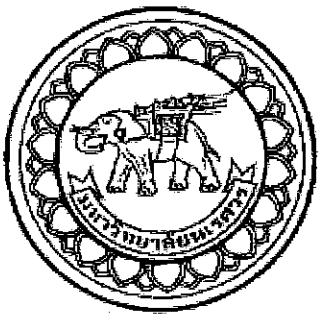
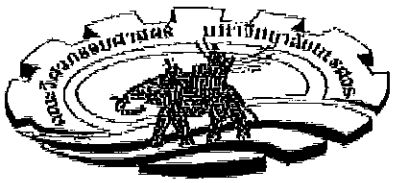


การควบคุมอุปกรณ์ผ่านระบบเครือข่าย

CONTROLLING HARDWARE VIA NETWORK

นายมานพ นาคมี รหัส 45360336

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร
ปีการศึกษา 2548



ใบรับรองโครงงานวิศวกรรม

หัวข้อโครงงาน	การควบคุมอุปกรณ์ผ่านระบบเครือข่าย
ผู้ดำเนินโครงงาน	นายมานพ นาคมี รหัส 45360336
อาจารย์ที่ปรึกษา	อาจารย์พงศ์พันธ์ กิจสนาโยธิน
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
ปีการศึกษา	2548

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้โครงงานฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์

คณะกรรมการสอบโครงงานวิศวกรรม

P. Kijuttim .....ประธานกรรมการ

(อาจารย์พงศ์พันธ์ กิจสนาโยธิน)

.....กรรมการ

(อาจารย์พนมขวัญ ริยะมงคล)

.....กรรมการ

(อาจารย์อัครพันธ์ วงศ์กังแห)

หัวข้อโครงงาน	การควบคุมอุปกรณ์ผ่านระบบเครือข่าย
ผู้ดำเนินโครงงาน	นายมานพ นาคมี รหัส 45360336
อาจารย์ที่ปรึกษา	อาจารย์พงศ์พันธ์ กิจสนาโยธิน
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
ปีการศึกษา	2548

.....

บทคัดย่อ

โครงงานการควบคุมอุปกรณ์ผ่านระบบเครือข่ายนี้ เป็นการศึกษาและพัฒนาโปรแกรมที่สามารถติดต่อและควบคุมอุปกรณ์ทางไฟฟ้าหรืออิเล็กทรอนิกส์ ผ่านทางระบบเครือข่ายคอมพิวเตอร์ โดยลักษณะการทำงานของโปรแกรมจะเป็นไปในรูปแบบเครื่องแม่ข่ายกับเครื่องลูกข่าย (Client/Server) ซึ่งโปรแกรมในส่วนของเครื่องแม่ข่าย (Server Program) จะแบ่งเป็นสองส่วน คือ ส่วนของเว็บเซิร์ฟเวอร์ (Web Server) ที่มีการติดต่อรับภาพจากกล้องเว็บแคม (Web Cam) เพื่อคอยให้บริการกับเครื่องลูกข่ายในการชมภาพแบบเวลาจริง (Real time) ผ่านทางเว็บบราวเซอร์ (Web Browser) ส่วนที่สองคือ ส่วนควบคุมการเปลี่ยนมุมการถ่ายภาพของกล้องเว็บแคม ที่อยู่ในส่วนแรก และควบคุมการเปิด-ปิดของอุปกรณ์ไฟฟ้า โดยที่เครื่องแม่ข่ายจะติดต่อควบคุมอุปกรณ์นั้นผ่านทางไมโครคอนโทรลเลอร์ (Microcontroller) ซึ่งคำสั่งที่ใช้สั่งงานในการควบคุมนั้นจะได้รับจากโปรแกรมที่อยู่ทางฝั่งเครื่องลูกข่าย โดยในโครงงานได้เลือกใช้ภาษาจาวา (Java) ในการพัฒนาโปรแกรม ผลที่ได้จากโครงงานก็คือผู้ใช้สามารถรับชมภาพจากกล้องเว็บแคม พร้อมกันนั้นยังสามารถเปลี่ยนมุมมองของกล้องได้ และสั่งเปิด-ปิดอุปกรณ์ไฟฟ้าจากระยะไกลได้

Project Title	Controlling Hardware via Network
Name	Mr. Manop Nakmee ID. 45360336
Project Advisor	Mr. Phongphun Kijsanayothin
Major	Computer Engineering
Department	Electrical and Computer Engineering
Academic Year	2005

ABSTRACT

This project is about controlling hardware through the network system. The purposes of this project are to study and develop programs which can connect and control electrical equipments or electronics through the network system. It is a client/server form. The server can be separated into two parts. The first part is a web server which receives pictures from a web cam and sends them to client to monitoring in the real time through a web browser. The second part is a control view changing of a web cam from the first part and control electrical equipment. The server connects and controls the hardware through a microcontroller. The hardware will be controlled from the client program. This project used java language in developing program. In the result, a user can watch pictures from a web cam. Moreover, the user can change the view of the web cam and control electrical equipment to be turned on or off from the long distance.

กิตติกรรมประกาศ

โครงงานการควบคุมอุปกรณ์ผ่านระบบเครือข่าย ได้จัดทำขึ้นและสำเร็จลุล่วงไปได้ด้วยดี ในระหว่างการจัดทำโครงงานได้เกิดปัญหาและอุปสรรคมากมาย ซึ่งปัญหาและอุปสรรคเหล่านั้นก็ถูกแก้ไขจนสำเร็จลุล่วงได้ด้วยดี ทั้งนี้เนื่องมาจากข้าพเจ้าได้รับคำปรึกษาที่ดีจากอาจารย์ที่ปรึกษา ซึ่งก็คือ อาจารย์พงศ์พันธ์ กิจสนาโยธิน ที่ยินดีให้คำปรึกษา และให้คำแนะนำที่ดีทุกเรื่องตลอดมา ซึ่งต้องขอขอบพระคุณอาจารย์เป็นอย่างมาก อีกทั้งรุ่นพี่และเพื่อนๆ ในภาควิชาวิศวกรรมคอมพิวเตอร์ รวมถึงนายอภิสิทธิ์ ตั้งวชิรกุล เพื่อนร่วมรุ่นในภาควิชาเครื่องกล ซึ่งต่างก็มีส่วนร่วมช่วยให้โครงงานนี้สำเร็จไปด้วยดี

ที่สำคัญต้องขอบพระคุณบุคคลที่สำคัญยิ่ง ที่ทำให้ข้าพเจ้ามีวันนี้ก็คือ บิดา มารดา อันเป็นที่เคารพรักยิ่ง ซึ่งได้เลี้ยงดูข้าพเจ้าเป็นอย่างดี พร้อมให้โอกาสในการศึกษาอย่างเต็มที่ และยังให้กำลังใจ เอาใจใส่เสมอมา ในทุกๆ ด้านอันหาที่เปรียบมิได้ ข้าพเจ้าขอระลึกในพระคุณอันสุดประมาณ และขอกราบขอบพระคุณมา ณ ที่นี้

นายมานพ นาคมี

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

สารบัญรูป

รูปที่ หน้า

สารบัญรูป (ต่อ)

รูปที่ หน้า

บทที่ 1
บทนำ

1.1 แนวคิดและที่มาของปริญญานิพนธ์

ปัจจุบันระบบเครือข่ายคอมพิวเตอร์ ได้เข้ามามีบทบาทในสังคมแห่งเทคโนโลยีเป็นอย่างมาก ซึ่งประโยชน์ที่ได้รับจากเครือข่ายคอมพิวเตอร์มีหลายด้านด้วยกัน ประโยชน์ด้านที่เห็นได้ชัดที่สุดก็คือ การใช้งานเป็นเครือข่ายสำหรับการติดต่อสื่อสารข้อมูลทางด้านอินเตอร์เน็ต ซึ่งก็เป็นที่ทราบกันดีว่าระบบเครือข่ายอินเตอร์เน็ตเป็นระบบเครือข่ายคอมพิวเตอร์ที่ครอบคลุมไปทั่วทุกมุมโลก ดังนั้นจึงทำให้มีการใช้งานกันอย่างแพร่หลาย นอกจากนี้ยังมีการใช้ประโยชน์จากระบบเครือข่ายคอมพิวเตอร์ในด้านอื่น ๆ อีก ซึ่งการใช้งานก็จะมีลักษณะแตกต่างกันออกไปตามกลุ่มผู้ใช้ อาทิเช่น

1.1.1 การควบคุมการปิด-เปิด เครื่องใช้ไฟฟ้าที่บ้าน จากสำนักงานหรือนอกบ้าน ผ่านระบบเครือข่ายอินเตอร์เน็ต

1.1.2 สำหรับงานด้านอุตสาหกรรมก็สามารถควบคุมอุปกรณ์ต่างๆ ในโรงงาอุตสาหกรรม เช่น วัดค่าพลังงานไฟฟ้า , แรงดัน , อุณหภูมิ , หรือค่าต่างๆ เชิงตังเลขจากจุดใดๆ ที่อยู่ห่างไกล เพื่อมาใช้งานต่อไปได้ และ ยังสามารถควบคุมอุปกรณ์เหล่านั้นได้

1.1.3 สำหรับทางด้านการแพทย์ การวัดค่าความดันโลหิต อัตราการเต้นของหัวใจ หรืออื่นๆ เพื่อตรวจดูคนไข้ที่อยู่ห่างไกลได้

1.1.4 นักสิ่งแวดล้อม นักเคมี สามารถวัดค่าความเป็นกรด- ด่าง (pH) ความชื้นในอุณหภูมิ หรือค่าอื่นๆ จากระยะไกลได้ และยังสามารถควบคุมการทำงานของเครื่องมือ อุปกรณ์ต่างๆ ได้อีกด้วย

1.1.5 การเกษตรสมัยใหม่ ใช้ในการควบคุมระบบ การวัด หรือ การให้น้ำ , ปุ๋ย , สารเคมี , ความชื้น , ค่าความเป็นกรด- ด่าง (pH)

จากตัวอย่างข้างต้นพบว่า ผลที่ได้รับจากการใช้งานระบบเครือข่ายคอมพิวเตอร์นั้นมีประโยชน์มากมาย กับกลุ่มบุคคลในหลายๆ สาขา ดังนั้นในการจัดทำโครงงาน การควบคุมฮาร์ดแวร์ผ่านระบบเครือข่ายขึ้นนั้น ก็เพื่อเป็นการสร้างประโยชน์จากระบบเครือข่ายคอมพิวเตอร์ขึ้นในอีกด้านหนึ่ง ประกอบกับการที่จะใช้โครงงานที่สร้างขึ้นนี้ เป็นแนวทางในการพัฒนาและประยุกต์ใช้ในงานด้านต่างๆ เพื่อเป็นประโยชน์ต่อไป

1.2 วัตถุประสงค์ของโครงงาน

โครงงานการควบคุมอุปกรณ์ผ่านระบบเครือข่ายนี้ จัดทำขึ้นภายใต้วัตถุประสงค์หลัก คือ เพื่อเป็นการสร้างโปรแกรมคอมพิวเตอร์ที่สามารถติดต่อและควบคุมอุปกรณ์ทางไฟฟ้าหรืออิเล็กทรอนิกส์ ผ่านทางระบบเครือข่ายคอมพิวเตอร์

1.3 ขอบเขตของโครงงาน

1. สร้างโปรแกรมเพื่อติดต่อและรับภาพจากกล้องวีดีโอ (Web Cam) ผ่านทางพอร์ตยูเอสบี (USB Port)

2. สร้างโปรแกรมของเครื่องแม่ข่าย (Server program) ในการเป็นเว็บเซิร์ฟเวอร์ (Web - server) เพื่อให้บริการภาพวีดีโอที่เป็นแบบวงจรปิดกับเครื่องคอมพิวเตอร์ลูกข่าย (Client) ผ่านระบบอินเตอร์เน็ตได้

3. สร้างโปรแกรมเพื่อติดต่อกับพอร์ตอนุกรม (Serial Port) เพื่อส่งข้อมูล ใน มาตรฐาน RS232 ไปยัง MCS-51

4. สามารถใช้ MCS-51 ในการควบคุมอุปกรณ์ทางไฟฟ้าและอิเล็กทรอนิกส์ ซึ่งในโครงงานนี้จะเป็นการควบคุมสเต็ปปิ้งมอเตอร์ (Stepping Motor) เพื่อใช้ควบคุมการเปลี่ยนมุมของการถ่ายภาพของกล้องวีดีโอ (Web Cam)

5. สร้างโปรแกรมของเครื่องแม่ข่าย (Server program) เพื่อให้บริการกับเครื่องลูกข่ายที่ต้องการเข้ามาควบคุมฮาร์ดแวร์ที่ต่ออยู่

6. สร้างโปรแกรมของเครื่องลูกข่าย (Client program) ที่สามารถควบคุมอุปกรณ์ทางไฟฟ้าหรืออิเล็กทรอนิกส์ ซึ่งในโครงงานนี้คือ สเต็ปปิ้งมอเตอร์ ที่อยู่ที่เครื่องแม่ข่ายผ่านทางระบบเครือข่าย

