ARM® mbed™

Programování kitu F0-Lab v C++ pomocí on-line IDE mbed

Kurz praktické elektroniky pro nastupující studenty ČVUT – FEL programu KyR

Náplň přednášky

a) JAK VYTVOŘIT PROGRAM PRO MIKROKONTROLER

Velmi krátké seznámení s možností programovat desku F0- LAB v C/C++ s využitím on-line IDE mbed (+ studio.keil.arm.com)

Příklady na blikání LED, použití PWM, převodníku ADC

Pro bližší seznámení – najít si učebnici C; použití API mbed – viz studium bakalářských prací

Lukáš Bielesch, Jan Kočí na:

https://embedded.fel.cvut.cz/kurzy/elektronika/informace/program

b) JAK DOSTAT *.BIN SOUBOR DO MIKROKONTROLERU

DFU, CubeProgrammer, USB a serial rozhraní, programování pomocí ST-Linku a „kopírování na flashku“

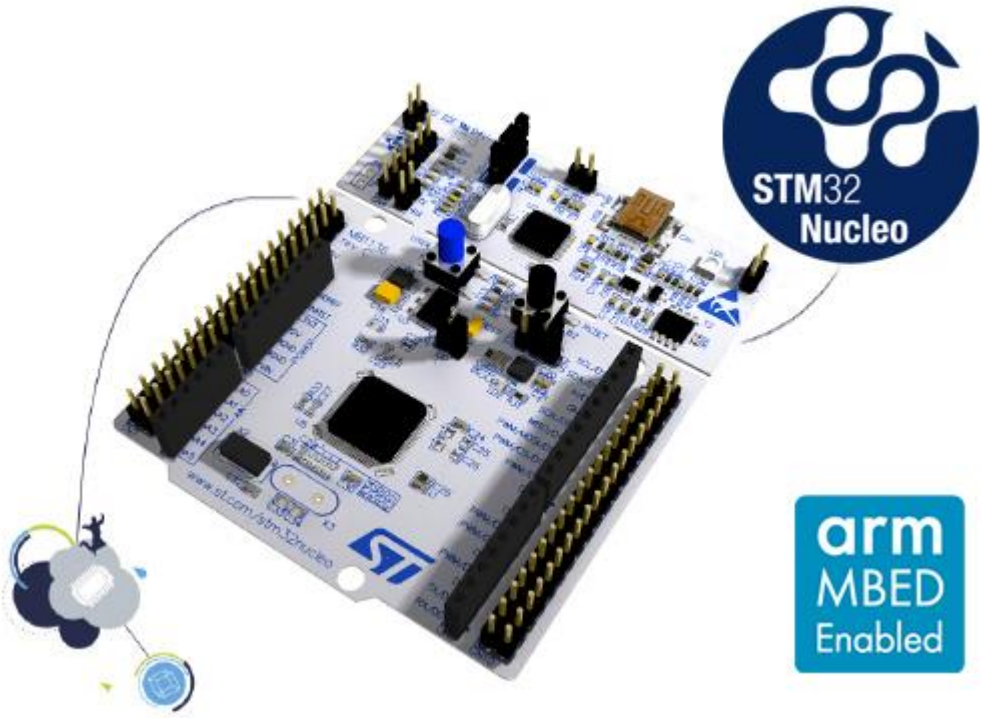
Desky STM 32 Nucleo

STM32 Nucleo desky **pro seznámení s procesory STM32**

Celá řada desek (cca 40 desek) viz **st.com/stm32nucleo**

Dotované ceny, výhodné **Nucleo** STM32F303RE (269 Kč bez DPH)

Pozn. pro Nucleo STM32F303RE – máme program LEO - osciloskop, funkční generátor, voltmetr https://embedded.fel.cvut.cz/platformy/leo

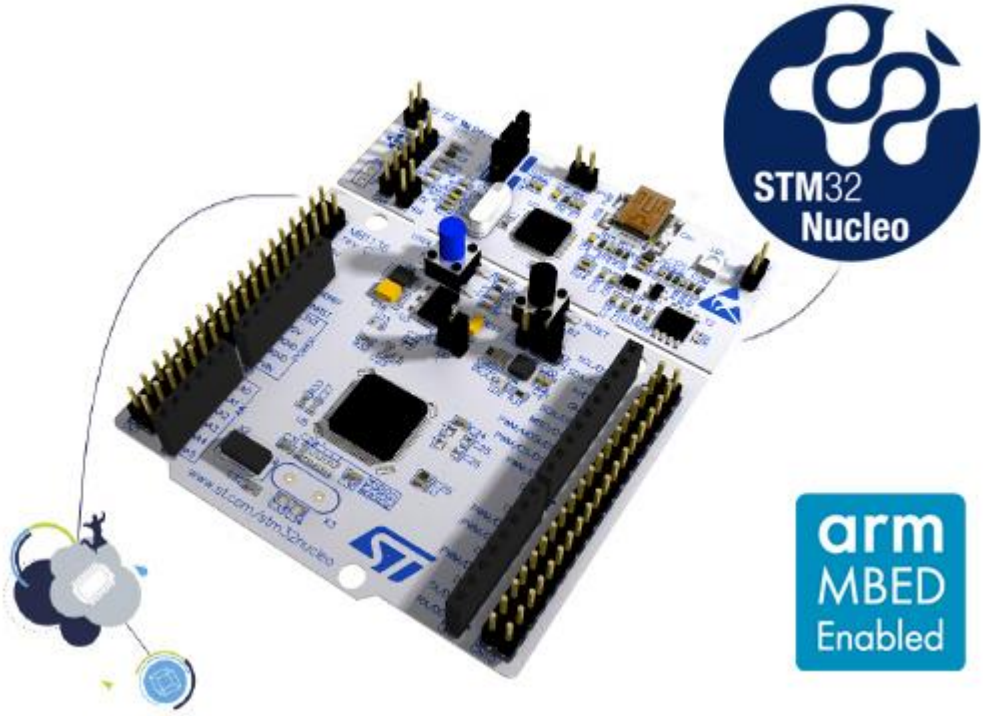


Desky STM 32 Nucleo

Možnost **programování** pomocí různých nástrojů – od asembleru až po MATLAB (výrobce doporučuje např. **STM32CubeIDE**).

Podpora programování desek Nucleo také *On-line IDE mbed*

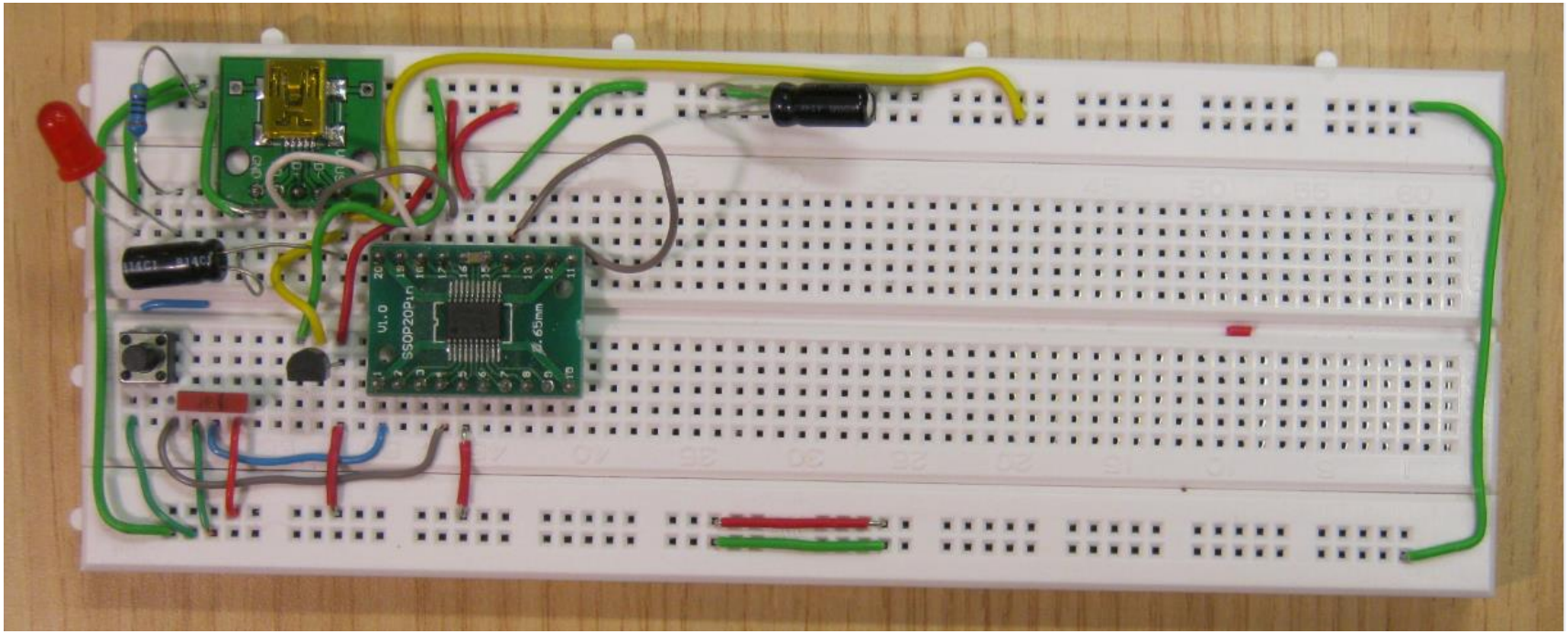
Snaha dosáhnout jednoduchosti, **jako u Arduina,** desky obsahují také konektory kompatibilní s Arduino konektory a podobné pojmenování signálů



Další použití desky F0 - Lab

Procesor firmy STMicroelectronics **STM32F042F6P6**
s jádrem ARM Cortex – M0, stejný jako v deskách ST Nucleo

Je nějaká možnost **tvorby programů** pro jednoduché nenáročné aplikace bez studia struktury procesoru (nastavování periferií,…..)?



MBED.com Deska F0- Lab a deska Nucleo ?

Na desce Nucleo-32 je procesor STM32F042K6 v pouzdře LQFP 32 s 32 vývody

Náš procesor STM32F042F6P6 v pouzdře TSSOP20 má **pouze 20 vývodů,** má však **stejně velkou paměť Flash i SRAM** i shodné periferie (vstupně výstupní brány, komunikační kanály UART, SPI, IIC Bus,..) s shodným programovým ovládáním.

Řešení – při tvorbě programu pomocí **On-line IDE mbed „předstírat,** že se tvoří **program pro STM32 Nucleo-F042,** a používat **pouze ty piny,** které **má** náš STM32F042F6P6 **v menším pouzdře.**

Závěr - využití mbed pro Nucleo F042 (nebo studio.keil.arm.com)

Co znamená **On line IDE mbed** ? **Program se tvoří (edituje a překládá)** pomocí webového prohlížeče přímo na příslušné **www stránce.** Stáhne se **výsledný binární kód**, který se nahraje do kitu Nucleo.

MBED -detailně – KyR, Bio, Ek v-1. ročníku, v předmětu ***Programování***

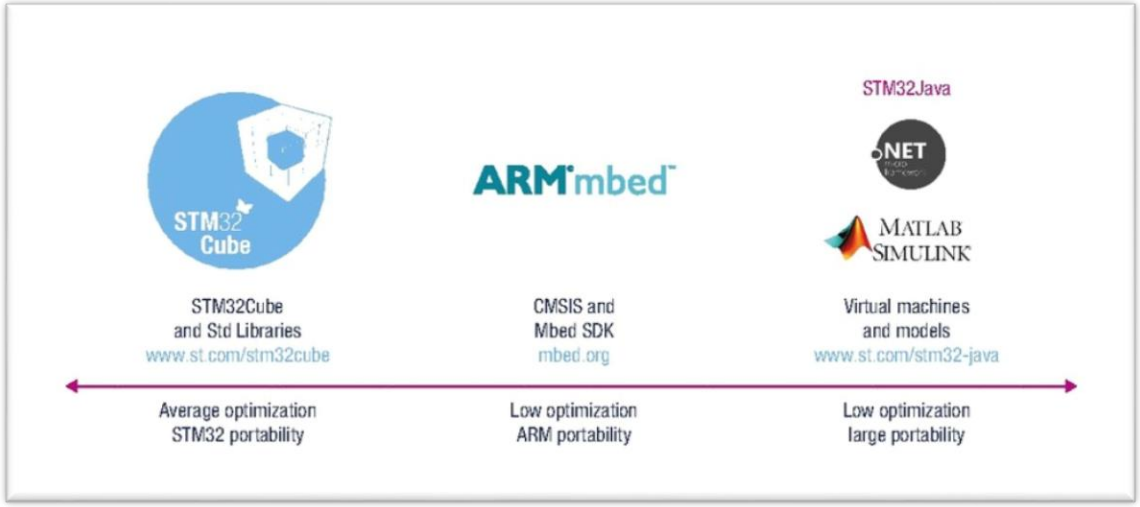
Co je to IDE MBED ?

