Smart Gate

Introduction

Smart Gate is an automated system that can operate according to the need of the user and mainly used in an unguided system.

The basic tasks of an *Smart Gate* can be divided into 3 sections, namely

- Navigation
- Processing
- Execution

The system must be equipped with some means by which it can navigate its surrounding to check the existence of a car or human. In this project we have used IR Sensors for this purpose of navigation .

After navigating the surrounding the system must be capable of processing the input data from the navigation section. In this project, the processing section is done with an EAB, which comes along with the microcontroller PIC18F26K22.

While processing, the robot takes decision according to the algorithm designed for it.

After processing the navigated data the robot must do some work (e.g. Movement). In this project we have used motor driver IC L293D to drive the motor/gate according to the processed output from the EAB.

IR Sensor

IR Sensor is composed of an IR Emitter and an IR Receiver. When the IR Sensor is powered up, the IR emitter emits infrared signal continuously and the IR receiver is for receiving that infrared signal when it is echoed back after striking an obstacle. So, whenever the IR receiver receives an infrared signal, the LED associated with it glows i.e. in ON state and if the IR receiver does not receive any signal, the LED is in the OFF state. So, this change in LED state can be measured with the EAB.

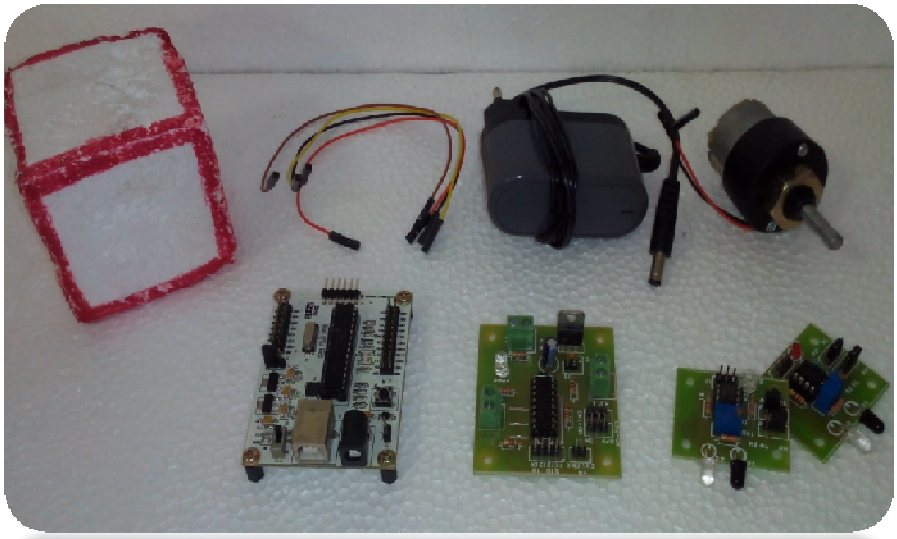
Motor Driver Board

The motor driver board consists of IC L293D. It has 4 inputs and 4 outputs. Since, the current, from the EAB is not sufficient enough to drive the motors, so we have to use this Driver to increase the current by which we can drive the DC Motors.

Components

The Components required for building the Smart Gate are:

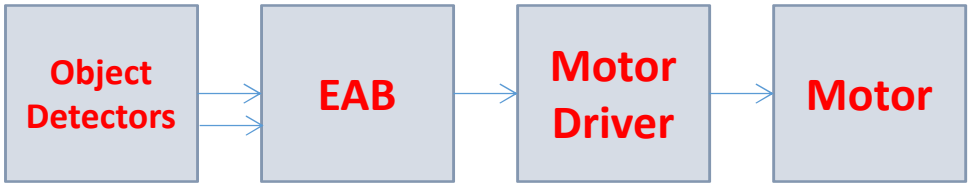
- Embedded Application Board
- Motor Driver Board (containing L293D)
- IR Sensor (2 nos)
- DC Motors (1 nos)
- Connectors(Jumpers)
- Power Adapter
- Gate
- Stand



