บทที่ 6

การประยุกต์ใช้งานกับคอมพิวเตอร์ส่วนบุคคล

การออกแบบลักษณะการใช้งานโปรแกรมของวิทยานิพนธ์ฉบับนี้ ได้ออกแบบการทำงานของโปรแกรมภายใต้ระบบปฏิบัติการลีนุกซ์ และมีการเชื่อมต่อระหว่างคอมพิวเตอร์ส่วนบุคคลกับฮาร์ดแวร์ผ่านทางพอร์ตอนุกรม โดยวิทยานิพนธ์ในบทนี้จะกล่าวถึงคุณสมบัติของระบบปฏิบัติการและการเชื่อมต่อสื่อสารทางพอร์ตอนุกรม

6.1 ระบบปฏิบัติการลีนุกซ์

6.1.1 พื้นฐานของระบบปฏิบัติการลีนุกซ์

ลีนุกซ์เป็นระบบปฏิบัติการ (Operating System) ที่ถูกถอดแบบโครงสร้างมาจากระบบปฏิบัติการยูนิกซ์แต่มีส่วนที่แตกต่างกับระบบปฏิบัติการยูนิกซ์ที่ลีนุกซ์ถูกพัฒนาขึ้นมาตามมาตราฐานของ POSIX (Portable Operating System Interface) ซึ่งมีความสามารถในการทำงานกับเครื่องคอมพิวเตอร์ได้หลายตระกูลแต่ยูนิกซ์ส่วนใหญ่จะมีความจำเพาะกับคอมพิวเตอร์บางรุ่นเท่านั้น เช่น SUN SPARC เป็นต้น อีกทั้งลีนุกซ์ยังอยู่ภายใต้ลิขสิทธิ์แบ GPL (GNU General Public License) ซึ่งหมายถึงการอนุญาตให้นำรหัสต้นฉบับมาทำการแก้ไข ปรับปรุงและแจกจ่ายได้ตามความต้องการอย่างเป็นอิสระแต่ซอฟต์แวร์นั้นจะต้องยังคงเป็นลิขสิทธิ์แบบ GPL อยู่[19]

ระบบปฏิบัติการลีนุกซ์ถูกสร้างขึ้นโดยนาย Linus Torvalds ขณะที่ยังเป็นนักศึกษาอยู่ที่มหาวิทยาลัย Helsinki ประเทศฟินแลนด์ โดยเริ่มแรกได้สร้างตามแบบของระบบปฏิบัติการมินิกซ์ (Minix) ก่อน แล้วพัฒนาจนสามารถใช้งานกับเครื่องคอมพิวเตอร์ตระกูล Intel ได้ด้วย จากนั้นได้มีการแจกจ่ายรหัสต้นฉบับกันทางอินเตอร์เน็ตทำให้มีผู้สนใจกันมากขึ้นและร่วมพัฒนากันอย่างแพร่หลายทั้งในภาคธุรกิจ และในมหาวิทยาลัยต่างๆ ในปัจจุบันนี้เราสามารถหาระบบปฏิบัติการลีนุกซ์จากค่ายต่างๆมาใช้ได้สะดวกมากขึ้นเช่น Slackware, Red Hat, Debian, SuSE, Mandrake และ Ziif เป็นต้น แต่ยังคงอยู่ภายใต้ GPL เหมือนเดิม

6.1.2 คุณสมบัติของระบบปฏิบัติการลีนุกซ์ที่สนับสนุนการทำงานของโปรแกรมเครื่องวัดคลื่นไฟฟ้าหัวใจ

6.1.2.1 ประสิทธิภาพ ลีนุกซ์มีประสิทธิภาพที่สูงเพียงพอกับความต้องการของโปรแกรมเครื่องวัดคลื่นไฟฟ้าหัวใจ เนื่องจากลีนุกซ์ได้ถูกออกแบบมาให้ใช้งานอุปกรณ์ฮาร์ดแวร์ทุกอย่างของเครื่องได้อย่างเต็มประสิทธิภาพ อย่างเช่น

