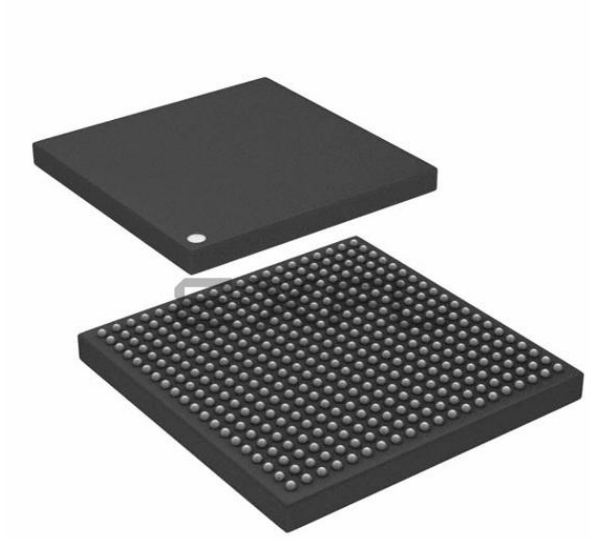


Welcome to E-XFL.COM

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	Obsolete
Type	Fixed Point
Interface	CAN, SPI, SSP, TWI, UART
Clock Rate	533MHz
Non-Volatile Memory	External
On-Chip RAM	196kB
Voltage - I/O	2.50V, 3.30V
Voltage - Core	1.25V
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	400-LFBGA, CSPBGA
Supplier Device Package	400-CSPBGA (17x17)
Purchase URL	https://www.e-xfl.com/product-detail/analog-devices/adbf544wbbcz504

TABLE OF CONTENTS

REVISION HISTORY

03/14—Rev. D to Rev. E

Updated Development Tools 21

Corrected SPI2 pin count in Port B configuration in Pin Multiplexing 24

Corrected typographical error of parameter name in External DMA Request Timing 58

Added note to Table 42 in Serial Ports—Enable and Three-State 63

Corrected t_{WL} and t_{WH} minimum specifications from t_{SCLK} +1 to $1 \times t_{SCLK}$ in Timer Cycle Timing 69

Added/changed package dimensions to Figure 88 in Outline Dimensions 100

Added low Alpha Package model to Ordering Guide 101

- SIC interrupt mask registers (SIC_IMASKx). These registers control the masking and unmasking of each peripheral interrupt event. When a bit is set in a register, that peripheral event is unmasked and is processed by the system when asserted. A cleared bit in the register masks the peripheral event, preventing the processor from servicing the event.
- SIC interrupt status registers (SIC_ISRx). As multiple peripherals can be mapped to a single event, these registers allow the software to determine which peripheral event source triggered the interrupt. A set bit indicates the peripheral is asserting the interrupt, and a cleared bit indicates the peripheral is not asserting the event.
- SIC interrupt wakeup enable registers (SIC_IWRx). By enabling the corresponding bit in this register, a peripheral can be configured to wake up the processor, should the core be idled or in Sleep mode when the event is generated. (For more information, see Dynamic Power Management on Page 15.)

Because multiple interrupt sources can map to a single general-purpose interrupt, multiple pulse assertions can occur simultaneously, before or during interrupt processing for an interrupt event already detected on this interrupt input. The IPEND register contents are monitored by the SIC as the interrupt acknowledgement.

The appropriate ILAT register bit is set when an interrupt rising edge is detected. (Detection requires two core clock cycles.) The bit is cleared when the respective IPEND register bit is set. The IPEND bit indicates that the event has entered into the processor pipeline. At this point the CEC recognizes and queues the next rising edge event on the corresponding event input. The minimum latency from the rising edge transition of the general-purpose interrupt to the IPEND output asserted is three core clock cycles; however, the latency can be much higher, depending on the activity within and the state of the processor.

DMA CONTROLLERS

ADSP-BF54x Blackfin processors have multiple, independent DMA channels that support automated data transfers with minimal overhead for the processor core. DMA transfers can occur between the ADSP-BF54x processors' internal memories and any of the DMA-capable peripherals. Additionally, DMA transfers can be accomplished between any of the DMA-capable peripherals and external devices connected to the external memory interfaces, including DDR and asynchronous memory controllers.

While the USB controller and MXVR have their own dedicated DMA controllers, the other on-chip peripherals are managed by two centralized DMA controllers, called DMAC1 (32-bit) and DMAC0 (16-bit). Both operate in the SCLK domain. Each DMA controller manages 12 independent peripheral DMA channels, as well as two independent memory DMA streams. The DMAC1 controller masters high-bandwidth peripherals over a dedicated 32-bit DMA access bus (DAB32). Similarly, the DMAC0 controller masters most serial interfaces over the 16-bit DAB16 bus. Individual DMA channels have fixed access priority on the DAB buses. DMA priority of peripherals is managed by a flexible peripheral-to-DMA channel assignment scheme.

All four DMA controllers use the same 32-bit DCB bus to exchange data with L1 memory. This includes L1 ROM, but excludes scratchpad memory. Fine granulation of L1 memory and special DMA buffers minimize potential memory conflicts when the L1 memory is accessed simultaneously by the core. Similarly, there are dedicated DMA buses between the external bus interface unit (EBIU) and the three DMA controllers (DMAC1, DMAC0, and USB) that arbitrate DMA accesses to external memories and the boot ROM.

The ADSP-BF54x Blackfin processors' DMA controllers support both 1-dimensional (1D) and 2-dimensional (2D) DMA transfers. DMA transfer initialization can be implemented from registers or from sets of parameters called descriptor blocks.

The 2D DMA capability supports arbitrary row and column sizes up to 64K elements by 64K elements, and arbitrary row and column step sizes up to ±32K elements. Furthermore, the column step size can be less than the row step size, allowing implementation of interleaved data streams. This feature is especially useful in video applications where data can be de-interleaved on the fly.

Examples of DMA types supported by the ADSP-BF54x Blackfin processors' DMA controllers include:

- A single, linear buffer that stops upon completion
- A circular, auto-refreshing buffer that interrupts on each full or fractionally full buffer
- 1D or 2D DMA using a linked list of descriptors
- 2D DMA using an array of descriptors, specifying only the base DMA address within a common page

In addition to the dedicated peripheral DMA channels, the DMAC1 and DMAC0 controllers each feature two memory DMA channel pairs for transfers between the various memories of the ADSP-BF54x Blackfin processors. This enables transfers of blocks of data between any of the memories—including external DDR, ROM, SRAM, and flash memory—with minimal processor intervention. Like peripheral DMAs, memory DMA transfers can be controlled by a very flexible descriptor-based methodology or by a standard register-based autobuffer mechanism.

The memory DMA channels of the DMAC1 controller (MDMA2 and MDMA3) can be controlled optionally by the external DMA request input pins. When used in conjunction with the External Bus Interface Unit (EBIU), this handshaked memory DMA (HMDMA) scheme can be used to efficiently exchange data with block-buffered or FIFO-style devices connected externally. Users can select whether the DMA request pins control the source or the destination side of the memory DMA. It allows control of the number of data transfers for memory DMA. The number of transfers per edge is programmable. This feature can be programmed to allow memory DMA to have an increased priority on the external bus relative to the core.

For large page NAND flash devices, the 4-byte electronic signature is read in order to configure the kernel for booting. This allows support for multiple large page devices. The fourth byte of the electronic signature must comply with the specifications in Table 9.

Any configuration from Table 9 that also complies with the command set listed below is directly supported by the boot kernel. There are no restrictions on the page size or block size as imposed by the small-page boot kernel.

Table 9. Byte 4 Electronic Signature Specification

Page Size (excluding spare area)	D1:D0	00	1K bytes
		01	2K bytes
		10	4K bytes
		11	8K bytes
Spare Area Size	D2	0	8 bytes/512 bytes
		1	16 bytes/512 bytes
Block Size (excluding spare area)	D5:4	00	64K bytes
		01	128K bytes
		10	256K bytes
		11	512K bytes
Bus Width	D6	0	x8
		1	x16
Not Used for Configuration	D3, D7		

Large page devices must support the following command set:

Reset: 0xFF
Read Electronic Signature: 0x90
Read: 0x00, 0x30 (confirm command)

Large page devices must not support or react to NAND flash command 0x50. This is a small page NAND flash command used for device auto detection.

By default, the boot kernel will always issue five address cycles; therefore, if a large page device requires only four cycles, the device must be capable of ignoring the additional address cycle.

16-bit NAND flash memory devices must only support the issuing of command and address cycles via the lower eight bits of the data bus. Devices that use the full 16-bit bus for command and address cycles are not supported.

