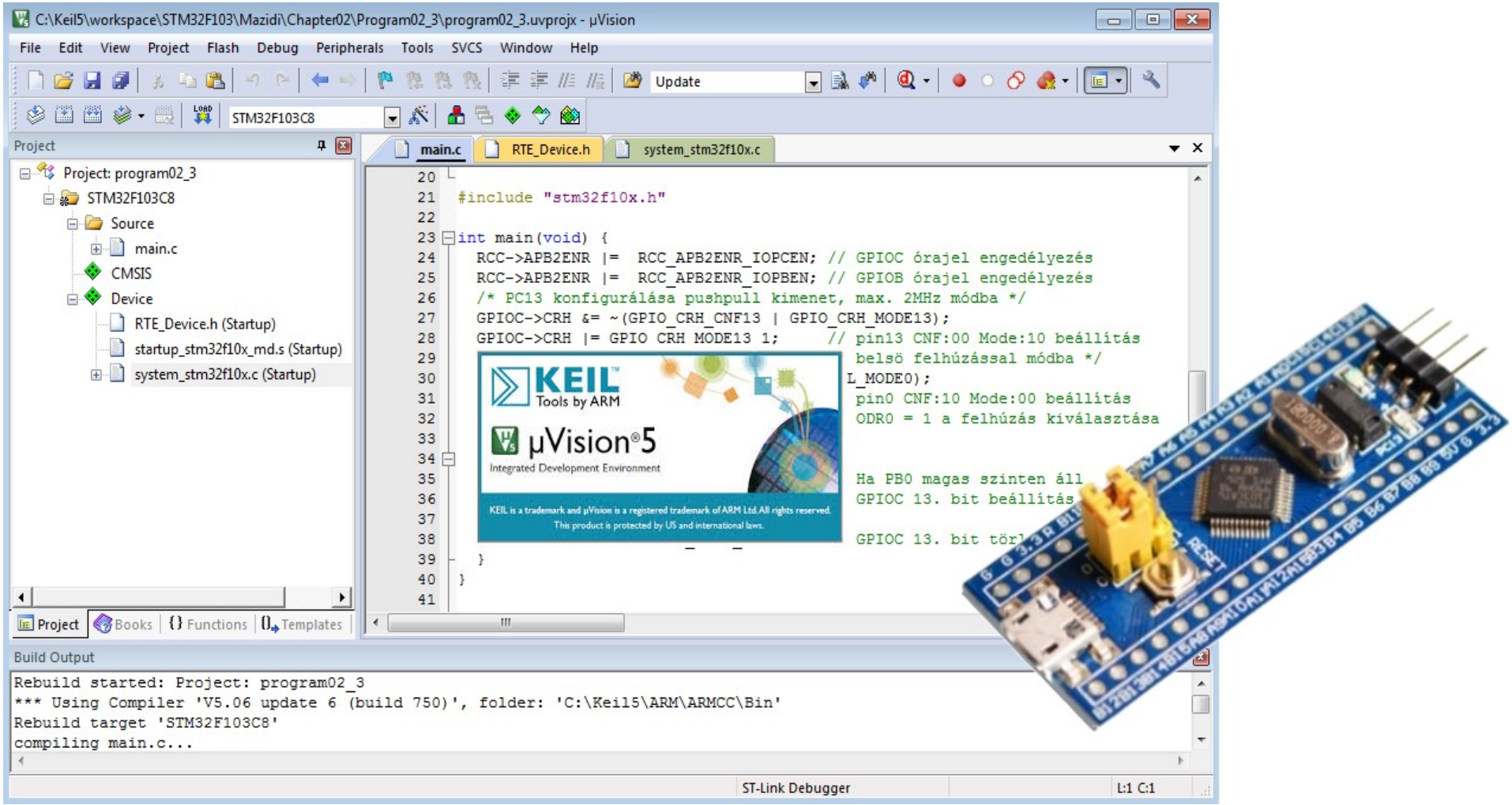
STM32 mikrovezérlők programozása ARM Keil környezetben



11. Az I2C kommunikációs csatorna – 3. rész

Felhasznált és ajánlott irodalom

- Joseph Yiu: Cortex-M for Beginners
- Joseph Yiu: The Definitive Guide To The ARM CORTEX-M3
- Muhammad Ali Mazidi, Shujen Chen, Eshragh Ghaemi: STM32 Arm Programming for Embedded Systems
- Alexander Tarasov: **Курс «Штурмуем STM32»**
- Warren Gay: Beginning STM32 - Developing with FreeRTOS, libopencm3 and GCC
- ARM Keil MDK Getting started
- **STM32F103C8** adatlap és termékinfo
- **STM32F103** Family Reference Manual

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 SSD1316 vezérlőjű 0.96” képátlójú, 128x64 felbontású kijelzővel, de az inicializálásban van néhány különbség

Program09_07: SSD1306 128x32 demo

- A projekthez az alábbi forrásfájlokat kell hozzáadni:
  - **main.c** – a főprogram
  - **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