- เคอร์เนล (Kernel) ของลีนุกซ์สนับสนุนการ Demand – Paged Loaded Executable นั่นคือเฉพาะส่วนของโปรแกรมที่กำลังถูกเรียกทำงานเท่านั้นที่จะถูกอ่านจาก ดิสก์เข้าสู่หน่อยความจำของเครื่อง ทำให้ระบบใช้งานหน่วยความจำได้อย่างมีประสิทธิภาพ นอกจากนั้นตัวเคอร์เนล ยังสามารถโหลดโปรแกรมขึ้นมาทำงานได้ด้วยวิธี " Shared Copy - on – write pages" คือยอมให้หลายๆโปรเซสใช้งานในหน่วยความจำเดียวกันได้ซึ่งทำให้ระบบทำงานเร็วขึ้นและลดขนาดการใช้งานในหน่วยความจำได้
- ลีนุกซ์จะมองอุปกรณ์ฮาร์ดแวร์เป็นไฟล์ดังนั้นการทำงานกับอุปกรณ์จึงมีความคล่องตัวและยืดหยุ่นสูงในการนำไปใช้งาน

6.1.2.2 เสถียรภาพ ลีนุกซ์มีการพัฒนาแนวคิดพื้นฐานมาจากระบบยูนิกซ์ ซึ่งมีชื่อเสียงทางด้านเสถียรภาพ ในการทำงานแต่เพิ่มความสามารถในการทำงานบนคอมพิวเตอร์ได้หลายรุ่นและมีการทำงานแบบระบบหลายผู้ใช้และหลายภารกิจ (Multi-User and Multi -Tasking)

6.1.2.3 การถ่ายโอนข้อมูล ลีนุกซ์มีความสามารถในการใช้ไฟล์ร่วมกับระบบปฏิบัติการอื่นๆ ได้เช่น FAT32 ของระบบปฏิบัติการ Window 98/Me NTFS ของ Window 2000/NT เป็นต้นจึงทำให้สะดวกต่อการถ่ายโอนข้อมูล

6.1.2.4 ระบบเครือข่าย ลีนุกซ์สนับสนุนการทำงานกับระบบเครือข่าย (Network System)บน TCP/IP ได้อย่างสมบูรณ์แบบ

6.1.2.5 มีผู้ร่วมพัฒนาอยู่ทั่วโลก มีการแลกเปลี่ยนข้อมูลตลอดเวลาถึงข้อดีและข้อเสียของระบบปฏิบัติการและคำแนะนำในการแก้ปัญหา

6.1.2.6 ระบบปฏิบัติการที่เปิด (Open Source) สิ่งที่สำคัญที่สุดคือลีนุกซ์เป็นระบบปฏิบัติการเปิดทำให้เราสามารถแก้ไขและเปลี่ยนแปลงได้ทุกอย่างของรหัสต้นฉบับได้ตามความต้องการและเหมาะสมกับระบบที่ต้องการพัฒนาให้มีประสิทธิภาพในระยะยาวอย่างต่อเนื่อง

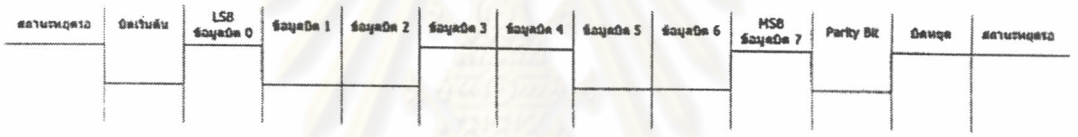
6.2 พื้นฐานการสื่อสารข้อมูลอนุกรม

การสื่อสารข้อมูลอนุกรมหมายถึงการรับส่งสัญญาณในลักษณะกลุ่มหรือบิตคราวละหนึ่งบิตเป็นลำดับเรื่อยไปจนสิ้นสุด ซึ่งเหมาะสมกับการสื่อสารในระยะทางไกลเพราะใช้สายส่งสัญญาณน้อยมากโดยเราสามารถแบ่งประเภทการสื่อสารได้ดังต่อไปนี้

