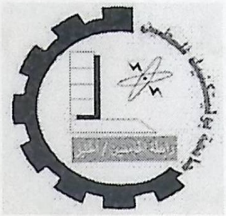
Palestine Polytechnic University



College of Engineering & Technology
Computer & Electrical Engineering Department

Graduate Project

General Purpose Control Board Based on 80C535 Microcontroller

Project Team

Ali Abu-Sabha **Khalaf Idais**

Osama Jabari

Project Supervisor
Eng. Mazen Zalloum

Hebron - Palestine

June 2005

Abstract

This project is a general purpose control board (GPCB), based on microcontroller. It depends on a closed loop mechanism, and it has several switches as interfaces to external applications. The control depends on software written on associated program memory, so this board is extensible and flexible to accommodate many applications. It can control many different applications at the same time automatically.

المشروع هو عبارة عن لوحة تحكم متعددة الأغراض مبنية على الميكروكنترولر، تستخدم هذه اللوحة نظام التحكم المغلق، و تحتوي على عدة مفاتيح مصممة لتوصيلها مع التطبيقات الخارجية المراد التحكم بها تلقائيا. عملية التحكم تتم بالاعتماد على برنامج معين تتم كتابته على ذاكرة القراءة المتصلة بهذه اللوحة، لذلك فإنها تناسب عدة تطبيقات مختلفة، كما أنه يمكن زيادة عدد التطبيقات المراد التحكم بها.

Table of contents

List of Figures

Chapter Five

Chapter Six

List of Tables

LIST OF ABBREVIATIONS

AC	Alternative Current
ACC	Accumulator
ADCON	A/D converter Control Register
ADDAT	A/D Converter Data Register
ALE	Address Latch Enable
AN0-AN7	Analog Input 0-7
Av	Voltage Gain
B	Base Register
BCD	Binary Coded Decimal
BSY	Busy Flag
CE	Chip Enable
CI	Input Capacitance
CPU	Central processing Unit
DAPR	D/A Converter Program Register
DC	Direct Current
EA	External Access
EPROM	Erasable programmable ROM
GND	Ground
GPR	general purpose register
H	Hex Decimal
I/O	Input Output
LED	Light Emitting Diode
OE	Output Enable
OP-Amp	Operational Amplifier
P0,1,…,6	Port 0,1,…,6
PC	Personal Computer
PCON	Power Control Register

R0-R7	Register 0-7
PSW	Program Status Word
RAM	Read Access Memory
SFR	Special Function Register
tC	Conversion Time
tL	Load Time
tS	Sample Time
Vcc	Supply Voltage
Vss	Ground

Chapter One

Introduction

1.1 General idea about the project

1.2 Problem statement

1.3 Importance

1.4 Literature Review

1.5 Estimated cost

1.6 Time planning

1.7 Report contents

Chapter One

Introduction

1.1 General idea about the project

This project is about designing and implementing general purpose control board based on microcontroller, this board can control several different applications automatically. The control is a closed loop and depends on software which can be updated; therefore, the board will be flexible to accommodate more and more applications.

1.2 Problem statement

In recent years, embedded systems are becoming increasingly more important due to their wide spread utilization in every aspect of our lives. Embedded systems are affecting our lives in many ways. Embedded systems are becoming smaller and are able to perform more functions at the same time. The utilization of microcontroller, called embedded microcontrollers in this setting, has greatly contributed to the digital data processing performance of many embedded systems.

Given the importance of embedded systems, we have developed a general purpose control board (GPCB) based on 80C535 microcontroller with many I/O designed switches for connecting to external applications, it uses a closed loop system to control several applications at the same time automatically. The control is software dependent, so it is flexible and suitable to many different applications.

1.3 Importance

- It is a microcontroller based system, the microcontroller is a computer on chip, with CPU, memory, I/O port, timer counter and other component fabricated on one chip.
- Software control which can be updated.
- It is a closed loop control system.
- Study and implement interfaces of microcontroller with real life.
- Can control many different applications.
- The first project of general purpose board which based on microcontroller.

1.4 Literature Review

After searching in our university library we didn't find projects based on our microcontroller, but we find the following PC based control projects:

1. **Switch Control from PC Parallel Port:**
   This project has 4 switches controller. It can only used with a PC's parallel output port and a special program running the parallel port control program.

2. **Use of PC Printer Port for Control and data acquisition**:
   This project discusses how to use and program the printer port.

3. **Ten-Relay Board:**
   There are 10 relays DPCO (Dual Pole Change Over).

4. **General Purpose Printed Board Based on PC:**
   Control 8 relays work as switches to control different machines.

In the internet search we find single control applications based mostly on PIC microcontroller, but we didn't find any application based on our microcontroller.

1.4 Estimated cost

The cost for the project is estimated as follow:

Equipment	*Cost (NIS)*
Microcontroller	200
EPROM	20
RAM	40
Board	50
Printing	300
Others	200
Total cost	**810**

Table 1-1: Estimated costs

1.5 Time planning

This project lasts 32 Weeks divided into nine tasks, explained in ***table 1.2***.

Weeks

Project phases	1		3		5		7		9		11		13		15		17		19		21		23		25		27		29		31	
Literature review																																
Requirement analyses																																
Study the microcontroller																																
Study some applications																																
System design																																
System implementation																																
System testing																																
documentation																																

Table 1-2: Time planning

1.6 Report contents

This project divided into seven chapters, these chapters are as follows:

Chapter 1: "Introduction", it describes the general idea about the project, explains the importance of the project, and the cost of the project.

Chapter 2: "Theoretical Background", it includes theoretical information about the project and its components.

Chapter 3: "Design concept", it describes the purpose and objectives of project, general block diagram of the project, and explains the way of the project work.

Chapter 4: "Hardware System Design", describes the schematic diagram of the project and describes all components involved in the project.

Chapter 5: "Software System Design", it includes the general algorithms, and flowcharts of the programs that we write it to control the applications.

Chapter 6:"Implemintation and Testing", it describes the implementation and testing procedures and stages of the project.

Chapter 7:" Conclusions and Future work ", it shows the problems, risks, conclusions, and hints for future work.

Chapter Two

Theoretical Background

2.1 Preface

2.2 Microcontroller

2.3 Relays

2.4 Isolation Circuit

2.5 Amplifiers

2.6 Sensors

Chapter Two

Theoretical Background

2.1 Preface

Because our project aims to design general purpose control board, the microcontroller will control switches ON and OFF, these switches contain optocouplers, and relays. And also, our project is a closed loop system; this board will accept signals from external environment. The input and output signals may be amplified using amplifiers to be recognizable.

2.2 Microcontroller

2.2.1 Introduction [4]

The SAB 80C515/80C535 is a new, powerful member of the Siemens SAB 8051 family of 8-bit microcontrollers. It is designed in Siemens ACMOS technology and it is functionally compatible with the SAB 80515/80535 devices designed in MYMOS technology.

The ACMOS and the MYMOS versions are stand-alone, high performance single-chip microcontrollers based on the SAB 8051/80C51 architecture. While maintaining all the SAB 80(C)51 operating characteristics, the SAB 80(C)515/80(C)535 incorporate several enhancements which significantly increase design flexibility and overall system performance.

The difference between ACMOS and MYMOS technology is the low power consumption properties of the ACOMS technology, and also there is a difference concerned to the microcontroller construction, this difference will be mentioned later when talking about the parallel I/O ports of the microcontroller.

The SAB 80(C)515/ 80(C)535 has the following features [1]:

- 8 Kbytes on-chip program memory for 80(C)515 only.
- 256 byte on-chip RAM
- Six 8-bit parallel I/O ports
- One analog input port.
- Full-duplex serial port.
- Three 16-bit timer/counters
- A/D converter, 8 multiplexed analog inputs, programmable reference voltages
- 256 directly addressable bits
- 12 interrupt sources (7 external,5 internal),4 priority levels
- Stack depth up to 256 byte
- 1 ms instruction cycle at 12-MHz operation
- External program and data memory expandable up to 64 Kbytes each
- Compatible with standard SAB 8080/8085 peripherals and memories

See the logical symbol of the SAB 80C515/ 80C535 in ***Figure 2-1***.

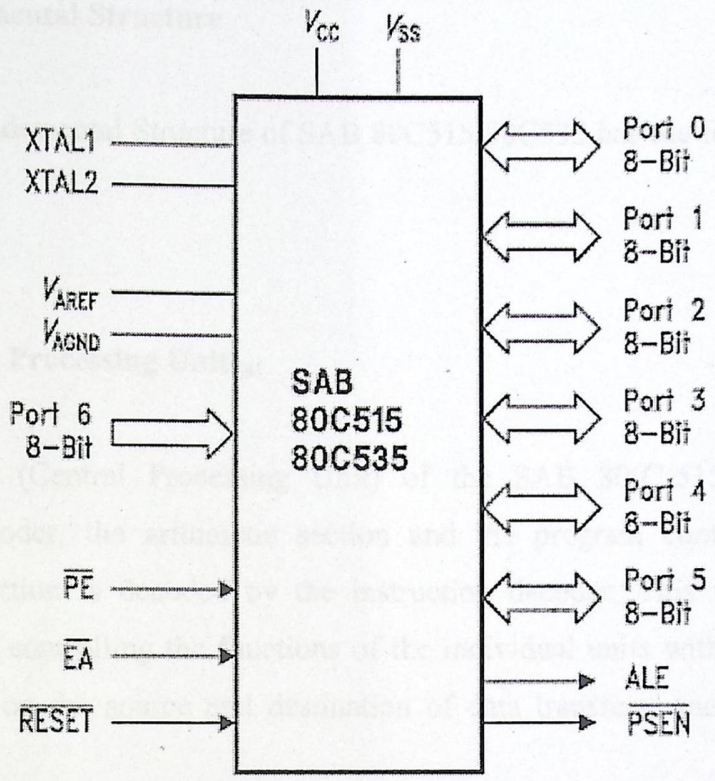


Fig 2-1: SAB 80C515/80C535 Logical Symbol [5]

2.2.2 Functional Description [1]

The members of the SAB 80515 family of microcontrollers are:

- SAB 80C515: Microcontroller, designed by Siemens ACMOS technology, with 8-Kbyte factory mask-programmable ROM
- SAB 80C535: ROM-less version, identical to the SAB 80C515
- SAB 80515: Microcontroller, designed in Siemens MYMOS technology, with 8-Kbyte factory mask-programmable ROM
- SAB 80535: ROM-less version, identical to the SAB 80515
- SAB 80515K: Special ROM-less version of the SAB 80515 with an additional interface for program memory accesses. An external ROM that is accessed via the interface substitutes the SAB 80515's internal ROM.

2.2.3 Fundamental Structure

The Fundamental Structure of SAB 80C515/80C535 has the following units, see ***figure 2-2***

2.2.3.1 Central Processing Unit [4]

The CPU (Central Processing Unit) of the SAB 80(C)515 consists of the instruction decoder, the arithmetic section and the program control section. Each program instruction is decoded by the instruction decoder. This unit generates the internal signals controlling the functions of the individual units within the CPU. They have an effect on the source and destination of data transfers, and control the ALU processing.

The arithmetic section of the processor performs extensive data manipulation and is comprised of the Arithmetic/Logic Unit (ALU), an A, B, and PSW register. The ALU accepts 8-bit data words from one or two sources and generates an 8-bit result under the control of the instruction decoder. The ALU performs the arithmetic operations add, subtract, multiply, divide, increment, decrement, BCD operations, compare, and the logic operations AND, OR, Exclusive OR, complement and rotate (right, left or swap nibble (left four)). Also it includes a Boolean processor which performs the bit operations of set, clear, complement, jump-if-not-set, jump-if-set-and-clear and move to/from carry.

The program control section controls the sequence in which the instructions stored in program memory are executed. The 16-bit program counter (PC) holds the address of the next instruction to be executed. The conditional branch logic enables internal and external events to the processor to cause a change in the program execution sequence.

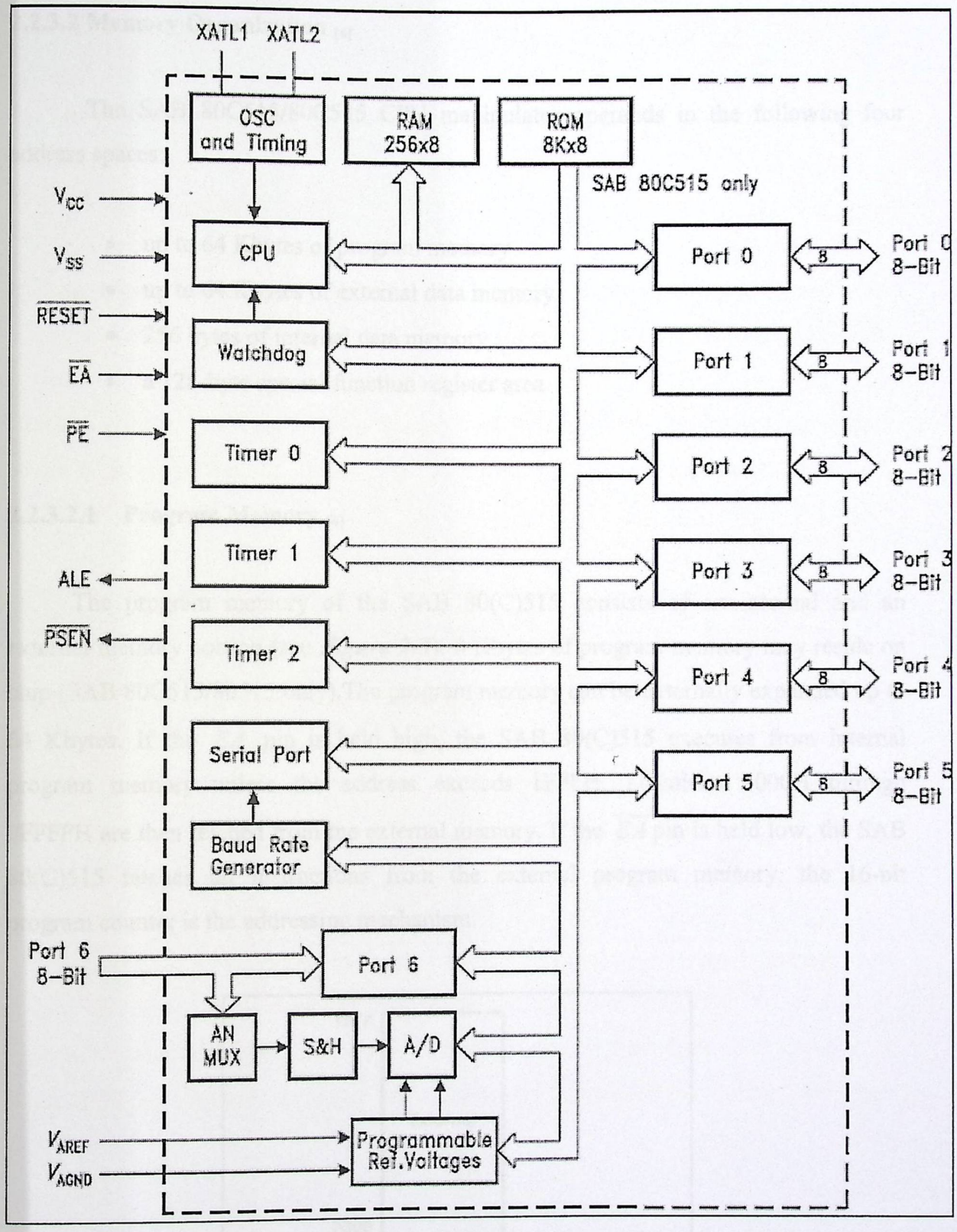


Fig 2-2: The block diagram of the SAB 80C515

2.2.3.2 Memory Organization [4]

The SAB 80C515/80C535 CPU manipulates operands in the following four address spaces:

- up to 64 Kbytes of program memory
- up to 64 Kbytes of external data memory
- 256 bytes of internal data memory
- a 128-byte special function register area

2.2.3.2.1 Program Memory [4]

The program memory of the SAB 80(C)515 consists of an internal and an external memory portion (see ***Figure 2-3***). 8 Kbytes of program memory may reside on chip (SAB 80C515/80515 only).The program memory can be externally expanded up to 64 Kbytes. If the $\overline{EA}$ pin is held high, the SAB 80(C)515 executes from internal program memory unless the address exceeds 1FFFH. Locations 2000H through 0FFFFH are then fetched from the external memory. If the $\overline{EA}$ pin is held low, the SAB 80(C)515 fetches all instructions from the external program memory, the 16-bit program counter is the addressing mechanism.

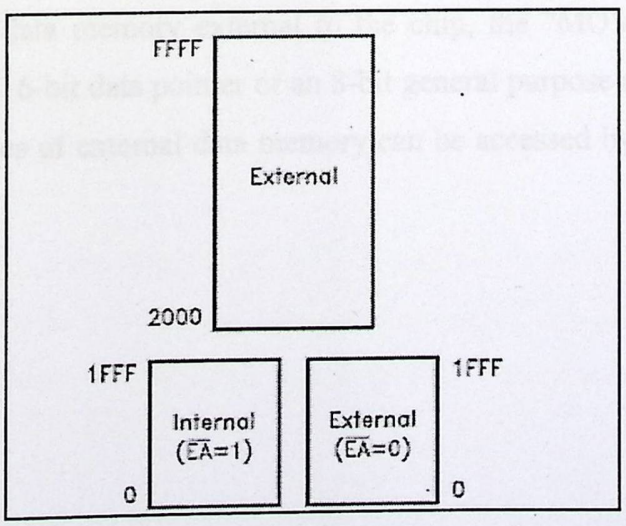


Fig 2-3: Program Memory Address Space

2.2.3.2.2 Data Memory [4]

The data memory address space consists of an internal and an external memory portion.

- **Internal Data Memory**

The internal data memory address space is divided into three physically separate and distinct blocks: the lower 128 bytes of RAM, the upper 128-byte RAM area, and the 128 -byte special function register (SFR) area (see ***Figure 2-4***). Since the latter SFR area and the upper RAM area share the same address locations, they must be accessed through different addressing modes. The following table shows the addressing modes used for the different RAM/SFR spaces.

Address Space	Locations	Addressing Mode
Lower 128 bytes of RAM	00_H to $7F_H$	direct/indirect
Upper 128 bytes of RAM	80_H to $0FF_H$	indirect
Special function registers	80_H to $0FF_H$	direct

Table 2-1: Addressing modes for different RAM/SFR spaces

- **External Data Memory**

Figure 2-4 contains memory maps which illustrate the internal/external data memory. To address data memory external to the chip, the "MOVX" instructions in combination with the 16-bit data pointer or an 8-bit general purpose register are used. A maximum of 64 Kbytes of external data memory can be accessed by instructions using 16-bit address.

