ČVUT FEL

mbed tutoriál

Filip Schwank
Katedra měření
13.9.2016

Úvod

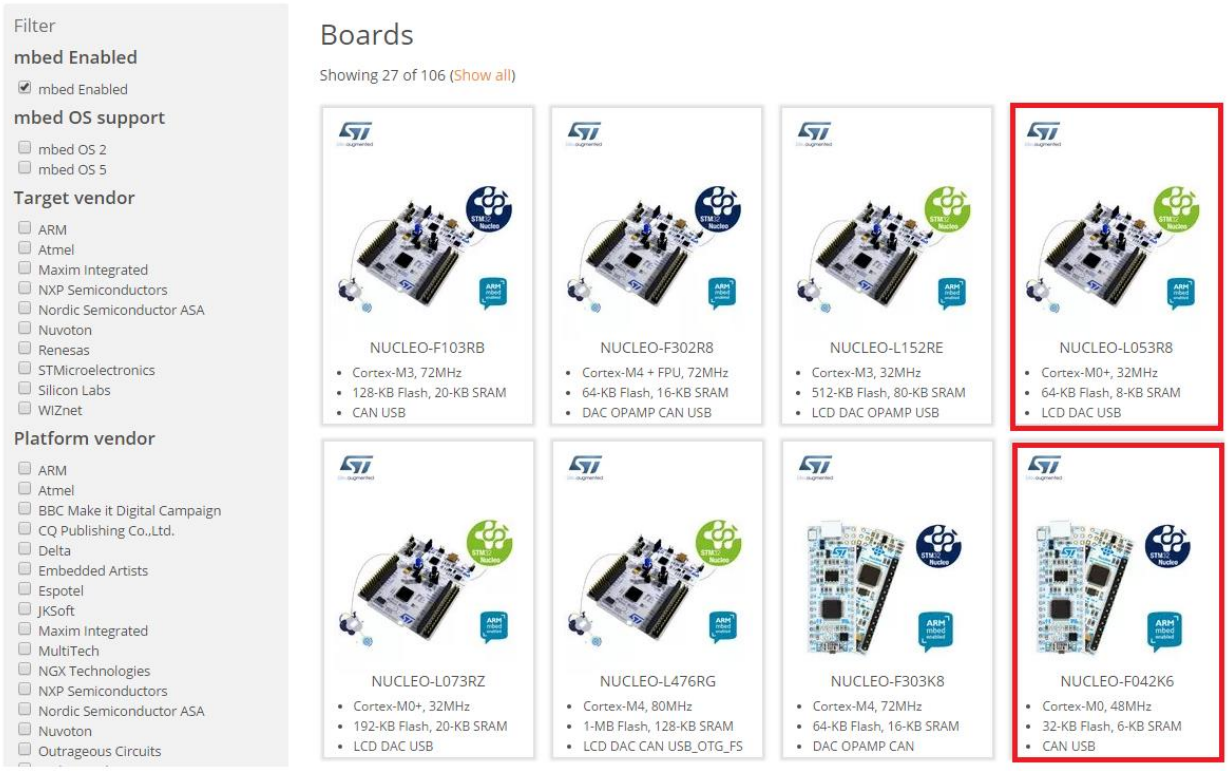
Mbed je jednoduchý nástroj pro programování mikrokontrolérů v jazyce C++. Člověk se v něm snadno zorientuje i bez znalosti programování. Tento tutoriál by tedy měl sloužit jako takový úvod do programování jako takového, zároveň ale by měl jednoduchými příklady ukázat, co skrývá svět mikrokontrolérů.

Vývojové prostředí a mikrokontrolér

Pro tento tutoriál bude využit mikrokontrolér firmy ST Microelectronics - STM32 Nucleo-64 s procesorem STM32L053R8, případně procesor STM32F042F6P6 na vlastní „bastl desce“.

Pro vývoj aplikací v mbedu se dá použít online vývojové prostředí na developer.mbed.org/compiler.

