THE MICROPROCESSOR-CONTROLLED DUPLEX LIFT SYSTEM

by

Abdulkadir OZCAN

B.S. in E.E., Bogazici University, 1981

Submitted to the Institute of Graduate Studies in Science and Engineering in Partial Fulfillment of the Requirements for the Degree of

Master of Science

in

Electrical Engineering

Bogazici University

1984

ACKNOWLEDGEMENTS

I wish to express my gratitude to Dr. Omer CERID for his encouragements, and helpful discussions.

I thank also Dr. Ozdemir GULAN, the director of 'BURKHARD GANTENBEIN GMBH, TURKEY' who helped me with some of the problems of conventional duplex lift systems.

I wish to thank to several individuals who contributed to the final version of my work, Semih BINGOL -a graduate student- , our laboratory technician Cevat GUL, and Fatih COLGAR -a µPhobbyst.

I would like to thank finally Ifakat OZCAN, my wife, for her kindly benefits during my study.

TABLE OF CONTENTS

ABSTRACT

The application areas of microprocessors have been growing recently. In addition to various areas, efficient and reliable results can be obteined in duplex elevator control systems.

The object of this thesis is to apply microprocessor control on a duplex lift system.

Lifts, as known, transport passengens between floors upwards or downwards; consequently these systems dissipate much power. Most of the power is spent by lift motors and partially by control units.The aim of the thesis is to construct a more powerful control system reducing the power consumption and increasing the reliability. Since all logic combinations are considered the performance of the duplex lift system is maximized, the waiting time of passengers is minimized, thus up-down traffic speed is increased.

ÖZET

Son günlerde, mikroişlemcilerin uygulama alanları gittikçe genişlemektedir. Çok çeşitli uygulama alanları yanında, mikroişlemciler ikili (duplex) asansör sistemlerinin kontrolünde de güvenilir ve etkin olarak kullanılır. Mikroişlemcilerin kullanımlarının yaygın olmasının başlıca nedenlerinden birisi, az güç sarfetmeleridir.

Bilindiği gibi asansörler yolcuları aşağı veya yukarı doğru taşırlar. Dolayısıyla bu sistemler çok güç harcarlar. Harcanan gücün büyük bir kısmı asansör motorlarında, kalan kısmı da kontrol ünitelerinde harcanır. Bu tezin ana hedefi mikroişlemci kullanarak kontrol ünitelerindeki güç tasarrufunu azamiye çıkarmak ve yolcuların hizmet bekleme sürelerini asgariye indirmek, dolayısıyle yukarı-aşağı trafik hızını yükseltmektir.

LIST OF FIGURES

LIST OF TABLES

CHAPTER I.

DEFINITION OF THE GENERAL PROBLEM

I.1. The Characteristics of a Duplex Lift System

A duplex lift system is built with two cabins which co-operate in an optimum manner. Automatically both cars adapt to the frequently changing traffic situations in office buildings, hotels, hospitals, e.t.c. The main goal in a duplex lift system is to minimize the waiting time of passengers and to save energy by operating both cars optimally.

First, the cars should be allocated automatically at the optimum waiting floors. The control system divides upward and downward travel into variable zones. The arrangement of these zones depends upon the locations of cabins, the directions of both cars, adapting to the given situation with every scanning cycle of the control system.

Up and down landing calls are registered in a common storage and assigned to the lifts in accordance with the zones momentarely established by the automatic zone allocator.

The 'flexible service zone principle' in a duplex lift system is studied via Figs.I.1., I.2., and I.3. In Fig.I.1. the lift group is at rest. Neither car nor landing calls have been registered. The positions of the cars are the ones which are last served floors. The automatic zone allocator has established zones for allocation of the landing calls in accordance with the locations of the two cars.

As an example, suppose car B has been started downwards in response to an in-cabin or a landing request like in Fig.I.2. Since car B is now no longer available for upward travel the automatic allocator extends the operating zone of car A upwards to the landing call down of the 6^{th} floor.

If any of both lifts is being used for special uses, e.g. for furniture transportation, and is thus disconnected from the common external control by operation of a key switch, the case is shown in Fig.I.3. The automatic zone

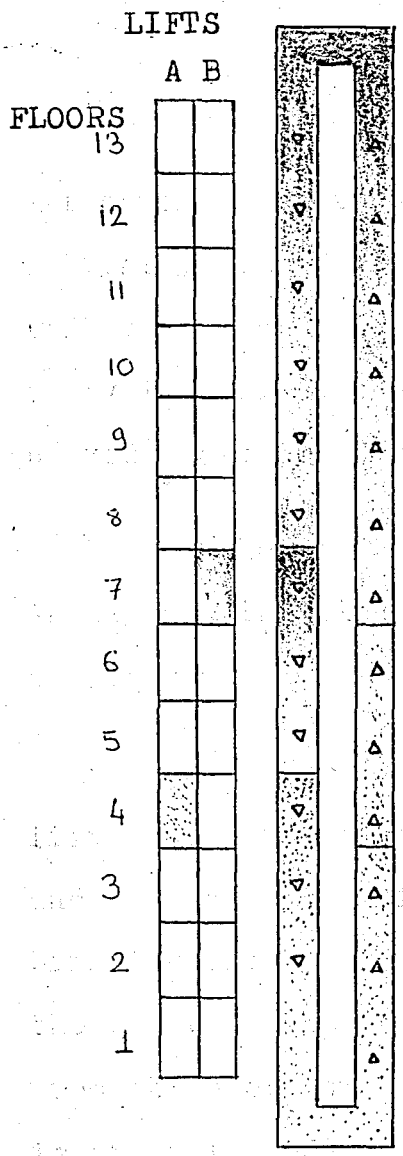


Fig. I.1.

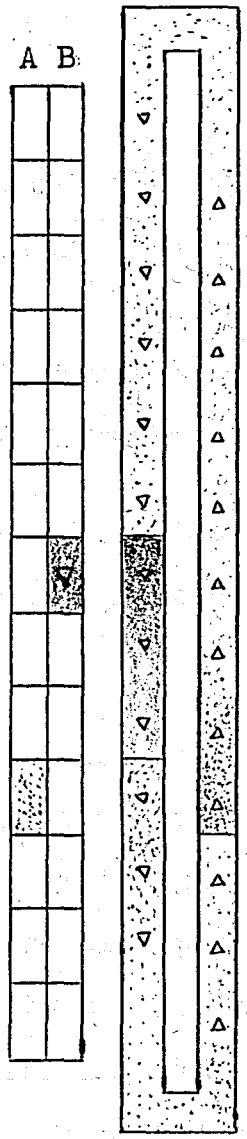


Fig. I.2

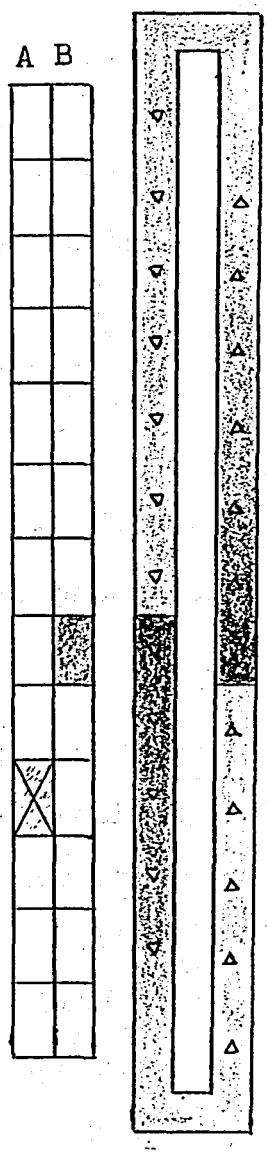


Fig. I.3

△ Up landing call
▽ Down " "

Car with predetermined up or down direction

Car without predetermined direction of travel

Car disconnected from external control

allocator takes into account of this situation by automatically transferring the entire zone allocation to the lift B. The same thing occurs as B is busy or a car is fully loaded. So the flexible service zone principle provides optimum car service distribution.

A duplex lift system has also 'collective' character. As any car is going downwards it must collect all passengers on its way to travel.

The major advantages of the microprocessor based duplex lift system are so called short waiting times, reliability, and low cost operation. Short waiting times are performed by bringing free cars to ideal waiting positions to provide the shortest approach times to calls. The reliability comes from the microcomputer controlling. The cost of the system is greately reduced by using memories instead of some logic gates.

I.2. Problem Statement and Design Goals

The scope of this thesis is to show that microcomputers can be successfully employed in the control of common elevator systems.

The aim is to construct and operate the control of a duplex lift group that has five floors.

All lift signals are maintained in a logical manner and sent to lift motors, doors, passengers, lift operators, e.t.c.

The control system receives and registers 'Incoming Signals' from landing call buttons, in-cabin stop buttons and door light sensors (shown as dashed lines in Fig. I.4). Incoming signals are generated by pushing buttons, keying switches and passing through in front of doors.

'Information signals' display the right informations about the positions of cabins. The information signals are drawn as dotted lines in Fig. I.4.

The microcomputer gathers together all these signals cited before and decides what to do for the lift group.

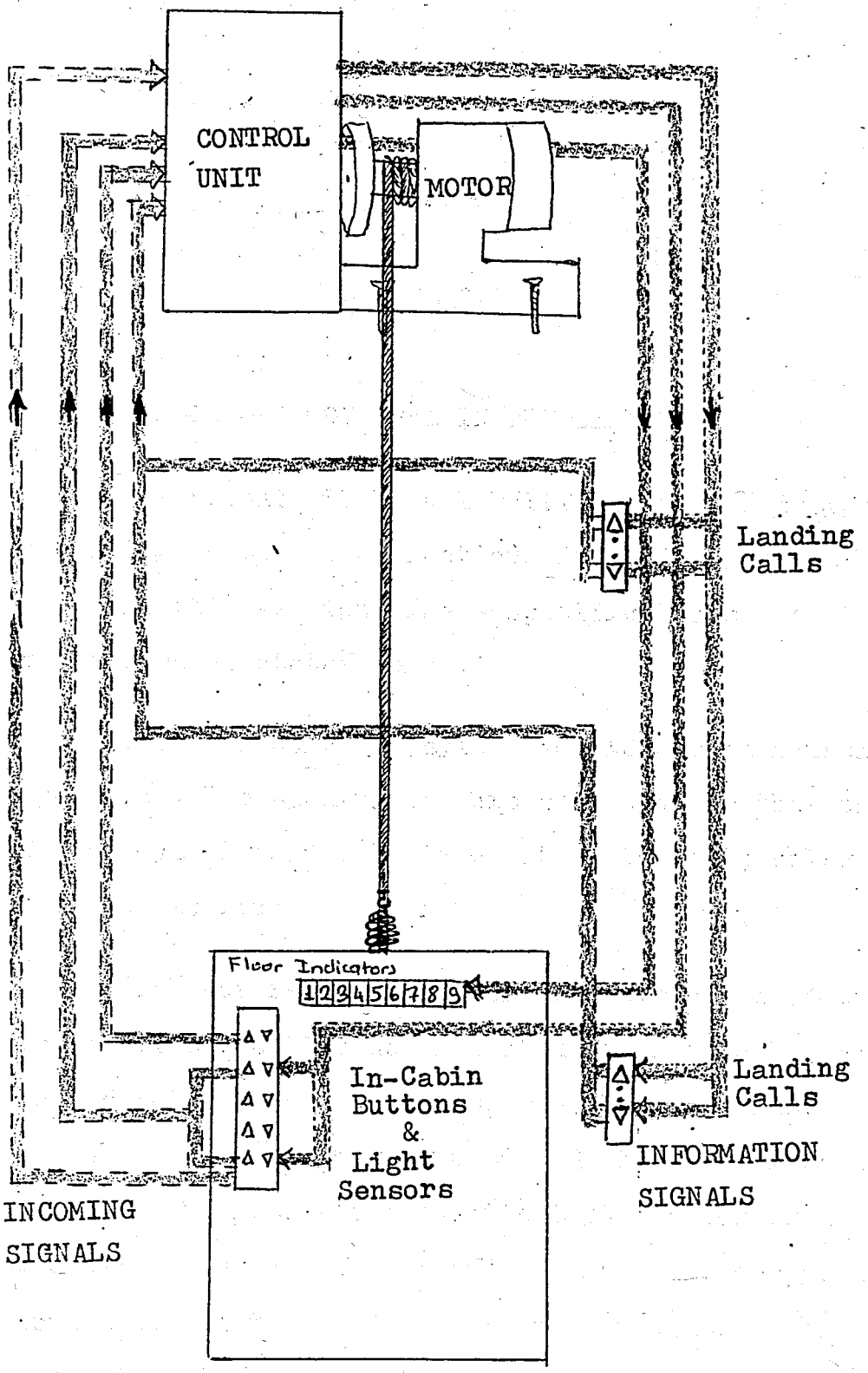

CONTROL
UNIT
MOTOR
Landing
Calls
Floor Indicators
1 2 3 4 5 6 7 8 9
In-Cabin
Buttons
&
Light
Sensors
Landing
Calls
INFORMATION
SIGNALS
INCOMING
SIGNALS

Fig. I.4

CHAPTER II.

METHODOLOGY USED IN THE DESIGN

The µPcontrolled duplex lift system is designed according to the specifications and underlying goals given in this section. These specifications are determined as stated below:

-The microcomputer system should ensure noiseless operation. The meaning of this statement is that the duplex system should be insensitive to heat, coldness, dust and moisture.

-Each car retains the selected direction of travel until all car callsaand allocated landing calls in that direction have been answered. When the last call in one direction has been answered the next direction of travel is reserved automatically. If, for example, the car answers a down landing call at the end of its upward travel the DOWN direction is registered as the next direction of travel. At the same time the DOWN

hall lantern illuminates.

-The buttons and indicators should include:

1. Floor buttons, illuminating as call acceptance; stop buttons; and alarm buttons.
2. Car position indicators above the enterance.
3. One 'UP' and one 'DOWN' call button for each intermediate floor illuminating as call acceptance.
4. Two up or down direction arrows for each floor in front of each cabin door.

-A load measurement circuit with a full-load signal should prevent a fully loaded car from stopping for landing calls.

-Reservation control for disconnecting a lift from the external control during special uses, car cleaning, e.t.c. should be provided.

The microprocessor control should provide all of these specifications which have been mentioned. Under this aspect the main system can be divided into subparts as shown in Fig II.1.

Four types of signals are sent to the CPU and memory unit of the system. These signals come from landing calls, floor indicators, in-cabin calls

and light sensors. The CPU handles these signals logically, and then sends several signals to button displays, lift motors, and doors.

The design procedure is started by describing these sub-parts, and then constructing a flow chart.

Among different choices, the simplest and cheapest ones with minimum number of hardware components are chosen. The emphasise of the system is shifted to software for economy.

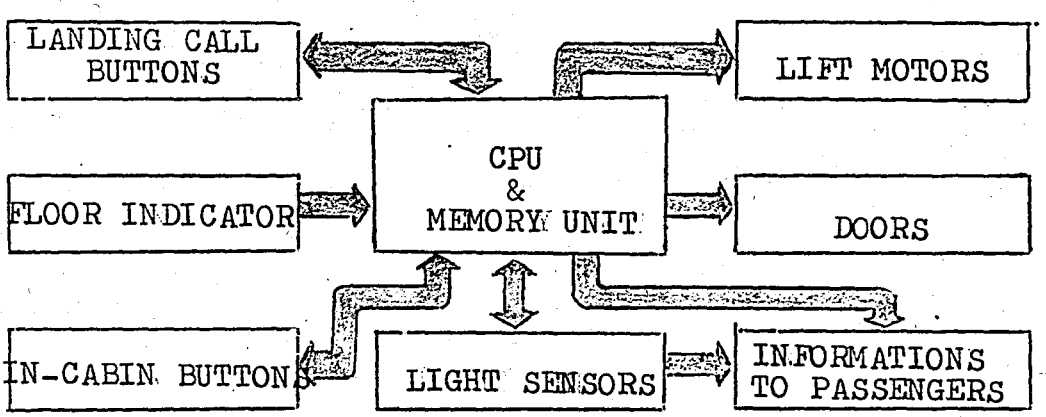


Fig. II.1 System Block Diagram

CHAPTER III.

SYSTEM HARDWARE

The block diagram of the system is shown in Fig. III.1. The microcomputer system consist of:

1. One microprocessing Unit (CPU)
2. Two Read Only Memories (ROM)
3. Three Parallel Input/ Output Interface Circuits (PIA).
4. One Address Decoder Circuit.

This block diagram represents all of the hardware required for a fully operational microcomputer system. The 'Data Bus' is shared between all devices in the system. The 'Control Bus' is shared by all devices which get their required signals from the bus. Different combinations of signals may be received from the 'Address Bus' to define where each device is located.

The study will first begin with the I/O devices in the following sections.

