# TMS320C645x DSP External Memory Interface (EMIF)

# User's Guide

Literature Number: SPRU971E
December 2005–Revised August 2010

# List of Figures

# List of Tables

# Read This First

## About This Manual

This document describes the operation of the external memory interface (EMIF) in the digital signal processors (DSPs) of the TMS320C645x DSP family.

## Notational Conventions

This document uses the following conventions.

- Hexadecimal numbers are shown with the suffix h. For example, the following number is 40 hexadecimal (decimal 64): 40h.
- Registers in this document are shown in figures and described in tables.
  - Each register figure shows a rectangle divided into fields that represent the fields of the register. Each field is labeled with its bit name, its beginning and ending bit numbers above, and its read/write properties below. A legend explains the notation used for the properties.
  - Reserved bits in a register figure designate a bit that is used for future device expansion.

## Related Documentation From Texas Instruments

The following documents describe the C6000™ devices and related support tools. Copies of these documents are available on the Internet at www.ti.com. *Tip:* Enter the literature number in the search box provided at www.ti.com.

The current documentation that describes the C6000 devices, related peripherals, and other technical collateral, is available in the C6000 DSP product folder at: www.ti.com/c6000.

**SPRU189** — ***TMS320C6000 DSP CPU and Instruction Set Reference Guide.*** Describes the CPU architecture, pipeline, instruction set, and interrupts for the TMS320C6000 digital signal processors (DSPs).

**SPRU965** — ***TMS320C6455 Technical Reference.*** An introduction to the TMS320C6455 DSP and discusses the application areas that are enhanced.

**SPRU198** — ***TMS320C6000 Programmer's Guide.*** Describes ways to optimize C and assembly code for the TMS320C6000™ DSPs and includes application program examples.

**SPRU301** — ***TMS320C6000 Code Composer Studio Tutorial.*** Introduces the Code Composer Studio™ integrated development environment and software tools.

**SPRU321** — ***Code Composer Studio Application Programming Interface Reference Guide.*** Describes the Code Composer Studio™ application programming interface (API), which allows you to program custom plug-ins for Code Composer.

**SPRU871** — ***TMS320C64x+ Megamodule Reference Guide.*** Describes the TMS320C64x+ digital signal processor (DSP) megamodule. Included is a discussion on the internal direct memory access (IDMA) controller, the interrupt controller, the power-down controller, memory protection, bandwidth management, and the memory and cache.

**SPRU190** — ***TMS320C6000 DSP Peripherals Overview Reference Guide.*** Provides a brief description of the peripherals available on the TMS320C6000 digital signal processors (DSPs).

**SPRC234** — ***TMS320C6455 Chip Support Libraries (CSL).*** A download with the latest chip support libraries.

C6000, TMS320C6000, Code Composer Studio are trademarks of Texas Instruments.
All other trademarks are the property of their respective owners.

User's Guide

SPRU971E–December 2005–Revised August 2010

# TMS320C645x DSP EMIF

This document describes the operation and registers of the External Memory Interface (EMIF) in the TMS320C645x DSP.

## 1 Overview

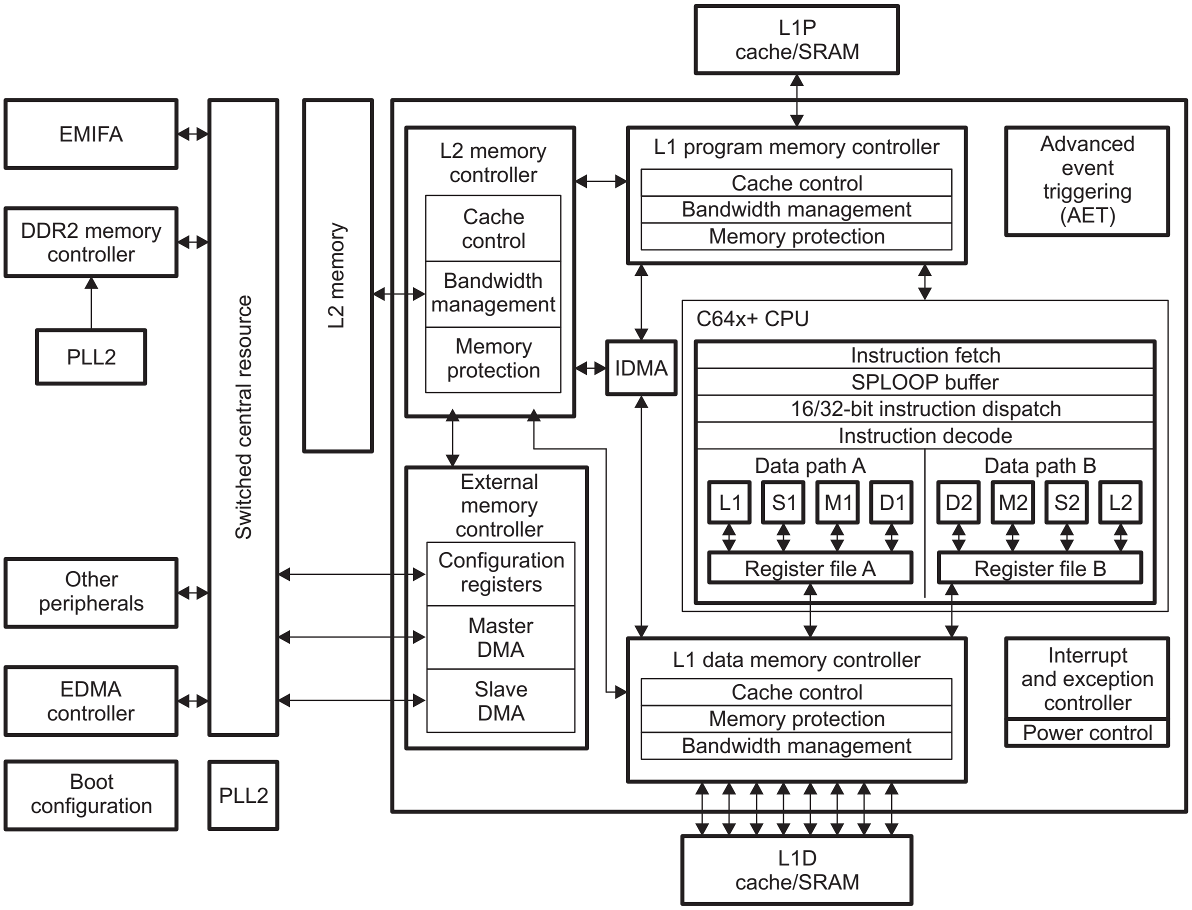
The C645x DSP EMIF interfaces to a variety of external devices, including:

- Pipelined and flow-through synchronous-burst SRAM (SBSRAM)
- ZBT (zero bus turnaround) SRAM and Late Write SRAM
- Synchronous FIFOs
- Asynchronous memory, including SRAM, ROM, and Flash

A block diagram of the C645x DSP is shown in Figure 1. In this document, the term EMIF refers to the EMIFA of C645x devices.

The EMIF services requests of the external bus from on-chip masters such as the enhanced direct-memory access (EDMA) controller and the C64x+ Megamodule, as well as external shared-memory device controllers (through the hold interface, Section 10). On-chip masters place requests to the EMIF through the switched control resource (SCR). For more information on the SCR, see the device-specific data manual.

**Figure 1. TMS320C645x DSP Block Diagram**



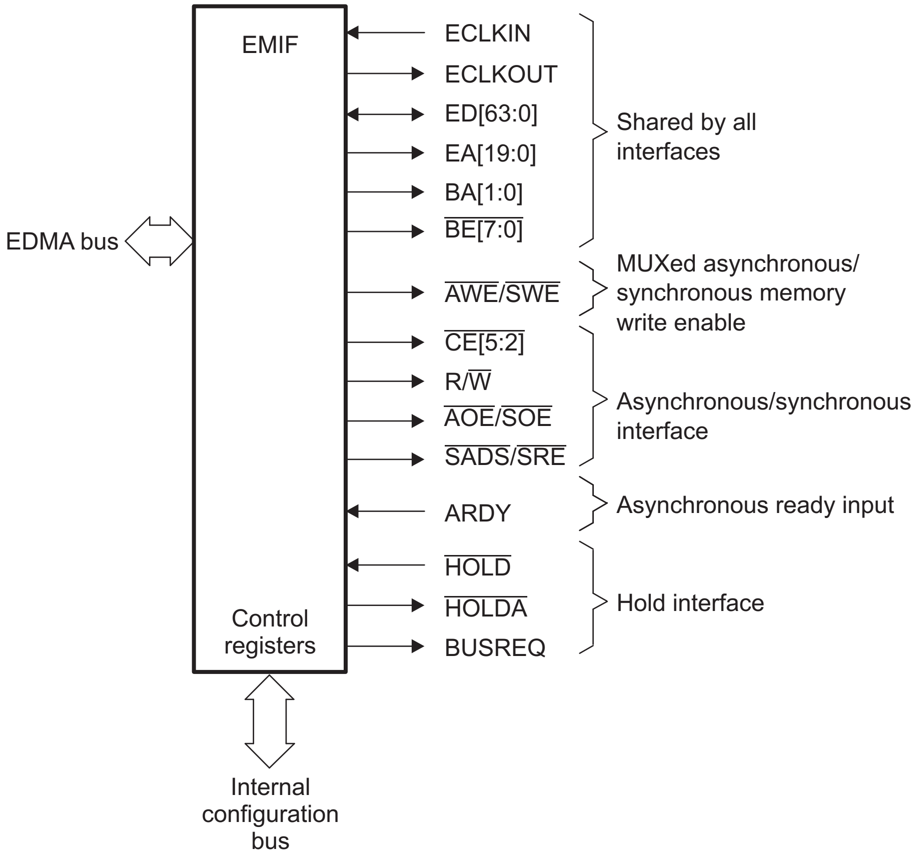
## 2 EMIF Interface Signals

The EMIF signals of the DSP are shown in the figure below and described in the following tables. The EMIF has the following features:

- A 64-bit data bus which can also be configured to be 32-, 16-, and 8-bits wide.
- An output clock, ECLKOUT, generated internally based on the EMIF input clock. You can select one of the following two clocks as the EMIF input clock source at device reset: internal SYSCLK4 for C6455/54; SYSCLK7 for C6457 or external ECLKIN. All of the memories interfacing with the EMIF should operate using ECLKOUT (EMIF clock cycle). The ECLKOUT frequency equals the EMIF input clock frequency.
- A programmable synchronous interface allowing glueless interfaces to synchronous devices such as ZBT SRAM, Late Write SRAM, and Pipelined and Flow-Through SBSRAM devices. Interfaces to synchronous FIFOs are also supported with the addition of external logic.
- A configurable asynchronous interface allowing interfaces to asynchronous devices such as SRAM, EPROM, and Flash, as well as FPGA and ASIC designs.
- Four EMIF spaces (CE2-5) reserved for either asynchronous or synchronous memory accesses.

**NOTE:** The state of the EMIF control pins is not defined while the chip enable pins (CE[5:2]) are high. Furthermore, some control pins may become active while the chip enable pins are driven high. To avoid erroneously activating devices connected to the EMIF, the chip enable pins should be used to select/deselect these devices. If a device being used does not have a chip enable input, then the control pins going to that device should be qualified with a chip enable pin. For example memory interface diagrams, see Section 4 and Section 5.

### Figure 2. EMIF Interface Signals


EMIF
ECLKIN
ECLKOUT
ED[63:0]
EA[19:0]
BA[1:0]
$\overline{\mathrm{BE}}$[7:0]
Shared by all interfaces
EDMA bus
$\overline{\mathrm{AWE}}/\overline{\mathrm{SWE}}$
MUXed asynchronous/ synchronous memory write enable
$\overline{\mathrm{CE}}$[5:2]
R/$\overline{\mathrm{W}}$
$\overline{\mathrm{AOE}}/\overline{\mathrm{SOE}}$
$\overline{\mathrm{SADS}}/\overline{\mathrm{SRE}}$
Asynchronous/synchronous interface
ARDY
Asynchronous ready input
$\overline{\mathrm{HOLD}}$
$\overline{\mathrm{HOLDA}}$
BUSREQ
Hold interface
Control registers
Internal configuration bus

### Table 1. EMIF Pins Used to Access All Device Types

| Pin | I/O/Z | Description |
|---|---|---|
| ECLKIN | I | EMIF external input clock |
| ED[63:0] | I/O/Z | EMIF 64-bit data bus I/O |
| EA[19:0] | O/Z | External address output for EMIF |
| BA[1:0] | O/Z | Address outputs for async/sync interface when the data bus is configured as 8- or 16-bits wide. |
| $\overline{\mathrm{BE}[7:0]}$ | O/Z | Active-low byte enables. Byte enables go active for only the appropriate byte lane for writes and all byte lanes on reads. |
| $\overline{\mathrm{CE2}}$ | O/Z | Active-low chip select for memory space CE2 |
| $\overline{\mathrm{CE3}}$ | O/Z | Active-low chip select for memory space CE3 |
| $\overline{\mathrm{CE4}}$ | O/Z | Active-low chip select for memory space CE4 |
| $\overline{\mathrm{CE5}}$ | O/Z | Active-low chip select for memory space CE5 |

### Table 2. EMIF Pins Specific to Asynchronous Devices

| Pin | I/O/Z | Description |
|---|---|---|
| ARDY | I | Active-high asynchronous ready input used to insert wait states for slow memories and peripherals |
| $\overline{\mathrm{R/W}}$ | O/Z | Read/write enable for asynchronous memory interface |
| $\overline{\mathrm{AOE}}$ | O/Z | Active-low output enable for asynchronous memory interface |
| $\overline{\mathrm{AWE}}$ | O/Z | Active-low write strobe for asynchronous memory interface |

### Table 3. EMIF Pins Specific to Synchronous Devices

| Pin | I/O/Z | Description |
|---|---|---|
| ECLKOUT | O/Z | EMIF output clock at EMIF input clock (ECLKIN or SYSCLK4 for C6455/54; SYSCLK7 for C6457) frequency |
| $\overline{\text{SOE}}$ | O/Z | Programmable synchronous interface output enable |
| $\overline{\text{SADS}}$/$\overline{\text{SRE}}$ | O/Z | Synchronous memory address strobe or read enable.<br>The R_ENABLE field in the CE*n* Configuration Register (CE*n*CFG) selects between $\overline{\text{SADS}}$ and $\overline{\text{SRE}}$:<br>If R_ENABLE = 0, then the $\overline{\text{SADS}}$/$\overline{\text{SRE}}$ signal functions as the $\overline{\text{SADS}}$ signal.<br>If R_ENABLE = 1, then the $\overline{\text{SADS}}$/$\overline{\text{SRE}}$ signal functions as the $\overline{\text{SRE}}$ signal. |
| $\overline{\text{SWE}}$ | O/Z | Synchronous memory write enable |

### Table 4. EMIF Pins Used for Hold Interface

| Pin | I/O/Z | Description |
|---|---|---|
| $\overline{\text{HOLD}}$ | I | Active-low external bus hold (3-state) request from host |
| $\overline{\text{HOLDA}}$ | O | Active-low hold-request-acknowledge to the host |
| BUSREQ | O | Active-high bus request signal. Indicates pending refresh or memory access. |

## 3 Memory Width and Byte Alignment

The EMIF supports memory widths of 8, 16, 32, and 64 bits. The EMIF automatically performs packing and unpacking for accesses to external memories of less than the requested transfer length. Both big and little-endian formats are supported.

