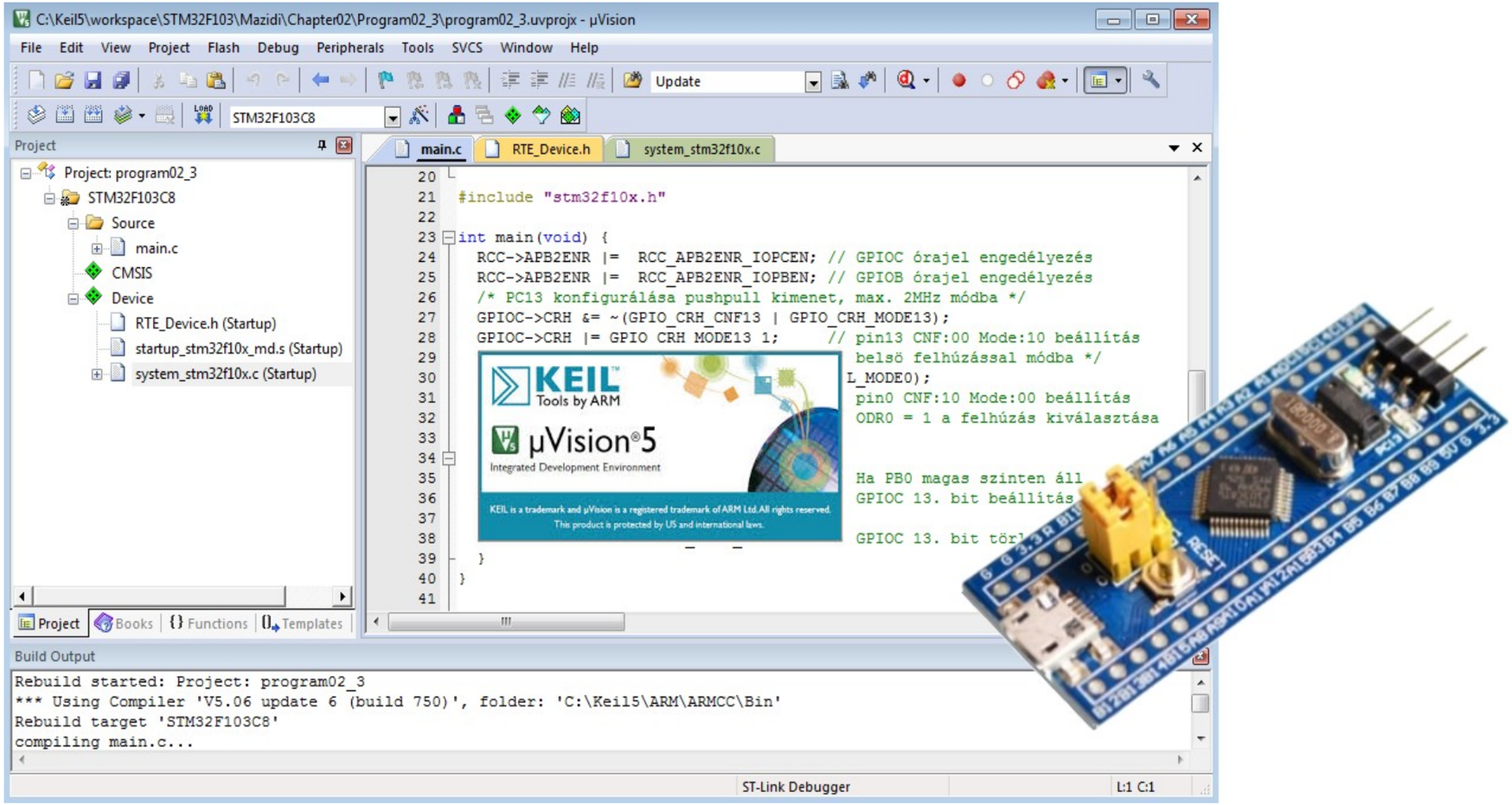
STM32 mikrovezérlők programozása ARM Keil környezetben



