ECSE-2610
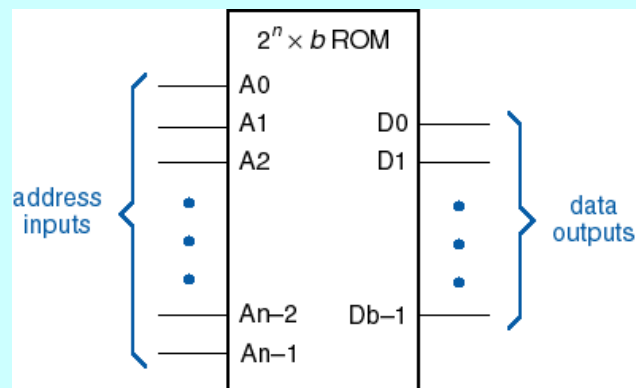# Computer Components & Operations (COCO)

Part 18: Random Access Memory

1

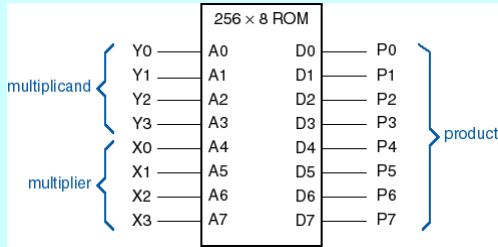## Read-Only Memories

2

Why "ROM"?

- **Program storage**
  - **Boot ROM for personal computers**
  - **Complete application storage for embedded systems.**
- **Actually, a ROM is a combinational circuit, basically a truth-table lookup.**
  - **Can perform any combinational logic function**
  - **Address inputs = function inputs**
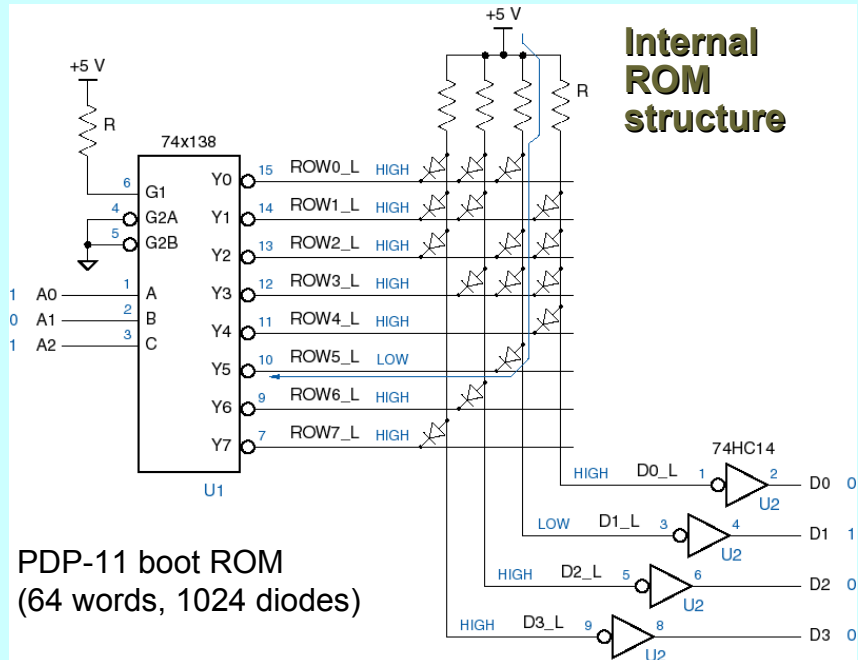 - **Data outputs = function outputs**

3

Logic-in-ROM example

8 × 4 ROM

I0 —	A0	D0	— Y0
I1 —	A1	D1	— Y1
POL —	A2	D2	— Y2
		D3	— Y3

Inputs			Outputs			
A2	A1	A0	D3	D2	D1	D0
0	0	0	1	1	1	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	1	1	1
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

