Interfacing IO Devices

Dr. M. Shiple

Microprocessor Architecture, 2019


بسم الله الرحمن الرحيم


وقل رب زدني علما

Instruction cycles of IO

Out instruction

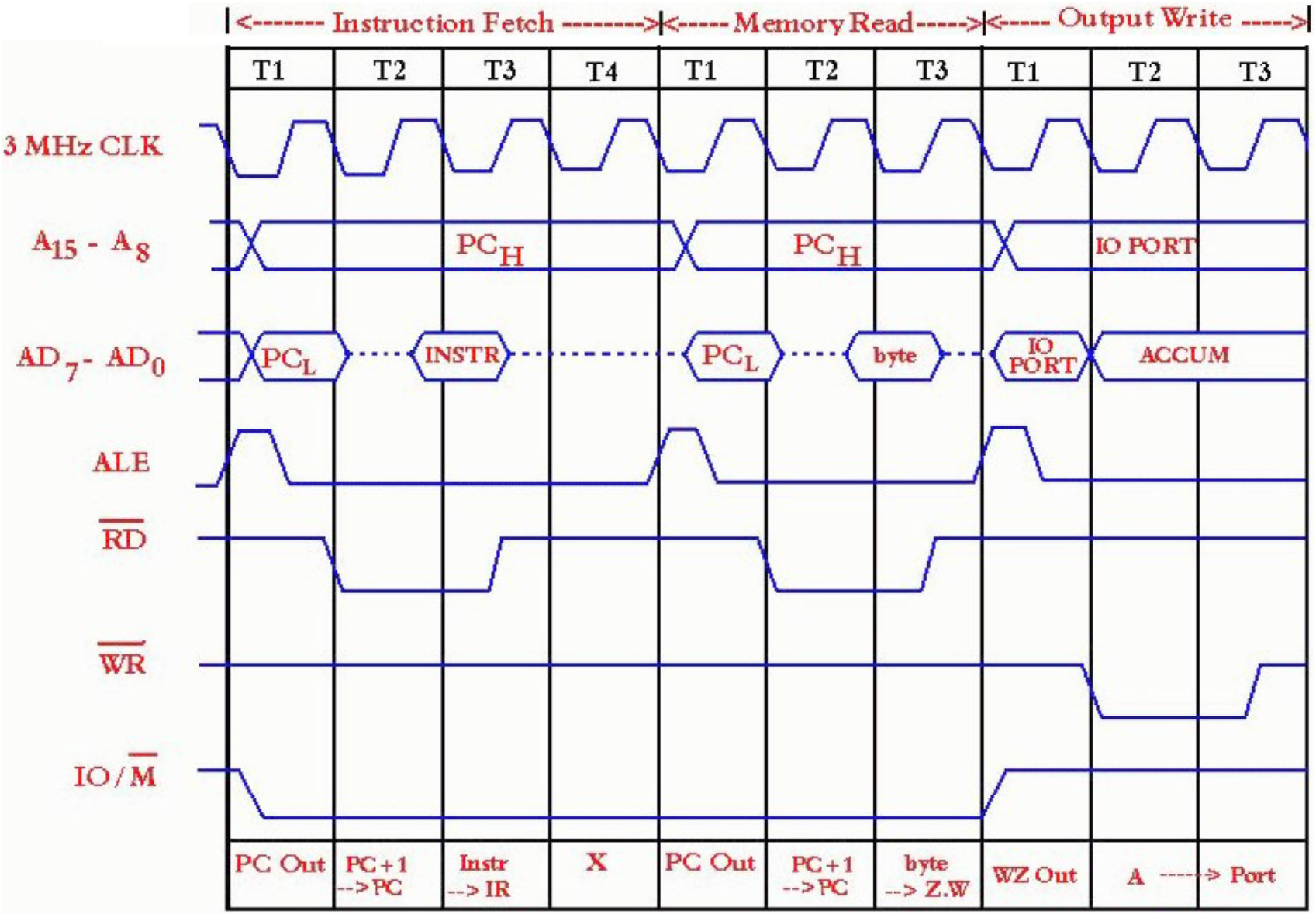
OUT instruction:- data output from accumulator to device e.g printer, display

Syntax

Out p ;[p]= A Note the address of port 8 bits (max. 256 ports only)

Note: Add(high) =Add(Low)=p at output write cycle

Out instruction



Out instruction

OPFMC: $IO/\overline{M} = 0,\ S_1 = 1,\ S_0 = 1$

T_1: A_{15}–$A_8 \leftarrow$ (PCH); AD_7–$AD_0 \leftarrow$ (PCL); ALE

T_2: PC $\leftarrow$ PC + 1; $\overline{RD} = 0$; BDB $\leftarrow$ M (AB)

