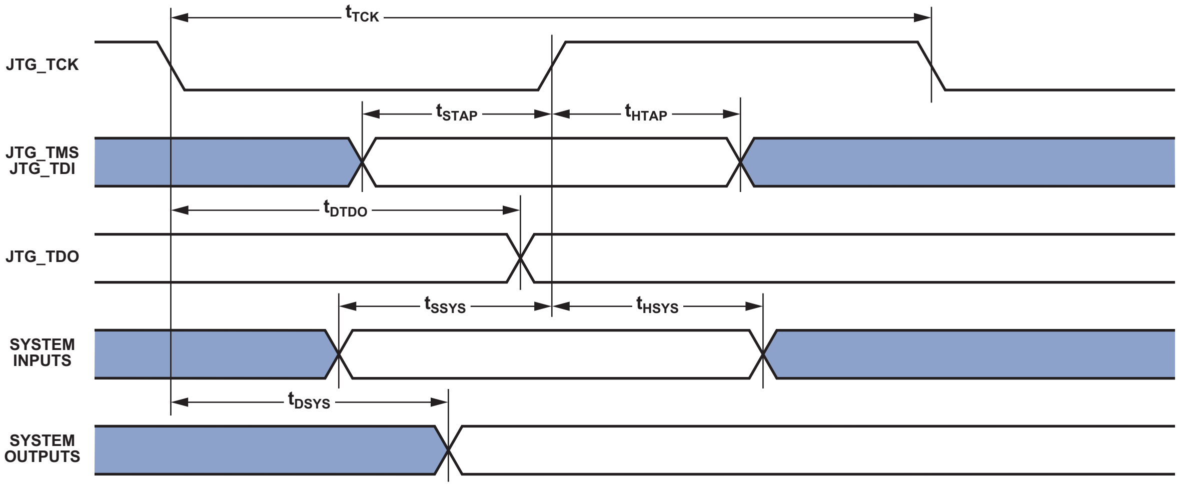


Figure 26. JTAG Port Timing

Serial Ports

To determine whether serial port (SPORT) communication is possible between two devices at clock speed n, the following specifications must be confirmed: 1) frame sync delay and frame sync setup and hold, 2) data delay and data setup and hold, and 3) serial clock (SPT_CLK) width. In Figure 27 either the rising edge or the falling edge of SPT_CLK (external or internal) can be used as the active sampling edge.

When externally generated the SPORT clock is called $f_{SPTCLKEXT}$:

$$t_{SPTCLKEXT} = \frac{1}{f_{SPTCLKEXT}}$$

When internally generated, the programmed SPORT clock ($f_{SPTCLKPROG}$) frequency in MHz is set by the following equation where CLKDIV is a field in the SPORT_DIV register that can be set from 0 to 65,535:

$$f_{SPTCLKPROG} = \frac{f_{SCLK0}}{(CLKDIV + 1)}$$

$$t_{SPTCLKPROG} = \frac{1}{f_{SPTCLKPROG}}$$

Table 49. Serial Ports—External Clock

Parameter		V_{DD_EXT} 1.8 V Nominal Min	V_{DD_EXT} 1.8 V Nominal Max	V_{DD_EXT} 3.3 V Nominal Min	V_{DD_EXT} 3.3 V Nominal Max	Unit
Timing Requirements						
t_{SFSE}	Frame Sync Setup Before SPT_CLK (Externally Generated Frame Sync in Either Transmit or Receive Mode)[1]	1.5		1		ns
t_{HFSE}	Frame Sync Hold After SPT_CLK (Externally Generated Frame Sync in Either Transmit or Receive Mode)[1]	3		3		ns
t_{SDRE}	Receive Data Setup Before Receive SPT_CLK[1]	1.5		1		ns
t_{HDRE}	Receive Data Hold After SPT_CLK[1]	3		3		ns
t_{SCLKW}	SPT_CLK Width[2]	$(0.5 \times t_{SPTCLKEXT}) - 1$		$(0.5 \times t_{SPTCLKEXT}) - 1$		ns
$t_{SPTCLKE}$	SPT_CLK Period[2]	$t_{SPTCLKEXT} - 1$		$t_{SPTCLKEXT} - 1$		ns
Switching Characteristics						
t_{DFSE}	Frame Sync Delay After SPT_CLK (Internally Generated Frame Sync in Either Transmit or Receive Mode)[3]		18		15	ns
t_{HOFSE}	Frame Sync Hold After SPT_CLK (Internally Generated Frame Sync in Either Transmit or Receive Mode)[3]	2.5		2.5		ns
t_{DDTE}	Transmit Data Delay After Transmit SPT_CLK[3]		18		15	ns
t_{HDTE}	Transmit Data Hold After Transmit SPT_CLK[3]	2.5		2.5		ns