Figure 3 shows the byte lane used by the EMIF for each of the bus size configurations. The external memory is always right aligned to the ED[7:0] side of the bus. The endianness mode determines whether byte lane 0 (ED[7:0]) is accessed as byte address 0 (little endian) or as byte address N (big endian), where $2^N$ is memory width in bytes. Similarly, byte lane N is addressed as either byte address 0 (big endian) or as byte address N (little endian).

Table 5 summarizes the addressable memory ranges, as well as the internal address bus to external address bus translation for each of the supported memory types.

**Figure 3. Byte Alignment By Bus Size Configuration**

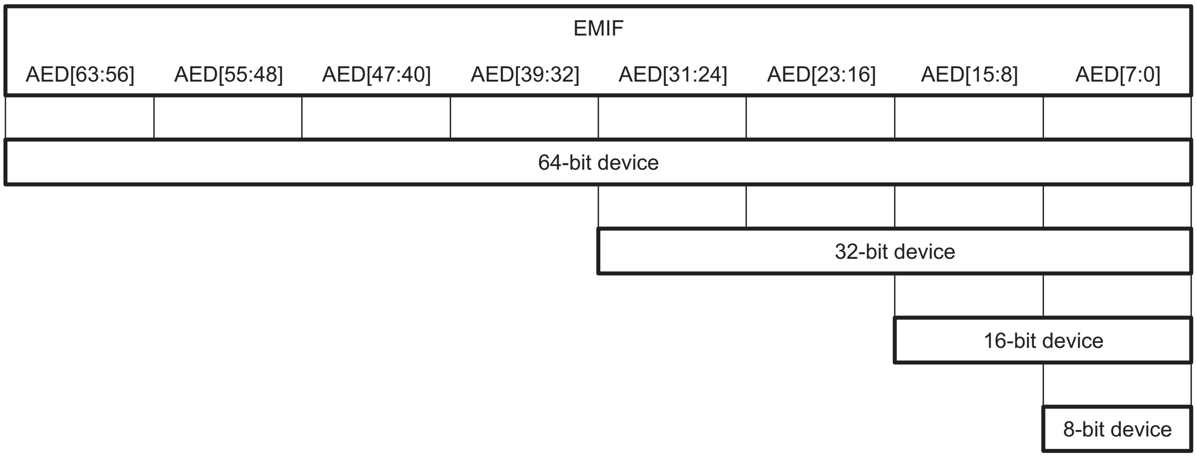


**Table 5. Addressable Memory Ranges and Internal to External Address Bus Translation**

| Memory Type | Memory Width | Maximum Addressable Bytes per CE Space | Internal Address Bus to External Address Bus Translation | Represents |
|---|---|---|---|---|
| Async memory | ×8 | 4M | EA[19:0] = A[21:2]<br>ABA[1:0] = A[1:0] | Byte address |
| | ×16 | 4M | EA[19:0] = A[21:2]<br>ABA1 = A1 | Halfword address |
| | ×32 | 4M | EA[19:0] = A[21:2] | Word address |
| | ×64 | 8M | EA[19:0] = A[22:3] | Doubleword address |
| Programmable sync memory | ×8 | 4M | EA[19:0] = A[21:2]<br>ABA[1:0] = A[1:0] | Byte address |
| | ×16 | 4M | EA[19:0] = A[21:2]<br>ABA1 = A1 | Halfword address |
| | ×32 | 4M | EA[19:0] = A[21:2] | Word address |
| | ×64 | 8M | EA[19:0] = A[22:3] | Doubleword address |

## 4 Asynchronous Interface

The asynchronous interface offers configurable memory cycle types to interface to a variety of memory and peripheral types, including SRAM, EPROM, and flash memory, as well as FPGA and ASIC designs.

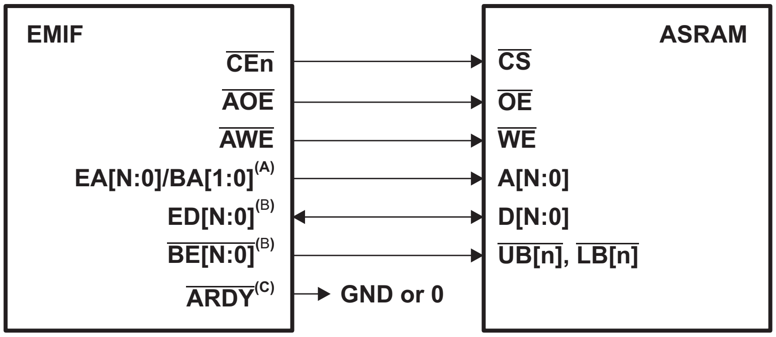
The EMIF allows widths of 64, 32, 16, and 8 bits on any of the CE spaces, as shown in the ASIZE description of the CE*n* Configuration Register (CE*n*CFG) register. The asynchronous interface signals on the EMIF are combined with the SBSRAM memory interface. To avoid bus contention, a programmable turnaround time is inserted between back-to-back accesses to the same or different CE spaces (see Section 6).

Table 6 lists the asynchronous interface pins. Figure 4 shows an EMIF interface to standard SRAM. Figure 5, Figure 6, and Figure 7 show EMIF interfaces to 8-, 16-, and 32-bit ROMs, respectively.

**Table 6. Asynchronous Interface Signal Descriptions**

| EMIF Signal | Function |
|---|---|
| $\overline{\text{CE}}$[5:2] | Active-low chip select |
| ED[63:0] | 64-bit data bus I/O |
| EA[19:0] | External address output |
| BA[1:0] | Bank select outputs or address outputs |
| $\overline{\text{AOE}}$ | Active-low output enable for asynchronous memory interface |
| $\overline{\text{AWE}}$ | Active-low write strobe for asynchronous memory interface |
| R/$\overline{\text{W}}$ | Read/write control for asynchronous memory interface |
| ARDY | Asynchronous ready. Input used to insert wait states into the memory cycle. |

**Figure 4. EMIF-to-32-Bit SRAM Interface Block Diagram**

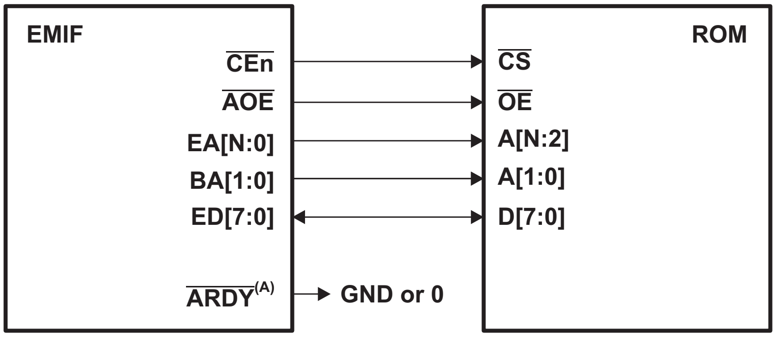


A For information on memory addressing, see Section 4.1.

B For interface to a 64-bit data bus: $\overline{\text{BE}}$[7:0] and ED[63:0] are used.
For interface to a 32-bit data bus: $\overline{\text{BE}}$[3:0] and ED[31:0] are used.
For interface to a 16-bit data bus: $\overline{\text{BE}}$[1:0] and ED[15:0] are used.
For interface to an 8-bit data bus: $\overline{\text{BE0}}$ and ED[7:0] are used.

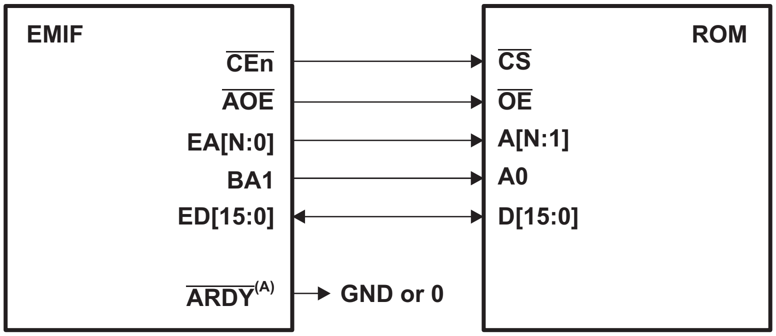
C The disabled state of the read input (ARDY) can be configured using the AP bit of the Asynchronous Wait Cycle Configuration register.

**Figure 5. EMIF-to-8-Bit ROM Interface Block Diagram**



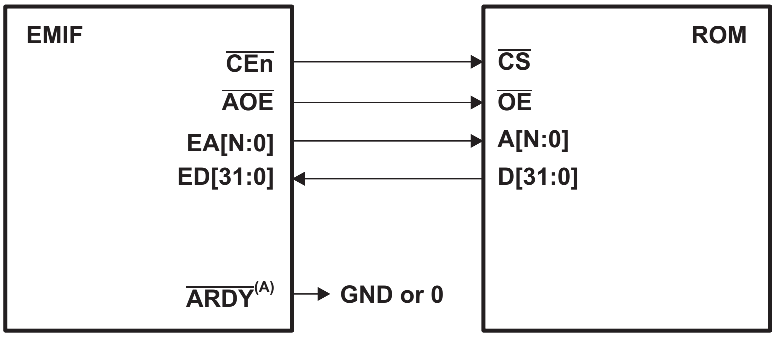
A The disabled state of the read input (ARDY) can be configured using the AP bit of the Asynchronous Wait Cycle Configuration register.

Figure 6. EMIF-to-16-Bit ROM Interface Block Diagram



A The disabled state of the read input (ARDY) can be configured using the AP bit of the Asynchronous Wait Cycle Configuration register.

Figure 7. EMIF-to-32-Bit ROM Interface Block Diagram



A The disabled state of the read input (ARDY) can be configured using the AP bit of the Asynchronous Wait Cycle Configuration register.

### 4.1 Asynchronous Interface Addressing

The EMIF uses the EA[19:0] and BA[1:0] pins to define the address bus that connects to memory devices. For 8-, 16-, and 32-bit devices, EA[19:0] always carries a 32-bit aligned address. For 64-bit devices, EA[19:0] always carries a 64-bit aligned address.

The functionality of BA[1:0] depends on the width of the addressed device, as follows:

- For 64-bit and 32-bit devices, BA[1:0] are not used and should be left unconnected.
- For 16-bit devices, BA1 defines bit 0 of the device's address; BA0 is not used and should be left unconnected.
- For 8-bit devices, BA[1:0] define bits 1 and 0 of the device's address.

### 4.2 Programmable ASRAM Parameters

The EMIF allows a high degree of programmability for shaping asynchronous accesses. The programmable parameters are:

- **Setup:** The time between the beginning of a memory cycle (CE low, address valid) and the activation of AWE (writes) or AOE (reads).
- **Strobe:** The time between the activation and deactivation of AWE (writes) or AOE (reads). The read and write strobe period must not be programmed to be less than two ECLKOUT cycles when AE = 1 (ARDY extends the strobe cycle).
- **Hold:** The time between the deactivation of AWE (writes) or AOE (reads) and the end of the cycle, which can be either an address change or the deactivation of the CE signal.

These parameters are programmed in terms of ECLKOUT cycles. Separate setup, strobe, and hold timing parameters are available for read and write accesses. Minimum values for ASRAM are:

- SETUP ≥1
- STROBE ≥1 (must be greater than or equal to 2 when ARDY is used)
- HOLD ≥1

### 4.3 Asynchronous Reads

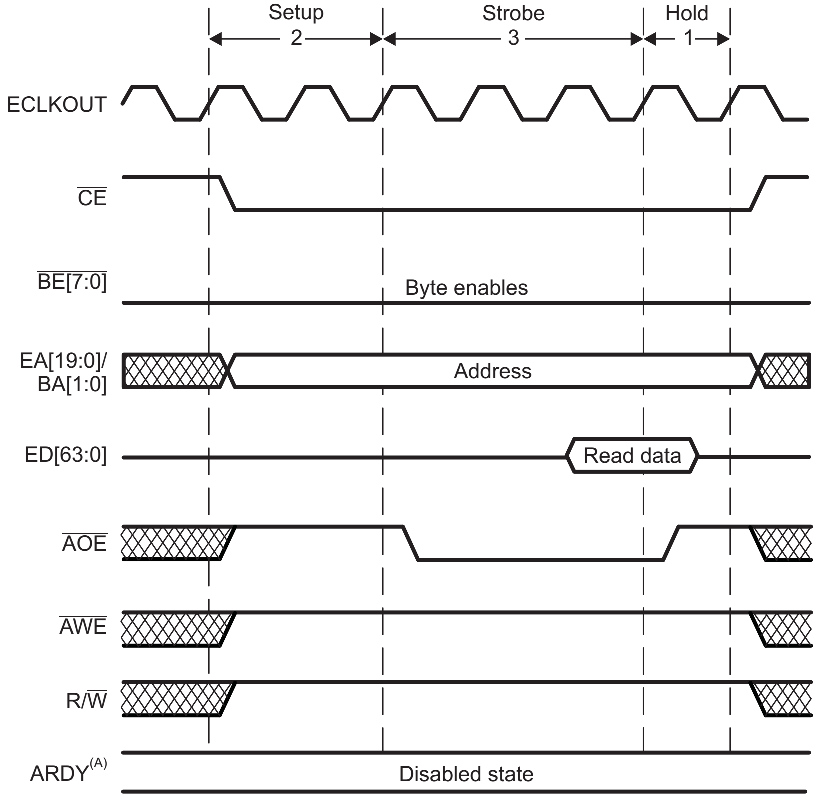
Figure 8 shows an asynchronous read with the ARDY signal always disabled. The disabled state of ARDY depends on the setting of AP. The R_SETUP, R_STROBE, and R_HOLD parameters are programmed with the values 1, 2, and 0, respectively. An asynchronous read proceeds as follows:

- At the beginning of the setup period:
  - CEn becomes active, if not already active from a previous access.
  - BE[7:0] become active.
  - EA[19:0] and BA[1:0] become valid.
- At the beginning of a strobe period, AOE becomes active.
- At the beginning of a hold period:
  - AOE becomes inactive (high).
  - Data is sampled on the ECLKOUT rising edge concurrent with the beginning of the hold period (the end of the strobe period).
- At the end of the hold period:
  - CEn becomes inactive only if another read or write access to the same CEn space is not pending.
  - BE[7:0] become inactive.
  - EA[19:0] and BA[1:0] become invalid.

The ARDY pin can be activated by the external device to extend the strobe period, giving it more time to provide the data. For details on using the ARDY pin., see Section 4.5.

To avoid bus contention, a programmable turnaround time is inserted between back-to-back accesses to the same or different CE spaces (see Section 6).

**Figure 8. Asynchronous Read Timing Diagram**



A The disabled state of the ARDY pin depends on the setting of the AP bit in the Asynchronous Wait Cycle Configuration Register.