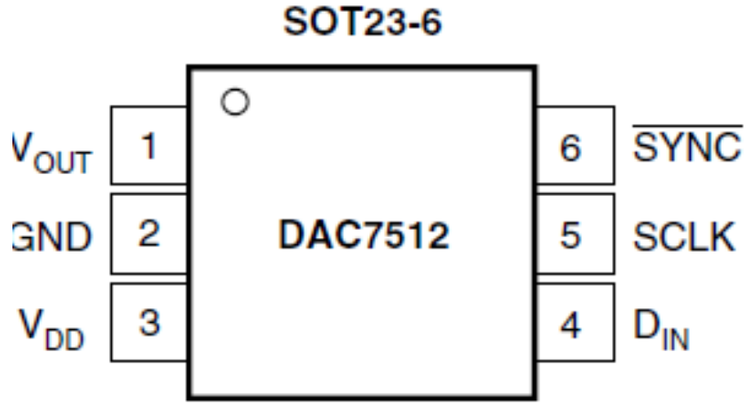
13. Soros Periféria Illesztő (SPI) – 2. rész

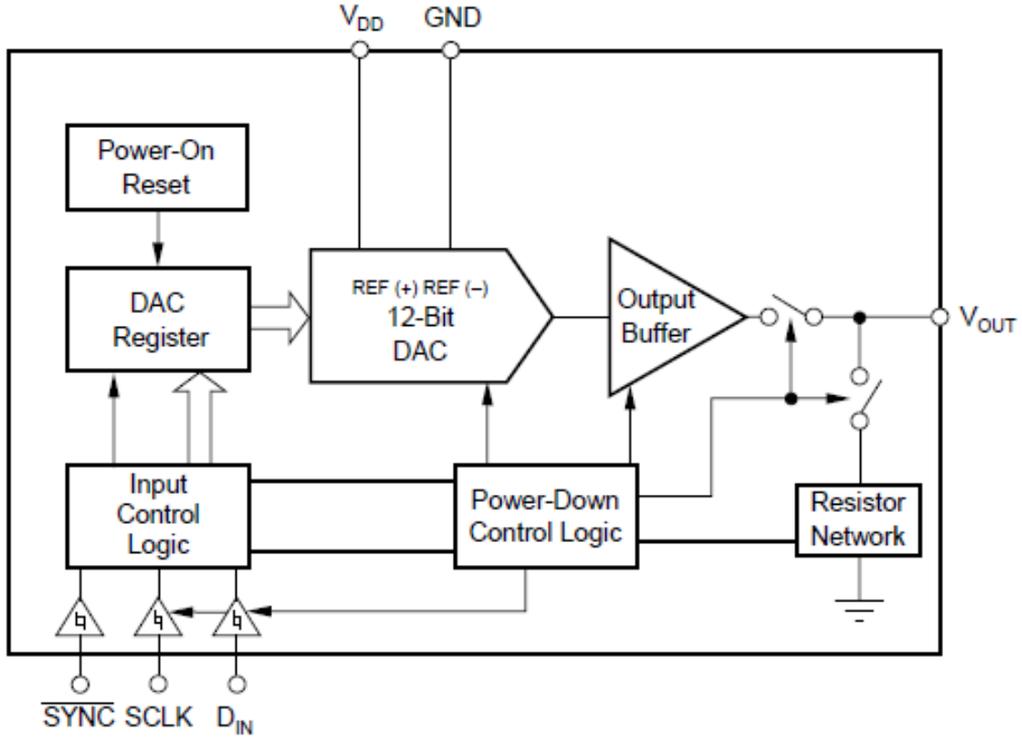
Felhasznált és ajánlott irodalom

- 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

DAC7512 digitális-analóg konverter

- A **DAC7512** típusú 12-bites digitális átalakító a **Texas Instruments** terméke
- Főbb paraméterei: 2.7 – 5.5 V tápfeszültség, kis fogyasztás (135µA @ 5V), SPI illesztő, 3 energiatakarékos mód, rail-to-rail működés, SOT23-6 és MSOP-8 tokozás
- **A kivezetések:** (SOT23-6 tokozás esetén)
  1. V_{OUT} kimenet,
  2. GND,
  3. V_{DD} tápfeszültség,
  6. $\overline{SYNC}$ vezérlőjel (active LOW)
  5. SCLK órajel (SCK)
  4. D_{IN} adat bemenet (MOSI)
- Az SPI órajel frekvenciája max. 30 MHz lehet
- A DAC 16 bitet vár, D0 – D11 az adat, D13 és D12 pedig a normál/energiatakarékos mód vezérlő bitek (**00:** normál mód , **01:** 1 kΩ, **10:** 10 kΩ, **11:** High-Z)
- Küldésnél a legnagyobb helyiérték az első (MSB first)





