★ Architecture of 8051:-

Features of 8051 :-

* 8-bit Microcontroller.
* 16-bit address lines, 8 bit data lines
* 4-ports - port 0 to port 3 (32 bidirectional I/O ports).
* Five Interrupts.
  - → Two external interrupts
  - → Three internal interrupts.
* Program memory (ROM), data memory (RAM)

  internal:
  - → 4 Kbytes on-chip ROM.
  - → 128 Kbytes of on-chip RAM.

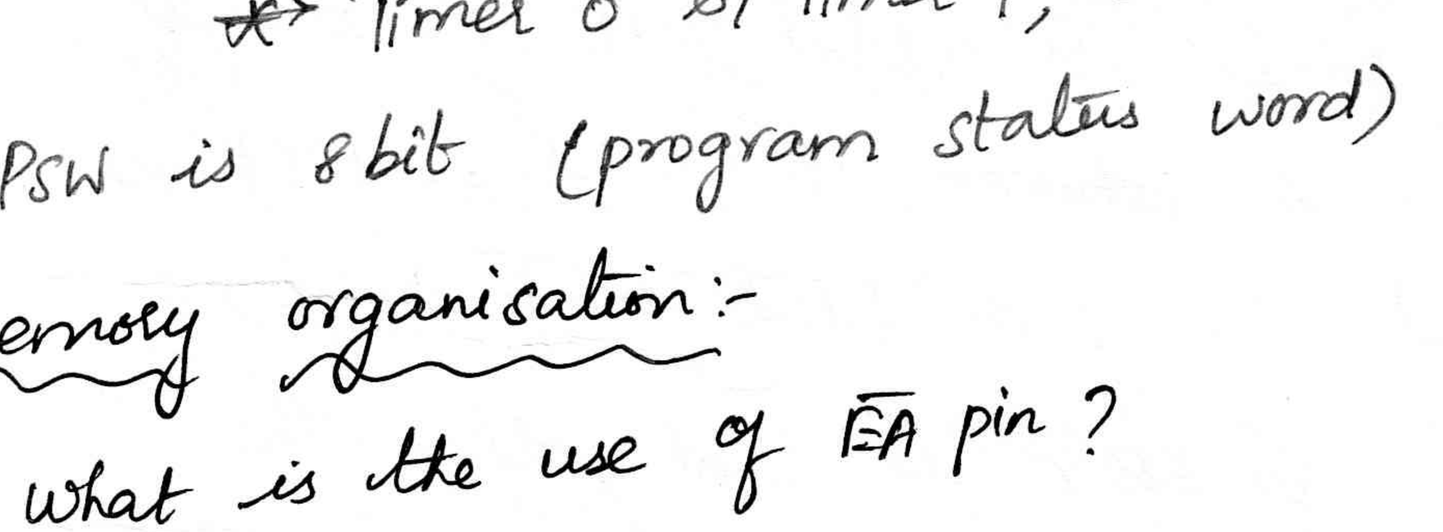
  external:
  - → 64 Kbytes external RAM, ROM
  - → Timer 0 & Timer 1, each is 16-bit.
* PSW is 8 bit (program status word)

Memory organisation:-

What is the use of $\overline{EA}$ pin?

$\overline{EA}$ = external access

Program memory - ROM



Architecture of 8051:-

→ It is a 8-bit controller, it has 8-bit data line & 16 bit address lines. Internal program memory is 4kbytes ROM; 128 Kbytes of RAM & External 64 Kbytes of RAM, ROM (or) program memory.

Registers A & B:-

There are two main register A & B. Each register is 8 bit. To hold value arithmetic & logical operation A & B register are used finally the result is stored in A, hence it is called as accumulator.

RAM:- 128 kbytes here 30 to 7F is used as general purpose & 00 to 2F.