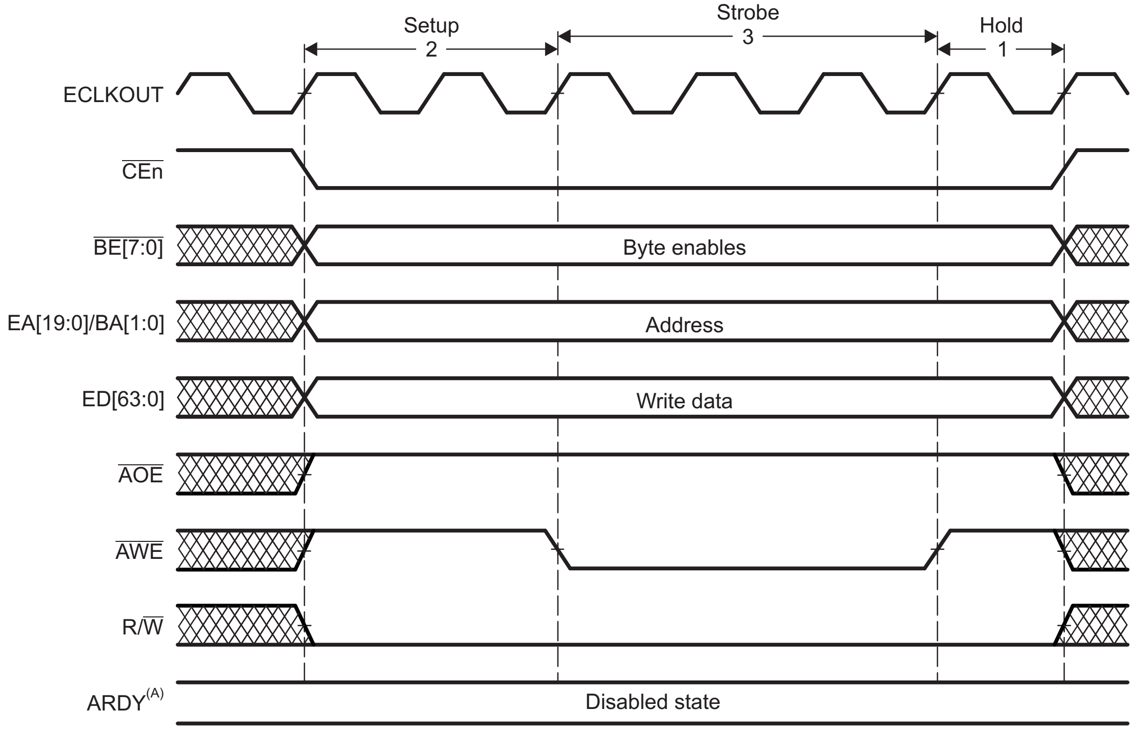
### 4.4 Asynchronous Writes

Figure 9 shows an asynchronous write cycle with the ARDY signal always disabled. The disabled state of ARDY depends on the setting of AP. The R_SETUP, R_STROBE, and R_HOLD parameters are programmed to 1, 2, and 0, respectively. An asynchronous write proceeds as:

- At the beginning of the setup period:
  - CEn becomes active, if not already active from a previous access.
  - BE[3:0] become valid.
  - EA[19:0] and BA[1:0] become valid.
  - ED is driven.
  - R/W becomes active (low).
- At the beginning of a strobe period, AWE becomes active.
- At the beginning of a hold period, AWE becomes inactive.
- At the end of the hold period:
  - EA[19:0] and BA[1:0] become invalid.
  - ED[63:0] becomes invalid.
  - CEn becomes inactive (if no additional read or write accesses to the same CE*n* space are pending).

To avoid bus contention, a programmable turnaround time is inserted between back-to-back accesses to the same or different CE spaces (see Section 6).

Figure 9. Asynchronous Write Timing Diagram



B The disabled state of the ARDY pin depends on the setting of the AP bit in the Asynchronous Wait Cycle Configuration Register.

## 4.5 Ready Input

The EMIF external asynchronous devices may assert control over the length of the strobe period through the use of the ready input (ARDY) pin. The ARDY pin can be activated by setting the AE bit in the CE*n* Configuration Register (CE*n*CFG). When this bit is set, the EMIF monitors the ARDY pin to determine if the attached device wishes to extend the strobe period of the current access cycle beyond the programmed number of clock cycles.

The ARDY pin must be asserted by the second rising edge of the ECLKOUT pin before the end of the programmed strobe period to be registered by the EMIF. When the EMIF detects that the ARDY pin has been asserted, it will begin inserting extra strobe cycles into the operation until the ARDY pin is deactivated by the external device. The ARDY pin must be held in the asserted and deasserted states for a minimum of two ECLKOUT cycles to be synchronized inside of the EMIF. In addition to the two cycles of internal synchronization to deassert ARDY, the EMIF will insert two more wait cycles before returning to the last cycle of the programmed strobe period. The operation will proceed as usual at this point.

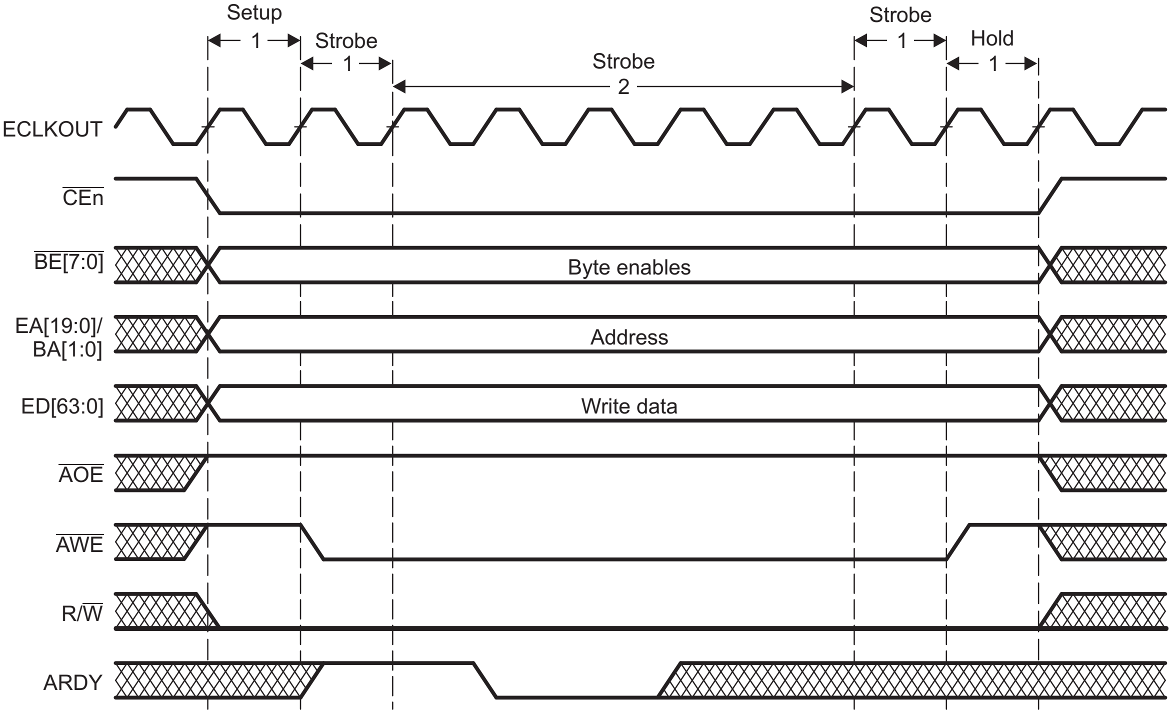
The ARDY pin cannot be used to extend the strobe period indefinitely. The programmable MAX_EXT_WAIT field in the Asynchronous Wait Cycle Configuration register (AWCC) determines the maximum number of ECLKOUT cycles the strobe period may be extended beyond the programmed length. When the counter expires, the EMIF proceeds to the hold period of the operation, regardless of the state of the ARDY pin. The EMIF can also generate an interrupt upon expiration of this counter. For details on enabling this interrupt, see Section 4.8.

For the ARDY pin to function properly, the AP bit of AWCC must be programmed to match the polarity used by the external device. In its reset state of 1, the EMIF will insert wait cycles when the ARDY pin is sampled high. When set to 0, the EMIF will insert wait cycles only when ARDY is sampled low. This programmability allows for a glueless connection to a larger variety of synchronous devices.

Finally, a restriction is placed on the strobe period timing parameters when using the ARDY pin. Specifically, the W_STROBE and R_STROBE fields must not be set to 0 for proper operation.

Figure 10 shows an example of extending a write operation using the ARDY pin, and Figure 11 shows a similar case for a read operation.

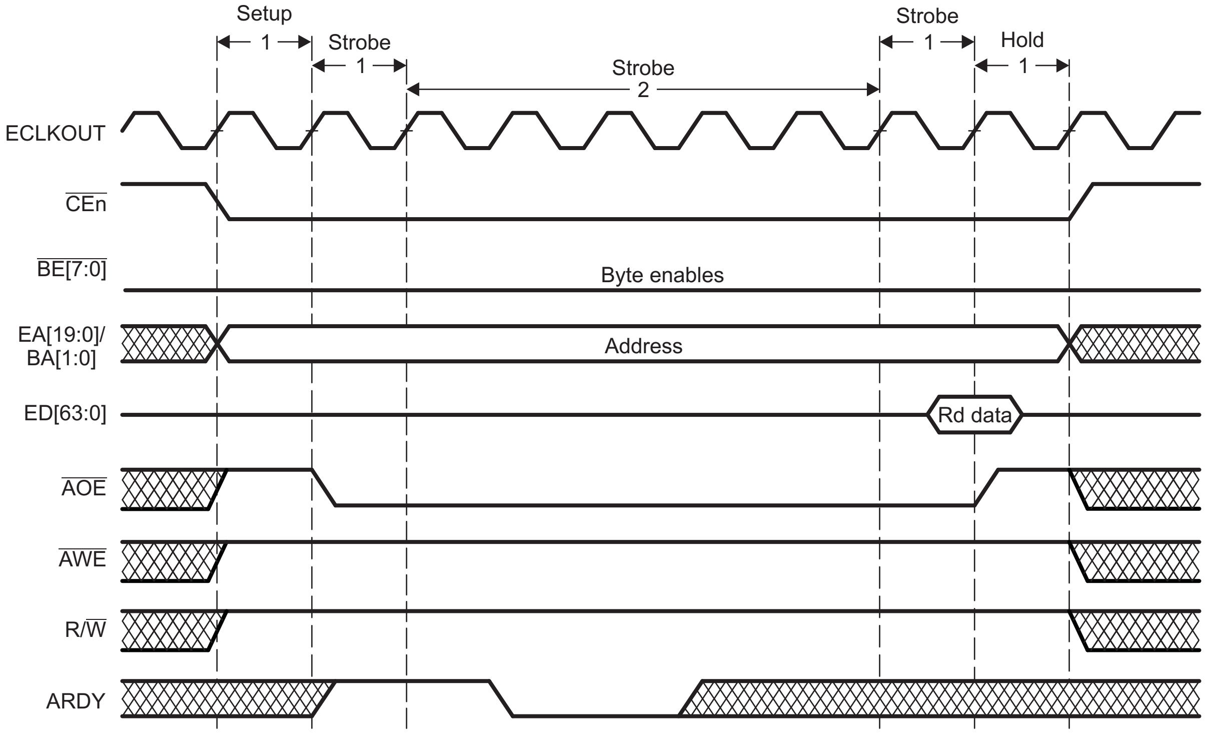
### Figure 10. Asynchronous Write Timing Diagram Using Ready Input



C In this figure:

- SSEL = 0, SS = 0, BWEM = 0, and AE = 1 in CE*n* Configuration register
- AP = 1 in Asynchronous Wait Cycle Configuration register
- W_SETUP = 0, W_STROBE = 1, and W_HOLD = 0 in CE*n* Configuration register

Figure 11. Asynchronous Read Timing Diagram Using Ready Input



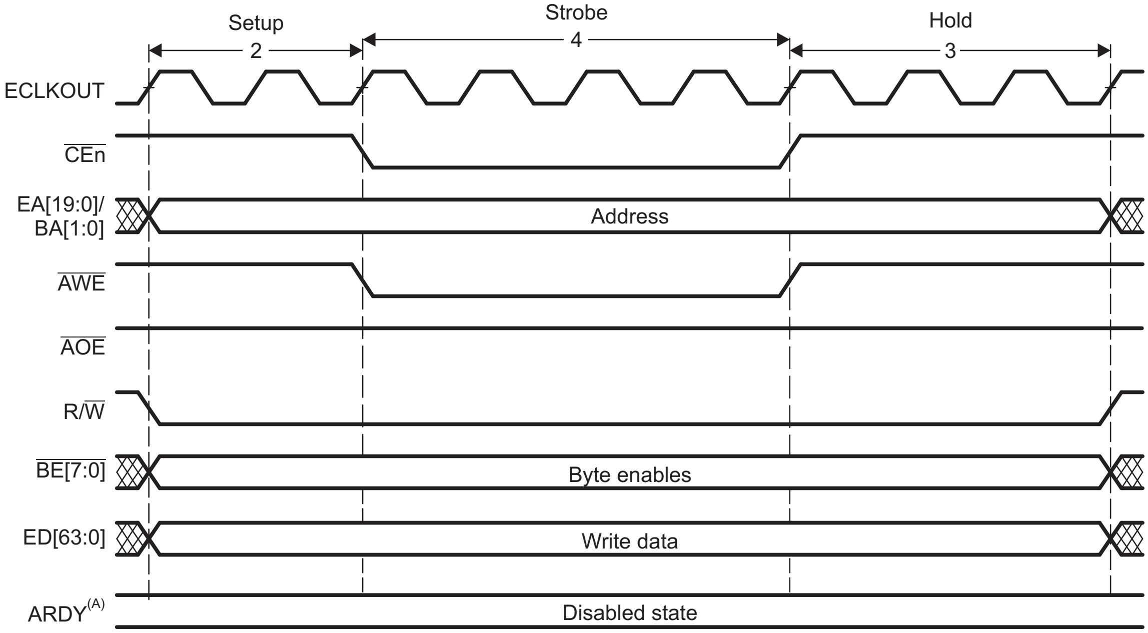
D In this figure:

- SSEL = 0, SS = 0, BWEM = 0, and AE = 1 in CE*n* Configuration register
- AP = 1 in Asynchronous Wait Cycle Configuration register
- W_SETUP = 0, W_STROBE = 1, and W_HOLD = 0 in CE*n* Configuration register

## 4.6 Asynchronous Memory Access in Select Strobe Mode

If the SS bit in the Asynchronous Wait Cycle Configuration register for a particular chip select is set, that chip select acts as a strobe. In other words, the timing of CEn is the same as AWE and AOE. The value of the BWEM field in the Asynchronous Wait Cycle Configuration register is ignored in this mode; i.e., the BE[7:0] pins act as byte enables. See Figure 12 and Figure 13.

