SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

PROJEKT K PREDMETU VNORENÉ RIADIACE SYSTÉMY

Ústav: Robotika a kybernetika

Dňa: Bratislava 3.11.2016

Vypracoval: Peter Kmeť, Miroslav Kohút

Cvičenie: Streda 15:00

Úloha 1: Ovládanie LED PA5, nastavenie GPIO periférie a príslušného portu.

Podľa zadania sme aktivovali Clock na GPIO perifériach:

```
RCC_AHBPeriphClockCmd(RCC_AHBPeriph_GPIOA, ENABLE);
RCC_AHBPeriphClockCmd(RCC_AHBPeriph_GPIOC, ENABLE);
```

Nastavenie GPIO mode => OUT

```
GPIOA->MODER |= (1<<10);
GPIOA->MODER &= ~(uint32_t)(1<<11);
```

Nastavenie Output type => PushPull

```
GPIOA->OTYPER &= ~(uint32_t)(1<<5);
```

Nastavenie PullUp PullDown => UP

```
GPIOA->PUPDR |= (1<<10);
GPIOA->PUPDR &= ~(uint32_t)(1<<11);
```

Nastavenie GPIO speed => Very high speed

```
GPIOA->OSPEEDR |= (1<<10);
GPIOA->OSPEEDR |= (1<<11);
```

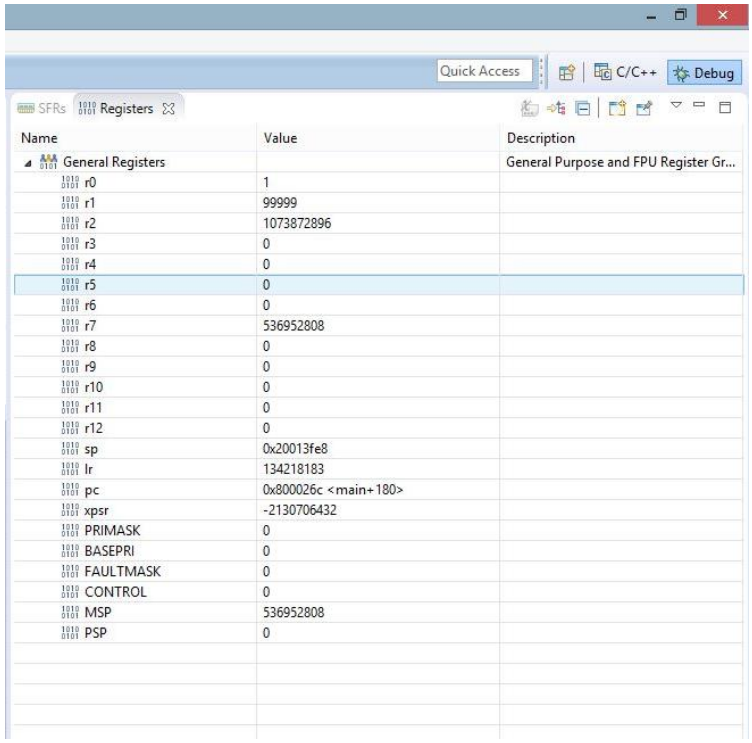
Nastavenie konkrétneho registra sme spravili presne pre konkrétny bit pomocou bitových operácií, aby sme neovplyvnili iné hodnoty registra.

Nastavenie LED bliká použitím všetkých troch možností BSRR ODR a aj toggle ODR.