Ezekben van a fő különbség

```
#include "stm32f10x.h"
#include "ssd1306.h"
int main(void) {
  SSD1306_Init();             // inicializálás
  //--- Írjunk ki valamit! -------
  SSD1306_GotoXY(5, 1);
  SSD1306_Puts("2020-FEB-12 TUE",&Font_7x10,SSD1306_COLOR_WHITE);
  SSD1306_GotoXY(5, 12);
  SSD1306_Puts(" 17:30:45", &Font_11x18, SSD1306_COLOR_WHITE);
  //--- Keretező téglalap kirajzolása ------
  //SSD1306_DrawRectangle(0,0,127,31,SSD1306_COLOR_WHITE);
  SSD1306_UpdateScreen();     // Megjelenítés
  while(1) { }
}
```

Módosítások az ssd1306.h állományban

Részletek az ssd1306.h fejléc állomány elejéről

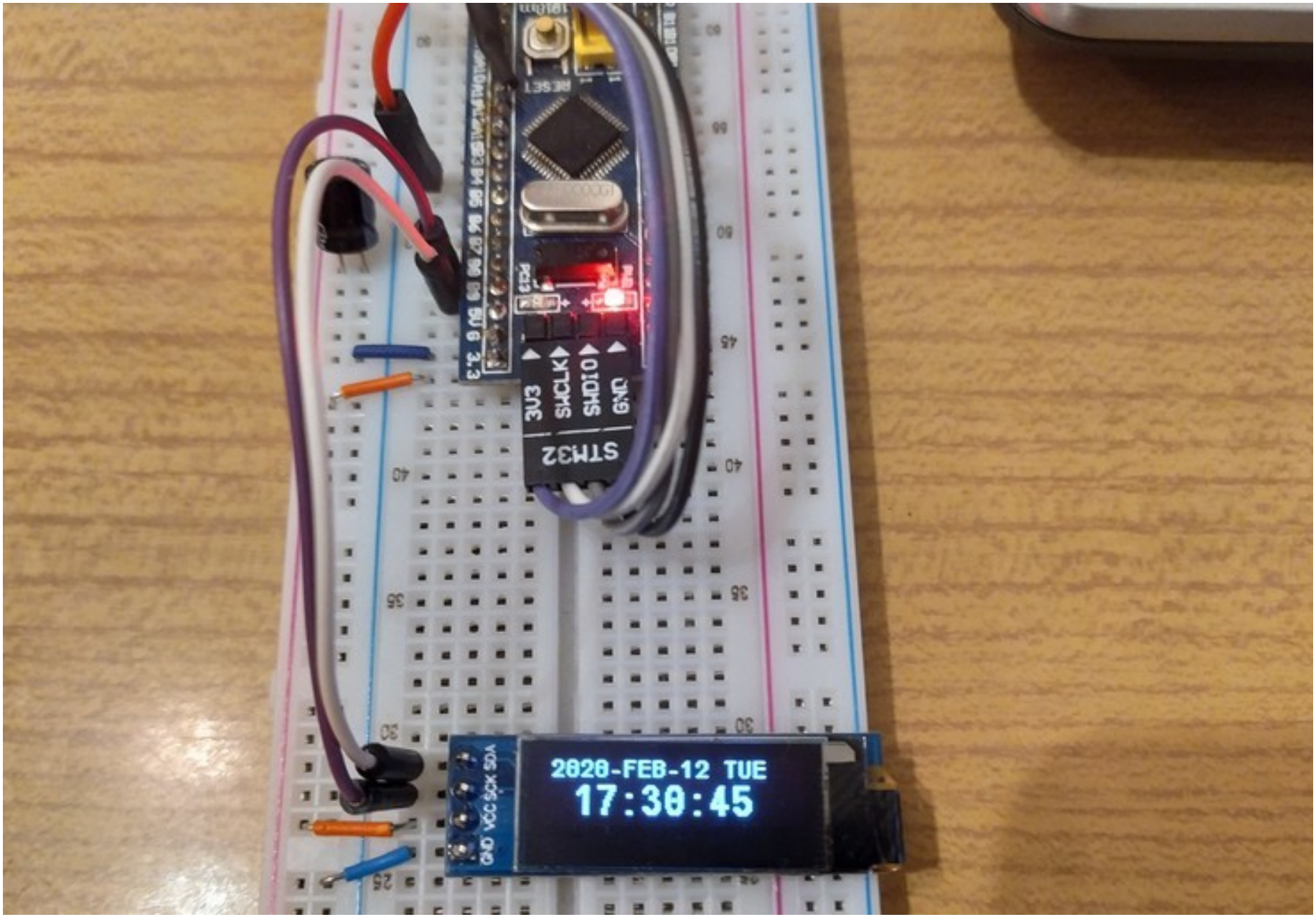
```
#include "stm32f10x.h"
#include "fonts.h"
#include "stdlib.h"
#include "string.h"

/* I2C address */
#define SSD1306_I2C_ADDR  0x78 // 3C<<1


/* SSD1306 settings */
/* SSD1306 width in pixels */
#define SSD1306_WIDTH            128

/* SSD1306 LCD height in pixels */
#define SSD1306_HEIGHT           32  // ← Itt van változás

/* SSD1306 color enumeration */
typedef enum {
  SSD1306_COLOR_BLACK = 0x00,
  SSD1306_COLOR_WHITE = 0x01
} SSD1306_COLOR_t;
```