Figure 12. Asynchronous Write in Select Strobe Mode

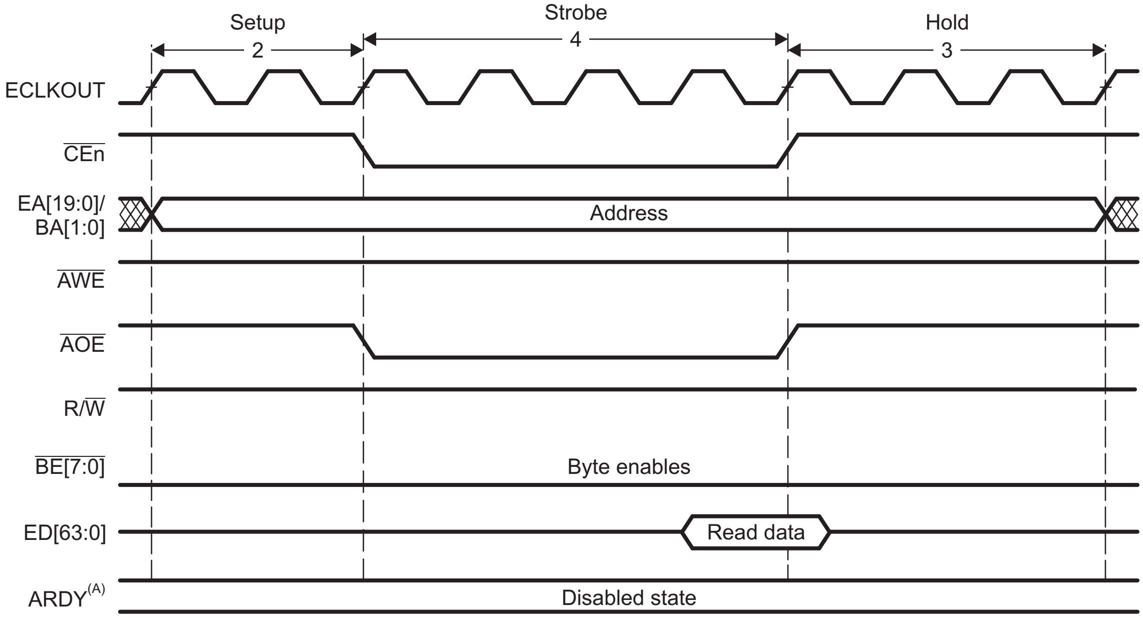


E The disabled state of the ARDY pin depends on the setting of the AP bit in the Asynchronous Wait Cycle Configuration Register.

F In this figure:

- SSEL = 0, SS = 1, BWEM = 0, and AE = 0 in CE*n* Configuration register
- W_SETUP = 1, W_STROBE = 3, and W_HOLD = 2 in CE*n* Configuration register
- ARDY is set to its inactive state

**Figure 13. Asynchronous Read in Select Strobe Mode**



A The disabled state of the ARDY pin depends on the setting of the AP bit in the Asynchronous Wait Cycle Configuration Register.

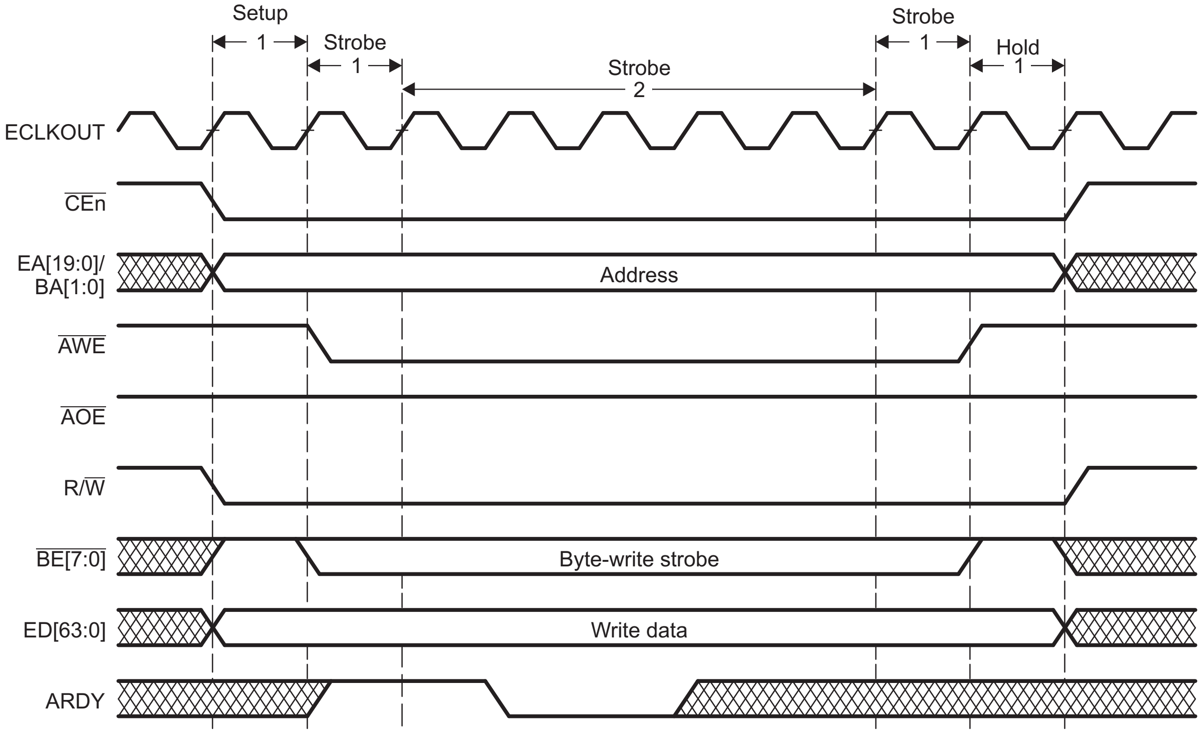
B In this figure:

- SSEL = 0, SS = 1, BWEM = 0, and AE = 0 in CE*n* Configuration register
- R_SETUP = 1, R_STROBE = 3, and R_HOLD = 2 in CE*n* Configuration register
- ARDY is set to its inactive state

## *4.7 Asynchronous Memory Access in WE Strobe Mode*

The $\overline{\text{BE[7:0]}}$ pins act as write strobes when the WE Strobe Mode is enabled (the BWEM bit in Asynchronous Wait Cycle Configuration register is set). See Figure 14 and Figure 15. The WE Strobe Mode is useful when combining multiple 8-bit devices to create a 16-, 32-, or 64-bit data bus. This mode allows the EMIF to perform byte writes to a group of 8-bit devices which do not have byte enable inputs. In this configuration, the byte enable pins are connected to the write strobes of the eight 8-bit devices. This mode cannot be used when in the Select Strobe mode, as the Select Strobe mode overrides this mode.

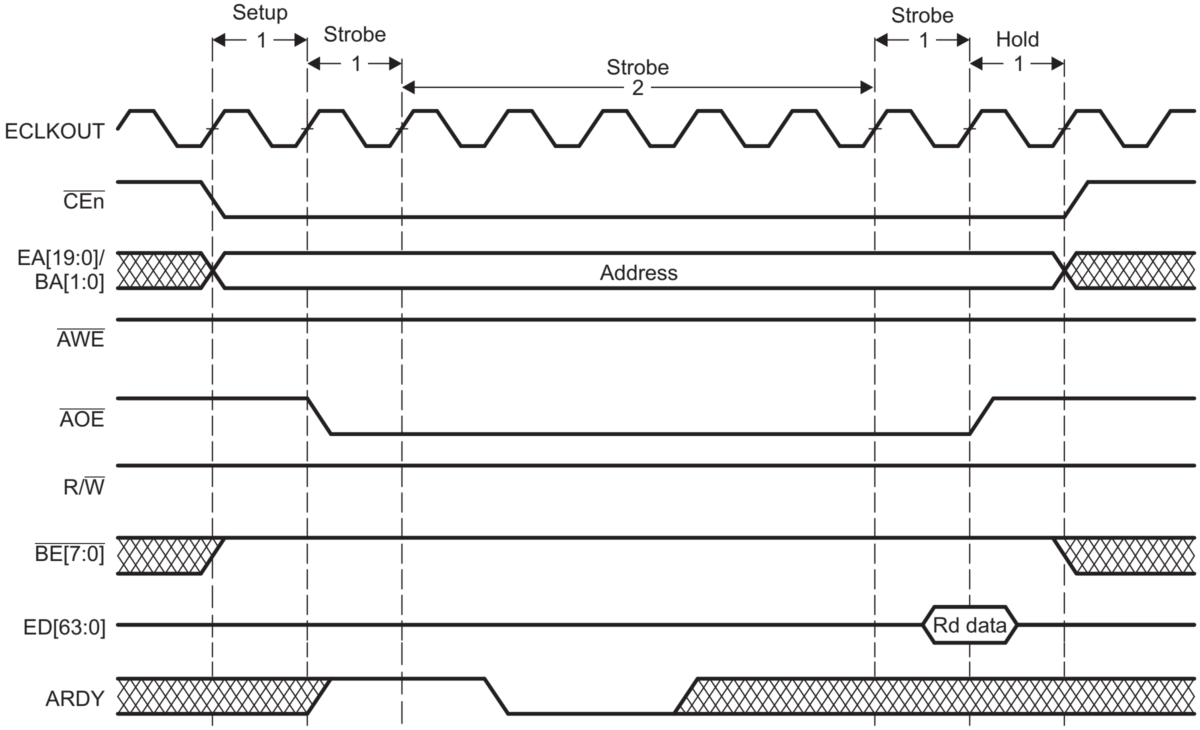
**Figure 14. Asynchronous Write in WE Strobe Mode**



A In this figure:

- SSEL = 0, SS = 0, BWEM = 1, and AE = 1 in CE*n* Configuration register
- AP = 1 in Asynchronous Wait Cycle Configuration register
- W_SETUP = 0, W_STROBE = 1, and W_HOLD = 0 in CE*n* Configuration register

**Figure 15. Asynchronous Read in WE Strobe Mode**



A In this figure:

- SSEL = 0, SS = 0, BWEM = 1, and AE = 1 in CE*n* Configuration register
- AP = 1 in Asynchronous Wait Cycle Configuration register
- R_SETUP = 0, R_STROBE = 1, and R_HOLD = 0 in CE*n* Configuration register

## 4.8 Asynchronous Time-Out Interrupt

The EMIF can generate an asynchronous time-out interrupt to the CPU when the attached device fails to deassert the ARDY pin within the number of cycles defined in the MAX_EXT_WAIT field of the Asynchronous Wait Cycle Configuration register (AWCC). This interrupt is enabled by writing a 1 to the AT_MASK_SET bit of the Interrupt Mask Set register (INTMSKSET) and is disabled by writing a 2 to the AT_MASK_CLR field of the Interrupt Mask Clear register (INTMSKCLR). Both AT_MASK_SET and AT_MASK_CLR bits read 1 if the interrupt is enabled and read 0 if the interrupt is disabled.

Two other bits monitor the status of each interrupt. The AT bit of the Interrupt Raw register (INTRAW) is set when an asynchronous time-out occurs, regardless of whether or not the interrupt has been enabled. The AT_MASKED bit of the Interrupt Masked register (INTMSK) is set when an asynchronous time-out occurs and the interrupt has been enabled. The AT_MASKED bit will not be set if the interrupt is disabled. Writing a 1 to either the AT bit or the AT_MASKED bit will clear both bits.

Table 7 contains a summary of the interrupt monitor and control bit fields. For complete details on the register fields, see Section 12.

**Table 7. Interrupt Monitor and Control Bit Fields**

| Bit Name | Register Name | Description |
|---|---|---|
| AT | Interrupt Raw register (INTRAW) | This bit is always set when an asynchronous time-out occurs. |
| AT_MASKED | Interrupt Masked register (INTMSK) | This bit is only set when an asynchronous time-out occurs and the interrupt has been enabled by writing a 1 to AT_MASK_SET. |
| AT_MASK_SET | Interrupt Mask Set register (INTMSKSET) | Writing a 1 to this bit enables the asynchronous time-out interrupt. |
| AT_MASK_CLR | Interrupt Mask Clear register (INTMSKCLR) | Writing a 1 to this bit disables the asynchronous time-out interrupt. |

# 5 Programmable Synchronous Interface

The programmable synchronous interface of the EMIF supports glueless interfaces to the following devices:

- Pipelined and flow-through SBSRAM
- Zero bus turnaround (ZBT) synchronous pipeline SRAM, Late Write SRAM

The programmable synchronous interface can also interface to Standard Synchronous FIFOs with the addition of glue logic. Note that the EDMA constant addressing mode should not be used when using the EDMA to move data to/from FIFOs connected to the EMIF. For more information, see Section 9, *EDMA Programming Considerations*.

The bit fields in the CE*n* Configuration Registers (CE*n*CFG) control the cycle timing parameters for programmable synchronous interface synchronization. For a description of the CE*n*CFG registers, see Section 12.4. To avoid bus contention, a programmable turnaround time is inserted between back-to-back accesses to the same or different CE spaces (see Section 6).

Table 8 shows the programmable synchronous interface pins.

**Table 8. Programmable Synchronous Interface Pins**

| EMIF Signal | Signal Function |
|---|---|
| $\overline{\text{CE}}$[5:2] | Chip select |
| ED[63:0] | 64-bit data bus I/O |
| EA[19:0] | External address output |
| BA[1:0] | Bank select outputs or address outputs |
| $\overline{\text{BE}}$[7:0] | Byte enable |
| $\overline{\text{SADS/SRE}}$ | Synchronous memory address strobe or read enable |
| $\overline{\text{SOE}}$ | Synchronous memory output enables |
| $\overline{\text{SWE}}$ | Synchronous memory write enable |
| ECLKOUT | EMIF output clock at EMIF input clock (ECLKIN or SYSCLK4 for C6455/54; SYSCLK7 for C6457) clock frequency |

## 5.1 *Programmable Synchronous Interface Addressing*

The EMIF uses the EA[19:0] and BA[1:0] pins to define the address bus that connects to memory devices. For 8-, 16-, and 32-bit devices, EA[19:0] always carries a 32-bit aligned address. For 64-bit devices, EA[19:0] always carries a 64-bit aligned address.

The functionality of BA[1:0] depends on the width of the device being addressed, as follows:

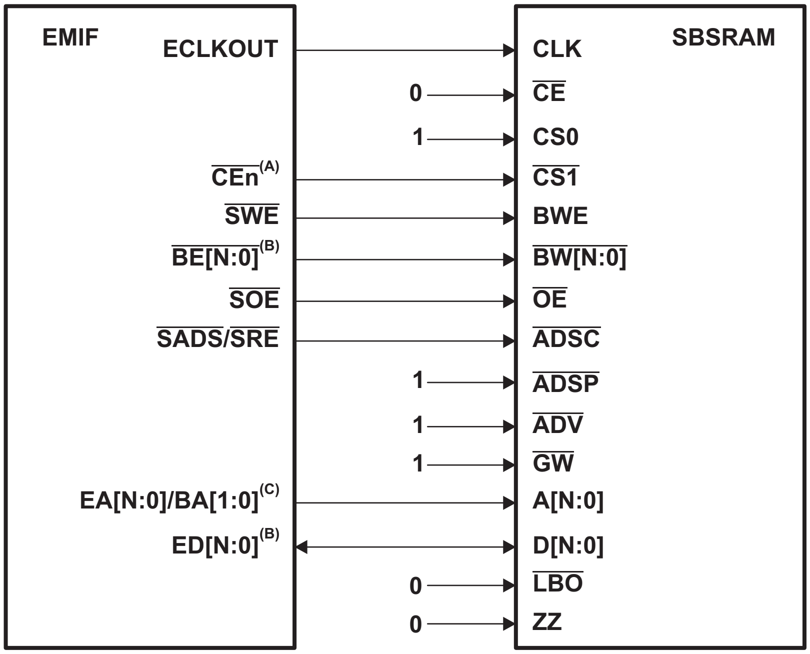
- For 64-bit and 32-bit devices, BA[1:0] are not used and should be left unconnected.
- For 16-bit devices, BA1 defines bit 0 of the device's address; BA0 is not used and should be left unconnected.
- For 8-bit devices, BA[1:0] define bits 1 and 0 of the device's address.