4

4x4 multiplier example

256 × 8 ROM

multiplicand: Y0→A0, Y1→A1, Y2→A2, Y3→A3
multiplier: X0→A4, X1→A5, X2→A6, X3→A7
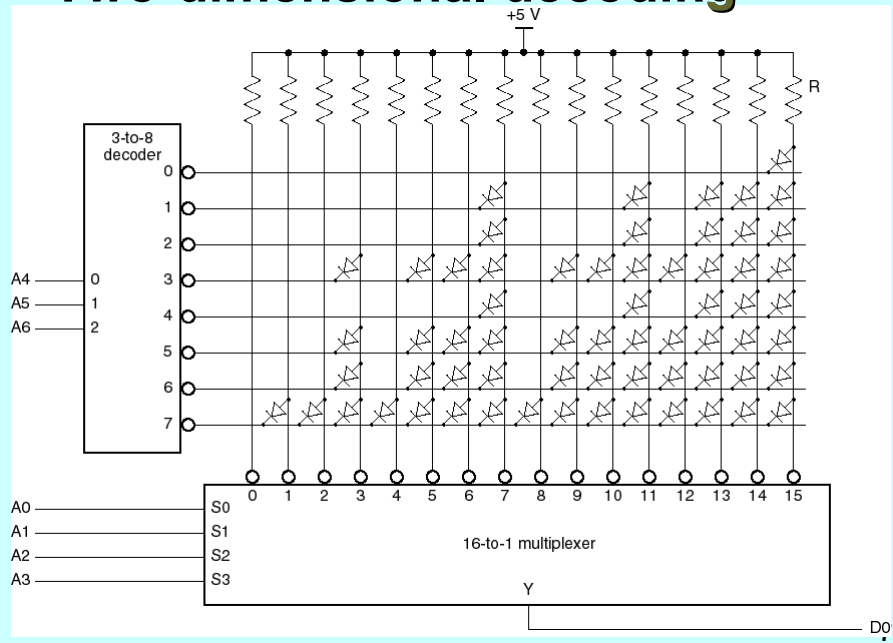product: D0→P0, D1→P1, D2→P2, D3→P3, D4→P4, D5→P5, D6→P6, D7→P7

```
00: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
10: 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
20: 00 02 04 06 08 0A 0C 0E 10 12 14 16 18 1A 1C 1E
30: 00 03 06 09 0C 0F 12 15 18 1B 1E 21 24 27 2A 2D
40: 00 04 08 0C 10 14 18 1C 20 24 28 2C 30 34 38 3C
50: 00 05 0A 0F 14 19 1E 23 28 2D 32 37 3C 41 46 4B
60: 00 06 0C 12 18 1E 24 2A 30 36 3C 42 48 4E 54 5A
70: 00 07 0E 15 1C 23 2A 31 38 3F 46 4D 54 5B 62 69
80: 00 08 10 18 20 28 30 38 40 48 50 58 60 68 70 78
90: 00 09 12 1B 24 2D 36 3F 48 51 5A 63 6C 75 7E 87
A0: 00 0A 14 1E 28 32 3C 46 50 5A 64 6E 78 82 8C 96
B0: 00 0B 16 21 2C 37 42 4D 58 63 6E 79 84 8F 9A A5
C0: 00 0C 18 24 30 3C 48 54 60 6C 78 84 90 9C A8 B4
D0: 00 0D 1A 27 34 41 4E 5B 68 75 82 8F 9C A9 B6 C3
E0: 00 0E 1C 2A 38 46 54 62 70 7E 8C 9A A8 B6 C4 D2
F0: 00 0F 1E 2D 3C 4B 5A 69 78 87 96 A5 B4 C3 D2 E1
```

5

Internal ROM structure

+5 V

74x138 (U1)

6 G1
4 G2A
5 G2B

1 A0 → 1 A
0 A1 → 2 B
1 A2 → 3 C

Y0 15 — ROW0_L HIGH
Y1 14 — ROW1_L HIGH
Y2 13 — ROW2_L HIGH
Y3 12 — ROW3_L HIGH
Y4 11 — ROW4_L HIGH
Y5 10 — ROW5_L LOW
Y6 9 — ROW6_L HIGH
Y7 7 — ROW7_L HIGH

R

74HC14
HIGH D0_L 1 ▷ 2 D0 0 (U2)
LOW D1_L 3 ▷ 4 D1 1 (U2)
HIGH D2_L 5 ▷ 6 D2 0 (U2)
HIGH D3_L 9 ▷ 8 D3 0 (U2)