6.2.1 การสื่อสารแบบไม่ประสานเวลา (Asynchronous Communication)

คือการสื่อสารที่อุปกรณ์ตัวรับกับตัวส่งไม่ได้ใช้เวลาเดียวกันในการส่ง มีการกำหนดเวลารับ/ส่ง สัญญาณไม่แน่นอน โดยทางอุปกรณ์ตัวรับจะต้องคอยตรวจสอบสัญญาณเริ่มต้นของอุปกรณ์ตัวส่งทุกครั้ง แม้ว่าการสื่อสารแบบไม่ประสานเวลาจะไม่ต้องการสัญญาณนาฬิกาของฝ่ายส่งแต่ท่งฝ่ายรับเองต้องมีสัญญาณนาฬิกาของตนเองเพื่อที่จะใช้ในการพิจารณาแยกอักขระจุดเริ่มต้นและจุดสิ้นสุดของข้อมูลได้

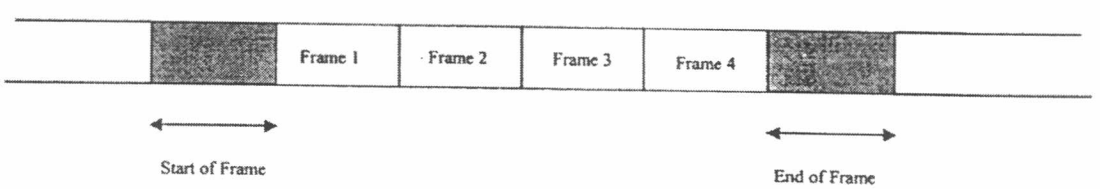
รูปแบบส่วนใหญ่ของการเข้ารหัสแบบไม่ประสานเวลาคือ การเข้ารหัสแบบแอสกี (ASCII Code) โดยมีการส่ง บิตเริ่มต้น (Start bit) 1 บิต ข้อมูล (Data bit) 7 บิต พาริตีบิต 1 บิต และสุดท้ายคือบิตสิ้นสุด (Stop bit) ซึ่งนิยมใช้ 2 บิต แสดงดังรูปที่ 6.1



รูปที่ 6.1 รูปแบบการสื่อสารแบบไม่ประสานเวลา

6.2.2 การสื่อสารแบบประสานเวลา (Synchronous Communication)

การสื่อสารแบบประสานเวลาคือการส่งสัญญาณระหว่างอุปกรณ์ตัวรับกับตัวส่งที่เวลาเดียวกันหรือที่ตัวรับจะต้องมีภาครับที่สามารถสร้างสัญญาณนาฬิกาจากข้อมูลที่ได้รับโดยใช้หลักการของเฟสล็อกลูป (Phase Lock Loop) หรือ ออสซิลเลเตอร์ (Coherent Oscillator) สัญญาณนาฬิกานี้จะใช้เป็นฐานในการรับข้อมูล ดังนั้นการสื่อสารแบบนี้จึงสำคัญที่การเชื่อมต่อสัญญาณนาฬิกาของอุปกรณ์ตัวรับตัวส่ง ด้วยการสื่อสารแบบนี้จะทำให้ข้อมูลดิจิตอลจะถูกแบ่งเป็นกลุ่มเรียกว่ากรอบ (Frame) แต่ละกรอบจะถูกส่งไปเป็นขบวนโดยไม่มีการหยุดรหว่างแต่ละอักขระและเหมือนกับการสื่อสารแบบไม่ประสานเวลาที่อักขระแต่ละตัวต้องมีบิตเริ่มต้นและบิตสิ้นสุด การสื่อสารแบบประสานเวลาในแต่ละกรอบก็ต้องมีอักขระบอกจุดเริ่มต้นและสิ้นสุดของกรอบนั้นๆและกรอบถัดไป แสดงในรูปที่ 6.2