Application Notes

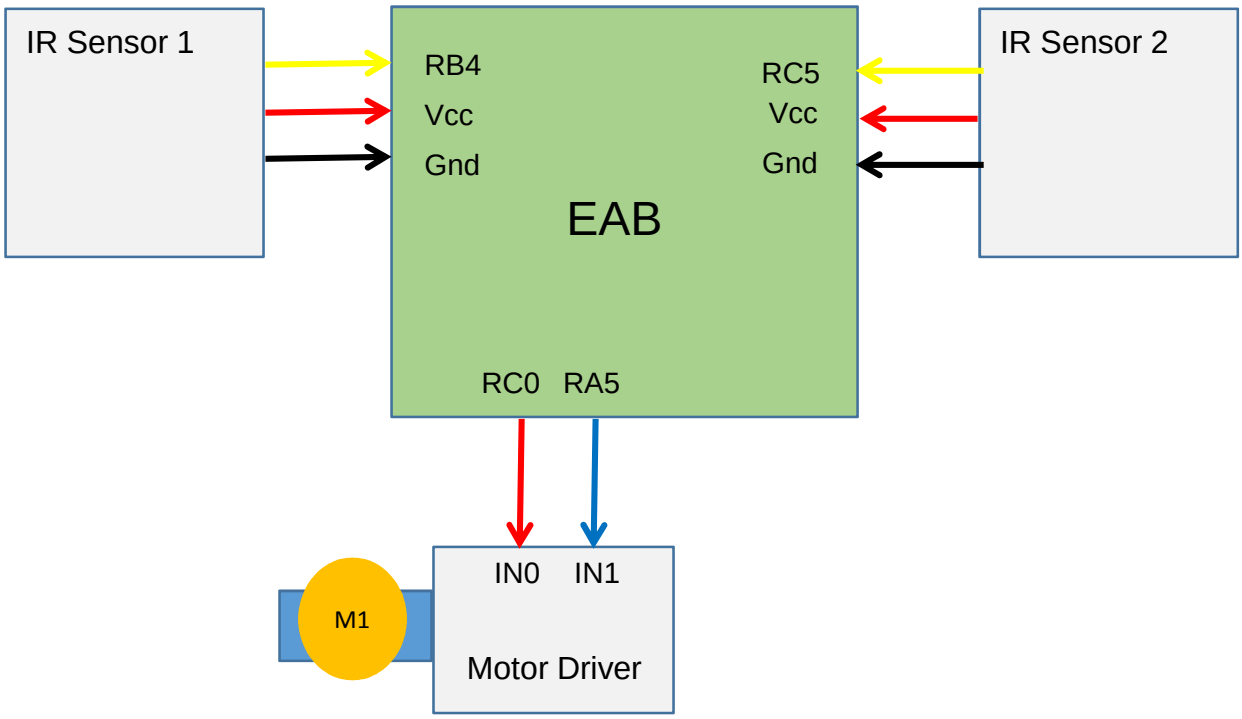
Block Diagram

Block level representation of the different blocks of the Smart Gate.



Schematic Diagram

The Schematic diagram illustrates the circuit connections for designing the application.

