Arduino กับการสื่อสารแบบอนุกรม

Serial Communication

รศ.ณรงค์ บวบทอง

หัวข้อ

- บทนำ
  - รูปแบบของการสื่อสาร
  - รูปแบบการสื่อสารแบบอนุกรม
  - การสื่อสารข้อมูลแบบซิงโครนัส (Synchronous)
  - การสื่อสารข้อมูลแบบอะซิงโครนัส (Asynchronous)
  - การสื่อสารแบบข้อมูลแบบไอโซโครนัส (Isochronous Transmission)
- การสื่อสารข้อมูลแบบอนุกรมของไมโครคอนโทรลเลอร์ AVR
  - บล็อกไดอะแกรมอย่างง่ายๆของพอร์ทอนุกรม ATMega168 - USART
  - รีจิสเตอร์ที่เกี่ยวข้องกับการทำงานในโหมด USARTRelated Registers
  - ตัวอย่างโปรแกรม Serial Output โดยใช้ AVR Studio
  - Arduino : Serial
  - มาตราฐาน RS-232

บทนำ

การสื่อสารข้อมูลแบบอนุกรม เป็นการรับส่งข้อมูลทีละบิต แทนที่จะทำการรับส่งข้อมูลพร้อมกันทุกบิตในเวลาเดียวกัน ข้อดีของการสื่อสารแบบนี้คือ ใช้จำนวนสายในการสื่อสารน้อย สามารถรับส่งได้ในระยะ ทางที่ไกล ๆ แต่ก็มีข้อเสียในด้านเวลา เพราะต้องใช้เวลาในการสื่อสารมาก เมื่อเทียบกับการสื่อสารแบบขนาน อีกทั้งโอกาสเกิดการผิดพลาดของข้อมูลก็สูงกว่าแบบขนาน

รูปแบบของการสื่อสาร

รูปแบบของการสื่อสาร แบ่งได้ 3 แบบ คือ

1. แบบซิมเพล็กซ์ (Simplex) เป็นการสื่อสารทางเดียว
2. แบบฮาล์ฟดูเพล็กซ์(Half-duplex) เป็นการสื่อสารได้ทั้งสองทาง แต่จะต้องผลัดกันรับ-ส่ง
3. แบบฟูลดูเพล็กซ์(Full-duplex) เป็นการสื่อสารได้ทั้งสองทางและทำได้ในเวลาเดียวกัน

รูปแบบการสื่อสารข้อมูลแบบอนุกรม

การติดต่อแบบอนุกรมเมื่อแบ่งตามลักษณะของการส่งข้อมูล แบ่งได้ 2 แบบ คือ

1. การสื่อสารแบบซิงโครนัส (Synchronous)
2. การสื่อสารแบบอะซิงโครนัส(Asynchronous)
3. การส่งข้อมูลแบบไอโซโครนัส (Isochronous Transmission)

การสื่อสารข้อมูลแบบอะซิงโครนัส (Asynchronous)

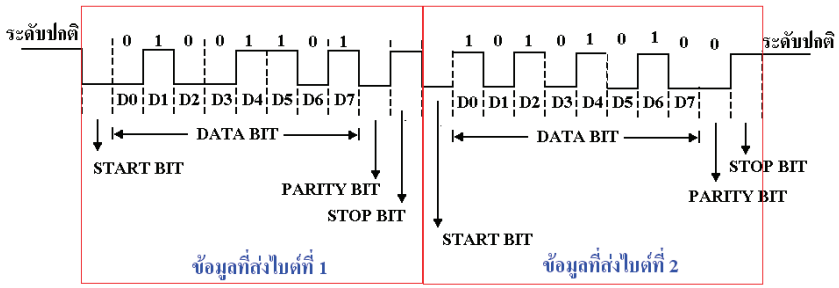
การสื่อสารแบบนี้ใช้มากในเครื่องไมโครคอมพิวเตอร์พีซี รูปแบบการสื่อสารจะเป็นการรับและส่งข้อมูลครั้งละ 1 ไบต์

