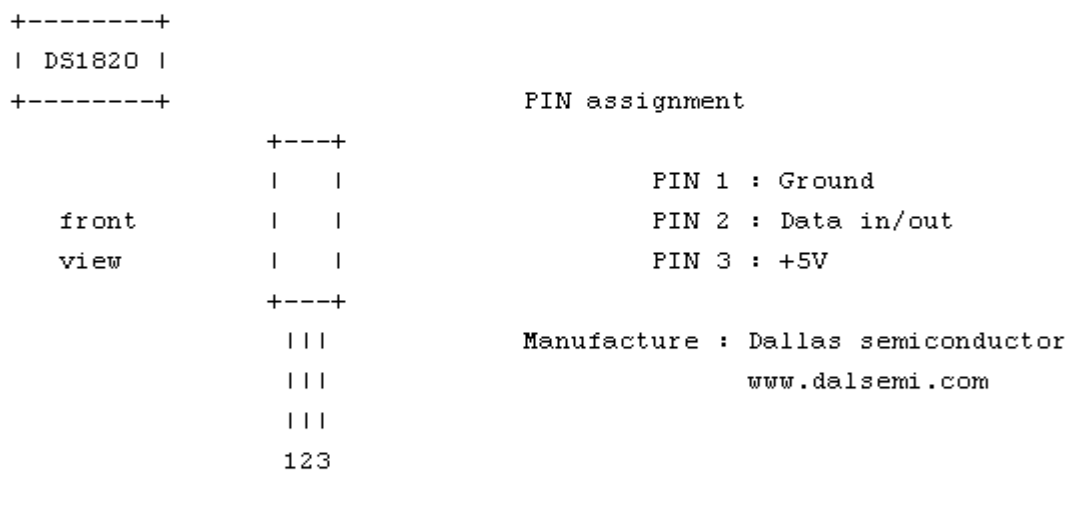
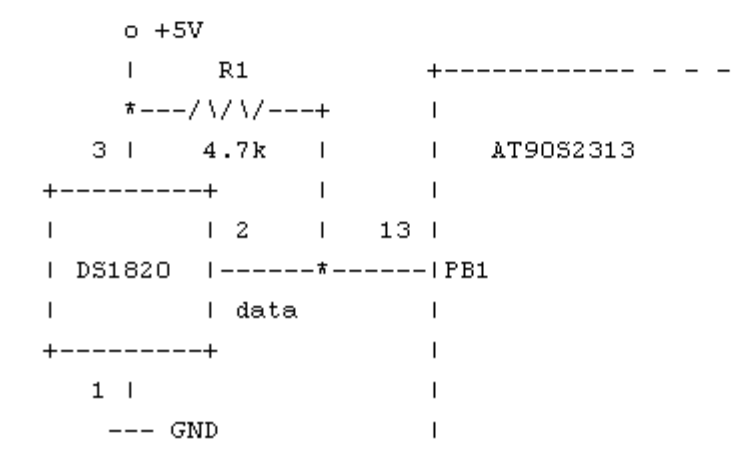
+-----+
| MCU | การอ่านข้อมูลจาก DS1820
| U07 | โดย อ.กำชร เรือนฝายกาศ
+-----+

(1) กล่าวโดยทั่วไป

ไอซี DS1820 จากบริษัท dallas semiconductor เป็นไอซีที่มีขนาดเล็ก
สามารถนำมวัดอุณหภูมิได้อย่างแม่นยำ โดยมีขนาดตัวถังใกล้เคียงกับทรานซิสเตอร์
เบอร์ 2SC458 สีดำ มี 3 ขา ถ้าหันเข้าหาตัวเรา ขาที่ 3 (ด้านขวาสุด)
จะเป็นขาไฟเลี้ยง ใช้ไฟ +5V ขากลางเป็นขาข้อมูล เป็นได้ทั้งอินพุตและเอ้าต์พุต
และขาสุดท้ายด้านซ้ายสุดเป็นขากราวด์ ในกรณีที่ระบบของเราใช้ไอซีหลายตัว เรา
สามารถต่อขา data ของไอซีแต่ละตัวเข้าด้วยกันได้ เพราะว่าไอซีแต่ละตัวจะมีรหัส
ประจำตัวที่ไม่เหมือนกัน ทำให้ไมโครคอนโทรลเลอร์ติดต่อกับไอซีวัดอุณหภูมิตัวใดตัวหนึ่งได้
โดยที่ไม่มีการขัดแย้งในเรื่องการแย่งใช้สายข้อมูลเดียวกัน แต่ในที่นี้เราใช้ไอซีวัดอุณหภูมิ
เพียงตัวเดียว(single drop) จึงข้ามการขั้นตอนการตรวจสอบรหัสประจำตัวไปได้



