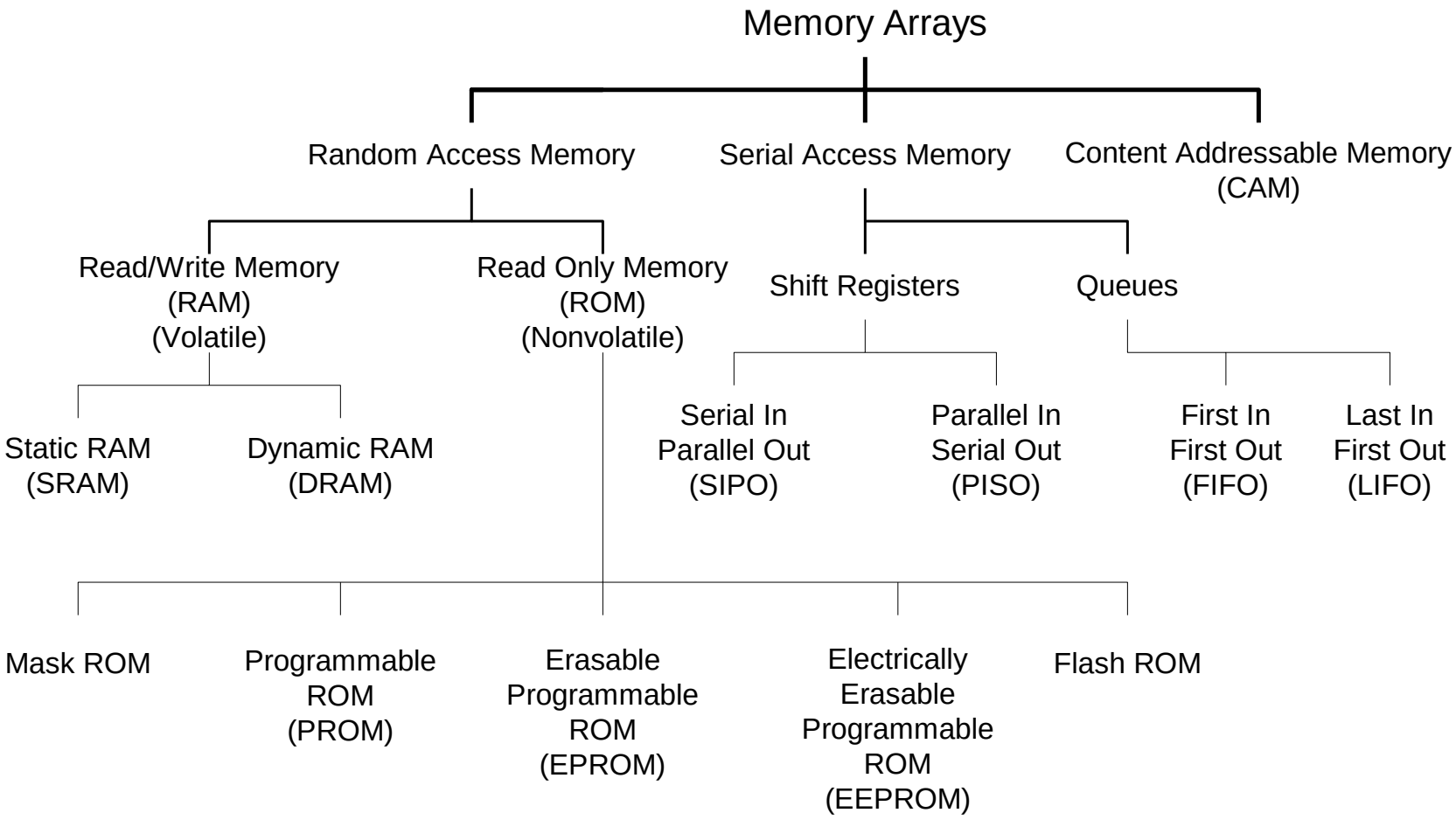
Memorias de semiconductor

Index

- **4.1.** Types of memories, definitions…
- **4.2.** Dynamic RAM (DRAM): cell, logic diagram, structure, logical organization, R/W timing, timing parameters, types.
- **4.3.** Synchronous DRAM (SDRAM):Bandwidth, access time, R/W burst, commands, EMC signals, EMC registers, EMC connection examples.
- **4.4.** DUAL-PORT Memory: block diagram, cell, arbitration, timing diagram, EMC connection examples.
- **4.5.** FIFO Memory: block diagram, expansion, EMC connection examples.

4.1. Semiconductor Memories



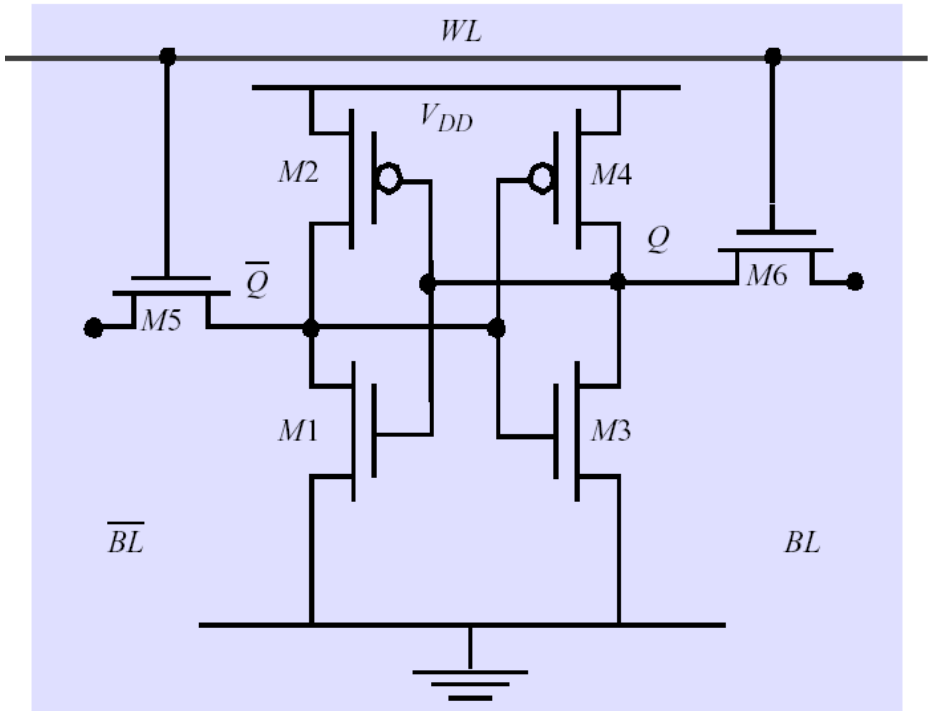
4.1. Definitions

- Memory Interfaces for Accessing Data
  - **Asynchronous** (unclocked): A change in the address results in data appearing
  - **Synchronous** (clocked): A change in address, followed by an edge on CLK results in data appearing or write operation occurring.

- A common arrangement is to have synchronous write operations and asynchronous read operations.
- Volatile:
  - Looses its state when the power goes off.
- Nonvolatile:
  - Retains it state when power goes off.

4.1. Static RAM (SRAM)

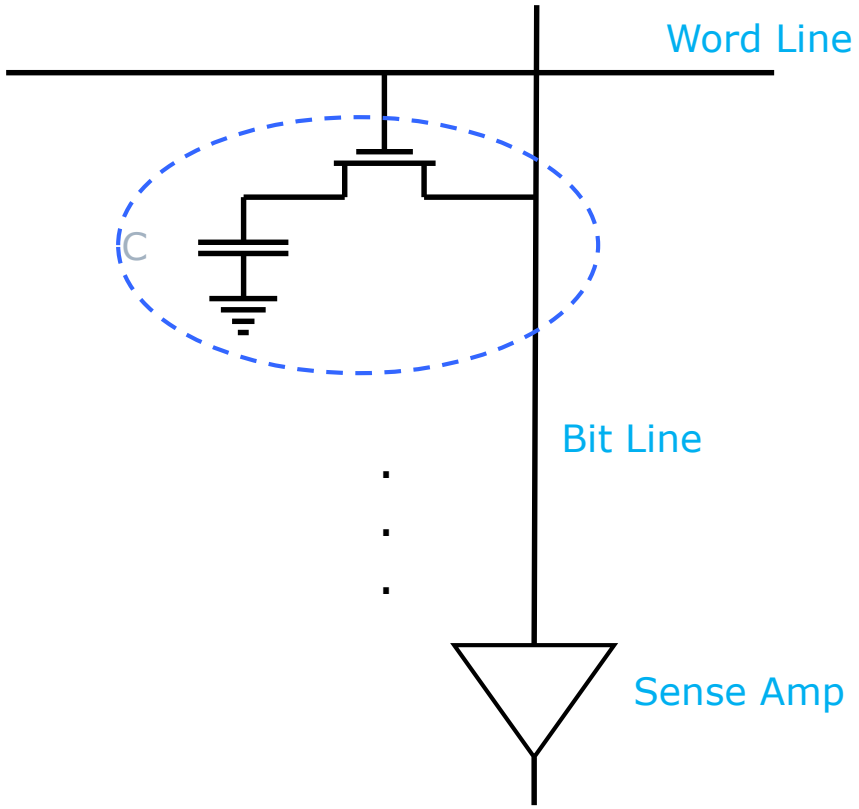
- Six transistors in cross connected fashion
  - Provides regular AND inverted outputs
  - Implemented in CMOS process



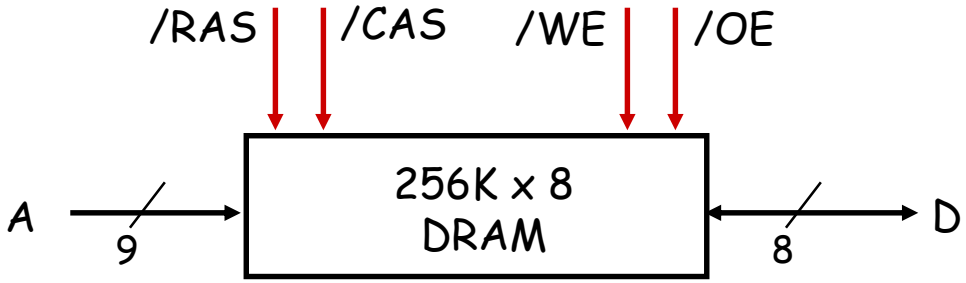
Single Port 6-T SRAM Cell

4.2. DRAM: Dynamic RAM

- SRAM cells exhibit high speed/poor density
- DRAM: simple transistor/capacitor pairs in high density form

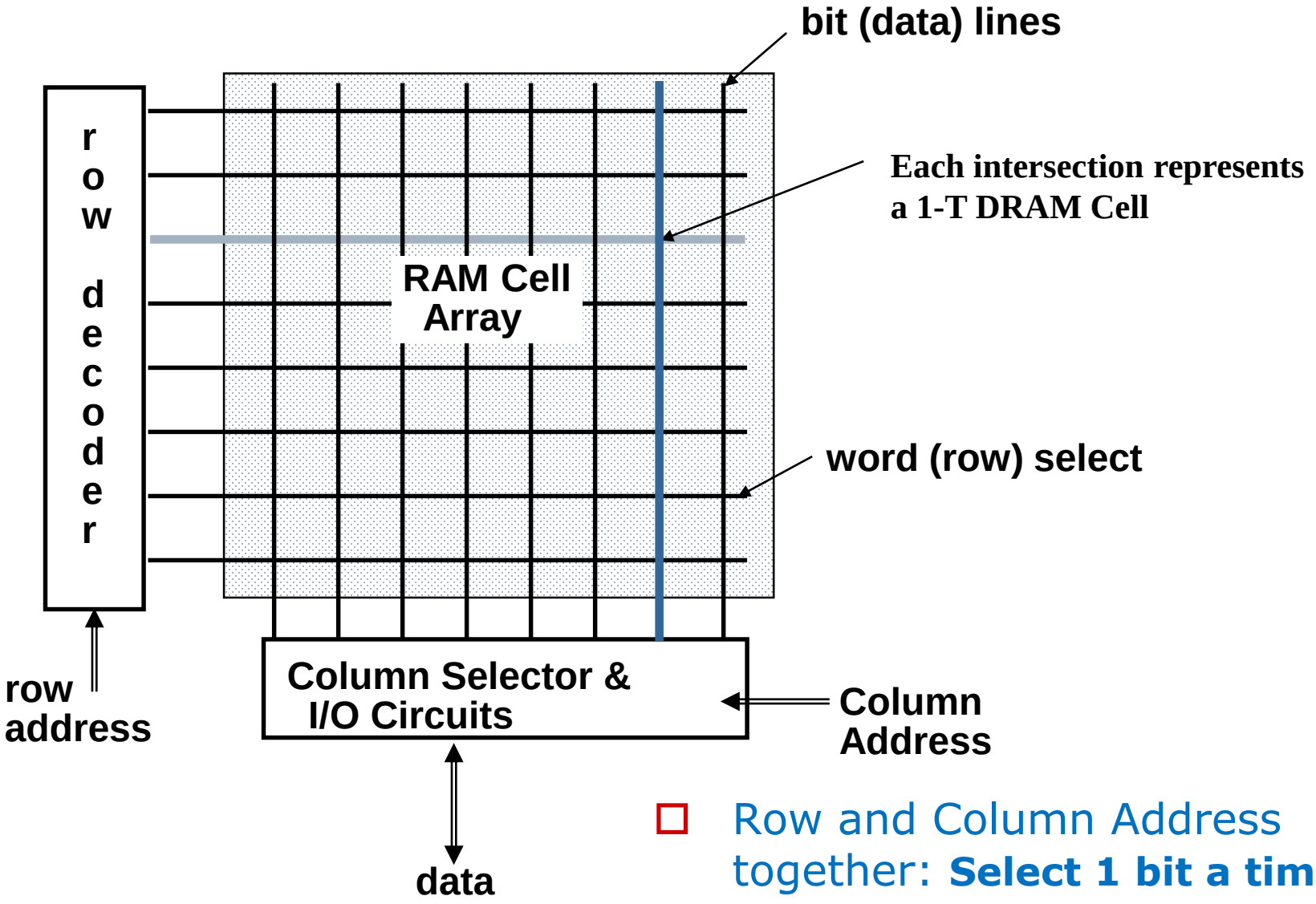


4.2. DRAM: Logic Diagram



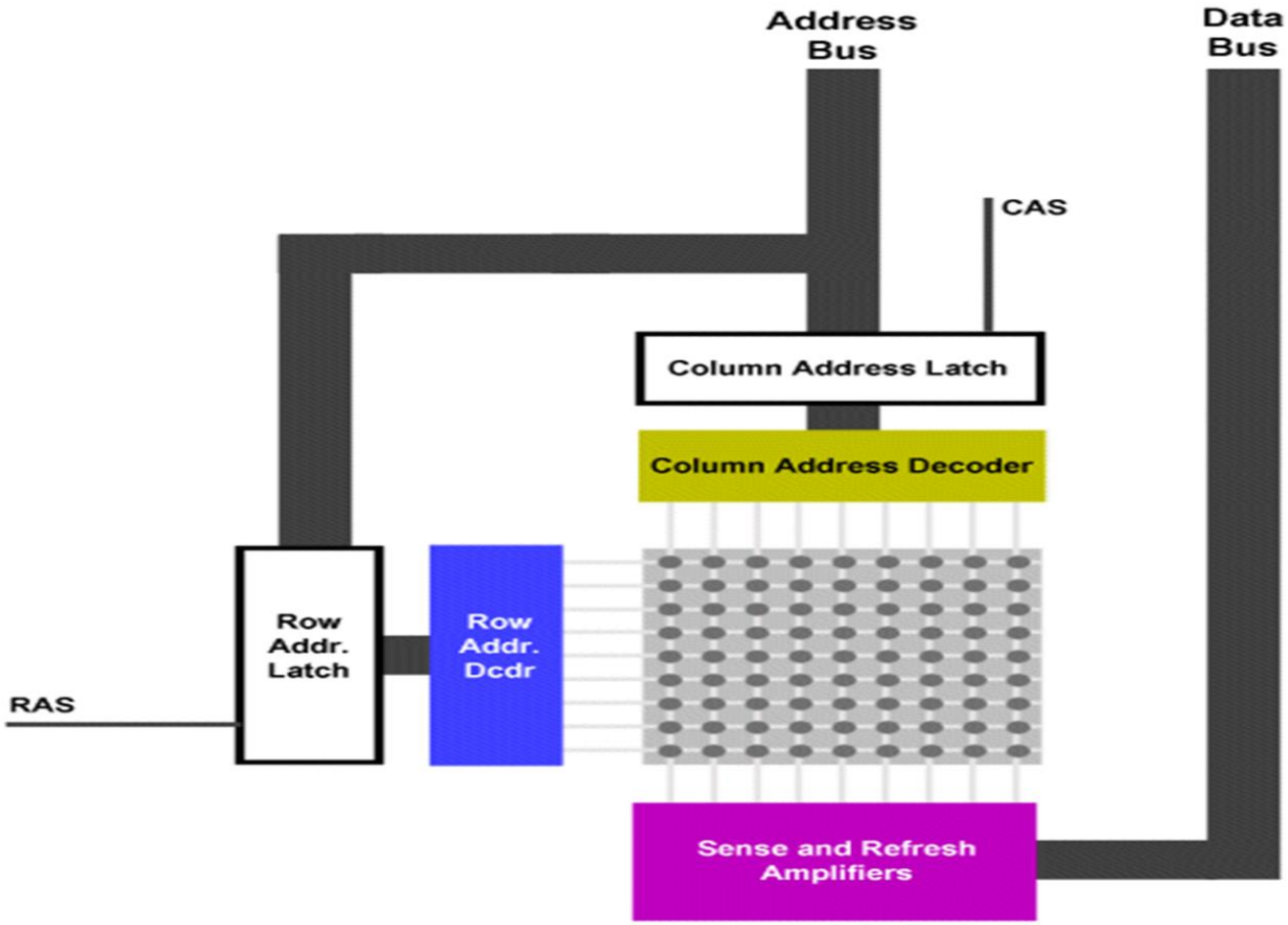
- Control Signals (/RAS, /CAS, /WE, /OE) are all **active low**
- Din and Dout are combined (D):
  - /WE is asserted (Low), /OE is disasserted (High)
    - D serves as the data input pin
  - /WE is disasserted (High), /OE is asserted (Low)
    - D is the data output pin
- Row and column addresses share the same pins (A)
  - /RAS goes low: Pins A are latched in as row address
  - /CAS goes low: Pins A are latched in as column address
  - RAS/CAS edge-sensitive

4.2. Structure of Memory Chip

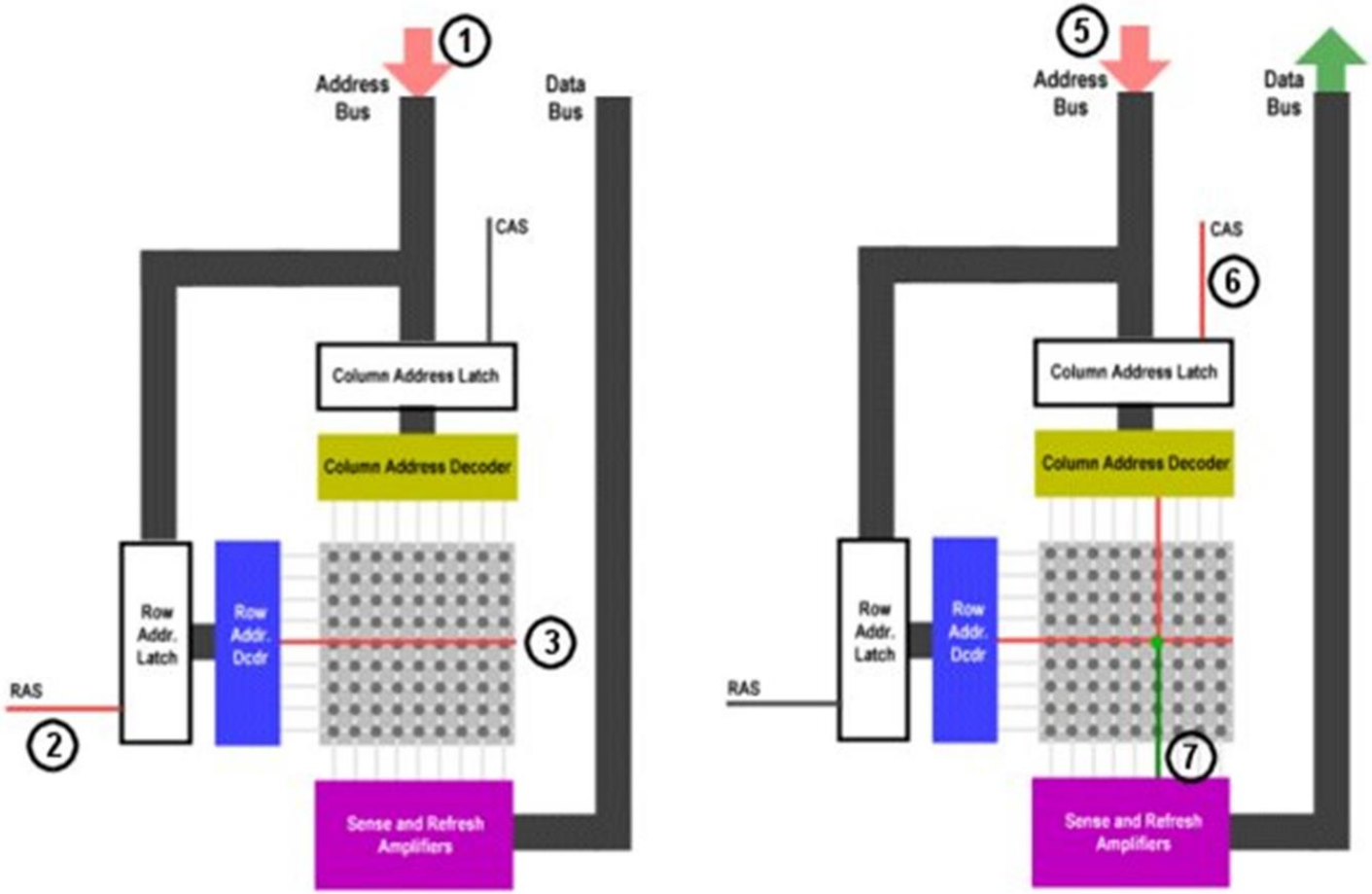


- Row and Column Address together: **Select 1 bit a time**

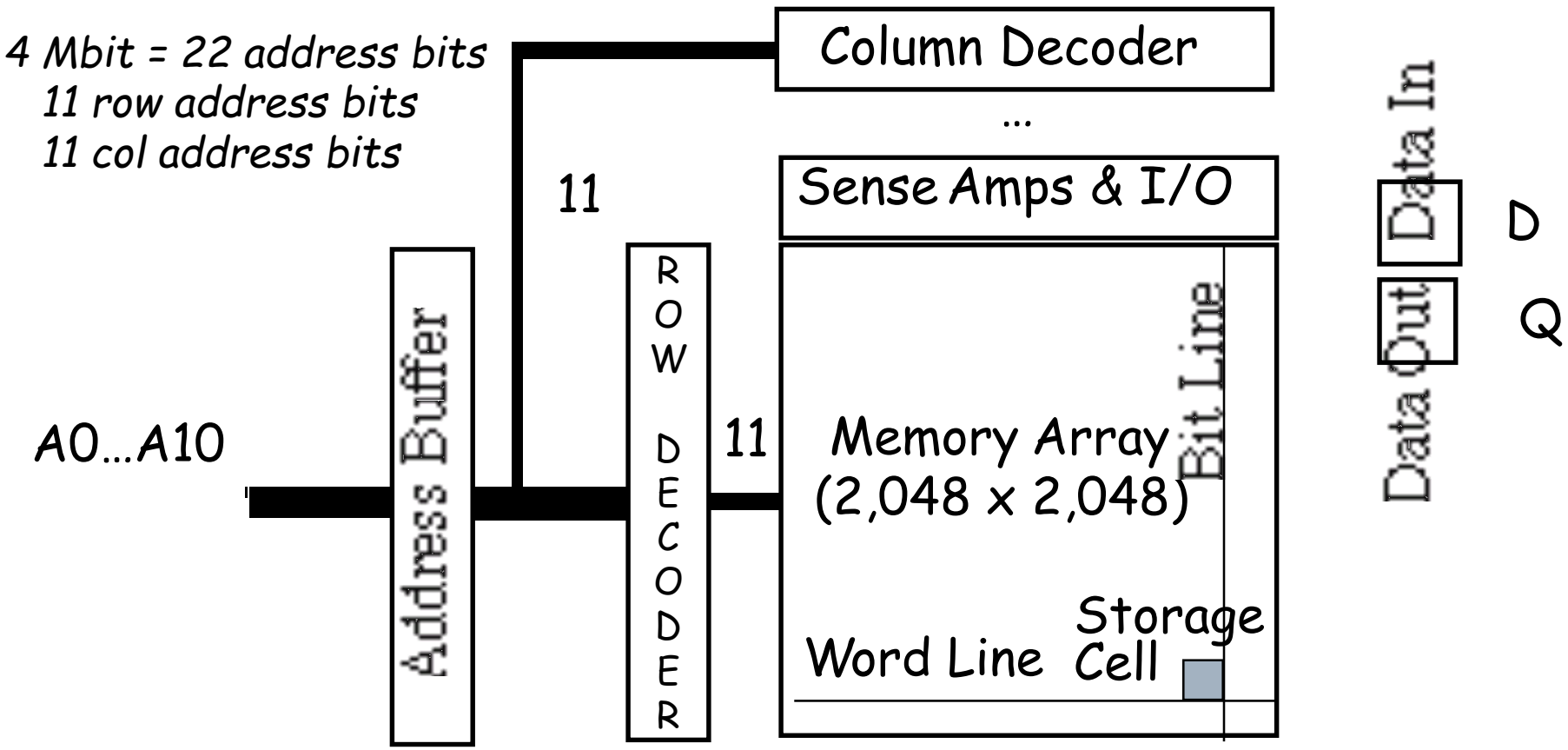
4.2. DRAM: Structure of Memory Chip



4.2. DRAM: Structure of Memory Chip



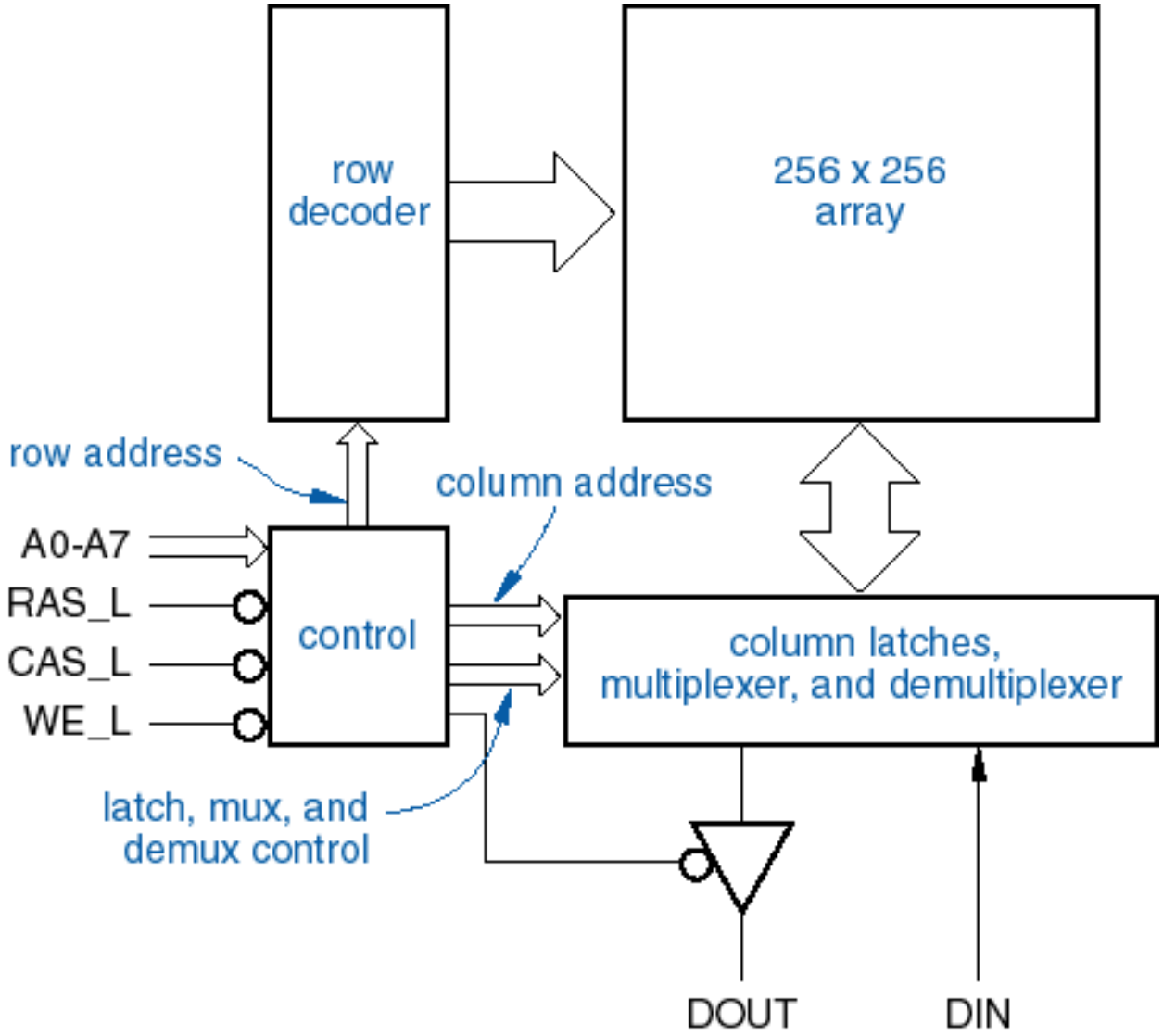
4.2. DRAM Logical Organization (4Mbit)