7. กำหนดรูปแบบโปรโตคอลการสื่อสารระหว่างเครื่องคอมพิวเตอร์แม่ข่าย (Server) และโปรแกรมจากเครื่องลูกข่าย (Client program)

1.4 ขั้นตอนการดำเนินงาน

กิจกรรม	ก.พ. 48	มี.ค. 48	เม.ย. 48	พ.ค. 48	มิ.ย. 48	ก.ค. 48	ส.ค. 48
1. เก็บรวบรวมข้อมูล	←	—	→				
2. ศึกษาทฤษฎีที่ใช้ในปริญญานิพนธ์		←	—	→			
3. เขียนโปรแกรมส่วนติดต่อกับกล้องวีดีโอ(Web Cam)		←	—	—	→		
4. เขียนโปรแกรมส่วนติดต่อกับพอร์ตอนุกรม MCS-51 และสร้างส่วนอุปกรณ์ควบคุม				←	—	→	
5. เขียนโปรแกรมเพื่อสื่อสารข้อมูล และควบคุมฮาร์ดแวร์ผ่านทางระบบเครือข่าย				←	—	—	→
6. ติดตั้งใช้งาน แก้ไขข้อผิดพลาด และสรุปผล							← — →

1.5 ผลที่คาดว่าจะได้รับ

1. สามารถควบคุมฮาร์ดแวร์จากระยะไกลผ่านทางระบบเครือข่ายได้

2. สามารถนำโครงงานนี้ไปติดตั้งใช้งานจริงเพื่อเป็นกล้องวงจรปิดเพื่อดูภาพจากระยะไกลผ่านทางอินเตอร์เน็ต

3. สามารถนำโครงงานนี้ไปประยุกต์ใช้งานในด้านอื่นๆ หากมีการปรับปรุงและพัฒนาต่อไปจะเกิดประโยชน์มากยิ่งขึ้น

4. เป็นแนวทาง ให้ผู้ที่มีความสนใจเรื่องการควบคุมฮาร์ดแวร์ผ่านระบบเครือข่าย ได้มีแนวทางในการศึกษาโดยใช้แนวคิดจากโครงงานนี้

1.6 งบประมาณที่ใช้

1. กล้องวีดีโอ (Web Cam)	1,200 บาท
2. ชุดแผงวงจรควบคุม (MCS-51 Board, Stepping Motor)	1,500 บาท
3. สายไฟ, อุปกรณ์เชื่อมต่อ (Connector) ต่าง ๆ	200 บาท
4. ค่าเอกสารและวัสดุอุปกรณ์อื่น ๆ	1,000 บาท
รวมเป็นเงินทั้งสิ้น	3,900 บาท

บทที่ 2
หลักการและทฤษฎีที่เกี่ยวข้อง

เนื่องจากในการทำโครงงานนี้ ประกอบด้วยทฤษฎีในหลายๆ แขนงมาประกอบเข้ารวมกัน ซึ่งมีทั้งส่วนที่เป็นฮาร์ดแวร์และซอร์ฟแวร์ ดังนั้นก่อนที่จะเริ่มดำเนินการสร้างโครงงาน ควรมีการศึกษาและทำความเข้าใจเกี่ยวกับหลักการและทฤษฎีต่าง ๆ ก่อน ซึ่งจะช่วยให้การปฏิบัติงานเป็นได้อย่างรวดเร็วขึ้น และเกิดปัญหาระหว่างการปฏิบัติงานน้อยลง ดังนั้นจึงได้มีการศึกษาและรวบรวมหลักการและทฤษฎีต่างๆ ที่ใช้ในการทำปริญญานิพนธ์นี้ ซึ่งแบ่งเป็นหัวข้อตามหลักการและทฤษฎีต่างๆ ได้ ดังต่อไปนี้

2.1 The Java Communications API

Java Communications API [1] เป็น API (Application Programming Interface) มาตรฐานของจาวา (Java) ตัวหนึ่ง ที่ได้จัดเตรียมความสามารถในการติดต่อสื่อสารกับพอร์ตอนุกรม (RS232 Serial Port) และ พอร์ตขนาน (IEEE 1284 Parallel Port) ไว้ ซึ่ง API นี้ ไม่ได้รวมมากับ Java Development Kit (JDK) แต่สามารถที่จะเพิ่มเติมเข้าไปเป็น API เสริมให้กับ JDK ได้

โดย Java Communications API รุ่น 2.0 นั้น ได้ถูกสร้างขึ้นในปี 1998 ซึ่งสามารถใช้ API นี้ในการเขียนโปรแกรมเพื่อติดต่อกับกับพอร์ตอนุกรม และ พอร์ตขนาน ด้วยภาษาจาวาได้ และสามารถใช้ได้ ในหลายระบบปฏิบัติการ ถึงแม้ว่า ภาษาจาวาจะไม่ขึ้นอยู่กับระบบปฏิบัติการก็ตาม แต่การเขียนโปรแกรมเพื่อติดต่อกับฮาร์ดแวร์โดยตรง จะต้องดำเนินการผ่านระบบปฏิบัติการอีกขั้นหนึ่ง ดังนั้น API ที่ถูกสร้างขึ้นจึงรองรับเฉพาะบางระบบปฏิบัติการ

โฮมเพจที่สามารถค้นหา Java Communications API แต่ละระบบปฏิบัติการได้

1. Windows และ Solaris: http://java.sun.com/product/javacomm
2. Linux: http://www.rxtx.org
3. Mac OS: http://homepage.mac.com.pcbeard/javax.comm.MRJ

สำหรับการสื่อสารข้อมูลผ่านพอร์ตของเครื่องคอมพิวเตอร์ จะเป็นการดำเนินการตามหลักมาตรฐานอินพุต / เอาต์พุตสตรีม (I/O Stream) ของภาษาจาวา [2] โดยที่การควบคุมพอร์ตในการที่จะ เปิดพอร์ต, อ่าน/เขียน และปิดพอร์ต จะรวมอยู่ในคลาสชื่อ Commport คลาสนี้จะรวมอยู่ในแพ็กเกจของ Java Communications API ซึ่งมีชื่อว่า javax.comm ในไลบารี่ (Library) นี้จะประกอบไปด้วย Classes, Interface และ Exceptions ดังแสดงในตารางที่ 2.1

ตารางที่ 2.1 ส่วนประกอบของ javax.comm Package

Classes	Interface	Exceptions
Commport	CommDriver	NoSuchPortException
CommPortIdentifier	CommPortOwnershipListener	PortInUseException
ParellelPort	ParellelPortEvenListener	UnsupportedCommOperationException
ParellelPortEvent	SerialPortEventListener	
SerialPort		
SerialPortEvent		

2.1.1 การติดตั้งและกำหนดค่าต่างๆ ให้กับ Java Communications API

ในโครงงานนี้เป็นการพัฒนาโปรแกรมบนระบบปฏิบัติการวินโดว์ (Windows OS) จึงได้เลือกศึกษา Java Communications API ในรูปแบบของระบบปฏิบัติการวินโดว์ เมื่อเครื่องคอมพิวเตอร์ที่จะทำการเขียนโปรแกรมได้ทำการลง JDK (Java Developer Kit) ไว้เรียบร้อยแล้ว จากนั้นปฏิบัติตามขั้นตอนดังต่อไปนี้

1. ทำการดาวน์โหลด **javacomm20-win32.zip** จากhttp://java.sun.com/product/javacomm จากนั้นทำการขยายไฟล์โดยเก็บไว้ใน C:\ เมื่อทำการขยายไฟล์แล้วจะได้ C:\commapi

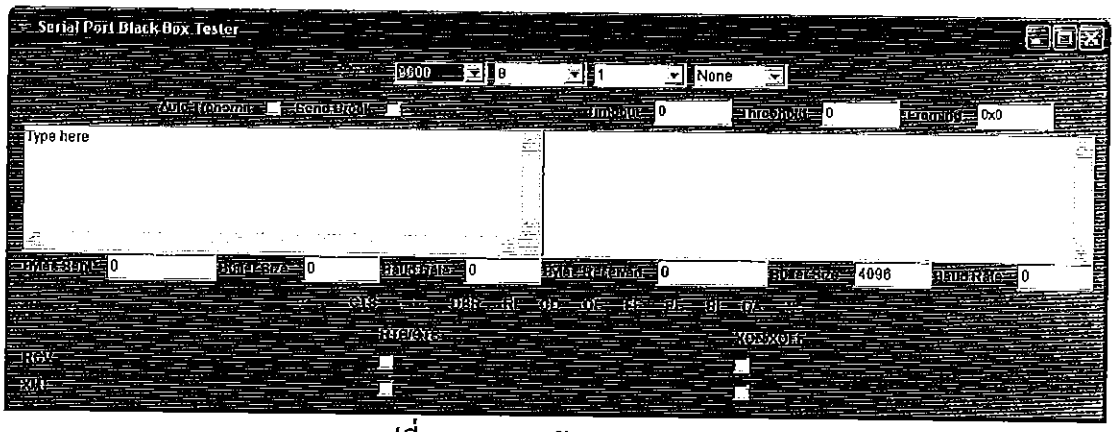
2. คัดลอก win32com.dll ใน C:\commapi ไว้ใน JAVA_HOME\bin ที่ได้ติดตั้งไว้

3. คัดลอก comm.jar ใน C:\commapi ไว้ใน JAVA_HOME\bin ที่ได้ติดตั้งไว้

4. คัดลอก javax.comm.properties ใน C:\commapi ไว้ใน JAVA_HOME\lib ที่ได้ติดตั้งไว้

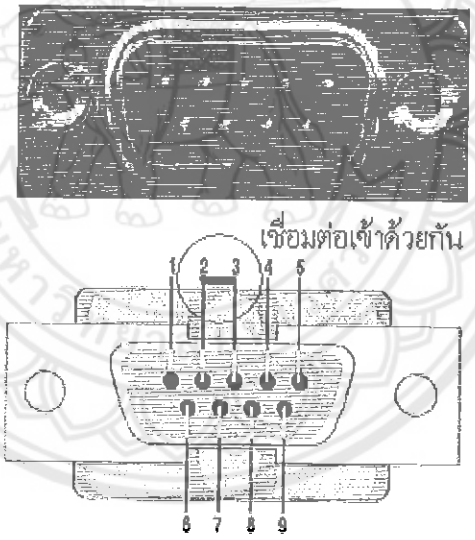
5. เพิ่ม comm.jar เข้าไปใน CLASSPATH

6. ทำการทดสอบโดยใช้โปรแกรมตัวอย่างของ Java Communications API ที่อยู่ในโฟลเดอร์ commapi\samples\BlackBox ที่ได้ทำการขยายไว้ โดยโปรแกรมที่ชื่อว่า BlackBox.jar ซึ่งโปรแกรมนี้จะเป็นโปรแกรมที่ใช้ในการตรวจสอบว่าการติดตั้ง Java Communications API เป็นไปอย่างเสร็จสมบูรณ์และสามารถติดต่อสื่อสารกับคอมพอร์ตได้ ในขั้นแรกให้ทำการเพิ่ม BlackBox.jar เข้าไปใน CLASSPATH จากนั้นใช้คำสั่ง C:\java BlackBox ซึ่งผลการรันจะได้ตามรูปที่ 2.1



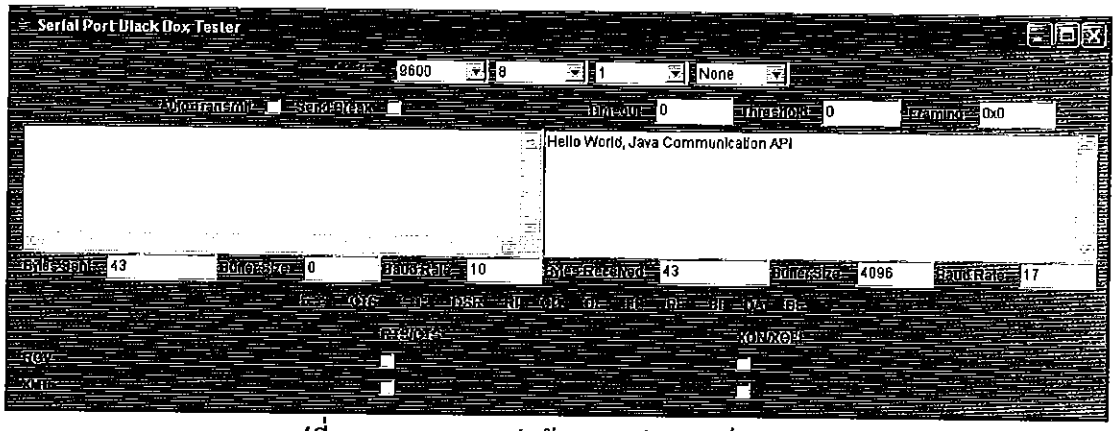
รูปที่ 2.1 ผลการรัน BlackBox.jar

7. เมื่อโปรแกรมรันขึ้นมาแล้ว ถ้าจะทำทดสอบการสื่อสารข้อมูลผ่านคอมพอร์ตก็ให้ไปต่อสายที่พอร์ตอนุกรมตามชื่อพอร์ตที่แสดงขึ้นมา โดยการต่อขาที่ 2 (Received Data) เข้ากับ ขาที่ 3 (Transmitted Data) ตามรูปที่ 2.2