การต่อ DS1820 เข้ากับ AT90S2313



(2) การอ่านข้อมูลจาก DS1820

ในที่นี้จะอธิบายเฉพาะฟังค์ชั่น DS1820 ซึ่งทำหน้าที่อ่านอุณหภูมิจาก DS1820 ค่าที่ได้จากฟังค์ชั่นนี้จะเป็นค่าอุณหภูมิ 0-127 องศา เก็บไว้ในรีจีสเตอร์ r16

ลำดับขั้นตอนการอ่านข้อมูลจาก DS1820

1) ใช้ฟังค์ชั่น Reset ไอซี DS1820 โดย

1.1 ให้ MCU ดึงขา data ของ DS1820 ที่ต่อกับขา PB1(dspin) ลงกราวด์เป็นเวลา 600usec

```
FS_DS1820Reset:                         ;
        sbi     DDRB,   dspin           ; dspin=output
        cbi     PORTB,  dspin           ; pull down dspin to reset DS1820
        ldi     r16,    200             ; 600us delay
        rcall           F_usDelay       ; -
        rcall           F_usDelay       ; -
        rcall           F_usDelay       ; -
```

1.2 คอยการตอบกลับมา(check device present) ของ DS1820 โดยสั่งให้ลอย(floating) ขา PB1 เป็นเวลา 100usec

```
        sbi     PORTB,  dspin           ; pull dspin high
        cbi     DDRB,   dspin           ; floating ds_pin
        ldi     r16,    100             ; 100us delay
        rcall           F_usDelay       ; -
```

1.3 โปรแกรมให้ PB1 เป็นอินพุต แล้วเคลียร์บิต t ของ SREG

```
        cbi     DDRB,   dspin           ; dspin=input
        clt                             ; assume device not present clear T
```

1.4 ถ้า PB1 เป็นลอจิก 1 แสดงว่าไม่พบไอซี DS1820 แต่ถ้าพบ DS1820 จะดึงขา PB1 ลงกราวด์ ถ้าคำสั่ง sbis จะตรวจสอบพบว่า PB1 เป็นลอจิก 0 ก็จะทำคำสั่ง set คือให้บิต t ใน SREG เป็น 1 แต่ถ้าไม่ก็จะกระโดดข้ามไป บิต t จึงยังคงเป็น 0 เช่นเดิมเพื่อแสดงว่าไม่พบไอซี DS1820 ซึ่งอาจจะเกิดจากไอซี DS1820 เสีย หรือต่อสายไม่ถูกต้อง จากนั้นก็หน่วงเวลาไปอีก 600usec ก็เป็นอันสิ้นสุดการรีเซ็ตไอซี DS1820

```
        sbis    PINB,   dspin          ; skip if dspin is set(high)
        set                            ; found device present set t bit
        ldi     r16,    200            ; 600us delay
        rcall           F_usDelay      ; -
        rcall           F_usDelay      ; -
        rcall           F_usDelay      ; -
        ret                            ;
```

2) ส่งคำสั่ง skip rom คือ 0b11001100 ออกไปยัง DS1820 เพื่อข้ามขั้นตอน
การตรวจสอบรหัสประจำตัวไอซี ฟังค์ชั่นย่อย(SubFunction) DS1820Write
จะรับผิดชอบในการคำสั่งนี้ออกไป

```
L_DS1820ReadData:                              ;
        ldi     r17,            0b11001100    ; # <skip rom> command,0
        rcall                   FS_DS1820Write ; -
```

3) ส่งคำสั่ง convert T คือ 0b01000100 ออกไปเพื่อให้ DS1820 เปลี่ยน
อุณหภูมิแล้วเก็บไว้ในตัวเองเป็นตัวเลขดิจิตอล ช่วงนี้ DS1820 จะดึงขา
PB1 ลงกราวด์

```
        ldi     r17,            0b01000100    ; # <convert T> command,
        rcall                   FS_DS1820Write ; -
```

4) รอจนกระทั่ง convert T เสร็จ เราทราบได้เพราะถ้า DS1820 ทำเสร็จ
ก็จะปล่อยให้ขา PB1 เป็นลอจิก 1 เราจึงใช้คำสั่ง sbis คอยตรวจว่า PB1
เป็นลอจิก 1 หรือยัง ถ้ายังไม่เป็นก็จะวนอยู่ในหลูป L_DS1820Wait
ต่อไป แต่ถ้า PB1 เป็นลอจิก 1 แสดงว่า DS1820 แปลงอุณหภูมิเสร็จแล้ว
ก็จะกระโดดออกจากหลูปไปยังคำสั่ง cli

```
        sei
L_DS1820Wait:                          ; wait until convert T compleat
        cbi     DDRB,   dspin          ; check DS1820 conversion compleat
        cbi     PORTB,  dspin          ; -
        sbis    PINB,   dspin          ; -
        rjmp            L_DS1820Wait   ; -
```

