VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

PROJECT REPORT

MICROCONTROLLER AND ITS APPLICATION (ECE3003)

SUBMITTED TO:

Prof. Karthikeyan A

AUTOMATIC RAILWAY GATE CONTROLLER

BY-

J Joyce Imaculate -17BEC0567

Yogendra Bijwe-17BEC0563

Karthik-17BEC0695

ABSTRACT

As we are aware that the number of accidents at railway crossing has been increasing lately, more precautionary measures are being undertaken. Accidents take place either due to the absence of crossing gates, or due to impatience and carelessness of pedestrians. There are many places where the railroad crossings are open, without any gates. This becomes an open threat to the people or vehicles who wish to cross the railroad. Whereas, there are places where the crossing gates at railroad is being controlled and managed by a guard. In such cases, threat may arise due to carelessness of the guard. So, our project aims at automating this process. Automatic Railway Gate Control (ARGC) is managed and controlled by microcontroller 8051 and is more reliable since there is no human intervention in the process. Our model, most primarily, is different from others in terms of the programming method and considering various possibilities that the actually railroad crossing system encounters. We have programmed 8051 using assembly language program (.asm). Infrared (IR) sensors are used which senses the presence of train and conveys the information to 8051. 8051 is programmed accordingly to open or close the gate whenever required. Also, the steps for creating hex file from the program are elaborated. Keil uvision3 is being used to create the hex file. Keywords: Hex File, Keil, L293D Driver IC, Microcontroller 8051, Proteus6.

INTRODUCTION:

This system is designed to manage the control system of railway gate using the microcontroller. The main purpose of this paper is to propose an idea to make the process automated so that the probability of accidents reduces drastically. This system is designed using 8051 microcontroller to avoid railway accidents happening at railroad crossings. As a train approaches the railway crossing from either side, the sensors placed at a certain distance from the gate detects the approaching train and informs the microcontroller about it [1]. This signal is used to trigger the microcontroller for opening or closing of gates. The abstraction of this system is to provide the advanced control system available to everywhere.

AIM:

This system is to manage the control system of railway gate using the microcontroller. The main purpose of this system is about railway gate control system and level crossing between railroad and highway for decreasing railroad-related accident and increasing safety. In addition, it also provides safety to road users by reducing the accidents that usually occur due to carelessness of road users and errors made by the gatekeepers. Railways are the preferred cheapest mode of transportation over all the other means. Also, it is evident from the experimental results that accidents at railroad crossings with the gate control are far less than those crossings that are devoid of gates.

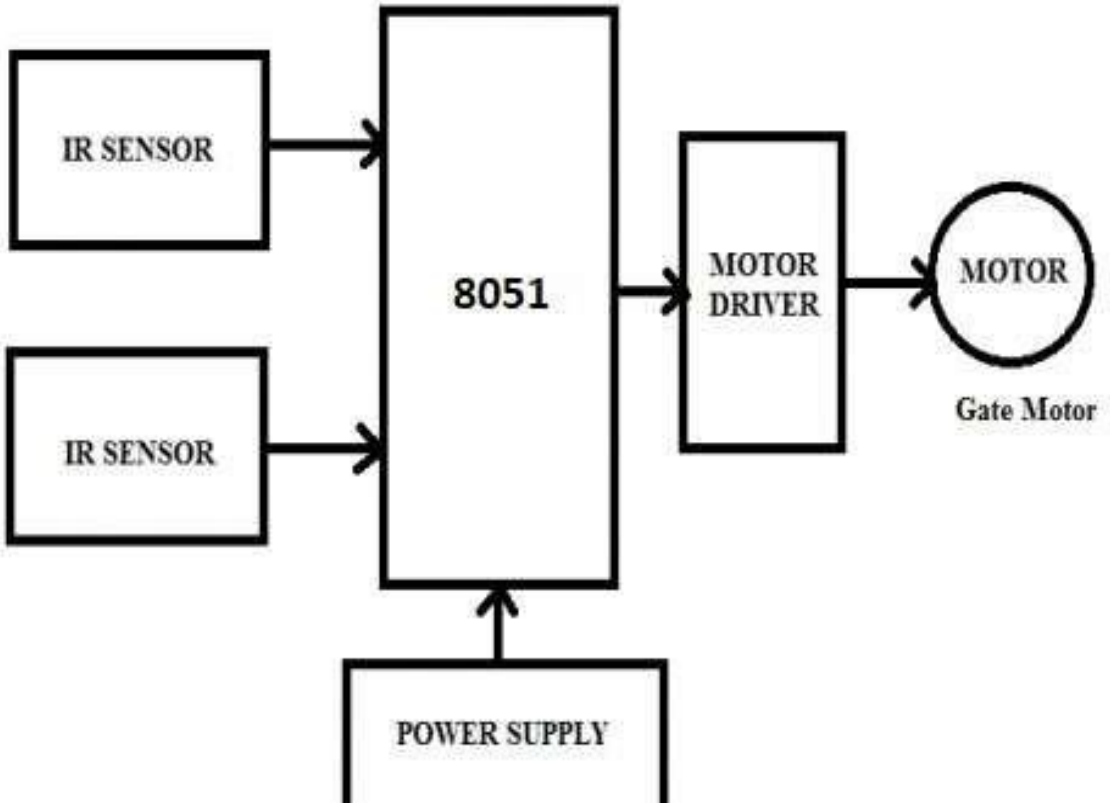
COMPONENTS REQUIRED:

