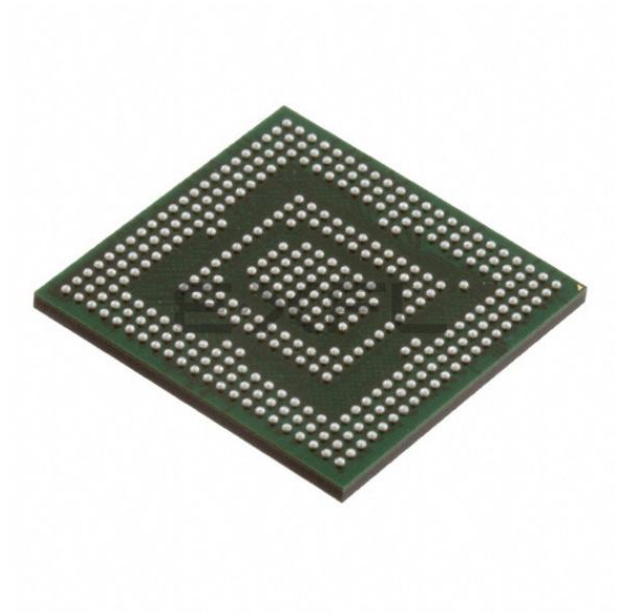


Welcome to E-XFL.COM

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	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	-40°C ~ 105°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/adbf608wcbcz502

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.

- 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.

CrossCore Embedded Studio or VisualDSP++ installed (sold separately), engineers can develop software for supported EZ-KITs or any custom system utilizing supported Analog Devices processors.

Software Add-Ins for CrossCore Embedded Studio

Analog Devices offers software add-ins which seamlessly integrate with CrossCore Embedded Studio to extend its capabilities and reduce development time. Add-ins include board support packages for evaluation hardware, various middleware packages, and algorithmic modules. Documentation, help, configuration dialogs, and coding examples present in these add-ins are viewable through the CrossCore Embedded Studio IDE once the add-in is installed.

Board Support Packages for Evaluation Hardware

Software support for the EZ-KIT Lite evaluation boards and EZ-Extender daughter cards is provided by software add-ins called Board Support Packages (BSPs). The BSPs contain the required drivers, pertinent release notes, and select example code for the given evaluation hardware. A download link for a specific BSP is located on the web page for the associated EZ-KIT or EZ-Extender product. The link is found in the *Product Download* area of the product web page.

Middleware Packages

Analog Devices separately offers middleware add-ins such as real time operating systems, file systems, USB stacks, and TCP/IP stacks. For more information see the following web pages:

- www.analog.com/ucos3
- www.analog.com/ucfs
- www.analog.com/ucusbd
- www.analog.com/lwip

Algorithmic Modules

To speed development, Analog Devices offers add-ins that perform popular audio and video processing algorithms. These are available for use with both CrossCore Embedded Studio and VisualDSP++. For more information visit www.analog.com and search on "Blackfin software modules".

Designing an Emulator-Compatible DSP Board (Target)

For embedded system test and debug, Analog Devices provides a family of emulators. On each JTAG DSP, Analog Devices supplies an IEEE 1149.1 JTAG Test Access Port (TAP). In-circuit emulation is facilitated by use of this JTAG interface. The emulator accesses the processor's internal features via the processor's TAP, allowing the developer to load code, set breakpoints, and view variables, memory, and registers. The processor must be halted to send data and commands, but once an operation is completed by the emulator, the DSP system is set to run at full speed with no impact on system timing. The emulators require the target board to include a header that supports connection of the DSP's JTAG port to the emulator.

For details on target board design issues including mechanical layout, single processor connections, signal buffering, signal termination, and emulator pod logic, see the *EE-68: Analog Devices JTAG Emulation Technical Reference* on the Analog Devices website (www.analog.com)—use site search on "EE-68." This document is updated regularly to keep pace with improvements to emulator support.

ADDITIONAL INFORMATION

The following publications that describe the ADSP-BF606/ADSP-BF607/ADSP-BF608/ADSP-BF609 processors (and related processors) can be ordered from any Analog Devices sales office or accessed electronically on our website:

- *Getting Started With Blackfin Processors*
- *ADSP-BF60x Blackfin Processor Hardware Reference*
- *Blackfin Processor Programming Reference*
- *ADSP-BF60x Blackfin Processor Anomaly List*

RELATED SIGNAL CHAINS

A *signal chain* is a series of signal-conditioning electronic components that receive input (data acquired from sampling either real-time phenomena or from stored data) in tandem, with the output of one portion of the chain supplying input to the next. Signal chains are often used in signal processing applications to gather and process data or to apply system controls based on analysis of real-time phenomena. For more information about this term and related topics, see the "signal chain" entry in the Glossary of EE Terms on the Analog Devices website.

Analog Devices eases signal processing system development by providing signal processing components that are designed to work together well. A tool for viewing relationships between specific applications and related components is available on the www.analog.com website.

The Application Signal Chains page in the Circuits from the Lab™ site (http:\\www.analog.com\circuits) provides:

- Graphical circuit block diagram presentation of signal chains for a variety of circuit types and applications
- Drill down links for components in each chain to selection guides and application information
- Reference designs applying best practice design techniques

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

Signal Name	Description	Port	Pin Name
LP3_D6	LP3 Data 6	F	PF_14
LP3_D7	LP3 Data 7	F	PF_15
PA_00 – PA_15	PORTA Position 00 through PORTA Position 15	A	PA_00 – PA_15
PB_00 – PB_15	PORTB Position 00 through PORTB Position 15	B	PB_00 – PB_15
PC_00 – PC_15	PORTC Position 00 through PORTC Position 15	C	PC_00 – PC_15
PD_00 – PD_15	PORTD Position 00 through PORTD Position 15	D	PD_00 – PD_15
PE_00 – PE_15	PORTE Position 00 through PORTE Position 15	E	PE_00 – PE_15
PF_00 – PF_15	PORTF Position 00 through PORTF Position 15	F	PF_00 – PF_15
PG_00 – PG_15	PORTG Position 00 through PORTG Position 15	G	PG_00 – PG_15
PPI0_CLK	EPPI0 Clock	E	PE_09
PPI0_D00	EPPI0 Data 0	F	PF_00
PPI0_D01	EPPI0 Data 1	F	PF_01
PPI0_D02	EPPI0 Data 2	F	PF_02
PPI0_D03	EPPI0 Data 3	F	PF_03
PPI0_D04	EPPI0 Data 4	F	PF_04
PPI0_D05	EPPI0 Data 5	F	PF_05
PPI0_D06	EPPI0 Data 6	F	PF_06
PPI0_D07	EPPI0 Data 7	F	PF_07
PPI0_D08	EPPI0 Data 8	F	PF_08
PPI0_D09	EPPI0 Data 9	F	PF_09
PPI0_D10	EPPI0 Data 10	F	PF_10
PPI0_D11	EPPI0 Data 11	F	PF_11
PPI0_D12	EPPI0 Data 12	F	PF_12
PPI0_D13	EPPI0 Data 13	F	PF_13
PPI0_D14	EPPI0 Data 14	F	PF_14
PPI0_D15	EPPI0 Data 15	F	PF_15
PPI0_D16	EPPI0 Data 16	E	PE_03
PPI0_D17	EPPI0 Data 17	E	PE_04
PPI0_D18	EPPI0 Data 18	E	PE_00
PPI0_D19	EPPI0 Data 19	E	PE_01
PPI0_D20	EPPI0 Data 20	D	PD_12
PPI0_D21	EPPI0 Data 21	D	PD_15
PPI0_D22	EPPI0 Data 22	E	PE_02
PPI0_D23	EPPI0 Data 23	E	PE_05
PPI0_FS1	EPPI0 Frame Sync 1 (HSYNC)	E	PE_08
PPI0_FS2	EPPI0 Frame Sync 2 (VSYNC)	E	PE_07
PPI0_FS3	EPPI0 Frame Sync 3 (FIELD)	E	PE_06
PPI1_CLK	EPPI1 Clock	B	PB_14
PPI1_D00	EPPI1 Data 0	C	PC_00
PPI1_D01	EPPI1 Data 1	C	PC_01
PPI1_D02	EPPI1 Data 2	C	PC_02
PPI1_D03	EPPI1 Data 3	C	PC_03
PPI1_D04	EPPI1 Data 4	C	PC_04
PPI1_D05	EPPI1 Data 5	C	PC_05
PPI1_D06	EPPI1 Data 6	C	PC_06
PPI1_D07	EPPI1 Data 7	C	PC_07