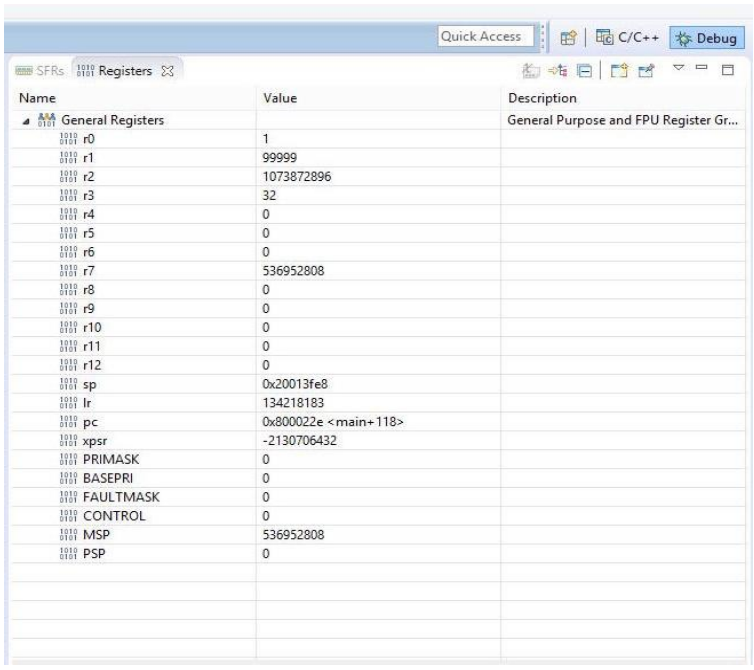
```
//ODR control
GPIOA->ODR |= (1 << 5);
for(i = 0; i<hranica;i++){
}
GPIOA->ODR &= ~(uint32_t)(1<<5);
for(i = 0; i<hranica;i++){
}

//BSRR control
GPIOA->BSRRH |= (1 << 5);
for(i = 0; i<100000;i++);
GPIOA->BSRRL |= (1 << 5);
for(i = 0; i<100000;i++);
//ODR XOR control

GPIOA->ODR ^= (1 << 5);
for(i = 0; i<100000;i++);
```



Obr.1 Stav registrov LED nesvieti



Obr.2 Stav registrov: LED svieti

Úloha 2. Nastavenie Pinu PC13 ako tlačidlo, snímanie stavu stlačenia

Nastavenie GPIO mode => IN

```
GPIOC->MODER &= ~(uint32_t)(1<<26);
GPIOC->MODER &= ~(uint32_t)(1<<27);
```

Nastavenie Output type => PushPull

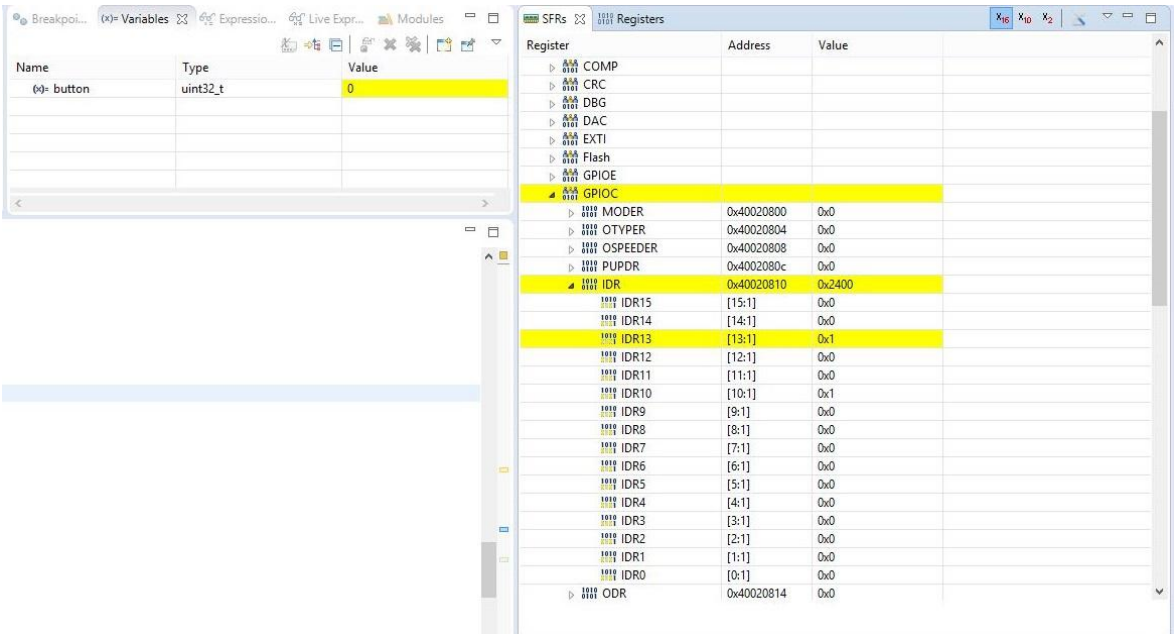
```
GPIOC->OTYPER &= ~(uint32_t)(1<<13);
```

