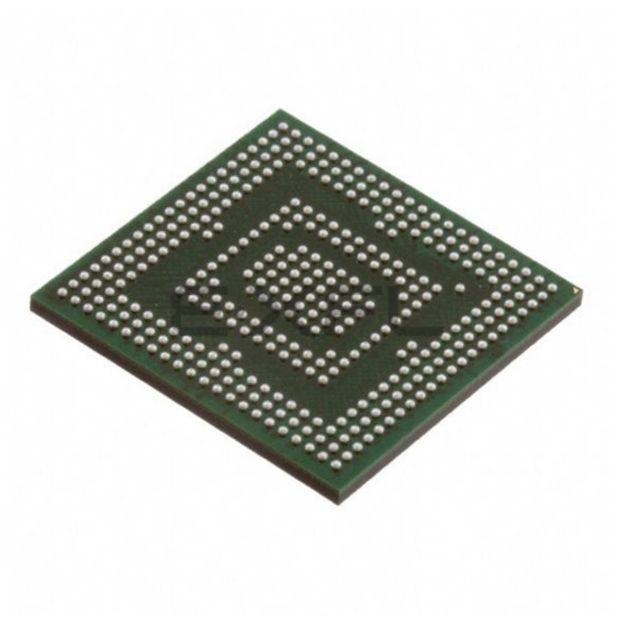
Analog Devices Inc. - ADSP-BF607KBCZ-5 Datasheet



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	Dual Core
Interface	CAN, EBI/EMI, Ethernet, I²C, SPI, SPORT, UART/USART, USB OTG
Clock Rate	500MHz
Non-Volatile Memory	ROM (64kB)
On-Chip RAM	808K x 8
Voltage - I/O	1.8V, 3.3V
Voltage - Core	1.25V
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	349-LFBGA, CSPBGA
Supplier Device Package	349-CSPBGA (19x19)
Purchase URL	https://www.e-xfl.com/product-detail/analog-devices/adsp-bf607kbcz-5

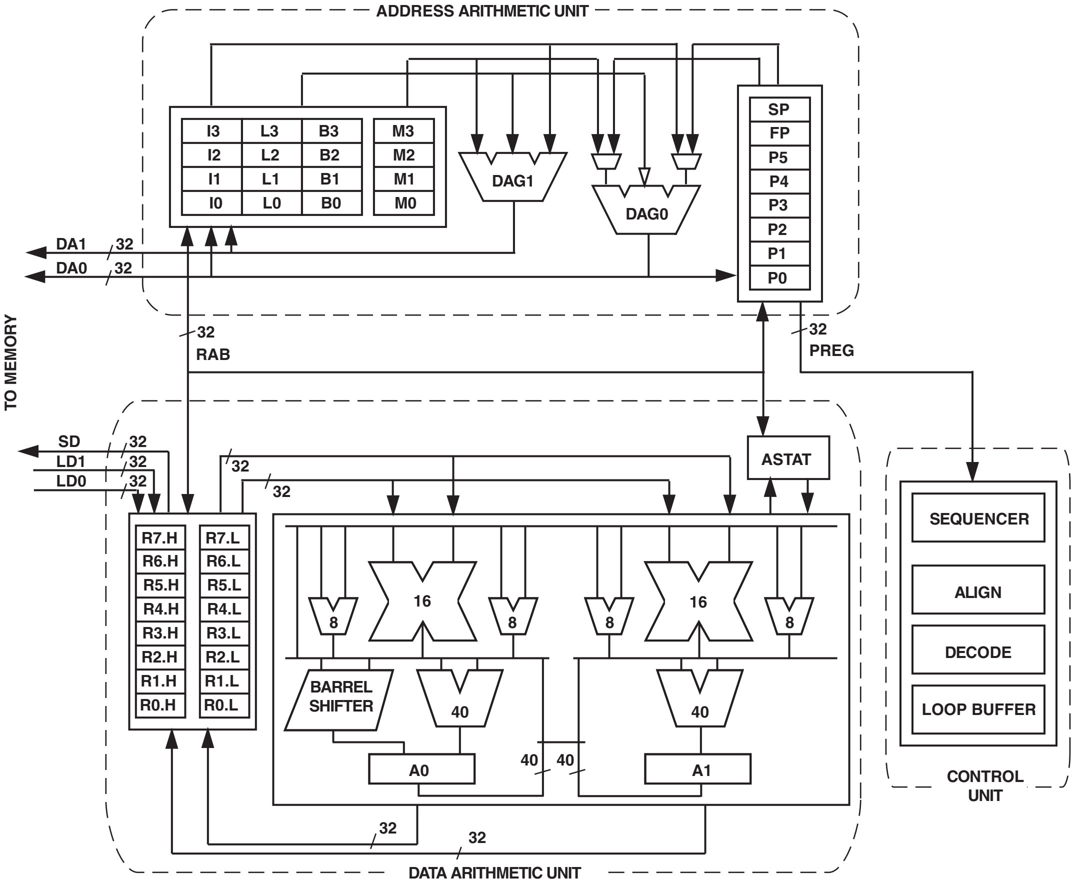


Figure 2. Blackfin Processor Core

The 40-bit shifter can perform shifts and rotates and is used to support normalization, field extract, and field deposit instructions.

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 static 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).

Blackfin processors support 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. Blackfin processors support 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-BF609 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.
- A similar buffer that interrupts on fractional buffers (for example, 1/2, 1/4).
- 1D DMA – uses a set of identical ping-pong buffers defined by a linked ring of two-word descriptor sets, each containing a link pointer and an address.
- 1D DMA – uses a linked list of 4 word descriptor sets containing a link pointer, an address, a length, and a configuration.
- 2D DMA – uses an array of one-word descriptor sets, specifying only the base DMA address.
- 2D DMA – uses a linked list of multi-word descriptor sets, specifying everything.

CRC Protection

The two CRC protection modules allow system software to periodically calculate the signature of code and/or data in memory, the content of memory-mapped registers, or communication message objects. Dedicated hardware circuitry compares the signature with pre calculated values and triggers appropriate fault events.

For example, every 100 ms the system software might initiate the signature calculation of the entire memory contents and compare these contents with expected, pre calculated values. If a mismatch occurs, a fault condition can be generated (via the processor core or the trigger routing unit).

The CRC is a hardware module based on a CRC32 engine that computes the CRC value of the 32-bit data words presented to it. Data is provided by the source channel of the memory-to-memory DMA (in memory scan mode) and is optionally forwarded to the destination channel (memory transfer mode). The main features of the CRC peripheral are:

- Memory scan mode
- Memory transfer mode
- Data verify mode
- Data fill mode
- User-programmable CRC32 polynomial
- Bit/byte mirroring option (endianness)
- Fault/error interrupt mechanisms
- 1D and 2D fill block to initialize array with constants.
- 32-bit CRC signature of a block of a memory or MMR block.

Event Handling

The processor provides event handling that supports both nesting and prioritization. Nesting allows multiple event service routines to be active simultaneously. Prioritization ensures that servicing of a higher-priority event takes precedence over servicing of a lower-priority event. The processor provides support for five different types of events:

- Emulation – An emulation event causes the processor to enter emulation mode, allowing command and control of the processor via the JTAG interface.
- Reset – This event resets the processor.
- Nonmaskable Interrupt (NMI) – The NMI event can be generated either by the software watchdog timer, by the $\overline{\text{NMI}}$ input signal to the processor, or by software. The NMI event is frequently used as a power-down indicator to initiate an orderly shutdown of the system.
- Exceptions – Events that occur synchronously to program flow (in other words, the exception is taken before the instruction is allowed to complete). Conditions such as data alignment violations and undefined instructions cause exceptions.
- Interrupts – Events that occur asynchronously to program flow. They are caused by input signals, timers, and other peripherals, as well as by an explicit software instruction.

