



# MICROCONTROLLER BASED 24V RIG GENERATOR

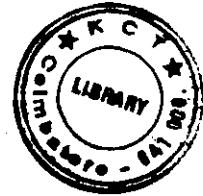


**A Project Report**

*Submitted by*

**S.Gayathri Devi  
S. Kanimoli  
V.Ramkumar  
V.R.Shankar**

**71203105012  
71203105019  
71203105038  
71203105044**



P - 2083

*in partial fulfillment for the award of the degree  
of*

**Bachelor of Engineering  
in  
Electrical & Electronics Engineering**

**DEPARTMENT OF ELECTRICAL & ELECTRONICS  
ENGINEERING**

**KUMARAGURU COLLEGE OF TECHNOLOGY  
COIMBATORE - 641 006**

**ANNA UNIVERSITY: CHENNAI 600 025**

**APRIL - 2007**

# ANNA UNIVERSITY: CHENNAI 600 025

## BONAFIDE CERTIFICATE

Certified that this project report entitled “Microcontroller based 24V Rig Generator” is the bonafide work of

Ms. S.Gayathri Devi	-	Register No. 71203105012
Ms. S.Kanimoli	-	Register No. 71203105019
Mr. V.Ramkumar	-	Register No. 71203105038
Mr. V.R.Shankar	-	Register No. 71203105044

Who carried out the project work under my supervision.



Signature of the Head of the Department

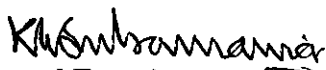
(Prof.K.Regupathy Subramanian)



Signature of the Supervisor

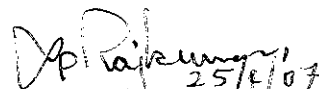
(Mr.P.Thirumoorthi)

Certified that the candidate with University Register No: \_\_\_\_\_ was examined in project viva voce Examination held on 25.04.2007



Internal Examiner

25/4/07



External Examiner

**DEPARTMENT OF ELECTRICAL & ELECTRONICS  
ENGINEERING**

**KUMARAGURU COLLEGE OF TECHNOLOGY**

**COIMBATORE 641 006**



pricol limited

PL/TRG/PROJ/2007  
09-04-2007

**PROJECT COMPLETION CERTIFICATE**

We are pleased to issue this certificate in the process of operationalising our "Industry- Institute Interaction Synergy" drive.

Name : S. Gayathri devi [Reg# 71203105012]  
S. Kanimoli [Reg# 71203105019]  
V.Ramkumar [Reg# 71203105038]  
V.R. Shankar [Reg# 71203105044]

Institution : Kumaraguru College of Technology,  
Coimbatore-641 006

Qualification : B E (Electrical & Electronics)

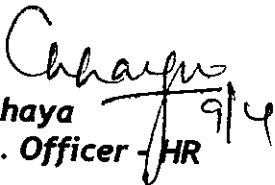
Project Title : "Micro Controller Based 24V Rig Generator"

Project Duration : Dec'06 to March'07

Department : P D E (Electronics)

Performance/ Conduct : Good

**For Pricol Limited,**

  
Chhaya  
SR. Officer - HR

## COMPANY PROFILE

Premier Instruments and Control Limited was established in 1974 for manufacturing automotive products. Pricol has grown into a market leader with its high quality and reliable products and services. It enters into Electronics Product Group in 1980. It diversified into Machine tools, Defence Products and Industrial gauges during 1981, 1988 and 1989 respectively. Pricol started its second Plant at Gurgaon; to cater to the needs of the cluster of customers. Pricol established its third and fourth plants at Coimbatore in 1999 with the view to rationalize the manufacturing operation. Pricol has a corporate Office functioning at Coimbatore city, to give better service by way of easy access and communication to customers, Vendors and Shareholders.

In addition to this, Pricol has offices at Chennai, Delhi, Mumbai and representatives are placed at important cities within India for interaction with customers. Pricol has dedicated dealers/representatives in countries like USA, UK, Germany, Turkey and other places abroad for better services to foreign customers.

Pricol entered into its technological collaboration with Nippon-Seiki in 1985 for two-wheelers instruments. In 1992, Pricol entered into a technical tie-up with Denso for four-wheelers instruments. Pricol now exports its products to countries like USA, UK, South Africa, México, West Germany, Brazil, Argentina, Italy, Belgium, China, Korea, etc.

The Corporate Vision is to strive for excellence in all they do through socially and environmentally acceptable means. They will have market leadership through customer delight. They will be a responsible corporate citizen and share the benefits with society. They manufacture Dashboard Instruments, Cables, Oil Pumps for two wheelers and stationary engineers. It also manufactures Electronic Counters and Controllers for Textile Machinery and Instrument Panel for Defence Combat Vehicles and Special Purpose Machines & General Purpose Machines etc.

Pricol has been accredited with ISO 9001 certification by BYQI for its Quality Systems in 1993 and is going through successful audits to sustain its Certification; the meticulous march towards QS 9000 is on. The Organization successfully re-engineered its key business processes in 1995 to improve the efficiency and effectiveness of its functioning and services. Its esteemed customers are Hero Honda Motors Ltd, Maruthi-Udyog Ltd, Bajaj Auto Ltd, Mahindra & Mahindra, etc.

Being a learning organization, Pricol Benchmarks itself with the leading corporate to keep abreast of the technology and management and to become a proactive and progressive company in the true spirit of the times. Pricol's TQM motto is, "Total Quality Management is not Management of Quality, but Quality of Management".

## **KEY POLICIES**

### **CORPORATE CORE VALUES:**

- Respect and concern for individuals.
- Customers, Employees and Suppliers-partners in the value chain
- Encourage innovation and improvement-accept noble failures.
- Continuous learning

### **QUALITY POLICY:**

Pricol will provide value and satisfaction to customers on products and services. This is achieved through systematic training and motivation of the employees.

### **TPM POLICY:**

Establish an excellent Total Productive Manufacturing system with total involvement of all employees to achieve effective utilization of all resources

## ABSTRACT

Electronic components in automobiles are prone to surge voltages at the time of starting. Speedometers are to be protected against such voltages. At the quality control division of the company, they are subjected to various tests viz., electrical tests, temperature tests (hot and cold chamber tests), pressure tests and so on. Rig generators are used for power supply fluctuation test under electrical test. A rig generator is a device that tests the speedometers for surge voltages and hence to assure the reliability of the speedometers. A rig generator is an emulated model of the battery which would generate surges as a battery would, during the start of the vehicle. There are various testing standards specified either by the manufacturer or the clients. Every client may ask for different testing standard or combination of many testing standards. Our project is to design a rig generator which would generate surges as a 24V battery would, during the start of the vehicle and can be used for any test specifications i.e. a programmable one in which the user can enter the values of time intervals and magnitude of the voltage. The basic wave shape is generated using a PIC controller employed with user interface for setting wave parameters like rise time, fall time, pulse width & total period and a high resolution digital to analog converter (DAC). The output of DAC is then fed to the output driver circuit to produce the necessary amplitude. The firmware is developed using MPLAB IDE v7.3 with MICROCHIP C18 language tool suite. MPLAB ICD2 is then used to download the code into the PIC microcontroller. The output driver circuit constituting of a high power handling audio amplifier provides the necessary voltage level to the wave shape. The generated waveform is applied to the sample (speedometer) using the test circuit diagram as specified by the testing standards and checked for any malfunction of the sample.

## ACKNOWLEDGEMENT

. We are highly privileged to thank **Dr.Joeseph V Thanikal**, Principal, Kumaraguru College of Technology and the management of Kumaraguru College of Technology for allowing us to do this project. We express our heart felt gratitude and thanks to the Dean / HOD of Electrical & Electronics Engineering, **Prof. K.Regupathy Subramanian**, for encouraging us and for being with us right from beginning of the project and guiding us at every step. We wish to place on record our deep sense of gratitude and profound thanks to our guide **Mr.P.Thirumoorthi**,Sr.Lecturer,Electrical and Electronics Engineering Department, for his valuable guidance, constant encouragement, continuous support and co-operation rendered throughout the project.

We thank our sponsors **M/s PRICOL Ltd**, for sponsoring the required materials and helping us to complete the project. We sincerely thank our guide in the company, **Mr.V.Ramamoorthy**, who has been our guiding light, spending time to clarify our doubts in spite of his busy schedule. We thank all the staff in the company for their valuable assistance. We are also thankful to our teaching and non-teaching staffs of Electrical and Electronics Engineering department, for their kind help and encouragement.

Last but not least, we extend our sincere thanks to all our **parents and friends** who have contributed their ideas and encouraged us for completing the project.

DEDICATED  
TO OUR  
BELOVED PARENTS



# CONTENTS

<b>Title</b>	<b>Page No.</b>
Bonafide Certificate	ii
Certificate of the Company	iii
Company profile	iv
Abstract	vi
Acknowledgement	vii
Contents	ix
List of tables	xi
List of figures	xii
List of symbols and abbreviations	xiii

## CHAPTER 1 INTRODUCTION

1.1	Objective	2
1.2	Introduction to electrical noises	2
1.3	Supply line Transients	4
1.4	Automobile Electrical System	5
1.5	Scope of the project	6
1.6	Organization of the report	6

## CHAPTER 2 PRODUCT RELIABILITY TEST DETAILS

2.1	Introduction	8
2.2	Types of test	8
2.3	Power supply fluctuation test	10

## **CHAPTER 3 ANALYSIS AND DESIGN OF RIG GENERATOR**

3.1	Requirements Gathered	13
3.2	Existing System	13
3.3	Design overview	13
3.4	Block Diagram of hardware set-up	14

## **CHAPTER 4 FEATURES OF MICROCONTROLLER PIC18F452**

4.1	Introduction	19
4.2	Features of PIC used	20

## **CHAPTER 5 HARDWARE DESCRIPTION OF RIG GENERATOR**

5.1	Introduction	30
5.2	Circuit description	30
5.3	Digital part	30
5.4	Analog part	34
5.5	Power supply	36
5.6	Working of the circuit	37

## **CHAPTER 6 FIRMWARE DEVELOPMENT FOR RIG GENERATOR**

6.1	Algorithm of Rig generation	40
6.2	Flowchart of Rig generation	41
6.3	Flowchart for LCD and keyboard function	44
6.4	MPLAB IDE	47
6.5	MPLAB C18	48
6.6	Device Programming	48

## **CHAPTER 7 RESULTS AND DISCUSSIONS**

7.1	Microcontroller based 24V Rig generator	52
7.2	Output waveform	53
7.3	Output waveform-Total time period	53
7.4	Output waveform- Rise time	54
7.5	Output waveform- fall time	55
7.6	Output waveform-amplitude	56

## **CHAPTER 8 CONCLUSIONS AND RECOMMENDATIONS**

8. 1	Conclusion	58
8. 2	future scope of the project	58

## **APPENDIX A- PIN CONFIGURATIONS**

60

## **APPENDIX B- CODING**

64

## **APPENDIX C- PHOTOS**

74

## **REFERENCES**

76

## LIST OF TABLES

<b>Table No</b>	<b>Title</b>	<b>Page No</b>
4.1	Timer0 control register	21
4.2	SPI control register	23
4.3	Synchronous serial port status register	24
4.4	PortA functions	27
4.5	PortC functions	27
4.6	PortD functions	28
6.1	ICD 2 Pin Connections	50

## LIST OF FIGURES

<b>Figure No</b>	<b>Title</b>	<b>Page No</b>
2.1	Test circuit diagram	10
2.2	Voltage waveform	11
3.1	Block Diagram of Hardware Setup	14
5.1	Circuit Diagram-Digital part	32
5.2	Keypad Scanning	32
5.3	Circuit Diagram -Analog part	33
5.4	circuit diagram-power supply	36
6.1	Flowchart of Rig generation	42
6.2	Flowchart for LCD and keyboard function	45
6.3	MPLAB IDE Project Manager	47
6.4	MPLAB ICD 2	49
6.5	ICD 2 Connector Pin Out	49
7.1	Microcontroller based 24V Rig generator	52
7.2	Output waveform	53
7.3	Output waveform- total time period	53
7.4	Output waveform- Rise time	54
7.5	Output waveform- Fall time	55
7.6	Output waveform- Amplitude	56

## LIST OF SYMBOLS & ABBREVIATIONS

EMP	Electromagnetic pulses
RFI	Radio-frequency interference
ESD	Electro static discharge
TCOCON	Timer0 Control Register
SSP	Synchronous Serial Port
SPI	Serial Peripheral Interface
IC	Inter-Integrated Circuit
EEPROM	Electrically Erasable Programmable Read Only Memory
SSPCON	SSP Control Register
SSPSTAT	SSP Status Register
SSPBUF	Serial Receive/Transmit Buffer
SSPSR	SSP Shift Register
SFR	Special Function Register

**CHAPTER 1**  
**INTRODUCTION**

# 1. INTRODUCTION

## 1.1 OBJECTIVE:

Our project is to design a rig generator which would generate surges as a 24V battery would, during the start of the vehicle. The Rig generator accepts input for various time intervals and generates the standard wave shape accordingly. There are various testing standards specified either by the manufacturer or the client. . Every client may ask for different testing standard or combination of many testing standards. This Rig generator has to generate wave shapes for any testing standard for power supply fluctuation test

## 1.2 INTRODUCTION TO ELECTRICAL NOISES:

Several noise conditions, those involving electrostatic discharges, or as found in automotive environments, can do permanent damage to the hardware. Electrostatic discharges of up to even 400v can occur in the 12v power line in automotive environments.

One symptom of electrical noise problems is randomness, both in occurrence of the problem and in what the system does in its failure. But all operational upsets that occur at seemingly random intervals are not necessarily caused by noise in the system. The name given to the electrical noises other than those that are inherent in the circuit components (such as thermal noise) is EMI-Electromagnetic interference. There are different types of EMI of which the predominant ones are listed below.

### 1.2.1 ELECTROMAGNETIC PULSES AND RADIO FREQUENCY INTERFERENCE:

Sparks or arcs produced in a system will radiate electromagnetic pulses (EMP) or radio-frequency interference (RFI). Arcs and sparks occur in automotive ignition systems, electric motors, static discharges, etc. The commutating bars in the electric motors produce the arc, while in switches the same inductive kick phenomenon produces the spark.



## **2.2 ELECTROSTATIC DISCHARGE:**

Electro static discharge (ESD) is the spark that occurs when a person picks up a static charge by walking across a carpet, and then discharges it into a keyboard, or whatever else can be touched. This transient has a very short rise time (in nanoseconds) and the voltage can be as large as 35KV.

## **2.3 GROUND NOISE:**

Currents in ground lines are another source of noise. This noise occurs because of the potential difference that exists between the ground lines. If two ends of a wire are grounded at different locations the current drive may be significant.

## **2.4 RADIATED AND CONDUCTED NOISE:**

Radiated noise is the noise that arrives at the victim circuit in the form of electromagnetic radiation, while extraneous voltages carried in the supply lines cause the conducted noise. The radiated noise can be reduced to a large extent by providing appropriate shielding to the victim circuit.

## **2.5 SIMULATING THE ENVIRONMENT:**

Addressing noise problems after the design of a system has been completed is an expensive proposition. It is cheaper in the long run to invest a little time and money in learning about noise and noise simulation of equipment so that controlled tests can be done before manufacturing the system in the large scale.

Simulating the environment is a two step process, first we have to recognize what the noise environment is that, we have to know what kind of electrical noises are present and which of them are going to cause problems. The next stage is to generate the electrical noise in a controlled manner. With these steps in mind the surge transients that are produced in the automobiles have been identified.

### **3 SUPPLY LINE TRANSIENTS:**

Switching heavy current loads (AC or DC) on or off will cause large transients in power lines. When a load is suddenly switched off large transients occur as the field collapses in the inductance present in the line. This is called an inductive kick and occurs because of contact bounce, when switches are turned on or off.

#### **3.1 LOAD DUMP TRANSIENTS:**

Load dump transients are high voltage, high energy positive transients. The response time of the output voltage of an alternator to load variations is very large because of the large time constant of the excitation winding and the mechanical inertia.

Consequently when the load is reduced instantaneously (by turning off the head lights, cooling fans, etc) the output voltage of the alternator tends to present a positive peak. Under normal conditions this does not cause any problem since the battery has a high ampere-hour rating. But manufacturers impose stringent standards that electronic devices must be protected against load dump transients because it is possible for the connection (fuse) between the battery and the alternator to break. Motor manufactures require that the voltage regulators be able to protect themselves and the load against peak voltages of (60-100)V ,with an equivalent series resistance of 0.1 to 1 ohm.