Nejdříve je třeba se na webu zaregistrovat a potom si přidat platformu na adrese: https://developer.mbed.org/platforms/



Obrázek 1 - výběr mikrokontroléru

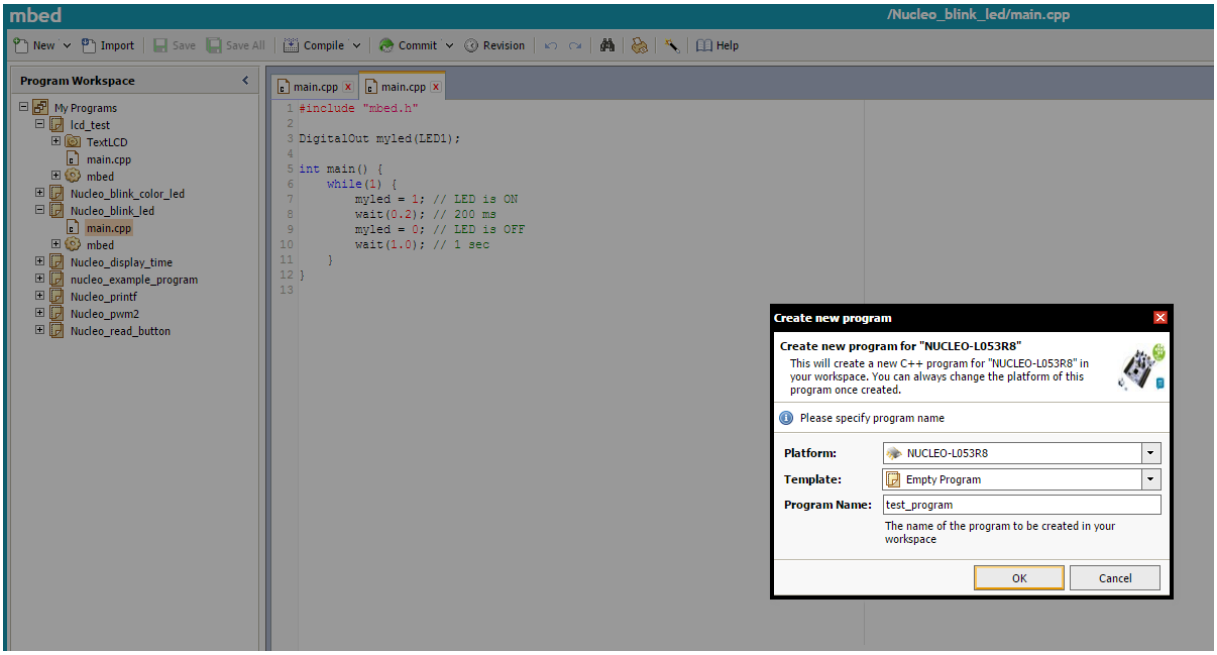
Zde se zvolí kit nucleo-L053R8, nebo F042K6 (pro „bastl desku“). Na další stránce se jen potvrdí výběr tlačítkem „Add to your compiler“.

Nyní lze ve vývojovém prostředí vytvořit nový projekt pro daný kit a začít s programováním (New – New Program – zvolit kit a předlohu – lze zobrazit příklady, nebo prázdný projekt).

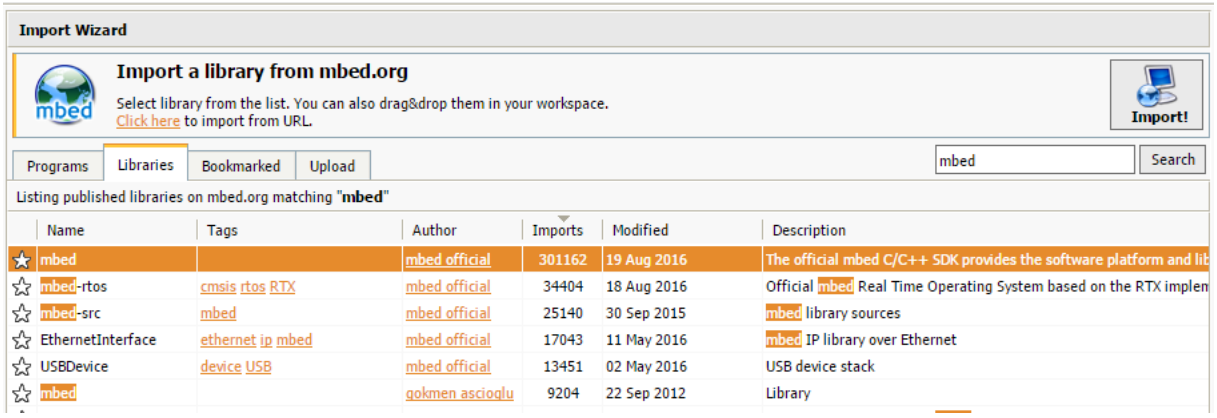
Dále pravým tlačítkem na název projektu – New File… - pojmenovat main.cpp

Nakonec tlačítko Import – vyhledat mbed – první (oficiální) knihovna a potvrdit tlačítkem Import!