Core Event Controller (CEC)

The CEC supports nine general-purpose interrupts (IVG15–7), in addition to the dedicated interrupt and exception events. Of these general-purpose interrupts, the two lowest-priority interrupts (IVG15–14) are recommended to be reserved for software interrupt handlers. For more information, see the *ADSP-BF60x Processor Programmer's Reference*.

System Event Controller (SEC)

The SEC manages the enabling, prioritization, and routing of events from each system interrupt or fault source. Additionally, it provides notification and identification of the highest priority active system interrupt request to each core and routes system fault sources to its integrated fault management unit.

Trigger Routing Unit (TRU)

The TRU provides system-level sequence control without core intervention. The TRU maps trigger masters (generators of triggers) to trigger slaves (receivers of triggers). Slave endpoints can be configured to respond to triggers in various ways. Common applications enabled by the TRU include:

- Automatically triggering the start of a DMA sequence after a sequence from another DMA channel completes
- Software triggering
- Synchronization of concurrent activities

Pin Interrupts

Every port pin on the processor can request interrupts in either an edge-sensitive or a level-sensitive manner with programmable polarity. Interrupt functionality is decoupled from GPIO operation. Six system-level interrupt channels (PINT0–5) are reserved for this purpose. Each of these interrupt channels can manage up to 32 interrupt pins. The assignment from pin to interrupt is not performed on a pin-by-pin basis. Rather, groups of eight pins (half ports) can be flexibly assigned to interrupt channels.

Every pin interrupt channel features a special set of 32-bit memory-mapped registers that enable half-port assignment and interrupt management. This includes masking, identification, and clearing of requests. These registers also enable access to the respective pin states and use of the interrupt latches, regardless of whether the interrupt is masked or not. Most control registers feature multiple MMR address entries to write-one-to-set or write-one-to-clear them individually.

General-Purpose I/O (GPIO)

Each general-purpose port pin can be individually controlled by manipulation of the port control, status, and interrupt registers:

- GPIO direction control register – Specifies the direction of each individual GPIO pin as input or output.
- GPIO control and status registers – A "write one to modify" mechanism allows any combination of individual GPIO pins to be modified in a single instruction, without affecting the level of any other GPIO pins.
- GPIO interrupt mask registers – Allow each individual GPIO pin to function as an interrupt to the processor. GPIO pins defined as inputs can be configured to generate hardware interrupts, while output pins can be triggered by software interrupts.
- GPIO interrupt sensitivity registers – Specify whether individual pins are level- or edge-sensitive and specify—if edge-sensitive—whether just the rising edge or both the rising and falling edges of the signal are significant.

Pin Multiplexing

The processor supports a flexible multiplexing scheme that multiplexes the GPIO pins with various peripherals. A maximum of 4 peripherals plus GPIO functionality is shared by each GPIO pin. All GPIO pins have a bypass path feature – that is, when the 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. For more information, see GP I/O Multiplexing for 349-Ball CSP_BGA on Page 33.

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 and Figure 4.

- A 32-bit threshold block with 16 thresholds, a histogram, and run-length encoding
- Two 32-bit integral blocks that support regular and diagonal integrals
- An up- and down-scaling unit with independent scaling ratios for horizontal and vertical components
- Input and output formatters for compatibility with many data formats, including Bayer input format

The PVP can form a pipe of all the constituent algorithmic modules and is dynamically reconfigurable to form different pipeline structures.

The PVP supports the simultaneous processing of up to four data streams. The memory pipe stream operates on data received by DMA from any L1, L2, or L3 memory. The three camera pipe streams operate on a common input received directly from any of the three PPI inputs. Optionally, the PIXC can convert color data received by the PPI and forward luma values to the PVP's monochrome engine. Each stream has a dedicated DMA output. This preprocessing concept ensures careful use of available power and bandwidth budgets and frees up the processor cores for other tasks.

The PVP provides for direct core MMR access to all control/status registers. Two hardware interrupts interface to the system event controller. For optimal performance, the PVP allows register programming through its control DMA interface, as well as outputting selected status registers through the status DMA interface. This mechanism enables the PVP to automatically process job lists completely independent of the Blackfin cores.

Pixel Compositor (PIXC)

The pixel compositor (PIXC) provides image overlays with transparent-color support, alpha blending, and color space conversion capabilities for output to TFT LCDs and NTSC/PAL video encoders. It provides all of the control to allow two data streams from two separate data buffers to be combined, blended, and converted into appropriate forms for both LCD panels and digital video outputs. The main image buffer provides the basic background image, which is presented in the data stream. The overlay image buffer allows the user to add multiple foreground text, graphics, or video objects on top of the main image or video data stream.

Parallel Peripheral Interface (PPI)

The processor provides up to three parallel peripheral interfaces (PPIs), supporting data widths up to 24 bits. The PPI supports direct connection to TFT LCD panels, parallel analog-to-digital and digital-to-analog converters, video encoders and decoders, image sensor modules and other general-purpose peripherals.

The following features are supported in the PPI module:

- Programmable data length: 8 bits, 10 bits, 12 bits, 14 bits, 16 bits, 18 bits, and 24 bits per clock.
- Various framed, non-framed, and general-purpose operating modes. Frame syncs can be generated internally or can be supplied by an external device.
- ITU-656 status word error detection and correction for ITU-656 receive modes and ITU-656 preamble and status word decode.
- Optional packing and unpacking of data to/from 32 bits from/to 8 bits, 16 bits and 24 bits. If packing/unpacking is enabled, endianness can be configured to change the order of packing/unpacking of bytes/words.
- RGB888 can be converted to RGB666 or RGB565 for transmit modes.
- Various de-interleaving/interleaving modes for receiving/transmitting 4:2:2 YCrCb data.
- Configurable LCD data enable (DEN) output available on Frame Sync 3.

PROCESSOR SAFETY FEATURES

The ADSP-BF60x processor has been designed for functional safety applications. While the level of safety is mainly dominated by the system concept, the following primitives are provided by the devices to build a robust safety concept.

Dual Core Supervision

The processor has been implemented as dual-core devices to separate critical tasks to large independency. Software models support mutual supervision of the cores in symmetrical fashion.

Multi-Parity-Bit-Protected L1 Memories

In the processor's L1 memory space, whether SRAM or cache, each word is protected by multiple parity bits to detect the single event upsets that occur in all RAMs. This applies both to L1 instruction and data memory spaces.

ECC-Protected L2 Memories

Error correcting codes (ECC) are used to correct single event upsets. The L2 memory is protected with a Single Error Correct-Double Error Detect (SEC-DED) code. By default ECC is enabled, but it can be disabled on a per-bank basis. Single-bit errors are transparently corrected. Dual-bit errors can issue a system event or fault if enabled. ECC protection is fully transparent to the user, even if L2 memory is read or written by 8-bit or 16-bit entities.

CRC-Protected Memories

While parity bit and ECC protection mainly protect against random soft errors in L1 and L2 memory cells, the CRC engines can be used to protect against systematic errors (pointer errors) and static content (instruction code) of L1, L2 and even L3 memories (DDR2, LPDDR). The processors feature two CRC engines which are embedded in the memory-to-memory DMA controllers. CRC check sums can be calculated or compared on the fly during memory transfers, or one or multiple memory regions can be continuously scrubbed by single DMA work unit as per DMA descriptor chain instructions. The CRC engine also protects data loaded during the boot process.

ADSP-BF60x DETAILED SIGNAL DESCRIPTIONS

Table 6 provides a detailed description of each signal.