#### **3.2 FIELD DECAY TRANSIENTS:**

Field decay transients are high energy, high voltage negative transients. If the ignition switch is turned off while current is flowing in inductive loads (electric motors, alternator field coils, etc) a negative voltage transient appears on the supply line. The modulus of the peak value of this transient is of the same value as that of a load dump transient. The system must be capable of withstanding such transients.

## **1.4 AUTOMOBILE ELECTRICAL SYSTEM**

Electrical system of automobile contains four main circuits as

- Generating circuit
- Starting circuit
- Ignition circuit
- Lighting circuit

All the main circuits are connected together and linked to the battery. The starting motor is connected directly to the battery through cables and a switch to provide a low resistance path for the large currents required by the motor.

The generating circuit is connected to one end of the car ammeter. If there is one, so that the meter registers in the 'charge' direction when current is being sent to the battery a wire is connected to the other end of the ammeter or the generator directly to the cable leading to the ungrounded pole of the battery.

The ignition primary circuit, the lighting circuit and all the branch circuits are connected to the same side of the ammeter same as the generator, so that when the generator is operating, they receive current directly from it without going through the ammeter. When the generator is not running, these circuits draw current directly from the battery or through these ammeter which reads in the discharge direction because through it is opposite in directions from the charging current.

The voltage and current output of the generator are regulated by the generator controls in accordance with the current needs the circuits and the state of charge of the battery.

### **1.4.1 GENERATING SYSTEM**

The generating circuit starts at the generator and runs through its controls to the battery, the ammeter and the other devices in the automobile. When generator is sufficiently turned by the engine, it furnishes the electrical energy for all the vehicle's circuits and replenishes the battery to keep it fully charged.

## **1.4.2 GENERATOR CONTROLS**

All motor vehicle generators must be controlled because they operate at a wide range of speeds and if they were uncontrolled would deliver a wide range of voltages. The charging rate of generator must be adjusted to the condition of the battery. The battery must be disconnected from the generator when the battery is not being turned reconnected when it is able to furnish electrical energy

## **1.5 SCOPE OF THE PROJECT:**

The Rig generator designed has fulfilled our objective. The standard wave shape for power supply fluctuation test is generated by getting the time settings from the user. The voltage levels for the wave shape are constant values, which are obtained accurately. This Rig generator is used in Pricol industries (Product reliability assurance lab of the testing division) to test their speedometers and fuel indicators for power supply fluctuation test.

## **1.6 ORGANISATION OF THE REPORT:**

This report consists of 8 chapters organized as follows:

In the first chapter, a general introduction on supply line transients and automobile electrical system are provided. In the second chapter the various tests conducted in the Quality control section of the company are discussed and stress is made on the test taken into account for our project. The fourth chapter deals with features of PIC18F452 used in the project. The fourth chapter deals with the description of hardware design which is followed by the algorithms and flowcharts in firmware development in the sixth chapter. The seventh chapter discusses the results and conclusions and future improvements are discussed in last chapter. Three appendices constituting pin configurations, coding and photographs are provided.

## **CHAPTER 2**

# **PRODUCT RELIABILITY TEST DETAILS**

## **2. PRODUCT RELIABILITY TEST DETAILS**

### **2.1 INTRODUCTION**

Various noises found in the automotive environments are discussed in chapter 1. Various tests are conducted on speedometers to check their reliability against the various noises found in the automotive environments.

### **2.2 TYPES OF TEST**

The following tests are conducted in the product reliability assurance laboratory of the company to test the products before delivery

- Environmental test
- Dynamic test
- Endurance test
- Electrical test

#### **2.2.1 ENVIRONMENTAL TEST:**

These tests are encountered since there is a chance that the meters may be subject to various climatic changes such as hot, humidity, dust etc., that may spoil the meters and their performance may be affected. The meters are made tolerable to these conditions for prolonged operation. The various tests conducted under this are:

- Temperature test
  - Hot chamber (Temperature: Room temp to 200 C)
  - Hot cold chamber: 10-80 C
  - Cold chamber: 0 C to -40 C
  - Vacuum cold heat and humidity test
  - Cold heat and climatic chamber
- Moisture resistance test/Humidity test

- Other environmental test
  - Water spray
  - Corrosive fog exposure
  - Ozone test

### **2.2.2 DYNAMIC TEST:**

When there are various conditions at which the meter operates in a vehicle it may undergo vibration, shock, bumps, etc., under running conditions

### **2.2.3 ENDURANCE TEST:**

The life of the meters fitted in an automobile for various purposes may have a good performance but shortest lifetime. The endurance of speedometer, cables, electrical gauges, pressure gauges, are tested for their endurance. The various tests conducted under this are:

- Speedometer endurance
- Cable endurance
- Gauge endurance
- Sensor endurance

### **2.2.4 ELECTRICAL TEST:**

Switching current loads on or off will cause transients in power lines. Hence speedometers, fuel indicators are tested for their reliability when exposed to such transients. The various tests conducted to check the reliability of the product are:

- Power supply fluctuation test
- Digital gauss meter
- Transient voltage surge generator
- High frequency pulse tester
- Load dump tester
- Field decay tester
- Insulator and break down tester

## 2.3 POWER SUPPLY FLUCTUATION TEST:

### 2.3.1 SAMPLE TEST STANDARD - DENSO STANDARD (JAPAN):

#### 2.3.1.1 PURPOSE:

This test is conducted to check for possible fluctuation in supply voltage at startup of the starter, under the test conditions described below. This is a Reliability assurance test and the speedometer and fuel indicator needles should not deflect beyond the range ,through the test.

#### 2.3.1.2 METHOD:

Apply the voltage waveform given below to the line between the sample's power terminals, using a test circuit illustrated below under the following conditions

-SPEED: 0Km/h

-TACHO: 0 rpm

-FUEL: F

-TEMP: Center

The above mentioned operating conditions are maintained through the test. Speed and tachometer readings must be 0, fuel should be full and temperature must be normal

#### 2.3.1.3 TEST CIRCUIT DIAGRAM:

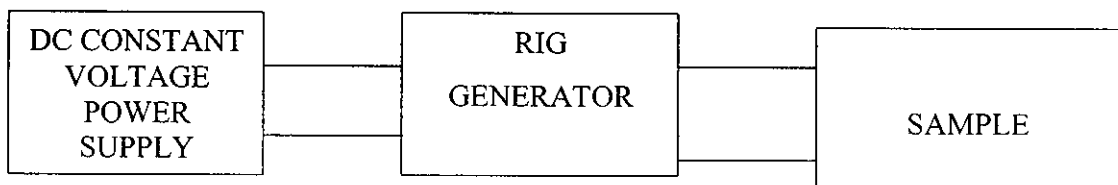


Figure 2.1

Test Circuit Diagram

A DC constant voltage source supplies power to the Rig generator circuitry. The Rig generator

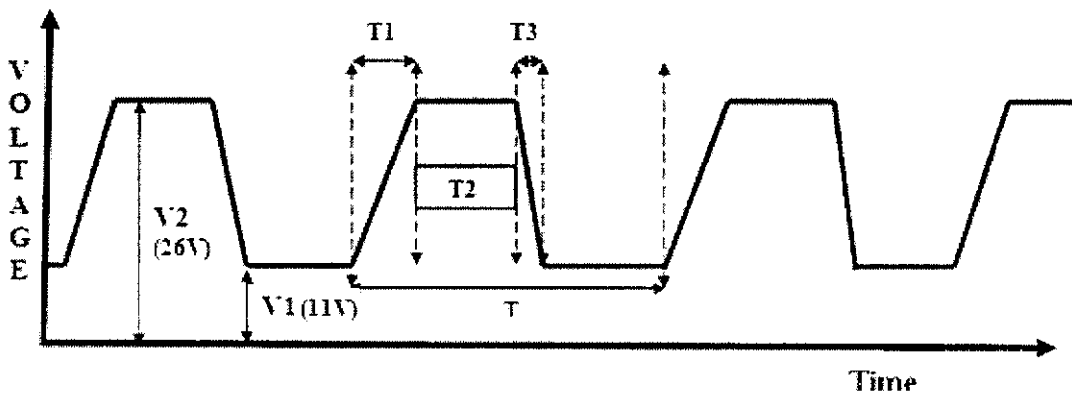


s directly connected to the test sample as shown and the test is carried out for the specified time. The voltage waveform shown below is generated in the Rig generator and applied to the test sample to check whether it withstands the power supply fluctuation test.

### 3.1.4 CONDITIONS:

- VOLTAGE WAVEFORM:

Figure 2.2 Voltage waveform



- |                        |                             |
|------------------------|-----------------------------|
| <b>T1- Rise time</b>   | <b>T - Total period</b>     |
| <b>T2- Pulse Width</b> | <b>V1-DC Offset Voltage</b> |
| <b>T3- Fall time</b>   | <b>V2- Amplitude</b>        |

The above shown waveform is the standard waveform for power supply fluctuation test. Only the values of T, T1, T2 and T3 vary for different standards whereas the basic wave shape remains the same. By varying the time periods the slope of the wave changes while retaining the basic shape.



P-2083

**CHAPTER 3**  
**ANALYSIS AND DESIGN OF RIG**  
**GENERATOR**

## **3. ANALYSIS AND DESIGN OF RIG GENERATOR**

### **3.1 REQUIREMENTS GATHERED**

The following are the requirements of the project:

- To design a rig generator for 24V battery with adjustable pulse width to check for the possible fluctuation in supply voltage at start up of the starter.
- The obtained voltage waveform would be used to test the samples by means of a standard test circuit.

### **3.2 EXISTING SYSTEM**

- The existing rig generators are standard specific (i.e.) only one standard (say ISO) can be tested for.
- Amplitude and parameters like rise time, fall time, width cannot be varied.
- Only equipments compatible with 12V DC supply can be tested

### **3.3 DESIGN OVERVIEW:**

The hardware circuit is designed such that there are provisions for getting input from the user and displaying the same. The input values are then processed to generate the waveform. Hence the hardware circuitry involves two main modules:

1. Digital part
2. Analog part

#### **3.3.1 DIGITAL PART:**

The digital circuitry here is responsible for getting the various parameters of the waveform as inputs from the user viz- rise time, fall time, width, total period and producing the basic wave shape. It includes the following parts:

- FIRMWARE - PIC controller
- USER INTERFACE – LCD & keypad
- DIGITAL TO ANALOG CONVERTER

### 3.3.2 ANALOG PART:

Analog part is responsible for producing the wave shape of required amplitude and also constitutes of the power supply circuitry for the various components involved. It consists of:

- Amplifier
- Non-inverting buffer
- Power supply units -  $\pm 12V, +36V, +5V$

### 3.4 BLOCK DIAGRAM:

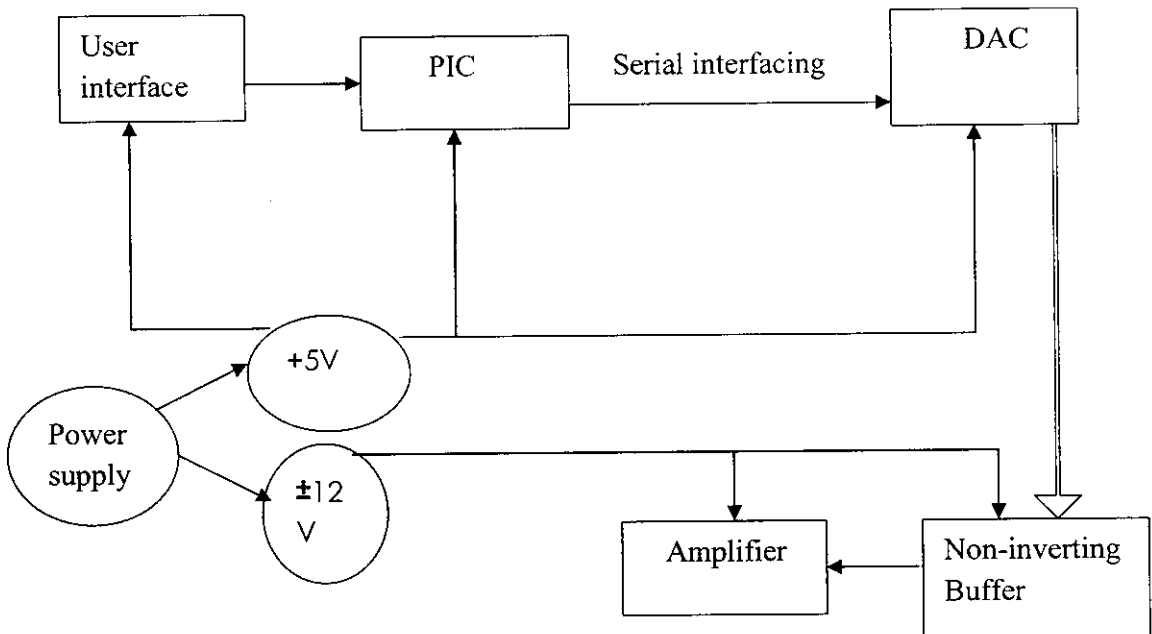


Figure 3.1

Block Diagram of Hardware Setup

The constituents of the block diagram shown above are explained below

### 3.4.1 KEYBOARD SWITCH:

Keyboard scanning is the process of having the microprocessor look at the keyboard at a regular interval to see if a key has been pressed. Once the processor determines that a key has been pressed, the keyboard scanning software filters out the bounce and determine which of the key was pressed. Each key is assigned a unique identifier called a scan code.

- Rollover – pressing more than one key at a time
- Buffering – prevents losing key strokes
- Auto repeat – allows the scan code of the key pressed to be repeatedly inserted into the buffer.
- Shift key – to assign more then one functions.

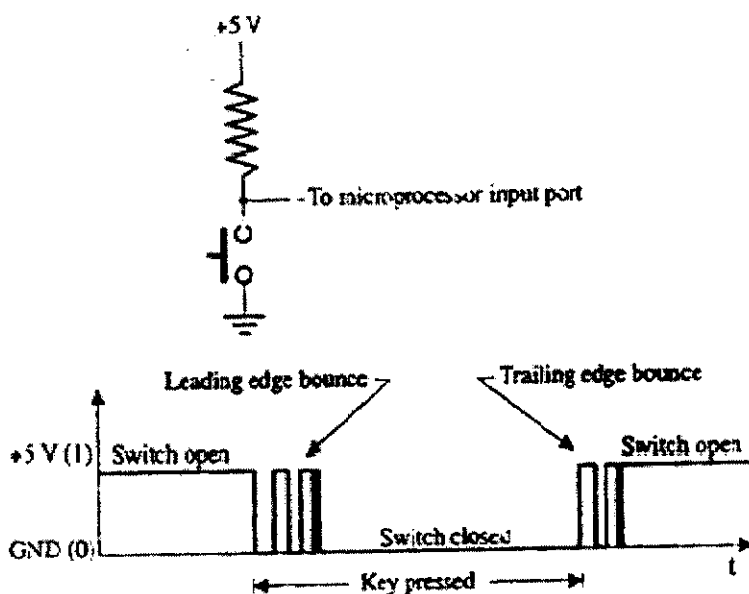


Figure 3.2

Keypad Scanning

### **3.4.2 PIC MICROCONTROLLER:**

The 18 series is the new high end pic architecture, and has proven to be very popular with many device variants being made available by the manufacturer. The PIC 18F4520 architecture is distinctively minimalist. It is characterised by the following features:

- separate code and data spaces (Harvard architecture)
- a small number of fixed length instructions
- most instructions are single cycle execution (4 clock cycles), with single delay cycles upon branches and skips
- a single accumulator (W), the use of which (as source operand) is implied (ie is not encoded in the opcode)
- All RAM locations function as registers as both source and/or destination of math and other functions.
- a fairly small amount of addressable data space (typically 256 bytes), extended through banking

### **3.4.3 DIGITAL TO ANALOG CONVERTER:**

The DAC fundamentally converts finite-precision numbers into a physical quantity, usually an electrical voltage. Normally the output voltage is a linear function of the input number. Usually these numbers are updated at uniform sampling intervals and can be thought of as numbers obtained from a sampling process. These numbers are written to the DAC, sometimes along with a clock signal that causes each number to be latched in sequence, at which time the DAC output voltage changes rapidly from the previous value to the value represented by the currently latched number. The effect of this is that the output voltage is held in time at the current value until the next input number is latched resulting in a piecewise constant output. This is equivalently a zero-order hold operation and has an effect on the frequency response of the reconstructed signal.

