EICoM
ELECTRICAL COMPUTER MECHATRONICS

انظمة معالجات دقيقة

د. اسلام ملكاوي

للطالبة المبدعة
فلسطين حمدان

إرادة - ثقة - تغيير

(31) HOLD, (30) HOLA : two Signal support direct memory access ~~address~~ interface (external divice want to access directly in main memory without going through microprocessor

usually any external divice want to read or write data from memory, so it should take data from microprocessor

مثلا بدي اقرأ الداتا من address بروح للمايكروبروسيسور بجيب الداتا منها؟ وبعطيها لل external divice وبحطها د registe

في طريقة تانية انه ال external divice ~~microprocessor~~ بسأل ال microprocess ← give me data bus انه انا بروح ع ال memory بكتب وبقرأ لحالي

External device ask to permition access (direct memory address) directly from main memory ايام بدون ما انت تخزن عندك صار بنفسه

HOLD input (بعطيني ال data bus)

HOLA hold Acknolg ال micro برد عليه
(replay) → H موافق
→ L : رفض (بكون ال microp ماله بدو access for memory)

28 determine type of communication → with memory or → with I/O
[اذا ال micro بدو access الميموري يكون Communicat مع ال memory] H
→ [اذا بدو connect مع I/O Communicat مع I/O] L
عكسها بال 88

27 data transmite recive determine direction of data (transmited outside the microprocessor or recive it in microprocessor)
H → transimt خارج المايكرو
L → recive داخل ال micro

ال data bus يكون by directional (output, input) عكس ال address bus تبعا output لانه بيطلع من ال micro لل memory

26 data enable → يحدد انه ال multiplex line between address and data → carry direct data Information

multiplex يعني ← they can perform two task or operation either cary address information or world data

يعني كيف بدي اعرف اذا ال bit الموجودة ع ال multiplex line اذا كانوا data or address ← عن طريق control signal DEN وال ALE (address latch enable)

DEN L → direct data Information
ALE H → معناه ال multipler بين ال address وال data وال status
] لازم واحد منهم يكون active

cary Valid addres information ← 20 bit.

18 INTR, 24 $\overline{INTR}$, 17 NMI, 23 test, 21 reset } 5 control signal belonge to Interrept Interface [interrupter ~~line~~ interface using 5

mause, printer, ...

(susbend operation) وقف الشغل give me your attenchen اعطيني انتباهك

(18) External device is request an interupt from microprocessor → ويرد عليه ال micro عن طريق (24)

(17) Nonmastable (Cant be mask by Interupt request), يعني اذا اجى عندي 2 intrupt واحد INTR و NMI رح يرد ع ال NMI لانه ال priority اله اكبر
يكون عندي مشاكل داخلية internal او خارجية

(23) susbend of microprocessor operation and wating to recive Interupt request

مثلا عندي عليه بحاجة ال input ~~[illegible]~~ من External device بعمل test يعني suspend لحد ما يوصل ال External

(21) reset (بعمل restart لل microp يعني بعمل Insialation to Initial)

ال priority
reset > NMI > TEST > INTR

(22) Ready : بكون ال (0) اذا كانت بال waiting state
order to inform microprocessor mean ready to complet operation ← input signal from external device

(19) clk : Conect with clk signal (pulses) determine speed of microprocessor

على ايا Freq ان بيشتغل ، بأي بيجالي ال clk ← عباره عن other IC
called clk generater

كل ما زادت ال freq تبعت ال clk signal كل ما زاد سرعه ال microprocessor رفعتهم بشتغل ع (5 MHZ

* الاسماء الموجودين بين ال IC بالاقواس هيسلب يكونوا active in MAX mode operation , عنا 2 mode (Min, MAX)

use functionality writen ~~between~~ inside pin ← (Min, MAX) → use functionality ~~between~~ writen between in barkesis ( )

لي مابنكون بين اقواس بتكون Common بين ال MAX , ال Min عشان 31 Hold ← min
(RQ̄ , GT̄0) ← MAX
number zero

30 ← min HolA
1 (RQ̄/GT̄1) MAX

29 ← min write , 24 — 28 ال MAX ← status
lock MAX غير ال Common

---

slide (12)

20 bin support for address bus because the size of main memory 1 M

First 8 address / Data ⇐⇒ input, output / Data ، address ب
next 8 single address ⇒ output
Final 4 multiplex with status

ALE → enable address bus ، carry vaild ← address information H
يعرف ال data الموجوده ع ال 20 b.t ، or not ← L

بيتعامل وحدة قصد [(Enable data bus) يلي بتشبهها ال DEN ← L ← multiplex line between carry valid data information

موجوده بس بالـ 88

SSO → system status output بتحدد نوع الداتا يلي بدي اقراها من الميموري Code or normal data (0 / 1)

IO/M → determine type of communication (inp/out, Memory)

DT/R → determine direction of data within transimate outside microprocessor or recive inside microp.
الفائدة بحدد ال direction of data bus (bidirectional) ~ اذا انه input او output
recive transimate

RD, WR : read data or write data

Ready : give start information of external device with to complete operation (H) or waite (busy) (L)

CPE 408330
Assembly Language and Microprocessors

Chapter 8: THE 8088 AND 8086 MICROPROCESSORS AND THEIR MEMORY AND INPUT/OUTPUT INTERFACES

[Computer Engineering Department,
Hashemite University, © 2008]

Lecture Outline

- 8.1 The 8088 and 8086 Microprocessors
- 8.2 Minimum-Mode and Maximum-Mode System
- 8.3 Minimum-Mode Interface
- 8.4 Maximum-Mode Interface
- 8.5 Electrical Characteristics
- 8.6 System Clock
- 8.7 Bus Cycle and Time States
- 8.8 Hardware Organization of the Memory Address Space

Lecture Outline

- 8.9 Memory Bus Status Codes
- 8.10 Memory Control Signals
- 8.11 Read and Write Bus Cycles
- 8.12 Memory Interface Circuits
- 8.13 Programmable Logic Arrays
- 8.14 Types of Input/Output
- 8.15 An Isolated Input/Output Interface
- 8.16 Input/Output Data Transfer
- 8.17 Input/Output Instructions
- 8.18 Input/Output Bus Cycles

8.1 The 8088 and 8086 Microprocessors

□ The 8086, announced in 1978, was the first 16-bit microprocessor introduced by Intel Corporation.

□ 8086 and 8088 are internally 16-bit MPU. However, externally the 8086 has a 16-bit data bus and the 8088 has an 8-bit data bus.

8.1 The 8088 and 8086 Microprocessors

□ 8086 and 8088 both have the ability to address up to 1 Mbyte of memory and 64K of input/output port.

□ The 8088 and 8086 are both manufactured using *high-performance metal-oxide semiconductor* (HMOS) *technology.*

□ The 8088 and 8086 are housed in a 40-pin dual inline package and many pins have multiple functions.

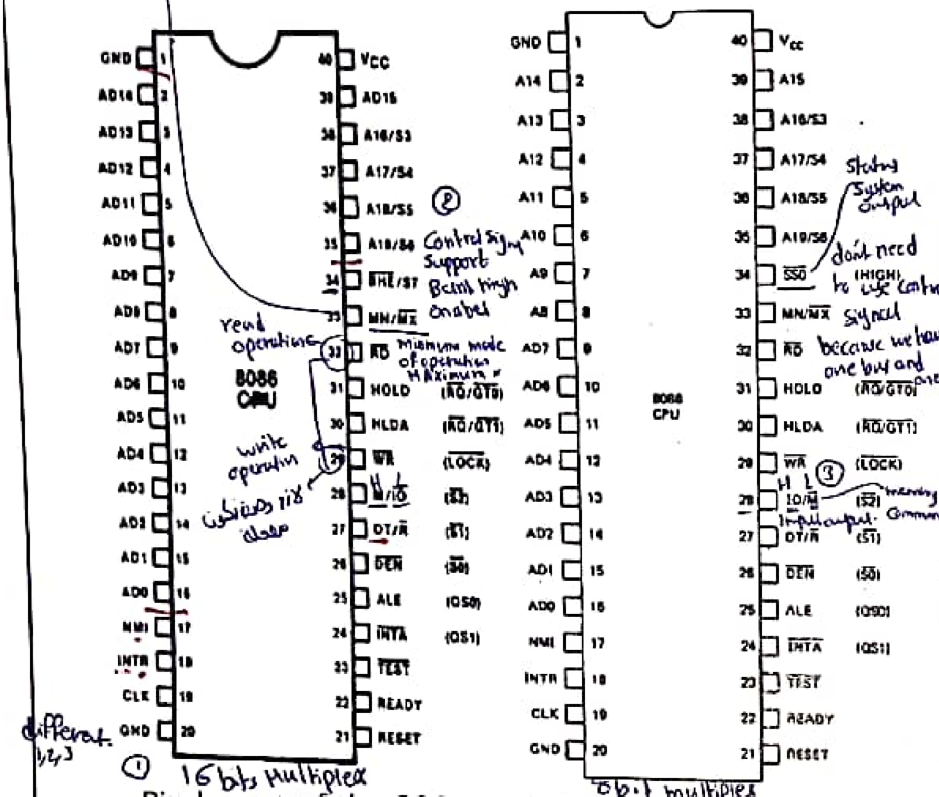
8.1 The 8088 and 8086 Microprocessors

□ **CMOS, Complementary Metal-Oxide-Semiconductor, is a major class of integrated** circuits used in chips such as microprocessors, microcontrollers, static RAM, digital logic circuits, and analog circuits such as image sensors.

□ Two important characteristics of CMOS devices are high noise immunity and low static power supply drain. Significant power is only drawn when its transistors are switching between on and off states; consequently, CMOS devices do not produce as much heat as other forms of logic such as TTL. CMOS also allows a high density of logic functions on a chip.

minimum my System Contain one single microprocessor
Maximum my System Contain Multiple signal microprocessor (sharing global recorses) data bus, main memory
and each microprocessor have local recorce (Register, Cash)

8.1 The 8088 and 8086 Microprocessors



Pin layout of the 8086 and 8088 microprocessor

between address and data 16bit (Data bus)

between address and data 8bit (data bus)

- Pin functions
- Most pins are independent and serve a single function
- Examples:

CLK—clock

INTR—interrupt request

READY—bus ready

- Some multi-functions pins- different times/different mode
- Examples:

AD0–AD15– multiplexed address/data lines at different times

A16/S3—multiplexed address and status line at different times

IO/M* or S2* Control line in one mode or bus

8.2 Minimum-Mode and Maximum-Mode Systems

☒ The 8086 and 8088 microprocessors can be configured to work in either of two modes:

- ☒ The minimum mode - MN/MX'=1 my system Small Consist only on microprocess
- ☒ The maximum mode - MN/MX'=0 my system large Contain multiple microprocess

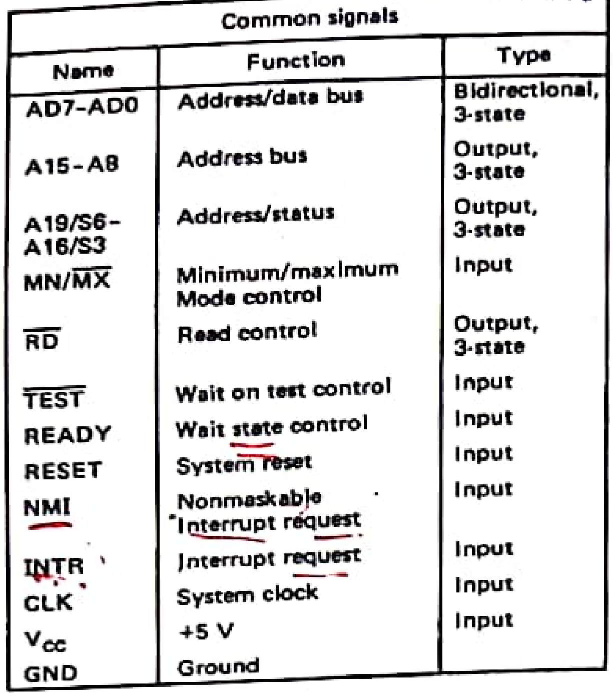
☒ The mode selection feature lets the 8088 or 8086 better meet the needs of a wide variety of system requirement.

□ Minimum mode 8088/8086 systems are typically smaller and contain a single processor.

□ Depending on the mode of operation selected, the assignment for a number of the pins on the microprocessor package are changed.

8.2 Minimum-Mode and Maximum-Mode Systems

in Min and MAX



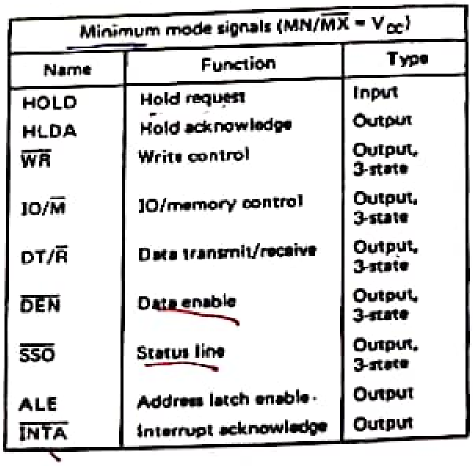
Common signals		
Name	Function	Type
AD7-AD0	Address/data bus	Bidirectional, 3-state
A15-A8	Address bus	Output, 3-state
A19/S6- A16/S3	Address/status	Output, 3-state
MN/$\overline{MX}$	Minimum/maximum Mode control	Input
$\overline{RD}$	Read control	Output, 3-state
$\overline{TEST}$	Wait on test control	Input
READY	Wait state control	Input
RESET	System reset	Input
NMI	Nonmaskable Interrupt request	Input
INTR	Interrupt request	Input
CLK	System clock	Input
V_{CC}	+5 V	Input
GND	Ground	

(a)

- 8088 signals/pins categorized as
  - Common—same function both modes
    Examples: Pin 9 (AD_7)- pin 16 (AD_0)
  - Minimum Mode—special minimum mode operations
    Examples: pins 26-28 are DEN*, DT/R*, and IO/M*
  - Maximum Mode—special maximum mode operations
    Example: pins 26-28 are S_0*, S_1*, and S_2*

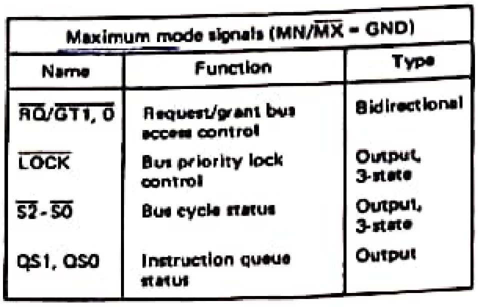
- Signals common to both minimum and maximum mode

8.2 Minimum-Mode and Maximum-Mode Systems



Minimum mode signals (MN/$\overline{MX}$ = V_{CC})		
Name	Function	Type
HOLD	Hold request	Input
HLDA	Hold acknowledge	Output
$\overline{WR}$	Write control	Output, 3-state
IO/$\overline{M}$	IO/memory control	Output, 3-state
DT/$\overline{R}$	Data transmit/receive	Output, 3-state
$\overline{DEN}$	Data enable	Output, 3-state
$\overline{SSO}$	Status line	Output, 3-state
ALE	Address latch enable	Output
$\overline{INTA}$	Interrupt acknowledge	Output

(b)



Maximum mode signals (MN/$\overline{MX}$ = GND)		
Name	Function	Type
$\overline{RQ}$/$\overline{GT1, 0}$	Request/grant bus access control	Bidirectional
$\overline{LOCK}$	Bus priority lock control	Output, 3-state
$\overline{S2}$-$\overline{S0}$	Bus cycle status	Output, 3-state
QS1, QS0	Instruction queue status	Output

(c)

(b)Unique minimum-mode signals (c) Unique maximum-mode signals

8.2 Minimum-Mode and Maximum-Mode Systems

- EXAMPLE

  Which pins provide different signal functions in the minimum-mode 8088 and minimum-mode 8086?

① # of multiplex line between address and data
8086 16
8088 8

- Solution:

(a) Pins 2 through 8 on the 8088 are address lines A_{14} through A_8, but on the 8086 they are address/data lines AD_{14} through AD_8.

(b) Pin 28 on the 8088 is IO/M' output and on the 8086 it is the M/IO' output.

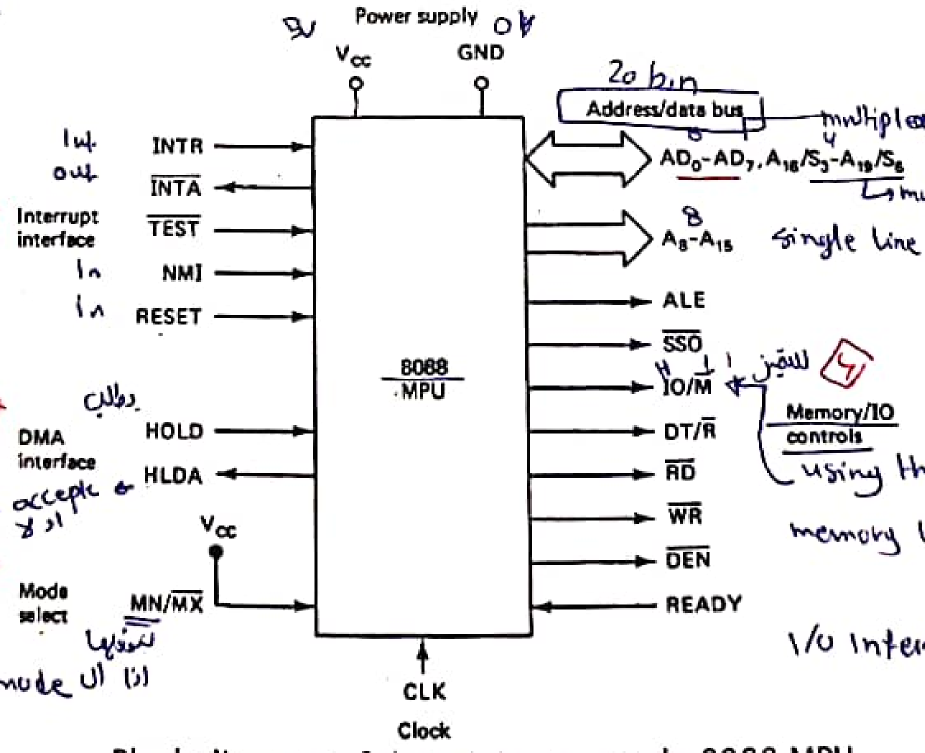
(c) Pin 34 of the 8088 is the SSO' output, and on the 8086 this pin supplies the BHE'/S_7.

② control Bank high enable in 8086 in order to enable high part in data bus and high bank in memory ,
في ال 8088
SSO بيشتغل صح

③ control signal $\overset{high}{\text{Memory}}/\overset{low}{\text{IO}}$ 8086
$\overset{L}{\text{memory}}/\overset{H}{\text{IO}}$ 8088

8.3 Minimum-Mode Interface

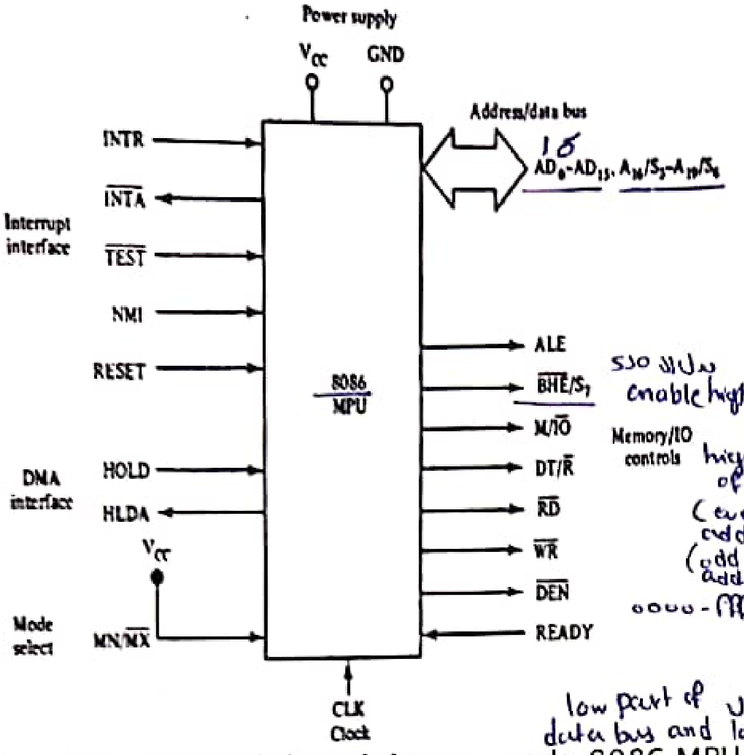
Block digram



Block diagram of the minimum-mode 8088 MPU

multiplex line between address and data
multiplier line Address and status
single line of address
using the same signal
memory interface: communication between microp and main memory
I/O interface: communication between microp and IO
direct memory address
بناءً انو المايكرو
بردعليها بال HLDA
اذا ال minimum mode

8.3 Minimum-Mode Interface–Differences



Block diagram of the minimum-mode 8086 MPU

بدل الـ SSO

enable high part of data bus and (8-15)

high bank of memory

(even address)

(odd address)

0000-MPP

low part of data bus and low bank of memory

يفعل الـ

- Data bus
  - 16-bit wide
  - $D_{15}-D_0$
  - Multiplexed with A_{15} through A_0
  - Allows 3 types of data transfers
    - Word—over $D_{15}-D_0$
    - Low byte—over D_7-D_0
    - High byte—over $D_{15}-D_8$
- Memory/IO Controls
- SSO* → BHE* (bank high enable)
  - Used to signal external circuitry whether or not a byte transfer is taking place over the upper 8 data bus lines
- A0 now does the same for a byte transfer over the lower 8 data bus line

8.3 Minimum-Mode Interface

□ The minimum-mode signals can be divided into the following basic groups:

1. Address/Data bus
2. Status signals
3. Control signals
4. Interrupt signals
5. DMA interface signals

8.3 Minimum-Mode

□ Address/Data bus

- □ The address bus is used to carry address information to the memory and I/O ports.
- □ The address bus is in 8086-8088 20-bit long and consists of signal lines A_0 through A_{19}.

ازا بدو عنوان main memory بستخدم 20

- □ A 20-bit address gives the 8088 a 1 MByte memory address space.
- □ Only address line A_0 through A_{15} are used when addressing I/O. This give an I/O address space of 64 Kbytes. $\log_2 64K = \boxed{16}$ $6+10$

ازا بدو I/O بستخدم اول 16 من ال 20

88

اول 16 ← multiplex with data

اخر 4 ← & status

- □ The 8088 has 8 multiplexed address/data bus lines (A_0~A_7) while 8086 has 16 multiplexed address/data bus lines (A_0~A_{15}).

8.3 Minimum-Mode Interface

S6 S5 S4 S3

□ Status signals

- □ The four most significant address, A_{19} through A_{16} are multiplexed with *status signal S_6 through S_3.*
- □ Bits S_4 and S_3 مع together form a 2-bit binary code that identifies which of the internal segment registers was used to generate the physical address. S_5 is the logic level of the internal interrupt flag. S_6 is always at the 0 logic level.

غير مستخدم دائما zero لان ~~logic status~~

enable intrupt from external device

نفسه قيمه I flag داخل ال Flag register (one bit)

0 → request ايلي من ال external بده يتجاهلو ignore

بالاساس كنت طالع ال logical بدي ال logical اطلع ال physical

بتكون من 2 segment base address offset وبيكون 16

لبين استخدمنا ال logical لانه ماعندي مكان store 20 bit لانه الكل size 16

shift base + offset = physical

حفظ

S_4	S_3	Address Status
① 0	0	Alternate (relative to the Extra ES segment)
② 0	1	Stack (relative to the stack SS segment)
③ 1	0	Code/None (relative to the code CS segment or a default of zero)
④ 1	1	Data (relative to the Data DS segment)

↳ indeate whitch segment rigester I use to generate physical address

8.3 Minimum-Mode Interface

□ Control signals

منفذة جميع نوع
interface ال

□ The *control signals* are provided to support the memory and I/O interfaces of the 8088 and 8086.

- ALE – Address Latch Enable
- IO/M' – IO/Memory (8088) memory Iterface L
- M/IO' – Memory/IO (8086) memory Iterface H
- DT/R' – Data Transmit/Receive (8088/8086)
- SSO' – System Status Output (8088) Just
- BHE' – Bank High Enable (8086) just
- RD' – Read (8088/8086)
- WR' – Write (8088/8086) First 16
- DEN' – Data Enable (8088/8086) First 8 data bus
- READY – Ready (8088/8086) input [Complete byse

8.3 Minimum-Mode Interface

□ Interrupt signals Subend

□ The *interrupt signals* can be used by an external device to signal that it needs to be serviced.

- INTR' – Interrupt Request
- INTA' – Interrupt Acknowledge
- TEST' – Test (can be use to synchronize MPU)
- NMI – Nonmaskable Interrupt
- RESET – Reset (hardware reset of the MPU)

8.3 Minimum-Mode Interface

□ DMA interface signals

□ When an external device wants to take control of the system bus, it signals this fact to the MPU by switching HOLD to the 1 logic level.

□ When in the hold state, signal lines AD_0 through AD_7, A_8 through A_{15}, A_{16}/S_3 through A_{19}/S_6, SSO’, IO/M’, DT/R’, RD’, WR’, DEN’, and INTR are all put into high-Z state.

High embedune value not belong to low rang or high rang

□ The 8088 signals external devices that the signal lines are in the high-Z state by switching its HLDA output to the 1 logic level.

H → 5
L → 0
Z → 2 تق

قبل ما يوافق لازم يعمل

اذا ما بدو يوافق 0=HLDA

ليش بعمل هيك؟ لأنه رح يسلمهم لل External device

8.4 Maximum-Mode Interface

multiple microprocessor

□ The maximum-mode configuration is mainly used for implementing a ***multiprocessor/coprocessor system environment.***

لكن microprocessor اله local خاص فيه فقط

sharing among processor

□ Global resources and local resources

□ In the maximum-mode, facilities are provided for implementing allocation of global resources and passing bus control to other microprocessors sharing the system bus.

Bus Controller:

maney access global resources among microprocessor in the maximum mode operation

(لحتى ما يصير حدث) بين collesion multiple processor لو (لو كلهم ما بدهم ع نفس المكان بنفس الوقت

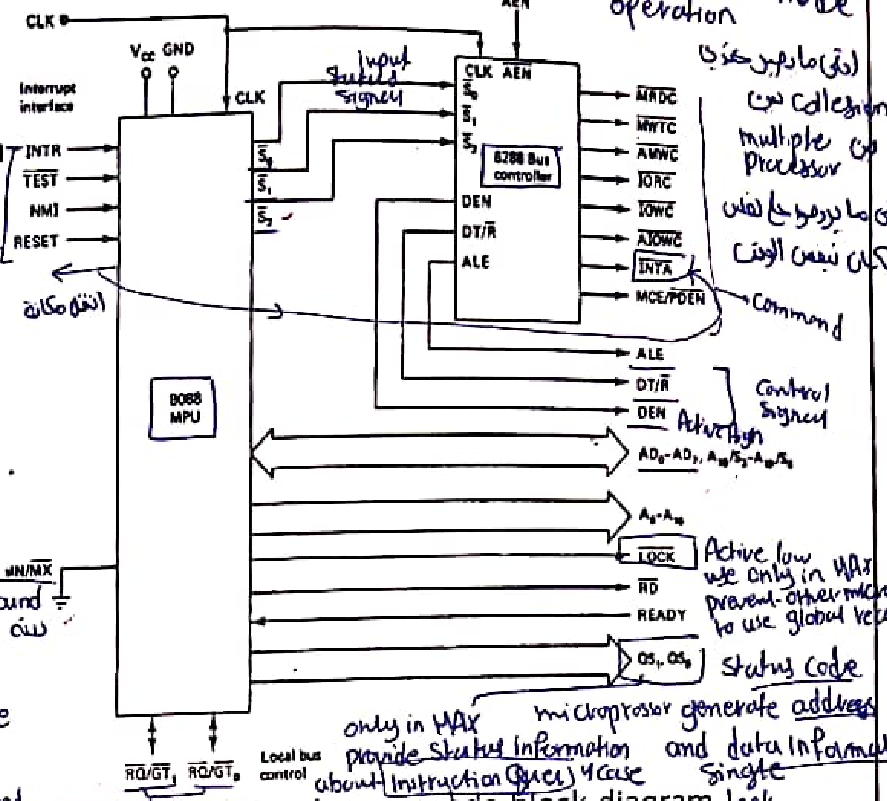
8.4 Maximum-Mode Interface

□ 8288 bus controller: (Coprocessor)

- Maximum-mode configuration
  - MN/MX* pin = 0 → GND
  - Most memory, IO, and interrupt interface outputs produced by an external 8288 bus controller

microprocessor not generated controll signal by it self, it will generate status signal (code) use in controller will be input, base in the value of it one or ~~two~~ command will be generate 3 control processor

* in minmum the microprocessor generate all control signals



input status signal

input

انقل مكانه

Command

Control Signal

Active High

Active low use only in MAx prevent other microprocessor to use global recorses

ground MAx بس

status code

only in MAx provide status information about instruction queue 4 case

microprosor generate address and data information single

lock

request grand 0,1 access local recourse

place store instruction temperory ~~waiting for execution by microprocessor~~

بعمل lock لل other لانه ما بدي حد access for memory or I/O

8088 maximum-mode block diagram

ماعندنا

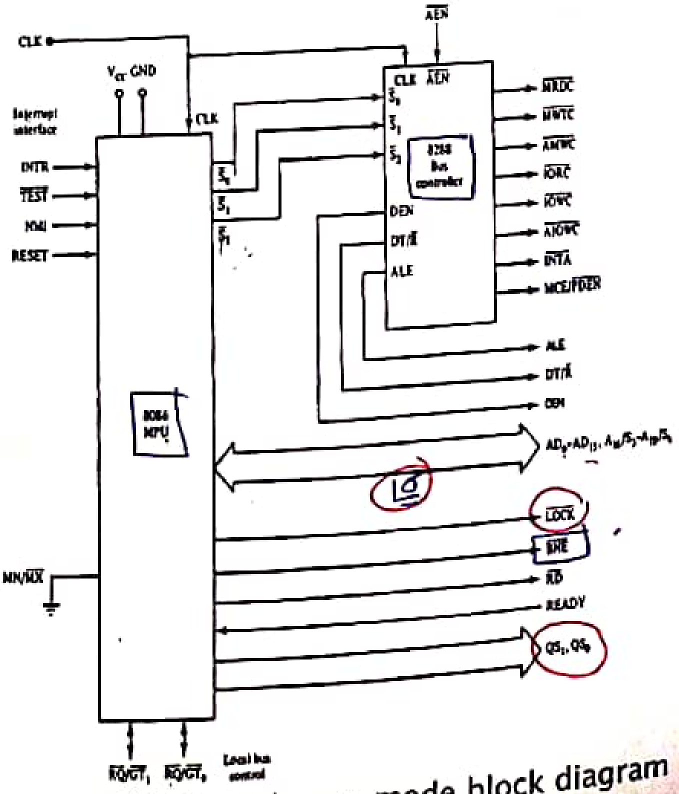
* SSO, M/IO

لأنه ال Command بتعطيهم

8.4 Maximum-Mode Interface

□ 8288 bus controller:

- Differences from 8088 Maximum mode interface
  1. 16-bit multiplexed data bus
  2. BHE* output



8086 maximum-mode block diagram

8.4 Maximum-Mode Interface

□ 8288 bus controller

□ In the maximum-mode, 8088/8086 outputs a status code on three signal line, S_0, S_1, S_2, prior to the initialization of each bus cycle.

□ The 3-bit bus status code identifies which type of bus cycle is to follow and are input to the external bus controller device, 8288.

□ The 8288 produces one or two command signals for each bus cycle.

8.4 Maximum-Mode Interface

□ *8288 bus controller:*

بدو حفظ

	Status Inputs			CPU Cycle	8288 Command
	$\overline{S2}$	$\overline{S1}$	$\overline{S0}$		
1	0	0	0	Interrupt Acknowledge	$\overline{\text{INTA}}$
2	0	0	1	Read I/O Port	$\overline{\text{IORC}}$
3	0	1	0	Write I/O Port	$\overline{\text{IOWC}}$, $\overline{\text{AIOWC}}$
4	0	1	1	Halt	None
5	1	0	0	Instruction Fetch	$\overline{\text{MRDC}}$
6	1	0	1	data from memory Read Memory نفس مبدأ هو يعني ما بنستخدمه	$\overline{\text{MRDC}}$ نفس الشي دايمًا كالعادة [illegible]
7	1	1	0	Write Memory	$\overline{\text{MWTC}}$, $\overline{\text{AMWC}}$
8	1	1	1	Passive	None

Bus status code

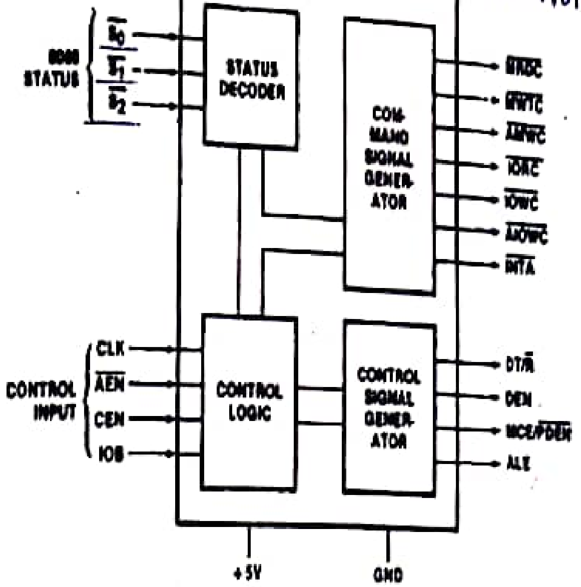
8.4 Maximum-Mode Interface

- 8288 bus controller connection
  - Inputs are codes from the 3-bit bus status lines $S_2*S_1*S_0*$ = bus status code
  - Outputs produced by 8288 instead of 8088
    - Based on bus status code → active 0

      MRDC*= Memory read command
      MWTC*= Memory write command
      AMWC*= Advanced memory write command
      IORC*= I/O read command
      IOWC*= I/O write command
      AIOWC*= advanced I/O write command
  - Produced for all bus cycles

    ALE= Address latch enable
    DT/R*= Data transmit/receive
    DEN= Data enable (complement) Active High
    INTA*= Interrupt acknowledge

Internal architecure for bus controller



Block diagram of 8288

8.4 Maximum-Mode Interface

□ Lock signal

□ The lock signal (LOCK') is meant to be output (logic 0) whenever the processor wants to lock out the other processor from using the bus.

□ Local bus control signals ①

□ The use request/grant signals (RQ'/GT'_0, ② RQ'/GT'_1) provide a prioritized accessory bus access mechanism for accessing the local bus.

8.4 Maximum-Mode Interface

□ Queue status signals

□ The 2-bit queue status code QS_0 and QS_1 tells the external circuitry what type of information was removed form the queue during the previous clock cycle.

know the statuse of queue

QS1	QS0	Queue Status
0 (low)	0	No Operation. During the last clock cycle, nothing was taken from the queue. no fitch, microprocessor not do thing
0	1	fitch First Byte. The byte taken from the queue was the first byte of the instruction.
1 (high)	0	Queue Empty. The queue has been reinitialized as a result of the execution of a transfer instruction.
1	1	fitch second, third — Subsequent Byte. The byte taken from the queue was a subsequent byte of the instruction.

Queue status code

8.4 Maximum-Mode Interface

- EXAMPLE

If the bus status code $S'_2S'_1S'_0$ equals 101, what type of bus activity is taking place? Which command output is produced by the 8288?

- Solution:

Looking at the bus status table, we see that bus status code 101 identifies a read memory bus cycle and causes the MRDC' output of the bus controller to switch to logic 0.

8.5 Electrical Characteristics

- □ Power is applied between pin 40 (Vcc) and pins 1 (GND) and 20 (GND) (pin1 and 20 are connected together).
- □ The nominal value of Vcc is specified as +5V dc with a tolerance of ±10% (4.5V~5.5V will work normal correctly)
  الجهد او اقل ما رح يشتغل منيح
- □ Both 8088 and 8086 draw a maximum of 340mA from the supply.

low
(−0.5 − 0.8)

Symbol	Meaning	Minimum	Maximum	Test condition
V_{IL}	Input low voltage	−0.5 V	+0.8 V	
V_{IH}	Input high voltage	+2.0 V	V_{cc} + 0.5 V	
V_{OL}	Output low voltage		+0.45 V	I_{OL} = 2.0mA
V_{OH}	Output high voltage	+2.4 V		I_{OH} = −400 uA

(2 H − 5.5) ←

ما لازم يتعدى يكون
Max

min

1V → high-Z
لا يعني ال low ولا ال high

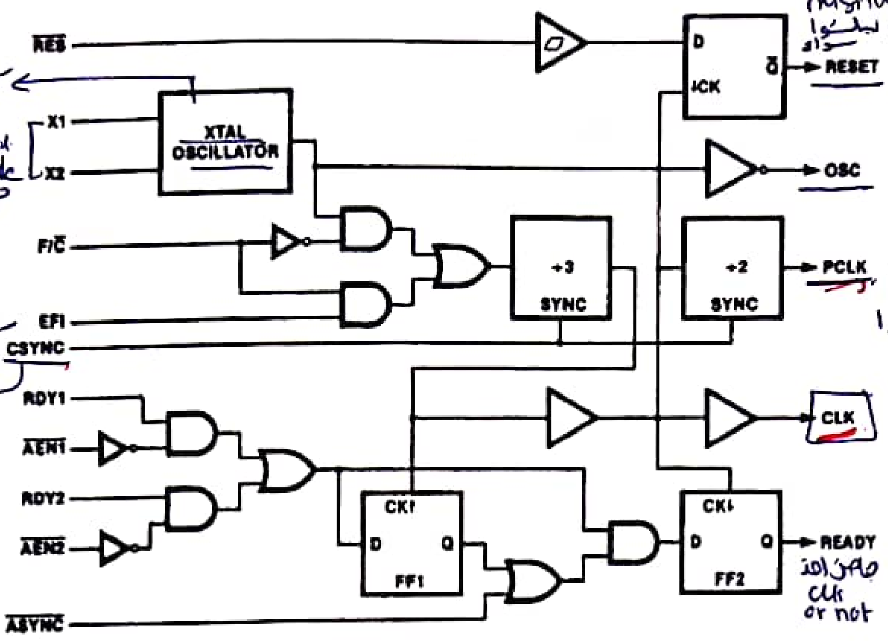
8.6 System Clock

clock signal

- □ The time base for synchronization of the internal and external operations of the microprocessor in a microcomputer system is provided by the *clock (CLK)* input signal.
- □ 8088 is available in two speeds. The standard 8088 operates at 5 MHz and the 8088-2 operates at 8 MHz.
  Freq of ~~microprocessor~~
  يعني السرعة هي ال clk signal
- □ The 8086 is manufactured in three speeds: 5-MHz 8086, 8-MHz 8086-2, and the 10-MHz 8086-1.
- □ The CLK is externally generated by the 8284 clock generator and driver IC.
  modul

8.6 System Clock

- □ Block diagram of the 8284 clock generator



square wave كرستال رح يكون

in current inside
base on matelial كرستال بتحدد ال Freq

External Freq input
مثلا باخد ال square wave من مكان ثاني غير الكرستال

بيستخدم في حال كان عندي اكثر من clk للـ large system

insrilation لحتى يبلشوا سوا

OSC oscilat كرستال Freq

PCLK preferal clk
I/O

CLK in for microprocess

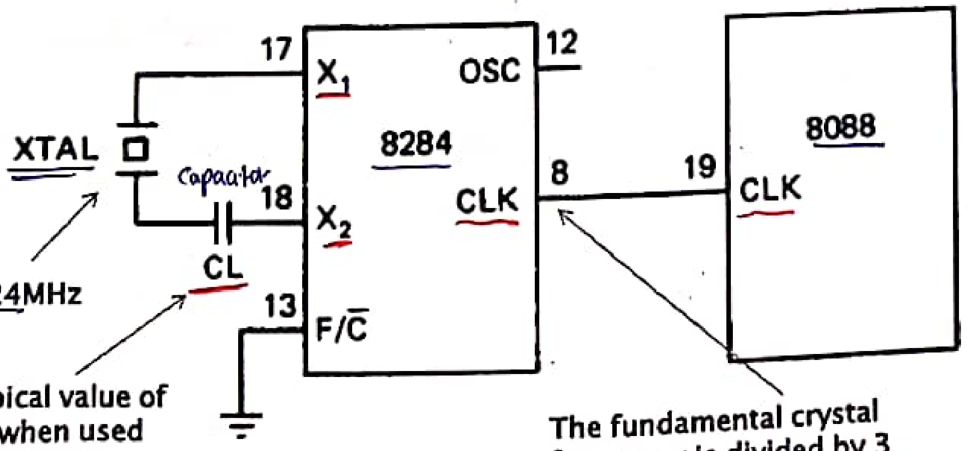
$clk = \boxed{\frac{1}{3}} OSC$ OSC=24
$= \frac{1}{3}(24)$
$= 8$
$clk = 2pclk$

READY جاهز اذا clk or not

10 من اسرع microprocUs

8.6 System Clock

- □ Connecting the 8284 to the 8088



15- or 24MHz crystal

Typical value of CL when used with 15MHz crystal is 12pF

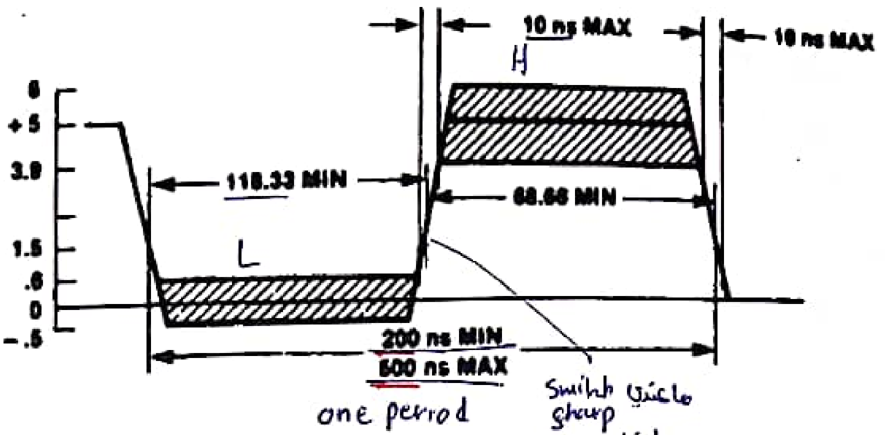
smooth for ripple

The fundamental crystal frequency is divided by 3 within the 8284 to give either a 5- or 8-MHz clock signal

$\frac{15}{3}$ $\frac{24}{3}$

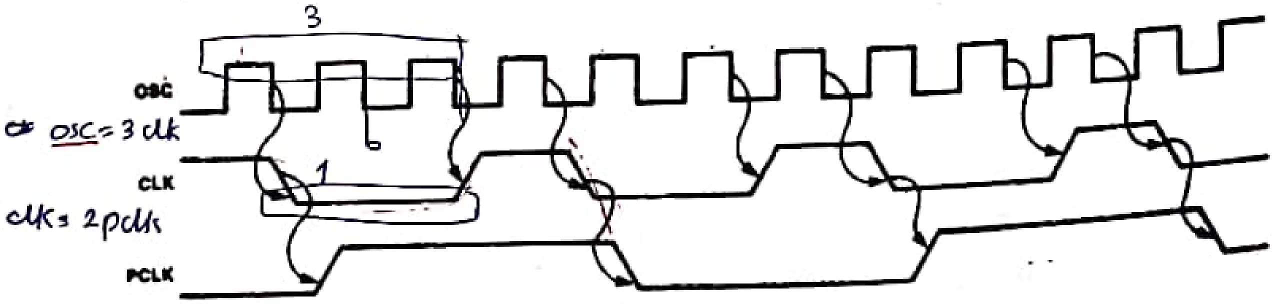
8.6 System Clock

- □ CLK waveform
  - □ The signal is specified at Metal Oxide Semiconductor (MOS)-compatible voltage level (rather than TTL)
  - □ The period of the 5-MHz 8088 can range from 200 ns to 500 ns, and the maximum rise and fall times of its edges equal 10 ns.



8.6 System Clock

- □ PCLK and OSC signals
  - □ The peripheral clock (PCLK) and oscillator clock (OSC) signals are provided to drive peripheral ICs.
  - □ The clock output at PCLK is half the frequency of CLK. The OSC output is at the crystal frequency which is three times of CLK.



osc = 6 pclk

8.6 System Clock

- EXAMPLE

Standard 5 احد 5 ميغا
Version 2 8 ال 8 ال //
Version 1 10 ال 10 ال //

If the CLK input of an 8086 MPU is to be driven by a 9-MHz signal, what speed version of the 8086 must be used and what frequency crystal must be attached to the 8284

- Solution:

The 8086-1 is the version of the 8086 that can be run at 9-MHz. To create the 9-MHz clock, a 27-MHz crystal must be used on the 8284.

meeser of time

8.7 Bus Cycle and Time States

الوقت الي بحتاجه ال micrp to communi للتواصل مع ال I0

time need to perform basic operation

minimum time need to complet basic operation

□ A bus cycle defines the basic operation that a microprocessor performs to communicate with external devices.

□ Examples of bus cycles are the memory read, memory write, input/output read, and input/output write. [basic operation]

□ The bus cycle of the 8088 and 8086 microprocessors consists of at least four clock periods. low and high

□ If no bus cycles are required, the microprocessor performs what are known as *idle states*.

□ When READY is held at the 0 level, *wait states* are inserted between states T3 and T4 of the bus cycle.

1 لما ال read ما تكون جاهزة ال micrp بستنى (read ما يتقرا ال) micropro

مكونات ال bus cycle need 4 clock periods

Basic operation

□ For write cycle, at T1 the address is prepared, data is ready at T2 and maintained during T3 and T4.

general address

بتجهز ال data لحتى تكون جاهزة

data avl. data bus

□ For read cycle, at T1 the address is prepared, at T2 the bus is at Z state, and the data is ready at T3 and maintained at T4.

ال data مو جاهزة بجهز ال data bus

ال write بتاخد 2 clock maintained و ال read بتاخد one clock

read

فاضي T2 لما تكون ال data جاهزة ready ب T3 بتكون

write

T2 بتكون ready و valid 2 clock

محتاج ال external device بحط ال data عليه

main ال T4 بتاخد 2 clock [illegible] ال device ال write

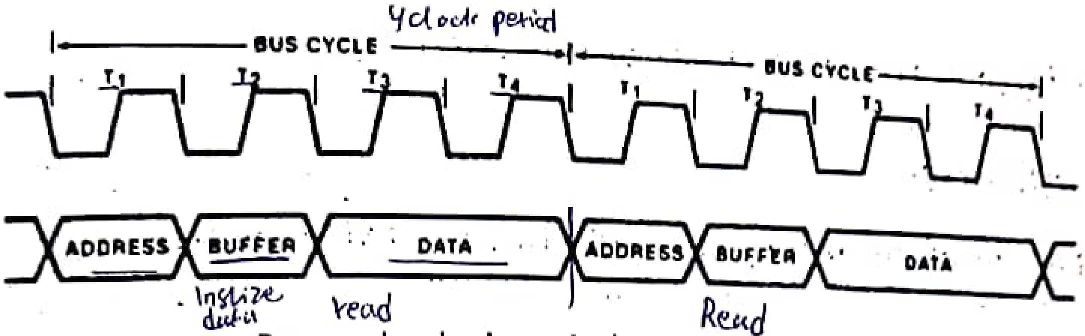
pclk مع clk ال microprocessor (write ال I/O external device)

8.7 Bus Cycle and Time States

- Multiplexed address/data transfer operation
  - Address output during T1
  - Bus lines in high-Z state in T2
  - Data transfer takes place during states T3 and T4

normal state

4clock period



Insize data

read

Read

Bus cycle clock periods

8.7 Bus Cycle and Time States

minimum

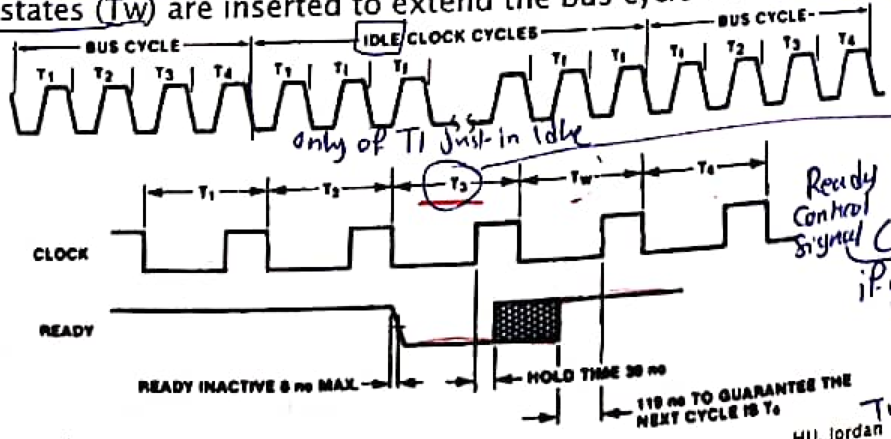
②. Bus cycle with idle   not access (not doing any operation)

- If no bus activity is necessary, microprocessor inserts idle states between bus cycles   like ↳ has full instruction queu
- Identified as T1
- May be due to the fact that the instruction queue is already full so no instructions need to be fetched   or doing processor , [illegible] access to memory

③. Bus cycle with wait states

- If the memory or I/O device is not able to respond in the duration of a bus cycle (500ns @8MHz) (slow I/O and memory) , it must make READY 0 during T3 to extend the bus cycle   more than one period   wait : Insert more clock period
- Wait states (Tw) are inserted to extend the bus cycle until READY returns to 1

wait clock period



only of T1 Just in Idle

check during T3

Ready Control Signal (busy)   بفحص ال

if external device not ready within T4

ازا كانت 1 ح اكمل T4

ازا كانت 0 بعمل extend bus Tw

Bus cycle with idle state, and wait state

وبعد ما اخلص بيرجع check other ازا لسه zero بضل بنفس الشي لحد ما يجي 1

8.7 Bus Cycle and Time States

- EXAMPLE

What is the duration of the bus cycle in the 8088-based microcomputer if the clock is 8 MHz and the two wait states are inserted.