Módosítások az ssd1306.c állományban

```
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(0xC8); //Set COM Output Scan Direction
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 0 to 127
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); //
```

Részletek az
SSD1306_Init() függvényből

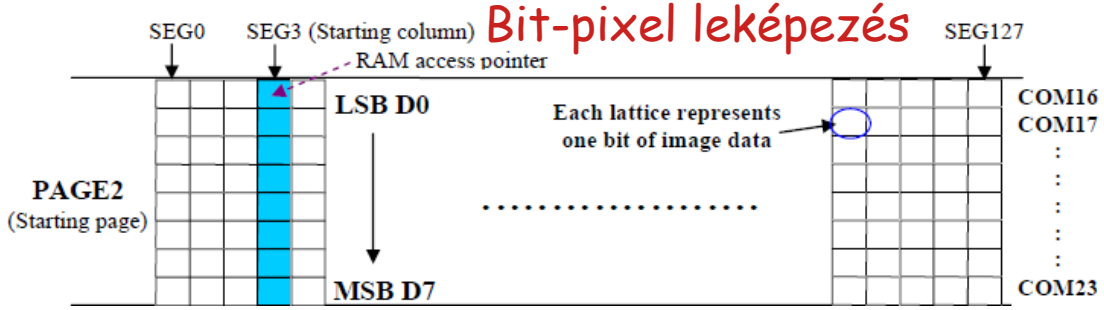
Program09_07 futási eredmény



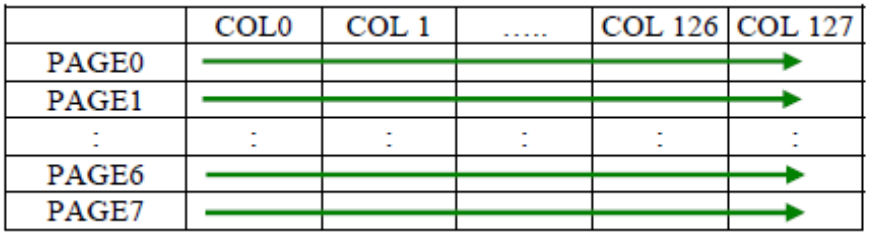
SSD1306/SH1106 parancsok

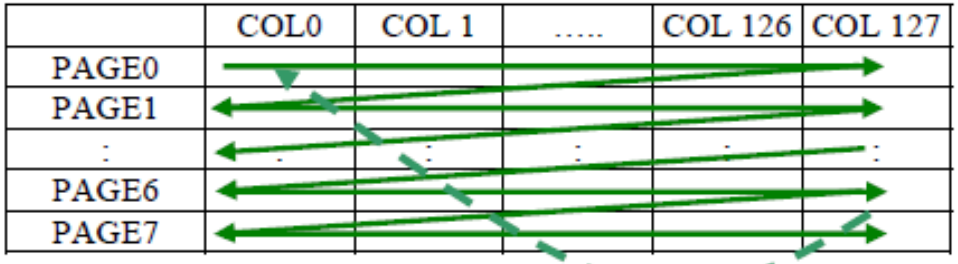
- Set Lower Column Start Address for Page Addressing Mode (00h~0Fh) az első oszlop címének alsó 4 bitje (ssd1306: 0000, sh1106: 0010)
- Set Higher Column Start Address for Page Addressing Mode (10h~1Fh) az első oszlop címének felső 4 bitje (általában 10h)
- Set Memory Addressing Mode (20h + adat[1:0])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

Bit-pixel leképezés

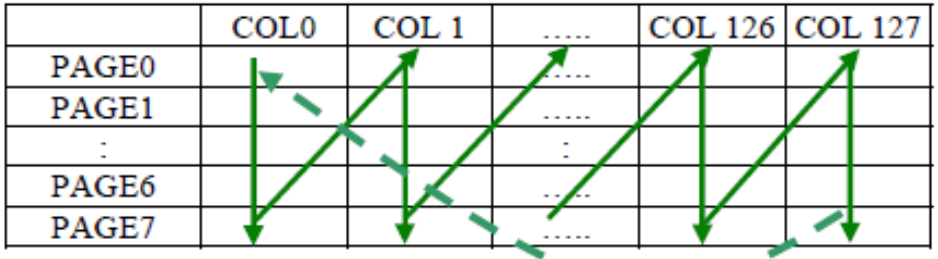


Page addressing





Horizontal addressing



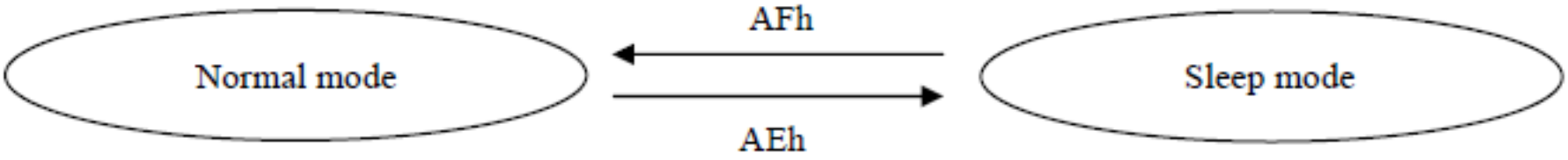
Vertical addressing

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 (POR=80h)
- Set Segment Re-map (A0h/A1h) – a leképezést állítja be (POR=A0h)
  A0: col. addr 0 ⟶ segment 0, **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 (B**0**~B**7**: 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)