T_3: $\overline{RD} = 1$; IR $\leftarrow$ BDB

T_4: Out p is set to 1

MRDMC: $IO/\overline{M} = 0,\ S_1 = 1,\ S_0 = 0$

T_1: AD_0–$AD_7 \leftarrow$ PCL; A_8–$A_{15} \leftarrow$ PCH; ALE

T_2: PC $\leftarrow$ PC + 1; $\overline{RD} = 0$; BDB $\leftarrow$ M (AB)

T_3: $\overline{RD} = 1$; Z $\leftarrow$ (BDB)

IWRMC : $IO/\overline{M} = 1,\ S_1 = 0,\ S_0 = 1$

T_1: A_{15}–$A_8 \leftarrow$ (P); AD_7–$AD_0 \leftarrow$ (P); ALE

T_2: $\overline{WR} = 0$; $(AD_7$–$AD_{0)}) \leftarrow$A ;

T_3: $\overline{WR} = 1$; (BDB) $\leftarrow$ A

In instruction

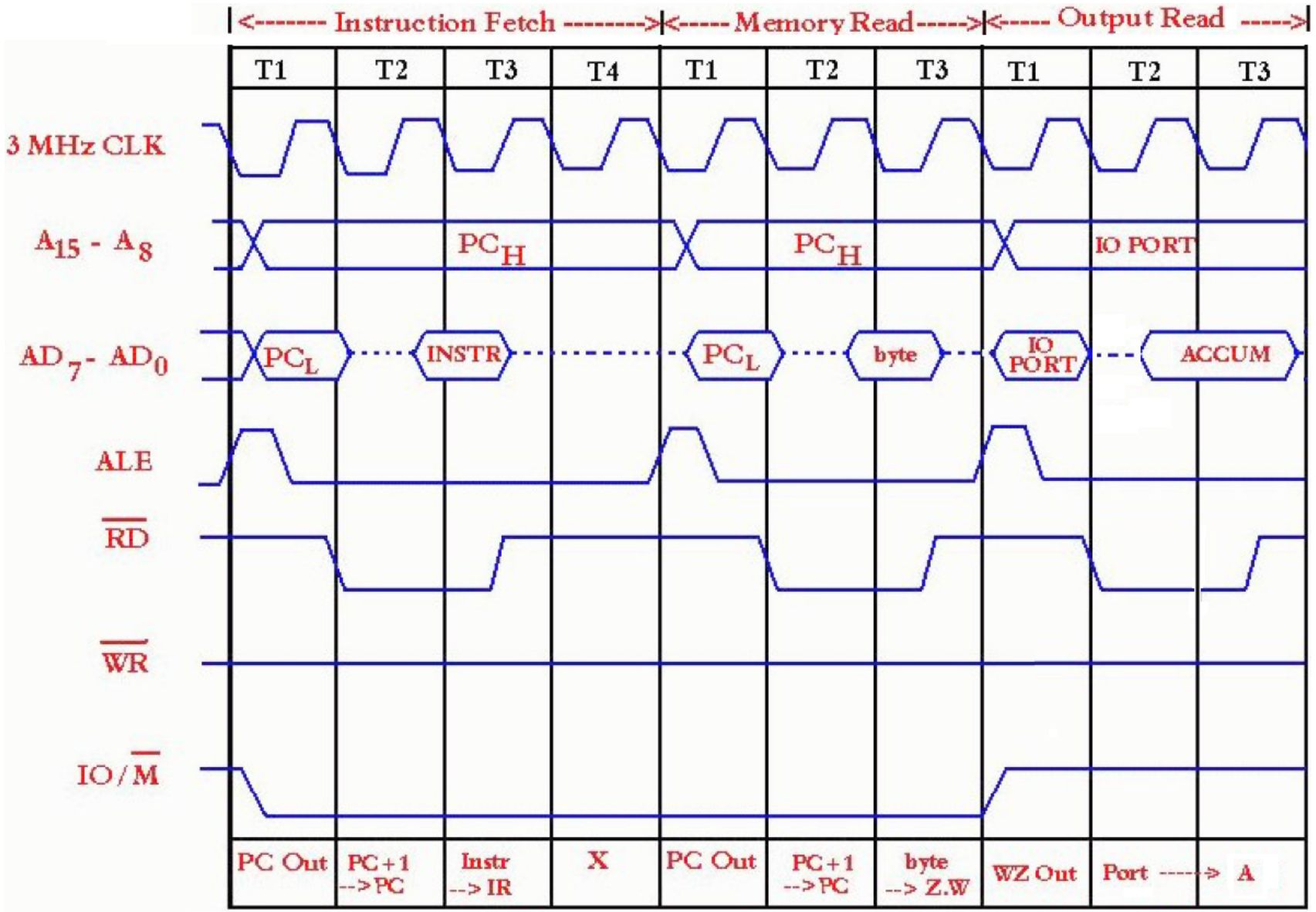
IN instruction:- data input into the accumulator from device eg. Keyboard