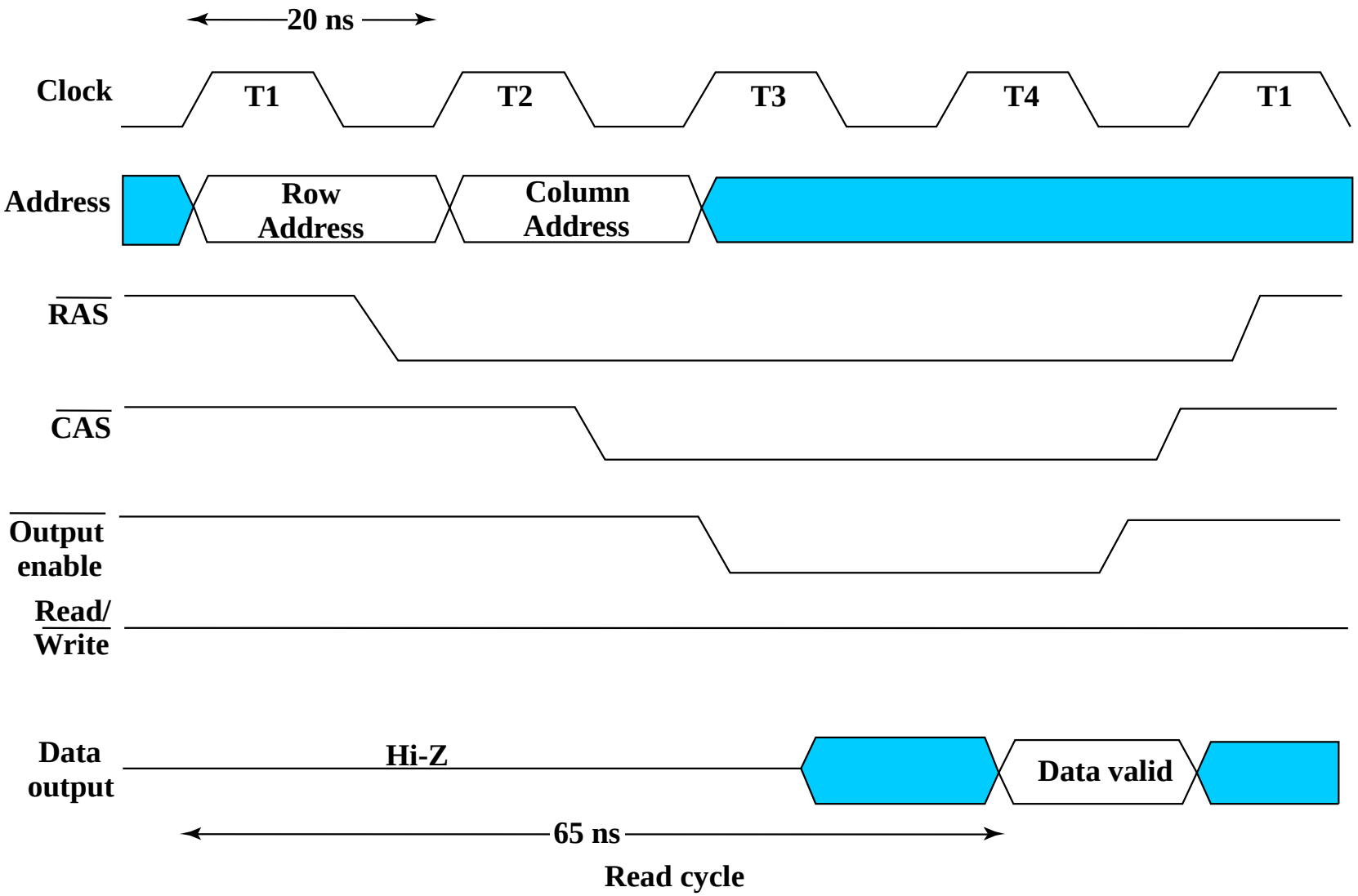
- Square root of bits per RAS/CAS
  - Row selects 1 row of 2048 bits from 2048 rows
  - Col selects 1 bit out of 2048 bits in such a row

4.2. DRAM Logical Organization (64Kx1)

64K x 1
DRAM

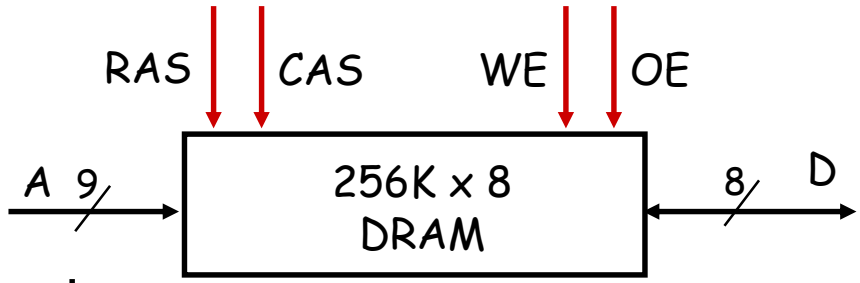


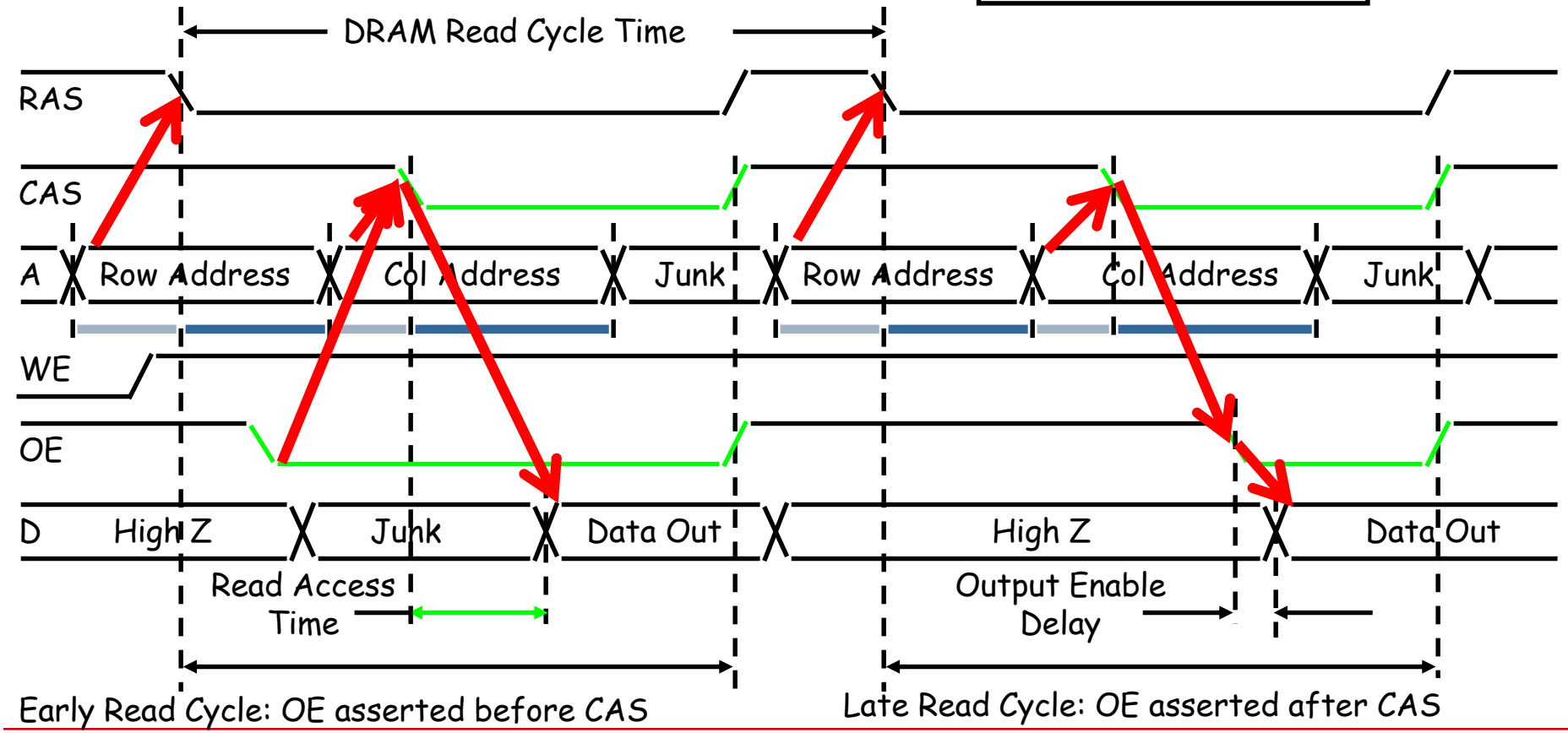
4.2. DRAM Read Timing (I)



4.2. DRAM Read Timing (II)

- Every DRAM access begins at:
  - Assertion of the RAS
  - 2 ways to read: early or late v. CAS





4.2. DRAM Early Read Sequencing

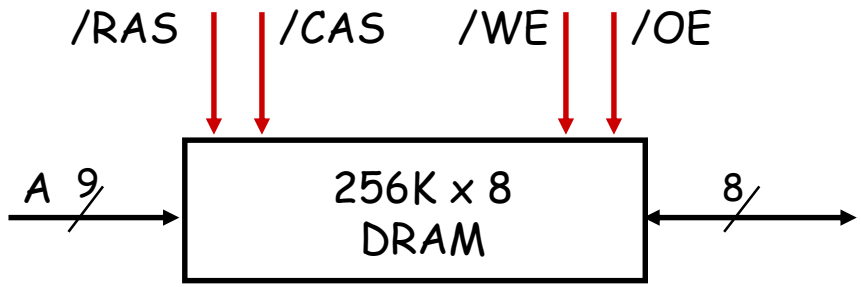
- Assert **Row Address**
- Assert /**RAS**
  - Commence read cycle
  - Meet Row Addr setup time before RAS/hold time after RAS
- Assert /**OE**
- Assert **Col Address**
- Assert /**CAS**
  - Meet Col Addr setup time before CAS/hold time after CAS
- Valid Data Out after access time
- Disassert OE, CAS, RAS to end cycle

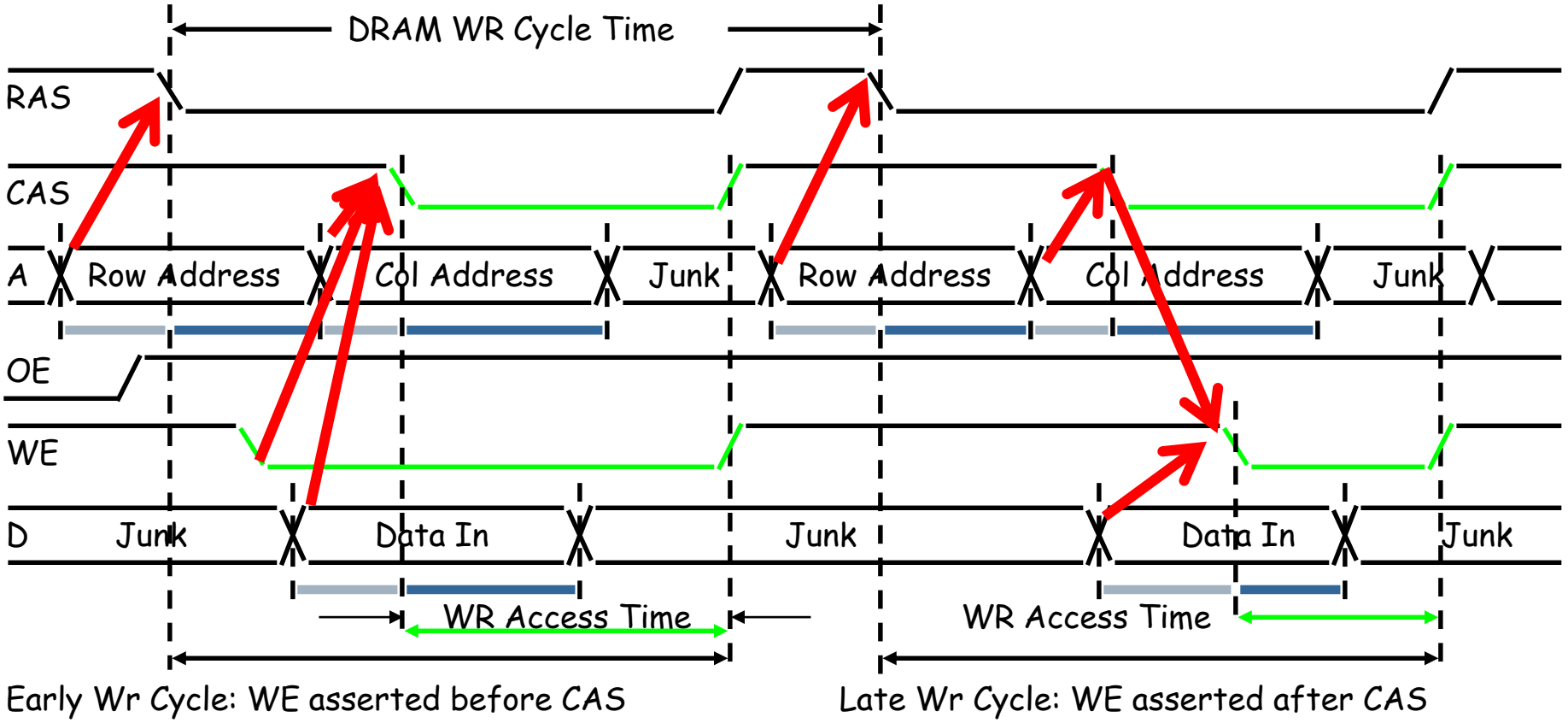
4.2. DRAM Late Read Sequencing

- Assert **Row Address**
- Assert /**RAS**
  - Commence read cycle
  - Meet Row Addr setup time before RAS/hold time after RAS
- Assert **Col Address**
- Assert /**CAS**
  - *Meet Col Addr setup time before CAS/hold time after CAS*
- Assert /**OE**
- Valid **Data Out** after access time
- Disassert OE, CAS, RAS to end cycle

4.2. DRAM Write Timing

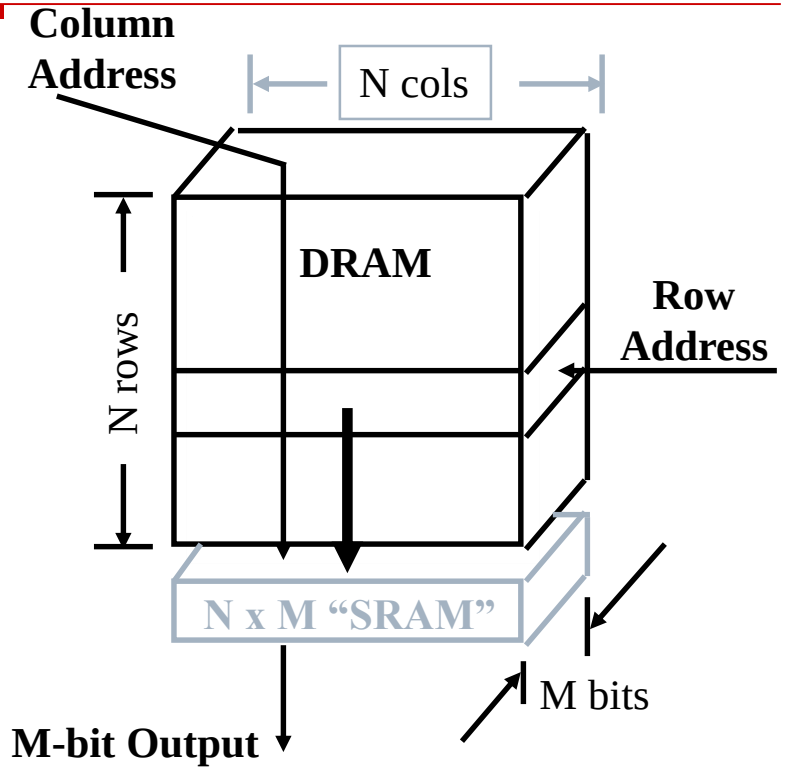
- Every DRAM access begins at:
  - The assertion of the RAS_L
  - 2 ways to write: early or late v. CAS

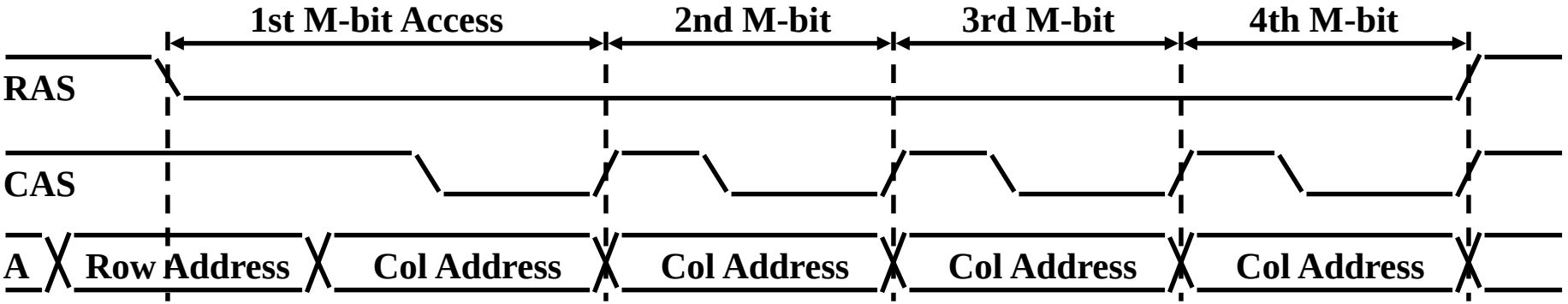




4.2. DRAM: Fast Page Mode Operation

- Regular DRAM Organization:
  - **N** rows x **N** column x **M**-bit.
  - Read & Write M-bit at a time.
  - Each M-bit access requires a RAS / CAS cycle.
- Fast Page Mode DRAM
  - N x M "SRAM" to save a row.
- After a row is read into the register:
  - Only CAS is needed to access other M-bit blocks on that row.
  - RAS remains asserted while CAS is toggled.





4.2. DRAM Timing Parameters

- t_{RAC}: minimum time from RAS line falling to the valid data output.
  - Quoted as the speed of a DRAM when buy.
  - A typical 4Mb DRAM t_{RAC} = 60 ns.
  - Speed of DRAM since on purchase sheet?
- t_{RC}: minimum time from the start of one row access to the start of the next.
  - t_{RC} = 110 ns for a 4Mbit DRAM with a t_{RAC} of 60 ns
- t_{CAC}: minimum time from CAS line falling to valid data output.
  - 15 ns for a 4Mbit DRAM with a t_{RAC} of 60 ns.
- t_{PC}: minimum time from the start of one column access to the start of the next.
  - 35 ns for a 4Mbit DRAM with a t_{RAC} of 60 ns.

4.2. DRAM Types

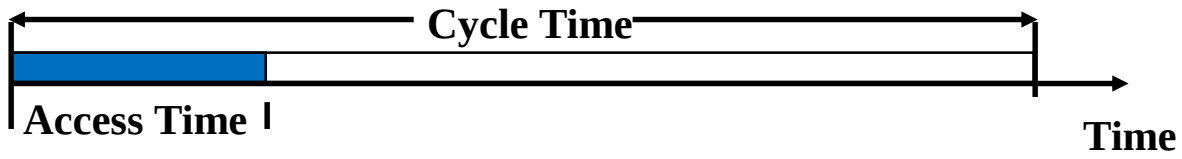
- **EDO** - Extended Data Out (similar to fast-page mode)
  - RAS cycle fetched rows of data from cell array blocks (long access time, around 100ns)
  - Subsequent CAS cycles quickly access data from row buffers if within an address page (page is around 256 Bytes)
- **SDRAM** - Synchronous DRAM
  - Clocked interface.
  - Uses dual banks internally. Start access in one bank then next, then receive data from first then second.
- **DDR** - Double Data Rate SDRAM
  - Uses both rising (positive edge) and falling (negative) edge of clock for data transfer. (typical 100MHz clock with 200 MHz transfer).
- **RDRAM** - Rambus DRAM
  - Entire data blocks are access and transferred out on a high-speed bus-like interface (500 MB/s, 1.6 GB/s).
  - Tricky system level design. More expensive memory chips.

4.3. Synchronous DRAM

- Transfers to and from the DRAM are synchronize with a Clock.
- Synchronous registers appear on:
  - Address input
  - Data input
  - Data output
- Column address counter
  - For addressing internal data to be transferred on each clock cycle beginning with the column address counts up to:
    column address + burst size – 1

- **Example:** Memory data path width: 1 word = 4 bytes
  - **Burst size**: **8 words** = 32 bytes
  - Memory **clock frequency**: **5 ns**
  - **Latency time** (from application of row address until first word available): **4 clock cycles**
  - **Read cycle time**: (4 + 8) x 5 ns = 60 ns
  - **Memory Bandwidth**: $32/(60 \times 10^{-9})$ = 533 Mbytes/sec

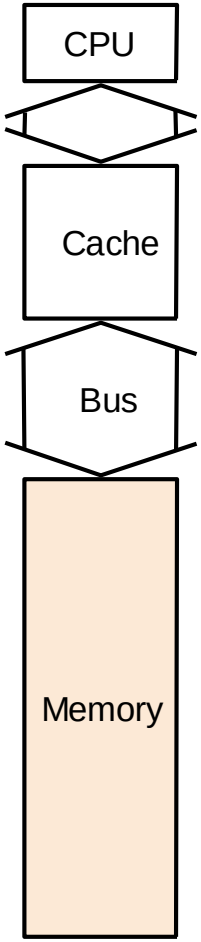
4.3. Main Memory Performance



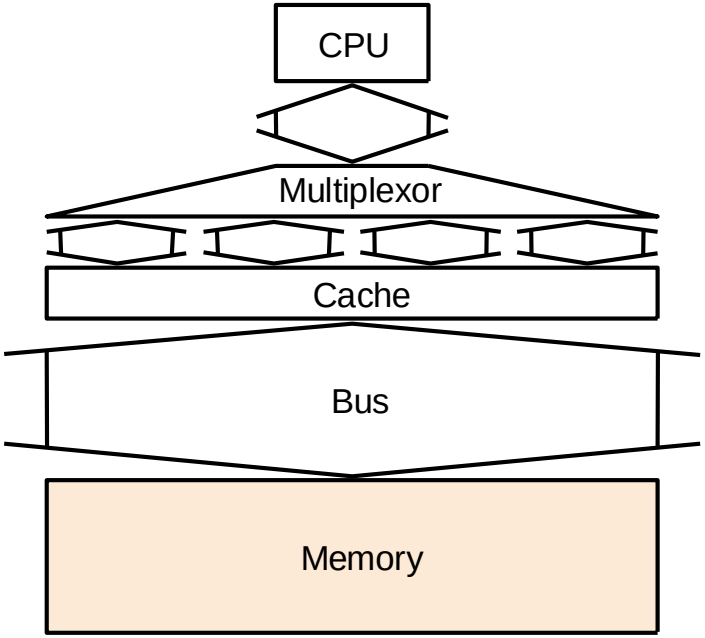
- **DRAM Cycle Time >> DRAM Access Time.**

- DRAM (Read/Write) **Cycle Time** :
  - How frequent can you initiate an access?
  - Analogy: A little kid can only ask his father for money on Saturday.
- DRAM (Read/Write) **Access Time**:
  - How quickly will you get what you want once you initiate an access?
  - Analogy: As soon as he asks, his father will give him the money.
- DRAM Bandwidth Limitation analogy:
  - What happens if he runs out of money on Wednesday?

