**VRS 2. Cvičenie – Zoltán Czinege, Tomáš Nagy**

1. **Úloha**

Na pine PA5 máme pripojenú zelenú LED. Na to aby sme túto LED mohli ovládať potrebujeme nastaviť príslušnú GPIO perifériu a jej príslušný port následovne:

GPIO mode => OUT

Output type => PushPull

PullUp PullDown => UP

GPIO speed => 40MHz

Riešenie:

RCC\_AHBPeriphClockCmd(RCC\_AHBPeriph\_GPIOA, *ENABLE*);

GPIOA->MODER |= (0b01) << (5\*2);

GPIOA->OTYPER &= ~((uint16\_t)(1<<5));

GPIOA->PUPDR |= (0b01) << (5\*2);

GPIOA->OSPEEDR |= (0b11) << (5\*2);

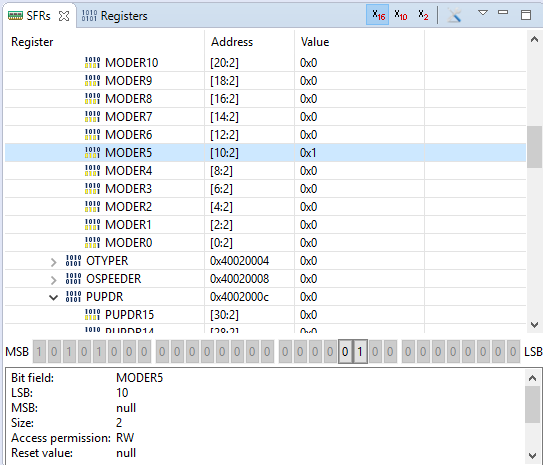
Samozrejme, že LED budeme chcieť aj zapnúť, vypnúť alebo prepnuť jej stav. Na toto sa dá použiť buď prístup pomocou dátoveho registru ODR, alebo register pre atomárny prístup BSRR, v prípade definíce v stm32l1xx.h je tento rozdelený na dve časti BSRRL - tu sa nastavuje pin a BSRRH - tu sa nuluje pin.

/\* zapinanie a vypinanie \*/

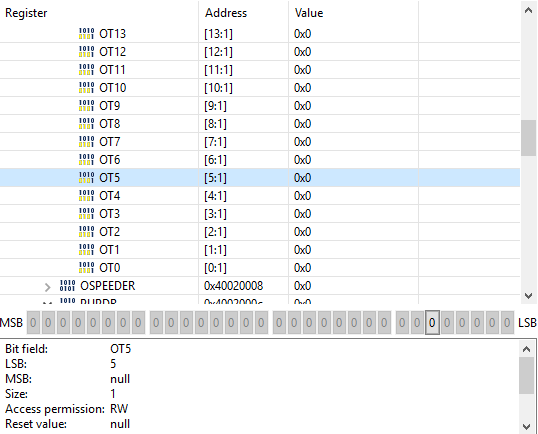
GPIOA->ODR |= 0b0000000000100000;

