```c
/*
 * hissen.c
 *
 * Created: 2016-04-26 11:44:02
 *  Author: digpi17
 */


#include <avr/io.h>
#include <stdio.h>
#include <avr/interrupt.h>
#include <avr/delay.h>


int alarm = 0;
int stop = 1;
int inMotion = 0;
int goingUp = 0;
int goingDown = 0;
int currentFloor = 0;
int previousFloor = 0;
int targetFloor = 0;
int hissQ[5] = {0, 0, 0, 0, 0};
int queueDown[4] = {0, 0 ,0 ,0};
int queueUp[4] = {0, 0, 0, 0};
int sensorVect[5] = {0, 0, 0, 0, 0};
int prio[8] = {0,0,0,0,0,0,0,0};
int direction;
int waitTime;
volatile uint8_t tot_overflow;

char LedHall = 0;
char LedHiss = 0;
char sensorInput = 0;
char hallInput = 0;
char hissInput = 0;
int start = 1;
int nodStop = 0;

char temp = 0;
char disp = 0;

void Startupmode();
void checkTKinHall();
void executeQueue();
void CheckTKinHiss();
void CheckSensors();
void goDown();
void goUp();
void dispUpdate();
void addToPrio();
void removePrio();
void removeSpecificPrio();
void checkLogicinHall();
void refreshPrio();
void hissUpdate();

int main(void)
{
    Startupmode();
```

```
        currentFloor = 4;
        //targetFloor = 5;
        while(start)
        {

            checkTKinHall();
            CheckTKinHiss();
            while(nodStop){

            }
            executeQueue();
            hissUpdate();
            while(waitTime){
                CheckTKinHiss();
                checkTKinHall();
                refreshPrio();

            }
            TCCR1B = 0;
            cli();
            //start = 1;

        }
    }

    void timer1_init(void){
        TCCR1B |= (1 << CS11);
        TCNT1 = 0;
        TIMSK |= (1 << TOIE1);
        sei();
        tot_overflow = 0;
    }

    ISR(TIMER1_OVF_vect){
        tot_overflow++;
        if(tot_overflow >= 7){
            waitTime = 0;
            tot_overflow = 0;
        }
    }

    void checkTKinHall(void){
        DDRB = 0b11100000; //S‰tt PB5 och PB6 och PB7 som output
        PORTB = 0b10100000; // s‰tt PB6 som output 1 och PB5 som output 0. Port D
            l‰ser nu frÂn hallen.
        hallInput = PIND;

        LedHall = hallInput | LedHall;
        PORTC = 0b00000001;
        PORTA = LedHall;



        if((hallInput & 0b00000001) != 0){
            queueUp[0] = 1;
            addToPrio(1);
        }

        if((hallInput & 0b00000010) != 0){
            queueDown[0] = 1;
            addToPrio(2);


        }

        if((hallInput & 0b00000100) != 0){
            queueUp[1] = 1;
            addToPrio(3);


        }

        if((hallInput & 0b00001000) != 0){
            queueDown[1] = 1;
            addToPrio(4);
        }

        if((hallInput & 0b00010000) != 0){
            queueUp[2] = 1;
            addToPrio(5);

        }

        if((hallInput & 0b00100000) != 0){
            queueDown[2] = 1;
            addToPrio(6);

        }

        if((hallInput & 0b01000000) != 0){
            queueUp[3] = 1;
            addToPrio(7);

        }

        if((hallInput & 0b10000000) != 0){
            addToPrio(8);
            queueDown[3] = 1;

        }

        PORTB = 0b11100000;
        PORTC = 0b00000000;

    }

    void executeQueue(void){
        if(inMotion == 0 ){ //&& NÂgot som ser till att denna inte kˆrs om vi l
           ‰ser frÂn hallen ist‰llet -> remove kˆrs fˆr ofta//
            int x = prio[0];
            switch (x)
            {

                case 1:
                targetFloor = 1;
                direction = 1;
                //removePrio(); // borde kanske ta bort denna funktionen, ist‰llet
                    gˆr vi sÂ att den fˆrst tas ur prion n‰r detta faktiskt ‰r
                    uppnÂtt. AlltsÂ tas prion bort dÂ vi stannar pÂ fˆrsta
                    vÂningen. PÂ sÂ vis kan vi inte lyckas fylla pÂ med en 1a i
                    prioque direkt.
