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	Fixed Point
Interface	Host Interface, SPI, SSP, UART
Clock Rate	160MHz
Non-Volatile Memory	External
On-Chip RAM	160kB
Voltage - I/O	3.00V, 3.30V
Voltage - Core	2.50V
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	144-LQFP
Supplier Device Package	144-LQFP (20x20)
Purchase URL	https://www.e-xfl.com/product-detail/analog-devices/adsp-2191mkstz-160

ADSP-2191M* PRODUCT PAGE QUICK LINKS

Last Content Update: 02/23/2017

COMPARABLE PARTS

View a parametric search of comparable parts.

EVALUATION KITS

- USB-Based Emulator and High Performance USB-Based Emulator

DOCUMENTATION

Application Notes

- AN-227: Digital Control System Design with the ADSP-2100 Family
- AN-572: Overlay Linking on the ADSP-219x
- EE-06: ADSP-21xx Serial Port Startup Issues
- EE-100: ADSP-218x External Overlay Memory
- EE-102: Mode D and ADSP-218x Pin Compatibility - the FAQs
- EE-104: Setting Up Streams with the VisualDSP Debugger
- EE-11: ADSP-2181 Priority Chain & IDMA Holdoffs
- EE-115: ADSP-2189 IDMA Interface to Motorola MC68300 Family of Microprocessors
- EE-12: Interrupts and Programmable Flags on the ADSP-2185/2186
- EE-121: Porting Code from ADSP-21xx to ADSP-219x
- EE-122: Coding for Performance on the ADSP-219x
- EE-123: An Overview of the ADSP-219x Pipeline
- EE-124: Booting up the ADSP-2192
- EE-125: ADSP-218x Embedded System Software Management and In-System-Programming (ISP)
- EE-128: DSP in C++: Calling Assembly Class Member Functions From C++
- EE-129: ADSP-2192 Interprocessor Communication
- EE-130: Making Fast Transition from ADSP-21xx to ADSP-219x
- EE-131: Booting the ADSP-2191/95/96 DSPs
- EE-133: Converting From Legacy Architecture Files To Linker Description Files for the ADSP-218x
- EE-139: Interfacing the ADSP-2191 to an AD7476 via the SPI Port
- EE-142: Autobuffering, C and FFTs on the ADSP-218x
- EE-144: Creating a Master-Slave SPI Interface Between Two ADSP-2191 DSPs
- EE-145: SPI Booting of the ADSP-2191 using the Atmel AD25020N on an EZ-KIT Lite Evaluation Board
- EE-146: Implementing a Boot Manager for ADSP-218x Family DSPs
- EE-152: Using Software Overlays with the ADSP-219x and VisualDSP 2.0++
- EE-153: ADSP-2191 Programmable PLL
- EE-154: ADSP-2191 Host Port Interface

REFERENCE MATERIALS

Product Selection Guide

- ADI Complementary Parts Guide - Supervisory Devices and DSP Processors

DESIGN RESOURCES

- ADSP-2191M Material Declaration
- PCN-PDN Information
- Quality And Reliability
- Symbols and Footprints

DISCUSSIONS

View all ADSP-2191M EngineerZone Discussions.

SAMPLE AND BUY

Visit the product page to see pricing options.

TECHNICAL SUPPORT

Submit a technical question or find your regional support number.

DOCUMENT FEEDBACK

Submit feedback for this data sheet.

GENERAL DESCRIPTION

The ADSP-2191M DSP is a single-chip microcomputer optimized for digital signal processing (DSP) and other high speed numeric processing applications.

The ADSP-2191M combines the ADSP-219x family base architecture (three computational units, two data address generators, and a program sequencer) with three serial ports, two SPI-compatible ports, one UART port, a DMA controller, three programmable timers, general-purpose Programmable Flag pins, extensive interrupt capabilities, and on-chip program and data memory spaces.

The ADSP-2191M architecture is code-compatible with DSPs of the ADSP-218x family. Although the architectures are compatible, the ADSP-2191M architecture has a number of enhancements over the ADSP-218x architecture. The enhancements to computational units, data address generators, and program sequencer make the ADSP-2191M more flexible and even easier to program.

Indirect addressing options provide addressing flexibility—premodify with no update, pre- and post-modify by an immediate 8-bit, two's-complement value and base address registers for easier implementation of circular buffering.

The ADSP-2191M integrates 64K words of on-chip memory configured as 32K words (24-bit) of program RAM, and 32K words (16-bit) of data RAM. Power-down circuitry is also provided to reduce power consumption. The ADSP-2191M is available in 144-lead LQFP and 144-ball mini-BGA packages.

Fabricated in a high speed, low power, CMOS process, the ADSP-2191M operates with a 6.25 ns instruction cycle time (160 MIPS). All instructions, except single-word instructions, execute in one processor.

The ADSP-2191M's flexible architecture and comprehensive instruction set support multiple operations in parallel. For example, in one processor cycle, the ADSP-2191M can:

- Generate an address for the next instruction fetch
- Fetch the next instruction
- Perform one or two data moves
- Update one or two data address pointers
- Perform a computational operation

These operations take place while the processor continues to:

- Receive and transmit data through two serial ports
- Receive and/or transmit data from a Host
- Receive or transmit data through the UART
- Receive or transmit data over two SPI ports
- Access external memory through the external memory interface
- Decrement the timers

DSP Core Architecture

The ADSP-2191M instruction set provides flexible data moves and multifunction (one or two data moves with a computation) instructions. Every single-word instruction can be executed in a single processor cycle. The ADSP-2191M assembly language uses an algebraic syntax for ease of coding and readability. A comprehensive set of development tools supports program development.

The functional block diagram on Page 1 shows the architecture of the ADSP-219x core. It contains three independent computational units: the ALU, the multiplier/accumulator (MAC), and the shifter. The computational units process 16-bit data from the register file and have provisions to support multiprecision computations. The ALU performs a standard set of arithmetic and logic operations; division primitives are also supported. The MAC performs single-cycle multiply, multiply/add, and multiply/subtract operations. The MAC has two 40-bit accumulators, which help with overflow. The shifter performs logical and arithmetic shifts, normalization, denormalization, and derive exponent operations. The shifter can be used to efficiently implement numeric format control, including multiword and block floating-point representations.

Register-usage rules influence placement of input and results within the computational units. For most operations, the computational units' data registers act as a data register file, permitting any input or result register to provide input to any unit for a computation. For feedback operations, the computational units let the output (result) of any unit be input to any unit on the next cycle. For conditional or multifunction instructions, there are restrictions on which data registers may provide inputs or receive results from each computational unit. For more information, see the *ADSP-219x DSP Instruction Set Reference*.

A powerful program sequencer controls the flow of instruction execution. The sequencer supports conditional jumps, subroutine calls, and low interrupt overhead. With internal loop counters and loop stacks, the ADSP-2191M executes looped code with zero overhead; no explicit jump instructions are required to maintain loops.

Two data address generators (DAGs) provide addresses for simultaneous dual operand fetches (from data memory and program memory). Each DAG maintains and updates four 16-bit address pointers. Whenever the pointer is used to access data (indirect addressing), it is pre- or post-modified by the value of one of four possible modify registers. A length value and base address may be associated with each pointer to implement automatic modulo addressing for circular buffers. Page registers in the DAGs allow circular addressing within 64K-word boundaries of each of the 256 memory pages, but these buffers may not cross page boundaries. Secondary registers duplicate all the primary registers in the DAGs; switching between primary and secondary registers provides a fast context switch.

Efficient data transfer in the core is achieved with the use of internal buses:

- Program Memory Address (PMA) Bus
- Program Memory Data (PMD) Bus
- Data Memory Address (DMA) Bus
- Data Memory Data (DMD) Bus
- DMA Address Bus
- DMA Data Bus

The two address buses (PMA and DMA) share a single external address bus, allowing memory to be expanded off-chip, and the two data buses (PMD and DMD) share a single external data bus. Boot memory space and I/O memory space also share the external buses.

Program memory can store both instructions and data, permitting the ADSP-2191M to fetch two operands in a single cycle, one from program memory and one from data memory. The DSP's dual memory buses also let the ADSP-219x core fetch an operand from data memory and the next instruction from program memory in a single cycle.

DSP Peripherals Architecture

The functional block diagram on Page 1 shows the DSP's on-chip peripherals, which include the external memory interface, Host port, serial ports, SPI-compatible ports, UART port, JTAG test and emulation port, timers, flags, and interrupt controller. These on-chip peripherals can connect to off-chip devices as shown in Figure 1.

The ADSP-2191M has a 16-bit Host port with DMA capability that lets external Hosts access on-chip memory. This 24-pin parallel port consists of a 16-pin multiplexed data/address bus and provides a lowservice overhead data move capability. Configurable for 8 or 16 bits, this port provides a glueless interface to a wide variety of 8- and 16-bit microcontrollers. Two chip-selects provide Hosts access to the DSP's entire memory map. The DSP is bootable through this port.

The ADSP-2191M also has an external memory interface that is shared by the DSP's core, the DMA controller, and DMA capable peripherals, which include the UART, SPORT0, SPORT1, SPORT2, SPI0, SPI1, and the Host port. The external port consists of a 16-bit data bus, a 22-bit address bus, and control signals. The data bus is configurable to provide an 8- or 16-bit interface to external memory. Support for word packing lets the DSP access 16- or 24-bit words from external memory regardless of the external data bus width. When configured for an 8-bit interface, the unused eight lines provide eight programmable, bidirectional general-purpose Programmable Flag lines, six of which can be mapped to software condition signals.

