# Chapter 8

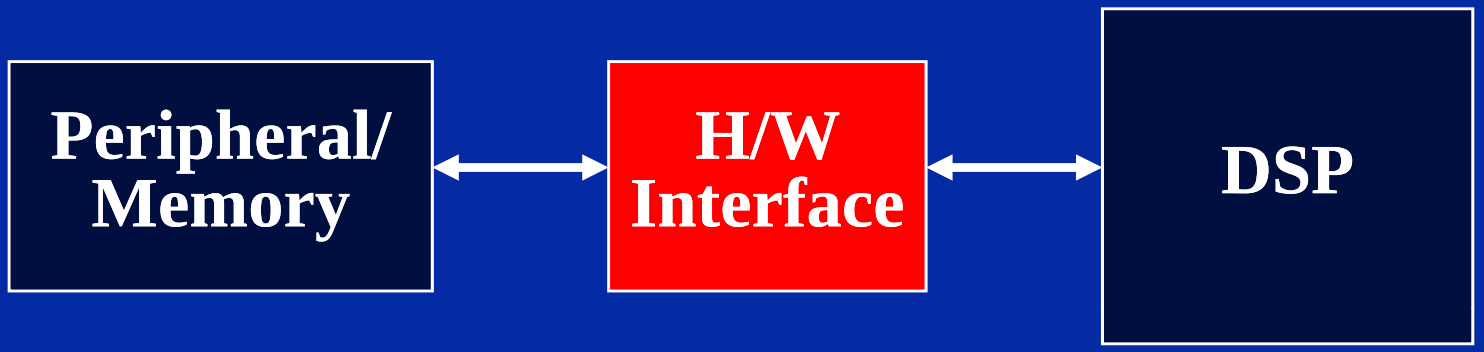
# External Memory Interface (EMIF)

# Learning Objectives

- The need for an External Memory Interface (EMIF).
- Memory types.
- C6211/C6711 memory map.
- C6211/C6711 EMIF features and signals.
- Memory space control registers.
- Asynchronous interface (A/D & D/A).
- Internal Timer.
- Application notes: SRAM, SBSRAM, SDRAM

# Need for an EMIF

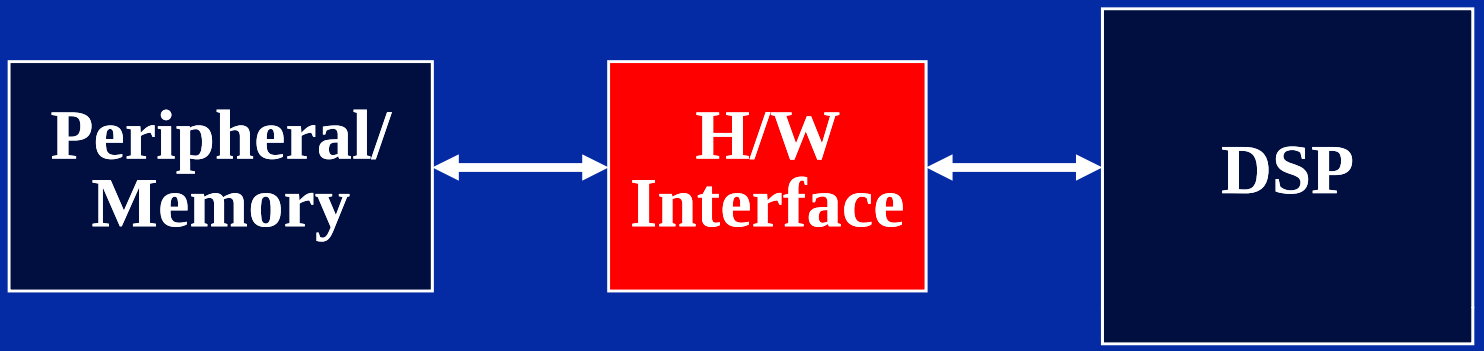
- **Traditional DSP (with no EMIF):**




- **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

- **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

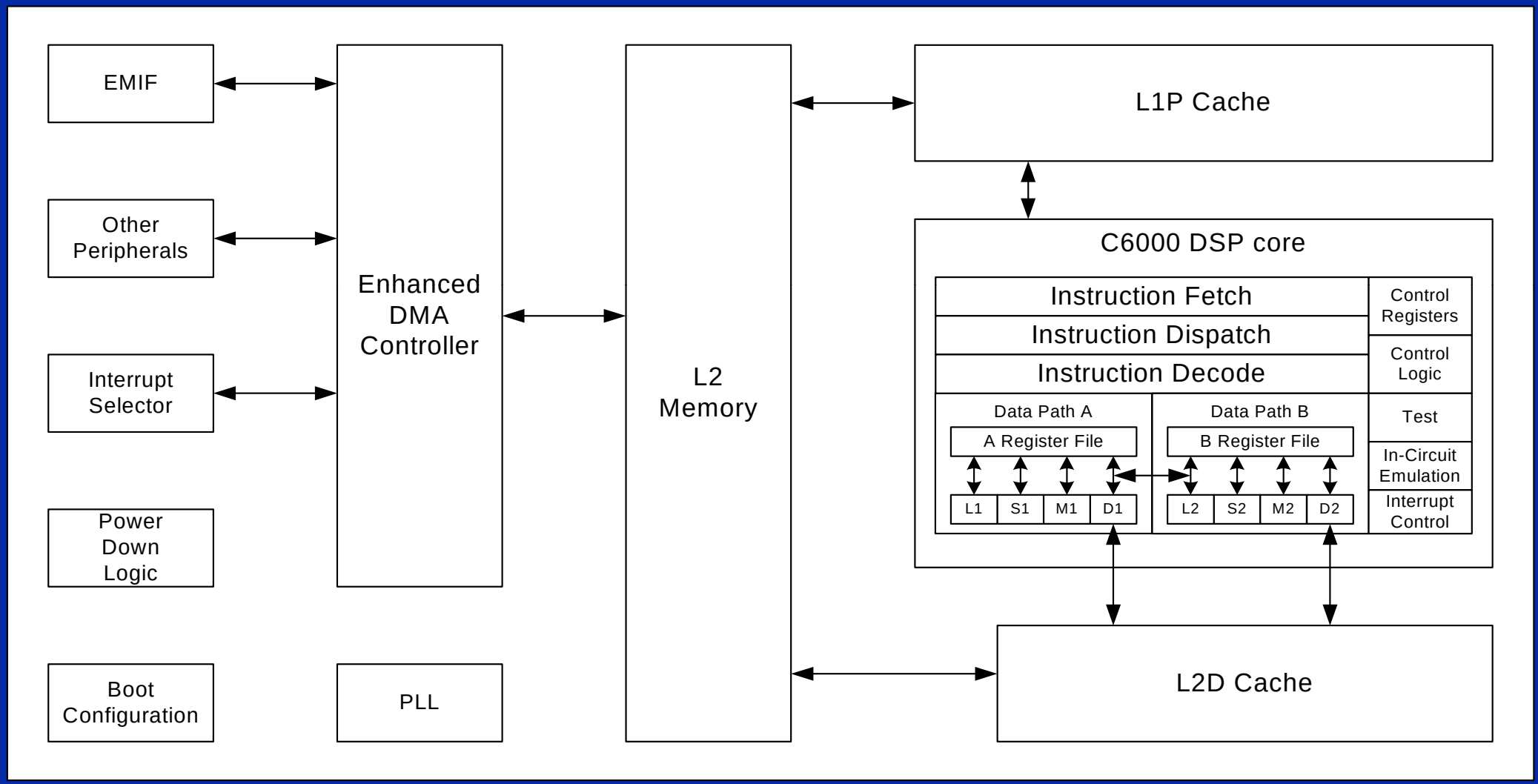
- The EMIF supports a glueless interface to several external devices, including:
  - Synchronous burst SRAM (SBSRAM).
  - Synchronous DRAM (SDRAM).
  - Asynchronous devices, including SRAM, ROM and FIFO's.
  - An external shared-memory device.

- For more information on different memory types see Links\SPRA631.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.