Každý program se potom zkompiluje tlačítkem Compile a automaticky se stáhne do počítače.



Obrázek 2 - založení projektu



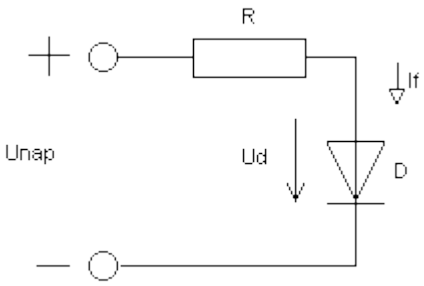
Obrázek 3 - Vložení mbed knihovny

První program – blikání led

Jelikož kity nemají jednoduchý textový výstup, tak „Hello World“ bude až později a příklad tedy bude o něco jednodušší – blikání LED která je umístěna na kitu / bude připojena k „bastl desce“.

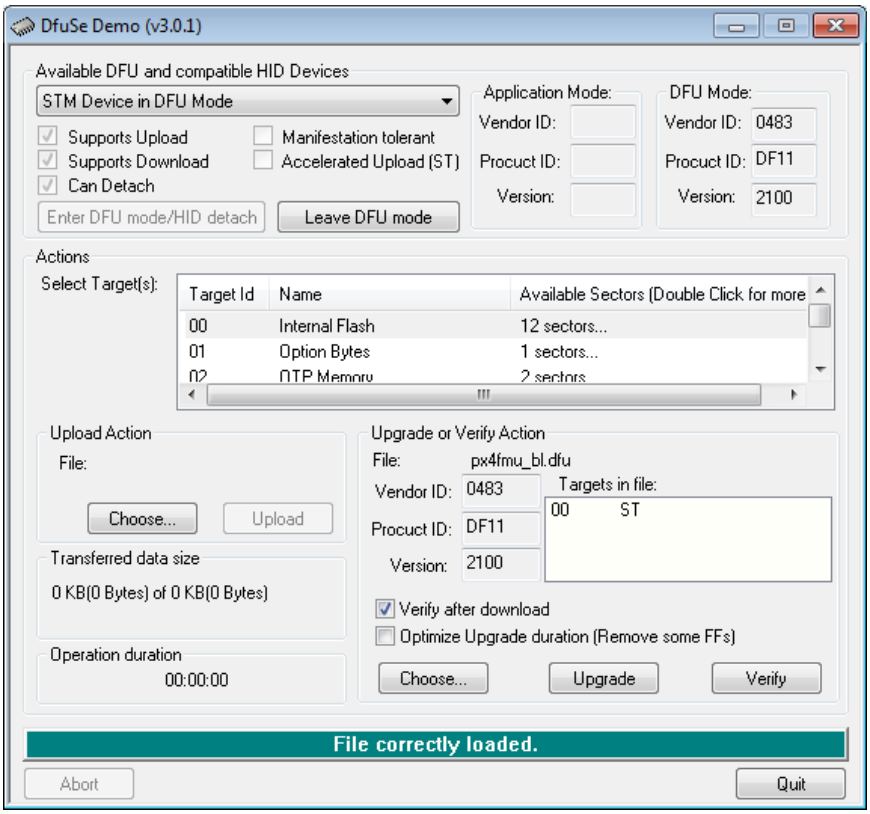
```
#include "mbed.h" // hlavička programu – říká, že použijeme mbed

DigitalOut myled(PA_5); // 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
    }
}
```

Tento kus kódu stačí nakopírovat do prázdného projektu a příkazem Compile vytvořit spustitelný soubor pro kit. Ten se v případě L053 jednoduše překopíruje jako na Flash – kit se jako paměť objeví a nahraje. Pak už program běží a zelená LED by měla blikat.