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

Signal Name	Description	Port	Pin Name
UART0_RX	UART0 Receive	D	PD_08
UART0_TX	UART0 Transmit	D	PD_07
UART1_CTS	UART1 Clear to Send	G	PG_13
UART1_RTS	UART1 Request to Send	G	PG_10
UART1_RX	UART1 Receive	G	PG_14
UART1_TX	UART1 Transmit	G	PG_15
USB0_CLKIN	USB0 Clock/Crystal Input	Not Muxed	USB0_CLKIN
USB0_DM	USB0 Data –	Not Muxed	USB0_DM
USB0_DP	USB0 Data +	Not Muxed	USB0_DP
USB0_ID	USB0 OTG ID	Not Muxed	USB0_ID
USB0_VBC	USB0 VBUS Control	Not Muxed	USB0_VBC
USB0_VBUS	USB0 Bus Voltage	Not Muxed	USB0_VBUS
VDD_DMC	VDD for DMC	Not Muxed	VDD_DMC
VDD_EXT	External VDD	Not Muxed	VDD_EXT
VDD_INT	Internal VDD	Not Muxed	VDD_INT
VDD_TD	VDD for Thermal Diode	Not Muxed	VDD_TD
VDD_USB	VDD for USB	Not Muxed	VDD_USB
VREF_DMC	VREF for DMC	Not Muxed	VREF_DMC

Table 15. ADSP-BF60x Designer Quick Reference (Continued)

Signal Name	Type	Driver Type	Int Term	Reset Term	Reset Drive	Hiber Term	Hiber Drive	Power Domain	Description and Notes
DMC0_A11	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Address 11. Notes: No notes.
DMC0_A12	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Address 12. Notes: No notes.
DMC0_A13	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Address 13. Notes: No notes.
DMC0_BA0	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Bank Address Input 0. Notes: No notes.
DMC0_BA1	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Bank Address Input 1. Notes: No notes.
DMC0_BA2	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Bank Address Input 2. Notes: For LPDDR, leave unconnected.
$\overline{\text{DMC0\_CAS}}$	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Column Address Strobe. Notes: No notes.
DMC0_CK	I/O	C	none	none	L	none	L	VDD_DMC	Desc: DMC0 Clock. Notes: No notes.
$\overline{\text{DMC0\_CK}}$	I/O	C	none	none	L	none	L	VDD_DMC	Desc: DMC0 Clock (complement). Notes: No notes.
DMC0_CKE	I/O	B	none	none	L	none	L	VDD_DMC	Desc: DMC0 Clock enable. Notes: No notes.
$\overline{\text{DMC0\_CS0}}$	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Chip Select 0. Notes: No notes.
DMC0_DQ00	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Data 0. Notes: No notes.
DMC0_DQ01	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Data 1. Notes: No notes.
DMC0_DQ02	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Data 2. Notes: No notes.
DMC0_DQ03	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Data 3. Notes: No notes.
DMC0_DQ04	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Data 4. Notes: No notes.
DMC0_DQ05	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Data 5. Notes: No notes.
DMC0_DQ06	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Data 6. Notes: No notes.
DMC0_DQ07	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Data 7. Notes: No notes.
DMC0_DQ08	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Data 8. Notes: No notes.
DMC0_DQ09	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Data 9. Notes: No notes.
DMC0_DQ10	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Data 10. Notes: No notes.
DMC0_DQ11	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Data 11. Notes: No notes.
DMC0_DQ12	I/O	B	none	none	none	none	none	VDD_DMC	Desc: DMC0 Data 12. Notes: No notes.

Table 15. ADSP-BF60x Designer Quick Reference (Continued)