DAC7512 digitális-analóg konverter

- A $\overline{\text{SYNC}}$ jel alacsony állapotban engedélyezi az adatbitek beléptetését
- Az adatok rögzítése a 16. órajelnél történik
- Ha a $\overline{\text{SYNC}}$ jel korábban visszavált magas szintre, akkor megszakítja az adatfogadást

DB15															DB0
X	X	PD1	PD0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

FIGURE 3. Data Input Register.

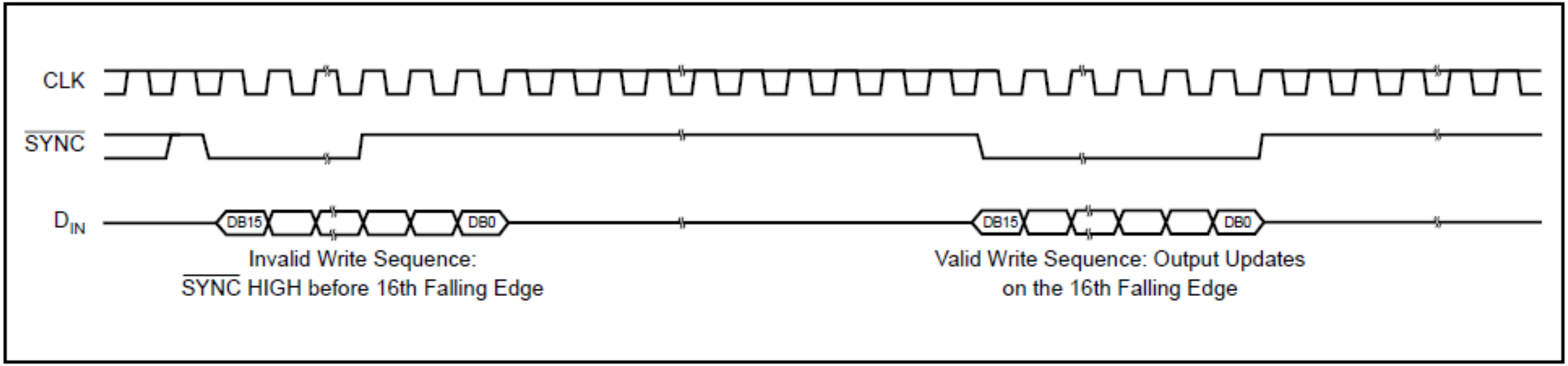
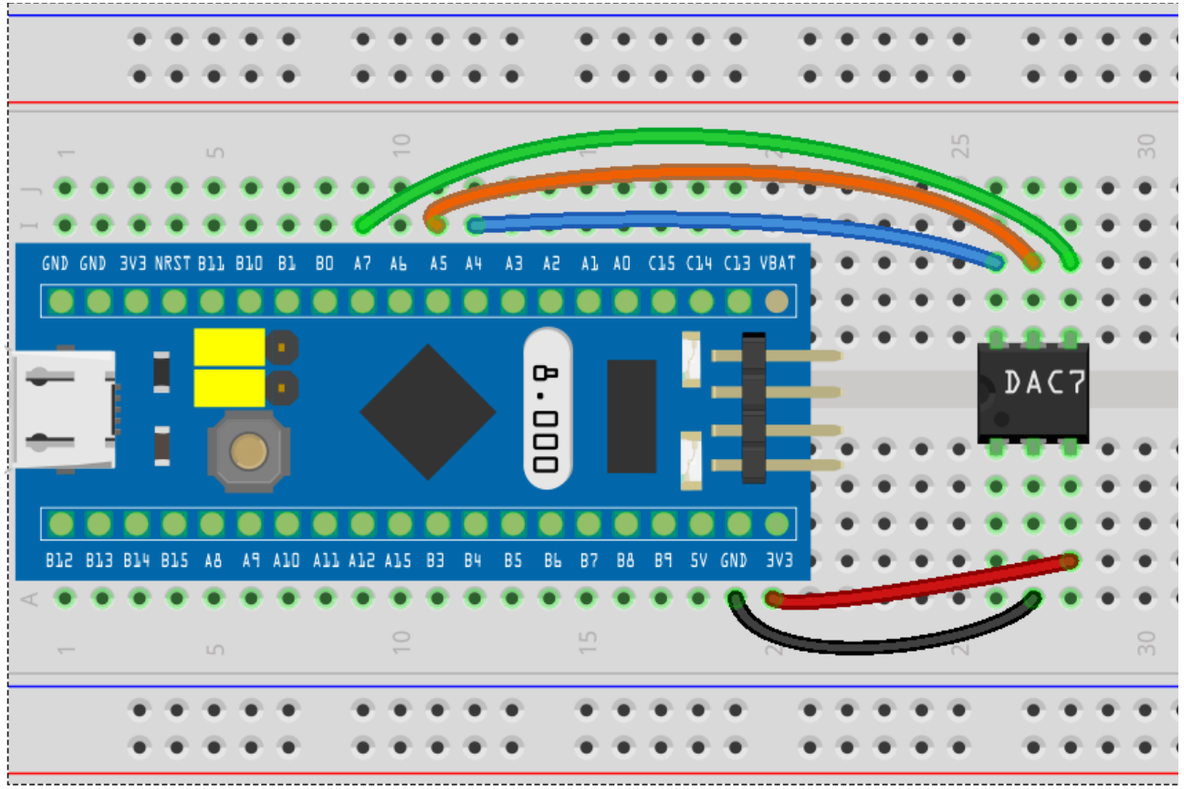
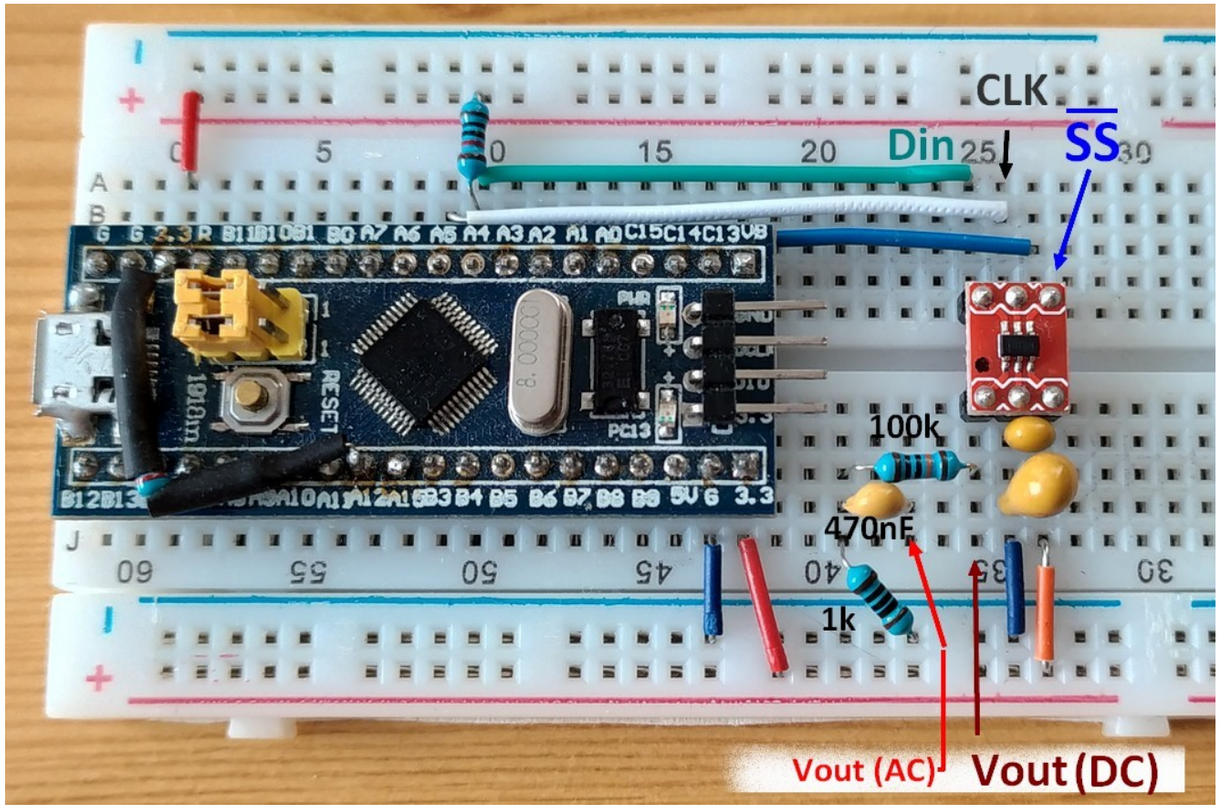


