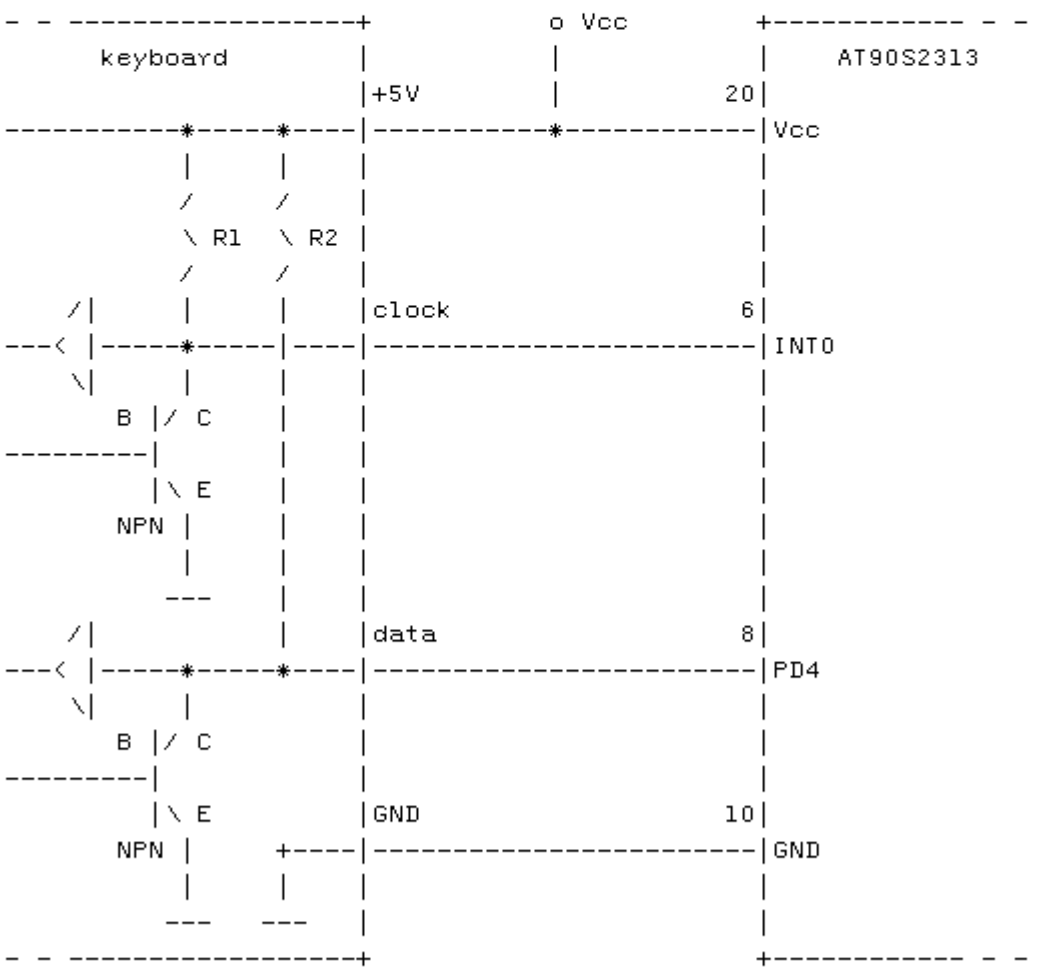
```
+-----+
| MCU | UART and External Interrupt
| U09 | การเชื่อมต่อ standard PC คีย์บอร์ดเข้ากับ MCU
+-----+
```

(1) กล่าวโดยทั่วไป

1) ในการทดลองนี้จะต่อคีย์บอร์ดมาตรฐานของคอมพิวเตอร์เข้ากับ MCU เพื่ออ่าน scan code ของคีย์บอร์ดแล้วแสดงตัวอักษรออกทาง LCD โดยมีคุณสมบัติของโปรแกรมดังนี้

- <> อินเตอร์เฟสคีย์บอร์ดพีซีเอทีมาตรฐานเข้ากับ MCU
- <> แสดงผลทาง LCD module 4 บรรทัด 30 ตัวอักษรต่อบรรทัด แสดงได้ทั้งภาษาไทย และภาษาอังกฤษ

การต่อคีย์บอร์ดกับ MCU

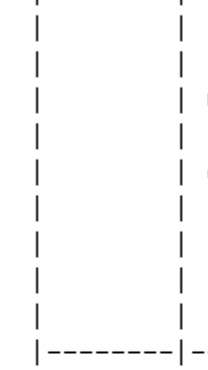
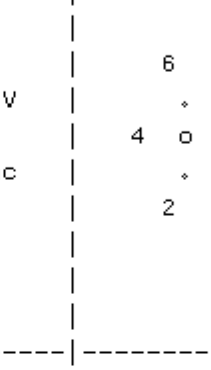


2) การเชื่อมต่อระหว่างคีย์บอร์ดกับ MCU จะมีสาย data 1 เส้น สาย clock อีกหนึ่งเส้น ภายในตัวคีย์บอร์ดมีตัวต้านทาน pullup สายสัญญาณทั้งสองนี้ทำให้ MCU หรือ คีย์บอร์ดสามารถบังคับให้สายสัญญาณทั้งสองเป็นลอจิก 0 ได้

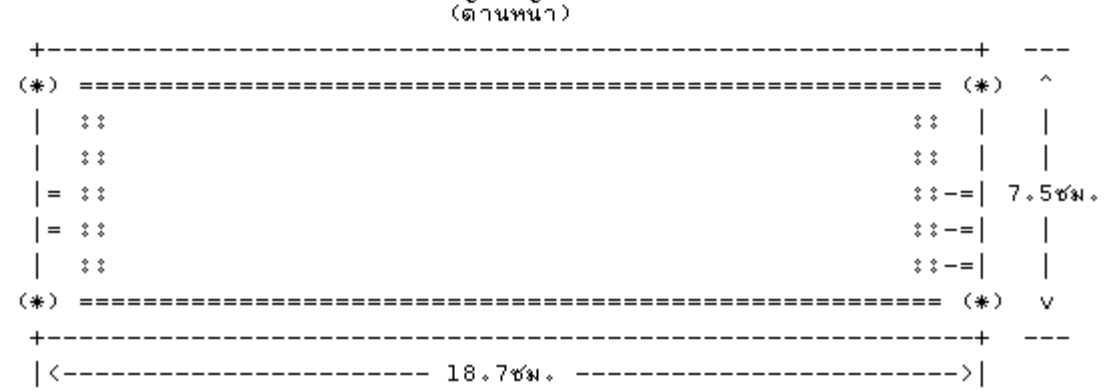
เราต่อสาย clock ของคีย์บอร์ดเข้ากับขา INT0 ของ MCU สัญญาณ clock นี้จะเป็นตัวกระตุ้นให้เกิดการอินเตอร์รัพท์ขึ้น ขา data ของคีย์บอร์ดต่อเข้ากับขา PD4 ของ MCU เราจะรับข้อมูล scan code แต่ละบิต เข้ามาทาง PD4 นี้

3) หัวต่อของคีย์บอร์ดมีสองแบบคือ 5-pin DIN connector แบบ "5D" และหัวต่อแบบเล็กหกขา six-pin mini DIN ทั้งสองแบบนำมาใช้งานได้เหมือนกัน แต่จำนวนและตำแหน่งของขาต่อ จะแตกต่างกันดังนี้

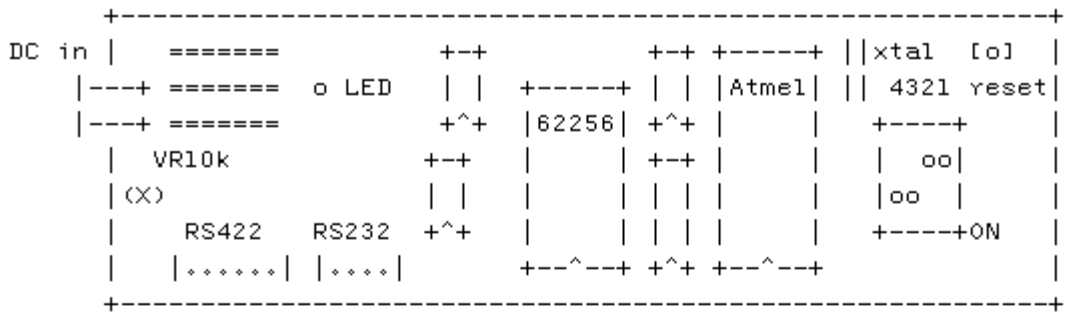
ผังการต่อหัวสาย AT keyboard (แจ็คตัวเมีย,มองจากด้านหน้า)

SIGNAL	DIN41524 Female at computer 5 pin DIN 180'	6 pin mini DIN PS2 style female at computer
		
Clock	1	5
Data	2	1
NC	3	2 and 6
GND	4	3
+5V	5	4
Shield	shell	shell

4) สำหรับ LCD module ที่เรานำมาใช้เป็นของ บริษัท อีทีที แสดงผลได้ทั้งโหมดรูปภาพ และโหมดข้อความ 32 ตัวอักษรต่อหนึ่งบรรทัด มีทั้งหมด 4 บรรทัด ในที่นี้เราจะใช้โหมดข้อความโดยต้องเซ็ทดิพสวิตช์ ที่อยู่ด้านหลังบอร์ดแอลซีดี ตามคู่มือที่บริษัทให้มา



(ด้านหลัง)