Signal Name	Type	Driver Type	Int Term	Reset Term	Reset Drive	Hiber Term	Hiber Drive	Power Domain	Description and Notes
PF_03	I/O	A	wk	wk	none	wk	none	VDD_EXT	Desc: PF Position 3 | PWM0 Channel B High Side | EPPI0 Data 3 | LP2 Data 3. Notes: No notes.
PF_04	I/O	A	wk	wk	none	wk	none	VDD_EXT	Desc: PF Position 4 | PWM0 Channel C Low Side | EPPI0 Data 4 | LP2 Data 4. Notes: No notes.
PF_05	I/O	A	wk	wk	none	wk	none	VDD_EXT	Desc: PF Position 5 | PWM0 Channel C High Side | EPPI0 Data 5 | LP2 Data 5. Notes: No notes.
PF_06	I/O	A	wk	wk	none	wk	none	VDD_EXT	Desc: PF Position 6 | PWM0 Channel D Low Side | EPPI0 Data 6 | LP2 Data 6. Notes: No notes.
PF_07	I/O	A	wk	wk	none	wk	none	VDD_EXT	Desc: PF Position 7 | PWM0 Channel D High Side | EPPI0 Data 7 | LP2 Data 7. Notes: No notes.
PF_08	I/O	A	wk	wk	none	wk	none	VDD_EXT	Desc: PF Position 8 | SPI1 Slave Select Output b | EPPI0 Data 8 | LP3 Data 0. Notes: No notes.
PF_09	I/O	A	wk	wk	none	wk	none	VDD_EXT	Desc: PF Position 9 | SPI1 Slave Select Output b | EPPI0 Data 9 | LP3 Data 1. Notes: No notes.
PF_10	I/O	A	wk	wk	none	wk	none	VDD_EXT	Desc: PF Position 10 | ACM0 Address 4 | EPPI0 Data 10 | LP3 Data 2. Notes: No notes.
PF_11	I/O	A	wk	wk	none	wk	none	VDD_EXT	Desc: PF Position 11 | EPPI0 Data 11 | LP3 Data 3 | PWM0 Shutdown Input. Notes: No notes.
PF_12	I/O	A	wk	wk	none	wk	none	VDD_EXT	Desc: PF Position 12 | ACM0 Address 2 | EPPI0 Data 12 | LP3 Data 4. Notes: No notes.
PF_13	I/O	A	wk	wk	none	wk	none	VDD_EXT	Desc: PF Position 13 | ACM0 Address 3 | EPPI0 Data 13 | LP3 Data 5. Notes: No notes.
PF_14	I/O	A	wk	wk	none	wk	none	VDD_EXT	Desc: PF Position 14 | EPPI0 Data 14 | ACM0 Address 0 | LP3 Data 6. Notes: No notes.
PF_15	I/O	A	wk	wk	none	wk	none	VDD_EXT	Desc: PF Position 15 | ACM0 Address 1 | EPPI0 Data 15 | LP3 Data 7. Notes: No notes.
PG_00	I/O	A	wk	wk	none	wk	none	VDD_EXT	Desc: PG Position 0 | PWM1 Channel B High Side | RSI0 Data 2 | ETH1 Receive Data 0. Notes: Has an optional internal pull-up resistor for use with RSI. See the RSI chapter in the processor hardware reference for more details.
PG_01	I/O	A	wk	wk	none	wk	none	VDD_EXT	Desc: PG Position 1 | SPORT2 Channel A Frame Sync | TIMER0 Timer 2 | CAN0 Transmit. Notes: No notes.

Clock Related Operating Conditions

Table 17 describes the core clock, system clock, and peripheral clock timing requirements. The data presented in the tables applies to all speed grades (found in Automotive Products on Page 112 and Ordering Guide on Page 112) except where expressly noted. Figure 8 provides a graphical representation of the various clocks and their available divider values.

Table 17. Clock Operating Conditions

Parameter		Restriction	Min	Typ	Max	Unit
f_{CCLK}	Core Clock Frequency	$f_{CCLK} \geq f_{SYSCLK}$			500	MHz
f_{SYSCLK}	SYSCLK Frequency				250	MHz
f_{SCLK0}	SCLK0 Frequency[1]	$f_{SYSCLK} \geq f_{SCLK0}$	30		125	MHz
f_{SCLK1}	SCLK1 Frequency	$f_{SYSCLK} \geq f_{SCLK1}$			125	MHz
f_{DCLK}	DDR2/LPDDR Clock Frequency	$f_{SYSCLK} \geq f_{DCLK}$			250	MHz
f_{OCLK}	Output Clock Frequency				125	MHz
$f_{SYS\_CLKOUTJ}$	SYS_CLKOUT Period Jitter[2, 3]			±1		%
f_{PVPCLK}	PVP Clock Frequency				83.3	MHz
$f_{NRCLKPROG}$	Programmed NOR Burst Clock				66.67	MHz
$f_{PCLKPROG}$	Programmed PPI Clock When Transmitting Data and Frame Sync				83.3	MHz
$f_{PCLKPROG}$	Programmed PPI Clock When Receiving Data or Frame Sync				62.5	MHz
$f_{PCLKEXT}$	External PPI Clock When Receiving Data and Frame Sync[4, 5]	$f_{PCLKEXT} \leq f_{SCLK0}$			83.3	MHz
$f_{PCLKEXT}$	External PPI Clock Transmitting Data or Frame Sync[4, 5]	$f_{PCLKEXT} \leq f_{SCLK0}$			58.8	MHz
$f_{LCLKTPROG}$	Programmed Link Port Transmit Clock				83.3	MHz
$f_{LCLKREXT}$	External Link Port Receive Clock[4, 5]	$f_{LCLKEXT} \leq f_{SCLK0}$			83.3	MHz
$f_{SPTCLKPROG}$	Programmed SPT Clock When Transmitting Data and Frame Sync				83.3	MHz
$f_{SPTCLKPROG}$	Programmed SPT Clock When Receiving Data or Frame Sync				62.5	MHz
$f_{SPTCLKEXT}$	External SPT Clock When Receiving Data and Frame Sync[4, 5]	$f_{SPTCLKEXT} \leq f_{SCLK1}$			83.3	MHz
$f_{SPTCLKEXT}$	External SPT Clock Transmitting Data or Frame Sync[4, 5]	$f_{SPTCLKEXT} \leq f_{SCLK1}$			58.8	MHz
$f_{SPICLKPROG}$	Programmed SPI Clock When Transmitting Data				83.3	MHz
$f_{SPICLKPROG}$	Programmed SPI Clock When Receiving Data				75	MHz
$f_{SPICLKEXT}$	External SPI Clock When Receiving Data[4, 5]	$f_{SPICLKEXT} \leq f_{SCLK1}$			83.3	MHz
$f_{SPICLKEXT}$	External SPI Clock When Transmitting Data[4, 5]	$f_{SPICLKEXT} \leq f_{SCLK1}$			58.8	MHz
$f_{ACLKPROG}$	Programmed ACM Clock				62.5	MHz

[1] The minimum frequency for SCLK0 applies only when the USB is used.
[2] SYS_CLKOUT jitter is dependent on the application system design including pin switching activity, board layout, and the jitter characteristics of the SYS_CLKIN source. Due to the dependency on these factors the measured jitter may be higher or lower than this typical specification for each end application.
[3] The value in the Typ field is the percentage of the SYS_CLKOUT period.
[4] The maximum achievable frequency for any peripheral in external clock mode is dependent on being able to meet the setup and hold times in the AC timing specifications section for that peripheral. Pay particular attention to setup and hold times for VDD_EXT = 1.8 V which may preclude the maximum frequency listed here.
[5] The peripheral external clock frequency must also be less than or equal to the f_{SCLK} (f_{SCLK0} or f_{SCLK1}) that clocks the peripheral.