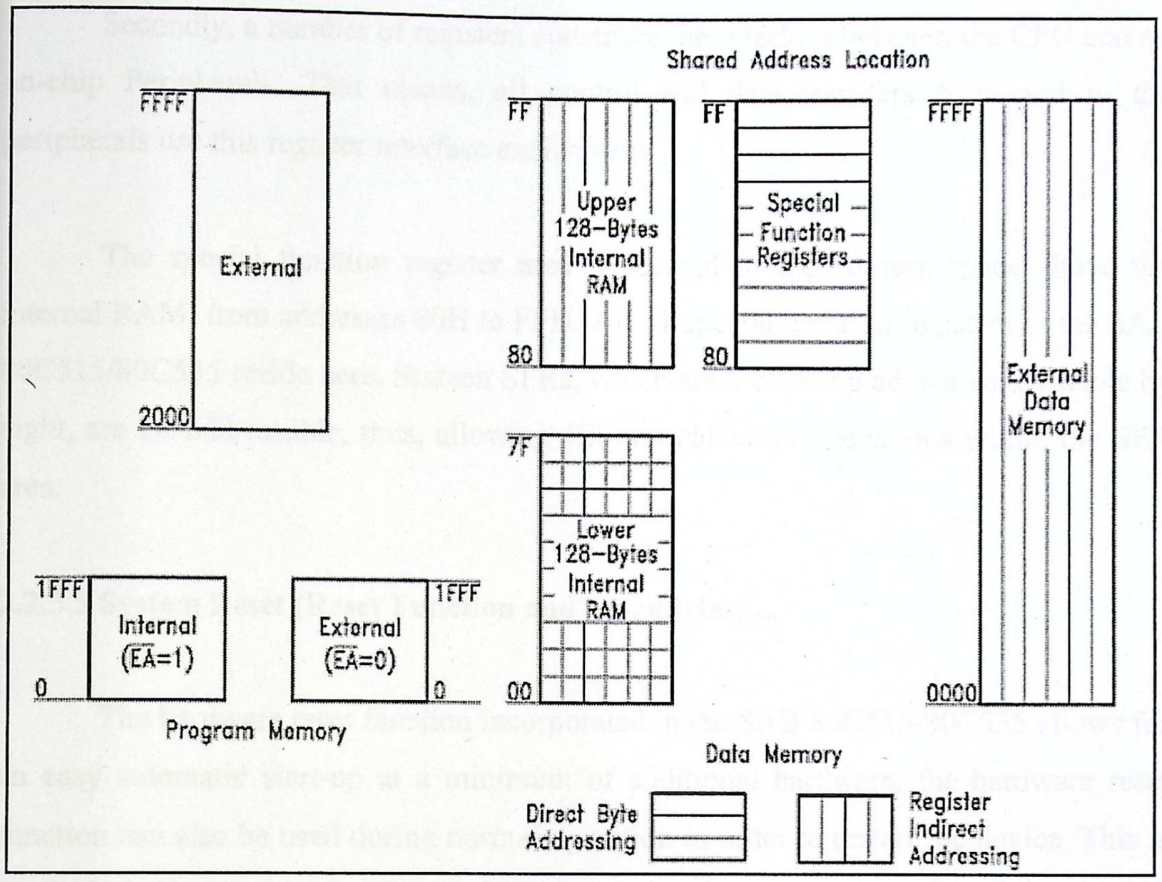


Fig 2-4: Data Memory /SFR Address Space

2.2.3.3 General Purpose Registers

The lower 32 locations of the internal RAM are assigned to four banks with eight general purpose register (GPRs) each. Only one of these banks may be enabled at a time. Two bits in the program status word, PSW.3 and PSW.4, select the active register bank. Bank 0 is the default to the SAB 80C515/80C535.

2.2.3.4 Special Function Registers

The Special Function Register (SFR) area has two important functions. Firstly, all CPU register except the program counter and the four register banks reside here. The CPU registers are the arithmetic registers like A, B, PSW and pointers like SP, DPH and DPL.

Secondly, a number of registers constitute the interface between the CPU and all on-chip Peripherals. That means, all control and data transfers from and to the peripherals use this register interface exclusively.

The special function register area is located in the address space above the internal RAM, from addresses 80H to FFH. All 41 special function registers of the SAB 80C515/80C535 reside here. Sixteen SFRs, which are located on addresses dividable by eight, are bit-addressable, thus, allowing 128 bit-addressable locations within the SFR area.

2.2.3.5 System Reset (Reset Function and Circuitries) [4]

The hardware reset function incorporated in the SAB 80C515/80C535 allows for an easy automatic start-up at a minimum of additional hardware, the hardware reset function can also be used during normal operation in order to restart the device, This is particularly done when the power-down mode is to be terminated, in addition to the hardware reset, which is applied externally to the SAB 80C515/80C535, there is also the possibility of an internal hardware reset. This internal reset will be initiated by the watchdog timer. The reset input is an active low input at pin 10 ($\overline{RESET}$).

An automatic reset can be obtained when VCC is applied by connecting the reset pin to VSS via a capacitor as shown in ***Figure 2-5*** **a),** and **c)**. The same considerations apply if the reset signal is generated externally (***Figure 2-5 b***).

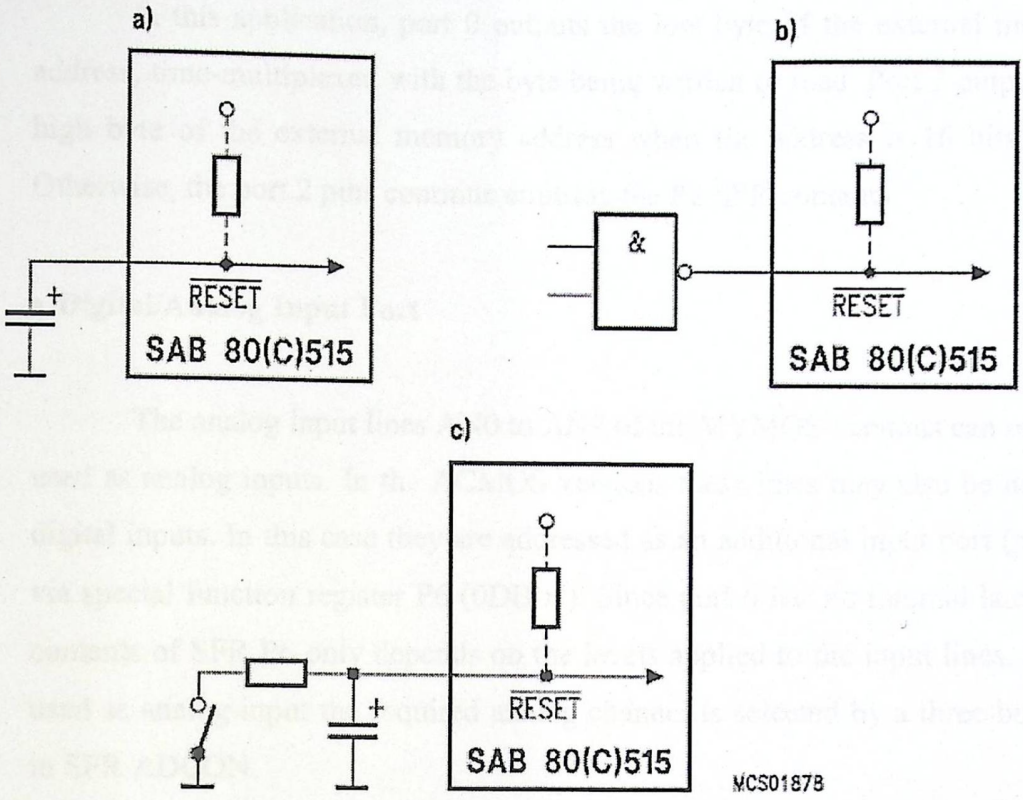


Fig 2-5: Reset Circuitries

2.2.3.6 Parallel I/O [4]

The SAB 80C515/80C535 has six 8-bit digital I/O ports and one 8-bit digital/analog input port.

- **Digital I/O ports**

The SAB 80C515/80C535 allows for digital I/O on 48 lines grouped into 6 bidirectional 8-bit ports. Each port bit consists of a latch, an output driver and an input buffer. Read and write accesses to the I/O ports P0 through P5 are performed via their corresponding special function registers P0 to P5, the output drivers of port 0 and 2 and the input buffers of port 0 are also used for accessing external memory.

In this application, port 0 outputs the low byte of the external memory address, time-multiplexed with the byte being written or read. Port 2 outputs the high byte of the external memory address when the address is 16 bits wide. Otherwise, the port 2 pins continue emitting the P2 SFR contents

- **Digital/Analog Input Port**

The analog input lines AN0 to AN7 of the MYMOS versions can only be used as analog inputs. In the ACMOS versions these lines may also be used as digital inputs. In this case they are addressed as an additional input port (port 6) via special function register P6 (0DB H). Since port 6 has no internal latch, the contents of SFR P6 only depends on the levels applied to the input lines. When used as analog input the required analog channel is selected by a three-bit field in SFR ADCON.

Since P6 is not a bit-addressable register, all input lines of P6 are read at the same time, by byte instructions. Nevertheless, it is possible to use port 6 simultaneously for analog and digital input. In order to guarantee a high-quality A/D conversion, digital input lines of port 6 should not toggle while a neighboring port pin is executing an A/D conversion, this could produce crosstalk to the analog signal.

2.2.3.7 A/D Converter [4]

The SAB 80C515/80C535 provides an A/D converter with the following features:

- Eight multiplexed input channels.
- The possibility of using the analog input channels (port 6) as digital inputs (ACMOS version only).
- Programmable internal reference voltages (16 steps each) via resistor array.
- 8-bit resolution within the selected reference voltage range.
- 13 machine cycle conversion time for ACMOS versions (including sample time).

- 15 machine cycle conversion time for MYMOS versions (including sample time).
- Internal start-of-conversion trigger
- Interrupt request generation after each conversion

For the conversion, the method of successive approximation via capacitor array is used; the externally applied reference voltage range has to be held on a fixed value within the specifications. The internal reference voltages can be varied to reduce the reference voltage range of the A/D converter and thus to achieve a higher resolution.

Figure 2-6 shows a block diagram of the A/D converter. There are three user-accessible special function registers: ADCON (A/D converter control register), ADDAT (A/D converter data register), and DAPR (D/A converter program register) for the programmable reference voltages.

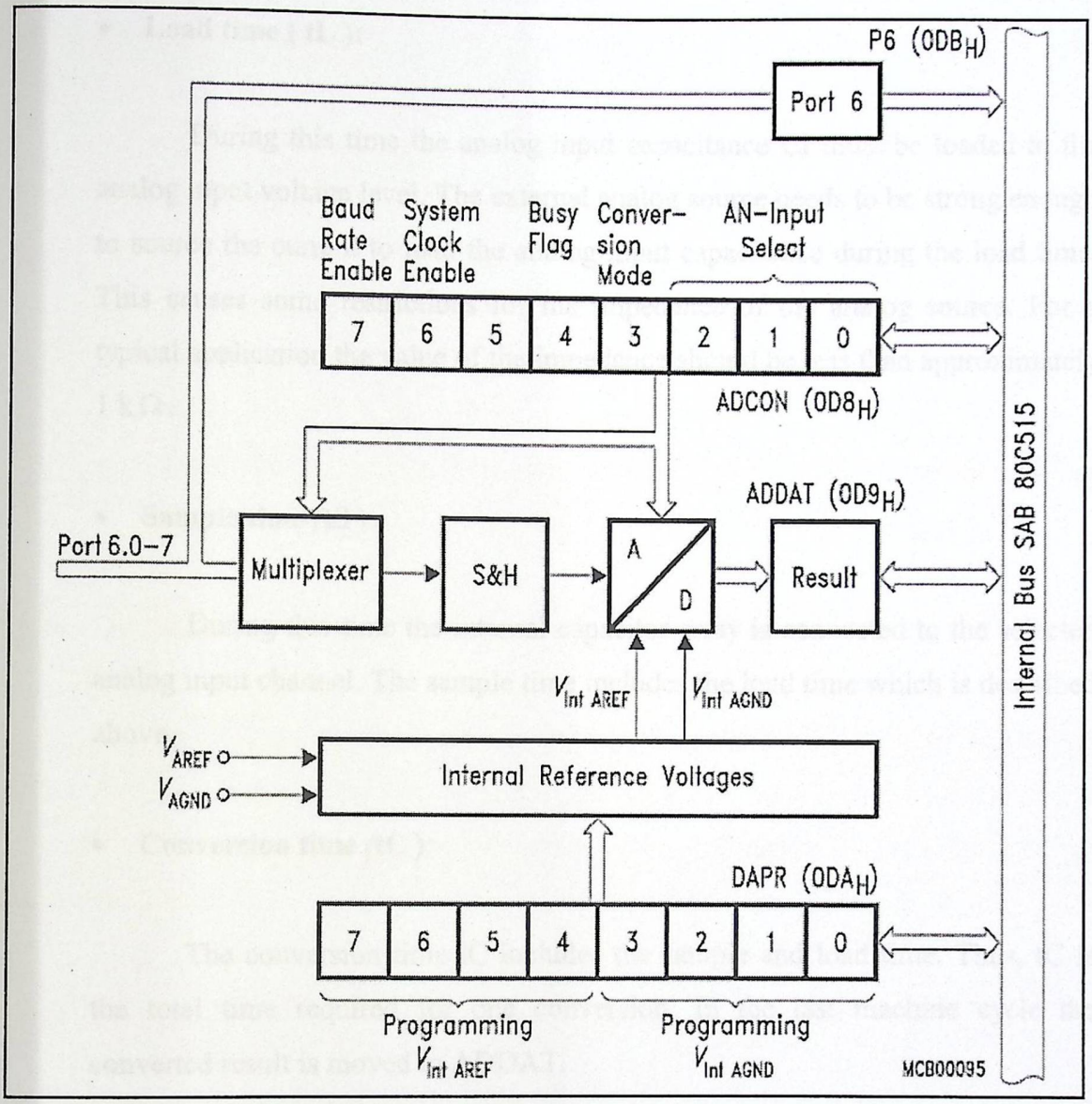


Fig 2-6: A/D Converter Block Diagram

2.2.3.7.1 A/D Converter Timing

A conversion is started by writing into special function register DAPR. A write to DAPR will start a new conversion even if a conversion is currently in progress. The conversion begins with the next machine cycle and the busy flag BSY will be set. The conversion procedure is divided into three parts (see ***Figure 2-7***):

- **Load time (tL):**

During this time the analog input capacitance CI must be loaded to the analog input voltage level. The external analog source needs to be strong enough to source the current to load the analog input capacitance during the load time. This causes some restrictions for the impedance of the analog source. For a typical application the value of the impedance should be less than approximately 1 kΩ.

- **Sample time (tS):**

During this time the internal capacitor array is connected to the selected analog input channel. The sample time includes the load time which is described above.

- **Conversion time (tC):**

The conversion time tC includes the sample and load time. Thus, tC is the total time required for one conversion. In the last machine cycle the converted result is moved to ADDAT.

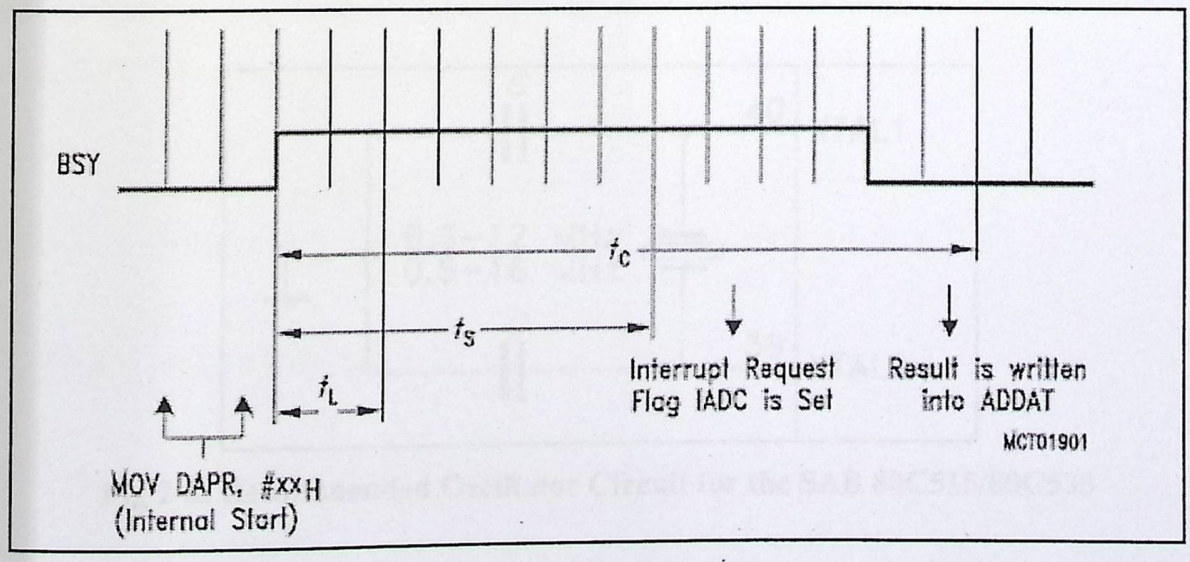


Fig 2-7: Timing Diagram of an A/D Converter

2.2.3.8 Power Saving Modes

For significantly reducing power consumption, the SAB 80C515/80C535 provides two Power Saving Modes:

- The Power-Down Mode: Operation of the component stops completely, the oscillator is turned off. Only the internal RAM is supplied with a very low standby current.
- The Idle Mode: The CPU is gated off from the oscillator. All peripherals are further supplied by the oscillator clock and are able to do their jobs.

2.2.3.9 Oscillator and Clock Circuit [4]

The oscillator is connected to XTAL1 and XTAL2 pins of the SAB 80C515/80C535. It drives the internal clock generator. The clock generator provides the internal clock signals to the chip at half the oscillator frequency. These signals define the internal phases.

2.2.3.9.1 Crystal Oscillator Mode

Figure 2-8 shows the recommended oscillator circuit.

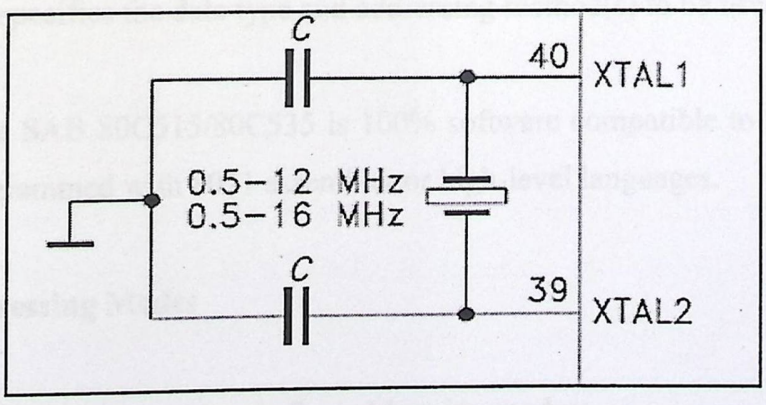


Fig 2-8: Recommended Oscillator Circuit for the SAB 80C515/80C535

2.2.3.9.2 Driving from External Source

For driving the SAB 80C515/80C535 from an external clock source, the external clock signal is to be applied to XTAL2, as shown in ***Figure 2-9***. XTAL1 has to be left unconnected.