Nastavenie PullUp PullDown => NOPULL

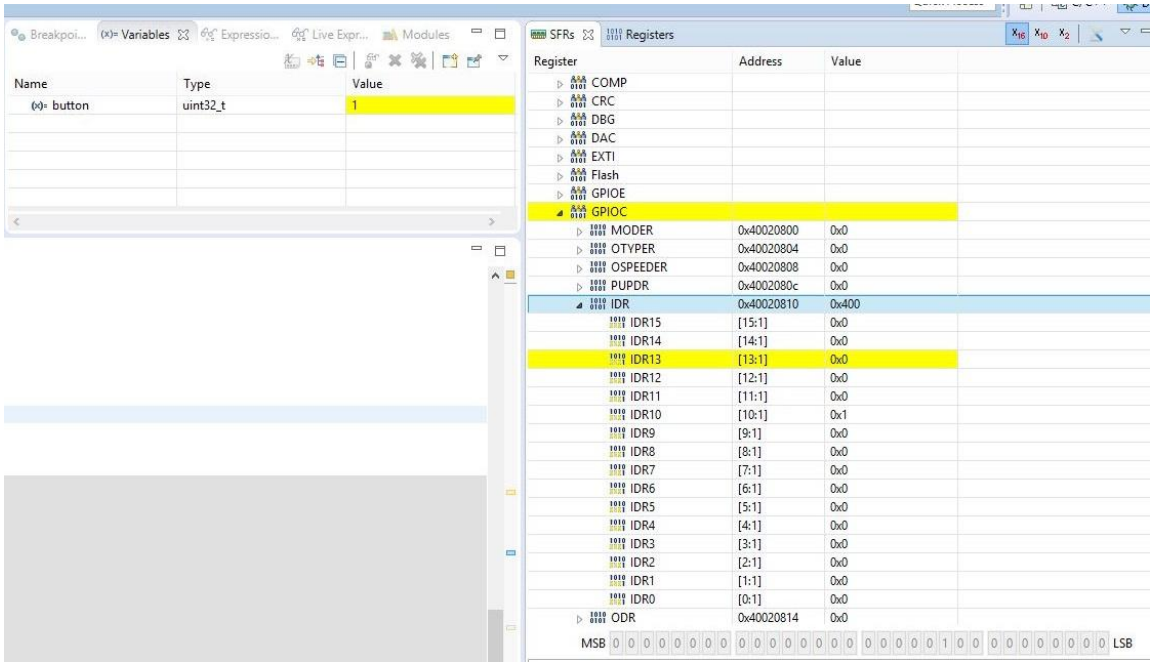
```
GPIOC->PUPDR &= ~(uint32_t)(1<<26);
GPIOC->PUPDR &= ~(uint32_t)(1<<27);
```

Prepínanie premennej pre Button

```
BUTTON = !(GPIOC->IDR & (uint32_t)(1<<13));
```



Obr. 3. Stav registrov: Nestlačené tlačidlo



Obr. 4. Stav registrov: Stlačené tlačidlo

Úloha 3:

Rozdelili sme ju na 3 časti jednotlivé časti kódu môžete nájsť v commitoch na githube.

Úloha 3.1

Ledka bliká delay je nastavený for cyklom.

```
uint64_t hranica = 500000;
GPIOA->ODR |= (1 << 5);
for(i = 0; i<hranica;i++){
}
GPIOA->ODR &= ~(uint32_t)(1<<5);
for(i = 0; i<hranica;i++){
}
```

Úloha 3.2

Program sleduje tlačidlo a zobrazuje ho na LED plus filtrovanie prekmitov.

Funkcia na filtrovanie

```
uint32_t button_control(){

	while(sum_good_values < 20){
		button_current = !(GPIOC->IDR & (uint32_t)(1<<13));
		if(button_previous == button_current)
			sum_good_values++;
		else
```

```
                sum_good_values = 0;
            button_previous = !(GPIOC->IDR & (uint32_t)(1<<13));
            for(i = 0; i<10;i++);
      }
      sum_good_values =0;
      return button_current;
}
```

Blikanie led

```
if(button_control())
      GPIOA->ODR |= (1 << 5);

else
      GPIOA->ODR &= ~(uint32_t)(1<<5);
```

Úloha 3.3

Kontrola nábežnej hrany a zmena stavu ledky stlačením tlačidla.

```
if(button_control() && !hrana_check){
      GPIOA->ODR ^= (1 << 5);
      hrana_check = 1;
}
else if(!button_control()&& hrana_check){
      hrana_check = 0;
}
```

Záver:

Úspešne sa nám podarilo splniť zadanie a zariadenie sa správalo presne tak ako bolo požadované. Taktiež sme naprogramovali funkciu, ktorá filtruje prekmity pri zmene stavu tlačidla a v programe sme detekovali nábežnú hranu pri stláčaní tlačidla. Registre boli nastavené bez vplyvu na ostatné periférie.