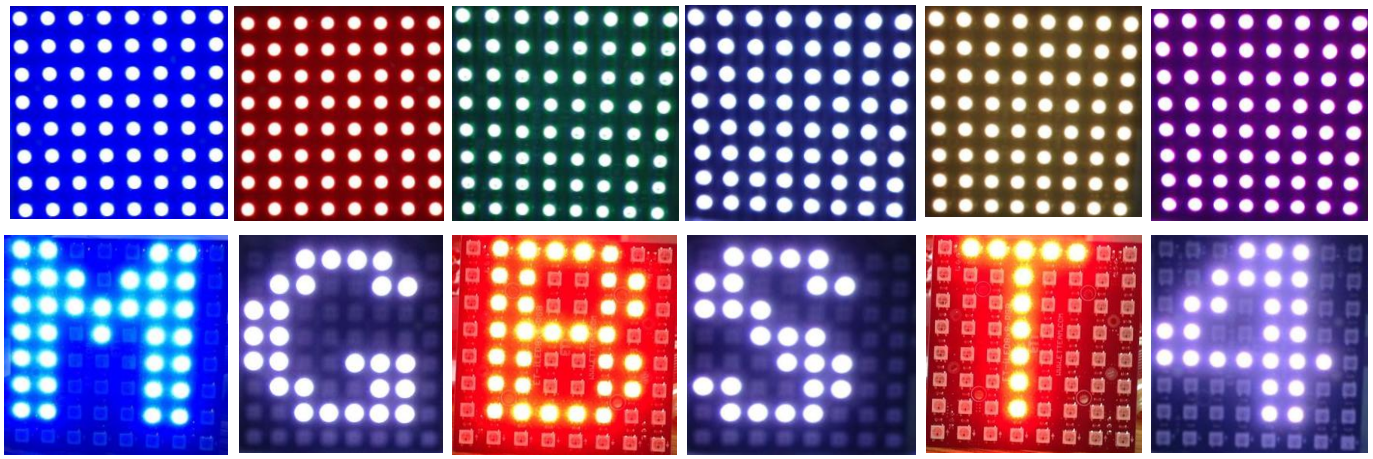
ET- iLED 8x8-RGB

ET-TFT-iLED8x8-RGB เป็น Module LED RGB ขนาด 8x8 Dot โดยเป็นการนำเอา LED เบอร์ WS2812B มาต่อ Cascade กัน 64 ดวง ซึ่ง LED แต่ละดวงจะให้ความละเอียดบิตสีได้สูงถึง 24 Bit Color RGB นั่นหมายความว่าสามารถแสดงสีได้ทั้งหมด 16 ล้านสี โดยเราสามารถนำ Module แต่ละ Moduleมาต่ออนุกรมรวมจำนวน LED ได้ถึง 1024 ดวง เป็นอย่างน้อยโดยไม่ต้องเพิ่มวงจรใดๆ การ Control จะใช้การ Interface ด้วยสายเพียงเส้นเดียวแบบ NZR คือ ส่ง Data ในแบบ Serial โดย Data 0 หรือ 1 นั้น จะถูกกำหนดด้วยคาบเวลาซึ่งมีลักษณะเหมือนกับสัญญาณพัลส์โดยจะใช้สัญญาณ 1 คาบเวลาต่อ 1 Bit Data เป็นตัวกำหนด Data 0 หรือ 1 ตัว Module สามารถรองรับ Input Data ในแบบ TTL 5V และ 3.3V จึงสามารถนำไปต่อใช้งานได้ทั้ง MCU 5 V หรือ 3.3 V แต่ไฟเลี้ยง Module ต้อง DC 5 V เท่านั้น

