```
+-----+
| MCU | ไทม์เมอร์อินเตอร์รัพท์(Timer/Counter Interrupt)
| U06 | โดย อ.กำธร เรือนฝายกาศ
+-----+
```

(1) กล่าวโดยทั่วไป

ภายใน AT90S2313 มีไทม์เมอร์/เคาท์เตอร์ สองชุด ชุดที่หนึ่งเป็นแบบ 8 บิต
ชุดที่สองเป็นแบบ 16 บิต

8-bit Timer/Counter0
====================

1. รีจิสเตอร์ TCCR0 - Timer/Counter0 Control Register

bit	7	6	5	4	3	2	1	0
	-	-	-	-	-	CS02	CS01	CS00
R/W	R	R	R	R	R	R/W	R/W	R/W
initial-value	0	0	0	0	0	0	0	0

CS02	CS01	CS00	Description
0	0	0	stop,the timer/counter 0
0	0	1	ck/1
0	1	0	ck/8
0	1	1	ck/64
1	0	0	ck/256
1	0	1	ck/1024
1	1	0	external (t0),falling edge
1	1	1	external (t0),rising edge

รีจิสเตอร์นี้ใช้กำหนดสัญญาณ clock ที่จะป้อนให้ตัวนับ(counter) เช่น
ถ้าเราใช้คลิสตอล 4MHz และให้รีจิสเตอร์ TCCR0 มีค่าเท่ากับ
0b00000100 สัญญาณ clock 4MHz จะถูกหารด้วย 256 เท่ากับ 15625Hz
แล้วนำไปเข้า เป็นสัญญาณ clock ของวงจรนับ การนับของ timer/counter
สเต็ปต่อสเต็ป เช่นนับจาก 00000000 เป็น 00000001 จะห่างกัน 64us
เป็นต้น

2. รีจิสเตอร์ TCNT0 - Timer/Counter0

bit	7	6	5	4	3	2	1	0
	MSB							LSB
R/W	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

รีจิสเตอร์นี้จะแสดงค่าที่กำลังนับอยู่ในปัจจุบันของ timer/counter
เราสามารถอ่านหรือเขียนรีจิสเตอร์นี้ได้ การเขียนค่าลงไปในรีจิสเตอร์
ตัวนี้ จะทำให้ค่า timer/counter ที่กำลังนับ กลายเป็นค่าใหม่ที่เรา
ได้เขียนลงไป

3. Overflow interrupt หมายเลข 6 จาก Timer/Counter0 จะเกิดขึ้นทุกครั้งที่ counter เกิดการ overflow การที่จะให้อินเตอร์รัพท์นี้ทำงาน มีวิธีการดังนี้คือ

1) ตั้งอินเตอร์รัพท์เวคเตอร์(interrupt vector) หมายเลข 6 ให้ไปยังอินเตอร์รัพท์รูทีนที่ต้องการให้ทำงาน ในที่นี้คือคำสั่ง

```
rcall   I_7Segment              ;018 #6 TIMER0 OVF0
```

เป็นการตั้งให้อินเตอร์รัพท์นี้ กระโดดไปทำงานที่ I_7Segment เมื่อทำงานเสร็จที่ท้ายของอินเตอร์รัพท์รูทีนนี้ จะจบด้วยการสั่ง enable ให้อินเตอร์รัพท์นี้ทำงานอีกครั้งด้วยคำสั่ง sei (global interrupt enable) ตามด้วยคำสั่ง reti ซึ่งแทนที่จะเป็นคำสั่ง ret (subroutine return) เหมือนที่ผ่านมา

```
sei                             ;232 global interrupt enable
reti                            ;233 interrupt return
```

คำสั่ง reti ก็คือ interrupt return ซึ่งจะทำงานต่างกับคำสั่ง ret ก็คือคำสั่ง ret จะ pop ค่าของ PC คืนจากสแตค แต่คำสั่ง reti นอกจากจะ pop ค่า PC คืนจากสแตคแล้ว ยังสั่งเคลียร์บิต I ในรีจิสเตอร์ SREG ซึ่งเป็นบิตควบคุมการอินเตอร์รัพท์รวมของทุกอินเตอร์รัพท์

2) ทำการ enable ให้ timer0 overflow interrupt ทำงานโดยให้บิตที่ 1 คือบิต TOIE0 เป็นลอจิก 1

```
ldi     r16,    0b00000010      ;303 enable TIMER0 overflow int(#6)
out     TIMSK,  r16             ;304 -
```