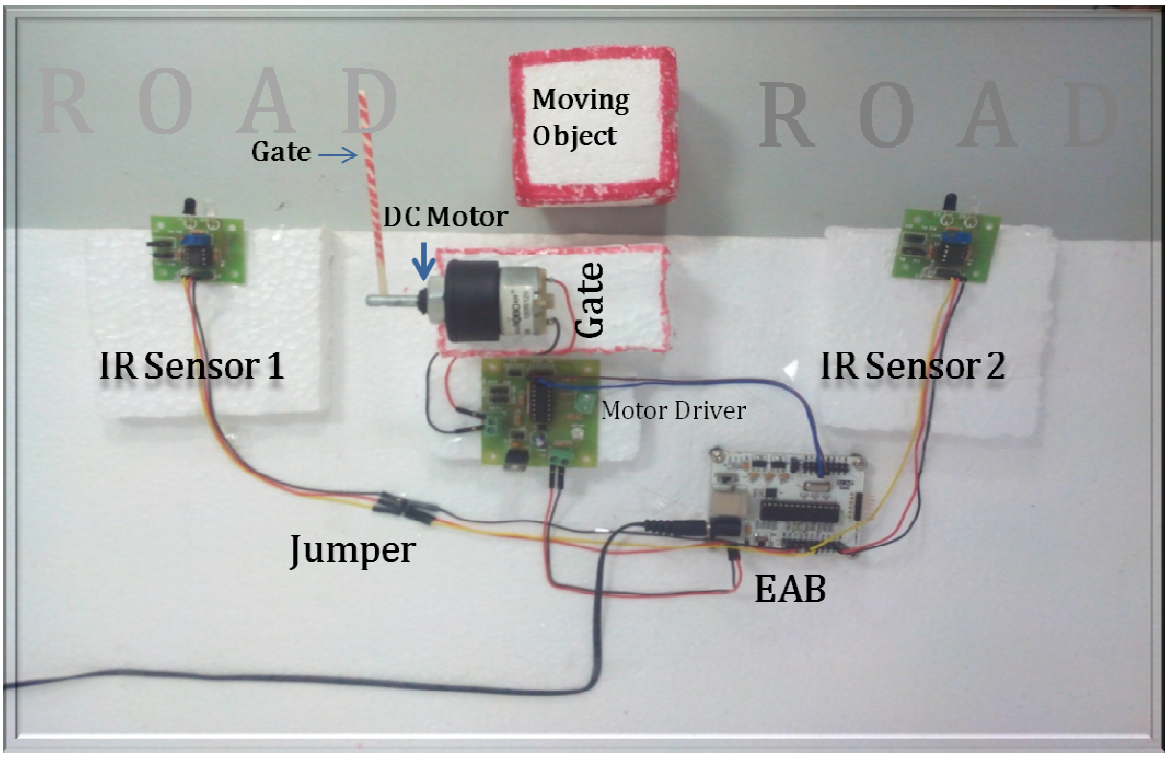


Connection Description

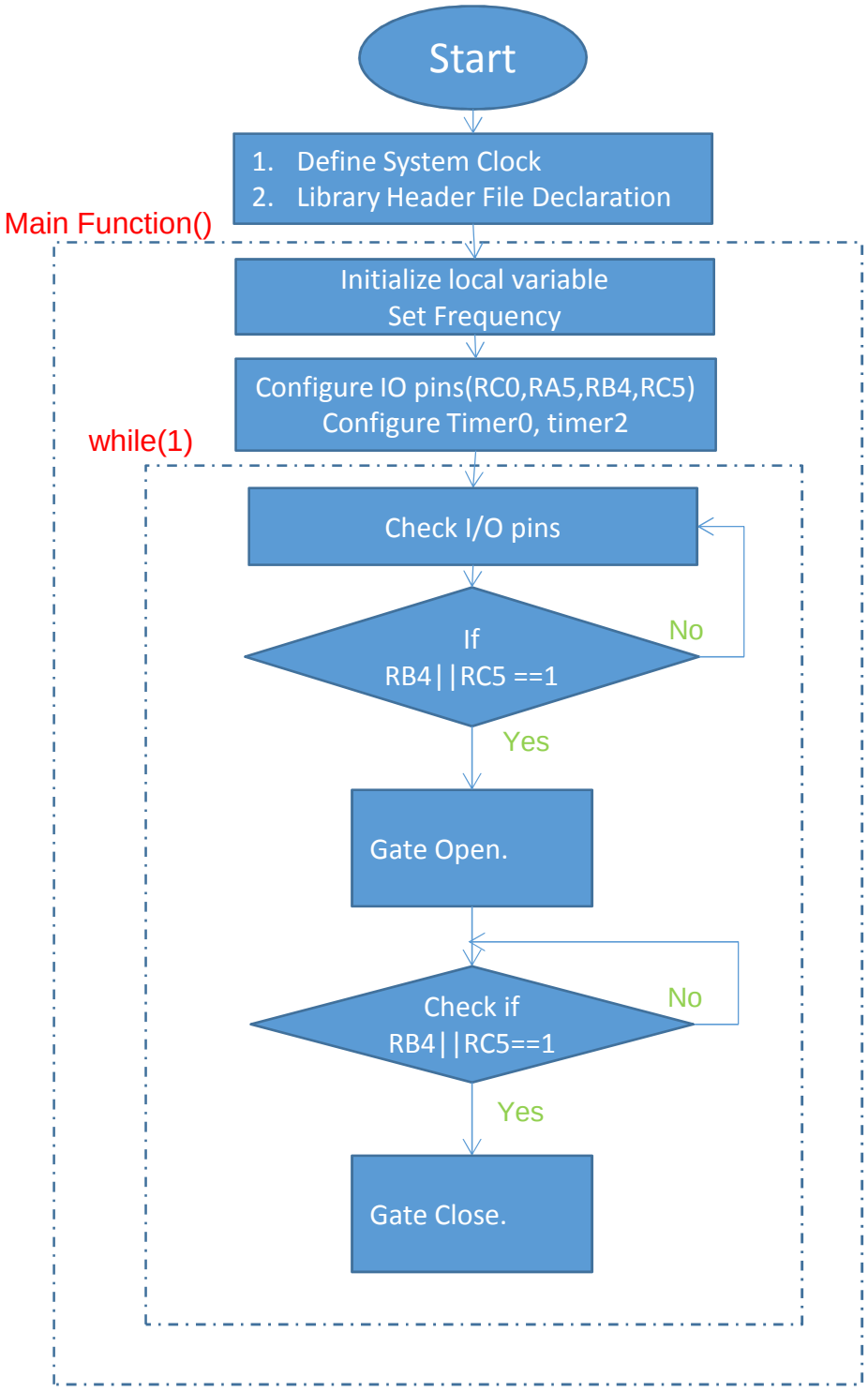
In this project we have used two IR Sensor modules as object/person detector. One is placed in the Left Hand Side and another is placed in the Right Hand Side of the gate. Each module has 3-pins. These are GND, VCC and DATA. The GND and VCC pins are connected to the GND and VCC pins of an EAB. The DATA pin of IR Sensor1 is connected to RB4 pin of EAB while that of IR Sensor2 is connected to RC5 pin of EAB.