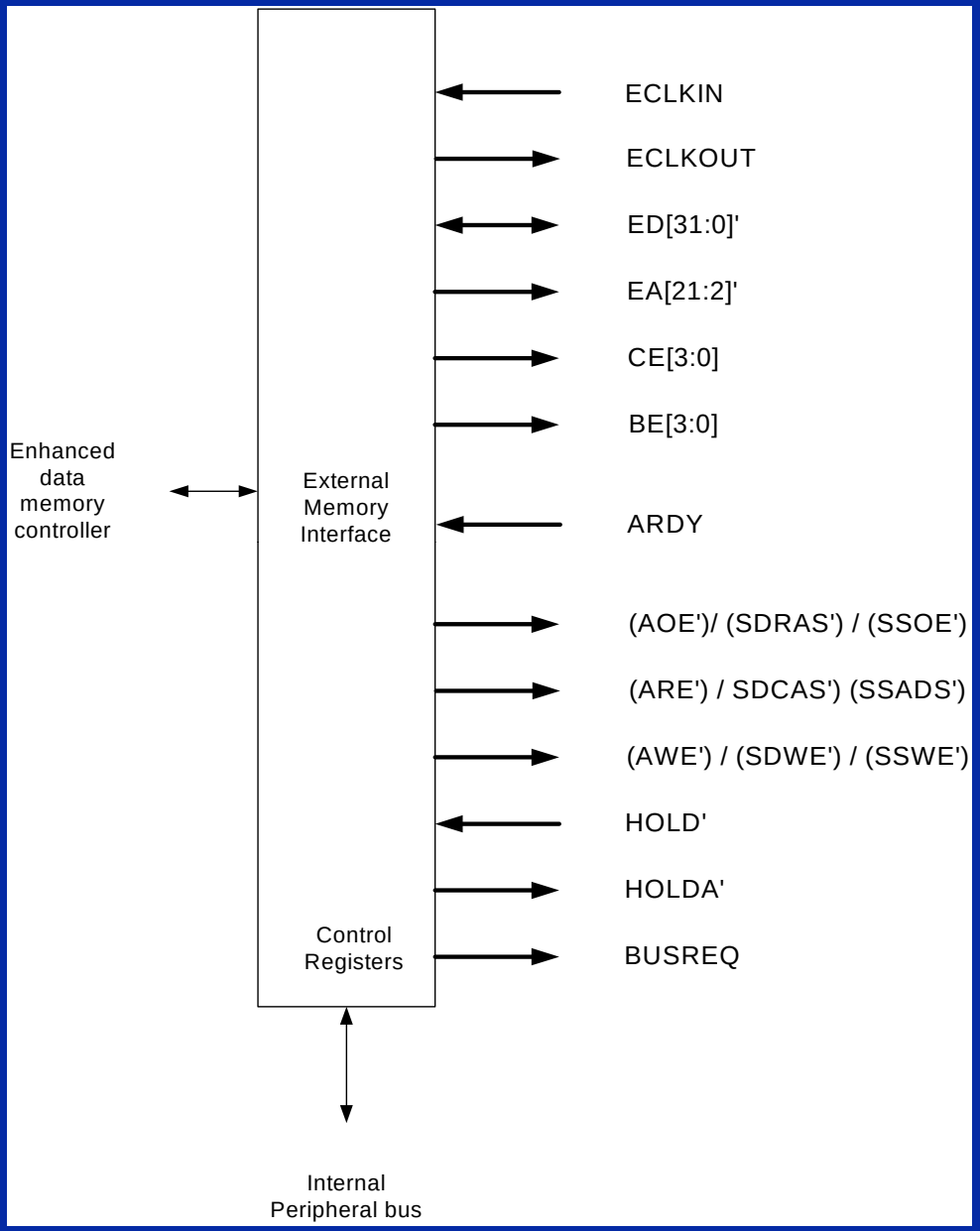
# The EMIF



# 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 |

# C6211/C6711 EMIF Signals




- **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.**

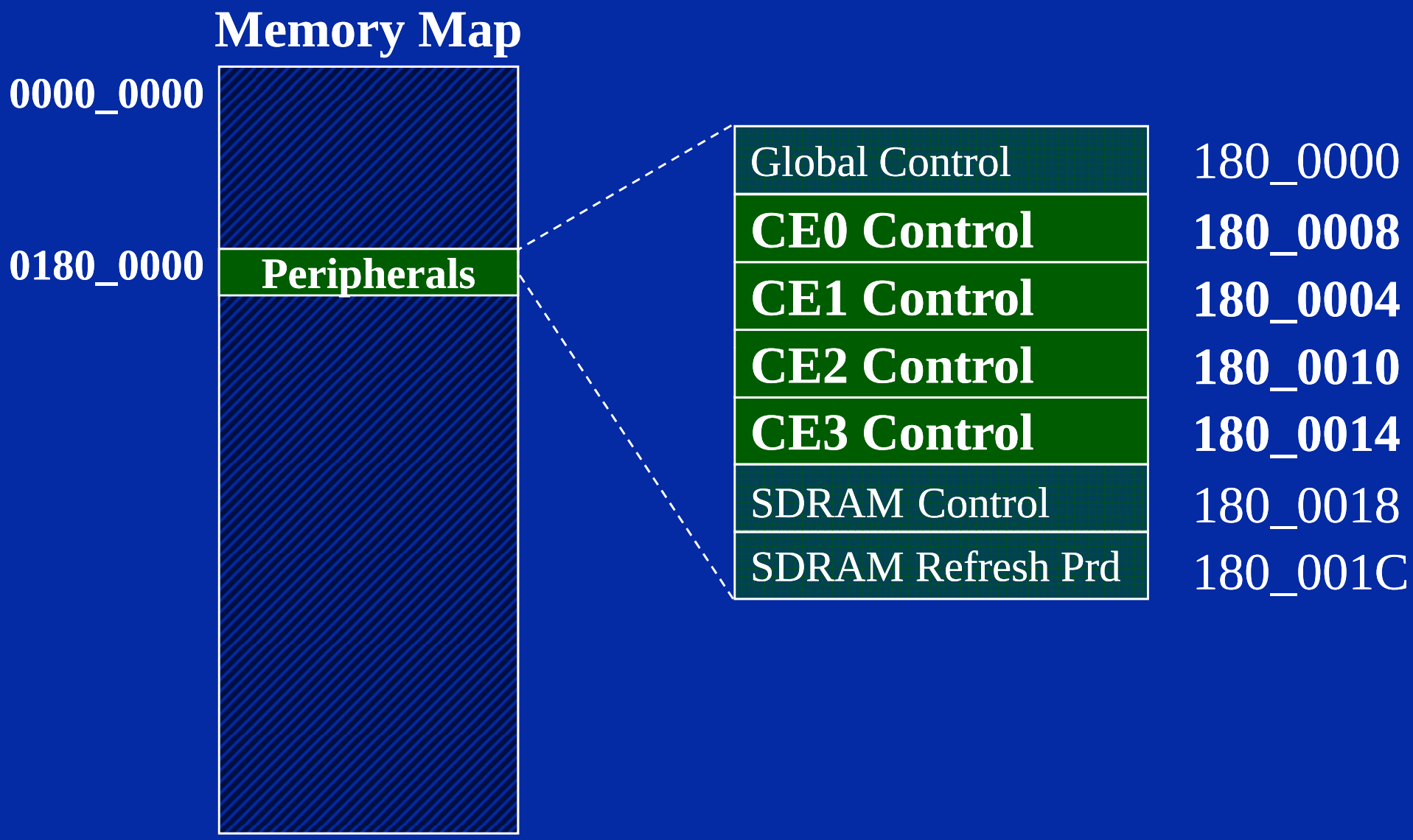
  **(2) EMIF chip enable (hardware):**

  **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

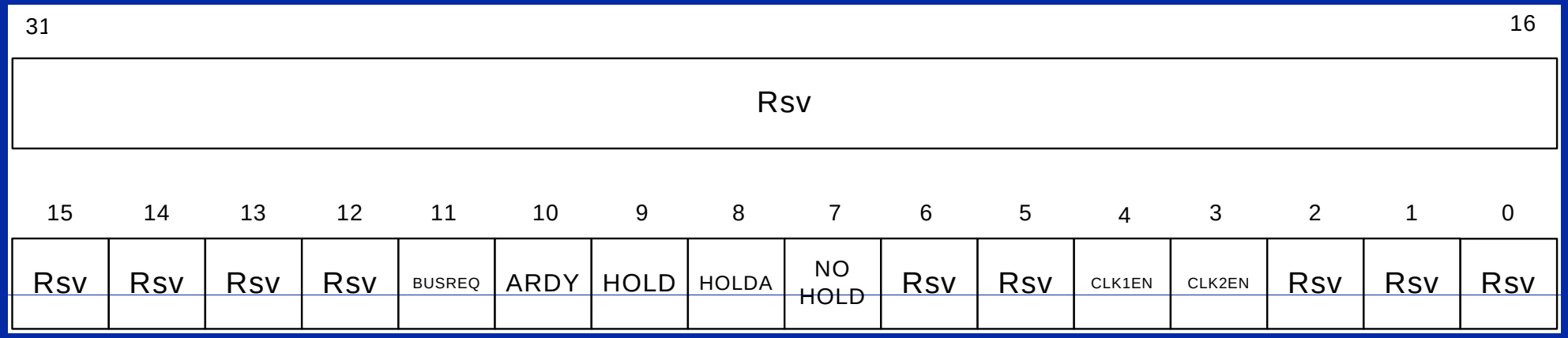
| Memory Block Description | Block Size (Bytes) | HEX Address Range |
|---|---|---|
| Internal RAM (L2) | 64K | 0000 0000 - 0000 FFFF |
| Reserved | 24M-64K | 0001 0000 - 017F FFFF |
| EMIF Registers | 256k | 0180 0000 - 0183 FFFF |
| L2 Registers | 256k | 0184 0000 - 0187 FFFF |
| HPI Registers | 256k | 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