## 5.2 *SBSRAM Interface*

The programmable synchronous mode supports the SBSRAM interface shown in Figure 16. The EMIF interface does not explicitly make use of the burst mode of the SBSRAM (the ADV signal on the SBSRAM is tied high). Instead, the EMIF performs SBSRAM bursts by issuing a new command every cycle. At the end of a burst where no accesses are pending in that CE space, the EMIF issues a deselect cycle. The R_ENABLE field in CE*n*CFG should be cleared for the SBSRAM interface to enable the $\overline{\text{SADS}}$ signal.

The EMIF also supports programmable read and write latency, which allows it to interface with different types of synchronous memories.

**Figure 16. EMIF-to-SBSRAM Interface Block Diagram**



A Only $\overline{\text{CE}}$[5:2] can be used for synchronous memory interfaces.

B For interface to a 64-bit data bus: $\overline{\text{BE}}$[7:0] and ED[63:0] are used.
For interface to a 32-bit data bus: $\overline{\text{BE}}$[3:0] and ED[31:0] are used.
For interface to a 16-bit data bus: $\overline{\text{BE}}$[1:0] and ED[15:0] are used.
For interface to an 8-bit data bus: $\overline{\text{BE0}}$ and ED[7:0] are used.

C For information on memory addressing, see Section 5.1.

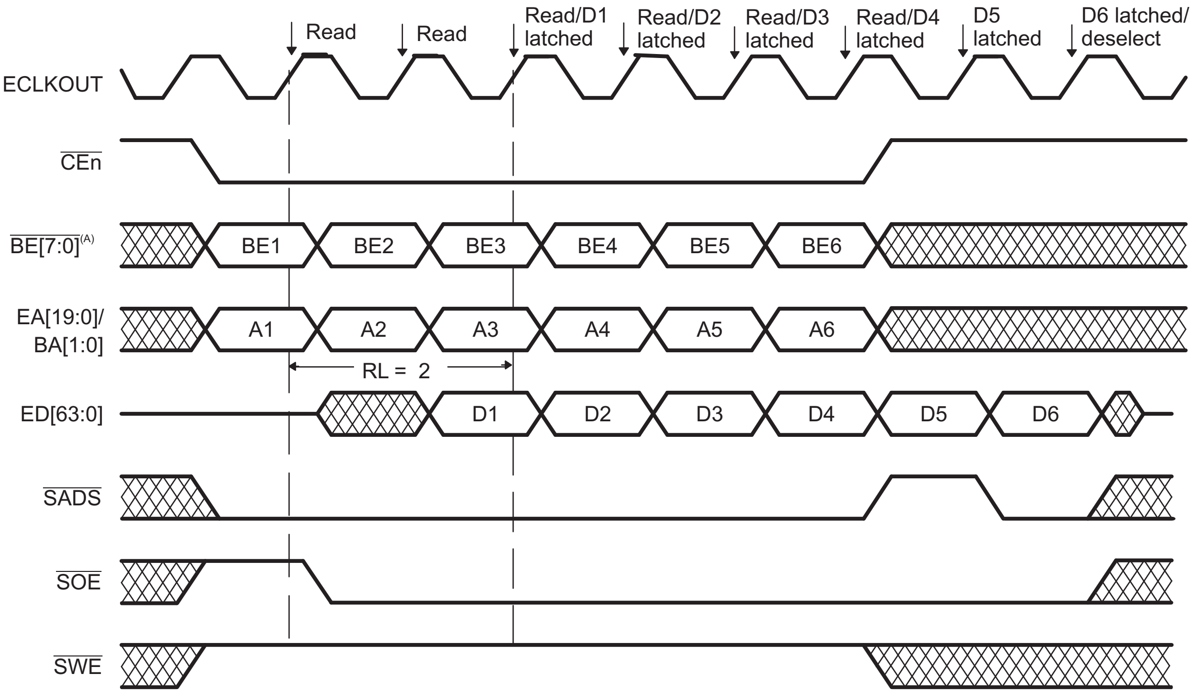
### 5.2.1 SBSRAM Read

Figure 17 shows an SBSRAM read cycle. The EMIF issues a read command to the SBSRAM by asserting CEn and SADS low. The EMIF provides the memory address on EA/BA on each successive cycle. The data mask on BE is always driven high. The SOE is asserted one cycle before the programmed read latency for the chip select. A deselect command is issued by driving CEn high and SADS low on the clock cycle during the last data out. The SOE pin is de-asserted after the deselect cycle.

For the standard SBSRAM interface, set the following fields in the CE*n* Configuration Register (CE*n*CFG, Section 12.4):

- SSEL = 1; to configure chip select for synchronous memory
- R_LTNCY = 10b; 2 cycle read latency
- W_LTNCY = 00b; 0 cycle write latency
- CE_EXT = 0; CEn goes inactive after the final command has been issued
- R_ENABLE = 0; SADS/SRE signal acts as SADS signal

**Figure 17. SBSRAM Six-Element Read Timing Diagram**



A BE[7:0] is driven during reads only if the RD_BE_EN bit of the CE*n* Configuration register is set to 1. BE[7:0] stays high during reads if RD_BE_EN = 0.

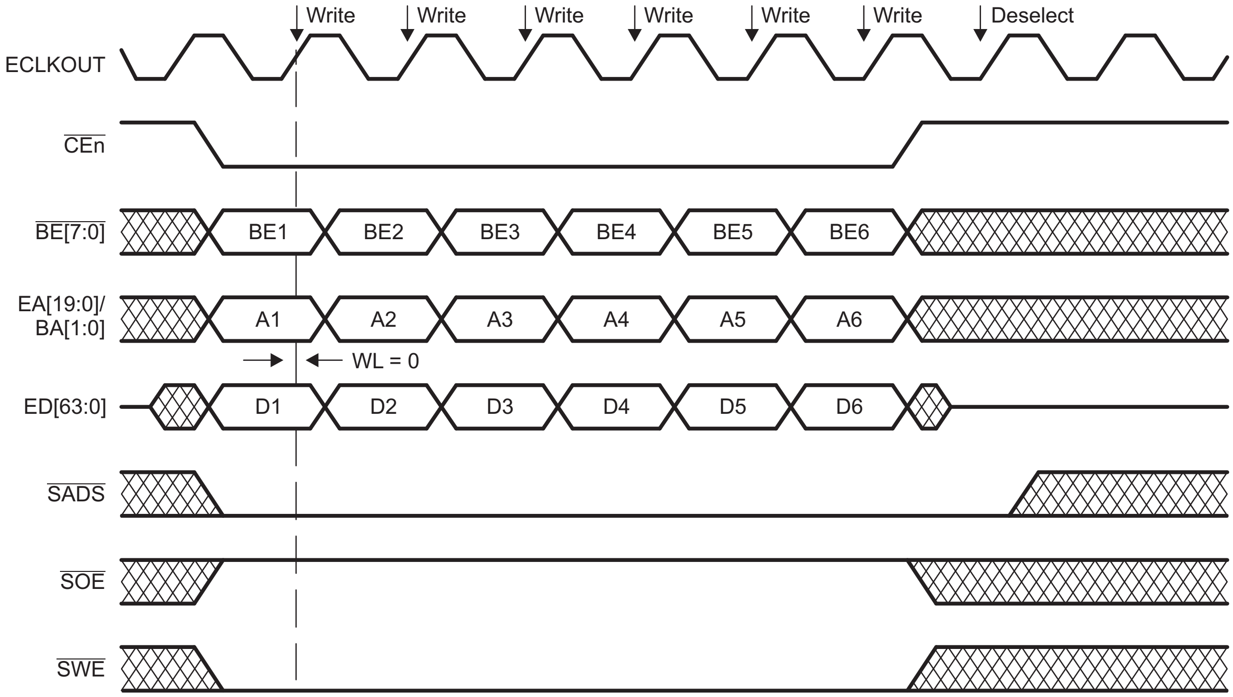
### 5.2.2 SBSRAM Write

Figure 18 shows an SBSRAM write cycle. The EMIF issues a write command to the SBSRAM by asserting CEn, SADS, and SWE low. The address, byte enables, and write data are presented to the memory on EA/BA, BE, and ED, respectively, on each cycle without any latency on the write data. A deselect command is issued by de-asserting the CE signal after the write burst is complete.

For the standard SBSRAM interface, set the following fields in CE*n*CFG to their default state:

- SSEL = 1; to configure chip select for synchronous memory
- R_LTNCY = 10b; 2 cycle read latency
- W_LTNCY = 00b; 0 cycle write latency
- CE_EXT = 0b; CEn goes inactive after the final command has been issued
- R_ENABLE = 0; SADS/SRE signal acts as SADS signal

**Figure 18. SBSRAM Six-Element Write Timing Diagram**

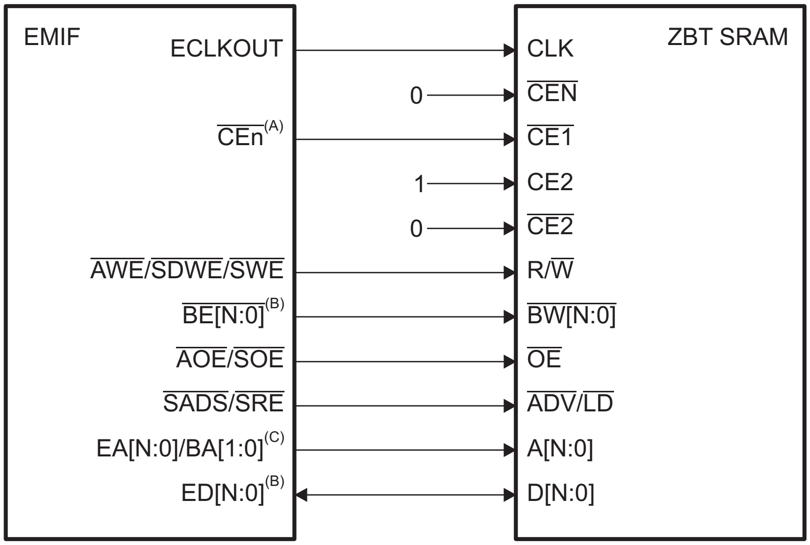


## 5.3 Zero Bus Turnaround (ZBT) SRAM Interface

The programmable synchronous mode supports the zero bus turnaround (ZBT) SRAM interface shown in Figure 19. For the ZBT SRAM interface, set the following fields in the CE*n* Configuration Register (CE*n*CFG):

- SSEL = 1; to configure chip select for synchronous memory
- R_LTNCY = 10b; 2 cycle read latency
- W_LTNCY = 10b; 2 cycle write latency
- CE_EXT = 0b; $\overline{\text{CEn}}$ goes inactive after the final command has been issued
- R_ENABLE = 0b; $\overline{\text{SADS}}/\overline{\text{SRE}}$ signal acts as $\overline{\text{SADS}}$ signal

**Figure 19. EMIF-to-Zero Bus Turnaround (ZBT) SRAM Interface Block Diagram**



B Only $\overline{\text{CE[5:2]}}$ can be used for synchronous memory interfaces.

C For interface to a 64-bit data bus: $\overline{\text{BE[7:0]}}$ and ED[63:0] are used.
For 32-bit interface, $\overline{\text{BE[3:0]}}$ and ED[31:0] are used.
For 16-bit interface, $\overline{\text{BE[1:0]}}$ and ED[15:0] are used.
For 8-bit interface, $\overline{\text{BE[0]}}$ and ED[7:0] are used.

D The EDMA constant addressing mode should not be used when using the EDMA to move data to/from FIFOs connected to the EMIF. For more information, see Section 9, *EDMA Programming Considerations*.

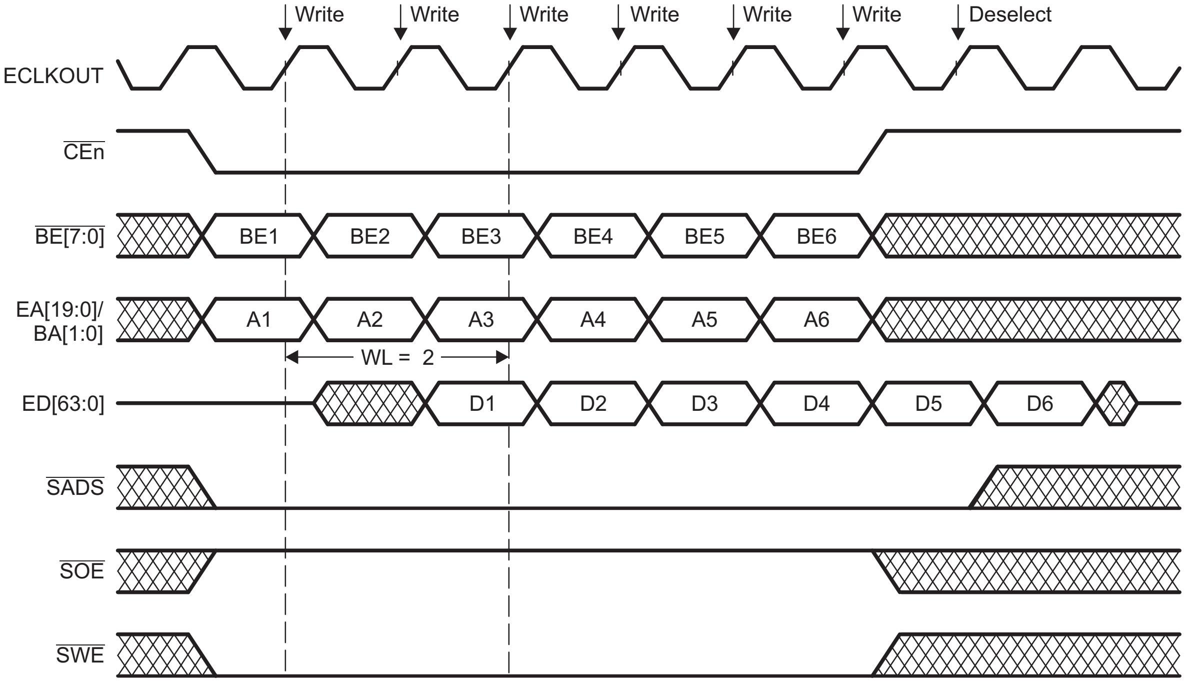
### 5.3.1 ZBT SRAM Read

The ZBT SRAM read waveforms are identical to the SBSRAM read waveforms, as the register settings corresponding to the reads are identical. For details, see Section 5.2.1.

### 5.3.2 ZBT SRAM Write

For ZBT SRAM writes, the control signal waveforms are the same as standard SRAM writes. However, the write data is delayed by two cycles, as controlled by W_LTNCY = 10b. Figure 20 shows the ZBT SRAM write timing.