3) ตั้งบิต I ของ SERG เท่ากับ 1 ด้วยคำสั่ง sei(global interrupt enable)

```
sei                             ;305 global interrupt enable
```

4) ตั้งค่าตัวหารในรีจิสเตอร์ TCCR0 ตามที่เราต้องการ ในที่นี้ตั้งไว้เท่ากับ 256 ดังนั้นความถี่ในการเกิดอินเตอร์รัพท์จะเท่ากับ

4MHz/256/256 = 61Hz

ถ้าตั้งให้ตัวหารเท่ากับ 1024 คือให้ค่ารีจิสเตอร์ TCCR0 มีค่าเท่ากับ 00000101 จะได้ความถี่ในการเกิดอินเตอร์รัพท์หมายเลข 6 นี้เท่ากับ

4MHz/1024/256 = 15.3Hz

เลข 256 ตัวหลัง เป็นจำนวนสเต็ปที่ counter นับไปจนครบ 1 รอบ กล่าวคือ counter มีขนาด 8 บิต ดังนั้นจึงนับได้ตั้งแต่ 00000000,00000001,00000010,... ไปจนถึง 11111111 รวมแล้วได้ 256 สเต็ป ก็จะเกิด timer0 overflow interrupt ขึ้นครั้งหนึ่งนั่นเอง

```
ldi     r16,    0b00000100      ;306 TIMER0 source = internal_clock/256
out     TCCR0,  r16             ;307 -
```

5) จากจุดนี้เป็นต้นไป Timer0 overflow interrupt ก็จะเริ่มทำงานทันที ถ้าไล่โปรแกรมดูจะพบว่าในฟังค์ชั่น main นั้นเราได้สั่งให้ MCU วนทำงานอยู่ในลูปไม่รู้จบคือ

```
L_MainLoop:                             ;311
        rjmp            L_MainLoop      ;312
```

ที่จริงแล้วถ้าเราไม่ได้ใช้อินเตอร์ริพท์ seven segment
ก็จะดับเพราะ MCU วนอยู่ในลูปนี้ตลอดเวลา แต่เนื่องจากเราใช้
อินเตอร์ริพท์ 6 ในการเรียกใช้ฟังค์ชัน 7Segment ทำงานทุก
ครั้งที่ counter เกิด overflow ขึ้น จึงทำให้ seven segment
แสดงตัวเลขออกมาได้

6) ประโยชน์ในการใช้งานอินเตอร์ริพท์แบบนี้ทำให้เรา สามารถกำหนด
เวลาที่แม่นยำในการที่จะให้เกิดการทำงานอะไรบางอย่างขึ้น เช่น
การสุ่มสัญญาณในกระบวนการแปลงสัญญาณอนาลอกให้เป็นดิจิตอล
เป็นต้น