Obrázek 4 - zapojení LED

V případě F042 je třeba využít program DFU loader, který vytvořený spustitelný soubor nahraje do procesoru. Dále je třeba mít připojenou LED ke správnému pinu – v tomto případě pin PA5 (pin 11 na pouzdře)



Obrázek 5 - DFU loader

Druhý program – svícení LED tlačítkem

První program bude upraven na zapnutí/vypnutí LED po stisku tlačítka. Následující kód ukazuje jak na to:

```
#include "mbed.h"
                                //určení kde je tlačítko
DigitalIn mybutton(USER_BUTTON);//L053
//DigitalIn mybutton(PB_1);     //F042
DigitalOut myled(PA_5);         //umístění LED

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

Při použití L053 stačí kód překopírovat, na „bastl desku“ (F042) je nutné ještě odkomentovat řádku s definicí tlačítka na PB1 (pin 14 na pouzdře) pinu a naopak zakomentovat řádku s definicí na USER_BUTTON (ten je na L053 na pinu PC13, který „bastl deska“ vyvedený nemá). Také je třeba tlačítko zapojit do desky na zmíněný 14. pin pouzdra – stačí zapojit tlačítko jednou stranou k pinu a druhou k zemi (GND).

Po nahrání programu se při stisku tlačítka rozsvítí LED, druhým stiskem zhasne. Ani při „zuřivém“ mačkání tlačítka LED nebude blikat rychleji jak 200 milisekund díky eliminaci dvojkliku pomocí čekání.

BONUS: pomocí zmíněných funkcí vytvořte program, který po stisku tlačítka 3x problikne LED a pak ji nechá zhasnutou.

Třetí program – jas LED

I v tomto příkladu bude svítit LED. Tentokrát ale tlačítko bude řídit její jas:

```
#include "mbed.h"
                                //určení kde je tlačítko
DigitalIn mybutton(USER_BUTTON);//L053
//DigitalIn mybutton(PB_1);     //F042
PwmOut led_pwm(PA_5);           //umístění LED (L053)
//PwmOut led_pwm(PA_6);           //pro PWM na F042 musí být LED na PA6
float strida=0.5;                 //proměnná určující nastavení střídy

int main() {                    //hlavní funkce

    led_pwm.period_ms(10);       //základní nastavení periody PWM
    led_pwm.write(strida);          //a střídy PWM (50%)

  while(1) {                    //smyčka
    if (mybutton == 0) {        //podmínka zda bylo zmáčknuto tlačítko
                                //0 - ANO, 1 - NE
        strida = strida + 0.25; //zvýšení střídy o 25%
                                //(také se dá použít zkrácený
                                //zápis strida+=0.25);
        if(strida == 1.25){     //pokud byla přesažena hodnota 100%
            strida = 0.0;       //tak se střída resetuje na 0%
            }
        led_pwm.write(strida);  //zápis nové střídy
      wait(0.2);                // 200ms pro eliminaci dvojkliku
    }
  }
}
```

Tlačítkem se tedy mění jas LED – postupně střídou 0% ,25%, 50%, 75%, 100%. Pro L053 tu není žádná změna nastavení, ale pro „bastl desku“ je nutné buďto LED přesunout na pin PA6 (na pouzdru 12), nebo přidat druhou LED, protože na původním pinu PA5 není generátor PWM.

Čtvrtý program – morseovka na LED

Na známé LED nyní bude blikat posloupnost zakódovaná v Morseově abecedě. Pro jednoduchost programu se budou zadávat přímo „tečky a čárky“. Pro zakódování poslouží online konvertor - http://morsecode.scphillips.com/translator.html

Do programu lze zadat jakoukoli posloupnost teček a čárek (v maximální délce 255 znaků), ta se potom po stisku tlačítka postupně „vybliká“ na LED.

```
#include "mbed.h"
                                  //určení kde je tlačítko
DigitalIn mybutton(USER_BUTTON);//L053
//DigitalIn mybutton(PB_1);     //F042
DigitalOut myled(PA_5);           //umístění LED

//zde je zakódovaná posloupnost znaků
char morse[255]=".... . .-.. .-.. --- / .-- --- .-. .-.. -..";