หมายเหตุ คำสั่ง sei ใส่ไว้เพื่อ enable การอินเตอร์รัพท์ของ
======= seven segment เพราะถ้าไม่ใส่ไว้ seven segment จะดับตลอด
ช่วงของการรอคอยนี้

5) ส่งคำสั่ง reset ไอซี DS1820 อีกครั้งหนึ่ง โดยเรียกใช้ฟังค์ชั่น
DS1820Reset จากนั้นส่งคำสั่ง skip rom ด้วยฟังค์ชั่น DS1820Write
และตามด้วยคำสั่ง read scratchpad คือ 0b10111110 คำสั่งนี้จะเป็นการ
อ่านค่าอุณหภูมิจาก DS1820 มายัง MCU

```
cli                             ;
rcall           FS_DS1820Reset  ; reset DS1820 again
ldi     r17,    0b11001100      ; # <skip rom> command,0xCC
rcall           FS_DS1820Write  ; -
ldi     r17,    0b10111110      ; # <read scratchpad> command,0xBE
rcall           FS_DS1820Write  ; -
```

6) อ่านไบต์ที่หนึ่งเป็นค่าอุณหภูมิ 8 บิตแรก (low byte) และอ่านอีกครั้งหนึ่งจะเป็น 8 บิตหลัง (high byte) ค่าที่อ่านได้จากฟังค์ชั่น DS1820Read จะส่งกลับมาในรีจีสเตอร์ r19 เรานำไบต์ต่ำไปเก็บไว้ในรีจีสเตอร์ r16 แล้วเลื่อนบิต r16 ไบทางขวาเพื่อกำจัดอุณหภูมิ 0.5 องศา ทิ้งไปดังนี้

สมมุติให้ r16 = 01000111 เท่ากับ 35.5 องศา

```
                |||||||| 
                |||||||+------> บิตนี้แสดงอุณหภูมิทศนิยม ตำแหน่ง
                |||||||         ที่หนึ่ง มีความละเอียด 0.5 องศา
                |||||||         ในที่นี้เท่ากับ 0.5 องศา
                ++++++++------> 7 บิตที่เหลือ้ เป็นอุณหภูมิในหลัก
                                หน่วย,สิบ และร้อย ในที่นี้อ่านอุณหภูมิ
                                ได้เท่ากับ 35 (100011)
```

ถ้า r16 = 01000110 เท่ากับ 35.0 องศา

ถ้า r16 = 11110111 เท่ากับ 123.5 องศา

เนื่องจากเราต้องการความละเอียดในระดับ 1 องศา ใช้คำสั่ง lsr (Logical Shift Right) ลบบิตที่แสดงทศนิยมทิ้งไป โดย

```
ก่อน            r16 = 11110111 เท่ากับ 123.5 องศา
หลังจากเลื่อขวา r16 = 01111011 เท่ากับ 123
```

ค่าที่ได้ใน r16 หลังการเลื่อสามารถส่งไปให้ seven segment แสดงผลได้อย่างถูกต้อง

```
rcall   FS_DS1820Read           ; read low byte temperature
mov     r16,    r19             ;
lsr     r16                     ; destroy 1/2 degree
push    r16                     ;
rcall           FS_DS1820Read   ; read high byte temperature
;mov    TempHi, r19             ;
```

สำหรับไบต์สูงที่แสดงเครื่องหมายของอุณหภูมิว่าเป็น (+) หรืออุณหภูมิที่ติดลบ (-) นั้น เราไม่ต้องการ จึงทำเป็น comment เสีย โดยการใส่ตัวอักษร ';' ไว้ข้างหน้าคำสั่ง mov อันนี้แล้วแต่ว่าท่านต้องการตรวจวัดอุณหภูมิที่ติดลบหรือไม่ ถ้าต้องการไบต์นี้จะมีความหมายดังนี้ ถ้าข้อมูลที่ได้จากฟังค์ชั่น DS1820Read ใน r19 เป็น

r19 = xxxxxxx1 แสดงว่าอุณหภูมิต่ำกว่า 0 องศา(ติดลบ)
r19 = xxxxxxx0 แสดงว่าอุณหภูมิเป็นบวก
บิตที่แสดงด้วย x หมายถึงเป็น 0 หรือ 1 ก็ได้

7) สั่งให้ reset ไอซี DS1820 อีกครั้งหนึ่ง เป็นอันเสร็จขั้นตอนการอ่านค่าอุณหภูมิจาก DS1820

```
rcall           FS_DS1820Reset  ; reset DS1820
```

