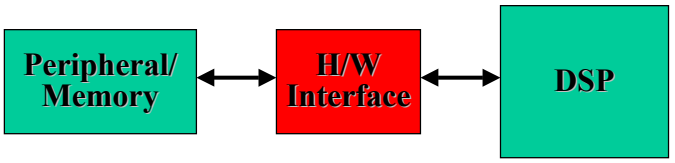
TMS320C6000 FAMILY: EMIF

TMS320C6000 FAMILY: EMIF

- **Introduction.**
  - Characteristics, signals, memory map, alignment.
  - Configuration registers.
  - Types of interface.
- **Asynchronous interface.**
  - Introduction, waveforms.
  - Case Study I: Peripheral connection.
  - Case Study II: Memory connection.
- **Interface with synchronous static memories.**
  - Synchronous static memories.
  - Interface description.
  - Example.
- **Interface with synchronous dynamic memories.**
  - SDRAM memories.
  - Interface description.
  - Example.

Need for an EMIF

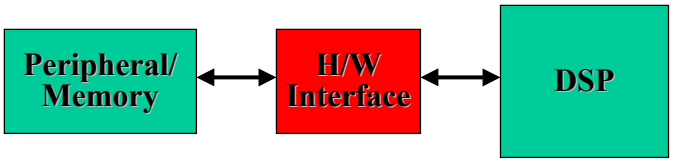
- Traditional DSP (with no EMIF):



- When interfacing a slow peripheral/memory to a fast DSP, some hardware interface is required.
- This hardware interface requires fast components in order to keep up with the DSP.

Need for an EMIF (II)

- Traditional DSP (with no EMIF):



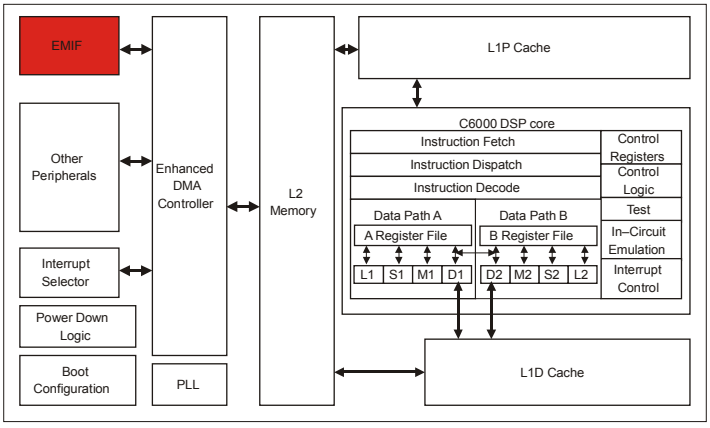
- Drawback of the hardware interface:
  - High cost (additional components).
  - Power consumption.
  - Difficult to debug.
  - Cannot be upgraded.
  - Prone to errors.

The EMIF

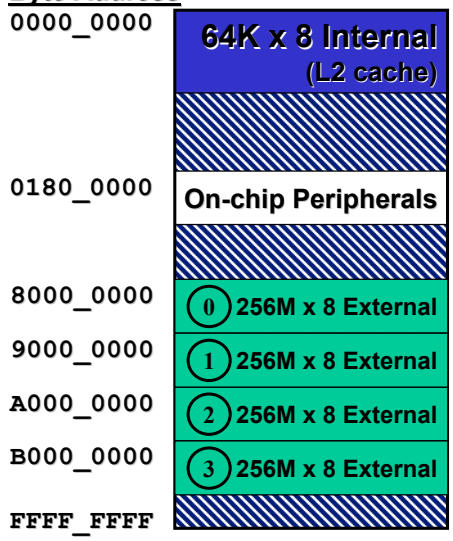
- The EMIF supports a glueless interface to several external devices, including:
  - Synchronous burst SRAM (SBSRAM).
  - Synchronous DRAM (SDRAM).
  - Asynchronous devices, including SRAM, ROM and FIFO's.
  - An external shared-memory device.
- For more information on different memory types see spra631.pdf
- A detailed description for each memory type interface can be found in:
  - Asynchronous SRAM: spra542a.pdf
  - Synchronous burst SRAM: spra533.pdf
  - Synchronous DRAM: spra433b.pdf

The EMIF

- The C621x/C671x services requests of the external bus from the requestors:
  - On-chip Enhanced Direct Memory Access (EDMA) controller.
  - External shared-memory device controller.



MEMORY MAP



External Memory

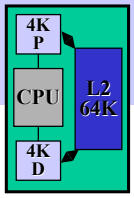
- Async (SRAM, ROM, etc.)
- Sync (SBSRAM, SDRAM)

Internal Memory

- Unified (data or prog)
- 4 blocks - each can be RAM or cache

Level 1 Cache

- 4KB Program
- 4KB Data
- Not in map



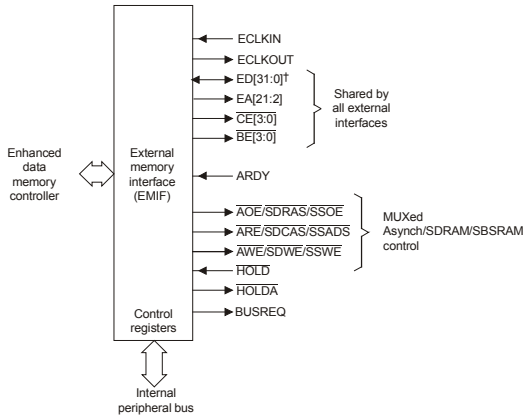
MEMORY MAP (II)

Description	Origin	Length
Internal RAM (L2) mem	**0x00000000**	**0x00010000**
EMIF control regs	0x01800000	0x00000024
Cache configuration reg	0x01840000	0x00000004
L2 base addr & count regs	0x01844000	0x00000020
L1 base addr & count regs	0x01844020	0x00000020
L2 flush & clean regs	0x01845000	0x00000008
CE0 mem attribute regs	0x01848200	0x00000010
CE1 mem attribute regs	0x01848240	0x00000010
CE2 mem attribute regs	0x01848280	0x00000010
CE3 mem attribute regs	0x018482c0	0x00000010
HPI control reg	0x01880000	0x00000004
McBSP0 regs	0x018c0000	0x00000028
McBSP1 regs	0x01900000	0x00000028
Timer0 regs	0x01940000	0x0000000c
Timer1 regs	0x01980000	0x0000000c
Interrupt selector regs	0x019c0000	0x0000000c
EDMA parameter RAM	0x01a00000	0x00000800
EDMA control regs	0x01a0ffe0	0x00000020
QDMA regs	0x02000000	0x00000014
QDMA pseudo-regs	0x02000020	0x00000014
McBSP0 data	0x30000000	0x04000000
McBSP1 data	0x34000000	0x04000000
CE0, 256 MBytes	**0x80000000**	**0x10000000**
CE1, 256 MBytes	**0x90000000**	**0x10000000**
CE2, 256 MBytes	**0xA0000000**	**0x10000000**
CE3, 256 MBytes	**0xB0000000**	**0x10000000**

C6211/C6711 EMIF Features

Features	C621x / C671x
Bus Width	32
# Memory Spaces	4
Addressable Space (Mbytes)	512
Synchronous Clocking	Independent ECLKIN
Width Support	8/16/32
Supported Memory Type at CE1	All types
Control Signals	Mixed all control signals
Synchronous memory in system	Both SDRAM and SBSRAM
Additional registers	SDEXT
PDT Support	No
ROM/Flash	Yes
Asynchronous memory I/O	Yes
Pipeline SBSRAM	Yes

C6713 EMIF Signals



- **Clock signals:**
  - **ECLKIN**
  - **ECLKOUT**
- **Data bus: ED[31:0]**
- **Address bus: EA[21:2]**
- **Byte enable: BE[3:0]**
- **Control signals:**
  - **Asynchronous ready.**
  - **Memory type depending.**
- **Bus arbitration:**
  - **Hold**
  - **Hold acknowledge.**
  - **Bus request.**

- **For a description of the signals see: \Links\signals.pdf**

C6211/C6711 EMIF Configuration

- **The following need to be configured when interfacing the DSP to an external device using the EMIF:**

(1) Memory space control registers (software):

These registers describe the type and timing of the external memory to be used.

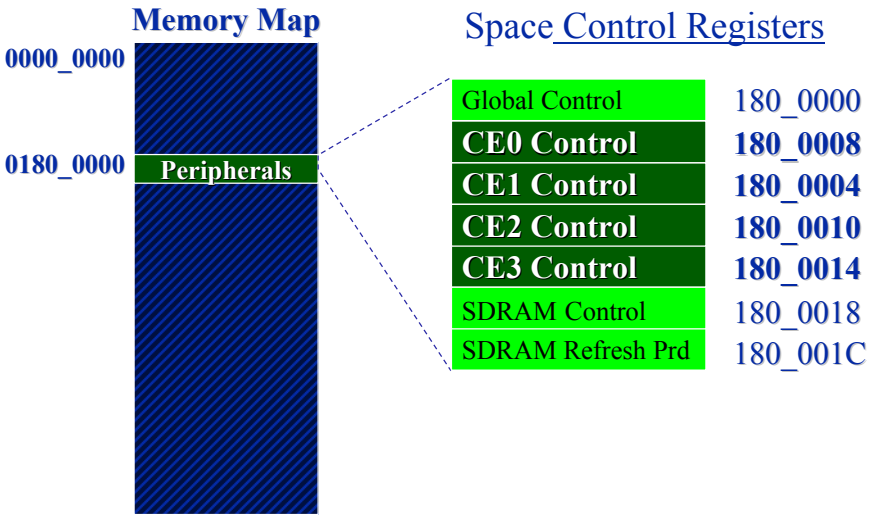
(2) EMIF chip enable (hardware):

There are four chip enable (CE0, CE1, CE2 and CE3) that are used when accessing a specific memory location (e.g. if you try to access memory 0x9000 0000 then CE1 will be activated, see next slide).