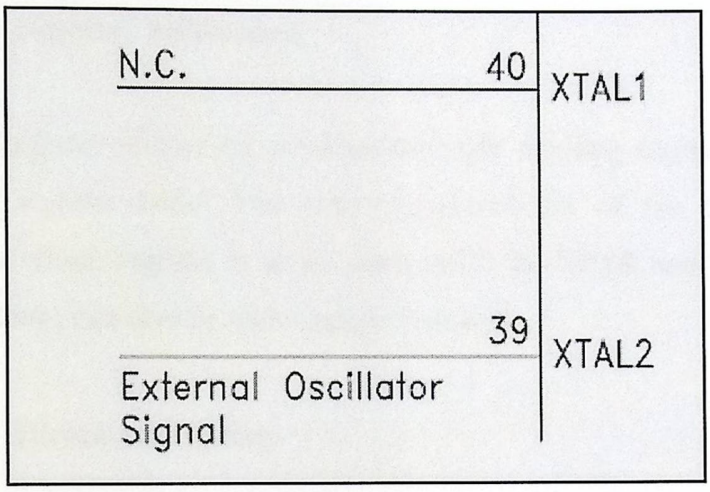


Fig 2-9: External Clock Source

2.2.3.10 Instruction Set [4][2]

The SAB 80C515/80C535 instruction set includes 111 instructions, 49 of which are single-byte, 45 two-byte and 17 three-byte instructions. The instruction opcode format consists of a function mnemonic followed by a "destination, source" operand field. This field specifies the data type and addressing method(s) to be used.

Thus, the SAB 80C515/80C535 is 100% software compatible to the SAB 8051 and may be programmed with 8051 assembler or high-level languages.

2.2.3.10.1 Addressing Modes

The SAB 80C515/80C535 uses five addressing modes:

- register
- direct
- immediate

- register indirect
- base register plus index-register indirect

Table 2-2 summarizes the memory spaces which may be accessed by each of the addressing modes.

- **Register Addressing**

Register addressing accesses the eight working registers (R0-R7) of the selected register bank. The least significant bit of the instruction opcode indicates which register is to be used. ACC, B, DPTR and Boolean processor accumulator, can also be addressed as registers.

- **Direct Addressing**

Direct addressing is the only method of accessing the special function registers. The lower 128 bytes of internal RAM are also directly addressable.

- **Immediate Addressing**

Immediate addressing allows constants to be part of the instruction in program memory.

- **Register Indirect Addressing**

Register indirect addressing uses the contents of either R0 or R1 (in the selected register bank) as a pointer to locations in a 256-byte block: the 256 bytes of internal RAM or the lower 256 bytes of external data memory. Note that the special function registers are not accessible by this method. The upper half of the internal RAM can be accessed by indirect addressing only. Access to the full 64 Kbytes of external data memory address space is accomplished by using the 16-bit data pointer.

Execution of PUSH and POP instructions also uses register indirect addressing. The stack may reside anywhere in the internal RAM.

- **Base Register plus Index Register Addressing**

Base register plus index register addressing allows a byte to be accessed from program memory via an indirect move from the location whose address is the sum of a base register (DPTR or PC) and index register (ACC).

Addressing Modes	**Associated Memory Spaces**
Register addressing	R0 through R7 of selected register bank, ACC, B, CY (Bit), DPTR
Direct addressing	Lower 128 bytes of internal RAM, special function registers
Immediate addressing	Program memory
Register indirect addressing	Internal RAM (@R1, @R0, SP), external data memory (@R1, @R0, @DPTR)
Base register plus index register addressing	Program memory (@DPTR + A, @PC + A)

Table 2-2: Addressing Modes and Associated Memory Spaces.

2.2.3.10.2 Introduction to the Instruction Set

The instruction set is divided into four functional groups:

- data transfer
- arithmetic
- logic
- control transfer

1. **Data Transfer**

Data operations are divided into three classes:

- general-purpose
- accumulator-specific
- address-object

None of these operations affects the PSW flag settings except a POP or MOV directly to the PSW.

2. **Arithmetic**

The SAB 80C515/80C535 has four basic mathematical operations: Addition, Subtraction, Multiplication, and Division, on both signed and unsigned integers.

3. **Logic**

The SAB 80C515/80C535 performs basic logic operations on both bit and byte operands.

4. **Control Transfer**

There are three classes of control transfer operations: unconditional (calls, returns, and jumps), conditional jumps, and interrupts. All control transfer operations, cause the program execution to continue a non-sequential location in program memory.

2.3 Relays [6]

A relay is an electrically operated switch. Current flowing through the coil of the relay creates a magnetic field which attracts a lever and changes the switch contacts. The coil current can be on or off so relays have two switch positions and they are double throw (changeover) switches.

Relays allow one circuit to switch a second circuit which can be completely separate from the first. For example a low voltage battery circuit can use a relay to switch a 230V AC mains circuit. There is no electrical connection inside the relay between the two circuits; the link is magnetic and mechanical.

The coil of a relay passes a relatively large current, typically 30mA for a 12V relay, but it can be as much as 100mA for relays designed to operate from lower voltages. Most ICs (chips) cannot provide this current and a transistor is usually used to amplify the small IC current to the larger value required for the relay coil.

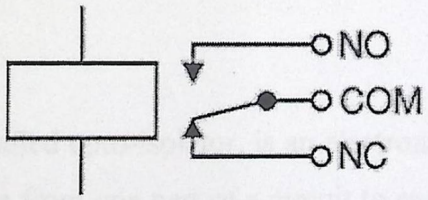


Figure 2-10: Circuit symbol for a relay

The relay's switch connections are usually labeled COM, NC and NO:

- COM = Common, always connect to this; it is the moving part of the switch.
- NC = Normally Closed, COM is connected to this when the relay coil is off.
- NO = Normally Open, COM is connected to this when the relay coil is on.

Advantages of relays:

- Relays can switch AC and DC.
- Relays can switch high voltages.

- Relays are a better choice for switching large currents (> 5A).
- Relays can switch many contacts at once.

Disadvantages of relays:

- Relays cannot switch rapidly.
- Relays use more power due to the current flowing through their coil.
- Relays require more current than many chips can provide, so a low power transistor may be needed to switch the current for the relay's coil.

2.4 Isolation Circuit

The aim of the isolation circuit is isolating the microcontroller board and other ICs from high voltage and back currents, the control may have high voltage that will damage the microcontroller, thus isolation eliminate ground loop; using an optocoupler a signal voltage is coupled from the input circuit to the output circuit with electrical isolation between the two circuits.

2.4.1 Optocouplers [5]

An optocoupler, also called opto-isolator, is an electronic component that transfers an electrical signal or voltage from one part of a circuit to another or from one circuit to another, while electrically isolating the two circuits from each other. It consists of an infrared emitting LED chip that is optically in-line with a light-sensitive silicon semiconductor chip, all enclosed in the same package. The silicon chip could be in the form of a photo diode, photo transistor, or photo Darlington.

2.4.1.1 Optocoupler Functions

- To isolate one section of a circuit from another, each section having different signal voltage levels to ensure compatibility between them.
- To prevent electrical noise or other voltage transients that may exist in a section of a circuit from affecting another section when both sections have a

common circuit reference. Noise or voltage transients can be caused by a poor printed circuit board layout.

2.5 Amplifiers [3][5]

It is an electronic circuit having the capability to amplify power, voltage, or current. The standard operational amplifier (op-amp) symbol is shown in ***Figure 2-11***, It has two input terminals, the inverting input (—) and the noninverting input (+), and one output terminal. The typical op-amp operates with two dc supply voltages, one positive and the other is negative.

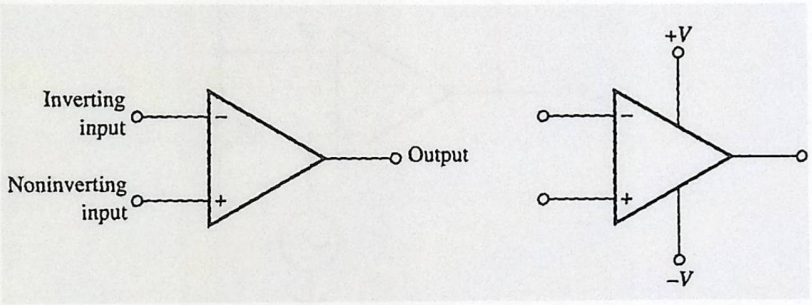


Fig 2-11: operational amplifier (op-amp) symbol

- Closed-Loop Voltage Gain (Av):

The closed-loop voltage gain is the voltage gain of an op-amp with external feedback; the amplifier configuration consists of the op-amp and an external negative feedback circuit that connects the output to the inverting input. The closed-loop voltage gain is determined by the external component values and can be precisely controlled by them.

The closed loop gain of the non-inverting amplifier (Av) is expressed as:

$$Av = 1 + \frac{Rf}{Ri}$$

The closed loop gain can be set by selecting values of Ri and Rf

- Noninverting Amplifier:

An op-amp connected in a closed-loop configuration as a noninverting amplifier with a controlled amount of voltage gain is shown in ***Figure 2-12***. The input signal is applied to the noninverting (+) input. The output is applied back to the inverting (—) input through the feedback circuit (closed loop) formed by Rf and Ri. The output voltage is expressed as:

$$Vout = (Av) * (Vin)$$

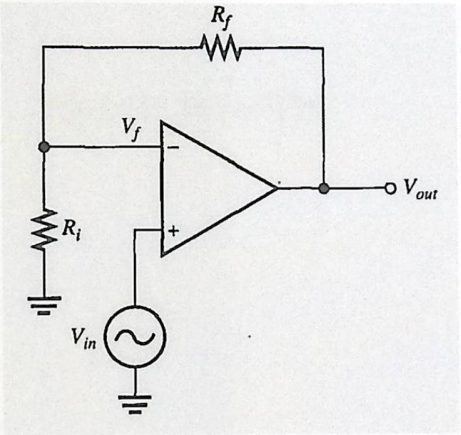


Figure 2-12: Noninverting Amplifier

2.6 Sensors

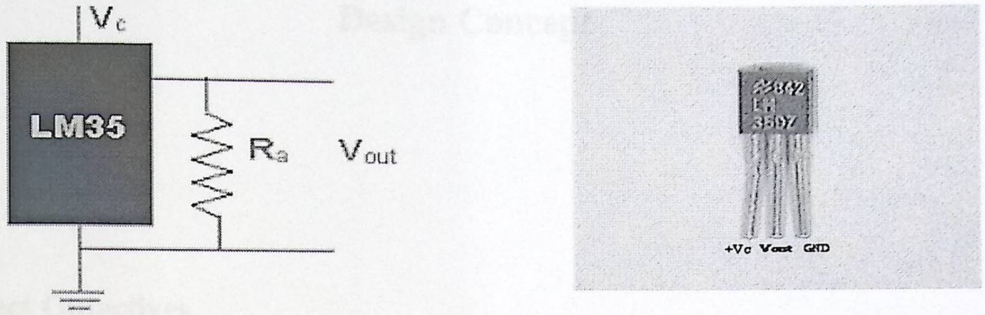
A sensor is a device, which responds to an input quantity by generating a functionally related output usually in the form of an electrical or optical signal. Some of the environmental parameters can be detected by human senses such as light, hearing, taste, smell, and touch, but sensor devices are even able to measure parameters like magnetic and electric field, radiation, gas concentration, chemical activity... etc .

2.6.1 Temperature Sensor - The LM35 [5]

The LM35 is an integrated circuit sensor that can be used to measure temperature with an electrical output proportional to the temperature (in °C).

The LM35 features:

- We can measure temperature more accurately than a using a thermistor.
- It has an output voltage that is proportional to the Celsius temperature.
- The scale factor is 0.01V/°C



a) b)

Fig 2-13: The LM35

Chapter Three

Design Concept

3.1 Project Objectives

3.2 General Block Diagram

3.3 Design Options

3.4 How the system will work

Chapter Three

Design Concept

3.1 Project Objectives

The goals and objectives of this project are:

- Design and implement a general purpose control board based on microcontroller, to Sense, measure, and control processes and events.
- Use a feed back to construct a closed loop system, to monitor and maintain the needed characteristics for the desired applications.
- Study the microcontroller concept, importance, structure, and programming.
- Study the applications of the microcontroller in the real life.
- design a required interface circuits for applications
- Determine required sensors for applications.
- Explain how the control will be done.

3.2 General Block Diagram

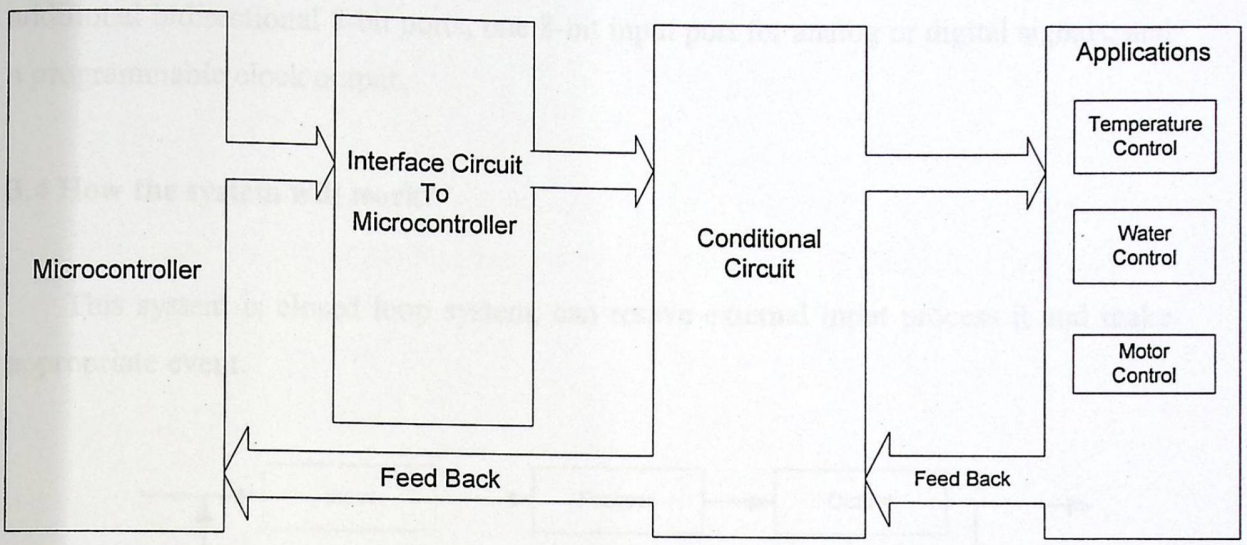


Fig 3-1: General Block Diagram

3.3 Design Options

1. Why using embedded system rather than computer system

Our project can be implemented using two options: first option is by using a microcontroller (embedded system), and the second option is by using a personal computer. We implement our project using a microcontroller rather than a personal computer because of the following reasons:

- We aim to deign a small and portable board, this aim achieved by using a microcontroller rather than a personal computer.
- Cost of microcontroller is cheaper than personal computer.

○ Why using SAB 80C515/80C535 microcontroller rather than others

SAB 80C515/80C535 is advanced microcontroller, and have many components that we need, so it saves time, efforts, and costs for connecting external additional components. The external data and program memory can be expanded up to 64 Kbytes and can be accessed by instructions that use a 16-bit or an 8-bit address.

The SAB 80C515/80C535 has a new 16-bit timer/counter. It also contains 16-bit watchdog timer, an 8-bit A/D converter with programmable reference voltages, two additional bidirectional 8-bit ports, one 8-bit input port for analog or digital signals, and a programmable clock output.

3.4 How the system will work?

This system is closed loop system, can resave external input process it and make appropriate event.

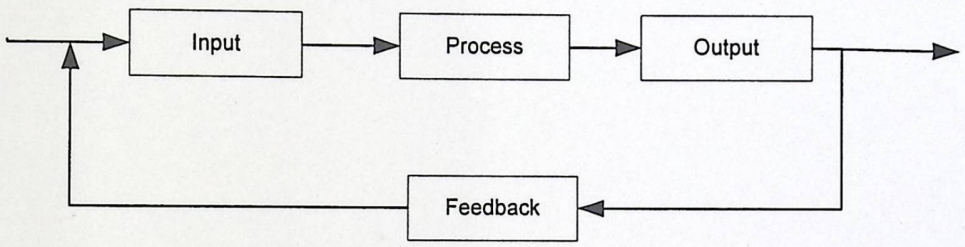


Figure 3-2: Closed loop system

Appropriate program will be written in the microcontroller depending on the application.

To approve our aims from this project we used three applications controlled by our designed system. These applications are:

- Motor direction control:

  Appropriate program depending on timer is written to control the direction of the motor directly, the motor turns left and right periodically.

- Water level control:

  To keep appropriate quantity of water in a tank, the microcontroller receives signals from three level sensors; these sensors should be placed at

appropriate levels in the tank. Depending on the combination of these signals, the microcontroller will turn a pump ON or OFF.

- Temperature control:

The microcontroller recognizes the temperature during temperature sensor, if it high the microcontroller turns cooler ON and turns heater OFF, and vice versa.

Chapter Four

Hardware System Design

4.1 Project Components

Chapter Four

Hardware System Design

4.1 Project Components

The project consists of many components; these components are connected to each other as shown in the detailed block diagram ***Figure 4-1.***

We divided the project into three phases, each phase represent a module. These phases are:

- Phase 1: Microcontroller and Interface circuits module.
- Phase 2: Switching circuits module.
- Phase 3: Feedback circuits module.

