STM32 mikrovezérlők programozása STM32CubeIDE környezetben

9. Az I2C kommunikációs csatorna – 2. rész

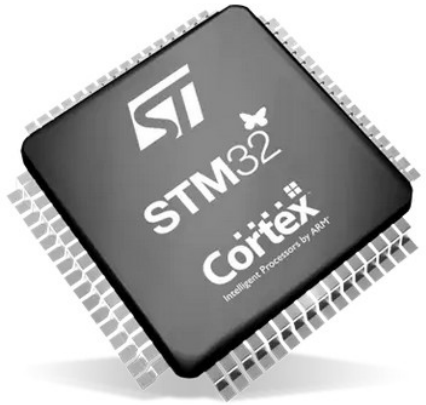
Felhasznált és ajánlott irodalom

- Carmine Noviello: **Mastering STM32**

  A könyv mintapéldái: **github.com/cnoviello/mastering-stm32**
- ControllersTech: **OLED display using I2C with STM32**
- Alexander Lutsai: **Блиблиотека OLED дисплея SSD1306 для STM32...**

Adatlapok, kézikönyvek:

- **STM32F103C8 adatlap és termékinfo**
- **STM32F103 Family Reference Manual**
- **STM32F446RE adatlap és termékinfo**
- **STM32F446 Family Reference Manual**
- Solomon Sytech: **SSD1306 adatlap**
- Sino Wealth: **SH1106 adatlap**
- Maxim Integrated: **DS3231 RTC adatlap**
- Bosch Sensortec: **BME280 adatlap**
- STmicro: **Description of STM32F4 HAL and low-layer drivers**



OLED I2C kijelző SSD1306 vezérlővel

Jellemzők:

- OLED technológia
- 128x64 képpont
- 0,96” (2,4 cm) képátló
- I2C illesztő
- 2.5V-5.5V tápfeszültség
- SSD1306 vezérlő
- Monokróm (egyes változatoknál a felső harmad más színű)
- Grafikus megjelenítés
- Inverz mód
- Görgetés több irányba
- Dokumentáció: Solomon Systech SSD1306 adatlap

OLED I2C kijelző SSD1306 vezérlővel

Jellemzők:

- **OLED** technológia
- **128x32** képpont
- 0,91” (2,3 cm) képátló
- I2C illesztő
- 2.5V-5.5V tápfeszültség
- **SSD1306** vezérlő
- Monokróm
- Grafikus megjelenítés
- Inverz mód
- Görgetés több irányba
- Dokumentáció: Solomon Systech SSD1306 adatlap

Kompatibilis az SSD1306 vezérlőjű 0.96” képátlójú, 128x64 felbontású kijelzővel, de az inicializálásban van néhány különbség

OLED I2C kijelző SH1106 vezérlővel

Jellemzők:

- OLED technológia
- 128x64 képpont (az IC 132x64-at tudna!)
- 1.3” (3,3 cm) képátló
- I2C illesztő
- 3,3V-5.0V tápfeszültség
- **SH1106** vezérlő
- Monokróm (egyes változatoknál a felső harmad más színű)
- Grafikus megjelenítés
- Inverz mód
- **Eltérések SSD1306-tól:** nincs görgetés, és 2 pixellel el van tolva a kép
- Dokumentáció: Sino Wealth SH1106 datasheet

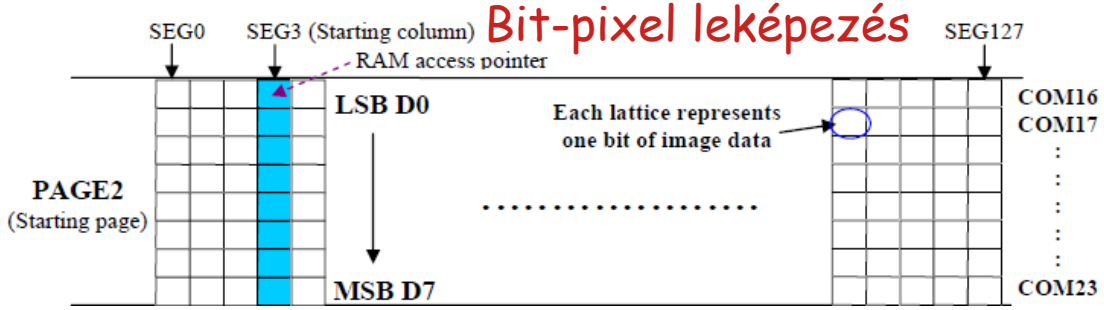
Az SSD1306 programkönyvtár

- Az általunk használt programkönyvtár eredete ide nyúlik vissza: Tilen Majerle: Library 61- SSD1306 OLED I2C LCD for STM32F4xx
- A fenti könyvtárat **STM32F103C8** mikrovezérlőre átdolgozta Alexander Lutsai: Блиблиотека OLED дисплея SSD1306 для STM32 микроконтроллеров
- Alexander Lutsai programkönyvtárát átdolgozták az **STM32 CubeMx**-hez és két demó programmal együtt közzétették a **Controllerstech.com** honlapon: Oled display using I2C and STM32
- A **Conrollerstech** honlapjáról letölthető programkönyvtárat 2019/2020-ban adaptáltuk a **Keil MDK5 Lite** környezethez és az **SH1106** vezérlőhöz is készítettünk egy módosított változatot
- Most egy újabb átdolgozás során közös forrásfájlba egyesítettük az **SSD1306** 128x64 és 128x32, illetve az **SH1106** 128x64 kezelését

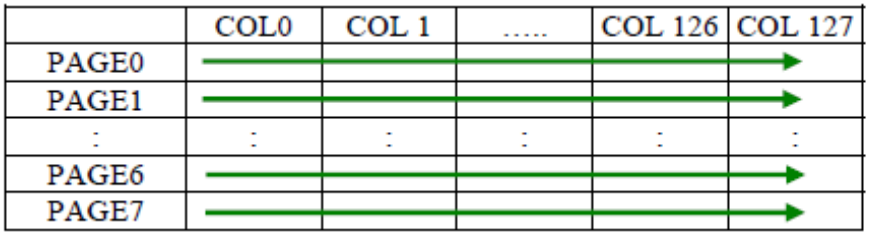
SSD1306/SH1106 parancsok

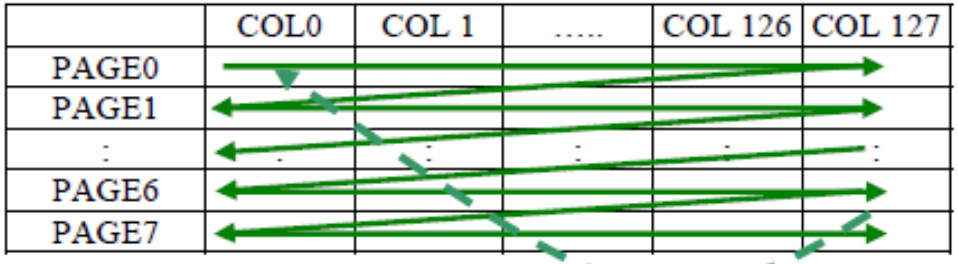
- Set Lower Column Start Address for Page Addressing Mode (00h~0Fh) az első oszlop címének alsó 4 bitje (ssd1306: 0000b, sh1106: 0010b)
- Set Higher Column Start Address for Page Addressing Mode (10h~1Fh) az első oszlop címének felső 4 bitje (általában 10h)
- Set Memory Addressing Mode (20h + adat[1:0]) – beállítja a címzésmódot 00: Horizontal, 01: Vertical, **10: Page,** 11: Invalid adressing mode (sh1106: csak a page addressing módot ismeri)

Bit-pixel leképezés

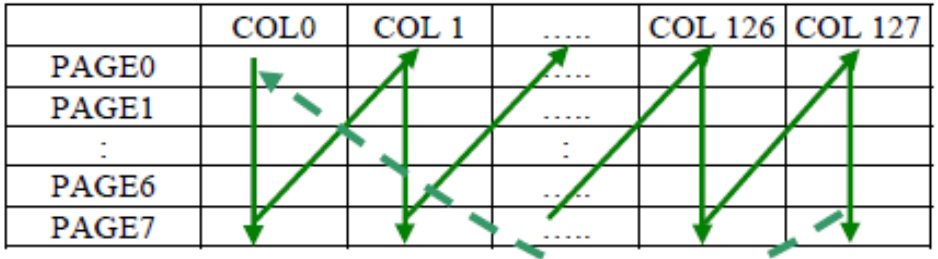


Page addressing





Horizontal addressing



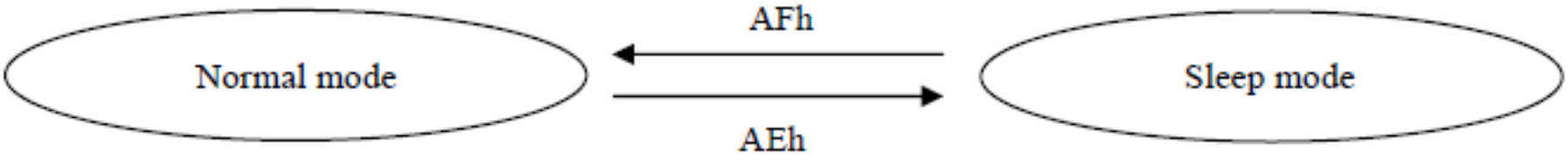
Vertical addressing

SSD1306/SH1106 parancsok

- Set Column Address (21h + start + end) – kezdő és végző oszlopcím megadása Horizontal címzésmódhoz (csak SSD1306 estén)
- Set Page Address (22h + start + end) – kezdő és végző lapcím megadása Horizontal/Vertical címzésmódhoz (csak SSD1306 estén)
- Set Pump Voltage value (30H~33H) – a Vpp értékét állítja be (30: 6.4V, 31: 7.4V, **32: 8.0V** (POR), 33: 9.0V) (csak SH1106 esetén!)
- Set Display Start Line (40h~7Fh) – a kezdősor megadása (POR=40h)
- Set Contrast Control (81h + data) – kontraszt beállítás
- Set Segment Re-map (A0h/A1h) – a leképezést állítja be (POR=A0h)
  A0: col. addr 0 ⟶ segment 127, **A1:** col. addr 127 ⟶ segment 0
- Entire Display ON (A4h/A5h) – az egész képernyő bekapcsolása
  A4: normál módba, **A5:** minden képpont kigyújtása
- Set Normal/Inverse Display (A6h/A7h) – normál/inverz mód

SSD1306/SH1106 parancsok

- Set Multiplex Ratio (A8h + 0Fh~3Fh) – multiplexelt sorok számának megadása, com0 - com63 közül ennyit kapcsolgatunk a COM jelre
- Set DC-DC ON/OFF (ADh + 8Ah/8Bh) – a DC-DC konverter ki/be kapcsolása **8A:** ki, **8B:** be (POR) (csak SH1106 esetén)
- Set DC-DC ON/OFF (8Dh + 10h/14h) – a DC-DC konverter ki/be kapcsolása **10:** ki, 14: be (POR) (csak SSD1306 esetén)
- Set Display ON/OFF (AEh/AFh) – a kijelző ki és bekapcsolása Konfigurálás elején is ki kell kapcsolni!



