2024年度(令和6年)版

Ver. 2024-10-11a

Course number: CSC.T363

コンピュータアーキテクチャ
Computer Architecture

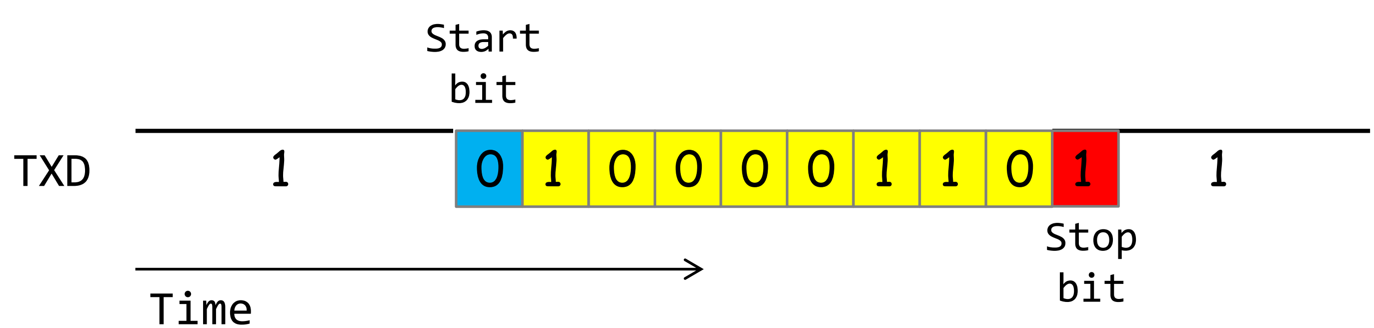
3. 半導体メモリ
Memory Technologies

www.arch.cs.titech.ac.jp/lecture/CA/
Tue 13:30-15:10, 15:25-17:05
Fri 13:30-15:10

吉瀬 謙二 情報工学系
Kenji Kise, Department of Computer Science
kise _at_ c.titech.ac.jp

UART (Universal Asynchronous Receiver/Transmitter)

- 調歩同期方式によるシリアル信号をパラレル信号に変換したり，その逆方向の変換をおこなう集積回路をUARTと呼ぶ．8ビット(1バイト)単位でデータを送信・受信する．
- UARTを用いることで，FPGAとコンピュータの間でのお手軽なデータ通信が可能．
- 例えば，'a' という文字を送信する場合，'a' は 8'h61, 8'b01100001 (次スライドのASCII Tableを参照)なので，下図のタイミングで送信線TXDを制御する．
  - データが送信されるまで送信線TXD を1とする．
  - まず，青色で示した0 (これをスタートビットと呼ぶ)を送信することで，データ送信の開始を明示．
  - 次に，黄色で示した様に送信したいデータ 8'b01100001 の最下位ビットから順番に送信する．
  - 最後に，赤色で示した1(これをストップビットと呼ぶ)を送信する．
- 1ビットを送受信するための時間間隔は送信側と受信側で同じレートを用いる．これをボー・レート (baud) と呼ぶ．例えば，1000 baud であれば，1ビット送信の間隔は 1msec となる．



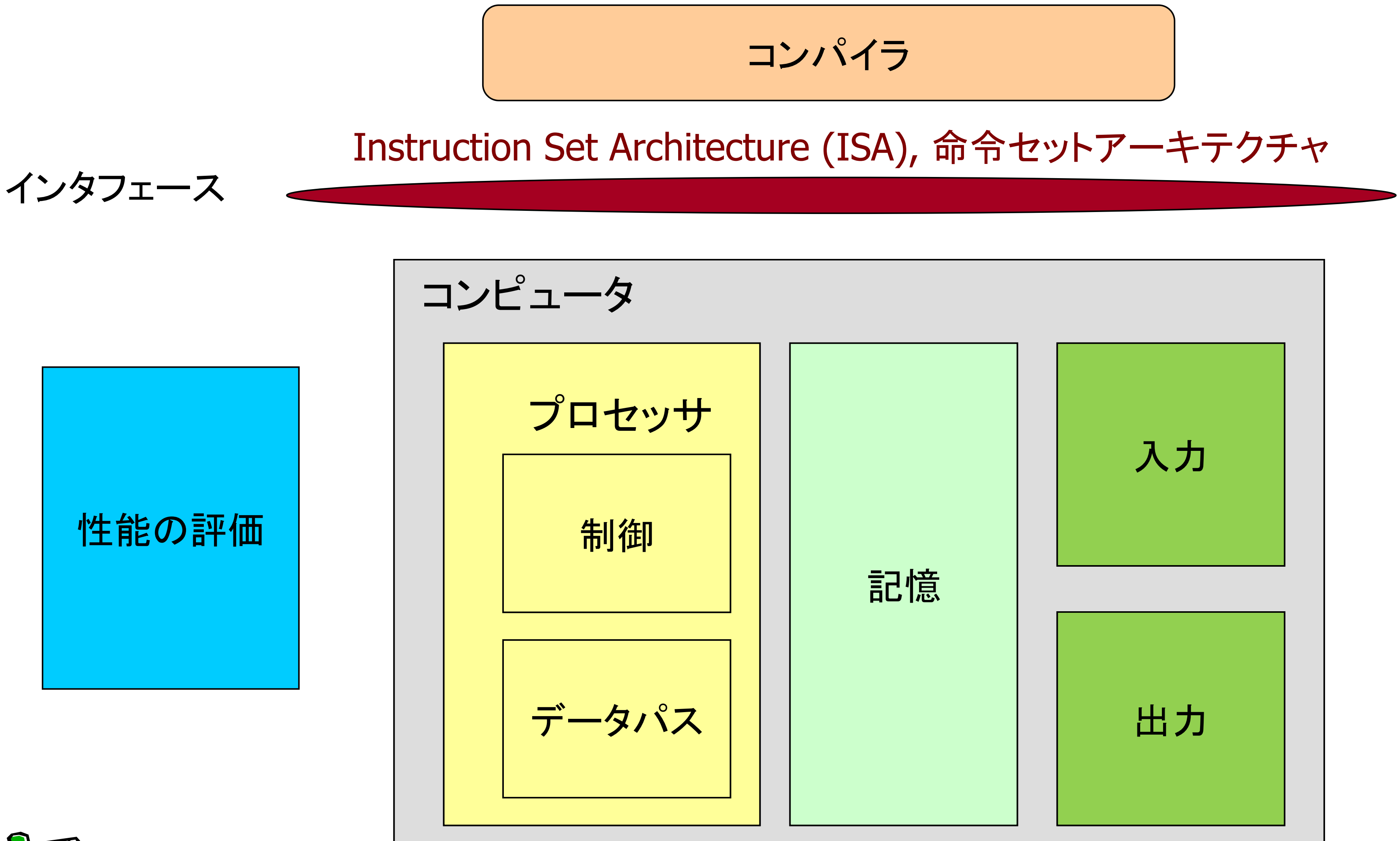
シリアル通信による受信回路 m_uart_rx

- システムクロック 100MHz，1Mbaud を想定する受信回路．
- 受信した8ビットのデータを r_dout に出力し，そのことを伝えるために r_en を1にする．

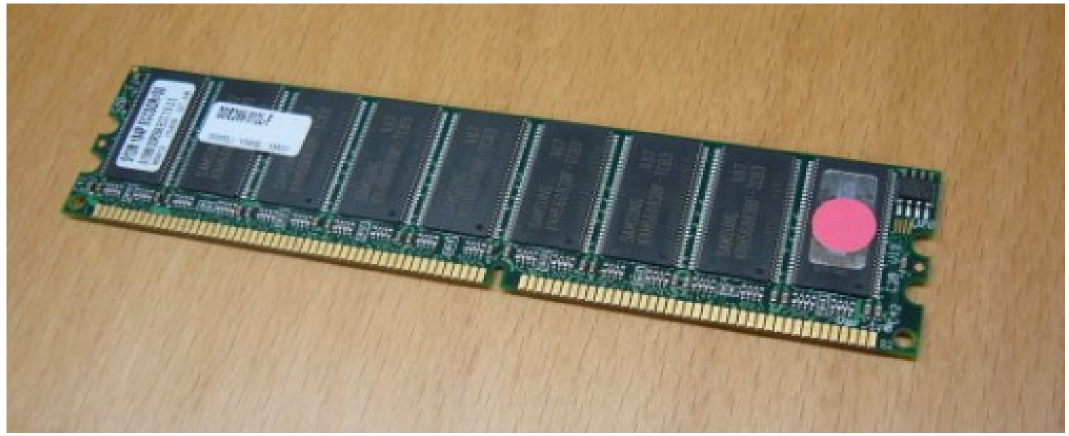
code202.v

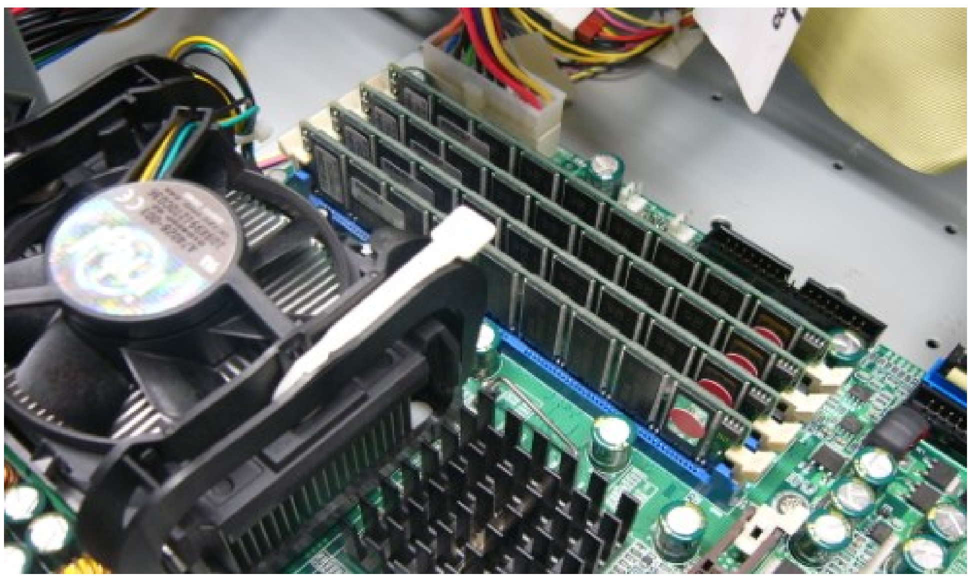
```
module m_uart_rx (
    input  wire       w_clk,   // 100MHz clock signal
    input  wire       w_rxd,   // UART rx, data line from PC to FPGA
    output wire [7:0] w_char,  // 8-bit data received
    output reg        r_en = 0 // data enable
);
    reg [2:0] r_detect_cnt = 0; /* to detect the start bit */
    always @(posedge w_clk) r_detect_cnt <= (w_rxd) ? 0 : r_detect_cnt + 1;
    wire w_detected = (r_detect_cnt>2);

    reg       r_busy = 0; // r_busy is set while receiving 9-bits data
    reg [3:0] r_bit  = 0; // the number of received bits
    reg [7:0] r_cnt  = 0; // wait count for 1Mbaud
    always@(posedge w_clk) r_cnt <= (r_busy==0) ? 1 : (r_cnt==`UART_CNT) ? 1 : r_cnt + 1;

    reg [8:0] r_data = 0;
    always@(posedge w_clk) begin
        if (r_busy==0) begin
            {r_data, r_bit, r_en} <= 0;
            if (w_detected) r_busy <= 1;
        end
        else if (r_cnt>= `UART_CNT) begin
            r_bit <= r_bit + 1;
            r_data <= {w_rxd, r_data[8:1]};
            if (r_bit==8) begin r_en <= 1; r_busy <= 0; end
        end
    end
    assign w_char = r_data[7:0];
endmodule
```