- **Knihovna pro programování mikrokontrolérů**
- **Jazyk C++**

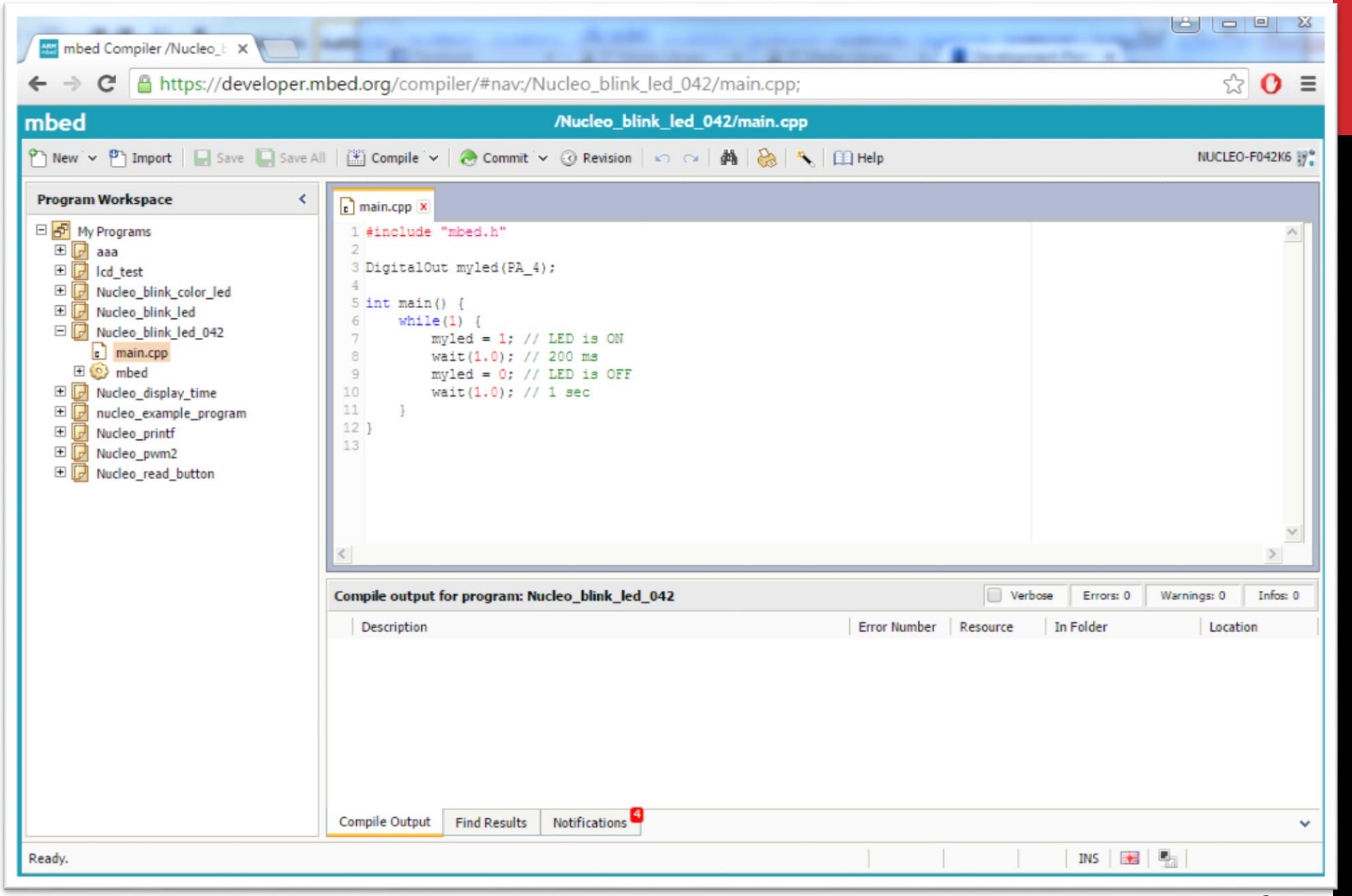
+ Jednoduché funkce dělají složité věci

- Špatná kontrola nad tím co se opravdu v MCU děje, veliký kód i pro triviální programy

Přehled podporovaných desek na : https://os.mbed.com/platforms/



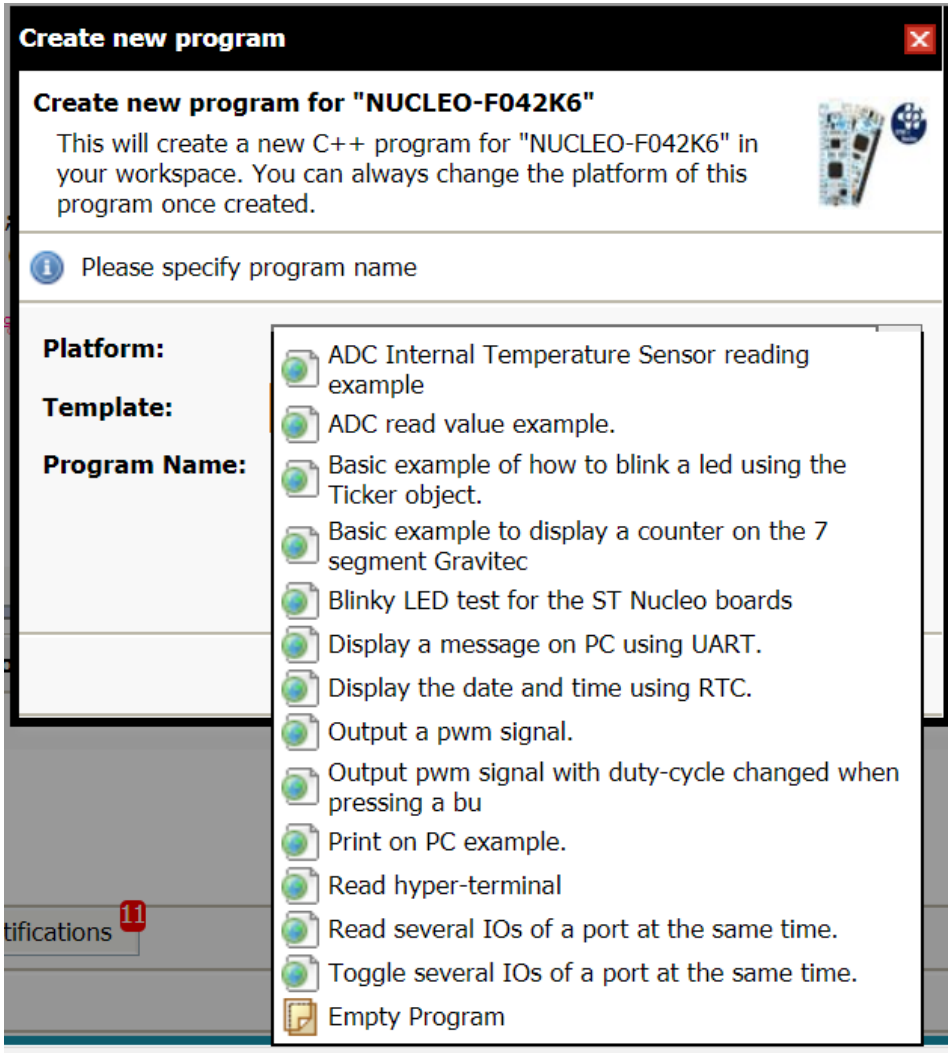
Registrovat se na mbed.com a vytvořit si účet



JAK NA TO

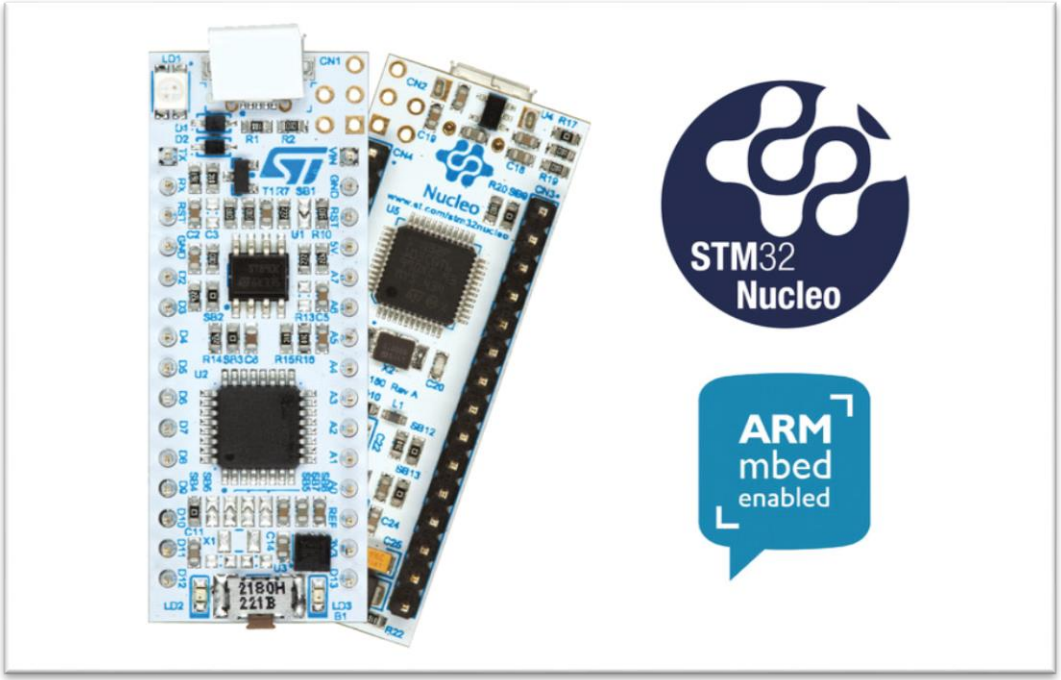
Samotné prostředí mbed poskytuje mnoho základních příkladů, ze kterých se dá vycházet

Nejdříve si do prostředí „přidat“ konkrétní HW platformu „Select a Platform“ pro ni se pak dá vybrat vzorový kód, ze kterého lze vyjít...

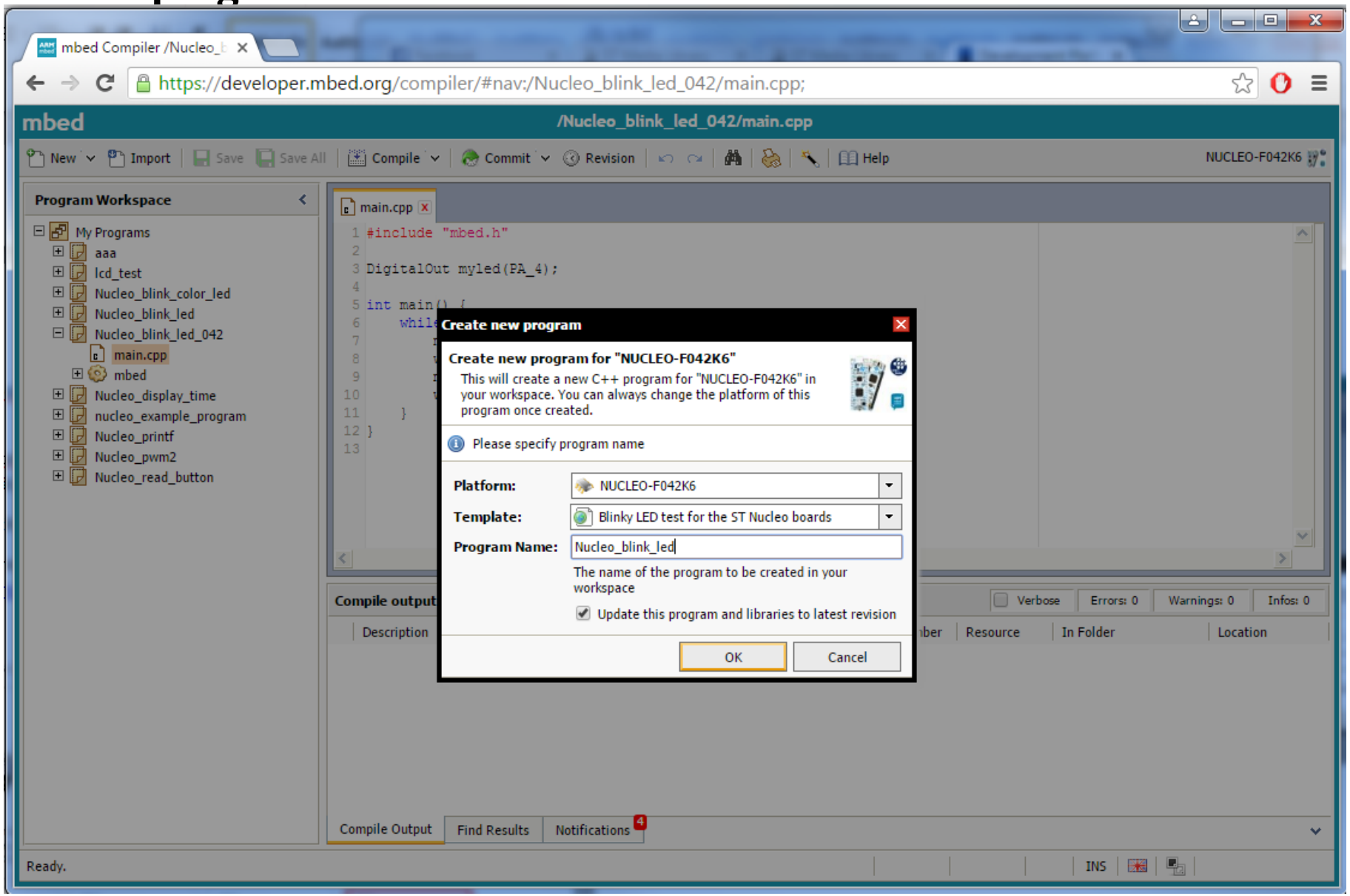


PRVNÍ PROGRAM

- **Stačí založit projekt s některou z předloh**
- **Blikání LED**
- **Předloha funguje na demonstračním kitu, s drobnou modifikací i na desce F0 - Lab**



První program



První program, blikání LED pro Nucleo F042

```
#include "mbed.h" // hlavička programu – říká, že použijeme
mbed

DigitalOut myled(LED1); // definice,kde se LED rozsvítí

int main() { // hlavní funkce programu; musí vždy existovat
    while(1) { // nekonečná smyčka;bude se stále blikat
        myled = 1; // zapni LED
        wait(0.2); // počkej 200 milisekund
        myled = 0; // zhasni LED
        wait(1.0); // 1 sec
    }
}
```

Problém, my nemáme Nucleo F042, máme LED na jiných pinech, proto je nutno piny určit jinak**.** Označení - piny **PA_4**, **PA_3**,..