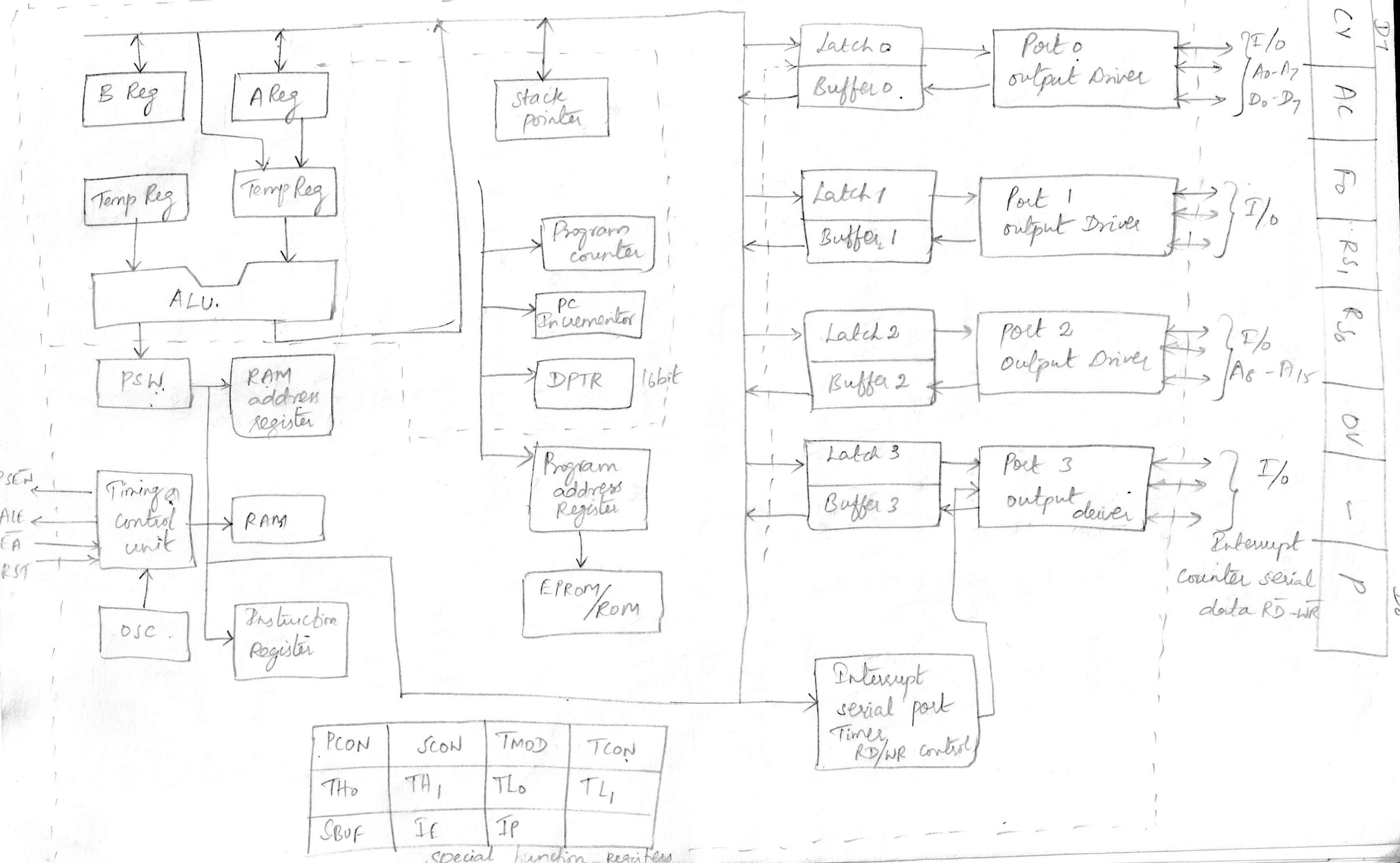
→ Program counter is used to hold the address of next instruction to be executed. It does not has internal address.

→ stack pointer is 8 bit, it can occupy any memory address is between 00 to 7H of 128 Kbytes. It is accessed by LIFO (last in first out).

→ If the processor is reset, then the stack pointer always points 07H.

→ Data pointer (DPTR):- It is a 16 bit register, it is used to hold the external data memory address, It can be divided into two DP_H, DP_L it points internal data memory. The DP_H, DP_L has internal address 83H, 82H respectively.

Architecture of 8051



PCON	SCON	TMOD	TCON
TH_0	TH_1	TL_0	TL_1
SBUF	IE	IP	

Special function registers

Timing & control unit:-

→ It is used to provide necessary timing control.

PSEN - program states enable.

EA →0 = External, address

External access →1 = upto 4 bytes internal address

(5)

→ There are 4 ports; port 0, 1, 2, 3. All ports acts as I/O port & some ports have multifunctions like (ie) eg port 0 acts as I/O, Address & data bus. port 2 acts as I/O & address bus. port 3 acts as I/O & interrupts (SI).

→ port 0 requires external pull up register.

→ PSW:- Program states word. It is a 8 bit flag register

i) CY - where there is a overflow in D_7 then carry flag sets.

ii) AC - when there is a carry from D_3 to D_4, then auxiliary carry flag sets.

iii) F0 - reversed for future purpose.

iv) overflow - when there is a overflow in signed bit then overflow flag sets.

v) Parity - when there is odd no of 1's then parity flag sets.

vi) RS_0, RS_1 - Register select, there are four bank registers (0, 1, 2, 3)

→ Each bank register has 8-bit (R_0-R_7) thus total 32 register available.

→ But defaultly Bank 0 is selected, if we need to access to other bank then we should select by set instruction.

eg:- SET B PSW3 BANK 1 selected

SETB PSW4 BANK 2 selected.

(*) $\overline{EA}$ (external access):-

→ When $\overline{EA}$ pin is high, program fetches to address 0000H through 0FFFH are directed to the internal ROM and program fetches to address 1000H through FFFFH are directed to external ROM.

→ $\overline{EA}$ pin is low, all address (0000H to FFFFH) fetched by program are directed by to external ROM.

(*) $\overline{PSEN}$: (program store Enable)

→ It is a output pin.

→ 8051 based system in which an external ROM holds the program code, this pin is connected to the OE pin of ROM.

(*) ALE :- (Address Latch Enable)

→ ALE is the output pin & it is active high.

→ ALE pin is used for demultiplexing the address & data.

Special function registers :-

* All the resources in the 8051 can be accessed through special function registers in the internal data memory.

* There are 21 special function register.

Serial Data Buffer (SBUF)

Register holds the data that is to be transmitted through the serial port and also hold the data that is received.

Control & Status Register :

These registers contain the control and status bits of the interrupt systems, timers and serial port operations.

* SPECIAL FUNCTION REGISTER :- (SFRs)

→ 8051 is provided with 21 special function registers and they are used for selecting various programmable features of the microcontroller.

→ Each SFR has an Internal one-bytes address assigned to it.

A & B registers :-

A & B registers are called CPU registers. They are used to hold the data for most of CPU (AL operations. These registers are bit addressable.

	Bit address		Byte address	
	255	248		
B	247	B 240	240	(F0)
ACC	231	A 224	224	(E0)
PSW	215	PSW 208	208	(D0)
IP	191	184	184	(B8H)
P3	183	176	176	(B0H)
IE	175	168	168	(A8H)
P2	167	160	160	(A0H)
SBUF			153	(99H)
SCON	159	152	152	(98H)
P1	151	144	144	(90H)
TH1			141	(8DH)
TH0			140	(8CH)
TL1			139	(8BH)
TL0			138	(8AH)
TMOD			137	(89H)
TCON			136	(88H)
PCON			135	(87H)
DPH			131	(83H)
DPL			130	(82H)
SP			129	(81H)
P0	135	128	128	(80H)

DPTR - Data pointer:

→ The data pointer is a 16-bit register, which is used to hold the 16-bit address of data memory.

