**CHAPTER 1**

**INTRODUCTION**

**1.1 INTRODUCTION**

Road accidents at the railway crossing are today a leading cause of death and injury worldwide. Surveys conducted by Indian Railway found that about 17% of total railway accidents in India is crossing accidents of which majority occurs at passive railway crossings. The operation of railway gates at level crossings is not so reliable nowadays. Primarily, the road users have to wait a very long time before the arrival of the train and even after the train left. And secondly the chances of accidents that usually made by the carelessness of the road users or due to the time errors made by the gatekeepers is more. Here comes the importance of automatic railway gate control system. In this project we detect the arrival of train and warn the road users about the arrival of train. After the obstacles are cleared, the gate is closed and train is passed .We will make sure that the train passes and the gate is reopened. The system deals with two things. Firstly, it deals with the reduction of time for which the gate is being kept closed. And secondly, provides safety to the road users by reducing the accidents. In the automatic railway gate control system, at the level crossing the arrival of the train is detected by the sensor placed near the gate. Hence, the time for which it is closed is less compared to the manually operated gates and also reduces the human labour and further the errors thus caused.

* 1. **AIM**

The main aim of this project is atomizing the unmanned railway gate. i.e., the gate closed automatically whenever the train comes and gate is opened after the train leaves the railway-road crossing. The arrival of the train in either directions can be identified using this project. The IR sensors placed on either side of the gate sense the arrival of train. The microcontroller allows dynamic and faster control. PIC12F683 microcontroller is the heart of the circuit as it controls all the functions.

* 1. **TECHNICAL SPECIFICATIONS**

Title of the project : Automatic railway gate control system

Domain : Embedded systems design

Software : Embedded C, MPlabX,

Microcontroller : PIC12F683

Hardware components : Resistors,

Capacitors,

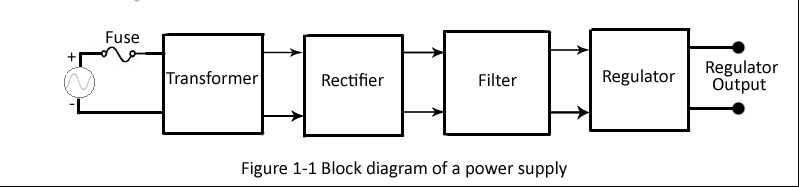
IR LEDs,

(TSOP1738),

L293D (Motor Driver)

Power Supply : +5v, 500ma Regulated Power supply

* 1. **BLOCK DIAGRAM**

****

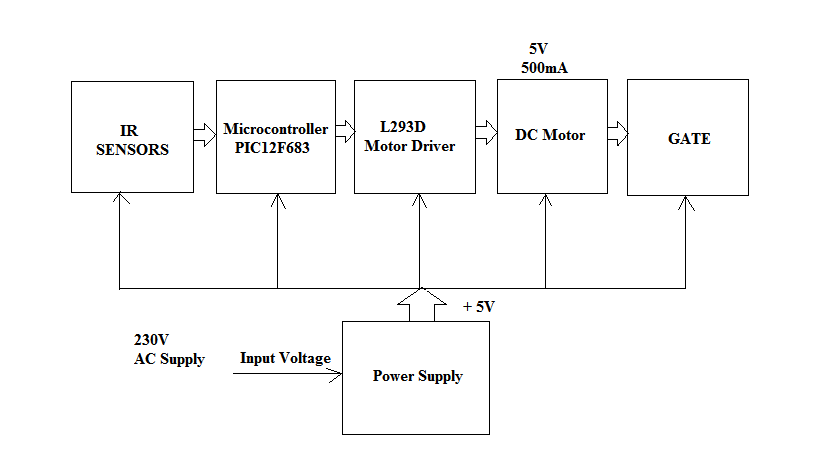
****

Figure 1.2 Block diagram

**1.5 NATIONAL STATUS**

The automatic railway gate controlling at the Railway crossing is implemented by using microcontroller based circuitry where the arrival of the train is detected by the IR sensors placed near to the gate. This type of gates can be employed in an unmanned level crossing where the chances of accidents are higher and reliable operation is required. Since, the operation is automatic, error due to manual operation is prevented. Automatic railway gate control is highly economical microcontroller based arrangement, designed for use in almost all the unmanned level crossings in the country.  
  
**Existing System**  
1. Manual / Physical gate closing & opening.  
2. Manual switch based gate closing & opening.  
**Limitations of exiting system**  
1. Chances of human error.  
2. Time consuming.  
3. Lot of human resource is required.  
**Features of Proposed System**  
1. The system will consist of 2 IR sensors pairs.  
2. Micro controller based circuit design.  
3. Automatic train sensing & gate controlling.  
4. Bidirectional gate controlling or Bidirectional train sensing.  
5. If required PC based GUI for better interface.  
6. The gate will be closed till the whole train passes out.  
7. The opening of gate will be sensor based not delay based.

**1.6 INTERNATIONAL STATUS**

Though these railway gate systems were brought into existence since a long period of time, the effective automatic implementation of these gates has not been achieved so far in many parts. To illustrate this case let us take the case of Asia. All [Indonesia](http://en.wikipedia.org/wiki/Indonesia) level crossings are operated automatically, but level crossings in [Thailand](http://en.wikipedia.org/wiki/Thailand), and [Malaysia](http://en.wikipedia.org/wiki/Malaysia) are still largely manually operated, where the barriers are lowered using a manual switch when trains approach. A significant number of crossings are without barriers. [Railway electrification in Malaysia](http://en.wikipedia.org/wiki/Railway_electrification_in_Malaysia) has gradually eliminated level crossings in [Peninsular Malaysia](http://en.wikipedia.org/wiki/Peninsular_Malaysia), replacing those along nearly all upgraded lines with large overhead viaducts or deep underground tunnels, and simply cutting off non-essential crossings outright.

These systems were also implemented across the world in several developed countries like, Belgium, Netherlands, United Kingdom, Norway, Germany, Japan, Taiwan etc. Statistics reveal that the scale of accidents that took place across the level crossing has decreased from around 97 in 1967 to around 6 in 2015 in USA because of the implementation. The ultimate aim of the project is to bring in a still further efficient unmanned automatic gate system that can avoid the 10% cases of accidents that have been occurring still.

**1.7 CONCLUSION**

The idea of automating the process of railway gate operation in level crossings has been undertaken. As the system is completely automated, it avoids manual errors and thus provides ultimate safety to road users. By this mechanism, presence of a gatekeeper is not necessary and automatic operation of the gate through the motor action is achieved. Microcontroller PIC12F683 performs the complete operation i.e., sensing, gate closing and opening operation is done by software coding written for the controller. The mechanism works on a simple principle and there is not much of complexity needed in the circuit. Moreover, due to the usage of TSOP1738, the system has become much reliable, dynamic and faster. This implementation is an advancement of the so far developed systems for automatic railway gating techniques.

**CHAPTER 2**

**LITERATURE SURVEY**

**2.1 PIC MICRO-CONTROLLER**

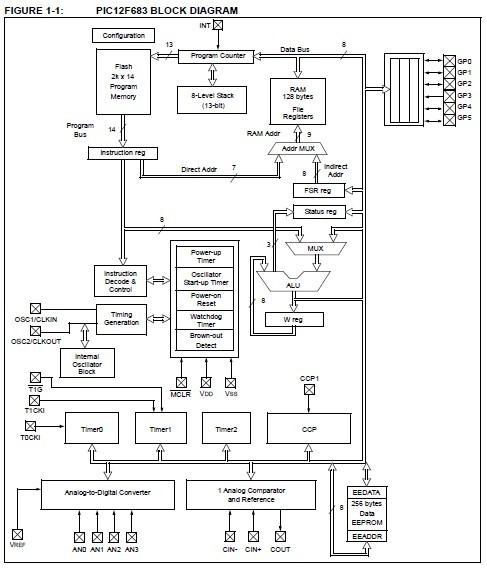
**2.1.1 GENERAL DESCRIPTION**

The PIC12F683 is a low power, high performance RISC 8-bit microcontroller with 2K bytes of Flash programmable memory. The device is manufactured by Microchip technologies. The on-chip flash allows the program memory to be reprogrammed in-system or by a conventional non-volatile memory programmer. The Microchip PIC12F683 is a powerful micro-controller, which provides a highly flexible and cost effective solution to many embedded control applications.

**2.1.2 FEATURES OF MICROCONTROLLER PIC12F683**

The PIC12F683 provides the following standard features: 2K bytes of Flash, 128 bytes of SRAM, 256 bytes of EEPROM, 6 I/O lines, 3 Timers-two of 8bit and one timer with 16bits, an Analog Comparator module, an Analog to digital converter with 10bit resolution and 4 channels, 8-level deep hardware stack, precision internal oscillator, power saving sleep mode, Wide operating voltage range, Power-on Reset (POR),Power-up Timer (PWRT) and Oscillator Start-up Timer (OST), Brown-out Reset (BOR) with software control option, Enhanced Low-Current Watchdog Timer (WDT) with on-chip oscillator (software selectable nominal 268 seconds with full pre-scaler) with software enable, Multiplexed Master Clear with pull-up, programmable code protection.

**2.1.3 PIC12F683 BLOCK DIAGRAM DESCRIPTION**



**2.1.3.1 Program Memory Organization**

The PIC12F683 has a 13-bit program counter capable of addressing an 8k x 14 program memory space. Only the first 2k x 14 (0000h-07FFh) for the PIC12F683 is physically implemented. Accessing a location above these boundaries will cause a wraparound within the first 2K x 14 space. The Reset vector is at 0000h and the interrupt vector is at 0004h.

