
	ใบเนื้อหา		หน้าที่ 1
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004		หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ		
ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire			
<div>หน่วยที่ 8 การรับส่งข้อมูลอนุกรมแบบต่าง ๆ</div> <div>การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire</div> <div>1. รูปแบบของการสื่อสารข้อมูล</div> <div>การสื่อสารข้อมูล คือการแลกเปลี่ยนข้อมูลระหว่างอุปกรณ์ตัวส่งและตัวรับผ่านสื่อตัวกลาง ซึ่งรูปแบบของการสื่อสารข้อมูลสามารถแบ่งออกได้ 3 ลักษณะคือ</div> <div>1.1 รูปแบบการสื่อสารข้อมูลที่แบ่งตามคุณสมบัติของสัญญาณ ซึ่งสามารถแบ่งออกได้เป็น 2 ประเภทคือ</div> <div>1.1.1 การสื่อสารข้อมูลแบบอนาล็อก (Analog Signal) เป็นการสื่อสารข้อมูลด้วยสัญญาณแบบต่อเนื่องมีลักษณะเป็นคลื่นไซน์ (Sine wave) โดยแต่ละคลื่นจะมีความถี่และความเข้มของสัญญาณที่ต่างกัน ทุก ๆ ค่าที่เปลี่ยนแปลงไปของระดับสัญญาณจะมีความหมายผิดพลาดได้ง่าย เนื่องจากค่าทุกค่าถูกนำมาใช้งาน เช่น สัญญาณเสียงในสายโทรศัพท์ เป็นต้น</div> <div>1.1.2 การสื่อสารข้อมูลแบบดิจิทัล (Digital Signal) เป็นการสื่อสารข้อมูลด้วยสัญญาณที่มีขนาดเปลี่ยนแปลงเป็นค่าของเลข 2 ระดับ โดยปกติมักแทนด้วยระดับแรงดันที่แสดงสถานะเป็น " 0 " และ " 1 " เป็นการส่งสัญญาณข้อมูลที่มีแต่เปิด/ปิด หรือเป็นเลขไบนารี (Binary) เมื่อระยะทางเพิ่มมากขึ้นจะทำให้สัญญาณดิจิทัลลดหายไปได้ จึงต้องใช้อุปกรณ์ทบทวนสัญญาณที่เรียกว่า รีฟิตเตอร์ เพื่อกู้ (Recover) รูปแบบของสัญญาณที่มีลักษณะเป็น 1 และ 0 เสียก่อน แล้วจึงส่งสัญญาณต่อไป</div> <div>1.2 รูปแบบการสื่อสารข้อมูลที่แบ่งตามทิศทางของการสื่อสารข้อมูล ซึ่งสามารถแบ่งออกได้เป็น 3 ประเภทคือ</div> <div>1.2.1 การสื่อสารข้อมูลแบบซิมเพล็กซ์ (Simplex) เป็นการติดต่อสื่อสารข้อมูลทางเดียว เมื่ออุปกรณ์ตัวหนึ่งทำหน้าที่ส่งข้อมูล อุปกรณ์อีกตัวหนึ่งจะต้องทำหน้าที่เป็นฝ่ายรับข้อมูลเสมอ ตัวอย่างเช่น อุปกรณ์เครื่องรับวิทยุ AM/FM เครื่องรับโทรทัศน์ เครื่องส่งสัญญาณวิทยุ เครื่องส่งสัญญาณโทรทัศน์ เป็นต้น</div> <div>1.2.2 การสื่อสารข้อมูลแบบฮาล์ฟดูเพล็กซ์ (Half Duplex) เป็นการติดต่อสื่อสารข้อมูลแบบกึ่งสองทาง คือตัวอุปกรณ์จะเป็นการเปลี่ยนเส้นทางในการรับส่งข้อมูลได้แต่คนละเวลา ตัวอย่างเช่น เครื่องรับส่งวิทยุสมัครเล่น เป็นต้น</div> <div>1.2.3 การสื่อสารข้อมูลแบบฟูลดูเพล็กซ์ (Full Duplex) เป็นการติดต่อสื่อสารข้อมูลสองทิศทาง คืออุปกรณ์สื่อสารจะมีความสามารถเป็นผู้รับข้อมูลและผู้ส่งข้อมูลในเวลาเดียวกันได้ ตัวอย่างเช่น อุปกรณ์โทรศัพท์ เป็นต้น</div>			

	ใบเนื้อหา	หน้าที่ 2
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004	หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ	

ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire

1.3 แบ่งตามลักษณะการสื่อสารข้อมูล ซึ่งสามารถแบ่งออกได้เป็น 2 ประเภทคือ

1.3.1 การสื่อสารข้อมูลแบบอนุกรม (Serial Data Transmission) เป็นการสื่อสารข้อมูลที่มีลักษณะการรับส่งข้อมูลครั้งละ 1 บิต ไปบนสายสัญญาณจนครบจำนวนข้อมูลที่มีอยู่ โดยสามารถนำไปใช้กับสื่อส่งข้อมูลที่มีเพียง 1 ช่องสัญญาณ การสื่อสารข้อมูลแบบอนุกรมแบ่งเป็น 2 ประเภท คือ

- การสื่อสารข้อมูลแบบอะซิงโครนัส (Asynchronous Data Transmission) เป็นวิธีการรับส่งข้อมูลไปบนสื่อส่งข้อมูล โดยไม่ใช้สายสัญญาณนาฬิกากำหนดจังหวะการรับส่งข้อมูล แต่จะใช้วิธีการกำหนดความเร็วของคาบเวลาของการรับส่งข้อมูลแทนสัญญาณการเข้าจังหวะ
- การสื่อสารข้อมูลแบบซิงโครนัส (Synchronous Data Transmission) เป็นการรับส่งข้อมูลไปบนสื่อส่งข้อมูลที่มีลักษณะเป็นกลุ่มของข้อมูลที่ต่อเนื่องกันอย่างเป็นจังหวะ โดยมีสายสัญญาณในการกำหนดจังหวะในการรับส่งข้อมูล


1.3.2 การสื่อสารข้อมูลแบบขนาน (Parallel Data Transmission) เป็นการสื่อสารข้อมูลที่มีลักษณะการรับส่งข้อมูลครั้งละหลายบิตขนานกันไปบนสื่อส่งข้อมูลที่มีหลายช่องสัญญาณ วิธีนี้เป็นวิธีการรับส่งข้อมูลที่เร็วกว่าการรับส่งข้อมูลแบบอนุกรม

การเชื่อมต่อกับอุปกรณ์ภายนอกของระบบงานที่ใช้ไมโครคอนโทรลเลอร์เป็นตัวประมวลผลหลัก ส่วนใหญ่จะเป็นการติดต่อสื่อสารในรูปแบบการสื่อสารข้อมูลแบบอนุกรม เพื่อเป็นการประหยัดค่าสัญญาณของไมโครคอนโทรลเลอร์ และเพื่อให้ง่ายต่อการออกแบบ ซึ่งการสื่อสารข้อมูลแบบอนุกรมของไมโครคอนโทรลเลอร์จะมีหลายรูปแบบได้แก่ การสื่อสารแบบ One-Wire , UART , I2C และ SPI เป็นต้น โดยการสื่อสารแบบ UART , I2C และ SPI ไมโครคอนโทรลเลอร์บางตระกูลจะมีโมดูลพิเศษรองรับไว้ภายในไมโครคอนโทรลเลอร์ที่เรียกว่า Hardware Port แต่ถ้าไมโครคอนโทรลเลอร์บอร์ดใดไม่มีพอร์ตสื่อสารเหล่านี้รองรับก็สามารถสร้างด้วยวิธีการเขียนโปรแกรมเพื่อจำลองการทำงานของพอร์ตการสื่อสารของพอร์ตนั้น ๆ ขึ้นมาเองเรียกว่าการทำ Software Port


2. รูปแบบข้อมูลของการสื่อสารแบบ UART

การสื่อสาร UART เป็นการสื่อสารข้อมูลในรูปแบบอะซิงโครนัส คือการรับส่งข้อมูลภายในสายสัญญาณโดยไม่จำเป็นต้องมีสัญญาณนาฬิกาไปด้วย แต่จะใช้การกำหนดค่าสัญญาณนาฬิกาที่แทนด้วยคาบเวลาของสัญญาณทั้งภาครับและภาคส่งให้ตรงกันที่เรียกว่า อัตราการถ่ายทอดข้อมูล หรือ บอดเรต (baud rate) ที่มีหน่วยเป็น บิตต่อวินาที (bit per second : bps) ซึ่งรูปแบบของข้อมูลที่ใช้ในการรับส่งข้อมูลแบบอะซิงโครนัสประกอบด้วย 4 ส่วน ได้แก่

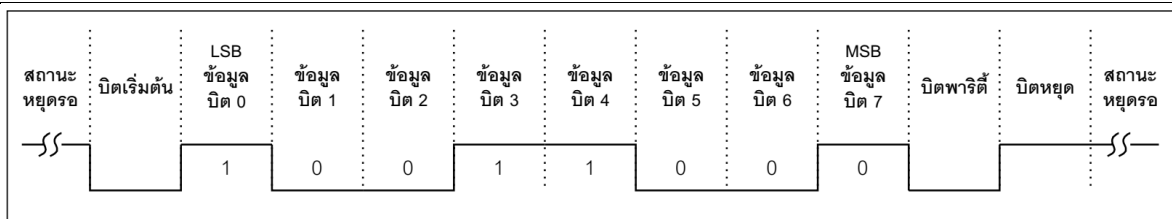
1. บิตเริ่มต้น (Start Bit) ซึ่งจะมีขนาด 1 บิต
2. บิตข้อมูลแบบอนุกรมจะมีขนาด 5,6,7,8 หรือ 9 บิต โดยปัจจุบันส่วนใหญ่จะกำหนดบิตข้อมูลเท่ากับ 8 บิต
3. บิตตรวจสอบพาริตี (Parity Bit) จะมีขนาด 1 บิตหรือไม่ โดยปัจจุบันส่วนใหญ่จะกำหนดแบบไม่มี
4. บิตปิดท้าย (Stop Bit) จะมีขนาด 1,1.5 หรือ 2 บิต โดยปัจจุบันส่วนใหญ่จะกำหนดบิตปิดท้ายเท่ากับ 1 บิต

	ใบเนื้อหา	หน้าที่ 3
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004	หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ	