GPIOA->BSRRL |= (uint16\_t)(1<<5); //set

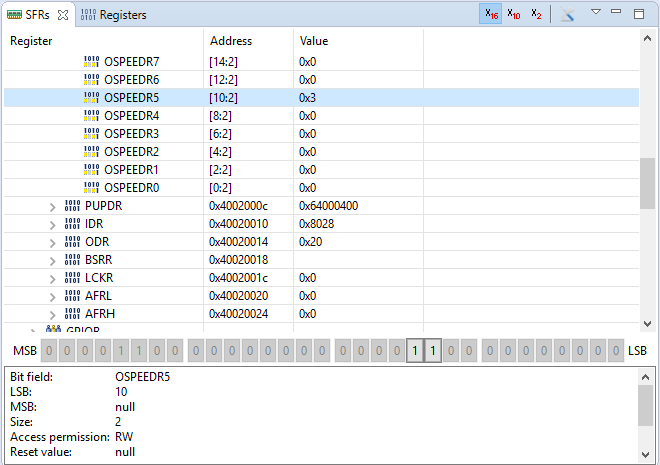
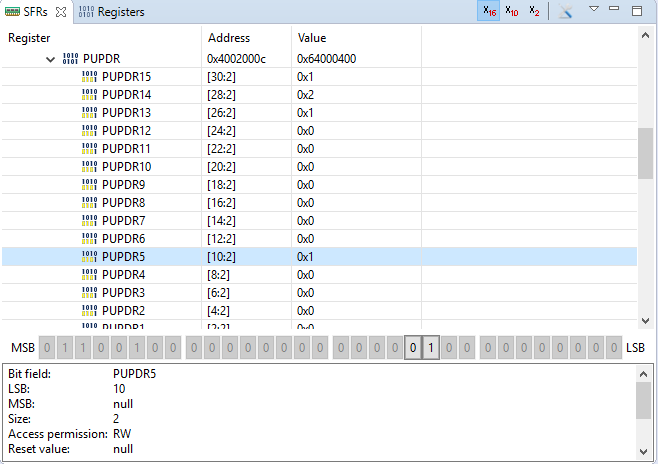
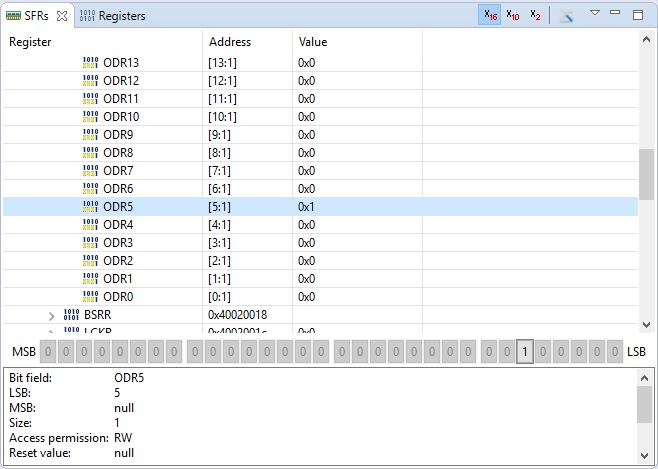
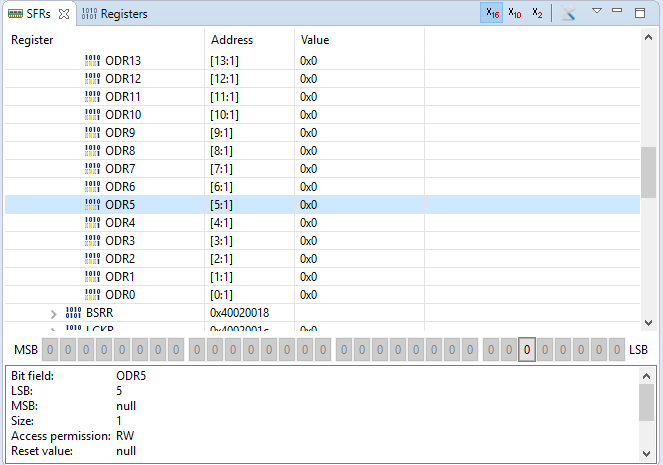
GPIOA->ODR &= ~(0b0000000000100000);

GPIOA->BSRRH |= (uint16\_t)(1<<5); //reset

GPIOA->MODER



GPIOA->OTYPER



GPIOA->ODR vypnutý

GPIOA->ODR zapnutý

GPIOA->PUPDR

GPIOA->OSPEEDR

BSSR

1. **Úloha**

Na pine PC13 máme pripojené tlačidlo, ktorého stavy chceme snímať. Nastavenia periférie bude v tomto prípade následovné:

GPIO mode => IN

Output type => PushPull

PullUp PullDown => NOPULL

Riešenie:

RCC\_AHBPeriphClockCmd(RCC\_AHBPeriph\_GPIOC, *ENABLE*);