[1] Referenced to sample edge.

[2] This specification indicates the minimum instantaneous width or period that can be tolerated due to duty cycle variation or jitter on the external SPT_CLK. For the external SPT_CLK ideal maximum frequency, see the $f_{SPTCLKEXT}$ specification in Table 18 on Page 52 in Clock Related Operating Conditions.

[3] Referenced to drive edge.

Table 50. Serial Ports—Internal Clock

		V_{DD_EXT} 1.8 V Nominal		V_{DD_EXT} 3.3 V Nominal		
Parameter		**Min**	**Max**	**Min**	**Max**	**Unit**
Timing Requirements						
t_{SFSI}	Frame Sync Setup Before SPT_CLK (Externally Generated Frame Sync in Either Transmit or Receive Mode)[1]	17		14.5		ns
t_{HFSI}	Frame Sync Hold After SPT_CLK (Externally Generated Frame Sync in Either Transmit or Receive Mode)[1]	–0.5		–0.5		ns
t_{SDRI}	Receive Data Setup Before SPT_CLK[1]	6.5		5		ns
t_{HDRI}	Receive Data Hold After SPT_CLK[1]	1.5		1		ns
Switching Characteristics						
t_{DFSI}	Frame Sync Delay After SPT_CLK (Internally Generated Frame Sync in Transmit or Receive Mode)[2]		2		2	ns
t_{HOFSI}	Frame Sync Hold After SPT_CLK (Internally Generated Frame Sync in Transmit or Receive Mode)[2]	–4.5		–3.5		ns
t_{DDTI}	Transmit Data Delay After SPT_CLK[2]		2		2	ns
t_{HDTI}	Transmit Data Hold After SPT_CLK[2]	–5		–3.5		ns
t_{SCLKIW}	SPT_CLK Width[3]	$0.5 \times t_{SPTCLKPROG} - 1.5$		$0.5 \times t_{SPTCLKPROG} - 1.5$		ns
$t_{SPTCLKI}$	SPT_CLK Period[3]	$t_{SPTCLKPROG} - 1.5$		$t_{SPTCLKPROG} - 1.5$		ns

[1] Referenced to the sample edge.

[2] Referenced to drive edge.

[3] See Table 18 on Page 52 in Clock Related Operating Conditions for details on the minimum period that may be programmed for $t_{SPTCLKPROG}$.

Table 61. Enhanced Parallel Peripheral Interface—External Clock

Parameter		V_{DD_EXT} 1.8 V Nominal Min	V_{DD_EXT} 1.8 V Nominal Max	V_{DD_EXT} 3.3 V Nominal Min	V_{DD_EXT} 3.3 V Nominal Max	Unit
Timing Requirements						
t_{PCLKW}	EPPI_CLK Width[1]	$(0.5 \times t_{PCLKEXT}) - 1$		$(0.5 \times t_{PCLKEXT}) - 1$		ns
t_{PCLK}	EPPI_CLK Period[1]	$t_{PCLKEXT} - 1$		$t_{PCLKEXT} - 1$		ns
t_{SFSPE}	External FS Setup Before EPPI_CLK	1.5		1		ns
t_{HFSPE}	External FS Hold After EPPI_CLK	3.3		3		ns
t_{SDRPE}	Receive Data Setup Before EPPI_CLK	1		1		ns
t_{HDRPE}	Receive Data Hold After EPPI_CLK	3		3		ns
Switching Characteristics						
t_{DFSPE}	Internal FS Delay After EPPI_CLK		17.5		14.5	ns
t_{HOFSPE}	Internal FS Hold After EPPI_CLK	2.5		2.5		ns
t_{DDTPE}	Transmit Data Delay After EPPI_CLK		17.5		14.5	ns
t_{HDTPE}	Transmit Data Hold After EPPI_CLK	2.5		2.5		ns