コンピュータの古典的な要素



DRAM (dynamic random access memory)



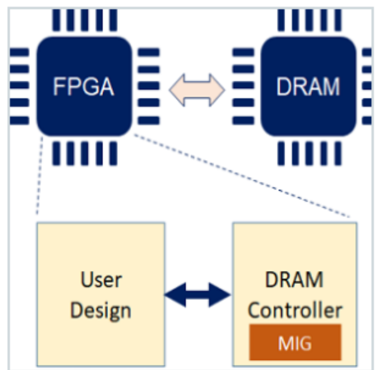


参考資料

- MIG を使って DRAM メモリを動かそう (1)
  - https://www.acri.c.titech.ac.jp/wordpress/archives/6048

ACRi

MIG を使って DRAM メモリを動かそう (1)

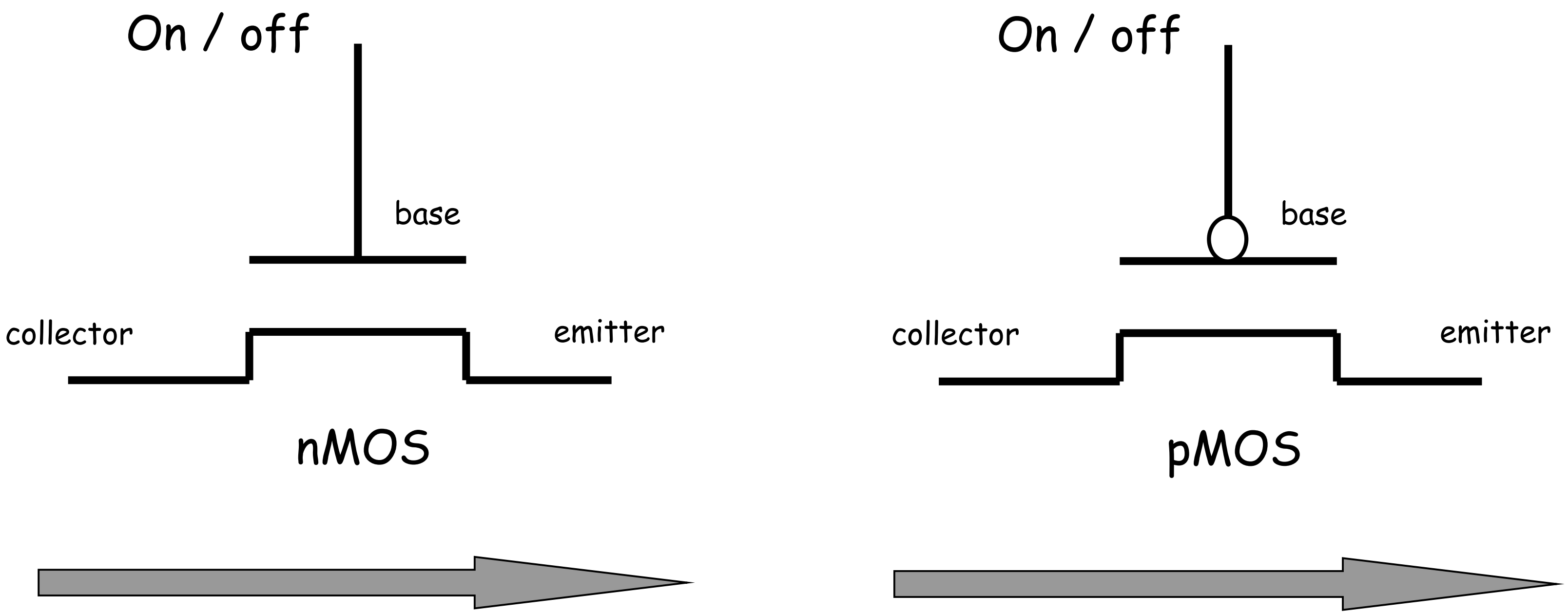


2020.09.20 2020.07.16

みなさんこんにちは。この「MIG を使って DRAM メモリを動かそう」のシリーズでは、全5回を通じて Xilinx Memory Interface Generator (MIG) という IP コアをベースに Xilinx FPGA で DRAM メモリを動かす方法を紹介していきます。説明では教育向けに設計された Arty A7-35T FPGA ボードを用いますが、他の FPGA ボードにも同様に適用できます。

Transistor

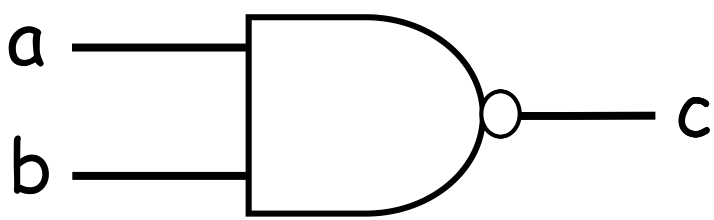
- トランジスタは、電気的なオン／オフ動作をするスイッチと捉えることができる。



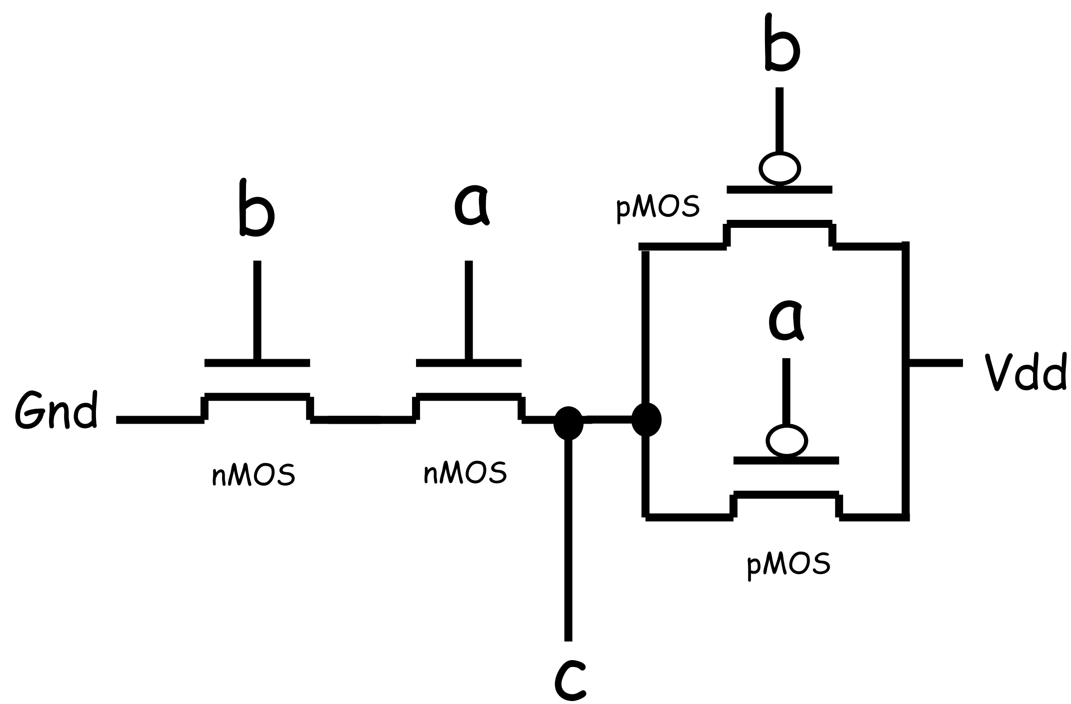
Transistor and Gate

- トランジスタは電気的なオン／オフ動作をするスイッチ
- 幾つかのトランジスタで，少し機能の高いゲートを構成する。

NANDゲート

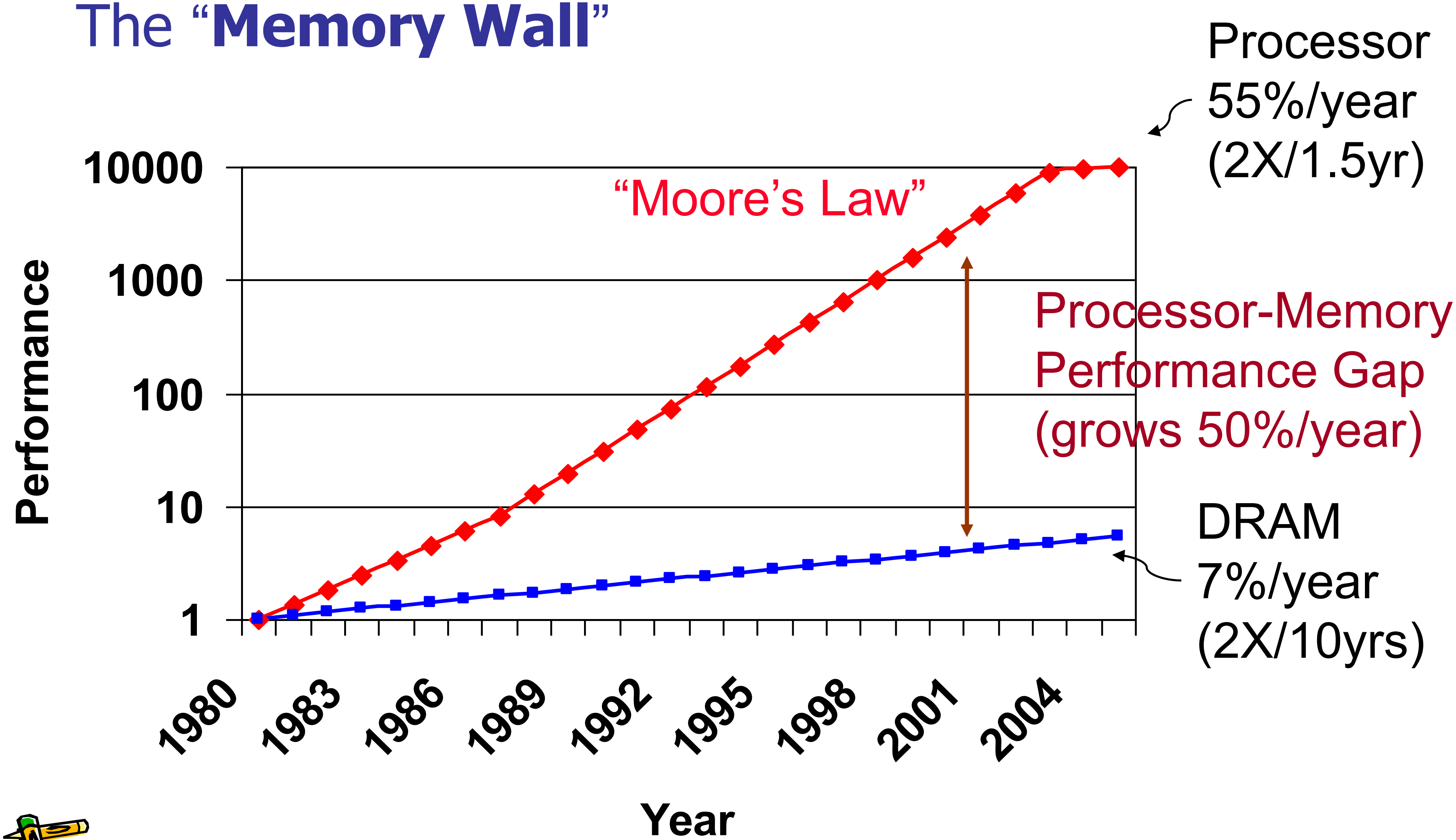


a	b	c
0	0	1
1	0	1
0	1	1
1	1	0



Processor-Memory(DRAM) Performance Gap

The “**Memory Wall**”