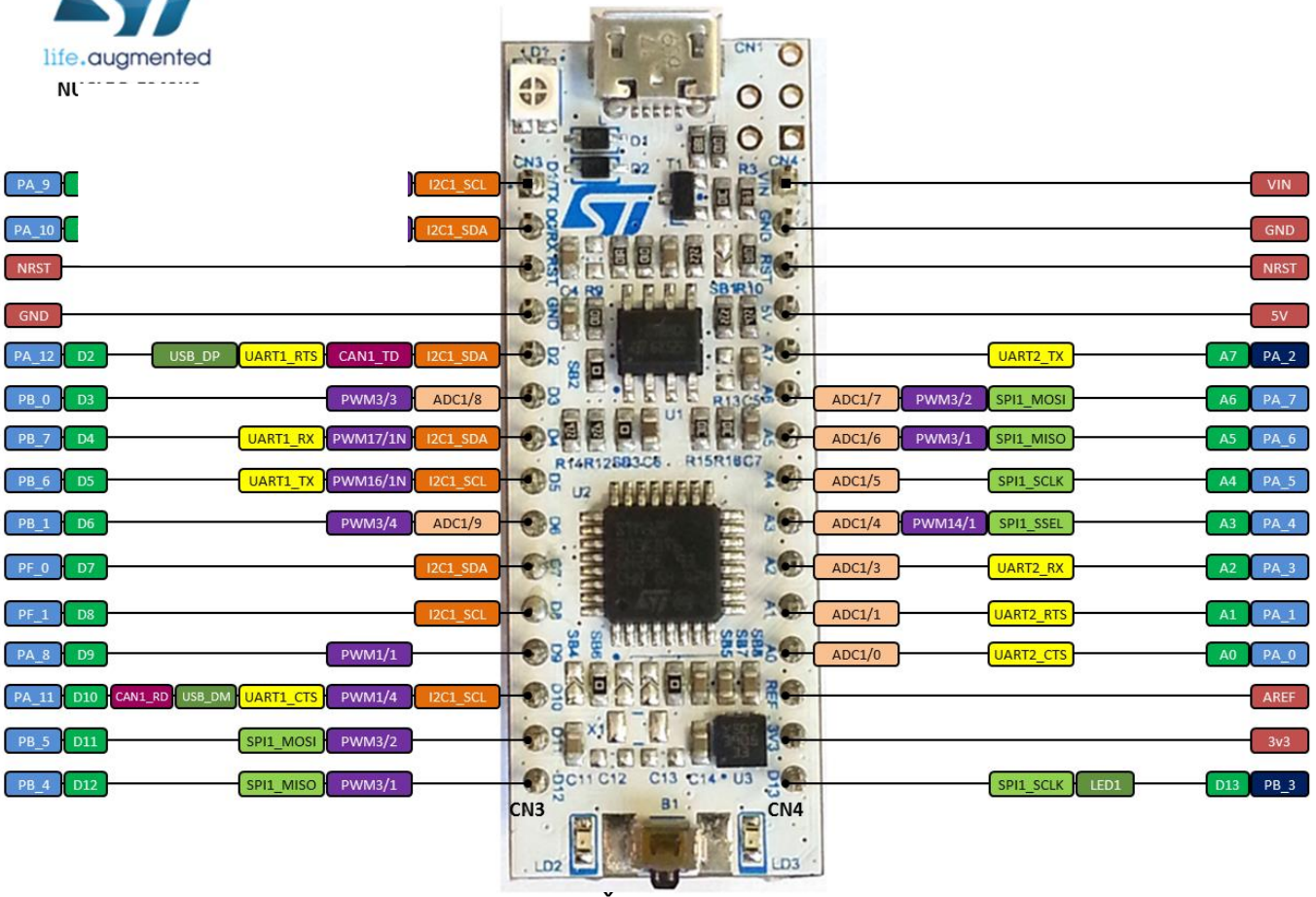
Blikání LED uprav. pro F0-Lab na kontakt. poli , pin PA_4

```
#include "mbed.h" // hlavička programu – říká, že použijeme
                                                    mbed

DigitalOut myled(PA_4); // definice,kde se LED rozsvítí

int main() { // hlavní funkce programu; musí vždy existovat
    while(1) { // nekonečná smyčka;bude se stále blikat
        myled = 1; // zapni LED
        wait(0.2); // počkej 0,2 sekundy
        myled = 0; // zhasni LED
        wait(1.0); // 1 sec
    }
}
```

My máme LED na PA_4 , což je pin číslo 10

Piny a jejich označení v mbed

- **Samotný mikrokontrolér definuje výstupní piny na několika bránách - P(A/B/C)_0-15**
- **Kity Nucleo mají *Arduino headery* značené D_0-x**
- **mbed zavádí své označení pinů, od digitálních Dx, analogových Ax, až po samotné periferie (např.PWM3/2), nebo definice tlačítek (user button) a LED (LED1)**

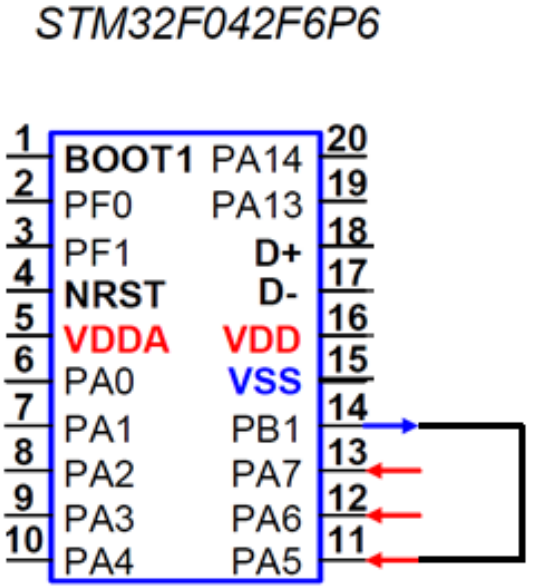


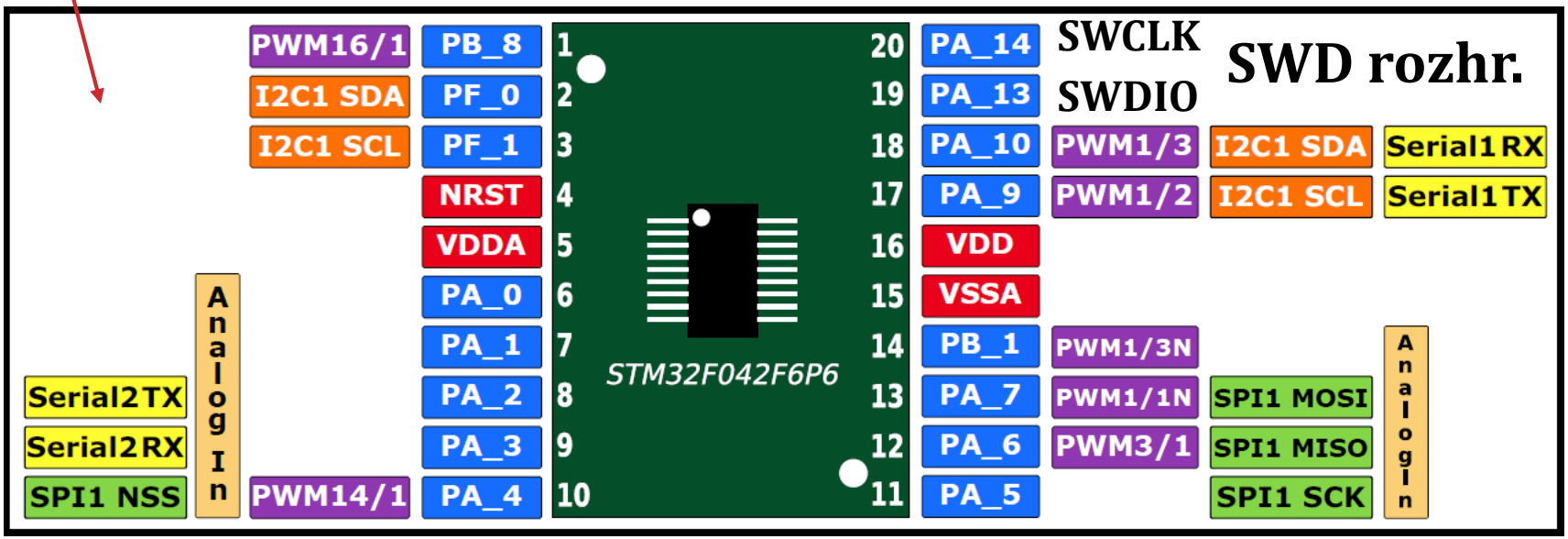
Piny a jejich označení v F0-Lab s STM32F042F6P6

Na **F0-Lab** je k dispozii **méně pinů, než** je na **Nucleo F042.**
budeme používat **označení PA_4, ...PA_0, ...**

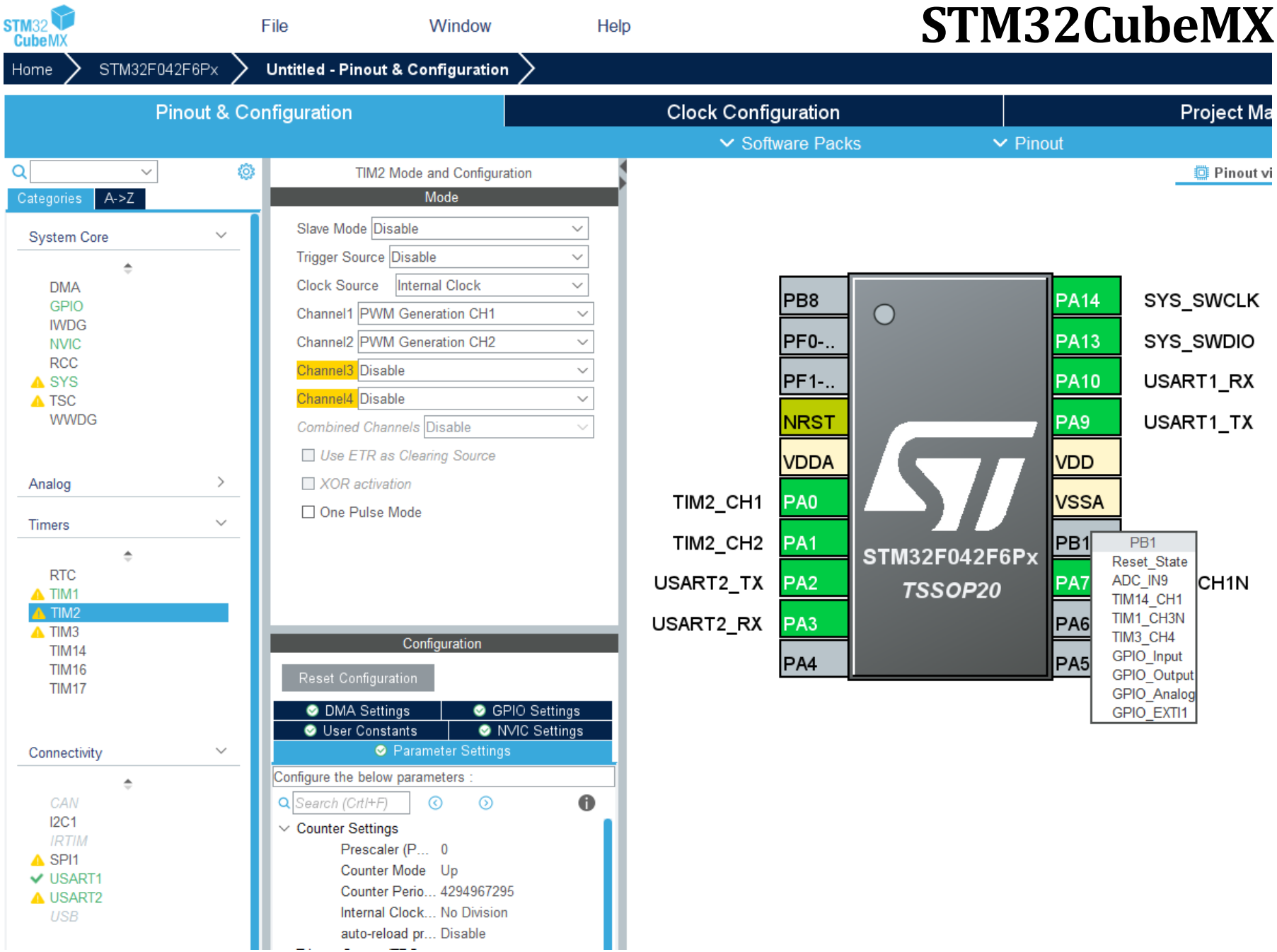
Korigovaný přehled API je k dispozici u STM32F042F6P6,
viz. BP. L. Bielesch, ČVUT- FEL, 2019

Jak to zjistíte? RTFM! nebo spustit CubeMX...