Syntax

In p	;A=[p]	Note the address of port 8bits (max. 256 ports only)

Note: Add(high) =Add(Low)=p at output read cycle

In instruction

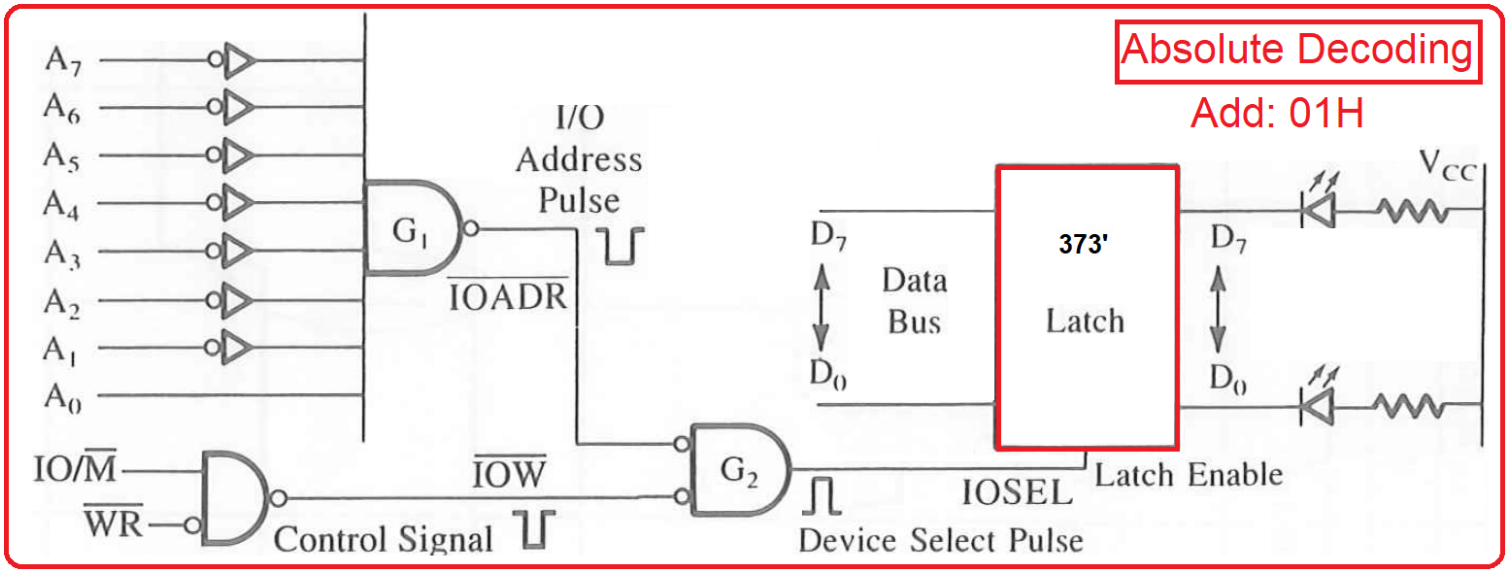


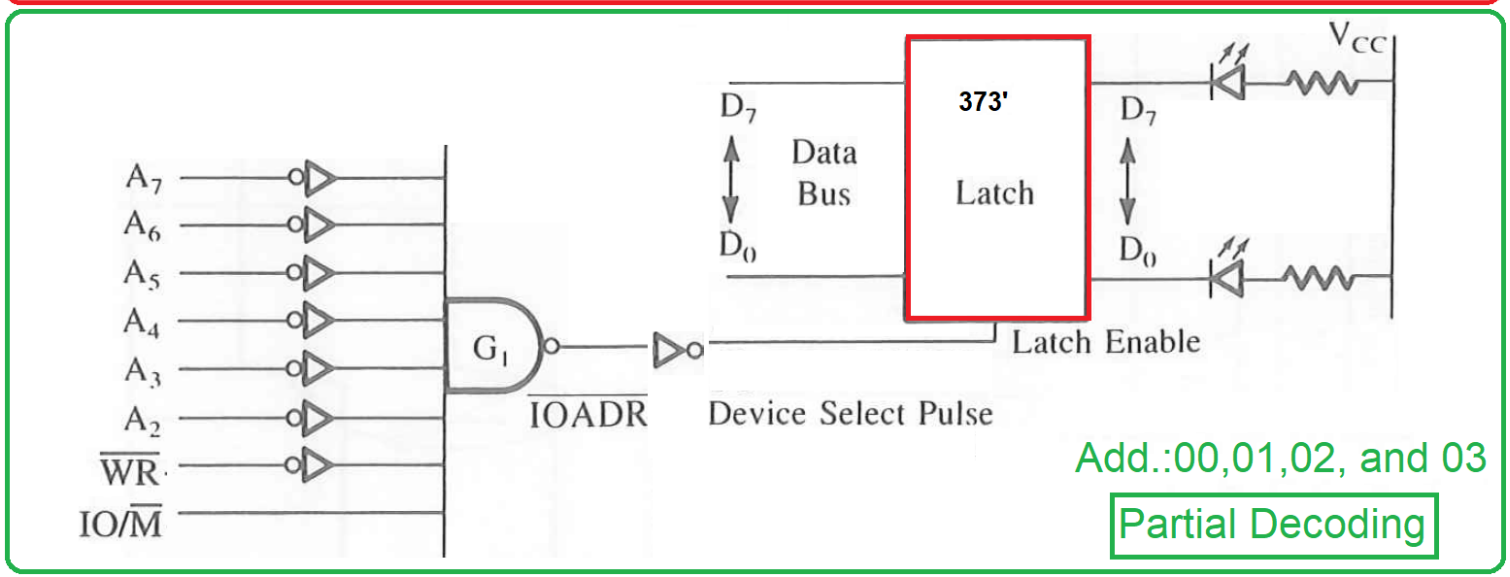
IO interface

I/O Port Identification and Addressing

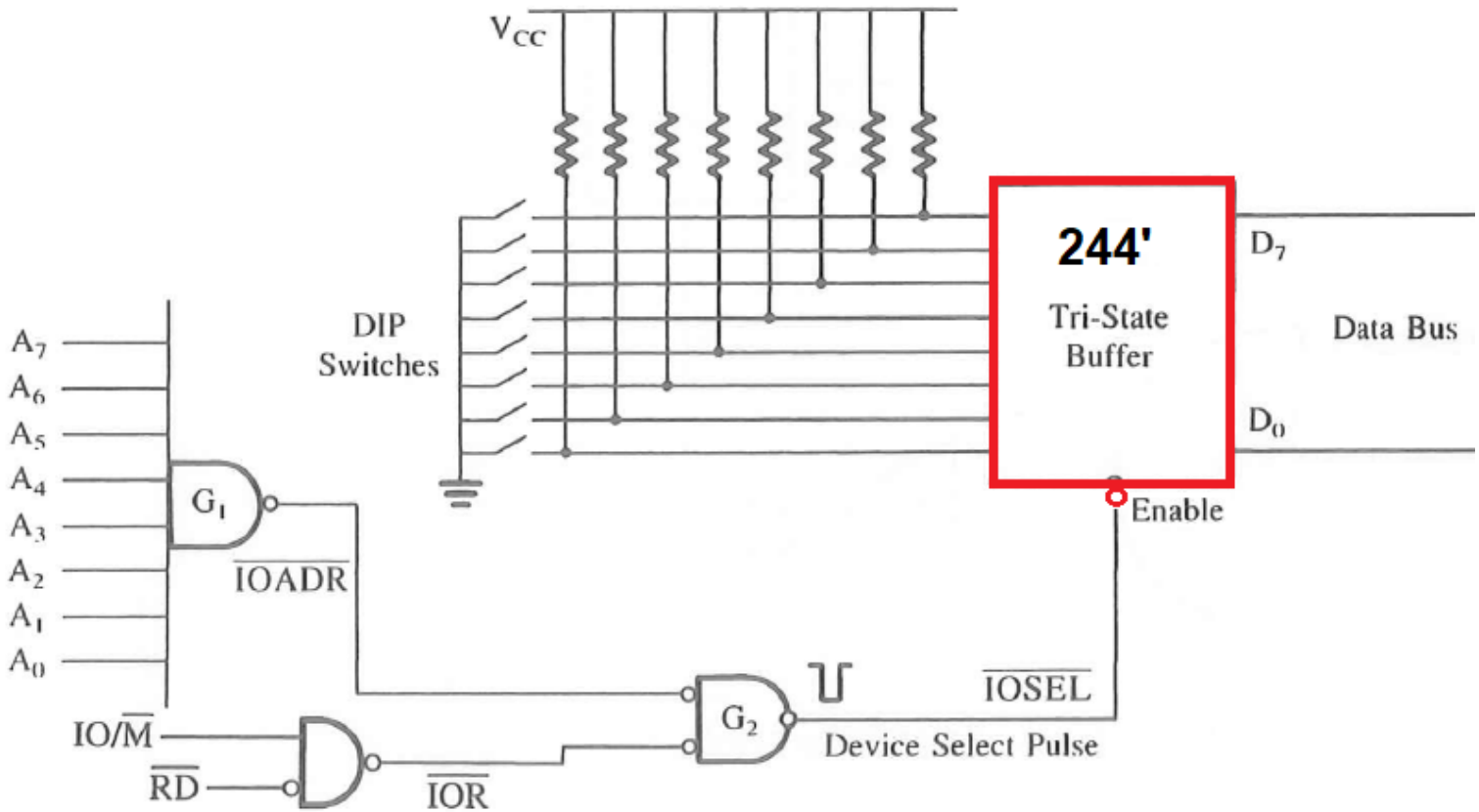
1. Memory Mapped I/O:
   1. Access and identify as memory registers using memory space.
   2. Memory related control signal0
   3. 8085 treats I/O ports as if it was communicating with a memory location.
   4. 16-bit address i.e. from 0000H to FFFFH
2. Peripheral Mapped I/O
   1. Separate address scheme (8-bit i.e from 00H to FFH)
   2. Enabled and identified by I/O related control signals