รูปที่ 2.2 การติดตั้งด้านพอร์ตอนุกรม

8. จากนั้นพิมพ์ข้อความที่ช่องกรอกข้อความที่เขียนว่า "Type here" ผลที่ได้คือข้อความที่พิมพ์จะถูกส่งออกไปทางขาที่ 3 ของพอร์ตอนุกรมและรับข้อความที่พิมพ์เข้าทางขาที่ 2 ของพอร์ตอนุกรม แล้วแสดงข้อความที่กรอบข้อความทางด้านซ้าย ดังรูปที่ 2.3



รูปที่ 2.3 ผลจากการส่งข้อความผ่านพอร์ตอนุกรม

การใช้งาน Java Communication API ในโครงงานนี้ ก็จะเป็นการเขียนโปรแกรมเพื่อติดต่อสื่อสารกับไมโครคอนโทลเลอร์ผ่านทางพอร์ตอนุกรม เพื่อใช้ในการควบคุมอุปกรณ์ต่าง ๆ ต่อไป

2.2 Java Network Programming

คอมพิวเตอร์เน็ตเวิร์ค (Computer Network) คือ กลุ่มเครือข่ายของคอมพิวเตอร์หลายเครื่องที่เชื่อมต่อกันด้วยตัวกลางบางอย่าง ทำให้สามารถรับส่งข้อมูลระหว่างกันได้ คอมพิวเตอร์ทีอยู่ในเครือข่ายเดียวกัน อาจมีโครงสร้างและระบบปฏิบัติการแตกต่างกันได้ แต่สามารถติดต่อกันได้เนื่องจากใช้โปรโตคอล (Protocol) ในการติดต่อสื่อสารเดียวกัน โดยที่โปรโตคอล คือ กฎเกณฑ์ที่กำหนดขั้นตอนในการติดต่อ แลกเปลี่ยนข้อมูลระหว่างกัน รวมทั้งรูปแบบของข้อมูลที่ถูกส่งไปมา โดยปกติเรามักใช้ โปรโตคอลที่เป็นที่นิยมอยู่แล้ว ซึ่งมีกฎเกณฑ์เป็นมาตรฐานคนทั่วไปรู้จัก อย่างเช่น TCP , UDP และ HTTP เป็นต้น เพื่อช่วยให้การพัฒนาโปรแกรมเป็นอิสระจากภาษา ระบบปฏิบัติการ และอุปกรณ์สื่อสารที่ใช้

ในการนำส่งข้อมูลจากเครื่องหนึ่งไปสู่เครื่องหนึ่ง ระบบเครือข่ายจะส่งข้อมูลนั้นออกไปเป็นส่วนย่อย ๆ ทีละส่วนซึ่งเรียกว่า Packet ด้านเครื่องที่เป็นผู้ส่งจะส่ง Packet ผ่านชั้น (Layer) ของโปรแกรมหลายชั้น เพื่อทำการเพิ่มเติมและปรับเปลี่ยนข้อมูลให้ Packet นั้นสามารถเดินทางผ่านทางตัวกลางไปสู่ผู้รับได้ ซึ่งอาจอยู่ในรูปสัญญาณไฟฟ้า แสง หรือคลื่นแม่เหล็กไฟฟ้า ขึ้นกับตัวกลาง เมื่อไปถึงเครื่องที่เป็นผู้รับแล้วฮาร์ดแวร์ทีเป็นโปรโตคอล ชั้นล่างสุด ได้ข้อมูลจากตัวกลางมาแล้ว ก็จะส่งข้อมูลนั้นผ่านชั้นของโปรแกรมหายชั้น เพื่อทำการเปลี่ยนคืนกลับเป็น Packet ดังเดิม

มีผู้ออกแบบชั้นของ protocol layers ไว้หลายแบบ เช่น Open System Interconnect (OSI) แบ่งโปรโตคอล เป็น 7 ชั้นคือ Application, Presentation, Session, Transport, Network, Data-link และ Physical ส่วน Internet Protocol (IP) แบ่ง protocal layers ไว้เป็น 5 ชั้น คือ Application, Transport, Network, Data-Link และ Physical

เป็นที่ทราบกันดีแล้วว่าภาษาจาวา (Java) เป็นภาษาที่ไม่ขึ้นกับแพลตฟอร์ม (Platform Independent) แล้วผู้เขียนโปรแกรมยังไม่จำเป็นเข้าใจรายละเอียดเกี่ยวกับ protocol layers, hardware หรือตัวกลางที่ใช้ในการสื่อสาร เนื่องจากกลไกที่จำเป็นในการส่งข้อมูลผ่านระบบเครือข่ายถูกซ่อนรายละเอียด (encapsulated) อยู่ภายในคลาสของ java.net package [3-4]

2.2.1 InetAddress

ในการส่งข้อมูลระหว่างเครื่อง เราต้องระบุ (identify) เครื่องที่จะเป็นผู้รับและผู้ส่ง เราเรียกคอมพิวเตอร์แต่ละเครื่องที่ต่ออยู่ในระบบอินเตอร์เน็ตว่า Host โดยปกติทุกโฮสต์ (Host) จะถูกกำหนดเลขที่ประจำตัวเรียกว่า Internet address (หรือ IP address) มีขนาด 4 ไบต์ เพื่อความสะดวกเราจะเขียน IP Address เป็นเลขจำนวนเต็มบวก 4 ตัว แบ่งแยกกันด้วยจุด "." โดยเลขจำนวนเต็มตัวหนึ่งสำหรับค่าในหนึ่งไบต์จึงมีค่าได้ตั้งแต่ 0 ถึง 255 ตัวอย่างเช่น 192.168.167.60 โดยปกติบริษัทผู้ให้บริการอินเตอร์เน็ต (Internet Service Provider) จะเป็นผู้กำหนด IP Address ให้แก่เครื่องของเรา หรือถ้าเครื่องเราอยู่ในระบบเครือข่ายของมหาวิทยาลัย บริษัท หรือองค์กรขนาดใหญ่ก็จะมี System Administator เป็นบุคคลที่มีหน้าที่ดูแลรักษาระบบเครือข่าย และยังไม่ได้กำหนด IP Address ไว้ เราก็ยังสามารถอ้างถึงเครื่องของเราเองได้โดยใช้ Loopback IP Address คือ 127.0.0.1 ซึ่งเป็นเบอร์ที่จะถูกตีความเป็นเครื่องที่กำลังทำงานโดยปกติ (default)

IP Address เป็นตัวเลขจึงเหมาะสำหรับถูกจัดการโดยเครื่องคอมพิวเตอร์ แต่ผู้ใช้ที่เป็นมนุษย์จะมีปัญหาในการจดจำตัวเลข ดังนั้นเพื่อช่วยให้เราสามารถระบุเครื่องคอมพิวเตอร์ได้ด้วยชื่อที่จำง่ายกกว่า จึงมีการกำหนด Domain Name ไว้คู่กับ IP Address เช่น 192.168.68.2 มี Domain Name เป็น ecpe.nu.ac.th ทำให้เราสามารถอ้างถึงเครื่องนี้โดยใช้ 192.168.68.2 หรือ ecpe.nu.ac.th ก็ได้ แต่เราจะใช้ Domain Name ก็ต้องมี DNS (Domain Name System) ซึ่งเป็น server ที่ให้บริการหารสอบถามว่า Domain Name หนึ่งมี IP Address เป็นอะไร นั่นหมายความว่า DNS จะเก็บข้อมูลเกี่ยวกับ Domain Name และ IP Address ของเครื่องที่อยู่ในเครือข่ายขององค์กรไว้ รวมทั้งบางเครื่องที่ถูกอ้างถึงบ่อย ๆ เมื่อโปรแกรมถามมาที่ DNS ก็จะเริ่มค้นหา Domain Name ที่มีอยู่ก่อน หากไม่พบ ก็จะทำการร้องขอไปยัง DNS ที่อยู่ในเครือข่ายใกล้เคียง ให้ช่วยค้นหาให้ได้

ใน java.net package มีคลาส InetAddress สำหรับเก็บแสดง IP Address ซึ่งใช้ได้กับทั้ง TCP และ UDP protocols โดยปกติใน Instance ของคลาส InetAddress จะมีข้อมูลเกี่ยวกับ IP Address และอาจจะมี Domain Name ของ IP Address นั้นด้วยก็ได้ ขึ้นกับว่าอินสแตนซ์ (instance) นั้นถูกสร้างขึ้นโดยมี Domain Name กำหนดให้หรือไม่ การที่โปรแกรมภาษา Java ใช้ InetAddress ในการอ้างถึงเครื่องหนึ่ง ๆ ก็เพื่อให้โปรแกรมไม่ขึ้นกับแพลตฟอร์ม ซึ่งอาจจะมีวิธีในการเก็บแสดง IP Address แตกต่างกัน และในอนาคตคงต้องมีการเปลี่ยนแปลง IP Address ให้สามารถรองรับจำนวนเครื่องคอมพิวเตอร์ที่เพิ่มมากขึ้น เมื่อถึงเวลานั้นคลาส InetAddress จะถูกเปลี่ยนแปลงให้สนับสนุนมาตรฐานใหม่นั้น โดยที่โปรแกรมของเรานั้นไม่ต้องถูกเปลี่ยนแปลง อย่างไรก็ตาม

ในตอนนี้เริ่มมีการทดลองใช้มาตรฐาน IPv6 จึงมีการเพิ่มคลาส Inet4Address สำหรับ IPv4 และคลาส Inet6Address สำหรับ IPv6

คลาส InetAddress ไม่มี public constuctor แต่มี public static factory methods สำหรับสร้าง instance ของคลาส อย่างเช่น

public static InetAddress getLocalHost() throws UnknownHostException;

ซึ่งจะให้ InetAddress ของเครื่องที่ใช้ทำงาน หลังจากนั้น เราอาจใช้

public String getHostName();

public String getHostAddress();

เรียกขอ Domain Name และ IP ของ InetAddress นั้นออกมาในรูปของ String

2.2.2 Socket

ในการส่งข้อมูลจากเครื่องคอมพิวเตอร์หนึ่งไปสู่เครื่องหนหึ่งในระบบเครือข่าย โปรแกรมด้านผู้ส่ง (Sender) จะต้องนำข้อมูลที่ถูกส่งไปนั้น มาตัดออกเป็นส่วนย่อย ๆ แล้วบรรจุลงใน packet แต่ละ packet จะมีส่วนประกอบสองส่วน ส่วนแรกกคือ header เป็นข้อมูลเกี่ยวกับ Address และ port ของผู้รับและผู้ส่ง รวมทั้งข้อมูลเท่าที่จำเป็นในการนำ packet มาประกอบกันเป็นข้อมูลดั้งเดิม อีกส่วนเรียกว่า payload คือข้อมูลย่อยที่จะถูกส่งไปนั่นเอง ผู้เขียนโปรแกรมต้องเขียนโปรแกรมทั้งด้านผู้ส่งและผู้รับ ที่ด้านผู้ส่ง ต้องทราบวิธีการสร้าง packet และส่ง packet ผ่านชั้นของโปรแกรมลงไปสู่ชั้นกลางเพื่อเดินทางไปในระบบเครือข่าย ที่ด้านผู้รับ ต้องทราบวิธีรับ packet จากระบบเครือข่าย ขึ้นมาประกอบเป็นลำดับที่ถูกต้อง แล้วจึงดึงข้อมูลออกมา จะเห็นว่า การเขียนโปรแกรมรับส่งข้อมูลผ่านระบบเครือข่ายเป็นเรื่องยุ่งยากมาก และต้องใช้ผู้เชี่ยวชาญในการสร้างโปรแกรมนี้

จนกระทั่งใน Berkeley UNIX มีการเสนอ Socket ขึ้น เพื่อใช้ในการเขียนโปรแกรมส่งข้อมูลผ่านระบบเครือข่าย โดยที่ Socket ซ่อนรายละเอียดเกี่ยวกับ packet รวมทั้งวิธีการติดต่อกับระบบเครือข่ายไปจากผู้เขียนโปรแกรมแล้วให้ผู้เขียนโปรแกรม สามารถเขียนโปรแกรมส่งและรับข้อมูลผ่านระบบเครือข่ายได้เหมือนกับการเขียนและอ่านข้อมูลจาก streams เมื่อมี Soket แล้ว การเขียนโปรแกรมส่งข้อมูลผ่านระบบเครือข่ายง่ายขึ้นอย่างมาก จึงแพร่หลายจาก UNIX ไปสู่ระบบปฏิบัติการอื่น ๆ เช่น Windows และ Macintosh รวมทั้งยังอาจเขียนโปรแกรมเกี่ยวกับ Soket ด้วยภาษาต่าง ๆ โดยเริ่มจากภาษา C ไปสู่ C++ , VB และ Java ซึ่งรายละเอียดการใช้งาน Socket ในภาษา Java มีดังต่อไปนี้