ตัวอย่างโปรโตคอลอะซิงโครนัส Serial Commincation - Example Protocols

- Morse code
- RS-232 - Recommended Standard 232
- RS422, RS-423, RS-485
- I2C - Inter-Integrated Circuit
- SPI - Serial Peripheral Interface
- USB - Universal Serial Bus
- Firewire
- Ethernet
- Serial ATA - Serial Advanced TEchnology Attachment
- Serial Attach SCSI - Serial Attached Small Computer System
- Interface
- SONET - Synchronous Optical Network
- PCI Express - Peripheral Component Interconnect Express

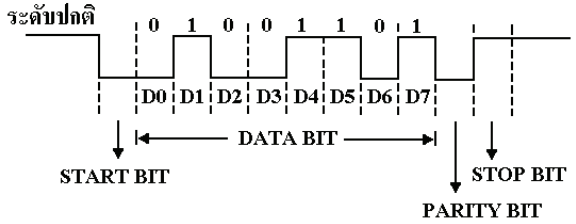
การสื่อสารแบบอะซิงโครนัส (Asynchronous)

การสื่อสารแบบนี้ใช้มากในเครื่องไมโครคอมพิวเตอร์พีซี รูปแบบการส่งข้อมูลจะเป็นการส่งครั้งละ 1 ไบต์ โดยมีรูปแบบดังนี้



การสื่อสารแบบอะซิงโครนัส (Asynchronous) (ต่อ)

ความหมายของบิต



Start Bit บอกจุดเริ่มต้นข้อมูล มีขนาด 1 บิต

Data Bit ค่าข้อมูลมีได้ 5 ถึง 8 บิต

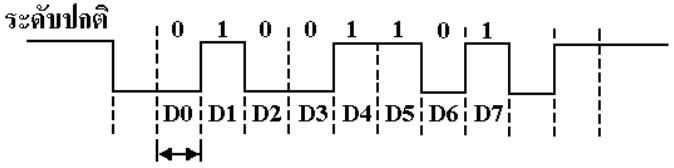
Parity Bit บิตสำหรับใช้ตรวจสอบความผิดพลาดของข้อมูล มีได้ 0 ถึง 1 บิต

Stop Bit บิตใช้บอกจุดสิ้นสุดข้อมูล มีได้ 1 1.5 และ 2 บิต

ความเร็วในการสื่อสาร

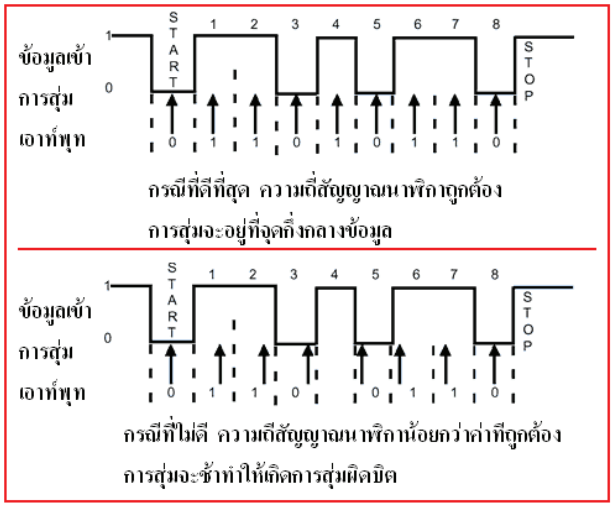
ความเร็วในการสื่อสาร หมายถึงจำนวนบิตที่ใช้รับส่งข้อมูลต่อวินาที โดยปกติจะมีค่าเท่ากับ 110 150 300 1200 2400 4800 9600 และ 19200 บิตต่อวินาที อัตราความเร็วนี้บางครั้งก็เรียกว่าอัตราบอด (Baud rate) ทั้งตัวส่งและตัวรับต้องกำหนดให้มีความเร็วในการสื่อสารเท่ากัน

ตัวอย่างข้อมูลส่งด้วยความเร็ว 2400 บิตต่อวินาที ดังนั้นแต่ละบิตใช้เวลาส่งเท่ากับ 1/2400 = 416.67 ไมโครวินาที