→ The 16 bit data pointer can also be used as two numbers of 8-bit data pointers namely DPH, DPL.

→ 8 bit data pointers are used for accessing internal RAM or SFR.

→ 16 bit data pointer is used for accessing external data memory.

→ Internal memory byte address for DPH - 83H & DPL - 82H

Program status word:

→ Program status word stores the status of the result of the ALU operation and some of the status the processor by means of a 1-bit status called flag.

D7							D0
CF	AF	F0	RS_1	RS_0	OF	-	PF

Stack pointer :-

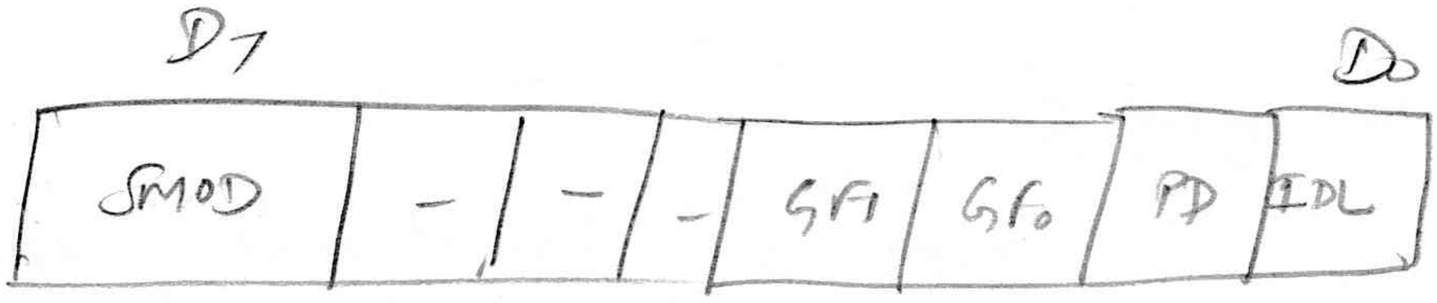
→ It always holds 8 bit address at the top of the stack.

→ 8051 supports the LIFO stack, stack may reside anywhere in on-chip RAM.

→ After reset, the stack pointer is initialized to 07H. The stack can be accessed by using PUSH & POP instructions.

Power control register : (PCON)

→ PCON register is used for power control & baud rate selection.



IDL

IDL = 1; Controller can be driven to the idle mode.

→ when Idle mode is activated, the clock signal is stopped to CPU, but the clock is supplied to Interrupt, timer and serial port blocks.

→ Idle mode can be terminated by interrupt (or) by hardware reset.

PD : Power down mode

PD = 1, Controller driven to power down mode.

During power down mode, the internal oscillator is stopped and the content of SFR and internal RAM are preserved.

when the controller remains in power down mode the Vcc can be reduced to 2V.

GF1 & GF0 : General purpose flag bit.

Used by the programmer to indicate the status of certain events during program execution.

Serial mode baudrate selection: (SMOD)

SMOD is used to decide the baud rate in serial port operation modes 1, 2, or 3.

mode 2:

SMOD = 0, baud rate = $1/64$ fosc.

= 1, baudrate = $1/32$ x fosc.

Serial data Buffer Register (SBUF)

→ It is used to hold the parallel data during transmission and reception.

→ SBUF Internal address = 99.

serial port control Register: (SCON)

D_7							D_0
SM_0	SM_1	SM_2	REN	TB8	RB8	TI	DI

SM_0	SM_1	
0	0	– Mode 0
0	1	– Mode 1
1	0	– Mode 2
1	1	– Mode 3

SM_2 = serial mode bit 2

RI = Receive Interrupt Flag

TI = Transmit Interrupt Flag.

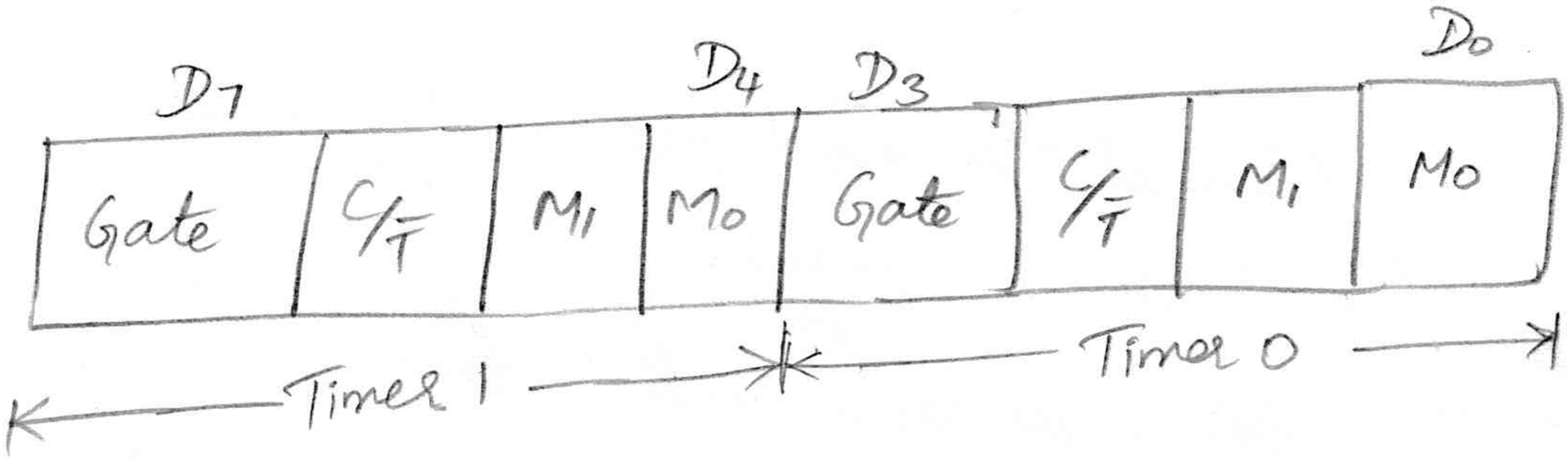
RB8 = Received 9th Bit (ie) D8 of Received data)

