1 Anexos

1.1 Main

```
#include "max7219.h"
#include "utils.h"
#include "drawing.h"
#include <msp430g2553.h>
#include <legacymsp430.h> // Para rodar interrupcoes

/*
                  MSP430G2553
               -----------------
              |VCC           GND|
RIGHT_BTN-->|P1.0           XIN|
              |P1.1         XOUT|
              |P1.2          TST|
       DIN<--|P1.3           RST|
        CS<--|P1.4          P1.7|<--STOP_BTN
       CLK<--|P1.5          P1.6|<--LEFT_BTN
              |P2.0          P2.5|
              |P2.1          P2.4|
              |P2.2          P2.3|
               -----------------
*/

int main(){
  WDTCTL = WDTPW + WDTHOLD;     // Desabilita WDT
  DCOCTL = CALDCO_1MHZ;        // 1 Mhz DCO
  BCSCTL1 = CALBC1_1MHZ;

  initialise();
  setTestMode(0);
  setShutdown(0);
  setBrightness(0xff);
  showDigits(8);

  configure_buttons();

  __enable_interrupt();

  clear_screen();

  while(1){
    ahead_arrow();
  }
  return 0;
}

#pragma vector=PORT1_VECTOR
__interrupt void Port_1(void){
  if(P1IFG & RIGHT_BTN){
    while(!(P1IN & RIGHT_BTN)==0){
      right_arrow();
    }
    P1IFG &= ~RIGHT_BTN;
  }
  if(P1IFG & LEFT_BTN){
    while(!(P1IN & LEFT_BTN)==0){
      left_arrow();
    }
    P1IFG &= ~LEFT_BTN;
  }
  if(P1IFG & STOP_BTN){
    while((P1IN & STOP_BTN)==0){
      delay(20);
      stop();
    }
    P1IFG &= ~STOP_BTN;
  }
  return 0;
}
```

1.2 MAX7219

```
#include "max7219.h"
#include "utils.h"
#include <msp430g2553.h>

#define MAX7219_DIN  BIT3
#define MAX7219_CS   BIT4
#define MAX7219_CLK  BIT5


static void MAX7219_SendByte (unsigned char dataout)
{
  char i;
  for (i=8; i>0; i--) {
    unsigned char mask = 1 << (i - 1);
    P1OUT &= ~(MAX7219_CLK);
    if (dataout & mask)
```

```c
      P1OUT |= MAX7219_DIN;
    else
      P1OUT &= ~(MAX7219_DIN);
    P1OUT |= MAX7219_CLK;
  }
}

void initialise(){
  P1OUT |= MAX7219_CS;

  P1DIR |= MAX7219_DIN;

  P1DIR |= MAX7219_CS;

  P1DIR |= MAX7219_CLK;

  output(0x0b, 7);
  output(0x09, 0x00);
}

void output(char address, char data){
  P1OUT |= MAX7219_CS;
  MAX7219_SendByte(address);
  MAX7219_SendByte(data);
  P1OUT &= ~(MAX7219_CS);
  P1OUT |= MAX7219_CS;
}

void setTestMode(int on){
  output(0x0f, on ? 0x01 : 0x00);
}

void setShutdown(int off){
  output(0x0c, off ? 0x00 : 0x01);
}

void showDigits(char numDigits){
  output(0x0b, numDigits-1);
}

void setBrightness(char brightness){
  output(0x0a, brightness);
}

void put_byte(char data) {
  char i = 8;
  char mask;
  while(i > 0) {
    mask = 0x01 << (i - 1);
    P1OUT &= ~(MAX7219_CLK);
    if (data & mask){
      P1OUT |= MAX7219_DIN;
    }else{
      P1OUT &= ~(MAX7219_DIN);
    }
    P1OUT |= MAX7219_CLK;
    --i;
  }
}

void max_single(char reg, char col) {
  P1OUT &= ~(MAX7219_CS);
  put_byte(reg);
  //asm("mov.w reg, R15");
  //asm("call #putByte");
  //asm("pop R15");
  put_byte(col);
  P1OUT &= ~(MAX7219_CS);
  P1OUT |= (MAX7219_CS);
}

void write8x8(char a, char b, char c, char d, char e, char f, char g, char h)
  max_single(1,a);
  max_single(2,b);
  max_single(3,c);
  max_single(4,d);
  max_single(5,e);
  max_single(6,f);
  max_single(7,g);
  max_single(8,h);
  delay(5000);
}
```