FIGURE 4. $\overline{\text{SYNC}}$ Interrupt Facility.

A kíséleti áramkör

- Az **STM32** mikrovezérlő **SPI1** csatornáját használjuk (**NSS=PA4, SCK=PA5, MOSI=PA7**)
- Az **NSS** kimenetet egy 10 kΩ-os ellenállással fel kell húzni (open drain kimenet)
- Hangkártya bemenetre feszültségosztó és leválasztó kondenzátor segítségével csatlakozhatunk (pl. 100 kΩ / 1 kΩ, 100 nF)





Az SPI1 csatorna engedélyezése és beállítása

- Az **RCC_APB2ENR** regiszterben 1-be állítjuk az SPI1 csatorna engedélyező bitjét

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
rw															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	USART 1EN	Res.	SPI1 EN	TIM1 EN	ADC2 EN	ADC1 EN	Reserved		IOPE EN	IOPD EN	IOPC EN	IOPB EN	IOPA EN	Res.	AFIO EN
	rw		rw	rw	rw	rw			rw	rw	rw	rw	rw		rw

- Az **SPI1_CR1** regiszterben 1-be állítjuk a megjelölt biteket (Master mód, 1:4 előosztás (baud rate = 72 MHz /4 = 18 MHz, 16-bites adatformátum, MSB elől, single master mód, hardveres NSS vezérlés)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BIDI MODE	BIDI OE	CRC EN	CRC NEXT	DFF	RX ONLY	SSM	SSI	*LSB FIRST*	SPE	BR [2:0]		1	MSTR	CPOL	CPHA
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

- Az **SPI1_CR2** regiszterben engedélyezni kell az NSS kimenet vezérlését

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TXEIE	RXNEIE	ERRIE	Res.	Res.	*SSOE*	TXDMAEN	RXDMAEN
								rw	rw	rw			rw	rw	rw

program08_4/main.c

```c
#include "stm32f10x.h"
void SPI1_init16(void);
void SPI1_write16(uint16_t data);

int main(void) {
  SPI1_init16();                          // Az SPI1 modul konfigurálása
  while(1) {
    for (uint16_t i=0; i<4096; i++) {
      SPI1_write16(i);                    // Adatkiírás
    }
  }
}

/*-------------------------------------------------------------------
    16 bit adat kiküldése az SPI1 csatornán
 *-------------------------------------------------------------------*/
void SPI1_write16(uint16_t data) {
  SPI1->CR1 |= SPI_CR1_SPE;               // SPI1 engedélyezés
  while (!(SPI1->SR & 2)) {}              // TXE jelre várunk
  SPI1->DR = data;                        // Adat küldése
  while (SPI1->SR & SPI_SR_BSY) {}        // Átvitel végére várunk
  SPI1->CR1 &= ~SPI_CR1_SPE;              // SPI1 letiltás
}
```

Folytatás a következő lapon...

program08_4/main.c (folytatás)