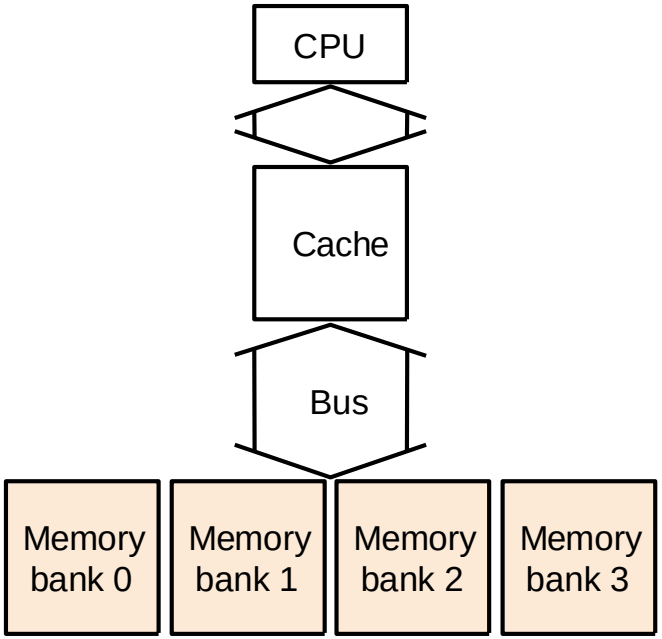
4.3. Main Memory Organizations



One-word wide
memory organization



Wide memory organization

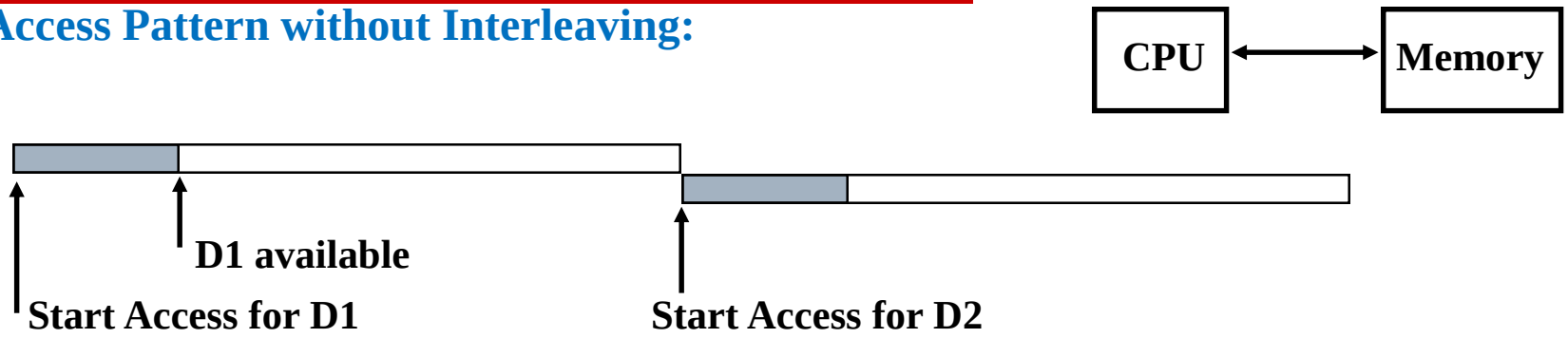


interleaved
memory organization

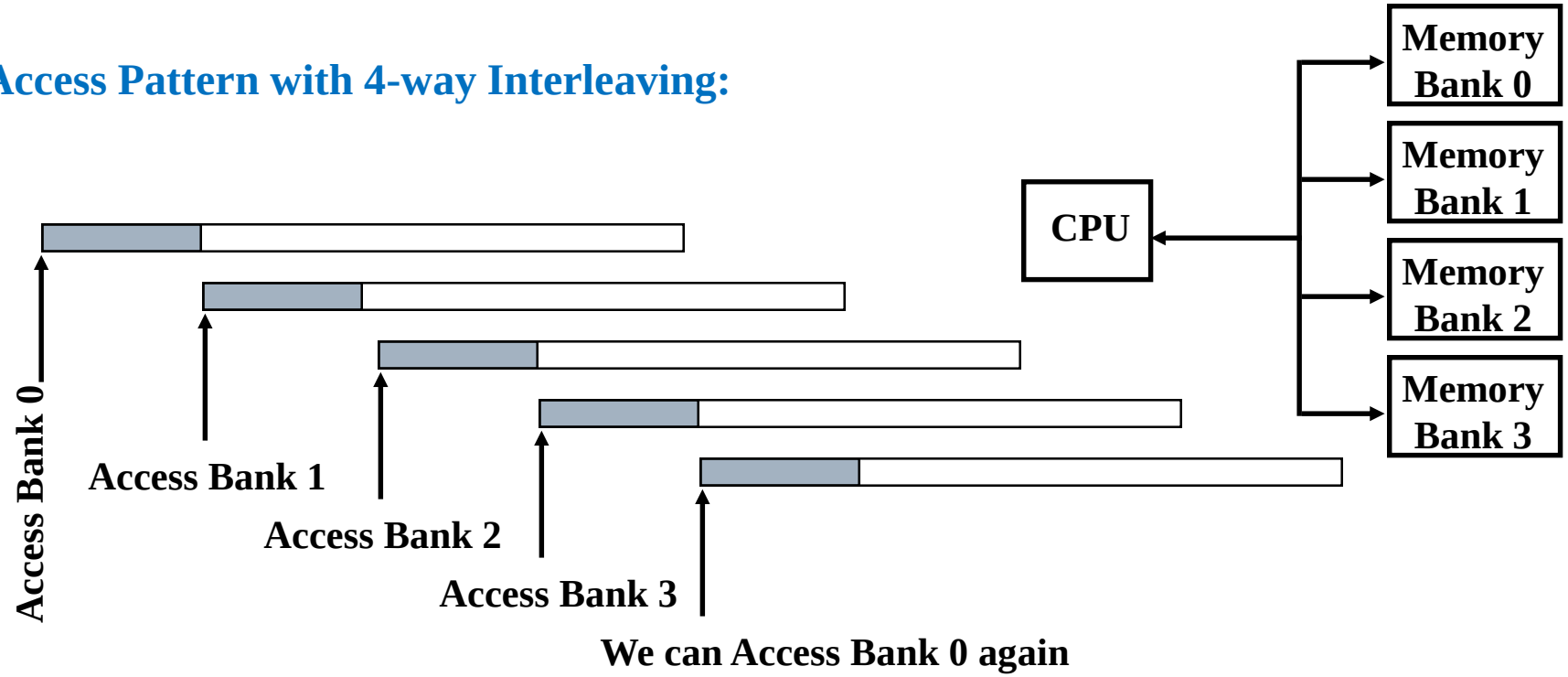
DRAM access time >> bus transfer time

4.3. Increasing Bandwidth: Interleaving

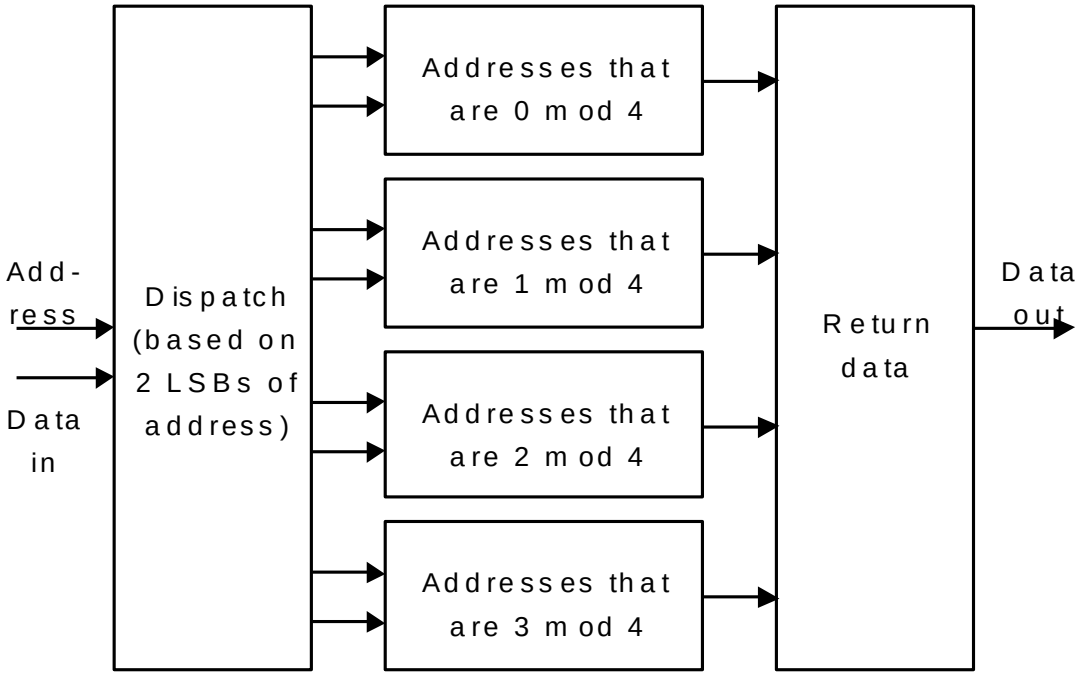
Access Pattern without Interleaving:

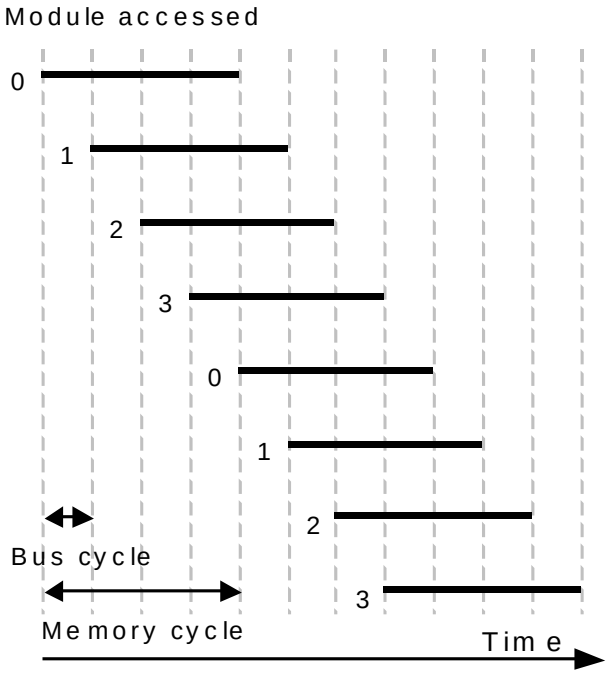


Access Pattern with 4-way Interleaving:



4.3. Increasing Bandwidth: Interleaving





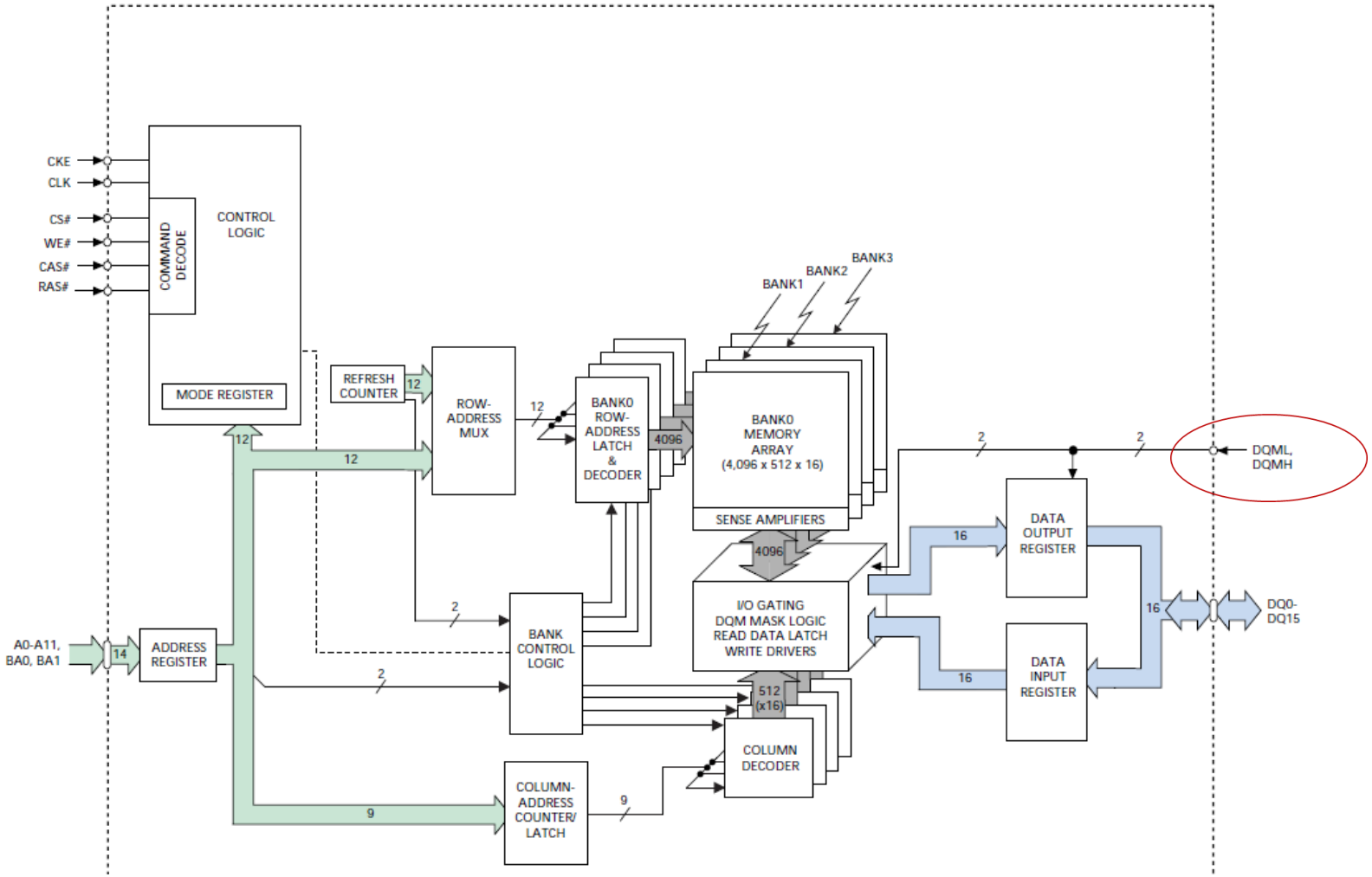
Interleaved memory is more flexible than wide-access memory in that it can handle multiple independent accesses at once.

4.3. Memory Access Time Example

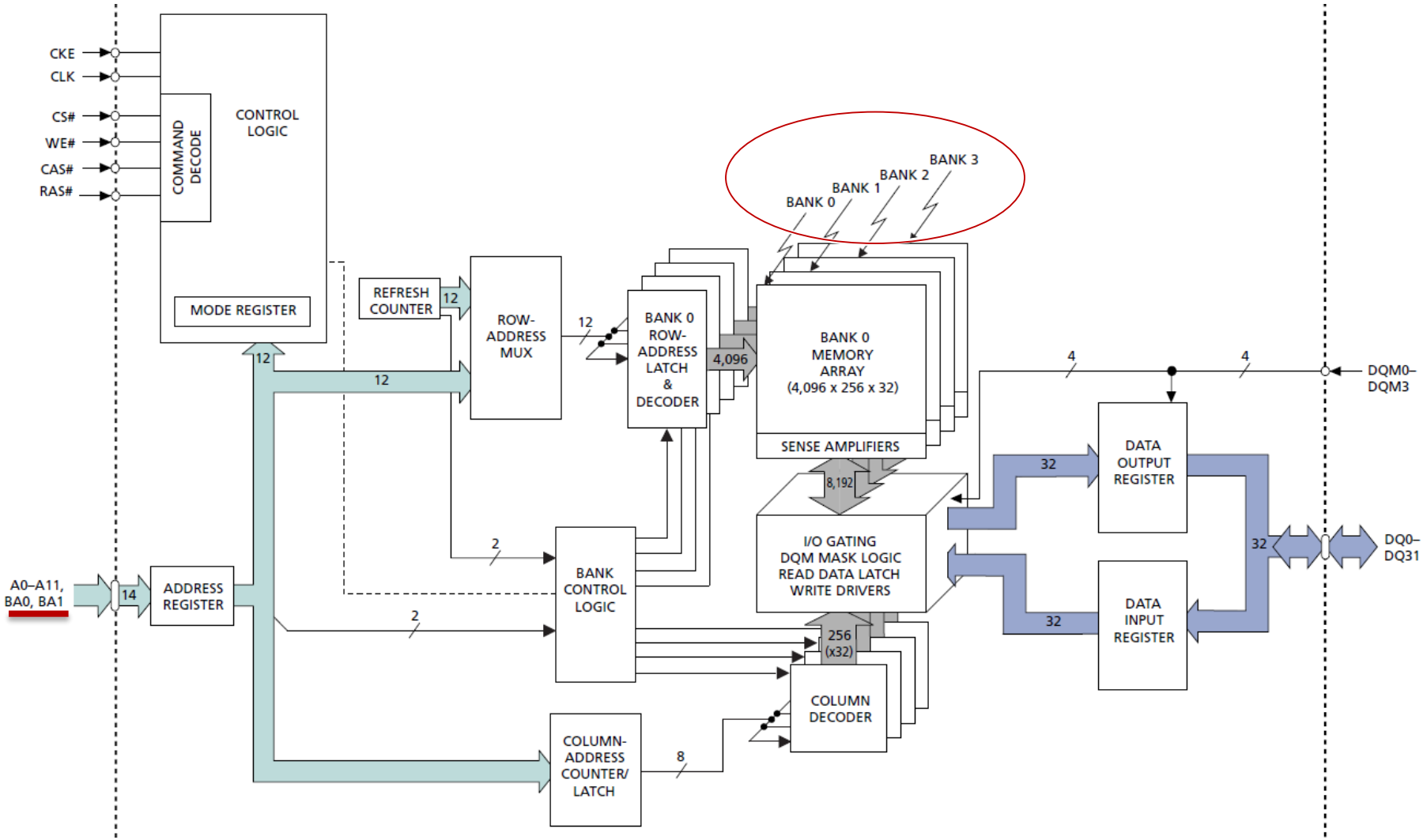
- Assume that it takes 1 cycle to send the address, 15 cycles for each DRAM access and 1 cycle to send a word of data:
  - Assuming a cache block of 4 words and **one-word wide** DRAM, miss penalty = 1 + 4x15 + 4x1 = **65 cycles**.
  - With main memory and **bus width of 2 words**, miss penalty = 1 + 2x15 + 2x1 = **33 cycles**. For **4-word wide memory**, miss penalty is **17 cycles**. Expensive due to wide bus and control circuits.
  - With **interleaved** memory of **4 memory banks** and same bus width, the miss penalty = 1 + 1x15 + 4x1 = **20 cycles**. The memory controller must supply consecutive addresses to different memory banks. Interleaving is universally adapted in high-performance computers.

4.3. SDRAM: MT48LC8M16A2 128Mb: x4, x8, **x16**

FUNCTIONAL BLOCK DIAGRAM
8 Meg x 16 SDRAM



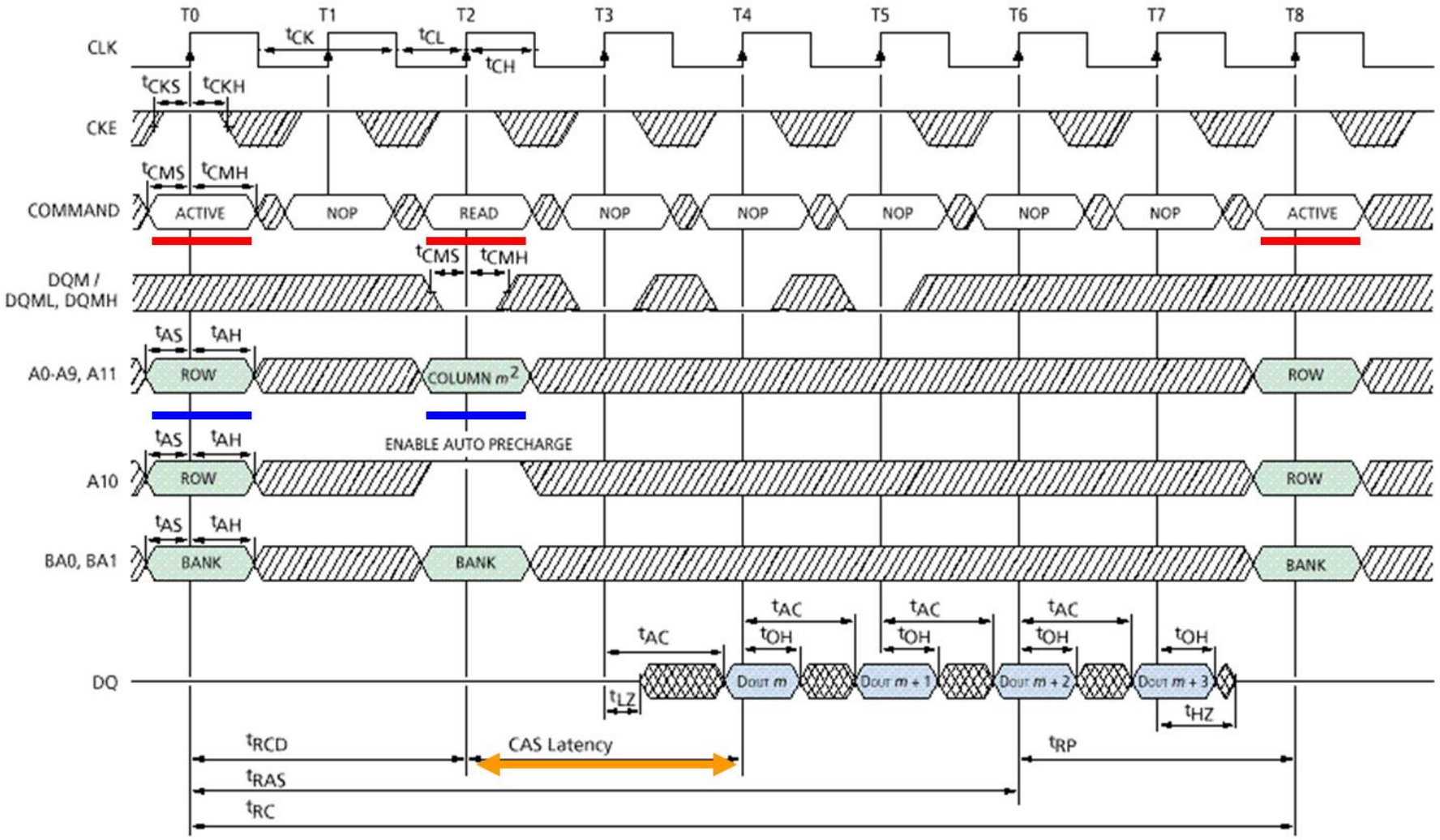
4.3. SDRAM: MT48LC8M32B2 (2Mx32x4)