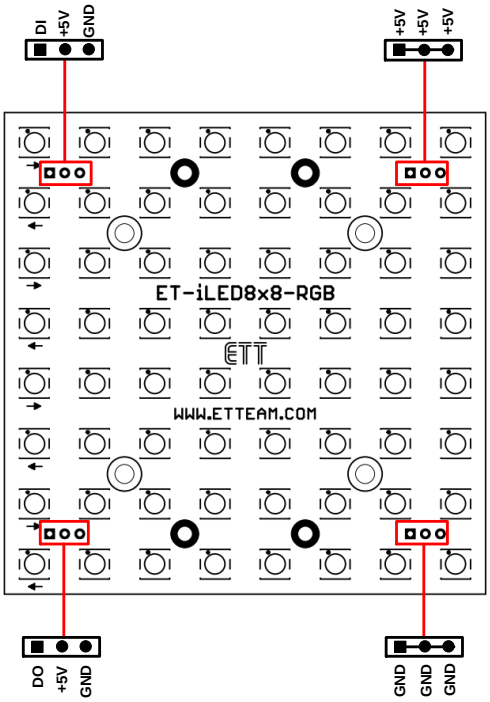


1. คุณสมบัติ Module ET-iLED8x8-RGB

-ไฟเลี้ยง Module DC 5 V

- Input DI รองรับสัญญาณ TTL 5V และ 3.3 V ดังนั้นสามารถใช้กับ MCU 5 Vและ 3.3V ได้
- ใช้การ Control Interface Module แบบ Serial NZR ด้วยสายสัญญาณ Control เพียงเส้นเดียว รวมทั้งการต่อ Module Cascade กัน ก็ใช้สายสัญญาณ Control เพียงเส้นเดียวเช่นกัน
- ภายใน LED เบอร์ WS2812B จะประกอบไปด้วย วงจรปรับรูปสัญญาณ,วงจร Drive ,วงจรควบคุม Pixel RGB,วงจร Electric Reset และวงจร Power Lose Reset
- LED 1 Module (64ดวง) ที่แสงสีขาว(Color=0xFFFFFF) จะกินกระแสไฟ ประมาณ 3.840 A หรือ 19.2 W
- ภายใน LED ดวงหนึ่งจะประกอบด้วยแม่สี 3 สี คือ RGB ซึ่งมีความละเอียดอยู่ที่ 24 bit Color โดยในแต่ละสีจะแสดงความสว่างได้ 256 เฉดดังนั้น LED 1ดวงสามารถแสดงสีได้ 16,777,216 สี โดยใช้ความถี่ในการสแกนไม่น้อยกว่า 400 Hz/s
- การต่อสัญญาณ Data Control ระหว่าง 2 Module สามารถต่อห่างกันได้มากกว่า 5 เมตร โดยไม่ต้องเพิ่มวงจรใดๆ
- เมื่อใช้ อัตราการ Refresh อยู่ที่ 30 fps จะสามารถนำ Module มาต่อ Cascade หรืออนุกรมรวมจำนวน LED ได้ไม่น้อยกว่า 1024 ดวง
- ความเร็วในการส่งข้อมูลแบบ Serial ในแต่ละบิต จะอยู่ที่ 800 Kbps
- LED นี้จะให้แสงที่มีความสม่ำเสมอสูง และ Output ของ LED แต่ละดวงจะเป็นแบบ Late คือจะติดค้างสถานะเดิมอยู่จนกว่าจะมีการส่ง Data มา Set สี ใหม่

2. โครงสร้างของ Module ET-iLED8x8-RGB



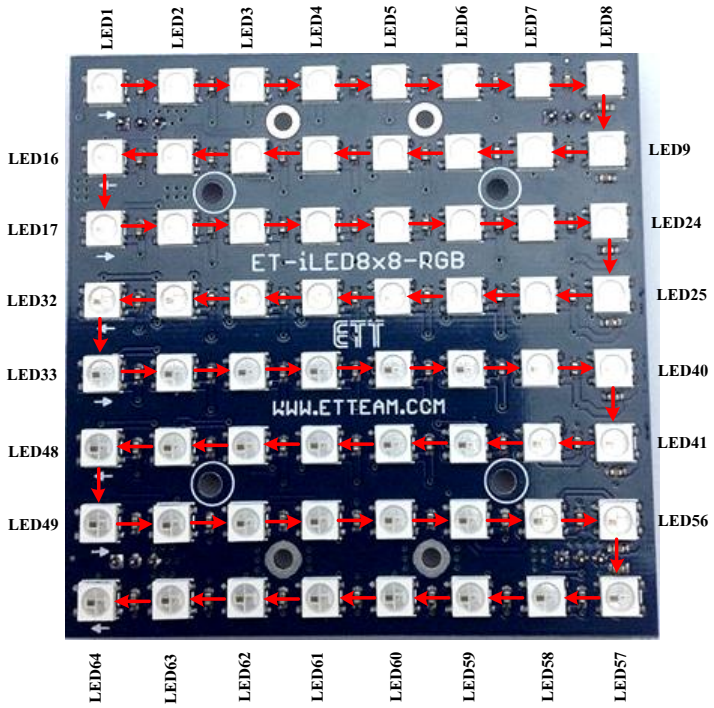
รูปแสดงตำแหน่งขาต่อใช้งานของ Module ET-iLED8x8-RGB

DI : เป็นขา Input รับสัญญาณ Serial Data Bit Color รองรับสัญญาณ TTL 5V และ 3.3V

DO : เป็นขา Output สำหรับส่ง Serial Data Bit Color ใช้ต่อไปยังขา DI ของ Module ตัวต่อไป

5V : เป็นขาไฟเลี้ยง Module 5V รวมทั้งใช้จ่ายไฟเลี้ยง 5V ให้กับ Module ตัวต่อไป สามารถเลือกต่อจุดใดก็ได้

GND : เป็นขา Ground ของ Module รวมทั้งใช้ต่อไปยังขา GND ของ Module ตัวต่อไป สามารถเลือกต่อจุดใดก็ได้



รูปแสดงลำดับการต่อ LED Cascade และแสดงตำแหน่งของ LED ภายใน Module ET-iLED8x8-RGB

สำหรับทิศทางลำดับการต่อ LED Cascade ใน Module รวมทั้งลำดับของ LED ที่แสดงดังรูปด้านบนจะมีความสำคัญมากในการเขียนโปรแกรมเพื่อกำหนดรูปแบบการแสดงผลบน LED เนื่องจาก Data ที่ส่งมา Frame แรก จะถูก Set ให้กับ LED1, Data Frameที่ 2 ก็จะวิ่ง

ผ่าน LED1(ไม่มีผลต่อ LED1) และถูก Set ให้กับ LED2 , Data Frame3 ก็จะวิ่งผ่าน LED1,LED2 และถูก Set ให้กับ LED3 ไล่ไปตามลำดับเช่นนี้จนครบ LED ลำดับที่ 64 ดังนั้นในการเขียนโปรแกรมผู้ใช้จะต้องคำนึงถึงรูปแบบการต่อไว้ด้วยเพื่อจะทำให้ผู้ใช้สามารถกำหนด LED ให้ติดตามตำแหน่งที่ผู้ใช้ต้องการได้