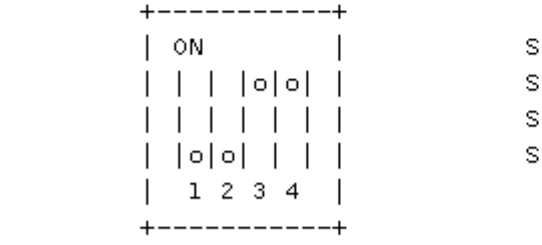
การปรับ DIP switch (ตัวที่อยู่ติดกับ crystal) จะเป็นการปรับโหมดการทำงาน ของ LCD และ อัตราการส่งข้อมูล (Baud rate) ดังตารางต่อไปนี้

SW1	SW2	Description
OFF	OFF	Charactor mode , Demo mode OFF
OFF	ON	Charactor mode , Demo mode ON
ON	OFF	Graphic mode , Demo mode OFF
ON	On	Graphic mode , Demo mode ON

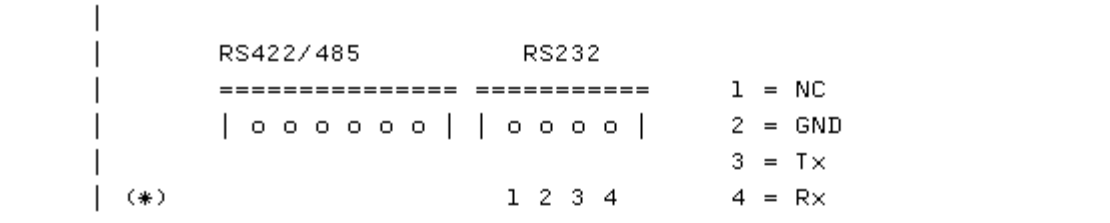
DIP switch 3,4 จะเป็นตัวตั้งอัตราการส่งข้อมูล ระหว่าง LCD module กับ MCU ดังนี้

SW3	SW4	Baud rate(charactor)	Baud rate(graphic)
OFF	OFF	1200	1200
OFF	ON	2400	2400
ON	OFF	4800	9600
ON	ON	9600	19200

ในที่นี้เราจะตั้ง DIP switch ให้อยู่ในตำแหน่งต่างๆ ดังรูปต่อไปนี้

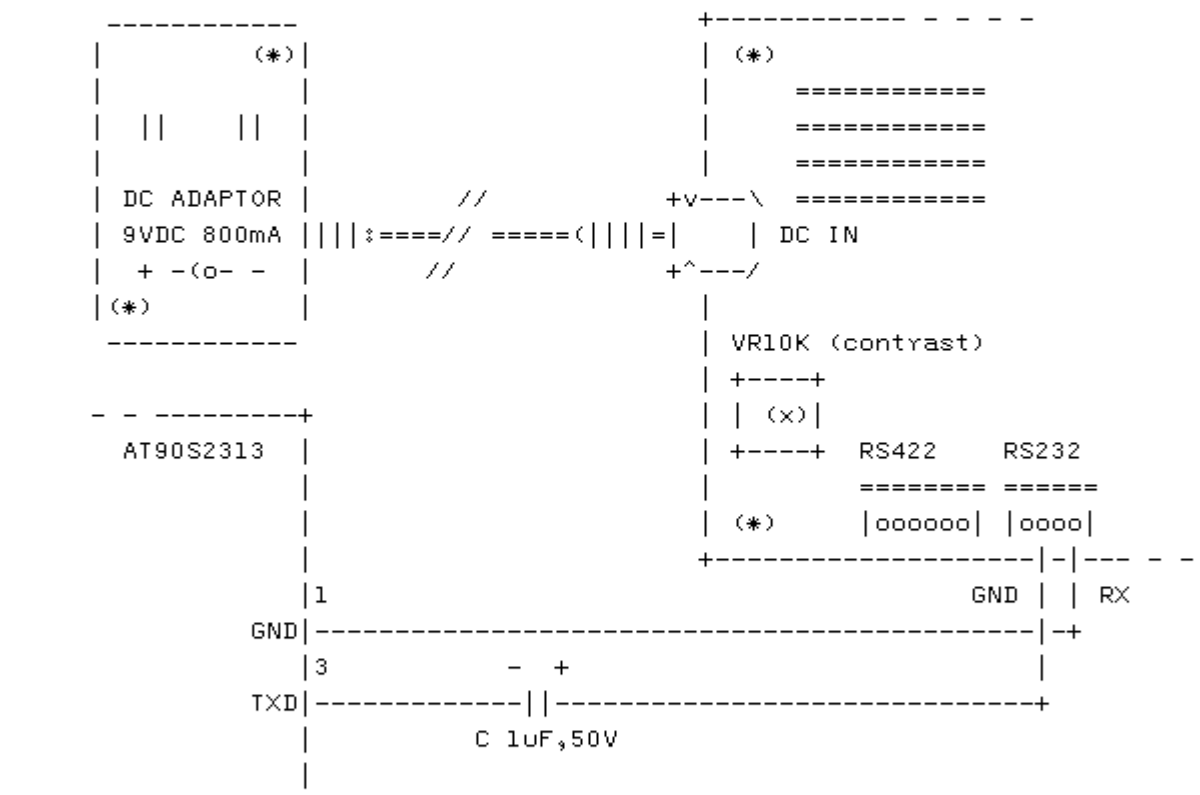


ซึ่งเราตั้งให้ LCD แสดงผลในโหมดข้อความ ปิดโหมดสาธิต(demo mode) และให้ค่าอัตราการส่งข้อมูลอนุกรม เท่ากับ 9600 baud เราจะใช้คอนเนคเตอร์ RS232 ของ LCD module ในการรับข้อมูลจาก MCU โดยหัวต่อ RS232 มีผังการต่อสายดังนี้



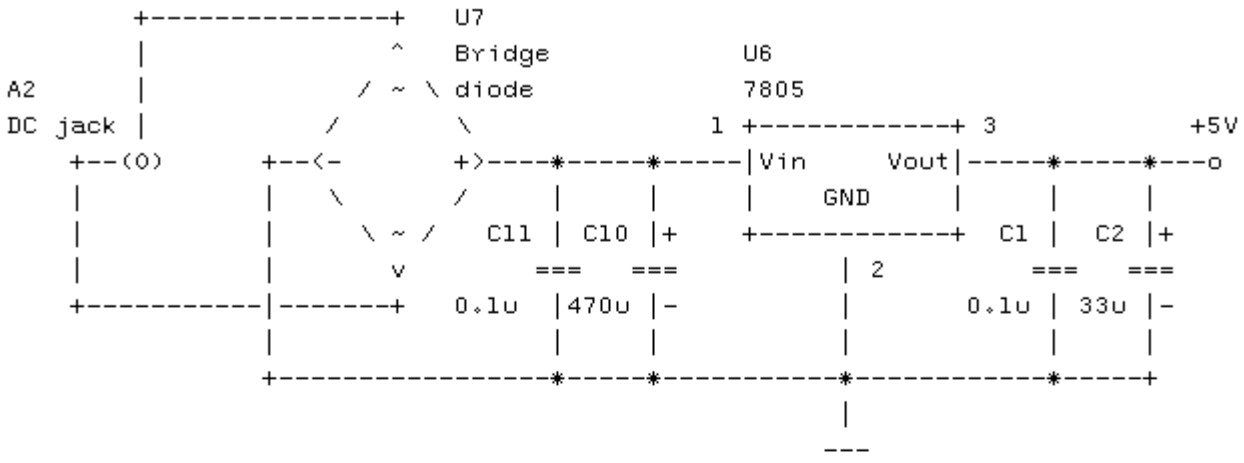
โดยเราจะใช้เฉพาะขา 2(GMD) และขา 4(Rx) เท่านั้น เนื่องจากชุด LCD ที่บริษัทอีทีที ส่งมาให้เรา จะไม่มีสายต่อมาให้ จึงต้องทำเองโดยใช้สายไฟธรรมดาบัดกรีเข้ากับขาทั้งสอง ที่ได้กล่าวมา หรืออาจจะใช้สายแข็งพันกับหลักของตอนเนคเตอร์ก็ได้ แต่ถ้าจะให้สวยงามก็ตง ต้องซื้อหัวต่อมาเข้าสายเอง

การต่อสายจาก MCU มายังแผง LCD
==========

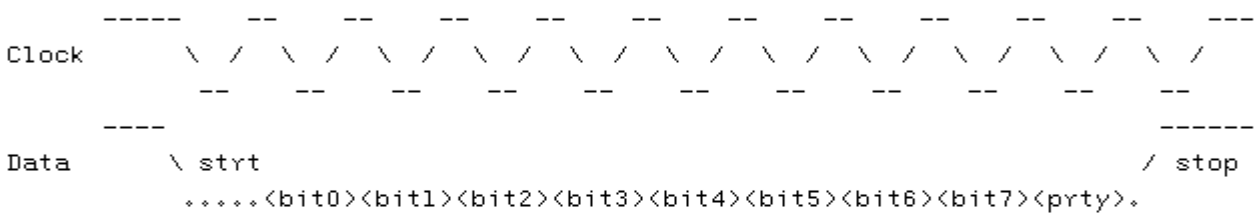


+--------+ ชุด LCD จะต้องใช้แหล่งจ่ายไฟแยกกับ MCU เพราะเราจะกลับเฟส
| คำเตือน | สัญญาณที่ออกจาก MCU โดยต่อ GND ของ MCU เข้ากับ RX ของบอร์ด LCD และ
+--------+ ต่อขา TXD ที่ใช้ส่งข้อมูลออกจาก MCU ไปเข้า GND ของบอร์ด LCD นอกเหนือ จากนั้น เราจะต้องต่อตัวเก็บประจุค่า 1uF 50 โวลต์ อนุกรมกับทาง เดินของข้อมูลอนุกรม เพื่อปรับระดับสัญญาณจาก MCU ให้เหมาะสม กับระดับที่บอร์ด LCD ต้องการ สำหรับแหล่งจ่ายไฟที่ใช้ป้อนให้บอร์ด LCD จะใช้ DC adaptor 9V หรืออาจจะใช้ AC adaptor 6V ก็ได้ เพราะตัว ของ LCD module จะมีบริดจ์เรคติไฟร์ พร้อมทั้งไอซีเรคกูเลตเตอร์ 3 ขา เบอร์ 7805 อยู่ข้างในเรียบร้อยแล้ว