- Solution: 4 min + 2 wait = 6 , duration of one clock period $= \frac{1}{f} = \frac{1}{8M} = \boxed{125n}$

The duration of the bus cycle in an 8 MHz system is given by $\boxed{125 \times 6 = 750}$

$$t_{cyc} = \underbrace{\overset{4 \times 125}{500\text{ ns}}}_{\text{minimum}} + \overset{\text{Extra}}{N} \times 125\text{ ns}$$

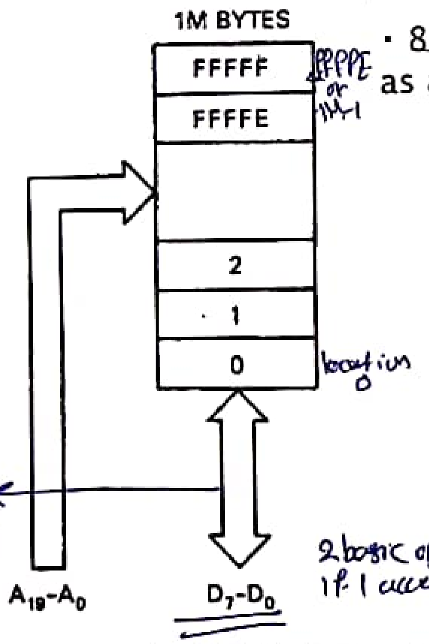
In this expression the N stands for the number of waits states. For a bus cycle with two wait states, we get

$$t_{cyc} = 500\text{ ns} + \underbrace{2}_{2\text{wait}} \times 125\text{ ns} = 500\text{ ns} + 250\text{ ns}$$

$$= \underline{750}\text{ ns}$$

$$\frac{1}{8M} \quad 125 \times \overset{}{6} \quad \underbrace{4}_{\text{min}} \; \underbrace{2}_{\text{wait}}$$

8.8 Hardware Organization of the Memory Address Space



1Mx8 memory bank of the 8088

the location I access I read data using data bus (8 bit) 8bit all external arc-microp.

2 basic operation if I access word

- 8088 memory hardware is organized as a single byte-wide memory bank
  - Size—1M X 8 bits
  - Physical address range— 0H-FFFFFH
  - Address/data bus de-multiplexed in external hardware
  - Input:

    20-bit address bus— A_{19} through A_0
  - Input/output:

    8-bit data bus—D_7 Through D_0

Size main memory 1M
required in one bank, each locat store 8 bit
1M

Memory size = $2^{\text{number of address lines}}$

→ 1M = 2^{20} (needs 20 address line)

if I want to access any location I use 20 bit because # of location 1M

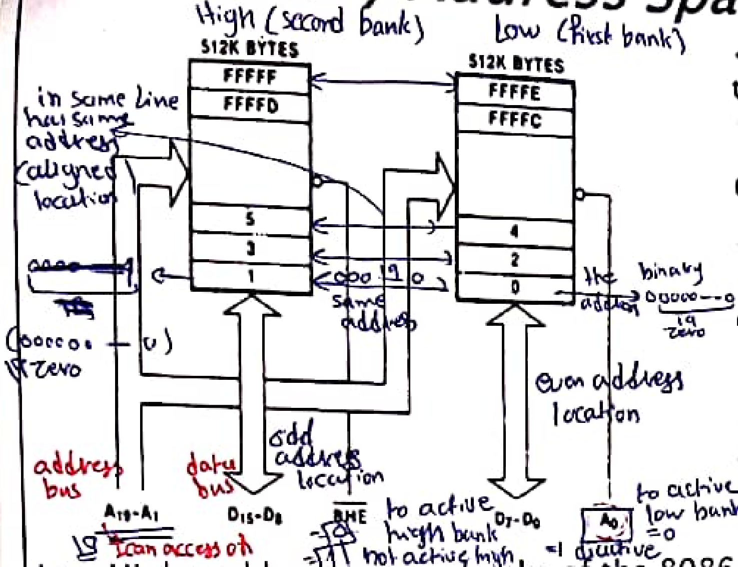
First location in high bank is address 1
" " " low " " " 0

× world 0 → 1 0 aligned
one world contain
world 1 → mys aligned 2 1

8.8 Hardware Organization of the Memory Address Space

location اذا كان بلوك ال line
1 بلوك ال line 2

Size main memory 1M location in two bank



High and low memory banks of the 8086

to access location for one address bus
A0 لازم غير منطق
enable or disable low bank

Memory size = $2^{\text{number of address lines}}$
$\log_2 512K = 19$
→ 512K = 2^{19} (needs 19 address line)

Data bus = 16bit in 8086 split in two 8bit

each bank has half M, the total 1M

- 8086 memory hardware is organized as a two bytewide memory bank
  - 2 location have the same address
  - Bank size—512K X 8 bits
  - Low-bank holds even addressed bytes— 0H through FFFFEH
    - مارح يسبب مشاكل لانه كل وحده ب bank
  - High-bank holds odd addressed bytes— 1H through FFFFFH
  - Address/data bus demultiplexed in external hardware
    - separate data bus
- Input:
  - zero 19 معاك
  - 20-bit+ address bus— A19 through A0, and BHE*
    - رح نفعل address اوله 1
    - انا بدي 0 نفعل
  - A1–A19 = selects storage location
  - to active A0 = 0 enables low bank 1 = disactive
    - low bank 0 = A0
  - BHE* = 0 enables high bank
- Input/Output:
  - بدي اقرأ من ال 2 بفعل الثنين عشان لانه عندي 2 data bus
  - 16-bit data bus—D15 Through D0
  - D7–D0 → even addressed byte accesses
  - D15–D8 → odd addressed byte

$0 \overset{20}{-} 000000\cancel{0}$ = location zero بال 8088 كان
$0 \overset{20}{-} 000000\cancel{1}$ = " 1 "
$0 \overset{19}{-} 0000 0$ = A0 بنستخدم 8086 بال

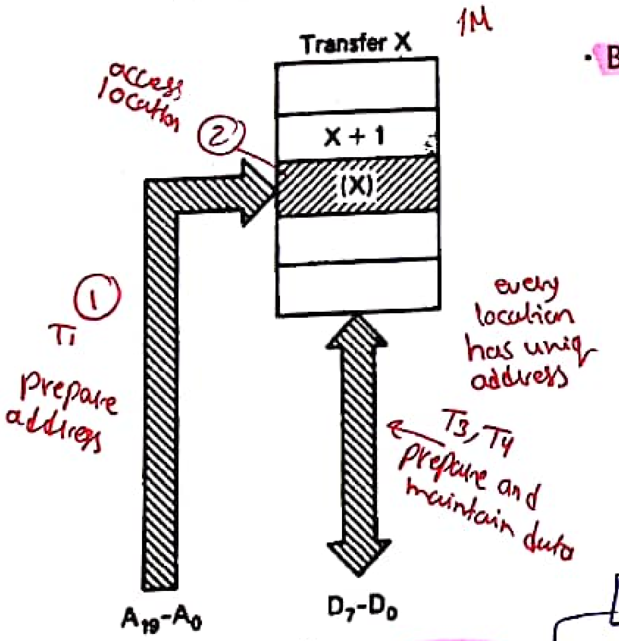
15 ——— 8 7 ——— 0
high part Connect with high bank
low part connet with low bank

, A0 بفعل → low part of data bus and " bank " memory
BHE بفعل → high part of data bus high " bank memory

8.8 Hardware Organization of the Memory Address Space

$0 \overset{18}{-} 000000\,1\cancel{0}$] 2
$0 \overset{18}{-} 00000\,\cancel{1}$] 3
نفس ال address

1M



Byte transfer by the 8088

- Byte access bus cycle
  - MPU applies address of storage location to be accessed over address lines A19–A0
    A19—most significant bit
    A0—least significant bit
  - Byte of data written into or read from address X transferred over data lines D0 through D7
    D7—most significant bit
    D0—least significant bit
  - Byte access takes a minimum of one bus cycle of duration
    - @5MHz—800ns one bus cycle $\frac{1}{5\times10^6} \times 4$ (Four period clk) = 800ns
    - @8MHz—500ns $\frac{1}{8\times10^6}\times4$ = 500ns (read or write)

كل ما كان اسرع كل ما كان اقل time needed to finish operation

How many bus cycle I need to access one byte of data from the memory of microprocessor 8088? I need only one bus cycle

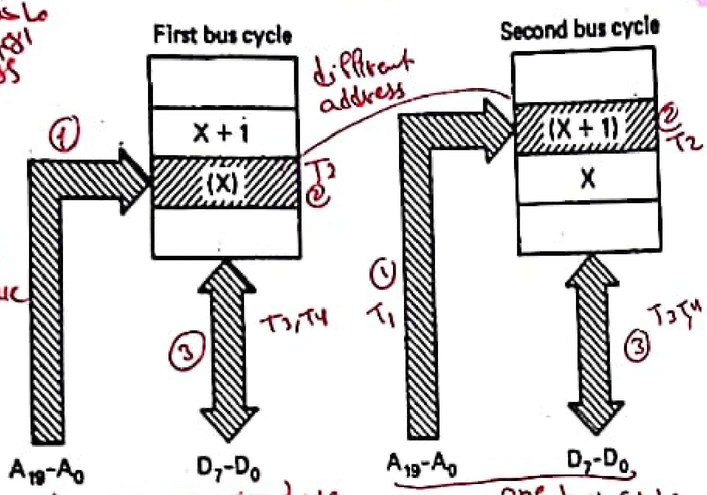
1 bus cycle (4 clock period)

8.8 Hardware Organization of the Memory Address Space

1word = 2byte with diffrent address , address byte 2 diffrent from address byte1 because in 8088 in the same bank

2byte → 2 diffrend address

2 locati [illegible] address



one bus cycle = 2bus cycle

Word transfer by the 8088

2 bus cycle

2 cycle

- Word access bus cycles 2byte
  - MPU must access two consecutive storage locations in memory—X and X+1
  - Requires two bus cycles
  - Address X accessed during cycle 1
  - Address X+1 accessed during cycle 2
  - Word access duration is a minimum of two bus cycle

  @5MHz—2 X 800ns = 1600ns $\frac{2}{5MHz} \times 8$

  @8MHz—2 X 500ns = 1000ns $\frac{2}{5MHz} \times 4 = 1600$ ns

$\frac{2}{8 \times 10^6} \times 4 = 1000ns$

8.8 Hardware Organization of the Memory Address Space

- Low bank byte access bus cycle
  - MPU applies even address X to both banks over address lines A19–A0
  - MPU enables just the low bank BHE*A0 = 10 → enables low bank
  - Byte of data written into or read from address X transferred over data lines D0 through D7
- High bank access bus cycle differences
  - Odd address X+1 applied to both banks
  - High bank enabled BHE*A0 =01 = → enable high bank
  - Byte-wide data transfer takes place over data line D8 through D15
- Word access bus cycle differences
  - Even word address X applied to both banks
  - MPU enables both banks BHE*A0 =00 = → enable low and high bank
  - Word-wide data transfer takes place over D0 through D15
  - All accesses takes a minimum of one bus cycle of duration

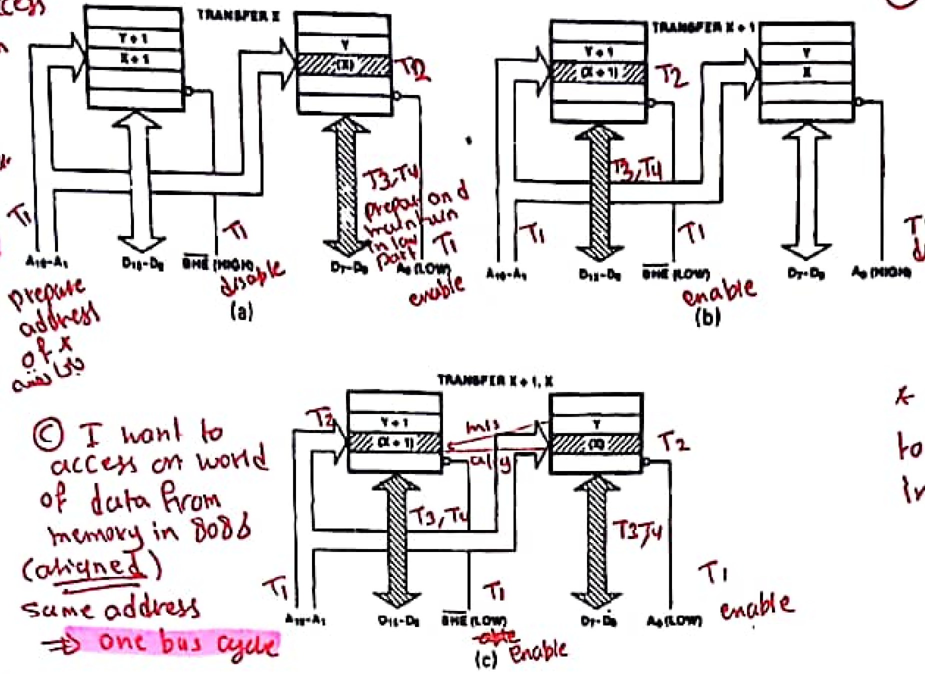
  @5MHz—800ns

  @8MHz—500ns

8.8 Hardware Organization of the Memory Address Space

Ⓐ I want to access one ~~bank~~ byte on data from low bank of microprocessor 8086
⇒ one bus cycle

Ⓑ I want to access one byte of data from high bank of memory in microp. 8086
⇒ one bus cycle



© I want to access on world of data from memory in 8086 (aligned)
same address
⇒ one bus cycle

* Same performance to access one byte in 8088 and 8086

aligned (X, X+1) even address first byte at even address and second part at odd address

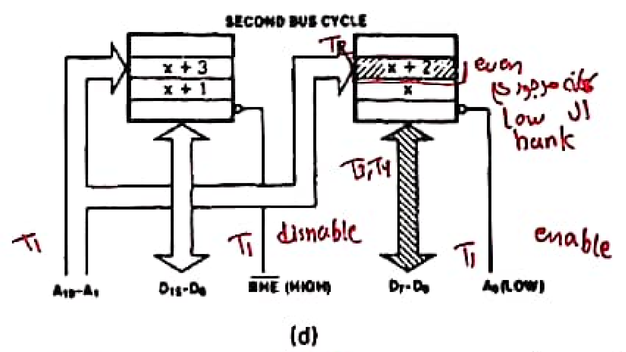
(X+1, X+2) misaligned in different line (odd address)

(a) Even address byte (b) Odd address byte transfer by the 8086
(c) Even address word transfer

8086 عكس 8088 ← بتفرق اذا كانت aligned or misalignd

* in 8088 → سواء كان even او odd
aligned او misaligned لازم
access word بياخذ دائما 2 bus cycle

8.8 Hardware Organization of the Memory Address Space



Odd address word transfer by the 8086

- Misaligned-word access bus cycles
  - Word starting at address X+1 is misaligned   mis address (diffrent)
  - Requires two bus cycles
    - Access byte at address X+1 during cycle 1
      A_{19}–A_0 = X+1
      BHE*A_0 =01→ enables high bank
      D_{15}–D_8 → carries data
    - Access byte at address X+2 during cycle 2
      A_{19}–A_0 = X+2
      BHE*A_0 =10 → enables low bank
      D_7–D_0 → carries data
  - Word access duration is a minimum of two bus cycle
    @5MHz—2 X 800ns = 1600ns
    @8MHz—2 X 500ns = 1000ns
  - Impact on performance—software should minimize accessing

odd address first byte in odd address and second part at even address

I need 2 bus cycle in this case and similar performance with 8088 (misaligned)

8.8 Hardware Organization of the Memory Address Space

odd number = odd address

- EXAMPLE

  Is the word at memory address $0123\boxed{1}_{16}$ of an 8086-based microcomputer aligned or misaligned? How many cycle are required to read it from memory?

- Solution:

  The first byte of the word is the second byte at the aligned-word address 01230_{16}. Therefore, the word is misaligned and required two bus cycles to be read from memory.

8.9 Address Bus Status Codes

* If I want to expand the size of memory 1M — larger like 4M

□ Whenever a memory bus cycle is in progress, an address bus status code S_4S_3 is output by the processor.

□ S_4S_3 identifies which one of the four segment register is used to generate the physical address in the current bus cycle:

□ S_4S_3 =00 identifies the extra segment register (ES)

□ S_4S_3 =01 identifies the stack segment register (SS)

□ S_4S_3 =10 identifies the code segment register (CS)

□ S_4S_3 =11 identifies the data segment register (DS)

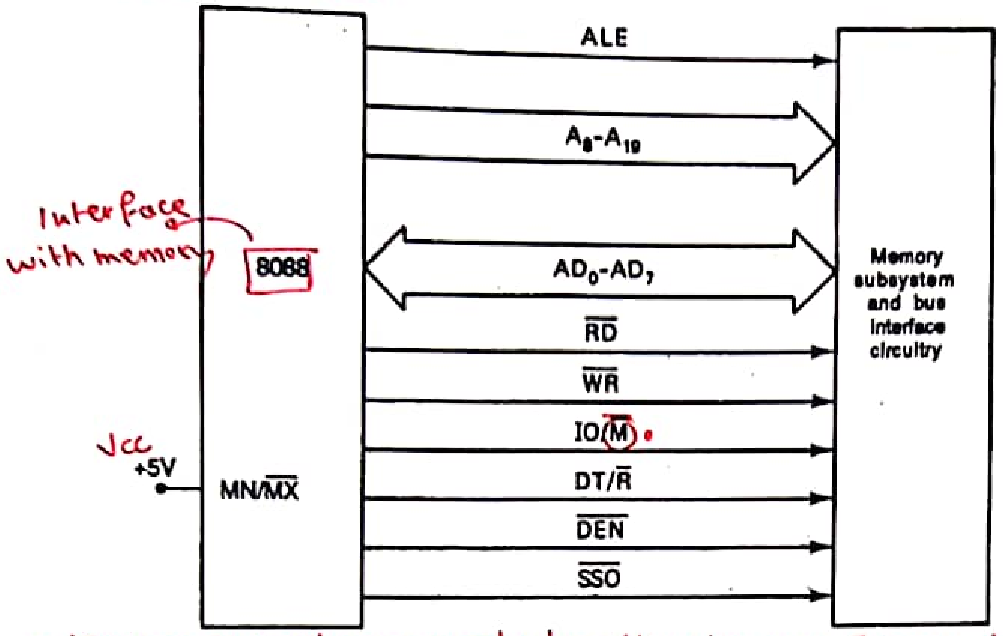
□ Since each combination of S_4S_3 leads to select different memory segment, the memory address reach of the microprocessor can thus be expanded to 4 Mbytes.

اعشانة لازم
نزيد ال address bus
(Hardware)
wire

22 address line
$2^{22} = 4M$

8.10 Memory Control Signals

□ Minimum-mode memory control signals



8088 8086

8 → 16

$\overline{M}$/IO → $\overline{IO}$/M

SSO → BHE

microprocessor not communicate directly with any External device

Interface circuit [illegible]

8.10 Memory Control Signals

□ Minimum-mode memory control signals

- □ **ALE – Address Latch Enable** – used to latch the address in external memory (valid address on the bus).
- □ **IO/M' – Input-Output/Memory** – signal external circuitry whether a memory of I/O bus cycle is in progress.
- □ **DT/R' – Data Transmit/Receive** – signal external circuitry whether the 8088 is transmitting or receiving data over the bus.
- □ **RD' – Read** – identifies that a read bus cycle is in progress.
- □ **WR' – Write** – identifies that a write bus cycle is in progress.
- □ **DEN' – Data Enable** – used to enable the data bus.
- □ **SSO' – Status Line** – identifies whether a code or data access is in progress.

8.10 Memory Control Signals

- □ The control signals for the 8086's minimum mode memory interface differs in three ways:
  - □ IO/M signal is replaced by M/IO signal.
  - □ The signal SSO is removed from the interface.
  - □ BHE (bank high enable) is added to the interface and is used to select input for the high bank of memory in the 8086's memory subsystem.

8.10 Memory Control Signals

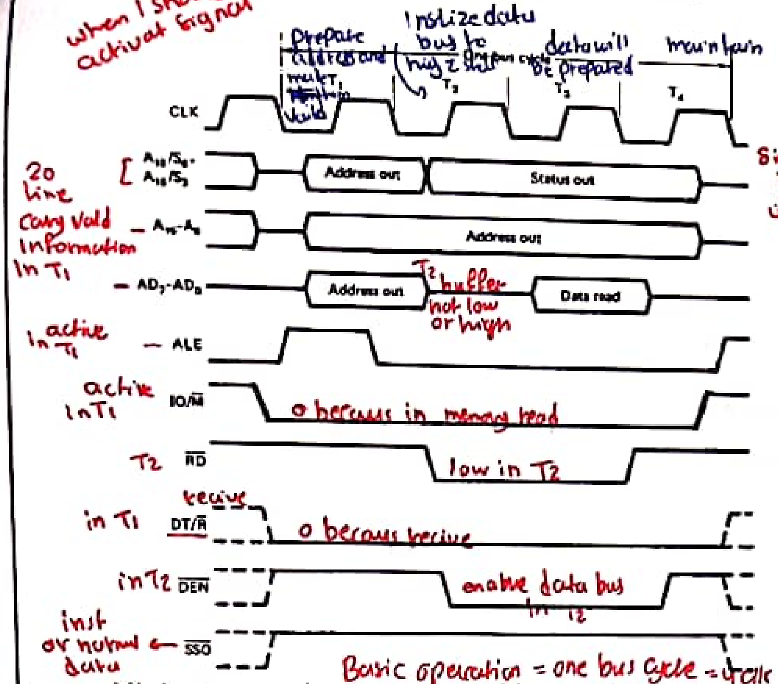
- □ Maximum-mode memory control signals
  - □ MRDC – Memory Read Command
  - □ MWTC – Memory Write Command
  - □ AMWC – Advanced Memory Write Command

not general Control signal
general status Code
S_0, S_1, S_2

Status Inputs			CPU Cycle	8288 Command
$\overline{S_2}$	$\overline{S_1}$	$\overline{S_0}$		
0	0	0	Interrupt acknowledge	$\overline{INTA}$
0	0	1	Read I/O port	$\overline{IORC}$
0	1	0	Write I/O port	$\overline{IOWC}$, $\overline{AIOWC}$
0	1	1	Halt	None
1	0	0	Instruction fetch	$\overline{MRDC}$ } for Read
1	0	1	Read memory	$\overline{MRDC}$
1	1	0	Write memory	$\overline{MWTC}$, $\overline{AMWC}$ } for write
1	1	1	Passive	None

3 command belong to memory

8.11 Read and Write Bus Cycle



Basic operation = one bus cycle = 4 clk period

Minimum-mode memory read bus cycle of the 8088

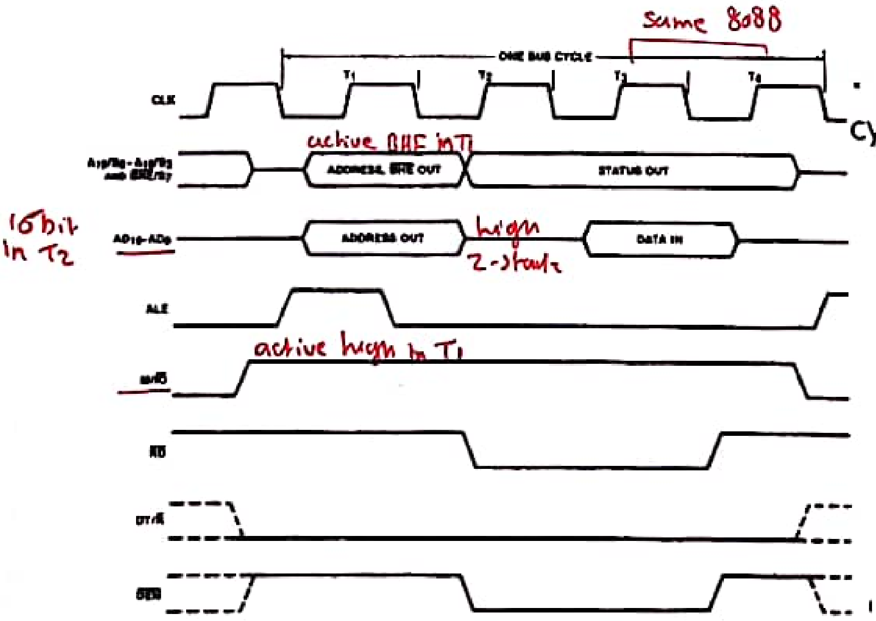
timing diagram explain time when we should generate different signal In order to interface microprocessor and other device

- Read bus cycle timing diagram—shows relationship between signals relative to times states
  - T1 state—read cycle begins

signals لازم يضلوا لل T4

- Address output on A0-A19
- Pulse produced at ALE--address should be latched in external circuitry on trailing edge of ALE
- IO/M* set to 0 → memory bus cycle
- DT/R* set to 0 → set external data bus control circuitry for receive mode (read)
- T2 state
  - Status code output on S3-S6
  - AD0 through AD7 tri-stated in preparation for data bus operation
  - RD* set to 0 → read cycle
  - DEN* set to 0 → enable external data bus control circuitry
- T3 state
  - Data on D0-D7 read by the MPU
- T4 state—read cycle finishes
  - RD* returns to 1→ inactive level
  - Complete address/data bus tristate
  - IO/M* returned to 1 → IO bus cycle
  - DEN* returned to 1→ inactive level
  - DT/R* returns to 1→ transmit level

8.11 Read and Write Bus Cycle



- Differences of 8086 read bus cycle
  - BHE* is output along with the address in T1
  - Data read by the MPU can be carried over all 16 data bus lines
  - M/IO*—which replaces IO/M*—switches to 1 instead of 0 at the beginning of T1
  - SSO* signals is not produce

don't use SSO in T1

Minimum-mode memory read bus cycle of the 8086

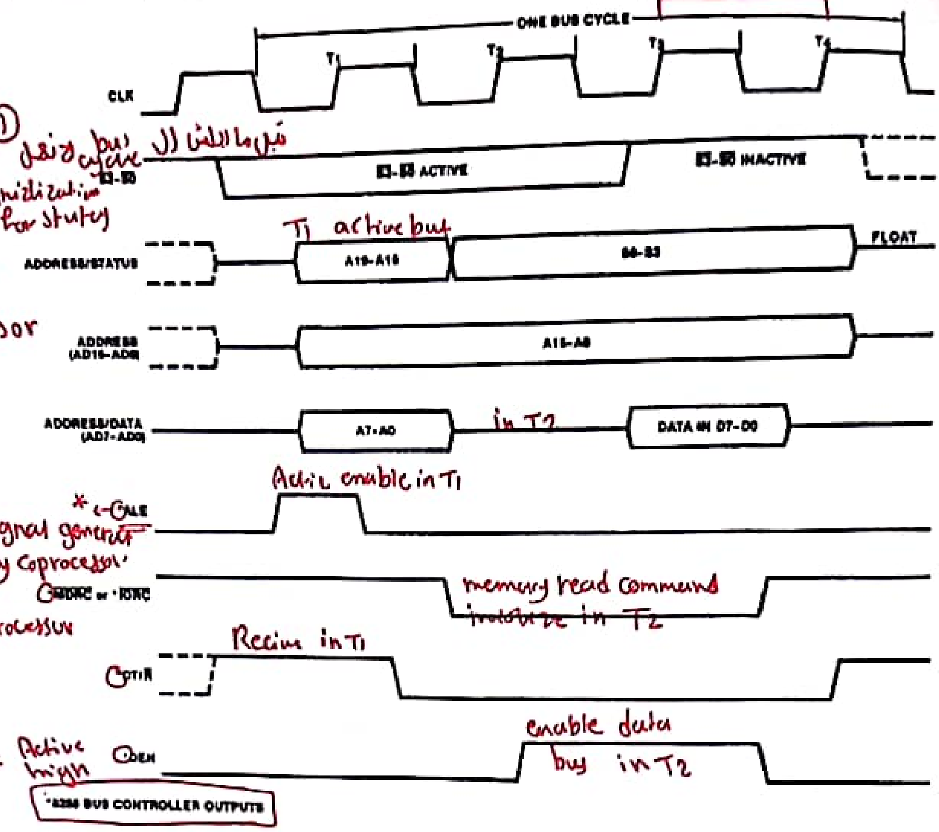
8.11 Read and Write Bus Cycle

□ Read cycle ① قبل ما اطلع ال bus cycle بعمل initialization for status

① general status code

② general add. information in T1

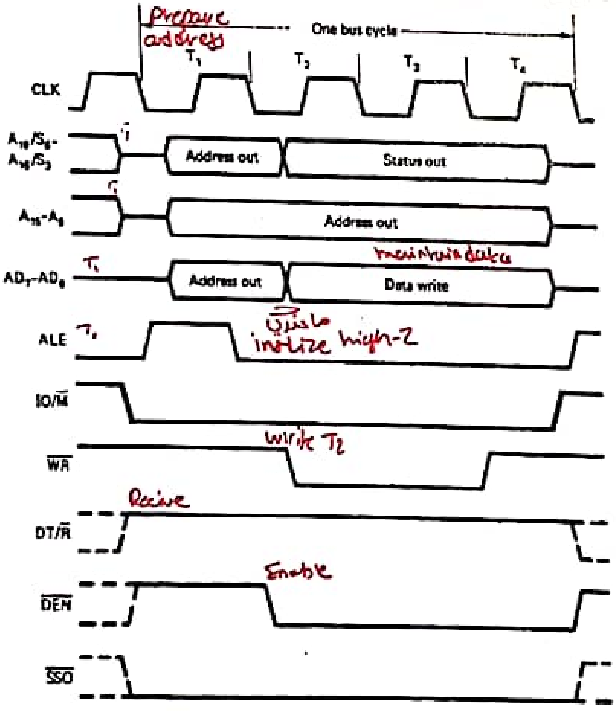
③ ALE, DT/R from coprocessor



Maximum-mode memory read bus cycle of the 8086

not general. control signal ← microprocessor ال maximum بال [illegible]

8.11 Read and Write Bus Cycle



Minimum-mode memory write bus cycle of the 8088

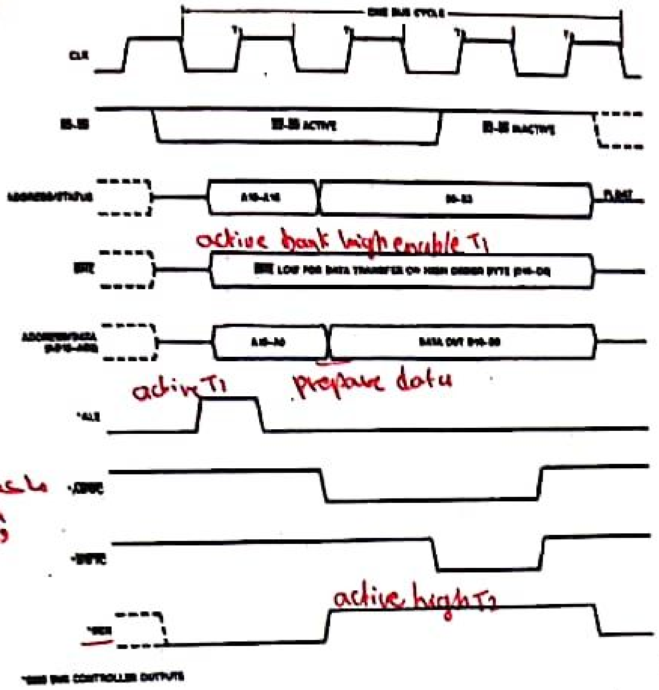
- Write bus cycle timing diagram—shows relationship between signals relative to times states
  - T1 state—write cycle begins
    - Address output on A0-A19
    - Pulse produced at ALE and address latched in external circuitry on trailing edge of ALE
    - IO/M* set to 0 → memory bus cycle
    - DT/R* remains at 1→ external data bus control circuitry for transmit mode (write)
  - T2 state
    - Status code output on S3-S6
    - AD0 through AD7 transitioned to data bus and write data placed on bus
    - DEN* set to 0 → enable external data bus control circuitry
    - WR* set to 0 → write cycle
  - T3 or T4 state
    - Data on D0-D7 written into memory (memory decides when!)
  - T4 state—write cycle finishes
    - WR* returns to 1→ inactive level
    - Complete address/data bus tri-stated
    - IO/M* returned to 1 → IO bus cycle
    - DEN* returned to 1→ inactive level

① one difference between T1, Recive (write, read)

② prepare data, ~~enable data bus~~ in T2

initize high-Z ON write operation

8.11 Read and Write Bus Cycle



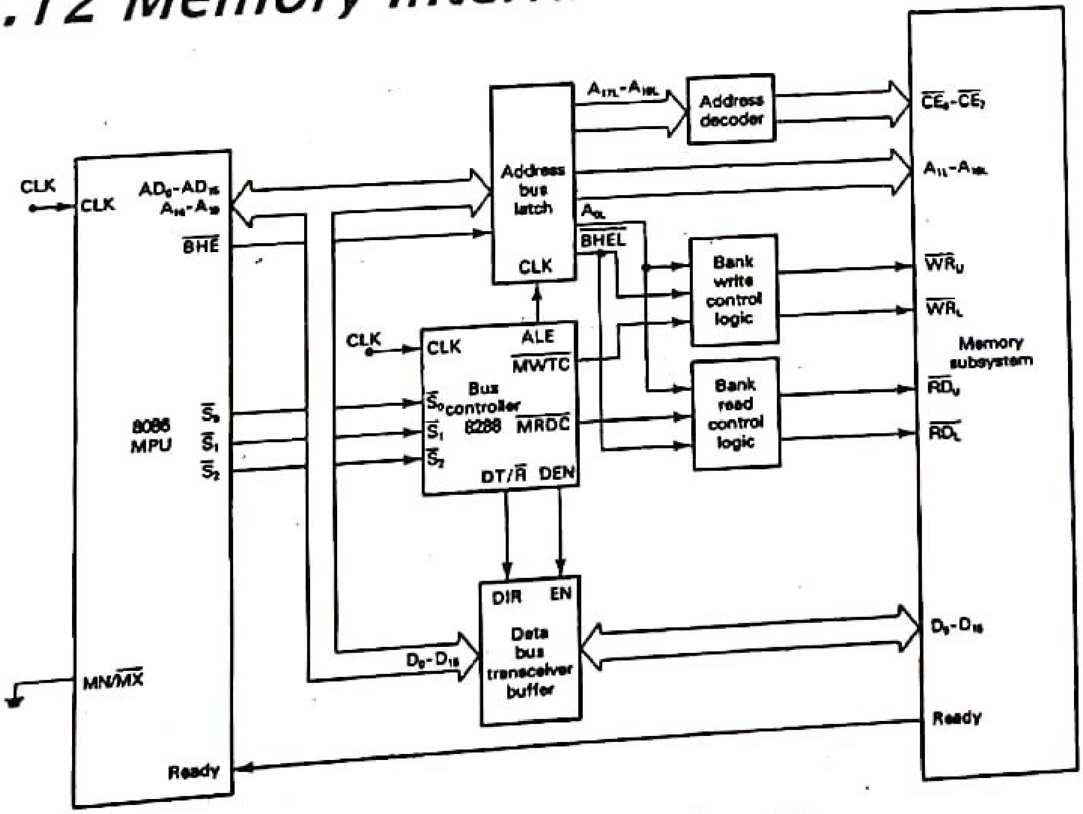
ما عندنا
10/4
[illegible]

Maximum-mode memory write bus cycle of the 8088

- Similar to 8088/8086 minimum-mode write bus cycle
  - Address and data transfer operation identical
  - Transfer may be a high-byte, low-byte, word
  - Differences is the 8288 produces the bus control signals—ALE, DEN, AMWC*, and MWTC*
  - Bus status code S2*-S0* output prior to T1 and held through T2
  - AMWC* and MWTC* replace WR* (Note timing difference)
  - DEN =1 produced instead of DEN* =0 (change in external circuitry!)

8.12 و 8.13 محذوف

X 8.12 Memory Interface Circuit

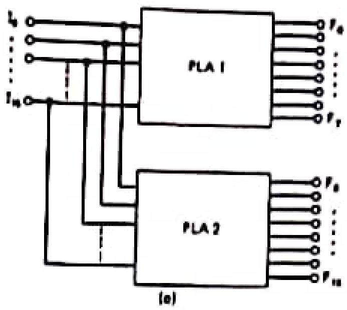


Memory interface block diagram

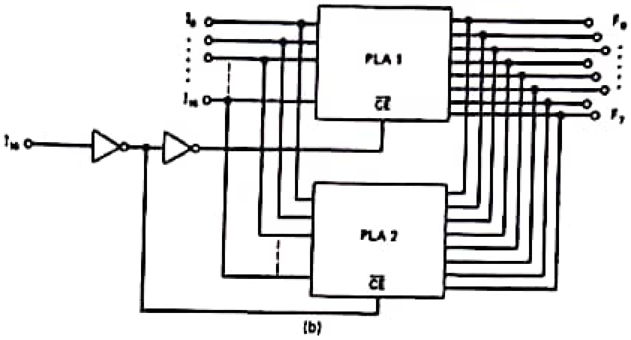
8.13 Programmable Logic Arrays

□ Expanding PLA capacity

Expanding output word length



Expanding input word length



8.14 Types of Input/Output

③ use two Instruction in/out
→ to read data from I/O space
→ to write data to the I/O space
bitter ~~little~~ performance and highe ~~space~~ speed (fast)

① □ Isolated input/output · Interface with input/out operation

□ When using isolated I/O in a microcomputer system, the I/O device are treated separate from memory.

access need only 16 bit address log₂64k

× access IO space much faster than accessing main memory of micropro.

□ The memory address space contains 1 M consecutive byte address in the range 00000H through FFFFFH; and that the I/O address space contains 64K consecutive byte addresses in the range 0000H through FFFFH.

If I use the Extra space for input/output we called Isolated

لذا نسميه

. Isolated space rather than memory

× disadvantage: I should use accumulator in order to transfer data between microp and IO space

□ All input and output data transfers must take place between the AL or AX register and I/O port.

if the data byte

if the data world

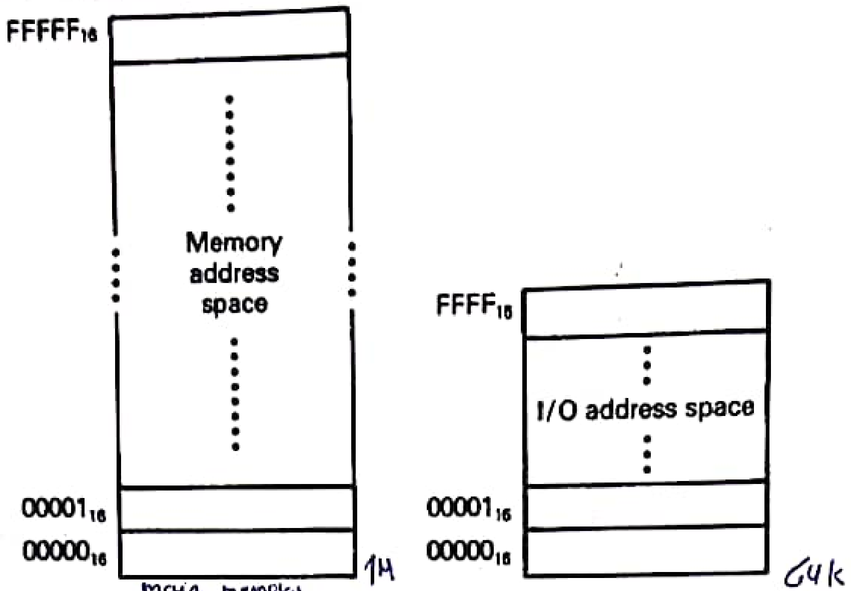
advantage: ① I have explicit extra space of 64K location addition 1M memory

② space for Isolated 64K than space for memory mapped (4K).

② use special instruction different the function access main memory Mov, add, sub ...

8.14 Types of Input/Output

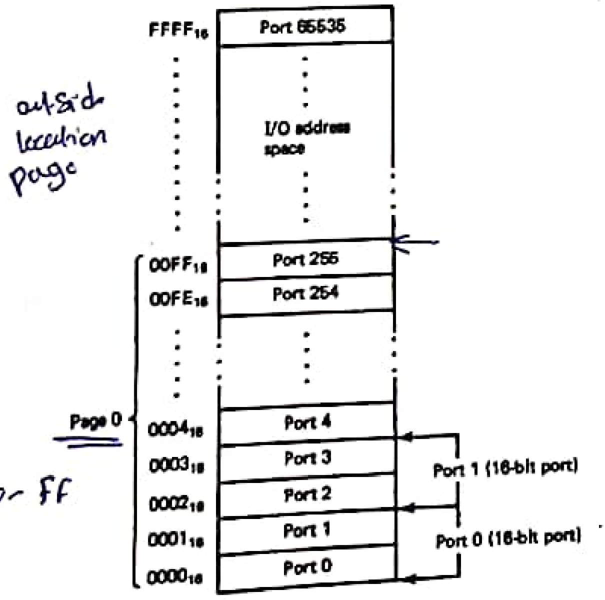
□ Isolated input/output



8088/8086 memory and I/O address spaces

8.14 Types of Input/Output

□ Isolated input/output



Isolated I/O ports

location == port

connect عنده with input-output device

- Input/output data organization
  - Supports byte and word I/O ports
    - 64K independent byte-wide I/O ports
    - 32K independent aligned word-wide I/O ports
    - Word ports may also be misaligned
  - Examples:

Byte ports 0,1,2 → addresses 0000H, 0001H, and 0002H

Word ports 0,1,2 → addresses 0000H, 0002H, 0004H

  - Advantages of isolated I/O
    - Complete memory address space available for use by memory
    - I/O instructions tailored to maximize performance
  - Disadvantage of Isolated I/O
    - All inputs/outputs must take place between an I/O port and accumulator register (A)

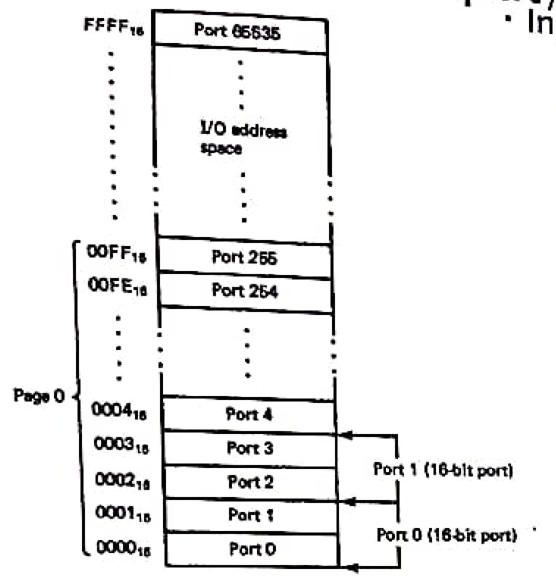
نعني مين ال memory عبارة عن location وكل واحد 8bit بتكون من

and can transfer 1 byte of data or more like 1 word

8.14 Types of Input/Output

□ Isolated input/output

نفس مبدأ ال memory



Isolated I/O ports

one bank, access of one byte → one bus cycle 8088
" " world → 2 " "

- Input/output data organization
  - All I/O accesses take either one or two bus cycles
    - Byte input/output= 1 bus cycle
    - Aligned word input/output= 1 bus cycle—on 8086
    - Misaligned word input/output = 2 bus cycles
- Page 0
  - First 256 byte addresses → 0000H – 00FFH
  - Can be accessed with direct or variable I/O instructions
  - Ports F8H through FFH reserved

Page 0 بنهاية

اذا كانت ثانوية " " " → 1 bus 8086
" " world → 1 for aligned
→ 2 " mis aligned

8.14 Types of Input/Output

② □ Memory-mapped input/output

use part of space from memory

- □ In the case of memory-mapped I/O, MPU looks at the I/O port as though it is a storage location in memory.
- ☒ Some of the memory address space is dedicated to I/O ports.
- □ Instructions that affect data in memory are used instead of the special I/O instructions.
- □ The memory instructions tend to execute slower than those specifically designed for isolated I/O.

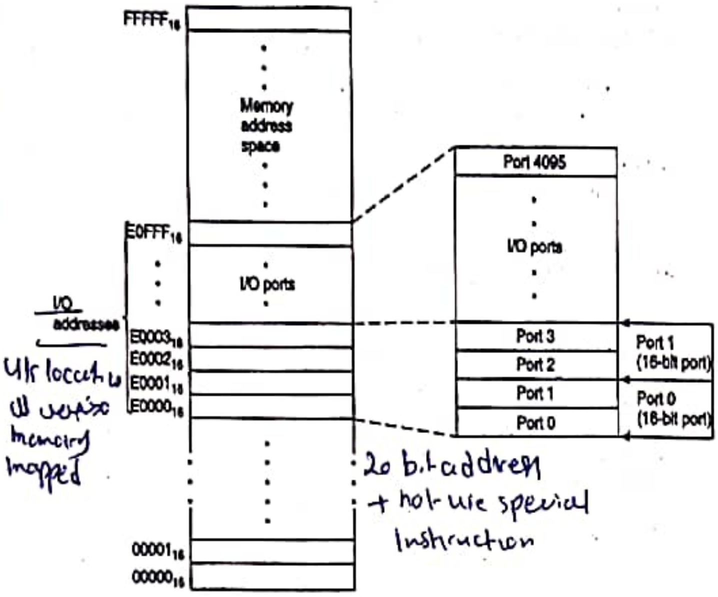
in order to access I need 20 bit address

adv:
① no Instruction to use acumilator
can I use ~~only~~ any registar to transfer data between microprocessor and memory mapped
② I have wide rang of Instruction can be use for memory mapped

dis: ① less space for memory mapped just 4k location not exploit Extra space
Extra دااستغل مساحة ال
② memory Instruction they have less performance than special Inst.

