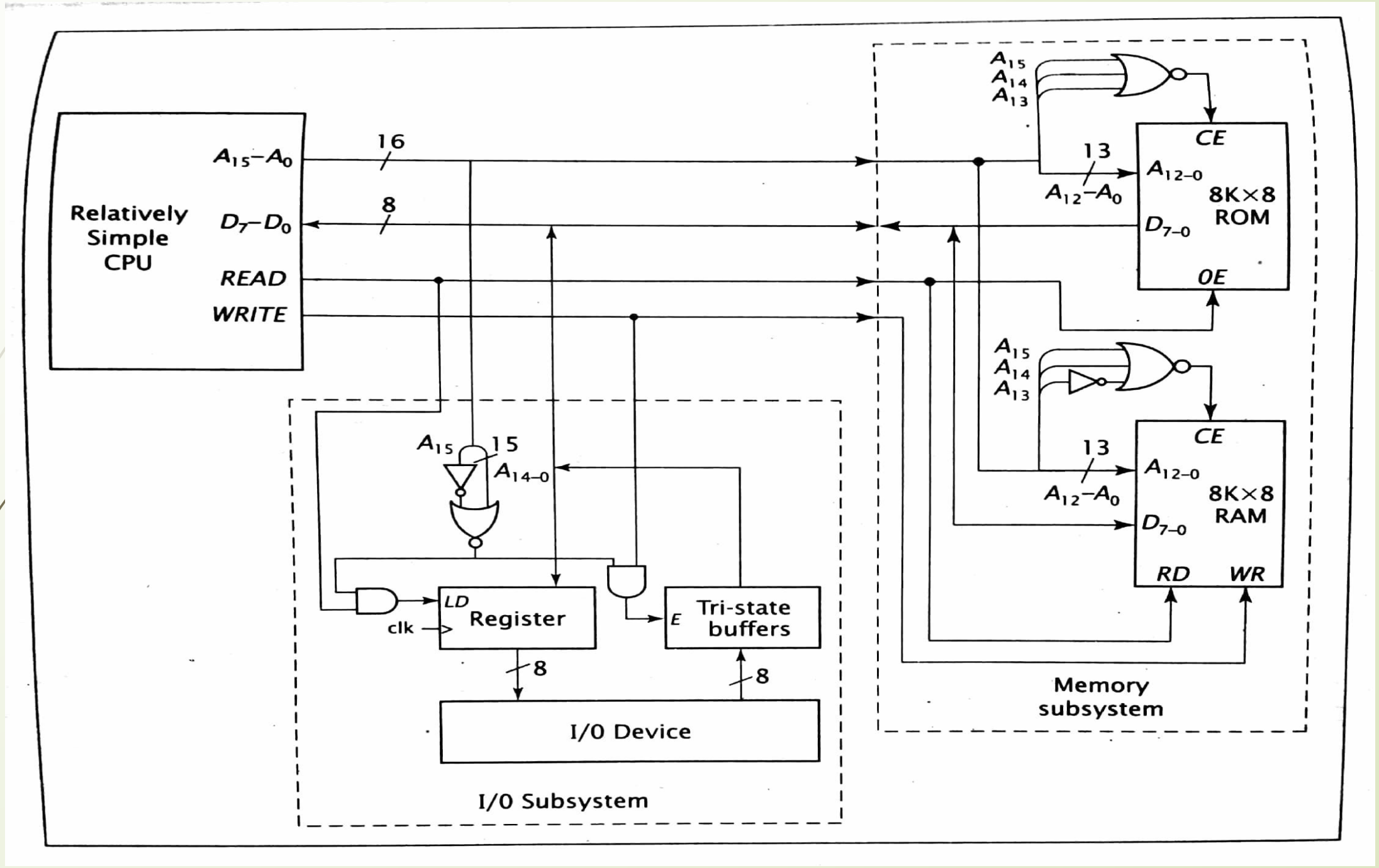