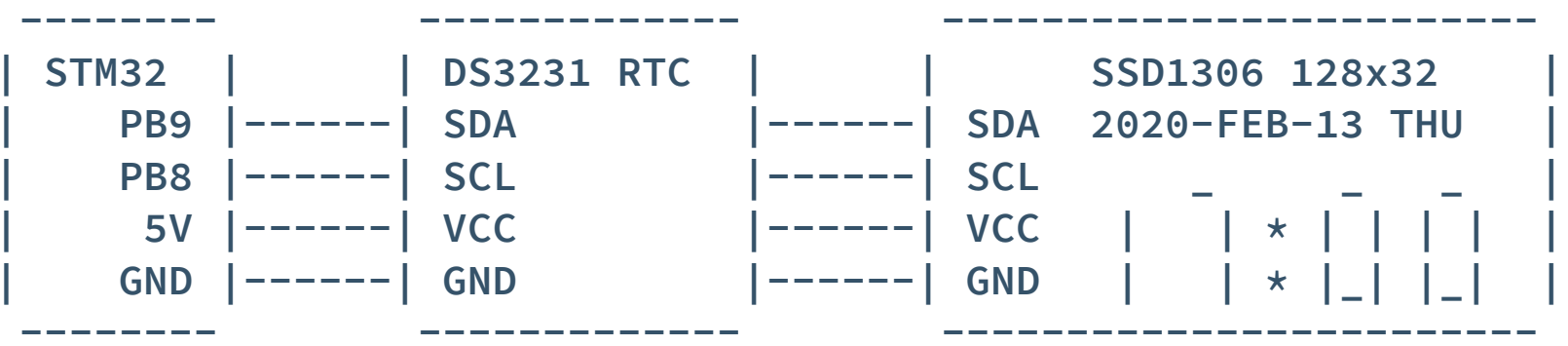
SH1106 inicializálás felülvizsgálva

```
uint8_t SSD1306_Init(void) {
    ssd1306_I2C_Init();          /* Init I2C */
    SSD1306_WRITECOMMAND(0xAE); //--display off
    SSD1306_WRITECOMMAND(0xA0); //--set segment re-map 0 to 127
    SSD1306_WRITECOMMAND(0xC0); //--Set COM Output Scan Direction 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
    SSD1306_Fill(SSD1306_COLOR_BLACK);
    SSD1306_UpdateScreen();
}
```

Ezt az inicializálást a Program09_8 projektben próbáltuk ki

oled_clock projekt

- A **Program09_4** projektben már foglalkoztunk a **DS3231 RTC** óra kiolvasásával és az eredmények feldolgozásával
- Most az **oled_clock** ugyanúgy kiolvasot adatokból az időt és a dátumot az előadás elején bemutatott SSD1306 128x32 kijezőn jelenítjük meg
- A **DS33 231 RTC**-vel szerzett tapasztalatok miatt az **I2C** buszon a standard (100 kHz) sebességet használjuk
- A kapcsolás:



- Az I2C busz kezelése az ***ssd0306.h*** és ***ssd1306.c*** állományokba került át!

oled_clock/main.c

```
#include "stm32f10x.h"
#include "ssd1306.h"

#define SLAVE_ADDR 0x68                 /* 1101 000.    DS3231 */
void displayTime(uint8_t *data);

#define dec1(x)     (x &0x0f)+48
#define dec10(x)    ((x>>4)&0x07)+48

int main(void) {
  uint8_t timeDateReadback[15];

  uint8_t lasttime =0;
  SSD1306_Init();

  while (1) {
    I2C1_ReadMulti(SLAVE_ADDR<<1, 0, timeDateReadback, 7);
    if (timeDateReadback[0] != lasttime) {
      lasttime = timeDateReadback[0];
      displayTime(timeDateReadback);
    }
  }
}
```

Az ssd1306.c állományban balra igazítva kezeljük az I2C címeket, ezért a 7-bites címet balra kell léptetni!

Csak akkor frissítjük a kijelzést, ha a másodperc lépett egyet

oled_clock/main.c (folytatás)

```
void displayTime(uint8_t *timeDate) {
  char daysOfTheWeek[8][3] = {"MON", "TUE", "WED", "THU",
                              "FRI", "SAT", "SUN"};
  char sDate[15], sTime[10];
  //--- A dátum kiíratása YYYY-MM-DD DOW formátumban ---
  sDate[0] = '2';                                  // Évszázad = 20xx
  sDate[1] = '0';
  sDate[2] = dec10(timeDate[6]);                   // Évtizedek
  sDate[3] = dec1(timeDate[6]);                    // Évek
  sDate[4] = '-';
  sDate[5] = dec10(timeDate[5]);                   // Tízhónapok
  sDate[6] = dec1(timeDate[5]);                    // Hónapok
  sDate[7] = '-';
  sDate[8] = dec10(timeDate[4]);                   // Tíznapok
  sDate[9] = dec1(timeDate[4]);                    // Napok
  sDate[10] = ' ';
  for(int i=0; i<3; i++) {
    sDate[11+i] = daysOfTheWeek[timeDate[3]-1][i];
  }
  sDate[14] = 0;
```