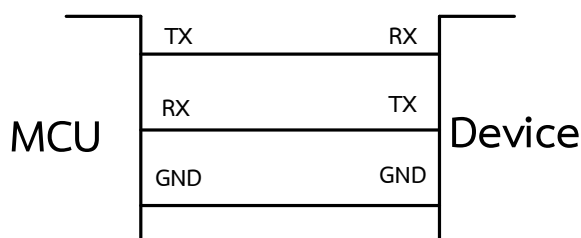
ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire
<p>รูปที่ 1.1 แสดงรูปแบบของการสื่อสารข้อมูลอนุกรมแบบอะซิงโครนัส เมื่อไม่มีข้อมูลที่จะส่ง Data จะมีสถานะเป็นลอจิก ‘1’ ซึ่งจะเรียกสถานะนี้ว่าสถานะหยุดรอ (Waiting Stage) การเริ่มต้นการส่งข้อมูลจะเริ่มจากการให้ขา Data มีลอจิกเป็น ‘0’ ด้วยช่วงระยะเวลา 1 บิต เรียกบิตนี้ว่า บิตเริ่มต้น จากนั้นบิตข้อมูลจะถูกส่งออกไป โดยเริ่มจากบิตที่มีนัยสำคัญต่ำสุด (LSB) ก่อน เมื่อส่งบิตข้อมูลออกไปครบทุกบิตแล้วจะตามด้วย บิตพาริตี เมื่อกำหนดการรับส่งข้อมูลว่าให้มีการรับส่งบิตพาริตี ส่วนสุดท้ายก็จะตามด้วย บิตปิดท้ายที่มีสถานะเป็นลอจิก ‘1’ ที่คาบเวลาขนาด 1,1.5 หรือ 2 บิต ตามที่ผู้ใช้งานกำหนด</p> <p>อุปกรณ์ที่มีพอร์ตการสื่อสารข้อมูลอนุกรมแบบอะซิงโครนัสที่เรียกว่า UART ย่อมาจากคำว่า Universal Asynchronous Receiver Transmitter ซึ่งเป็นพอร์ตที่ใช้สำหรับการเชื่อมต่อและการสื่อสารข้อมูลในรูปแบบอนุกรมกับอุปกรณ์ต่าง ๆ เช่น คอมพิวเตอร์ , RFID, GPS, GSM Module , WIFI Module เป็นต้น ซึ่งสามารถกำหนดความเร็วในการสื่อสารที่มีหน่วยเป็นบิตต่อวินาที ได้แก่ 1200bps , 2400bps , 4800bps , 9600bps , 19200bps , 38400bps , 57600bps และ 115200bps เป็นต้น</p> <p>ปัจจุบันอุปกรณ์ต่อพ่วง เช่น RFID , GPS และ GSM Module จะทำการกำหนดความเร็วในการเชื่อมต่อสื่อสารด้วยพอร์ต UART ที่เป็นค่าเริ่มต้นมีค่าความเร็วในการติดต่อสื่อสารเท่ากับ 9600bps และถ้ากำหนดคุณสมบัติการสื่อสารเป็น บิตเริ่มต้น (Start Bit) มีขนาด 1 บิต บิตข้อมูลมีขนาด 8 บิต บิตตรวจสอบพาริตี (Parity Bit) ไม่มี และบิตปิดท้าย (Stop Bit) ขนาด 1 บิต ดังนั้นการส่งข้อมูล 1 ชุดที่มีขนาด 1 ไบต์จะใช้เวลา 10 บิต แสดงว่าใน 1 วินาทีสามารถรับส่งข้อมูลได้เท่ากับ 9600bps/10 เป็น 960Byte ในกรณีความเร็วในการติดต่อสื่อสาร 9600bps และกำหนดคุณสมบัติการรับส่งข้อมูลดังที่กล่าวมา</p> <p>การสื่อสารข้อมูลด้วยพอร์ต UART จะต้องพิจารณาในเรื่องของระดับแรงดันของลอจิกด้วย เนื่องจากในปัจจุบันอุปกรณ์ไมโครคอนโทรลเลอร์ และอุปกรณ์ต่อพ่วงมีการใช้งานแหล่งจากแรดตันที่ไม่เหมือนกับโดย อุปกรณ์บางชนิดจะใช้ระดับแรงดันไฟฟ้าเป็นมาตรฐาน TTL คือใช้แรงดันไฟฟ้า 5V แต่อุปกรณ์บางชนิดจะใช้ระดับแรงดันไฟฟ้า LVTT คือใช้แรงดันไฟฟ้า 3.3V ดังนั้นก่อนการเลือกใช้อุปกรณ์จะต้องศึกษาว่าอุปกรณ์ที่สื่อสารด้วยพอร์ต UART ใช้ระดับแรงดันไฟเท่ากันหรือไม่ ถ้าไม่จะต้องต่ออุปกรณ์เสริมหรือไม่เพื่อป้องกันความเสียหายที่อาจเกิดขึ้นจากการต่อแรงดันไฟฟ้าที่ต่างระดับกัน และการสื่อสารด้วยพอร์ต UART ยังสามารถแปลงเป็นมาตรฐานอื่นเพื่อทำการเชื่อมต่อกับอุปกรณ์ภายนอก เช่น ถ้าต้องการเชื่อมต่อไมโครคอนโทรลเลอร์กับคอมพิวเตอร์โดยใช้พอร์ต UART จะต้องทำการแปลงสัญญาณ UART ในระดับ TTL ให้เป็นแรงดันในมาตรฐาน RS232 ของพอร์ตคอมพิวเตอร์ก่อน โดยใช้ไอซี MAX232 คือลอจิก ‘0’ ของมาตรฐาน TTL จะถูกเปลี่ยนเป็นแรงดัน 3V – 15V และลอจิก ‘1’ ของมาตรฐาน TTL จะถูกเปลี่ยนเป็นแรงดัน (-3V) – (-15V) เพื่อที่จะสามารถต่อสายสัญญาณได้ไกลสุดประมาณ 50 ฟุต แต่ถ้าต้องการสื่อสารด้วยพอร์ต UART ของไมโครคอนโทรลเลอร์ได้ไกลมากขึ้น หรือต่อแบบ Network ได้จะต้องใช้ไอซี MAX485 หรือ SN75176 เป็นอุปกรณ์ขับที่สามารถต่อสายได้ไกลสุดประมาณ 1.2กิโลเมตร เป็นต้น</p>

	ใบเนื้อหา		หน้าที่ 4
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004		หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ		

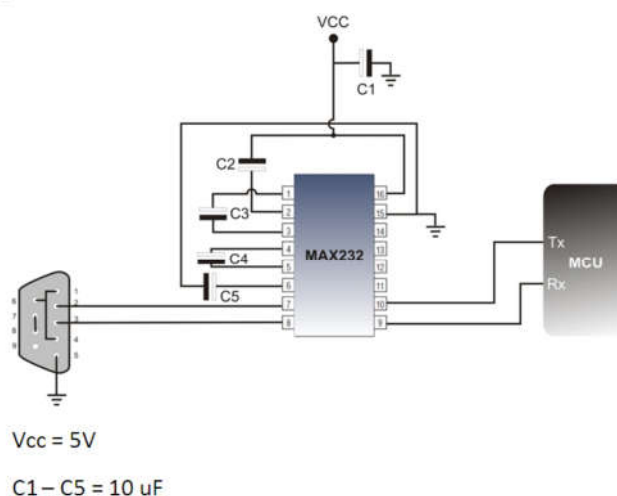
ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire




รูปที่ 1.1 แสดงรูปแบบของสัญญาณการสื่อสารข้อมูลในมาตรฐาน UART
(ที่มา : inex หนังสือเรียนรู้และปฏิบัติการเชื่อมต่อคอมพิวเตอร์กับอุปกรณ์ภายนอกผ่านพอร์ตอนุกรม)



รูปที่ 1.2 แสดงการเชื่อมต่อไมโครคอนโทรลเลอร์กับอุปกรณ์ภายนอกด้วยพอร์ต UART TTL



รูปที่ 1.3 แสดงการเชื่อมต่อไมโครคอนโทรลเลอร์กับคอมพิวเตอร์ด้วยพอร์ต UART TTL to RS232
(ที่มา : <https://www.thaieasyelec.com>)

	ใบเนื้อหา	หน้าที่ 5
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004	หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ	

ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire

3. รูปแบบข้อมูลของการสื่อสารแบบ One-Wire

ระบบบัส 1 สาย (One-Wire) เป็นระบบสื่อสารข้อมูลอนุกรมที่ใช้จำนวนสายสัญญาณเพียง 1 เส้นเท่านั้น โดยไม่ต้องมีสัญญาณนาฬิกาควบคุมจังหวะการถ่ายเทข้อมูลเหมือนกับระบบสื่อสารข้อมูลอนุกรมในแบบอื่น ๆ เนื่องจากสายข้อมูลจะทำหน้าที่เสมือนหนึ่งเป็นสายสัญญาณนาฬิกาในตัว ส่วนค่าของข้อมูลจะพิจารณาจากลักษณะของรูปสัญญาณที่ปรากฏบนสายสัญญาณในแต่ละช่องของเวลาหรือต่อไปนี้จะขอเรียกว่า ไทม์สล็อต (time-slot) โดยคาบเวลาต่ำสุดและสูงสุดของสถานะต่าง ๆ ในการสื่อสารข้อมูลในแต่ละไทม์สล็อต มีการกำหนดขอบเขตไว้อย่างชัดเจน การถ่ายเทข้อมูลจะเกิดขึ้นในแต่ละไทม์สล็อตนั้น รูปแบบการถ่ายเทข้อมูลเป็นแบบอะซิงโครนัสในระดับบิต ไม่มีการกำหนดความยาวของข้อมูลเป็นระดับไบนารี มาตรฐานการสื่อสารข้อมูลแบบ One-Wire เป็นรูปแบบการสื่อสารข้อมูลที่คิดค้นโดยบริษัท ดัลลัสเซมิคอนดักเตอร์ บางครั้งจึงเรียกระบบสื่อสารข้อมูลแบบนี้ว่า ระบบสื่อสารข้อมูลดัลลัสหนึ่งสาย (The Dallas 1-Wire Bus)

คุณสมบัติของไทม์สล็อต

อุปกรณ์มาสเตอร์เป็นอุปกรณ์เพียงตัวเดียวบนระบบบัสที่สามารถอินิเชียลสายสัญญาณได้ โดยอุปกรณ์มาสเตอร์จะเริ่มต้นไทม์สล็อตด้วยการทำให้สายสัญญาณเป็นลอจิกต่ำในช่วงเวลาหนึ่ง จากนั้นทำให้กลับมาเป็นลอจิกสูง ถ้าหากอุปกรณ์สเลฟต้องการส่งข้อมูลมายังอุปกรณ์มาสเตอร์ อุปกรณ์สเลฟจะเป็นตัวควบคุมสถานะของสายสัญญาณต่อไป จนเสร็จสิ้นกระบวนการ แต่ถ้าอุปกรณ์มาสเตอร์ต้องการส่งข้อมูลก็จะสามารถดำเนินการต่อไปได้

เลย


ฟังก์ชันของไทม์สล็อตที่กำหนดโดยอุปกรณ์มาสเตอร์ มีด้วยกัน 4 ฟังก์ชัน คือ

1. รีเซต (RESET) ใช้ในการเริ่มต้นติดต่อกับอุปกรณ์สเลฟ
2. อ่านข้อมูล (READ DATA) ใช้อ่านข้อมูลที่ส่งมาจากอุปกรณ์สเลฟ
3. เขียนข้อมูล ‘1’ (WRITE ONE) ใช้เขียนข้อมูล ‘1’ ไปยังอุปกรณ์สเลฟผ่านสายสัญญาณของระบบ
4. เขียนข้อมูล ‘0’ (WRITE ZERO) ใช้เขียนข้อมูล ‘0’ ไปยังอุปกรณ์สเลฟผ่านสายสัญญาณของระบบ

