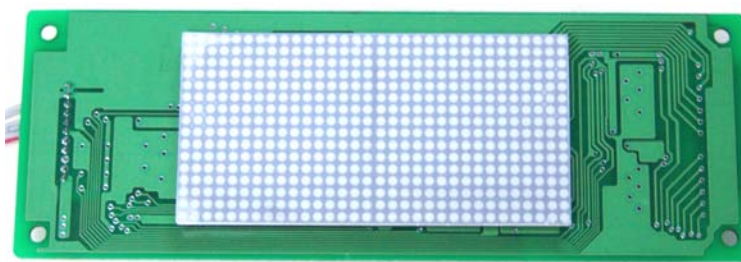


คำอธิบายโปรแกรม “8BY_ROT.EASM” และ “FIXMOTION.ASM”

เนื้อหาในไฟล์เอกสารนี้เป็นการอธิบายการทำงานของโปรแกรมแสดงผลตัวอักษรภาษาไทย และ ภาษาอังกฤษ บน Dot-Matrix LED Display ขนาด 16x32 รุ่น AD-501-B ดังแสดงในรูปด้านล่าง โดยโปรแกรมที่พัฒนา ร่วมกับโมดูล AD-501-B จะอ้างอิงกับบอร์ด MCS-51 รุ่น CP-JR51 USB v1.0 แต่ผู้อ่านสามารถนำไฟล์ .HEX ไปรันกับ CPU รุ่น AT89C51AC2 หรือ CPU ตระกูล AT89SXX ได้โดยทันที เนื่องจากโปรแกรมที่เขียนนั้นใช้ทรัพยากรพื้นฐานของ AT89C51 แต่ CPU ตระกูล AT89SXX ซึ่งไม่มีโหมด x2 ก็จะทำให้การแสดงผลนั้นไม่ราบเรียบ



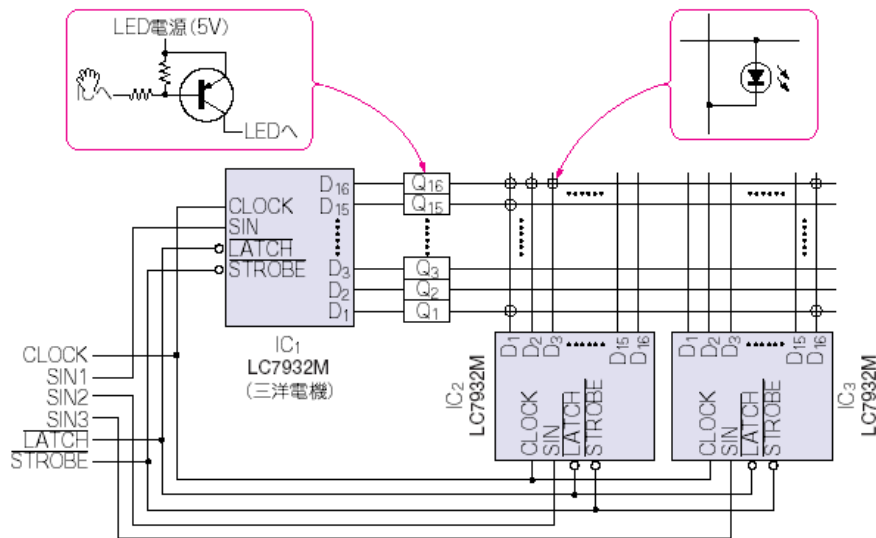
รูปแสดงแผงแสดงผล Dot-Matrix LED Display รุ่น AD-501-B

โปรแกรม **8BY_ROT.EASM** และ **FIXMOTION.ASM** คือ โปรแกรมแสดงตัวอักษรวิ่งจากขวาไปซ้าย และ โปรแกรมแสดงตัวอักษรแบบไม่เคลื่อนที่ โดยโปรแกรมทั้ง 2 นี้จะมีหลักการการทำงานที่คล้ายกันซึ่งจะอธิบายในหัวข้อต่อไป แต่ก่อนอื่นผู้อ่านควรทราบโครงสร้างภายในโมดูลแสดงผล รุ่น AD-501-B ก่อน