DC-DC STATUS	DISPLAY ON/OFF STATUS	Description
0	0	Sleep mode
0	1	External V_{PP} must be used.
1	0	Sleep mode
1	1	Built-in DC-DC is used, Normal Display

SSD1306/SH1106 parancsok

- Set Page Address for Page Addressing Mode (B0h~B7h) – a lap címét adja meg (B0~B7: 64 soros, B0~B3: 32 soros kijelzőhöz)
- Set COM Output Scan Direction (C0h/C8h) – a sorok sorrendjét lehet megfordítani **C0:** com0 ⟶ comN, **C8:** comN ⟶ com0
  Képernyő forgatás: A0+C0 normál, illetve A1+C8 „fejre állított”
- Set Display Offset (D3h + data) – sorok függőleges eltolása (normálisan 0 eltolást használunk)
- Set Display Clock Divide Ratio/ Oscillator Frequency (D5h + data) oszcillátor frekvencia bit[7:4] és leosztás bit[3:0] megadása
  (a POR érték SSD1306: 80h, SH1106: 50h)
- Set Dis-charge/Pre-charge Period (D9h data) – a képpont kioltás/felkapcsolás ideje DCLK egységben (POR data = 22h)

SSD1306/SH1106 parancsok

- Set COM Pads Hardware Configuration (DAh + 02h/12h) – a com sorvezérlők aktiválási sorrendje (02: szekvenciális, 12: alternáló) (64 soros kijelzőnél 12h, 32 soros kijelzőnél 02h értéket használunk) Megjegyzés: SSD1306 esetén az 5. bit is használható: COM Left/Right remap enable, így 22h és 32h paraméter is használható, ha a hardver ezt igényli
- Set V_{COMH} Deselect Level (DBh + data) – a V_{COMH} feszültséget állítja be (POR érték SSD1306: 20h, 0.77*Vcc, SH1106: 35h, 0.77*Vref)

SH1106 inicializálás

```
SSD1306_WRITECOMMAND(0xAE); //--display off
SSD1306_WRITECOMMAND(0xA0); //--set segment re-map 0 to 127
SSD1306_WRITECOMMAND(0xC0); //--Set COM Out Scan Dir 0 to 63
SSD1306_WRITECOMMAND(0x32); //--set pump voltage value to 8.0V (SH1106 only)
SSD1306_WRITECOMMAND(0x40); //--set start line address
SSD1306_WRITECOMMAND(0x81); //--set contrast control register
SSD1306_WRITECOMMAND(0x80); //  POR value = 80
SSD1306_WRITECOMMAND(0xA4); //--set normal mode (A5: test mode)
SSD1306_WRITECOMMAND(0xA6); //--set normal display (i.e non-inverted mode)
SSD1306_WRITECOMMAND(0xA8); //--set multiplex ratio(1 to 64)
SSD1306_WRITECOMMAND(0x3F); //
SSD1306_WRITECOMMAND(0xAD); //--set DC-DC mode
SSD1306_WRITECOMMAND(0x8B); //  DC-DC converter ON
SSD1306_WRITECOMMAND(0xD3); //--set display offset
SSD1306_WRITECOMMAND(0x00); //  no offset
SSD1306_WRITECOMMAND(0xD5); //--set display clock divide ratio/oscillator frequency
SSD1306_WRITECOMMAND(0x50); //  set frequency and divide ratio
SSD1306_WRITECOMMAND(0xD9); //--set dis-charge/pre-charge period
SSD1306_WRITECOMMAND(0x22); //
SSD1306_WRITECOMMAND(0xDA); //--set com pins hardware configuration
SSD1306_WRITECOMMAND(0x12);
SSD1306_WRITECOMMAND(0xDB); //--set vcomh
SSD1306_WRITECOMMAND(0x35); //  SH1106: 0x35, SSD1306: 0x20 0.77xVcc
SSD1306_WRITECOMMAND(0xAF); //--turn on SSD1306 panel
```

Képfordítás:
A0/C0 álló kép
A1/C8 fejreállított kép

SSD1306 inicializálás

```
SSD1306_WRITECOMMAND(0xAE); //display off
SSD1306_WRITECOMMAND(0x20); //Set Memory Addressing Mode
SSD1306_WRITECOMMAND(0x10); //00:Horizontal, 01:Vertical,10:Page, 11:Invalid
SSD1306_WRITECOMMAND(0xB0); //Set Page Start Address for Page Addressing Mode,0-7
SSD1306_WRITECOMMAND(0x00); //---set low column address
SSD1306_WRITECOMMAND(0x10); //---set high column address
SSD1306_WRITECOMMAND(0x40); //--set start line address
SSD1306_WRITECOMMAND(0x81); //--set contrast control register
SSD1306_WRITECOMMAND(0x8F); //*** 8f: for 32 line ff: 64 line
SSD1306_WRITECOMMAND(0xA1); //--set segment re-map 127 to 0
SSD1306_WRITECOMMAND(0xC8); //Set COM Out Scan Dir 63 to 0
SSD1306_WRITECOMMAND(0xA6); //--set normal display
SSD1306_WRITECOMMAND(0xA8); //--set multiplex ratio(1 to 64)
SSD1306_WRITECOMMAND(SSD1306_HEIGHT-1); //*** 31 or 63
SSD1306_WRITECOMMAND(0xA4); //0xa4,Output follows RAM;0xa5,Output ignores RAM content
SSD1306_WRITECOMMAND(0xD3); //-set display offset
SSD1306_WRITECOMMAND(0x00); //-not offset
SSD1306_WRITECOMMAND(0xD5); //--set display clock divide ratio/oscillator frequency
SSD1306_WRITECOMMAND(0x80); //--set recommended divide ratio
SSD1306_WRITECOMMAND(0xD9); //--set pre-charge period
SSD1306_WRITECOMMAND(0xF1); //-- Nem lényeges változtatás ...
SSD1306_WRITECOMMAND(0xDA); //--set com pins hardware configuration
SSD1306_WRITECOMMAND(0x02); //*** 02: for 32, 12:for 64 line
SSD1306_WRITECOMMAND(0xDB); //--set vcomh
SSD1306_WRITECOMMAND(0x20); //0x20,0.77xVcc
SSD1306_WRITECOMMAND(0x8D); //--set DC-DC enable
SSD1306_WRITECOMMAND(0x14); //
```

Képfordítás:
A0/C0 álló kép
A1/C8 fejreállított kép

Részletek az
SSD1306_Init() függvényből

Az egyesített inicializálás 3/2.

```
uint8_t SSD1306_Init(OledHandleTypedef *mydev) {
    dev = mydev;

    SSD1306_WRITECOMMAND(0xAE);      //display off
    if(dev->type == SSD1306_I2C) {
        SSD1306_WRITECOMMAND(0x20); //Set Memory Addressing Mode
        SSD1306_WRITECOMMAND(0x10); //00:Horizontal, 01:Vertical,10:Page, 11:Invalid
    }                               // Note: SH1106 has only Page mode
    SSD1306_WRITECOMMAND(0xB0);      //Set Page Address for Page Addressing Mode,0-7
    if(dev->type == SH1106_I2C) {
        SSD1306_WRITECOMMAND(0x02); //---set low column address    2: for SH1106 ***
    } else {
        SSD1306_WRITECOMMAND(0x00); //---set low column address    0: for SSD1306 ***
    }
    SSD1306_WRITECOMMAND(0x10);      //---set high column address
    if(dev->orientation == 0) {      //--- This is the "normal" direction
        SSD1306_WRITECOMMAND(0xA0); //--set segment re-map 0 to 127
        SSD1306_WRITECOMMAND(0xC0); //Set COM Output Scan Direction 0 to 63
    } else {                         //--- This is the "upside down" direction
        SSD1306_WRITECOMMAND(0xA1); //--set segment re-map 127 to 0
        SSD1306_WRITECOMMAND(0xC8); //Set COM Output Scan Direction 63 to 0
    }
    if(dev->type == SH1106_I2C) {
        SSD1306_WRITECOMMAND(0x32); //--set pump voltage value to 8.0V (SH1106 only)
    }
    SSD1306_WRITECOMMAND(0x40);      //--set start line address
```

Az egyesített inicializálás 3/2.

```
    SSD1306_WRITECOMMAND(0x81);         //--set contrast control register
    if(dev->type == SH1106_I2C) {
        SSD1306_WRITECOMMAND(0x80);     //  POR value = 80 for SH1106
    } else if(SSD1306_HEIGHT == 32) {
        SSD1306_WRITECOMMAND(0x8F);     // 0x8F for 32 line SSD1306
    } else {
        SSD1306_WRITECOMMAND(0xFF);     // 0xFF for 64 line SSD1306
    }
    SSD1306_WRITECOMMAND(0xA4);         // 0xa4,normal output 0xa5,highlight all pixels
    SSD1306_WRITECOMMAND(0xA6);         //--set normal display
    SSD1306_WRITECOMMAND(0xA8);         //--set multiplex ratio(1 to 64)
    SSD1306_WRITECOMMAND(SSD1306_HEIGHT-1); //*** 31 or 63
    SSD1306_WRITECOMMAND(0xD3);         //--set display offset
    SSD1306_WRITECOMMAND(0x00);
    if(dev->type == SH1106_I2C) {       // SH1106 with 64 line display
        SSD1306_WRITECOMMAND(0xD5);     //--set display clock divide ratio/osc. frequency
        SSD1306_WRITECOMMAND(0x50);     //  set frequency and divide ratio
        SSD1306_WRITECOMMAND(0xD9);     //--set dis-charge/pre-charge period
        SSD1306_WRITECOMMAND(0x22);     //
    } else if(SSD1306_HEIGHT == 32) { // SSD1306 with 32 line display
        SSD1306_WRITECOMMAND(0xD5);     //--set display clock divide ratio/osc. frequency
        SSD1306_WRITECOMMAND(0x80);     //  set recommended divide ratio
        SSD1306_WRITECOMMAND(0xD9);     //--set pre-charge period
        SSD1306_WRITECOMMAND(0xF1);     //
    } else {                            // SSD1306 with 64 line display
```

Az egyesített inicializálás 3/3.

```
        SSD1306_WRITECOMMAND(0xD5);   //--set display clock divide ratio/osc. frequency
        SSD1306_WRITECOMMAND(0xF0);   //  set divide ratio
        SSD1306_WRITECOMMAND(0xD9);   //--set pre-charge period
        SSD1306_WRITECOMMAND(0x22);   //
    }
    SSD1306_WRITECOMMAND(0xDA);       //--set com pins hardware configuration
    if(SSD1306_HEIGHT == 64) {
        SSD1306_WRITECOMMAND(0x12);   //*** 12:for 64 line
    } else {
        SSD1306_WRITECOMMAND(0x02);   //*** 02: for 32 line
    }
    SSD1306_WRITECOMMAND(0xDB);       //--set vcomh
    if(dev->type == SH1106_I2C) {
        SSD1306_WRITECOMMAND(0x35);   //  SH1106: 0x35 0.77xVcc
        SSD1306_WRITECOMMAND(0xAD);   //--set DC-DC mode
        SSD1306_WRITECOMMAND(0x8B);   //  DC-DC converter ON
    } else {
        SSD1306_WRITECOMMAND(0x20);   //  SSD1306: 0x20 0.77xVcc
        SSD1306_WRITECOMMAND(0x8D);   //--Charge Pump Setting
        SSD1306_WRITECOMMAND(0x14);   //  Enable Charge Pump
        SSD1306_WRITECOMMAND(0x2E);   //--Stop scrolling
    }
    SSD1306_WRITECOMMAND(0xAF);       //--turn on SH1106/SSD1306 panel
    SSD1306_Fill(SSD1306_COLOR_BLACK);//-- Clear screen
    return 1;
}
```

