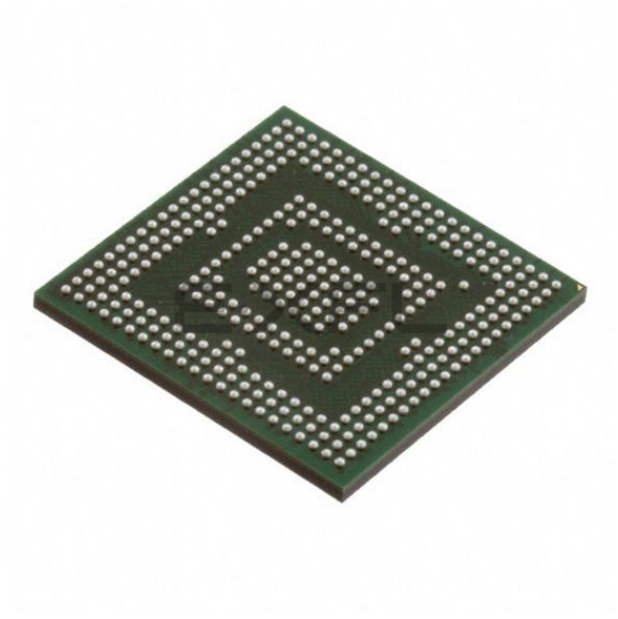


**Welcome to E-XFL.COM**

## 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 | 400MHz |
| Non-Volatile Memory | ROM (64kB) |
| On-Chip RAM | 552kB |
| 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/adbf606wcbcz402 |

### *Event Handling*

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

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

#### Core Event Controller (CEC)

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

#### System Event Controller (SEC)

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

### *Trigger Routing Unit (TRU)*

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

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

### *Pin Interrupts*

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

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

### *General-Purpose I/O (GPIO)*

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

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

### *Pin Multiplexing*

The processor supports a flexible multiplexing scheme that multiplexes the GPIO pins with various peripherals. A maximum of 4 peripherals plus GPIO functionality is shared by each GPIO pin. All GPIO pins have a bypass path feature – that is, when the output enable and the input enable of a GPIO pin are both active, the data signal before the pad driver is looped back to the receive path for the same GPIO pin. For more information, see GP I/O Multiplexing for 349-Ball CSP_BGA on Page 33.

## MEMORY ARCHITECTURE

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

### *Memory Protection*

The Blackfin cores feature a memory protection concept, which grants data and/or instruction accesses from enabled memory regions only. A supervisor mode vs. user mode programming model supports dynamically varying access rights. Increased flexibility in memory page size options supports a simple method of static memory partitioning.

### *System Protection*

All system resources and L2 memory banks can be controlled by either the processor cores, memory-to-memory DMA, or the system debug unit (SDU). A system protection unit (SPU) enables write accesses to specific resources that are locked to any of four masters: Core 0, Core 1, Memory DMA, and the System Debug Unit. System protection is enabled in greater granularity for some modules (L2, SEC and GPIO controllers) through a *global lock* concept.

### *Watchpoint Protection*

The primary purpose of watchpoints and hardware breakpoints is to serve emulator needs. When enabled, they signal an emulator event whenever user-defined system resources are accessed or a core executes from user-defined addresses. Watchpoint events can be configured such that they signal the events to the other Blackfin core or to the fault management unit.

### *Dual Watchdog*

The two on-chip watchdog timers each may supervise one Blackfin core.

### *Bandwidth Monitor*

All DMA channels that operate in memory-to-memory mode (Memory DMA, PVP Memory Pipe DMA, PIXC DMA) are equipped with a bandwidth monitor mechanism. They can signal a system event or fault when transactions tend to starve because system buses are fully loaded with higher-priority traffic.

### *Signal Watchdogs*

The eight general-purpose timers feature two new modes to monitor off-chip signals. The Watchdog Period mode monitors whether external signals toggle with a period within an expected range. The Watchdog Width mode monitors whether the pulse widths of external signals are in an expected range. Both modes help to detect incorrect undesired toggling (or lack thereof) of system-level signals.

### *Up/Down Count Mismatch Detection*

The up/down counter can monitor external signal pairs, such as request/grant strobes. If the edge count mismatch exceeds the expected range, the up/down counter can flag this to the processor or to the fault management unit.

### *Fault Management*

The fault management unit is part of the system event controller (SEC). Any system event, whether a dual-bit uncorrectable ECC error, or any peripheral status interrupt, can be defined as being a "fault". Additionally, the system events can be defined as an interrupt to the cores. If defined as such, the SEC forwards the event to the fault management unit which may automatically reset the entire device for reboot, or simply toggle the SYS_FAULT output pins to signal off-chip hardware. Optionally, the fault management unit can delay the action taken via a keyed sequence, to provide a final chance for the Blackfin cores to resolve the crisis and to prevent the fault action from being taken.

## ADDITIONAL PROCESSOR PERIPHERALS

The processor contains a rich set of peripherals connected to the core via several high-bandwidth buses, providing flexibility in system configuration as well as excellent overall system performance (see the block diagram on Page 1). The processors contain high-speed serial and parallel ports, an interrupt controller for flexible management of interrupts from the on-chip peripherals or external sources, and power management control functions to tailor the performance and power characteristics of the processor and system to many application scenarios.

The following sections describe additional peripherals that were not described in the previous sections.

### *Timers*

The processor includes several timers which are described in the following sections.

#### General-Purpose Timers

There is one GP timer unit and it provides eight general-purpose programmable timers. Each timer has an external pin that can be configured either as a pulse width modulator (PWM) or timer output, as an input to clock the timer, or as a mechanism for measuring pulse widths and periods of external events. These timers can be synchronized to an external clock input on the TMRx pins, an external clock TMRCLK input pin, or to the internal SCLK0.

The timer units can be used in conjunction with the UARTs and the CAN controller to measure the width of the pulses in the data stream to provide a software auto-baud detect function for the respective serial channels.

The timers can generate interrupts to the processor core, providing periodic events for synchronization to either the system clock or to external signals. Timer events can also trigger other peripherals via the TRU (for instance, to signal a fault).

#### Core Timers

Each processor core also has its own dedicated timer. This extra timer is clocked by the internal processor clock and is typically used as a system tick clock for generating periodic operating system interrupts.

#### Watchdog Timers

Each core includes a 32-bit timer, which may be used to implement a software watchdog function. A software watchdog can improve system availability by forcing the processor to a known state, via generation of a hardware reset, nonmaskable interrupt (NMI), or general-purpose interrupt, if the timer expires before

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 6. Detailed Signal Descriptions (Continued)**

