Interrupts vs. Polling

Polling

- Constantly reading a memory location, in order receive updates of an input value

```
#include <avr/io.h>
#include <util/delay.h>
#include <stdint.h>
int main(void){
      unsigned char key_cur,key_pre;
      DDRB=0x0f;
      DDRD=0x00;
      PORTB=0x01;  //turn on the led on the left side as initial state
      PORTD=0xff;  //configurate as pull-up input port
      //waiting for key pressed
      while(1){
            key_pre=key_cur;
            key_cur=PIND&0x1;    //read the key state
            _delay_ms(20); // deltet key jitter
            if(key_cur==0&&key_pre==1) {
            if(PORTB==0x8)
                        PORTB=0x01;
            else
                        PORTB=PORTB<<1;
            }
            //execute another tasks here.
            }
}
```

Polling

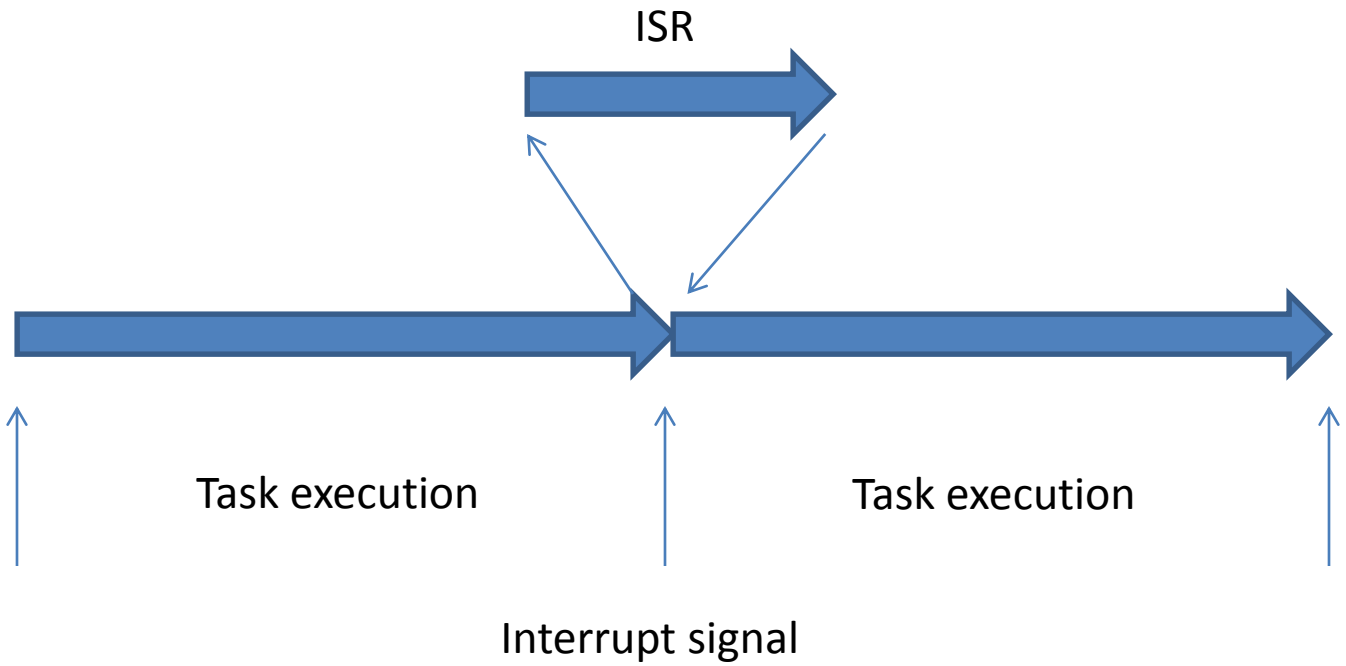
What happens if task need to run for 1 Sec

Miss some action of key pressed……

Interrupt Handling

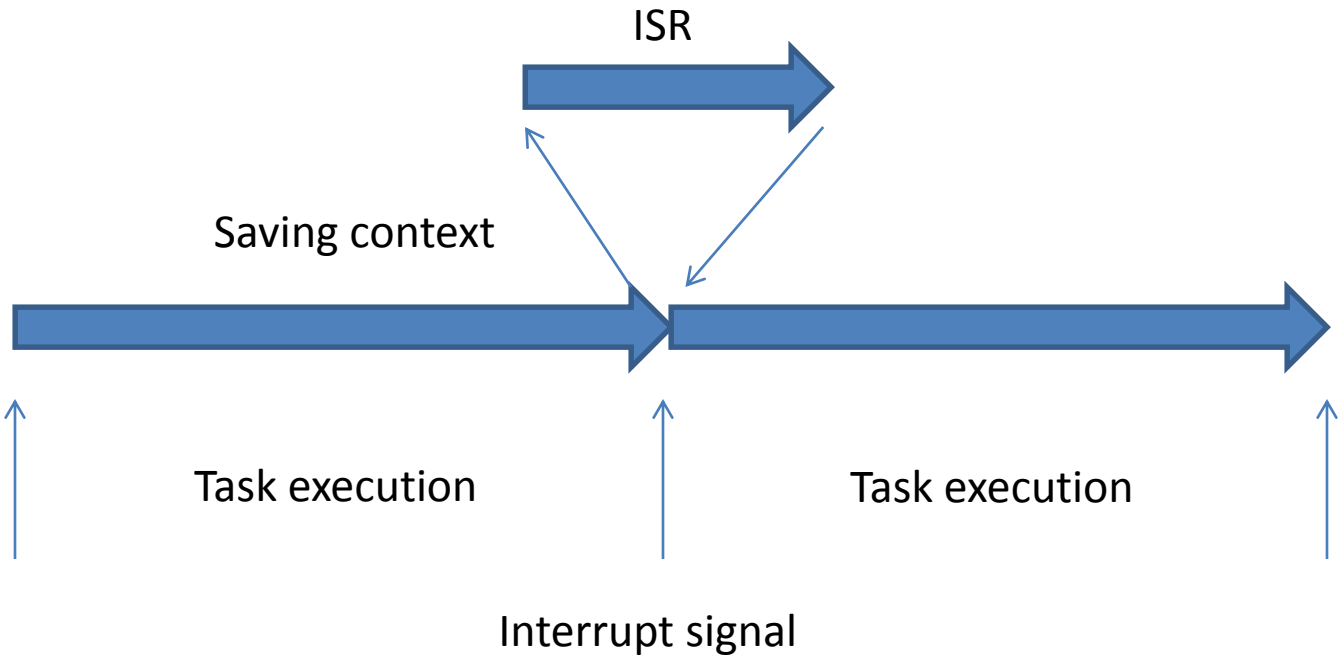
- Processor interrupts (preempts) the current flow of control
- Time spent in interrupt handlers should be kept as short as possible
- Microcontroller offers interrupts for various conditions
  - Not all are useful all the time: enable required interrupts
  - Some critical may require atomic execution (no interruptions guaranteed)
  - Disable / re-enable interrupts around critical section