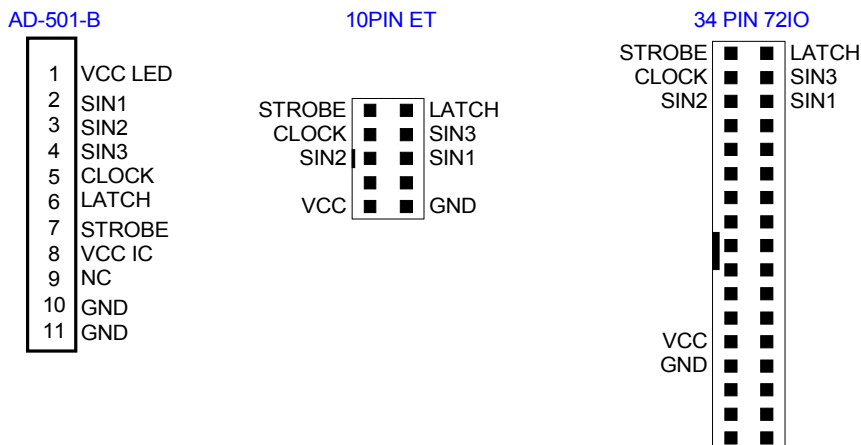
โครงสร้างภายในโมดูลแสดงผลรุ่น AD-501-B

โครงสร้างภายในของโมดูลแสดงผล รุ่น AD-501-B จะประกอบไปด้วยไอซีสแกนทางแถว (Row) จำนวน 1 ตัว และ ไอซีสแกนทางหลัก (Column) 2 ตัว ซึ่งไอซีทั้ง 3 ตัวนี้เบอร์ LC7932M ซึ่งเป็นไอซี 16-bit bi-directional shift register ซึ่งสามารถขับ LED ที่ต่อแบบ Dot matrix หรือ Dot array ได้โดยตรง นอกจากนั้นตัวมันยังมีคุณสมบัติเด่นๆ อีก คือ

- ตัวมันเป็น Silicon gate C-MOS สามารถทำงานที่ High-speed ได้
- ภาคเอาต์พุตของ LC7932M จะมีทรานซิสเตอร์แบบ N-channel ต่อวงจรแบบ Open drain output (สัญญาณจะกลับสถานะทางเอาต์พุต)
- การเลื่อนข้อมูลในแต่ละบิตจะสนใจสัญญาณ Clock ขอบขาขึ้น (Positive transition)
- แรงดันในช่วงใช้งานในระดับลอจิก : $V_{DD} = 4.5v$ ถึง $5.5 v$
- ความถี่ของสัญญาณ Clock ที่สามารถทำงานได้ : $f_{CLK} = DC$ ถึง $5 MHz (Max)$
- ฯลฯ