C6211/C6711 EMIF Memory Spaces

Memory Block Description	Block Size (Bytes)	HEX Address Range
Internal RAM (L2)	64K	0000 0000 - 0000 FFFF
Reserved	24M-64K	0001 0000 - 017F FFFF
EMIF Registers	256k	0180 0000 - 0183 FFFF
L2 Registers	256k	0184 0000 - 0187 FFFF
HPI Registers	256k	0188 0000 - 018B FFFF
McBSP 0 Registers	256k	018C 0000 - 018F FFFF
McBSP 1 Registers	256k	0190 0000 - 0193 FFFF
Timer 0 Registers	256k	0194 0000 - 0197 FFFF
Timer 1 Registers	256k	0198 0000 - 019B FFFF
Interrupt selector Registers	256k	019C 0000 - 019F FFFF
EDMA RAM and EDMA Registers	256k	01A0 0000 - 01A3 FFFF
Reserved	64M-256k	01A4 0000 - 01FF FFFF
QDMA Registers	52	0200 0000 - 0200 FFFF
Reserved	736M-52	0200 0034 - 2FFF FFFF
MCBSP 0/1 Data	256M	3000 0000 - 3FFF FFFF
Reserved	1G	4000 0000 - 7FFF FFFF
EMIF CE0	256M	8000 0000 - 8FFF FFFF
EMIF CE1	256M	9000 0000 - 9FFF FFFF
EMIF CE2	256M	A000 0000 - AFFF FFFF
EMIF CE3	256M	B000 0000 - BFFF FFFF
Reserved	1G	C000 0000 - FFFF FFFF

Memory Space Control Registers



Memory Space Control Registers

- **Global Control (GBLCTL): the EMIF global control register configures parameters that are common to all the CE spaces.**
  - **Polarity definition (x).**
  - **Signal enable (x).**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsv															
R,+0															

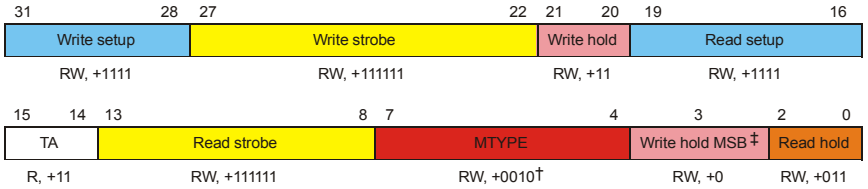
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rsv	Rsv	Rsv	Rsv	BUSREQ	ARDY	HOLD	HOLDA	NO HOLD	Rsv	Rsv	CLK1EN	CLK2EN	Rsv	Rsv	Rsv
R,+0	R/W,+0	R/W,+1	R/W,+1	R,+0	R,+x	R,+x	R,+x	R/W,+0	R,+1	R,+1	R/W,+1	R/W,+1	R,+0	R,+0	R,+0

EMIF Registers

Question: Why do we need different spaces?

Answer: Different spaces allow different types of devices to be used at the same time.

- **CE0, CE1, CE2, CE3 space control registers (CECTL): are used to specify the type and the read and write timing used for a particular space.**



† MTYPE default value is RW, +0000.
‡ For C621x/C671x, this field is reserved. R,+0.

EMIF Registers

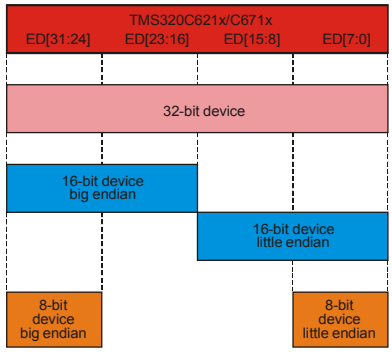
Field	Description
READ SETUP WRITE SETUP	Setup width. Number of clock† cycles of setup for address (EA) and byte enables (/BE(0-3)) before read strobe (/ARE) or write strobe (/AWE) falling. On the first access to a CE space, this is also the setup after /CE falling.
READ STROBE WRITE STROBE	Strobe width. The width of read strobe (/ARE) and write strobe (/AWE) in clock* cycles.
READ HOLD WRITE HOLD	Hold width. Number of clock† cycles that address (EA) and byte strobes (/BE(0-3)) are held after read strobe (/ARE) or write strobe (/AWE) rising. These fields are extended by one bit on the 'C6211/'C6711.
MTYPE	Memory type 'C6201/'C6202/'C6701 only: MTYPE = 000b: 8-bit-wide ROM (CE1 only) MTYPE = 001b: 16-bit-wide ROM (CE1 only) MTYPE = 010b: 32-bit-wide asynchronous interface 'C6211/'C6711 only: MTYPE = 0000b: 8-bit-wide asynchronous interface MTYPE = 0001b: 16-bit-wide asynchronous interface MTYPE = 0010b: 32-bit-wide asynchronous interface
TA‡	Turnaround time. Controls the number of ECLKOUT cycles between a read and a write or between two reads.

† Clock = CLKOUT1 for 'C6201/'C6202/'C6701.Clock = ECLKOUT for 'C6211/'C6711.
‡ Applies to 'C6211/'C6711 only.

Avoid bus contention

MEMORY WIDTH & BYTE ALIGNMENT

Memory type	Memory width	Maximum addressable bytes per CE space	Address output on EA[21:2]	Represents
ASRAM	x8	1M	A[19:0]	Byte address
	x16	2M	A[20:1]	Halfword address
	x32	4M	A[21:2]	Word address
SBSRAM	x8	1M	A[19:0]	Byte address
	x16	2M	A[20:1]	Halfword address
	x32†	4M	A[21:2]	Word address
SDRAM	x8	32M	See section 10.5	Byte address
	x16	64M	See section 10.5	Halfword address
	x32	128M	See section 10.5	Word address

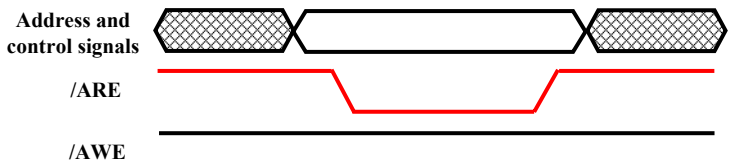


TMS320C6000 FAMILY: EMIF

- Introduction.
  - Characteristics, signals, memory map, alignment.
  - Configuration registers.
  - Types of interface.
- **Asynchronous interface.**
  - Introduction, waveforms.
  - Case Study I: Peripheral connection.
  - Case Study II: Memory connection.
- Interface with synchronous static memories.
  - Synchronous static memories.
  - Interface description.
  - Example.
- Interface with synchronous dynamic memories.
  - SDRAM memories.
  - Interface description.
  - Example.

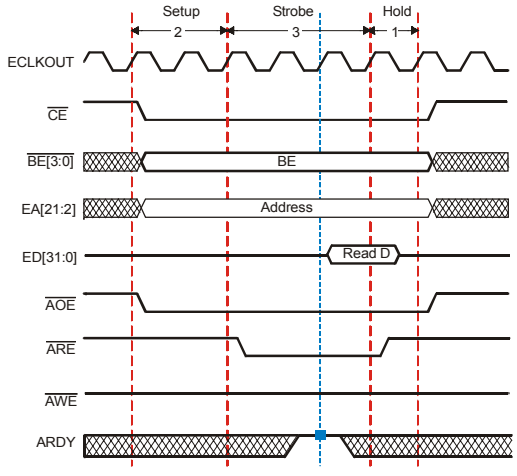
ASYNCHRONOUS INTERFACE

- It uses a Intel approach: two command signals to define the access direction and validate the rest of the signals used for decoding:
  - Read: ARE
  - Write: AWE
- These signals define the active part of the access cycle.
- The rest of signals used for decoding must be stables during the active part.



ASYNCHRONOUS INTERFACE

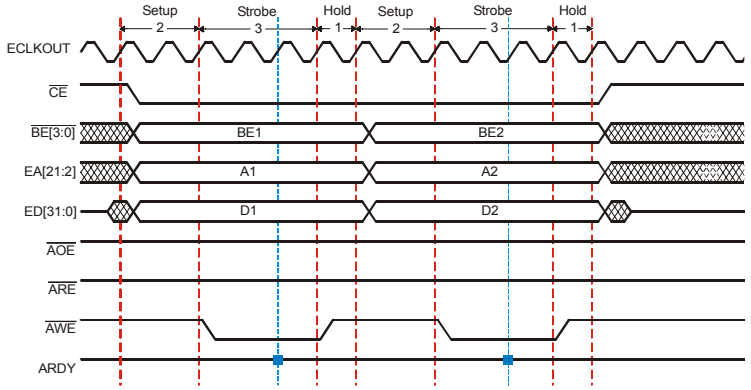
Asynchronous read timing example (2/3/1)



- Active part: 3 cycles.
- Decoded signals are stable 2 cycles before active part.
- Data captured at the end of active part.
- Decoded signals remain stable 1 cycle after active part.
- RDY signal is tested at the beginning of the last cycle of the active part.