(2) รายการอุปกรณ์

ไอซี AT90S2313	1	ตัว
seven segment common cathode	3	ตัว

(3) โปรแกรม

```
;***************************************************************************
; AT90S2313
; Timer interrupt
; copyright (c) 2001 by Gumtorn Ruanfaigad.Allright reserved.
; Electronics devision.RIT Chiangmai campus.
; ~ XTAL        = 4.0MHz
; ~ Ucc         = 5v
; ~ MCU         = AT90S2313,Atmel corperation,www.atmel.com
;
;         ---------    ---------
;        |*         \__/         |
;        |                       |
;      -=|1RESET     VCC      20|=-
;      -=|2PD0(RXD)  PB7(SCK) 19|=-
;      -=|3PD1(TXD)  PB6(MISO)18|=-
;      -=|4XTAL2     PB5(MOSI)17|=-
;      -=|5XTAL1     PB4      16|=-
;      -=|6PD2(INT0) PB3(OC1) 15|=-
;      -=|7PD3(INT1) PB2      14|=-
;      -=|8PD4(T0)   PB1(AIN1)13|=-
;      -=|9PD5(T1)   PB0(AIN0)12|=-
;      -=|10GND      PD6(ICP) 11|=-
;        |                       |
;         -----------------------
;
;***************************************************************************

.include        "2313def.inc"           ;000
.DEVICE         AT90S2313               ;001
.DSEG                                   ;002 data segment begin hear
                                        ;003 ==============
                                        ;004 register usage
                                        ;005 ==============
        .def    D_7Seg  = r21           ;006 7 segment value (all 3 digits)
.CSEG                                   ;007
.ORG            0x0000                  ;008
```

```
                                        ;009 ======================
                                        ;010 interrupt vector table
                                        ;011 ======================
        rjmp    F_main                  ;012 #0 RESET
        reti                            ;013 #1 INT0
        reti                            ;014 #2 INT1
        reti                            ;015 #3 TIMER1 CAPT1l
        reti                            ;016 #4 TIMER1 COMP1
        reti                            ;017 #5 TIMER1 OVF1
        rcall   I_7Segment              ;018 #6 TIMER0 OVF0
        reti                            ;019 #7 UART,RX
        reti                            ;020 #8 UART,UDRE
        reti                            ;021 #9 UART,TX
        reti                            ;022 #10 ANA_COMP

;=============================================================================
;Function OutSegment
;{rin}  r16=number to display 0..9
;       r17=digit select,0bllll0llll=3 0bllll10lll=2 0bllllll0ll=0
;{rout} none
;{rdel} r0,r16,r17,Z
;{port} PORTD[x65432l0] (segment)
;               |||||||  *** use with common cathode 7 segment only ***
;               |||||||
;               |||||||                +-----+    +-----+    +-----+
;               ||||||+- PD0 -------a-|      |--a-|      |--a-|      |
;               |||||+-- PD1 -------b-|  -   |--b-|  -   |--b-|  -   |
;               ||||+--- PD2 -------c-| | |  |--c-| | |  |--c-| | |  |
;               |||+---- PD3 -------d-|  -   |--d-|  -   |--d-|  -   |
;               ||+----- PD4 -------e-| | |  |--e-| | |  |--e-| | |  |
;               |+------ PD5 -------f-|  - *|--f-|  - *|--f-|  - *|
;               +------- PD6 -------g-|      |--g-|      |--g-|      |
;                                     +-----+    +-----+    +-----+
;       PORTB[765432l0] (digits)        |          |          |
;                |||                    |          |          |
;                ||+--- digit 0 --------|----------|----------+
;                |+---- digit 1 --------|----------+
;                +----- digit 3 --------+
;
;=========================================
F_OutSeg:                               ;100
        push    r0                      ;101
        push    r16                     ;102
        push    r17                     ;103
        push    ZL                      ;104
        push    ZH                      ;105
        ldi     ZL,     LOW(LSegTab*2)  ;106 get seg table base address
        ldi     ZH,     HIGH(LSegTab*2) ;107 -
        cpi     r17,    0               ;108 if(digit=0)
        brne            L_OutSegSkip1   ;109 else
        ldi     r17,    0bllllll0ll     ;110 then
        rjmp            L_OutSegL1      ;111 -
L_OutSegSkip1:                          ;112
        cpi     r17,    1               ;113 if(digit=1)
        brne            L_OutSegSkip2   ;114 else
        ldi     r17,    0bllll10lll     ;115 then
        rjmp            L_OutSegL1      ;116 -
L_OutSegSkip2:                          ;117
        cpi     r17,    2               ;118 if(digit=1)
```

```
        brne            L_OutSegExit    ;119 else
        ldi     r17,    0b11101111      ;120 then
        rjmp            L_OutSegL1      ;121 -
L_OutSegL1:                             ;122
        cpi     r16,0                   ;123 if(num==0)
        breq            L_OutSegL2      ;124 then jump to L_OutSegL2
        adiw    ZL,     2               ;125 else increase table offset
        subi    r16,1                   ;126 - num=num-1
        rjmp            L_OutSegL1      ;127 - do it again
L_OutSegL2:                             ;128
        lpm                             ;129 get program memory  R0<-(Z)
        ldi     r16,    0b11111111      ;130 port D : all pin=output
        out     DDRD,   r16             ;131 -
        ldi     r16,    0b11111111      ;132 port B : all pin=output
        out     DDRB,   r16             ;133 -
        out     PORTB,  r17             ;134 select 7segment digit
        mov     r16,    r0              ;135 output number to 7segment
        out     PORTD,  r16             ;136 -