Table 18. Phase-Locked Loop Operating Conditions

Parameter		Min	Max	Unit
f_{PLLCLK}	PLL Clock Frequency	250	1000	MHz

Table 20. Activity Scaling Factors (ASF)

I_{DDINT} Power Vector	ASF
$I_{DD-PEAK}$	1.34
$I_{DD-HIGH}$	1.25
$I_{DD-FULL-ON-TYP}$	1.00
I_{DD-APP}	0.86
I_{DD-NOP}	0.72
$I_{DD-IDLE}$	0.14

Table 21. Static Current—$I_{DD\_DEEPSLEEP}$ (mA)

	Voltage (V_{DD_INT})						
T_J (°C)	1.190	1.200	1.225	1.250	1.275	1.300	1.320
–40	1.7	1.8	2.2	2.5	2.7	3.1	3.4
–20	4.0	4.2	4.6	5.1	5.6	6.2	6.8
0	8.4	9.0	9.6	10.6	11.5	12.5	13.4
25	19.0	19.8	21.5	23.2	25.3	27.2	29.0
40	29.9	31.7	34.4	36.8	40.0	42.8	45.4
55	46.6	48.9	52.4	56.4	60.6	65.0	68.1
70	66.4	70.4	75.5	80.6	86.2	92.4	97.9
85	93.9	99.3	105.9	113.0	120.7	128.9	136.4
100	137.2	144.2	153.6	163.4	173.9	185.1	194.1
105	153.8	162.4	172.5	183.4	195.2	207.5	217.5
115	193.3	203.7	216.2	229.5	243.9	258.6	271.1
125	236.1	247.2	261.8	277.3	294.0	311.9	326.4

Table 22. $I_{DDINT\_PVP\_DYN}$ (mA)

PVP Activity Level	PVPSF (PVP Scaling Factor)
High	42.4
Medium	20
Low	0

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}.

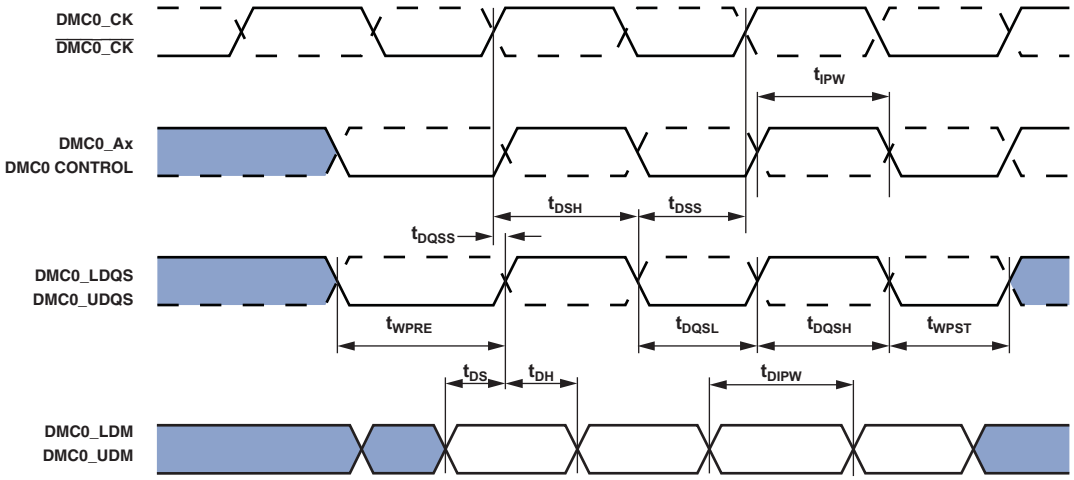


Figure 21. DDR2 SDRAM Controller Output AC Timing

Serial Ports

To determine whether 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 36 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 65535:

$$f_{SPTCLKPROG} = \frac{f_{SCLK1}}{(CLKDIV + 1)}$$

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

Table 46. 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]	2		2		ns
t_{HFSE}	Frame Sync Hold After SPT_CLK (Externally Generated Frame Sync in either Transmit or Receive Mode)[1]	2.7		2.7		ns
t_{SDRE}	Receive Data Setup Before Receive SPT_CLK[1]	2		2		ns
t_{HDRE}	Receive Data Hold After SPT_CLK[1]	2.7		2.7		ns
t_{SCLKW}	SPT_CLK Width[2]	$(0.5 \times t_{SPTCLKEXT}) - 1.5$		$(0.5 \times t_{SPTCLKEXT}) - 1.5$		ns
t_{SPTCLK}	SPT_CLK Period[2]	$t_{SPTCLKEXT} - 1.5$		$t_{SPTCLKEXT} - 1.5$		ns
Switching Characteristics						
t_{DFSE}	Frame Sync Delay After SPT_CLK (Internally Generated Frame Sync in either Transmit or Receive Mode)[3]		19.3		14.5	ns
t_{HOFSE}	Frame Sync Hold After SPT_CLK (Internally Generated Frame Sync in either Transmit or Receive Mode)[3]	2		2		ns
t_{DDTE}	Transmit Data Delay After Transmit SPT_CLK[3]		18.8		14	ns
t_{HDTE}	Transmit Data Hold After Transmit SPT_CLK[3]	2		2		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 17 on Page 53 in Clock Related Operating Conditions.

[3] Referenced to drive edge.