                break;

                case 2:
                targetFloor = 2;
                direction = 0;
                //removePrio();
                break;

                case 3:
                targetFloor = 2;
                direction = 1;
                //removePrio();
                break;

                case 4:
                targetFloor = 3;
                direction = 0;
                //removePrio();
                break;

                case 5:
                targetFloor = 3;
                direction = 1;
                //removePrio();
                break;

                case 6:
                targetFloor = 4;
                direction = 0;
                //removePrio();
                break;

                case 7:
                targetFloor = 4;
                direction = 1;
                //removePrio();
                break;

                case 8:
                targetFloor = 5;
                direction = 0;
                //removePrio();
                break;
            }

        if(currentFloor > targetFloor && targetFloor != 0){
            stop = 1;
            goDown();
        }else if( targetFloor > currentFloor && targetFloor != 0){
            stop = 1;
            goUp();
        }

        }

    }

    void checkLogicinHall(void){
        if(goingDown == 1 && inMotion == 1){
            int i = 0;
            int v;
            for(i = 0; i < 8; i = i +1){
                v = prio[i];
                switch(v){
                    case 2:
                    if(currentFloor > 2 && targetFloor < 2){
                        targetFloor = 2;
                    }
                    break;

                    case 4:
                    if(currentFloor > 3 && targetFloor < 3){
                        targetFloor = 3;
                    }
                    break;

                    case 6:
                    if(currentFloor > 4 && targetFloor < 4){
                        targetFloor = 4;
                    }
                    break;
                }

            }

        }

        if(goingUp == 1 && inMotion){
            int i = 0;
            int v;
            for(i = 0; i < 8; i = i +1){
                v = prio[i];
                switch(v){
                    case 3:
                    if(currentFloor < 2 && targetFloor > 2){
                        targetFloor = 2;
                    }
                    break;

                    case 5:
                    if(currentFloor < 3 && targetFloor > 3){
                        targetFloor = 3;
                    }
                    break;

                    case 7:
                    if(currentFloor < 4 && targetFloor > 4){
                        targetFloor = 4;
                    }
                    break;
                }
            }
        }
    }


    void CheckTKinHiss(void){
        DDRB = 0b11100000;
```

```
        PORTB = 0b11000000;
        hissInput = PIND;

        hissInput = hissInput << 3;
        hissInput = hissInput >> 3;

        LedHiss = hissInput | LedHiss;
        PORTC = 0b00000010;
        PORTA = LedHiss;

        hissInput = PIND;

        if((hissInput & 0b00000001) != 0){
            hissQ[0] = 1;
        }

        if((hissInput & 0b00000010) != 0){
            hissQ[1] = 1;
        }

        if((hissInput & 0b00000100) != 0){
            hissQ[2] = 1;
        }

        if((hissInput & 0b00001000) != 0){
            hissQ[3] = 1;
        }

        if((hissInput & 0b00010000) != 0){
            hissQ[4] = 1;
        }

        if((hissInput & 0b00100000) != 0){
            nodStop = 1;
            stop = 0;
        }

        if((hallInput & 0b01000000) != 0){
            alarm = 1;
        }

        PORTB = 0b11100000;
        PORTC = 0b00000000;
    }

    void CheckSensors(void){
        DDRB = 0b11100000;
        sensorInput = PINB;

        char temp;
        temp = sensorInput << 3;
        sensorInput = temp >> 3;


        if((sensorInput & 0b00000001 ) != 0){
            sensorVect[0] = 1;
        }else{
            sensorVect[0] = 0;
        }
        if((sensorInput & 0b00000010) != 0 && goingUp == 1){
            sensorVect[1] = 1;
            if(previousFloor == 2 || previousFloor == 0){
            previousFloor = 1;