- 8051 MICRO-CONTROLLER
- Keil MICRO-VISION
- PROTEUS SOFTWARE
- FLASH MAGIC
- IC555(TIMER)
- TSOP1738
- BC557
- L293D MOTOR DRIVER
- IR LED
- BATTERY
- CAPACITORS
- RESISTORS

ALGORITHM:

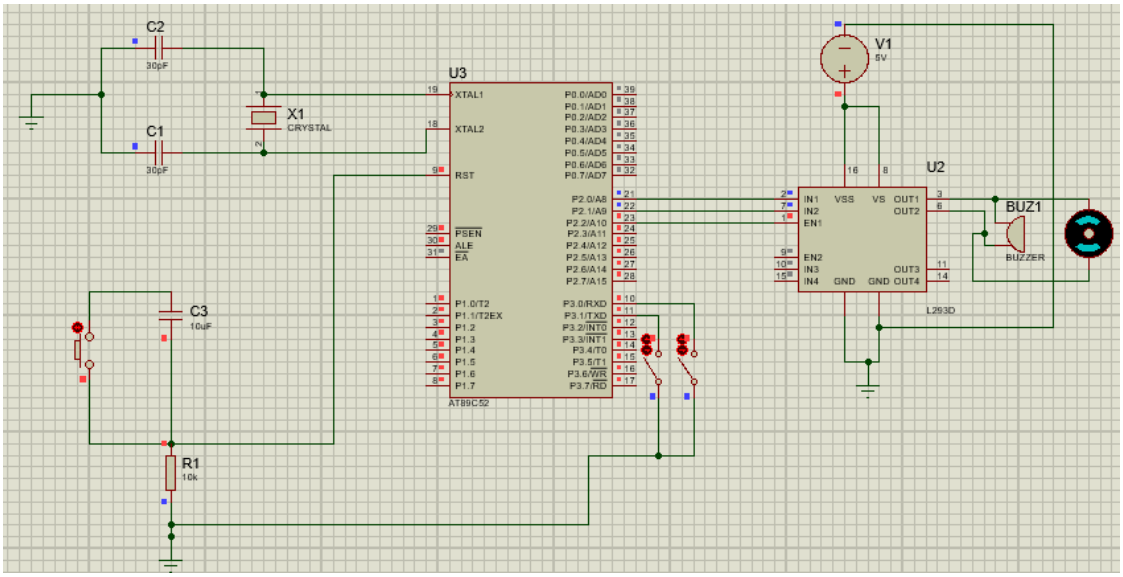
1. We take two pairs of IR transmitter and receiver circuit. One before the crossing and other after the crossing. which will detect the incoming train.
2. It is interfaced with an 8051 microcontroller which keeps on checking for any blocking in the IR led.
3. If there is a blockage it will now send input to the motor driver L293D which in turn will move the motors of the gate and close the gate.
4. When the blockage is gone i.e. the train has crossed 8051 micro- controller will again send input to the L293D board and open the gate.

BLOCK DIAGRAM:



INTERFACE DIAGRAM:

8051 and L293D connections

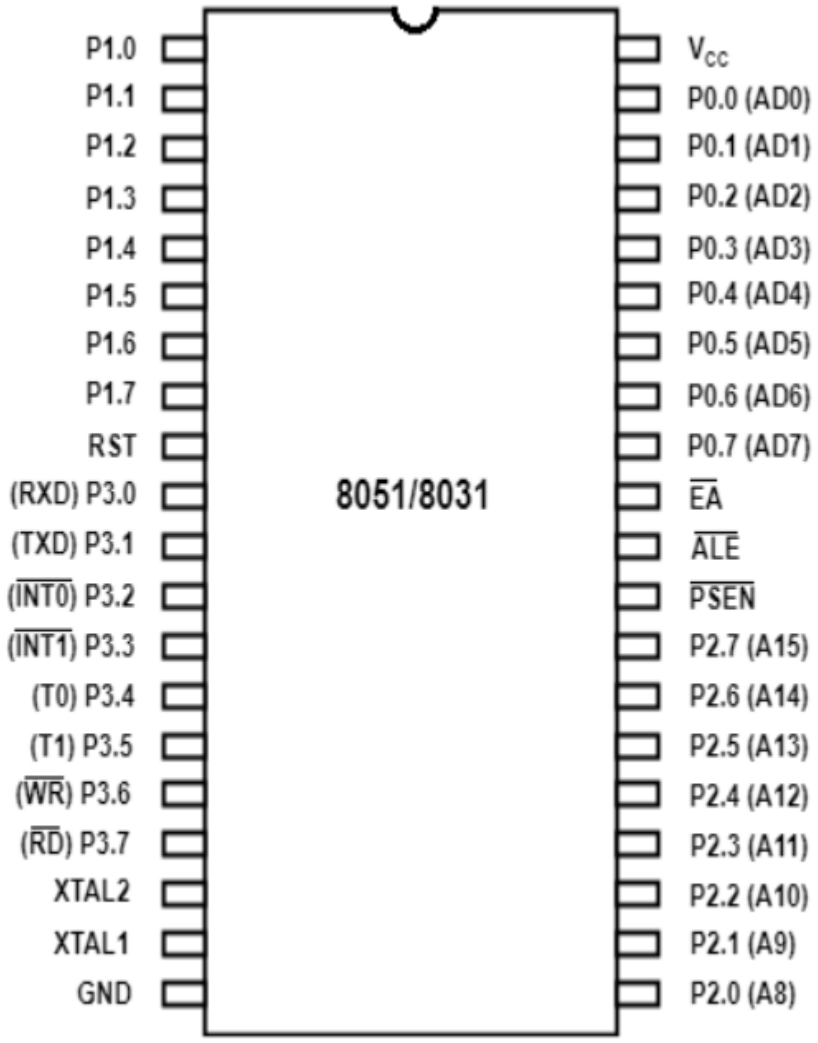


LITERATURE SURVEY

IR Transmitter and receiver

WORKING:
The circuit has two parts:
IR Transmitter: the IR LED emitting infrared light is placed in the transmitting unit generate IR signal
IR Receiver: the receiver unit consists of a sensor, which detects IR pulses transmitted by IR LED

PIN DIAGRAM AND DESCRIPTION:



Pin Diagram of 8051 Microcontroller

Pin Description

VCC

Supply voltage.

GND

Ground.

Port 0

Port 0 is an 8-bit open drain bidirectional I/O port. As an output port, each pin can sink eight TTL inputs. When 1s are written to port 0 pins, the pins can be used as high-impedance inputs.

Port 0 can also be configured to be the multiplexed low-order address/data bus during accesses to external program and data memory. In this mode, P0 has internal pull-ups.

Port 0 also receives the code bytes during Flash programming and outputs the code bytes during program verification. **External pull-ups are required during program verification**.

Port 1

Port 1 is an 8-bit bidirectional I/O port with internal pull-ups. The Port 1 output buffers can sink/source four TTL inputs. When 1s are written to Port 1 pins, they are pulled high by the internal pull-ups and can be used as inputs. As inputs, Port 1 pins that are externally being pulled low will source current (I_{IL}) because of the internal pull-ups.

Port 1 also receives the low-order address bytes during Flash programming and verification.

Port Pin	Alternate Functions
P1.5	MOSI (used for In-System Programming)
P1.6	MISO (used for In-System Programming)
P1.7	SCK (used for In-System Programming)

Port 2

Port 2 is an 8-bit bidirectional I/O port with internal pull-ups. The Port 2 output buffers can sink/source four TTL inputs. When 1s are written to Port 2 pins, they are pulled high by the internal pull-ups and can be used as inputs. As inputs, Port 2 pins that are externally being pulled low will source current (I_{IL}) because of the internal pull-ups.

Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that use 16-bit addresses (MOVX @ DPTR). In this application, Port 2 uses strong internal pull-ups when emitting 1s. During accesses to external data memory that use 8-bit addresses (MOVX @ RI), Port 2 emits the contents of the P2 Special Function Register.

Port 2 also receives the high-order address bits and some control signals during Flash programming and verification.

Port 3

Port 3 is an 8-bit bidirectional I/O port with internal pull-ups. The Port 3 output buffers can sink/source four TTL inputs. When 1s are written to Port 3 pins, they are pulled high by the internal pull-ups and can be used as inputs. As inputs, Port 3 pins that are externally being pulled low will source current (I_{IL}) because of the pull-ups.

Port 3 receives some control signals for Flash programming and verification.

Port 3 also serves the functions of various special features of the AT89S51, as shown in the following table.

Port Pin	Alternate Functions
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	$\overline{INT0}$ (external interrupt 0)
P3.3	$\overline{INT1}$ (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)
P3.6	$\overline{WR}$ (external data memory write strobe)
P3.7	$\overline{RD}$ (external data memory read strobe)

RST

Reset input. A high on this pin for two machine cycles while the oscillator is running resets the device. This pin drives High for 98 oscillator periods after the Watchdog times out. The DISRTO bit in SFR AUXR (address 8EH) can be used to disable this feature. In the default state of bit DISRTO, the RESET HIGH out feature is enabled.

ALE/$\overline{PROG}$

Address Latch Enable (ALE) is an output pulse for latching the low byte of the address during accesses to external memory. This pin is also the program pulse input ($\overline{PROG}$) during Flash programming.

In normal operation, ALE is emitted at a constant rate of 1/6 the oscillator frequency and may be used for external timing or clocking purposes. Note, however, that one ALE pulse is skipped during each access to external data memory.

If desired, ALE operation can be disabled by setting bit 0 of SFR location 8EH. With the bit set, ALE is active only during a MOVX or MOVC instruction. Otherwise, the pin is weakly pulled high. Setting the ALE-disable bit has no effect if the microcontroller is in external execution mode.

$\overline{PSEN}$

Program Store Enable ($\overline{PSEN}$) is the read strobe to external program memory.

When the AT89S51 is executing code from external program memory, $\overline{PSEN}$ is activated twice each machine cycle, except that two $\overline{PSEN}$ activations are skipped during each access to external data memory.