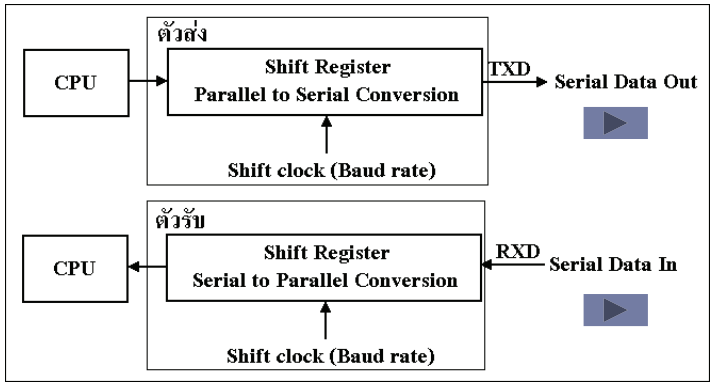
ส่งข้อมูล 1 ไบต์ใช้เวลา = 416.67x10 = 4.167 mS

การสุ่มข้อมูลแบบอะซิงโครนัส



การสุ่มข้อมูลแบบอะซิงโครนัส (Asynchronous)

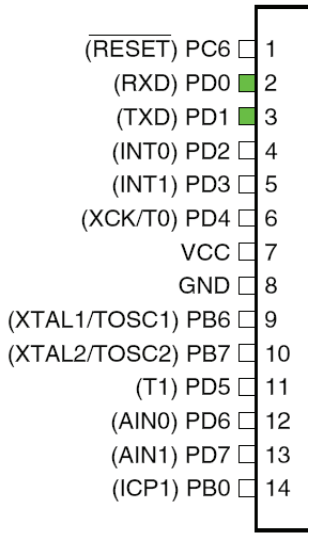
หลักการรับ-ส่งข้อมูลแบบอนุกรม



การสื่อสารข้อมูลแบบอนุกรมของไมโครคอนโทรลเลอร์ AVR

ไมโครคอนโทรลเลอร์ AVR สามารถสื่อสารข้อมูลแบบอนุกรมได้โดยใช้โมดูล USART ((Universal Synchronous and Asynchronous serial Receiver and Transmitter) สำหรับ ATMega 16 ขาพอร์ทอนุกรมกำหนดไว้ที่

