Palestine Polytechnic University

College of Engineering and Technology
Electrical and Computer Engineering Department

Graduation Project

Experimental Interfacing Boards for 8085 Microprocessor

Project Team
AbedFattah Ahmad Najjar
Ra'ed Azmi Tailakh
Tahanee Al-darabe

Project Supervisor
Eng: Elayn Abu Gharbyeh

Hebron – Palestine

January, 2005

ABSTRACT

A programmable interface device is designed to perform various input/output functions. Such a device can be set up to perform specific functions by writing an instruction. This project introduces four of these devices; that is 8284, 8255, 8259, and 8251. Explains the theory of each chip then provide many Suggested experiments for each chip with 8085 mp.

Therefore, the laboratory becomes better and better since our project mainly has tow advantages the first is using 8085KIT in educational way the second it is the first step in building interfacing manual for the MIC LAB.

إن القطع المبرمجة قد وجدت لتؤدي عدة وظائف من الإدخال و الإخراج عند ربطها مع المعالج وهذه القطع تؤدي هذه الوظائف المختلفة تبعا للبرامج التي تكتب و تبرمج بها القطع.هذا المشروع يقدم أربع قطع مبرمجة و هي (PPI,PIT,PIC,USART) بحيث يشرح تركيب ووظائف كل قطعة و من ثم بناء صندوق تجارب لربط هذه القطع مع المعالج 8085 و بالتالي إتاحة الفرصة أمام الطالب لعمل تجارب ربط الأنظمة المتعلقة بهذه القطع بكل سهولة و يكون الربط مع لوحة المعالج التعليمية الموجودة في المختبر حيث يؤدي ربط صناديق التجارب معها إلى استخدامها بشكل تعليمي مميز و بناء التطبيقات اللازمة من التجارب عليها و أخيرا الخروج بكتاب تجارب للوحة 8085 الموجودة في مختبر المعالج الدقيق و ربط الأنظمة .

Table of Contents

List of Tables

List of Figure

CHAPTER 1

INTRODUCTION

Introduction

1-1 General Overview:

The Microprocessor Laboratory is at the center of the practical learning experience in Computer Engineering. This laboratory is designed to give students "hands-on" experience with Microprocessor hardware and software design concepts and applications and provides students with the opportunity to investigate the architecture of microprocessors and their associated systems.

With these subjects we want to build circuits for it, students learn to write, debug and run 8085 assembly language programs. The programs provide students with hands-on experience in topics of interface with I/O such as Programmable Peripheral Interface (PPI), Programmable Interrupt Controller (PIC), Programmable Interval Timer (PIT) and Universal Synchronous Asynchronous Receiver Transmitter (USART). In addition, students use the laboratory to design and implement their individual class projects, supervised by a faculty. The laboratory offers a rich design experience and prepares students for facing real world design challenges.

In the laboratory, there are six 8085 MP kits, which are used to connect to the related equipment of the Lab.

Conclusion, our laboratory is in progress. Besides of adding advance equipment, we develop the 8085MP kits by adding some boards. Therefore, the laboratory becomes better and better and our project mainly has tow advantages the first is using 8085KIT in educational way the second it is the first step in building interfacing manual for the 8085 KIT.

1-2 Project requirements

1-2-1 Hardware requirements:

1) Micro Processor 8085 (MC Vol-1 and MC Vol. -2).
2) The following programmable I/O interface devices are needed: -
    a. (8055A) PPI: Programmable Peripheral Interrupt
    b. (8059A) PIC: Programmable Interrupt Controller
    c. (8053A) PIT: Programmable Interval Timer.
    d. (8051A) USART: Universal Synchronous Asynchronous Receiver Transmitter
3) –Printed boards to build the interface circuit on it.
4) – Breadboards, Wires, cables and other equipments from library to build the designs.
5) Pentium computer
6) Printer

1-2-2 Software requirements:

1) Microprocessor assembly
2) Circuit Maker2000
3) Orcad 7.1
4) Windows XP
5) Office XP

1-3 Cost Estimation for hardware

IC	Price(American $)	#of components
Programmable Peripheral Interface PPI (8255)	$4	5
Programmable Interval Timer 8254	$6	5
Programmable Interrupt Controller PIC (8259)	$2	5
USART 8251	$5	5
Printed board	$8	5
Bread Board	$6	10
50 BIN connectors	$9	10
Wires	$10	
Latch	$2	5
Decoder	$3	5
Buffer	$3	5
NAND Gates	$3	5

Table 1-1 Cost Estimation

1-4 Time scheduling

For our project we have planed the following time estimation schedule, which includes the tasks related to studying and analysis the system.

Task//Week	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Collection of data																
Study 8085 kit																
Study interfacing devices																
Design																
Testing and Implementation																
Documentation																

Table 1-2: Time Schedule.

1-5 Report Contents

This report is divided into six chapters, the first chapter is the introduction, and it outlines general preview, literature review, project requirements cost estimation, time scheduling, and report content.

The second chapter is the theoretical background; it includes theoretical subjects related to the main ideas of the project, and information about the 8085Kit microprocessor and programmable interfacing devices(USART.PIC.DMA, PPI, TIMER).

CHAPTER 2

THEORITACAL BACKGROUNED

THEORITACAL BACKGROUNED

2.1 Preface

The following chapter includes the theoretical background for the system components: 8085 kit, PPI, PIC, USART, and PIT. Theses tutorial should be understood by the student before designing an interface circuits for 8085 mp.

2-2 8085kit

The micro processor 8085 (MC Vol1and2) has the following specifications: -

Characteristics		Specifications
(1)	CPU	8085 (4.7 MHz Crystal Oscillator)
(2)	RAM	62526X2 EA (64KB)
(3)	ROM	27256X2 EA (64KB)
(4)	User Memory	27256X2 EA or 62256X2 EA (64KB)
(5)	ADC0809	A/D converter (8bit X 8 channel)
(6)	DAC0808	D/A converter (8bit X 1 channel)
(7)	KEY	24 Keyboards
(8)	Display	LCD (16X2line)
(9)	SOUND	2W speaker
(10)	BUTTON	for input
(11)	LED	for output (Digital Display)
(12)	FND	for output (Numerical Display)
(13)	Connectors	
	a- parallel port	2EA
	b- Serial Port	2EA
	c- Expansion Port	1EA
	d- INTR	1EA

	c-	Expansion Port	1EA
	d-	INTR	1EA
(14)	Power supply		input AC 85~269V, Output: 5V 2A,-12V 0.3A

Table 2.1: 8085Kit Specifications

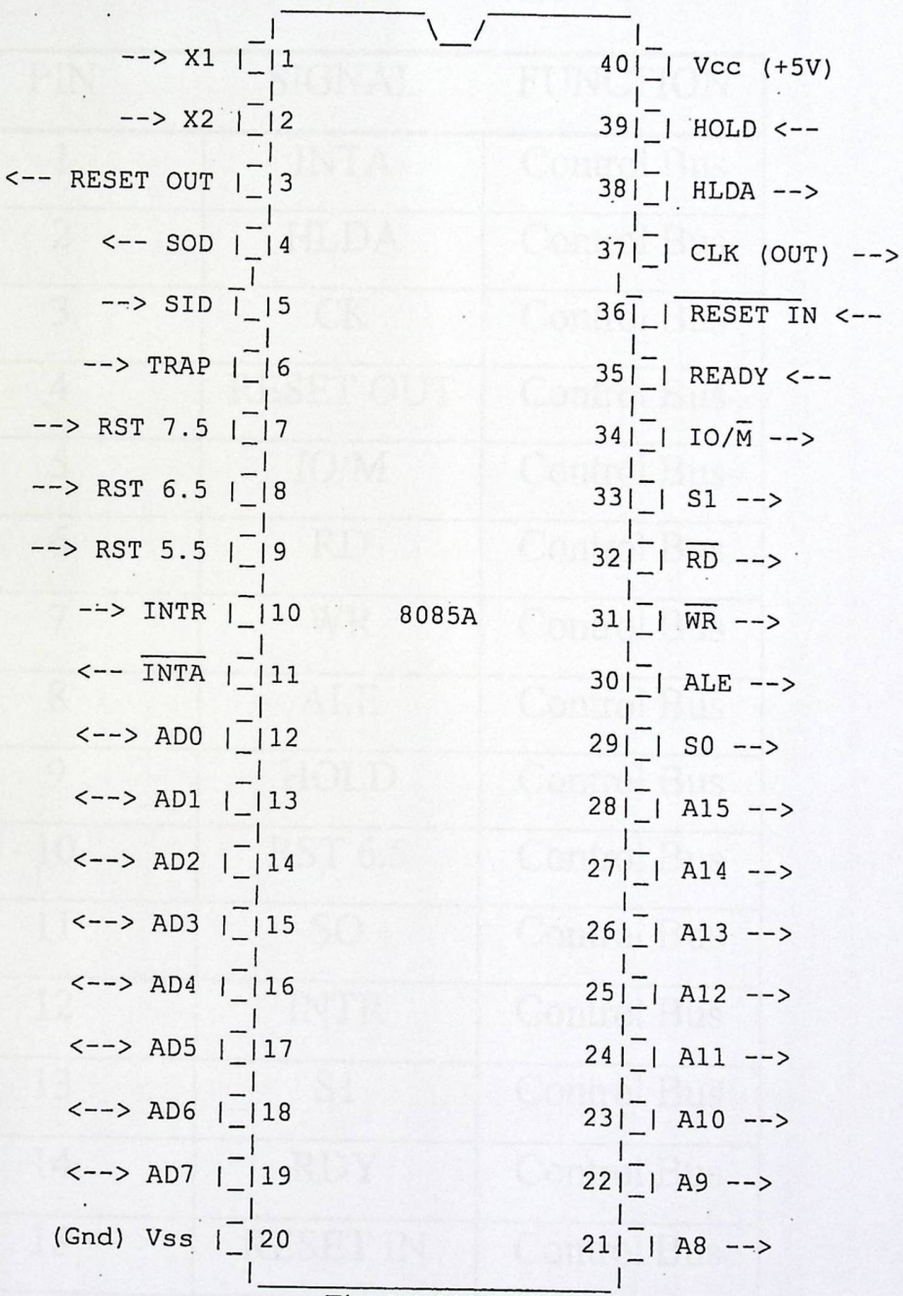


Fig. 2.1 8085 Pins Out

Each unit of the MC-Lab has 50-pin ribbon bus connector on the front panel; this provides the connection between the MC-1 microprocessor and the peripheral devices. The pins and their designated signal functions for the bus connector are listed bellow:

PIN	SIGNAL	FUNCTION
1	INTA	Control Bus
2	HLDA	Control Bus
3	CK	Control Bus
4	RESET OUT	Control Bus
5	IO/M	Control Bus
6	RD	Control Bus
7	WR	Control Bus
8	ALE	Control Bus
9	HOLD	Control Bus
10	RST 6.5	Control Bus
11	SO	Control Bus
12	INTR	Control Bus
13	S1	Control Bus
14	RDY	Control Bus
15	RESET IN	Control Bus
16	A4	Address Bus

17	A3	Address Bus
18	A2	Address Bus
19	A1	Address Bus
20	A0	Address Bus
21	-	Spare
22	-	Spare
23	-	Spare
24	-	Spare
25	-	Spare
26	D0	Data Bus
27	D1	Data Bus
28	D2	Data Bus
29	D3	Data Bus
30	D4	Data Bus
31	D5	Data Bus
32	D6	Data Bus
33	D7	Data Bus
34	A15	Address Bus
35	A14	Address Bus
36	A13	Address Bus
37	A12	Address Bus

38	A11	Address Bus
39	A10	Address Bus
40	A9	Address Bus
41	A8	Address Bus
42	A7	Address Bus
43	A6	Address Bus
44	A6	Address Bus
45	GND	Ground
46	-	Spare
47	-	Spare
48	-	Spare
49	-	Spare
50	-	Spare

Table 2-2: 8085 Cable Signals

2-4 (82C55A) Programmable Peripheral Interfaces (PPI)

2-4-1 Introduction

PPI is a general purpose programmable I/O device which may be used with many different microprocessors. There are 24 I/O pins which may be individually programmed in 2 groups of 12 and used in 3 major modes of operation. The high performance and industry standard configuration of the 82C55A make it compatible with the 80C86, 80C88 and other microprocessors.

2-4-2 Functional Description

Data Bus Buffer: -

This three-state bi-directional 8-bit buffer is used to interface the 82C55A to the system data bus. Data is transmitted or received by the buffer upon execution of input or output instructions by the CPU. Control words and status information are also transferred through the data bus buffer.

Read/Write and Control Logic: -

The function of this logic is to manage all of the internal and external transfers of both Data and Control or Status words. It accepts inputs from the CPU Address and Control busses and in turn, issues commands to both of the Control Groups.

(CS) Chip Select: -

A "low" on this input pin enables the communication between the 82C55A and the CPU.

(RD) Read: -

A "low" on this input pin enables 82C55A to send the data or status information to the CPU on the data bus. In essence, it allows the CPU to "read from" the 82C55A.

(WR) Write: -

A "low" on this input pin enables the CPU to write data or control words into the 82C55A.

(A0 and A1) Port Select 0 and Port Select 1: -

These input signals, in conjunction with the RD and WR inputs, control the selection of one of the three ports or the control word register. They are normally connected to the least significant bits of the address bus (A0 and A1).

(RESET) Reset: -

A "high" on this input initializes the control register to 9Bh and all ports (A, B, C) are set to the input mode. "Bus hold" devices internal to the 82C55A will hold the I/O port inputs to a logic "1" state with a maximum hold current of 400mA.

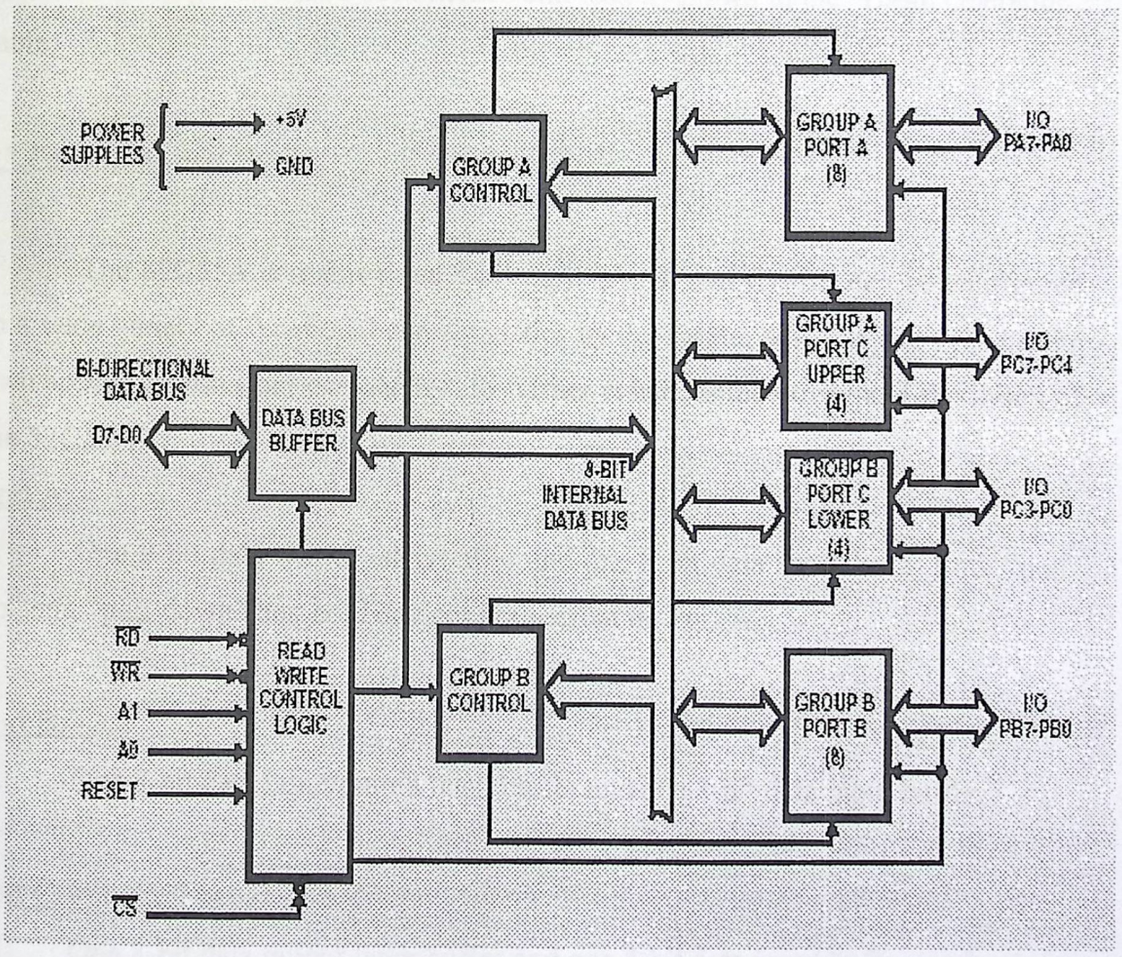


Fig 2-3: 82C55A Block Diagram.

2-4-3 Group A and Group B Controls:

The functional configuration of each port is programmed by the systems software. In essence, the CPU "outputs" a control word to the 82C55A. The control word contains information such as "mode", "bit set", "bit reset", etc., that initializes the functional configuration of the 82C55A.

Each of the Control blocks (Group A and Group B) accepts "commands" from the Read/Write Control logic, receives "control words" from the internal data bus and issues the proper commands to its associated ports. Control Group A - Port A and Port C upper (C7 - C4) Control Group B - Port B and Port C lower (C3 - C0) The control word register can be both written and read as shown in the "Basic Operation" table.

*** Ports A, B, and C: -*

The 82C55A contains three 8-bit ports (A, B, and C). All can be configured to a wide variety of functional characteristics by the system software but each has its own special features or "personality" to further enhances the power and flexibility of the 82C55A.

Port A: -

One 8-bit data output latch/buffer and one 8-bit data input latch. Both "pull-up" and "pull-down" bus-hold devices are present on Port A.

Port B: -

One 8-bit data input/output latch/buffer and one 8-bit data input buffer.

Port C: -

One 8-bit data output latch/buffer and one 8-bit data input buffer (no latch for input). This port can be divided into two 4-bit ports under the mode control. Each 4-

4-bit port contains a 4-bit latch and it can be used for the control signal output and status signal inputs in conjunction with ports A and B.

2-4-4 Control Word:

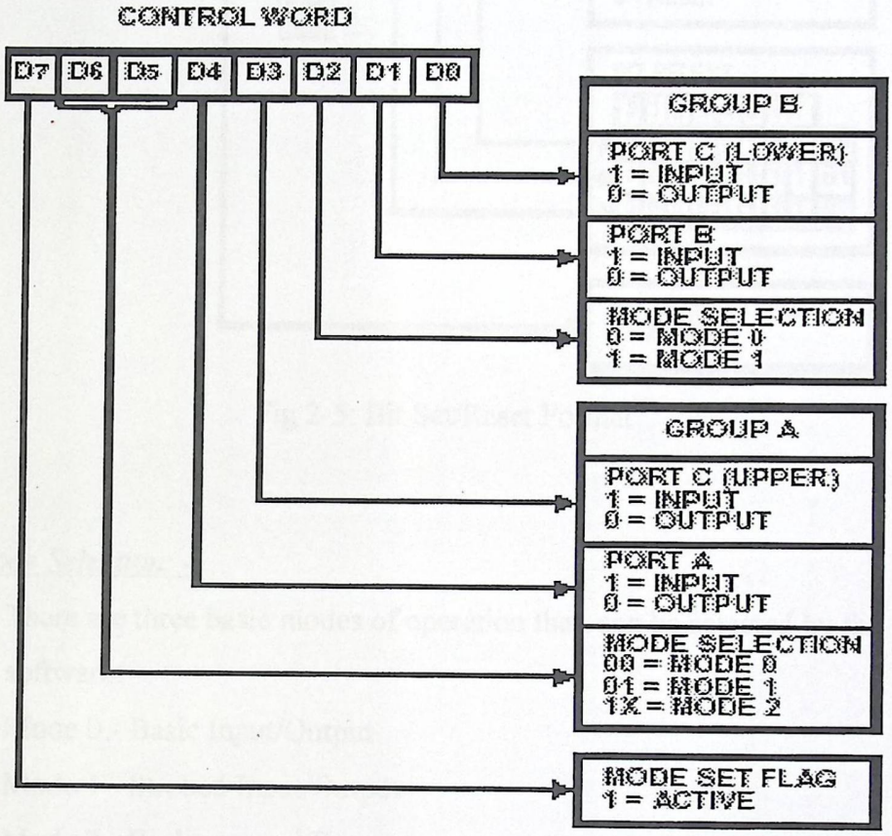


Fig 2-4: Control Word of 82C55A

*** Single Bit Set/Reset Feature: -*

Any of the eight bits of Port C can be Set or Reset using a single Output instruction. This feature reduces software requirements in control-based applications. When Port C is being used as status/control for Port A or B, these bits can be set or reset by using the Bit Set/Reset operation just as if they were output ports.

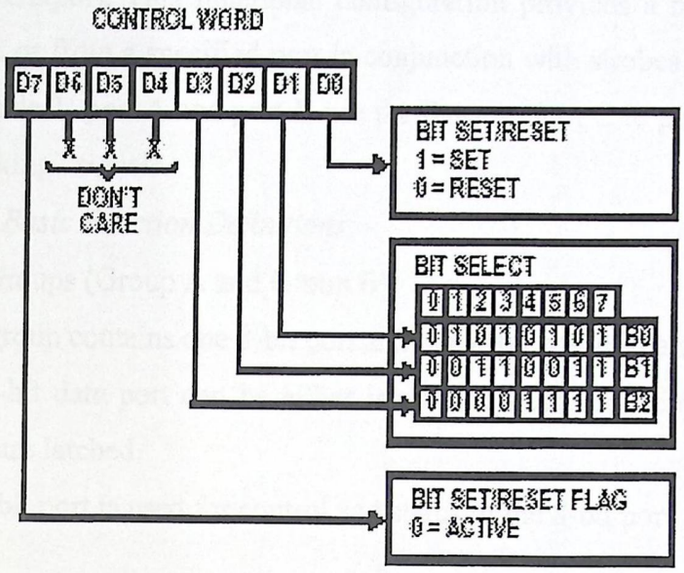


Fig 2-5: Bit Set/Reset Format

**** Mode Selection: -*

There are three basic modes of operation than can be selected by the system software:

- Mode 0 - Basic Input/Output
- Mode 1 - Strobed Input/Output
- Mode 2 - Bi-directional Bus

Mode 0

(Basic Input/Output). This functional configuration provides simple input and output operations for each of the three ports. No handshaking is required, data is simply written to or read from a specific port.

Mode 0 Basic Functional Definitions: -

- Two 8-bit ports and two 4-bit ports
- Any Port can be input or output
- Outputs are latched
- Input are not latched

Mode 1:

(Strobed Input/Output). This functional configuration provides a means for transferring I/O data to or from a specified port in conjunction with strobes or "hand shaking" signals. In mode 1, port A and port B use the lines on port C to generate or accept these "hand shaking" signals.

Mode 1 Basic Function Definitions: -

- Two Groups (Group A and Group B)
- Each group contains one 8-bit port and one 4-bit control/data port
- The 8-bit data port can be either input or output. Both inputs and outputs are latched.
- The 4-bit port is used for control and status of the 8-bit port.

2-4-5 Input Control Signal Definition

STB (Strobe Input):

A "low" on this input loads data into the input latch.

IBF (Input Buffer Full F/F):

A "high" on this output indicates that the data has been loaded into the input latch: in essence, and acknowledgment. IBF is set by STB input being low and is reset by the rising edge of the RD input.

INTR (Interrupt Request):

A "high" on this output can be used to interrupt the CPU when and input device is requesting service. INTR is set by the condition: STB is a "one", IBF is a "one" and INTE is a "one". It is reset by the falling edge of RD. This procedure allows an input device to request service from the CPU by simply strobing its data into the port.

INTE A:

Controlled by bit set/reset of PC4.

INTE B:

Controlled by bit set/reset of PC2.

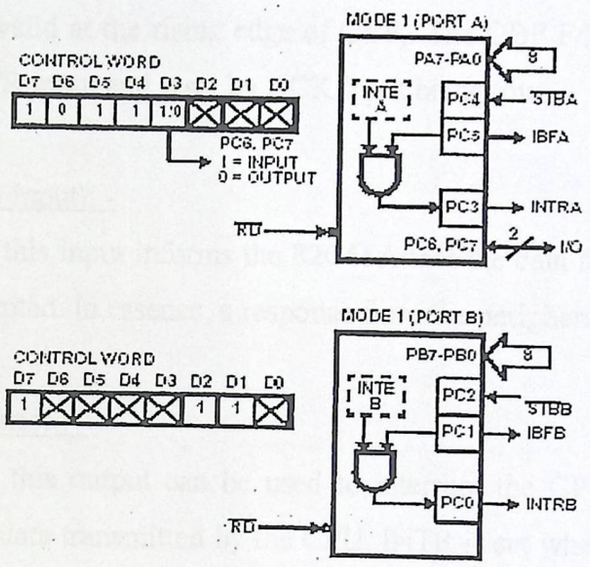


Fig 2-5: Mode 1 Inputs

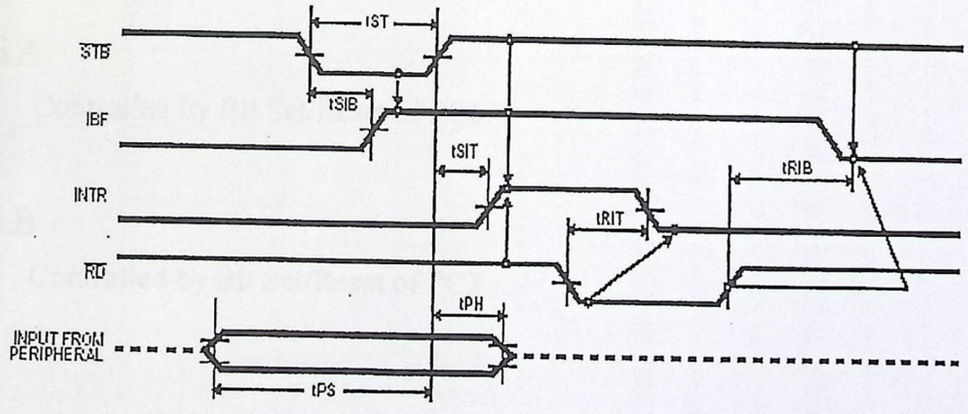


Fig 2-6: Mode1 (Strobe Input)

2-4-6 Output Control Signal Definition

OBF (Output Buffer Full F/F): -

The OBF output will go "low" to indicate that the CPU has written data out to be specified port. This does not mean valid data is sent out of the part at this time since OBF can go true before data is available.

Data is guaranteed valid at the rising edge of OBF,. The OBF F/F will be set by the rising edge of the WR input and reset by ACK input being low.

ACK (Acknowledge Input): -

A "low" on this input informs the 82C55A that the data from Port A or Port B is ready to be accepted. In essence, a response from the peripheral device

INTR (Interrupt Request): -

A "high" on this output can be used to interrupt the CPU when an output device has accepted data transmitted by the CPU. INTR is set when ACK is a "one", OBF is a "one" and INTE is a "one". It is reset by the falling edge of WR.

INTE A

Controlled by Bit Set/Reset of PC6.

INTE B

Controlled by Bit Set/Reset of PC2.

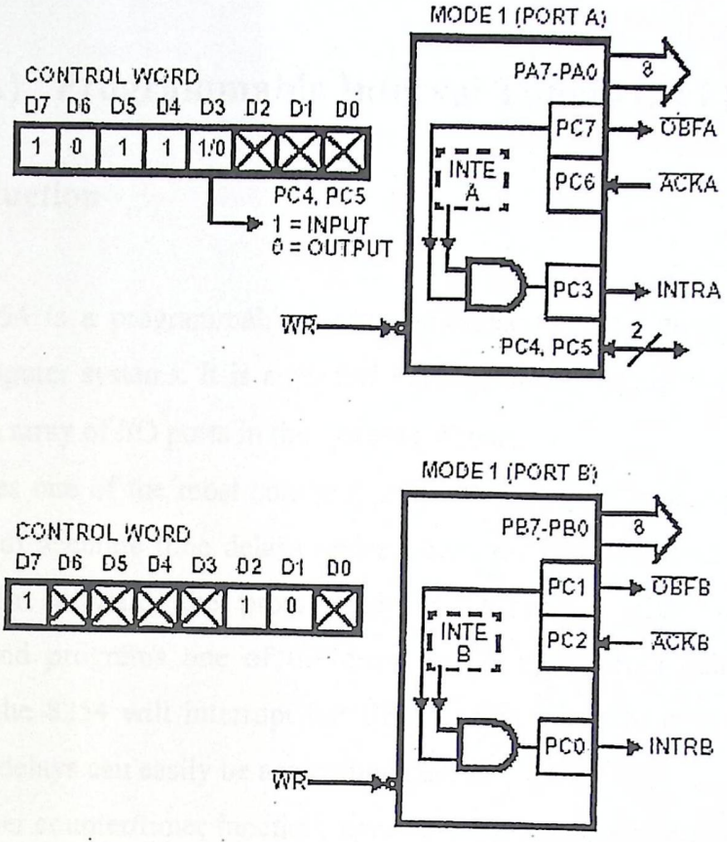


Fig2-7: Mode 1 Output

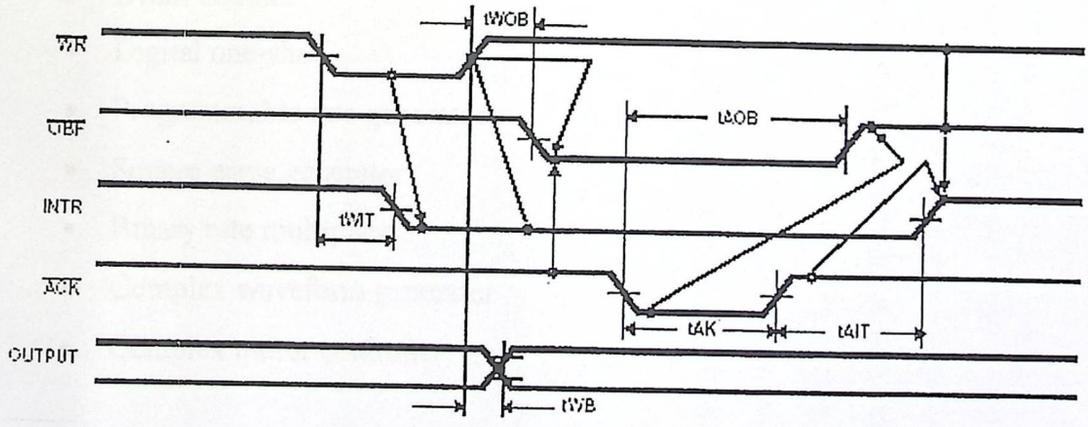


Fig2-8: Mode1 (Strobed Output)

2-5 (8054A) Programmable Interval Timers (PIT)

2-5-1 Introduction

The 8254 is a programmable interval timer/counter designed for use with Intel microcomputer systems. It is a general purpose, multi-timing element that can be treated as an array of I/O ports in the system software.