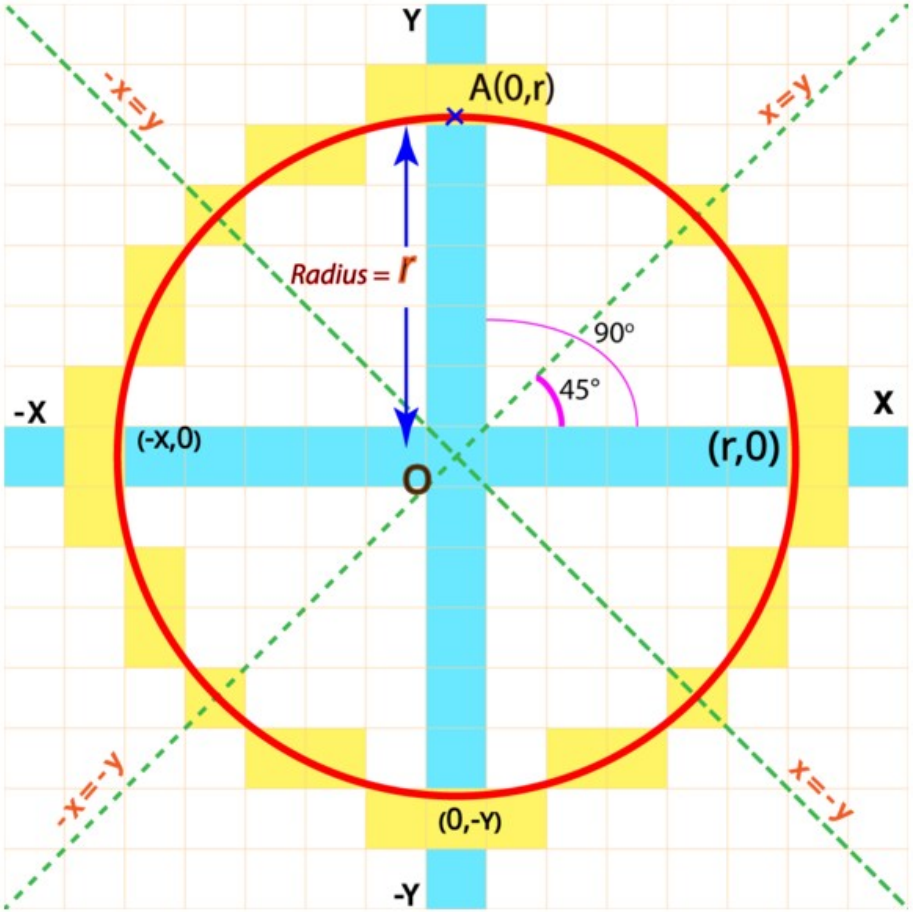
oled_clock/main.c (folytatás)

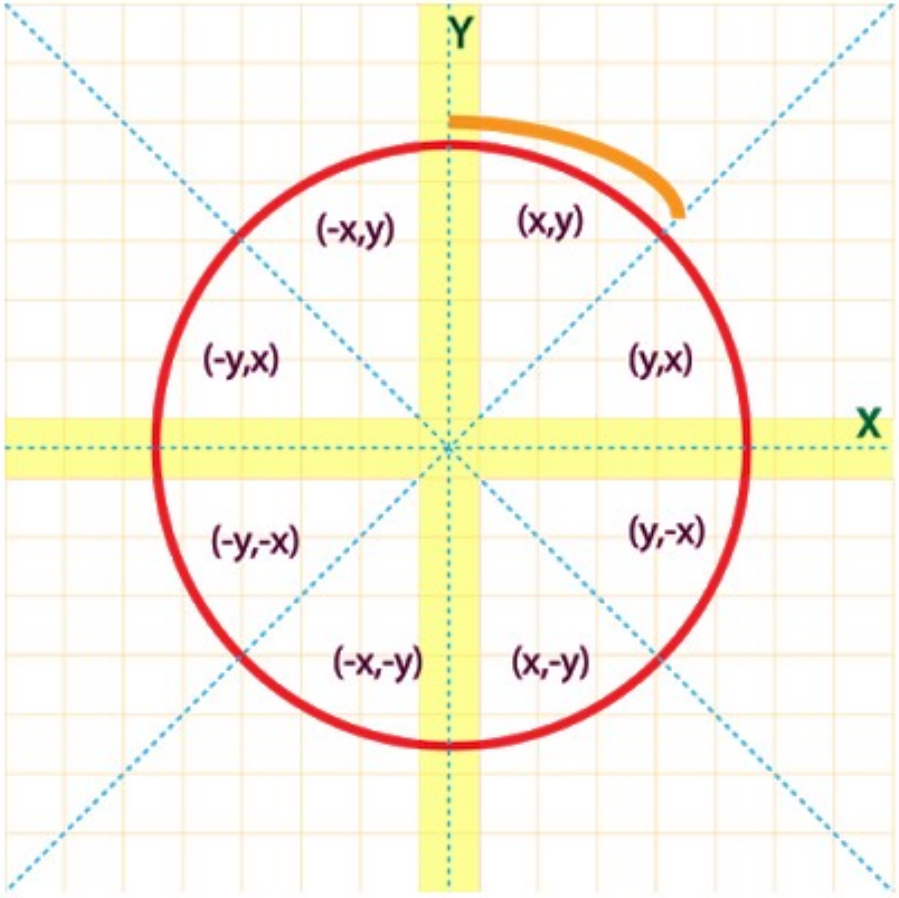
```
  //--- Az idő kiíratása HH:MM:SS formátumban ---
  sTime[0] = dec10(timeDate[2]);                    // Tízórák
  sTime[1] = dec1(timeDate[2]);                     // Órák
  sTime[2] = ':';
  sTime[3] = dec10(timeDate[1]);                    // Tízpercek
  sTime[4] = dec1(timeDate[1]);                     // Percek
  sTime[5] = ':';
  sTime[6] = dec10(timeDate[0]);                    // Tízmásodpercek
  sTime[7] = dec1(timeDate[0]);                      // másodpercek
  sTime[8] = 0;
  SSD1306_GotoXY(5, 1);
  SSD1306_Puts(sDate, &Font_7x10, SSD1306_COLOR_WHITE);
  SSD1306_GotoXY(12, 12);
  SSD1306_Puts(sTime, &Font_11x18, SSD1306_COLOR_WHITE);
  SSD1306_UpdateScreen();                           // Megjelenítés
}
```

