CSE 2317 | Week 5 | Class 2 | Sunday · 11:00 AM – 12:20 PM | 80 min

Memory systems and PLAs

A memory is a decoder, a grid and something at every crosspoint: a diode, a latch, a capacitor or a floating gate. Program the grid instead of storing data in it and you get a PLA.

---

01 COLD OPEN 0–4 min

One byte out of 137 billion

Your phone has 128 GB of storage: about **137 billion bytes**.

To read one particular byte, the processor has to name it. How many wires (address bits) does it need? And what, physically, is a stored 1?

> 128 GB = 2^{37} bytes, so **37 address bits** name every byte. Each extra wire doubles the memory. A stored 1 is, depending on the memory: a diode that is there (ROM), a latch that is set (SRAM), a capacitor that is charged (DRAM) or electrons trapped on a floating gate (flash).

POP CULTURE

Today's four cells: the **diode** from Week 1, the **latch** from Week 4, a **capacitor**, and a **MOSFET** from Week 2 with a trapped charge.

---

02 CONCEPT 4–9 min

Every memory: decoder, word lines, bit lines

- 2^k words × n bits. k address bits go into a **decoder** that raises exactly one **word line**.
- The cells on that line put their bits on the n **bit lines** (read) or take them from there (write).
- Control: **CS̄** (chip select), **OĒ** (output enable, drives the data bus), **WĒ** or R/W̄.

$$\text{capacity} = 2^k \times n \text{ bits} \qquad 64\text{K} \times 8 : \; k = 16,\; n = 8,\; 512\ \text{Kbit}$$

TYPE	CELL	VOLATILE	WRITE
Mask ROM	diode / transistor present or not	no	at the factory
PROM	fuse	no	once
EPROM	floating gate	no	UV erase

TYPE	CELL	VOLATILE	WRITE
EEPROM / Flash	floating gate	no	electrically
SRAM	6-transistor latch	yes	any time, fast
DRAM	1 transistor + 1 capacitor	yes	any time, refresh

03 WORKED EXAMPLE 9–14 min

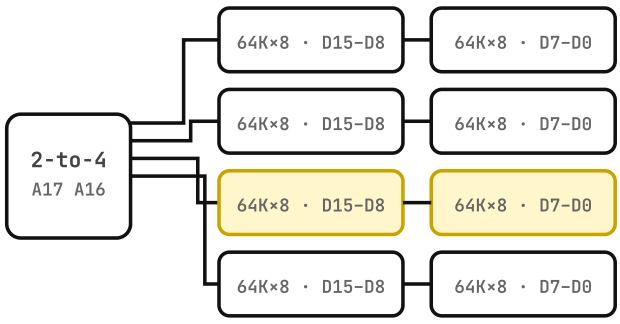
Worked example: building 256K × 16 from 64K × 8 chips

GIVEN

64K × 8 memory chips with active-low $\overline{CS}$. Any decoders.

FIND

How many chips, how the address lines are connected, and which addresses row 2 answers.



Method: memory expansion: width and depth (Mano)

SOLUTION

1. **Width**: 16 bits from 8-bit chips → 2 chips side by side, sharing address and control, one giving D15–D8, the other D7–D0.
2. **Depth**: 256K words from 64K-word chips → 4 rows of chips.

$$\text{chips} = \frac{256\text{K}}{64\text{K}} \times \frac{16}{8} = 4 \times 2 = 8$$

3. Address lines: 256K = 2^{18} → **A17–A0**. Each chip takes A15–A0 (2^{16} = 64K). The two top lines A17A16 go into a **2-to-4 decoder** whose outputs drive the $\overline{CS}$ of row 0 … row 3.
4. Row r answers for addresses r × 64K to (r + 1) × 64K − 1, e.g. row 2: 20000h–2FFFFh.

> Width expansion needs no logic at all; depth expansion needs a decoder on the high address bits.