Az SSD1306 I2C programkönyvtár használata

- A projekthez az alábbi forrásfájlokat kell hozzáadni:
  - **ssd1306.h** – az SSD1306 programkönyvtár fejléc állománya
  - **ssd1306.c** – az SSD1306 programkönyvtár (az I2C kezeléssel)
  - **fonts.h** – a fontleíró adattömbök típusdefiníciós fejléc állománya
  - **fonts.c** – a fontleíró adattömbök

```
#include "main.h"
#include "ssd1306.h"
I2C_HandleTypeDef hi2c1;
OledHandleTypedef oled = {
  .type = SSD1306_I2C,          // SH1106 or SSD1306
  .orientation = 0,             // 1: downward 0: upward
  .i2c = &hi2c1,                // I2C1 at RB9/RB8
  .i2c_address = 0x78           // 0x78 (0x3c<<1) or 0x7A (0x3D<<1)
};

int main(void) {
  HAL_Init(); SystemClock_Config();
  MX_GPIO_Init(); MX_I2C1_Init();
  SSD1306_Init(&oled);          // Inicializálás
  SD1306_GotoXY(5, 5);
  SD1306_Puts("HELLO", &Font_11x18, SSD1306_COLOR_WHITE);
  SSD1306_UpdateScreen();       // Megjelenítés
  while(1) { }
}
```

ssd1306.h

```
#include "stm32f4xx.h"
#include "fonts.h"
#ifndef SSD1306_WIDTH
  #define SSD1306_WIDTH           128   /* SSD1306 width in pixels */
#endif
#ifndef SSD1306_HEIGHT
  #define SSD1306_HEIGHT           64   /* SSD1306 LCD height in pixels */
#endif
typedef enum {
  SSD1306_I2C = 1,
  SH1106_I2C  = 2,
} oled_type;
typedef struct {
  oled_type type;
  uint16_t orientation;
  uint16_t i2c_address;
  I2C_HandleTypeDef* i2c;
  uint8_t* buffer;
} OledHandleTypedef;
typedef enum {
    SSD1306_COLOR_BLACK = 0x00,       /* Black color, no pixel */
    SSD1306_COLOR_WHITE = 0x01        /* Pixel is set. Color depends on LCD */
} SSD1306_COLOR_t;

uint8_t SSD1306_Init(OledHandleTypedef *dev);
void SSD1306_UpdateScreen(void);
. . .
```

Függvények az SSD1306 kijező használatához

- SSD1306_Init(OledHandleTypedef *dev) - a képernyő inicializálása
- SSD1306_UpdateScreen() - képkirajzolás
- SSD1306_ToggleInvert(void) – invertálás ki/bekapcsolása
- SSD1306_Fill(Color) – adott színnel tölti ki a képet
- SSD1306_DrawPixel(x, y, color) – egy pixel kirajzolása
- SSD1306_GotoXY(x, y) – kurzor pozicionálás
- SSD1306_Putc(ch, Font, color) – egy karakter kirajzolása
- SSD1306_Puts(str, Font, color) – egy string kiírása
- SSD1306_DrawLine(x0, y0, x1, y1, color) – egyenesszakasz kirajzolása
- SSD1306_DrawRectangle(x, y, w, h, color) – téglalap rajzolása
- SSD1306_DrawFilledRectangle(x,y,w,h,color) – kitöltött téglalap rajzolása
- SSD1306_DrawTriangle(x1,y1,x2,y2,x3,y3,color) – háromszög rajzolása
- SSD1306_DrawCircle(x0, y0, r, color) – kör rajzolása
- SSD1306_DrawFilledCircle(x0, y0, r, color) – kitöltött kör rajzolása
- SSD1306_DrawBitmap(x0, y0, bitmap, w, h, color) – bitkép kirajzolása

A HAL_I2C fontosabb függvényei

- HAL_StatusTypeDef HAL_I2C_IsDeviceReady(I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint32_t Trials, uint32_t Timeout) - ellenőrzi, hogy a megcímzett eszköz válaszol-e (pl. a flash memória befejezte-e az írás műveletet…)
- HAL_StatusTypeDef HAL_I2C_Master_Transmit(I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint8_t * pData, uint16_t Size, uint32_t Timeout) - master módú írás a megcímzett slave eszközbe
- HAL_StatusTypeDef HAL_I2C_Master_Receive(I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint8_t * pData, uint16_t Size, uint32_t Timeout) - master módú olvasás a megcímzett eszközből
- HAL_StatusTypeDef HAL_I2C_Mem_Write(I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint16_t MemAddress, uint16_t MemAddSize, uint8_t * pData, uint16_t Size, uint32_t Timeout) - master módú írás: az adatok kiküldése előtt egy címet is kiküldve
- HAL_StatusTypeDef HAL_I2C_Mem_Read(I2C_HandleTypeDef * hi2c, uint16_t DevAddress, uint16_t MemAddress, uint16_t MemAddSize, uint8_t * pData, uint16_t Size, uint32_t Timeout) - master módú olvasás: az adatok olvasása előtt egy címet is kiküldve, majd repeat start feltétel után az olvasással folytatva

ssd1306.c (részlet)

- A `HAL_I2C_Mem_Write()` függvényt arra használjuk, hogy a külön változókban kapott címet és adatot egyetlen tranzakcióban kiküldhessük az I2C buszon

```
////////////////////////////////////////////////////////////////////////////////
//  _____ ___   _____
// |_   _|__ \ / ____|
//   | |    ) | |
//   | |   / /| |
//  _| |_ / /_| |____
// |_____|____|\_____|
//
////////////////////////////////////////////////////////////////////////////////

void ssd1306_I2C_WriteMulti(uint8_t reg, uint8_t* data, uint16_t count) {
  HAL_I2C_Mem_Write(dev->i2c, dev->i2c_address, (uint16_t)reg, 1, data, count, 20);
}

void ssd1306_I2C_Write(uint8_t reg, uint8_t data) {
    uint8_t dt[2];
    dt[0] = reg;
    dt[1] = data;
    HAL_I2C_Master_Transmit(dev->i2c, dev->i2c_address, dt, 2, 20);
}
```

Példaprogramok

F446RE_oled_I2C – SSD1306/SH1106 OLED kijelző használata

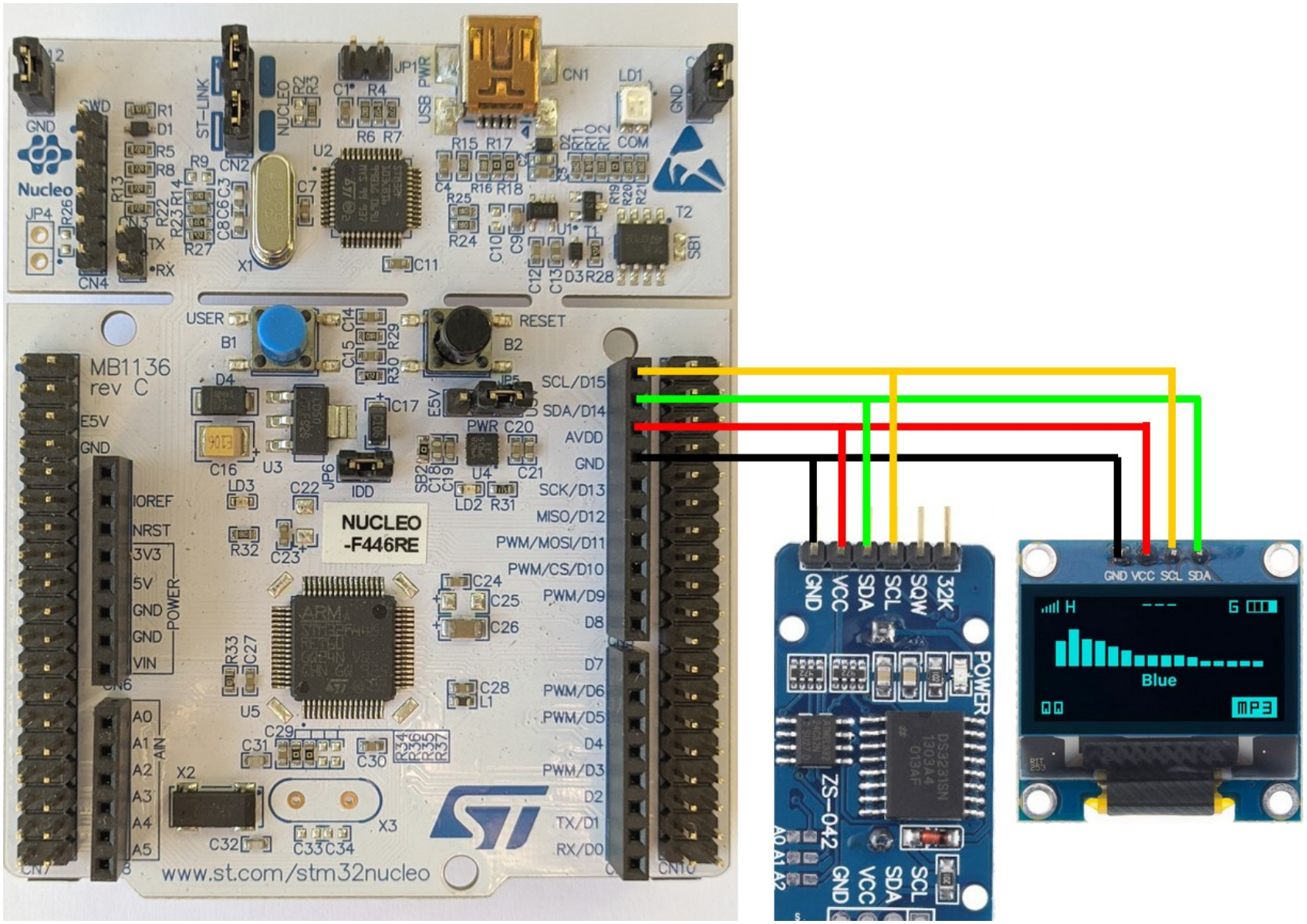
F446RE_horse_anim – bitmap animáció (vágtató ló)