**2.1.3.2 Data Memory Organization**

The data memory is partitioned into two banks, which contain the General Purpose Registers (GPR) and the Special Function Registers (SFR). The Special Function Registers are located in the first 32locations of each bank. Register locations 20h-7Fh in Bank 0 and A0h-BFh in Bank 1 are General Purpose Registers, implemented as static RAM. Register locations F0h-FFh in Bank 1 point to addresses 70h-7Fhin Bank 0. All other RAM is unimplemented and returns‘0’ when read. RP0 of the STATUS register is the bank select bit.

**2.1.4 TIMER0 MODULE**

When used as a timer, the Timer0 module can be used as either an 8-bit timer or an 8-bit counter. The Timer0 module is an 8-bit timer/counter with the following features:

• 8-bit timer/counter register (TMR0)

• 8-bit pre scaler (shared with Watchdog Timer)

• Programmable internal or external clock source

• Programmable external clock edge selection

• Interrupt on overflow

**2.1.5 TIMER1 MODULE WITH GATE CONTROL**

The Timer1 module is a 16-bit incrementing counter which is accessed through the TMR1H:TMR1L register pair. Writes to TMR1H or TMR1L directly update the counter.

When used with an internal clock source, the module is a timer. When used with an external clock source, the module can be used as either a timer or counter. It has the following features:

• 16-bit timer/counter register pair (TMR1H:TMR1L)

• Programmable internal or external clock source

• 3-bit prescalar

• Optional LP oscillator

• Synchronous or asynchronous operation

• Timer1 gate (count enable) via comparator or T1Gpin

• Interrupt on overflow

• Wake-up on overflow (external clock, Asynchronous mode only)

• Special Event Trigger (with CCP)

• Comparator output synchronization to Timer1 clock

**2.1.6 TIMER2 MODULE:**

The Timer2 module is an 8-bit timer with the following features:

• 8-bit timer register (TMR2)

• 8-bit period register (PR2)

• Interrupt on TMR2 match with PR2

• Software programmable pre scaler (1:1, 1:4, 1:16)

• Software programmable post scaler (1:1 to 1:16

**2.1.7 COMPARATOR MODULE**

Comparators are used to interface analog circuits to adigital circuit by comparing two analog voltages and providing a digital indication of their relative magnitudes. The comparators are very useful mixed signal building blocks because they provide analog functionality independent of the program execution. The analog comparator module includes the following features:

• Multiple comparator configurations

• Comparator output is available internally/externally

• Programmable output polarity

• Interrupt-on-change

• Wake-up from Sleep

• Timer1 gate (count enable)

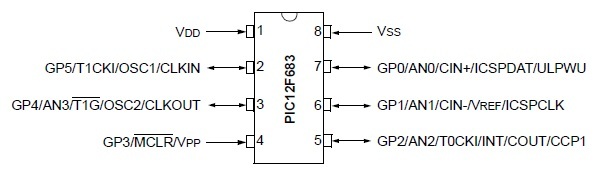
• Output synchronization to Timer1 clock input

• Programmable voltage reference

**2.1.8 ANALOG-TO-DIGITAL CONVERTER (ADC) MODULE**

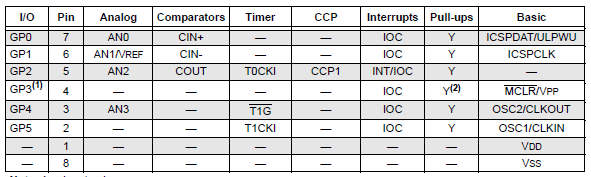
The Analog-to-Digital Converter (ADC) allows conversion of an analog input signal to a 10-bit binary representation of that signal. This device uses analog inputs, which are multiplexed into a single sample and hold circuit. The output of the sample and hold is connected to the input of the converter. The converter generates a 10-bit binary result via successive approximation and stores the conversion result into the ADC result registers (ADRESL and ADRESH).The ADC voltage reference is software selectable to either VDD or a voltage applied to the external reference pins. The ADC can generate an interrupt upon completion ofa conversion. This interrupt can be used to wake-up the device from Sleep.

**2.1.9 PIC12F683 PIN DIAGRAM**



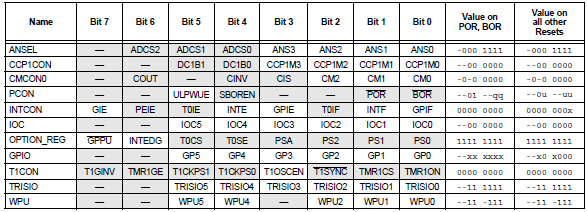
**Figure 2.2 Pin diagram of PIC12F683**

**2.1.10 PIN DESCRIPTION**

****

**2.1.11 GPIO PORT**

There are as many as six general purpose I/O pins available. Depending on which peripherals are enabled, some or all of the pins may not be available as general purpose I/O. In general, when a peripheral is enabled, the associated pin may not be used as a general purpose I/O pin.



**2.1.12 GPIO and the TRISIO Registers**

GPIO is a 6-bit wide, bidirectional port. The corresponding data direction register is TRISIO.

Setting a TRISIO bit (= 1) will make the corresponding GPIO pin an input (i.e., put the corresponding output driver in a High-Impedance mode). Clearing a TRISIO bit (= 0) will make the corresponding GPIO pin an output (i.e., put the contents of the output latch on the selected pin). An exception is GP3, which is input only and its TRISIO bit will always read as ‘1’. Example below shows how to initialize GPIO. Reading the GPIO register reads the status of the pins,whereas writing to it will write to the PORT latch. All write operations are read-modify-write operations. Therefore, a write to a port implies that the port pins are read, this value is modified and then written to the PORT data latch. GP3 reads ‘0’ when MCLRE = 1.The TRISIO register controls the direction of the GPIO pins, even when they are being used as analog inputs.

The user must ensure the bits in the TRISIO register are maintained set when using them as analog inputs. I/O pins configured as analog input always read ‘0’.

**Note:** The ANSEL and CMCON0 registers must be initialized to configure an analog channel as a digital input. Pins configured as analog inputs will read ‘0’.

**EXAMPLE: INITIALIZING GPIO**

BANKSEL GPIO;

CLRF GPIO ;Init GPIO

MOVLW 07h ;Set GP<2:0> to

MOVWF CMCON0 ;digital I/O

BANKSEL ANSEL ;

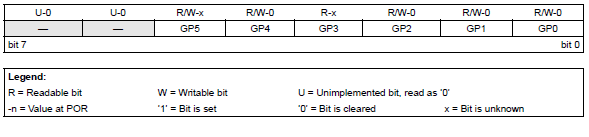
CLRF ANSEL ;digital I/O

MOVLW 0Ch ;Set GP<3:2> as inputs

MOVWF TRISIO ;and set GP<5:4,1:0>

;as outputs

**2.1.12.1 GPIO: GENERAL PURPOSE I/O REGISTER**



bit 7-6 **Unimplemented**: Read as ‘0’

bit 5-0 **GP<5:0>**: GPIO I/O Pin bit

1 = Port pin is > VIH

0 = Port pin is < VIL

**2.1.12.2 TRISIO GPIO TRI-STATE REGISTER**



**Legend:**

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as ‘0’

-n = Value at POR

‘1’ = Bit is set

‘0’ = Bit is cleared

x = Bit is unknown

bit 7-6 **Unimplemented**: Read as ‘0’

bit 5:4 **TRISIO<5:4>:** GPIO Tri-State Control bit

1 = GPIO pin configured as an input (tri-stated)

0 = GPIO pin configured as an output

bit 3 **TRISIO<3>:** GPIO Tri-State Control bit

Input only

bit 2:0 **TRISIO<2:0>:** GPIO Tri-State Control bit

1 = GPIO pin configured as an input (tri-stated)

0 = GPIO pin configured as an output

**Note 1:** TRISIO<3> always reads ‘1’.

**2:** TRISIO<5:4> always reads ‘1’ in XT, HS and LP OSC modes.

**3:** TRISIO<5> always reads ‘1’ in RC and RCIO and EC modes.

**2.1.13 ADDITIONAL PIN FUNCTIONS**

Every GPIO pin on the PIC12F683 has an interrupt-on-change option and a weak pull-up option.

GP0 has an Ultra Low-Power Wake-up option. The next three sections describe these functions.

**2.1.14 ANSEL REGISTER**

The ANSEL register is used to configure the Input mode of an I/O pin to analog. Setting the appropriate ANSEL bit high will cause all digital reads on the pin to be read as ‘0’ and allow analog functions on the pin to operate correctly. The state of the ANSEL bits has no affect on digital output functions. A pin with TRIS clear and ANSEL set will still operate as a digital output, but the Input mode will be analog. This can cause unexpected behavior when executing read-modify-write instructions on the affected port.

**2.1.14.1 ANSEL: ANALOG SELECT REGISTER**



**Legend:**

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as ‘0’

-n = Value at POR

‘1’ = Bit is set

‘0’ = Bit is cleared

x = Bit is unknown

bit 7 **Unimplemented**: Read as ‘0’

bit 6-4 **ADCS<2:0>:** A/D Conversion Clock Select bits

000 = FOSC/2

001 = FOSC/8

010 = FOSC/32

x11 = FRC (clock derived from a dedicated internal oscillator = 500 kHz max)

100 = FOSC/4

101 = FOSC/16

110 = FOSC/64

bit 3-0 **ANS<3:0>**: Analog Select bits

Analog select between analog or digital function on pins AN<3:0>, respectively.