| Signal Name | Direction | Description |
|---|---|---|
| SPT_BD1 | I/O | **Channel B Data 1** Secondary bidirectional data I/O. This signal can be configured as an output to transmit serial data, or as an input to receive serial data. |
| SPT_BFS | I/O | **Channel B Frame Sync** The frame sync pulse initiates shifting of serial data. This signal is either generated internally or externally. |
| SPT_BTDV | Output | **Channel B Transmit Data Valid** This signal is optional and only active when SPORT is configured in multi-channel transmit mode. It is asserted during enabled slots. |
| SYS_BMODEn | Input | **Boot Mode Control n** Selects the boot mode of the processor. |
| SYS_CLKIN | Input | **Clock/Crystal Input** Connect to an external clock source or crystal. |
| SYS_CLKOUT | Output | **Processor Clock Output** Outputs internal clocks. Clocks may be divided down. See the CGU chapter in the processor hardware reference for more details. |
| SYS_EXTWAKE | Output | **External Wake Control** Drives low during hibernate and high all other times. Typically connected to the enable input of the voltage regulator controlling the $V_{DD\_INT}$ supply. |
| $\overline{\text{SYS\_FAULT}}$ | I/O | **Complementary Fault** Complement of SYS_FAULT. |
| SYS_FAULT | I/O | **Fault** Indicates internal faults or senses external faults depending on the operating mode. |
| $\overline{\text{SYS\_HWRST}}$ | Input | **Processor Hardware Reset Control** Resets the device when asserted. |
| $\overline{\text{SYS\_IDLEn}}$ | Output | **Core n Idle Indicator** When low indicates that core n is in idle mode or being held in reset. |
| $\overline{\text{SYS\_NMI}}$ | Input | **Non-maskable Interrupt** Priority depends on the core that receives the interrupt. See the processor hardware and programming references for more details. |
| SYS_PWRGD | Input | **Power Good Indicator** When high it indicates to the processor that the $V_{DD\_INT}$ level is within specifications such that it is safe to begin booting upon return from hibernate. |
| $\overline{\text{SYS\_RESOUT}}$ | Output | **Reset Output** Indicates that the device is in the reset state. |
| $\overline{\text{SYS\_SLEEP}}$ | Output | **Processor Sleep Indicator** When low indicates that the processor is in the deep sleep power saving mode. |
| SYS_TDA | Input | **Thermal Diode Anode** May be used by an external temperature sensor to measure the die temperature. |
| SYS_TDK | Input | **Thermal Diode Cathode** May be used by an external temperature sensor to measure the die temperature. |
| SYS_XTAL | Output | **Crystal Output** Drives an external crystal. Must be left unconnected if an external clock is driving CLKIN. |
| TMR_ACIn | Input | **Alternate Capture Input n** Provides an additional input for WIDCAP, WATCHDOG, and PININT modes. |
| TMR_ACLKn | Input | **Alternate Clock n** Provides an additional time base for use by an individual timer. |
| TMR_CLK | Input | **Clock** Provides an additional global time base for use by all the GP timers. |
| TMR_TMRn | I/O | **Timer n** The main input/output signal for each timer. |
| TWI_SCL | I/O | **Serial Clock** Clock output when master, clock input when slave. |
| TWI_SDA | I/O | **Serial Data** Receives or transmits data. |
| $\overline{\text{UART\_CTS}}$ | Input | **Clear to Send** Flow control signal. |
| $\overline{\text{UART\_RTS}}$ | Output | **Request to Send** Flow control signal. |
| $\overline{\text{UART\_RX}}$ | Input | **Receive** Receive input. Typically connects to a transceiver that meets the electrical requirements of the device being communicated with. |
| $\overline{\text{UART\_TX}}$ | Output | **Transmit** Transmit output. Typically connects to a transceiver that meets the electrical requirements of the device being communicated with. |
| USB_CLKIN | Input | **Clock/Crystal Input** This clock input is multiplied by a PLL to form the USB clock. See Universal Serial Bus (USB) On-The-Go—Receive and Transmit Timing for frequency/tolerance information. |
| USB_DM | I/O | **Data –** Bidirectional differential data line. |
| USB_DP | I/O | **Data +** Bidirectional differential data line. |
| USB_ID | Input | **OTG ID** Senses whether the controller is a host or device. This signal is pulled low when an A-type plug is sensed (signifying that the USB controller is the A device), but the input is high when a B-type plug is sensed (signifying that the USB controller is the B device). |
| USB_VBC | Output | **VBUS Control** Controls an external voltage source to supply VBUS when in host mode. May be configured as open drain. Polarity is configurable as well. |
| USB_VBUS | I/O | **Bus Voltage** Connects to bus voltage in host and device modes. |

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

| Signal Name | Description | Port | Pin Name |
|---|---|---|---|
| PWM1_AH | PWM1 Channel A High Side | G | PG_03 |
| PWM1_AL | PWM1 Channel A Low Side | G | PG_02 |
| PWM1_BH | PWM1 Channel B High Side | G | PG_00 |
| PWM1_BL | PWM1 Channel B Low Side | E | PE_15 |
| PWM1_CH | PWM1 Channel C High Side | E | PE_13 |
| PWM1_CL | PWM1 Channel C Low Side | E | PE_12 |
| PWM1_DH | PWM1 Channel D High Side | E | PE_11 |
| PWM1_DL | PWM1 Channel D Low Side | E | PE_10 |
| PWM1_SYNC | PWM1 Sync | G | PG_05 |
| $\overline{\text{PWM1\_TRIP0}}$ | PWM1 Shutdown Input 0 | G | PG_06 |
| $\overline{\text{PWM1\_TRIP1}}$ | PWM1 Shutdown Input 1 | G | PG_08 |
| RSI0_CLK | RSI0 Clock | G | PG_06 |
| RSI0_CMD | RSI0 Command | G | PG_05 |
| RSI0_D0 | RSI0 Data 0 | G | PG_03 |
| RSI0_D1 | RSI0 Data 1 | G | PG_02 |
| RSI0_D2 | RSI0 Data 2 | G | PG_00 |
| RSI0_D3 | RSI0 Data 3 | E | PE_15 |
| RSI0_D4 | RSI0 Data 4 | E | PE_13 |
| RSI0_D5 | RSI0 Data 5 | E | PE_12 |
| RSI0_D6 | RSI0 Data 6 | E | PE_10 |
| RSI0_D7 | RSI0 Data 7 | E | PE_11 |
| SMC0_A01 | SMC0 Address 1 | Not Muxed | SMC0_A01 |
| SMC0_A02 | SMC0 Address 2 | Not Muxed | SMC0_A02 |
| SMC0_A03 | SMC0 Address 3 | A | PA_00 |
| SMC0_A04 | SMC0 Address 4 | A | PA_01 |
| SMC0_A05 | SMC0 Address 5 | A | PA_02 |
| SMC0_A06 | SMC0 Address 6 | A | PA_03 |
| SMC0_A07 | SMC0 Address 7 | A | PA_04 |
| SMC0_A08 | SMC0 Address 8 | A | PA_05 |
| SMC0_A09 | SMC0 Address 9 | A | PA_06 |
| SMC0_A10 | SMC0 Address 10 | A | PA_07 |
| SMC0_A11 | SMC0 Address 11 | A | PA_08 |
| SMC0_A12 | SMC0 Address 12 | A | PA_09 |
| SMC0_A13 | SMC0 Address 13 | B | PB_02 |
| SMC0_A14 | SMC0 Address 14 | A | PA_10 |
| SMC0_A15 | SMC0 Address 15 | A | PA_11 |
| SMC0_A16 | SMC0 Address 16 | B | PB_03 |
| SMC0_A17 | SMC0 Address 17 | A | PA_12 |
| SMC0_A18 | SMC0 Address 18 | A | PA_13 |
| SMC0_A19 | SMC0 Address 19 | A | PA_14 |
| SMC0_A20 | SMC0 Address 20 | A | PA_15 |
| SMC0_A21 | SMC0 Address 21 | B | PB_06 |
| SMC0_A22 | SMC0 Address 22 | B | PB_07 |
| SMC0_A23 | SMC0 Address 23 | B | PB_08 |
| SMC0_A24 | SMC0 Address 24 | B | PB_10 |
| SMC0_A25 | SMC0 Address 25 | B | PB_11 |

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

| Signal Name | Description | Port | Pin Name |
|---|---|---|---|
| SYS_BMODE2 | Boot Mode Control 2 | Not Muxed | SYS_BMODE2 |
| SYS_CLKIN | Clock/Crystal Input | Not Muxed | SYS_CLKIN |
| SYS_CLKOUT | Processor Clock Output | Not Muxed | SYS_CLKOUT |
| SYS_EXTWAKE | External Wake Control | Not Muxed | SYS_EXTWAKE |
| SYS_FAULT | Fault Output | Not Muxed | SYS_FAULT |
| $\overline{\text{SYS\_FAULT}}$ | Complementary Fault Output | Not Muxed | $\overline{\text{SYS\_FAULT}}$ |
| $\overline{\text{SYS\_HWRST}}$ | Processor Hardware Reset Control | Not Muxed | $\overline{\text{SYS\_HWRST}}$ |
| $\overline{\text{SYS\_IDLE0}}$ | Core 0 Idle Indicator | G | PG_15 |
| $\overline{\text{SYS\_IDLE1}}$ | Core 1 Idle Indicator | G | PG_14 |
| $\overline{\text{SYS\_NMI}}$ | Non-maskable Interrupt | Not Muxed | $\overline{\text{SYS\_NMI\_RESOUT}}$ |
| SYS_PWRGD | Power Good Indicator | Not Muxed | SYS_PWRGD |
| $\overline{\text{SYS\_RESOUT}}$ | Reset Output | Not Muxed | $\overline{\text{SYS\_NMI\_RESOUT}}$ |
| $\overline{\text{SYS\_SLEEP}}$ | Processor Sleep Indicator | G | PG_15 |
| SYS_TDA | Thermal Diode Anode | Not Muxed | SYS_TDA |
| SYS_TDK | Thermal Diode Cathode | Not Muxed | SYS_TDK |
| SYS_XTAL | Crystal Output | Not Muxed | SYS_XTAL |
| TM0_ACI0 | TIMER0 Alternate Capture Input 0 | D | PD_08 |
| TM0_ACI1 | TIMER0 Alternate Capture Input 1 | G | PG_14 |
| TM0_ACI2 | TIMER0 Alternate Capture Input 2 | G | PG_04 |
| TM0_ACI3 | TIMER0 Alternate Capture Input 3 | D | PD_07 |
| TM0_ACI4 | TIMER0 Alternate Capture Input 4 | G | PG_15 |
| TM0_ACI5 | TIMER0 Alternate Capture Input 5 | D | PD_06 |
| TM0_ACI6 | TIMER0 Alternate Capture Input 6 | B | PB_13 |
| TM0_ACLK0 | TIMER0 Alternate Clock 0 | B | PB_10 |
| TM0_ACLK1 | TIMER0 Alternate Clock 1 | B | PB_12 |
| TM0_ACLK2 | TIMER0 Alternate Clock 2 | B | PB_09 |
| TM0_ACLK3 | TIMER0 Alternate Clock 3 | B | PB_11 |
| TM0_ACLK4 | TIMER0 Alternate Clock 4 | B | PB_06 |
| TM0_ACLK5 | TIMER0 Alternate Clock 5 | D | PD_13 |
| TM0_ACLK6 | TIMER0 Alternate Clock 6 | D | PD_14 |
| TM0_ACLK7 | TIMER0 Alternate Clock 7 | D | PD_05 |
| TM0_CLK | TIMER0 Clock | G | PG_13 |
| TM0_TMR0 | TIMER0 Timer 0 | E | PE_14 |
| TM0_TMR1 | TIMER0 Timer 1 | G | PG_04 |
| TM0_TMR2 | TIMER0 Timer 2 | G | PG_01 |
| TM0_TMR3 | TIMER0 Timer 3 | G | PG_08 |
| TM0_TMR4 | TIMER0 Timer 4 | G | PG_09 |
| TM0_TMR5 | TIMER0 Timer 5 | G | PG_07 |
| TM0_TMR6 | TIMER0 Timer 6 | G | PG_11 |
| TM0_TMR7 | TIMER0 Timer 7 | G | PG_12 |
| TWI0_SCL | TWI0 Serial Clock | Not Muxed | TWI0_SCL |
| TWI0_SDA | TWI0 Serial Data | Not Muxed | TWI0_SDA |
| TWI1_SCL | TWI1 Serial Clock | Not Muxed | TWI1_SCL |
| TWI1_SDA | TWI1 Serial Data | Not Muxed | TWI1_SDA |
| $\overline{\text{UART0\_CTS}}$ | UART0 Clear to Send | D | PD_10 |
| $\overline{\text{UART0\_RTS}}$ | UART0 Request to Send | D | PD_09 |