รูปแสดงวงจรภายในของโมดูลรุ่น AD-501-B

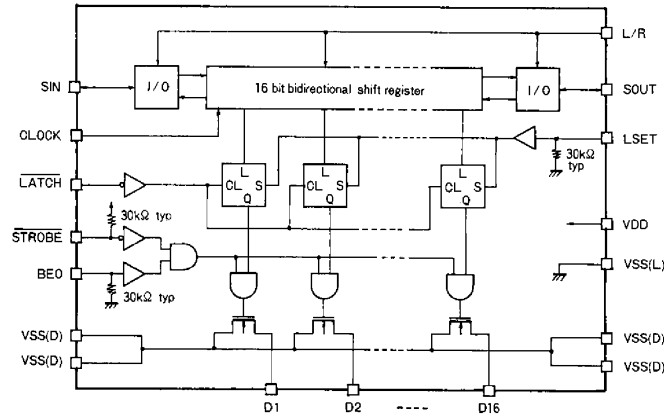


รูปแสดงพอร์ตเชื่อมต่อระหว่างโมดูล AD-501-B กับ บอร์ดไมโครคอนโทรลเลอร์

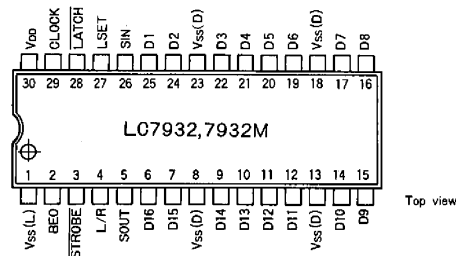
จากรูปทางด้านบนเป็นการแสดงวงจรภายใน และ ตำแหน่งขาใช้งานต่างๆ บนตัวโมดูล AD-501-B โดยภายในตัวโมดูลได้แยกแหล่งจ่ายไฟ (5 VDC) ออกเป็น 2 ชุด คือ ไฟเลี้ยงไอซี LC7932 และ ไฟเลี้ยง LED แสดงผล ซึ่ง 2 ขานี้ (ขา VCC LED และ VCC IC) สามารถจัมมรวมกันได้

ในโมดูลจะมีขาสัญญาณข้อมูล 3 ขา ได้แก่ SIN1, SIN2 และ SIN3 โดยขา SIN1 จะเป็นขาข้อมูลของ IC1 , ขา SIN2 จะเป็นขาข้อมูลของ IC2 และ ขา SIN3 จะเป็นขาข้อมูลของ IC3 ซึ่งโครงสร้างภายในของไอซี LC7932 แสดงดังภาพด้านล่าง คือ

Equivalent Circuit



Pin Assignment



รูปแสดง Equivalent Circuit ของไอซี LC7932

- STROBE เป็นขาที่ใช้ในการ เปิด/ปิด การแสดงผลของ Dot-Matrix LED Display, ลอจิก '0' เปิดการแสดงผล
- LATCH เป็นขาที่ใช้ในการควบคุม การส่งถ่ายข้อมูลไปยังเอาต์พุต โดยถ้าให้เป็นลอจิก '0' ผู้อ่านจะสามารถ เปลี่ยนแปลงข้อมูลทางเอาต์พุตได้จากข้อมูลอินพุตที่เข้ามาทาง SIN แต่ถ้าให้เป็น '1' ข้อมูลทางเอาต์พุตจะไม่มี การเปลี่ยนแปลงค่าตามอินพุต
- CLOCK เป็นขาที่ใช้ในการควบคุมจังหวะการถ่ายเทข้อมูลของ SIN1,SIN2,SIN3
- SIN1 เป็นขาที่ใช้ในการรับข้อมูล serial 16 บิต สำหรับ IC1
- SIN2 เป็นขาที่ใช้ในการรับข้อมูล serial 16 บิต สำหรับ IC2
- SIN3 เป็นขาที่ใช้ในการรับข้อมูล serial 16 บิต สำหรับ IC3
- VCC_LED เป็นขาแรงดันไฟเลี้ยง Dot-Matrix LED Display
- VCC_IC เป็นขาแรงดันไฟเลี้ยง IC1, IC2, IC3

หลักการการทำงานของโปรแกรม FIXMOTION.ASM

โปรแกรม FIXMOTION.ASM คือ โปรแกรมแสดงตัวอักษรภาษาไทย/อังกฤษ บนโมดูล AD-501-B ซึ่งจะแสดงผลในรูปแบบไม่มีการเคลื่อนที่ของตัวอักษร โดยสามารถใส่ตัวอักษรได้ 4 หลัก โดยไม่นับรวม พยัญชนะ ซึ่งพื้นที่การใส่ตัวอักษรในโปรแกรมจะอยู่ตอนท้ายของโปรแกรม ดังตัวอย่างด้านล่าง คือ