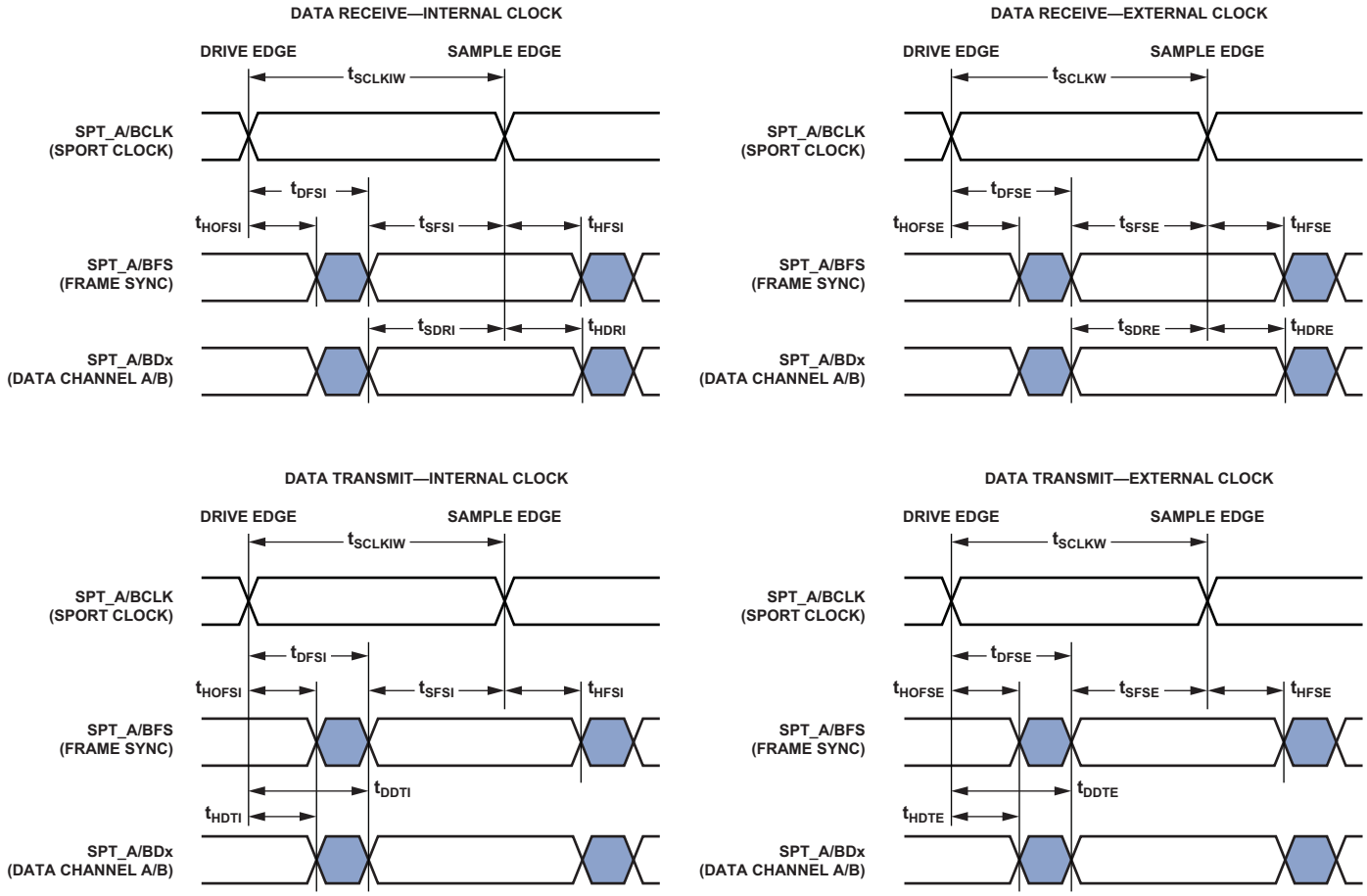


Figure 36. Serial Ports

Table 50. Serial Ports—External Late Frame Sync

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
Switching Characteristics						
$t_{DDTLFSE}$	Data Delay from Late External Transmit Frame Sync or External Receive Frame Sync with MCE = 1, MFD = 0[1]		18.8		14	ns
$t_{DDTENFS}$	Data Enable for MCE = 1, MFD = 0[1]	0.5		0.5		ns

[1] The $t_{DDTLFSE}$ and $t_{DDTENFS}$ parameters apply to left-justified as well as standard serial mode, and MCE = 1, MFD = 0.

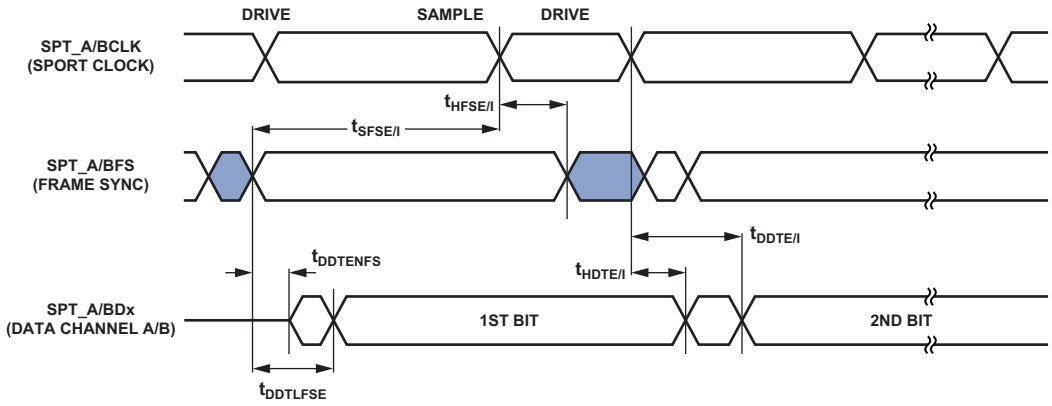


Figure 39. External Late Frame Sync

Serial Peripheral Interface (SPI) Port—Master Timing

Table 51 and Figure 40 describe SPI port master operations.

When internally generated, the programmed SPI clock ($f_{SPICLKPROG}$) frequency in MHz is set by the following equation where BAUD is a field in the SPI_CLK register that can be set from 0 to 65535:

$$f_{SPICLKPROG} = \frac{f_{SCLK1}}{(BAUD + 1)}$$

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

Note that:

- In dual mode data transmit the SPI_MISO signal is also an output.
- In quad mode data transmit the SPI_MISO, SPI_D2, and SPI_D3 signals are also outputs.
- In dual mode data receive the SPI_MOSI signal is also an input.
- In quad mode data receive the SPI_MOSI, SPI_D2, and SPI_D3 signals are also inputs.
- To add additional frame delays see the documentation for the SPI_DLY register in the hardware reference manual.

Table 51. Serial Peripheral Interface (SPI) Port—Master 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_{SSPIDM}	Data Input Valid to SPI_CLK Edge (Data Input Setup)	4.6		3.2		ns
t_{HSPIDM}	SPI_CLK Sampling Edge to Data Input Invalid	1.3		1.3		ns
Switching Characteristics						
t_{SDSCIM}	$\overline{SPI\_SEL}$ low to First SPI_CLK Edge	$0.5 \times t_{SCLK1} - 2$		$0.5 \times t_{SCLK1} - 2$		ns
t_{SPICHM}	SPI_CLK High Period[1]	$0.5 \times t_{SPICLKPROG} - 1.5$		$0.5 \times t_{SPICLKPROG} - 1.5$		ns
t_{SPICLM}	SPI_CLK Low Period[1]	$0.5 \times t_{SPICLKPROG} - 1.5$		$0.5 \times t_{SPICLKPROG} - 1.5$		ns
t_{SPICLK}	SPI_CLK Period[1]	$t_{SPICLKPROG} - 1.5$		$t_{SPICLKPROG} - 1.5$		ns
t_{HDSM}	Last SPI_CLK Edge to $\overline{SPI\_SEL}$ High	$(0.5 \times t_{SCLK1}) - 1.5$		$(0.5 \times t_{SCLK1}) - 1.5$		ns
t_{SPITDM}	Sequential Transfer Delay	$t_{SCLK1} - 1.5$		$t_{SCLK1} - 1.5$		ns
$t_{DDSPIDM}$	SPI_CLK Edge to Data Out Valid (Data Out Delay)		2.6		2.6	ns
$t_{HDSPIDM}$	SPI_CLK Edge to Data Out Invalid (Data Out Hold)	–1		–1		ns