# GP I/O MULTIPLEXING FOR 349-BALL CSP_BGA

Table 8 through Table 14 identifies the pin functions that are multiplexed on the general-purpose I/O pins of the 349-ball CSP_BGA package.

**Table 8. Signal Multiplexing for Port A**

| Signal Name | Multiplexed Function 0 | Multiplexed Function 1 | Multiplexed Function 2 | Multiplexed Function Input Tap |
|---|---|---|---|---|
| PA_00 | SMC0_A03 | PPI2_D00 | LP0_D0 | |
| PA_01 | SMC0_A04 | PPI2_D01 | LP0_D1 | |
| PA_02 | SMC0_A05 | PPI2_D02 | LP0_D2 | |
| PA_03 | SMC0_A06 | PPI2_D03 | LP0_D3 | |
| PA_04 | SMC0_A07 | PPI2_D04 | LP0_D4 | |
| PA_05 | SMC0_A08 | PPI2_D05 | LP0_D5 | |
| PA_06 | SMC0_A09 | PPI2_D06 | LP0_D6 | |
| PA_07 | SMC0_A10 | PPI2_D07 | LP0_D7 | |
| PA_08 | SMC0_A11 | PPI2_D08 | LP1_D0 | |
| PA_09 | SMC0_A12 | PPI2_D09 | LP1_D1 | |
| PA_10 | SMC0_A14 | PPI2_D10 | LP1_D2 | |
| PA_11 | SMC0_A15 | PPI2_D11 | LP1_D3 | |
| PA_12 | SMC0_A17 | PPI2_D12 | LP1_D4 | |
| PA_13 | SMC0_A18 | PPI2_D13 | LP1_D5 | |
| PA_14 | SMC0_A19 | PPI2_D14 | LP1_D6 | |
| PA_15 | SMC0_A20 | PPI2_D15 | LP1_D7 | |

**Table 9. Signal Multiplexing for Port B**

| Signal Name | Multiplexed Function 0 | Multiplexed Function 1 | Multiplexed Function 2 | Multiplexed Function Input Tap |
|---|---|---|---|---|
| PB_00 | SMC0_NORCLK | PPI2_CLK | LP0_CLK | |
| PB_01 | $\overline{\text{SMC0\_AMS1}}$ | PPI2_FS1 | LP0_ACK | |
| PB_02 | SMC0_A13 | PPI2_FS2 | LP1_ACK | |
| PB_03 | SMC0_A16 | PPI2_FS3 | LP1_CLK | |
| PB_04 | $\overline{\text{SMC0\_AMS2}}$ | $\overline{\text{SMC0\_ABE0}}$ | SPT0_AFS | |
| PB_05 | $\overline{\text{SMC0\_AMS3}}$ | $\overline{\text{SMC0\_ABE1}}$ | SPT0_ACLK | |
| PB_06 | SMC0_A21 | SPT0_ATDV | | TM0_ACLK4 |
| PB_07 | SMC0_A22 | PPI2_D16 | SPT0_BFS | |
| PB_08 | SMC0_A23 | PPI2_D17 | SPT0_BCLK | |
| PB_09 | $\overline{\text{SMC0\_BGH}}$ | | SPT0_AD0 | TM0_ACLK2 |
| PB_10 | SMC0_A24 | | SPT0_BD1 | TM0_ACLK0 |
| PB_11 | SMC0_A25 | | SPT0_BD0 | TM0_ACLK3 |
| PB_12 | $\overline{\text{SMC0\_BG}}$ | SPT0_BTDV | SPT0_AD1 | TM0_ACLK1 |
| PB_13 | ETH0_TXEN | PPI1_FS1 | | TM0_ACI6 |
| PB_14 | ETH0_REFCLK | PPI1_CLK | | |
| PB_15 | ETH0_PTPPPS | PPI1_FS3 | | |

**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 |
|---|---|---|---|---|---|---|---|---|---|
| PA_00 | I/O | A | wk | wk | none | wk | none | VDD_EXT | Desc: PA Position 0 \| SMC0 Address 3 \| EPPI2 Data 0 \| LP0 Data 0.<br>Notes: No notes. |
| PA_01 | I/O | A | wk | wk | none | wk | none | VDD_EXT | Desc: PA Position 1 \| SMC0 Address 4 \| EPPI2 Data 1 \| LP0 Data 1.<br>Notes: No notes. |
| PA_02 | I/O | A | wk | wk | none | wk | none | VDD_EXT | Desc: PA Position 2 \| SMC0 Address 5 \| EPPI2 Data 2 \| LP0 Data 2.<br>Notes: No notes. |
| PA_03 | I/O | A | wk | wk | none | wk | none | VDD_EXT | Desc: PA Position 3 \| SMC0 Address 6 \| EPPI2 Data 3 \| LP0 Data 3.<br>Notes: No notes. |
| PA_04 | I/O | A | wk | wk | none | wk | none | VDD_EXT | Desc: PA Position 4 \| SMC0 Address 7 \| EPPI2 Data 4 \| LP0 Data 4.<br>Notes: No notes. |
| PA_05 | I/O | A | wk | wk | none | wk | none | VDD_EXT | Desc: PA Position 5 \| SMC0 Address 8 \| EPPI2 Data 5 \| LP0 Data 5.<br>Notes: No notes. |
| PA_06 | I/O | A | wk | wk | none | wk | none | VDD_EXT | Desc: PA Position 6 \| SMC0 Address 9 \| EPPI2 Data 6 \| LP0 Data 6.<br>Notes: No notes. |
| PA_07 | I/O | A | wk | wk | none | wk | none | VDD_EXT | Desc: PA Position 7 \| SMC0 Address 10 \| EPPI2 Data 7 \| LP0 Data 7.<br>Notes: No notes. |
| PA_08 | I/O | A | wk | wk | none | wk | none | VDD_EXT | Desc: PA Position 8 \| SMC0 Address 11 \| EPPI2 Data 8 \| LP1 Data 0.<br>Notes: No notes. |
| PA_09 | I/O | A | wk | wk | none | wk | none | VDD_EXT | Desc: PA Position 9 \| SMC0 Address 12 \| EPPI2 Data 9 \| LP1 Data 1.<br>Notes: No notes. |
| PA_10 | I/O | A | wk | wk | none | wk | none | VDD_EXT | Desc: PA Position 10 \| SMC0 Address 14 \| EPPI2 Data 10 \| LP1 Data 2.<br>Notes: No notes. |
| PA_11 | I/O | A | wk | wk | none | wk | none | VDD_EXT | Desc: PA Position 11 \| SMC0 Address 15 \| EPPI2 Data 11 \| LP1 Data 3.<br>Notes: No notes. |
| PA_12 | I/O | A | wk | wk | none | wk | none | VDD_EXT | Desc: PA Position 12 \| SMC0 Address 17 \| EPPI2 Data 12 \| LP1 Data 4.<br>Notes: No notes. |
| PA_13 | I/O | A | wk | wk | none | wk | none | VDD_EXT | Desc: PA Position 13 \| SMC0 Address 18 \| EPPI2 Data 13 \| LP1 Data 5.<br>Notes: No notes. |
| PA_14 | I/O | A | wk | wk | none | wk | none | VDD_EXT | Desc: PA Position 14 \| SMC0 Address 19 \| EPPI2 Data 14 \| LP1 Data 6.<br>Notes: No notes. |
| PA_15 | I/O | A | wk | wk | none | wk | none | VDD_EXT | Desc: PA Position 15 \| SMC0 Address 20 \| EPPI2 Data 15 \| LP1 Data 7.<br>Notes: May be used to wake the processor from hibernate or deep sleep mode. |