**Figure 20. Zero Bus Turnaround (ZBT) SRAM Six-Element Write Timing Diagram**



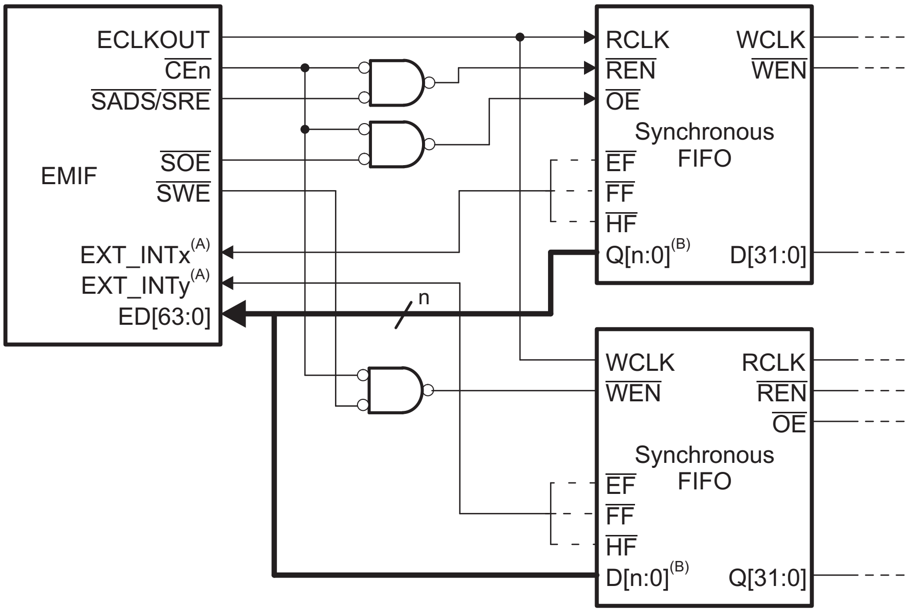
### 5.4 Synchronous FIFO Interface

The programmable synchronous mode supports interface to standard timing synchronous FIFOs with the addition of glue logic as shown in Figure 21. For a synchronous FIFO interface, set the following fields in the CE*n* Configuration Register (CE*n*CFG):

- R_ENABLE = 1b; $\overline{\text{SADS}}/\overline{\text{SRE}}$ signal acts as $\overline{\text{SRE}}$ signal

Figure 21 shows the synchronous FIFO interface with glue.

**Figure 21. Read and Write Synchronous FIFO Interface With Glue Block Diagram**



A GPIO pins on C645x devices may be configured as external interrupt sources to the CPU. For more details, see the *TMS320C645x DSP General-Purpose Input/Output (GPIO) User's Guide* (SPRU724).

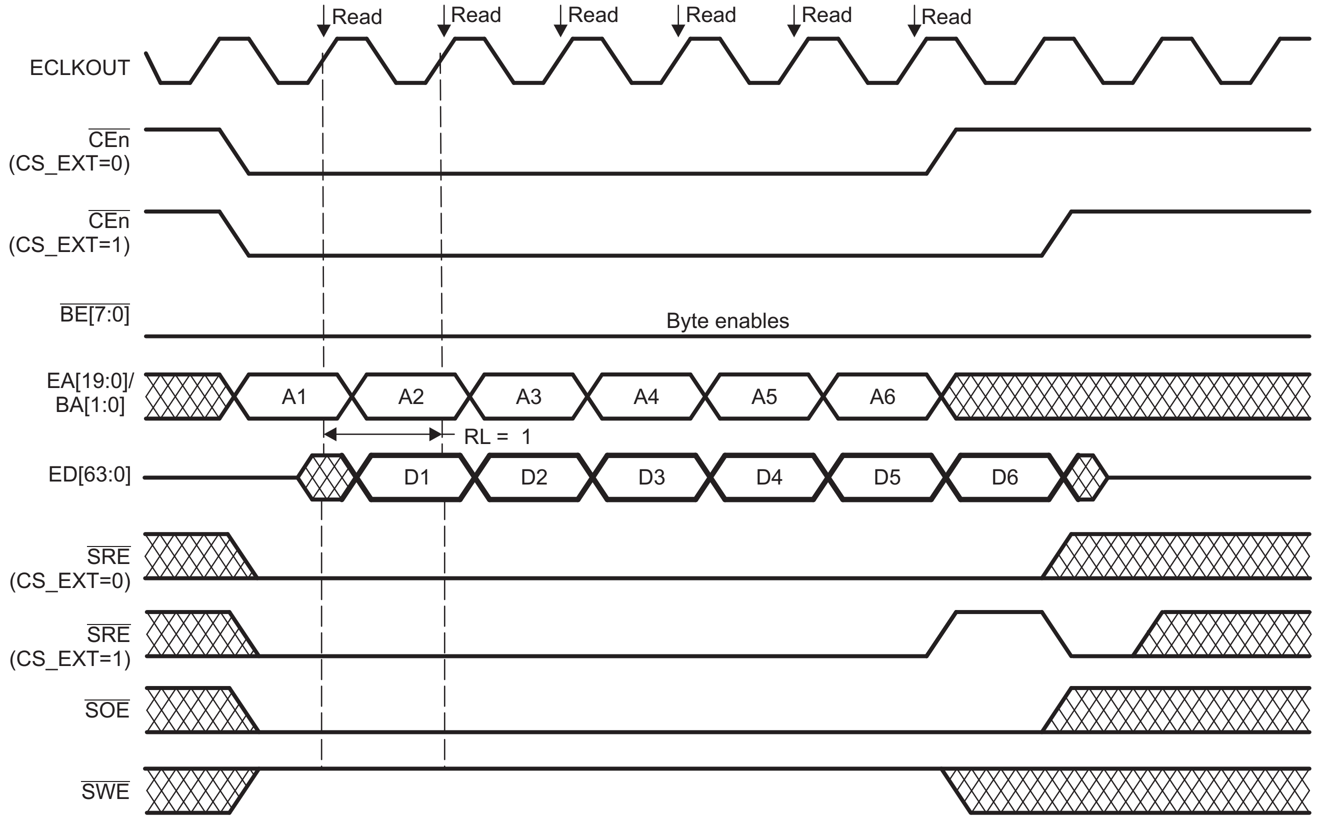
B The EDMA constant addressing mode should not be used when using the EDMA to move data to/from FIFOs connected to the EMIF. For more information, see Section 9, *EDMA Programming Considerations*.

### 5.4.1 Standard Synchronous FIFO Read

Figure 22 shows a six-word read from a standard synchronous FIFO. The CE*n*CFG settings are:

- SSEL = 1; to configure chip select for synchronous memory
- R_LTNCY = 01b; 1 cycle read latency
- CE_EXT = 0; CEn goes inactive after the final command has been issued
- CE_EXT = 1; during reads, CEn goes active when SOE goes active and will stay active until SOE goes inactive
- R_ENABLE = 1; SADS/SRE signal acts as SRE signal

**Figure 22. Standard Synchronous FIFO Read Timing Diagram**

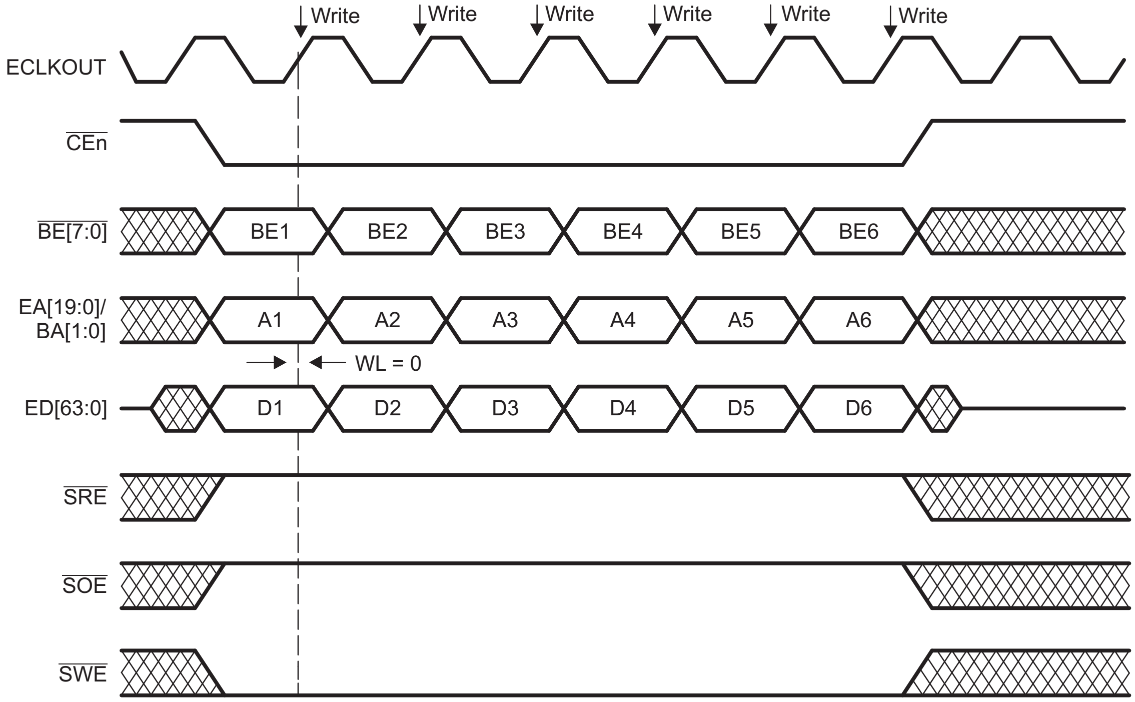


### 5.4.2 Standard Synchronous FIFO Write

Figure 23 shows a six-word write to a standard synchronous FIFO. The CE*n*CFG settings are:

- SSEL = 1; to configure chip select for synchronous memory
- CE_EXT = 0; $\overline{\text{CEn}}$ goes inactive after the final command has been issued
- W_LTNCY = 00b; 0 cycle write latency
- R_ENABLE = 1b; $\overline{\text{SADS}}/\overline{\text{SRE}}$ signal acts as $\overline{\text{SRE}}$ signal

**Figure 23. Standard Synchronous FIFO Write Timing Diagram**



## 6 Turnaround Time

To avoid bus contention, a programmable turnaround time is inserted between back-to-back accesses to the same or different CE spaces. Table 9 shows the turnaround time introduced by the EMIF on the data bus for various back-to-back accesses. The turnaround time is programmed through the TA bits of the Asynchronous Wait Cycle Configuration register (AWCC).

**Table 9. Turnaround Time**

| Previous Access | Next Access | Turnaround Time (Number of ECLKOUT Cycles) |
|---|---|---|
| Async read/write | Any read/write | TA + 2 |
| Sync read | Async read/write | TA + 2 |
| | Sync read to same chip select | 0 |
| | Sync read to different chip select | 1 |
| | Sync write | 1 |
| Sync write | Async read/write | TA + 2 |
| | Sync read | 1 |
| | Sync write | 0 |

## 7 Command FIFO and Scheduling

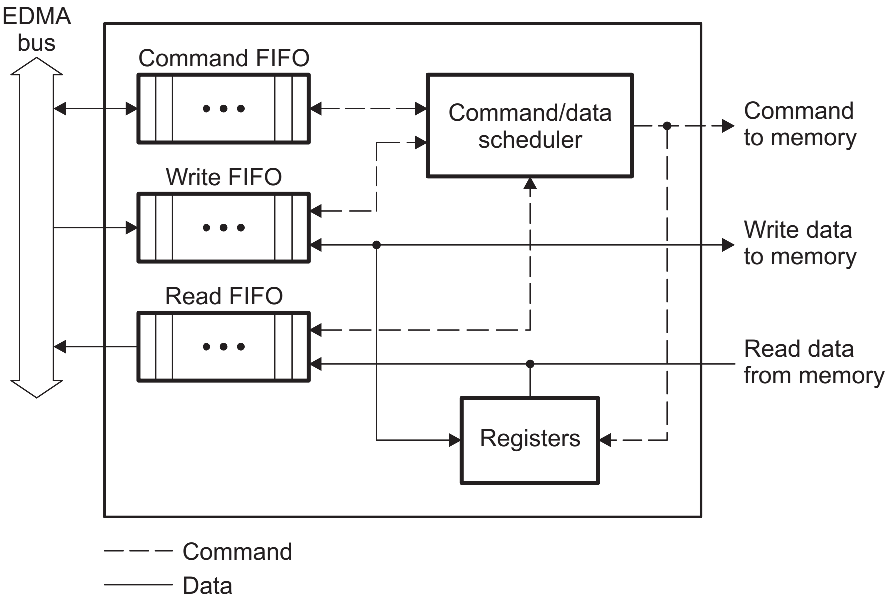
To move data efficiently from on-chip resources to external memory, the EMIF makes use of a command FIFO, a write FIFO, a read FIFO, and command and data schedulers. Table 10 describes the purpose of each FIFO.

**Table 10. EMIF FIFO Description**

| FIFO | Description | Depth (64-Bit Doublewords) |
|---|---|---|
| Command | Stores all commands coming from on-chip requestors | 7 |
| Write | Stores write data coming from on-chip requestors to memory | 11 |
| Read | Stores write data coming from memory to on-chip requestors | 15 |

Figure 24 shows the block diagram of the EMIF FIFOs. Commands, write data, and read data arrive at the EMIF parallel to each other. The EDMA bus is used to write and read data from external memory as well as internal memory-mapped registers.

**Figure 24. EMIF FIFO Block Diagram**



### 7.1 Command Ordering and Scheduling

The EMIF performs command re-ordering and scheduling in an attempt to achieve efficient transfers with maximum throughput. The goal is to maximize the utilization of external memory bandwidth.

The EMIF looks at all the commands stored in the Command FIFO to schedule commands to the external memory. For each master the EMIF reorders the commands based on the following rules:

- Selects the oldest command
- A read command is advanced before an older write command if the read is to a different block address (2048 bytes) and the read priority is equal to or greater than the write priority

**NOTE:** Most masters issue commands on a single priority level. Also, EDMA Transfer Controller Read and Write ports are considered different masters, and thus the above rule does not apply.

Once the commands from each master are reordered, the EMIF will have at most one pending read or write from each master. The EMIF then selects the highest priority read from the pending reads, and the highest priority write from the pending writes. If two or more commands have the highest priority, the EMIF selects the oldest command.

As a result, the EMIF may have a final read and a final write command. If the Read FIFO is not full, then the read command will be performed before the write command, otherwise the write command will be performed before the read command.

The following results from the above scheduling algorithm:

- All writes from a single master will complete in order
- All reads from a single master will complete in order
- From the same master, any read to the same location (or within 2048 bytes) as a previous write will complete in order

## 7.2 Command Starvation

The reordering and scheduling rules listed above may lead to command starvation, which is the prevention of certain commands from being processed by the EMIF. Command starvation can result when a continuous stream of high-priority read commands blocks a low-priority write command.

To avoid this condition, the EMIF momentarily raises the priority of the oldest command in the command FIFO after a set number of transfers have been made. The PRIO_RAISE field in the Burst Priority Register (BPRIO) sets the number of the transfers that must be made before the EMIF will raise the priority of the oldest command.