$\overline{EA}$/VPP

External Access Enable. $\overline{EA}$ must be strapped to GND in order to enable the device to fetch code from external program memory locations starting at 0000H up to FFFFH. Note, however, that if lock bit 1 is programmed, $\overline{EA}$ will be internally latched on reset.

$\overline{EA}$ should be strapped to V_{CC} for internal program executions.

This pin also receives the 12-volt programming enable voltage (V_{PP}) during Flash programming.

XTAL1

Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

XTAL2

Output from the inverting oscillator amplifier

DATA SHEET (8051):AT89S51

Features

• Compatible with MCS-51® Products
• 4K Bytes of In-System Programmable (ISP) Flash Memory
– Endurance: 1000 Write/Erase Cycles
• 4.0V to 5.5V Operating Range
• Fully Static Operation: 0 Hz to 33 MHz
• Three-level Program Memory Lock
• 128 x 8-bit Internal RAM
• 32 Programmable I/O Lines
• Two 16-bit Timer/Counters
• Six Interrupt Sources
• Full Duplex UART Serial Channel
• Low-power Idle and Power-down Modes
• Interrupt Recovery from Power-down Mode
• Watchdog Timer
• Dual Data Pointer
• Power-off Flag
• Fast Programming Time
• Flexible ISP Programming (Byte and Page Mode)

Description

The AT89S51 is a low-power, high-performance CMOS 8-bit microcontroller with 4K bytes of in-system programmable Flash memory. The device is manufactured using Atmel's high-density nonvolatile memory technology and is compatible with the industry-standard 80C51 instruction set and pinout. The on-chip Flash allows the program memory to be reprogrammed in-system or by a conventional nonvolatile memory programmer. By combining a versatile 8-bit CPU with in-system programmable Flash on a monolithic chip, the Atmel AT89S51 is a powerful microcontroller which provides a highly-flexible and cost-effective solution to many embedded control applications. The AT89S51 provides the following standard features: 4K bytes of Flash, 128 bytes of RAM, 32 I/O lines, Watchdog timer, two data pointers, two 16-bit timer/counters, a fivevector two-level interrupt architecture, a full duplex serial port, on-chip oscillator, and clock circuitry. In addition, the AT89S51 is designed with static logic for operation down to zero frequency and supports two software selectable power saving modes. The Idle Mode stops the CPU while allowing the RAM, timer/counters, serial port, and interrupt system to continue functioning. The Power-down mode saves the RAM contents but freezes the oscillator, disabling all other chip functions until the next external interrupt or hardware reset.

8051- INSTRUCATION SET:

Mnemonic		Description	Byte	Cycle
Arithmetic operations				
ADD	A,Rn	Add register to accumulator	1	1
ADD	A,direct	Add direct byte to accumulator	2	1
ADD	A,@Ri	Add indirect RAM to accumulator	1	1
ADD	A,#data	Add immediate data to accumulator	2	1
ADDC	A,Rn	Add register to accumulator with carry flag	1	1
ADDC	A,direct	Add direct byte to A with carry flag	2	1
ADDC	A,@Ri	Add indirect RAM to A with carry flag	1	1
ADDC	A,#data	Add immediate data to A with carry flag	2	1
SUBB	A,Rn	Subtract register to accumulator with borrow	1	1
SUBB	A,direct	Subtract direct byte to A with carry borrow	2	1
SUBB	A,@Ri	Subtract indirect RAM to A with carry borrow	1	1
SUBB	A,#data	Subtract immediate data to A with carry borrow	2	1
INC	A	Increment accumulator	1	1
INC	Rn	Increment register	1	1
INC	direct	Increment direct byte	2	1
INC	@Ri	Increment indirect RAM	1	1
DEC	A	Decrement accumulator	1	1
DEC	Rn	Decrement register	1	1
DEC	direct	Decrement direct byte	2	1
DEC	@Ri	Decrement indirect RAM	1	1
INC	DPTR	Increment data pointer	1	2
MUL	AB	Multiply A and B -> [B hi]:[A lo]	1	4
DIV	AB	Divide A by B -> A=result, B=remainder	1	4
DA	A	Decimal adjust accumulator	1	1
CLR	A	Clear accumulator	1	1
CPL	A	Complement accumulator	1	1
RL	A	Rotate accumulator left	1	1
RLC	A	Rotate accumulator left through carry	1	1
RR	A	Rotate accumulator right	1	1
RRC	A	Rotate accumulator right through carry	1	1