The Memory System's **Fact and Goal**

Fact:

Large memories are slow, and
fast memories are small

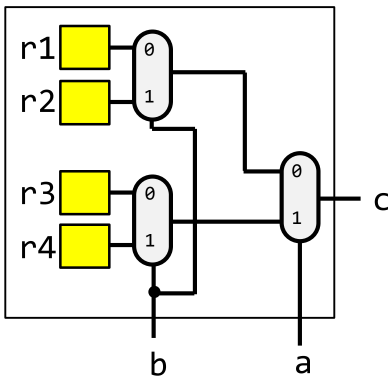
How do we create a memory that gives the **illusion** of being large, fast, and cheap ?

With **hierarchy** （階層）

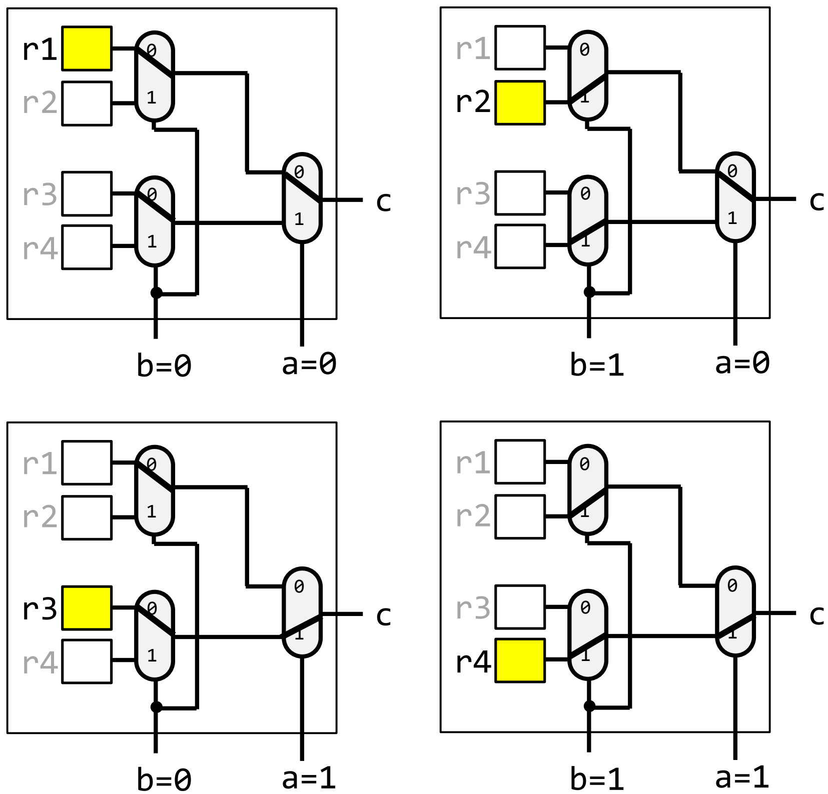
With **parallelism** （並列性）

ルックアップテーブル (Lookup Table, LUT)

a, b を入力として、c を出力とする LUT
値を保持するレジスタ(黄色)の値を選択する回路



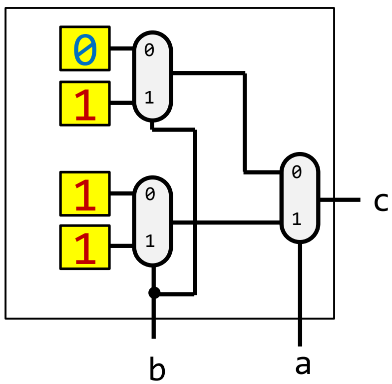
2入力のLUTの構成



入力		出力
a	b	c
0	0	r1
0	1	r2
1	0	r3
1	1	r4

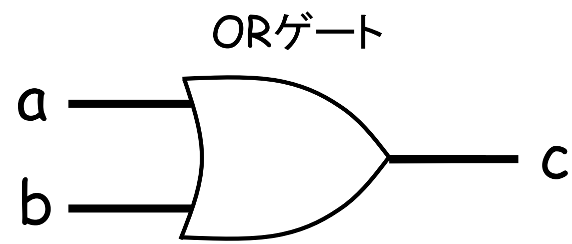
ルックアップテーブル (Lookup Table, LUT)

- レジスタの値を上から 0, 1, 1, 1 に設定すると、このLUTはORゲートと同じ動作をする。

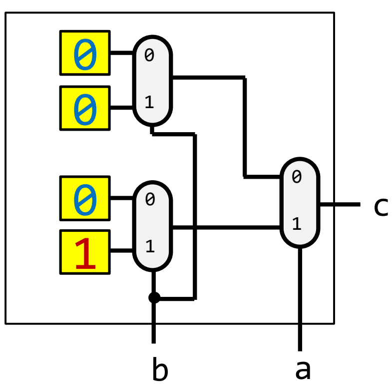


a	b	c
0	0	0
0	1	1
1	0	1
1	1	1

真理値表

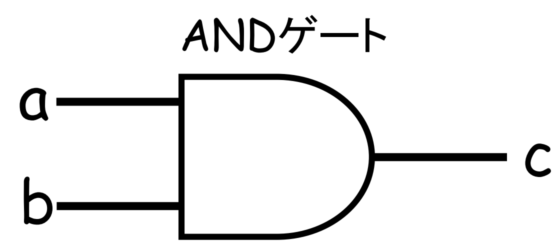


- レジスタの値を上から 0, 0, 0, 1 に設定すると、このLUTはANDゲートと同じ動作をする。

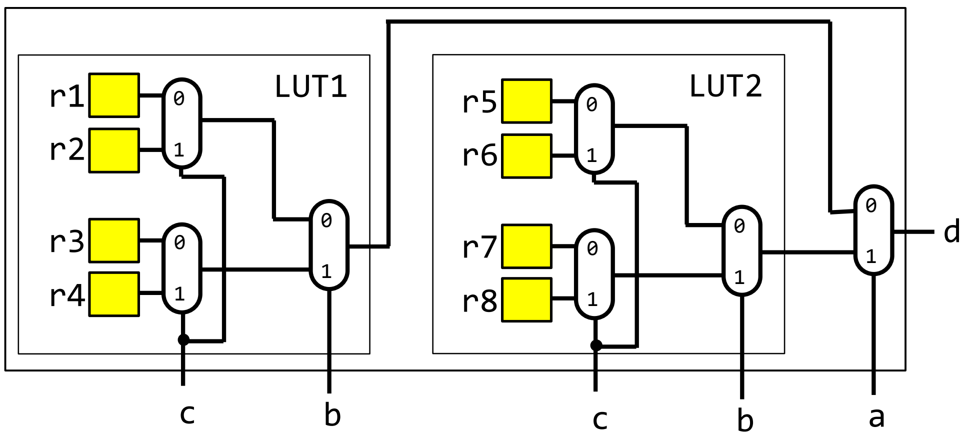


a	b	c
0	0	0
0	1	0
1	0	0
1	1	1

真理値表



ルックアップテーブル (Lookup Table, LUT)



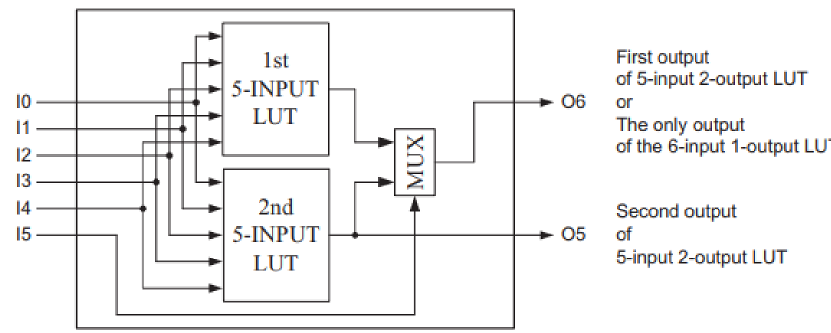
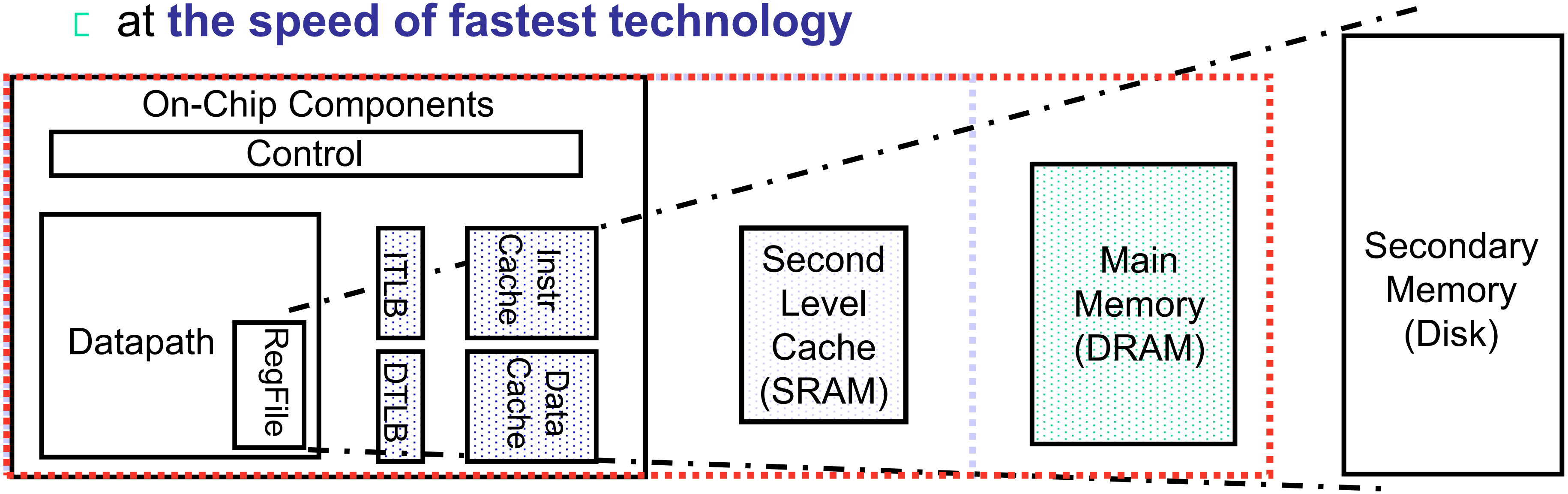


Fig. 3. Choice between 1-output 6-input LUT and 2-output 5-input LUT in Xilinx FPGA devices.