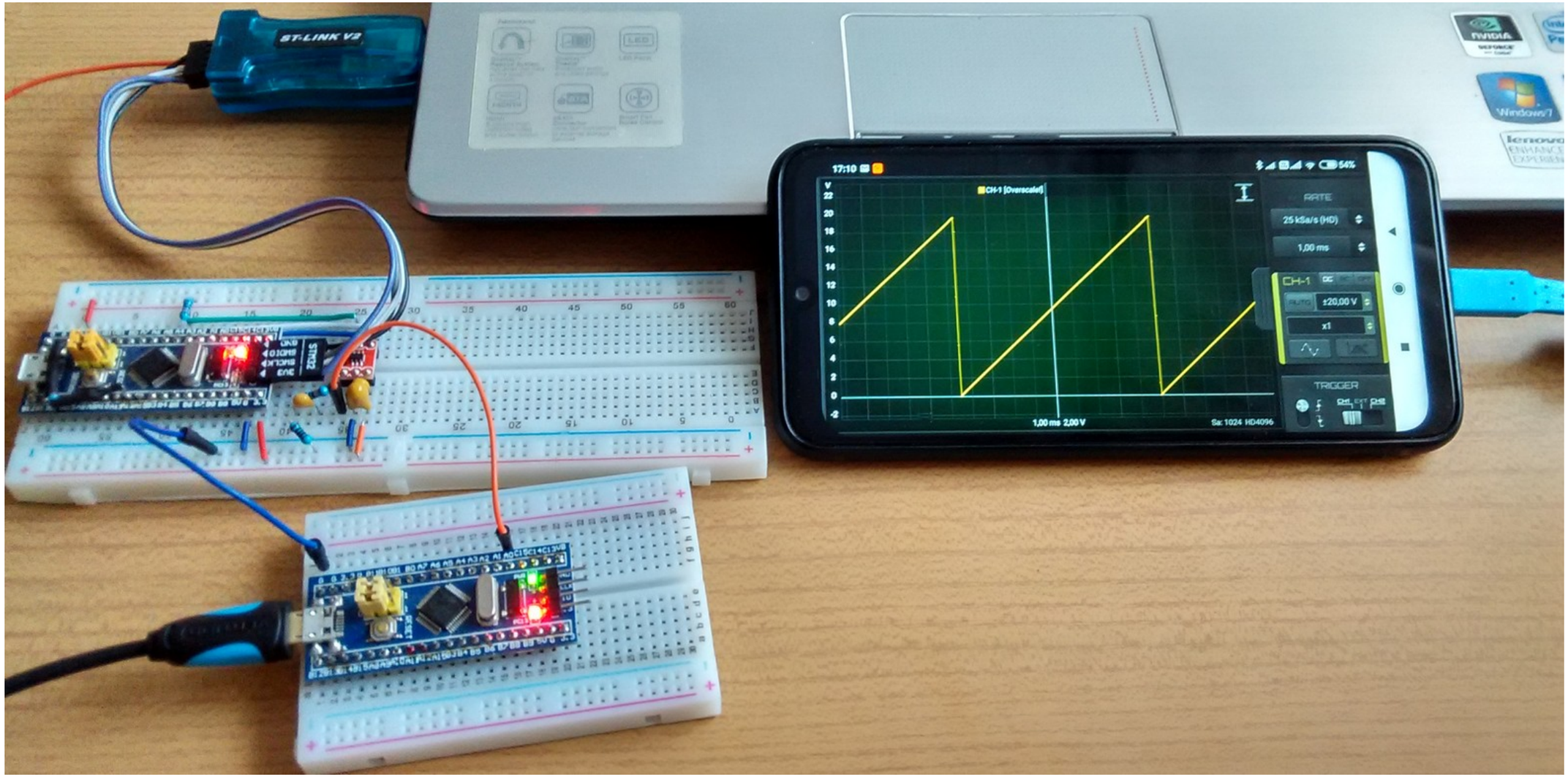
```
/*-------------------------------------------------------------------
    Az SPI1 csatorna konfigurálása 16 bites módba, 18 MHz bitrátával
    Az alapértelmezett PA4=SS, PA5=SCK, PA7=MOSI kivezetéseket használjuk
 *-------------------------------------------------------------------*/

void SPI1_init16(void) {
  RCC->APB2ENR |= RCC_APB2ENR_SPI1EN;  // SPI1 órajel engedélyezés
  RCC->APB2ENR |= RCC_APB2ENR_AFIOEN;  // AFIO órajel engedélyezés
  RCC->APB2ENR |=  RCC_APB2ENR_IOPAEN; // GPIOA órajel engedélyezés
  AFIO->MAPR &= ~AFIO_MAPR_SPI1_REMAP; // Törli SPI1 REMAP bitjét

  /* PA5, PA7 konfigurálása, mint SPI1 SCK és MOSI */
  /* PA4 konfigurálás, mint GPIO kimenet SPI1 SS-nek */
  GPIOA->CRL &= ~0xF0FF0000;           // CNF és MODE bitek törlése
  GPIOA->CRL |=  0xB0BF0000;           // CNF=10 MODE=11 (PA5,PA7)
                                       // CNF=11 MODE=11 (PA4)
  SPI1->CR1 = SPI_CR1_DFF |            // 16 bit mód
              SPI_CR1_BR_0|            // Baud rate 72/4 = 18MHz
              SPI_CR1_MSTR;            // Master mód beállítás
  SPI1->CR2 = SPI_CR2_SSOE;            // Single Master mód
  SPI1->CR1 |= SPI_CR1_SPE;            // SPI1 engedélyezés
}
```