The 8254 solves one of the most common problems in any microcomputer system, the generation of accurate time delays under software control. Instead of setting up timing loops in software, the programmer configures the 8254 to match his requirements and programs one of the counters for the desired delay. After the desired delay, the 8254 will interrupt the CPU. Software overhead is minimal and variable length delays can easily be accommodated.

Some of the other counter/timer functions common to microcomputers which can be implemented with the 8254 are: -

- Real time clock
- Event-counter
- Digital one-shot
- Programmable rate generator
- Square wave generator
- Binary rate multiplier
- Complex waveform generator
- Complex motor controller

2-5-2 Block diagram Description

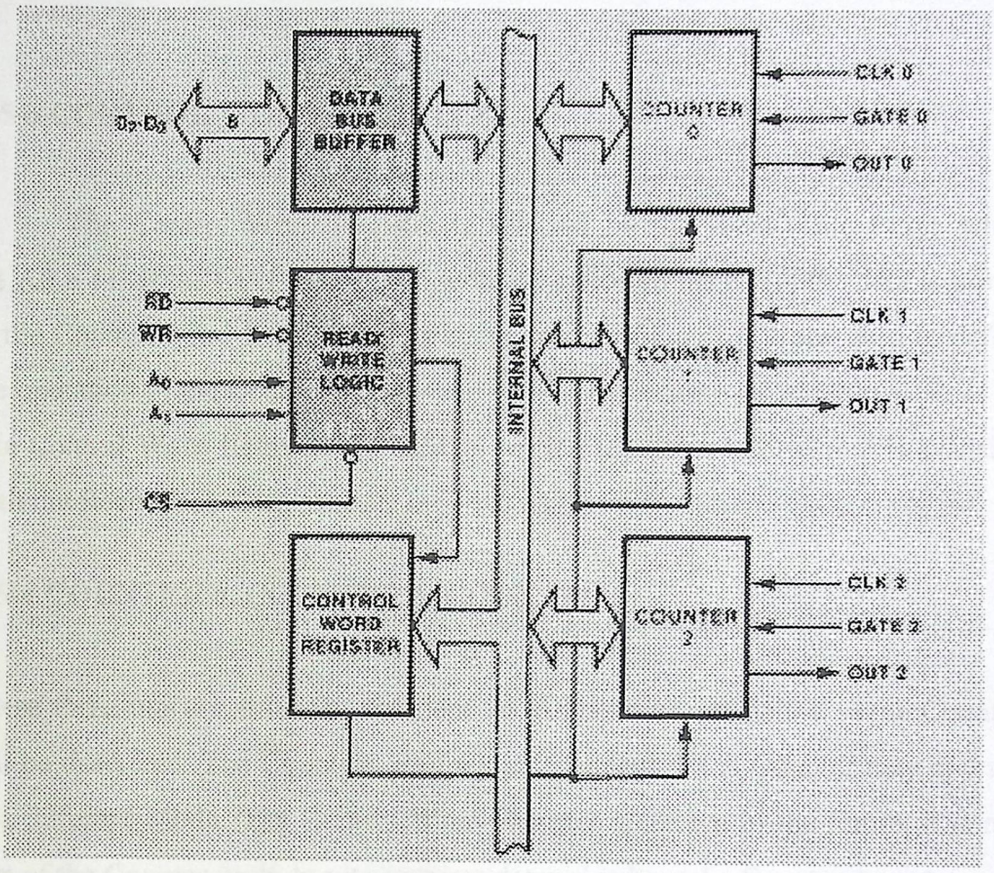


Fig 2-12: 8254 Block Diagram

data bus buffer: -

This 3-state, bi-directional, 8-bit buffer is used to interface the 8254 to the system bus.

read/write logic: -

The Read/Write Logic accepts inputs from the system bus and generates control signals for the other functional blocks of the 8254. A1 and A0 select one of the three counters or the Control Word Register to be read from/written into. A ``low" on the RD input tells the 8254 that the CPU is reading one of the counters. A ``low" on the WR input tells the 8254 that the CPU is writing either a Control Word

or an initial count. Both RD and WR are qualified by CS; RD and WR are ignored unless the 8254 has been selected by holding CS low.

2-5-3 CONTROL WORD REGISTER

The Control Word Register is selected by the Read/Write Logic when A1,A0 e 11. If the CPU then does a write operation to the 8254, the data is stored in the Control Word Register and is interpreted as a Control Word used to define the operation of the Counters.

The Control Word Register can only be written to; status information is available with the Read-Back Command.

2-5-4 COUNTER 0, COUNTER 1, COUNTER 2

These three functional blocks are identical in operation, so only a single Counter will be described. The Counters are fully independent. Each Counter may operate in a different Mode. The Control Word Register is shown in the figure; it is not part of the Counter itself, but its contents determine how the Counter operates.

The actual counter is labeled CE (for ``Counting Element"). It is a 16-bit preset table synchronous down counter.

OLM and OLL are two 8-bit latches. OL stands for ``Output Latch"; the subscripts M and L stand for ``Most significant byte" and ``Least significant byte"

Fig 2-10: Internal Block Diagram of 825

2-5-5 Control Word of 8254:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
SC1	SC0	RW1	RW0	M2	M1	M0	BCD

SC—Select Counter

SC1	SC0	
0	0	Select Counter 0
0	1	Select Counter 1
1	0	Select Counter 2
1	1	Read-Back Command (see Read Operations)

M—Mode

M2	M1	M0	
0	0	0	Mode 0
0	0	1	Mode 1
X	1	0	Mode 2
X	1	1	Mode 3
1	0	0	Mode 4
1	0	1	Mode 5

RW—Read/Write

RW1	RW0	
0	0	Counter Latch Command (see Read Operations)
0	1	Read/Write least significant byte only
1	0	Read/Write most significant byte only
1	1	Read/Write least significant byte first, then most significant byte

BCD

0	Binary Counter 16-bits
1	Binary Coded Decimal (BCD) Counter (4 Decades)

Fig 2-11: Control Word Format

Write Operations

The programming procedure for the 8254 is very flexible. Only two conventions need to be remembered: -

1) For each Counter, the Control Word must be written before the initial count is written.
2) The initial count must follow the count format specified in the Control Word (least significant byte only, most significant byte only, or least significant byte and then most significant byte). Since the Control Word Register and the three Counters have separate addresses (selected by the A1,A0 inputs), and each Control Word specifies the Counter it applies to (SC0,SC1 bits), no special instruction sequence is required. Any programming sequence that follows the conventions is acceptable. A new initial count may be written to a Counter at any time without affecting the Counter's programmed Mode in any

way. Counting will be affected as described in the Mode definitions. The new count must follow the programmed count format. If a Counter is programmed to read/write two-byte counts, the following precaution applies: A program must not transfer control between writing the first and second byte to another routine which also writes into that same Counter. Otherwise, the Counter will be loaded with an incorrect count.

Read Operations

It is often desirable to read the value of a Counter without disturbing the count in progress. This is easily done in the 8254. There are three possible methods for reading the counters: a simple read operation, the Counter Latch Command, and the Read-Back Command.

Each is explained below. The first method is to perform a simple read operation. To read the Counter, which is selected with the A1, A0 inputs, the CLK input of the selected Counter must be inhibited by using either the GATE input or external logic. Otherwise, the count may be in the process of changing when it is read, giving an undefined result.

2-5-6 Mode Definitions

The following are defined for use in describing the operation of the 8254: -

- CLK Pulse: a rising edge, then a falling edge, in that order, of a Counter's CLK input.
- Trigger: a rising edge of a Counter's GATE input.
- Counter loading: the transfer of a count from the CR to the CE (refer to the ``Functional Description'').

MODE 0: INTERRUPT ON TERMINAL COUNT

MODE 0: INTERRUPT ON TERMINAL COUNT

Mode 0 is typically used for event counting. After the Control Word is written, OUT is initially low, and will remain low until the Counter reaches zero. OUT then goes high and remains high until a new count or a new Mode 0 Control Word is written into the Counter.

GATE e 1 enables counting; GATE e 0 disables counting. GATE has no effect on OUT. After the Control Word and initial count are written to a Counter, the initial count will be loaded on the next CLK pulse. This CLK pulse does not decrement the count, so for an initial count of N, OUT does not go high until N a 1 CLK pulses after the initial count is written. If a new count is written to the Counter, it will be loaded on the next CLK pulse and counting will continue from the new count. If a two-byte count is written, the *following happens:* -

1) Writing the first byte disables counting. OUT is set low immediately (no clock pulse required).
2) Writing the second byte allows the new count to be loaded on the next CLK pulse. This allows the counting sequence to be synchronized by software. Again, OUT does not go high until Na1 CLK pulses after the new count of N is written. If an initial count is written while GATE e 0, it will still be loaded on the next CLK pulse. When GATE goes high, OUT will go high N CLK pulses later; no CLK pulse is needed to load the Counter as this has already been done.

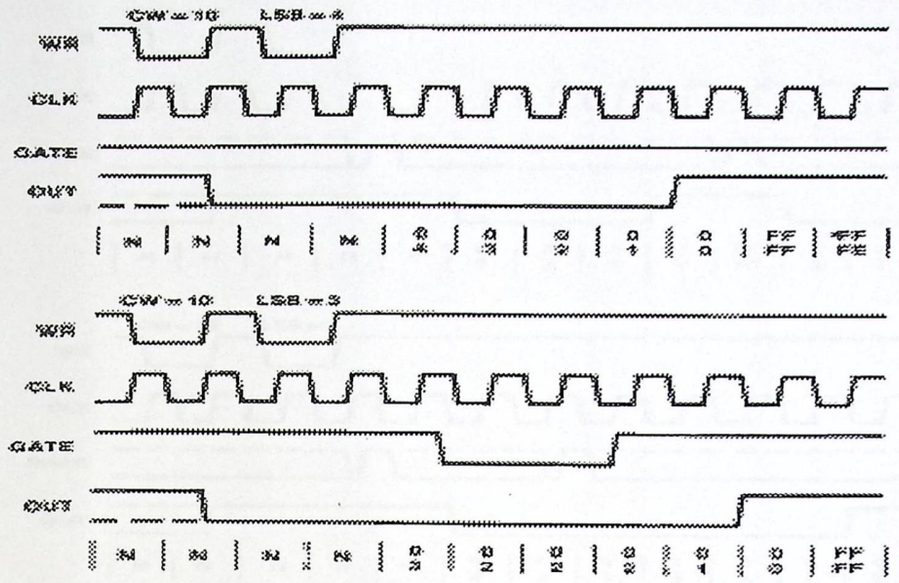


Fig 2-15: Mode0 for 8254.

MODE 1: HARDWARE RETRIGGERABLE

ONE-SHOT OUT will be initially high. OUT will go low on the CLK pulse following a trigger to begin the one-shot pulse, and will remain low until the Counter reaches zero.

OUT will then go high and remain high until the CLK pulse after the next trigger. After writing the Control Word and initial count, the Counter is armed. A trigger results in loading the Counter and setting OUT low on the next CLK pulse, thus starting the one-shot pulse. An initial count of N will result in a one-shot pulse N CLK cycles in duration. The one-shot is retrigger able; hence OUT will remain low for N CLK pulses after any trigger. The one-shot pulse can be repeated without rewriting the same count into the counter. GATE has no effect on OUT.

If a new count is written to the Counter during a one shot pulse, the current one-shot is not affected unless the counter is retriggered. In that case, the Counter is loaded with the new count and the one shot pulse continues until the new count expires.

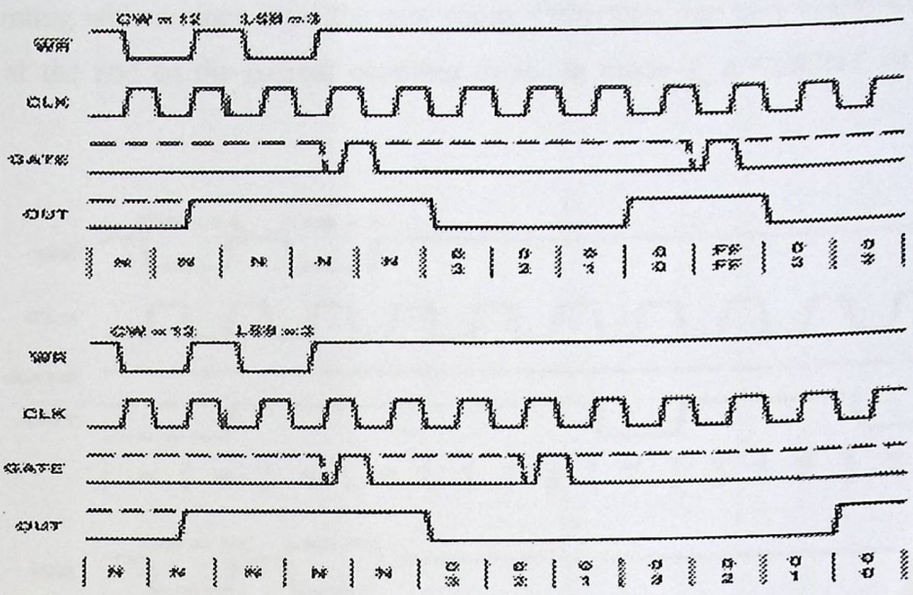


Fig 2-16: Mode 1 for 8254.

MODE 2: RATE GENERATOR

This Mode functions like a divide-by-N counter. It is typically used to generate a Real Time Clock interrupt. OUT will initially be high. When the initial count has decremented to 1, OUT goes low for one CLK pulse. OUT then goes high again, the Counter reloads the initial count and the process is repeated. Mode 2 is periodic; the same sequence is repeated indefinitely. For an initial count of N, the sequence repeats every N CLK cycles. GATE e 1 enables counting; GATE e 0 disables counting. If GATE goes low during an output pulse, OUT is set high immediately. A trigger reloads the Counter with the initial count on the next CLK pulse; OUT goes low N CLK pulses after the trigger. Thus the GATE input can be used to synchronize the Counter. After writing a Control Word and initial count, the Counter will be loaded on the next CLK pulse. OUT goes low N CLK Pulses after the initial count is written. This allows the Counter to be synchronized by software also. Writing a new count while counting does not affect the current counting sequence. If a trigger is received after writing a new count but before the end of the current period, the Counter will be loaded with the new count on the next CLK pulse and counting will continue from the new count.

and counting will continue from the new count. Otherwise, the new count will be loaded at the end of the current counting cycle. In mode 2, a COUNT of 1 is illegal.

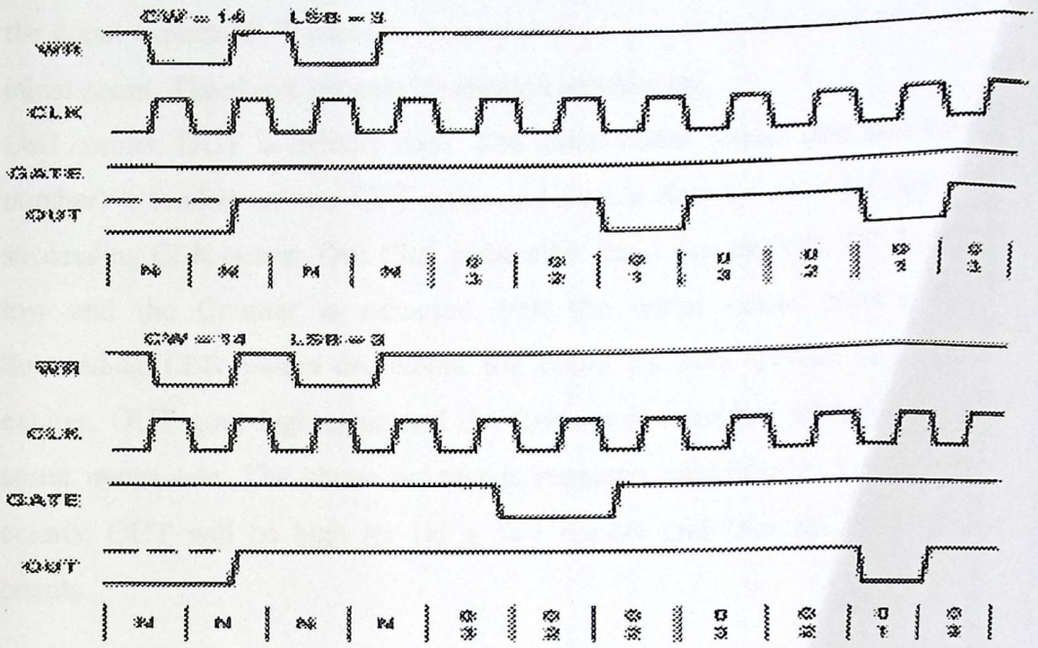


Fig 2.17: Mode2 for 8254.

MODE 3: SQUARE WAVE MODE

Mode 3 is typically used for Baud rate generation. Mode 3 is similar to Mode 2 except for the duty cycle of OUT. OUT will initially be high. When half the initial count has expired, OUT goes low for the remainder of the count. Mode 3 is periodic; the sequence above is repeated indefinitely. An initial count of N results in a square wave with a period of N CLK cycles. GATE e 1 enables counting; GATE e 0 disables counting. If GATE goes low while OUT is low, OUT is set high immediately; no CLK pulse is required. A trigger reloads the Counter with the initial count on the next CLK pulse. Thus the GATE input can be used to synchronize the Counter. After writing a Control Word and initial

new count will be loaded at the end of the current half-cycle. Mode 3 is implemented as follows: -

- *Even counts:* OUT is initially high. The initial count is loaded on one CLK pulse and then is decremented by two on succeeding CLK pulses. When the count expires OUT changes value and the Counter is reloaded with the initial count. The above process is repeated indefinitely.
- *Odd counts*: OUT is initially high. The initial count minus one (an even number) is loaded on one CLK pulse and then is decremented by two on succeeding CLK pulses. One CLK pulse after the count expires, OUT goes low and the Counter is reloaded with the initial count minus one. Succeeding CLK pulses decrement the count by two. When the count expires, OUT goes high again and the Counter is reloaded with the initial count minus one. The above process is repeated indefinitely. So for odd counts, OUT will be high for (N a 1)/2 counts and low for (N b 1)/2 counts.

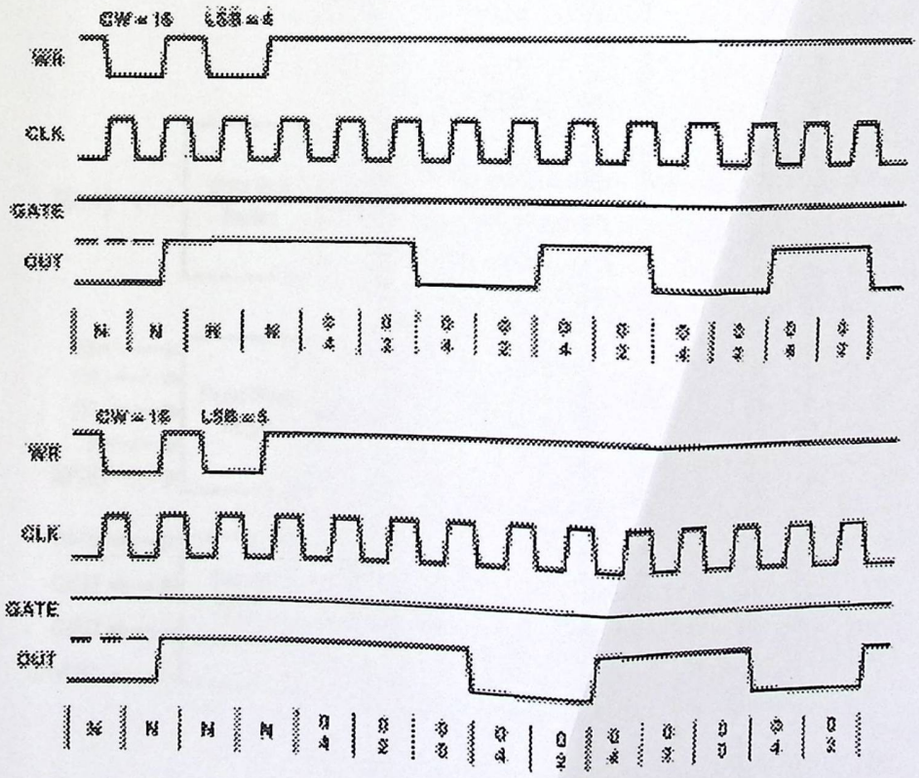


Fig 2.18 Mode3 for 8254.

2-7 (8259A) Programmable Interrupt Controller (PIC)

2-7-1 Features

- Eight vectored priority interrupts per mega function. Up to 64 vectored priority interrupts with cascading
- Programming for all 8259A modes & operational features
  - MCS-80/85 & 8088/8085 processor modes
  - Fully nested mode & special fully nested mode
  - Special mask mode
  - Buffered mode
  - Specific rotation
  - Edge- & level-triggered interrupt input modes
  - Reading of interrupt request register (IRR) & in-service register (ISR) through data bus
  - Writing & reading of interrupt mask register (IMR) through data bus

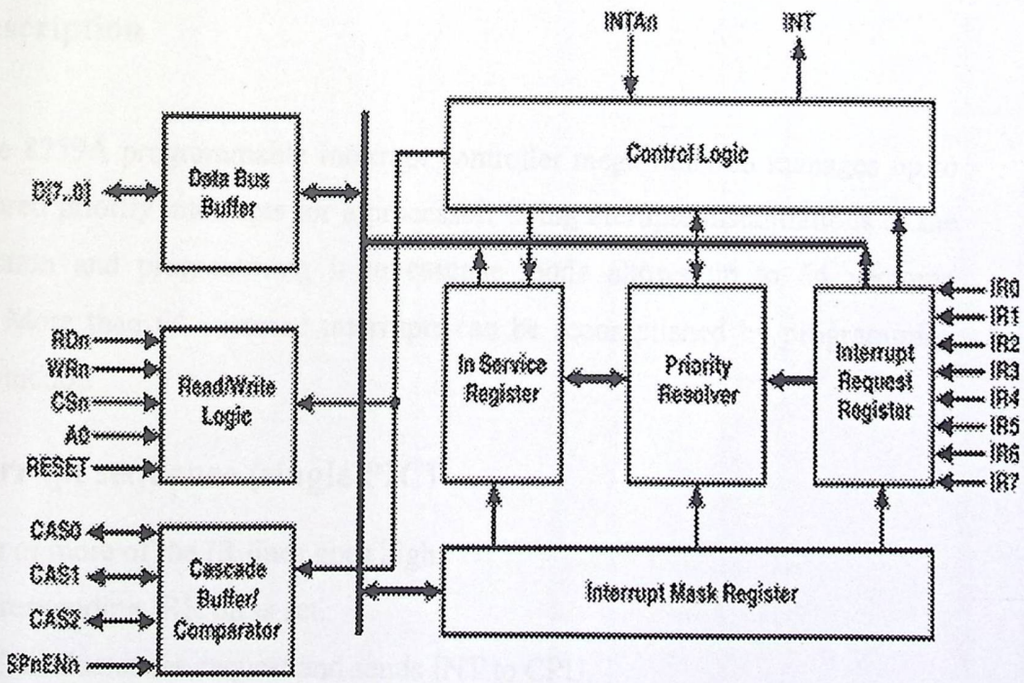


Fig 2- 24: 8259 block diagram

D0-D7	Bi-directional, restated, buffered data lines. Connected to data bus directly or through buffers
RD-bar	Active low read control
WR-bar	Active low write control
A0	Address input line, used to select control register
CS-bar	Active low chip select
CAS0-2	Bi-directional, 3 bit cascade lines. In master mode, PIC places slave ID no. on these lines. In slave mode, the PIC reads slave ID no. from master on these lines. It may be regarded as slave-select.
SP-bar / EN-bar	Slave program / enable. In non-buffered mode, it is SP-bar input, used to distinguish master/slave PIC. In buffered mode, it is output line used to enable buffers
INT	Interrupt line, connected to INTR of microprocessor
INTA-bar	Interrupt ack, received active low from microprocessor
IR0-7	Asynchronous IRQ input lines, generated by peripherals.

Table 2.3 PIN out of 8259:

2-7-3 Description

The 8259A programmable interrupt controller mega function manages up to eight vectored priority interrupts for a processor. Using multiple instantiations of the mega function and programming it to cascade mode allows up to 64 vectored interrupts. More than 64 vectored interrupts can be accomplished by programming the mega function

2.7.4Interrupt sequence (single PIC)

1. One or more of the IR lines goes high.
2. Corresponding IRR bit is set.
3. 8259 evaluates the request and sends INT to CPU.
4. CPU sends INTA-bar.

0-D7	Bi-directional, through buffers
D-bar	Active low read control
R-bar	Active low write control
0	Address input line, used to select, first low byte then high byte.
S-bar	Active low chip select
AS0-2	Bi-directional, 3 bit cascade lines. In master mode, the PIC reads these lines. In slave mode, these lines are input lines.
-bar / V-bar	Slave program / enable buffers
T	Interrupt line, connected to INTR of microprocessor
TA-bar	Interrupt ack, received active low from microprocessor
0-7	Asynchronous R/Q data lines. Connected to peripherals

Synchronous Asynchronous Receiver (USART)

Description

The 8251 programmable communications interface (USART) core provides formatting and control to a serial communication channel.

The core has select, read/write, interrupt and bus interface logic features that allow data transfers over an 8-bit bi-directional parallel data bus system. With proper formatting and error checking, the core can transmit and receive serial data, supporting both synchronous and asynchronous operation.

2-8-2 Features

- Synchronous and asynchronous operation
- Programmable data word length, parity and stop bits
- Parity, overrun and framing error checking instructions and counting loop interactions
- Supports up to 1.750 Mbps transmission rates
- Divide-by 1,-16,-64 mode
- False start bit deletion
- Automatic break detection
- Internal and external synch character detection

5. Highest priority ISR is set. IRR is reset.
6. 8259 releases CALL instruction on data bus.
7. CALL causes CPU to initiate two more INTA-bar's.
8. 8259 releases the subroutine address, first low byte then high byte.
9. ISR bit is reset depending on mode

2-8 (8251A) Universal Synchronous Asynchronous Receiver Transmitter (USART)

2-8-1 Function Description

The C8251 programmable communications interface (USART) core provides data formatting and control to a serial communication channel.

The core has select, read/write, interrupt and bus interface logic features that allow data transfers over an 8-bit bi-directional parallel data bus system. With proper formatting and error checking, the core can transmit and receive serial data, supporting both synchronous and asynchronous operation.

2-8-2 Features

- Synchronous and asynchronous operation
- Programmable data word length, parity and stop bits
- Parity, overrun and framing error checking instructions and counting loop interactions
- Supports up to 1.750 Mbps transmission rates
- Divide-by 1,-16,-64 mode
- False start bit deletion
- Automatic break detection
- Internal and external synch character detection

- Peripheral modem control functions
- The C8251 was developed in VHDL and synthesizes to approximately 2,300 gates depending on the technology used
- Functionality based on the Intel 8251A device

2-8-3 Symbol

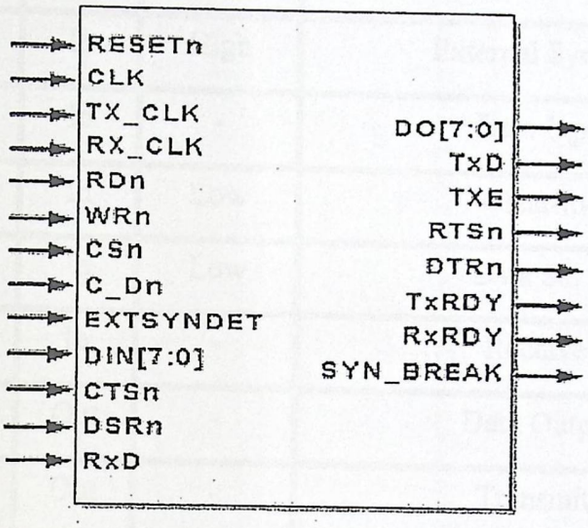


Fig 2-17: 8251 Symbol

2-8-4 Pin Description

The following table presents each pin in the 8251 chip with its function: -

Table 2-1: 8251 Pin Description.