Table 6. Detailed Signal Descriptions

Signal Name	Direction	Description
ACM_An	Output	**ADC Control Signals** Function varies by mode.
ACM_CLK	Output	**Clock** SCLK derived clock for connecting to an ADC.
ACM_FS	Output	**Frame Sync** Typically used as an ADC chip select.
ACM_Tn	Input	**External Trigger n** Input for external trigger events.
CAN_RX	Input	**Receive** Typically an external CAN transceiver's RX output.
CAN_TX	Output	**Transmit** Typically an external CAN transceiver's TX input.
CNT_DG	Input	**Count Down and Gate** Depending on the mode of operation this input acts either as a count down signal or a gate signal. Count Down: This input causes the GP counter to decrement. Gate: Stops the GP counter from incrementing or decrementing.
CNT_UD	Input	**Count Up and Direction** Depending on the mode of operation this input acts either as a count up signal or a direction signal. Count Up: This input causes the GP counter to increment. Direction: Selects whether the GP counter is incrementing or decrementing.
CNT_ZM	Input	**Count Zero Marker** Input that connects to the zero marker output of a rotary device or detects the pressing of a push button.
DMC_Ann	Output	**Address n** Address bus.
DMC_BAn	Output	**Bank Address Input n** Defines which internal bank an ACTIVATE, READ, WRITE, or PRECHARGE command is being applied to on the dynamic memory. Also defines which mode registers (MR, EMR, EMR2, and/or EMR3) are loaded during the LOAD MODE REGISTER command.
$\overline{\text{DMC\_CAS}}$	Output	**Column Address Strobe** Defines the operation for external dynamic memory to perform in conjunction with other DMC command signals. Connect to the CAS input of dynamic memory.
$\overline{\text{DMC\_CK}}$	Output	**Clock (complement)** Complement of DMC_CK.
DMC_CK	Output	**Clock** Outputs DCLK to external dynamic memory.
DMC_CKE	Output	**Clock enable** Active high clock enables. Connects to the dynamic memory's CKE input.
$\overline{\text{DMC\_CSn}}$	Output	**Chip Select n** Commands are recognized by the memory only when this signal is asserted.
DMC_DQnn	I/O	**Data n** Bidirectional data bus.
DMC_LDM	Output	**Data Mask for Lower Byte** Mask for DMC_DQ07:DMC_DQ00 write data when driven high. Sampled on both edges of the data strobe by the dynamic memory.
$\overline{\text{DMC\_LDQS}}$	I/O	**Data Strobe for Lower Byte (complement)** Complement of LDQS. Not used in single-ended mode.
DMC_LDQS	I/O	**Data Strobe for Lower Byte** DMC_DQ07:DMC_DQ00 data strobe. Output with Write Data. Input with Read Data. May be single-ended or differential depending on register settings.
DMC_ODT	Output	**On-die Termination** Enables dynamic memory termination resistances when driven high (assuming the memory is properly configured). ODT is enabled/disabled regardless of read or write commands.
$\overline{\text{DMC\_RAS}}$	Output	**Row Address Strobe** Defines the operation for external dynamic memory to perform in conjunction with other DMC command signals. Connect to the RAS input of dynamic memory.
DMC_UDM	Output	**Data Mask for Upper Byte** Mask for DMC_DQ15:DMC_DQ08 write data when driven high. Sampled on both edges of the data strobe by the dynamic memory.
$\overline{\text{DMC\_UDQS}}$	I/O	**Data Strobe for Upper Byte (complement)** Complement of UDQS. Not used in single-ended mode.
DMC_UDQS	I/O	**Data Strobe for Upper Byte** DMC_DQ15:DMC_DQ08 data strobe. Output with Write Data. Input with Read Data. May be single-ended or differential depending on register settings.
$\overline{\text{DMC\_WE}}$	Output	**Write Enable** Defines the operation for external dynamic memory to perform in conjunction with other DMC command signals. Connect to the $\overline{\text{WE}}$ input of dynamic memory.

Table 7. ADSP-BF60x 349-Ball CSP_BGA Signal Descriptions (Continued)

Signal Name	Description	Port	Pin Name
ETH_PTPCLKIN	EMAC0/EMAC1 PTP Clock Input	C	PC_13
GND	Ground	Not Muxed	GND
$\overline{\text{JTG\_EMU}}$	Emulation Output	Not Muxed	$\overline{\text{JTG\_EMU}}$
JTG_TCK	JTAG Clock	Not Muxed	JTG_TCK
JTG_TDI	JTAG Serial Data Input	Not Muxed	JTG_TDI
JTG_TDO	JTAG Serial Data Output	Not Muxed	JTG_TDO
JTG_TMS	JTAG Mode Select	Not Muxed	JTG_TMS
$\overline{\text{JTG\_TRST}}$	JTAG Reset	Not Muxed	$\overline{\text{JTG\_TRST}}$
LP0_ACK	LP0 Acknowledge	B	PB_01
LP0_CLK	LP0 Clock	B	PB_00
LP0_D0	LP0 Data 0	A	PA_00
LP0_D1	LP0 Data 1	A	PA_01
LP0_D2	LP0 Data 2	A	PA_02
LP0_D3	LP0 Data 3	A	PA_03
LP0_D4	LP0 Data 4	A	PA_04
LP0_D5	LP0 Data 5	A	PA_05
LP0_D6	LP0 Data 6	A	PA_06
LP0_D7	LP0 Data 7	A	PA_07
LP1_ACK	LP1 Acknowledge	B	PB_02
LP1_CLK	LP1 Clock	B	PB_03
LP1_D0	LP1 Data 0	A	PA_08
LP1_D1	LP1 Data 1	A	PA_09
LP1_D2	LP1 Data 2	A	PA_10
LP1_D3	LP1 Data 3	A	PA_11
LP1_D4	LP1 Data 4	A	PA_12
LP1_D5	LP1 Data 5	A	PA_13
LP1_D6	LP1 Data 6	A	PA_14
LP1_D7	LP1 Data 7	A	PA_15
LP2_ACK	LP2 Acknowledge	E	PE_08
LP2_CLK	LP2 Clock	E	PE_09
LP2_D0	LP2 Data 0	F	PF_00
LP2_D1	LP2 Data 1	F	PF_01
LP2_D2	LP2 Data 2	F	PF_02
LP2_D3	LP2 Data 3	F	PF_03
LP2_D4	LP2 Data 4	F	PF_04
LP2_D5	LP2 Data 5	F	PF_05
LP2_D6	LP2 Data 6	F	PF_06
LP2_D7	LP2 Data 7	F	PF_07
LP3_ACK	LP3 Acknowledge	E	PE_07
LP3_CLK	LP3 Clock	E	PE_06
LP3_D0	LP3 Data 0	F	PF_08
LP3_D1	LP3 Data 1	F	PF_09
LP3_D2	LP3 Data 2	F	PF_10
LP3_D3	LP3 Data 3	F	PF_11
LP3_D4	LP3 Data 4	F	PF_12
LP3_D5	LP3 Data 5	F	PF_13

Asynchronous Page Mode Read

Table 30. Asynchronous Page Mode Read

Parameter		V_{DD_EXT} 1.8 V /3.3 V Nominal Min	Max	Unit
Switching Characteristics				
t_{AV}	SMC0_Ax (Address) Valid for First Address Min Width[1]	(PREST + RST + PREAT + RAT) × t_{SCLK0} – 2		ns
t_{AV1}	SMC0_Ax (Address) Valid for Subsequent SMC0_Ax (Address) Min Width	PGWS × t_{SCLK0} – 2		ns
t_{WADV}	SMC0_NORDV Active Low Width[2]	RST × t_{SCLK0} – 2		ns
t_{HARE}	Output[3] Hold After $\overline{\text{SMC0\_ARE}}$ High[4]	RHT × t_{SCLK0} – 2		ns
t_{WARE}[5]	$\overline{\text{SMC0\_ARE}}$ Active Low Width[6, 7]	(RAT + (Nw – 1) × PGWS) × t_{SCLK0} – 2		ns

[1] PREST, RST, PREAT and RAT values set using the SMC_BxETIM.PREST bits, SMC_BxTIM.RST bits, SMC_BxETIM.PREAT bits, and the SMC_BxTIM.RAT bits.
[2] RST value set using the SMC_BxTIM.RST bits.
[3] Output signals are SMC0_Ax, $\overline{\text{SMC0\_AMSx}}$, $\overline{\text{SMC0\_AOE}}$.
[4] RHT value set using the SMC_BxTIM.RHT bits.
[5] SMC_BxCTL.ARDYEN bit = 0.
[6] RAT value set using the SMC_BxTIM.RAT bits.
[7] Nw = Number of 16-bit data words read.