ANSWER

8 chips in 4 rows of 2. A15–A0 to every chip; A17A16 → 2-to-4 decoder → $C\bar{S}$ of each row. Row 2: 20000h–2FFFFh.

04 CONCEPT 14–19 min

ROM: a grid of diodes

- Word lines run across, bit lines run down; each bit line has a **pull-down resistor**.
- The decoder drives the selected word line HIGH (all others LOW). Where a **diode** joins that word line to a bit line, it conducts: the bit line rises to ≈ V_H − 0.7 V, a 1. No diode: the resistor holds it at 0.
- Each bit line is a **diode OR** (Week 1!) of the word lines that have a diode on it. A ROM is a decoder (all minterms) followed by OR gates.

✦ PROGRAMMING = CHOOSING THE DIODES

Mask ROM: the factory leaves diodes (today: transistors) in or out. **PROM:** every crosspoint has a diode in series with a fuse; you blow the fuses of the 0s. **EPROM / flash:** the switch is a MOSFET whose threshold is set by charge on a floating gate.

! GOTCHA

Diodes are essential: a plain wire at a crosspoint would short the selected word line to every other word line through the bit lines.

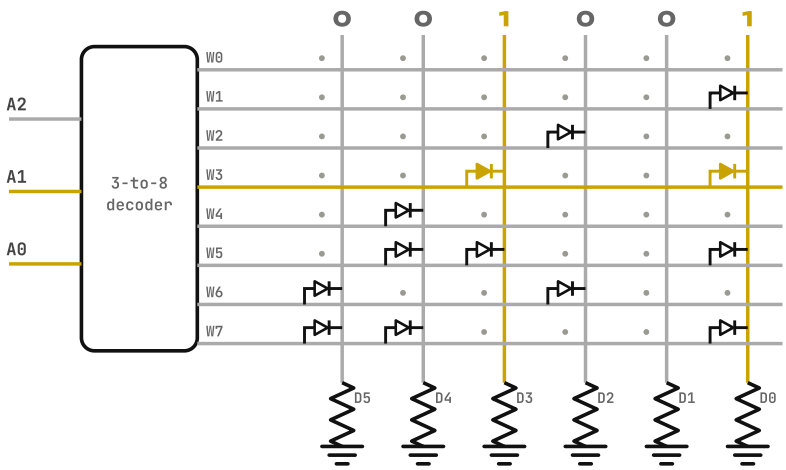
05 LIVE LAB 19–25 min

Live lab: read and

program a diode ROM

A diode-matrix ROM Pick an address: the decoder raises one word line, and each diode on it pulls its bit line to 1. Click any crosspoint to add or remove a diode.

Address



ADDRESS
011 = 3

DATA D5…D0
001001 = 9

A	D5…D0	
000	000000	0
001	000001	1
010	000100	4
011	**001001**	**9**
100	010000	16
101	011001	25
110	100100	36
111	110001	49

13 diodes = 13 stored 1s out of 48 bits.

06 WORKED EXAMPLE 25–29 min

Worked example: a full adder in an 8 × 2 ROM

GIVEN

A ROM with 3 address inputs and 2 outputs (8 words × 2 bits), diode matrix.

FIND

Its contents for a full adder (S, C_{out}) and the number of diodes.

Method: combinational logic with ROM (Mano)

SOLUTION

1. Address = A B C_{in} (3 bits → 8 words). Data = S and C_{out} (2 bits). The ROM stores the truth table directly; no simplification needed.

2. Write each row: S = A ⊕ B ⊕ C_{in}, C_{out} = majority(A, B, C_{in}):

ABC_{in}	000	001	010	011	100	101	110	111
S	0	1	1	0	1	0	0	1
C_{out}	0	0	0	1	0	1	1	1

3. Every 1 is one diode from that word line to that bit line: 4 for S + 4 for C_{out} = **8 diodes**.

> A ROM implements any function of its address bits: it is a truth table in silicon. The cost is 2^n words, used or not.