3. การทำงานของ Module ET-iLED8x8-RGB

เริ่มต้น ผู้ใช้จะต้องสร้างสัญญาณขึ้นมา 3 แบบ คือ สัญญาณ Logic 0 ,Logic 1 และ สัญญาณ Reset โดยสัญญาณทั้ง 3 รูปแบบนี้จะเกี่ยวข้องกับเวลาในการ On/Off ของสัญญาณ โดยสามารถดูรูปแบบของสัญญาณได้ในหัวข้อการ Control เมื่อสร้างสัญญาณ Logic ได้แล้ว ต่อไปให้เราส่ง Logic ตาม Data Bit สี ที่เราต้องการ ไปที่ขา DI ของ module ทีละบิตแบบ serial โดยจะต้องส่งทั้งหมด 24 bit สี ซึ่งการจัดเรียงบิตสีจะเรียงแบบ GRB โดยให้ส่งบิต MSB ออกไปก่อน เมื่อครบ 24 บิต หรือ 1 Frame ก็ให้ผู้ใช้ ส่ง data บิตสี ของ frame ที่2 ตามออกมาอีกซึ่งช่วงเวลาในการส่งระหว่างแต่ละ Frame ควรน้อยกว่า 50 us โดยให้ส่งทั้งหมด 64 Frame เนื่องจาก Module นี้มี LED ต่อ Cascade กันอยู่ 64 ดวง โดย Data บิตสีของ Frame แรกก็จะถูก Set ให้กับ LED1 , Data Frame2 จะถูก Set ให้กับ LED ดวงที่ 2 ตามลำดับเช่นนี้ไปเรื่อยๆจนผู้ใช้ส่ง Data ครบ 64 Frame ก็ให้ตามด้วยสัญญาณ Reset LED ทั้งหมด 64 ดวงก็จะติดตามสีที่ผู้ใช้ต้องการ และการส่ง data ครั้งต่อไปหลังจากส่งสัญญาณ Reset data สีที่ส่งมาก็จะกลับไปเริ่มต้น Set ให้กับ LED ดวงที่ 1 ใหม่เป็นต้น ดังนั้นการ Control LED ของ Module นี้จะประกอบไปด้วย สัญญาณ สี 24 bit จำนวน 64 Frame + สัญญาณ Reset

ในกรณีที่ต่อ Module LED Cascade กันตั้งแต่ 2 Module ขึ้นไป ผู้ใช้ก็จะต้องส่ง Data บิตสีหรือจำนวน Frame ตามจำนวน LED ที่ต่อ กล่าวคือจากการทำงานที่กล่าวข้างต้นในหนึ่ง Module (64LED)จะต้องส่ง Data bit สี 64 Frame ดังนั้นถ้าเราต่อ 2 Module Cascade กันก็จะต้องส่ง Data สีออกไปทั้งหมด 128 Frame ต่อเนื่องกันระยะห่างระหว่าง Frame จะต้องน้อยกว่า 50 us โดย Data สีแต่ละFrame จะเป็นสีเดียวกัน หรือต่างสีก็ได้ เมื่อส่ง Data ครบ 128 Frame แล้ว ให้จบด้วยสัญญาณ Reset LED ทั้ง 2 Module จำนวน 128 ดวงก็จะติดตามสีที่ส่ง โดยจำไว้ว่า สีที่ส่งไปเฟรมแรกจะเป็นสีของ LED ดวงแรก สีของเฟรมที่2 ก็จะเป็นสีของ LED ตัวที่2 ที่ต่อ Cascade อยู่ ไล่ไปตามลำดับเช่นนี้เรื่อยๆ ตามจำนวน LED ที่ต่อกันทั้งหมด ดังนั้นการ Control Module LED ที่นำมาต่อ Cascade กัน ผู้ใช้จะต้องส่งจำนวนเฟรมสีเท่ากับจำนวน LED ที่นำมาต่อทั้งหมด+สัญญาณ Reset