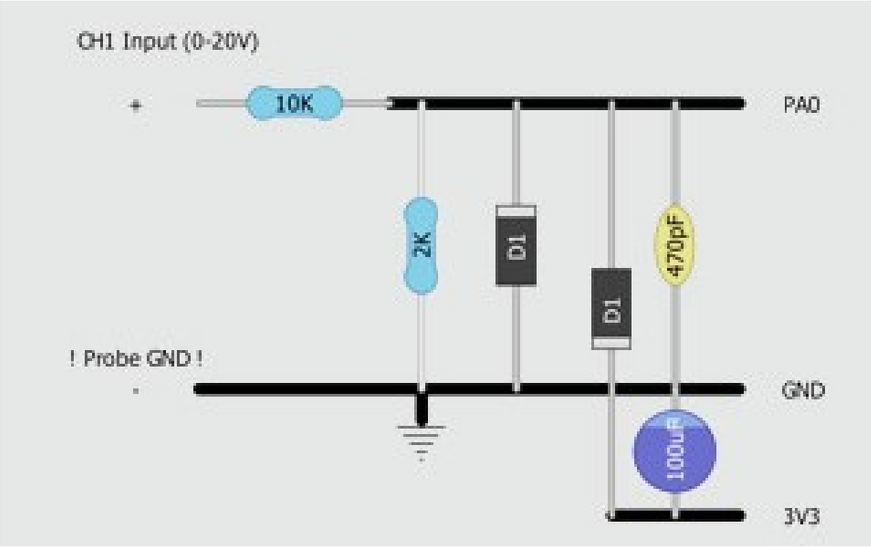
A programfutás eredményének ellenőrzése

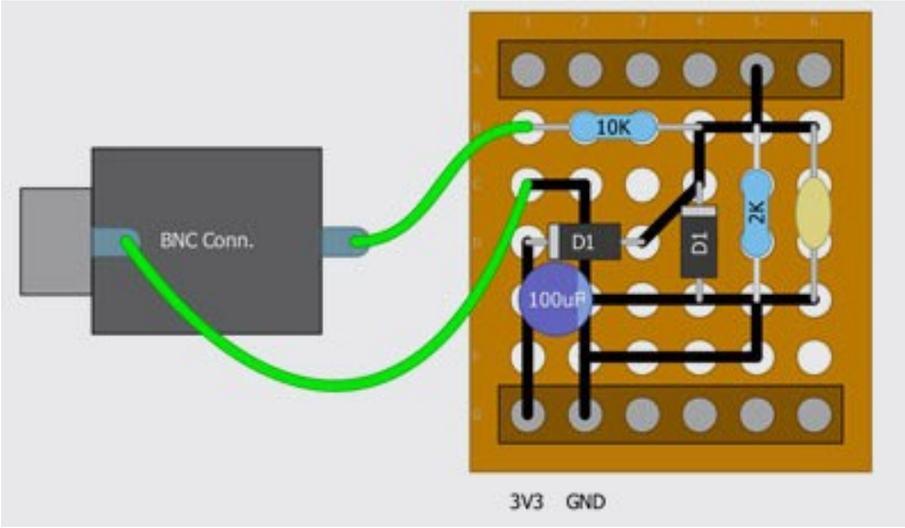
- A programmal keltett jelet egy másik **STM32** és a Hscope alkalmazás segítségével vizsgáltuk