The memory DMA controller lets the ADSP-2191M move data and instructions from between memory spaces: internal-to-external, internal-to-internal, and external-to-external. On-chip peripherals can also use this controller for DMA transfers.

The ADSP-2191M can respond to up to seventeen interrupts at any given time: three internal (stack, emulator kernel, and power-down), two external (emulator and reset), and twelve user-defined (peripherals) interrupts. The programmer assigns a peripheral to one of the 12 user-defined interrupts. The priority of each peripheral for interrupt service is determined by these assignments.

There are three serial ports on the ADSP-2191M that provide a complete synchronous, full-duplex serial interface. This interface includes optional companding in hardware as well as a wide variety of framed or frameless data transmit and receive modes

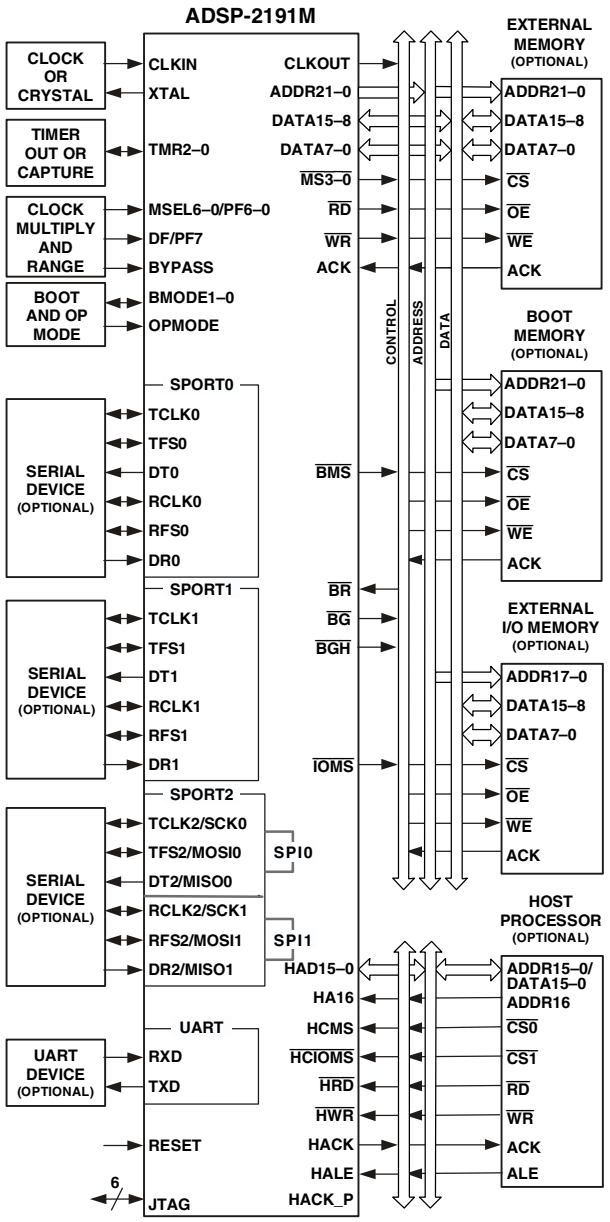

Figure 1. System Diagram

of operation. Each serial port can transmit or receive an internal or external, programmable serial clock and frame syncs. Each serial port supports 128-channel Time Division Multiplexing.

The ADSP-2191M provides up to sixteen general-purpose I/O pins, which are programmable as either inputs or outputs. Eight of these pins are dedicated-general purpose Programmable Flag pins. The other eight of them are multifunctional pins, acting as general-purpose I/O pins when the DSP connects to an 8-bit external data bus and acting as the upper eight data pins when the DSP connects to a 16-bit external data bus. These Programmable Flag pins can implement edge- or level-sensitive interrupts, some of which can be used to base the execution of conditional instructions.

Three programmable interval timers generate periodic interrupts. Each timer can be independently set to operate in one of three modes:

- Pulse Waveform Generation mode
- Pulsewidth Count/Capture mode
- External Event Watchdog mode

Each timer has one bidirectional pin and four registers that implement its mode of operation: A 7-bit configuration register, a 32-bit count register, a 32-bit period register, and a 32-bit pulsewidth register. A single status register supports all three timers. A bit in each timer's configuration register enables or disables the corresponding timer independently of the others.

Memory Architecture

The ADSP-2191M DSP provides 64K words of on-chip SRAM memory. This memory is divided into four 16K blocks located on memory Page 0 in the DSP's memory map. In addition to the internal and external memory space, the ADSP-2191M can address two additional and separate off-chip memory spaces: I/O space and boot space.

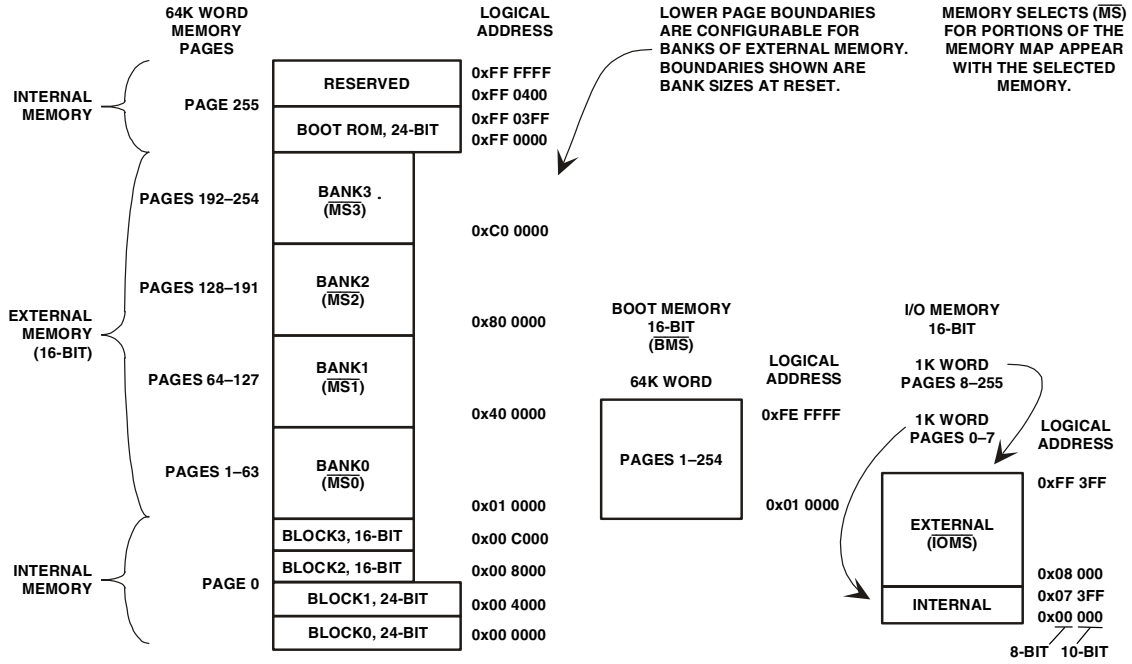

Figure 2. Memory Map

As shown in Figure 2, the DSP's two internal memory blocks populate all of Page 0. The entire DSP memory map consists of 256 pages (Pages 0–255), and each page is 64K words long. External memory space consists of four memory banks (banks 0–3) and supports a wide variety of SRAM memory devices. Each bank is selectable using the memory select pins ($\overline{MS3–0}$) and has configurable page boundaries, waitstates, and waitstate modes. The 1K word of on-chip boot-ROM populates the top of Page 255 while the remaining 254 pages are addressable off-chip. I/O memory pages differ from external memory pages in that I/O pages are 1K word long, and the external I/O pages have their own select pin ($\overline{IOMS}$). Pages 0–7 of I/O memory space reside on-chip and contain the configuration registers for the peripherals. Both the core and DMA-capable peripherals can access the DSP's entire memory map.

Internal (On-Chip) Memory

The ADSP-2191M's unified program and data memory space consists of 16M locations that are accessible through two 24-bit address buses, the PMA and DMA buses. The DSP uses slightly different mechanisms to generate a 24-bit address for each bus. The DSP has three functions that support access to the full memory map.

- The DAGs generate 24-bit addresses for data fetches from the entire DSP memory address range. Because DAG index (address) registers are 16 bits wide and hold the lower 16 bits of the address, each of the DAGs has its own 8-bit page register (DMPGx) to hold the most significant eight address bits. Before a DAG generates an address, the program must set the DAG's DMPGx register to the appropriate memory page.
- The Program Sequencer generates the addresses for instruction fetches. For relative addressing instructions, the program sequencer bases addresses for relative jumps, calls, and loops on the 24-bit Program Counter (PC). In direct addressing instructions (two-word instructions),

Host can directly access the DSP's entire memory space map, boot memory space, and internal I/O space. To access the DSP's internal memory space, a Host steals one cycle per access from the DSP. A Host access to the DSP's external memory uses the external port interface and does not stall (or steal cycles from) the DSP's core. Because a Host can access internal I/O memory space, a Host can control any of the DSP's I/O mapped peripherals.