- Boot from OTP memory (BMODE = 0xB)—This provides a standalone booting method. The boot stream is loaded from on-chip OTP memory. By default, the boot stream is expected to start from OTP page 0x40 and can occupy all public OTP memory up to page 0xDF (2560 bytes). Since the start page is programmable, the maximum size of the boot stream can be extended to 3072 bytes.
- Boot from 16-bit host DMA (BMODE = 0xE)—In this mode, the host DMA port is configured in 16-bit acknowledge mode with little endian data format. Unlike other modes, the host is responsible for interpreting the boot stream. It writes data blocks individually into the host DMA port. Before configuring the DMA settings for each block, the host may either poll the ALLOW_CONFIG bit in HOST_STATUS or wait to be interrupted by the HWAIT signal. When using HWAIT, the host must still check ALLOW_CONFIG at least once before beginning to configure the host DMA port. After completing the configuration, the host is required to poll the READY bit in HOST_STATUS before beginning to transfer data. When the host sends an HIRQ control command, the boot kernel issues a CALL instruction to address 0xFFA0 0000. It is the host's responsibility to ensure valid code has been placed at this address. The routine at address 0xFFA0 0000 can be a simple initialization routine to configure internal resources, such as the SDRAM controller, which then returns using an RTS instruction. The routine may also be the final application, which will never return to the boot kernel.
- Boot from 8-bit host DMA (BMODE = 0xF)—In this mode, the host DMA port is configured in 8-bit interrupt mode with little endian data format. Unlike other modes, the host is responsible for interpreting the boot stream. It writes data blocks individually to the host DMA port. Before configuring the DMA settings for each block, the host may either poll the ALLOW_CONFIG bit in HOST_STATUS or wait to be interrupted by the HWAIT signal. When using HWAIT, the host must still check ALLOW_CONFIG at least once before beginning to configure the host DMA port. The host will receive an interrupt from the HOST_ACK signal every time it is allowed to send the next FIFO depth's worth (sixteen 32-bit words) of information. When the host sends an HIRQ control command, the boot kernel issues a CALL instruction to address 0xFFA0 0000. It is the host's responsibility to ensure valid code has been placed at this address. The routine at address 0xFFA0 0000 can be a simple initialization routine to configure internal resources, such as the SDRAM controller, which then returns using an RTS instruction. The routine may also be the final application, which will never return to the boot kernel.

For each of the boot modes, a 16-byte header is first read from an external memory device. The header specifies the number of bytes to be transferred and the memory destination address. Multiple memory blocks may be loaded by any boot sequence. Once all blocks are loaded, program execution commences from the address stored in the EVT1 register.

Prior to booting, the pre-boot routine interrogates the OTP memory. Individual boot modes can be customized or disabled based on OTP programming. External hardware, especially booting hosts, may monitor the HWAIT signal to determine

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” or “SHARC 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 Engineer-to-Engineer Note *“Analog Devices JTAG Emulation Technical Reference”* (EE-68) 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.

MXVR BOARD LAYOUT GUIDELINES

The MXVR Loop Filter RC network is connected between the MLF_P and MLF_M pins in the following manner:

Capacitors:

- C1: 0.047 µF (PPS type, 2% tolerance recommended)
- C2: 330 pF (PPS type, 2% tolerance recommended)

Resistor:

- R1: 330 Ω (1% tolerance)

The RC network should be located physically close to the MLF_P and MLF_M pins on the board.

The RC network should be shielded using GND_{MP} traces.

Avoid routing other switching signals near the RC network to avoid crosstalk.

MXI driven with external clock oscillator IC:

- MXI should be driven with the clock output of a clock oscillator IC running at a frequency of 49.152 MHz or 45.1584 MHz.
- MXO should be left unconnected.
- Avoid routing other switching signals near the oscillator and clock output trace to avoid crosstalk. When not possible, shield traces with ground.

MXI/MXO with external crystal:

- The crystal must be a fundamental mode crystal running at a frequency of 49.152 MHz or 45.1584 MHz.
- The crystal and load capacitors should be placed physically close to the MXI and MXO pins on the board.
- Board trace capacitance on each lead should not be more than 3 pF.
- Trace capacitance plus load capacitance should equal the load capacitance specification for the crystal.
- Avoid routing other switching signals near the crystal and components to avoid crosstalk. When not possible, shield traces and components with ground.

V_{DDMP}/GND_{MP}—MXVR PLL power domain:

- Route V_{DDMP} and GND_{MP} with wide traces or as isolated power planes.
- Drive V_{DDMP} to same level as V_{DDINT}.
- Place a ferrite bead between the V_{DDINT} power plane and the V_{DDMP} pin for noise isolation.
- Locally bypass V_{DDMP} with 0.1 µF and 0.01 µF decoupling capacitors to GND_{MP}.
- Avoid routing switching signals near to V_{DDMP} and GND_{MP} traces to avoid crosstalk.

Table 11. Pin Descriptions (Continued)

Pin Name	I/O[1]	Function (First/Second/Third/Fourth)	Driver Type[2]
Port E: GPIO/*SPI0/UART0-1/PPI1/TWI0/Keypad*			
PE0/*SPI0SCK*/*KEY_COL7*[3]	I/O	GPIO/*SPI0 Clock*/*Keypad Column Output*	A
PE1/*SPI0MISO*/*KEY_ROW6*[3]	I/O	GPIO/*SPI0 Master In Slave Out*/*Keypad Row Input*	C
PE2/*SPI0MOSI*/*KEY_COL6*	I/O	GPIO/*SPI0 Master Out Slave In*/*Keypad Column Output*	C
PE3/$\overline{SPI0SS}$/*KEY_ROW5*	I/O	GPIO/*SPI0 Slave Select Input*/*Keypad Row Input*	A
PE4/$\overline{SPI0SEL1}$/*KEY_COL*[3]	I/O	GPIO/*SPI0 Slave Select Enable 1*/*Keypad Column Output*	A
PE5/$\overline{SPI0SEL2}$/*KEY_ROW4*	I/O	GPIO/*SPI0 Slave Select Enable 2*/*Keypad Row Input*	A
PE6/$\overline{SPI0SEL3}$/*KEY_COL4*	I/O	GPIO/*SPI0 Slave Select Enable 3*/*Keypad Column Output*	A
PE7/*UART0TX*/*KEY_ROW7*	I/O	GPIO/*UART0 Transmit*/*Keypad Row Input*	A
PE8/*UART0RX*/*TACI0*	I/O	GPIO/*UART0 Receive*/*Alternate Capture Input 0*	A
PE9/$\overline{UART1RTS}$	I/O	GPIO/*UART1 Request to Send*	A
PE10/$\overline{UART1CTS}$	I/O	GPIO/*UART1 Clear to Send*	A
PE11/*PPI1_CLK*	I/O	GPIO/*PPI1 Clock*	A
PE12/*PPI1_FS1*	I/O	GPIO/*PPI1 Frame Sync 1*	A
PE13/*PPI1_FS2*	I/O	GPIO/*PPI1 Frame Sync 2*	A
PE14/*SCL0*	I/O	GPIO/*TWI0 Serial Clock* (Open-drain output: requires a pull-up resistor.)	E
PE15/*SDA0*	I/O	GPIO/*TWI0 Serial Data* (Open-drain output: requires a pull-up resistor.)	E
Port F: *GPIO/PPI0/Alternate ATAPI Data*			
PF0/*PPI0_D0/ATAPI_D0A*	I/O	GPIO/*PPI0 Data/Alternate ATAPI Data*	A
PF1/*PPI0_D1/ATAPI_D1A*	I/O	GPIO/*PPI0 Data/Alternate ATAPI Data*	A
PF2/*PPI0_D2/ATAPI_D2A*	I/O	GPIO/*PPI0 Data/Alternate ATAPI Data*	A
PF3/*PPI0_D3/ATAPI_D3A*	I/O	GPIO/*PPI0 Data/Alternate ATAPI Data*	A
PF4/*PPI0_D4/ATAPI_D4A*	I/O	GPIO/*PPI0 Data/Alternate ATAPI Data*	A
PF5/*PPI0_D5/ATAPI_D5A*	I/O	GPIO/*PPI0 Data/Alternate ATAPI Data*	A
PF6/*PPI0_D6/ATAPI_D6A*	I/O	GPIO/*PPI0 Data/Alternate ATAPI Data*	A
PF7/*PPI0_D7/ATAPI_D7A*	I/O	GPIO/*PPI0 Data/Alternate ATAPI Data*	A
PF8/*PPI0_D8/ATAPI_D8A*	I/O	GPIO/*PPI0 Data/Alternate ATAPI Data*	A
PF9/*PPI0_D9/ATAPI_D9A*	I/O	GPIO/*PPI0 Data/Alternate ATAPI Data*	A
PF10/*PPI0_D10/ATAPI_D10A*	I/O	GPIO/*PPI0 Data/Alternate ATAPI Data*	A
PF11/*PPI0_D11/ATAPI_D11A*	I/O	GPIO/*PPI0 Data/Alternate ATAPI Data*	A
PF12/*PPI0_D12/ATAPI_D12A*	I/O	GPIO/*PPI0 Data/Alternate ATAPI Data*	A
PF13/*PPI0_D13/ATAPI_D13A*	I/O	GPIO/*PPI0 Data/Alternate ATAPI Data*	A
PF14/*PPI0_D14/ATAPI_D14A*	I/O	GPIO/*PPI0 Data/Alternate ATAPI Data*	A
PF15/*PPI0_D15/ATAPI_D15A*	I/O	GPIO/*PPI0 Data/Alternate ATAPI Data*	A

Table 11. Pin Descriptions (Continued)

