Stavový registr MCU AVR, větvení programu, podprogramy, funkce zásobníku, obsluha a typy přerušení

Stavový registr (PSW)

- Skládá se z jednotlivých bitů, každý má svůj specifický význam
- Příznaky (flag) vycházejí z výsledků ALU, ale mohou být nastavovány i přímo instrukcí

C **Carry Flag** – přenos z nejvyššího bitu
Z **Zero Flag** – výsledek roven nule
N **Negative Flag** – výsledek je záporný
V **Indikátor přetečení** ve dvojkovém doplňku
S N ⊕ V – test na proměnnou se znaménkem
H Half Carry Flag – poloviční přenos (přenosu nebo výpůjčka z 3. bitu)
T Transfer bit používaný instrukcemi BLD a BST (bit load, store)
I **Global Interrupt Enable**/Disable Flag – povolení přerušení

- Větvení programu probíhá pomocí instrukcí podmíněného skoku na základě stavu příslušného bitu stavového registru

Podprogramy

- Podprogram (subrutina) je část kódu, která se volá opakovaně v hlavní úloze, nebo při zavolání obsluhy přerušení
- Lze využít pro rekurzivní volání podprogramu (sebe opakování dílčí úlohy řešené stejným způsobem, stav jednotlivých kroků rekurze se ukládají do zásobníku)
- **K uložení rozpracované úlohy** před skokem do podprogramu se používá **zásobník**, jako první se uloží návratová adresa (PC + 1), na které mikrokontrolér pokračuje v programu bezprostředně po návratu z podprogramu

Instrukce volání podprogramu

Mnemonic	Description
call	long call to subroutine
icall	indirect call to subroutine
rcall	relative call to subroutine
ret	return from subroutine

Uložení registrů

- Když je vyvolán podprogram, může dojít ke změně hodnot pracovních registrů v hlavním programu, které mohou být později potřeba
- Na začátku podprogramu je nutno tyto registry přesunout do zásobníku a na konci znovu načíst

Zásobník

- **Slouží k uložení rozpracované úlohy**, např. před skokem do podprogramu

Ukazatel zásobníku – Stack Pointer

- Speciální 16bitový registr rozdělený na horní (SPH) a spodní (SPL) bity registr v I / O části paměti v SRAM, označované jako Stack
- Používá se **k dočasnému uložení hodnot registrů a návratových adres v PC při volání podprogramů, nebo obsluze přerušení**
- Obvykle začíná na konci SRAM a roste při ukládání dat z vyšších na nižší hodnoty adres
- Vždy ukazuje na vrchol zásobníku
- Inicializace ukazatele zásobníku - ukazatel zásobníku se při zapnutí inicializuje na poslední adresu SRAM, u starších uP musí být nastaven ručně na začátku libovolného programu