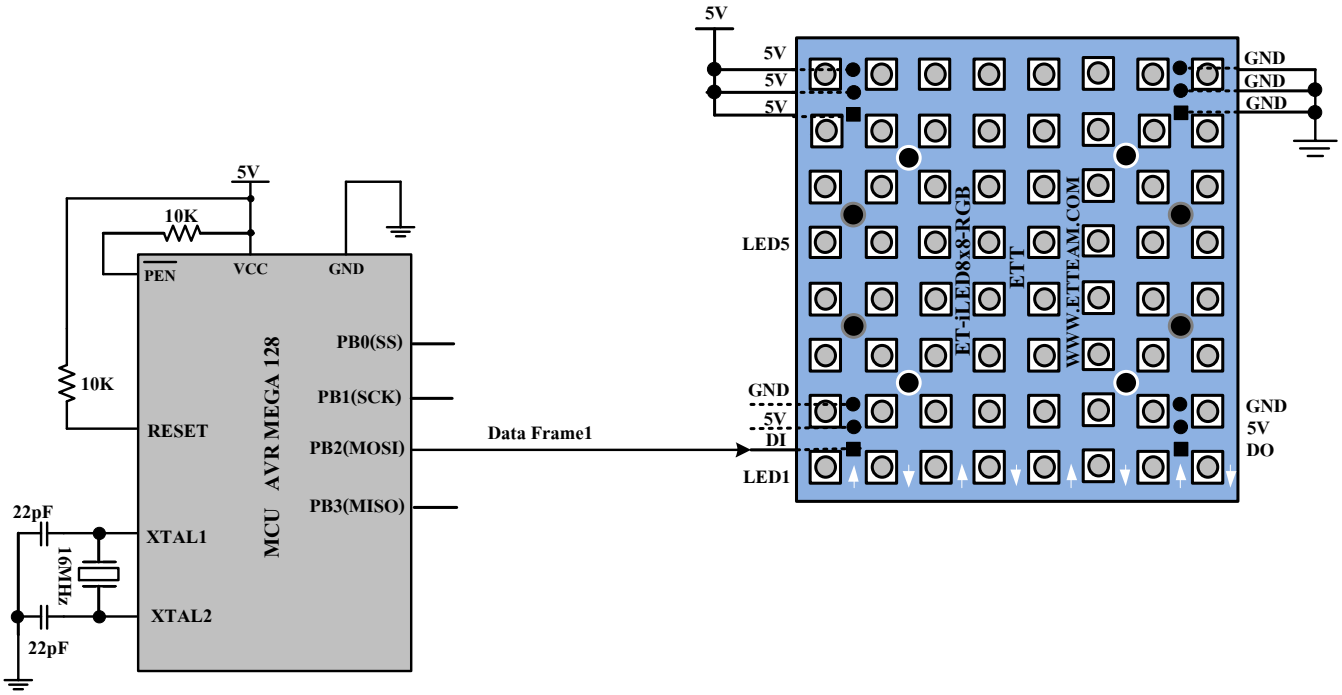
การทำงานในส่วนของการต่อ Cascade กันนี้ เมื่อ LED ดวงแรกได้รับบิตสีครบ 24 บิต(frame1) แล้ว มันจะจำสถานะ การรับรู้ไว้ จากนั้นเมื่อบิตสีชุดที่2ส่งมา(frame2) LED ดวงแรกจะไม่สนใจข้อมูลส่วนนี้จะปล่อยผ่านข้อมูลไปยังขา DO ส่งผ่านข้อมูลนี้ไปยังขา DI ของ LED ดวงที่ 2 LED ดวงที่2 ก็จะรับ Data บิตสีของเฟรมที่ 2 นี้ไว้ และจำสถานะ การรับรู้ไว้ เมื่อบิตสีชุดที่ 3ส่งมา(frame3) LED ดวงที่ 1 และ 2 ก็จะไม่สนใจ Data ส่วนนี้ เนื่องจากบิตรับรู้ภายในของ LED ถูก Set ไว้แล้ว Data เฟรมที่3 ก็จะถูก ส่งไปให้กับ LED ดวงที่3 ตราบใดที่ยังไม่มีการส่งสัญญาณ Reset Data บิตสีในแต่ละเฟรมก็จะถูกส่งไปยัง LED แต่ละดวงตามลำดับของการต่อ Cascade กัน หลังจากส่งเฟรม Data ครบตามจำนวน LED ที่ต่อใช้งานแล้ว ให้ผู้ใช้ส่งสัญญาณ Reset ปิดท้าย LED ก็จะติดตามสีที่ผู้ใช้กำหนด และสถานะของบิตรับรู้ของ LED ทั้งหมดจะถูก Clear นั่นหมายความว่า หลังส่งสัญญาณ Reset ถ้าผู้ใช้ส่งเฟรมสีเข้ามาอีก Data ของเฟรมสีก็จะเริ่มต้นถูกรับโดย LED ดวงที่ 1 ใหม่เป็นต้น ดังนั้นผู้ใช้จะต้องส่งเฟรม Data สี ให้เท่ากับจำนวน LED ที่ต่อใช้งาน แล้วจึงจบด้วยสัญญาณ Reset การส่ง Data ออกไปแต่ละเฟรมนั้นจะต้องใช้เวลาให้น้อยกว่า 50 us มิฉะนั้น LED จะรับรู้เป็นสัญญาณ Reset ทำให้ เฟรม Data สีนั้นถูกเริ่มรับโดย LED ดวงแรกใหม่ ทำให้ LED ได้รับ Data ไม่ครบทุกดวง การแสดงผลของ LED ก็จะผิดพลาดได้

<u>หมายเหตุ</u> เพื่อลดความร้อนจากการทำงานของ LED ให้กับ PCB ไม่ควรให้แสงสีขาวติดค้างนานเกินไป

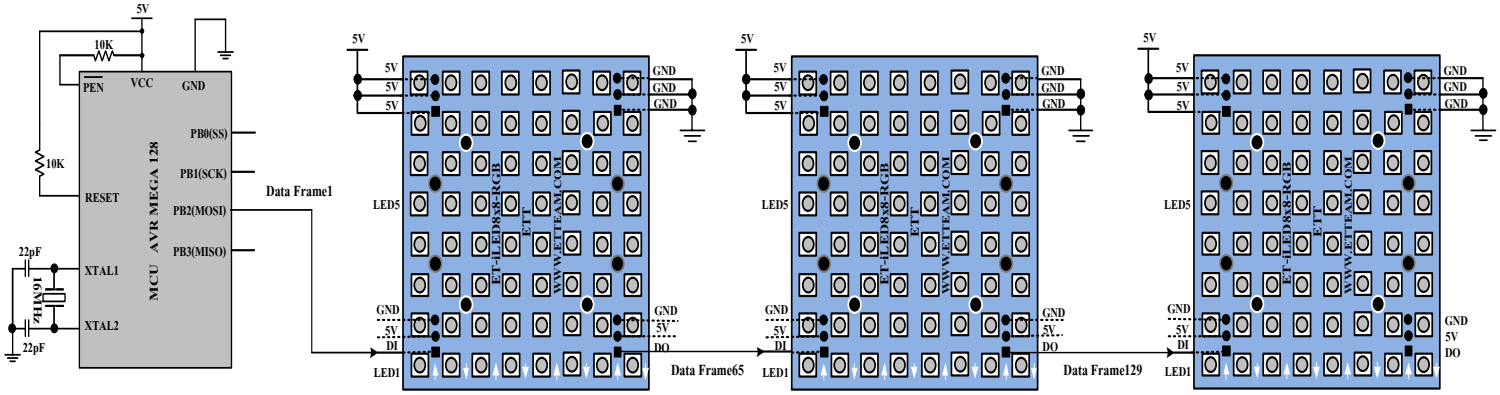
4. การ ต่อใช้งานกับ MCU และการ Control ET-iLED8x8-RGB