## ELECTRICAL CHARACTERISTICS

| Parameter | | Test Conditions | Min | Typical | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{OH}$[1] | High Level Output Voltage | $V_{DD\_EXT}$ = 1.7 V, $I_{OH}$ = –0.5 mA | $V_{DD\_EXT}$ – 0.40 | | | V |
| $V_{OH}$[1] | High Level Output Voltage | $V_{DD\_EXT}$ = 3.13 V, $I_{OH}$ = –0.5 mA | $V_{DD\_EXT}$ – 0.40 | | | V |
| $V_{OH\_DDR2}$[2] | High Level Output Voltage, ds = 00 | $V_{DD\_DMC}$ = 1.70 V, $I_{OH}$ = –13.4 mA | 1.388 | | | V |
| $V_{OH\_DDR2}$[3] | High Level Output Voltage, ds = 10 | $V_{DD\_DMC}$ = 1.70 V, $I_{OH}$ = –6.70 mA | 1.311 | | | V |
| $V_{OH\_LPDDR}$[4] | High Level Output Voltage, ds = 00 | $V_{DD\_DMC}$ = 1.70 V, $I_{OH}$ = –11.2 mA | 1.300 | | | V |
| $V_{OH\_LPDDR}$[5] | High Level Output Voltage, ds = 01 | $V_{DD\_DMC}$ = 1.70 V, $I_{OH}$ = –7.85 mA | 1.300 | | | V |
| $V_{OH\_LPDDR}$[6] | High Level Output Voltage, ds = 10 | $V_{DD\_DMC}$ = 1.70 V, $I_{OH}$ = –5.10 mA | 1.300 | | | V |
| $V_{OH\_LPDDR}$[7] | High Level Output Voltage, ds = 11 | $V_{DD\_DMC}$ = 1.70 V, $I_{OH}$ = –2.55 mA | 1.300 | | | V |
| $V_{OL}$[8] | Low Level Output Voltage | $V_{DD\_EXT}$ = 1.7 V, $I_{OL}$ = 2.0 mA | | | 0.400 | V |
| $V_{OL}$[8] | Low Level Output Voltage | $V_{DD\_EXT}$ = 3.13 V, $I_{OL}$ = 2.0 mA | | | 0.400 | V |
| $V_{OL\_DDR2}$[2] | Low Level Output Voltage, ds = 00 | $V_{DD\_DMC}$ = 1.70 V, $I_{OL}$13.4 mA | | | 0.312 | V |
| $V_{OL\_DDR2}$[3] | Low Level Output Voltage, ds = 10 | $V_{DD\_DMC}$ = 1.70 V, $I_{OL}$ = 6.70 mA | | | 0.390 | V |
| $V_{OL\_LPDDR}$[4] | Low Level Output Voltage, ds = 00 | $V_{DD\_DMC}$ = 1.70 V, $I_{OL}$ = 11.2 mA | | | 0.400 | V |
| $V_{OL\_LPDDR}$[5] | Low Level Output Voltage, ds = 01 | $V_{DD\_DMC}$ = 1.70 V, $I_{OL}$ = 7.85 mA | | | 0.400 | V |
| $V_{OL\_LPDDR}$[6] | Low Level Output Voltage, ds = 10 | $V_{DD\_DMC}$ = 1.70 V, $I_{OL}$ = 5.10 mA | | | 0.400 | V |
| $V_{OL\_LPDDR}$[7] | Low Level Output Voltage, ds = 11 | $V_{DD\_DMC}$ = 1.70 V, $I_{OL}$ = 2.55 mA | | | 0.400 | V |
| $I_{IH}$[9] | High Level Input Current | $V_{DD\_EXT}$ = 3.47 V, $V_{DD\_DMC}$ = 1.9 V, $V_{DD\_USB}$ = 3.47 V, $V_{IN}$ = 3.47 V | | | 10 | µA |
| $I_{IH\_PD}$[10] | High Level Input Current with Pull-down Resistor | $V_{DD\_EXT}$ = 3.47 V, $V_{DD\_DMC}$ = 1.9 V, $V_{DD\_USB}$ = 3.47 V, $V_{IN}$ = 3.47 V | | | 110 | µA |
| $I_{IL}$[11] | Low Level Input Current | $V_{DD\_EXT}$ = 3.47 V, $V_{DD\_DMC}$ = 1.9 V, $V_{DD\_USB}$ = 3.47 V, $V_{IN}$ = 0 V | | | 10 | µA |
| $I_{IL\_PU}$[12] | Low Level Input Current with Pull-up Resistor | $V_{DD\_EXT}$ = 3.47 V, $V_{DD\_DMC}$ = 1.9 V, $V_{DD\_USB}$ = 3.47 V, $V_{IN}$ = 0 V | | | 100 | µA |
| $I_{IH\_USB0}$[13] | High Level Input Current | $V_{DD\_EXT}$ = 3.47 V, $V_{DD\_DMC}$ = 1.9 V, $V_{DD\_USB}$ = 3.47 V, $V_{IN}$ = 3.47 V | | | 240 | µA |
| $I_{IL\_USB0}$[13] | Low Level Input Current | $V_{DD\_EXT}$ = 3.47 V, $V_{DD\_DMC}$ = 1.9 V, $V_{DD\_USB}$ = 3.47 V, $V_{IN}$ = 0 V | | | 100 | µA |
| $I_{OZH}$[14] | Three-State Leakage Current | $V_{DD\_EXT}$ = 3.47 V, $V_{DD\_DMC}$ = 1.9 V, $V_{DD\_USB}$ = 3.47 V, $V_{IN}$ = 3.47 V | | | 10 | µA |
| $I_{OZH}$[15] | Three-State Leakage Current | $V_{DD\_EXT}$ = 3.47 V, $V_{DD\_DMC}$ = 1.9 V, $V_{DD\_USB}$ = 3.47 V, $V_{IN}$ = 1.9 V | | | 10 | µA |
| $I_{OZL}$[16] | Three-State Leakage Current | $V_{DD\_EXT}$ = 3.47 V, $V_{DD\_DMC}$ = 1.9 V, $V_{DD\_USB}$ = 3.47 V, $V_{IN}$ = 0 V | | | 10 | µA |
| $I_{OZL\_PU}$[17] | Three-State Leakage Current with Pull-up Resistor | $V_{DD\_EXT}$ = 3.47 V, $V_{DD\_DMC}$ = 1.9 V, $V_{DD\_USB}$ = 3.47 V, $V_{IN}$ = 0 V | | | 100 | µA |
| $I_{OZH\_TWI}$[18] | Three-State Leakage Current | $V_{DD\_EXT}$ = 3.47 V, $V_{DD\_DMC}$ = 1.9 V, $V_{DD\_USB}$ = 3.47 V, $V_{IN}$ = 5.5 V | | | 10 | µA |
| $C_{IN}$[19, 20] | Input Capacitance | $T_{AMBIENT}$ = 25°C | | 4.9 | 6.7 | pF |
| $C_{IN\_TWI}$[18, 20] | Input Capacitance | $T_{AMBIENT}$ = 25°C | | 8.9 | 9.9 | pF |
| $C_{IN\_DDR}$[20, 21] | Input Capacitance | $T_{AMBIENT}$ = 25°C | | 5.8 | 6.6 | pF |
| $I_{DD\_TD}$ | $V_{DD\_TD}$ Current | $V_{DD\_TD}$ = 3.3 V | | | 1 | µA |
| $I_{DD\_DEEPSLEEP}$[22, 23] | $V_{DD\_INT}$ Current in Deep Sleep Mode | $f_{CCLK}$ = 0 MHz<br>$f_{SCLK0/1}$ = 0 MHz | | Table 21 on Page 58 | | mA |