ฟังก์ชันของไทม์สล็อตในอุปกรณ์สเลฟ มีด้วยกัน 3 ฟังก์ชัน คือ

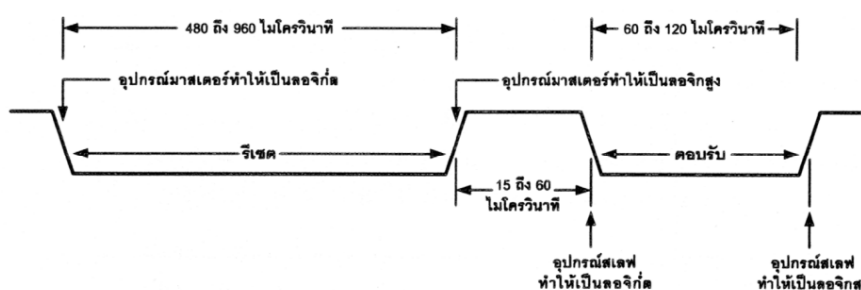
1. ตอบสนอง (PRESENCE) ใช้สำหรับตอบสนองการติดต่อจากอุปกรณ์มาสเตอร์ โดยอุปกรณ์สเลฟตัวที่ถูกเลือกจากอุปกรณ์มาสเตอร์ จะต้องส่งสัญญาณตอบสนองลงบนสายสัญญาณ เพื่อแจ้งให้อุปกรณ์มาสเตอร์ทราบว่าขณะนี้สามารถติดต่อกันได้แล้ว
2. เขียนข้อมูล ‘1’ (WRITE ONE) ใช้สำหรับส่งข้อมูล ‘1’ ไปยังอุปกรณ์มาสเตอร์ผ่านสายสัญญาณของระบบ ซึ่งจะสัมพันธ์กับไทม์สล็อตการอ่านข้อมูลของอุปกรณ์มาสเตอร์
3. เขียนข้อมูล ‘0’ (WRITE ZERO) ใช้สำหรับส่งข้อมูล ‘0’ ไปยังอุปกรณ์มาสเตอร์ผ่านสายสัญญาณของระบบ ซึ่งจะสัมพันธ์กับไทม์สล็อตการอ่านข้อมูลของอุปกรณ์มาสเตอร์

การแยกแยะฟังก์ชันของแต่ละไทม์สล็อต จะใช้ความยาวของคาบเวลาและลักษณะของรูปสัญญาณเป็นตัวกำหนด และทุกครั้งที่มีการเปลี่ยนแปลงฟังก์ชัน ต้องทำให้สายสัญญาณอยู่ในสภาวะว่างเสมอ ซึ่งก็คือ การทำให้สายสัญญาณเป็นลอจิกสูงอย่างน้อยเป็นเวลา 1 ไมโครวินาที

	ใบเนื้อหา		หน้าที่ 6
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004		หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ		

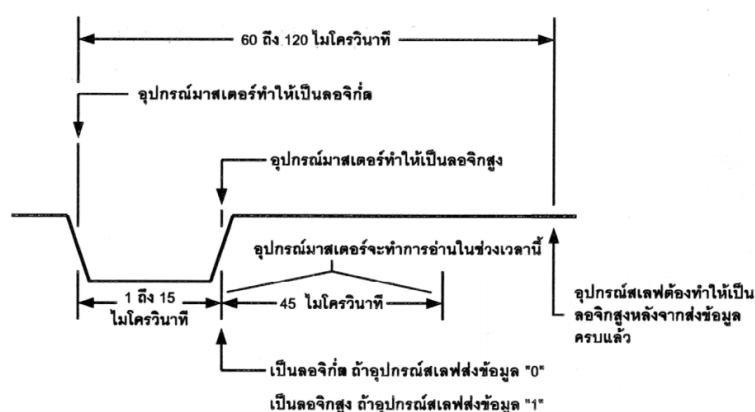
ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire

ไทม์สล็อตการรีเซตและตอบสนอง อุปกรณ์มาสเตอร์ซึ่งในที่นี้ คือ ไมโครคอนโทรลเลอร์จะทำให้เกิดการรีเซตบนสายสัญญาณเพื่อแจ้งแก่อุปกรณ์สเลฟ ด้วยการทำให้สายสัญญาณเป็นลอจิก '0' ต่อเนื่องอย่างน้อย 480 ไมโครวินาที และจะต้องทำให้สายสัญญาณกลับมาเป็นลอจิก '1' ภายใน 480 ไมโครวินาทีหลังจากนั้น ถ้าหากมีอุปกรณ์สเลฟต่ออยู่บนสายสัญญาณจะมีการตอบสนองสัญญาณรีเซตนั้นด้วยสัญญาณตอบสนอง (PRESENCE) ด้วยการทำให้สายสัญญาณเป็นลอจิก '0' ต่อเนื่องนานประมาณ 60 ถึง 240 ไมโครวินาทีหลังจากที่สัญญาณรีเซตปรากฏขึ้นประมาณ 15 ถึง 60 ไมโครวินาที




รูปที่ 1.4 แสดงคาบเวลาของไทม์สล็อตการรีเซตและตอบสนอง

ไทม์สล็อตการอ่านข้อมูลของไมโครคอนโทรลเลอร์และการเขียนข้อมูลของอุปกรณ์สเลฟ เมื่อต้องการอ่านข้อมูลจากอุปกรณ์สเลฟ ไมโครคอนโทรลเลอร์จะทำให้สายสัญญาณเป็นลอจิก '0' ประมาณ 1 ถึง 15 ไมโครวินาที จากนั้นต้องทำให้สถานะของสายกลับมาเป็นลอจิก '1' อุปกรณ์สเลฟจะส่งข้อมูลกลับมาให้ไมโครคอนโทรลเลอร์ ถ้าข้อมูลเป็น '0' อุปกรณ์สเลฟจะทำให้สายสัญญาณเป็นลอจิก '0' นานประมาณ 45 ไมโครวินาที แล้วทำให้สายสัญญาณกลับมาสู่สภาวะลอจิก '1' อีกครั้ง แต่ถ้าเป็นข้อมูล '1' อุปกรณ์สเลฟจะทำให้สายสัญญาณเป็นลอจิก '1' ต่อเนื่องไปอีก 45 ไมโครวินาที รวมเวลาทั้งหมดประมาณ 60 ถึง 120 ไมโครวินาที นั่นคือในไทม์สล็อตนี้ต้องใช้เวลารวมไม่เกิน 120 ไมโครวินาที ในขณะที่ไมโครคอนโทรลเลอร์จะใช้เวลาในการอ่านข้อมูลอยู่ระหว่าง 15 และ 60 ไมโครวินาทีหลังจากเริ่มต้นไทม์สล็อตนี้

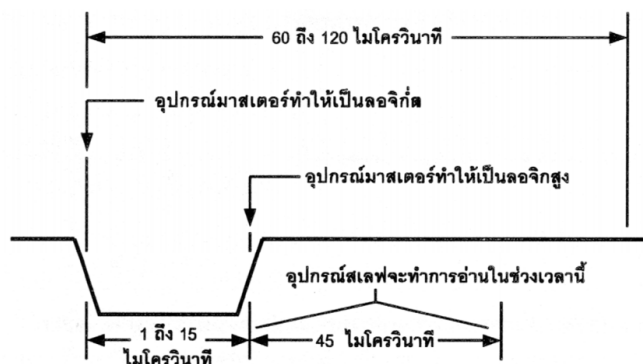


รูปที่ 1.5 แสดงไทม์สล็อตการอ่านข้อมูลของไมโครคอนโทรลเลอร์และการเขียนข้อมูลของอุปกรณ์สเลฟ

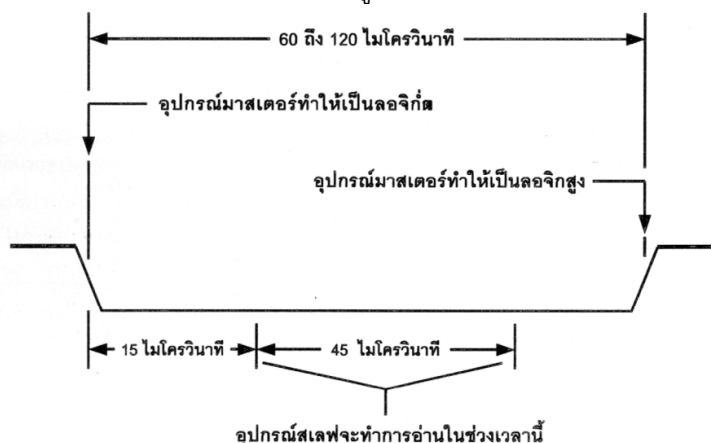
	ใบเนื้อหา		หน้าที่ 7
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004		หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ		

ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire

ไทม์สล็อตการเขียนข้อมูลของไมโครคอนโทรลเลอร์ เมื่อไมโครคอนโทรลเลอร์ต้องการเขียนข้อมูล จะเริ่มต้นกระบวนการด้วยการทำให้สายสัญญาณเป็นลอจิก '0' ประมาณ 1 ถึง 15 ไมโครวินาที จากนั้นทำให้สถานะของสายกลับมาเป็นลอจิก '1' แล้วดำเนินการเขียนข้อมูลได้ในทันที ถ้าต้องการเขียนข้อมูล '0' ไมโครคอนโทรลเลอร์จะทำให้สายสัญญาณเป็นลอจิก '0' นานประมาณ 45 ไมโครวินาที แล้วทำให้สายสัญญาณกลับมาสู่สภาวะลอจิก '1' อีกครั้ง แต่ถ้าต้องการเขียนข้อมูล '1' ไมโครคอนโทรลเลอร์จะทำให้สายสัญญาณเป็นลอจิก '1' ต่อเนื่องไปอีก 45 ไมโครวินาที รวมเวลาทั้งหมดในไทม์สล็อตนี้ประมาณ 60 ถึง 120 ไมโครวินาที




รูปที่ 1.6 แสดงไทม์สล็อตการเขียนข้อมูลลอจิก '1' ของไมโครคอนโทรลเลอร์




รูปที่ 1.7 แสดงไทม์สล็อตการเขียนข้อมูลลอจิก '0' ของไมโครคอนโทรลเลอร์

รูปแบบของการสื่อสารข้อมูลแบบ 1 สาย การสื่อสารข้อมูลในระบบบัสหนึ่งสายไมโครคอนโทรลเลอร์จะติดต่อกับอุปกรณ์สเลฟได้ครั้งละ 1 ตัวเท่านั้น ดังนั้น อุปกรณ์สเลฟแต่ละตัวต้องมีข้อมูลกำหนดแอดเดรสเฉพาะตัว โดยจะเก็บไว้ในหน่วยความจำรวมภายในอุปกรณ์สเลฟตัวนั้น ๆ โดยปกติอุปกรณ์สเลฟในระบบบัสหนึ่งสายนี้จะมีหน่วยความจำขนาด 64 บิต หรือ 8 ไบต์ สำหรับเก็บข้อมูลต่าง ๆ ที่สำคัญของอุปกรณ์แต่ละตัว ซึ่งประกอบด้วย

1. รหัสของตระกูล จำนวน 8 บิต
2. เลขหมายประจำตัว (serial number) จำนวน 48 บิต
3. รหัสตรวจสอบความผิดพลาด (CRC: Cyclical Redundancy Check) จำนวน 8 บิต

	ใบเนื้อหา	หน้าที่ 8
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004	หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ	

ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire
<p>ผู้ใช้งานสามารถอ่านข้อมูลประจำตัวของอุปกรณ์สเลฟได้ด้วยการใช้คำสั่ง Read ROM หรือคำสั่งอ่านหน่วยความจำรวม ในกรณีที่บนสายสัญญาณมีอุปกรณ์สเลฟเพียงตัวเดียวไม่จำเป็นต้องอ้างแอดเดรสในการติดต่อ</p> <p>รูปแบบการติดต่อบนระบบบัสหนึ่งสายเริ่มต้นขึ้น เมื่ออุปกรณ์มาสเตอร์ทำการรีเซตและกำหนดแอดเดรสของอุปกรณ์ที่ทำการติดต่อ ถ้าหากมีอุปกรณ์สเลฟเพียงตัวเดียวสามารถข้ามขั้นตอนการติดต่อกับหน่วยความจำรวมในอุปกรณ์สเลฟได้ เรียกวิธีการดังกล่าวว่า การไม่ติดต่อหน่วยความจำรวม หรือ สคิปรอม (Skip ROM) จากนั้นรอการตอบรับจากอุปกรณ์สเลฟ เมื่อการตอบรับสมบูรณ์ก็จะสามารถเริ่มต้นขั้นตอนการอ่านหรือเขียนข้อมูลได้ต่อไป</p>
<p>4. รีจิสเตอร์ที่เกี่ยวข้องกับการใช้งานพอร์ต UART ของไมโครคอนโทรลเลอร์</p> <p>การใช้งานพอร์ต UART ของไมโครคอนโทรลเลอร์แต่ละตระกูลจะต้องสั่งผ่านรีจิสเตอร์ที่เกี่ยวข้องต่าง ๆ ของพอร์ต UART โดยรีจิสเตอร์ที่เกี่ยวข้องจะแบ่งออกเป็น 2 กลุ่ม ได้แก่ กลุ่มควบคุมการทำงานของพอร์ต UART และกลุ่มที่ใช้ในการเก็บค่าข้อมูลที่รับส่งผ่านพอร์ต UART ซึ่งไมโครคอนโทรลเลอร์แต่ละตระกูลจะมีรีจิสเตอร์ที่เกี่ยวข้องดังนี้</p> <p>4.1 รีจิสเตอร์ที่เกี่ยวข้องกับการใช้งานพอร์ต UART ของ AT89C51ED2</p> <p>พอร์ตอนุกรมของ MCS-51 สามารถสื่อสารข้อมูลทั้งแบบรับและส่งในเวลาเดียวกัน เรียกว่า Full duplex โดยจะใช้ขา P3.0 สำหรับรับข้อมูล และขา P3.1 สำหรับส่งข้อมูล ซึ่งภายในไมโครคอนโทรลเลอร์จะมีรีจิสเตอร์สำหรับทำหน้าที่รับและส่งข้อมูลแยกกันต่างหาก แต่การใช้งานจะอ้างถึงรีจิสเตอร์ 2 ตัวนี้ ด้วยชื่อเดียวกันคือ SBUF (แอดเดรสที่ 99H ใน SFR) เช่น ถ้าต้องการส่งข้อมูลก็เขียนข้อมูลลงไป SBUF หรือถ้าต้องการรับข้อมูลก็ให้อ่านข้อมูลจาก SBUF ดังนั้น ก่อนการใช้งานพอร์ตอนุกรมจะต้องกำหนดให้ MCS-51 รับรู้ก่อนว่าจำนวนบิตที่จะรับส่งแต่ละครั้งเท่าไร Baud rate ที่ใช้รับส่งข้อมูลมีความเร็วเท่าไร โดยการกำหนดโหมดการทำงานของพอร์ตอนุกรมจะกำหนดผ่านรีจิสเตอร์ควบคุมพอร์ตอนุกรม หรือ SCON (Serial Port Control Register) ซึ่งหน้าที่แต่ละบิตของรีจิสเตอร์ SCON อธิบายได้ดังนี้</p> <p>SM0 (Serial port mode bit 0) สำหรับกำหนดโหมดการทำงานของพอร์ตอนุกรม โดยจะใช้ร่วมกับบิต SM1</p> <p>SM1 (Serial port mode bit 1) ใช้ร่วมกับบิต SM0 ในการกำหนดโหมดการทำงานของพอร์ตอนุกรม ซึ่งมีรายละเอียดดังรูปที่ 1.8</p> <p>SM2 (Serial ports mode bit 2) ใช้ควบคุมเมื่อมีการเชื่อมต่อไมโครคอนโทรลเลอร์หลายตัวในโหมด 2 และโหมด 3 นั่นคือ เมื่อกำหนดเป็นลอจิก ‘0’ จะให้ทำงานในโหมด 0 และถ้ากำหนดเป็นลอจิก ‘1’ จะให้ทำงานในโหมดหลายไมโครคอนโทรลเลอร์</p> <p>REN (Received enable bit) ใช้ Enable การรับข้อมูล นั่นคือ เมื่อกำหนดค่าเป็นลอจิก ‘1’ ให้สามารถรับข้อมูลได้ เมื่อกำหนดเป็นลอจิก ‘0’ เป็นการกำหนดไม่ให้อ่านข้อมูล</p> <p>TB8 (Transmit data bit 8) สำหรับเก็บข้อมูลบิตที่ 9 เมื่อต้องการส่งข้อมูลออกพอร์ตอนุกรมในโหมด 2 และโหมด 3</p>

	ใบเนื้อหา		หน้าที่ 9
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004		หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ		

ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire

RB8 (Receive data bit 8) สำหรับเก็บข้อมูลบิตที่ 9 เมื่อต้องการรับข้อมูลจากพอร์ตอนุกรมในโหมด 2 และโหมด 3

TI (Transmit Interrupt Flag) เป็นบิตที่ใช้แสดงสถานะการส่งข้อมูลออกพอร์ตอนุกรม เมื่อ MCS-51 ส่งข้อมูลเรียบร้อยแล้วก็จะเซตบิต TI เป็นลอจิก 1 (ไม่มีข้อมูลใน SBUF) เพื่อบอกให้โปรแกรมรู้ว่าสามารถส่งข้อมูลอื่นได้ (นำข้อมูลมาใส่ใน SBUF)

RI (Receive Interrupt Flag) เป็นบิตที่ใช้แสดงสถานะการรับข้อมูลจากพอร์ตอนุกรม เมื่อ MCS-51 รับข้อมูลเรียบร้อยแล้ว (มีข้อมูลมาที่ SBUF) ก็จะเซตบิต RI เป็นลอจิก 1 เพื่อบอกให้โปรแกรมรู้ว่าต้องอ่านข้อมูลมาจาก SBUF อย่างรวดเร็ว ก่อนที่ข้อมูลอื่นจะเข้ามาแทนที่

SCON: SERIAL PORT CONTROL REGISTER. BIT ADDRESSABLE.

SM0	SM1	SM2	REN	TB8	RB8	TI	RI
-----	-----	-----	-----	-----	-----	----	----

SM0 SCON. 7 Serial Port mode specifier. (NOTE 1).

SM1 SCON. 6 Serial Port mode specifier. (NOTE 1).

SM2 SCON. 5 Enables the multiprocessor communication feature in modes 2 & 3. In mode 2 or 3, if SM2 is set to 1 then RI will not be activated if the received 9th data bit (RB8) is 0. In mode 1, if SM2 = 1 then RI will not be activated if a valid stop bit was not received. In mode 0, SM2 should be 0. (See Table 9).

REN SCON. 4 Set/Cleared by software to Enable/Disable reception.

TB8 SCON. 3 The 9th bit that will be transmitted in modes 2 & 3. Set/Cleared by software.

RB8 SCON. 2 In modes 2 & 3, is the 9th data bit that was received. In mode 1, if SM2 = 0, RB8 is the stop bit that was received. In mode 0, RB8 is not used.

TI SCON. 1 Transmit interrupt flag. Set by hardware at the end of the 8th bit time in mode 0, or at the beginning of the stop bit in the other modes. Must be cleared by software.

RI SCON. 0 Receive interrupt flag. Set by hardware at the end of the 8th bit time in mode 0, or halfway through the stop bit time in the other modes (except see SM2). Must be cleared by software.

NOTE 1:


SM0	SM1	Mode	Description	Baud Rate
0	0	0	SHIFT REGISTER	Fosc./12
0	1	1	8-Bit UART	Variable
1	0	2	9-Bit UART	Fosc./64 OR Fosc./32
1	1	3	9-Bit UART	Variable

SERIAL PORT SET-UP:


Table 9

MODE	SCON	SM2 VARIATION
0	10H	Single Processor Environment (SM2 = 0)
1	50H	
2	90H	
3	D0H	
0	NA	Multiprocessor Environment (SM2 = 1)
1	70H	
2	B0H	
3	F0H	

รูปที่ 1.8 แสดงหน้าที่การทำงานของบิตต่าง ๆ ภายในรีจิสเตอร์ SCON

	ใบเนื้อหา	หน้าที่ 10
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004	หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ	

ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire
<p>การกำหนด Baud rate</p> <p>การกำหนด Baud rate คือการกำหนดความเร็วที่ใช้ในการรับ-ส่งข้อมูลของพอร์ตอนุกรม การกำหนด Baud rate นี้จะทำเฉพาะในโหมด 1 และ 3 เท่านั้น ส่วนในโหมด 0 และโหมด 2 จะถูกกำหนดให้โดยอัตโนมัติตามความถี่สัญญาณนาฬิกา (Crystal) ที่ป้อนให้กับ MCS-51 สำหรับโหมด 0 ค่า Baud rate เกิดจากการนำเอาความถี่สัญญาณนาฬิกาหารด้วย 12 เช่น ถ้า Crystal ความถี่ 11.0592 MHz ค่า Baud rate ที่ได้จะเป็น 921,583 bps (bps = baud per second) สำหรับในโหมด 2 ค่า Baud rate เกิดจากการนำเอาความถี่สัญญาณนาฬิกาหารด้วย 64 เช่น ถ้า Crystal ความถี่ 11.0592 MHz ค่า Baud rate ที่ได้จะเป็น 172,797 bps ส่วนในโหมด 1 และโหมด 3 ค่า Baud rate จะกำหนดโดยการให้ไทมเมอร์ 1 เกิดการนับเกิน (Overflow) วิธีที่ทำให้เกิดการนับเกินของไทมเมอร์ 1 มีหลายวิธี แต่ที่นิยมใช้คือ กำหนดให้ไทมเมอร์ 1 ทำงานในโหมด 2 (8 bit auto-reload mode) และโหลดค่าที่จะทำให้ เกิดการนับเกินลงใน TH1</p> <p>การกำหนดค่าให้ TH1 เพื่อกำเนิด Baud rate กรณีที่บิต PCON.7 เป็นลอจิก 0 คำนวณได้ดังนี้ $TH1 = 256 - ((Crystal/384)/Baud)$ ถ้า PCON.7 เป็นลอจิก 1 ค่า Baud rate จะเป็น 2 เท่า คำนวณได้ดังนี้ $TH1 = 256 - ((Crystal/192)/Baud)$</p> <p>ตัวอย่าง สมมติว่า Crystal ความถี่ 11.0592 MHz ต้องการกำหนด Baud rate ที่ 19,200 bps จงคำนวณหาค่าที่จะโหลดลงใน TH1</p> <p>วิธีคิด</p> $TH1 = 256 - ((Crystal)/384)/Baud$ $TH1 = 256 - ((11059200)/384)/19200$ $TH1 = 256 - ((28800)/19200)$ $TH1 = 256 - 1.5 = 254.5$ <p>ซึ่งค่าที่ได้เป็นเลขทศนิยม ซึ่งการใช้งานจริงจะต้องกำหนดเป็นจำนวนเต็ม เมื่อทำการปัดขึ้นหรือลง ค่า Baud rate ที่ได้จะไม่ถูกต้อง จึงต้องเปลี่ยนการคำนวณเป็น Baud rate 2 เท่า คำนวณใหม่ได้ดังนี้</p> $TH1 = 256 - ((Crystal)/192)/Baud$ $TH1 = 256 - ((11059200)/192)/19200$ $TH1 = 256 - ((57600)/19200)$ $TH1 = 256 - 3 = 253$ <p>จากนั้นนำค่าที่ได้แปลงเป็นฐานสิบหกก่อนจะโหลดลง TH1 ซึ่งค่าที่ได้คืออัตราบอดเรต สรุปลงขั้นตอนในการกำหนดค่า Baud rate 19200 bps ให้กับ MCS-51 ได้ดังนี้</p> <ol style="list-style-type: none">1. กำหนดพอร์ตอนุกรมทำงานในโหมด 1 หรือ 32. กำหนดให้ไทมเมอร์ 1 ทำงานในโหมด 2 (8-bit auto-reload)3. กำหนดค่า TH1 เป็นค่าการนับที่สร้างอัตราบอดเรต4. เซตบิต PCON.7 (SMOD) เป็นลอจิก 1 เพื่อกำหนด Baud rate เป็นแบบ 2 เท่า