2個の2入力のLUTで3入力のLUTを構成

A Typical Memory Hierarchy

- By taking advantage of **the principle of locality** (局所性)
  - Present **much memory** in **the cheapest technology**
  - at **the speed of fastest technology**



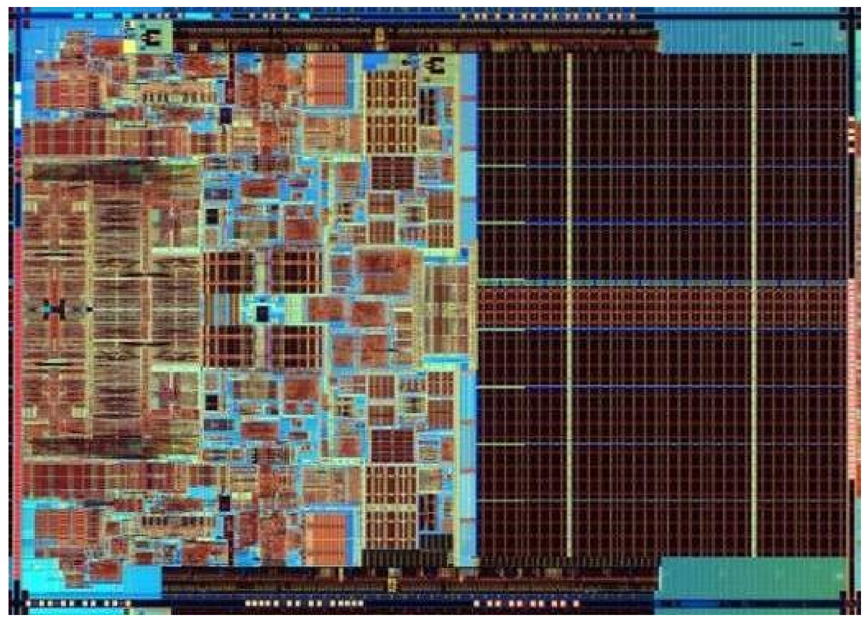
Speed (%cycles):	½'s	1's	10's	100's	1,000's
Size (bytes):	100's	K's	10K's	M's	G's to T's
Cost:	highest				lowest

TLB: Translation Lookaside Buffer

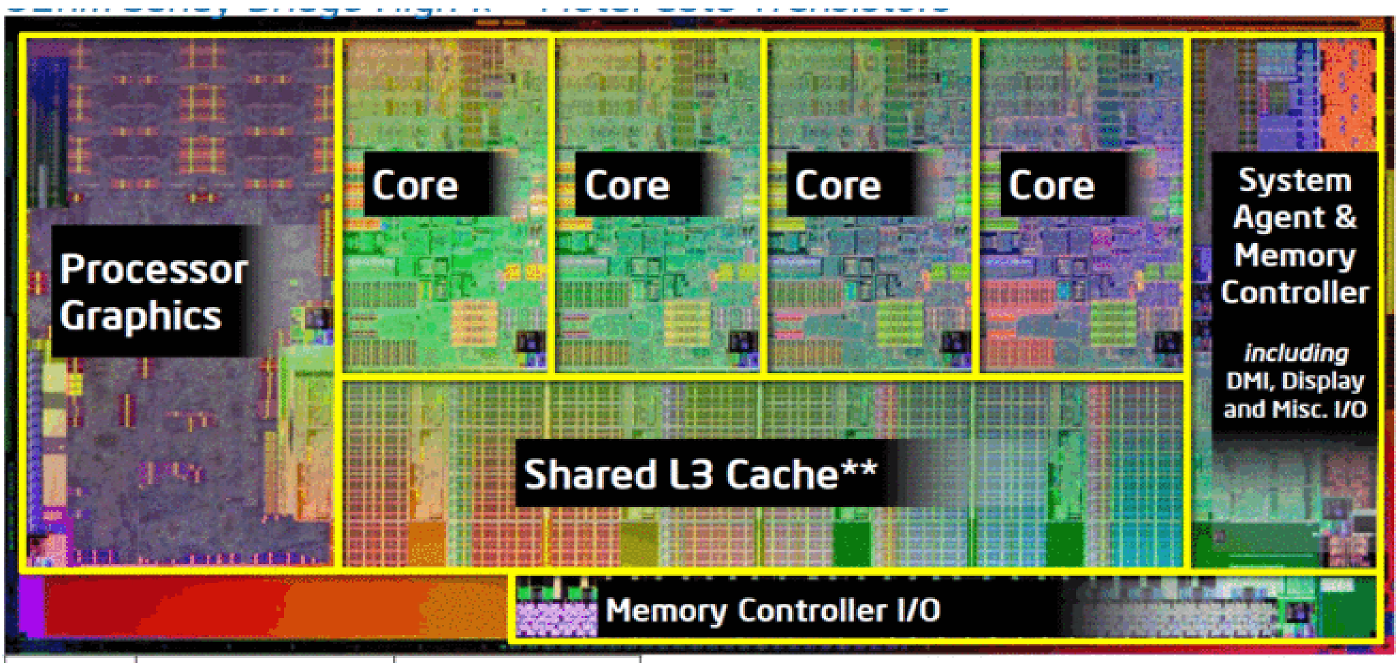
Cache

- *Cache memory* consists of a small, fast memory that acts as a buffer for the large memory.
- The nontechnical definition of *cache* is a safe place for hiding things.

Intel Core 2 Duo

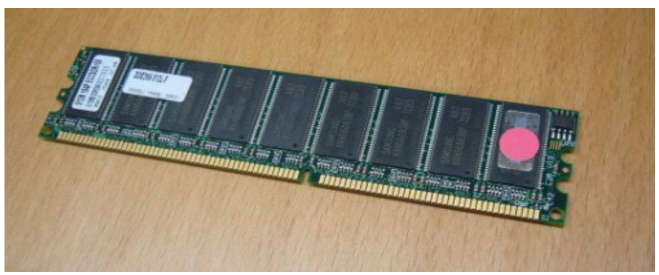


Intel Sandy Bridge, January 2011



Processor

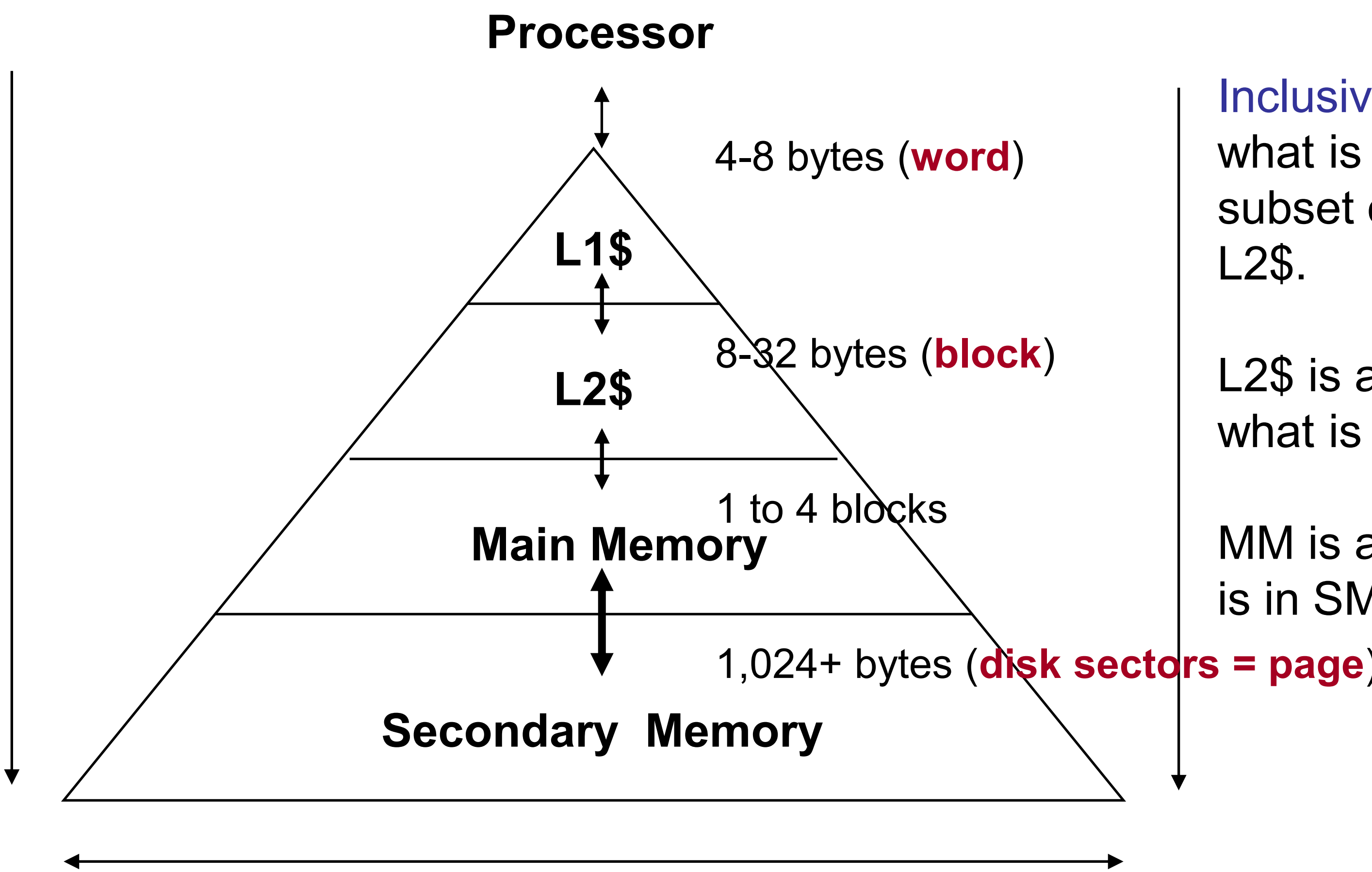
Main memory



Disk

Characteristics of the Memory Hierarchy

Increasing distance from the processor in access time



(Relative) size of the memory at each level

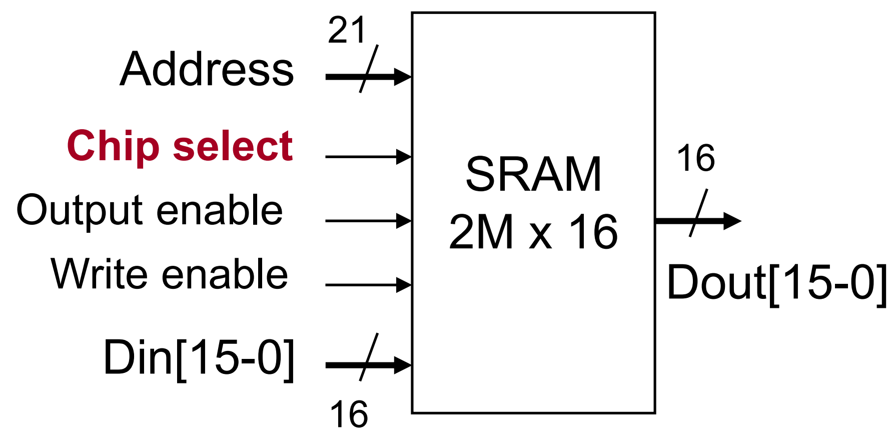
Inclusive (包括)– what is in L1$ is a subset of what is in L2$.