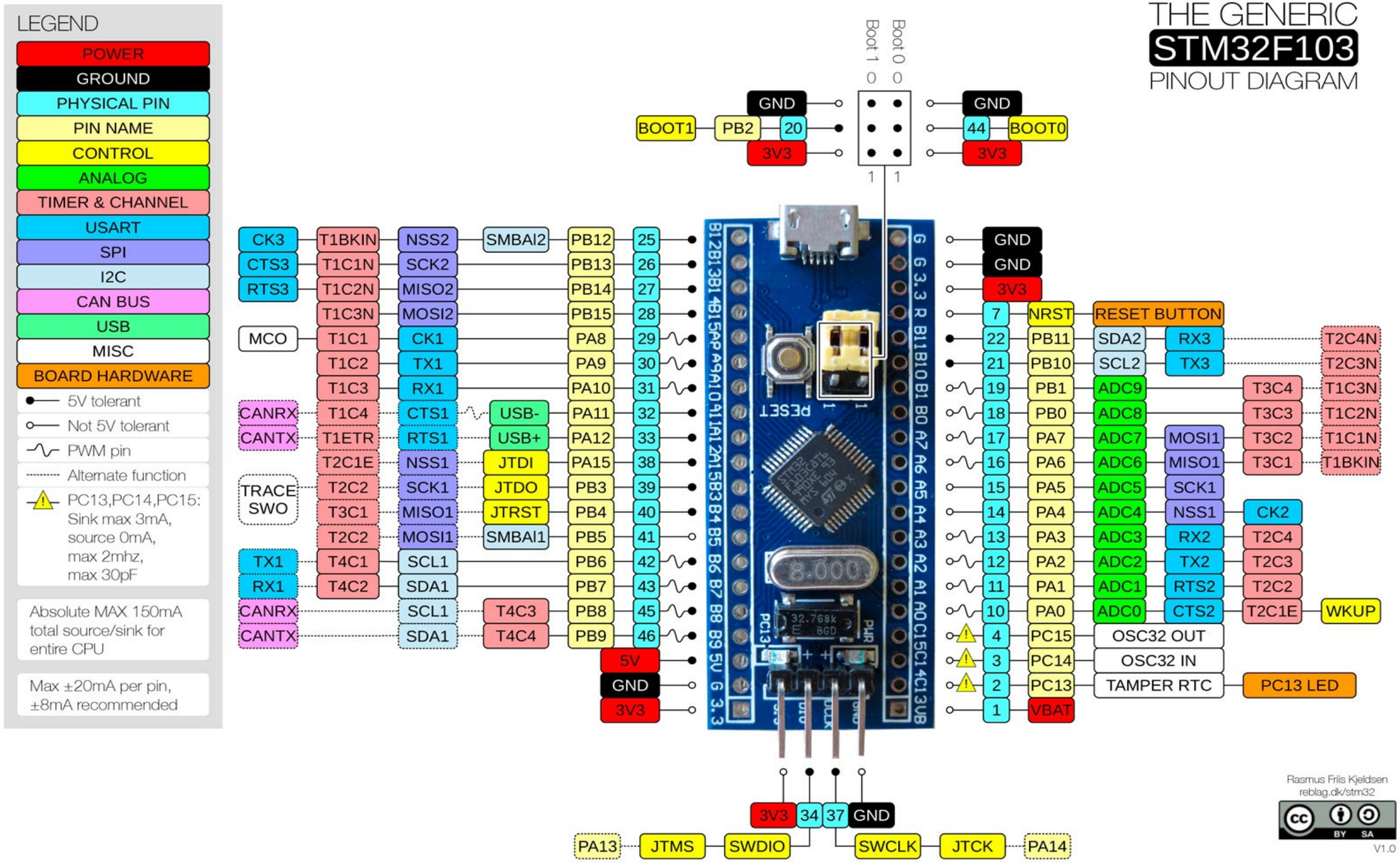


Oszcilloszkóp házilag

- Az előző oldalon bemutatott jelvizsgálóhoz két dolog kell:
  - Martin Loren Hscope nevű Android alkalmazása (ingyenes változata korlátozással)
  - Egy támogatott digitális oszcilloszkóp (vásárolt, vagy saját készítésű)
- A HS101 egycsatornás készülék a „Blue pill”
- Építési leírása: hscope.martinloren.com/HS101-oscilloscope.html
- Firmaware letöltés: github.com/martinloren/HScope/tree/master/HS 10X






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
Boot 1
Boot 0
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