„Majdnem" így néz ki a végeredmény ☺
(a hónapot elfelejtettem betűvel írni!)

Számítógépes grafika: kör rajzolása

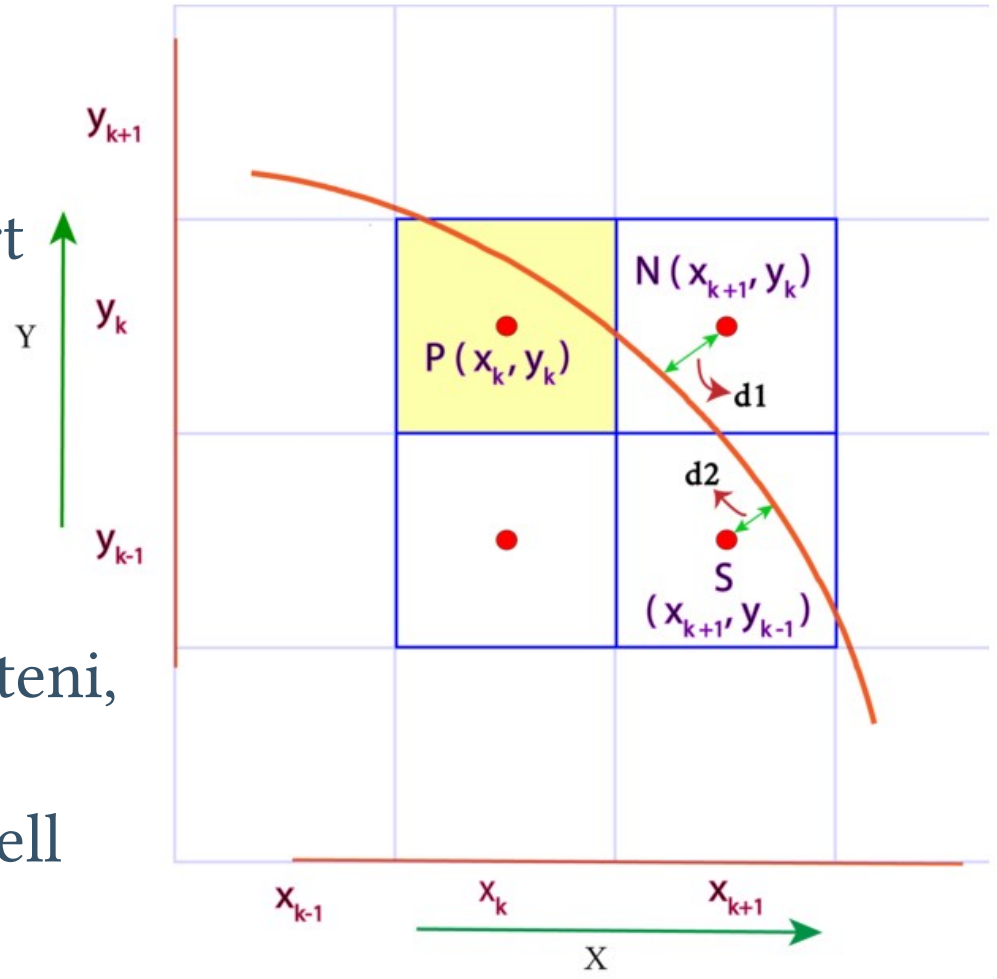
- Az $x^2 + y^2 = r^2$ egyenlettel leírható kört pixelekkel közelítjük
  Felhasznált forrás: getsetcg.com: Computer Graphics Tutorial
- A szimmetria miatt elég egy térnyolcadot kiszámolni, a többi pont koordinátái tükrözésekkel előállnak