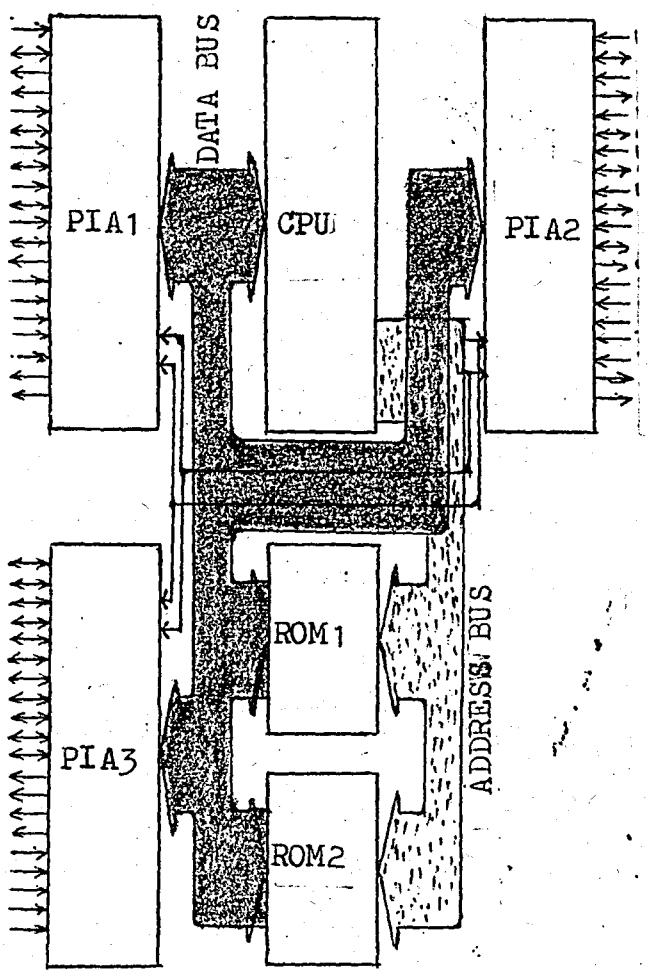


Fig. III.1 The Control System

III. 1. Buttons and Switches

Buttons transmit signals to the CPU as they are pressed. According to hardware characteristics the buttons can be studied in three types:

A. Push Buttons

These are In-Cabin Stop Request, Landing Call, and Alarm Buttons.

'In-Cabin' buttons send requests to the CPU to move the cabin towards the desired floor. When the CPU admits the request the corresponding LED lights, the cabin will be locked and started towards the floor whose button has been pressed.

'Landing Call' buttons signal to the CPU that someone would like to go to another floor from the one whose button has been pushed. The CPU chooses the most suitable cabin, and starts it immediately towards the passenger. To indicate that the pushed landing call has been accepted the LED of the pushed button will light. If not so, it means both cars have been occupied by preceding requests.

'Alarm Buttons' are pushed when any dangerous

III. 1. Buttons and Switches

Buttons transmit signals to the CPU as they are pressed. According to hardware characteristics the buttons can be studied in three types:

A. Push Buttons

These are In-Cabin Stop Request, Landing Call, and Alarm Buttons.

'In-Cabin' buttons send requests to the CPU to move the cabin towards the desired floor. When the CPU admits the request the corresponding LED lights, the cabin will be locked and started towards the floor whose button has been pressed.

'Landing Call' buttons signal to the CPU that someone would like to go to another floor from the one whose button has been pushed. The CPU chooses the most suitable cabin, and starts it immediately towards the passenger. To indicate that the pushed landing call has been accepted the LED of the pushed button will light. If not so, it means both cars have been occupied by preceding requests.

'Alarm Buttons' are pushed when any dangerous

condition occurs. A red LED begins to flash on and off to indicate the emergency.

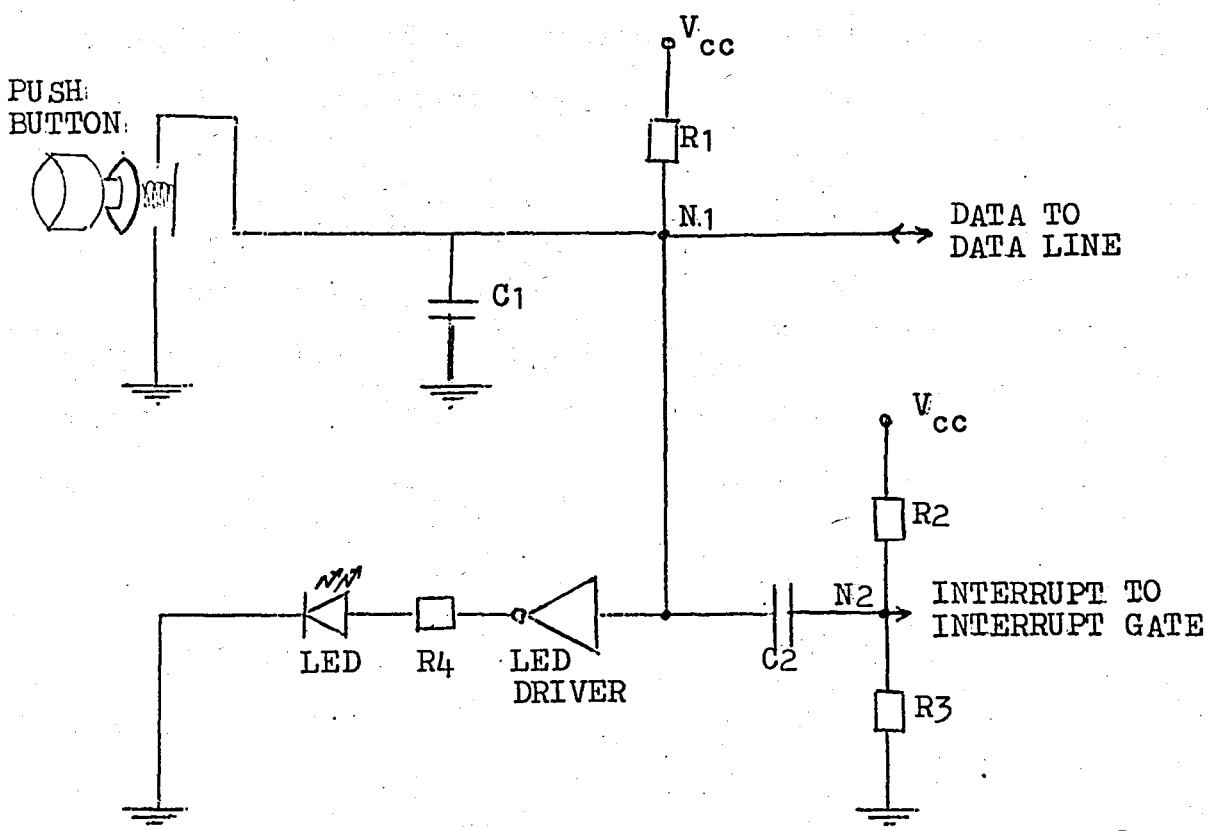
All of these three types of push buttons are connected to bidirectional PIA data lines.

The push buttons explained here are designed as shown in Fig. IIIL 2.

As the button is free there is some voltage on the capacitor C1 so that it is enough to pull the PIA data line to logic 1 level. The voltage at Node 2 is 3.4 Volts, the inverter output is low, and the LED is turned off. If the button is pressed the voltage at Node 1 drops to zero, theLED lights showing that someone wants a car.On the other hand the voltage at Node 2 drops to -1 V. temporarily, then rises to 3.4 V. This negative going pulse feeds to the interrupt gate, causing the interrupt bit of the PIA is set. If any interrupt is received the CPU scans all PIA data lines to find out the pressed button. The CPU checks whether the request is acceptable or not. If acceptable it changes the corresponding PIA line to output; so the LED remains lighted.

Since the voltage on the capacitor C_2 has been charged to 3.4 V multiple presses do not create interrupt

Fig.III.2 The Circuit Diagram of Landing Calls,
In-Cabin Stop Request, and Alarm Buttons.



pulses any longer. The interrupt pulse is generated if and only if the direction of the data line is programmed as input.

B. Alarm Clear Button

The 'Alarm Clear' button is located in the operator's room to clear the alarm after the emergency has been overcome. The circuit diagram is shown in Fig, III.3. As pushed the voltage at Node 2 drops to nearly zero Volt, then rises to logic 1 level. This negative going pulse sets the interrupt bit in the control register of the PIA.

C. Special Transportation & Stop Switches

They have two states. When the routine is finished they should be turned off. These types of switches send interrupt pulses not only as turned on but also as turned off. The circuit diagram is shown in Fig. III.4.

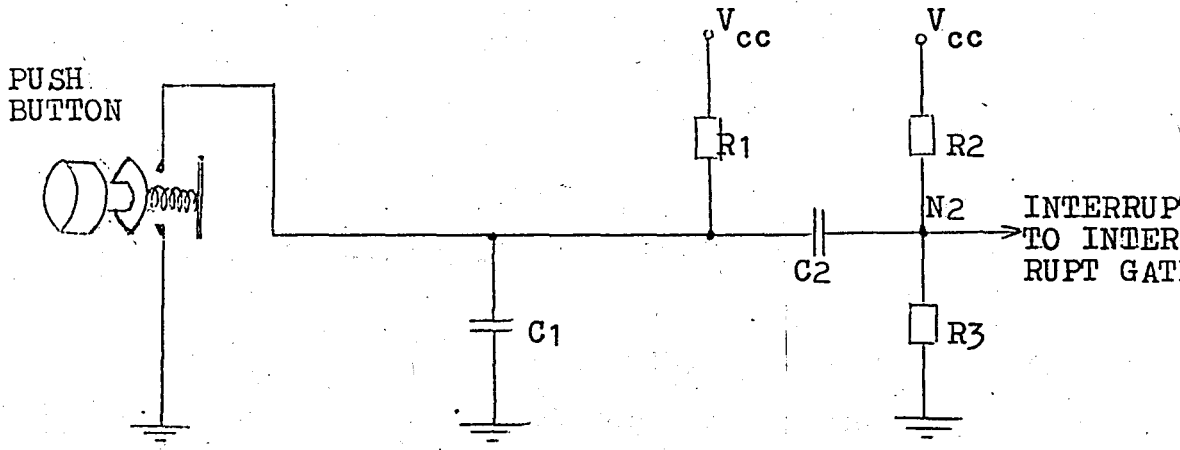


Fig. III.3 Push Button for Alarm Clear.

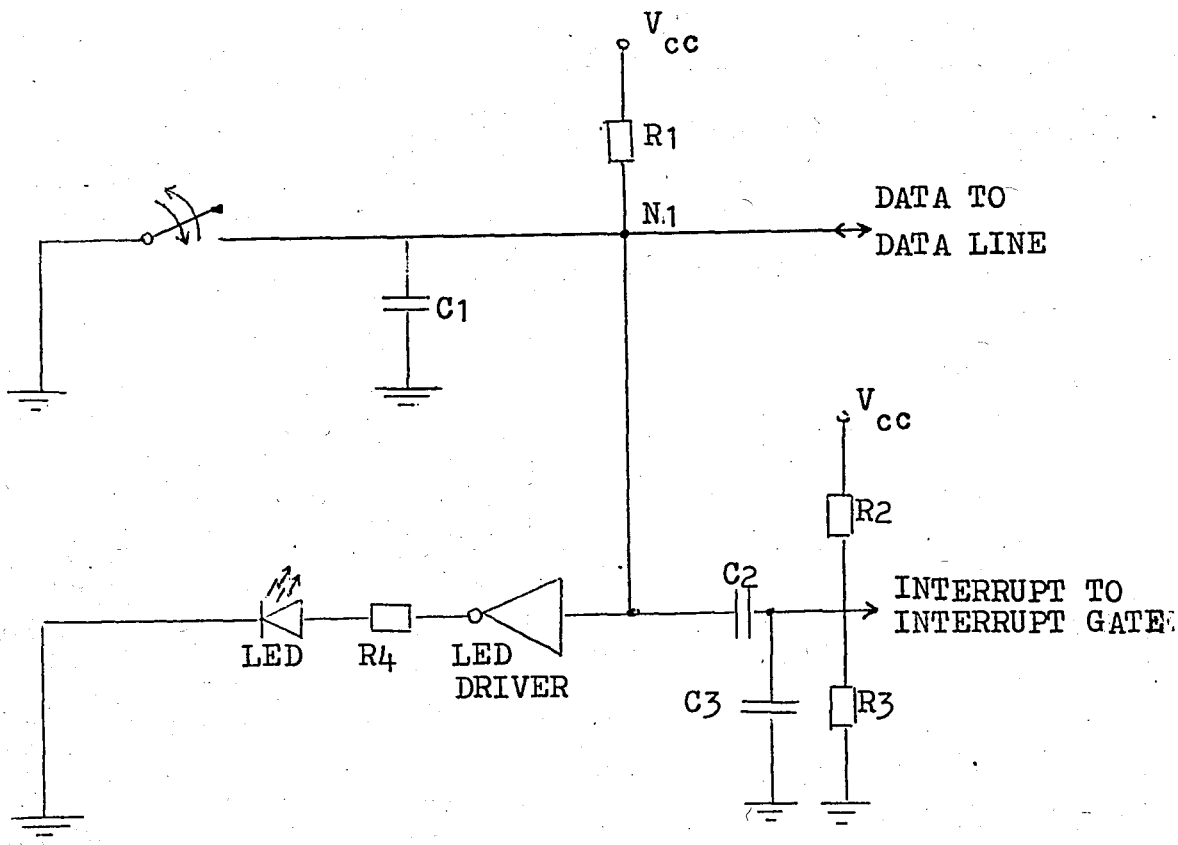


Fig. III.4 The Special Transportation and Start/Stop Switch.

III.2. Light Sensors

Light sensors are mounted in front of cabin doors of both lifts. They are used for counting the passengers entering or exiting the cabins. A seven segment display is connected to the outputs of each light sensor to indicate the number of passengers within any cabin.

To detect the direction of motion two sensors for each door are built up. The block diagram is drawn in Fig. III.5.

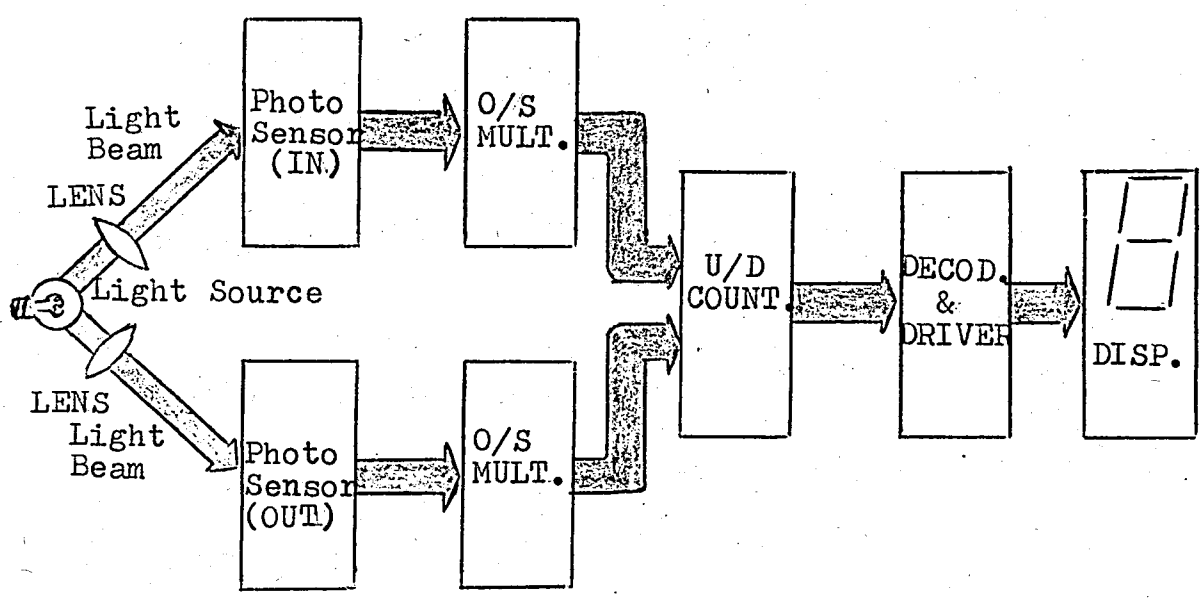


Fig. III.5 The Block Diagram of the Light Sensor.

A light beam produced by a light source is split and directed towards two phototransistors. If the incident beam path is broken due to such as a person passing through, a signal is produced. This is fed to the trigger input of the one-shot multivibrator to get the count pulse. The counter counts up or down by one depending on the direction of motion.

The circuit diagram of the light sensor is shown in Fig. III.6. When the beam falls on the phototransistor the collector to emitter resistance decreases to a few hundred ohms. If the beam path is broken the collector to emitter resistance increases to several megohms, causing relatively high positive voltage at the input of inverter N1. The steep edge of the rapidly falling output is differantiated by C1, R2, and R3 to produce a sharp pulse. This pulse is then inverted by the inverter N3 and applied to the count-up input of the counter. The pulse is also used to hold the other one-shot at reset for awhile not to generate a down count pulse.

Each successive breaking of the beam causes the counter to count up by one.