	ใบเนื้อหา		หน้าที่ 11
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004		หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ		

ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire

4.2 รีจิสเตอร์ที่เกี่ยวข้องกับการใช้งานพอร์ต UART ของ PIC16F887

พอร์ตอนุกรมของ PIC16F887 สามารถสื่อสารข้อมูลทั้งแบบรับและส่งในเวลาเดียวกัน เรียกว่า Full duplex โดยจะใช้ขา RC7 สำหรับรับข้อมูล และขา RC6 สำหรับส่งข้อมูล ซึ่งภายในไมโครคอนโทรลเลอร์จะมี รีจิสเตอร์สำหรับทำหน้าที่รับและส่งข้อมูลแยกกันต่างหาก โดยรีจิสเตอร์ที่ทำหน้าที่ในการรับข้อมูลชื่อ RCREG ส่วน รีจิสเตอร์ที่ทำหน้าที่ในการส่งข้อมูลชื่อ TXREG ซึ่งก่อนการใช้งานพอร์ตอนุกรมจะต้องกำหนดให้ PIC16F887 รับรู้ ก่อนว่าจำนวนบิตที่จะรับส่งแต่ละครั้งเท่าไร Baud rate ที่ใช้รับส่งข้อมูลมีความเร็วเท่าไร โดยการกำหนดคุณสมบัติ การทำงานของพอร์ตอนุกรม จะกำหนดผ่านรีจิสเตอร์ควบคุมพอร์ตอนุกรมได้แก่รีจิสเตอร์ RCSTA และ TXSTA ดังนี้

RCSTA คือรีจิสเตอร์ที่ทำหน้าที่ในการกำหนดคุณสมบัติการรับข้อมูลจากพอร์ตอนุกรม UART ของ ไมโครคอนโทรลเลอร์ PIC16F887 ซึ่งมีรายละเอียดดังรูปที่ 1.9

REGISTER 12-2: RCSTA: RECEIVE STATUS AND CONTROL REGISTER⁽¹⁾


R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
bit 7							bit 0


Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared
		x = Bit is unknown

bit 7	SPEN: Serial Port Enable bit 1 = Serial port enabled (configures RX/DT and TX/CK pins as serial port pins) 0 = Serial port disabled (held in Reset)
bit 6	RX9: 9-bit Receive Enable bit 1 = Selects 9-bit reception 0 = Selects 8-bit reception
bit 5	SREN: Single Receive Enable bit <u>Asynchronous mode:</u> Don't care <u>Synchronous mode – Master:</u> 1 = Enables single receive 0 = Disables single receive This bit is cleared after reception is complete. <u>Synchronous mode – Slave</u> Don't care
bit 4	CREN: Continuous Receive Enable bit <u>Asynchronous mode:</u> 1 = Enables receiver 0 = Disables receiver <u>Synchronous mode:</u> 1 = Enables continuous receive until enable bit CREN is cleared (CREN overrides SREN) 0 = Disables continuous receive
bit 3	ADDEN: Address Detect Enable bit <u>Asynchronous mode 9-bit (RX9 = 1):</u> 1 = Enables address detection, enable interrupt and load the receive buffer when RSR<8> is set 0 = Disables address detection, all bytes are received and ninth bit can be used as parity bit <u>Asynchronous mode 8-bit (RX9 = 0):</u> Don't care
bit 2	FERR: Framing Error bit 1 = Framing error (can be updated by reading RCREG register and receive next valid byte) 0 = No framing error
bit 1	OERR: Overrun Error bit 1 = Overrun error (can be cleared by clearing bit CREN) 0 = No overrun error
bit 0	RX9D: Ninth bit of Received Data This can be address/data bit or a parity bit and must be calculated by user firmware.

รูปที่ 1.9 แสดงรายละเอียดการใช้งานบิตต่าง ๆ ของรีจิสเตอร์ RCSTA

	ใบเนื้อหา		หน้าที่ 12																								
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004		หน่วยที่ 8																								
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ																										
ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire																											
TXSTA คือรีจิสเตอร์ที่ทำหน้าที่ในการกำหนดคุณสมบัติการส่งข้อมูลออกจากพอร์ตอนุกรม UART ของไมโครคอนโทรลเลอร์ PIC16F887 ซึ่งมีรายละเอียดดังรูปที่ 1.10																											
REGISTER 12-1: TXSTA: TRANSMIT STATUS AND CONTROL REGISTER																											
<table><tr><td>R/W-0</td><td>R/W-0</td><td>R/W-0</td><td>R/W-0</td><td>R/W-0</td><td>R/W-0</td><td>R-1</td><td>R/W-0</td></tr><tr><td>CSRC</td><td>TX9</td><td>TXEN⁽¹⁾</td><td>SYNC</td><td>SENDB</td><td>BRGH</td><td>TRMT</td><td>TX9D</td></tr><tr><td colspan="6">bit 7</td><td colspan="2">bit 0</td></tr></table>				R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-1	R/W-0	CSRC	TX9	TXEN ⁽¹⁾	SYNC	SENDB	BRGH	TRMT	TX9D	bit 7						bit 0	
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-1	R/W-0																				
CSRC	TX9	TXEN ⁽¹⁾	SYNC	SENDB	BRGH	TRMT	TX9D																				
bit 7						bit 0																					
<div>Legend:</div> <div><div>R = Readable bit</div><div>W = Writable bit</div><div>U = Unimplemented bit, read as '0'</div><div>-n = Value at POR</div><div>'1' = Bit is set</div><div>'0' = Bit is cleared</div><div>x = Bit is unknown</div></div>																											
<div>bit 7</div> <div>CSRC: Clock Source Select bit</div> <div>Asynchronous mode:</div> <div>Don't care</div> <div>Synchronous mode:</div> <div>1 = Master mode (clock generated internally from BRG)</div> <div>0 = Slave mode (clock from external source)</div> <div>bit 6</div> <div>TX9: 9-bit Transmit Enable bit</div> <div>1 = Selects 9-bit transmission</div> <div>0 = Selects 8-bit transmission</div> <div>bit 5</div> <div>TXEN: Transmit Enable bit⁽¹⁾</div> <div>1 = Transmit enabled</div> <div>0 = Transmit disabled</div> <div>bit 4</div> <div>SYNC: EUSART Mode Select bit</div> <div>1 = Synchronous mode</div> <div>0 = Asynchronous mode</div> <div>bit 3</div> <div>SENDB: Send Break Character bit</div> <div>Asynchronous mode:</div> <div>1 = Send Sync Break on next transmission (cleared by hardware upon completion)</div> <div>0 = Sync Break transmission completed</div> <div>Synchronous mode:</div> <div>Don't care</div> <div>bit 2</div> <div>BRGH: High Baud Rate Select bit</div> <div>Asynchronous mode:</div> <div>1 = High speed</div> <div>0 = Low speed</div> <div>Synchronous mode:</div> <div>Unused in this mode</div> <div>bit 1</div> <div>TRMT: Transmit Shift Register Status bit</div> <div>1 = TSR empty</div> <div>0 = TSR full</div> <div>bit 0</div> <div>TX9D: Ninth bit of Transmit Data</div> <div>Can be address/data bit or a parity bit.</div>																											
Note 1: SREN/CREN overrides TXEN in Sync mode.																											
รูปที่ 1.10 แสดงรายละเอียดการใช้งานบิตต่าง ๆ ของรีจิสเตอร์ TXSTA																											

	ใบเนื้อหา	หน้าที่ 13
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004	หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ	

ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire
<p>การกำหนด Baud rate</p> <p>การกำหนด Baud rate ของไมโครคอนโทรลเลอร์ PIC16F887 เพื่อกำหนดอัตราความเร็วในการติดต่อสื่อสารด้วยพอร์ต UART สามารถทำได้โดยกำหนดที่รีจิสเตอร์ BAUDCTL , SPBRG และ SPBRGH ซึ่งการหาค่าพารามิเตอร์ต่าง ๆ เกี่ยวกับการกำหนดอัตราความเร็วในการติดต่อสื่อสารของพอร์ต UART สามารถกระทำได้ตามรูปที่ 1.11 ดังนี้</p>


For a device with FOSC of 16 MHz, desired baud rate of 9600, Asynchronous mode, 8-bit BRG:

$$\text{Desired Baud Rate} = \frac{F_{OSC}}{64([SPBRGH:SPBRG] + 1)}$$

Solving for SPBRGH:SPBRG:

$$X = \frac{\frac{F_{OSC}}{\text{Desired Baud Rate}}}{64} - 1$$
$$= \frac{\frac{16000000}{9600}}{64} - 1$$
$$= [25.042] = 25$$
$$\text{Calculated Baud Rate} = \frac{16000000}{64(25 + 1)}$$
$$= 9615$$
$$\text{Error} = \frac{\text{Calc. Baud Rate} - \text{Desired Baud Rate}}{\text{Desired Baud Rate}}$$
$$= \frac{(9615 - 9600)}{9600} = 0.16\%$$

รูปที่ 1.11 แสดงตัวอย่างการคำนวณค่าของรีจิสเตอร์ SPBRGH และ SPBRG และค่าความผิดพลาดในการกำหนดอัตราความเร็วการสื่อสารที่ 9600bps โดยใช้ค่า XTAL 16MHz

	ใบเนื้อหา		หน้าที่ 14
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004		หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ		

ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire

SPBRG และ SPBRGH คือรีจิสเตอร์ที่ทำหน้าที่ในการกำหนดค่าความเร็วในการติดต่อสื่อสารด้วยพอร์ต UART ซึ่งสามารถคำนวณค่าได้ตามรูปที่ 1.11

BAUDCTL คือรีจิสเตอร์ที่ทำหน้าที่ในการกำหนดคุณสมบัติในการกำหนดค่าความเร็วในการติดต่อสื่อสารของพอร์ต UART ซึ่งมีรายละเอียดดังรูปที่ 1.12