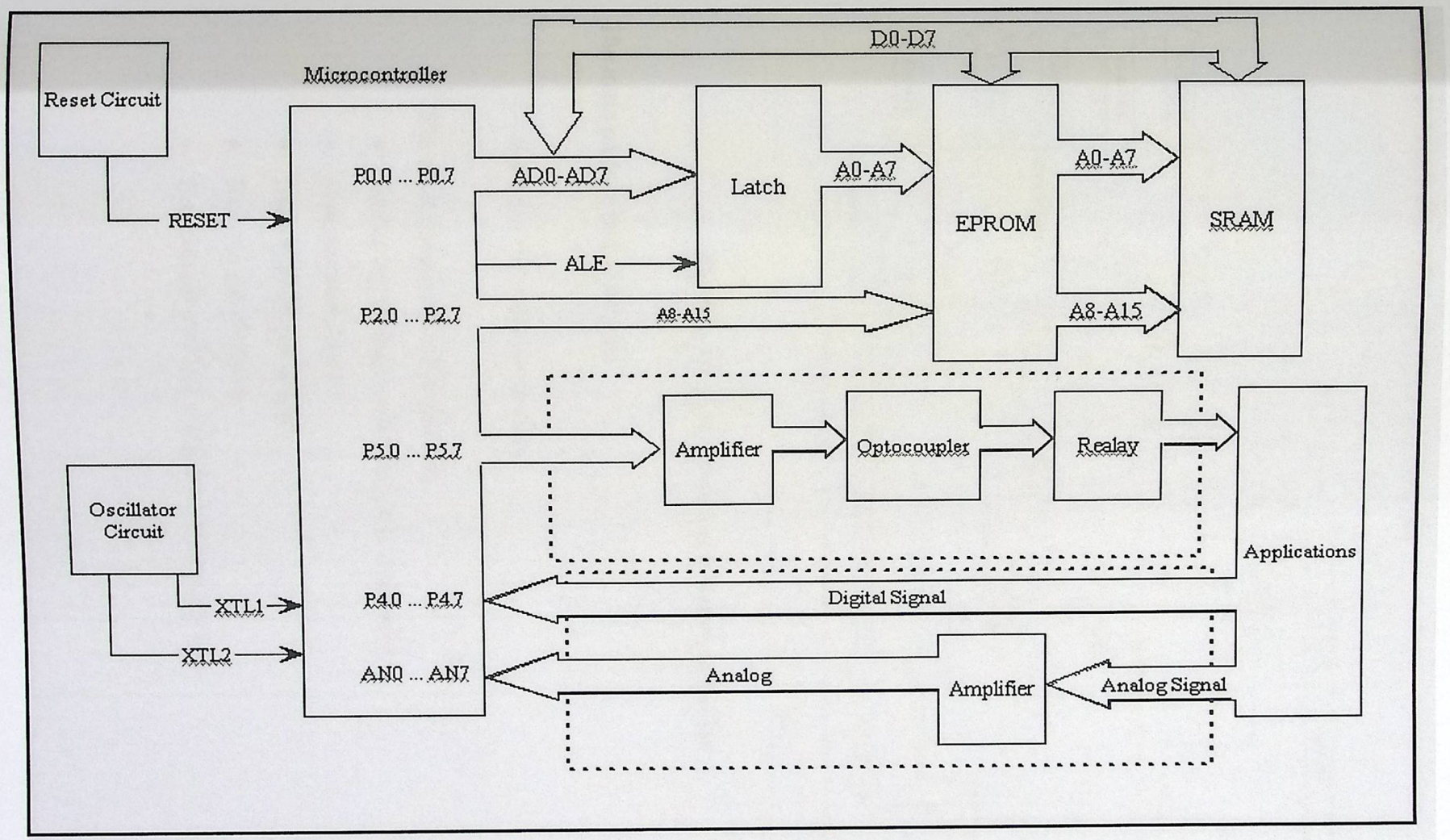


Fig 4-1: Detailed Block Diagram

Phase I: Microcontroller and Interface circuits

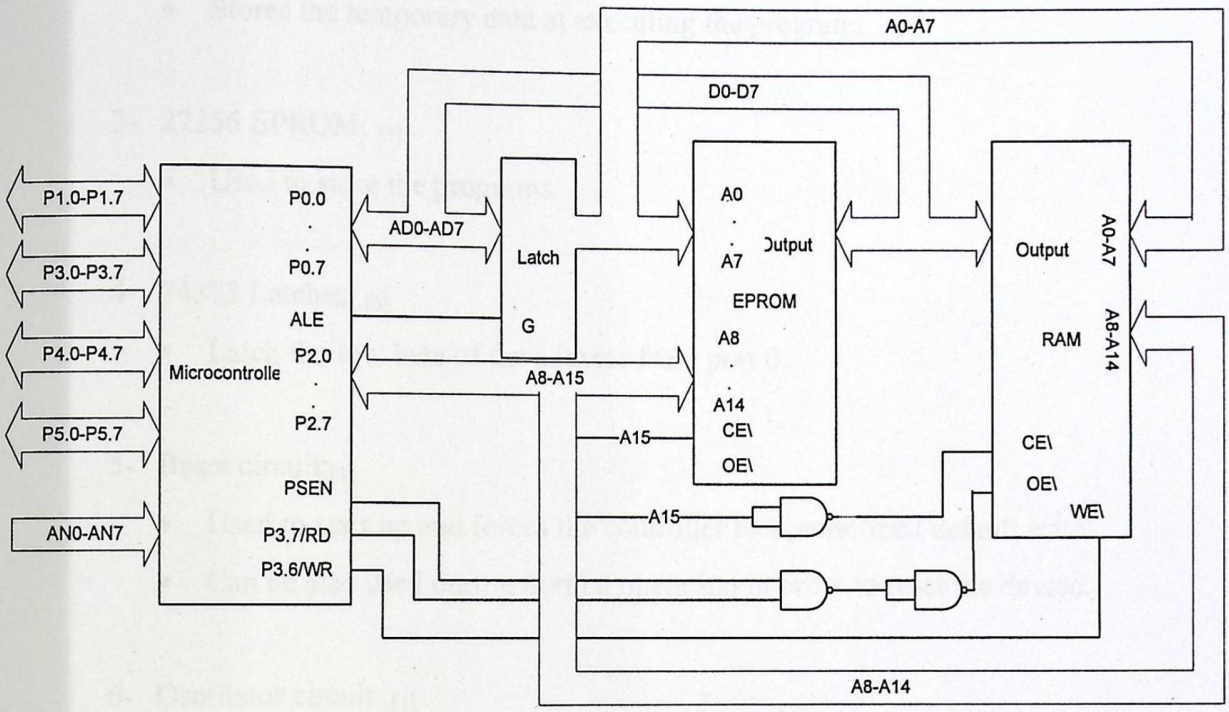


Fig 4-2: Microcontroller and Interface circuit

The main parts in this phase are:

1- SAB 80C535 microcontroller:

The main functions of this part are:

- Gather inputs from various sensors (feedback).
- Process these inputs.
- Issue appropriate outputs.
- Execute the program.
- Controls all attached applications and processes.

2- 62256 SRAM: [5]

- Used for executable code.
- Stores the temporary data at executing the program.

3- 27256 EPROM: [5]

- Used to store the programs.

4- 74373 Latches: [5]

- Latch the low byte of the address from port 0.

5- Reset circuit:[1]

- Used to start up and forces the controller to a predefined default state.
- Can be also used during normal operation in order to reset the device.

6- Oscillator circuit :[1]

- It drives the internal clock generator, which provides the internal clock signals to the chip at half the oscillator frequency

7- TL7805 regulator: [5]

- Used to obtain adjustable voltages and currents.
- Gives fixed output voltage, nearly 5 volts.
- The input voltage should be 2 steps larger than output voltage.

8- 74HCT00 NAND gate [5]

- It combines $\overline{PSEN}$ and $\overline{RD}$ signals to enable the external RAM.

9- Bypass capacitors: [5],[6]

- Connected between Vcc and GND for each chip.
- Allow circuits to function or work properly.

- Filter the electrical noise out circuits.
- Remove the alternating currents caused by ripple voltage.

Phase II: switching circuit module

In this module we designed several switches to interface external applications. Each switch includes the following components:

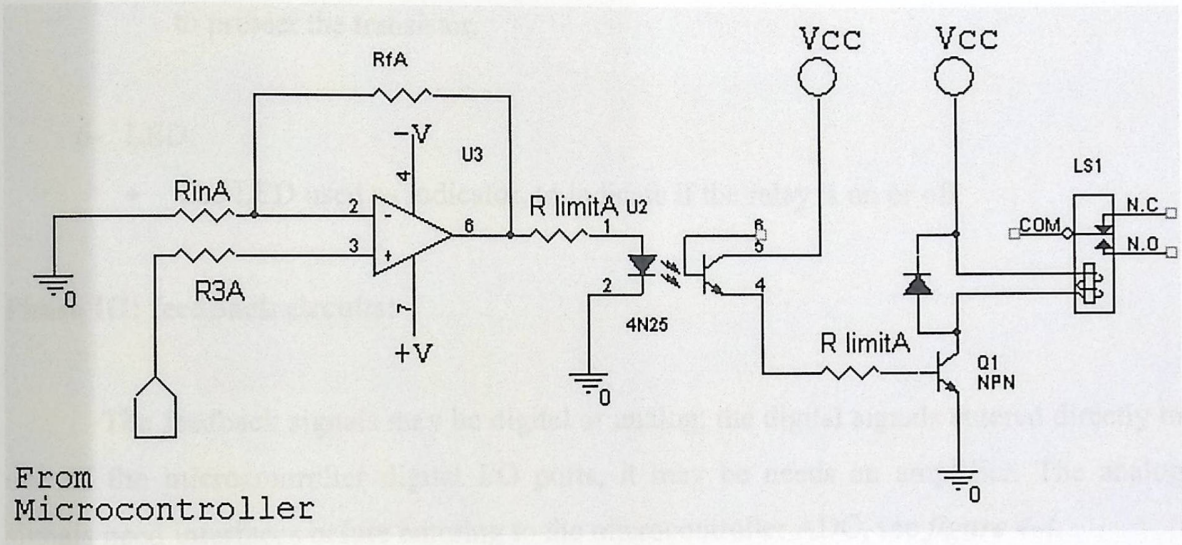


Fig 4-3: switch circuit

1- Fixed gain Op-Amplifier:[3]

- Used to Amplify the microcontroller output signal

2- Optocoupler: [5]

- Transfer the electrical signal from M.C transistor to switch the relay.
- To protect the M.C board from the high voltage.

3- Transistor: [5]

- It operates in cutoff and saturation modes to drive the selected relay, when a particular bit goes high the transistor connected to it is turned ON (in saturation mode), then the corresponding relay is turned ON.

4- Relay: [6]

- Electrically operated on/off switch.
- When the transistor is turned on, a small current passes through the coil, the coil generate magnetic field which move the switch.

5- Fly back diode: [5][6]

- It is connected across the relay coil to clamp the back EMF caused by the coil, to protect the transistor.

6- LED:

- The LED used as indicator, to indicate if the relay is on or off

Phase III: feedback circuits:

The feedback signals may be digital or analog; the digital signals entered directly to one of the microcontroller digital I/O ports, it may be needs an amplifier. The analog signals need interfaces before entering to the microcontroller ADC, see ***figure 4-4.***

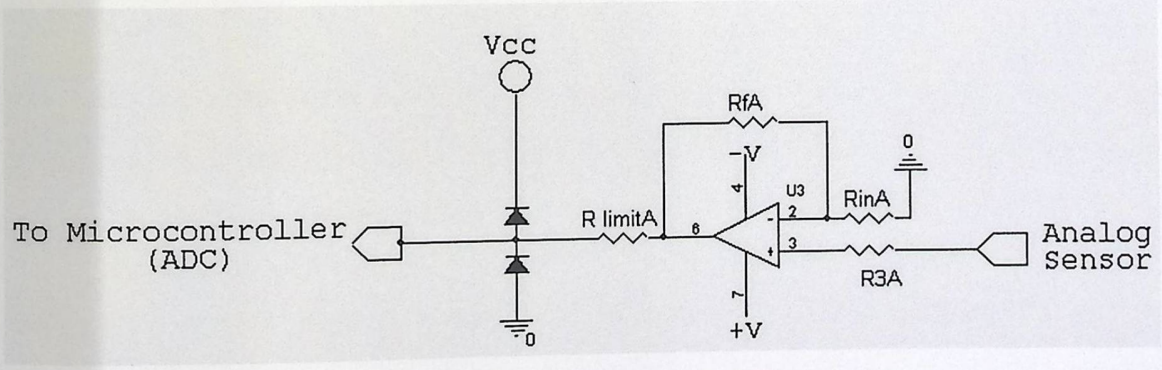


Fig 4-4: Feedback analog interface circuit

In this phase we used the following components:

1- Level sensor:

- Used to sense the water level in a tank.

- We used three-carbon rods taken from 1.5 volt dry batteries, as level sensors.

2- Fixed gain Op-Amplifier:

- Amplifies the output signal received from external application (sensors), before these signals entered to the microcontroller.

3- Clamping diodes: [5][6]

- To ensure that the input to the microcontroller ADC is between Vcc and GND.

Chapter Five

System Software

5.1 Introduction

5.2 Main Flowchart of the System

5.3 Temperature control

5.4 ADC operation

5.5 Water level control

Chapter Five

System Software

5.1 Introduction

Software plays the main role in developing control systems, and each application needs appropriate software for controlling.

In our project the software is essential, since the board is used to control the relays and sends appropriate signal depending on a predefined condition and a received feedback signals.

The language used to write the software is the assembly language for the 80c515/80c535 microcontroller which has the following feature:

- Similar to a well known 8051 microcontroller assembly language.
- There is a good simulator to write, compile, and debug the programs and trace the execution of the program.

5.2 Main Flowchart of the System

The microcontroller reads the signal (digital or analog), and according to a predefined condition, it performs some events (like switch an application ON or OFF), but if the signal is analog, the microcontroller converts it to digital value before taking any event. And so on. As shown in ***figure 5-1***.

The microcontroller reads signals during the ports, if the reading ports are port 0 to port 5 the input signal will be digital, but if reading port is port6 the input signal will be digital or analog and the microcontroller can distinguish between them by the software. Reading the analog signal will be described later in ADC operation flowchart.

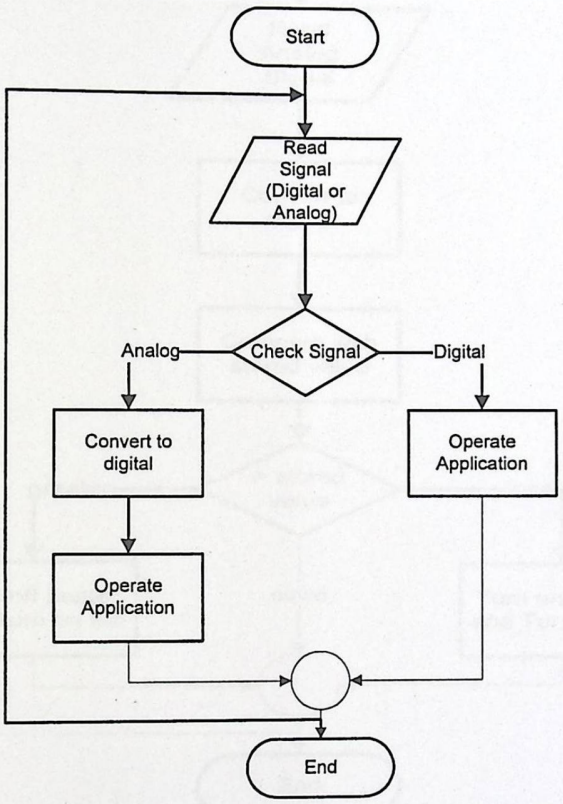


Fig 5-1: Main flowchart

5.3 Temperature control

The microcontroller reads the signal from an analog temperature sensor, converts the analog signal to a digital value (this event will be described later in ADC operation flowchart), and compare this value to a value or range of values (according to the application) stored in EPROM, if the temperature is high, the microcontroller turn the heater off and turn the cooler (fan) on, if the temperature is low, the microcontroller turn the heater on and turn the cooler (fan) off. But if the reading value equal to the stored value

the microcontroller turns off the heater and cooler, and continue checking sensor signal. As shown in ***figure 5-2***.

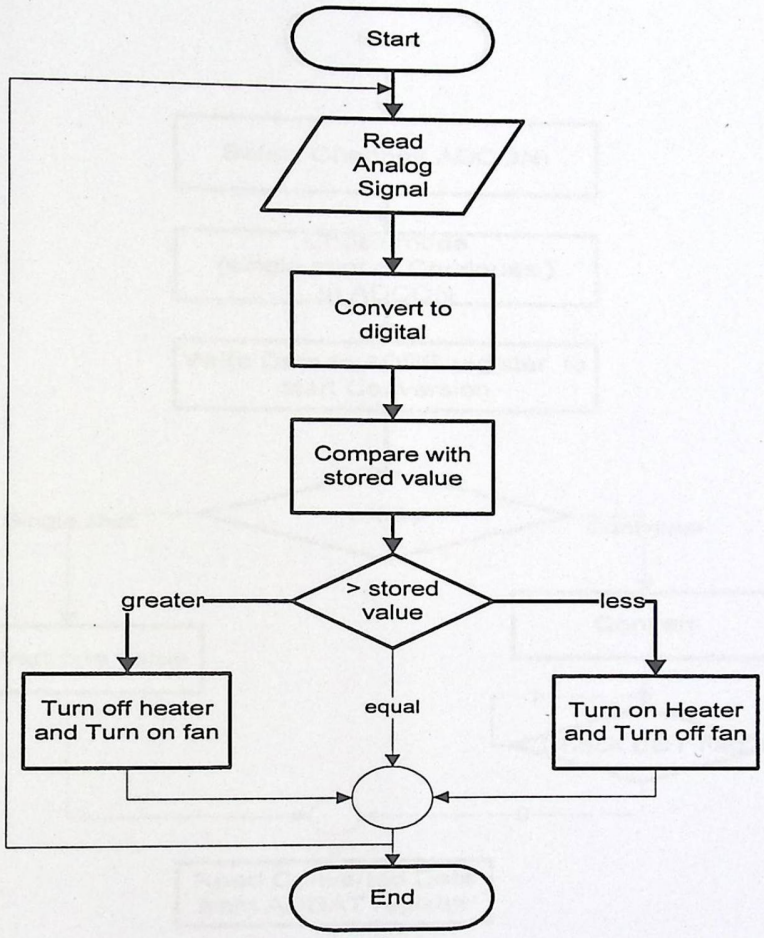


Fig 5-2: temperature flowchart

5.4 ADC operation

There are four general steps involved in analog-to-digital conversion. The first task is setup, followed by the conversion start instruction (The ADC starts when a byte is written to DAPR register), wait for conversion completion (the status bit BSY of ADCON register is set while the conversion is in progress, the application program poll this bit and read the value when BSY is cleared), and finally, reading the conversion results (the result is stored in ADDAT register).

In ADC setup, three tasks need to be performed: setup the reference voltages, select the input channel, and select the conversion mode (one shot or continuous). As shown in *figure 5-3*.