ใน package java.net มีคลาส Socket สำหรับสร้าง Socket ซึ่งมีกลไกสำหรับ

1. สร้างการติดต่อไปสู่เครื่องเป้าหมายที่ port หนึ่ง โดยเครื่องเป้าหมายอาจเป็นผู้รับหรือส่งก็ได้

2. สร้าง streams สำหรับอ่านหรือเขียนข้อมูลไปที่เครื่องเป้าหมาย

คลาส Socket มี constructor หลายตัว เช่น

public Socket (String, int) throws UnknownHostException, IOException;

public Socket (InetAddress, int) throws IOException;

จะสร้าง socket ที่ไปยังเครื่อง host ที่ระบุโดยพารามิเตอร์ตัวแรกที่เป็น String หรือ InetAddress โดยติดต่อเข้าไปที่ port เลขที่ระบุด้วยพารามิเตอร์ตัวที่สองที่เป็น int

และยังมี socket constructors อีกสองตัวคือ

public Socket (String, int, InetAddress) throws IOException;

public Socket (InetAddress, int, InetAddress, int) throws IOException;

ทั้งคู่มีพารามิเตอร์สองตัวแรกระบุ host และ port ที่เครื่องเป้าหมาย ส่วนพารามิเตอร์ที่เป็น InetAddress และ int ต่อท้ายเพิ่มขึ้นมาเพื่อระบุ host และ port ทางด้านผู้สร้าง socket เป็นเครื่องเริ่มต้น

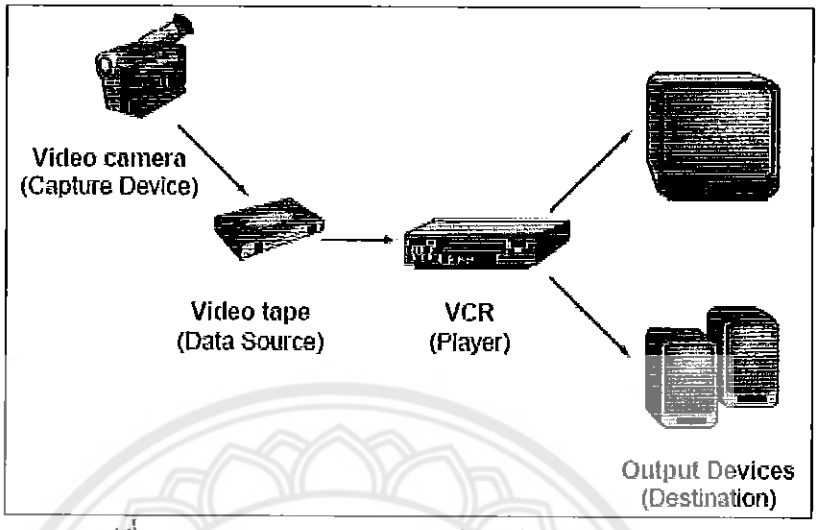
เมื่อสร้าง Socket จะมีการติดต่อไปที่เครื่องเป้าหมายทันที โดยใช้ TCP/IP หลังจากเกิดการติดต่อแล้ว เครื่องเริ่มต้นจะสามารถขอ input stream และ output stream จาก Socket เพื่อทำการอ่านและเขียนไปที่เครื่องเป้าหมาย ดังนั้นต้องมีข้อตกลงระหว่างเครื่องทั้งสองว่า มีกติกาในการรับส่งข้อมูลอย่างไร เช่น ใครจะเขียนหรืออ่านก่อน เป็นต้น และข้อมูลต้องมีรูปแบบใด เราเรียกข้อตกลงนี้ว่า protocol เรากำหนดให้แต่ละ port ต้องมี protocol ที่แน่นอน เพื่อให้เราสามารถติดต่อเข้าไปที่ port นั้นได้ ถ้าเข้าใจ protocol แต่โปรแกรมนั้นอาจถูกเขียนด้วยภาษาใด หรือทำงานบนแพลตฟอร์ม (platform) ใดก็ได้ ค่าของ port ต้องเป็นเลขจำนวนเต็ม มีค่าได้ตั้งแต่ 1 ถึง 65,535 แต่ในเครื่องทั่วไป port เบอร์ 1 ถึง 1024 ถูกกำหนดเป็น port ที่มี protocol มาตรฐานไว้แล้ว เช่น เบอร์ 21 เป็น FTP , 23 เป็น Telnet, 25 เป็น SMTP, 80 เป็น HTTP เป็นต้นและยังมีโปรแกรมของผู้ผลิตบางรายนำ port ที่ค่าสูงกว่า 1024 ไปใช้แล้ว ดังนั้นหากเราจะเปิด port เพื่อทำการติดต่อกันเอง ควรเลือกค่าที่สูงกว่า 1024 ขึ้นไป และตรวจสอบว่ามีโปรแกรมอื่นที่ใช้ port เบอร์นั้นอยู่หรือไม่

2.3 Java Media Framework (JMF)

ปัจจุบัน Multimedia ได้เข้ามามีบทบาทสำคัญในการนำเสนอข้อมูลมากยิ่งขึ้นภาษา Java เองก็ได้เพิ่มความสามารถในการนำเสนอข้อมูลด้วย Multimedia ให้มากยิ่งขึ้นเช่นกัน โดยได้มีการสร้าง Java API ขึ้นมาใหม่เรียกว่า "Java Media Framework" [5] หรือ JMF ปัจจุบันเป็นเวอร์ชั่น 2.1.1e แต่ JMF ไม่ได้รวมอยู่ใน JDK ดังนั้นหากต้องการใช้งาน JMF ก็ต้องไปดาวน์โหลดเพื่อนำมาใช้งานได้ที่ http://java.sun.com หลังจากทำการติดตั้ง JMF เสร็จแล้วก็จะมี package เพิ่มขึ้นมาคือ package "javax.media" รวมถึงเครื่องมือสำเร็จรูปที่สามารถเรียกใช้ได้ทันทีคือ Java Media Studio หรือเรียกสั้นๆ ว่า JMStudio

การทำงานของ JMF จะเริ่มจากการจับภาพและเสียงจาก capture device ต่างๆ ซึ่งเปรียบเสมือนกับกล้องวิดีโอ และนำข้อมูลที่ได้มาเก็บไว้ใน Data Source ที่เปรียบเสมือนกับม้วน

วีดีโอ จากนั้น DataSource จะถูกนำไปประมวลผลและแสดงออกทาง Output Device ต่างๆโดย Player หรือ Processor เสมือนเป็นเครื่องเล่นวิดีโอนั่นเอง ดังแสดงในรูปที่ 2.4



รูปที่ 2.4 การทำงานของ JMF เปรียบเทียบกับระบบวิดีโอ

ในการทำงานทั้ง 3 ส่วนนี้ จะมี Manager เป็นตัวเชื่อมโยงการทำงาน ส่วนประกอบต่างๆของ JMF อธิบายได้ดังนี้

2.3.1 Capture

ในการจับภาพ (Capture) นั้น สามารถนำเอาอุปกรณ์ที่ใช้บันทึกสัญญาณภาพและเสียง (Multimedia Capturing Device) มาเป็นแหล่งข้อมูลสำหรับ JMF ได้ เช่น ไมโครโฟนที่ใช้ในการบันทึกเสียง หรือกล้องวิดีโอที่ต่อกับการ์ดบันทึกภาพ ขั้นตอนในการจับภาพ มีดังนี้

1. นำ Capture Device ทั้งหมดที่มีมาเก็บไว้โดยใช้ Class CaptureDeviceManager และระบุตัวที่ต้องการใช้

2. ดึงเอา Object ของ Class CaptureDeviceInfo จาก Device นั้น ซึ่ง Object นี้จะเก็บข้อมูลต่างๆของ Device

3. ดึงเอา MediaLocator ซึ่งหมายถึงที่อยู่ของ Device นั้น จาก object ของ Class CaptureDeviceInfo และนำเอา MediaLocator นั้น ไปสร้าง DataSource

4. สร้าง Player หรือ Processor จาก DataSource ที่ได้

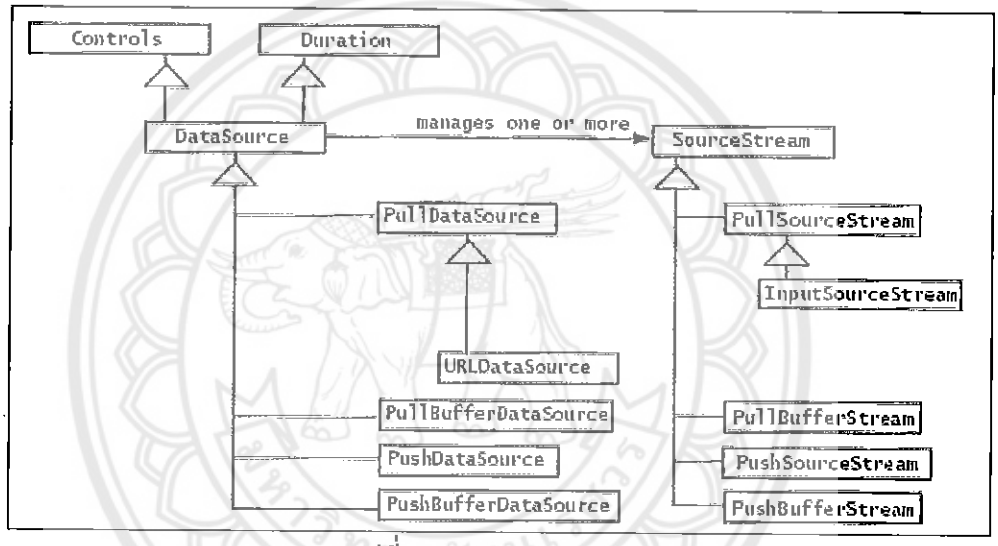
5. สั่งให้ Player หรือ Processor เริ่มการ Capture ถ้าใช้ DataSource กับ Player ในการ Capture จะสามารถแสดงผลได้อย่างเดียว ถ้าต้องการนำข้อมูลที่ได้ไปประมวลผล หรือเก็บไว้ต้องใช้ Processor

2.3.2 Data Source

JMF Media Player ใช้ DataSource ในการจัดการส่งข้อมูล โดย DataSource จะเก็บตำแหน่งของข้อมูล และวิธีในการส่งไว้ นอกจากนี้ DataSource ยังจัดการกับ SourceStream Object ซึ่งเป็น Stream ของ Media ที่เข้ามา Standard Data Source จะใช้ Byte Array ในการเก็บข้อมูลเพื่อใช้ในการส่งข้อมูล เรียกว่า Unit Of Transfer ส่วน Buffer Data Source ใช้ Buffer Object เป็น Unit Of Transfer โดย JMF ได้แบ่ง DataSource ออกเป็น 2 ประเภทตามการส่งข้อมูล

1. Pull Data-Source จะทำการส่งข้อมูลก็ต่อเมื่อ มีการสั่งให้ส่งข้อมูลมาให้ซึ่ง Client จะเป็นผู้ควบคุมทั้งหมด

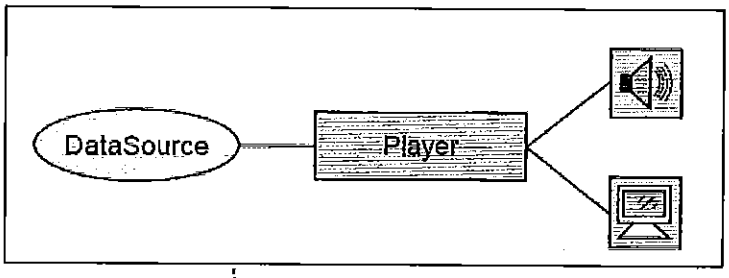
2. Push Data-Source จะทำการส่งข้อมูลมาเรื่อยๆ โดย Client ไม่สามารถควบคุมได้



รูปที่ 2.5 JMF Data Model

2.3.3 Players และ Processors

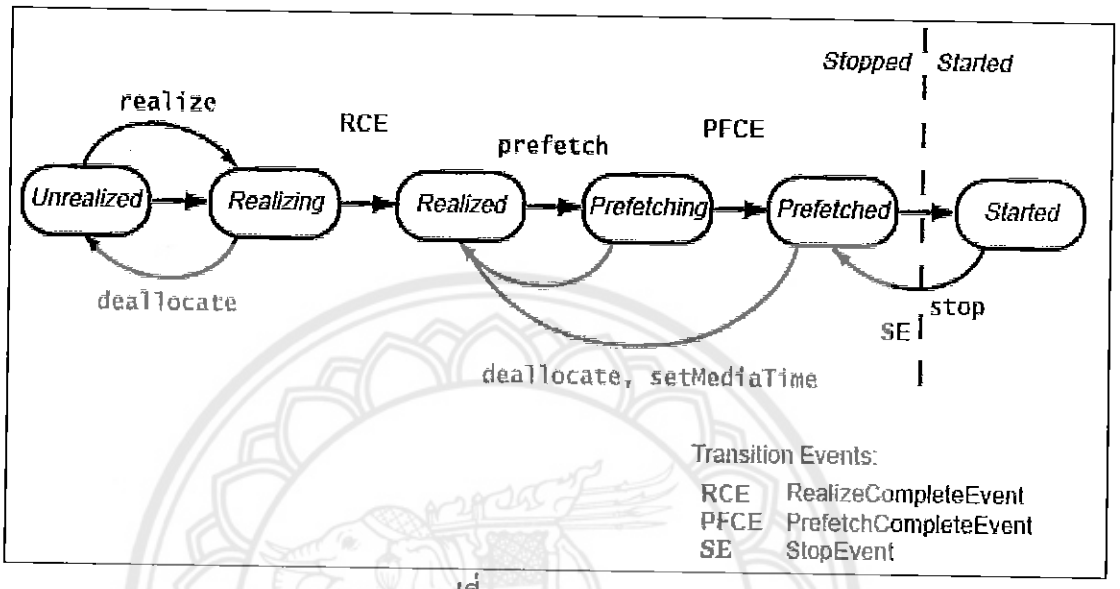
Player เป็นส่วนที่ใช้ในการประมวลผล Input Stream ของ Media Data และแสดงผล โดย DataSource เป็นตัวจ่าย Input Stream ให้แก่ Player ส่วนการแสดงผลนั้นขึ้นอยู่กับชนิดของ Media เช่น ถ้า Media เป็น ชนิดวิดีโอก็จะแสดงออกทางจอภาพ