Pin Name	I/O[1]	Function (First/Second/Third/Fourth)	Driver Type[2]
Port G: GPIO/*PPI0/SPI1/PPI2/Up-Down Counter/CAN0–1/Host DMA/MXVR (MOST)/ATAPI*			
PG0/*PPI0_CLK/TMRCLK*	I/O	GPIO/*PPI0 Clock/External Timer Reference*	A
PG1/*PPI0_FS1*	I/O	GPIO/*PPI0 Frame Sync 1*	A
PG2/*PPI0_FS2/ATAPI_A0A*	I/O	GPIO/*PPI0 Frame Sync 2/Alternate ATAPI Address*	A
PG3/*PPI0_D16/ATAPI_A1A*	I/O	GPIO/*PPI0 Data/Alternate ATAPI Address*	A
PG4/*PPI0_D17/ATAPI_A2A*	I/O	GPIO/*PPI0 Data/Alternate ATAPI Address*	A
PG5/$\overline{SPI1SEL1}$/$\overline{HOST\_CE}$/*PPI2_FS2/CZM*	I/O	GPIO/*SPI1 Slave Select/Host DMA Chip Enable/PPI2 Frame Sync 2/Counter Zero Marker*	A
PG6/$\overline{SPI1SEL2}$/$\overline{HOST\_RD}$/*PPI2_FS1*	I/O	GPIO/*SPI1 Slave Select/ Host DMA Read/PPI2 Frame Sync* 1	A
PG7/$\overline{SPI1SEL3}$/$\overline{HOST\_WR}$/*PPI2_CLK*	I/O	GPIO/*SPI1 Slave Select/Host DMA Write/PPI2 Clock*	A
PG8/*SPI1SCK*	I/O	GPIO/*SPI1 Clock*	C
PG9/*SPI1MISO*	I/O	GPIO/*SPI1 Master In Slave Out*	C
PG10/*SPI1MOSI*	I/O	GPIO/*SPI1 Master Out Slave In*	C
PG11/$\overline{SPI1SS}$/$\overline{MTXON}$	I/O	GPIO/*SPI1 Slave Select Input/MXVR Transmit Phy On*	A
PG12/*CAN0TX*	I/O	GPIO/*CAN0 Transmit*	A
PG13/*CAN0RX/TACI4*	I/O	GPIO/*CAN0 Receive/Alternate Capture Input 4*	A
PG14/*CAN1TX*	I/O	GPIO/*CAN1 Transmit*	A
PG15/*CAN1RX/TACI5*	I/O	GPIO/*CAN1 Receive/Alternate Capture Input 5*	A
Port H: GPIO/*AMC/EXTDMA/UART1/PPI0–2/ATAPI/Up-Down Counter/TMR8-10/Host DMA/MXVR (MOST)*			
PH0/*UART1TX/PPI1_FS3_DEN*	I/O	GPIO/*UART1 Transmit/PPI1 Frame Sync* 3	A
PH1/*UART1RX/PPI0_FS3_DEN/TACI1*	I/O	GPIO/*UART 1 Receive/ PPI0 Frame Sync* 3/*Alternate Capture Input 1*	A
PH2/$\overline{ATAPI\_RESET}$/*TMR8/PPI2_FS3_DEN*	I/O	GPIO/*ATAPI Interface Hard Reset Signal/Timer 8/PPI2 Frame Sync* 3	A
PH3/*HOST_ADDR/TMR9/CDG*	I/O	GPIO/*HOST Address/Timer 9/Count Down and Gate*	A
PH4/*HOST_ACK/TMR10/CUD*	I/O	GPIO/*HOST Acknowledge/Timer 10/Count Up and Direction*	A
PH5/*MTX/DMAR0/TACI8 and TACLK8*	I/O	GPIO/*MXVR Transmit Data/Ext. DMA Request/Alt Capt. In. 8/Alt In. Clk 8*	C
PH6/*MRX/DMAR1/TACI9 and TACLK9*	I/O	GPIO/*MXVR Receive Data/Ext. DMA Request/Alt Capt. In. 9/Alt In. Clk 9*	A
PH7/$\overline{MRXON}$/$\overline{GPW}$/*TACI10 and TACLK10/HWAITA*[4,5]	I/O	GPIO/*MXVR Receive Phy On/Alt Capt. In. 10/Alt In. Clk 10/Alternate Boot Host Wait*	A
PH8/*A4*[6]	I/O	GPIO/*Address Bus for Async Access*	A
PH9/*A5*[6]	I/O	GPIO/*Address Bus for Async Access*	A
PH10/*A6*[6]	I/O	GPIO/*Address Bus for Async Access*	A
PH11/*A7*[6]	I/O	GPIO/*Address Bus for Async Access*	A
PH12/*A8*[6]	I/O	GPIO/*Address Bus for Async Access*	A
PH13/*A9*[6]	I/O	GPIO/*Address Bus for Async Access*	A

ELECTRICAL CHARACTERISTICS

Parameter		Test Conditions	Nonautomotive 400 MHz[1] Min	Typ	Max	All Other Devices[2] Min	Typ	Max	Unit
V_{OH}	High Level Output Voltage for 3.3 V I/O[3]	V_{DDEXT} = 2.7 V, I_{OH} = –0.5 mA	2.4			2.4			V
	High Level Output Voltage for 2.5 V I/O[3]	V_{DDEXT} = 2.25 V, I_{OH} = –0.5 mA	2.0			2.0			V
V_{OHDDR}	High Level Output Voltage for DDR SDRAM[4]	V_{DDDDR} = 2.5 V, I_{OH} = –8.1 mA	1.74			1.74			V
	High Level Output Voltage for Mobile DDR SDRAM[4]	V_{DDDDR} = 1.8 V, I_{OH} = –0.1 mA	1.62			1.62			V
V_{OL}	Low Level Output Voltage for 3.3 V I/O[3]	V_{DDEXT} = 2.7 V, I_{OL} = 2.0 mA			0.4			0.4	V
	Low Level Output Voltage for 2.5 V I/O[3]	V_{DDEXT} = 2.25 V, I_{OL} = 2.0 mA			0.4			0.4	V
V_{OLDDR}	Low Level Output Voltage for DDR SDRAM[4]	V_{DDDDR} = 2.5 V, I_{OL} = 8.1 mA			0.56			0.56	V
	Low Level Output Voltage for Mobile DDR SDRAM[4]	V_{DDDDR} = 1.8 V, I_{OL} = 0.1 mA			0.18			0.18	V
I_{IH}	High Level Input Current[5]	V_{DDEXT} = 3.6 V, V_{IN} = V_{IN} Max			10.0			10.0	µA
I_{IHP}	High Level Input Current[6]	V_{DDEXT} = 3.6 V, V_{IN} = V_{IN} Max			50.0			50.0	µA
I_{IHDDR_VREF}	High Level Input Current for DDR SDRAM[7]	V_{DDDDR} = 2.7 V, V_{IN} = 0.51 × V_{DDDDR}			30.0			30.0	µA
	High Level Input Current for Mobile DDR SDRAM[7]	V_{DDDDR} = 1.95 V, V_{IN} = 0.51 × V_{DDDDR}			30.0			30.0	µA
I_{IL}[8]	Low Level Input Current	V_{DDEXT} = 3.6 V, V_{IN} = 0 V			10.0			10.0	µA
I_{OZH}[9]	Three-State Leakage Current[10]	V_{DDEXT} = 3.6 V, V_{IN} = V_{IN} Max			10.0			10.0	µA
I_{OZL}[11]	Three-State Leakage Current[10]	V_{DDEXT} = 3.6 V, V_{IN} = 0 V			10.0			10.0	µA
C_{IN}	Input Capacitance[12]	f_{IN} = 1 MHz, $T_{AMBIENT}$ = 25°C, V_{IN} = 2.5 V		4[12]	8[12]		4[12]	8[12]	pF
$I_{DDDEEPSLEEP}$[13]	V_{DDINT} Current in Deep Sleep Mode	V_{DDINT} = 1.0 V, f_{CCLK} = 0 MHz, f_{SCLK} = 0 MHz, T_J = 25°C, ASF = 0.00		22			37		mA
$I_{DDSLEEP}$	V_{DDINT} Current in Sleep Mode	V_{DDINT} = 1.0 V, f_{SCLK} = 25 MHz, T_J = 25°C		35			50		mA
$I_{DD-IDLE}$	V_{DDINT} Current in Idle	V_{DDINT} = 1.0 V, f_{CCLK} = 50 MHz, T_J = 25°C, ASF = 0.47		44			59		mA

Table 28. Asynchronous Memory Read Cycle Timing with Asynchronous ARDY

Parameter		Min	Max	Unit
Timing Requirements				
t_{SDAT}	DATA15–0 Setup Before CLKOUT	5.0		ns
t_{HDAT}	DATA15–0 Hold After CLKOUT	0.8		ns
t_{DANR}	ARDY Negated Delay from $\overline{AMSx}$ Asserted[1]		$(S + RA - 2) \times t_{SCLK}$	ns
t_{HAA}	ARDY Asserted Hold After $\overline{ARE}$ Negated	0.0		ns
Switching Characteristics				
t_{DO}	Output Delay After CLKOUT[2]		6.0	ns
t_{HO}	Output Hold After CLKOUT[2]	0.3		ns

[1] S = number of programmed setup cycles, RA = number of programmed read access cycles.
[2] Output pins include $\overline{AMS3-0}$, $\overline{ABE1-0}$, ADDR19–1, $\overline{AOE}$, and $\overline{ARE}$.