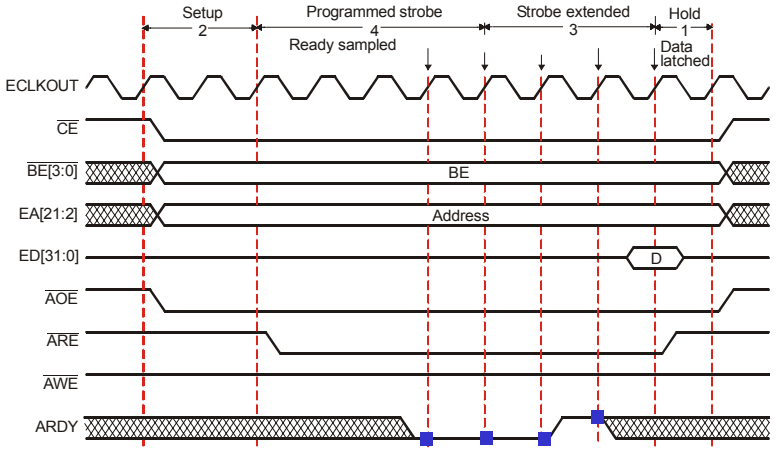
ASYNCHRONOUS INTERFACE

Asynchronous write timing example (2/3/1)



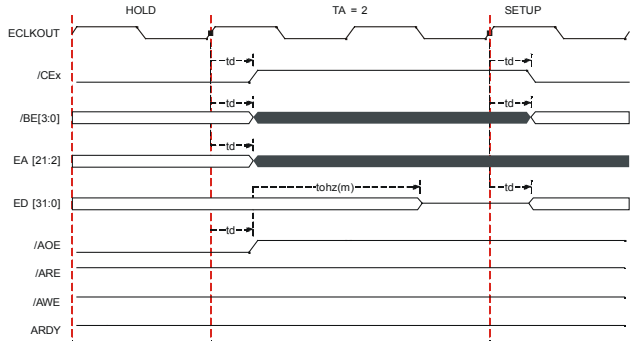
ASYNCHRONOUS INTERFACE

Ready operation: Wait state insertion

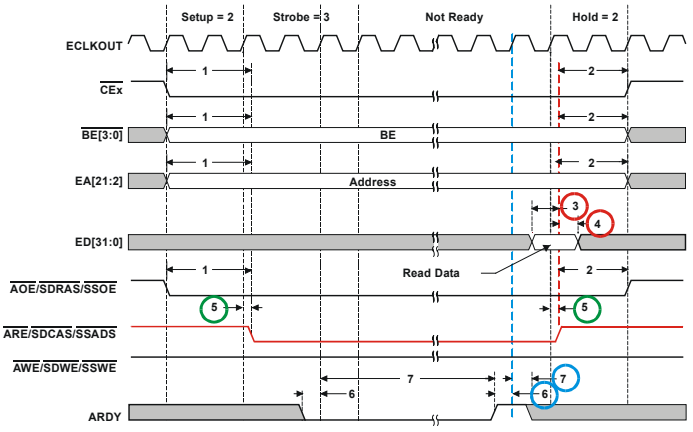


ASYNCHRONOUS INTERFACE

Turnaround Time



REAL WAVEFORMS: READ ACCESS

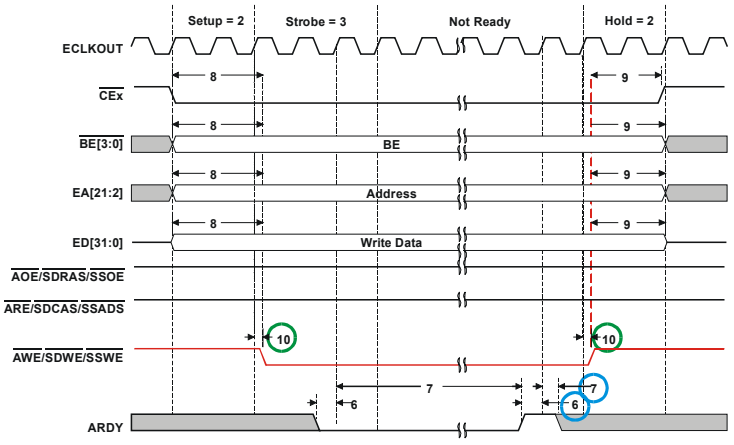


Outputs:
Delay time max: 7 ns

Data capture:
Setup time min: 6.5 ns
Hold time min: 1 ns

ARDY test:
Setup time min: 3 ns
Hold time min: 2.3 ns

REAL WAVEFORMS: WRITE ACCESS



Outputs:
Delay time max: 7 ns

ARDY test:
Setup time min: 3 ns
Hold time min: 2.3 ns

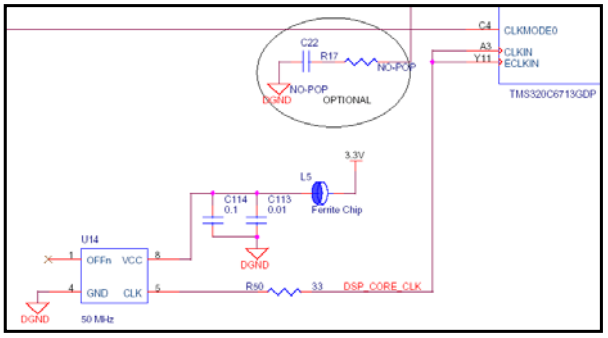
PROGRAMMABLE PARAMETERS

All of then are expressed as $T_{ECLKOUT}$ units:

- **SETUP** ≥ 1 (0 is treated as 1)
- **STROBE** ≥ 1 (0 is treated as 1)
- **HOLD** ≥ 0
- **TURNAROUND** ≥ 0

$T_{ECLKOUT}$ depends on system design:

- **For DSK6713: $T_{ECLKOUT}$= 50 MHz => $T_{ECLKOUT}$ = 20 ns.**



TYPICAL APPLICATIONS

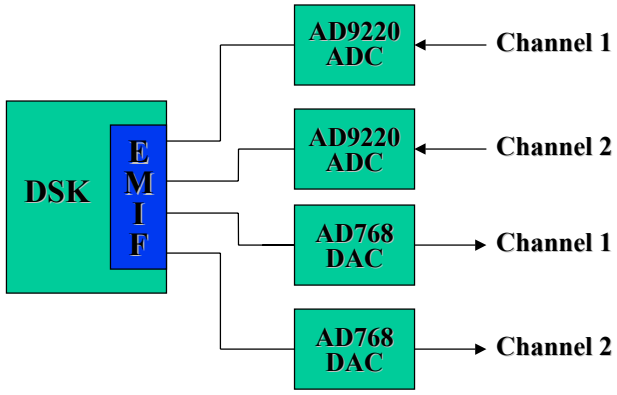
- The asynchronous interface is usually used to add extend the system resources by adding:
  - External peripheral with parallel interface: This kind of peripherals usually have an asynchronous interface.
  - Asynchronous memory banks: Easy design but reduced band width => restricted to non volatile memories (i.e FLASH)
- We will see two examples:
  - Peripheral connection: A/D y D/A.
  - Asynchronous memory banks connection.

TMS320C6000 FAMILY: EMIF

- Introduction.
  - Characteristics, signals, memory map, alignment.
  - Configuration registers.
  - Types of interface.
- Asynchronous interface.
  - Introduction, waveforms.
  - **Case Study I: Peripheral connection.**
  - Case Study II: Memory connection.
- Interface with synchronous static memories.
  - Synchronous static memories.
  - Interface description.
  - Example.
- Interface with synchronous dynamic memories.
  - SDRAM memories.
  - Interface description.
  - Example.

EMIF CASE STUDY (I)

- DSK interface to:
  - AD768 DAC.
  - AD9220 ADC.



EMIF CASE STUDY: AD768 DAC

- Specification:

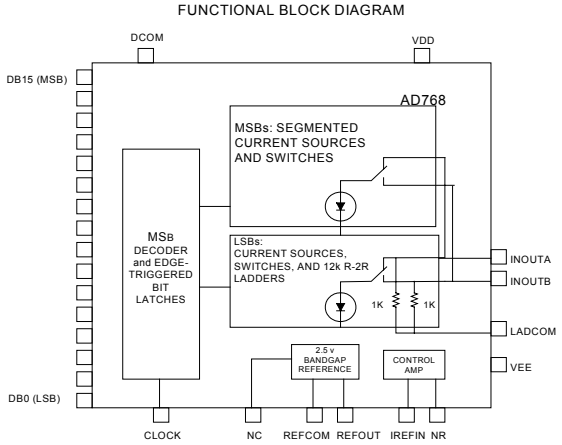
FEATURES:
30 msps Update Rate
16-Bit Resolution
Linearity: 1/2 LSB DNL @ 14 Bits
1 LSB INL @ 14 Bits
Fast Settling: 25ns Full-Scale Settling to 0.025%
SFDR @ 1 MHZ Output: 86 dBc
THD @ 1 MHZ Output: 71 dBc
Low Glitch Impulse: 35 pV-s
Power Dissipation: 465 mW
On-chip 2.5V reference
Edge Triggered Latches
Multiplying Reference Capability

APPLICATIONS:
Arbitrary Waveform Generation
Communications Waveform Reconstruction
Vector Stroke Display

- AD768 data sheet

EMIF CASE STUDY: AD768 DAC

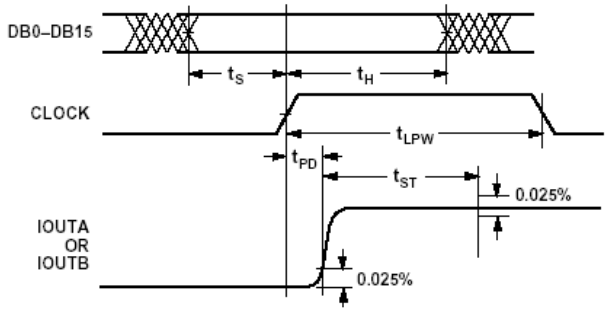
- **Functional Block Diagram:**



- AD768 data sheet

EMIF CASE STUDY: AD768 DAC

- **Timing:**



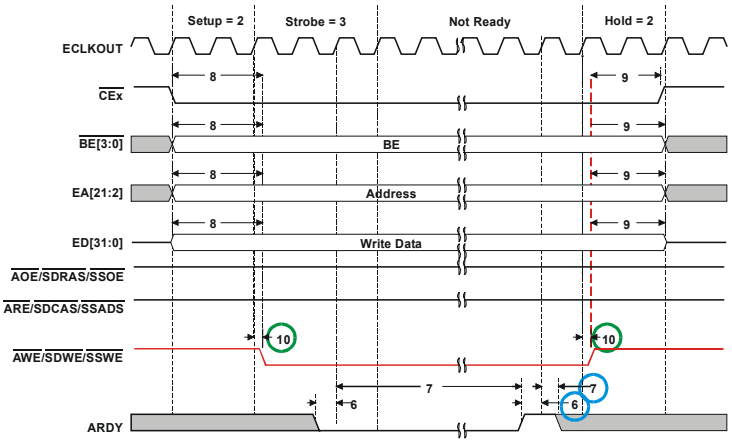
t_S = 10 ns
t_H = 5 ns
t_{LPW} = 10 ns

NOTE: for DSK6713 $T_{ECLKOUT}$= 50 MHz => $T_{ECLKOUT}$ = 20 ns.