Name	Type	Polarity	Description
RESET	In	Low	External Rest
CLK	In	-	Master Clock
TX_CLK	In	-	Transmit Clock

RX_CLK	In	-	Receive Clock
RDn	In	Low	Read Control
WRn	In	Low	Write Control
CSn	In	Low	Chip Select
C_Dn	In	-	Control/Data Select
EXTSYNDET	In	High	External Synch Detect
DIN[7:0]	In	-	Data Input Bus
CTSn	In	Low	Clear-to-Send
DSRn	In	Low	Data Set Ready
RxD	In	-	Receive Data
D0[7:0]	Out	-	Data Output Bus
TxD	Out	-	Transmit Data
TxE	Out	Low	Transmitter Empty
RTSn	Out	Low	Request-to-Send
DTRn	Out	Low	Data Terminal Ready
TxRDY	Out	High	Transmit Ready
RxRDY	Out	High	Receiver Ready
SYN_BREAK	Out	Low	Sync/Break Detect

Table 2.4 USART PIN Description

2-8-5 Block Diagram

The following diagram describes the components of the 8251 chip: -

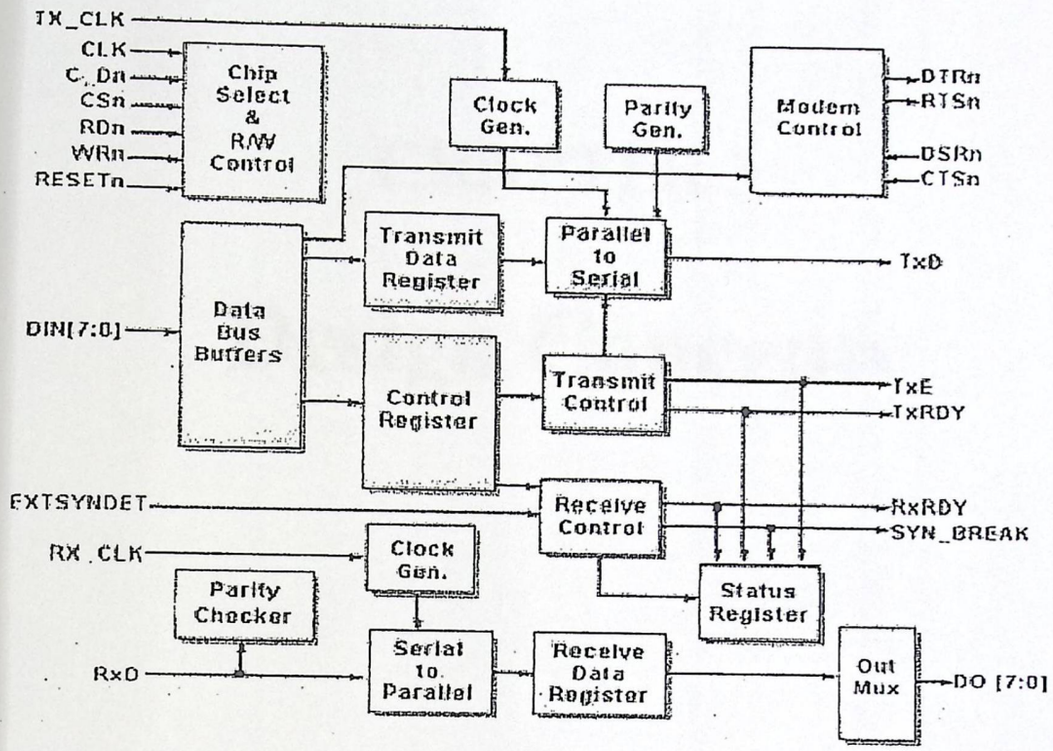


Fig 2-18: 8251 block diagram

CHAPTER 3

Design Concepts

Design Concept

Project Objective

The basic objective of this project is to design experimental board that solve the problem of unaccomplished experiments in the 8085 lab at PPU , some experiment are already available in the lab using 8085 kit while others are not available, so the team work's attention is to build the necessary boards that help to do the experiments, the main objectives behind those experiment is to understand the 8085 microprocessor and its buses ,ports and input/output interface through understanding the following subjects: -

- I/O address space and data transfers
- I/O instructions
- Eight-byte-wide output ports wide output port

The specified boards that the teams going to build are: -

a. Programmable peripheral interface (PPI-8255)
b. Universal Synchronous Asynchronous Receiver Transmitter (USART-8251)
c. Programmable interrupt controller (PIC-8259)
d. Programmable Interval Timer (PIT-8254)

General Block Diagram

The following diagram represents the five chips with the 8085μp: -

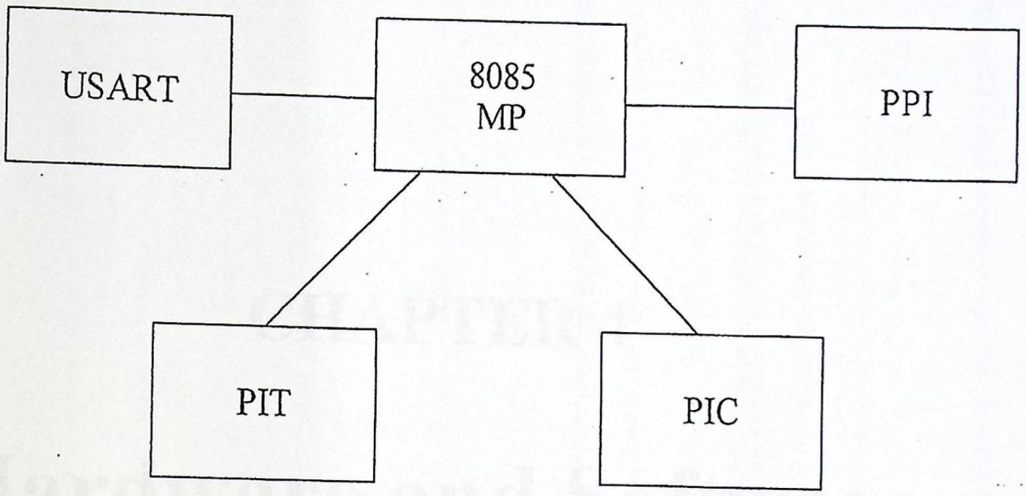


Fig 3-1: Block Diagram.

How System Works

The system includes the 8085 kit , interfaces with the experimental board , each to board contains all the desired connections to work and connected to data bus and address bus through the 50 bin cable we built ,that will do the appropriate data transferring the student can be able to see the data on the output port or input ports and address as well, it depends on the design circuits and the circuit will work according to the program that operate that circuit and executed by the 8085 microprocessor assembly .

CHAPTER 4

Hardware and Software System Design

Hardware and Software Design

4-1 8255 experiments:

TABLE 4 -1: POERT ADDRESSE OF 8255.

PORT	ADDRESS
PORT A	A0
PORT B	A1
PORT C	A2

4-1-1 8255 at MODE 0

The goal of this experiment is to understanding the way to program 8255 at I/O (Mode0). Fig 4-1 is show the flowchart of this mode; and fig 4-2 show the control word. We will use the port A as the input port and the port b as the output.

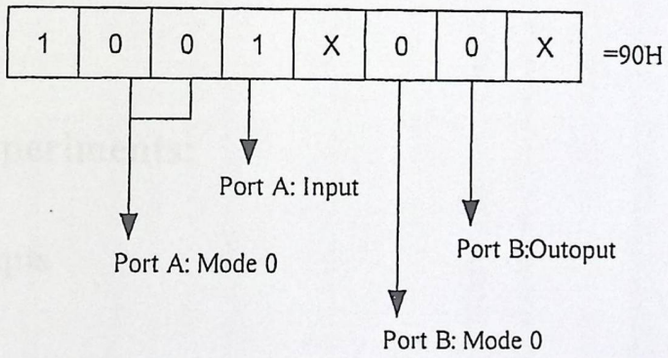


Fig 4-2: Control word of port A and port B at mode0

In this experiment we concentrate on programming and interfacing techniques by which I/O devices can be addressed and interfaced to microcomputer.

The control word 82 (see Mode definition Format and port definition table for mode 0) thereby selecting ports A and C(upper and lower) as output ports and ports B as input port.

Connect 8 switches to port B of the MC unit ,connect LEDs to port A and C of the MC unit, execute the given program in the following table and you will be able to immediately observe the changes in data input at port Bs switches at the output ports A and C.

LXI SP,20B0
MVI A,82
OUT A3
IN A1
OUT A0
OUT A2
JMP 2007

4-2 8259 experiments:

4-2-1 Interrupts

An interrupt occurs when a device needs to notify the CPU of some exceptional event. Some events which cause interrupts:

- mouse click or movement
- keystrokes
- completion of a disk or printer operation

On a typical I/O bus, some of the bus lines are allocated for the communication between CPU and device when an interrupt occurs.

1. Device requests an interrupt by asserting its INT line.
2. CPU acknowledges the interrupt by asserting its INTA line.
3. Device deserts its INT line.
4. CPU disserts its INTA line.
5. CPU transfers control to the interrupt handling routine.

The 8259A contains several registers which the CPU can use to program it (e.g., setting priorities on the interrupt lines). The RD, WR, A0, CS, and D0-D7 lines are used to read and write those registers as if the 8259A were an I/O device

4-2-2 PIC Experiments and applications:

The objective of these experiments is to process single and multiple interrupts on the microprocessor and to write assembly routines to initialize interrupt vector. The 8259 PIC is functionally based on the Intel 8259A. Eight Interrupt requests are prioritized for a processor. To minimize size and maximize performance.

- Selected features of industry standard 8259.
- Eight-level priority controller.
- Individual request mask capability.
- Edge triggered detection.
- Can be cascaded to allow more requests.
- Automatic End of Interrupt.
- Read of interrupt and interrupt in-service.

For the more general INTR request, good for all interrupt types from 0 to 255, the following sequence takes place:

1. INTR is activated, requesting an interrupt.
2. The present instruction is allowed to be completed.
3. The CS, IP, and flags register are automatically pushed onto the stack.
4. INTA (low active) goes low and externally circuitry supplies a type number (0 -255) by away of data bus.
5. The type number is automatically multiplied by four within 8085 to give the address of the interrupt vector table.
6. The CS: IP address of the ISR is pulled from the interrupt vector table and jammed into the CS: IP register.
7. The ISR is executed.
8. The special IRET instruction pops the CS: IP return address and flag register off the stack, and we return to the main program.

We can use this circuit an many applications since the request is coming from any external device we represent it by the switches and the generated interrupt from the mp can be used to serves any other circuitry like on/off of LED or stopping steeper motor or in PPI applications .

In the final experimental board, the application should be completed in single board to be clear for student every part of the interrupts interface circuit.

The following program explain how PIC accept an interrupt from a button

```
LXI     SP,20B0
MVI     A,0F
OUT     20
MVI     A,37            ;ICW1
OUT     B0
MVI     A,20            ;ICW2
OUT     B1
MVI     A,02            ;ICW4
OUT     B1
MVI     A,F8            ;OCW4
OUT     B1
EI
HLT
2020: MVI A,55
OUT 21
EI
RET
```

4-3 Timer Experiments and applications:

The goal of this experiment to show the different output for different modes fro 8254, and the address for each counter (look at table 2) and control register.

Experiments design:

Two experiments are done for the timer, one in mode2: Rate generator mode to operate a DC motor and control its speed by varying the counter value and the second is in mode3: Square Wave Generator. And music tone is delivered on a speaker that connected on the timer output.

Experement#1: Music tones

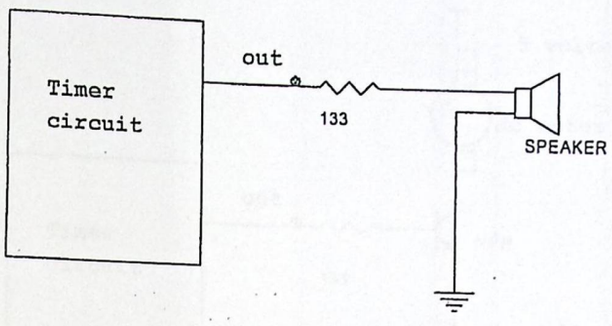


Fig.4-3: Music Tones Circuit

Program: -

```
        LXI SP,20B0
START:  MVI B,03
        LXI H,2060
AGAIN:  MOV A, M
        OUT 93          ;C.W Reg
        INX H
        MOV A, M
        OUT  90H        ;L.S.B
        INX H
        MOV A, M
        OUT 90          ;M.S,B
        INX H
        LXI D,FFFF      ;DELAY
        CAL DEALY
        DCR B
        JNZ AGAIN
        JMP START
```

Experiment#: DC motor control

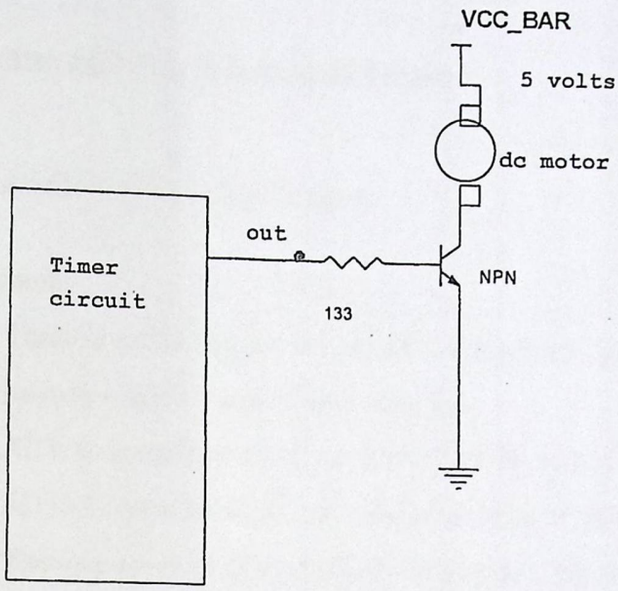


Fig.4-4: DC motor Control Circuit.

Program: -

```
LXI SP,20B0
MVI A,B4
OUT 93
MVI A,04  ;The value of the counter can be changed
            to control the speed of the motor
OUT 92
MVI A,00
OUT 92
HLT
```

4.4 USART Experiments:

4.4.1 Synchronous and Asynchronous Buses

A bus may be synchronous or asynchronous.

- synchronous
  - There is a clock signal on one of the bus lines, which can be read by every device connected to the bus.
  - All bus operations are synchronized to the clock.
  - All bus operations take an integral number of bus clock cycles.
  - The bus clock is different from the one driving the CPU, and is generally slower.
    - Pentium: 2 GHz
    - ISA: 8.33 MHz
    - PCI: 33 or 66 MHz
- asynchronous
  - No clock
  - Bus operations depend on "handshaking" protocols between two communicating devices.

4.4.2 The experiment

This experiment shows step by step how to operate the USART as receiver in the asynchronous mode. The receiver accepts serial data on RxD pin and converts it to parallel data according to the appropriate format. The program given in table () defines an asynchronous mode that requires a START bit preceding eight data characters and a single STOP bit following them.

When the USART is in the asynchronous mode and is ready to accept a character (i.e,it is not in the process of receiving a character),it looks for a low level on the RxD line. When it sees a low level it assumes that it is a START bit and enables an internal counter. At a count equivalent to one-half a bit time, the RxD line is sampled again. If the line is still low, a valid START bit has probably been received and the USART processed to assemble the character. In either case the receiver aborts its operation and prepares itself to accept a new character. After the successful reception of a START bit the USART clocks in the data bits and the STOP bit, and then transfers the data to the receive data register.

Connect a toggle switch to RxD and set it high .Connect a debounced pushbutton to RxC .Execute the program .Set the toggle switch to low and press the pushbutton Now the microcomputer has received the START bit and it is ready to accept the eight data bits. Set the toggle switch to high and press eight times the pushbutton Thus, eight data bits (ones) have been received by the USART.

Another RxX pulses terminates the operation; press the pushbutton (STOP) bit and observe how port A LED s are turned on. Try again with another character remembering to follow this sequence: one START bit (low), eight data bit (high or low) and STOP bit (high or low).

LXI SP,20B0
MVI A,01
OUT 20
MVI A,4D
OUT B1
MVI A,04
OUT B1
IN B1

CHAPTER 5
Testing and Implementation

Testing and Implementation

5.1 Introduction

The design of the circuits come after many testing stages first of all we suppose a design to test the signal that is out of the kit to verify the function of each signal so we built a simple circuit and execute the many codes of programs to ensure that .

The second stage in testing is to build each design using bread boards.

Third step in testing is writing the programs for each circuit with the supposed application then using the assembly program to generates the object code for every program to be entered in the kit.

Finally after we test the circuit and run the program if there is in exchange for any ICs we do it until the design be excellent for the desired application so we use Orcad 9 to draw the design and adopt it.

5.2 The Signal Testing Programs

5.2.1 Using The Data bus port

Program:

```
LXI SP,20B0      ;initialize stack pointer
IN CO            ; Read Switches
OUT              ; Turn LEDs
CALL UPDDT       ; Displays A in the field of the display
JMP 2003         ; Do it again
```

5.2.2 Using The address bus

Program

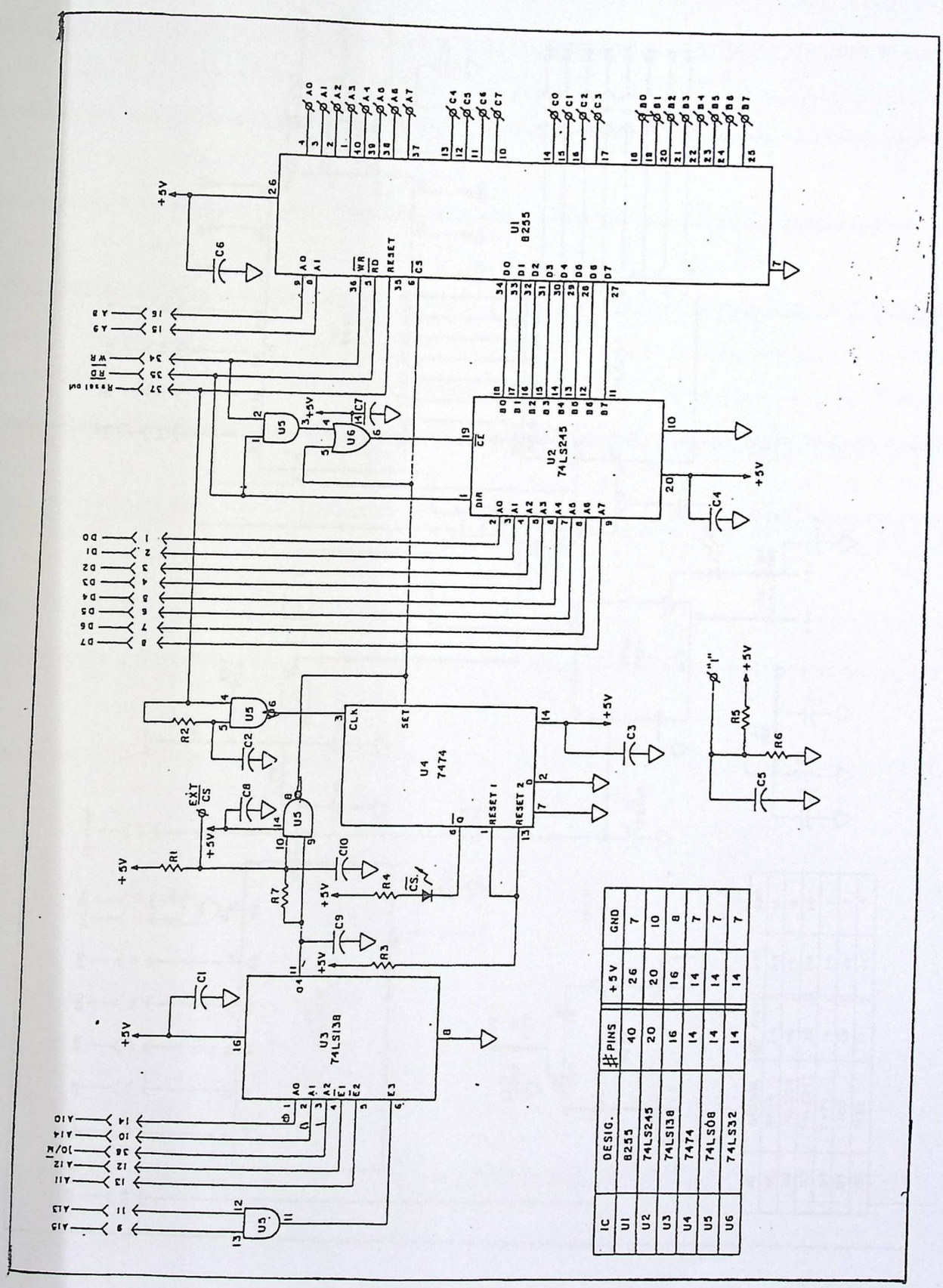
```
 LXI H, 100        ; load address value (low and high)
MOV M,A            ;Write in memory location H,L
JMP 2003           ; jump to loop
```

5.2.3 Using The Restart Interrupt

Program

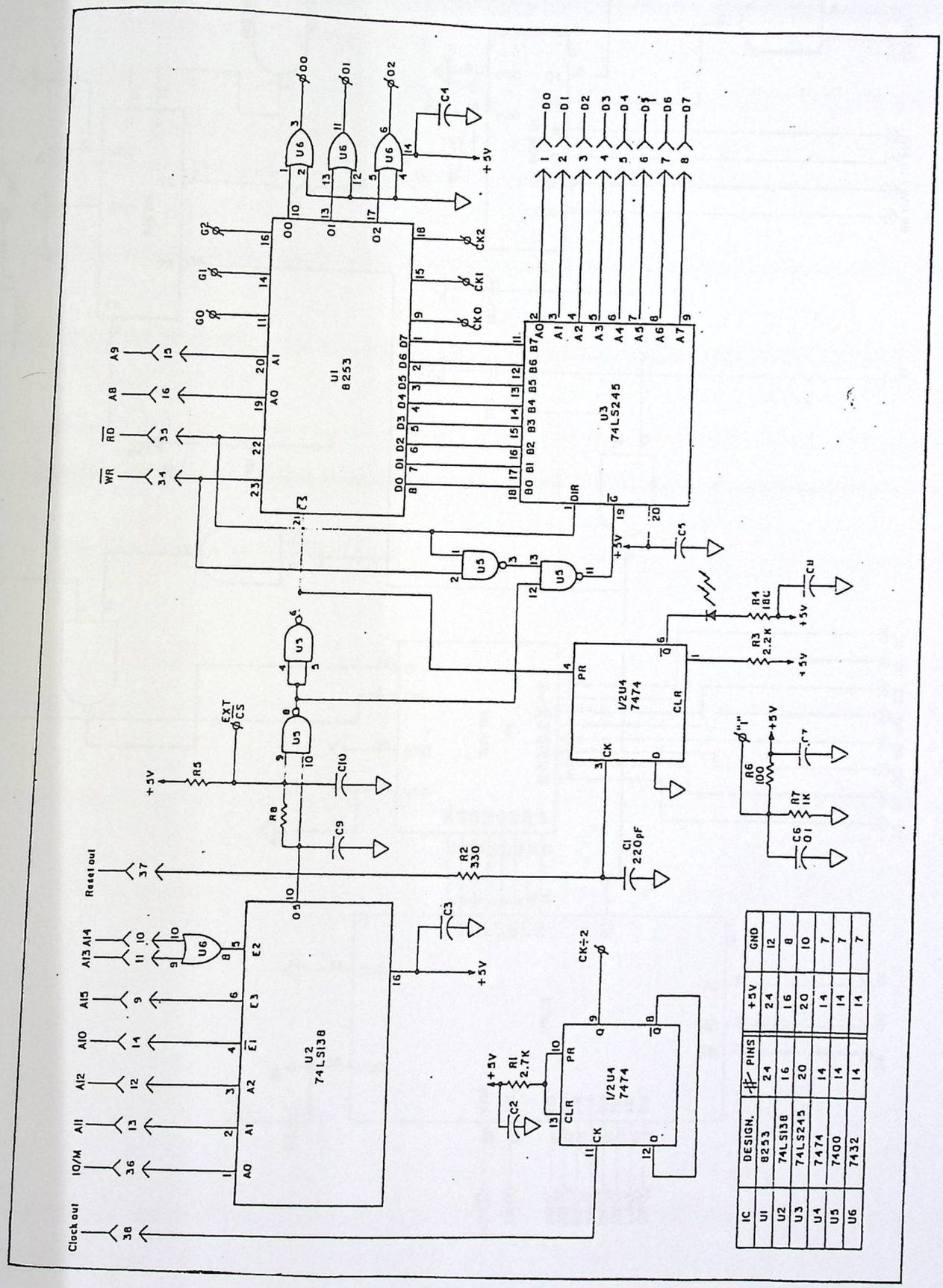
```
LXI SP, 20B0       ; initialize stack pointer
MVI A, 08          ; Prepare mask
SIM                ; Set interrupt mask
EI                 ; Enable interrupts
HLT                ; wait for interrupt
JMP 2002           ; Jump back
DI                 ; Disable interrupt
MVI A, 65
CALL UPDDT         ; Display 65 in the data field
RET                ; return to address 2008
DI                 Disable interrupts
Call UPDDT
RET                ; return to the address 2008
JNP 200B           ; Jump to RST 6.5 routine
JMP 2012           ; Jump to RST 7.5 routine
```

5.3 PPI Design Circuit

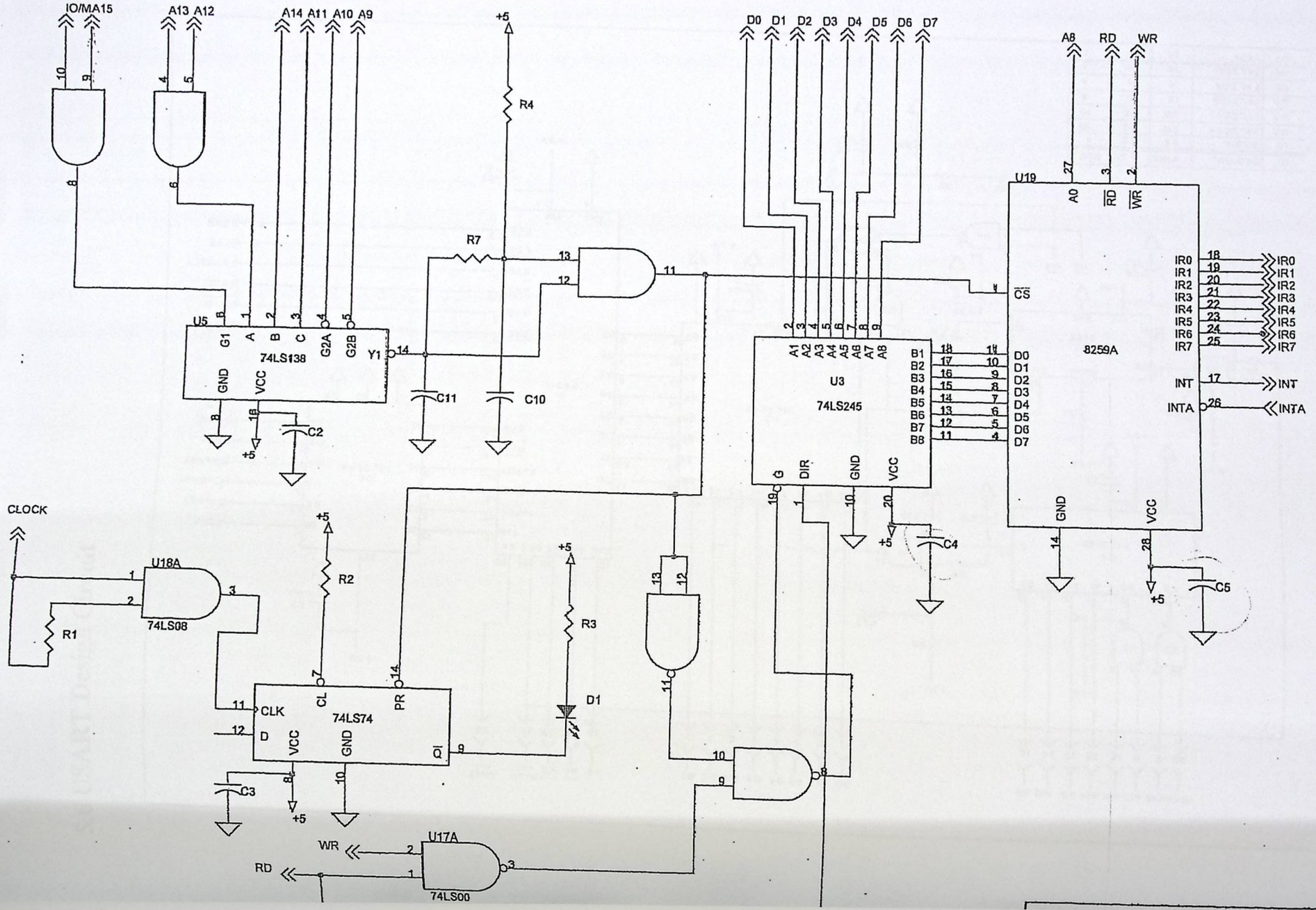


IC	DESIG.	# PINS	+5 V	GND
U1	8255	40	26	7
U2	74LS245	20	20	10
U3	74LS138	16	16	8
U4	7474	14	14	7
U5	74LS08	14	14	7
U6	74LS32	14	14	7

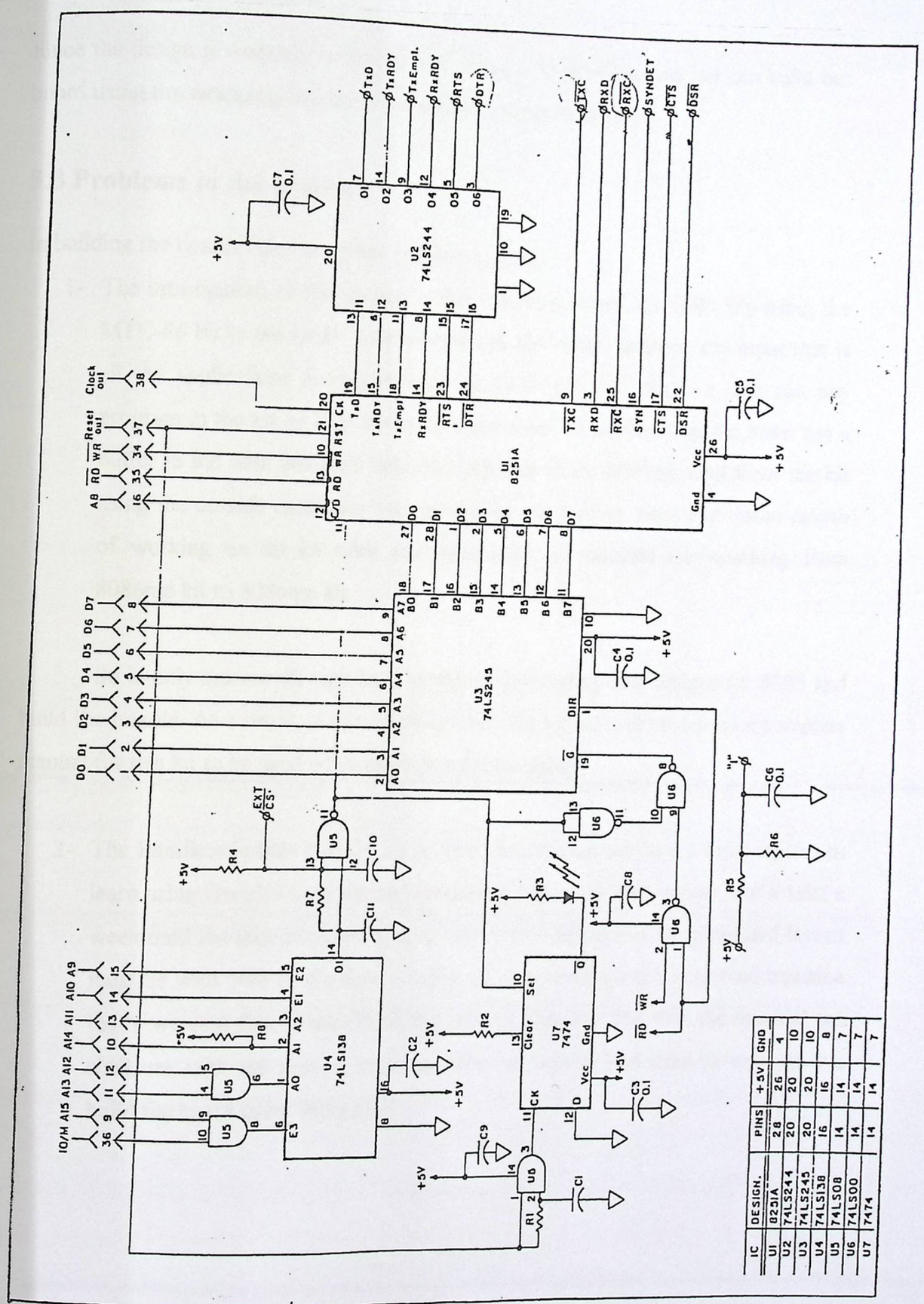
5.4 PIT Design Circuit



IC	DESIGN.	PINS	+5V	GND
U1	8253	24	24	12
U2	74LS138	16	16	8
U3	74LS245	20	20	10
U4	7474	14	14	7
U5	7400	14	14	7
U6	7432	14	14	7


IO/MA15
A13 A12
A14 A11 A10 A9
+5
R4
D0 D1 D2 D3 D4 D5 D6 D7
A8 RD WR
U19
A0
RD
WR
R7
U5
G1
A
B
C
G2A
G2B
74LS138
Y1
GND
VCC
C11
C10
C2
CS
8259A
IR0
IR1
IR2
IR3
IR4
IR5
IR6
IR7
INT
INTA
U3
74LS245
A1
A2
A3
A4
A5
A6
A7
A8
B1
B2
B3
B4
B5
B6
B7
B8
D0
D1
D2
D3
D4
D5
D6
D7
G
DIR
GND
VCC
C4
C5
CLOCK
U18A
74LS08
R1
R2
R3
CL
PR
CLK
D
Q
74LS74
D1
C3
U17A
WR
RD
74LS00

5.6 USART Design Circuit



5.7 The Implementation

Since the design is working as desired and there is no problem now we can build our board using the wrapping circuits to be ready for using in the lab.