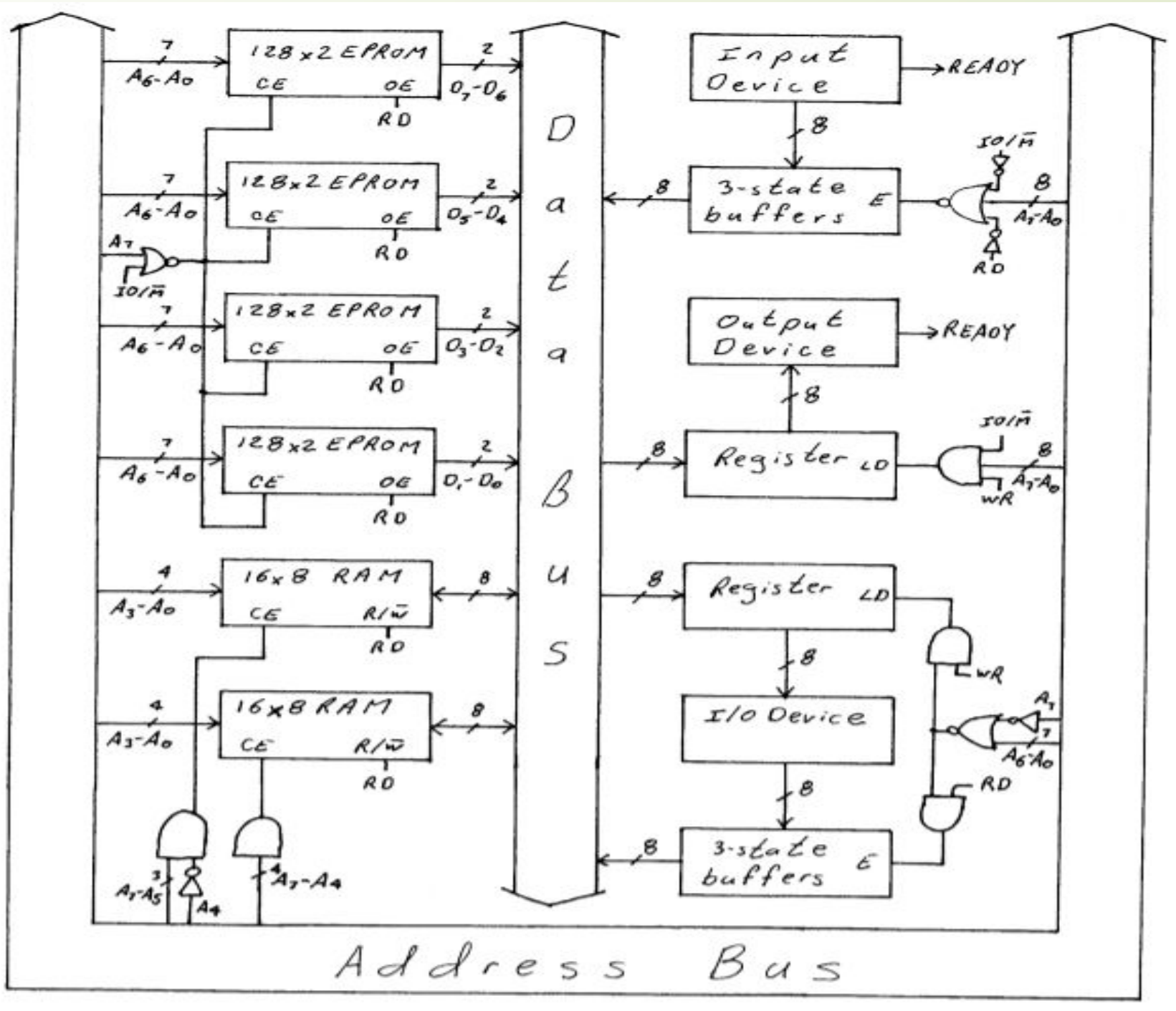
DESIGN CONTINUED

