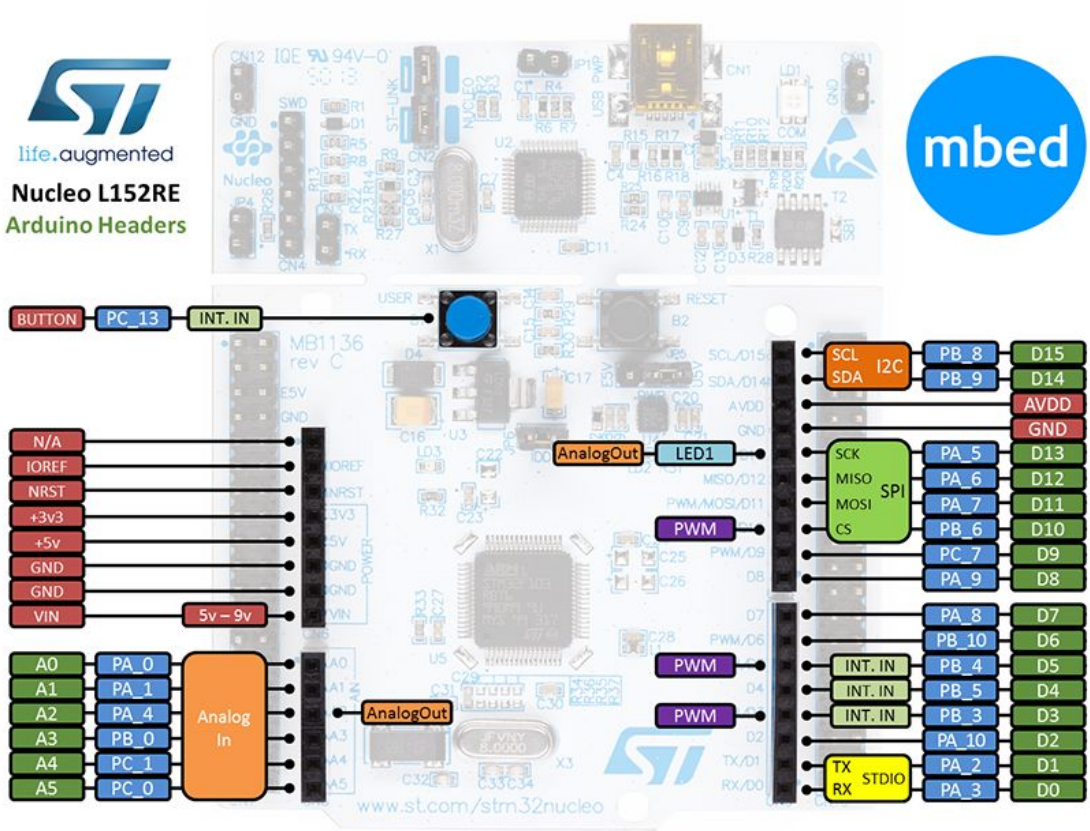
Téma : Prerušenia, UART a oddelený preklad

1. **Príprava na prácu na cvičení (max 40 minút)**

    - Vytvoriť si embedded C projekt s názvom **vrs_cv5**, pričom ako HW bude zvolená nie vývojová doska ale MCU STM32L152RE a interfejs na debugovanie bude STLink
    - V prípade prerušení s AD budete postupovať podľa zadania cvičenia č.4.
    - Keďže budete využívať princíp oddeleného prekladu bude potrebné doplniť do projektu dva nové súbory s názvom “vrs_cv5.c” a “vrs_cv5.h” nezabudnite zabezpečiť “*.h” súbor proti viacnásobnému vloženiu. Odporúčam použiť funkcionalitu IDE na vytváranie nových súborov.



Popis vývodov NUCLEO L152RE

- Nastavenie ADC bude tentokrát potebné upraviť tak, aby bol zabezpečený kontinuálny prevod. Prevod samotný bude spustený okamžite po kompletnej inicializácii ADC spolu s prerušením
- Príklad nastavenie NVIC prerušenia :

```
NVIC_PriorityGroupConfig(NVIC_PriorityGroup_0);
```

```
NVIC_InitTypeDef NVIC_InitStructure;
NVIC_InitStructure.NVIC_IRQChannel = EXTI0_IRQn; //zoznam prerušení nájdete v súbore stm32l1xx.h
NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 1;
NVIC_InitStructure.NVIC_IRQChannelSubPriority = 0;
NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
NVIC_Init(&NVIC_InitStructure);
```

- Nastavte NVIC na 16 preemption prerušení a prerušenie na ADC1
- Povolenie prerušenia na EOC a OVR v prípade ADC nastavíte pomocou funkcie :

```
void ADC_ITConfig(ADC_TypeDef* ADCx, uint16_t ADC_IT, FunctionalState NewState)
```

- Funkcia prerušenia musí mať názov taký ako je zadefinovaný v asm. Súbore **startup_stm32l1xx_hd.s**, pričom návratova hodnota bude void a taktiež nebude táto funkcia príjmať žiadne parametre.
- Na overenie zdroja prerušenia v prípade ADC použijete funkciu

```
FlagStatus ADC_GetFlagStatus(ADC_TypeDef* ADCx, uint16_t ADC_FLAG)
```