F446RE_clown_anim– saját animáció készítése

F446RE_oled_RTC – óra 128x32 OLED kijelzővel

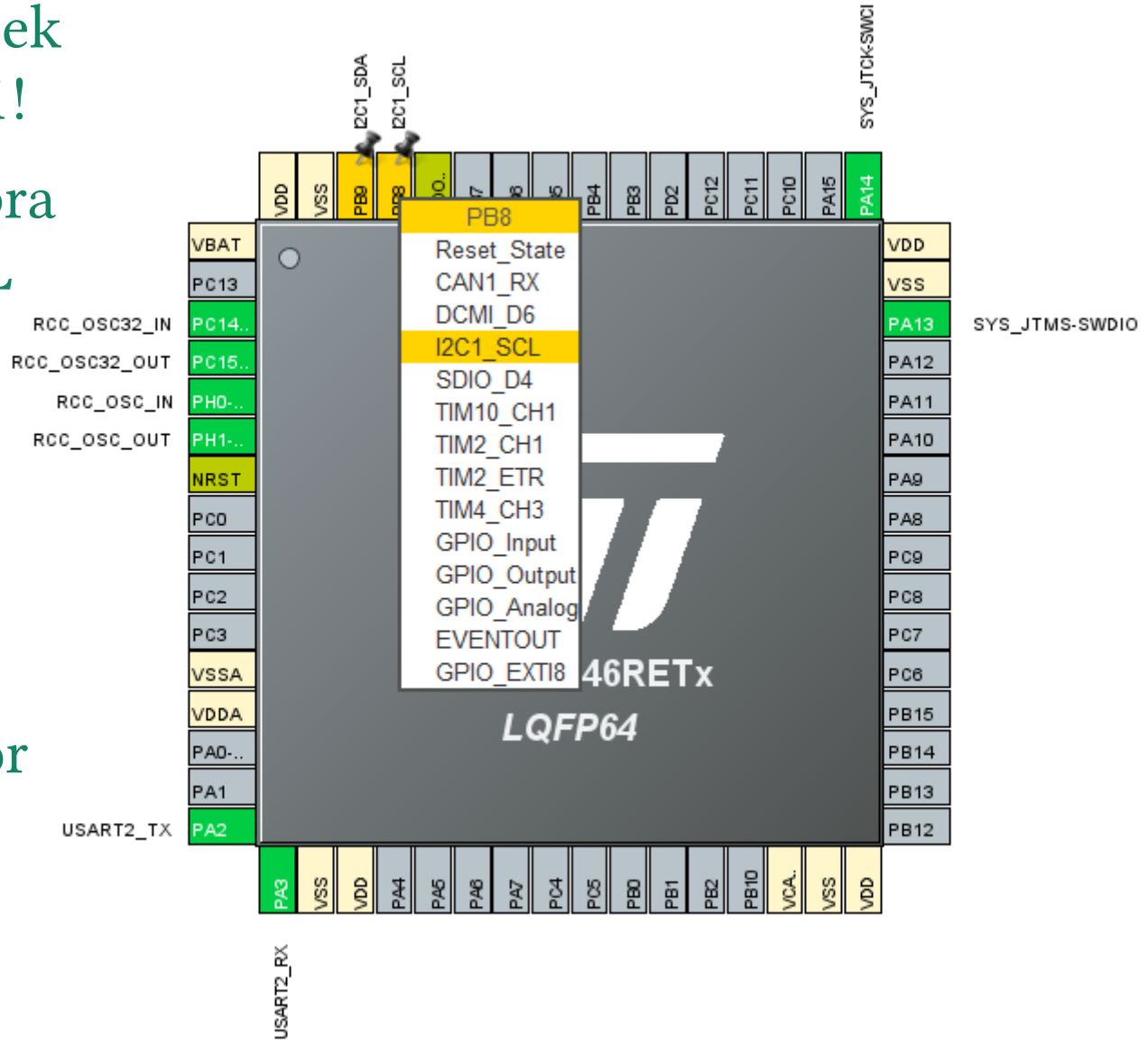
Nucleo-F446RE – a kapcsolási elrendezés

- Az alábbi kapcsolás mindegyik mintaprogramunkhoz megfelel



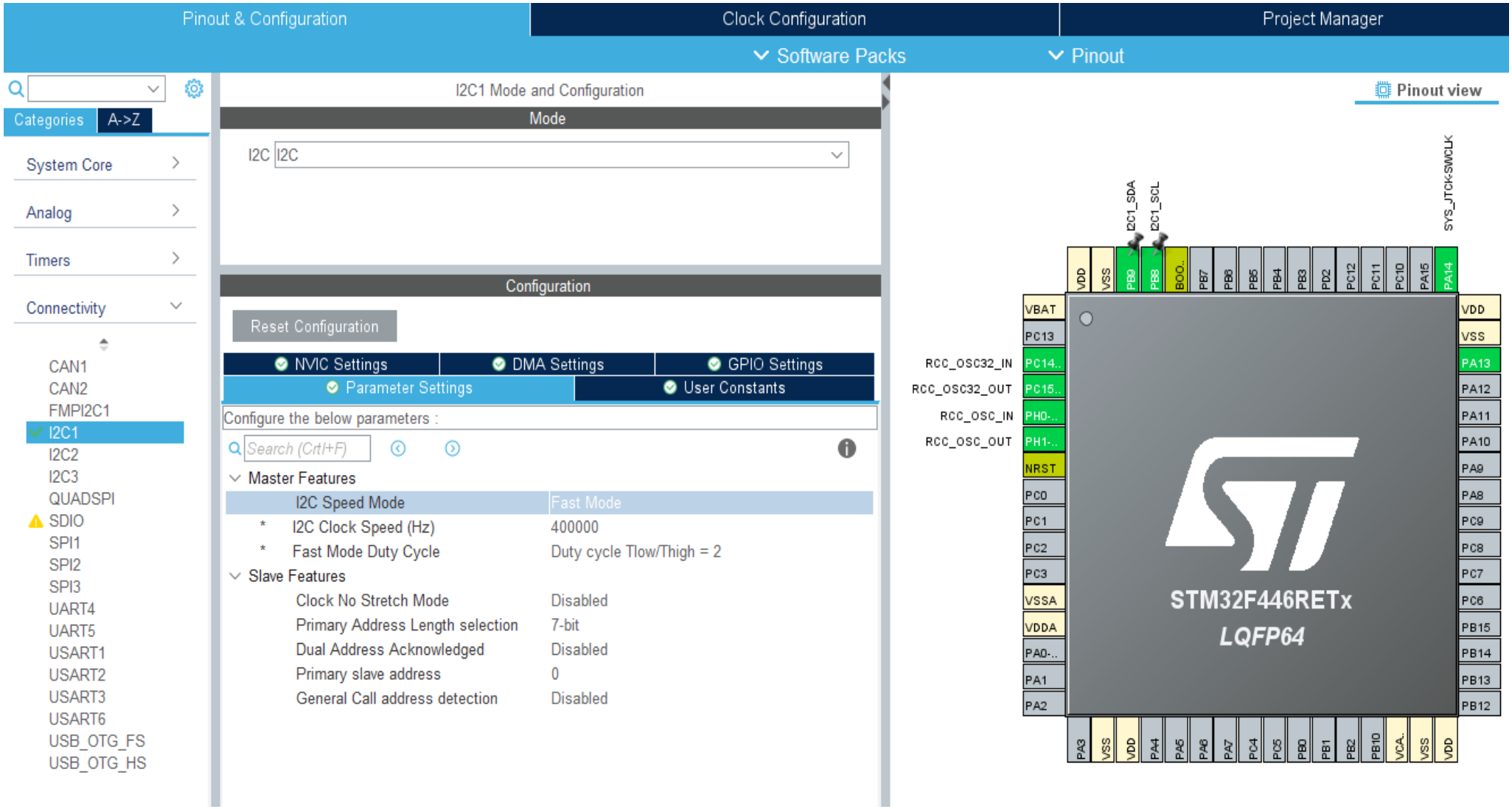
Az I2C1 kivezetések konfigurálása

- Ha nem az alapértelmezett lábkiosztást akarjuk használni, akkor a konfigurálást a kivezetések beállításával kell kezdeni!
- Kattintsunk rá a **PB8** lábra és válasszuk az **I2C_SCL** alternatív funkciót!
- Kattintsunk rá a **PB9** lábra és válasszuk az **I2C_SDA** funkciót!
- Ekkor **PB8** és **PB9** még narancssárga (majd akkor zöldülnek ki, ha engedélyezzük az **I2C1** csatornát)



Az I2C modul konfigurálása

- I2C mód, Fast mód, 400 000 bit/s, 1:2 kitöltésű órajel, 7 bites címzés



F446RE_oled_I2C: main.c (részlet)

```
static void MX_I2C1_Init(void) {

  hi2c1.Instance = I2C1;
  hi2c1.Init.ClockSpeed = 400000;
  hi2c1.Init.DutyCycle = I2C_DUTYCYCLE_2;
  hi2c1.Init.OwnAddress1 = 0;
  hi2c1.Init.AddressingMode = I2C_ADDRESSINGMODE_7BIT;
  hi2c1.Init.DualAddressMode = I2C_DUALADDRESS_DISABLE;
  hi2c1.Init.OwnAddress2 = 0;
  hi2c1.Init.GeneralCallMode = I2C_GENERALCALL_DISABLE;
  hi2c1.Init.NoStretchMode = I2C_NOSTRETCH_DISABLE;
  if (HAL_I2C_Init(&hi2c1) != HAL_OK)
  {
    Error_Handler();
  }
}
```

Forrásfájlok hozzáadása a projekthez

- A grafikus konfigurálás után a létrehozott projekthez még hozzá kell adnunk az OLED kijelzők kezeléséhez szükséges fejléc- és forrásfájlokat (`ssd1306.h, fonts.h, ssd1306.c` és `fonts.c`), amire több út is kínálkozik:
  - Bemásoljuk a fájlokat egy üres mappába, a mappát importáljuk az **STM32CubeIDE** környezetbe, majd egérrel áthúzzuk a fájlokat a projektünkbe
  - File→New paranccsal új header ill .c forrásállományt hozunk létre, majd ezekbe Copy/Paste módszerrel bemásoljuk a kívánt tartalmat
- A forrásfájlok hozzáadása után nyissuk meg a `main.c` állományt, majd egészítsük ki a következő oldalon bemutatott programlista szerint!

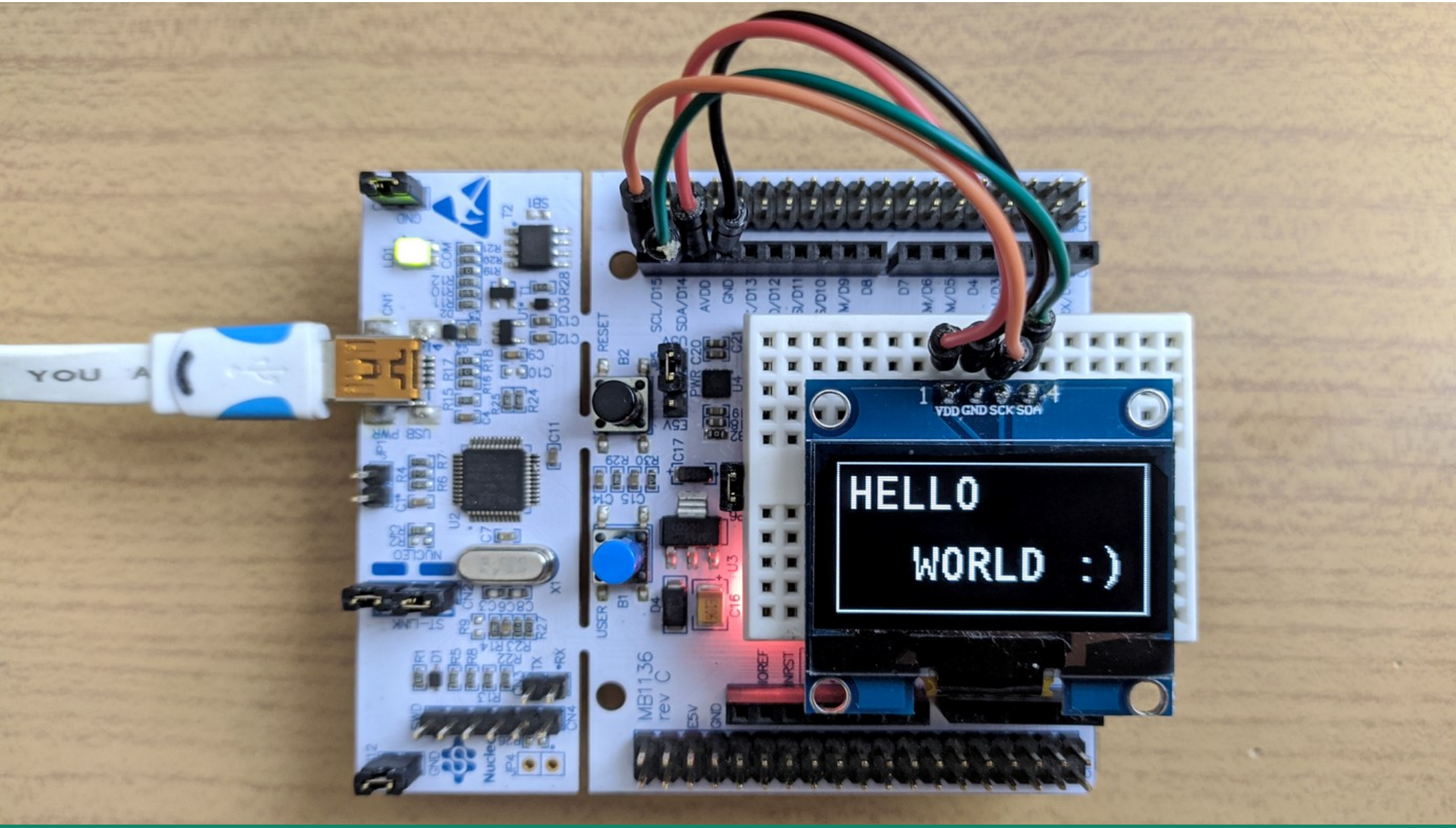
F446RE_oled_I2C: main.c (részlet)

```
#include "main.h"
#include "ssd1306.h"
I2C_HandleTypeDef hi2c1;
OledHandleTypedef oled = {
  .type = SSD1306_I2C,          // SH1106 or SSD1306
  .orientation = 1,             // 1: downward 0: upward
  .i2c = &hi2c1,                // I2C1 (now at RB9/RB8)
  .i2c_address = 0x78           // 0x78 (0x3c<<1) or 0x7A (0x3D<<1)
};

void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_I2C1_Init(void);

int main(void) {
  HAL_Init(); SystemClock_Config(); MX_GPIO_Init(); MX_I2C1_Init();
  SSD1306_Init(&oled);          // Inicializálás
  //--- Írjunk ki valamit! -------
  SSD1306_GotoXY(5, 5);
  SSD1306_Puts("HELLO", &Font_11x18, SSD1306_COLOR_WHITE);
  SSD1306_GotoXY(10, 35);
  SSD1306_Puts("  WORLD :)", &Font_11x18, SSD1306_COLOR_WHITE);
  //--- Keretező téglalap kirajzolása ------
  SSD1306_DrawRectangle(0,0,127,63,SSD1306_COLOR_WHITE);
  SSD1306_UpdateScreen();       // Megjelenítés
  while (1) {  }
}
```

A kijelző konfigurálása

F446RE_oled_I2C: futási eredmény

- Ha a körvonal látszik, akkor jól állítottuk be a kijelző típusát...



