

Källkod

```
/*
 * Fyra i rad.c
 *
 * Created: 2017-04-03 08:29:07
 * Author : inel4lra
 */

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

#include <util/delay.h>
#define F_CPU 8000000UL

int matrix[8][10];
int arrow;
int side; // Vilken chipselect: side = 0 --> CS1
int player;
char global;

/* Avbrottsrutin */

ISR(INT0_vect){
    _delay_ms(100);
    global = PINA & 0x0F;
}

/* Fylla matrix med nollor*/
void createMatrix(){
    for (int i = 0; i<8; i++){
        for (int j = 0; j<10; j++){
            matrix[i][j]=0;
        }
    }
}

/* Instruktioner till displayen*/

void executeInstruction(){
    _delay_ms(10);
    if(side == 0){
        PORTD = 0x21; //skriv instruction istället för data
        PORTC = 0x01;
        PORTD = 0x01; //Execute
    } else {
        PORTD = 0x22;
        PORTC = 0x01;
        PORTD = 0x02;
    }
}
}
```

```

/* Sätter adressen på displayen*/

void setXY(int x, int y){
    PORTB = 0xB8 + x; //Sätt x
    executeInstruction(); //sätt adress; därför instruction
    if(y<64){ //CS storlek, CS1
        PORTB = 0x40 + y;
        side = 0;
        executeInstruction();
    } else { //CS2
        y = y - 64;
        PORTB = 0x40 + y;
        side = 1;
        executeInstruction();
    }
}

/* Förbereder display för att skriva*/

void write(){
    _delay_ms(1);
    if(side==0){
        PORTD = 0x31; //Sätter porten till att skriva data, CS1
        Behövs reset port c
        PORTD = 0x11; //Nu skriver den
    } else {
        PORTD = 0x32;
        PORTD = 0x12;
    }
}

/* Skriver på displayen*/
void draw(int x, int y, char data){
    setXY(x,y);
    PORTB = data;
    write();
}

/* Kollar om en spelare har vunnit */

int victory(int x, int y){
    int count = 0;
    int stop1 = 0;
    int stop2 = 0;
    for(int i = 1; i<4; i++){
        if (matrix[x][y] == matrix[x+i][y] && stop1 == 0){ //kolla
            neråt.
                count++;
            }else{
                stop1 = 1;
            }
        }
        if(count == 3){ //om fyra i rad

```

```

        return 1;
    }
    count = 0;
    stop1 = 0;
    for(int i = 1; i<4; i++){
        if(matrix[x][y] == matrix[x][y+i] && stop1 ==
0){//Kollar åt höger
            count++;
        }else{
            stop1 =1;
        }
        if(matrix[x][y] == matrix[x][y-i] && stop2 ==
0){//Kollar åt vänster
            count++;
        }else{
            stop2 = 1;
        }
    }
    if(count == 3){//om fyra i rad
        return 1;
    }
    count =0;
    stop1=0;
    stop2=0;
    for(int i = 1;i<4;i++){
        if(matrix[x][y] == matrix[x+i][y+i] &&
stop1 == 0){//Kollar snett neråt höger
            count++;
        }else{
            stop1=1;
        }
        if(matrix[x][y] == matrix[x-i][y-i] &&
stop2 == 0){//Kollar snett uppåt vänster
            count++;
        }else{
            stop2=1;
        }
    }
    if(count==3){//om fyra i rad
        return 1;
    }
    count=0;
    stop1=0;
    stop2=0;
    for(int i = 1;i<4;i++){
        if(matrix[x][y] == matrix[x-i][y+i] && stop1 ==
0){//Kollar snett uppåt höger
            count++;
        }else{
            stop1=1;
        }
        if(matrix[x][y] == matrix[x+i][y-i] && stop2 ==
0){//Kollar snett neråt vänster
            count++;
        }else{
            stop2=1;
        }
    }

```

```

        }
        if(count==3){//om fyra i rad
            return 1;
        }
        return 0;
    }

/* Hittar plats på displayen*/

int findOurPlace(){
    int currentPlace = 0;
    if(matrix[currentPlace][arrow] !=0){//om kolonnen redan är
ifylld
        return -1;
    }
    while (matrix[currentPlace+1][arrow] == 0 && currentPlace
<7){//7 pga har kollat första platsen
        currentPlace++;
    }
    return currentPlace;
}
/* Tar bort strecken som visar vilken kolumn spelare befinner sig
i*/

void clearArrow(int y){
    y=8*y;
    for(int i = 0; i<8; i++){
        draw(i,y,0x00);
        draw(i,y+7,0x00);
    }
}

/*Rita strecken som visar vilken kolumn spelare befinner sig i*/

void drawArrow(int y) {
    if(y<10){
        y= 8*y;//gör om till dots
        for(int i =0; i<8;i++){
            draw(i,y,0xFF);
            draw(i,y+7,0xFF);
        }
    }
}

/*Ritar spelpjäs för spelare 1*/
void player1(int x, int y){
    y=y*8;
    draw(x,y+1,0x00);
    draw(x,y+2,0x3C);
    draw(x,y+3,0x3C);
    draw(x,y+4,0x3C);
    draw(x,y+5,0x3C);
    draw(x,y+6,0x00);
}
/*Ritar spelpjäs för spelare 2*/
void player2(int x, int y){
    y=y*8;