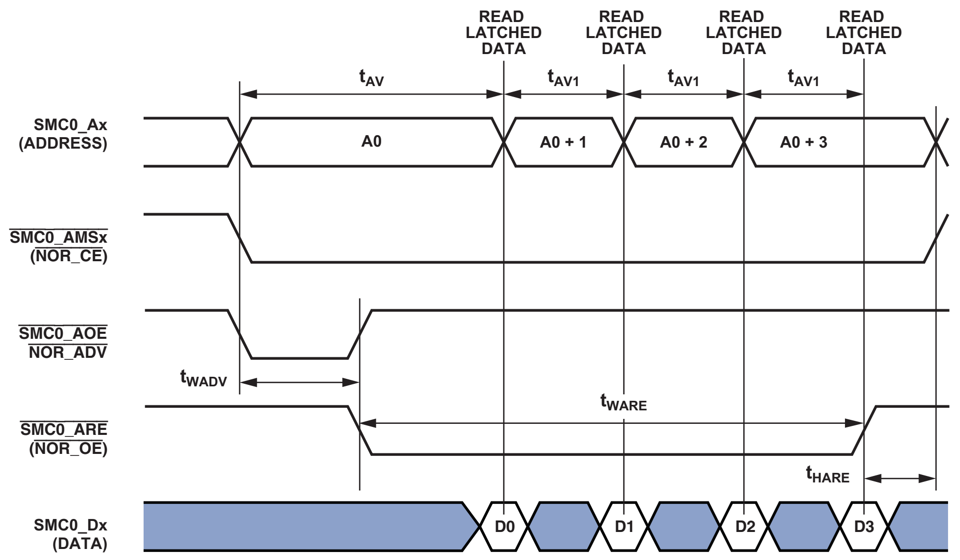
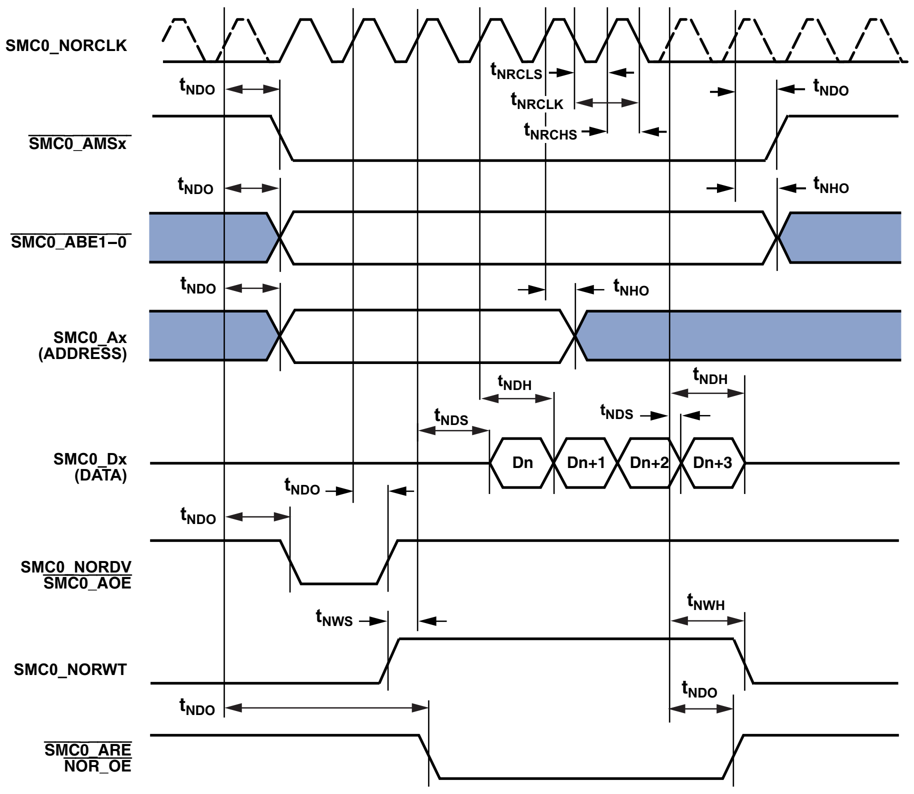


Figure 14. Asynchronous Page Mode Read



NOTE: SMC0_NORCLK dotted line represents a free running version of SMC0_NORCLK that is not visible on the SMC0_NORCLK pin.

Figure 15. Synchronous Burst AC Interface Timing

DDR2 SDRAM Read Cycle Timing

Table 37. DDR2 SDRAM Read Cycle Timing, V_{DD_DMC} Nominal 1.8 V

Parameter		250 MHz[1] Min	250 MHz[1] Max	Unit
Timing Requirements				
t_{DQSQ}	DMC0_DQS-DMC0_DQ Skew for DMC0_DQS and Associated DMC0_DQ Signals		0.35	ns
t_{QH}	DMC0_DQ, DMC0_DQS Output Hold Time From DMC0_DQS	1.6		ns
t_{RPRE}	Read Preamble	0.9		t_{CK}
t_{RPST}	Read Postamble	0.4		t_{CK}

[1] In order to ensure proper operation of the DDR2, all the DDR2 guidelines have to be strictly followed.

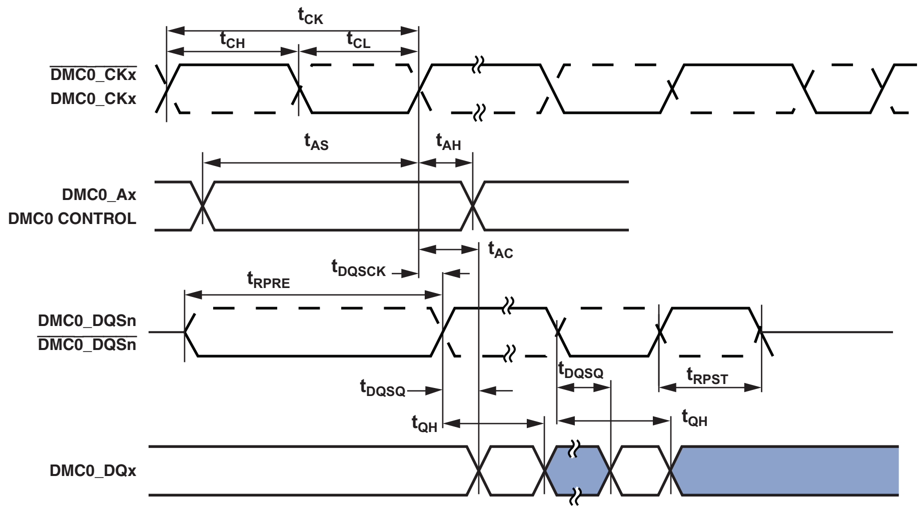


Figure 20. DDR2 SDRAM Controller Input AC Timing

DDR2 SDRAM Write Cycle Timing

Table 38. DDR2 SDRAM Write Cycle Timing, V_{DD_DMC} Nominal 1.8 V

Parameter		250 MHz[1] Min	250 MHz[1] Max	Unit
Switching Characteristics				
t_{DQSS}[2]	DMC0_DQS Latching Rising Transitions to Associated Clock Edges	–0.15	0.15	t_{CK}
t_{DS}	Last Data Valid to DMC0_DQS Delay	0.15		ns
t_{DH}	DMC0_DQS to First Data Invalid Delay	0.3		ns
t_{DSS}	DMC0_DQS Falling Edge to Clock Setup Time	0.25		t_{CK}
t_{DSH}	DMC0_DQS Falling Edge Hold Time From DMC0_CK	0.25		t_{CK}
t_{DQSH}	DMC0_DQS Input High Pulse Width	0.35		t_{CK}
t_{DQSL}	DMC0_DQS Input Low Pulse Width	0.35		t_{CK}
t_{WPRE}	Write Preamble	0.35		t_{CK}
t_{WPST}	Write Postamble	0.4		t_{CK}
t_{IPW}	Address and Control Output Pulse Width	0.6		t_{CK}
t_{DIPW}	DMC0_DQ and DMC0_DM Output Pulse Width	0.35		t_{CK}

[1] In order to ensure proper operation of the DDR2, all the DDR2 guidelines have to be strictly followed.
[2] Write command to first DMC0_DQS delay = WL × t_{CK} + t_{DQSS}.