L_OutSegExit:                           ;137
        ldi     r16,    5               ;138
        rcall           F_msDelay       ;139
        pop     ZH                      ;140
        pop     ZL                      ;141
        pop     r17                     ;142
        pop     r16                     ;143
        pop     r0                      ;144
        ret                             ;145
                                        ;146 =====================
                                        ;147 7segment lookup table
LSegTab:                                ;148 =====================
                 ;xabcdefg              ;149 number
        .DW     0b01111110              ;150 0           a
        .DW     0b00110000              ;151 1          ---
        .DW     0b01101101              ;152 2       f |   | b
        .DW     0b01111001              ;153 3         | g |
        .DW     0b00110011              ;154 4          ---
        .DW     0b01011011              ;155 5       e |   | c
        .DW     0b01011111              ;156 6         |   |
        .DW     0b01110000              ;157 7          ---
        .DW     0b01111111              ;158 8           d
        .DW     0b01111011              ;159 9
;.....................................................................

;=====================================================================
;Function 7Segment
;(rin)  D_7Seg=number to display (3 digits ,range 0 to 127)
;(rout) none
;(rdel) none
;(port) none
;=========================================
I_7Segment:                             ;200
        cli                             ;201 disable global interrupt
        push    r16                     ;202
        push    r17                     ;203
        push    r18                     ;204
        mov     r18,    D_7Seg          ;205 store number
        ldi     r16,    0               ;206 reset counter
L_7Segment3cal:                         ;207 digit 2 bin->decimal calculation
        cpi     r18,    100             ;208 -
```

```
        brlt            L_7Segment3dis  ;209 -
        inc     r16                     ;210 -
        subi    r18,    100             ;211 -
        rjmp            L_7Segment3cal  ;212 -
L_7Segment3dis:                         ;213 -
        ldi     r17,    2               ;214 digit 2 selected
        rcall           F_OutSeg        ;215 -
        ldi     r16,    0               ;216 -
L_7Segment2cal:                         ;217 digit 1 bin->decimal calculation
        cpi     r18,    10              ;218 -
        brlt            L_7Segment2dis  ;219 -
        inc     r16                     ;220 -
        subi    r18,    10              ;221 -
        rjmp            L_7Segment2cal  ;222 -
L_7Segment2dis:                         ;223 -
        ldi     r17,    1               ;224 digit 1 selected
        rcall           F_OutSeg        ;225 -
        mov     r16,    r18             ;226 get number
        ldi     r17,    0               ;227 digit 0 selected
        rcall           F_OutSeg        ;228 -
        pop     r18                     ;229
        pop     r17                     ;230
        pop     r16                     ;231
        sei                             ;232 global interrupt enable
        reti                            ;233 interrupt return
;.....................................................................

;=====================================================================
;Function msDelay for MCU working at 4MHz
;{rin}  r16=delay time (unit in millisec.) 0->255ms
;{rout} none
;{rdel} sreg,r16,r17,r18
;{port} none
;=========================================
F_msDelay:                              ;
        push    r16                     ;
        push    r17                     ;
        push    r18                     ;
        cpi     r16,    0               ;
        breq            L_msDelayExit   ;
L_msDelayL3:                            ;
        ldi     r17,    11              ; 11
L_msDelayL2:                            ;
        ldi     r18,    113             ; 113
L_msDelayL1:                            ;
        dec     r18                     ;
        brne    L_msDelayL1             ;
        dec     r17                     ;
        brne    L_msDelayL2             ;
        dec     r16                     ;
        brne    L_msDelayL3             ;
L_msDelayExit:                          ;
        pop     r18                     ;
        pop     r17                     ;
        pop     r16                     ;
        ret                             ;
;.....................................................................
```

```
;***************************************************************************
;void main(void)
;***************************************
F_Main:                                 ;300 main function start here
        ldi     r16,    low(RAMEND)     ;301 set stack pointer
        out     SPL,    r16             ;302 -
        ldi     r16,    0b00000010      ;303 enable TIMER0 overflow interrupt(6)
        out     TIMSK,  r16             ;304 -
        sei                             ;305 global interrupt enable
        ldi     r16,    0b00000100      ;306 TIMER0 source = internal_clock/256
        out     TCCR0,  r16             ;307 -
        ldi     r16,    0b00000101      ;308 TIMER1 source = internal_clock/1024
        out     TCCR1B, r16             ;309 -
        ldi     D_7Seg, 123             ;310 7segment start value
L_MainLoop:                             ;311
        rjmp            L_MainLoop      ;312
;.....................................................
```