Peripheral Mapped I/O decoding circuit (Output)

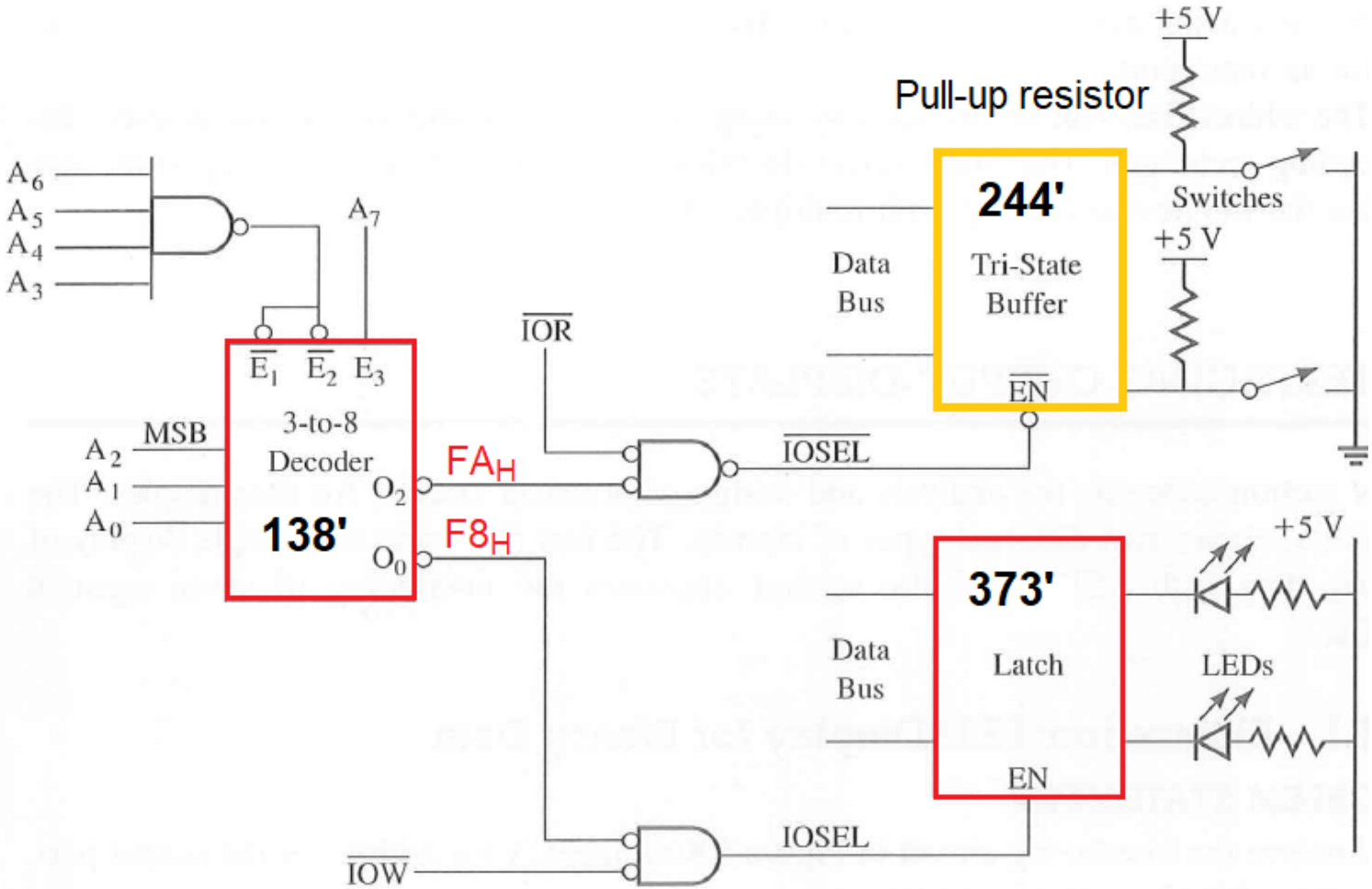




Peripheral Mapped I/O decoding circuit (Input)