ANSWER

S = 0 1 1 0 1 0 0 1, C_{out} = 0 0 0 1 0 1 1 1 for addresses 000 ... 111. 8 diodes.

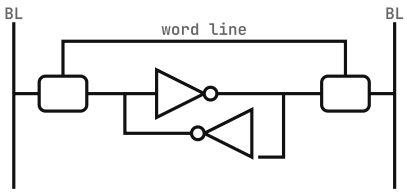
---

07 CONCEPT 29–35 min

RAM: a latch or a capacitor at every crosspoint

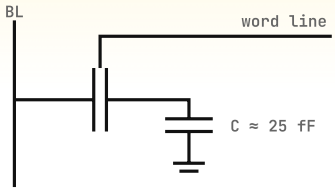
SRAM: 6 TRANSISTORS

Two cross-coupled CMOS inverters (the bistable latch of Week 4) hold the bit; two nMOS access transistors connect it to BL and $\overline{BL}$ when the word line is high. Holds as long as power is on, no refresh, fast (cache memory).



DRAM: 1 TRANSISTOR + 1 CAPACITOR

A charged capacitor (≈ 25 fF) is a 1. Reading connects it to the bit line, whose sense amplifier sees a tiny voltage change: the read is **destructive**, so the row is written back. Leakage drains it, so every row is **refreshed** every 64 ms. Tiny cell: main memory.

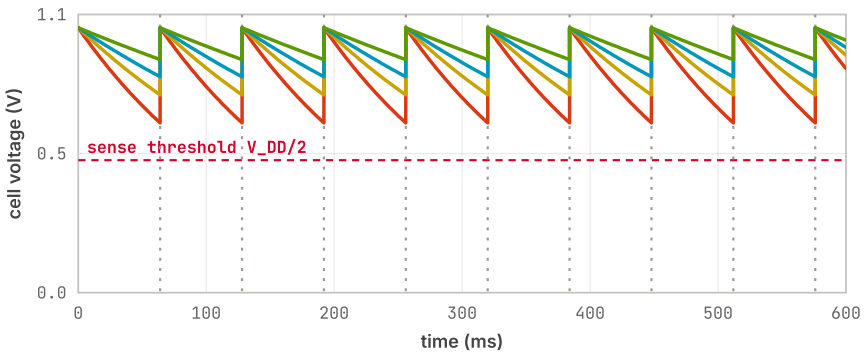


08 LIVE LAB 35–40 min

Live lab: refresh, or lose the bit

DRAM: a bit that leaks away Each cell is a tiny capacitor holding charge for a 1. It leaks. Refresh reads every row and writes it back before any cell drops below the sense threshold (V_DD/2).

Refresh every **64 ms**



BITS LOST
0 of 4

WEAKEST CELL HOLDS
100 ms

REFRESH OVERHEAD
0.64 %

Overhead = 8192 rows × 50 ns ÷ 64 ms: the fraction of time the memory is busy refreshing instead of answering the processor.

09 WORKED EXAMPLE 40–44 min

Worked example: how often must DRAM refresh?

GIVEN

Cell capacitance C = 25 fF, written to 1.2 V, sensed correctly while above 0.6 V. Worst-case leakage I = 0.1 pA. The chip has 8192 rows; refreshing one row takes 50 ns.

FIND

The retention time, the margin over the 64 ms standard, and the refresh overhead.

Method: DRAM retention and refresh (Taub & Schilling)

SOLUTION

1. The cell is read correctly while V > $V_{DD}/2$. Allowed drop: 1.2 → 0.6 V, ΔV = 0.6 V. A constant leakage current I discharges C linearly:

$$t_{ret} = \frac{C\,\Delta V}{I} = \frac{(25 \times 10^{-15})(0.6)}{0.1 \times 10^{-12}} = 0.15\ \text{s} = 150\ \text{ms}$$

2. The 64 ms standard leaves a margin of 150/64 ≈ 2.3 for weak cells and heat (leakage roughly doubles every 10–15 °C).

3. All 8192 rows in 64 ms → one row every 7.8 µs. Each takes 50 ns:

$$\text{overhead} = \frac{8192 \times 50\ \text{ns}}{64\ \text{ms}} = 0.64\,\%$$

ANSWER

t_{ret} = 150 ms (margin ≈ 2.3 over 64 ms). One row every 7.8 µs; overhead 0.64 %.