F446RE_horse_anim: bitmap animáció

- Ebben a programban azt mutatjuk be, hogy a DrawBitmap() függvény segítségével hogyan jeleníthetjük meg egy mozgás fázisképeit, a folyamatos mozgás illúzióját keltve
- Ez az animáció is a Controllerstech.com honlapon közzétett demóprogram (Oled display using I2C and STM32) része, amelyet adaptáltunk és két változatban is bemutatjuk:
  - **F446RE_horse_anim1** – az eredeti, **DrawBitmap**() segítségével rajzoló program, amelyet az általunk átírt **ssd1306** OLED könyvtárral összeházasítottunk
  - **F446RE_horse_anim2** – a módosított (felgyorsított) program, amelyben a **DrawBitmap**() függvényt csak a fázisképek áttranszformálására használjuk
- **A projekt létrehozása és konfigurálása** ugyanúgy történik, mint az előző programnál, ezen kívül a projekthez hozz kell adni a `horse_anim.h` állományt, amely a fázisképeket tárolja

horse_anim.h (részlet)

```
const unsigned char horse1 [] = {
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x3F, 0xF8, 0x60, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x1F, 0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x7F, 0xFF, 0xFF, 0xC0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x05, 0xFF, 0xFF, 0xFF, 0xE4, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x0F, 0xFF, 0xFF, 0xFF, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x0F, 0x00, 0x0F, 0xFF, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x06, 0x00, 0x00, 0xFF, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x18, 0x60, 0x00, 0x1F, 0xF8, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
 . . .
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};

const unsigned char horse2 [] = {
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
 . . .
```

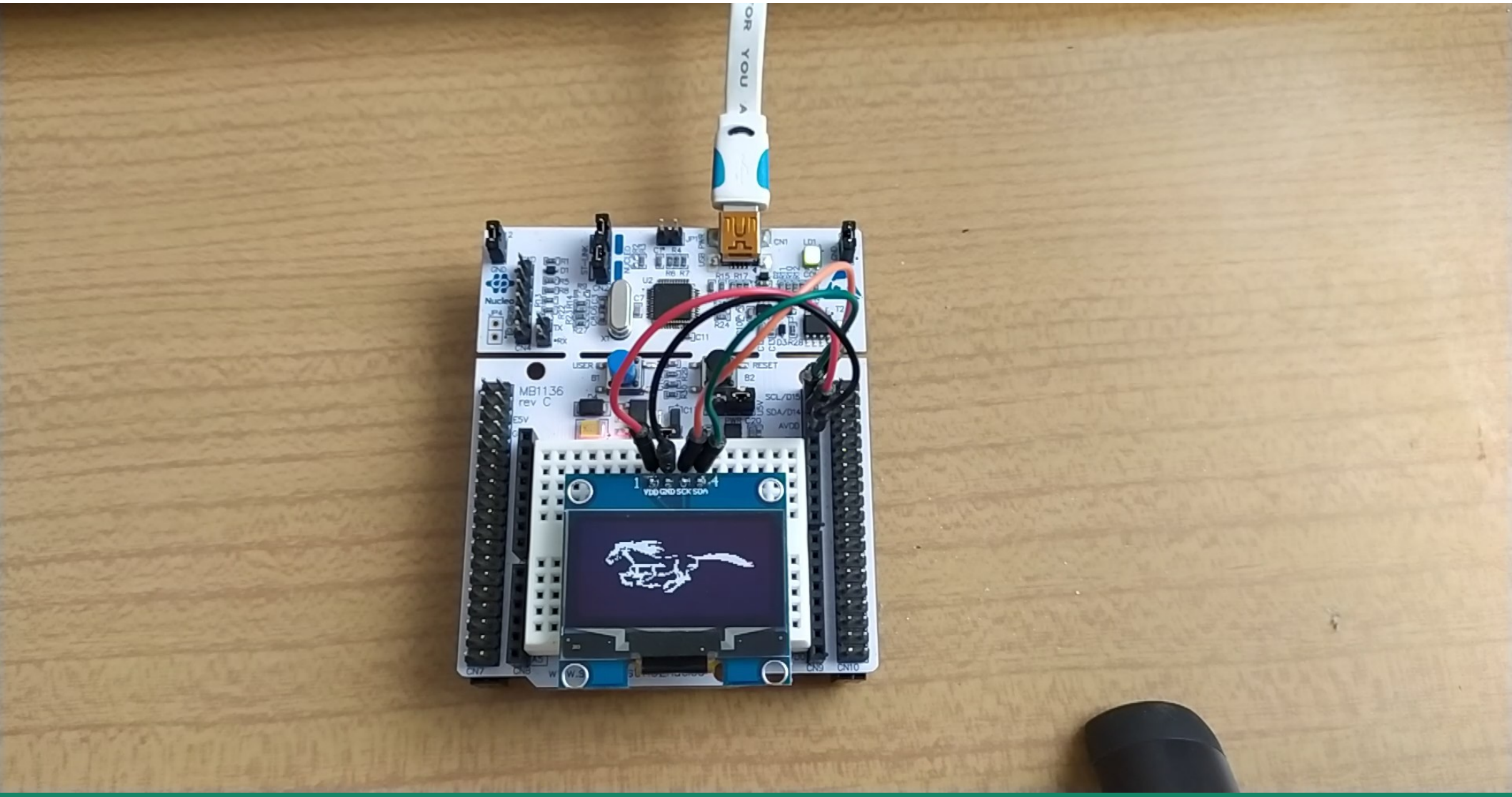
F446RE_horse_anim1: main.c (részlet)

```
#include "main.h"
#include "ssd1306.h"
#include "horse_anim.h"
I2C_HandleTypeDef hi2c1;
OledHandleTypedef oled = {
  .type = SSD1306_I2C,          // SH1106 or SSD1306
  .orientation = 1,             // 1: downward 0: upward
  .i2c = &hi2c1,                // I2C1 (now at RB9/RB8)
  .i2c_address = 0x78           // 0x78 (0x3c<<1) or 0x7A (0x3D<<1)
};
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_I2C1_Init(void);

int main(void) {
  HAL_Init(); SystemClock_Config(); MX_GPIO_Init(); MX_I2C1_Init();
  SSD1306_Init(&oled);          // Inicializálás
  while (1) {
    SSD1306_Clear();
    SSD1306_DrawBitmap(0, 0, horse1, 128, 64, 1);  // Első fáziskép kirajzolása
    SSD1306_UpdateScreen();
    . . .
    SSD1306_Clear();
    SSD1306_DrawBitmap(0, 0, horse10, 128, 64, 1);  // Utolsó fáziskép kirajzolása
    SSD1306_UpdateScreen();
  }
}
```

A kijelző konfigurálása

F446RE_horse_anim1: futási eredmény

- Mit tehetünk a kép zavaró villogása ellen? Az `SSD1306_Clear()` nem hagyható ki, mert az `SSD1306_DrawBitmap()` csak az 1 tartalmú pontokat írja ki



F446RE_horse_anim2: main.c (részlet)

```c
#include "main.h"
#include <string.h>
#include "ssd1306.h"
#include "horse_anim.h"
I2C_HandleTypeDef hi2c1;
OledHandleTypedef oled = {
  .type = SSD1306_I2C,          // SH1106 or SSD1306
  .orientation = 1,             // 1: downward 0: upward
  .i2c = &hi2c1,                // I2C1 (now at RB9/RB8)
  .i2c_address = 0x78           // 0x78 (0x3c<<1) or 0x7A (0x3D<<1)
};
uint8_t bitmap[10][1024];       // Itt tároljuk a fázisképeket!!!

void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_I2C1_Init(void);

int main(void) {
  HAL_Init(); SystemClock_Config(); MX_GPIO_Init(); MX_I2C1_Init();
  SSD1306_Init(&oled);          // Inicializálás
  SSD1306_Clear();
  SSD1306_DrawBitmap(0, 0, horse1, 128, 64, 1);
  memcpy(&bitmap[0], oled.buffer, 1024);
    . . .
  SSD1306_Clear();
  SSD1306_DrawBitmap(0, 0, horse10, 128, 64, 1);  // Utolsó fáziskép kirajzolása
  memcpy(&bitmap[9], oled.buffer, 1024);
```

A kijelző konfigurálása

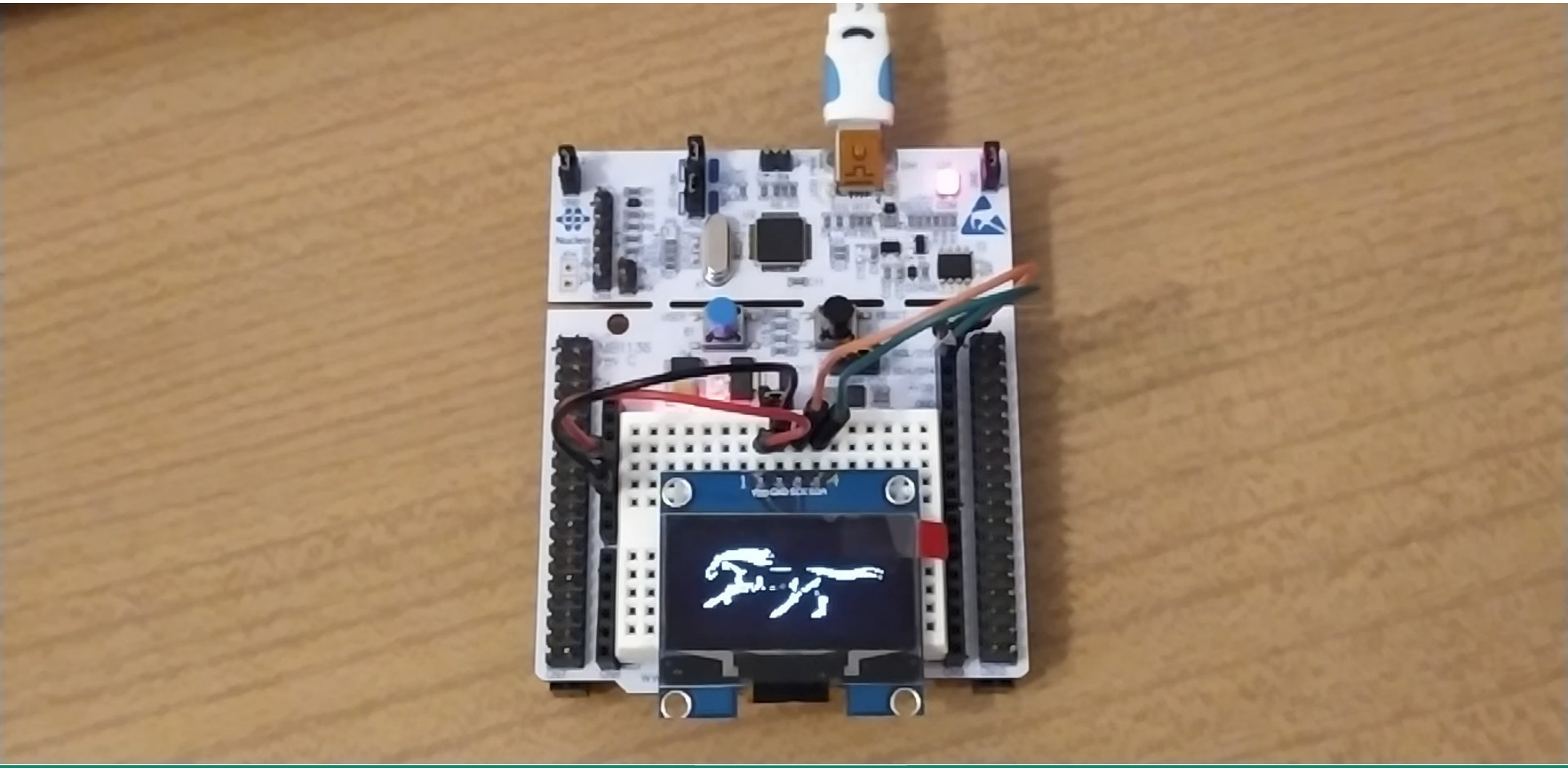
F446RE_horse_anim2: main.c (részlet)

```
  /* Infinite loop */
  while (1)  {
    for(int j=0; j<10; j++) {
      memcpy(oled.buffer, &bitmap[j][0], 1024);     // Kép másolása
      SSD1306_UpdateScreen();                       // Megjelenítés
      HAL_Delay(10);                                // Még várnunk is kell...
    }
  }
}
```

- A **memcpy(dest, source, size)** függvény memóriából memóriába másol
- A fenti megoldás helyett még hatékonyabb volna, ha közvetlenül az **I2C** buszon küldenénk ki az adatokat, vagy mi adnánk át a buffer címét az **ssd1306.c** programkönyvtárnak, és nem fordítva...