วงจรแหล่งจ่ายไฟบนบอร์ด LCD
==========



(2) การส่งข้อมูลจากคีย์บอร์ดไปยัง MCU



ข้อมูลจะเริ่มต้นด้วย start bit(strt) ซึ่งเป็นลอจิก 0 ตามด้วยบิตข้อมูล 8 บิต,พาริตี้หนึ่งบิต และ stop bit ซึ่งเป็นลอจิก 1 เสมอ ข้อมูลที่ถูกส่งออกมาที่ถูกต้องจะตรงกับช่วงลอจิก 0 ของ clock คีย์บอร์ดมักจะสร้างสัญญาณ clock ที่มีเวลา 30-50us สำหรับลอจิก 1 และ 30-50us สำหรับลอจิก 0

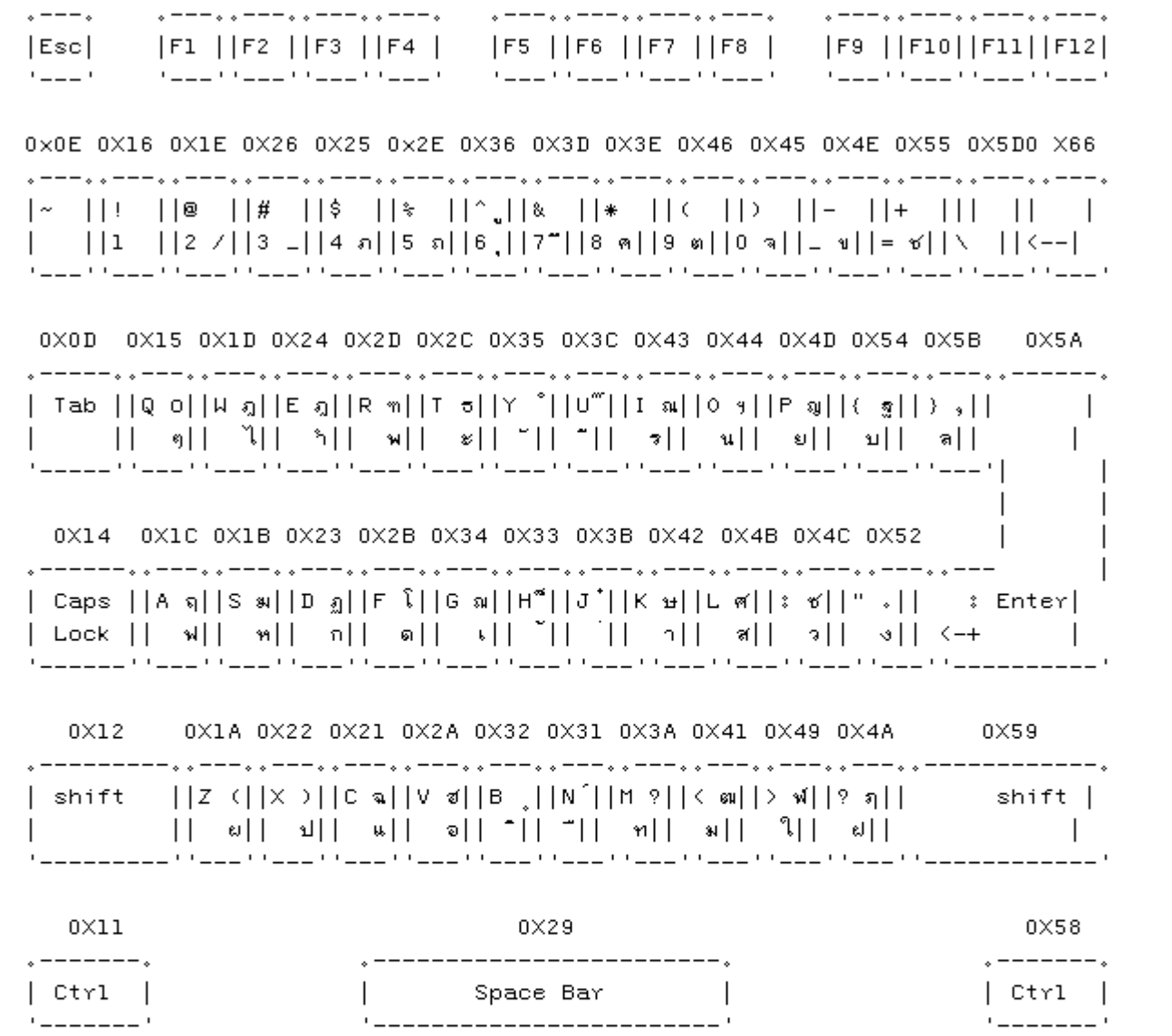
(3) สแกนโค้ด(scan code)

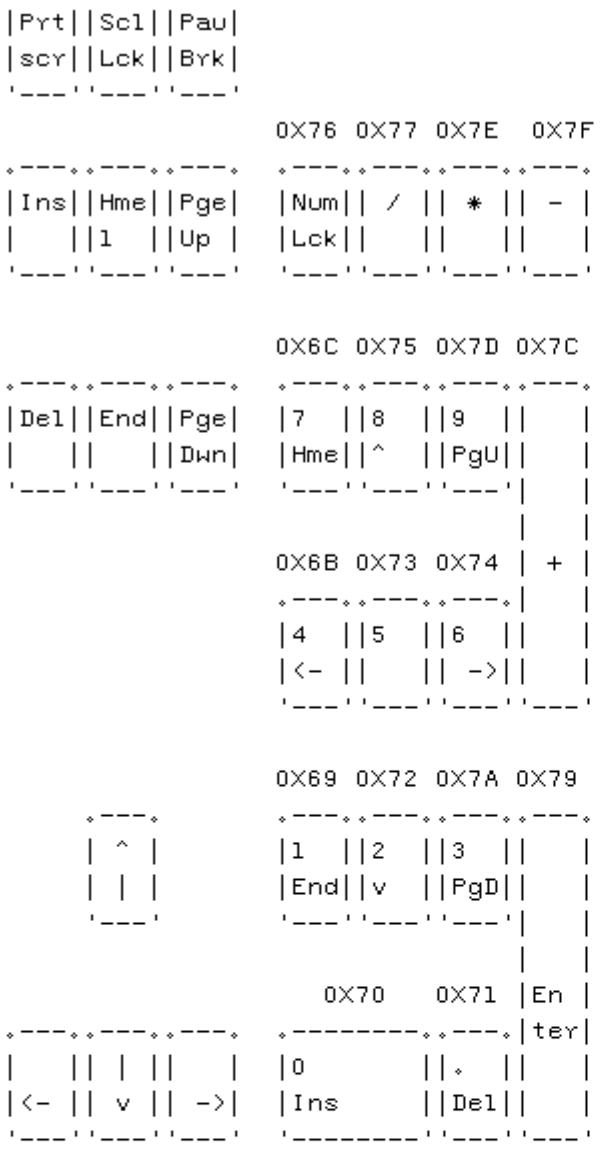
คีย์บอร์ดจะกำหนดสแกนโค้ดให้สัมพันธ์กับคีย์ เมื่อคีย์ถูกกด สแกนโค้ดจะถูกส่งออกมา เมื่อคีย์ถูกปล่อย จะเกิดรหัส "break" ($F0) ตามด้วยสแกนโค้ด คีย์ส่วนใหญ่จะมีสแกนโค้ดเท่ากับหนึ่งไบต์ แต่บางคีย์เช่น Home, Insert และ Delete จะมีรหัสสแกนโค้ดส่วนขยาย (extended scan code) จำนวน 2 ถึง 5 ไบต์ โดยไบต์แรกจะเป็น $E0 เสมอ และตามด้วยรหัส "break" เช่น E0 F0 xx .. คีย์บอร์ด จะมีสแกนโค้ดจำนวน 3 ชุด โดยชุดที่ 2 จะเป็นชุดที่ใช้งานตามปกติ

วิธีการรับข้อมูลก็คือ เราจะเก็บค่าที่ออกมาจากสาย data ตรงตำแหน่งขอบขานำของขา clock แต่ละลูก โดยการต่อขา clock เข้ากับขา INT0 ของ MCU ฟังค์ชั่นรองรับการอินเตอร์รัพท์จะทำงานทุกครั้งที่ขอบขานำของขา clock

หลังจากที่เราได้ข้อมูลครบทุกบิตแล้ว ก็จะนำข้อมูลที่ได้ไปถอดรหัส ให้เป็นรหัสแอสกี้ (ASCII) อีกครั้งหนึ่ง การแมปสแกนโค้ดกับรหัสแอสกี้ จะใช้วิธีเปิดดูตาราง ทั้งหมด 4 ตารางแบ่งเป็นภาษาไทย,ภาษาอังกฤษ ในแบบ shifted charactor) และ un-shifed charactor