| Parameter | | Test Conditions | Min | Typical | Max | Unit |
|---|---|---|---|---|---|---|
| $I_{DD\_IDLE}$[23] | $V_{DD\_INT}$ Current in Idle | $f_{CCLK}$ = 500 MHz<br>ASFC0 = 0.14 (Idle)<br>ASFC1 = 0 (Disabled)<br>$f_{SYSCLK}$ = 250 MHz, $f_{SCLK0/1}$ = 125 MHz<br>$f_{DCLK}$ = 0 MHz (DDR Disabled)<br>$f_{USBCLK}$ = 0 MHz (USB Disabled)<br>No PVP or DMA activity<br>$T_J$ = 25°C | | 137 | | mA |
| $I_{DD\_TYP}$[23] | $V_{DD\_INT}$ Current | $f_{CCLK}$ = 500 MHz<br>ASFC0 = 1.0 (Full-on Typical)<br>ASFC1 = 0.86 (App)<br>$f_{SYSCLK}$ = 250 MHz, $f_{SCLK0/1}$ = 125 MHz<br>$f_{DCLK}$ = 250 MHz<br>$f_{USBCLK}$ = 0 MHz (USB Disabled)<br>DMA Data Rate = 124 MB/s<br>Medium PVP Activity<br>$T_J$ = 25°C | | 357 | | mA |
| $I_{DD\_HIBERNATE}$[22, 24] | Hibernate State Current | $V_{DD\_INT}$ = 0 V,<br>$V_{DD\_EXT}$ = $V_{DD\_TD}$ = $V_{DD\_USB}$ = 3.3 V,<br>$V_{DD\_DMC}$ = 1.8 V, $V_{REF\_DMC}$ = 0.9 V,<br>$T_J$ = 25°C, $f_{CLKIN}$ = 0 MHz | | 40 | | µA |
| $I_{DD\_HIBERNATE}$[22, 24] | Hibernate State Current Without USB | $V_{DD\_INT}$ = 0 V,<br>$V_{DD\_EXT}$ = $V_{DD\_TD}$ = $V_{DD\_USB}$ = 3.3 V,<br>$V_{DD\_DMC}$ = 1.8 V, $V_{REF\_DMC}$ = 0.9 V,<br>$T_J$ = 25°C,<br>$f_{CLKIN}$ = 0 MHz, USB protection disabled (USB0_PHY_CTL.DIS=1) | | 10 | | µA |
| $I_{DD\_INT}$[23] | $V_{DD\_INT}$ Current | $f_{CCLK}$ > 0 MHz<br>$f_{SCLK0/1}$ ≥ 0 MHz | | | See $I_{DDINT\_TOT}$ equation on Page 57 | mA |

[1] Applies to all output and bidirectional signals except DMC0 signals, TWI signals and USB0 signals.
[2] Applies to all DMC0 output and bidirectional signals in DDR2 full drive strength mode.
[3] Applies to all DMC0 output and bidirectional signals in DDR2 half drive strength mode.
[4] Applies to all DMC0 output and bidirectional signals in LPDDR full drive strength mode.
[5] Applies to all DMC0 output and bidirectional signals in LPDDR three-quarter drive strength mode.
[6] Applies to all DMC0 output and bidirectional signals in LPDDR half drive strength mode.
[7] Applies to all DMC0 output and bidirectional signals in LPDDR one-quarter drive strength mode.
[8] Applies to all output and bidirectional signals except DMC0 signals and USB0 signals.
[9] Applies to signals SMC0_ARDY, $\overline{\text{SMC0\_BR}}$, SYS_BMODE0–2, SYS_CLKIN, SYS_HWRST, SYS_PWRGD, JTG_TDI, and $\overline{\text{JTG\_TMS}}$.
[10] Applies to signals JTG_TCK and $\overline{\text{JTG\_TRST}}$.
[11] Applies to signals SMC0_ARDY, $\overline{\text{SMC0\_BR}}$, SYS_BMODE0–2, SYS_CLKIN, SYS_HWRST, SYS_PWRGD, JTG_TCK, and $\overline{\text{JTG\_TRST}}$.
[12] Applies to signals JTG_TDI, JTG_TMS.
[13] Applies to signal USB0_CLKIN.
[14] Applies to signals PA0–15, PB0–15, PC0–15, PD0–15, PE0–15, PF0–15, PG0–15, $\overline{\text{SMC0\_AMS0}}$, $\overline{\text{SMC0\_ARE}}$, $\overline{\text{SMC0\_AWE}}$, $\overline{\text{SMC0\_A0E}}$, SMC0_A01–02, SMC0_D00–15, SYS_FAULT, $\overline{\text{SYS\_FAULT}}$, $\overline{\text{JTG\_EMU}}$, JTG_TDO, USB0_DM, USB0_DP, USB0_ID, USB0_VBC, USB0_VBUS.
[15] Applies to DMC0_A[00:13], $\overline{\text{DMC0\_BA}}$[0:2], $\overline{\text{DMC0\_CAS}}$, $\overline{\text{DMC0\_CS0}}$, DMC0_DQ[00:15], DMC0_LQDS, $\overline{\text{DMC0\_LDQS}}$, DMC0_UDQS, $\overline{\text{DMC0\_UDQS}}$, DMC0_LDM, DMC0_UDM, DMC0_ODT, $\overline{\text{DMC0\_RAS}}$, and $\overline{\text{DMC0\_WE}}$.
[16] Applies to signals PA0–15, PB0–15, PC0–15, PD0–15, PE0–15, PF0–15, PG0–15, $\overline{\text{SMC0\_A0E}}$, SMC0_A01–02, SMC0_D00–15, SYS_FAULT, $\overline{\text{SYS\_FAULT}}$, $\overline{\text{JTG\_EMU}}$, JTG_TDO, USB0_DM, USB0_DP, USB0_ID, USB0_VBC, USB0_VBUS, DMC0_A00–13, DMC0_BA0–2, $\overline{\text{DMC0\_CAS}}$, $\overline{\text{DMC0\_CS0}}$, DMC0_DQ00–15, DMC0_LQDS, $\overline{\text{DMC0\_LDQS}}$, DMC0_UDQS, $\overline{\text{DMC0\_UDQS}}$, DMC0_LDM, DMC0_UDM, DMC0_ODT, $\overline{\text{DMC0\_RAS}}$, $\overline{\text{DMC0\_WE}}$, and TWI signals.
[17] Applies to signals $\overline{\text{SMC0\_AMS0}}$, $\overline{\text{SMC0\_ARE}}$, $\overline{\text{SMC0\_AWE}}$, and when RSI pull-up resistors are enabled, PE10–13, 15 and PG00, 02, 03, 05.
[18] Applies to all TWI signals.
[19] Applies to all signals, except DMC0 and TWI signals.
[20] Guaranteed, but not tested.
[21] Applies to all DMC0 signals.
[22] See the *ADSP-BF60x Blackfin Processor Hardware Reference Manual* for definition of deep sleep and hibernate operating modes.
[23] Additional information can be found at Total Internal Power Dissipation on Page 57.
[24] Applies to $V_{DD\_EXT}$, $V_{DD\_DMC}$, $V_{DD\_USB}$ and $V_{DD\_TD}$ supply signals only. Clock inputs are tied high or low.

### *Asynchronous Flash Read*

**Table 29. Asynchronous Flash Read**

| Parameter | | $V_{DD\_EXT}$ 1.8 V/3.3 V Nominal Min | Max | Unit |
|---|---|---|---|---|
| *Switching Characteristics* | | | | |
| $t_{AMSADV}$ | SMC0_Ax (Address)/$\overline{\text{SMC0\_AMSx}}$ Assertion Before SMC0_NORDV Low[1] | PREST × $t_{SCLK0}$ – 2 | | ns |
| $t_{WADV}$ | SMC0_NORDV Active Low Width[2] | RST × $t_{SCLK0}$ – 2 | | ns |
| $t_{DADVARE}$ | $\overline{\text{SMC0\_ARE}}$ Low Delay From SMC0_NORDV High[3] | PREAT × $t_{SCLK0}$ – 2 | | ns |
| $t_{HARE}$ | Output[4] Hold After $\overline{\text{SMC0\_ARE}}$ High[5] | RHT × $t_{SCLK0}$ – 2 | | ns |
| $t_{WARE}$[6] | $\overline{\text{SMC0\_ARE}}$ Active Low Width[7] | RAT × $t_{SCLK0}$ – 2 | | ns |

[1] PREST value set using the SMC_BxETIM.PREST bits.
[2] RST value set using the SMC_BxTIM.RST bits.
[3] PREAT value set using the SMC_BxETIM.PREAT bits.
[4] Output signals are SMC0_Ax, $\overline{\text{SMC0\_AMS}}$, $\overline{\text{SMC0\_AOE}}$.
[5] RHT value set using the SMC_BxTIM.RHT bits.
[6] SMC0_BxCTL.ARDYEN bit = 0.
[7] RAT value set using the SMC_BxTIM.RAT bits.