1 = Analog input. Pin is assigned as analog input**(1)**.

0 = Digital I/O. Pin is assigned to port or special function.

**Note 1:** Setting a pin to an analog input automatically disables the digital input circuitry, weak pull-ups and interrupt-on-change,

if available. The corresponding TRIS bit must be set to Input mode in order to allow external control of the voltage on

the pin.

**2.1.15 WEAK PULL-UPS**

Each of the GPIO pins, except GP3, has an individually configurable internal weak pull-up. Control bits WPUx enable or disable each pull-up. Refer to Register fig.below. Each weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on a Power-on Reset by the GPPU bit of the OPTION register). A weak pull-up is automatically enabled for GP3 when configured as MCLR and disabled when GP3 is an I/O. There is no software control of the MCLR pull-up.

**2.1.15.1 WPU: WEAK PULL-UP REGISTER**



**Legend:**

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as ‘0’

-n = Value at POR

‘1’ = Bit is set

‘0’ = Bit is cleared

x = Bit is unknown

bit 7-6 **Unimplemented**: Read as ‘0’

bit 5-4 **WPU<5:4>:** Weak Pull-up Control bits

1 = Pull-up enabled

0 = Pull-up disabled

bit 3 **Unimplemented**: Read as ‘0’

bit 2-0 **WPU<2:0>:** Weak Pull-up Control bits

1 = Pull-up enabled

0 = Pull-up disabled

**Note 1:** Global GPPU must be enabled for individual pull-ups to be enabled.

**2:** The weak pull-up device is automatically disabled if the pin is in Output mode (TRISIO = 0).

**3:** The GP3 pull-up is enabled when configured as MCLR and disabled as an I/O in the Configuration Word.

**4:** WPU<5:4> always reads ‘1’ in XT, HS and LP OSC modes.

**2.1.16 OPTION\_REG: OPTION REGISTER**



**Legend:**

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as ‘0’

-n = Value at POR

‘1’ = Bit is set

‘0’ = Bit is cleared

x = Bit is unknown

bit 7 **GPPU:** GPIO Pull-up Enable bit

1 = GPIO pull-ups are disabled

0 = GPIO pull-ups are enabled by individual PORT latch values in WPU register

bit 6 **INTEDG:** Interrupt Edge Select bit

1 = Interrupt on rising edge of INT pin

0 = Interrupt on falling edge of INT pin

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

1 = Transition on T0CKI pin

0 = Internal instruction cycle clock (FOSC/4)

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:** Prescaler Assignment bit

1 = Prescaler is assigned to the WDT

0 = Prescaler is assigned to the Timer0 module

bit 2-0 **PS<2:0>:** Prescaler Rate Select bits

**2.2 EMBEDDED C**

**2.2.1 INTRODUCTION**

The microprocessor’s functions are controlled, guided and overseen by the embedded system software. Just like your computer is controlled by the Operating System (like Windows), your camera is controlled by the embedded software. The embedded software and embedded hardware form an embedded system**.**

Embedded C is the most popular embedded software language in the world. Most embedded software is written in Embedded C. Embedded C is very similar to C- if we know C, we won’t have a problem learning Embedded C. Embedded C takes it a step further and lets us write C like programs, suitable for the microprocessor environment.

**2.2.2 BASIC CONCEPTS OF EMBEDDED C**

Embedded C, even if it’s similar to C, and embedded languages in general requires a different kind of thought process to use. Embedded systems, like cameras or TV boxes, are simple computers that are designed to perform a single specific task. They are also designed to be efficient and cheap when performing their task. For example, they aren’t supposed to use a lot of power to operate and they are supposed to be as cheap as possible. As an embedded system programmer, we will have simple hardware to work with. We will have very little RAM, ROM and very little processing power and stack space. Our goal is to write programs that are able to leverage this limited processing power for maximum effect.

The reason why most embedded systems use Embedded C as a programming language is because Embedded C lies somewhere between being a high level language and a low level language. Embedded C, unlike low level assembly languages, is portable. It can run on a wide variety of processors, regardless of their architecture. Unlike high level languages, Embedded C requires less resources to run and isn’t as complex. Some experts estimate that C is 20% more efficient than a modern language like C++. Another advantage of Embedded C is that it is comparatively easy to debug.

**2.2.3 EMBEDDED C vs REGULAR C**

A major difference that drastically affects the structure of an Embedded C program and sets it apart from an ordinary C program is; when we write a regular C program, we access it from within our operating system software, run it and then, when we’re done, we exit back into our operating system. With an Embedded C program, we have no operating system to fall back on! Our program will, for all intents and purposes, act like the operating system for the embedded device.

Obviously, our program can never stop running, as this will cause the device it’s supposed to be operating to crash. Therefore, every Embedded C program must have a structured loop that keeps it running constantly. We can use a simple for loop or a while loop to do that. A normal Embedded C program will follow this format, for example:

void main ()

{

//initialize

while (condition) {

//keep doing this

}

}

As we know, every C program starts with a main declaration. It’s the same with Embedded C. The only difference is that an infinite loop will have to be included, and it will contain the most important parts of the code.

**2.3 IR SENSORS**

**2.3.1 INTRODUCTION**

Sensors are devices that measure various characteristics of an object of interest. They have so many industrial and consumer applications today that they have become necessary components of our daily lives. Most home appliances contain several and present-day automobiles will not work without them. Sensors make consumer products safer, more user friendly, and more functional. Sensors are vital to the functionality, performance, and distinctiveness of an expanding array of established and emerging industries and applications, including automotive , process control, consumer electronics and office and industrial automation products, machinery monitoring , air and ground transportation , environmental monitoring. Sensors are a key enabling technology that continues to find opportunities to allow for achieving new and promising, potentially high-volume applications and expanded sensor networks in industrial, commercial or homeland security applications.

However, the sensor’s signal must be conditioned and processed to be rendered suitable for use in the product, equipment, or system in which the sensor resides. For example, there are doors that automatically swing open when a person walks close by or passes a gate. In essence they operate as follows: The emitter passes an infrared beam which is detected by a phototransistor. When a person walks by, he “breaks” the beam. Upon this event, the phototransistor no longer can detect infrared light and another event is triggered-namely the door opens.

**2.3.2 DESCRIPTION**

Infrared (IR) radiation is part of the electromagnetic spectrum, which includes radio waves, microwaves, visible light, and ultraviolet light, as well as gamma rays and X-rays. The IR range falls between the visible portion of the spectrum and radio waves. IR wavelengths are usually expressed in microns, with the IR spectrum extending from 0.7 to 1000 microns.

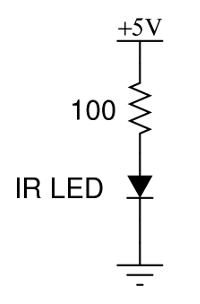
The detection section includes the IR transmitter and IR receiver to identify the trains on either side. Whenever the first detection circuit receives a signal the gate would be closed and will not allow anybody until another detection circuit receives a signal and also checks whether the specified time has been elapsed or not, if both the conditions are satisfied then only gate would be opened. For opening and closing the gate here we are using the motor.

Infrared sensors are essentially beam interruption detectors and can cope more readily with extremes of climate and temperature. It consists of just two components. The first is an Infrared (IR) transmitter (usually an LED), while the second is an Infrared (IR) receiver. Infrared light is invisible since its frequency is below that of visible red. Otherwise, it is like any other light source, operating under the same laws of physics. In most cases, the IR signals are produced by an LED source. IR transmit and receive systems are inexpensive and are gently reliable. However, interference from other sources is worth considering. Hence we go for modulation of the IR signal.

**2.3.3 MODULATED IR SIGNAL**

IR receiver can sense any IR radiation from surroundings or from any IR radiation emitting sources. In order to be specific and to maintain accurate response, the better solution would be to modulate the transmitted IR, and then have the receiver circuitry only to respond to the level of received, matching modulated IR signal. Here we use a carrier frequency of 36KHz.

**2.3.4 IR TRANSMITTER AND RECEIVER CIRCUIT DIAGRAM**

****

**Figure 3.1 Circuit diagram**

**2.3.5 IR TRANSMITTER**

Light Emitting Diodes are silicon devices that produce light. The light is produced only when current passes through in the forward direction. To produce light, the forward voltage must be higher than the diode’s internal barrier voltage.

Like any other diode, LEDs pass current in the forward direction, but block current in the reverse direction. This means the LED will only light up if connected with its cathode on the negative side of the circuit, and its anode on the positive side. Too much reverse voltage will destroy LEDs and diodes.

**2.3.6 GENERAL VIEW OF EMITTER**

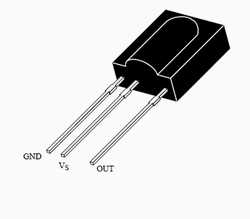


**Figure 3.2 LED**

It is important to note that once the voltage across the LED reaches +Vfthe diode conducts current extremely well. The LED can be easily destroyed by an excess of current. To protect the LED, a series current limiting resistor is added. The cathode side is usually marked with a flag spot on the flag that rings the body of the diode. The cathode wire is also shorter than the anode wire.