PDP-11 boot ROM
(64 words, 1024 diodes)

6

Two-dimensional decoding

Today's ROMs

- **256K bytes, 1M byte, or larger**
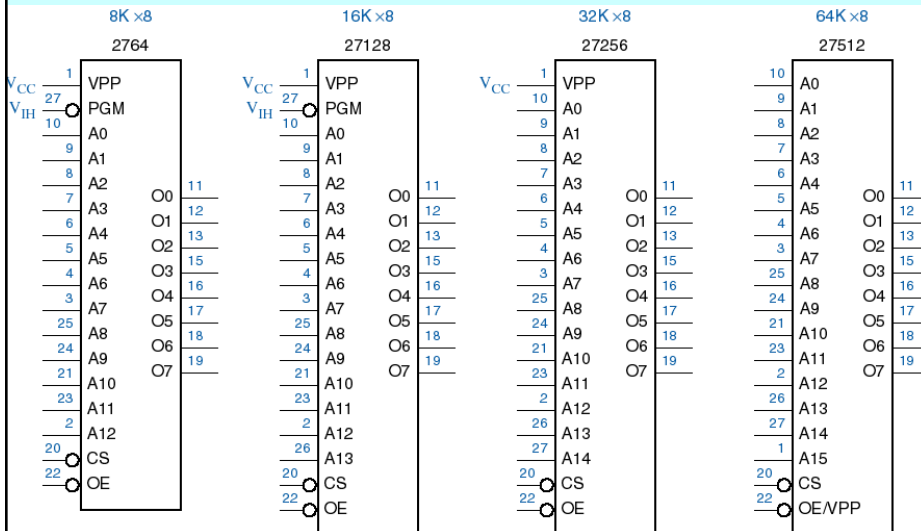- **Use MOS transistors**

8
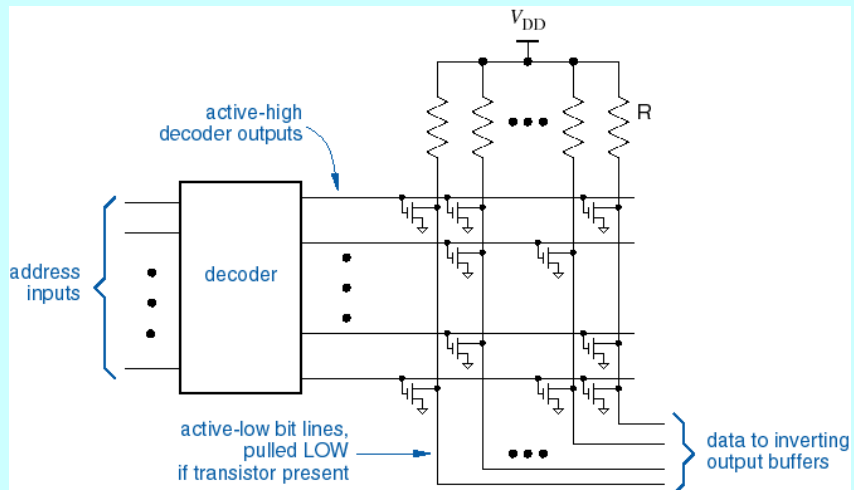
EEPROMs, Flash PROMs
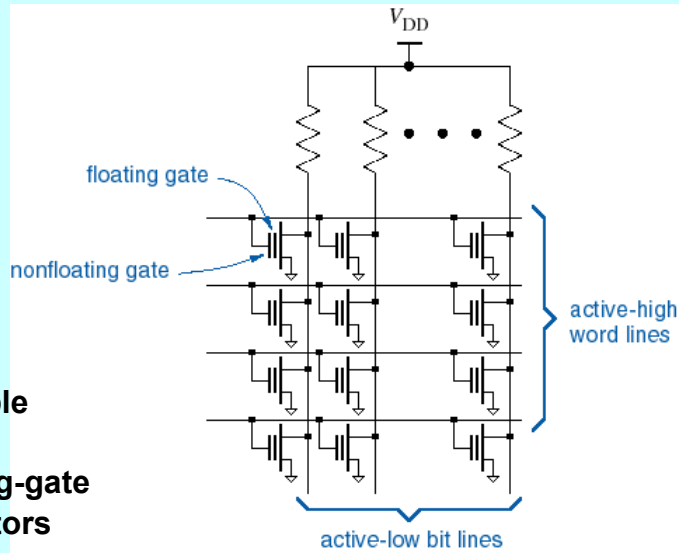
Programmable
and *erasable*
using floating-gate
MOS transistors