- AD768 data sheet

EMIF CASE STUDY: AD768 DAC

- C6713 Asynchronous Write Timing:



SETTING ASYNC TIMING

31 28	27 22	21 20	19 16
Write Setup	Write Strobe	Write Hold	Read Setup
RW, +1111	RW, +111111	RW, +11	RW, +1111

15	14 13	8	7 4	3	2 0
TA	Read Strobe		MTYPE	Write Hold MSB	Read Hold
	RW, + 111111		RW, +0010	RW, +0	RW, +11

Set CE3 and to 32-bit ASYNC

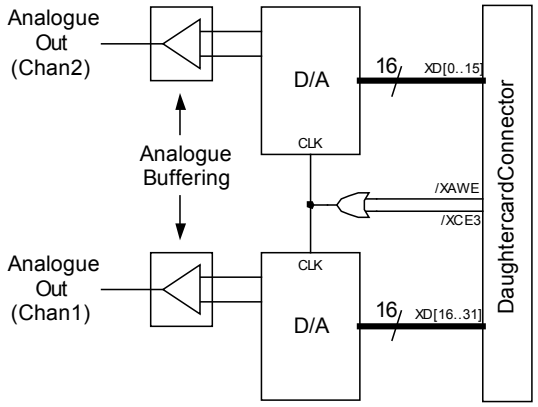
```
CE3     .equ        1800014h
        mvkl.s1     CE3, A0
        mvkh.s1     CE3, A0
        ldw         *A0, A1
        nop 4
        and         A1, 0xff0f, A1
        set         A1, 5, 5, A1
        stw .d1     A1, *A0
```

000b = 8-bit-wide ROM
001b = 16-bit-wide ROM
010b = 32-bit-wide Async
011b = 32-bit-wide SDRAM
100b = 32-bit-wide SBSRAM

Note: There are more MTYPE options. See: \Links\spru190d.pdf

SETTING ASYNC TIMING

- **Hardware Interface:**



- AD768 data sheet

EMIF CASE STUDY: AD768 DAC

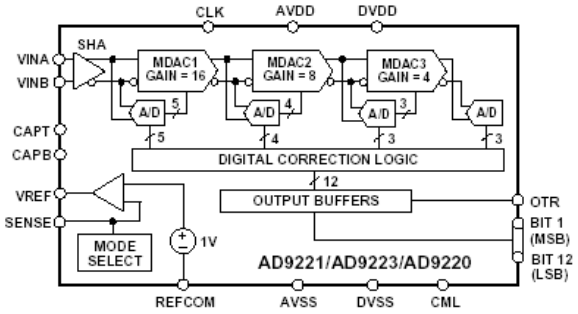
- **Specifications:**

FEATURES
Monolithic 12-Bit A/D Converter Product Family
Family Members Are: AD9221, AD9223, and AD9220
Flexible Sampling Rates: 1.5 MSPS, 3.0 MSPS and 10 MSPS
Low Power Dissipation: 59 mW, 100 mW and 250 mW
Single +5V Supply
Integral Nonlinearity Error: 0.5 LSB
Differential Nonlinearity Error: 0.3 LSB
Input Referred Noise: 0.09LSB
Complete On-Chip Sample-and-Hold Amplifier and Voltage Reference
Signal-to-Noise and Distortion Ratio: 70dB
Spurious-Free Dynamic Range: 86dB
Out-of-range Indicator
Straight Binary Output Data
28-Lead SOIC and 28-Lead SSOP

- AD9220 data sheet

EMIF CASE STUDY: AD9220 ADC

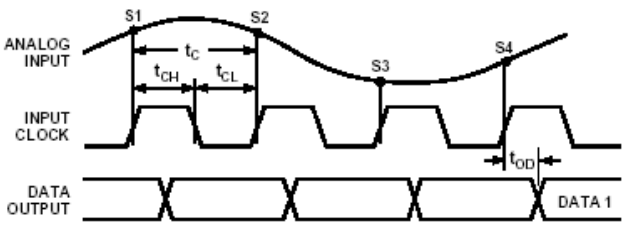
- **Functional Block Diagram:**



- AD9220 data sheet

EMIF CASE STUDY: AD9220 ADC

- **Timing:**



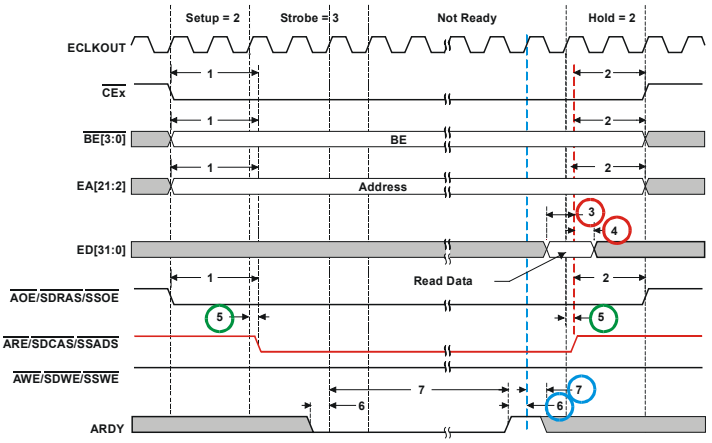
t_C = 100 ns
t_{CH} = 45 ns
t_{CL} = 45 ns
t_{OD} = 19 ns

NOTE: for DSK6713 $T_{ECLKOUT}$= 50 MHz => $T_{ECLKOUT}$ = 20 ns.

- AD9220 data sheet

EMIF CASE STUDY: AD9220 ADC

- C6713 Asynchronous Read Timing:



SETTING ASYNC TIMING

31–28	27–22	21–20	19–16
Write Setup	Write Strobe	Write Hold	Read Setup
RW, +1111	RW, +111111	RW, +11	RW, +1111

15	14	13–8	7–4	3	2–0
TA	Read Strobe		MTYPE	Write Hold MSB	Read Hold
	RW, + 111111		RW, +0010	RW, +0	RW, +011

Set CE3 and to 32-bit ASYNC

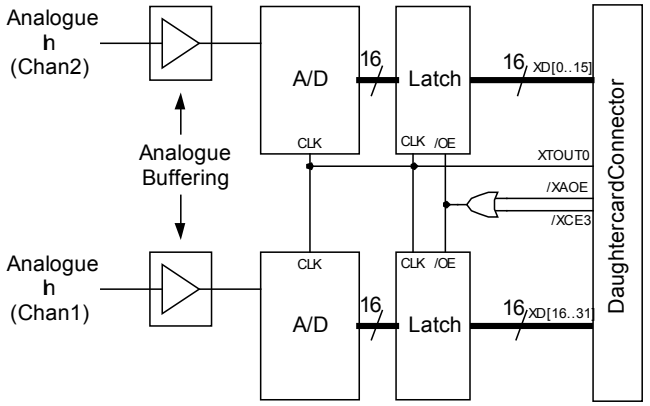
```
CE3     .equ        1800014h
        mvkl.s1     CE3, A0
        mvkh.s1     CE3, A0
        ldw         *A0, A1
        nop 4
        and         A1, 0xff0f, A1
        set         A1, 5, 5, A1
        stw .d1     A1, *A0
```

000b = 8-bit-wide ROM
001b = 16-bit-wide ROM
010b = 32-bit-wide Async
011b = 32-bit-wide SDRAM
100b = 32-bit-wide SBSRAM

Note: There are more MTYPE options. See: \Links\spru190d.pdf

EMIF CASE STUDY: AD9220 ADC

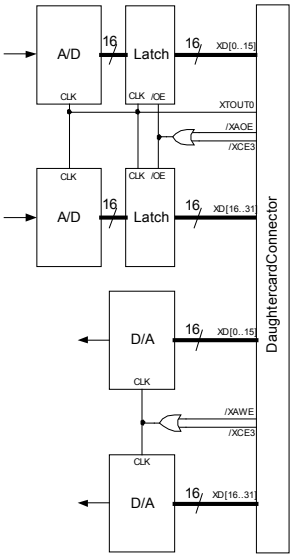
- **Hardware Interface:**



- AD9220 data sheet

EMIF CASE STUDY: Sharing the bus

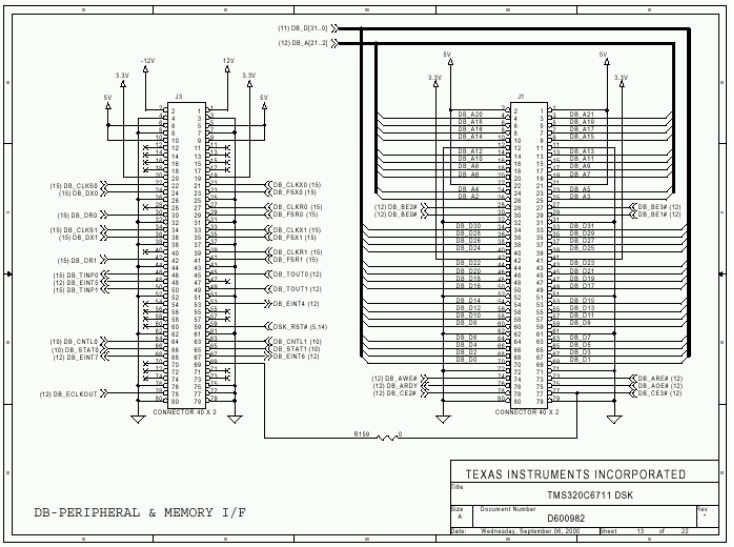
- **Both ADCs and DACs are mapped to the same address space (CE3 = 0xB000 0000).**