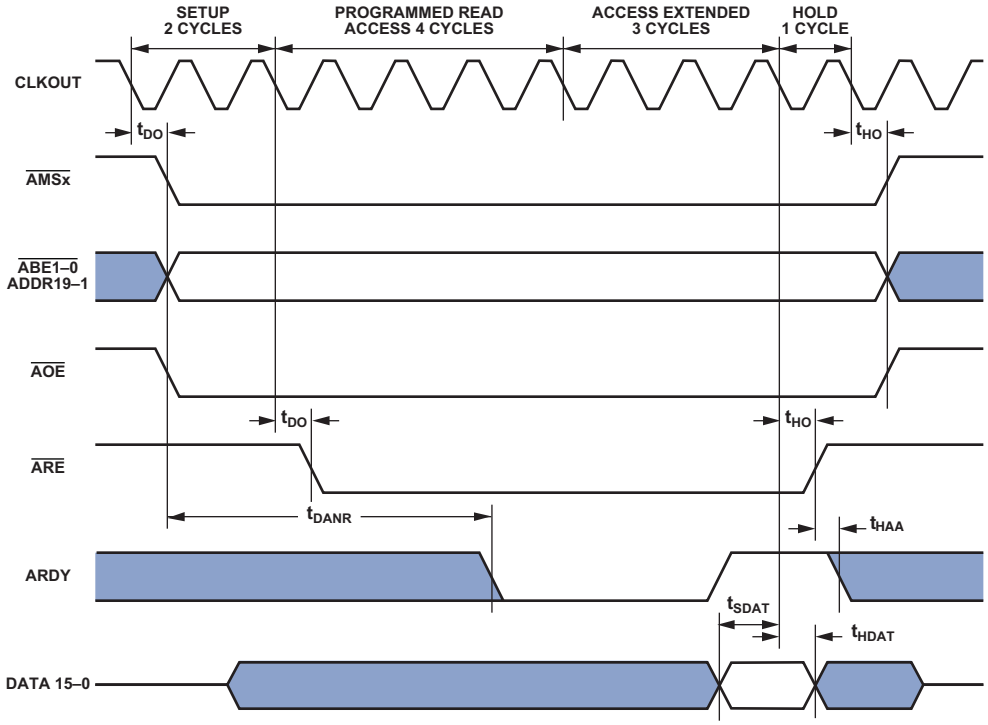


Figure 14. Asynchronous Memory Read Cycle Timing with Asynchronous ARDY

DDR SDRAM/Mobile DDR SDRAM Timing

Table 32 and Figure 18/Figure 19 describe DDR SDRAM/mobile DDR SDRAM read cycle timing.

Table 32. DDR SDRAM/Mobile DDR SDRAM Read Cycle Timing

Parameter		DDR SDRAM Min	DDR SDRAM Max	Mobile DDR SDRAM Min	Mobile DDR SDRAM Max	Unit
Timing Requirements						
t_{AC}	Access Window of DQ0-15 to DCK0-1	–1.25	+1.25	0.0	6.00	ns
t_{DQSCK}	Access Window of DQS0-1 to DCK0-1	–1.25	+1.25	0.0	6.00	ns
t_{DQSQ}	DQS0-1 to DQ0-15 Skew, DQS0-1 to Last DQ0-15 Valid		0.90		0.85	ns
t_{QH}	DQ0-15 to DQS0-1 Hold, DQS0-1 to First DQ0-15 to Go Invalid	$t_{CK}/2 - 1.25$[1] $t_{CK}/2 - 1.75$[2]		$t_{CK}/2 - 1.25$		ns
t_{RPRE}	DQS0-1 Read Preamble	0.9	1.1	0.9	1.1	t_{CK}
t_{RPST}	DQS0-1 Read Postamble	0.4	0.6	0.4	0.6	t_{CK}

[1] For 7.50 ns $\leq t_{CK} < 10$ ns.
[2] For $t_{CK} \geq 10$ ns.

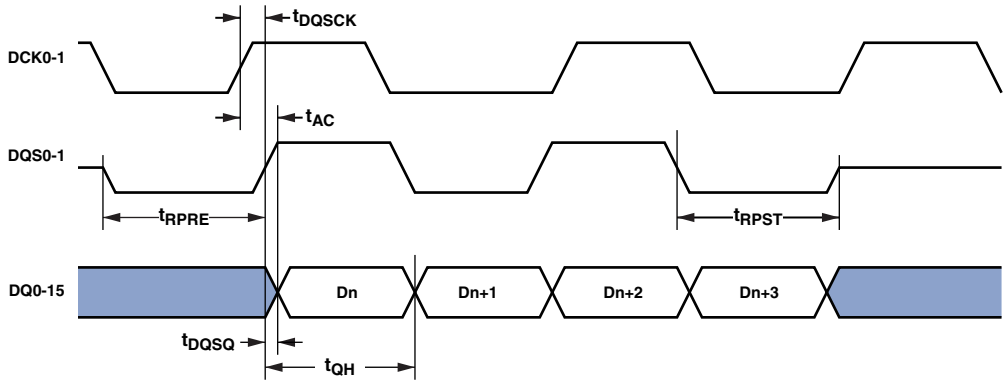


Figure 18. DDR SDRAM Controller Read Cycle Timing

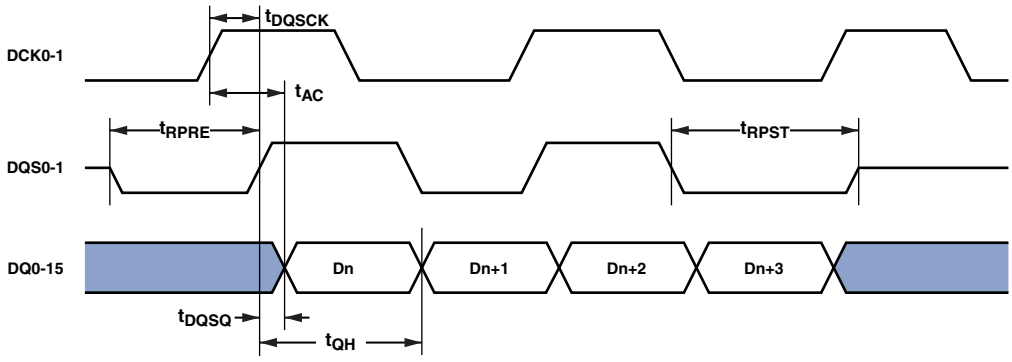


Figure 19. Mobile DDR SDRAM Controller Read Cycle Timing

Table 35. External Port Bus Request and Grant Cycle Timing with Asynchronous $\overline{BR}$

Parameter		Min	Max	Unit
Timing Requirement				
t_{WBR}	$\overline{BR}$ Pulsewidth	$2 \times t_{SCLK}$		ns
Switching Characteristics				
t_{SD}	CLKOUT Low to $\overline{AMSx}$, Address, and $\overline{ARE}/\overline{AWE}$ Disable		5.0	ns
t_{SE}	CLKOUT Low to $\overline{AMSx}$, Address, and $\overline{ARE}/\overline{AWE}$ Enable		5.0	ns
t_{DBG}	CLKOUT Low to $\overline{BG}$ Asserted Output Delay		4.0	ns
t_{EBG}	CLKOUT Low to $\overline{BG}$ Deasserted Output Hold		4.0	ns
t_{DBH}	CLKOUT Low to $\overline{BGH}$ Asserted Output Delay		3.6	ns
t_{EBH}	CLKOUT Low to $\overline{BGH}$ Deasserted Output Hold		3.6	ns

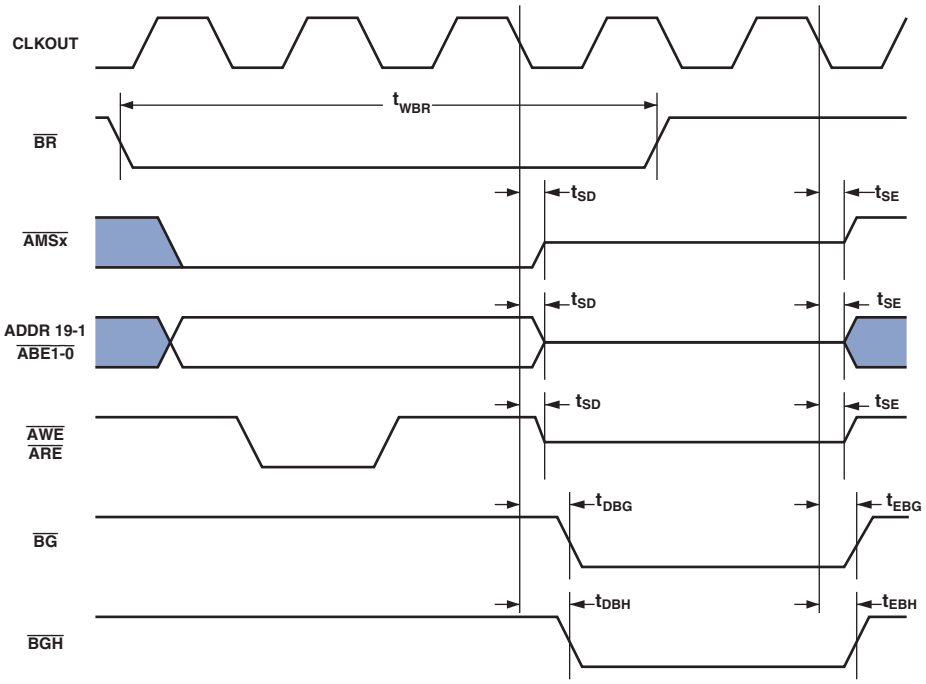


Figure 22. External Port Bus Request and Grant Cycle Timing with Asynchronous $\overline{BR}$