The down counting part of the circuit operates like the previous one.

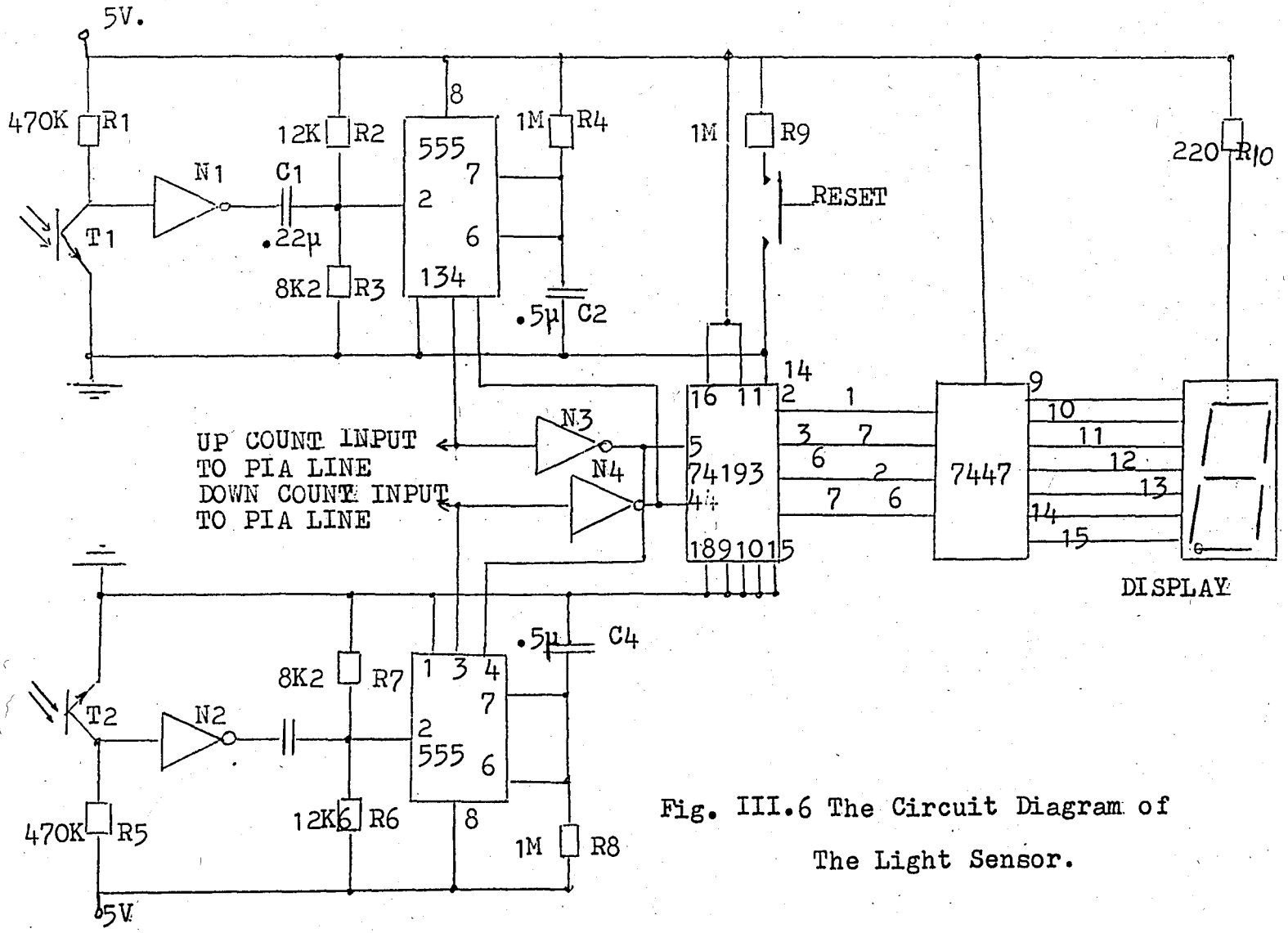


Fig. III.6 The Circuit Diagram of The Light Sensor.

The BCD outputs of the Up/Down counter are connected to a driver&decoder circuit to drive a seven segment display, individuals thus are aware of the number of passengers in cabins.

The outputs of both light sensors are fed to PIA data lines and interrupt gate. As the control program is executed the interrupt pulses can be sent to the CPU. If the CPU is appropriate to handle the count interrupt it jumps to scan the light sensor outputs, finds out the direction of motion, and finally increments -or decrements- by one the number of passengers stored in the RAM.

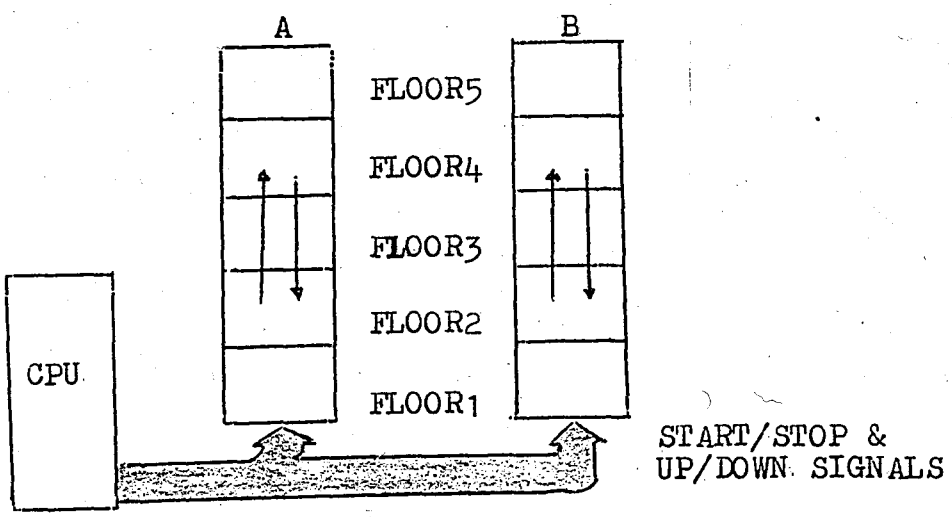
The transportation capacity of each cabin is supposed to be six in this thesis; although it can be easily changed by changing the software.

III.3 The Duplex Lift Simulator

The duplex lift simulator has been constructed to see the movement of the cabins in laboratory, since this thesis doesn't include the interface circuitry between the control part and a real lift group.

The lift simulator is designed with shift registers and their outputs are connected to LEDs. A "1"may be shifted up or down, or stopped to replace the lift's motion. The simulator also contains "Lock/Unlock Door", "Door Open/Closed", and "Overloaded" signals. Here only the duplex lift simulator part is explained, while the others will be studied in the software sections.

Fig. III.7 The Block Diagram of the Lift Simulator.



The block diagram of the lift simulator is shown in Fig. III.7, and the circuit diagram in Fig. III.8.

The CPU loads the lift simulator at restart. The directions of the cabins should be set to UP position before loading. All the inputs of shift registers are set to zero except the first input; the MODE CONTROL input is pulled up; and then the LOAD pulse is applied to the CLK2 inputs of all shift registers. Thus all zeros and a "1" at the inputs are loaded to the outputs.

An astable multivibrator has been designed to generate shift clock pulses.

If the CPU sets the UP output to high the shift right clock pulses are enabled and the "1" simulating the cabin begins to shift right. The same thing appears as the DOWN signal is applied.

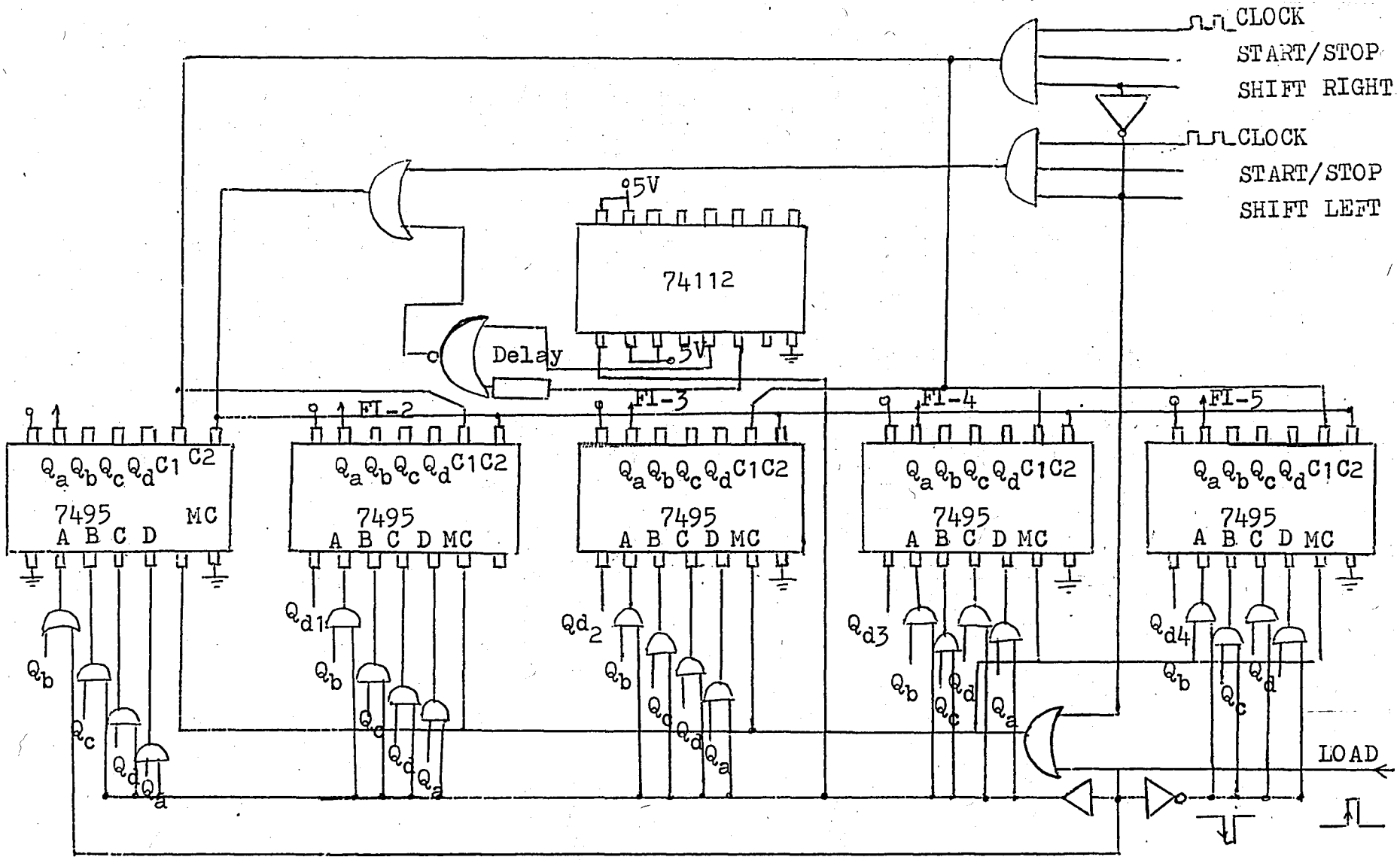


Fig. III.8 The Circuit Diagram of the Lift Simulator.

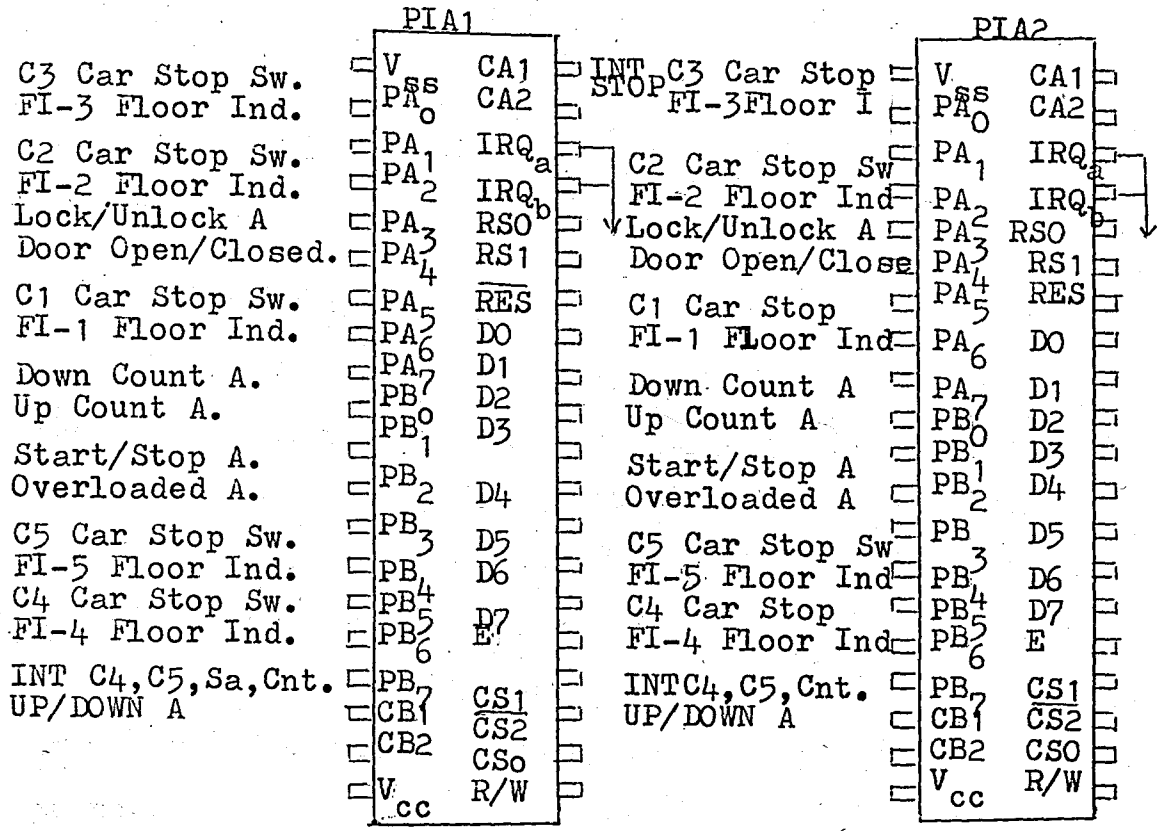
III.4 The CPU & Memory Unit

The CPU and memory part of the lift control system is constructed with: Microprocessing Unit, Read Only Memories, Peripheral Interface Adapters, and Addres Decoder circuits. Fig. III.1 shows the block diagram of the microcomputer system. The detailed I/O configuration is redrawn in Fig. III.9.

MC 6802 is the heart of the system. It is a monolithic 8-bit microprocessor performing the centralcontrol function for the system. The MPU can execute 72 different instructions including arithmetic, logical, shift, rotate, load, store, conditional or unconditional branch, jump, interrupt, and stack manipulation instructions.

The MC 6802 has seven addressing modes that can be used by the programmer, with the addressing mode a function of both the type of the instruction and the coding within the instructions.

A RAM of 128-bytes and a clock generator is also present in the MC 6802.



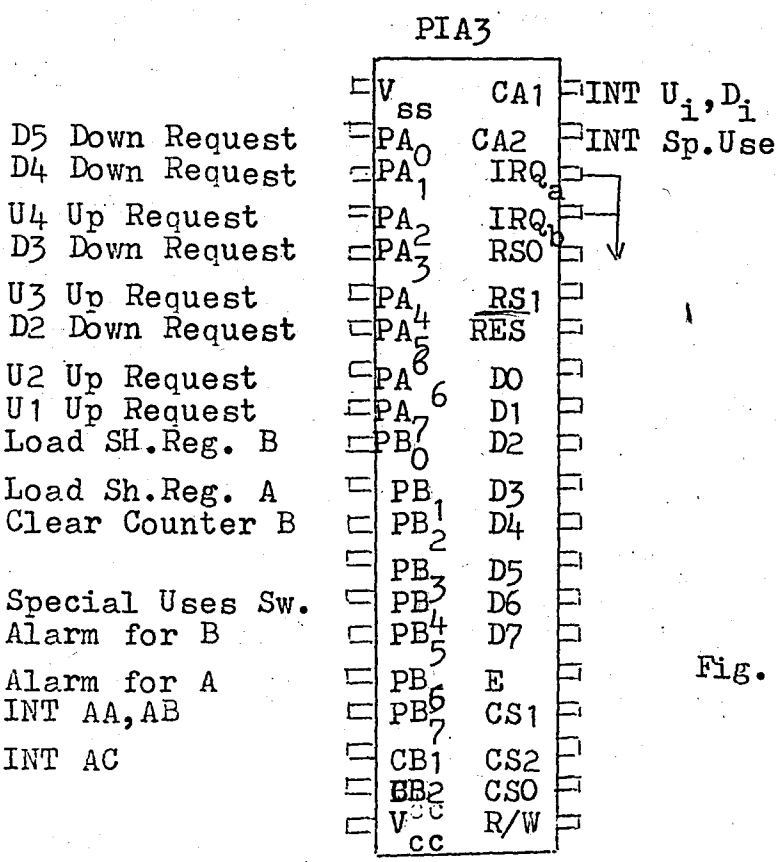


Fig. III.9 I/O Lines of PIAs.