ค่า scan code สำหรับปุ่มกดแต่ละปุ่ม
==========




Prt Scl Pau
scr Lck Brk
0X76 0X77 0X7E 0X7F
Ins Hme Pge
1 Up
Num / * -
Lck
0X6C 0X75 0X7D 0X7C
Del End Pge
Dwn
7 8 9
Hme ^ PgU
0X6B 0X73 0X74 +
4 5 6
<- ->
0X69 0X72 0X7A 0X79
^
1 2 3
End v PgD
0X70 0X71 En
ter
0 .
Ins Del
<- v ->
Ins Del

1) เริ่มต้นด้วยส่วนของ data segment เราได้กำหนด(define) ค่าต่างๆ สำหรับโปรแกรม ดังนี้

```
.DSEG                                   ; data segment begin hear
                                        ; ==============
                                        ; register usage
                                        ; ==============
        .def    ScanCode    = r22       ;
        .def    BitCounter  = r23       ;
        .def    KeyStatus   = r24       ; KeyStatus mapping  7 6 5 4 3 2 1 0
                                        ; logic 1/0          | | |
                                        ;   break/make    ---+ | |
                                        ;   shift/unshift -----+ |
                                        ;    thai/english -------+
```

<> ScanCode ใช้เก็บค่า scan code ที่ได้จากคีย์บอร์ด
<> BitCounter ใช้นับจำนวนบิตของ data ที่รับจากคีย์บอร์ด
<> KeyStatus ใช้บอกสภาวะของคีย์บอร์ดโดยฝังของแต่ละบิต เป็นดังนี้

bit 7 - 1 หมายถึง break code
จะเกิดขึ้นเมื่อมีการปล่อยหลังจากการกดคีย์
- 0 หมายถึง make code
จะเกิดขึ้นเมื่อมีเริ่มกดคีย์
bit 6 - 1 หมายถึงปุ่ม shift ด้านขวาหรือ
ด้านซ้าย ถูกกด
- 0 หมายถึงปุ่ม shift ด้านขวาและ
ด้านซ้ายไม่ถูกกด
bit 4 - 1 หมายถึง ภาษาไทย
- 0 หมายถึง ภาษาอังกฤษ
- การเปลี่ยนคีย์ระหว่างภาษาไทยหรือภาษาอังกฤษ
ทำได้โดยการกดปุ่มที่มีเครื่องหมาย ~ (ปุ่ม
นี้มีค่า scan code เท่ากับ 0x0E)

2) ฟังค์ชั่น main เรากำหนดค่าเริ่มต้นให้กับ ScanCode,BitCounter และ KeyStatus ให้เท่ากับ 0

```
;***************************************************************************
;void main(void)
;***************************************************************************
F_Main:                                 ; main function start here
        ldi     r16,    low(RAMEND)     ; set stack pointer
        out     SPL,    r16             ; -
        ldi     ScanCode,       0       ; preset ScanCode
        ldi     BitCounter,     0       ; preset BitCounter
        ldi     KeyStatus,      0       ; preset BreakCode
```

<> จากนั้นทำการ initial INT0 โดย pullup ขา INT0 (เพราะเราใช้ขา INT0 เป็นขาอินพุตสำหรับการอินเตอร์รัพท์) แล้วกำหนด MCUCR เพื่อให้การอินเตอร์รัพท์ INT0 เกิดขึ้นที่ขอบขาลงของพัลส์ที่เข้ามาทางขา INT0 แล้วตั้งค่ารีจิสเตอร์ GIMSK ให้ enable อินเตอร์รัพท์ INT0 และ enable global interrupt ด้วยคำสั่ง sei

```
                                ; =========
                                ; INT0 init
                                ; =========
                                ;
        sbi     PORTD,  2       ; pullup INT0 pin
        ldi     r16,    0b00000010      ; falling edge generates an interrupt
        out     MCUCR,  r16     ; -
        ldi     r16,    0b01000000      ; INT0 enable
        out     GIMSK,  r16     ; -
        sei                     ; enble global interrupt
```

<> หลังจากนั้นเรามา init ในส่วนของ LCD โดยคำสั่ง(command) ของ LCD มีอยู่หลายคำสั่งดังตารางต่อไปนี้

LCD command

คำสั่ง	หน้าที่
0x04	Cursor ON
0x05	Cursor OFF
0x06	Reset LCD
0x08	Back space
0x1B	ESC

คำสั่งแรกที่เราส่งไปยัง LCD คือคำสั่ง RESET โดยเราใส่รหัสไว้ใน รีจิสเตอร์ r16 เพื่อผ่านค่า 6 นี้เข้าไปฟังค์ชั่น F_Serial0ut ฟังค์ชั่นนี้ก็จะทำหน้าที่ส่งรหัสนี้ ออกไปยัง LCD ทำให้เกิดการรีเซตหน้าจอ LCD จะถูก clear ตัวอักษรถูกลบทิ้งทั้งหมด เคอร์เซอร์จะหายไป

```
                                ; ========
                                ; LCD init
                                ; ========
        ldi     r16,    6       ; RESET LCD command
        rcall           F_Serial0ut     ; -
        ldi     r16,    10      ; 10ms delay
        rcall           F_msDelay       ; -
```

<> ในส่วนของ MainLoop จะเป็นหลูปที่วนทำงานตลอดเวลา โดยเริ่มจากตรวจสอบว่าจำนวนบิตข้อมูลที่เข้ามาจาก keyboard ครบ 11 บิตหรือยัง ถ้ายังไม่ครบก็จะวนอยู่ในหลูปนี้ ฟังค์ชั่นที่ update ค่า BitCounter ก็คือ อินเตอร์รัพท์ฟังค์ชั่น GetKey ซึ่งทำหน้าที่รับข้อมูลจากคีย์บอร์ด ซึ่งการทำงานของฟังค์ชั่นนี้เราจะกล่าวถึงในภายหลัง

```
L_MainLoop:                             ;
        cpi     BitCounter,     11      ; wait until BitCounter==11
        brne                    L_MainLoop      ;
```

เมื่อได้จำนวนบิตครบ 11 บิตแล้ว ก็จะทำการตรวจสอบว่า scan code ที่รับมา
จากคีย์บอร์ดเป็น Left shift(0x59),Right shift(0x12) หรือ
Break code(0xF0) โดย

- ถ้ากดปุ่ม Left shift หรือ Right shift จะทำให้
  ค่า KeyStatus บิตที่ 6 เกิดการ toggle คือถ้าเดิมเป็นลอจิก 0
  ก็จะกลายเป็น 1 หรือถ้าเดิมเป็น 1 ก็จะกลายเป็น 0 ด้วยคำสั่ง
  eor (exclusive OR)

```
                                        ; ==============
                                        ; Left shift key
                                        ; ==============
        cpi     ScanCode,       0x59    ; if(ScanCode==RightShift)
        brne                    L_MainExit0     ; - else goto L_MainSkip0
        ldi     r16,            0b01000000      ; toggle shift/UnShift
        eor     KeyStatus,      r16             ; -
        cbr     KeyStatus,      0b10000000      ; reset BreakCode
        rjmp                    L_MainSkip1     ;
```

- ถ้ากดปุ่มแล้วปล่อยจะได้ Scan Code เป็นรหัส Break code(0xF0)
  KeyStatus บิต 7 จะถูก set ด้วยคำสั่ง sbr(set bit
  in register)

```
                                        ; ==========
                                        ; Break code
L_MainExit0:                            ; ==========
        cpi     ScanCode,       0xF0    ; if(ScanCode==BreakCode)
        brne                    L_MainExit2     ; - else L_MainExt0
        sbr     KeyStatus,      0b10000000      ; - then BreakCode=1
        rjmp                    L_MainSkip1     ;
```

- ถ้ากดปุ่ม Right shift ก็จะส่งผลต่อค่า StatusKey เหมือนกับการ
  กดปุ่ม Left shift

```
                                        ; ===============
                                        ; Right shift key
L_MainExit2:                            ; ===============
        cpi     ScanCode,       0x12    ; if(ScanCode==RightShift)
        brne                    L_MainSkip0     ; - else goto L_MainSkip0
        ldi     r16,            0b01000000      ; toggle shift/UnShift
        eor     KeyStatus,      r16             ; -
        cbr     KeyStatus,      0b10000000      ; reset BreakCode
        rjmp                    L_MainSkip1     ;
```

- จากนั้นก็จะทำการตรวจสอบว่าบิต 7 ที่แสดง Break code มีค่าเป็น
ลอจิก 1 หรือไม่ ถ้าไม่เป็น 1 ก็จะกระโดดข้ามไปที่ L_MainSkip1 เพื่อรอ
รับ scan code ใหม่ ที่ทำอย่างนี้เพราะไม่ต้องการให้เกิดการ repeat ของ
คีย์บอร์ดขึ้น คือการกดปุ่มแช่ไว้จะไม่ทำให้เกิดการพิมพ์ซ้ำ แต่ถ้าหากบิต
7 เป็นลอจิก 1 แสดงว่าปุ่มถูกกดแล้วปล่อยก็จะทำการแปลง scan code ไปเป็น
ASC II code ต่อไป โดยจะเริ่มจากการตรวจดูว่าปุ่มเปลี่ยนรหัส
ภาษาไทย/อังกฤษ คือปุ่ม [~] ถูกกดหรือไม่ ถ้าถูกกดก็จะทำการ toggle บิต
ที่ 5 ของ StatusKey แล้วกระโดดไป L_MainSkip1 เพื่อรอรับ scan code
เข้ามาใหม่ แต่ถ้าปุ่ม [~] ไม่ถูกกดก็จะข้ามการทำงานในส่วนนี้ไปยัง
L_MainExit1 เพื่อเปิดดูตารางรหัสแอสกี้

```
L_MainSkip0:                                ;
                                            ; ================
                                            ; Break code check
                                            ; ================
        sbrs    KeyStatus,      7           ; if(BreakCode)
        rjmp                    L_MainSkip1 ; - else L_MainSkip1
                                            ;
                                            ; ====================
                                            ; Langauge switch key
                                            ; ====================
        cpi     ScanCode,       0x0E        ; if(ScanCode==LangSwitch)
        brne                    L_MainExit1 ; - else goto L_MainSkip0
        ldi     r16,            0b00100000  ; toggle shift/UnShift
        eor     KeyStatus,      r16         ; -
        cbr     KeyStatus,      0b10000000  ; reset BreakCode
        rjmp                    L_MainSkip1 ;
```