8.14 Types of Input/Output

- □ Memory-mapped input/output



4k location لل [illegible] memory mapped

20 bit address + not use special Instruction

Memory mapped I/O ports

- Example:
- E0000H-E0FFFH → 4096 memory addresses assigned to I/O ports
- E0000H, E0001H, and E0002H correspond to bytewide ports 0,1, and 2
- E0000H and E0001H correspond to word-wide port 0 at address E0000H
- Advantages of memory mapped I/O
  - Instructions that affect data in memory (MOV, ADD, AND, etc.) can be used to perform I/O operations
  - I/O transfers can take place between and I/O port and any of the registers
- Disadvantage of memory mapped I/O
  - Memory instructions perform slower
  - Part of the memory address space cannot be used to implement memory

8.15 Isolated Input/Output Interface

- □ I/O devices:
  - □ Keyboard (input) send data to microprocessor
  - □ Printer (output) read " from "
  - □ Mouse (input)
  - □ 82C55A, etc. (PPI section .13)
- □ Functions of interface circuit: ↳ to exchange data from microprocessor ↳ for Isolated and external device
  - □ 1 Select the I/O port — address infor. بختار الـ device اكترونز connect with microprocessor
  - □ 2 Latch output data — the output will maintain in 2 clock period 125 ns
  - □ 3 Sample input data — memory اذا الـ memory بيختص memory mapped interface
  - □ 4 Synchronize data transfer
  - □ 5 Translate between TTL voltage levels and those required to operate the I/O devices. — كطابقها نقلها valid لجهاز الـ

3) determine ~~data~~ value of data from IO device (H L)

1) speed Synchronize between IO device and microprocessor (لازم كونوا مختلفين)

5) voltage translate (الـ V مختلفة تحت)

8.15 Isolated Input/Output Interface

□ Minimum-mode interface:

- Similar in structure and operation to memory interface
  - I/O devices—can represent LEDs, switches, keyboard, serial communication port, printer port, etc.
  - I/O data transfers take place between I/O devices and MPU over the multiplexed-address data bus

    AD0-AD7
    A8-A15
- Control signal review Smillar to memory Interface
  - ALE = pulse to logic 1 tells bus interface circuitry to latch I/O address
  - RD* = logic 0 tells the I/O interface circuitry that an input (read) is in progress
  - WR*= logic 0 tells the I/O interface circuitry that an output (write) is in progress
  - IO/M*= logic 1 tells I/O interface circuits that the data transfer operation is for the IO subsystem
  - DT/R* = sets the direction of the data bus for input (read) or output (write) operation
  - DEN*= enables the interface between the I/O subsystem and MPU data bus

بحدد النوع
اذا Isolated (IO) بكون
اذا memory mapped بكون (M)

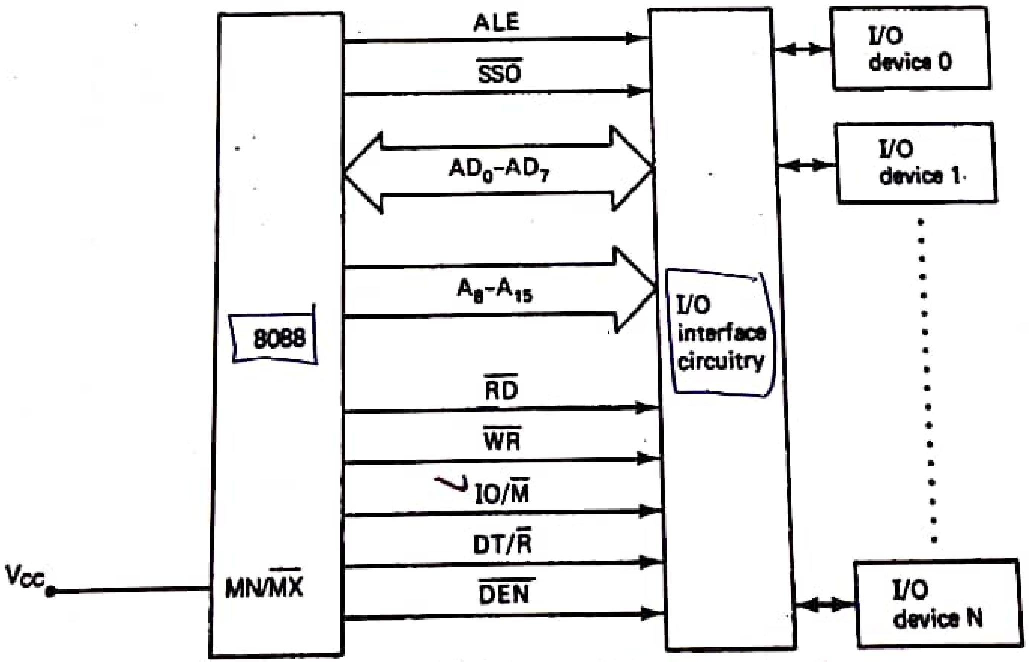
* 2 diffrent IO Interface, Micruprocessor

① ≠ of address line 16 بدل 20

② IO Inisection Communication بستخدم IO (IO/M)

8.15 Isolated Input/Output Interface

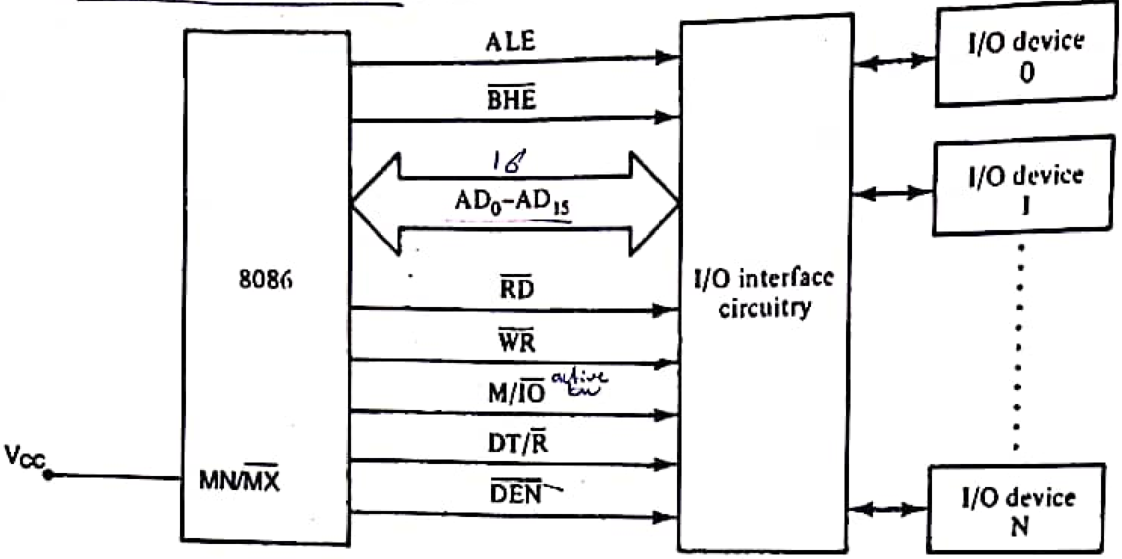
□ Minimum-mode interface:



Minimum-mode 8088 system I/O interface

8.15 Isolated Input/Output Interface

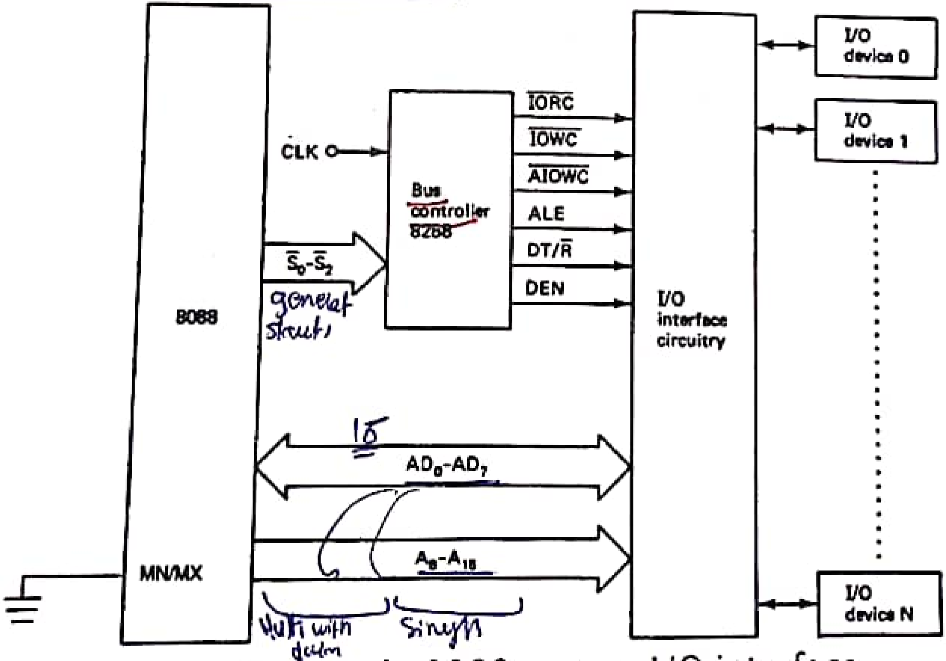
□ Minimum-mode interface:



Minimum-mode 8086 system I/O interface

8.15 Isolated Input/Output Interface

□ Maximum -mode interface:



Maximum-mode 8088 system I/O interface

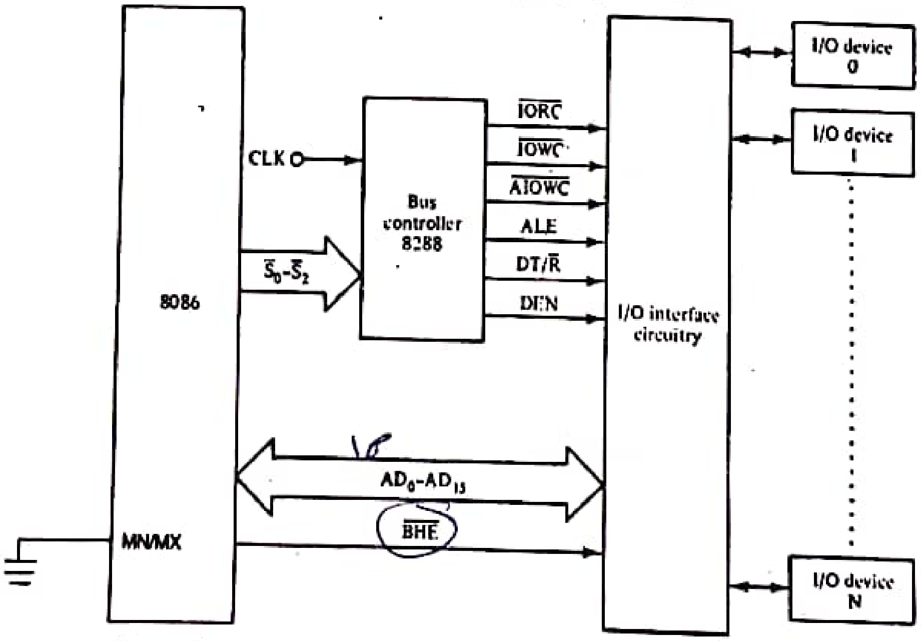
8.15 Isolated Input/Output Interface

□ Maximum -mode interface:

- 8288 bus controller produces the control signals
- Signal changes
  - IORC* replaces RD*
  - IOWC* and AIOWC* replace WR*
  - DEN is complement of DEN*
  - IO/M* no longer needed (bus controller creates separate IO read/write controls)
  - SSO* no longer part of interface

8.15 Isolated Input/Output Interface

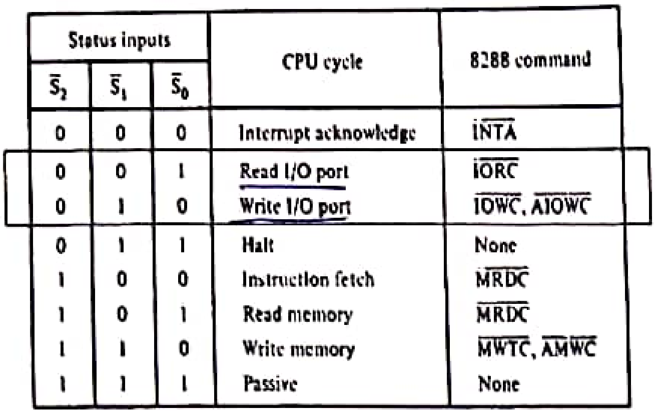
□ Maximum -mode interface:



Maximum-mode 8086 system I/O interface

8.15 Isolated Input/Output Interface

□ Maximum -mode interface:



Status inputs			CPU cycle	8288 command
$\overline{S}_2$	$\overline{S}_1$	$\overline{S}_0$		
0	0	0	Interrupt acknowledge	$\overline{INTA}$
0	0	1	Read I/O port	$\overline{IORC}$
0	1	0	Write I/O port	$\overline{IOWC}$, $\overline{AIOWC}$
0	1	1	Halt	None
1	0	0	Instruction fetch	$\overline{MRDC}$
1	0	1	Read memory	$\overline{MRDC}$
1	1	0	Write memory	$\overline{MWTC}$, $\overline{AMWC}$
1	1	1	Passive	None

2case

I/O bus cycle status codes

- Bus status code review
  - During all I/O accesses one of two bus cycle status code are output by the MPU
    - Read I/O port
    - Write I/O port
  - 8288 decodes to produce appropriate control command signals
    - IORC* → input (read I/O)
    - IOWC* → output (write I/O)
    - AIOWC* → output (write I/O)

8.16 Input/Output Data Transfers

□ Input/output data transfers in the 8088 and 8086 microcomputers can be either byte-wide or word-wide.

□ The port that is to be addressed is specified by the IO address.

□ I/O addresses are 16 bits in length and are output by the 8088 to the I/O interface over bus lines AD_0 through AD_7 and A_8 through A_{15}.why?

□ In 8088, the word transfers is performed as two consecutive byte-wide data transfer and takes two bus cycle.

□ In 8086, the word transfers can takes either one or two bus cycle.

□ Word-wide I/O ports should be aligned at even-address boundaries to ensure that one cycle is enough to complete the word operation.

IN: Read from Extra space
OUT: write to Extra "

8.17 Input/Output Instructions

- Types of instructions
  - Direct I/O instructions—only allow access to ports at page 0 addresses 00 – FF
  - indirect Variable I/O instructions—allows access of ports anywhere in the I/O address space 1000 – FFFF
- Direct I/O instructions → Source (address of port) بكون بين [ ]

(distenation) accumulator ← IN Acc,Port acc. ال will be transfer ← address الداتا الموجودة بهاد ال ×

dist. address of port ← OUT Port,Acc → Soure (acc) → مرح تاخذ الداتا الموجودة بال acc وتخزنها بال address

  - Port = 8-bit direct address—limited to 0H through FFH (page 0)
  - Acc = accumulator register AX (word transfer); AH or AL (byte transfer)
  - Example: within Page0

    IN AL, 0FEH

    (FE) → AL (byte input operation)

    read one byte of data that the address FE (Iospace, memory mapped)
  - Also known as accumulator I/O—because source or destination must always be in accumulator (Acc) register

Mnemonic	Meaning	Format	Operation	
IN	Input direct	IN Acc,Port	(Acc) ← (Port)	Acc = AL or AX
	Input indirect (variable)	IN Acc,DX	(Acc) ← ((DX))	
OUT	Output direct	OUT Port,Acc	(Port) ← (Acc)	
	Output indirect (variable)	OUT DX,Acc	((DX)) ← (Acc)	

الاختلاف بين indirect & direct

ال address كيف بنكتب of port

بال Inst ازا بال page0

بنكتب بشكل Direct

ازا كان indirect بحدد بالاول

بال DX وبعدها بكتب

محل ال port

read one byte from IO

IN — AL, AH — IN — OUT

direct address within page0

indirect address not within page0

[illegible]

8.17 Input/Output Instructions

- Types of instructions
  - Direct I/O instructions—only allow access to ports at page 0 addresses
  - Variable I/O instructions—allows access of ports anywhere in the I/O address space
- Variable I/O instructions

  IN Acc,DX → value of DX represent address of port

  OUT DX,Acc
  - DX = 16-bit indirect address—allows access to full I/O address space
  - Acc = accumulator register AX (word transfer); AH or AL (byte transfer)
  - Example:

    MOV DX,A000H ;load I/O address DX=A000

    IN AL,DX ;input value to AL

    MOV BL,AL ;copy value to BL مخزن بال BL

    (A000H) → BL (byte input operation)

IN, out

indirect IN read byte (AL, AH) of the input port address A000 and store it in BL Register

Mnemonic	Meaning	Format	Operation	
IN	Input direct	IN Acc,Port	(Acc) ← (Port)	Acc = AL or AX
	Input indirect (variable)	IN Acc,DX	(Acc) ← ((DX))	
OUT	Output direct	OUT Port,Acc	(Port) ← (Acc)	
	Output indirect (variable)	OUT DX,Acc	((DX)) ← (Acc)	

8.17 Input/Output Instructions

- EXAMPLE

Write a sequence of instructions that will output the data FFH to a byte-wide output port at address ABH of the I/O address space.

OUT.
direct. indirect
with AB

- Solution:

First, the AL register is loaded with FFH as an immediate operand in the instruction

MOV AL, FFH

لازم ال data تكون
موجودة بال AL

Now the data in AL can be output to the byte-wide output port with the instruction

OUT ABH, AL

address

لو غيرنا السوال ل main memory

بنزلها ياها

Mov AL, FF
~~Mov [AB], AL~~
or
Mov [AB], FF

8.17 Input/Output Instructions

- EXAMPLE

Write a series of instructions that will output data FFH to an output port located at address B000H of the I/O address space.

- Solution: special Inst.

The DX register must first be loaded with the address of the output port. This is done with the instruction

MOV DX, B000H

Next, the data that are to be output must be loaded into AL with the instruction

MOV AL, FFH

لازم ال data تكون بال AL

Finally, the data are output with the instruction

OUT DX, AL

اذا غيرنا بدي memory

Mov [DS:B000] ~~[B000]~~, AL
Physical

او

MOV DS, 0000H
Mov [DS:B000], FF
لا يصلح (لازم يكونوا جوا ال [ ] يكون offset)

8.17 Input/Output Instructions

- EXAMPLE

Data are to be read in from two byte-wide input ports at addresses AAH and A9H and then output as a word-wide output port at address B000H. Write a sequence of instructions to perform this input/output operation.

in

8

16

special

- Solution:

First read in the byte at address AAH into AL and move it into AH.

```
IN AL, AAH
MOV AH, AL
```

عشان لا استخدم ال AL مرة ثانية تكون

IN AH, AAH

من الاصل

Now the other byte can be read into AL by the instruction

```
IN AL, A9H
```

And to write out the word of data

```
MOV DX, B000H
OUT DX, AX
```

word

او

IN AX, A9

الداتا الموجودة بال A9 تخزنت بال AL والموجودة بال AA تخزنت بال AH بتروح بحالة كلنوا دا بحوي
ال address
تكمل الباقي

* From main memory

MOV BX, [A9]

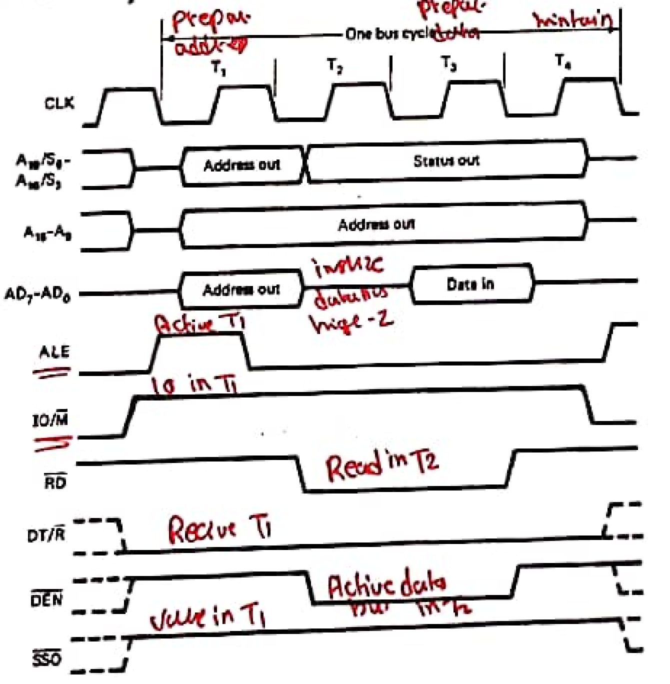
MOV [B000], BX , MOV [B000], [A9] not allowed memory to memory

8.18 Input/Output Bus Cycle

اذا قصده ع ال memory mapped
يكون memory input

Isolated

□ Input bus cycle of the 8088



in T1
address, ALE
IO, Receive, SSO

in T2
Initilize data bus
high-Z
Read

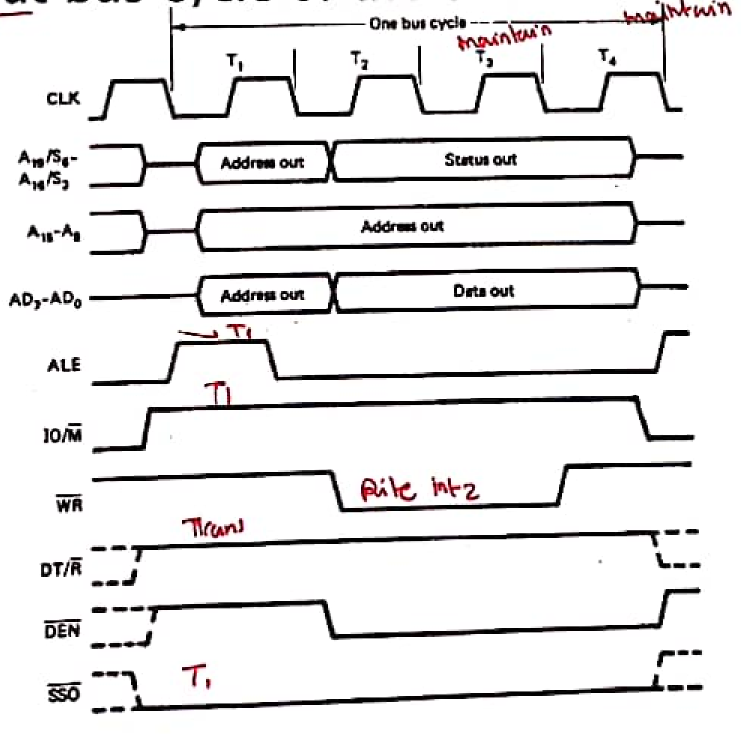
8.18 Input/Output Bus Cycle

□ Input bus cycle of the 8088

- Input (I/O read) bus cycle timing diagram—shows relationship between signals relative to time states
  - T1 state—input cycle begins
    - Address output on A0-A15
    - Pulse produced at ALE--address should be latched in external circuitry on trailing edge of ALE
    - IO/M* set to 1→ I/O bus cycle
    - DT/R* set to 0 → set external data bus control circuitry for receive mode (input)
  - T2 state
    - Status code output on S3-S6
    - AD0 through AD7 tri-stated in preparation for data bus operation
    - RD* set to 0 → input cycle
    - DEN* set to 0 → enable external data bus control circuitry
  - T3 state
    - Data on D0-D7 input (read) by the MPU
  - T4 state—input cycle finishes
    - RD* returns to 1→ inactive level
    - Complete address/data bus tri-stated
    - IO/M* returned to 0 → memory bus cycle
    - DEN* returned to 1→ inactive level
    - DT/R* returns to 1→ transmit level

8.18 Input/Output Bus Cycle

□ Output bus cycle of the 8088



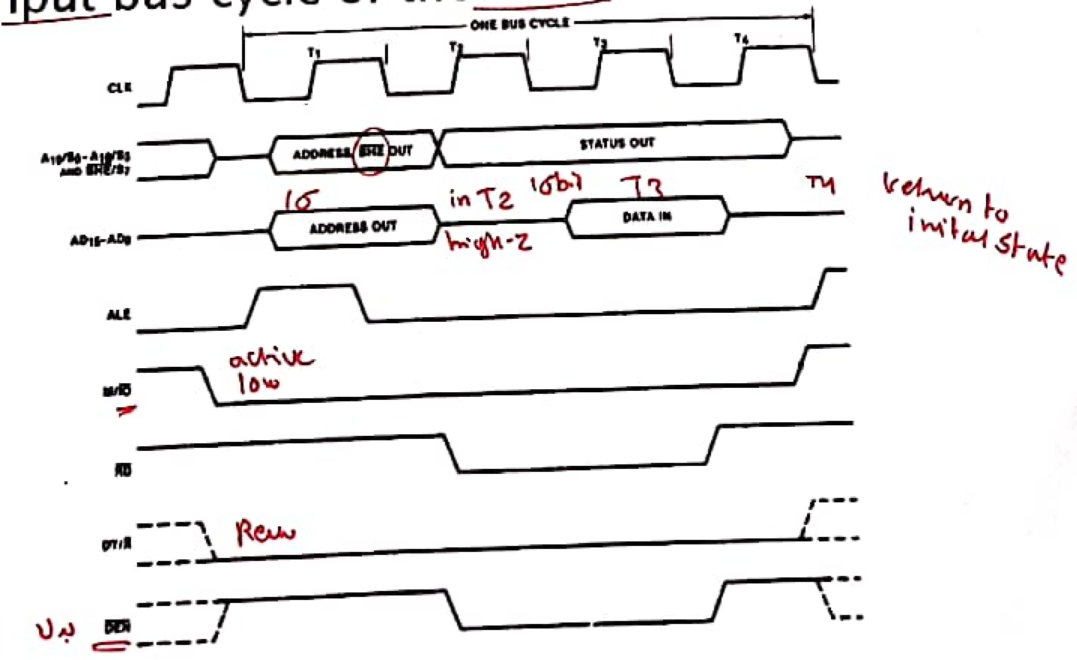
8.18 Input/Output Bus Cycle

□ Output bus cycle timing diagram of the 8088

- T1 state—output cycle begins
  - Address output on A0–A15
  - Pulse produced at ALE and address latched in external circuitry on trailing edge of ALE
  - IO/M* set to 1→ I/O bus cycle
  - DT/R* set to 1→ external data bus control circuitry for transmit mode (output)
- T2 state
  - Status code output on S3–S6
  - AD0 through AD7 transitioned to data bus and output data placed on bus
  - DEN* set to 0 → enable external data bus control circuitry
  - WR* set to 0· output cycle
- T3 or T4 state
  - Data on D0–D7 output (write) into I/O port (I/O device decides when!)
- T4 state—output cycle finishes
  - WR* returns to 1→ inactive level
  - Complete address/data bus tri-stated
  - IO/M* returned to 0 → memory bus cycle
  - DEN* returned to 1→ inactive level

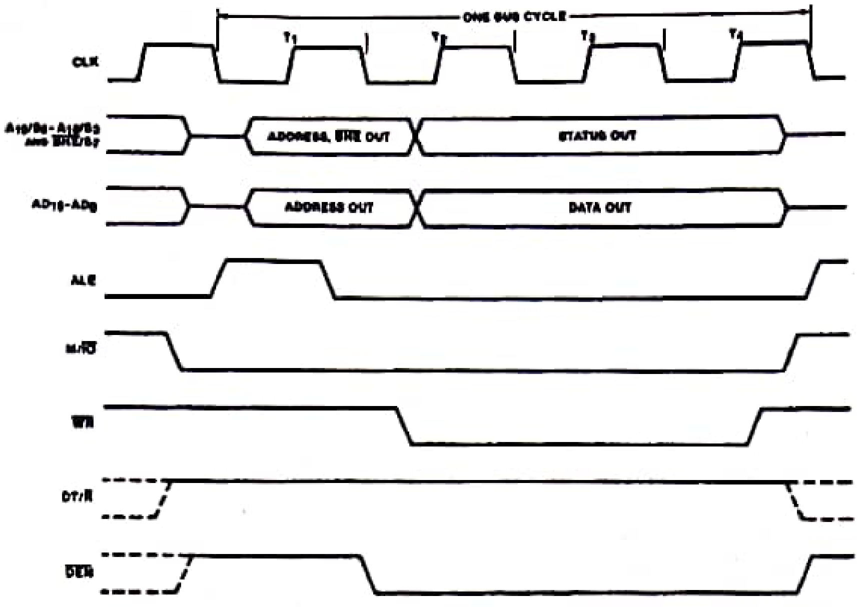
8.18 Input/Output Bus Cycle

□ Input bus cycle of the 8086



8.18 Input/Output Bus Cycle

- □ Output bus cycle of the 8086



H.W. #8

- ❑ Solve the following problems from Chapter 8 from the course textbook:

8, 22, 26, 35, 39, 49, 55, 59, 66, 86, 89, 99, 101, 107

Chapter 9

Memory Devices, Circuits, and Subsystem Design

Introduction

9.1 Program and Data Storage Memory—

9.2 Read-Only Memory—

9.3 Random Access Read/Write Memories—

9.4 Parity, Parity Bit, and Parity-Checker/Generator Circuit

9.5 FLASH Memory

9.6 Wait-State Circuitry—

9.7 8088/8086 Microcomputer System Memory Interface Circuitry—

memory unit of microprocessor consist of two part

① Primary storge memory

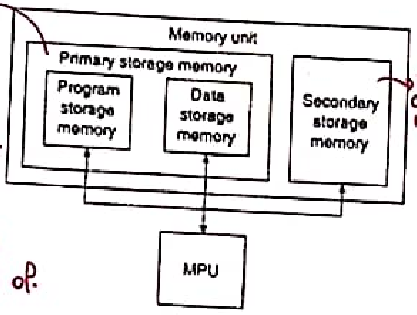
② secondary " "

9.1 Program and Data Storage Memory- The Memory Unit

Memory—provides the ability to store and retrieve digital information
- Instructions of a program
- Data to be processed
- Results produced by processing

main memory —> Primary storage memory

Organization of the Microcomputer memory unit
- Secondary storage—stores information that is not currently in use
  - Slow-speed
  - Very large storage capacity
  - Implemented with magnetic/optical storage devices—in PC
    - Hard disk drive , ssd , DVD
    - Floppy disk drive
    - Zip drive
- Primary storage—stores programs and data that are currently active
  - High-speed
  - Smaller storage capacity
  - Implemented with semiconductor memory
- Partitioning of Primary Storage
  - Program storage memory—holds instructions of the program and constant information such as look-up tables
    - EPROM (BIOS in PC)
    - FLASH memory
    - DRAM (volatile code storage in a PC)
  - Data storage memory—holds data that frequently changes such as the information to be processed by a program
    - SRAM
    - DRAM (PC)



all data and program are currently use by microprocessor.

EX: RAM and RoM

Primary consist of two part.

program storge ROM

Data storge (proces, modifiy, chang) RAM

Data can be

secondary larger than Primary

" slower " "

" low cost " "

9.2 Read-Only Memory- Types

charictrist:

code/inst

① non-volatily غير متطاير
the data can't be lost if I turn off the power supply

② use for read data only not for modifiy data

③ access RoM sequntialy

Read-only memory (ROM)
- Used for storage of machine code of program
- Stored information can only be read by the MPU
- Information is nonvolatile—not lost when power turned off
- Types: تبرمج ، يكتب
  - *ROM—mask-programmable read only memory
    - Programmed as part of manufacturing process
    - Lowest cost
    - High volume applications , Video game
  - *PROM—one-time programmable read-only memory → when we buy this type of ROM بتكون ال data empty the user program it just one time. using speical divice
    - Permanently programmed with a programming instrument
  - *EPROM—erasable programmable read-only memory
    - Programmed like a PROM
    - Erasable by Ultraviolet light
- Electrically alterable ROM-like devices
  - FLASH memory USB
  - EEROM (E_2ROM)

data will be stored in this type of RoM will be done during the Manfuction improcess

proplem: if there is error can't be program after munfacturing انه فقط برمج فلا مناعاد

* EEPROM

data store in this RoM can be eras but use speical device (ultraviolet)

① For program

② For eras

③ For reading

I can program this using my laptop ما بستخدم speical

EPROM : مثلا لو نتكت الفلاشة المعلومات لسة موجودة ما راحت

اما لو نتكنا ال RAM كل المعلومات رح تروح

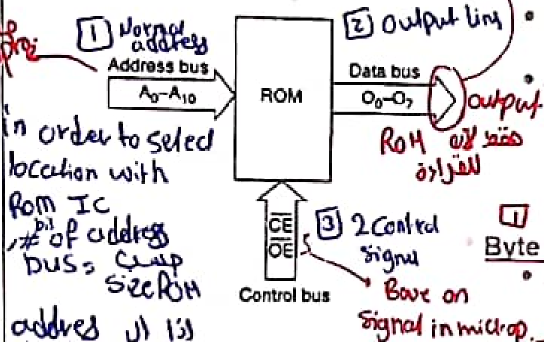
ROM/RAM ← main memory ال
1M حجم كل
2K اذا كان IC واحدة
(Single IC)

9.2 Read-Only Memory- Block Diagram

Block diagram of the ROM, PROM, and EPROM are essentially the same

Signal interfaces

- Address bus (A10-A0)—MPU inputs address information that selects the storage location to be accessed
- Data Bus (D7-D0)—information from the accessed storage location output to be read by MPU
- Control bus—enables device and/or enables output from device
  - CE* = chip enable—active 0; 1 low-power stand by mode
  - OE* = output enable—active 0; 1 high-Z state → disable output line



address of the bus of microp. — 1 Normal address
2 output line — connect to the Data bus of microp. — output — ROM فقط للقراءة
3 2 Control signal → Base on signal in microp.

in order to select location with Rom IC, # of address bit bus = chip size ROM

اذا ال address bus = 11
$2^{11} = 2^{1} \times 2^{10} = \boxed{2k}$ location size ROM

* # of bit output lines = عدد ال bit لكل ~~[illegible]~~ location

8bit → 2k byte size of ROM
↳ 8bit

1 - Byte capacity- number of bytes a device can store → maximum number of byte that store in memory
- Calculated from number of address bits

EX: Address = 11-bit address
2 Storage capacity = 2^{11} = 2048 bytes

- Organization—how the size of a ROM is described → # of location X # of bit per location
  - Formed from capacity and data bus width 1k = 1024

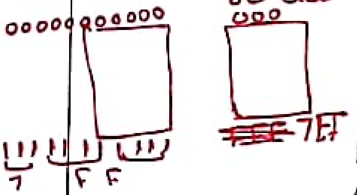
3 EX: 2048 X 8 or just 2K X 8

- Storage density—number of bits of storage in a ROM
  - Calculated from byte capacity and data width

EX: Storage density = 2048 X 8 = 16384 bits (16K bits)
هو حاصل ضرب ال organized ويكون بال bit

① enable chip by CE
② access location using access line
③ enable output line OE
④ read data using out-put line to the memory Interface cct

4 ~~address range~~ = address of first location to address of last "



9.2 Read-Only Memory- Organization and Capacity

Example:

A ROM device has 15 address lines and 8 data lines. What are the address range, byte capacity, organization, and storage density?

لو غيرنا 8 data line → 4 data line
16 k byte Byte capacity

Solution:

- Address range
  A14-A0 = 000 0000 0000 0000_2 - 111 1111 1111 1111_2
  = 0000H - 7FFFH
- Byte capacity
  2^{15} = 32,768 bytes = 32K bytes
- Organization
  32,768 X 8 bit
- Storage density
  32,768 x 8 = 262144 bits = 256K bits

$= 2^9 = $ IC ال هيكي عدد
<u>512</u>

memory ال منه
بعدين انه لازم يكمل
enable chip

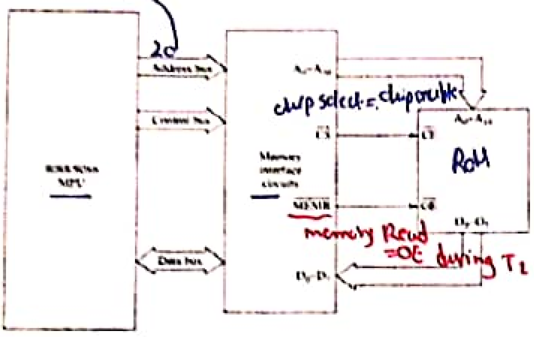
512x2K
= <u>1M</u>
size
ال كل
memory
هلو

00000 0000
select ←
IC 0
0000 0001 ←
IC 1
⋮

ويرتين ٩ مدرس علمي decod
to Select IC (for CS)
chip select
during T1

اول الـ to acce
location of RoM

9.2 Read-Only Memory- Operation



Read operation
- MPU outputs address and control information on its bus.
- Interface circuit applies Address A10-A0 to the address inputs of the ROM to select a specific byte wide storage location
- Interface circuits decode additional address bits to produce a chip select output
- Logic 0 at CS* applied to the CE* input of the ROM to enable it for operation
- Memory interface circuitry produces appropriately timed MEMR* output
- MEMR* applied to OE* input of the ROM to enable the information at the addressed storage location onto the output bus D7-D0
- Memory interface supplies the byte of data from the ROM to the MPUs data bus
- MPU reads the byte of data from the ROM from its data bus

9.2 Read-Only Memory- Standard EPROM ICs

EPROM part numbers formed by adding the prefix "27" to the device total Kbits of storage capacity
- Examples:
  - 16K bit EPROM · 2716
  - 32K bit EPROM · 2732
  - 1M bit EPROM · 27C010
- Most EPROM available in byte wide organization
  - Examples:
    - 2764 · 8K X 8
    - 27C020 · 256K X8
- NMOS versus CMOS process
  - Manufacturing processes used to make EPROMs
    - NMOS=N-channel metal-oxide semiconductor
    - CMOS= complementary symmetry metal-oxide semiconductor
    - "CMOS" designated by "C" in part number
  - NMOS—older devices such as 2716 and 2732
  - CMOS—all newer devices 27C64 and up

$2k \rightarrow 2^{11} = \boxed{11}$ address bit

organisation

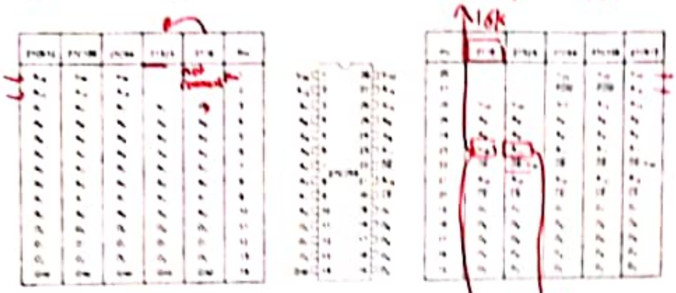
27 بداية
و النهاية عن ال density
كلهم output-line 8
Cmos

EPROM	Density (bits)	Capacity (bytes)
2716	16K	2K × 8
2732	32K	4K × 8
27C64	64K	8K × 8
27C128	128K	16K × 8
27C256	256K	32K × 8
27C512	512K	64K × 8
27C010	1M	128K × 8
27C020	2M	256K × 8
27C040	4M	512K × 8

$\frac{16}{8} = 2$
$\frac{32}{8} = 4$

9.2 Read-Only Memory- Pin Layouts

ال stage بغير ال control


vpp for power suply
16k
not connected
استخدمنا 1,2

16k → 32k
بزيد خط ال address line
↓ 8k ↓ 4k
11 12

الفرق انه بيزيد one additional address bit

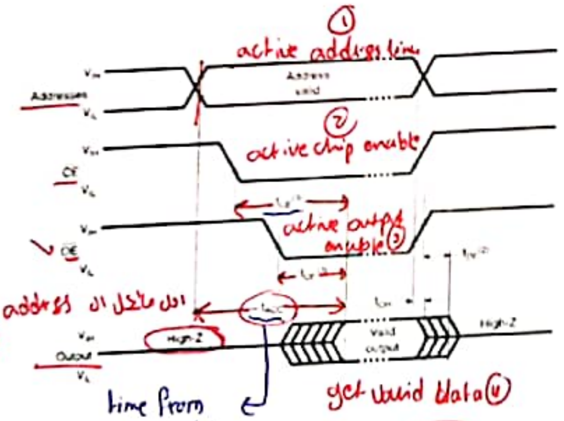
multiplexed of chip

- EPROM pin layouts are designed for compatibility
  - Permit easy upgrade from lower to higher density
  - Publish pin layouts of future densities
  - Allows design of circuit boards to support drop in upgrade to higher densities
- Most pins are independent and serve a common function for all densities
  - Examples:
    - pin 10- A0
    - pin 11--O0
    - pin 14- Gnd
- Some have one multi-function pin* OE*/Vpp
  - Vpp mode during programming only

9.2 Read-Only Memory- EPROM Switching Waveforms

general diffrent signal

Timing diagram to access valid information



من اول ما اعمل ال address لحد ما يكون في valid data

time from generat address until to get data
tacc (access time)
الوقت من اول ما اعمل generat address لحد ما يكون في data

tCE
chip enable time
الوقت من اول ما اعمل ال chip enable لحد ما يكون في data يعني
between active chip enable until I get valid data

- Timing of the read operation
  - Output data is not immediately available at the outputs
    - Delays exist between the application of the address,CE* and OE* signals and the occurrence of a valid output
    - tacc= access time—address to valid output delay time
    - tCE= chip-enable time—chip enable to valid output delay
    - tOE=output-enable time—output enable to valid data delay
  - To assure that the MPU reads valid data, these inputs must be applied at the appropriate times
    - Responsibility of the memory interface circuitry
  - Another delay occurs at the removal of OE* before the outputs lines are returned to the high-Z state
    - tDF= chip-deselect time—time for the outputs to recover

(time between deactive output enable until the data covabled)
in same time ~ tCE , tol

tOE
output enable time
time difference between active output enable to ~~active~~ get valid data

عبارة عن الوقت من اول ما اعمل active لحد ال data

(زيادة الlocation)

increase # of byte that I can store in memory

①

9.2 Read-Only Memory- Expanding Byte Capacity

4H الsize انا حدنا لكم
2 design يوجلان عندي
لكن نزيد الحجم

→ system need more 32k location need to store up to 64k byte ⇒ memory design

- Many applications require more ROM capacity than is available in a single device
  - Need more bytes of storage
  - Connects to a wider data bus
- Expanding byte capacity with 2 EPROMS
  - Connect address bus lines in parallel
  - Connect output lines in parallel
  - Connect OE* in parallel
  - Enable chips with separate chip selects
    - Address bit A15 decoded to produce CS0* and CS1*
      - A15=0 · CS0*
      - A15=1 · CS1*
      - Implemented with inverting buffer
- Byte capacity
  - 2^{16} = 64K bytes
- Organization
  - 64K X 8 bit
- Storage density
  - 2 X 32K x 8 = 512K bits

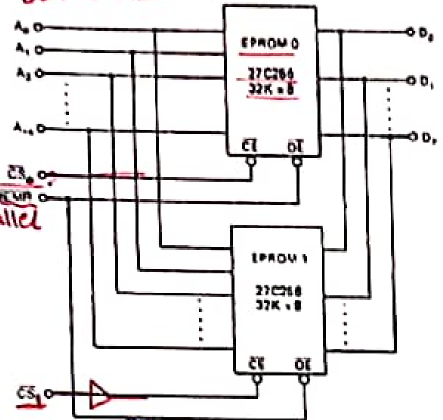
بدون زيادة عدد ال out

address لما range] افصلنا بينهم

ازا حصل لازم ارا remaining كل ال

بدل عليه الموجود

32k location
8bit بيحتفظ بال location



parallel

① divide needed size over size of single Ic

$$\frac{64k \times 8}{32k \times 8} = \underline{\underline{2}}$$ number of Ic 32kx8 كل وحدة

① اول شي لازم نعرف عدد ال Ic

3 group وكل وحدة بحفظها
address → 15 address line for 32kx8
out → 8
control → 2

② ثاني شي نحدد ال 3group

③ connect (in parrallel) same line together [address], ~~out~~, ~~control~~ same for Ic

remaing		need
	1 ←	2
	2 ←	4
	3 ←	8

Connect Signal chip
↳ Expanding → 2 location مفهوم اقرأ من
بنفس الوقت يعني انا طلبت فقط واحد enable line للبنى وفتح وقرات ال output رح يعطي خطأ كانه جهاز 8bit

2 chip select (in series) different line
remaining address line يلي عم decode بعمل

2 different chip select so I need one line
A15 0 → CS0
A15 1 → CS1 } inverter
② بتبدل ع invert

9.2 Read-Only Memory- Expanding Word Length

فاهمه بال 8086

increase number of output line
مابدو يزيد عدد ال location

- Expanding word length with 2 EPROM
  - Connecting to 8086 16-bit data bus
    - Connect address bus lines in parallel
    - Connect CE* in parallel
    - Connect OE* in parallel
    - 8 data outputs of EPROM 0 used to supply the lower data bus lines D0-D7
    - 8 data outputs of EPROM 1 used to supply the upper 8 data bus lines D8-D15
  - Byte capacity
    - 2 X 2^{15} = 64K byte
  - Organization
    - 32K X 16 bit
  - Storage density
    - 32K x 16 = 512K bits

① determine number of Ic

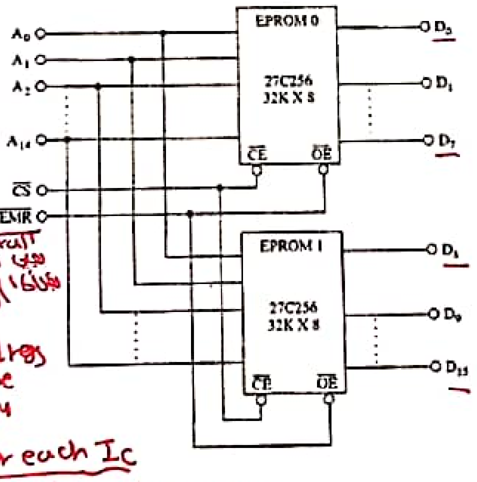
den $\frac{32k \times 16}{32k \times 8} = \boxed{2}$
ou

② determine # of each group

Address ⇒ 15 address line (parrallel) A₀ - A₁₄

output ⇒ 8 for each Ic

control ⇒ 2



parrallel نفس الخط للاثنين

ما زادت الlocation
زدت عدد ال bit

↳ design ال طالب 16

لازم نزيده
16 output line

Connect in parraller.
(same line, because (Connect in series, different size) I need to read from 2 location)

remaining time 5 → just 6 نستفد
address range كاملة

* MEMR : اذا طبقنا كل واحد لحال
معناته ال flexibity بتزيد
يعني نقدر نقرأ اول 8 او ثاني 8
او الثنين سوا
انا بدي اياه fixed يعني parallel

9.3 Random Access Read/Write Memories-

difference RAM by ROM:

① Read and write
② Volatile
③ access randomly : I can access any location مو شرط تكون sequentidy

Types of RAMs

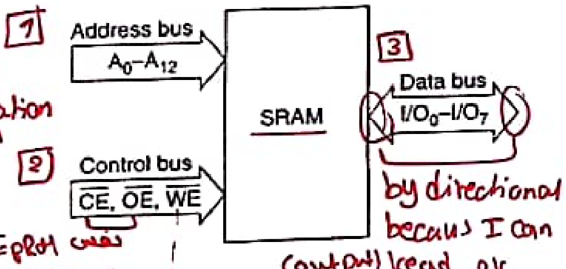
Random Access Read/Write Memory (RAM)

- Used for temporary storage of data and program information
- Stored information can be altered by MPU—read or written
  - Information read from RAM
  - Modified by processing
  - Written back to RAM for reuse at a later time

اذا modified لو data or change نخزنها بال RAM

- Information normally more frequently randomly accessed than ROM
- Information is volatile— lost when power turns off
- Types: base on material
  1. Static RAM (SRAM)— data once entered remains valid as long as power supply is not turned off
     For valid data → *keep power supply ON
     implement by transistor TTL
     - Lower densities (Small size)
     - Higher cost
     - Higher speeds
  2. Dynamic RAM (DRAM)—data once entered requires both the power to be maintained and a periodic refresh
     For valid data → 1 keep power supply ON and 2 make periodic refreshig
     implent by capacitor
     refreshig: recharging for data in capacitor
     السبب انه ال capacito مع الوقت بيصير discharge
     - Higher densities
     - Lower cost
     - Lower speeds
     - Refresh requires additional circuitry

9.3 Random Access Read/Write Memories-
SRAM Block Diagram



I can access up to 2^{13} location 8K location

by directional because I can (output) read or write (input)

Size 8K × 8

Eplot تفعيل enable out enable chip

تحديد نوع ال operation
0 → write (input Data bus)
1 → read (output Data bus)

عدد = عدد ال bit لكل location
8bit

Signal interfaces

- Address bus (A12-A0)—MPU inputs address information that selects the storage location to be accessed
- Data Bus (I/O7-I/O0)—input/output of information for the accessed storage location from/to MPU
- Control bus—enables device, enables output from device, and selects read/write operation
  - CE* = chip enable—active 0
  - OE* = output enable—active 0
  - WE* = write enable
    - 0 = write to RAM
    - 1 = read from RAM

byte capacity → 8kbyte
organization → 8k × 8 ⇒
storage → 64k

address range → ~~000~~0 0000 0000 0000
0 0 0 0 — 1FFF

9.3 Random Access Read/Write Memories- Standard SRAM ICs

SRAM	Density (bits)	Organization
4361	64K	64K × 1
4363	64K	16K × 4
4364	64K	8K × 8
43254	256K	64K × 4
43256A	256K	32K × 8
431000A	1M	128K × 8

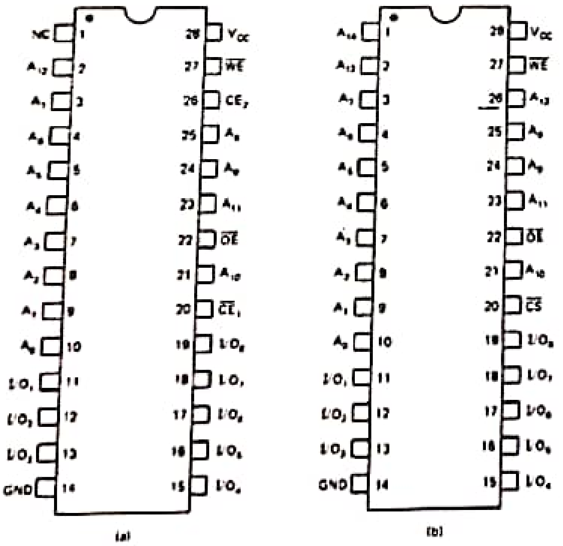
Part numbers vary widely by manufacturer—Hitachi/NEC use "43xxx

SRAMs are available in a variety of densities and organization

- Typical SRAM densities
  - 64K bit
  - 256K bit
  - 1M bit
- Typical organizations of the 64K bit

SRAM
- 64K X 1 bit
- 16K X 4 bit
- 8K X 8 bit

9.3 Random Access Read/Write Memories- Pin Layout of SRAMs



4364 and 43256A pin layouts are designed for compatibility

4364 pin configuration (Fig a)

- A12-A0 · 13-bit address

  2^{13} = 8K bytes
- I/O7-I/O0 · byte wide
- Pin 1 NC = no connect
- Pin 27 WE*
- Pin 20 CE1* · active 0
- Pin 26 CE2 · active 1
- Pin 22 · OE*
- Pin 28 Vcc
- Pin 14 GND

43256A differences (Fig b)

- Pin 1 · A14
- Pin 26 · A13
- Pin 20 called CS* (function unchanged)

9.3 Random Access Read/Write Memories- Expanding Word-Width and Capacity

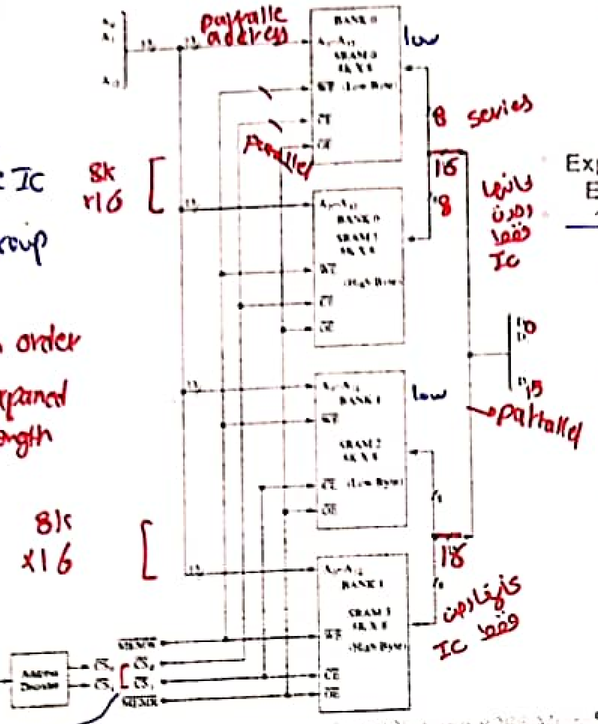
① request size

size for single IC

② ~~No of~~ find group

③ connect

$\frac{16K * 16}{8K * 8} = \boxed{4}$ in order to expand word length



word length

word leng

capacity يكبر

2 CS

chip enable for first bank different than chip ~~enable~~ for second bank

A13

Most SRAM subsystems first

- Require both word-width and bit capacity expansion second
- Require the ability to write on byte-wide or word wide basis- design only supports words

Expansions performed in a similar way as for EPROMs

16K X 16-bit SRAM circuit 8k x 8

- A0-A12 in parallel
- A13 decoded to form CS0* and CS1*
  - CS0* enable Bank 0
  - CS1* enable Bank 1
- SRAMs 0 & 2—input/outputs connected in parallel and supply low byte of data bus
- SRAMs 1 & 3—input/ outputs connected in parallel and supply high byte of data bus
- MEMW* and MEMR' produces independent write and read enables

MEMW*	MEMR*	Data Transfer
0	0	Invalid
0	1	Word write
1	0	Word read
1	1	Inactive

- How can the circuit be modified to support byte wide write?

9.3 Random Access Read/Write Memories- Standard Read/Write Cycle Times

الاسرع →

Part number	Read/write cycle time
4364-10	100 ns
4364-12	120 ns
4364-15	150 ns
4364-20	200 ns

different speed

Speed of a SRAM identified as read/write cycle time

- Variety of speeds available—4364 available in speeds ranging from 100ns to 200ns
- Shorter the cycle time the better.