---

10 YOUR TURN 44–49 min

Your turn · 1: 32K × 8 from 8K × 4

1 EXAM CQ

Build a **32K × 8** memory from **8K × 4** chips.

(a) How many chips? (b) How many address lines, and how are they connected? (c) Which row of chips answers address **5A3Ch**?

SOLUTION

1. Chips:

$$\frac{32\text{K}}{8\text{K}} \times \frac{8}{4} = 4 \times 2 = 8\ \text{chips}$$

2. 32K = 2^{15}: **A14–A0**. A12–A0 go to every chip; A14A13 drive a **2-to-4 decoder** selecting one of 4 rows (each row = 2 chips for 8 bits).

3. Address 5A3Ch:

$$5\text{A3C}_h = 0\,\underline{10}1\ 1010\ 0011\ 1100_2 \ \Rightarrow\ A_{14}A_{13} = 10$$

4. Decoder output 2: **row 2**, which covers 4000h–5FFFh.

ANSWER

8 chips (4 rows × 2), 15 address lines, A14A13 → 2-to-4 decoder. Address 5A3Ch is in row 2 (4000h–5FFFh).

---

11 CONCEPT 49–54 min

PLA: program the logic, not the data

- A ROM generates **every** minterm (2^n of them) and ORs the ones you want. Most functions need only a few product terms.
- A **PLA** has a programmable **AND array** (make only the product terms you need, from inputs and their complements) and a programmable **OR array** (sum them into each output). Shared terms are made once.
- Size: n inputs, k product terms, m outputs:

$$\text{crosspoints} = 2n \cdot k\ (\text{AND}) + k \cdot m\ (\text{OR})$$

DEVICE	AND ARRAY	OR ARRAY
PROM	fixed (full decoder)	programmable
PLA	programmable	programmable
PAL / GAL	programmable	fixed

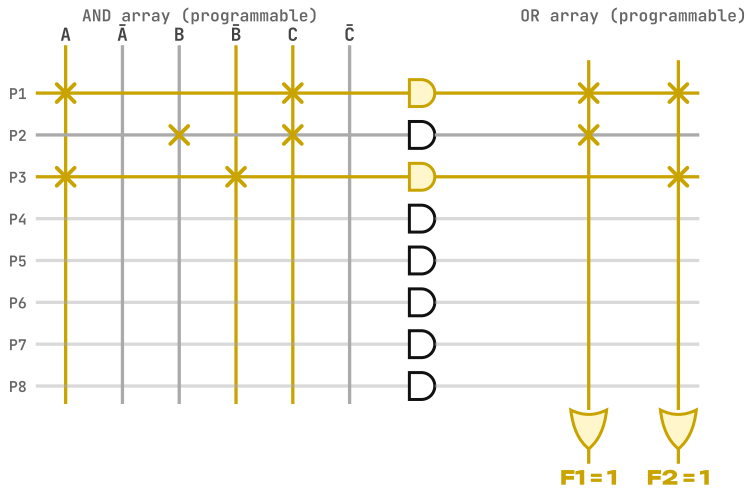
PRO TIP

Write the functions as **minimised sums of products** first, and look for product terms the outputs can share.

12 LIVE LAB 54–60 min

Live lab: program a PLA

Programmable logic array Click a crosspoint in the AND array (left) to put that literal into the product term; click one in the OR array (right) to add the product to an output. Set A, B, C and watch the lines.