รูปที่ 2.6 การทำงานของ Player

2.3.3.1 Player States

ในการทำงานของ Player นั้นไม่สามารถประมวลผลและแสดงผลได้ทันที แต่ต้องผ่านสถานะต่างๆ 6 สถานะ คือ Unrealized, Realizing, Realized, Prefetching, Prefetched และ Started หรือแบ่งเป็น 2 สถานะใหญ่คือ Stopped และ Started



รูปที่ 2.7 Player States

สถานะทั้ง 6 สถานะมีหน้าที่ และการทำงานดังนี้

1. Unrealized: Player จะอยู่ในสถานะนี้ทันทีที่มีการสร้าง Player แต่จะยังไม่ทราบข้อมูลเกี่ยวกับ media

2. Realizing: Player อยู่ในสถานะนี้เมื่อมีการเรียก realize ซึ่งในสถานะนี้ resource ต่างๆ ที่จำเป็นจะถูกกำหนดขึ้น

3. Realized: Player จะเปลี่ยนสถานะเป็น Realized เมื่อการทำงานในสถานะ Realizing สิ้นสุดลง โดยในสถานะนี้ Player จะทราบว่า Resource ใดบ้างที่ต้องการ และทราบชนิดข้อมูลที่จะใช้ในการแสดงผล

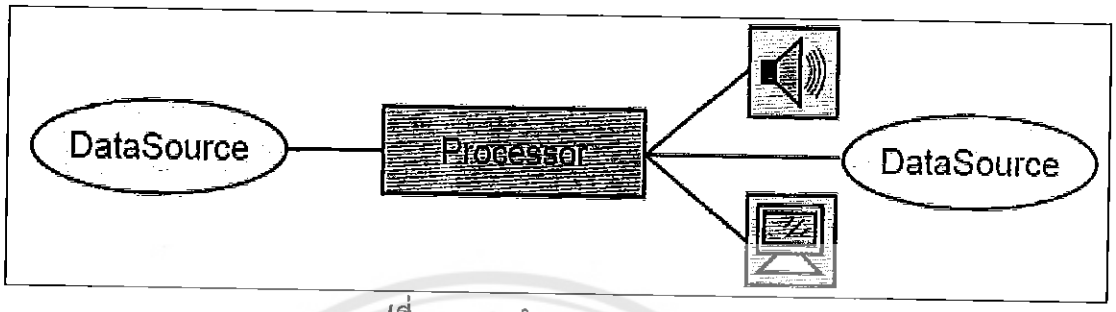
4. Prefetching: Player จะอยู่ในสถานะนี้เมื่อมีการเรียก Prefetch (ก่อนหน้านี้ Player ต้องอยู่ในสถานะ Realized) โดยในสถานะนี้เป็นการเตรียมตัวของ Player เพื่อแสดงผลของ Media ที่ได้รับจาก DataSource

5. Perfected: Player จะเปลี่ยนสถานะเป็น Prefetched เมื่อการทำงานในสถาน Prefetching สิ้นสุดลง โดยสถานะนี้ Player พร้อมจะเริ่มแสดงผลแล้ว เพียงแต่รอคำสั่งให้เริ่มแสดงผลเท่านั้น

6. Started: Player จะเข้าสู่สถานะนี้เมื่อเรียก Start และเริ่มแสดงผล Media Data

2.3.3.2 Processor

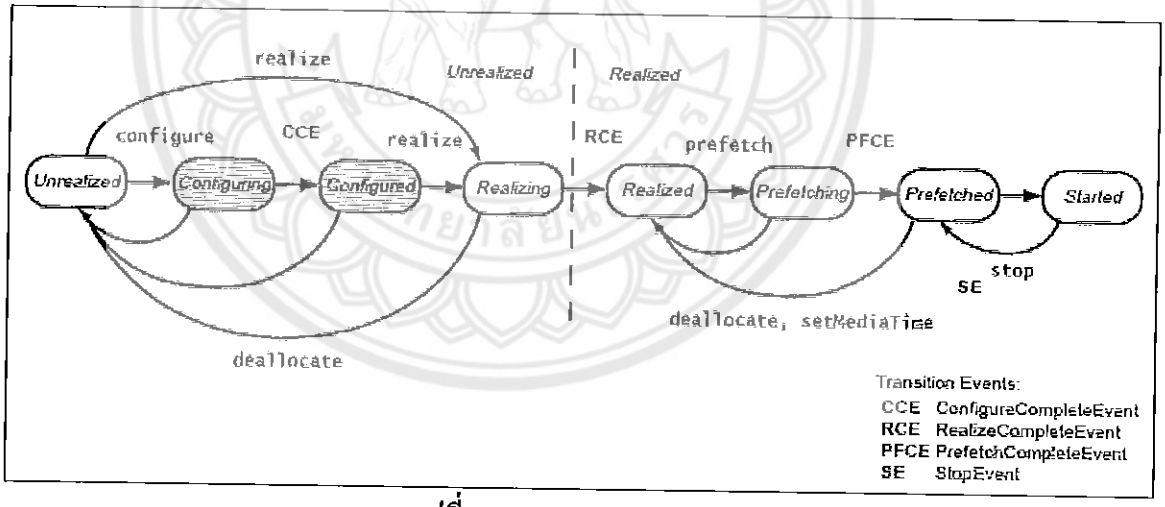
Processor คือ Player ประเภทหนึ่งซึ่งสามารถนำเอาท์พุตที่ได้ ส่งออกทาง Output Deviceเหมือนกับ Player หรือส่งต่อให้กับ DataSource ซึ่งจะเป็นอินพุตให้กับ Player หรือ Processor ตัวอื่นๆต่อไป หรือส่งไปยังที่อื่นๆได้ เช่น ส่งไปยังไฟล์



รูปที่ 2.8 การทำงานของ Processor

2.3.3.3 Processor States

Processor จะเพิ่มสถานะ Standby อีก 2 สถานะ จาก Player States คือ Configuring และConfigured ก่อนที่ Processor จะเข้าสู่สถานะ Realizing



รูปที่ 2.9 Processor States

1. Configuring: Processor จะอยู่ในสถานะนี้เมื่อเรียก Configure โดยสถานะนี้ Processor จะทำการติดต่อกับ DataSource จากนั้นจะทำการเลือก Input Stream และดึงเอาข้อมูลรูปแบบของ Input Data นั้นออกมา

2. Configured: Processor จะเปลี่ยนจากสถานะ Configuring มาเป็นสถานะนี้เมื่อ การติดต่อกับ DataSource สิ้นสุดลง และ Data Format ถูกเลือกแล้ว

2.3.4 DataSink

DataSink เป็นอินเตอร์เฟสหลักสำหรับอ็อบเจคไว้อ่านข้อมูลสื่อประสม (Media) ที่ถูกส่งมาโดย DataSource และส่งข้อมูลสื่อประสม (Media) ไปยังปลายทาง

2.3.5 Format

Format เป็นตัวที่ไว้กำหนดรูปแบบของข้อมูลสื่อประสม ว่าใช้การเข้ารหัสการจัดเก็บข้อมูลสื่อประสมในมาตรฐานใด ซึ่งคลาสลูกของ Format ก็จะประกอบไปด้วยคลาส AudioFormat และ VideoFormat ซึ่ง VideoFormat ก็จะมีคลาสลูกที่เป็นตัวจัดการการเข้ารหัสในรูปแบบต่าง ๆ ดังนี้

- H261Format
- H263Format
- IndexedColorFormat
- JPEGFormat
- RPGFormat
- YUVFormat

2.3.5 Managers

ใน JMF API โปรแกรมอาจจะมีการทำงานในส่วนต่างๆ หลายส่วน เช่น Capture, Process และการ Present Time-Based Media ซึ่งการทำงานทั้งหมดนี้ต้องมี Object ที่เรียกว่า Managers เชื่อมการทำงานทั้งหมดเข้าด้วยกัน ใน JMF มี Managers ทั้งหมด 4 แบบคือ

1. Manager เป็นเครื่องมือในการสร้าง Resource ต่างๆ เช่น Players, Processors, DataSources และ DataSinks
2. PackageManager ใช้จัดการกับแพ็กเก็ตต่างๆ ที่ต้องการใช้กับโปแกรม
3. CaptureDeviceManager ใช้ดึงข้อมูลจาก Capture Devices ที่ใช้งานได้
4. PlugInManagerใช้ค้นหา Plugin ที่ต้องการและทำให้ Plugin สามารถใช้กับโปรแกรมได้

2.4 โปรโตคอลทีซีพี/ไอพี [6]

ปัจจุบันโปรโตคอล TCP/IP (Transmission Control Protocol/Internet Protocol) เป็นโปรโตคอลที่นิยมใช้ในเครือข่ายมากที่สุด เหตุผลหนึ่งที่โปรโตคอลชุดนี้เป็นที่นิยมมาก เนื่องจากบริษัทที่ผลิตอุปกรณ์หรือซอฟต์แวร์ของเครือข่ายนำมาใช้เป็นมาตรฐาน และอีกอย่าง TCP/IP เป็นโปรโตคอลพื้นฐานของเครือข่ายอินเตอร์เน็ต (Internet) ซึ่งเป็นเครือข่ายที่ใหญ่ที่สุดในโลก และเป็นเครือข่ายที่ทำให้คอมพิวเตอร์กลายเป็นส่วนที่สำคัญในชีวิตประจำวันของเราในปัจจุบัน ดังนั้น TCP/IP จึงกลายเป็นโปรโตคอลมาตรฐานที่ใช้ในองค์กรธุรกิจและรัฐบาล

ชุดโปรโตคอล TCP/IP ได้ถูกพัฒนามาแล้วกว่า 30 ปี ซึ่งเริ่มจากการวิจัยที่สนับสนุนโดยกระทรวงกลาโหมสหรัฐฯ จุดประสงค์ของการวิจัยก็เพื่อใช้ในการเชื่อมต่อคอมพิวเตอร์ที่ต่างระบบกัน ให้สามารถสื่อสารกันผ่านเครือข่ายได้ ซึ่งสมัยนั้นคอมพิวเตอร์ที่เป็นแพลตฟอร์มเดียวกันเท่านั้นจึงจะสามารถสื่อสารกันผ่านเครือข่ายได้ ดังนั้นจึงได้พัฒนาโปรโตคอลชุดนี้ขึ้น เนื่องจากขั้นตอนการสื่อสารระหว่างคอมพิวเตอร์เป็นสิ่งที่ค่อนข้างซับซ้อน ดังนั้นโปรโตคอลจึงแบ่งเป็นชั้นย่อยหรือเลเยอร์ (Layer) เพื่อเป็นการแยกการทำงานของแอพพลิเคชันของผู้ใช้ออกจากฮาร์ดแวร์ที่ใช้รับส่งข้อมูลผ่านเครือข่าย โปรโตคอลชุดนี้จะมีการจัดรูปแบบที่แตกต่างจากแบบอ้างอิง OSI (Open System Interconnect) เล็กน้อย

เครือข่ายคอมพิวเตอร์ในปัจจุบันประกอบด้วยหลากหลายอุปกรณ์ และฮาร์ดแวร์ที่ผลิตโดยบริษัทต่างๆ แต่ละอุปกรณ์เครือข่ายเหล่านี้มีระบบการสื่อสารข้อมูลที่เหมือนกัน เครือข่ายที่ใช้อุปกรณ์จากหลายบริษัทนี้สามารถทำงานร่วมกันได้ เนื่องจากอุปกรณ์แต่ละชิ้นผลิตตามมาตรฐานที่กำหนดโดยองค์กรกลาง โปรโตคอล TCP/IP เป็นมาตรฐานที่ได้รับความนิยมมากที่สุดสำหรับเครือข่ายในปัจจุบันเนื่องจากหลายเหตุผลดังนี้

1. เป็นโปรโตคอลระบบเปิด (Open System) ที่ไม่มีบริษัทใดบริษัทหนึ่งเป็นเจ้าของลิขสิทธิ์ และข้อกำหนดของทุกโปรโตคอลจะถูกพัฒนาโดยองค์กรสาธารณะและมีพิมพ์ให้ทราบ

