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

#define F_CPU 8000000L  // 8 MHz
    //#define F_CPU 14.7456E6
#include <util/delay.h>

    unsigned char num;
    unsigned short int button;
    unsigned short int blackPaddleUp = 0xBF;
    unsigned short int blackPaddleDown = 0xBE;
    unsigned short int redPaddleUp = 0xB9;
    unsigned short int redPaddleDown = 0xB8;
    unsigned short int redScore;
    unsigned short int blackScore;
    unsigned short int right = 0x40; //Y-koordinaten längst till höger på respektive skärm.
    unsigned short int left = 0x7f; //Y-koordinaten längst till vänster på respektive skärm.
    unsigned short int ballX = 0xB9; //Bollens position i X-led (0-7)
    unsigned short int ballY = 0x40; // Bollens högra sida i Y-led (0, 8, 16, 24, 32, 40, 48 ↙
    eller 56)
    unsigned short int screen = 1; //Alternerar mellan 1 (röd sida) & 2 (svart sida)
    unsigned short int dir = 0; //(1 = uppåt, 0 = rakt, -1 = neråt)
    unsigned short int gameStatus = 1; // 1 = Meny, 2 = spel pågår
    unsigned short int display = 1; //1: Displayen är på, 2 Displayen är av
    unsigned short int speed = 100; //Hastighet för bollförflyttning i ms.



void main(void)
 {
    initialize();
    playGame();
    menu();

    while(1)
    {
        PORTC = num;
        num++;
    }
    return;
}

void menu()
{
    drawScore();
    while(gameStatus == 1) { }
    playGame();

}

void playGame()
{
    speed = 100;
    gameStatus = 2;
    redraw();
    if(screen == 1)
    {
        moveLeft();
    } else
    {
        moveRight();
    }
    menu();
}

void initialize()
{
    //set PORT A for out
```

```
    DDRA = 0xff;
    DDRB = 0xff;
    DDRC = 0x00;
    DDRD = 0x83;
    GICR = 0xC0; //Aktiverar INT0 & INT1 enligt s.68
    MCUCR = 0x0f; // Sätter att INT0 & INT1 reagerar på en ökning enligt s.67.
    sei();
    num = 0;
    displayOn();
    redraw();
}

//Kör en "LCD-toggle"
void toggle()
{
    PORTA = PORTA | 0x80; //Or med "Toggle hög" (1 0 0 0 0 0 0 0) => E hög
    PORTA = PORTA & 0x7F; //And med inverterad "Toggle hög" (0 1 1 1 1 1 1 1) => E låg
    PORTA = PORTA | 0x80; //Or med "Toggle hög" (Se första steget) => E hög
}

void isReady()
{
    while(PORTC < 32)
    {
        PORTC += 1;
    }
    PORTC = 0;
}

//Sätter på skärmen.
void displayOn()
{
    PORTB = 0x3F;
    PORTA = 0x9C;
    toggle();
}

//Stänger av skärmen
void displayOff()
{
    PORTB = 0x3E;
    PORTA = 0x9C;
    toggle();
}

//Ritar upp allt på skärmen
void redraw()
{
    clearDisplay();
    drawBall();
    drawRedPaddle();
    drawBlackPaddle();
}


//Hamnar här vid en interrupt. Här vill vi kolla vilken knapp som är intryckt och utföra önskat↙
     kommando.
ISR(INT0_vect)
{
    _delay_ms(45);   //Undviker "dubbla knapptryckningar"
    if(display == 0)
    {
        displayOn();
        display = 1;
    }
    gameStatus = 2;
        switch(PIND)
        {
```

```
            case 0x44: //Knapp 1
            if (redPaddleUp != 0xBF) //Kollar om den redan är i toppen av skärmen.
            {
                redPaddleDown = redPaddleUp;
                redPaddleUp += 1;
                redraw();
            }
            break;

            case 0x24: //Knapp 2
            if (redPaddleDown != 0xB8)
            {
                redPaddleUp = redPaddleDown;
                redPaddleDown -= 1;
                redraw();
            }
            break;
        }
}

ISR(INT1_vect)
{

    _delay_ms(45);
    if(gameStatus == 1)
    {
        displayOff();
        display = 0;
        blackScore = 0;
        redScore = 0;
    }

    switch(PIND)
    {
        case 0x18: //Knapp 3
        if (blackPaddleUp != 0xBF)
            {
                blackPaddleDown = blackPaddleUp;
                blackPaddleUp += 1;
                redraw();
            }
            break;

        case 0x88: //Knapp 4
            if (blackPaddleDown != 0xB8)
            {
                blackPaddleUp = blackPaddleDown;
                blackPaddleDown -= 1;
                redraw();
            }
            break;
    }
}