TB8 = Transmitted 9th Bit (ie) D8 of transmitted data)

Timer mode control (TMOD) Register :

→ TMOD register is used to select the operating mode and timer/counter operation of timers.

→ The lower 4-bits of TMOD is used to control timer-0 and upper 4 bits are used to control timer-1.



M_1 M_0 – Mode selection

M_1	M_0	– operating mode
0	0	– Mode 0
0	1	– mode 1
1	0	– mode 2
1	1	– Mode 3

$C/\overline{T}$ – operation

0 – Timer
1 – counter

Gate – clock

0 – Internal clock
1 – External clock.

Timer Control (TCON) Register :

TCON register consits of timer overflow flags, timer run control bits, external interrupt and External interrupt type control bits.

TF1	TR1	TF0	TR0	IE1	IT_1	IE_0	IT_0

TF_1 – Timer 1 overflow flag.

set by hardware when timer/counter 1 overflow cleared by hardware as the processor vectors to interrupt service routine.

TR1 - Timer 1 run control bit

Set/cleared by software to turn timer/counter on/off.

TF0 - Timer 0 overflow flag.

Set by hardware when timer/counter 0 overflows.

Cleared by hardware as the processor vectors to Interrupt service routine.

TR0 - Timer 0 run control bit

Set/cleared by software to turn timer/counter ON/OFF.

$IE_{I/0}$ - External Interrupt I/O edge

Set by CPU when (H to L transition) detected on external interrupt.

cleared when Interrupt is processed.

$IT_{I/0}$ - Interrupt I/O type control bit.

Set/cleared by software to specify falling edge/low-level triggered external Interrupt.

Interrupt Enable (IE) Register:

IE register is used to enable/disable the Interrupt of 8051.

EA - Enable all the interrupts.

D_7							D_0
EA	X	ET_2	ES	ET_1	EX_1	ET_0	EX_0

EA = 1, Each Interrupt source is individually Enabled (or) disabled by setting (or) clearing the enable bit.

EA = 0, disable all the interrupts.

ET_2 - Enables (or) disables time 2 overflow (or) capture Interrupt.

ES - Enables (or) disables serial port Interrupt.

ET_1 - Enables (or) Disables timer 1 overflow Interrupt.

EX_1 - Enables (or) disables External interrupt 1.

ET_0 - Enables (or) disables Timer 0 overflow Interrupt.

EX_0 - Enables (or) disables External Interrupt 0.

Interrupt priority (IP) Register :-

→ 8051 has five interrupts, three internal and two External Interrupts.

→ IP register can be programmed to make the priority of any of the Interrupt as highest.

-	-	PT_2	PS	PT_1	PX_1	PT_0	PX_0

PT_2 - Defines the timer 2 Interrupt priority level (8052 only).

PS - Defines the serial port Interrupt priority level.

PT_1 - Defines the Timer 1 Interrupt → priority level.

PX_1 - Defines External Interrupt 1 priority level.