**2.3.7 IR RECEIVER**

The TSOP module is commonly found at the receiving end of an IR remote control system; e.g., in TVs, CD players etc. These modules require the incoming data to be modulated at a particular frequency and would ignore any other IR signals. It is also immune to ambient IR light, so one can easily use these sensors outdoors or under heavily lit conditions. Such modules are available for different carrier frequencies from 32 kHz to 42 kHz. In this particular proximity sensor, we will be generating a constant stream of square wave signal using PIC12F683 centred at 38 kHz and would use it to drive an IR led. So whenever this signal bounces off the obstacles, the receiver would detect it and change its output. Since the TSOP 1738 module works in the active-low configuration, its output would normally remain high and would go low when it detects the signal (the obstacle).



**Figure 3.3 TSOP**

**2.4 POWER SUPPLY**

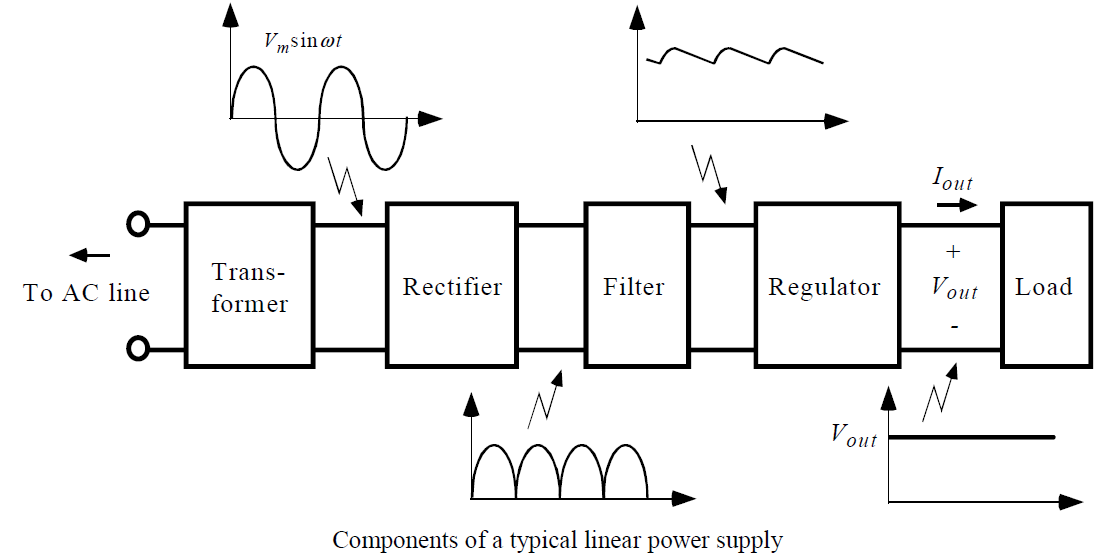
**2.4.1 DESCRIPTION**

A variable regulated power supply, also called a variable bench power supply, is one where you can continuously adjust the output voltage to your requirements. Varying the output of the power supply is the recommended way to test a project after having double checked parts placement against circuit drawings and the parts placement guide.

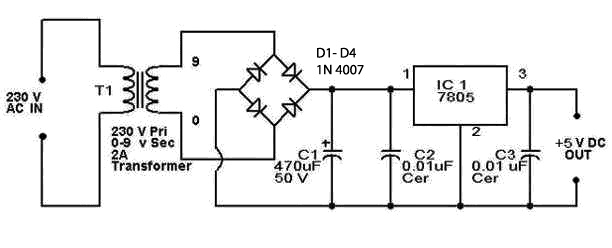
This type of regulation is ideal for having a simple variable bench power supply. Actually this is quite important because one of the first projects a hobbyist should undertake is the construction of a variable regulated power supply. While a dedicated supply is quite handy e.g. 5V or 12V, it’s much handier to have a variable supply on hand, especially for testing.

Most digital logic circuits and processors need a 5V power supply. To use these parts we need to build a regulated 5V source. Usually you start with an unregulated power supply ranging from 9V to 24V DC.

**2.4.2 BLOCK DIAGRAM**



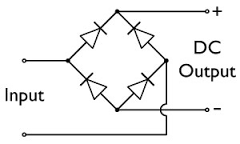
**2.4.3 CIRCUIT DIAGRAM**



The power will typically come from mains power supply. The power supply as a whole can be divided into two sections; the power source and the local regulator. If the regulator is properly designed and constructed, the power source is not that critical. The power source can be as simple as a transformer- rectifier-capacitor.

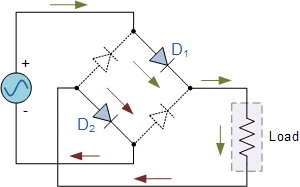
**2.4.4 BRIDGE RECTIFIER**

The bridge circuit is 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 a cost of a centre tapped transformer) but has two diode drops rather than one reducing efficiency over a centre tap based design for the same output voltage.

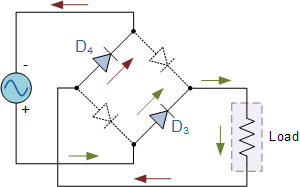


**2.4.5 BASIC OPERATION**

When the input connected at the left corner of the diamond is positive with respect to the one connected at the right hand corner, current flows to the right along the upper coloured path to the output, and returns to the input supply via the lower one.



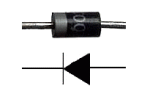
When the right hand corner is positive relative to the left hand corner, current flows along the upper coloured path and returns to the supply via the lower coloured path.



**2.5 1N4007 DIODES**

**2.5.1 SPECIFICATIONS**

1n4007 is a rectifier diode, 1amp, 1000volt, DO-41 package. It is a general purpose diode.



**Figure: Diode**

**2.5.2 FEATURES**

1. Low forward voltage drop

IF 1.0A

VF 1.1V

IR 6.0µA

Tj max. 150 ºC

1. Low leakage current
2. High forward surge capability

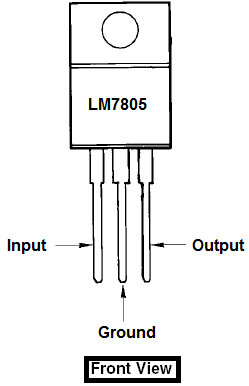
**2.5.3 APPLICATIONS**

1N4007 diodes are used in general purpose rectification of power supplies, inverters, converters and freewheeling diodes.

**2.6 LM7805 VOLTAGE REGULATOR**

**2.6.1 INTRODUCTION**

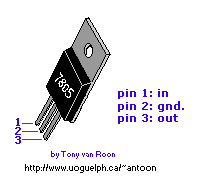
These are all linear regulators. They require input and output capacitors located close to the regulator in order to operate reliably. Without these capacitors or if they are too small or too far away, the regulator can oscillate at high frequencies depending on the load. A minimum of 100µF on the input and 10µF on the output always seem to work. Also include a series diode (usually a common 1N4007) on the input positive side. This prevents reverse polarity of the incoming supply damaging the regulator or other circuitry. The two elements of a stabilizer circuit, viz. voltage reference and voltage amplifier, can easily be combined into one IC, offering the advantages of extremely good regulation, compact size and ease to use, Many such IC regulators are designed for a specified fixed voltage output, such as 5V for logic circuits, or 15V for operational amplifiers. One such regulator is LM7805 IC.



To make a 5V power supply, we use this LM7085 voltage regulator IC. The LM7805 is simple to use. It consists of three pins. We can simply connect the positive lead of the unregulated DC power supply ( anything from 9V DC to 24V DC ) to the input pin (the left pin,1), connect the negative lead to the common pin and the centre pin(2) is connected to ground. When the power is turned on, we get a 5V power supply from the output pin (the right most pin 3).

**2.6.2 CIRCUIT FEATURES**

1. Gives out well regulated +5V output, output current capability of 100Ma
2. Circuit protection: Built-in overheating protection shuts down output when regulator IC gets too hot.
3. Circuit complexity: Very simple and easy to build
4. Very stable +5V output voltage, reliable operation
5. Availability of components: Easy to get, uses only very common basic components
6. Power supply voltage: Unregulated DC 8-18V power supply



**2.6.3 CONCLUSION**

The operation of the microcontroller must not be affected by the power supply. The power supply itself must be reliable and stable. The power supply should not cause problems during development. Put a series diode in the positive line before the input capacitor. This protects it when the plug-pack is connected the wrong way round (it happens). The input capacitor should be close to the regulator and at least 100µF at 25V. On the output side of the regulator you should have a 10µF capacitor.

IC power supplies don’t put out very smooth DC; it contains quite a lot of high frequency noise. A small linear 7805 series regulator will provide a better quality supply to our microcontroller.

**2.7 L293D MOTOR DRIVER**

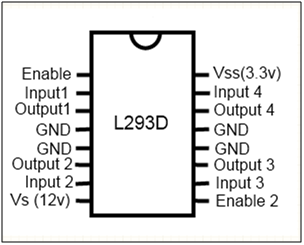
**2.7.1 INTRODUCTION**

L293D is a dual H-bridge motor driver integrated circuit (IC). Motor drivers act as current amplifiers since they take a low-current control signal and provide a higher-current signal. This higher current signal is used to drive the motors.

L293D contains two inbuilt H-bridge driver circuits. In its common mode of operation, two DC motors can be driven simultaneously, both in forward and reverse direction. The motor operations of two motors can be controlled by input logic at pins 2 & 7 and 10 & 15. Input logic 00 or 11 will stop the corresponding motor. Logic 01 and 10 will rotate it in clockwise and anticlockwise directions, respectively.