STM32CubeMX

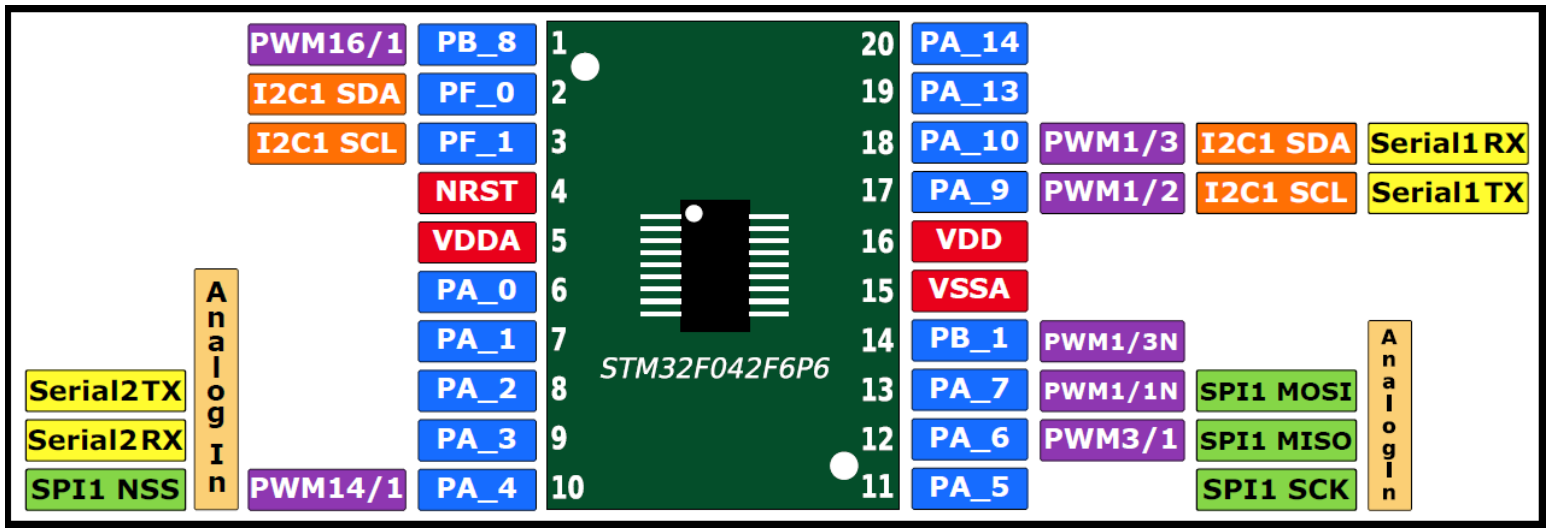


Korekce chyb mbed a upřesnění informací k PWM u F042

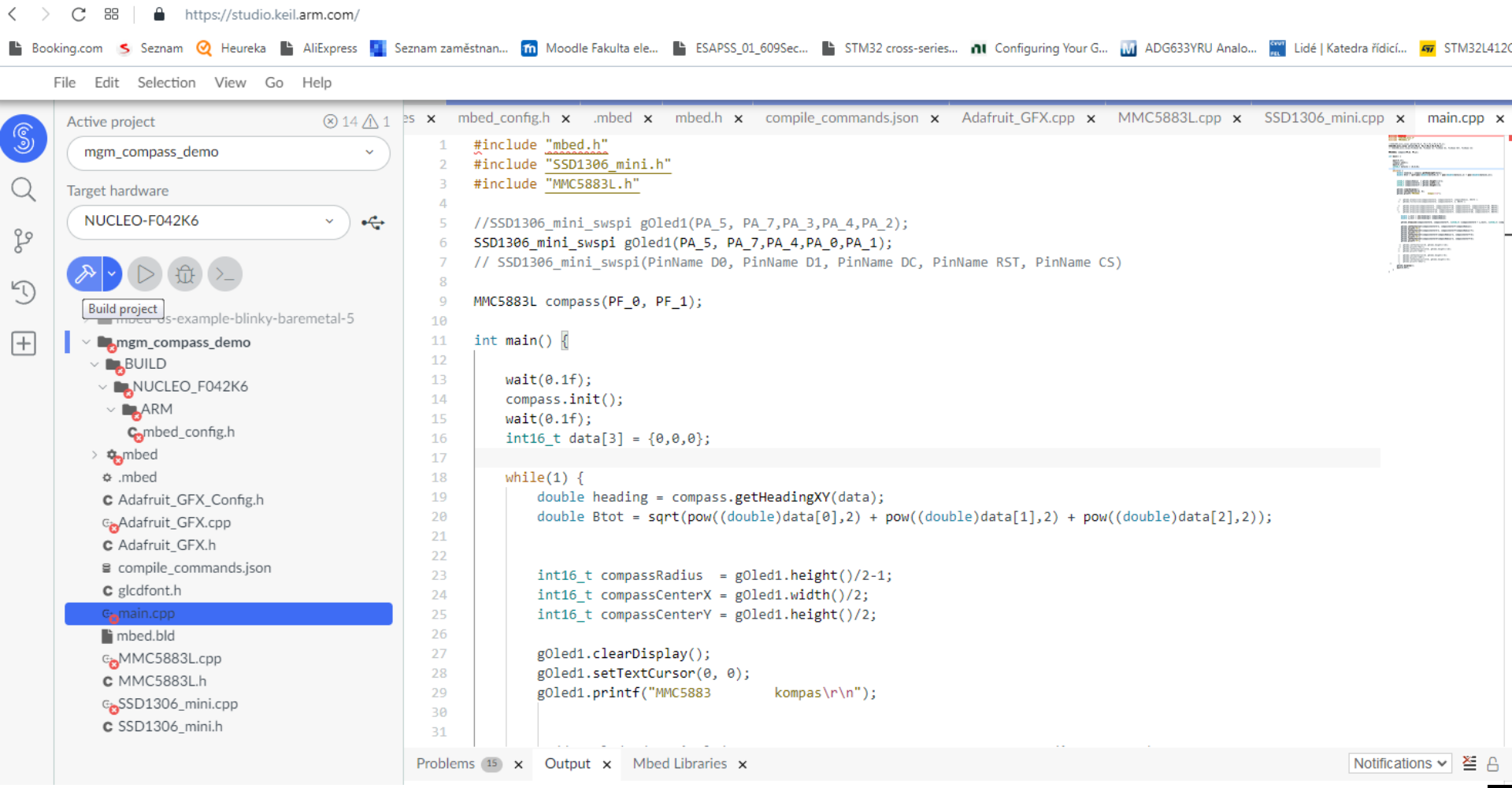
PWM 1/3. PWM1/2, PWM1N lze použít **pouze se shodně** (společně) nastavenou **periodou**, avšak **lze** použít **různou střídu**

PWM14/1 používá Timer 14 kanál1, PWM16/1 – Timer 16, kan. 1;
PWM3/1 – Timer 3, kan.1
PWM1/3 –Timer 1, kan.3; PWM1/2 stejný Timer1, ale kanál 2

Nelze použít současně PWM1/3 a PWM1/3N (pouze jeden z obou), protože jsou založeny na stejném Timeru 1 a kanálu 3; pouze používají různé výstupy (přímý nebo negovaný). Detaily- viz BP L. Bielesch



https://studio.keil.arm.com/



Užitečné třídy v mbedu (popis a použití na netu...)

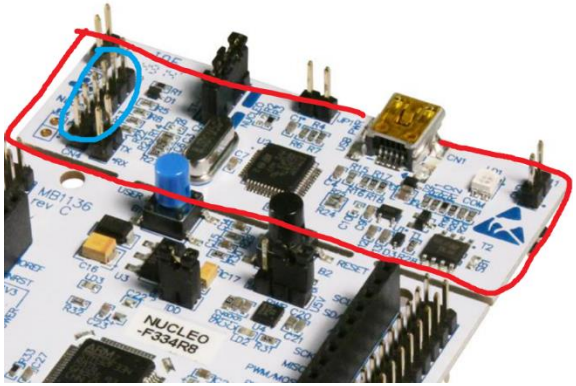
AnalogIn - měření napětí, AD převod
DigitalIn - čtení stavu vstupu (0-1)
DigitalOut - nastavení digitálního výstupu (0-1 => 0V – 3.3V)
DigitalInOut - digitální vstup/výstup - lze měnit za běhu programu
BusOut - sdružení více vývodů do jednoho log. celku
PwmOut - generování Pulzně Šířkové Modulace
InterruptIn - přerušení od externího vstupu (například stisk tlačítka)
Timer - měření časových intervalů (us až sek.)
Ticker - opakované volání funkce v pravidelném intervalu
wait - čekání
Serial - sériová komunikace (přes RXD, TXD)
USBSerial - sériová komunikace přes rozhraní USB

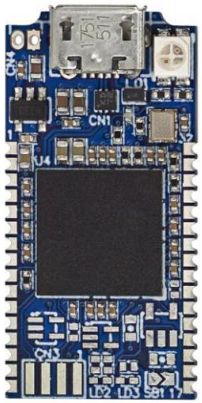
SPI – sériová komunikace na sběrnici SPI (např. řízení displejů)
I2C – sériová komunikace přes sběrnici I2C (např. čtení dat ze senzorů)
RTOS – více vláknové aplikace (již dost komplikované) – MCU je fyzicky jednovl.

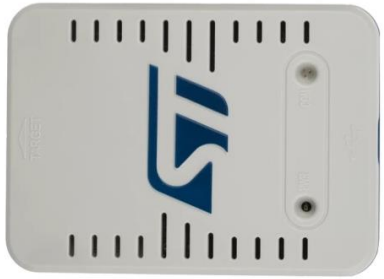
JAK DOSTAT *.BIN SOUBOR DO MIKROKONTROLERU

- **DFuse demo** (otravné – nutná mezioperace - konverze z ***.bin** do ***.dfu**)
- **CubeProgrammer** (lepší, ale občas problém na některých PC s Javou-USB, připojení USB, SWD, Serial)
- **STlink utility** (SWD připojení)
- **„kopírování na Flashdisk“** – připojení přes SWD a STlink v2.1 vytvoří virtuální disk na který lze nakopírovat *.bin soubor bez dalších SW nástrojů

STlink...





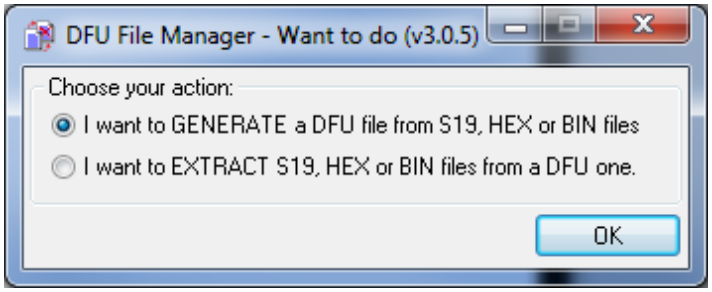


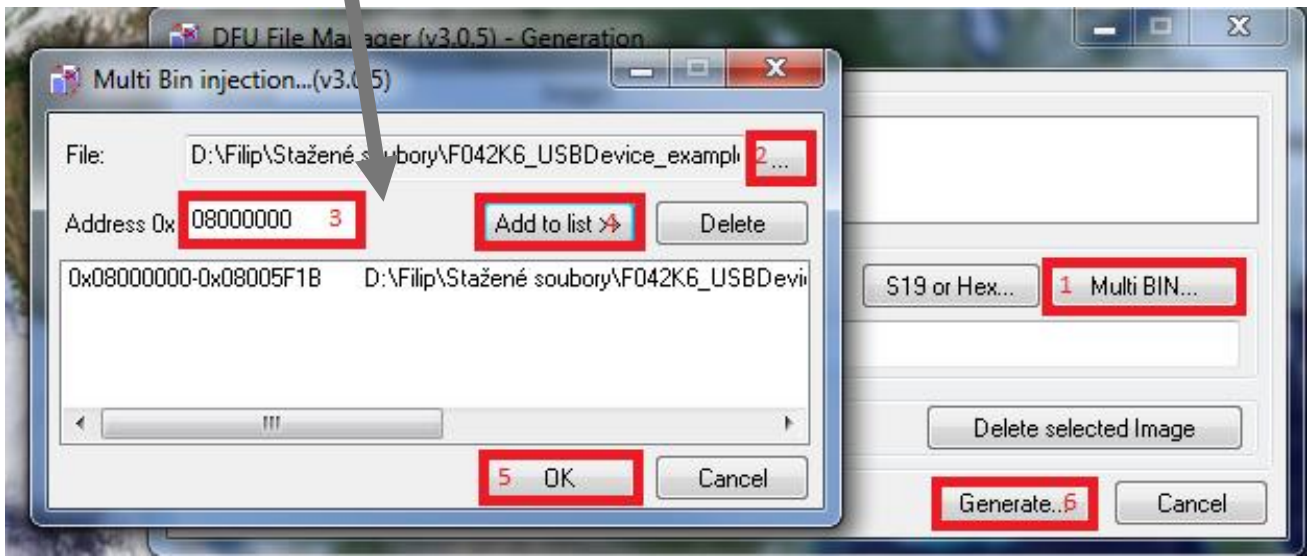
DFuse demo

- Pro desku F0 - Lab je třeba použít **DfuSe – třeba soubor.dfu**

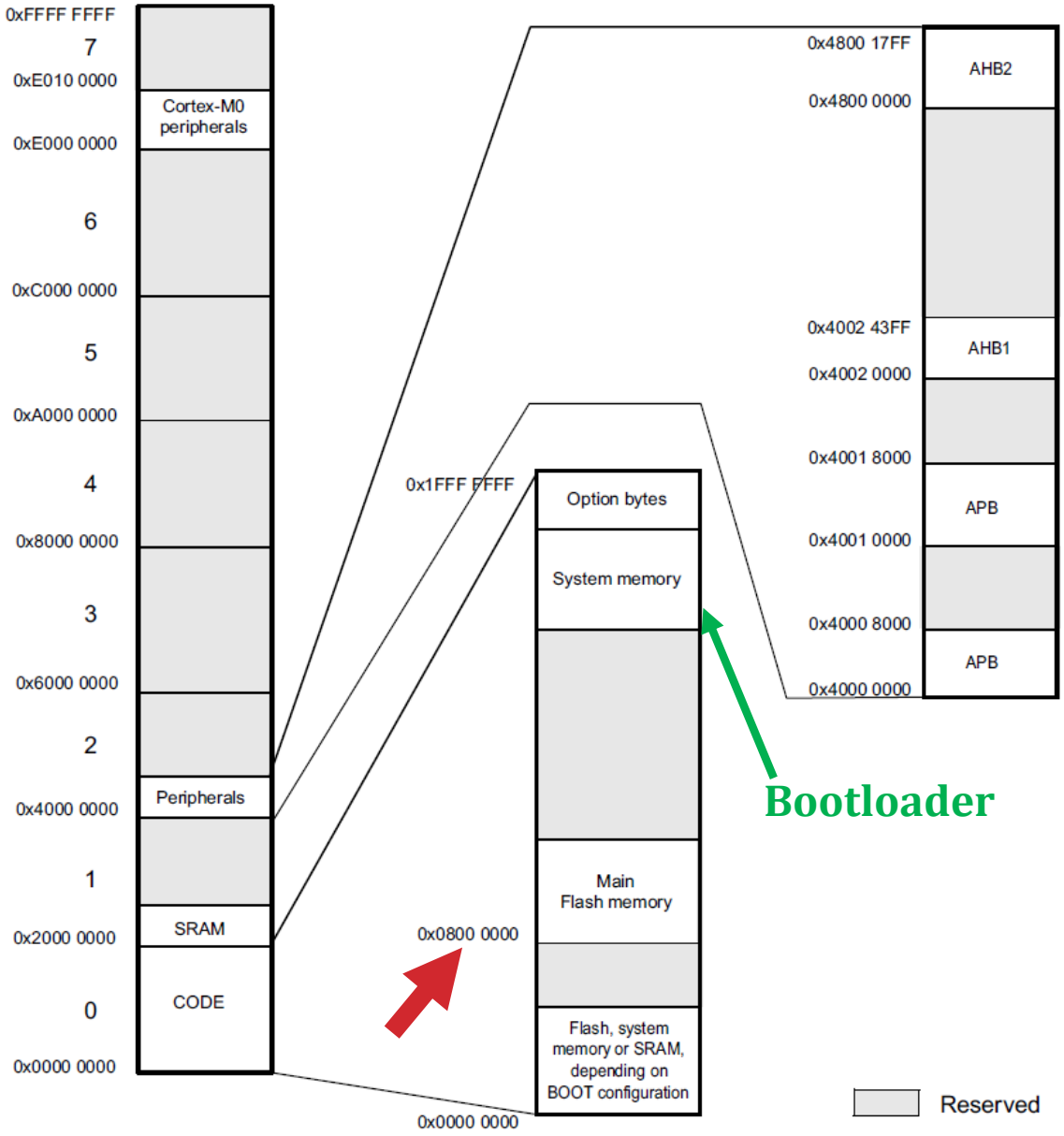
Vygenerování ***xxx.dfu*** z ***xxx.bin*** pomocí **DFU manager** (ten se nainstaloval spolu s DfUSE Demo)

nutno zadat adresu kam se má program naflashovat: 0x08000000...