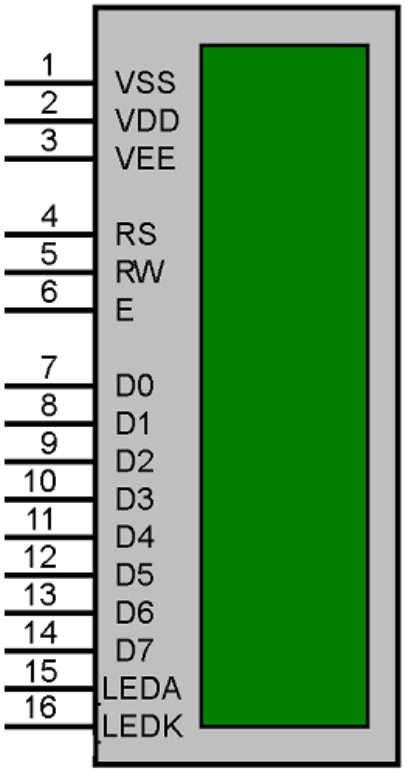
Logic operations				
ANL	A,Rn	AND register to accumulator	1	1
ANL	A,direct	AND direct byte to accumulator	2	1
ANL	A,@Ri	AND indirect RAM to accumulator	1	1
ANL	A,#data	AND immediate data to accumulator	2	1
ANL	direct,A	AND accumulator to direct byte	2	1
ANL	direct,#data	AND immediate data to direct byte	3	2
ORL	A,Rn	OR register to accumulator	1	1
ORL	A,direct	OR direct byte to accumulator	2	1
ORL	A,@Ri	OR indirect RAM to accumulator	1	1
ORL	A,#data	OR immediate data to accumulator	2	1
ORL	direct,A	OR accumulator to direct byte	2	1
ORL	direct,#data	OR immediate data to direct byte	3	2
XRL	A,Rn	Exclusive OR register to accumulator	1	1
XRL	A,direct	Exclusive OR direct byte to accumulator	2	1
XRL	A,@Ri	Exclusive OR indirect RAM to accumulator	1	1
XRL	A,#data	Exclusive OR immediate data to accumulator	2	1
XRL	direct,A	Exclusive OR accumulator to direct byte	2	1
XRL	direct,#data	Exclusive OR immediate data to direct byte	3	2

Data transfer

MOV A,Rn	Move register to accumulator	1	1
MOV A,direct*)	Move direct byte to accumulator	2	1
MOV A,@Ri	Move indirect RAM to accumulator	1	1
MOV A,#data	Move immediate data to accumulator	2	1
MOV Rn,A	Move accumulator to register	1	1
MOV Rn,direct	Move direct byte to register	2	2
MOV Rn,#data	Move immediate data to register	2	1
MOV direct,A	Move accumulator to direct byte	2	1
MOV direct,Rn	Move register to direct byte	2	2
MOV direct,direct	Move direct byte to direct byte	3	2
MOV direct,@Ri	Move indirect RAM to direct byte	2	2
MOV direct,#data	Move immediate data to direct byte	3	2
MOV @Ri,A	Move accumulator to indirect RAM	1	1
MOV @Ri,direct	Move direct byte to indirect RAM	2	2
MOV @Ri,#data	Move immediate data to indirect RAM	2	1
MOV DPTR,#data16	Load data pointer with a 16-bit constant	3	2
MOVC A,@A+DPTR	Move code byte relative to DPTR to accumulator	1	2
MOVC A,@A+PC	Move code byte relative to PC to accumulator	1	2
MOVX A,@Ri	Move external RAM (8-bit addr.) to A	1	2
MOVX A,@DPTR	Move external RAM (16-bit addr.) to A	1	2
MOVX @Ri,A	Move A to external RAM (8-bit addr.)	1	2
MOVX @DPTR,A	Move A to external RAM (16-bit addr.)	1	2
PUSH direct	Push direct byte onto stack	2	2
POP direct	Pop direct byte from stack	2	2
XCH A,Rn	Exchange register to accumulator	1	1
XCH A,direct	Exchange direct byte to accumulator	2	1
XCH A,@Ri	Exchange indirect RAM to accumulator	1	1
XCHD A,@Ri	Exchange low-order nibble indir. RAM with A	1	1