REGISTER 12-3: BAUDCTL: BAUD RATE CONTROL REGISTER


R-0	R-1	U-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN
bit 7			bit 0				

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
 -n = Value at POR '1' = Bit is set '0' = Bit is cleared x = Bit is unknown

bit 7	ABDOVF: Auto-Baud Detect Overflow bit <u>Asynchronous mode:</u> 1 = Auto-baud timer overflowed 0 = Auto-baud timer did not overflow <u>Synchronous mode:</u> Don't care
bit 6	RCIDL: Receive Idle Flag bit <u>Asynchronous mode:</u> 1 = Receiver is Idle 0 = Start bit has been received and the receiver is receiving <u>Synchronous mode:</u> Don't care
bit 5	Unimplemented: Read as '0'
bit 4	SCKP: Synchronous Clock Polarity Select bit <u>Asynchronous mode:</u> 1 = Transmit inverted data to the RB7/TX/CK pin 0 = Transmit non-inverted data to the RB7/TX/CK pin <u>Synchronous mode:</u> 1 = Data is clocked on rising edge of the clock 0 = Data is clocked on falling edge of the clock
bit 3	BRG16: 16-bit Baud Rate Generator bit 1 = 16-bit Baud Rate Generator is used 0 = 8-bit Baud Rate Generator is used
bit 2	Unimplemented: Read as '0'
bit 1	WUE: Wake-up Enable bit <u>Asynchronous mode:</u> 1 = Receiver is waiting for a falling edge. No character will be received byte RCIF will be set. WUE will automatically clear after RCIF is set. 0 = Receiver is operating normally <u>Synchronous mode:</u> Don't care
bit 0	ABDEN: Auto-Baud Detect Enable bit <u>Asynchronous mode:</u> 1 = Auto-Baud Detect mode is enabled (clears when auto-baud is complete) 0 = Auto-Baud Detect mode is disabled <u>Synchronous mode:</u> Don't care

รูปที่ 1.12 แสดงรายละเอียดการใช้งานบิตต่าง ๆ ของรีจิสเตอร์ BAUDCTL

	ใบเนื้อหา		หน้าที่ 15
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004		หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ		

ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire

4.3 รีจิสเตอร์ที่เกี่ยวข้องกับการใช้งานพอร์ต UART ของ ATMEGA32

พอร์ตอนุกรมของ ATMEGA32 สามารถสื่อสารข้อมูลทั้งแบบรับและส่งในเวลาเดียวกัน เรียกว่า Full duplex โดยจะใช้ขา PD0 สำหรับรับข้อมูล และขา PD1 สำหรับส่งข้อมูล ซึ่งภายในไมโครคอนโทรลเลอร์จะมี รีจิสเตอร์สำหรับทำหน้าที่รับและส่งข้อมูลแยกกันต่างหาก โดยรีจิสเตอร์ที่ทำหน้าที่ในการรับและส่งข้อมูลจะมีชื่อเดียวกันคือ UDR ซึ่งก่อนการใช้งานพอร์ตอนุกรมจะต้องกำหนดให้ ATMEGA32 รับรู้ก่อนว่าจำนวนบิตที่จะรับส่งแต่ละครั้งเท่าไร Baud rate ที่ใช้รับส่งข้อมูลมีความเร็วเท่าไร โดยการกำหนดคุณสมบัติการทำงานของพอร์ตอนุกรม จะกำหนดผ่านรีจิสเตอร์ควบคุมพอร์ตอนุกรมได้แก่รีจิสเตอร์ UCSRA , UCSRB และ UCSRC ดังนี้

UCSRA คือรีจิสเตอร์แสดงสถานะการรับส่งข้อมูลของพอร์ต UART และควบคุมการทำงานของพอร์ต UART ซึ่งมีรายละเอียดดังรูปที่ 1.13

Bit	7	6	5	4	3	2	1	0	
	RXC	TXC	UDRE	FE	DOR	PE	U2X	MPCM	UCSRA
Read/Write	R	R/W	R	R	R	R	R/W	R/W	
Initial Value	0	0	1	0	0	0	0	0	

- **Bit 7 – RXC: USART Receive Complete**

This flag bit is set when there are unread data in the receive buffer and cleared when the receive buffer is empty (i.e., does not contain any unread data). If the receiver is disabled, the receive buffer will be flushed and consequently the RXC bit will become zero. The RXC Flag can be used to generate a Receive Complete interrupt (see description of the RXCIE bit).

- **Bit 6 – TXC: USART Transmit Complete**

This flag bit is set when the entire frame in the transmit Shift Register has been shifted out and there are no new data currently present in the transmit buffer (UDR). The TXC Flag bit is automatically cleared when a transmit complete interrupt is executed, or it can be cleared by writing a one to its bit location. The TXC Flag can generate a Transmit Complete interrupt (see description of the TXCIE bit).

- **Bit 5 – UDRE: USART Data Register Empty**

The UDRE Flag indicates if the transmit buffer (UDR) is ready to receive new data. If UDRE is one, the buffer is empty, and therefore ready to be written. The UDRE Flag can generate a Data Register empty Interrupt (see description of the UDRIE bit).

UDRE is set after a reset to indicate that the transmitter is ready.


- **Bit 4 – FE: Frame Error**

This bit is set if the next character in the receive buffer had a Frame Error when received. i.e., when the first stop bit of the next character in the receive buffer is zero. This bit is valid until the receive buffer (UDR) is read. The FE bit is zero when the stop bit of received data is one. Always set this bit to zero when writing to UCSRA.

- **Bit 3 – DOR: Data OverRun**

This bit is set if a Data OverRun condition is detected. A Data OverRun occurs when the receive buffer is full (two characters), it is a new character waiting in the receive Shift Register, and a new start bit is detected. This bit is valid until the receive buffer (UDR) is read. Always set this bit to zero when writing to UCSRA.

รูปที่ 1.13 แสดงรายละเอียดการใช้งานบิตต่าง ๆ ของรีจิสเตอร์ UCSRA

	ใบเนื้อหา	หน้าที่ 16
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004	หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ	

ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire

- Bit 2 – PE: Parity Error**

This bit is set if the next character in the receive buffer had a Parity Error when received and the parity checking was enabled at that point (UPM1 = 1). This bit is valid until the receive buffer (UDR) is read. Always set this bit to zero when writing to UCSRA.
- Bit 1 – U2X: Double the USART Transmission Speed**

This bit only has effect for the asynchronous operation. Write this bit to zero when using synchronous operation.

Writing this bit to one will reduce the divisor of the baud rate divider from 16 to 8 effectively doubling the transfer rate for asynchronous communication.
- Bit 0 – MPCM: Multi-processor Communication Mode**

This bit enables the Multi-processor Communication mode. When the MPCM bit is written to one, all the incoming frames received by the USART receiver that do not contain address information will be ignored. The transmitter is unaffected by the MPCM setting. For more detailed information see “Multi-processor Communication Mode” on page 157.

รูปที่ 1.14 แสดงรายละเอียดการใช้งานบิตต่าง ๆ ของรีจิสเตอร์ UCSRA (ต่อ)

UCSRB คือรีจิสเตอร์ที่ทำหน้าที่ควบคุมการทำงานของพอร์ต UART ซึ่งมีรายละเอียดดังรูปที่ 1.15

Bit	7	6	5	4	3	2	1	0	
	RXCIE	TXCIE	UDRIE	RXEN	TXEN	UCSZ2	RXB8	TXB8	UCSRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- Bit 7 – RXCIE: RX Complete Interrupt Enable**

Writing this bit to one enables interrupt on the RXC Flag. A USART Receive Complete Interrupt will be generated only if the RXCIE bit is written to one, the Global Interrupt Flag in SREG is written to one and the RXC bit in UCSRA is set.
- Bit 6 – TXCIE: TX Complete Interrupt Enable**


Writing this bit to one enables interrupt on the TXC Flag. A USART Transmit Complete Interrupt will be generated only if the TXCIE bit is written to one, the Global Interrupt Flag in SREG is written to one and the TXC bit in UCSRA is set.
- Bit 5 – UDRIE: USART Data Register Empty Interrupt Enable**

Writing this bit to one enables interrupt on the UDRE Flag. A Data Register Empty Interrupt will be generated only if the UDRIE bit is written to one, the Global Interrupt Flag in SREG is written to one and the UDRE bit in UCSRA is set.
- Bit 4 – RXEN: Receiver Enable**

Writing this bit to one enables the USART Receiver. The Receiver will override normal port operation for the RxD pin when enabled. Disabling the Receiver will flush the receive buffer invalidating the FE, DOR, and PE Flags.
- Bit 3 – TXEN: Transmitter Enable**

Writing this bit to one enables the USART Transmitter. The Transmitter will override normal port operation for the TxD pin when enabled. The disabling of the Transmitter (writing TXEN to zero) will not become effective until ongoing and pending transmissions are completed, i.e., when the transmit Shift Register and transmit Buffer Register

รูปที่ 1.15 แสดงรายละเอียดการใช้งานบิตต่าง ๆ ของรีจิสเตอร์ UCSRB

	ใบเนื้อหา		หน้าที่ 17
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004		หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ		

ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire

do not contain data to be transmitted. When disabled, the transmitter will no longer override the TxD port.

- **Bit 2 – UCSZ2: Character Size**

The UCSZ2 bits combined with the UCSZ1:0 bit in UCSRC sets the number of data bits (Character Size) in a frame the receiver and transmitter use.

- **Bit 1 – RXB8: Receive Data Bit 8**

RXB8 is the ninth data bit of the received character when operating with serial frames with nine data bits. Must be read before reading the low bits from UDR.

- **Bit 0 – TXB8: Transmit Data Bit 8**

TXB8 is the ninth data bit in the character to be transmitted when operating with serial frames with nine data bits. Must be written before writing the low bits to UDR.

รูปที่ 1.16 แสดงรายละเอียดการใช้งานบิตต่าง ๆ ของรีจิสเตอร์ UCSRB (ต่อ)

UCSRC คือรีจิสเตอร์ที่ทำหน้าที่ควบคุมการทำงานของพอร์ต UART ซึ่งมีรายละเอียดดังรูปที่ 1.17

Bit	7	6	5	4	3	2	1	0	
	URSEL	UMSEL	UPM1	UPM0	USBS	UCSZ1	UCSZ0	UCPOL	UCSRC
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	1	0	0	0	0	1	1	0	

The UCSRC Register shares the same I/O location as the UBRRH Register. See the “Accessing UBRRH/ UCSRC Registers” on page 158 section which describes how to access this register.

- **Bit 7 – URSEL: Register Select**

This bit selects between accessing the UCSRC or the UBRRH Register. It is read as one when reading UCSRC. The URSEL must be one when writing the UCSRC.

- **Bit 6 – UMSEL: USART Mode Select**

This bit selects between Asynchronous and Synchronous mode of operation.

Table 63. UMSEL Bit Settings

UMSEL	Mode
0	Asynchronous Operation
1	Synchronous Operation


- **Bit 5:4 – UPM1:0: Parity Mode**

These bits enable and set type of parity generation and check. If enabled, the transmitter will automatically generate and send the parity of the transmitted data bits within each frame. The Receiver will generate a parity value for the incoming data and compare it to the UPM0 setting. If a mismatch is detected, the PE Flag in UCSRA will be set.

Table 64. UPM Bits Settings

UPM1	UPM0	Parity Mode
0	0	Disabled
0	1	Reserved
1	0	Enabled, Even Parity
1	1	Enabled, Odd Parity

รูปที่ 1.17 แสดงรายละเอียดการใช้งานบิตต่าง ๆ ของรีจิสเตอร์ UCSRC

	ใบเนื้อหา		หน้าที่ 18
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004		หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ		

ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire

• Bit 3 – USBS: Stop Bit Select

This bit selects the number of Stop Bits to be inserted by the Transmitter. The Receiver ignores this setting.

Table 65. USBS Bit Settings

USBS	Stop Bit(s)
0	1-bit
1	2-bit

• Bit 2:1 – UCSZ1:0: Character Size

The UCSZ1:0 bits combined with the UCSZ2 bit in UCSRB sets the number of data bits (Character Size) in a frame the Receiver and Transmitter use.

