PROGRAMMING CODES FOR ATMEGA-8 AVR BOARD

(CONTROLLING A ROBOT)

BLACK LINE FOLLOWER :

```
#include<avr/io.h>
void main()
{
    DDRD=0b11111111;  // set PD4 as output bit
    DDRC=0b0000000;   // set PORTC as input port
    DDRB=0b00011110;  // PB1, PB2, PB3, PB4 as output port
 int ls=0, rs=0, a=1;   // define & initialize ls, rs integer as 0 to
                      // acquire the left sensor status in ls and right sensor
                      // status in rs

    while(1)          // create infinite loop
    {
    rs=(PINC&0b0000001);    // acquire only left sensor status connected at PC0
    ls=(PINC&0b0001000);    // acquire only right sensor status connected at PC3
    PORTD = ~PORTD;
        if((rs==0b0000001)||(ls==0b0001000))
        {
        PORTD=(1<<4);
        }
        if((rs==0b0000000)&(ls==0b0000000)) // check sensor status for both sensor OFF
        {

            PORTB=0b00011110;  // stop
            ls=0;              // set sensor status off
            rs=0;              // set sensor status off

        }
        if((rs==0b0000001)&(ls==0b0000000))  // check sensor status for left sensor=ON and
                                  // right sensor=OFF
        {
            PORTB=0b00010000;    // turn right
            PORTD =(1<<4);
            ls=0;               // set sensor status off
            rs=0;                                   // set sensor status off
        }

        if((rs==0b0000000)&(ls==0b0001000))  //check sensor status for left sensor=OFF and
                                  // right sensor=ON
        {


            PORTB=0b00000010;     //turn left
            PORTD =(1<<4);
            ls=0;                // set sensor status off
            rs=0;                                    // set sensor status off


        }

    if((rs==0b0000001)&&(ls==0b0001000) ) // check sensor status for both sensor ON
        {
            PORTB=0b00010010;    //move forward
            PORTD =~PORTD;
            ls=0;              //set sensor status off
            rs=0;              //set sensor status off
            }
            }
}
```

DTMF CONTROL :

```
#define F_CPU 12000000UL
#include<avr/io.h>
#include "robosapiens.c"
int main(void)
{
int d=0;
int b=0;
DDRB=0b00011110;  //PB1, PB2, PB3, PB4 as output bits connected to motors and PB0 as input bit
connected to DTMF decoder IC
DDRD=0b00010000;  //PD7, PD6, PD5 connected to DTMF decoder IC hence input bits and
                                        //PD4 connected to buzzer hence output bit.

while(1)        // infinite loop
{
b=PINB&0b00000001;
d=PIND&0b11100000;
PORTD &= ~(1<<4);

if(d==0b10000000 && b==0b00000000)        //if Key 2 of cell phone pressed
{
PORTB=0b00010010;    // move straight
}

if(d==0b01000000 && b==0b00000000)        // if Key 4 of cell phone pressed
{
PORTB=0b000010000;    // turn left
PORTD = (1<<4);
wait(0.2);
```

```
PORTD &= ~(1<<4);

wait(0.2);

}

if(d==0b11000000 && b==0b00000000)        // if Key 6 of cell phone pressed

{

PORTB=0b00000010;   // turn right

PORTD = (1<<4);

wait(0.2);

PORTD &= ~(1<<4);

wait(0.2);

}

if(d==0b00100000 && b==0b00000000)        // if Key 8 of cell phone pressed

{

PORTB=0b00001100;    // move back

}


if(d==0b00000000 && b==0b00000001)                              // if key 1 of cell phone pressed

{

PORTB=0b000010000; // left turn

PORTD = (1<<4);

wait(0.5);

PORTB=0b00010010;   // move forward

wait(10);

}

if(d==0b10000000 && b==0b00000001)                              // if key 3 of cell phone pressed

{

PORTB=0b00000010;   // right turn

PORTD = (1<<4);

wait(0.5);
```

```
PORTB=0b00010010;   // move forward
wait(10);
}

if(d==0b11000000 && b==0b00000001)                    // if key 7 of cell phone pressed
{
PORTB=0b00001000;   // left turn in backward direction
PORTD = (1<<4);
wait(0.5);
PORTB=0b00001100;   // move back
wait(10);
}

if(d==0b00100000 && b==0b00000001)                    // if key 9 of cell phone
{
PORTB=0b00000100;   // right turn in backward direction
PORTD = (1<<4);
wait(0.5);
PORTB=0b00001100;   // move back
wait(10);
}

if(d==0b01000000 && b==0b00000001)      // if Key 5 of cell phone pressed
{
PORTB=0b00000000;    //stop
}

}   //while closed
}   //main closed
```