- ในส่วนของการเปิดตารางเพื่อแปลง scan code ไปเป็น ASC II code จะเริ่มจากตรวจสอบว่าบิต 5 ของ KeyStatus เป็น 0 หรือไม่ถ้าเป็น 1 แสดงว่าเป็นภาษาไทย ก็จะกระโดดไป L_MainThai แต่ถ้าเป็นลอจิก 0 ก็จะทำในส่วนของภาษาอังกฤษ คำสั่ง sbrc KeyStatus,5 หมายถึง ให้กระโดดข้ามคำสั่งถัดไปจากนี้ 1 คำสั่งถ้าบิตที่ 5 ของ KeyStatus เป็นลอจิก 0

```
L_MainExit1:                                ;
                                            ; ============
                                            ; Table lookup
                                            ; ============
        sbrc    KeyStatus,      5           ; if(Lang==english) then skip
        rjmp                    L_MainThai  ; - else goto L_MainThai
                                            ;--------
```

- ในส่วนของภาษาอังกฤษนี้ จะเริ่มด้วยการเซ็ตบิตที่ 6 ของ KeyStatus ว่าปุ่ม shift ซ้ายและ/หรือขวา ถูกกดหรือไม่ถ้าไม่ถูกกดก็จะกระโดดไปทำที่ L_MainEngUnShift แต่ถ้าถูกกด ก็จะโหลดแอดเดรสของชุดตัวอักษร EnglShift เข้าไปไว้ในรีจิสเตอร์ Z แล้วกระโดดไป L_MainLookUp เพื่อเปิดดูรหัสจากตารางนี้ต่อไป

```
        sbrs    KeyStatus,      6                       ; if(shift)
        rjmp                    L_MainEngUnShift        ; then goto L_MainUnShif
        ldi     ZL,             LOW(T_EngShiftChar*2)   ; else get shift
        ldi     ZH,             HIGH(T_EngShiftChar*2)  ; -
        rjmp                    L_MainLookUp            ;
```

- ถ้าปุ่ม shift ไม่ถูกกดก็จะโหลดแอดเดรสของตารางชุด EngUnShift เข้ามาไว้ในรีจิสเตอร์ Z แล้วกระโดดไป L_MainLookUp

```
L_MainEngUnShift:                                       ;
        ldi     ZL,             LOW(T_EngUnshiftChar*2) ; get unshift
        ldi     ZH,             HIGH(T_EngUnshiftChar*2); -
        rjmp                    L_MainLookUp            ;
```

- สำหรับกลุ่มของภาษาไทยจะทำงานคล้ายกับในกรณีภาษาอังกฤษ โดยเริ่มจากการตรวจดูว่าปุ่ม shift ถูกกดหรือไม่ ถ้าถูกกดก็จะโหลดแอดเดรสของตาราง ThaiShift มาเข้ารีจิสเตอร์ Z แต่ถ้าไม่ถูกกดก็จะโหลดแอดเดรสของตาราง ThaiUnShift มาเข้ารีจิสเตอร์ Z แล้วกระโดดไป L_MainLookUp เพื่อเปิดตารางดูรหัส ASCII ต่อไป

```
L_MainThai:                                             ;
        sbrs    KeyStatus,      6                       ; if(shift)
        rjmp                    L_MainThaiUnShift       ; then goto L_MainThaiUn
        ldi     ZL,             LOW(T_ThaiShiftChar*2)  ; else get shift
        ldi     ZH,             HIGH(T_ThaiShiftChar*2) ; -
        rjmp                    L_MainLookUp            ;

L_MainThaiUnShift:                                      ;--------
        ldi     ZL,             LOW(T_ThaiUnshiftChar*2)        ; get unshift
        ldi     ZH,             HIGH(T_ThaiUnshiftChar*2)       ; -
                                                ;----------------
```

- การนำ scan code มาแปลงเป็นรหัส ASC II จะเริ่มจาก การรีเซ็ตบิต BreakCode ใน KeyStatus แล้วเริ่มต้นหารหัสจากตาราง โดยโหลดค่า scan code จากตารางเข้ามาด้วยคำสั่ง lpm แล้วเก็บไว้ใน r16 จากนั้นก็ตรวจสอบว่าต้นจนหมดตารางหรือยังด้วยคำสั่ง cpi r16,0 เพราะว่าท้ายสุดของแต่ละตารางเราจะใส่รหัส 0 ไว้ ถ้าต้นจนหมดตารางแล้วยังไม่เจอ scan code ที่ตรงกับ scan code จากคีย์บอร์ด ก็จะกระโดดไป L_MainSkip1 เพื่อไปรอรับค่า scan code จากคีย์บอร์ดตัวต่อไป แต่ถ้ายังไม่สุดตารางก็จะนำ scan code จากตารางที่เก็บไว้ใน r16 มาเปรียบเทียบกับ scan code ที่รับมาจากคีย์บอร์ด(ScanCode) ถ้าเท่ากันก็แสดงว่าเจอตัวอักษร(ASCII) ที่ต้องการแล้ว ก็จะกระโดดไป L_MainSkip เพื่อดึงรหัสตัวอักษร ASCII ออกมาจากตาราง แต่ถ้ารหัส scan code ยังไม่ตรงกันก็จะเพิ่มค่ารีจิสเตอร์ Z ขึ้นอีก 2 ไบต์ เพราะว่าการเก็บข้อมูลในตารางเราเก็บเป็นไบต์

เช่น .db 0x0d, 0 ,0x0e,'|',0x15,'q' เราเก็บ scan code สลับกับ ASCII ไบต์เว้นไบต์ scan code แต่ละตัวจะคู่กับรหัส ASCII ของตัวมันเอง รีจิสเตอร์ ZL ก็คือ r30 และ ZH ก็คือ r31 คำสั่ง adiw ZL,2 จะบวกค่า 2 ให้กับ ZL ถ้าบวกแล้วเกินก็จะทดไปยัง ZH คำสั่ง adiw ย่อมาจากคำว่า Add Immediate to Word การทำงานสำหรับคำสั่ง adiw ZL,2 จะเป็นดังนี้ ZH:ZL <- ZH:ZL+2

```
L_MainLookup:                                           ;
        cbr     KeyStatus,      0b10000000      ; reset BreakCode
        lpm                                     ; get table data
        mov     r16,            r0              ;
        cpi     r16,            0               ; if(end table)
        breq                    L_MainSkip1     ; - then goto L_MainSkip1
        cp      r0,             ScanCode        ; if(table_data==ScanCode)
```

```
        breq                    L_MainSkip      ; - then L_MainSkip         9-13
        adiw    ZL,             2               ; else -
        ijmp                    L_MainLookup    ;      -
```

- เมื่อค้นตารางเจอรหัสที่ต้องการแล้ว ก็จะนำรหัส ASCII ออกมา โดยบวก 1 เข้ากับรีจิสเตอร์ ZL เพื่อเลื่อนไปชี้รหัส ASCII จากนั้นใช้คำสั่ง lpm เพื่อดึงค่า ASCII จากตารางเข้ามาไว้ใน r0 แล้วย้ายข้อมูลจาก r0 เข้าไปเก็บไว้ใน r16 จากนั้นก็เรียกฟังก์ชั่น F_Serial0ut ให้ทำงานส่งรหัส ASCII ไปยัง LCD module ให้แสดงตัวอักษรนั้น

```
L_MainSkip:                             ;
        adiw    ZL,             1       ; get ASCII
        lpm                             ; -
        mov     r16,            r0      ; Serial0ut to LCD
        rcall                   F_Serial0ut     ; -
```

- โปรแกรมส่วนนี้จะทำหน้าที่ reset ค่า BitCounter และ ScanCode ให้เป็น 0 แล้วกระโดดกลับไปวนทำงานที่ L_MainLoop

```
L_MainSkip1:                            ;
        ldi     BitCounter,     0       ; reset BitCounter
        ldi     ScanCode,       0       ; reset ScanCode
        rjmp                    L_MainLoop      ; eternal loop
```