```

```

        draw(x,y+1,0x00);
        draw(x,y+2,0x3C);
        draw(x,y+3,0x24);
        draw(x,y+4,0x24);
        draw(x,y+5,0x3C);
        draw(x,y+6,0x00);
    }

    /* Tömmer displayen*/
    void clearDisplay(){
        for(int i = 0; i <128; i++){//??128
            for(int j = 0; j<8; j++){
                setXY(j,i);
                PORTB = 0x00;
                write(); // töm displayen; därför write
            }
        }
    }

    /*läser av knappar */
    void readButton(){
        if(global == 0x04){ //högerknapp
            if(arrow < 10){
                clearArrow(arrow);
                arrow++;
                drawArrow(arrow);
                if(player == 1){
                    PORTD = 0x80;
                }else{
                    PORTD = 0x40;
                }
            }
        }
        else if(global == 0x01){ //vänsterknapp
            if(arrow>0){
                clearArrow(arrow);
                arrow--;
                drawArrow(arrow);
                if(player == 1){
                    PORTD = 0x80;
                }else{
                    PORTD = 0x40;
                }
            }
        }
        else if(global == 0x02){ //put-knapp
            int x = findOurPlace();
            if(x!= -1){
                if(player == 1){
                    player1(x,arrow);
                    matrix[x][arrow] = 1;
                    if(victory(x,arrow) ==1){
                        PORTD =0x80;
                        _delay_ms(1000);
                        PORTD=0x00;
                        _delay_ms(1000);
                    }
                }
            }
        }
    }

```

```

PORTD =0x80;
_delay_ms(1000);
PORTD=0x00;
_delay_ms(1000);
PORTD =0x80;
_delay_ms(1000);
PORTD=0x00;
_delay_ms(1000);
PORTD=0x80;
_delay_ms(1000);
PORTD =0x00;
_delay_ms(1000);
PORTD=0x80;
_delay_ms(1000);
PORTD =0x00;
_delay_ms(1000);
PORTD=0x80;
_delay_ms(1000);
PORTD =0x00;
clearDisplay();
}else{
player =2;
PORTD = 0x40;
}
}else{
player2(x,arrow);
matrix[x][arrow] = 2;
if(victory(x,arrow) ==1){
PORTD =0x40;
_delay_ms(1000);
PORTD=0x00;
_delay_ms(1000);
PORTD =0x40;
_delay_ms(1000);
PORTD=0x00;
_delay_ms(1000);
PORTD =0x40;
_delay_ms(1000);
PORTD=0x00;
_delay_ms(1000);
PORTD=0x40;
_delay_ms(1000);
PORTD =0x00;
_delay_ms(1000);
PORTD=0x40;
_delay_ms(1000);
PORTD =0x00;
_delay_ms(1000);
PORTD=0x40;
_delay_ms(1000);
PORTD =0x00;
clearDisplay();
}else{
player = 1;
PORTD = 0x80;
}
}

```

```

        }
    }
    else if(global ==0x08){
        createMatrix();
        arrow = 0;
        side = 0;
        clearDisplay();
        drawArrow(0);
        player = 1;
        PORTD = 0x80;
    }
}

/* Sätter på displayen*/
void turnOnDisplay(){
    PORTB = 0x3F; //se sidan 10 i displayens datablad
    executeInstruction();
    side = 1;
    PORTB = 0x3F; //se sidan 10 i displayens datablad
    executeInstruction();
}

int main(void)
{
    DDRA = 0x00;
    DDRB = 0xFF;
    DDRD = 0xfb;
    global = 0x00;
    GICR = 0b01000000;
    MCUCR = 0b00000011;
    sei();
    createMatrix();
    arrow = 0;
    side =0;
    turnOnDisplay();
    clearDisplay();
    drawArrow(0);
    player = 1;
    PORTD = 0b10000000;

    while (1)
    {
        if(global != 0x00){
            readButton();
            global = 0x00;
        }
    }
}

```

