ISR, tlačítka se zákmity, časovač SysTick

1 Zadání

- Vytvořte nový projekt, nakonfigurujte LED1 a LED2 jako výstup, S1 a S2 jako vstup s interním pullupem. Povolte přerušení spouštěné sestupnou hranou na EXTI0 (tlačítko S2), vytvořte ISR, které provede negaci LED2 **(0,5b)**.
- Nakonfigurujte časovač SysTick na periodu 1ms. Vytvořte projekt, který bude blikat LED1 s periodou 300ms. Využijte časování založené na časovači SysTick. Obsluhu LED1 napište do funkce blikac(), tuto funkci volejte z main() **(0,5b)**.
- Vytvořte neblokující obsluhu tlačítek, která je bude číst s periodou 40 ms. Následně v obsluze zjišťujte stav tlačítek S1 a S2. Stav LED2 bude obsluhován na základě události tlačítka – po stisku S2 se rozsvítí na 100ms, po stisku S1 na 1s. Obsluhu napište do funkce tlacitka(), její volání přidejte do main() **(1b)**.
- Obsluhu S1 upravte tak, aby se tlačítko vzorkovalo s periodou 5 ms. Stav tlačítka S1 shiftujte pomocí bitového posuvu vlevo do statické 16-bitové proměnné, o rozsvícení LED2 rozhodněte na základě stavu 0x7FFF, který udává rozepnutí tlačítka s dozněním zákmitů **(1b)**.
- Hodnocena bude i úprava zdrojového kódu, zejména odsazování bloků. V modulu main.c nesmí být žádné globální proměnné kromě Tick, kód main()u smí v hlavní nekonečné smyčce pouze volat úlohy blikac() a tlacitka().

2 Návod

2.1 Základní seznámení

- Vytvořte si pracovní kopii svého repozitáře z Githubu (Git Clone), příp. aktualizujte repozitář ze serveru (Git Pull).
- Základní strukturu můžete využít (tj. okopírovat) z prvního cvičení, příp. lze založit nový projekt stejným postupem a přidat knihovny CMSIS. Po okopírování projektu je potřeba jej přidat do STM32CubeIDE pomocí File / Import / General / Existing Projects into Workspace. **Každý dokončený bod commitujte.**
- Budeme používat rozšiřující modul STMrel, zapojení potřebných periferií:
  - LED1 (vlevo) = PA4
  - LED2 (vpravo) = PB0
  - S1 (vpravo) = PC1
  - S2 (vlevo) = PC0 (EXTI0)
- Piny připojené k LED nastavte jako výstupní, u pinů k tlačítkům aktivujte pullup. Nezapomeňte povolit hodiny:

```
RCC->AHBENR |= RCC_AHBENR_GPIOAEN | RCC_.............. | RCC_..............; // enable
GPIOA->MODER |= GPIO_MODER_MODER4_0; // LED1 = PA4, output
GPIOB->MODER |= GPIO_..............; // LED2 = PB0, output
GPIOC->PUPDR |= GPIO_PUPDR_PUPDR0_0; // S2 = PC0, pullup
GPIOC->PUPDR |= GPIO_..............; // S1 = PC1, pullup
```

2.2 Externí přerušení

- Inspiraci ke zpracování externích přerušení lze najít v příkladu ExternalIT/01_HWInterruptSelection v balíku STM32SnippetsF0.
- Je nezbytné povolit hodiny pro SYSCFG:

```
RCC->APB2ENR |= RCC_APB2ENR_SYSCFGEN;
```

- Dále vybrat zdroj přerušení, povolit maskou a zvolit aktivní hranu. Nakonec přerušení aktivovat:

```
SYSCFG->EXTICR[0] |= SYSCFG_EXTICR1_EXTI0_PC; // select PC0 for EXTI0
EXTI->IMR |= EXTI_IMR_MR0; // mask
EXTI->FTSR |= EXTI_FTSR_TR0; // trigger on falling edge
NVIC_EnableIRQ(EXTI0_1_IRQn); // enable EXTI0_1
```

- O událost přerušení se bude starat obsluha, která po zkontrolování zdroje vymaže pending bit a provede akci, v našem případě negování stavu LED2, tj. pinu PB0:

```
void EXTI0_1_IRQHandler(void)
{
  if (EXTI->PR & EXTI_PR_PR0) { // check line 0 has triggered the IT
    EXTI->PR |= EXTI_PR_PR0; // clear the pending bit
    GPIOB->............;
  }
}
```

- Nekonečná smyčka ve funkci main() bude prázdná. Dokončený bod commitujte.

2.3 Časovač SysTick a neblokující čekání

- Časovač SysTick budeme používat pro časovou základnu, všechny úlohy (funkce) v supersmyčce main()u budou neblokující.

Stavový automat jako RTOS

#2

ÚREL
ÚSTAV RADIOELEKTRONIKY

- použití neblokujícího časovače (300 ms zap., 200 ms vyp.)

```
void blikatko(void)
{
  static enum { LED_OFF, LED_ON } stav;
  static uint32_t DelayCnt = 0;

  if (stav == LED_ON) {
    if (Tick > (DelayCnt + 300)) {
      GPIOD->BRR = (1<<0);
      DelayCnt = Tick;
      stav = LED_OFF;
    }
  } else {
    if (Tick > (DelayCnt + 200)) {
      GPIOD->BSRR = (1<<0);
      DelayCnt = Tick;
      stav = LED_ON;
    }
  }
}
```

časování přes SysTick

```
volatile uint32_t Tick;
void SysTick_Handler(void)
{
  Tick++; // perioda 1ms
}
```

voláno z funkce main()

```
while (1) {
  blikatko();
  // dalsi funkce
  // ktere neblokuji
}
```