Breshenham algoritmus

- Az algoritmus során azt kell eldönteni, hogy a $\mathbf{P}(x_k, y_k)$ pont után az $\boldsymbol{N}(x_k+1, y_k)$ vagy az $\boldsymbol{S}(x_k+1, y_k-1)$ pont van-e közelebb a körhöz
- Az $\boldsymbol{N}$ pont kívül esik a körön, ezért $\mathbf{F(N) = (x_k+1)^2 + y_k^2 - r^2}$ értéke pozitív
- Az $\boldsymbol{S}$ pont belül esik a körön, ezért $\mathbf{F(S) = (x_k+1)^2 + (y_k-1)^2 - r^2}$ értéke negatív
- **Apró trükk:** célszerűbb a $\mathbf{D_k = F(\boldsymbol{N})+F(\boldsymbol{S})}$ kifejezés értékét vizsgálni és lépésről lépésre frissíteni, ekkor $\mathbf{D_k < 0}$ esetén az $\boldsymbol{N}$ pontot, $\mathbf{D_k > 0}$ esetén pedig az $\boldsymbol{S}$ pontot kell választanunk



Az optimalizált Breshenham algoritmus

- Az egész számokkal végzett műveletekre optimalizált algoritmus csak az indulásnál végez egy szorzást
- A mellékelt **Python3** kód kilistázza a jobb felső térnyolcadba eső pixelek koordinátáit

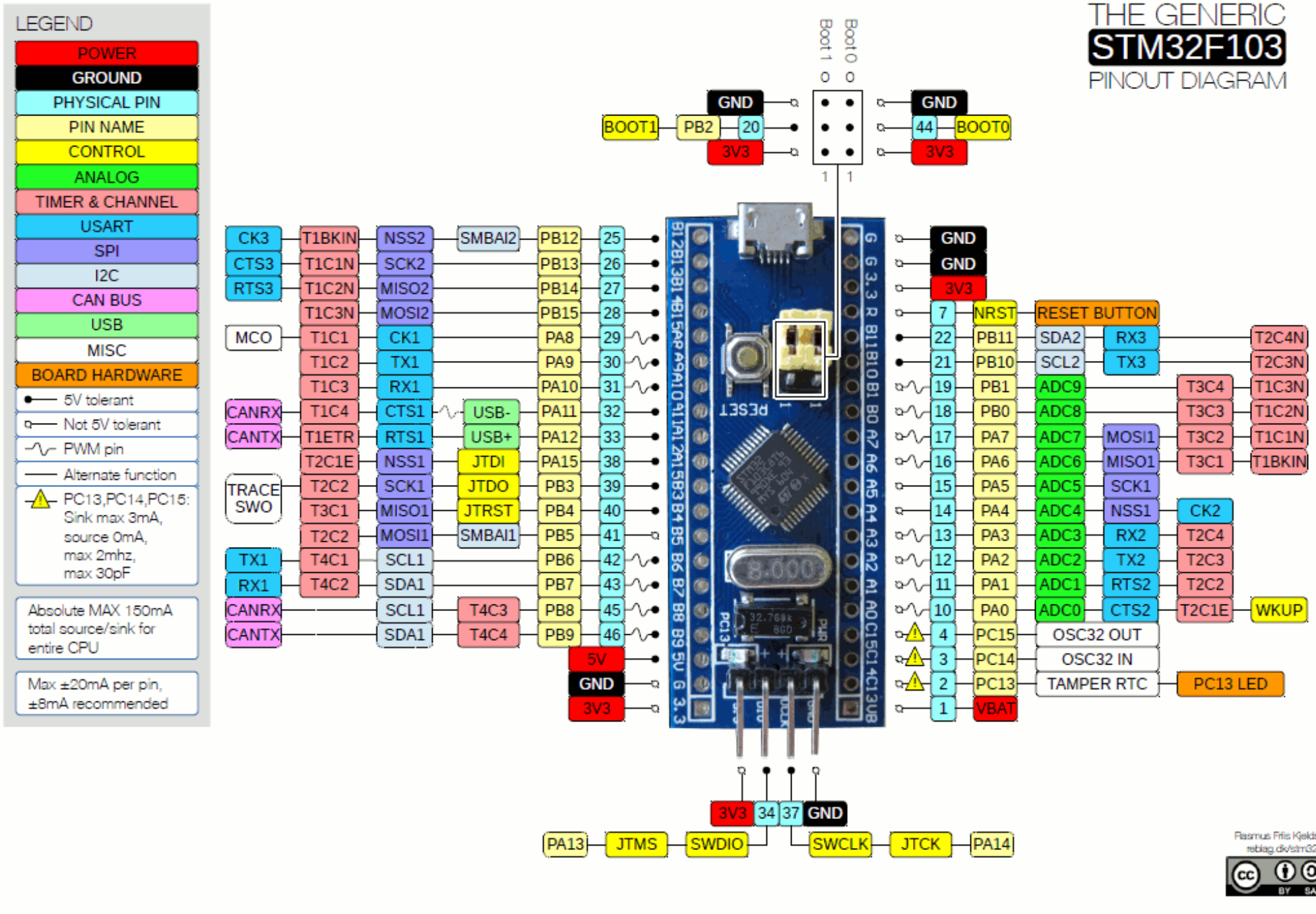
(0 , 6)
(1 , 6)
(2 , 6)
(3 , 5)
(4 , 4)

```
def bresenham(x0,y0,r):
  x = 0
  y = r
  d = 1-r
  ddx = 1
  ddy = -2*r
  while x <= y :
    print("(",x ,",",y ,")")
    if d > 0 :
      y = y - 1
      ddy += 2
      d += ddy
    x = x + 1
    ddx += 2
    d += ddx

if __name__=='__main__':
    x0 = 0
    y0 = 0
    r =  6
    bresenham(x0, y0, r)
```