9

Typical commercial EEPROMs

10

EEPROM programming
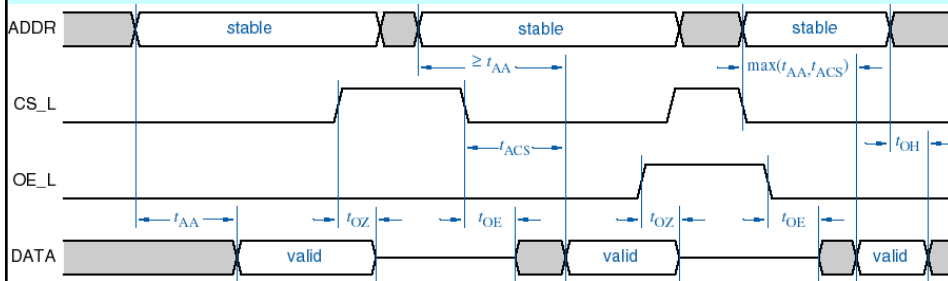
- **Apply a higher voltage to force bit change**
  - E.g., VPP = 12 V
  - On-chip high-voltage "charge pump" in newer chips
- **Erase bits**
  - Byte-byte
  - Entire chip ("flash")
  - One block (typically 32K - 66K bytes) at a time
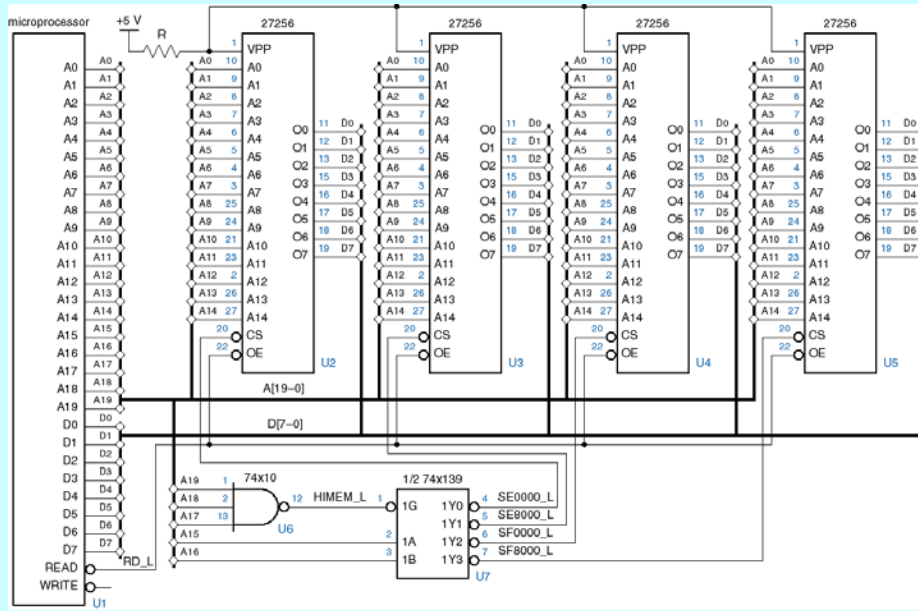- **Programming and erasing are a lot slower than reading (milliseconds vs. 10's of nanoseconds)**

