ATmega328P Interrupt Programming in C

TECH 3230 – Lecture 8

Prof Kohn

Presentation created with a little help from ChatGDP

THE UNIVERSITY OF MEMPHIS

1

What is an Interrupt?

Example:

You are watching a movie and your phone rings….what do you do?

You probably pause the movie and answer the phone. That is an interrupt in real life.

An **External Input** (ie the phone ringing) needs an **immediate response**, so you stop what you are doing….and then take care of the external input.

2

Interrupts

- In terms of computers and embedded systems, Interrupts are **HARDWARE TRIGGERED EVENTS**

- They can be triggered by:
  - External Interrupts (INT0, INT1)
  - Pin Change Interrupts (PCINT)
  - Timer/Counter Interrupts
  - ADC Conversion Complete Interrupt
  - USART Interrupts
  - Watchdog Timer Interrupts

3

Interrupt Vector Table

- Each interrupt has a unique vector address.
- When triggered, the CPU jumps to the Interrupt Service Routine (ISR).
- ISR functions are linked to vector names in avr/interrupt.h
- Reset vector is located at address 0x0000 and it points to the program to run on a restart (on a desktop PC this would be the BIOS, but in embedded systems, this is usually main)

4

Atmel 328p Interrupts

NumberInterrupt definition	Vector name
External Interrupt Request 0	INT0_vect
External Interrupt Request 1	INT1_vect
Pin Change Interrupt Request 0 (Port B)	PCINT0_vect
Pin Change Interrupt Request 1 (Port C)	PCINT1_vect
Pin Change Interrupt Request 2 (Port D)	PCINT2_vect
Watchdog Time-out Interrupt	WDT_vect
Timer/Counter2 Compare Match A	TIMER2_COMPA_vect
Timer/Counter2 Compare Match B	TIMER2_COMPB_vect
Timer/Counter2 Overflow	TIMER2_OVF_vect

5

Atmel 328p Interrupts

NumberInterrupt definition	Vector name
Timer/Counter1 Capture Event	TIMER1_CAPT_vect
Timer/Counter1 Compare Match A	TIMER1_COMPA_vect
Timer/Counter1 Compare Match B	TIMER1_COMPB_vect
Timer/Counter1 Overflow	TIMER1_OVF_vect
Timer/Counter0 Compare Match A	TIMER0_COMPA_vect
Timer/Counter0 Compare Match B	TIMER0_COMPB_vect
Timer/Counter0 Overflow	TIMER0_OVF_vect
SPI Serial Transfer Complete	SPI_STC_vect

6

Atmel 328p Interrupts

NumberInterrupt definition	Vector name
USART Rx Complete	USART_RX_vect
USART Data Register Empty	USART_UDRE_vect
USART Tx Complete	USART_TX_vect
ADC Conversion Complete	ADC_vect
EEPROM Ready	EE_READY_vect
Analog Comparator	ANALOG_COMP_vect
Two-wire Serial Interface	TWI_vect
Store Program Memory Read	SPM_READY_vect

7

Basic Interrupt Flow

- Event occurs (button press, timer overflow, etc.)
- Interrupt flag is set by hardware.
- CPU completes current instruction.
- Stores current state (eg where I am in the program)
- CPU jumps to ISR.
- ISR executes
- Restore previous state and return to main program.

8

Global Interrupt Enable

- Global interrupts are enabled using sei().
- Global interrupts are disabled using cli().
- Header file required: #include <avr/interrupt.h>
- Without global enable, ISRs will not execute.

9

External Interrupt Example

- INT0 connected to PD2 on ATmega328P.
- Triggered by rising edge, falling edge, or level.
- Commonly used for buttons and sensors.

10

External Interrupt Code Example

```
#include <avr/io.h>
#include <avr/interrupt.h>

ISR(INT0_vect)
{
   PORTB ^= (1 << PB5);
}
```

```
int main(void)
{
   DDRB |= (1 << PB5);

   EICRA |= (1 << ISC01);
   EIMSK |= (1 << INT0);

   sei();

   while(1)
   {
   }
}
```

11

Timer Interrupts

- Timers can generate periodic interrupts.
- Timer0 is 8-bit.
- Timer1 is 16-bit.
- Timer2 is 8-bit and often used for PWM.
- Useful for delays and scheduling tasks.

12

Timer Overflow Example

- TCCR1B |= (1 << CS12);
  TIMSK1 |= (1 << TOIE1);

```
ISR(TIMER1_OVF_vect)
{
   PORTB ^= (1 << PB5);
}
```

13

Pin Change Interrupts

- Available on most GPIO pins.
- More flexible than INT0/INT1.
- Grouped into PCINT0, PCINT1, and PCINT2.
- Any change on enabled pins triggers ISR.

14

ADC Interrupt Example

- ADCSRA |= (1 << ADIE);

```
ISR(ADC_vect)
{
   uint16_t value = ADC;
}
```

15

USART Receive Interrupt

```
• ISR(USART_RX_vect)
  {
     char data = UDR0;
  }
```

16

Writing Good ISRs

- Keep ISRs short and fast.
- Avoid long loops and delays.
- Do not use printf() inside ISRs.
- Use volatile variables when shared with main().
- Avoid nested interrupts unless necessary.

17

Using volatile Variables

```
volatile uint8_t flag = 0;

ISR(INT0_vect)
{
   flag = 1;
}

int main(void)
{
   while(1)
   {
      if(flag)
      {
         flag = 0;
      }
   }
}
```

18

Common Interrupt Registers

- EIMSK – External Interrupt Mask Register
- EICRA – External Interrupt Control Register
- TIMSK1 – Timer Interrupt Mask Register
- TCCR1B – Timer/Counter Control Register
- SREG – Status Register containing global interrupt flag

19

Debugging Interrupts

- Verify global interrupts are enabled.
- Check correct ISR vector name.
- Confirm interrupt mask bits are set.
- Use LEDs or UART messages for debugging.
- Check clock configuration and fuse settings.

20

Polling vs Interrupts

- Polling continuously checks hardware.
- Interrupts react only when events occur.
- Interrupts reduce CPU usage.
- Polling may be simpler for small tasks.

21

Practical Applications

- Button press detection
- Motor control timing
- Sensor data acquisition
- UART communication
- Real-time embedded systems

22

Summary

- Interrupts are essential for embedded systems.
- ATmega328P provides many interrupt sources.
- ISRs must be short and efficient.
- Timers and external interrupts are heavily used.
- Proper configuration is critical for reliable operation.

23