The Host port is most efficient when using the DSP as a slave and uses DMA to automate the incrementing of addresses for these accesses. In this case, an address does not have to be transferred from the Host for every data transfer.

Host Port Acknowledge (HACK) Modes

The Host port supports a number of modes (or protocols) for generating a HACK output for the host. The host selects ACK or Ready modes using the HACK_P and HACK pins. The Host port also supports two modes for address control: Address Latch Enable (ALE) and Address Cycle Control (ACC) modes. The DSP auto-detects ALE versus ACC mode from the HALE and $\overline{\text{HWR}}$ inputs.

The Host port HACK signal polarity is selected (only at reset) as active high or active low, depending on the value driven on the HACK_P pin.The HACK polarity is stored into the Host port configuration register as a read only bit.

The DSP uses HACK to indicate to the Host when to complete an access. For a read transaction, a Host can proceed and complete an access when valid data is present in the read buffer and the Host port is not busy doing a write. For a write transactions, a Host can complete an access when the write buffer is not full and the Host port is not busy doing a write.

Two mode bits in the Host Port configuration register HPCR [7:6] define the functionality of the HACK line. HPCR6 is initialized at reset based on the values driven on HACK and HACK_P pins (shown in Table 5); HPCR7 is always cleared (0) at reset. HPCR [7:6] can be modified after reset by a write access to the Host port configuration register.

Table 5. Host Port Acknowledge Mode Selection

Values Driven At Reset		HPCR [7:6] Initial Values		Acknowledge Mode
HACK_P	HACK	Bit 7	Bit 6	
0	0	0	1	Ready Mode
0	1	0	0	ACK Mode
1	0	0	0	ACK Mode
1	1	0	1	Ready Mode

The functional modes selected by HPCR [7:6] are as follows (assuming active high signal):

- **ACK Mode**—Acknowledge is active on strobes; HACK goes high from the leading edge of the strobe to indicate when the access can complete. After the Host samples the HACK active, it can complete the access by removing the strobe.The Host port then removes the HACK.
- **Ready Mode**—Ready active on strobes, goes low to insert waitstate during the access.If the Host port cannot complete the access, it deasserts the HACK/READY line. In this case, the Host has to extend the access by keeping the strobe asserted. When the Host samples the HACK asserted, it can then proceed and complete the access by deasserting the strobe.

While in Address Cycle Control (ACC) mode and the ACK or Ready acknowledge modes, the HACK is returned active for any address cycle.

Host Port Chip Selects

There are two chip-select signals associated with the Host port: $\overline{\text{HCMS}}$ and $\overline{\text{HCIOMS}}$. The Host Chip Memory Select ($\overline{\text{HCMS}}$) lets the Host select the DSP and directly access the DSP's internal/external memory space or boot memory space. The Host Chip I/O Memory Select ($\overline{\text{HCIOMS}}$) lets the Host select the DSP and directly access the DSP's internal I/O memory space.

Before starting a direct access, the Host configures Host port interface registers, specifying the width of external data bus (8- or 16-bit) and the target address page (in the IJPG register). The DSP generates the needed memory select signals during the access, based on the target address. The Host port interface combines the data from one, two, or three consecutive Host accesses (up to one 24-bit value) into a single DMA bus access to prefetch Host direct reads or to post direct writes. During assembly of larger words, the Host port interface asserts ACK for each byte access that does not start a read or complete a write. Otherwise, the Host port interface asserts ACK when it has completed the memory access successfully.

DSP Serial Ports (SPORTs)

The ADSP-2191M incorporates three complete synchronous serial ports (SPORT0, SPORT1, and SPORT2) for serial and multiprocessor communications. The SPORTs support the following features:

- Bidirectional operation—each SPORT has independent transmit and receive pins.
- Double-buffered transmit and receive ports—each port has a data register for transferring data words to and from memory and shift registers for shifting data in and out of the data registers.
- Clocking—each transmit and receive port can either use an external serial clock (40 MHz) or generate its own, in frequencies ranging from 19 Hz to 40 MHz.
- Word length—each SPORT supports serial data words from 3 to 16 bits in length transferred in Big Endian (MSB) or Little Endian (LSB) format.

- Framing—each transmit and receive port can run with or without frame sync signals for each data word. Frame sync signals can be generated internally or externally, active high or low, and with either of two pulsewidths and early or late frame sync.
- Companding in hardware—each SPORT can perform A-law or μ-law companding according to ITU recommendation G.711. Companding can be selected on the transmit and/or receive channel of the SPORT without additional latencies.
- DMA operations with single-cycle overhead—each SPORT can automatically receive and transmit multiple buffers of memory data, one data word each DSP cycle. Either the DSP's core or a Host processor can link or chain sequences of DMA transfers between a SPORT and memory. The chained DMA can be dynamically allocated and updated through the DMA descriptors (DMA transfer parameters) that set up the chain.
- Interrupts—each transmit and receive port generates an interrupt upon completing the transfer of a data word or after transferring an entire data buffer or buffers through DMA.
- Multichannel capability—each SPORT supports the H.100 standard.

Serial Peripheral Interface (SPI) Ports

The DSP has two SPI-compatible ports that enable the DSP to communicate with multiple SPI-compatible devices. These ports are multiplexed with SPORT2, so either SPORT2 or the SPI ports are active, depending on the state of the OPMODE pin during hardware reset.

The SPI interface uses three pins for transferring data: two data pins (Master Output-Slave Input, MOSIx, and Master Input-Slave Output, MISOx) and a clock pin (Serial Clock, SCKx). Two SPI chip select input pins ($\overline{\text{SPISSx}}$) let other SPI devices select the DSP, and fourteen SPI chip select output pins (SPIxSEL7–1) let the DSP select other SPI devices. The SPI select pins are reconfigured Programmable Flag pins. Using these pins, the SPI ports provide a full duplex, synchronous serial interface, which supports both master and slave modes and multimaster environments.

Each SPI port's baud rate and clock phase/polarities are programmable (see equation below for SPI clock rate calculation), and each has an integrated DMA controller, configurable to support both transmit and receive data streams. The SPI's DMA controller can only service unidirectional accesses at any given time.

$$SPI\ Clock\ Rate = \frac{HCLK}{2 \times SPIBAUD}$$

During transfers, the SPI ports simultaneously transmit and receive by serially shifting data in and out on their two serial data lines. The serial clock line synchronizes the shifting and sampling of data on the two serial data lines.

UART Port

The UART port provides a simplified UART interface to another peripheral or Host. It performs full duplex, asynchronous transfers of serial data. Options for the UART include support for 5–8 data bits; 1 or 2 stop bits; and none, even, or odd parity. The UART port supports two modes of operation:

- Programmed I/O

  The DSP's core sends or receives data by writing or reading I/O-mapped THR or RBR registers, respectively. The data is double-buffered on both transmit and receive.
- DMA (direct memory access)

  The DMA controller transfers both transmit and receive data. This reduces the number and frequency of interrupts required to transfer data to and from memory. The UART has two dedicated DMA channels. These DMA channels have lower priority than most DMA channels because of their relatively low service rates.

The UART's baud rate (see following equation for UART clock rate calculation), serial data format, error code generation and status, and interrupts are programmable:

- Supported bit rates range from 9.5 bits to 5M bits per second (80 MHz peripheral clock).
- Supported data formats are 7- to 12-bit frames.
- Transmit and receive status can be configured to generate maskable interrupts to the DSP's core.

The timers can be used to provide a hardware-assisted autobaud detection mechanism for the UART interface.

$$UART\ Clock\ Rate = \frac{HCLK}{16 \times D}$$

Where D is the programmable divisor = 1 to 65536.

Programmable Flag (PFx) Pins

The ADSP-2191M has 16 bidirectional, general-purpose I/O, Programmable Flag (PF15–0) pins. The PF7–0 pins are dedicated to general-purpose I/O. The PF15–8 pins serve either as general-purpose I/O pins (if the DSP is connected to an 8-bit external data bus) or serve as DATA15–8 lines (if the DSP is connected to a 16-bit external data bus). The Programmable Flag pins have special functions for clock multiplier selection and for SPI port operation. For more information, see Serial Peripheral

Interface (SPI) Ports on Page 9 and Clock Signals on Page 11.
Ten memory-mapped registers control operation of the Programmable Flag pins:

- Flag Direction register

  Specifies the direction of each individual PFx pin as input or output.

- Flag Control and Status registers

  Specify the value to drive on each individual PFx output pin. As input, software can predicate instruction execution on the value of individual PFx input pins captured in this register. One register sets bits, and one register clears bits.

- Flag Interrupt Mask registers

  Enable and disable each individual PFx pin to function as an interrupt to the DSP's core. One register sets bits to enable interrupt function, and one register clears bits to disable interrupt function. Input PFx pins function as hardware interrupts, and output PFx pins function as software interrupts—latching in the IMASK and IRPTL registers.

- Flag Interrupt Polarity register

  Specifies the polarity (active high or low) for interrupt sensitivity on each individual PFx pin.

- Flag Sensitivity registers

  Specify whether individual PFx 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. One register selects the type of sensitivity, and one register selects which edges are significant for edge-sensitivity.