# C6211/C6711 EMIF Registers

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

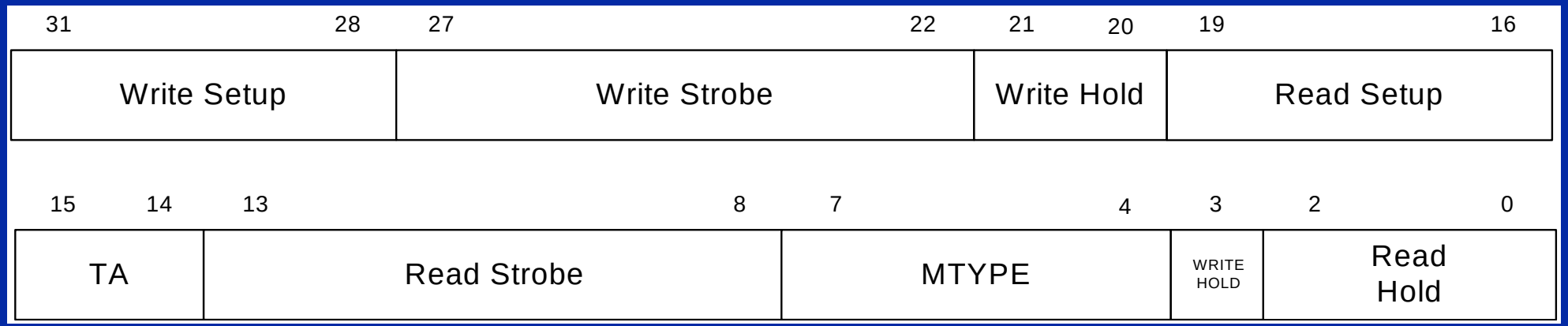


# C6211/C6711 EMIF Registers

**Question: Why do we need different spaces?**

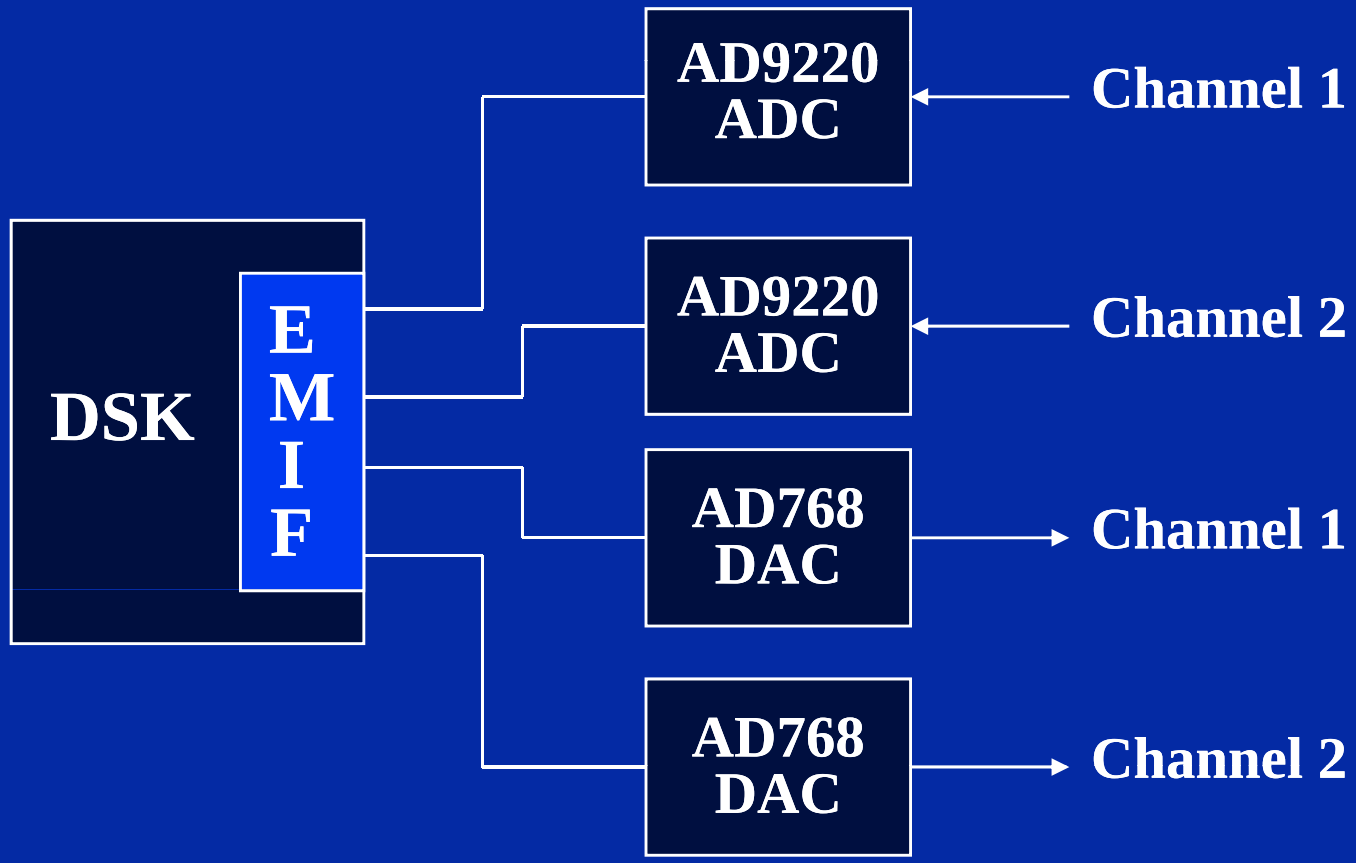
**Answer: Different spaces allow different types of devices to be used at the same time.**

- **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.**



# EMIF Case Study

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



# EMIF Case Study: AD768 DAC

- **Specification:**

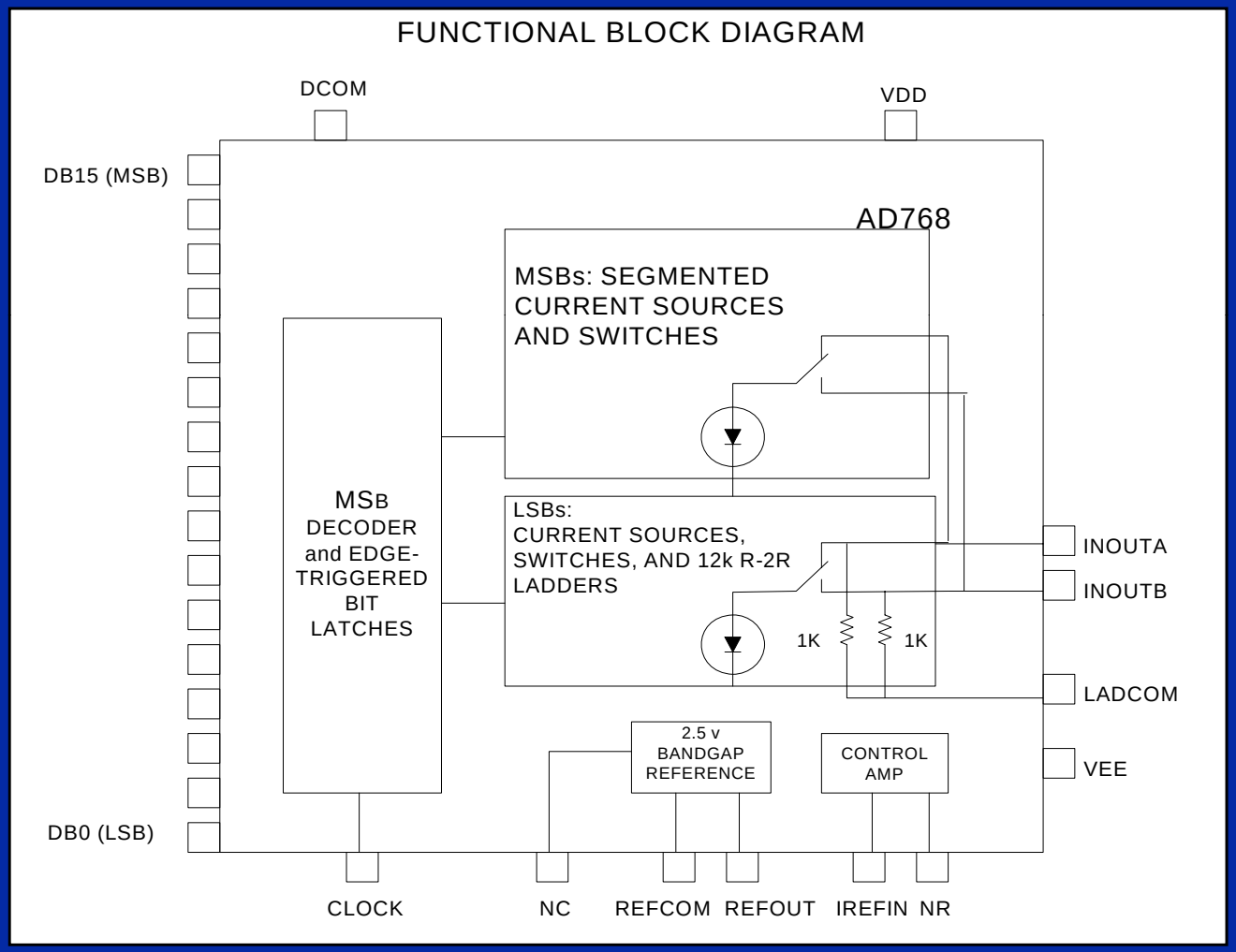
**FEATURES:**
**30 msps Update Rate**
**16-Bit Resolution**
**Linearity: 1/2 LSB DNL @ 14 Bits**
**1 LSB INL @ 14 Bits**
**Fast Settling: 25ns Full-Scale Settling to 0.025%**
**SFDR @ 1 MHZ Output: 86 dBc**
**THD @ 1 MHZ Output: 71 dBc**
**Low Glitch Impulse: 35 pV-s**
**Power Dissipation: 465 mW**
**On-chip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