2. TCP/IP ถูกออกแบบมาเพื่อให้แพลตฟอร์มต่างกันสามารถสื่อสารกันได้ โปรแกรมบริการต่างๆ เช่น FTP (File Tranfer Protocal) และ Telnet เป็นโปรแกรมที่ไม่ขึ้นต่อระบบ เพียงแต่บริษัทนั้นๆ พัฒนาระบบของตัวเองให้สามารถรองรับ TCP/IP ได้ก็สามารถสื่อสารระบบอื่นได้

3. โปรโตคอล TCP/IP ได้ถูกพิสูจน์แล้วว่าเป็นโปรโตคอลที่แข็งแกร่ง มีประสิทธิภาพสูง และมีอัตราการขยายตัวสูง ด้วยการใช้งานในเครือข่ายอินเตอร์เน็ต ซึ่งเป็นเครือข่ายที่ใหญ่ที่สุดในโลก

4. โปรโตคอล TCP/IP ได้กลายเป็นโปรโตคอลมาตรฐานกลางในการสื่อสารข้อมูลของคอมพิวเตอร์เนื่องจากเป็นภาษาที่ใช้ในระบบอินเตอร์เน็ต

ไม่ว่าจะด้วยเหตุผลอะไรก็แล้วแต่ที่ทำให้โปรโตคอล TCP/IP เป็นโปรโตคอลที่นิยมในปัจจุบัน แต่สิ่งที่สำคัญที่สุดคือ โปรโตคอลนี้ได้กลายเป็นมาตรฐานเครือข่ายไปโดยปริยาย ดังนั้นเครือข่ายสมัยใหม่จึงจำเป็นที่ต้องสร้างให้สามารถรองรับโปรโตคอลนี้

2.4.1 TCP/IP และแบบอ้างอิง OSI

โครงสร้างของเครือข่ายเป็นสิ่งที่ซับซ้อนมาก และยากต่อการออกแบบและพัฒนาทั้งระบบโดยคนกลุ่มใดกลุ่มหนึ่ง ดังนั้นจึงมีการแบ่งโครงสร้างออกเป็นชั้น หรือเลเยอร์ (Layer) ซึ่งจะช่วยให้ทั้งผู้ผลิตฮาร์ดแวร์และซอฟต์แวร์พัฒนาผลิตภัณฑ์ของตัวเองได้ โดยไม่ต้องกังวลกับส่วนอื่นๆ แต่ยังสามารถทำงานร่วมกันได้แบบอ้างอิง OSI [6] ได้อธิบายสถาปัตยกรรมเครือข่าย โดยแบ่งฟังก์ชันการเคลื่อนย้ายข้อมูลจากคอมพิวเตอร์เครื่องหนึ่งออกเป็น 7 เลเยอร์

การออกแบบชุดโปรโตคอล TCP/IP ก็จะคล้ายๆ กับแบบอ้างอิง OSI คือจะแบ่งออกเป็นเลเยอร์เช่นกัน แต่การออกแบบจะมุ่งเน้นไปที่การเชื่อมต่อระหว่างระบบที่แตกต่างกันมากกว่า ในขณะที่แบบอ้างอิง OSI จะเน้นไปที่การแบ่งการทำงานของโปรโตคอลออกเป็นเลเยอร์ จึงทำให้ OSI มีโครงสร้างที่ดีกว่า ดังนั้นส่วนใหญ่จะนิยมใช้โปรโตคอล OSI เป็นแบบอ้างอิงในการอธิบายการสื่อสารระหว่างคอมพิวเตอร์ในเครือข่าย ในขณะที่ชุดโปรโตคอล TCP/IP เป็นที่นิยมมากกว่าในการนำไปใช้งานจริง

<table>
<tr><th colspan="2">OSI Reference Model</th><th colspan="3">TCP/IP</th></tr>
<tr><td>7</td><td>Application</td><td rowspan="2">Application</td><td colspan="2" rowspan="2">FTP, Telnet, HTTP, SMTP,
SNMP, DNS, etc</td></tr>
<tr><td>6</td><td>Presentation</td></tr>
<tr><td rowspan="2">5</td><td rowspan="2">Session</td><td rowspan="2">Host-to-Host</td><td rowspan="2">TCP</td><td rowspan="2">UDP</td></tr>
<tr><td>4</td><td>Transport</td></tr>
<tr><td rowspan="2">3</td><td rowspan="2">Network</td><td rowspan="2">Internet</td><td>ICMP, IGMP</td><td rowspan="2">ARP, RARP</td></tr>
<tr><td>IP</td></tr>
<tr><td>2</td><td>Data Link</td><td rowspan="2">Network Access</td><td colspan="2" rowspan="2">Not Speccifiled</td></tr>
<tr><td>1</td><td>Physical</td></tr>
</table>

รูปที่ 2.10 เปรียบเทียบแบบอ้างอิง OSI และ TCP/IP

รูปที่ 2.10 แสดงการเปรียบเทียบชั้นของโปรโตคอลระหว่างแบบอ้างอิง OSI และ TCP/IP ซึ่งแบ่งโปรโตคอลออกเป็น 4 เลเยอร์คือ ชั้นประยุกต์ใช้งาน (Application Layer) ชั้นเชื่อมระหว่างโฮสต์ (Host to Host Layer) ชั้นอินเตอร์เน็ต (Internet Layer) และชั้นเข้าใช้เครือข่าย (Network Access Layer) การเปรียบเทียบการทำงานของโปรโตคอล TCP/IP กับแบบอ้างอิง OSI นั้นอาจไม่ตรงมากนัก เพราะมีบางโปรโตคอลของ TCP/IP ที่ทำงานมากกว่าหนึ่งชั้น แต่รูปดังกล่าวก็เปรียบเทียบให้เห็นภาพพอคร่าวๆ

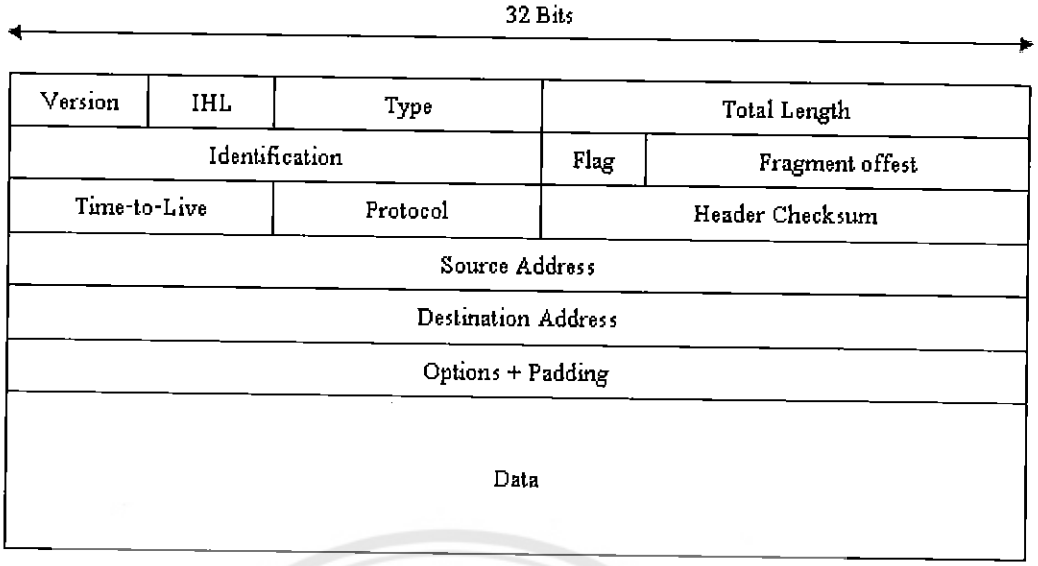
หลักการทำงานของโปรโตคอล TCP/IP สรุปได้คร่าวๆ ดังนี้คือ การสื่อสารจะเริ่มจากแอพพลิเคชันของผู้ใช้ส่งข้อมูลให้กับโปรโตคอลในชั้นแอพพลิเคชัน หลังจากนั้นชั้นแอพพลิเคชันจะเพิ่มข้อมูลส่วนหัวซึ่งจะประกอบไปด้วยชื่อของเครื่องคอมพิวเตอร์ที่ต้องการสื่อสารด้วยและหมายเลขพอร์ตของเครื่องนั้น ข้อมูลก็จะถูกส่งต่อไปยังชั้นเชื่อมต่อโฮสต์ ซึ่งอาจจะใช้โปรโตคอล TCP หรือ UDP ขึ้นอยู่กับแอพพลิเคชันที่ใช้ เมื่อชั้นนี้ได้รับข้อมูลก็จะแบ่งข้อมูลออกเป็นส่วนย่อยๆ ซึ่งแต่ละส่วนจะถูกเพิ่มข้อมูลส่วนหัวเข้าไป ข้อมูลส่วนย่อยๆ นี้จะเรียกว่า "เซ็กเมนต์ (Segment)"

ข้อมูลส่วนหัวของแต่ละเซ็กเมนต์จะถูกเพิ่มเข้าไปอย่างเหมาะสม หลังจากนั้นแต่ละเซ็กเมนต์ก็จะถูกส่งต่อไปให้ชั้นอินเตอร์เน็ต เมื่อข้อมูลมาถึงชั้นนี้ก็จะถูกเพิ่มข้อมูลส่วนหัวให้แต่ละเซ็กเมนต์เช่นกัน ข้อมูลที่เพิ่มเข้าไป เช่น หมายเลข IP ประเภทของโปรโตคอลที่ใช้ (TCP หรือ UDP) และ Checksum เป็นต้น ถ้าข้อมูลที่ส่งมามีการแบ่งย่อยอีกก็จะมีการเพิ่มข้อมูลที่เกี่ยวกับการแบ่งย่อยนี้เพิ่มเข้าไปด้วย ชุดข้อมูลที่อยู่ในชั้นนี้จะเรียกว่า "แพ็กเกจ" หลังจากนั้นแต่ละแพ็กเกจข้อมูลก็ส่งต่อไปให้ชั้นเข้าใช้เครือข่าย เพื่อทำการส่งข้อมูลไปตามช่องสื่อสารต่อไป เมื่อแพ็กเกจเดินทางไปถึงที่หมาย เครื่องปลายทางก็จะทำตามขั้นตอนที่ตรงกันข้ามกับเครื่องที่ส่ง และข้อมูลก็จะถูกส่งผ่านต่อไปให้แอพพลิเคชั่นเพื่อนำข้อมูลไปประมวลผลต่อไป

2.4.2 อินเตอร์เน็ตโปรโตคอล (Internet Protocol :IP)

โปรโตคอล IP ทำหน้าที่เสมือนกับที่ทำการไปรษณีย์ กล่าวคือ โปรโตคอล IP จะทำหน้าที่จัดการเกี่ยวกับการรับส่งแพ็กเกจ หรือบางทีก็เรียกว่า "ดาต้าแกรม (Datagram)" คือหน่วยของข้อมูลที่รับมาจากโปรโตคอลที่อยู่เลเยอร์ที่สูงกว่า เช่น TCP และ UDP ถ้าโฮสต์ปลายทางอยู่คนละเครือข่ายกับโฮสต์ที่ส่งข้อมูล IP จะรับผิดชอบในการจัดเส้นทาง (Routing) ให้แพ็กเกจส่งไปยังเครือข่ายที่โฮสต์นั้นอยู่ ซึ่งในการจัดส่งแพ็กเกจข้ามเครือข่ายนั้น IP จะใช้เราท์เตอร์ (Router) ในการเชื่อมต่อเครือข่ายเหล่านั้นโดยทั่วไปแล้วอุปกรณ์ที่ทำหน้าที่รับส่งข้อมูลระหว่างเครือข่ายจะเรียกว่าเราท์เตอร์ แต่บางทีอุปกรณ์ตัวนี้จะเรียกว่า "เกตเวย์ (Gateway)" ซึ่งทำหน้าที่เป็นเสมือนประตูไปยังเครือข่ายอื่นๆ อย่างไรก็ตามทั้งเราท์เตอร์และเกตเวย์เป็นอุปกรณ์ที่ทำหน้าที่ในเลเยอร์ที่ 3 เหมือนกัน

โปรโตคอล IP เป็นโปรโตคอลที่ให้บริการแบบคอนเน็กชันเลสส์ (Connectionless) ซึ่งทำให้มีความเชื่อถือน้อย เนื่องจากไม่มีการสร้างการเชื่อมต่อก่อนที่จะทำการรับส่งข้อมูล กล่าวคือ ในการรับส่งข้อมูลแต่ละครั้ง โฮสต์ส่งจะไม่ทำการติดต่อโฮสต์ปลายทางเพื่อตกลงเกี่ยวกับการรับส่งข้อมูลก่อน แต่โฮสต์ที่ต้องการส่งข้อมูลจะทำการส่งแพ็กเกจออกไปทันที โดยที่คาดหวังว่าโฮสต์ปลายทางจะได้รับแพ็กเกจนั้นในที่สุด ดังนั้นความเชื่อถือในการส่งข้อมูลจึงมีน้อยเพราะแพ็กเกจอาจสูญหายระหว่างทาง หรือถ้าข้อมูลประกอบด้วยหลายแพ็กเกจ แต่ละแพ็กเกจอาจเดินทางมาถึงปลายทางไม่เป็นลำดับได้ หรือมีการส่งแพ็กเกจซ้ำกันหรือแพ็กเกจส่งถึงล่าช้า การแก้ปัญหานี้จะปล่อยให้เป็นหน้าที่ของโปรโตคอลที่อยู่ในเลเยอร์ที่สูงกว่ารับผิดชอบ