รูปที่ 6.2 รูปแบบการสื่อสารแบบประสานเวลา

6.3 หน่วยควบคุมการติดต่อสื่อสาร

ในการสื่อสารผ่านทางพอร์ตอนุกรมระบบปฏิบัติการลีนุกซ์ ก่อนเริ่มใช้งานเราจำเป็นต้องกำหนดคุณสมบัติพื้นฐานบางอย่างให้กับพอร์ตอนุกรม เช่น ความเร็วที่ใช้สำหรับการติดต่อ จำนวนบิตของข้อมูลรวมทั้งขนาดความจำชั่วคราวที่ใช้เป็นบัฟเฟอร์ รวมถึงลักษณะการติดต่อฮาร์ดแวร์ของระบบปฏิบัติการลีนุกซ์ก็มีข้อกำหนดที่เป็นมาตรฐานที่ใช้ตามของ POSIX (Portable Operating System Interface) เพื่อความเข้าใจโดยง่าย จะทำการกำหนดคุณสมบัติการเข้าไปควบคุมการสื่อสารผ่านพอร์ตอนุกรมเป็นลำดับขั้นดังต่อไปนี้

1. กำหนดชนิดอุปกรณ์ที่ใช้ติดต่อสื่อสารอนุกรม ถ้าต้องการจะติดต่อพอร์ตอนุกรมบนระบบปฏิบัติการลีนุกซ์นั้นมีข้อกำหนดที่ต้องรู้ก่อนว่า "ลีนุกซ์มองอุปกรณ์ฮาร์ดแวร์ในรูปแบบของระบบไฟล์ (Files System) และมีการเรียกการทำงานผ่านระบบไฟล์เช่นกัน" โดยถ้าต้องการจะติดต่อพอร์ต 1 จะต้องอ้างไฟล์ "/dev/tys0" และพอร์ต 2 ที่ "/dev/ttys1"
2. การกำหนดความเร็วและคุณสมบัติอื่นๆที่ใช้ในการสื่อสาร ซึ่งเป็นการกำหนดคุณสมบัติให้กับพอร์ตสื่อสารตามข้อกำหนดของ POSIX สำหรับการสื่อสารแบบอนุกรม โดยกำหนดเข้าไปที่ตัวบ่งชี้ (Flag) ของโครงสร้างการกำหนดควบคุมเทอร์มินอลดังแสดงในตารางที่3.1 ซึ่งตัวบ่งชี้ C_Cflag จะเป็นตัวบอกได้ว่าสามารถกำหนดอะไรได้บ้างดังเสดงในตารางที่ 6.2

ตารางที่ 6.1 ตัวบ่งชี้ (flag) ในการกำหนดการทำงานให้กับพอร์ตอนุกรม

Member	Description
c_cflag	Control options
c_lflag	Line options
c_iflag	Input options
c_oflag	Output options
c_cc	Control characters
c_ispeed	Input baud (new interface)
c_ospeed	Output baud (new interface)

รูปแบบในการกำหนดความเร็วจะใช้คำสั่งต่อไปนี้

```
Var Comport : termios;
// กำหนดตัวแปร Comport ให้เป็นตัวแปรชนิด termios
Cfsetispeed (Comport,B19200)
//กำหนดความเร็วในการอ่านข้อมูลให้มีค่าเท่ากับ 19200 บิต / วินาที
Cfsetospeed (Comport,B19200
//กำหนดความเร็วในการส่งข้อมูลให้มีค่าเท่ากับ 19200 บิต / วินาที
```

รูปแบบในการกำหนดการใช้จำนวนบิตในการรับส่งข้อมูล

```
Comport.c_cflag :=comport.c_cflag and not CSTOPB
// กำหนดให้มีการใช้ 1 บิตหยุด (Stop bit) ในการส่งข้อมูล
Comport.c_cflag :=comport.c_cflag and not CSIZE
// กำหนดให้ไม่มีการใช้บิตพราง (Mask bit) ในรูปแบบการส่งข้อมูล
Comport.c_cflag :=comport.c_cflag or CS8
// กำหนดให้มีการส่งข้อมูลแบบ 8 บิต (ถ้า 7 บิตก็จะเป็น CS7)
```