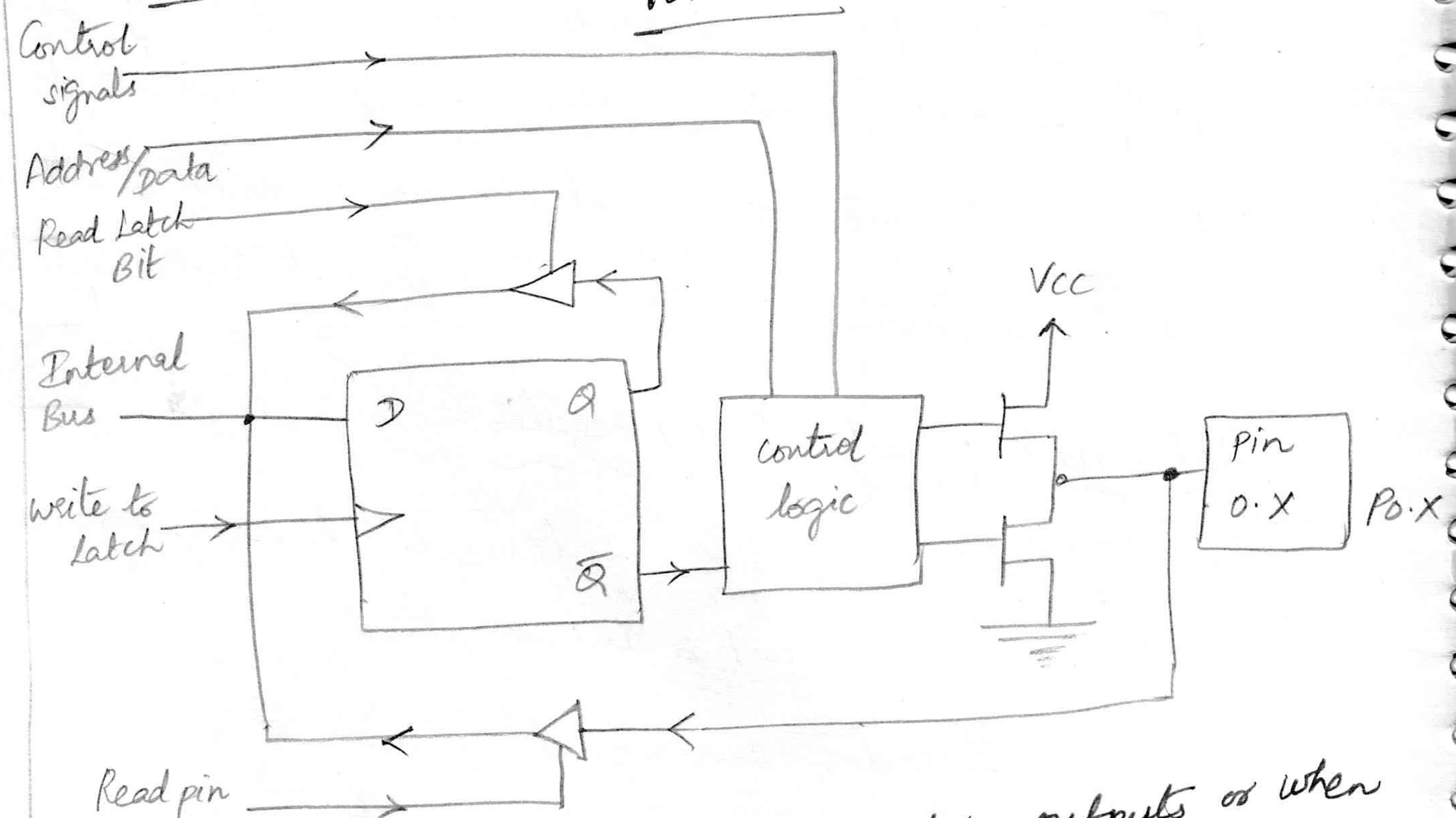
PT_0 - Defines the timer 0 Interrupt priority level.

PX_0 - Defines the external Interrupt 0 priority level

15

* I/O Pins Ports & circuits :-

Port operation



→ port 0 pins may serve as inputs, outputs or when used together, as a bi-directional low order address and data bus for external memory.

→ when a pin is to be used as an input a 1 must be written to the corresponding port 0 latch by the program, thus turning both of the output transistors off which causes the pin to float in a high impedance

state and the pin is essentially connected to the input buffer.

→ when used as an output, the pin latches are programmed to a 0 will turn on the lower FET, grounding the pin. All the latches that are programmed to a 1 still float, thus external pull up resistors will be needed to supply a logic ~~w~~high when using port 0 as output.

→ when port 0 is used as an address bus to external memory, internal control, signal SWI the address lines to the gates of FET.

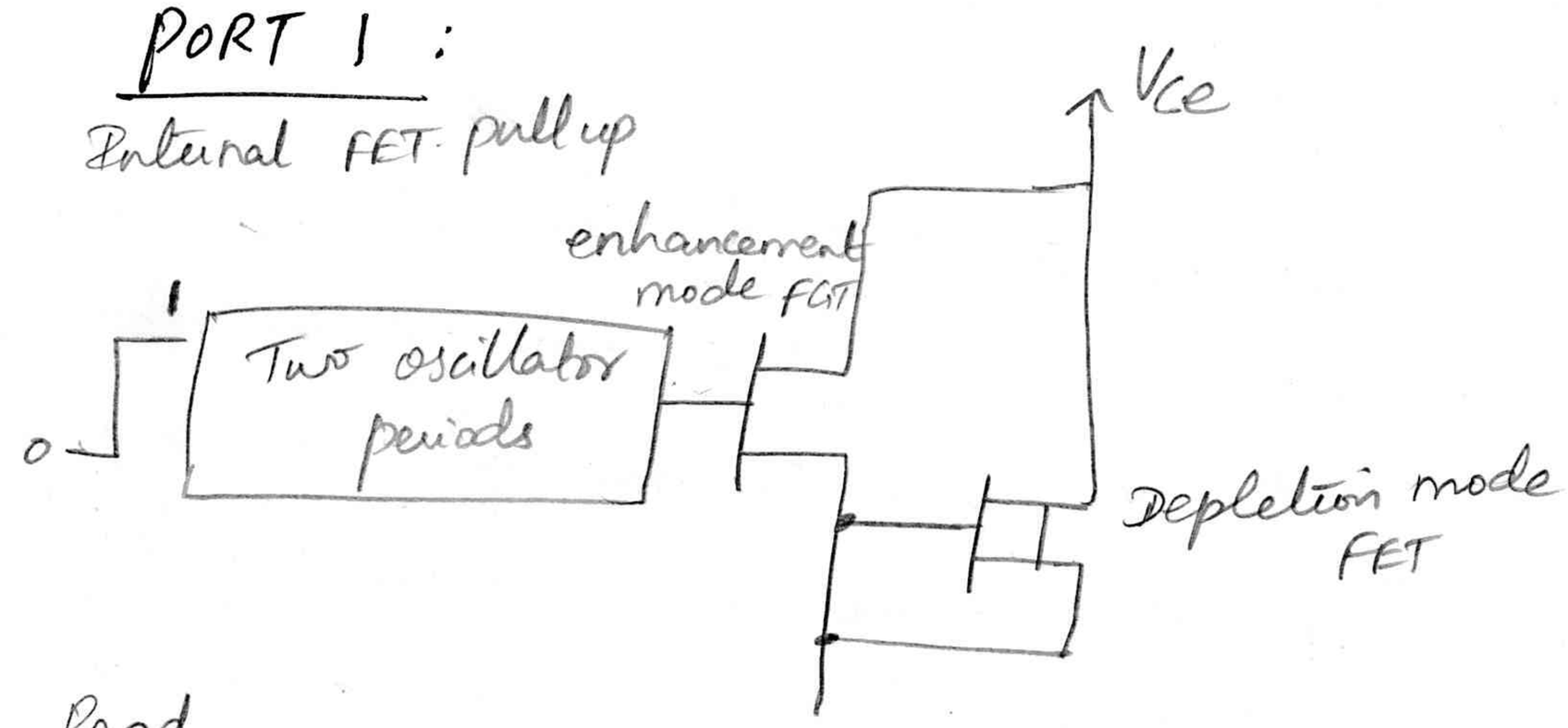
→ Logic 1 on the address bit will turn on upper FET and turn off lower FET to provide a logic high at the pin.