The outputs are from the pins RC0 and RA5 of the EAB. These pins are responsible to control the gate and they are connected to the IN0 and IN1 pins of the Motor Driver board, respectively. Both the Motor Driver Board and the EAB are powered with 9V DC source.

Note: *Any GPIO pin can be used for this project. Set the particular GPIO pins to output and provide a high /low signal to the pin .*



Code Flow Chart



Application Notes

Source Code

The Source code shown below is the firmware to be flashed in the microcontroller of the Embedded Application Board. The Source code is commented for better understanding of the user.

Refer to the EAB User Guide and EAB Programming Guide for more details on how to Flash(burn) program(Source Code) in the microcontroller of Embedded Application Board.

```
#define SYS_CLK 8000000         //Required for delay macro functions
                                //Default 1MHZ, else change as per configuration
/*** INCLUDE STANDARD HEADERS & LIBRARY ***/
#include <stdio.h>
#include <stdlib.h>

#include "EAB_Library.h"

/*** GLOBAL VARIABLES ****/

/*-------------------------------------------------------------------------*/
void main(void)
{
    /*** LOCAL VARIABLES ***/
    uint i=0;

    /*** INTITALIZE OSCILLATOR, PERIPHERAL & HARDWARE ***/
    Oscillator.SetFreq_8MHZ();                  // Select system clock at 8 MHz

    Timer0.SetPeriod(Timer0.config.PRESCALER_128,Timer0.config.COUNTER_16BIT) ;
                                                // Set Timer0 period with 4 sec
    Timer2.SetPeriod(Timer2.config.PRESCALER_16,Timer2.config.POSTSCALER_16,255);
                                        // set Timer2 period with maximum value

    PinDigitalOut(RC0);                         // RC0 as digital output
    PinDigitalOut(RA5);                         // RA5 as digital output
    PinDigitalIn(RB4);                          // RB4 as digital output
    PinDigitalIn(RC5);                          // RC5 as digital output
```

```
    /*** PLACE THE REPETITIVE TASKS IN THIS LOOP ***/
    while(1)
    {
        if((PinRead.RB4 || PinRead.RC5)==1)          // Check for logic high at
                                                     // RB4 or RC5
        {
            PinWrite.RC0=LOW;                        // Set RC0 output Low
            PinWrite.RA5=HIGH;                       // Set RA5 output Low
            for(i=0;i<10;i++)
            {
                Timer2_Flag=0;
                while(Timer2_Flag ==0);              // Wait for timer2 flag
            }
            PinWrite.RC0=LOW;                        // Set RC0 output Low
            PinWrite.RA5=LOW;                        // Set RA5 output Low
            Timer0_Flag=0;
            while(Timer0_Flag ==0);                  // Wait for timer0 flag

            while((PinRead.RB4 || PinRead.RC5)==0); // Check for logic low at
                                                     // RB4 or RC5
            PinWrite.RC0=HIGH;                       // Set RC0 output High
            PinWrite.RA5=LOW;                        // Set RA5 output Low
            for(i=0;i<10;i++)
            {
                Timer2_Flag=0;
                while(Timer2_Flag ==0);              // Wait for timer2 flag
            }
            PinWrite.RC0=LOW;                        // Set RC0 output Low
            PinWrite.RA5=LOW;                        // Set RA5 output Low
            Timer0_Flag =0;
            while(Timer0_Flag ==0);                  // Wait for timer0 flag
        }
    }
}
/*------------------------------End Of Code-----------------------------------*/
```

Application Notes

How to Operate

Follow the steps mentioned below in order to operate the project…

- Prepare an arena of your choice
- Flash the code into the microcontroller.
- Connect each and every part properly.
- Place the robot in the arena.
- Power the EAB, Sensor Board and Motor Driver Circuit with 9V/12V DC. Carefully check the polarities and then connect .
- Switch ON the EAB.

Now you can see the gate is opening/closing automatically whenever a person passes through this gate.

More Projects

Various other applications can be built using Smart Gate.

Some of such applications are given below:

- Industrial Automation
- Security System