In details



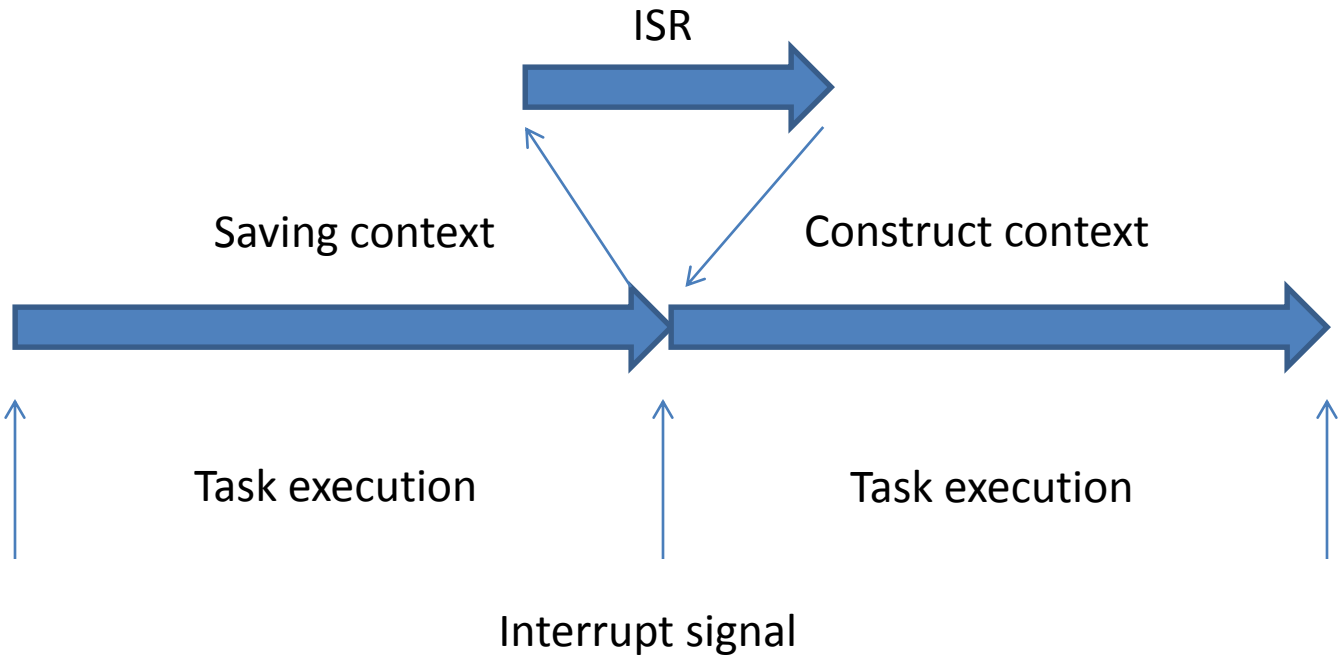
In details

- Saving context: Push all temporary variables (like program counter) into stack



In details

- Construct context: Pull all temporary variables (like program counter) out of stack



Sources of Interrupts

- Timers: System “ticks”, periodic tasks
- Communications
  - Ethernet
  - USB
  - Serial
- Periphery
  - E.g. ADC (Conversion complete)
  - Memory management
- Software

  Software interrupts (trap instructions) / illegal instructions
- Reset / Power-On

Interrupts vs. Polling

- Polling:
  - Continuously poll IOs for change of value
  - Cons:
    - Most polls are unneeded – value did not change
    - High CPU usage
    - Reaction time depends on #IOs
- Interrupt
  - Normal execution is interrupted when event occurs
  - Pro:
    - Processor resources are only used when necessary
  - Cons:
    - Program execution is interrupted in a non-deterministic manner

Interrupt Service Routine (ISR)

- Event handler for interrupt

- Special, user-defined function for handling the interrupt

Tasks

- Try out and understand the Interrupt based KEY-LED package