Designated by a dash speed indicator following the part number

-10 = 100ns

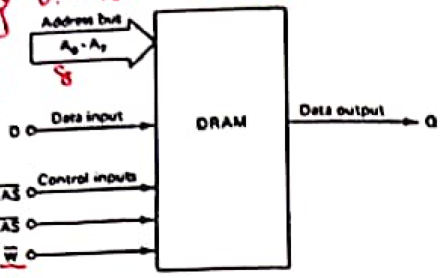
-12 = 120ns

9.3 Random Access Read/Write Memories-DRAM Block Diagram

A0 → 0, 8
a1 → 1, 9
A7 → 7, 15

2 carry → 8 → 2^8 $\boxed{16}$
[illegible]
$\boxed{1}$

2^{16} = size



multiplex the same line can carry different address inform.

first group ← $\overline{RAS}$

second group ← $\overline{CAS}$

write ← 0
read ← 1

DRAM signal interfaces

- Address multiplexed in external circuitry into a separate row and column address
  - Row address = A_7-A_0
  - Column address = A_{15}-A_8
- Special RAS* and CAS* inputs used to strobe address into DRAM
- Row and column addresses applied at different times to address inputs A_0 through A_7
  - Row address first
  - Column address second
  - Known as "RAS before CAS"
  - Address reassembled into 16-bit address inside DRAM
- Frequently data organizations are X1, X2, and X4
  - Separate data inputs and outputs
  - Data input labeled D
  - Data output labeled Q
- Read/write (W) input signals read or write operation

9.3 Random Access Read/Write Memories-Standard DRAM ICs

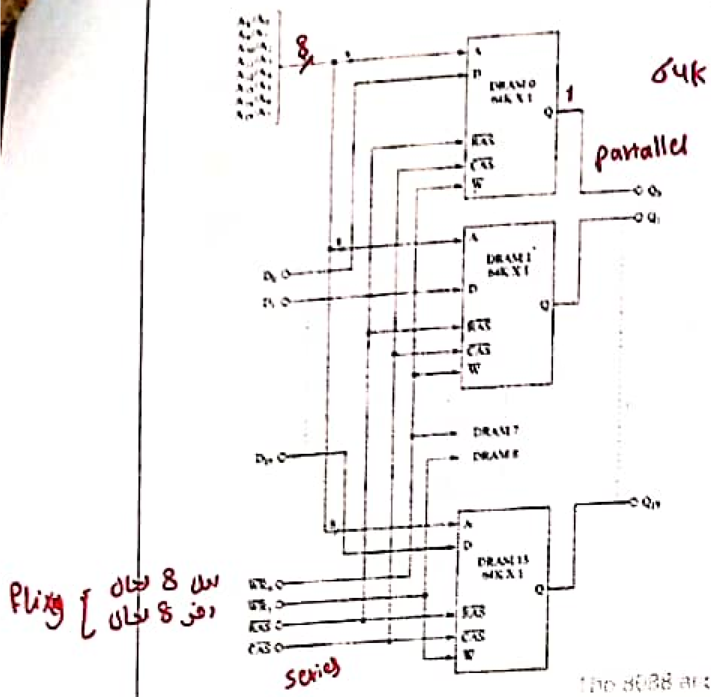
start number   O/L

DRAM	Density (bits)	Organization
2164B	64K	64K × 1
21256	256K	256K × 1
21464	256K	64K × 4
421000	1M	1M × 1
424256	1M	256K × 4
44100	4M	4M × 1
44400	4M	1M × 4
44160	4M	256K × 16
416800	16M	8M × 2
416400	16M	4M × 4
416160	16M	1M × 16

DRAMs are available in a variety of densities and organization

- Typical DRAM densities
  - 64K bit
  - 256K bit
  - 1M bit, Etc.
  - Modern DRAMS as large as 1G bit
- Typical organizations of the 4M bit DRAM
  - 4M X 1 bit
  - 1M X 4 bit
  - Modern higher density devices also available in X8, X16, and X32 organizations

9.3 Random Access Read/Write Memories-Circuit Design using DRAMS



64K X1

Sixteen 64KX1-bit DRAMs interconnected to form a 64K word memory subsystem—1M-bits of memory

64k×16 Circuit connections

- 8 multiplexed address inputs of all devices connected in parallel
- RAS and CAS lines of all devices connected in parallel
- Data input and output lines
  - Independent data lines arranged to form a 16-bit wide output bus
  - Independent input lines arranged to form a 16-bit wide input bus
  - In most microprocessor applications input and output lines are connected together
- Read/write lines
  - W inputs of upper 8 DRAMs connected together and driven by WR0*
  - W inputs of lower 8 DRAMs connected together and driven by WR1*
  - Permits byte-wide or word-wide reads and writes

9.3 Random Access Read/Write Memories

- □ The primary memory section of a microcomputer system is normally formed from both read-only memories and random access read/write memories (RAM)
- □ RAM is different from ROM in two ways:
  - □ Data stored in RAM is not permanent in nature.
  - □ RAM is volatile - that is, if power is removed from RAM, the stored data are lost.
- □ RAM is normally used to store temporarily data and application programs for execution.

9.3 Random Access Read/Write Memories

- □ Static and dynamic RAMs
  - □ For a static RAM (SRAM), data remain valid as long as the power supply is not turned off.
  - □ For a dynamic RAM (DRAM), we must both keep the power supply turned on and periodically restore the data in each location.
  - □ The recharging process is known as *refreshing* the DRAM.

③ location not determine

dis :

① can detect just odd number of error even(✗)

② odd number can detective but I can't know the number

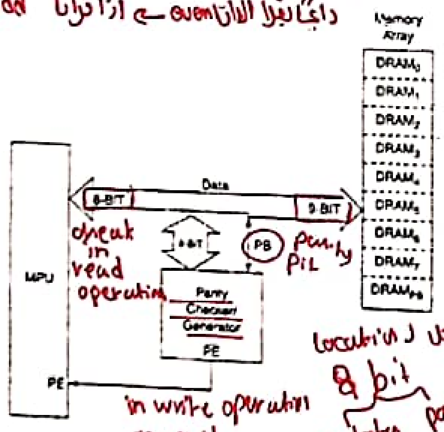
9.4 Parity, The Parity Bit, and Parity-Checker/Generator Circuit- Parity and the Parity Bit

Data exchange between the MPU and data memory subsystem in a microcomputer must be done without error

- Sources of errors
  - Emissions that affect data on the data bus line
  - Electrical noise signals—spikes or transients that affect data on data lines
  - Defective bit in a DRAM
  - Soft errors of DRAM
- Solutions for improving data integrity
  1. Parity
  2. Error correction code (ECC)

  Parity most frequently used
- Parity bit
  - Add an additional bit of data to each byte or word of data so that all elements of data have the same parity
  - Extra bit is known as the "parity bit"
    - → Even parity—element of data has an even number of bits at the 1 logic level
    - → Odd parity—element of data has an odd number of bit that are logic 1
  - Circuitry added to the DRAM memory interface to generate an appropriate parity bit on writes to memory
  - Extra DRAM required to store the parity bit
  - Circuitry checks element of data from correct parity during read operations
  - Parity errors (PE) reported to MPU usually as an interrupt

دايماً بنقرأ الآانات even ← اذا قرأنا odd (error)



error ← 0

no error ← 1

لكل location 9 bit

8 data parity bit

error detection add extra bit with data (function of original data)

destination separate بين ال original و ال extra and pass origial

parity bit

# of ones in data — even / odd

base on parity we add one extra bit (parity bit)

even → 0 بجيب بير ⇝ even

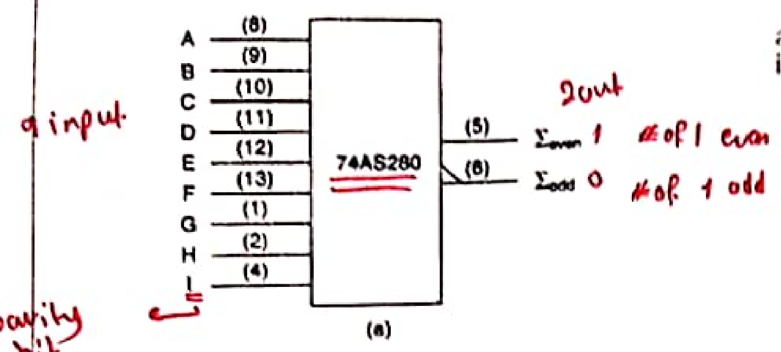
odd → 1 بجيب بير ⇝ even

في العكس

even → 1 ⇝ odd

odd → 0 ⇝ odd

9.4 Parity, The Parity Bit, and Parity-Checker/Generator Circuit- Parity Generator/Checker Circuitry



NUMBER OF INPUTS A THRU I THAT ARE HIGH	OUTPUTS	
	Σ EVEN	Σ ODD
0,2,4,6,8	H	L
1,3,5,7,9	L	H

(b)

Parity generator/checker circuit—circuit added to the data memory interface to implement parity

- May be implemented with a 74AS280 parity generator/checker IC
  - 9 inputs A through I
  - Two outputs Σodd and Σeven
  - Operation:
    - Even number of inputs are logic 1
      - Σeven = 1 and Σodd = 0
      - Signals that input has even parity
    - Odd number of inputs are logic 1
      - Σeven = 0 and Σodd = 1
      - Signals that input has odd parity

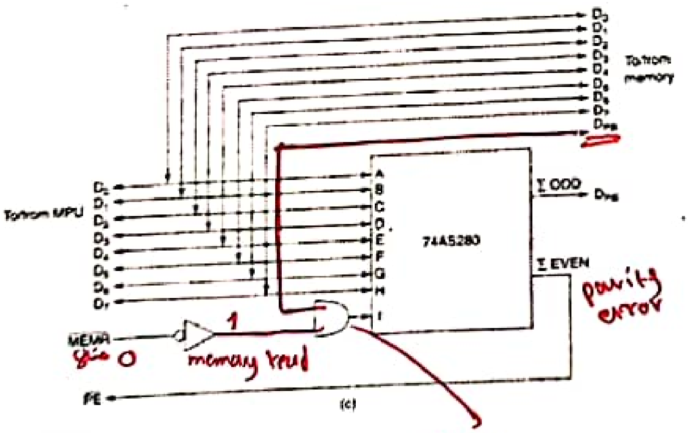
parity ~~als~~ data even → parity bit = $\boxed{0}$ → Σodd

" " even → " " = $\boxed{1}$ → Σeven

write Σodd

read Σeven

9.4 Parity, The Parity Bit, and Parity-Checker/Generator Circuit- Parity Generator/Checker Circuitry



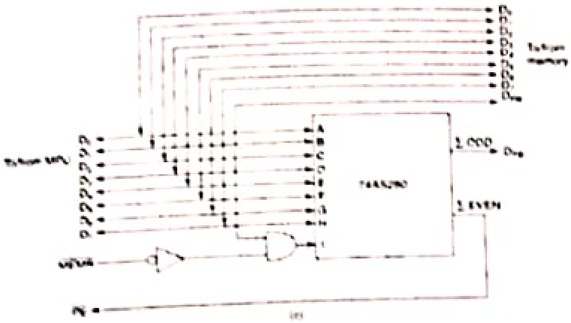
رح تكون ازا ح الستخدم ال parity bit

لل write ماستخدمه

Even parity generator circuit

- Circuit configuration
  - Inputs A through H attach in parallel to data bus lines D0 through D7
  - Input I is attached to the data output of the parity DRAM
    - Only activated during read operations
  - Σodd output is attached to the data input of parity DRAM
- MPU write operation
  - Accepts byte of data to be written to memory as input from the data bus
  - Data also applied in parallel to the input of the DRAMs for data lines D0 through D7
  - Circuit checks parity and generates Σodd and Σeven outputs
  - Σodd output supplied to input of parity DRAM for storage along with the byte in memory
  - If parity is even—Σodd = 0 and 9-bit value saved in memory still has even parity
  - If parity is odd—Σodd = 1 and parity of 9-bit value changed to even and saved in memory

9.4 Parity, The Parity Bit, and Parity-Checker/Generator Circuit- Parity Generator/Checker Circuitry



Read operation:

- Accepts 9-bit wide input from data outputs of the DRAM subsystem
- Checks the number of bits that are at the 1 logic level
- Produces appropriate logic level signals at odd parity and even parity outputs
  - If parity is even—Σeven = 1 and parity is correct
    - Memory operation completes normally
  - If parity is odd—Σeven = 0 and a parity error is detected
    - Error condition signaled to MPU by logic 0 at PE*
    - Usually applied as NMI input to the MPU
    - Must get serviced before executing next instruction
    - MPU may
      - Reattempt memory access
      - Initiate an orderly shut down of application

9.7 8088/8086 Microcomputer System Memory Circuitry

- □ Data storage memory
  - □ Information that frequently changes is normally implemented with random access read/write memory (RAM).
  - □ If the amount of memory required in the microcomputer is small, the memory subsystem is usually designed with SRAMs.
  - □ DRAMs require refresh support circuit which is not warranted if storage requirement are small.

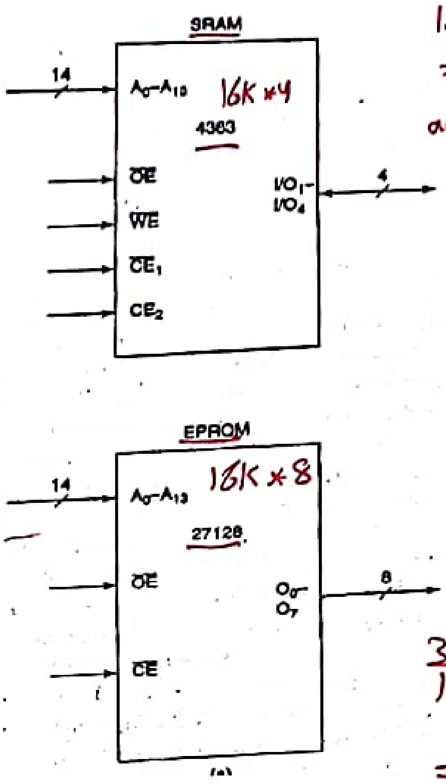
9.7 8088/8086 Microcomputer System Memory Circuitry

EXAMPLE 32k×8

Design a memory system consisting of 32Kbytes of R/W memory and 32Kbytes of ROM memory. Use SRAM devices to implement R/W memory and EPROM devices to implement ROM memory. The memory devices to be used are shown below. R/W memory is to reside over the address range 00000H through 07FFFH and the address range of ROM memory is to be F8000H through FFFFFH. Assume that the 8088 microprocessor system bus signals that follow are available for use: A_0 through A_{19}, D_0 through D_7, MEMR', MEMW'.

First part. ① design memory system

second part ② designed memory system in specific address range



$① \frac{32k \times 8}{16k \times 4} = 4$ ICs

address = 14

$\frac{32k \times 8}{16k \times 8} = \boxed{2}$

9.7 8088/8086 Microcomputer System Memory Circuitry

SOLUTION:

First let us determine the number of SRAM devices needed.

No. of SRAM devices = 32Kbyte/(16K x 4) = 4

To provide an 8-bit data bus, two SRAMs must be connected in parallel. Two pairs connected in this way are then placed in series to implement the R/W address range, and each pair implements 16Kbytes.

Next let us determine the number of EPROM devices needed.

No. of EPROM devices = 32Kbyte/16Kbyte = 2

These two devices must be connected in series to implement the ROM address range and each implement 16Kbytes of storage.

$0 + 2(16 \times 624 - 1) = 7FFE$

For 8086 → 2 bank High, low

لك عوني E IC موجودة بال low bank بتكون 16k

→ address of last location = Start address + 2 (# of location − 1)

لازم ننبه اذا ال IC نزيدوا 2-2

5 4
3 2
1 0

9.7 8088/8086 Microcomputer System Memory Circuitry

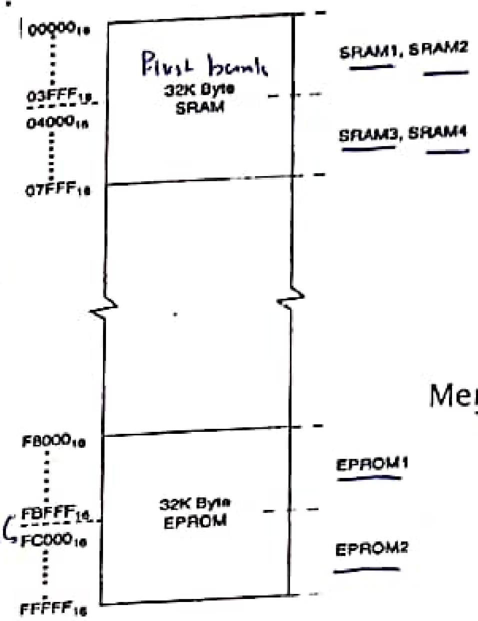
لان كان كل location لها Bank لا تكونوا نفس 8088

For 8088

address of last location = address of first location + # of location −1

لأنه دايمًا بلش من zero

SOLUTION:



2 bank

04 ~~000F~~

منكد بنبلش من 0000 وعندي 4 location

$0 + 4 - 1 = \boxed{3}$ but location

$0 + 16K - 1 =$

$0 + 16 \times 1024 - 1 = 16383$

$0011\ \boxed{\overset{1111}{3F}\ \overset{1111}{F}\ \overset{1111}{F}}$ +1

starting address for second bank 4000

$\underbrace{04000}_{\text{hexa}} + \underbrace{16 \times 1024 - 1}_{3FFF} = \boxed{7FFF}$

4 ÷ 3FFF

Memory map of the system

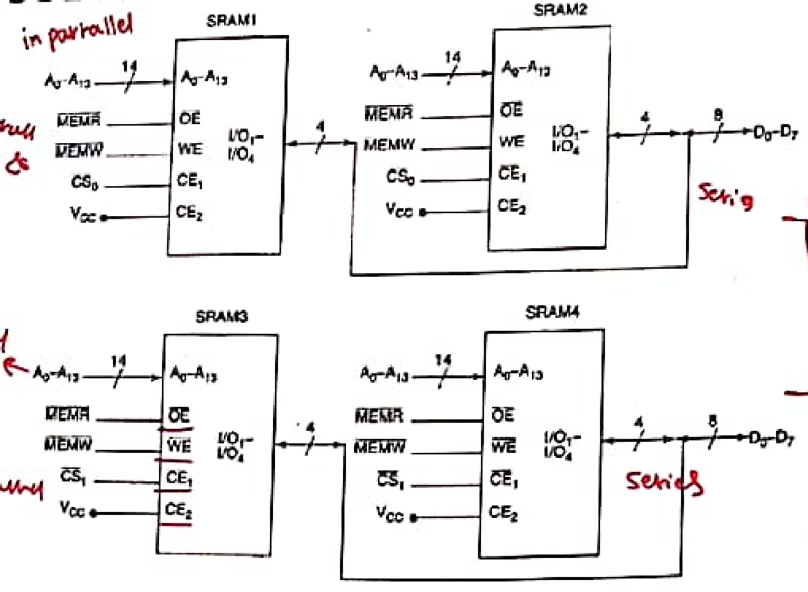
$F8000 + \underbrace{16 \times 1024 - 1}$

$F8000 + 3FFF = FBFFF$

$FC000 + \underbrace{16 \times 1024 - 1}_{3FFF} = FFFFF$

9.7 8088/8086 Microcomputer System Memory Circuitry

SOLUTION:



بشتبك كل 2 مع بعض

parrallel

بالمقارنة

32K x 8

وال cs in series

2cs → CS0, CS1

RAM memory organization for the system design

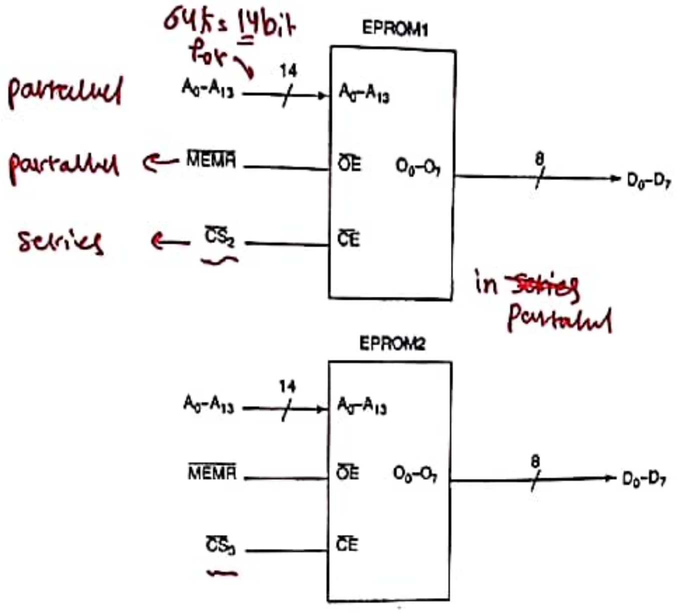
استخدمت 18 ← 6 remaining

لانه لازم استخدمهم كلهم

only one A19 ← 2CS

9.7 8088/8086 Microcomputer System Memory Circuitry

SOLUTION:



ROM memory organization for the system design 2 reming ← 4 chip select.

لبن لاه محدد range دلكل لازم الستخدم كلا8

00 → CS0
01 — CS1
10 — CS2
11 — CS3

9.7 8088/8086 Microcomputer System Memory Circuitry

SOLUTION:

address range mapped for 2Ic → in bank

address range

قرائة location لكل 2Ic من

3x8
3input
3 enable

$$A_{19}\cdots\cdots\cdots\cdots A_0$$

$$00000_{16} = 0000\ 0000\ 0000\ 0000\ 0000$$

bank1 ال location م
SRAM1 SRAM2

⋮ reming 14

$$03FFF_{16} = \underbrace{0000\ 00}_{\overline{CS}_0}11\ 1111\ 1111\ 1111$$

A14

+1

16k x 8
14
A13 — A0

$$04000_{16} = 0000\ 0100\ 0000\ 0000\ 0000$$

← Bank 2 start directly after Bank1

⋮

$$07FFF_{16} = \underbrace{0000\ 01}_{\overline{CS}_1}11\ 1111\ 1111\ 1111$$

هون فكي لازم البنى ال ROM في →

$$F8000_{16} = 1111\ 1000\ 0000\ 0000\ 0000$$

14 address

⋮

$$FBFFF_{16} = \underbrace{1111\ 10}_{\overline{CS}_2}11\ 1111\ 1111\ 1111$$

6 reming address line

+1

$$FC000_{16} = 1111\ 1100\ 0000\ 0000\ 0000$$

الافضل تستخدم 3x8 decoder.

14 address

⋮

$$FFFFF_{16} = \underbrace{1111\ 11}_{\overline{CS}_3}11\ 1111\ 1111\ 1111$$

لانه ال 2x4 ال 2input والعاerable.

3input و 3enable input.

Address range analysis for the design of chip select signals

address decoder لازم بعدها استخدم

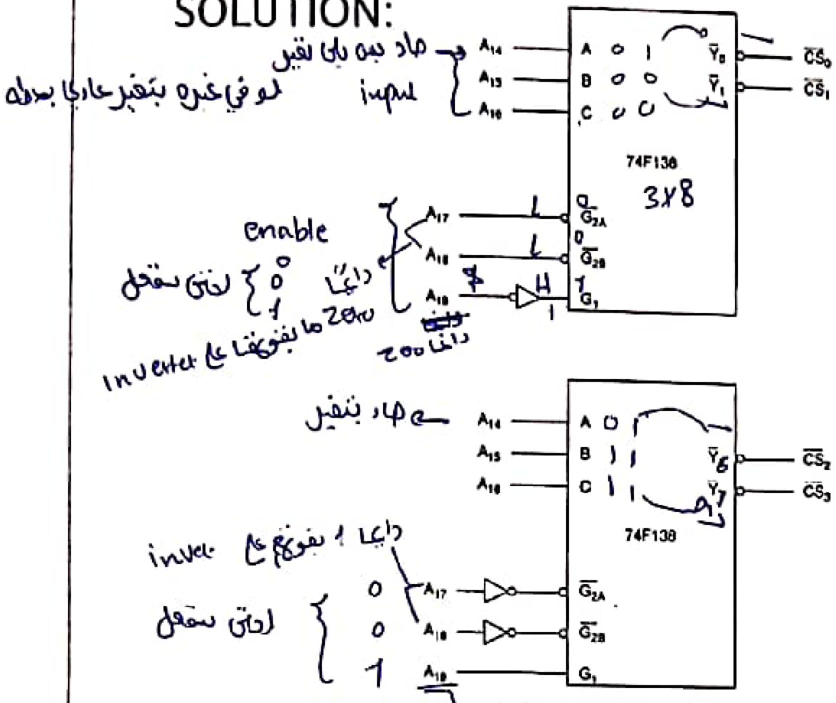
عدباتي بتغير بدخله عال input وباتي مابتغير ماله اي تأثير

مثلا 000001 ← انا ب Bank 2 عندي active CS1

RAM & ROM ايضا (ان الاختلافات كبيرة بيستخدم 2decoder

9.7 8088/8086 Microcomputer System Memory Circuitry

SOLUTION:



Chip-select logic

Microprocessor Systems

Chapter 10

Input/Output Interface Circuits and LSI Peripheral Device

Lecture Outline

- 10.1 Core and special-purpose I/O interfaces
- 10.2 Byte-Wide output ports using isolated I/O
- 10.3 Byte-Wide input ports using isolated I/O
- 10.4 Input/Output handshaking and parallel printer interface
- 10.5 the 8255 Programmable Peripheral Interface

10.1 Core and Special Purpose I/O Interfaces

we thesm if I need them

يعني مو شرط! كل comp
يكون فيه connect to ال
device

- * Special purpose I/O interfaces are implemented as add-on cards on the PC not nessesary
  - display
  - parallel printer interface
  - serial communication interface
  - local area network interface
  - not all microcomputer systems employ each of these types
- * Core input/output interfaces are considered to be the part of the I/O subsystem such as:
  - parallel I/O to read the settings of the DIP switches on the processor board
  - interval timers used in DRAM refresh process

it's nessesary to have it in any Computer system in order to work

We will study both

و اذا ما فيها timer ال
الداتا الي بالذاكرة
بال
DRAM
رح تضيع

10.2 I/O Design in the 8088/86 ~~need to use~~

I should use Interface circul ?

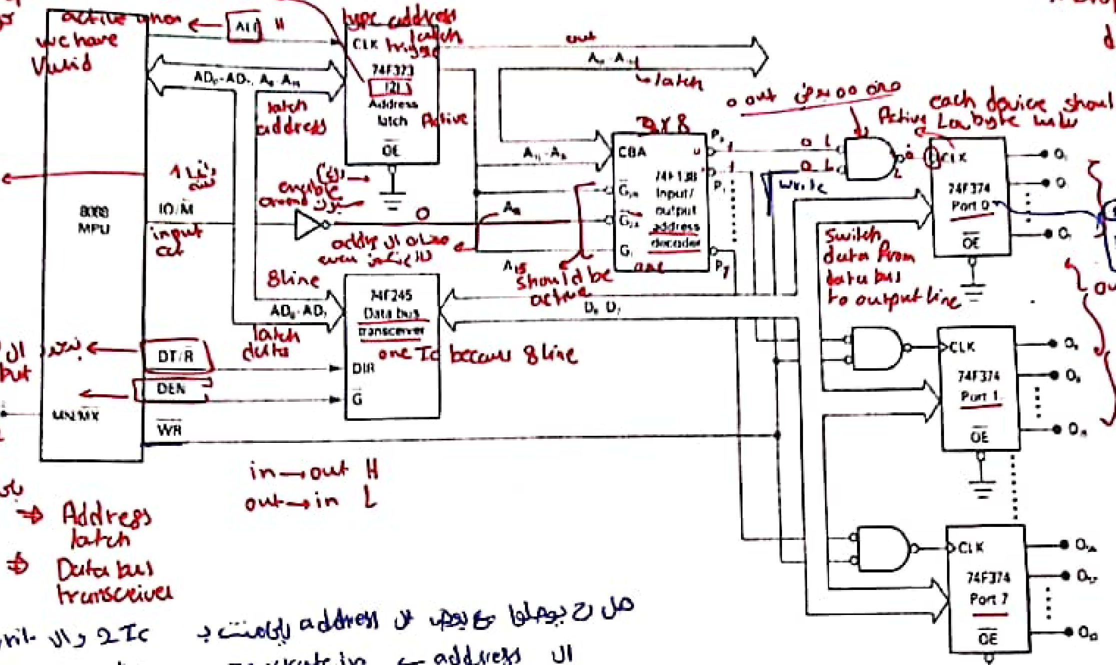
Task:

① latch output data (keep data in buffer)
② sample output data
③ speed sync
④ voltage tran.
⑤ select one of the sevical I/O (base on the address)

- In every computer, when data is sent out by the CPU, the data on the data bus must be latched by the receiving device
- While memories have an internal latch to grab the data on the data bus, a latching system must be designed for ports
- Since the data provided by the CPU to the port is on the system data bus for a limited amount of time (50 - 1000ns) it must be latched before it is lost
- Likewise, when data is coming in by way of a data bus (either from port or memory) it must come in through a three-state buffer

Example for parallel input output interface (cove input/out)

Example - 64 line parallel output circuit - 8088



out 64K
استخدمنا 2 كانه عندنا 10 line/line

latch data address لل 8

active when we have valid

ALE H

type address latch trigger

latch address

Active

out A8-A15 latch

enable

1 يعني out

IO/M input out

معناها ال address even

8line

latch data

one Ic became 8 line

isolated I/O mapped اذا بيكون zero D0 active

direction ال بيحدد of data but valid

data enable

in → out H
out → in L

latch address بيعمل ال → Address latch

latch data " → Data bus transceiver

ال 2 Ic وال write؟ generate write in T2

ال address ← generate in T1 ، بالتالي address ال عن بيعبر ح بيوصلها هل

3×8

should be active

A15 should be active

0 out لو 00 بيعطي

0 L
0 L
write

each device should Active Low byte

Switch data from data bus to output line

8 out line
① should selected from address decoder
② microprocess output port
- should creat write

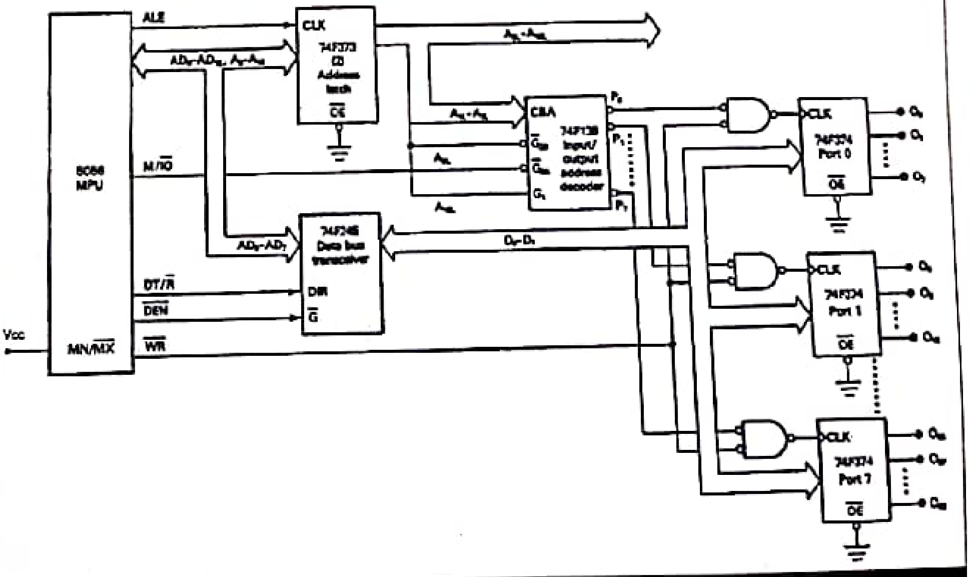
all outline 64 line

microprosser with out-put device I cant connect 8 out put device

just one port should be select or active

I/O decoding and the 8086 64-line parallel port

I/O port	I/O address
Port 0	$1XXXXXXXXXXX0000_2$
Port 1	$1XXXXXXXXXXX0010_2$
Port 2	$1XXXXXXXXXXX0100_2$
Port 3	$1XXXXXXXXXXX0110_2$
Port 4	$1XXXXXXXXXXX1000_2$
Port 5	$1XXXXXXXXXXX1010_2$
Port 6	$1XXXXXXXXXXX1100_2$
Port 7	$1XXXXXXXXXXX1110_2$



Q15? what the address should select by microprocess in order to select port-1?

Examples

- To which output port in the previous figure are data written when the address put on the bus during an output bus cycle is 8002h?
  - A15 .. A0 = 1000 0000 0000 0010b
  - A15L = 1
  - A0L = 0
  - A3L A2L A1L = 001
  - $\overline{P1}$ = 0

must for active
اذا ماكان شغالين ما بحكي
non of them selected

Port 1 →

- Write a sequence of instructions that output the byte contents of the memory address DATA to output port 0 in the previous figure

```
MOV DX, 8000h
MOV AL,DATA
OUT DX, AL
```

Data byte

لازم احدد
1 address ال for port zero

2 isolated or memory mapped ?
اذا كانت ماخضه مقالة Isolated
memory mapped " " memory "

3 بدي اشوف اذا ال address of zero غير page0 ولا لا

turn on and off periodicly

Time Delay Loop and Blinking a LED at an Output

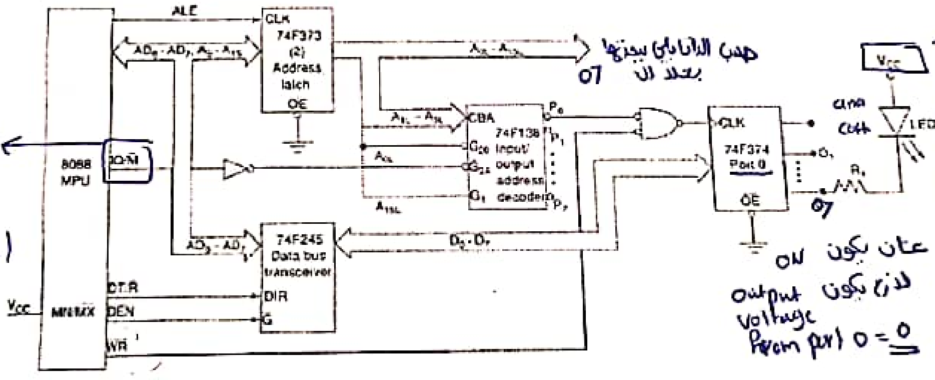


Figure 10-2 Driving an LED connected to an output port

جيب الداتا اللي بدي نعرضها
07 بعد ال
1
ano cath
light emiting
forward current
pass current → turn on current
عشان يكون ON
لازم يكون output voltage from port 0 = 0
معطل 10 الموجود ال
معناته INV
Isolated (special instruction)
indirect

address port zero = 8000
lead
لما العامل ال out ال يشتغل
بدي اطفي بدي اضوي
فيه Q7 = 1

```
MOV DX, 8000h      ;initialize address of port 0
MOV AL, 00h        ; load data with bit 7 as logic 0
ON_OFF: OUT DX,AL  ; turned on
MOV CX,0FFFFh      ; load delay count of FFFFh
HERE: LOOP HERE
XOR AL,80h         ; complement bit 7
JMP ON_OFF
```

Indirect
AL 00
شوط
toggle بتعمل
decrement cx by one and jump to label
loop ترجع تعمل
00 XOR 80 = 80
delay ON = delay off
اذا غير بدنا
ON delay
off delay

Aprox.
17 T states / 5K / Frequency
17 clock period / $\frac{1}{\text{Freq}}$

بحطلها د loop
عشان تكون on off
بدوم عما
ضمان الحالة مارح
تشوفه بطفي لانه
سرعة كتيركتير
كبيرة بالنسبة
لملاحظة العين

delay الحل العمل
بين on/off

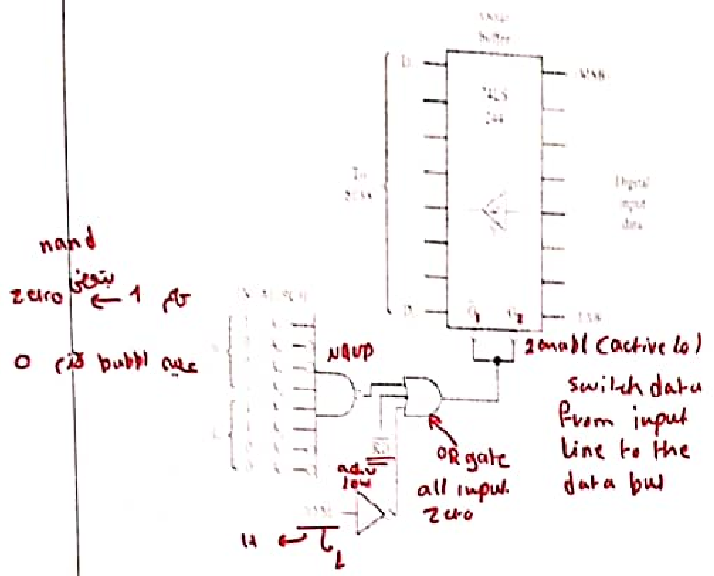
7 6 5 4 3 2 1 0
1 0 0 0 0 0 0 0
80
mov AL, 80
out DX, AL
off jump

1 loop → 17
X → 1m

10.3 IN port design using the 74LS244

Parallel input

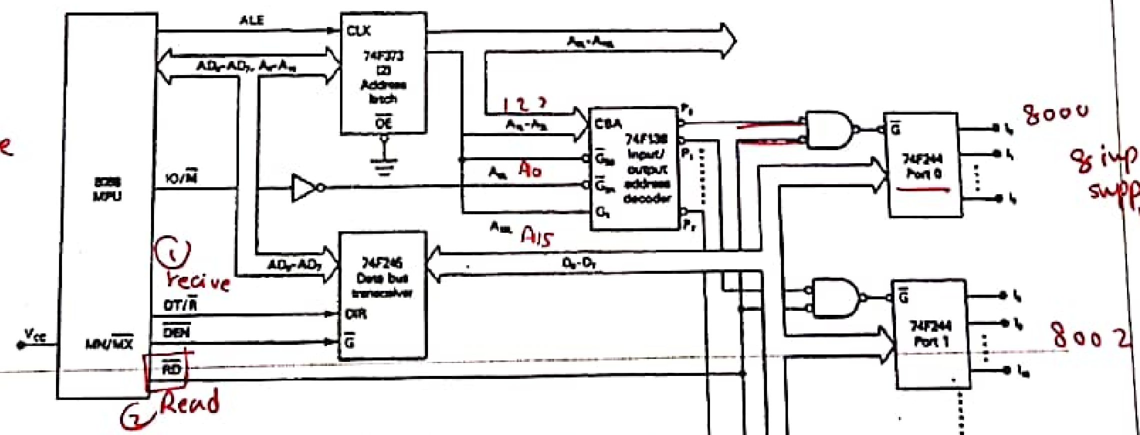
- Design for IN AL,9CH



- In order to prevent any unwanted data (garbage) to come into the system (global) data bus, all input devices must be isolated through the tri-state buffer. The 74LS244 not only plays this role but also provides the incoming signals sufficient strength (driving capability) to travel all the way to the CPU
- It must be emphasized that every device (memory, peripheral) connected to the global data bus must have a latch or a tri-state buffer. In some devices such as memory, they are internal but must be present.

Example - 64 line parallel input circuit

difference between 64 input 64 output 1,2



Read from input port 7 to the memory location DATA: address
MOV DX, 800Eh
IN AL,DX
MOV DATA, AL

800E port 7

حطالي المخزن بالمحتوى location AL من بال mov حطلي بعل

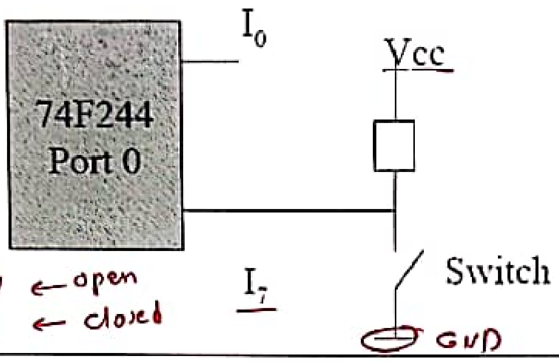
Mov [ ], AL

① Isolated / Memory mapped
② address port [with page 0 / not " " 0 → indirect
③ read or write
IN out

Example

- In practical applications, it is sometimes necessary within an I/O service routine to repeatedly read the value at an input line and test this value for a specific logic level.

I want to read status of switch



Vcc ⇐ I7 ← open

0 ⇐ I7 ← closed

poll the device [(dont have input) / press (we have input)

Shiking the status of switch

① address of port ② isolated ③ direct, indirect ④ read (IN)

```
Poll the switch waiting for it to close
        MOV DX,8000h
POLL:   IN AL,DX
        SHL AL,1  → رح يكون ال I7 عال carry
        JC POLL
```

Jump on Carry

احنا لازم نفحص ال I7 بس عشان هيك بنعمل shift وبعدها بنطلع على ال carry flag

او بنعمل MASK

SHL, SHR حسب ال bit كم وين اقرب

I7 اقرب على الشمال

AND AL, 80 (mask)

JNZ pool

وين ما كان I7 والباقي zero

بعمل check على ال zero

اذا بدي I6 ← SHL 2

19

11

10.4 Input Output Handshaking

PPI

لازم اشوف ال status

مثلاً ال printer

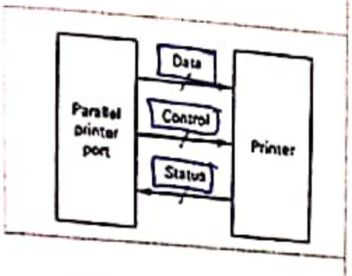
Handshaking Exchange Control signal

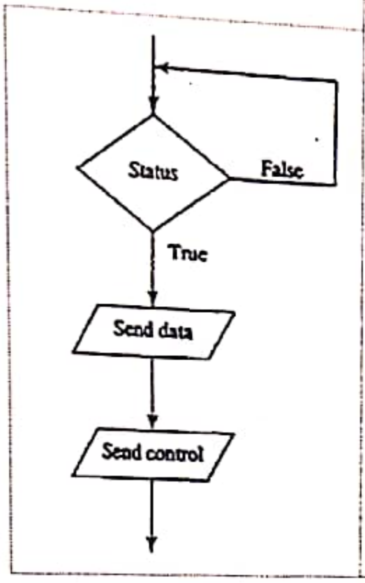
- ➢ The I/O ports of a computer typically operate at different data rates
- ➢ A hard disk drive, for example, might require the computer to input data at 10Mbps → 100Mbps
- ➢ CD-ROM drives operate at 300-600 Kbps
- ➢ However when inputting keystrokes from the operator, the data rate may fall to only one or two characters per sec.
- ➢ If the processor is to operate efficiently, one needs to develop a strategy to control or synchronize the flow of data between the processor and the widely varying rates of its I/O devices
- ➢ This type of synchronization is achieved by implementing what is known as handshaking as part of the input/output interface
- ➢ Printers typically have buffers that can be filled by the computer at high speed
- ➢ Once full the computer must wait while the data in the buffer is printed
- ➢ Most printer manufacturers have settled on a standard set of data and control signals Centronics Parallel Printer Interface

Parallel Printer Interface

بال

printer.





Pin	Assignment
1	Strobe
2	Data 0
3	Data 1
4	Data 2
5	Data 3
6	Data 4
7	Data 5
8	Data 6
9	Data 7
10	Ack
11	Busy
12	Paper Empty
13	Select
14	Auto Foxt
15	Error
16	Initialize
17	Slctin
18	Ground
19	Ground
20	Ground
21	Ground
22	Ground
23	Ground
24	Ground
25	Ground

25 pins

Data: Data0, Data1, ..., Data7

Control: 4 type of control
- Strobe → الأهم
- Auto Foxt
- Initialize
- Slctin

Status:
- Ack
- Busy → : الأهم
- Paper Empty
- Select
- Error

ACK is used by printer to acknowledge receipt of data and can accept a new character.