LCD DISPLAY

LCD PIN CONFIGURATION



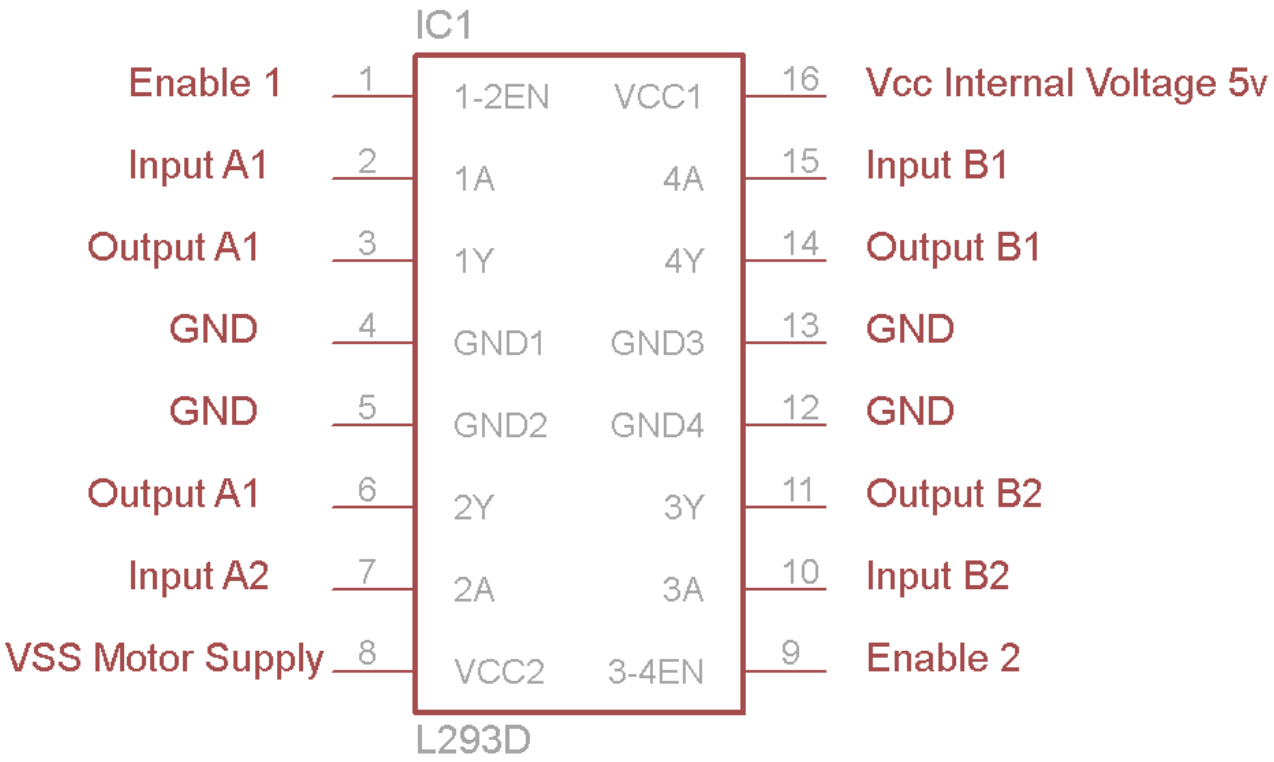
Pin Configuration

Pin No:	Pin Name:	Description
1	Vss (Ground)	Ground pin connected to system ground
2	Vdd (+5 Volt)	Powers the LCD with +5V (4.7V – 5.3V)
3	VE (Contrast V)	Decides the contrast level of display. Grounded to get maximum contrast.
4	Register Select	Connected to Microcontroller to shit between command/data register
5	Read/Write	Used to read or write data. Normally grounded to write data to LCD
6	Enable	Connected to Microcontroller Pin and toggled between 1 and 0 for data acknowledgement
7	Data Pin 0	Data pins 0 to 7 forms a 8-bit data line. They can be connected to Microcontroller to send 8-bit data. These LCD's can also operate on 4-bit mode in such case Data pin 4,5,6 and 7 will be left free.
8	Data Pin 1	
9	Data Pin 2	
10	Data Pin 3	
11	Data Pin 4	
12	Data Pin 5	
13	Data Pin 6	
14	Data Pin 7	
15	LED Positive	Backlight LED pin positive terminal
16	LED Negative	Backlight LED pin negative terminal

LCD commands:

Hex Code	Command to LCD Instruction Register
0F	LCD ON, cursor ON
01	Clear display screen
02	Return home
04	Decrement cursor (shift cursor to left)
06	Increment cursor (shift cursor to right)
05	Shift display right
07	Shift display left
0E	Display ON, cursor blinking
80	Force cursor to beginning of first line
C0	Force cursor to beginning of second line
38	2 lines and 5×7 matrix
83	Cursor line 1 position 3
3C	Activate second line
08	Display OFF, cursor OFF
C1	Jump to second line, position 1
OC	Display ON, cursor OFF
C1	Jump to second line, position 1
C2	Jump to second line, position 2

L293D MOTOR DRIVER:



Working of L293D

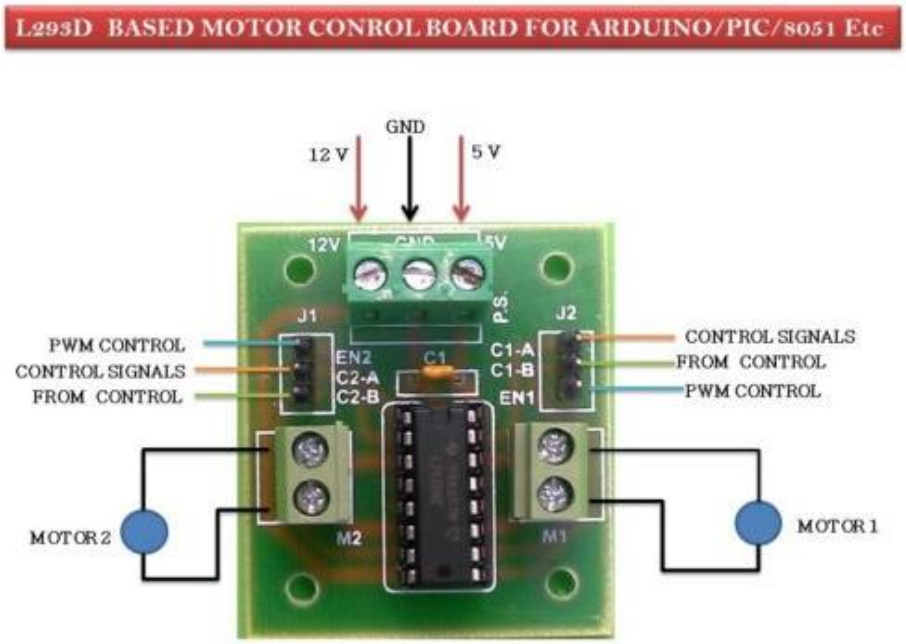
There are 4 input pins for l293d, pin 2,7 on the left and pin 15 ,10 on the right as shown on the pin diagram. Left input pins will regulate the rotation of motor connected across left side and right input for motor on the right hand side.The motors are rotated on the basis of the inputs provided across the input pins as LOGIC 0 or LOGIC 1.