5.8 Problems in the project

In building the boards there are many problems appear: -

1- The introduction of this project is Experimental board for 8086 Mp using the MTC-86 kit in the LAB. And we build all the circuit without any input that is all the applications is output until we reach the PIC that we cant run any program in the kit so after the testing and research we find that the 86kit has a buffer in the data bus with unknown address so no one can read from the kit using the outside connector we use during our project, thus after three month of working on 86 kit with our supervisor we convert our working from 8086mp kit to 8085mp kit.

So in only one month with hard working we develop new designs for 8085 and build the boards. As a result of our working with this kit we will introduce a complete manual for this kit to be used when necessary for teaching.

2- The Interface boards have to be printed circuits first we face a big problem to learn using Orcad 9 how we can generate a layout for each circuit and it take a week until the idea complete and we generate a layout but the generated layout must be with bold or big font in order to be printed using the printed machine that's we cant find in the Orcad the second thing we find that the material we shall use with the printed machine have be expired and cant be used so we build the board using Wrapping.

5-9 Future work

1- Build an Interfacing board for the DMA with 8085MP kit to be familiar with the direct memory access concept.
2- Print the interfacing board into printed circuit to be very efficient in LAB. experiments.

5-10 Conclusions

This project comes as an introduction to microprocessors and microprocessor based systems so coverage of microprocessor system architecture, instruction sets Addressing modes, system timing, software design, assembly language programming, interrupts, and interfacing concepts for memory and input/output systems.

We introduce four devices with their interfacing with 8085MP kit and an applications for each devices which are PPI,PIT,PIC, and USART .Finally we hope that theses boards will help the student to understand the interfacing concepts.

DATA SHEETS

E2O0017-27-X2

OKI Semiconductor

This version: Jan. 1998
Previous version: Aug. 1996

MSM82C51A-2RS/GS/JS

UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER

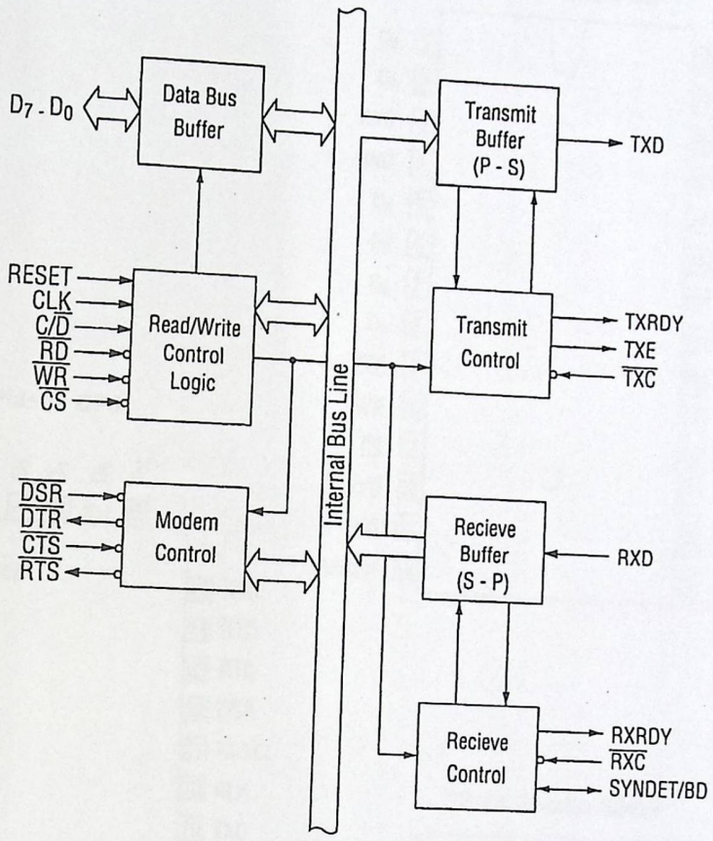
GENERAL DESCRIPTION

The MSM82C51A-2 is a USART (Universal Synchronous Asynchronous Receiver Transmitter) for serial data communication.
As a peripheral device of a microcomputer system, the MSM82C51A-2 receives parallel data from the CPU and transmits serial data after conversion. This device also receives serial data from the outside and transmits parallel data to the CPU after conversion.
The MSM82C51A-2 configures a fully static circuit using silicon gate CMOS technology. Therefore, it operates on extremely low power at 100 µA (max) of standby current by suspending all operations.

FEATURES

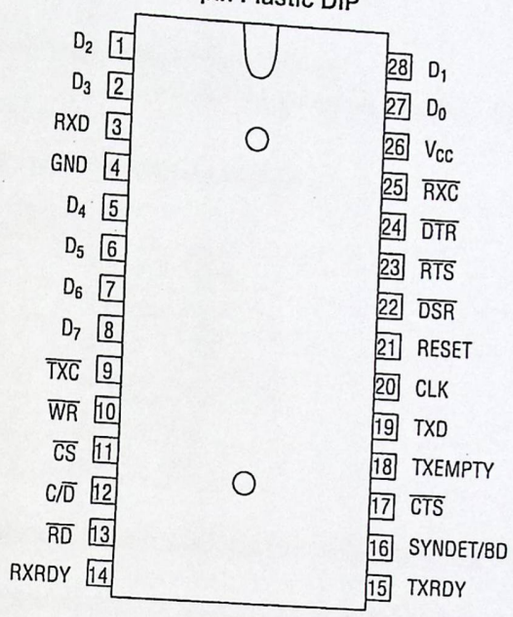
- Wide power supply voltage range from 3 V to 6 V
- Wide temperature range from –40°C to 85°C
- Synchronous communication upto 64 Kbaud
- Asynchronous communication upto 38.4 Kbaud
- Transmitting/receiving operations under double buffered configuration.
- Error detection (parity, overrun and framing)
- 28-pin Plastic DIP (DIP28-P-600-2.54): (Product name: MSM82C51A-2RS)
- 28-pin Plastic QFJ (QFJ28-P-S450-1.27): (Product name: MSM82C51A-2JS)
- 32-pin Plastic SSOP(SSOP32-P-430-1.00-K): (Product name: MSM82C51A-2GS-K)

FUNCTIONAL BLOCK DIAGRAM

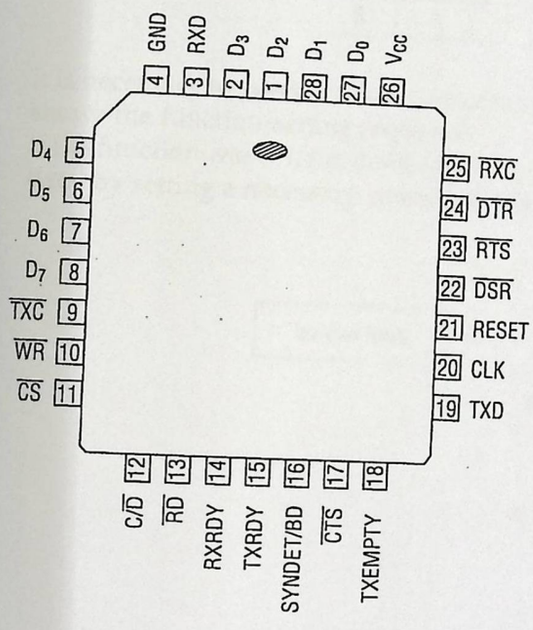


PIN CONFIGURATION (TOP VIEW)

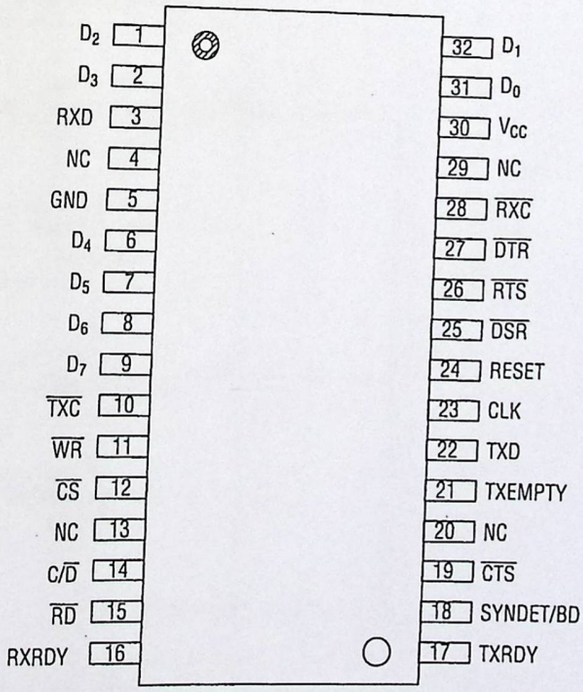
28 pin Plastic DIP



28 pin Plastic QFJ



32 pin Plastic SSOP



FUNCTION

Outline

The MSM82C51A-2's functional configuration is programed by software.
Operation between the MSM82C51A-2 and a CPU is executed by program control. Table 1 shows the operation between a CPU and the device.

Table 1 Operation between MSM82C51A and CPU

$\overline{CS}$	C/$\overline{D}$	$\overline{RD}$	$\overline{WR}$	
1	×	×	×	Data Bus 3-State
0	×	1	1	Data Bus 3-State
0	1	0	1	Status → CPU
0	1	1	0	Control Word ← CPU
0	0	0	1	Data → CPU
0	0	1	0	Data ← CPU

It is necessary to execute a function-setting sequence after resetting the MSM82C51A-2. Fig. 1 shows the function-setting sequence.
If the function was set, the device is ready to receive a command, thus enabling the transfer of data by setting a necessary command, reading a status and reading/writing data.

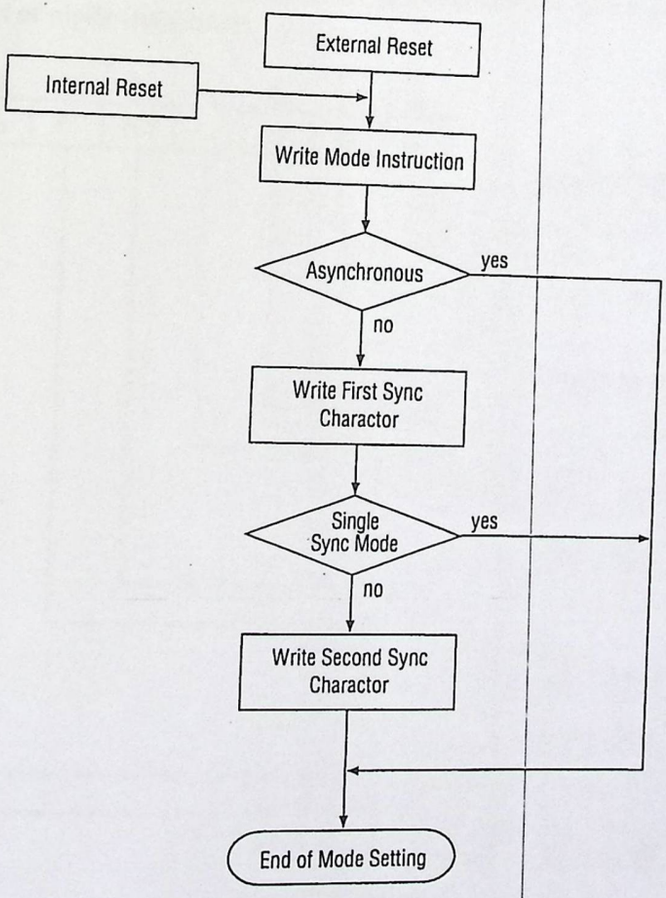


Fig. 1 Function-setting Sequence (Mode Instruction Sequence)

Control Words

There are two types of control word.
1. Mode instruction (setting of function)
2. Command (setting of operation)

1) Mode Instruction

Mode instruction is used for setting the function of the MSM82C51A-2. Mode instruction will be in "wait for write" at either internal reset or external reset. That is, the writing of a control word after resetting will be recognized as a "mode instruction."
Items set by mode instruction are as follows:

- Synchronous/asynchronous mode
- Stop bit length (asynchronous mode)
- Character length
- Parity bit
- Baud rate factor (asynchronous mode)
- Internal/external synchronization (synchronous mode)
- Number of synchronous characters (Synchronous mode)

The bit configuration of mode instruction is shown in Figures 2 and 3. In the case of synchronous mode, it is necessary to write one-or two byte sync characters.
If sync characters were written, a function will be set because the writing of sync characters constitutes part of mode instruction.

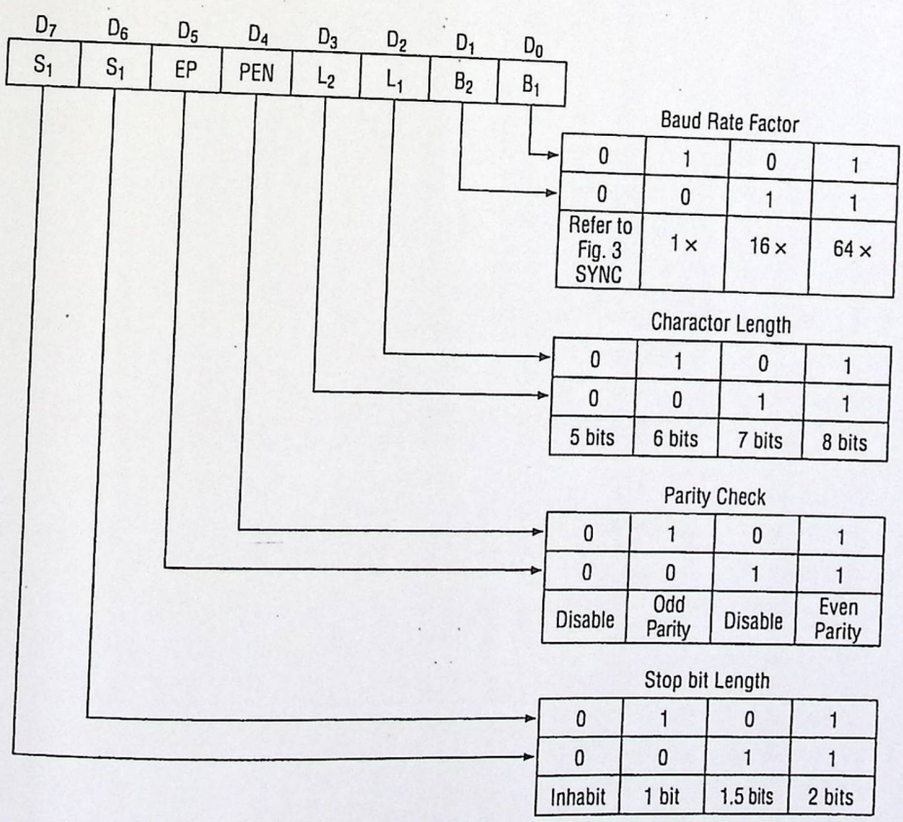


Fig. 2 Bit Configuration of Mode Instruction (Asynchronous)

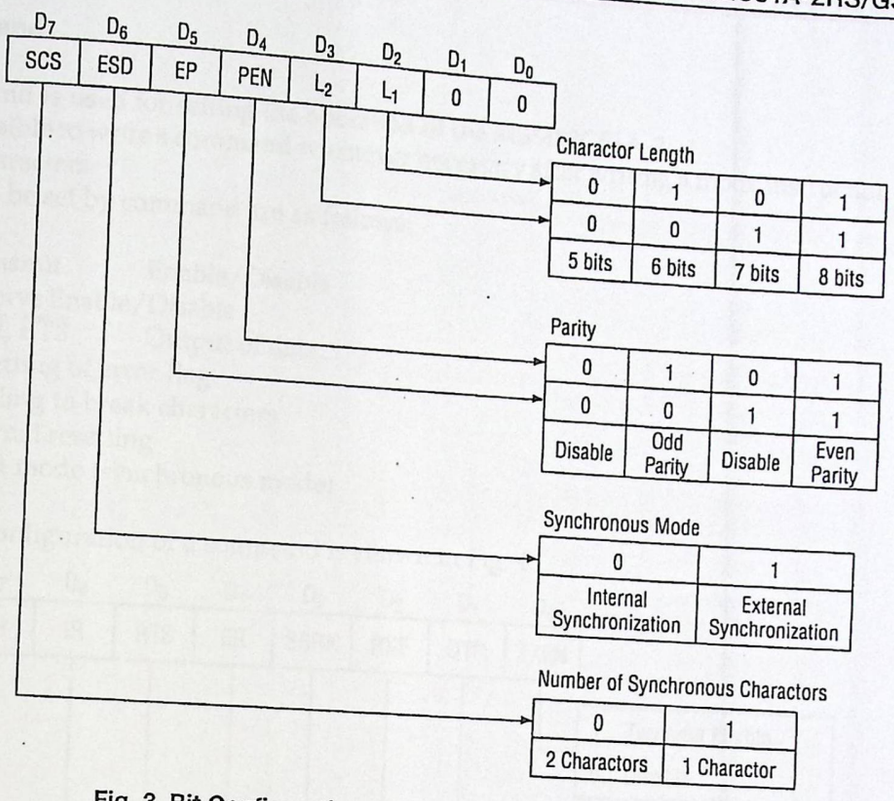


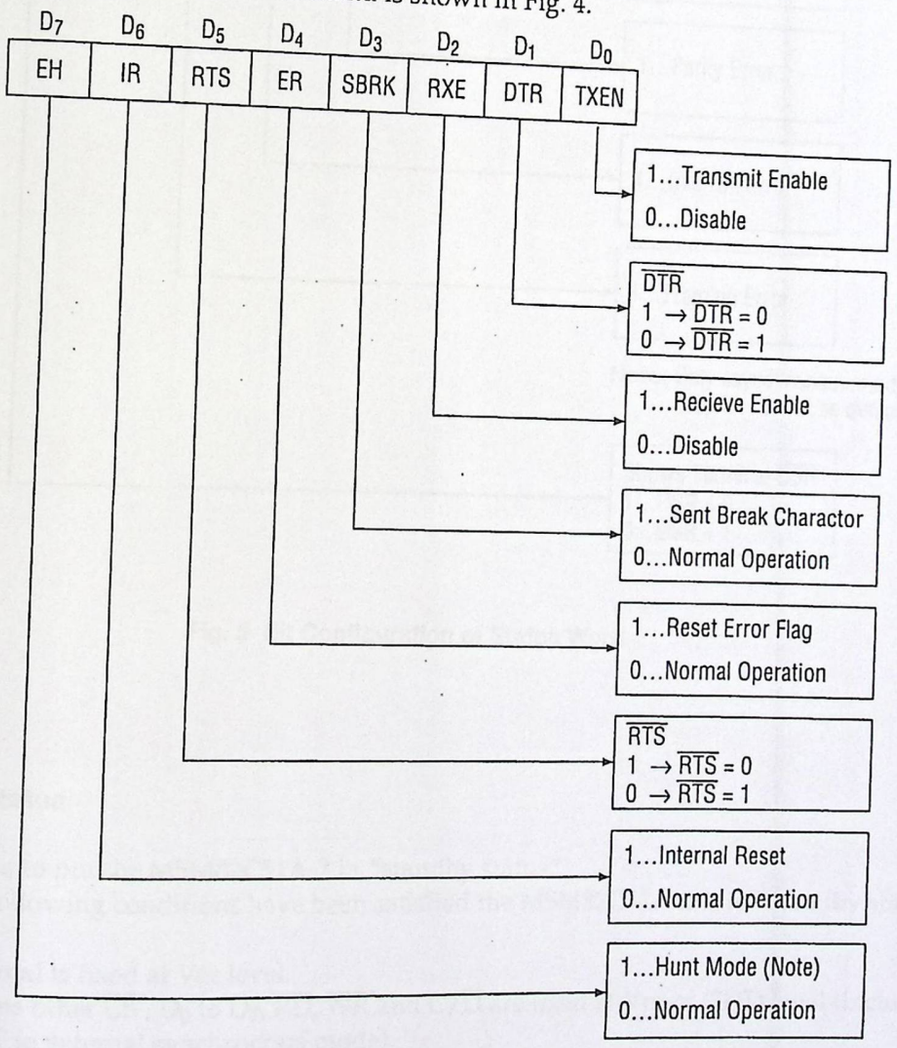
Fig. 3 Bit Configuration of Mode Instruction (Synchronous)

2) Command

Command is used for setting the operation of the MSM82C51A-2.
It is possible to write a command whenever necessary after writing a mode instruction and sync characters.
Items to be set by command are as follows:

- Transmit Enable/Disable
- Receive Enable/Disable
- $\overline{DTR}$, $\overline{RTS}$ Output of data.
- Resetting of error flag.
- Sending to break characters
- Internal resetting
- Hunt mode (synchronous mode)

The bit configuration of a command is shown in Fig. 4.



Note: Seach mode for synchronous charactors in synchronous mode.

Fig. 4 Bit Configuration of Command

Status Word

It is possible to see the internal status of MSM82C51A-2 by reading a status word.
The bit configuration of status word is shown in Fig. 5.

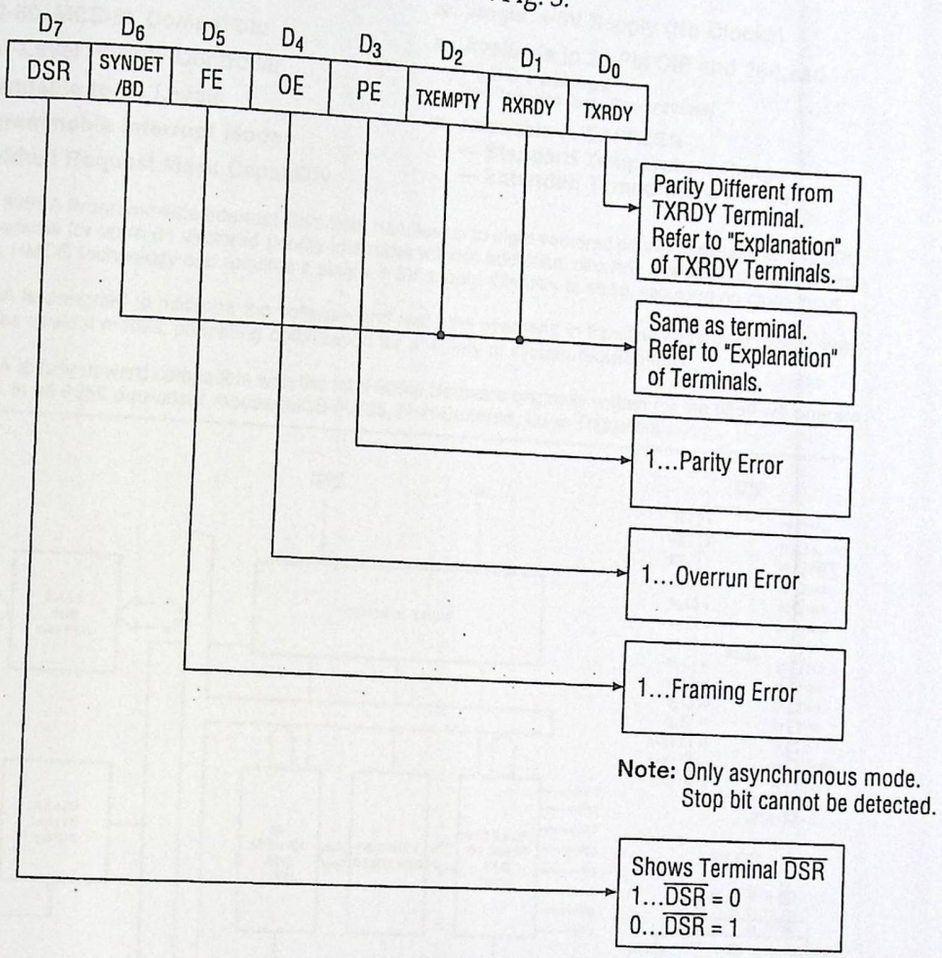


Fig. 5 Bit Configuration of Status Word

Standby Status

It is possible to put the MSM82C51A-2 in "standby status"
When the following conditions have been satisfied the MSM82C51A-2 is in "standby status."

(1) $\overline{CS}$ terminal is fixed at Vcc level.
(2) Input pins other $\overline{CS}$, D_0 to D_7, $\overline{RD}$, $\overline{WR}$ and C/$\overline{D}$ are fixed at Vcc or GND level (including SYNDET in external synchronous mode).

Note: When all output currents are 0, ICCS specification is applied.

intel

8259A
PROGRAMMABLE INTERRUPT CONTROLLER
(8259A/8259A-2)

- 8086, 8088 Compatible
- MCS-80, MCS-85 Compatible
- Eight-Level Priority Controller
- Expandable to 64 Levels
- Programmable Interrupt Modes
- Individual Request Mask Capability
- Single +5V Supply (No Clocks)
- Available in 28-Pin DIP and 28-Lead PLCC Package
  (See Packaging Spec., Order #231369)
- Available in EXPRESS
  — Standard Temperature Range
  — Extended Temperature Range

The Intel 8259A Programmable Interrupt Controller handles up to eight vectored priority interrupts for the CPU. It is cascadable for up to 64 vectored priority interrupts without additional circuitry. It is packaged in a 28-pin DIP, uses NMOS technology and requires a single +5V supply. Circuitry is static, requiring no clock input.

The 8259A is designed to minimize the software and real time overhead in handling multi-level priority interrupts. It has several modes, permitting optimization for a variety of system requirements.

The 8259A is fully upward compatible with the Intel 8259. Software originally written for the 8259 will operate the 8259A in all 8259 equivalent modes (MCS-80/85, Non-Buffered, Edge Triggered).

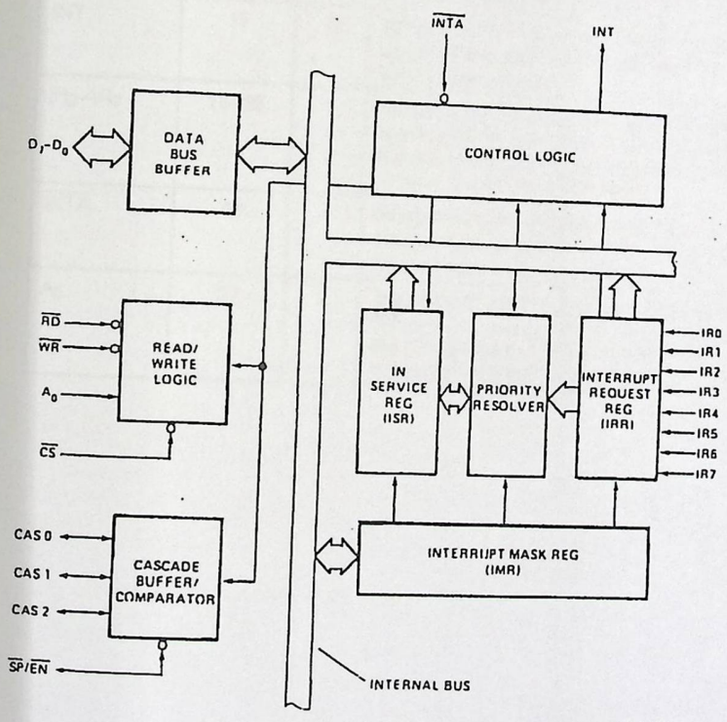


Figure 1. Block Diagram

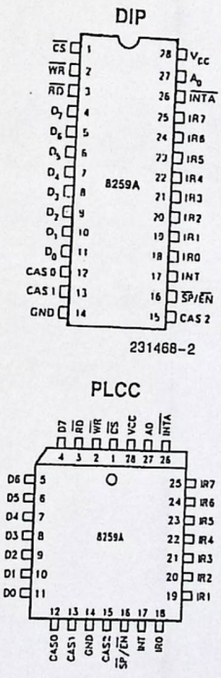


Figure 2. Pin Configurations

intel.