            }
        }else if((sensorInput & 0b00000010) != 0 && goingDown == 1){
            sensorVect[1] = 1;
            if(previousFloor == 1 || previousFloor == 3 || previousFloor == 0){
            previousFloor = 2;
            }
        }else{
            sensorVect[1] = 0;
        }

        if((sensorInput & 0b00000100) != 0 && goingUp == 1){
            sensorVect[2] = 1;
            if(previousFloor == 1 || previousFloor == 3 || previousFloor == 0){
            previousFloor = 2;
            }
        }else if((sensorInput & 0b00000100) != 0 && goingDown == 1){
            sensorVect[2] = 1;
            if(previousFloor == 4 || previousFloor == 2 || previousFloor == 0){
            previousFloor = 3;
            }
        }else{
            sensorVect[2] = 0;
        }

        if((sensorInput & 0b00001000) != 0 && goingUp == 1){
            sensorVect[3] = 1;
            if(previousFloor == 2 || previousFloor == 4 || previousFloor == 0){
            previousFloor = 3;
            }
        }else if((sensorInput & 0b00001000) != 0 && goingDown == 1){
            sensorVect[3] = 1;
            if(previousFloor == 5 || previousFloor == 3 || previousFloor == 0){
            previousFloor = 4;
            }
        }else{
            sensorVect[3] = 0;
        }


        if((sensorInput & 0b00010000) != 0 && goingUp == 1){
            sensorVect[4] = 1;
            if(previousFloor == 3 || previousFloor == 0){
            previousFloor = 4;
            }
        }else if((sensorInput & 0b00010000) != 0 && goingDown == 1){
            sensorVect[4] = 1;
            if(previousFloor == 4 || previousFloor == 0){
            previousFloor = 5;
            }

        }else{
            sensorVect[4] = 0;
        }

        if(sensorVect[1] == 0 && goingUp == 1 && previousFloor == 1 &&
            currentFloor == 1){
```

```
            currentFloor = 2;
        }else if( sensorVect[1] == 0 && goingDown == 1 && previousFloor == 2 &&
            currentFloor == 2){
            currentFloor = 1;
        }

        if(sensorVect[2] == 0 && goingUp == 1 && previousFloor == 2 &&
            currentFloor == 2){
            currentFloor = 3;
        }else if( sensorVect[2] == 0 && goingDown == 1 && previousFloor == 3 &&
            currentFloor == 3){
            currentFloor = 2;
        }

        if(sensorVect[3] == 0 && goingUp == 1 && previousFloor == 3 &&
            currentFloor == 3){
            currentFloor = 4;
        }else if( sensorVect[3] == 0 && goingDown == 1 && previousFloor == 4 &&
            currentFloor == 4){
            currentFloor = 3;
        }

        if(sensorVect[4] == 0 && goingUp == 1 && previousFloor == 4 &&
            currentFloor == 4){
            currentFloor = 5;
        }else if( sensorVect[4] == 0 && goingDown == 1 && previousFloor == 5 &&
            currentFloor == 5){
            currentFloor = 4;
        }

        if(sensorVect[0] == 1){
            stop = 0;
        }

        if((currentFloor == targetFloor) && (sensorVect[0] == 0) && (sensorVect[1]
            == 0)  && (sensorVect[2] == 0) && (sensorVect[3] == 0) && (sensorVect
            [4] == 0)){
            stop = 0;
            waitTime = 1;
            int f = currentFloor;
            switch(f){
                PORTC = 0b00000010;
                case 1:
                LedHiss = 0b11111110 & LedHiss;
                PORTA = LedHiss;
                break;

                case 2:
                LedHiss = 0b11111101 & LedHiss;
                PORTA = LedHiss;
                break;

                case 3:
                LedHiss = 0b11111011 & LedHiss;
                PORTA = LedHiss;
                break;

                case 4:
                LedHiss = 0b11110111 & LedHiss;
                PORTA = LedHiss;
                break;

                case 5:
                LedHiss = 0b11101111 & LedHiss;
                PORTA =  LedHiss;
                break;
            }

            PORTC = 0b00000000;
            PORTA = 0b00000000;

            if(goingDown == 1 && goingUp == 0){
                int f = currentFloor;
                switch(f){
                    PORTC = 0b00000001;
                    case 1:
                    LedHall = 0b11111110 & LedHall;