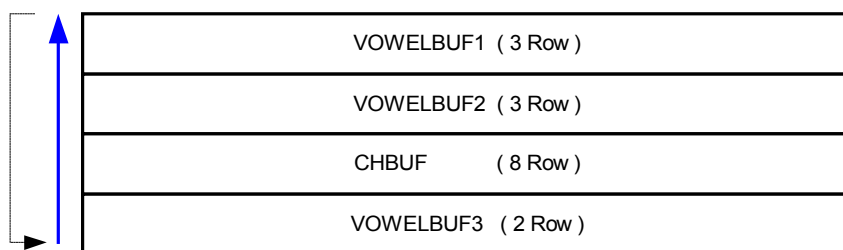
```

;=====
; pattum for demo
;=====
TABLE_1:      DFB      "เชื่อม"
               DFB      etx
    
```

จากตัวอย่างทางด้านบนจะเห็นว่า “ ชี ” ถือเป็น 1 หลัก ดังนั้น ข้อความนี้ใช้พื้นที่การแสดงผลทั้ง 4 หลัก ส่วน “etx” จะเป็นตัวบ่งบอกว่าจบข้อความแล้ว นอกจากนั้นข้อความที่ใส่สามารถผสมระหว่างภาษาไทย กับ ภาษาอังกฤษ ได้

✚ หลักการจัดหน่วยความจำในการแสดงผล

การจัดสรรหน่วยความจำในการแสดงตัวอักษรบนโมดูล AD-501-B จะแบ่งออกเป็น 4 ตัว เพื่อใช้เก็บข้อมูลตัวอักษรระดับต่างๆ เนื่องจากว่าการแสดงผลภาษาไทยจะแบ่งพื้นที่การแสดงผลออกเป็น 4 ระดับ ดังแสดงในรูปด้านล่าง

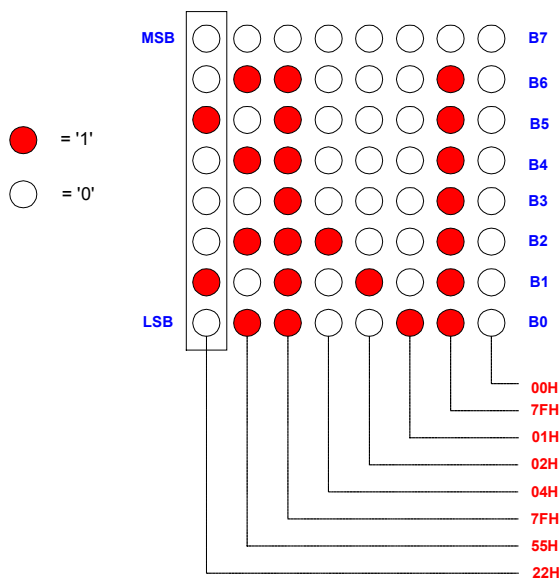


รูปแสดงหน่วยความจำที่ใช้เก็บตัวสระ และ พยัญชนะ บนหน้าจอแสดงผล

จากรูปทางด้านบนจะเห็นว่าการแบ่งพื้นที่หน้าจอแสดงผลออกเป็น 4 ส่วน คือ หนึ่งส่วนเก็บ “พยัญชนะ” และ อีกสามส่วนของบัพเฟอร์จะใช้เก็บ “สระ” โดยในการสแกนจะเป็นการสแกนทางแนวนอน (Row) จากล่างขึ้นบน

หลักการสร้างตัวอักษร

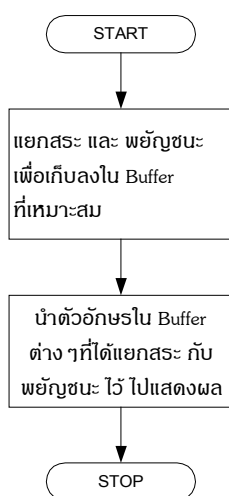
การสร้างตัวอักษร 1 ตัว เพื่อแสดงบนจอแสดงผลนั้นจะใช้ข้อมูลทั้งหมด 8 ไบต์ โดยข้อมูลแต่ละไบต์จะว่างเรียงกันในแนวตั้ง ดังแสดง ในรูปด้านล่าง



