```
#include <avr/io.h>
#include <util/delay.h>

#include "lib/adc/adc.h"
#include "lib/motor/motor.h"
#include "lib/led/led.h"

#define SENSOR_THRES 800

//Map Sensor Number to ADC Channel
#define SENSOR1 0
#define SENSOR2 1
#define SENSOR3 2
#define SENSOR4 3
#define SENSOR5 4

//Gloabal varriables
float pGain = 200;   //Proportional Gain
float iGain =  0.2;  //Integral Gain
float dGain =  120;  //Differential Gain
int delay = 10;

int32_t eInteg = 0;  //Integral accumulator
int32_t ePrev  =0;      //Previous Error


void  DelayMs(uint8_t ms);
float ReadSensors();
float PID(float cur_value,float req_value);


float control;
float s;

int main(void)
{
  //Initialize Motors subsystem.
  MotorInit();

  //Initialize LED subsystem
  LEDInit();

  //Initialize Analog to Digital Converter (ADC)
  InitADC();


   while(1)
   {
    //Previous Sensor Reading
```

```
float sprev;

//Take current sensor reading
//return value is between 0 to 5
//When the line is towards right of center then value tends to 5
//When the line is towards left of center then value tends to 1
//When line is in the exact center the the valeue is 3
  s=ReadSensors();

//If line is not found beneath any sensor, use last sensor value.
if(s==0xFF)
{
   s=sprev;
}

//PID Algorithm generates a control variable from the current value
//and the required value. Since the aim is to keep the line always
//beneath the center sensor so the required value is 3 (second parameter)
//The first argument is the current sensor reading.
//The more the difference between the two greater is the control variable.
//This control variable is used to produce turning in the robot.
//When current value is close to required value is close to 0.
control = PID(s,3.0);

//Limit the control
if(control > 510)
   control = 510;
if(control < -510)
   control = -510;

if(control > 0.0)//the left sensor sees the line so we must turn right
{
   if(control>255)
      MotorA(MOTOR_CW,control-255);
   else
      MotorA(MOTOR_CCW,255-control);

   MotorB(MOTOR_CW,255);
}
if(control <= 0.0)//the right sensor sees the line so we must turn left
{
   if(control<-255)
      MotorB(MOTOR_CCW,-(control+255));
   else
      MotorB(MOTOR_CW,255+control);

   MotorA(MOTOR_CCW,255);
}

//Delay
```

```
      DelayMs(delay);

      sprev=s;
    }
}

void DelayMs(uint8_t ms)
{
  uint8_t i;
  for(i=0;i<ms;i++)
  {
    _delay_ms(1);
  }
}

//Implements PID control
float PID(float cur_value,float req_value)
{
 float pid;
 float error;

 error = req_value - cur_value;
 pid = (pGain * error)  + (iGain * eInteg) + (dGain * (error - ePrev));

 eInteg += error;              // integral is simply a summation over time
 ePrev = error;                // save previous for derivative

 return pid;
}

float ReadSensors()
{
  uint16_t eright,right,middle,left,eleft;
  uint8_t     sensor1,sensor2, sensor3, sensor4,sensor5;

  float avgSensor = 0.0;

  eright=ReadADC(SENSOR5);
  if(eright>SENSOR_THRES)//Right black line sensor
  {
    sensor5 = 1;
    LEDOn(5);
  }
  else
  {
    sensor5 = 0;
    LEDOff(5);
  }

  // Read analog inputs
```

```
right=ReadADC(SENSOR4);
if(right>SENSOR_THRES)//Right black line sensor
{
  sensor4 = 1;
  LEDOn(4);
}
else
{
  sensor4 = 0;
  LEDOff(4);
}

middle=ReadADC(SENSOR3);
if(middle>SENSOR_THRES)// Middle black line sensor
{
  sensor3 = 1;
  LEDOn(3);
}
else
{
  sensor3 = 0;
  LEDOff(3);
}

left=ReadADC(SENSOR2);
if(left>SENSOR_THRES)// Left black line sensor
{
  sensor2 = 1;
  LEDOn(2);
}
else
{
  sensor2 = 0;
  LEDOff(2);
}

eleft=ReadADC(SENSOR1);
if(eleft>SENSOR_THRES)// Left black line sensor
{
  sensor1 = 1;
  LEDOn(1);
}
else
{
  sensor1 = 0;
  LEDOff(1);
}


if(sensor1==0 && sensor2==0 && sensor3==0 && sensor4==0 && sensor5==0)
```

```
  {
    return 0xFF;
  }

  // Calculate weighted mean
  avgSensor = (float) sensor1*1 + sensor2*2 + sensor3*3 + sensor4*4 + sensor5*5 ;
  avgSensor = (float) avgSensor / (sensor1 + sensor2 + sensor3 + sensor4 + sensor5);

  return avgSensor;
}
```