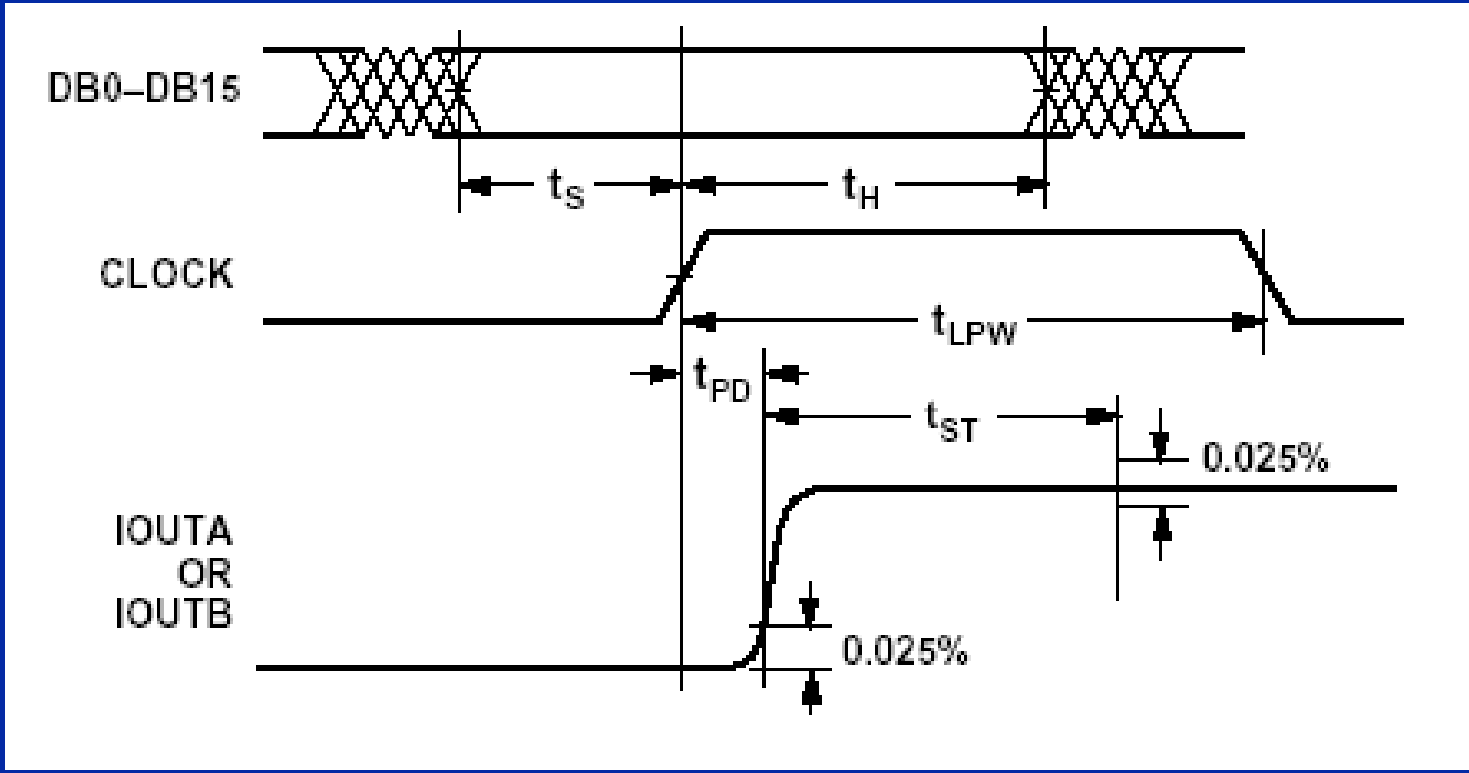
- **Functional Block Diagram:**




- **AD768 data sheet**

# EMIF Case Study: AD768 DAC

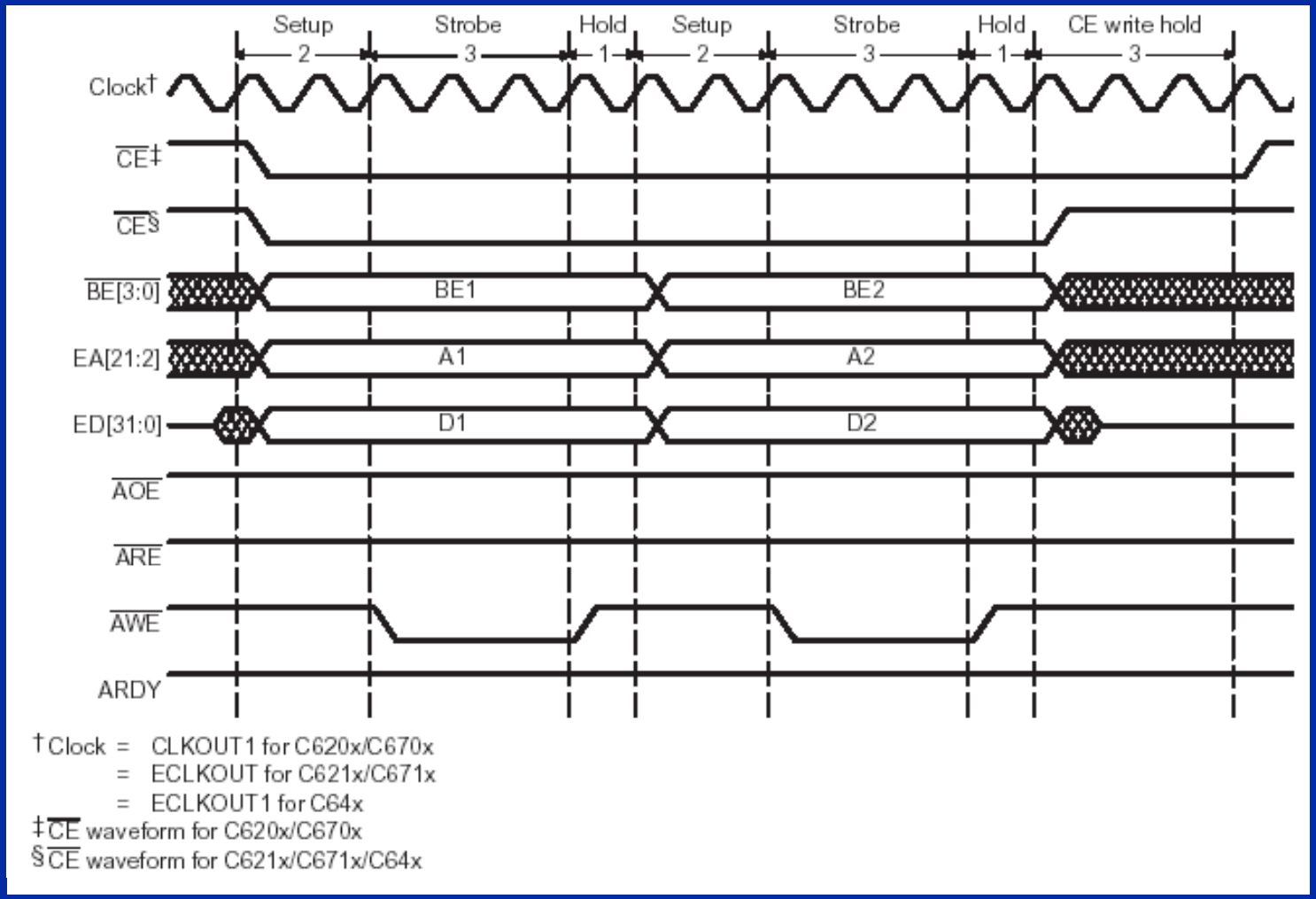
- **Timing:**



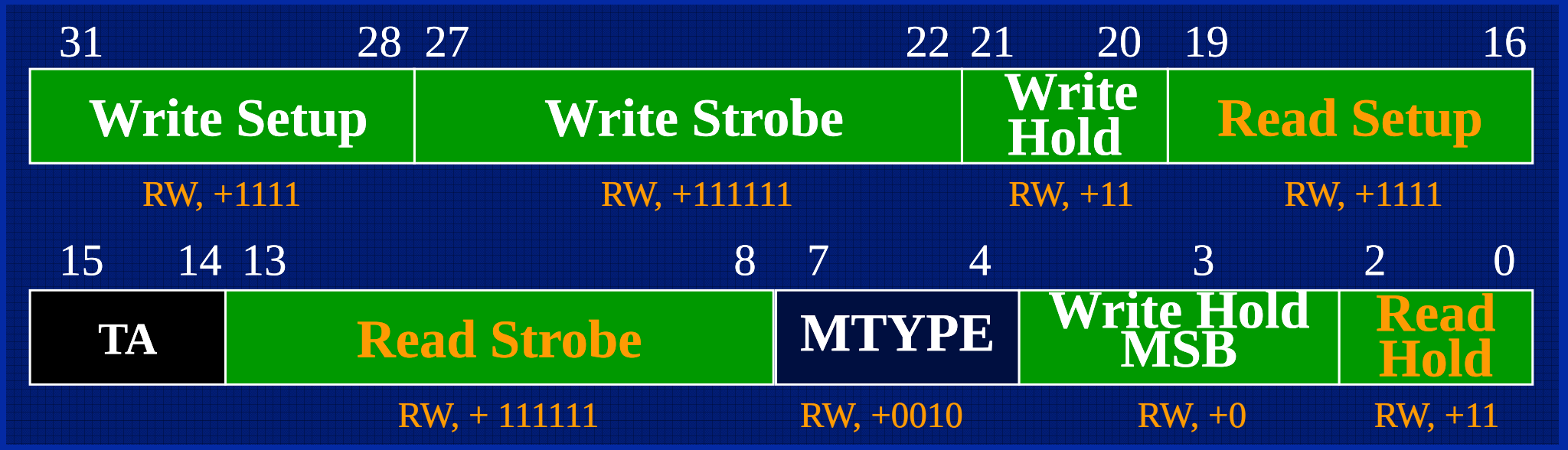

- **AD768 data sheet**

# EMIF Case Study: AD768 DAC

- **C6711 Asynchronous Write Timing:**



# Setting Async Timing




## Set CE3 and to 32-bit ASYNC

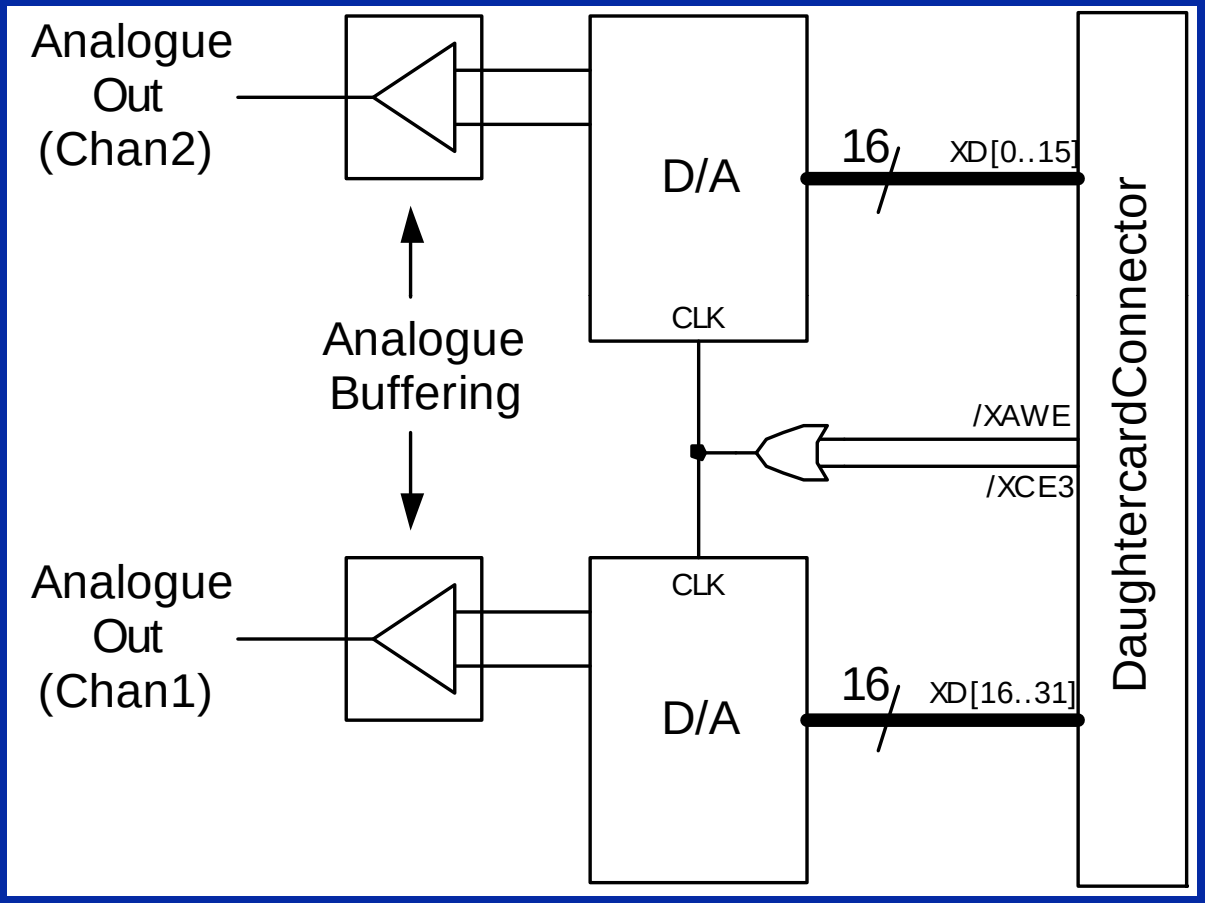
```
CE3     .equ        1800014h
        mvkl.s1     CE3, A0
        mvkh.s1     CE3, A0
        ldw         *A0, A1
        nop 4
        and         A1, 0xff0f, A1
        set         A1, 5, 5, A1
        stw .d1     A1, *A0
```