(3) รายการอุปกรณ์

ไอซี DS1820	1	ตัว
ไอซี AT90S2313	1	ตัว
seven segment common cathode	2	ตัว

(4) โปรแกรม

อธิบายแต่ละฟังค์ชั่น
==========

usDelay ใช้หน่วงเวลา(delay) เวลาที่หน่วงมีหน่วยเป็นไมโครเซค ให้ค่าเวลาที่ต้องการหน่วงไว้ในรีจีสเตอร์ r16 แล้วเรียก(rcall) เข้ามาที่ฟังค์ชั่นนี้ ยกตัวอย่างเช่นต้องการหน่วงเวลาเท่ากับ 25us คำสั่งจะเป็นดังนี้

```
ldi     r16,    25
rcall           F_usDelay
```

msDelay ทำงานคล้ายกับฟังค์ชั่น usDelay แต่เวลาที่หน่วงได้มีหน่วยเป็นมิลลิเซค ค่าที่ต้องการหน่วงให้กำหนดไว้ใน r16 เช่นถ้าต้องการหน่วงเวลา 45ms

```
ldi     r16,    45
rcall           F_msDelay
```

OutSegment เป็นฟังค์ชั่นสำหรับแสดงผลออก seven segment โดยสามารถกำหนดตัวเลขที่ต้องการแสดงได้ผ่าน r16 และ digit ที่ต้องการแสดงผ่านทาง r17 เช่น ต้องการแสดงตัวเลข 7 ที่ digit 1

```
        ldi     r16,    7
        ldi     r17,    1
        rcall           F_OutSegment
```

7Segment ฟังค์ชั่นนี้ใช้สำหรับแสดงผลออกทาง seven segment ได้ทีละ 3 digits โดยตัวเลขที่ต้องการแสดงให้ใส่มาใน D_7Seg การแสดงผลจะใช้ timer0 Overflow interrupt ในการเรียกฟังค์ชั่นนี้ขึ้นมาทำงาน(สังเกตุตรงท้ายฟังค์ชั่นจะจบด้วยคำสั่ง reti ไม่ใช่คำสั่ง ret) ดังนั้นจึงไม่ควรเรียกใช้ฟังค์ชั่นนี้เองด้วยคำสั่ง ดังนั้นถ้าต้องการให้ seven segment แสดงตัวเลขก็เพียงแต่เปลี่ยนค่าของ D_7Seg เท่านั้น เช่น ต้องการแสดงตัวเลข 107

```
        ldi     D_7Seg, 107
```

DS1820 เป็นฟังค์ชั่นสำหรับอ่านอุณหภูมิจากไอซี DS1820 ที่ต่อกับ PB1 อุณหภูมิที่อ่านได้จะส่งกลับมาในรีจิสเตอร์ r16 เป็นค่าอุณหภูมิที่ไม่ติดเครื่องหมาย มีหน่วยเป็นองศาเซลเซียส ถ้าต้องการให้อ่านอุณหภูมิแล้วนำไปแสดงผลออกทาง seven segment ก็สามารถเขียนโปรแกรมได้ดังนี้

```
        rcall           F_DS1820
        ldi     D_7Seg, r16
```

```
+-----------------------------------------+
| อธิบายฟังค์ชั่นย่อย DS1820Read และ DS1820Write |
+-----------------------------------------+
```

```
+------------+
| DS1820Read | ใช้อ่านข้อมูล 8 บิตแบบอนุกรมจาก DS1820 ไปยัง MCU
+------------+ ข้อมูลที่อ่านได้จะบรรจุไว้ r19 เพื่อส่งกลับไปให้
```

ผู้เรียกฟังค์ชั่นนี้ต่อไป การทำงานของฟังค์ชั่นนี้จะเริ่มจาก

ให้ r17 เป็น 1 ทุกบิตแล้ว กำหนดตัวนับจำนวนบิต r18 ให้มีค่าเท่ากับ 8 (8 บิต)

```
FS_DS1820Read:                          ;
        ldi     r17,    0b11111111      ; all dataout=0xFF for read
FS_DS1820Write:                         ; r17=command
        ldi     r18,    8               ; BitCount=8
```