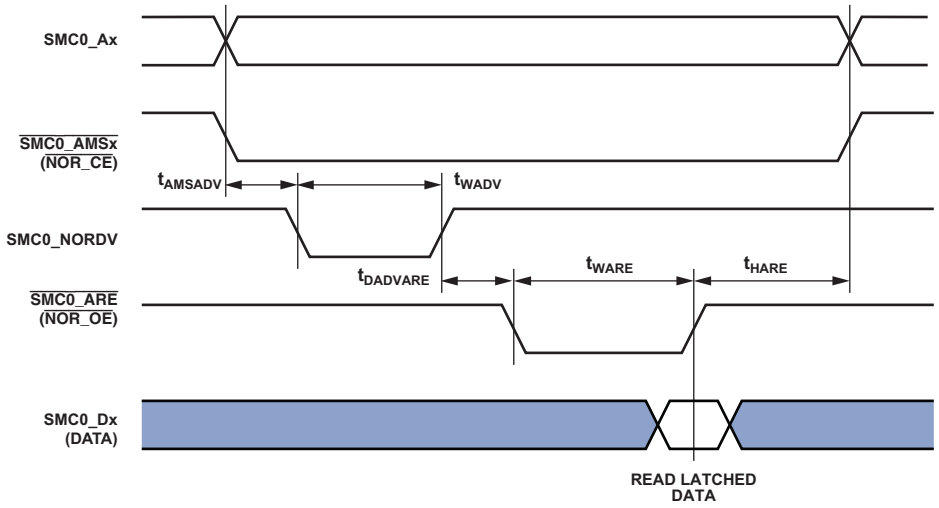
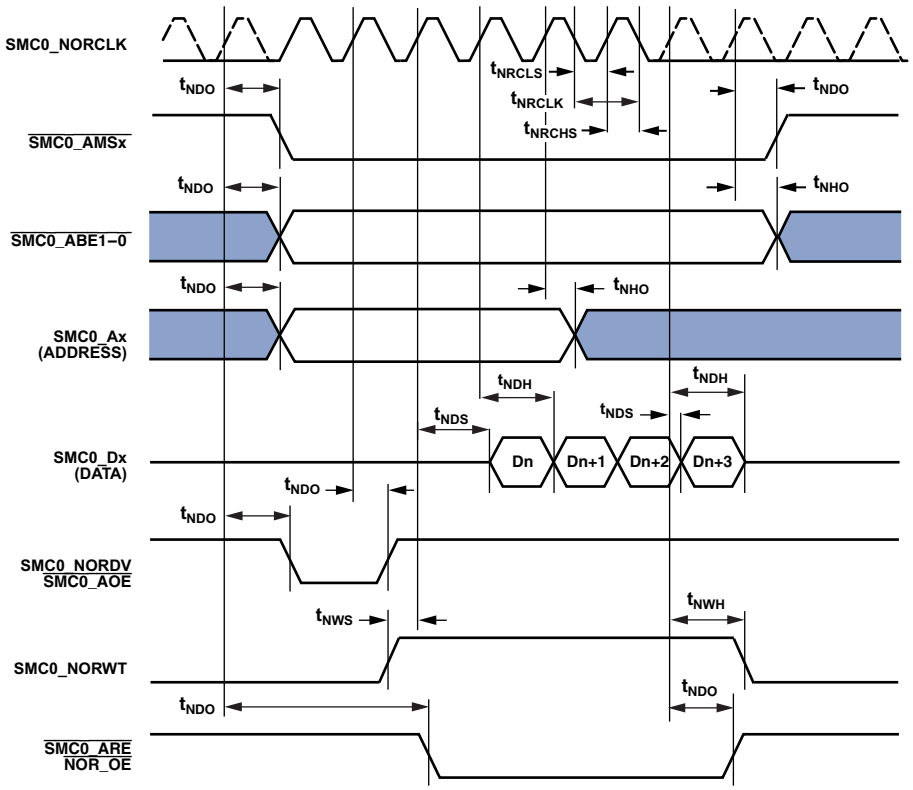



*Figure 13. Asynchronous Flash Read*




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

*Figure 15. Synchronous Burst AC Interface Timing*

### *DDR2 SDRAM Write Cycle Timing*

**Table 38. DDR2 SDRAM Write Cycle Timing, $V_{DD\_DMC}$ Nominal 1.8 V**

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

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

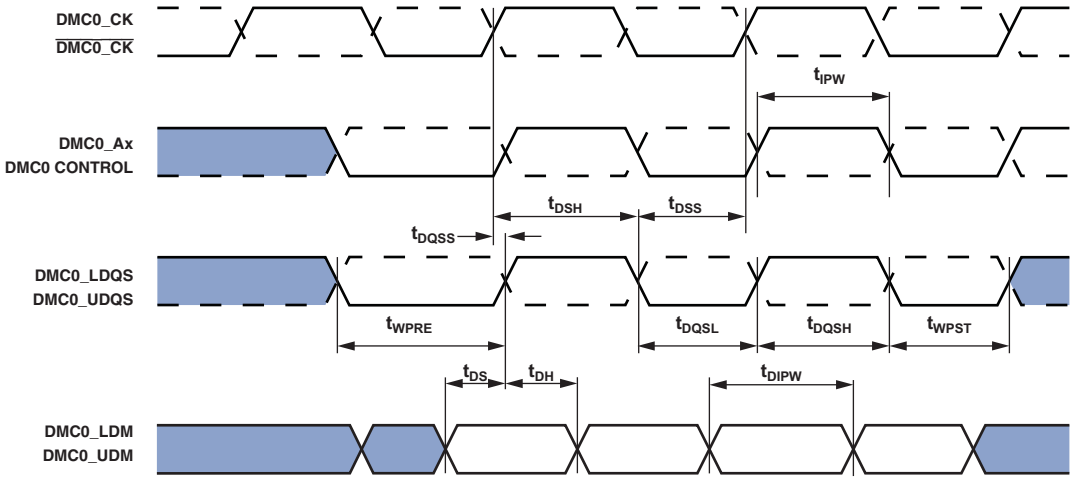



*Figure 21. DDR2 SDRAM Controller Output AC Timing*

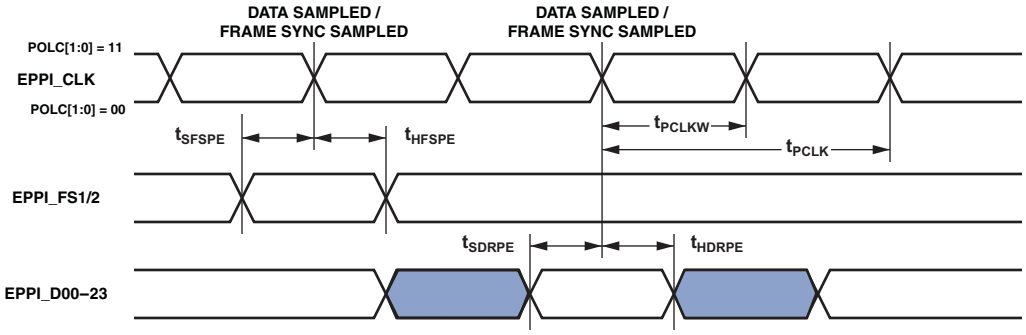



*Figure 32. PPI External Clock GP Receive Mode with External Frame Sync Timing*

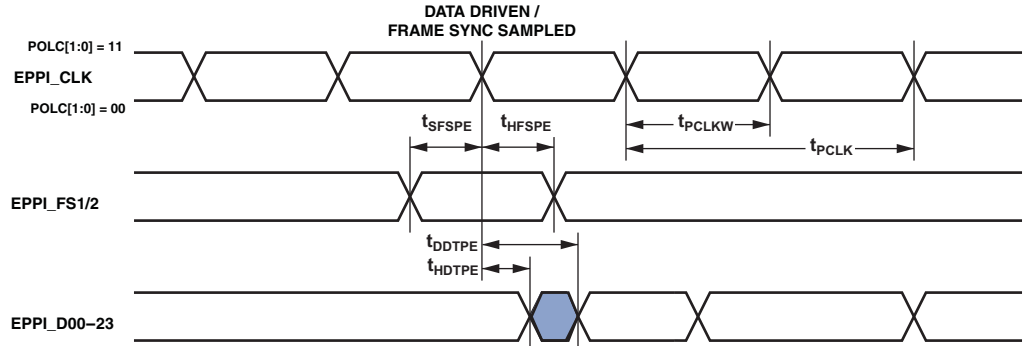



*Figure 33. PPI External Clock GP Transmit Mode with External Frame Sync Timing*

### Link Ports

In link port receive mode the link port clock is supplied externally and is called $f_{LCLKREXT}$:

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

In link port transmit mode the programmed link port clock ($f_{LCLKTPROG}$) frequency in MHz is set by the following equation where VALUE is a field in the LP_DIV register that can be set from 1 to 255:

$$f_{LCLKTPROG} = \frac{f_{SCLK0}}{(VALUE \times 2)}$$

In the case where VALUE = 0, $f_{LCLKTPROG} = f_{SCLK0}$. For all settings of VALUE the following equation also holds:

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

Calculation of link receiver data setup and hold relative to link clock is required to determine the maximum allowable skew that can be introduced in the transmission path length difference between LP_Dx (data) and LP_CLK. Setup skew is the maximum delay that can be introduced in LP_Dx relative to LP_CLK:
(setup skew = $t_{LCLKTWH}$ min – $t_{DLDCH}$ – $t_{SLDCL}$). Hold skew is the maximum delay that can be introduced in LP_CLK relative to LP_Dx: (hold skew = $t_{LCLKTWL}$ min – $t_{HLDCH}$ – $t_{HLDCL}$).