                    PORTA = LedHall;
                    break;

                    case 2:
                    LedHall = 0b11111101 & LedHall;
                    PORTA = LedHall;
                    break;

                    case 3:
                    LedHall = 0b11110111 & LedHall;
                    PORTA = LedHall;
                    break;

                    case 4:
                    LedHall = 0b11011111 & LedHall;
                    PORTA = LedHall;
                    break;

                }
            }

            if(goingDown == 0 && goingUp == 1){
                 int f = currentFloor;
                 switch(f){
                     PORTC = 0b00000001;
                     case 2:
                     LedHall = 0b11111011 & LedHall;
                     PORTA = LedHall;
                     break;

                     case 3:
                     LedHall = 0b11101111 & LedHall;
                     PORTA = LedHall;
                     break;

                     case 4:
                     LedHall = 0b10111111 & LedHall;
                     PORTA = LedHall;
                     break;

                     case 5:
                     LedHall = 0b01111111 & LedHall;
                     PORTA = LedHall;
```

```
                    break;
                }
            }

            PORTC = 0b00000000;
            PORTA = 0b00000000;
        }

        PORTB = 0b11100000;
        PORTC = 0b00000000;
        timer1_init();
    }

    void Startupmode(void){
        DDRB = 0b11100000;
        DDRC = 0b01000011;
        DDRD = 0b00000000;
        DDRA = 0b11111111;

        PORTC = 0b00000001;
        PORTA = 0b00000000;
        PORTC = 0b00000010;
        PORTA = 0b00000000;
        PORTC = 0b01000000;
        PORTA = 0b00000000;

    }

    void goDown(void){
        goingDown = 1;
        inMotion = 1;
        goingUp = 0;
        if(stop == 1){
        while(stop){

            PORTC = 0b01000000;

            PORTA = 0b00000110;
            _delay_ms(10);

            PORTA = 0b00001010;
            _delay_ms(10);

            PORTA = 0b00001001;
            _delay_ms(10);

            PORTA = 0b00000101;
            _delay_ms(10);

            checkTKinHall();
            CheckTKinHiss();
            CheckSensors();
            dispUpdate();
            hissUpdate();
            checkLogicinHall();

            PORTB = 0b11100000;
            PORTC = 0b00000000;
        }
        int i;
        for(i = 0; i < 15; i = i + 1){

            PORTC = 0b01000000;

            PORTA = 0b00000110;
            _delay_ms(10);

            PORTA = 0b00001010;
            _delay_ms(10);

            PORTA = 0b00001001;
            _delay_ms(10);

            PORTA = 0b00000101;
            _delay_ms(10);

            checkTKinHall();
            CheckTKinHiss();
            dispUpdate();

            PORTB = 0b11100000;
            PORTC = 0b00000000;
        }
        }
        inMotion = 0;
        }

    void goUp(void){
        goingUp = 1;
        inMotion = 1;
        goingDown = 0;
        while(stop){

            PORTC = 0b01000000;

            PORTA = 0b00000101;
            _delay_ms(10);


            PORTA = 0b00001001;
            _delay_ms(10);


            PORTA = 0b00001010;
            _delay_ms(10);


            PORTA = 0b00000110;
            _delay_ms(10);

            checkTKinHall();
            CheckTKinHiss();
            CheckSensors();
            dispUpdate();
            hissUpdate();
            checkLogicinHall();

            PORTB = 0b11100000;
            PORTC = 0b00000000;
        }
        inMotion = 0;
    }

    void dispUpdate(void){
        PORTC = 0b00000010;
        char temp;
        temp = LedHiss << 3;
        LedHiss = temp >> 3;


        if(currentFloor == 1){
            LedHiss = LedHiss | 0b00100000;
            PORTA = LedHiss;
        }

        if(currentFloor == 2){
            LedHiss = LedHiss | 0b01000000;
            PORTA = LedHiss;
        }

        if(currentFloor == 3){
            LedHiss = LedHiss | 0b01100000;
            PORTA = LedHiss;
        }

        if(currentFloor == 4){
            LedHiss = LedHiss | 0b10000000;
            PORTA = LedHiss;
        }

        if(currentFloor == 5){
            LedHiss = LedHiss | 0b10100000;