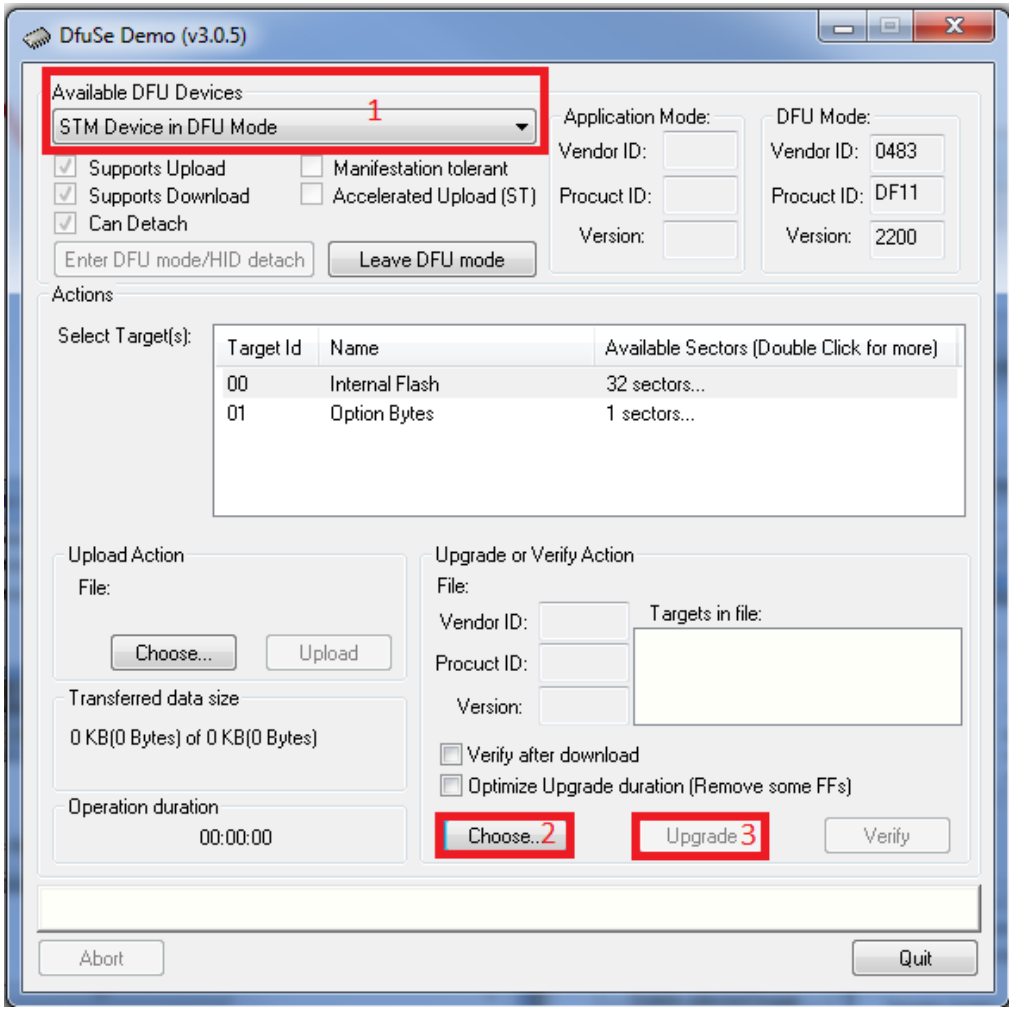




MAPA PAMĚTI STM32F042

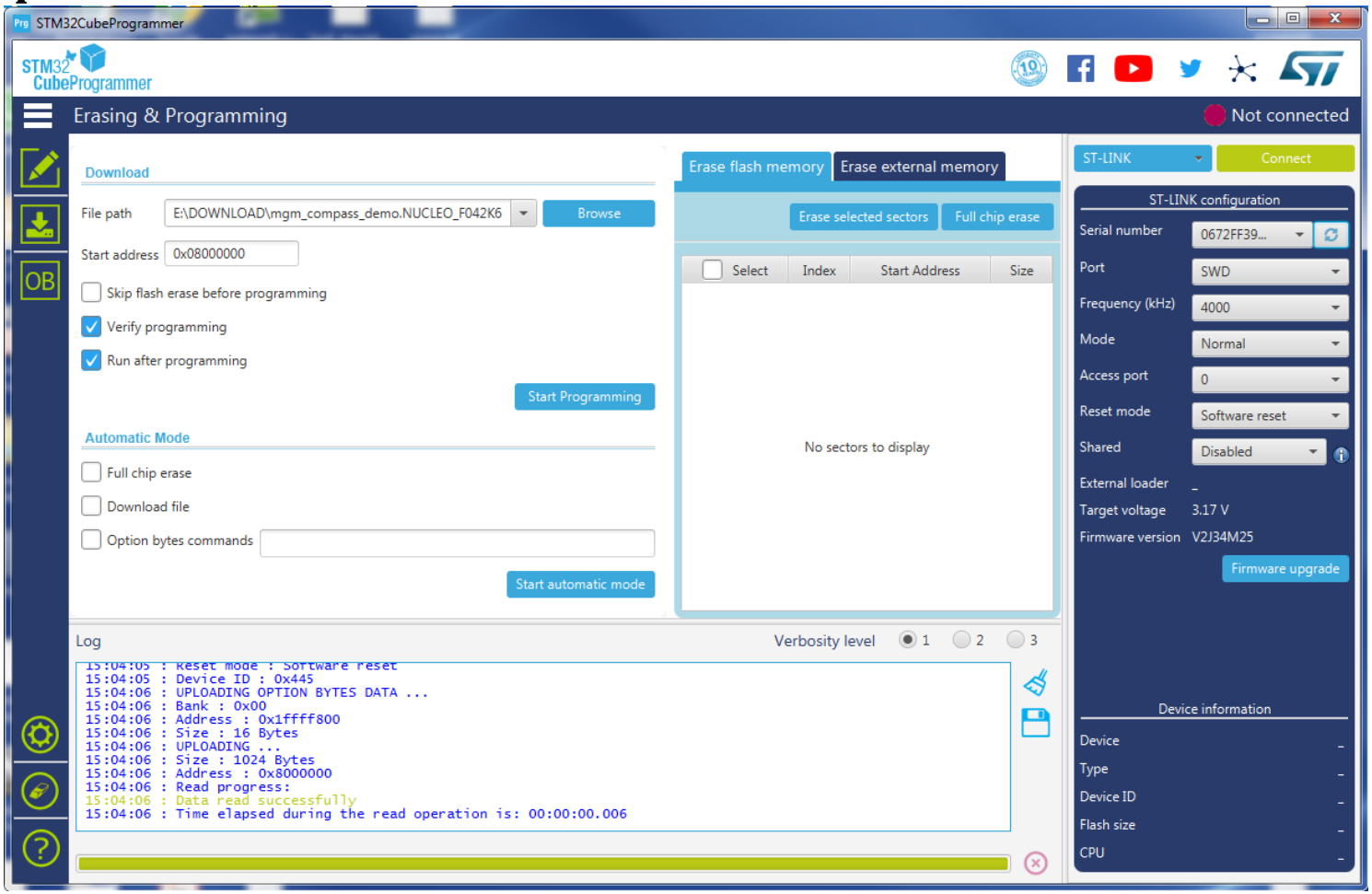


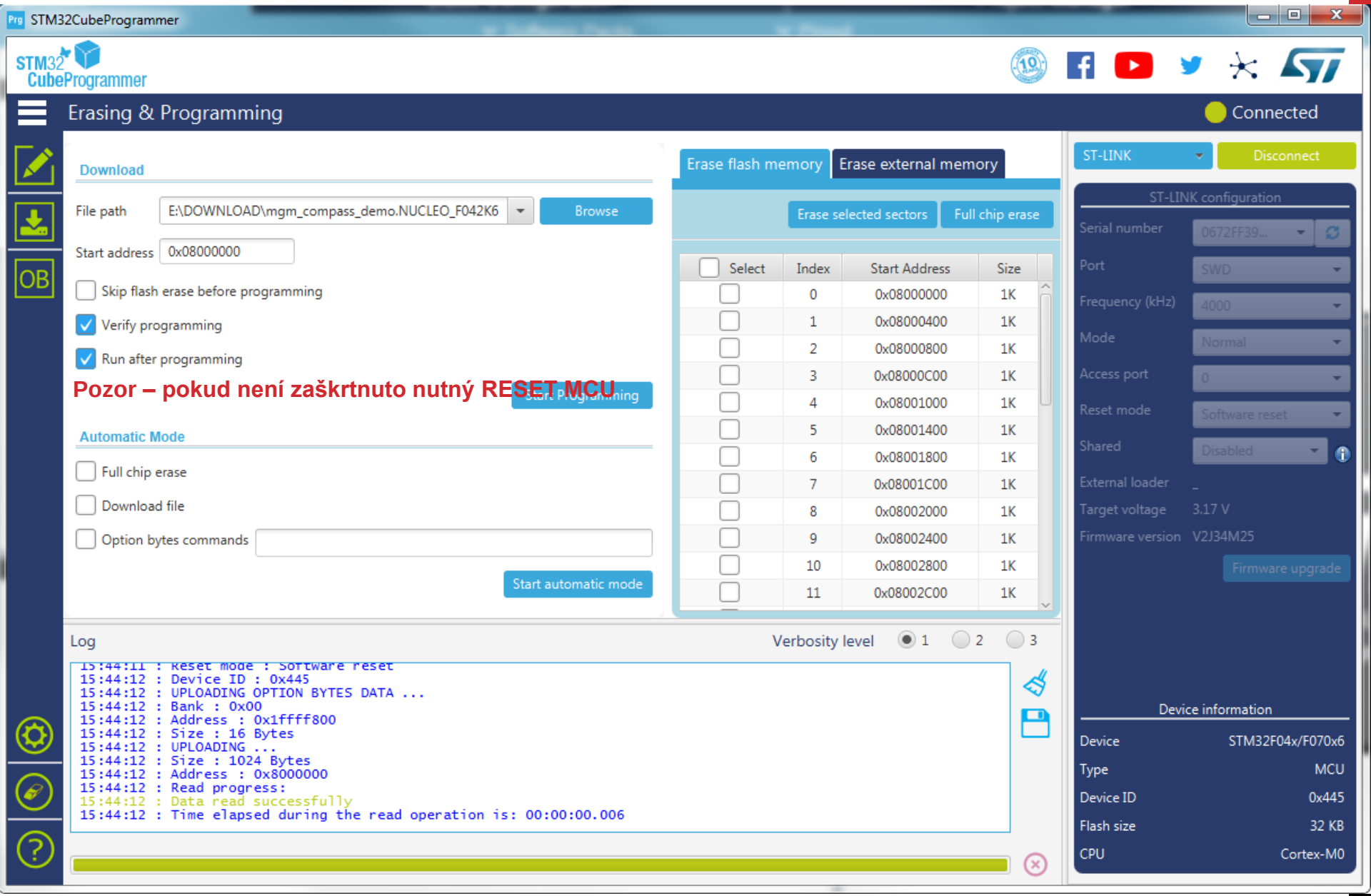
Nahrání programu pomocí DfuSE Demo

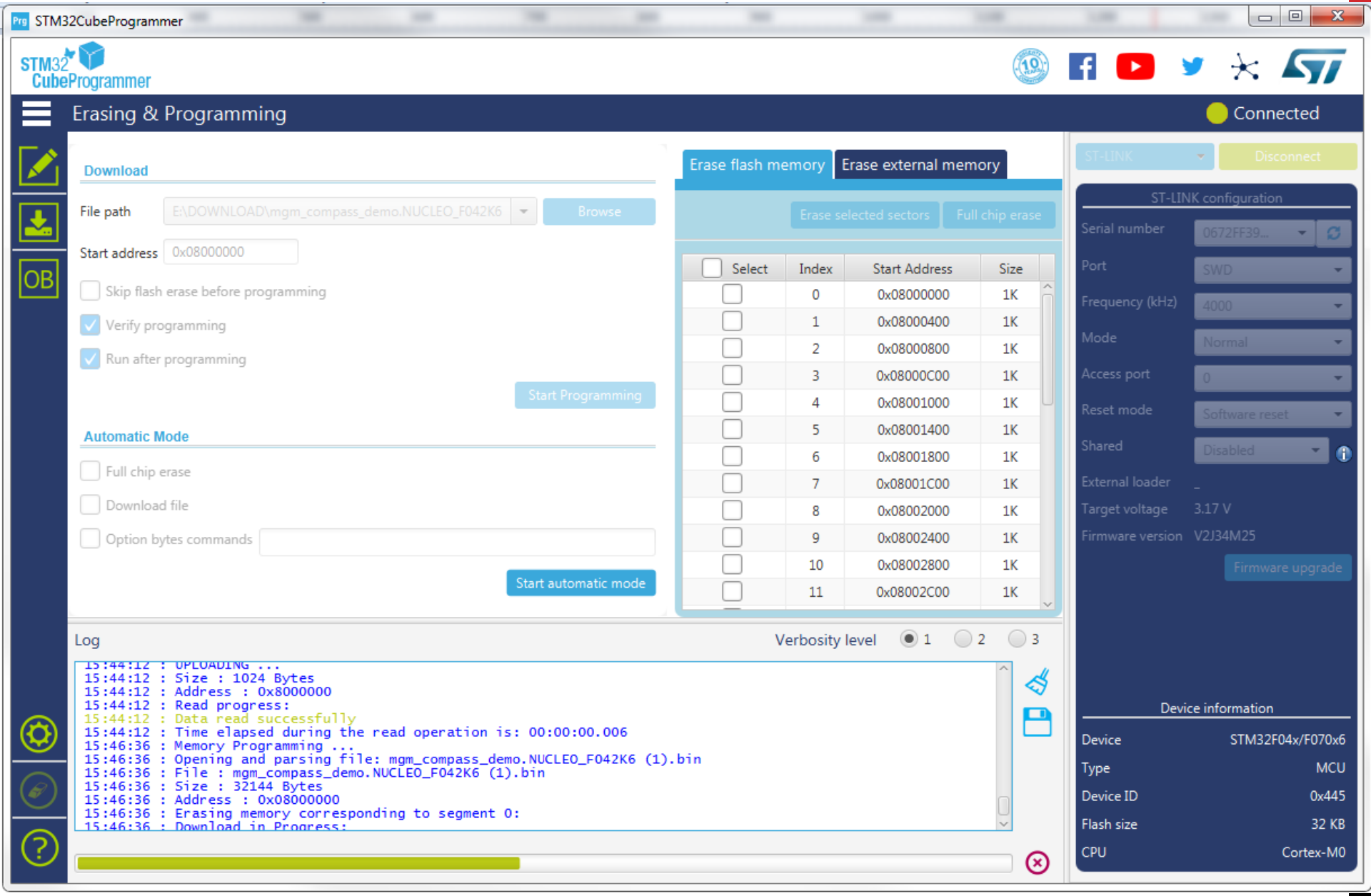


Nahrání programu pomocí Cube programmer

Novější program, umožňuje využít přímo soubor xxx.bin bez potřeby konverze do souboru typu xxx.Dfu, více možností HW připojení. Na některých PC „blbne“ problém s USB ovladači zřejmě, lze vyřešit použitím vhodného USB hubu.

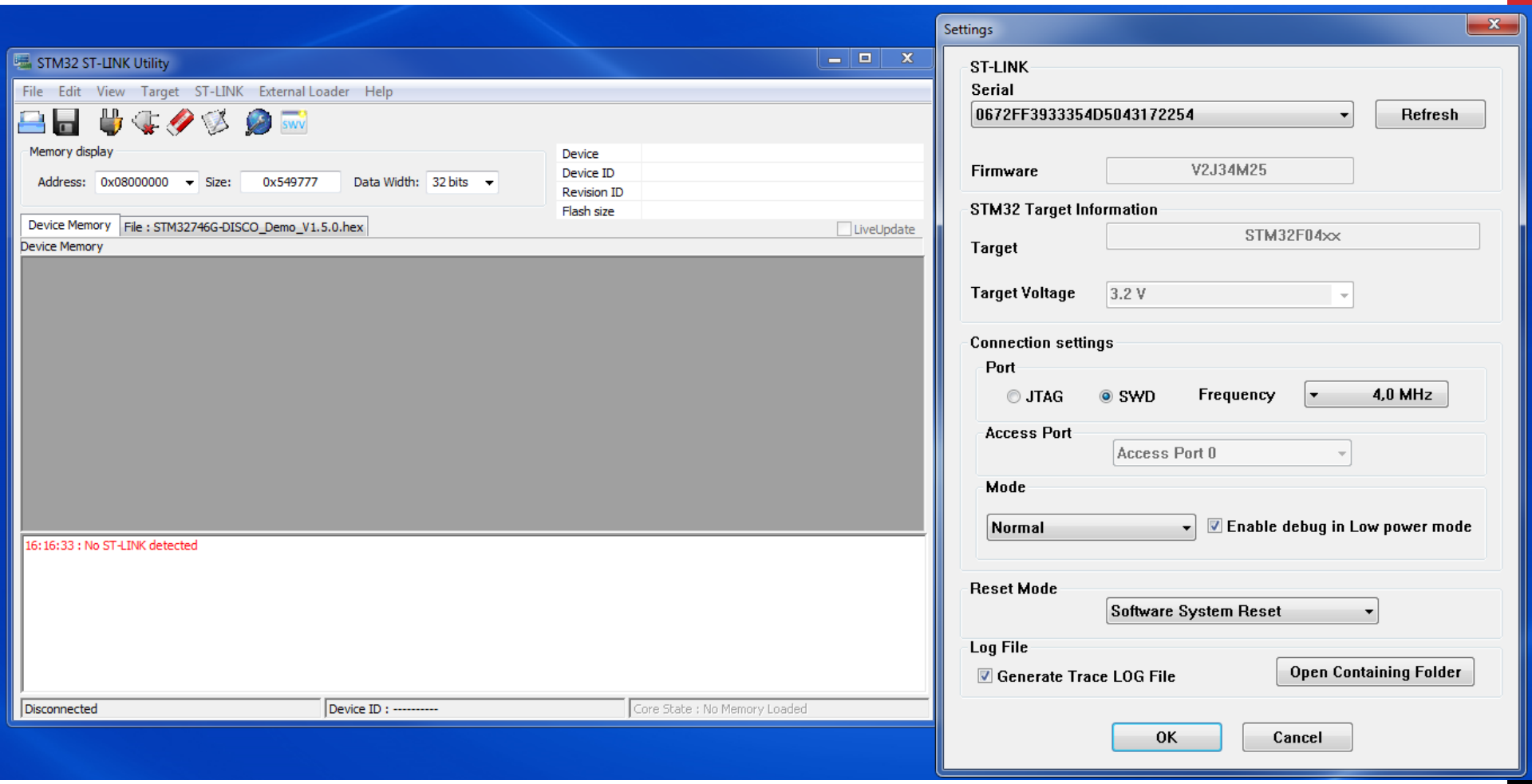



STM32CubeProgrammer
Erasing & Programming
Connected
Download
File path
E:\DOWNLOAD\mgm_compass_demo.NUCLEO_F042K6
Browse
Start address
0x08000000
Skip flash erase before programming
Verify programming
Run after programming
Pozor – pokud není zaškrtnuto nutný RESET MCU
Start Programming
Automatic Mode
Full chip erase
Download file
Option bytes commands
Start automatic mode
Erase flash memory
Erase external memory
Erase selected sectors
Full chip erase
Select
Index
Start Address
Size
0 0x08000000 1K
1 0x08000400 1K
2 0x08000800 1K
3 0x08000C00 1K
4 0x08001000 1K
5 0x08001400 1K
6 0x08001800 1K
7 0x08001C00 1K
8 0x08002000 1K
9 0x08002400 1K
10 0x08002800 1K
11 0x08002C00 1K
ST-LINK
Disconnect
ST-LINK configuration
Serial number
Port
SWD
Frequency (kHz)
4000
Mode
Normal
Access port
Reset mode
Software reset
Shared
Disabled
External loader
Target voltage
3.17 V
Firmware version
V2J34M25
Firmware upgrade
Log
Verbosity level
1
2
3
15:44:11 : Reset mode : Software reset
15:44:12 : Device ID : 0x445
15:44:12 : UPLOADING OPTION BYTES DATA ...
15:44:12 : Bank : 0x00
15:44:12 : Address : 0x1ffff800
15:44:12 : Size : 16 Bytes
15:44:12 : UPLOADING ...
15:44:12 : Size : 1024 Bytes
15:44:12 : Address : 0x8000000
15:44:12 : Read progress:
15:44:12 : Data read successfully
15:44:12 : Time elapsed during the read operation is: 00:00:00.006
Device information
Device STM32F04x/F070x6