L2$ is a subset of what is in MM.

MM is a subset of is in SM.

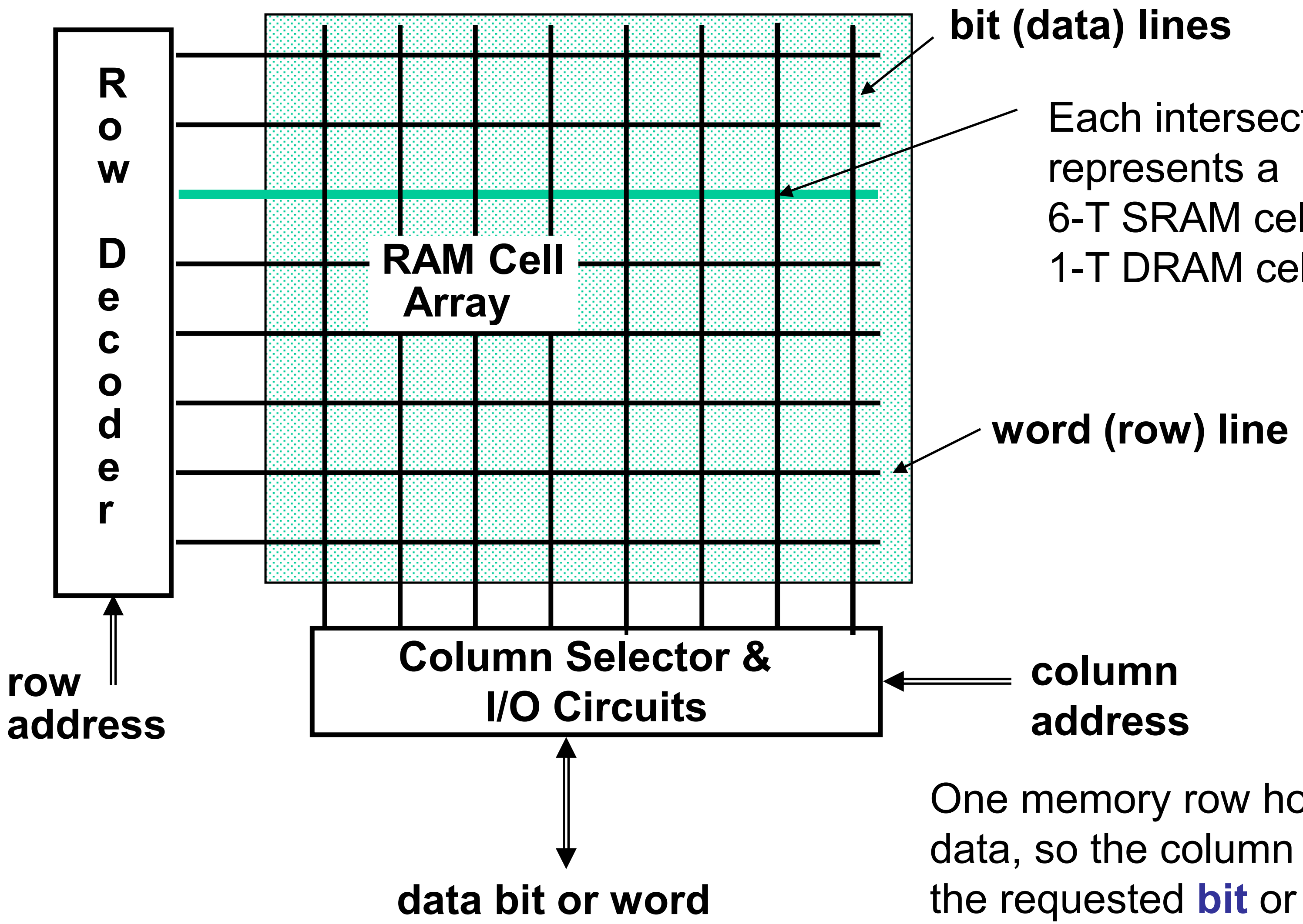
Memory Hierarchy Technologies

- Caches use **SRAM** (static random access memory) for speed and technology compatibility
  - **Low density (6 transistor cells), high power, expensive, fast**
  - **Static: content will last "forever" (until power turned off)**



- Main Memory uses **DRAM** for size (density)
  - **High density** (**1 transistor cells**), low power, cheap, slow
  - **Dynamic**: needs to be "**refreshed**" regularly (~ every 8 ms)
    - 1% to 2% of the active cycles of the DRAM
  - Addresses divided into 2 halves (row and column)
    - RAS or Row Access Strobe triggering row decoder
    - CAS or Column Access Strobe triggering column selector

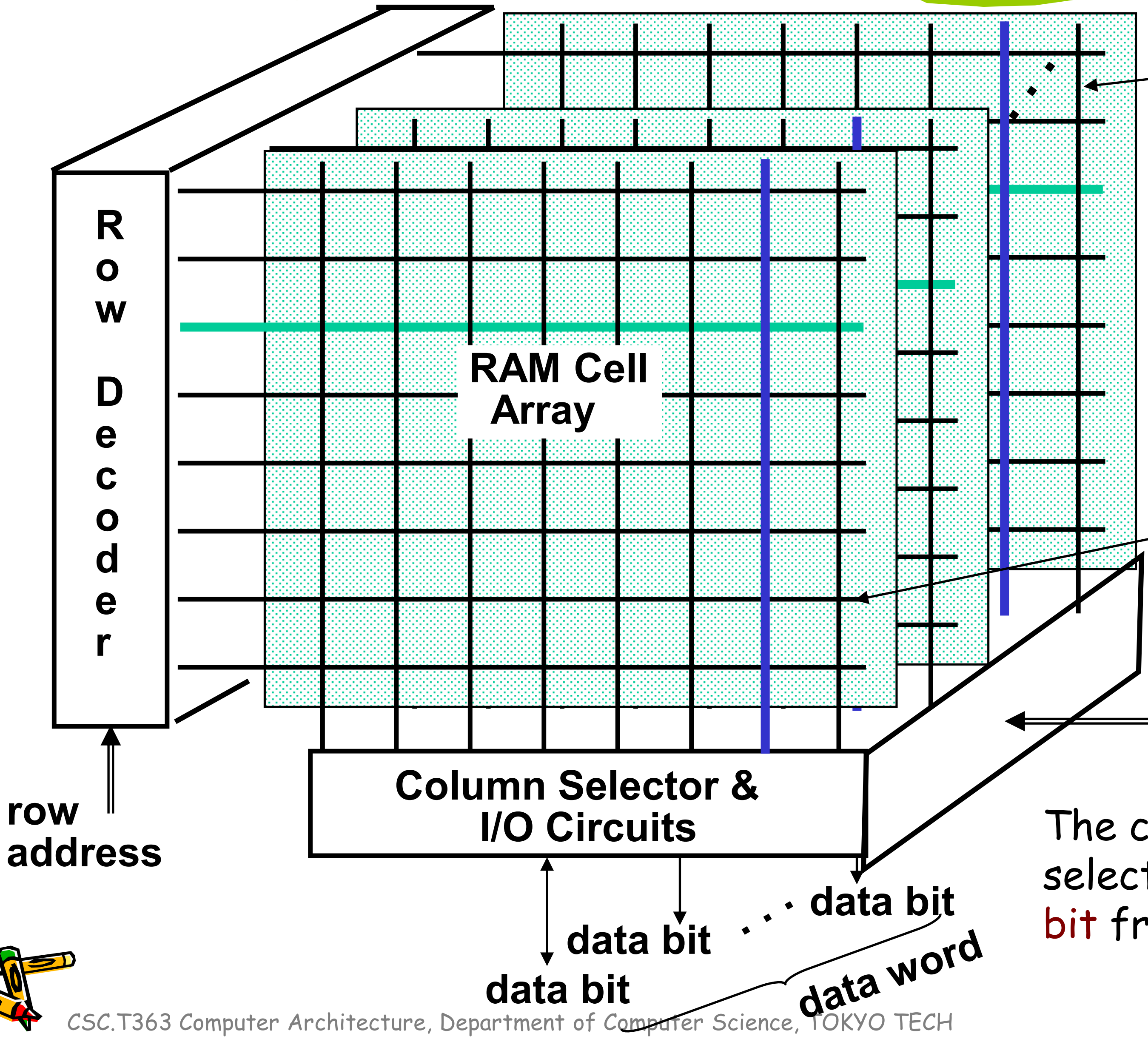
Classical RAM Organization (~Square)



Each intersection represents a 6-T SRAM cell or a 1-T DRAM cell

One memory row holds a block of data, so the column address selects the requested **bit** or **word** from that block

Classical RAM Organization (~Square Planes)



Each intersection represents a 1-T DRAM cell

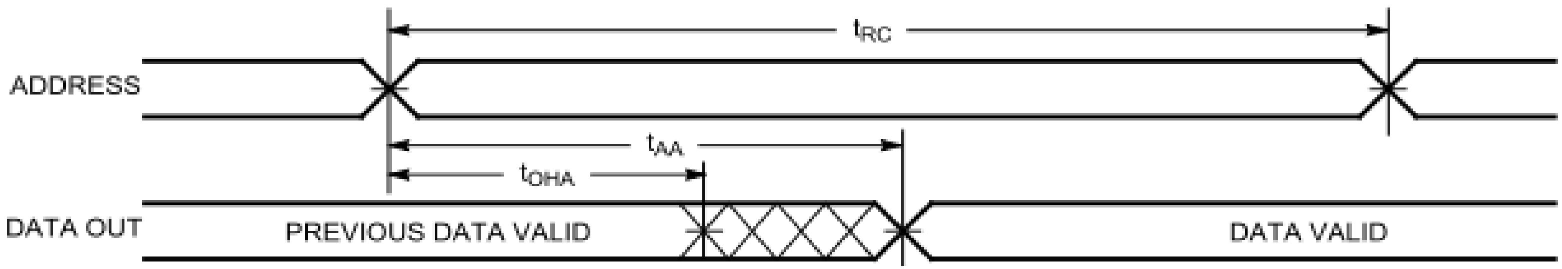
The column address selects the requested bit from the row in each plane