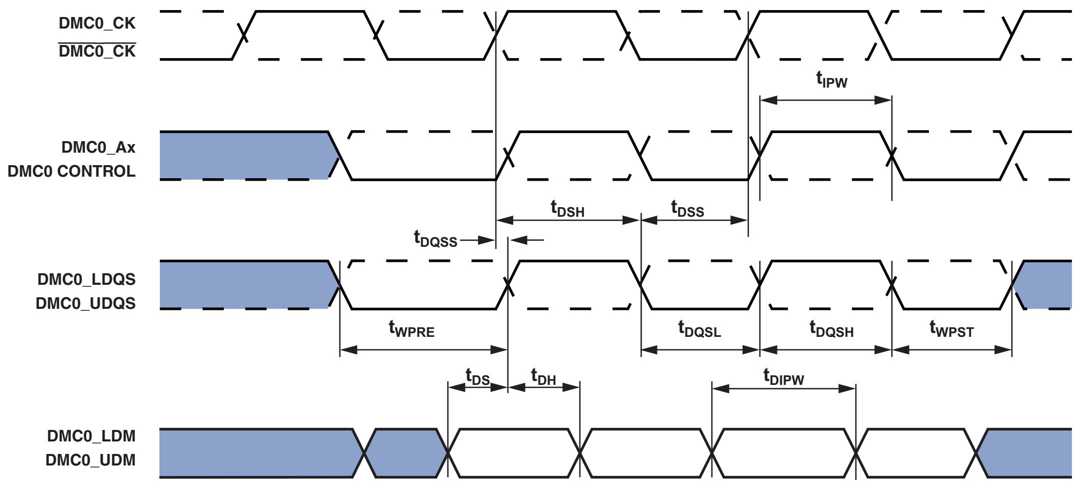


Figure 21. DDR2 SDRAM Controller Output AC Timing

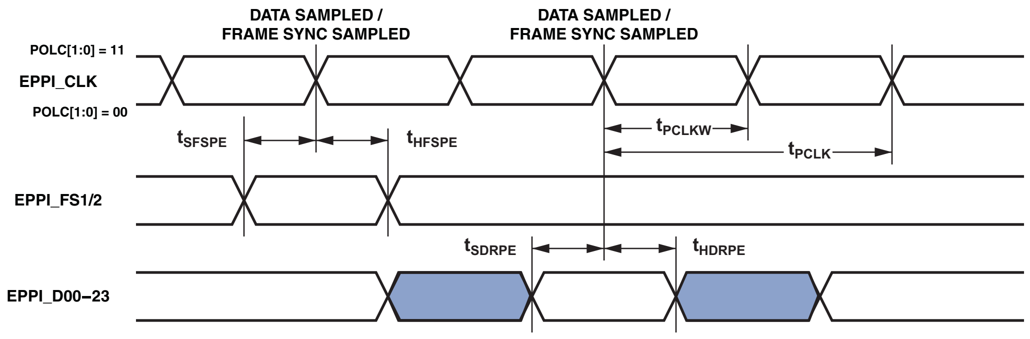


Figure 32. PPI External Clock GP Receive Mode with External Frame Sync Timing

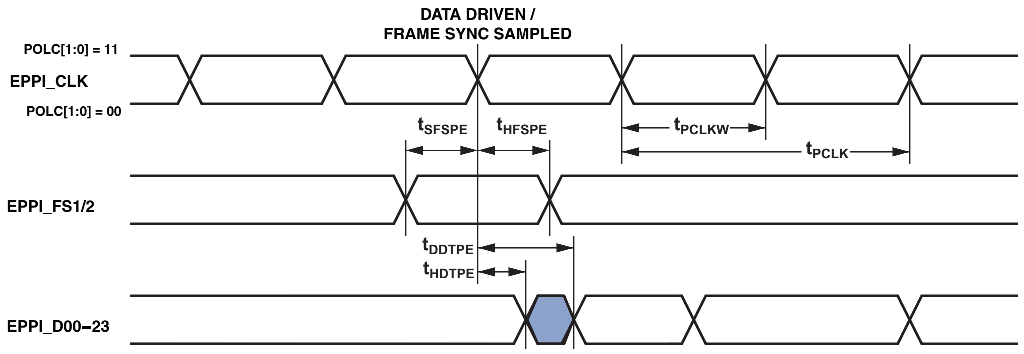


Figure 33. PPI External Clock GP Transmit Mode with External Frame Sync Timing

Table 47. Serial Ports—Internal 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_{SFSI}	Frame Sync Setup Before SPT_CLK (Externally Generated Frame Sync in either Transmit or Receive Mode)[1]	16.8		12		ns
t_{HFSI}	Frame Sync Hold After SPT_CLK (Externally Generated Frame Sync in either Transmit or Receive Mode)[1]	0		–0.5		ns
t_{SDRI}	Receive Data Setup Before SPT_CLK[1]	4.8		3.4		ns
t_{HDRI}	Receive Data Hold After SPT_CLK[1]	1.5		1.5		ns
Switching Characteristics						
t_{DFSI}	Frame Sync Delay After SPT_CLK (Internally Generated Frame Sync in Transmit or Receive Mode)[2]		3.5		3.5	ns
t_{HOFSI}	Frame Sync Hold After SPT_CLK (Internally Generated Frame Sync in Transmit or Receive Mode)[2]	–1		–1		ns
t_{DDTI}	Transmit Data Delay After SPT_CLK[2]		3.5		3.5	ns
t_{HDTI}	Transmit Data Hold After SPT_CLK[2]	–1		–1		ns
t_{SCLKIW}	SPT_CLK Width[3]	$0.5 \times t_{SPTCLKPROG} - 1.5$		$0.5 \times t_{SPTCLKPROG} - 1.5$		ns
t_{SPTCLK}	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 17 on Page 53 in Clock Related Operating Conditions for details on the minimum period that may be programmed for $t_{SPTCLKPROG}$.

The SPT_TDV output signal becomes active in SPORT multi-channel mode. During transmit slots (enabled with active channel selection registers) the SPT_TDV is asserted for communication with external devices.

Table 49. Serial Ports—TDV (Transmit Data Valid)

		V_{DD_EXT} 1.8 V Nominal		V_{DD_EXT} 3.3 V Nominal		
Parameter		**Min**	**Max**	**Min**	**Max**	**Unit**
Switching Characteristics						
t_{DRDVEN}	Data-Valid Enable Delay from Drive Edge of External Clock[1]	2		2		ns
t_{DFDVEN}	Data-Valid Disable Delay from Drive Edge of External Clock[1]		18.8		14	ns
t_{DRDVIN}	Data-Valid Enable Delay from Drive Edge of Internal Clock[1]	–1		–1		ns
t_{DFDVIN}	Data-Valid Disable Delay from Drive Edge of Internal Clock[1]		3.5		3.5	ns

[1] Referenced to drive edge.