WALL FOLLOWER :

```
#define F_CPU 12000000UL

#include<avr/io.h>

int main(void)

{

        DDRD=0b11111111;          // set PD4 as output bit

        DDRC=0b0000000;           // set PORTC as input port

        DDRB=0b00011110;          // PB1, PB2, PB3, PB4 as output port

   int rs=0;                          // define & initialize  rs integer as 0 to acquire the right sensor status
in rs


        while(1)                   // create infinite loop

        {


        rs=(PINC&0b0000001);      //acquire only right sensor status connected at PC0

        PORTD &= ~(1<<4);


                if((rs==0b0000000))    //check  right sensor status for OFF

                {

                        PORTD=(1<<4);

                        PORTB=0b00010000;  //right turn

                        rs=0;           //set sensor status off


                }


                else

                        PORTB=0b00000010;  //left turn

                        }

}
```

BLINKING LEDS :

```
#define F_CPU 120000000UL
#include<avr/io.h>
#include<util/delay.h>
#include"robosapiens.c"
int main(void)
{

DDRD=0b11111111;  // set PD4 as output bit
DDRB=0b00011110;  // PB1,PB2,PB3 and PB4 of PORTB are set as output.
while(1)                    // infinite while loop

{
          PORTD = ~PORTD;

   PORTB=0b00011110;  //PB1,PB2,PB3 and PB4 Led's are set ON

   wait(.5);        // wait function defined in robosapiens.c file function argument: time in seconds

          PORTD =(1<<4);
          PORTB=0b00000000; // PB1,PB2,PB3 and PB4 Led's are set OFF
   wait(.5);                      // wait function defined in robosapiens.c file function argument: time in seconds
          }

}
```

EDGE AVOIDER :

```
#define F_CPU 12000000UL
#include<avr/io.h>
#include<util/delay.h>
#include "robosapiens.c"
int main(void)
{
        DDRD=0b11111111;                        //set PD4 as output bit
        DDRC=0b0000000;                         //set PORTC as input port
        DDRB=0b00011110;                        //PB1, PB2, PB3, PB4 as output port
   int ls=0, rs=0;                      // define & initialize ls, rs integer as 0 to
                                                                // acquire the left sensor status in ls and
right sensor
                                                                // status in rs
        while(1)                        // create infinite loop
        {

        rs=(PINC&0b0000001);                    //acquire only left sensor status connected at PC0
        ls=(PINC&0b0001000);                    // acquire only right sensor status connected at PC3
        PORTD = ~PORTD;

                if((rs==0b0000000)||(ls==0b0000000))
                {
                PORTD=(1<<4);
                }

                if((rs==0b0000000)&&(ls==0b0000000)) //check sensor status for both sensor OFF
                {
```

```
        PORTB=0b00000000;   //stop
        PORTD = (1<<4);
        PORTB=0b00001100;   //backward
        wait(.8);
PORTB=0b00000010;   //turn right or user can define their own turn
        wait(.8);
        ls=0;           //set sensor status off
        rs=0;           //set sensor status off

}

if((rs==0b0000001)&&(ls==0b0000000))  //check sensor status for left sensor=ON and
                      // right sensor=OFF
{

        PORTD = (1<<4);
        PORTB=0b00001100;    //backward
        wait(.8);
        PORTB=0b00010000;    //turn right to avoid the edge
        wait(.8);
        ls=0;          //set sensor status off
        rs=0;                            //set sensor status off

}


if((rs==0b0000000)&&(ls==0b0001000))  //check sensor status for left sensor=OFF and
                      // right sensor=ON
{
```

```
                        PORTD = (1<<4);

                        PORTB=0b00001100;    //backward

                        wait(.8);

                        PORTB=0b00000010;    //turn left to avoid the edge

                        wait(.8);

                        ls=0;                //set sensor status off

                        rs=0;                                  ///set sensor status off


                }




    if((rs==0b0000001)&&(ls==0b0001000)) //check sensor status for both sensor ON

                {


                        PORTB=0b00010010;  //move forward

                        ls=0;              //set sensor status off

                        rs=0;              //set sensor status off


                }




        }


}
```