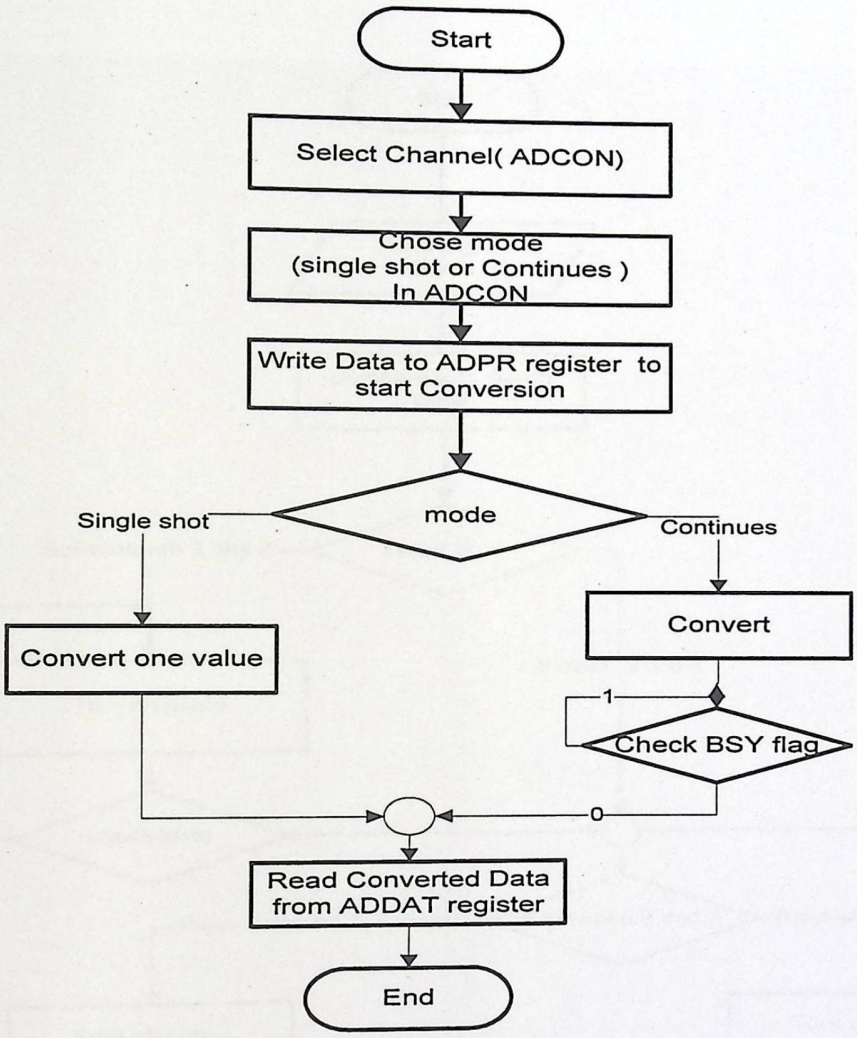


Fig 5-3: ADC flowchart

5.5 Water level control

We used three-carbon rods taken from 1.5 volt dry batteries, as level sensors to measure the water level in a tank. One is placed at the button, one at the top of the tank. and the other placed between them, If the water level is down under the middle sensor, the output of the inverter is low, then the microcontroller will turn on the pump to full the tank.

When the water reaches the top sensor, then the output of the inverter will be high, then the microcontroller will turn off the pump. the microcontroller continuously check the inverter output.

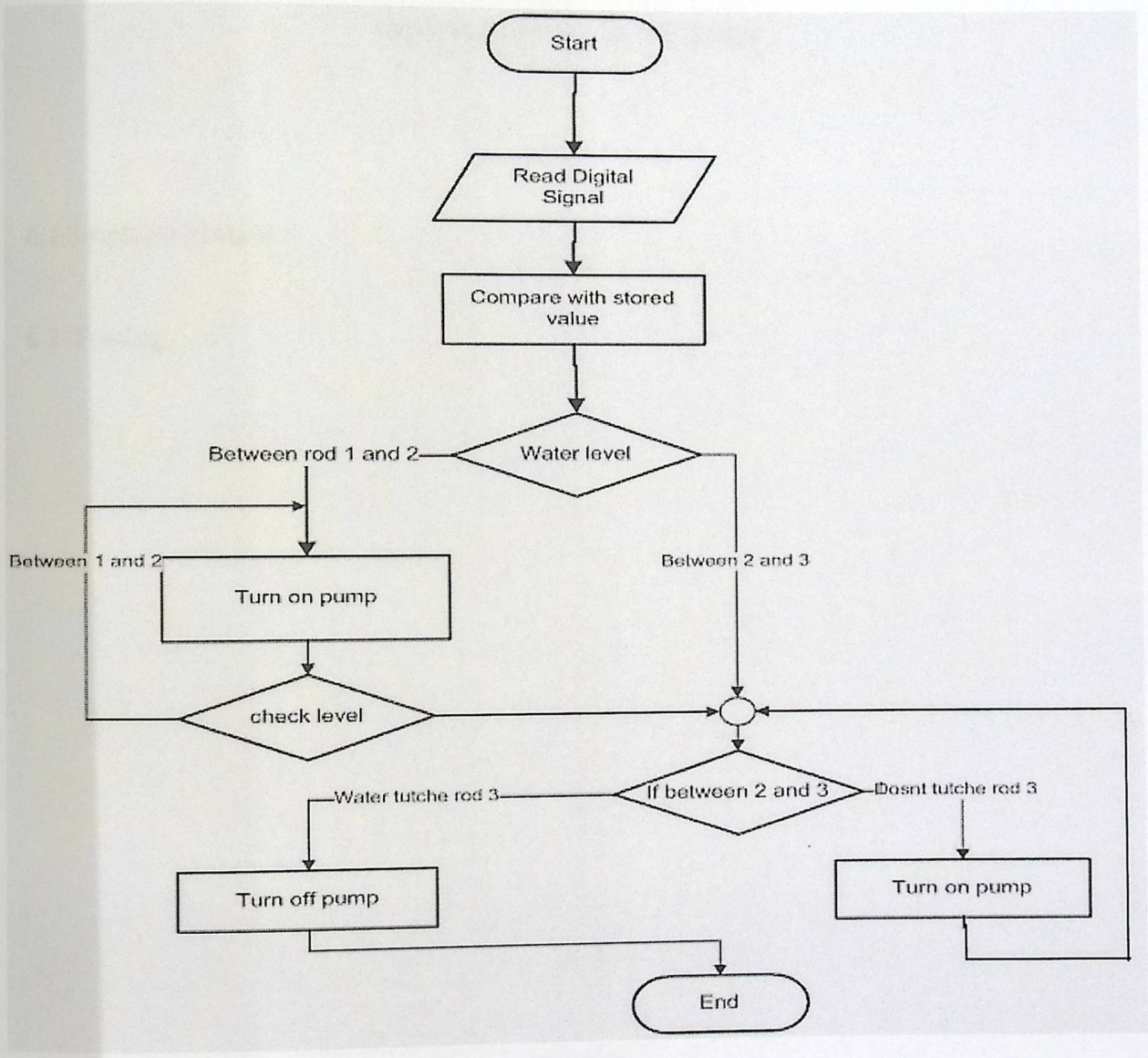


Fig 5-4: water level flowchart

Chapter Six

Implementation and Testing

6.1 Implementation

6.2 Testing

Chapter Six

Implementation and Testing

In this chapter we will discuss the implementation and testing procedure that have been followed in developing the system.

6.1 Implementation

After we finished the design of the system, drew the system schematic, and tested each chip individually, we started the implementation of the system using wire rapping, module by module. We used the following tools and equipments in implementation:

- Bread board.
- DC power supplies.
- Wires
- Wire raping.
- Plugs, sockets and terminals.
- Crocodile clips.
- Digital multimeter.
- Oscilloscope.
- Oscilloscope lead and probes.
- IC sockets.
- Soldering iron.
- Soldering iron stand.
- Desoldering pump (solder sucker)

- Solder remover wick (copper braid).
- Reel of solder.
- Side cutters.
- Small pliers.
- Small electric drill.

Firstly, we implemented the microcontroller interface circuit module; we placed the socket for each chip in this circuit (microcontroller, EPROM, RAM, latch, NAND gate, reset and oscillator) on in the board. Then we connected these sockets altogether as shown in the schematic diagram in appendix B.

We connected all pins of the microcontroller to external sockets to deal these pins easily, and connected the oscillator circuit (12 MHz, and two 22 pF capacitor) between the two pins XTL1 and XTL2 of the microcontroller, then connected the reset circuit (push button, capacitor, and two resistors) to the microcontroller RESET pin.

The $\overline{PE}$ and $\overline{EA}$ pins of the microcontroller was connected to the GND, to enable power saving modes and external memory respectively. VAREF and VAGND of the microcontroller was connected to Vcc and GND respectively, these two pins are the references of the internal microcontroller ADC.

The microcontroller port 0 was connected to a latch (to latch the low address byte), the enable pin of the latch (OE) was connected to the ALE output signal of the microcontroller. The microcontroller port 2 (high address byte) and the output of the latch was connected to 15 bit address of the EPROM, and RAM, the address line A15 (P2.7) was connected directly to chip enable pin ($\overline{CE}$) of the EPROM, and its connected through inverter to the $\overline{CE}$ pin of the RAM.

$\overline{PSEN}$ pin of the microcontroller was connected to the output enable ($\overline{OE}$) pin of the EPROM, and combined with read output signal ($\overline{RD}$) through NAND gate and

connected to the $\overline{OE}$ of the RAM. The write output signal ($\overline{WR}$) pin of the microcontroller was connected directly to the RAM $\overline{WR}$ pin.

Finally, we connected common Vcc and common GND for all ICs in this module, and for each IC we connected a bypass capacitor between Vcc and GND to protect the IC.

After that, we implemented the switching circuit module, this module contains several switches. Each switch consists of an amplifier, optocoupler, NPN transistor, and a relay. The sockets for these components are placed on the board, and connected as shown in the schematic diagram in appendix B. The input of the switches is coming from the microcontroller I/O port.

For each switch, firstly, we connected the amplifier and connected its output to the optocoupler, and then the optocoupler connected to the base of the transistor through limiting resistor. Finally, we connected the relay, one end of the coil connected to Vcc, and the other connected to collector of the transistor, we connected diode in parallel with relay coil, to prevent the relay from back feeding the circuit in an undesired manner. This process was repeated for each switch.

Then we implemented the final module (feedback circuits), each circuit contains amplifier (if necessary), resistor, and two clamping diodes - for the analog inputs -. The sockets for these components are placed on the board, and connected as shown in the schematic diagram in appendix B.

The input of the amplifier is coming from an external analog sensor, and its output was connected to the clamping diodes, then to one of the analog inputs of the microcontroller ADC. The feedback digital signal was connected directly to other digital I/O port of the microcontroller.

Finally, we connected these modules altogether to construct our full system board as shown in schematic diagram in appendix B.

To produce our system in a nice form we put the boards in a suitable box, and placed the necessary power supplies in it.

Note: we connected each IC as specified in its datasheet, see appendix A

6.2 Testing

In hardware design and implementation, debugging and testing are considered to be the most important phase. Testing made in a way that can easily detect the error. The procedures that we used in testing the system depend on step by step and bottom to up strategy. For testing, we used the same equipments and tools mentioned in implementation.

Our testing process included the following stages:

- Chip testing.
- Subsystem testing.
- System testing.
- Software testing.
- Integration testing.

6.2.1 Chip testing

Before connecting any chip, we made sure that each chip is working properly, as described below:

- Testing the regulator:

We connected it on a separate bread board, then we input to it nearly 7 volts, we obtained 5 volts on the output.

- Testing the microcontroller and 12 MHz crystal:

We connected the microcontroller with the 12MHz crystal, then connected Vcc and GND to the microcontroller, to see the ALE and PSEN signals we connected port0 (low byte address), to ground, i.e. the microcontroller will execute NOP instruction. Then we used the oscilloscope to see these signals. And also we checked the oscillator frequency and clock out signal. We noticed that these signals are similar to the signals in the microcontroller datasheet.

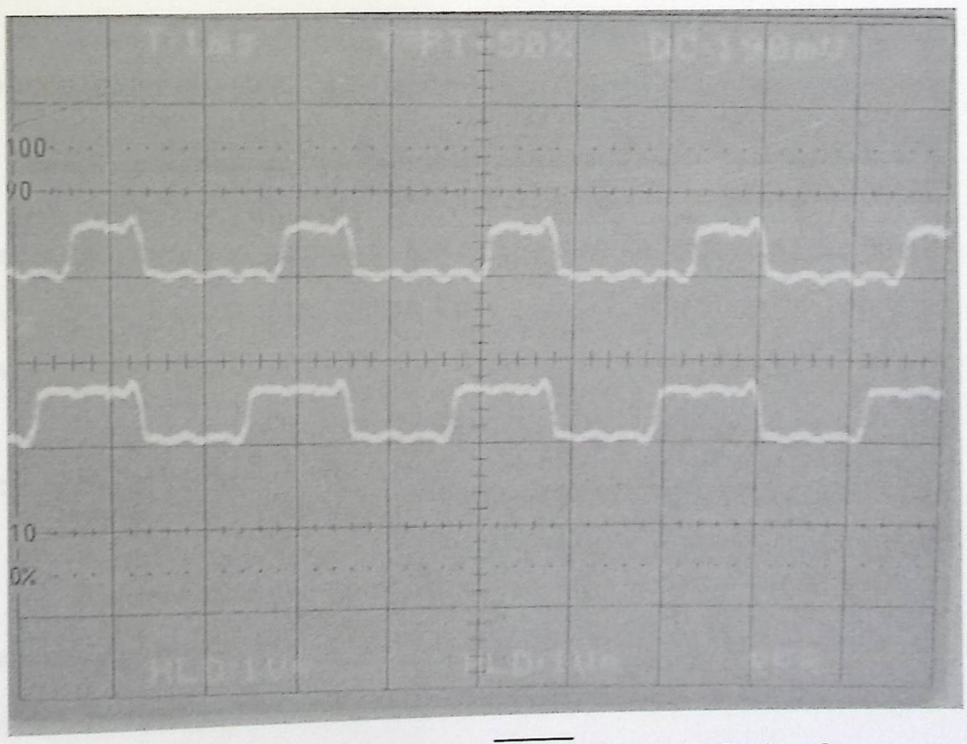


Fig 6-1: ALE (the first) and $\overline{PSEN}$ (the second) signals

- Testing the regulator:

We connected it on a separate bread board, then we input to it nearly 7 volts, we obtained 5 volts on the output.

- Testing the microcontroller and 12 MHz crystal:

We connected the microcontroller with the 12MHz crystal, then connected Vcc and GND to the microcontroller, to see the ALE and PSEN signals we connected port0 (low byte address), to ground, i.e. the microcontroller will execute NOP instruction. Then we used the oscilloscope to see these signals. And also we checked the oscillator frequency and clock out signal. We noticed that these signals are similar to the signals in the microcontroller datasheet.

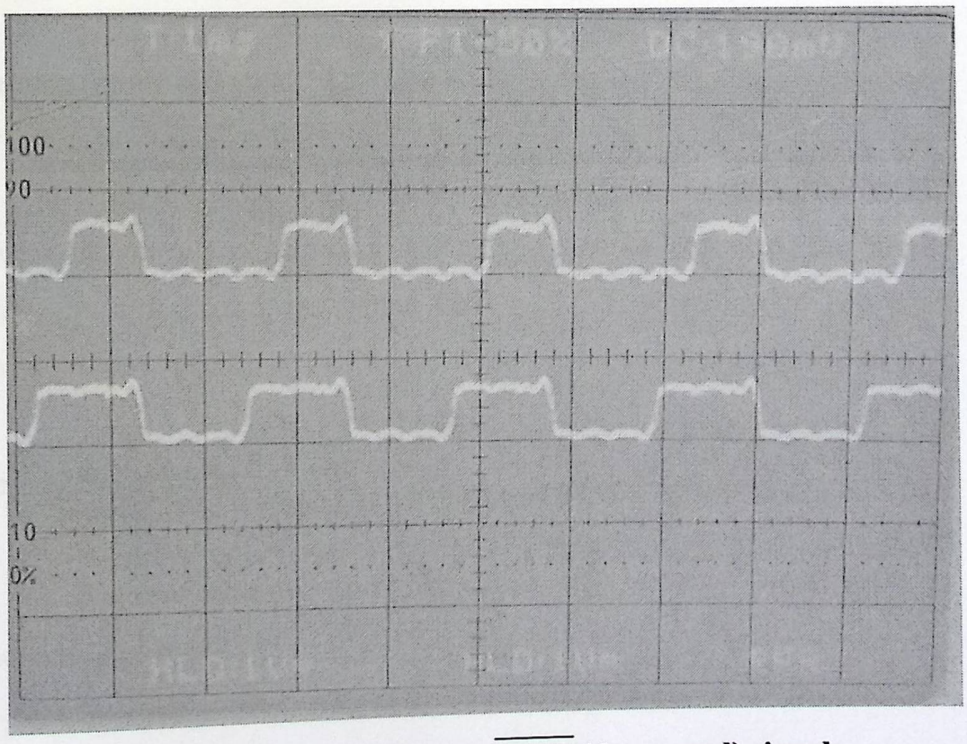


Fig 6-1: ALE (the first) and $\overline{PSEN}$ (the second) signals

- Testing the latch:

We connected it separately on a bread board, and connected its input to dip switch, and the output to LEDs, then when we activated the OE pin, the inputs are latched to output, i.e. the value of dip switch was shown on the output LEDs.

- Testing the EPROM:

Using EPROM programmer, firstly, we read the EPROM; we noticed that all bytes on it are FFh. Then we programmed and verified the EPROM by writing random bytes to it, this process was successfully finished, so we observed that our EPROM is properly worked, to be more surely, after programming EPROM, we read it again, and we saw the original written program, so the EPROM was properly worked.

- Testing the NAND gate:

Using a separate bread board, we connected the inputs of the NAND gate to switches and outputs to LEDs, and then we connected the NAND gate Vcc and ground, the output LED was ON, when one of two input switches was 0 (zero), and OFF only if the two input switches were 1's.

- Testing the amplifier:

We connected the amplifier circuit on a separate bread board as shown in ***figure 6-2***

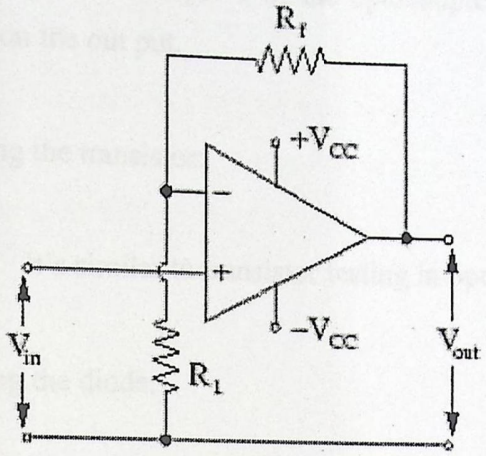


Fig 6-2: Noninverting operational amplifier

We used it as a non inverting fixed gain operational amplifier, the voltage gain depends on Rf and Rin ($Av = \frac{Rf}{Rin} + 1$), when we used Rf=100KΩ, Rin=50KΩ, and Vin=1 (Av=3), volt, we obtained 3 volts on the output, and when we used Rf=50KΩ, Rin=10KΩ, , and Vin=1 volt (Av=6), we obtained 6 volts on the output.

- Testing the optocoupler:

The optocoupler consists of two part, a Led and a transistor, firstly we tested the LED (between pins 1 and 2) using multimeter in diode mode, the value between them was approximately 1.02 when we connected 1 to positive and 2 to common of the multimeter, but when we changed the connection polarity between 1 and 2 no value was appeared on multimeter.

Then we tested the transistor, pin 5(collector), pin 4(emitter), and pin 6(base). The voltage between base and emitter (Vbe) was 0.7 V, and so between base and collector (Vbc). Finally to test two parts of optocoupler,

we entered 5 V to pin 1 of the optocoupler, we obtained approximately 5 volts on the out put.