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: AD768 DAC

- **Hardware Interface:**




- **AD768 data sheet**

# EMIF Case Study: AD9220 ADC

- **Specifications:**

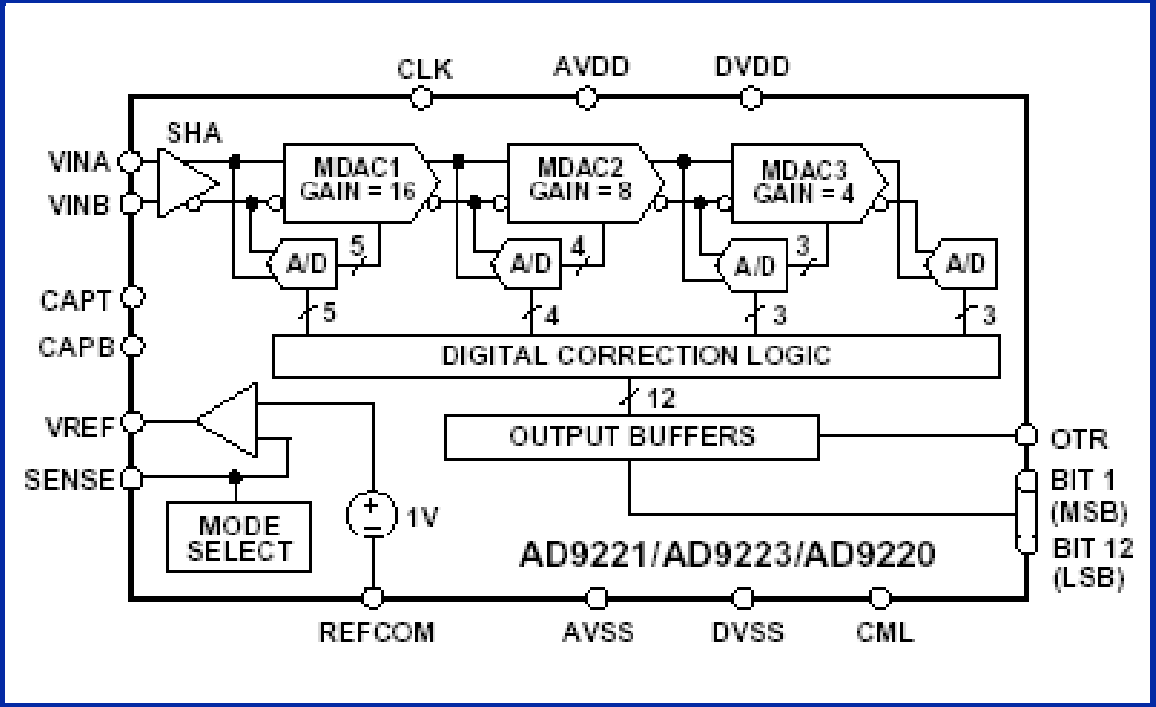
**FEATURES**
**Monolithic 12-Bit A/D Converter Product Family**
**Family Members Are: AD9221, AD9223, and AD9220**
**Flexible Sampling Rates: 1.5 MSPS, 3.0 MSPS and 10 MSPS**
**Low Power Dissipation: 59 mW, 100 mW and 250 mW**
**Single +5V Supply**
**Integral Nonlinearity Error: 0.5 LSB**
**Differential 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