- Nastavenie USART bude prebiehať v režime UART1 8N1 9600baud, prerušenie na príjem, blokované posielanie.
- Pre našu komunikáciu budeme používať výlučne porty RX, TX, ktorým nastavíme príslušné GPIO porty v nasledujúcej konfigurácii:

```
GPIO_Mode = Alternate function (FN)
GPIO_Speed = 40MHz
GPIO_OType = Push-Pull
GPIO_PuPd = No pull
```

- Ďalej je potrebné nakonfigurovať konkrétnu AF pre **oba** GPIO porty ako UART1. Hodnoty AF nájdete v súbore **stm32l1xx_gpio.h**. Na ich nastavenie využite nasledujúcu funkciu

```
void GPIO_PinAFConfig(GPIO_TypeDef* GPIOx, uint16_t GPIO_PinSource, uint8_t GPIO_AF)
```

- Pre prerifériu usart je potrebné zapnúť hodiny a nastaviť jej štruktúru podľa uvedných informácii, pričom nepoužívame HW riadenie toku

```
RCC_APB2PeriphClockCmd(RCC_APB2Periph_USART1, ENABLE);

USART_InitStructure.USART_BaudRate = 9600;
USART_InitStructure.USART_WordLength = USART_WordLength_8b;
USART_InitStructure.USART_StopBits = USART_StopBits_1;
USART_InitStructure.USART_Parity = USART_Parity_No;
USART_InitStructure.USART_HardwareFlowControl = USART_HardwareFlowControl_None;
USART_InitStructure.USART_Mode = USART_Mode_Rx | USART_Mode_Tx;
USART_Init(USART1, &USART_InitStructure);
```

- Samozrejme pre príjem informácii je potrebné vytvoriť štruktúru prerušenia rovnakú ako pri ADC, len je potrebné zvoliť nižšiu prioritu prerušenia a nájsť rovnako funkciu pre obsluhu prerušenia USART1.
- Povolenie prerušenia je potrebné nastaviť iba pre príjem flagom RXNE, ktorý sa nastaví pomocou funkcie

```
void USART_ITConfig(USART_TypeDef* USARTx, uint16_t USART_IT, FunctionalState NewState)
```

- Samozrejme je potrebné periférium zapnúť týmto príkazom

```
void USART_Cmd(USART_TypeDef* USARTx, FunctionalState NewState)
```

- Komunikácia je zabezpečená cez funkcie

```
void USART_SendData(USART_TypeDef* USARTx, uint16_t Data)
uint16_t USART_ReceiveData(USART_TypeDef* USARTx)
```

- Pre zdroj prerušenia a rovnako aj zistenie stavu odosielania indikované flagom TC, je možné využiť funkciu

```
FlagStatus USART_GetFlagStatus(USART_TypeDef* USARTx, uint16_t USART_FLAG)
```

- Flagy TC a RXNE je potrebné po ich setnutí resetovať pre ďalšie použite. To je možné viacerými spôsobmi, tie si zistíte v refenčnom manuály USART->SR (status register). V prípade ručného resetovanie je možné využiť funkciu

```
void USART_ClearFlag(USART_TypeDef* USARTx, uint16_t USART_FLAG)
```

- Komunikácia prebehne cez prevodník USART/USB, kde budete zapájať iba na oboch stranách iba piny TX,RX (GND v tomto prípade netreba lebo sú spojené cez USB v počítači). NA sledovanie komunikácie v počítači si vyberetie v Debug perspektíve okno Termina l(v záložke View), kde si nastavíte COM port a príslušné parametre linky. Terminal umožnuje obojsmernú komunikáciu.

2. Úloha č.1 (max 30 minút)

Prerobte Vašu aplikáciu z cvičenia č.4 tak aby ste merali napätie na vstupnom kanále ADC pomocou prerušenia. Zdrojový kód aplikácie by mal byť čitateľný, zrozumiteľný v prípade potreby aj vhodne okomentovaný a mal by sa nachádzať v oddelených súborov.

3. Úloha č.2 (max 30 minút)

Vytvorte aplikáciu, ktorá bude merané dáta z ADC posielať cez UART fo formáte “3.30V” alebo vo formáte “4095”. Zmena formátu posielania dát bude vykonávaná posielaním znaku ‘m’ do MCU. Nastavenie UART1 : 9600baud 8N1. Prerušenie ADC má mať vyššiu prioritu ako USART.

4. Pokyny k odovzdávaniu projektu

Odovzdajte vypracovanú správu spolu s linkom na projekt na Github obvyklým spôsobom na zdieľaný Google Drive priečinok do 23.10.2016 23.59. Vývojové dosky bude možné v ojedinelých prípadoch zapožičať, v stredu (19.10.2016) výlučne v čase 14:00 až 14:30 a odovzdanie v piatok (21.10.2016) výlučne v čase 10:00 až 10:30. Na zapožičanie je k dispozícii iba 15ks vývojových dosiek.