PD0	Serial input	RxD
PD1	Serial output	TxD

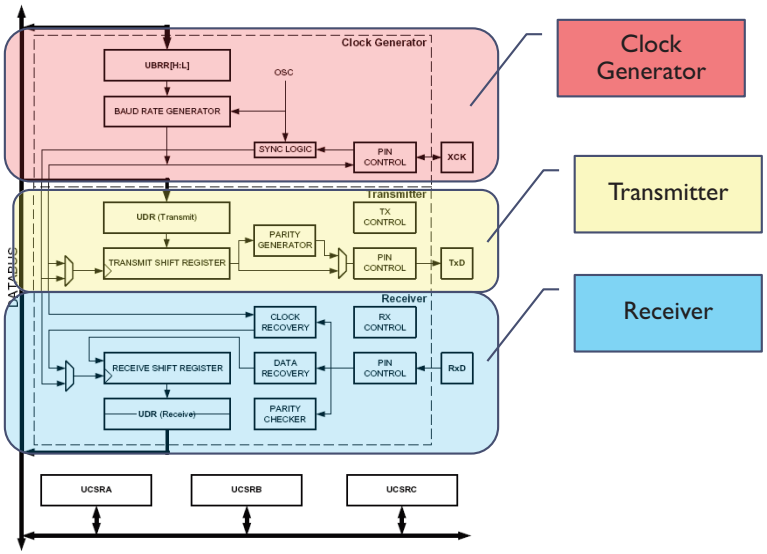


คุณลักษณะของ ATMega168-USART

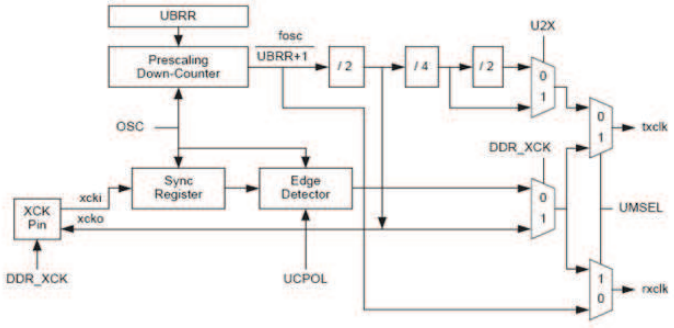
USART - Universal Synchronous Asynchronous Receiver Transmitter.

- ทำงานแบบ Full Duplex (การรับและส่งเป็นอิสระซึ่งกันและกัน)
- ทำงานได้ทั้งแบบ Asynchronous และ Synchronous
- Master or Slave Clocked Synchronous Operation
- High Resolution Baud Rate Generator
- รองรับการรับส่งข้อมูลแบบ 5, 6, 7, 8, or 9 Data Bits และ 1 หรือ 2 Stop Bits
- Odd or Even Parity Generation and Parity Check Supported by Hardware
- Data OverRun Detection
- Framing Error Detection
- Three Separate Interrupts on TX Complete, TX Data Register
- Empty and RX Complete

บล็อกไดอะแกรมอย่างง่ายๆของพอร์ทอนุกรม ATMega168 - USART



Clock Generation Logic, Block Diagram



Signal description:

- txclk Transmitter clock (internal signal).
- rxclk Receiver base clock (internal signal).
- xcki Input from XCK pin (internal signal). Used for synchronous slave operation.
- xcko Clock output to XCK pin (internal signal). Used for synchronous master operation.
- fosc System clock frequency.

การคำนวณหาอัตราบอต (Baud rate)

โหมดการทำงาน	อัตราบอต	ค่ารีจิสเตอร์ UBRR
อะซิงโครนัสปกติ (U2X = 0)	Baud = fosc/(16*(UBRR+1))	UBRR = (fosc/16*Baud)-1
อะซิงโครนัสทวีคูณ (U2X = 1)	Baud = fosc/(8*(UBRR+1))	UBRR = (fosc/8*Baud)-1
มาสเตอร์ซิงโครนัส	Baud = fosc/(2*(UBRR+1))	UBRR = (fosc/2*Baud)-1

UBRRnL and UBRRnH – USART baud rate registers

Bit	15	14	13	12	11	10	9	8	
	–	–	–	–	UBRRn[11:8]				UBRRnH
	UBRRn[7:0]								UBRRnL
	7	6	5	4	3	2	1	0	
Read/write	R	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	

Bit 15:12 – บิตสำรองเพื่อการใช้งาน บิตเหล่านี้ต้องให้เป็น 0

Bit 11:0 – UBRR11:0: USART บิตกำหนดอัตราบอด (baud rate) มีค่าได้ตั้งแต่ 0 - 4095

ตัวอย่างการกำหนดค่าอัตราบอต

$$\text{Error}[\%] = \left(\frac{\text{BaudRate}_{\text{Closest Match}}}{\text{BaudRate}} - 1\right) \bullet 100\%$$

Baud rate (bps)	f_{osc} = 8.0000MHz			
	U2Xn = 0		U2Xn = 1	
	UBRRn	Error	UBRRn	Error
2400	207	0.2%	416	-0.1%
4800	103	0.2%	207	0.2%
9600	51	0.2%	103	0.2%
14.4k	34	-0.8%	68	0.6%
19.2k	25	0.2%	51	0.2%
28.8k	16	2.1%	34	-0.8%
38.4k	12	0.2%	25	0.2%
57.6k	8	-3.5%	16	2.1%
76.8k	6	-7.0%	12	0.2%
115.2k	3	8.5%	8	-3.5%
230.4k	1	8.5%	3	8.5%
250k	1	0.0%	3	0.0%
0.5M	0	0.0%	1	0.0%
1M	–	–	0	0.0%
Max.(1)	0.5Mbps		1Mbps	