**Table 44. Link Ports—Receive**

| Parameter | | $V_{DD\_EXT}$ 1.8 V Nominal/3.3 V Nominal Min | Max | Unit |
|---|---|---|---|---|
| *Timing Requirements* | | | | |
| $t_{SLDCL}$ | Data Setup Before LP_CLK Low | 2 | | ns |
| $t_{HLDCL}$ | Data Hold After LP_CLK Low | 3 | | ns |
| $t_{LCLKIW}$ | LP_CLK Period[1] | $t_{LCLKREXT}$ – 1.5 | | ns |
| $t_{LCLKRWL}$ | LP_CLK Width Low[1] | (0.5 × $t_{LCLKREXT}$) – 1.5 | | ns |
| $t_{LCLKRWH}$ | LP_CLK Width High[1] | (0.5 × $t_{LCLKREXT}$) – 1.5 | | ns |
| *Switching Characteristic* | | | | |
| $t_{DLALC}$ | LP_ACK Low Delay After LP_CLK Low[2] | 1.5 × $t_{SCLK0}$ + 4 | 2.5 × $t_{SCLK0}$ + 12 | ns |

[1] This specification indicates the minimum instantaneous width or period that can be tolerated due to duty cycle variation or jitter on the external LP_CLK. For the external LP_CLK ideal maximum frequency see the $f_{LCLKTEXT}$ specification in Table 17 on Page 53 in Clock Related Operating Conditions.
[2] LP_ACK goes low with $t_{DLALC}$ relative to rise of LP_CLK after first byte, but does not go low if the receiver's link buffer is not about to fill.

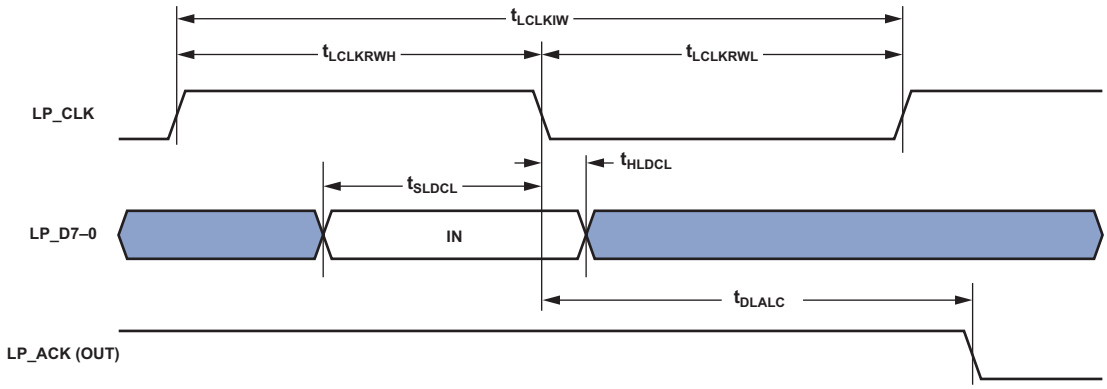
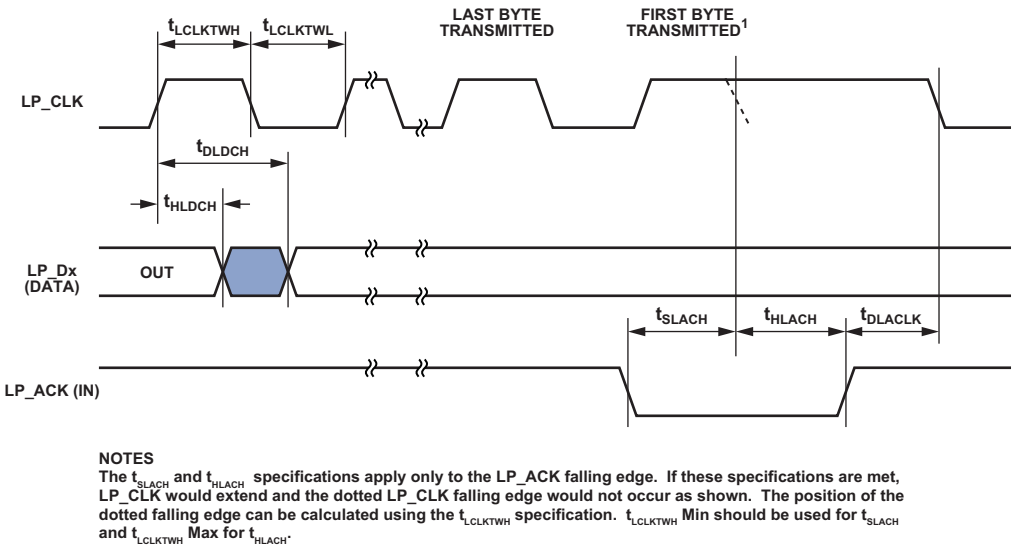



*Figure 34. Link Ports—Receive*

**Table 45. Link Ports—Transmit**

| 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* | | | | | | |
| $t_{SLACH}$ | LP_ACK Setup Before LP_CLK Low | $2 \times t_{SCLK0} + 17.5$ | | $2 \times t_{SCLK0} + 13.5$ | | ns |
| $t_{HLACH}$ | LP_ACK Hold After LP_CLK Low | 0 | | 0 | | ns |
| *Switching Characteristics* | | | | | | |
| $t_{DLDCH}$ | Data Delay After LP_CLK High | | 2.5 | | 2.5 | ns |
| $t_{HLDCH}$ | Data Hold After LP_CLK High | –1.5 | | –1.5 | | ns |
| $t_{LCLKTWL}$[1] | LP_CLK Width Low | $0.4 \times t_{LCLKTPROG}$ | $0.6 \times t_{LCLKTPROG}$ | $0.4 \times t_{LCLKTPROG}$ | $0.6 \times t_{LCLKTPROG}$ | ns |
| $t_{LCLKTWH}$[1] | LP_CLK Width High | $0.4 \times t_{LCLKTPROG}$ | $0.6 \times t_{LCLKTPROG}$ | $0.4 \times t_{LCLKTPROG}$ | $0.6 \times t_{LCLKTPROG}$ | ns |
| $t_{LCLKTW}$[1] | LP_CLK Period | $t_{LCLKTPROG} - 1.2$ | | $t_{LCLKTPROG} - 1.2$ | | ns |
| $t_{DLACLK}$ | LP_CLK Low Delay After LP_ACK High | $t_{SCLK0} + 4$ | $(2 \times t_{SCLK0}) + t_{LCLK} + 10$ | $t_{SCLK0} + 4$ | $(2 \times t_{SCLK0}) + t_{LCLK} + 10$ | ns |

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




NOTES
The $t_{SLACH}$ and $t_{HLACH}$ specifications apply only to the LP_ACK falling edge. If these specifications are met, LP_CLK would extend and the dotted LP_CLK falling edge would not occur as shown. The position of the dotted falling edge can be calculated using the $t_{LCLKTWH}$ specification. $t_{LCLKTWH}$ Min should be used for $t_{SLACH}$ and $t_{LCLKTWH}$ Max for $t_{HLACH}$.

*Figure 35. Link Ports—Transmit*

### 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.

### *Serial Peripheral Interface (SPI) Port—SPI_RDY Slave Timing*

**Table 53. SPI Port—SPI_RDY Slave Timing**

| Parameter | | $V_{DD\_EXT}$ 1.8 V/3.3 V Nominal Min | Max | Unit |
|---|---|---|---|---|
| *Switching Characteristics* | | | | |
| $t_{DSPISCKRDYSR}$ | SPI_RDY De-assertion from Valid Input SPI_CLK Edge in Slave Mode Receive | $2.5 \times t_{SCLK1}$ | $3.5 \times t_{SCLK1} + 17.5$ | ns |
| $t_{DSPISCKRDYST}$ | SPI_RDY De-assertion from Valid Input SPI_CLK Edge in Slave Mode Transmit | $3.5 \times t_{SCLK1}$ | $4.5 \times t_{SCLK1} + 17.5$ | ns |