ฟังก์ชั่น Serial0ut
==========
ฟังก์ชั่นนี้ทำหน้าที่ นำข้อมูล 8 บิต ที่ส่งเข้ามาทางรีจิสเตอร์ r16 ออกทางพอร์ตอนุกรมของ MCU ที่ขา 3 คือ TXD เราโปรแกรมให้หน่วย UART(Universal Asynchronous Receiver and Transmitter) ให้ส่งข้อมูลออกไปโดยมี buard rate เท่ากับ 9600 ซึ่งจะเท่ากับ buard rate ของ LCD module ที่เราตั้งด้วย dip switch ส่วนการตั้งค่านี้ของ MCU เราจะต้องโปรแกรมให้รีจิสเตอร์ UBRR(UART Buard Rate Register) มีค่าเท่ากับ 25 ถ้าเราต้องการให้เป็นค่าอื่น ก็ให้ดูค่า UBRR ได้จากตาราง(หรือดูตาต้าชีท หน้า 44) อย่าลืมว่าจะต้องตั้ง buard rate ของ LCD module ให้ตรงกันด้วย ไม่อย่างนั้น จะติดต่อกันไม่ได้

```
;=============================================================================
```

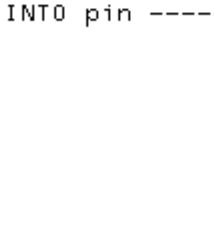
```
; Function      Serial0ut
;               UART transmission - 8bit,no-parity,9600baud
; (rin)  r16 = charactor to send
; (rout) none
; (port) PD1 as TxD
;=======================================
F_Serial0ut:                          ;
        push    r16                   ;
        sbi     UCR,    TXEN          ; transmitter enable
        ldi     r16,    25            ; 9600 buard at 4MHz clock

        out     UBRR,   r16           ; -
        pop     r16                   ;
        out     UDR,    r16           ; data to be send out
        ret                           ;
;.....................................................................................
```

UBRR setting at 4MHz crystal frequency

Buard Rate	UBRR value	%Error
2400	103	0.2
4800	51	0.2
9600	25	0.2
14400	16	2.1
19200	12	0.2
28800	8	3.7
38400	6	7.5
57600	3	7.8
76800	2	7.8
115200	1	7.8

ฟังค์ชั่น GetKey
============
ฟังค์ชั่นนี้ทำหน้าที่รับ interrupt ที่มาจากคีย์บอร์ด เข้ามา
ทางขา 6,INT0 เมื่อสัญญาณ clock จากคีย์บอร์ดตกจากลอจิก 1
เป็นลอจิก 0 จะทำให้เกิด INT0 interrupt ขึ้น MCU ก็จะกระโดด
เข้ามาทำงานที่ I_GetKey



เราจะเก็บข้อมูลไว้ใน ScanCode โดยมี BitCounter เป็นตัวนับจำนวน
บิตของข้อมูลที่ส่งมาจากคีย์บอร์ด การกดปุ่ม 1 ครั้ง จะเกิดชุด
ข้อมูล อนุกรมจำนวน 11 บิต เรียงตามลำดับการส่งข้อมูลก่อนหลัง
ได้ดังนี้

1. บิตเริ่มต้น(start bit) 1 บิต
2. ค่า scan code 8 บิต(บิต LSB มาก่อน)
3. บิตพาริตี้(parity bit) 1 บิต
4. บิตหยุด(stop bit) 1 บิต

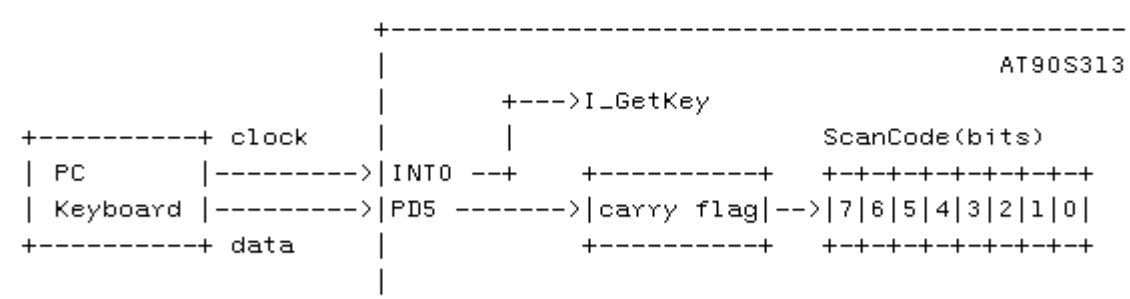
เราจะต้องรับข้อมูลทั้ง 11 บิตนี้เข้ามาแล้วตัดให้เหลือเฉพาะบิต
ที่แสดงค่า scan code คือตัดทิ้งไป 3 บิต

การทำงานเริ่มจากสั่ง cli เพื่อไม่ให้เกิดการอินเตอร์รัพท์ซ้ำ

ถ้าข้อมูลบิต 1 เข้ามา ให้กระโดดไปที่ L_GetKeySkip1 เพื่อข้าม
บิตนี้ไป แล้วเพิ่มค่า BitCounter
2 เข้ามา ให้กระโดดไป L_GetKeySkip1

```
I_GetKey:                           ;
        cli                         ; disable global interrupt
        cpi     BitCounter,1        ; if(BitCounter<1) then BitCounter++
        brlt    L_GetKeySkip1       ; -
        cpi     BitCounter,9        ; if(BitCounter<9) then get data
        brlt    L_GetKeySkip0       ; -
        cpi     BitCounter,11       ; if(BitCounter<11) then BitCounter++
        brlt    L_GetKeySkip1       ;
        sei                         ; -
        reti                        ; -
```

- หน่วงเวลาไป อีก 4 cycle ด้วยคำสั่ง nop แล้ว clear
carry flag จากนั้นก็เช็คว่าข้อมูลที่เข้ามาขา PD5 นั้น เป็น 0
หรือไม่ ถ้าไม่ก็ข้ามไปด้วยคำสั่ง sbic PIND,5 แต่ถ้าข้อมูล
เป็นที่เข้ามาขา PD5 เป็นลอจิก 1 ก็ให้ carry flag เท่ากับ 1
ด้วยคำสั่ง sec จากนั้น หมุนขวาค่า ScanCode แล้วเพิ่มค่า
BitCounter



จะเห็นว่าวิธีการนี้จะทำให้ข้อมูลจากขา PD5 เข้าไปอยู่ใน carry
flag ก่อนแล้วจึงใช้คำสั่ง ror หมุนค่าใน carry flag ให้เข้า
ไปใน ScanCode ทางบิต 7 เมื่อทำครบ 8 ครั้งก็จะได้ scan code
ต่อมาเมื่อบิตที่ 10 และ 11 เข้ามาก็จะถูกข้ามไปเช่นเดียว
กันกับบิตแรก

```
L_GetKeySkip0:                          ;
        nop                             ;
        nop                             ;
        nop                             ;
        nop                             ;
        clc                             ; clear carry flag
        sbic    PIND,   5               ; if(pin_INT0 == 0) then jump
        sec                             ; set carry flag
        ror     ScanCode                ; c->R(7),R(n+1)->R(n),R(0)->c
L_GetKeySkip1:                          ;
        inc     BitCounter              ; BitCounter++
L_GetKeySkip2:                          ;
        sei                             ;
        reti                            ;
;.....................................................................................
```