รีจิสเตอร์ที่เกี่ยวข้องกับการทำงานในโหมด USART

1. รีจิสเตอร์ UDR (USART I/O Data Register)
2. รีจิสเตอร์ UCSRA (USART Control and Status Register A)
3. รีจิสเตอร์ UCSRB (USART Control and Status Register B)
4. รีจิสเตอร์ UCSRC (USART Control and Status Register C)
5. รีจิสเตอร์ UBRRL และ UBRRH (USART Baud Rate Register)

Arduino : Serial

- if (Serial)

- available()
- begin()
- end()
- find()
- findUntil()
- flush()
- parseFloat()
- parseInt()
- peek()
- print()
- println()
- read()
- readBytes()
- readBytesUntil()
- setTimeout()
- write() begin()

อ้างอิง http://arduino.cc/en/Reference/Serial

ตัวอย่างการส่งข้อมูล โดยใช้ Arduino

```
int analogValue = 0;    // variable to hold the analog value
void setup() {
    Serial.begin(9600);        // open the serial port at 9600 bps:
}

void loop() {
   analogValue = analogRead(0);     // read the analog input on pin 0:

  // print it out in many formats:
  Serial.println(analogValue);       // print as an ASCII-encoded decimal
  Serial.println(analogValue, DEC);  // print as an ASCII-encoded decimal
  Serial.println(analogValue, HEX);  // print as an ASCII-encoded hexadecimal
  Serial.println(analogValue, OCT);  // print as an ASCII-encoded octal
  Serial.println(analogValue, BIN);  // print as an ASCII-encoded binary

  // delay 10 milliseconds before the next reading:
  delay(10);
}
```

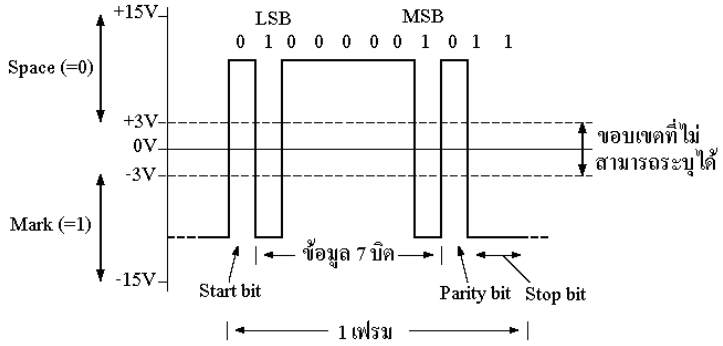

ตัวอย่างการรับข้อมูล โดยใช้ Arduino

```
int incomingByte = 0;    // for incoming serial data

void setup() {
   // opens serial port, sets data rate to 9600 bps
   Serial.begin(9600);
}
```

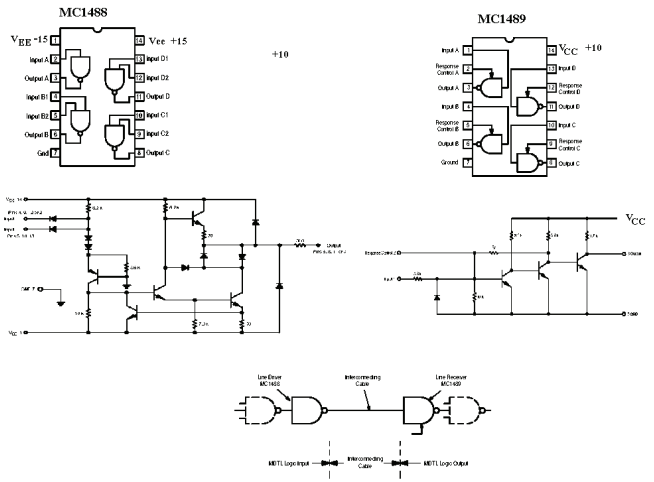

ตัวอย่างการรับข้อมูล โดยใช้ Arduino (ต่อ)

```
void loop() {
   // send data only when you receive data:
   if (Serial.available() > 0) {
           // read the incoming byte:
           incomingByte = Serial.read();

           // say what you got:
           Serial.print("I received: ");
           Serial.println(incomingByte, DEC);
   }
}
```