**NOTE:** Leaving the PRIO_RAISE bits at their default value (FFh) disables this feature of the EMIF. This means commands can stay in the command FIFO indefinitely. Therefore, these bits should be set to FEh immediately following reset to enable this feature with the highest level of allowable memory transfers. It is suggested that system-level prioritization be set to avoid placing high-bandwidth masters on the highest priority levels. These bits can be left as FEh unless advanced bandwidth/prioritization control is required.

## 7.3 Possible Race Condition

A race condition may exist when certain masters write data to the EMIF. For example, if master A passes a software message via a buffer in external memory and does not wait for indication that the write completes, when master B attempts to read the software message, the Master B Read may bypass the Master A write, and thus Master B may read stale data and therefore receive an incorrect message.

Some master peripherals (e.g., EDMA3 controller) will always wait for the write to complete before signaling an interrupt to the system, thus avoiding this race condition. For masters who do not have hardware guarantee of write-read ordering, it may be necessary to guarantee data ordering via software. If master A does not wait for indication that a write is complete, it must perform the following workaround:

1. Perform the required write.
2. Perform a dummy write to the EMIF module ID and revision register.
3. Perform a dummy read to the EMIF module ID and revision register.
4. Indicate to master B that the data is ready to be read after completion of the read in step 3. The completion of the read in step 3 ensures that the previous write was done.

For a list of the master peripherals that need this workaround, please refer to the device-specific data manual.

## 8 Resetting the EMIF

The EMIF can be reset through a hard reset or a soft reset. A hard reset resets the state machine, the FIFOs, and the internal registers. A soft reset only resets the state machine and the FIFOs. A soft reset does not reset the internal registers except for the interrupt registers. Register accesses cannot be performed while either reset is asserted.

The EMIF hard and soft reset are derived from device-level resets. C645x devices have several types of device-level resets: power-on reset, warm reset, max reset, system reset, and CPU reset. Table 11 shows the relationship between the device-level resets and the EMIF resets. For more information on the device-level resets, see the device-specific data manual.

**Table 11. Device and EMIF Reset Relationship**

| EMIF Reset | Effect | Initiated By: |
|---|---|---|
| Hard reset | Resets control logic and all EMIF registers | Power on reset<br>Warm reset<br>Max reset |
| Soft reset | Resets control logic and interrupt registers | System reset<br>CPU reset |

## 9 EDMA Programming Considerations

As documented in the *TMS320C645x DSP Enhanced DMA (EDMA3) Controller User's Guide* (SPRU966), the EDMA includes two types of addressing modes: increment mode (INCR) and constant addressing mode (CONST). One of these modes must be selected for the source address mode and the destination address mode (SAM and DAM in the channel options parameter (OPT)).

Even though the EDMA supports constant addressing mode configurations for SAM and DAM, the EMIF does not fully support constant addressing mode. Attempts to perform writes to the EMIF with constant addressing mode (DAM == CONST) will result in the destination address incrementing within a modulo-64 byte address range; i.e., the data will be written from the destination address to the destination address plus 63 bytes repeatedly, until the transfer count programmed for the EDMA transfer is exhausted.

In order to avoid the issues described above with reads/writes to EMIF when EDMA is configured for constant addressing mode, it is required that the EDMA always be programmed with SAM and DAM in increment mode (INCR). The ACNT and BCNT values, along with proper indexing, can be used to mimic constant addressing mode. This is addressed in the first and second recommendations/use cases below.

No special requirements exist for addresses accessed by the EDMA in increment mode or for single-word accesses by the CPU or DMA.

Two approaches are presented below for interfacing with FIFOs using the EMIF. Each approach describes the necessary EMIF connection to the FIFO and the configuration of the EDMA. In general, these approaches are numbered in order of preference.

The system designer should attempt to design a system/board/ASIC memory map such that a relatively large memory range is devoted to a FIFO. In this way, the EDMA can be programmed in increment mode for a given DMA transfer, and the transfer from start to finish will reside in the memory range dedicated to a given FIFO.

**Approach 1**: Address space dedicated to the FIFO is greater than or equal to the largest expected EDMA transfer. No performance hit since EDMA ACNT is not artificially constrained.

- The FIFO should be aligned on a 64-byte boundary in EMIF address space. System memory map and glue logic should be implemented to use EMIF most-significant address bits to decode FIFO address and select FIFO.
- The EDMA transfer starting address should match the FIFO's base address.
- Use increment addressing mode (INCR) for source or destination with ACNT = transfer size.
- Use SBIDX or DBIDX of 0 such that the next EDMA transfer will also begin at the base address of the FIFO (assuming BCNT and/or CCNT are greater than 1).

For example, if the largest possible DMA transfer to/from a FIFO interfaced to EMIF is 1024 bytes, then a memory range of at least 1024 bytes should be devoted to the FIFO in the EMIF's memory map. The system glue logic should use the chip enable signals and logical address bits 10 and above if multiple FIFOs reside in the chip enable space. The EDMA transfer can be set with ACNT = 1024 bytes, BCNT = X, CCNT = Y, and an EDMA synchronization type of A-synchronized. The index for the FIFO side of the transfer (either SRC or DST) should be set to 0 such that the same address is used for the next DMA trigger.

**Approach 2**: If the amount of space dedicated to the FIFO is less than the largest expected EDMA, then ACNT and BCNT value with appropriate indexing can be used to control access to the FIFO. This will result in a potential performance impact depending on the size of ACNT.

- The FIFO should be aligned on 64-byte boundary in EMIF address space. System memory map and glue logic should be implemented to use EMIF most-significant address bits to decode FIFO address and select FIFO.
- The EDMA transfer starting address should match the FIFO's base address.
- The EDMA size must be broken into:
  - ACNT x BCNT = transfer size.
  - ACNT must be less than or equal to the address space dedicated to the FIFO.
  - If the EDMA transfer size is not a multiple of ACNT, then two EDMA channels must be used. Completion of the first channel can chain to the second channel, where the second channel is used to transfer the remaining data.
  - Use SBIDX or DBIDX of 0 such that the next EDMA transfer will also begin at the base address of the FIFO (assuming BCNT and/or CCNT are greater than 1).

For example, if the desired DMA transfer size to/from a FIFO interfaced to EMIF is 1024 bytes, but the memory range dedicated to the FIFO is only 64 bytes, then the EDMA transfer must be broken into a 2-D transfer, with ACNT = 64 bytes, BCNT = 16, CCNT = X, and an EDMA synchronization type of AB-synchronized. The index for the FIFO side of the transfer (either SRC or DST) should be set to 0 such that the same address is used for the next DMA trigger.

With the same example, if the desired DMA transfer size is 1028-bytes, an additional channel with ACNT = 4 bytes must be used. Completion of the first channel needs to chain trigger the second channel.

## 10 Hold Interface

The hold interface allows an external master to request access to the devices connected to $\overline{\text{CE[5:2]}}$. The $\overline{\text{HOLD}}$, $\overline{\text{HOLDA}}$, and BUSREQ signals are used to interface with the external master.

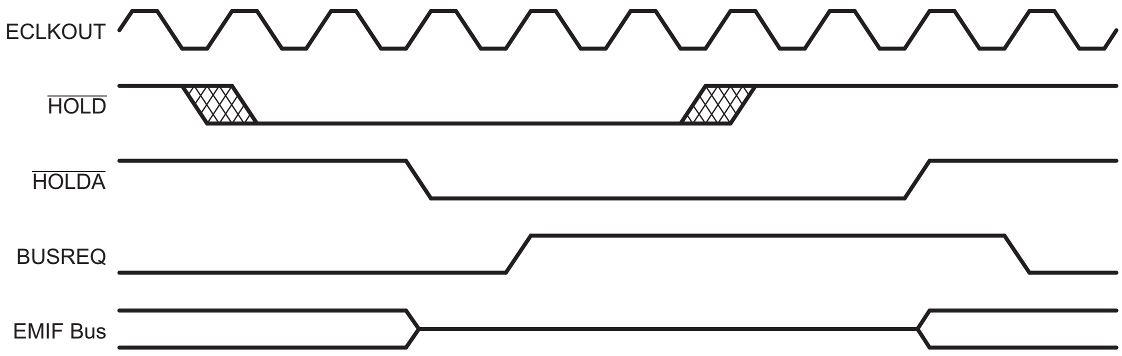
The $\overline{\text{HOLD}}$ signal is an asynchronous input that is driven low by an external master that needs to gain access to the EMIF memory bus. The asynchronous $\overline{\text{HOLD}}$ input is synchronized internally to ECLKIN. When $\overline{\text{HOLD}}$ is asserted and the synchronization of this input to ECLKIN is complete, the EMIF stops driving the external pins at the earliest possible moment. This may require that the current active transactions on the bus are completed. Upon completion of active transactions, the EMIF places all output pins in a high-impedance state except $\overline{\text{HOLDA}}$, BUSREQ, and ECLKOUT. The external device requesting the bus must keep the $\overline{\text{HOLD}}$ pin asserted as long as it requires access to the bus.

The EMIF drives the $\overline{\text{HOLDA}}$ signal low after it has placed all its output pins, except for the pins mentioned above, in a high-impedance state. During external master operation, the external master must drive all pins, except for the pins mentioned above, to an operational state.

The EMIF drives the BUSREQ signal high when any command to the EMIF is received or pending. The BUSREQ signal is asserted only when the $\overline{\text{HOLD}}$ pin is low. The external master can use the BUSREQ signal to release the control of the bus.

Active $\overline{\text{HOLD}}$ is treated as the highest priority after the completion of any in flight data to/from the external device.

**Figure 25. Hold Interface**



## 11 Emulation Halt Operation

The EMIF continues operating during emulation halts. Emulator accesses through the EMIF can work differently than the way the actual device works during EMIF accesses. This discrepancy can cause start-up penalties after a halt operation.

## 12 EMIF Registers

Control of the EMIF and the memory interfaces it supports is maintained through memory-mapped registers within the EMIF. Table 12 lists the memory-mapped registers of the EMIF. For the memory address of these registers, see the device-specific data manual.

**Table 12. EMIF Registers**

| Offset | Acronym | Register Name | Section |
|---|---|---|---|
| 0000 | MIDR | Module ID and Revision Register | Section 12.1 |
| 0004 | STAT | Status Register | Section 12.2 |
| 0020 | BPRIO | Burst Priority Register | Section 12.3 |
| 0080-008C | CE*n*CFG | CE*n* Configuration Registers (SSEL = 0) | Section 12.4 |
| 0080-008C | CE*n*CFG | CE*n* Configuration Registers (SSEL = 1) | Section 12.5 |
| 00A0 | AWCC | Asynchronous Wait Cycle Configuration Register | Section 12.6 |
| 00C0 | INTRAW | Interrupt RAW Register | Section 12.7 |
| 00C4 | INTMSK | Interrupt Masked Register | Section 12.8 |
| 00C8 | INTMSKSET | Interrupt Mask Set Register | Section 12.9 |
| 00CC | INTMSKCLR | Interrupt Mask Clear Register | Section 12.10 |

### 12.1 Module ID and Revision Register (MIDR)

This register reflects the latest changes made to the memory controller. The Module ID and Revision Register (MIDR) is shown in Figure 26 and described in Table 13.

**Figure 26. Module ID and Revision Register (MIDR)**

| 31 30 | 29 16 |
|---|---|
| Reserved | MOD_ID |
| R-0x0 | R-0x0032 |

| 15 8 | 7 0 |
|---|---|
| MJ_REV | MN_REV |
| R-0x03 | R-0x11 |

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

**Table 13. Module ID and Revision Register (MIDR) Field Descriptions**

| Bit | Field | Value | Description |
|---|---|---|---|
| 31-30 | Reserved | 0 | Reserved. |
| 29-16 | MOD_ID | | EMIF module ID. |
| 15-8 | MJ_REV | | Major revision. |
| 7-0 | MN_REV | | Minor revision. |

## *12.2 Status Register (STAT)*

This register reflects the configuration of the EMIF. The Status Register (STAT) is shown in Figure 27 and described in Table 14.

**Figure 27. Status Register (STAT)**

| 31 | 30 – 16 |
|---|---|
| BE | Reserved |
| R-0x0 | R-0x4000 |

| 15 – 0 |
|---|
| Reserved |
| R-0x0000 |

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

**Table 14. Status Register (STAT) Field Descriptions**

| Bit | Field | Value | Description |
|---|---|---|---|
| 31 | BE | | Big Endian. Reflects whether the EMIF is configured for big or little endian mode. |
| | | 1 | EMIF is configured for big endian mode. |
| | | 0 | EMIF is configured for little endian mode. |
| 30-0 | Reserved | | Reserved. |

## 12.3 Burst Priority Register (BPRIO)

This register is used to temporarily raise priority of old commands. The Burst Priority Register (BPRIO) is shown in Figure 28 and described in Table 15.

**Figure 28. Burst Priority Register (BPRIO)**

| 31–16 |
|---|
| Reserved |
| R-0x0 |

| 15–8 | 7–0 |
|---|---|
| Reserved | PRIO_RAISE |
| R-0x0 | RW-0xFF |

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

**Table 15. Burst Priority Register (BPRIO) Field Descriptions**

| Bit | Field | Value | Description |
|---|---|---|---|
| 31-8 | Reserved | | Reserved. |
| 7-0 | PRIO_RAISE | | Number of memory transfers after which the EMIF will elevate the priority of the oldest command in the command FIFO. Setting this field to FFh disables this feature, thereby allowing old commands to stay in the FIFO indefinitely. |
| | | 0 | 1 memory transfer. |
| | | 1 | 2 memory transfers. |
| | | 2 | 3 memory transfers. |
| | | 3-FEh | 4-FFh memory transfers. |
| | | FFh | Feature disabled, commands can stay in command FIFO indefinitely. |

## 12.4 CE*n* Configuration Registers (CE*n*CFG) if SSEL = 0

These registers select the access type (asynchronous or synchronous) and configure the access parameters for the CE*n* space. The contents of the CE*n* Configuration Registers (CE*n*CFG) are interpreted according to the SSEL bit. Figure 29 and Table 16 describe the CE*n*CFG registers when SSEL = 0.