1.3 Drawning

```
#include "drawing.h"
#include "utils.h"
#include "max7219.h"

void stop(){
  write8x8(0x0,0x42,0x24,0x18,0x18,0x24,0x42,0x0);
  delay(10000);
  write8x8(0xff,0xbd,0xdb,0xe7,0xe7,0xdb,0xbd,0xff);
  delay(10000);
}

void left_arrow(){
  write8x8(0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0);
  write8x8(0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x18);
  write8x8(0x0,0x0,0x0,0x0,0x0,0x0,0x18,0x3c);
  write8x8(0x0,0x0,0x0,0x0,0x0,0x18,0x3c,0x7e);
  write8x8(0x0,0x0,0x0,0x0,0x18,0x3c,0x7e,0xff);
  write8x8(0x0,0x0,0x0,0x18,0x3c,0x7e,0xff,0x18);
  write8x8(0x0,0x0,0x18,0x3c,0x7e,0xff,0x18,0x18);
  write8x8(0x0,0x18,0x3c,0x7e,0xff,0x18,0x18,0x18);
  write8x8(0x18,0x3c,0x7e,0xff,0x18,0x18,0x18,0x18);
  write8x8(0x18,0x3c,0x7e,0xff,0x18,0x18,0x18,0x18);
  write8x8(0x3c,0x7e,0xff,0x18,0x18,0x18,0x18,0x0);
  write8x8(0x7e,0xff,0x18,0x18,0x18,0x18,0x0,0x0);
  write8x8(0xff,0x18,0x18,0x18,0x18,0x0,0x0,0x0);
  write8x8(0x18,0x18,0x18,0x18,0x0,0x0,0x0,0x0);
  write8x8(0x18,0x18,0x18,0x0,0x0,0x0,0x0,0x0);
  write8x8(0x18,0x18,0x0,0x0,0x0,0x0,0x0,0x0);
  write8x8(0x18,0x0,0x0,0x0,0x0,0x0,0x0,0x0);
}

void right_arrow(){
  write8x8(0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0);
  write8x8(0x18,0x0,0x0,0x0,0x0,0x0,0x0,0x0);
  write8x8(0x3c,0x18,0x0,0x0,0x0,0x0,0x0,0x0);
  write8x8(0x7e,0x3c,0x18,0x0,0x0,0x0,0x0,0x0);
  write8x8(0xff,0x7e,0x3c,0x18,0x0,0x0,0x0,0x0);
  write8x8(0x18,0xff,0x7e,0x3c,0x18,0x0,0x0,0x0);
  write8x8(0x18,0x18,0xff,0x7e,0x3c,0x18,0x0,0x0);
  write8x8(0x18,0x18,0x18,0xff,0x7e,0x3c,0x18,0x0);
  write8x8(0x18,0x18,0x18,0x18,0xff,0x7e,0x3c,0x18);
  write8x8(0x18,0x18,0x18,0x18,0xff,0x7e,0x3c,0x18);
  write8x8(0x0,0x18,0x18,0x18,0x18,0xff,0x7e,0x3c);
  write8x8(0x0,0x0,0x18,0x18,0x18,0x18,0xff,0x7e);
  write8x8(0x0,0x0,0x0,0x18,0x18,0x18,0x18,0xff);
  write8x8(0x0,0x0,0x0,0x0,0x18,0x18,0x18,0x18);
  write8x8(0x0,0x0,0x0,0x0,0x0,0x18,0x18,0x18);
  write8x8(0x0,0x0,0x0,0x0,0x0,0x0,0x18,0x18);
  write8x8(0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x18);
}

void ahead_arrow(){
  write8x8(0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0);
  write8x8(0x0,0x0,0x0,0x80,0x80,0x0,0x0,0x0);
  write8x8(0x0,0x0,0x80,0xc0,0xc0,0x80,0x0,0x0);
  write8x8(0x0,0x80,0xc0,0xe0,0xe0,0xc0,0x80,0x0);
  write8x8(0x80,0xc0,0xe0,0xf0,0xf0,0xe0,0xc0,0x80);
  write8x8(0x40,0x60,0x70,0xf8,0xf8,0x70,0x60,0x40);
  write8x8(0x20,0x30,0x38,0xfc,0xfc,0x38,0x30,0x20);
  write8x8(0x10,0x18,0x1c,0xfe,0xfe,0x1c,0x18,0x10);
  write8x8(0x8,0xc,0xe,0xff,0xff,0xe,0xc,0x8);
  write8x8(0x8,0xc,0xe,0xff,0xff,0xe,0xc,0x8);
  write8x8(0x4,0x6,0x7,0x7f,0x7f,0x7,0x6,0x4);
  write8x8(0x2,0x3,0x3,0x3f,0x3f,0x3,0x3,0x2);
  write8x8(0x1,0x1,0x1,0x1f,0x1f,0x1,0x1,0x1);
  write8x8(0x0,0x0,0x0,0xf,0xf,0x0,0x0,0x0);
  write8x8(0x0,0x0,0x0,0x7,0x7,0x0,0x0,0x0);
  write8x8(0x0,0x0,0x0,0x3,0x3,0x0,0x0,0x0);
  write8x8(0x0,0x0,0x0,0x1,0x1,0x0,0x0,0x0);
}
```

1.4 Utils

```
#include "utils.h"
#include "max7219.h"
#include <msp430g2553.h>

void delay(volatile unsigned int i){
  while((i--)>0);
}

void clear_screen(){
  write8x8(0x0,0x0,0x0,0x0,0x0,0x0,0x0,0x0); // Cleaning screen
}

void configure_buttons(){
  P1DIR &= ~(RIGHT_BTN + LEFT_BTN + STOP_BTN); //Seta como entrada 0 = entrad
  P1OUT &= ~(RIGHT_BTN + LEFT_BTN + STOP_BTN); //Desliga ambos os leds
  P1IE |= (RIGHT_BTN + LEFT_BTN + STOP_BTN);
  P1IFG &= ~(RIGHT_BTN + LEFT_BTN + STOP_BTN);
  P1REN = (RIGHT_BTN + LEFT_BTN + STOP_BTN);
  P1IES |= (RIGHT_BTN + LEFT_BTN + STOP_BTN);
}
```