The PIA I/O lines can be divided into three types according to their directions:

A. Inputs:

1. Floor Indicators: They stay for indicating where the lifts are; namely F.I.1, F.I.2, F.I.3, F.I.4, and F.I.5.

2. Door Open/Closed Inputs: They show the doors of the cabins are closed or not. The lift won't start if its door is open.

3. Start/Stop Signals: They come from the Start/Stop switches mounted in the cabins to stop the lift for some purposes.

4. Passenger Count Signals: They are sent via light sensors to count the passengers in cabins. These signals are only enabled when the doors are open. The CPU counts up or down by one depending on the direction of motion of passengers.

5. Interrupt Inputs: Interrupt signals are created by pushing the request buttons. The interruptsignals are sent to PIAs using CA1 and CB1 inputs.

B. Outputs:

1. Lock/Unlock Doors: As the CPU decides the door of a cabin shall be locked it sends a zero to lock. A lock relay can be commanded using this signal.

2. Overloaded A/B: They indicate the bearing capacity of a cabin is exceeded or not. Only six passengers are allowed to travel; otherwise the lift ignores all requests and doesn't start.

3. Start/Stop A/B: They are used to start or stop the lifts. If the CPU sets the line to "1", the corresponding lift begins to move in the intended direction.

4. Up/Down A/B: These outputs are connected to the shift right/left inputs of the lift simulator.

5. Clear Counters: The CPU clears the passenger counters at restart by sending a pulse to CLEAR inputs of the counters.

6. Load Shift Registers: These outputs send pulses to load the lift simulator at the initialization.

C. Bidirectional Data Lines:

1. In-Cabin Buttons: They are mounted in each cabin signalling which floor one would like to reach. The in-cabin buttons are named as C1, C2, C3, C4, and C5. These are programmed as inputs at the initialization. When pressed, they send both informations to data lines and interrupts to interrupt gates. If the request is acceptable the MPU changes the direction of the line to output. The direction is reverted back to input after the request has been sreved.

2. Landing Calls:They are located in front of cabin doors to indicate a passenger demands a lift. Landing call

buttons are named as U1, U2, U3, U4, and D5, D4, D3, D2.

3. Alarm A/B Buttons: As pushed the CPU changes the direction of the data line to output and lights the ALARM lamp.

4. Special Use Switch: This switch is turned on to reserve the lift A for some special uses.

III.5 The Address Decoder and the Address Map

Decoding addresses is done by using a 3 to 8 decoder. The inputs of the address decocer are A_{15}, A_{14}, and A_{11}. Here A_{15} is the MSB, and A_{11} is the LSB. The pinouts of the decoder is drawn in Fig. III.10.

There exist three PIAs and two EPROMs to be addressed. The locations are chosen as:

PIA1: 80 00 - 80 03

PIA2: 80 04 - 80 07

PIA3: 80 08 - 80 0B

ROM1: F0 00 - F7 FF

ROM2: F8 00 - FF FF

The EPROM decoding is achieved using only $\overline{E}$/Prog pins. The Output Enable($\overline{G}$) pins are grounded.

The RAM in the chip is continuously enabled by connecting the RE input to V_{cc}.

The address map is drawn in Table III.1.

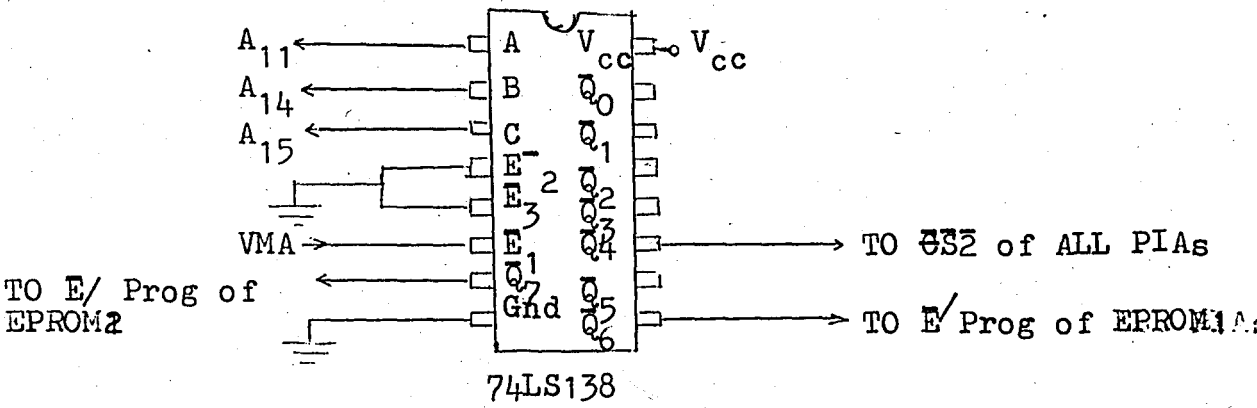


Fig. III.10 The Address Decoder.

		A_{15} ... A_0	
PIA1	80 00 80 03	1000 0000 0000 0000 1000 0000 0000 0011	A3 CS0 A2 CS1 Q4 CS2
PIA2	80 04 80 07	1000 0000 0000 0100 1000 0000 0000 0111	Vcc CS0 A2 CS1 Q4 CS2
PIA3	80 08 80 0B	1000 0000 0000 1000 1000 0000 0000 1011	Vcc CS0 A3 CS1 Q4 CS2
EPROM1	F0 00 F7 FF	1111 0000 0000 0000 1111 0111 1111 1111	Q6 E Gnd G
EPROM2	F8 00 FF FF	1111 1000 0000 0000 1111 1111 1111 1111	Q7 E Gnd G

Table III.1 The Address Map.

CHAPTER IV.

SYSTEM SOFTWARE

The program of the Duplex Lift Control System includes two parts : Main Program and Interrupt Service Routine. In this chapter the two parts of the program will be described completely.

When the system is turned on the prodessor begins to execute the Main Program whose flow chart is shown in Fig. IV.1. The routine can be divided into seven parts as stated below:

1. Restart and Initialization.
2. Bringing Cars to Optimum Waiting Floors.
3. Allowing Interrupts for Service Requests.
4. Scanning.
5. Starting Appropriate Lifts to Serve Requests.
6. Testing the Reached Floor.
7. Clearing the Requests Which Have Been Served.

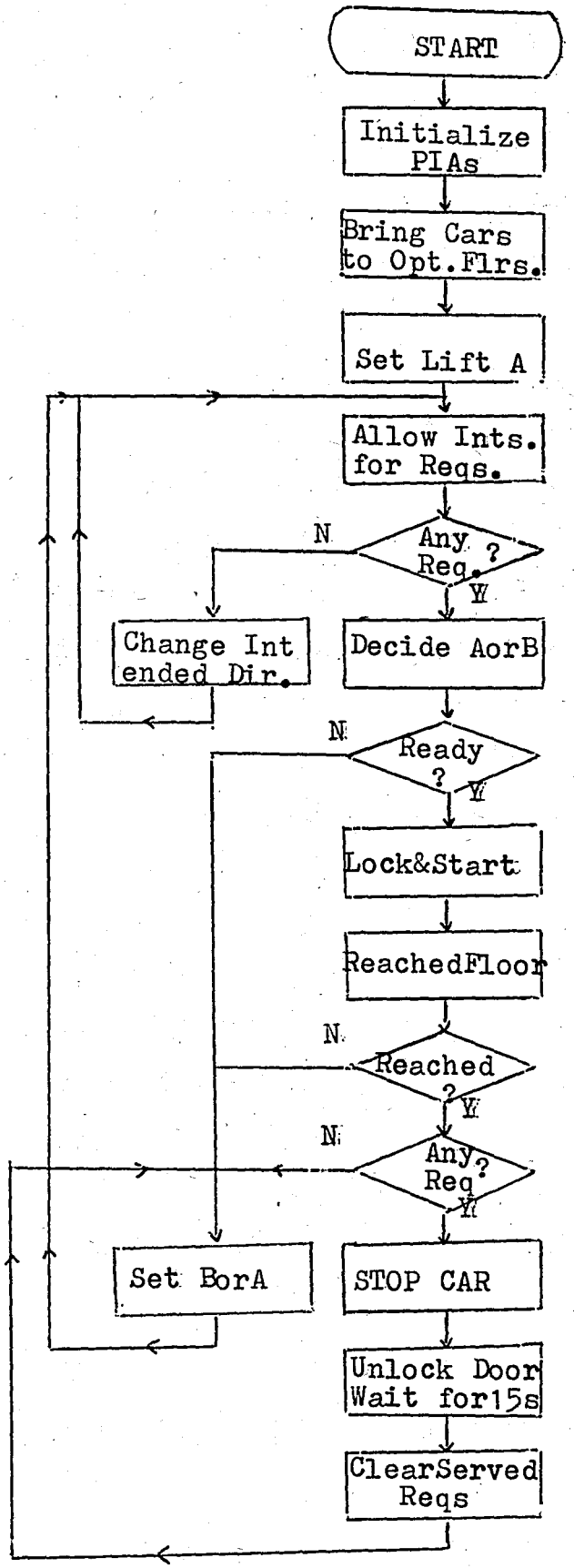


Fig. IV.1 The Flow Chart of the Whole Program.

All of these sub parts will be studied in subsequent sections. The algorithm of the program has been constructed for five-floored duplex systems; however it could be easily expanded for higher floored systems.

IV.1 Restart and Initialization

When the system is turned on the MPU clears and saves enough memory area in the 128-byte RAM for stack requirements and temporary data storage. The PIAs are initialized, the lift simulator is loaded, the passenger counters are cleared.

IV.2. Bringing Cars to Optimum Waiting Floors

The optimum waiting positions are determined as the 1^{st} floor for the lift A, and the 4^{th} floor for B. The flow chart for bringing cars to these floors is drawn in Fig. IV.2. As seen, the MPU should execute some subroutines listed below:

1. Ready to Start A/B Subroutine
2. Lock Door Subroutine
3. Unlock Door Subroutine.

Since they will be used frequently in later routines it would be useful to explain them briefly.

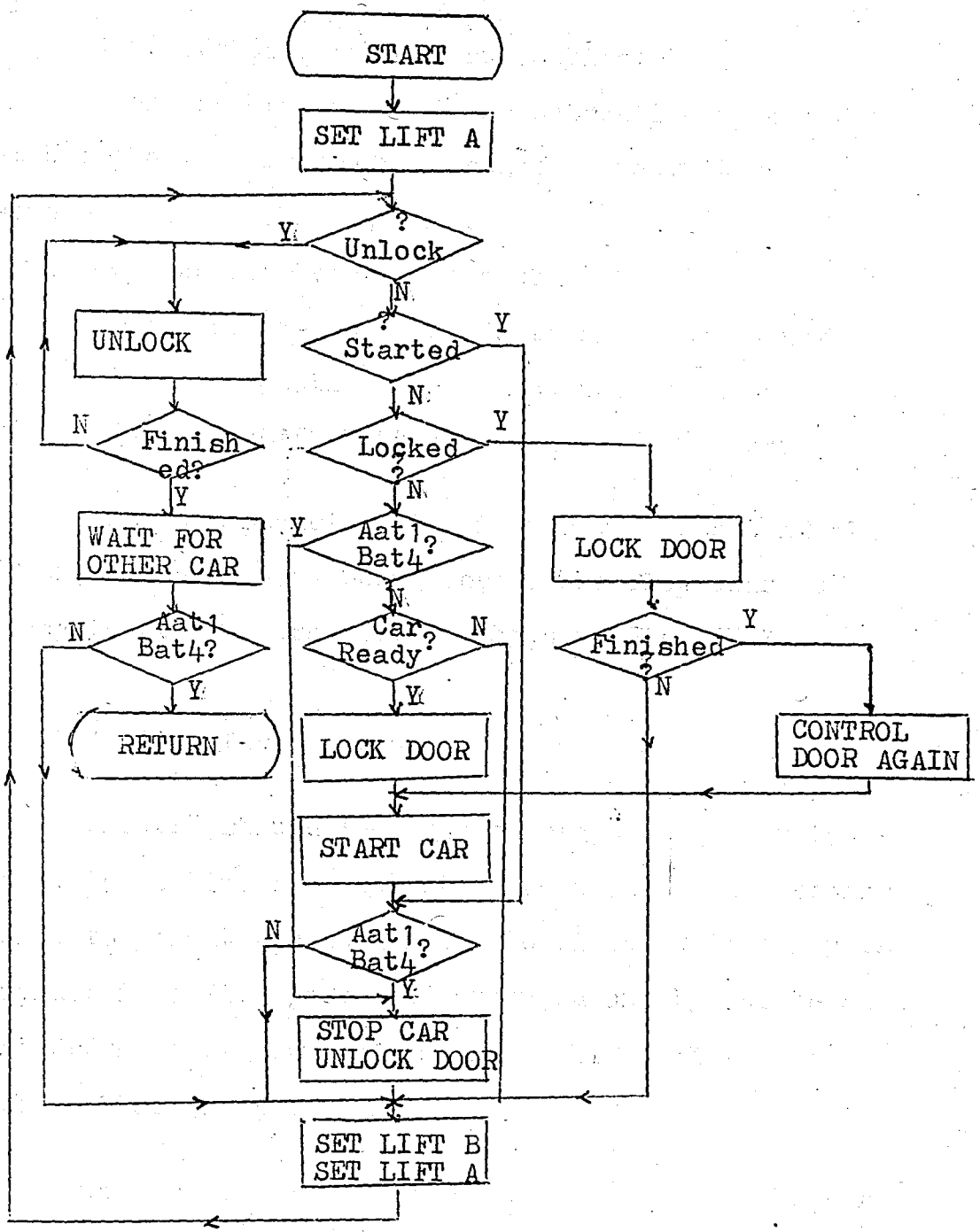


Fig. IV.2 Bringing Cars to Optimum Waiting Positions.

1. Ready to start A/B subroutine:

When the MPU executes the subroutine it decides either the lift is ready to start or not. It checks some data such as:

-Is the door closed or not?

-Does the overloading condition exist or not?

-Is the in-cabin STOP switch turned on or not?

-Has the ALARM button been pressed or not??

The corresponding lift is ready to start if all of these checks have been passed. The flow chart of Ready to Start subroutine is shown in Fig. IV.3.

2. Lock Door Subroutine:

The CPU should lock the corresponding door to start the cabin. The subroutine contains a half second delay to permit the lock relay to close. This delay is divided into five parts and executed one by one because the CPU must be aware of the other lift, while the lock relay delay is executed.

The processor again controls the door to be sure that it is closed and locked at the end of the delay.

The flow chart of Lock Door Subroutine is shown in Fig. IV.4.

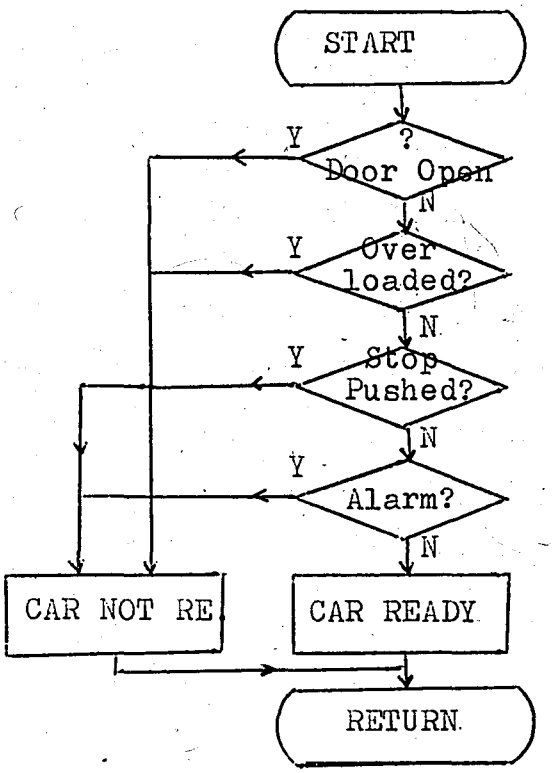


Fig. IV.3 "Ready to Start" Subroutine

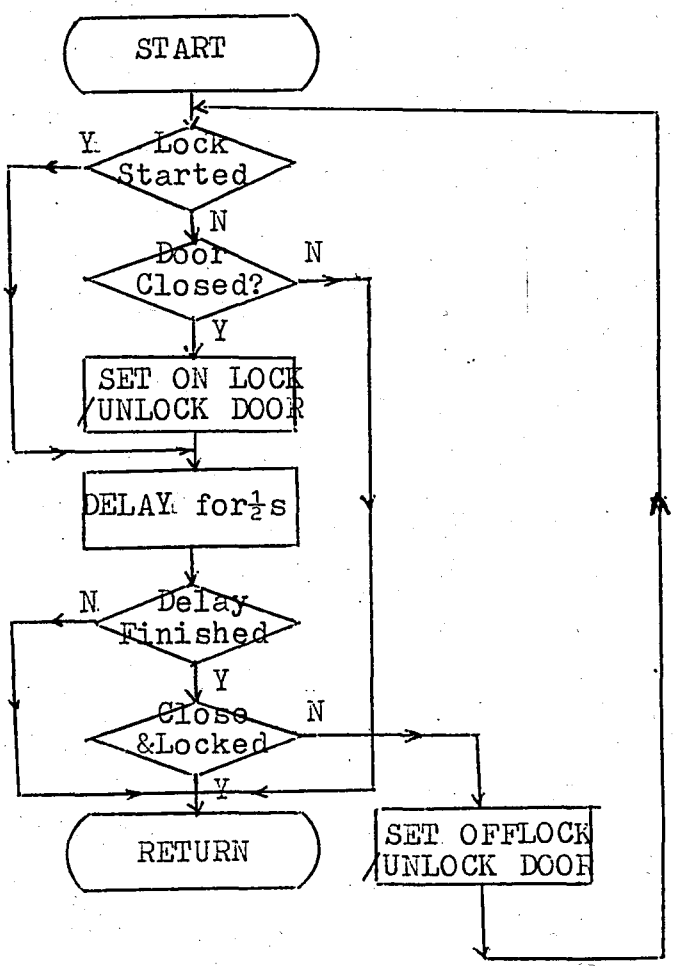


Fig. IV.4 "Lock Door"Subroutine

3. Unlock Door Subroutine:

This subroutine is called when the car has reached the floor where it will stop. Again the subroutine contains three kinds of delays: Motor Stop Delay (1 sec), Unlock Relay Delay (1 sec), and waiting (10secs).