Low Power Operation
The ADSP-2191M has four low power options that significantly reduce the power dissipation when the device operates under standby conditions. To enter any of these modes, the DSP executes an IDLE instruction. The ADSP-2191M uses configuration of the PDWN, STOPCK, and STOPALL bits in the PLLCTL register to select between the low power modes as the DSP executes the IDLE. Depending on the mode, an IDLE shuts off clocks to different parts of the DSP in the different modes. The low power modes are:

- Idle
- Power-Down Core
- Power-Down Core/Peripherals
- Power-Down All

Idle Mode
When the ADSP-2191M is in Idle mode, the DSP core stops executing instructions, retains the contents of the instruction pipeline, and waits for an interrupt. The core clock and peripheral clock continue running.

To enter Idle mode, the DSP can execute the IDLE instruction anywhere in code. To exit Idle mode, the DSP responds to an interrupt and (after two cycles of latency) resumes executing instructions with the instruction after the IDLE.

Power-Down Core Mode
When the ADSP-2191M is in Power-Down Core mode, the DSP core clock is off, but the DSP retains the contents of the pipeline and keeps the PLL running. The peripheral bus keeps running, letting the peripherals receive data.

To enter Power-Down Core mode, the DSP executes an IDLE instruction after performing the following tasks:

- Enter a power-down interrupt service routine
- Check for pending interrupts and I/O service routines
- Clear (= 0) the PDWN bit in the PLLCTL register
- Clear (= 0) the STOPALL bit in the PLLCTL register
- Set (= 1) the STOPCK bit in the PLLCTL register

To exit Power-Down Core mode, the DSP responds to an interrupt and (after two cycles of latency) resumes executing instructions with the instruction after the IDLE.

Power-Down Core/Peripherals Mode
When the ADSP-2191M is in Power-Down Core/Peripherals mode, the DSP core clock and peripheral bus clock are off, but the DSP keeps the PLL running. The DSP does not retain the contents of the instruction pipeline.The peripheral bus is stopped, so the peripherals cannot receive data.

To enter Power-Down Core/Peripherals mode, the DSP executes an IDLE instruction after performing the following tasks:

- Enter a power-down interrupt service routine
- Check for pending interrupts and I/O service routines
- Clear (= 0) the PDWN bit in the PLLCTL register
- Set (= 1) the STOPALL bit in the PLLCTL register

To exit Power-Down Core/Peripherals mode, the DSP responds to a wake-up event and (after five to six cycles of latency) resumes executing instructions with the instruction after the IDLE.

Power-Down All Mode
When the ADSP-2191M is in Power-Down All mode, the DSP core clock, the peripheral clock, and the PLL are all stopped. The DSP does not retain the contents of the instruction pipeline. The peripheral bus is stopped, so the peripherals cannot receive data.

To enter Power-Down All mode, the DSP executes an IDLE instruction after performing the following tasks:

- Enter a power-down interrupt service routine
- Check for pending interrupts and I/O service routines
- Set (= 1) the PDWN bit in the PLLCTL register

To exit Power-Down Core/Peripherals mode, the DSP responds to an interrupt and (after 500 cycles to restabilize the PLL) resumes executing instructions with the instruction after the IDLE.

Clock Signals

The ADSP-2191M can be clocked by a crystal oscillator or a buffered, shaped clock derived from an external clock oscillator. If a crystal oscillator is used, the crystal should be connected across the CLKIN and XTAL pins, with two capacitors and a 1 MΩ shunt resistor connected as shown in Figure 3. Capacitor values are dependent on crystal type and should be specified by the crystal manufacturer. A parallel-resonant, fundamental frequency, microprocessor-grade crystal should be used for this configuration.

If a buffered, shaped clock is used, this external clock connects to the DSP's CLKIN pin. CLKIN input cannot be halted, changed, or operated below the specified frequency during normal operation. When an external clock is used, the XTAL input must be left unconnected.

The DSP provides a user-programmable 1× to 32× multiplication of the input clock, including some fractional values, to support 128 external to internal (DSP core) clock ratios. The MSEL6–0, BYPASS, and DF pins decide the PLL multiplication factor at reset. At runtime, the multiplication factor can be controlled in software. The combination of pullup and pull-down resistors in Figure 3 sets up a core clock ratio of 6:1, which produces a 150 MHz core clock from the 25 MHz input. For other clock multiplier settings, see the *ADSP-219x/ADSP-2191 DSP Hardware Reference*.

The peripheral clock is supplied to the CLKOUT pin.

All on-chip peripherals for the ADSP-2191M operate at the rate set by the peripheral clock. The peripheral clock is either equal to the core clock rate or one-half the DSP core clock rate. This selection is controlled by the IOSEL bit in the PLLCTL register. The maximum core clock is 160 MHz and the maximum peripheral clock is 80 MHz—the combination of the input clock and core/peripheral clock ratios may not exceed these limits.

Reset

The $\overline{\text{RESET}}$ signal initiates a master reset of the ADSP-2191M. The $\overline{\text{RESET}}$ signal must be asserted during the powerup sequence to assure proper initialization. $\overline{\text{RESET}}$ during initial powerup must be held long enough to allow the internal clock to stabilize.

The powerup sequence is defined as the total time required for the crystal oscillator circuit to stabilize after a valid V_{DD} is applied to the processor, and for the internal phase-locked loop (PLL) to lock onto the specific crystal frequency. A minimum of 100 µs ensures that the PLL has locked, but does not include the crystal oscillator start-up time. During this powerup sequence the $\overline{\text{RESET}}$ signal should be held low. On any subsequent resets, the $\overline{\text{RESET}}$ signal must meet the minimum pulsewidth specification, t_{WRST}.

The $\overline{\text{RESET}}$ input contains some hysteresis. If using an RC circuit to generate your $\overline{\text{RESET}}$ signal, the circuit should use an external Schmidt trigger.

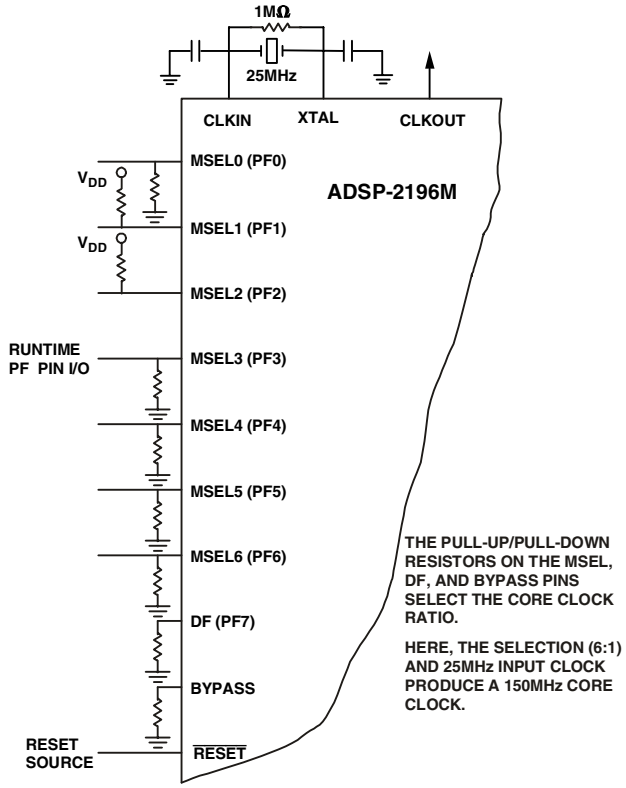

Figure 3. External Crystal Connections

The master reset sets all internal stack pointers to the empty stack condition, masks all interrupts, and resets all registers to their default values (where applicable). When $\overline{\text{RESET}}$ is released, if there is no pending bus request and the chip is configured for booting, the boot-loading sequence is performed. Program control jumps to the location of the on-chip boot ROM (0xFF 0000).

Power Supplies

The ADSP-2191M has separate power supply connections for the internal (V_{DDINT}) and external (V_{DDEXT}) power supplies. The internal supply must meet the 2.5 V requirement. The external supply must be connected to a 3.3 V supply. All external supply pins must be connected to the same supply.

Power-Up Sequence

Power up together the two supplies V_{DDEXT} and V_{DDINT}. If they cannot be powered up together, power up the internal (core) supply first (powering up the core supply first reduces the risk of latchup events.

Booting Modes

The ADSP-2191M has five mechanisms (listed in Table 6) for automatically loading internal program memory after reset. Two no-boot modes are also supported.

In addition to the software and hardware development tools available from Analog Devices, third parties provide a wide range of tools supporting the ADSP-219x processor family. Hardware tools include ADSP-219x PC plug-in cards. Third party software tools include DSP libraries, real-time operating systems, and block diagram design tools.

Designing an Emulator-Compatible DSP Board (Target)

The White Mountain DSP (Product Line of Analog Devices, Inc.) family of emulators are tools that every DSP developer needs to test and debug hardware and software systems. Analog Devices has supplied an IEEE 1149.1 JTAG Test Access Port (TAP) on each JTAG DSP. The emulator uses the TAP to access the internal features of the DSP, allowing the developer to load code, set breakpoints, observe variables, observe memory, and examine registers. The DSP must be halted to send data and commands, but once an operation has been completed by the emulator, the DSP system is set running at full speed with no impact on system timing.

To use these emulators, the target's design must include the interface between an Analog Devices JTAG DSP and the emulation header on a custom DSP target board.