4.1) การต่อใช้งาน สำหรับวงจรการต่อนี้จะใช้ MCU AVR Mega 128 Clock 16 MHz เป็นตัว Control โดยใช้บอร์ด ET-Base AVR ATmega128 r3 ในการทดลองโดยจะอ้างอิงกับตัวอย่างของอีทีที การ Control จะใช้ทรัพยากร SPI ของ AVR ในการ Control โดยใช้ Pin PB2(MOSI) เป็นตัวสร้างสัญญาณ Logic 0 , 1 และ Reset สำหรับส่งไปยัง Module LED

ในการวาง Module iLED8x8-RGB เพื่อให้สอดคล้องกับตัวอย่างของอีทีทีควรวางตามรูปการต่อใช้งานดังรูปด้านล่างด้วยเพื่อจะได้แสดงผลในตัวอย่างที่เกี่ยวกับ Font ได้ถูกต้อง



รูปแสดงการต่อ MCU AVR Mega128 Control Module ET-iLED8x8-RGB แบบ Module เดียว



รูปแสดงการต่อ MCU AVR Mega128 Control Module ET-iLED8x8-RGB แบบ Cascade

หมายเหตุ ในการต่อ Module LED Cascade กันหลายๆ Module ให้คำนึงถึงแหล่งจ่ายไฟเลี้ยง Module ด้วยว่าจ่ายกระแสพอหรือไม่ซึ่ง บอร์ด LED 1 Module จะกินกระแสประมาณ 3.840 A (64 LED)

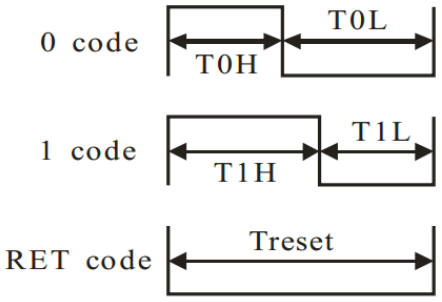
4.2) การ Control สำหรับการ Control ET-iLED8x8-RGB ที่จะกล่าวถึงนี้ จะอ้างอิงกับตัวอย่างของ อีทีที ซึ่งหัวใจสำคัญในการ Control นั้น จะอยู่ที่วิธีการสร้างสัญญาณ Logic 0,1 และ Reset ซึ่งจะขึ้นอยู่กับความสามารถของผู้ใช้และทรัพยากรของ MCU ว่าสามารถสร้างสัญญาณได้ตามที่ Module ต้องการหรือไม่ ซึ่งในตัวอย่างของอีทีทีจะใช้การส่ง Data ของ SPI เป็นตัวสร้างสัญญาณ โดยมีกระบวนการดังนี้

1) เขียนโปรแกรมสร้างสัญญาณ Logic 0 , 1 และ Reset ขึ้นมาก่อน ซึ่งลักษณะของสัญญาณจะต้องเป็นไปตาม Sequence chart ดังนี้

Data transfer time(TH+TL=1.25μs±150ns)

T0H	0 code ,high voltage time	0.35us	±150ns
T1H	1 code ,high voltage time	0.9us	±150ns
T0L	0 code , low voltage time	0.9us	±150ns
T1L	1 code ,low voltage time	0.35us	±150ns
RES	low voltage time	Above 50μs	

Sequence chart:



จากรูป Logic 0 : ผู้ใช้จะต้องสร้างสัญญาณออกมา 1คาบเวลาโดยมีคาบเวลาอยู่ที่ 1.25 us ± 150 ns โดยมีช่วงบวก (T0H) = 350 ns ±150 ns และมีช่วงลบ(T0L) = 900 ns ± 150ns สัญญาณนี้จะใช้ส่งแทน Logic 0 หนึ่งบิต Data

Logic 1 : ผู้ใช้จะต้องสร้างสัญญาณออกมา 1คาบเวลาโดยมีคาบเวลาอยู่ที่ 1.25 us ± 150 ns โดยมีช่วงบวก (T1H) = 900 ns ±150 ns และมีช่วงลบ(T1L) = 350 ns ± 150ns สัญญาณนี้จะใช้ส่งแทน Logic 1 หนึ่งบิต Data

Reset : ผู้ใช้จะต้องสร้างสัญญาณออกมา 1คาบเวลาโดยมีคาบเวลาอยู่ที่ 1.25 us ± 150 ns โดยมีช่วงลบ มากกว่า 50 us ซึ่งสัญญาณนี้จะใช้ส่งแทนการ Reset

สำหรับในตัวอย่างของอีทีที จะสร้างสัญญาณ Logic เหล่านี้โดยใช้หลักการส่ง Data ของ SPI โดยมีหลักการดังนี้(อ้างอิงกับ AVR Mega128 ที่ Clock 16 MHz)