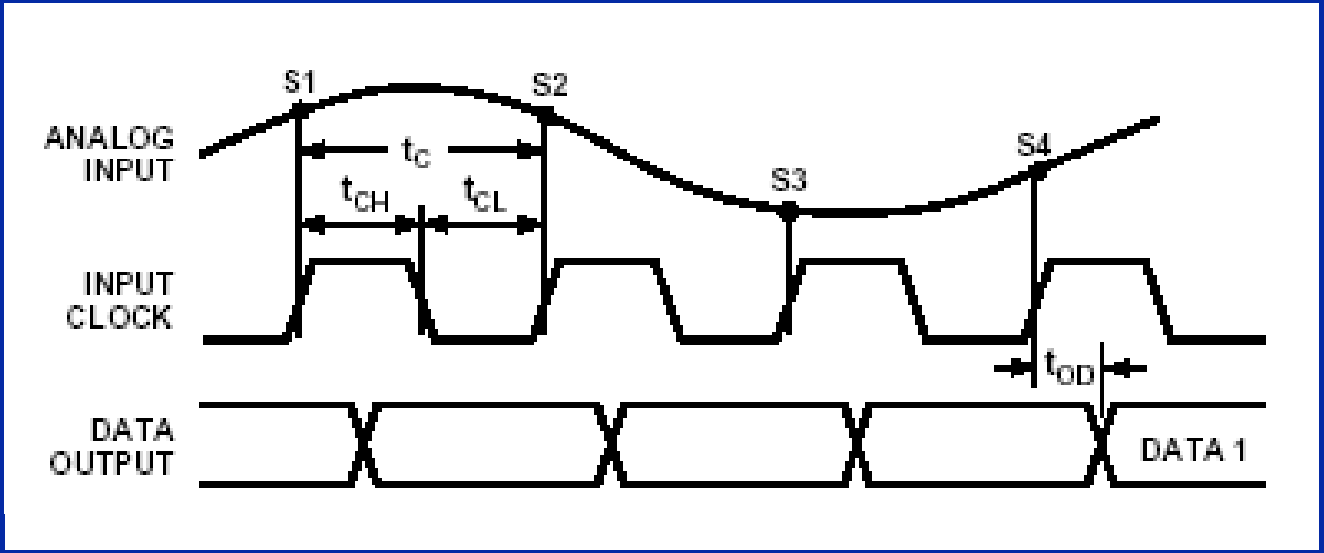
- **Functional Block Diagram:**




- **AD9220 data sheet**

# EMIF Case Study: AD9220 ADC

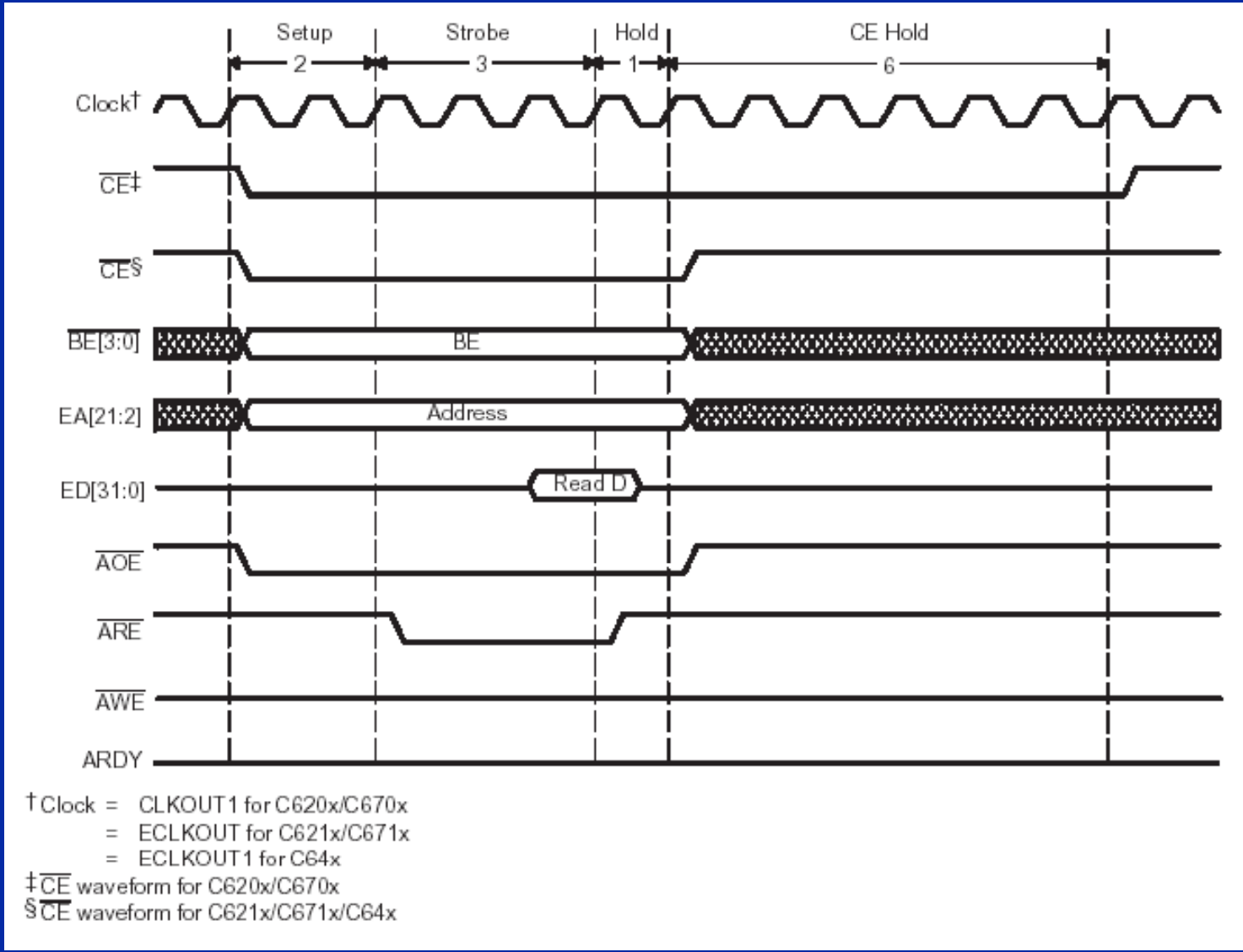
- **Timing:**



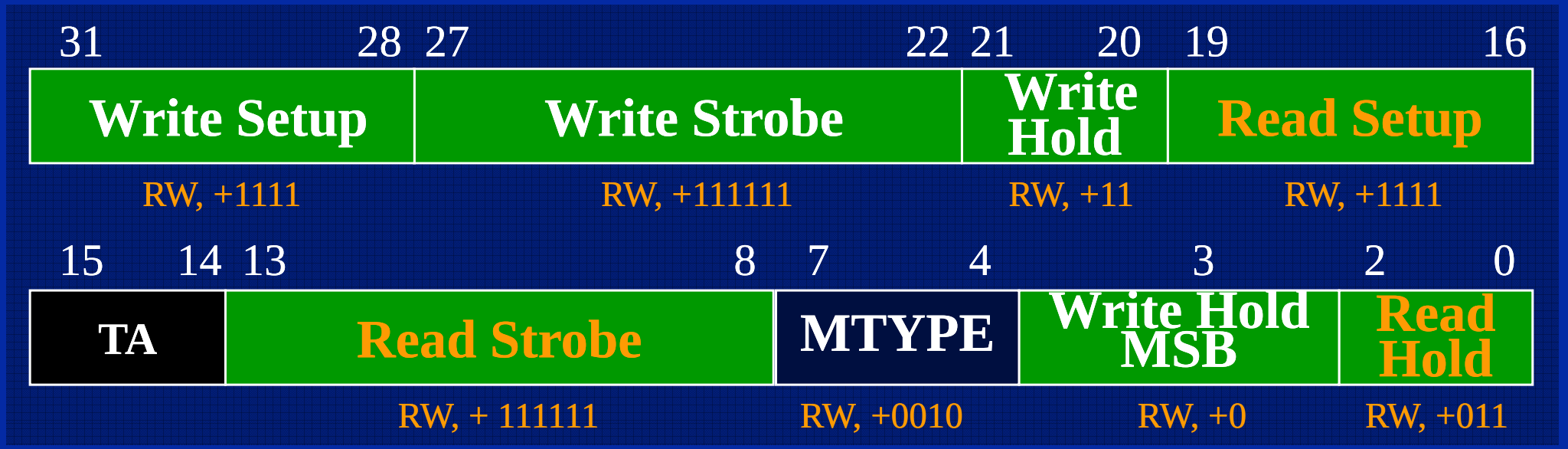

- **AD9220 data sheet**

# EMIF Case Study: AD9220 ADC

- **C6711 Asynchronous Read Timing:**



# Setting Async Timing




## Set CE3 and to 32-bit ASYNC

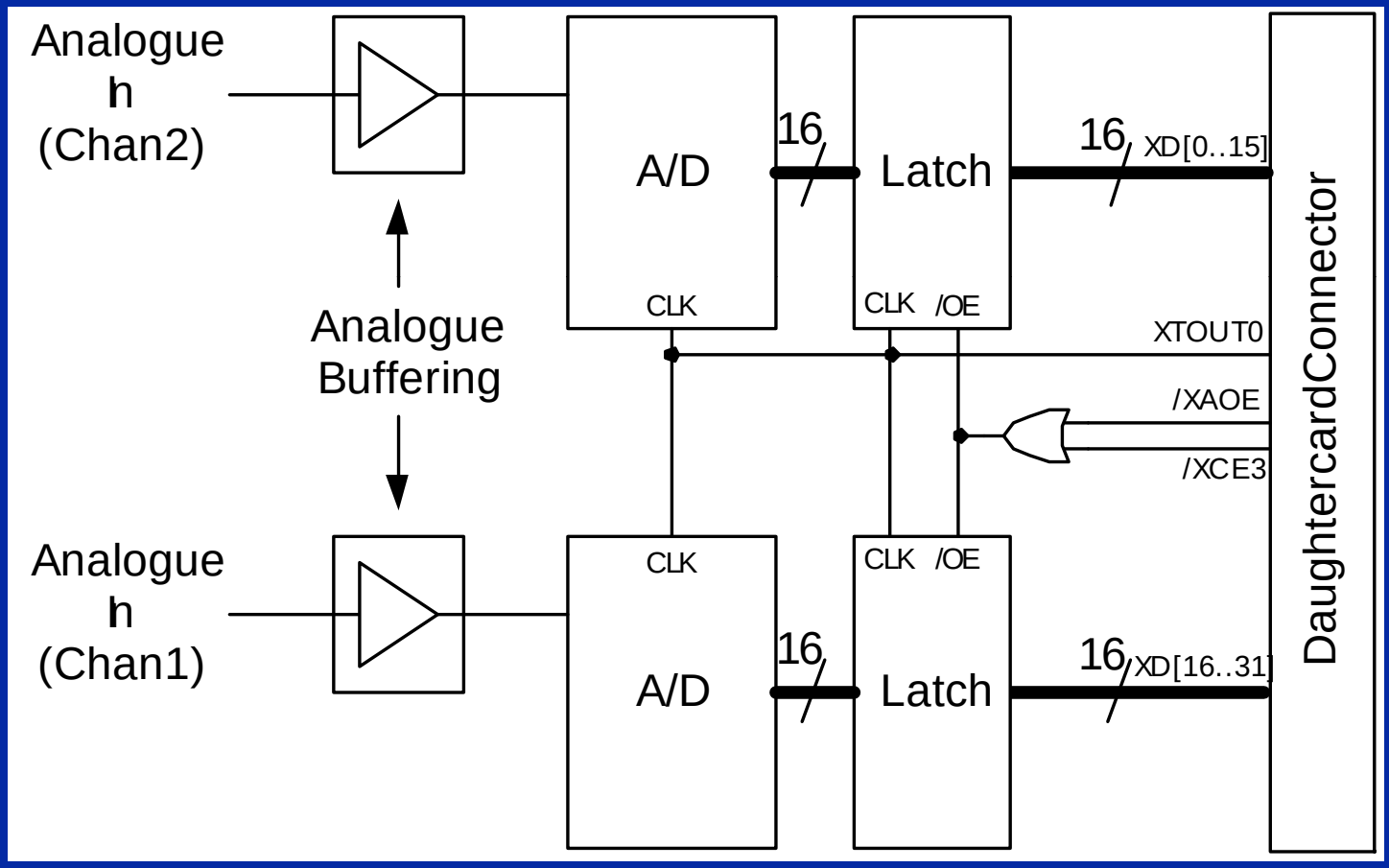
```
CE3     .equ        1800014h
        mvkl.s1     CE3, A0
        mvkh.s1     CE3, A0
        ldw         *A0, A1
        nop 4
        and         A1, 0xff0f, A1
        set         A1, 5, 5, A1
        stw .d1     A1, *A0
```