รูปแบบในการกำหนดกรตรวจสอบพาริตีบิต

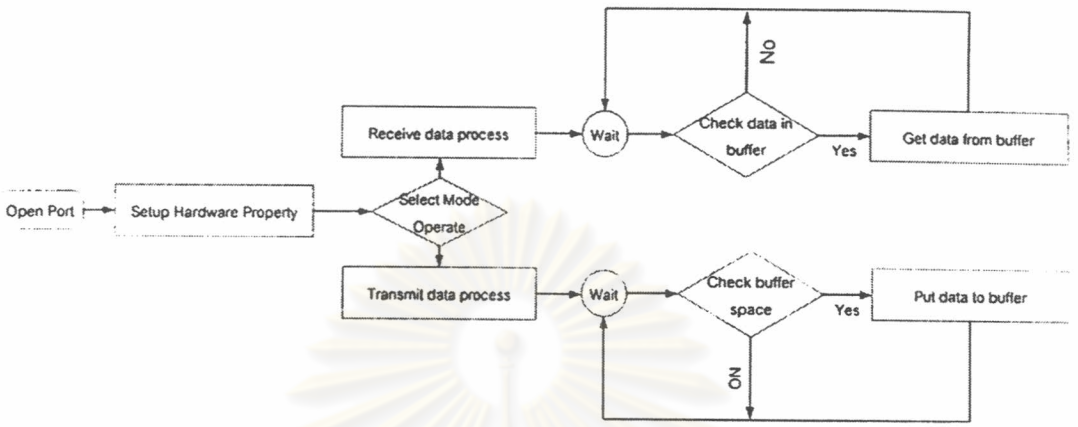
```
Comport.c_cflag :=comport.c_cflag and (not PARENB)
// กำหนดให้ไม่มีการตรวจสอบพาริตีบิต
Comport.c_cflag :=comport.c_cflag or PARENB
Comport.c_cflag :=comport.c_cflag and (not PARODD)
// กำหนดให้มีการตรวจสอบบิตคู่
Comport.c_cflag :=comport.c_cflag or PARENB
Comport.c_cflag :=comport.c_cflag or PARODD
// กำหนดให้มีการตรวจสอบบิตคู่
```

นอกจากนี้แล้วยังต้องกำหนดลักษณะการควบคุมสายงาน (Flow Control) ว่าต้องการควบคุมสายงานแบบใดในรูปของซอฟต์แวร์อัตโนมัติหรือปล่อยให้เป็นไปตามการทำงานของระบบฮาร์ดแวร์ ส่วนการกำหนดรายละเอียดในการสร้างซอฟต์แวร์ที่นอกเหนือที่กล่าวไปแล้วนั้น เพื่อที่จะเข้าไปควบคุมพอร์ตอนุกรมให้มีความสามารถในลักษณะใดก็เป็นเทคนิคของผู้ออกแบบว่าจะใช้ข้อกำหนดที่มีให้ในรูปแบบใด

ตารางที่ 6.2 คุณสมบัติของตัวบ่งชี้ (flag) c_cflag [16]

CONSTANT	DESCRIPTION
CBAUD	Bits mask for baud rate
B0	0 baud (drop DTR)
B50	50 baud
B75	75 baud
B110	110 baud
B134	134.5 baud
B150	150 baud
B 200	200 baud
B300	300 baud
B600	600 baud
B1200	1200 baud
B1800	1800 baud
B2400	2400 baud
B4800	4800 baud
B9600	9600 baud
B19200	19200 baud
B38400	38400 baud
B57600	57600 baud
B76800	76800 baud
B115200	115200 baud
EXTA	External rate clock
EXTB	External rate clock
CSIZE	Bit mask for data bits
CS5	5 data bits
CS6	6 data bits
CS7	7 data bits
CS8	8 data bits
CSTOPB	2 stop bits (1 otherwise)
CREAD	Enable reciever
PARENB	Enable parity bit
PARODD	Use odd parity instead of even
HUPCL	Hangup (drop DTR) on last close
CLOCAL	Local line – do not change "owner" port
LOBLK	Block job control output
CNEW_RTSCTS CRTSCTS	Enable hardware flow control (not supported on all platforms)

