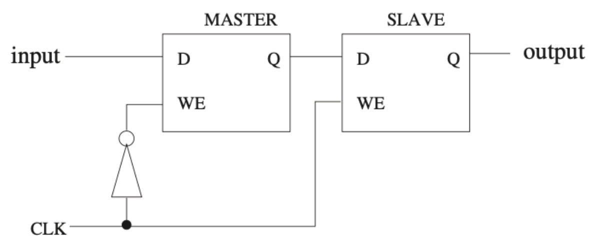
Homework 3

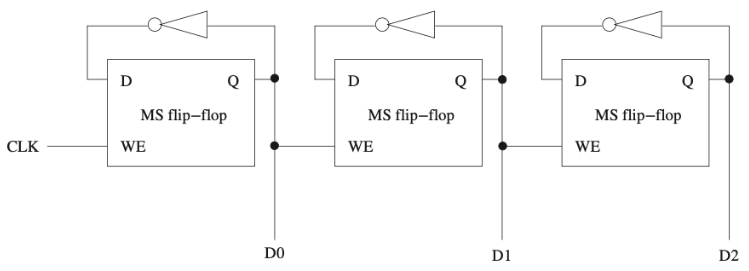
2023.07.14 沈韵沨 3200104392

Chap 3

★**3.53** The master/slave flip-flop we introduced in the chapter is shown below. Note that the input value is visible at the output after the clock transitions from 0 to 1.



Shown below is a circuit constructed with three of these flip-flops.



Fill in the entries for D2, D1, D0 for each of clock cycles shown

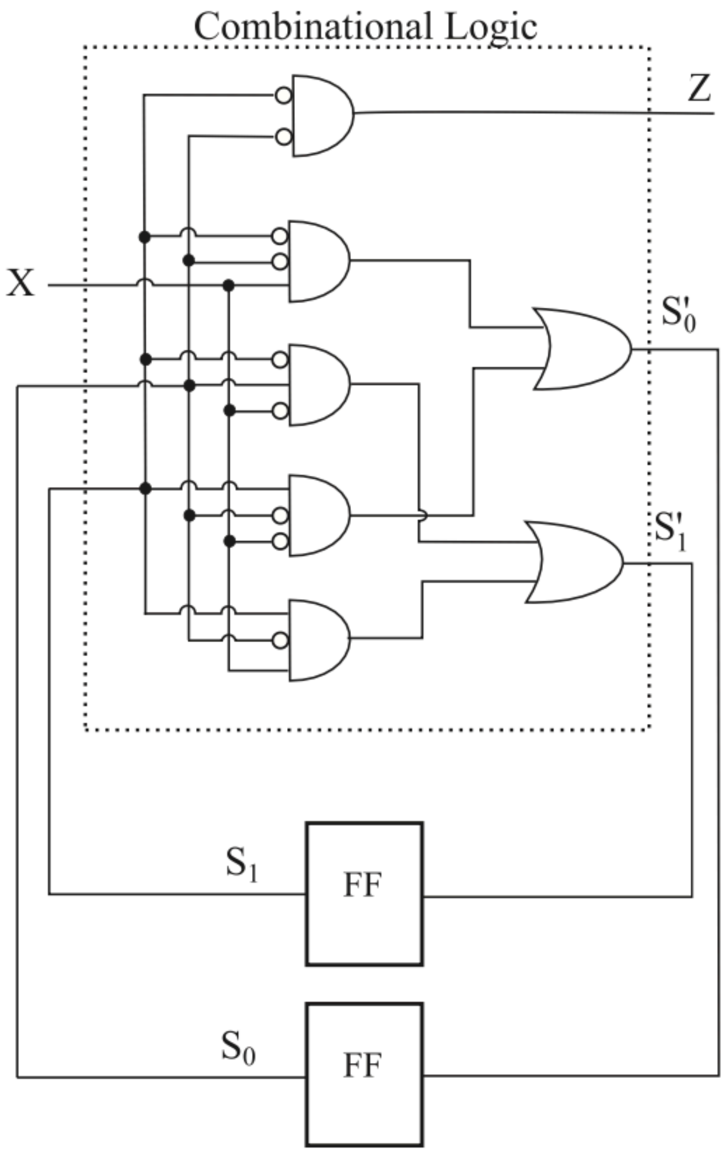
	cycle 0	cycle 1	cycle 2	cycle 3	cycle 4	cycle 5	cycle 6	cycle 7
D2	0							
D1	0							
D0	0							

In ten words or less, what is this circuit doing?

	cycle 0	cycle 1	cycle 2	cycle 3	cycle 4	cycle 5	cycle 6	cycle 7
D2	0	1	1	1	1	0	0	0
D0	0	1	1	0	0	1	1	0
D1	0	1	0	1	0	1	0	1

The circuit works as a `3-bit down counter`.