11

ROM control and I/O signals

12
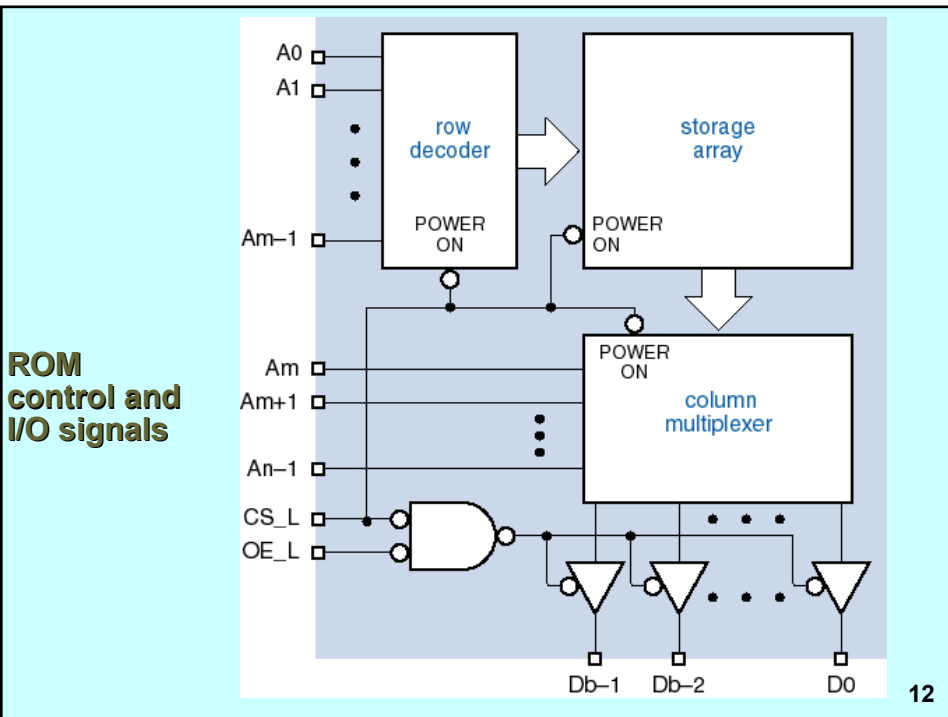
ROM timing

Microprocessor EPROM application

Static RAM Organization

Chip Select Line (active low)

Write Enable Line (active low)

10 Address Lines

1024 x 4 SRAM

CS
WE
A9
A8
A7
A6
A5
A4
A3
A2
A1
A0

IO3
IO2
IO1
IO0
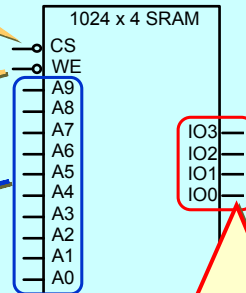
4 Bidirectional Data Lines (read & write)

Data readout is a nondestructive operation

Static RAM

Transistor efficient methods for implementing storage elements

Small RAM: 256 words by 4-bit

Large RAM: 4 million words by 1-bit

We will discuss a 1024 x 4 organization

RAM Organization

Long thin layouts are not the best organization for a RAM

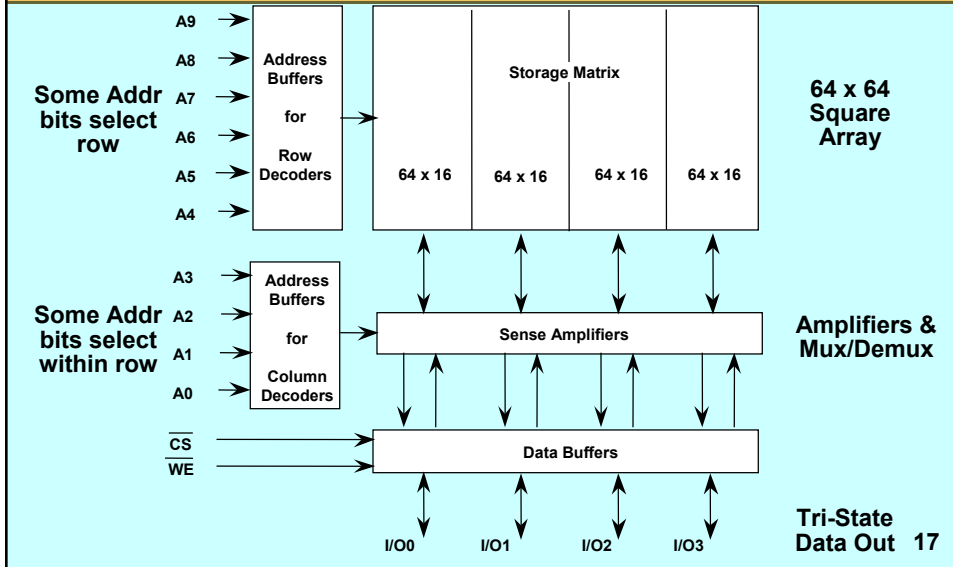|	
A9 →	
A8 →	**Address Buffers**
Some Addr bits select row A7 →	**for**
A6 →	**Row Decoders**
A5 →	
A4 →	