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

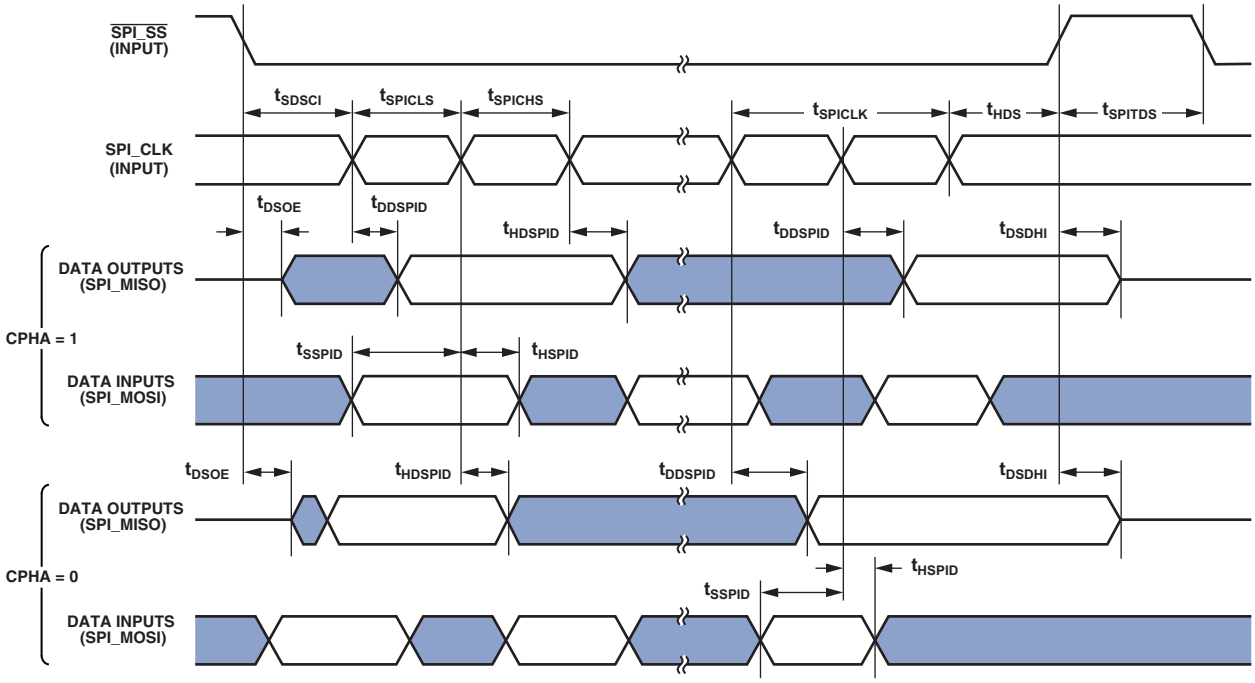


Figure 41. Serial Peripheral Interface (SPI) Port—Slave Timing

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

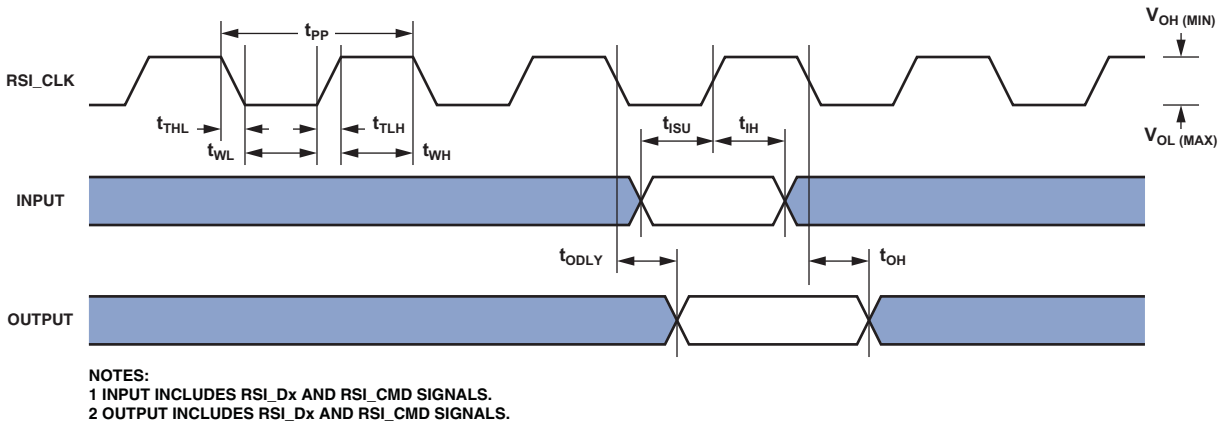


Figure 54. RSI Controller Timing

Table 66. 10/100 Ethernet MAC Controller Timing: RMII Station Management

Parameter[1]		V_{DD_EXT} 1.8 V/3.3 V Nominal Min	Max	Unit
Timing Requirements				
t_{MDIOS}	ETHx_MDIO Input Valid to ETHx_MDC Rising Edge (Setup)	14		ns
t_{MDCIH}	ETHx_MDC Rising Edge to ETHx_MDIO Input Invalid (Hold)	0		ns
Switching Characteristics				
t_{MDCOV}	ETHx_MDC Falling Edge to ETHx_MDIO Output Valid		t_{SCLK0} + 5	ns
t_{MDCOH}	ETHx_MDC Falling Edge to ETHx_MDIO Output Invalid (Hold)	t_{SCLK0} −1		ns

[1] ETHx_MDC/ETHx_MDIO is a 2-wire serial bidirectional port for controlling one or more external PHYs. ETHx_MDC is an output clock whose minimum period is programmable as a multiple of the system clock SCLK0. ETHx_MDIO is a bidirectional data line.

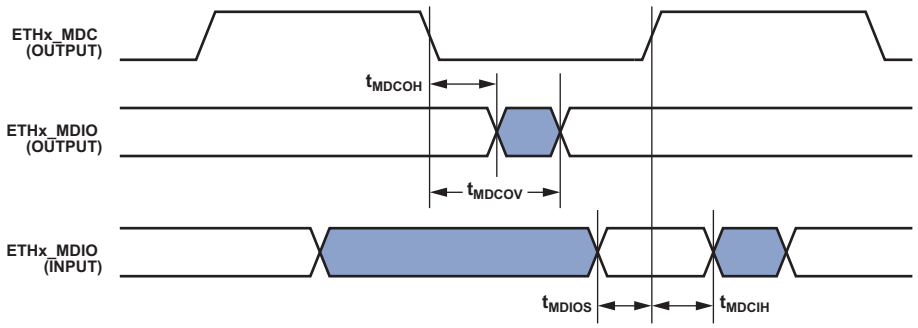


Figure 57. 10/100 Ethernet MAC Controller Timing: RMII Station Management

OUTPUT DRIVE CURRENTS

Figure 59 through Figure 64 show typical current-voltage characteristics for the output drivers of the ADSP-BF60x Blackfin processors. The curves represent the current drive capability of the output drivers as a function of output voltage.