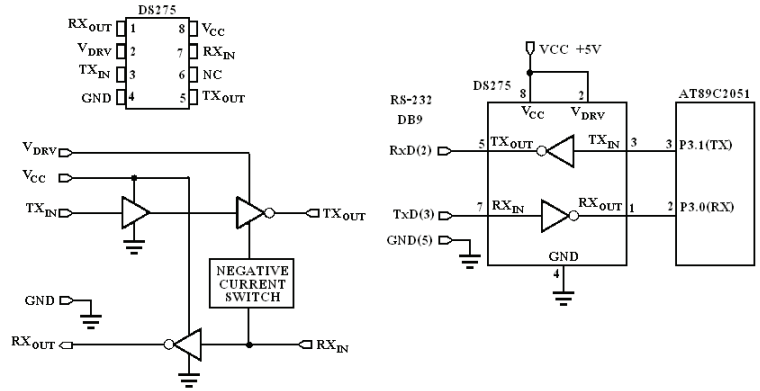
ระดับสัญญาณตามมาตราฐาน RS-232



Line EIA-232D Driver and Line Receivers

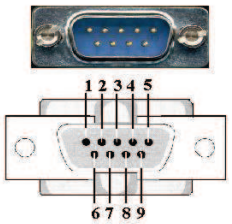


DS275 Line-Powered RS-232 Transceiver Chip



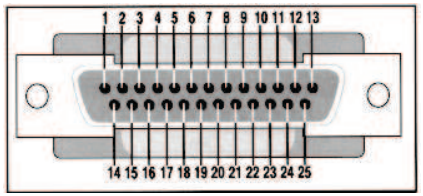
RXOUT	RS–232 Receiver Output	VDRV	Transmit driver +V
TXIN	RS–232 Driver Input	GND	System Ground (0V)
TXOUT	RS–232 Driver Output	NC	No Connection
RXIN	RS–232 Receive Input	VCC	System Logic Supply (+5V)

RS232 Pin Assignments (DB9 PC signal set)



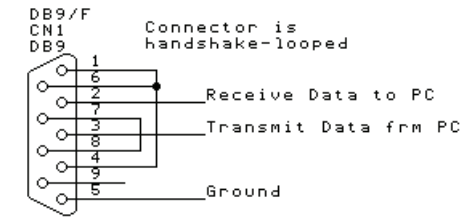
Pin 1 Received Line Signal Detector (Data Carrier Detect)	In
Pin 2 Received Data	In
Pin 3 Transmit Data	Out
Pin 4 Data Terminal Ready	Out
Pin 5 Signal Ground	
Pin 6 Data Set Ready	In
Pin 7 Request To Send	Out
Pin 8 Clear To Send	In
Pin 9 Ring Indicator	

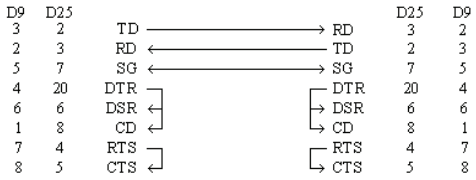
RS232 Pin Assignments (DB25 PC signal set)



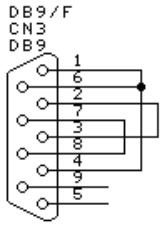
Pin 1 Protective Ground
Pin 2 Transmit Data
Pin 3 Received Data
Pin 4 Request To Send
Pin 5 Clear To Send
Pin 6 Data Set Ready
Pin 7 Signal Ground
Pin 8 Received Line Signal Detector (Data Carrier Detect)
Pin 20 Data Terminal Ready
Pin 22 Ring Indicator

การต่อสายแบบป้อนกลับ





Handshake looping a PC serial connector



RS232 DB9 PC Loopback test plug

ตัวอย่างการต่อสายแบบครบ