กำหนด PORTB ให้เป็นเอ้าต์พุต แล้วดึงขา dspin ซึ่งตอนนี้กำหนดให้เป็น
PB1 ลงกราวด์ แล้วรอไปอีก 2us โดยเรียกฟังค์ชั่น usDelay
หมุนบิต r17 ไปทางขวาเข้า carry flag แล้วตรวจสอบว่าเป็น 1
หรือไม่(ต้องเป็น 1 อยู่แล้วเพราะเราโหลด r17 ด้วย 11111111 ไว้แล้ว)
เมื่อพบว่าเป็น 1 ก็ให้ลอยขา dspin ด้วยคำสั่ง cbi DDRB,dspin แล้วหน่วง
เวลาไปอีก 10us จากนั้นก็สั่งให้ carry flag เป็น 0 ด้วยคำสั่ง clc
ไอซี DS1820 จะนำข้อมูลออกมาไว้บนขา data ถ้าข้อมูลเป็น 1 เราก็ทำการ
เซ็ท carry flag ให้เป็น 1 แต่ถ้า data เป็น 0 ค่า carry flag ก็จะยังคง
เป็น 0 เหมือนเดิม จากนั้นเราก็นำเอาบิตข้อมูลจาก DS1820 ใน carry flag
เข้าไปที่บิต 7 ของ r19 ด้วยคำสั่ง ror r19 แล้วหน่วงเวลาไปอีก 49us

```
L_DS1820Next1:                          ;
        sbi     DDRB,   dspin           ; set dspin as output
        cbi     PORTB,  dspin           ; pull down dspin
        ldi     r16,    2               ; 2us delay
        rcall           F_usDelay       ; -
        ror     r17                     ; shift right r17.0->carryflag
        brcc            L_DS1820Send0   ; if(carry flag==0) leave line low
        cbi     DDRB,   dspin           ; else pull up dspin go high
L_DS1820Send0:                          ; 10us delay
        ldi     r16,    10              ;
        rcall           F_usDelay       ;
        clc                             ; clear carry flag
        sbic    PINB,   dspin           ; skip if dspin==0
        sec                             ; - set carryflag
        ror             r19             ; get input data bit to ds_in
        ldi     r16,    49              ; 49us delay
        rcall           F_usDelay       ; -
```

ดึงขา dspin ให้เป็นลอจิก 1 แล้วหน่วงเวลาต่ออีก 2us จากนั้นลดค่าตัวนับ
จำนวนบิตลง 1 ด้วยคำสั่ง dec r18 จากนั้นก็เช็คดูว่าอ่านครบ 8 บิต
หรือยัง ด้วยคำสั่ง brne ถ้ายังก็กระโดดไปอ่านทำบิตต่อไปเข้ามา

```
        cbi     DDRB,   dspin           ; pull up dspin backto high
        ldi     r16,    2               ; 2us delay
        rcall           F_usDelay       ;
        dec     r18                     ; BitCount--
        brne            L_DS1820Next1   ; else jump L_DS1820Next1
        ret
```

