1. Nakopírovat do adresáře projektů „knihovnu“, tj. komplet adresář Libraries_3_5
    a. tj. strom adresářů je

```
+-  Libraries_XXX
|   +-- CMSIS
|    +-- STM32F10x_StdPeriph_Driver
+-  Projekt_1
|       Projekt_1.uvproj
|      Main.c
+-  Projekt_2
       Projekt_2.uvproj
       Main.c
```

2. Nový projekt do nového adresáře – **STM32F107VC**
3. Nechat přikopírovat statup_stm32f10x_cl.s, příp. novější lze najít v knihovně
    a. ve verzi Keil 4.72 a Libraries 3.5 jsou stejné

Libraries_3_5\CMSIS\CM3\DeviceSupport\ST\STM32F10x\startup\arm\startup_stm32f10x_cl.s

4. V „Source Group 1“ přidat nový soubor main.c
5. Úvodní kód:

```
#include <stm32f10x.h>
int main(void)
{
  while(1)
    ;
}
```

6. Nastavení projektu:
    a. Output – vytvořit nový adresář a do něj nasměrovat „Select Folder fo Objects …“
    b. Listing – nasměrovat do stejného adresáře jako „output“
    c. Debug – vybrat ULINK2/ME a nastavit (pozor, možné pouze při připojeném HW):
        i. Zaškrtnout SWJ
        ii. Vybrat Port SW
        iii. V záložce Flash download zaškrtnout „Reset and Run“
    d. C/C++ - vybrat „No Auto includes“ a uvést je manuálně, jinak se může stát, že použije nějaké defaultní nainstalované spolu s IDE (neznámé verze)
    e. C/C++ - přidat „Include Paths“

```
.
..\Libraries_3_5\CMSIS\CM3\CoreSupport
..\Libraries_3_5\CMSIS\CM3\DeviceSupport\ST\STM32F10x
..\Libraries_3_5\STM32F10x_StdPeriph_Driver\inc
```

    f. Přidat další „Source Group“ – např. pojmenovat „ARM-ST“ a přidat existující soubory:

```
..\Libraries_3_5\CMSIS\CM3\CoreSupport\core_cm3.c
..\Libraries_3_5\CMSIS\CM3\DeviceSupport\ST\STM32F10x\system_stm32f10x.c
```

7. Zkusit „Build“ s výsledkem:

```
Build target 'Target 1'
assembling startup_stm32f10x_cl.s...
compiling main1.c...
compiling core_cm3.c...
compiling system_stm32f10x.c...
```

```
..\Libraries_3_5\CMSIS\CM3\DeviceSupport\ST\STM32F10x\stm32f10x.h(96):
error:  #35: #error directive: "Please select first the target STM32F10x
device used in your application (in stm32f10x.h file)"
".\_output\xxxx.axf" - 1 Errors, 0 Warning(s).
Target not created
```

8. Pohledem do stm32f10x.h je zřejmé, že je třeba nastavit jeden ze symbolů popisujících typ procesoru, zde se jedná o **STM32F10X_CL**
    a. Nastavení projektu – C/C++ a do řádku „Define“ jej doplnit
    b. Vzhledem ke globálnímu nastavení je toto viditelné ve všech souborech projektu
9. Možno ověřit primitivní kód „počítadlo na LEDkách na GPIOE“:

```
#include <stm32f10x.h>

int main(void)
{
  unsigned char b = 0;

  RCC->APB2ENR   |= RCC_APB2ENR_IOPEEN;       // clock enable
  RCC->APB2RSTR  |= RCC_APB2RSTR_IOPERST;     // reset peripheral L-H
  RCC->APB2RSTR  &= ~RCC_APB2RSTR_IOPERST;    // reset H-L

  GPIOE->CRH = 0x33333333;                     // set ALL as Output

  while(1)
  {
    int i;

    GPIOE->ODR = (GPIOE->ODR & 0x00ff) | ((unsigned int)b << 8);
    b++;

    for(i = 0; i < 100000; i++)               // simple wait
      ;
  }
}
```

10. Aby bylo korektně použitelné SystemCoreClock, je defaultně nastaveno 72MHz (symbol pro to „naše“ _CL lze najít na řádku 115 v system_stm32f10x.c jako