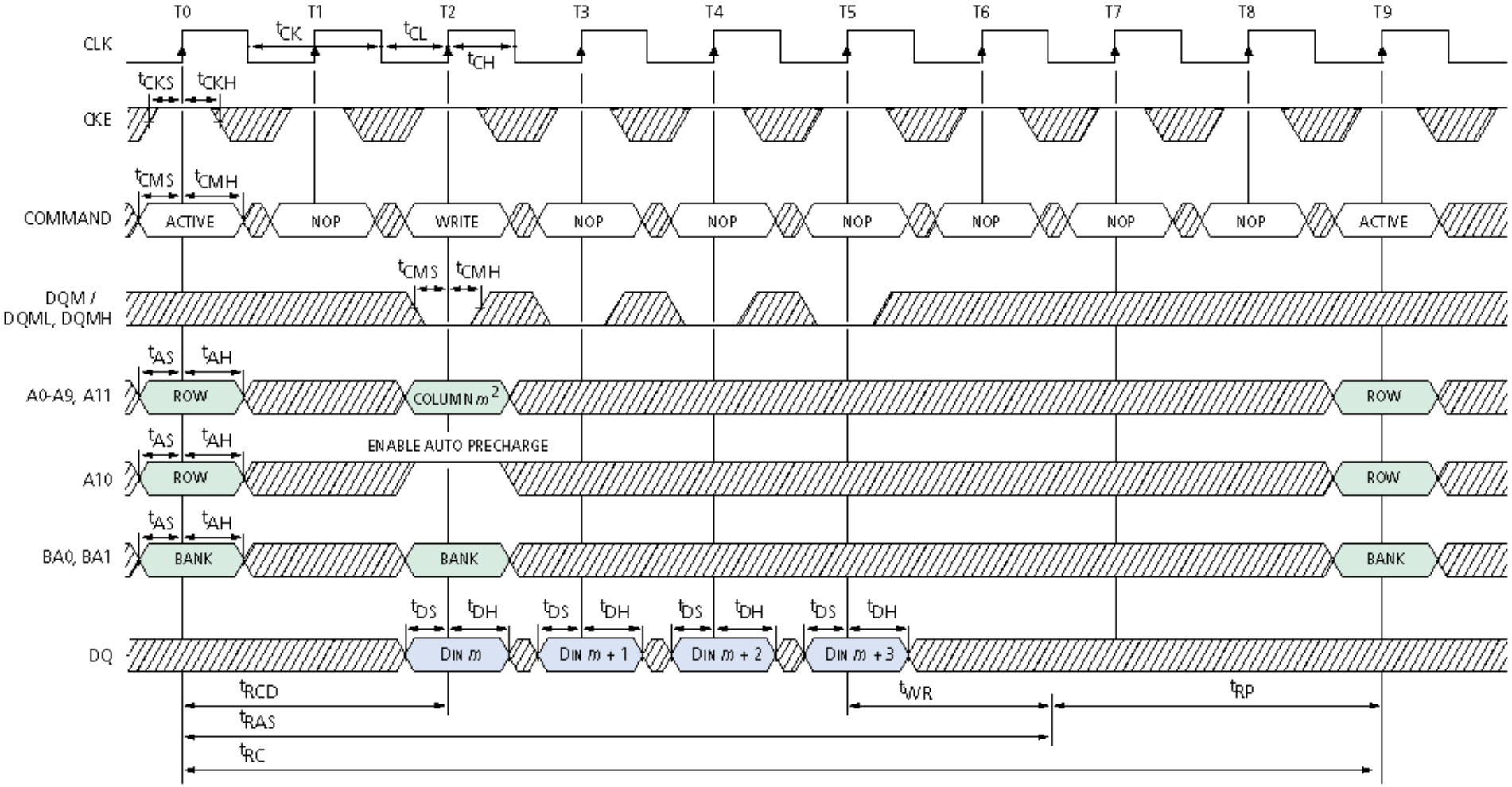
4.3. SDRAM: MT48LC8M32B2 Details

- Multiple "**banks**" of cell arrays are used to **reduce access time**:
  - Each bank is 4K rows by 512 "columns" by 16 bits (for our part)
- Read and Write operations as split into RAS (row access) followed by CAS (column access).
- These operations are controlled by **sending commands**.
  - Commands are sent using the RAS, CAS, CS, & WE pins.
- Address pins are "time multiplexed"
  - During **RAS** operation, address lines select the **bank** and **row**
  - During **CAS** operation, address lines select the **column**.
- "ACTIVE" command "opens" a row for operation.
  - transfers the contents of the entire to a row buffer
- Subsequent "READ" or "WRITE" commands modify the contents of the row buffer.
- For burst reads and writes during "READ" or "WRITE" the starting address of the block is supplied.
  - **Burst length** is programmable **as 1, 2, 4, 8** or a "**full page**" (entire row) with a burst terminate option.
- Special commands are used for initialization (burst options, etc.)
- A burst operation takes ≈ 4 + n cycles (for n words)

4.3. SDRAM Read Burst (Auto-Precharge)
(CAS Latency=2; Burst Length=4)



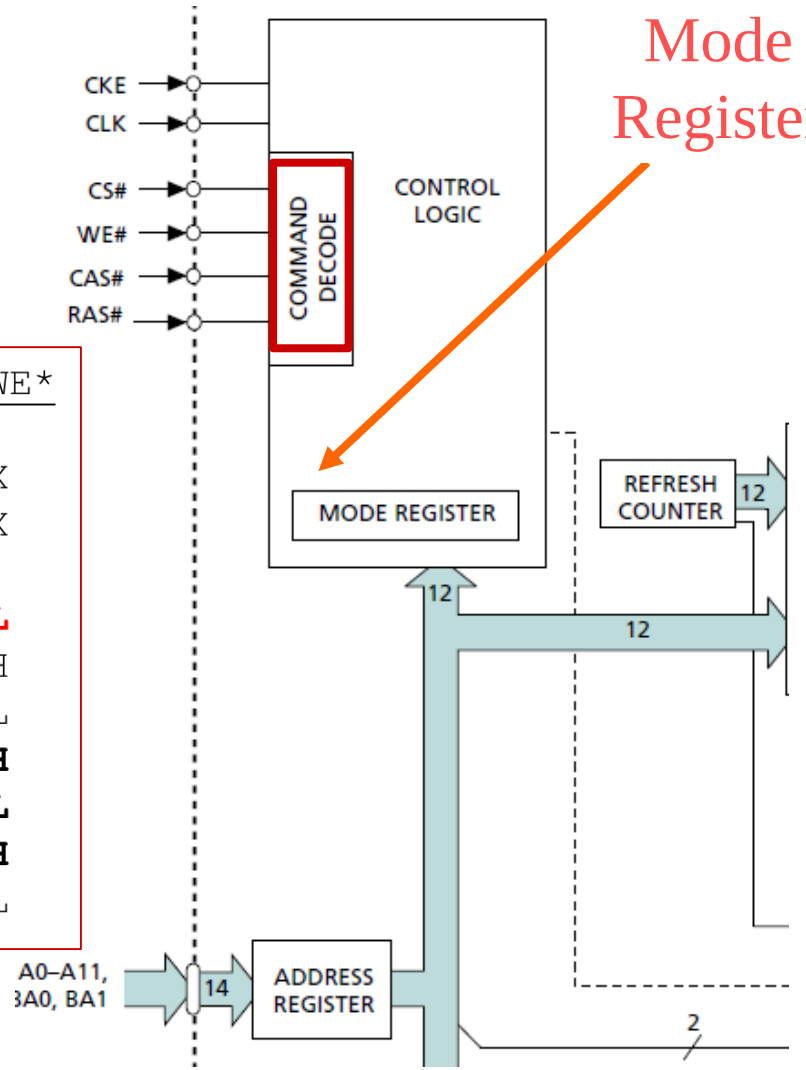
4.3. SDRAM Write Burst (Auto Precharge)



4.3. SDRAM: Commands (64Mbit, Micron)

Control Signals Sampled Synchronously

Function	CS*	RAS*	CAS*	WE*
COMMAND INHIBIT	H	X	X	X
NOP	H	X	X	X
LOAD MODE REGISTER	**L**	**L**	**L**	**L**
AUTO/SELF REFRESH	L	L	L	H
PRECHARGE	L	L	H	L
ACTIVE (SEL BANK/ROW)	**L**	**L**	**H**	**H**
WRITE	**L**	**H**	**L**	**L**
READ	**L**	**H**	**L**	**H**
BURST TERMINATE	L	H	H	L



4.3. SDRAM: Load Mode Register Command (I)

Address used as Operation Code

- A2:A0 → Burst Length
- A3 → Burst Type {Sequential, Interleaved}
- A6:A4 → CAS Latency
- A8:A7 → Operation Mode
- A9 → Write Burst Mode
- A11:A10 → Reserved

Burst Length

			BURST LENGTH	
A2	A1	A0	M3=0	M3=1
0	0	0	1	1
0	0	1	2	2
0	1	0	4	4
0	1	1	8	8
1	0	0	RESERVED	RESERVED
1	0	1	RESERVED	RESERVED
1	1	0	RESERVED	RESERVED
1	1	1	FULL PAGE	RESERVED

4.3. SDRAM: Load Mode Register Command (II)

CAS Latency

A6	A5	A4	CAS LATENCY
0	0	0	RESERVED
0	0	1	RESERVED
0	1	0	2
0	1	1	3
1	0	0	RESERVED
1	0	1	RESERVED
1	1	0	RESERVED
1	1	1	RESERVED

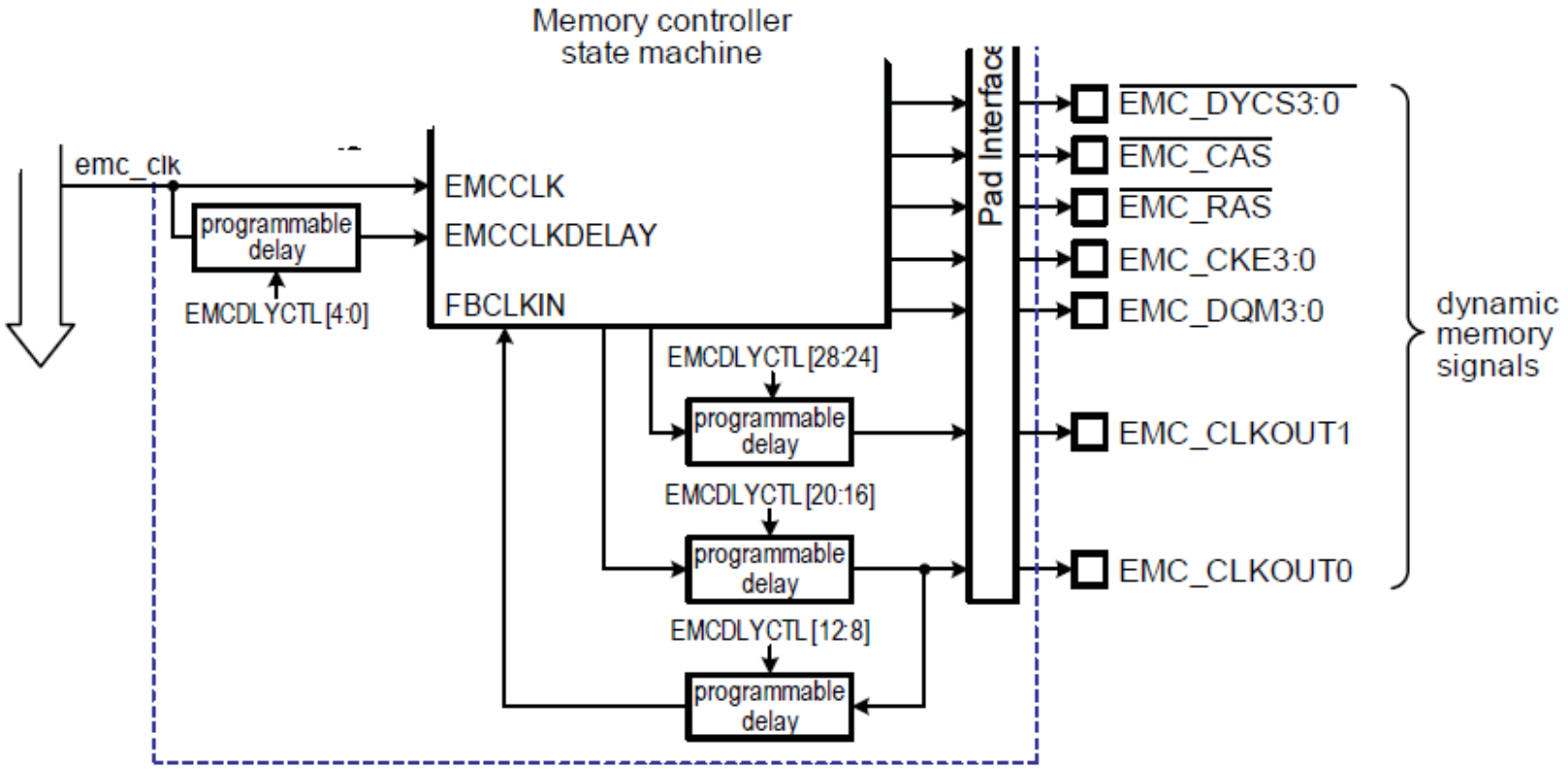
Op Mode

A8	A7	Operation Mode
0	0	Standard Operation
0	1	RESERVED
1	0	RESERVED
1	1	RESERVED

4.3. LPC178x: Using the EMC with SDRAM

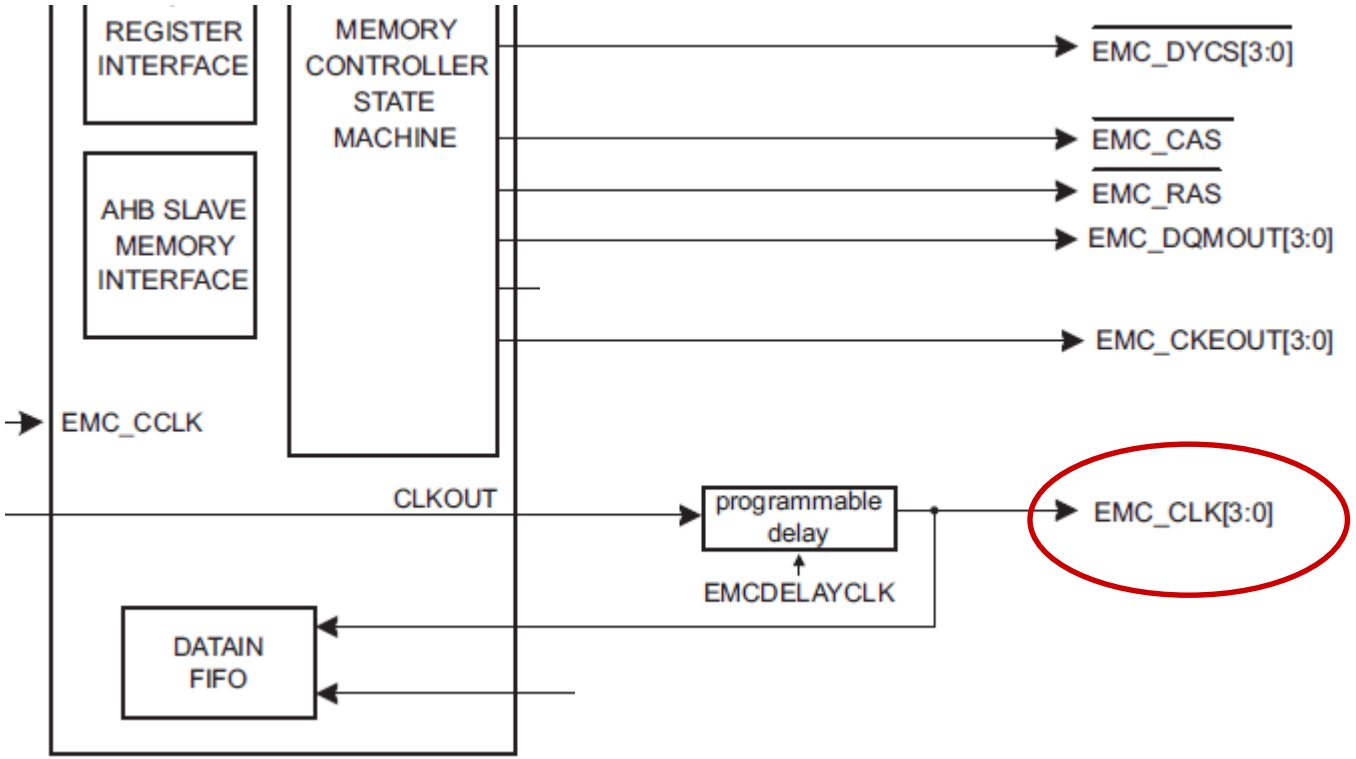
- **Dynamic chip selects** each support up to **256 MB** of data.
- Dynamic memory interface support including Single Data Rate SDRAM.
- Low transaction latency.
- **16-bit** and **32-bit wide** chip select SDRAM memory support.
- **Four chip selects** for synchronous memory devices.
- **Power-saving modes** dynamically control CKE and CLKOUT to SDRAMs.
- Dynamic memory **self-refresh** mode controlled by software.
- Controller supports **2 kbit**, **4 kbit**, and **8 kbit row address** synchronous memory parts.
  - That is typical 512 Mbit, 256 Mbit, and 128 Mbit parts, with 4, 8, 16, or 32 data bits per device.
- Separate reset domains allow the for auto-refresh through a chip reset if desired.
- **Programmable delay** elements allow fine-tuning **EMC timing**.

4.3. LPC178x: SDRAM signals of EMC



Chip select pin	Address range	Memory type	Size of range
EMC_$\overline{\text{DYCS0}}$	0xA000 0000 - 0xAFFF FFFF	Dynamic	256 MB
EMC_$\overline{\text{DYCS1}}$	0xB000 0000 - 0xBFFF FFFF	Dynamic	256 MB
EMC_$\overline{\text{DYCS2}}$	0xC000 0000 - 0xCFFF FFFF	Dynamic	256 MB
EMC_$\overline{\text{DYCS3}}$	0xD000 0000 - 0xDFFF FFFF	Dynamic	256 MB

4.3. LPC18xx: SDRAM signals of EMC



Chip select pin	Address range	Memory type	Size of range
$\overline{\text{EMC\_DYCS0}}$	0x2800 0000 - 0x2FFF FFFF	Dynamic	128 MB
$\overline{\text{EMC\_DYCS1}}$	0x3000 0000 - 0x3FFF FFFF	Dynamic	256 MB
$\overline{\text{EMC\_DYCS2}}$	0x6000 0000 - 0x6FFF FFFF	Dynamic	256 MB
$\overline{\text{EMC\_DYCS3}}$	0x7000 0000 - 0x7FFF FFFF	Dynamic	256 MB

4.3. EMC signals in LPC178x

- For the clock delayed operating mode, separate programmable delays are provided for each potential clock output, CLKOUT0 and CLKOUT1.
  - For the command delayed operating mode, a programmable delay is provided to control delay of all command outputs.
  - For both operating modes, a programmable delay is provided to control the time at which input data from SDRAM memory is sampled.
- For **32 bit wide** chip selects data is transferred to and from dynamic memory in SDRAM **bursts of four**.
- For **16 bit wide** chip selects SDRAM **bursts of eight** are used.

4.3. EMC: SDRAM Registers (I)

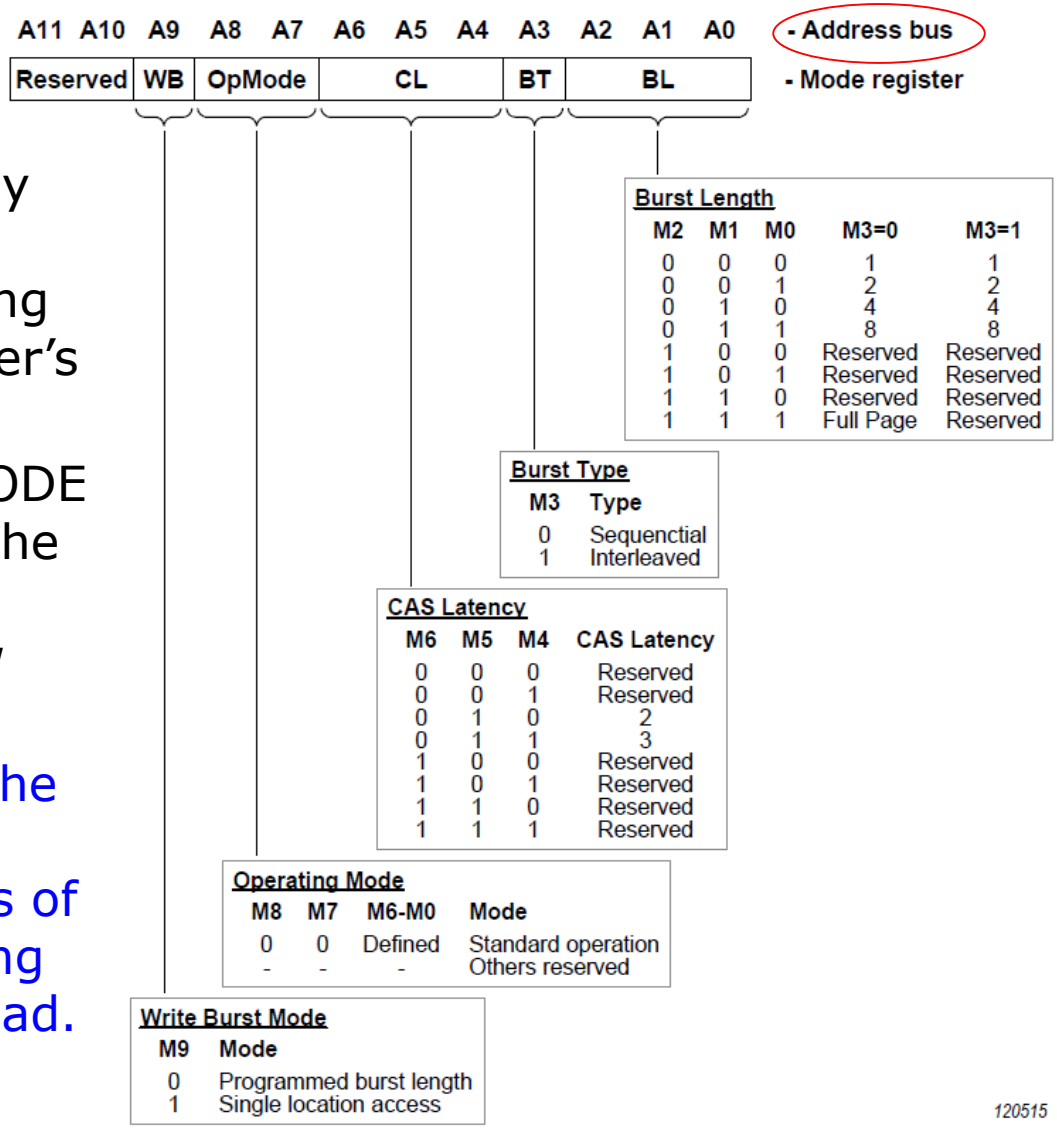
Register Name	Access	Address offset	Description	Warm Reset Value[1]	POR Reset Value[1]	Table
CONTROL	R/W	0x000	Controls operation of the memory controller.	0x1	0x3	114
STATUS	RO	0x004	Provides EMC status information.	-	0x5	115
CONFIG	R/W	0x008	Configures operation of the memory controller	-	0x0	116
DYNAMICCONTROL	R/W	0x020	Controls dynamic memory operation.	-	0x006	117
DYNAMICREFRESH	R/W	0x024	Configures dynamic memory refresh.	-	0x0	118
DYNAMICREADCONFIG	R/W	0x028	Configures dynamic memory read strategy.	-	0x0	119
DYNAMICRP	R/W	0x030	Precharge command period.	-	0x0F	120
DYNAMICRAS	R/W	0x034	Active to precharge command period.	-	0xF	121
DYNAMICSREX	R/W	0x038	Self-refresh exit time.	-	0xF	122
DYNAMICAPR	R/W	0x03C	Last-data-out to active command time.	-	0xF	123
DYNAMICDAL	R/W	0x040	Data-in to active command time.	-	0xF	124
DYNAMICWR	R/W	0x044	Write recovery time.	-	0xF	125
DYNAMICRC	R/W	0x048	Selects the active to active command period.	-	0x1F	126
DYNAMICRFC	R/W	0x04C	Selects the auto-refresh period.	-	0x1F	127
DYNAMICXSR	R/W	0x050	Time for exit self-refresh to active command.	-	0x1F	128
DYNAMICRRD	R/W	0x054	Latency for active bank A to active bank B.	-	0xF	129
DYNAMICMRD	R/W	0x058	Time for load mode register to active command.	-	0xF	130

4.3. EMC: SDRAM Registers (II)

Register Name	Access	Address offset	Description	Warm Reset Value[1]	POR Reset Value[1]	Table
STATICEXTENDEDWAIT	R/W	0080	Time for long static memory read and write transfers.	-	0x0	131
DYNAMICCONFIG0	R/W	0x100	Configuration information for EMC_DYCS0.	-	0x0	132
DYNAMICRASCAS0	R/W	0x104	RAS and CAS latencies for EMC_DYCS0.	-	0x303	134
DYNAMICCONFIG1	R/W	0x120	Configuration information for EMC_DYCS1.	-	0x0	132
DYNAMICRASCAS1	R/W	0x124	RAS and CAS latencies for EMC_DYCS1.	-	0x303	134
DYNAMICCONFIG2	R/W	0x140	Configuration information for EMC_DYCS2.	-	0x0	132
DYNAMICRASCAS2	R/W	0x144	RAS and CAS latencies for EMC_DYCS2.	-	0x303	134
DYNAMICCONFIG3	R/W	0x160	Configuration information for EMC_DYCS3.	-	0x0	132
DYNAMICRASCAS3	R/W	0x164	RAS and CAS latencies for EMC_DYCS3.	-	0x303	134
STATICCONFIG0	R/W	0x200	Configuration for EMC_CS0.	-	0x0	135
STATICWAITWEN0	R/W	0x204	Delay from EMC_CS0 to write enable.	-	0x0	136
STATICWAITOEN0	R/W	0x208	Delay from EMC_CS0 or address change, whichever is later, to output enable.	-	0x0	137
STATICWAITRD0	R/W	0x20C	Delay from EMC_CS0 to a read access.	-	0x1F	138

4.3. EMC: SDRAM Mode register

- The mode register is loaded by first sending the "Set Mode" **Command** to the SDRAM using the DYNAMICCONTROL register's SDRAM.
- Initialization bits to send a MODE command, and then reading the SDRAM at an address that is partially formed from the new mode register value.
- The actual value loaded into the mode register is taken by the SDRAM from the address lines of the EMC while they are sending the row address during the read.



4.3. EMC: SDRAM Mode register (example)

- **A single 8M by 16-bit external SDRAM chip in Row, Bank, Column mode on CS0.**
- **CAS latency of 2.**

- **Information needed:**
  - **Base address for Dynamic Chip Select 0**, found in Table 3. For this device, the address is 0xA000 0000.
  - **Mode register value**, based on information from both the SDRAM data sheet, as in Figure 16, and the EMC.

    Since the EMC uses bursts of 8 for a 16-bit external memory, we need to load the mode register with a burst length of 8 (8 x 16 bits memory width = 128 bits). In this example, the value will be 0x23.
  - **Bank bits** and **column bits**, look up in Table 133. In this example, it is 4 banks and 9 column bits.
  - **Bus width**, defined in this example to be 16 bits.