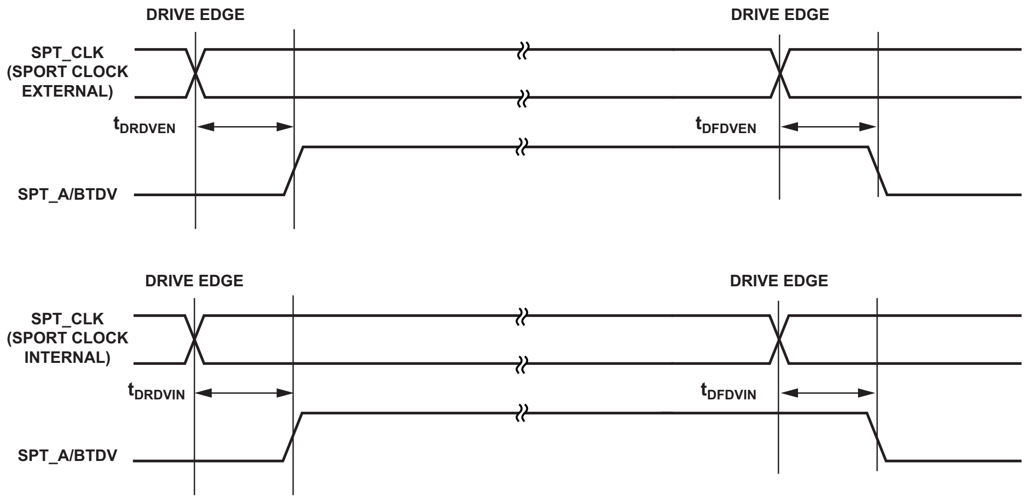


Figure 38. Serial Ports—Transmit Data Valid Internal and External Clock

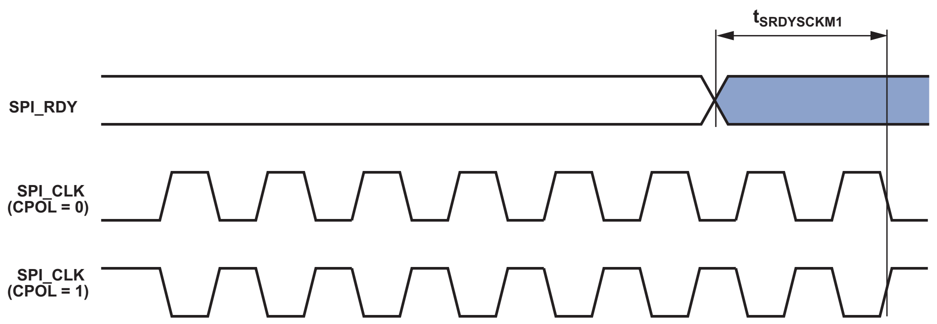


Figure 47. SPI_RDY Setup Before SPI_CLK with CPHA = 1

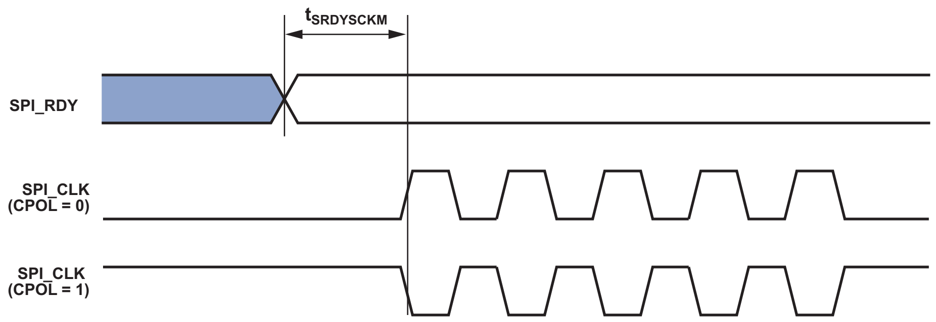


Figure 48. SPI_CLK Switching Diagram after SPI_RDY Assertion, CPHA = x

General-Purpose Port Timing

Table 57 and Figure 49 describe general-purpose port operations.

Table 57. General-Purpose Port Timing

Parameter		V_{DD_EXT} 1.8 V/3.3 V Nominal Min	Max	Unit
Timing Requirement				
t_{WFI}	General-Purpose Port Pin Input Pulse Width	$2 \times t_{SCLK0} - 1.5$		ns

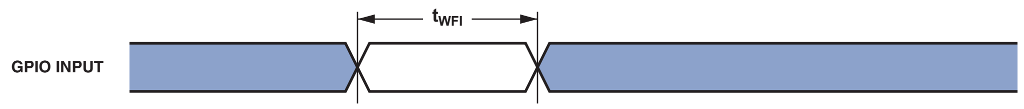


Figure 49. General-Purpose Port Timing

Timer Cycle Timing

Table 58 and Figure 50 describe timer expired operations. The input signal is asynchronous in "width capture mode" and "external clock mode" and has an ideal maximum input frequency of (f_{SCLK0}/4) MHz. The Period Value (VALUE) is the timer period assigned in the TMx_TMRn_PER register and can range from 2 to $2^{32} - 1$.

Table 58. Timer Cycle Timing

Parameter		V_{DD_EXT} 1.8 V Nominal Min	Max	V_{DD_EXT} 3.3 V Nominal Min	Max	Unit
Timing Requirements						
t_{WL}	Timer Pulse Width Input Low[1]	$2 \times t_{SCLK0} - 1.5$		$2 \times t_{SCLK0} - 1.5$		ns
t_{WH}	Timer Pulse Width Input High[1]	$2 \times t_{SCLK0} - 1.5$		$2 \times t_{SCLK0} - 1.5$		ns
Switching Characteristics						
t_{HTO}	Timer Pulse Width Output	$t_{SCLK0} \times$ VALUE – 1.5		$t_{SCLK0} \times$ VALUE – 1.5		ns

[1] This specification indicates the minimum instantaneous width that can be tolerated due to duty cycle variation or jitter for TMx signals in width capture and external clock modes. The ideal maximum frequency for TMx signals is listed in Timer Cycle Timing on this page.

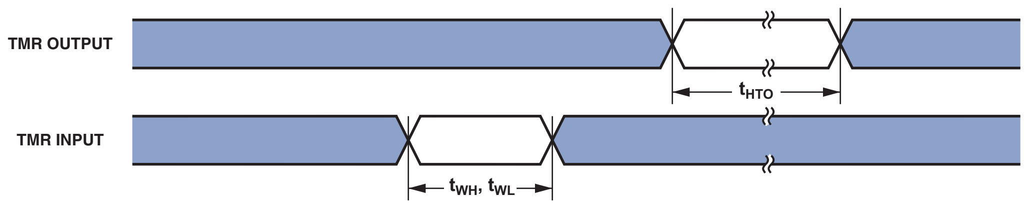


Figure 50. Timer Cycle Timing

ADC Controller Module (ACM) Timing

Table 61 and Figure 53 describe ACM operations.

When internally generated, the programmed ACM clock ($f_{ACLKPROG}$) frequency in MHz is set by the following equation where CKDIV is a field in the ACM_TC0 register and ranges from 1 to 255:

$$f_{ACLKPROG} = \frac{f_{SCLK1}}{CKDIV + 1}$$

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

Setup cycles (SC) in Table 61 is also a field in the ACM_TC0 register and ranges from 0 to 4095. Hold Cycles (HC) is a field in the ACM_TC1 register that ranges from 0 to 15.

Table 61. ACM Timing

		V_{DD_EXT} 1.8 V/3.3 V Nominal		
Parameter		**Min**	**Max**	**Unit**
Timing Requirements				
t_{SDR}	SPORT DRxPRI/DRxSEC Setup Before ACMx_CLK	3		ns
t_{HDR}	SPORT DRxPRI/DRxSEC Hold After ACMx_CLK	1.5		ns
Switching Characteristics				
t_{SCTLCS}	ACM Controls (ACMx_A[4:0]) Setup Before Assertion of $\overline{CS}$	$(SC + 1) \times t_{SCLK1} - 3$		ns
t_{HCTLCS}	ACM Control (ACMx_A[4:0]) Hold After De-assertion of $\overline{CS}$	$HC \times t_{ACLK} + 0.1$		ns
t_{ACLKW}	ACM Clock Pulse Width[1]	$(0.5 \times t_{ACLKPROG}) - 1.5$		ns
t_{ACLK}	ACM Clock Period[1]	$t_{ACLKPROG} - 1.5$		ns
$t_{HCSACLK}$	$\overline{CS}$ Hold to ACMx_CLK Edge	–0.1		ns
$t_{SCSACLK}$	$\overline{CS}$ Setup to ACMx_CLK Edge	$t_{ACLK} - 3.5$		ns

[1] See Table 17 on Page 53 in Clock Related Operating Conditions for details on the minimum period that may be programmed for $t_{ACLKPROG}$.