GPIOC->MODER |= (0b00) << (13\*2);

GPIOC->OTYPER &= ~((uint16\_t)(1<<13));

GPIOC->PUPDR |= (0b00) << (13\*2);

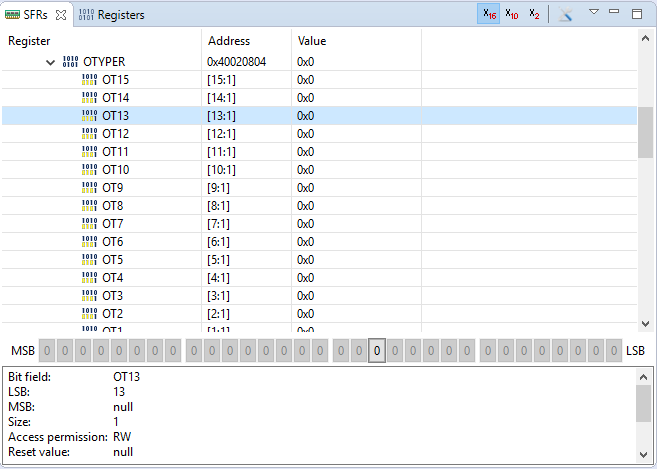
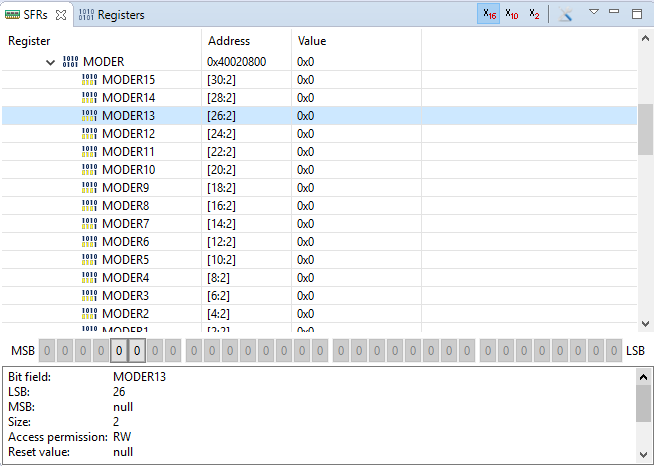
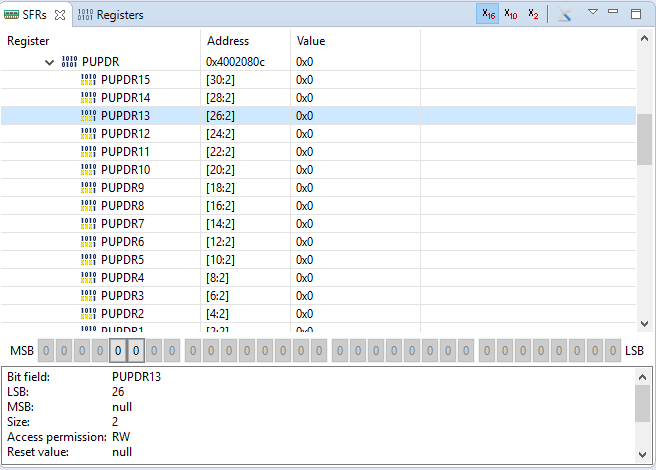
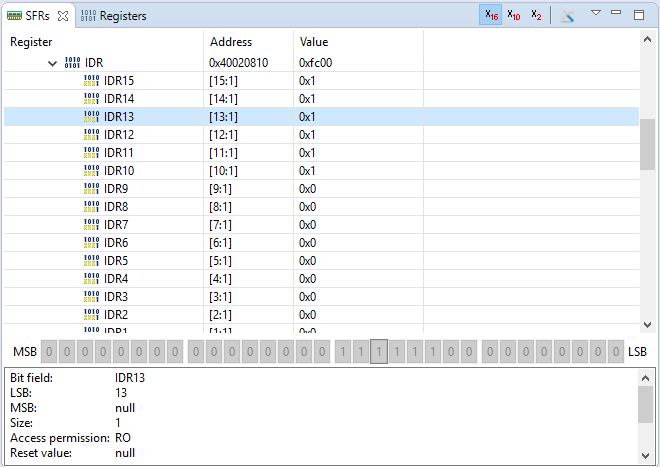
**int** button;