Asynchronous（非同期式）SRAMメモリ

CY7C1049DV33

Switching Waveforms

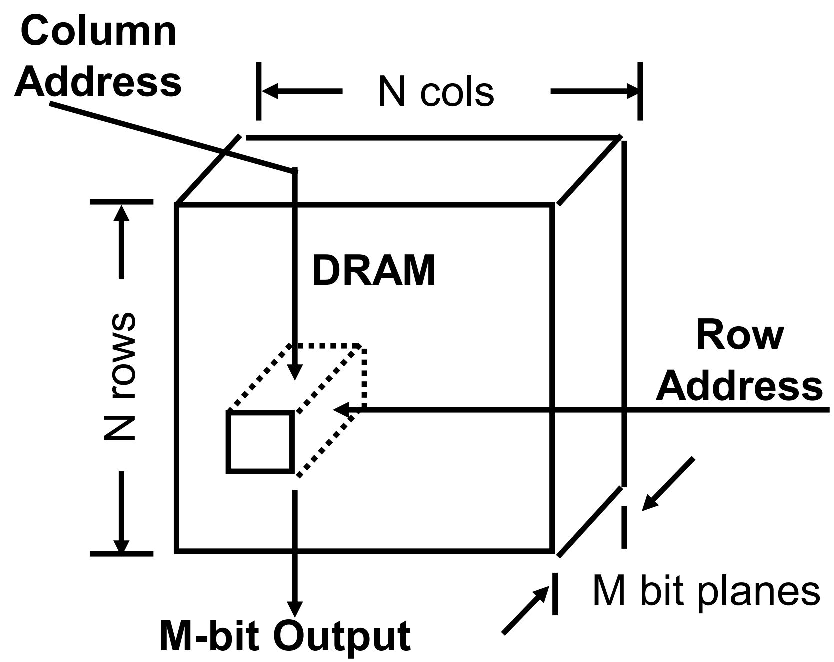
Figure 3. Read Cycle No. 1[13, 14]

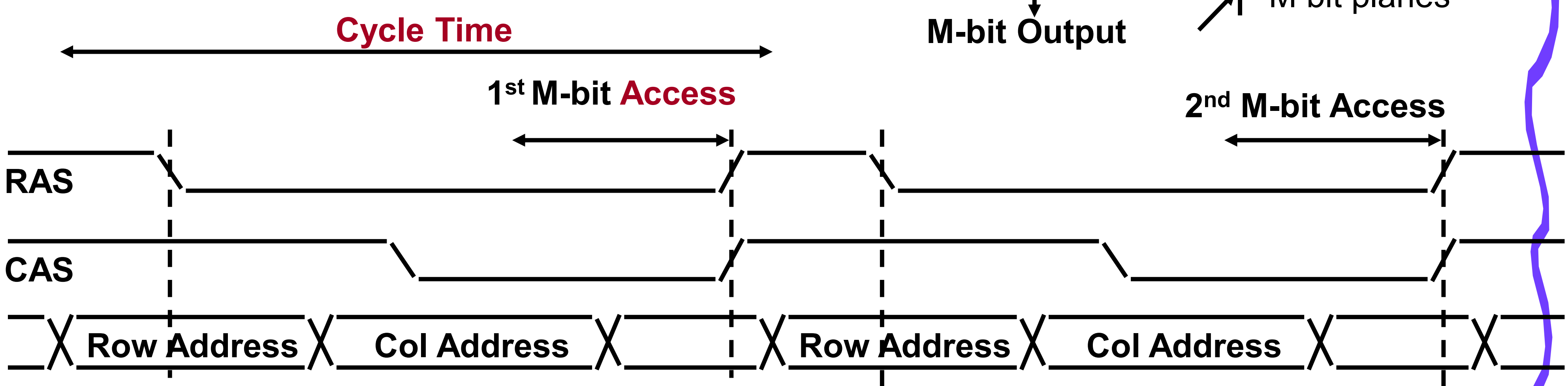


Datasheet

Classical DRAM Operation

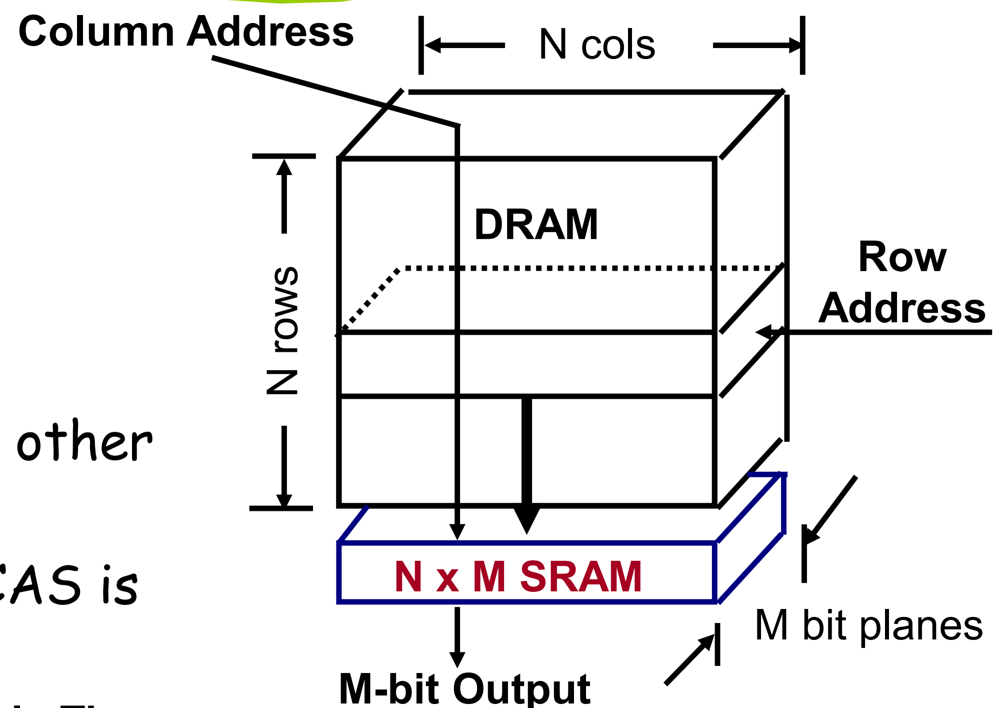
- DRAM Organization:
  - N rows x N column x M-bit
  - Read or Write M-bit at a time
  - Each M-bit access requires a RAS (Row Address Strobe) / CAS (Column Address Strobe) cycle

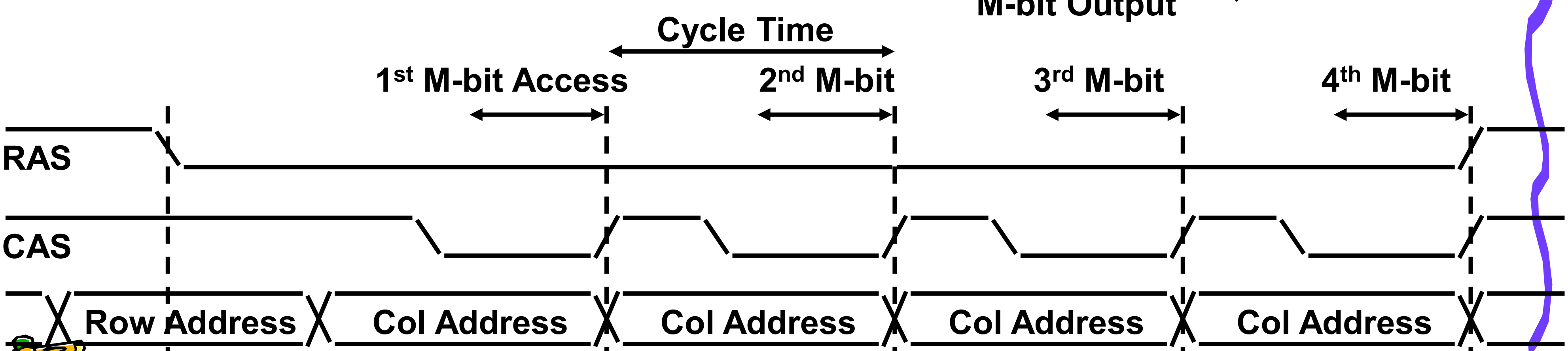




Page Mode DRAM Operation

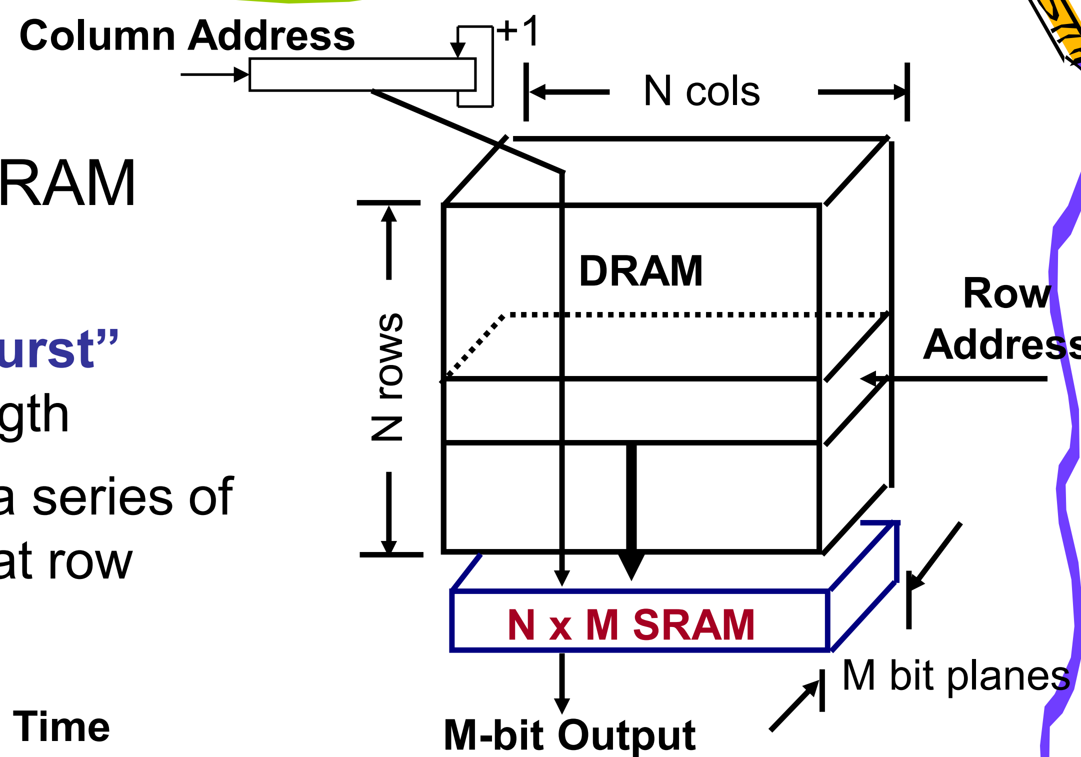
- Page Mode DRAM
  - N x M SRAM to save a row
- After a row is read into the SRAM "register"
  - Only CAS is needed to access other M-bit words on that row
  - RAS remains asserted while CAS is toggled