- เริ่มต้น คำนวณ SCK ของ SPI ในการส่ง Data ก่อน โดย เราจะหาร SCK ด้วย 2 ก็จะได้ 8 MHz นั่นแสดงว่า การส่ง Data ของ SPI 1 Bit data จะใช้เวลา 1/8 MHz = 125 ns ซึ่งการหารด้วย 2 จะทำให้การคำนวณคาบเวลาการสร้างสัญญาณ Logic 0,1 อยู่ในช่วงเวลาที่กำหนด ในการ Initial SPI นอกจากหาร SCK ด้วย 2 แล้ว ให้กำหนดขั้ว Clock (CPOL) และเฟส Clock (CPHA) = 1
- กำหนด Data SPI ที่จะใช้ส่งสำหรับสร้างสัญญาณ Logic 0,1ดังนี้ ในการส่ง SPI 1 ครั้ง จะเป็นการส่ง Data ออกไปยังขา MOSI 8 Bit เมื่อ Data 1Bit ใช้เวลาในการส่ง 125 ns โดยเราจะกำหนดให้ส่งบิต 7 ออกไปเป็นบิตแรก ดังนั้น Data SPI สำหรับสร้าง Logic 0 ก็คือ 0xC0 กล่าวคือ เราให้บิต7และ6 เป็น 1จำนวน2 บิต ก็จะได้ช่วงเวลาที่เป็นบวกเป็นเวลา 125ns x 2 = 250 ns และบิต 5-บิต0 เป็น 0 จำนวน 6 บิต เราก็จะได้ช่วงเวลาที่เป็นลบ เป็นเวลา 125 ns x 6 = 750 ns ซึ่งจะเห็นว่าช่วงเวลาทั้ง 2 ช่วงอยู่ในเกณฑ์ที่กำหนดของ Code 0 ดังนั้นการส่ง Data SPI 1ครั้งด้วยค่า 0xC0 ก็จะแทนการส่ง Data Logic0 1 Bit ไปยัง Module LED ในขณะเดียวกัน Data SPI สำหรับสร้าง Logic 1 ก็คือ 0xFC โดยใช้หลักการคิดตามที่กล่าวมาข้างต้น ดังนั้นการส่ง Data SPI 1ครั้งด้วยค่า 0xFC ก็จะแทนการส่ง Data Logic1 1 Bit ไปยัง Module LED เป็นต้น สำหรับ สัญญาณ Reset ผู้ใช้ก็เพียงส่ง Data SPI ค่า 0x00 ออกไปแล้ว delay ไว้ให้มากกว่า 50 us ก็จะได้สัญญาณ Reset ตามต้องการ

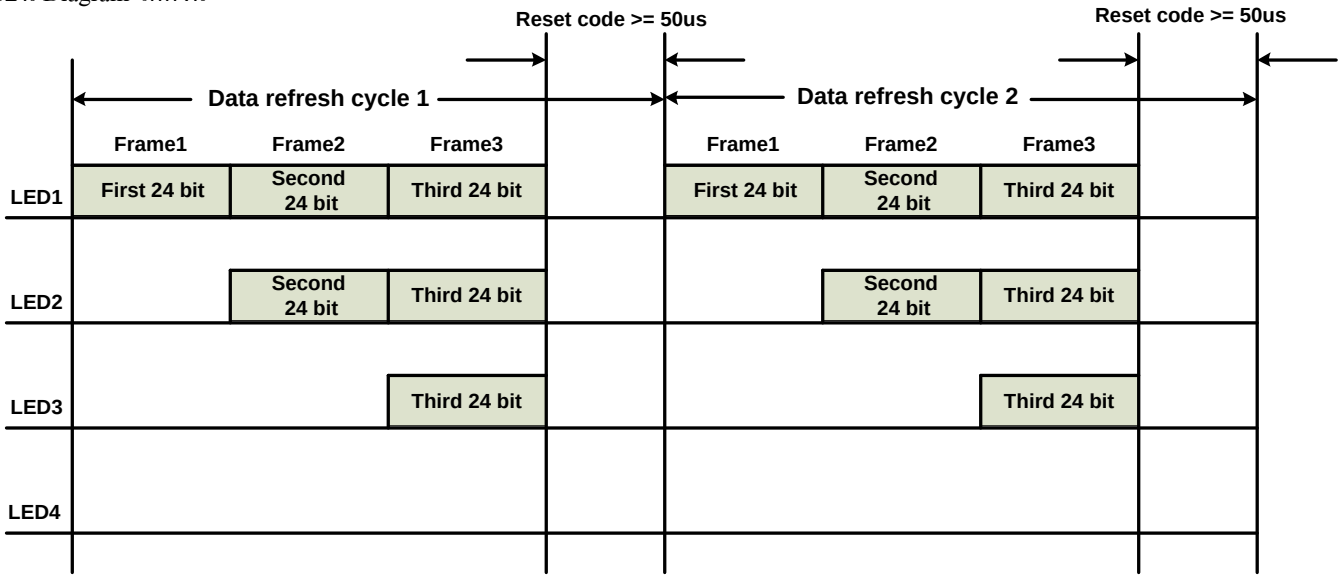
2) เขียนฟังก์ชันในส่วนของการส่ง Data Code สี 24 บิต ออกไปยังขา DI ของ Module LED โดย การส่ง Code 0 หรือ 1 ตาม Chart ด้านบนออกไป 1 คาบเวลา ก็จะหมายถึงการส่ง Data 0 หรือ 1 ออกไปให้ Module LED 1 บิต นั่นเอง ซึ่ง Code สี 24 บิต ที่จะส่งออกไปจะมีการจัดเรียงบิตแบบ GRB ซึ่งแสดงดังรูปด้านล่าง ในการส่งให้ส่ง Data บิต MSB ออกไปเป็นบิตแรก

Composition of 24bit data:

G7	G6	G5	G4	G3	G2	G1	G0	R7	R6	R5	R4	R3	R2	R1	R0	B7	B6	B5	B4	B3	B2	B1	B0

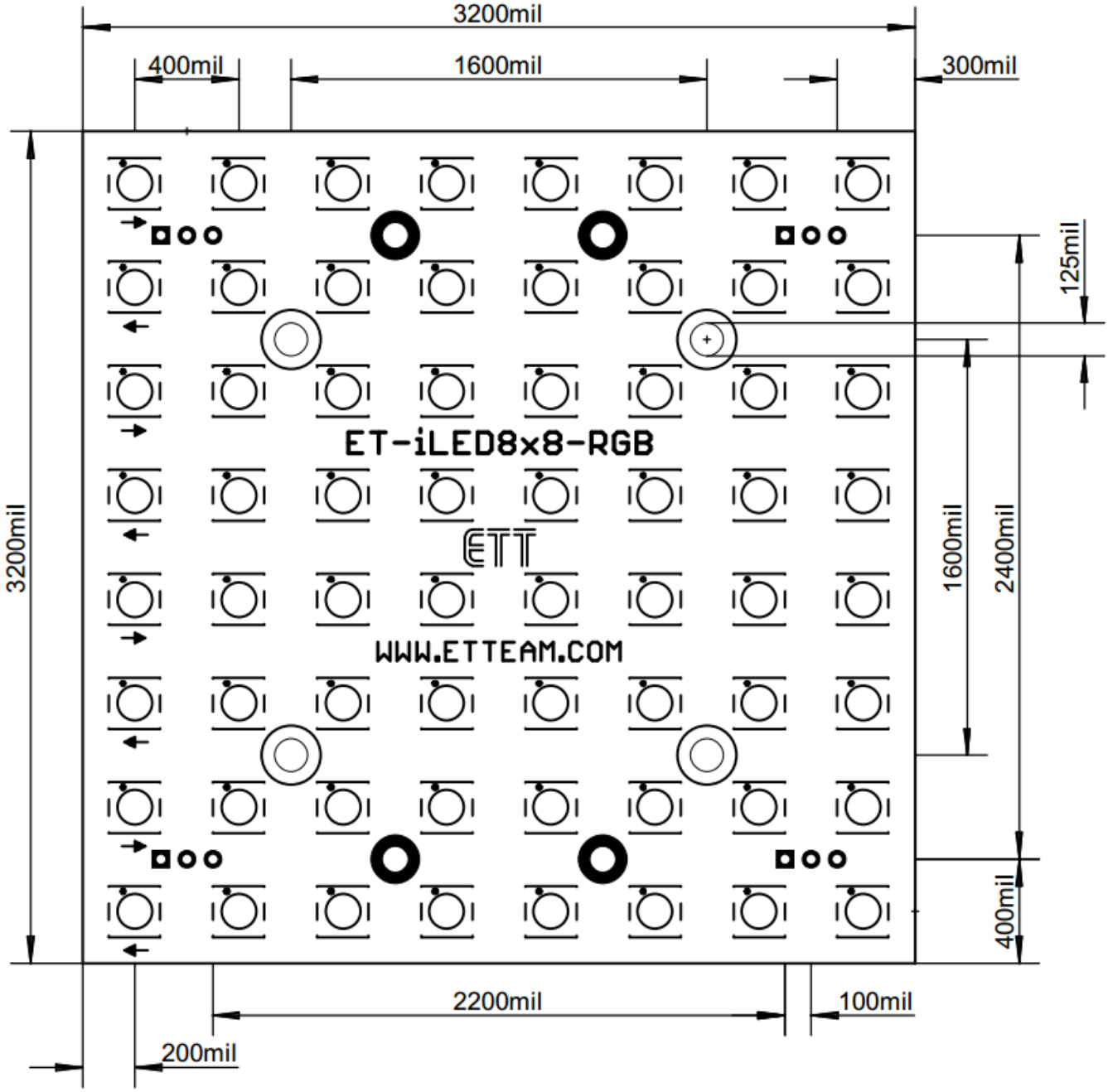
Note: Follow the order of GRB to sent data and the high bit sent at first.

3) เขียนฟังก์ชันสำหรับส่ง Frame Data โดยการส่ง Data 1 Frame ก็คือ การส่ง data Color 24 bit ในข้อ2นั่นเอง ซึ่งการส่ง Data 1 Frame นี้จะเป็นการกำหนดสีให้กับ LED 1 ดวง นั่นเอง ดังนั้น จำนวน Frame ที่ส่งแต่ละครั้งก็จะต้องเท่ากับจำนวน LED ที่ต่อ Cascade กันอยู่ทั้งหมดนั่นเอง โดย Frame ที่ส่งออกไป Frame แรก ก็จะเป็นการ Set สีให้กับ LED ดวงแรก Frame ที่ส่งออกไป Frame ที่2 ก็จะเป็นการ Set สีให้กับ LED ดวงที่2 ที่ต่อ cascade อยู่ ไล่ลำดับเช่นนี้ไปเรื่อยๆ เมื่อส่ง Frame ครบตามจำนวน LED ที่ต่อแล้วก็จะต้องปิดท้ายด้วยสัญญาณ Reset เสมอ เพื่อว่า Frame ที่ส่งต่อมา หลังสัญญาณ Reset จะได้เริ่มต้น Set ค่าสีให้กับ LED ดวงแรกใหม่ อีกครั้ง ซึ่งสามารถเขียนเป็น Diagram ได้ดังนี้