BUSY high if printer is not ready to accept a new character 0 → ready 1 → busy أهم

SELECT when printer is turned on

ERROR goes low when there are conditions such as paper jam, out of paper, offline

STROBE when PC presents a character

INITIALIZE Clear Printer Buffer and reset control

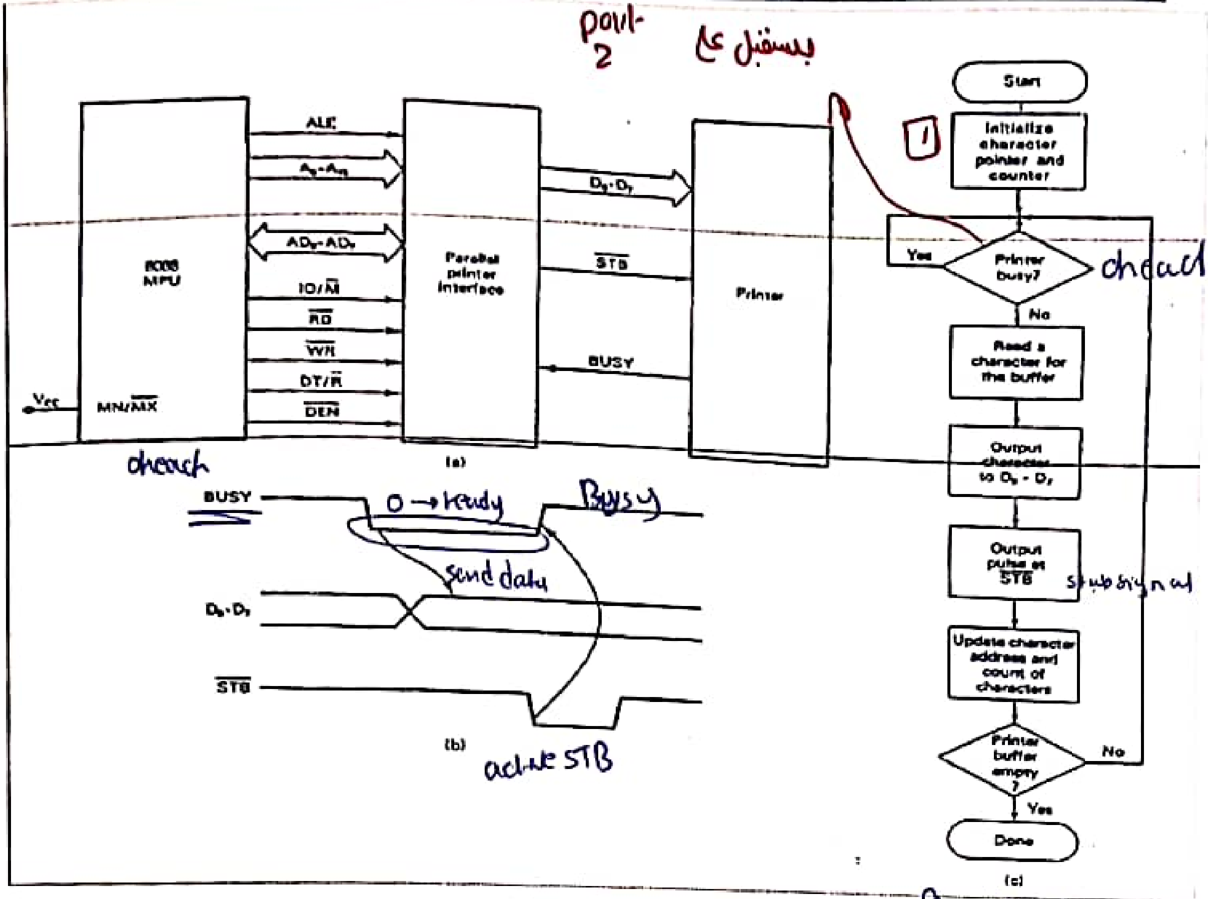
Operational Principle - Parallel Printer Port

- The computer checks the BUSY signal from the printer, if not BUSY then
- When the PC presents a character to the data pins of the printer, it activates the STROBE pin, telling it that there is a byte sitting at the data pins. Prior to asserting STROBE pin, the data must be at at the printer's data pins for at least 0.5 microsec. (data setup time) short tuny of time

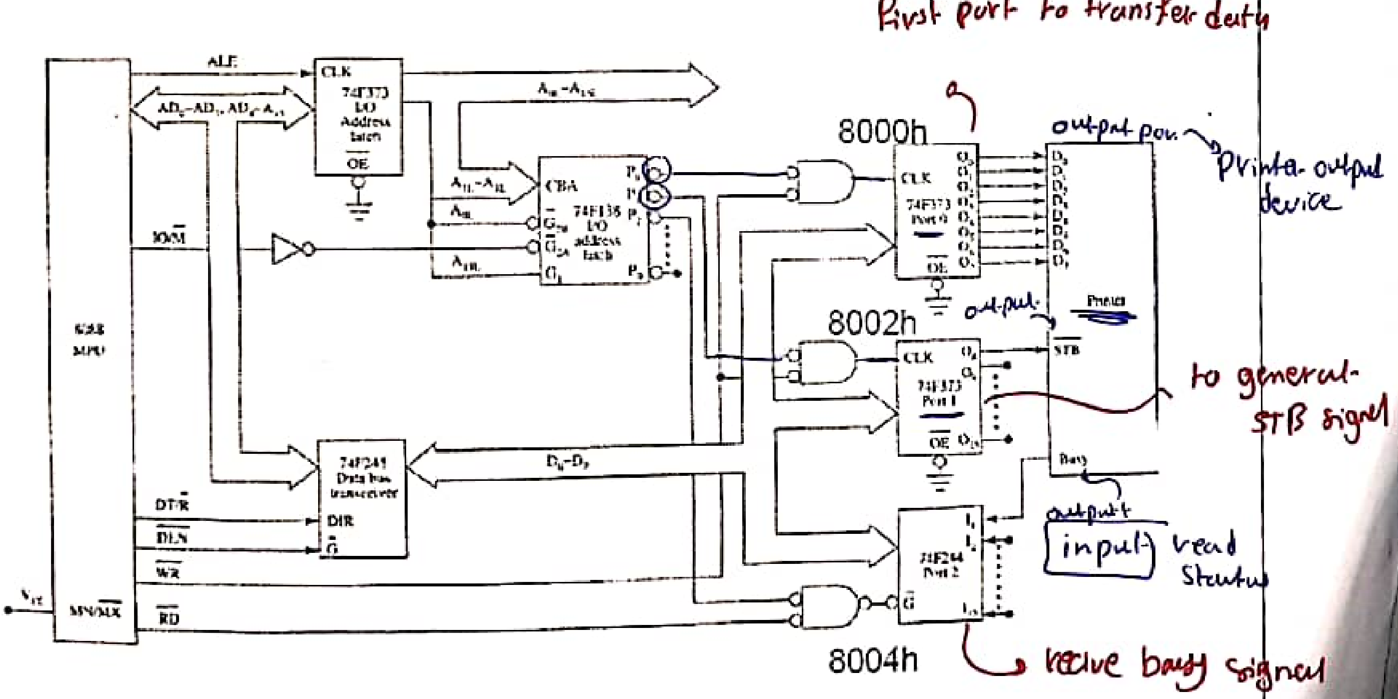
Control signal

- The STROBE must stay for 0.5 microsec
- The printer asserts BUSY pin indicating the computer to wait
- When the printer picks up the data, it sends back the ACK signal, keeps ACK low for 5 microsec.
- As the ACK signal is going high, the printer makes the BUSY pin low to indicate that it is ready to accept the next byte
- The CPU can use ACK or BUSY signals from the printer to initiate the process of sending another byte

Handshaking



Printer Interface Circuit



Instizution
① Counter → [illegible] char [illegible] Print
② Pointer → offset address where store in memory

Copy [illegible]

Example

- Write a program that implements the flowchart. Character data is held in memory starting at address PRNT_BUFF, the number of characters held in the buffer is identified by the count address CHAR_COUNT.

```
                       cr
                 MOV CL, CHAR_COUNT      # of charachter need to print [illegible] offset [illegible] 100 char
AX [illegible] ← 1 ←     MOV SI OFFSET PRNT_BUFF    instize pointer (offset address of first character)
DX
POLL_BUSY.       MOV DX,8004h   ] Isolated, Indirect
                 IN AL,DX
shif Right  mosk → AND AL,01h                      BUSY input checked
C AND      JNZero → JNZ POLL_BUSY
JC               MOV AL, [SI]   memory location
data           [ MOV DX,8000h   address port 0     Character is output
availel        [ → OUT DX,AL
                                        [illegible]
                 MOV AL, 00h       ← ;STB = 0
               [ MOV DX,8002h
               [ OUT DX,AL                           So as the strobe
                 MOV BX,0Fh              ; delay for STB = 0      200nx17 [illegible] → 0.5
                 STROBE: DEC BX
                 JNZ STROBE
                 MOV AL,01h                [illegible]
                 OUT DX,AL               ; STB bar = 1

udpat          [ INC SI
               [ DEC CL
               [ JNZ POLL_BUSY  → check for number of counter.
```

10.5 The 8255 Programmable Peripheral Interface PPI

Ic → This Ic can programm like in order to change functionality inp ⇄ out

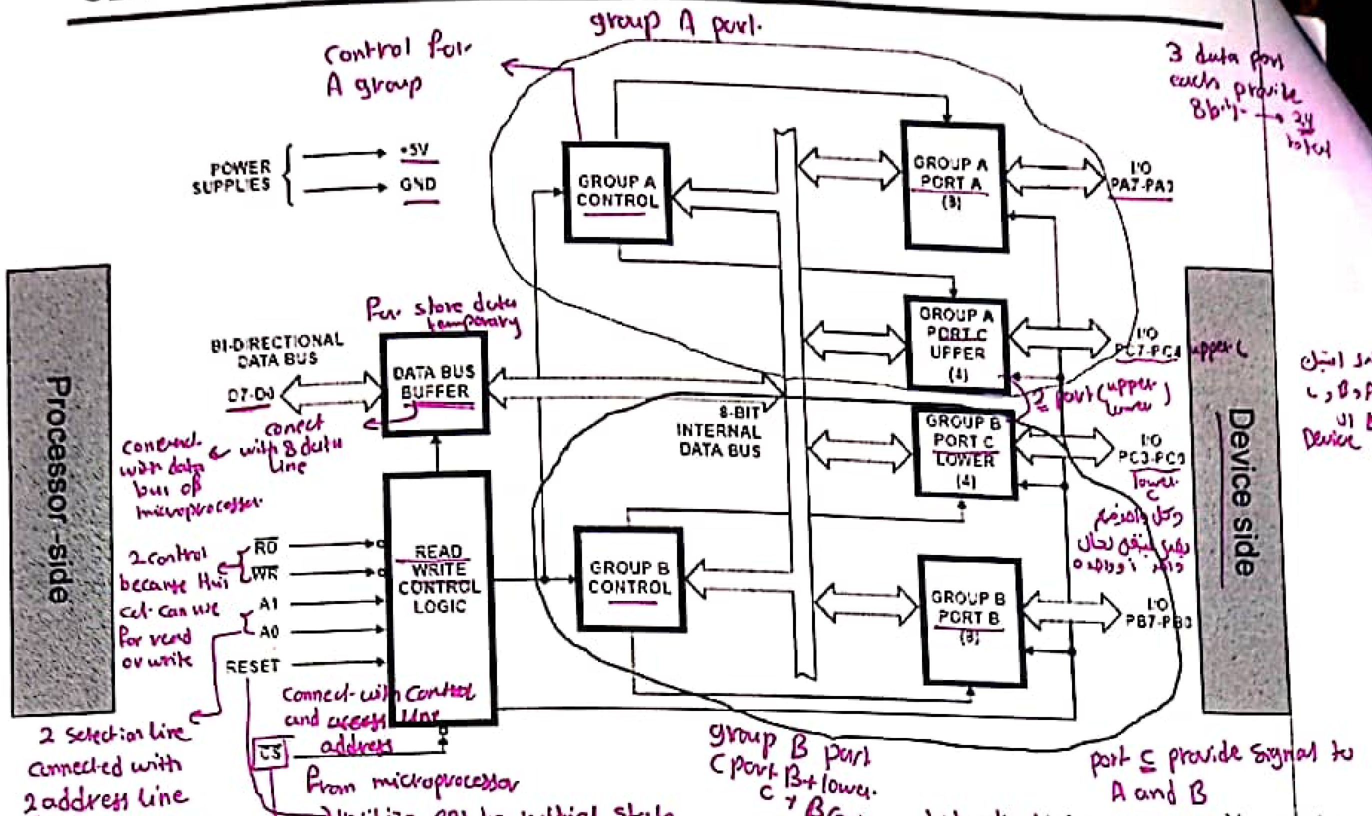
- Intel has developed several peripheral controller chips designed to support the 80x86 processor family. The intent is to provide a complete I/O interface in one chip.
- 8255 PPI provides three 8 bit input ports in one 40 pin package making it more economical than 74LS373 and 74LS244. (shaper; 1 Ic support 3 port each port has 8 data line)
- The chip interfaces directly to the data bus of the processor, allowing its functions to be programmed; that is in one application a port may appear as an output, but in another, by reprogramming it as an input. This is in contrast with the 74LS373 and 74LS244 which are hardwired and fixed
- Other peripheral controller chips include the 8259 Programmable Interrupt Controller (PIC), the 8253/54 Programmable Interval Timer (PIT) and the 8237 DMA controller

① Simple I-O
② strobe I-O (hand sheak)
③ bidirectional I-O

I can also select number of bit of data that I can transfier (byte-word..)

total 24 (can be input or output or bidertional)

8255A internal



2 selection line connected with 2 address line of the microprocessor.

From microprocessor

→ Insilize PPI to Initial State (output data بيرسل) in simple transfer mode

port C provide signal to A and B

في حال استخدمت الطرق الاخرى bidirectional hand check (strop I/O)

upper C provide control signal to A

lower C provide " " to B

A1 A0 مو بالضرورة يكونوا نفس to select وظيفة one of the data column

[we have 3 data port called A,B,C)

00 → port A

01 → port B

10 → port C

11 → Control Reg.

يعني 0 ← activate the IC (enable) generated from combinational line from address bus

8255 Pins

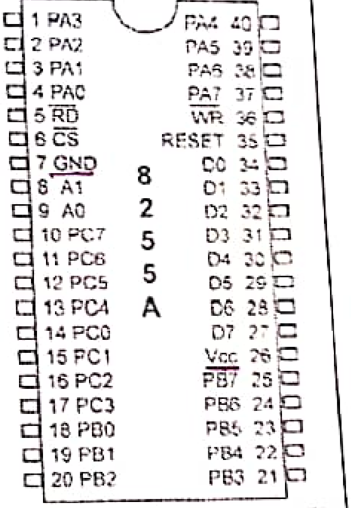
هذا تقسيم logically ولكن physically هو مو مقسمة

- PA0 - PA7: input, output, or bi-directional port
- PB0 - PB7: input or output
- PC0 - PC7: This 8 bit port can be all input or output. It can also be split into two parts, CU (PC4 - PC7) and CL (PC0 - PC3). Each can be used for input and output.
- $\overline{RD}$ or $\overline{WR}$
  - $\overline{IOR}$ and $\overline{IOW}$ of the system are connected
- RESET
- A0, A1, and $\overline{CS}$
  - $\overline{CS}$ selects the entire chip whereas A0 and A1 select the specific port (A, B, or C)

* address bus we use A1 A0 CE

RD, WR → من ال microp.

Reset → من ال clk generator



ure 11-11 8255 PPI Chip

Value ال cap for A,B I select the port

CSBAR	A1	A0	SELECTS:
0	0	0	Port A
0	0	1	Port B
0	1	0	Port C
0	1	1	Control Register
1	x	x	8255 not selected

active

not active

24 + 8 (data bus) = 32

A0 + A1 – selection line 2

R+W = 3

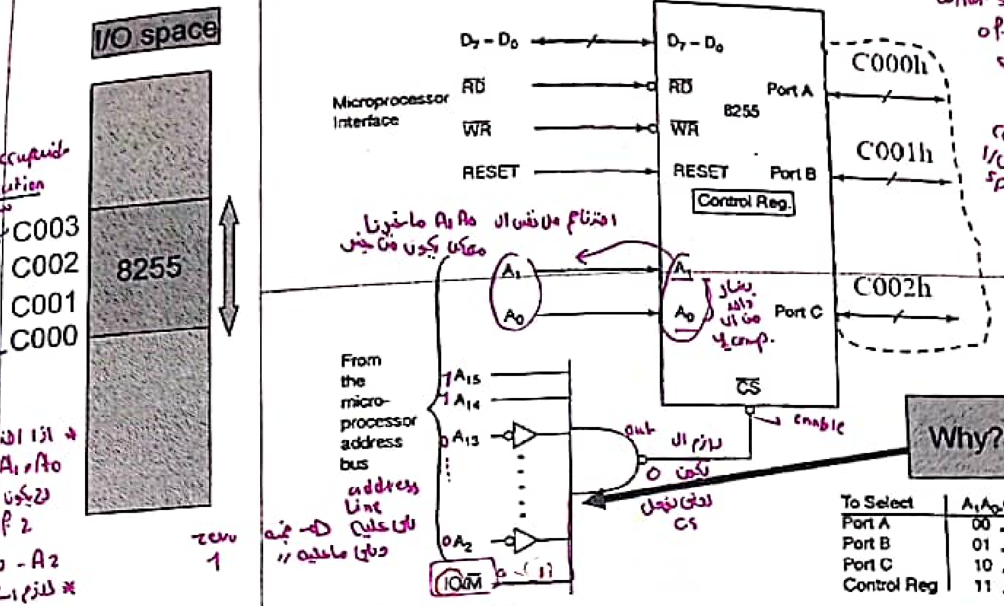
Vcc + GND = 2

Reset + CS = 2

totaly 40

Addressing an 8255

Explain interface the microprocessor with PPI



1 ppi occupied four location
memory isolated / dedicated
one location per port + location for control Reg.

* اذا الان مثلاً ال A_1, A_0 من A_2 و A_9 ديكون ال address multiple of 2
mult 4 $A_3 - A_2$
* لازم اشوف ال connect

اختام ال نفس ال $A_1 A_0$ ماخبرنا معكن يكون من جنس

بتاعت ال ports للcomp.

لازم ال out 0 عشان تفعل CS

لي علي ده جنبه دبلي ماشيه

كيف عرفت انها isolated من IO/M

داخلة ال NAND وناتجها لازم يكون zero عشان افعل ال CS

لهيك لازم تكون 1 (for IO ← active high)

داخل مباشرة بدون inverter

what should be the address of port A depend of connection shown in the figure?

كما ان ال address عن address فبكون ال address connected I/O space من A_{15} ل A_0 (isolated)

or (A_{19} - A_0) [memory mapped]

بالي مستخدم بحطه ضمن
والي مو مستخدم بحطه X

A_{15} A_{14} A_{13} — A_2 A_1 A_0
1 1 0 0 0 0 0 0 0

$\boxed{C000}$

Port B?

بنغير ال selection line

A_{15} A_{14} A_{13} — A_2 A_1 A_0
1 1 0 — 0 0 1

$\boxed{C001}$

Port C?

$\boxed{C002}$

Control reg.

$\boxed{C003}$

To Select	$A_1A_0\overline{CS}$
Port A	00 .0
Port B	01 .0
Port C	10 .0
Control Reg	11 .0

8255 Control Word Format

Value of Control Reg. but it's one byte (8bit)

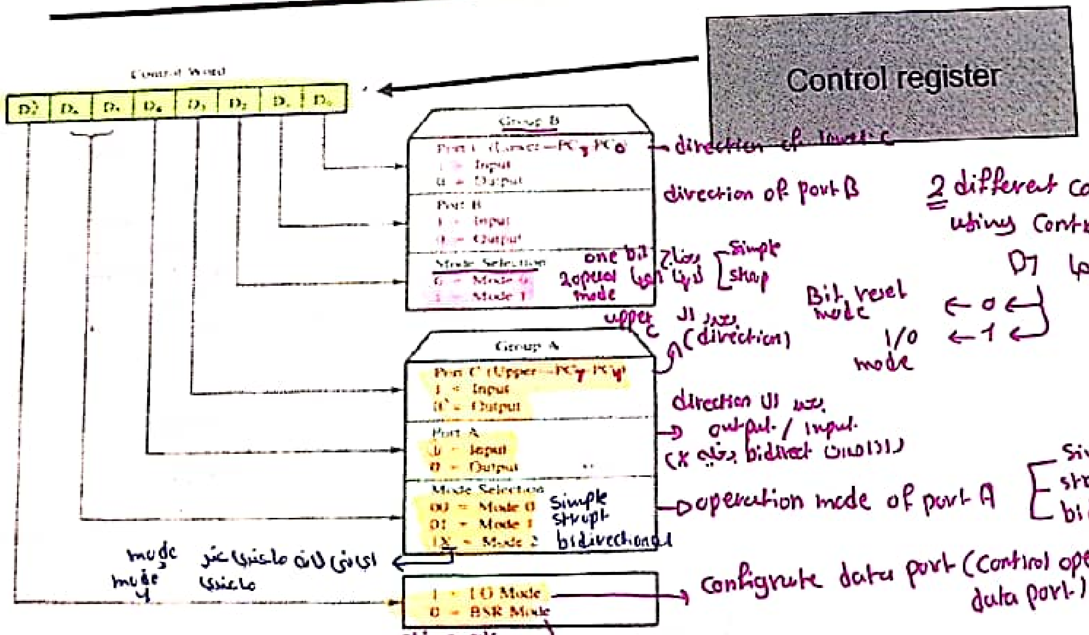


FIGURE 15.4
8255A Control Word Format for I/O Mode
SOURCE: Adapted from Intel Corporation, Peripheral Components (Santa Clara, Calif.: Author, 1983), p. 3

2 different configuration using Control world

D7 بيحدد

Bit reset mode ← 0
I/O mode ← 1

configrate data port (Control operation of data port.)

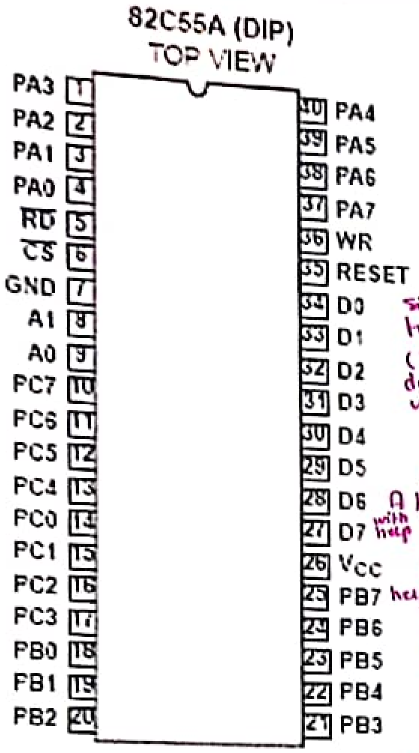
Simple (transfer data without use any control signal)
strobt
bidirectional → 2 direc.

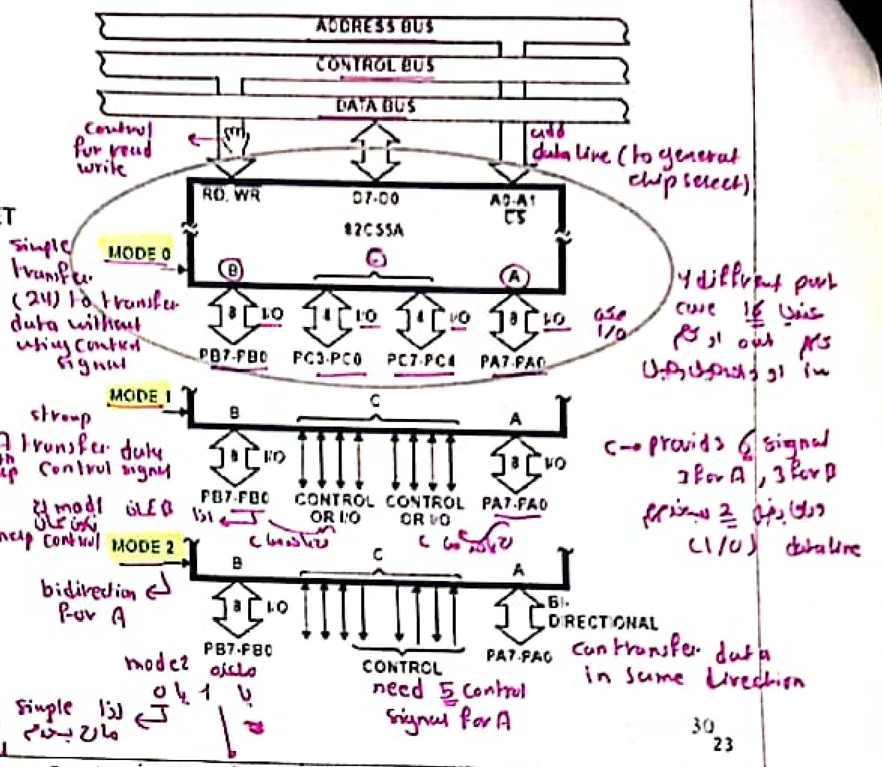
transfer data with help of control signal (like hand shek)

to send Control Reg. to Control word

(spical inst an isolated) PPI حسب النوع

(memory inst an memory mapped)




ADDRESS BUS
CONTROL BUS
DATA BUS
Control for read write
add data line (to general chip select)
RD, WR
D7-D0
A0-A1
CS
82C55A
MODE 0
B
C
A
8 I/O
4 I/O
4 I/O
8 I/O
PB7-PB0
PC3-PC0
PC7-PC4
PA7-PA0
Simple transfer (24) I/O transfer data without using control signal
عمله I/O
4 different port cure يعني إن كل أو out أو in البرمجة لكل
MODE 1
B
C
A
8 I/O
8 I/O
PB7-PB0
CONTROL OR I/O
CONTROL OR I/O
PA7-PA0
strop
A transfer data with help control signal
C→ provids 6 signal 3 for A, 3 for B
ودا يعني 3 بستخدم (I/O) data line
mode 1 بتقدر تخلي C الباقي للـ
C تستخدم
MODE 2
B
C
A
8 I/O
PB7-PB0
CONTROL
PA7-PA0
BI-DIRECTIONAL
help control
bidirection for A
Can transfer data in same direction
need 5 control signal for A
mode 2 ملهاش
0 و 1 بـ
إذا Simple ماله بستخدم control signal
I/O بيعني 3 control بيكون استخدام
3 بستخدم control signal from port (C)
محتاجين لـ control signal data

Mode 0 - Simple input/output

- Simple I/O mode: any of the ports A, B, CL, and CU can be programmed as input or output.

transfer data without using control signal

Programmit using control word

- Example: Configure port A as input, B as output, and all the bits of port C as output assuming a base address of 50h → 4 one PPI ال occupied 4 memory location

- Control word should be 1001/0000b = 90h

D7 inp out-

mode 0 for port A

output لـ C كلها

base address

لكل واحد عنوان

D7 D6 D5 D0

1 0 0

mode zero for port A

بحدد لكل وحدة A, B...

```
PORTA   EQU 50h
PORTB   EQU 51h
PORTC   EQU 52h
CNTREG EQU 53h
MOV AL, 90h
OUT CNTREG,AL
IN AL, PORTA
OUT PORTB, AL
OUT PORTC, AL
```

lable

input بنقرأ من A

out (B نعطي)

out

بيجي الـ first من الـ 4 address

Base Adress = address بتاع port A

من IO/M بحدد بين memory mapped / Isolated

من الـ selection line بحدد العنوان بيجي

مثلاً A0, A1 x2

A3, A2 x4

بحدد الـ value الـ selection line

Control word Value → Control registes

Configuration عملية

First I need to find value of control word

D7 → 1 (I/O)

D2 → mode selection of port B (simple)

D1 → direction of B (out)

D0 → " of lower C (out)

control word value (90)

I want to copy this to control Rey

↳ the value is 53

① Isolated → spical

② direct or indirect? address 53 within page 0 (direct)

Sequentaly لما افترض انو

وفرض إنه isolated

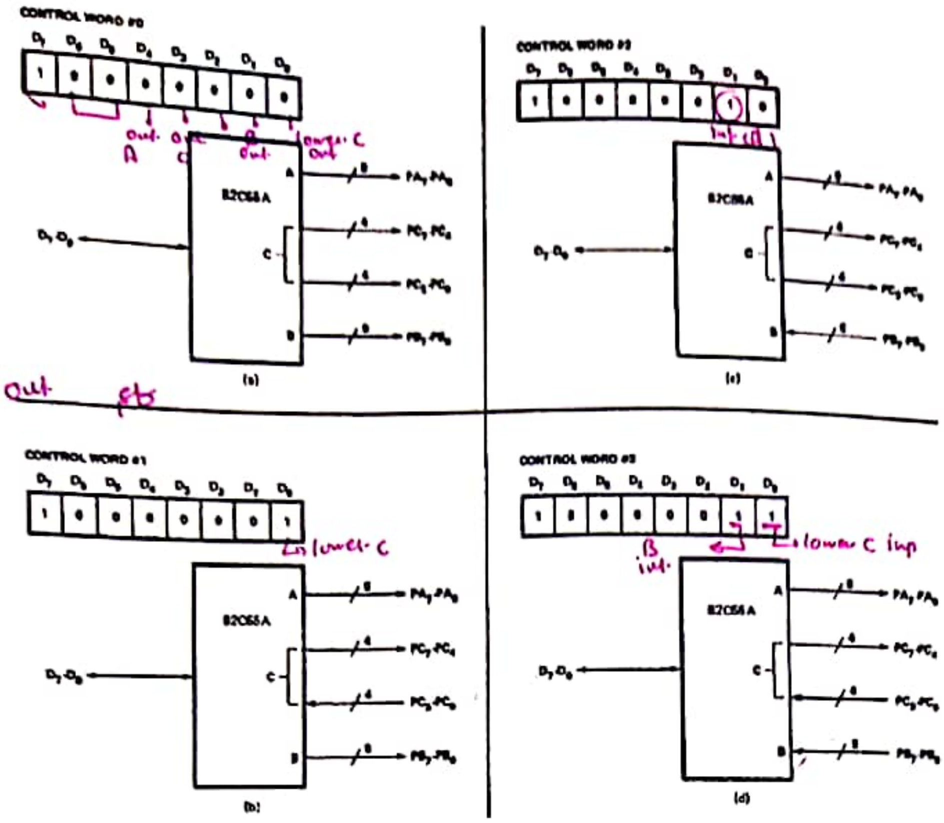
90 → AL

out

PORTA العنوان
lapel عبارة عن

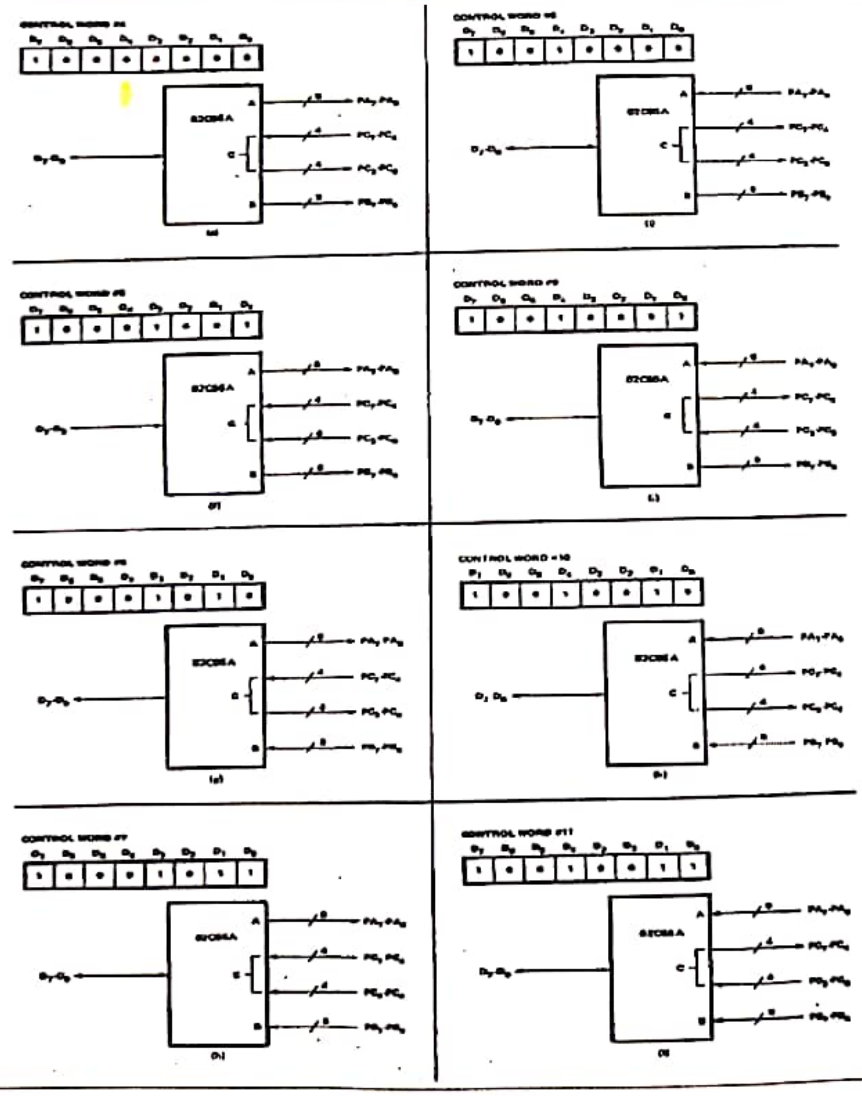
EQU
sign directive

in mode zero I have up to 16 cases
4 individual port. 4 } 0,1 → 16

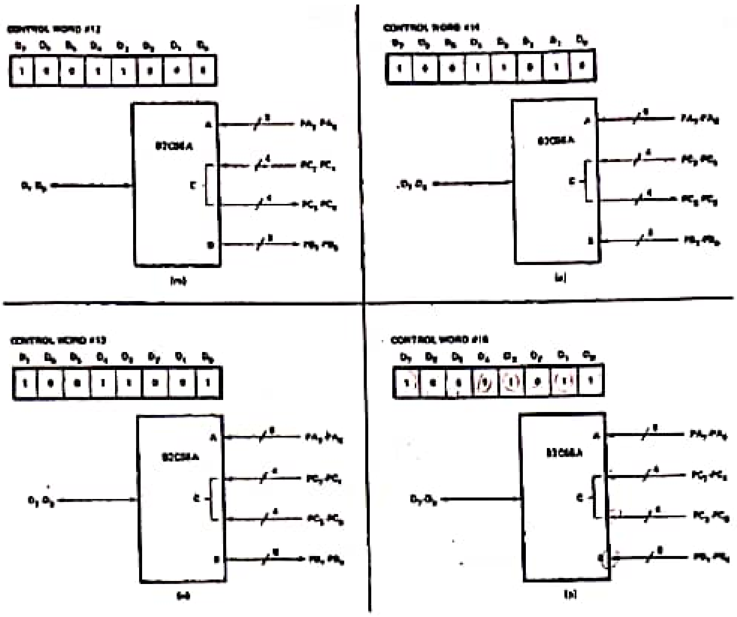


Mode 0 control words (I/O)

(16 cases)



Mode 0 control words (I/O)



Mode 0 control words (I/O)

Mode 1: I/O with Handshaking Capability

Stroped

transfer data with helping of control signal

port A hand cheak
port B hand cheak
port C provide control signal for port A and B (8 line)
3 for A, 3 fo B
2 for I/O

need 3 control signal

- Handshaking refers to the process of communicating back and forth between two intelligent devices
- Example: Process of communicating with a printer
  - a byte of data is presented to the data bus of the printer
  - the printer is informed of the presence of a byte of data to be printed by activating its strobe signal
  - whenever the printer receives the data it informs the sender by activating an output signal called ACK
  - the ACK signal initiates the process of providing another byte of data to the printer
- 8255 in mode 1 is equipped with resources to handle handshaking signals

Setup of Mode 1

	MODE 1	
Pin	IN	OUT
PA₀	IN	OUT
PA₁	IN	OUT
PA₂	IN	OUT
PA₃	IN	OUT
PA₄	IN	OUT
PA₅	IN	OUT
PA₆	IN	OUT
PA₇	IN	OUT
PB₀	IN	OUT
PB₁	IN	OUT
PB₂	IN	OUT
PB₃	IN	OUT
PB₄	IN	OUT
PB₅	IN	OUT
PB₆	IN	OUT
PB₇	IN	OUT
PC₀	$INTR_B$	$INTR_B$
PC₁	IBF_B	$\overline{OBF_B}$
PC₂	$\overline{STB_B}$	$\overline{ACK_B}$
PC₃	$INTR_A$	$INTR_A$
PC₄	$\overline{STB_A}$	I/O
PC₅	IBF_A	I/O
PC₆	I/O	$\overline{ACK_A}$
PC₇	I/O	$\overline{OBF_A}$

port A (strup) → input يا out يا

inp or out →

For B

For A

roming data line

out Buffer full

* port A → mode1 → IN
رح ياخد ال Control signal from C → 3,4,5

* port A → mode1 → out
رح ياخذ bit 3,6,7

* port B → mode1 → IN
رح ياخذ 0,1,2

* كان out → 0,1,2 { نفس الاشي بس مختلفين

reminy bit from C (IN direction) 6,7

reminy bit from C (out ") 4,5

* ال note ال control signal ال port بتحدد حسب اذا كان ال IN/out

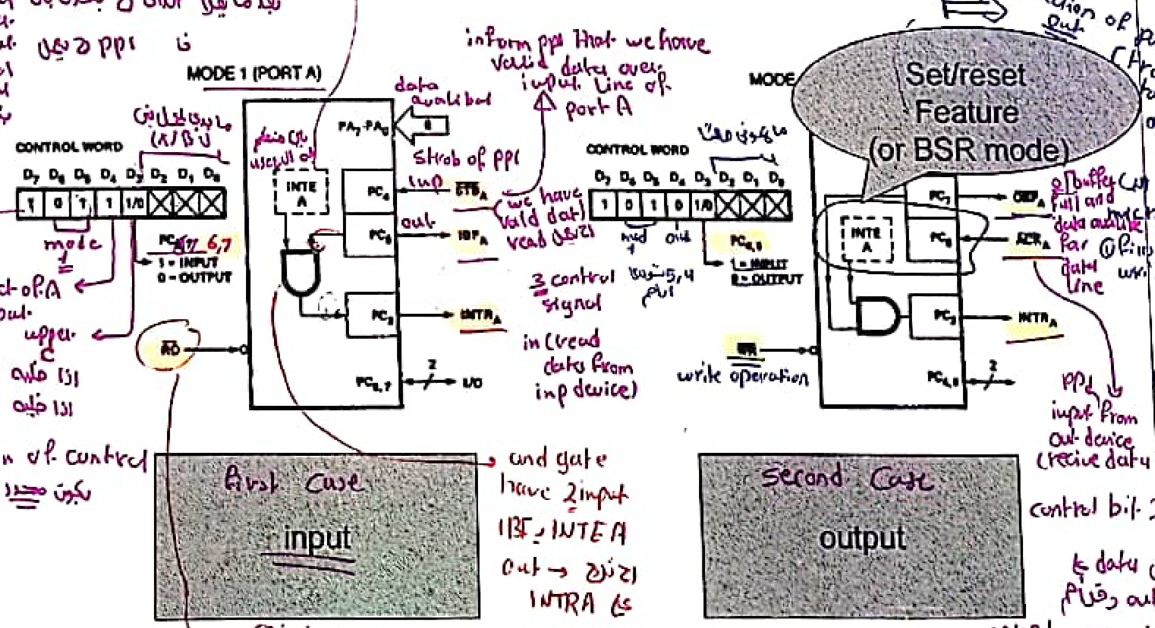
Mode 1 (configuration of port A)

intrepul enable : enable بفعل intrr req. هل مسموح لل PPI تعمل Full
Inter للمايكروبروسسور اولا

بعد ما يخزن الداتا رح بتخزن بال internal buffer ويصير Full

فا PPI رح تعمل generate control signal اسمها IBF

يعني Acknolg لل inp device انه read data successfully



ما بدي احددهن (A/B)

1/0 د ايه

mode 1

direct of A input

upper C

Bit 6,7 out ← 0 اذا خليه
Bit 6,7 in ← 1 اذا خليه

direction of control يكون محدد حسب وظيفته

بتحكم بال interrupt من البرنامج

data avalebal

inform ppi that we have valid data over input line of port A

strob of ppi

inp

out

(we have valid data) read اذا رح

3 control signal فيهم 5,4 ايضاً

in (read data from inp device)

ما يهمني فهن

mod 1

out

write operation

direction of port A out (transfer data to the out device)

First Case

and gate have 2 input IBF و INTE A
اذا نزلو → out
ع INTRA

وظيفة IBF → ① acknolge recive data succesfully
② generate INTR signal (request interrupt to microprocessor) يعني بحكي لل microp اطلع كاي

لما ال microproc. نسأل INTR رح يفهم انه في data موجودة بال PPI
لازم تنقل ال Buffer رح يصير empty
redy to recive (0) ← IBF next data
INTR ← (0) طفي ال flag
(to transfer data from PPI to microp.)
ورح يعمل initute read operation

Second Case

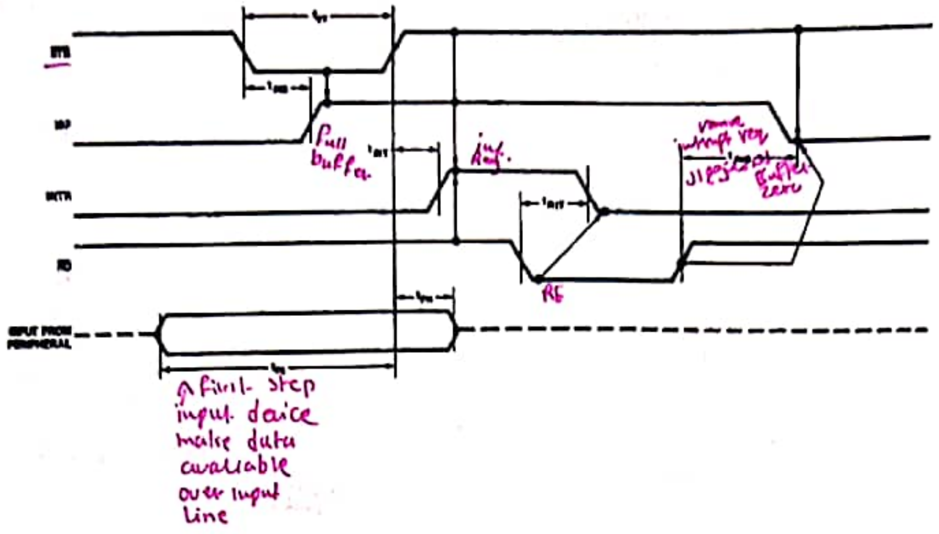
buffer full and data avalable for data line
يا ال data تم ارسالها للاجهزة output
① first step initial write operation
ال data transfer for microp to the inter buffer of PPI

PPI ← input from out device (recive data succsfuly)

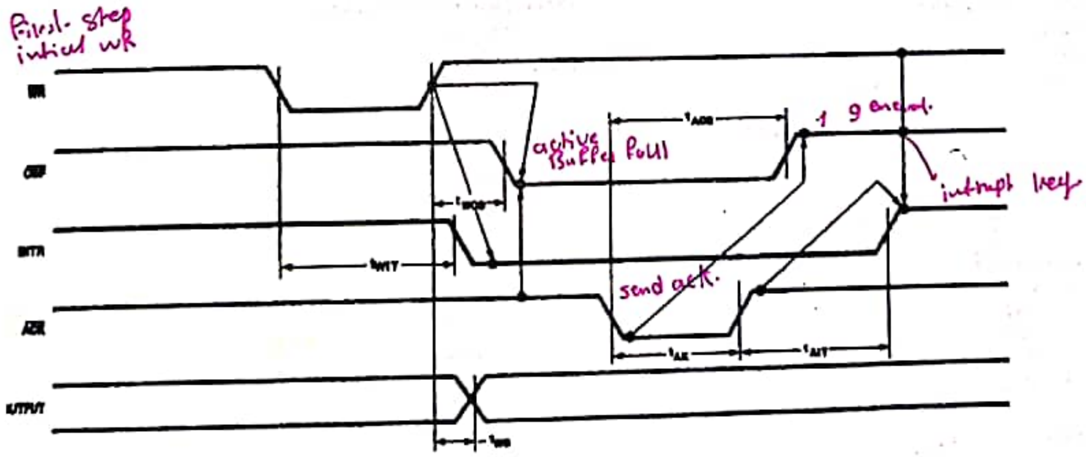
control bit 3,6,7

← عرفنا في data ع ال output رقام
وصلت لل PPI ← (ACK)
يعني انها read succfuly data
وال buffer صار empty
OBF (1) رح يعمل
generate INTR
بعدها نقل ال initial other write operation (new data)

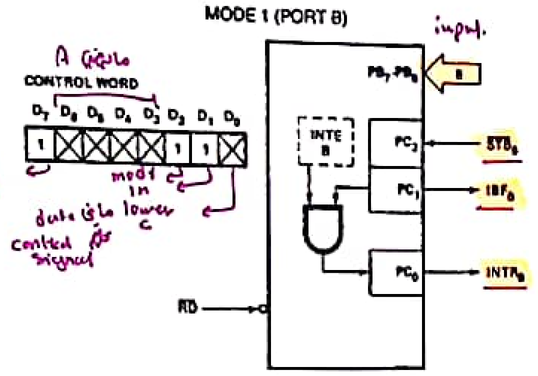
Timing diagram of port A (input)



Timing diagram of port A (output)



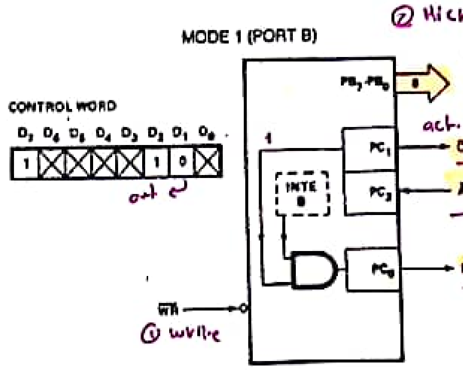
Mode 1 (configuration of port B)



config for data world

control for signal

control for signal 0 to 3



data trans
② Micrp to buffer and become full

(data avali')

لما يكون OBF = 1
لو INTE B = 1
generate INTR

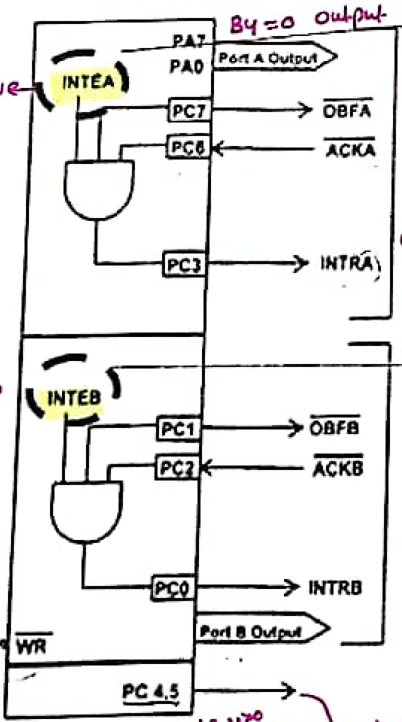
نقدر نعمل initiate to write operation

input
similar to port A

output
similar to port A

Mode 1 Strobed Output

لما نجعل port A و port B O/p



enable نجعل for INTRA

enable نجعل for INTR B

محدد outp دائما 0. بنحددها احنا.

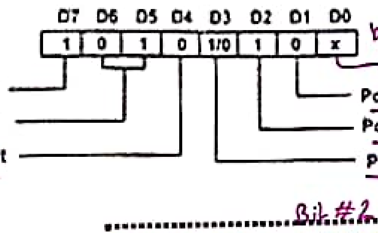
transfer data simple

Port A with Handshake Signals

Enabled through PC6 (Bit number 6) using the BSR mode

D7=0 لما disable و enable نجعل for port A or B

Control Word -- Mode 1 Output



باقي الـ 8 bit carry data

لوجود الـ control

(reaming of C)

sent from control word to control Reg.

Bit #2

Enabled through PC2 using the BSR mode

A, B (mode 1) and 2 is Out.

Port B with Handshake Signals

Status Word -- Mode 1 Output

كيف عرفت انه B in mode 1
لأنه الها 3 control signal
اذا مافي control signal معناته simple ونعمل الي A J

INTEA is controlled by PC6 in BSR mode.

INTEB is controlled by PC2 in BSR mode.

Mode 1 Strobed Output Signals

- $\overline{\text{OBFa}}$ (output buffer full for port A)
  - indicates that the CPU has written a byte of data into port A
  - must be connected to the STROBE of the receiving equipment
  - Goes back high again after ACK'ed by the peripheral.
- $\overline{\text{ACKa}}$ (acknowledge for port A)
  - through ACK, 8255 knows that data at port A has been picked up by the receiving device
  - 8255 then makes $\overline{\text{OBFa}}$ high to indicate that the data is old now. $\overline{\text{OBFa}}$ will not go low until the CPU writes a new byte of data to port A.
- INTRa (interrupt request for port A)
  - it is the rising edge of ACK that activates INTRa by making it high. INTRa is used to get the attention of the microprocessor.
  - it is important that INTRa is high only if INTEa, OBFa, ACKa are all high
  - it is reset to zero when the CPU writes a byte to port A
- The 8255 enables the monitoring the status signals INTR, OBF, and INTE for both ports A and B. This is done by **reading port C** into the accumulator and testing the bits. This feature allows the implementation of polling

Mode 1 Input Ports with Handshaking Signals

- $\overline{\text{STB}}$
  - When an external peripheral device provides a byte of data to an input port, it informs the 8255 through the STB pin. STB is of limited duration
- IBF (Input Buffer Full)
  - In response to STB, the 8255 latches into its internal register the data present at PA0-PA7 or PB0-PB7.
  - Through IBF it indicates that it has latched the data but it has not been read by the CPU yet
  - To get the attention of the CPU, IBF activates INTR
- INTR
  - Falling edge of RD makes INTR low
  - The RD signal from the CPU is of limited duration and when it goes high the 8255 in turn makes IBF inactive by setting it low
  - IBF in this way lets the peripheral know that the byte of data was latched by the 8255 and read into the CPU as well.
- The two flip flops INTEA and INTB are set/reset using the BSR mode. The INTEA is enabled or disabled through PC6 and INTEB is enabled or disabled through PC2.

Mode 2 Strobed Bidirectional I/O

- In this Mode Port A is A bidirectional I/O port in both direction
- Port C Provide the control functions for both directions
- To select this Mode D7 and D8 of the control register should be 1's (i.e. 11XXXXXX). select-mode 2

D_4 D_3 2 1 0

bidrect كل

control كلها

data line

for B (X)

if A configure in mode 2 he need 5 control signal

3 inp, 3 out

6 → interrupt الـ بس

common بس

Setup of Mode 2

Pin	MODE 2
	GROUP A ONLY
PA_0	←→
PA_1	←→
PA_2	←→
PA_3	←→
PA_4	←→
PA_5	←→
PA_6	←→
PA_7	←→
PB_0	—
PB_1	—
PB_2	—
PB_3	—
PB_4	—
PB_5	—
PB_6	—
PB_7	—
PC_0	I/O or $INTR_B$
PC_1	I/O or $\overline{OBF_B}$ or $\overline{IBF_B}$
PC_2	I/O or $\overline{ACK_B}$ or $\overline{STB_B}$
PC_3	$INTR_A$
PC_4	$\overline{STB_A}$
PC_5	IBF_A
PC_6	$\overline{ACK_A}$
PC_7	$\overline{OBF_A}$

MODE 0 OR MODE 1 ONLY

مابشتغلو mode 2

port B اذا كان (mode 0)

→ Mode 0 or 1 (port B)

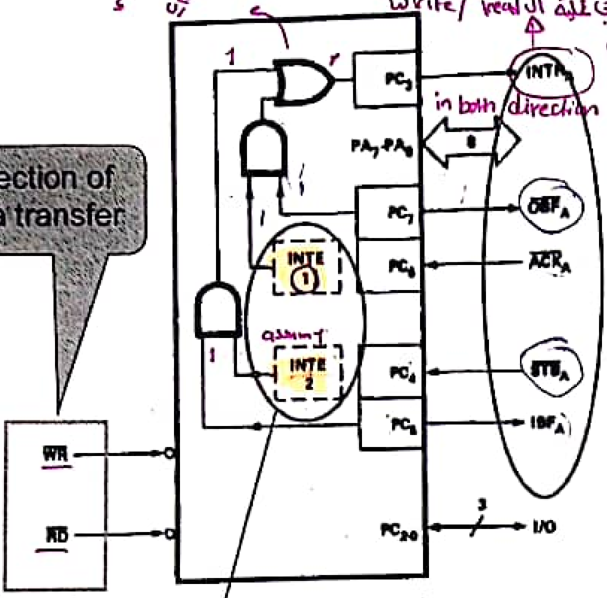
use 3 control signal if B in mode 1 (0,1,2) وحسب اذا in/out

control signal for port A

out → INTR
OBF
Ack

IN → INTR
IBF
STB

Mode 2 (configuration of port A)



Direction of data transfer

Port A (mode 2) control signals

2control read and write portA في bidirection

Port A-Input: INTE1=1 by setting PC4
Port A-Output: INTE2=1 by setting PC6

read → input device make data availiable activate STB ~ PPI الى transfer البيانات
PPI → active IBF ~ assume INTE = 1
generat INTR

write → microp intiate write operation and data will transfer from microp to the buffer, and buffer of PPI become full, OBF (0) active valid data over line, device read data and set ack. to inform PPI (read succefully, reactive buffer (1), assume INTE (1) ~ generate INTR to the microp.