Target Board Header

The emulator interface to an Analog Devices JTAG DSP is a 14-pin header, as shown in Figure 4. The customer must supply this header on the target board in order to communicate with the emulator. The interface consists of a standard dual row 0.025" square post header, set on 0.1" × 0.1" spacing, with a minimum post length of 0.235". Pin 3 is the key position used to prevent the pod from being inserted backwards. This pin must be clipped on the target board.

Also, the clearance (length, width, and height) around the header must be considered. Leave a clearance of at least 0.15" and 0.10" around the length and width of the header, and reserve a height clearance to attach and detach the pod connector.

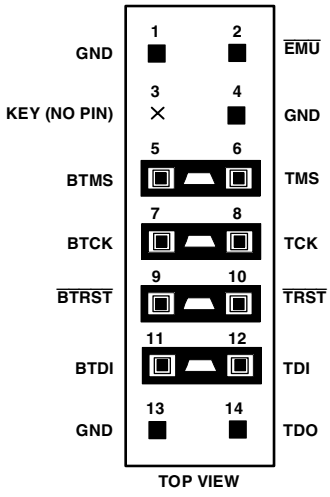

Figure 4. JTAG Target Board Connector for JTAG Equipped Analog Devices DSP (Jumpers in Place)

As can be seen in Figure 4, there are two sets of signals on the header. There are the standard JTAG signals TMS, TCK, TDI, TDO, $\overline{\text{TRST}}$, and $\overline{\text{EMU}}$ used for emulation purposes (via an emulator). There are also secondary JTAG signals BTMS, BTCK, BTDI, and $\overline{\text{BTRST}}$ that are optionally used for board-level (boundary scan) testing.

When the emulator is not connected to this header, place jumpers across BTMS, BTCK, $\overline{\text{BTRST}}$, and BTDI as shown in Figure 5. This holds the JTAG signals in the correct state to allow the DSP to run free. Remove all the jumpers when connecting the emulator to the JTAG header.

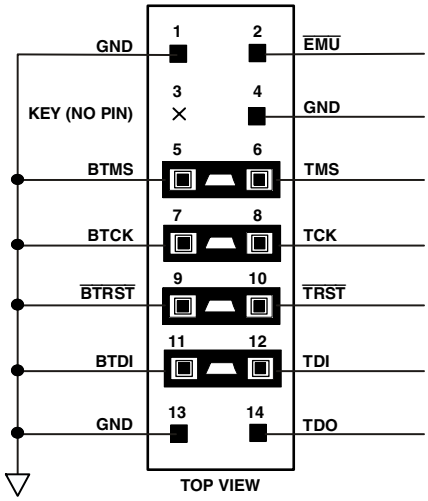

Figure 5. JTAG Target Board Connector with No Local Boundary Scan

JTAG Emulator Pod Connector

Figure 6 details the dimensions of the JTAG pod connector at the 14-pin target end. Figure 7 displays the keep-out area for a target board header. The keep-out area allows the pod connector to properly seat onto the target board header. This board area should contain no components (chips, resistors, capacitors, etc.). The dimensions are referenced to the center of the 0.25" square post pin.

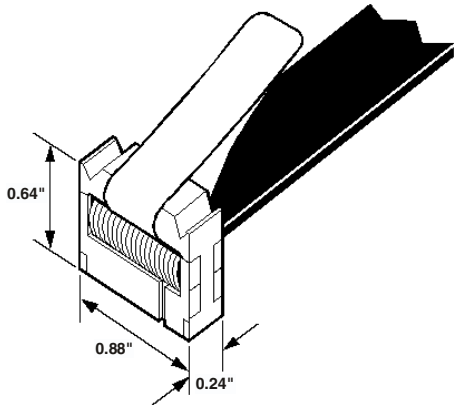

Figure 6. JTAG Pod Connector Dimensions

External Port Bus Request and Grant Cycle Timing

Table 14 and Figure 13 describe external port bus request and bus grant operations.

Table 14. External Port Bus Request and Grant Cycle Timing

Parameter[1, 2]		Min	Max	Unit
Switching Characteristics				
t_{SD}	CLKOUT High to $\overline{xMS}$, Address, and $\overline{RD}/\overline{WR}$ Disable		$0.5t_{HCLK}+1$	ns
t_{SE}	CLKOUT Low to $\overline{xMS}$, Address, and $\overline{RD}/\overline{WR}$ Enable	0	4	ns
t_{DBG}	CLKOUT High to $\overline{BG}$ Asserted Setup	0	4	ns
t_{EBG}	CLKOUT High to $\overline{BG}$ Deasserted Hold Time	0	4	ns
t_{DBH}	CLKOUT High to $\overline{BGH}$ Asserted Setup	0	4	ns
t_{EBH}	CLKOUT High to $\overline{BGH}$ Deasserted Hold Time	0	4	ns
Timing Requirements				
t_{BS}	$\overline{BR}$ Asserted to CLKOUT High Setup	4.6		ns
t_{BH}	CLKOUT High to $\overline{BR}$ Deasserted Hold Time	0		ns

[1] t_{HCLK} is the peripheral clock period.
[2] These are timing parameters that are based on worst-case operating conditions.

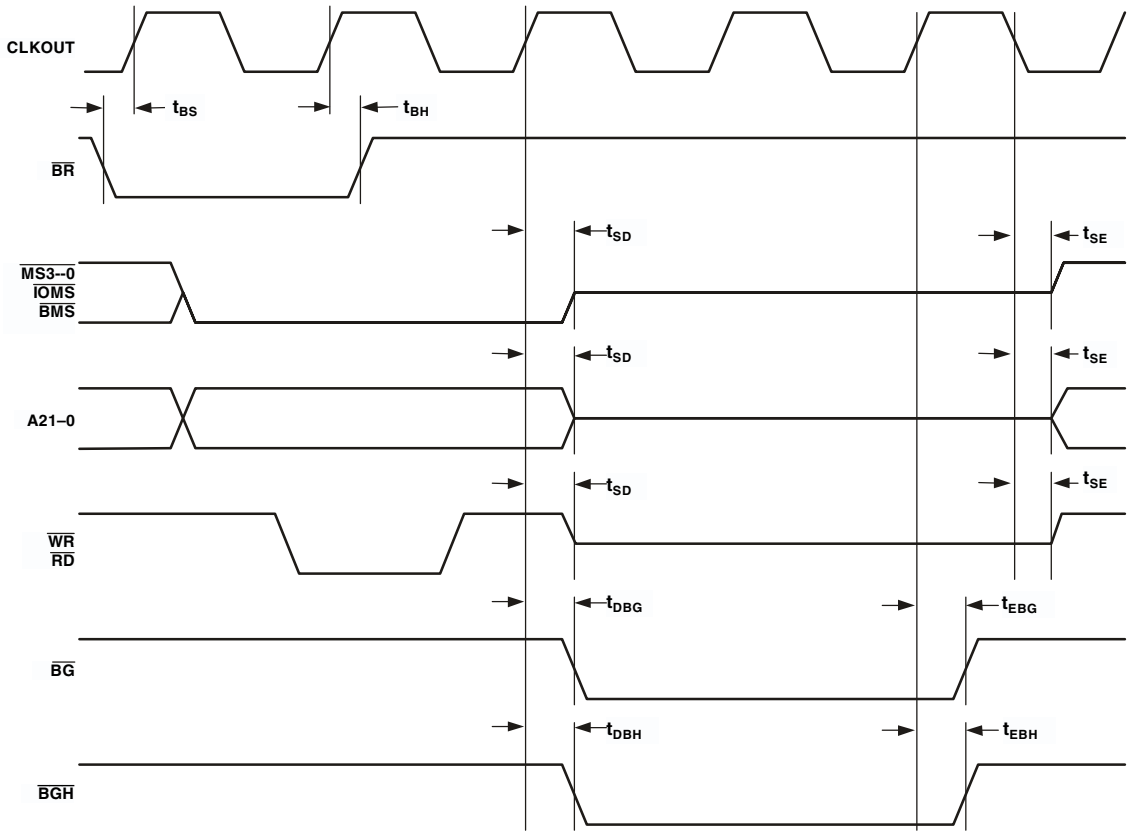

Figure 13. External Port Bus Request and Grant Cycle Timing

Host Port ALE Mode Write Cycle Timing

Table 15 and Figure 14 describe Host port write operations in Address Latch Enable (ALE) mode. For more information on ACK, Ready, ALE, and ACC mode selection, see the Host port modes description on Page 8.