Enable pins 1 and 9 (corresponding to the two motors) must be high for motors to start operating. When an enable input is high, the associated driver gets enabled. As a result, the outputs become active and work in phase with their inputs. Similarly, when the enable input is low, that driver is disabled, and their outputs are off and in the high-impedance state.

**2.7.2 PIN DIAGRAM**

****

**2.7.3 PIN DESCRIPTION**

|  |  |  |
| --- | --- | --- |
| **Pin No** | **Function** | **Name** |
| 1 | Enable pin for Motor 1; active high | Enable 1,2 |
| 2 | Input 1 for Motor 1 | Input 1 |
| 3 | Output 1 for Motor 1 | Output 1 |
| 4 | Ground (0V) | Ground |
| 5 | Ground (0V) | Ground |
| 6 | Output 2 for Motor 1 | Output 2 |
| 7 | Input 2 for Motor 1 | Input 2 |
| 8 | Supply voltage for Motors; 9-12V (up to 36V) | Vcc 2 |
| 9 | Enable pin for Motor 2; active high | Enable 3,4 |
| 10 | Input 1 for Motor 1 | Input 3 |
| 11 | Output 1 for Motor 1 | Output 3 |
| 12 | Ground (0V) | Ground |
| 13 | Ground (0V) | Ground |
| 14 | Output 2 for Motor 1 | Output 4 |
| 15 | Input2 for Motor 1 | Input 4 |
| 16 | Supply voltage; 5V (up to 36V) | Vcc 1 |

**2.8 PROTEUS**

**2.8.1 INTRODUCTION**

Proteus developed by Lab centre electronics, is a software with which u can easily generate schematic captures, develop PCB and simulate microprocessor. It has such a simple yet effective interface that is simplifies the task to be performed. This one aspect has attracted many users to select this tool amongst many others offering the same services.

Proteus provides a powerful working environment. The user can design different electronic circuits with different electronic components easily accessible from the simple yet effective interface like signal generators, power supply, simple resistor or a different microcontroller or a microprocessor.

**2.8.2INTELLIGENT SCHEMATIC INPUT SYSTEM (ISIS)**

**ISIS** lies right at the heart of the **PROTUES** system and is far more than just another schematic package. It has powerful environment to control most aspects of the drawing appearance. Whether your requirement is the rapid entry of complex design for simulation & PCB layout, or the creation of attractive Schematic for publication **ISIS** is the right tool for the job product features

• Produces publication quality schematic  
• Style templates allow customization of supplied library  
• Mouse driven context sensitive user interface  
• Automatic wire routing and junction dot placement  
• Full support for buses including sub- circuit ports and bus pins  
• Large and growing component library of over 8000 parts  
  
**2.8.3 VSM (Virtual System Modelling)**

**Proteus VSM** is an extension of the PROSPICE simulator that facilities co-simulation of microprocessor based design including all the associated electronics. Furthermore, you can interact with the microcontroller software through the use of animated keypads, switches, buttons, LEDs, lamps and even LCD displays.

**Features**

1. CPU models available for many popular microcontrollers including PIC, AVR, HC11 and 8051
2. Interactive device models include LED and LCD displays, RS232 terminal, universal keypad plus a range of switches, buttons, pots, LEDs, 7 segment displays and more.
3. Extensive debugging facilities including register and memory contents, breakpoints and single step modes.
4. Source level debugging for selected development tools including IAR C-SPY and Keil µvision 2.

**2.9 PIC PROGRAMMER (PICPgm)**

**2.9.1 Introduction**

PICPgm is a PC-Software to program PIC microcontrollers using an external programmer hardware connected to the PC. The PICPgm Development Programmer Software is Freeware. It is available with a Graphical User Interface (GUI) and a Command Line interface.

It allows to:

1. Program a HEX file into a PIC microcontroller
2. Read the content of a PIC microcontroller and save it to a HEX file
3. Erase a PIC microcontroller
4. Check if a PIC microcontroller is empty, i.e. not programmed (Blank Check).

**2.9.2 System Requirements**

To use PICPgm, you need the following environment:

1. PC running Windows or Linux
2. PIC programmer
3. Parallel port (LPT), serial port (COM) or USB port (only supported USB programmer so far is PICPgm USB programmer)

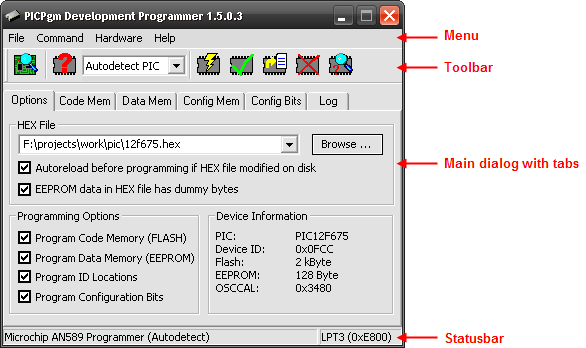
**Supported operating systems**

1. Windows 98/ME/2k/XP/Vista/7
2. Windows XP x64/Vista x64/7 x64
3. Linux (command line version of PICPgm only)
4. MacOS X (command line version of PICPgm only, experimental)

**2.9.3 PICPgm Graphical User Interface (GUI)**

The GUI version of the PICPgm Programmer is designed to be very simple and intuitive to use. It mainly consists of:

1. Menu
2. Toolbar
3. Main dialog with tabs
4. Statusbar



### 2.9.4 Menu Description

#### Table 2.9.4.1: File

|  |  |
| --- | --- |
|  |  |
| **Menu Item** | **Description** |
| Load HEX file ... | Load the HEX file which shall be programmed into the PIC. |
| Reload HEX file | Reload currently selected HEX file. |
| Exit | Quit PICPgm application. |

#### Table 2.9.4.2: Command

|  |  |
| --- | --- |
| **Menu Item** | **Description** |
| Auto detect Programmer | Automatically tries to find the PIC programmer hardware connected to your PC. |
| Auto detect PIC | Automatically tries to identify the PIC which is connected to the PIC programmer hardware. |
| Program PIC | Programs the content of the loaded HEX file into the PIC microcontroller. |
| Verify PIC | Reads the content of the PIC microcontroller and compares if it is equal to the loaded HEX file. |
| Read PIC and Save to File | Reads the content of the PIC microcontroller and compares if it is equal to the loaded HEX file. |
| Erase PIC | Erase the PIC flash memory. |
| Blank Check | Check if the PIC flash memory is blank, i.e. check if the PIC is empty. |

#### Table 2.9.4.3: Hardware

|  |  |
| --- | --- |
| **Menu Item** | **Description** |
| Hardware Selection/ Configuration | Manually select the programmer hardware to be used instead of auto-detecting it (useful if auto-detection fails). It also allows to change the programmer pin configuration. Further a hardware test dialog is implemented here to manually check if the programmer hardware is working properly. |

**Table 2.9.4.4 Tool bar description**

|  |  |  |
| --- | --- | --- |
| **Symbol** | **Name** | **Description** |
| PICPgm Toolbar Icon Autodetect Programmer Hardware | Auto-detect Programmer | Automatically tries to find the PIC programmer hardware connected to your PC. |
| PICPgm Toolbar Icon Autodetect PIC Microcontroller | Auto-detect PIC | Automatically tries to identify the PIC which is connected to the PIC programmer hardware. |
| PICPgm Toolbar Icon select PIC Microcontroller | Select PIC | Manually select PIC microcontroller (instead of auto detection).  Note: Some PICs cannot be automatically identified because they don't have a device ID. This PICs are marked with a "\*" at the end of the name. This PICs must be selected manually! |
| PICPgm Toolbar Icon Program PIC Microcontroller | Program PIC | Programs the content of the loaded HEX file into the PIC microcontroller. |
| PICPgm Toolbar Icon Verify PIC Microcontroller | Verify PIC | Reads the content of the PIC microcontroller and compares if it is equal to the loaded HEX file. |
| PICPgm Toolbar Icon Read PIC Microcontroller and Save to HEX file | Read PIC and Save to File | Reads the content of the PIC microcontroller and compares if it is equal to the loaded HEX file. |
| PICPgm Toolbar Icon Erase PIC Microcontroller | Erase PIC | Erase the PIC flash memory. |
| PICPgm Toolbar Icon PIC Microcontroller Blank Check | Blank Check | Check if the PIC fash memory is blank, i.e. check if the PIC is empty. |

#### Table 2.9.4.5: Extras

|  |  |
| --- | --- |
| **Menu Item** | **Description** |
| Clear Log Window | Clears the logging messages in the Log tab. |

#### Table 2.9.4.6: Help

|  |  |
| --- | --- |
| **Menu Item** | **Description** |
| Check for Updates | Checks if newer version PICPgm is available for download. |
| About | Provides information about PICPgm application (version number, author, build date). |

**2.10 MPLAB® X IDE**

**2.10.1 INTRODUCTION**

MPLAB® X IDE is a software program that runs on a PC (Windows®, Mac OS®, Linux®) to develop applications for Microchip microcontrollers and digital signal controllers. It is called an Integrated Development Environment (IDE), because it provides a single integrated "environment" to develop code for embedded microcontrollers.