รูปที่ 2.11 ฟอร์แมตของแพ็กเกจ IP

ฟอร์แมตของแพ็กเกจ IP ประกอบด้วยหลายฟิลด์ดังแสดงใน รูปที่ 2.11 ข้อมูลในส่วนหัวของแพ็กเกจ IP มีดังนี้

1. Version (4 บิต) : ข้อมูล 4 บิตแรกจะเป็นข้อมูลที่บอกถึงเวอร์ชันของโปรโตคอล IP ที่ใช้ ซึ่งในปัจจุบันจะใช้เวอร์ชัน 4 หรือเรียกสั้นๆ ว่า IPv4 ในอนาคตอันใกล้อาจจะมีการเปลี่ยนแปลงไปใช้เวอร์ชันใหม่คือ เวอร์ชัน 6 หรือ IPv6 เนื่องจากเวอร์ชัน 4 กำลังมีปัญหาเกี่ยวกับหมายเลข IP ไม่เพียงพอต่อการใช้งาน

2. Internet Header Length หรือ IHL (4 บิต) : เป็นตัวเลขที่บอกความยาวของข้อมูลในส่วนหัว (Header)

3. Type of Service (8 บิต) : ในแต่ละบิตของส่วนข้อมูลนี้จะเป็นธงหรือแฟลก (Flag) ที่แสดงถึงลำดับความสำคัญ (Precedence), ความล่าช้า (Delay), อัตราส่งผ่าน (Throughput) และค่าที่กำหนดความเชื่อถือได้ของแพ็กเกจข้อมูลนี้

4. Total Length (16 บิต) : ข้อมูลส่วนนี้จะบอกถึงความยาวของแพ็กเกจทั้งหมดซึ่งมีหน่วยเป็นไบต์ ซึ่งความยาวของแพ็กเกจนี้เป็นไปได้ตั้งแต่ 576-65,536 ไบต์

5. Identifiler (16บิต) : ถ้าดาต้าแกรมประกอบด้วยหลายแพ็กเกจ หมายเลขนี้จะถูกกำหนดให้กับแต่ละแพ็กเกจย่อย ซึ่งแพ็กเกจแต่ละแพ็กเกจจะมีหมายเลขนี้ที่ไม่ซ้ำกับหมายเลขแพ็กเกจอื่นในช่วงเวลานั้นๆ

6. Flags (3บิต) : เป็นฟิลด์ที่ใช้ในการจัดการเกี่ยวกับการแบ่งข้อมูลเป็นแพ็กเกจย่อย

7. Fragment Offset (13บิต) : เป็นค่าที่บอกจุดเริ่มต้นในส่วนของข้อมูลย่อย (Flagmented Content) ซึ่งเป็นตัวเลขที่บอกว่าแพ็กเกจนี้อยู่ห่างจากจุดเริ่มต้นของดาต้าแกรมทั้งหมดเท่าใด โดยจำนวนนี้มีหน่วยวัดเป็น 64 บิต

8. Time to Live หรือ TTL (8 บิต) : แพ็กเกจจะไหลเวียนอยู่ในเครือข่ายได้ในช่วงเวลาหนึ่งเท่านั้น การกำหนดว่าแพ็กเกจแต่ละแพ็กเกจจะอยู่ได้ในเครือข่ายนานเท่าใดนั้น จะบอกเป็นจำนวนของ Hop หรือจำนวนครั้งที่ผ่านเราท์เตอร์ (Router) ทุกครั้งที่ผ่านเราท์เตอร์ (Router) ค่า TTL จะลดลงทีละหนึ่ง เมื่อค่านี้เป็นศูนย์แพ็กเกจนี้จะถูกละทิ้งไป

9. Protocol (8 บิต) : เป็นข้อมูลที่บอกโปรโตคอลของชั้นที่เหนือกว่า เช่น TCP, UDP, VINES เป็นต้น

10. Header Checksum (16 บิต) : เป็นข้อมูลส่วนที่ใช้ในการตรวจสอบข้อผิดพลาดในส่วนเฮดเดอร์ของแพ็กเกจ ซึ่งเมื่อผ่านอุปกรณ์เครือข่ายแต่ละครั้งก็จะทำการเช็คข้อผิดพลาดทุกครั้งไป

11. Source IP Address (32 บิต) : หมายเลข IP ของเครื่องที่ส่งข้อมูล

12. Destination IP Address (32 บิต) : หมายเลข IP ของเครื่องปลายทาง

13. Padding : เป็นเลข 0 ที่เพิ่มให้กับส่วนหัวของแพ็กเกจเพื่อให้ส่วนหัวมีความยาวที่หารด้วย 32 บิตลงตัว หรือเป็นข้อมูลเกี่ยวกับฟีเจอร์อื่นๆ เช่น การรักษาความปลอดภัย

14. Data : เป็นข้อมูลของโปรโตคอลที่อยู่สูงกว่าซึ่งความยาวจะไม่คงที่

2.4.3 ทีซีพี (Transmission Control Protocol :TCP)

โปรโตคอล TCP (Transmission Control Protocol) เป็นโปรโตคอลที่ให้บริการแบบคอนเน็กชันโอเรียนเต็ด (Connected-Oriented) ซึ่งเป็นการส่งข้อมูลที่เชื่อถือได้ TCP จะส่งข้อมูลทั้งหมดจนสำเร็จ ซึ่งถ้าข้อมูลมีขนาดใหญ่ก็จะถูกแบ่งย่อยเป็นหลายแพ็กเกจ โปรโตคอล TCP จะทำหน้าที่ควบคุมการรับส่งแพ็กเกจข้อมูลย่อยๆเหล่านี้ สำหรับกลไกในการควบคุมการไหลของข้อมูลที่มีรายละเอียดดังนี้

2.4.3.1 การจัดการเกี่ยวกับเซสชั่น

เนื่องจาก TCP เป็นโปรโตคอลที่ให้บริการแบบคอนเน็กชันโอเรียนเต็ด (Connecttion Oreinted) ดังนั้นก่อนที่จะมีการส่งข้อมูลจำเป็นที่ต้องสร้างเซสชั่นเพื่อเชื่อมต่อกับโฮสต์ปลายทางก่อน เซสชั่นเป็นการสร้างการสนทนาอย่างเป็นรูปแบบระหว่างทั้งสองโฮสต์เพื่อใช้สำหรับการกู้คืนข้อมูลเมื่อเกิดข้อผิดพลาดระหว่างการรับส่งข้อมูล ขั้นตอนในการสร้างเซสชั่นนี้จะมีอยู่ 3 ขั้นตอนซึ่งบางทีก็เรียกว่า "ทรีเวย์แฮนด์เช็ค (Three-Way Handshake)"

1. โฮสต์ที่ต้องการส่งข้อมูลจะส่งแพ็กเกจไปยังโฮสต์ปลายทางเพื่อแจ้งให้ทราบว่าต้องการส่งข้อมูล

2. โฮสต์ปลายทางก็จะตอบตกลงกลับมาพร้อมทั้งรหัสที่จะใช้ในการรับส่งข้อมูล

3. โฮสต์ต้นทางก็จะส่งแพ็กเกจพร้อมรหัสที่ได้รับ เพื่อเป็นการยืนยันการเชื่อมต่อหลังจากที่ ได้มีการสร้างเซสชั่นสำเร็จแล้วถึงจะเริ่มขบวนการรับส่งข้อมูลจริงๆ ซึ่งการรับส่งข้อมูลแต่ละครั้งก็จะมีการยืนยันการรับข้อมูลจากโฮสต์ปลายทางทุกครั้ง เมื่อรับข้อมูลเสร็จก็เป็นขั้นตอนการยกเลิกการเซสชั่น ซึ่งจะคล้ายๆกับการสร้างเซสชั่น

2.4.3.2 การควบคุมการไหลและการกู้ข้อมูล

ในแต่ละเซสชั่น (Session)โฮสต์ฝ่ายรับต้องตอบกลับทุกๆ แพ็กเกจที่ได้รับภายในเวลาที่กำหนด เพื่อเป็นการยืนยันการรับข้อมูลทุกๆแพ็กเกจที่ส่ง ฝ่ายรับจะทำการเช็คความถูกต้องของแพ็กเกจข้อมูลทุกครั้ง และแจ้งให้ทราบถึงผลการตรวจสอบนั้น ถ้าฝ่ายส่งไม่ได้รับการตอบรับจากฝ่ายรับภายในเวลาที่กำหนด ฝ่ายรับก็จะคาดเอาว่าแพ็กเกจสูญหายระหว่างทาง ฝ่ายรับก็จะทำการส่งแพ็กเกจนั้นใหม่อีกครั้ง เพื่อให้มั่นใจได้ว่าข้อมูลทุกๆ แพ็กเกจส่งถึงปลายทางอย่างสมบูรณ์ นอกจากนี้การแบ่งข้อมูลขนาดใหญ่ออกเป็นแพ็กเกจย่อยๆ TCP ก็จะกำหนดหมายเลขลำดับ (Sequence Number) ให้แต่ละแพ็กเกจ เพื่อใช้สำหรับการจัดรวมแพ็กเกจย่อยๆ เหล่านั้นให้เป็นข้อมูลเหมือนเดิม นอกจากนี้หมายเลขลำดับยังใช้สำหรับตรวจสอบว่าข้อมูลส่งถึงปลายทางครบทุกแพ็กเกจหรือไม่

กลไกการตอบกลับแพ็กเกจนั้นมีอยู่ 2 ประเภท ประเภทแรกคือ PAR (Positive Acknowledgment and Retransmission) กลไกการทำงานก็คือ เมื่อฝ่ายส่งทำการส่งแพ็กเกจหนึ่ง ก็จะรอการตอบกลับจากฝ่ายรับแล้วค่อยส่งแพ็กเกจต่อไป ถ้าไม่ได้รับการตอบกลับภายในเวลาที่กำหนดก็จะส่งแพ็กเกจนั้นอีกครั้ง ปัญหาของกลไกนี้ก็คือ ถ้าข้อมูลประกอบด้วยหลายๆ แพ็กเกจ และการที่ฝ่ายรับต้องส่งแพ็กเกจตอบรับทุกๆ แพ็กเกจที่ได้รับนั้นอาจเป็นการสิ้นเปลืองแบนด์วิธ (Bandwide) และเป็นขบวนการที่ไร้ประสิทธิภาพ เนื่องจากฝ่ายส่งจะใช้เวลาในการรอมากกว่าการส่งข้อมูล กลไกที่สองจะแก้ปัญหานี้ซึ่งกลไกนี้จะเรียกว่า "สไลดิ้งวินโดว์ (Sliding Window)" กลไกการทำงานคือ ฝ่ายรับสามารถยืนยันการรับแพ็กเกจโดยส่งแพ็กเกจเดียวสำหรับยืนยันการรับหลายแพ็กเกจ วิธีนี้จะช่วยลดจำนวนแพ็กเกจที่ต้องไหลเวียนในเครือข่าย และฝ่ายส่งสามารถส่งทีละหลายๆ แพ็กเกจก่อนที่จะรอการตอบรับ

เมื่อสร้างเซสชั่นเสร็จ ขั้นตอนต่อไปคือการต่อรองเกี่ยวกับขนาดของวินโดว์ (Window Size) ขนาดของวินโดว์คือ จำนวนไบต์ที่ฝ่ายรับได้รับก่อนที่จะทำการตอบกลับ หรือจำนวนไบต์ที่ฝ่ายส่งสามารถส่งได้ก่อนที่จะรอการตอบกลับ การทำงานของสไลดิ้งวินโดว์มีขั้นตอนดังนี้

1. เมื่อโฮสต์ต้องการที่จะส่งข้อมูล TCP จะย้ายข้อมูลไปไว้ในบัฟเฟอร์ที่จะใช้ส่งข้อมูล ซึ่งข้อมูลส่วนนี้จะเรียกว่า "เซกเมนต์ (Segment)" ซึ่งแต่ละเซกเมนต์อาจจะถูกแบ่งย่อยเป็นหลายแพ็กเกจ ซึ่งแต่ละแพ็กเกจก็จะถูกกำหนดหมายเลขลำดับ

2. ทุกๆแพ็กเกจในเซ็กเมนต์จะถูกส่งต่อไปให้โปรโตคอล IP เพื่อทำการส่งไปยังโฮสต์ปลายทาง