Table 15. Host Port ALE Mode Write Cycle Timing

Parameter		Min	Max	Unit
Switching Characteristics				
t_{WHKS1}	$\overline{HWR}$ Asserted to HACK Asserted (Setup, ACK Mode) First Byte	10	$5t_{HCLK}+t_{NH}$[1]	ns
t_{WHKS2}	$\overline{HWR}$ Asserted to HACK Asserted (Setup, ACK Mode)[2]		10	ns
t_{WHKH}	$\overline{HWR}$ Deasserted to HACK Deasserted (Hold, ACK Mode)		10	ns
t_{WHS}	$\overline{HWR}$ Asserted to HACK Asserted (Setup, Ready Mode)		10	ns
t_{WHH}	$\overline{HWR}$ Asserted to HACK Deasserted (Hold, Ready Mode) First Byte	0	$5t_{HCLK}+t_{NH}$[1]	ns
Timing Requirements				
t_{CSAL}	$\overline{HCMS}$ or $\overline{HCIOMS}$ Asserted to HALE Asserted	0		ns
t_{ALPW}	HALE Asserted Pulsewidth	4		ns
t_{ALCSW}	HALE Deasserted to $\overline{HCMS}$ or $\overline{HCIOMS}$ Deasserted	1		ns
t_{WCSW}	$\overline{HWR}$ Deasserted to $\overline{HCMS}$ or $\overline{HCIOMS}$ Deasserted	0		ns
t_{ALW}	HALE Deasserted to $\overline{HWR}$ Asserted	1		ns
t_{WCS}	$\overline{HWR}$ Deasserted (After Last Byte) to $\overline{HCMS}$ or $\overline{HCIOMS}$ Deasserted (Ready for Next Write)	0		ns
t_{HKWD}	HACK Asserted to $\overline{HWR}$ Deasserted (Hold, ACK Mode)	1.5		ns
t_{AALS}	Address Valid to HALE Deasserted (Setup)	2		ns
t_{ALAH}	HALE Deasserted to Address Invalid (Hold)	4		ns
t_{DWS}	Data Valid to $\overline{HWR}$ Deasserted (Setup)	4		ns
t_{WDH}	$\overline{HWR}$ Deasserted to Data Invalid (Hold)	1		ns

[1] t_{NH} are peripheral bus latencies ($n \times t_{HCLK}$); these are internal DSP latencies related to the number of peripheral DMAs attempting to access DSP memory at the same time.

[2] Measurement is for the second, third, or fourth byte of a host write transaction. The quantity of bytes to complete a host write transaction is dependent on the data bus size (8 or 16 bits) and the data type (16 or 24 bits).

Host Port ALE Mode Read Cycle Timing

Table 17 and Figure 16 describe Host port read operations in Address Latch Enable (ALE) mode. For more information on ACK, Ready, ALE, and ACC mode selection, see the Host port modes description on Page 8.

Table 17. Host Port ALE Mode Read Cycle Timing

Parameter		Min	Max	Unit
Switching Characteristics				
t_{RHKS1}	$\overline{HRD}$ Asserted to HACK Asserted (ACK Mode) First Byte	$12t_{HCLK}$	$15t_{HCLK}+t_{NH}$[1]	ns
t_{RHKS2}	$\overline{HRD}$ Asserted to HACK Asserted (Setup, ACK Mode)[2]		12	ns
t_{RHKH}	$\overline{HRD}$ Deasserted to HACK Deasserted (Hold, ACK Mode)		10	ns
t_{RHS}	$\overline{HRD}$ Asserted to HACK Asserted (Setup, Ready Mode)		10	ns
t_{RHH}	$\overline{HRD}$ Asserted to HACK Deasserted (Hold, Ready Mode) First Byte	$12t_{HCLK}$	$15t_{HCLK}+t_{NH}$[1]	ns
t_{RDH}	$\overline{HRD}$ Deasserted to Data Invalid (Hold)	1		ns
t_{RDD}	$\overline{HRD}$ Deasserted to Data Disable		10	ns
Timing Requirements				
t_{CSAL}	$\overline{HCMS}$ or $\overline{HCIOMS}$ Asserted to HALE Asserted (Delay)	0		ns
t_{ALCS}	HALE Deasserted to Optional $\overline{HCMS}$ or $\overline{HCIOMS}$ Deasserted	1		ns
t_{RCSW}	$\overline{HRD}$ Deasserted to $\overline{HCMS}$ or $\overline{HCIOMS}$ Deasserted	0		ns
t_{ALR}	HALE Deasserted to $\overline{HRD}$ Asserted	5		ns
t_{RCS}	$\overline{HRD}$ Deasserted (After Last Byte) to $\overline{HCMS}$ or $\overline{HCIOMS}$ Deasserted (Ready for Next Read)	0		ns
t_{ALPW}	HALE Asserted Pulsewidth	4		ns
t_{HKRD}	HACK Asserted to $\overline{HRD}$ Deasserted (Hold, ACK Mode)	1.5		ns
t_{AALS}	Address Valid to HALE Deasserted (Setup)	2		ns
t_{ALAH}	HALE Deasserted to Address Invalid (Hold)	4		ns

[1] t_{NH} are peripheral bus latencies ($n \times t_{HCLK}$); these are internal DSP latencies related to the number of peripherals attempting to access DSP memory at the same time.

[2] Measurement is for the second, third, or fourth byte of a host read transaction. The quantity of bytes to complete a host read transaction is dependent on the data bus size (8 or 16 bits) and the data type (16 or 24 bits).

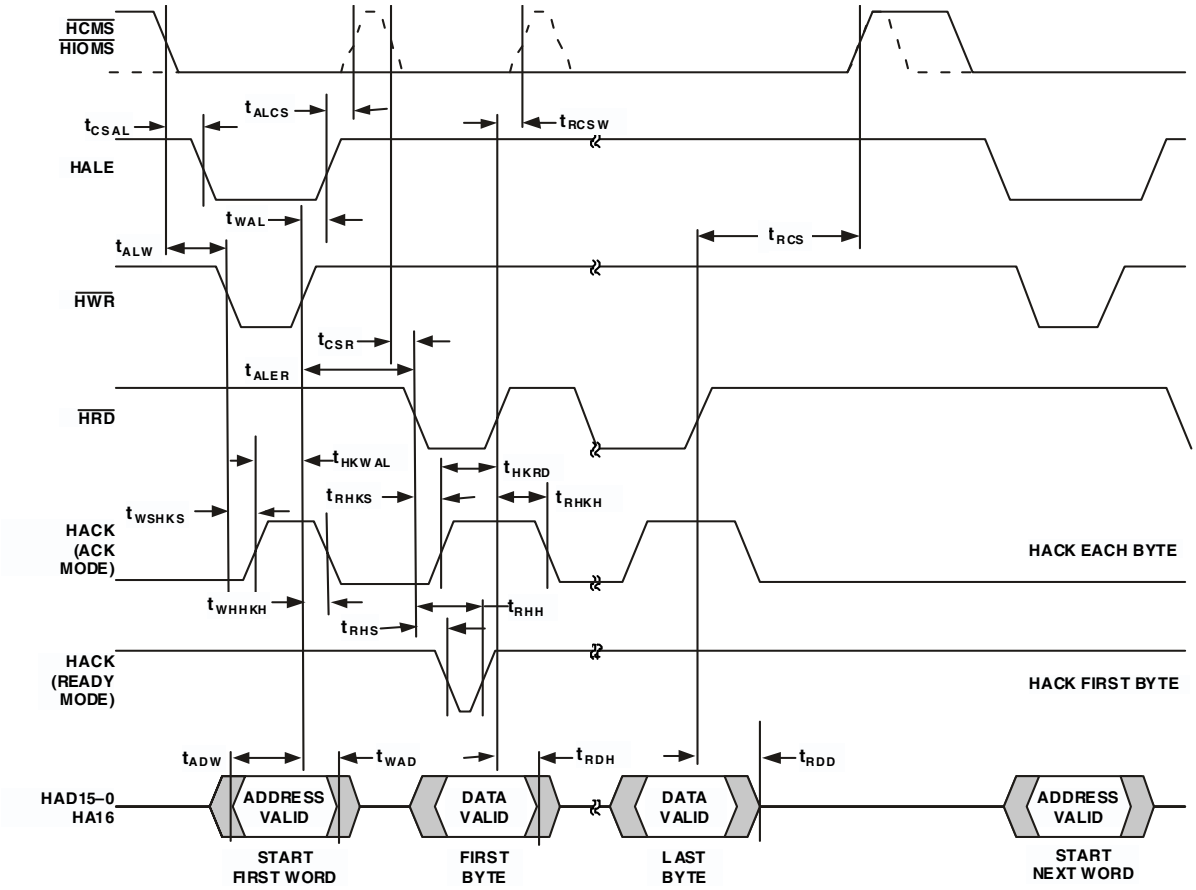

Figure 17. Host Port ACC Mode Read Cycle Timing

Serial Ports

Table 19 and Figure 18 describe SPORT transmit and receive operations, while Figure 19 and Figure 20 describe SPORT Frame Sync operations.