**2.10.2 FEATURES:**

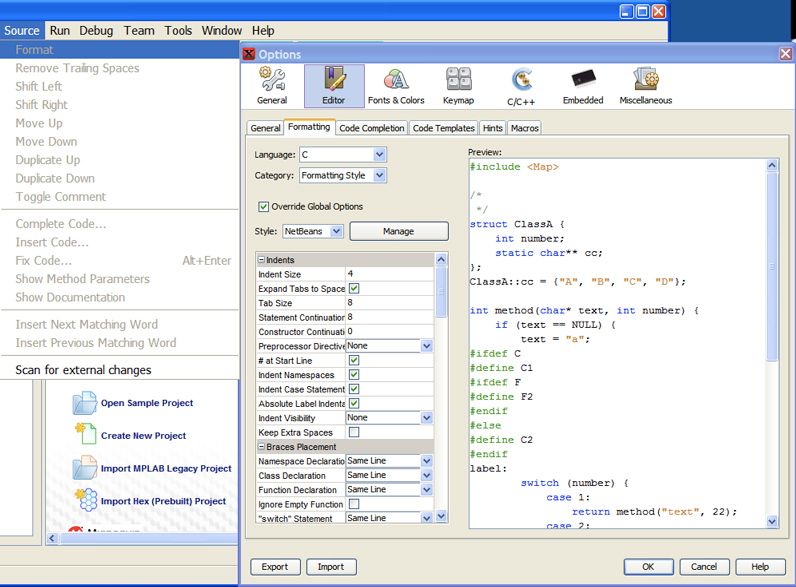
**1) Supports “one click”**

****

"One Click" Make, Program, Debug / Execute operation. Unlike other IDEs where you build, have to connect to the hardware tool, program the target and then start your debug session.

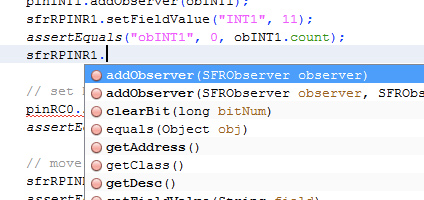
Under MPLAB® X it is all compiled into one action button. Run, Program, or Debug Run starts ‘Make’ which will check for changes and build any relevant updates, connect to the tool program the images and either start a debug session or start an execution of the programmed image. (There is also the option of doing these individually if the customer prefers).

**2) Within MPLab X IDE the user can configure their own Code Format Style**

****

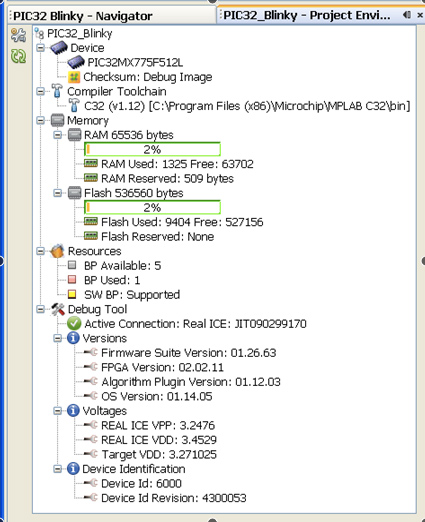
Either an individual or a company can set up a code format standard to be used within the editor. Just select the file to format the code in and menu Source>/Format to reapply the template to your source code.

**3) Provides Auto-compilation:**



Save time entering programs, and ensure correctness using the MPLAB® X editor’s auto-completion features.

**4) The status of the project is well defined**

****

When we need one place to summarize you project, For convenience there is a single window that gathers all the relevant project information and its environment. Device name, Debug Tool, Language Build Tool, and Connection state are presented. The Memory section shows Total, Used and Reserved by Debug Tool for RAM and Flash memory. Checksum and Breakpoint (silicon resource) status is also shown. The Debug tool provides additional status for Device ID, firmware versions and voltages.

**2.11 DC MOTOR**

**2.11.1 Introduction**

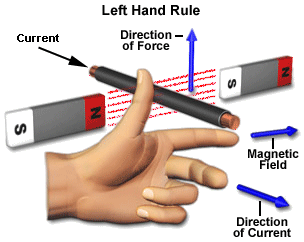
A **DC motor** is any of a class of electrical machines that converts direct current electrical power into mechanical power. The most common types rely on the forces produced by magnetic fields. Nearly all types of DC motors have some internal mechanism, either electromechanical or electronic, to periodically change the direction of current flow in part of the motor. Most types produce rotary motion; a linear motor directly produces force and motion in a straight line.



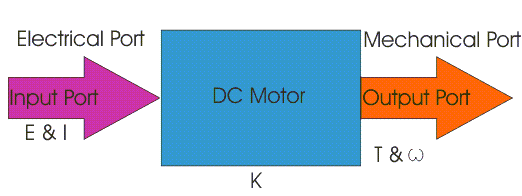
**Figure: DC motor**

**2.11.2 Principle of DC motor**

This DC or direct current motor works on the principle when a current carrying conductor is placed in a magnetic field, it experiences a torque and has a tendency to move. This is known as motoring action. If the direction of current in the wire is reversed, the direction of rotation also reverses. When magnetic field and electric field interact they produce a mechanical force, and based on that the working principle of DC motor established. The direction of rotation of this motor is given by Fleming’s left hand rule, which states that if the index finger, middle finger and the thumb of our left hand are extended mutually perpendicular to each other and if the index finger represent the direction in which force is experienced by the shaft of the DC motor.



Structurally and construction wise a direct current motor is exactly similar to a DC generator, but electrically it is just the opposite. Here, unlike a generator we supply electrical energy to the input port and derive mechanical energy from the output port. We can represent it by the block diagram shown below.



Here in a DC motor, the supply voltage E and current is given to the electrical port or the input port and we derive the mechanical output i.e torque T and speed ω from the mechanical port or output port.

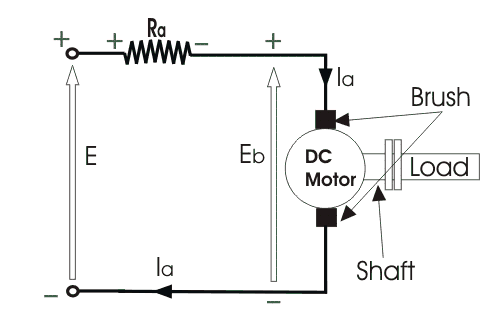
The input and output port variables of the direct current motor are related by the parameter K.

*T = KI and E = K ω*

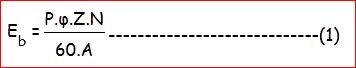
So from the picture above we can well understand that motor is just the opposite phenomena of a DC generator, and we can derive both motoring and generating operation from the same machine by simply reversing the ports.

**2.11.3 Detailed Description of DC motor**

To understand the DC motor in detail, let us consider the below diagram,

****

The direct [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) motor is represented by the circle in the centre, on which is mounted the brushes, where we connect the external terminals, from where supply [voltage](http://www.electrical4u.com/voltage-or-electric-potential-difference/) is given. On the mechanical terminal we have a shaft coming out of the Motor, and connected to the armature, and the armature-shaft is coupled to the mechanical load. On the supply terminals we represent the armature [resistance](http://www.electrical4u.com/electrical-resistance-and-laws-of-resistance/) Ra in series. Now, let the input [voltage](http://www.electrical4u.com/voltage-or-electric-potential-difference/) E, is applied across the brushes. Electric [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) which flows through the rotor armature via brushes, in presence of the [magnetic field](http://www.electrical4u.com/what-is-magnetic-field/), produces a torque Tg . Due to this torque Tg the dc motor armature rotates. As the armature conductors are carrying currents and the armature rotates inside the stator [magnetic field](http://www.electrical4u.com/what-is-magnetic-field/), it also produces an emf Eb in the manner very similar to that of a generator. The generated Emf Eb is directed opposite to the supplied [voltage](http://www.electrical4u.com/voltage-or-electric-potential-difference/) and is known as the back Emf, as it counters the forward voltage.  
The back emf like in case of a generator is represented by



Where, P = no of poles

φ = flux per pole

Z= No. of conductors

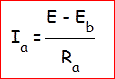
A = No. of parallel paths

and N is the speed of the DC Motor.

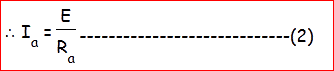
So from the above equation we can see Eb is proportional to speed ‘N’. That is whenever a direct [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) motor rotates, it results in the generation of back Emf. Now let us represent the rotor speed by ω in rad/sec. So Eb is proportional to ω.

So when the speed of the motor is reduced by the application of load, Eb decreases. Thus the [voltage difference](http://www.electrical4u.com/voltage-or-electric-potential-difference/) between supply voltage and back emf increases that means E − Eb increases. Due to this increased [voltage difference](http://www.electrical4u.com/voltage-or-electric-potential-difference/), armature [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) will increase and therefore torque and hence speed increases. Thus a DC Motor is capable of maintaining the same speed under variable load.

Now armature [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) Ia is represented by



Now at starting, speed ω = 0 so at starting Eb = 0.



Now since the armature winding [electrical resistance](http://www.electrical4u.com/electrical-resistance-and-laws-of-resistance/) Ra is small, this motor has a very high starting [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) in the absence of back Emf. As a result we need to use a starter for starting a DC Motor.