[1] This specification indicates the minimum instantaneous width or period that can be tolerated due to duty cycle variation or jitter on the external EPPI_CLK. For the external EPPI_CLK ideal maximum frequency, see the $f_{PCLKEXT}$ specification in Table 18 on Page 52 in Clock Related Operating Conditions.

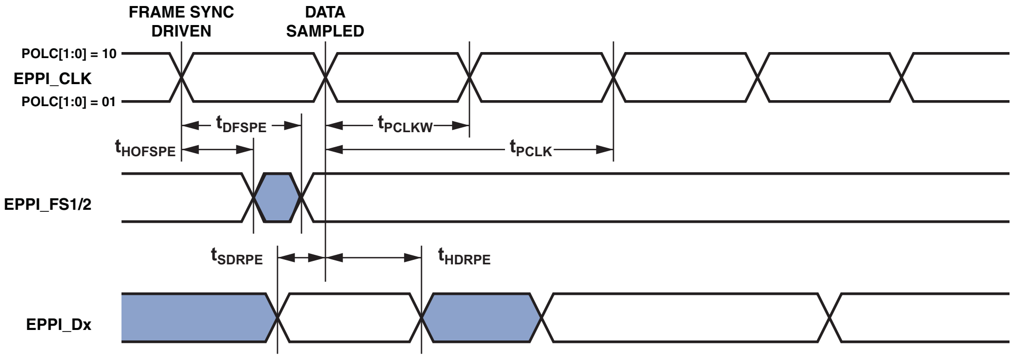


Figure 45. PPI External Clock GP Receive Mode with Internal Frame Sync Timing

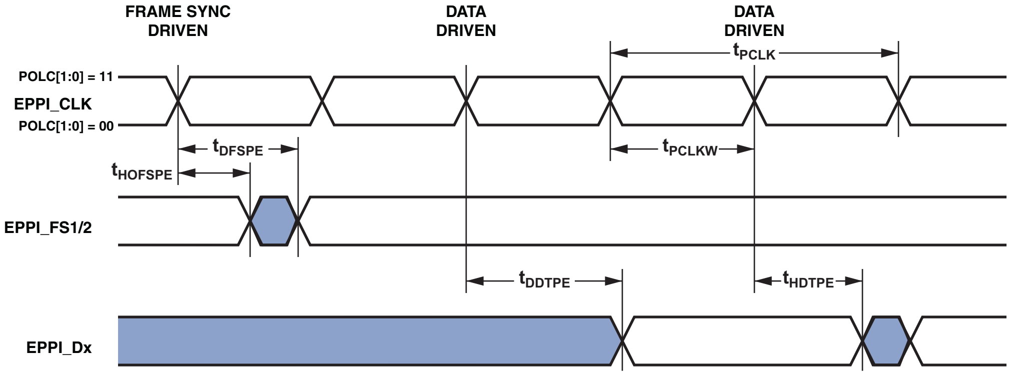


Figure 46. PPI External Clock GP Transmit Mode with Internal Frame Sync Timing

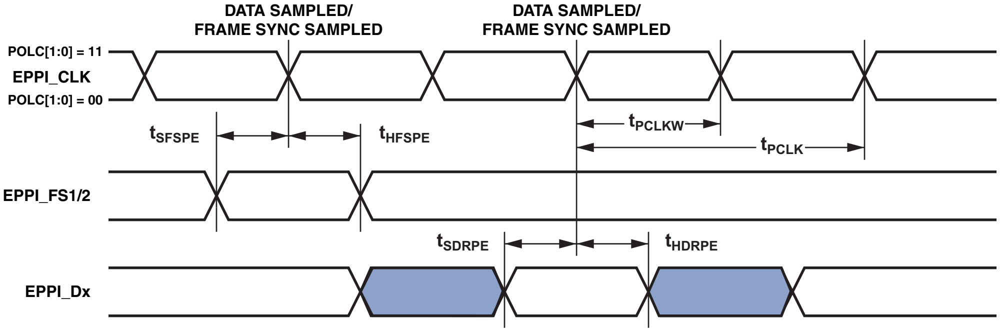


Figure 47. PPI External Clock GP Receive Mode with External Frame Sync Timing

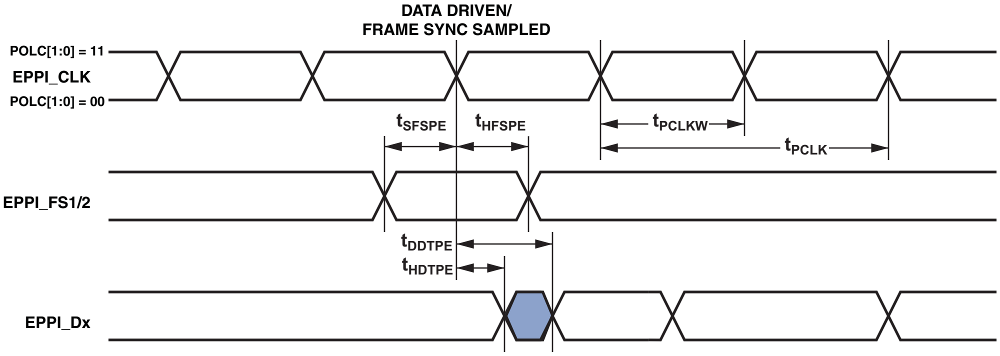


Figure 48. PPI External Clock GP Transmit Mode with External Frame Sync Timing

Universal Asynchronous Receiver-Transmitter (UART) Ports—Receive and Transmit Timing

The universal asynchronous receiver-transmitter (UART) ports receive and transmit operations are described in the *ADSP-BF70x Blackfin+ Processor Hardware Reference*.

Controller Area Network (CAN) Interface

The controller area network (CAN) interface timing is described in the *ADSP-BF70x Blackfin+ Processor Hardware Reference*.