Combined Modes

Port A-mode 2
Port B-mode 0 (input)

كما عرفنا انه يستخدم 5 control signal و الـ bidirectional

dosn't use any control signal direction input

CONTROL WORD

D7	D6	D5	D4	D3	D2	D1	D0
1	1	☒	☒	☒	0	1	1/0

bidrectional portA

mode0 inp

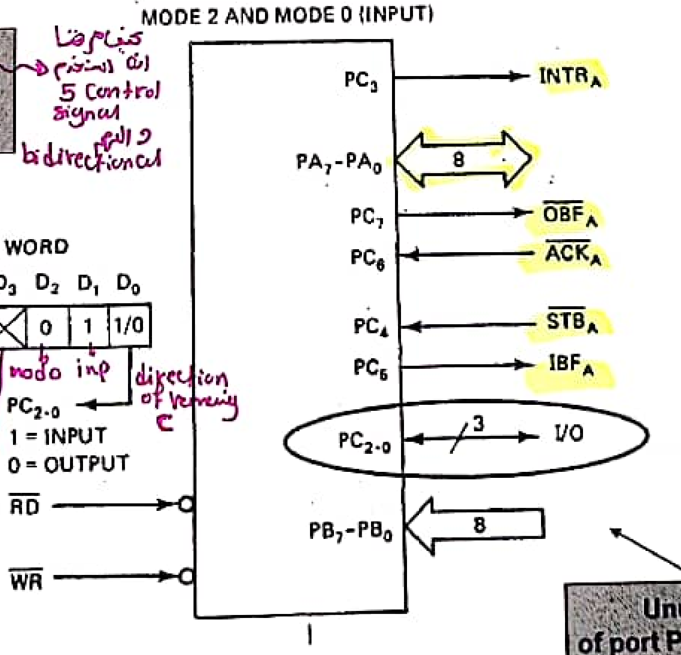
direction of remaining C

direction of A) dont-care

for C all is control data line لحالها

PC2-0
1 = INPUT
0 = OUTPUT

MODE 2 AND MODE 0 (INPUT)



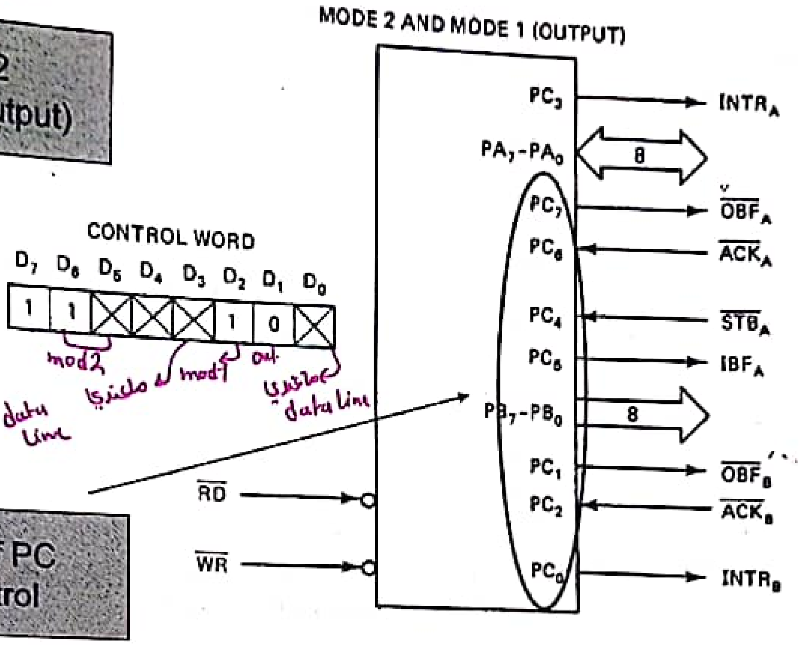
Unused bits of port PC; they can be used for general I/O of single or group of signals

Combined Modes (cont'd)

bidrechinl والاتجاهين 5 control signal استخدم

Port A-mode 2
Port B-mode 1 (output)

لازم استخدم 3 control signal



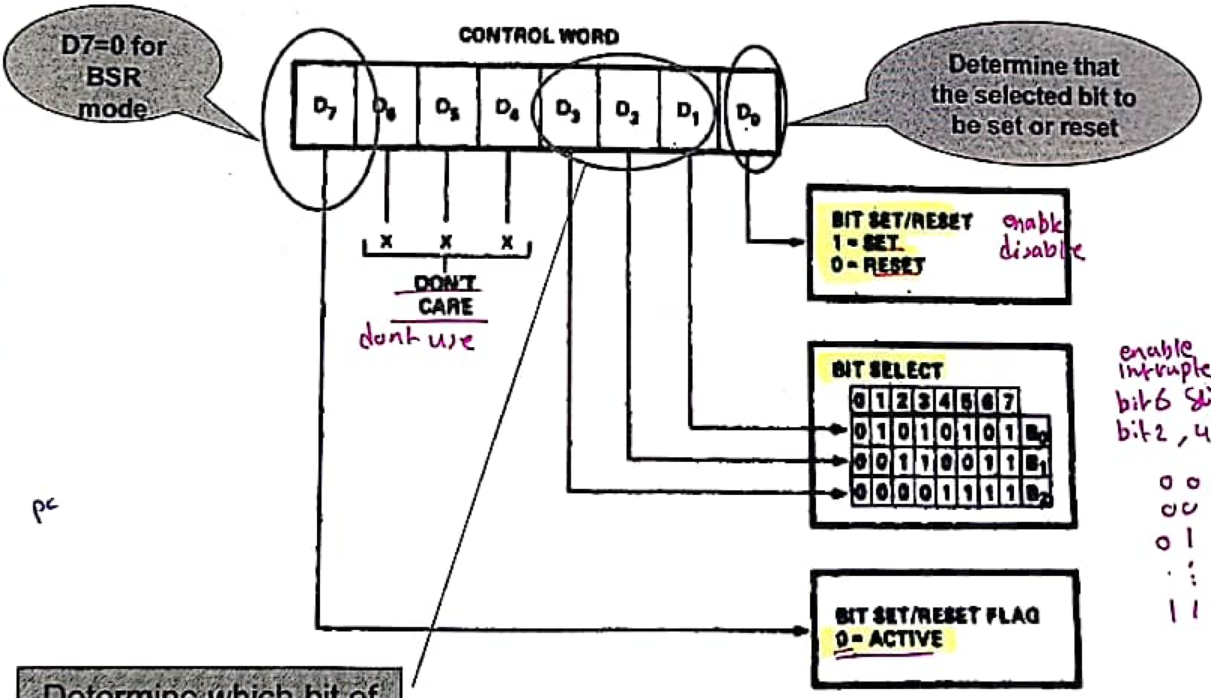
x assume x=0

$\underbrace{1\,1\,0\,0}_{C4}\,\underbrace{0\,1\,0\,0}$

Note that all bits of PC are used for control

~~in order to enable and disable Intrupt for A or B~~

Bit set/reset Mode (BSR mode)



D7=0 for BSR mode

Determine that the selected bit to be set or reset

BIT SET/RESET
1 = SET
0 = RESET

enable
disable

BIT SELECT

0	1	2	3	4	5	6	7	
0	1	0	1	0	1	0	1	B0
0	0	1	1	0	0	1	1	B1
0	0	0	0	1	1	1	1	B2

BIT SET/RESET FLAG
0 = ACTIVE

enable intruple كالي bit6 مش عايزه bit2, 4 ..

000 (bit 0)
001 (bit 1)
010 (bit 2)
⋮
111 (bit 7)

pc

Determine which bit of the 8 bits of port PC to be set or reset

enable Intrupt for port A ?

set bit number 6 of port C usig BSR mode

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	1	1	0	1,0

$\underbrace{1\;1\;0}_{Pc6}$; $\underline{1,0}$ set, reset.

بنبعث لل control Reg.

port B

$0\,\underline{0\,0\,0}\;\;\underbrace{0\,1\,0}_{2}\;\;\underbrace{0/1}_{set/Rest}$

Example 1: BSR mode

The interrupt control flag INTEA for port A is controlled by PC6. Using te BSR mode of 8255A. What configuration code must be written to the control register to set it to enable this control flag?

Solution:

1. D7=0
2. INTEA is to be set, hence, D0=1
3. To select PC6 then D3 D2 D1=110.
4. The remaining bits are don't care.

So.....

Control register=0XXX110$\underset{\text{set}}{1}$ or 00001101.

$\boxed{0D}$h

Example 2: BSR mode

address port A= base address   Isolated

Assume that the 8255 is mapped to the address 0080H in the I/O space:

write to control word

➢Ex: Write a $\boxed{\text{BSR}}$ word subroutine to set PC7 and PC3

enable

To Set PC7 → OFH : To set PC3 → 07H

enable

$\underbrace{0000}_{0}\,\underbrace{1111}_{F}$ $\underbrace{0000}_{0}\,\underbrace{0111}_{7}$

83 (address control Reg)

```
MOV AL,0FH
OUT 83H,AL
MOV AL.07H
OUT 83h,AL
```

] PC7

] Pc3

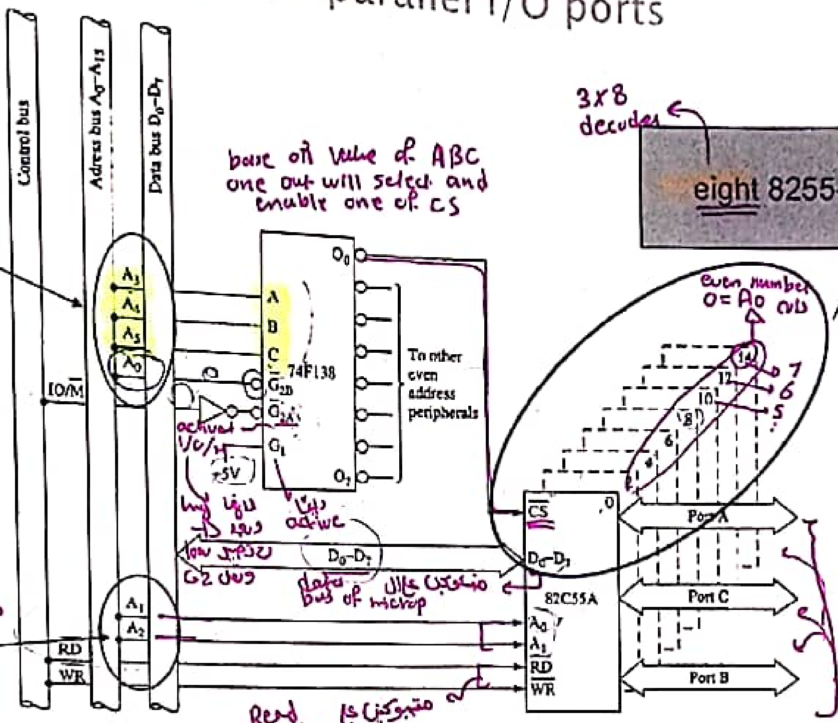
Note that we sent two different control words to the control register to set, respectively, PC7 and PC3.

0 00 0 1111

microprocessor with several ppl by using address decoder.

Fig 10.21

8255 implementation of parallel I/O ports



To select one 8255-PPI

Select عابلي نعلوا لواحدة ال PPI

000 → select out 0 رح تعمل select for First ppl

001 → select out. رح تعمل select ال second PPI

base on value of ABC one out will select and enable one of CS

activat V/c/4 لأنها دايماً low رح تعطي G2 دايماً active

3x8 decoder

eight 8255-PPI

even number 0 = A0 لأن

I can connect one ppl directly to the microp.

If I need to use several ppl I need to use decoder to select one out of 4, 8 ...

2x4 decoder
3x8 decoder

Connect to the I/O device رح يكونوا multiple of 2

مشبوكين على ال data bus of microp

with PPI ~~address~~ هو بأي يحدد ال mode اي نستخدم

Read write متبوكين ع

select one port in 8255-PPI

لأن A1A0 انخدم من A2A1

even or odd حسب إذا مستخدم A0

إذا استخدمه كام بيكون even استخدم لتفعيل ال enable ال G2 وهو active low

- 8255 parallel I/O ports in 8088 based Microcomputer.

zero يعني

* what should the value of port A in ppl 0 ?

0000h	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
	x	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0

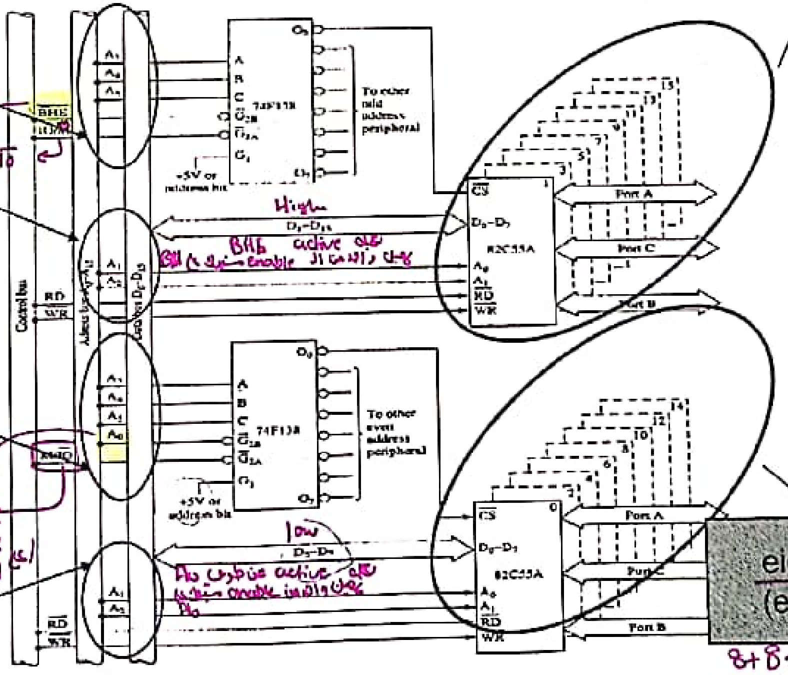
إذا عرفت فقط ال port ال A2A1 رح يحطوا

إذا اعتزت ال PPI ال A5A4A3 رح يعطوا

بعدت انه في Do بعدين معطى 10 (violated)
يعني ما 0 – 15 ←

8255 implementation of parallel I/O port

I want to interface microprocess 8086



eight 8255-PPI (odd address)

I can connect 2 ppl without using decoder because 8086 has 16 bit data bus, split in 2 part low high

إذا بدي several ppl بنشبك H ال decoder و L ال decoder

To select one 8255-PPI

high part of data bus نجزي الخلام النقل

و ال high address decoder نفعل BHE

M/IO

To select one 8255-PPI (odd address)

High

BHE active على شان ال enable بتستخدم ال BHE

To select one 8255-PPI

low part of data bus و ال low address decoder نفعل

active low (connect direct ال G2A)

To select one 8255-PPI (even address)

low

A0 بتكون active على شان enable ما يتم A0

eight 8255-PPI (even address)

8+8=16 ppl

3x8 , 3x8

- 8255 parallel I/O ports in 8086 based Microcomputer.

إذا استخدمت 2x4

رح يكون 4 لل high و 4 لل low = 8

32 ← 4x16

Example 1:

- What must be the address bus inputs of the circuit shown in Fig 10.21 if port C of PPI 14 is to be accessed?

IO   ↳ out⑦

- Answer:

1. To enable PPI 14, the decoder 74138 must be enabled and O7 must be 0 (active), G2B=0 and CBA=111.
2. A_0=0 to enable decoder (74138) and $A_5A_4A_3$=111
3. Port C of PPI is selected A1A0=10 or (A2A1=10 from the bus)
4. The rest of addresses are don't care

A_{13}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
x	x	x	x	x	x	x	x	x	x	1	1	1	1	0	0

$\boxed{0\ 0\ 3C}$ h

(7)
111

Example 2:

- Assume that PPI 14 of Fig 10.21 is configured so that Port A is output and C and B are inputs. All in mode 0. Write a program that input data from ports C and B and then find the difference between C and B (PC–PB) and then output it to port A?

طرح

① Config the PPI

- Answer: 10001011 [Control word] = 8B I should send it to ~~addr~~ Reg

اول شي بحسب ال address لل A, B, C.

1. To enable PPI 14, the decoder 74138 must be enabled and O7 must be (0) ~~enable~~ active, G2B=0 and CBA=111

A_0=0 to enable decoder (74138) and $A_5A_4A_3$=111

Port A address = 00111000 = 38H
Port B address = 00111010 = 3AH
Port C address = 00111100 = 3CH

mult of 2 number (even; A_0=0
multiple of 2; A_2A_1)

003E (value of the control register)

Mov AL,8B
out 3E,AL
ما طلع لانه
assume انه is configured

```
IN AL, 3Ah     : read port B
MOV BL, AL     Save data from port B
IN AL, 3C      read port C
SUB AL, BL
```

AH نفس الـ ط
بنزل acc

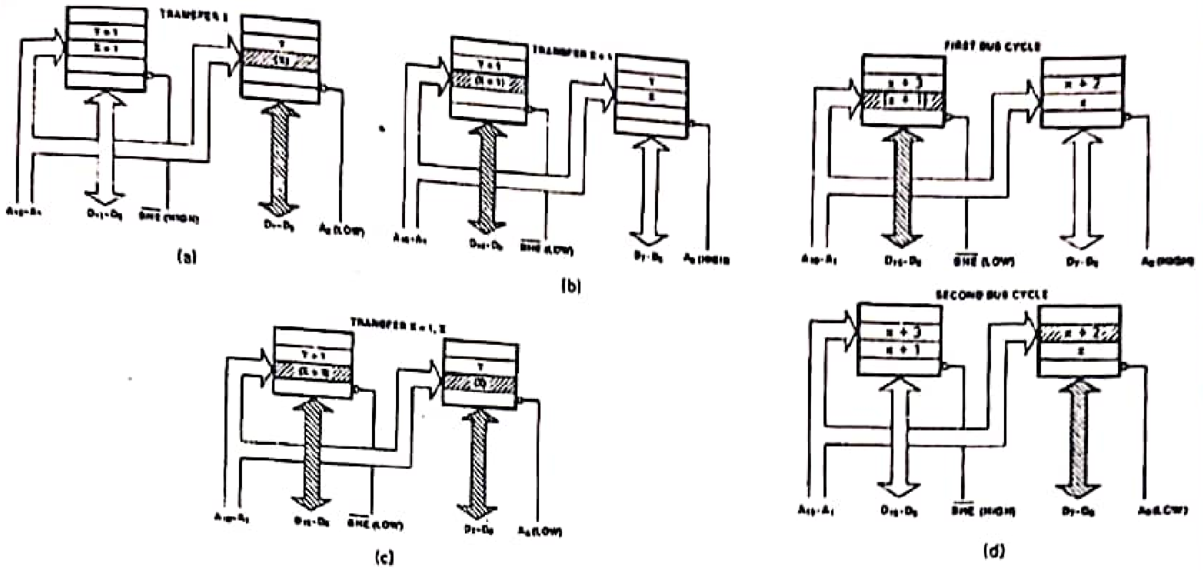
move Al,8B
out 3E,AL

```
OUT 38H, AL
```

عنوان
Port A

Memory-processor data transfer

7



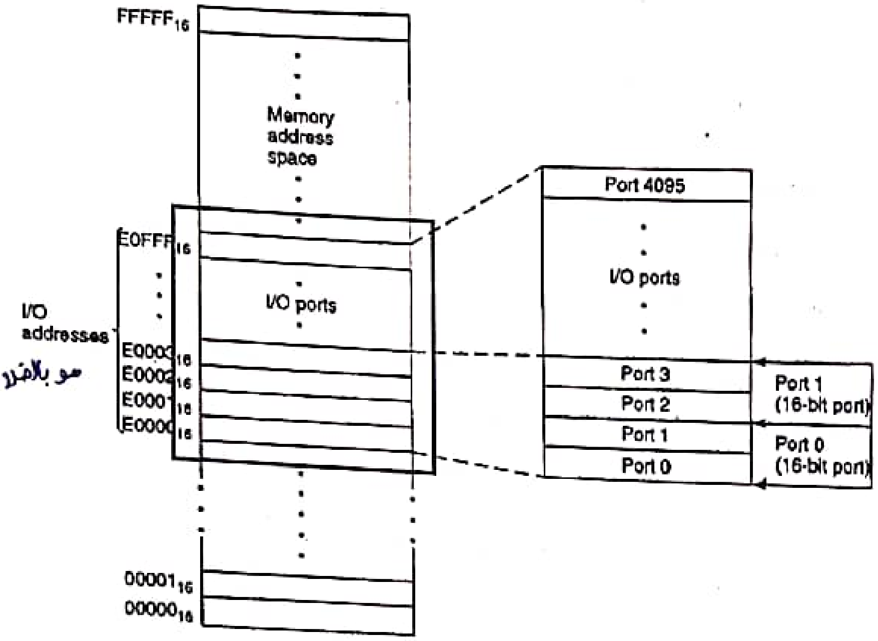
Misaligned word

Memory mapped I/O

- I/O devices can be mapped to the memory address space.
- MPU looks at the I/O port as a storage location in memory.
- In micro-computer with M-M I/O, some memory addresses are dedicated for I/O port.

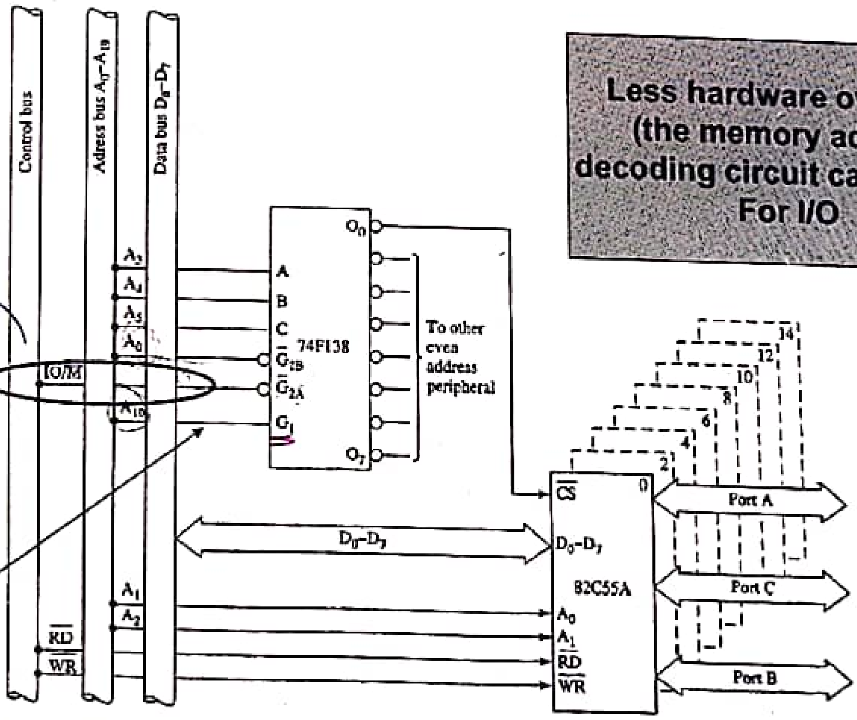
- Advantages:
  - Instruction that affect data in memory are used for I/O instead of special I/O instruction.
  - Hence, much more instructions and addressing modes are available for I/O operation.
  - For example data transfer can be performed not only with AL and AX but also with the other internal registers.
- Disadvantages:
  - Slower operations than those specially designed for I/O
  - Part of memory space is lost.

Memory-mapped I/O



هو بالضرورة يلزم فيه

Memory-mapped I/O



Less hardware overhead (the memory address decoding circuit can be used For I/O

Not inverted

Why????

اذا قلنا ال inverted
رح يكون memory mapped

Why A10 not Vcc?

دايمًا قيمتها 1
ادخل هل
use
For reserved location
يعني ادخل بيكون عن ال Range

بنستخدم location ال memory
access the reserved location of memory using memory instruction

بقدر استخدم الا Reg
وبستخدم ال address كامل
19 – 0

Example 1 for M-M I/O:

which port and ppi ?

- Which I/O port in Fig 10.23 is selected for operation when Memory address bus contains 00402H

معطيني الaddress
وبسأل اي ppi, واي port select ؟

يهمني A0 و A10 لانهم بدون تشاكين عال enable

- Answer:

1. Convert to binary: 000000000010000000010
   (10 9 8 7 6 5 4 3 2 1 0)
2. Then, we find, A10=1 and A0=0 and IO/Mprime=0 (memory operation)
3. Now, A5A4A3=000, then PPI 0 is selected
4. Moreover, A2A1=01, then Port B of PPI 0 is selected.

دو في غير سؤال بيعطي
address و ccl
وبحكي غير عل ال ccl
لحتى يعطيني عاد ال address

① بحوّلها لل binary
② لازم اتأكد اذا address decoder بشتغل ، اذا ما كان بشتغل بحكيله non of ppi select

A10 A0
0 1 00 0000 0010
PPI0 port B

Example 2 for M-M I/O:

- Write a sequence of instruction to initialize the control register of PPI 0 in Fig 10.23 so Port A is output and ports B and C are inputs, all ports in mode 0.

D7 D6 D5 D4 D3 D2 D1 D0
1 0 0 0 1 0 1 1

- Answer:

نبعث ال control world to control Reg.

1. value Control ~~register~~ word=10001011=$\boxed{8BH}$
2. Memory address of PPI0=000000000010000000110=00406 (A0, A1, PPI0, control Reg.)
3. Code:

هون لحتى ال DS تكون zero

```
MOV AX, 0
MOV DS, AX
MOV AL, 8BH
MOV [406H], AL
```

any Reg.

DS: 406
offset address

لحتى يكون ال address نفسه مايرجع تاني address لوكان قيمته تالفه
DS → shift left +0
وبعدها بنجمعها الoffset.

MOV [406H], 8B ← OR

Example 3 for M-M I/O:

- Assume the same configuration of Example 2. Write a sequence of instruction so that the content of B and C are ANDED then output the result to port A.

- Answer:

1. Address of the ports: address port A=00400, address port B=00402, and address port C=00404.
2. Code:

in priviay example

```
MOV AX, 0
MOV DS, AX
MOV AL, 8BH
MOV [406H], AL
MOV BL, [402H]
MOV AL, [404H]
AND AL, BL
MOV [400H], AL
```

بزبط احط وحده بس

AND AL, [402]

بنقدر اعمل and واحد منهم Reg: والثاني memory

result

لازم يكون ب Regist

لانه مابتنفع من ميموري ل memory

Microprocessor Systems

Chapter 11

Interrupt Interface of 8088/8086 Microprocessors

External or Internal
× external device or event request a service from microprocessor
× Normaly operation in microp. execute main program. and given attention

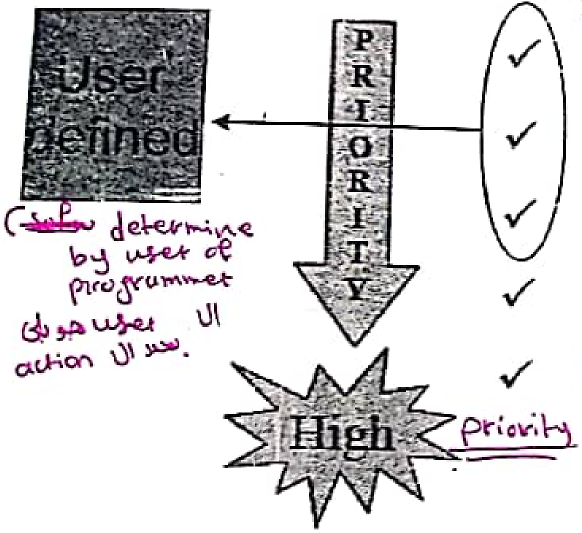
INTERRUPT INTERFACE

مقاطعة

Interrupts provide a mechanism for quickly changing program environment.

The section of the program which the control is passed: Interrupt Service Routine,
ex: For printers it is the printer driver.

8088 and 8086 interrupts: 256



(determine by user of programmer
اللي بحددها user
action اللي بحدد.

priority

1. ✓ External Hardware Interrupts — can't be mask by external hardware interrupt / can't be ignor by external HW — less priority
2. ✓ ~~Nonmaskable Interrupt~~
3. ✓ Software Interrupts — 224
4. ✓ Internal Interrupts — 32 Int. — system defined action tak by the micrp defined by syst. hotwer
5. ✓ Reset ← high priority (not include to the 256)
   ↳ just click to the Reset (micro لل يعيد)

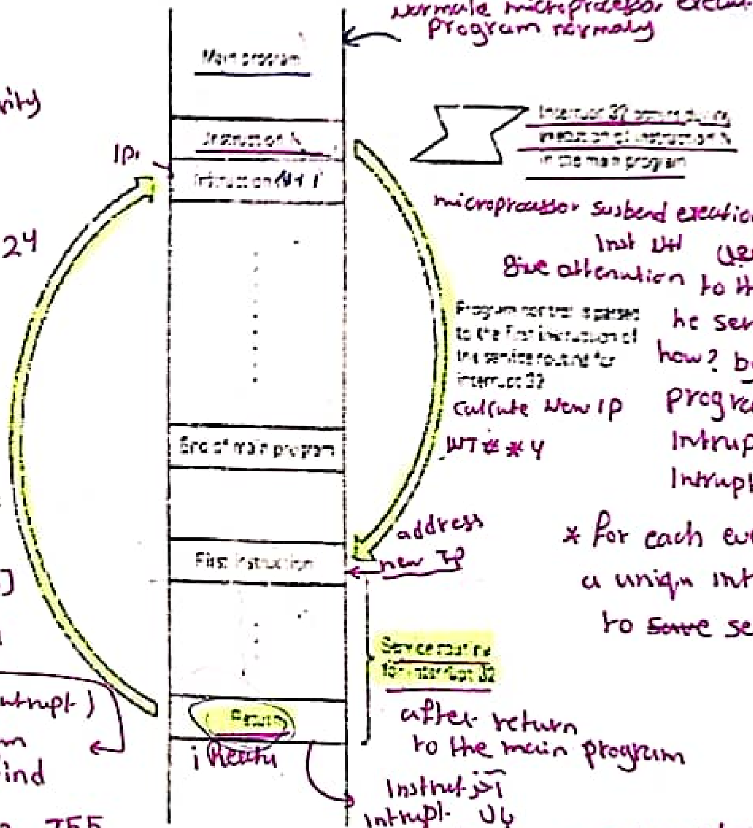
الاولى priority 0, 1, 2, 3, 4 ← .. 31
لها 5 اسم
internal interrupt (has 5 Intrupt)
System defind

32 - 255 (user defined)

Called Interrupt type number or vector

we have 256 Intrupt handlar in order to serve the event.

256 interrupts are supported by 8088/8086. They are Categorized into 5 Categories



IP
Main program
Instruction N
Instruction N + 1
End of main program
First instruction
Return
Interrupt 32 occurs during execution of instruction N in the main program
Program control is passed to the first instruction of the service routine for interrupt 32
Service routine for interrupt 32

normale microprocessor execut the main Program normaly

interrupt 32 يعني

microprocessor susbend execution Inst لل N

give attention to the Intrupt (event) he serve (عم يخدمه
how? by executed special program called Intrupt handlar or Intrupt service routine

Calculate New IP
INT # * 4

address new IP

* For each event we have a uniqe intrupt handlar to ~~save~~ serive Intrupt

after return to the main program

Instruct آخر Intrupt handler لل

Intrupt returned (restor old value of PI)
يعني شو اللي قعيدة حفظنا و الان Stack بنرجع
① flag ② CS ③ IP (by pop vale)

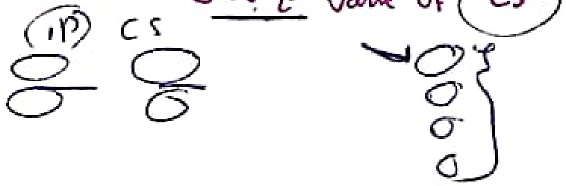
we have a relashionship between Intrupt # and start address of the handler, (next slide)

8088/8086 Interrupts

- An interrupt is an external event which informs the CPU that a device needs service
- In the 8088 & 8086 there are are a total of 256 interrupts (or interrupt types)
  - INT 00
  - INT 01
  - ...
  - INT FF
- When an interrupt is executed, the microprocessor automatically saves the flags register (FR), the instruction pointer (IP) and the code segment register (CS) on the stack and goes to a fixed memory location.
- In 80x86, the memory location to which an interrupt goes is always four times the value of the interrupt number
- INT 03h goes to 000Ch First address value of new IP

3x4 → 12 → C

000E value of CS



in the beginning of memory at address zero we have a Table (IVT) in this Table we store the starting address of Intrupt handler of each Intrupt number

Interrupt Service Routine

- For every interrupt, there must be a program associated with it
- This program is called an Interrupt Service Routine (ISR)
- It is also called an interrupt handler
- When an interrupt occurs, CPU runs the interrupt handler but where is the handler?
  - In the interrupt Vector Table (IVT)

total number is 256 Intrupt and each has Intrupt handler

INT Number	Physical Address	Contains
INT 00 each # store 2 Vector occupied 4 Byte 2 for IP, 2 for CS	00000h 0 1 2 3	IP0:CS0 2byte 2byte
INT 01	00004h 4 5 IP 6 7 CS	IP1:CS1
INT 02	00008h 9 10 11	IP2:CS2
. . .	0000C . . .	. . .
INT FF	003FCh 3FD 3FE 3FF	IP255:CS255

logical address
from IP
Base + offset → from CS
and each number contain 2byte

Vector Table

العلاقة بين ال INT# و ال address
(multiply by 4)

0×0 → 0
1×4 → 4
2×4 → 8
⋮

INT# ×4 + 2 → address of CS
بيتخزن فيه

0 - 255
00 - FF
FF × 4

(0 - 255)
(00 - FF)

Total size 256 × 4 → 1K (1024) byte

$\boxed{0 - 3FF}$

$\boxed{400}$

Interrupt Vector Table

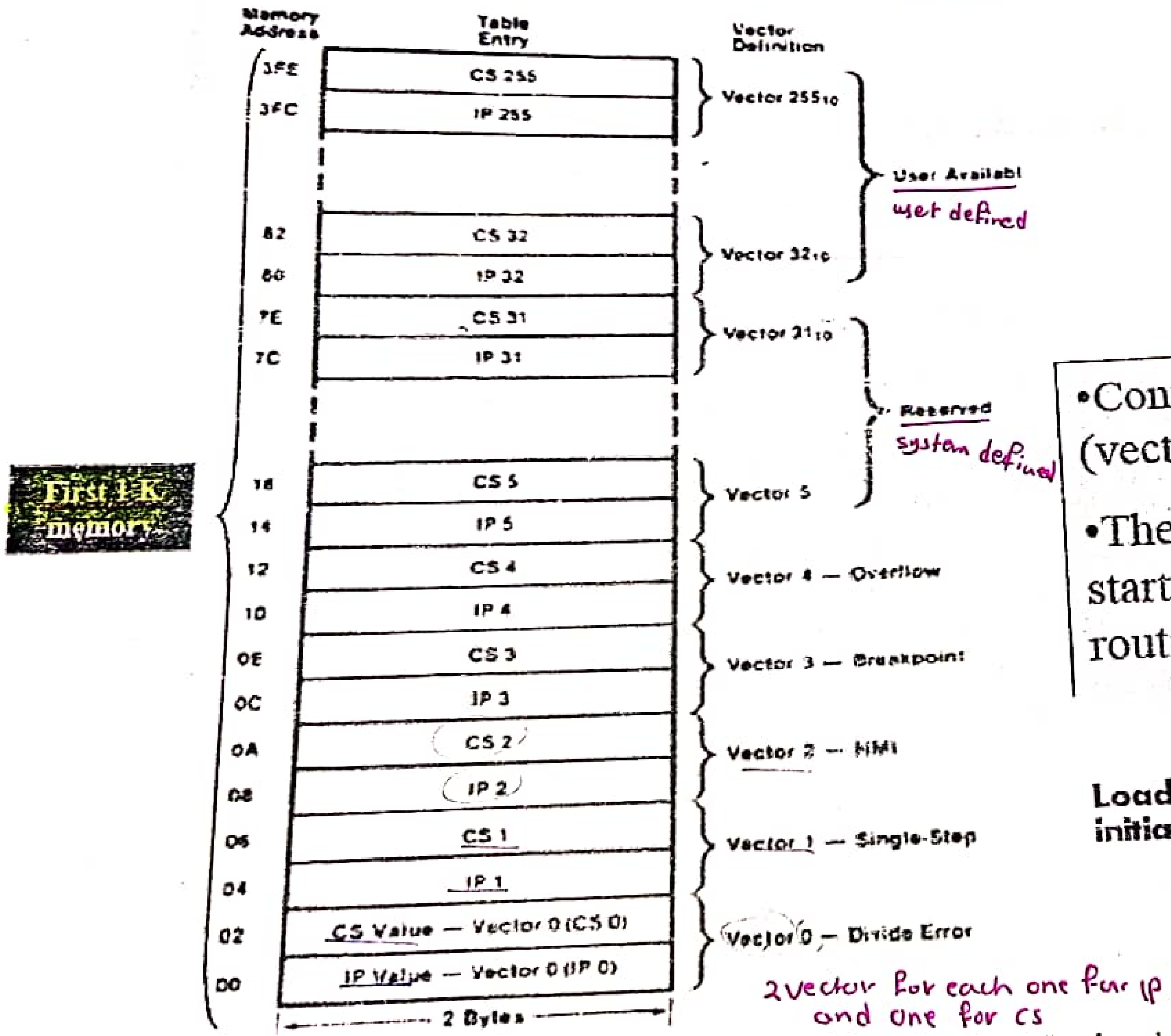
- Interrupt vector table consists of 256 entries each containing 4 bytes.
- Each entry contains the offset and the segment address of the interrupt vector each 2 bytes long.
- Table starts at the memory address 00000H.
- First 32 vectors are spared for various microprocessor operations.
- The rest 224 vectors are user definable.
- **The lower the vector number, the higher the priority.**

serve the less number

INT 3 →

INT 4

Interrupt Vector Table



- Contains 256 address pointers (vectors)
- These pointers identify the starting location of their service routines in program memory.

Loaded to this part of memory as a system initialization or as a firmware

Simple firmware typically reside in ROM or OTP/PROM, while more complex firmware often employ flash memory to allow for updates. Common reasons for updating firmware include fixing bugs or add to the device

Example

decimal

For example: vector 50: CS and IP?

decimal → hexa

Physical Address 200 = (4 x 50) = 200 = 11001000 = C8H

000C8 contains IP: and 000CA contains CS information

+2 ⇒ new for CS

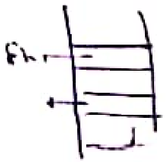
- INT 12h (or vector 12)
- The physical address 30h (4 x 12 = 48 = 30h) contains

  0030h and 0031h contain IP of the ISR

  0032h and 0033h contain CS of the ISR

Interrupt instructions

- Interrupt enable flag (IF) causes external interrupts to be enabled.
- INT n initiates a vectored call of a subroutine.
- INTO instruction should be used after each arithmethic instruction where there is a possibility of an overflow.
- HLT waits for an interrupt to occur.
- WAIT waits for TEST input to go high.



any req from any External device will ignore

Interrupt Instructions

set c reset. علاقة ال interrupt flag

Mnemonic	Meaning	Format	Operation	Flags Affected
CLI	Clear interrupt flag	CLI	0 → (IF) In order to accept intrpt request from external hardware device For external hardware intruple	IF Intrpt flag
STI	Set interrupt flag	STI	1 → (IF)	IF
INT *n* Intrupt Instruction	Type n software interrupt any intrupt request اذا اكانكرد. اشغل accept this micro. فال should execut this Instruction and it's take one operand (n) intrupt vector number	INT n	(Flags) → ((SP) − 2) Top pointer 0 → TF,IF ← clear address (CS) → ((SP) − 4) bush to CS (2 + 4 · n) → (CS) (IP) → ((SP) − 6) bush to Ip (4 · n) → (IP) updat sp يعاد بحسب ال address الجديد after top of stack يخزن ال flag	TF, IF
IRET	Interrupt return revirce operation of Intrupt Instruction	IRET	top ((SP)) → (IP) ((SP) + 2) → (CS) ((SP) + 4) → (Flags) (SP) + 6 → (SP) بعد pop	All
INTO	Interrupt on overflow نفس INTn و 4=n → interrupt أول internal بال interrupt 0,1,2,3,4 الخمس	INTO	INT 4 steps	TF, IF
HLT	Halt	HLT	Wait for an external interrupt or reset to occur	None
WAIT	Wait	WAIT	Wait for TEST input to go active	None

ch = CS + Ip

2C02A = 2C00 + Ip

X Ip = 2942A + 0

0 1 2 (4)

susbent operation (stope operation of microp) وسبب ال intrupt ← HLT

نفس ال HLT ← WAIT

نفعل susbent operation of microp.

base on بيتشكي test signal (if $\overline{Test}$ = 0 (active) so, it's we can use wait)

0 - 31

(0) 1, 2, 3, (4)

The Operation of Real Mode Interrupt

1. The contents of the FLAG REGISTERS are pushed onto the stack
2. Both the interrupt (IF) and (TF) flags are cleared. This disables the INTR pin and the trap or single-step feature. (Depending on the nature of the interrupt, a programmer can unmask the INTR pin by the STI instruction)
3. The contents of the code segment register (CS) is pushed onto the stack.
4. The contents of the instruction pointer (IP) is pushed onto the stack.
5. The interrupt vector contents are fetched, and then placed into both IP and CS so that the next instruction executes at the interrupt service procedure addressed by the interrupt vector.
6. While returning from the interrupt-service routine by the instruction IRET, flags return to their state prior to the interrupt and and operation restarts at the prior IP address.

① suspend operation of main program
② push value into the stack flag, CS, IP
③ calcute new vale of. IP, CS
by multipleg number ~~for~~ by 4 to get address new
IP, +2 to get CS
④ jump to exect Interrupt handlin
⑤ last instruction is return to pop the 11 old value and return to execute program

First 5 Intrr. are the most priority

called: اذا جاء عنا مشكلة بحالة ال divide
من القسمة ع zero

INT 00 (divide error)

```
MOV AL,92        // AL=92
SUB CL, CL       // CL=0
DIV CL  ; 92/0 undefined
```

$\frac{92}{0} = \frac{AX}{CL}$ بنقسم

ممكن الرقم الناتج (مجموع او الباقي)
واه مم الجزء من size the diderelion

```
; Also invoked if the quotient is too large to fit into the assigned register
MOV AX,0FFFh
MOV BL,2
DIV BL
```

الناتج الكبير = $\frac{0FFF}{2}$
من حجم ال
Reg AL

```
; WRITE A DIVIDE ERROR ISR
Prompt db 'Division by zero attempted$'
Diverr: PUSH DX
Mov ah,09h
Mov dx, offset prompt
int 21h
POP DX
```

lapel — defined bayte (allocate)

lable — أول char جوا ال ... لـ اول

نهاية ال string

store this in memory

Subroutin
PUSH تجعل مجموعة
body دريجها لل
pop ونرجعها مجموعة
Reg وبكونوا هدول
انا استخدم بال body
وما بدي قيمهم تتغير
وبكونوا كترتب المعاني

32 255
20 – FF
(user)
user

Intrupt 21h
مربوطة بقيمة ah
down string
الموجود ال offset dx
ع ال monitor

iret → return to main program

لانه استخدم بال dx
body لهيك عملت
push و pop
قيمة ال dx ما بتتغير

INT 01 (Single Step)

(tray) Trap Flag نفس مبدأ عمل ال

=1 → Single stelling mode بعمل enable ل (I will execute program instruction by exkation and show the output of each instruction, بعمل execute for first instruction ~~of program then via Interrupt~~ (INT01)

*In executing a sequence of instructions, there is often a need to examine the contents of the CPU's registers and system memory.

*This is done by executing one instruction at a time and then inspecting the registers and memory

*This is called the tracing or the single stepping

يعني تتبع الخروج

*TF must be set (D8 of the flag register)

بعمل show the output of execution of program and get return to the main program to execute the second instruction

بعد ماننفذ بعمل rise to Intrupt

لشي؟ لحتى يعرض ال out بعد مانفذ ال second Instruction

بعد return to program وبعمل execute instruction وبعرف

اسمه single step لانه بنفذ instruction وبعرضها

الفائدة : debbuging (testing)

عندي بس طريقة وحدة اغير فيها عن طريق ال PUSHF و POPF

```
PUSHF
POP AX
OR AX,0000000100000000B
PUSH AX
POPF
```

PUSHF (push flag Reg into stack after top of stack) خزنت كل ال Reg بال stack

POP AX — General purpose — طلعت لي خزنتها ع ال AX

set — OR

Toggle XOR
Reset AND

↳ set for 8 (trac flags) اغير فقط بـ AX عشان ارجعه ع ال Flag Reg.

PUSH AX — رجعت كل AX ع ال stack

POPF — بحط القيمة الموجودة بال stack و ادخلها بال Flag Reg.

Other Interrupts

- INT 02h (NMI)
  - Intel has set aside INT 02h for the NMI interrupt non maskable
  - There is an NMI pin on the CPU اله علاقة بال pin NMI
  - If the NMI pin is activated by a H signal, the CPU jumps to 00008H 2x4 to fetch the CS:IP of the ISR associated with NMI
- INT 03h (breakpoint) similer to single step Interupt بعمل after execution group of Inst not one فاهمه لما يكون عدد ال Inst كبير
- INT 04H (signed number overflow) or INTO,
  - If OF=0 goes to 00010h to get the address of the ISR
  - Otherwise, it is equivalent to NOP
- Example: Use debug dump command to see the IVT
  - D 0000:0000 0013

Differences between INT and CALL

2 type of subrotine
- near (Progeser - within code segment)
- far (Proceyure in other code segment)

do same think ↩ ① push old value ② jump to other location ③ after finish return to main program

❖A CALL FAR instruction can jump any location within the 1 MB address range but INT nn goes to a fixed memory location in the Interrupt Vector Table to get the address of the interrupt service routine

If I want to call a near I want to exchang just IP

IF I want to call Far I want to chang CS and IP

❖A CALL FAR instruction is used by the programmer in the sequence of instruction in the program but externally activated hardware interrupt can come at any time

difference:

① Far do push the value ② CS, IP but INT push 3 Flags, CS, IP

نفس الcall بال pop ال INT بنعمل 3 value لل pop وال Far بنعمل pop ل 2 value

❖A CALL FAR cannot be masked but INT nn in hardware can be blocked.

② call Far I should executed this call but If I have Interrupt instruction depend in priority (execute or ignor) like ~~external hardware~~ Interrupt for number to the number I have

❖A CALL FAR saves CS:IP but INT nn saves Flags and CS:IP

❖At the end of the subroutine RET is used whereas for Interrupt routine IRET should be the last statement

③ I know when I execute the call instruction محله محدد بالبرنامج

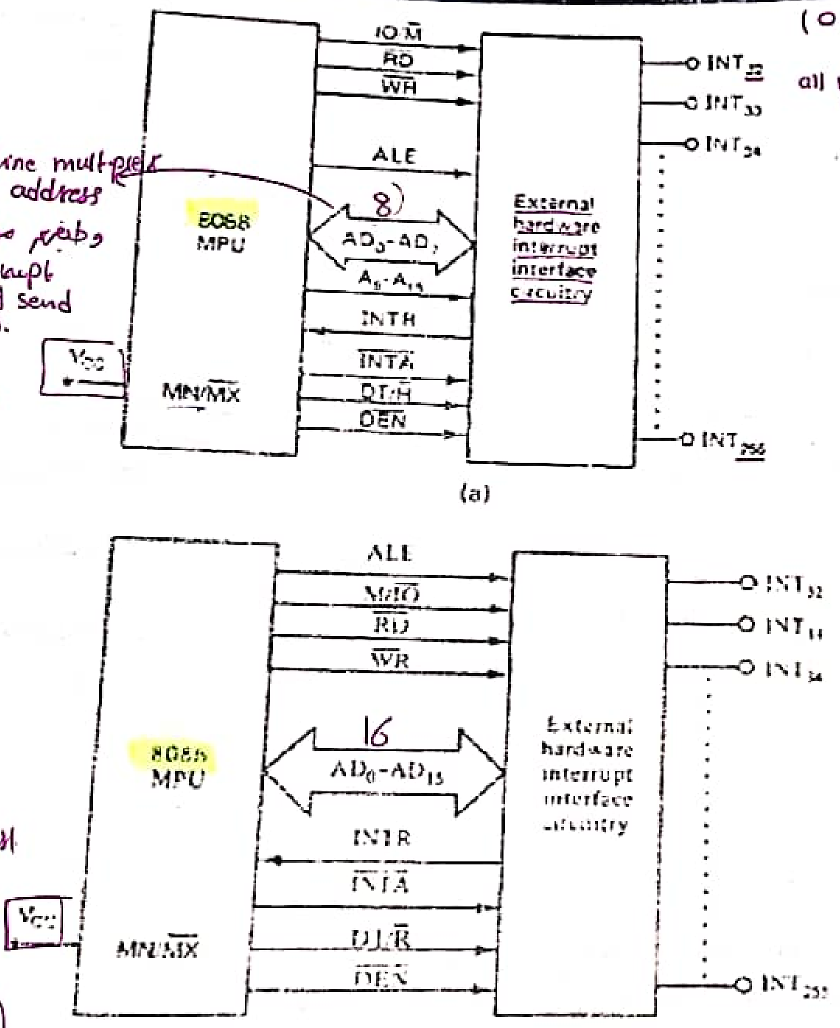
Interrupte → ما اله وقت ومكان محدد

④ Procesyer of Prog. can write any Subrotin and Store it in any location but Interrupt should store in specific location

⑤ Interrupt can happen in any time not like in Call

External Hardware Interrupt Interface

Connect the Intrupt device with micropracesre 8088



(a)

(b)

data line multplexe with address

Carry data ويجيب و manage Intrupt request and send it to microp. to serve it (should send its number)

(0-31) system defined

all user define intrupt

Minimum Mode

Because the MAX pin to VCC

✓The interrupt circuitry must identify which of the pending interrupts has the highest priority.

✓Then passes its type number to the MPU

✓The MPU samples the INTR at the last clock period of each instruction execution cycle. Its active high level must be maintained.

✓When recognized INTRA generated.

microp sample check the value after finish IV

دايمًا بتعامل ع ال interuple و IF بحاول عمل sushat program

the step that will help the external hardware in order to mange request in external hardware devices

① External Hw will connected directly to external Hw intrupt ccl by using (Intrupt input) = 242
the device that request service is one and the device not use any server is zero.
Active request they will be resolved by external hardware in order to understan what of one that has higher priority

② External resolved priority of ~~dente~~ active request.

③ after defined the higher priority it will rise to Intrupt req. signal to mi

② In order to request intrupt tag number and send it over data line after that microp. to remain steps

send $\boxed{IF = \phi}$
① in order to infor the Intrupt ccl has been accepted

when microp. reseved to Intrupt request he will check the value of $\boxed{IF}$ if set to 1 microp accept the Intrupt ~~① inorder~~

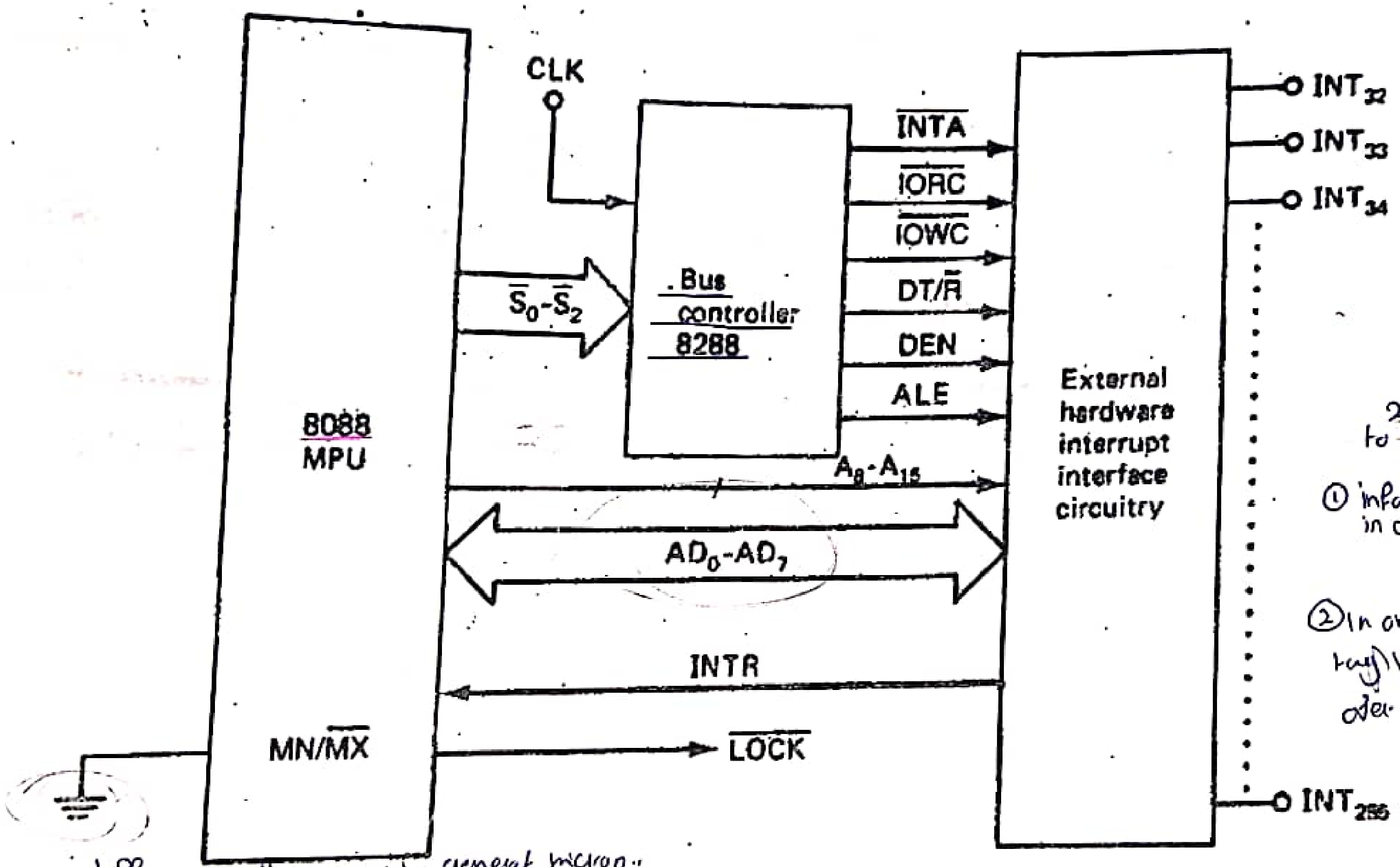
Divice
1, 2, 3
1 / 1 / ϕ req
يعني ال Priority
number 1 higher and

External hardware-interrupt Interface

- Minimum mode hardware-interrupt interface:
  - 8088 samples INTR input during the last clock period of each instruction execution cycle. INTR is a level triggered input; therefore logic 1 input must be maintained there until it is sampled. Moreover, it must be removed before it is sampled next time. Otherwise, the same interrupt Service is repeated twice.
  - $\overline{INTA}$ goes to 0 in the first interrupt bus cycle to acknowledge the interrupt after it was decided to respond to the interrupt.
  - It goes to 0 again the second bus cycle too, to request for the interrupt type number from the external device.
  - The interrupt type number is read by the processor and the corresponding int. CS and IP numbers are again read from the memory.