Q. Design a computer having 8k of ROM starting at address 0, followed by 8k of RAM. It will also have a memory mapped, bidirectional I/O port at address 8000H. It has 16-bit address bus and 8-bit data bus.


Relatively Simple CPU
A_{15}–A_0
D_7–D_0
READ
WRITE
16
8
A_{15}
15
A_{14-0}
LD
clk
Register
E
Tri-state buffers
8
8
I/O Device
I/O Subsystem
A_{15}
A_{14}
A_{13}
CE
13
A_{12-0}
A_{12}–A_0
A_{12-0}
8K×8 ROM
D_{7-0}
OE
A_{15}
A_{14}
A_{13}
CE
13
A_{12-0}
A_{12}–A_0
A_{12-0}
8K×8 RAM
D_{7-0}
RD
WR
Memory subsystem

Q. A computer system with an 8-bit address bus and an 8-bit data bus uses isolated I/O. It has 128 bytes of EPROM starting at address 00H constructed using 128 x 2 chips; 32 bytes of RAM starting at address E0H constructed using 16 x 8 chips; an input device with a ready signal at address 00H; an output device with a READY signal at address FFH; and a bidirectional input/output device with no READY signal at address 80H. Show the design for this system. Include all enable and load logic.


128x2 EPROM
CE
OE
RD
A6-A0
O7-O6
O5-O4
O3-O2
O1-O0
A7
IO/M
16x8 RAM
R/W
A3-A0
A7-A5
A4
A7-A4
Input Device
READY
3-state buffers
E
Output Device
Register
LD
WR
I/O Device
Data Bus
Address Bus

RELATIVELY SIMPLE CPU

As in the Very Simple CPU, this Relatively Simple CPU contains several registers in addition to those specified in its instruction set architecture. Differences between these registers and those of the Very Simple CPU are italicized:

- A *16-bit* address register, *AR*, which supplies an address to memory via A[*15..0*]
- A *16-bit* program counter, *PC*, which contains the address of the next instruction to be executed *or the address of the next required operand of the instruction*
- An 8-bit data register, *DR*, which receives instructions and data from memory *and transfers data to memory* via D[7..0]
- An *8-bit* instruction register, *IR*, which stores the opcode fetched from memory
- *An* 8-bit *temporary register,* TR, *which temporarily stores data during instruction execution*

Contd…

- LDAC Instruction
  - LDAC1: DR←M, PC←PC + 1, AR←AR + 1
  - LDAC2: TR←DR, DR←M, PC←PC + 1
  - LDAC3: AR←DR, TR
  - LDAC4: DR←M
  - LDAC5: AC←DR
- STAC Instruction
  - STAC1: DR←M, PC←PC + 1, AR←AR + 1
  - STAC2: TR←DR, DR←M, PC←PC + 1
  - STAC3: AR←DR, TR
  - STAC4: DR←AC
  - STAC5: M←DR

