Universitatea Tehnică din Cluj-Napoca
Departamentul Calculatoare

Proiectarea cu Micro-Procesoare

Lector: Mihai Negru

An 3 – Calculatoare și Tehnologia Informației

Seria B

Curs 3: Întreruperi Externe

http://users.utcluj.ro/~negrum/

Întreruperi

- Mecanismul de întreruperi permite micro-controlerului sa răspundă la evenimente externe, sau la evenimente produse de perifericele integrate pe chip.
- In lipsa evenimentelor, procesorul poate executa programul principal, sau poate intra in stare de inactivitate (SLEEP) pentru a conserva energie.
- Tratarea întreruperilor este activată sau dezactivată prin bit-ul 7 din registrul SREG

Bit	7	6	5	4	3	2	1	0	
0x3F (0x5F)	I	T	H	S	V	N	Z	C	SREG
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- Instrucțiuni
  - **SEI** – activează sistemul de întreruperi (SREG(7) = 1)
  - **CLI** – dezactivează sistemul de întreruperi (SREG(7)=0)

Întreruperi

- Tratarea unei întreruperi – cazul ATmega64

  1. Dispozitivul periferic generează cererea de întrerupere
  2. Se finalizează execuția instrucțiunii curente
  3. PC se salvează pe stivă
     - **TOS = PC**
     - **SP = SP – 2**
  4. Accesarea vectorului specific tipului de întrerupere
  5. Execuția saltului la Procedura de Tratare a Întreruperii (*Interrupt Service Routine, ISR*)
  6. Blocare flag întreruperi (**CLI**)
  7. Execuție ISR
  8. Revenire din ISR (**reti**)
     - **SP = SP + 2**
     - **PC = TOS**
     - Activare flag întreruperi (**SEI**)

  reti este echivalent cu **sei + ret**

- *Daca in ISR se activează întreruperile prin apel SEI, se pot executa întreruperi imbricate (nested).*

Întreruperi – Surse si vectori de întrerupere (1)

Adrese absolute in memoria program (flash)

Vector No.	Program Address(2)	Source	Interrupt Definition
1	0x0000(1)	RESET	External Pin, Power-on Reset, Brown-out Reset, Watchdog Reset, and JTAG AVR Reset
2	0x0002	INT0	External Interrupt Request 0
3	0x0004	INT1	External Interrupt Request 1
4	0x0006	INT2	External Interrupt Request 2
5	0x0008	INT3	External Interrupt Request 3
6	0x000A	INT4	External Interrupt Request 4
7	0x000C	INT5	External Interrupt Request 5
8	0x000E	INT6	External Interrupt Request 6
9	0x0010	INT7	External Interrupt Request 7
10	0x0012	TIMER2 COMP	Timer/Counter2 Compare Match
11	0x0014	TIMER2 OVF	Timer/Counter2 Overflow
12	0x0016	TIMER1 CAPT	Timer/Counter1 Capture Event
13	0x0018	TIMER1 COMPA	Timer/Counter1 Compare Match A
14	0x001A	TIMER1 COMPB	Timer/Counter1 Compare Match B
15	0x001C	TIMER1 OVF	Timer/Counter1 Overflow
16	0x001E	TIMER0 COMP	Timer/Counter0 Compare Match
17	0x0020	TIMER0 OVF	Timer/Counter0 Overflow
18	0x0022	SPI, STC	SPI Serial Transfer Complete

Întreruperi – Surse si vectori de întrerupere (2)

Vector No.	Program Address(2)	Source	Interrupt Definition
19	0x0024	USART0, RX	USART0, Rx Complete
20	0x0026	USART0, UDRE	USART0 Data Register Empty
21	0x0028	USART0, TX	USART0, Tx Complete
22	0x002A	ADC	ADC Conversion Complete
23	0x002C	EE READY	EEPROM Ready
24	0x002E	ANALOG COMP	Analog Comparator
25	0x0030(3)	TIMER1 COMPC	Timer/Countre1 Compare Match C
26	0x0032(3)	TIMER3 CAPT	Timer/Counter3 Capture Event
27	0x0034(3)	TIMER3 COMPA	Timer/Counter3 Compare Match A
28	0x0036(3)	TIMER3 COMPB	Timer/Counter3 Compare Match B
29	0x0038(3)	TIMER3 COMPC	Timer/Counter3 Compare Match C
30	0x003A(3)	TIMER3 OVF	Timer/Counter3 Overflow
31	0x003C(3)	USART1, RX	USART1, Rx Complete
32	0x003E(3)	USART1, UDRE	USART1 Data Register Empty
33	0x0040(3)	USART1, TX	USART1, Tx Complete
34	0x0042(3)	TWI	Two-wire Serial Interface
35	0x0044(3)	SPM READY	Store Program Memory Ready