Table 66. UCSZ Bits Settings

UCSZ2	UCSZ1	UCSZ0	Character Size
0	0	0	5-bit
0	0	1	6-bit
0	1	0	7-bit
0	1	1	8-bit
1	0	0	Reserved
1	0	1	Reserved
1	1	0	Reserved
1	1	1	9-bit

• Bit 0 – UCPOL: Clock Polarity

This bit is used for Synchronous mode only. Write this bit to zero when Asynchronous mode is used. The UCPOL bit sets the relationship between data output change and data input sample, and the synchronous clock (XCK).


Table 67. UCPOL Bit Settings

UCPOL	Transmitted Data Changed (Output of TxD Pin)	Received Data Sampled (Input on RxD Pin)
0	Rising XCK Edge	Falling XCK Edge
1	Falling XCK Edge	Rising XCK Edge

รูปที่ 1.18 แสดงรายละเอียดการใช้งานบิตต่าง ๆ ของรีจิสเตอร์ UCSRC (ต่อ)

การกำหนด Baud rate

การกำหนด Baud rate ของไมโครคอนโทรลเลอร์ ATMEGA32 เพื่อกำหนดอัตราความเร็วในการติดต่อสื่อสารด้วยพอร์ต UART สามารถกระทำได้โดยกำหนดที่รีจิสเตอร์ UBRRH และ UBRL ซึ่งการกำหนดให้แก่รีจิสเตอร์ทั้งสองเพื่อกำหนดอัตราความเร็วในการติดต่อสื่อสารของพอร์ต UART สามารถกระทำได้ตามรูปที่ 1.19 ดังนี้

	ใบเนื้อหา		หน้าที่ 19
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004		หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ		

ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire

Operating Mode	Equation for Calculating Baud Rate ⁽¹⁾	Equation for Calculating UBRR Value
Asynchronous Normal Mode (U2X = 0)	$BAUD = \frac{f_{osc}}{16(UBRR + 1)}$	$UBRR = \frac{f_{osc}}{16BAUD} - 1$
Asynchronous Double Speed Mode (U2X = 1)	$BAUD = \frac{f_{osc}}{8(UBRR + 1)}$	$UBRR = \frac{f_{osc}}{8BAUD} - 1$
Synchronous Master Mode	$BAUD = \frac{f_{osc}}{2(UBRR + 1)}$	$UBRR = \frac{f_{osc}}{2BAUD} - 1$

รูปที่ 1.19 แสดงการคำนวณหาค่าของรีจิสเตอร์ UBRR เพื่อกำหนดอัตราบอดเรต

Bit	15	14	13	12	11	10	9	8	
	URSEL	-	-	-	UBRR[11:8]				UBRRH
	UBRR[7:0]								UBRRL
	7	6	5	4	3	2	1	0	
Read/Write	R/W	R	R	R	R/W	R/W	R/W	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

The UBRRH Register shares the same I/O location as the UCSRC Register. See the "Accessing UBRRH/ UCSRC Registers" on page 158 section which describes how to access this register.

- **Bit 15 – URSEL: Register Select**

This bit selects between accessing the UBRRH or the UCSRC Register. It is read as zero when reading UBRRH. The URSEL must be zero when writing the UBRRH.


- **Bit 14:12 – Reserved Bits**

These bits are reserved for future use. For compatibility with future devices, these bit must be written to zero when UBRRH is written.


- **Bit 11:0 – UBRR11:0: USART Baud Rate Register**

This is a 12-bit register which contains the USART baud rate. The UBRRH contains the four most significant bits, and the UBRRL contains the 8 least significant bits of the USART baud rate. Ongoing transmissions by the transmitter and receiver will be corrupted if the baud rate is changed. Writing UBRRL will trigger an immediate update of the baud rate prescaler.


รูปที่ 1.20 แสดงรายละเอียดการใช้งานรีจิสเตอร์ UBRR

	ใบเนื้อหา	หน้าที่ 20
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004	หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ	


ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire
<p>5. การใช้เขียนฟังก์ชันภาษาซีเพื่อควบคุมการรับส่งข้อมูลอนุกรมในรูปแบบ UART ของไมโครคอนโทรลเลอร์</p> <p>การใช้เขียนฟังก์ชันภาษาซีเพื่อควบคุมการรับส่งข้อมูลอนุกรมในรูปแบบ UART ของไมโครคอนโทรลเลอร์ นั้นจะต้องประกอบไปด้วย 3 ฟังก์ชันหลักได้แก่ ฟังก์ชันกำหนดค่าเริ่มต้นของโมดูล UART , ฟังก์ชันการรับค่าจากพอร์ต UART และฟังก์ชันการส่งค่าจากพอร์ต UART ซึ่งไมโครคอนโทรลเลอร์ต่าง ๆ สามารถเขียนได้ดังนี้</p> <p>5.1 การใช้เขียนฟังก์ชันภาษาซีเพื่อควบคุมการรับส่งข้อมูลอนุกรมในรูปแบบ UART ของ AT89C51ED2</p> <p>5.1.1 การเขียนฟังก์ชันกำหนดค่าเริ่มต้นของโมดูล UART ของ AT89C51ED2</p> <pre>void uart_init(int baud){ //Use XTAL 11.0592MHz SCON = 0x50; //RX Enable Data 8 bit Baud rate variable TMOD = (TMOD & 0x0f) 0x20; // Timer1 mode 8 Auto reload TH1 = 256 - ((XTAL/384)/baud); TL1 = TH1; TR1 = 1; }</pre> <p>5.1.2 ฟังก์ชันการรับค่าจากพอร์ต UART ของ AT89C51ED2</p> <pre>unsigned char uart_getc(){ while(RI == 0); RI = 0; return SBUF; }</pre> <p>5.1.3 ฟังก์ชันการส่งค่าจากพอร์ต UART ของ AT89C51ED2</p> <pre>void uart_putc(unsigned char dat){ //Send 1 Character SBUF = dat; while(TI == 0); TI = 0; }</pre> <pre>void uart_puts(unsigned char *str){ //Send String while(*str != '\0') uart_putc(*str++); }</pre>

	ใบเนื้อหา	หน้าที่ 21
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004	หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ	


ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire
<p>5.2 การใช้เขียนฟังก์ชันภาษาซีเพื่อควบคุมการรับส่งข้อมูลอนุกรมในรูปแบบ UART ของ PIC16F887</p> <p>5.2.1 การเขียนฟังก์ชันกำหนดค่าเริ่มต้นของโมดูล UART ของ PIC16F887</p> <pre>void uart_init(unsigned int baud) { //Xtal = 20MHz unsigned int SpeedUart; SpeedUart = ((_XTAL_FREQ/ baud)/16) - 1; SPBRG = SpeedUart; SPBRGH = SpeedUart >> 8; SPEN = 1; //Serial Port Enable & Continuous Enable CREN = 1; TXEN = 1; //Tx Enable & High Speed mode BRGH = 1; }</pre> <p>5.2.2 ฟังก์ชันการรับค่าจากพอร์ต UART ของ PIC16F887</p> <pre>unsigned char uart_getc(){ unsigned char dat; while (!RCIF); dat = RCREG; return dat; }</pre> <p>5.2.3 ฟังก์ชันการส่งค่าจากพอร์ต UART ของ PIC16F887</p> <pre>void uart_putc(unsigned char c){ //Send 1 Character while(!TRMT); TXREG = c; } void uart_puts (char *s) { //Send String while (*s) { uart_putc(*s); s++; } }</pre>


	ใบเนื้อหา	หน้าที่ 22
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004	หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ	

ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire
<p>5.3 การใช้เขียนฟังก์ชันภาษาซีเพื่อควบคุมการรับส่งข้อมูลอนุกรมในรูปแบบ UART ของ ATMEGA32</p> <p>5.3.1 การเขียนฟังก์ชันกำหนดค่าเริ่มต้นของโมดูล UART ของ ATMEGA32</p> <pre>void uart_init(unsigned int baud){ //Xtal = 16MHz uint32_t SpeedUart; SpeedUart = (uint32_t) (F_CPU/((uint32_t)16*baud)) - 1; UBRRL = SpeedUart; UBRRH = SpeedUart >> 8; UCSRB = (1<<RXEN) (1<<TXEN); UCSRC = (1<<URSEL) (3<<UCSZ0); }</pre> <p>5.3.2 ฟังก์ชันการรับค่าจากพอร์ต UART ของ ATMEGA32</p> <pre>unsigned char uart_getc(){ unsigned char dat; while (!(UCSRA & (1<<RXC))); dat = UDR; return dat; }</pre> <p>5.3.3 ฟังก์ชันการส่งค่าจากพอร์ต UART ของ ATMEGA32</p> <pre>void uart_putc(unsigned char c){ //Send 1 Character while(!(UCSRA & (1 << UDRE))); UDR = c; } void uart_puts (char *s){ //Send String while (*s) { uart_putc(*s); s++; } }</pre>

	ใบเนื้อหา	หน้าที่ 23
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004	หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ	


ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire
6. การใช้เขียนฟังก์ชันภาษาซีเพื่อควบคุมการรับส่งข้อมูลอนุกรมในรูปแบบ One-Wire ของไมโครคอนโทรลเลอร์
6.1 การใช้เขียนฟังก์ชันภาษาซีเพื่อควบคุมการรับส่งข้อมูลอนุกรมในรูปแบบ One-Wire ของ AT89C51ED2
<p>การเขียนฟังก์ชันภาษาซีเพื่อควบคุมการรับส่งข้อมูลอนุกรมในรูปแบบ One-Wire ของ AT89C51ED2 นั้นจะอ้างอิงรูปแบบการสื่อสารของอุปกรณ์เซ็นเซอร์วัดอุณหภูมิ DS18S20 เป็นหลัก ซึ่งรูปแบบการเขียนโปรแกรมเพื่อติดต่อสื่อสารจะอ้างอิงตามเนื้อหาหัวข้อที่ 3 โดยตัวอย่างโปรแกรมที่นำเสนอจะใช้สำหรับการสร้าง Library นามสกุล .c เพื่อนำไปรวมเข้ากับโปรแกรมหลักด้วยวิธีการ include เพื่อนำไปใช้งานต่อไป และสำหรับไมโครคอนโทรลเลอร์ AT89C51ED2 นั้น code ตัวอย่างเมื่อนำไปใช้งานจริงอาจจะสามารถทำงานได้ แต่อาจจะใช้สำหรับการจำลองการทำงานด้วยโปรแกรม Proteus ไม่ได้ แต่ถ้าจำลองการทำงานด้วยโปรแกรม Proteus ได้ แต่อาจจะใช้งานจริงไม่ได้ เป็นผลเนื่องมาจากค่า XTAL ของไมโครคอนโทรลเลอร์ที่มีความเร็วในการทำงานที่ต่ำ ส่วนการแก้ไขให้ code สามารถทำงานได้หรือไม่ได้ กระทำได้โดยการปรับค่าของฟังก์ชัน delayOneWire และการหน่วงเวลาด้วยวิธีการวนลูป</p> <p>ตัวอย่างโปรแกรม Library สำหรับการเขียนโปรแกรมเพื่ออ่านค่าอุณหภูมิจาก DS18S20</p> <pre>//Xtal 22.1184MHz #include <intrins.h> sbit onewire = P3^2; // Bit data 1-wire bus void delayOneWire(unsigned int usecond){ while(usecond--); } unsigned char ow_reset(){ unsigned char presence,i; onewire = 0; // pull DQ line low delayOneWire(480); // leave it low for 480us onewire = 1; // allow line to return high delayOneWire(10); // wait for presence i = 60; do{ presence = onewire; // get presence signal i--; // wait for end of timeslot }while((i>0)&&(presence>0)); i = 120 - i; delayOneWire(i); return presence; // presence signal returned 0=presence, 1 = no part }</pre>