Execution trace of very simple CPU

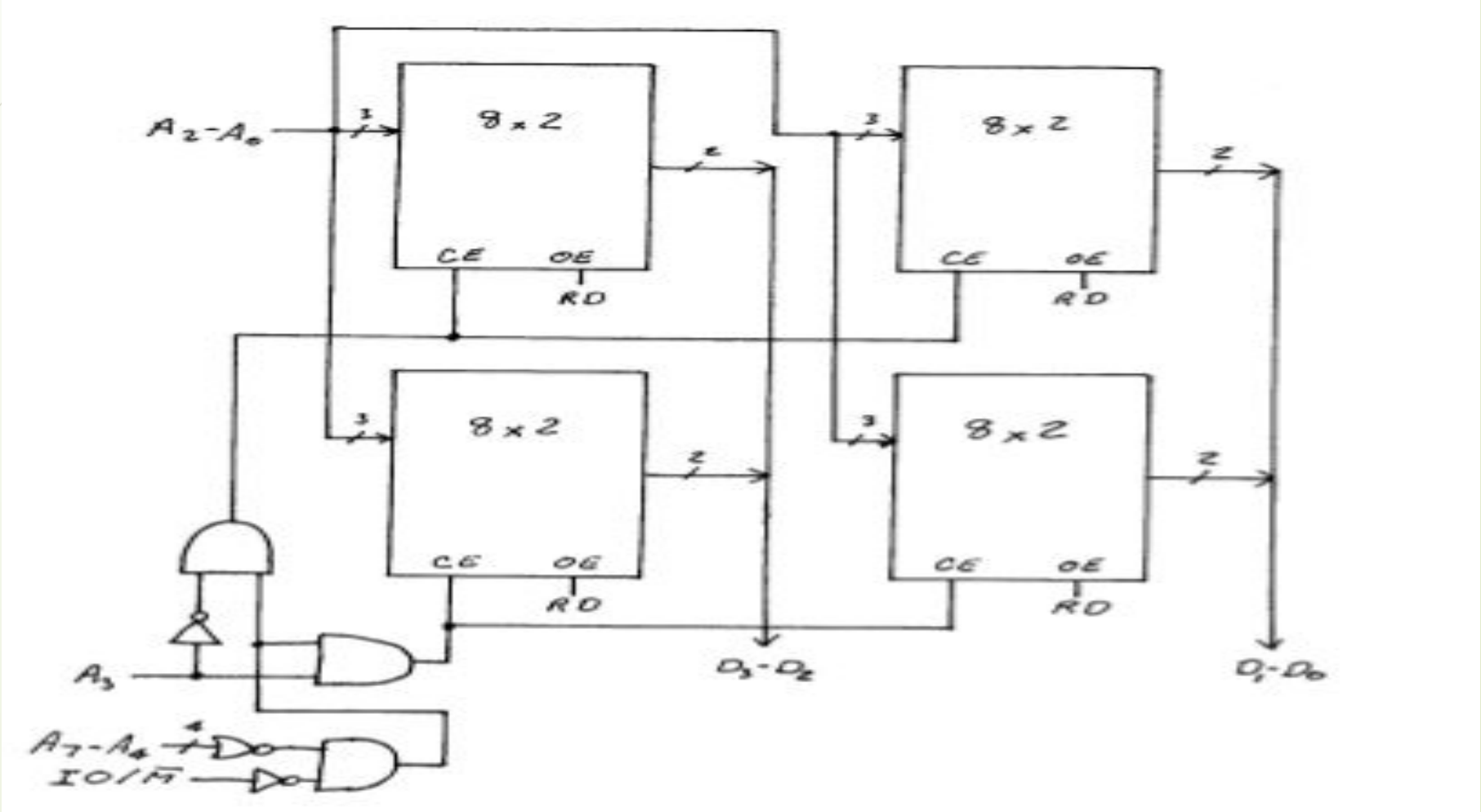
- Consider the following code segment:

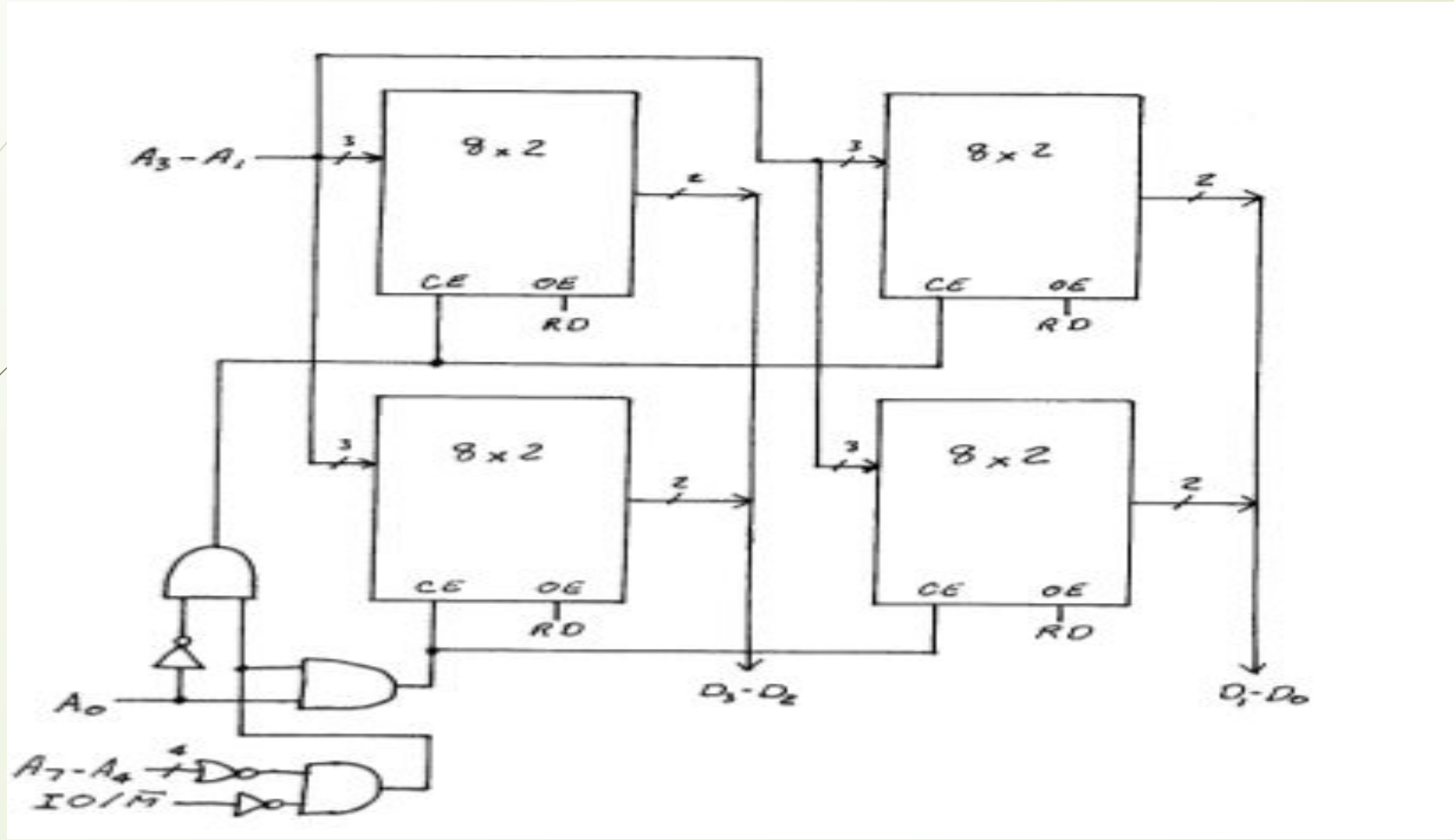
```
0:  ADD4
1:  AND5
2:  INC
3:  JMP0
4:  27H
5:  39H
```

- Initially all registers contain the value 0.

Instruction	State	Active Signals	Operations Performed	Next State
ADD 4	FETCH1	PCBUS, ARLOAD	*AR*←0	FETCH2
	FETCH2	READ, MEMBUS, DRLOAD, PCINC	*DR*←04H, *PC*←1	FETCH3
	FETCH3	DRBUS, ARLOAD, IRLOAD	*IR*←00, *AR*←04H	ADD1
	ADD1	READ, MEMBUS, DRLOAD	*DR*←27H	ADD2
	ADD2	DRBUS, ACLOAD	*AC*←0 + 27H = 27H	FETCH1
AND 5	FETCH1	PCBUS, ARLOAD	*AR*←1	FETCH2
	FETCH2	READ, MEMBUS, DRLOAD, PCINC	*DR*←45H, *PC*←2	FETCH3
	FETCH3	DRBUS, ARLOAD, IRLOAD	*IR*←01, *AR*←05H	AND1
	AND1	READ, MEMBUS, DRLOAD	*DR*←39H	AND2
	AND2	DRBUS, ALUSEL, ACLOAD	*AC*←27H ^ 39H = 21H	FETCH1
INC	FETCH1	PCBUS, ARLOAD	*AR*←2	FETCH2
	FETCH2	READ, MEMBUS, DRLOAD, PCINC	*DR*←C0H, *PC*←3	FETCH3
	FETCH3	DRBUS, ARLOAD, IRLOAD	*IR*←11, *AR*←00H	INC1
	INC1	ACINC	*AC*←21H + 1 = 22H	FETCH1
JMP 0	FETCH1	PCBUS, ARLOAD	*AR*←3	FETCH2
	FETCH2	READ, MEMBUS, DRLOAD, PCINC	*DR*←80H, *PC*←4	FETCH3
	FETCH3	DRBUS, ARLOAD, IRLOAD	*IR*←10, *AR*←00H	JMP1
	JMP1	DRBUS, PCLOAD	*PC*←0	FETCH1