The most important characteristics of these devices are:

- **Resolution:** This is the number of possible output levels the DAC is designed to reproduce. This is usually stated as the number of bits it uses, which is the base two logarithm of the number of levels.. Resolution is related to the **Effective Number of Bits** (ENOB) which is a measurement of the actual resolution attained by the DAC.
- **Maximum sampling frequency:** This is a measurement of the maximum speed at which the DACs circuitry can operate and still produce the correct output.
- **Monotonicity:** This characteristic is very important for DACs used as a low frequency signal source or as a digitally programmable trim element.

The MCP4922 device is voltage output string DAC. These devices include input amplifiers, rail-to-rail output amplifiers, reference buffers, shutdown and reset management circuitry. Serial communication conforms to the SPI protocol. The MCP4922 operates from 2.7V to 5.5V supplies. The coding of these devices is straight binary and the ideal output voltage is given by Equation

$$V_{OUT} = \frac{V_{REF}GD_N}{2^n}$$

where  $G$  is the selected gain (1x or 2x),  $DN$  represents the digital input value and  $n$  represents the number of bits of resolution ( $n = 12$ ).

#### 3.4.4 AUDIO AMPLIFIER:

The LM3886 is a high-performance audio power amplifier capable of delivering 68W. The performance of the LM3886, utilizing its Self Peak Instantaneous Temperature (Spike) Protection Circuitry, puts it in a class above discrete and hybrid amplifiers. Spike Protection means that these parts are completely safeguarded at the output against overvoltage, undervoltage, overloads, thermal runaway and instantaneous temperature peaks. The LM3886 also maintains an excellent Signal-to-Noise Ratio. Mute Function: The muting function of the LM3886 allows the user to mute the music going into the amplifier by drawing less than 0.5 mA out of pin 8 of the device. The heat sink is chosen to dissipate the maximum IC power for a given supply voltage and rated load.

**CHAPTER 4**

**FEATURES OF**

**MICROCONTROLLER**

**PIC 18F452**



## **4.FEATURES OF MICROCONTROLLER PIC18F452**

### **4.1 INTRODUCTION**

The high performance of the PIC18F452 devices can be attributed to a number of architectural features commonly found in RISC microprocessors. These include:

- Harvard architecture
- Long Word Instructions
- Single Word Instructions
- Single Cycle Instructions
- Instruction Pipelining
- Reduced Instruction Set

#### **4.1.1 HARVARD ARCHITECTURE:**

Harvard architecture has the program memory and data memory as separate memories, which are accessed from separate buses. This improves bandwidth over traditional von Neumann architecture in which program and data are fetched from the same memory using the same bus. To execute an instruction, a von Neumann machine must make one or more (generally more) accesses across the 8-bit bus to fetch the instruction. Then data may need to be fetched, operated on and possibly written. As can be seen from this description, the bus can become extremely congested. With Harvard architecture, the instruction is fetched in a single instruction cycle (all 16 bits). While the program memory is being accessed, the data memory is on an independent bus and can be read and written. These separated busses allow one instruction to execute, while the next instruction is fetched.

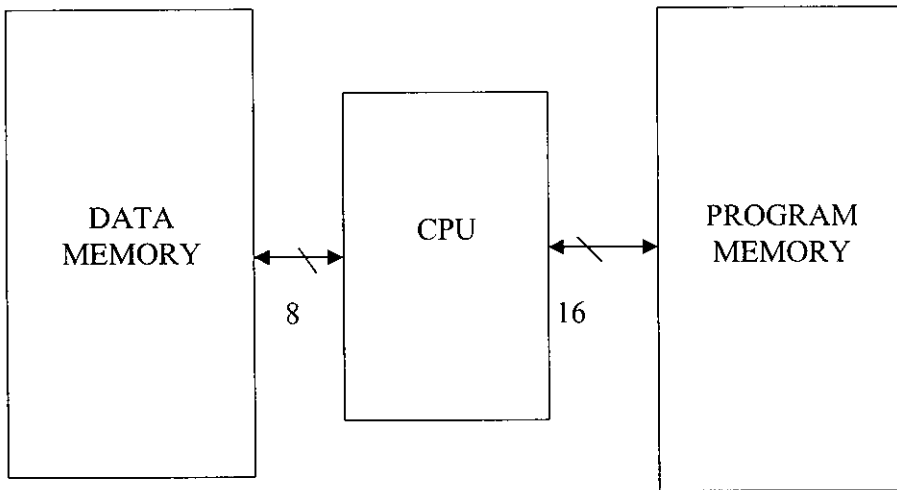


Figure 4.1

Harvard Architecture

#### 4.1.2 REDUCED INSTRUCTION SET:

When an instruction set is well designed and highly orthogonal (symmetric), fewer instructions are required to perform all needed tasks. With fewer instructions, the whole set can be more rapidly learned.

#### 4.1.3 INSTRUCTION PIPELINE:

The instruction pipeline is a two-stage pipeline that overlaps the fetch and execution of instructions. The fetch of the instruction takes one TCY, while the execution takes another TCY. However, due to the overlap of the fetch of current instruction and execution of previous instruction, an instruction is fetched and another instruction is executed every TCY.

### 4.2 FEATURES OF PIC USED

#### 4.2.1 TIMER 0:

The Timer0 module has the following features:

- Software selectable as an 8-bit or 16-bit timer/counter
- Readable and writable
- Dedicated 8-bit software programmable prescaler

- Clock source selectable to be external or internal
- Interrupt on overflow from FFh to 00h (FFFFh to 0000h in 16-bit mode)
- Edge select for external clock

#### 4.2.1.1 CONTROL REGISTER:

The T0CON register is a readable and writable register that controls all the aspects of Timer0, including the prescale selection.

Table 4.1: T0CON: TImeR0 Control Register

TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
--------	--------	------	------	-----	-------	-------	-------

bit 7 **TMR0ON**: Timer0 On/Off Control bit

1 = Enables Timer0

0 = Stops Timer0

bit 6 **T08BIT**: Timer0 8-bit/16-bit Control bit

1 = Timer0 is configured as an 8-bit timer/counter

0 = Timer0 is configured as a 16-bit timer/counter

bit 5 **T0CS**: Timer0 Clock Source Select bit

1 = Transition on T0CKI pin is clock (counter mode)

0 = Internal instruction cycle is clock (timer mode)

bit 4 **T0SE**: Timer0 Source Edge Select bit

1 = Increment on high-to-low transition on T0CKI pin

0 = Increment on low-to-high transition on T0CKI pin

bit 3 **PSA**: Timer0 Prescaler Assignment bit

1 = Timer0 prescaler is NOT assigned. Timer0 clock input bypasses prescaler.

0 = Timer0 prescaler is assigned. Timer0 clock input comes from prescaler output.

bit 2-0 **T0PS2:T0PS0**: Timer0 Prescaler Select bits

#### **4.2.1.2 TIMER/COUNTER MODES:**

Timer mode is selected by clearing the T0CS bit (T0CON register). In timer mode, the Timer0 module will increment every instruction cycle (without prescaler). If the TMR0 register is written, the increment is inhibited for the following two instruction cycles. The user can work around this by writing an adjusted value to the TMR0 register.

Counter mode is selected by setting the T0CS bit (T0CON register). In counter mode, Timer0 will increment either on every rising or falling edge of the T0CKI pin. The incrementing edge is determined by the Timer0 Source Edge Select bit T0SE (T0CON register). Clearing the T0SE bit selects the rising edge.

#### **4.2.2 SYNCHRONOUS SERIAL PORT:**

The Synchronous Serial Port (SSP) module is a serial interface useful for communicating with other peripherals or microcontroller devices. The SSP module can operate in one of two modes:

- Serial Peripheral Interface (SPI)
- Inter-Integrated Circuit (I2C)

##### **4.2.2.1 SPI MODE:**

The SPI mode allows 8-bits of data to be synchronously transmitted and received, simultaneously. All four modes of SPI are supported. To accomplish communication, typically three pins are used:

- Serial Data Out (SDO) - RC5/SDO
- Serial Data In (SDI) - RC4/SDI/SDA
- Serial Clock (SCK) - RC3/SCK/SCL/LVDIN

Four registers for SPI mode operation. These are:

- SSP Control Register1 (SSPCON1)
- SSP Status Register (SSPSTAT)
- Serial Receive/Transmit Buffer (SSPBUF)
- SSP Shift Register (SSPSR) - Not directly accessible

SSPCON1 and SSPSTAT are the control and status registers in SPI mode operation. The SSPCON1 register is readable and writable. The lower 6 bits of the SSPSTAT are read only. The upper two bits of the SSPSTAT are read/write.

SSPSR is the shift register used for shifting data in or out. SSPBUF is the buffer register to which data bytes are written to or read from.

In receive operations, SSPSR and SSPBUF together create a double buffered receiver. When SSPSR receives a complete byte, it is transferred to SSPBUF and the SSPIF interrupt is set.

During transmission, the SSPBUF is not double buffered. A write to SSPBUF will write to both SSPBUF and SSPSR.

Table 4.2 SPI CONTROL REGISTER

WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
------	-------	-------	-----	-------	-------	-------	-------

bit 7 **WCOL**: Write Collision Detect bit (Transmit mode only)

1 = The SSPBUF register is written while it is still transmitting the previous word (must be cleared in software)

0 = No collision

bit 6 **SSPOV**: Receive Overflow Indicator bit

1 = A new byte is received while the SSPBUF register is still holding the previous data

0 = No overflow

bit 5 **SSPEN**: Synchronous Serial Port Enable bit

1 = Enables serial port and configures SCK, SDO, SDI, and SS as serial port pins

0 = Disables serial port and configures these pins as I/O port pins

bit 4 **CKP**: Clock Polarity Select bit

1 = IDLE state for clock is a high level

0 = IDLE state for clock is a low level

bit 3-0 **SSPM3:SSPM0**: Synchronous Serial Port Mode Select bits

#### 4.2.2.2 SSPSTAT REGISTER & SSPCON REGISTER:

Table 4.3 Synchronous Serial Port Status Register

SMP	CKE	D/A	P	S	R/W	UA	BF
-----	-----	-----	---	---	-----	----	----

bit 7 **SMP**: SPI data input sample phase

1 = Input data sampled at end of data output time

0 = Input data sampled at middle of data output time

bit 6 **CKE**: SPI Clock Edge Select

CKP = 0 (SSPCON<4>)

1 = Data transmitted on rising edge of SCK

0 = Data transmitted on falling edge of SCK

CKP = 1 (SSPCON<4>)

1 = Data transmitted on falling edge of SCK

0 = Data transmitted on rising edge of SCK

bit 5 **D/A**: Data/Address bit (I<sup>2</sup>C mode only)

bit 4 **P**: Stop bit(I<sup>2</sup>C mode only)

bit 3 **S**: Start bit(I<sup>2</sup>C mode only)

bit 2 **R/W**: Read/Write bit information (I<sup>2</sup>C mode only)

bit 1 **UA**: Update Address (10-bit I<sup>2</sup>C mode only)

bit 0 **BF**: Buffer Full Status bit

Receive (SPI and I<sup>2</sup>C modes)

1 = Receive complete, SSPBUF is full

0 = Receive not complete, SSPBUF is empty

#### **4.2.3 DATA EEPROM MEMORY:**

The Data EEPROM is readable and writable during normal operation over the entire VDD range. There are four SFRs used to read and write the program and data EEPROM memory. These registers are:

- EECON1
- EECON2
- EEDATA
- EEADR

The EEPROM data memory allows byte read and writes. When interfacing to the data memory block, EEDATA holds the 8-bit data for read/write and EEADR holds the address of the EEPROM location being accessed. These devices have 256 bytes of data EEPROM with an address range from 0h to FFh.

##### **4.2.3.1 READING THE DATA EEPROM MEMORY:**

To read a data memory location, the user must write the address to the EEADR register, clear the EEPGD control bit (Data EEPROM Memory Select bit EECON1<7>), clear the CFGS control bit (Configuration Select bit EECON1<6>), and then set control

bit RD (Read Control bit EECON1<0>). The data is available for the very next instruction cycle; therefore, the EEDATA register can be read by the next instruction. EEDATA will hold this value until another read operation, or until it is written to by the user (during a write operation).

#### **4.2.3.1 WRITING TO THE DATA EEPROM MEMORY:**

To write an EEPROM data location, the address must first be written to the EEADR register and the data written to the EEDATA register.

Additionally, the WREN bit in EECON1 must be set to enable writes. This mechanism prevents accidental writes to data EEPROM due to unexpected code execution (i.e., runaway programs). The WREN bit should be kept clear at all times, except when updating the EEPROM. The WREN bit is not cleared by hardware. At the completion of the write cycle, the WR bit is cleared in hardware and the EEPROM Write Complete Interrupt Flag bit (EEIF) is set.

#### **4.2.4 I/O PORTS:**

Each port has three registers for its operation. These registers are:

- TRIS register (data direction register)
- PORT register (reads the levels on the pins of the device)
- LAT register (output latch)

##### **4.2.4.1 PORTA:**

In PORTA is a 7-bit wide, bi-directional port. The corresponding Data Direction register is TRISA.

TRISA bit = 1 PORTA pin an input

TRISA bit = 0 PORTA pin an output

Here port A is configured as input port.



TABLE 4.4 PORTA FUNCTIONS

Name	Bit#	Buffer	Function
RA0/AN0	bit0	TTL	Input/output or analog input.
RA1/AN1	bit 1	TTL	Input/output or analog input.
RA2/AN2/REF-	bit2	TTL	Input/output or analog input or VREF-.
RA3/AN3/REF+	bit3	TTL	Input/output or analog input or VREF+.
RA4/T0CKI	bit4	ST	Input/output or external clock input for Timer0. Output is open drain type.
RA5/SS/AN4/LVDIN	bit5	TTL	Input/output or slave select input for synchronous serial port or analog input, or low voltage detect input.
OSC2/CLKO/RA6	bit6	TTL	OSC2 or clock output or I/O pin.

**4.2.4.2 PORTC:**

PORTC is an 8-bit wide, bi-directional port. The corresponding Data Direction register is TRISC.

TRISC bit = 1 PORTC pin an input

TRISC bit = 0 PORTC pin an output

Here port C is used for serial communication with DAC.

TABLE 4.5 PORTC FUNCTIONS

Name	Bit#	Buffer Type	Function
RC0/T1OSO/T1CKI	bit0	ST	Input/output port pin or Timer1 oscillator output/Timer1 clock input.
RC1/T1OSI/CCP2	bit 1	ST	Input/output port pin, Timer1 oscillator input, or Capture2 input/Compare2 output/PWM output when CCP2MX configuration bit is set.
RC2/CCP1	bit2	ST	Input/output port pin or Capture1 input/Compare1 output/PWM1 output.
RC3/SCK/SCL	bit3	ST	RC3 can also be the synchronous serial clock for both SPI and I <sup>2</sup> C modes.
RC4/SDI/SDA	bit4	ST	RC4 can also be the SPI Data In (SPI mode) or Data I/O (I <sup>2</sup> C mode).
RC5/SDO	bit5	ST	Input/output port pin or Synchronous Serial Port data output.
RC6/TX/CK	bit6	ST	Input/output port pin, Addressable USART Asynchronous Transmit, or Addressable USART Synchronous Clock.
RC7/RX/DT	bit7	ST	Input/output port pin, Addressable USART Asynchronous Receive, or Addressable USART Synchronous Data.

#### 4.2.4.3 PORTD:

PORTD is an 8-bit wide, bi-directional port. The corresponding Data Direction register is TRISD.

TRISD bit = 1 PORTD pin an input

TRISD bit = 0 PORTD pin an output

Here port D is configured as output port for parallel transmission of data to the LCD

TABLE 4.6 PORTD FUNCTIONS

Name	Bit#	Buffer Type	Function
RD0/PSP0	bit0	ST/TTL <sup>(1)</sup>	Input/output port pin or parallel slave port bit0.
RD1/PSP1	bit1	ST/TTL <sup>(1)</sup>	Input/output port pin or parallel slave port bit1.
RD2/PSP2	bit2	ST/TTL <sup>(1)</sup>	Input/output port pin or parallel slave port bit2.
RD3/PSP3	bit3	ST/TTL <sup>(1)</sup>	Input/output port pin or parallel slave port bit3.
RD4/PSP4	bit4	ST/TTL <sup>(1)</sup>	Input/output port pin or parallel slave port bit4.
RD5/PSP5	bit5	ST/TTL <sup>(1)</sup>	Input/output port pin or parallel slave port bit5.
RD6/PSP6	bit6	ST/TTL <sup>(1)</sup>	Input/output port pin or parallel slave port bit6.
RD7/PSP7	bit7	ST/TTL <sup>(1)</sup>	Input/output port pin or parallel slave port bit7.