    ```
    #define SYSCLK_FREQ_72MHz  72000000
    ```

    Pokud by bylo potřeba nastavit jiné hodiny, je možné symbol doplnit v C/C++ záložce projektu. V knihovně je to ošetřeno pomocí #if a #elif, takže se vybere buď ten nastavený nebo nakonec 72MHz. Možné konstanty lze najít ve funkci static void SetSysClock(void) v souboru system_stm32f10x.c na řádku 419.,
11. Využití přerušení – např. časovač TIM6
    a. Nastavit zdroj hodin a inicializační reset periférie

```
RCC->APB1ENR   |= RCC_APB1ENR_TIM6EN;
RCC->APB1RSTR  |=   RCC_APB1RSTR_TIM6RST;
RCC->APB1RSTR  &= ~RCC_APB1RSTR_TIM6RST;
```

    b. Nastavit režim, takt apod. pro TIM6, na závěr povolit běh časovače

```
TIM6->CR1 = 0x0000;                          // ARPE=0, URS=0, UDIS=0, CEN=0
TIM6->PSC = SystemCoreClock/1000000 - 1;    // Prescale to 1 us
TIM6->ARR = 999;          // Autoreload overflow value settings (N + 1) * 1us (1ms)
```

```
TIM6->EGR = 0x0001;      // Reinitialize and update timer, prescaler and reload reg
TIM6->CR1 |= TIM_CR1_CEN;                    // Enable timer
```

c. Povolit generování přerušení z TIM6

```
TIM6->DIER |= TIM_DIER_UIE;   // Enable TIM6 update event interrupt - irq mode
```

d. Povolit v NVIC zpracování požadavku přerušení (funkce je v core_cm3.h)

```
NVIC_EnableIRQ(TIM6_IRQn);
```

e. Napsat obslužnou funkci – musí mít název uvedený v tabulce vektorů v souboru startup_stm32f10x_cl.s

```
void  TIM6_IRQHandler(void)
{
  TIM6->SR &= ~TIM_SR_UIF;        // Clear update event interrupt flag

  {
    static unsigned char b = 0;

    GPIOE->ODR = (GPIOE->ODR & 0x00ff) | ((unsigned int)b << 8);
    b++;
  }
}
```

1. Nový projekt založený na **STM32F107VC**
    a. V „Packs“ je třeba mít nainstalovánu podporu pro STM32F1xx
    b. Je to možné jako podporu pro desku Keil MCBSTM32C
2. V nastavení Run-time je potřeba přidat Device/Startup pro STM32F1xx, jinak nic jiného
    a. V projektu kromě „Source Group 1“ bude ještě „Device“ obsahující soubory startup_stm32f10x_cl.s a startup_stm32f10x_cl.c
3. Přidat nový soubor s funkcí main (jako výše)
4. Soubory ARM knihovny přidat jako adresář

```
+-  STM32F1xx_DFP_1_0_5
|   +-- Include
|   +-- Source
|    +-- StdPeriph_Driver
+-  CMSIS_4_2_0
|   +-- Include
+-  Projekt_1
|       Projekt_1.uvproj
|      Main.c
+-  Projekt_2
       Projekt_2.uvproj
       Main.c
```

5. Keil je pomocí funkce „Pack“ nainstaloval do „Svého“ adresáře (verze se zřejmě budou měnit):

```
d:\keil\ARM\PACK\Keil\STM32F1xx_DFP\1.0.5\Device\
d:\keil\ARM\PACK\ARM\CMSIS\4.2.0\CMSIS\
```

6. Nastavení projektu
    a. Target – Use MicroLIB (pro budoucí stdio operace)
    b. C/C++ - Include Path

```
.
..\STM32F1xx_DFP_1_0_5\Include
..\STM32F1xx_DFP_1_0_5\StdPeriph_Driver\inc
..\CMSIS_4_2_0\Include
```

    c. C/C++ - Define přidat
    d. Debug – ULINK2/ME, v Settings vybrat:
        i. Debug - SWJ, Port SW
        ii. Flash Download – Reset and Run

    Případně bude chtít upgrade ULINKu – kompatibilita s