L293D Logic Table.

Lets consider a Motor connected on left side output pins (pin 3,6). For rotating the motor in clockwise direction the input pins has to be provided with Logic 1 and Logic 0.

- **Pin 2** = **Logic 1** and **Pin 7** = **Logic 0** | Clockwise Direction
- **Pin 2** = **Logic 0** and **Pin 7** = **Logic 1** | Anticlockwise Direction
- **Pin 2** = **Logic 0** and **Pin 7** = **Logic 0** | Idle [No rotation] [Hi-Impedance state]
- **Pin 2** = **Logic 1** and **Pin 7** = **Logic 1** | Idle [No rotation]

L293D Module:



PROGRAM IN ASSEMBLY LANGUAGE:

```
ORG 0000H
        START: MOV P3,#0FFH
        CLR P2.0
        CLR P2.1
        SETB P2.2
        CheckIR: ;ACALL DELAY
        MOV R7,P3
        MOV A,R7
        RRC A
        RRC A
        RRC A
        JC MotorForward
        SJMP CheckIR

        BACK1: MOV R6,P3
        MOV A,R6
        RRC A
        RRC A
        JC SecondIRDetected
        SJMP BACK1

        MotorForward: ACALL port_ini
        ACALL Lcd_ini
        SETB P2.0
        CLR P2.1
        ACALL OFF
        ACALL DELAY1
        ACALL DELAY1
        ACALL DELAY
        CLR P2.0
        SJMP BACK1

        SecondIRDetected: MOV R5,P3
        MOV A,R5
        RRC A
        RRC A
        JC MotorReverse_Check_FirstIR
        SJMP SecondIRDetected

        MotorReverse_Check_FirstIR: MOV R4,P3
        MOV A,R4
        RRC A
        RRC A
        RRC A
        JNC MotorReverse_Check_FirstIR

        SecondIR_Check: MOV R3,P3
        MOV A,R3
        RRC A
        RRC A
        JNC SecondIR_Check
        ACALL port_ini
```

```
ACALL Lcd_ini
CLR P2.0
SETB P2.1
ACALL ONN
ACALL DELAY1
ACALL DELAY1
ACALL DELAY
CLR P2.1
SJMP Start
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

port_ini:
mov p1,#00h
setb p3.4
setb p3.5
setb p3.6
acall delay_1s

Lcd_ini:
mov a,#38h
acall command
mov a,#06h
acall command
mov a,#0ch
acall command
mov a,#01h
acall command
mov a,#80h
acall command
acall delay_1s

OFF:
mov a,#'C'
acall write
mov a,#'L'
acall write
mov a,#'O'
acall write
mov a,#'S'
acall write
mov a,#'E'
acall write

ONN:
mov a,#'O'
acall write
mov a,#'P'
acall write
mov a,#'E'
acall write
mov a,#'N'
acall write

command:
mov p1,a
```

```
clr p3.4
clr p3.5
setb p3.6
acall delay2
clr p3.6
acall delay2
acall delay2
ret

write:
lcall ready
mov p1,a
setb p3.4
clr p3.5
setb p3.6
acall delay2
clr p3.6
acall delay2
acall delay2
ret

ready:
setb p1.7
clr p3.4
setb p3.5
wait2:
clr p3.6
acall delay2
setb p3.6
jb p1.7,wait2
ret

delay2:
mov r0,#1ch
rep:
djnz r0,rep
ret
delay_1s:
mov r3,#08h
df1s:
mov r2,#0ffh
d1s:
mov r1,#0ffh
de1s:
djnz r1,de1s
djnz r2,d1s
djnz r3,df1s
ret
```