F446RE_horse_anim2: futási eredmény

- Az előző projekthez képest eltűnt a villogás és a képkockák kirajzolása közben még várakoznunk is kellett...
- A gyorsaság ára a RAM-ban lefoglalt 10 db. framebuffer (10 kbájt)



Hogyan készíthetünk saját animációt?

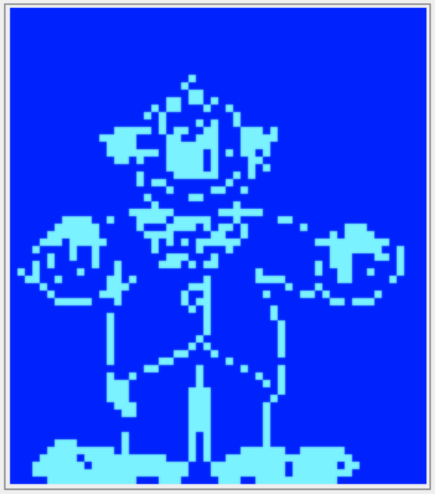
- Szükségünk lesz a megrajzolt fázisképekre: 5 – 10 db fekete-fehér, legfeljebb 128x64 képpont (vagy annál kisebb méretű) .PNG vagy .BMP képre
- A Nuts and Volts magazinban Tom Kibalo egy 2010-es cikkében egy segédprogramot ajánl a képek konvertálására és egy letölthető mintát is ad (small_clowns.zip) ami egy zsonglőrködő bohócot ábrázol 8 db 56x64 pixeles kép formájában
- A kissé már elavult segédprogram telepítése helyett most egy online képkonvertálási lehetőséget ajánlok a Marlin honlapján: https://marlinfw.org/tools/u8glib/converter.html
- Ennek az eszköznek a segítségével a képeket egyenként C konstans tömbökké konvertáljuk, majd összefűzzük a **clown_anim.h** állományban
- A megjelenítés a **F446RE_horse_anim1** projekthez hasonlóan…

https://marlinfw.org/tools/u8glib/converter.html

Marlin | About Marlin | Download | Configure | Install | Tools | Help | Search | Edit Page

Bitmap Converter

Convert image files and data into Marlin Boot Screens and Status Screen Logos for Graphical Displays. Dark colors show up as "on" pixels. Light colors will be transparent.



Paste image or C/C++ here. Choose File CLOWN1.BMP

○ Marlin 1.x ◉ Marlin 2.x

☐ Light ☐ Invert ☐ Binary ☑ ASCII Art ☐ Narrow

◉ Bitmap ○ Boot ○ Status

Copy the following code into your program:

```
/**
 * Made with Marlin Bitmap Converter
 * https://marlinfw.org/tools/u8glib/converter.html
 *
 * This bitmap from the file 'CLOWN1.BMP'
 */
#pragma once

const unsigned char bitmap_31xox4[] PROGMEM = {
  0x00,0x00,0x00,0x00,0x00,0x00,0x00, // .......................................................
  0x00,0x00,0x00,0x00,0x00,0x00,0x00, // .......................................................
  0x00,0x00,0x00,0x00,0x00,0x00,0x00, // .......................................................
  0x00,0x00,0x00,0x00,0x00,0x00,0x00, // .......................................................
```

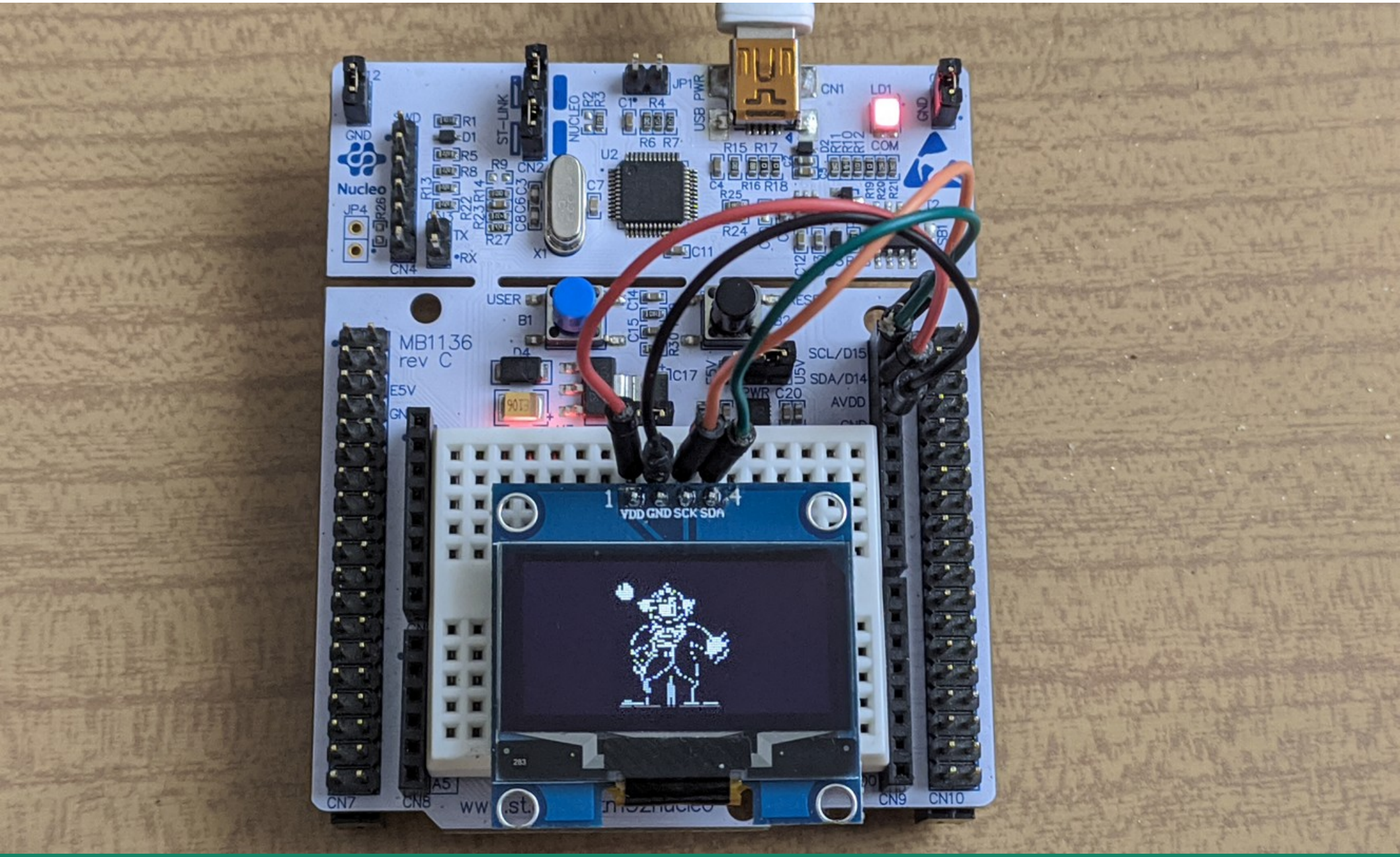
```
const unsigned char bitmap_31xox4[] PROGMEM = {
  0x00,0x00,0x00,0x00,0x00,0x00,0x00,  // ........................................................
  0x00,0x00,0x00,0x00,0x00,0x00,0x00,  // ........................................................
  0x00,0x00,0x00,0x00,0x00,0x00,0x00,  // ........................................................
  0x00,0x00,0x00,0x00,0x00,0x00,0x00,  // ........................................................
  0x00,0x00,0x00,0x00,0x00,0x00,0x00,  // ........................................................
  0x00,0x00,0x00,0x00,0x00,0x00,0x00,  // ........................................................
  0x00,0x00,0x00,0x00,0x00,0x00,0x00,  // ........................................................
  0x00,0x00,0x00,0x00,0x00,0x00,0x00,  // ........................................................
  0x00,0x00,0x00,0x00,0x00,0x00,0x00,  // ........................................................
  0x00,0x00,0x00,0x80,0x00,0x00,0x00,  // ........................#...............................
  0x00,0x00,0x01,0x00,0x00,0x00,0x00,  // .......................#................................
  0x00,0x00,0x00,0xC0,0x00,0x00,0x00,  // ........................##..............................
  0x00,0x00,0x06,0xD0,0x00,0x00,0x00,  // .....................##.##.#............................
  0x00,0x00,0x16,0x24,0x00,0x00,0x00,  // ...................#.##...#..#..........................
  0x00,0x00,0x28,0x02,0x00,0x00,0x00,  // ..................#.#.........#.........................
  0x00,0x00,0x08,0x52,0x00,0x00,0x00,  // ....................#....#.#..#.........................
  0x00,0x03,0xEB,0x31,0xD0,0x00,0x00,  // ..............#####.#.##..##...###.#....................
  0x00,0x0F,0x86,0x71,0xF0,0x00,0x00,  // ............#####....##..###...#####....................
  0x00,0x07,0x87,0xF1,0xE0,0x00,0x00,  // .............####....#######...####.....................
  0x00,0x07,0xF7,0xD5,0xA0,0x00,0x00,  // .............#######.#####.#.#.##.#.....................
  0x00,0x03,0x47,0xD0,0xC0,0x00,0x00,  // ..............##.#...#####.#....##......................
  0x00,0x00,0x07,0xD0,0xA0,0x00,0x00,  // .....................#####.#....#.#.....................
  0x00,0x00,0x43,0xF2,0x80,0x00,0x00,  // .................#....######..#.#.......................
  0x00,0x00,0x58,0x04,0x00,0x00,0x00,  // .................#.##........#..........................
  0x00,0x00,0x04,0x19,0x00,0x00,0x00,  // .....................#.....##..#........................
  0x00,0x00,0x20,0xC0,0x00,0x00,0x00,  // ..................#.....##..............................
  0x00,0x00,0x10,0x02,0x00,0x00,0x00,  // ...................#..........#.........................
  0x00,0x00,0xFC,0x0F,0xC0,0x00,0x00,  // ................######......######......................
  0x01,0xE4,0x7B,0xE2,0x0C,0x00,0x00,  // .......####..#...####.#####...#.....##..................
  0x03,0xF0,0x47,0xAD,0x01,0x07,0x00,  // ......######.....#...####.#.##.#.......#.....###........
  0x07,0xF0,0x3C,0x19,0x00,0x17,0x80,  // .....#######......####.....##..#...........#.####.......
  0x0E,0xF8,0x15,0x7E,0x00,0x7F,0xE0,  // ....###.#####......#.#.#.######..........##########.....
  0x10,0x90,0x10,0x44,0x00,0x1F,0xC8,  // ...#....#..#.......#.....#...#.............#######..#...
  0x04,0x90,0x07,0x10,0x00,0x1C,0x68,  // .....#..#..#.........###...#...............###...##.#...
  0x14,0x12,0x0E,0xB0,0x00,0x5C,0x08,  // ...#.#.....#..#.....###.#.##.............#.###.....#...
  0x50,0x42,0x00,0x00,0x40,0x4C,0x88,  // .#.#.....#....#..................#.......#..##..#...#...
  0x32,0x06,0x80,0x20,0x78,0x0C,0x10,  // ..##..#......##.#.........#......####.......##.....#....
  0x08,0x07,0x00,0xE0,0x04,0x40,0x00,  // ....#........###........###..........#...#..............
  0x04,0x0E,0x01,0x20,0x21,0xA0,0x40,  // .....#......###........#..#.......#....##.#......#......
  0x03,0xE2,0x01,0x60,0x00,0x1B,0x80,  // ......#####...#........#.##................##.###.......
  0x00,0x00,0x00,0xA0,0x10,0x00,0x00,  // ........................#.#........#....................
  0x00,0x04,0x00,0x20,0x10,0x00,0x00,  // .............#............#........#....................
  0x00,0x04,0x00,0x20,0x08,0x00,0x00,  // .............#............#.........#...................
  0x00,0x04,0x00,0x20,0x08,0x00,0x00,  // .............#............#.........#...................
  0x00,0x04,0x00,0x40,0x08,0x00,0x00,  // .............#...........#..........#...................
  0x00,0x04,0x00,0xA0,0x08,0x00,0x00,  // .............#..........#.#.........#...................
  0x00,0x04,0x03,0x10,0x08,0x00,0x00,  // .............#........##...#........#...................
  0x00,0x04,0x0C,0x04,0x00,0x00,0x00,  // .............#......##.......#..........................
  0x00,0x00,0x30,0x41,0x08,0x00,0x00,  // ..................##.....#.....#....#...................
  0x00,0x04,0x80,0x40,0x48,0x00,0x00,  // .............#..#........#.......#..#...................
  0x00,0x07,0x00,0x40,0x28,0x00,0x00,  // .............###.........#........#.#...................
  0x00,0x07,0x00,0xE0,0x08,0x00,0x00,  // .............###........###.........#...................
  0x00,0x07,0x00,0xE0,0x10,0x00,0x00,  // .............###........###........#....................
  0x00,0x03,0x80,0xE0,0x20,0x00,0x00,  // ..............###.......###.......#.....................
  0x00,0x01,0x80,0xE0,0x20,0x00,0x00,  // ...............##.......###.......#.....................
  0x00,0x00,0x00,0xE0,0x20,0x00,0x00,  // ........................###.......#.....................
  0x00,0x00,0x00,0xE0,0x20,0x00,0x00,  // ........................###.......#.....................
  0x00,0x01,0x00,0xA0,0x20,0x00,0x00,  // ...............#........#.#.......#.....................
  0x03,0x81,0x00,0xA0,0x20,0x00,0x00,  // ......###......#........#.#.......#.....................
  0x07,0xFD,0x00,0xA1,0xF9,0xF8,0x00,  // .....#########.#........#.#....######..######...........
  0x0F,0xBF,0xF8,0xA7,0xFF,0xFC,0x00,  // ....#####.###########...#.#..################...........
  0x1F,0xDF,0xFF,0x1F,0xFB,0xF6,0x00,  // ...#######.#############...##########.######.##.........
  0x1F,0xFF,0xFF,0x1F,0xFB,0xFC,0x00,  // ...#####################...##########.########.........
  0x07,0xFF,0x8E,0x0E,0x7F,0xF0,0x00   // .....############...###.....###..###########............
};
```