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

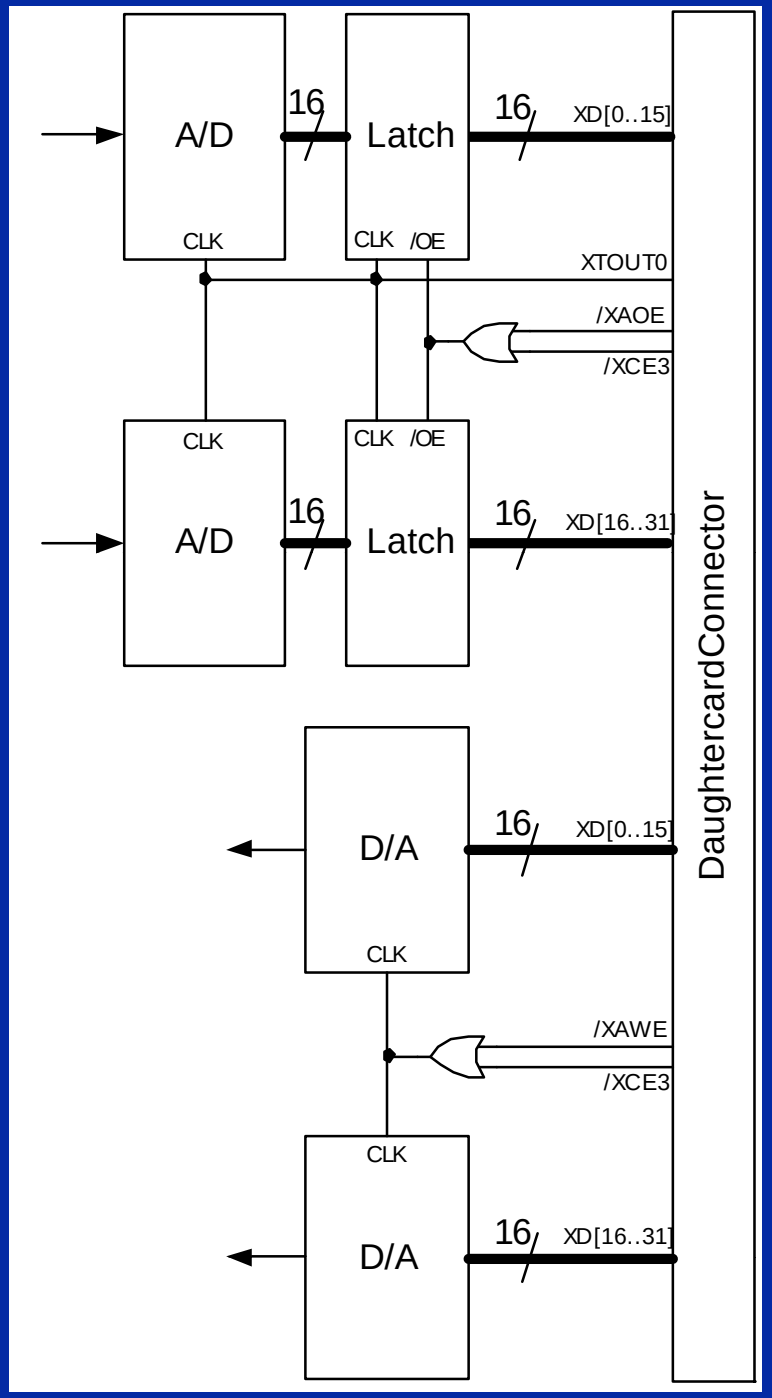
- **Hardware Interface:**




- **AD9220 data sheet**

# EMIF Case Study: Sharing the Bus

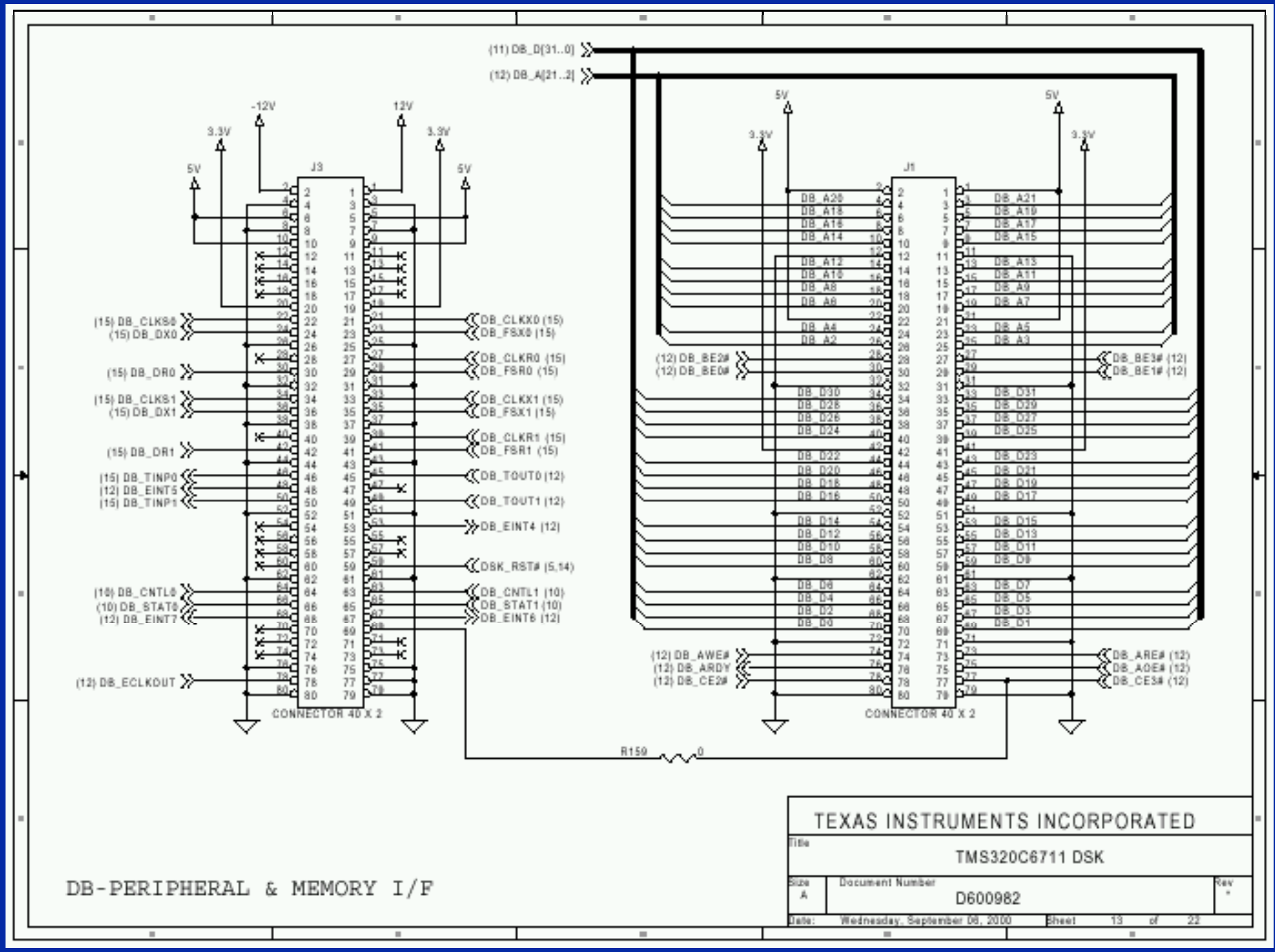
- **Both ADCs and DACs are mapped to the same address space (CE3 = 0xB000 0000).**




| /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



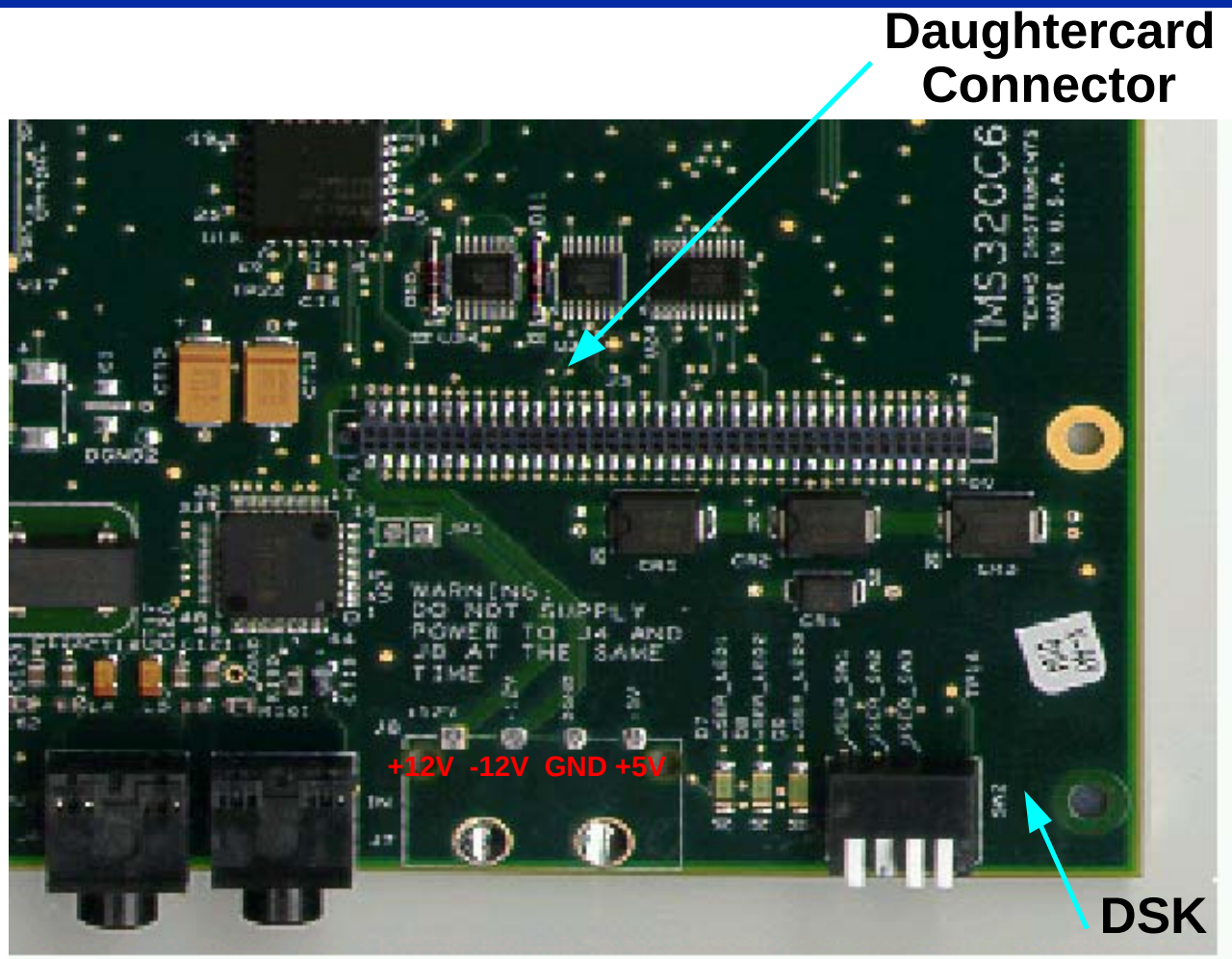
# EMIF Case Study: Hardware

- The INTDSK1115 daughtercard 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 |