//Funktion för att hänvisa interruppts från de svarta knapparna till funktionen ovan.
//ISR(INT1_vect, ISR_ALIASOF(INT0_vect));


void write(int x, int y, int data, int displayScreen)
{
    if(displayScreen == 1)
    {
        PORTA = 0x94;
    } else if (displayScreen == 2)
    {
        PORTA = 0x98;
    }
    PORTB = x;
```

```
    isReady();
    toggle();

    PORTB = y;
    isReady();
    toggle();

    if(displayScreen == 1)
    {
        PORTA = 0xD4;
    } else if (displayScreen == 2)
    {
        PORTA = 0xD8;
    }
    PORTB = data;
    isReady();
    toggle();

    PORTA = 0x90;
    isReady();
    toggle();
}



void moveLeft()
{
    changeDirection();

    while(ballY != 0x78) //Kollar om bollen har nått vänstersidan av skärmen
    {
    _delay_ms(speed); //Fördröjer i SPEED sekunder.
    moveOneStepLeft();
    }


    checkIfBlackLost();
    moveRight();
}


void moveOneStepLeft()
{
    if (ballX == 0xbf || ballX == 0xb8) //Kollar ifall den slår i "taket" eller "golvet".
    {
    dir *=-1;
    }

    ballX += dir;
    ballY += 8;

    redraw();

    if (screen == 1 && ballY == 0x78)
    {
    changeScreen();
    }
}

void moveRight()
{
    changeDirection();

    while(ballY != right)
    {
    _delay_ms(speed); //Fördröjer i SPEED sekunder.
    moveOneStepRight();
    }
```

```
    checkIfRedLost();
    moveLeft();
}

void moveOneStepRight()
{
    if (ballX == 0xbf || ballX == 0xb8) //Kollar ifall den slår i "taket" eller "golvet".
    {
    dir *=-1;
    }

    ballX += dir;
    ballY -= 8;

    redraw();

    if (screen == 2 && ballY == right)
    {
        changeScreen();
    }
}

void changeScreen()
{
    if (screen == 1)
    {
        screen = 2;
        ballY = right - 8; //- de 8 som kommer att plussas på i nästa steg.
    } else if (screen == 2)
    {
        screen = 1;
        ballY = left +1; //+ de 8 som kommer att dras bort i nästa steg.
    }
}

void checkIfBlackLost()
{
    if (ballX != blackPaddleUp && ballX != blackPaddleDown)
    {
        gameStatus = 1;
        redScore += 1;
        menu();
    }

}

void checkIfRedLost()
{
    if (ballX != redPaddleUp && ballX != redPaddleDown)
    {
        gameStatus = 1;
        blackScore += 1;
        menu();
    }

}


void changeDirection()
{
    unsigned short int number = random();

    if(speed >= 40)
    {
        speed -= 5;
    }
```

```
    switch(number)
    {
        case 0:
        dir = 1;
        break;

        case 1:
        dir = 0;
        break;

        case 2:
        dir = -1;
        break;
    }

    if ((dir == 1 && ballX == 0xb8) || (dir == -1 && ballX == 0xbf))
    {
        dir *= -1;
    }
}

int random()
{
    return (redPaddleUp + blackPaddleDown + redScore + screen) % 3;
}


void clearDisplay()
{

    for(int k = 0xB8; k <= (0xB8 + 0x07); k++)
    {
        PORTA = 0x9C; //Markera båda skärmhalvorna
        PORTB = k; //Markera den aktuella byten
        isReady();
        toggle();
        PORTB = 0x40; //Markera längst till höger
        isReady();
        toggle();
        PORTA = 0xDC; //Markera för utskrift
        PORTB = 0x00; //Rensa markerade pixlar
        for(int i = 0; i < 64; i++)
        {
        isReady();
        toggle();
        }
    }
}

void drawBlackPaddle()
{
    write(blackPaddleUp, left, 0xff, 2);
    write(blackPaddleDown, left, 0xff, 2);

}

void drawRedPaddle()
{
    write(redPaddleUp, right, 0xff, 1);
    write(redPaddleDown, right, 0xff, 1);
}

void drawBall()
{
    for (unsigned short int k = ballY; k < (ballY + 8) ; k +=1)
    {
        write(ballX, k, 0xff, screen);
```

```
    }
}


void drawScore()
{
    for (unsigned short int k = blackScore; k > 0; k -= 1)
    {
        write(0xBE, (0x40 + 10 + (2*k)), 0xff, 2);
    }

    for (unsigned short int k = redScore; k > 0; k -= 1)
    {
        write(0xBE, (0x40 + 45 + (2*k)), 0xff, 1);
    }
    if(blackScore >= 5 || redScore >= 5)
    {
        _delay_ms(10000);
        blackScore = 0;
        redScore = 0;
        gameStatus = 1;
        menu();
    }
}
```