#### 4.2.4.4 PORTE:

PORTE is a 3-bit wide, bi-directional port. The corresponding Data Direction register is TRISE

TRISE bit = 1 PORTE pin an input

TRISE bit = 0 PORTE pin an output

**CHAPTER 5**  
**HARDWARE DESCRIPTION**

## **5. HARDWARE DESCRIPTION**

### **5.1 INTRODUCTION**

This chapter explains about the circuit diagrams (analog, digital, power supply) , various components (LCD, keypad, PIC, DAC, audio amplifier) and interfacing techniques used in the development of the microcontroller based 24V Rig generator.

### **5.2 CIRCUIT DESCRIPTION**

The hardware circuit is designed such that there are provisions for getting input from the user and displaying the same. The input values are then processed to generate the waveform. Hence the hardware circuitry involves two main modules:

1. Digital part
2. Analog part

### **5.3 DIGITAL PART**

The digital part consists of

- Key pad
- LCD display
- Microcontroller PIC18F452
- DAC

### 5.3.1 CIRCUIT DIAGRAM OF THE DIGITAL PART

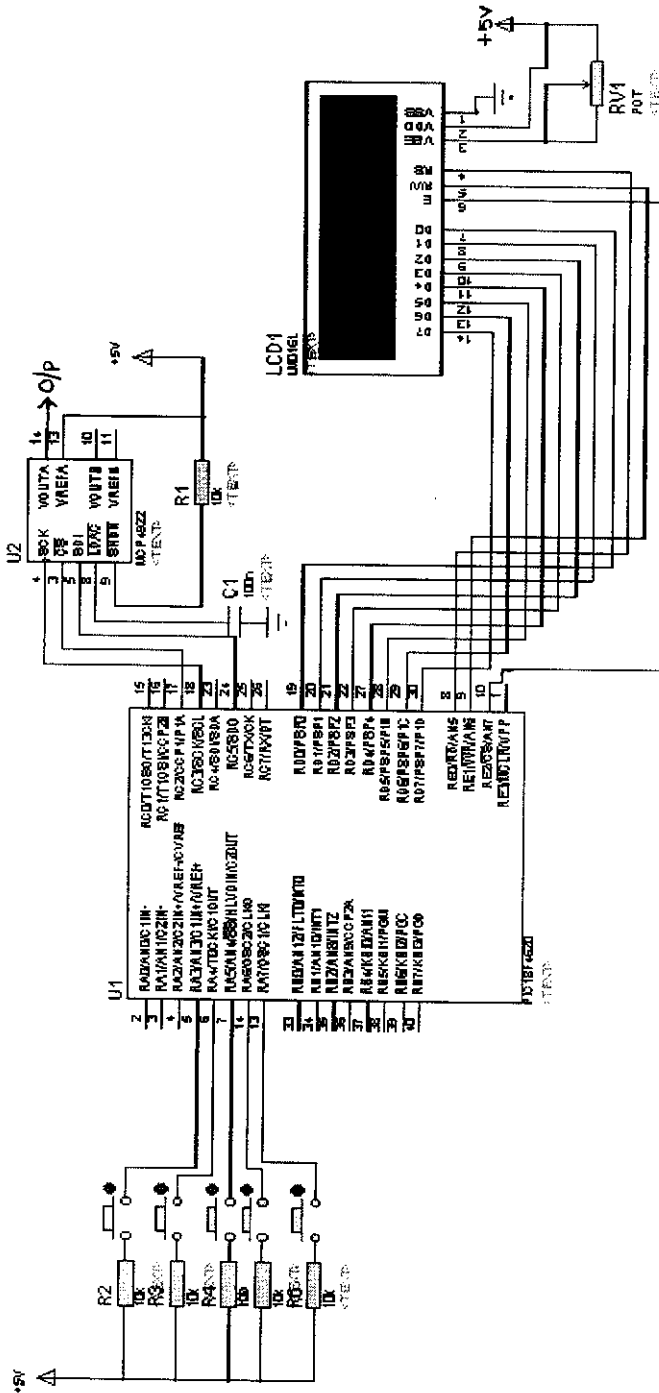


Figure 5.1 circuit diagram-digital part

### 5.3.2 INTERFACING OF LCD WITH MICROCONTROLLER:

The LCD can be interfaced with the microcontroller .The data lines of LCD denoted by the pin numbers 7,8,9,10,11,12,13,14 are connected PORT D pins of the microcontroller configured as output port. The pin 1 of LCD is grounded pin 3 is connected through a 10K potentiometer to 5V for contrast adjustment and pin 2 is given to the VCC, 5V. The LCD panel's Enable and Register Select is connected to the Control Port of microcontroller (PORT E pins configured as output port). We make no effort to place the Data bus into reverse direction. Therefore we hard wire the R/W line of the LCD panel, into write mode. This will cause no bus conflicts on the data lines.

### 5.3.3 KEY PAD MODULE:

The keyboard is used to input the numerical data

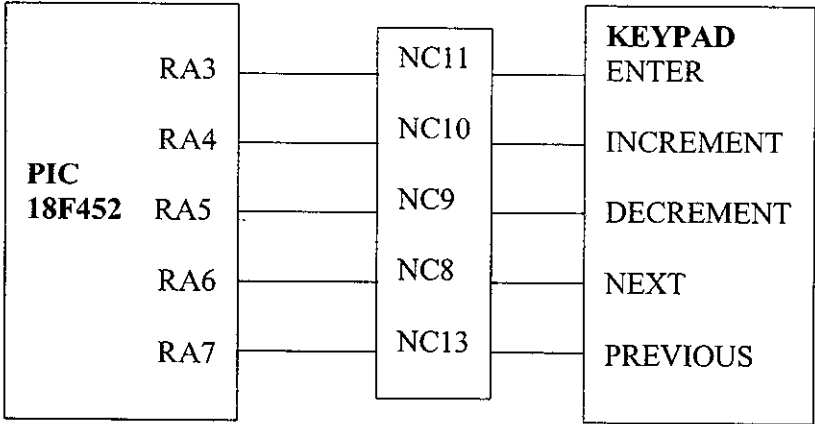


Fig 5.2 Key pad scanning

The port pins are connected to connector 10 of the microcontroller board. This is further connected to the keys of the keypad. Here we have used the following keys:

- ENTER – This key accepts the input value and goes to the next parameter.
- INCREMENT - This key increments the value of the desired parameter.
- DECREMENT - This key decrements the value of the desired parameter.
- NEXT – This key moves the cursor to the next value.
- PREVIOUS – This key moves the cursor to the previous value.

### 5.3.4 SERIAL COMMUNICATION WITH DAC:

The MCP4922 family is designed to interface directly with the Serial Peripheral Interface (SPI) port, available on the microcontroller in Mode 0, 0. Commands and data are sent to the device via the SDI pin, with data being clocked-in on the rising edge of SCK. The communications are unidirectional and, thus, data cannot be read out of the MCP4922. The CS pin is held low for the duration of a write command. The write command consists of 16 bits and is used to configure the DAC's control and data latches.

The write command is initiated by driving the CS pin low, followed by clocking the four configuration bits and the 12 data bits into the SDI pin on the rising edge of SCK. The CS pin is then raised, causing the data to be latched into the selected DAC's input registers. The MCP4922 utilizes a double-buffered latch structure to allow both DACA's and DACB's outputs to be synchronized with the LDAC pin, if desired. Upon the LDAC pin achieving a low state, the values held in the DAC's input registers are transferred into the DACs' output registers. The outputs will transition to the value and held in the DACX register. All writes to the MCP4922 are 16-bit words. Any clocks past 16 will be ignored. The most significant four bits are configuration bits. The remaining 12 bits are data bits. No data can be transferred into the device with CS high. This transfer will only occur if 16 clocks have been transferred into the device. If the rising edge of CS occurs prior, shifting of data into the input registers will be aborted

### 5.3.5 WRITE COMMAND REGISTER:

Bit 15 **A/B**: DACA or DACB Select bit

1 = Write to DACB

0 = Write to DACA

Bit 14 **BUF**: VREF Input Buffer Control bit

1 = Buffered

0 = Unbuffered

Bit 13 **GA**: Output Gain Select bit

1 = 1x ( $V_{OUT} = V_{REF} * D/4096$ )

0 = 2x ( $V_{OUT} = 2 * V_{REF} * D/4096$ )

Bit 12 **SHDN**: Output Power down Control bit

1 = Output Power down Control bit

0 = Output buffer disabled, Output is high impedance

Bit 11-0 **D11:D0**: DAC Data bits

12 bit number "D" which sets the output value. Contains a value between 0 and 4095.

## 5.4 ANALOG PART

Analog part is responsible for producing the wave shape of required amplitude and also constitutes of the power supply circuitry for the various components involved. It consists of:

- Amplifier
- Non-inverting buffer
- Power supply units -  $\pm 12V, +36V, +5V$



# CIRCUIT DIAGRAM-ANALOG PART

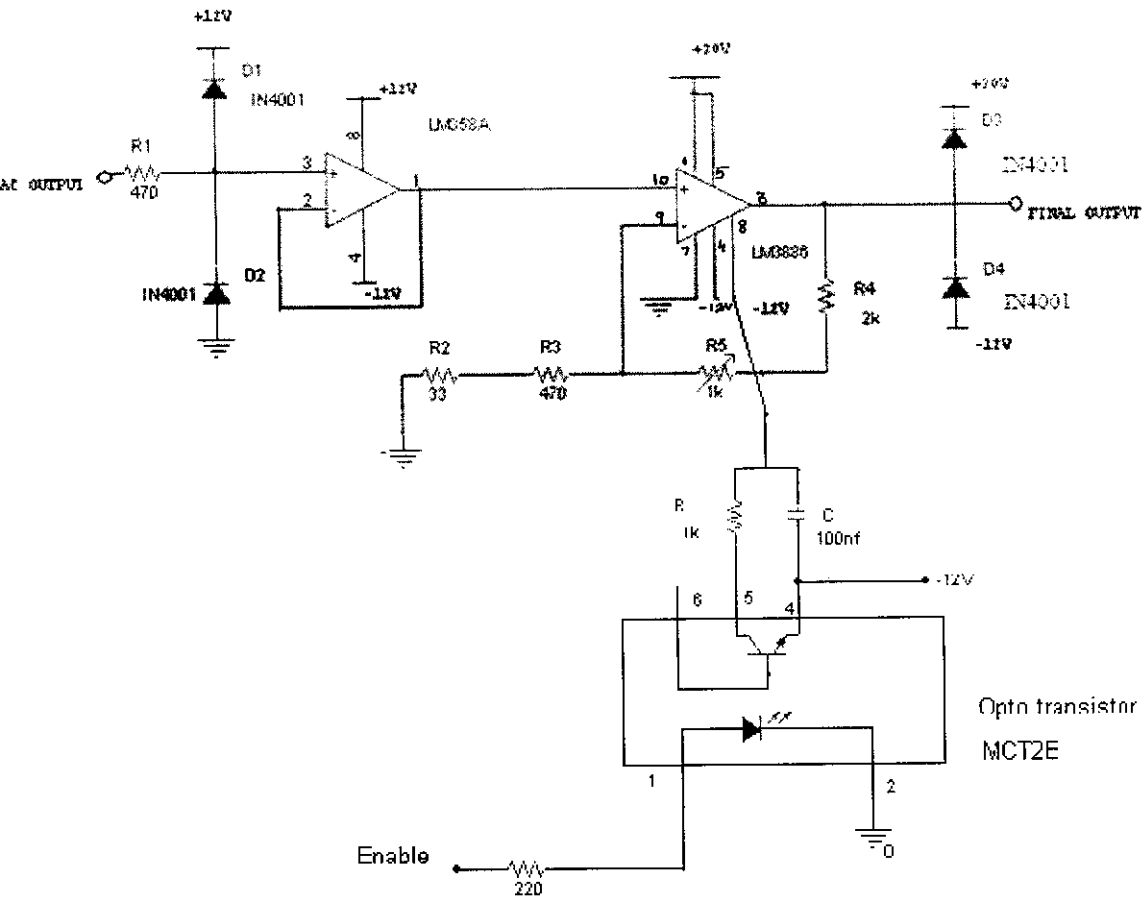


figure 5.3 circuit diagram-analog part

R<sub>i</sub> - Inverting input resistance to provide AC Gain in conjunction with R<sub>f</sub>.

$$R_i = R_4 + R_5$$

R<sub>f</sub> - Feedback resistance to provide AC Gain in conjunction with R<sub>i</sub>.

$$R_f = R_2 + R_3$$

$$\text{Gain} = \left[ \frac{R_f}{R_i} \right] + 1 \longrightarrow \textcircled{1}$$

Output from DAC is 4.95V and voltage required at the final output stage is 30V

Hence required gain is 6

Assuming  $R_i = 500 \Omega$ , then from eq1  $R_f = 3000 \Omega$

The output from DAC is given to the pin 2 of LM358. The pins 4 and 8 of LM358 are supply pins; 3 - non-inverting input; 2 - inverting input and pin 1 gives the output. LM358 is configured as unity gain voltage follower.

This voltage follower is followed by the Audio amplifier LM3886. The pins 1 and 5 of this audio amplifier are connected to +12V; pin 4 to -12V. Pin 10 is the non-inverting input; pin 9 is the inverting input and pin 3 gives the output. The mute function of LM3886 is activated by connecting a negative polarity DC voltage to its mute pin (pin 8). The opto transistor is used for this operation. When 5V is given to pin 1 of the opto transistor it gets turned on and connects pin (mute pin) of LM3886 to -12V thus muting the audio amplifier. The diodes are used for protection purpose.

## 5.5 POWER SUPPLY

### 5.5.1 CIRCUIT DIAGRAM

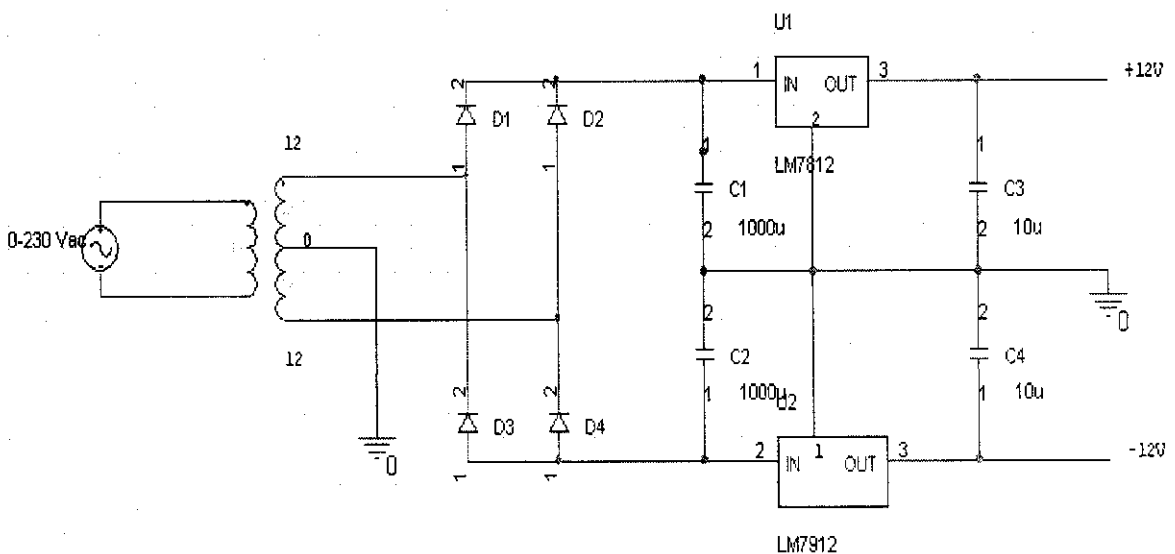


figure 5.4 circuit diagram-power supply

The power circuit consists of bridge rectifier which converts 12V AC from secondary of a (0-230V/12-0-12V) transformer to DC. A diode bridge or bridge rectifier is an arrangement of four diodes connected in a bridge circuit as shown below, that provides the same polarity of output voltage for any polarity of the input voltage. When used in its most common application, for conversion of alternating current (AC) input into direct current (DC) output, it is known as a bridge rectifier. The bridge rectifier provides full wave rectification from a two wire AC input (saving the cost of a center tapped transformer) but has two diode drops rather than one reducing efficiency over a center tap based design for the same output voltage. The output from bridge rectifier is given to voltage regulators LM7812 (for +12V) and LM912 (for -12V). The OUT pin of LM7812 gives +12V and OUT pin of LM7912 gives -12V. The +12V obtained when given to LM7805 gives +5V output. The capacitors used in the circuit are for filtering purpose.