Type MCU
Device ID 0x445
Flash size 32 KB
CPU Cortex-M0


STM32CubeProgrammer
STM32 CubeProgrammer
Erasing & Programming
Connected
Download
File path
E:\DOWNLOAD\mgm_compass_demo.NUCLEO_F042K6
Browse
Start address
0x08000000
Skip flash erase before programming
Verify programming
Run after programming
Start Programming
Automatic Mode
Full chip erase
Download file
Option bytes commands
Start automatic mode
Erase flash memory
Erase external memory
Erase selected sectors
Full chip erase
Select
Index
Start Address
Size
0 0x08000000 1K
1 0x08000400 1K
2 0x08000800 1K
3 0x08000C00 1K
4 0x08001000 1K
5 0x08001400 1K
6 0x08001800 1K
7 0x08001C00 1K
8 0x08002000 1K
9 0x08002400 1K
10 0x08002800 1K
11 0x08002C00 1K
ST-LINK
Disconnect
ST-LINK configuration
Serial number
0672FF39...
Port
SWD
Frequency (kHz)
4000
Mode
Normal
Access port
0
Reset mode
Software reset
Shared
Disabled
External loader
Target voltage
3.17 V
Firmware version
V2J34M25
Firmware upgrade
Log
Verbosity level
1
2
3
15:44:12 : Size : 1024 Bytes
15:44:12 : Address : 0x8000000
15:44:12 : Read progress:
15:44:12 : Data read successfully
15:44:12 : Time elapsed during the read operation is: 00:00:00.006
15:46:36 : Memory Programming ...
15:46:36 : Opening and parsing file: mgm_compass_demo.NUCLEO_F042K6 (1).bin
15:46:36 : File : mgm_compass_demo.NUCLEO_F042K6 (1).bin
15:46:36 : Size : 32144 Bytes
15:46:36 : Address : 0x08000000
15:46:36 : Erasing memory corresponding to segment 0:
Device information
Device
STM32F04x/F070x6
Type
MCU
Device ID
0x445
Flash size
32 KB
CPU
Cortex-M0

STlink utility

starší, ale funkční držák...



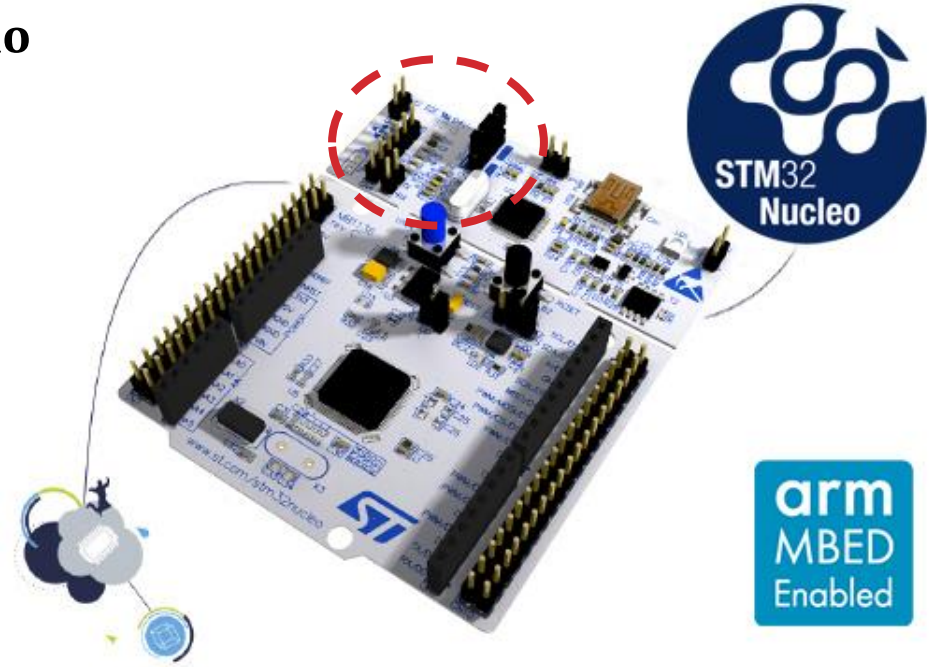
Nahrávání programu do F0 – Lab pomocí Nucleo 64

Alternativní způsob nahrávání s využitím kteréhokoliv kitu Nucleo 64

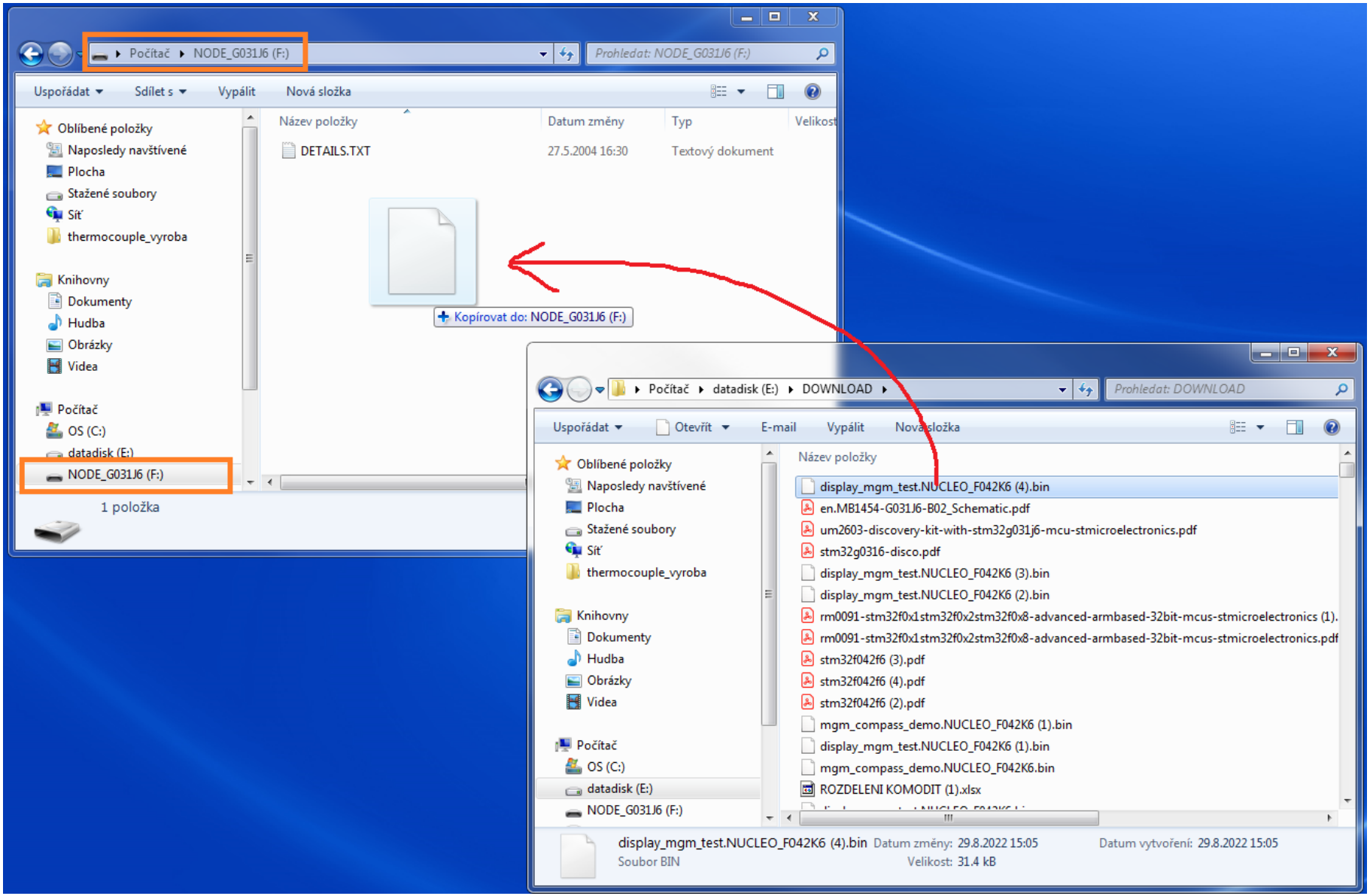
Odpojit vytažením „jumperů“ zabudovaný procesor a pomocí vodičů napojit na desku F0 – lab na piny č. 19 a č. 20 (rozhraní SWD).