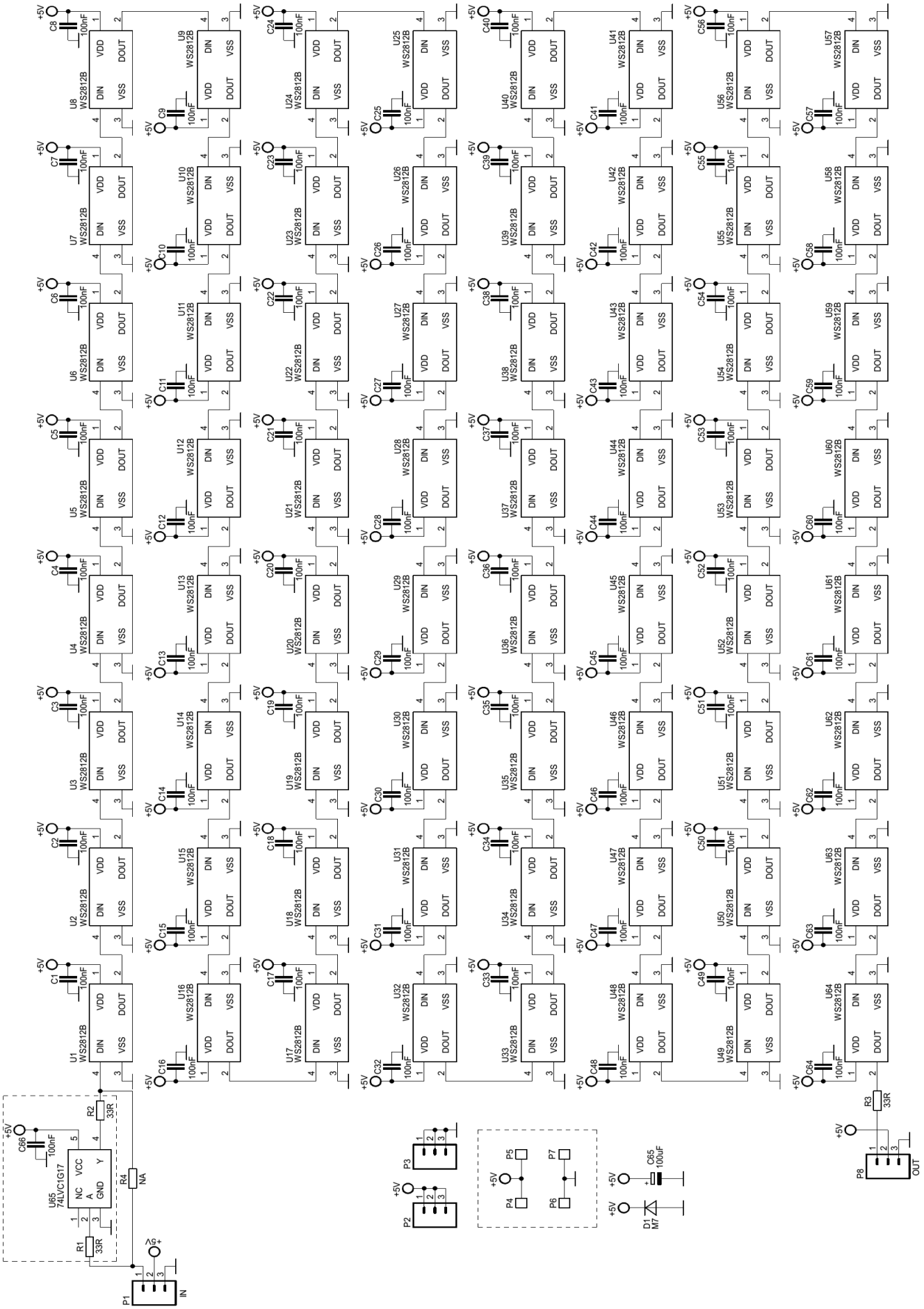


4) เมื่อผู้ใช้เขียนฟังก์ชันหลักๆตามที่กล่าวไปข้างต้นได้แล้วก็จะสามารถเขียนโปรแกรมสั่งการ Module ได้แล้ว สำหรับตัวอย่างของอีทีทีนั้นเราจะรับค่าสีจากผู้ใช้แบบ RGB เนื่องจากเป็นมาตรฐานการเรียงบิตสี แต่จะทำการสลับบิตสีเป็น GRB ก่อนส่งไปยัง Module LED

<u>หมายเหตุ</u> ในการเลือกใช้ MCU Control Module LED ผู้ใช้ควรเลือก MCU ที่มี Speed การทำงานที่สูงๆ ไว้ก่อน และพิจารณาว่า จะสร้างสัญญาณ Logic 0,1 และ Reset ด้วยวิธีการใด ซึ่งในตัวอย่างจะใช้การส่งข้อมูลของ SPI เป็นตัวสร้างสัญญาณ Logic ซึ่งบางครั้ง MCU ทำงานด้วย Speed สูงก็จริงแต่ไม่สามารถกำหนดความถี่ให้ได้ตามสัญญาณ Logic การทำงานที่กำหนดไว้ได้ และข้อควรระวังอีกอย่างหนึ่งในการเขียนโปรแกรมคือ MCU สามารถสร้าง logic การทำงานได้ตามต้องการและสามารถ Control LED 1 ดวงได้ไม่มีปัญหา แต่พอต่อ LED Cascade กันหลายๆดวงจะมีปัญหาซึ่งเกิดจากความเร็วในการส่งเฟรม Data สีออกไปในแต่ละเฟรมไม่ต่อเนื่องกัน กล่าวคือระยะห่างระหว่างเฟรมที่ส่งจะต้องน้อยกว่า 50 us ถ้ามากกว่าจะทำให้เหมือนเป็นการส่ง สัญญาณ Reset ออกไป จะทำให้ LED ดวงแรกกลับมารับเฟรมใหม่ทั้งๆที่ยังส่งเฟรมไม่ครบตามจำนวน LED ที่ต่ออยู่ อาจส่งผลให้ LED ที่ต่ออยู่ติดไม่ครบและ แสดงสีออกมาผิดพลาดได้



ขนาด Module ET-iLED8x8-RGB (1 mil=0.0254mm)



วงจร Module ET-iLED8x8-RGB