void blik_carka(){ //funkce pro vyblikání čárky
    myled=1;
    wait(1.5);
    myled=0;
    }
void blik_tecka(){  //tečky
    myled=1;
    wait(0.4);
    myled=0;
    }
void blik_morse(){//funkce která postupně vybliká každý znak
    int i;//index v posloupnosti
    for (i=0;i<255;i++){//smyčka která projde posloupnost
        //pokud je znak tečka tak jí blikni
        if(morse[i]=='.') blik_tecka();
        //stejně tak čárku
        else if(morse[i]=='-') blik_carka();
        //mezera mezi slovy - čekej
        else if(morse[i]=='/') wait(3);
        //mezera mezi písmeny - čekej taky
        else if(morse[i]==' ') wait(1);
        //v případě jiného znaku přeskoč
        else continue;
        wait(1);
        }
    }
int main() {                      //hlavní funkce
  while(1) {                      //smyčka
    if (mybutton == 0) {          //podmínka zda bylo zmáčknuto tlačítko
      wait(0.2);                  // 200ms pro eliminaci dvojkliku
      blik_morse();
    }
  }
}
```

Výpis do terminálu přes USB

Výpisy z mikrokontroléru lze do počítače dostat několika způsoby – pomocí sériového USART rozraní. Na to má mbed sice přímo vlastní knihovnu (stačí includovat *serial.h*) , ale ke zprovoznění komunikace s počítačem by byl třeba ještě převodní USART -> USB, nebo přímo sériový port, který ale funguje na vyšším napětí. Proto se dá využít tzv. Virtual COM port, kdy se v USB paketech posílají sériová data a počítač si pak myslí, jako by tam sériový port byl. Na to ale neexistuje oficiální mbed knihovna, ale lze použít následující komunitní program, který si každý může importovat do svého kompilátoru.

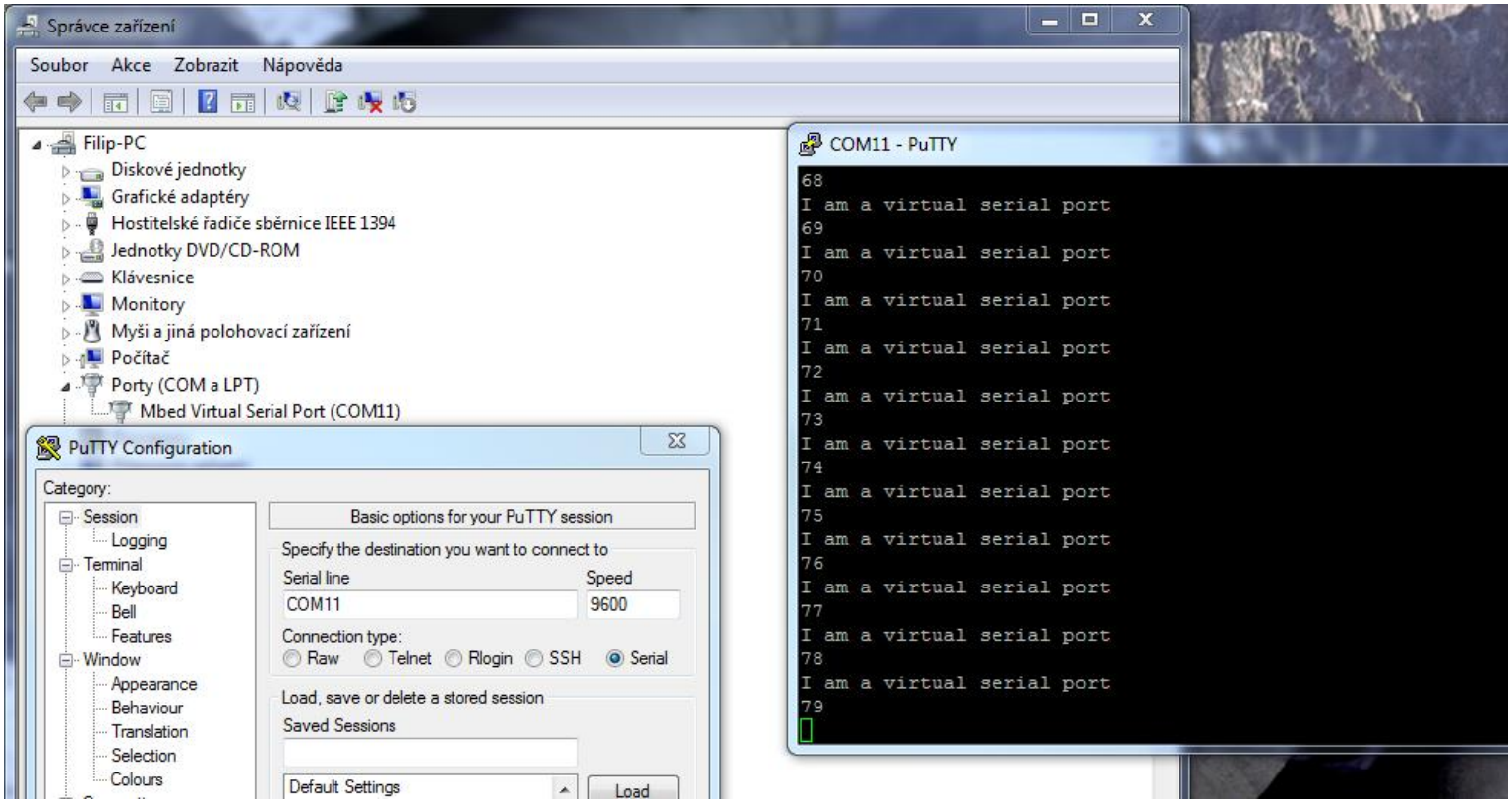
https://developer.mbed.org/users/va009039/code/F042K6_USBDevice/

Tento program běží na obou kitech používaných v tomto tutoriálu (dále na většině STM32F4, STM32L1 a některých F1, F0 – ověřeno zatím jen pro naše 2 kity). Bastl deska má jisté úskalí v tom, že je to malý procesor v malém pouzdře, a tak je třeba USB piny tzv. remapovat (na pinech 17, 18 jsou výchozí PA9, PA10 – po remapu mají funkci pinů PA12, PA13). To se provede jediným příkazem před inicializací programu a jinak není třeba nic dalšího měnit – viz kód.

Pro Windows 7 a níže je třeba také stáhnout do počítače ovladač pro sériový port (ve Správci zařízení se po připojení kitu objeví neznámé zařízení a funkcí Aktualizovat Ovladač nainstalujeme sériový port)- https://developer.mbed.org/handbook/USBSerial

Pro zobrazení dat v počítači je třeba nějaký terminál – např. Putty, CoolTerm a jiné. V každém terminálu je nutné definovat, na kterém COM portu se bude komunikovat a jakou rychlostí (COM lze najít ve Správci zařízení Windows, rychlost je 9600 baud).