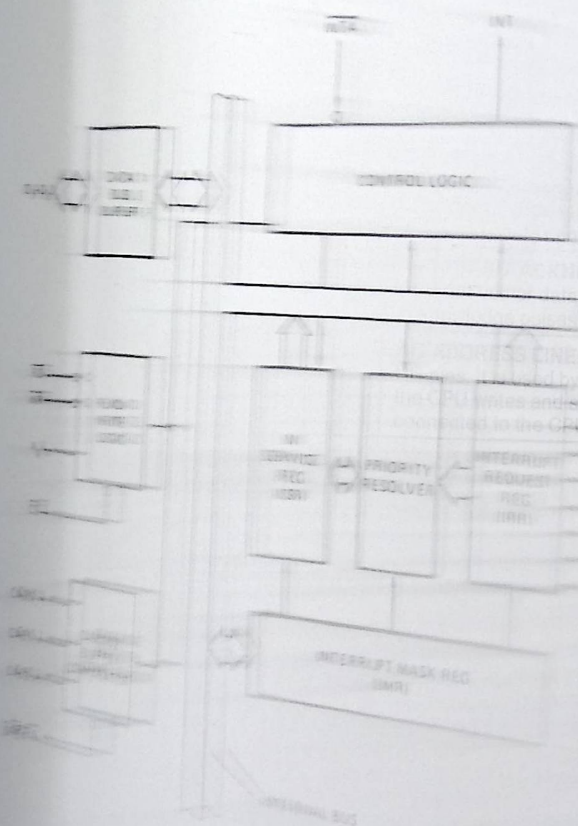
8259A PROGRAMMABLE INTERRUPT CONTROLLER (8259A/8259A-2)

- 8086, 8088 Compatible
- MCS-80, MCS-85 Compatible
- Eight-Level Priority Controller
- Expandable to 64 Levels
- Programmable Interrupt Modes
- Individual Request Mask Capability
- Single +5V Supply (No Clocks)
- Available in 28-Pin DIP and 28-Lead PLCC Package
  (See Packaging Spec., Order #231369)
- Available in EXPRESS
  - Standard Temperature Range
  - Extended Temperature Range

The Intel 8259A Programmable Interrupt Controller handles up to eight vectored priority interrupts [illegible] It is cascadable for up to 64 vectored priority interrupts without additional circuitry. It is [illegible] DIP, uses NMOS technology and requires a single +5V supply. Circuitry is static, requiring [illegible]

The 8259A is designed to minimize the software and real time overhead in handling multi-level [illegible] rupts. It has several modes, permitting optimization for a variety of system requirements.

The 8259A is fully upward compatible with the Intel 8259. Software originally written for the [illegible] the 8259A in all 8259 equivalent modes (MCS-80/85, Non-Buffered, Edge Triggered).



[illegible]	[illegible] control [illegible] 8259A
[illegible]	[illegible] function pin; [illegible] to control [illegible] it is used as [illegible] (SP = 0).
[illegible]	[illegible] interrupt request is [illegible] connected to the [illegible]
[illegible]	[illegible] inputs. An interrupt request [illegible] (low to high), and holding it high until [illegible] (Edge Triggered Mode), or just by a high level on an [illegible] Mode).
[illegible]	[illegible] ACKNOWLEDGE: This pin is used to enable 8259A [illegible] onto the data bus by a sequence of interrupt [illegible] issued by the CPU.
[illegible]	ADDRESS LINE: This pin acts in conjunction with the $\overline{CS}$, $\overline{WR}$, and [illegible] used by the 8259A to decipher various Command Words the CPU writes and status the CPU wishes to read. It is typically connected to the CPU A0 address line (A1 for 8086, 8088).

Figure 1. Block Diagram

Table 1. Pin Description

Symbol	Pin No.	Type	Name and Function
V_{CC}	28	I	**SUPPLY:** +5V Supply.
GND	14	I	**GROUND**
$\overline{CS}$	1	I	**CHIP SELECT:** A low on this pin enables $\overline{RD}$ and $\overline{WR}$ communication between the CPU and the 8259A. INTA functions are independent of CS.
$\overline{WR}$	2	I	**WRITE:** A low on this pin when CS is low enables the 8259A to accept command words from the CPU.
$\overline{RD}$	3	I	**READ:** A low on this pin when CS is low enables the 8259A to release status onto the data bus for the CPU.
D_7–D_0	4–11	I/O	**BIDIRECTIONAL DATA BUS:** Control, status and interrupt-vector information is transferred via this bus.
CAS_0–CAS_2	12, 13, 15	I/O	**CASCADE LINES:** The CAS lines form a private 8259A bus to control a multiple 8259A structure. These pins are outputs for a master 8259A and inputs for a slave 8259A.
$\overline{SP}/\overline{EN}$	16	I/O	**SLAVE PROGRAM/ENABLE BUFFER:** This is a dual function pin. When in the Buffered Mode it can be used as an output to control buffer transceivers (EN). When not in the buffered mode it is used as an input to designate a master (SP = 1) or slave (SP = 0).
INT	17	O	**INTERRUPT:** This pin goes high whenever a valid interrupt request is asserted. It is used to interrupt the CPU, thus it is connected to the CPU's interrupt pin.
IR_0–IR_7	18–25	I	**INTERRUPT REQUESTS:** Asynchronous inputs. An interrupt request is executed by raising an IR input (low to high), and holding it high until it is acknowledged (Edge Triggered Mode), or just by a high level on an IR input (Level Triggered Mode).
$\overline{INTA}$	26	I	**INTERRUPT ACKNOWLEDGE:** This pin is used to enable 8259A interrupt-vector data onto the data bus by a sequence of interrupt acknowledge pulses issued by the CPU.
A_0	27	I	**A0 ADDRESS LINE:** This pin acts in conjunction with the $\overline{CS}$, $\overline{WR}$, and $\overline{RD}$ pins. It is used by the 8259A to decipher various Command Words the CPU writes and status the CPU wishes to read. It is typically connected to the CPU A0 address line (A1 for 8086, 8088).

FUNCTIONAL DESCRIPTION

Interrupts in Microcomputer Systems

Microcomputer system design requires that I.O devices such as keyboards, displays, sensors and other components receive servicing in a an efficient manner so that large amounts of the total system tasks can be assumed by the microcomputer with little or no effect on throughput.

The most common method of servicing such devices is the *Polled* approach. This is where the processor must test each device in sequence and in effect "ask" each one if it needs servicing. It is easy to see that a large portion of the main program is looping through this continuous polling cycle and that such a method would have a serious detrimental effect on system throughput, thus limiting the tasks that could be assumed by the microcomputer and reducing the cost effectiveness of using such devices.

A more desirable method would be one that would allow the microprocessor to be executing its main program and only stop to service peripheral devices when it is told to do so by the device itself. In effect, the method would provide an external asynchronous input that would inform the processor that it should complete whatever instruction that is currently being executed and fetch a new routine that will service the requesting device. Once this servicing is complete, however, the processor would resume exactly where it left off.

This method is called *Interrupt*. It is easy to see that system throughput would drastically increase, and thus more tasks could be assumed by the microcomputer to further enhance its cost effectiveness.

The Programmable Interrupt Controller (PIC) functions as an overall manager in an Interrupt-Driven system environment. It accepts requests from the peripheral equipment, determines which of the incoming requests is of the highest importance (priority), ascertains whether the incoming request has a higher priority value than the level currently being serviced, and issues an interrupt to the CPU based on this determination.

Each peripheral device or structure usually has a special program or "routine" that is associated with its specific functional or operational requirements; this is referred to as a "service routine". The PIC, after issuing an Interrupt to the CPU, must somehow input information into the CPU that can "point" the Program Counter to the service routine associated with the requesting device. This "pointer" is an address in a vectoring table and will often be referred to, in this document, as vectoring data.

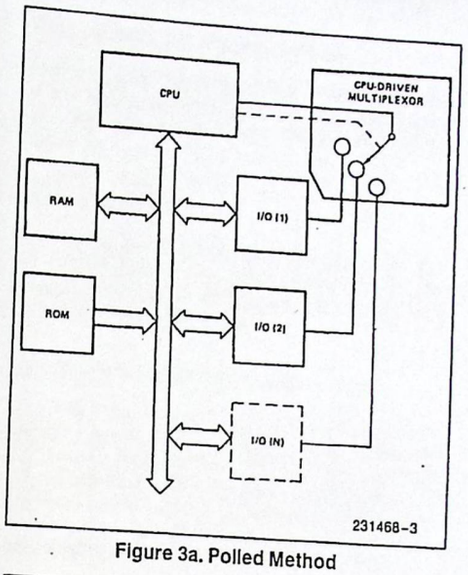


Figure 3a. Polled Method

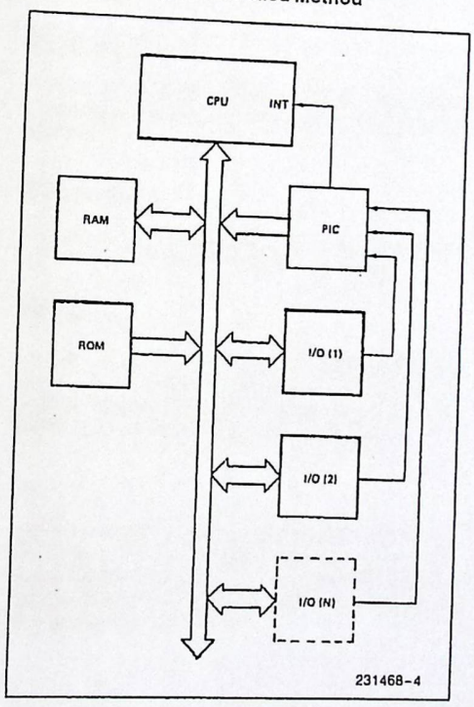


Figure 3b. Interrupt Method

The 8259A is a device specifically designed for use in real time, interrupt driven microcomputer systems. It manages eight levels or requests and has built-in features for expandability to other 8259A's (up to 64 levels). It is programmed by the system's software as an I/O peripheral. A selection of priority modes is available to the programmer so that the manner in which the requests are processed by the 8259A can be configured to match his system requirements. The priority modes can be changed or reconfigured dynamically at any time during the main program. This means that the complete interrupt structure can be defined as required, based on the total system environment.

INTERRUPT REQUEST REGISTER (IRR) AND IN-SERVICE REGISTER (ISR)

The interrupts at the IR input lines are handled by two registers in cascade, the Interrupt Request Register (IRR) and the In-Service (ISR). The IRR is used to store all the interrupt levels which are requesting service; and the ISR is used to store all the interrupt levels which are being serviced.

PRIORITY RESOLVER

This logic block determines the priorites of the bits set in the IRR. The highest priority is selected and strobed into the corresponding bit of the ISR during $\overline{INTA}$ pulse.

INTERRUPT MASK REGISTER (IMR)

The IMR stores the bits which mask the interrupt lines to be masked. The IMR operates on the IRR. Masking of a higher priority input will not affect the interrupt request lines of lower quality.

INT (INTERRUPT)

This output goes directly to the CPU interrupt input. The V_{OH} level on this line is designed to be fully compatible with the 8080A, 8085A and 8086 input levels.

$\overline{INTA}$ (INTERRUPT ACKNOWLEDGE)

$\overline{INTA}$ pulses will cause the 8259A to release vectoring information onto the data bus. The format of this data depends on the system mode (μPM) of the 8259A.

DATA BUS BUFFER

This 3-state, bidirectional 8-bit buffer is used to interface the 8259A to the system Data Bus. Control words and status information are transferred through the Data Bus Buffer.

READ/WRITE CONTROL LOGIC

The function of this block is to accept OUTput commands from the CPU. It contains the Initialization Command Word (ICW) registers and Operation Command Word (OCW) registers which store the various control formats for device operation. This function block also allows the status of the 8259A to be transferred onto the Data Bus.

$\overline{CS}$ (CHIP SELECT)

A LOW on this input enables the 8259A. No reading or writing of the chip will occur unless the device is selected.

$\overline{WR}$ (WRITE)

A LOW on this input enables the CPU to write control words (ICWs and OCWs) to the 8259A.

$\overline{RD}$ (READ)

A LOW on this input enables the 8259A to send the status of the Interrupt Request Register (IRR), In Service Register (ISR), the Interrupt Mask Register (IMR), or the Interrupt level onto the Data Bus.

A_0

This input signal is used in conjunction with $\overline{WR}$ and $\overline{RD}$ signals to write commands into the various command registers, as well as reading the various status registers of the chip. This line can be tied directly to one of the address lines.

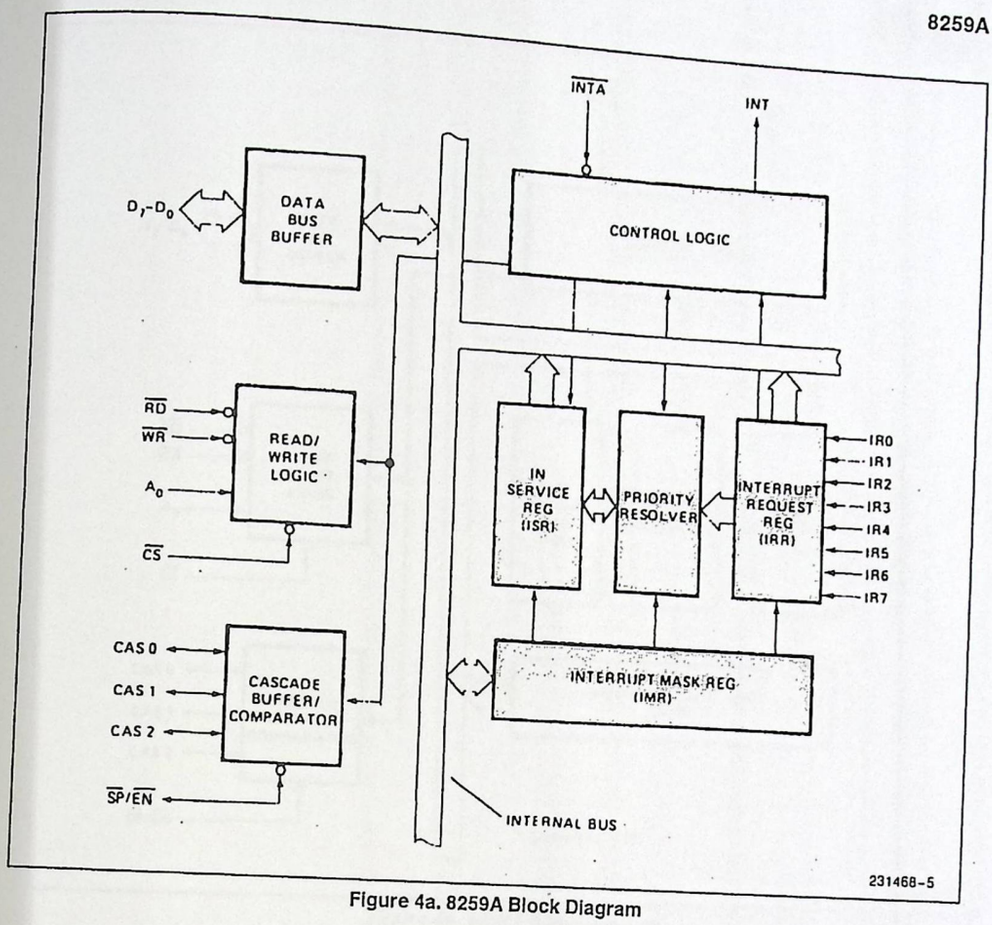


Figure 4a. 8259A Block Diagram

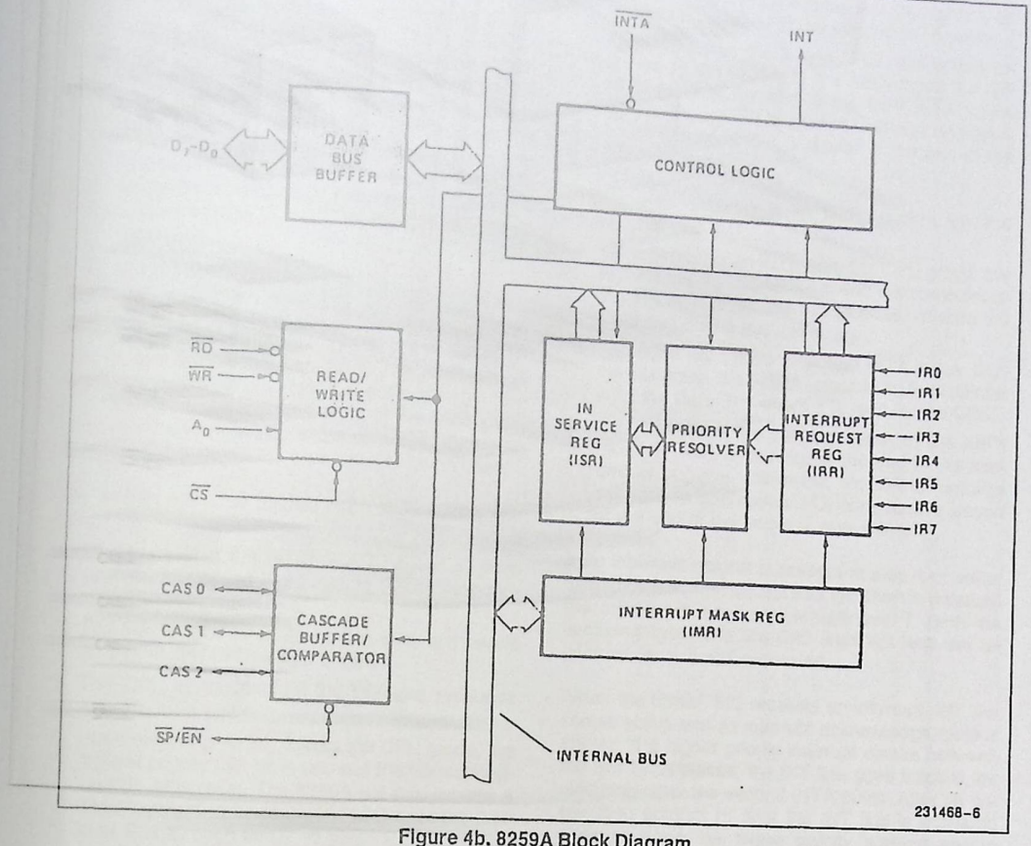


Figure 4b. 8259A Block Diagram

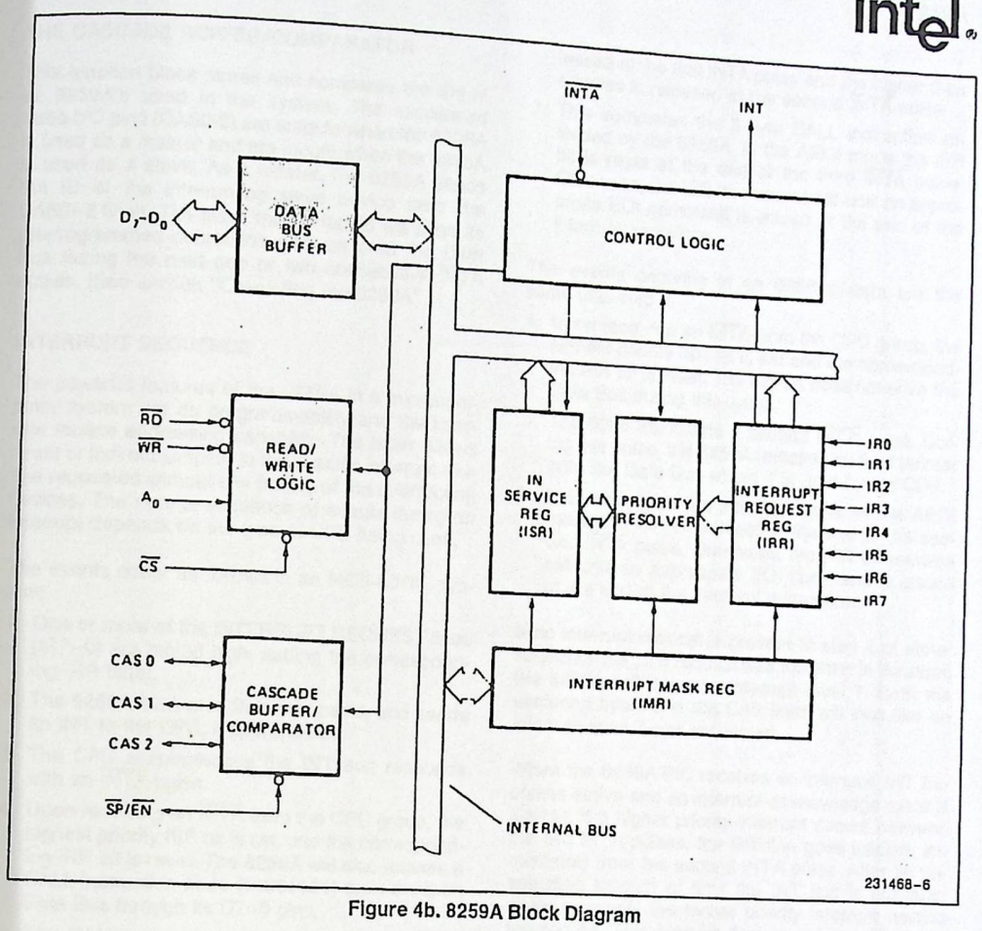


Figure 4b. 8259A Block Diagram

THE CASCADE BUFFER/COMPARATOR

This function block stores and compares the IDs of all 8259A's used in the system. The associated three I/O pins (CAS0-2) are outputs when the 8259A is used as a master and are inputs when the 8259A is used as a slave. As a master, the 8259A sends the ID of the interrupting slave device onto the CAS0–2 lines. The slave thus selected will send its preprogrammed subroutine address onto the Data Bus during the next one or two consecutive $\overline{INTA}$ pulses. (See section "Cascading the 8259A".)

INTERRUPT SEQUENCE

The powerful features of the 8259A in a microcomputer system are its programmability and the interrupt routine addressing capability. The latter allows direct or indirect jumping to the specific interrupt routine requested without any polling of the interrupting devices. The normal sequence of events during an interrupt depends on the type of CPU being used.

The events occur as follows in an MCS-80/85 system:

1. One or more of the INTERRUPT REQUEST lines (IR7–0) are raised high, setting the corresponding IRR bit(s).
2. The 8259A evaluates these requests, and sends an INT to the CPU, if appropriate.
3. The CPU acknowledges the INT and responds with an $\overline{INTA}$ pulse.
4. Upon receiving an $\overline{INTA}$ from the CPU group, the highest priority ISR bit is set, and the corresponding IRR bit is reset. The 8259A will also release a CALL instruction code (11001101) onto the 8-bit Data Bus through its D7–0 pins.
5. This CALL instruction will initiate two more $\overline{INTA}$ pulses to be sent to the 8259A from the CPU group.
6. These two $\overline{INTA}$ pulses allow the 8259A to release its preprogrammed subroutine address onto the Data Bus. The lower 8-bit address is released at the first $\overline{INTA}$ pulse and the higher 8-bit address is released at the second $\overline{INTA}$ pulse.
7. This completes the 3-byte CALL instruction released by the 8259A. In the AEOI mode the ISR bit is reset at the end of the third $\overline{INTA}$ pulse. Otherwise, the ISR bit remains set until an appropriate EOI command is issued at the end of the interrupt sequence.

The events occuring in an 8086 system are the same until step 4.

4. Upon receiving an $\overline{INTA}$ from the CPU group, the highest priority ISR bit is set and the corresponding IRR bit is reset. The 8259A does not drive the Data Bus during this cycle.
5. The 8086 will initiate a second $\overline{INTA}$ pulse. During this pulse, the 8259A releases an 8-bit pointer onto the Data Bus where it is read by the CPU.
6. This completes the interrupt cycle. In the AEOI mode the ISR bit is reset at the end of the second $\overline{INTA}$ pulse. Otherwise, the ISR bit remains set until an appropriate EOI command is issued at the end of the interrupt subroutine.

If no interrupt request is present at step 4 of either sequence (i.e., the request was too short in duration) the 8259A will issue an interrupt level 7. Both the vectoring bytes and the CAS lines will look like an interrupt level 7 was requested.

When the 8259A PIC receives an interrupt, INT becomes active and an interrupt acknowledge cycle is started. If a higher priority interrupt occurs between the two INTA pulses, the INT line goes inactive immediately after the second INTA pulse. After an unspecified amount of time the INT line is activated again to signify the higher priority interrupt waiting for service. This inactive time is not specified and can vary between parts. The designer should be aware of this consideration when designing a system which uses the 8259A. It is recommended that proper asynchronous design techniques be followed.

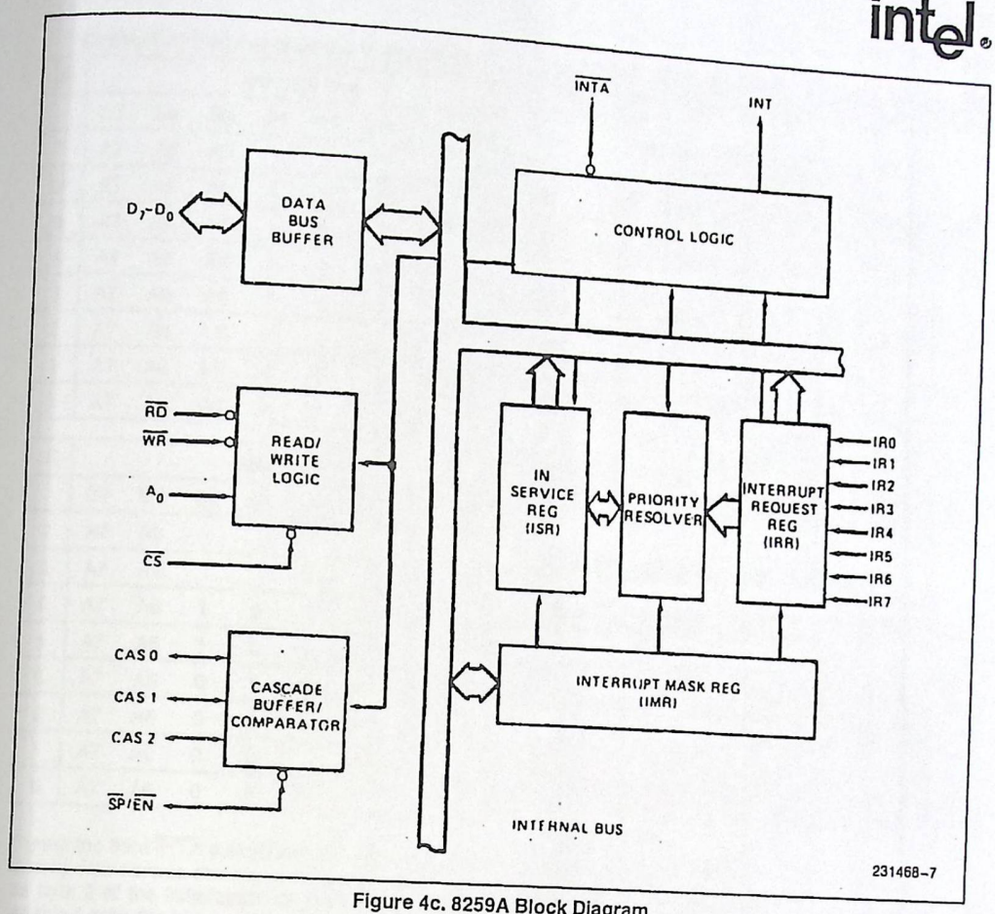


Figure 4c. 8259A Block Diagram

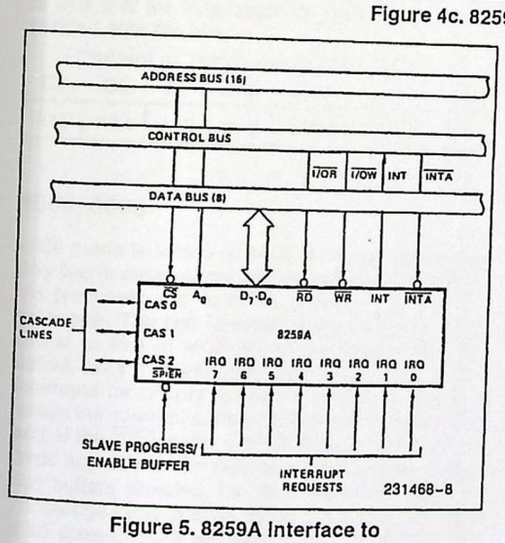


Figure 5. 8259A Interface to Standard System Bus

INTERRUPT SEQUENCE OUTPUTS

MCS-80, MCS-85

This sequence is timed by three $\overline{INTA}$ pulses. During the first $\overline{INTA}$ pulse the CALL opcode is enabled onto the data bus.

Content of First Interrupt Vector Byte

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

During the second $\overline{INTA}$ pulse the lower address of the appropriate service routine is enabled onto the data bus. When Interval = 4 bits A_5–A_7 are programmed, while A_0–A_4 are automatically inserted by the 8259A. When Interval = 8 only A_6 and A_7 are programmed, while A_0–A_5 are automatically inserted.

Content of Second Interrupt Vector Byte

IR	Interval = 4							
	D7	D6	D5	D4	D3	D2	D1	D0
7	A7	A6	A5	1	1	1	0	0
6	A7	A6	A5	1	1	0	0	0
5	A7	A6	A5	1	0	1	0	0
4	A7	A6	A5	1	0	0	0	0
3	A7	A6	A5	0	1	1	0	0
2	A7	A6	A5	0	1	0	0	0
1	A7	A6	A5	0	0	1	0	0
0	A7	A6	A5	0	0	0	0	0

IR	Interval = 8							
	D7	D6	D5	D4	D3	D2	D1	D0
7	A7	A6	1	1	1	0	0	0
6	A7	A6	1	1	0	0	0	0
5	A7	A6	1	0	1	0	0	0
4	A7	A6	1	0	0	0	0	0
3	A7	A6	0	1	1	0	0	0
2	A7	A6	0	1	0	0	0	0
1	A7	A6	0	0	1	0	0	0
0	A7	A6	0	0	0	0	0	0

During the third $\overline{INTA}$ pulse the higher address of the appropriate service routine, which was programmed as byte 2 of the initialization sequence (A_8–A_{15}), is enabled onto the bus.

Content of Third Interrupt Vector Byte

D7	D6	D5	D4	D3	D2	D1	D0
A15	A14	A13	A12	A11	A10	A9	A8

8086, 8088

8086 mode is similar to MCS-80 mode except that only two Interrupt Acknowledge cycles are issued by the processor and no CALL opcode is sent to the processor. The first interrupt acknowledge cycle is similar to that of MCS-80, 85 systems in that the 8259A uses it to internally freeze the state of the interrupts for priority resolution and as a master it issues the interrupt code on the cascade lines at the end of the $\overline{INTA}$ pulse. On this first cycle it does not issue any data to the processor and leaves its data bus buffers disabled. On the second interrupt acknowledge cycle in 8086 mode the master (or slave if so programmed) will send a byte of data to the processor with the acknowledged interrupt code composed as follows (note the state of the ADI mode control is ignored and A_5–A_{11} are unused in 8086 mode):

Content of Interrupt Vector Byte for 8086 System Mode

	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

PROGRAMMING THE 8259A

The 8259A accepts two types of command words generated by the CPU:

1. *Initialization Command Words (ICWs):* Before normal operation can begin, each 8259A in the system must be brought to a starting point—by a sequence of 2 to 4 bytes timed by $\overline{WR}$ pulses.
2. *Operation Command Words (OCWs):* These are the command words which command the 8259A to operate in various interrupt modes. These modes are:
   a. Fully nested mode
   b. Rotating priority mode
   c. Special mask mode
   d. Polled mode

The OCWs can be written into the 8259A anytime after initialization.