$F_1 = AC + BC$

$F_2 = AC + A\bar{B}$

ABC	F_1	F_2
000	0	0
001	0	0
010	0	0
011	1	0
100	0	1
101	**1**	**1**
110	0	0
111	1	1

3 of 8 product terms used · 10 connections.

13 WORKED EXAMPLE 60–64 min

Worked example: a PLA programming table

GIVEN

$F_1(A, B, C) = \Sigma m(3, 5, 7)$ and $F_2(A, B, C) = \Sigma m(4, 5, 7)$.

FIND

The minimised functions and the PLA programming table with the fewest product terms.

Method: PLA programming table (Mano)

SOLUTION

1. Minimise each function: $F_1 = \Sigma m(3, 5, 7)$ and $F_2 = \Sigma m(4, 5, 7)$.

$$F_1 = AC + BC, \qquad F_2 = A\bar{B} + AC$$

2. **AC appears in both**: a PLA generates it once in the AND array and connects it to both outputs in the OR array. 3 product terms, not 4.

3. Programming table: 1 = true literal, 0 = complement, – = not connected; in the output columns 1 = connected.

	A	B	C	F_1	F_2
AC	1	–	1	1	1
BC	–	1	1	1	–
$A\bar{B}$	1	0	–	–	1

4. Load it into the lab ("Example") and check all 8 rows of the truth table.

ANSWER

F_1 = AC + BC, F_2 = A$\bar{B}$ + AC. Three product terms (AC shared): 6 AND connections + 4 OR connections.

14 YOUR TURN 64–68 min

Your turn · 2: PLA or ROM?

2 EXAM CQ

The classic 82S100 PLA has **16 inputs, 48 product terms and 8 outputs**.

(a) How many programmable crosspoints does it have? (b) How many bits would a ROM need to implement any 8 functions of the same 16 inputs? (c) When would you choose each?

SOLUTION

1. PLA with n = 16, k = 48, m = 8:

$$2(16)(48) + (48)(8) = 1536 + 384 = 1920 \text{ crosspoints}$$

2. ROM with 16 inputs and 8 outputs:

$$2^{16} \times 8 = 524{,}288 \text{ bits}$$

3. The PLA is about **270×** smaller, *if* the 8 functions fit in 48 shared product terms. The ROM works for *any* 8 functions of 16 inputs.

ANSWER

PLA: 1,920 crosspoints; ROM: 524,288 bits. PLA when there are many inputs and few product terms; ROM when the function is arbitrary or a table (sine, character fonts).

15 THINK DEEPER 68–71 min

Critical thinking · the

seven-segment decoder

THINK DEEPER

A BCD to 7-segment decoder: 4 inputs, 7 outputs. ROM or PLA?

ROM. With 4 inputs the full table is only 112 bits, it needs no minimisation, and changing the font (say, a different "7" or "9") is just new contents. The PLA wins only when 2^n gets large.

A STRONG ANSWER MENTIONS

- ROM: $2^4 \times 7 = 112$ bits, write the table straight in.
- PLA: minimise 7 functions (with don't-cares for 10–15) and count shared terms: about 9–14 product terms.
- With only 4 inputs the ROM is small and needs no design effort.

THINK DEEPER

Why did FPGAs replace PLAs and PALs for big designs?

Because an FPGA is thousands of **small ROMs** (look-up tables of 4–6 inputs) with flip-flops and programmable wiring between them. Today's two ideas, "a memory is any function" and "programmable interconnect", combined.

A STRONG ANSWER MENTIONS

- A PLA's AND array grows as inputs × products: large ones get slow and wasteful.
- FPGAs use thousands of small look-up tables, each a tiny 16 × 1 or 64 × 1 RAM (a ROM you can rewrite), plus flip-flops and programmable wiring.