Table 19. Serial Ports[1, 2]

Parameter		Min	Max	Unit
External Clock Timing Requirements				
t_{SFSE}	TFS/RFS Setup Before TCLK/RCLK[3]	4		ns
t_{HFSE}	TFS/RFS Hold After TCLK/RCLK[3]	4		ns
t_{SDRE}	Receive Data Setup Before RCLK[3]	1.5		ns
t_{HDRE}	Receive Data Hold After RCLK[3]	4		ns
t_{SCLKW}	TCLK/RCLK Width	$0.5t_{HCLK}-1$		ns
t_{SCLK}	TCLK/RCLK Period	$2t_{HCLK}$		ns
Internal Clock Timing Requirements				
t_{SFSI}	TFS Setup Before TCLK[4]; RFS Setup Before RCLK[3]	4		ns
t_{HFSI}	TFS/RFS Hold After TCLK/RCLK[3]	3		ns
t_{SDRI}	Receive Data Setup Before RCLK[3]	2		ns
t_{HDRI}	Receive Data Hold After RCLK[3]	5		ns
External or Internal Clock Switching Characteristics				
t_{DFSE}	TFS/RFS Delay After TCLK/RCLK (Internally Generated FS)[4]		14	ns
t_{HOFSE}	TFS/RFS Hold After TCLK/RCLK (Internally Generated FS)[4]	3		ns
External Clock Switching Characteristics				
t_{DDTE}	Transmit Data Delay After TCLK[4]		13.4	ns
t_{HDTE}	Transmit Data Hold After TCLK[4]	4		ns
Internal Clock Switching Characteristics				
t_{DDTI}	Transmit Data Delay After TCLK[4]		13.4	ns
t_{HDTI}	Transmit Data Hold After TCLK[4]	4		ns
t_{SCLKIW}	TCLK/RCLK Width	$0.5t_{HCLK}-3.5$	$0.5t_{HCLK}+2.5$	ns
Enable and Three-State[5] Switching Characteristics				
t_{DTENE}	Data Enable from External TCLK[4]	0	12.1	ns
t_{DDTTE}	Data Disable from External TCLK[4]		13	ns
t_{DTENI}	Data Enable from Internal TCLK[4]	0	13	ns
t_{DDTTI}	Data Disable from External TCLK[4]		12	ns
External Late Frame Sync Switching Characteristics				
$t_{DDTLFSE}$	Data Delay from Late External TFS with MCE=1, MFD=0[6, 7]		10.5	ns
$t_{DTENLFSE}$	Data Enable from Late FS or MCE=1, MFD=0[6, 7]	3.5		ns

[1]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) SCLK width.
[2]Word selected timing for I^2S mode is the same as TFS/RFS timing (normal framing only).
[3]Referenced to sample edge.
[4]Referenced to drive edge.
[5]Only applies to SPORT0/1.
[6]MCE=1, TFS enable, and TFS valid follow $t_{DDTENFS}$ and $t_{DDTLFSE}$.
[7]If external RFSD/TFS setup to RCLK/TCLK>$0.5t_{LSCK}$, $t_{DDTLSCK}$ and $t_{DTENLSCK}$ apply; otherwise $t_{DDTLFSE}$ and $t_{DTENLFS}$ apply.

Serial Peripheral Interface (SPI) Port—Master Timing

Table 20 and Figure 21 describe SPI port master operations.

Table 20. Serial Peripheral Interface (SPI) Port—Master Timing

Parameter		Min	Max	Unit
Switching Characteristics				
t_{SDSCIM}	$\overline{SPIxSEL}$ Low to First SCLK edge (x=0 or 1)	$2t_{HCLK}-3$		ns
t_{SPICHM}	Serial Clock High Period	$2t_{HCLK}-3$		ns
t_{SPICLM}	Serial Clock Low Period	$2t_{HCLK}-3$		ns
t_{SPICLK}	Serial Clock Period	$4t_{HCLK}-1$		ns
t_{HDSM}	Last SCLK Edge to $\overline{SPIxSEL}$ High (x=0 or 1)	$2t_{HCLK}-3$		ns
t_{SPITDM}	Sequential Transfer Delay	$2t_{HCLK}-2$		ns
t_{DDSPID}	SCLK Edge to Data Output Valid (Data Out Delay)	0	6	ns
t_{HDSPID}	SCLK Edge to Data Output Invalid (Data Out Hold)	0	5	ns
Timing Requirements				
t_{SSPID}	Data Input Valid to SCLK Edge (Data Input Setup)	8		ns
t_{HSPID}	SCLK Sampling Edge to Data Input Invalid (Data In Hold)	1		ns

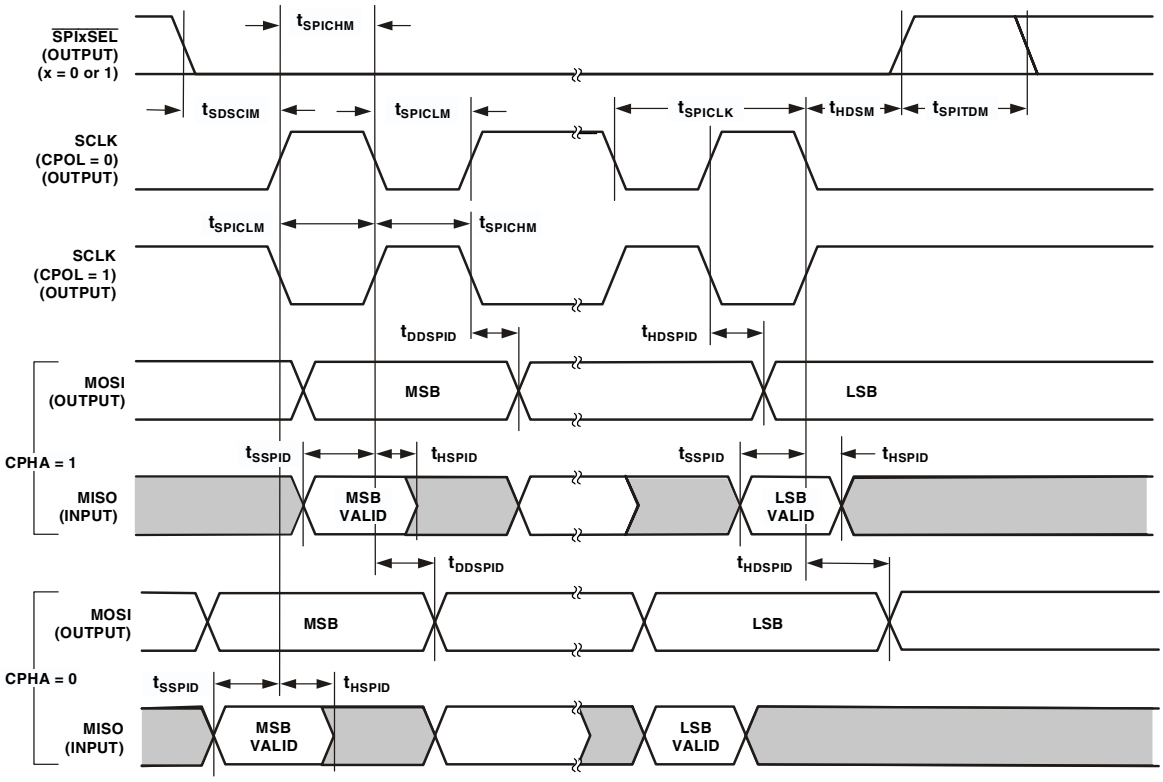

Figure 21. Serial Peripheral Interface (SPI) Port—Master Timing

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

Figure 23 describes UART port receive and transmit operations. The maximum baud rate is HCLK/16. As shown in Figure 23 there is some latency between the generation internal UART interrupts and the external data operations. These latencies are negligible at the data transmission rates for the UART.

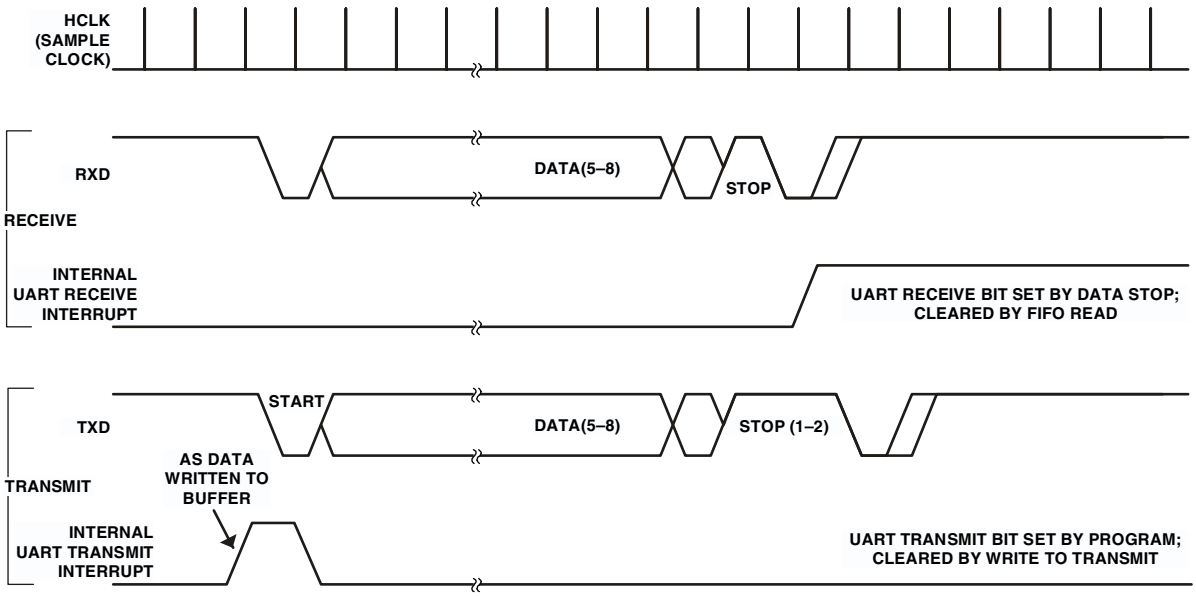

Figure 23. UART Port—Receive and Transmit Timing

JTAG Test And Emulation Port Timing

Table 22 and Figure 24 describe JTAG port operations.