**Figure 29. CE*n* Configuration Registers (CE*n*CFG) if SSEL = 0**

| 31 | 30 | 29 | 28 | 27 24 | 23 18 | 17 16 |
|---|---|---|---|---|---|---|
| SSEL | SS | BWEM | AE | W_SETUP | W_STROBE | W_HOLD |
| RW-0x0 | RW-0x0 | RW-0x0 | RW-0x0 | RW-0xF | RW-0x3F | RW-0x7 |

| 15 | 14 11 | 10 5 | 4 2 | 1 0 |
|---|---|---|---|---|
| W_HOLD | R_SETUP | R_STROBE | R_HOLD | ASIZE |
| RW-0x7 | RW-0xF | RW-0x3F | RW-0x7 | RW-0x0 |

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

**Table 16. CE*n* Configuration Registers (CE*n*CFG) if SSEL = 0 Field Descriptions**

| Bit | Field | Value | Description |
|---|---|---|---|
| 31 | SSEL | | Synchronous/asynchronous memory select. This bit specifies whether CE*n* is configured for synchronous or asynchronous memory accesses. The other fields in this table define the bits in the CE*n*CFG when SSEL is cleared to 0 (asynchronous memory mode). |
| | | 1 | Synchronous memory mode |
| | | 0 | Asynchronous memory mode |
| 30 | SS | | Select Strobe mode enable. When set to 1, the CE*n* pin will have read and write strobe timing. |
| | | 1 | Select strobe mode enabled |
| | | 0 | Select strobe mode disabled |
| 29 | BWEM | | WE Strobe mode enable. When set to 1, the BE[7:0] output pins will act as active low byte write enables when accessing CE*n* space. When cleared to 0, the BE[7:0] output pins will act as active low byte enables when accessing CE*n*. |
| | | 1 | WE Strobe mode enabled |
| | | 0 | WE Strobe mode disabled |
| 28 | AE | | Asynchronous ready input enable. Set to 1 to enable the asynchronous ready (ARDY) input pin during accesses to the CE*n* space. When enabled, the ARDY pin can be used to extend the strobe period during asynchronous accesses. |
| | | 1 | ARDY pin enabled |
| | | 0 | ARDY pin disabled |
| 27-24 | W_SETUP | | Write setup width. Number of ECLKOUT cycles from EA[19:0], BA[1:0], D[63:0], BE[7:0], and CE*n* being set to $\overline{\text{AWE}}$ asserted, minus one cycle. |
| 23-18 | W_STROBE | | Write strobe width. Number of ECLKOUT cycles for which $\overline{\text{AWE}}$ is held active, minus one cycle. |
| 17-15 | W_HOLD | | Write hold width. Number of ECLKOUT cycles for which EA[19:0], BA[1:0], D[63:0], BE[7:0], and CE*n* are held after $\overline{\text{AWE}}$ has been deasserted, minus one cycle. |
| 14-11 | R_SETUP | | Read setup width. Number of ECLKOUT cycles from EA[19:0], BA[1:0], BE[7:0], and CE*n* being set to $\overline{\text{AOE}}$ asserted, minus one cycle. |
| 10-5 | R_STROBE | | Read strobe width. Number of ECLKOUT cycles for which $\overline{\text{AOE}}$ is held active, minus one cycle. |
| 4-2 | R_HOLD | | Read hold width. Number of ECLKOUT cycles for which EA[19:0], BA[1:0], BE[7:0], and CE*n* are held after $\overline{\text{AOE}}$ has been deasserted, minus one cycle. |
| 1-0 | ASIZE | | Asynchronous Memory Size. Defines the width of the asynchronous device's data bus. |
| | | 0 | 8-bit data bus. |
| | | 1 | 16-bit data bus. |
| | | 2 | 32-bit data bus. |
| | | 3 | 64-bit data bus. |

## 12.5 CE*n* Configuration Registers (CE*n*CFG) Field Descriptions if SSEL = 1

These registers select the access type (asynchronous or synchronous) and configure the access parameters for the CE*n* space. The contents of the CE*n* Configuration Registers (CE*n*CFG) are interpreted according to the SSEL bit. Figure 30 and Table 17 describe the CE*n*CFG registers when SSEL = 1.

**Figure 30. CE*n* Configuration Registers (CE*n*CFG) Field Descriptions if SSEL = 1**

| 31 | 16 |
|---|---|
| SSEL | Reserved |
| RW-0x0 | R-0x0 |

| 15 11 | 10 | 9 | 8 | 7 6 | 5 4 | 3 2 | 1 0 |
|---|---|---|---|---|---|---|---|
| Reserved | RD_BE_EN | CE_EXT | R_ENABLE | W_LTNCY | Reserved | R_LTNCY | SBSIZE |
| R-0x0 | RW-0x0 | RW-0x0 | RW-0x0 | RW-0x0 | R-0x0 | RW-0x0 | RW-0x0 |

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

**Table 17. CE*n* Configuration Registers (CE*n*CFG) Field Descriptions if SSEL = 1**

| Bit | Field | Value | Description |
|---|---|---|---|
| 31 | SSEL | | Synchronous/asynchronous memory select. This bit specifies whether CE*n* is configured for synchronous or asynchronous memory accesses. The other fields in this table define the bits in the CE*n*CFG when SSEL is set to 1 (synchronous memory mode). |
| | | 1 | Synchronous memory mode |
| | | 0 | Asynchronous memory mode |
| 30-11 | Reserved | | Reserved. |
| 10 | RD_BE_EN | | Read Byte Enable. If set to 1, the byte enable pins (BE[7:0]) are driven during asynchronous memory reads. If cleared to 0, BE[7:0] stay high during synchronous memory reads. Not supported for R_LTNCY = 0. |
| | | 1 | Byte enables are driven during synchronous memory reads. |
| | | 0 | Byte enables stay high during synchronous memory reads. |
| 9 | CE_EXT | | Synchronous Memory Chip Enable Extend. Defines the behavior of the CE*n* pin during synchronous accesses. |
| | | 1 | CE*n* goes active when $\overline{\text{SOE}}$ goes active and will stay active until $\overline{\text{SOE}}$ goes inactive. The timing of $\overline{\text{SOE}}$ is controlled by R_LTNCY. Used for synchronous FIFO interfaces where $\overline{\text{CE}}$ gates $\overline{\text{SOE}}$. |
| | | 0 | CE*n* goes inactive after final command has been issued. |
| 8 | R_ENABLE | | Synchronous Memory Read Enable Mode. Defines the behavior of the $\overline{\text{SADS/SRE}}$ signal during synchronous accesses. |
| | | 1 | The $\overline{\text{SADS/SRE}}$ pin acts as $\overline{\text{SRE}}$. The $\overline{\text{SRE}}$ pin goes low only for reads. No deselect command is issued. Used for FIFO interfaces. |
| | | 0 | The $\overline{\text{SADS/SRE}}$ pin acts as the $\overline{\text{SADS}}$ pin. The $\overline{\text{SADS}}$ pin goes active low for reads and writes. A deselect command is issued by driving $\overline{\text{SADS}}$ active high after a command if no new command is pending for the same CE space. Used for SBSRAM and ZBT devices. |
| 7-6 | W_LTNCY | | Synchronous Memory Write Latency. Defines the synchronous device's write latency in EMIF clock cycles. |
| | | 0 | 0 cycle write latency |
| | | 1 | 1 cycle write latency |
| | | 2 | 2 cycle write latency |
| | | 3 | 3 cycle write latency |
| 5-4 | Reserved | | Reserved. |

### Table 17. CE*n* Configuration Registers (CE*n*CFG) Field Descriptions if SSEL = 1 (continued)

| Bit | Field | Value | Description |
|---|---|---|---|
| 3-2 | R_LTNCY | | Synchronous Memory Read Latency. Defines the synchronous device's read latency in EMIF clock cycles. Read latency of 0 is not supported. |
| | | 1 | 1 cycle read latency |
| | | 2 | 2 cycle read latency |
| | | 3 | 3 cycle read latency |
| 1-0 | SBSIZE | | Synchronous Memory Device Size. Defines the width of the synchronous device's data bus. |
| | | 0 | 8-bit data bus. |
| | | 1 | 16-bit data bus. |
| | | 2 | 32-bit data bus. |
| | | 3 | 64-bit data bus. |

## 12.6 Asynchronous Wait Cycle Configuration Register (AWCC)

The Asynchronous Wait Cycle Configuration Register (AWCC) controls asynchronous memory access features such as the turn-around time between accesses and the asynchronous ready pin (ARDY) polarity. AWCC is shown in Figure 31 and described in Table 18.

**Figure 31. Asynchronous Wait Cycle Configuration Register (AWCC)**

<table>
<tr><td>31</td><td>30</td><td>29</td><td>16</td></tr>
<tr><td>Rsvd</td><td>AP</td><td colspan="2">Reserved</td></tr>
<tr><td>R-0x0</td><td>RW-0x1</td><td colspan="2">R-0x0</td></tr>
</table>

<table>
<tr><td>15</td><td>11</td><td>10</td><td>8</td><td>7</td><td>0</td></tr>
<tr><td colspan="2">Reserved</td><td colspan="2">TA</td><td colspan="2">MAX_EXT_WAIT</td></tr>
<tr><td colspan="2">R-0x0</td><td colspan="2">RW-0x3</td><td colspan="2">RW-0x80</td></tr>
</table>

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

**Table 18. Asynchronous Wait Cycle Configuration Register (AWCC) Field Descriptions**

| Bit | Field | Value | Description |
|---|---|---|---|
| 31 | Reserved | | Reserved. |
| 30 | AP | | Asynchronous ready (ARDY) pin polarity. Defines the polarity of the ARDY pin. |
| | | 1 | ARDY pin is active-high (strobe period extended when ARDY is high) |
| | | 0 | ARDY pin is active-low (strobe period extended when ARDY is low) |
| 29-11 | Reserved | | Reserved. |
| 10-8 | TA | | Turn Around cycles. Number of ECLKOUT cycles between the end of one asynchronous memory access and the start of another asynchronous memory access, minus two cycles. |
| 7-0 | MAX_EXT_WAIT | | Maximum Extended Wait cycles. The value in this field defines the number of 16 EMIF cycle periods the EMIF will wait for an extended asynchronous cycle before the cycle is terminated. |

### 12.7 Interrupt RAW Register (INTRAW)

This register displays the status of the EMIF interrupt regardless of whether or not the interrupt is enabled. The interrupt RAW register (INTRAW) is shown in Figure 32 and described in Table 19.

**Figure 32. Interrupt RAW Register (INTRAW)**

| 31 – 16 |
|---|
| Reserved |
| R-0x0 |

| 15 – 1 | 0 |
|---|---|
| Reserved | AT |
| R-0x0 | RW-0x0 |

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

**Table 19. Interrupt RAW Register (INTRAW) Field Descriptions**

| Bit | Field | Value | Description |
|---|---|---|---|
| 31-1 | Reserved | | Reserved. |
| 0 | AT | | Asynchronous timeout interrupt. Set to 1 by the EMIF to indicate that the ARDY pin did not go inactive within the number of cycles defined by the MAX_EXT_WAIT field in the Async Wait Cycle Configuration register (AWCC). Writing a 1 will clear this bit as well as the AT_MASKED bit in the Interrupt Masked register (INTMSK). Writing a 0 has no effect. |
| | | 1 | An asynchronous access timeout has occurred. |
| | | 0 | An asynchronous access timeout has not occurred. |

## 12.8 Interrupt Masked Register (INTMSK)

This register displays the status of the EMIF interrupt only when the interrupt is enabled. The interrupt masked register (INTMSK) is shown in Figure 33 and described in Table 20.

**Figure 33. Interrupt Masked Register (INTMSK)**

| 31 | | 16 |
|---|---|---|
| | Reserved | |
| | R-0x0 | |

| 15 | 1 | 0 |
|---|---|---|
| Reserved | | AT_<br>MASK-<br>ED |
| R-0x0 | | RW-<br>0x0 |

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

**Table 20. Interrupt Masked Register (INTMSK) Field Descriptions**

| Bit | Field | Value | Description |
|---|---|---|---|
| 31-1 | Reserved | | Reserved. |
| 0 | AT_MASKED | | Asynchronous timeout interrupt masked. Set to 1 by the EMIF to indicate that the ARDY pin did not go inactive within the number of cycles defined by the MAX_EXT_WAIT field in the Asynchronous Wait Cycle Configuration register (AWCC) only if the AT_MASK_SET bit in the Interrupt Mask Set register is set to 1. Writing a 1 will clear this bit as well as the AT bit in the Interrupt Raw register (INTRAW). Writing a 0 has no effect. |
| | | 1 | An asynchronous access timeout has occurred. |
| | | 0 | An asynchronous access timeout has not occurred. |

## 12.9 Interrupt Mask Set Register (INTMSKSET)

The interrupt mask set register (INTMSKSET) enables the EMIF interrupt. It is shown in Figure 34 and described in Table 21.

**Figure 34. Interrupt Mask Set Register (INTMSKSET)**

| 31 | 16 |
|---|---|
| Reserved | |
| R-0x0 | |

| 15 | 1 | 0 |
|---|---|---|
| Reserved | | AT_<br>MASK_<br>SET |
| R-0x0 | | RW-0x0 |

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

**Table 21. Interrupt Mask Set Register (INTMSKSET) Field Descriptions**

| Bit | Field | Value | Description |
|---|---|---|---|
| 31-1 | Reserved | | Reserved. |
| 0 | AT_MASK_SET | | Mask set for AT_MASKED bit in the Interrupt Masked register. Writing a 1 to this bit will enable the interrupt and set this bit as well as the AT_MASK_CLR bit in the Interrupt Mask Set register. Writing a 0 has no effect. |
| | | 1 | The asynchronous access timeout interrupt is enabled. |
| | | 0 | The asynchronous access timeout interrupt is disabled. |

## 12.10 Interrupt Mask Clear Register (INTMSKCLR)

The interrupt mask clear register (INTMSKCLR) disables the EMIF interrupt. It is shown in Figure 35 and described in Table 22.

**Figure 35. Interrupt Mask Clear Register (INTMSKCLR)**

| 31 | 16 |
|---|---|
| Reserved | |
| R-0x0 | |

| 15 1 | 0 |
|---|---|
| Reserved | AT_<br>MASK_<br>CLR |
| R-0x0 | RW-0x0 |

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

**Table 22. Interrupt Mask Clear Register (INTMSKCLR) Field Descriptions**

| Bit | Field | Value | Description |
|---|---|---|---|
| 31-1 | Reserved | | Reserved. |
| 0 | AT_MASK_CLR | | Mask clear for AT_MASKED bit in the Interrupt Masked register. Writing a 1 to this bit will disable the interrupt and clear this bit as well as the AT_MASK_SET bit in the Interrupt Mask Set register. Writing a 0 has no effect. |
| | | 1 | The asynchronous access timeout interrupt is enabled. |
| | | 0 | The asynchronous access timeout interrupt is disabled. |

## Appendix A Revision History

This revision history highlights the technical changes made to the document in this revision.

**Scope:** Applicable updates to the C64x device family, specifically relating to the TMS320C645x devices, have been incorporated.

Table 23. C645x Revision History

| See | Additions/Modifications/Deletions |
|---|---|
| Section 2 | Modified ECLKOUT text |
| Table 3 | Modified ECLKOUT Description |
| Table 8 | Modified ECLKOUT Signal Function |

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

| **Products** | | **Applications** | |
|---|---|---|---|
| Amplifiers | amplifier.ti.com | Audio | www.ti.com/audio |
| Data Converters | dataconverter.ti.com | Automotive | www.ti.com/automotive |
| DLP® Products | www.dlp.com | Communications and Telecom | www.ti.com/communications |
| DSP | dsp.ti.com | Computers and Peripherals | www.ti.com/computers |
| Clocks and Timers | www.ti.com/clocks | Consumer Electronics | www.ti.com/consumer-apps |
| Interface | interface.ti.com | Energy | www.ti.com/energy |
| Logic | logic.ti.com | Industrial | www.ti.com/industrial |
| Power Mgmt | power.ti.com | Medical | www.ti.com/medical |
| Microcontrollers | microcontroller.ti.com | Security | www.ti.com/security |
| RFID | www.ti-rfid.com | Space, Avionics & Defense | www.ti.com/space-avionics-defense |
| RF/IF and ZigBee® Solutions | www.ti.com/lprf | Video and Imaging | www.ti.com/video |
| | | Wireless | www.ti.com/wireless-apps |

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265