            PORTA = LedHiss;
        }

        PORTB = 0b11100000;
        PORTC = 0b00000000;
        PORTA = 0b00000000;
    }

    void addToPrio(int x){

    int count;
    int prioValue;
    for(count = 0; count < 8; count = count + 1){
        prioValue = prio[count];
        if(prioValue == x){
            count = 8;
        }
        if(prioValue == 0){
            prio[count] = x;
            count = 8;
        }
    }

    }

    void removePrio(void){
    int temp[8] = {0, 0, 0, 0, 0, 0, 0, 0};
    int t = 0;
    int l;
    for(l = 1; l < 8; l = l + 1){
        t = prio[l];
        temp[l-1] = t;
    }

    int j;
    for(j = 0; j < 8; j = j + 1){
        t = temp[j];
        prio[j] = t;
    }

     }

    void removeSpecificPrio(int pos){
        int i;
        int tempValue;
        int tempV[8] = {0,0,0,0,0,0,0,0};
        for(i = 0; i < pos; i = i +1){
            tempValue = prio[i];
            tempV[i] = tempValue;
        }

        for(i = pos + 1; i < 8; i = i + 1){
            tempValue = prio[i];
            tempV[i-1] = tempValue;
        }

        for(i = 0; i < 8; i = i + 1){
            tempValue = tempV[i];
            prio[i] = tempValue;
        }

    }

    void test(void){
        currentFloor = 2;
        targetFloor = 5;
        goUp();
        _delay_ms(500);
        stop = 1;
        currentFloor = 5;
        targetFloor = 1;
        goDown();
        _delay_ms(500);
        stop = 1;
        currentFloor = 1;
        targetFloor = 2;
        goUp();
        _delay_ms(500);
        stop = 1;
        currentFloor = 2;
        targetFloor = 4;
        goUp();
        _delay_ms(500);
        stop = 1;
        currentFloor = 4;
        targetFloor = 2;
        goDown();
    }

    void refreshPrio(void){
        int i;
        int e;
        for(i = 0; i < 8; i = i + 1){
            e = prio[i];
            if(goingDown == 1){
                switch(e){
                    case 1:
                    if(currentFloor == 1){
                        removeSpecificPrio(i);
                        hissQ[0] = 0;
                    }
                    break;

                    case 2:
                    if(currentFloor == 2){
                        removeSpecificPrio(i);
                        hissQ[1] = 0;
                    }
                    break;

                    case 4:
                    if(currentFloor == 3){
                        removeSpecificPrio(i);
                        hissQ[2] = 0;
                    }
                    break;

                    case 6:
                    if(currentFloor == 4){
                    removeSpecificPrio(i);
                    hissQ[3] = 0;
                    }
                    break;
                }
            }

            if(goingUp == 1){
                switch(e){
                    case 3:
                    if(currentFloor == 2){
                        removeSpecificPrio(i);
                        hissQ[1] = 0;
                    }
                    break;

                    case 5:
                    if(currentFloor == 3){
                        removeSpecificPrio(i);
                        hissQ[2] = 0;
                    }
                    break;

                    case 7:
                    if(currentFloor == 4){
                        removeSpecificPrio(i);
                        hissQ[3] = 0;
                    }
```

```c
                break;

                case 8:
                if(currentFloor == 5){
                    removeSpecificPrio(i);
                    hissQ[4] = 0;
                }
                break;
            }
        }
}
}

void hissUpdate(void){
    if(hissQ[0] == 1 && currentFloor > 1){
        addToPrio(1);
    }

    if(hissQ[1] == 1 && currentFloor > 2){
        addToPrio(2);
    }

    if(hissQ[1] == 1 && currentFloor < 2){
        addToPrio(3);
    }

    if(hissQ[2] == 1 && currentFloor > 3){
        addToPrio(4);
    }

    if(hissQ[2] == 1 && currentFloor < 3){
        addToPrio(5);
    }

    if(hissQ[3] == 1 && currentFloor > 4){
        addToPrio(6);
    }

    if(hissQ[3] == 1 && currentFloor < 4){
        addToPrio(7);
    }

    if(hissQ[4] == 1 && currentFloor < 5){
        addToPrio(8);
    }
}
```