```
// https://developer.mbed.org/handbook/USBSerial

//https://developer.mbed.org/users/va009039/code/F042K6_USBDevice/
#include "mbed.h"
#include "USBSerial.h"


int main() {
    SYSCFG->CFGR1 |=0x10;          //remap to USB
    USBSerial serial;              //USB init

    for(int n = 0;; n++) {
        serial.printf("I am a virtual serial port\r\n");
        serial.printf("%d\r\n", n);
        wait_ms(1000);
    }
}
```



Obrázek 6 - Konfigurace PuTTy a běh programu v terminálu

Rozšíření výpisu o čtení z klávesnice

Kromě výpisů lze také pomocí konzole posílat do mikrokontroléru znaky z klávesnice. Ten je přečte a může reagovat různými způsoby – například udělá echo (odešle zpět, co přečetl), nebo může rozsvítit třeba LED, aktivovat PWM a atd., dle programu.

Čtení probíhá v tzv. blokujícím módu, proto pokud by se zavolal *serial.getc()* tak by program čekal, dokud neobdrží nějaký znak. Toto se dá řešit tak, že se v programové smyčce v každé iteraci kontroluje, zda není nový znak k přečtení (*serial.readable()*) – pokud je, tak se použije výše zmíněná funkce k přečtení a pokračuje se dál.

Číst by se dalo také z přerušení, ale kvůli remapu, který musí být před deklarací USBSerial instance, nelze mít instanci globální a nešlo by číst znaky pomocí funkce, která se po přerušení zavolá (stačilo by zavolat *serial.attach(pointer na čtecí funkci)*).

Následující kód je rozšířením předchozího příkladu o čtení z klávesnice a rozsvícení LED pomocí klávesy „L“.

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

int main()
{
    SYSCFG->CFGR1 |=0x10;                   //remap to USB
    USBSerial serial;                       //USB init
    DigitalOut myled(PA_4);                 //LED on PA4

    for(int n = 0;; n++) {                  //endless loop

        if(serial.readable()) {             //is there a new char to read?
            char c=serial.getc();           //if yes get it and
            serial.printf("Received: %c\r\n\0",c);//print it
            if(c=='l') myled=!myled;
        }

        serial.printf("I am a virtual serial port\r\n");
        serial.printf("%d\r\n", n);         //info about how many "seconds"
        wait_ms(1000);                      //we are running
    }
}
```