Întreruperi Externe

- Întreruperi externe – cauzate de activitate pe pinii externi INT7…INT0
- Pinii INT7:INT0 sunt comuni cu pinii porturilor **D** si **E** – daca porturile sunt configurate ca ieșire, se pot declanșa întreruperi software prin scrierea acestor porturi.
- Configurarea modului de sesizare a întreruperilor externe – regiștrii **EICRA** si **EICRB** – in total 16 biți, 2 biți / întrerupere

Bit	7	6	5	4	3	2	1	0	
(0x6A)	ISC31	ISC30	ISC21	ISC20	ISC11	ISC10	ISC01	ISC00	EICRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit	7	6	5	4	3	2	1	0	
0x3A (0x5A)	ISC71	ISC70	ISC61	ISC60	ISC51	ISC50	ISC41	ISC40	EICRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

ISCn1	ISCn0	
0	0	Nivel ‘0’ pe INTn generează cerere de întrerupere
0	1	Rezervat
1	0	Front descrescător pe INTn generează cerere de întrerupere
1	1	Front crescător pe INTn generează cerere de întrerupere

Întreruperi Externe

- Scrierea/citirea regiștrilor de control a întreruperilor externe
  - **EICRB** se poate scrie/citi cu **in**, **out**
  - **EICRA** se poate scrie/citi cu **lds**, **sts**

- Activarea punctuală a întreruperilor externe – folosirea registrului **EIMSK**
- Fiecare întrerupere este controlată de un bit al acestui registru
- Setarea la ‘1’ a bitului corespunzător activează întreruperea
- Registrul **EIMSK** se poate citi/scrie cu **in**, **out**

Bit	7	6	5	4	3	2	1	0	
0x39 (0x59)	**INT7**	**INT6**	**INT5**	**INT4**	**INT3**	**INT2**	**INT1**	**INT0**	**EIMSK**
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Întreruperi Externe – Exemplu

- Incrementarea unui numărător prin apăsarea unui buton, folosind mecanismul întreruperilor externe – butonul este conectat la întreruperea externa INT0

```
.org 0x0000                         ; adresa vectorului pentru întreruperea reset
        rjmp main
.org 0x0002                         ; adresa vectorului pentru întreruperea externa INT0
        rjmp isr_INT0

main:
        ldi r16, high(RAMEND)       ; inițializare stiva – necesara când folosim întreruperi!
        out SPH, R16
        ldi r16, low(RAMEND)
        out SPL, R16

        ldi r16, 0b00000011         ; configurare mod de tratare INT0 – front crescător
        sts EICRA, r16
        ldi r16, 0b00000001         ; activare întrerupere INT0
        out EIMSK, r16

        ldi r16, 0xFF               ; setăm direcția portului E – ieșire pentru numărător
        out DDRE, r16

        ldi r17, 0                  ; valoarea inițială a numărătorului
        sei                         ; activare globală a sistemului de întreruperi
```

Întreruperi Externe – Exemplu

- Incrementarea unui numărător prin apăsarea unui buton, folosind mecanismul întreruperilor externe – butonul este conectat la întreruperea externa INT0

```
loop:
        out PORTE, r17          ; scriem numărătorul pe portul E (LED-uri)
rjmp loop

isr_INT0:                       ; începutul rutinei de tratare a întreruperii INT0
        inc r17                 ; doar incrementam numărătorul
reti                            ; revenire din întrerupere
```

- Exercițiu: cum se rezolvă problema incrementării unui numărător prin apăsarea unui buton, fără folosirea sistemului de întreruperi?
- Folosiți două butoane, unul pentru incrementare si unul pentru decrementare.