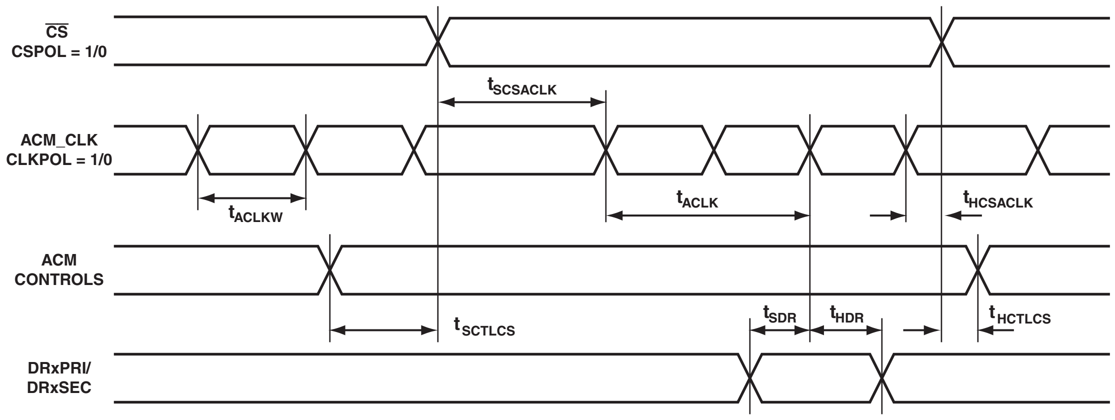


Figure 53. ACM Timing

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

The UART ports receive and transmit operations are described in the *ADSP-BF60x Blackfin Processor Hardware Reference Manual.*

CAN Interface

The CAN interface timing is described in the *ADSP-BF60x Blackfin Processor Hardware Reference Manual.*

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

Table 62 describes the USB On-The-Go receive and transmit operations.

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

Parameter		V_{DD_USB} 3.3 V Nominal Min	Max	Unit
Timing Requirements				
f_{USBS}	USB_XI Frequency	48	48	MHz
fs_{USB}	USB_XI Clock Frequency Stability	–50	+50	ppm

RSI Controller Timing

Table 63 and Figure 54 describe RSI controller timing.

Table 63. RSI Controller Timing

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_{ISU}	Input Setup Time	11		9.6		ns
t_{IH}	Input Hold Time	2		2		ns
Switching Characteristics						
f_{PP}	Clock Frequency Data Transfer Mode[1]		41.67		41.67	MHz
t_{WL}	Clock Low Time	8		8		ns
t_{WH}	Clock High Time	8		8		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		2.5		2.5	ns
t_{OH}	Output Hold Time	–1		–1		ns

[1] $t_{PP} = 1/f_{PP}$

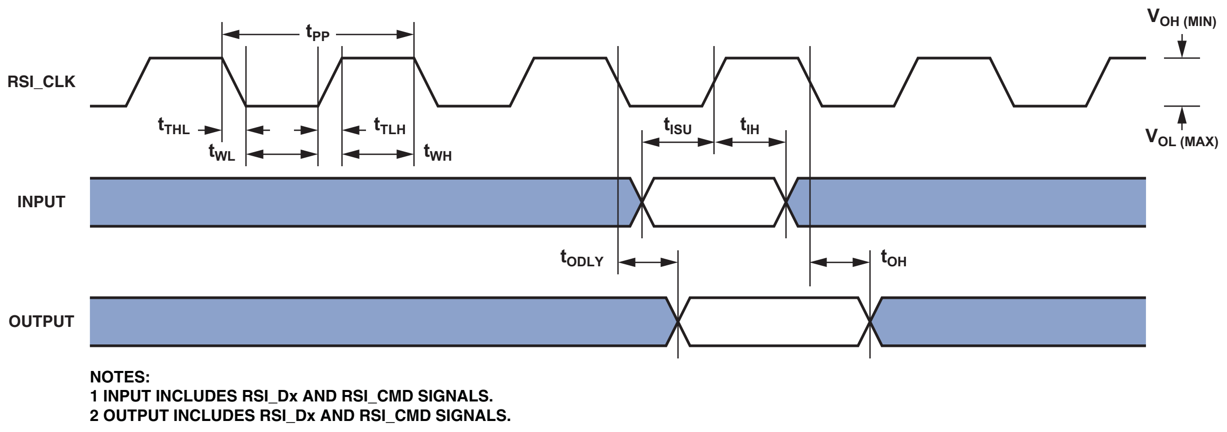


Figure 54. RSI Controller Timing

JTAG Test And Emulation Port Timing

Table 67 and Figure 58 describe JTAG port operations.

Table 67. 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	4		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]	12		12		ns
t_{HSYS}	System Inputs Hold After JTG_TCK High[1]	5		5		ns
t_{TRSTW}	$\overline{\text{JTG\_TRST}}$ Pulse Width (measured in JTG_TCK cycles)[2]	4		4		T_{CK}
Switching Characteristics						
t_{DTDO}	JTG_TDO Delay from JTG_TCK Low		18		13.5	ns
t_{DSYS}	System Outputs Delay After JTG_TCK Low[3]		22		17	ns

[1] System Inputs = DMC0_DQ00–15, DMC0_LDQS, $\overline{\text{DMC0\_LDQS}}$, DMC0_UDQS, $\overline{\text{DMC0\_UDQS}}$, PA_15–0, PB_15–0, PC_15–0, PD_15–0, PE_15–0, PF_15–0, PG_15–0, SMC0_ARDY_NORWT, $\overline{\text{SMC0\_BR}}$, SMC0_D15–0, SYS_BMODE0–2, $\overline{\text{SYS\_HWRST}}$, SYS_FAULT, $\overline{\text{SYS\_FAULT}}$, $\overline{\text{SYS\_NMI\_RESOUT}}$, SYS_PWRGD, TWI0_SCL, TWI0_SDA, TWI1_SCL, TWI1_SDA.

[2] 50 MHz Maximum.

[3] System Outputs = DMC0_A00–13, DMC0_BA0–2, $\overline{\text{DMC0\_CAS}}$, DMC0_CK, $\overline{\text{DMC0\_CK}}$, DMC0_CKE, $\overline{\text{DMC0\_CS0}}$, DMC0_DQ00–15, DMC0_LDM, DMC0_LDQS, $\overline{\text{DMC0\_LDQS}}$, DMC0_ODT, $\overline{\text{DMC0\_RAS}}$, DMC0_UDM, DMC0_UDQS, $\overline{\text{DMC0\_UDQS}}$, $\overline{\text{DMC0\_WE}}$, $\overline{\text{JTG\_EMU}}$, PA_15–0, PB_15–0, PC_15–0, PD_15–0, PE_15–0, PF_15–0, PG_15–0, $\overline{\text{SMC0\_AMS0}}$, $\overline{\text{SMC0\_AOE\_NORDV}}$, $\overline{\text{SMC0\_ARE}}$, $\overline{\text{SMC0\_AWE}}$, SMC0_A01, SMC0_A02, SMC0_D15–0, SYS_CLKOUT, SYS_FAULT, $\overline{\text{SYS\_FAULT}}$, $\overline{\text{SYS\_NMI\_RESOUT}}$.