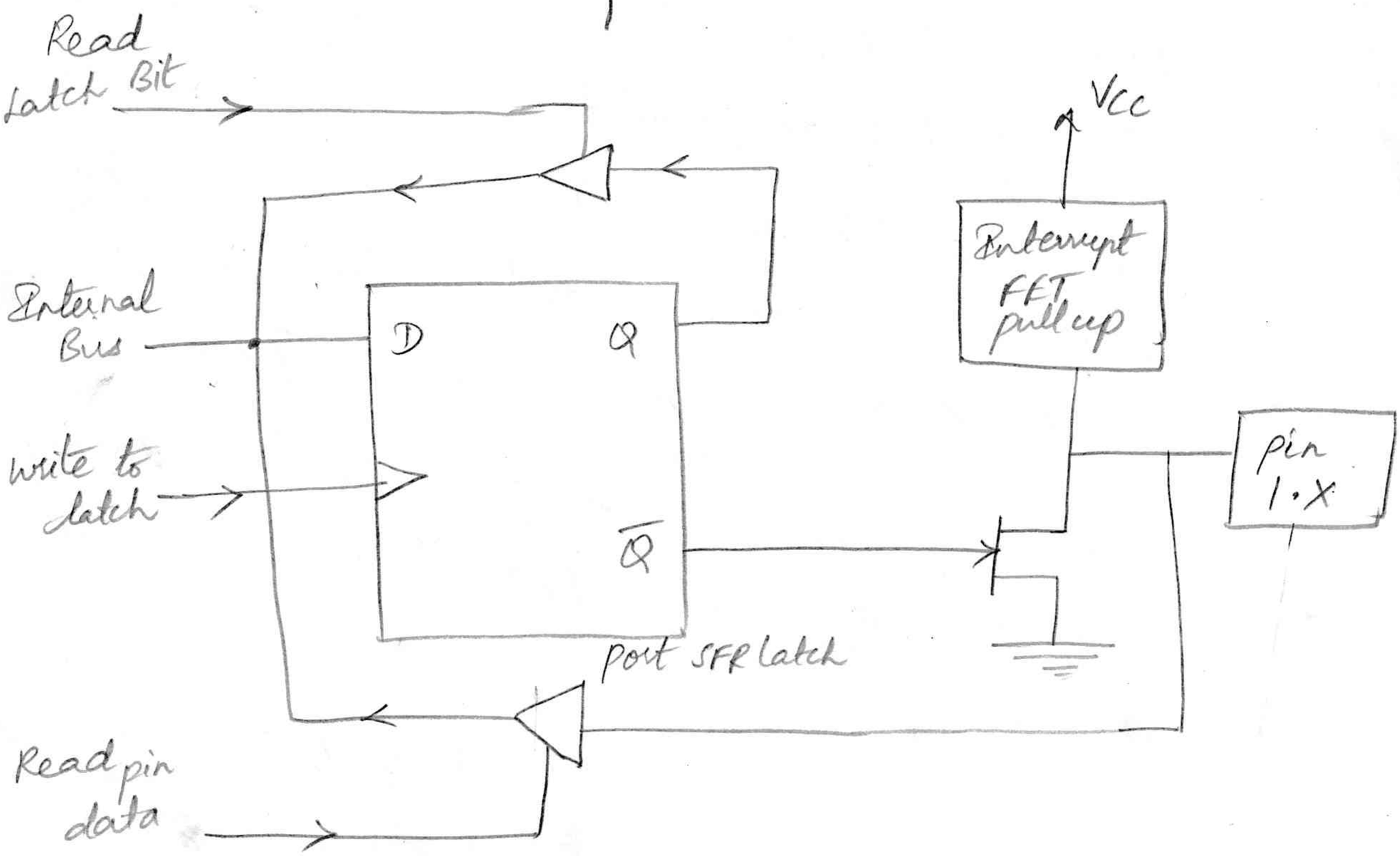
→ Logic 0 on the address bit will turn on lower FET and upper FET off to provide logic low at the pin.

→ After the address has been formed and latched onto external circuits by the ALE pulse, bus is turned around to become a data bus.

→ port 0 now reads data from the external memory and must be configued as an input, so a logic 1 is automatically written by internal control logic to all port 0 latches.