Tratarea Întreruperilor Externe – Arduino

- Detectarea unor evenimente pe pini, fară a verifica in permanență starea acestora prin digitalRead

- Pentru utilizarea unei întreruperi, trebuie să îi atașăm o procedura de tratare a întreruperii (Interrupt Service Routine – ISR). In acest scop, se foloseste funcția **attachInterrupt()**, cu sintaxa:

attachInterrupt(interrupt, ISR, mode)

interrupt – numărul întreruperii externe (0, 1, 2, …)
ISR – numele procedurii de tratare a intreruperii (o procedură din program)
mode – modul de declanșare:

LOW – declanșare pe nivel ‘0’
CHANGE – declanșare când se schimbă nivelul pinului
RISING – declanșare pe front crescător
FALLING – declanșare pe front descrescător

Tratarea Întreruperilor Externe – Arduino

- Dezactivarea tratării unei întreruperi se face prin apelul funcției **detachInterrupt()**, cu sintaxa:

detachInterrupt(interrupt)

interrupt – numarul intreruperii

- Dacă se dorește dezactivarea temporară a tuturor întreruperilor, se apelează funcția **noInterrupts()**, fară parametri. Pentru re-activarea întreruperilor, se apelează funcția **interrupts()**.

- Întreruperile sunt active implicit! Dezactivarea lor trebuie facută doar pentru perioade scurte de timp, în caz contrar alte funcții arduino pot fi afectate.

Tratarea Întreruperilor Externe – Arduino

- **Exemplu:** măsurarea lățimii pulsurilor unui semnal (de exemplu, semnalul recepționat de un senzor IR de la o telecomandă, unde lățimea pulsului face diferența dintre un ‘0’ și un ‘1’)

```
const int irReceiverPin = 2;          // Pinul 2, unde se gaseste intreruperea externa 0
const int numberOfEntries = 64;       // Numarul de tranzitii de analizat

volatile unsigned long microseconds; //Variabila care tine numarul de microsecunde trecute de la pornirea programului
volatile byte index = 0;             //Pozitia in sirul de tranzitii
volatile unsigned long results[numberOfEntries];  //Sirul cu duratele tranzitiilor analizate (rezultat)

void setup()
{
  pinMode(irReceiverPin, INPUT);          //Se declara pinul intreruperii ca intrare
  Serial.begin(9600);                     //Se activeaza comunicarea prin USB, pentru afisare date
  attachInterrupt(0, analyze, CHANGE);    //Se ataseaza ISR-ul analyze la intreruperea 0, declansata la schimbarea
  results[0]=0;                           semnalului
}
        void loop()
        {
          if(index >= numberOfEntries)                  // Se verifica daca numarul de tranzitii maxim a fost analizat
          {
            Serial.println("Durations in Microseconds are:") ; // Daca da, se afiseaza toate intervalele masurate
            for( byte i=0; i < numberOfEntries; i++)
            {
              Serial.println(results[i]);
            }
            index = 0; // Dupa afisare, se re-initializeaza contorul de tranzitii la 0, si procesul se reia
          }
          delay(1000);
        }
```

Tratarea Întreruperilor Externe – Arduino

- **Exemplu:** măsurarea lățimii pulsurilor unui semnal (de exemplu, semnalul recepționat de un senzor IR de la o telecomandă, unde lățimea pulsului face diferența dintre un ‘0’ și un ‘1’)

```
void analyze()              // Procedura de tratare a intreruperii
{
  if(index < numberOfEntries  )     // Daca nu s-a ajuns la capatul sirului
  {
    if(index > 0)                   // Dar nu este prima tranzitie detectata
    {
      results[index] = micros() - microseconds;   // Se masoara timpul trecut de la ultima tranzitie
    }
    index = index + 1;              // Se incrementeaza indexul
  }
  microseconds = micros();          // Se retine timpul curent, pentru a fi etalon pentru tranzitia urmatoare
}
```

- Funcția **micros()** returnează numărul de microsecunde de la pornirea programului.
- Pentru măsurarea unor intervale de timp mai mari, cu precizie mai mică, se poate folosi funcția **millis()**, care returnează numărul de milisecunde de la pornirea programului.