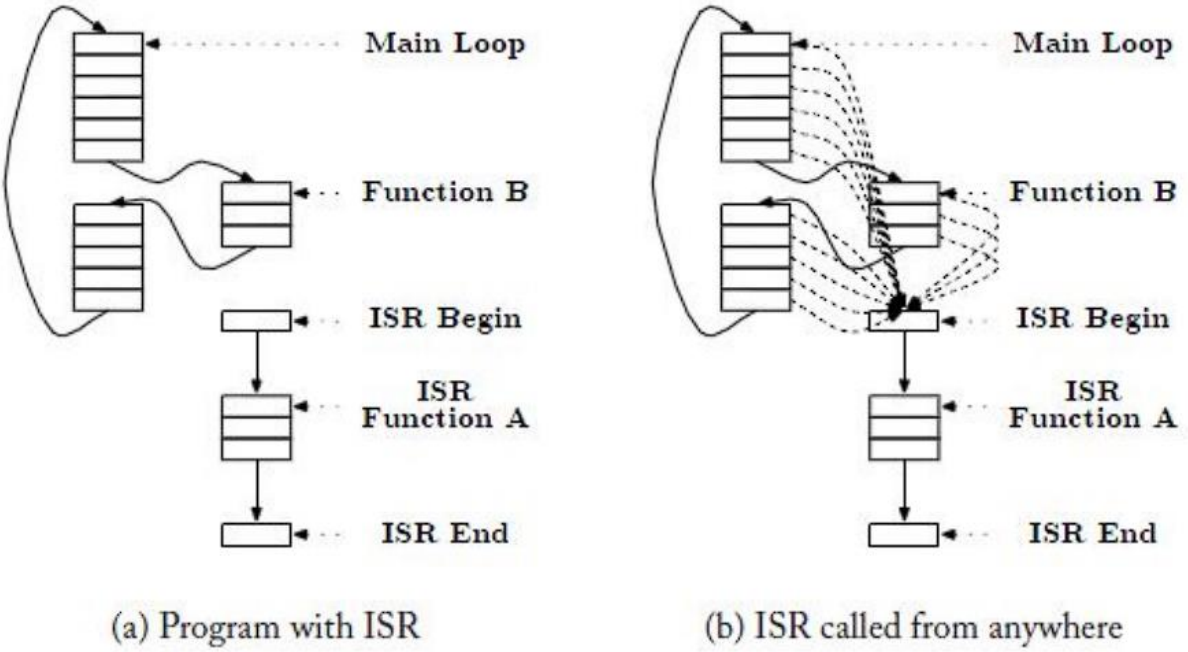
Uložení dat do zásobníku

- Registry jsou uloženy nebo načteny do zásobníku operací push nebo pop
  - **push** r0 - push r0 do zásobníku
  - **pop** r0 - restore r0 ze zásobníku
- Při volání pop s registrem se tento registr načte s obsahem na vrcholu zásobníku. Ukazatel zásobníku se následně automaticky zvýší.
- Přesunutí registru do zásobníku (PUSH) nevymaže hodnotu registru, jednoduše zkopíruje jeho obsah do SRAM, podobně výběr hodnoty ze zásobníku nevymaže obsah na této adrese v zásobníku.
- Při ukládání více registrů do zásobníku pomocí PUSH musí být zpět volány pomocí POP v opačném pořadí, aby se obnovily hodnoty do původních registrů (princip **LIFO**)
- Paměť zásobníku není nekonečná – může přetéct

Přerušení *viz otázka č.16*

- **Zpracování neplánované události**, která se může vyskytnout uvnitř nebo vně mikrokontroléru, přitom nevíme, kdy k události dojde
- Při žádosti o přerušení mikrokontrolér dokončí instrukci a (pokud je povoleno globální přerušení v PSW) přejde na adresu obsluhy přerušení (Interrupt Service Routine - ISR) uloženou v paměti na adrese odpovídajícího vektoru přerušení (podle typu žádosti)
- Po provedení obsluhy přerušení se PC nastaví na další adresu v hlavním programu (v obsluze přerušení je nutno nejprve uložit a pak znovu načíst ze zásobníku obsahy pracovních registrů, které mají být zachovány po návratu z obsluhy přerušení)

Sekvenční zpracování a přerušení



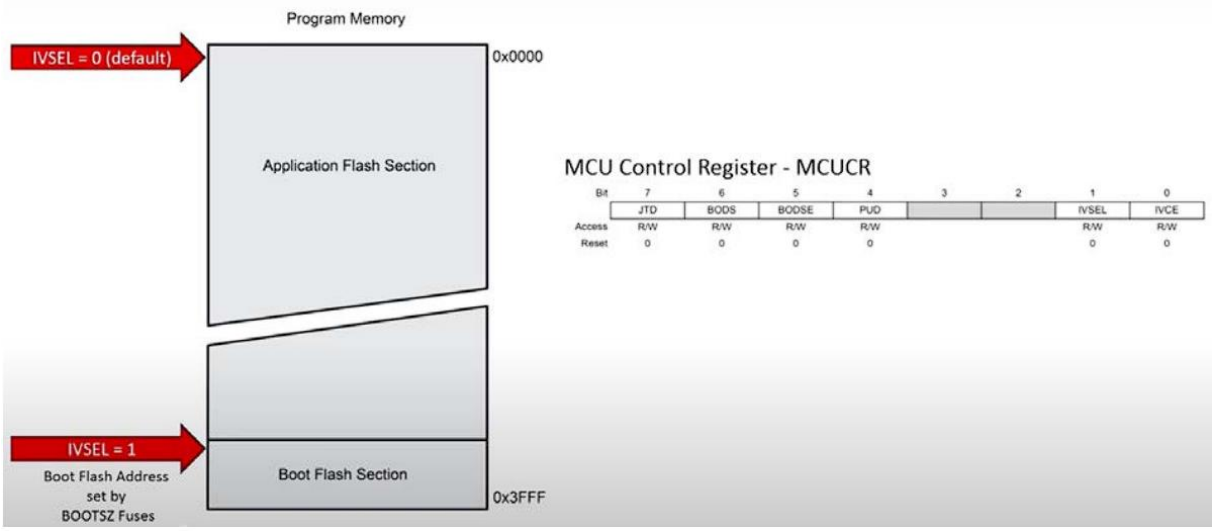
(a) Program with ISR (b) ISR called from anywhere