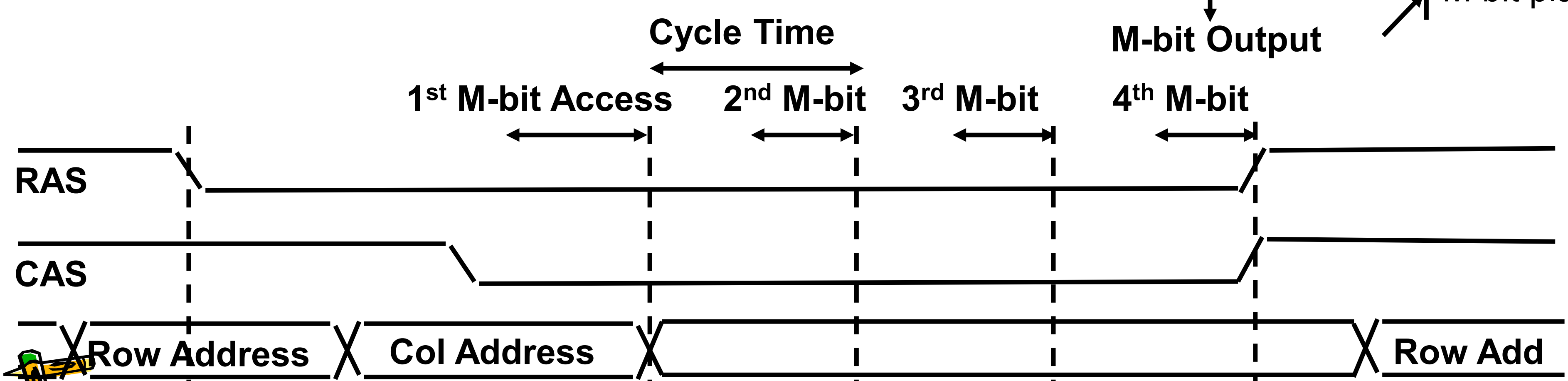




Synchronous DRAM (SDRAM) Operation

- After a **row** is read into the SRAM register
  - Inputs CAS as **the starting "burst" address** along with a burst length
  - Transfers a burst of data from a series of sequential addresses within that row

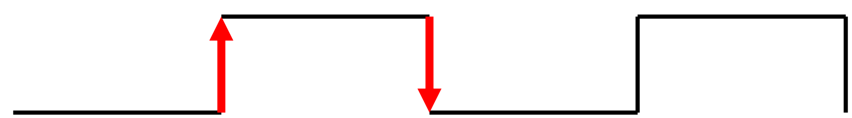




Other DRAM Architectures

- Double Data Rate SDRAMs – **DDR-SDRAMs** (and DDR-SRAMs)
  - Double data rate because they transfer data on both the rising and falling edge of the clock
  - Are the most widely used form of SDRAMs

- **DDR2-SDRAMs**
- **DDR3-SDRAMs**



DRAM Memory Latency & Bandwidth Milestones

	DRAM	Page DRAM	FastPage DRAM	FastPage DRAM	Synch DRAM	DDR SDRAM
Module Width	16b	16b	32b	64b	64b	64b
Year	1980	1983	1986	1993	1997	2000
Mb/chip	0.06	0.25	1	16	64	256
Die size (mm^2)	35	45	70	130	170	204
Pins/chip	16	16	18	20	54	66
BWidth (MB/s)	**13**	**40**	**160**	**267**	**640**	**1600**
Latency (nsec)	**225**	**170**	**125**	**75**	**62**	**52**

Patterson, CACM Vol 47, #10, 2004

In the time that the memory to processor bandwidth doubles the memory latency improves by a factor of only 1.2 to 1.4

To deliver such high bandwidth, the internal DRAM has to be organized as **interleaved memory banks**

DDR4 SDRAM

- 規格：DDR4 デスクトップ用 動作電圧：1.2v JEDEC準拠品（XMP2.0非搭載）
- 速度：PC4-25600 3200Mhz CL値：22-22-22-52 / 容量：32GBx2枚（64GI
- 対応チップセット：Intel：Z590/H570/B560/H510/Z490 ・AMD：

チップ規格	モジュール規格	メモリクロック (MHz)	バスクロック (MHz)	転送速度 (GB/秒)	JEDEC 規格
DDR4-800	PC4-6400	50	400	6.4	
DDR4-1066	PC4-8528	66	533	8.5	
DDR4-1333	PC4-10664	83	666	10.6	
DDR4-1600	PC4-12800	100	800	12.8	○
DDR4-1866	PC4-14900	116	933	14.9	○
DDR4-2133	PC4-17000	133	1066	17.0	○
DDR4-2400	PC4-19200	150	1200	19.2	○
DDR4-2666	PC4-21333	166	1333	21.3	○
DDR4-2800	PC4-22400	175	1400	22.4	
DDR4-2933	PC4-23466	183	1466	23.4	○
DDR4-3000	PC4-24000	188	1500	24.0	
DDR4-3200	PC4-25600	200	1600	25.6	○
DDR4-3300	PC4-26400	206	1650	26.4	

Amazon, Wikipedia

Xilinx 7 Series FPGA Configuration Logic Block (CLB)

7 Series FPGAs
Configurable Logic Block

User Guide

UG474 (v1.8) September 27, 2016

Slices = SLICEL + SLICEM
Distributed RAM (bit) = SLICEM * 256

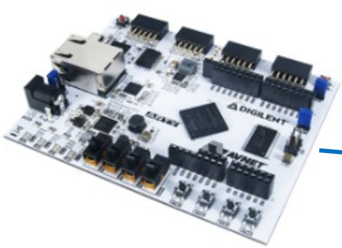


Table 1-2: **Artix-7 FPGA CLB Resources**

Device	Slices[1]	SLICEL	SLICEM	6-input LUTs	Distributed RAM (Kb)	Shift Register (Kb)	Flip-Flops
7A12T	2,000[2]	1,316	684	8,000	171	86	16,000
7A15T	2,600[2]	1,800	800	10,400	200	100	20,800
7A25T	3,650	2,400	1,250	14,600	313	156	29,200
7A35T	5,200[2]	3,600	1,600	20,800	400	200	41,600
7A50T	8,150	5,750	2,400	32,600	600	300	65,200
7A75T	11,800[2]	8,232	3,568	47,200	892	446	94,400
7A100T	15,850	11,100	4,750	63,400	1,188	594	126,800
7A200T	33,650	22,100	11,550	134,600	2,888	1,444	269,200

Xilinx 7 Series Configuration Logic Block (CLB)