Now as the motor continues to rotate, the back Emf starts being generated and gradually the [current](http://www.electrical4u.com/electric-current-and-theory-of-electricity/) decreases as the motor picks up speed

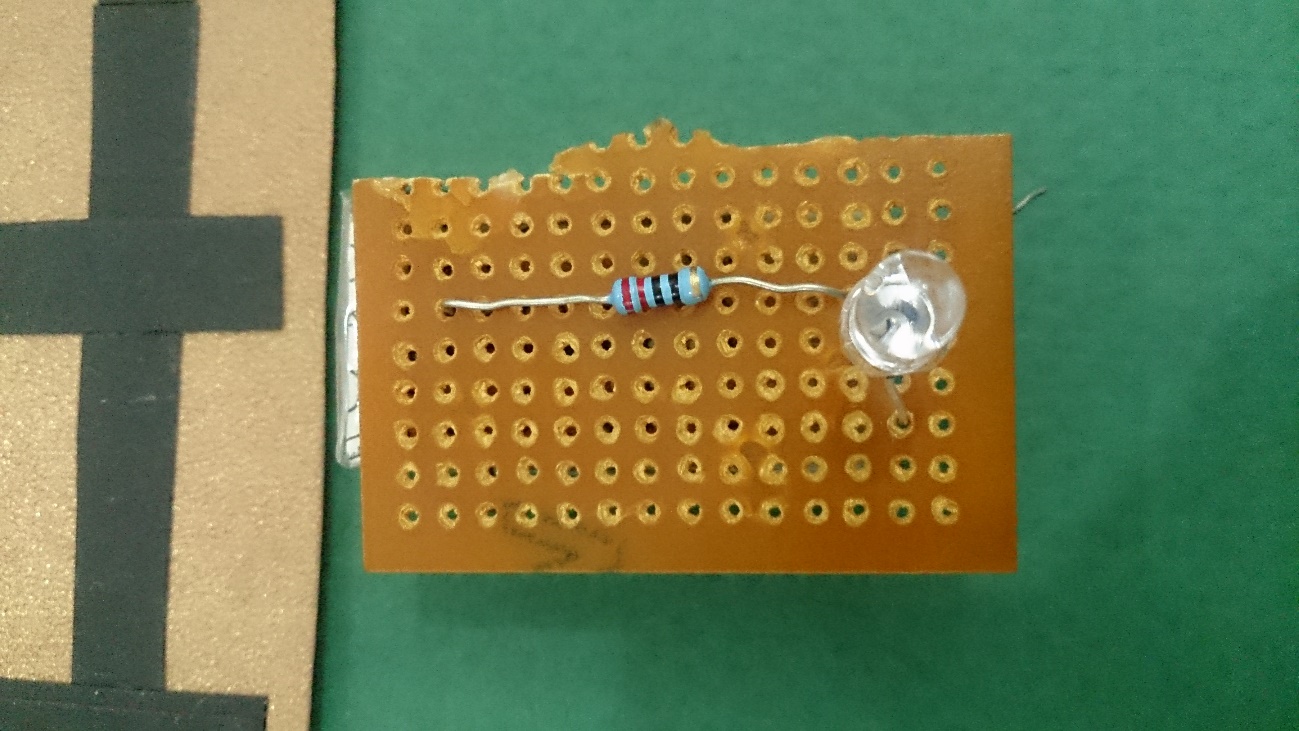
**CHAPTER 3**

**CIRCUIT CONNECTIONS & SIMULATION RESULTS**

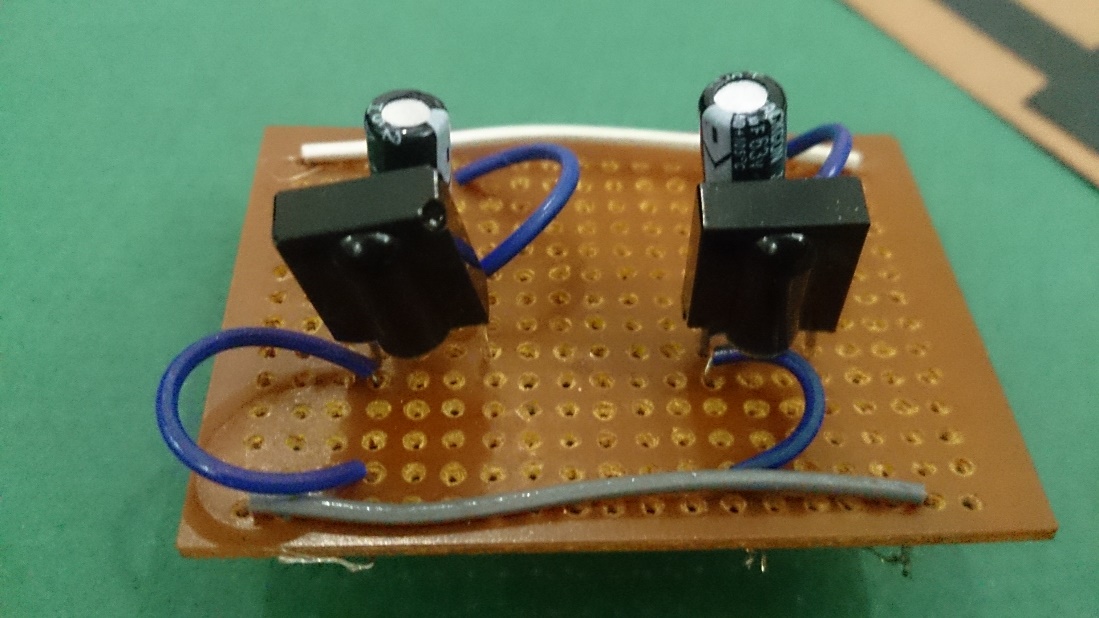
**3.1 Circuit Connections**

We have soldered the components on a general PCB and the connections are as follows.