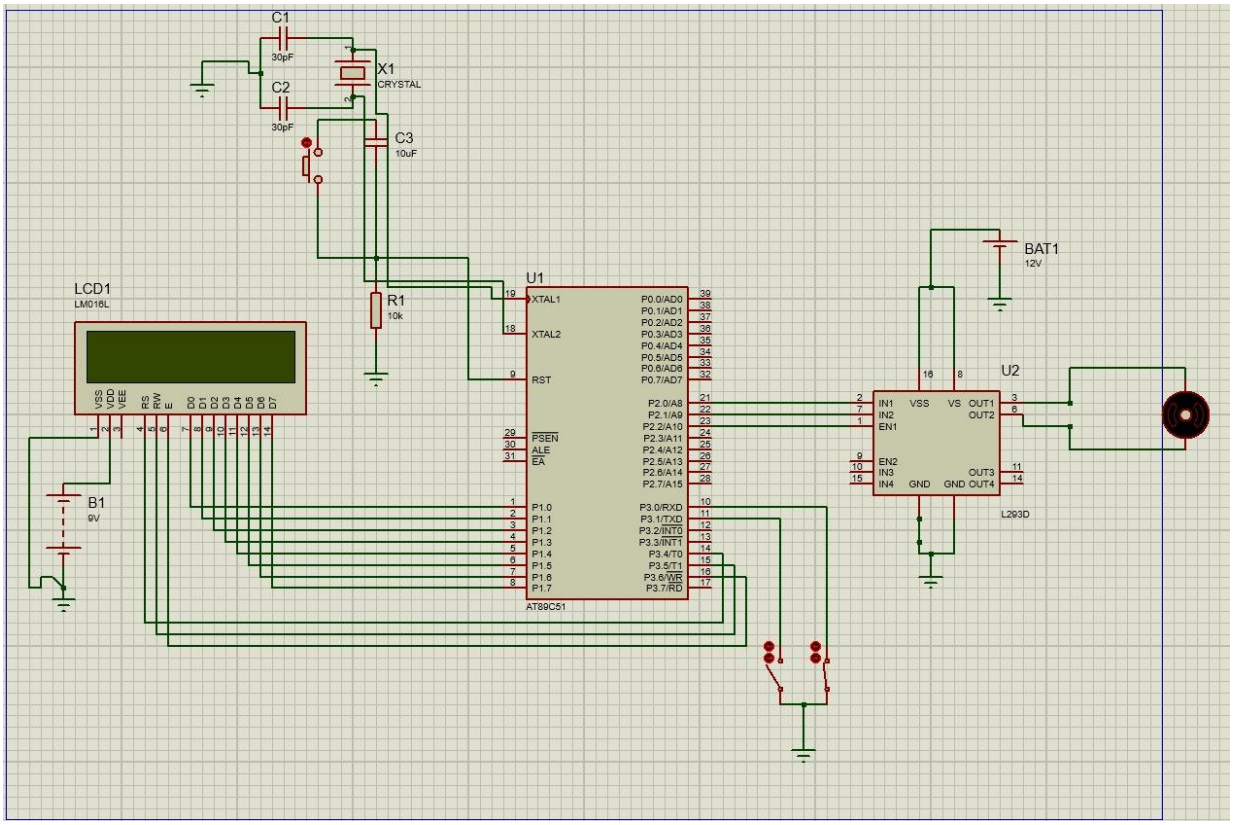
```
;;;;;;;;;;;;;;;;;;;;;;;;;

        DELAY: MOV R7,#0FFH
        TOP: MOV R6,#0FFH
        MOV R5,#0FFH
        BACK: DJNZ R5,BACK
        UP: DJNZ R6,UP
        DJNZ R7,TOP
        RET

        DELAY1: MOV R4,#14H
        OneSec: MOV TMOD,#01H
        MOV TL0,#0AFH
        MOV TH0,#3CH
        SETB TR0

        WAIT: JNB TF0,WAIT
        CLR TR0
        CLR TF0
        DJNZ R4,OneSec
        RET
        END
```

PROTEUS SIMULATION:



APPLICATION:

- It can be used for railway crossing and in other countries where road trams are popular. it will be applicable in safety of people.
- It can also be used as automatic entrance gate like in libraries.

ADVANTAGES:

- Human dependency will be over
- More accurate
- Less accidents
- Can be installed in village area

COMPARISON:

- it can be installed in village areas as it is fully automatic.
- It can work more efficiently as there may be some errors due to human interference.
- It can work 24 hours and just installation cost will be required but in existing system you need to employ people.
- Right now, people depend on the station employee to give them knowledge about the incoming train, but in out system it can be done automatically.

INFERENCE:

In the past 20 years, there have been a total of 21 collisions between vehicles and trains at crossings on the New Haven Line, with safeguards at the crossings ranging from the higher level of protection of drop arm gates to as little as stop signs and cross buck crossing signs at private crossings. The accidents caused 15 injuries and two fatalities, according to the statistics. In 1980, there were only 16,291 public railroad-crossings equipped with automated gates and lights, compared with 34,296 such crossings in 2000. This 111% increase in the number of gated crossings could explain at least 50% of the reduction in railroad-crossing accidents. This is because gates tend to be installed at the most densely-travelled crossings, and on a unit-of traffic basis, numerous studies have shown gates to be 85-90% more effective than passive devices in saving lives.

CONCLUSION:

Automatic gate control system offers an effective way to reduce the occurrence of railway accidents. This system can contribute a lot of benefit either to the road users or to the railway management. Since the design is completely automated it can be used in remote villages where no station master or line man is present. Railway sensors are placed at two sides of gate. It is used to sense the arrival and departure of the train. This system uses the DC motor to open and close the gates automatically when it is rotated clockwise or anticlockwise direction. Now a day's automatic system occupies each and every sector of applications as it is reliable and accurate.

REFERENCE:

1. The 8051 Microcontroller and Embedded Systems, Using Assembly and C, Muhammad Ali Mazidi ,Janice Gillispie Mazidi , Rolin D. Mckinlay.
2. Assembly Language Programmed Automatic railway gate control ,Vignesh Shankar ,IJIRST ,Volume 3, Issue 6,November 2016.
3. 3. C.R.Balamurugan*, P.Vijayshankarganth, R.Alagarraja, V.E.Subramanian, R.Ragupathy, Automatic Railway Gate Control System Using 8051micro Controller, International Journal of ChemTech Research ,2018
4. "Karthik Krishnamurthi" , Monica Bobby, Vidya V, Edwin Baby: - International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 4 Issue 2, February 2015.