SLICEM

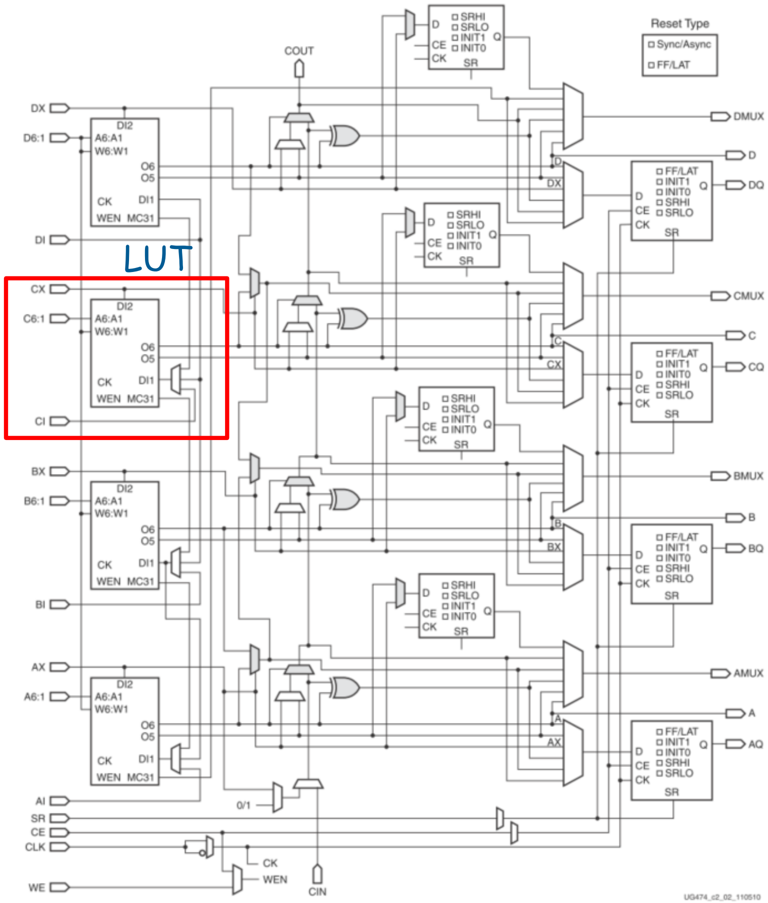


Figure 2-3: **Diagram of SLICEM**

SLICEL

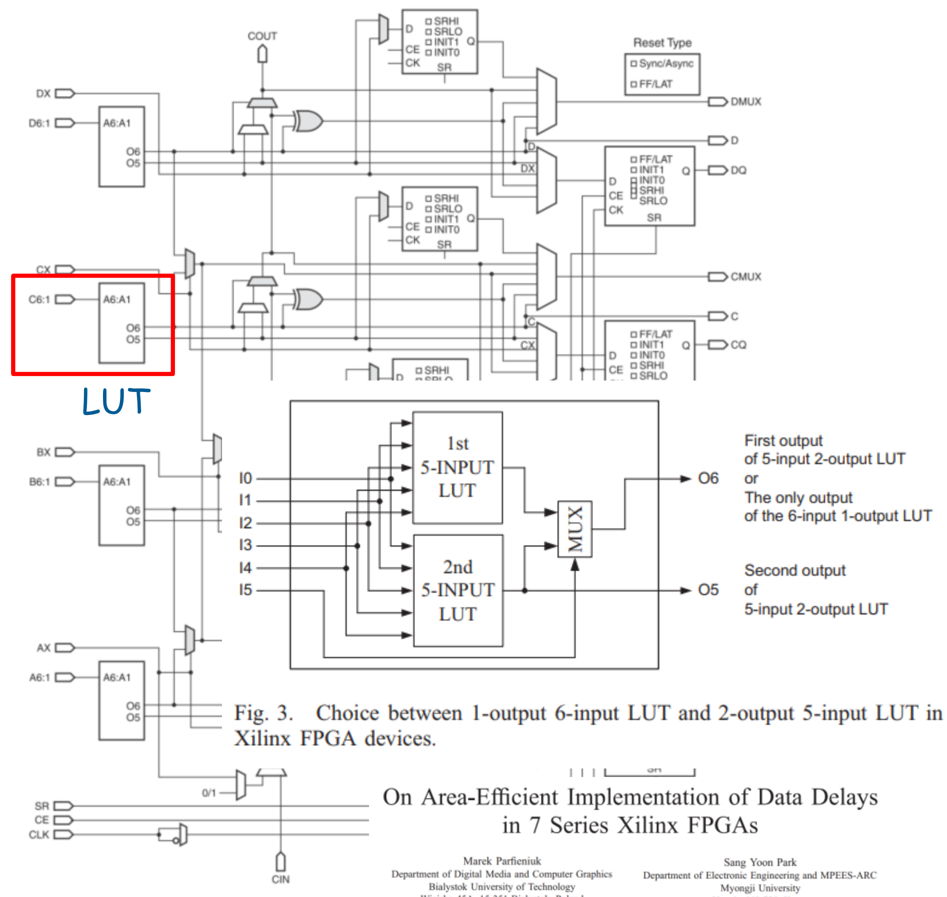


Figure 2-4:

Fig. 3. Choice between 1-output 6-input LUT and 2-output 5-input LUT in Xilinx FPGA devices.

On Area-Efficient Implementation of Data Delays in 7 Series Xilinx FPGAs

Marek Parfieniuk
Department of Digital Media and Computer Graphics
Bialystok University of Technology
Wiejska 45A, 15-351 Bialystok, Poland
Email: m.parfieniuk@pb.edu.pl

Sang Yoon Park
Department of Electronic Engineering and MPEES-ARC
Myongji University
Yongin 449-728, Korea
Email: sypark@mju.ac.kr

Distributed RAM

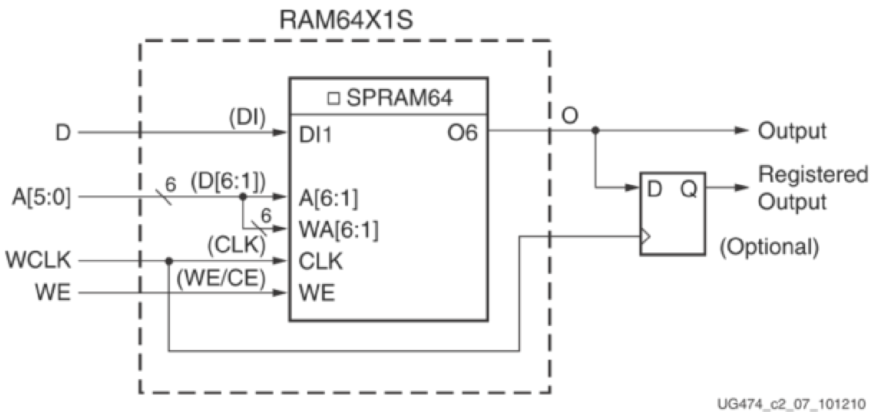


Figure 2-8: **64 X 1 Single Port Distributed RAM (RAM64X1S)**

```
module m_RAM64X1S (
  input  wire       clk,
  input  wire [5:0] a,
  input  wire       d,
  input  wire       we,
  output wire       dout
);

  reg [0:0] mem [0:63];
  assign dout = mem[a];
  always @(posedge clk) if(we) mem[a] <= d;
endmodule
```

LUTRAM = 1

Table 2-3: **Distributed RAM Configuration**

RAM	Description	Primitive	Number of LUTs
32 x 1S	Single port	RAM32X1S	1
32 x 1D	Dual port	RAM32X1D	2
32 x 2Q	Quad port	RAM32M	4
32 x 6SDP	Simple dual port	RAM32M	4
64 x 1S	Single port	RAM64X1S	1
64 x 1D	Dual port	RAM64X1D	2
64 x 1Q	Quad port	RAM64M	4
64 x 3SDP	Simple dual port	RAM64M	4
128 x 1S	Single port	RAM128X1S	2
128 x 1D	Dual port	RAM128X1D	4
256 x 1S	Single port	RAM256X1S	4

- Single port
  - Common address port for synchronous writes and asynchronous reads
    - Read and write addresses share the same address bus

Distributed RAM

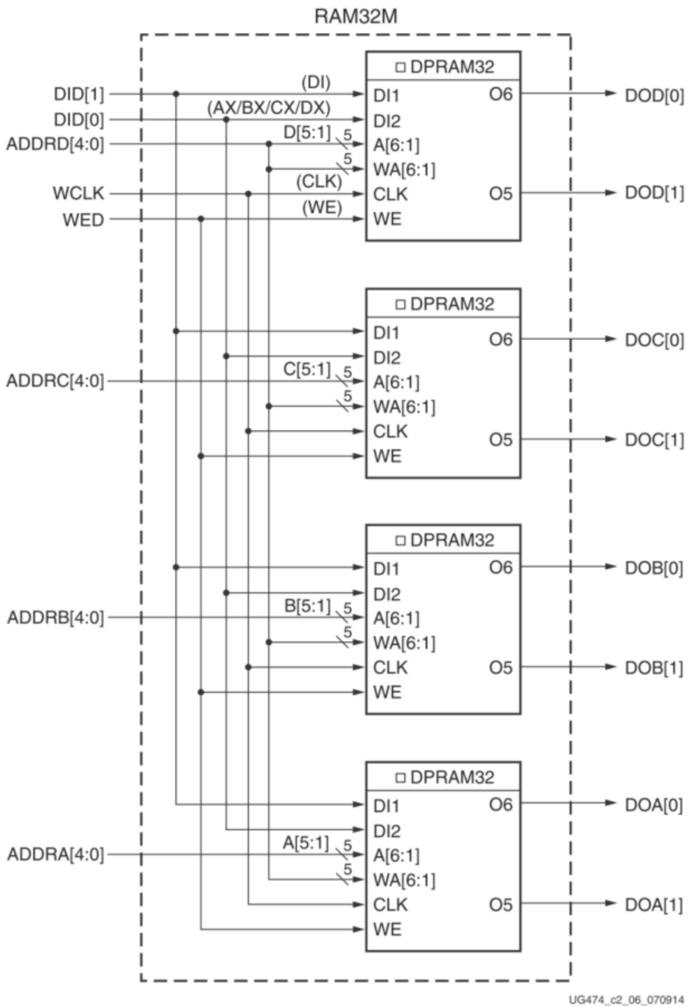


Figure 2-6: **32 X 2 Quad Port Distributed RAM (RAM32M)**

Table 2-3: **Distributed RAM Configuration**

RAM	Description	Primitive	Number of LUTs
32 x 1S	Single port	RAM32X1S	1
32 x 1D	Dual port	RAM32X1D	2
32 x 2Q	Quad port	RAM32M	4
32 x 6SDP	Simple dual port	RAM32M	4
64 x 1S	Single port	RAM64X1S	1
64 x 1D	Dual port	RAM64X1D	2
64 x 1Q	Quad port	RAM64M	4
64 x 3SDP	Simple dual port	RAM64M	4
128 x 1S	Single port	RAM128X1S	2
128 x 1D	Dual port	RAM128X1D	4
256 x 1S	Single port	RAM256X1S	4

- Quad port
  - One port for synchronous writes and asynchronous reads
  - Three ports for asynchronous reads

Distributed RAM

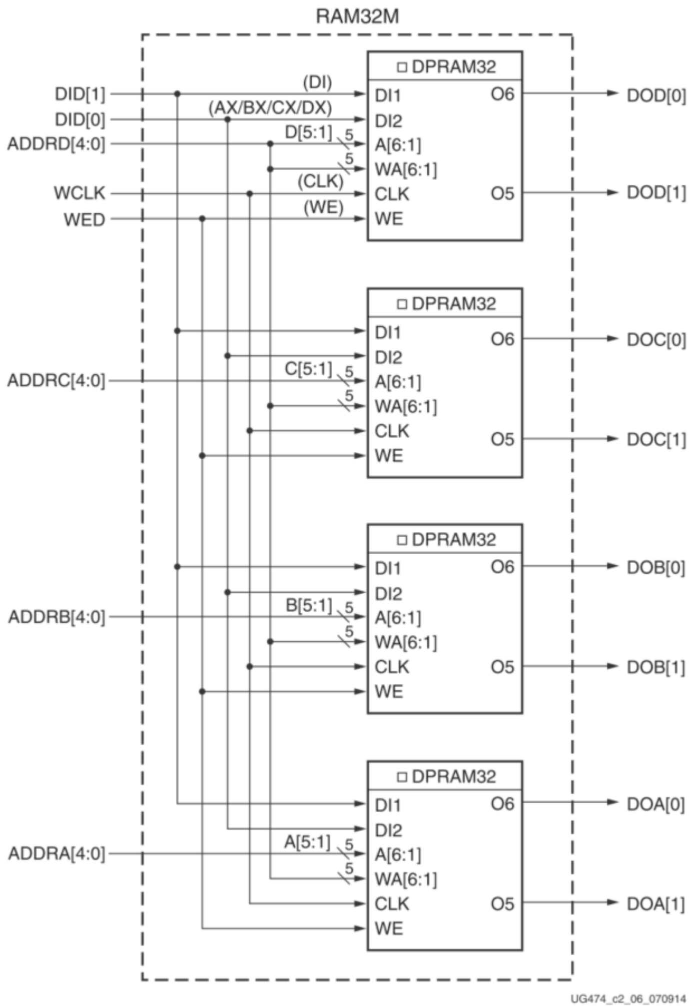


Figure 2-6: **32 X 2 Quad Port Distributed RAM (RAM32M)**

```
module m_RAM32M_Q (
  input wire clk,
  input wire [4:0] a1,
  input wire [4:0] a2,
  input wire [4:0] a3,
  input wire [4:0] a4,
  input wire [1:0] d,
  input wire we,
  output wire [1:0] dout1,
  output wire [1:0] dout2,
  output wire [1:0] dout3,
  output wire [1:0] dout4
);

  reg [1:0] mem [0:31];
  assign dout1 = mem[a1];
  assign dout2 = mem[a2];
  assign dout3 = mem[a3];
  assign dout4 = mem[a4];
  always @(posedge clk) if(we) mem[a1] <= d;
endmodule
```

Failed Routes	LUT	FF	BRAM	URAM	DSP
	4	0	0.0	0	0
0	4	0	0.0	0	0

LUTRAM = 4