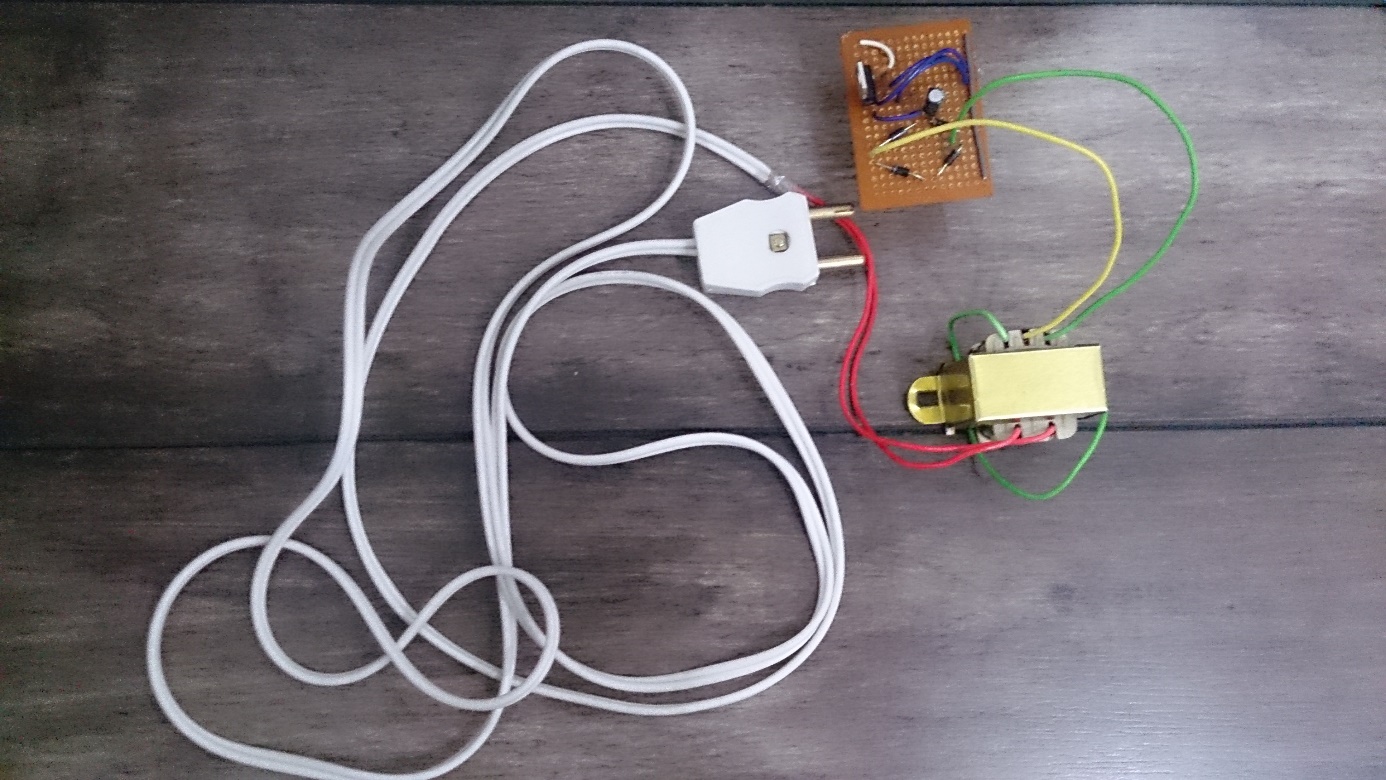
**3.1.1 IR LED Circuit**



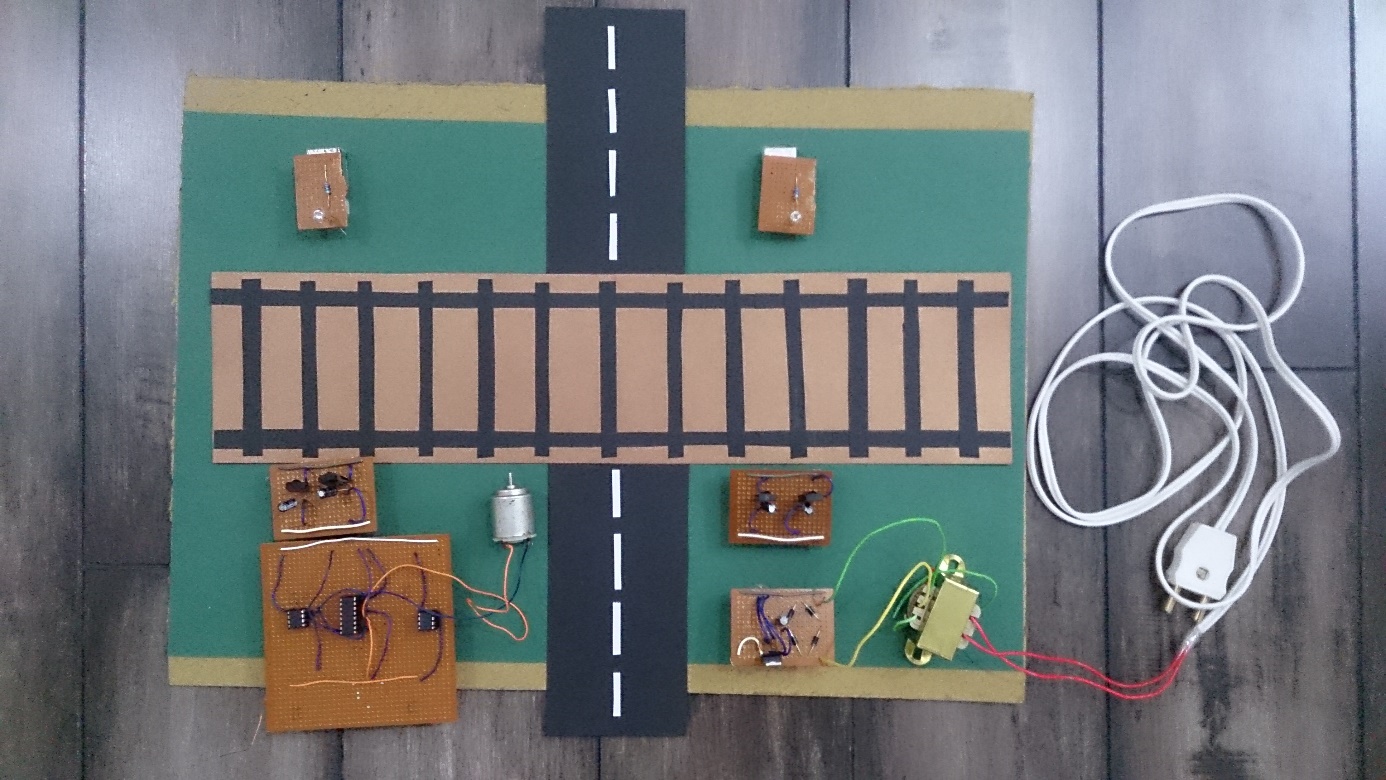
**3.1.2 TSOP Sensor Circuit**



**3.1.3 Power Supply Circuit**



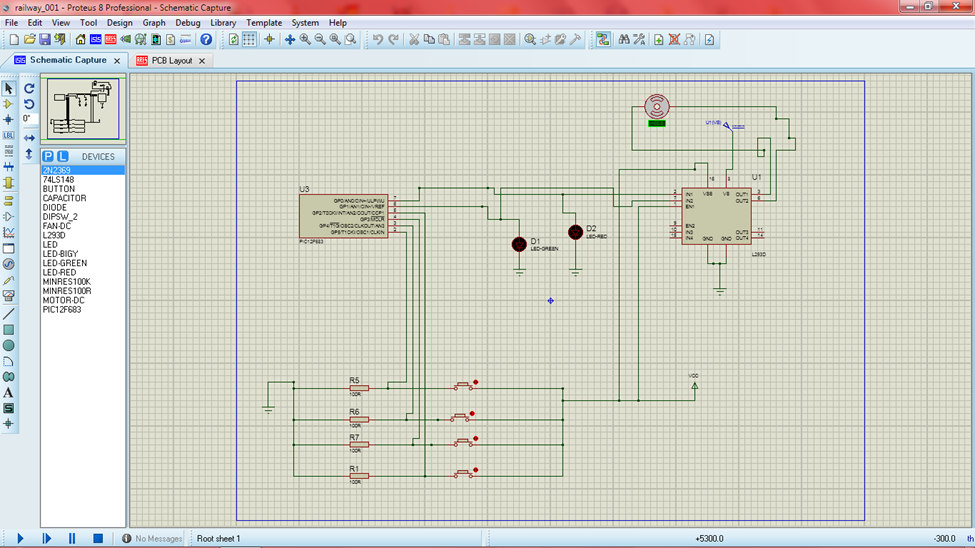
**3.1.4 Complete Circuit Connections**



**3.2 Simulation Results**

The following are the screen shots of the outputs we got for the program we have written. These screenshots are taken from Proteus Simulation software.

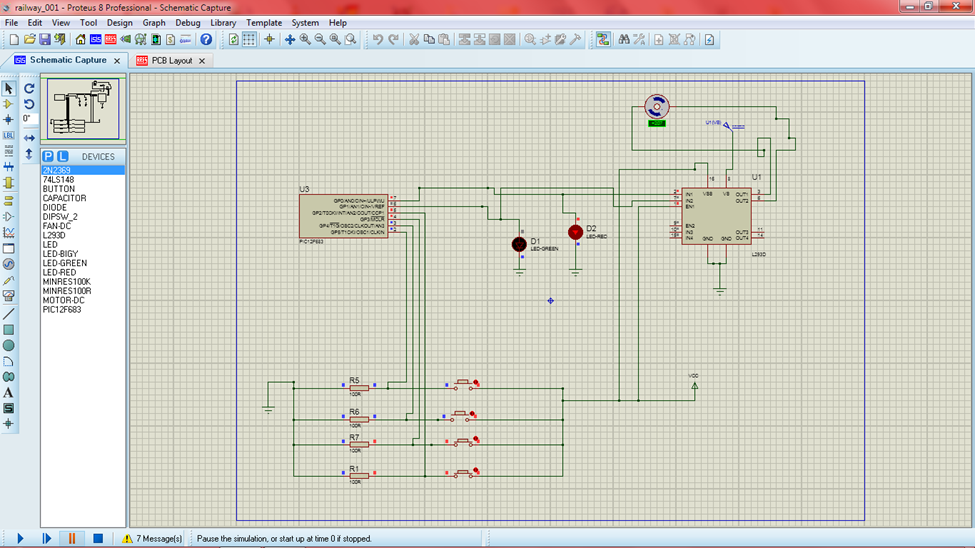
**Figure 3.2.1 Output when there is no train**

****

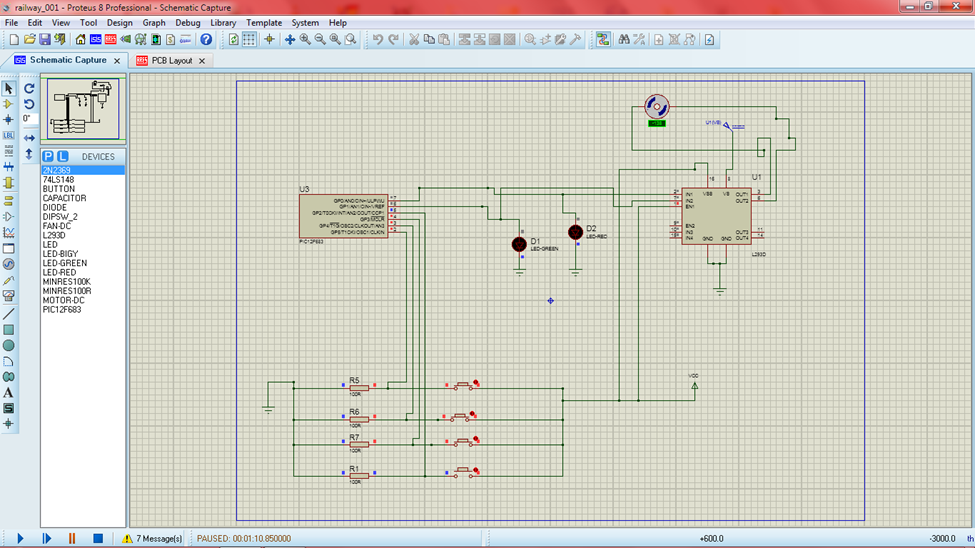
In the above figure we can see that all the sensors are blinking red in colour, this means that no object or train is interrupting the sensors on either side of the gate.

**Figure 3.2.2 Output when the first pair of sensors detect the train**

Here we can see that the first two sensors are active i.e. they sensed an object. Also, the values shown in the following picture below the motor are positive values, i.e. the motor is rotating in forward direction. This means that the gate will close.

****

**Figure 3.2.3 Output when the train leaves the track**

****

Here we can see that all the sensors are inactive except the last sensor, this means the train is leaving the track and the gate should be opened. For this to happen, the motor should rotate in reverse direction. We can see in the above figure that the values below the motor are negative values i.e. the motor is rotating in reverse direction. Hence we got the required output.

**CHAPTER 4**

**CONCLUSION**

The project presents unmanned railway gate control which has been designed to sense the occurrence of train and managing the railway gate. Though the concept of automatic railway gates is old, the implementation is entirely new. We have used PIC12F683 microcontroller to drive the sensors & gates; and LED & TSOPs as sensors. Using these components we have made the circuit less complex and less expensive. The project has very good scope in the future.