/CE3	/XAOE	/XAWE	/OE	DAC_CLK
0	0	1	0	1
0	1	0	1	0
1	x	x	1	1

/XAOE activates the latched A/D output only during the read sequence

EMIF CASE STUDY: Daughter board interface



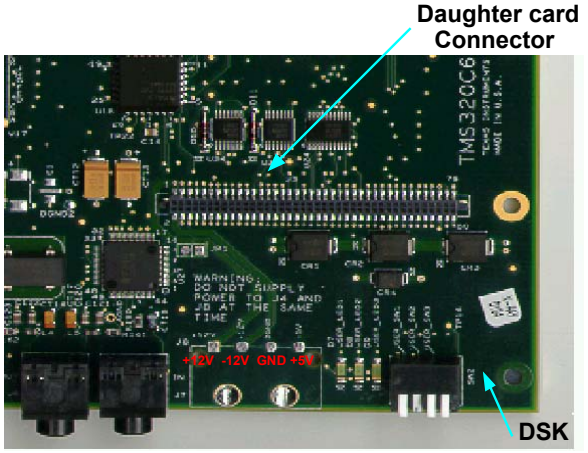
EMIF CASE STUDY: Hardware

- **The INTDSK1115 daughter card from ATE Communications contains:**
  - **CODEC.**
  - **2 x ADC (AD9920).**
  - **2 x DAC (AD768).**
- **See schematics for further details:**
  - \Links\Schematics Page 1.pdf
  - \Links\Schematics Page 2.pdf
  - \Links\Schematics Page 3.pdf
  - \Links\Schematics Page 4.pdf

EMIF CASE STUDY: Hardware

- It requires +12V, -12V and 5V power supplies:

Pin	Signal
1	+12V
2	-12V
3	DGND
4	+5V



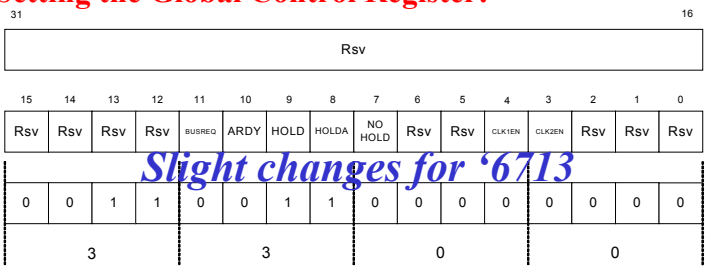
Warning: Do NOT supply power to J4 and J8 at the same time.

EMIF CASE STUDY: Hardware

- **Procedure:**
  - (1) Set the EMIF registers.
  - (2) Set the internal timer to generate the sampling frequency.
  - (3) Ensure that the DSK6211_6711.gel is loaded.
  - (4) Write the functions for reading and writing from/to the ADC and DAC respectively.
  - (5) Set the interrupts.

EMIF CASE STUDY: Software EMIF

(1) Setting the Global Control Register:



Slight changes for '6713

- The GBLCTL register is common to all spaces and can be configured as follows:

```
#define EMIF_GCTL 0x01800000

*(unsigned int *) EMIF_GCTL = 0x3300
```


EMIF CASE STUDY: Software EMIF

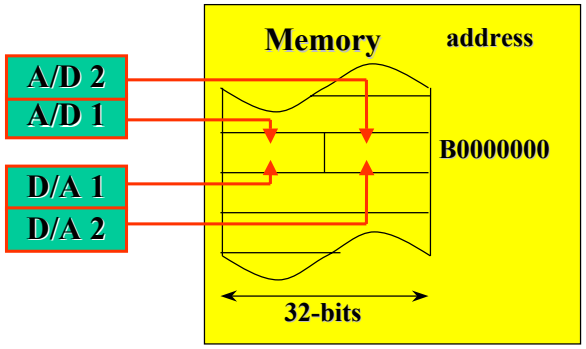
Setting the CE Control Register:

- Which space can be used to access the ADCs?
- From the DSK6211_6711.gel (DSK6211_6711_gel.pdf) file we can see that the CE2 and CE3 are not used and are available on the Daughter card interface.
- In this application the CE3 space has been used.

EMIF CASE STUDY: Software EMIF

Setting the CE3 Control Register:

- MTYPE?



The memory is configured as 32-bit asynchronous.

Therefore: MTYPE = 0010b.

EMIF CASE STUDY: Software EMIF

Setting the CE3 Control Register:

- MTYPE = 0010b: 32-bit async
- Read/Write Hold = 011b: 3 x ECLKOUT
- Read/Write Strobe = 111111b: 31 x ECLKOUT
- Read/Write Setup = 1111b: 15 x ECLKOUT

◆ **Therefore the CE3 space can be configured as follows:** *Too conservative timing desing*

```
#define EMIF_CE3   0x01800014

*(unsigned int *) EMIF_CE3 = 0xffffff23
```

EMIF CASE STUDY: Software Timer

Setting the sample rate: Using the internal timer

(2) Select a timer: there are two timers available, Timer 0 and Timer 1.

- **The two internal timers are controlled by six memory-mapped registers (3 registers each):**
  - **(a) Timer control registers: sets the operating modes.**
  - **(b) Timer period registers: holds the number of timer clock cycles to count.**
  - **(c) Timer counters: holds current value of the incrementing counter.**

Note: the timer clock is the CPU clock divided by 4.

EMIF CASE STUDY: Software Timer

Register	Address		Description
	Timer 0	Timer 1	
Timer control	0x0194 0000	0x0198 0000	Sets the operating mode
Timer period	0x0194 0004	0x0198 0004	Holds the number of timer clock cycles to count
Timer counter	0x0194 0008	0x0198 0008	Holds the current counter value

EMIF CASE STUDY: Software Timer

Initialise the timer:

CPU Frequency = FCPU = 150000000 Hz

Sampling rate = SRATE = 4000 Hz

$$TPRD = \frac{FCPU}{4 \times 2 \times 4000} = \frac{150000000}{32000} = 468.75$$

$$= 0x01D5$$

EMIF CASE STUDY: Software Timer

```
#define FCPU      150000000          /* CPU clock frequency */
#define SRATE     800000             /* data sample rate 800kHz */
#define TPRD      (FCPU/(4*2*SRATE)) /* timer period, using the clock mode */

TIMER_Handle hTimer;         /* Handle for the timer device  */

void start_timer1()
{
      *(unsigned volatile int *)TIMER1_CTRL = 0x000; /* Disable output of Timer 1 */

      IRQ_map(IRQ_EVT_TINT1,8);
      hTimer = TIMER_open(TIMER_DEV1, TIMER_OPEN_RESET);

      /* Configure up the timer. */
      TIMER_configArgs(hTimer,
         TIMER_CTL_OF(0x000003c1),
         TIMER_PRD_OF(TPRD),
         TIMER_CNT_OF(0) );

      /* Start Timer 1 in clock mode */
      *(unsigned volatile int *)TIMER1_CTRL = 0x3C1;//clock mode
      /* Finally, enable the timer which will drive everything. */
      TIMER_start(hTimer);
}
```

EMIF CASE STUDY: Loading GEL

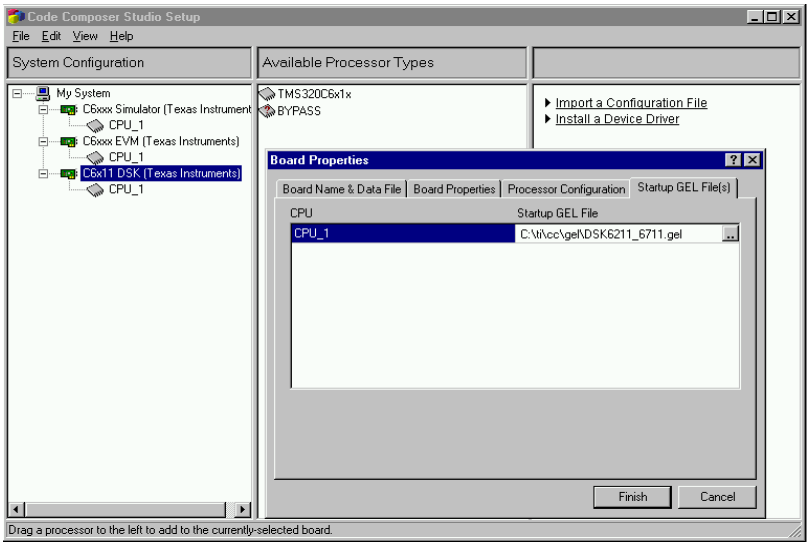
(3) For the DSK6211 and DSK6711 select the DSK6211_6711.gel using:

Method 1 File:Load GEL
Location: ti\cc\gel\

Method 2: You can automatically execute a specific GEL function at startup as follows:

(1) Select Setup CCS.
(2) Select the C6x11 DSK and right click.
(3) Select the "Startup GEL file(s)".
(4) Type the file location as shown:

EMIF CASE STUDY: Loading GEL



EMIF CASE STUDY: Reading and writing to the A/D y D/A

(4) The ADC and DAC are memory-mapped and therefore can be accessed just like accessing a memory.

```
#define INTDSK_CE3 0xB0000000
unsigned int analogue_in = 0;
unsigned int analogue_out = 0;

interrupt void timerINT1 (void)
{
        analogue_in = *(unsigned volatile int *) INTDSK_CE3;

        /* data processing */
        ad1 = analogue_in & 0xffff0000;    /* mask ad2 */
        ad2 = analogue_in & 0x0000ffff;    /* mask ad1 */
        ad1 = ad1 << 4;
        ad2 = ad2 << 4;
        *(unsigned volatile int *) INTDSK_CE3 = analogue_out;
}
```


EMIF CASE STUDY: Setting the Interrupt

(5) Timer1 is used to generate the interrupts:

The interrupt causes the execution of an ISR to take place (e.g. “InoutISR”).