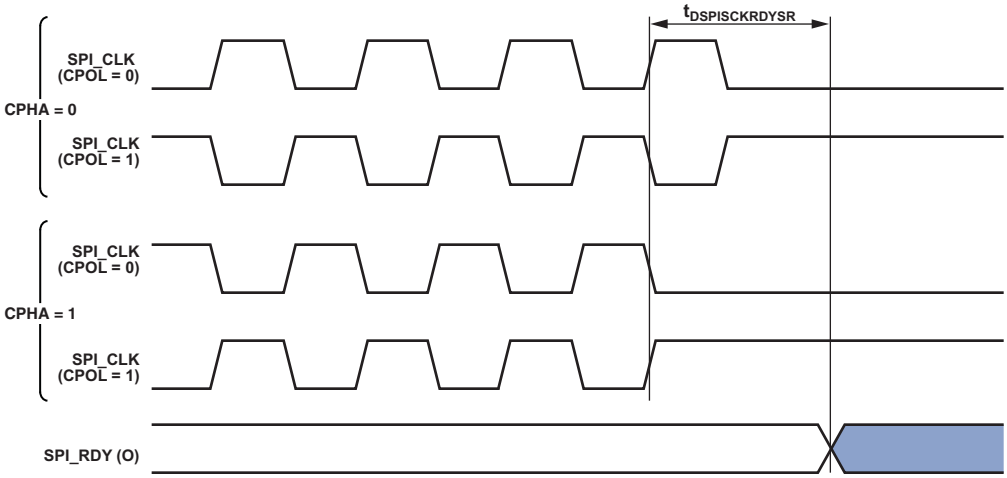



*Figure 42. SPI_RDY De-assertion from Valid Input SPI_CLK Edge in Slave Mode Receive (FCCH = 0)*

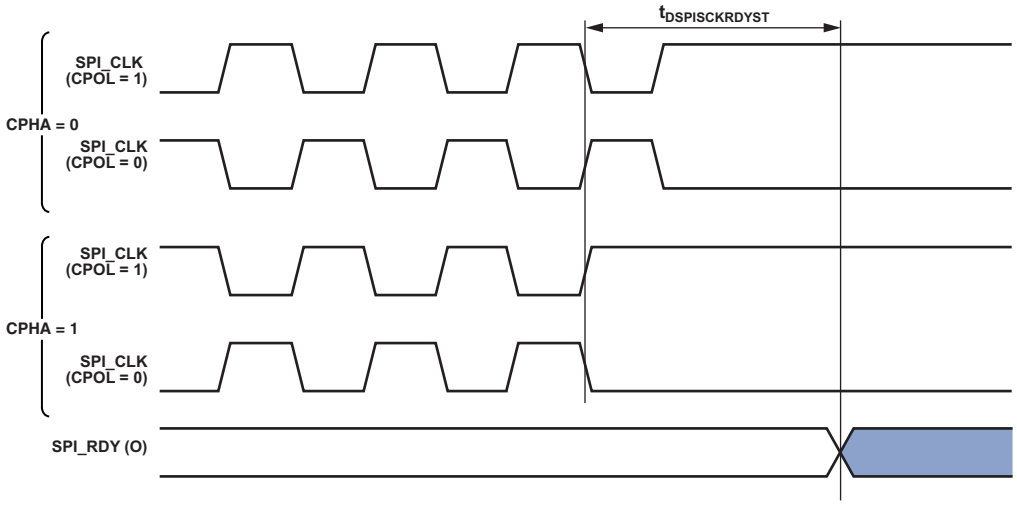



*Figure 43. SPI_RDY De-assertion from Valid Input SPI_CLK Edge in Slave Mode Transmit (FCCH = 1)*

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

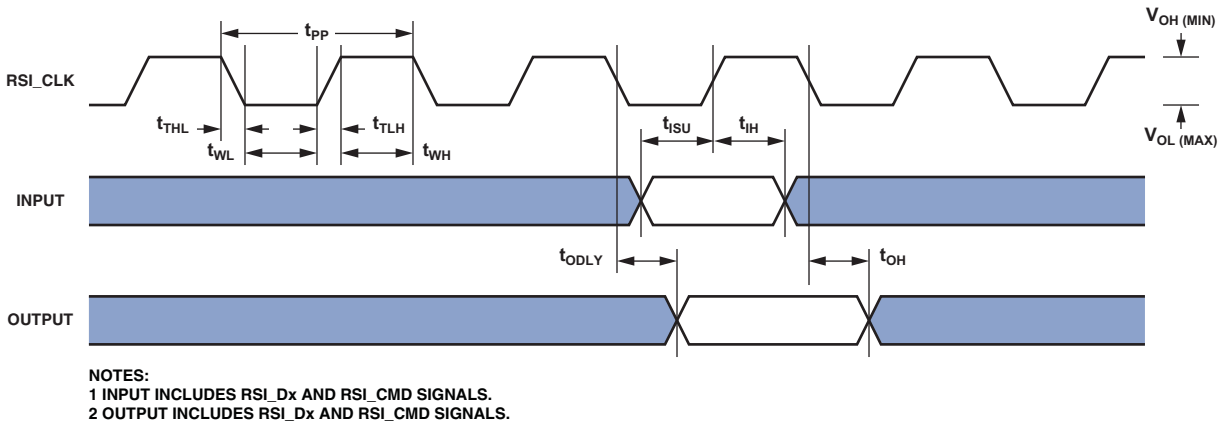



*Figure 54. RSI Controller Timing*

## 349-BALL CSP_BGA BALL CONFIGURATION

Figure 71 shows an overview of signal placement on the 349-ball CSP_BGA package.

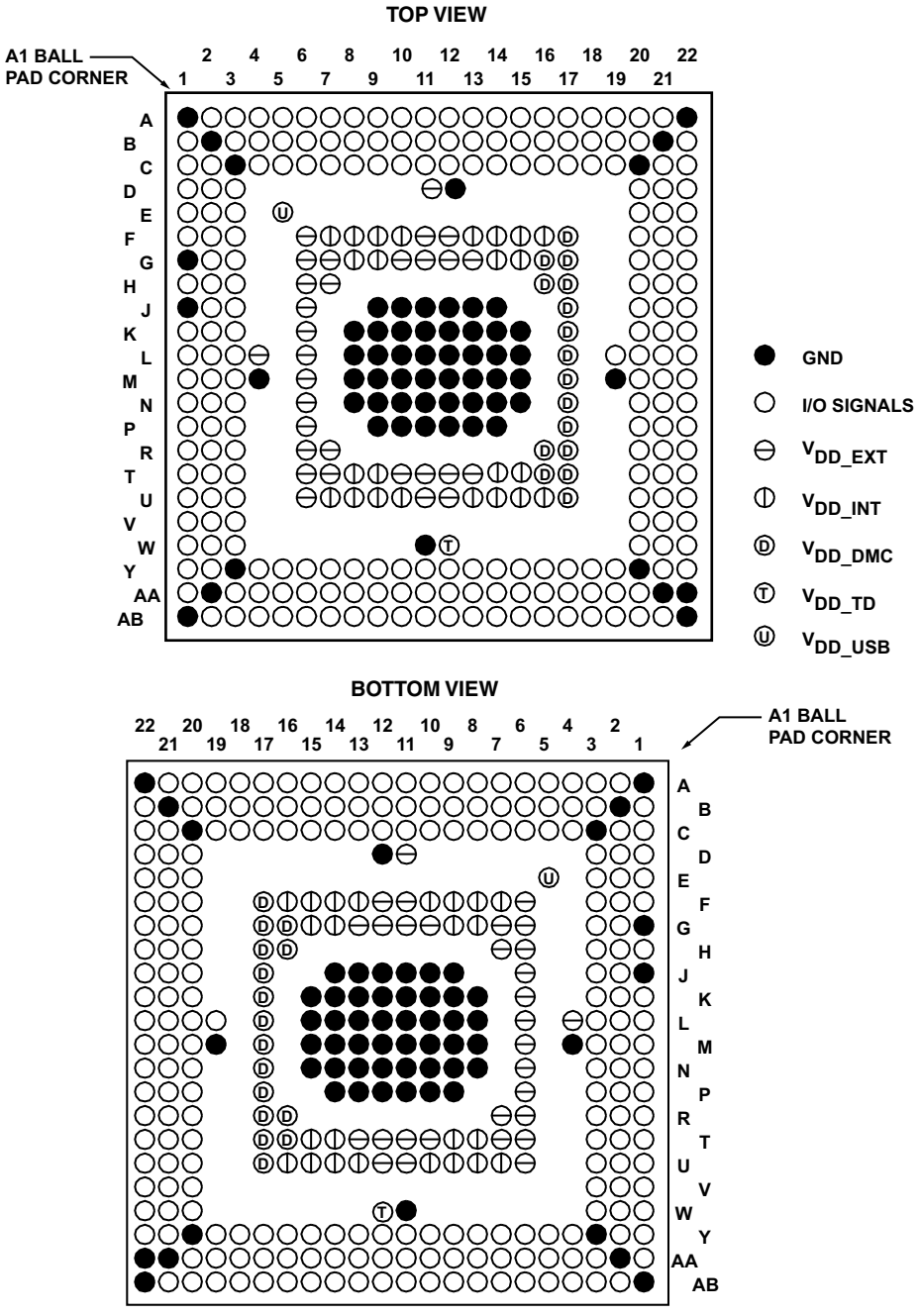



*Figure 71. 349-Ball CSP_BGA Ball Configuration*