The flow chart of Unlock Door Subroutine is drawn in Fig. IV.5.

IV.3 Allowing Interrupts for Service Requests

The CPU clears the interrupt mask bit to allow interrupts before scanning. All request lines have been programmed as inputs at the initialization. If any button is pushed an interrupt pulse is generated to the CPU and the corresponding data line falls to zero. The CPU jumps to the "Interrupt Service Routine" shown in Fig. IV.6.

The interrupt service routine begins to scan all PIA data lines to catch the low line. It reverts the direction of the low line to output to indicate that the request has been accepted.

If the interrupt pulse comes from the count inputs, it increases or decreases the number of passengers in the lift, compares it with six, and decides whether overloading condition exists or not.

IV.4 Scanning

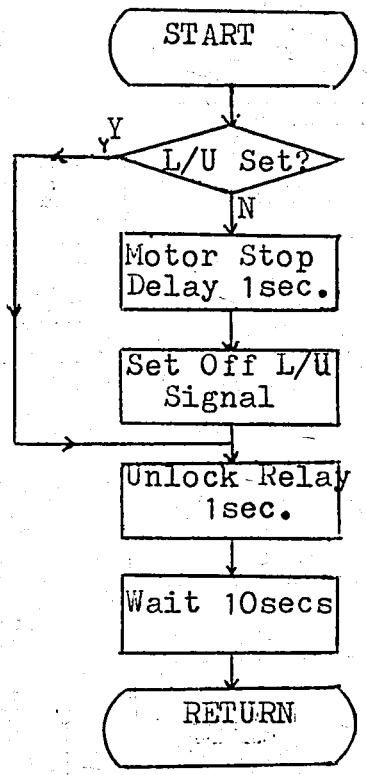


Fig. IV.5 The Unlock Door Subroutine.

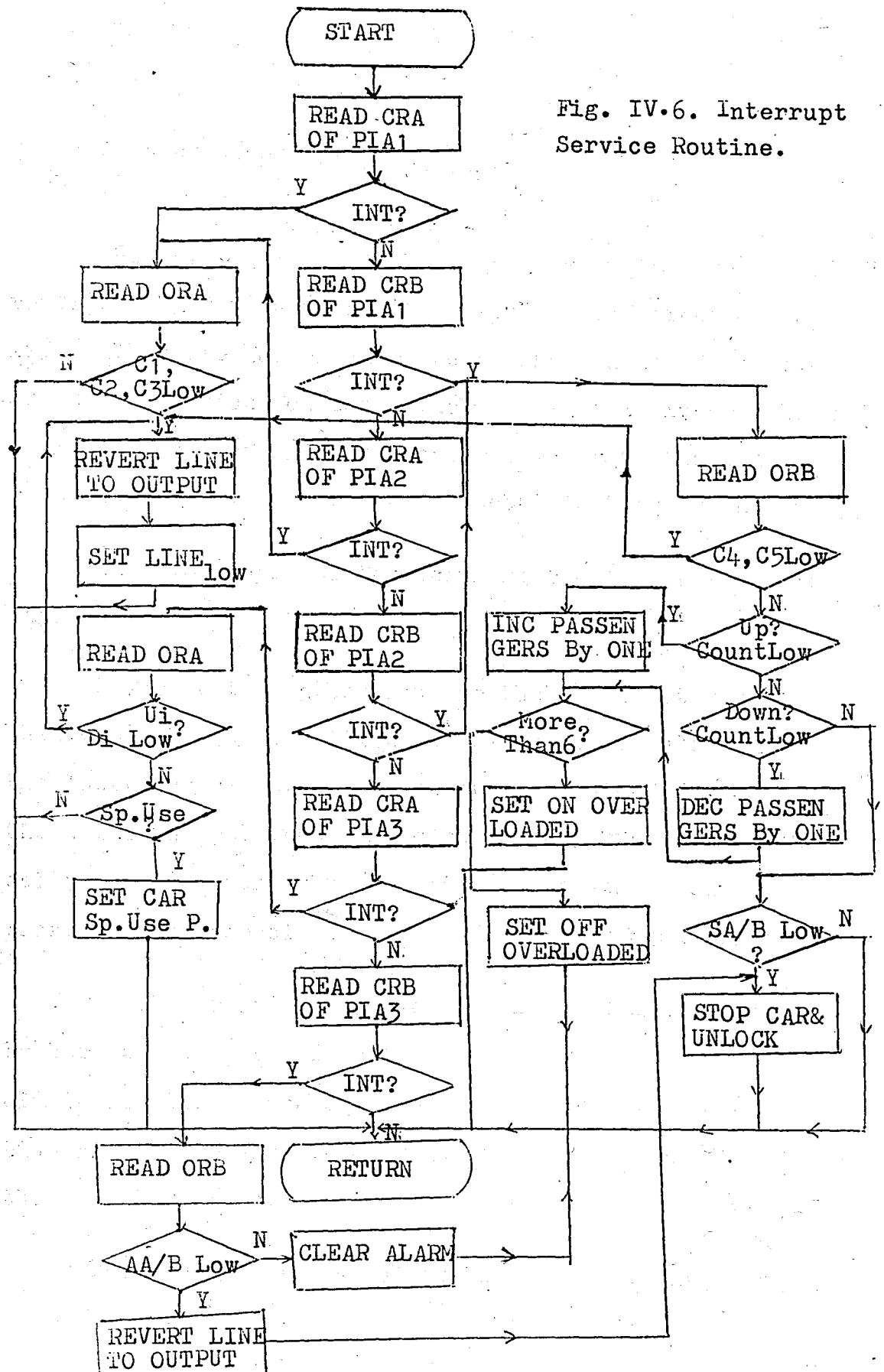


Fig. IV.6. Interrupt Service Routine.

IV.4 Scanning

The largest part of the main program is the "Scanning" routine. The CPU scans all the inputs and decides which lift will serve for which request. The routine first finds out the conditions of cars. The cars may be either unlocked or locked, or moving. The processor jumps to the Scanning routine if the car is at rest without locking or unlocking.

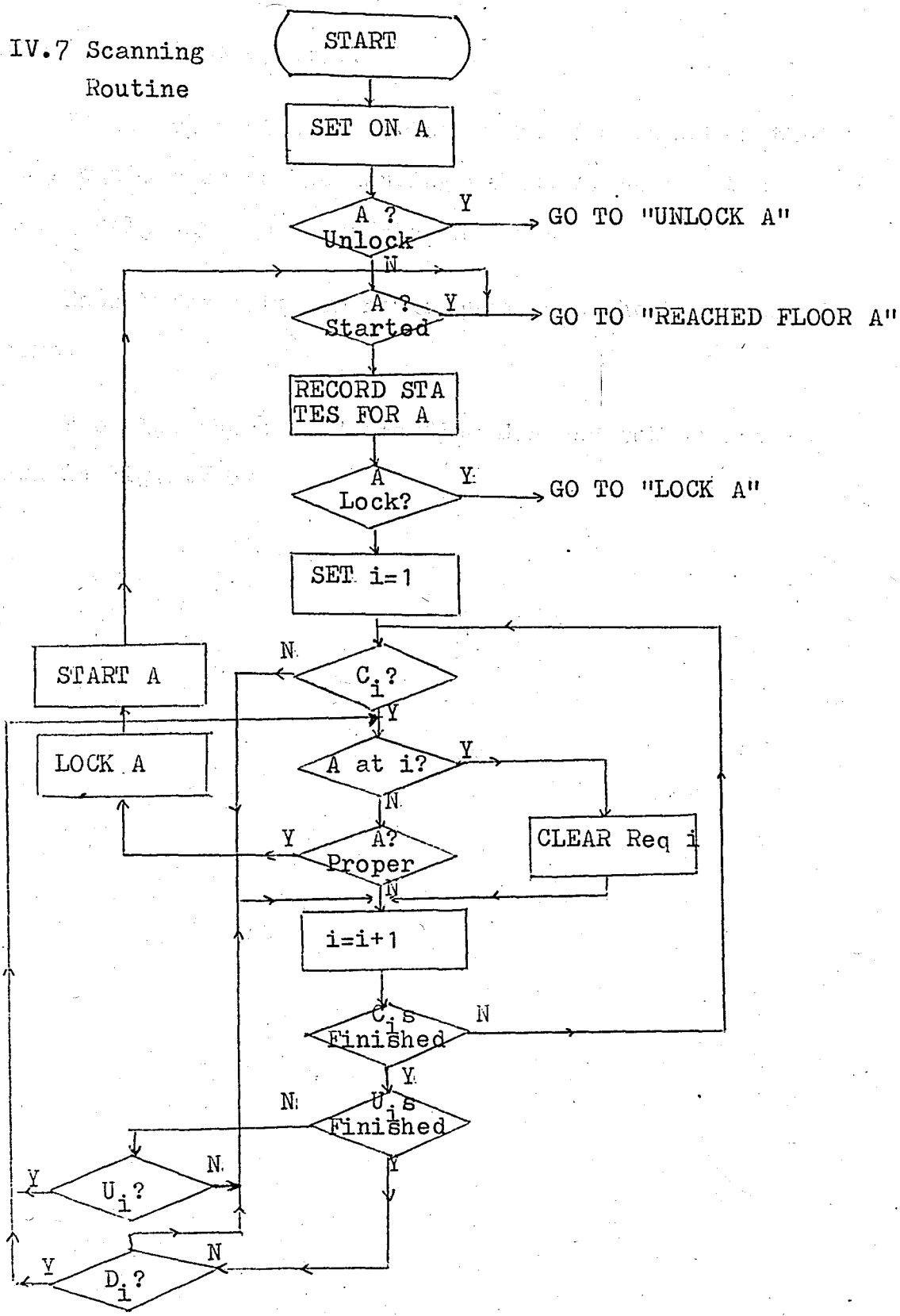
The flow chart of the Scanning routine is shown in Fig. IV.7.

Before entering the routine the CPU first records the states of the two lifts. These recorded data include the locations of both cars, the predetermined directions, the number of passengers in each cabins, and the remainder delays of the lock or unlock relay. The stored data provides the ease of testing conditions frequently.

The priority is given to the In-Cabin calls rather than the Landing calls. The scanning routine eliminates the requests which are at the opposite direction and the service is only given to those which are in the intended direction.

The requests which belong to the floor where the

Fig. IV.7 Scanning Routine



cabin is shall be ignored.

If the special transportation key is turned on the lift A won't respond the landing calls. If so all the landing calls are performed by one lift.

Both lifts wait for new requests at the last served floors.

The flow chart of "Recording States" subroutine is shown in Fig. IV.8.

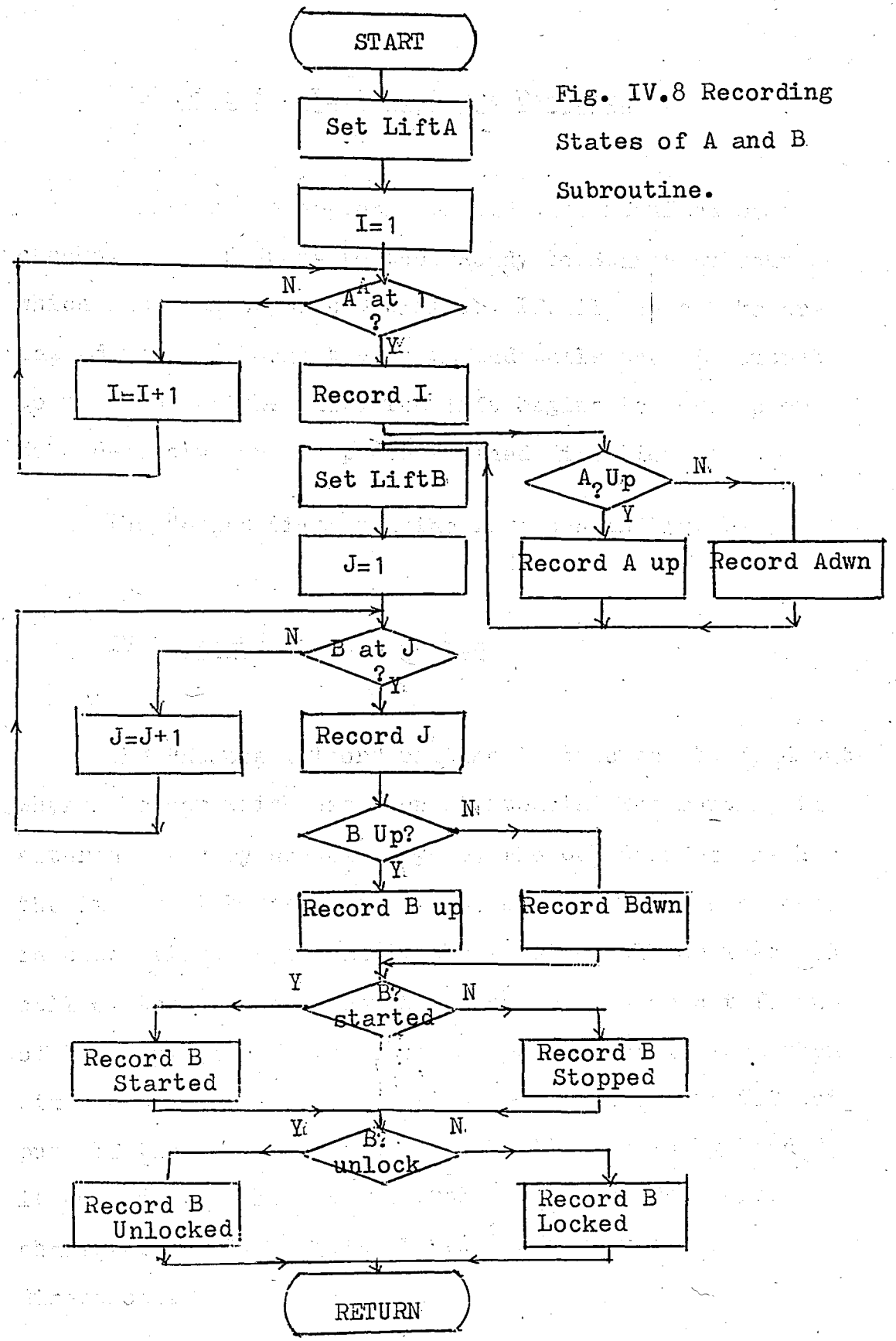


Fig. IV.8 Recording States of A and B Subroutine.

IV.5 Starting Lift to Serve Requests

Starting the appropriate car also requires some checks. The CPU jumps to the "Ready to Start" subroutine which has been explained earlier. If all the checks are passed the CPU locks the door, and pulls the CA2 output to up to start the lift. The lift begins to move up or down depending on the predetermined direction.

The "Start Lift" routine is drawn in Fig. IV.9.

IV.6 Reached Floor Routine

The "Reached Floor" routine is executed to find out where the car which has been started is. The routine is entered at every scanning cycle. The CPU decides whether the lift will be stopped or not when any floor has been reached. First it checks if the landing call or in-cabin call of the reached floor has been pressed or not. If one of both exists and the number of passengers is less than six the CPU sends the STOP pulse to the lift via CA2 output. If there is no request or the lift is fully loaded, it continues. Since the control system has "collective" character the lifts collect waiting passengers on their directions.