- Design a 16 x 4 memory subsystem with high-order and low-order interleaving using 8 x 2 memory chips for a computer system with an 8-bit address bus.


A2-A0
3
8 x 2
CE
OE
RD
2
A3
A7-A4
4
IO/M
O3-O2
O1-O0


A3 - A1
8 x 2
CE
OE
RD
8 x 2
CE
OE
RD
8 x 2
CE
OE
RD
8 x 2
CE
OE
RD
A0
A7 - A4
IO/M
D3 - D2
D1 - D0

- Show how the following values are stored in memory in big endian and little endian formats. Each value starts at location 22H.
  A. 12345678H
  b. 0927H
  c. 5551212H

Mainly two ways to store multibyte data: **Big Endian and Little Endian**
In Big Endian, MSB is stored in location X, the following byte in location X+1 and so on.
In Little Endian, LSB is stored in location X, the following byte in location X+1 and so on.

	Big Endian		Little Endian	
a)	22	12H	22	78H
	23	34H	23	56H
	24	56H	24	34H
	25	78H	25	12H
b)	22	09H	22	27H
	23	27H	23	09H
c)	22	05H	22	12H
	23	55H	23	12H
	24	12H	24	55H
	25	12H	25	05H

Assignment

- A computer system with an 8-bit address bus and an 8-bit data bus uses memory mapped I/O. It has 32 bytes of ROM starting at address 00H constructed using 32 x 8 chip; 128 bytes of RAM starting at address 80H constructed using 64 x 4 chips; a bidirectional input/output device with no READY signal at address 40H; and an input device with no READY signal at address 60H. Show the design for this system. Include all enable and load logic.