ดังนั้นเมื่ออ่านข้อมูลเข้ามาแต่ละบิต ข้อมูลใน r19 ก็จะหมุนไปขวาทีละบิต
ข้อมูลจะเข้าไปเรียงอยู่ใน r19 จนครบ 8 บิต แล้วจบฟังค์ชั่นนี้ที่คำสั่ง ret

```
        +--------------+
        | DS1820Write  | ฟังค์ชั่นย่อยนี้ ใช้ในการส่งข้อมูลอนุกรม 8 บิต
        +--------------+ จาก MCU ไปยัง DS1820 ค่าที่ต้องการส่งให้ใส่ไว้ใน
        r17 แล้วเรียกฟังค์ชั่นนี้ ด้วยคำสั่ง rcall FS_DS1820Write จะเห็นว่า
        ฟังค์ชั่นนี้ต่างกับฟังค์ชั่น DS1820Read ตรงที่การเซ็ทค่า r17 เท่านั้น

        ข้อมูล(command) จะส่งเข้ามาทาง r17 แล้วตั้งตัวนับบิต r18 ให้เท่ากับ
        8 แล้วโปรแกรมให้ dspin ของ PORTB เป็นเอ้าต์พุต แล้วดึง dspin ลงกราวด์
        จากนั้นหน่วงเวลา 2us แล้ว หมุนบิต r17 ไปทางขวาเข้า carry flag เพื่อนำ
        บิตใน carry flag ส่งออกไปให้ DS1820 โดยถ้า carry flag เป็น 0 ก็จะปล่อย
        ให้ dspin เป็น 0 ต่อไป แต่ถ้าเป็น 1 ก็จะให้ขา data ของ DS1820 เป็น 1
        ด้วยคำสั่ง cbi DDRB,dspin แล้วหน่วงเวลา 10us ตั้งแต่คำสั่ง
        clc PINB,dspin ไปจนถึงคำสั่ง ror r19 เป็นการอ่านค่าออกจาก DS1820
        ซึ่งจะทิ้งไว้ไม่ขออธิบายเพราะว่าตอนนี้เราไม่ได้ต้องการอ่านค่าออกมา
        จากนั้นก็หน่วงเวลาอีก 49us ดึงขา dspin ให้เป็น 1 แล้วหน่วงเวลาอีก 2us
        ลดค่า r18 ลง 1 แล้วตรวจสอบว่าเขียนครบ 8 บิตหรือยัง ถ้ายังก็ให้วนกลับไป
        ส่งบิตต่อไปอีก จนครบ 8 บิตแล้วออกจากฟังค์ชั่นย่อยนี้ด้วยคำสั่ง ret

FS_DS1820Write:                         ; r17=command
        ldi     r18,    8               ; BitCount=8
L_DS1820Next1:                          ;
        sbi     DDRB,   dspin           ; set dspin as output
        cbi     PORTB,  dspin           ; pull down dspin
        ldi     r16,    2               ; 2us delay
        rcall           F_usDelay       ; -
        ror     r17                     ; shift right r17.0->carryflag
        brcc            L_DS1820Send0   ; if(carry flag==0) leave line low
        cbi     DDRB,   dspin           ; else pull up dspin go high
L_DS1820Send0:                          ; 10us delay
        ldi     r16,    10              ;
        rcall           F_usDelay       ;
        clc                             ; clear carry flag
        sbic    PINB,   dspin           ; skip if dspin==0
        sec                             ; - set carryflag
        ror             r19             ; get input data bit to ds_in
        ldi     r16,    49              ; 49us delay
        rcall           F_usDelay       ; -
        cbi     DDRB,   dspin           ; pull up dspin backto high

        ldi     r16,    2               ; 2us delay
        rcall           F_usDelay       ;
        dec     r18                     ; BitCount--
        brne            L_DS1820Next1   ; else jump L_DS1820Next1
        ret
```

```
;*****************************************************************************
; AT90S2313
; i/o control,2 segment,2 switch
; copyright (c) 2001 by Gumtorn Ruanfaigad.Allright reserved.
; ~ 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"           ;
.DEVICE         AT90S2313               ;
.DSEG                                   ; data segment begin hear
                                        ; ==============
                                        ; register usage
                                        ; ==============
        .equ    dspin   = PB1           ;
        .def    D_7Seg  = r21           ; 7 segment value (all 3 digits)
.CSEG                                   ;
.ORG            0x0000                  ;
                                        ; ======================
                                        ; interrupt vector table
                                        ; ======================
        rjmp    F_main                  ; #0 RESET
        reti                            ; #1 INT0
        reti                            ; #2 INT1
        reti                            ; #3 TIMER1 CAPT11
        reti                            ; #4 TIMER1 COMP1
```

```
        reti                            ; #5 TIMER1 OVF1
        rcall   I_7Segment              ; #6 TIMER0 OVF0
        reti                            ; #7 UART,RX
        reti                            ; #8 UART,UDRE
        reti                            ; #9 UART,TX
        reti                            ; #10 ANA_COMP

;***************************************************************************
;Function OutSegment
;(rin)  r16=number to display 0..9
;       r17=digit select,0b11101111=3 0b11110111=2 0b11111011=0
;(rout) none
;(rdel) r0,r16,r17,Z
;(port) PORTD[x6543210] (segment)
;              |||||||  *** use with common cathode 7 segment only ***
;              |||||||
;              |||||||               +-----+    +-----+    +-----+
;              ||||||+- PD0 -------g-|     |----|     |----|     |
;              |||||+-- PD1 -------f-|  -  |----|  -  |----|  -  |
;              ||||+--- PD2 -------e-| | | |----| | | |----| | | |
;              |||+---- PD3 -------d-|  -  |----|  -  |----|  -  |
;              ||+----- PD4 -------c-| | | |----| | | |----| | | |
;              |+------ PD5 -------b-|  - *|----|  - *|----|  - *|
;              +------- PD6 -------a-|     |----|     |----|     |
;                                    +-----+    +-----+    +-----+
;       PORTB[7654321O] (digits)        |          |          |
;                |||                    |          |          |
;                ||+--- digit 0 --------|----------|----------+
;                |+---- digit 1 --------|----------+
;                +----- digit 3 --------+
;
;****************************************
F_OutSeg:                               ;
        push    r0                      ;
        push    r16                     ;
        push    r17                     ;
        push    ZL                      ;
        push    ZH                      ;
        ldi     ZL,     LOW(LSegTab*2)  ; get seg table base address
        ldi     ZH,     HIGH(LSegTab*2) ; -
        cpi     r17,    0               ; if(digit=0)
        brne            L_OutSegSkip1   ; else
        ldi     r17,    0b11111011      ; then
        rjmp            L_OutSegL1      ; -
```

```
L_OutSegSkip1:                          ;
        cpi     r17,    1               ; if(digit=1)
        brne            L_OutSegSkip2   ; else
        ldi     r17,    Ob11110111      ; then
        rjmp            L_OutSegL1      ; -
L_OutSegSkip2:                          ;
        cpi     r17,    2               ; if(digit=1)