รูปแสดงตัวอย่างการสร้างตัวอักษร “ม” บนโมดูลแสดงผล

Flowchart การทำงานของโปรแกรม FIXMOTION.ASM

จาก Flowchart ที่จะแสดงในด้านข้างนี้จะแบ่งการทำงานโดยรวมออกเป็น 2 ส่วนใหญ่ๆ คือ 1.) ส่วนการแยกแยะตัวอักษรว่าเป็นสระ หรือ พยัญชนะ เพื่อที่จะเก็บตัวอักษรเหล่านี้ไว้ในตำแหน่งที่เหมาะสม 2.) ส่วนการนำตัวอักษรที่ได้เรียงลำดับ และ ตำแหน่งแล้วมาสแกนบนโมดูล AD-501-B



รูปแสดงการทำงานของโปรแกรมโดยรวมของโปรแกรม FIXMOTION.ASM

จากรูป Flowchart ทางด้านบน, ในส่วนของบล็อกที่ 1 กับ บล็อกที่ 2 นั้นจะใช้ตัวแปรชื่อเดียวกันบางตัว เพื่อเป็นการประหยัดทรัพยากร เนื่องจากว่าโปรแกรมจะต้องทำส่วนของ บล็อกที่ 1 จนเสร็จก่อน แล้วจึงทำส่วนของ บล็อกที่ 2 ได้ ซึ่งรายละเอียดอธิบายได้ดังนี้ คือ

- โครงสร้างการแยกแยะตัวอักษร

เนื่องจากตำแหน่งของตัวอักษรแต่ละตัว ทั้งภาษาไทย และ ภาษาอังกฤษ จะเป็นไปตามหลักรหัส สมอ. ดังแสดงใน รูปด้านล่าง ดังนั้น การแยกแยะตัวอักษรแต่ละตัวจะดูจากตารางรหัส สมอ. นี้

| L : H | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|-------|---|---|-------|---|---|---|---|---|
| 0 | | | Space | 0 | @ | P | ' | p |
| 1 | | | ! | 1 | A | Q | a | q |
| 2 | | | " | 2 | B | R | b | r |
| 3 | | | # | 3 | C | S | c | s |
| 4 | | | \$ | 4 | D | T | d | t |
| 5 | | | % | 5 | E | U | e | u |
| 6 | | | & | 6 | F | V | f | v |
| 7 | | | ' | 7 | G | W | g | w |
| 8 | | | (| 8 | H | X | h | x |
| 9 | | |) | 9 | I | Y | I | y |
| A | | | * | : | J | Z | j | z |
| B | | | + | ; | K | [| k | { |
| C | | | , | < | L | \ | l | |
| D | | | - | = | M |] | m | } |
| E | | | . | > | N | ^ | n | ~ |
| F | | | / | ? | O | _ | o | |

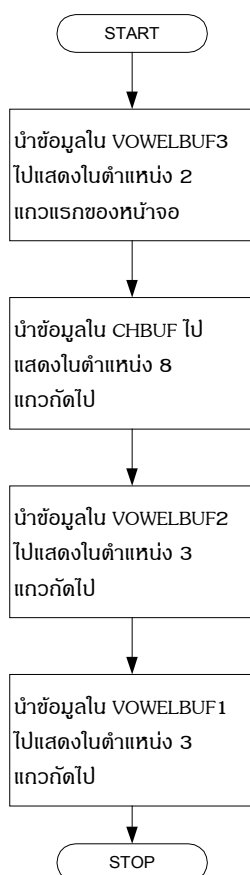
ตาราง รหัสตัวอักษร สมอ. (1/2)