Table 22. JTAG Port Timing

Parameter		Min	Max	Unit
Switching Characteristics				
t_{DTDO}	TDO Delay from TCK Low		8	ns
t_{DSYS}	System Outputs Delay After TCK Low[1]	0	22	ns
Timing Requirements				
t_{TCK}	TCK Period	20		ns
t_{STAP}	TDI, TMS Setup Before TCK High		4	ns
t_{HTAP}	TDI, TMS Hold After TCK High		4	ns
t_{SSYS}	System Inputs Setup Before TCK Low[2]		4	ns
t_{HSYS}	System Inputs Hold After TCK Low[2]		5	ns
t_{TRSTW}	$\overline{TRST}$ Pulsewidth[3]	$4t_{TCK}$		ns

[1]System Outputs = DATA15–0, ADDR21–0, $\overline{MS3–0}$, $\overline{RD}$, $\overline{WR}$, ACK, CLKOUT, $\overline{BG}$, PF7–0, TIMEXP, DT0, DT1, TCLK0, TCLK1, RCLK0, RCLK1, TFS0, TFS1, RFS0, RFS1, $\overline{BMS}$.
[2]System Inputs = DATA15–0, ADDR21–0, $\overline{RD}$, $\overline{WR}$, ACK, $\overline{BR}$, $\overline{BG}$, PF7–0, DR0, DR1, TCLK0, TCLK1, RCLK0, RCLK1, TFS0, TFS1, RFS0, RFS1, CLKIN, $\overline{RESET}$.
[3]50 MHz max.

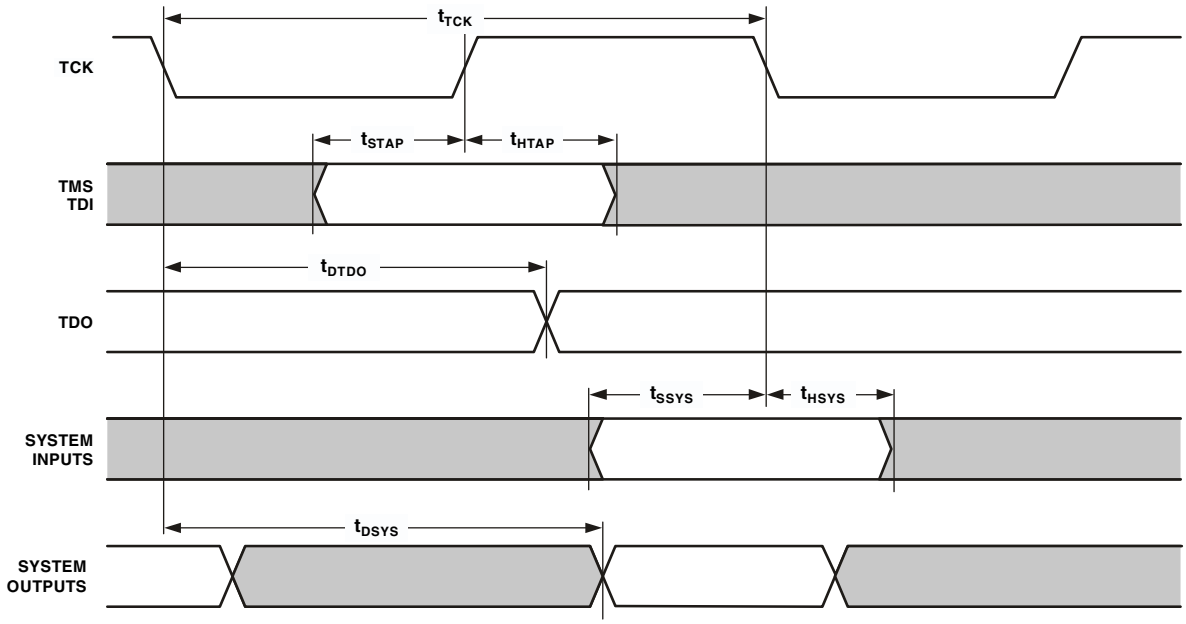

Figure 24. JTAG Port Timing

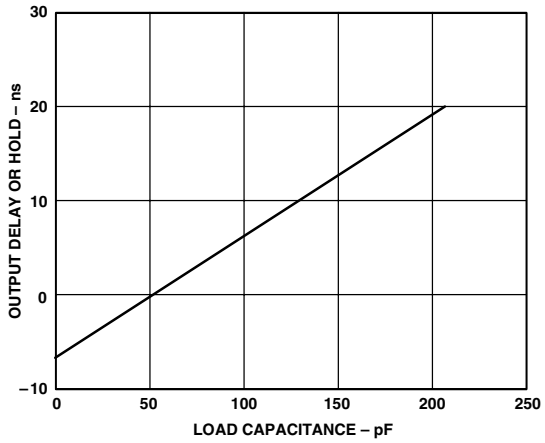

Figure 30. Typical Output Delay or Hold vs. Load Capacitance (at Maximum Case Temperature)

To ensure that the T_{AMB} data sheet specification is not exceeded, a heatsink and/or an air flow source may be used. A heatsink should be attached to the ground plane (as close as possible to the thermal pathways) with a thermal adhesive.

$$T_{AMB} = T_{CASE} - PD \times \theta_{CA}$$

Where:

- T_{AMB} = Ambient temperature (measured near top surface of package)
- PD = Power dissipation in W (this value depends upon the specific application; a method for calculating PD is shown under Power Dissipation).
- θ_{CA} = Value from Table 24.
- For the LQFP package: θ_{JC} = 0.96°C/W
  For the mini-BGA package: θ_{JC} = 8.4°C/W

Table 24. θ_{CA} Values

Airflow (Linear Ft./Min.)	0	100	200	400	600
Airflow (Meters/Second)	0	0.5	1	2	3
LQFP: θ_{CA} (°C/W)	44.3	41.4	38.5	35.3	32.1
Mini-BGA: θ_{CA} (°C/W)	26	24	22	20.9	19.8

144-Lead Mini-BGA Pinout

Table 27 lists the mini-BGA pinout by signal name. Table 28 lists the mini-BGA pinout by ball number.

Table 27. 144-Lead Mini-BGA Pins (Alphabetically by Signal)

Signal	Ball No.	Signal	Ball No.	Signal	Ball No.	Signal	Ball No.	Signal	Ball No.
A0	J11	BYPASS	M11	GND	F7	HALE	J1	TCLK0	J6
A1	H9	CLKIN	A5	GND	F8	$\overline{\text{HCIOMS}}$	J3	TCLK1	M9
A2	H10	CLKOUT	C6	GND	F9	$\overline{\text{HCMS}}$	H1	TCLK2	K5
A3	G12	D0	D7	GND	G4	$\overline{\text{HRD}}$	J2	TDI	K12
A4	H11	D1	A7	GND	G5	$\overline{\text{HWR}}$	K2	TDO	L11
A5	G10	D2	C7	GND	G6	$\overline{\text{IOMS}}$	E8	TFS0	M8
A6	F12	D3	A6	GND	H5	$\overline{\text{MS0}}$	D9	TFS1	J8
A7	G11	D4	B7	GND	L6	$\overline{\text{MS1}}$	A9	TFS2	M5
A8	F10	D5	A4	GND	M1	$\overline{\text{MS2}}$	C9	TMR0	K4
A9	F11	D6	C5	GND	M12	$\overline{\text{MS3}}$	D8	TMR1	L4
A10	E12	D7	B5	HACK	H3	OPMODE	H12	TMR2	J4
A11	E11	D8	D5	HACK_P	G1	PF0	K1	TMS	K10
A12	E10	D9	A3	HAD0	C1	PF1	L1	$\overline{\text{TRST}}$	J12
A13	E9	D10	C4	HAD1	B3	PF2	M2	TXD	M7
A14	D11	D11	B4	HAD2	C2	PF3	L2	V_{DDEXT}	E5
A15	D10	D12	C3	HAD3	D1	PF4	M3	V_{DDEXT}	E6
A16	D12	D13	A2	HAD4	D4	PF5	L3	V_{DDEXT}	F5
A17	C11	D14	B1	HAD5	D3	PF6	K3	V_{DDEXT}	F6
A18	C12	D15	B2	HAD6	D2	PF7	M4	V_{DDEXT}	G7
A19	B12	DR0	L7	HAD7	E1	RCLK0	K7	V_{DDEXT}	G8
A20	B11	DR1	K9	HAD8	E4	RCLK1	J9	V_{DDEXT}	H7
A21	A11	DR2	L5	HAD9	E2	RCLK2	J5	V_{DDEXT}	H8
ACK	A8	DT0	H6	HAD10	F1	$\overline{\text{RD}}$	B8	V_{DDINT}	D6
$\overline{\text{BG}}$	C10	DT1	L8	HAD11	E3	$\overline{\text{RESET}}$	L12	V_{DDINT}	F4
$\overline{\text{BGH}}$	B10	DT2	H4	HAD12	F2	RFS0	K8	V_{DDINT}	G9
BMODE0	L10	$\overline{\text{EMU}}$	J10	HAD13	G2	RFS1	M10	V_{DDINT}	J7
BMODE1	L9	GND	A1	HAD14	F3	RFS2	M6	$\overline{\text{WR}}$	C8
$\overline{\text{BMS}}$	A10	GND	A12	HAD15	G3	RXD	K6	XTAL	B6
$\overline{\text{BR}}$	B9	GND	E7	HA16	H2	TCK	K11		