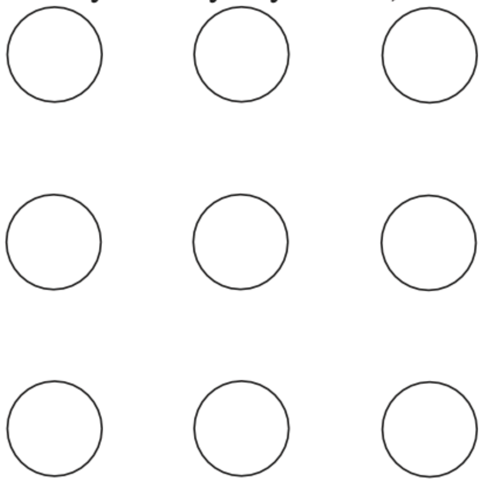
★**3.61** The logic diagram shown below is a finite state machine.



a. Construct the truth table for the combinational logic:

S1	S0	X	Z	S1’	S0’
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

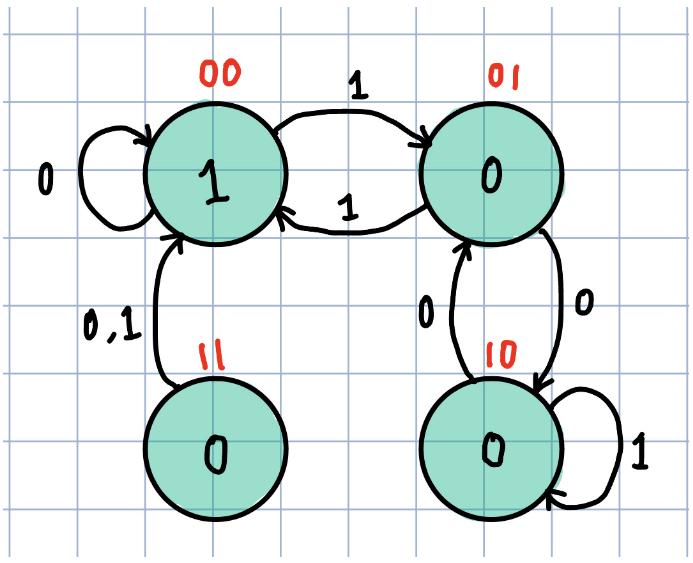
b. Complete the state machine.
(We have provided nine states. You will not need all of them. Use only as many as you need):



a. Truth table for combinationtional logic:

S1	S0	X	Z	S1'	S0'
0	0	0	1	0	0
0	0	1	1	0	1
0	1	0	0	1	0
0	1	1	0	0	0
1	0	0	0	0	1
1	0	1	0	1	0
1	1	0	0	0	0
1	1	1	0	0	0

b. The state machine:



Chap 4

4.1 Name the five components of the von Neumann model. For each component, state its purpose.

The five components of the von Neumann model are:

1. `Memory` : Store instructions & data.
2. `Processing Unit` : Execute instructions & process data.
3. `Input` : Load instructions & data.
4. `Output` : Store instructions & data, display result.
5. `Control Unit` : Keep track of both where we are within the process of executing the program and where we are in the process of executing each instruction, in order to coordination of the work of all components.

4.3 What is misleading about the name *program counter*? Why is the name *instruction pointer* more insightful?

- The misleading is the register called `program counter` is not a *counter* , but store the **next instruction's address** in fact.
- The name `instruction pointer` (by Intel) might be more insightful.

4.8 Suppose a 32-bit instruction takes the following format:

OPCODE	DR	SR1	SR2	UNUSED

If there are 225 opcodes and 120 registers,

a. What is the minimum number of bits required to represent the OPCODE?
b. What is the minimum number of bits required to represent the destination register (DR)?
c. What is the maximum number of UNUSED bits in the instruction encoding?

a.

$$\because \begin{cases} n_{OPCODE} = 255 \\ 2^7 = 128 < 255 < 256 = 2^8 \end{cases}$$

$$\therefore The\ minimum\ number\ of\ bits\ required\ to\ represent\ the\ OPCODE = 8$$

b.