Nahrání binárního souboru pouhým **nakopírováním přes USB**

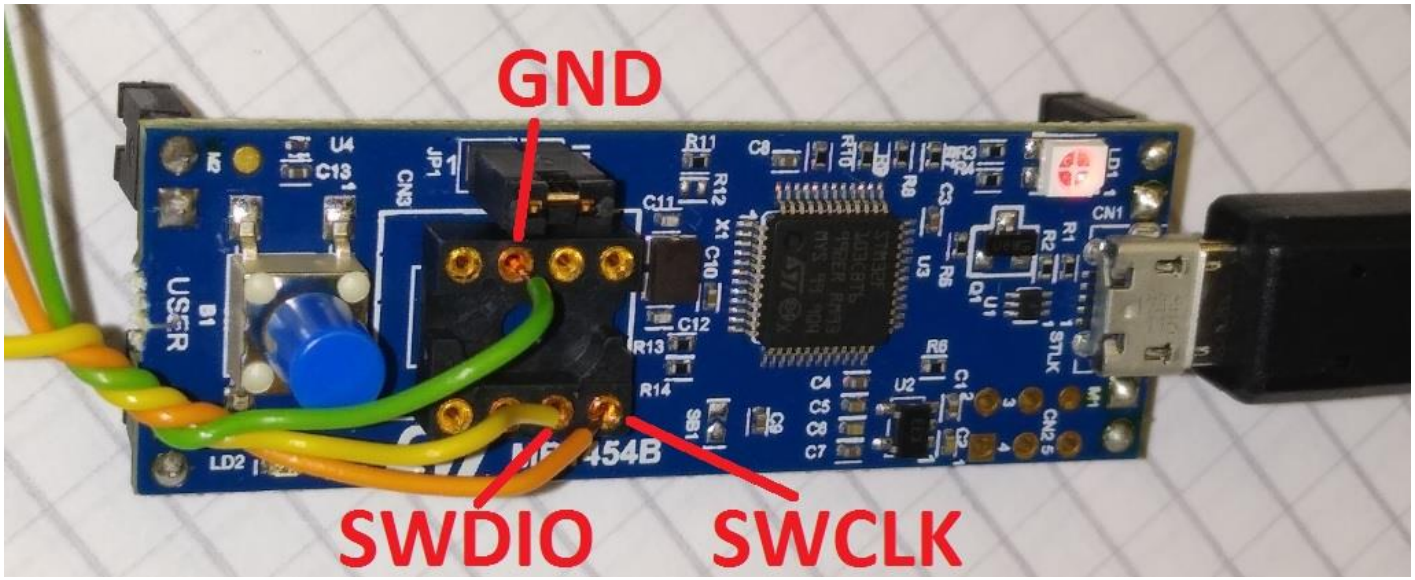
Možnost využití **zabudovaného převodníku UART – USB**.



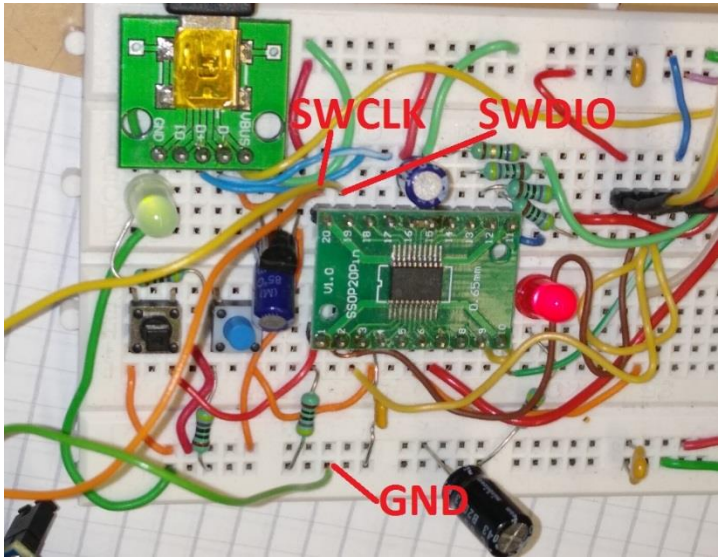
PROGRAMOVÁNÍ – KOPÍROVÁNÍ NA FLASHDISK



PROGRAMOVÁNÍ – KOPÍROVÁNÍ NA FLASHDISK - ZAPOJENÍ



3 - GND
7 – SWDIO
8 - SWCLK



vnější řádky - GND
19 – SWDIO
20 - SWCLK

DRUHÝ PROGRAM

- **„Lampička“ s LED – tlačítko rozsvítí LEDku**
- **Malý demo kit nemá vlastní tlačítko**
- **Pro desku pouze definujeme kde je tlačítko a LED umístěna**

DRUHÝ PROGRAM

```
#include "mbed.h"
                                        //určení kde je tlačítko


DigitalIn mybutton(PB_1);           // tlačítko na PB_1 u F042
DigitalOut myled(PA_4);               //umístění LED na PA_5 u F042

int main() {                          //hlavní funkce
  while(1) {                          //smyčka
    if (mybutton == 0) {              //podmínka zda bylo stisknuto
tlačítko
                                      //0 - ANO, 1 - NE
      myled = !myled;                 // pokud ano, tak zapni/vypni LED
      wait(0.2);                      // 200ms pro eliminaci dvojkliku
    }
  }
}
```

Označení pinu (např. PA_4 je na pinu č. 10) dle dokumentace procesoru a schématu F0 – Lab.

Externí Pull- Up rezistor (68k nebo 22k… 10 k) na PB_1

Zápis Wait

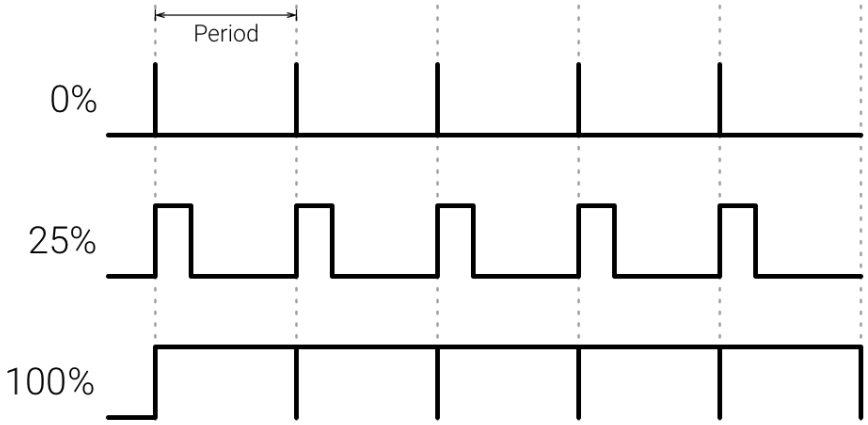
Různý zápis stejného čekání

```
#include "mbed.h"

int main() {
    while(1) {
        wait(0.2); // 200 ms
        wait_ms(200); // 200 ms
        wait_us(200000); // 200 ms
    }
}
```

TIMER - PWM

Pulzně šířková modulace PWM

- **Signál nabývá hodnot log0/log1**
- **Poměr stavů zapnuto/vypnuto je střída**
- **Čas přenosu jedné střídy je perioda**
- **Příklad: LED osvětlení, DC motory, topení, řízení polohy – servo motor, atd.**



Příklad PWM

PWM na pinu PA_4, postupné rozsvěcování a zhasínání LED

```
#include "mbed.h"
PwmOut led(PA_4);
short r;

main() {
    r=0 ;
    led.period(0.001);
    while(1) {
        led = 0.5+ 0.5*sin((r++)/32.0);
          wait_ms(10);
   }
}
```

Short - 2 bytes -32768 to 32767

Krátký tutorial pro C http://www.stat.cmu.edu/~brian/cprog.html

Příklad použití analogového vstupu - třída AnalogIn

Čtení analogového napětí na PA_3, stálé nastavování střídy PWM na pinu PA_4 podle velikosti napětí na PA_3

```
#include "mbed.h"

AnalogIn voltage(A2); // PA_3
PwmOut led(PA_4);

int main() {
    led.period(0.001);
    while(1) {
        led = voltage;
    }
}
```

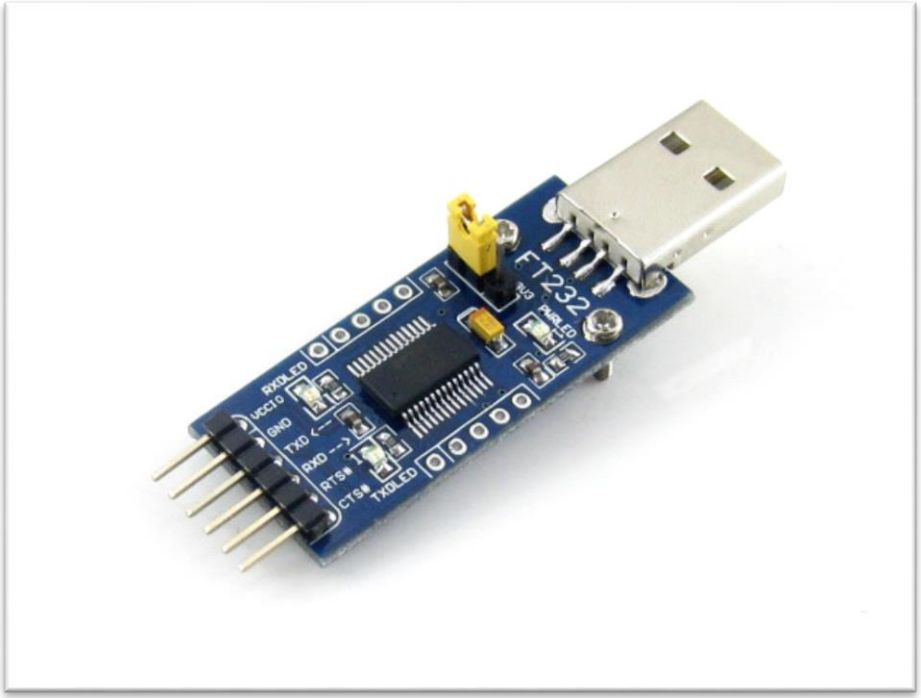
Příklad použití třídy Ticker

Umožňuje provádět operace v konstantních časových intervalech, aniž by byla zatížena hlavní smyčka programu (zde sice nedělá nic, „ale mohla by“). Ticker aktivuje funkci **překlápění po 0,5 sec.**

```
#include "mbed.h"

DigitalOut led(PA_4);
Ticker ticker;

void function() {
    led = !led;
}

int main() {
    ticker.attach(&function, 0.5);
    while(1);
}
```

Jak dále - textová komunikace

- V mbed na demo kitu lze také snadno **komunikovat pomocí terminálu** – „povídání s mikrokontrolérem“, případně **jeho řízení pomocí PC, komunikační kanál UART**
- Pro desku F0 – Lab potřebujeme **externí převodník „UART -> USB**“ převodník (cca 90 Kč v GME)

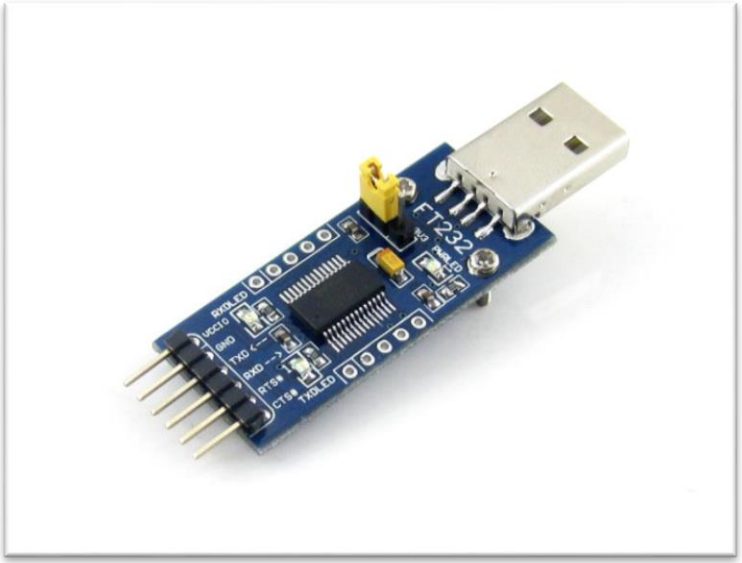


Komunikace pro ladění

Při použití mbed **není k dispozici „ladění“** programu **– debugging**

Možné způsoby:

- **Využít textové výpisy při průchodu nějakou části programu**
- **Využít blikání LED – bliknout několikrát, nebo použít PWM pro nastavení jasu,** příp. využít **několik** signalizačních **LED**
- **PWM signalizace pomocí sluchátka** (různé tóny)



Kam dále

- **Mnoho dalších funkcí – PWM, časovače, AD a DA převodníky**
- **Jak začít**
  - Použít předlohy na jednoduché programy a zkoušet modifikovat
  - Dále samotná mbed komunita vytváří mnoho programů
  - Nucleo a jeho extension boardy – např. Bluetooth, P-nucleo
  - Internet je plný tutoriálů
- **Pro pokročilé funkce i programátory**
  - STM Cube MX

Více informací BP Lukáš Bilesch, další možnosti ladění s mbed, popis tříd, možnost volání funkcí v asemberu

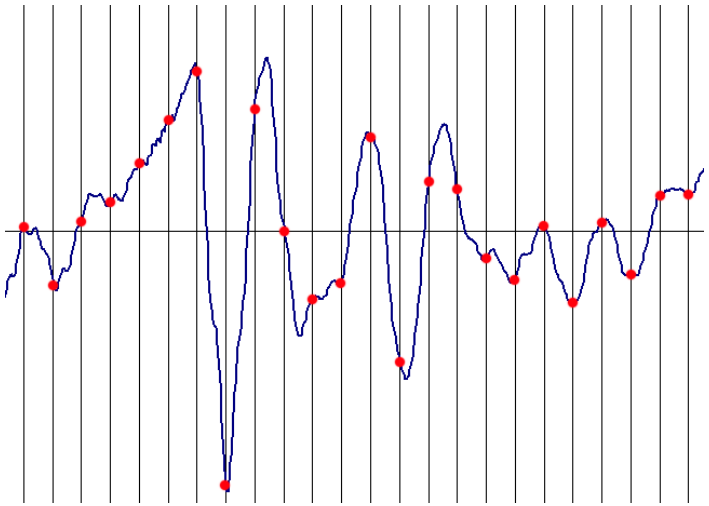
ADC/DAC

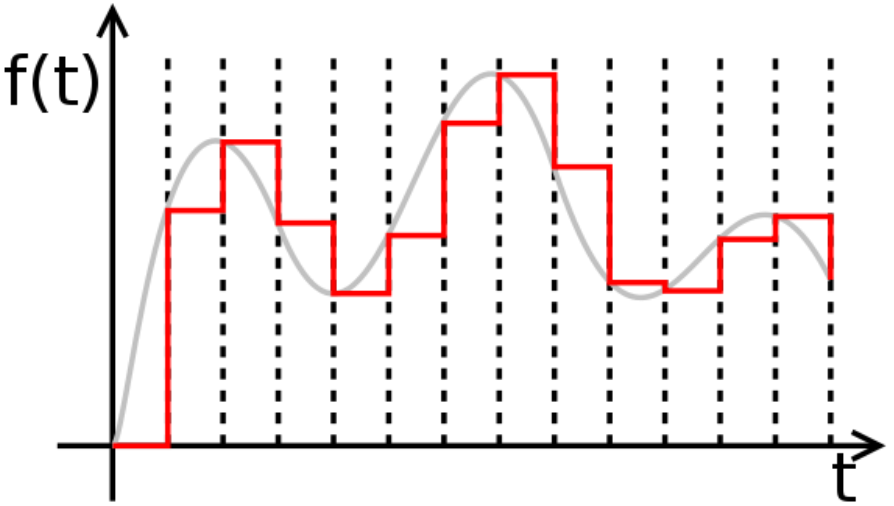
ADC -Analogově digitální převodník

- **Převod analogového signálu na digitální**
- **Příklad: voltmetr, mikrofon, záznam signálu**

Digitálně analogový převodník DAC (opak ADC)

- **Příklad: výstup přehrávače, tvorba signálu** (STM32F042 nemá DAC)





AD/DAC

- **Pro ADC - AnalogIn nazev(pin)**
- **Čtení hodnoty promenna = nazev.read()**
- **Hodnota je v rozmezí 0 až 1, procentuálně mezi 0 a maximálním napětím + 3.3V**
- **Reálné napětí se určí jako promenna * 3.3 V**

- **Pro DAC – AnalogOut nazev(pin)**
- **Zápis hodnoty nazev.write(hodnota)**
- **Hodnota je také v rozmezí 0-1, tj. kolik procent max napětí bude na výstupu**

Zvuk- generace pomocí reproduktoru

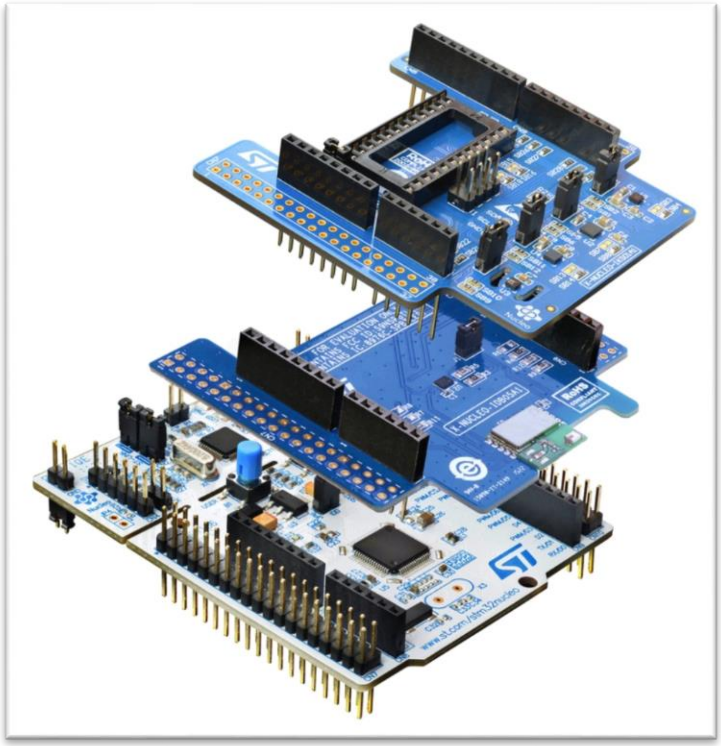
Nasledující příklad ukazuje jednoduchý program – hrací skříňka.
Na pin PA_4 stačí připojit reproduktor s budičem a pak doplnit melodii vybrané písně. (Program pokračuje na dalším snímku.)

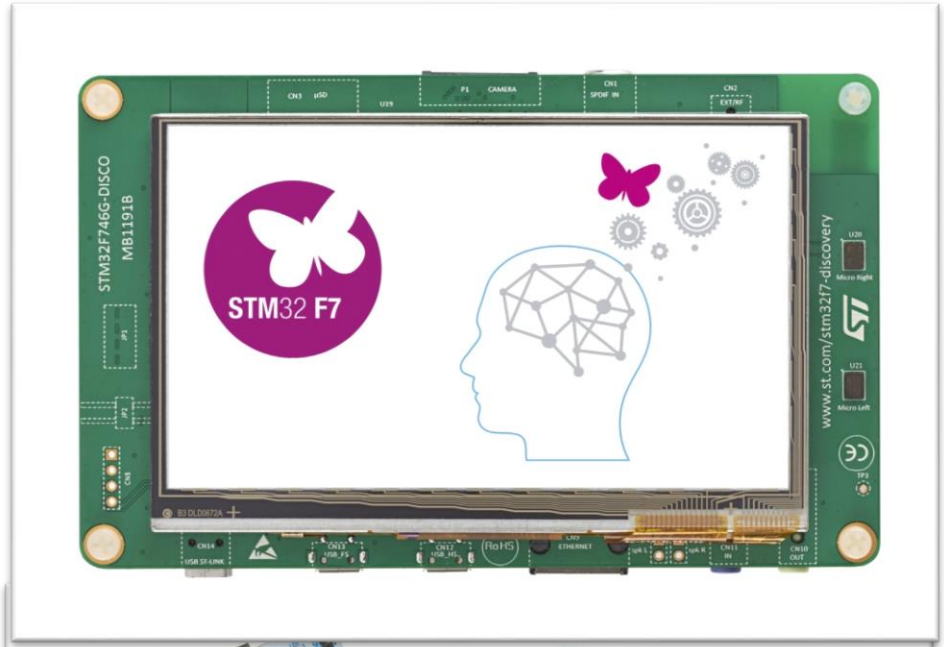
```
#include "mbed.h"
#define C 261.63
#define C_S 277.18
#define D 293.66
#define D_S 311.13
#define E 329.63
#define F 349.23
#define F_S 369.99
#define G 392
#define G_S 415.3
#define A 440
#define A_S 466.16
#define B 493.88
class Buzzer {
public:
    Buzzer(PinName pin) : _pin(pin) {}
```

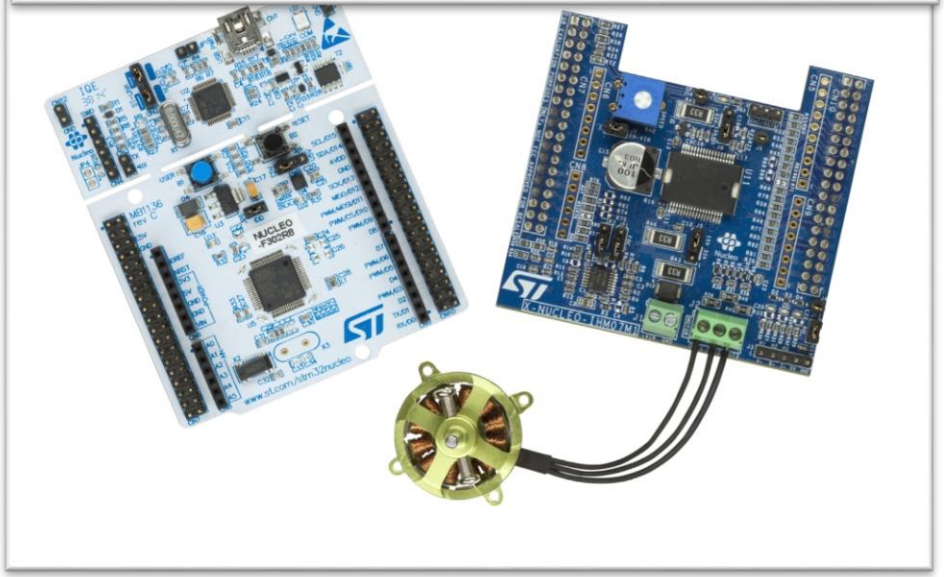
Zvuk

```
. void beep(int freq, float time) {
        _pin = freq? 0.5:0;
        _pin.period(freq? 1.0/freq : 1);
        wait(time);
   }
private:
     PwmOut _pin;
};
Buzzer buzzer(PA_4);
int main() {
    buzzer.beep(G, 0.5);
    buzzer.beep(G, 0.5);
    buzzer.beep(E, 0.5);
    buzzer.beep(0, 0.5);
    buzzer.beep(G, 0.5);
    buzzer.beep(G, 0.5);
    buzzer.beep(E, 0.5);
    buzzer.beep(0, 0.5);
    buzzer.beep(G, 0.5);
    buzzer.beep(G, 0.5);
    buzzer.beep(A, 0.5);
    buzzer.beep(G, 0.5);
    buzzer.beep(G, 1);
    buzzer.beep(F, 1);
    buzzer.beep(0, 0);
}
```

KAM DÁLE




STM32F746G-DISCO
MB1191B
STM32 F7
www.st.com/stm32f7-discovery



Na závěr

Poznámka: S využitím mbed lze pracovat i s **knihovnami** „HAL“ a „Low Layer“ pro STM32.

Lze tvořit i programy a funkce v **assembleru** procesoru ARM cortex – M0 (M3, M4), které se vlají z C.

Mbed – první krok pro první seznámení , dále využít standardní profesionální nástroje, Cube IDE, IDE Keil , IDE - IAR,…

Mbed není profesionální nástroj, je to podobné , jako Arduino IDE, ale pro ARM. Je zde „korektní C“.

V předmětu programování - prof. Faigl, doc. Vítek - se používá stejný nástroj (mbed) ve spojení s kitem Nucleo F446- tedy malá příprava.

ZÁVĚREČNÁ SOUTĚŽ

- **Vytvořte CO NEJKRATŠÍ program, který 5x zabliká LEDkou a pak ji nechá svítit 2 sekundy a navždy zhasne.**
- **Doba svitu bude 0.5 sekundy, zhasnuto bude 1 sekundu – LED musí zasvítit 5x.**

Nápověda:

```
for(int i = 0;i<5;i++){

/*cyklus se provede 5x, i je
proměnná, která se
inkrementuje, v každém cyklu
se porovná s finální
hodnotou */



//light magic here


}
```

VÝPISY DO TERMINÁLU

- Deska ani kit nemají displej -> vypisovat na PC terminál
- Kit má Virtual COM port – sériové rozhraní, není třeba žádný převodník
- V mbed funkce printf("Text %d %f \n", decimalni_hodnota, desetinna_hodnota)
- Stáhnout terminál pro PC – putty (musí se zadat ručně COM port), Cool Term, RealTerm (uživatelsky přívětivější)

- Deska nemá druhý procesor – nemá COM port
- Je třeba vycházet z tohoto příkladu:
- https://os.mbed.com/users/va009039/code/F042K6_USBDevice/
- A přidat jako první řádek v main():
- SYSCFG->CFGR1 |=0x10; //remap to USB

```
#include "mbed.h"
#include "USBSerial.h"

unsigned char comIN =0;
unsigned char state_var =0;

int main() {
 int i,j,k;

   SYSCFG->CFGR1 |=0x10;   //Pins PA11/12 instead of pins PA9/10 (umožňuje použít USB)
   USBSerial serial;

   while(1){
     wait(0.025); //  25 ms  rychlejší zasílání může způsobovat problémy
                  // při nesplnění vzorkovací podmínky bude zase vznikat aliasing
     if(state_var){
       serial.printf("Test ANSI VT100\r\n");
       while(1){
        wait(0.1);
```