## **5.6 WORKING OF THE CIRCUIT**

The time values set using keyboard is stored in data EEPROM of the PIC microcontroller. Port A, used for key pad is configured as input port by setting bits of TRISA register. PORT C is connected to DAC. serial clock SCK and serial DATAOUT pins (RC4 and RC5 respectively) are connected to the corresponding pins of DAC. RC3 is used as CS (Active low). If 0 is passed to this pin the DAC gets selected. PORT D is configured as output port by resetting the bits of TRISB register. The control pins for LCD namely the R/W, RS, ENABLE are connected to the pins of port E. PORT E is configured as output port by resetting pins of TRISE register. Timer 0 is used in scanning keypad and producing Software delay. Timer mode is selected by clearing the T0CS bit (T0CON register). In timer mode, the Timer0 module will increment every instruction cycle (without prescaler). If the TMR0 register is written, the increment is inhibited for the following two instruction cycles. The user can work around this by writing an adjusted value to the TMR0 register. Then the time values are sampled into small step sizes and serially transmitted to the digital to analog converter at end of each pulse of the SCK (serial clock), RC<sub>4</sub> of PIC controller.

The DAC fundamentally converts finite-precision numbers into a physical quantity, usually an electrical voltage. Normally the output voltage is a linear function of the input number. Usually these numbers are updated at uniform sampling intervals and can be thought of as numbers obtained from a sampling process. These numbers are written to the DAC, sometimes along with a clock signal that causes each number to be latched in sequence, at which time the DAC output voltage changes rapidly from the previous value to the value represented by the currently latched number. The effect of this is that the output voltage is held in time at the current value until the next input number is latched resulting in a piecewise constant output. This is equivalently a zero-order hold operation and has an effect on the frequency response of the reconstructed signal.

The output of the DAC is around 4.96V. This signal is then fed to the output drive circuit. LM 386, dual op-amp, in the drive circuit is connected as non-inverting, unity gain amplifier for impedance matching. This is followed by LM 3886 audio amplifier. The mute function of LM 3886 is activated by connecting a negative polarity DC voltage to its mute pin (pin 8). The opto transistor is used to do this operation. When 5V is given to pin 1 of the opto transistor it gets turned on and connects 8<sup>th</sup> pin (mute pin) of LM3886 to -12V thus muting the audio amplifier. The gain of the amplifier is set as 6, by suitable values of  $R_i$ ,  $R_f$ , thus it provides necessary amplification, and the final output is 30V. The heat sink is chosen to dissipate the maximum IC power for a given supply voltage and rated load.

**CHAPTER 6**  
**FIRMWARE DEVELOPMENT**

## 6. FIRMWARE DEVELOPMENT

### 6.1 ALGORITHM OF RIG GENERATION

The following algorithm is for the entire operation of Rig Generator. It shows the step by step procedure and logic followed in generating the wave shape.

- Step 1: Start.
- Step 2: Display options like 'set', 'run' and 'reset'.
- Step 3: Check for selection of option .If 'set' option then go to next step else if 'run' option then go to step10 else if 'reset 'option goto step17
- Step 4: Read various input parameters like rise time, fall time, pulse width, total period and dc offset from keyboard.
- Step 4a: Display 'setting' screen. Set cursor at last digit entry position.
- Step 4b: Check for the key closure.
- Step 4c: If 'up arrow' increment the digit value by one.
- Step 4d: If 'down arrow' decrement the digit value by one.
- Step 4e: If 'right arrow' move cursor to previous bit position.
- Step 4f: If 'left arrow' move cursor to next bit position.
- Step 4g: If 'enter' move to next screen until total period value entered else go to step 5.
- Step 4h: Store the value entered.
- Step 4i: Go to step 4a.
- Step 5: Check if all the input parameters are non-zero then goto step7.
- Step 6: Display error message and re-prompt for user input for those wrong entries and go to step5.
- Step 7: Check if total period is the highest else go to step6.
- Step 8: Capture key-in values in corresponding parameters.

Step 9: Go to step2.

Step 10: Check for selection of 'run' option.

Step 11: Check if the parameters are non-zero then goto step12.

Step 12: Else set default values for all parameters.

Step 13: Calculate step values corresponding to rise time and fall time.

Step 14: Calculate initial value for reference level of the wave pattern.

Step 15: Check for stop key closure once in 20ms. If closed go to step2 else go to next step.

Step 16: Call DAC function.

Step 16a: For the rise time interval pass the cumulative sum of initial value and step value to the DAC register.

Step16b: Maintain the last obtained level for 'pulse width' duration.

Step16c: For the fall time interval pass the cumulative difference between the previous level and step value to the DAC register.

Step16d: Maintain the DAC output at the last calculated value for the remaining time of the total period.

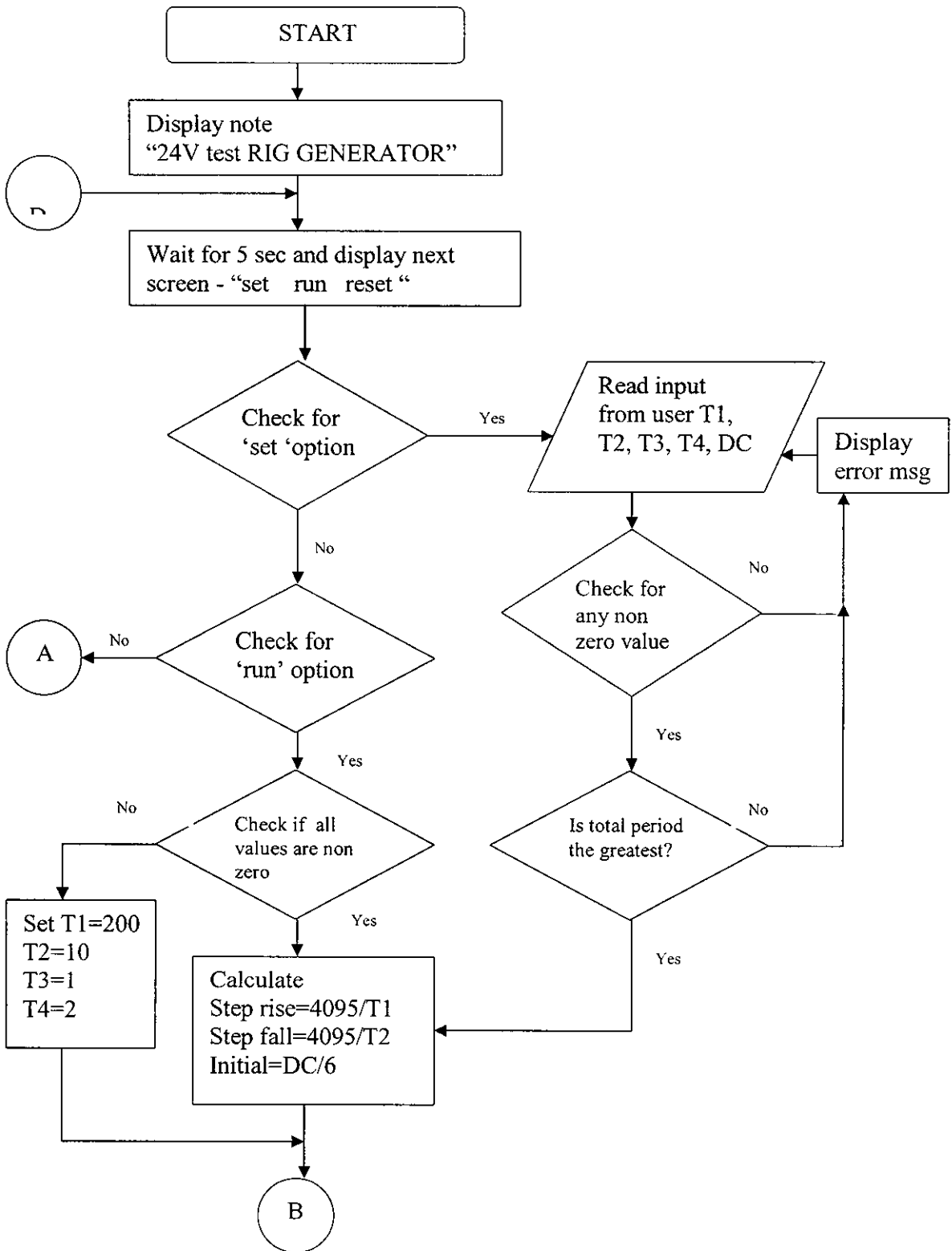
Step 16e: Go to step 16a.

Step 17: Check for selection of 'reset' option then reset all parameter input values to zero.

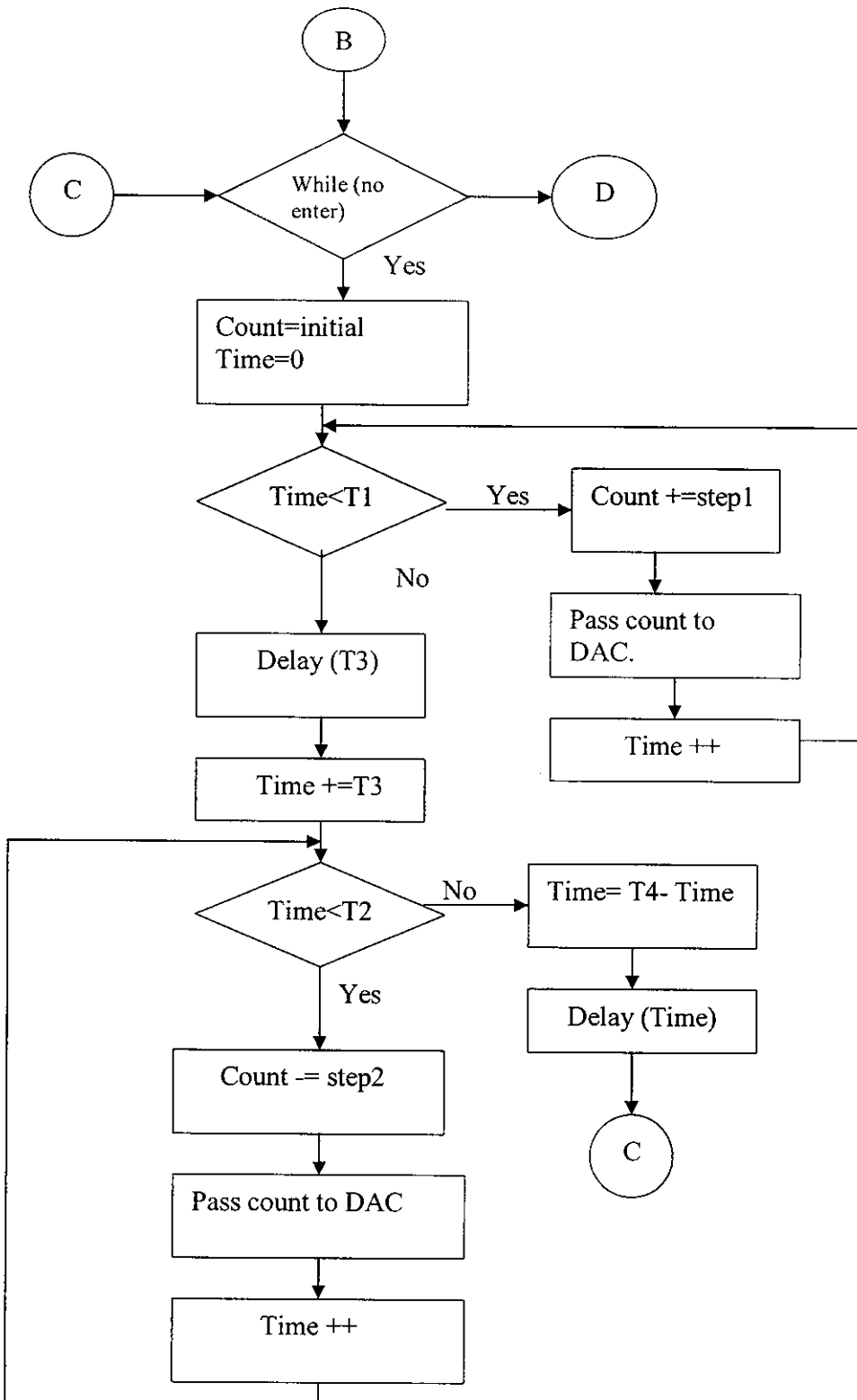
Step 18: Go to step2.

## **6.2 FLOW CHART OF RIG GENERATION**

The flow chart for Rig generation illustrates the coding logic behind the generation of the wave shape. It describes how inputs are got from the user; how step values for rise time and fall time are calculated and the generation of wave shape and passage to DAC







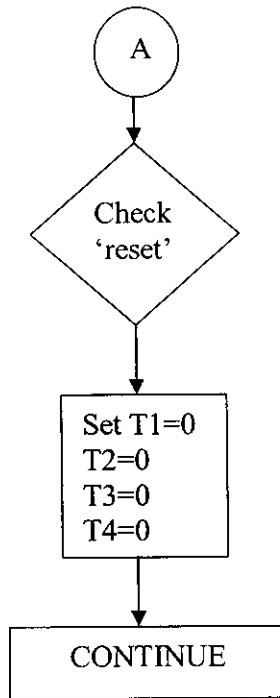
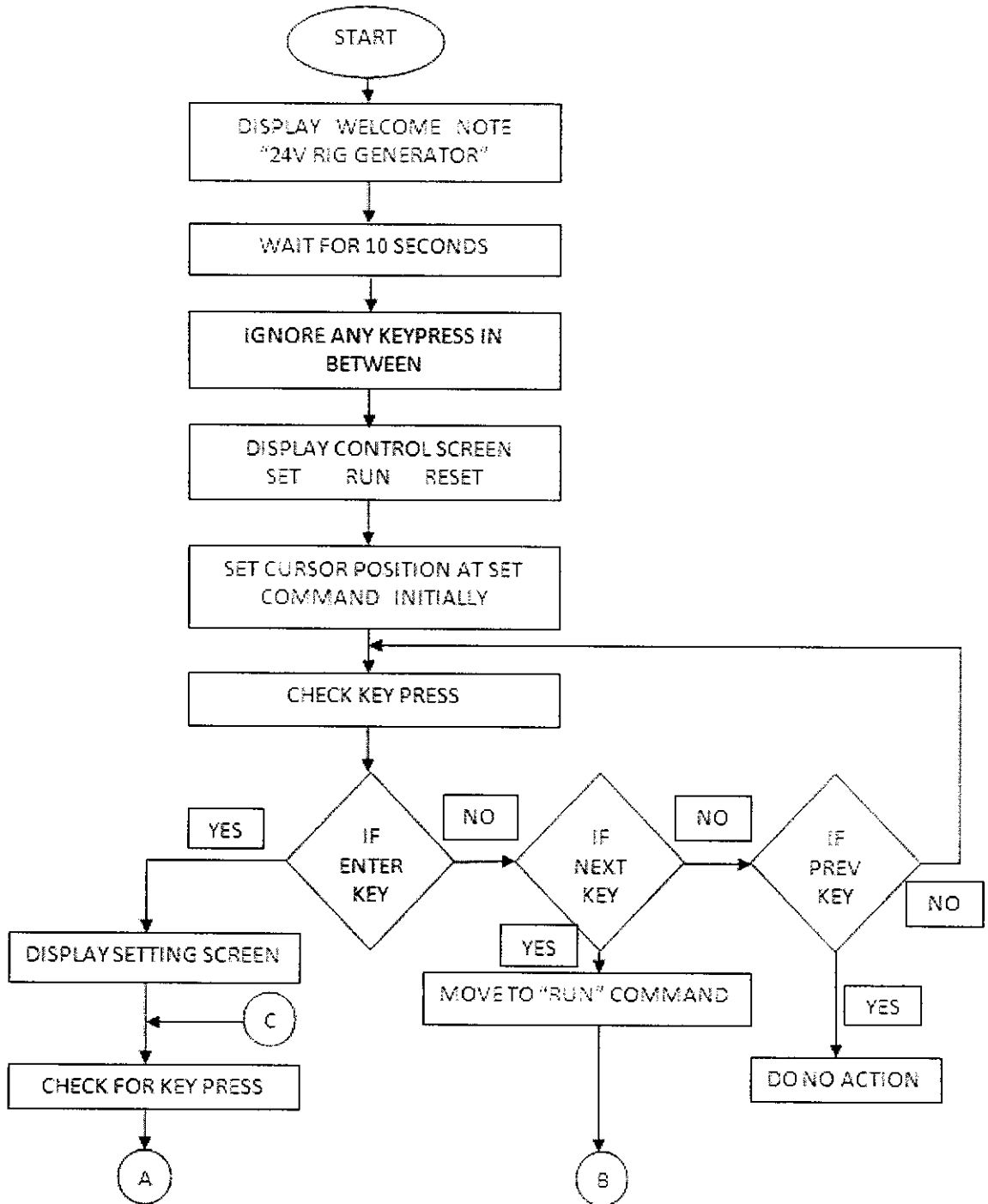


Figure 6.1  
Flowchart of Rig generation

### 6.3 FLOWCHART FOR LCD AND KEYBOARD FUNCTION

This flowchart is for the coding used in keyboard and display initialization. It describes the key pad validations and the screen transitions in getting the inputs from the user.