Universal Serial Bus (USB) On-The-Go—Receive and Transmit Timing

Table 62 describes the universal serial bus (USB) on-the-go receive and transmit operations.

Table 62. USB On-The-Go—Receive and Transmit Timing

		V_{DD_USB} 3.3 V Nominal		
Parameter		**Min**	**Max**	**Unit**
Timing Requirements				
f_{USBS}	USB_XI Frequency	24	24	MHz
fs_{USB}	USB_XI Clock Frequency Stability	–50	+50	ppm

Mobile Storage Interface (MSI) Controller Timing

Table 64 and Figure 49 show I/O timing, related to the mobile storage interface (MSI).

The MSI timing depends on the period of the input clock that has been routed to the MSI peripheral ($t_{MSICLKIN}$) by setting the MSI0_UHS_EXT register. See Table 63 for this information.

Table 63. $t_{MSICLKIN}$ Settings

EXT_CLK_MUX_CTRL[31:30]	$t_{MSICLKIN}$
00	$t_{SCLK0} \times 2$
01	t_{SCLK0}
10	$t_{SCLK1} \times 3$

$$t_{MSICLKIN} = \frac{1}{f_{MSICLKIN}}$$

($f_{MSICLKPROG}$) frequency in MHz is set by the following equation where DIV0 is a field in the MSI_CLKDIV register that can be set from 0 to 255. When DIV0 is set between 1 and 255, the following equation is used to determine $f_{MSICLKPROG}$:

$$f_{MSICLKPROG} = \frac{f_{MSICLKIN}}{DIV0 \times 2}$$

When DIV0 = 0,

$$f_{MSICLKPROG} = f_{MSICLKIN}$$

Also note the following:

$$t_{MSICLKPROG} = \frac{1}{f_{MSICLKPROG}}$$

Table 64. MSI Controller Timing

	V_{DD_EXT} 1.8 V Nominal		V_{DD_EXT} 3.3 V Nominal		
Parameter	**Min**	**Max**	**Min**	**Max**	**Unit**
Timing Requirements					
t_{ISU} Input Setup Time	5.5		4.7		ns
t_{IH} Input Hold Time	2		0.5		ns
Switching Characteristics					
t_{MSICLK} Clock Period Data Transfer Mode[1]	$t_{MSICLKPROG} - 1.5$		$t_{MSICLKPROG} - 1.5$		ns
t_{WL} Clock Low Time	7		7		ns
t_{WH} Clock High Time	7		7		ns
t_{TLH} Clock Rise Time		3		3	ns
t_{THL} Clock Fall Time		3		3	ns
t_{ODLY} Output Delay Time During Data Transfer Mode		$(0.5 \times t_{MSICLKIN}) + 3.2$		$(0.5 \times t_{MSICLKIN}) + 3$	ns
t_{OH} Output Hold Time	$(0.5 \times t_{MSICLKIN}) - 4$		$(0.5 \times t_{MSICLKIN}) - 3$		ns

[1] See Table 18 on Page 52 in Clock Related Operating Conditions for details on the minimum period that may be programmed for $t_{MSICLKPROG}$.

ADSP-BF70x 184-BALL CSP_BGA BALL ASSIGNMENTS (NUMERICAL BY BALL NUMBER)

Figure 69 shows an overview of signal placement on the 184-ball CSP_BGA.

Table 67 lists the 184-ball CSP_BGA package by ball number for the ADSP-BF70x. Table 68 lists the 184-ball CSP_BGA package by signal.

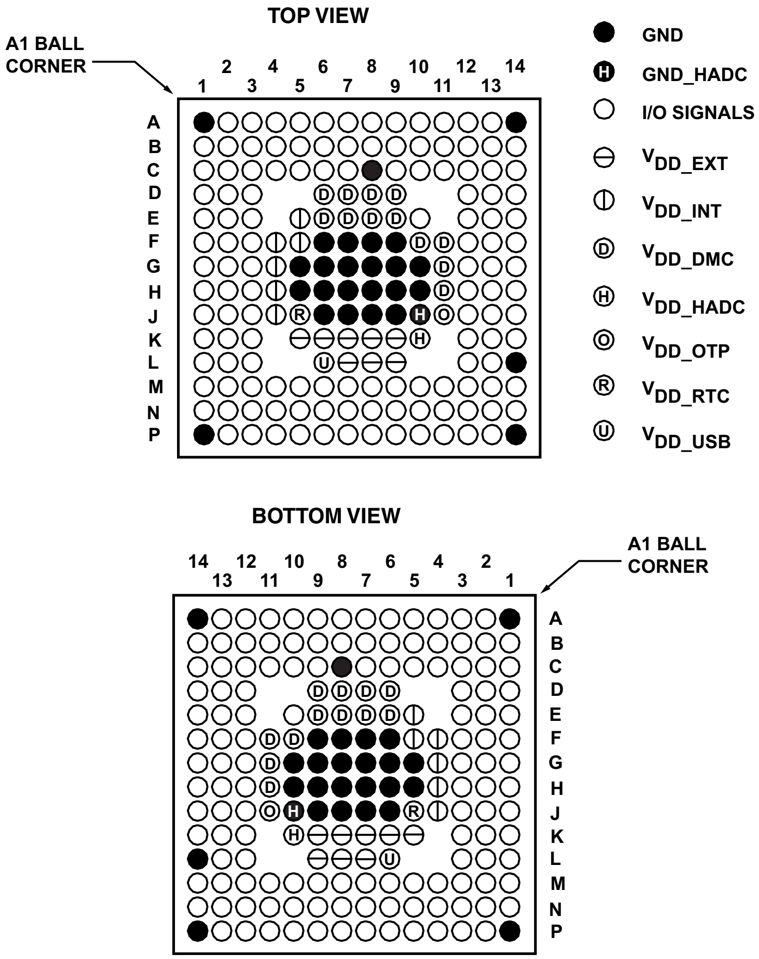


Figure 69. 184-Ball CSP_BGA Configuration