- Na úrovní modulu vytvoříme globální uint32_t proměnnou Tick, vzhledem k přístupu z ISR i hlavního kódu ji označíme jako volatilní.
- V úvodní části main()u inicializujeme SysTick časovač, systémové hodiny jsou 8MHz a chceme periodu 1ms:

```
SysTick_Config(8000); // 1ms
```

- Vytvoříme obsluhu události časovače SysTick::

```
void SysTick_Handler(void)
{
  Tick++;
}
```

- Úloha blikac() bude jednoduše neblokujícím způsobem řešit obsluhu LED1, tj. její negaci. Symbol LED_TIME_BLINK definujte na hodnotu 300.

```
void blikac(void)
{
  static uint32_t delay;

  if (Tick > delay + LED_TIME_BLINK) {
    GPIOA->ODR .........;
    delay = Tick;
  }
}
```

2.4 Obsluha tlačítek

- Vytvořte funkci tlacitka(). Funkce bude podobně jako blikač využívat neblokující čekání, každých 40ms navzorkuje piny připojené k tlačítkům a v případě jejich stisku na definovanou dobu rozsvítí LED2. Vzor čtení S2, které rozsvítí LED2 na dobu LED_TIME_SHORT, tj. 100ms:

```
static uint32_t old_s2;
uint32_t new_s2 = GPIOC->IDR & (1<<0);

if (old_s2 && !new_s2) { // falling edge
  off_time = Tick + LED_TIME_SHORT;
  GPIOB->BSRR = (1<<0);
}
old_s2 = new_s2;
```

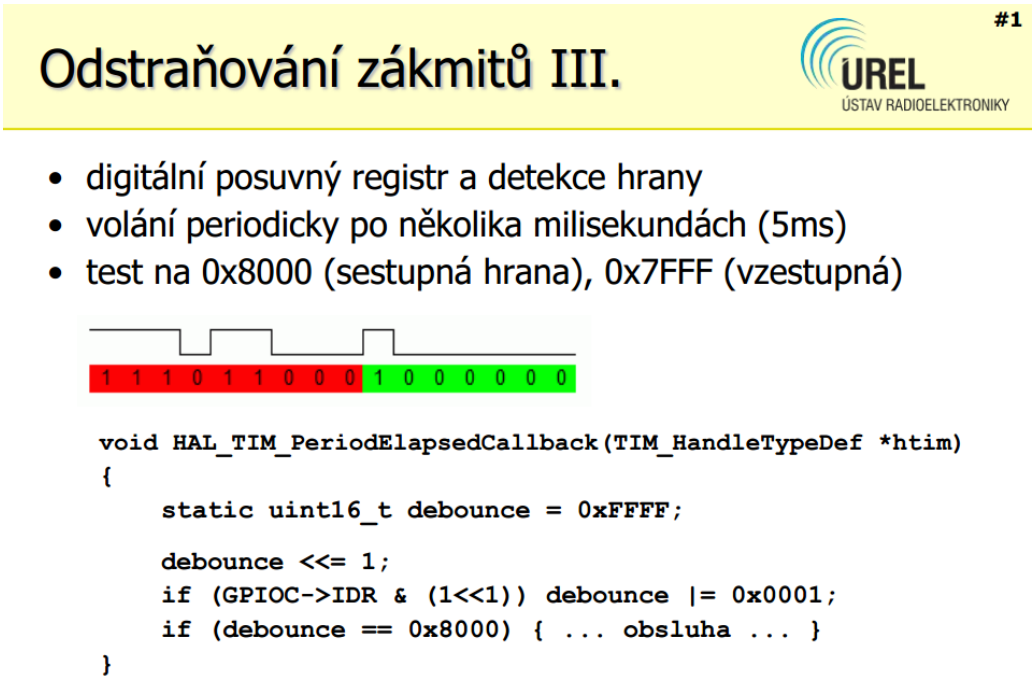
- O zhasnutí LED se postará podmínka, testující zda již uplynul definovaný čas:

```
static uint32_t off_time;

if (Tick > off_time) {
  GPIOB->BRR = (1<<0);
}
```

- Kód doplňte o obsluhu tlačítka S1, které LED2 rozsvítí na dobu LED_TIME_LONG, tj. 1000ms.
- Proveďte commit pracovní kopie do Gitu.
- Základní odstranění zákmitů jednoduchým čtením po definovaném čase (zde 40 ms) se používá pro nenáročné aplikace, typicky pro vstupy uživatelského rozhraní, kde výjimečné zakmitání není na škodu.

2.5 Pokročilé odstranění zákmitů



- Upravte funkci tlacitka() tak, aby obsahovala část volanou s periodou 5ms. Zadefinujte si statickou proměnnou:

```
static uint16_t debounce = 0xFFFF;
```

- Každých 5ms proveďte bitovou rotaci vlevo a do nejnižšího bitu (LSB) nastavte 1 v případě, že je příslušný GPIO pin tlačítka S1 v log. 1.
- Testujte proměnnou na hodnotu 0x7FFF, v případě shody na definovanou dobu rozsviťte LED2. Hodnota 0x7FFF udává, že tlačítko bylo sepnuté (MSB je nulový) a následně bylo bez zákmitů rozepnuté po dobu min. 15x 5ms = 75ms. Tato metoda odstranění zákmitů se používá pro výrazně zakmitávající tlačítka, případně pro vstupy, kde zakmitání způsobí zásadní problém, který nemůže být snadno napraven zpětnou vazbou od uživatele.
- Proveďte commit pracovní kopie do Gitu, uložte repozitář pomocí Git Push.