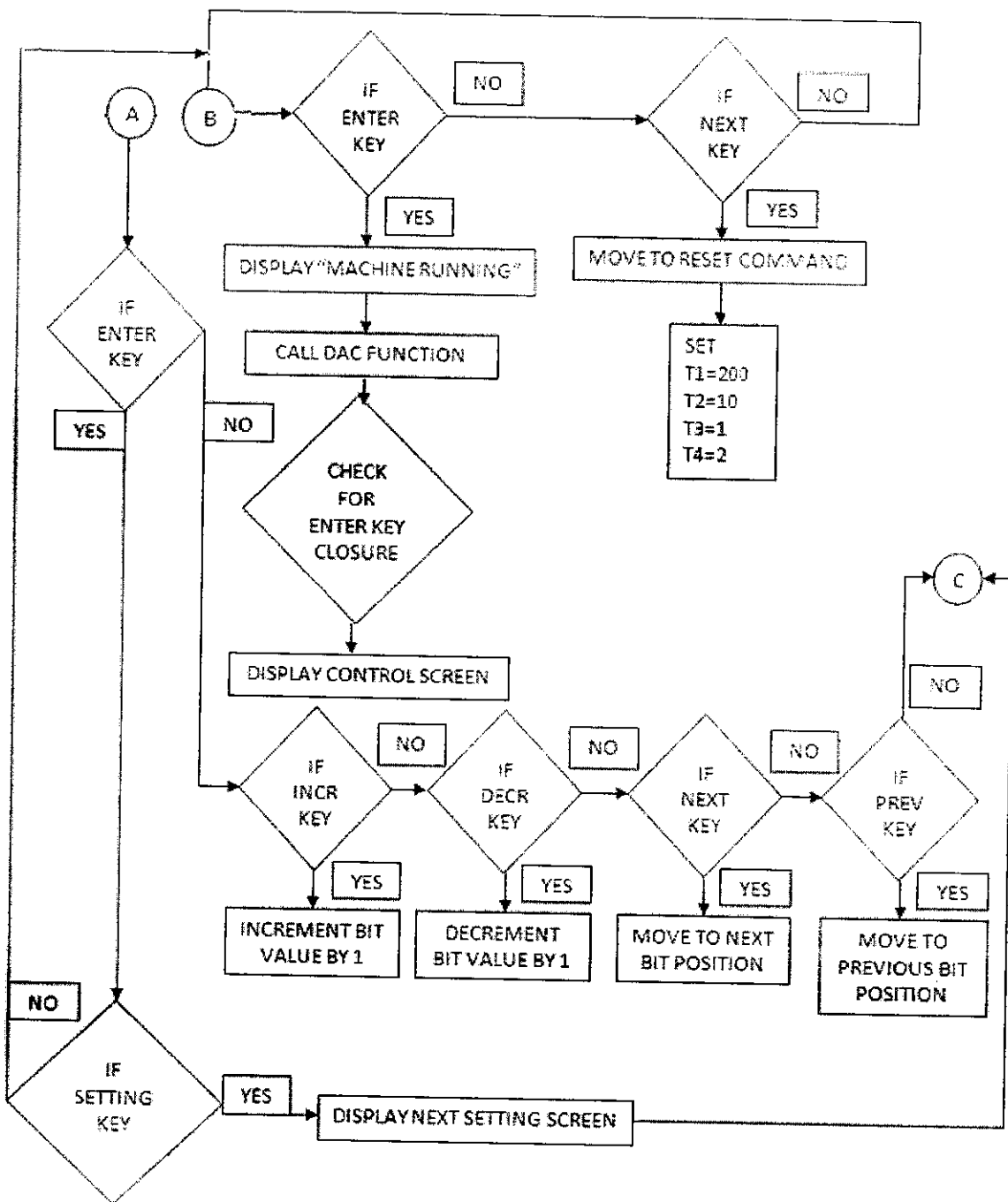


Figure 6.2  
Flow chart for LCD and keyboard function

## 6.4 MPLAB IDE

The programming is done using MPLAB IDE.

MPLAB IDE is a software program that runs on a PC to develop applications for Microchip microcontrollers. It is called an Integrated Development Environment, or IDE, because it provides a single integrated “environment” to develop code for embedded microcontrollers.

The MPLAB IDE project manager organizes the files to be edited and other associated files so they can be sent to the language tools for assembly or compilation, and ultimately to a linker. The linker has the task of placing the object code fragments from the assembler, compiler and libraries into the proper memory areas of the embedded controller, and ensures that the modules function with each other.

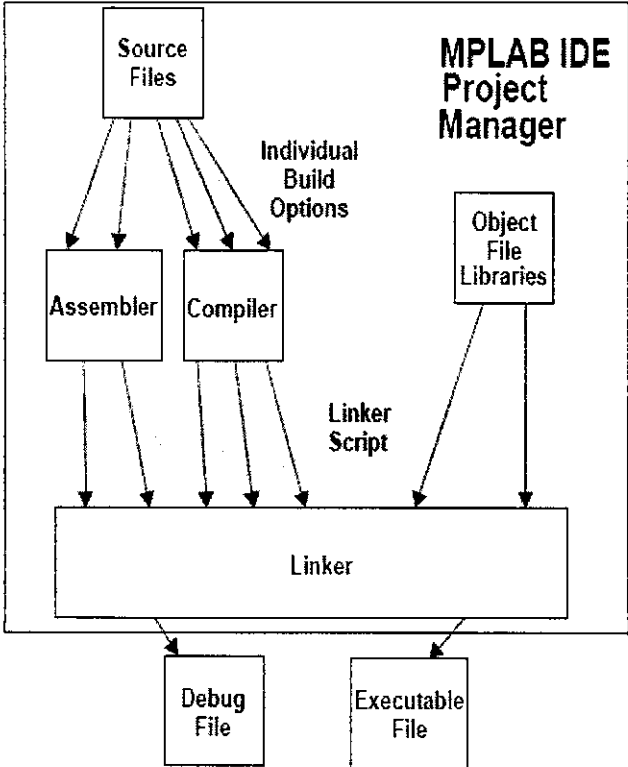


Figure 6.3  
MPLAB IDE Project Manager

The source files are text files that are written conforming to the rules of the assembler or compiler. The assembler and compiler convert them into intermediate modules of machine code and place holders for references to functions and data storage. The linker resolves these place holders and combines all the modules into a file of executable machine code. The linker also produces a debug file which allows MPLAB IDE to relate the executing machine codes back to the source files.

## **6.5 MPLAB C18**

Language tools are programs such as cross-assemblers and cross-compilers. The language tool used here is microchip C18 compiler. The MPLAB C18 compiler is a full-featured ANSI compliant C compiler for the PIC18 family of PICmicro 8-bit controllers. Like an assembler, the MPLAB C18 compiler translates human-understandable statements into ones and zeros for the microcontroller to execute. Code is written using standard ANSI C notation. Source code is compiled into blocks of program code and data which are then “linked” with other blocks of code and data, then placed into the various memory regions of the PIC18F4520 microcontroller. This process is called a “build,” and it is often executed many times in program development as code is written, tested and debugged. MPLAB C18 takes standard C statements, such as “if(x==y)” and “temp=0x27”, and converts them into PIC18F4520 machine code.

## **6.6 DEVICE PROGRAMMING**

MPLAB ICD 2 is a low cost, real-time debugger and programmer. Using Microchip Technology's proprietary In-Circuit Debug functions, programs can be downloaded, executed in real time and examined in detail with the debug functions of MPLAB. Set watch variables and breakpoints from symbolic labels in C or assembly source code, and single step through C source lines or into assembly code. MPLAB ICD 2 can also be used as a development programmer for supported MCUs. The secret behind In Circuit Debugging is two dedicated hardware lines (microcontroller pins used only during debugging mode) that control In Circuit Serial Programming (ICSP) of the device and, afterwards, debugging through proprietary, on-chip firmware.

The ICD 2 debug features are built into the microcontroller and activated by programming the debug code into the target processor. There is some shared overhead expense that includes one stack level, some general purpose file registers and a small area of program memory when in the debug mode

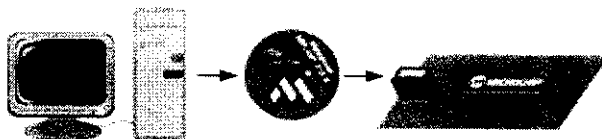
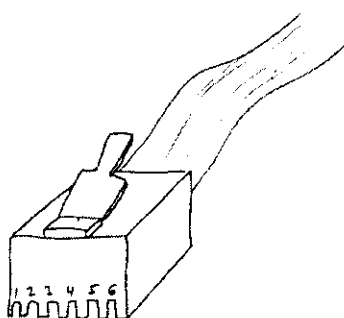


Figure 6.4 – MPLAB ICD 2

The MPLAB ICD 2 connects using USB or RS-232 between the PC operating with MPLAB IDE and the product board being developed. It acts as an intelligent interface/translator between the two, allows us to look into the active target board's microcontroller, viewing variables and registers at breakpoints with MPLAB watch windows. A breakpoint can be set to halt the program at a specific location. The program can be single-stepped or run at full speed. At breakpoints, data and program memory can be read and modified. Additionally, the MPLAB ICD 2 can be used to program or reprogram the Flash-based microcontroller while installed on the board.

### 6.6.1 ICD2 Connector Pin out



ICD2 Connector Pin out

Figure 6.5

The standard ICD2 cable is wired so that the pins are flipped between the ends. In other words, pin 1 on one end is connected to pin 6 on the other end, pin 2 to pin 5, etc. The pin out of each end is:

Table 6.1- ICD 2 Pin Connections

Signal	ICD2 end pin	Target end pin
Vpp	6	1
Vdd	5	2
GND	4	3
PGD	3	4
PGC	2	5
not connected	1	6

- **GND**-Negative power input to the PIC and the ground reference for the remaining signals.
- **Vdd** - This is the positive power input to the PIC.
- **Vpp** - Programming mode voltage. This is connected to the MCLR pin of the target PIC
- **PGC** - Clock line of the serial data interface. This line swings from GND to Vdd and is always driven by the programmer. Data is transferred on the falling edge.
- **PGD** - Serial data line. The serial interface is bi-directional, so this line can be driven by either the programmer or the PIC depending on the current operation. In either case this line swings from GND to Vdd. A bit is transferred on the falling edge of PGC.



**CHAPTER 7**  
**RESULTS AND DISCUSSIONS**

## 7. RESULTS AND DISCUSSIONS

In this chapter the results obtained for a typical set of input values are discussed.

### 7.1 MICROCONTROLLER BASED 24V RIG GENERATOR

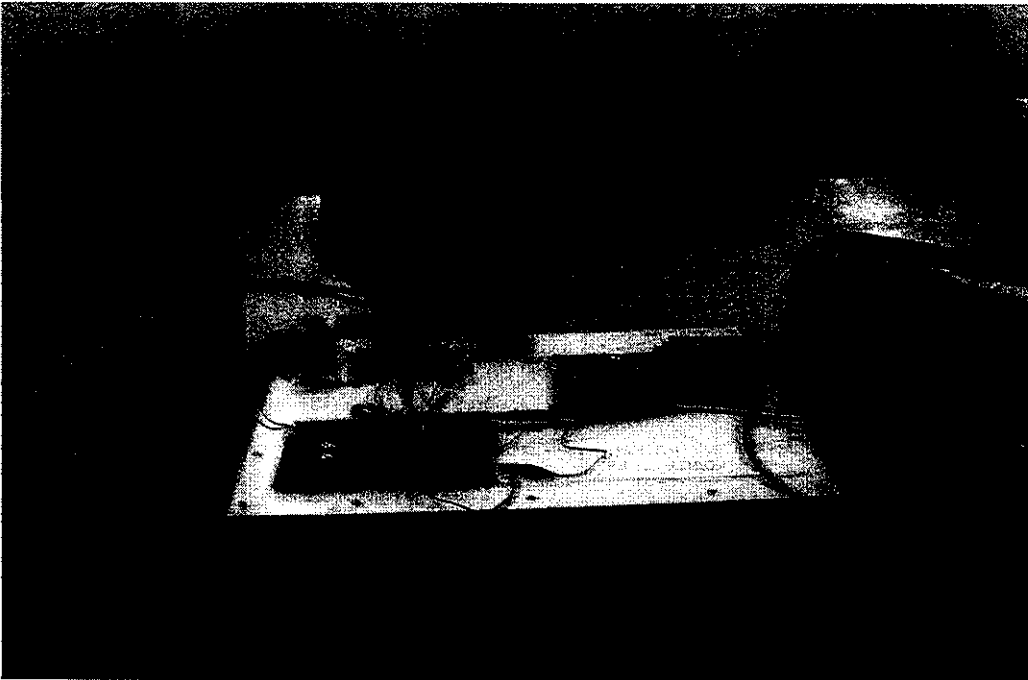


Figure 7.1 Microcontroller based 24V Rig generator

Input values for T, T1, T2, and T3:

Total time, T = 9.3s

Rise time, T1 = 300ms

Pulse width, T2 = 8.5s

Fall time, T3 = 300ms

For power supply Fluctuation Test, the typical offset voltage for all types of standards(DENSO,ISI,GERMAN,etc..) is  $11\pm 0.5V$  and peak value is  $26\pm 0.5V$

## 7.2 OUTPUT WAVEFORM

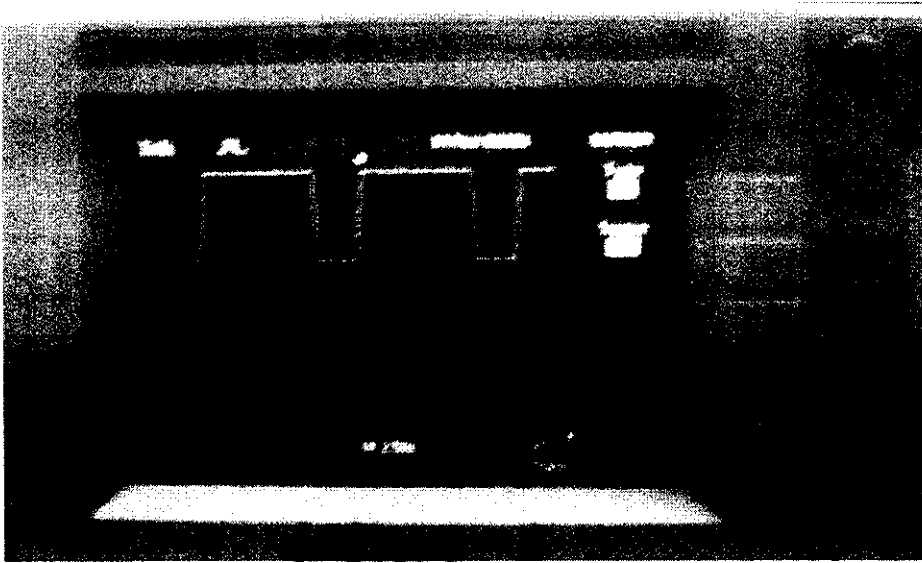


Figure 7.2 Output waveform

This picture shows the output from the output pin of the audio amplifier as measured in the Digital Oscilloscope. In the following pictures, cursors are selected to measure the magnitude of time period and voltage