INITIALIZATION COMMAND WORDS (ICWS)

General

Whenever a command is issued with A0 = 0 and D4 = 1, this is interpreted as Initialization Command Word 1 (ICW1). ICW1 starts the intiitalization sequence during which the following automatically occur.

a. The edge sense circuit is reset, which means that following initialization, an interrupt request (IR) input must make a low-to-high transistion to generate an interrupt.

b. The Interrupt Mask Register is cleared.
c. IR7 input is assigned priority 7.
d. The slave mode address is set to 7.
e. Special Mask Mode is cleared and Status Read is set to IRR.
f. If IC4 = 0, then all functions selected in ICW4 are set to zero. (Non-Buffered mode*, no Auto-EOI, MCS-80, 85 system).

*NOTE:
Master/Slave in ICW4 is only used in the buffered mode.

Initialization Command Words 1 and 2 (ICW1, ICW2)

A_5–A_{15}: *Page starting address of service routines.* In an MCS 80/85 system, the 8 request levels will generate CALLs to 8 locations equally spaced in memory. These can be programmed to be spaced at intervals of 4 or 8 memory locations, thus the 8 routines will occupy a page of 32 or 64 bytes, respectively.

The address format is 2 bytes long (A_0–A_{15}). When the routine interval is 4, A_0–A_4 are automatically inserted by the 8259A, while A_5–A_{15} are programmed externally. When the routine interval is 8, A_0–A_5 are automatically inserted by the 8259A, while A_6–A_{15} are programmed externally.

The 8-byte interval will maintain compatibility with current software, while the 4-byte interval is best for a compact jump table.

In an 8086 system A_{15}–A_{11} are inserted in the five most significant bits of the vectoring byte and the 8259A sets the three least significant bits according to the interrupt level. A_{10}–A_5 are ignored and ADI (Address interval) has no effect.

LTIM: If LTIM = 1, then the 8259A will operate in the level interrupt mode. Edge detect logic on the interrupt inputs will be disabled.

ADI: CALL address interval. ADI = 1 then interval = 4; ADI = 0 then interval = 8.

SNGL: Single. Means that this is the only 8259A in the system. If SNGL = 1 no ICW3 will be issued.

IC4: If this bit is set—ICW4 has to be read. If ICW4 is not needed, set IC4 = 0.

Initialization Command Word 3 (ICW3)

This word is read only when there is more than one 8259A in the system and cascading is used, in which case SNGL = 0. It will load the 8-bit slave register. The functions of this register are:

a. In the master mode (either when SP = 1, or in buffered mode when M/S = 1 in ICW4) a "1" is set for each slave in the system. The master then will release byte 1 of the call sequence (for MCS-80/85 system) and will enable the corresponding slave to release bytes 2 and 3 (for 8086 only byte 2) through the cascade lines.
b. In the slave mode (either when $\overline{SP}$ = 0, or if BUF = 1 and M/S = 0 in ICW4) bits 2–0 identify the slave. The slave compares its cascade input with these bits and, if they are equal, bytes 2 and 3 of the call sequence (or just byte 2 for 8086) are released by it on the Data Bus.

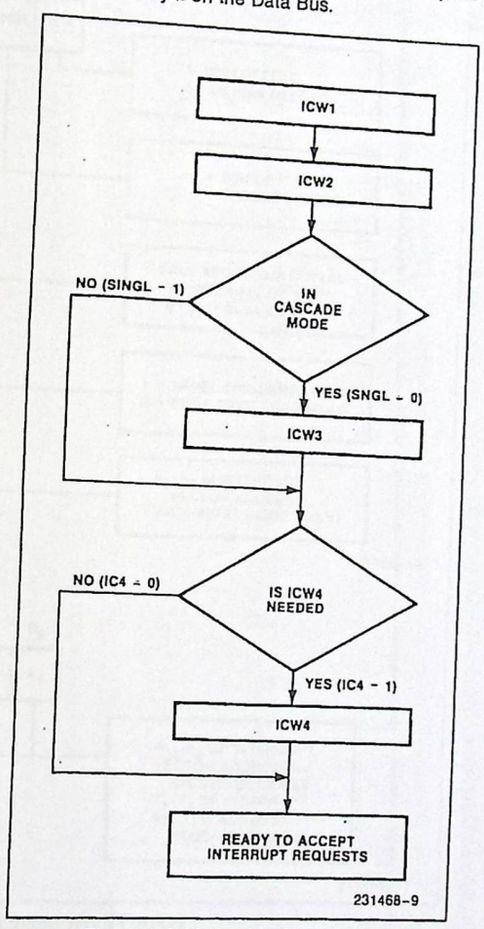


Figure 6. Initialization Sequence

Initialization Command Word 4 (ICW4)

SFNM: If SFNM = 1 the special fully nested mode is programmed.

BUF: If BUF = 1 the buffered mode is programmed. In buffered mode $\overline{SP}/\overline{EN}$ becomes an enable output and the master/slave determination is by M/S.

M/S: If buffered mode is selected: M/S = 1 means the 8259A is programmed to be a master, M/S = 0 means the 8259A is programmed to be a slave. If BUF = 0, M/S has no function.

AEOI: If AEOI = 1 the automatic end of interrupt mode is programmed.

μPM: Microprocessor mode: μPM = 0 sets the 8259A for MCS-80, 85 system operation, μPM = 1 sets the 8259A for 8086 system operation.

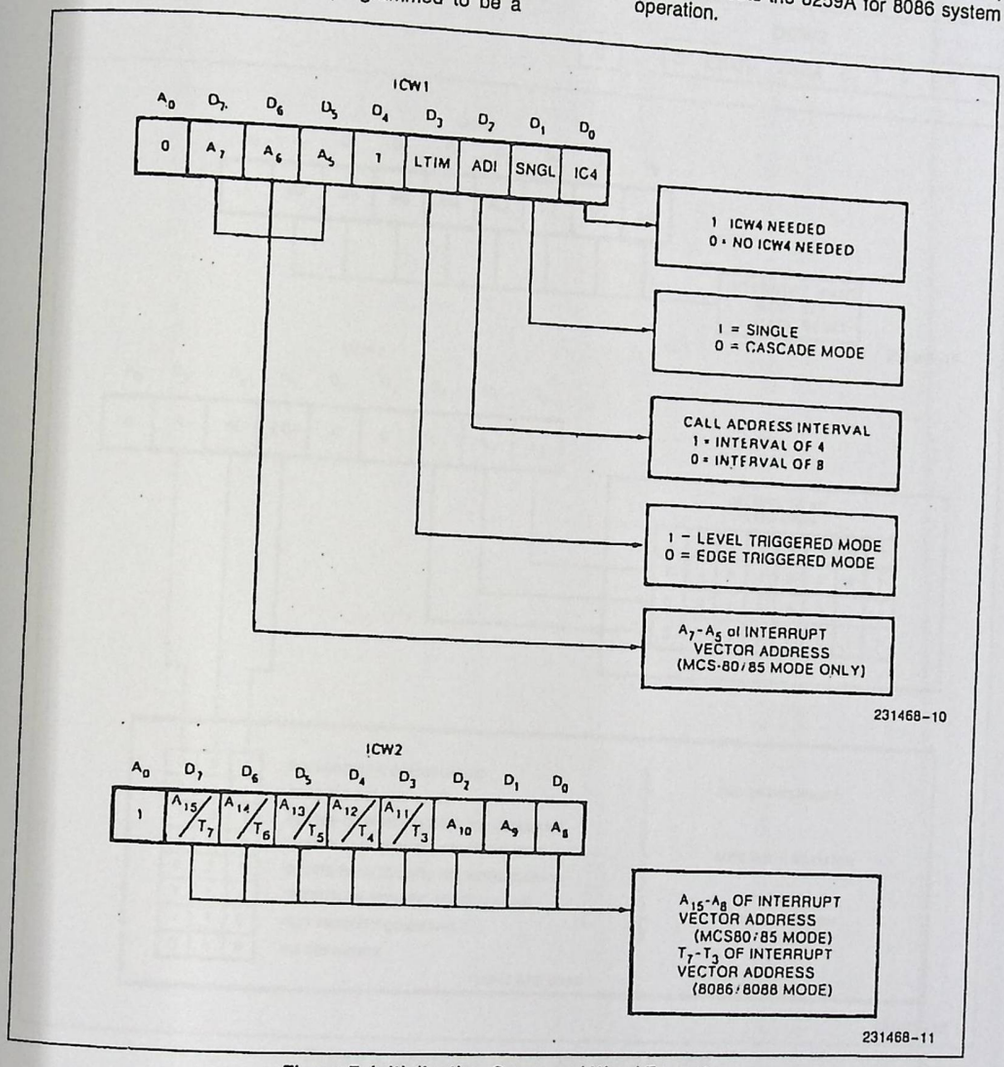


Figure 7. Initialization Command Word Format

OPERATION COMMAND WORDS (OCWS)

After the Initialization Command Words (ICWs) are programmed into the 8259A, the chip is ready to accept interrupt requests at its input lines. However, during the 8259A operation, a selection of algorithms can command the 8259A to operate in various modes through the Operation Command Words (OCWs).

Operation Control Words (OCWs)

A0		D7	D6	D5	D4	D3	D2	D1	D0
1	OCW1	M7	M6	M5	M4	M3	M2	M1	M0
0	OCW2	R	SL	EOI	0	0	L2	L1	L0
0	OCW3	0	ESMM	SMM	0	1	P	RR	RIS

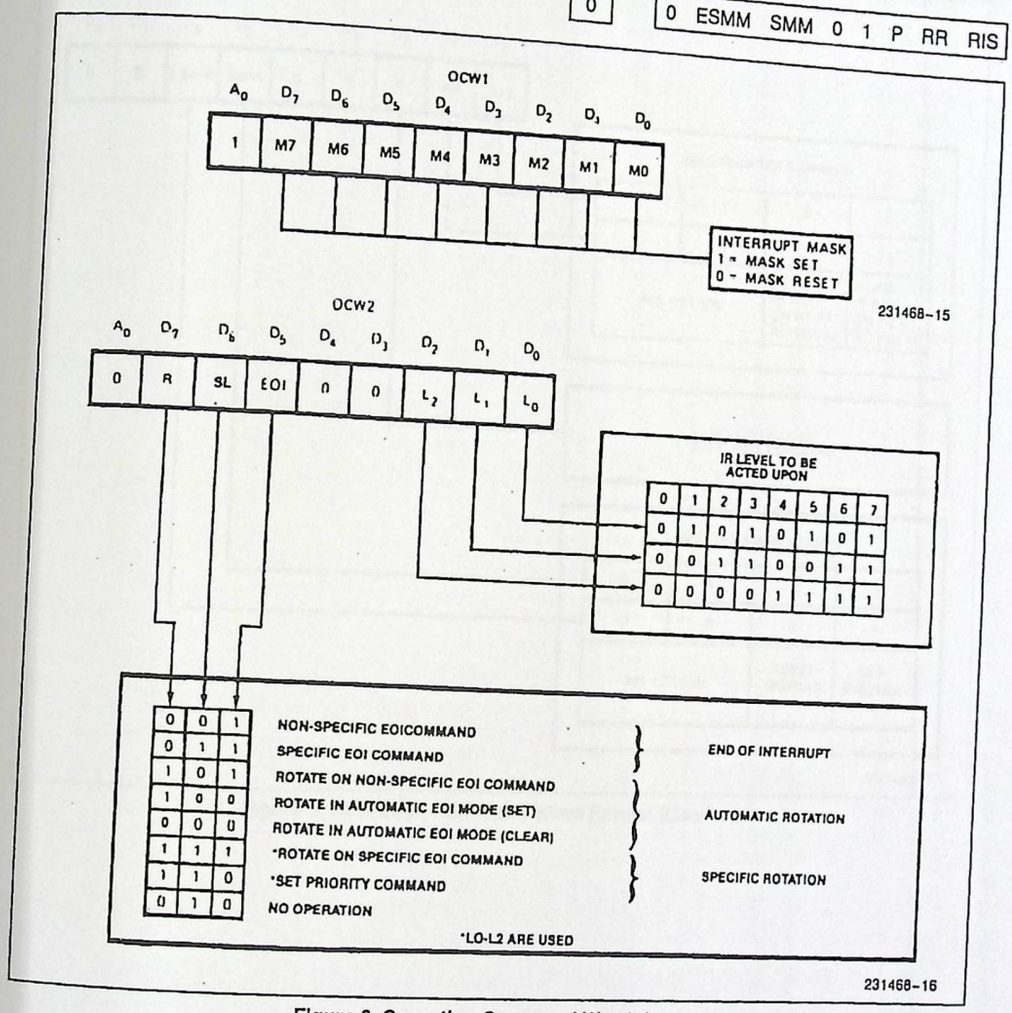


Figure 8. Operation Command Word Format

Operation Control Word 1 (OCW1)

OCW1 sets and clears the mask bits in the interrupt Mask Register (IMR). M_7–M_0 represent the eight mask bits. M = 1 indicates the channel is masked (inhibited), M = 0 indicates the channel is enabled.

Operation Control Word 2 (OCW2)

R, SL, EOI—These three bits control the Rotate and End of Interrupt modes and combinations of the two. A chart of these combinations can be found on the Operation Command Word Format.

L_2, L_1, L_0—These bits determine the interrupt level acted upon when the SL bit is active.

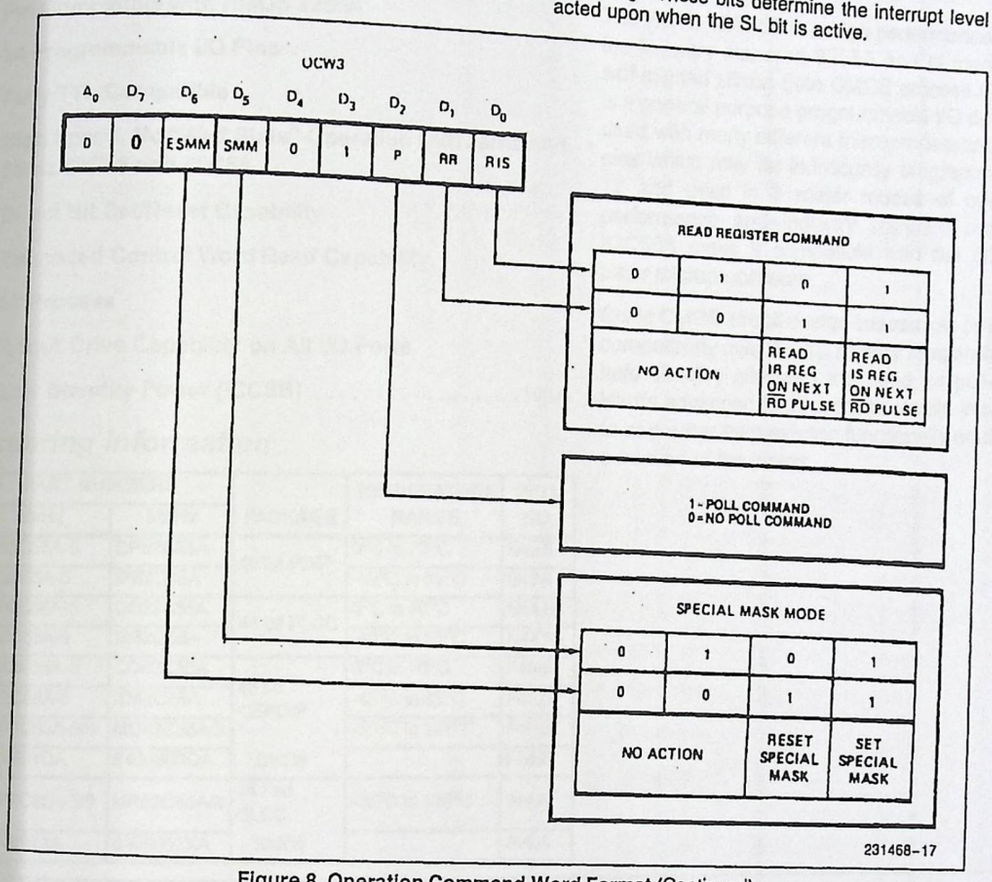


Figure 8. Operation Command Word Format (Continued)

HARRIS Semiconductor

82C55A

CMOS Programmable Peripheral Interface

June 1998

Features

- Pin Compatible with NMOS 8255A
- 24 Programmable I/O Pins
- Fully TTL Compatible
- High Speed, No "Wait State" Operation with 5MHz and 8MHz 80C86 and 80C88
- Direct Bit Set/Reset Capability
- Enhanced Control Word Read Capability
- L7 Process
- 2.5mA Drive Capability on All I/O Ports
- Low Standby Power (ICCSB) 10µA

Ordering Information

PART NUMBERS		PACKAGE	TEMPERATURE RANGE	PKG. NO.
5MHz	8MHz			
CP82C55A-5	CP82C55A	40 Ld PDIP	0°C to 70°C	E40.6
IP82C55A-5	IP82C55A		-40°C to 85°C	E40.6
CS82C55A-5	CS82C55A	44 Ld PLCC	0°C to 70°C	N44.65
IS82C55A-5	IS82C55A		-40°C to 85°C	N44.65
CD82C55A-5	CD82C55A	40 Ld CERDIP	0°C to 70°C	F40.6
ID82C55A-5	ID82C55A		-40°C to 85°C	F40.6
MD82C55A-5/B	MD82C55A/B		-55°C to 125°C	F40.6
8406601QA	8406602QA	SMD#		F40.6
MR82C55A-5/B	MR82C55A/B	44 Pad CLCC	-55°C to 125°C	J44.A
8406601XA	8406602XA	SMD#		J44.A

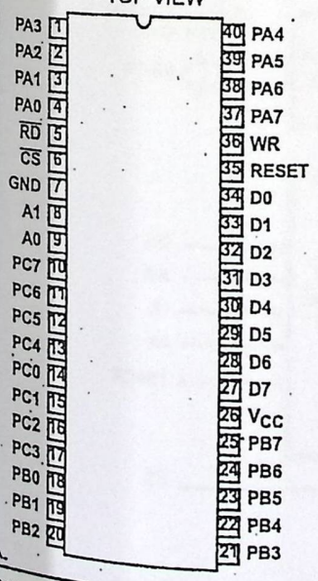
Description

The Harris 82C55A is a high performance CMOS version of the industry standard 8255A and is manufactured using a self-aligned silicon gate CMOS process (Scaled SAJI IV). It is a general purpose programmable I/O device which may be used with many different microprocessors. There are 24 I/O pins which may be individually programmed in 2 groups of 12 and used in 3 major modes of operation. The high performance and industry standard configuration of the 82C55A make it compatible with the 80C86, 80C88 and other microprocessors.

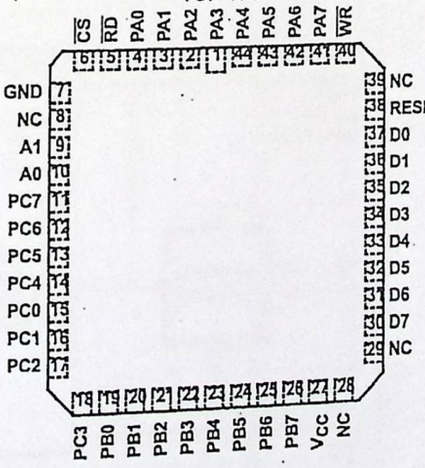
Static CMOS circuit design insures low operating power. TTL compatibility over the full military temperature range and bus hold circuitry eliminate the need for pull-up resistors. The Harris advanced SAJI process results in performance equal to or greater than existing functionally equivalent products at a fraction of the power.

Pinouts

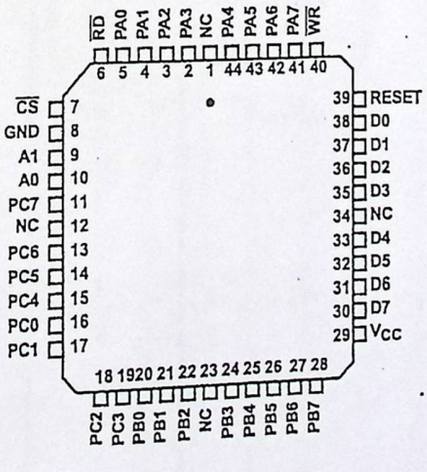
82C55A (DIP)
TOP VIEW



82C55A (CLCC)
TOP VIEW



82C55A (PLCC)
TOP VIEW

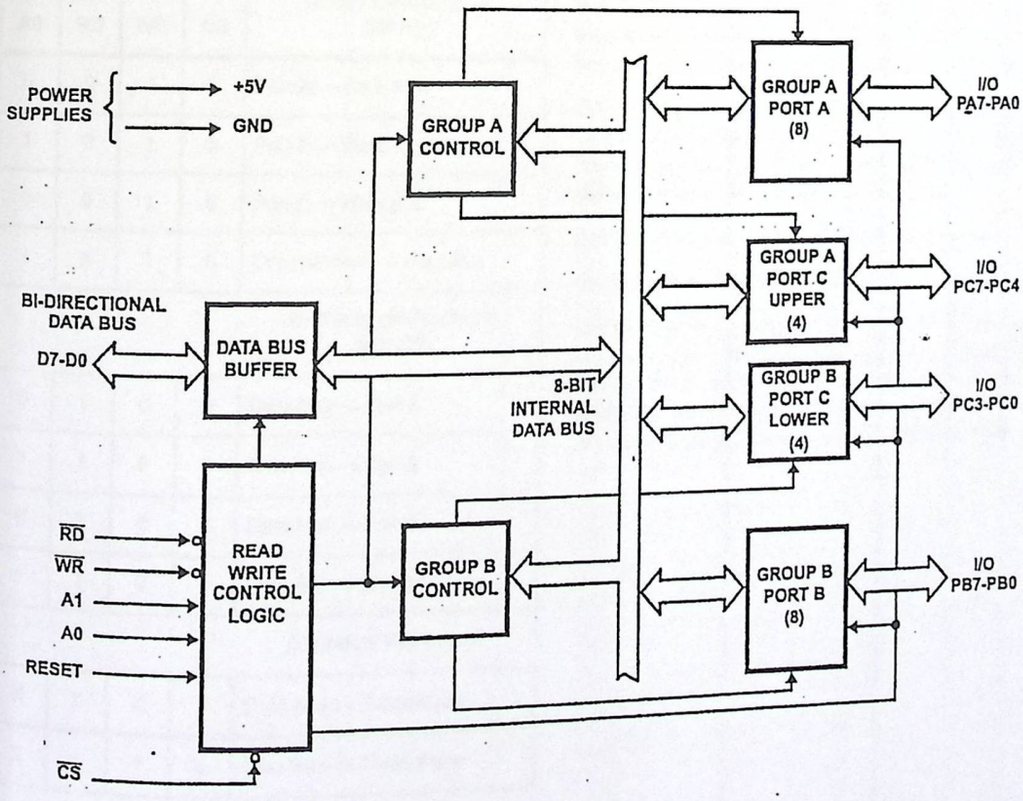


File Number 2969.2

Pin Description

SYMBOL	PIN NUMBER	TYPE	DESCRIPTION
V_{CC}	26		V_{CC}: The +5V power supply pin. A 0.1µF capacitor between pins 26 and 7 is recommended for decoupling.
GND	7		GROUND
D0-D7	27-34	I/O	DATA BUS: The Data Bus lines are bidirectional three-state pins connected to the system data bus.
RESET	35	I	RESET: A high on this input clears the control register and all ports (A, B, C) are set to the input mode with the "Bus Hold" circuitry turned on.
$\overline{CS}$	6	I	CHIP SELECT: Chip select is an active low input used to enable the 82C55A onto the Data Bus for CPU communications.
$\overline{RD}$	5	I	READ: Read is an active low input control signal used by the CPU to read status information or data via the data bus.
$\overline{WR}$	36	I	WRITE: Write is an active low input control signal used by the CPU to load control words and data into the 82C55A.
A0-A1	8, 9	I	ADDRESS: These input signals, in conjunction with the $\overline{RD}$ and $\overline{WR}$ inputs, control the selection of one of the three ports or the control word register. A0 and A1 are normally connected to the least significant bits of the Address Bus A0, A1.
PA0-PA7	1-4, 37-40	I/O	PORT A: 8-bit input and output port. Both bus hold high and bus hold low circuitry are present on this port.
PB0-PB7	18-25	I/O	PORT B: 8-bit input and output port. Bus hold high circuitry is present on this port.
PC0-PC7	10-17	I/O	PORT C: 8-bit input and output port. Bus hold circuitry is present on this port.

Functional Diagram



Functional Description

Data Bus Buffer

This three-state bi-directional 8-bit buffer is used to interface the 82C55A to the system data bus. Data is transmitted or received by the buffer upon execution of input or output instructions by the CPU. Control words and status information are also transferred through the data bus buffer.

Read/Write and Control Logic

The function of this block is to manage all of the internal and external transfers of both Data and Control or Status words. It accepts inputs from the CPU Address and Control busses and in turn, issues commands to both of the Control Groups.

(CS) Chip Select. A "low" on this input pin enables the communcation between the 82C55A and the CPU.

(RD) Read. A "low" on this input pin enables 82C55A to send the data or status information to the CPU on the data bus. In essence, it allows the CPU to "read from" the 82C55A.

(WR) Write. A "low" on this input pin enables the CPU to write data or control words into the 82C55A.

(A0 and A1) Port Select 0 and Port Select 1. These input signals, in conjunction with the RD and WR inputs, control the selection of one of the three ports or the control word register. They are normally connected to the least significant bits of the address bus (A0 and A1).

82C55A BASIC OPERATION

A1	A0	RD	WR	CS	INPUT OPERATION (READ)
0	0	0	1	0	Port A → Data Bus
0	1	0	1	0	Port B → Data Bus
1	0	0	1	0	Port C → Data Bus
1	1	0	1	0	Control Word → Data Bus
					OUTPUT OPERATION (WRITE)
0	0	1	0	0	Data Bus → Port A
0	1	1	0	0	Data Bus → Port B
1	0	1	0	0	Data Bus → Port C
1	1	1	0	0	Data Bus → Control
					DISABLE FUNCTION
X	X	X	X	1	Data Bus → Three-State
X	X	1	1	0	Data Bus → Three-State

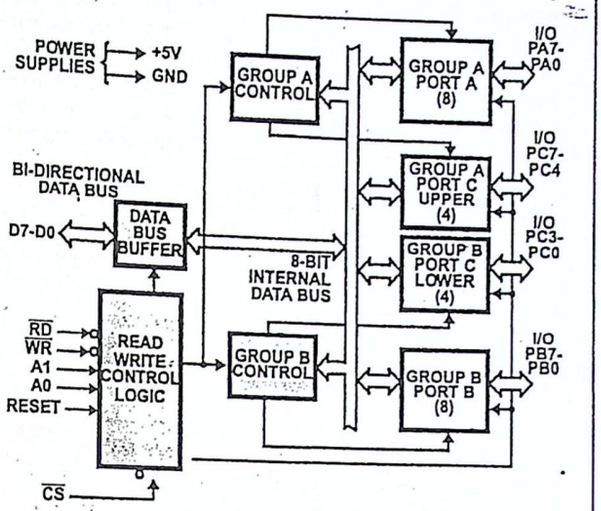


FIGURE 1. 82C55A BLOCK DIAGRAM. DATA BUS BUFFER, READ/WRITE, GROUP A & B CONTROL LOGIC FUNCTIONS

(RESET) Reset. A "high" on this input initializes the control register to 9Bh and all ports (A, B, C) are set to the input mode. "Bus hold" devices internal to the 82C55A will hold the I/O port inputs to a logic "1" state with a maximum hold current of 400µA.

Group A and Group B Controls

The functional configuration of each port is programmed by the systems software. In essence, the CPU "outputs" a control word to the 82C55A. The control word contains information such as "mode", "bit set", "bit reset", etc., that initializes the functional configuration of the 82C55A.

Each of the Control blocks (Group A and Group B) accepts "commands" from the Read/Write Control logic, receives "control words" from the internal data bus and issues the proper commands to its associated ports.

Control Group A - Port A and Port C upper (C7 - C4)

Control Group B - Port B and Port C lower (C3 - C0)

The control word register can be both written and read as shown in the "Basic Operation" table. Figure 4 shows the control word format for both Read and Write operations. When the control word is read, bit D7 will always be a logic "1", as this implies control word mode information.

Ports A, B, and C

The 82C55A contains three 8-bit ports (A, B, and C). All can be configured to a wide variety of functional characteristics by the system software but each has its own special features or "personality" to further enhance the power and flexibility of the 82C55A.

Port A One 8-bit data output latch/buffer and one 8-bit data input latch. Both "pull-up" and "pull-down" bus-hold devices are present on Port A. See Figure 2A.

Port B One 8-bit data input/output latch/buffer and one 8-bit data input buffer. See Figure 2B.

Port C One 8-bit data output latch/buffer and one 8-bit data input buffer (no latch for input). This port can be divided into two 4-bit ports under the mode control. Each 4-bit port contains a 4-bit latch and it can be used for the control signal output and status signal inputs in conjunction with ports A and B. See Figure 2B.