/\* nacitavanie s IDR v cykle while \*/

**if** ((GPIOC->IDR & 0b0010000000000000) == 0)

button = 1;

**else** **if** ((GPIOC->IDR & 0b0010000000000000) == 8192)

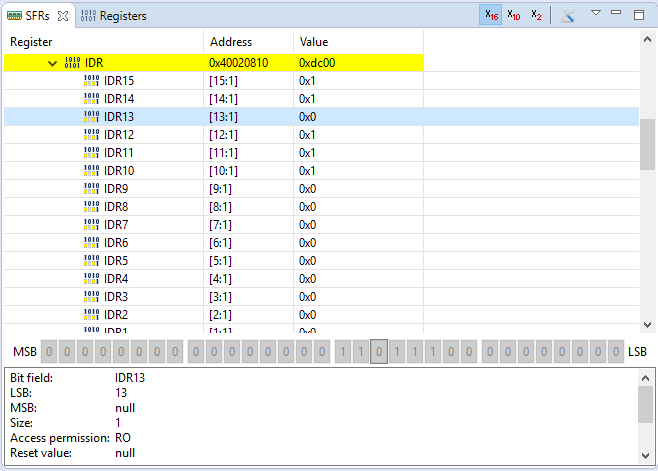
 button = 0;

GPIOC->IDR – button=0

GPIOC->PUPDR

GPIOC->MODER

GPIOC->OTYPER



GPIOC->IDR - button=1

1. **Úloha**

* Vytvorte program, v ktorom bude blikať LED vo Vami vybranom intervale, tak aby bolo blikanie viditeľné. Na časovanie môžete použiť for cyklus.

/\* blikanie s intervalom uloha 3 prva v cykle while\*/

GPIOA->BSRRL |= (uint16\_t)(1<<5); //set

c=1, d=1;

**for** (c=1; c<= 1000; c++){

**for** (d=1; d<= 1000; d++)

{}

}

GPIOA->BSRRH |= (uint16\_t)(1<<5); //reset

c=1, d=1;

**for** (c=1; c<= 1000; c++){

**for** (d=1; d<= 1000; d++)

{}

}

* Vytvorte program, ktorý bude sledovať stav tlačidla a tento stav bude zobrazovať na LED.

/\* stav tlacidla na led uloha 3 druha v cykle while \*/

**if** ((GPIOC->IDR & 0b0010000000000000) == 0){

GPIOA->BSRRL |= (uint16\_t)(1<<5);

}

**else** **if** ((GPIOC->IDR & 0b0010000000000000) == 8192){

GPIOA->BSRRH |= (uint16\_t)(1<<5);

}

* Vytvorte program, ktorý po každom stlačení tlačidla zmení stav LED (zo svieti na nesvieti a naopak). Nezabudnite, že stlačenie tlačidla znamená, že tlačidlo prejde stavmi z 0 - 1 - 0. Taktiež si je dobre uvedomiť, že počas stlačenia tlačidla nastávajú prechodové deje zákmitu kontaktu a preto treba ošetriť aj tento problém. Google je tu pre Vás.

/\* uloha 3 tretia cast v cykle while \*/

**for** (**int** cc=1; cc<= 5; cc++){

**if** ((GPIOC->IDR & 0b0010000000000000) == 0){

temp\_inc++;

}

}

**if** (temp\_inc > 0){

inc++;

temp\_inc = 0;

}

**if** (inc == 1)

GPIOA->BSRRL |= (uint16\_t)(1<<5);

**else**{

GPIOA->BSRRH |= (uint16\_t)(1<<5);

inc = 0;

}