- Testing the transistor:

  It's similar to transistor testing in optocoupler.

- Testing the diode:

  It's similar to LED testing in optocoupler.

- Testing the relay:

  We connected two LEDs to the relay, one connected between Vcc and normally open pin of the relay, and the other connected between Vcc and normally close pin of the relay, we connected the common pin of the relay to ground. When Vcc of coil = 5 V, the LED connected on normally open was ON, but when Vcc was = 0 V, the LED of normally close was ON. We noticed that only one of them operated at a time.

- Testing the temperature sensor:

  We connected the circuit of temperature sensor on a separate board, then we measured the output voltage from the out pin of the sensor, which directly proportional to ambient temperature (each 1mV = 1 C). We measured different temperatures using this sensor and using thermometer, we noticed that the measured values are approximately equal.

- Testing level sensor:

We used two carbon rods taken from 1.5 V batteries as level sensors. One of them was connected to ground, and the other connected to an inverter, we noticed that the output of inverter is 5 V when the two rods are in water, but when one of them didn't touch water, the output was zero volts.

6.2.2 Subsystem testing

- Testing the microcontroller board

After we implemented this module, we wrote a simple program that read switches connected to port 4 of the microcontroller, and out the value to LEDs connected to port 5 of the microcontroller; from this test we get the following results:

1. The microcontroller successfully read and successfully writ using I/O ports.
2. ALE, $\overline{PSEN}$, $\overline{RD}$, and $\overline{WR}$ signals are appeared properly on the oscilloscope.

We tested all I/O ports of the microcontroller, and we obtained the same results. From these result we observed that this subsystem is worked properly.

- Testing the relay and isolation circuit

After we implemented this module, we connected a LED to each relay (to normally open), we entered 1.5 volt on the amplifier (Av = 3), we noticed that the LED of corresponding relay was ON. We repeated this test for all switches, and we get the same result. From these results we observed

that each component of the designed switch (amplifier, optocoupler, transistor, and relay) was successfully operated, and also the integration between them is correct and successfully operated.

6.2.3 System testing

After we tested each module separately we combined them altogether to construct our system, then we do the following:

1. We connected the inputs of the switch circuit to the port 5 of the microcontroller, and then we wrote a program to turn on the LEDs connected to the switches periodically.

2. We let the same connection as previous test, and then wrote a program which check P4.0 (the first bit of port 4 of the microcontroller), if it high, the microcontroller will out FFh on port 5, so the LEDs connected to the switches will be turned ON, but if port 4.0 is low, the LEDs will be OFF.

From these tests we obtained a correct result, and so, we observed that the interface between subsystems is correct and our system is worked properly.

6.2.4 Integration testing

At this stage of testing we tested our final system connecting to external applications, these applications are:

- DC motor direction control.
- Water level control.

We wrote appropriate programs to EPROM to control these applications; these programs are listed in appendix C.

We noticed that these applications worked properly in our system, from these result we sure that our system is working properly.

Chapter Seven

Conclusion and future work

7.1 Problems and risks

7.2 Conclusions

7.3 Future Work

Chapter Seven

Conclusion and future work

In this chapter we will explain the problems and risks we faced during the work in the project, and also we will explain the results, conclusions, and future works of the project.

7.1 Problems and risks

During the work in the project, many problems, risks, and difficulties faced us, such as:

1. The lack of project component and chips which were not available for long time, so we take time to find it.
2. The late delivering of the microcontroller.
3. Connection problems; because we used a plenty of wires and wire wrapping technique, we faced several problems, such as sudden wires disconnection. And also we faced several occurrences of shorts between wires and between pins; because of very short distances between wires and between pins.

7.2 Conclusions

In the project, we have passed through many experiences and we faced many problems until we reaching the project goals, we have learned different approaches and skills from the practical implementation of the board, such as the way of thinking to complete some tasks or to solve some problems, and how to use different tools and utilities.

There were different problems that have faced us from the starting to the end of the project especially in the implementation stage; in this stage we learned how to trace the different signals step by step, chip by chip, and part by part, and how to develop an approach to solve the problems.

By the end of the project time, we have finished designing and implementing the General Purpose Control Board; this system enables controlling many applications, and receiving feedback, which satisfies the user needs, and can be applicable in real life

7.3 Future Work

1. Developing the system to control several applications and to read from more sensors.

2. Developing the system to construct the following suggested projects:

   A. Intelligent Instrumentation, such as:

   - Multimeter

     Using a keypad and an LCD screen, use the analog-to-digital convener to measure voltage, with additional circuitry; to measure current, resistance, inductance, and capacitance.

   - Function Generator

     Store the wave forms digitally in EPROM and generate the wave forms by a digital to-analog converter. Use the timers to control the wave form frequency.

B. Weather Center

Use the microcontroller as a data collection device measuring temperature, humidity, and wind direction, and strength. Use a timer to implement a time-of-day clock, store daily highs and lows.

C. Protocol Converter

Having both serial and parallel ports, a microcontroller is an ideal instrument to convert parallel information to serial information. For example, printers that require parallel inputs may be connected to the protocol converter and driven from a longer distance from a serial port.

D. Toys and Hobbies

There are many opportunities to make toys more intelligent.

3. Implementing a printed circuit board for this project.

References

1. Sencer Yeralan, Ashutosh Ahluwlia-"Programming and Interfacing the 8051 microcontroller"-5th Edition-Addison Wesley Longman, Inc-1997.

2. Ramesh S.Gaonkar- "Microprocessor Architecture, Programming and applications with the 8085"-5th Edition- Prentice Hall – New Jersey-1999.

3. Floyd- "Electronic Devices"- 5th Edition- Prentice Hall- Upper Saddle River, New Jersey-1999.

4. Infineon technologies: www.infineon.com

5. datasheets: www.datasheetcataluge.com

Appendices

Appendix A: Microcontroller and data sheets

Appendix B: Schematic diagram

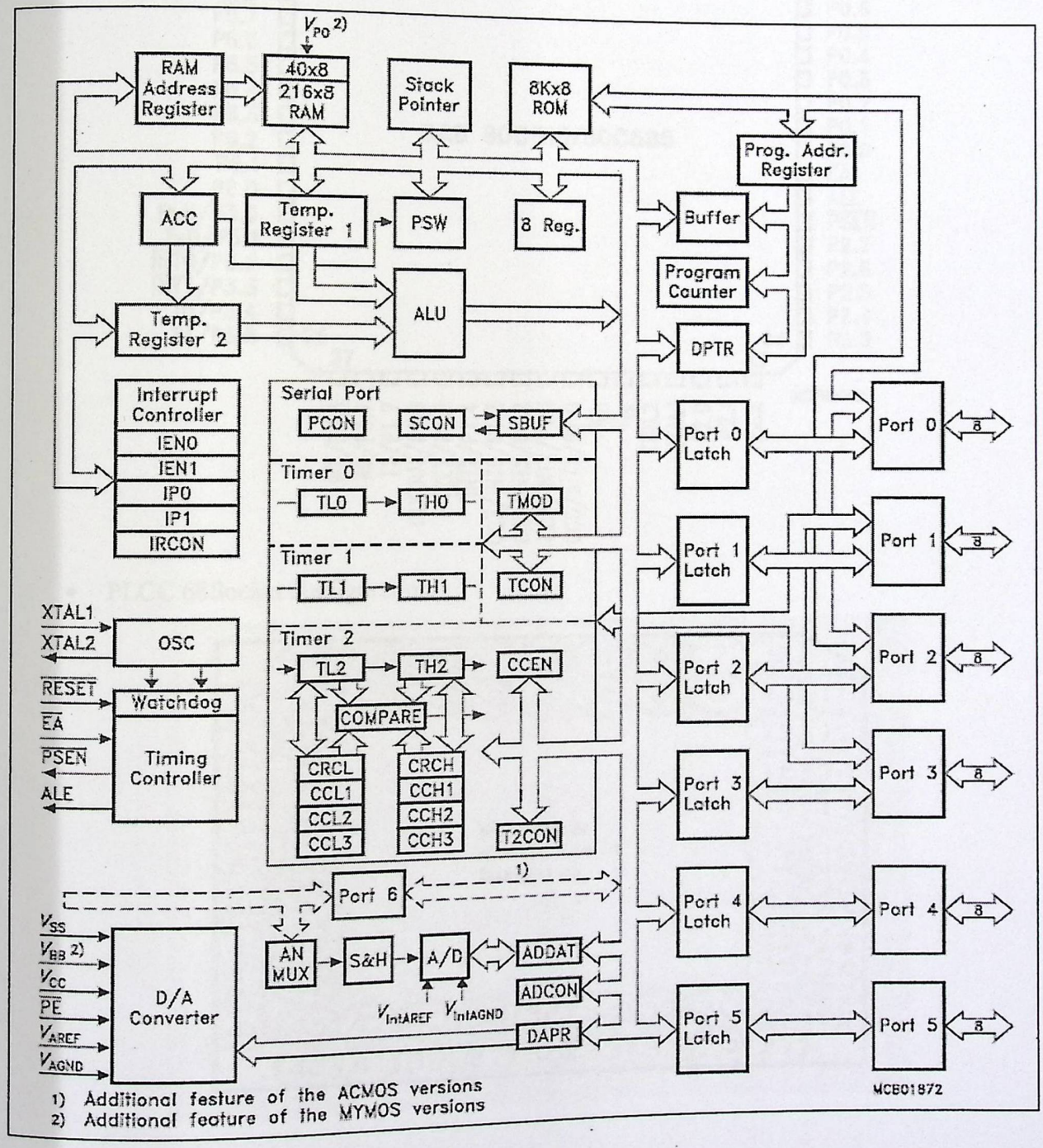
Appendix C: Software programs and subroutine

Appendix A

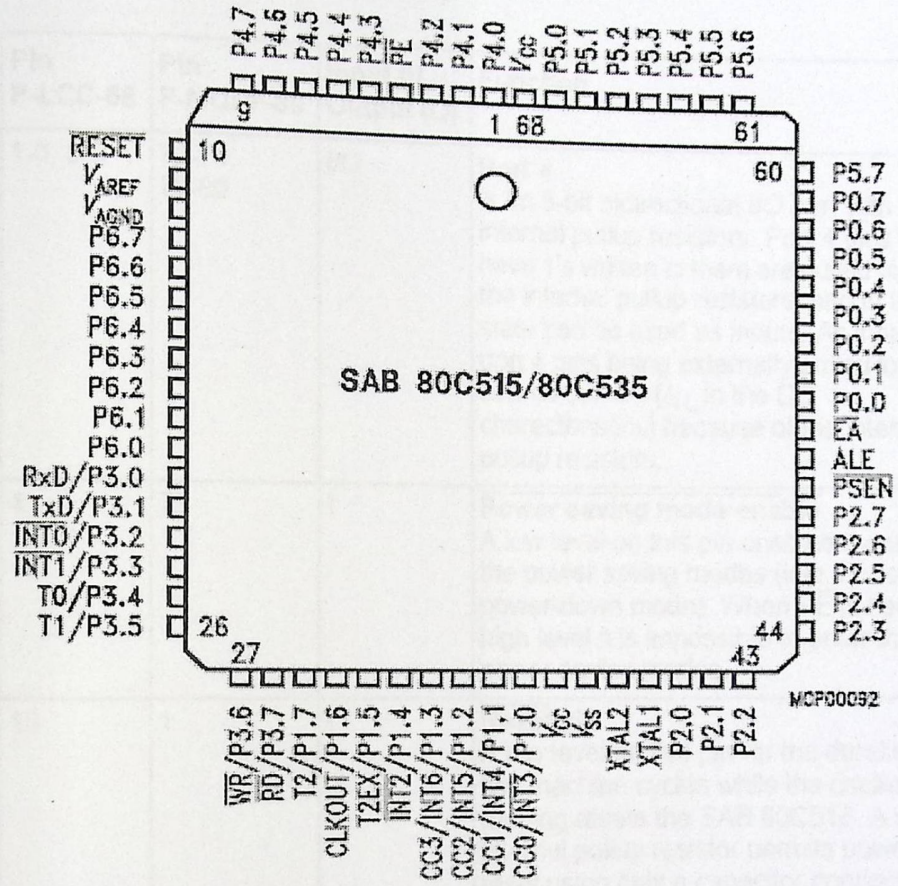
Microcontroller and data sheets

A. Microcontroller

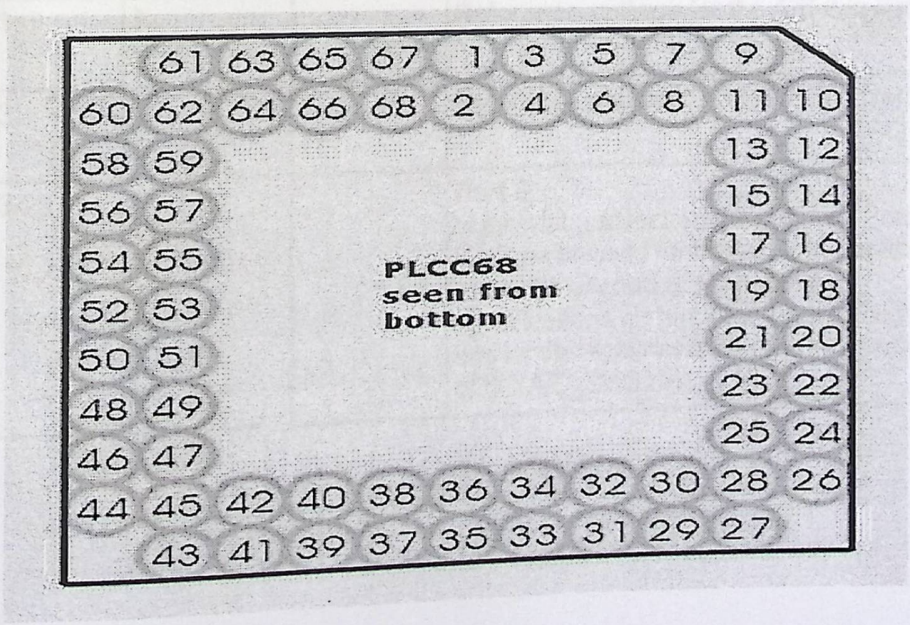
- Detailed block diagram



- PLCC 68 chip pin configuration



- PLCC 68Socket configuration



- Pin Definitions and Functions

Symbol	Pin P-LCC-68	Pin P-MQFP-80	Input (I) Output (O)	Function
P4.0-P4.7	1-3, 5-9	72-74, 76-80	I/O	**Port 4** is an 8-bit bidirectional I/O port with internal pullup resistors. Port 4 pins that have 1's written to them are pulled high by the internal pullup resistors, and in that state can be used as inputs. As inputs, port 4 pins being externally pulled low will source current (I_{IL} in the DC characteristics) because of the internal pullup resistors.
$\overline{PE}$	4	75	I	**$\overline{\text{Power saving mode enable}}$** A low level on this pin enables the use of the power saving modes (idle mode and power-down mode). When $\overline{PE}$ is held on high level it is impossible to enter the power saving modes.
$\overline{RESET}$	10	1	I	**$\overline{\text{Reset}}$ pin** A low level on this pin for the duration of two machine cycles while the oscillator is running resets the SAB 80C515. A small internal pullup resistor permits power-on reset using only a capacitor connected to V_{SS}.
V_{AREF}	11	3		**Reference voltage** for the A/D converter
V_{AGND}	12	4		**Reference ground** for the A/D converter
P6.7-P6.0	13-20	5-12		**Port 6** is an 8-bit undirectional input port. Port pins can be used for digital input if voltage levels simultaneously meet the specifications for high/low input voltages and for the eight multiplexed analog inputs of the A/D converter.

Symbol	Pin P-LCC-68	Pin P-MQFP-80	Input (I) Output (O)	Function
P3.0-P3.7	21-28	15-22	I/O	**Port 3** is an 8-bit bidirectional I/O port with internal pullup resistors. Port 3 pins that have 1's written to them are pulled high by the internal pullup resistors, and in that state can be used as inputs. As inputs, port 3 pins being externally pulled low will source current (I_{IL}, in the DC characteristics) because of the internal pullup resistors. Port 3 also contains the interrupt, timer, serial port and external memory strobe pins that are used by various options. The output latch corresponding to a secondary function must be programmed to a one (1) for that function to operate. The secondary functions are assigned to the pins of port 3, as follows: – RxD (P3.0): serial port's receiver data input (asynchronous) or data input/ output (synchronous) – TxD (P3.1): serial port's transmitter data output (asynchronous) or clock output (synchronous) – $\overline{INT0}$ (P3.2): interrupt 0 input/timer 0 gate control input – $\overline{INT1}$ (P3.3): interrupt 1 input/timer 1 gate control input – T0 (P3.4): counter 0 input – T1 (P3.5): counter 1 input – $\overline{WR}$ (P3.6): the write control signal latches the data byte from port 0 into the external data memory – $\overline{RD}$ (P3.7): the read control signal enables the external data memory to port 0