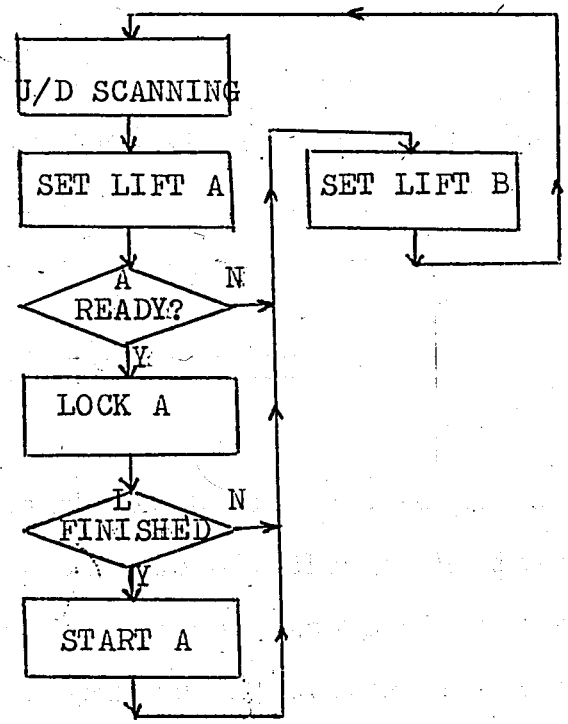


Fig. IV.9 Start Car Routine

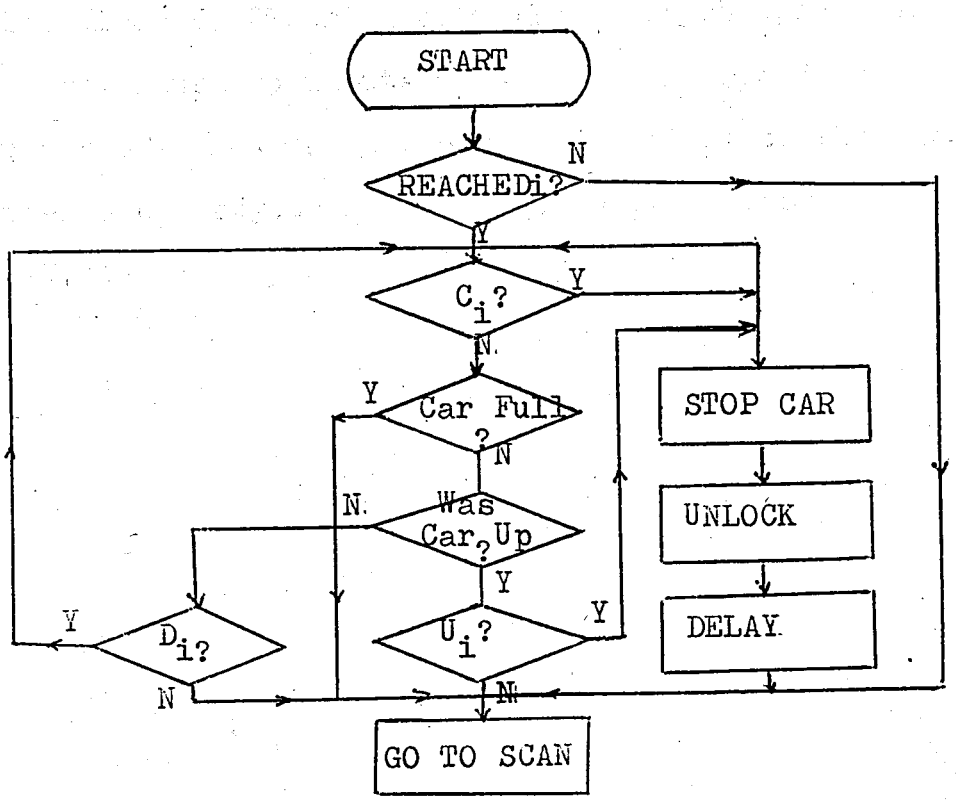


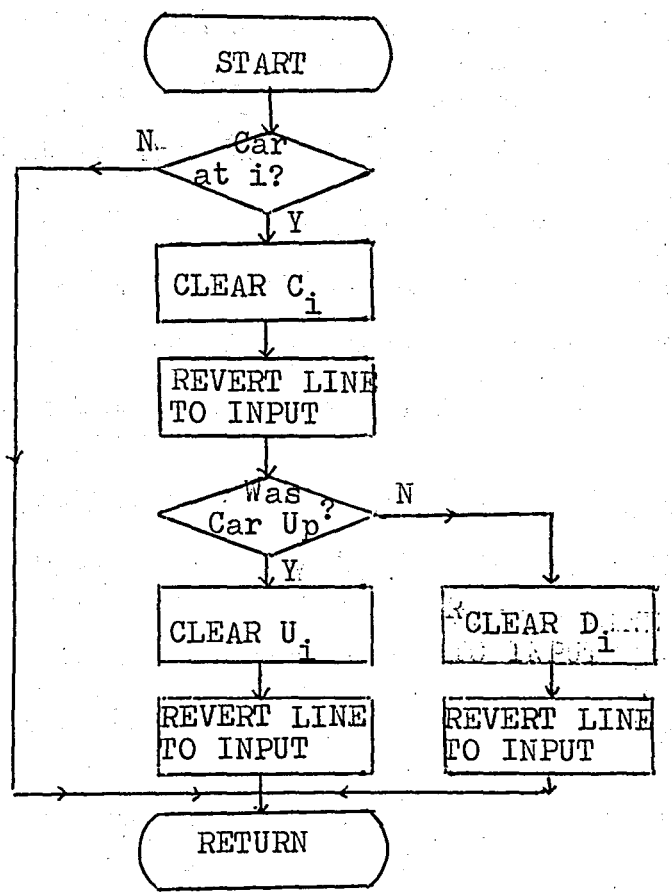
Fig. IV.10 Reached Floor Routine.

The flow chart of Reached Floor routine is shown in Fig. IV.10.

IV.7 Clearing the Served Requests

The served requests should be cleared to prepare the buttons for new requests. The CPU clears the in-cabin requests which belong to the reached floors without taking into account of the predetermined directions. The landing calls which are matched with the directions of the lifts are also cleared. Clearing actually means changing the data output lines to inputs; thus the corresponding LEDs are turned off. Now the buttons which belong to the cleared PIA lines are ready to receive successive presses. (Fig. IV.11).

Fig. IV.11 Clearing the Served Requests



CHAPTER V.

RESULTS AND CONCLUSIONS

V.1 Results

So far a microprocessor controlled duplex lift system has been designed and built.

With the start of the design implementation many unforseen problems arose. The supply problem has been seen very important to feed the microprocessing unit; although it is less pronounced for TTL devices. The supply problem has been held with attention since the system is considerably large. V_{cc} supply voltage has been refined by using bypass capacitors near to the CPU.

Choosing capacitor and resistor values are important to send the interrupt pulses and the data without debouncing.

All the TTL devices concerned with the CPU have been chosen as LS TTL ones in order not to overload the μP bus.

Two independent clock pulse generators have been

constructed since the two lift simulators operate independently.

In this design three PIAs have been used completely for I/O requirements between peripherals and the CPU.

26 buttons and switches have been used for various purposes; so a lot of capacitors and resistors have been required to prevent debouncing problem.

The data from the light sensors have been first taken directly from the counter outputs to PIA data lines; but afterwards it has been seen advantageous to count via interrupt pulses to save I/O lines.

V.2 Possible Variations in the Design and Recommendations

Several variations in the design are possible. Only some of them will be mentioned here.

In this thesis the duplex system has been designed for only five floors. It would be possible to build up a control system for six, seven, or higher floors; just adding some parts into the software. Enlarging the system is straightforward.

The maximum number of passengers that a lift can bear has been chosen as six; although the program is flexible for changing this number.

Another variation could be the method of counting passengers. One may cancel the light sensors and install weight sensors instead. Actually present lift systems use this method.The disadvantage of counting people via light sensors is that the CPU may not detectttwo individuals who are passing through side by side. In this case the counter would count wrong. However counting via light sensors is seen deceiving in some conditions, it is fairly acceptable because the bearing capacity of a lift is held in big margins.

In this study the safety requirements have been also considered. The meaning of this statement is that what would happen to the system if any component of the central control unit had unfortunately burnt? If any such unexpected accident about the CPU chip or PIAs occurs, the control system would fail. The worst case is that the two lifts would continue their ways without stopping at any floor until they would reach the first or the last floors. The safety system automatically stops the lifts at these floors by ANDing the CPU's Start/Stop output and the floor indicatorsbelonging those floors. The other method of stopping the cars is to turn on the In-Cabin Stop switches mounted in each cabin.

The power-down condition has not been considered in this study. One may expand the study writing an adequate non-maskable interrupt program into the software, and adding a continuous supply.

The most important result of this thesis is the potential for further studies.

BIBLIOGRAPHY

1. J. Millmann and H. Taub, "Pulse, Digital, and Switching Waveforms", Mc. Graw-Hill, N.Y., 1975.
2. J. Millmann and C.C. Halkias, "Integrated Electronics", Mc. Graw-Hill, Kogakusha, Tokyo, 1978.
3. Texas Instruments Inc., "The Integrated Circuits Catalogue", 1972.
4. Motorola Inc., "The Integrated Circuits Catalogue", 1975.
5. Texas Instrument Inc.,"The Memory and Microprocessor Data Book", 1976.
6. Motorola Inc.,"MEK 6802D5E Microcomputer Evaluation Board User's Manual", U.S.A., 1980.

MOTOROLA Semiconductors

MC6821 (1.0 MHz)
MC68A21 (1.5 MHz)
MC68B21 (2.0 MHz)

Abdülkadir ÖZCAN

MOS

(N-CHANNEL, SILICON-GATE, DEPLETION LOAD)

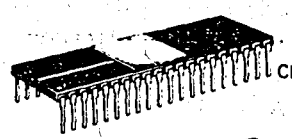
PERIPHERAL INTERFACE ADAPTER

PERIPHERAL INTERFACE ADAPTER (PIA)

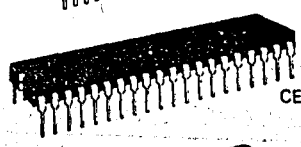
The MC6821 Peripheral Interface Adapter provides the universal means of interfacing peripheral equipment to the M6800 family of microprocessors. This device is capable of interfacing the MPU to peripherals through two 8-bit bidirectional peripheral data buses and four control lines. No external logic is required for interfacing to most peripheral devices.

The functional configuration of the PIA is programmed by the MPU during system initialization. Each of the peripheral data lines can be programmed to act as an input or output, and each of the four control/interrupt lines may be programmed for one of several control modes. This allows a high degree of flexibility in the overall operation of the interface.

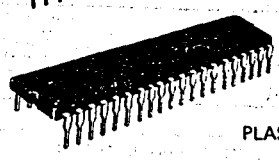
- 8-Bit Bidirectional Data Bus for Communication with the MPU
- Two Bidirectional 8-Bit Buses for Interface to Peripherals
- Two Programmable Control Registers
- Two Programmable Data Direction Registers
- Four Individually-Controlled Interrupt Input Lines; Two Usable as Peripheral Control Outputs
- Handshake Control Logic for Input and Output Peripheral Operation
- High-Impedance Three-State and Direct Transistor Drive Peripheral Lines
- Program Controlled Interrupt and Interrupt Disable Capability
- CMOS Drive Capability on Side A Peripheral Lines
- Two TTL Drive Capability on All A and B Side Buffers
- TTL-Compatible
- Static Operation



L SUFFIX
CERAMIC PACKAGE
CASE 715

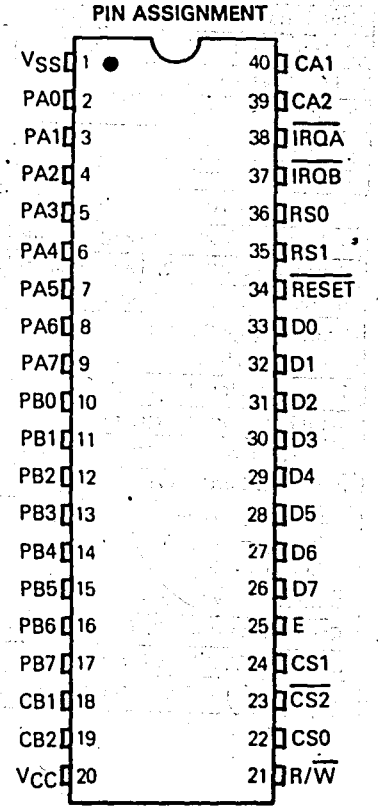


S SUFFIX
CERDIP PACKAGE
CASE 734



P SUFFIX
PLASTIC PACKAGE
CASE 711

PIN ASSIGNMENT



MAXIMUM RATINGS

Characteristics	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range MC6821, MC68A21, MC68B21 MC6821C, MC68A21C, MC68B21C	T_A	T_L to T_H 0 to 70 −40 to +85	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Ceramic Plastic Cerdip	θJA	 50 100 60	°C/W

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage (i.e., either V_{SS} or V_{CC}).

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \bullet \theta_{JA}) \quad (1)$$

Where:

T_A ≡ Ambient Temperature, °C

θ_{JA} ≡ Package Thermal Resistance, Junction-to-Ambient, °C/W

$P_D \equiv P_{INT} + P_{PORT}$

$P_{INT} \equiv I_{CC} \times V_{CC}$, Watts — Chip Internal Power

P_{PORT} ≡ Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} \ll P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K + (T_J + 273°C) \quad (2)$$

Solving equations 1 and 2 for K gives:

$$K = P_D \bullet (T_A + 273°C) + \theta_{JA} \bullet P_D^2 \quad (3)$$

Where K is a constant pertaining to the particular part. K can be determined from equation 3 by measuring P_D (at equilibrium) for a known T_A. Using this value of K the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

DC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 Vdc ±5%, V_{SS} = 0, $T_A = T_L$ to T_H unless otherwise noted).

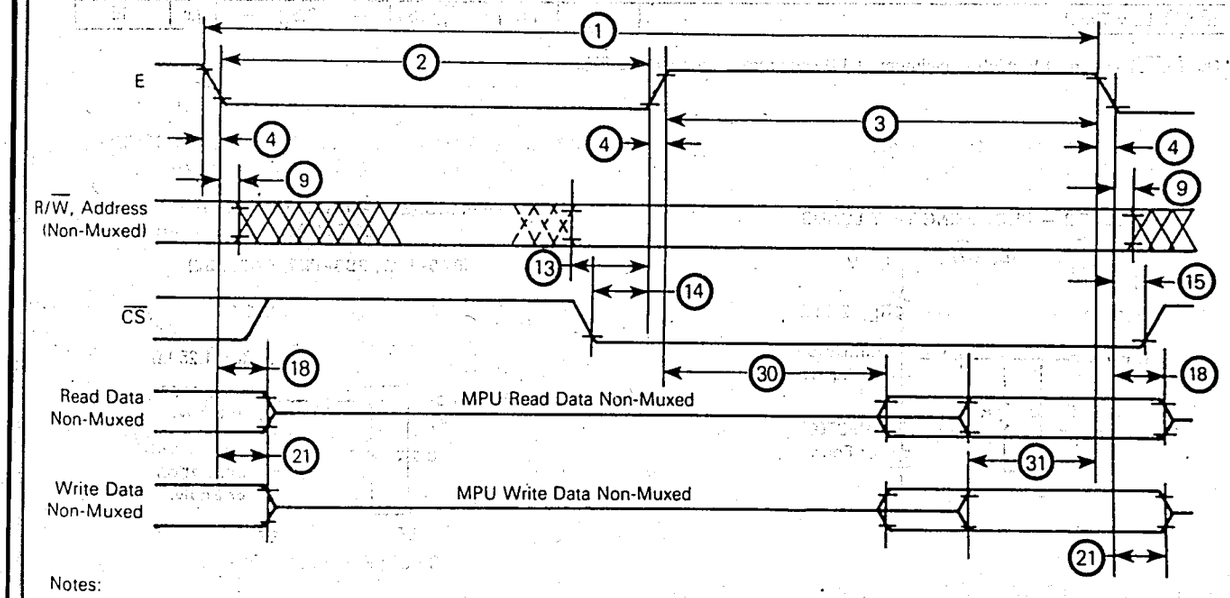
Characteristic	Symbol	Min	Typ	Max	Unit
BUS CONTROL INPUTS (R/$\overline{W}$, Enable, $\overline{RESET}$, RS0, RS1, CS0, CS1, CS2)					
Input High Voltage	V_{IH}	$V_{SS}+2.0$	–	V_{CC}	V
Input Low Voltage	V_{IL}	$V_{SS}-0.3$	–	$V_{SS}+0.8$	V
Input Leakage Current (V_{in} = 0 to 5.25 V)	I_{in}	–	1.0	2.5	µA
Capacitance (V_{in} = 0, T_A = 25°C, f = 1.0 MHz)	C_{in}	–	–	7.5	pF
INTERRUPT OUTPUTS ($\overline{IRQA}$, $\overline{IRQB}$)					
Output Low Voltage (I_{Load} = 3.2 mA)	V_{OL}	–	–	$V_{SS}+0.4$	V
Three-State Output Leakage Current	I_{OZ}	–	1.0	10	µA
Capacitance (V_{in} = 0, T_A = 25°C, f = 1.0 MHz)	C_{out}	–	–	5.0	pF
DATA BUS (D0-D7)					
Input High Voltage	V_{IH}	$V_{SS}+2.0$	–	V_{CC}	V
Input Low Voltage	V_{IL}	$V_{SS}-0.3$	–	$V_{SS}+0.8$	V
Three-State Input Leakage Current (V_{in} = 0.4 to 2.4 V)	I_{IZ}	–	2.0	10	µA
Output High Voltage (I_{Load} = – 205 µA)	V_{OH}	$V_{SS}+2.4$	–	–	V
Output Low Voltage (I_{Load} = 1.6 mA)	V_{OL}	–	–	$V_{SS}+0.4$	V
Capacitance (V_{in} = 0, T_A = 25°C, f = 1.0 MHz)	C_{in}	–	–	12.5	pF
PERIPHERAL BUS (PA0-PA7, PB0-PB7, CA1, CA2, CB1, CB2)					
Input Leakage Current (V_{in} = 0 to 5.25 V) R/$\overline{W}$, $\overline{RESET}$, RS0, RS1, CS0, CS1, $\overline{CS2}$, CA1, CB1, Enable	I_{in}	–	1.0	2.5	µA
Three-State Input Leakage Current (V_{in} = 0.4 to 2.4 V) PB0-PB7, CB2	I_{IZ}	–	2.0	10	µA
Input High Current (V_{IH} = 2.4 V) PA0-PA7, CA2	I_{IH}	– 200	– 400	–	µA
Darlington Drive Current (V_O = 1.5 V) PB0-PB7, CB2	I_{OH}	– 1.0	–	– 10	mA
Input Low Current (V_{IL} = 0.4 V) PA0-PA7, CA2	I_{IL}	–	– 1.3	– 2.4	mA
Output High Voltage (I_{Load} = – 200 µA) PA0-PA7, PB0-PB7, CA2, CB2	V_{OH}	$V_{SS}+2.4$	–	–	V
(I_{Load} = – 10 µA) PA0-PA7, CA2		$V_{CC}-1.0$	–	–	
Output Low Voltage (I_{Load} = 3.2 mA)	V_{OL}	–	–	$V_{SS}+0.4$	V
Capacitance (V_{in} = 0, T_A = 25°C, f = 1.0 MHz)	C_{in}	–	–	10	pF
POWER REQUIREMENTS					
Internal Power Dissipation (Measured at $T_A = T_L$)	P_{INT}	–	–	550	mW