(5) โปรแกรม

```
;*************************************************************************************
; AT90S2313
; PC keyboard & LCD module
; copyright (c) 2001 by Gumtorn Ruanfaigad.Allright reserved.
; ~ XTAL        = 4.0MHz
; ~ Ucc         = 5v
; ~ MCU         = AT90S2313,Atmel corperation,www.atmel.com
;*************************************************************************************

.include        "2313def.inc"           ;
.DEVICE         AT90S2313               ;
.DSEG                                   ; data segment begin hear
                                        ; ==============
                                        ; register usage
                                        ; ==============
        .def    D_7Seg      = r21       ; 7 segment value (all 3 digits)
        .def    ScanCode    = r22       ;
        .def    BitCounter  = r23       ;
        .def    KeyStatus   = r24       ; KeyStatus mapping  7 6 5 4 3 2 1 0
                                        ; logic 1/0           | | |
                                        ;   break/make     ---+ | |
                                        ;   shift/unshift -----+ |
                                        ;    thai/english -------+
                                        ;
```

```
.CSEG                                   ;                                       9-17
.ORG            0x0000                  ;
                                        ; ======================
                                        ; interrupt vector table
                                        ; ======================
        rjmp    F_main                  ; #0 RESET
        rcall   I_GetKey                ; #1 INT0
        reti                            ; #2 INT1
        reti                            ; #3 TIMER1 CAPT11
        reti                            ; #4 TIMER1 COMP1
        reti                            ; #5 TIMER1 OVF1
        reti                            ; #6 TIMER0 OVF0
        reti                            ; #7 UART,RX
        reti                            ; #8 UART,UDRE
        reti                            ; #9 UART,TX
        reti                            ; #10 ANA_COMP

;=============================================================================
;void main(void)
;=============================================
F_Main:                                 ; main function start here
        ldi     r16,    low(RAMEND)     ; set stack pointer
        out     SPL,    r16             ; -

        ldi     D_7Seg, 123             ; 7segment start,      0       ; preset
        ldi     KeyStatus,      0       ; preset BreakCode
                                        ;
                                        ; =========
                                        ; INT0 init
                                        ; =========
                                        ;
        sbi     PORTD,  2               ; pullup INT0 pin
        sbi     PORTD,  3               ; pullup INT1 pin
        ldi     r16,    0b00001010      ; falling edge generates an interrupt
        out     MCUCR,  r16             ; -
        ldi     r16,    0b11000000      ; enable INT1 and INT0
        out     GIMSK,  r16             ; -
        sei                             ; enble global interrupt
                                        ;
                                        ; ========
                                        ; LCD init
                                        ; ========
        ldi     r16,    6               ; RESET LCD command
        rcall           F_Serial0ut     ; -

        ldi     r16,    10              ; 10ms delay
        rcall           F_msDelay       ; -
        ldi     r16,    5               ; CURSOR ON command
        rcall           F_Serial0ut     ; -
        ldi     r16,    10              ; 10ms delay
```

```
        rcall           F_msDelay       ; -
        ldi     r16,    3               ; GOTO POSITION command
        rcall           F_Serial0ut     ; -
        ldi     r16,    10              ; 10ms delay
        rcall           F_msDelay       ; -
        ldi     r16,    0x5A            ; GOTO LINE 3 command
        rcall           F_Serial0ut     ; -
                                        ;--------
L_MainLoop:                                     ;
        cpi     BitCounter,     11              ; wait until BitCounter==11
        brne                    L_MainLoop      ;
                                                ; ==============
                                                ; Left shift key
                                                ; ==============
        cpi     ScanCode,       0x59            ; if(ScanCode==RightShift)
        brne                    L_MainExit0     ; - else goto L_MainSkip0
        ldi     r16,            0b01000000      ; toggle shift/UnShift
        eor     KeyStatus,      r16             ; -
        cbr     KeyStatus,      0b10000000      ; reset BreakCode
        rjmp                    L_MainSkip1     ;
                                                ; ==========
                                                ; Break code
L_MainExit0:                                    ; ==========
        cpi     ScanCode,       0xF0            ; if(ScanCode==BreakCode)
        brne                    L_MainExit2     ; - else L_MainExt0
        sbr     KeyStatus,      0b10000000      ; - then BreakCode=1
        rjmp                    L_MainSkip1     ;
                                                ; ===============
                                                ; Right shift key
L_MainExit2:                                    ; ===============
        cpi     ScanCode,       0x12            ; if(ScanCode==RightShift)
        brne                    L_MainSkip0     ; - else goto L_MainSkip0
        ldi     r16,            0b01000000      ; toggle shift/UnShift
        eor     KeyStatus,      r16             ; -
        cbr     KeyStatus,      0b10000000      ; reset BreakCode
        rjmp                    L_MainSkip1     ;
L_MainSkip0:                                    ;



                                                ; ================
                                                ; Break code check
                                                ; ================
        sbrs    KeyStatus,      7               ; if(BreakCode)
        rjmp                    L_MainSkip1     ; - else L_MainSkip1
                                                ;
                                                ; ====================
                                                ; Langauge switch key
                                                ; ====================
        cpi     ScanCode,       0x0E            ; if(ScanCode==LangSwitch)
```

```
        brne                    L_MainExit1     ; - else goto L_MainSkip0   9-19
        ldi     r16,            0b00100000      ; toggle shift/UnShift
        eor     KeyStatus,      r16             ; -
        cbr     KeyStatus,      0b10000000      ; reset BreakCode
        rjmp                    L_MainSkip1     ;
L_MainExit1:                                    ;
                                                ; ============
                                                ; Table lookup
                                                ; ============
        sbrc    KeyStatus,      5               ; if(Lang==english) then skip
        rjmp                    L_MainThai      ; - else goto L_MainThai
                                                ;--------
                                                        ;
        sbrs    KeyStatus,      6                       ; if(shift)
        rjmp                    L_MainEngUnShift        ; then goto L_MainUnShif
        ldi     ZL,             LOW(T_EngShiftChar*2)   ; else get shift
        ldi     ZH,             HIGH(T_EngShiftChar*2)  ; -
        rjmp                    L_MainLookUp            ;
L_MainEngUnShift:                                       ;
        ldi     ZL,             LOW(T_EngUnshiftChar*2) ; get unshift
        ldi     ZH,             HIGH(T_EngUnshiftChar*2); -
        rjmp                    L_MainLookUp            ;
L_MainThai:                                             ;
        sbrs    KeyStatus,      6                       ; if(shift)
        rjmp                    L_MainThaiUnShift       ; then goto L_MainThaiUn
        ldi     ZL,             LOW(T_ThaiShiftChar*2)  ; else get shift
        ldi     ZH,             HIGH(T_ThaiShiftChar*2) ; -
        rjmp                    L_MainLookUp            ;
L_MainThaiUnShift:                                      ;--------
        ldi     ZL,             LOW(T_ThaiUnshiftChar*2)        ; get unshift
        ldi     ZH,             HIGH(T_ThaiUnshiftChar*2)       ; -
                                                ;----------------
L_MainLookup:                                   ;
        cbr     KeyStatus,      0b10000000      ; reset BreakCode
        lpm                                     ; get table data
        mov     r16,            r0              ;
        cpi     r16,            0               ; if(end table)
        breq                    L_MainSkip1     ; - then goto L_MainSkip1
        cp      r0,             ScanCode        ; if(table_data==ScanCode)
        breq                    L_MainSkip      ; - then L_MainSkip
        adiw    ZL,             2               ; else -
        rjmp                    L_MainLookup    ;      -
L_MainSkip:                                     ;
        adiw    ZL,             1               ; get ASCII
        lpm                                     ; -
        mov     r16,            r0              ; Serial0ut to LCD
        rcall                   F_Serial0ut     ; -
L_MainSkip1:                                    ;
        ldi     BitCounter,     0               ; reset BitCounter
        ldi     ScanCode,       0               ; reset ScanCode
```

```
        rjmp                    L_MainLoop      ; eternal loop              9-20