## 7.3 OUTPUT WAVEFORM-TOTAL TIME PERIOD

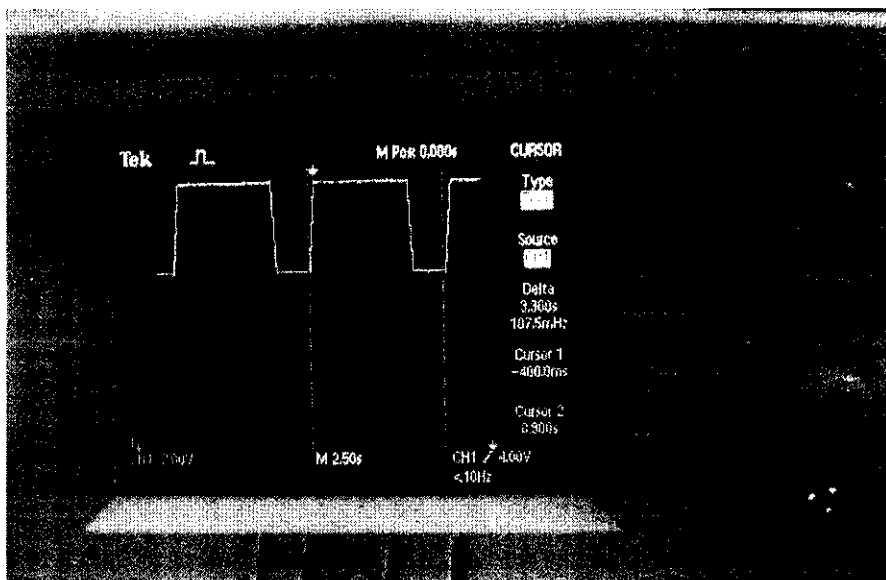


Figure7.3 Output waveform- total time period

This picture shows the total time period as measured with cursors. The cursor1 is at -400ms and cursor2 is at 8.9s. Thus the difference 9.3 s obtained agrees with the input total time: 9.3s

## 7.4 OUTPUT WAVEFORM-RISE TIME

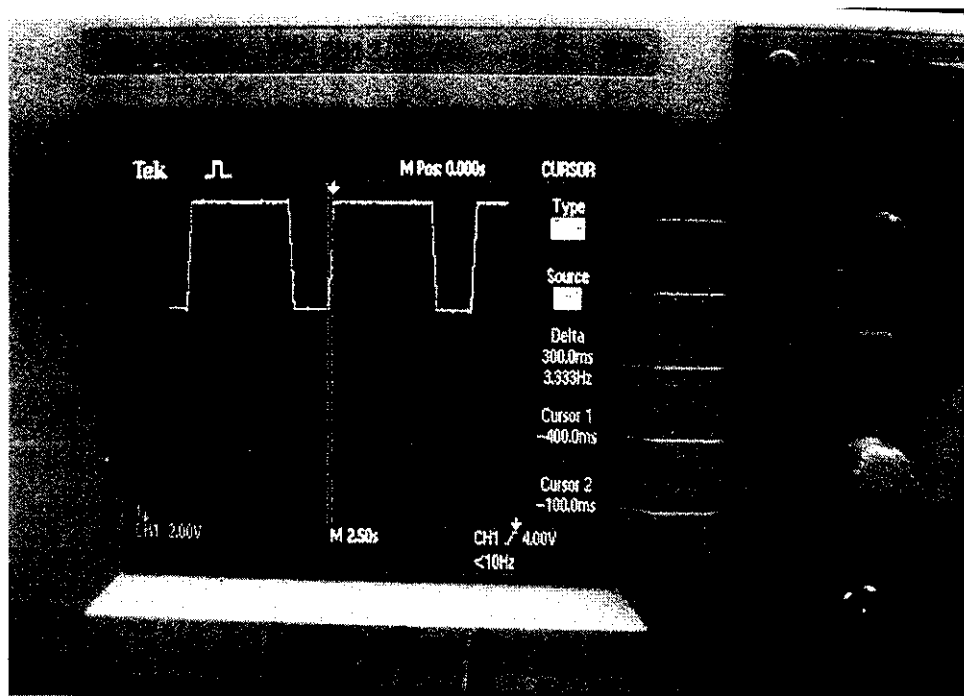


Figure 7.4 Output waveform- Rise time

This picture shows the Rise time as measured with cursors. The cursor1 is at -400ms and cursor2 at -100ms, thus the difference 300ms obtained agrees with the given input of  $T_1=300\text{ms}$

## 7.5 OUTPUT WAVEFORM-FALL TIME

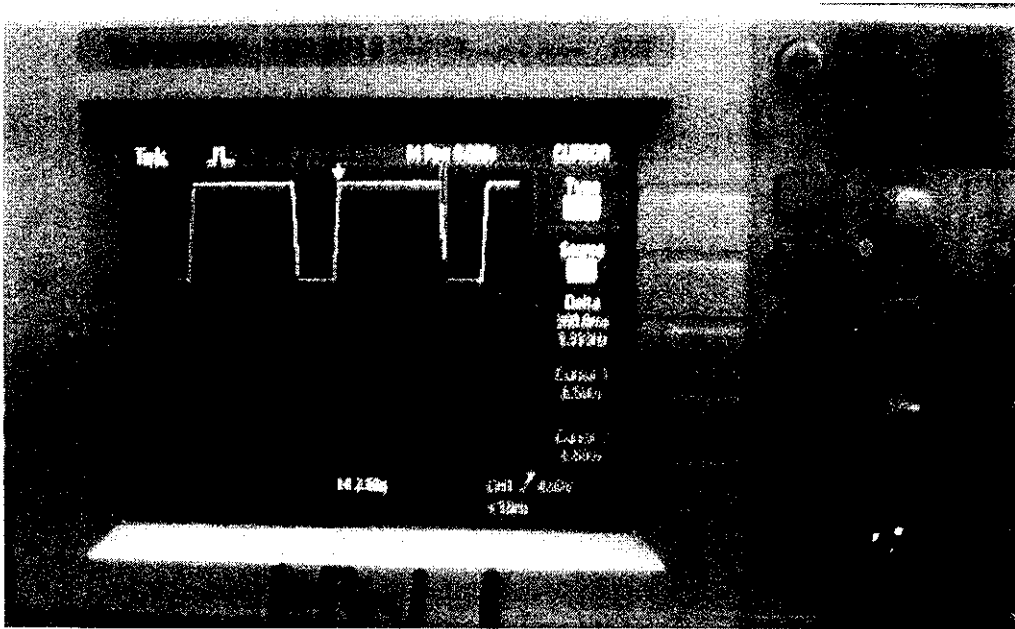


Figure 7.5 Output waveform- Fall time

This picture shows the Rise time as measured with cursors. The cursor1 is at 8.4s and cursor2 at 8.7, thus the difference 300ms obtained agrees with the given input of  $T_3=300\text{ms}$ . Also from the previous picture we had the cursor2 at -100ms. Thus the difference between the cursor positions 8.4s and -100ms is 8.5s which is the pulse width of the wave obtained. Thus the pulse width is also obtained accurately.

## 7.6 OUTPUT WAVEFORM- AMPLITUDE

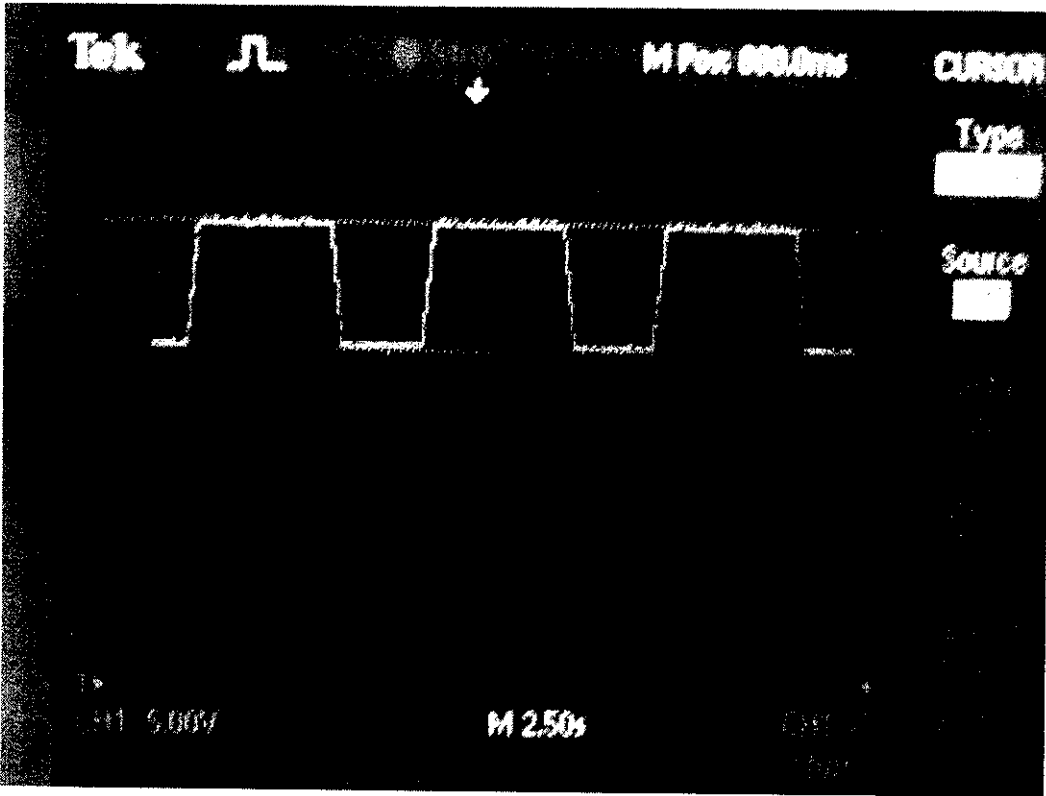


Figure 7.6 Output waveform-Amplitude

In the previous picture the time periods were measured using vertical cursors. Here horizontal cursors are chosen to measure the voltage magnitude. Here the wave is offset from the base by 11.6V. The peak value is 26.6V. Thus the difference is 15V which is according to the testing standard.

## **CHAPTER 8**

# **CONCLUSIONS AND RECOMMENDATIONS**

## **8. CONCLUSION AND RECOMMENDATIONS**

### **8.1 CONCLUSION:**

A Rig generator that can be used for power supply fluctuation tests provided under different standards has been designed. This Rig generator actually simulates the environment in which electrical noises (ie.)Transient voltages are generated.

Simulating the environment is a two-step process, first recognition of the noise environment - what kind of electrical noises are present and which of them are going to cause problems. The next stage is generation of the electrical noise in a controlled manner.

### **8.2 FUTURE SCOPE OF THE PROJECT:**

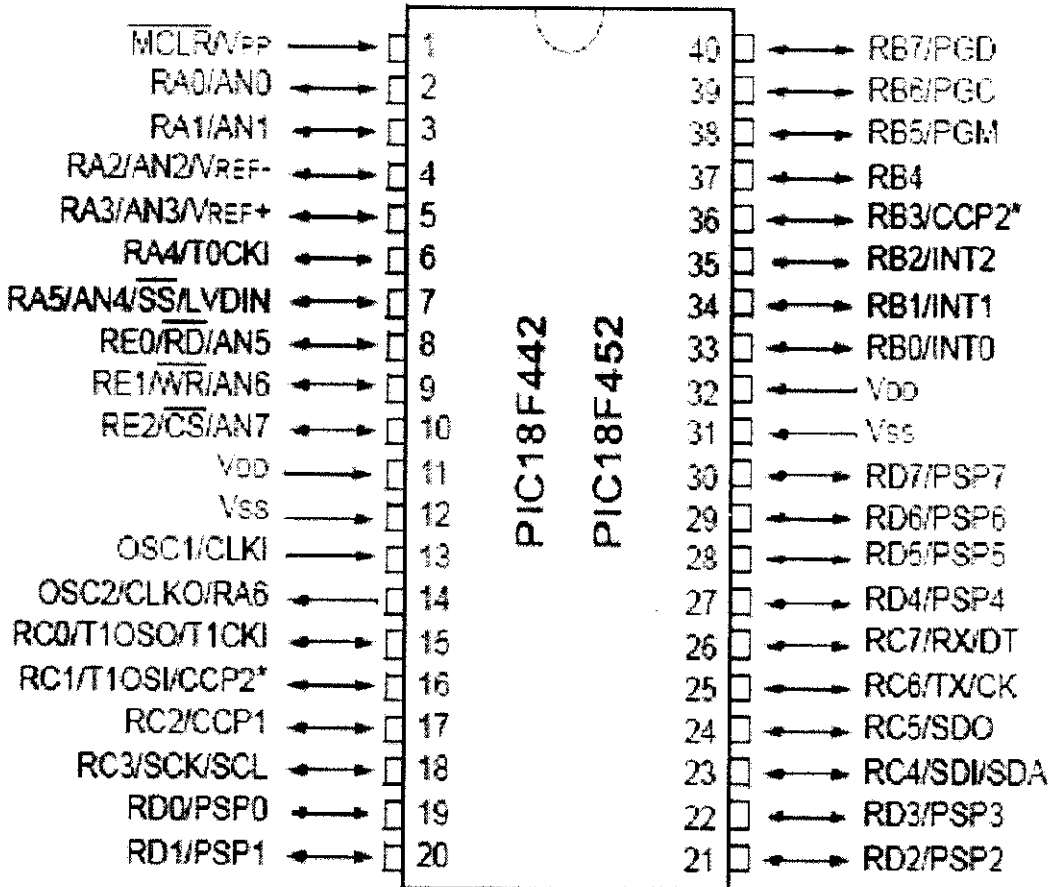
As an improvement of this project, in future, the voltage level can also be made variable. Moreover, this can be implemented on other tests like high frequency pulse test and momentary interruption power test. Memory slots may be provided in the apparatus to store some standard test specifications so as to avoid entering the values for time intervals every time.



**APPENDIX A**

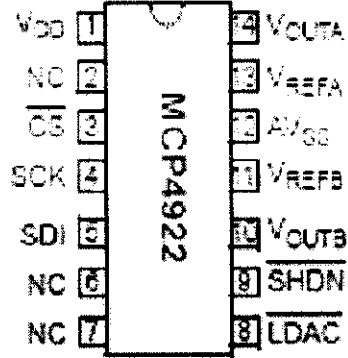
**PIN CONFIGURATIONS**

## A.1 PIC 18F452

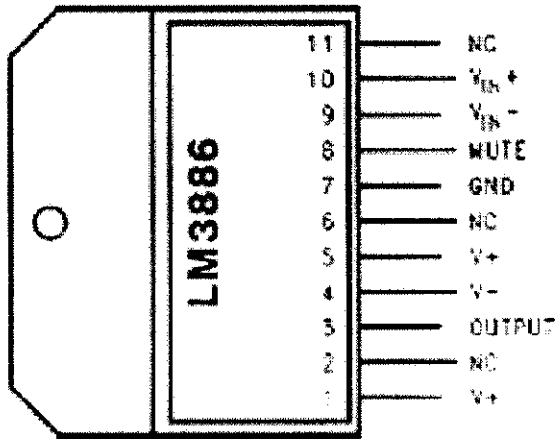


### A.2 MCP 4922:

14-Pin PDIP, SOIC, TSSOP



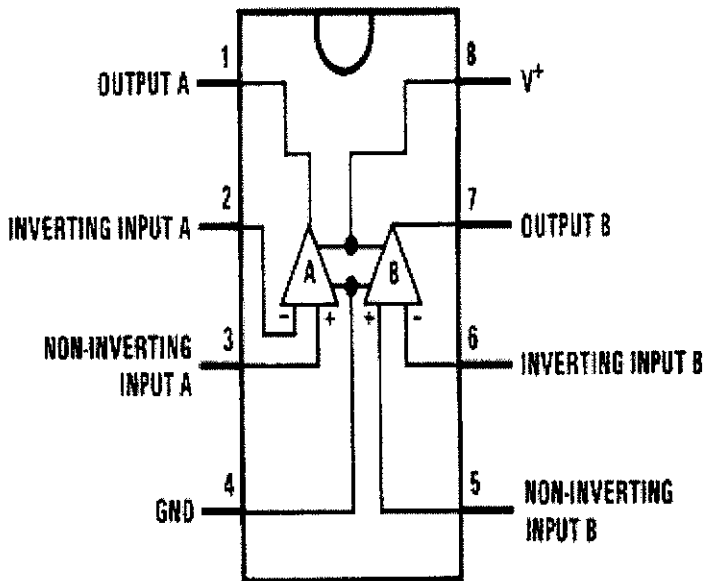
### A.3 LM3886:



TL74711833-2

Top View

## A.4 LM 358:



**APPENDIX B**  
**CODING FOR THE MICROCONTROLLER**

## CODING FOR THE MICROCONTROLLER

### define.h

```
#include <p18f452.h>

typedef unsigned char U1;

#define LCD_RS_LO (PORTBbits.RB5 = 0)
#define LCD_RS_HI (PORTBbits.RB5 = 1)

#define LCD_ENABLE (PORTCbits.RC1 = 1)
#define LCD_DISABLE (PORTCbits.RC1 = 0)

#define LCD_RW_LO (PORTAbits.RA2 = 0)
#define LCD_RW_HI (PORTAbits.RA2 = 1)

#define KEY_ENTER PORTEbits.RE2
#define KEY_NEXT PORTEbits.RE0
#define KEY_PREVIOUS PORTAbits.RA3
#define KEY_INCREMENT PORTAbits.RA4
#define KEY_DECREMENT PORTEbits.RE1
```

```
U1 header1[] = " 24 VOLTS TEST ";
U1 header2[] = " RIG GENERATOR ";
```

```
U1 time0[] = " T1:000.0 ms ";
U1 time1[] = " T2:000.0 ms ";
U1 time2[] = " T3:000.0 s ";
U1 time3[] = " T :000.0 s ";
```

```
U1 control[] = " CONTROL ";
U1 control1[] = "SET RUN RESET ";
U1 dsetting[] = " SETTING ";
U1 run1[] = " MACHINE ";
U1 run2[] = " RUNNING... ";
```

```
void printf(U1 ch,U1 *c)
```

```
{
    U1 count;

    LCD_RS_LO;
    delay(8);
    LCD_RW_LO;
    delay(8);
    PORTD=ch;
    delay(8);
    LCD_ENABLE;
    delay(8);
    LCD_DISABLE;
    delay(8);
    LCD_RS_HI;
```

```

delay(8);
LCD_RW_LO;
for(count = 0;count < 16;count++)
{
    LCD_ENABLE;
    delay(8);
    PORTD = *c;
    LCD_DISABLE;
    delay(8);
    c++;
}
}

```

```

void Display(void)

```

```

{
    printf(0x80,disbuff1);
    printf(0xC0,disbuff2);
}

```

```

U1 Key_Check(void)

```

```

{
    U1 key;

    if(!KEY_ENTER)
    {
        key = ENTER_KEY;
    }
    else if(!KEY_INCREMENT)
    {
        key = INCREMENT_KEY;
    }
    else if(!KEY_DECREMENT)
    {
        key = DECREMENT_KEY;
    }
    else if(!KEY_NEXT)
    {
        key = NEXT_KEY;
    }
    else if(!KEY_PREVIOUS) //inc
    {
        key = PREVIOUS_KEY;
    }
    else
    {
        key = 0;
    }
    return key;
}

```

## Main.c

```
#include "Timer.h"
#include "Global.h"
#include "lcd.h"
#include "Setting.h"

void FrontScreen(void);
void Value (U2 dacvalue);
void InitParam(void);

void Calculation(void);
void WaveGen(UF4 step3,UF4 step4);
void Rise (UF4 step);
void Fall(UF4 step);

U1 ssetcvar;
U2 step1,step2,count, i,j;

#define VALDAC1_CS PORTCbits.RC2

void main(void)
{
    U1 setdiscur;

    InitInterrupt();
    InitTimer();

    InitializeLcd();
    delay(200);

    FrontScreen();

    u1_newcursor = 0;
    sec          = 0;

    set_rise_time = ReadEeprom(ADDR_TIME1,2);
    set_fall_time = ReadEeprom(ADDR_TIME2,2);
    set_width_time = ReadEeprom(ADDR_TIME3,2);
    total_time    = ReadEeprom(ADDR_TIME4,2);

    if(set_rise_time > 999)
    {
        set_rise_time = 0;
    }
    if(set_fall_time > 999)
    {
        set_fall_time = 0;
    }
    if(set_width_time > 999)
```



```

{
    set_width_time = 0;
}
if(total_time > 999)
{
    total_time = 0;
}

for(setdiscur = 0;setdiscur < 16;setdiscur++)
{
    disbuff1[setdiscur] = control[setdiscur];
}

for(setdiscur = 0;setdiscur < 16;setdiscur++)
{
    disbuff2[setdiscur] = control1[setdiscur];
}

Display();

SetCursor(setlcddef[0]);

while(TRUE)
{
    if(keyscan == 1)
    {
        keyscan = 0;
        Setting();
    }
}
}

```

```

void Value (U2 dacvalue)

```

```

{
    U1 ahiger;
    U1 athiger;
    U1 alower;

    ahiger = 0;
    athiger = 0;
    alower = 0;

    athiger = ((U1 *) &dacvalue)[1];

    ahiger = 0x70 | athiger;
}

```

```

alower = ((U1 *) &dacvalue)[0];

SSPSTAT = 0x80;           //configuration register
SSPCON1 = 0x30;

VALDAC1_CS = 0;

SSPBUF = ahiger;
while( PIR1bits.SSPIF != 1);    // Check for end of transmission
PIR1bits.SSPIF = 0;

SSPBUF = alower;
while( PIR1bits.SSPIF != 1);
PIR1bits.SSPIF = 0;

SSPCON1 = 0x00;

VALDAC1_CS = 1;
}

void FrontScreen(void)
{
    U1 tmpcvar;

    for(tmpcvar = 0;tmpcvar < 16;tmpcvar++)
    {
        disbuff1[tmpcvar] = header1[tmpcvar];
    }
    for(tmpcvar = 0;tmpcvar < 16;tmpcvar++)
    {
        disbuff2[tmpcvar] = header2[tmpcvar];
    }

    Display();
}

void Calculation()
{
    step1 = 3800 / set_rise_time;
    step2 = 3800 / set_fall_time;

    r_set_width_time = set_width_time * 50;

    total_width = r_set_width_time + set_rise_time;

    total_fall = total_width + set_fall_time;
}

```

```

initial = 1500;

r_total_time = total_time * 50;

WaveGen(step1,step2);
}

void WaveGen(UF4 step3,Uf4 step4)
{
    U1 discvar;

    u1_newcursor = 0;

    while(1)
    {
        count = initial;
        //count = 0;

        for(i = 0; i <= r_total_time;i ++)
        {
            keypress = Key_Check();

            if(keypress == ENTER_KEY)
            {
                for(discvar = 0;discvar < 16;discvar++)
                {
                    disbuff1[discvar] = control[discvar];
                }

                for(discvar = 0;discvar < 16;discvar++)
                {
                    disbuff2[discvar] = controll1[discvar];
                }
                u1_newcursor = 0;

                Display();
                SetCursor(setlcddef[u1_newcursor]);
                T1CONbits.TMR1ON = 1;
                KeyRel();
                return;
            }

            if(i <= set_rise_time)
            {
                Rise(step3);
            }
            if(i > set_rise_time && i <= total_width)
            {
                //do nothing
            }
        }
    }
}

```

```

                f(i > total_width && i <= total_fall)
                {
                    Fall(step4);
                }

                if(i == total_fall + 1)
                {
                    Valve1(0);
                }

                delay(1);
            }
        }
    }
}

```

void Rise (UF4 step)

```

{
    U2 tmpstep;
    tmpstep = (U2) count;

    Valve1(tmpstep);

    count = count+step;
}

```

void Fall(UF4 step)

```

{
    U2 tmpstep;

    tmpstep = (U2) count;

    Valve1(tmpstep);

    count = count-step;
}

```

void Set(void)

```

{
    U1 setcursor;
    U1 setcvar;

    setcursor = 0;

    for(setcvar = 0;setcvar < 16;setcvar++)
    {
        disbuff1[setcvar] = control[setcvar];
    }

    for(setcvar = 0;setcvar < 16;setcvar++)
    {

```

```

        isbuff2[setcvar] = control1[setcvar];
    }

Display();
SetCursor(setlcddef[setcursor]);
KeyRel();

do
{
    keypress = Key_Check();

    if(keypress == ENTER_KEY)
    {
        if(setcursor == 0)
        {
            set_rise_time = ReadEeprom(ADDR_TIME1,2);
            set_fall_time = ReadEeprom(ADDR_TIME2,2);
            set_width_time = ReadEeprom(ADDR_TIME3,2);
            total_time = ReadEeprom(ADDR_TIME4,2);

            if(set_rise_time > 9999)
            {
                set_rise_time = 0;
            }
            if(set_fall_time > 9999)
            {
                set_fall_time = 0;
            }
            if(set_width_time > 9999)
            {
                set_width_time = 0;
            }
            if(total_time > 9999)
            {
                total_time = 0;
            }

            Setting1();
        }

        else if(setcursor == 1)
        {
            Calculation();
        }
        else if(setcursor == 2)
        {
            set_rise_time = 0;
            set_fall_time = 0;
            set_width_time = 0;
        }
    }
}

```

```

        WritEeprom(ADDR_TIME1,set_rise_time,2);
        WritEeprom(ADDR_TIME2,set_fall_time,2);
        WritEeprom(ADDR_TIME3,set_width_time,2);
        WritEeprom(ADDR_TIME4,total_time,2);
    }
    else
    {
        }

    return;
}
else if(keypress == INCREMENT_KEY || keypress == NEXT_KEY)
{
    ++setcursor;

    if(setcursor > 2)
    {
        setcursor = 2;
    }

    SetCursor(setlcddef[setcursor]);
    KeyRel();
}
else if(keypress == DECREMENT_KEY || keypress == PREVIOUS_KEY)
{
    if(setcursor != 0)
    {
        --setcursor;
    }
    else
    {
        setcursor = 0;
    }

    SetCursor(setlcddef[setcursor]);
    KeyRel();
}
else
{
    //do nothing
}

}while(TRUE);
}

void InitInterrupt(void)
{

```

INTCON = 0xC0;

```

    TRISA = 0x18;
    TRISB = 0x00;
    TRISC      = 0b11010000;           // setting the ports to SPI
    TRISD      = 0x00;                 // chip select pin for adc & I2C module
    TRISE      = 0x0F;                 // chip select pin for adc & I2C module
    ADCON1     = 0x0F;
}

void InitTimer(void)
{
    PIE1bits.TMR1IE      = 1;

    T1CON = 0x30;
    TMR1H = 0xEC;
    TMR1L = 0x80;

    T1CONbits.TMR1ON = 1;
}

```

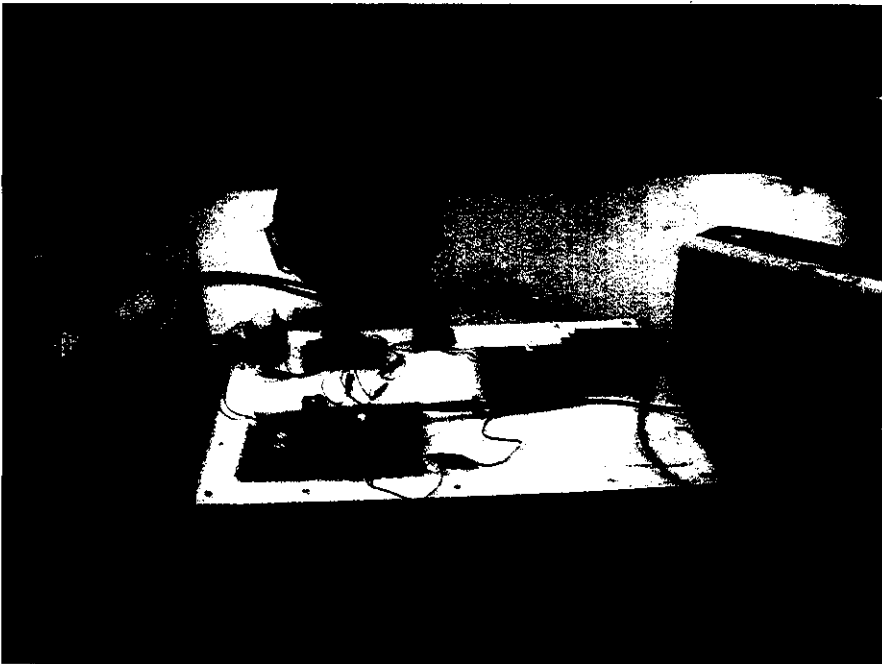
**APPENDIX C**  
**PHOTOS OF THE PROJECT**



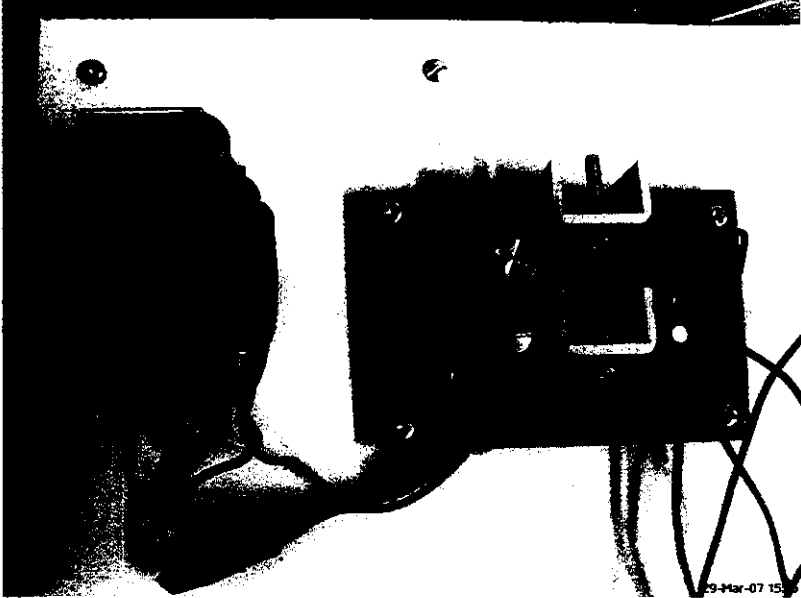
## C.1 DIGITAL PART



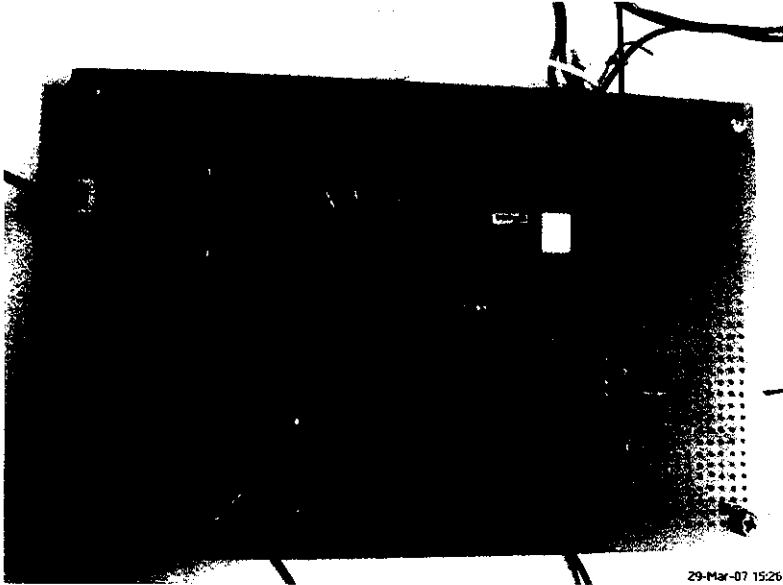
## C.2 RIG GENERATOR



# C.3 POWER SUPPLY CIRCIUT



# C.4 ANALOG PART



## REFERENCES

1. [www.microchip.com](http://www.microchip.com)
2. [www.datasheets4u.com](http://www.datasheets4u.com)
3. Testing standard-Denso Standard as provided by PRICOL INDUSTRIES
4. ‘Design with PIC controllers’ by John.B.Peatman, Prentice Hall;  
1st edition (August 8, 1997)
5. [http://webvia.techni-tool.com/VIA/NEWS/2007-03/MSO-Debug\\_PIC18.pdf](http://webvia.techni-tool.com/VIA/NEWS/2007-03/MSO-Debug_PIC18.pdf)
6. [http://www.maxim-ic.com/appnotes10.cfm/ac\\_pk/2](http://www.maxim-ic.com/appnotes10.cfm/ac_pk/2)
7. <http://www.rdrop.com/~cary/html/serialportdocs.html>
- 8.” PIC in Practice, Second Edition: A Project -based Approach” by David W Smith,  
Newnes; 2nd edition (January 27, 2006)