Procedure for setting interrupt:

(1) Map the CPU interrupt and the source:

```
#include <intr.h>
#include <regs.h>

IRQ_map (IRQ_EVT_TINIT, 8);
```

(2) Enable the appropriate bit of the IER:

```
IRQ_enable (IRQ_EVT_TINT1);
```

(3) Enable the NMI:

```
IRQ_nmiEnable ();
```

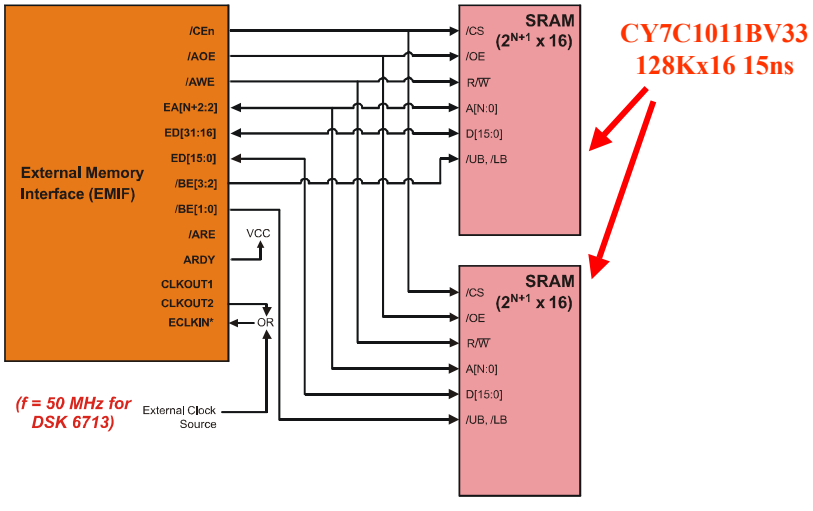
(4) Enable global interrupts:

```
IRQ_globalEnable ();
```

TMS320C6000 FAMILY: EMIF

- Introduction.
  - Characteristics, signals, memory map, alignment.
  - Configuration registers.
  - Types of interface.
- Asynchronous interface.
  - Introduction, waveforms.
  - Case Study I: Peripheral connection.
  - **Case Study II: Memory connection.**
- Interface with synchronous static memories.
  - Synchronous static memories.
  - Interface description.
  - Example.
- Interface with synchronous dynamic memories.
  - SDRAM memories.
  - Interface description.
  - Example.

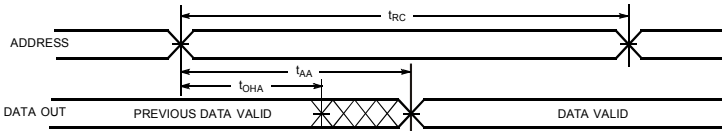
CASE STUDY II: ASYNC SRAM CONNECTION



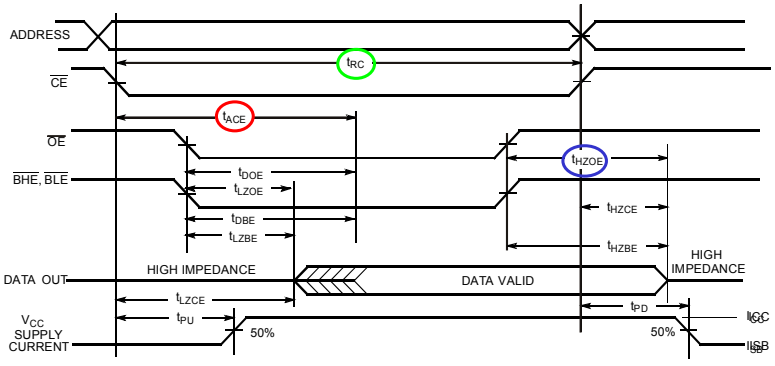
Which waveforms take into account?

- Access cycle controlled by XXX signal.
  - **Read**: XXX is the last one to get active (signaling the new data appearing) and the first one to get disable (signaling its disappearing).
  - **Write**: XXX is the last one to get active (the rest of the signals should be stables at this moment) and the first one to get inactive (signaling the data capture).

READ CYCLE: address controlled.

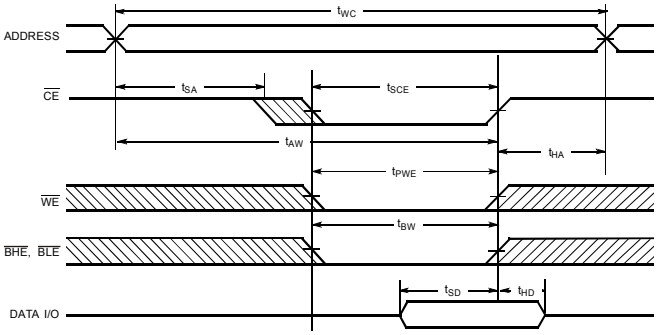


READ CYCLE: /OE or /BE controlled.



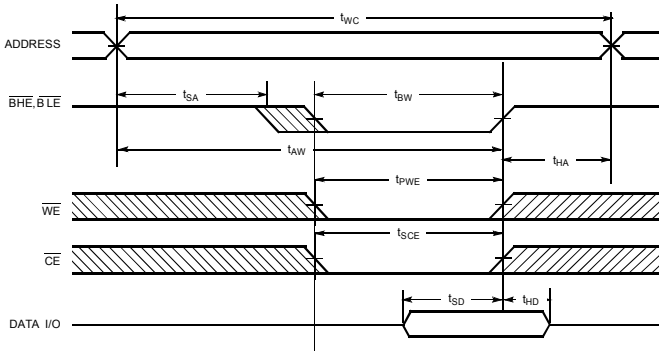
CASE STUDY: ASYNC SRAM CONNECTION

WRITE CYCLE: /CE controlled

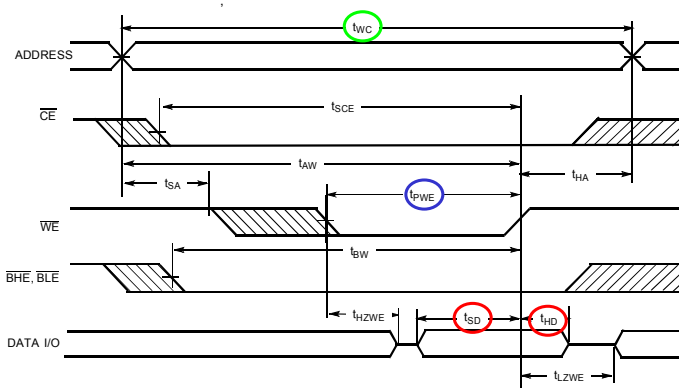


CASE STUDY: ASYNC SRAM CONNECTION

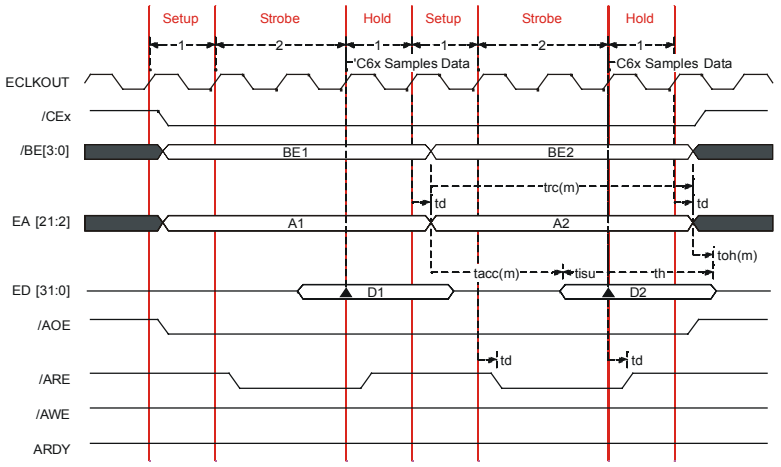
WRITE CYCLE: /BE controlled



WRITE CYCLE: /WE controlled, /OE low

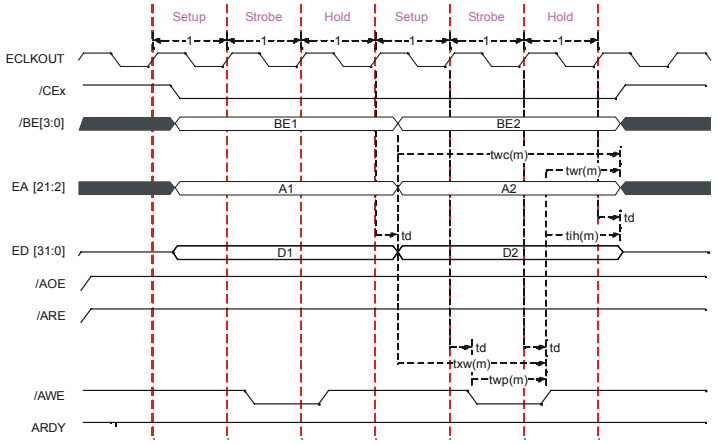


Asynchronous read timing example (1/2/1)



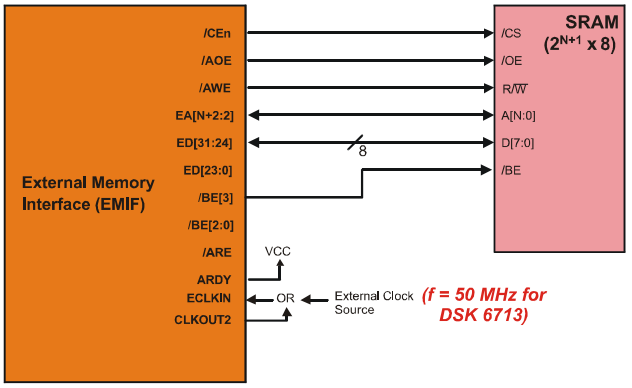
CASE STUDY: ASYNC SRAM CONNECTION

Asynchronous write timing example (1/1/1)



CASE STUDY: FLASH CONNECTION

EMIF-Big-Endian x8 ASRAM Interface



TMS320C6000 FAMILY: EMIF

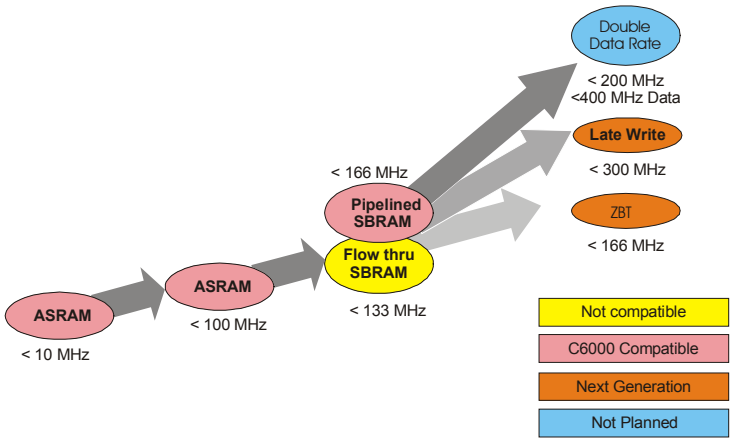
- Introduction.
  - Characteristics, signals, memory map, alignment.
  - Configuration registers.
  - Types of interface.
- Asynchronous interface.
  - Introduction, waveforms.
  - Case Study I: Peripheral connection.
  - Case Study II: Memory connection.
- **Interface with synchronous static memories.**
  - Synchronous static memories.
  - Interface description.
  - Example.
- Interface with synchronous dynamic memories.
  - SDRAM memories.
  - Interface description.
  - Example.