หลังจากกำหนดคุณสมบัติของการสื่อสารทางพอร์ตอนุกรมเรียบแล้วขั้นตอนต่อไปที่จะต้องทำคือจะต้องกำหนดกิจกรรมที่สำหรับการองรับการเรียกใช้พอร์ตอนุกรมในแต่ละครั้ง ดังอธิบายได้ตามรูปที่ 6.3



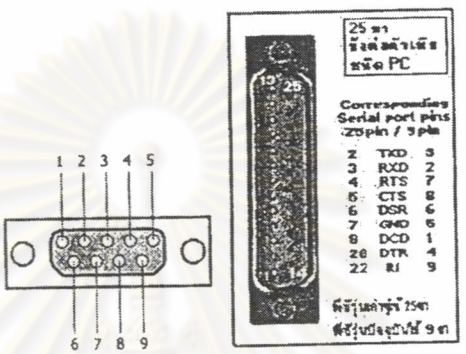
รูปที่ 6.3 ขั้นตอนของกิจกรรมเมื่อมีการเรียกใช้พอร์ตอนุกรม [20]

ในขั้นตอนแรกของการเรียกใช้พอร์ตอนุกรมนั้นจะเริ่มที่การตรวจสอบการเปิดหรือปิดพอร์ตก่อนแล้วจากนั้นก็จะกำหนดคุณสมบัติที่ต้องการให้พอร์ตอนุกรมทำงานได้ตามจุดประสงค์ลงไปตามลำดับที่กล่าวไปแล้วในหัวข้อก่อนหน้านี้หลังจากกำหนดคุณสมบัติเสร็จก็จะต้องทำการเลือกลักษณะการทำงานว่าขณะนี้ต้องการที่จะส่งข้อมูลหรือต้องการที่จะรับข้อมุล หลังจากนี้จะเป็นส่วนหน้าที่ของการทำงานของโปรแกรมคอยตรวจสอบว่าการทำงานตรงกับที่กำหนดไว้ตามคุณสมบัติหรือไม่ เช่นเลือกเหตุการณ์การทำงานแบบรับข้อมูลแบบอัตโนมัติ การทำงานของโปรแกรมต้องมีการตรวจสอบภายในบัฟเฟอร์ของพอร์ตอนุกรมตลอดเวลาว่ามีข้อมูลอยู่หรือไม่ถ้ามีข้อมูลอยู่ก็จะสร้างสัญญาณอินเตอร์รัพต์ไปบอกเมนโปรแกรมที่ทำงานอยู่ว่ามีข้อมูลมาแล้วพร้อมจะรับหรือไม่เป็นต้น

6.4 การสื่อสารด้วยมาตรฐาน RS – 232/RS-485

6.4.1 มาตรฐาน RS-232

มาตรฐาน RS-232 เป็นมาตรฐานการสื่อสารผ่านพอร์ตอนุกรมถูกกำหนดโดย EIA (Electronic Industrial Association) ซึ่งเราสามมารถแบ่งได้ตามคุณลักษณะได้ 2 ประเภทคือ คุณสมบัติทางกลได้แก่คุณสมบัติของสายไฟและการกำหนดคุณสมบัติจุดข้อต่อรวมถึงระยะทางที่สามารถทำงานได้ดังแสดงในรูปที่ 6.4