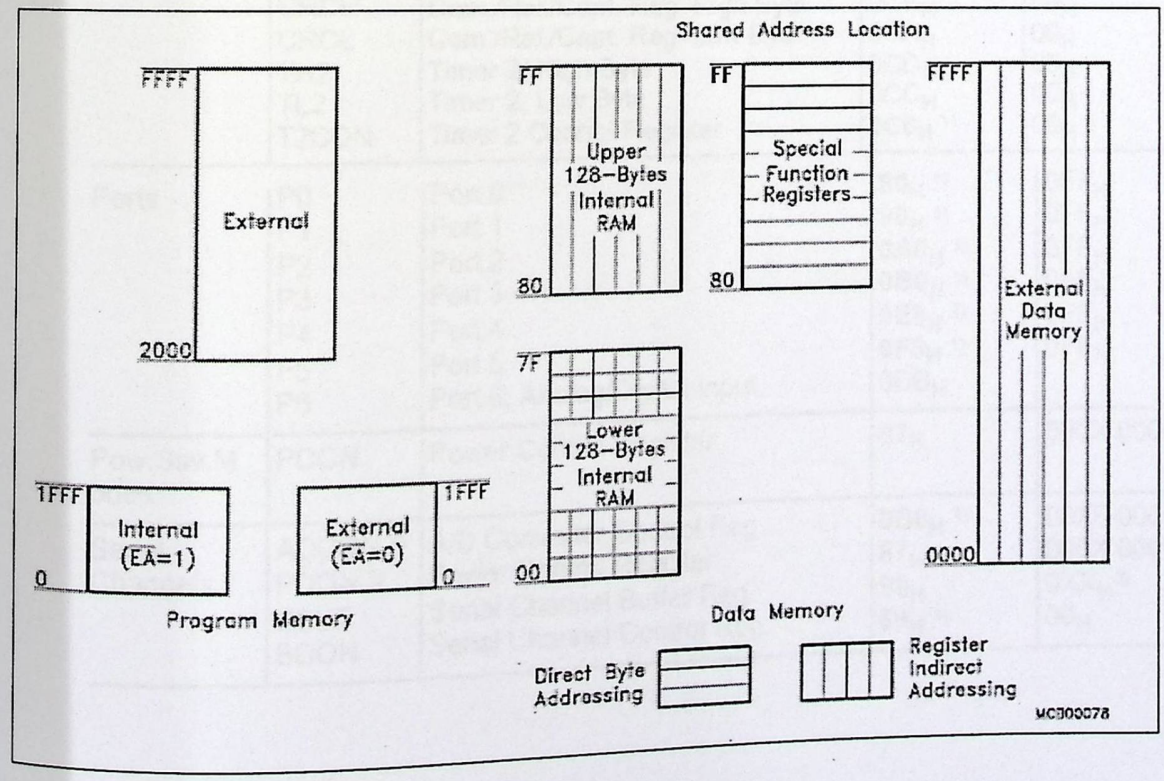
Symbol	Pin P-LCC-68	Pin P-MQFP-80	Input (I) Output (O)	Function
P1.7-P1.0	29-36	24-31	I/O	**Port 1** is an 8-bit bidirectional I/O port with internal pullup resistors. Port 1 pins that have 1's written to them are pulled high by the internal pullup resistors, and in that state can be used as inputs. As inputs, port 1 pins being externally pulled low will source current (I_{IL} in the DC characteristics) because of the internal pullup resistors. The port is used for the low-order address byte during program verification. Port 1 also contains the interrupt, timer, clock, capture and compare pins that are used by various options. The output latch corresponding to a secondary function must be programmed to a one (1) for that function to operate (except when used for the compare functions). The secondary functions are assigned to the port 1 pins as follows: – $\overline{\text{INT3}}$/CC0 (P1.0): interrupt 3 input/ compare 0 output/capture 0 input – INT4/CC1 (P1.1): interrupt 4 input/ compare 1 output/capture 1 input – INT5/CC2 (P1.2): interrupt 5 input/ compare 2 output/capture 2 input – INT6/CC3 (P1.3): interrupt 6 input/ compare 3 output/capture 3 input – $\overline{\text{INT2}}$ (P1.4): interrupt 2 input – T2EX (P1.5): timer 2 external reload trigger input – CLKOUT (P1.6): system clock output – T2 (P1.7): counter 2 input

Symbol	Pin P-LCC-68	Pin P-MQFP-80	Input (I) Output (O)	Function
XTAL2 XTAL1	39 40	36 37		**XTAL2** Input to the inverting oscillator amplifier and input to the internal clock generator circuits. **XTAL1** Output of the inverting oscillator amplifier. To drive the device from an external clock source, XTAL2 should be driven, while XTAL1 is left unconnected. There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is divided down by a divide-by-two flip-flop. Minimum and maximum high and low times and rise/fall times specified in the AC characteristics must be observed.
P2.0-P2.7	41-48	38-45	I/O	**Port 2** is an 8-bit bidirectional I/O port with internal pullup resistors. Port 2 pins that have 1's written to them are pulled high by the internal pullup resistors, and in that state can be used as inputs. As inputs, port 2 pins being externally pulled low will source current (I_{IL}, in the DC characteristics) because of the internal pullup resistors. Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that use 16-bit addresses (MOVX@DPTR). In this application it uses strong internal pullup resistors when issuing 1's. During accesses to external data memory that use 8-bit addresses (MOVX@Ri), port 2 issues the contents of the P2 special function register.

Symbol	Pin P-LCC-68	Pin P-MQFP-80	Input (I) Output (O)	Function
$\overline{\text{PSEN}}$	49	47	O	The **Program store enable** output is a control signal that enables the external program memory to the bus during external fetch operations. It is activated every six oscillator periods, except during external data memory accesses. The signal remains high during internal program execution.
ALE	50	48	O	The **Address latch enable** output is used for latching the address into external memory during normal operation. It is activated every six oscillator periods, except during an external data memory access.
$\overline{\text{EA}}$	51	49	I	**External access enable** When held high, the SAB 80C515 executes instructions from the internal ROM as long as the PC is less than 8192. When held low, the SAB 80C515 fetches all instructions from external program memory. For the SAB 80C535 this pin must be tied low.
P0.0-P0.7	52-59	52-59	I/O	**Port 0** is an 8-bit open-drain bidirectional I/O port. Port 0 pins that have 1's written to them float, and in that state can be used as high-impedance inputs. Port 0 is also the multiplexed low-order address and data bus during accesses to external program and data memory. In this application it uses strong internal pullup resistors when issuing 1's. Port 0 also outputs the code bytes during program verification in the SAB 80C515. External pullup resistors are required during program verification.

Symbol	Pin P-LCC-68	Pin P-MQFP-80	Input (I) Output (O)	Function
P5.7-P5.0	60-67	60-67	I/O	**Port 5** is an 8-bit bidirectional I/O port with internal pullup resistors. Port 5 pins that have 1's written to them are pulled high by the internal pullup resistors, and in that state can be used as inputs. As inputs, port 5 pins being externally pulled low will source current (I_{IL} in the DC characteristics) because of the internal pullup resistors.
V_{CC}	37	33	–	**Supply voltage** during normal, idle, and power-down operation. Internally connected to pin 68.
V_{SS}	38	34	–	**Ground** (0 V)
V_{CC}	68	69	–	**Supply voltage** during normal, idle, and power-down operation. Internally connected to pin 37.
N. C.	–	2, 13, 14, 23, 32, 35, 46, 50, 51, 68, 70, 71	–	**Not connected** These pins of the P-MQFP-80 package must not be connected

- Memory organization



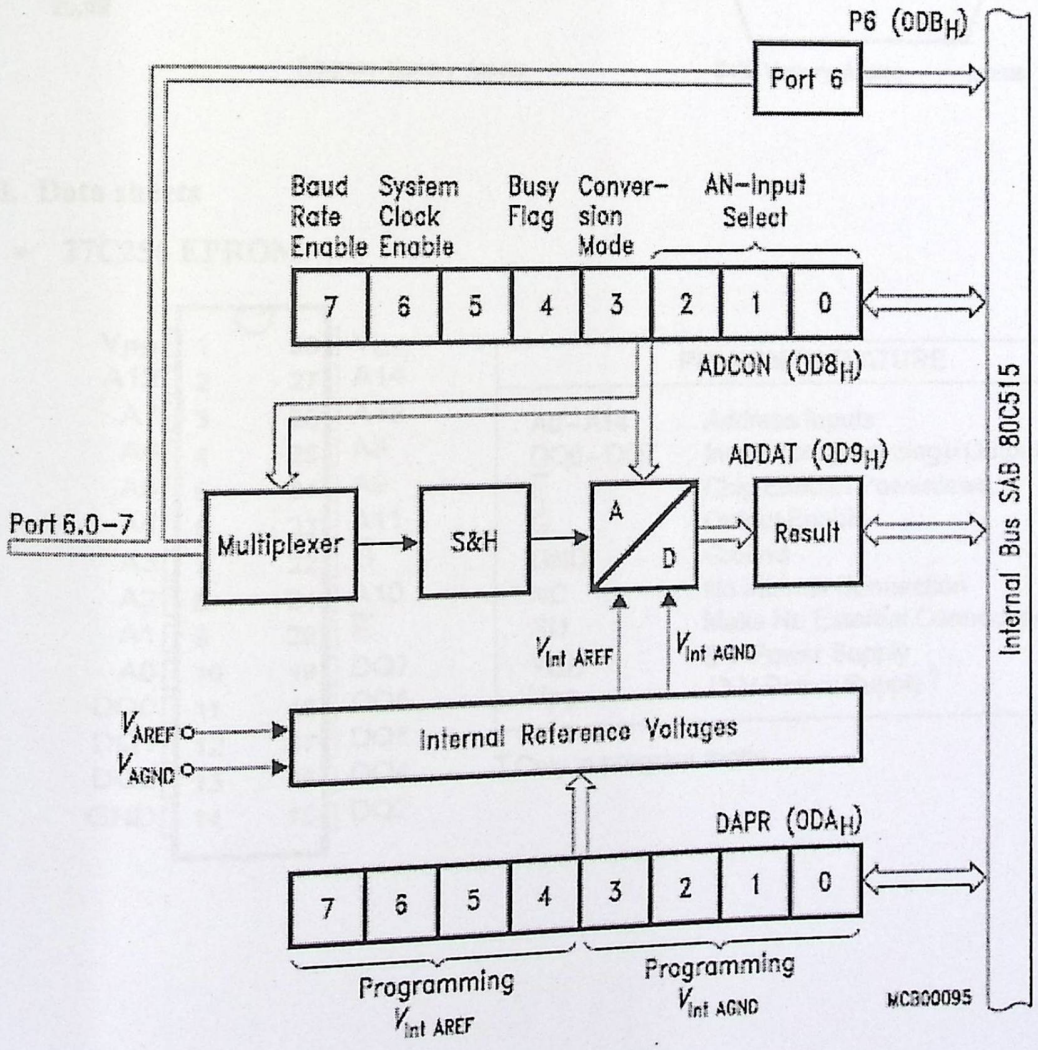
- Special Function Registers

Block	Symbol	Name	Address	Contents after Reset
CPU	ACC	Accumulator	**$0E0_H$** [1]	00_H
	B	B-Register	**$0F0_H$** [1]	00_H
	DPH	Data Pointer, High Byte	83_H	00_H
	DPL	Data Pointer, Low Byte	82_H	00_H
	PSW	Program Status Word Register	**$0D0_H$** [1]	00_H
	SP	Stack Pointer	81_H	07_H
A/D-Converter	ADCON	A/D Converter Control Register	**$0D8_H$** [1]	$00X0\ 0000_B$ [2]
	ADDAT	A/D Converter Data Register	$0D9_H$	00_H
	DAPR	D/A Converter Program Register	$0DA_H$	00_H
Interrupt System	EN0	Interrupt Enable Register 0	**$0A8_H$** [1]	00_H
	IEN1	Interrupt Enable Register 1	**$0B8_H$** [1]	00_H
	IP0	Interrupt Priority Register 0	$0A9_H$	00_H
	IP1	Interrupt Priority Register 1	$0B9_H$	$X000\ 0000_B$ [2]
	IRCON	Interrupt Request Control Register	**$0C0_H$** [1]	$XX00\ 0000_B$ [3]
	TCON [2]	Timer Control Register	**88_H** [1]	00_H
	T2CON [2]	Timer 2 Control Register	**$0C8_H$** [1]	00_H
Compare/ Capture-Unit (CCU)	CCEN	Comp./Capture Enable Reg.	$0C1_H$	00_H
	CCH1	Comp./Capture Reg. 1, High Byte	$0C3_H$	00_H
	CCH2	Comp./Capture Reg. 2, High Byte	$0C5_H$	00_H
	CCH3	Comp./Capture Reg. 3, High Byte	$0C7_H$	00_H
	CCL1	Comp./Capture Reg. 1, Low Byte	$0C2_H$	00_H
	CCL2	Comp./Capture Reg. 2, Low Byte	$0C4_H$	00_H
	CCL3	Comp./Capture Reg. 3, Low Byte	$0C6_H$	00_H
	CRCH	Com./Rel./Capt. Reg. High Byte	$0CB_H$	00_H
	CRCL	Com./Rel./Capt. Reg. Low Byte	$0CA_H$	00_H
	TH2	Timer 2, High Byte	$0CD_H$	00_H
	TL2	Timer 2, Low Byte	$0CC_H$	00_H
	T2CON	Timer 2 Control Register	**$0C8_H$** [1]	00_H
Ports	P0	Port 0	**80_H** [1]	$0FF_H$
	P1	Port 1	**90_H** [1]	$0FF_H$
	P2	Port 2	**$0A0_H$** [1]	$0FF_H$
	P3	Port 3	**$0B0_H$** [1]	$0FF_H$
	P4	Port 4	**$0E8_H$** [1]	$0FF_H$
	P5	Port 5	**$0F8_H$** [1]	$0FF_H$
	P6	Port 6, Analog/Digital Input	$0DB_H$	
Pow.Sav.Modes	PCON	Power Control Register	87_H	$000X\ 0000_B$ [2]
Serial Channels	ADCON [2]	A/D Converter Control Reg.	**$0D8_H$** [1]	$00X0\ 0000_B$ [2]
	PCON [2]	Power Control Register	87_H	$000X\ 0000_B$ [2]
	SBUF	Serial Channel Buffer Reg.	99_H	$0XX_H$ [3]
	SCON	Serial Channel Control Reg.	**98_H** [1]	00_H

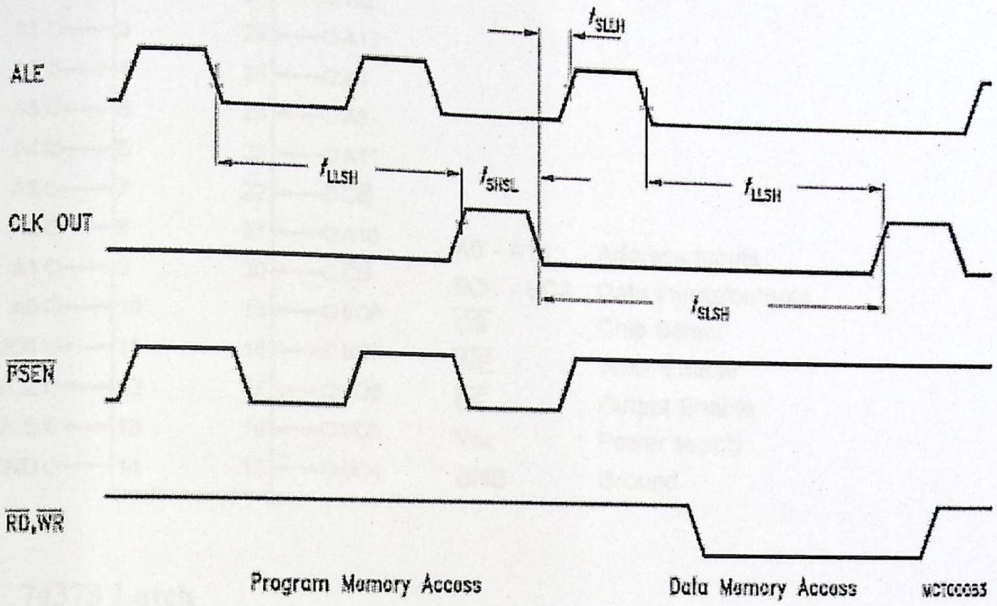
Timer 0/ Timer 1	TCON TH0 TH1 TL0 TL1 TMOD	Timer Control Register Timer 0, High Byte Timer 1, High Byte Timer 0, Low Byte Timer 1, Low Byte Timer Mode Register	88_H 1) $8C_H$ $8D_H$ $8A_H$ $8B_H$ 89_H	00_H 00_H 00_H 00_H 00_H 00_H
Watchdog	IEN0 2) IEN1 2) IP0 2) IP1 2)	Interrupt Enable Register 0 Interrupt Enable Register 1 Interrupt Priority Register 0 Interrupt Priority Register 1	$0A8_H$ 1) $0B8_H$ 1) $0A9_H$ $0B9_H$	00_H 00_H $X000\ 0000_B$ 2) $XX00\ 0000_B$ 3)

1) Bit-addressable special function registers
2) This special function register is listed repeatedly since some bits of it also belong to other functional blocks.
3) X means that the value is indeterminate and the location is reserved

- Block Diagram of the A/D Converter

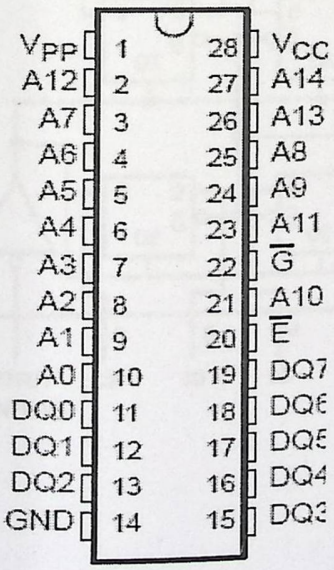


- **System timing**



B. Data sheets

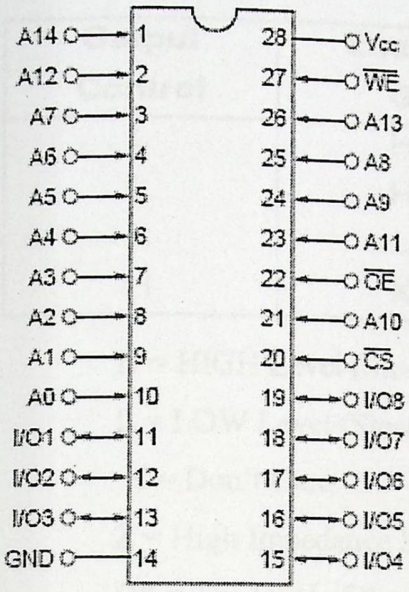
- **27C256 EPROM.**



PIN NOMENCLATURE	
A0–A14	Address Inputs
DQ0–DQ7	Inputs (programming)/Outputs
$\overline{E}$	Chip Enable/Powerdown
$\overline{G}$	Output Enable
GND	Ground
NC	No Internal Connection
NU	Make No External Connection
VCC	5-V Power Supply
VPP	13-V Power Supply†

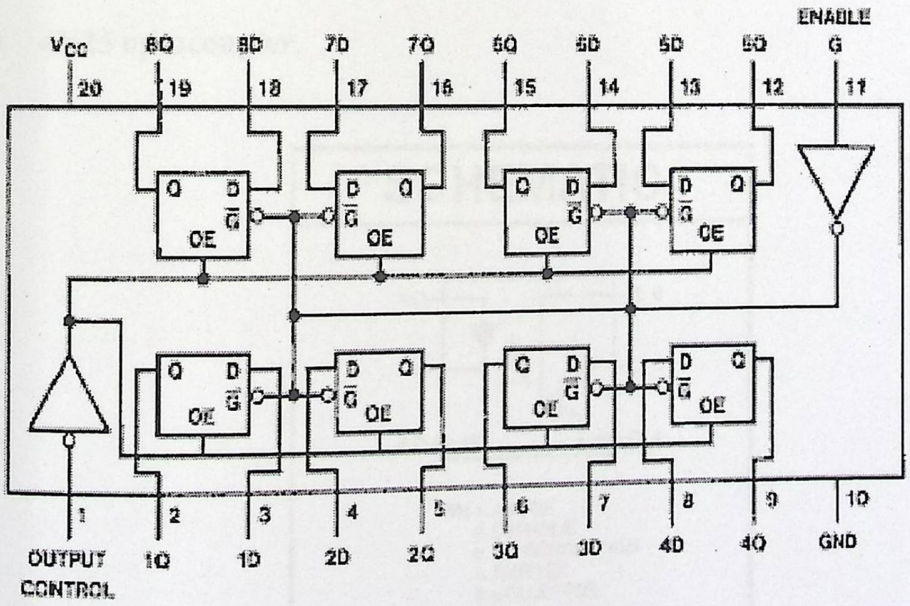
† Only in program mode

- **62C256 SRAM.**



A0 - A14 : Address inputs
I/O1 - I/O8 : Data inputs/outputs
$\overline{CS}$: Chip Select
$\overline{WE}$: Write Enable
$\overline{OE}$: Output Enable
Vcc : Power supply
GND : Ground

- **74373 Latch**



Function Table:

Output Control	Enable G	D	Output
L	H	H	H
L	H	L	L
L	L	X	Q_0
H	X	X	Z

H = HIGH Level (Steady State)

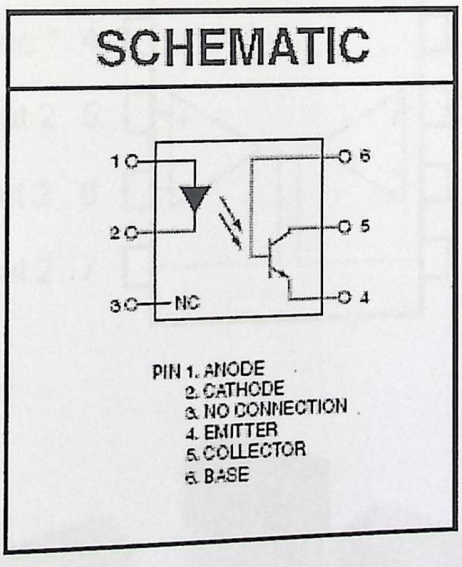
L = LOW Level (Steady State)

X = Don't Care

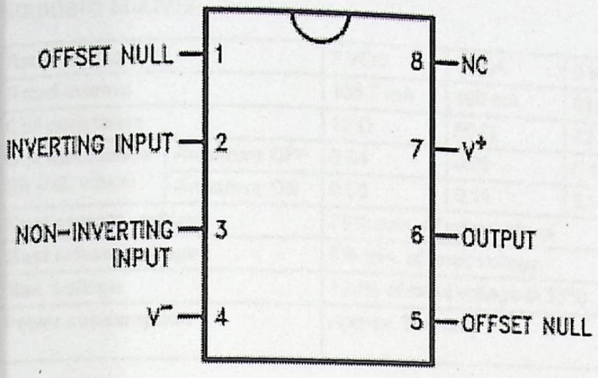
Z = High Impedance State

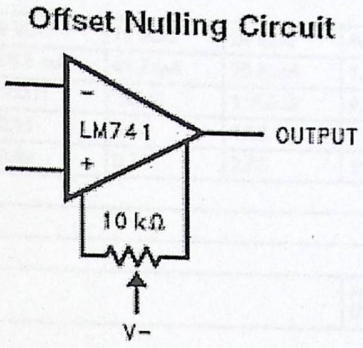
Q0 = The level of the output before steady-state input conditions were established.

