Микропроцесорски системи

6. блок: Управљање прекидима

др Срђан Митровић, дипл. инж.
e-mail: srdjan.mitrovic@va.mod.gov.rs

зимски семестар, 2016./2017. год.

Прекиди

Микроконтролери се примењују у системима код којих се очекује да реагују на одређење спољашње догађаје. Тип реакције може бити једноставан, нпр. инкрементирање бројача, али и захтеван, као што је искључивање напајања када човек уђе у опасан радни простор неког уређаја или машине.
Ако препостaвимо да микроконтролер може надгледати догађај, поставља се питање на који начин треба то надгледање спровести:

- периодично (континуирано) проверавати стање (*polling*)
  - непотребно трошење процесорског времена, јер се ови догађаји дешавају ретко,
  - контролер има "друга посла"
  - тешко је модификовати кôд
- користити механизам прекида (*interrupts*)
  - главни програм се прекида само када дође до промене
  - иначе се главни програм извршава без утицаја прекида

Механизам прекида

Микроконлери имају механизам који омогућава извршавање главног програма независно од надгледања прекида. Прекиде надгледа посебан механизам, који прекида извршавање главног програма тек када се прекид деси и рри томе позива одговарајућу ISR (*Interrupt Service Routine*). Проблем дешавања више прекида у исто време се решава одређивањем њиховог приоритета.

Joe Hobbyist, stumped on his timing loops.

Управљање прекидима

Интерфејс прекида

У микроконтролеру се интерфејс прекида формира помоћу два бита:

1. **IE** – *interrupt enable*
2. **IF** – *interrupt flag*

Interrupt Enable

Помоћу **IE** бита се одређује да ли контролер треба да реагује на прекиде.

Interrupt Flag

IF обезбеђује информацију да ли се прекид десио.

Управљање прекидима

IE и **IF** битови у општем случају постоје за сваки извор прекида којим процесор управља. Али се некад, да би се уштедели битови, они мапирају у један бит. На пример, код моторолиног контролера HCS12 сваки улазни сигнал дигиталног I/O порта може генерисати прекид. Али постоји само један IE и само једна ISR за цео порт. Међутим, сваки пин на порту има свој IF, што ипак омогућава кореткно обрађивање прекида.

Додатне могућности

- прекид на предњу ивицу сигнала
- прекид на задњу ивицу сигнала
- без промене ивице (на промену нивоа – *level interrupt*)
- *global interrupt enable*

Управљање прекидима

- Процедура за обраду прекида (ISR) ← IE и IF укључени (*enabled*)
- IE *set* не значи увек да су прекиди дозвољени (*set* / *clear)*
- искључивање прекидна не значи сигурно да ће прекид бити пропуштен
  - IE искључен (глобални или локални)
  - прекид се десио: IF сетован, без обзира на IE (глобални или локални)
  - IE укључен → позива се одговарајућа ISR
  - изгубљено време, али не и догађај.
- NMI – NonMaskable Interrupt, не може се искључити глобалним IE
  - NMI поседује властито управљање (TMSP430 фамилија)
- После ресета је уобичајено да су прекиди искључени!

Табела прекида

Само регистровање дешавања прекида није довољно:

- Коју ISR треба позвати?

Мапирање прекида се изводи помоћу табеле прекида *interrupt vector table*. Садржи улаз за сваки вектор прекида:

- Вектор прекида је једноставан. 1 прекид $\rightarrow$ 1 број
- Сваки вектор има фиксну адресу, која опет има фиксну адресу у програмској меморији.
- Програмер уноси почетну адресу ISR, или инструкцију скока (JMP) на ISR
- прекид $\rightarrow \begin{cases} \text{JMP на локацију у табели} \\ \text{JMP на одговарајући вектор} \end{cases} \Rightarrow$ ISR

ATmega16 Interrupt Vector Table

Vector No.	Program Address(2)	Source	Interrupt Definition
1	$000(1)	RESET	External Pin, Power-on Reset, Brown-out Reset, Watchdog Reset, and JTAG AVR Reset
2	$002	INT0	External Interrupt Request 0
3	$004	INT1	External Interrupt Request 1
4	$006	TIMER2 COMP	Timer/Counter2 Compare Match
5	$008	TIMER2 OVF	Timer/Counter2 Overflow
6	$00A	TIMER1 CAPT	Timer/Counter1 Capture Event
7	$00C	TIMER1 COMPA	Timer/Counter1 Compare Match A

Табела вектора почиње са адресе $0x0000$. Адреса вектора k је $2(k-1)$ и очекује се контролер скочи на одговарујућу ISR. Приоритет прекида је код ATmega16 фиксан: мањи редни број вектора $\rightarrow$ већи приоритет.

Приоритет прекида

- Више прекида истовремено?
- ATmega16:мањи редни број $\rightarrow$ већи приоритет.
- Да ли се сме прекинути ISR?
- Неки контролери нуде динамичку доделу приоритета

The 6502's interrupt latency is one of the 8-bit world's fastest.

Руковање прекидима

Detecting the Interrupt Condition

- интерни
- екстерни:
  - noise cancellation
  - духови (*spurious–ghost interrupts*)

Ghost Interrupts

Позивање процедуре обраде прекида

Calling the ISR

1. адреса повратка се уписује на стек
2. неки контролери снимају садржаје регистара
3. 1 прекид $\rightarrow$ бришемо IF
4. прекиди се најчешће искључују помоћу глобалног IE
   - ISR се може извршити без прекида
   - ако мора неки други преки да се обради у токоу ISR, IE се може укључути у самој ISR
   - овакав поступак компликује налажење грешака у коду.
5. извршава се прва инструкција ISR
6. ISR је налик обичној процедури, али се завршава специјално командом – RETI.

Кратак преглед

Од детекције прекида извршавају се следећи кораци:

- Поставља се IF (*Set interrupt flag*)
- Завршава се текућа инструкција (*Finish current instruction*)
- Одређује се ISR (*Identify ISR*)
- Позива се ISR (*Call ISR*)

Мора садржати сав кôд неопходан за реаговање на прекид:

- [брисање IF]
- [искључивање IE]
- [кôд акције]

Interrupt или Pooling?

Interrupt

- догађај се не дешава често
- дуг интервал између догађаја
- промена стања је битна
- кратки импулси
- догађај генерише хардвер, без бочних ефеката
- ако се ништа друго не ради, прелазимо у *sleep* мод

Pooling

- оператор је човек
- не захтева се прецизан „тајминг“
- стање је важно
- импусли су дуги
- сигнал је зашумљен
- контролер „има посла“, али не превише

Спољашњи прекид

```
void EksterniInt0() org IVT_ADDR_INT0 {
    PORTB = ~PORTB;   // toggle PORTB
}

void main() {

  DDRB   =  0xFF;     // set PORTB as output
  PORTB  =  0;        // clear PORTB

  SREG_I_bit = 1;     // Interrupt enable
  ISC01_bit  = 1;     // Int0 rising edge
  ISC00_bit = 1;
  INT0_bit   = 1;     // Enable int0

  while (1) ;         // Endless loop, port is changed
                      // inside ISR
}
```

Микропроцесорски системи

6. блок: Управљање прекидима

др Срђан Митровић, дипл. инж.
e-mail: srdjan.mitrovic@va.mod.gov.rs

зимски семестар, 2016./2017. год.