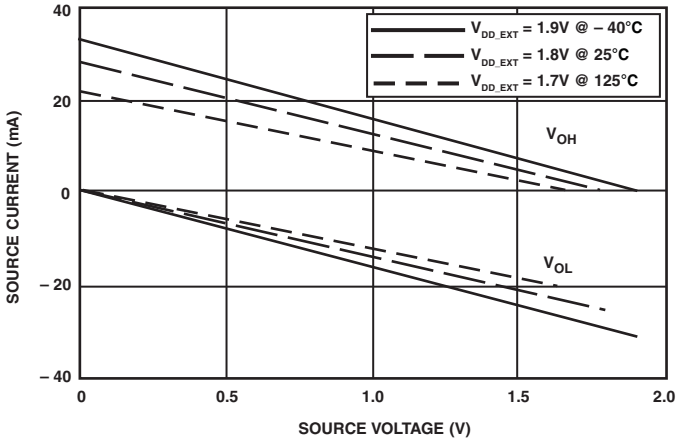


Figure 59. Driver Type A Current (1.8 V V_{DD_EXT})

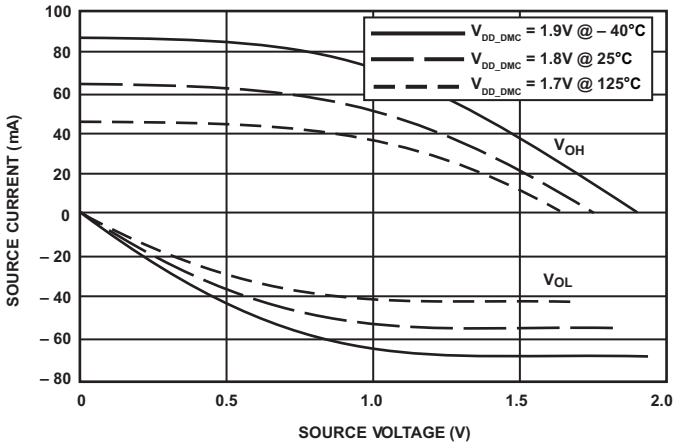


Figure 61. Driver Type B Current (1.8 V V_{DD_DMC})

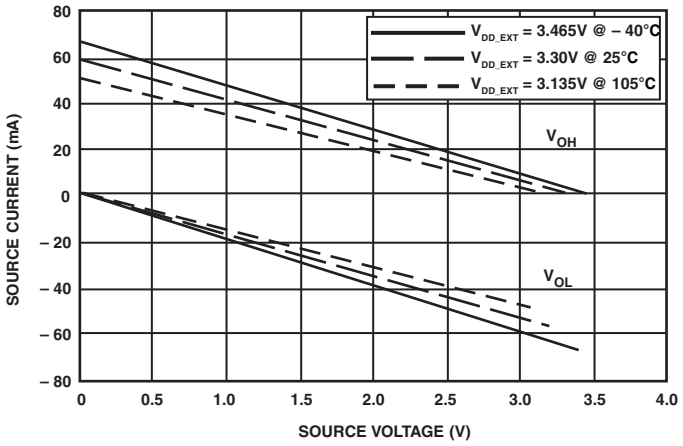


Figure 60. Driver Type A Current (3.3 V V_{DD_EXT})

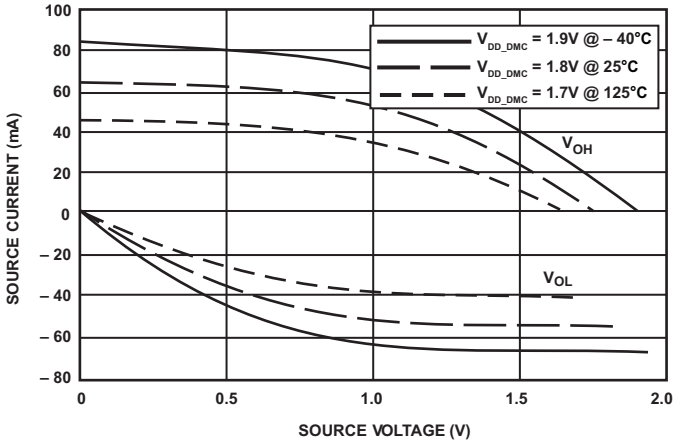


Figure 62. Driver Type C Current (1.8 V V_{DD_DMC})

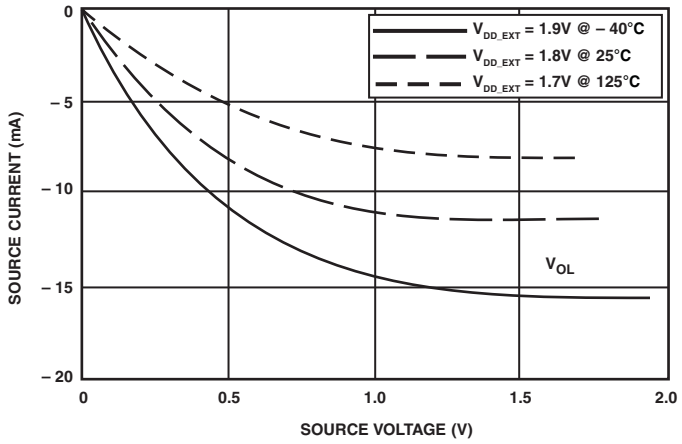


Figure 63. Driver Type D Current (1.8 V V_{DD_EXT})

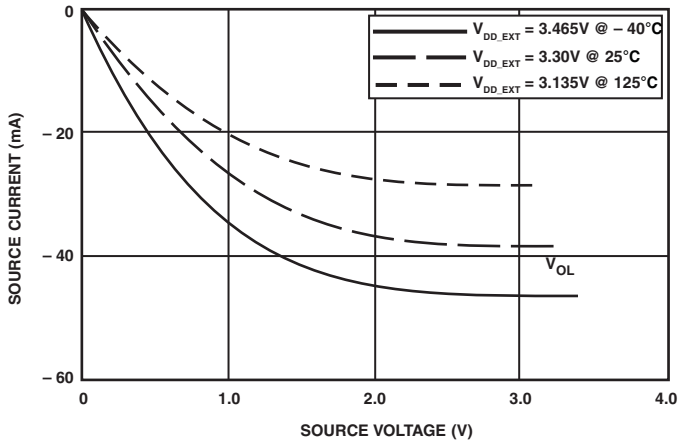


Figure 64. Driver Type D Current (3.3 V V_{DD_EXT})

TEST CONDITIONS

All Timing Requirements appearing in this data sheet were measured under the conditions described in this section. Figure 65 shows the measurement point for AC measurements (except output enable/disable). The measurement point V_{MEAS} is $V_{DDEXT}/2$ or $V_{DDMEM}/2$ for V_{DDEXT}/V_{DDMEM} (nominal) = 1.8 V/ 2.5 V/3.3 V.

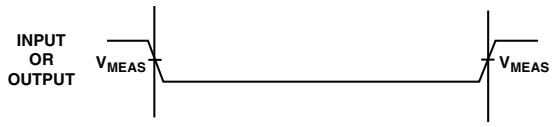


Figure 65. Voltage Reference Levels for AC Measurements (Except Output Enable/Disable)

Output Enable Time Measurement

Output balls are considered to be enabled when they have made a transition from a high impedance state to the point when they start driving.