Peripheral Mapped I/O decoding circuit (In/Out)



Memory-mapped I/O Vs. Peripheral I/O

Memory Address	Machine Code	Mnemonics	Comments
2050	32	STA 8000H	; Store Contents of accumulator in memory location 8000H
2050	0		
2050	80		

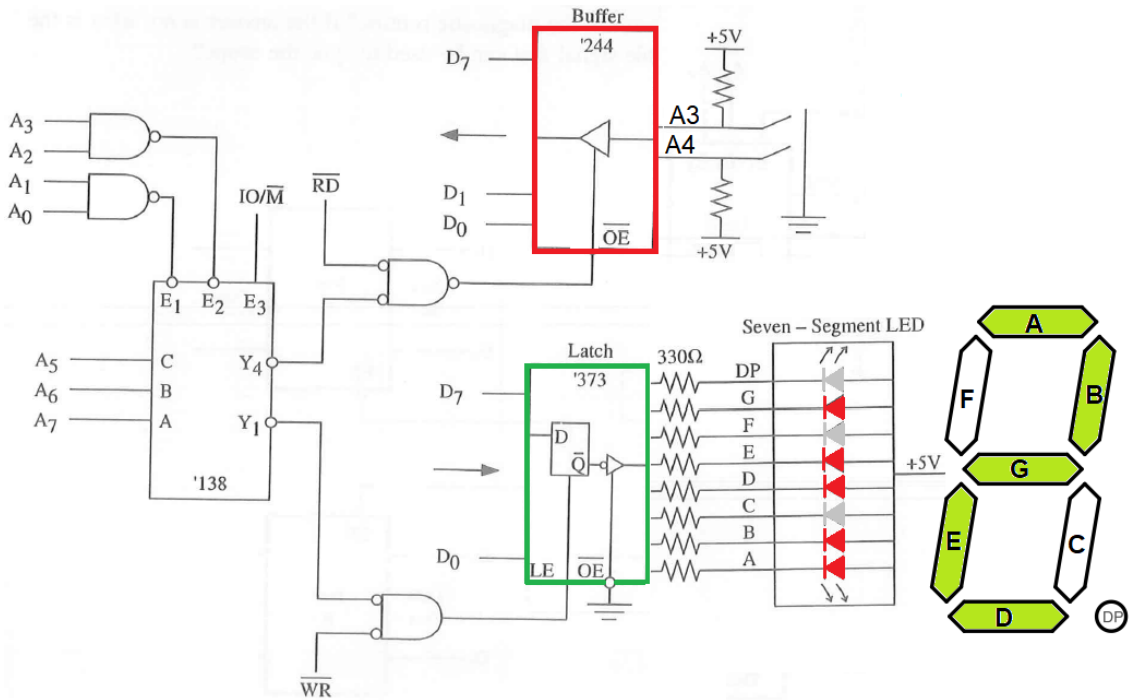
1. STA (Store A Direct) is used to store the content of accumulator to the specified memory register.
2. Here the memory address is 8000H (16-bit)
3. If we connect an output device with this address (8000H), the accumulator contents will transfer to the output device.
4. Try arithmetic and logical operations ✓.
5. Compare between STA 8000H and Out 80H (from point of view memory & speed)

Memory-mapped I/O Vs. Peripheral I/O

Characteristics	Memory-mapped I/O	Peripheral I/O
Control signals for Input/Output	$\overline{MEMR}/\overline{MEMW}$	$\overline{IOR}/\overline{IOW}$
3.Instructions available	STA; LDA; ADD M; SUB M; ANA M: etc	IN and OUT
4.Data transfer	any register $\Leftrightarrow$ I/O devices	Only I/O $\Leftrightarrow$ the Accumulator
5.Maximum number of I/Os possible	64K is shared between I/Os and system memory	256 input devices and 256 output devices
6.Execution speed	13 T-states (STA,LDA) 7 T-sates (MOV M,R)	10 T-states
7.Hardware requirements	More hardware for decode 16-bit	Less hardware for decode 8-bit
8.Other features	Arithmetic or logical directly performed	Not available