F446RE_clown_anim: main.c (részlet)

```
#include "main.h"
#include "ssd1306.h"
#include "clown_anim.h"
I2C_HandleTypeDef hi2c1;
OledHandleTypedef oled = {
  .type = SSD1306_I2C,          // SH1106 or SSD1306
  .orientation = 1,             // 1: downward 0: upward
  .i2c = &hi2c1,                // I2C1 (now at RB9/RB8)
  .i2c_address = 0x78           // 0x78 (0x3c<<1) or 0x7A (0x3D<<1)
};
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_I2C1_Init(void);

int main(void) {
  HAL_Init(); SystemClock_Config(); MX_GPIO_Init(); MX_I2C1_Init();
  SSD1306_Init(&oled);          // Inicializálás
  while (1) {
    SSD1306_Clear();
    SSD1306_DrawBitmap(36, 0, clown1, 56, 64, 1);  // Első fáziskép kirajzolása
    SSD1306_UpdateScreen();
    . . .
    SSD1306_Clear();
    SSD1306_DrawBitmap(36, 0, clown8, 56, 64, 1);  // Utolsó fáziskép kirajzolása
    SSD1306_UpdateScreen();
  }
}
```

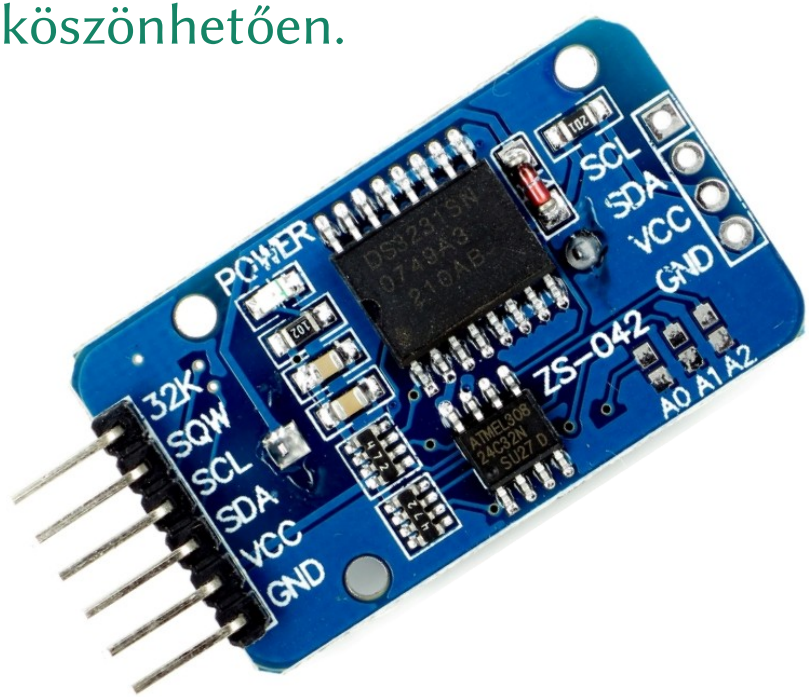
A kijelző konfigurálása

F446RE_clown_anim: futási eredmény



DS3231 Real-time óra modul

- Oszcillátor, óra és naptár egy tokban. A tápfeszültség megszűnésekor a hátoldalán elhelyezett telepről üzemel tovább.
- A modul többé-kevésbé cserekompatibilis a **DS1307**-tel, egy 4 kB EEPROM is tartalmaz, de van néhány eltérés:
  - pontosabb óra (±2 ppm a –40 – 80 °C tartományban) a beépített hőkompenzált oszcillátornak köszönhetően.
  - két riasztási időpont is megadható
  - van beépített hőmérője
  - nincs belső RAM
  - Vbat 4.2 V-os Li akkuval is táplálható!
- EEPROM I2C címe: **0x57** (állítható)
- DS3231 I2C címe: **0x68**



- Adatlap: datasheets.maximintegrated.com/en/ds/DS3231.pdf

DS3231 regiszterek

ADDRESS	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0 LSB	FUNCTION	RANGE
00h	0	10 Seconds			Seconds				Seconds	00–59
01h	0	10 Minutes			Minutes				Minutes	00–59
02h	0	12/$\overline{24}$	$\overline{AM}$/PM 20 Hour	10 Hour	Hour				Hours	1–12 + $\overline{AM}$/PM 00–23
03h	0	0	0	0	0	Day			Day	1–7
04h	0	0	10 Date		Date				Date	01–31
05h	Century	0	0	10 Month	Month				Month/ Century	01–12 + Century
06h	10 Year				Year				Year	00–99
07h	A1M1	10 Seconds			Seconds				Alarm 1 Seconds	00–59
08h	A1M2	10 Minutes			Minutes				Alarm 1 Minutes	00–59
09h	A1M3	12/$\overline{24}$	$\overline{AM}$/PM 20 Hour	10 Hour	Hour				Alarm 1 Hours	1–12 + $\overline{AM}$/PM 00–23
0Ah	A1M4	DY/$\overline{DT}$	10 Date		Day				Alarm 1 Day	1–7
					Date				Alarm 1 Date	1–31
0Bh	A2M2	10 Minutes			Minutes				Alarm 2 Minutes	00–59
0Ch	A2M3	12/$\overline{24}$	$\overline{AM}$/PM 20 Hour	10 Hour	Hour				Alarm 2 Hours	1–12 + $\overline{AM}$/PM 00–23
0Dh	A2M4	DY/$\overline{DT}$	10 Date		Day				Alarm 2 Day	1–7
					Date				Alarm 2 Date	1–31
0Eh	$\overline{EOSC}$	BBSQW	CONV	RS2	RS1	INTCN	A2IE	A1IE	Control	—
0Fh	OSF	0	0	0	EN32kHz	BSY	A2F	A1F	Control/Status	—
10h	SIGN	DATA	DATA	DATA	DATA	DATA	DATA	DATA	Aging Offset	—
11h	SIGN	DATA	DATA	DATA	DATA	DATA	DATA	DATA	MSB of Temp	—
12h	DATA	DATA	0	0	0	0	0	0	LSB of Temp	—

A DS3231 programkönyvtár

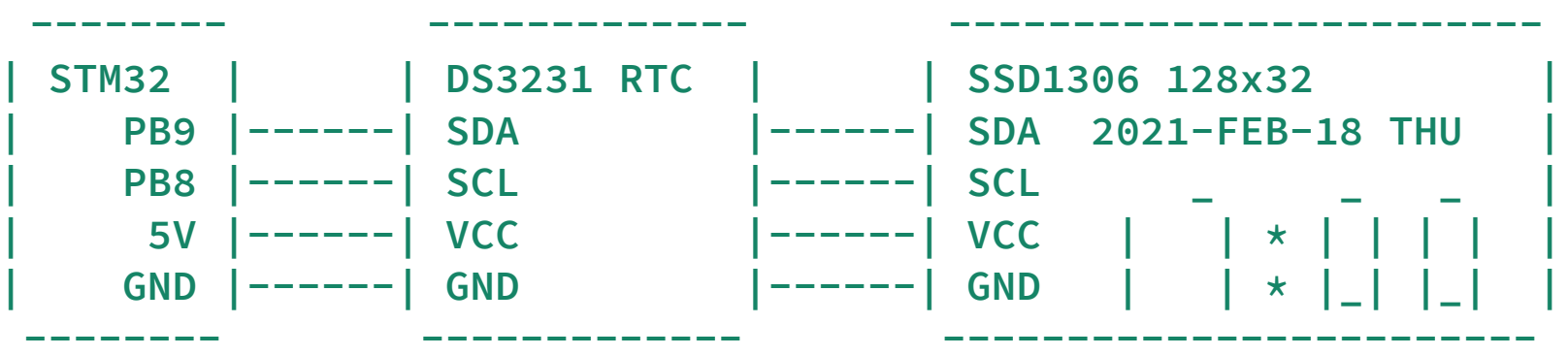
- Mintapéldánkban *Jihoon Lee* **DS3231** programkönyvtárát használjuk (github.com/eziya/STM32_HAL_DS3231), mint az előző előadásban is
- `void DS3231_Init(I2C_HandleTypeDef *handle);`
- `bool DS3231_GetTime(_RTC *rtc);`
- `bool DS3231_SetTime(_RTC *rtc);`
- `bool DS3231_ReadTemperature(float *temp);`
- `bool DS3231_SetAlarm1(AlarmMode mode, uint8_t date, uint8_t hour, uint8_t min, uint8_t sec);`
- `bool DS3231_ClearAlarm1();`
- `bool ReadRegister(uint8_t regAddr, uint8_t *value);`
- `bool WriteRegister(uint8_t regAddr, uint8_t value);`
- Az **F446RE_DS3231RTC** projektben inicializáljuk, illetve periodikusan kiolvassuk az óra modult (idő, hőmérséklet), majd az **SSD1306 128x32** OLED kijelzőn az I2C buszon keresztül kiíratjuk az eredményt
- A projekt létrehozása és konfigurálása ugyanúgy történik, mint az előző projekteknél, de az I2C buszt **standard módban** (100 kbit/s) használjuk