        brne            L_OutSegExit    ; else
        ldi     r17,    Ob11101111      ; then
        rjmp            L_OutSegL1      ; -
L_OutSegL1:                             ;
        cpi     r16,0                   ; if(num==0)
        breq            L_OutSegL2      ; then jump to L_OutSegL2
        adiw    ZL,     2               ; else increase table offset
        subi    r16,1                   ; - num=num-1
        rjmp            L_OutSegL1      ; - do it again
L_OutSegL2:                             ;
        lpm                             ; get program memory  r0<-(Z)
        ldi     r16,    Ob11111111      ; port D : all pin=output
        out     DDRD,   r16             ; -
        ldi     r16,    Ob11111111      ; port B : all pin=output
        out     DDRB,   r16             ; -
        out     PORTB,  r17             ; select 7segment digit
        mov     r16,    r0              ; output number to 7segment
        out     PORTD,  r16             ; -
L_OutSegExit:                           ;
        ldi     r16,    5               ;
        rcall           F_msDelay       ;
        pop     ZH                      ;
        pop     ZL                      ;
        pop     r17                     ;
        pop     r16                     ;
        pop     r0                      ;
        ret                             ;
```

```
                                        ; =====================
                                        ; 7segment lookup table
LSegTab:                                ; =====================
                  ;xabcdefg             ; number
        .DW     0b01111110              ; 0           a
        .DW     0b00110000              ; 1          ---
        .DW     0b01101101              ; 2       f |   | b
        .DW     0b01111001              ; 3         | g |
        .DW     0b00110011              ; 4          ---
        .DW     0b01011011              ; 5       e |   | c
        .DW     0b01011111              ; 6         |   |
        .DW     0b01110000              ; 7          ---
        .DW     0b01111111              ; 8           d
        .DW     0b01111011              ; 9
;.....................................................................

;=====================================================================
;Function 7Segment
;(rin)  D_7Seg=number to display (3 digits ,range 0 to 127)
;(rout) none
;(rdel) none
;(port) none
;=======================================
I_7Segment:                             ;
        cli                             ; disable global interrupt
        push    r16                     ;
        push    r17                     ;
        push    r18                     ;
        mov     r18,    D_7Seg          ; store number
        ldi     r16,    0               ; reset counter
L_7Segment3cal:                         ; digit 2 bin->decimal calculation
        cpi     r18,    100             ; -
        brlt            L_7Segment3dis  ; -
        inc     r16                     ; -
        subi    r18,    100             ; -
        rjmp            L_7Segment3cal  ; -
L_7Segment3dis:                         ; -
        ldi     r17,    2               ; digit 2 selected
        rcall           F_OutSeg        ; -
        ldi     r16,    0               ; -
L_7Segment2cal:                         ; digit 1 bin->decimal calculation
        cpi     r18,    10              ; -
        brlt            L_7Segment2dis  ; -
        inc     r16                     ; -
```

```
        subi    r18,    10              ; -
        rjmp            L_7Segment2cal  ; -
L_7Segment2dis:                         ; -
        ldi     r17,    1               ; digit 1 selected
        rcall           F_OutSeg        ; -
        mov     r16,    r18             ; get number
        ldi     r17,    0               ; digit 0 selected
        rcall           F_OutSeg        ; -
        pop     r18                     ;
        pop     r17                     ;
        pop     r16                     ;
        sei                             ; global interrupt enable
        reti                            ;
;.....................................................................
;=====================================================================
;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                             ;
;.....................................................................
```

```
;***************************************************************************
;Function usDelay for MCU working at 4.43MHz
;(rin)  r16=delay time (unit in microsec.) 0->255us
;(rout) none
;(rdel) r16
;(port) none
;*****************************************
F_usDelay:                              ;
        push    r16                     ;
L_usDelayL1:                            ;
        dec     r16                     ;
        nop                             ;
        brne    L_usDelayL1             ;
        pop     r16                     ;
        ret                             ;
;.........................................................................

;***************************************************************************
;function DS1820
;(rin)  none
;(rout) r16 = unsigned temperature value ('C)
;(rdel) r17,r18,r19
;                  ::::::::::::::::::::::
;                  :: DS1820 schematic ::
;                  ::::::::::::::::::::::
;DS1820  ___          o 5V
;       |   |         |
;       |   |   +----*
;front  |   |   |    |
;view   |123|   |    / R1    ------------
;        ---    |    \ 4.7k | ...........
;        |||    |    /       | :AT90S2313:
;        ||+----+    |       | ''''''''''''
;        ||          |    13|
;        |+----------*------|PB1
;        |      data        |
;        |                  |
;       --- GND             |
;       ///
;*****************************************
F_DS1820:                               ;
        cli                             ; disable global interrupt
        push    r17                     ;
        push    r18                     ;
```

```
        push    r19                     ;
        rcall           FS_DS1820Reset  ;
        brts            L_DS1820ReadData;
        rjmp            L_DS1820Skip    ; exit this function
L_DS1820ReadData:                       ;
        ldi     r17,    0b11001100      ; # <skip rom> command,0xCC
        rcall           FS_DS1820Write  ; -