| L : H | 8 | 9 | A | B | C | D | E | F |
|-------|---|---|---|---|---|---|---|---|
| 0 | | | | ฐ | ภ | ๕ | ! | ๐ |
| 1 | | | ก | ท | ม | ๕ | แ | ๑ |
| 2 | | | ข | ฒ | ย | า | โ | ๒ |
| 3 | | | ฃ | ณ | ร | ำ | ใ | ๓ |
| 4 | | | ค | ด | ฤ | ๖ | ๗ | ๔ |
| 5 | | | ค | ต | ถ | ๗ | | ๕ |
| 6 | | | ฅ | ถ | ภ | ๘ | ๙ | ๖ |
| 7 | | | ง | ท | ว | ๘ | ๙ | ๗ |
| 8 | | | จ | ธ | ศ | ๙ | ' | ๘ |
| 9 | | | ฉ | น | ษ | ๙ | ๙ | ๙ |
| A | | | ช | บ | ส | | ๙ | |
| B | | | ช | ป | ห | | + | |
| C | | | ฌ | ผ | พ | | ' | |
| D | | | ญ | ฝ | อ | | | |
| E | | | ฎ | พ | ฮ | | | |
| F | | | ฎ | ฟ | ๙ | ฿ | | |

ตาราง รหัสตัวอักษร สมอ. (2/2)

- การนำตัวอักษรใน Buffer ไปสแกนบนหน้าจอแสดงผล

เมื่อมาถึงขั้นตอนนี้, ในบัฟเฟอร์ CHBUF, VOWELBUF1, VOWELBUF2 และ VOWELBUF3 จะมีค่าข้อมูลตัวอักษรที่ถูกแบ่งกลุ่มและจัดตำแหน่งตัวอักษรเรียบร้อยแล้ว ดังนั้น ขั้นตอนการทำงานในส่วนนี้จะเป็นการเอาค่า Ascii เหล่านี้ไปทำการคำนวณเพื่อแสดงตัวอักษรออกทางจอแสดงผล ซึ่ง Flowchart แสดงดังภาพด้านล่าง คือ

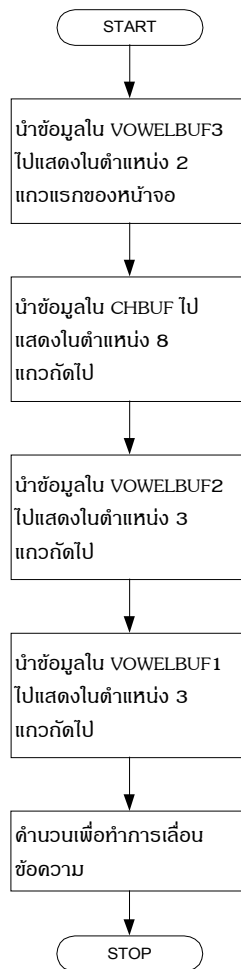


รูปแสดง Flowchart ลำดับการสแกนบนหน้าจอแสดงผล

จาก Flowchart ด้านบน, จะเห็นว่าการสแกนจะสแกนจากล่างขึ้นบน ซึ่งเมื่อทำการสแกนเสร็จทั้งหน้าจอแล้วจะวนกลับไปสแกนตั้งแต่ต้นใหม่

การทำงานของโปรแกรม 8BY_ROTATE.ASM

โปรแกรม 8BY_ROTATE.ASM คือ โปรแกรมแสดงข้อความแบบเลื่อนไปทางซ้ายซึ่งสามารถใส่ตัวอักษรใน 1 ข้อความได้ 28 ตัวอักษร ในส่วนของการทำงานจะมีลักษณะคล้ายคลึงกับโปรแกรม FIXMOTION.ASM เพียงแต่ในการสแกนแต่ละรอบจะมีการควบคุมการเพิ่มขึ้นของ R6 และ POINTER ในแต่ละรอบของการสแกนข้อความซึ่งแสดงดัง Flowchart ด้านล่าง



รูปแสดง Flowchart ลำดับการสแกนจอภาพแบบเลื่อนข้อความ

ในส่วนของการเปลี่ยนแปลงข้อความนั้นผู้อ่านสามารถเปลี่ยนแปลงได้จากตอนท้ายของโปรแกรมซึ่งจะอยู่ประมาณบรรทัดที่ 1154 ดังแสดงในรูปด้านล่าง นอกจากนี้ผู้อ่านสามารถเปลี่ยนแปลงความเร็วในการแสดงผลได้จากบรรทัดที่ 109 ในส่วนของการประกาศตัวแปร “ LOOP_R EQU 5 “