$$\because \begin{cases} n_{register} = 120 \\ 2^6 = 64 < 255 < 128 = 2^7 \end{cases}$$

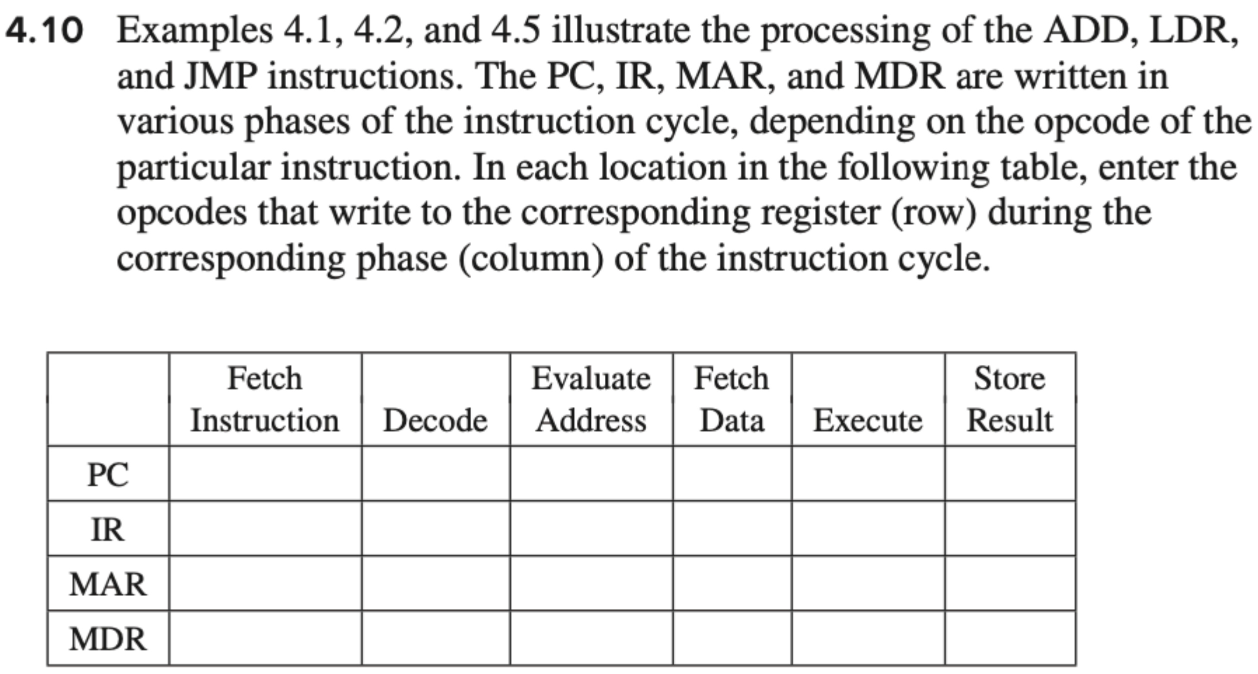
$$\therefore The\ minimum\ number\ of\ bits\ required\ to\ represent\ the\ DR = 7$$

c.

$$According\ to\ a.\ \&\ b. \begin{cases} min(OPCODE) = 8\ bit \\ min(DR) = 7\ bit \\ min(SR1) = min(SR2) = min(DR) = 7\ bit \end{cases}$$

$$\begin{aligned} max(UNUSED) &= total - (min(OPCODE) + min(DR) + min(SR1) + min(SR2)) \\ &= 32 - (8 + 7 + 7 + 7) = 3\ bit \end{aligned}$$

$$\therefore The\ maximum\ number\ of\ UNUSED\ bits\ in\ the\ instruction\ coding\ = 3$$



4.10 Examples 4.1, 4.2, and 4.5 illustrate the processing of the ADD, LDR, and JMP instructions. The PC, IR, MAR, and MDR are written in various phases of the instruction cycle, depending on the opcode of the particular instruction. In each location in the following table, enter the opcodes that write to the corresponding register (row) during the corresponding phase (column) of the instruction cycle.

	Fetch Instruction	Decode	Evaluate Address	Fetch Data	Execute	Store Result
PC						
IR						
MAR						
MDR						

	Fetch Instruction	Decode	Evaluate Address	Fetch Data	Execute	Store Result
PC	`0001` , `0110` , `1100`				`1100`	
IR	`0001` , `0110` , `1100`					
MAR	`0001` , `0110` , `1100`			`0110`		
MDR	`0001` , `0110` , `1100`			`0110`		