Tratarea Întreruperilor Externe – Arduino

- **Atenție:**

- Toate variabilele care se pot modifica într-o funcție de tip ISR trebuie să fie declarate ca "**volatile**". Astfel, compilatorul va ști că aceste variabile se pot modifica in orice moment, și nu va optimiza codul prin atașarea acestora la regiștri, ci le va stoca întotdeauna in RAM

- Doar o procedură ISR poate rula la un moment dat, celelalte întreruperi fiind dezactivate în acest timp

- Deoarece **delay()** si **millis()** folosesc la rândul lor întreruperi, ele nu vor funcționa în timpul execuției unei proceduri ISR.

- Pentru întârzieri scurte într-o procedură ISR, se poate folosi functia **delayMicroseconds()**, care nu folosește intreruperi.

- Nu se recomandă scrierea datelor prin interfața serială într-o procedură ISR

Confuzia numărului de întrerupere – Arduino

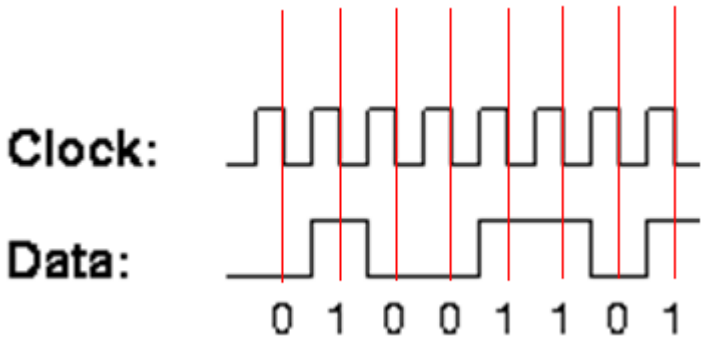
- Numărul specificat ca parametru nu este același cu numărul întreruperii externe al microcontrollerului AVR:

attachInterrupt	Name	Pin on chip	Pin on board
		(TQFP)	
0	INT4	6	D2
1	INT5	7	D3
2	INT0	43	D21
3	INT1	44	D20
4	INT2	45	D19
5	INT3	46	D18

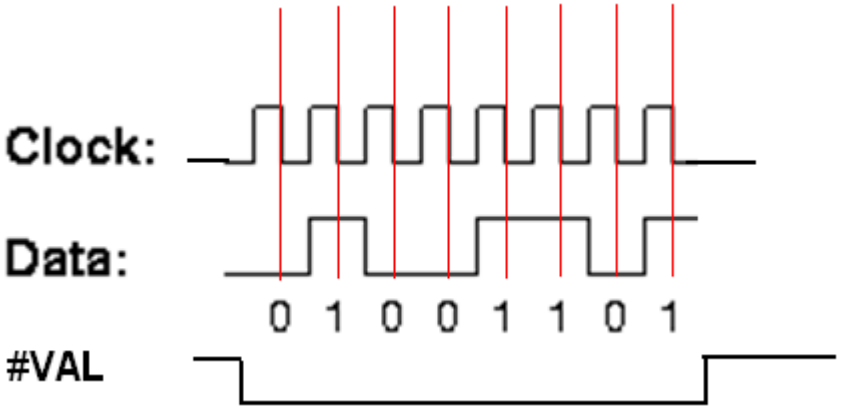
- Soluţia: utilizarea funcţiei digitalPinToInterrupt(pin)
  - Exemplu:
  - attachInterrupt(digitalPinToInterrupt(21), isr, FALLING) va ataşa întreruperii Arduino **2**, corespunzătoare întreruperii externe **INT0** de la ATMega2560, accesibilă prin pinul digital 21, procedura **isr**, care se va apela când pe pin se va efectua o tranziţie din HIGH în LOW.
- Dacă un pin digital nu are și funcție de întrerupere, funcţia digitalPinToInterrupt va returna valoarea -1.

Exerciţii – Arduino

- Afişaţi, prin interfaţa serială, numărul pinilor care au și funcție de întrerupere.
- Scrieţi un program capabil să preia date seriale, sincronizate pe un semnal de ceas, ca în figura de mai jos:



- Modificaţi programul pentru a utiliza un semnal suplimentar, care marchează începutul şi sfârşitul octetului:



Referințe

1. **Atmel ATmega640/V-1280/V-1281/V-2560/V-2561/V datasheet**
2. http://arduino.cc/en/Hacking/PinMapping2560
3. **Atmel Atmega64 datasheet**