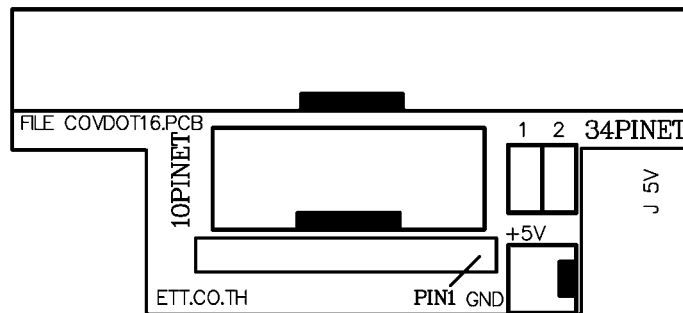
```

1150 ;=====
1151 ; patturn for demo
1152 ;=====
1153
1154 TABLE_1:
1155     DFB "การเชื่อมต่ออุปกรณ์แสดงผลแบบ LED"
1156     DFB etx
1157 ;=====
1158
    
```

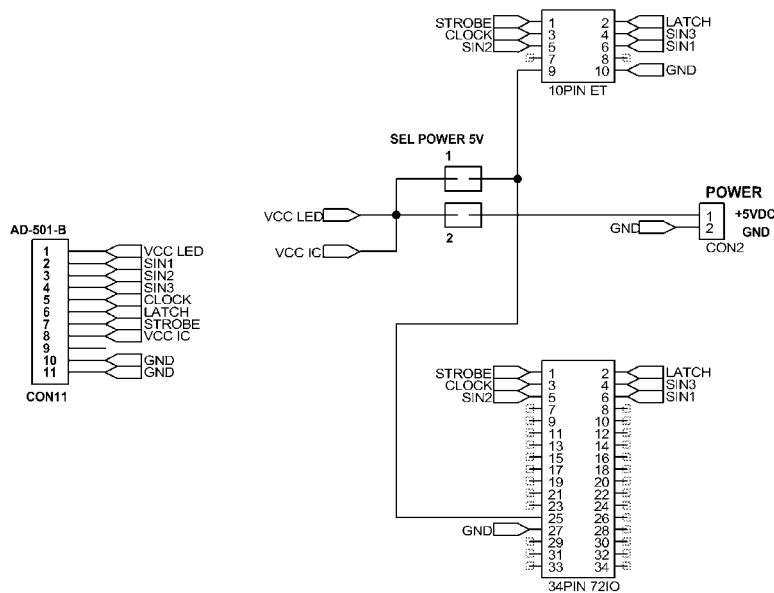
รูปแสดงตำแหน่งข้อความที่ใช้แสดงบนโมดูลแสดงผล

Note : ผู้อ่านควรปรับ Font ในโปรแกรมให้เป็น ชนิด MS Sans Serif ขนาด 8

FILE COVDOT16.PCB



รูปแสดงตำแหน่งพอร์ต 10 PIN และ 34 PIN บนโมดูล ซึ่งสามารถใช้ได้กับผลิตภัณฑ์ของ ETT



รูปแสดงวงจรของพอร์ตบนโมดูล

จากรูปวงจรทางด้านบน จะมีจัมเปอร์ 2 ตัว คือ จัมเปอร์หมายเลข 1 (ไฟเลี้ยงโมดูล AD-501-B มาจากบอร์ด Controller) และ จัมเปอร์หมายเลข 2 (ไฟเลี้ยงโมดูล AD-501-B มาจากขั้ว CON2) โดย ผู้อ่านสามารถเลือกแหล่งจ่ายไฟสำหรับโมดูล AD-501-B ได้ตามต้องการ