Storage Matrix

| 64 x 16 | 64 x 16 | 64 x 16 | 64 x 16 |

64 x 64 Square Array

A3 →	**Address Buffers**
Some Addr A2 →	**for**
bits select within row A1 →	**Column Decoders**
A0 →	

Sense Amplifiers

Amplifiers & Mux/Demux

$\overline{CS}$ →
$\overline{WE}$ →

Data Buffers

I/O0   I/O1   I/O2   I/O3

Tri-State Data Out   17

Timing Diagram Conventions

	Input Signal	Output Signal
	Must be steady high or low	**Will be steady high or low**
	High-to-low changes permitted	**Will be changing from high-to-low during designated interval**
	Low-to high changes permitted	**Will be changing from low-to-high during designated interval**
	Don't-Care	**State changing**
	(Does not apply)	**Centerline represents high impedance (off) state**

18

RAM Timing

Simplified Read Timing

WE

CS

Address — Valid Address

Access Time

The time it takes for new data to be ready to appear at the output

Access Time

Data Out — Data Out
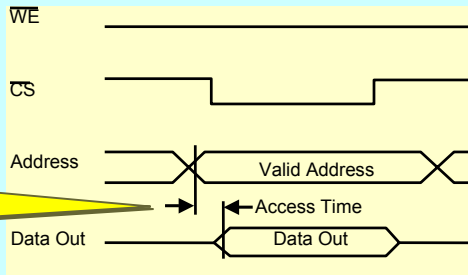
Simplified Write Timing

WE

CS

Memory Cycle Time

Address — Valid Address

Data In — Input Data

19
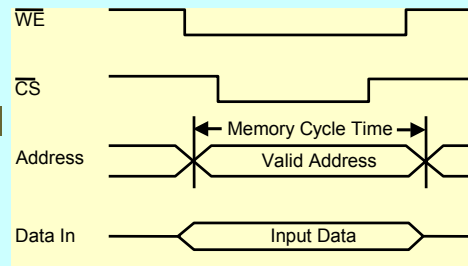
Dynamic RAMs (DRAMs)
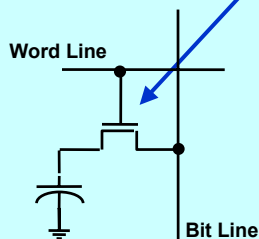
Single Transistor (and a capacitor) memory element