รูปที่ 6.4 จุดข้อต่อแบบ DB-9 และ DB-25

ตารางที่ 6.3 รายละเอียดสายสัญญาณของจุดข้อต่อทั้งแบบ DB-9 และ DB-25

คอนเน็กเตอร์ DB-9	คอนเน็กเตอร์ DB-25	ชื่อของสายสัญญาณ	ชนิดของสายสัญญาณ
1	8	Data carrier Detect :DCD	Input
2	3	Received Data : RxD	Input
3	2	Transmitted Data : TxD	Output
4	20	Data Terminal Ready : DTR	Out put
5	7	Signal Ground	-
6	6	Data Set Ready : DSR	Input
7	4	Request to Send : RTS	Output
8	5	Clear to Send : CTS	Input
9	22	Ring Indicator : RI	Input

- Receive Data : Rd หรือ RxD ขาชนิดนี้ใช้เพื่อรับสัญญาณอนุกรมเข้ามายังคอมพิวเตอร์ โดยนำข้อมูลที่อ่านได้เก็บไว้ในรีจิสเตอร์บัฟเฟอร์
- Transmitted Data : TDหรือ TxD ขานี้ใช้เพื่อส่งข้อมูลออกจากคอมพิวเตอร์โดยนำข้อมูลที่เก็บอยู่ในบัฟเฟอร์สำหรับส่งข้อมูลส่งออกไป

- Data Terminal Ready : DTR เป็นขาสัญญาณที่ส่งออกจากคอมพิวเตอร์เพื่อให้อุปกรณ์ปลายทางรับรู้ว่าต้องการติดต่อด้วย โดยขา DTR นี้จะต้องเชื่อมต่อกับขา DSR ของอุปกรณ์ปลายทาง และขา DTR ของอุปกรณ์ปลายทางจะต้องเชื่อมต่อกับขา DSR ของคอมพิวเตอร์ ถ้าใช้การเชื่อมต่อแบบ Null Modem ซึ่งใช้สายในการเชื่อมต่อเพียง 3 เส้น จะต้องต่อขา DTR และ DSR ของตัวมันเองเข้าด้วยกันและต้องต่อกับขา DCD ด้วยในกรณีที่โปรแกรมสื่อสารที่ใช้มีการตรวจจับสัญญาณพาห์
- Signal Ground : GND ขากราวนด์ของระบบ
- Data Set Ready : DSR ขานี้จะใช้คู่กับขา DTR เพื่อตรวจสอบการเชื่อมต่อกันระหว่างคอมพิวเตอร์กับอุปกรณ์ปลายทางซึ่งขา DSR นี้จะเป็นขาสำหรับรับข้อมูลจากภายนอกซึ่งถูกส่งมาจากขา DTR
- Request to Send : RTS เป็นขาสำหรับส่งสัญญาณร้องขอให้ทางอุปกรณ์ปลายทางส่งข้อมูลส่งกลับมายังคอมพิวเตอร์ โดยขาที่รับสัญญาณ RTS ก็คือขา CTS ในกรณีที่ใช้การเชื่อมต่อแบบ Null Modem 3 สาย จะต้องเชื่อมต่อขา RTS และ CTS ของตัวมันเองเข้าด้วยกันเพื่อจะให้การรับและส่งข้อมูลสามารถเกิดขึ้นได้ตลอดเวลา
- Clear to Send : CTS ขานี้จะคอยรับสัญญาณจากขา RTS เมื่อนับสัญญาณได้ ข้อมูลที่ขานี้จึงถูกใช้เพื่อตรวจสอบอุปกรณ์ต่อพ่วงว่าพร้อมที่จะรับข้อมูลหรือไม่

คุณสมบัติของสัญญาณไฟฟ้า

1. การสื่อสารในมาตรฐาน RS-232 จะใช้สัญญาณระดับ (Level) ในการสื่อสารข้อมูลโดยอาจพิจารณาในคู่สัญญาณดังต่อไปนี้