BUS TIMING CHARACTERISTICS (See Notes 1 and 2)

Ident. Number	Characteristic	Symbol	MC6821 Min	MC6821 Max	MC68A21 Min	MC68A21 Max	MC68B21 Min	MC68B21 Max	Unit
1	Cycle Time	t_{cyc}	1.0	10	0.67	10	0.5	10	µs
2	Pulse Width, E Low	PW_{EL}	430	–	280	–	210	–	ns
3	Pulse Width, E High	PW_{EH}	450	–	280	–	220	–	ns
4	Clock Rise and Fall Time	t_r, t_f	–	25	–	25	–	20	ns
9	Address Hold Time	t_{AH}	10	–	10	–	10	–	ns
13	Address Setup Time Before E	t_{AS}	80	–	60	–	40	–	ns
14	Chip Select Setup Time Before E	t_{CS}	80	–	60	–	40	–	ns
15	Chip Select Hold Time	t_{CH}	10	–	10	–	10	–	ns
18	Read Data Hold Time	t_{DHR}	20	50*	20	50*	20	50*	ns
21	Write Data Hold Time	t_{DHW}	10	–	10	–	10	–	ns
30	Output Data Delay Time	t_{DDR}	–	290	–	180	–	150	ns
31	Input Data Setup Time	t_{DSW}	165	–	80	–	60	–	ns

*The data bus output buffers are no longer sourcing or sinking current by t_{DHR}max (High Impedance).

FIGURE 1 — BUS TIMING



Notes:

1. Voltage levels shown are $V_L \leq 0.4$ V, $V_H \geq 2.4$ V, unless otherwise specified.
2. Measurement points shown are 0.8 V and 2.0 V, unless otherwise specified.

PERIPHERAL TIMING CHARACTERISTICS (V_{CC}=5.0 V ±5%, V_{SS}=0 V, T_A=T_L to T_H unless otherwise specified)

Characteristic	Symbol	MC6821 Min	MC6821 Max	MC68A21 Min	MC68A21 Max	MC68B21 Min	MC68B21 Max	Unit	Reference Fig. No.
Data Setup Time	t_{PDS}	200	–	135	–	100	–	ns	6
Data Hold Time	t_{PDH}	0	–	0	–	0	–	ns	6
Delay Time, Enable Negative Transition to CA2 Negative Transition	t_{CA2}	–	1.0	–	0.670	–	0.500	µs	3, 7, 8
Delay Time, Enable Negative Transition to CA2 Positive Transition	T_{RS1}	–	1.0	–	0.670	–	0.500	µs	3, 7
Rise and Fall Times for CA1 and CA2 Input Signals	t_r, t_f	–	1.0	–	1.0	–	1.0	µs	8
Delay Time from CA1 Active Transition to CA2 Positive Transition	t_{RS2}	–	2.0	–	1.35	–	1.0	µs	3, 8
Delay Time, Enable Negative Transition to Data Valid	t_{PDW}	–	1.0	–	0.670	–	0.5	µs	3, 9, 10
Delay Time, Enable Negative Transition to CMOS Data Valid PA0-PA7, CA2	t_{CMOS}	–	2.0	–	1.35	–	1.0	µs	4, 9
Delay Time, Enable Positive Transition to CB2 Negative Transition	t_{CB2}	–	1.0	–	0.670	–	0.5	µs	3, 11, 12
Delay Time, Data Valid to CB2 Negative Transition	t_{DC}	20	–	20	–	20	–	ns	3, 10
Delay Time, Enable Positive Transition to CB2 Positive Transition	t_{RS1}	–	1.0	–	0.670	–	0.5	µs	3, 11
Control Output Pulse Width, CA2/CB2	PW_{CT}	500	–	375	–	250	–	ns	3, 11
Rise and Fall Time for CB1 and CB2 Input Signals	t_r, t_f	–	1.0	–	1.0	–	1.0	µ	12
Delay Time, CB1 Active Transition to CB2 Positive Transition	t_{RS2}	–	2.0	–	1.35	–	1.0	µs	3, 12
Interrupt Release Time, $\overline{IRQA}$ and $\overline{IRQB}$	t_{IR}	–	1.60	–	1.10	–	0.85	µs	5, 14
Interrupt Response Time	t_{RS3}	–	1.0	–	1.0	–	1.0	µs	5, 13
Interrupt Input Pulse Time	PW_I	500	–	500	–	500	–	ns	13
$\overline{RESET}$ Low Time*	t_{RL}	1.0	–	0.66	–	0.5	–	µs	15

*The $\overline{RESET}$ line must be high a minimum of 1.0 µs before addressing the PIA.

FIGURE 2 — BUS TIMING TEST LOADS

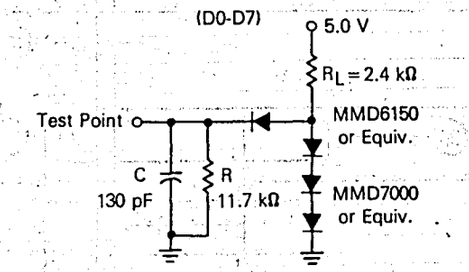
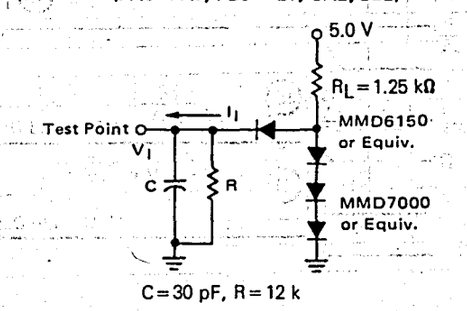


FIGURE 3 — TTL EQUIVALENT TEST LOAD

(PA0–PA7, PB0–PB7, CA2, CB2)



C=30 pF, R=12 k

FIGURE 4 — CMOS EQUIVALENT TEST LOAD

(PA0-PA7, CA2)

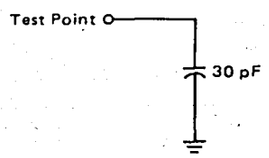


FIGURE 5 — NMOS EQUIVALENT TEST LOAD

($\overline{IRQ}$ Only)

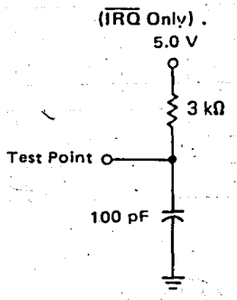


FIGURE 6 — PERIPHERAL DATA SETUP AND HOLD TIMES
(Read Mode)

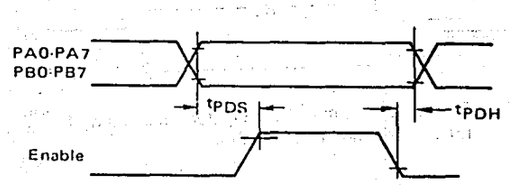
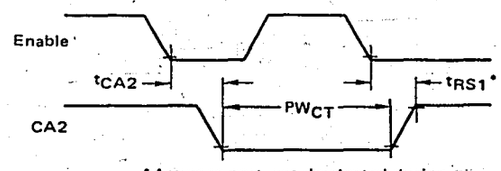


FIGURE 7 — CA2 DELAY TIME
(Read Mode; CRA-5=CRA3=1, CRA-4=0)



*Assumes part was deselected during the previous E pulse.

FIGURE 8 — CA2 DELAY TIME
(Read Mode; CRA-5=1, CRA-3=CRA-4=0)

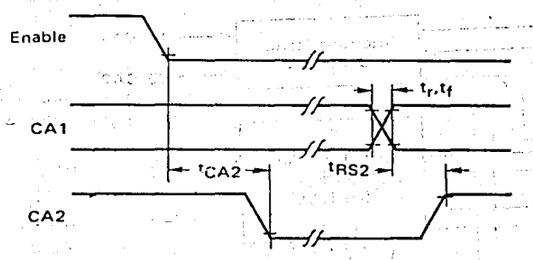


FIGURE 9 — PERIPHERAL CMOS DATA DELAY TIMES
(Write Mode; CRA-5=CRA-3=1, CRA-4=0)

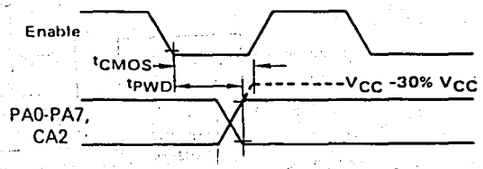


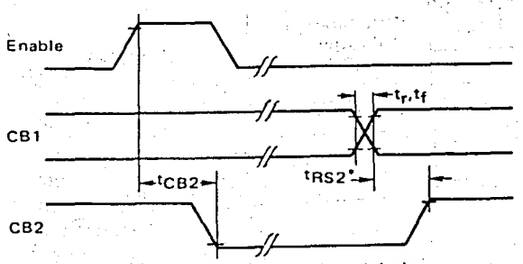
FIGURE 10 — PERIPHERAL DATA AND CB2 DELAY TIMES
(Write Mode; CRB-5=CRB-3=1, CRB-4=0)

*CB2 goes low as a result of the positive transition of Enable.

FIGURE 11 — CB2 DELAY TIME
(Write Mode; CRB-5=CRB-3=1, CRB-4=0)

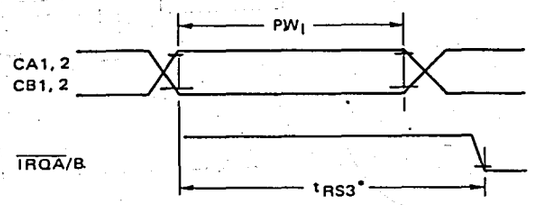
*Assumes part was deselected during the previous E pulse.

FIGURE 12 — CB2 DELAY TIME
(Write Mode; CRB-5=1, CRB-3=CRB-4=0)



*Assumes part was deselected during any previous E pulse.

FIGURE 13 — INTERRUPT PULSE WIDTH AND $\overline{IRQ}$ RESPONSE



*Assumes Interrupt Enable Bits are set.

Note: Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted.

FIGURE 14 — $\overline{IRQ}$ RELEASE TIME

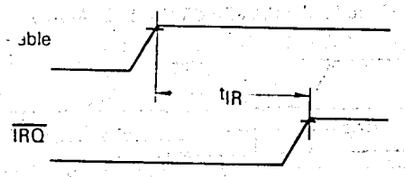
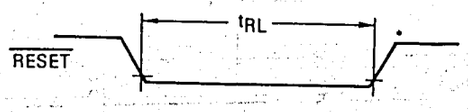


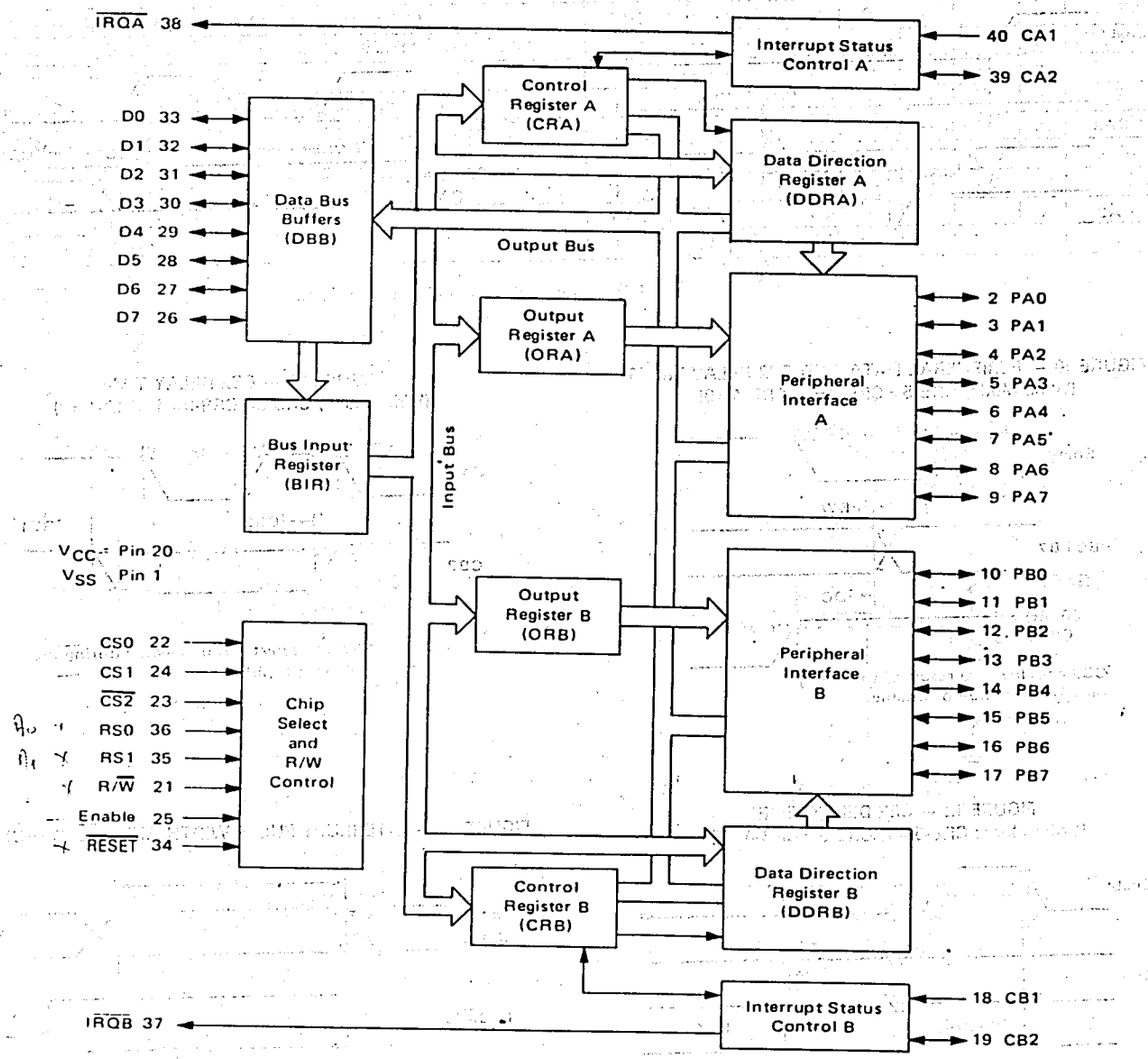
FIGURE 15 — $\overline{RESET}$ LOW TIME



*The $\overline{RESET}$ line must be a V_{IH} for a minimum of 1.0 μs before addressing the PIA.

Note: Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted.

FIGURE 16 — EXPANDED BLOCK DIAGRAM



PIA INTERFACE SIGNALS FOR MPU

The PIA interfaces to the M6800 bus with an 8-bit bidirectional data bus, three chip select lines, two register select lines, two interrupt request lines, a read/write line, an enable line and a reset line. To ensure proper operation with the MC6800, MC6802, or MC6808 microprocessors, VMA should be used as an active part of the address decoding.