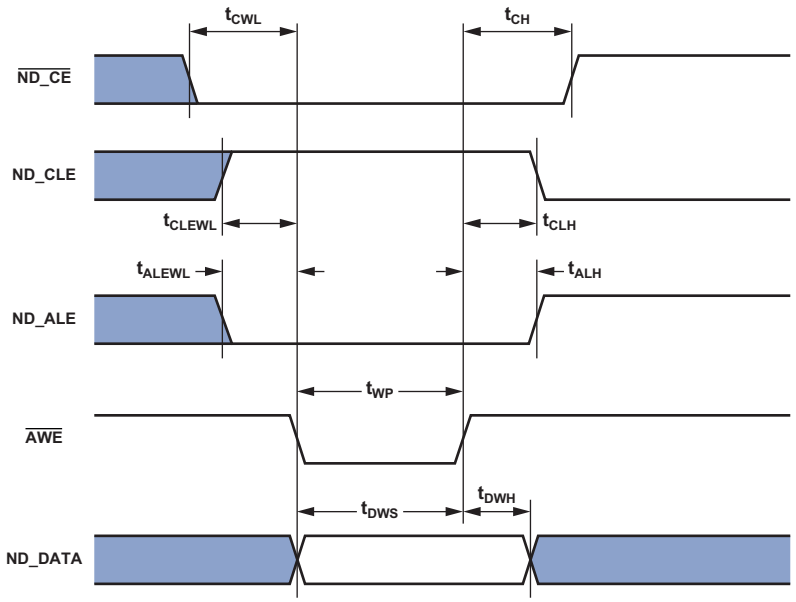


Figure 23. NAND Flash Controller Interface Timing—Command Write Cycle

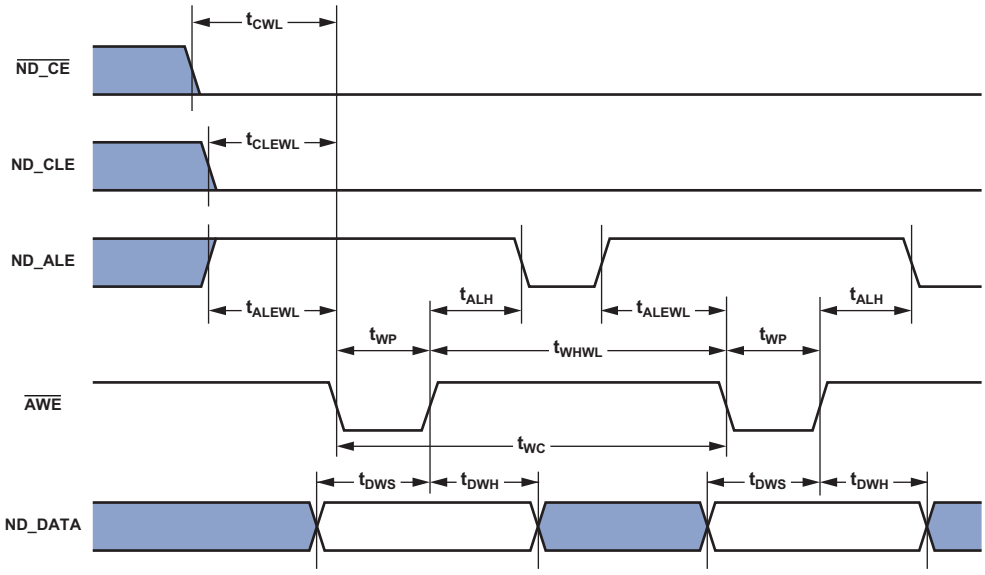


Figure 24. NAND Flash Controller Interface Timing—Address Write Cycle

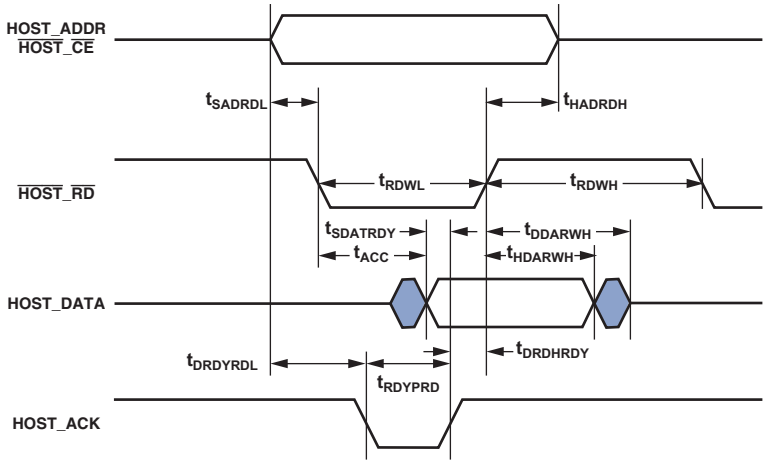
HOSTDP A/C Timing-Host Read Cycle

Table 54 and Figure 46 describe the HOSTDP A/C host read cycle timing requirements.

Table 54. Host Read Cycle Timing Requirements

Parameter		Min	Max	Unit
Timing Requirements				
t_{SADRDL}	HOST_ADDR and $\overline{\text{HOST\_CE}}$ Setup Before $\overline{\text{HOST\_RD}}$ Falling Edge	4		ns
t_{HADRDH}	HOST_ADDR and $\overline{\text{HOST\_CE}}$ Hold After $\overline{\text{HOST\_RD}}$ Rising Edge	2.5		ns
t_{RDWL}	$\overline{\text{HOST\_RD}}$ Pulse Width Low (ACK Mode)	$t_{DRDYRDL} + t_{RDYPRD} + t_{DRDHRDY}$		ns
	$\overline{\text{HOST\_RD}}$ Pulse Width Low (INT Mode)	$1.5 \times t_{SCLK} + 8.7$		ns
t_{RDWH}	$\overline{\text{HOST\_RD}}$ Pulse Width High or Time Between $\overline{\text{HOST\_RD}}$ Rising Edge and $\overline{\text{HOST\_WR}}$ Falling Edge	$2 \times t_{SCLK}$		ns
$t_{DRDHRDY}$	$\overline{\text{HOST\_RD}}$ Rising Edge Delay After HOST_ACK Rising Edge (ACK Mode)	0		ns
Switching Characteristics				
$t_{SDATRDY}$	HOST_D15–0 Valid Prior HOST_ACK Rising Edge (ACK Mode)	$t_{SCLK} - 4.0$		ns
$t_{DRDYRDL}$	HOST_ACK Falling Edge After $\overline{\text{HOST\_CE}}$ (ACK Mode)		11.25	ns
t_{RDYPRD}	HOST_ACK Low Pulse-Width for Read Access (ACK Mode)		NM[1]	ns
t_{DDARWH}	HOST_D15–0 Disable After $\overline{\text{HOST\_RD}}$		8.0	ns
t_{ACC}	HOST_D15–0 Valid After $\overline{\text{HOST\_RD}}$ Falling Edge (INT Mode)		$1.5 \times t_{SCLK}$	ns
t_{HDARWH}	HOST_D15–0 Hold After $\overline{\text{HOST\_RD}}$ Rising Edge	1.0		ns

[1] NM (Not Measured) — This parameter is based on t_{SCLK}. It is not measured because the number of SCLK cycles for which HOST_ACK remains low depends on the Host DMA FIFO status. This is system design dependent.



In Figure 46, HOST_DATA is HOST_D0–D15.

Figure 46. HOSTDP A/C—Host Read Cycle

ATA/ATAPI-6 Interface Timing

The following tables and figures specify ATAPI timing parameters. For detailed parameter descriptions, refer to the ATAPI specification (ANSI INCITS 361-2002). Table 58 to Table 61 include ATAPI timing parameter equations. System designers should use these equations along with the parameters provided in Table 56 and Table 57. ATAPI timing control registers should be programmed such that ANSI INCITS 361-2002 specifications are met for the desired transfer type and mode.

Table 56. ATA/ATAPI-6 Timing Parameters

Parameter		Min	Max	Unit
t_{SK1}	Difference in output delay after CLKOUT for ATAPI output pins[1]		6	ns
t_{OD}	Output delay after CLKOUT for outputs[1]		12	ns
t_{SUD}	ATAPI_D0-15 or ATAPI_D0-15A Setup Before CLKOUT	6		ns
t_{SUI}	$\overline{\text{ATAPI\_IORDY}}$ Setup Before CLKOUT	6		ns
t_{SUDU}	ATAPI_D0-15 or ATAPI_D0-15A Setup Before $\overline{\text{ATAPI\_IORDY}}$ (UDMA-in only)	2		ns
t_{HDU}	ATAPI_D0-15 or ATAPI_D0-15A Hold After $\overline{\text{ATAPI\_IORDY}}$ (UDMA-in only)	2.6		ns

[1] ATAPI output pins include $\overline{\text{ATAPI\_CS0}}$, $\overline{\text{ATAPI\_CS1}}$, A1-3, $\overline{\text{ATAPI\_DIOR}}$, $\overline{\text{ATAPI\_DIOW}}$, $\overline{\text{ATAPI\_DMACK}}$, ATAPI_D0-15, ATAPI_A0-2A, and ATAPI_D0-15A.