4.3. EMC: SDRAM Mode register (example sol.)

- **Procedure:**
  - Determine the **shift value OFFSET** to shift the mode register content by. This shift value depends on the SDRAM device organization and it is calculated as:
    - OFFSET = number of columns + total bus width + bank select bits (RBC mode).
    - OFFSET = number of columns + total bus width (BRC mode).
  - Select the SDRAM memory mapped address **DYCSX**.
  - The SDRAM read address is:

    ADDRESS = DYCSX + (MODE << OFFSET).
- **The Mode register value calculation is:**
  - Base address + (mode register value << (bank bits + column bits + bus width/16).
  - The shift operation aligns the mode register value with the row address bits.
  - **In this example:** 0xA000 0000 + (0x23 << (2 + 9 + 1)) = 0xA000 0000 + 0x23000 = 0xA002 3000

4.3. EMC: DM Configurations registers

Bit	Symbol	Value	Description	Reset Value
2:0	-		Reserved. Read value is undefined, only zero should be written.	NA
4:3	MD		Memory device.	0
		0x0	SDRAM (POR reset value).	
		0x1	Low-power SDRAM.	
		0x2	Reserved.	
		0x3	Reserved.	
6:5	-		Reserved. Read value is undefined, only zero should be written.	NA
12:7	AM0		See Table 133. 000000 = reset value.[1]	0
13	-		Reserved. Read value is undefined, only zero should be written.	NA
14	AM1		See Table 133. 0 = reset value.	0
18:15	-		Reserved. Read value is undefined, only zero should be written.	NA
19	B		Buffer enable.	0
		0	Buffer disabled for accesses to this chip select (POR reset value).	
		1	Buffer enabled for accesses to this chip select.[2]	
20	P		Write protect.	0
		0	Writes not protected (POR reset value).	
		1	Writes protected.	
31:21	-		Reserved. Read value is undefined, only zero should be written.	NA

[1] The SDRAM column and row width and number of banks are computed automatically from the address mapping.

[2] The buffers must be disabled during SDRAM initialization. The buffers must be enabled during normal operation.

4.3. EMC: DM Address Mapping

- A chip select can be connected to a single memory device, in this case the chip select data bus width is the same as the device width.
- Alternatively the chip select can be connected to a number of external devices. In this case the chip select data bus width is the sum of the memory device data bus widths.
- For example, for a chip select connected to:
  - a 32 bit wide memory device, choose a 32 bit wide address mapping.
  - a 16 bit wide memory device, choose a 16 bit wide address mapping.
  - four x 8 bit wide memory devices, choose a 32 bit wide address mapping.
  - two x 8 bit wide memory devices, choose a 16 bit wide address mapping.

- The **SDRAM bank** select pins **BA1** and **BA0** are connected to address lines A14 and A13, respectively..

4.3. EMC: DM RAS/CAS Delay latency registers

Bit	Symbol	Value	Description	Reset Value
1:0	RAS		RAS latency (active to read/write delay).	11
		0x0	Reserved.	
		0x1	One EMCCLK cycle.	
		0x2	Two EMCCLK cycles.	
		0x3	Three EMCCLK cycles (POR reset value).	
7:2	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
9:8	CAS		CAS latency.	11
		0x0	Reserved.	
		0x1	One EMCCLK cycle.	
		0x2	Two EMCCLK cycles.	
		0x3	Three EMCCLK cycles (POR reset value).	
31:10	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

4.3. EMC: DM Refresh Timer register

Table 118. Dynamic Memory Refresh Timer register (DYNAMICREFRESH - address 0x2009 C024) bit description

Bit	Symbol	Description	Reset value
10:0	REFRESH	Refresh timer. Indicates the multiple of 16 EMCCLKs between SDRAM refresh cycles. 0x0 = Refresh disabled (POR reset value). 0x1 - 0x7FF = n x16 = 16n EMCCLKs between SDRAM refresh cycles. For example: 0x1 = 1 x 16 = 16 EMCCLKs between SDRAM refresh cycles. 0x8 = 8 x 16 = 128 EMCCLKs between SDRAM refresh cycles	0
31:11	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

For example, for the refresh period of 16 µs, and a EMCCLK frequency of 50 MHz, the following value must be programmed into this register:

(16 x 10-6 x 50 x 106) / 16 = 50 or 0x32

4.3. SDRAM functions support for Micron MT48LC8M32LFB5 (Keil)

sdram_mt48lc8m32lfb5.c

```
/*********************************************************************
 * @brief                   Initialize external SDRAM memory Micron MT48LC8M32LFB5
 *                          256Mbit (8M x 32)
 * @param[in]               None
 * @return                  None
 *********************************************************************/
void SDRAMInit( void )
{
        uint32_t i, dwtemp;
        TIM_TIMERCFG_Type TIM_ConfigStruct;
        /* Initialize EMC */
        EMC_Init();
        TIM_ConfigStruct.PrescaleOption = TIM_PRESCALE_USVAL;
        TIM_ConfigStruct.PrescaleValue          = 1;
        // Set configuration for Tim_config and Tim_MatchConfig
        TIM_Init(LPC_TIM0, TIM_TIMER_MODE,&TIM_ConfigStruct);
        //Configure memory layout, but MUST DISABLE BUFFERs during configuration
        LPC_EMC->DynamicConfig0    = 0x00004480; /* 256MB, 8Mx32, 4 banks, row=12, column=9 */
        /*Configure timing for  Micron SDRAM MT48LC8M32LFB5-8 */
        //Timing for 48MHz Bus
        LPC_EMC->DynamicRasCas0    =        0x00000201; /* 1 RAS, 2 CAS latency */
        LPC_EMC->DynamicReadConfig =        0x00000001; /* Command delayed strategy, using EMCCLKDELAY */
        LPC_EMC->DynamicRP         =        0x00000000; /* ( n + 1 ) -> 1 clock cycles */
        LPC_EMC->DynamicRAS        =        0x00000002; /* ( n + 1 ) -> 3 clock cycles */
        LPC_EMC->DynamicSREX       =        0x00000003; /* ( n + 1 ) -> 4 clock cycles */
        LPC_EMC->DynamicAPR        =        0x00000001; /* ( n + 1 ) -> 2 clock cycles */
        LPC_EMC->DynamicDAL        =        0x00000002; /* ( n ) -> 2 clock cycles */
        LPC_EMC->DynamicWR         =        0x00000001; /* ( n + 1 ) -> 2 clock cycles */
        LPC_EMC->DynamicRC         =        0x00000003; /* ( n + 1 ) -> 4 clock cycles */
        LPC_EMC->DynamicRFC        =        0x00000003; /* ( n + 1 ) -> 4 clock cycles */
        LPC_EMC->DynamicXSR        =        0x00000003; /* ( n + 1 ) -> 4 clock cycles */
        LPC_EMC->DynamicRRD        =        0x00000000; /* ( n + 1 ) -> 1 clock cycles */
        LPC_EMC->DynamicMRD        =        0x00000000; /* ( n + 1 ) -> 1 clock cycles */
```

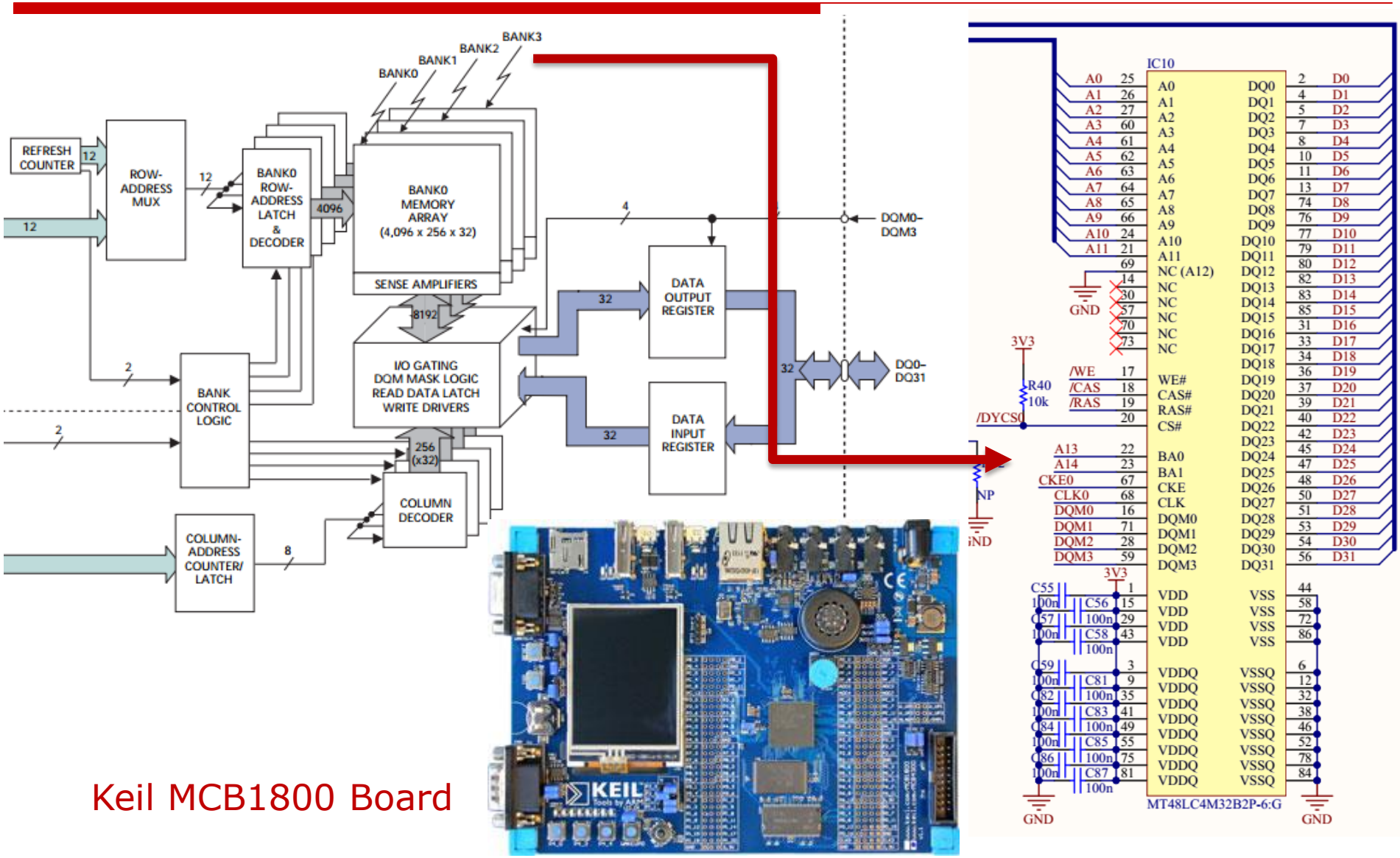
4.3. SDRAM functions support for Micron MT48LC8M32LFB5 (Keil)

```
    TIM_Waitms(100);                          /* wait 100ms */
    LPC_EMC->DynamicControl     = 0x00000183; /* Issue NOP command */
    TIM_Waitms(200);                          /* wait 200ms */
    LPC_EMC->DynamicControl     = 0x00000103; /* Issue PALL command */
    LPC_EMC->DynamicRefresh     = 0x00000002; /* ( n * 16 ) -> 32 clock cycles */

    for(i = 0; i < 0x80; i++);          /* wait 128 AHB clock cycles */

    //Timing for 48MHz Bus
    LPC_EMC->DynamicRefresh     = 0x0000002E; /* ( n * 16 ) -> 736 clock cycles -> 15.330uS
                                                 at 48MHz <= 15.625uS ( 64ms / 4096 row ) */

    LPC_EMC->DynamicControl     = 0x00000083; /* Issue MODE command */

    //Timing for 48/60/72MHZ Bus
    dwtemp = *((volatile uint32_t *)(SDRAM_BASE_ADDR | (0x22<<(2+2+9)))); /* 4 burst, 2 CAS latency */
    LPC_EMC->DynamicControl     = 0x00000000; /* Issue NORMAL command */

    //[re]enable buffers
    LPC_EMC->DynamicConfig0     = 0x00084480; /* 256MB, 8Mx32, 4 banks, row=12, column=9 */
}
```

sdram_mt48lc8m32lfb5.h

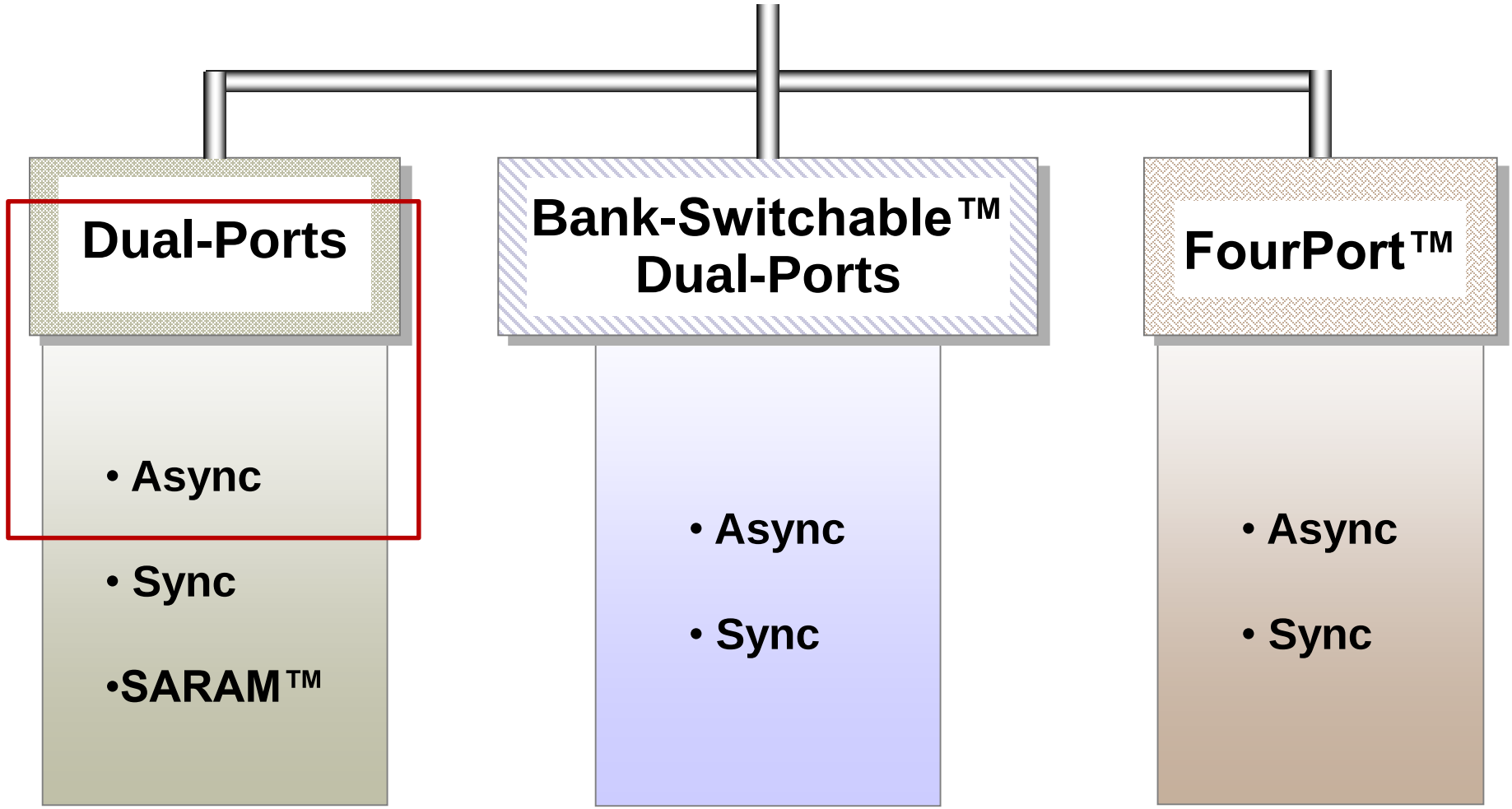
```
/* Peripheral group ----------------------------------------------------------- */
/** @defgroup  Sdram_MT48LC8M32FLB5 Sdram MT48LC8M32FLB5

#ifndef __SDRAM_MT48LC8M32LFB5_H
#define __SDRAM_MT48LC8M32LFB5_H
#define SDRAM_BASE_ADDR      0xA0000000
#define SDRAM_SIZE           0x10000000
#if (_CURR_USING_BRD != _IAR_OLIMEX_BOARD)
extern void SDRAMInit( void );
#endif
#endif //__SDRAM_MT48LC8M32LFB5_H
```