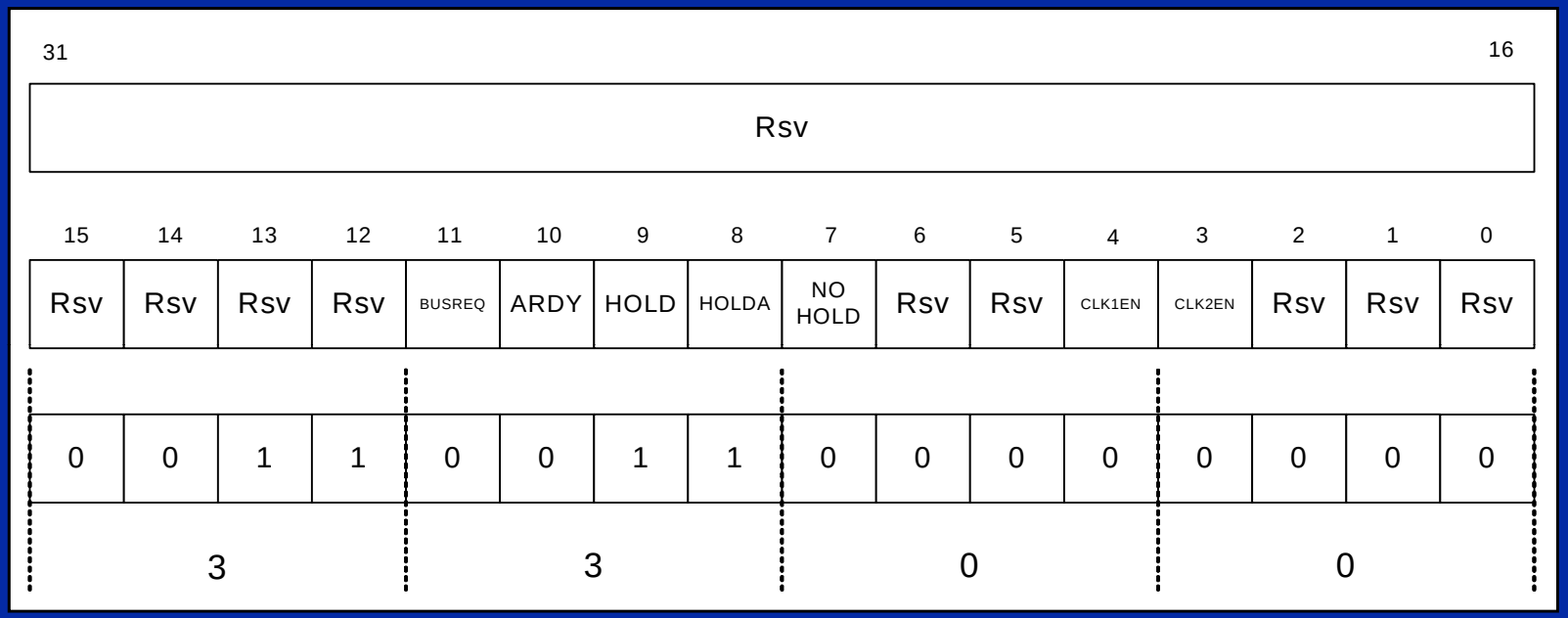

**Warning:** **Do NOT supply power to J4 and J8 at the same time.**

# EMIF Case Study: Software

- **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:




- 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

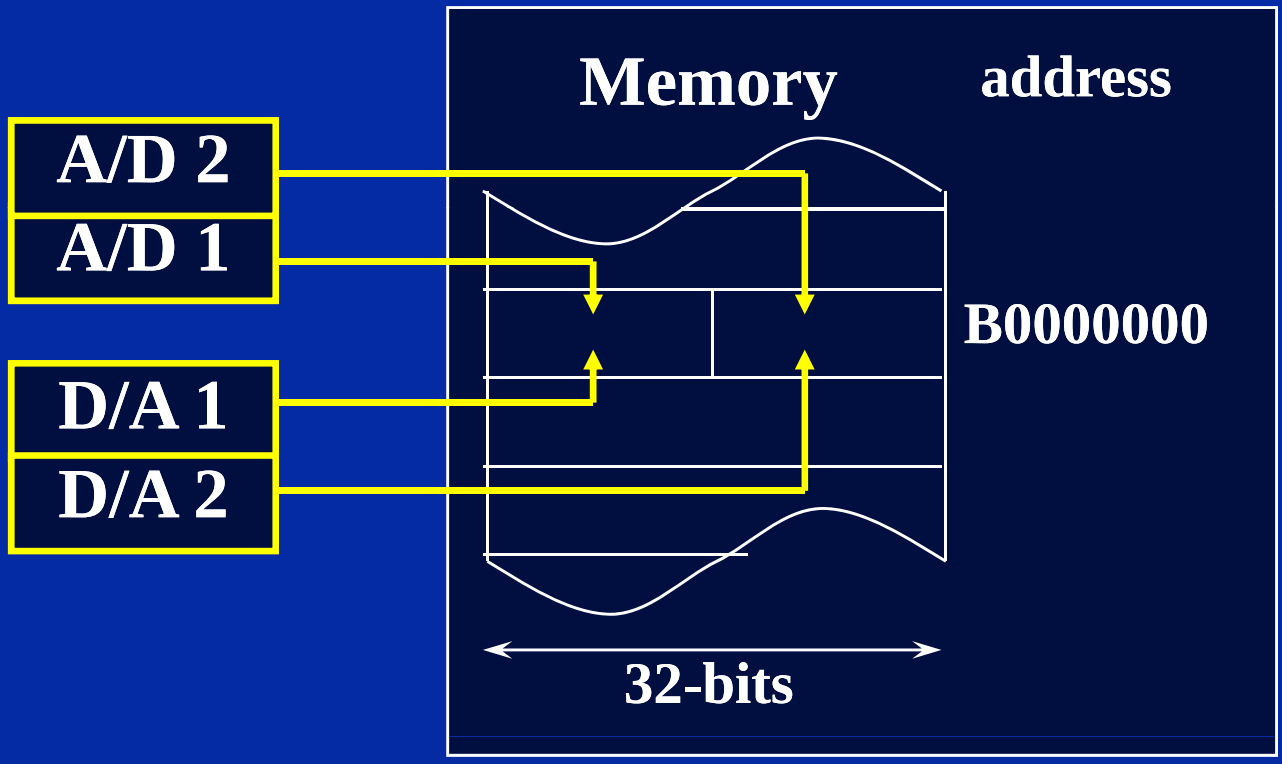
## Setting the CE Control Register:

- 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 Daughtercard 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:**

```
#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

- **The timer can be programmed as follows:**

```
#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: Software - 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:

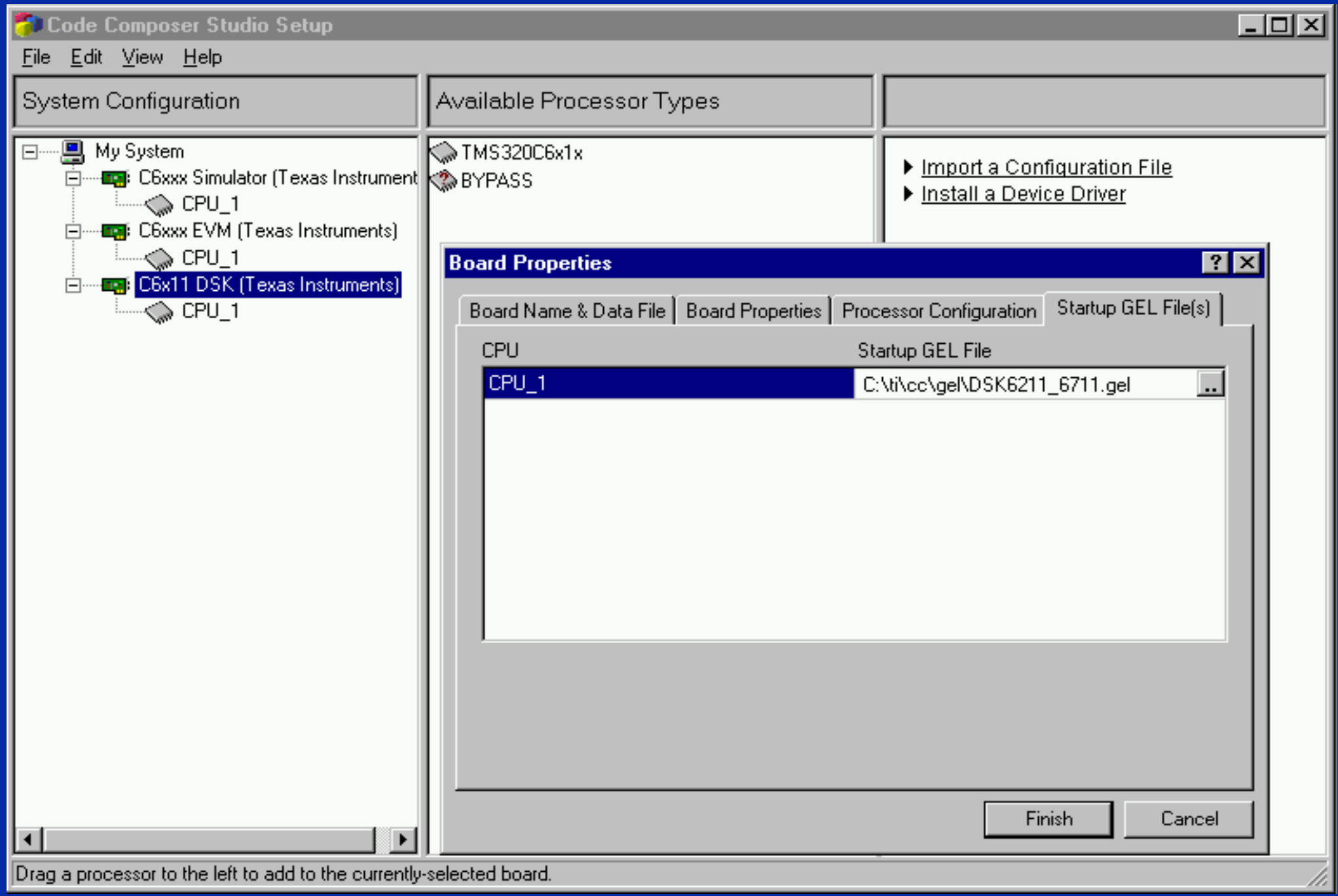
(1) Select Setup CCS.

(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: Software - Loading GEL



# EMIF Case Study: Reading and Writing to the A/D and 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 ();
```

# EMIF Case Study: Setting the Interrupt

**(4) Enable global interrupts:**

```
IRQ_globalEnable ();
```

**Note: (4) has to be done last as it releases all the interrupts.**

# EMIF Case Study: Complete Code

- **Project location:**
  - **\Code\Chapter 08 - EMIF\A_D_D_A_inout**
- **Project name:**
  - **A_D_D_A_inout.pjt**
- **Links:**
  - **\Links\main.pdf**
  - **\Links\Daughtercard Photo.pdf**
  - **\Links\Daughtercard Datasheet.pdf**
  - **\Links\AD768.pdf**
  - **\Links\AD9220.pdf**

# INTDSK1115 Daughtercard Test Code

- **Project location:**
  - **\Code\Chapter 08 - EMIF**
- **Project name:**
  - **Test A/D, D/A and Codec: \TestApp\testapp_bios.pjt**
  - **Test A/D and D/A: \WaveGen\wavegen_bios.pjt**
- **Links:**
  - **\Links\Test Procedure.pdf**

# Chapter 8

# External Memory Interface (EMIF)

- End -