PORT 1 :





→ Port 1 have no dual functions

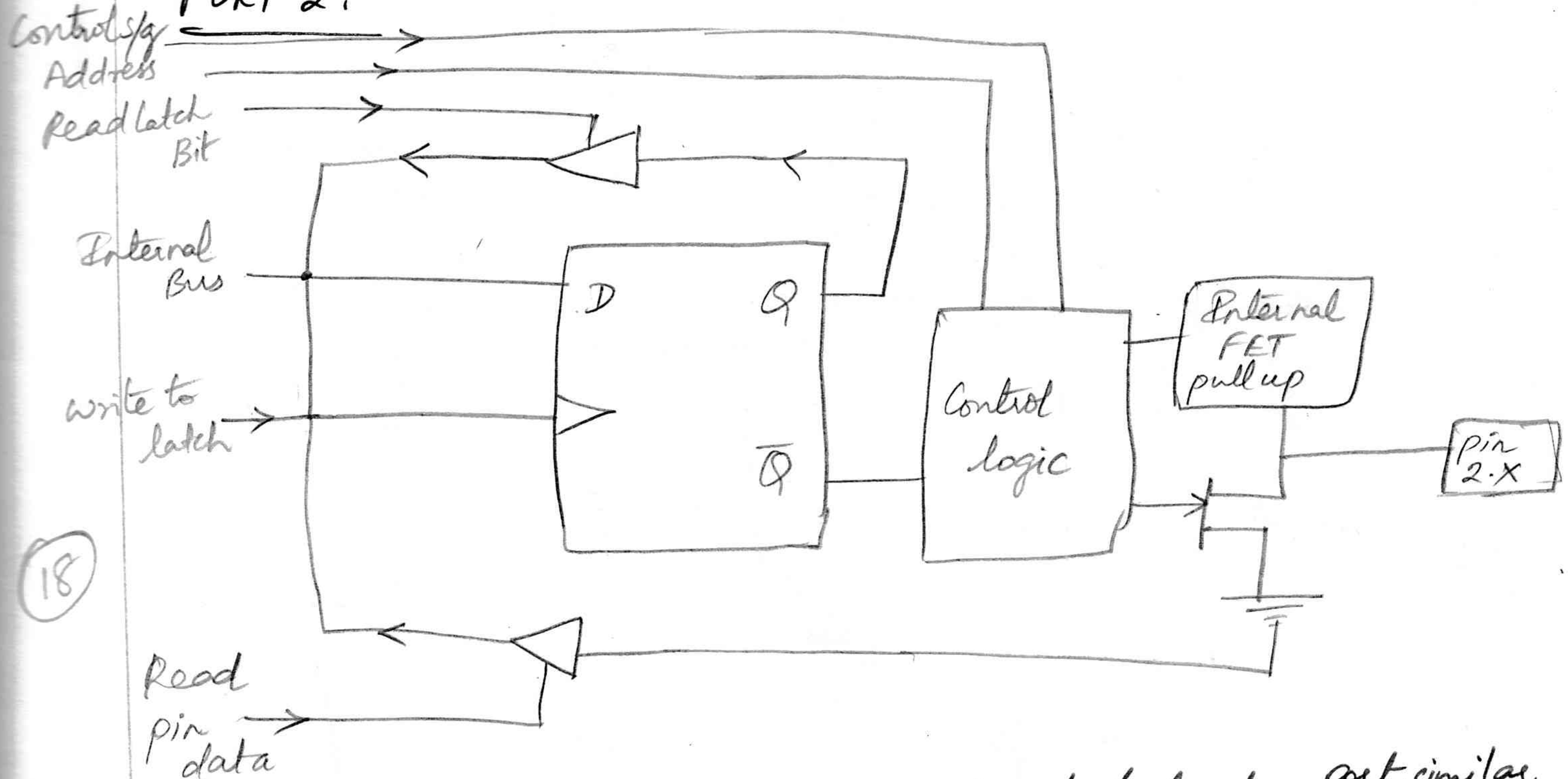
→ output latch is connected directly to the gate of the lower FET, which has an active pull up load.

→ used as an input, a 1 is written to the latch, turning the lower FET off, the pin and the input to the pin buffer are pulled high by the FET load.

→ used as an output, the latches containing a 1 can drive the input of an external circuit high through the pullup.

→ If '0' is written to the latch, the lower FET is ON, the pull up is off & the pin can drive the input of the external circuit low.

PORT 2:-



→ Port 2 may be used as an input/output. port similar in operation to port 1.

→ Alternate use of port 2 is to supply a high order address byte in conjunction with the port 0 lower byte to address external memory.