Maximum Mode-External hardware interrupt.

the operation in MAXimum
not general control signal



Microp:
2state [illegible] 2'
to coprocesser in order to gen intrup ack.

① inform device in order to request / accipt

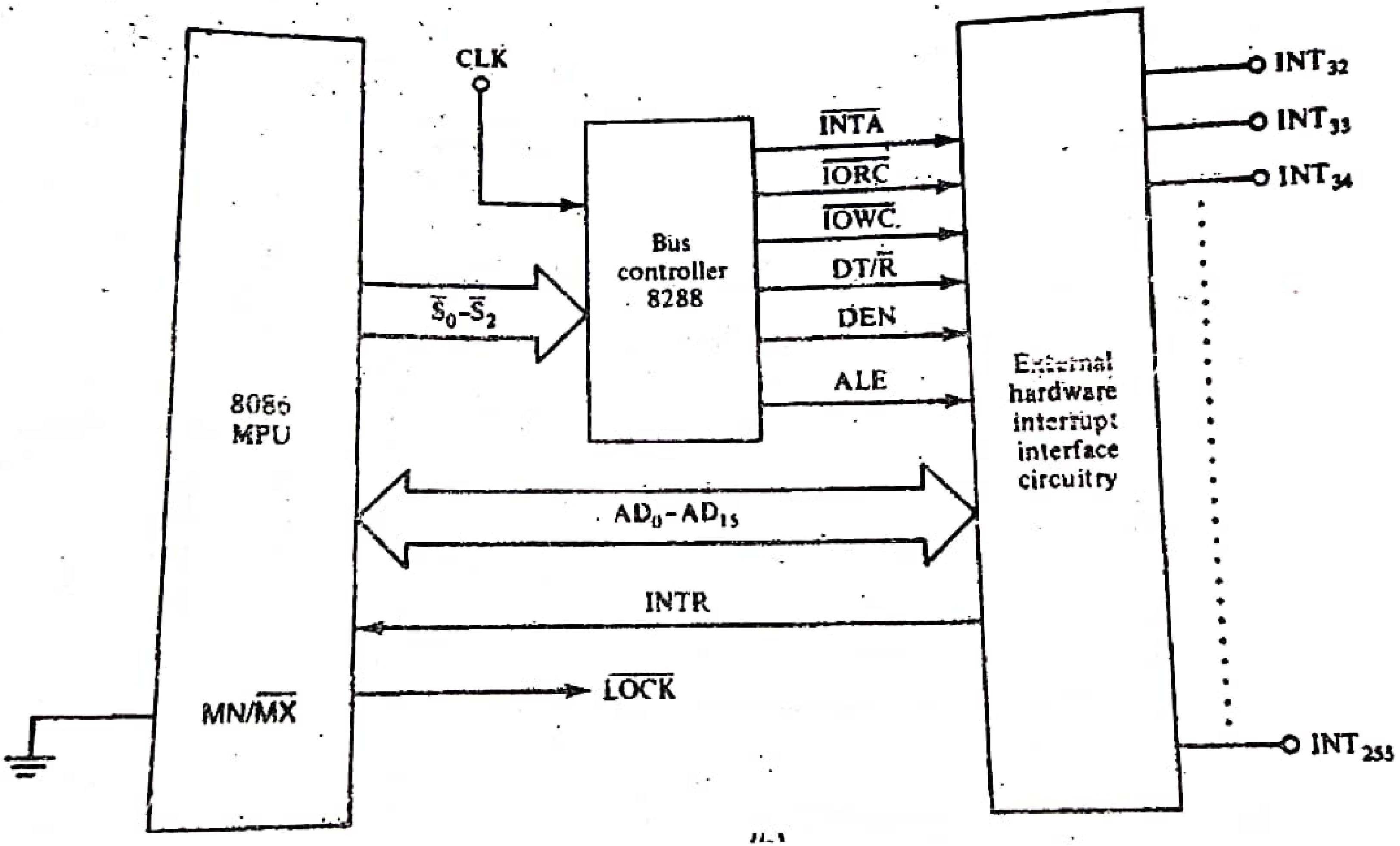
② In order to request humber type interupt to be send over data line

difference theg way to generat microp.:

① in Minimum mode directly control signal

② in Maximum mode send stauts code and baye the value to coprocessor generate control signal

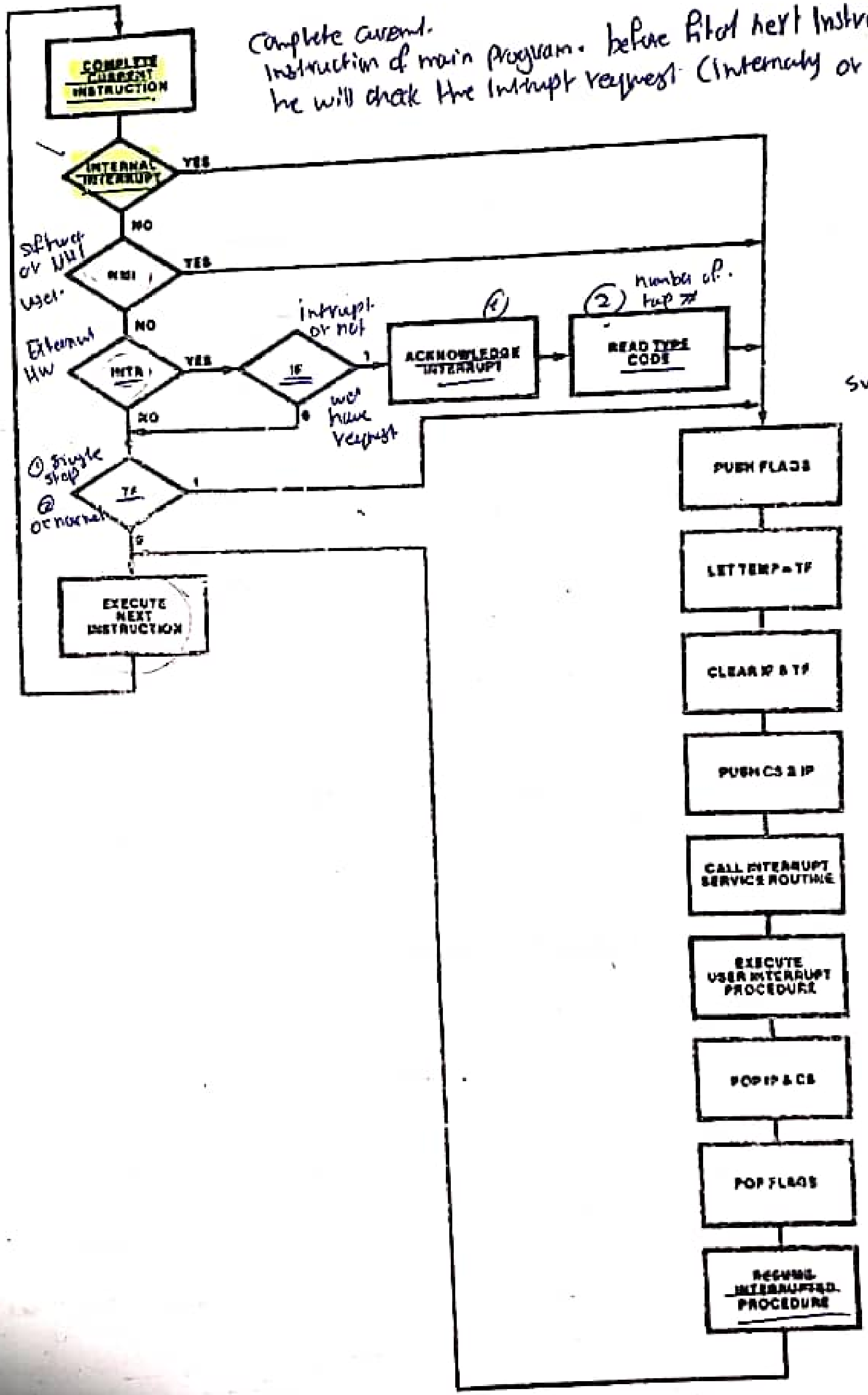
Maximum Mode-External hardware interrupt (cont'd)



Status inputs			CPU cycle	8288 command
$\overline{S}_2$	$\overline{S}_1$	$\overline{S}_0$		
0	0	0	Interrupt acknowledge	$\overline{\text{INTA}}$
0	0	1	Read I/O port	$\overline{\text{IORC}}$
0	1	0	Write I/O port	$\overline{\text{IOWC}}$, $\overline{\text{AIOWC}}$
0	1	1	Halt	None
1	0	0	Instruction fetch	$\overline{\text{MRDC}}$
1	0	1	Read memory	$\overline{\text{MRDC}}$
1	1	0	Write memory	$\overline{\text{MWTC}}$, $\overline{\text{AMWC}}$
1	1	1	Passive	None

2 start up
cod to generat

11.6 External Hardware-Interrupt Sequence



suspend

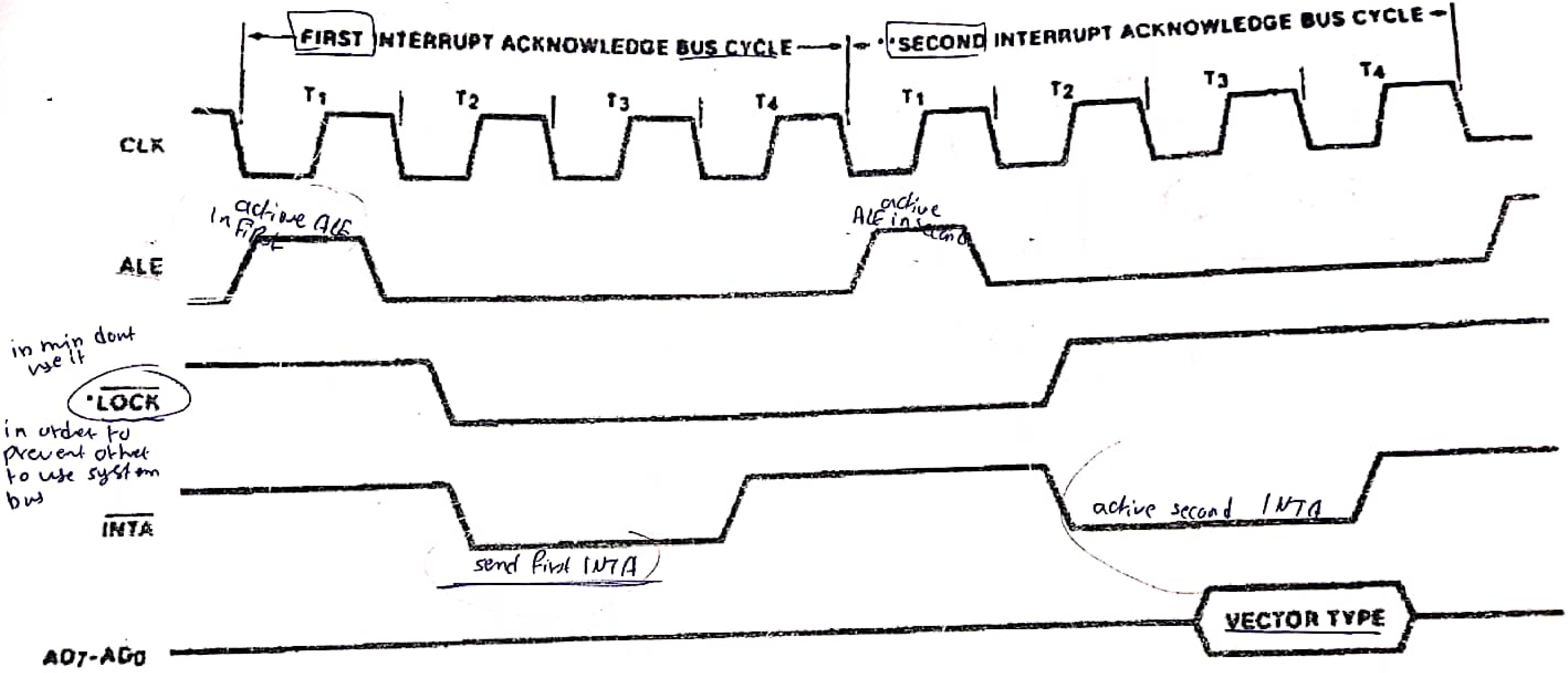
push old vale

clear vale IF, TF

calculate new vale of TF, IP

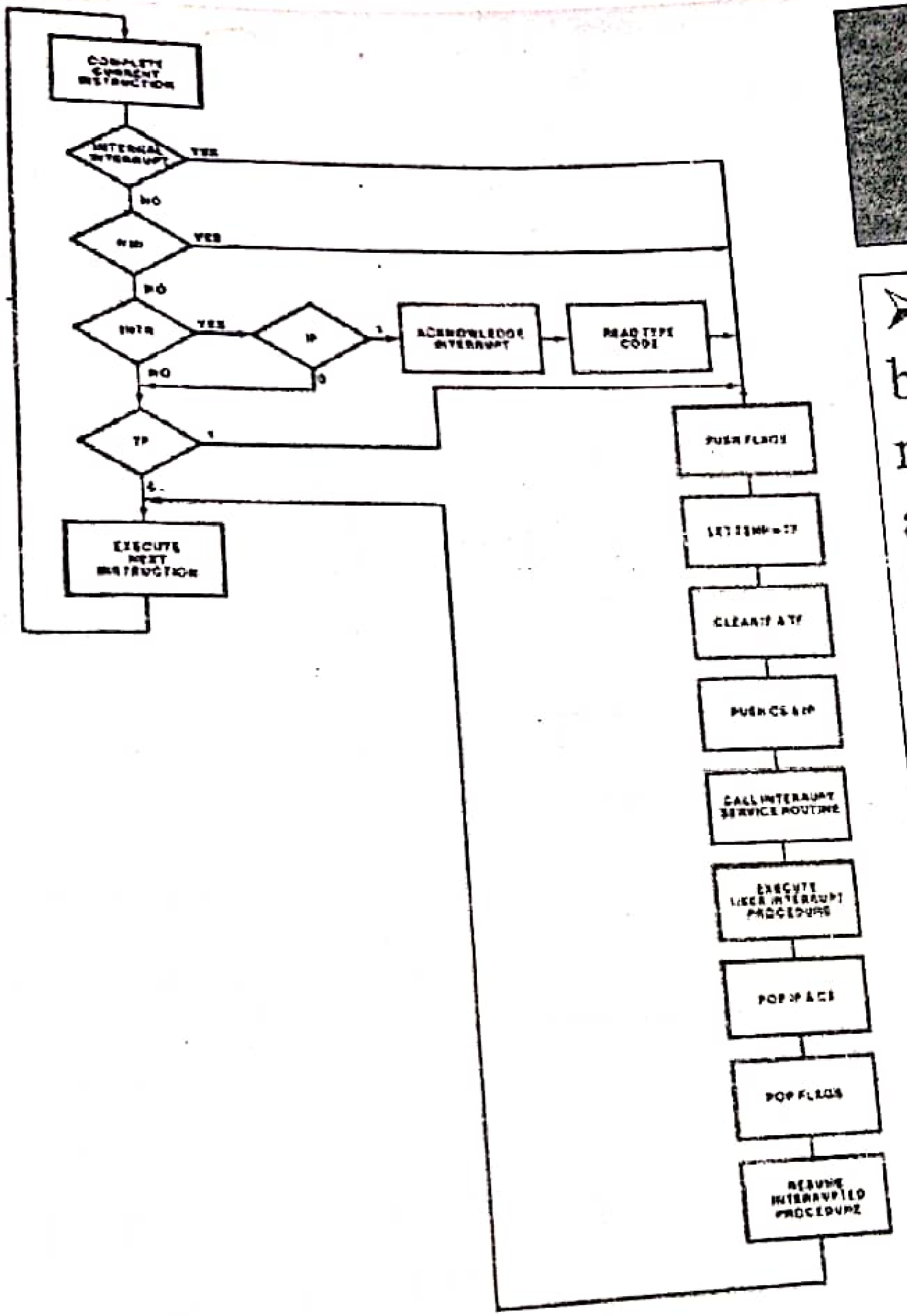
intrupt return

External hardware-interrupt Sequence



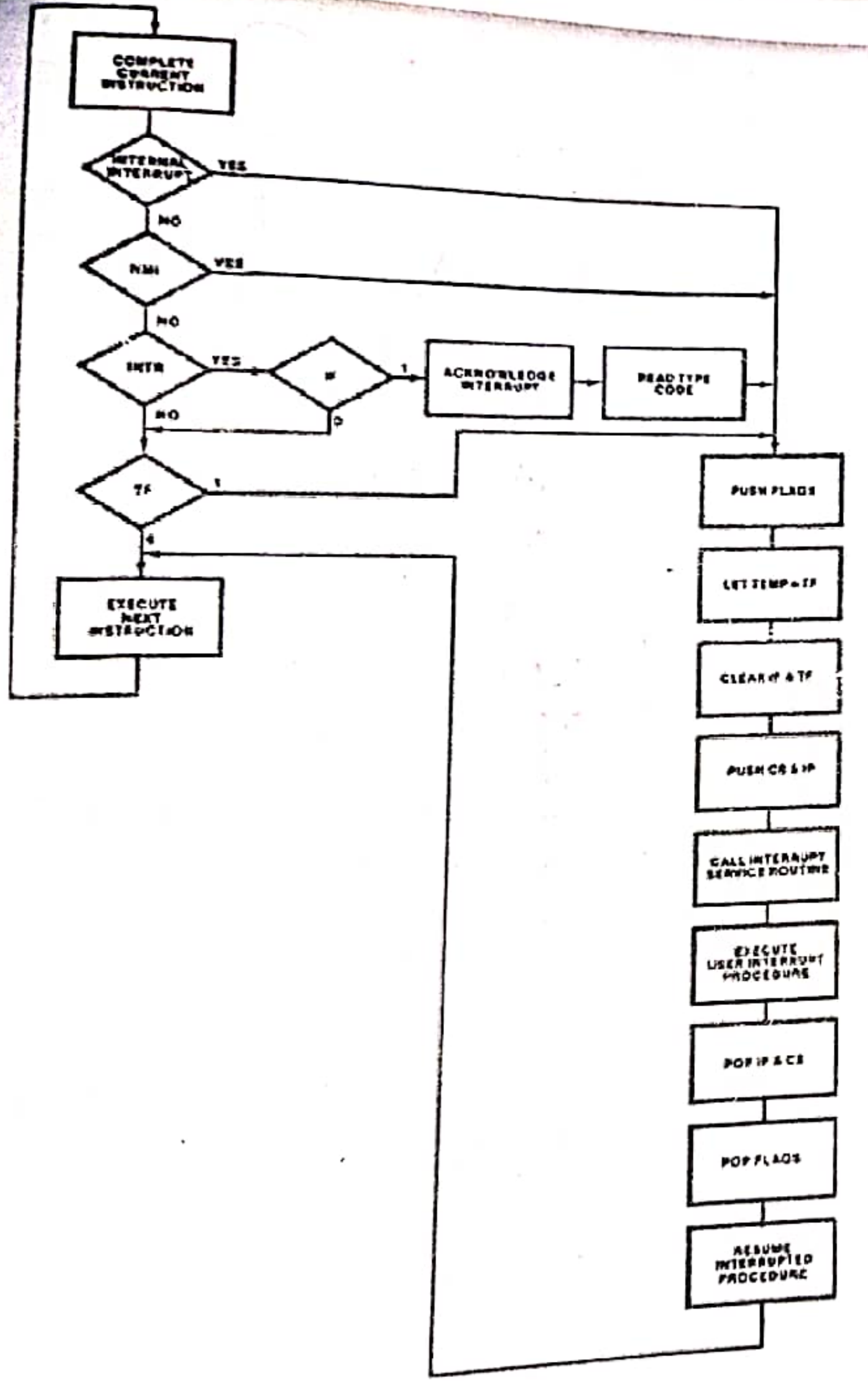
active 2 Intrpt acknoleye

Figure 11–9 Interrupt-acknowledge bus cycle (Reprinted by permission of Intel Corporation. Copyright/Intel Corp. 1979) microp send 2 acknowledge by using 2 different bus cycles



Interrupt Sequence

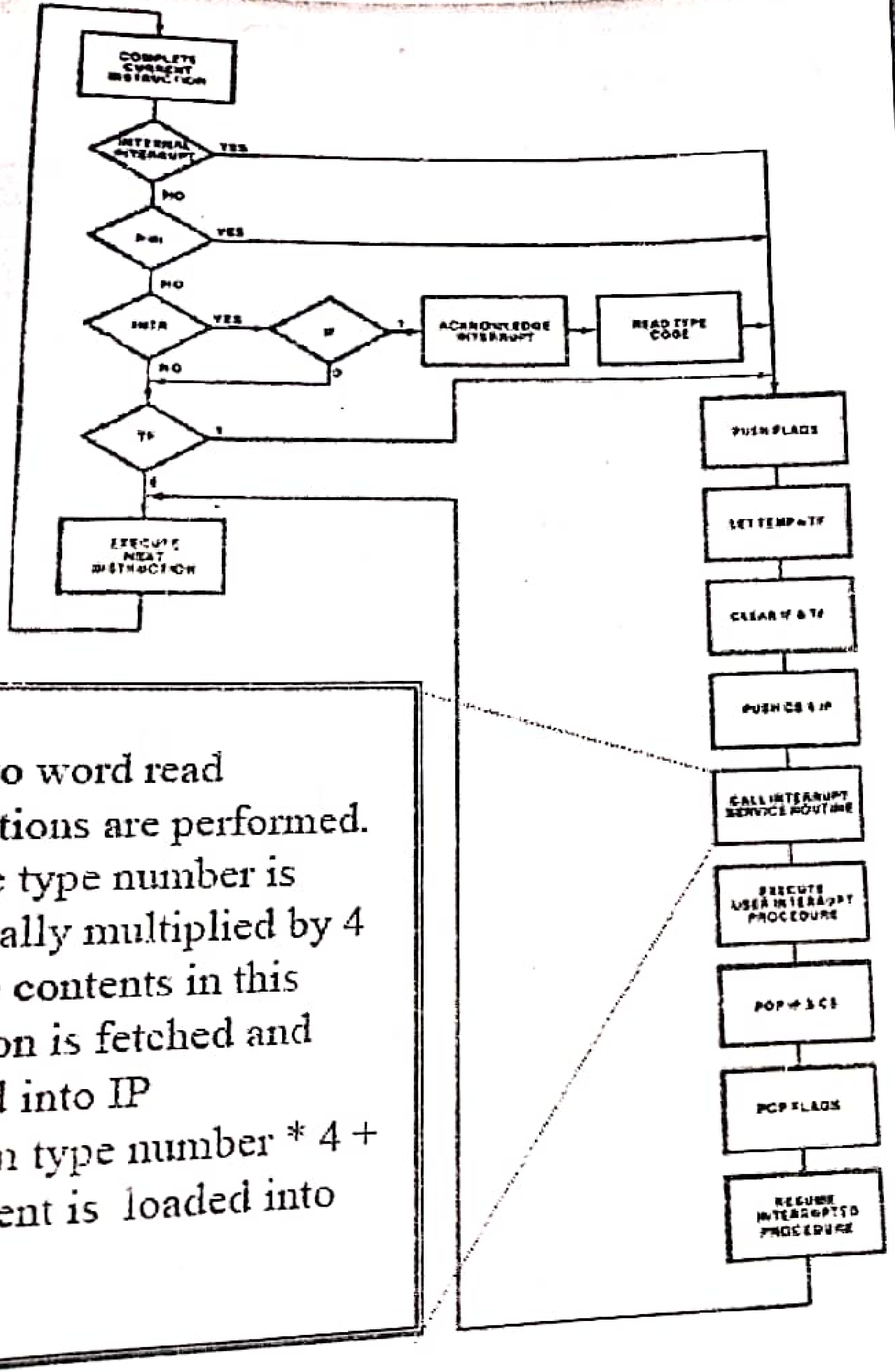
- The interrupt sequence begins when external device requests service by activating one of the interrupt inputs.
- The external device evaluates the priority of this interrupt
- INTR → 1
- 80x86 checks for the INTR at the last T state of the instruction
- Check for IF before granting INTA



Interrupt Sequence

- ➢80x86 initiates the INTA bus cycle. During T1 of the first bus cycle ALE is sent and bus is at Z state and stays high for the bus cycle.
- ➢LOCK is provided in maxmode operation
- ➢During the second interrupt acknowledge bus cycle, external circuitry gates one of the interrupts 20→FF onto data bus lines
- ➢Must be valid during T3 and T4 of second bus cycle

Interrupt Sequence



❖Two word read operations are performed.

❖The type number is internally multiplied by 4

❖The contents in this location is fetched and loaded into IP

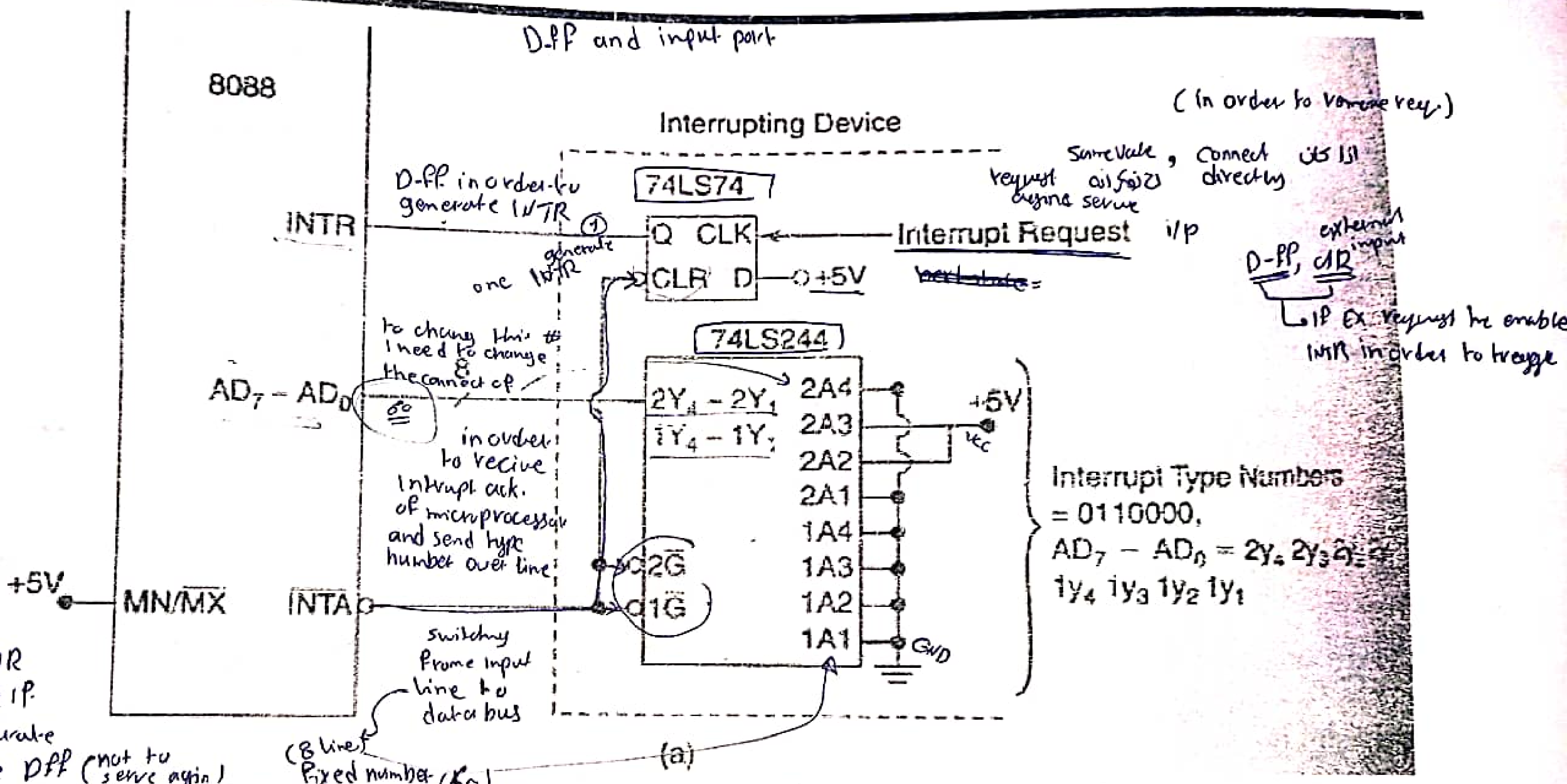
❖Then type number * 4 + 2 content is loaded into CS

- DT/R and DEN are at logic zero and IO/M is at 1.
- Next save the contents of the flag register
- TF and IF are cleared
- CS and IP are pushed

- Upon return by IRET
- CS and IP are popped
- Flags are popped

Interrupt Example

2 IC in order to Implement a simple external hardware


D.ff and input port
8088
Interrupting Device
74LS74
74LS244
INTR
AD7 – AD0
+5V
MN/MX
INTA
Q CLK
CLR D
+5V
Interrupt Request
2Y4 – 2Y1
1Y4 – 1Y1
2A4
2A3
2A2
2A1
1A4
1A3
1A2
1A1
2G
1G
+5V
GND
(a)
D-ff in order to generate INTR
① generate one INTR
to chang this # I need to change the connect of
60
in order to recive Intrupt ack. of microprocessor and send type number over line
switching frome input line to data bus
(8 line) fixed number (60)
(In order to ~~remove~~ req.)
Same Vale, connect اذا كان directly
request [illegible] agina serve
i/p
D-ff, CLR external input
IP Ex request he enable INTR in order to trage
Interrupt Type Numbers
= 0110000,
AD7 – AD0 = 2Y4 2Y3 2Y2 2Y1
1Y4 1Y3 1Y2 1Y1

① Q general INTR microp check the IP. IP equal to generate INTA and clk the Dff (not to serve agin) [illegible] and enable the input port in order to switch when microp. resend the Intrupt number, then ~~[illegible]~~ quite like XY to get address of IP and fitch new value then Jump to handler then return pop vale

Interrupts the microprocessor each time the interrupt request signal has a transition from 0→ 1. The corresponding interrupt number generated by the hardware in response to INTA is 60H

Interrupt Example

- An interrupting device interrupts the microprocessor each time the interrupt request input has a transition from 0 to 1.
- 74LS244 creates the interrupt type number 60H as a response to INTA
- Assume:
  - CS=DS=1000H   Mov Ax, 1000 Mov Ds, Ax
  - SS=4000H
  - Main program offset is 200H   mov Ax, 400
  - Count (counts the number of interrupts) offset is 100H
  - Interrupt-service routine code segment is 2000H   CS new
  - Interrupt-service routine code offset is 1000H   IP new
  - Stack has an offset of 500H to the current stack segment

IP → always the offset

store in Ds

60x4 ~ 180
أول 400
10 أول 180

Mov Ax, 0
Mov Ds, Ax ← نخليها zero
Mov [Ds:180] , 1000
Mov [Es, 182] 2000)

segment update عليه ع طول

enable Intrup
STI (IF=1)

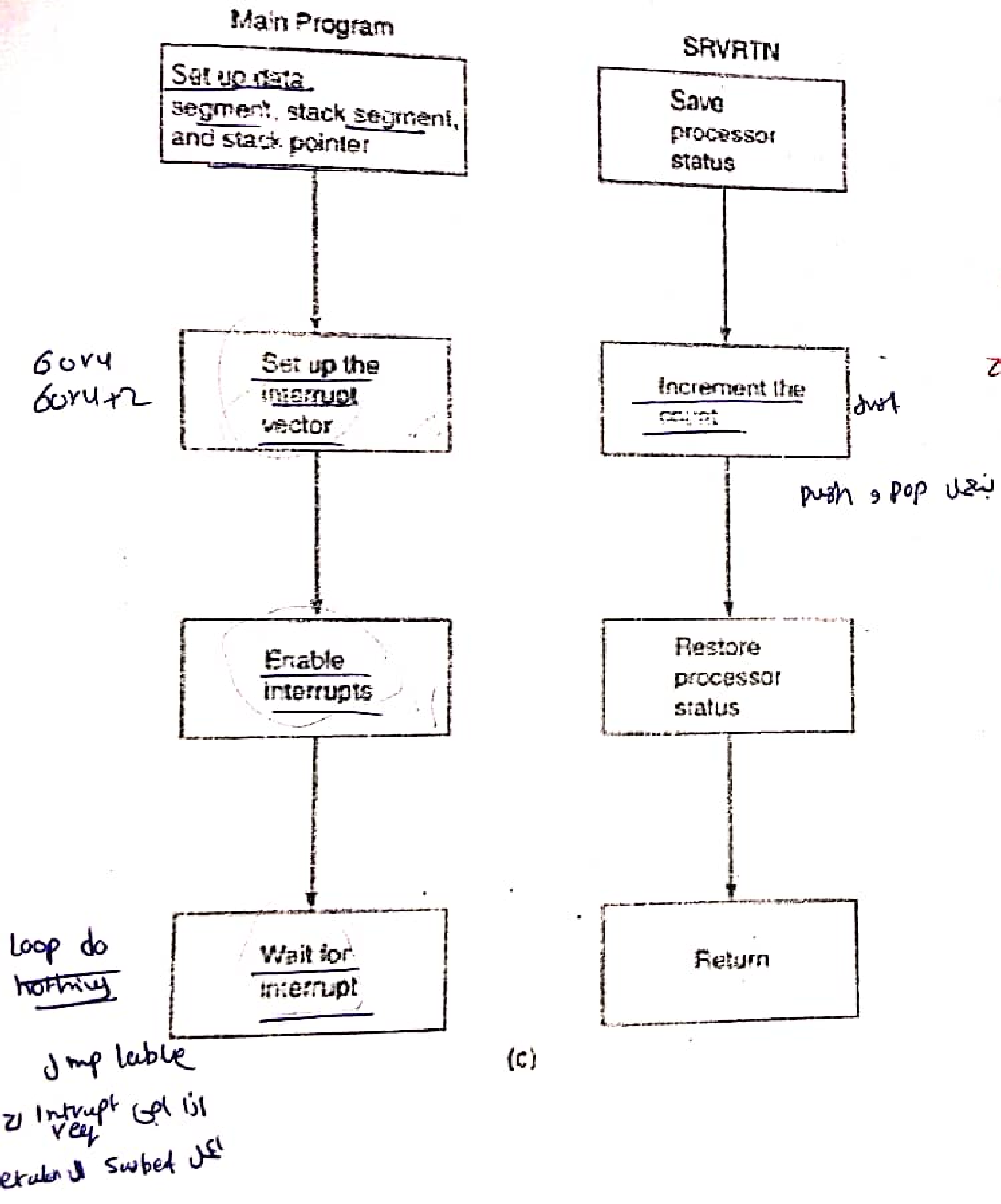
① – Make a map of the memory space organisation

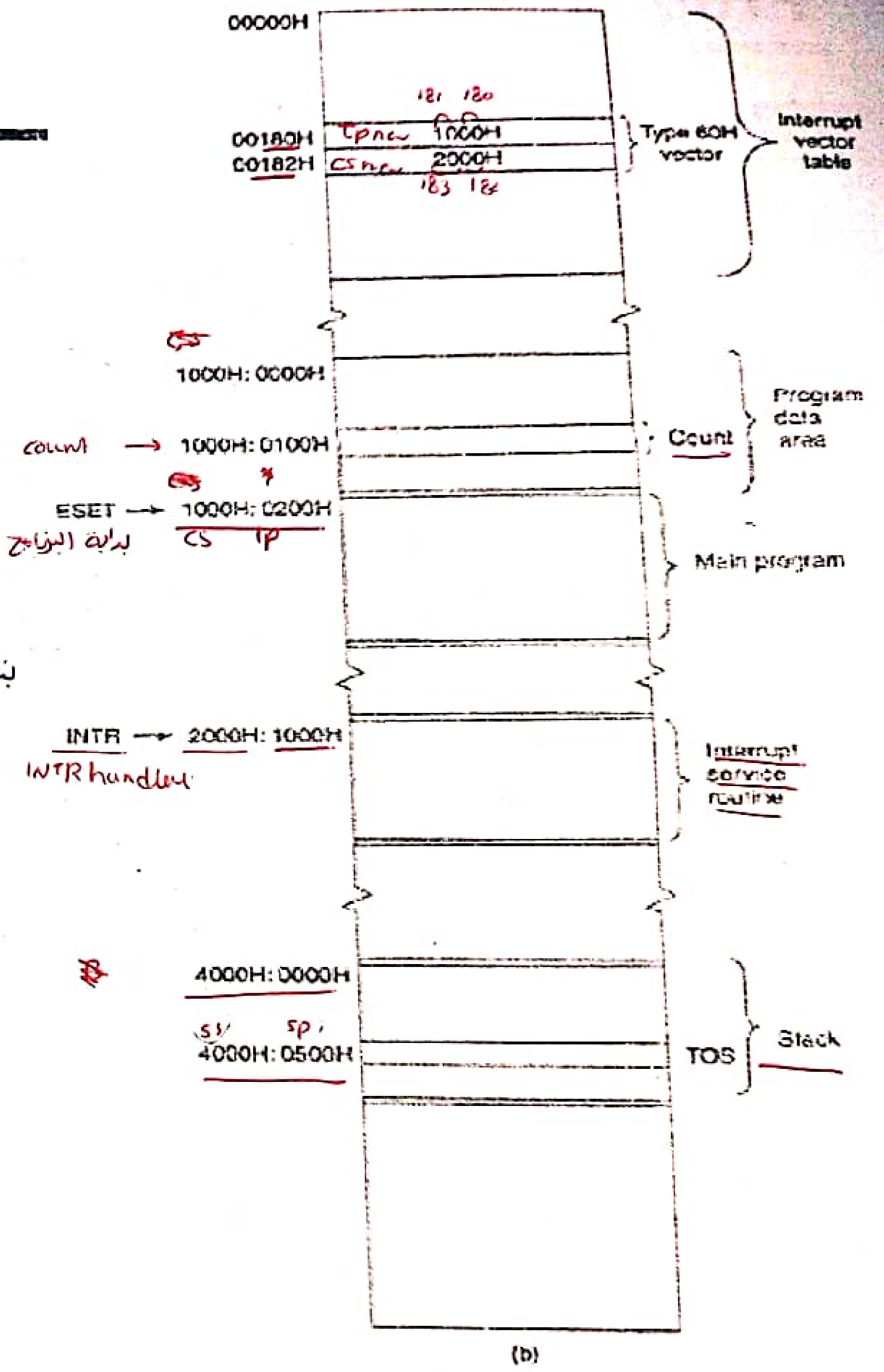
② – Write a main program and a service routine to count the number of positive interrupt transitions.

make memory map show all this?

صار ال clk
تبعت ال DFF
وال input port

Memory organization





Program

```
;Main Program, START = 1000H:0200H
START:          MOV  AX,1000H          ;Setup data segment at 1000H:0000H
                MOV  DS,AX
                MOV  AX,4000H          ;Setup stack segment at 4000H:0000H
                MOV  SS,AX
                MOV  SP,0500H          ;TOS is at 4000H;0500H
                MOV  AX,0000H          ;Segment for interrupt vector table
                MOV  ES,AX
                MOV  AX,1000H          ;Service routine offset
                MOV  [ES:180H],AX
                MOV  AX,2000H          ;Service routine segment
                MOV  [ES:182H],AX
                STI                    ;Enable interrupts
HERE:           JMP  HERE              ;Wait for interrupt

;Interrupt Service Routine, SRVRTN = 2000H:1000H
SRVRTN:         PUSH AX                ;Save register to be used
                MOV  AL,[0100H]        ;Get the count
                INC  AL                ;Increment the count
                DAA                    ;Decimal asdjust the count
                MOV  [0100H],AL        ;Save the updated count
                POP  AX                ;Restore the register used
                IRET                   ;Return from the interrupt
```

(d)

DS= 1000

SS = 4000

segmen عادي بالعكس مطلق مو

IVT بدايا

1 p new

CS new

بعد اعط بدون ما اكثر

[ES:182]AX, 2000

wait intrupt

unconditional jump (infit loop)

just increment count

INC [100];

بردو ممكن لعمل
بدون لا push
ولا pop.

يحول من decimal
الى hexa
وما يستعمل الا AL

8259 Programmable Interrupt Controller

Can be programmed Simult.ppl

- The 8259 programmable interrupt controller (PIC) adds eight vectored priority encoded interrupts to the microprocessor.
- This controller can be expanded to accept up to 64 interrupt requests. This requires a master 8259 and eight 8259 slaves.
- Vector an Interrupt request anywhere in the memory map.
- Resolve eight levels of interrupt priorities in a variety of modes, such as fully nested mode, automatic rotation mode, and specific rotation mode.
- Mask each of the interrupt request individually
- Read the status of the pending interrupts, in-service interrupts and masked interrupts.

① I can support ma
device

not like one
device with # (

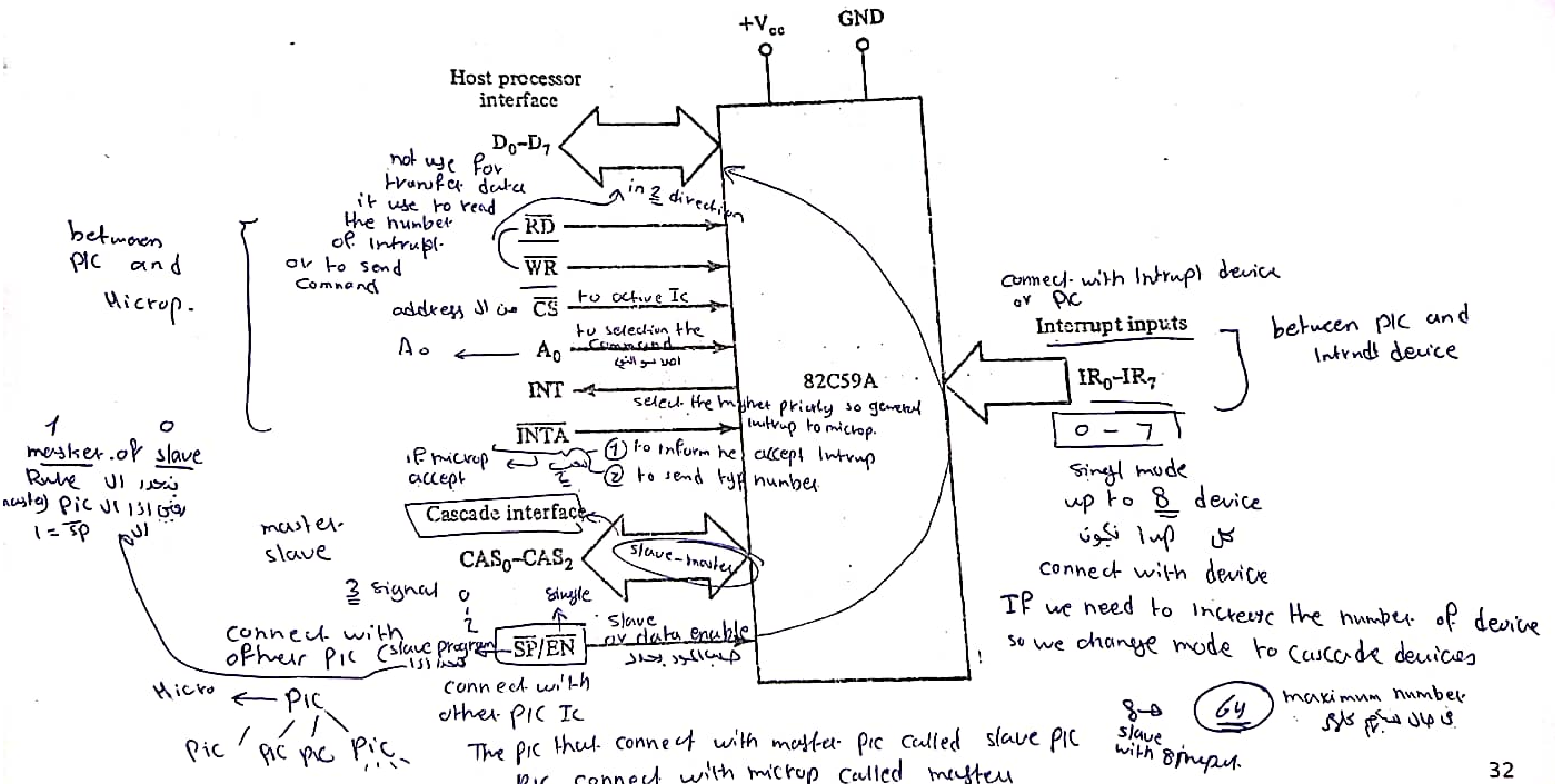
master - slave mode

8 - 64 mode

and I can change the Intrupt type number also I can chang the rule to select the priority.