EMIF: Other interfaces

Field	Description
Read setup Write setup	Setup width. Number of clock† cycles of setup time for address (EA), chip enable ($\overline{CE}$), and byte enables ($\overline{BE}$[0-3]) before read strobe or write strobe falls. For asynchronous read accesses, this is also the setup time of $\overline{AOE}$ before $\overline{ARE}$ falls.
Read strobe Write strobe	Strobe width. The width of read strobe ($\overline{ARE}$) and write strobe ($\overline{AWE}$) in clock† cycles
Read hold Write hold	Hold width. Number of clock† cycles that address (EA) and byte strobes ($\overline{BE}$[0-3]) are held after read strobe or write strobe rises. For asynchronous read accesses, this is also the hold time of $\overline{AOE}$ after $\overline{ARE}$ rising.
MTYPE	Memory type of the corresponding CE spaces for C620x/C670x MTYPE Definitions for C621x/C671x/C64x‡ MTYPE = 0000b: 8-bit-wide asynchronous interface MTYPE = 0001b: 16-bit-wide asynchronous interface MTYPE = 0010b: 32-bit-wide asynchronous interface MTYPE = 0011b: 32-bit-wide SDRAM MTYPE = 0100b: 32-bit-wide SBSRAM (C621x/C671x) 32-bit-wide programmable synchronous memory (C64x) MTYPE = 1000b: 8-bit-wide SDRAM MTYPE = 1001b: 16-bit-wide SDRAM MTYPE = 1010b: 8-bit-wide SBSRAM (C621x/C671x) 8-bit-wide programmable synchronous memory (C64x) MTYPE = 1011b: 16-bit-wide SBSRAM (C621x/C671x) 16-bit-wide programmable synchronous memory (C64x) MTYPE = 1100b: 64-bit-wide asynchronous interface (C64x only) MTYPE = 1101b: 64-bit-wide SDRAM (C64x only) MTYPE = 1110b: 64-bit-wide programmable synchronous memory (C64x only)
TA	Turn-around time (C621x/C671x/C64x only). Turn-around time controls the number of ECLKOUT cycles between a read, and a write, or between reads, to different CE spaces (asynchronous memory types only).

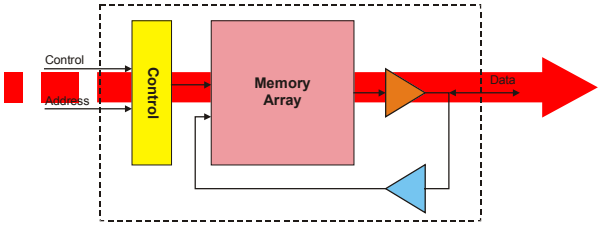
† Clock cycles are in terms of CLKOUT1 for C620x/C670x, ECLKOUT for the C621x/C671x, and ECLKOUT1 for the C64x.
‡ 32-bit and 64–bit interfaces (MTYPE=0010b, 0011b, 0100b, 1100b, 1101b, 1110b) do not apply to C6712 and C64x EMIFB.

SYNCHRONOUS SRAMs

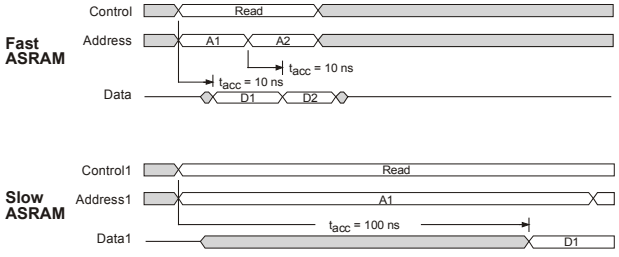


SYNCHRONOUS SRAMs

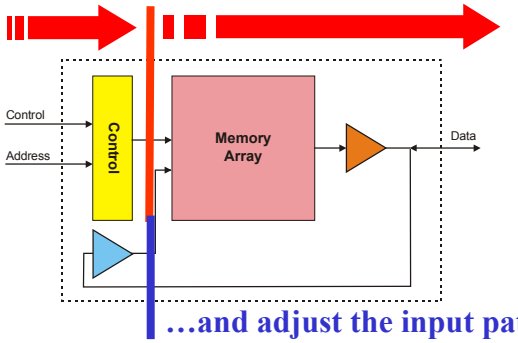
Asynchronous SRAMs present long critical path...



…and then a high access time



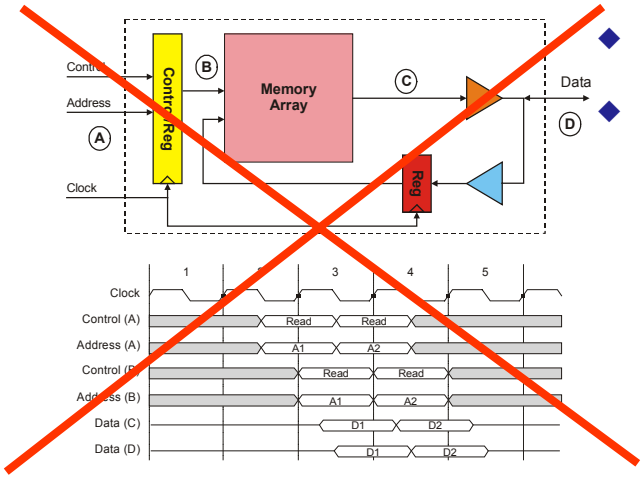
Solution: cut the output critical path...



…and adjust the input path

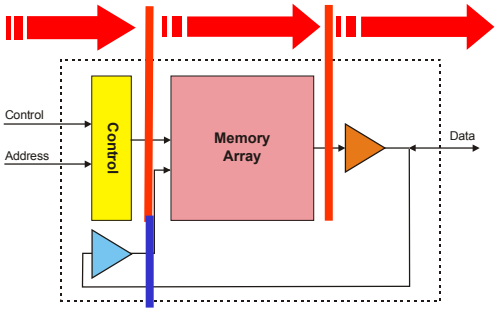
FLOW-THROUGH SYNCRONOUS SRAM

FLOW-THROUGH SYNCRONOUS SRAM



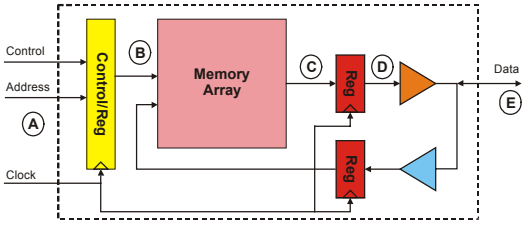
- **Read:**
  - 1 cycle latency
- **Incompatible with EMIF**

A further improvement

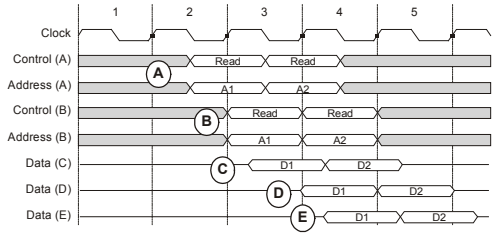


PIPELINED SYNCRONOUS SRAM

PIPELINED SYNCRONOUS SRAM

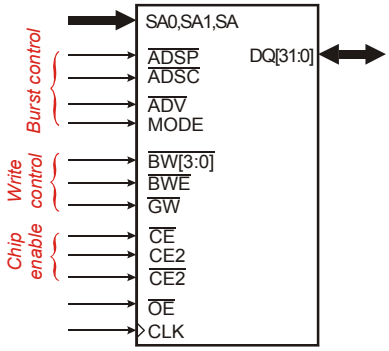


- **Read:**
  - **2 cycles latency**



PIPELINED SYNCHRONOUS SRAMs

MT58128L32P1
128Kx32 225 MHz

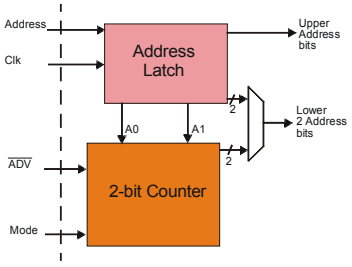


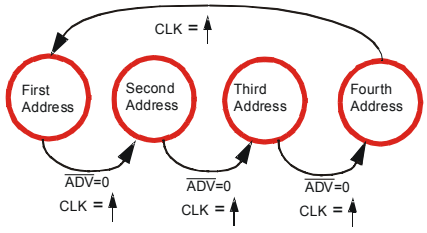
- **Address bus**
  - *SA0, SA1, SA*
- **Data bus**
  - *DQ[31:0]*
- **Burst control**
  - *#ADSP, #ADSC*
  - *#ADV*, MODE
- **Write control**
  - *#BW[3:0], #BWE*
  - *#GW*
- **Chip enable**
  - */CE, CE2, /CE2*
- **Output enable**
  - /OE
- **CLK.**

- **FEATURES:**
  - Burst access.
  - One cycle Deselect for READ access.