The output enable time t_{ENA} is the interval from the point when a reference signal reaches a high or low voltage level to the point when the output starts driving as shown on the right side of Figure 66.

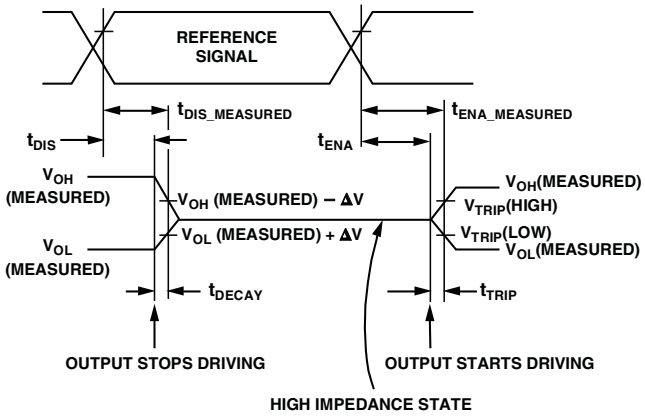


Figure 66. Output Enable/Disable

The time $t_{ENA\_MEASURED}$ is the interval *from when* the reference signal switches *to when* the output voltage reaches V_{TRIP}(high) or V_{TRIP}(low). For V_{DDEXT}/V_{DDMEM} (nominal) = 1.8 V, V_{TRIP} (high) is 1.05 V, and V_{TRIP} (low) is 0.75 V. For V_{DDEXT}/V_{DDMEM} (nominal) = 2.5 V, V_{TRIP} (high) is 1.5 V and V_{TRIP} (low) is 1.0 V. For V_{DDEXT}/V_{DDMEM} (nominal) = 3.3 V, V_{TRIP} (high) is 1.9 V, and V_{TRIP} (low) is 1.4 V. Time t_{TRIP} is the interval *from when* the output starts driving *to when* the output reaches the V_{TRIP}(high) or V_{TRIP}(low) trip voltage.

Time t_{ENA} is calculated as shown in the equation:

$$t_{ENA} = t_{ENA\_MEASURED} - t_{TRIP}$$

If multiple balls (such as the data bus) are enabled, the measurement value is that of the first ball to start driving.

Output Disable Time Measurement

Output balls are considered to be disabled when they stop driving, go into a high impedance state, and start to decay from their output high or low voltage. The output disable time t_{DIS} is the difference between $t_{DIS\_MEASURED}$ and t_{DECAY} as shown on the left side of Figure 66.

$$t_{DIS} = t_{DIS\_MEASURED} - t_{DECAY}$$

The time for the voltage on the bus to decay by ΔV is dependent on the capacitive load C_L and the load current I_L. This decay time can be approximated by the equation:

$$t_{DECAY} = (C_L \Delta V)/I_L$$

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.

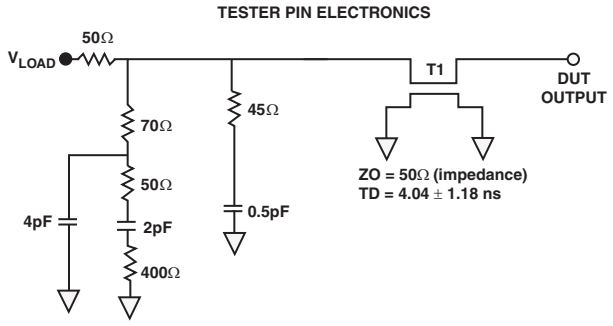
The time $t_{DIS\_MEASURED}$ is the interval from when the reference signal switches, to when the output voltage decays ΔV from the measured output high or output low voltage.

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.

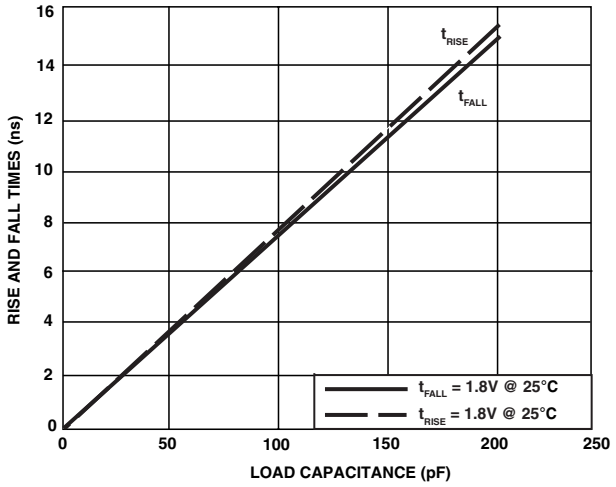


Figure 68. Driver Type A Typical Rise and Fall Times (10%-90%) vs. Load Capacitance (V_{DD_EXT} = 1.8 V)

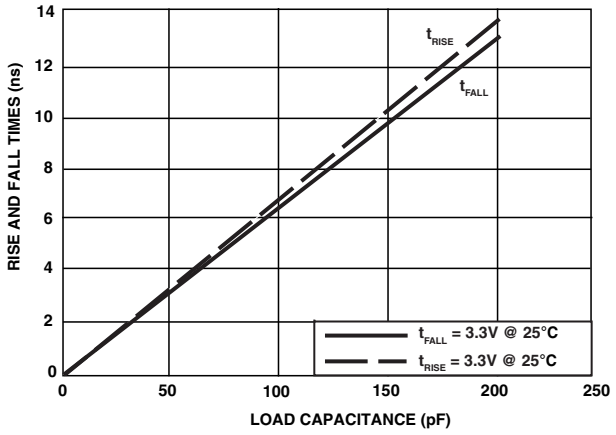


Figure 69. Driver Type A Typical Rise and Fall Times (10%-90%) vs. Load Capacitance (V_{DD_EXT} = 3.3 V)

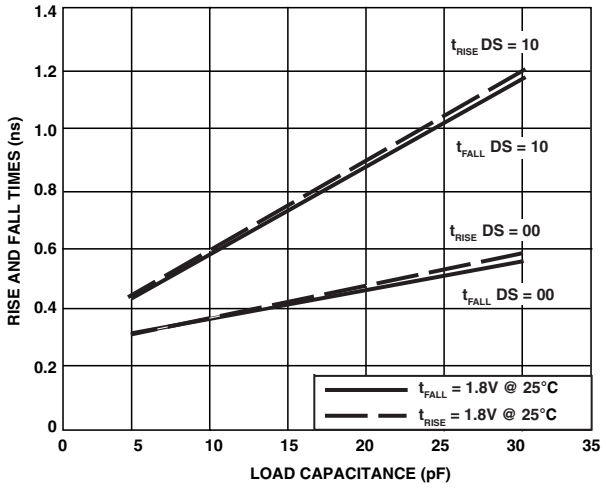


Figure 70. Driver Type B & C Typical Rise and Fall Times (10%-90%) vs. Load Capacitance (V_{DD_DMC} = 1.8 V)