   MARK / SPACE

   OFF / On

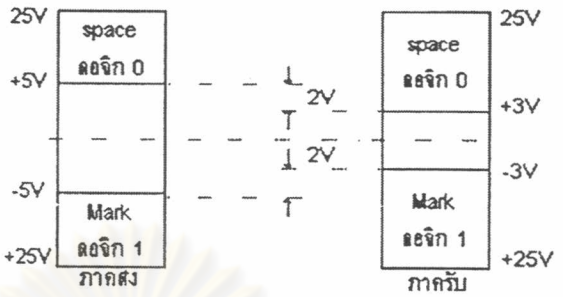
   ลอจิก 1 / ลอจิก 0

ระดับสัญญาณจะแทนด้วยระดับลอจิกลบ (Negative Logic) โดยระดับสัญญาณจะมีลักษณะดังนี้

ตารางที่ 6.4 ลักษณะของระดับสัญญาณ

สถานะ	ระดับแรงดันสัญญาณ	
	-3V ถึง –25V	3V ถึง 25V
	1	0
ระดับสัญญาณลอจิก	MARK	SPACE
	OFF	ON

2. ตัวขับสัญญาณจะต้องส่งสัญญาณระหว่าง-3ถึง-25V และ3 ถึง 25V และยอมให้มีการลดทอนของสัญญาณได้ไม่เกิน 2V



รูปที่ 6.5 ระดับสัญญาณของภาครับและภาคส่ง

ในกรณีเช่นนี้แสดงว่าการรับส่งสัญญาณตามมาตรฐาน RS-232-C จะยอมให้มี noise margin ได้ไม่เกิน 2 V นอกจากนั้น MARK ของ SPACE ยังอาจแทนได้ด้วยการไหลของกระแสไฟฟ้า (Current Loop)

2. ตัวเก็บประจุ CL ซึ่งขนานกับอุปกรณ์ปลายทาง จะต้องมีค่าไม่เกิน 2500 pF โดยไม่รวมความจุของเคเบิล
4. แรงดันไฟฟ้าของเปิดวงจรจะต้องมีค่าไม่เกิน +-25 V
5. วงจรรับสัญญาณ RS-232-C จะต้องทนต่อการลัดวงจรของสัญญาณได้ โดยไม่ทำให้ภาครับ และอุปกรณ์ต่อพ่วงเสียหาย

6.5 การออกแบบโปรแกรมเพื่อใช้งานส่วนการเชื่อมต่อฮาร์ดแวร์กับคอมพิวเตอร์ส่วนบุคคล

เนื่องจากวิทยานิพนธ์ฉบับนี้นำเสนอโครงงานที่ต้องรับข้อมูลจาก ไมโครคอนโทรลเลอร์ ซึ่งส่งข้อมูลจากการแปลงสัญญาณอนาลอกเป็นดิจิตอล จำนวน 12 บิตเข้าเครื่องคอมพิวเตอร์ส่วนบุคคล ผ่านทางพอร์ตอนุกรม ในส่วนการออกแบบเลือกการสื่อสารแบบไม่ประสานเวลา และข้อมูล 12 บิต ต้องส่งทีละ 8 บิต 2 ครั้ง โดยจะตั้งรหัสในการส่ง 8 บิตบนและ 8 บิตล่าง ในส่วนภาครับเมื่อได้รับข้อมูลเข้าไปต้องตรวจสอบข้อมูลที่ได้รับว่าเป็นข้อมูล 8 บิตบนหรือ 8 บิตล่างในส่วนนี้ต้องออกแบบอัลกอริทึมให้ตรวจสอบข้อมูลได้อย่างถูกต้อง (ผนวก)

ตัวอย่างการส่งข้อมูล

1	0	X	X	X	X	X	X

ก) แสดงการส่ง 8 บิต บน

0	1	X	X	X	X	X	X

ข) แสดงการส่ง 8 บิต ล่าง

รูปที่ 6.6 แสดงรหัสการส่งข้อมูล 16 บิต

หมายเหตุ

1 0 แทนรหัสข้อมูล 8 บิตบน

0 1 แทนรหัสข้อมูล 8 บิตล่าง

X แทน ข้อมูล