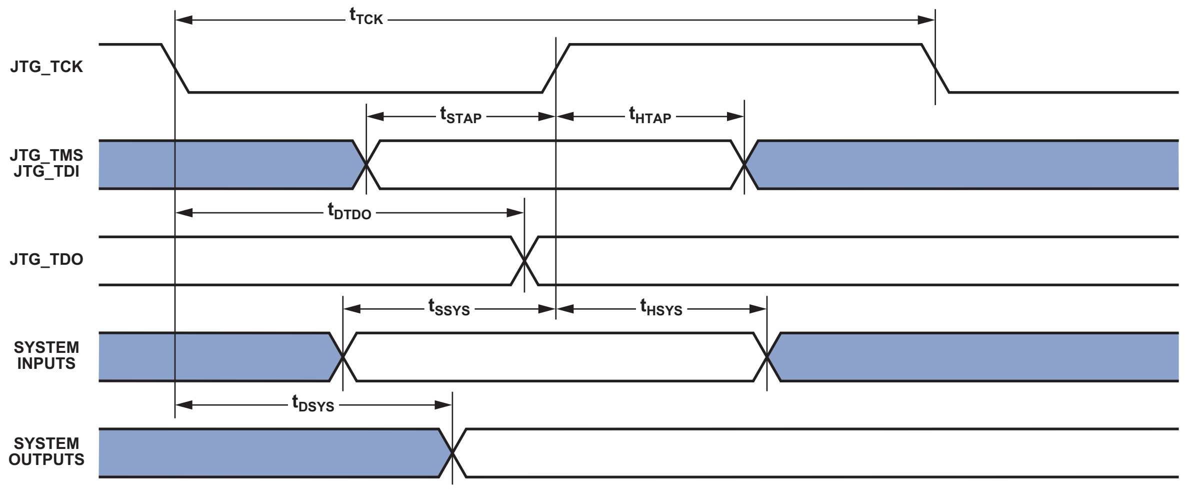


Figure 58. JTAG Port Timing

The time t_{DECAY} is calculated with test loads C_L and I_L, and with ΔV equal to 0.25 V for V_{DDEXT}/V_{DDMEM} (nominal) = 2.5 V/3.3 V and 0.15 V for V_{DDEXT}/V_{DDMEM} (nominal) = 1.8V.

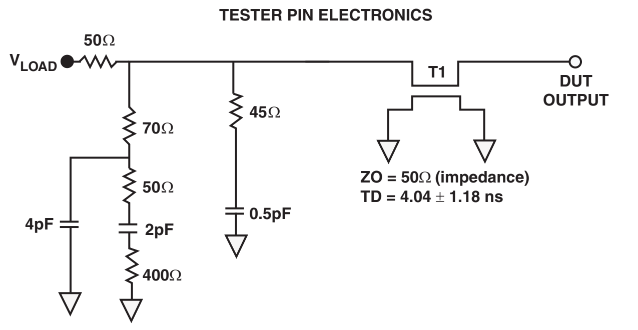
The time $t_{DIS\_MEASURED}$ is the interval from when the reference signal switches, to when the output voltage decays ΔV from the measured output high or output low voltage.

Example System Hold Time Calculation

To determine the data output hold time in a particular system, first calculate t_{DECAY} using the equation given above. Choose ΔV to be the difference between the processor's output voltage and the input threshold for the device requiring the hold time. C_L is the total bus capacitance (per data line), and I_L is the total leakage or three-state current (per data line). The hold time is t_{DECAY} plus the various output disable times as specified in the Timing Specifications on Page 60.

Capacitive Loading

Output delays and holds are based on standard capacitive loads of an average of 6 pF on all pins (see Figure 67). V_{LOAD} is equal to $(V_{DD\_EXT})/2$.



NOTES:
THE WORST CASE TRANSMISSION LINE DELAY IS SHOWN AND CAN BE USED FOR THE OUTPUT TIMING ANALYSIS TO REFELECT THE TRANSMISSION LINE EFFECT AND MUST BE CONSIDERED. THE TRANSMISSION LINE (TD) IS FOR LOAD ONLY AND DOES NOT AFFECT THE DATA SHEET TIMING SPECIFICATIONS.

ANALOG DEVICES RECOMMENDS USING THE IBIS MODEL TIMING FOR A GIVEN SYSTEM REQUIREMENT. IF NECESSARY, A SYSTEM MAY INCORPORATE EXTERNAL DRIVERS TO COMPENSATE FOR ANY TIMING DIFFERENCES.

Figure 67. Equivalent Device Loading for AC Measurements (Includes All Fixtures)

The graphs of Figure 68 through Figure 70 show how output rise and fall times vary with capacitance. The delay and hold specifications given should be derated by a factor derived from these figures. The graphs in these figures may not be linear outside the ranges shown.

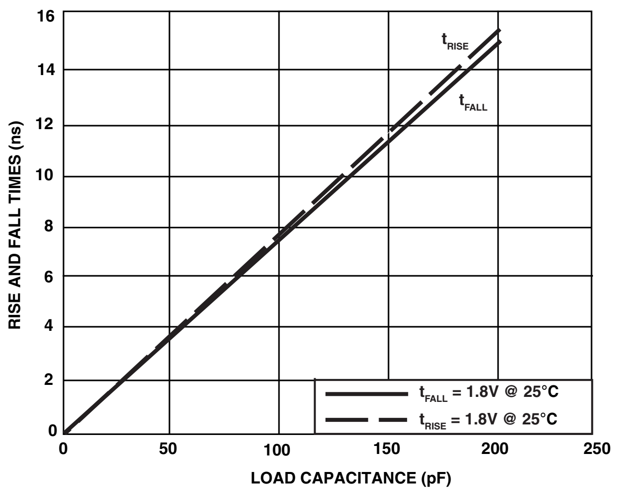


Figure 68. Driver Type A Typical Rise and Fall Times (10%-90%) vs. Load Capacitance (V_{DD_EXT} = 1.8 V)

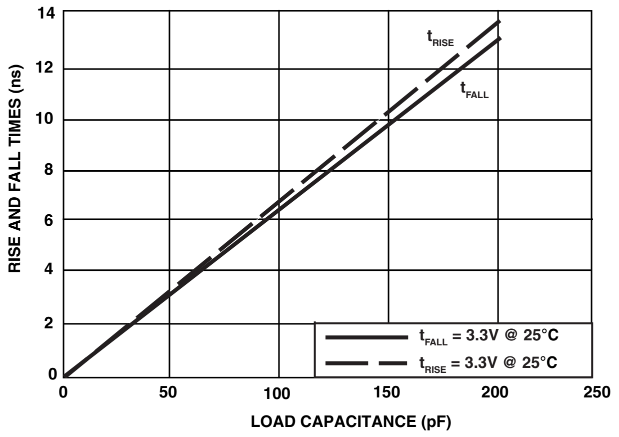


Figure 69. Driver Type A Typical Rise and Fall Times (10%-90%) vs. Load Capacitance (V_{DD_EXT} = 3.3 V)

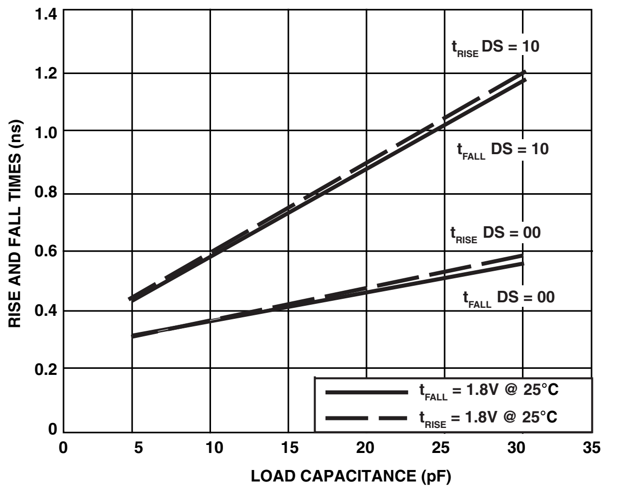


Figure 70. Driver Type B & C Typical Rise and Fall Times (10%-90%) vs. Load Capacitance (V_{DD_DMC} = 1.8 V)