	ใบเนื้อหา		หน้าที่ 24
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004		หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ		
ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire			
<pre>bit read_bit (void){ unsigned char i; bit dat; // Bit for keep data onewire = 0; // Clear bit 1-wire _nop_(); // Delay _nop_(); // Delay _nop_(); // Delay _nop_(); // Delay onewire = 1; // Set bit 1-wire _nop_(); // Delay _nop_(); // Delay _nop_(); // Delay _nop_(); // Delay _nop_(); // Delay _nop_(); // Delay _nop_(); // Delay dat = onewire; // Read data bit keep in dat for(i=0;i<16;i++) _nop_();// Delay 64 microsec return dat; // Return dat } unsigned char ReadByte(){ unsigned char i,j,dat; // For counter and keep data dat = 0; // Clear data for(i=0;i<8;i++){ // For loop Read data 1 byte(8 time) j = read_bit(); // Keep data bit to j dat = (j<<7) (dat>>1); // Shift left j 7 time OR } return dat; // Return dat }</pre>			

	ใบเนื้อหา	หน้าที่ 25
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004	หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ	

ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire
--

```
void WriteByte(unsigned char com){
    unsigned char j,i;          // For counter
    bit send;                   // Bit send data 1-wire
    for(j=0;j<8;j++) {          // For loop Write data 1 byte(8 time)
        send = com & 0x01; // LSB bit keep to send
        com = com>>1;        // Shift Right 1 time
        if(send) {            // Check send = 1?
            onewire = 0;      // Clear bit 1-wire
            _nop_();           // Delay delay 4 microsec
            _nop_();           // Delay
            _nop_();           // Delay
            _nop_();           // Delay
            _nop_();           // Delay
            _nop_();           // Delay
            _nop_();           // Delay
            _nop_();           // Delay
            onewire = 1;       // Set bit 1-wire
            for(i=0;i<16;i++) _nop_(); //delay 64 microsec
        }else{
            onewire = 0;      // Clear bit 1-wire
            for(i=0;i<16;i++) _nop_(); // delay 64 microsec
            onewire = 1;       // Set bit 1-wire
            _nop_();           // Delay delay 4 microsec
            _nop_();           // Delay
            _nop_();           // Delay
            _nop_();           // Delay
            _nop_();           // Delay
            _nop_();           // Delay
            _nop_();           // Delay
            _nop_();           // Delay
        }
    }
}
```

	ใบเนื้อหา	หน้าที่ 26
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004	หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ	

ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire

6.2 การใช้เขียนฟังก์ชันภาษาซีเพื่อควบคุมการรับส่งข้อมูลอนุกรมในรูปแบบ One-Wire ของ PIC16F887


การเขียนฟังก์ชันภาษาซีเพื่อควบคุมการรับส่งข้อมูลอนุกรมในรูปแบบ One-Wire ของ PIC16F887 นั้นจะอ้างอิงรูปแบบการสื่อสารของอุปกรณ์เซ็นเซอร์วัดอุณหภูมิ DS18S20 เป็นหลัก ซึ่งรูปแบบการเขียนโปรแกรมเพื่อติดต่อสื่อสารจะอ้างอิงตามเนื้อหาหัวข้อที่ 3 โดยตัวอย่างโปรแกรมที่นำเสนอจะใช้สำหรับการสร้าง Library นามสกุล .c เพื่อนำไปรวมเข้ากับโปรแกรมหลักด้วยวิธีการ include เพื่อนำไปใช้งานต่อไป


ตัวอย่างโปรแกรม Library สำหรับการเขียนโปรแกรมเพื่ออ่านค่าอุณหภูมิจาก DS18S20

```
#define onewire RD0          // Bit data 1-wire bus
#define TRISD1wire TRISD0

void delay_us(int us){
    for(;us>0;us--)
        __delay_us(1);
}

char ow_reset(){
    char DQ;
    int i;
    TRISD1wire = 0;
    onewire = 0;          // Clear bit 1-wire
    __delay_us(480);      // Delay 480 microsec
    onewire = 1;          // Set bit 1-wire
    __delay_us(16);       // Delay 16 microsec
    TRISD1wire = 1;
    i = 240;
    do{
        DQ = onewire;     // get presence signal
        i--;
        __delay_us(1);
    }while((i>0)&&(DQ>0));
    TRISD1wire = 0;
    onewire = 1;
    i = 480 - i;
    delay_us(i);          // wait for end of timeslot
    return DQ;            // returned 0 = presence, 1 = no part
}
```

	ใบเนื้อหา	หน้าที่ 27
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004	หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ	
ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire		
<pre>char read_bit(void){ TRISD1wire = 0; onewire = 0; // pull onewire low to start timeslot __delay_us(10); onewire = 1; // then return high __delay_us(5); TRISD1wire = 1; return onewire; // return value of DQ line } void write_bit(unsigned char DQ){ TRISD1wire = 0; onewire = 0; // pull DQ low to start timeslot __delay_us(10); if(DQ==1) onewire = 1; // return DQ high if write 1 __delay_us(65); // hold value for remainder of timeslot onewire = 1; __delay_us(15); } unsigned char ReadByte(void){ unsigned char i; unsigned char value = 0; for(i=0;i<8;i++){ if(read_bit()) value =0x01<<i; // reads byte in, one byte at a time and then shifts it left __delay_us(60); // wait for rest of timeslot TRISD1wire = 0; onewire = 1; __delay_us(15); } return value; }</pre>		

	ใบเนื้อหา	หน้าที่ 28
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004	หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ	

ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire

```
void WriteByte(char val){
    unsigned char i;
    unsigned char temp;
    TRISD1wire = 0;
    for (i=0; i<8; i++){
        temp = val>>i;
        temp &= 0x01;
        write_bit(temp);
    }
}
```

6.3 การใช้เขียนฟังก์ชันภาษาซีเพื่อควบคุมการรับส่งข้อมูลอนุกรมในรูปแบบ One-Wire ของ ATMEGA32


การเขียนฟังก์ชันภาษาซีเพื่อควบคุมการรับส่งข้อมูลอนุกรมในรูปแบบ One-Wire ของ ATMEGA32 นั้นจะอ้างอิงรูปแบบการสื่อสารของอุปกรณ์เซ็นเซอร์วัดอุณหภูมิ DS18S20 เป็นหลัก ซึ่งรูปแบบการเขียนโปรแกรมเพื่อติดต่อสื่อสารจะอ้างอิงตามเนื้อหาหัวข้อที่ 3 โดยตัวอย่างโปรแกรมที่นำเสนอจะใช้สำหรับการสร้าง Library นามสกุล .c เพื่อนำไปรวมเข้ากับโปรแกรมหลักด้วยวิธีการ include เพื่อนำไปใช้งานต่อไป


ตัวอย่างโปรแกรม Library สำหรับการเขียนโปรแกรมเพื่ออ่านค่าอุณหภูมิจาก DS18S20


```
#define BIT_DQ 6                // 1-Wire Interface (DS18B20 Temp. Sensor)

#define SET_IN_DQ    (DDRD &= ~(1<<BIT_DQ))
#define SET_OUT_DQ   (DDRD |= (1<<BIT_DQ))
#define OUT_HIGH_DQ  (PORTD |= (1<<BIT_DQ))
#define OUT_LOW_DQ   (PORTD &= ~(1<<BIT_DQ))
#define IN_DQ        (PIND & (1<<BIT_DQ))

void delay_us(unsigned int time_us)
{
    for (;time_us>0; time_us--)
        _delay_us(1);
}
```


	ใบเนื้อหา		หน้าที่ 29
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004		หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ		
ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire			
<pre>unsigned char ow_reset(void){ unsigned char presence; unsigned int i; SET_OUT_DQ; // Set output port delay_us(1); OUT_LOW_DQ; // pull DQ line low delay_us(480); // leave it low for 480us OUT_HIGH_DQ; // allow line to return high delay_us(15); // wait for presence SET_IN_DQ; // Set input port i = 120; do{ presence = IN_DQ; // get presence signal _delay_us(1); // wait for end of timeslot i--; }while((i > 0)&&(presence > 0)); i = 480 - i; delay_us(i); return(presence); // presence signal returned 0=presence, 1 = no part } unsigned char read_bit(void){ SET_OUT_DQ; // Set output port OUT_LOW_DQ; // pull DQ low to start timeslot delay_us(10); // delay_us 15us from start of timeslot OUT_HIGH_DQ; // then return high SET_IN_DQ; // Set input port delay_us(5); return(IN_DQ); } void write_bit(char bitval){ OUT_LOW_DQ; // pull DQ low to start timeslot delay_us(10); if(bitval == 1) // return DQ high if wirte 1 OUT_HIGH_DQ; delay_us(65); // hold value for remainder of timeslot (100us) OUT_HIGH_DQ; delay_us(15); }</pre>			

	แบบฝึกหัด	หน้าที่ 1
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004	หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ	
ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire		
<p><u>คำสั่ง</u> จงตอบคำถามต่อไปนี้ให้ถูกต้อง</p> <ol style="list-style-type: none"> 1. จงอธิบายรูปแบบการสื่อสารแบบ UART 2. จงอธิบายรูปแบบการสื่อสารแบบ One-Wire 3. การสื่อสารในรูปแบบ One-Wire ถูกคิดค้นโดยบริษัทใด 4. การสื่อสารแบบ UART ของไมโครคอนโทรลเลอร์ถ้าต้องการสื่อสารกับคอมพิวเตอร์ต้องทำอย่างไร 5. การสื่อสารแบบ UART ถ้าต้องการต่อแบบ Network จะต้องทำอย่างไร 6. จงอธิบายวิธีการคำนวณค่า TH1 ของ AT89C51ED2 เมื่อต้องการติดต่อสื่อสารด้วยพอร์ต UART 7. จงอธิบายวิธีการคำนวณค่า SPBRGH: SPBRG ของ PIC16F887 เมื่อต้องการติดต่อสื่อสารด้วยพอร์ต UART 		

	แบบฝึกหัด	หน้าที่ 2
	ชื่อวิชา ดิจิทัลและไมโครคอนโทรลเลอร์ รหัสวิชา 30127-2004	หน่วยที่ 8
	ชื่อหน่วย การรับส่งข้อมูลอนุกรมแบบต่าง ๆ	
ชื่อเรื่อง การรับส่งข้อมูลอนุกรมในรูปแบบ UART และ One-Wire		
<p>8. จงอธิบายวิธีการคำนวณหาค่า UBRH: UBRRL ของ ATMEGA32 เมื่อต้องการติดต่อสื่อสารด้วยพอร์ต UART</p> <p>.....</p> <p>.....</p> <p>.....</p> <p>9. จงอธิบายรูปแบบของไทม์สล็อตของการเขียนข้อมูลของการสื่อสารแบบ One-Wire ของมาสเตอร์ไปอุปกรณ์</p> <p>.....</p> <p>.....</p> <p>.....</p> <p>10. จงอธิบายรูปแบบของไทม์สล็อตของการอ่านข้อมูลของการสื่อสารแบบ One-Wire จากอุปกรณ์มาที่มาสเตอร์</p> <p>.....</p> <p>.....</p> <p>.....</p>		