Read: assert **Word Line**, sense **Bit Line**

Write: assert **Word Line**, drive **Bit Line**

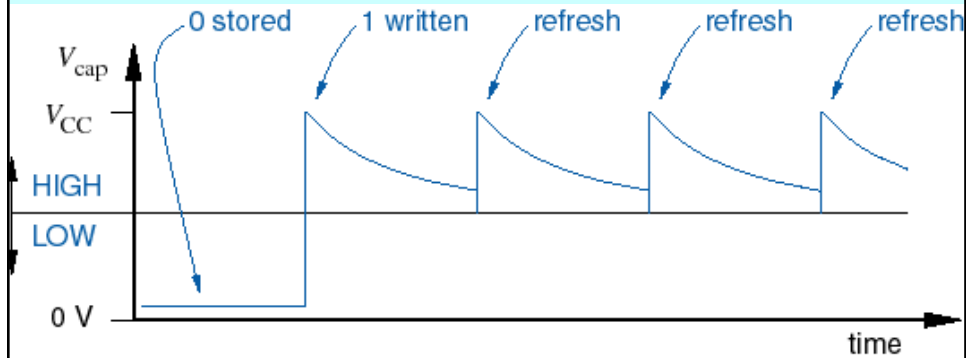**Destructive Read-Out**

Refresh Cycles needed: storage decays in **ms**

　　Internal circuits read word and write back

　　Higher density and cheaper than SRAM

　　Longer access time than SRAM

Word Line

Bit Line

20

DRAM charge leakage

DRAM Refresh

Refresh Frequency

4096 word RAM -- refresh each word once every 4 ms

Assume 120 ns memory access cycle

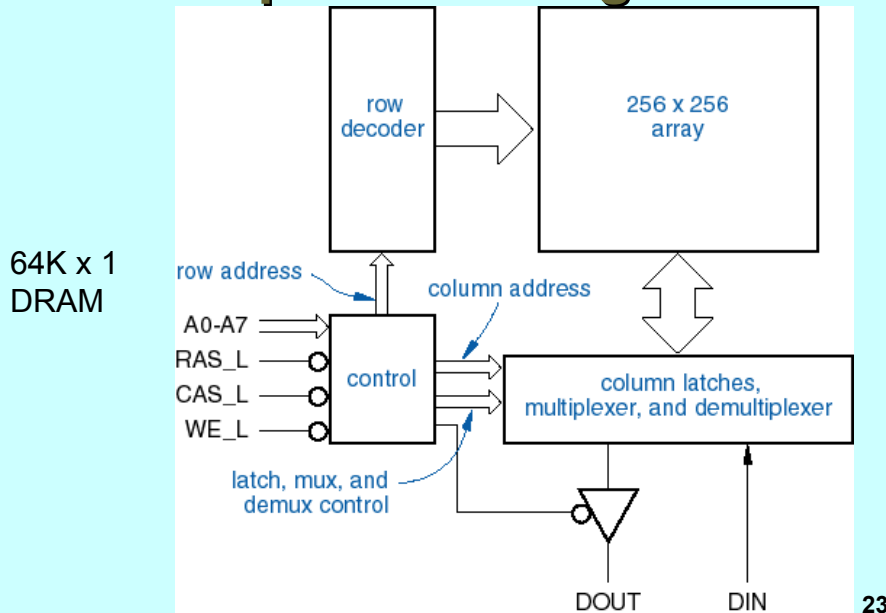This is one refresh cycle every 976 ns (**1 in 8 DRAM accesses**)!

But this RAM is really organized into 64 rows

This is one refresh cycle every 62.5 µs (**1 in 500 DRAM accesses**)

Large capacity DRAMs have 256 rows, **refresh once every 16 µs**

DRAM-chip internal organization

64K x 1
DRAM

row decoder

256 x 256 array

A0-A7
RAS_L
CAS_L
WE_L

control

row address
column address

column latches, multiplexer, and demultiplexer

latch, mux, and demux control
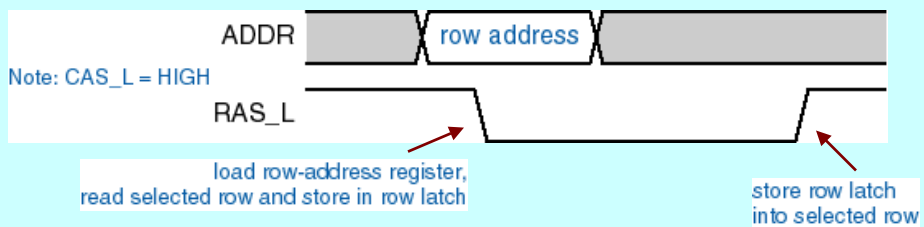
DOUT   DIN

RAS/CAS operation

- **Row Address Strobe, Column Address Strobe**
  - *n* **address bits are provided in two steps using** *n*/2 **pins, referenced to the falling edges of RAS_L and CAS_L**
  - **Traditional method of DRAM operation for 20 years.**
  - **Now being supplanted by synchronous, clocked interfaces in SDRAM (synchronous DRAM).**

DRAM read timing

Note: WE_L = HIGH

ADDR: row address | column address

RAS_L
- load row-address register, read selected row and store in row latch
- store row latch into selected row

CAS_L
- load column-address register, output-enable DOUT, drive with selected bit
- output-disable DOUT

DOUT: valid

25

DRAM refresh timing

Note: CAS_L = HIGH

ADDR: row address

RAS_L
- load row-address register, read selected row and store in row latch
- store row latch into selected row

26

DRAM write timing

ADDR	row address ╳ column address
RAS_L	

load row-address register,
read selected row and store in row latch

store row latch
into selected row

WE_L	

Note: DOUT = Hi-Z

DIN	valid
CAS_L	

load column-address register,
merge DIN into selected column of row latch

27