Bidirectional Data (D0-D7) — The bidirectional data lines (D0-D7) allow the transfer of data between the MPU and the PIA. The data bus output drivers are three-state devices that remain in the high-impedance (off) state except when the MPU performs a PIA read operation. The read/write line is in the read (high) state when the PIA is selected for a read operation.

Enable (E) — The enable pulse, E, is the only timing signal that is supplied to the PIA. Timing of all other signals is referenced to the leading and trailing edges of the E pulse.

Read/Write (R/$\overline{W}$) — This signal is generated by the MPU to control the direction of data transfers on the data bus. A low state on the PIA read/write line enables the input buffers and data is transferred from the MPU to the PIA on the E signal if the device has been selected. A high on the read/write line sets up the PIA for a transfer of data to the bus. The PIA output buffers are enabled when the proper address and the enable pulse E are present.

$\overline{RESET}$ — The active low $\overline{RESET}$ line is used to reset all register bits in the PIA to a logical zero (low). This line can be used as a power-on reset and as a master reset during system operation.

Chip Selects (CS0, CS1, and $\overline{CS2}$) — These three input signals are used to select the PIA. CS0 and CS1 must be high and $\overline{CS2}$ must be low for selection of the device. Data transfers are then performed under the control of the enable and read/write signals. The chip select lines must be stable for the duration of the E pulse. The device is deselected when any of the chip selects are in the inactive state.

Register Selects (RS0 and RS1) — The two register select lines are used to select the various registers inside the PIA. These two lines are used in conjunction with internal Control Registers to select a particular register that is to be written or read.

The register and chip select lines should be stable for the duration of the E pulse while in the read or write cycle.

Interrupt Request ($\overline{IRQA}$ and $\overline{IRQB}$) — The active low Interrupt Request lines ($\overline{IRQA}$ and $\overline{IRQB}$) act to interrupt the MPU either directly or through interrupt priority circuitry. These lines are "open drain" (no load device on the chip). This permits all interrupt request lines to be tied together in a wire-OR configuration.

Each Interrupt Request line has two internal interrupt flag bits that can cause the Interrupt Request line to go low. Each flag bit is associated with a particular peripheral interrupt line. Also, four interrupt enable bits are provided in the PIA which may be used to inhibit a particular interrupt from a peripheral device.

Servicing an interrupt by the MPU may be accomplished by a software routine that, on a prioritized basis, sequentially reads and tests the two control registers in each PIA for interrupt flag bits that are set.

The interrupt flags are cleared (zeroed) as a result of an MPU Read Peripheral Data Operation of the corresponding data register. After being cleared, the interrupt flag bit cannot be enabled to be set until the PIA is deselected during an E pulse. The E pulse is used to condition the interrupt control lines (CA1, CA2, CB1, CB2). When these lines are used as interrupt inputs, at least one E pulse must occur from the inactive edge to the active edge of the interrupt input signal to condition the edge sense network. If the interrupt flag has been enabled and the edge sense circuit has been properly conditioned, the interrupt flag will be set on the next active transition of the interrupt input pin.

PIA PERIPHERAL INTERFACE LINES

The PIA provides two 8-bit bidirectional data buses and four interrupt/control lines for interfacing to peripheral devices.

Section A Peripheral Data (PA0-PA7) — Each of the peripheral data lines can be programmed to act as an input or output. This is accomplished by setting a "1" in the corresponding Data Direction Register bit for those lines which are to be outputs. A "0" in a bit of the Data Direction Register causes the corresponding peripheral data line to act as an input. During an MPU Read Peripheral Data Operation, the data on peripheral lines programmed to act as inputs appears directly on the corresponding MPU Data Bus lines. In the input mode, the internal pullup resistor on these lines represents a maximum of 1.5 standard TTL loads.

The data in Output Register A will appear on the data lines that are programmed to be outputs. A logical "1" written into the register will cause a "high" on the corresponding data line while a "0" results in a "low." Data in Output Register A may be read by an MPU "Read Peripheral Data A" operation when the corresponding lines are programmed as outputs. This data will be read property if the voltage on the peripheral data lines is greater than 2.0 volts for a logic "1" output and less than 0.8 volt for a logic "0" output. Loading the output lines such that the voltage on these lines does not reach full voltage causes the data transferred into the MPU on a Read operation to differ from that contained in the respective bit of Output Register A.

Section B Peripheral Data (PB0-PB7) — The peripheral data lines in the B Section of the PIA can be programmed to act as either inputs or outputs in a similar manner to PA0-PA7. They have three-state capabiity, allowing them to enter a high-impedance state when the peripheral data line is used as an input. In addition, data on the peripheral data lines

PB0-PB7 will be read properly from those lines programmed as outputs even if the voltages are below 2.0 volts for a "high" or above 0.8 V for a "low". As outputs, these lines are compatible with standard TTL and may also be used as a source of up to 1 milliampere at 1.5 volts to directly drive the base of a transistor switch.

Interrupt Input (CA1 and CB1) — Peripheral input lines CA1 and CB1 are input only lines that set the interrupt flags of the control registers. The active transition for these signals is also programmed by the two control registers.

Peripheral Control (CA2) — The peripheral control line CA2 can be programmed to act as an interrupt input or as a peripheral control output. As an output, this line is compa ble with standard TTL; as an input the internal pullup resist on this line represents 1.5 standard TTL loads. The funcu of this signal line is programmed with Control Register A

Peripheral Control (CB2) — Peripheral Control line C may also be programmed to act as an interrupt input peripheral control output. As an input, this line has high put impedance and is compatible with standard TTL. As output it is compatible with standard TTL and may also used as a source of up to 1 milliampere at 1.5 volts to dire drive the base of a transistor switch. This line is program by Control Register B.

INTERNAL CONTROLS

INITIALIZATION

A $\overline{\text{RESET}}$ has the effect of zeroing all PIA registers. This will set PA0-PA7, PB0-PB7, CA2 and CB2 as inputs, and all interrupts disabled. The PIA must be configured during the restart program which follows the reset.

There are six locations within the PIA accessible to the MPU data bus: two Peripheral Registers, two Data Direction Registers, and two Control Registers. Selection of these locations is controlled by the RS0 and RS1 inputs together with bit 2 in the Control Register, as shown in Table 1.

Details of possible configurations of the Data Direction and Control Register are as follows:

TABLE 1 — INTERNAL ADDRESSING

		Control Register Bit		
RS1	RS0	CRA-2	CRB-2	Location Selected
0	0	1	X	Peripheral Register A
0	0	0	X	Data Direction Register A
0	1	X	X	Control Register A
1	0	X	1	Peripheral Register B
1	0	X	0	Data Direction Register B
1	1	X	X	Control Register B

X = Don't Care

PORT A-B HARDWARE CHARACTERISTICS

As shown in Figure 17, the MC6821 has a pair of I/O ports whose characteristics differ greatly. The A side is designed to drive CMOS logic to normal 30% to 70% levels, and incorporates an internal pullup device that remains connected even in the input mode. Because of this, the A side requires more drive current in the input mode than Port B. In contrast, the B side uses a normal three-state NMOS buffer which cannot pullup to CMOS levels without external resistors. The B side can drive extra loads such as Darlingtons without problem. When the PIA comes out of reset, the A port represents inputs with pullup resistors, whereas the B side (input mode also) will float high or low, depending upon the load connected to it.

Notice the differences between a Port A and Port B re operation when in the output mode. When reading Port the actual pin is read, whereas the B side read comes from output latch, ahead of the actual pin.

CONTROL REGISTERS (CRA and CRB)

The two Control Registers (CRA and CRB) allow the M to control the operation of the four peripheral control lin CA1, CA2, CB1, and CB2. In addition they allow the MPU enable the interrupt lines and monitor the status of the int rupt flags. Bits 0 through 5 of the two registers may be w ten or read by the MPU when the proper chip select a register select signals are applied. Bits 6 and 7 of the t registers are read only and are modified by external interru occurring on control lines CA1, CA2, CB1, or CB2. The f mat of the control words is shown in Figure 18.

DATA DIRECTION ACCESS CONTROL BIT (CRA-2 a CRB-2)

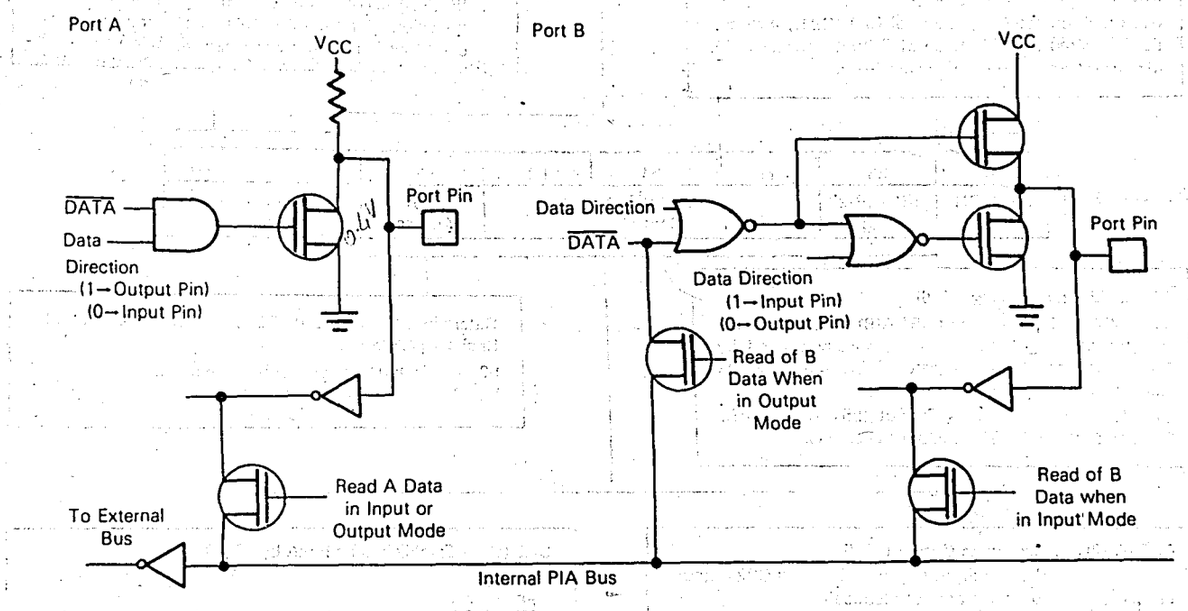
Bit 2, in each Control Register (CRA and CRB), det mines selection of either a Peripheral Output Register or t corresponding Data Direction E Register when the pro register select signals are applied to RS0 and RS1. A "1" bit 2 allows access of the Peripheral Interface Register, wh a "0" causes the Data Direction Register to be addressed

Interrupt Flags (CRA-6, CRA-7, CRB-6, and CRB-7) The four interrupt flag bits are set by active transitions signals on the four Interrupt and Peripheral Control lin when those lines are programmed to be inputs. These b cannot be set directly from the MPU Data Bus and are res indirectly by a Read Peripheral Data Operation on the a propriate section.

Control of CA2 and CB2 Peripheral Control Lines (CRA- CRA-4, CRA-5, CRB-3, CRB-4, and CRB-5) — Bits 3, 4, ar 5 of the two control registers are used to control the CA2 ar CB2 Peripheral Control lines. These bits determine if the co trol lines will be an interrupt input or an output contr signal. If bit CRA-5 (CRB-5) is low, CA2 (CB2) is an interru input line similar to CA1 (CB1). When CRA-5 (CRB-5) high, CA2 (CB2) becomes an output signal that may be use to control peripheral data transfers. When in the outp mode, CA2 and CB2 have slightly different loadin characteristics.

Control of CA1 and CB1 Interrupt Input Lines (CRA-0, CRB-1, CRA-1, and CRB-1) — The two lowest-order bits of the control registers are used to control the interrupt input lines CA1 and CB1. Bits CRA-0 and CRB-0 are used to enable the MPU interrupt signals $\overline{\text{IRQA}}$ and $\overline{\text{IRQB}}$, respectively. Bits CRA-1 and CRB-1 determine the active transition of the interrupt input signals CA1 and CB1.

FIGURE 17 — PORT A AND PORT B EQUIVALENT CIRCUITS



ORDERING INFORMATION

MC68A21CP

- Motorola Integrated Circuit
- M6800 Family
- Blanks = 1.0 MHz
- A = 1.5 MHz
- B = 2.0 MHz
- Device Designation
- In M6800 Family
- Temperature Range
- Blank = 0° — +70°C
- C = −40° — +85°C
- Package
- P = Plastic
- S = Cerdip
- L = Ceramic

BETTER PROGRAM

Better program processing is available on all types listed. Add suffix letters to part number.

Level 1 add "S" Level 2 add "D" Level 3 add "DS"

Level 1 "S" = 10 Temp Cycles — (−25 to 150°C); Hi Temp testing at T_A max.
Level 2 "D" = 168 Hour Burn-in at 125°C
Level 3 "DS" = Combination of Level 1 and 2.

Speed	Device	Temperature Range
1.0 MHz	MC6821P,L,S	0 to 70°C
	MC6821CP,CL,CS	−40 to +85°C
1.5 MHz	MC68A21P,L,S	0 to +70°C
	MC68A21CP,CL,CS	−40 to +85°C
2.0 MHz	MC68B21P,L,S	0 to +70°C

FIGURE 18 — CONTROL WORD FORMAT

Determine Active CA1 (CB1) Transition for Setting Interrupt Flag IRQA(B)1 — (bit 7)

b1 = 0: IRQA(B)1 set by high-to-low transition on CA1 (CB1)

b1 = 1: IRQA(B)1 set by low-to-high transition on CA1 (CB1).

CA1 (CB1) Interrupt Request Enable/Disable

b0 = 0: Disables IRQA(B) MPU Interrupt by CA1 (CB1) active transition.[1]

b0 = 1: Enable IRQA(B) MPU Interrupt by CA1 (CB1) active transition.

1. IRQA(B) will occur on next (MPU generated) positive transition of b0 if CA1 (CB1) active transition occurred while interrupt was disabled.

IRQA(B) 1 Interrupt Flag (bit 7)

Goes high on active transition of CA1 (CB1); Automatically cleared by MPU Read of Output Register A(B). May also be cleared by hardware Reset.

Control Register

b7	b6	b5	b4	b3	b2	b1	b0
IRQA(B)1 Flag	IRQA(B)2 Flag	CA2 (CB2) Control			DDR Access	CA1 (CB1) Control	

IRQA(B)2 Interrupt Flag (bit 6)

When CA2 (CB2) is an input, IRQA(B) goes high on active transition CA2 (CB2); Automatically cleared by MPU Read of Output Register A(B). May also be cleared by hardware Reset.

CA2 (CB2) Established as Output (b5 = 1): IRQA(B) 2 = 0, not affected by CA2 (CB2) transitions.

Determines Whether Data Direction Register Or Output Register is Addressed

b2 = 0: Data Direction Register selected.

b2 = 1: Output Register selected.

CA2 (CB2) Established as Output by b5 = 1

(Note that operation of CA2 and CB2 output functions are not identical)

b5 b4 b3

1 0 → CA2

b3 = 0: **Read Strobe with CA1 Restore**
CA2 goes low on first high-to-low E transition following an MPU read of Output Register A; returned high by next active CA1 transition, as specified by bit 1.

b3 = 1: **Read Strobe with E Restore**
CA2 goes low on first high-to-low E transition following an MPU read of Output Register A; returned high by next high-to-low E transition during a deselect.

→ CB2

b3 = 0: **Write Strobe with CB1 Restore**
CB2 goes low on first low-to-high E transition following an MPU write into Output Register B; returned high by the next active CB1 transition as specified by bit 1. CRB-b7 must first be cleared by a read of data.

b3 = 1: **Write Strobe with E Restore**
CB2 goes low on first low-to-high E transition following an MPU write into Output Register B; returned high by the next low-to-high E transition following an E pulse which occurred while the part was deselected.

b5 b4 b3

1 1 → **Set/Reset CA2 (CB2)**

CA2 (CB2) goes low as MPU writes b3 = 0 into Control Register.

CA2 (CB2) goes high as MPU writes b3 = 1 into Control Register.

CA2 (CB2) Established as Input by b5 = 0

b5 b4 b3

0 → **CA2 (CB2) Interrupt Request Enable/Disable**

b3 = 0: Disables IRQA(B) MPU Interrupt by CA2 (CB2) active transition.*

b3 = 1: Enables IRQA(B) MPU Interrupt by CA2 (CB2) active transition.

*IRQA(B) will occur on next (MPU generated) positive transition of b3 if CA2 (CB2) active transition occurred while interrupt was disabled.

→ **Determines Active CA2 (CB2) Transition for Setting Interrupt Flag IRQA(B)2 — (Bit b6)**

b4 = 0: IRQA(B)2 set by high-to-low transition on CA2 (CB2).

b4 = 1: IRQA(B)2 set by low-to-high transition on CA2 (CB2).