- **4N25 optocoupler.**

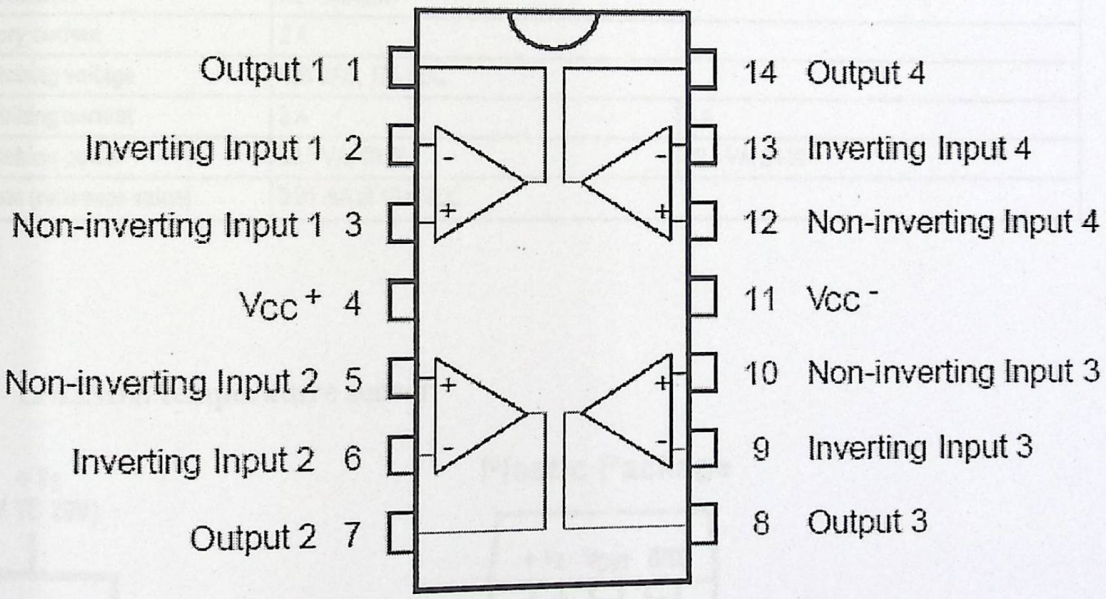


- **LM741 op-Amplifier.**

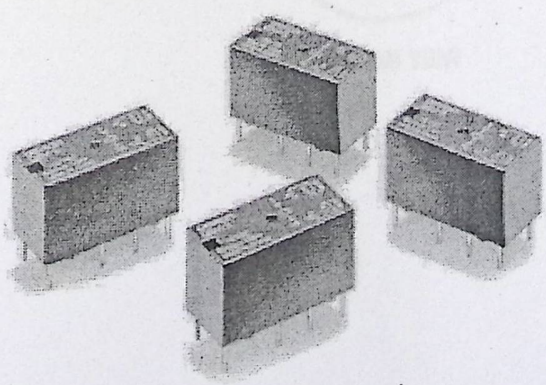




- **TL074CN Quad op-Amplifier.**



- **Relay**



■ Coil Ratings

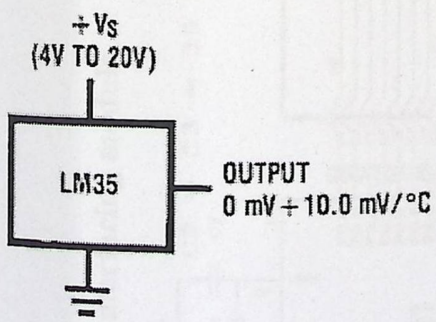
Standard Models

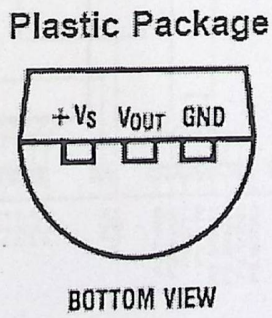
Rated voltage		3 VDC	5 VDC	6 VDC	9 VDC	12 VDC	24 VDC	48 VDC
Rated current		166.7 mA	100 mA	83.3 mA	55.6 mA	41.7 mA	20.8 mA	12 mA
Coil resistance		18 Ω	50 Ω	72 Ω	162 Ω	288 Ω	1,152 Ω	4,000 Ω
Coil inductance (H) (ref. value)	Armature OFF	0.04	0.09	0.16	0.31	0.47	1.99	7.23
	Armature ON	0.05	0.11	0.19	0.49	0.74	2.63	10.00
Must operate voltage		75% max. of rated voltage						
Must release voltage		5% min. of rated voltage						
Max. voltage		120% of rated voltage at 23°C						
Power consumption		Approx. 500 mW						Approx. 580 mW

■ Contact Ratings

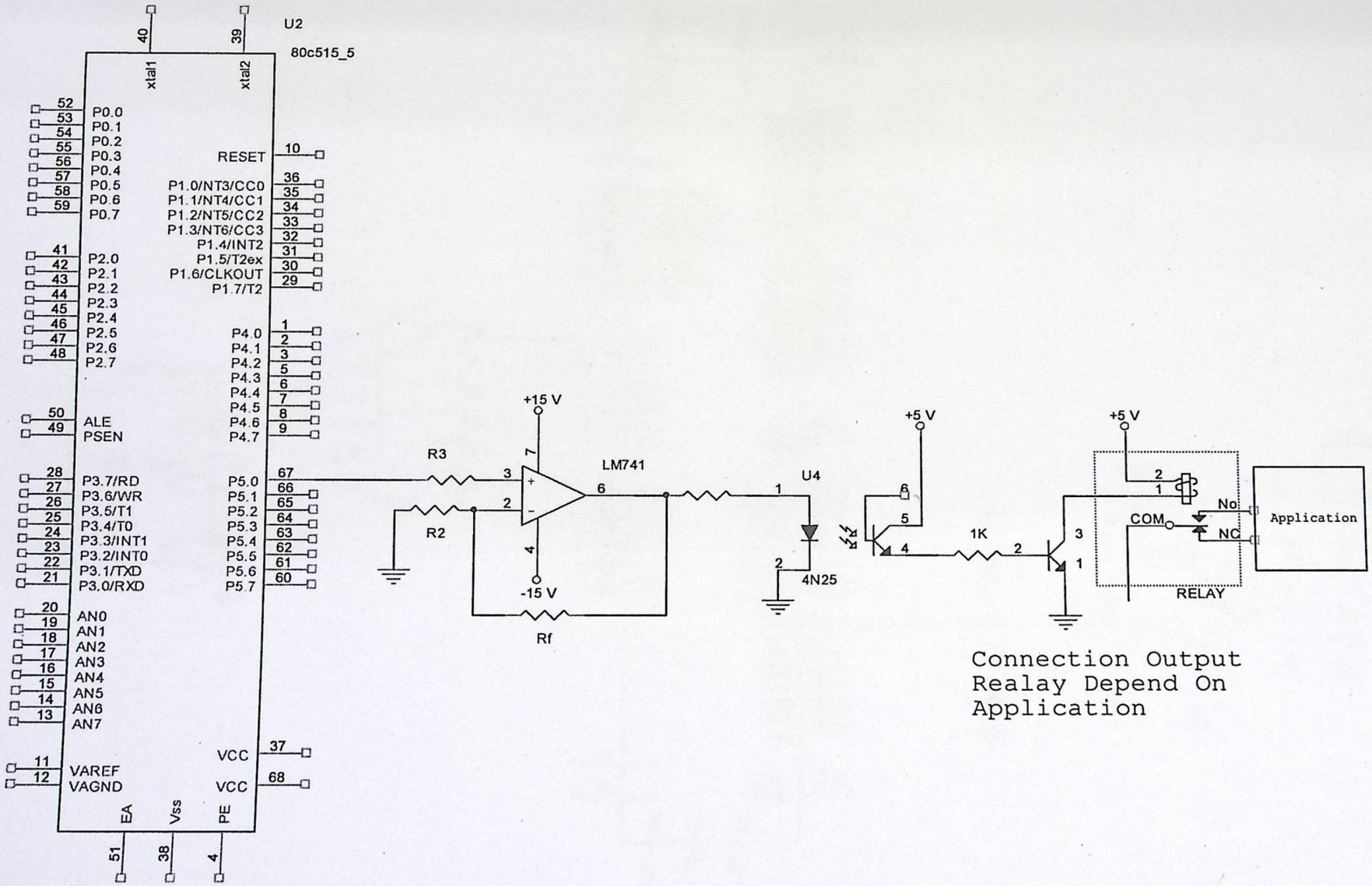
Item	Standard models	High sensitivity models
Load	Resistive load ($\cos\phi = 1$)	
Rated load	0.5 A at 125 VAC; 2 A at 30 VDC	0.5 A at 125 VAC; 1 A at 24 VDC
Contact material	Ag + Au-clad	
Rated carry current	2 A	
Max. switching voltage	125 VAC, 125 VDC	
Max. switching current	2 A	1 A
Max. switching power	62.5 VA, 60 W	62.5 VA, 24 W
Failure rate (reference value)	0.01 mA at 10 mVDC	

- **LM35DZ temperature sensor**

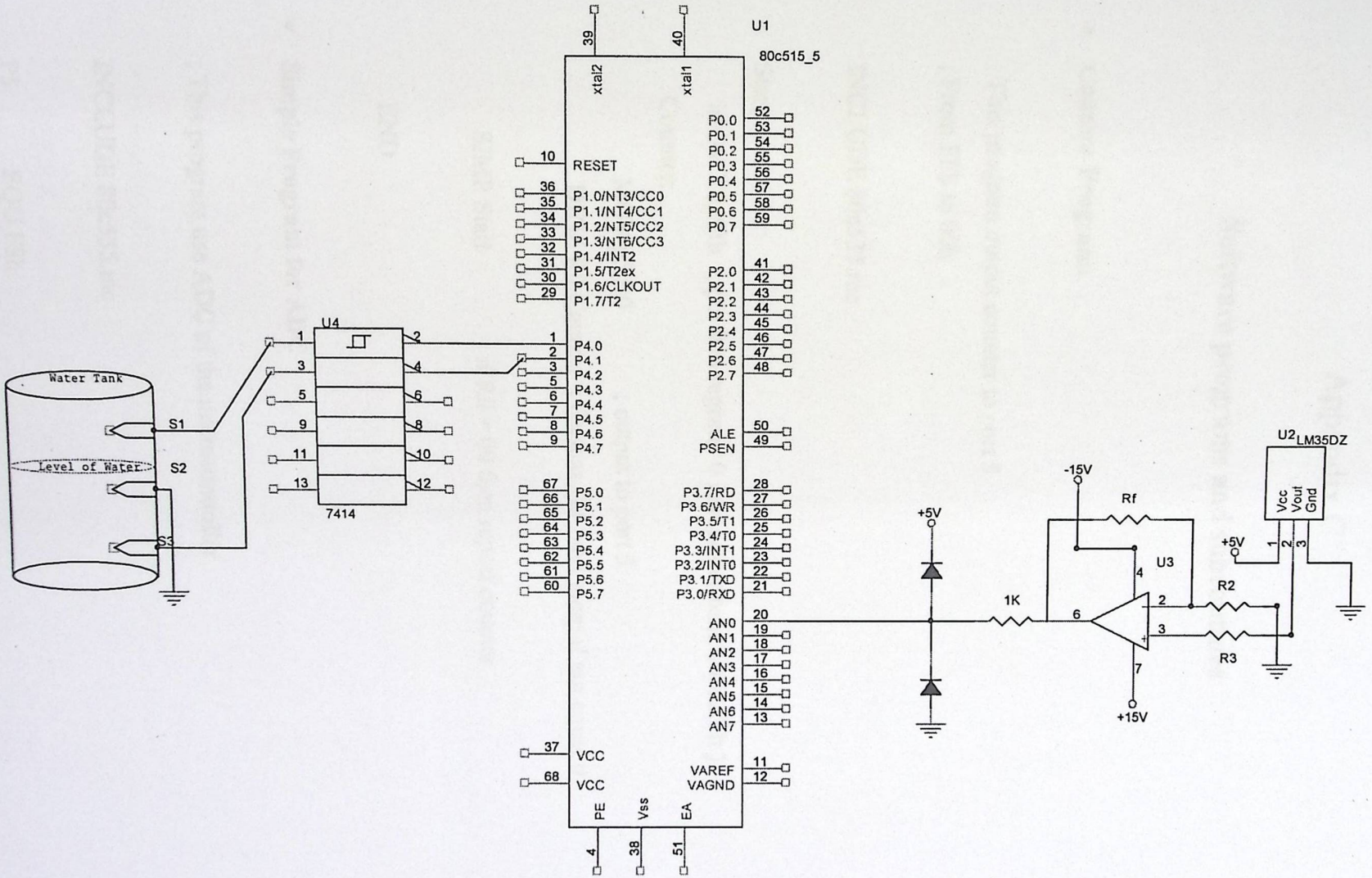




B. Switching circuit module



C. Feedback circuit module



Appendix C

Software programs and subroutines

- **Counter Program:**

```
; This program output counter to port 5
; From FFh to 00h

INCLUDE 80c535.mc

Start:
    MOV R0,#ffh   ; put 0 to regester 0 in regester bank 0 ( defult )
    Counter:
        MOV p5,R0          ; output to port 5
        DJNZ R0,Counter  ; decreas R0 , and jump if not equal 0

    SJMP Start          ; if R0 = 00 then repeat counter

    END
```

- **Simple Program for ADC**

```
; This program use ADC of the microcontroller

INCLUDE 80c535.mc

P5          EQU F8h
```

```
ADCON EQU D8h
DAPR    EQU D9h
ADDAT   EQU Dah
bsy     BIT  DCh

Start:
    MOV ADCON,#F8h   ; select channel 0 of ADC, and use Continues mode
                     ;conversion
    MOV DAPR,#00h    ; start conversion

    delay:

     DJNZ R0,delay

    conversion:

          JB bsy,conversion

     MOV A,ADDAT        ; at end conversion transfer the result of conversion
                        ; to ACC
     MOV P5,A           ; output the result at port 5

     END
```

- **Water level**

```
INCLUDE 80c535.mc
```

```
Start:
        SETB p4.0    ; set port 4 pin 0
        SETB p4.1      ‘set port 4 pin 0

readlevel:

        MOV A,p4     ;read level of water

        ANL A,#03h   ;mask all bit except first two bit

        CJNE A,#00h,notempty        ;compare ACC with low level of water

        MOV p5,#00000001b ;if the water in low level, then operate motor

        SJMP readlevel

        notempty:

                CJNE A,#03,notfull   ; compare ACC with high level of water

                MOV p5,#00000000b ; turn off motor

                SJMP readlevel

        notfull:
                SJMP readleve

                END
```

- **Combinational logic program**

```
;simple boolean function program
;y0 = (x0 AND x1)OR x2
;y1 =  (x0 OR x1)AND (x0 OR x1)

INCLUDE 80c535.mc

;input ports
x0 EQU p1.0
x1 EQU p1.1
x2 EQU p1.2

;output ports
y0 EQU p1.3
y1 EQU p1.4

temp EQU 40h    ; used to store immediate result

start:
      MOV p1,0ffh  ;prepare bits 0 to 2 as inputs
      ;now compute y0
      MOV C,x0
      ANL C,x1
      ORL C,x2

      MOV y0,C

      ;now compute y1
      MOV C,x0
```

```
        ORL C,x1
        MOV temp,C
        MOV C,x0
        ORL C,x2
        ANL C,temp

        MOV y1,C

        SJMP start

    END
```

- **Sequential logic program**

```
;boolean function
;y0 = (x0 AND x1)'
;y1 = (x1 AND y0)'

    Include 80c535.mc
    ;---port definition

    ;input ports
    X0 equ P1.0
    X1 ego P1.1
    ;output ports
    y0 equ P1.3
    y1 equ P1.4

    ;---program starts here
    start
```

```
mnov P1, OFFh ; prepare bits 0 to 2 as input
;---now compute 10
mov C, X0 ; place X0 in carry flag
anl C, y1 ; and X0 and y1, result in carry flag
cpl C ; complement the result

mnov y0, C
;---now compute y1
mov C, Xl ; place X0 in carry flag
anl C, y0 ; and Xl and y0, result in carry flag
cpl c     ;complement the result

mov y1, C

Sjmp Start ;repeat forever...
```