Exercise 1

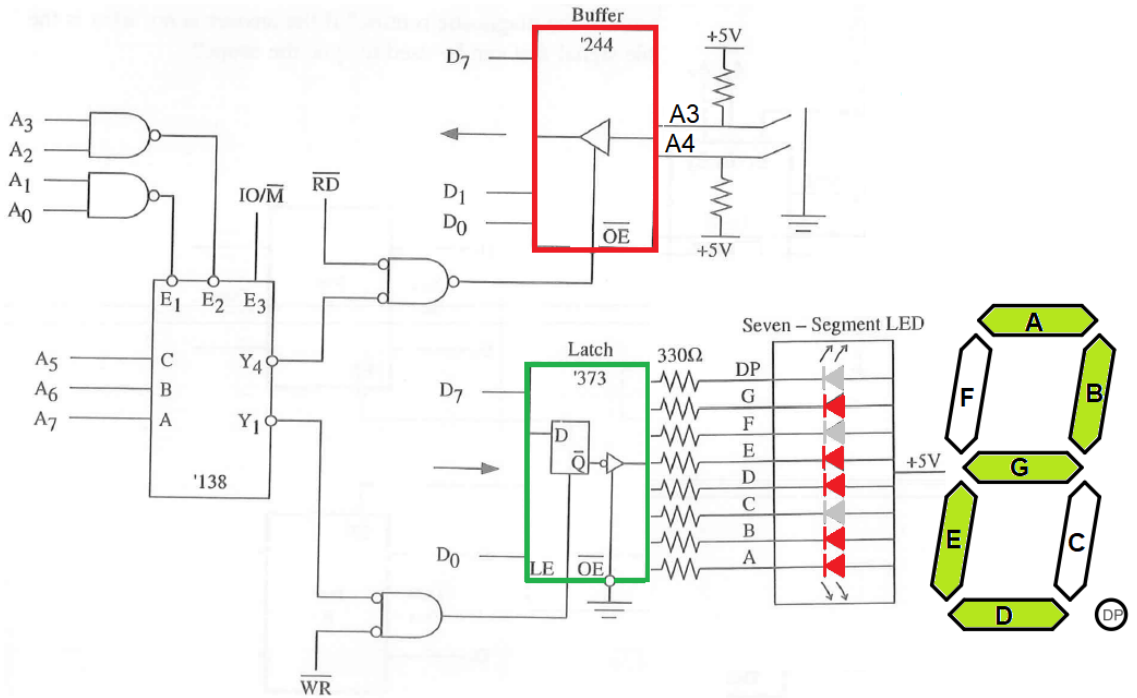
Specify the addresses of ports of next circuit and determine the port type (Memory-mapped I/O / Peripheral I/O) and mode (In/Out)?



Peripheral I/O : (In add=3FH or 2FH) (Out add=9FH or 8FH)

Exercise 1

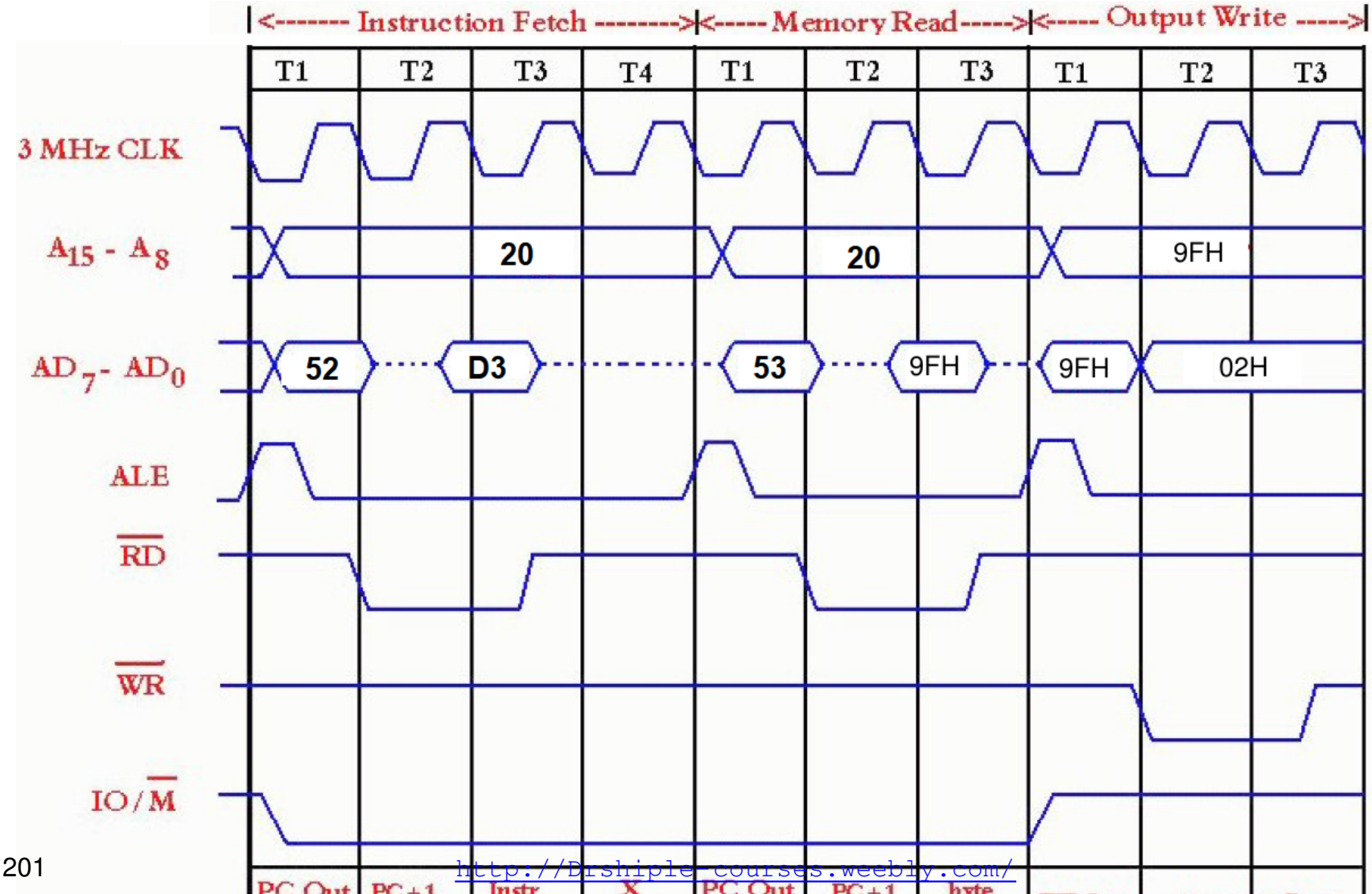
Write assembly code to display 2 on 7-segment display? assume code start at 2050 H ? Draw instruction cycle of Out instruction?