4.3. EMC: SDRAM connection example

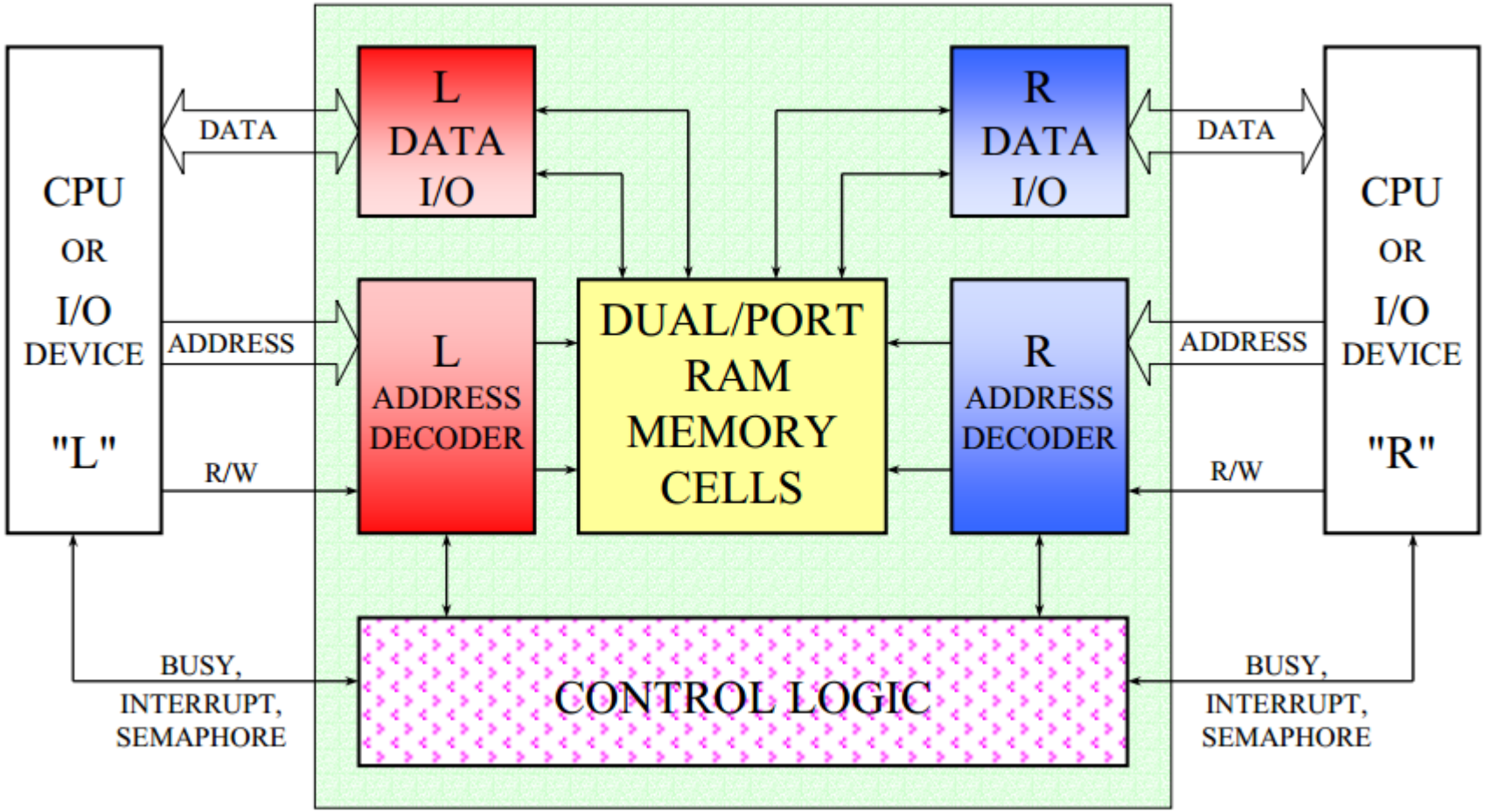


Keil MCB1800 Board

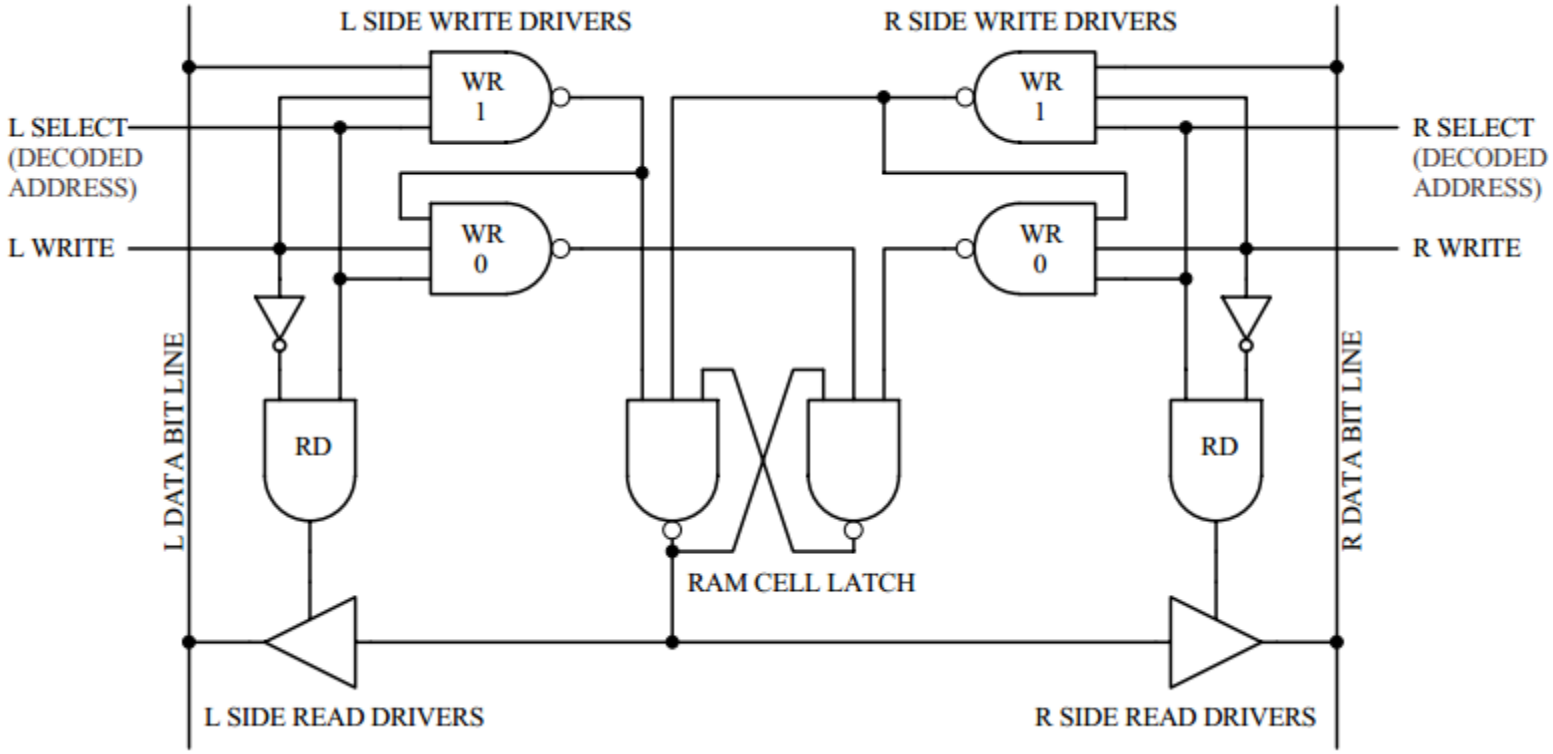
4.4. Multi-Ports Memories



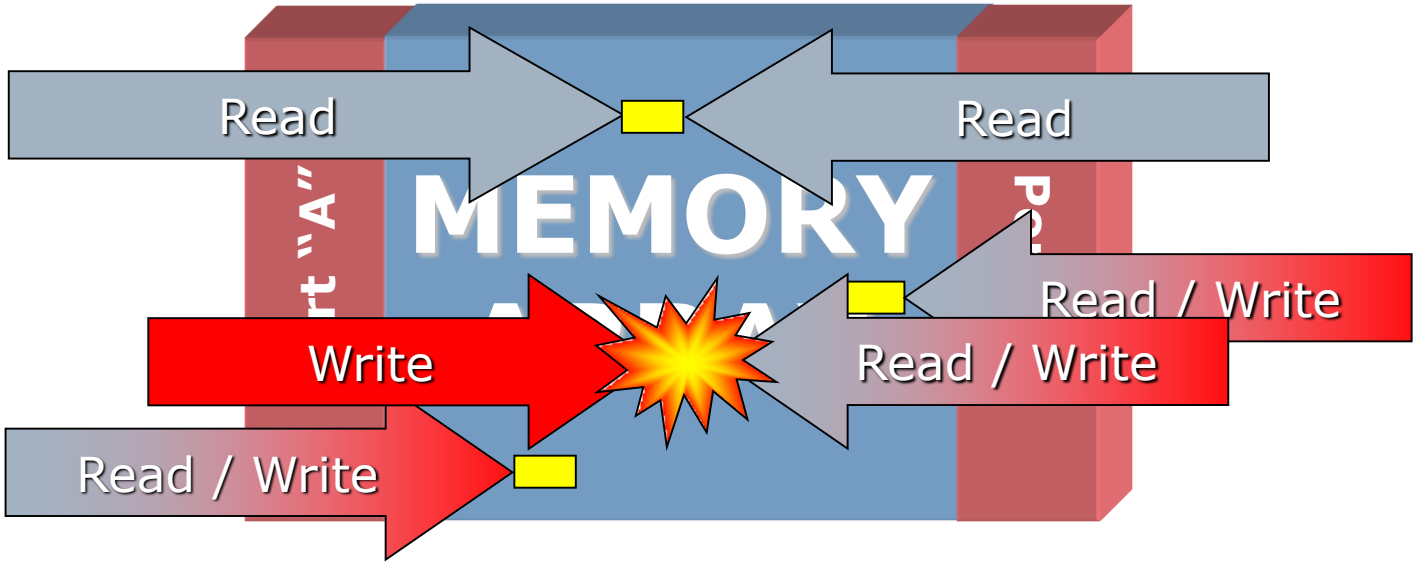
4.4. Dual-Port: Block Diagram



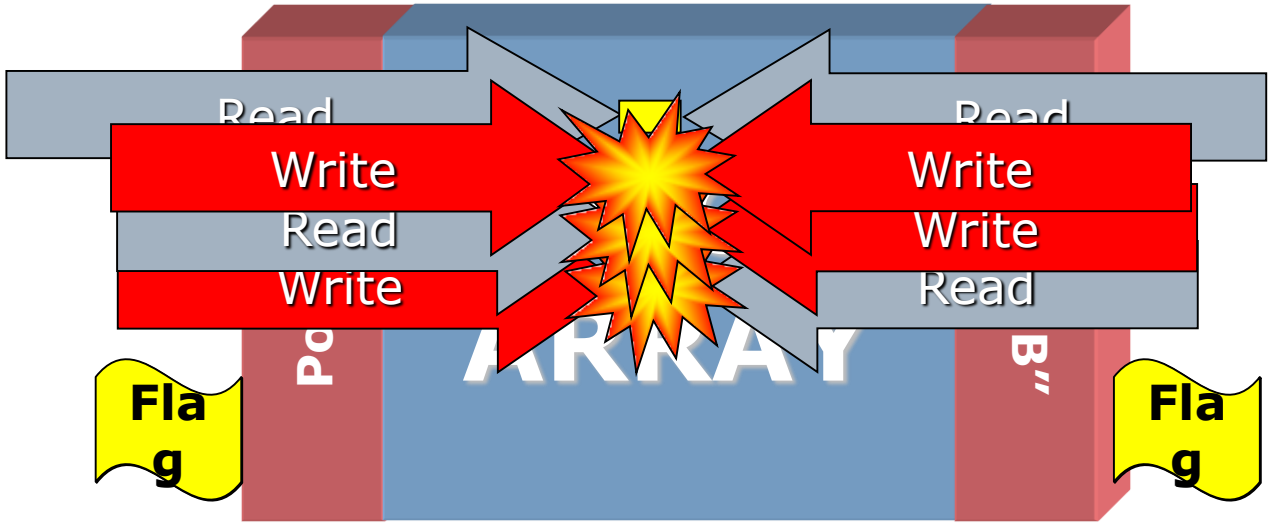
4.4. Dual-Port: RAM cell



4.4. Dual-Port: Access Ports



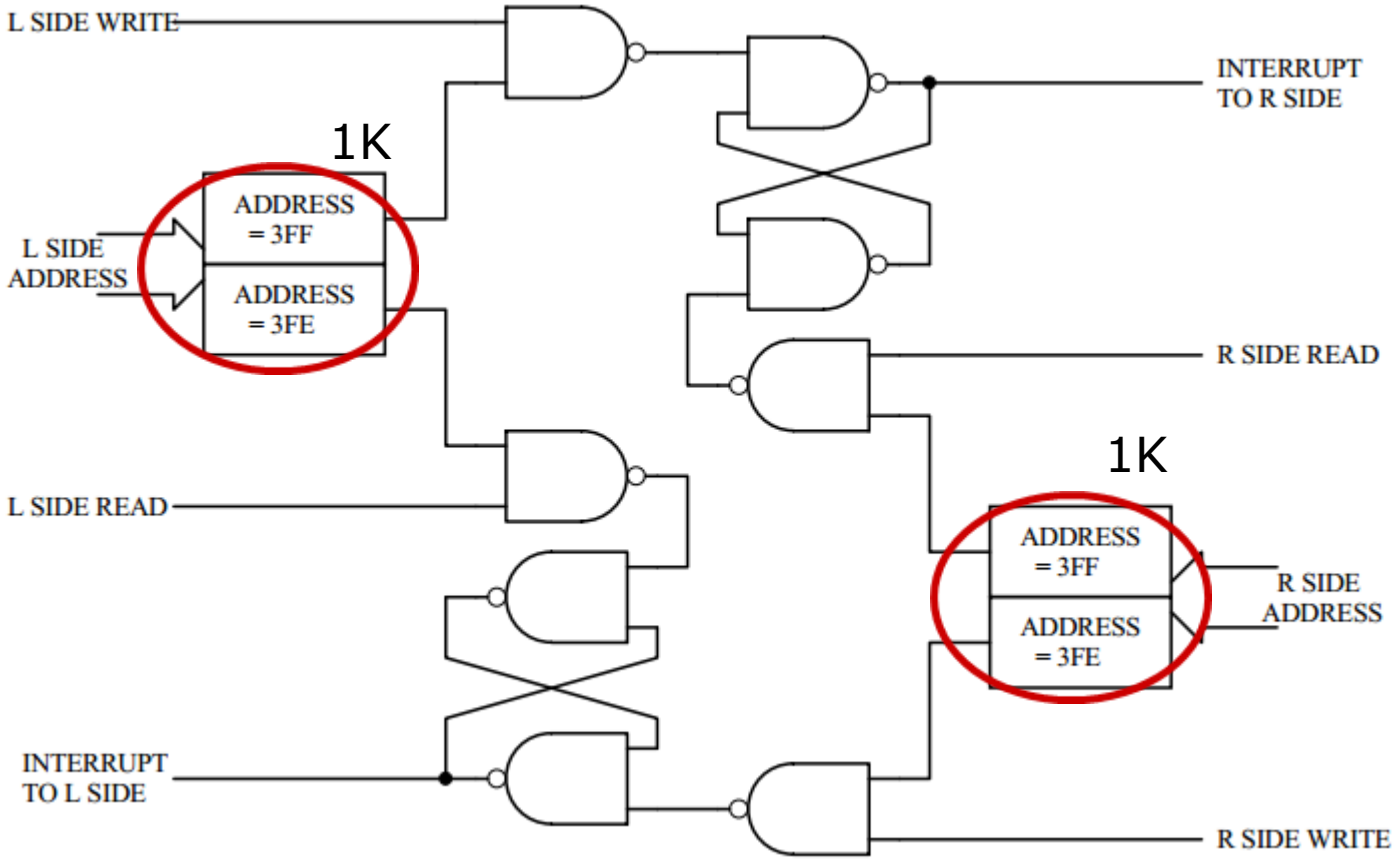
4.4. Dual-Port: Collision Detection



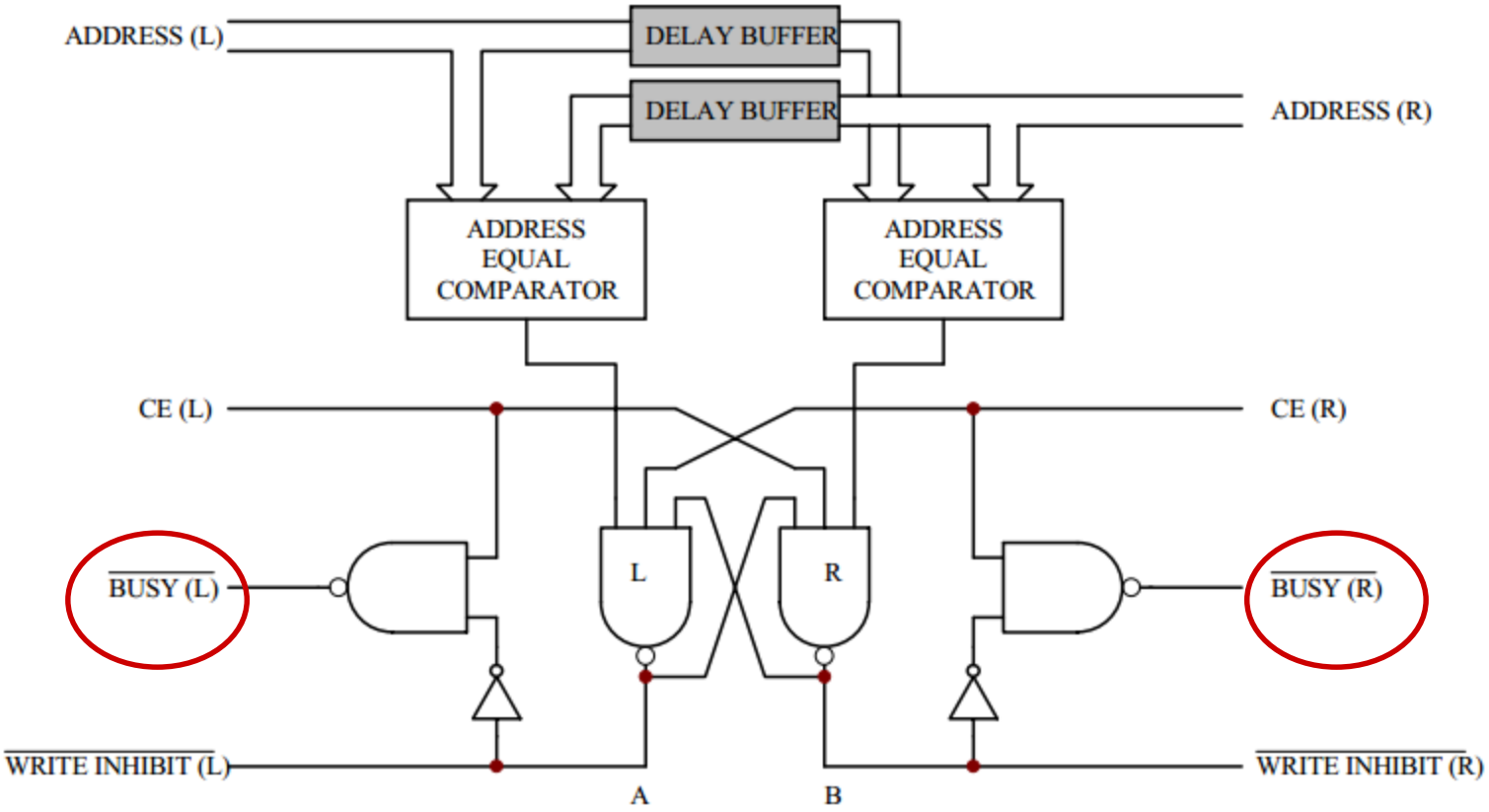
- **Arbitration mechanisms:**
  - Interrupts (software + hardware interrupt)
  - Hardware signal (/BUSY) → Only in CPUs with Asynchronous Buses!!
  - Semaphores (software)

4.4. DP: Interrupt arbitration

- **Use two last positions memory:**
  - L CPU write in last (**Odd**) and interrupt to **R** CPU.
  - **R** CPU write in penultimate (**Even**) and interrupt to **L** CPU.

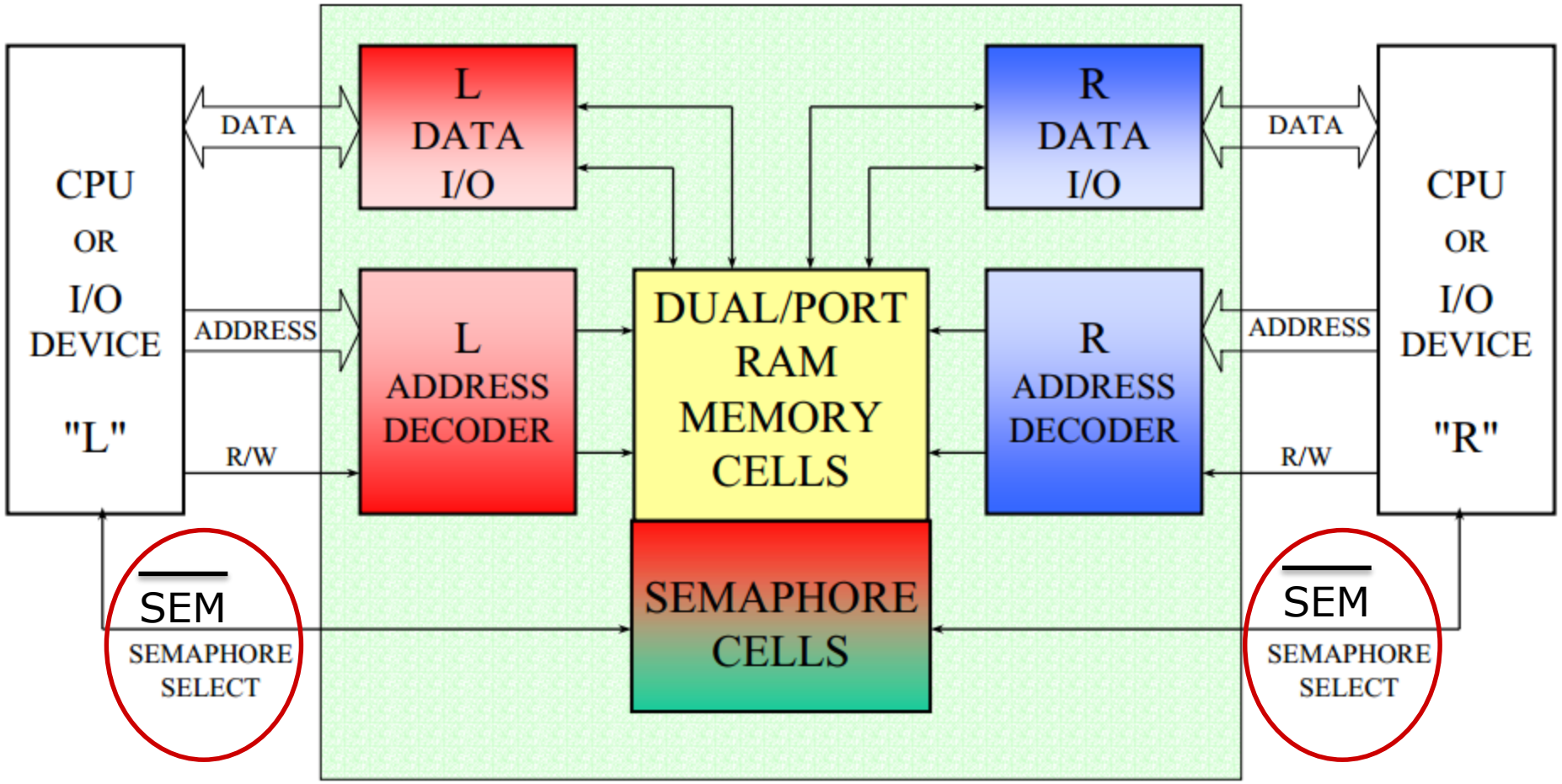


4.4. DP: Busy Signal arbitration

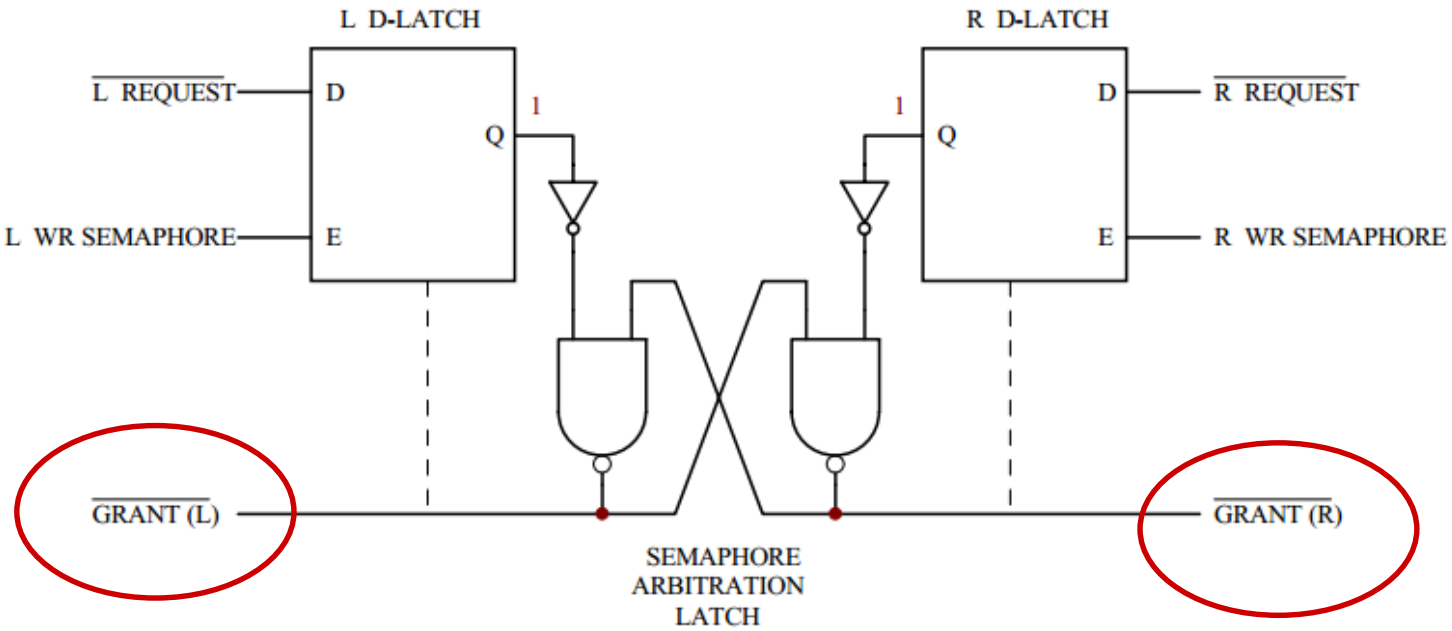


Insert automatic wait cycles in read/write access (ex. MC68000)

4.4. DP: Semaphore Arbitration



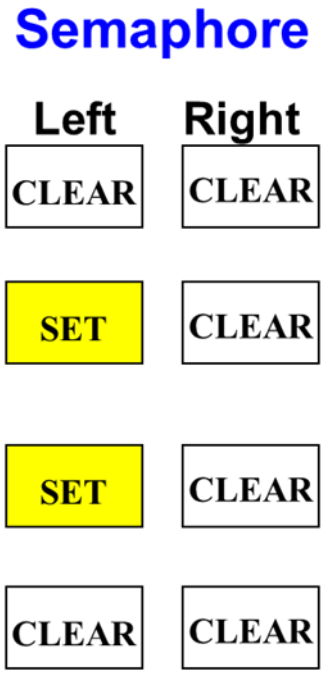
4.4. DP: Semaphore Arbitration (cell)



- **One Semaphore contains two latches, one for each port. Initially Clear.**
- **Left side requests then right side requests.**
- **Left side reads request granted, right side reads request denied.**
- **Left side clears, right side request may now be accepted.**

4.4. DP: Semaphore Arbitration

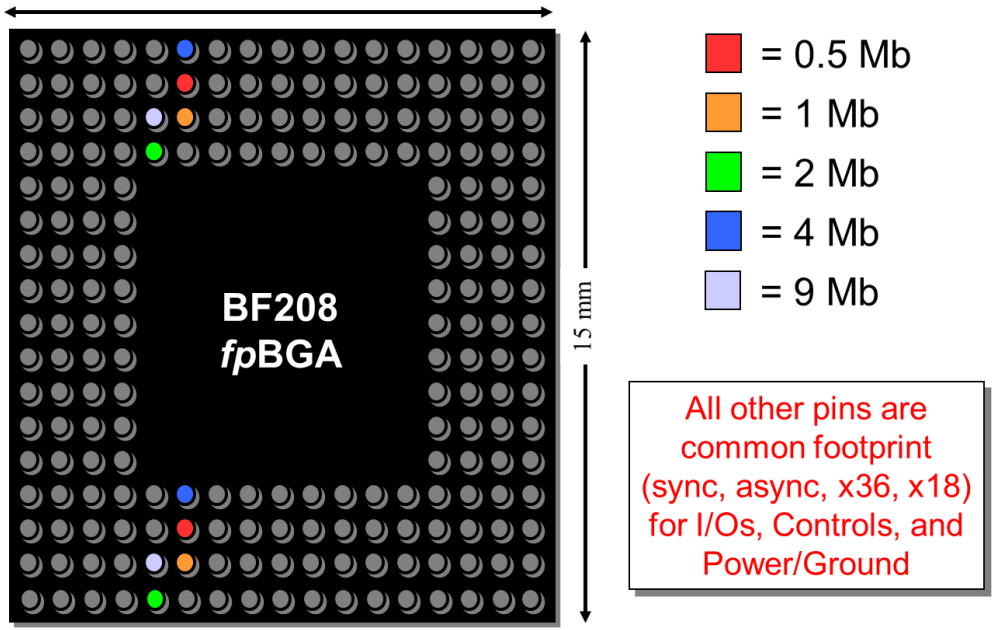
- Each semaphore consists of two latches one for each port.
- Initially both latches are clear.
- The left CPU sends a request for the semaphore. This causes the left latch to set and the right latch to be held clear.
  - At this time, even if the right CPU sends a request the right latch will remain clear.
  - The left CPU now reads the latch and sees the set condition, which means it has been granted permission to use the memory block.
  - During this time the right CPU can continue to send requests but when it reads its latch it will still see a clear condition meaning it does not have permission to use the memory block.
- When the left CPU is done it clears the semaphore allowing the next request by the right CPU to succeed.



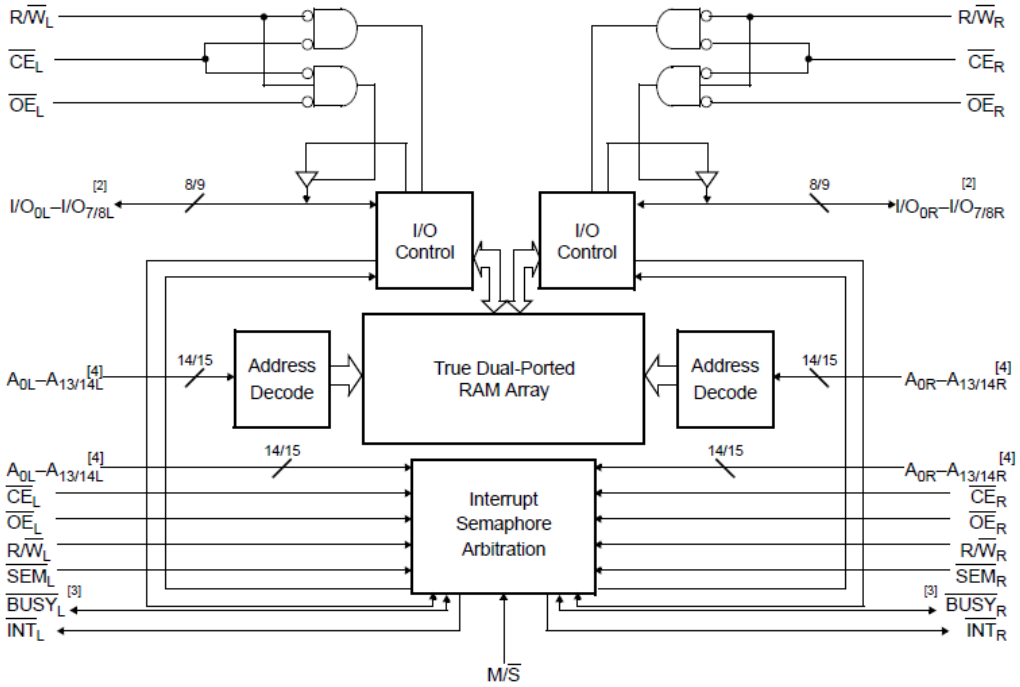
4.4. DP: IDT manufacturer example

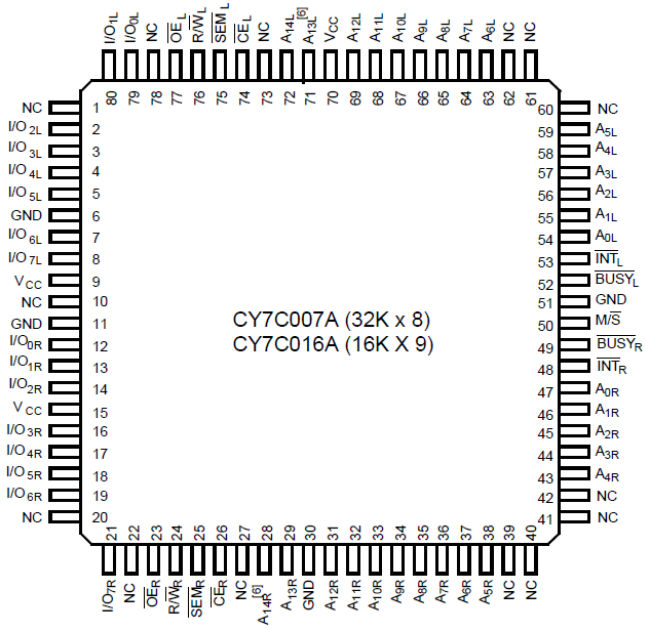
- Fastest Speeds
  - Sync at 200 MHz
  - Async at 10 ns tAA
- Multiple Depth/Width Combinations
  - Density from 0.5Mb up to 9Mb
  - X36, x18, x9 (@2Mb) configurations

- Common package for x36, x18 in BGA
- JTAG
- Selectable 3.3V / 2.5V I/Os



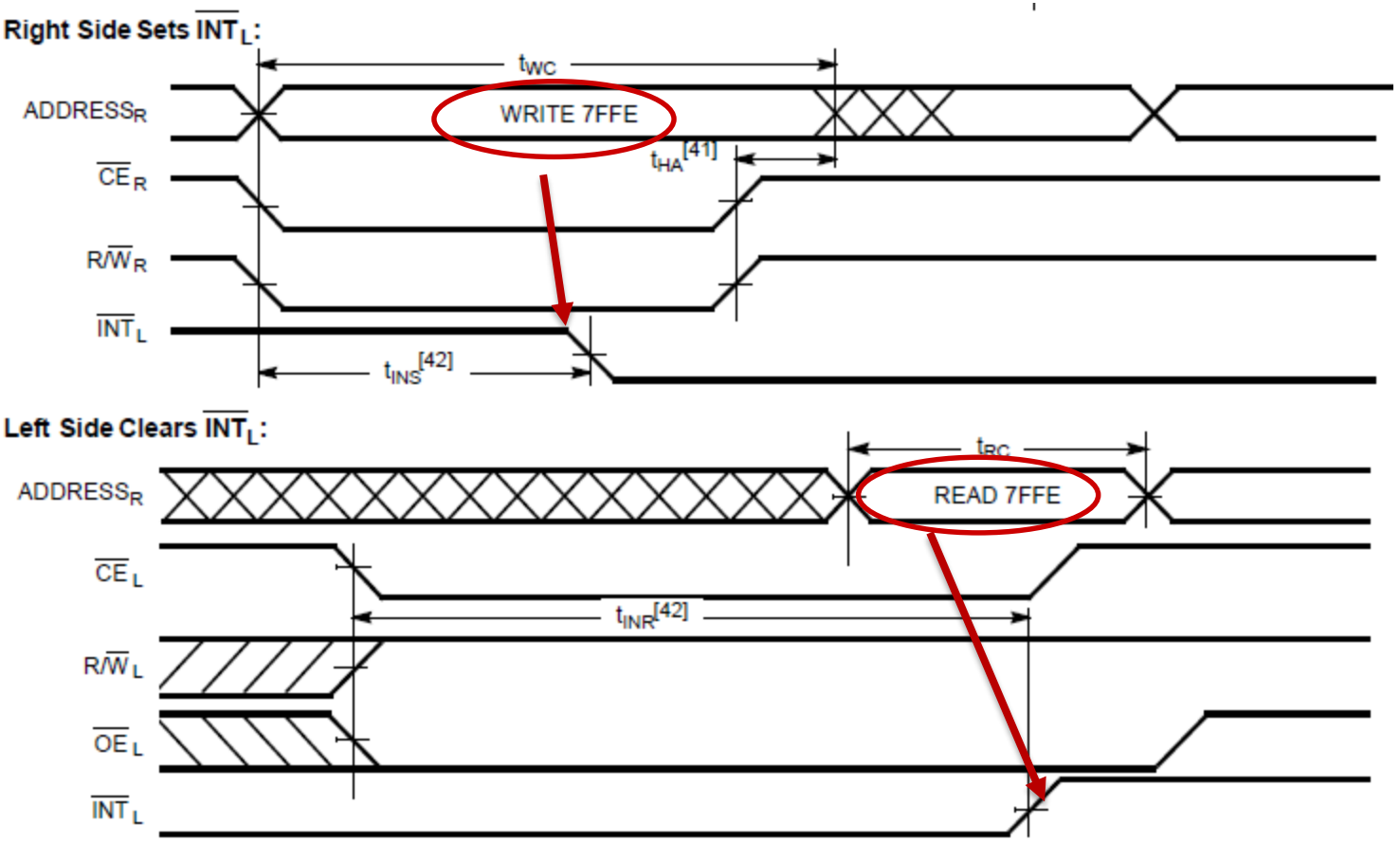
4.4. CY7C007/016A (32K/16K x8, x9)