Table 57. ATA/ATAPI-6 System Timing Parameters

Parameter		Source
t_{SK2}	Maximum difference in board propagation delay between any 2 ATAPI output pins[1]	System Design
t_{BD}	Maximum board propagation delay.	System Design
t_{SK3}	Maximum difference in board propagation delay during a read between $\overline{\text{ATAPI\_IORDY}}$ and ATAPI_D0-15/ATAPI_D0-15A.	System Design
t_{SK4}	Maximum difference in ATAPI cable propagation delay between output pin group A and output pin group B[2]	ATAPI Cable Specification
t_{CDD}	ATAPI cable propagation delay for ATAPI_D0-15 and ATAPI_D0-15A signals.	ATAPI Cable Specification
t_{CDC}	ATAPI cable propagation delay for $\overline{\text{ATAPI\_DIOR}}$, $\overline{\text{ATAPI\_DIOW}}$, $\overline{\text{ATAPI\_IORDY}}$, and $\overline{\text{ATAPI\_DMACK}}$ signals.	ATAPI Cable Specification

[1] ATAPI output pins include $\overline{\text{ATAPI\_CS0}}$, $\overline{\text{ATAPI\_CS1}}$, A1-3, $\overline{\text{ATAPI\_DIOR}}$, $\overline{\text{ATAPI\_DIOW}}$, $\overline{\text{ATAPI\_DMACK}}$, ATAPI_D0-15, ATAPI_A0-2A, and ATAPI_D0-15A.
[2] Output pin group A includes $\overline{\text{ATAPI\_DIOR}}$, $\overline{\text{ATAPI\_DIOW}}$, and $\overline{\text{ATAPI\_DMACK}}$. Output pin group B includes $\overline{\text{ATAPI\_CS0}}$, $\overline{\text{ATAPI\_CS1}}$, A1-3, ATAPI_D0-15, ATAPI_A0-2A, and ATAPI_D0-15A.

ATAPI Multiword DMA Transfer Timing

Table 59 and Figure 49 through Figure 52 describe the ATAPI multiword DMA transfer timing. The material in these figures is adapted from ATAPI-6 (INCITS 361-2002[R2007] and is used with permission of the American National Standards Institute (ANSI) on behalf of the Information Technology Industry Council ("ITIC"). Copies of ATAPI-6 (INCITS 361-2002 [R2007] can be purchased from ANSI.

Table 59. ATAPI Multiword DMA Transfer Timing

ATAPI Parameter/Description		ATAPI_MULTI_TIM_x Timing Register Setting[1]	Timing Equation
t_0	Cycle time	TD, TK	$(TD + TK) \times t_{SCLK}$
t_D	$\overline{ATAPI\_DIOR}$/$\overline{ATAPI\_DIOW}$ asserted Pulse Width	TD	$TD \times t_{SCLK}$
t_F	$\overline{ATAPI\_DIOR}$ data hold	N/A	0
$t_{G(write)}$	$\overline{ATAPI\_DIOW}$ data setup	TD	$TD \times t_{SCLK} - (t_{SK1} + t_{SK2} + t_{SK4})$
$t_{G(read)}$	$\overline{ATAPI\_DIOR}$ data setup	TD	$t_{OD} + t_{SUD} + 2 \times t_{BD} + t_{CDD} + t_{CDC}$
t_H	$\overline{ATAPI\_DIOW}$ data hold	TK	$TK \times t_{SCLK} - (t_{SK1} + t_{SK2} + t_{SK4})$
t_I	$\overline{ATAPI\_DMACK}$ to $\overline{ATAPI\_DIOR}$/$\overline{ATAPI\_DIOW}$ setup	TM	$TM \times t_{SCLK} - (t_{SK1} + t_{SK2} + t_{SK4})$
t_J	$\overline{ATAPI\_DIOR}$/$\overline{ATAPI\_DIOW}$ to $\overline{ATAPI\_DMACK}$ hold	TK, TEOC_MDMA	$(TK + TEOC\_MDMA) \times t_{SCLK} - (t_{SK1} + t_{SK2} + t_{SK4})$
t_{KR}	$\overline{ATAPI\_DIOR}$ negated pulse width	TKR	$TKR \times t_{SCLK}$
t_{KW}	$\overline{ATAPI\_DIOW}$ negated pulse width	TKW	$TKW \times t_{SCLK}$
t_{LR}	$\overline{ATAPI\_DIOR}$ to $\overline{ATAPI\_DMARQ}$ delay	N/A	$(TD + TK) \times t_{SCLK} - (t_{OD} + 2 \times t_{BD} + 2 \times t_{CDC})$
t_M	$\overline{ATAPI\_CS0\text{-}1}$ valid to $\overline{ATAPI\_DIOR}$/$\overline{ATAPI\_DIOW}$	TM	$TM \times t_{SCLK} - (t_{SK1} + t_{SK2} + t_{SK4})$
t_N	$\overline{ATAPI\_CS0\text{-}1}$ hold	TK, TEOC_MDMA	$(TK + TEOC\_MDMA) \times t_{SCLK} - (t_{SK1} + t_{SK2} + t_{SK4})$

[1] ATAPI timing register setting should be programmed with a value that guarantees parameter compliance with the ATA ANSI specification for an ATA device mode of operation.

Note that in Figure 49 an alternate ATAPI_D0–15 port bus is ATAPI_D0–15A.

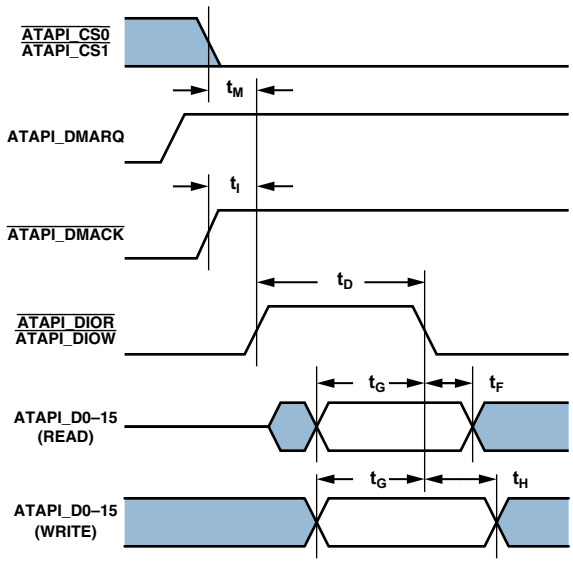


Figure 49. Initiating a Multiword DMA Data Burst

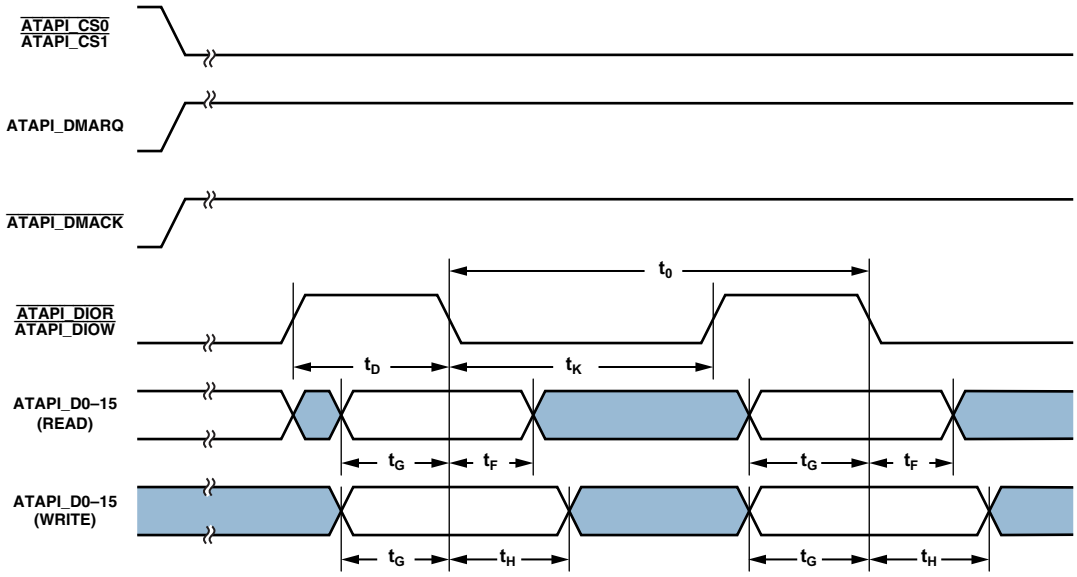


Figure 50. Sustained Multiword DMA Data Burst

In Figure 53 and Figure 54 an alternate ATAPI_D0–15 port bus is ATAPI_D0–15A.

Also note that ATAPI_ADDR pins include A1-3, $\overline{\text{ATAPI\_CS0}}$, and $\overline{\text{ATAPI\_CS1}}$. Alternate ATAPI port ATAPI _ADDR pins include ATAPI_A0A, ATAPI_A1A, ATAPI_A2A, $\overline{\text{ATAPI\_CS0}}$, and $\overline{\text{ATAPI\_CS1}}$.

