MSP430 Family

October 23, 2016, Bulat Valeev

Lection 4. Interrupt based program.

Challenges

Interrupt based program

Structure of the code

Control variables in the program
    Safety issues

Task in the class: Adjusted blinker
    Result

Hometask

Challenges

What you should know at the end of the day.

- Learn how write simple software in the MCU
- Connect MCU with PC
- Connect different little code parts in one program.

Interrupt based program

The most useful way to write the software for low performance system is interrupt based system.
Interrupt based system allows to write complex multi-threaded system without OS implementation.
There is no main loop inside of the code.

State machine symbiosis

The system should somehow control operations and data flows in the program.
Typically developers use state machine in the timer catch-compare interrupt for the operating purposes.
The disadvantage of this way is high requirements to reliability of the code. The joint state machine and interrupts can affect to the unpredictable behaviour

Structure of the code

The struture of the code is following:

- Initialization
- State machine initialization
- Almost empty while loop
- Main process with state machine control in the timer loop.
- Number of interrupt functions, which change state machine.

Structure of the code

The figure of the code is the following:

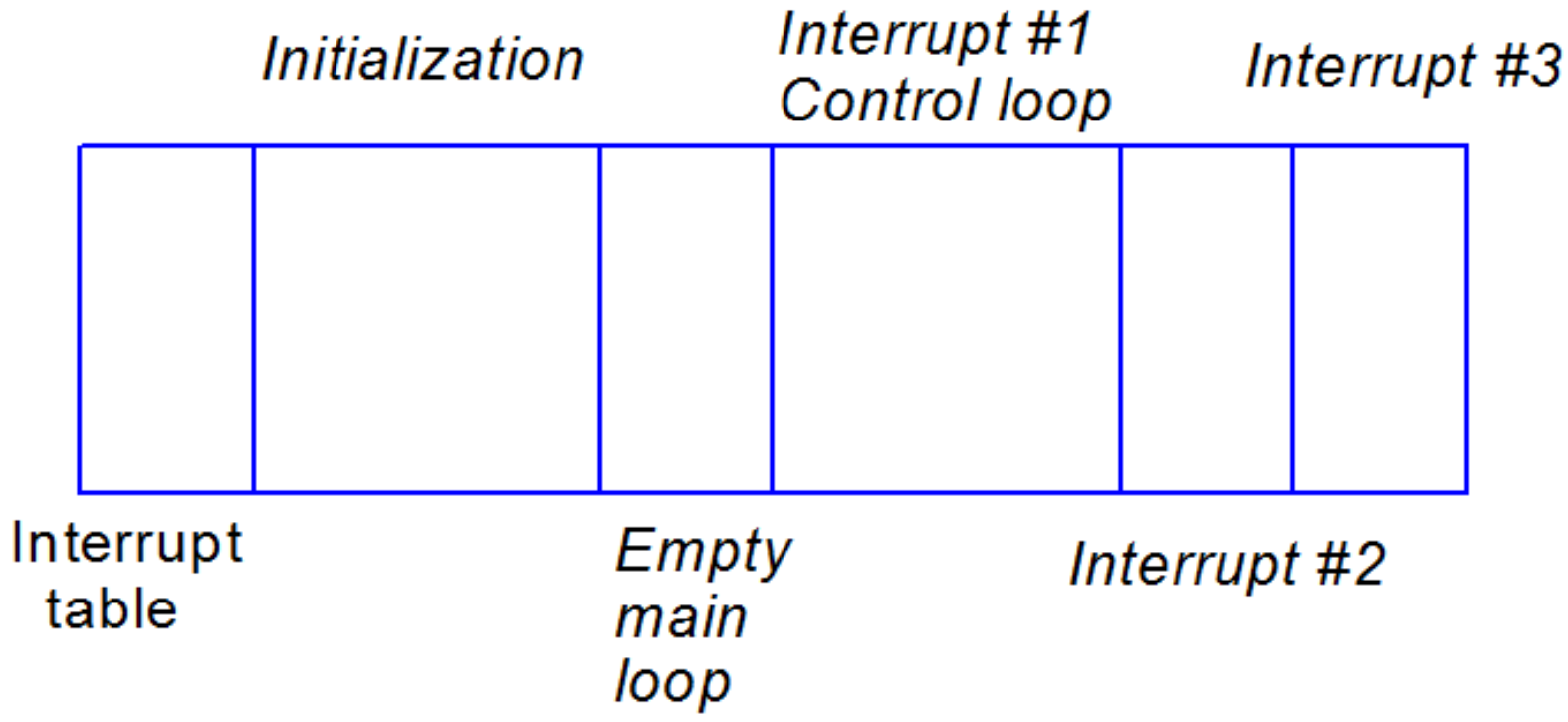

Figure: *MSP430 Launchpad*

How typical code work

The time function figure of the code:

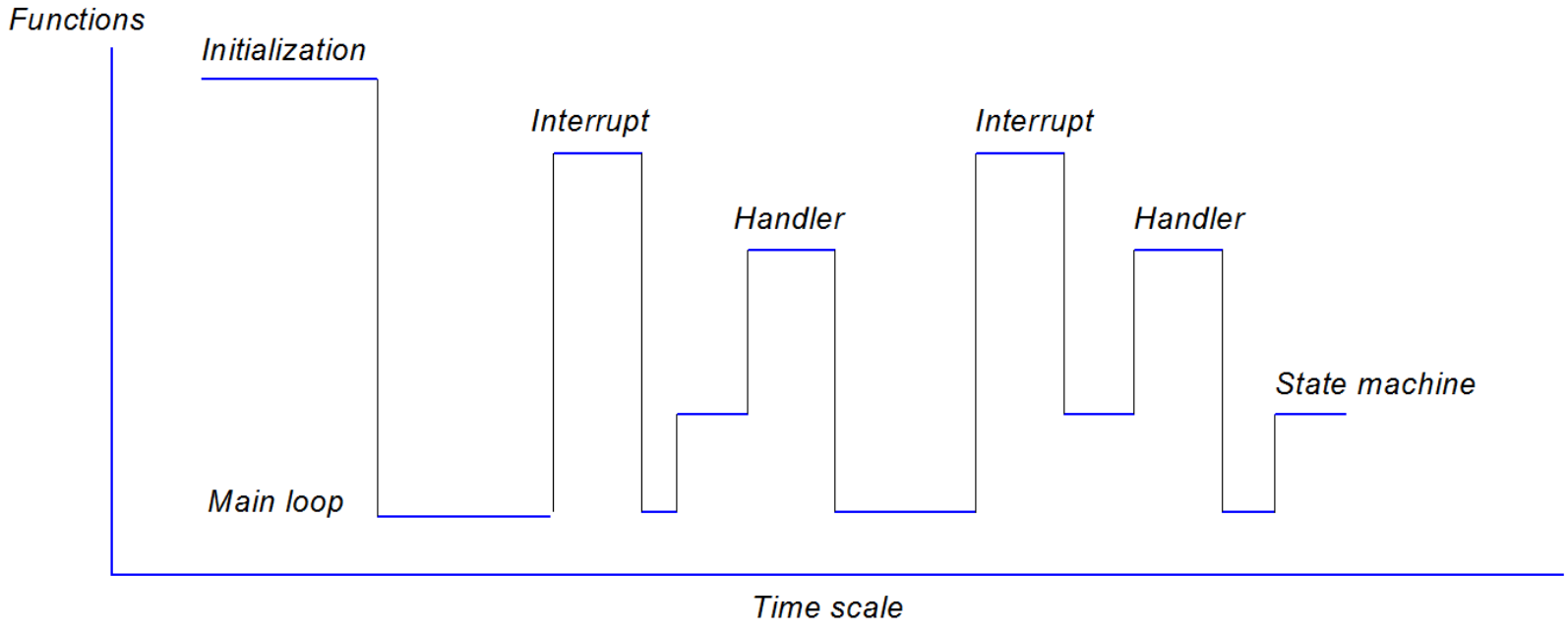

Figure: *MSP430 Launchpad*

Control variables in the program

The flags (control variables) is the global variables which set by the interrupt functions. The machine watch each flag and run the certain function for each flag.
Control variables allow to make well-structured code even without interrupts.

Example

```
int main(void){
Init(); /Initialization
while(1){
        if (flag_1){
                flag_1=0;//do something
                flag_2=1;
        }
        if (flag_2){
                flag_2=0;//do something
                flag_1=1;
        }
}
return;
}
```

Safety issues

It should be noted, that flags are very sensible to changes. Important to control the integrity of the operation which will somehow work with the control variables.

Atomic operations

An operation is atomic if it appears to the rest of the system to occur instantaneously.[]
In the simple words atomic operation will be never interrupted.

> Each operation with control variables MUST be atomic

The main point in the MCU to make operation atomic is interrupt shutdown during atomic operation.

Task in the class: Adjusted blinker

Make code which will turn on and off the blinker in the launchpad.
The blink frequency should be also adjusted.
Control over the blinker will be performed via UART from the computer.

Hints

You can use the timer and UART periphery.
Use the *Terminal.exe* to control the blinker.
Try to make while loop empty in the *main* function.

Initialization

The resulting initialization code is presented here:

```
int main(void)
{
Clock_init();
Timer_init();
Uart_init();
__bis_SR_register( GIE);// G I enable
while (1)
{
}
}
```

Control loop

The resulting timer code is presented here:

```
#pragma vector=TIMER1_A0_VECTOR
__interrupt void CCR0_ISR(void)
{
 if (flag_1){
                    P1OUT ^= RXLED;
         }
 TA1CCTL0 &=~CCIFG;
}
```

UART loop

The resulting UART code is presented here:

```
#pragma vector=USCIAB0RX_VECTOR
__interrupt void USCI0RX_ISR(void)
{
     if (UCA0RXBUF == 'a') // 'a' received?
     {
       flag1=1;
     }
    if (UCA0RXBUF == 'o') // 'o' received?
     {
       flag1=0;
     }
}
```

Hometask

The hometask today: write ADC, which will send digital data via UART to the computer. The ADC should calculate mean value over flexible number of samples. The PC user should adjust the interval between samples, length of the conversion window, should has three modes: single sample, simultaneous conversion and turned off. Deadline is xx.xx.xxxx.

Hints

Use timer as the state machine.

Use API which you have written previously.

Thanks for your attention

Reference slide

http://easyelectronics.ru/avr-uchebnyj-kurs-programmirovanie-na-si-atomarnye-operacii.html

http://easyelectronics.ru/avr-uchebnyj-kurs-operacionnaya-sistema-dispetcher-zadach.html