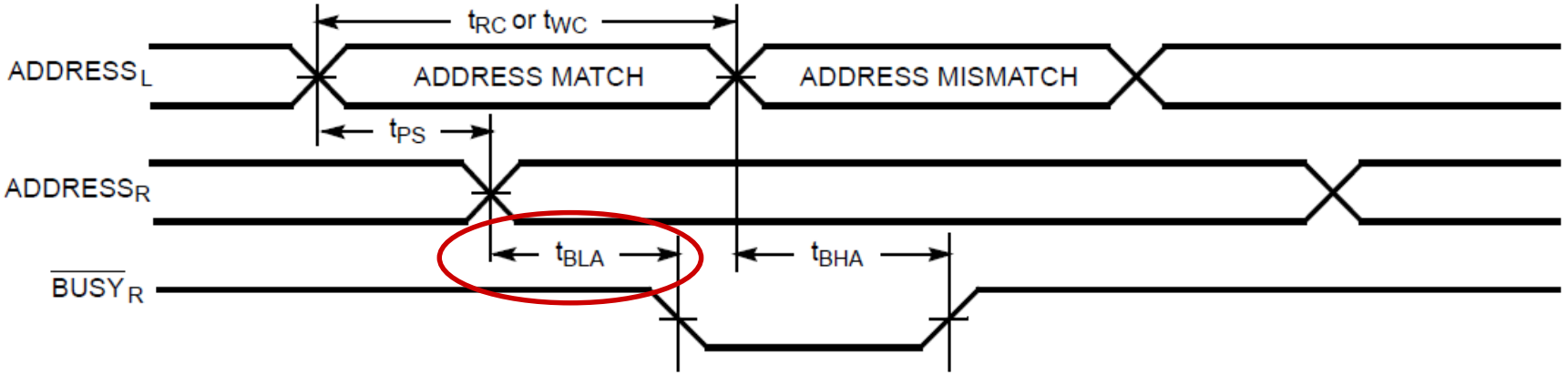
Left Port	Right Port	Description
$\overline{CE}_L$	$\overline{CE}_R$	Chip Enable
$R/\overline{W}_L$	$R/\overline{W}_R$	Read/Write Enable
$\overline{OE}_L$	$\overline{OE}_R$	Output Enable
A_{0L}–A_{14L}	A_{0R}–A_{14R}	Address
I/O_{0L}–I/O_{8L}	I/O_{0R}–I/O_{8R}	Data Bus Input/Output (I/O_0–I/O_7 for x8 devices and I/O_0–I/O_8 for x9)
$\overline{SEM}_L$	$\overline{SEM}_R$	Semaphore Enable
$\overline{INT}_L$	$\overline{INT}_R$	Interrupt Flag
$\overline{BUSY}_L$	$\overline{BUSY}_R$	Busy Flag
$M/\overline{S}$		Master or Slave Select
V_{CC}		Power
GND		Ground
NC		No Connect

4.4. CY7C007/016A (Interrupt Timing)

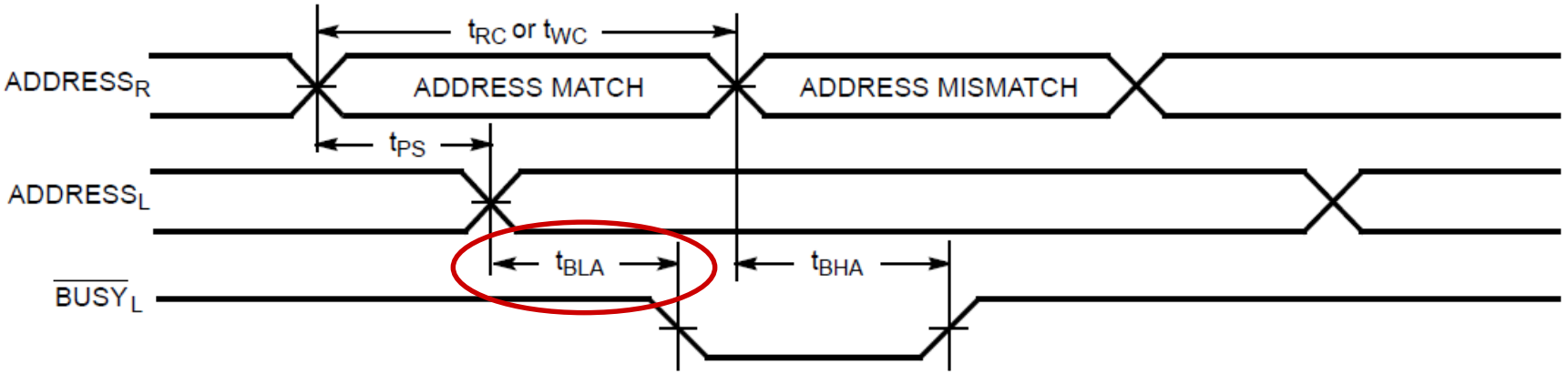


4.4. CY7C007/016A (Busy Timing)

Busy Timing Diagram No. 2 (Address Arbitration)[40]
Left Address Valid First:



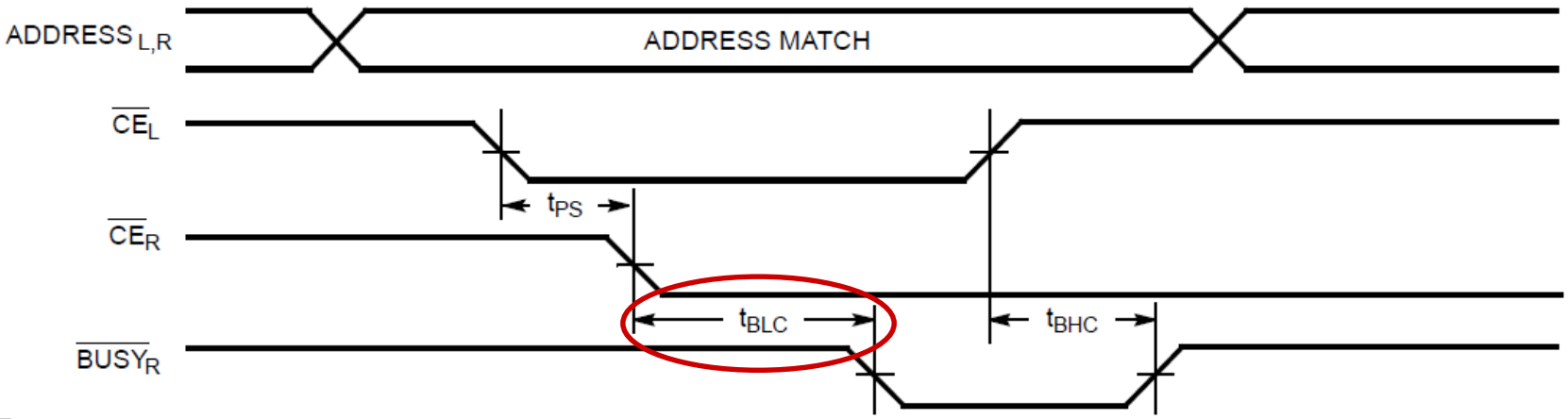
Right Address Valid First:



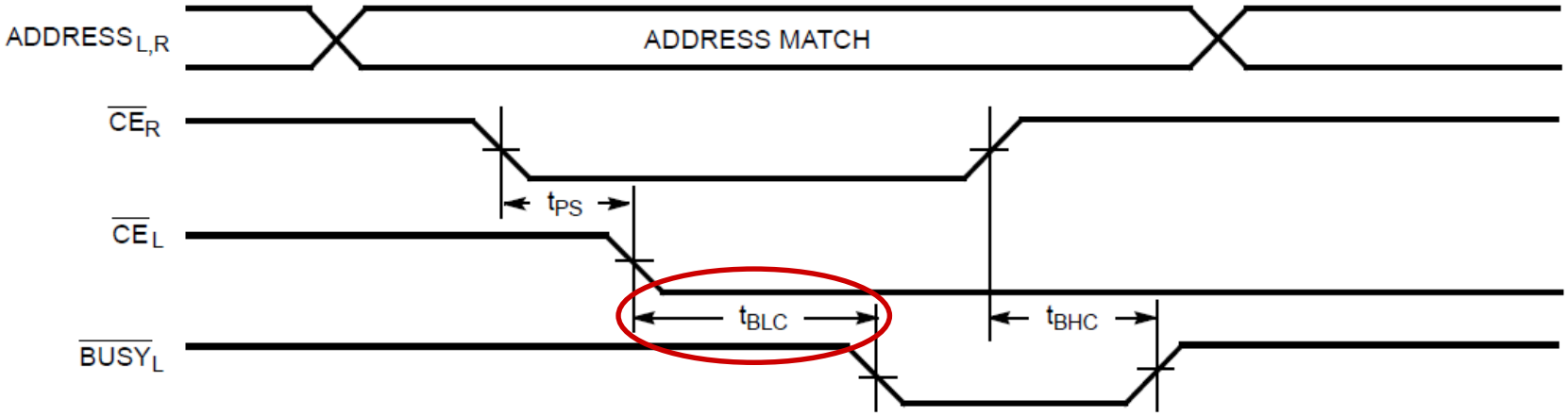
Note:
40. If t_{PS} is violated, the busy signal will be asserted on one side or the other, but there is no guarantee to which side $\overline{BUSY}$ will be asserted.

4.4. CY7C007/016A (Busy Timing)

Busy Timing Diagram No. 1 ($\overline{CE}$ Arbitration)[40]
CE_L Valid First:



$\overline{CE}_R$ Valid First:



4.4. CY7C007/016A (Semaphore Operation)

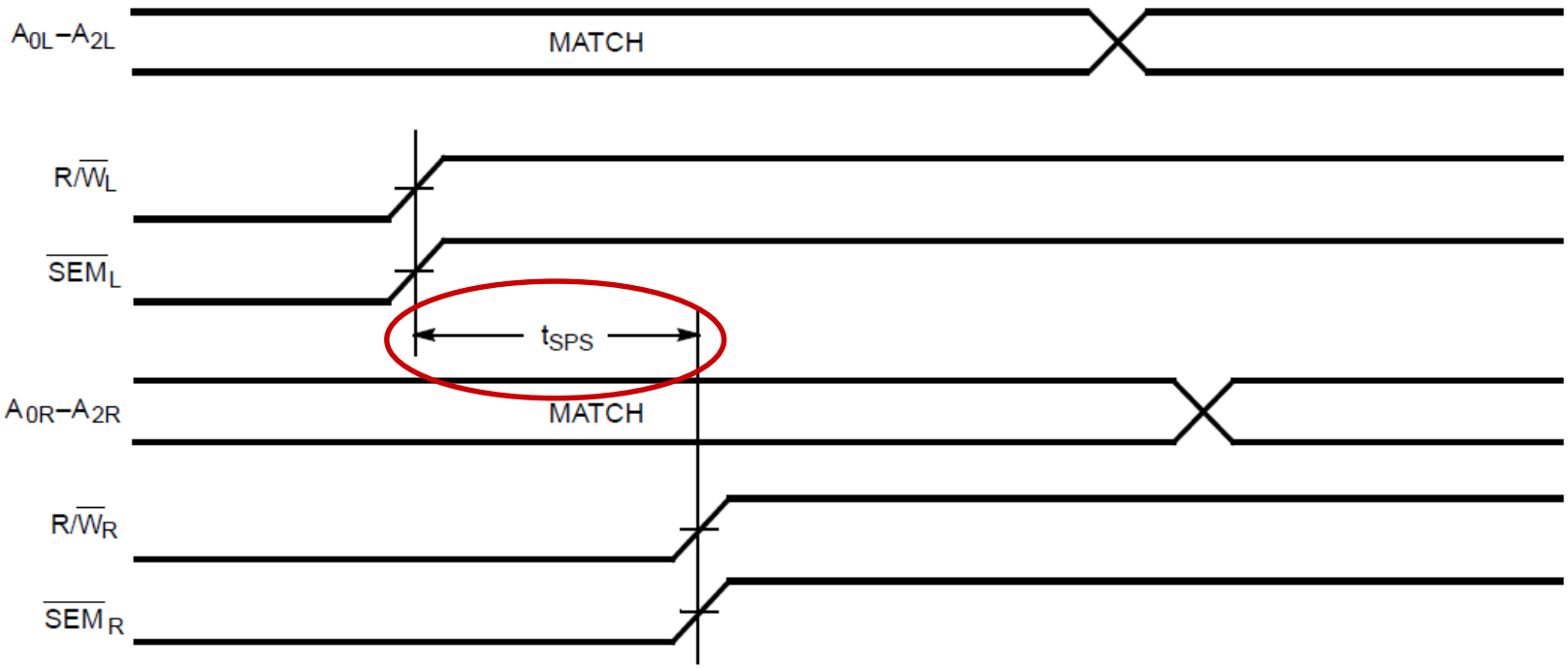
Table 3. Semaphore Operation Example

Function	I/O_0–I/O_8 Left	I/O_0–I/O_8 Right	Status
No action	1	1	Semaphore free
Left port writes 0 to semaphore	0	1	Left Port has semaphore token
Right port writes 0 to semaphore	0	1	No change. Right side has no write access to semaphore
Left port writes 1 to semaphore	1	0	Right port obtains semaphore token
Left port writes 0 to semaphore	1	0	No change. Left port has no write access to semaphore
Right port writes 1 to semaphore	0	1	Left port obtains semaphore token
Left port writes 1 to semaphore	1	1	Semaphore free
Right port writes 0 to semaphore	1	0	Right port has semaphore token
Right port writes 1 to semaphore	1	1	Semaphore free
Left port writes 0 to semaphore	0	1	Left port has semaphore token
Left port writes 1 to semaphore	1	1	Semaphore free

Notes:
43. If $\overline{BUSY}_R$ = L, then no change.
44. If $\overline{BUSY}_L$ = L, then no change.

4.4. CY7C007/016A (Semaphore Contention)

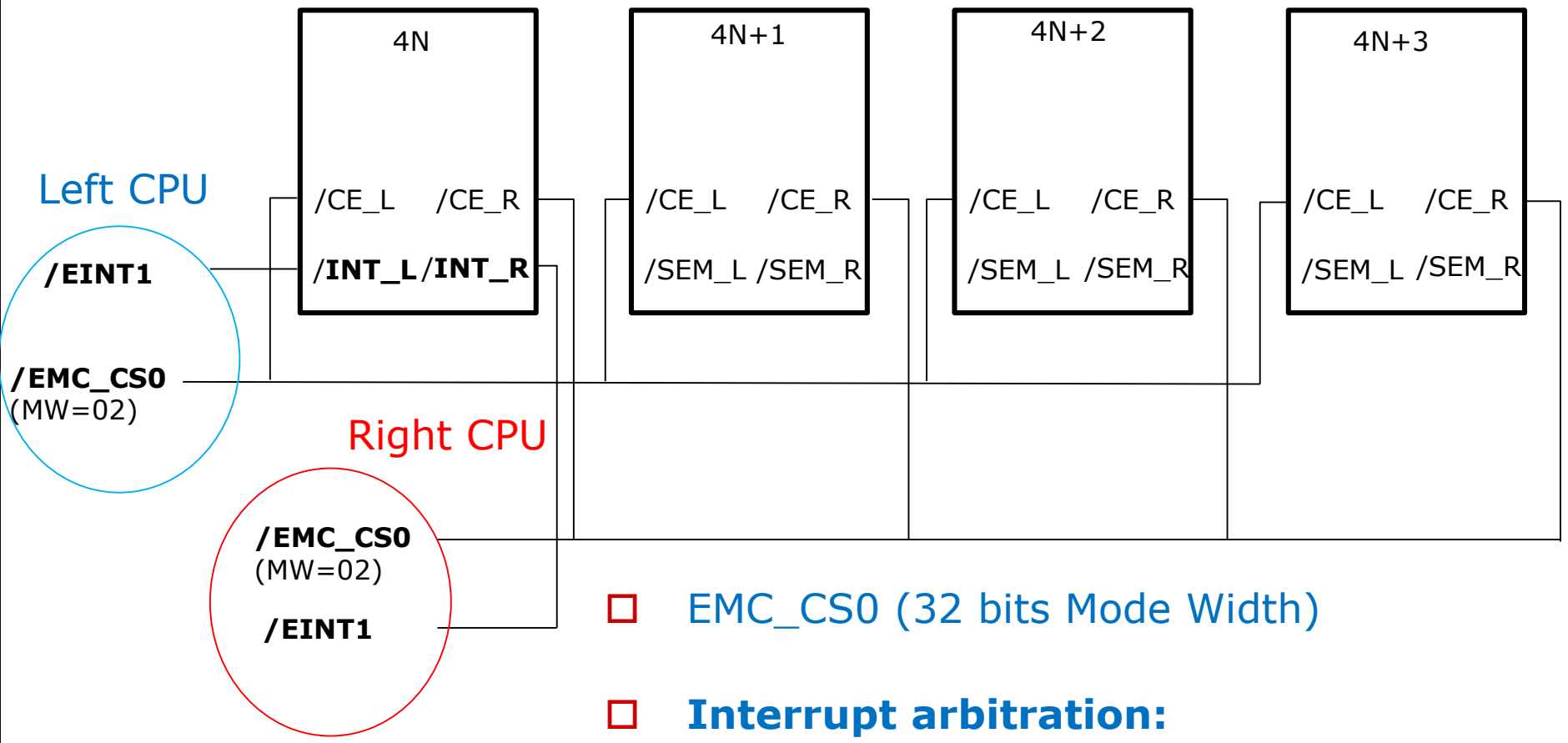
Timing Diagram of Semaphore Contention[36, 37, 38]



Notes:

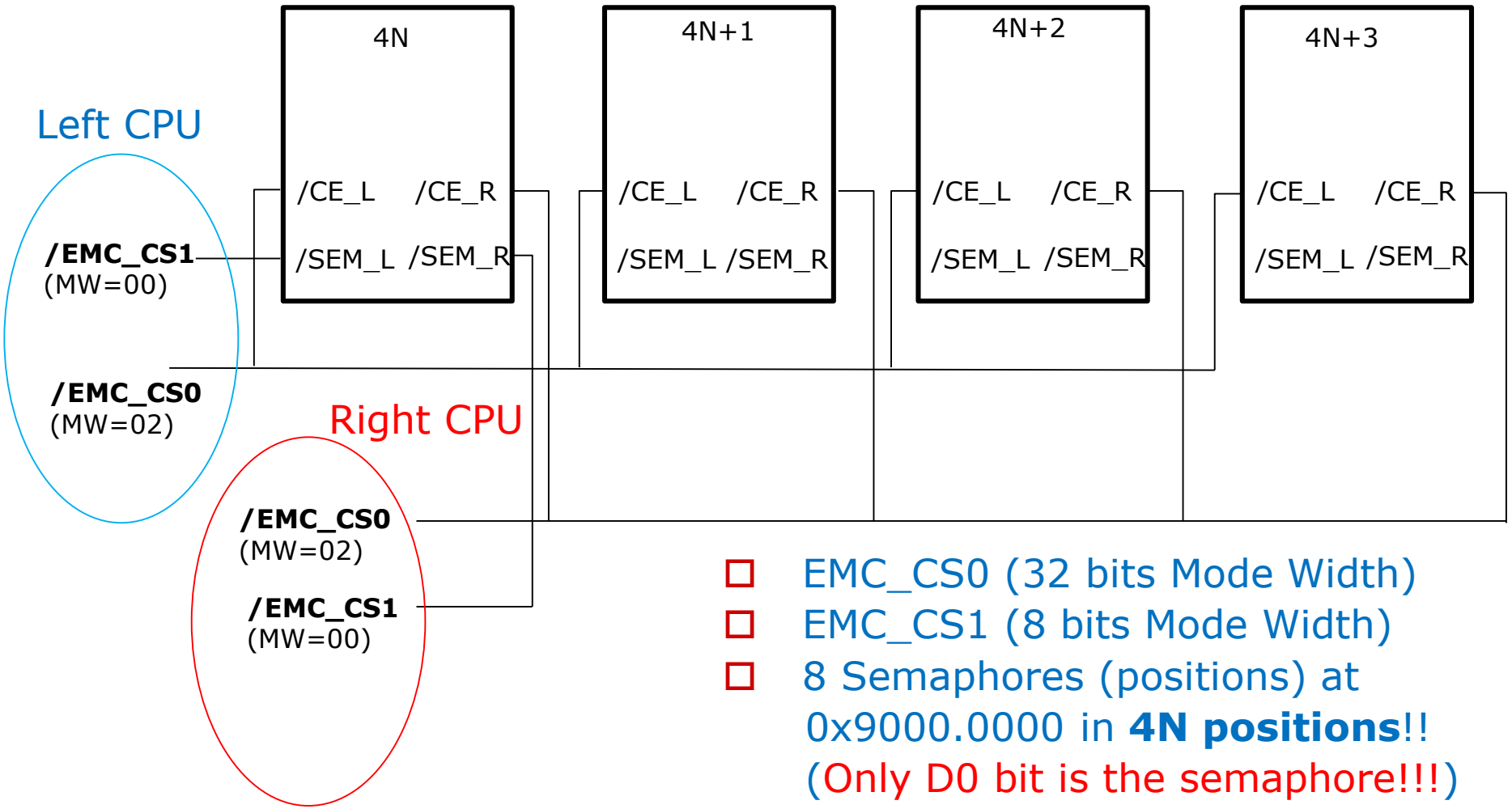
35. $\overline{CE}$ = HIGH for the duration of the above timing (both write and read cycle).
36. $I/O_{0R} = I/O_{0L}$ = LOW (request semaphore); $\overline{CE}_R = \overline{CE}_L$ = HIGH.
37. Semaphores are reset (available to both ports) at cycle start.
38. If t_{SPS} is violated, the semaphore will definitely be obtained by one side or the other, but which side will get the semaphore is unpredictable.

4.4. LPC178x EMC connection to DP (Example I)



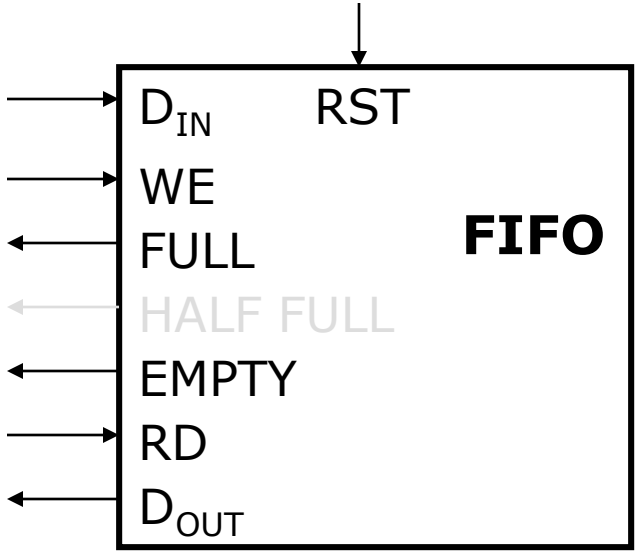
- EMC_CS0 (32 bits Mode Width)
- **Interrupt arbitration:**
  Two last 4N positions of memory 1 !!

4.4. LPC178x EMC connection to DP (Example II)



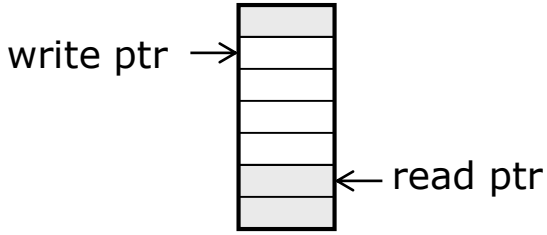
- EMC_CS0 (32 bits Mode Width)
- EMC_CS1 (8 bits Mode Width)
- 8 Semaphores (positions) at 0x9000.0000 in **4N positions**!! (Only D0 bit is the semaphore!!!)

4.5. First-Input-First-Output (FIFO)

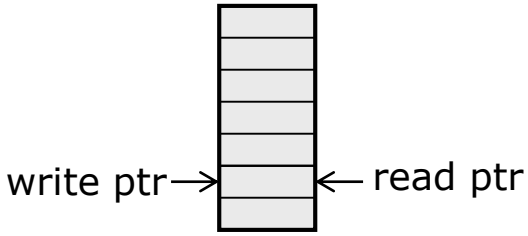


- After write or read operation, FULL and EMPTY indicate status of buffer.
- Used by external logic to control own reading from or writing to the buffer.
- FIFO resets to EMPTY state.
- HALF FULL (or other indicator of partial fullness) is optional.