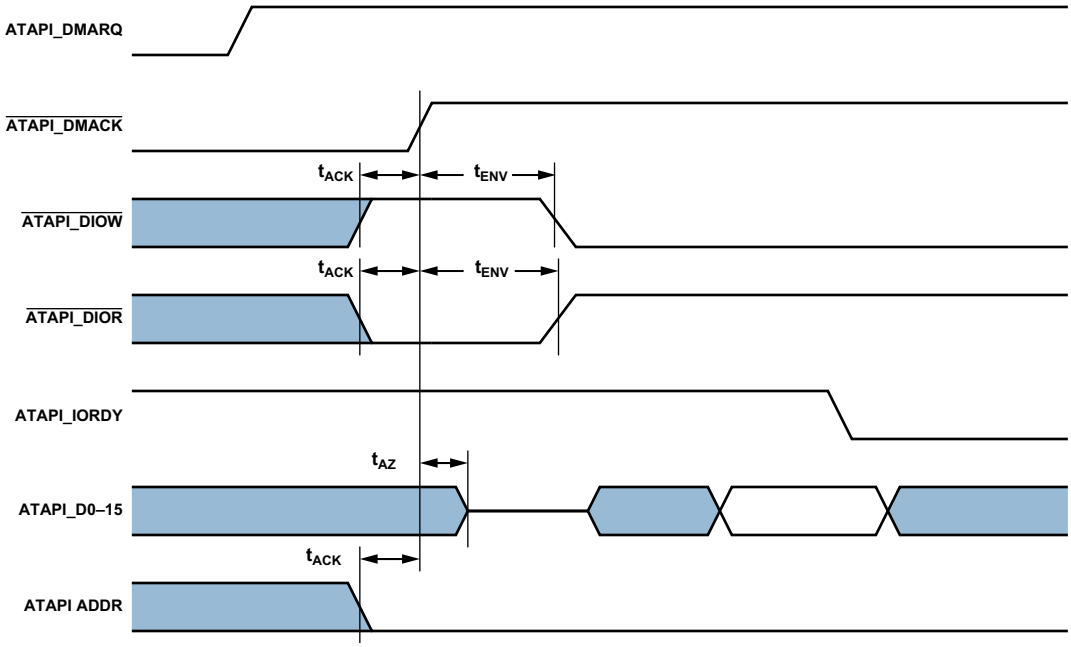


Figure 53. Initiating an Ultra DMA Data-In Burst

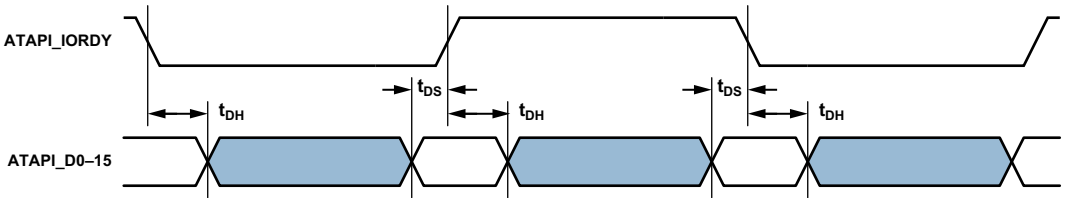


Figure 54. Sustained Ultra DMA Data-In Burst

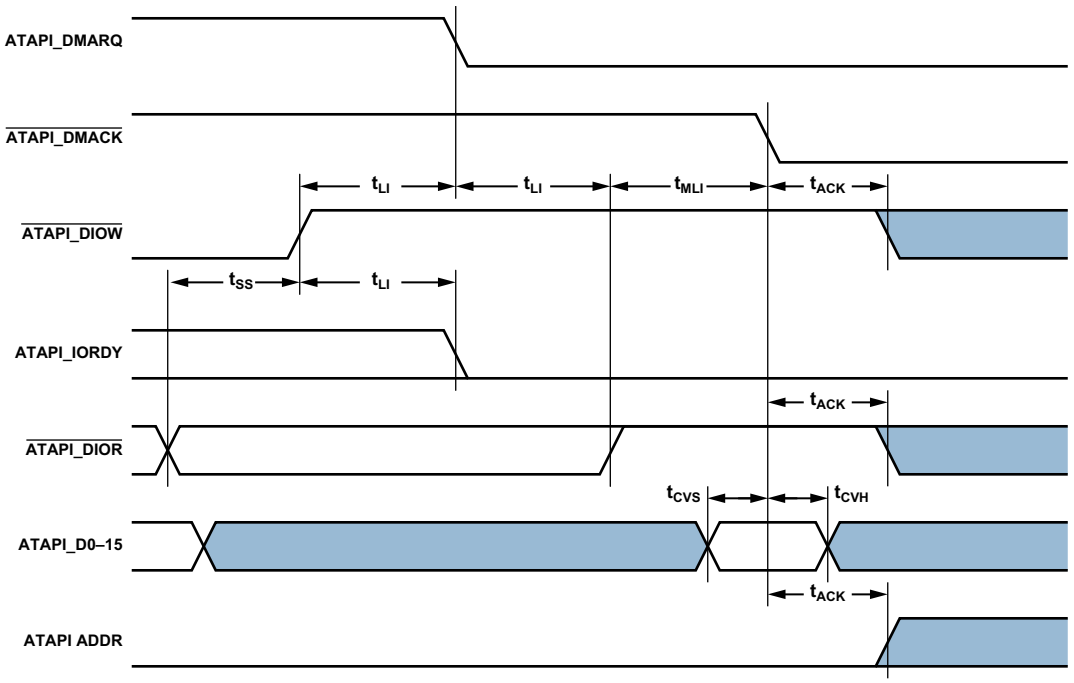


Figure 59. Host terminating an Ultra DMA Data-Out Burst

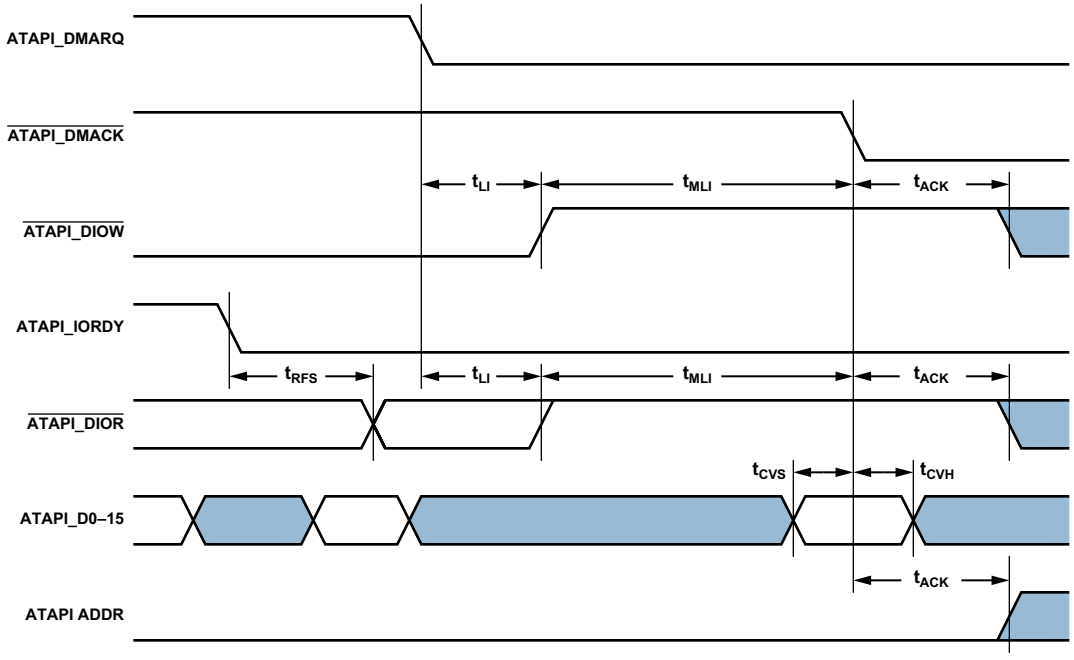


Figure 60. Device Terminating an Ultra DMA Data-Out Burst

400-BALL CSP_BGA PACKAGE

Table 65 lists the CSP_BGA package by signal for the ADSP-BF549. Table 66 on Page 97 lists the CSP_BGA package by ball number.