        ldi     r17,    0b01000100      ; # <convert T> command,0x44
        rcall           FS_DS1820Write  ; -
L_DS1820Wait:                           ; wait until convert T compleat
        cbi     DDRB,   dspin           ; check DS1820 conversion compleat
        cbi     PORTB,  dspin           ; -
        sbis    PINB,   dspin           ; -
        rjmp            L_DS1820Wait    ; -
        cli                             ;
        rcall           FS_DS1820Reset  ; reset DS1820 again
        ldi     r17,    0b11001100      ; # <skip rom> command,0xCC
        rcall           FS_DS1820Write  ; -
        ldi     r17,    0b10111110      ; # <read scratchpad> command,0xBE
        rcall           FS_DS1820Write  ; -
        rcall           FS_DS1820Read   ; read low byte temperature

        mov     r16,    r19             ;
        lsr     r16                     ; destroy 1/2 degree
        push    r16                     ;
        rcall           FS_DS1820Read   ; read high byte temperature
        ;mov    TempHi, r19             ;
        rcall           FS_DS1820Reset  ; reset DS1820
        pop     r16                     ;
        rjmp            L_DS1820Exit    ;
L_DS1820Skip:                           ;
        ldi     r16,    111             ; else return 111
L_DS1820Exit:                           ;
        pop     r19                     ;
        pop     r18                     ;
        pop     r17                     ;
        sei                             ; enable global interrupt
        ret                             ;
```

```
                                        ; ==================================
                                        ; subfunction DS1820 reset
                                        ; (rin) none
                                        ; (rout)T flag = 1(found DS1820)
                                        ; (rdel)r16
                                        ; (port)PORTB
                                        ; ==================================
FS_DS1820Reset:                         ;
        sbi     DDRB,   dspin           ; dspin=output
        cbi     PORTB,  dspin           ; pull down dspin to reset DS1820
        ldi     r16,    200             ; 600us delay
        rcall           F_usDelay       ; -
        rcall           F_usDelay       ; -
        rcall           F_usDelay       ; -
                                        ; ====================
                                        ; check device present
                                        ; ====================
        sbi     PORTB,  dspin           ; pull dspin high
        cbi     DDRB,   dspin           ; floating ds_pin
        ldi     r16,    100             ; 100us delay
        rcall           F_usDelay       ; -
        cbi     DDRB,   dspin           ; dspin=input
        clt                             ; assume device not present clear T
        sbis    PINB,   dspin           ; skip if dspin is set(high)
        set                             ; found device present set T
        ldi     r16,    200             ; 600us delay
        rcall           F_usDelay       ; -
        rcall           F_usDelay       ; -
        rcall           F_usDelay       ; -
        ret                             ;
                                        ; ==================================
                                        ; subfunction DS1820 i/o
                                        ; (rin)r17  = write data to DS1820
                                        ; (rout)dsin = read data from DS1820
                                        ; (rdel)r18
                                        ; (port)PORTB
                                        ; ==================================
FS_DS1820Read:                          ;
        ldi     r17,    0b11111111      ; all dataout=0xFF for read
FS_DS1820Write:                         ; r17=command
        ldi     r18,    8               ; BitCount=8
L_DS1820Next1:                          ;
        sbi     DDRB,   dspin           ; set dspin as output
        cbi     PORTB,  dspin           ; pull down dspin
```

```
        ldi     r16,    2               ; 2us delay
        rcall           F_usDelay       ; -
        ror     r17                     ; shift right r17.0->carryflag
        brcc            L_DS1820Send0   ; if(carry flag==0) leave line low
        cbi     DDRB,   dspin           ; else pull up dspin go high
L_DS1820Send0:                          ; 10us delay
        ldi     r16,    10              ;
        rcall           F_usDelay       ;
        clc                             ; clear carry flag
        sbic    PINB,   dspin           ; skip if dspin==0
        sec                             ; - set carryflag
        ror             r19             ; get input data bit to ds_in
        ldi     r16,    49              ; 49us delay
        rcall           F_usDelay       ; -
        cbi     DDRB,   dspin           ; pull up dspin backto high
        ldi     r16,    2               ; 2us delay
        rcall           F_usDelay       ;
        dec     r18                     ; BitCount--
        brne            L_DS1820Next1   ; else jump L_DS1820Next1
        ret
;.....................................................................

;=====================================================================
;void main(void)
;=======================================
F_Main:                                 ; main function start here
        ldi     r16,    low(RAMEND)     ; set stack pointer
        out     SPL,    r16             ; -
        ldi     r16,    0b00000010      ; enable TIMER0 overflow interrupt(6)
        out     TIMSK,  r16             ; -
        sei                             ; global interrupt enable
        ldi     r16,    0b00000100      ; TIMER0 source = internal_clock/256
        out     TCCR0,  r16             ; -
        ldi     r16,    0b00000101      ; TIMER1 source = internal_clock/1024
        out     TCCR1B, r16             ; -

        ldi     D_7Seg, 0               ; 7segment start value
L_MainLoop:                             ;
        push    r16                     ;
        rcall           F_DS1820        ;
        mov     D_7Seg, r16             ;
        pop     r16                     ;
        rjmp            L_MainLoop      ;
;.....................................................................
```