PIPELINED SYNCRONOUS SRAMs

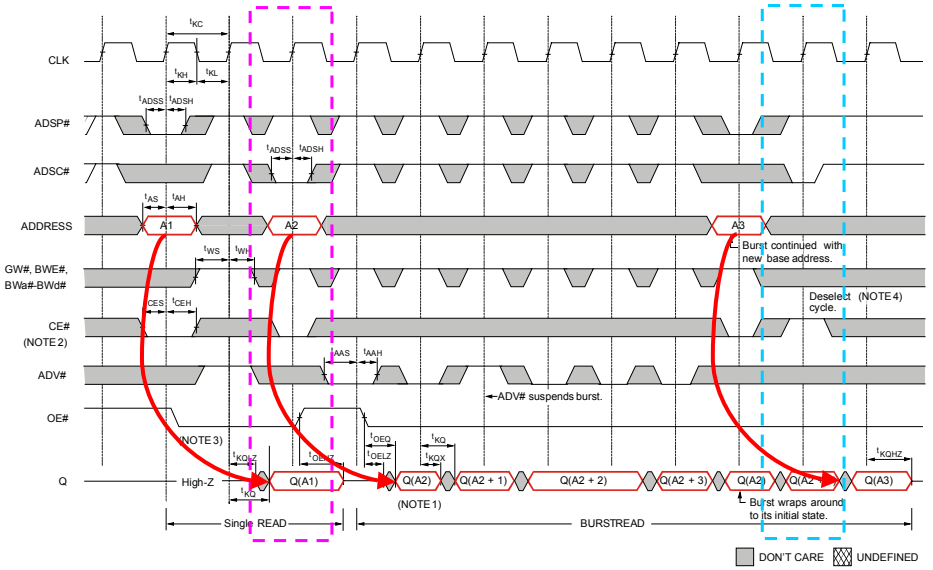
PIPELINED SYNCRONOUS SRAM: BURST MODE



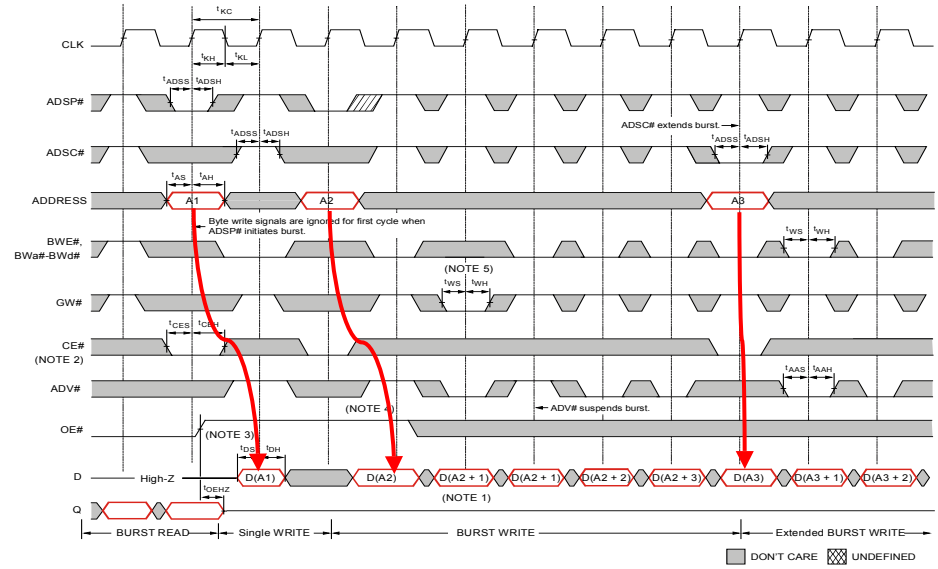


- It allows the automatic generation of the next address.
- Linear or interleaved increment.
- It automatically rolls over to 00 from 11.
  - if address 0111b were issued in burst mode, the subsequent access would be to 0100b. => **Error!**

PIPELINED SYNC SRAMs: READ CYCLE



PIPELINED SYNC SRAMs: WRITE CYCLE



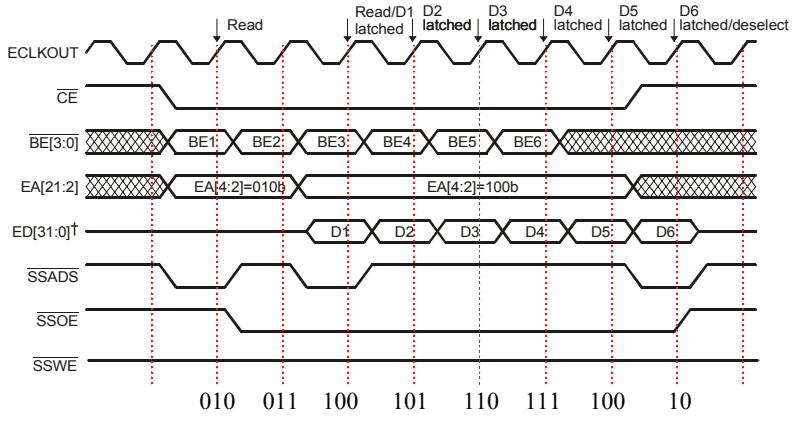
EMIF: SYNC. BURST SRAM INTERFACE

- **Read deselect.**
- **Burst access, linear increment.**
  - **Roll over correction: If address 0111b were issued in burst mode, the subsequent access would be to 0100b. => Error!**

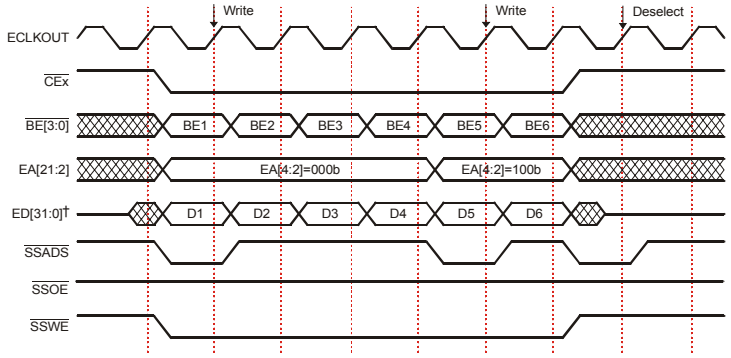
	Case 1	Case 2	Case 3	Case 4
SBSRAM Address	A[1:0]	A[1:0]	A[1:0]	A[1:0]
EMIF Address	EA[3:2]	EA[3:2]	EA[3:2]	EA[3:2]
First address	00	01	10	11
	01	10	11	00
	10	11	00	01
Fourth Address	11	00	01	10

SYNCHRONOUS BURST SRAM INTERFACE

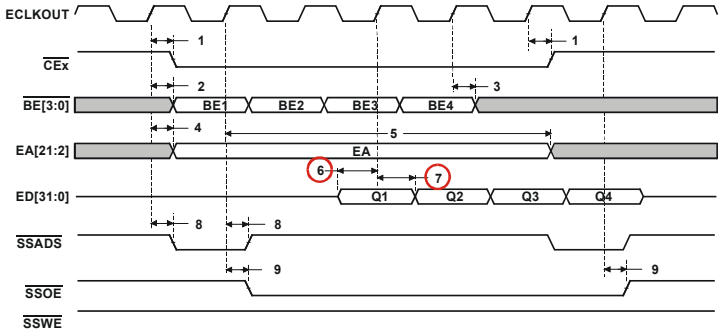
Read Example: six word read



Write Example: six word write

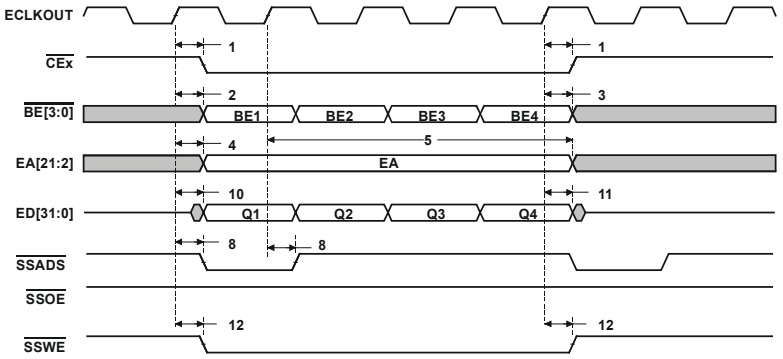


READ ACCESS

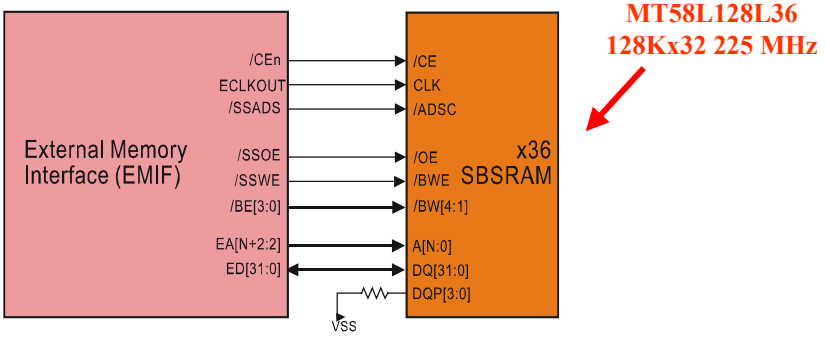


REAL WAVEFORMS

WRITE ACCESS



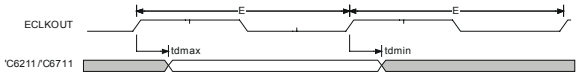
CONNECTION



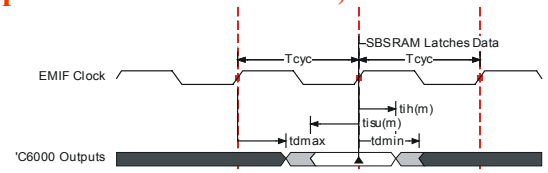
MT58L128L36
128Kx32 225 MHz

TIMING

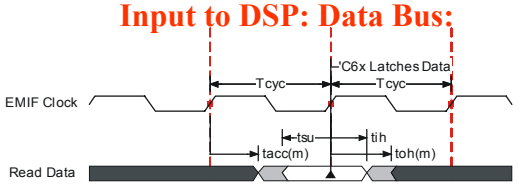
External clock interface:



Output from DSP: Data Bus, Control and Address:



Input to DSP: Data Bus:



TMS320C6000 FAMILY: EMIF

- Introduction.
  - Characteristics, signals, memory map, alignment.
  - Configuration registers.
  - Types of interface.
- Asynchronous interface.
  - Introduction, waveforms.
  - Case Study I: Peripheral connection.
  - Case Study II: Memory connection.
- Interface with synchronous static memories.
  - Synchronous static memories.
  - Interface description.
  - Example.
- **Interface with synchronous dynamic memories.**
  - SDRAM memories.
  - Interface description.
  - Example.

TO BE ADDED…

FINAL