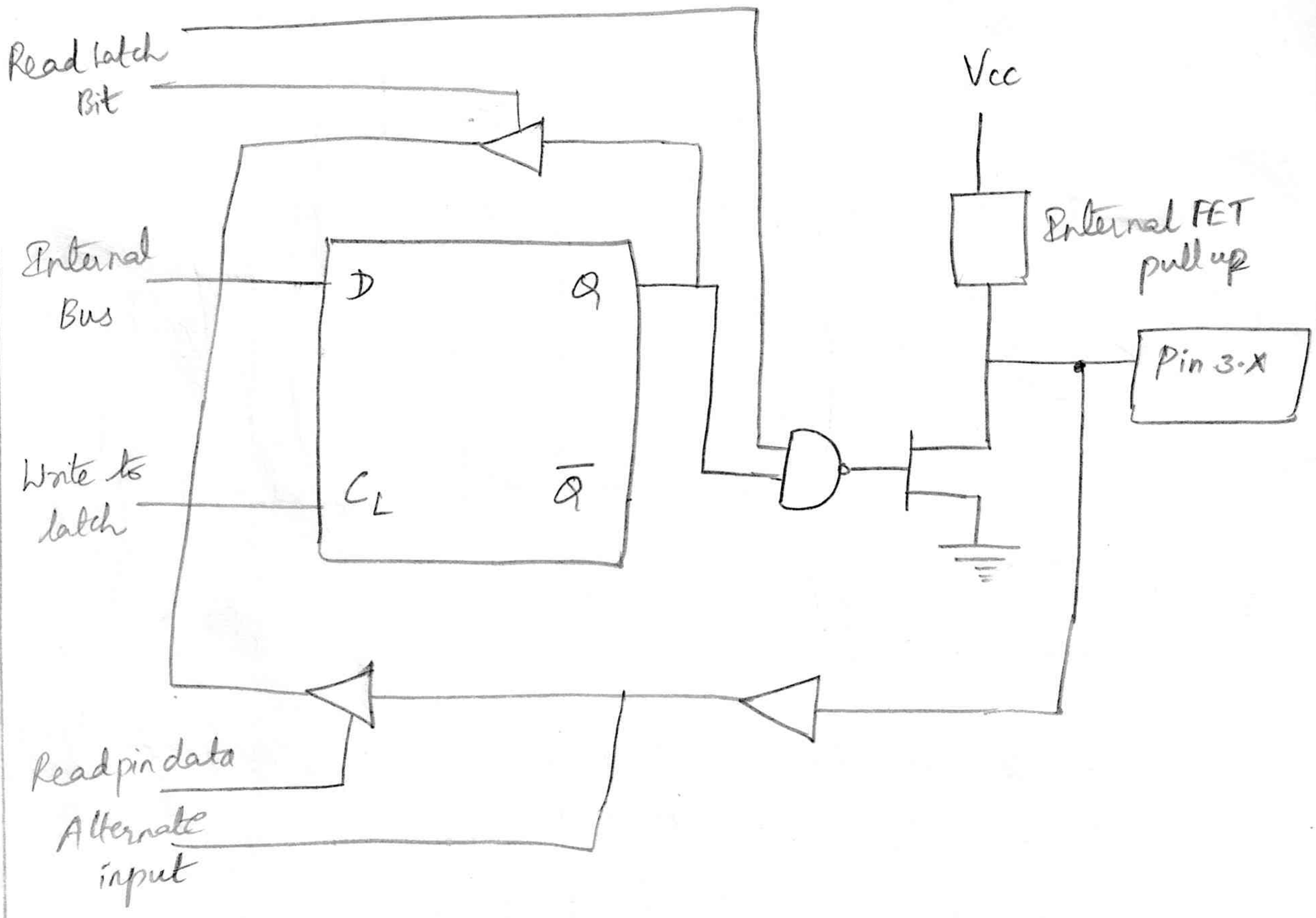
→ Port 2 pins are momentarily changed the address, control signals when supply the high byte of a 16-bit address.

→ Port 2 latches remain stable when external memory is addressed, as they do not have to turned around for data input as in the case of port 0.

PORT 3:

→ Port 3 is an input/output port similar to port 1.

→ The input and output functions can be programmed under the control of P3 latches (or) under the control of various other special function Register.



PIN	Alternate use	SFR
P3.0 – RXD	Serial data input	SBUF
P3.1 – TXD	Serial data output	SBUF
P3.2 – $\overline{INT0}$	External interrupt 0	TCON.1
P3.3 – $\overline{INT1}$	External interrupt 1	TCON.3
P3.4 – T0	External timer 0 input	TMOD
P3.5 – T1	External timer 1 input	TMOD
P3.6 – $\overline{WR}$	External memory write pulse –	
P3.7 – $\overline{RD}$	External memory read pulse –	

→ unlike ports 0 and 2, port 3 can have external addressing functions and change all eight port bits when in alternate use, each pins of port 3 may be individually programmed to be used either as I/O or as one of the alternate functions.

✱ INSTRUCTION SET

The following are 8051 Instruction set,

i) Data transfer Instruction
ii) Arithmetic Instruction
iii) Logical Instruction
iv) Boolean variable manipulation Instruction
v) Program & machine control instructions.

i) Data transfer Instruction:

a) MOV dest-byte, source byte:

* copies a byte from the source location to the destination.

eg:
MOV A, #data
MOV A, Rn ; n = 0 to 7
MOV A, @Ri ; i=0 (or) 1
MOV direct, #data
MOV direct, direct

b) MOV dest-bit, source bit:

* This instruction copies the source bit to the destination bit.

* one of the operand must be carry flag.

MOV P1.2, C
copy carry bit to port bit P1.2.

MOV C, P2.5
copy port bit P2.5 to carry bit.

c) MOV DPTR, #16 bit value:

This instruction loads the 16-bit DPTR register with a 16 bit immediate value.

eg:

MOV DPTR, #456F H

d) MOVC A., @A+DPTR

* This instruction moves a byte of data located in program (code) ROM into register A.

* Address of the desired byte in the code space (on-chip ROM) is formed by adding the original value of the accumulator to the 16-bit DPTR register.

e) MOVC A, @A+PC

* This instruction moves a byte of data located in the program (code) area to A.

* Address of the desired byte of the data is formed by adding the program counter (PC) register to the original value of the accumulator.

f) MOV X dest byte, source byte.

* This instruction transfers the data between external memory & register A.

eg:-

MOVX A, @DPTR

This moves into the accumulator, a byte from external memory whose address is pointed by DPTR.

g) MOV X @Ri, A.

* This moves a byte from register A to an external memory location whose 8-bit address is held by R_0 or R_1.

h) PUSH direct

* It copies the indicated byte onto the stack and increments SP by 1.

eg

PUSH A } illegal
PUSH R3 }

PUSH 0E0H } valid.
PUSH 03H }

i) POP direct.

* It copies the byte pointed to SP location whose address is indicated & decrements the SP by 1.

eg:

POP 0E0H

POP 03H

j) SWAP A

* This instruction interchanges the lower nibble with the upper nibble inside register A.

k) XCH A, Byte.

* This instruction swaps the contents of register A and the source byte.