2050	MVI	A,02H
2052	Out	9FH
2054	Hlt	

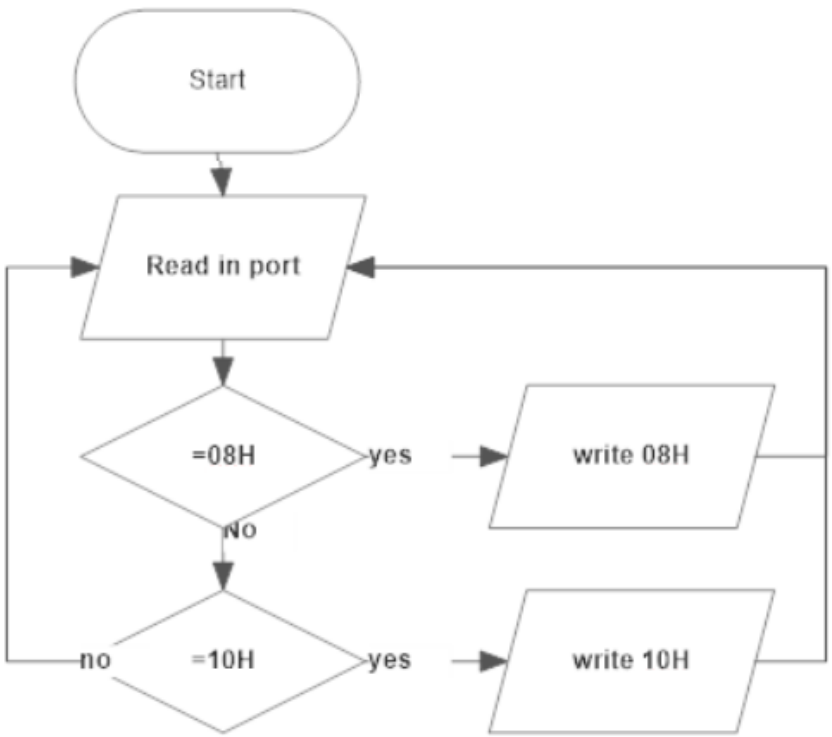
Exercise 1

Write assembly code to display 2 on 7-segment display? assume code start at 2050 H



Exercise 1

Write assembly code to read switches from input port and continue read until on of them is closed. Once a switch is closed , turn on the corresponding led (A or B) ? assume code start at 2050 H




```
4200 C3 03 042    jmp start

          ;data


          ;code
4203 DB 3F        start: IN 3Fh          ; read in port
4205 FE 08               cpi 08h         ; compare A=08
4207 CC 13 42            cz  display     ;if A=08 display
420A FE 10               cpi 10h         ; compare A=08
420C CC 13 42            cz  display     ;if A=08 display
420F C3 03 042           jmp start
4212 76           hlt

4213 D3 8F        display: Out 8Fh
4215 C9                    ret
```