RR [illegible], highest priority [illegible]

Block diagram of 82C59

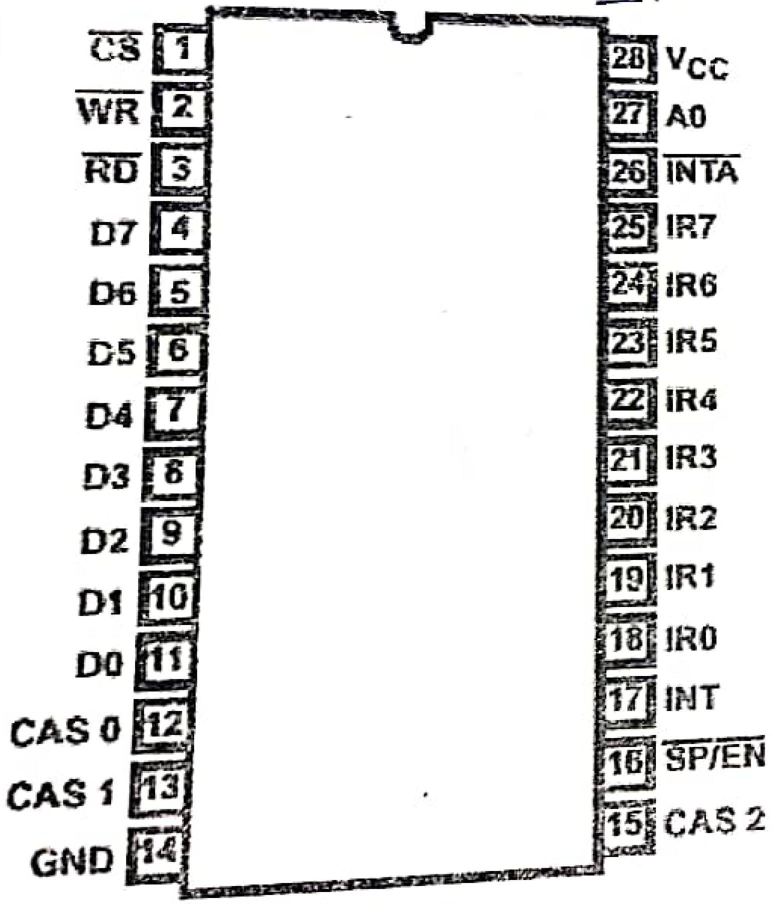


IF we need to increase the number of device so we change mode to cascade devices

The pic that connect with master pic called slave pic
pic connect with microp called master

Block Diagram

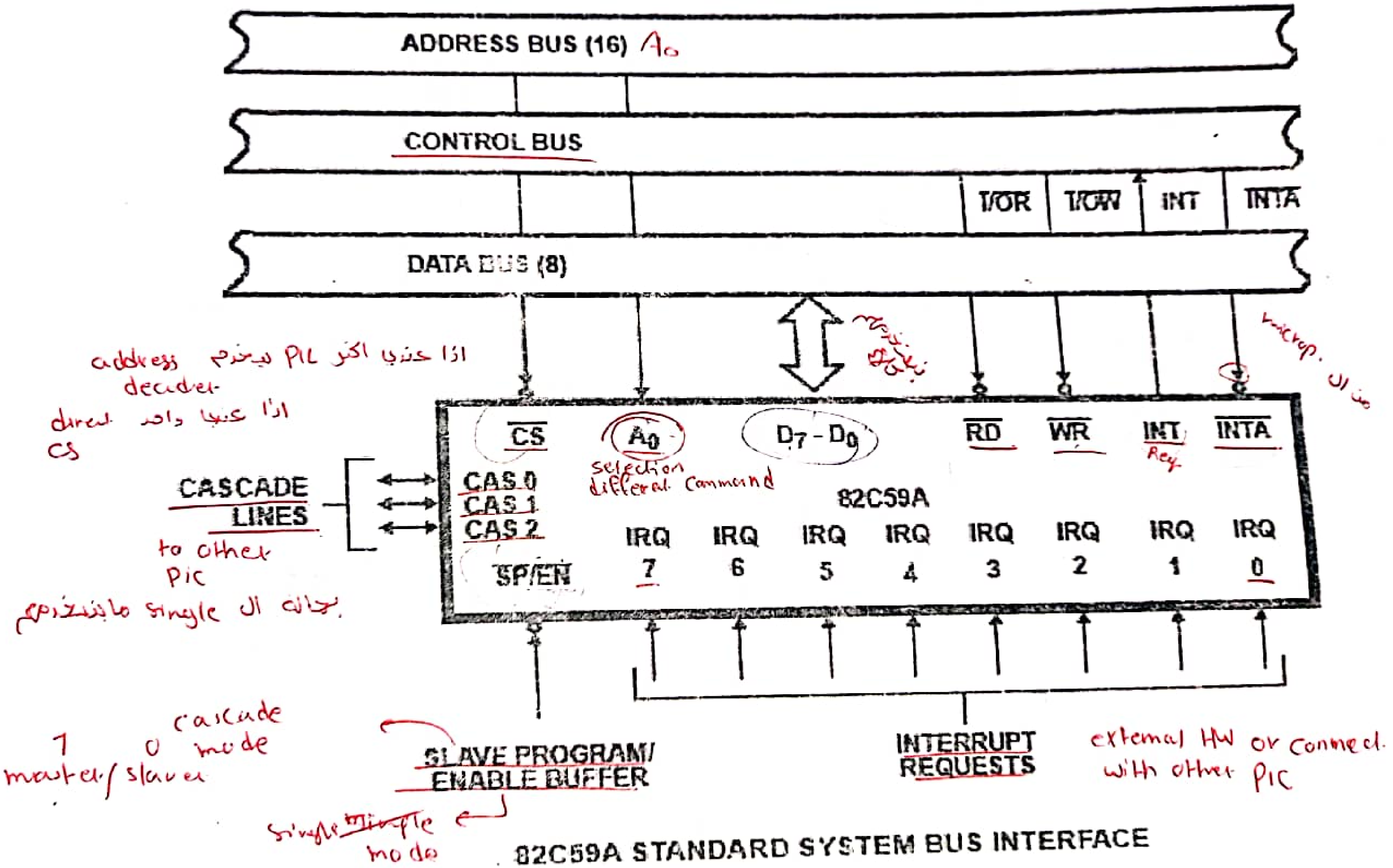
82C59A (PDIP, CERDIP, SOIC)
TOP VIEW 28 Pin



PIN	DESCRIPTION
D7 - D0	Data Bus (Bidirectional)
$\overline{RD}$	Read Input
$\overline{WR}$	Write input
A0	Command Select Address
$\overline{CS}$	Chip Select
CAS 2 - CAS 0	Cascade Lines
$\overline{SP}/\overline{EN}$	Slave Program Input Enable
INT	Interrupt Output
$\overline{INTA}$	Interrupt Acknowledge Input
IR0 - IR7	Interrupt Request Inputs

- I⁻ send Intrup type @ throug data bus

8259 System Bus



82C59A STANDARD SYSTEM BUS INTERFACE

82C59A Programmable Interrupt Controller

- Block diagram of 82C59A includes 8 blocks
  - 8259 is treated by the host processor as a peripheral device.
  - 8259 is configured by the host pocessor to select functions.
- **Data bus buffer and read-write logic**: are used to configure the internal registers of the chip.
  - A0 address selects different command words within the 8259

بحالة ال cascaded ال master connect with microp.

in 2

بس ال processor وحدة
slaver ← PIC

ال INT رح تكون ال master ويحل process

microp. ↑ master ↑ INT

رح يبعت ال winer لل cascaded
فتح input (win) → IRR cascad INT
ورح يبعت لل

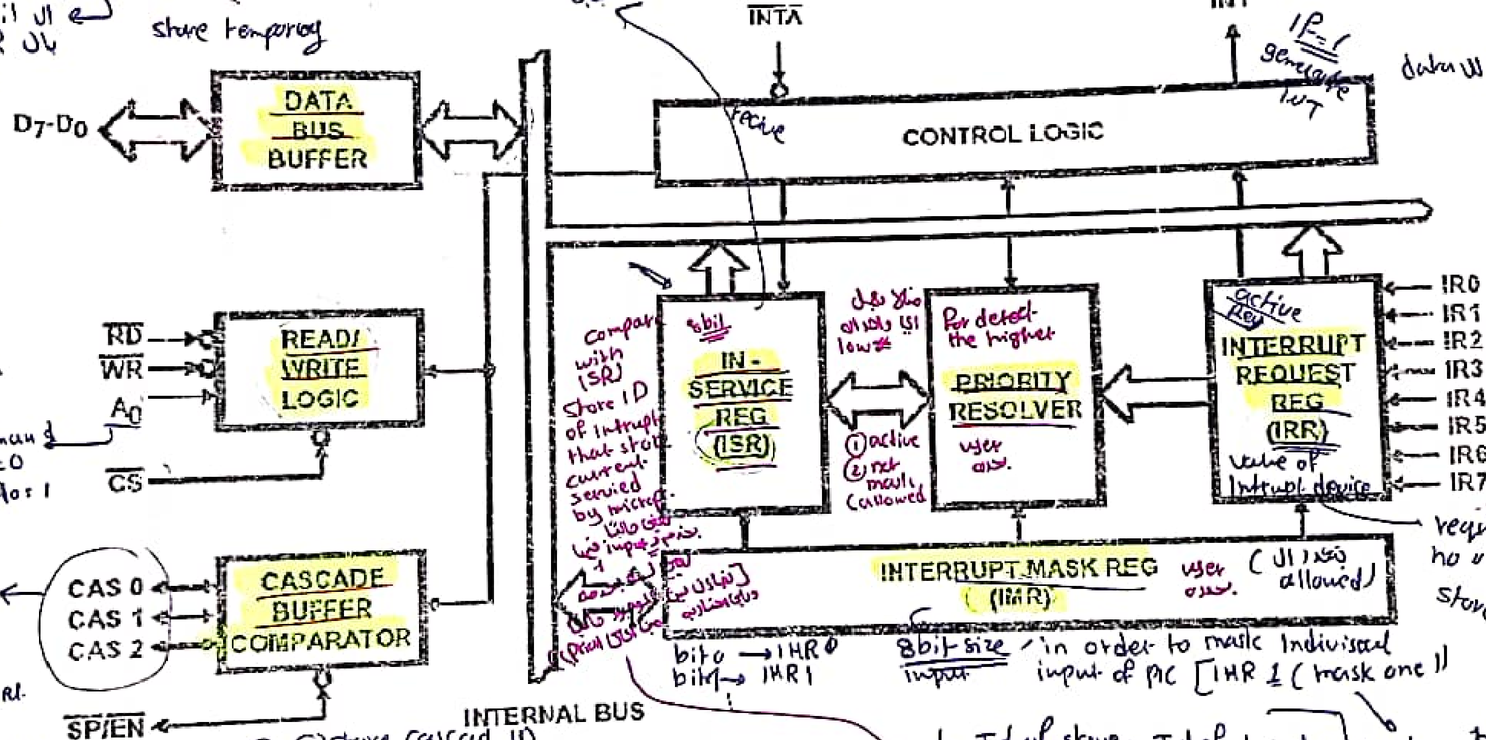
slave PIC رح يبعت ال number لل data
بعد ال akwoleg التانية

: automatic
Normal] type ال بيعمل remove بعد

when I recived second Intrup I allso remove the bit

ال bit الموجودة بال ISR ما رح تمحي
غير لما يدخل ال handler
eoi
end of Intrupt
لحتى ال pic يفعل
ISR bit ال remove
(لحتى ما يصير مشاكل
انه ما بحكي مين يخدم)

store temporary



connect control and address
some command has A0=0 an other A0=1

to connect and communicat. master with slaver
master want to connect with slaver 0 so I set the number zero

CAS2	CAS1	CAS0	
0	0	0	
0	0	1	1
0	1	0	2
:			:

بعملها
مقارنة بين ال ID الموجودة بال cascade buffer مع يلي طلبه ال master

② ① store cascad ID
② comparator (value send by master and send by cascade)

رح تنتقل من ال IRR لل ISR
↑ active INT
INTA رح يبعت اول واحدة
priority ما رح يقل نيا انا اختار ال اعلى
اما اذا ... اكيا رح يقل الاعلى

② لما يجي ال INTA التانية
send Intrup type # throw data bus

IR0 → PIC (slaver 0)
IR1 → PIC (slaver 1)
(data line)
بالكنك 2 رح تسحب D
بيرجع 0
(0C - F.) belog to it بيكون

request services 1
no " " 0 } for single mode
store intrupt input

in order to mask individual input of PIC [IMR 1 (mask one)]

Id of slave = Id of input
cascade buffer بخزن داخل ال

اذا 1= (cant intrupt the microp.)
اذا 0= (can intrupt microp)

82C59A Programmable Interrupt Controller

- Control Logic INT and INTA¯ ared used as the handshaking interface.
  - INT output connects to the INTR pin of the master and is connected to a master IR pin on a slave. INTA¯ is sent as a reply.
  - In a system with master and slaves, **only** the **master INTA¯ signal** is connected.

    بعرفوا عن طريق ال cascaded bus

و ممكن طريقه اخرى
انه INTA ثابت
مع الكل

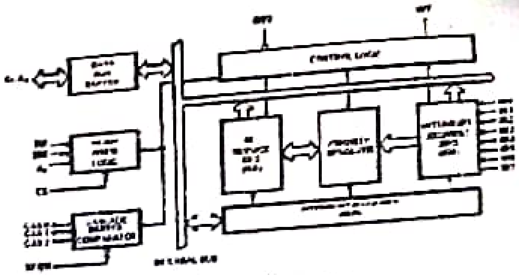
- Interrupt Registers and Priority Resolver: Interrupt inputs IR_0 to IR_7 can be configured as either *level-sensitive* or *edge-triggered* inputs. Edge-triggered inputs become active on 0 to 1 transitions.

1. **Interrupt request register (IRR)**: is used to indicate all interrupt levels requesting service.
2. **In service register (ISR)**: is used to store all interrupt levels which are currently being serviced.
3. **Interrupt mask register (IMR)**: is used to enable or mask out the individual interrupt inputs through bits M0 to M7. 0= enable, 1= masked out.
4. **Priority resolver**: This block determines the priorities of the bits set in the IRR. The highest priority is selected and strobed into the corresponding bit of the ISR during the INTA¯ sequence.
   - The priority resolver examines these 3 registers and determines whether INT should be sent to the MPU

82C59A Programmable Interrupt Controller

- **Cascade-buffer comparator:** Sends the address of the selected chip to the slaves in the master mode and decodes the status indicated by the master to find own address to respond.
  - Cascade interface CAS_0-CAS_2 and SP^{-}/EN^{-}:
    - Cascade interface CAS_0-CAS_2 carry the address of the slave to be serviced.
    - SP^{-}/EN^{-} :=1 selects the chip as the master in cascade mode
      :=0 selects the chip as the slave in cascade mode
      :in single mode it becomes the enable output for the data transiver

Interrupt Sequence



1) One or more of the INTERRUPT REQUEST lines (IR0 - IR7) are raised high, setting the corresponding IRR bit(s).
2) The 82C59A evaluates those requests in the priority resolver with the **IMR** and **ISR**, resolves the priority and sends an interrupt (INT) to the CPU, if appropriate.
3) The CPU acknowledges the INT and responds with first INTA pulse.
4) During this INTA pulse, the appropriate ISR bit is set and the corresponding bit in the IRR is reset (to remove request). The 82C59A does not drive the data bus during the first INTA pulse.
5) The 80C86/88/286 CPU will initiate a second INTA pulse. The 82C59A outputs the 8-bit pointer onto the data bus to be read by the CPU.
6) This completes the interrupt cycle. In the **Automatic End of Interrupt** (AEOI) mode, the ISR bit is reset at the end of the second INTA pulse. Otherwise, the ISR bit remains set until an appropriate End of Interrupt (EOI) command is issued at the end of the interrupt subroutine.

Fully Nested Mode (special Fully Nested mode)

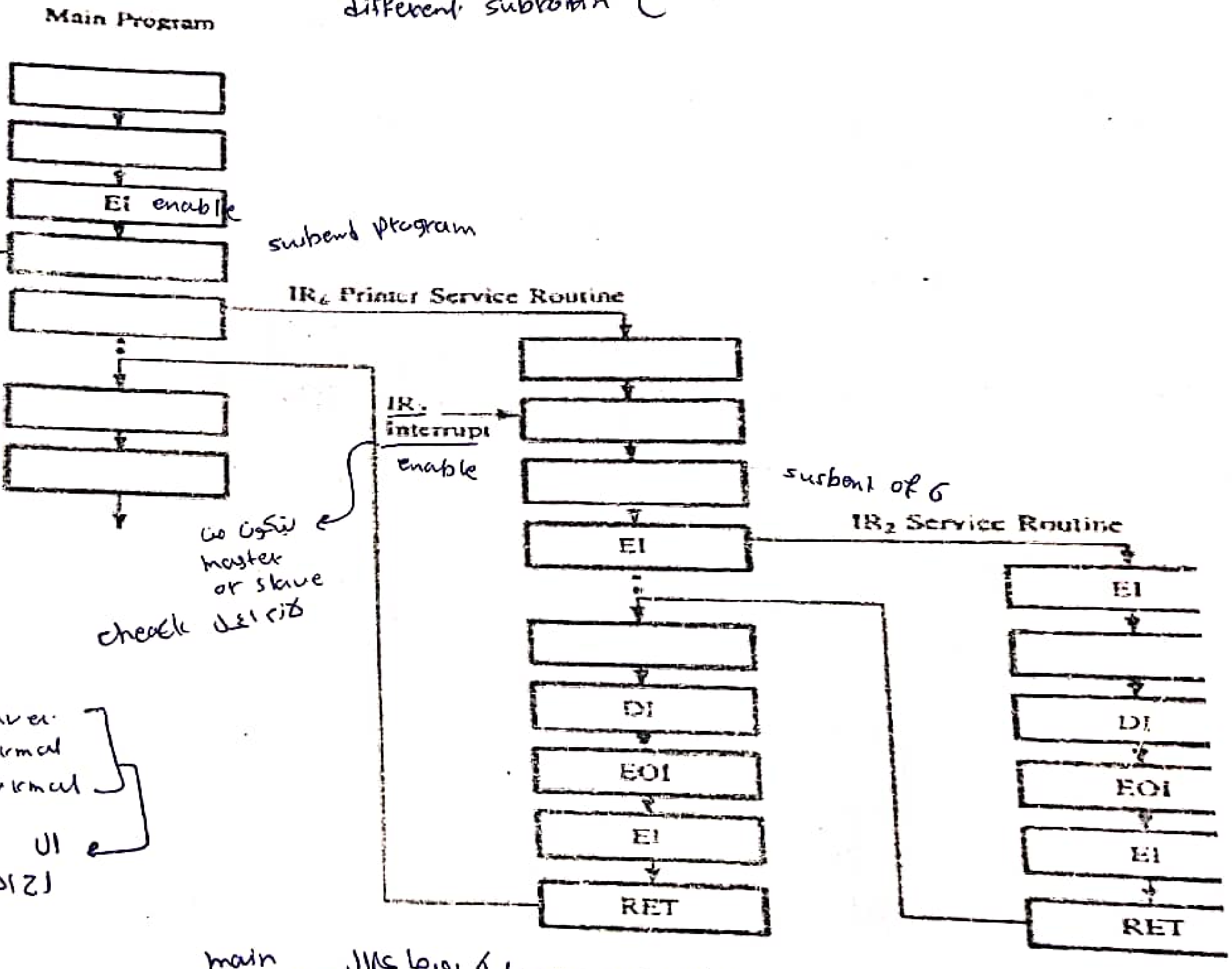
- It prioritizes the IR inputs such that IR0 has highest priority and IR7 has lowest priority
- This priority structure extends to interrupts currently in service as well as simultaneous interrupt requests
- For example, if an interrupt on IR3 is being serviced (IS3 = 1) and a request occurs on IR2, the controller will issue an interrupt request because IR2 has higher priority.
- But if an IR4 is received (or any interrupt higher than IR2), the controller will not issue the request
- Note however that the IR2 request will not be acknowledged unless the processor has set IF within the IR3 service routine
- In all operating modes, the IS bit corresponding to the active routine must be reset to allow other lower priority interrupts to be acknowledged
- This can be done by outputting **manually** a special nonspecific EOI instruction to the controller just before IRET
- Alternatively, the controller can be programmed to perform this nonspecific EOI **automatically** when the second INTA pulse occurs

Interrupt Process Fully Nested Mode

different subrotin (subend)

I execute the main program in same point I recived intrupt request

Interrupt IR6 allowed IF=1



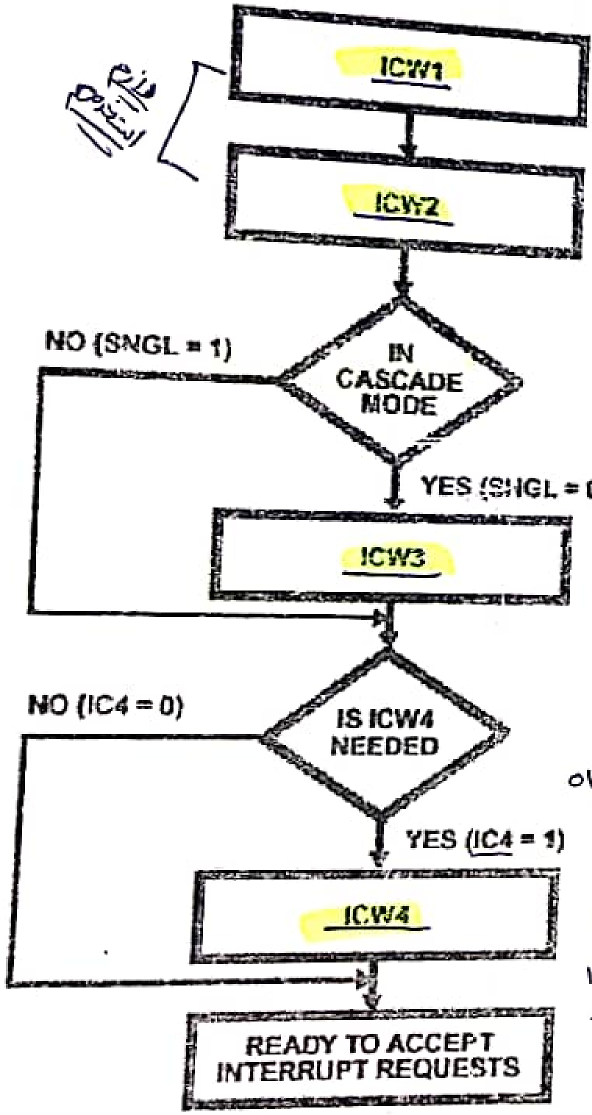
priority اعلى ← ① slaver
select Req. and Jump to execut.
2 normal
3 normal

special UI

اذا كان عندي slaver UI من input اكثر من
بضل ماشي بال slave ④ ← 4 من اكثر
privlyثاني ← 2
3

main program بنخلص من 2 بنروح لـ 6 بعدها عال

Initialization Sequence



لازم
ضروري

clear all Internal Reg
② select normal end of intrrup
and full
Comand
size 8 bit

دلالة ال
Cascaded
3 بحتاجه

اذا احتجناه
ولا لا
عن طريق وجود
bit معين
ICW 1
بتحدد اذا
بحتاجه او
لا

لازم يكونوا
in order

at least 2 command
cascaded 3-4

Two types of command words are provided to program the 8259:

1) The initialization command words (ICW) Initization PIC to Basic operation, when I start to use it

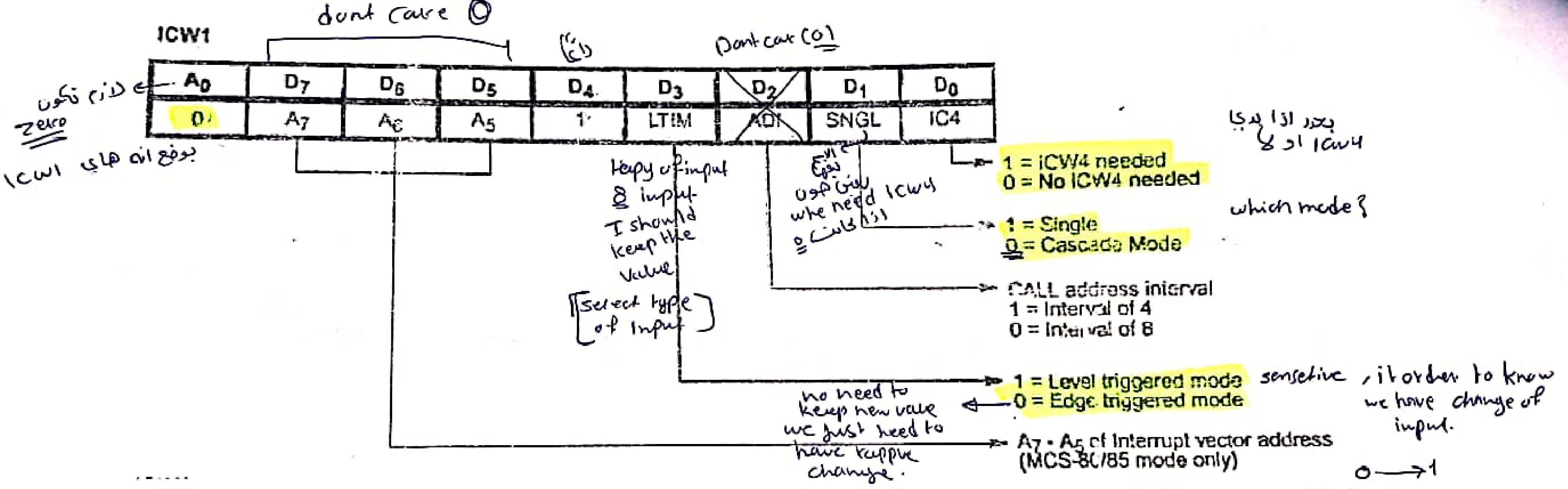
2) The operational command words (OCW) شرح ال PIC ← for more feature functionalty (more advance) more option

3 OCW
OCW
OCW

- Writing ICW1, clears ISR and IMR , IRR
- Also Special Masked mode SMM in OCW3, IRR in OCW3 and EOI in OCW2 are cleared to logic 0.
- Fully Nested Mode is entered.
- ICW3 and ICW4 are optional
- It is not possible to modify just one ICW. Whole ICW sequence must be repeated

ICW1

8 bit



ICW1

dont care (0)

(1)

Dont car (0)

A0	D7	D6	D5	D4	D3	D2	D1	D0
0	A7	A6	A5	1	LTIM	~~ADI~~	SNGL	IC4

لازم تكون zero

بوقع انه هاي ICW1

1 = ICW4 needed
0 = No ICW4 needed

بحدد اذا بدي ICW4 او لا

يعني بكون when need ICW4 اذا كانت 0

1 = Single
0 = Cascade Mode

which mode?

CALL address interval
1 = Interval of 4
0 = Interval of 8

keepy of input & input I should keep the value

[select type of input]

1 = Level triggered mode sensetive, it order to know we have change of input.
0 = Edge triggered mode

no need to keep new vale we just need to have rapple change.

0 → 1

A7 - A5 of Interrupt vector address
(MCS-80/85 mode only)

What value should be written to ICW1 in order to configure the 8259 so that ICW4 needed, the system is going to use multiple 8259s and its inputs are level sensitive?

$$\underset{x}{0}\underset{x}{0}\underset{x}{0}\underline{1}\underset{\text{level}}{1}\underset{x}{0}\underset{\text{Cascad}}{0}\underset{\text{use ICW1}}{1}b = 19h$$

ICW2

8 bit

select Range of Intrupt type number over Input of PIC

عاطينا بالنص ...

ICW2

(MCS-80/85 mode only)

A0	D7	D6	D5	D4	D3	D2	D1	D0
1	A_{15} / T_7	A_{14} / T_6	A_{13} / T_5	A_{12} / T_4	A_{11} / T_3	A_{10}	A_9	A_8

A_{15} - A_8 of interrupt vector address (MCS80/85 mode)

T_7 - T_3 of interrupt vector address (8086/8088 mode)

هذا حسب ال Rang ماي بدو اياها

مارح يغيروا 0 0 0 تبعوا حسب ال pik

What should be programmed into register ICW2 if type number output on the bus is to range from F0h to F7h

Intrupt 0 → بعطي F0
" 1 → بعطي F1

1 1 1 1 0 0 0 1 → F1

الباقي ال user سوين — intrup one

ICW2 بيستخدم

11110000b = F0h

Suppose IR6 is set to generate the value of 6E. Generate the addresses for the other interrupts.

(ICW2) بعطي input zero

Range

IR7 = 6F	IR3 = 6B
IR6 = 6E	IR2 = 6A
IR5 = 6D	IR1 = 69
IR4 = 6C	IR0 = 68

11011110 (6E)

1 1 0 1 1 1 0 ICW2

1 1 0 1 0 0 0 → 68

لانه ال input لحد 6F وضمن IR6

6F · 0 1 0 1 1 1 1

0 1101 1 1 0

01 1 0 1 0 0 0

68

6F

1 6F - 68

Content of the Interrupt Vector Byte

CONTENT OF INTERRUPT VECTOR BYTE FOR
80C86/88/286 SYSTEM MODE

اعطي شو بدك

حسب inpu

	D7	D6	D5	D4	D3	D2	D1	D0
IR7	T7	T6	T5	T4	T3	1	1	1
IR6	T7	T6	T5	T4	T3	1	1	0
IR5	T7	T6	T5	T4	T3	1	0	1
IR4	T7	T6	T5	T4	T3	1	0	0
IR3	T7	T6	T5	T4	T3	0	1	1
IR2	T7	T6	T5	T4	T3	0	1	0
IR1	T7	T6	T5	T4	T3	0	0	1
IR0	T7	T6	T5	T4	T3	0	0	0

ICW3

use only in cascade mode

2 Version

يحدد اذا ال master اذا connect with slave (1) not connect (0)

ICW3 (MASTER DEVICE)

A0	D7	D6	D5	D4	D3	D2	D1	D0
1	S7	S6	S5	S4	S3	S2	S1	S0

1 = IR input has a slave

0 = IR input does not have a slave or not connect.

ICW3 (SLAVE DEVICE)

A0	D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	0	0	ID2	ID1	ID0

الباقي كلهم 0

determain slave ID for slave PIC or Cascade

SLAVE ID (NOTE)

0	1	2	3	4	5	6	7
0	1	0	1	0	1	0	1
0	0	1	1	0	0	1	1
0	0	0	0	1	1	1	1

هي نفس رقم ال input
منوك بس S لا
ال master لازم
اعمل رقم S

لانه بكون up to 8

1 → 001
2 → 010
7 → 111

Q) Suppose we have two slaves connected to a master using IR0 and IR1.

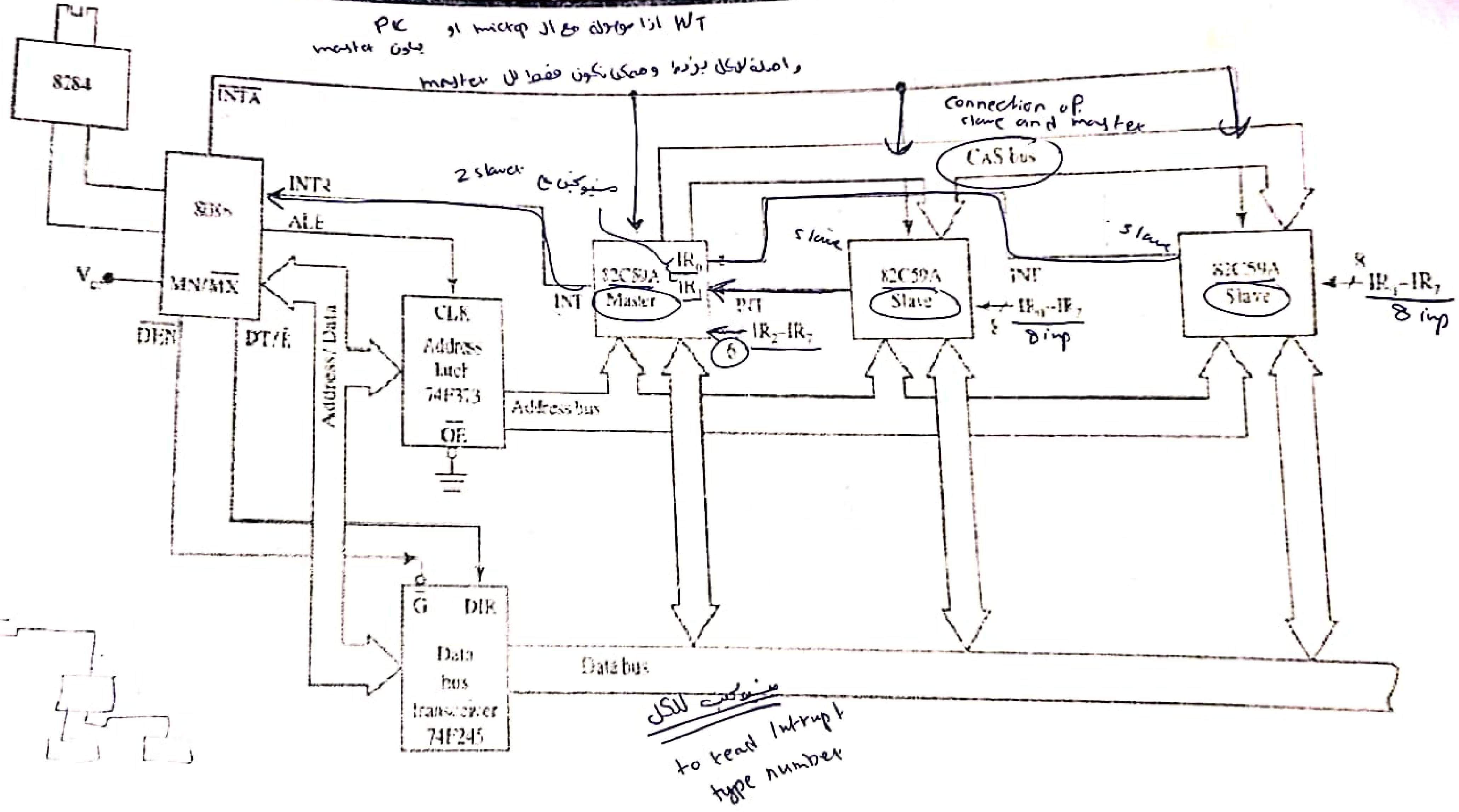
A0	D7	D6	D5	D4	D3	D2	D1	D0	
1	0	0	0	0	0	0	1	1	→ $\boxed{03}$

A) The master is programmed with an ICW3 of 03h, one slave is programmed with an ICW3 of 00h and the other with an ICW3 of 01h.

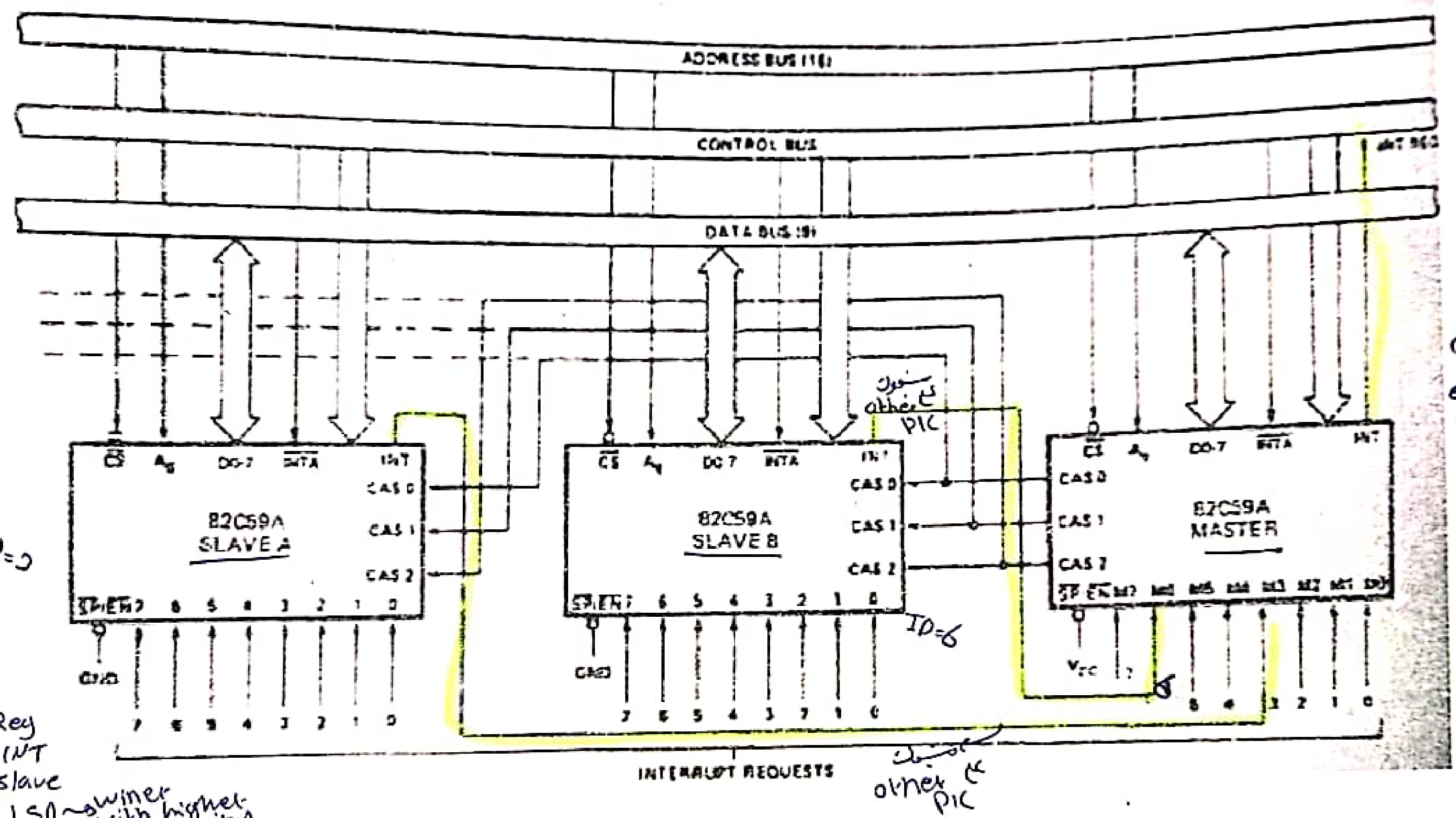
ICW3 for slave 0

D7	D6	D5	D4	D3	D2	D1	D0	
0	0	0	0	0	0	0	0	IR0 (00h)
0	0	0	0	0	0	0	1	IR1 (01h)

Master Slave Configuration



Master Slave Configuration



① slave process of Input Reg and gener INT
· For every slave
IRR, IMR, ISR ~ winer with higher priority
active alowed

✓When slave signals the master that an interrupt is active the master determines whether or not its priority is higher than that of any already active interrupt.

متلاقى من 2
010

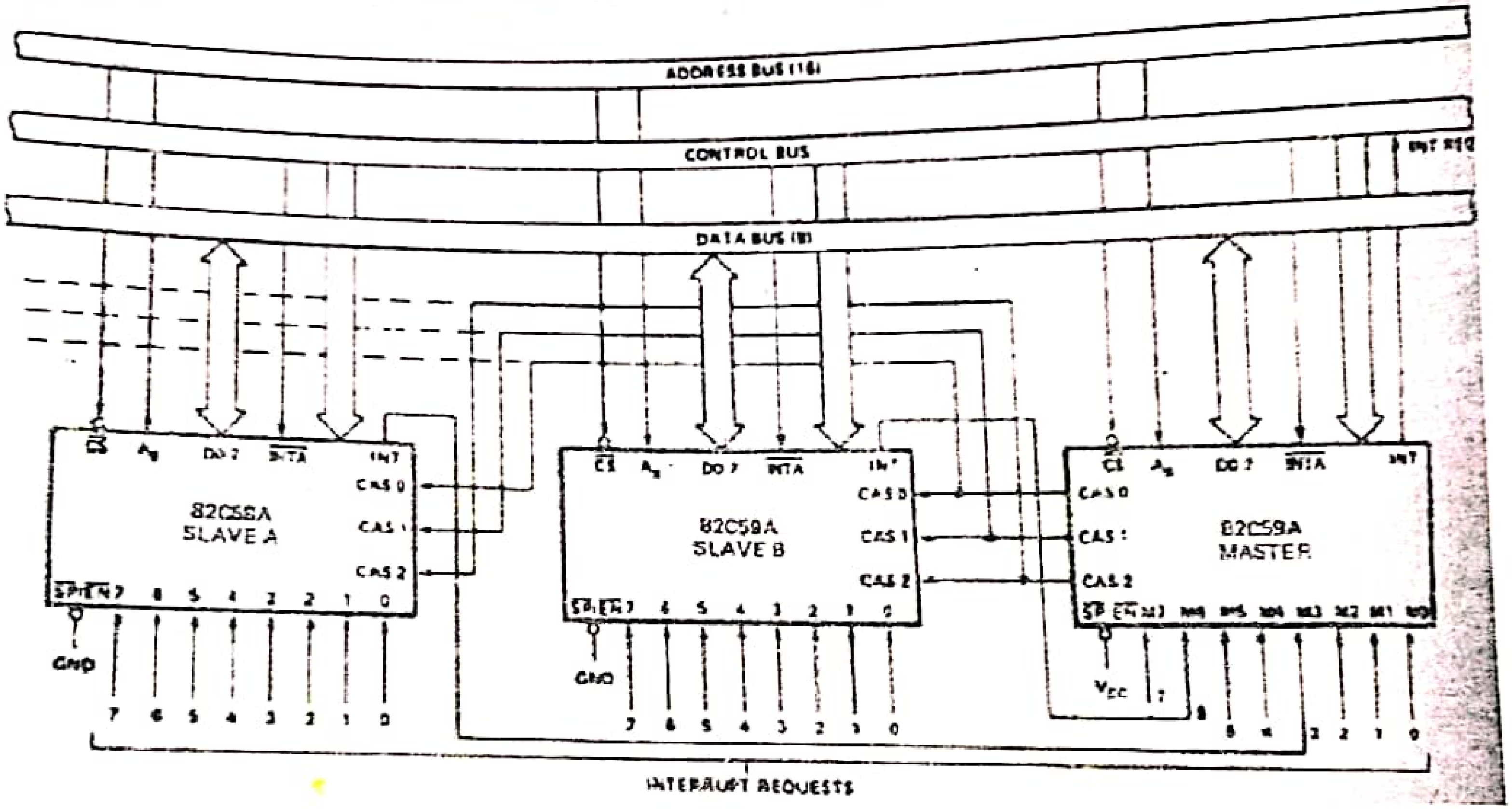
② master process قبل slave من other و IRR, ISR, IMR والـ winer

✓If the new interrupt is of higher priority the master controller switches INTR to logic 1

INTA
قبل generat اذا accep INT
② Inform accepet Reg.
③ In order to request Interrupt number
اذا الـ winer من slave
الـ PIC يعرف مين ع slave
مارح يعرف من اي input فـ لازم
الـ master يبعت ع cascade bus لل
Interupt type number ع malash
يبعت
ICW2
عن طريق

47

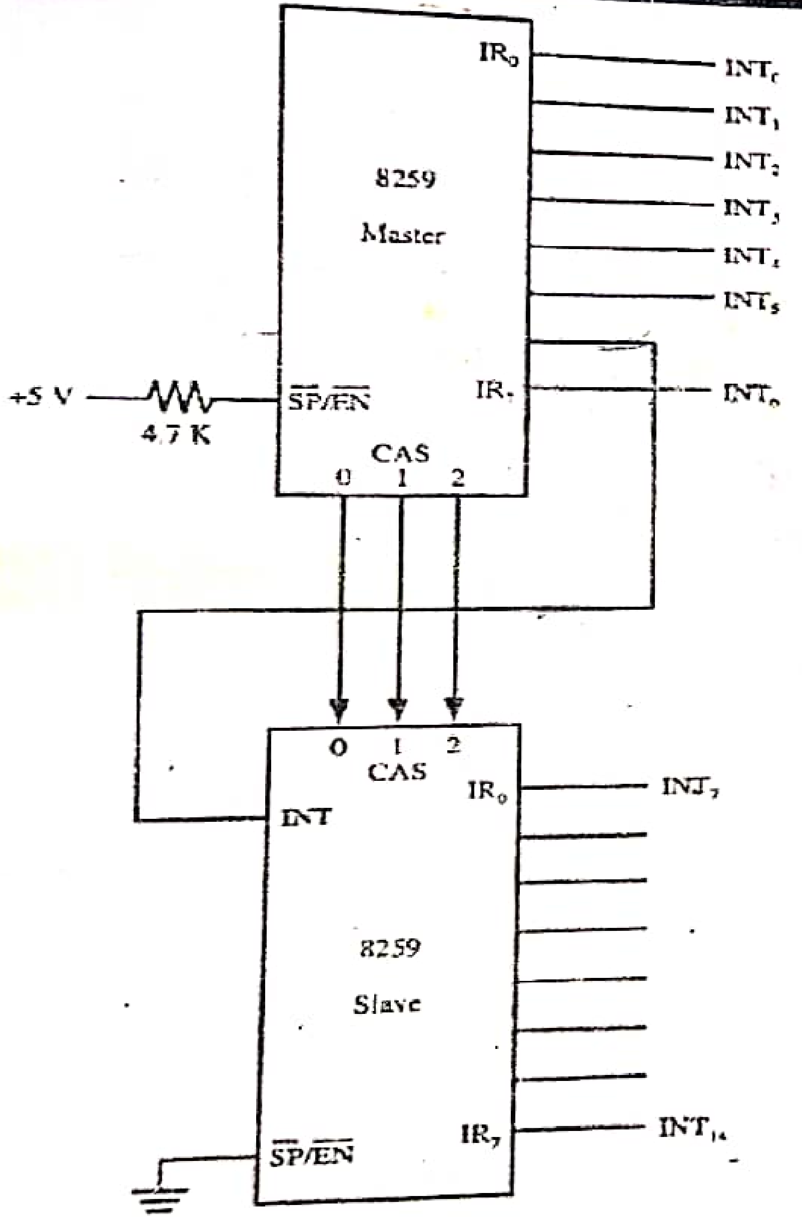
Master Slave Configuration



✓This signals MPU that external device needs to be serviced. If IF is set. As the first INTA is sent out the master is signaled to output the 3 bit cascade code of the slave device whose whose interrupt request is being acknowledged on the CAS bus. All slaves read this code and compare internally

✓The slave corresponding to the code is signaled to output the type number of its highest priority active interrupt on the data bus during the second INTA cycle.

Example Master-Slave



✔ Any requests on interrupt lines INT7 through INT14 will cause IR6 to be activated on the MASTER.

✔ The MASTER will then examine the bit 6 in its ICW3 to see if it is set.

✔ If so it will output the cascade number of the SLAVE on CAS0 through CAS2.

✔ These cascade bits are received by the SLAVE device which examines its ICW3 to see if there is a match..

✔ The programmer must have programmed 110 into the SLAVE'S ICW3. If there is a match between the cascade number and ICW3, the SLAVE device will output the appropriate vector number during the second INTA pulse.

ICW4

8 bit

ICW4

0 always

A0	D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	SFNM	BUF	M/S	AEOI	μPM

μPM → Type of microprocer

1 لل → 1= 8086/8088 mode
0 = MCS-80/85 mode

AEOI → 1 = Auto EOI ~~normal~~
0 = Normal EOI → بلش لحد ما يخلص handler ال PIC

BUF, M/S: تعين 26

non buffer

BUF	M/S	
0	X	- Non buffered mode
1	0	- Buffered mode slave
1	1	- Buffered mode master

Buffer

يحدد اذا slave or master

sing ED
Cascadet

SFNM → 1 = Special fully nested moded
0 = Not special fully nested mode

normal

AEOI mode requires no commands. During the second INTA the ISR bit is reset. The major drawback with this mode is that the ISR doesn't have info on which IR is served. Thus any IR with any priority can *now* Interrupt service routine.

BUF when 1 selects buffer mode. The SP/EN pin becomes an output for the data buffers.
When 0, the SP/EN pin becomes the input for the (MASTER/SLAVE) functionality

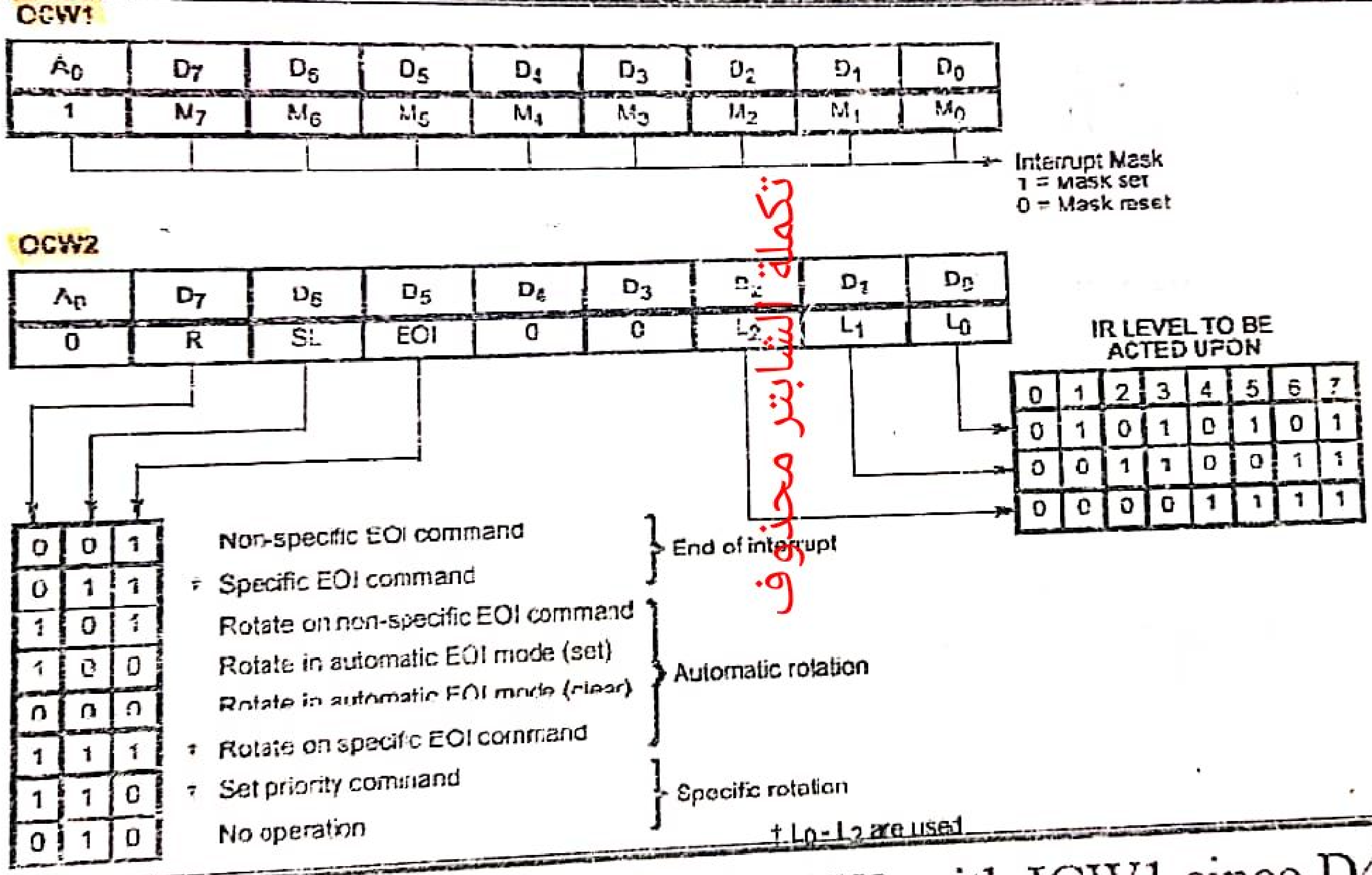
M/S is used to set the function of the 8259 when operated in buffered mode
If it is set the 8259 will function as the MASTER.
If cleared will function as SLAVE.

OCW1 - OCW2

more option or features

OCW1 is used to access the contents of the IMR. A READ operation can be performed to the IMR to determine the present setting of the mask. Write operations can be performed to mask or unmask certain bits.

مافي داعي
يكونوا
in order
اذا بدون واحد
ما بيوقف
الكل



OCW1

A0	D7	D6	D5	D4	D3	D2	D1	D0
1	M7	M6	M5	M4	M3	M2	M1	M0

Interrupt Mask
1 = Mask set
0 = Mask reset

OCW2

A0	D7	D6	D5	D4	D3	D2	D1	D0
0	R	SL	EOI	0	0	L2	L1	L0

R	SL	EOI		
0	0	1	Non-specific EOI command	End of interrupt
0	1	1	† Specific EOI command	
1	0	1	Rotate on non-specific EOI command	Automatic rotation
1	0	0	Rotate in automatic EOI mode (set)	
0	0	0	Rotate in automatic EOI mode (clear)	
1	1	1	† Rotate on specific EOI command	Specific rotation
1	1	0	† Set priority command	
0	1	0	No operation	

† L0 - L2 are used

IR LEVEL TO BE ACTED UPON

0	1	2	3	4	5	6	7
0	1	0	1	0	1	0	1
0	0	1	1	0	0	1	1
0	0	0	0	1	1	1	1

Controller will not confuse OCW2 with ICW1 since D4 = 1