Důvody žádostí o přerušení

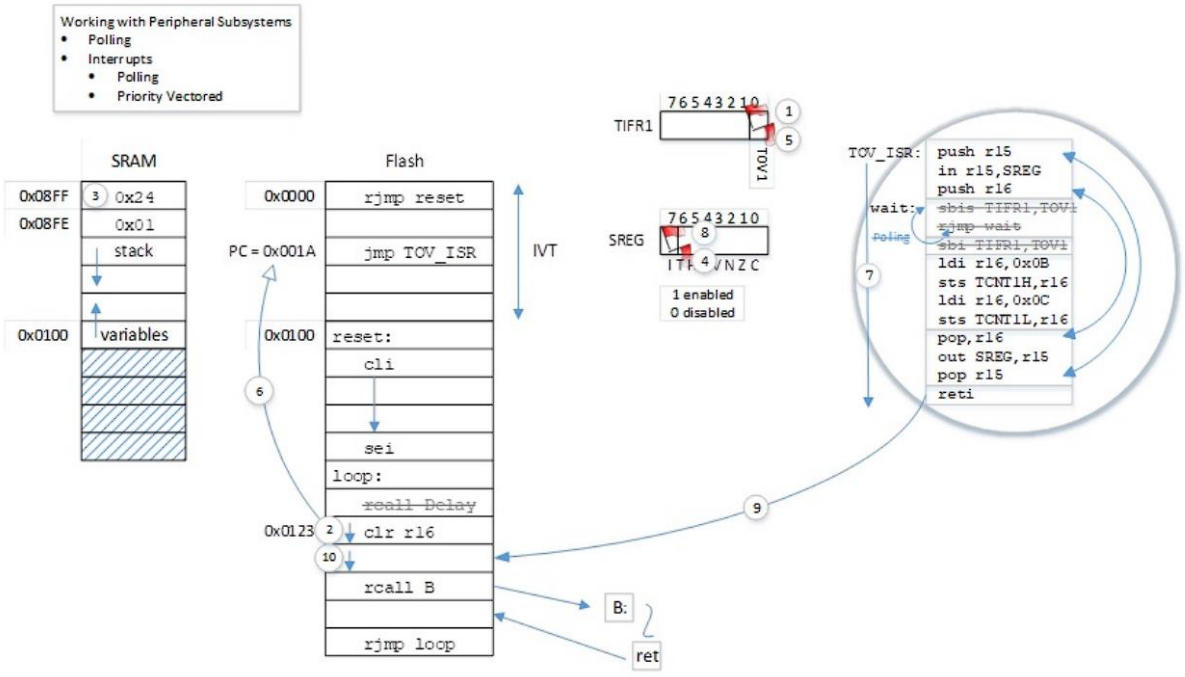
- Externí - detekce změn pinů (např. stisknutí tlačítek)
- Časovač hlídacího psa (např. Pokud se po 8 sekundách nic nestane, přerušte mě) - **Watchdog timer**
- Přerušení časovače - slouží k porovnání / přetečení časovačů • přenosy dat na sběrnicích (SPI, I2C, USART)
- A/D převody

Tabulka vektorů přerušení

- 25 různých zdrojů přerušení určených samostatným programovým vektorem
- Vektory přerušení jsou umístěny na nejnižších adresách v paměťovém prostoru programu Flash
- Adresa vektoru přerušení určuje úroveň priority přerušení. Nižší adresa má vyšší úroveň priority, RESET má nejvyšší prioritu a jeho vektor je na adrese nula (nebo adrese bootloaderu podle nastavení pojistky v bitu IVSEL řídicího registru MCUCR, používá se pro bootloader např. při programování pomocí USB)



Vektor přerušení



1. Nastane žádost o přerušení (např. TIFR1 - Bit 0 – TOV1)
2. Mikrokontrolér dokončí aktuální instrukci (clr r16 na adrese 0x123)
3. Uloží adresu další instrukce do zásobníku (0x24, 0x01)
4. Vypne systém přerušení, aby zabránil dalším přerušení pomocí vynulování I bitu (7bit - Global Interrupt Enable) ve stavovém registru SREG.
5. I-bit (příznak přerušení) je vynulován (pouze u přerušení typu 1)
6. Provedení ISR se provádí načtením počáteční adresy programu obsluhy přerušení do čítače instrukcí (0x001A) a spustí se obsluha přerušení
7. Obsluha ISR pokračuje, dokud nenastane návrat z instrukce přerušení – reti
8. I-bit SREG se automaticky nastaví, když se provede instrukce reti (tj. Interrupts enabled)

9. Když AVR vystoupí z přerušení, vrátí se k přerušenému programu
10. Provede zjištění dalšího nevyřízeného přerušení

Zásady psaní rutiny přerušení

- **Zpracovat obsluhu přerušení co nejrychleji** (nepoužít časovací smyčky uvnitř ISR, nepoužívat funkce, které čekají na nějakou další událost, ošetřit proměnné nesdílené s hlavním programem)

Typy přerušení

- Typ 1
  - Událost si **pamatuje**, když je deaktivováno přerušení
  - Pokud přerušení není povoleno, je nastaven příznak
  - Když je přerušení znovu povoleno, dojde k přerušení a příznak je resetován
- Typ 2
  - Událost si **nepamatuje**, když je deaktivováno přerušení
  - Žádost je ignorována