;***************************************************************************
;===========================================================================
;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      Serial0ut
;               UART transmission - 8bit,no-parity,9600buad
; (vin)  vl6 = charactor to send
; (vout) none
; (port) PD1 as TxD
;=========================================
F_Serial0ut:                            ;
        push    vl6                     ;
        sbi     UCR,    TXEN            ; transmitter enable
        ldi     vl6,    25              ; 9600 buard at 4MHz clock
        out     UBRR,   vl6             ; -
        pop     vl6                     ;
        out     UDR,    vl6             ; data to be send out
        vet                             ;
;.............................................................................
```

```
;=============================================================================
; Interrupt routine : GetKey
; (vin)  none
; (vout) ScanCode,BitCounter
; (port) PD5,INT0
;=========================================
I_GetKey:                               ;
        cli                             ; disable global interrupt
        cpi     BitCounter,1            ; if(BitCounter<1) then BitCounter++
        brlt    L_GetKeySkip1           ; -
        cpi     BitCounter,9            ; if(BitCounter<9) then get data
        brlt    L_GetKeySkip0           ; -
        cpi     BitCounter,11           ; if(BitCounter<11) then BitCounter++
        brlt    L_GetKeySkip1           ;
        sei                             ; -
        veti                            ; -
L_GetKeySkip0:                          ;
        nop                             ;
        nop                             ;
        nop                             ;
        nop                             ;
        clc                             ; clear carry flag
        sbic    PIND,   5               ; if(pin_INT0 == 0) then jump
        sec                             ; set carry flag
        vor     ScanCode                ; c->R(7),R(n+1)->R(n),R(0)->c
L_GetKeySkip1:                          ;
        inc     BitCounter              ; BitCounter++
L_GetKeySkip2:                          ;
```

```
        sei                             ;
        yeti                            ;
;.....................................................................................

;=====================================================================================
; TABLE : thai/english Keyboard scan code lookup table
;=====================================================

T_EngUnshiftChar:
=================
.db     0x0d, 0 ,0x0e,'|',0x15,'q',0x16,'1',0x1a,'z',0x1a,'z',0x1b,'s'
.db     0x1c,'a',0x1d,'w',0x1e,'2',0x21,'c',0x22,'x',0x23,'d',0x24,'e'
.db     0x25,'4',0x26,'3',0x29,' ',0x2a,'v',0x2b,'f',0x2c,'t',0x2d,'r'
.db     0x2e,'5',0x31,'n',0x32,'b',0x33,'h',0x34,'g',0x35,'y',0x36,'6'
.db     0x3a,'m',0x3b,'j',0x3c,'u',0x3d,'7',0x3e,'8',0x41,','
.db     0x42,'k',0x43,'i',0x44,'o',0x45,'0',0x46,'9',0x49,'.',0x4a,'-'
.db     0x4b,'l',0x4c,'ฬ',0x4d,'p',0x4e,'+',0x52,'ฯ',0x54,'ๆ',0x55,'\'
.db     0x5a, 13,0x5b,'ฺ',0x5d,'\',0x61,'<',0x66, 8 ,0x69,'1',0x6b,'4'
.db     0x6c,'7',0x70,'0',0x71,',',0x72,'2',0x73,'5',0x74,'6',0x75,'8'
.db     0x79,'+',0x7a,'3',0x7b,'-',0x7c,'*',0x7d,'9'
.db     0x03,  5,0x04,  4,0x05,  6,0x00,0

T_EngShiftChar:
===============
.db     0x0d, 0 ,0x0e,'|',0x15,'Q',0x16,'1',0x1a,'Z',0x1a,'Z',0x1b,'S'
.db     0x1c,'A',0x1d,'W',0x1e,'2',0x21,'C',0x22,'X',0x23,'D',0x24,'E'
.db     0x25,'4',0x26,'3',0x29,' ',0x2a,'V',0x2b,'F',0x2c,'T',0x2d,'R'
.db     0x2e,'5',0x31,'N',0x32,'B',0x33,'H',0x34,'G',0x35,'Y',0x36,'6'
.db     0x3a,'M',0x3b,'J',0x3c,'U',0x3d,'7',0x3e,'8',0x41,','
.db     0x42,'K',0x43,'I',0x44,'O',0x45,'0',0x46,'9',0x49,'.',0x4a,'-'
.db     0x4b,'L',0x4c,'ฬ',0x4d,'P',0x4e,'+',0x52,'ฯ',0x54,'ๆ',0x55,'\'
.db     0x5a, 13,0x5b,'ฺ',0x5d,'\',0x61,'<',0x66, 8 ,0x69,'1',0x6b,'4'
.db     0x6c,'7',0x70,'0',0x71,',',0x72,'2',0x73,'5',0x74,'6',0x75,'8'
.db     0x79,'+',0x7a,'3',0x7b,'-',0x7c,'*',0x7d,'9'
.db     0x03,  5,0x04,  0,0x05,  6,0x00,0
```

```
T_ThaiUnshiftChar:
==================
;               ๅ       ผ       ห       ฟ       ไ       แ       ป
.db     0x15,230,0x1a,188,0x1b,203,0x1c,191,0x1d,228,0x21,225,0x22,187
;               ก       ้       ภ                       อ       ด       ะ
.db     0x23,161,0x24,211,0x25,192,0x29,' ',0x2a,205,0x2b,180,0x2c,208
;               พ       ถ       ั       ิ       ี       เ       ื
.db     0x2d,190,0x2e,182,0x31,215,0x32,212,0x33,233,0x34,224,0x35,209
;               ุ       ท       ู       ฺ       ๋       ค       ม
.db     0x36,216,0x3a,183,0x3b,232,0x3c,213,0x3d,214,0x3e,164,0x41,193
;               า       ร       น       จ       ต       ใ       ฝ
.db     0x42,210,0x43,195,0x44,185,0x45,168,0x46,181,0x49,227,0x4a,189
;               ส       ว       ย       บ       ง       ป       ช
.db     0x4b,202,0x4C,199,0x4d,194,0x4e,162,0x52,167,0x54,186,0x55,170
;                       ล
.db     0x5a, 13,0x5b,197,0x66,  8
;
.db     0x69,'1',0x6b,'4',0x6c,'7',0x70,'0',0x72,'2',0x73,'5',0x74,'6'
;
.db     0x75,'8',0x79,'+',0x7a,'3',0x7b,'-',0x7c,'*',0x7d,'9'
;
.db     0x03,  5,0x04,  4,0x05,  6,0x00,0
T_ThaishiftChar:
================
;               (       ฆ       ฤ       "       ฉ       )
.db     0x1a, 40,0x1b,166,0x1c,196,0x1d,208,0x21,169,0x22, 41
;               ฏ       ฎ                       ฮ       ไ       ฮ
.db     0x23,175,0x24,174,0x29,' ',0x2a,206,0x2b,226,0x2c,184
;               ฑ       ์       ็       ฌ       ั
.db     0x2d,177,0x31,236,0x33,231,0x34,172,0x36,217
;               ?       ฯ       ๊       ฒ
.db     0x3a, 63,0x3b,235,0x3c,234,0x41,178
;               ษ       ฌ       ฯ       ฬ       ฦ
.db     0x42,201,0x43,195,0x44,207,0x49,204,0x4a,198
;               ศ       ซ       ฆ       ฐ
.db     0x4b,200,0x4C,171,0x4d,173,0x54,176,0x5a, 13
;               ,
.db     0x5b, 44,0x66,  8
;
.db     0x69,'1',0x6b,'4',0x6c,'7',0x70,'0',0x72,'2',0x73,'5',0x74,'6'
;
.db     0x75,'8',0x79,'+',0x7a,'3',0x7b,'-',0x7c,'*',0x7d,'9'
;               F3      F2      F1
.db     0x03,  5,0x04,  4,0x05,  6,0x00,0

;.....................................................
```

```
;=============================================================================
; TABLE : thai/english Keyboard scan code lookup table
;=====================================================

T_EngUnshiftChar:
=================
.db     0x0d, 0 ,0x0e,'|',0x15,'q',0x16,'1',0x1a,'z',0x1a,'z',0x1b,'s'
.db     0x1c,'a',0x1d,'w',0x1e,'2',0x21,'c',0x22,'x',0x23,'d',0x24,'e'
.db     0x25,'4',0x26,'3',0x29,' ',0x2a,'v',0x2b,'f',0x2c,'t',0x2d,'r'
.db     0x2e,'5',0x31,'n',0x32,'b',0x33,'h',0x34,'g',0x35,'y',0x36,'6'
.db     0x3a,'m',0x3b,'j',0x3c,'u',0x3d,'7',0x3e,'8',0x41,','
.db     0x42,'k',0x43,'i',0x44,'o',0x45,'0',0x46,'9',0x49,'.',0x4a,'-'
.db     0x4b,'l',0x4c,'ø',0x4d,'p',0x4e,'+',0x52,'ๆ',0x54,'ๅ',0x55,'\'
.db     0x5a, 13,0x5b,'ฯ',0x5d,'\',0x61,'<',0x66, 8 ,0x69,'1',0x6b,'4'
.db     0x6c,'7',0x70,'0',0x71,',',0x72,'2',0x73,'5',0x74,'6',0x75,'8'
.db     0x79,'+',0x7a,'3',0x7b,'-',0x7c,'*',0x7d,'9'
.db     0x03,  5,0x04,  4,0x05,  6,0x00,0

T_EngShiftChar:
===============
.db     0x0d, 0 ,0x0e,'|',0x15,'Q',0x16,'1',0x1a,'Z',0x1a,'Z',0x1b,'S'
.db     0x1c,'A',0x1d,'W',0x1e,'2',0x21,'C',0x22,'X',0x23,'D',0x24,'E'
.db     0x25,'4',0x26,'3',0x29,' ',0x2a,'V',0x2b,'F',0x2c,'T',0x2d,'R'
.db     0x2e,'5',0x31,'N',0x32,'B',0x33,'H',0x34,'G',0x35,'Y',0x36,'6'
.db     0x3a,'M',0x3b,'J',0x3c,'U',0x3d,'7',0x3e,'8',0x41,','
.db     0x42,'K',0x43,'I',0x44,'O',0x45,'0',0x46,'9',0x49,'.',0x4a,'-'
.db     0x4b,'L',0x4c,'ø',0x4d,'P',0x4e,'+',0x52,'ๆ',0x54,'ๅ',0x55,'\'
.db     0x5a, 13,0x5b,'ฯ',0x5d,'\',0x61,'<',0x66, 8 ,0x69,'1',0x6b,'4'
.db     0x6c,'7',0x70,'0',0x71,',',0x72,'2',0x73,'5',0x74,'6',0x75,'8'
.db     0x79,'+',0x7a,'3',0x7b,'-',0x7c,'*',0x7d,'9'
.db     0x03,  5,0x04,  0,0x05,  6,0x00,0
```

```
T_ThaiUnshiftChar:
==================
;             า        ผ        ห        ฟ        ไ        แ        ป
.db     0x15,230,0x1a,188,0x1b,203,0x1c,191,0x1d,228,0x21,225,0x22,187
;             ก        ำ        ภ                 อ        ด        ะ
.db     0x23,161,0x24,211,0x25,192,0x29,' ',0x2a,205,0x2b,180,0x2c,208
;             พ        ถ        ั        ิ        ุ        เ        ้
.db     0x2d,190,0x2e,182,0x31,215,0x32,212,0x33,233,0x34,224,0x35,209
;             ,        ท        ็        ์        ด        ม
.db     0x36,216,0x3a,183,0x3b,232,0x3c,213,0x3d,214,0x3e,164,0x41,193
;             า        ร        น        จ        ต        ใ        ฝ
.db     0x42,210,0x43,195,0x44,185,0x45,168,0x46,181,0x49,227,0x4a,189
;             ส        ว        ย        ข        ง        บ        ช
.db     0x4b,202,0x4C,199,0x4d,194,0x4e,162,0x52,167,0x54,186,0x55,170
;                      ล
.db     0x5a, 13,0x5b,197,0x66,  8
;
.db     0x69,'1',0x6b,'4',0x6c,'7',0x70,'0',0x72,'2',0x73,'5',0x74,'6'
;
.db     0x75,'8',0x79,'+',0x7a,'3',0x7b,'-',0x7c,'*',0x7d,'9'
;
.db     0x03,  5,0x04,  4,0x05,  6,0x00,0
T_ThaishiftChar:
================
;             (        ฆ        ฤ        "        ฉ        )
.db     0x1a, 40,0x1b,166,0x1c,196,0x1d,208,0x21,169,0x22, 41
;             ฏ        ฎ                 ฮ        ๊        ซ
.db     0x23,175,0x24,174,0x29,' ',0x2a,206,0x2b,226,0x2c,184
;             ฑ        ็        ๋        ฌ        ๆ
.db     0x2d,177,0x31,236,0x33,231,0x34,172,0x36,217
;             ?        ์        ๋        ฬ
.db     0x3a, 63,0x3b,235,0x3c,234,0x41,178
;             ษ        ฒ        ฯ        ฝ        ฦ
.db     0x42,201,0x43,195,0x44,207,0x49,204,0x4a,198
;             ศ        ซ        ญ        ฐ
.db     0x4b,200,0x4C,171,0x4d,173,0x54,176,0x5a, 13
;             ,
.db     0x5b, 44,0x66,  8
;
.db     0x69,'1',0x6b,'4',0x6c,'7',0x70,'0',0x72,'2',0x73,'5',0x74,'6'
;
.db     0x75,'8',0x79,'+',0x7a,'3',0x7b,'-',0x7c,'*',0x7d,'9'
;             F3       F2       F1
.db     0x03,  5,0x04,  4,0x05,  6,0x00,0
```