Kör rajzolása az ssd1306.c programkönyvtárban

Az (x0,y0) középpontú kör minden pontját eltoljuk

```
void SSD1306_DrawCircle(int16_t x0, int16_t y0, int16_t r, SSD1306_COLOR_t c) {
    int16_t f = 1 - r;
    int16_t ddF_x = 1;
    int16_t ddF_y = -2 * r;
    int16_t x = 0;
    int16_t y = r;
    SSD1306_DrawPixel(x0, y0 + r, c);
    SSD1306_DrawPixel(x0, y0 - r, c);
    SSD1306_DrawPixel(x0 + r, y0, c);
    SSD1306_DrawPixel(x0 - r, y0, c);
    while (x < y) {
        if (f >= 0) {
            y--; ddF_y += 2; f += ddF_y;
        }
        x++; ddF_x += 2; f += ddF_x;
        SSD1306_DrawPixel(x0 + x, y0 + y, c);
        SSD1306_DrawPixel(x0 - x, y0 + y, c);
        SSD1306_DrawPixel(x0 + x, y0 - y, c);
        SSD1306_DrawPixel(x0 - x, y0 - y, c);
        SSD1306_DrawPixel(x0 + y, y0 + x, c);
        SSD1306_DrawPixel(x0 - y, y0 + x, c);
        SSD1306_DrawPixel(x0 + y, y0 - x, c);
        SSD1306_DrawPixel(x0 - y, y0 - x, c);
    }
}
```


LEGEND
POWER
GROUND
PHYSICAL PIN
PIN NAME
CONTROL
ANALOG
TIMER & CHANNEL
USART
SPI
I2C
CAN BUS
USB
MISC
BOARD HARDWARE
5V tolerant
Not 5V tolerant
PWM pin
Alternate function
PC13,PC14,PC15: Sink max 3mA, source 0mA, max 2mhz, max 30pF
Absolute MAX 150mA total source/sink for entire CPU
Max ±20mA per pin, ±8mA recommended
THE GENERIC
STM32F103
PINOUT DIAGRAM
Boot1
Boot0
GND
BOOT1
PB2
20
3V3
GND
44
BOOT0
3V3
CK3
CTS3
RTS3
MCO
CANRX
CANTX
TRACE SWO
TX1
RX1
CANRX
CANTX
T1BKIN
T1C1N
T1C2N
T1C3N
T1C1
T1C2
T1C3
T1C4
T1ETR
T2C1E
T2C2
T3C1
T2C2
T4C1
T4C2
NSS2
SCK2
MISO2
MOSI2
CK1
TX1
RX1
CTS1
RTS1
NSS1
SCK1
MISO1
MOSI1
SCL1
SDA1
SCL1
SDA1
SMBAI2
USB-
USB+
JTDI
JTDO
JTRST
SMBAI1
T4C3
T4C4
PB12
PB13
PB14
PB15
PA8
PA9
PA10
PA11
PA12
PA15
PB3
PB4
PB5
PB6
PB7
PB8
PB9
25
26
27
28
29
30
31
32
33
38
39
40
41
42
43
45
46
5V
GND
3V3
GND
GND
3V3
7
22
21
19
18
17
16
15
14
13
12
11
10
4
3
2
1
NRST
PB11
PB10
PB1
PB0
PA7
PA6
PA5
PA4
PA3
PA2
PA1
PA0
PC15
PC14
PC13
VBAT
RESET BUTTON
SDA2
SCL2
ADC9
ADC8
ADC7
ADC6
ADC5
ADC4
ADC3
ADC2
ADC1
ADC0
OSC32 OUT
OSC32 IN
TAMPER RTC
RX3
TX3
MOSI1
MISO1
SCK1
NSS1
RX2
TX2
RTS2
CTS2
T3C4
T3C3
T3C2
T3C1
CK2
T2C4
T2C3
T2C2
T2C1E
PC13 LED
T2C4N
T2C3N
T1C3N
T1C2N
T1C1N
T1BKIN
WKUP
3V3
34
37
GND
PA13
JTMS
SWDIO
SWCLK
JTCK
PA14
Rasmus Friis Kjeldsen
reblag.dk/stm32
V1.0