F446RE_oled_RTC

- A grafikus konfigurálás után a létrehozott projekthez még hozzá kell adnunk a github.com/eziya/STM32_HAL_DS3231 archívumból letöltött **`stm32_ds3231.h`** és **`stm32_ds3231.c`** forrásfájlokat
- A projekthez természetese hozzá kell adni az OLED kijelző kezeléséhez az előző projektekben használt **`ssd1306.h, fonts.h, ssd1306.c`** és **`fonts.c`** állományokat is
- Az **SSD1306** 128x32 kijelző konfigurálásához a **`ssd1306.h`** állomány elejére szúrjuk be az alábbi sort:

```
#define SSD1306_HEIGHT          32
```

- **Kapcsolás:**



F446RE_clock_RTC: main.c (részlet)

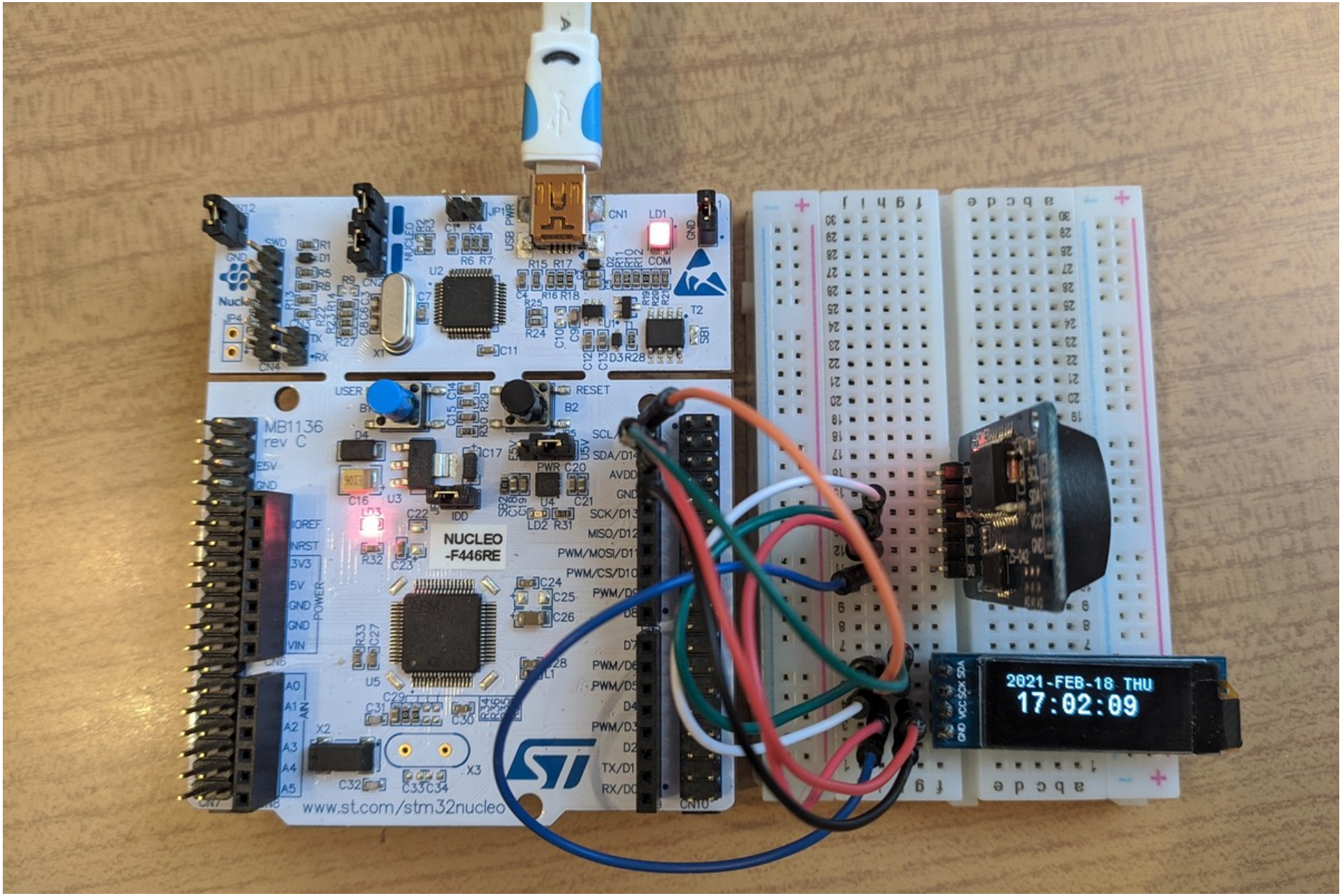
```
#include "main.h"
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include "ssd1306.h"
#include "stm32_ds3231.h"
I2C_HandleTypeDef hi2c1;
OledHandleTypedef oled = {
  .type = SSD1306_I2C,          // SH1106 or SSD1306
  .orientation = 1,             // 1: downward 0: upward
  .i2c = &hi2c1,                // I2C1 (now at RB9/RB8)
  .i2c_address = 0x78           // 0x78 (0x3c<<1) or 0x7A (0x3D<<1)
};

_RTC rtc = {
    .Year = 21, .Month = 2, .Date = 18,
    .DaysOfWeek = THURSDAY,
    .Hour = 17, .Min = 0, .Sec = 0
};

uint8_t status;
char msg[80];
char dow[8][4] = {"Err", "SUN", "MON", "TUE", "WED", "THU", "FRI", "SAT"};
char moy[13][4] = {"Err", "JAN", "FEB", "MAR", "APR", "MAY", "JUN", "JUL",
                   "AUG", "SEP", "OCT", "NOV", "DEC"};
```

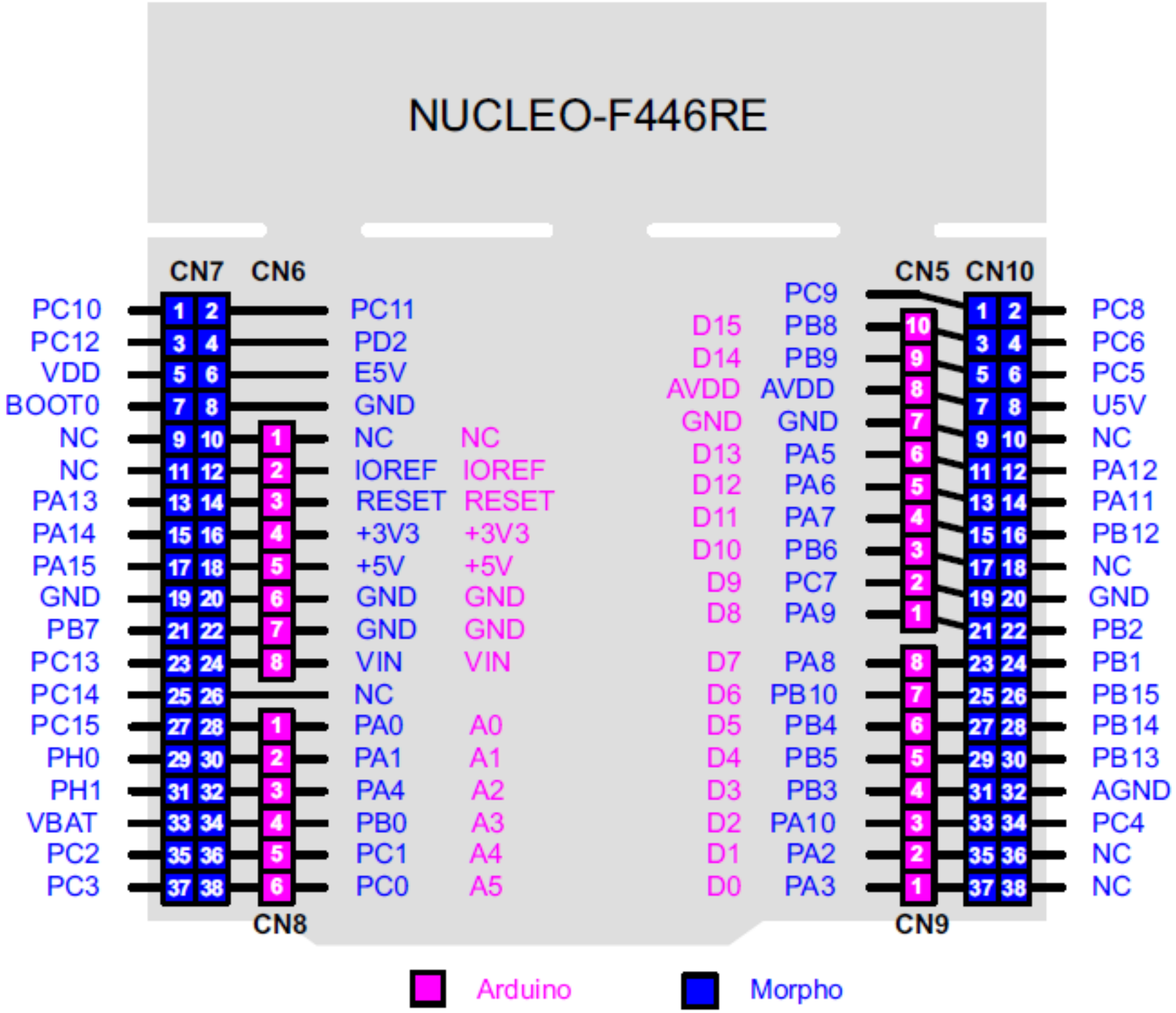
A kijelző konfigurálása

F446RE_clock_RTC: main.c (részlet)

```
int main(void) {
  HAL_Init();
  SystemClock_Config();
  MX_GPIO_Init();
  MX_I2C1_Init();

  DS3231_Init(&hi2c1);
  status = DS3231_SetTime(&rtc);
  SSD1306_Init(&oled);

  while (1) {
    status = DS3231_GetTime(&rtc);
    sprintf(msg, "20%02d-%s-%02d %s",rtc.Year, moy[rtc.Month],rtc.Date,
                                     dow[rtc.DaysOfWeek]);
    SSD1306_GotoXY(5, 1);
    SSD1306_Puts(msg, &Font_7x10, SSD1306_COLOR_WHITE);

    sprintf(msg,"%02d:%02d:%02d",rtc.Hour,rtc.Min,rtc.Sec);
    SSD1306_GotoXY(12, 12);
    SSD1306_Puts(msg, &Font_11x18, SSD1306_COLOR_WHITE);
    SSD1306_UpdateScreen();     // Megjelenítés
    HAL_Delay(1000);
  }
}
```

F446RE_clock_RTC: futási eredmény




NUCLEO-F446RE
CN7
CN6
PC10
PC12
VDD
BOOT0
NC
NC
PA13
PA14
PA15
GND
PB7
PC13
PC14
PC15
PH0
PH1
VBAT
PC2
PC3
PC11
PD2
E5V
GND
NC
IOREF
RESET
+3V3
+5V
GND
GND
VIN
NC
PA0
PA1
PA4
PB0
PC1
PC0
NC
IOREF
RESET
+3V3
+5V
GND
GND
VIN
A0
A1
A2
A3
A4
A5
CN8
CN5
CN10
PC9
D15
D14
AVDD
GND
D13
D12
D11
D10
D9
D8
D7
D6
D5
D4
D3
D2
D1
D0
PB8
PB9
AVDD
GND
PA5
PA6
PA7
PB6
PC7
PA9
PA8
PB10
PB4
PB5
PB3
PA10
PA2
PA3
PC8
PC6
PC5
U5V
NC
PA12
PA11
PB12
NC
GND
PB2
PB1
PB15
PB14
PB13
AGND
PC4
NC
NC
CN9
Arduino
Morpho