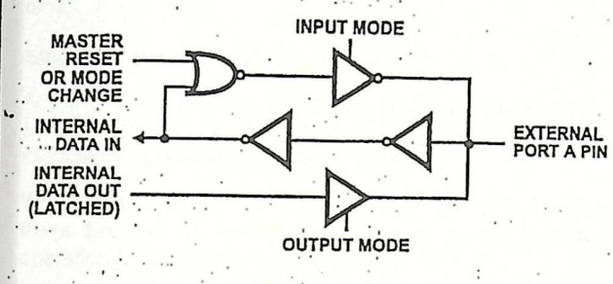


FIGURE 2A. PORT A BUS-HOLD CONFIGURATION

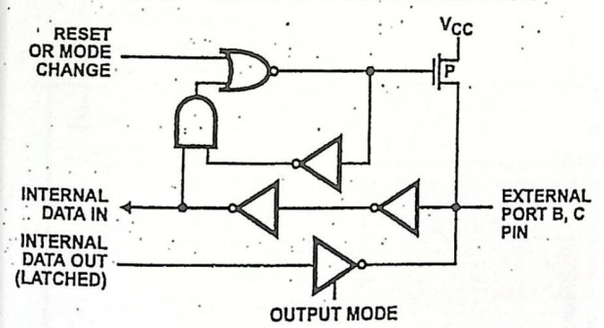


FIGURE 2B. PORT B AND C BUS-HOLD CONFIGURATION

FIGURE 2. BUS-HOLD CONFIGURATION

Operational Description

Mode Selection

There are three basic modes of operation than can be selected by the system software:

Mode 0 - Basic Input/Output
Mode 1 - Strobed Input/Output
Mode 2 - Bi-directional Bus

When the reset input goes "high", all ports will be set to the input mode with all 24 port lines held at a logic "one" level by internal bus hold devices. After the reset is removed, the 82C55A can remain in the input mode with no additional initialization required. This eliminates the need to pullup or pull-down resistors in all-CMOS designs. The control word register will contain 9Bh. During the execution of the system program, any of the other modes may be selected using a single output instruction. This allows a single 82C55A to service a variety of peripheral devices with a simple software maintenance routine. Any port programmed as an output port is initialized to all zeros when the control word is written.

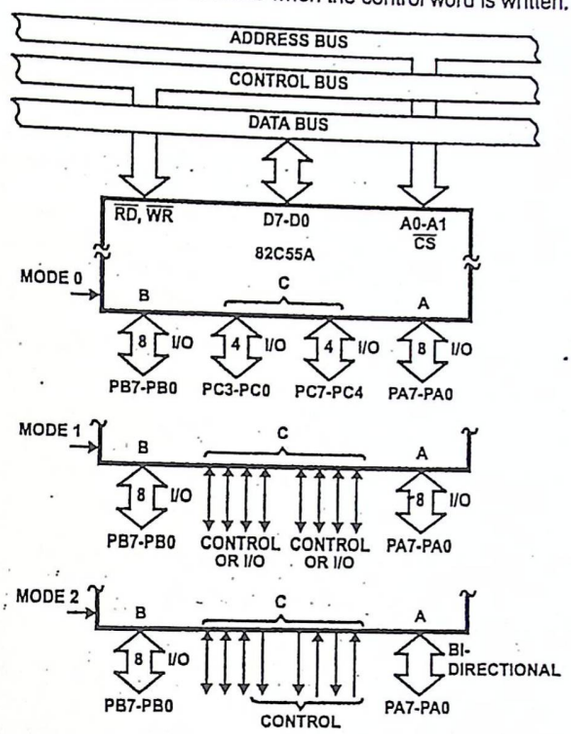


FIGURE 3. BASIC MODE DEFINITIONS AND BUS INTERFACE

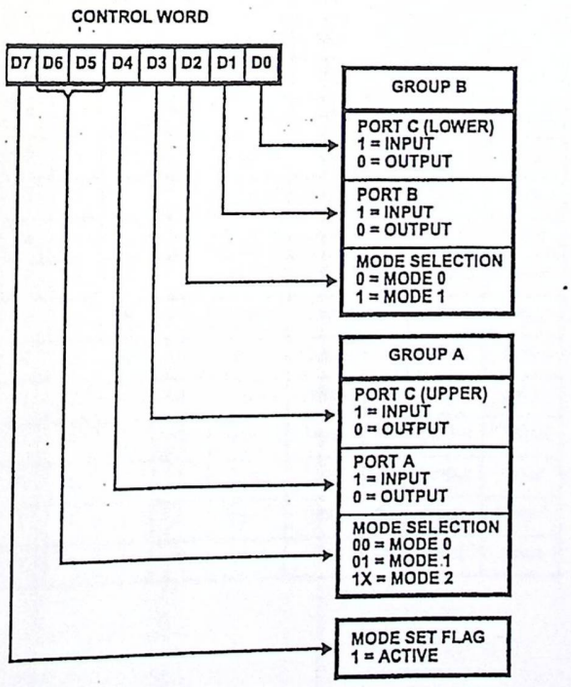


FIGURE 4. MODE DEFINITION FORMAT

The modes for Port A and Port B can be separately defined, while Port C is divided into two portions as required by the Port A and Port B definitions. All of the output registers, including the status flip-flops, will be reset whenever the mode is changed. Modes may be combined so that their functional definition can be "tailored" to almost any I/O structure. For instance: Group B can be programmed in Mode 0 to monitor simple switch closings or display computational results, Group A could be programmed in Mode 1 to monitor a keyboard or tape reader on an interrupt-driven basis.

The mode definitions and possible mode combinations may seem confusing at first, but after a cursory review of the complete device operation a simple, logical I/O approach will surface. The design of the 82C55A has taken into account things such as efficient PC board layout, control signal definition vs. PC layout and complete functional flexibility to support almost any peripheral device with no external logic. Such design represents the maximum use of the available pins.

Single Bit Set/Reset Feature (Figure 5)

Any of the eight bits of Port C can be Set or Reset using a single Output instruction. This feature reduces software requirements in control-based applications.

When Port C is being used as status/control for Port A or B, these bits can be set or reset by using the Bit Set/Reset operation just as if they were output ports.

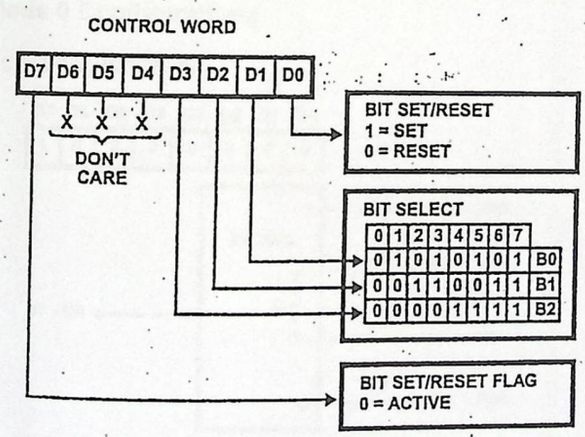


FIGURE 5. BIT SET/RESET FORMAT

Interrupt Control Functions

When the 82C55A is programmed to operate in mode 1 or mode 2, control signals are provided that can be used as interrupt request inputs to the CPU. The interrupt request signals, generated from port C, can be inhibited or enabled by setting or resetting the associated INTE flip-flop, using the bit set/reset function of port C.

This function allows the programmer to enable or disable a CPU interrupt by a specific I/O device without affecting any other device in the interrupt structure.

INTE Flip-Flop Definition

(BIT-SET)-INTE is SET - Interrupt Enable

(BIT-RESET)-INTE is Reset - Interrupt Disable

NOTE: All Mask flip-flops are automatically reset during mode selection and device Reset.

Operating Modes

Mode 0 (Basic Input/Output). This functional configuration provides simple input and output operations for each of the three ports. No handshaking is required, data is simply written to or read from a specific port.

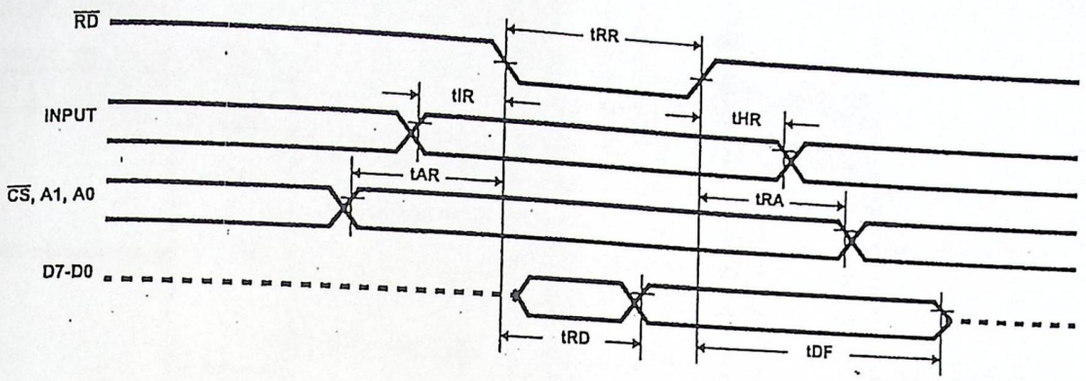
Mode 0 Basic Functional Definitions:

- Two 8-bit ports and two 4-bit ports
- Any Port can be input or output
- Outputs are latched
- Input are not latched
- 16 different Input/Output configurations possible

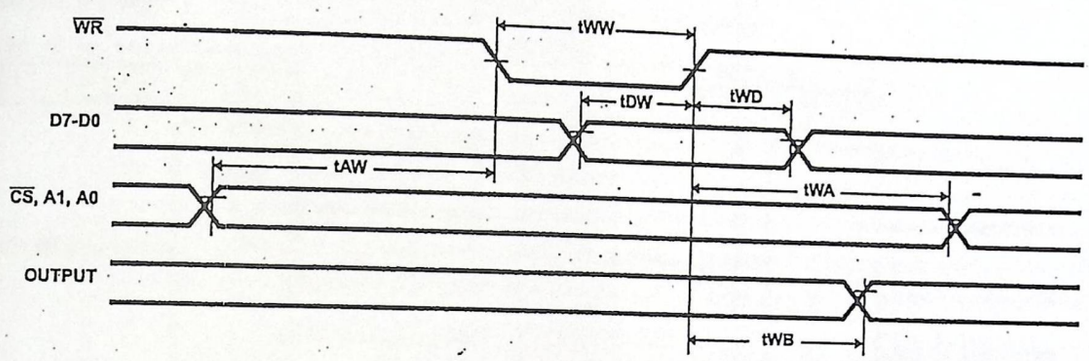
MODE 0 PORT DEFINITION

A		B		GROUP A			GROUP B	
D4	D3	D1	D0	PORT A	PORT C (Upper)	#	PORT B	PORT C (Lower)
0	0	0	0	Output	Output	0	Output	Output
0	0	0	1	Output	Output	1	Output	Input
0	0	1	0	Output	Output	2	Input	Output
0	0	1	1	Output	Output	3	Input	Input
0	1	0	0	Output	Input	4	Output	Output
0	1	0	1	Output	Input	5	Output	Input
0	1	1	0	Output	Input	6	Input	Output
0	1	1	1	Output	Input	7	Input	Input
1	0	0	0	Input	Output	8	Output	Output
1	0	0	1	Input	Output	9	Output	Input
1	0	1	0	Input	Output	10	Input	Output
1	0	1	1	Input	Output	11	Input	Input
1	1	0	0	Input	Input	12	Output	Output
1	1	0	1	Input	Input	13	Output	Input
1	1	1	0	Input	Input	14	Input	Output
1	1	1	1	Input	Input	15	Input	Input

Mode 0 (Basic Input)

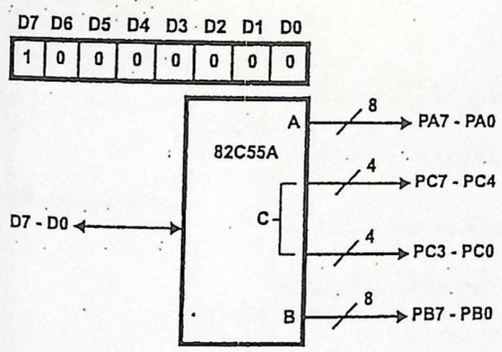


Mode 0 (Basic Output)

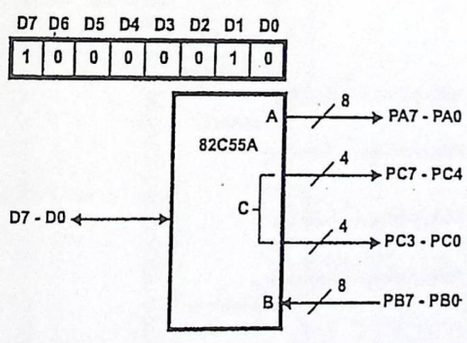


Mode 0 Configurations

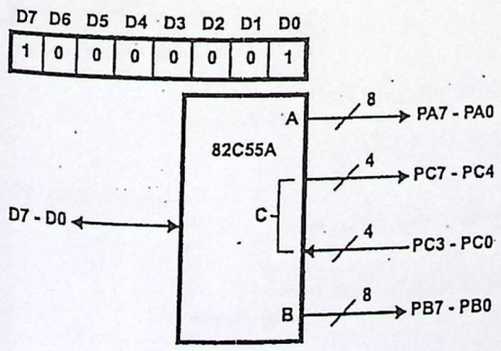
CONTROL WORD #0



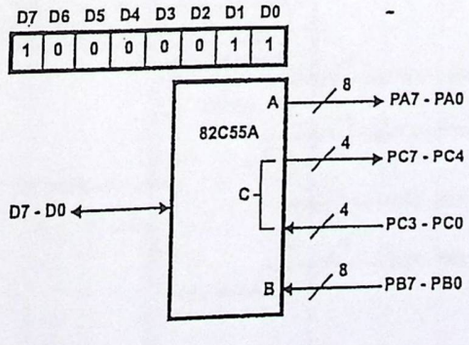
CONTROL WORD #2



CONTROL WORD #1



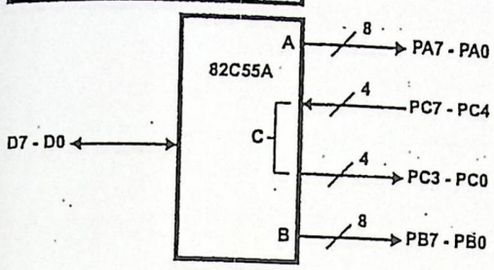
CONTROL WORD #3



Mode 0 Configurations (Continued)

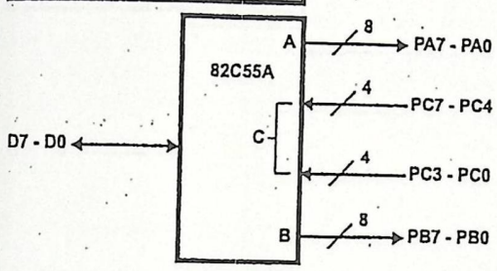
CONTROL WORD #4

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



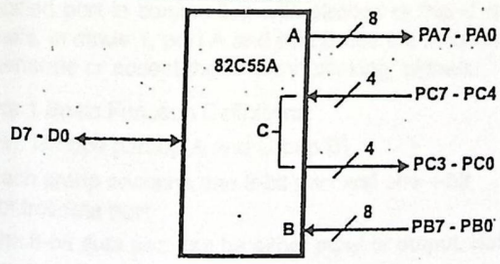
CONTROL WORD #5

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



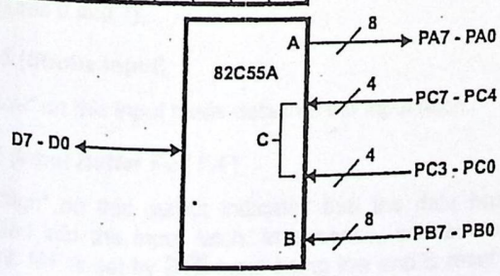
CONTROL WORD #6

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



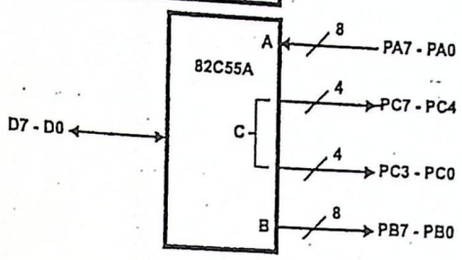
CONTROL WORD #7

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



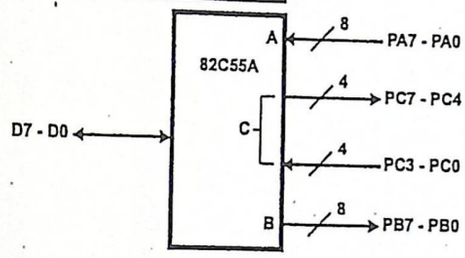
CONTROL WORD #8

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



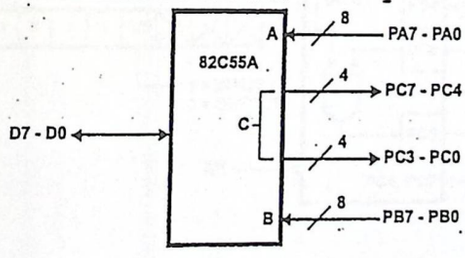
CONTROL WORD #9

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



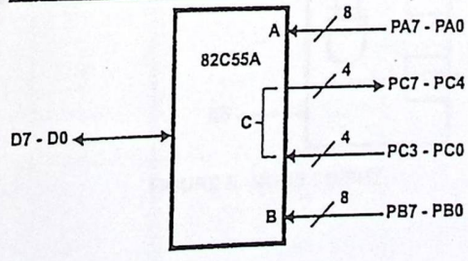
CONTROL WORD #10

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



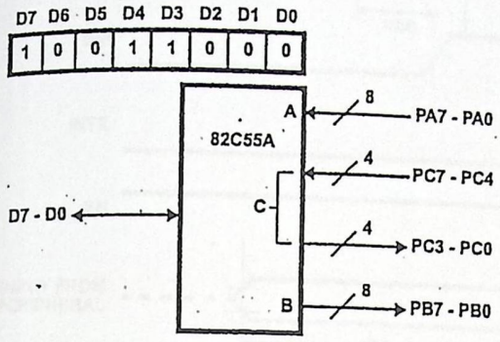
CONTROL WORD #11

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

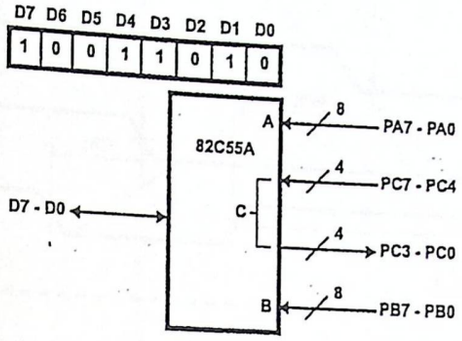


Mode 0 Configurations (Continued)

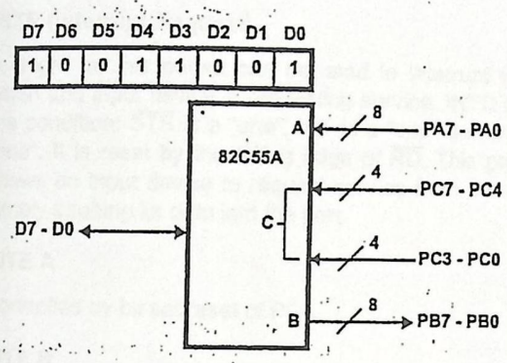
CONTROL WORD #12



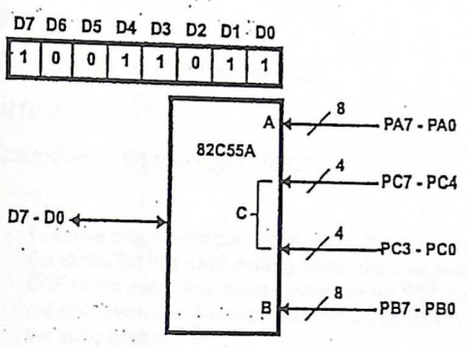
CONTROL WORD #14



CONTROL WORD #13



CONTROL WORD #15



Operating Modes

Mode 1 - (Strobed Input/Output). This functional configuration provides a means for transferring I/O data to or from a specified port in conjunction with strobes or "hand shaking" signals. In mode 1, port A and port B use the lines on port C to generate or accept these "hand shaking" signals.

Mode 1 Basic Function Definitions:

- Two Groups (Group A and Group B)
- Each group contains one 8-bit port and one 4-bit control/data port
- The 8-bit data port can be either input or output. Both inputs and outputs are latched.
- The 4-bit port is used for control and status of the 8-bit port.

Input Control Signal Definition

(Figures 6 and 7)

STB (Strobe Input)

A "low" on this input loads data into the input latch.

IBF (Input Buffer Full F/F)

A "high" on this output indicates that the data has been loaded into the input latch: in essence, and acknowledgment. IBF is set by $\overline{STB}$ input being low and is reset by the rising edge of the $\overline{RD}$ input.

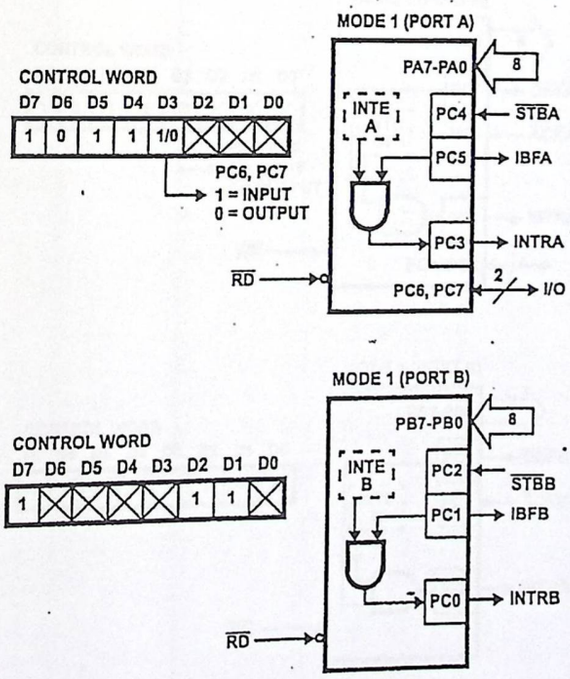


FIGURE 6. MODE 1 INPUT

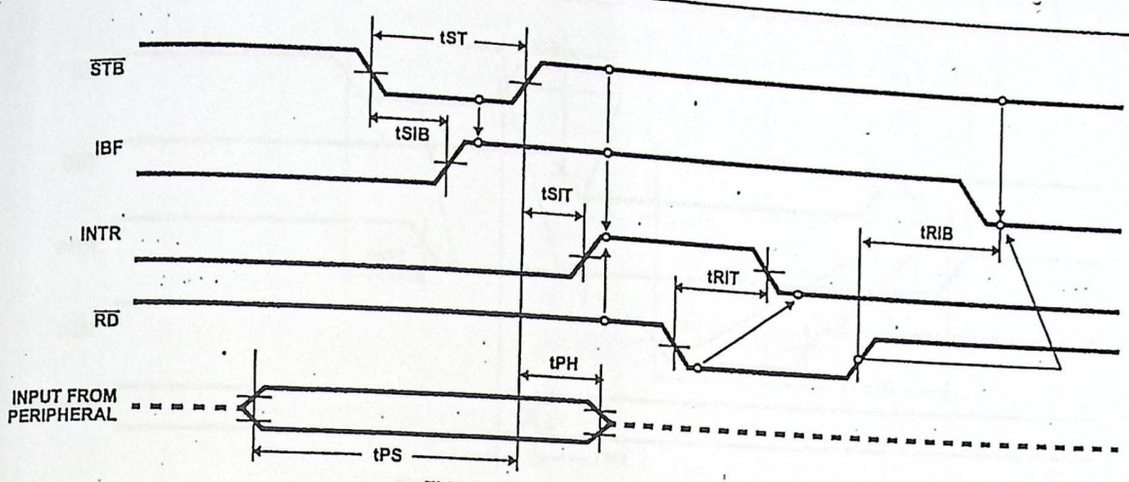


FIGURE 7. MODE 1 (STROBED INPUT)

INTR (Interrupt Request)

A "high" on this output can be used to interrupt the CPU when and input device is requesting service. INTR is set by the condition: $\overline{STB}$ is a "one", IBF is a "one" and INTE is a "one". It is reset by the falling edge of $\overline{RD}$. This procedure allows an input device to request service from the CPU by simply strobing its data into the port.

INTE A

Controlled by bit set/reset of PC4.

INTE B

Controlled by bit set/reset of PC2.

Output Control Signal Definition

(Figure 8 and 9)

$\overline{OBF}$ - Output Buffer Full F/F). The $\overline{OBF}$ output will go "low" to indicate that the CPU has written data out to be specified port. This does not mean valid data is sent out of the part at this time since $\overline{OBF}$ can go true before data is available. Data is guaranteed valid at the rising edge of $\overline{OBF}$, (See Note 1). The $\overline{OBF}$ F/F will be set by the rising edge of the $\overline{WR}$ input and reset by $\overline{ACK}$ input being low.

$\overline{ACK}$ - Acknowledge Input). A "low" on this input informs the 82C55A that the data from Port A or Port B is ready to be accepted. In essence, a response from the peripheral device indicating that it is ready to accept data, (See Note 1).

INTR - (Interrupt Request). A "high" on this output can be used to interrupt the CPU when an output device has accepted data transmitted by the CPU. INTR is set when $\overline{ACK}$ is a "one", OBF is a "one" and INTE is a "one". It is reset by the falling edge of $\overline{WR}$.

INTE A

Controlled by Bit Set/Reset of PC6.

INTE B

Controlled by Bit Set/Reset of PC2.

NOTE:

1. To strobe data into the peripheral device, the user must operate the strobe line in a hand shaking mode. The user needs to send $\overline{OBF}$ to the peripheral device, generates an $\overline{ACK}$ from the peripheral device and then latch data into the peripheral device on the rising edge of $\overline{OBF}$.

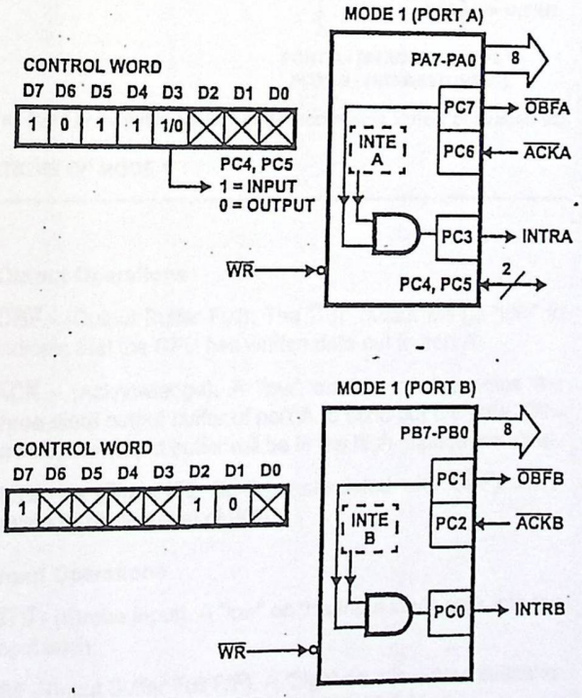


FIGURE 8. MODE 1 OUTPUT

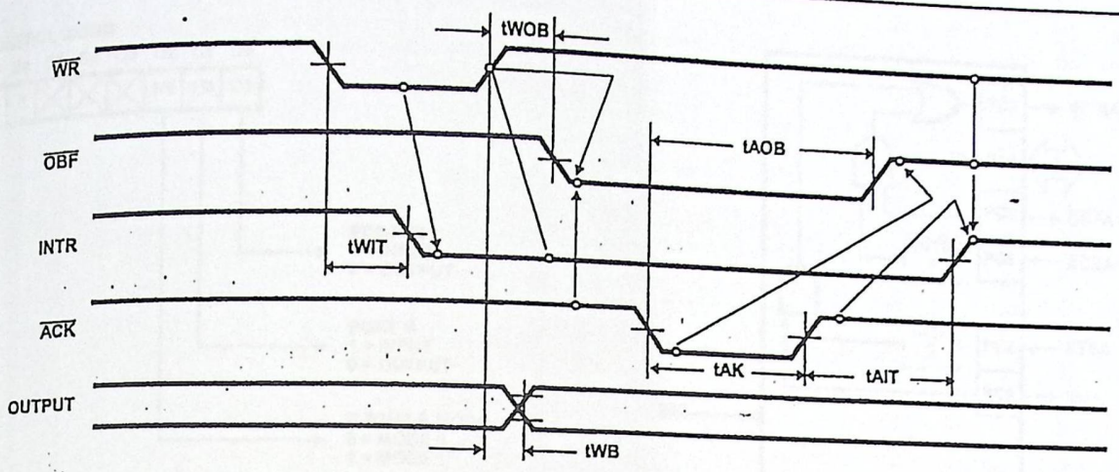
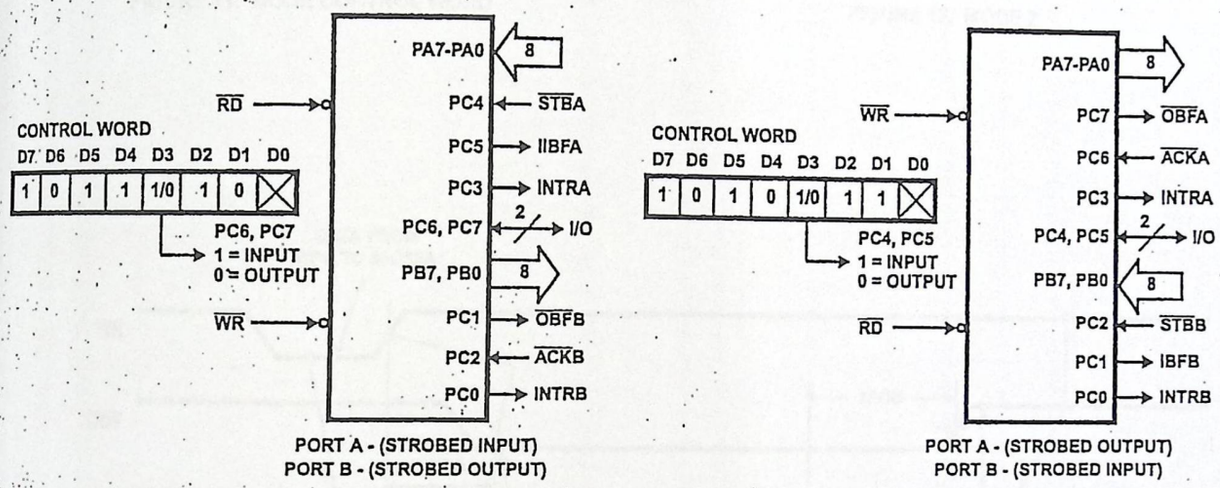


FIGURE 9. MODE 1 (STROBED OUTPUT)



Combinations of Mode 1: Port A and Port B can be individually defined as input or output in Mode 1 to support a wide variety of strobed I/O applications.