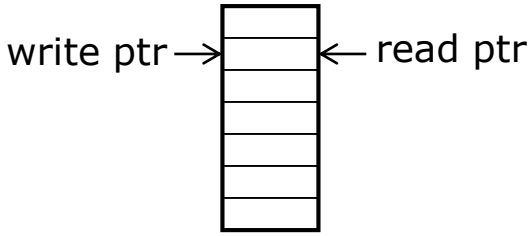
- Address pointers are used internally to keep next write position and next read position into a dual-port memory.



- If pointers equal after write ⇒ **FULL**:



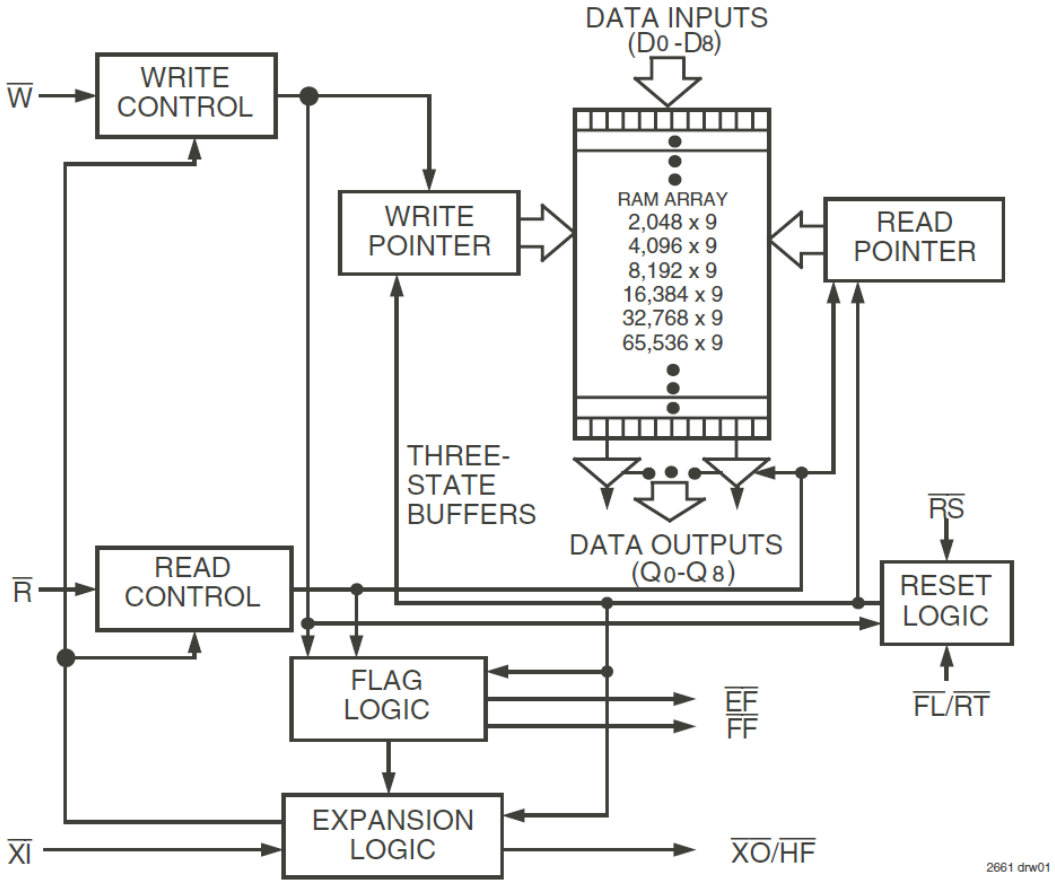
- If pointers equal after read ⇒ **EMPTY**:



4.5. FIFO: IDT72xx

CMOS ASYNCHRONOUS FIFO
2,048 x 9, 4,096 x 9
8,192 x 9, 16,384 x 9
32,768 x 9 and 65,536 x 9

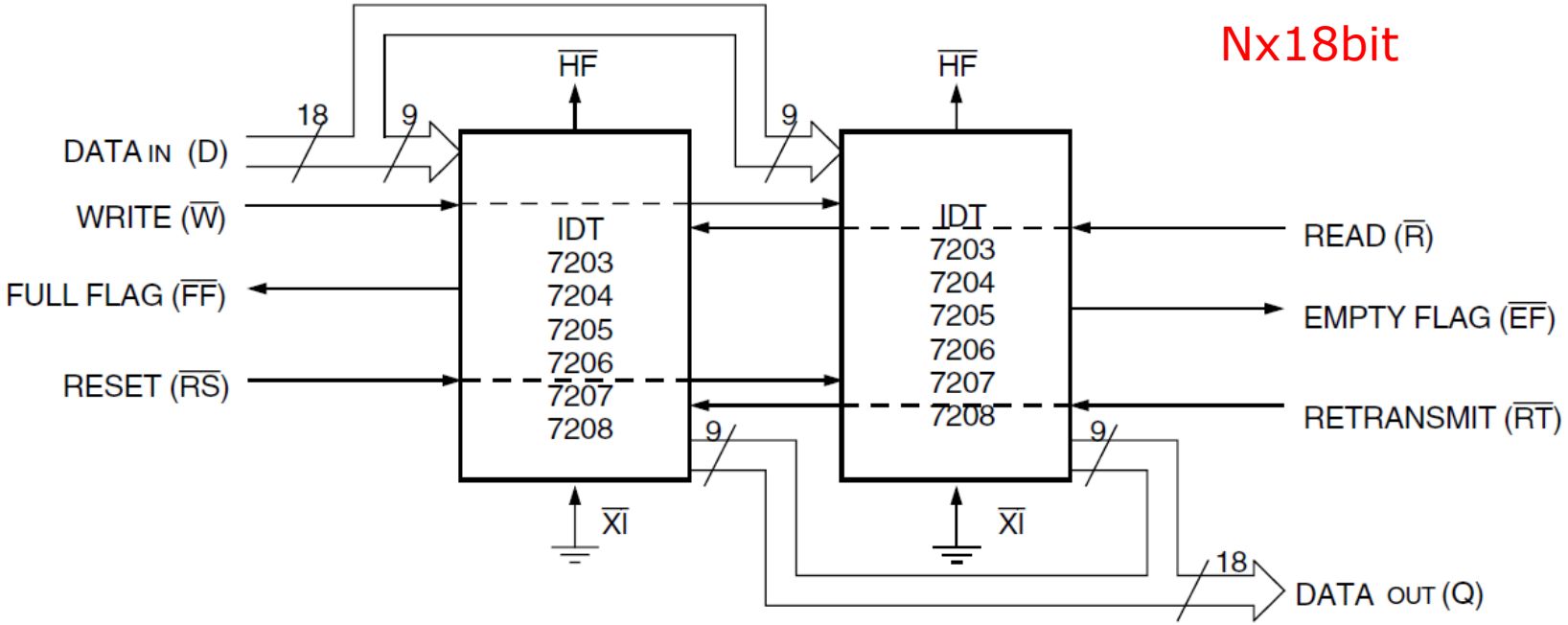
- Based in dual-port memory buffers with internal pointers that load and empty data on a basis FIFO resets to EMPTY state.
- The device's 9-bit width provides a bit for a control or parity at the user's option.
- Signals control for fully expandable in both word depth and width (XI, XO, FL).
- **Retransmit** (RT) capability that allows the read pointer to be reset to its initial position when RT is pulsed LOW.



4.5. FIFO: Word Width expansion capacity

Chips Nx9bit

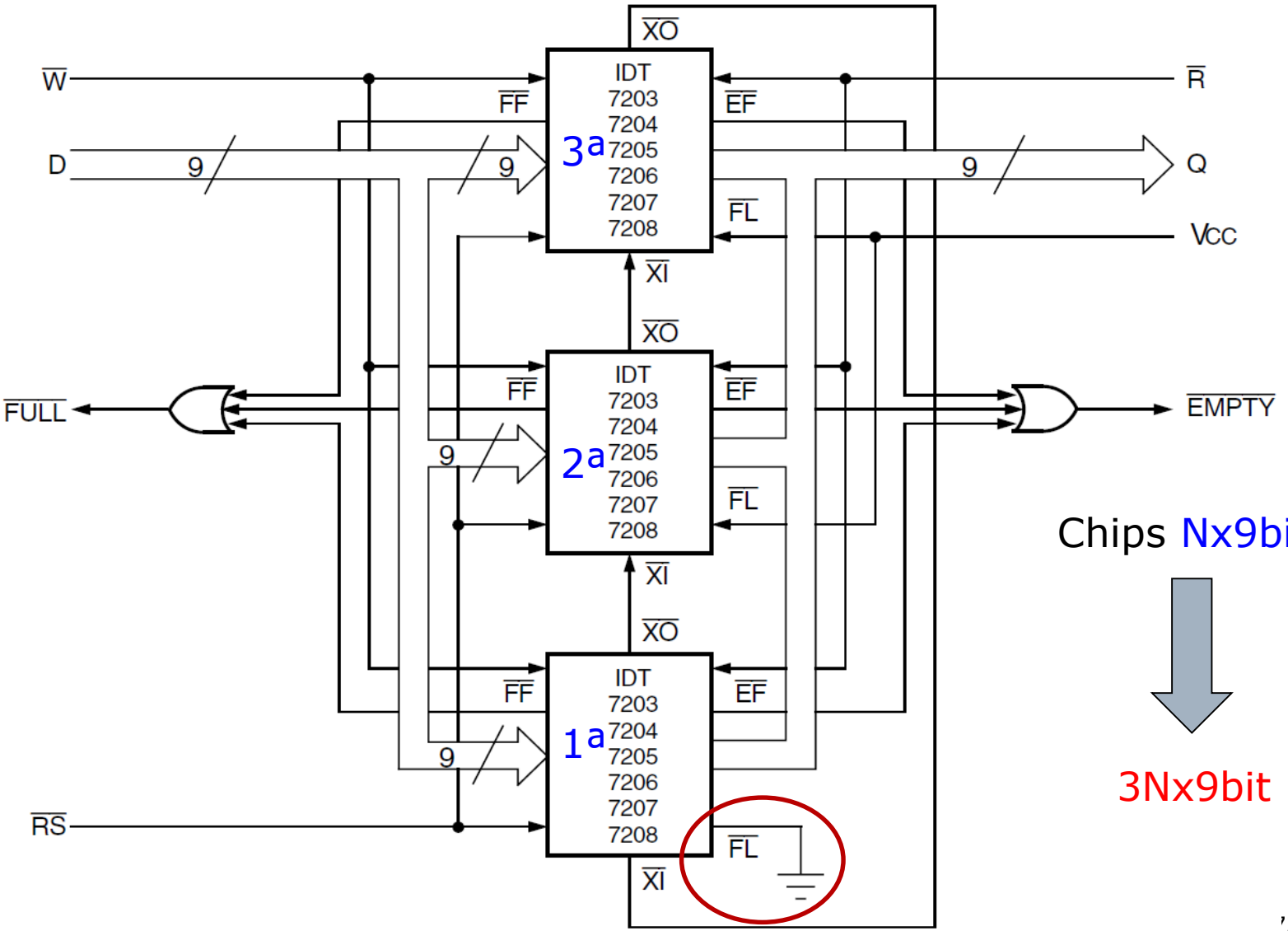
Nx18bit



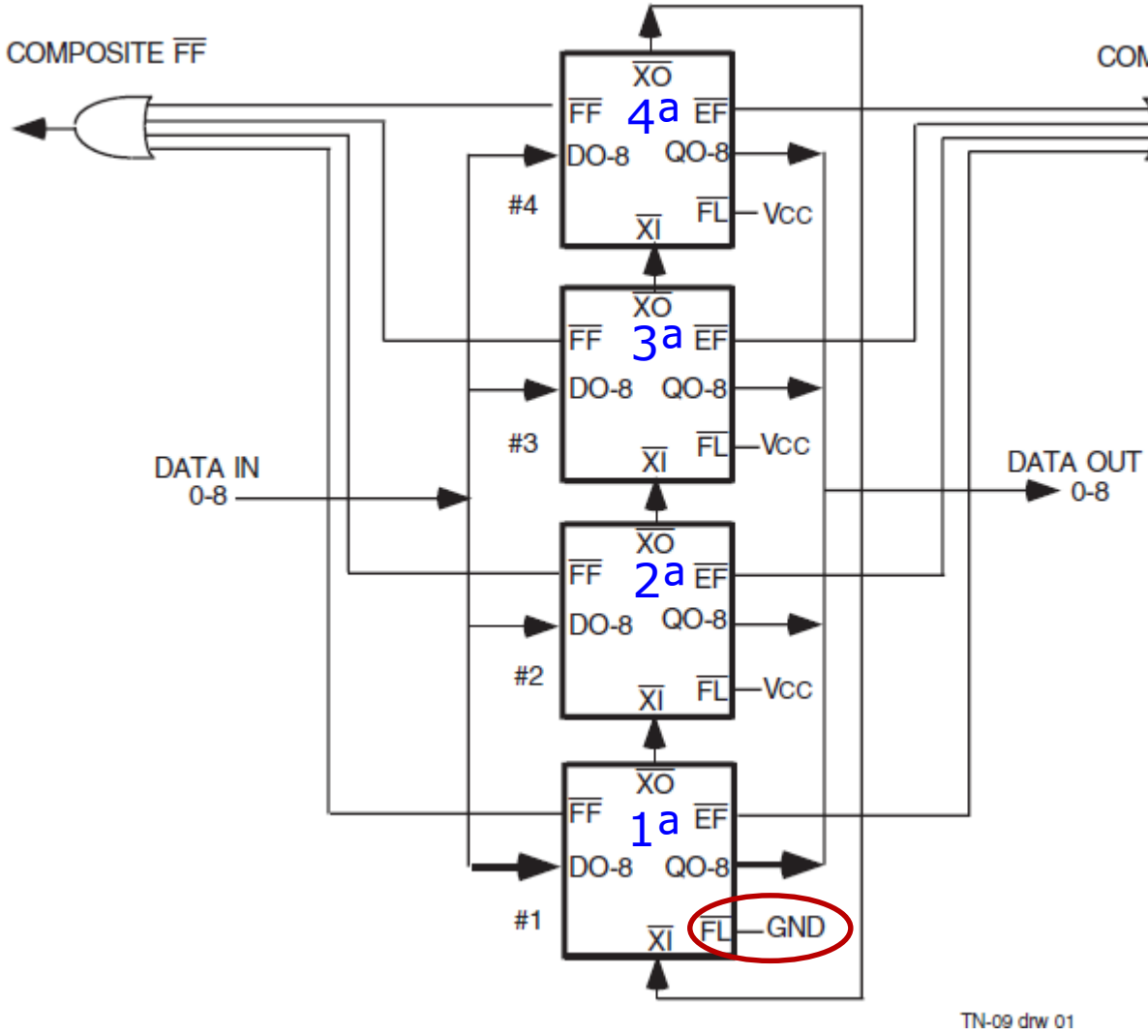
NOTE:
1. Flag detection is accomplished by monitoring the FF, EF and HF signals on either (any) device used in the width expansion configuration. Do not connect any output signals together.

2661 drw15

4.5. FIFO: Word Depth expansion capacity



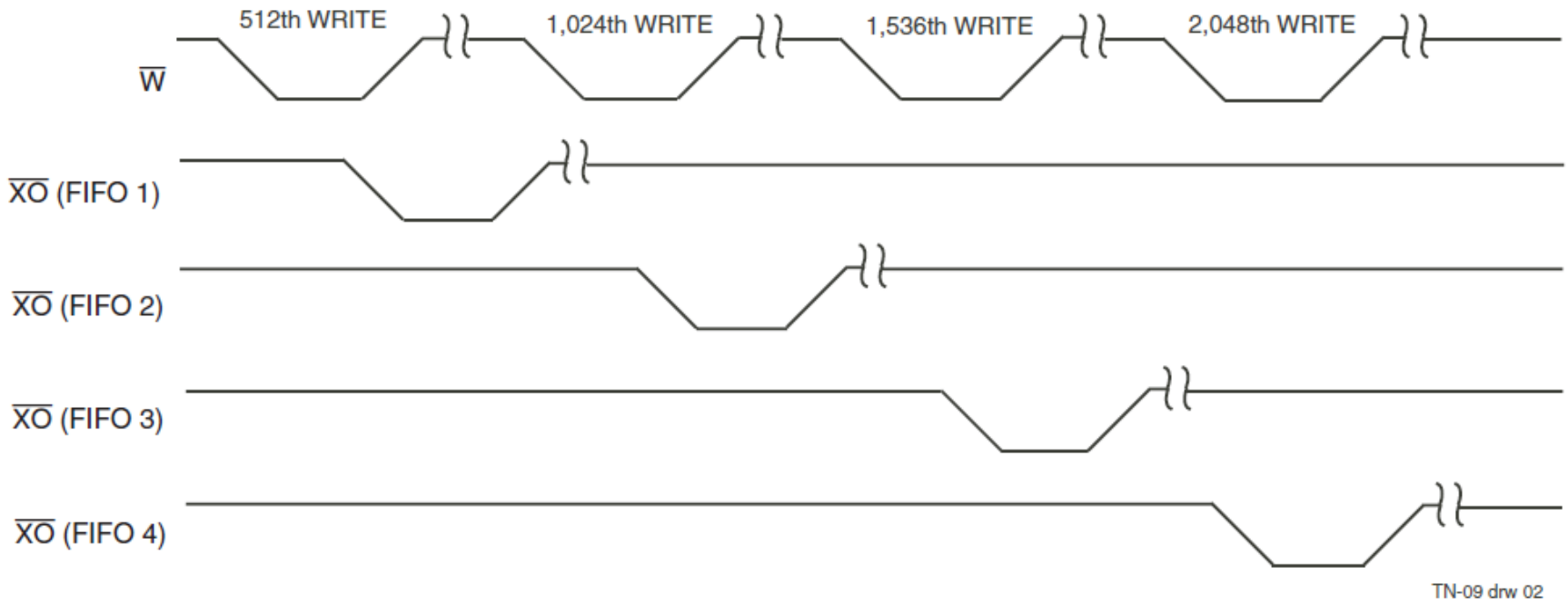
4.5. FIFO: Word Depth expansion capacity



Chips 512x9bit

2048x9bit

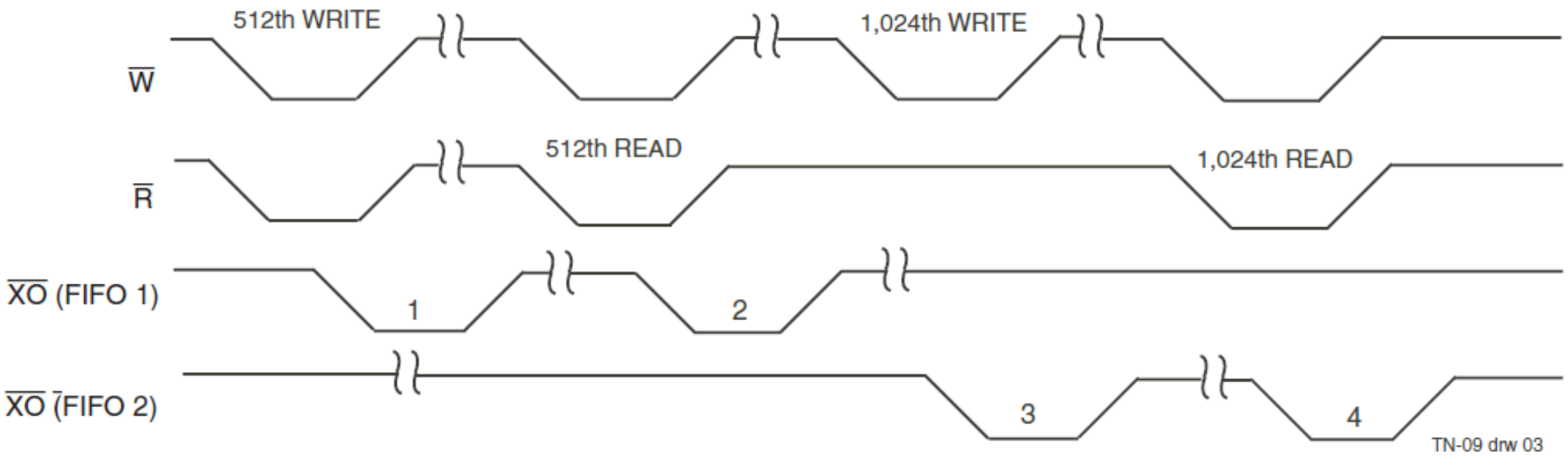
4.5. FIFO: Word Depth expansion (Timing)



NOTE:
Read line is assumed to be HIGH in this example

Figure 2. The XO/XI Timing Pulse for 2,048 Writes and Zero Reads

4.5. FIFO: Word Depth expansion (Timing)



NOTES:

1. Pulse 1 is created by the 512th write pulse; it is a delayed write pulse.
2. Pulse 2 is created by the 512th read pulse.
3. Pulse 3 from FIFO 2 is created by the 1,024th write pulse.
4. Pulse 4 is created by the 1,024th read pulse.
5. $\overline{XO}$ (FIFO 3) and $\overline{XO}$ (FIFO 4) are not shown, but they follow the same pattern.
6. $\overline{XO}$ (FIFO 4) will be created by the 2,048th write pulse and later by the 2,048th read pulse, thereby transferring pointer control back to FIFO 1.

Figure 3. The $\overline{XO}$ and $\overline{XI}$ PulseTimings

4.4. FIFO interface to 8 bit ADC: High rate data adquisition to external memory (example)

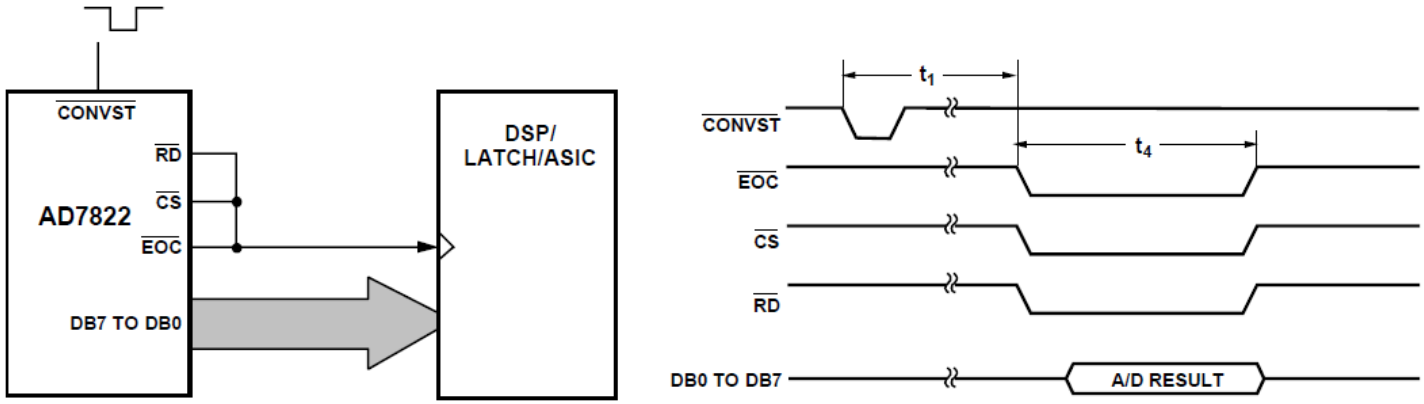
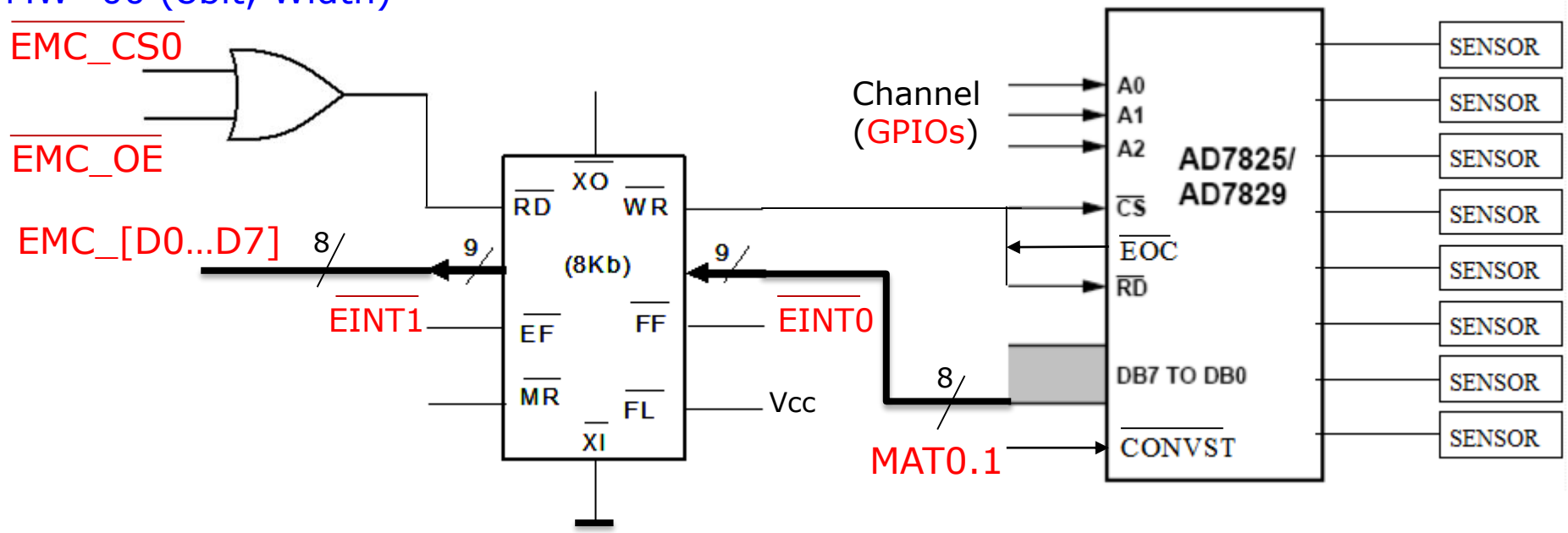


Figure 31. AD7822 Standalone Operation

MW=00 (8bit, Width)



4.4. FIFO interface to 16 bit ADC: High rate data adquisition to external memory (example)