Booth Algorithm

1: $U \leftarrow 0,\ Y_{-1} \leftarrow 0,\ i \leftarrow n$

$Y_0 Y_{-1}'$ **2**: $U \leftarrow U + X' + 1$

$Y_0' Y_{-1}$ **2**: $U \leftarrow U + X$

2: $i \leftarrow i - 1$

3: ashr(UV), cir(Y), $Y_{-1} \leftarrow Y_0$

Z'**3**: GOTO **2**

Z**3**: $FINISH \leftarrow 1$

1: $U \leftarrow 0,\ Y_{-1} \leftarrow 0,\ i \leftarrow n$

$(Y_0 \oplus Y_{-1})$ **2**: $U \leftarrow U + (X \oplus Y_0) + Y_0$

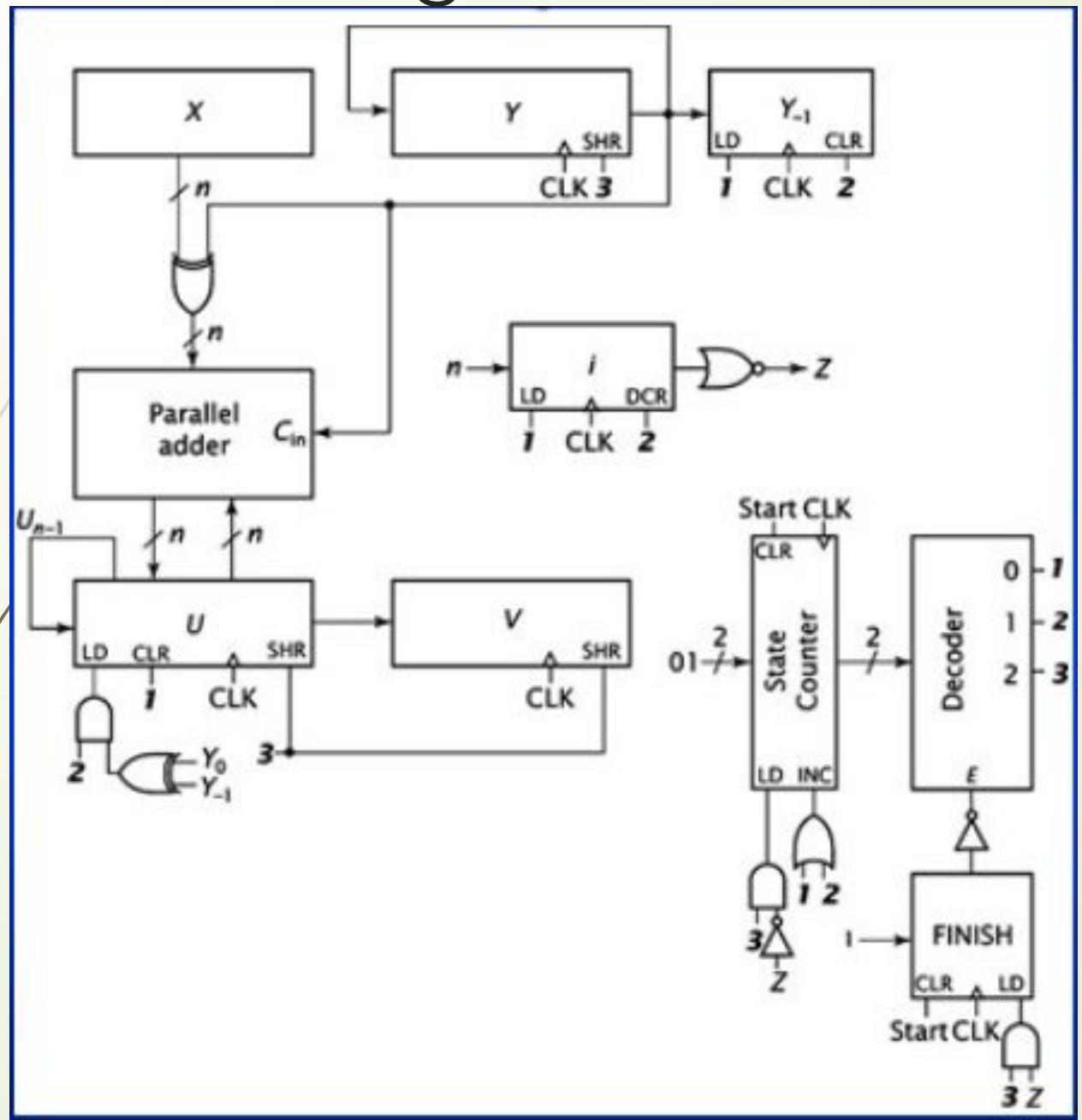
2: $i \leftarrow i - 1$

3: ashr(UV), cir(Y), $Y_{-1} \leftarrow Y_0$

Z'**3**: GOTO **2**

Z**3**: $FINISH \leftarrow 1$

Booth Algorithm



1:	$U \leftarrow 0,\ Y_{-1} \leftarrow 0,\ i \leftarrow n$
$(Y_0 \oplus Y_{-1})$ **2**:	$U \leftarrow U + (X \oplus Y_0) + Y_0$
2:	$i \leftarrow i - 1$
3:	ashr(UV), cir(Y), $Y_{-1} \leftarrow Y_0$
Z'**3**:	GOTO **2**
Z**3**:	$FINISH \leftarrow 1$