Table 65. 400-Ball CSP_BGA Ball Assignment (Alphabetical by Signal)

Signal	Ball No.	Signal	Ball No.	Signal	Ball No.	Signal	Ball No.
A1	B2	DA4	G16	DQS1	H18	GND	L10
A2	A2	DA5	F19	$\overline{\text{DRAS}}$	E17	GND	L11
A3	B3	DA6	D20	$\overline{\text{DWE}}$	E18	GND	L12
$\overline{\text{ABE0}}$	C17	DA7	C20	$\overline{\text{EMU}}$	R5	GND	L13
$\overline{\text{ABE1}}$	C16	DA8	F18	EXT_WAKE	M18	GND	L14
$\overline{\text{AMS0}}$	A10	DA9	E19	GND	A1	GND	M6
$\overline{\text{AMS1}}$	D9	DA10	B20	GND	A13	GND	M7
$\overline{\text{AMS2}}$	B10	DA11	F17	GND	A20	GND	M8
$\overline{\text{AMS3}}$	D10	DA12	D19	GND	B11	GND	M9
$\overline{\text{AOE}}$	C10	DBA0	H17	GND	D1	GND	M10
$\overline{\text{ARE}}$	B12	DBA1	H16	GND	D4	GND	M11
$\overline{\text{ATAPI\_PDIAG}}$	P19	$\overline{\text{DCAS}}$	F16	GND	E3	GND	M12
$\overline{\text{AWE}}$	D12	$\overline{\text{DCLK0}}$	E16	GND	F3	GND	M13
BMODE0	W1	DCLK0	D16	GND	F6	GND	M14
BMODE1	W2	DCLK1	C18	GND	F14	GND	N6
BMODE2	W3	$\overline{\text{DCLK1}}$	D18	GND	G9	GND	N7
BMODE3	W4	DCLKE	B18	GND	G10	GND	N8
CLKBUF	D11	$\overline{\text{DCS0}}$	C19	GND	G11	GND	N9
CLKIN	A11	$\overline{\text{DCS1}}$	B19	GND	H7	GND	N10
CLKOUT	L16	DDR_VREF	M20	GND	H8	GND	N11
D0	D13	DDR_VSSR	N20	GND	H9	GND	N12
D1	C13	DQ0	L18	GND	H10	GND	N13
D2	B13	DQ1	M19	GND	H11	GND	N14
D3	B15	DQ2	L19	GND	H12	GND	P8
D4	A15	DQ3	L20	GND	J7	GND	P9
D5	B16	DQ4	L17	GND	J8	GND	P10
D6	A16	DQ5	K16	GND	J9	GND	P11
D7	B17	DQ6	K20	GND	J10	GND	P12
D8	C14	DQ7	K17	GND	J11	GND	P13
D9	C15	DQ8	K19	GND	J12	GND	R9
D10	A17	DQ9	J20	GND	K7	GND	R13
D11	D14	DQ10	K18	GND	K8	GND	R14
D12	D15	DQ11	H20	GND	K9	GND	R16
D13	E15	DQ12	J19	GND	K10	GND	U8
D14	E14	DQ13	J18	GND	K11	GND	V6
D15	D17	DQ14	J17	GND	K12	GND	Y1
DA0	G19	DQ15	J16	GND	K13	GND	Y20
DA1	G17	DQM0	G20	GND	L7	GND_{MP}	E7
DA2	E20	DQM1	H19	GND	L8	MFS	E6
DA3	G18	DQS0	F20	GND	L9	MLF_M	F4

Table 65. 400-Ball CSP_BGA Ball Assignment (Alphabetical by Signal) (Continued)

Signal	Ball No.	Signal	Ball No.	Signal	Ball No.	Signal	Ball No.
MLF_P	E4	PC5	G1	PE15	W17	PH7	H4
MXI	C2	PC6	J5	PF0	K3	PH8	D5
MXO	C1	PC7	H3	PF1	J1	PH9	C4
$\overline{\text{NMI}}$	C11	PC8	Y14	PF2	K2	PH10	C7
PA0	U12	PC9	V13	PF3	K1	PH11	C5
PA1	V12	PC10	U13	PF4	L2	PH12	D7
PA2	W12	PC11	W14	PF5	L1	PH13	C6
PA3	Y12	PC12	Y15	PF6	L4	PI0	A3
PA4	W11	PC13	W15	PF7	K4	PI1	B4
PA5	V11	PD0	P3	PF8	L3	PI2	A4
PA6	Y11	PD1	P4	PF9	M1	PI3	B5
PA7	U11	PD2	R1	PF10	M2	PI4	A5
PA8	U10	PD3	R2	PF11	M3	PI5	B6
PA9	Y10	PD4	T1	PF12	M4	PI6	A6
PA10	Y9	PD5	R3	PF13	N4	PI7	B7
PA11	V10	PD6	T2	PF14	N1	PI8	A7
PA12	Y8	PD7	R4	PF15	N2	PI9	C8
PA13	W10	PD8	U1	PG0	J4	PI10	B8
PA14	Y7	PD9	U2	PG1	K5	PI11	A8
PA15	W9	PD10	T3	PG2	L5	PI12	A9
PB0	W5	PD11	V1	PG3	N3	PI13	C9
PB1	Y2	PD12	T4	PG4	P1	PI14	D8
PB2	T6	PD13	V2	PG5	V15	PI15	B9
PB3	U6	PD14	U4	PG6	Y17	PJ0	R20
PB4	Y4	PD15	U3	PG7	W16	PJ1	N18
PB5	Y3	PE0	V19	PG8	V16	PJ2	M16
PB6	W6	PE1	T17	PG9	Y19	PJ3	T20
PB7	V7	PE2	U18	PG10	Y18	PJ4	N17
PB8	W8	PE3	V14	PG11	U15	PJ5	U20
PB9	V8	PE4	Y16	PG12	P16	PJ6	P18
PB10	U7	PE5	W20	PG13	R18	PJ7	N16
PB11	W7	PE6	W19	PG14	Y13	PJ8	R19
PB12	Y6	PE7	R17	PG15	W13	PJ9	P17
PB13	V9	PE8	V20	PH0	W18	PJ10	T19
PB14	Y5	PE9	U19	PH1	U14	PJ11	M17
PC0	H2	PE10	T18	PH2	V17	PJ12	P20
PC1	J3	PE11	P2	PH3	V18	PJ13	N19
PC2	J2	PE12	M5	PH4	U17	$\overline{\text{RESET}}$	C12
PC3	H1	PE13	P5	PH5	C3	RTXI	A14
PC4	G2	PE14	U16	PH6	D6	RTXO	B14

Table 66 lists the CSP_BGA package by ball number for the ADSP-BF549. Table 65 on Page 94 lists the CSP_BGA package by signal.

Table 66. 400-Ball CSP_BGA Ball Assignment (Numerical by Ball Number)

Ball No.	Signal	Ball No.	Signal	Ball No.	Signal	Ball No.	Signal
A1	GND	C1	MXO	E1	USB_DP	G1	PC5
A2	A2	C2	MXI	E2	USB_DM	G2	PC4
A3	PI0	C3	PH5	E3	GND	G3	USB_ID
A4	PI2	C4	PH9	E4	MLF_P	G4	V_{DDUSB}
A5	PI4	C5	PH11	E5	V_{DDEXT}	G5	V_{DDEXT}
A6	PI6	C6	PH13	E6	MFS	G6	V_{DDEXT}
A7	PI8	C7	PH10	E7	GND_{MP}	G7	V_{DDEXT}
A8	PI11	C8	PI9	E8	V_{DDMP}	G8	V_{DDINT}
A9	PI12	C9	PI13	E9	V_{DDEXT}	G9	GND
A10	$\overline{AMS0}$	C10	$\overline{AOE}$	E10	V_{DDEXT}	G10	GND
A11	CLKIN	C11	$\overline{NMI}$	E11	V_{DDEXT}	G11	GND
A12	XTAL	C12	$\overline{RESET}$	E12	V_{DDEXT}	G12	V_{DDINT}
A13	GND	C13	D1	E13	V_{DDRTC}	G13	V_{DDINT}
A14	RTXI	C14	D8	E14	D14	G14	V_{DDEXT}
A15	D4	C15	D9	E15	D13	G15	V_{DDDDR}
A16	D6	C16	$\overline{ABE1}$	E16	$\overline{DCLK0}$	G16	DA4
A17	D10	C17	$\overline{ABE0}$	E17	$\overline{DRAS}$	G17	DA1
A18	$VROUT_0$	C18	DCLK1	E18	$\overline{DWE}$	G18	DA3
A19	$VROUT_1$	C19	$\overline{DCS0}$	E19	DA9	G19	DA0
A20	GND	C20	DA7	E20	DA2	G20	DQM0
B1	USB_VREF	D1	GND	F1	USB_XI	H1	PC3
B2	A1	D2	USB_VBUS	F2	USB_XO	H2	PC0
B3	A3	D3	USB_RSET	F3	GND	H3	PC7
B4	PI1	D4	GND	F4	MLF_M	H4	PH7
B5	PI3	D5	PH8	F5	V_{DDUSB}	H5	V_{DDEXT}
B6	PI5	D6	PH6	F6	GND	H6	V_{DDEXT}
B7	PI7	D7	PH12	F7	V_{DDEXT}	H7	GND
B8	PI10	D8	PI14	F8	V_{DDEXT}	H8	GND
B9	PI15	D9	$\overline{AMS1}$	F9	V_{DDINT}	H9	GND
B10	$\overline{AMS2}$	D10	$\overline{AMS3}$	F10	V_{DDDDR}	H10	GND
B11	GND	D11	CLKBUF	F11	V_{DDDDR}	H11	GND
B12	$\overline{ARE}$	D12	$\overline{AWE}$	F12	V_{DDDDR}	H12	GND
B13	D2	D13	D0	F13	V_{DDEXT}	H13	V_{DDDDR}
B14	RTXO	D14	D11	F14	GND	H14	V_{DDDDR}
B15	D3	D15	D12	F15	V_{DDVR}	H15	V_{DDDDR}
B16	D5	D16	DCLK0	F16	$\overline{DCAS}$	H16	DBA1
B17	D7	D17	D15	F17	DA11	H17	DBA0
B18	DCLKE	D18	$\overline{DCLK1}$	F18	DA8	H18	DQS1
B19	$\overline{DCS1}$	D19	DA12	F19	DA5	H19	DQM1
B20	DA10	D20	DA6	F20	DQS0	H20	DQ11