16 REAL WORLD 71–74 min

Real world: memories you carry

- **Phone:** SRAM caches inside the processor, a few GB of DRAM refreshed every 64 ms (32 ms when hot), and 128 GB of 3D NAND flash, each cell holding 3–4 bits as different amounts of trapped charge.
- **Cards:** bank cards, SIM cards and smart ID cards keep keys and records in **EEPROM** on the chip: non-volatile, rewritable byte by byte, no battery.
- **Every microcontroller:** flash for the program, SRAM for variables, a little EEPROM for settings.

BME CORNER

A pacemaker stores its pacing settings and an event log in EEPROM and its firmware in flash: they must survive a battery change and years of use. Flash endurance (≈ 10^4–10^5 erase cycles) is a design limit for medical loggers.

17 POP QUIZ 74–78 min

Pop quiz

1. **A memory chip has 12 address lines and 8 data lines. Its capacity is…**

   A. 96 bits

   B. 4K × 8 = 32 Kbit ✓

   C. 12K × 8

   D. 64 Kbit

   2^{12} = 4096 words of 8 bits = 32,768 bits (4 KB).

2. **In a diode-matrix ROM, a stored 1 is…**

   A. a charged capacitor

   B. a diode between the word line and the bit line ✓

   C. a missing pull-down resistor

   D. a latch

   When the decoder raises the word line, the diode conducts and pulls its bit line high. No diode: the pull-down resistor reads 0.

3. **DRAM must be refreshed because…**

   A. the access transistor wears out

   B. the charge on the cell capacitor leaks away ✓

   C. the sense amplifier drifts

   D. the address decoder forgets

   A 1 is a few femtofarads of charge. Leakage drains it in a fraction of a second, so every row is read and rewritten every 64 ms.

4. **An SRAM cell is essentially…**

   A. one transistor and one capacitor

   B. a floating-gate transistor

   C. a cross-coupled inverter pair (a latch) with two access transistors ✓

   D. a fuse

   Six transistors: the bistable latch of Week 4 plus two pass transistors to the bit lines. Fast, no refresh, but 6× the area of a DRAM cell.

5. **A PAL differs from a PLA in that…**

   A. its AND array is fixed

   B. its OR array is fixed ✓

   C. both arrays are fixed

   D. it has no AND array

   PAL: programmable AND, fixed OR (each output gets its own few product terms). PROM: fixed AND (a full decoder), programmable OR. PLA: both programmable.

---

18 RECAP 78–80 min

Three things to remember

01

Memory = decoder + word lines + bit lines. $2^k \times n$. Width: chips side by side; depth: decode the top address bits into $\bar{CS}$.

02

Cells: diode/fuse (ROM, PROM), floating gate (EPROM, flash), 6T latch (SRAM), 1T1C (DRAM, refresh every 64 ms).

03

PLA: programmable AND + OR, share product terms; PAL fixes the OR, PROM fixes the AND. ROM for tables, PLA for many inputs and few terms.

HW BEFORE NEXT CLASS

Implement $F_1 = \Sigma m(0, 1, 2, 3, 7)$ and $F_2 = \Sigma m(0, 1, 2, 3, 4)$ in a PLA with the fewest product terms. Give the programming table, and verify it in the PLA lab.

SOLUTION

1. Minterms 0–3 are all the A = 0 rows. F_1 adds 7 (pairs with 3: BC); F_2 adds 4 (pairs with 0: $\bar{B}\bar{C}$):

$$F_1 = \bar{A} + BC, \qquad F_2 = \bar{A} + \bar{B}\bar{C}$$

2. $\bar{A}$ is shared: 3 product terms.

	A	B	C	F_1	F_2
$\bar{A}$	0	–	–	1	1
BC	–	1	1	1	–
$\bar{B}\bar{C}$	–	0	0	–	1

ANSWER

F_1 = $\bar{A}$ + BC, F_2 = $\bar{A}$ + $\bar{B}\bar{C}$: three product terms, $\bar{A}$ shared by both outputs.

S. M. Sakeef Sani
Adjunct Lecturer, CSE · Northern University Bangladesh
smsakeefsani.hbitcorp.com