FIGURE 10. COMBINATIONS OF MODE 1

Operating Modes

Mode 2 (Strobed Bi-Directional Bus I/O)

The functional configuration provides a means for communicating with a peripheral device or structure on a single 8-bit bus for both transmitting and receiving data (bi-directional bus I/O). "Hand shaking" signals are provided to maintain proper bus flow discipline similar to Mode 1. Interrupt generation and enable/disable functions are also available.

Mode 2 Basic Functional Definitions:

- Used in Group A only
- One 8-bit, bi-directional bus Port (Port A) and a 5-bit control Port (Port C)
- Both inputs and outputs are latched
- The 5-bit control port (Port C) is used for control and status for the 8-bit, bi-directional bus port (Port A)

Bi-Directional Bus I/O Control Signal Definition

(Figures 11, 12, 13, 14)

INTR - (Interrupt Request). A high on this output can be used to interrupt the CPU for both input or output operations.

Output Operations

$\overline{OBF}$ - (Output Buffer Full). The $\overline{OBF}$ output will go "low" to indicate that the CPU has written data out to port A.

$\overline{ACK}$ - (Acknowledge). A "low" on this input enables the three-state output buffer of port A to send out the data. Otherwise, the output buffer will be in the high impedance state.

INTE 1 - (The INTE flip-flop associated with $\overline{OBF}$). Controlled by bit set/reset of PC4.

Input Operations

$\overline{STB}$ - (Strobe Input). A "low" on this input loads data into the input latch.

IBF - (Input Buffer Full F/F). A "high" on this output indicates that data has been loaded into the input latch.

INTE 2 - (The INTE flip-flop associated with IBF). Controlled by bit set/reset of PC4.

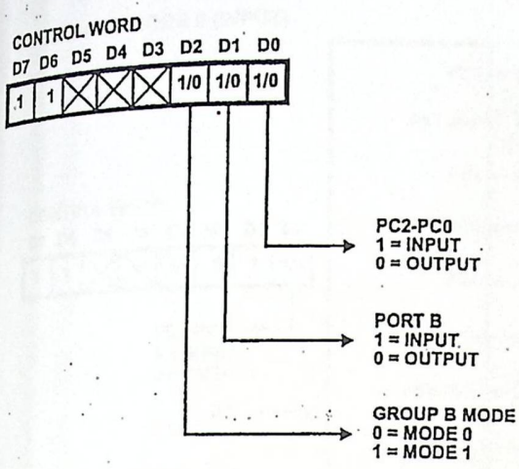


FIGURE 11. MODE CONTROL WORD

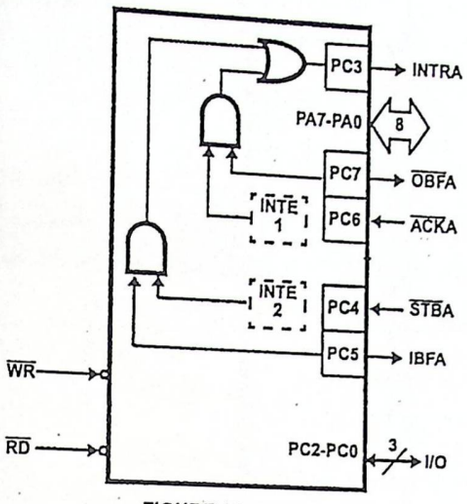
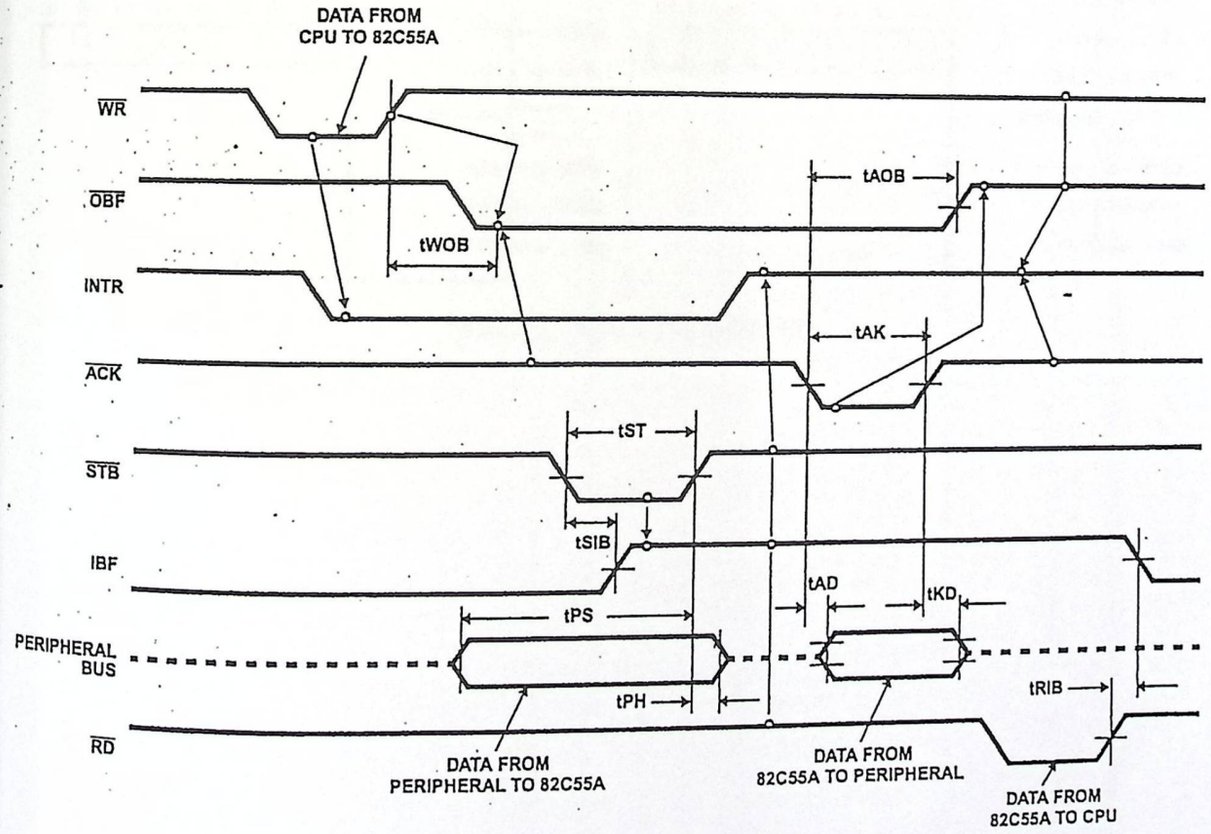


FIGURE 12. MODE 2



NOTE: Any sequence where $\overline{WR}$ occurs before $\overline{ACK}$ and $\overline{STB}$ occurs before RD is permissible. (INTR = IBF • MASK • $\overline{STB}$ • $\overline{RD}$ + $\overline{OBF}$ • MASK • $\overline{ACK}$ • $\overline{WR}$)

FIGURE 13. MODE 2 (BI-DIRECTIONAL)

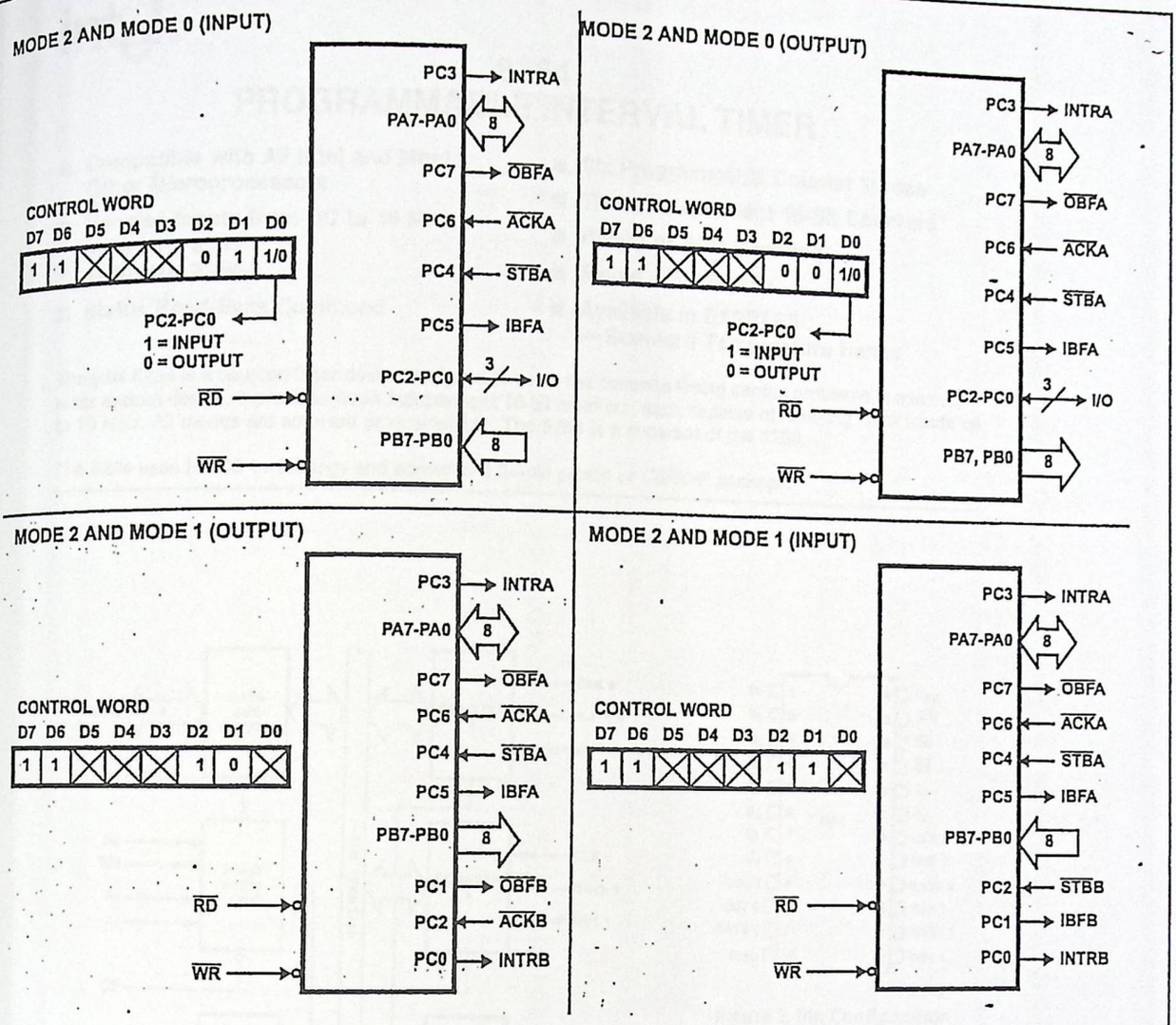


FIGURE 14. MODE 2 COMBINATIONS

Table 1. Pin Description

Symbol	Pin No.	Type	Name and Function
D_7–D_0	1–8	I/O	**DATA:** Bi-directional three state data bus lines, connected to system data bus.
CLK 0	9	I	**CLOCK 0:** Clock input of Counter 0.
OUT 0	10	O	**OUTPUT 0:** Output of Counter 0.
GATE 0	11	I	**GATE 0:** Gate input of Counter 0.
GND	12		**GROUND:** Power supply connection.
V_{CC}	24		**POWER:** +5V power supply connection.
$\overline{WR}$	23	I	**WRITE CONTROL:** This input is low during CPU write operations.
$\overline{RD}$	22	I	**READ CONTROL:** This input is low during CPU read operations.
$\overline{CS}$	21	I	**CHIP SELECT:** A low on this input enables the 8254 to respond to $\overline{RD}$ and $\overline{WR}$ signals. $\overline{RD}$ and $\overline{WR}$ are ignored otherwise.
A_1, A_0	20–19	I	**ADDRESS:** Used to select one of the three Counters or the Control Word Register for read or write operations. Normally connected to the system address bus.

A_1	A_0	Selects
0	0	Counter 0
0	1	Counter 1
1	0	Counter 2
1	1	Control Word Register

CLK 2	18	I	**CLOCK 2:** Clock input of Counter 2.
OUT 2	17	O	**OUT 2:** Output of Counter 2.
GATE 2	16	I	**GATE 2:** Gate input of Counter 2.
CLK 1	15	I	**CLOCK 1:** Clock input of Counter 1.
GATE 1	14	I	**GATE 1:** Gate input of Counter 1.
OUT 1	13	O	**OUT 1:** Output of Counter 1.

FUNCTIONAL DESCRIPTION

General

The 8254 is a programmable interval timer/counter designed for use with Intel microcomputer systems. It is a general purpose, multi-timing element that can be treated as an array of I/O ports in the system software.

The 8254 solves one of the most common problems in any microcomputer system, the generation of accurate time delays under software control. Instead of setting up timing loops in software, the programmer configures the 8254 to match his requirements and programs one of the counters for the desired delay. After the desired delay, the 8254 will interrupt the CPU. Software overhead is minimal and variable length delays can easily be accommodated.

Some of the other counter/timer functions common to microcomputers which can be implemented with the 8254 are:

- Real time clock
- Event-counter
- Digital one-shot
- Programmable rate generator
- Square wave generator
- Binary rate multiplier
- Complex waveform generator
- Complex motor controller

Block Diagram

DATA BUS BUFFER

This 3-state, bi-directional, 8-bit buffer is used to interface the 8254 to the system bus (see Figure 3).

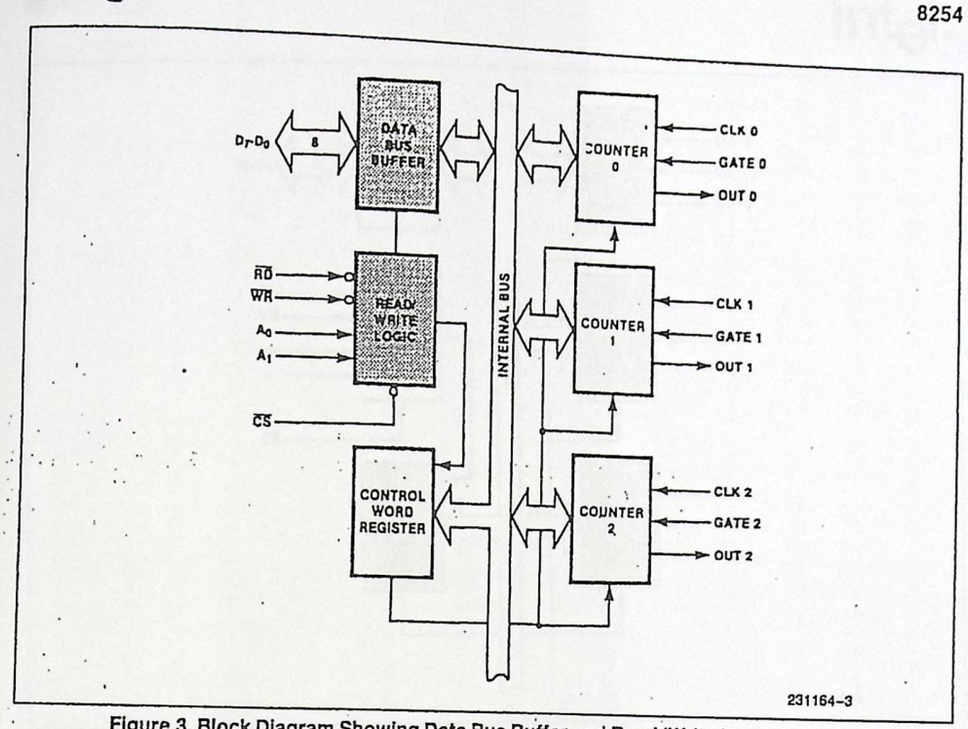


Figure 3. Block Diagram Showing Data Bus Buffer and Read/Write Logic Functions

READ/WRITE LOGIC

The Read/Write Logic accepts inputs from the system bus and generates control signals for the other functional blocks of the 8254. A_1 and A_0 select one of the three counters or the Control Word Register to be read from/written into. A "low" on the $\overline{RD}$ input tells the 8254 that the CPU is reading one of the counters. A "low" on the $\overline{WR}$ input tells the 8254 that the CPU is writing either a Control Word or an initial count. Both $\overline{RD}$ and $\overline{WR}$ are qualified by $\overline{CS}$; $\overline{RD}$ and $\overline{WR}$ are ignored unless the 8254 has been selected by holding $\overline{CS}$ low.

CONTROL WORD REGISTER

The Control Word Register (see Figure 4) is selected by the Read/Write Logic when $A_1,A_0 = 11$. If the CPU then does a write operation to the 8254, the data is stored in the Control Word Register and is interpreted as a Control Word used to define the operation of the Counters.

The Control Word Register can only be written to; status information is available with the Read-Back Command.

COUNTER 0, COUNTER 1, COUNTER 2

These three functional blocks are identical in operation, so only a single Counter will be described. The internal block diagram of a single counter is shown in Figure 5.

The Counters are fully independent. Each Counter may operate in a different Mode.

The Control Word Register is shown in the figure; it is not part of the Counter itself, but its contents determine how the Counter operates.

The status register, shown in Figure 5, when latched, contains the current contents of the Control Word Register and status of the output and null count flag. (See detailed explanation of the Read-Back command.)

The actual counter is labelled CE (for "Counting Element"). It is a 16-bit presettable synchronous down counter.

OL_M and OL_L are two 8-bit latches. OL stands for "Output Latch"; the subscripts M and L stand for "Most significant byte" and "Least significant byte"

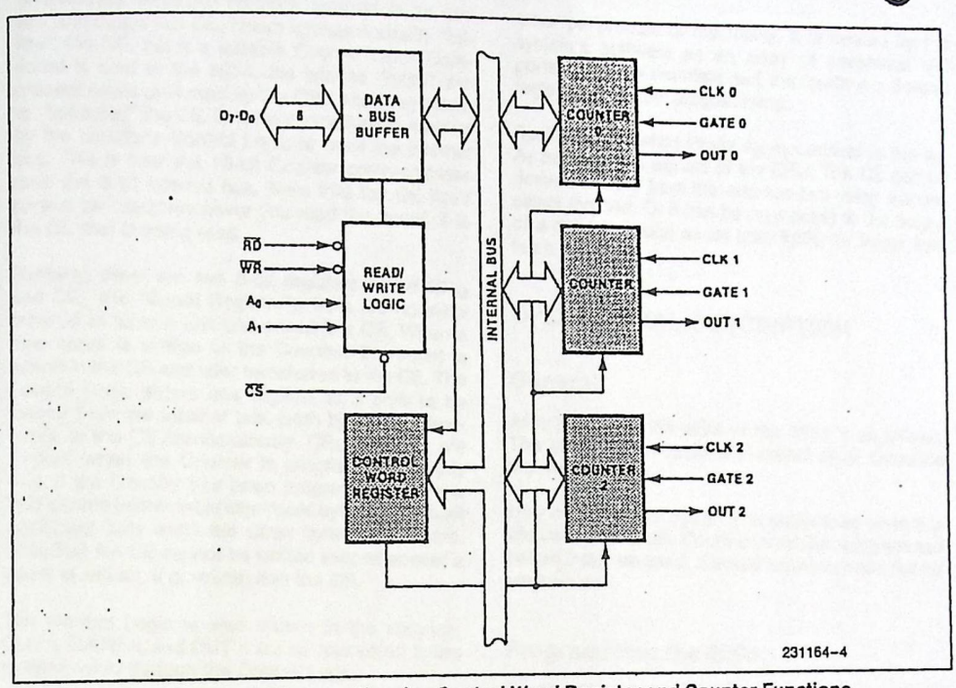


Figure 4. Block Diagram Showing Control Word Register and Counter Functions

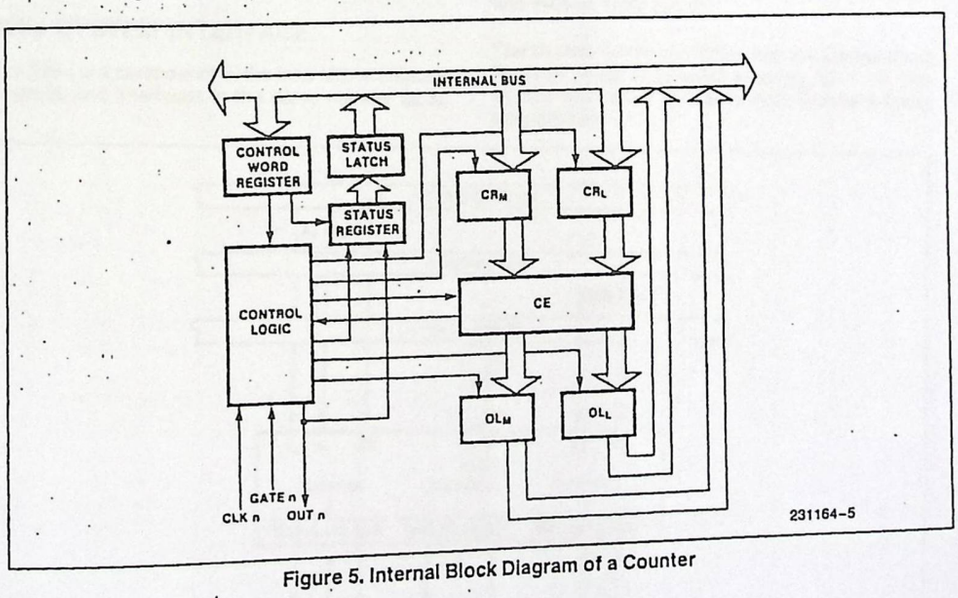


Figure 5. Internal Block Diagram of a Counter

respectively. Both are normally referred to as one unit and called just OL. These latches normally "follow" the CE, but if a suitable Counter Latch Command is sent to the 8254, the latches "latch" the present count until read by the CPU and then return to "following" the CE. One latch at a time is enabled by the counter's Control Logic to drive the internal bus. This is how the 16-bit Counter communicates over the 8-bit internal bus. Note that the CE itself cannot be read; whenever you read the count, it is the OL that is being read.

Similarly, there are two 8-bit registers called CR_M and CR_L (for "Count Register"). Both are normally referred to as one unit and called just CR. When a new count is written to the Counter, the count is stored in the CR and later transferred to the CE. The Control Logic allows one register at a time to be loaded from the internal bus. Both bytes are transferred to the CE simultaneously. CR_M and CR_L are cleared when the Counter is programmed. In this way, if the Counter has been programmed for one byte counts (either most significant byte only or least significant byte only) the other byte will be zero. Note that the CE cannot be written into; whenever a count is written, it is written into the CR.

The Control Logic is also shown in the diagram. CLK n, GATE n, and OUT n are all connected to the outside world through the Control Logic.

8254 SYSTEM INTERFACE

The 8254 is a component of the Intel Microcomputer Systems and interfaces in the same manner as all other peripherals of the family. It is treated by the system's software as an array of peripheral I/O ports; three are counters and the fourth is a control register for MODE programming.

Basically, the select inputs A_0,A_1 connect to the A_0, A_1 address bus signals of the CPU. The CS can be derived directly from the address bus using a linear select method. Or it can be connected to the output of a decoder, such as an Intel 8205 for larger systems.

OPERATIONAL DESCRIPTION

General

After power-up, the state of the 8254 is undefined. The Mode, count value, and output of all Counters are undefined.

How each Counter operates is determined when it is programmed. Each Counter must be programmed before it can be used. Unused counters need not be programmed.

Programming the 8254

Counters are programmed by writing a Control Word and then an initial count.

The Control Words are written into the Control Word Register, which is selected when A_1,A_0 = 11. The Control Word itself specifies which Counter is being programmed.

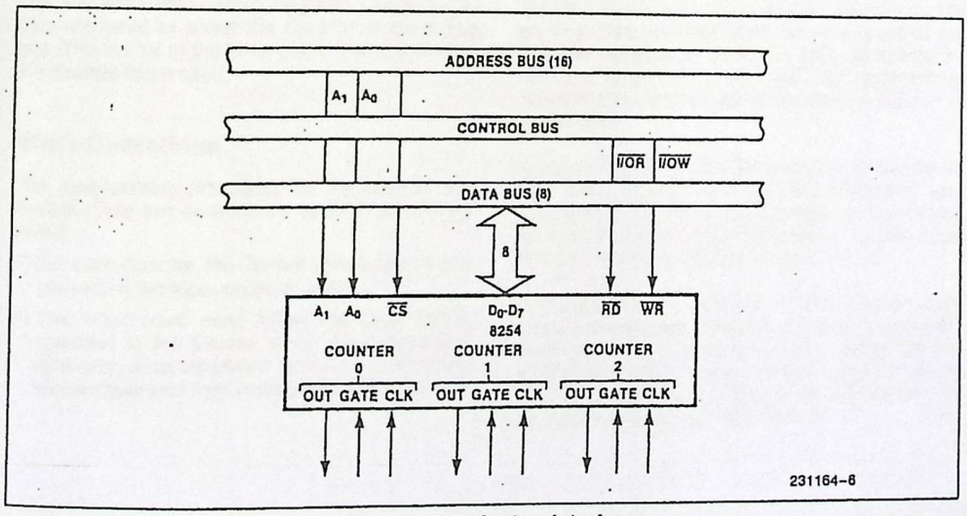


Figure 6. 8254 System Interface

Control Word Format

$A_1,A_0 = 11$ $\overline{CS} = 0$ $\overline{RD} = 1$ $\overline{WR} = 0$

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
SC1	SC0	RW1	RW0	M2	M1	M0	BCD

SC—Select Counter

SC1	SC0	
0	0	Select Counter 0
0	1	Select Counter 1
1	0	Select Counter 2
1	1	Read-Back Command (see Read Operations)

M—Mode

M2	M1	M0	
0	0	0	Mode 0
0	0	1	Mode 1
X	1	0	Mode 2
X	1	1	Mode 3
1	0	0	Mode 4
1	0	1	Mode 5

RW—Read/Write

RW1	RW0	
0	0	Counter Latch Command (see Read Operations)
0	1	Read/Write least significant byte only
1	0	Read/Write most significant byte only
1	1	Read/Write least significant byte first, then most significant byte

BCD

0	Binary Counter 16-bits
1	Binary Coded Decimal (BCD) Counter (4 Decades)

NOTE:
Don't care bits (X) should be 0 to insure compatibility with future Intel products.

Figure 7. Control Word Format

By contrast, initial counts are written into the Counters, not the Control Word Register. The A_1,A_0 inputs are used to select the Counter to be written into. The format of the initial count is determined by the Control Word used.

Write Operations

The programming procedure for the 8254 is very flexible. Only two conventions need to be remembered:

1) For each Counter, the Control Word must be written before the Initial count is written.
2) The initial count must follow the count format specified in the Control Word (least significant byte only, most significant byte only, or least significant byte and then most significant byte).

Since the Control Word Register and the three Counters have separate addresses (selected by the A_1,A_0 inputs), and each Control Word specifies the Counter it applies to (SC0,SC1 bits), no special instruction sequence is required. Any programming sequence that follows the conventions in Figure 7 is acceptable.

A new initial count may be written to a Counter at any time without affecting the Counter's programmed Mode in any way. Counting will be affected as described in the Mode definitions. The new count must follow the programmed count format.

If a Counter is programmed to read/write two-byte counts, the following precaution applies: A program must not transfer control between writing the first and second byte to another routine which also writes into that same Counter. Otherwise, the Counter will be loaded with an incorrect count.

	A_1	A_0
Control Word—Counter 0	1	1
LSB of count—Counter 0	0	0
MSB of count—Counter 0	0	0
Control Word—Counter 1	1	1
LSB of count—Counter 1	0	1
MSB of count—Counter 1	0	1
Control Word—Counter 2	1	1
LSB of count—Counter 2	1	0
MSB of count—Counter 2	1	0

	A_1	A_0
Control Word—Counter 2	1	1
Control Word—Counter 1	1	1
Control Word—Counter 0	1	1
LSB of count—Counter 2	1	0
MSB of count—Counter 2	1	0
LSB of count—Counter 1	0	1
MSB of count—Counter 1	0	1
LSB of count—Counter 0	0	0
MSB of count—Counter 0	0	0

	A_1	A_0
Control Word—Counter 0	1	1
Control Word—Counter 1	1	1
Control Word—Counter 2	1	1
LSB of count—Counter 2	1	0
LSB of count—Counter 1	0	1
LSB of count—Counter 0	0	0
MSB of count—Counter 0	0	0
MSB of count—Counter 1	0	1
MSB of count—Counter 2	1	0

	A_1	A_0
Control Word—Counter 1	1	1
Control Word—Counter 0	1	1
LSB of count—Counter 1	0	1
Control Word—Counter 2	1	1
LSB of count—Counter 0	0	0
MSB of count—Counter 1	0	1
LSB of count—Counter 2	1	0
MSB of count—Counter 0	0	0
MSB of count—Counter 2	1	0

NOTE:
In all four examples, all Counters are programmed to read/write two-byte counts. These are only four of many possible programming sequences.

Figure 8. A Few Possible Programming Sequences

Read Operations

It is often desirable to read the value of a Counter without disturbing the count in progress. This is easily done in the 8254.

There are three possible methods for reading the counters: a simple read operation, the Counter Latch Command, and the Read-Back Command. Each is explained below. The first method is to perform a simple read operation. To read the Counter, which is selected with the A1, A0 inputs, the CLK input of the selected Counter must be inhibited by using either the GATE input or external logic. Otherwise, the count may be in the process of changing when it is read, giving an undefined result.

COUNTER LATCH COMMAND

The second method uses the "Counter Latch Command". Like a Control Word, this command is written to the Control Word Register, which is selected when $A_1,A_0 = 11$. Also like a Control Word, the SC0, SC1 bits select one of the three Counters, but two other bits, D5 and D4, distinguish this command from a Control Word.

$A_1,A_0 = 11$; CS = 0; RD = 1; WR = 0

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
SC1	SC0	0	0	X	X	X	X

SC1,SC0—specify counter to be latched

SC1	SC0	Counter
0	0	0
0	1	1
1	0	2
1	1	Read-Back Command

D5,D4—00 designates Counter Latch Command

X—don't care

NOTE:
Don't care bits (X) should be 0 to insure compatibility with future Intel products.

Figure 9. Counter Latching Command Format

References

Artwick,B.A *Microprocessor Intrefacing*.Englewood Cliffs,NJ.Prentic-Hall,1980

Kurtz,R.L *Microprocessor and Design* Nework:Wily,1909

Lee,AHGYF,yahg *Microprocessor and Microcontroller*,1992

Wakerly,J."Intel MCS-84 *Microprocessor Family* Vol12no

Internet