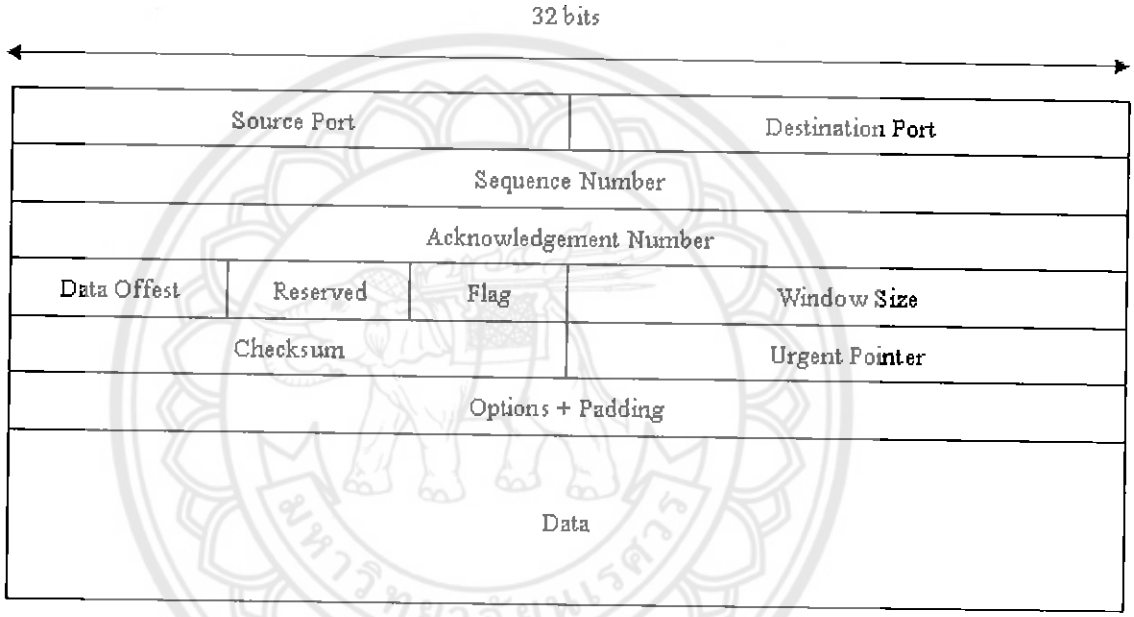
3. เซ็กเมนต์ข้อมูลจะยังคงถูกเก็บไว้ที่บัฟเฟอร์จนกว่าจะได้รับการตอบรับจากโฮสต์ฝ่ายรับก่อน และโฮสต์ฝ่ายส่งจะตั้งเวลาเพื่อรอการตอบกลับ ถ้าโฮสต์ฝ่ายรับไม่ตอบกลับภายในเวลาที่กำหนด ข้อมูลที่อยู่ในบัฟเฟอร์ก็จะถูกส่งใหม่อีกครั้ง

4. เมื่อแพ็กเกจเดินทางมาถึงฝ่ายรับ โฮสต์ฝ่ายรับก็จะใช้หมายเลขลำดับในการเรียบเรียงหมายเลขแพ็กเกจให้ได้เป็นเซ็กเมนต์เหมือนเดิม

5. เมื่อโฮสต์ฝ่ายรับได้รับแพ็กเกจครบและตรวจสอบแล้วไม่มีข้อผิดพลาดใดๆ ก็จะส่งแพ็กเกจตอบกลับไปยังโฮสต์ฝ่ายรับว่าได้รับข้อมูลครบหมดแล้ว

6. เมื่อโฮสต์ฝ่ายรับได้รับการตอบกลับ เซ็กเมนต์ในบัฟเฟอร์ก็จะถูกลบทิ้งไปแล้วทำการส่งเซ็กเมนต์ถัดไปถ้ามี จนกว่าข้อมูลจะถูกส่งทั้งหมด

ขบวนการส่งข้อมูลแบบนี้จะทำให้มั่นใจได้ว่าข้อมูลจะส่งถึงปลายทางอย่างแน่นอนและถูกต้อง ซึ่งการให้บริการแบบนี้จะเรียกว่า "คอนเน็กชันโอเรียนเต็ด (Connection-Oriented)" นั่นเอง



รูปที่ 2.12 ฟอร์แมตข้อมูลของแพ็กเกจ TCP

ข้อมูลในส่วนหัวของโปรโตคอล TCP จะประกอบด้วยข้อมูลมากที่สุด 20 ไบต์ และประกอบด้วยส่วนต่างๆ ดังแสดงในรูป 2.12 ซึ่งแต่ละฟิลด์มีความหมายดังนี้

1. TCP Source Port (16 บิต) : ส่วนนี้จะเป็นหมายเลขพอร์ตที่เป็นจุดเริ่มต้นการสื่อสาร หมายเลขพอร์ตเมื่อรวมกับหมายเลข IP จะเป็นที่อยู่ของการส่งข้อมูลกลับ

2. TCP Destination Port (16 บิต) : เป็นหมายเลขพอร์ตของเครื่องรับ ซึ่งพอร์ตนี้จะเป็นพอร์ตที่ใช้เชื่อมต่อกับแอพพลิเคชันที่จะนำข้อมูลที่จะส่งไปให้นี้ไปโพรเซสต่อไป

3. TCP Sequence Number (32 บิต) : เป็นหมายเลขลำดับแพ็กเกจที่จะใช้ โดยฝั่งเครื่องรับในการเรียงข้อมูลให้อยู่ในรูปแบบเดิม ในการส่งข้อมูลผ่านเครือข่ายที่สลับซับซ้อนนั้นแพ็กเกจแต่ละชุดอาจจะถูกส่งไปบนเส้นทางที่ต่างกัน ดังนั้นจึงเป็นไปได้ที่แพ็กเกจจะเดินทางมาถึงปลายทางไม่เป็นไปตามลำดับที่ส่ง หมายเลขนี้จะใช้ในการจัดเรียงแพ็กเกจเหล่านี้ให้อยู่ในลำดับเดิม

4. TCP Acknowledgement Number (32 บิต) : เป็นหมายเลขลำดับแพ็กเกจถัดไปที่ทางฝั่งรับคาดหวัง ซึ่งเป็นการบอกเป็นนัยว่าแพ็กเกจที่มีหมายเลขลำดับก่อนหน้านี้ได้รับหมดแล้วนั่นเอง

5. Data Offset (4 บิต) : เป็นตัวเลขที่บอกขนาดของข้อมูลส่วนหัว (TCP Header) ซึ่งมีหน่วยเป็น 32 บิต หรือ word

6. Reserved (6 บิต) : ส่วนนี้จะถูกกำหนดให้เป็นศูนย์ตลอด ซึ่งข้อมูลส่วนนี้ไม่มีความหมายอะไรเพียงแต่เป็นการสงวนไว้ใช้ในอนาคตเมื่อมีการปรับปรุงโปรโตคอล

7. Flags (6 บิต) : เป็นข้อมูลที่ใช้สำหรับควบคุมการรับส่งแพ็กเกจ เช่น บิต SYN และ ACK ใช้สำหรับการสร้างการเชื่อมต่อ ส่วนบิต FIN เป็นการแจ้งยกเลิกการเชื่อมต่อ เป็นต้น

8. Window Size (16 บิต) : เป็นตัวเลขที่เครื่องปลายทางบอกให้เครื่องต้นทางทราบขนาดวินโดว์ที่เครื่องปลายทางสามารถรับข้อมูลได้

9. Checksum (16 บิต) : เป็นข้อมูลที่ใช้ในการตรวจสอบข้อผิดพลาดของข้อมูลในส่วนหัว โดยเครื่องส่งจะทำการคำนวณค่า เช็คซัม (Checksum) ของข้อมูลส่วนหัว เมื่อเครื่องปลายทางได้รับข้อมูลก็จะทำการคำนวณเช็คซัมด้วยวิธีเดียวกัน แล้วทำการเปรียบเทียบข้อมูลค่าที่คำนวณได้กับค่าที่อยู่ในฟิลด์นี้ ถ้าเหมือนกันแสดงว่าไม่มีข้อผิดพลาดในข้อมูลที่ได้รับ

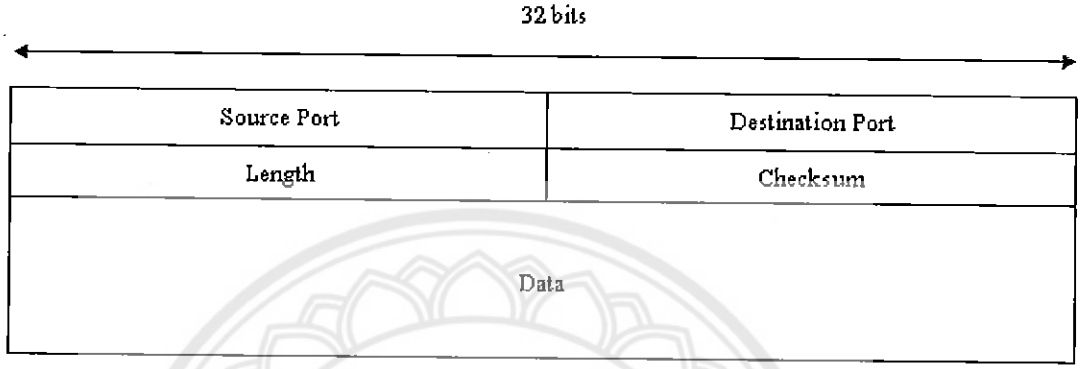
10. Padding : เป็นข้อมูลที่เพิ่มเพื่อให้ข้อมูลส่วนหัวมีจำนวนบิตที่หารด้วย 32 ลงตัว

2.4.4 User Datagram Protocol (UDP)

โปรโตคอล UDP (User Datagram Protocol) จะให้บริการส่งข้อมูลแบบคอนเน็กชันเลสส์ หรือบางทีก็เรียกว่า "ดาต้าแกรม (Datagram)" ซึ่งจะเป็นการให้บริการแบบตรงกันข้ามกับคอนเน็กชันโอเรียนเต็ดของโปรโตคอล TCP การส่งข้อมูลแบบนี้จะเป็นแบบที่เชื่อถือไม่ได้ โดยจะพยายามส่งข้อมูลให้ดีที่สุด ในการรับส่งข้อมูลในแต่ละครั้ง จะไม่มีการสร้างเซสชั่นก่อน และไม่มีกลไกการตอบกลับแพ็กเกจเหมือนโปรโตคอล TCP เหตุที่ตัดกลไกนี้ออกเพื่อเพิ่มประสิทธิภาพในการส่งข้อมูลนั่นเอง แต่ข้อเสียก็คือ การรับส่งเชื่อถือไม่ได้เพราะแพ็กเกจอาจสูญหายระหว่างทางซึ่งทางฝ่ายส่งจะไม่ทราบเลย ดังนั้นโปรโตคอลที่อยู่เหนือกว่าต้องรับผิดชอบเกี่ยวกับการตรวจสอบข้อผิดพลาดของการรับส่งข้อมูลเอง

ถึงแม้ว่าโปรโตคอล UDP จะมีความเชื่อถือน้อย แต่มันก็มีข้อดีหลายอย่าง เช่น ถ้าข้อมูลที่ต้องการส่งมีขนาดเล็กมากก็จะเป็นการเสียเวลา ถ้าต้องสร้างเซสชั่นการเชื่อมต่อระหว่าง 2 โฮสต์นั้นก่อนส่ง และอีกกรณีหนึ่งคือ การส่งข้อมูลแบบแพร่กระจาย หรือบรอดคาสต์ (Broadcast) และมัลติคาสต์ (Multicast) การสร้างเซสชั่นจะเป็นสิ่งที่เป็นไปไม่ได้ เนื่องจากเซสชั่นเป็นการเชื่อมต่อระหว่าง 2 โฮสต์เท่านั้น ดังนั้นการบรอดคาสต์และมัลติคาสต์จะใช้โปรโตคอล UDP เท่านั้น

เนื่องจากโปรโตคอล UDP ไม่รับรองว่าข้อมูลจะส่งถึงปลายทาง ดังนั้นถ้าแอพพลิเคชันที่ใช้โปรโตคอลนี้ต้องการความเชื่อถือได้ของข้อมูล แอพพลิเคชั่นนั้นจะต้องควบคุมและตรวจสอบข้อผิดพลาดเอง ส่วนใหญ่แอพพลิเคชั่นที่ใช้โปรโตคอลนี้จะไม่ต้องการการยืนยันการตอบรับ เช่น โปรโตคอลของระบบเครือข่ายของไมโครซอฟต์ เช่น การล็อกออน (Log on) การบราวซิ่ง (Browsing) และเนมรีโซลูชั่น (Name Resolution) เป็นต้น



รูปที่ 2.13 ฟอร์แมตข้อมูลของแพ็กเกจ UDP

ข้อมูลในส่วนหัวของโปรโตคอล UDP นั้นแสดงในรูปที่ 2.13 ที่แต่ละฟิลด์มีความหมายดังนี้

1. **UDP Source Port Number (16 บิต)** : เป็นหมายเลขพอร์ตของเครื่องส่ง เมื่อรวมหมายเลขพอร์ตนี้กับ IP ก็จะเป็นที่อยู่สำหรับเครื่องรับในการตอบกลับข้อความ

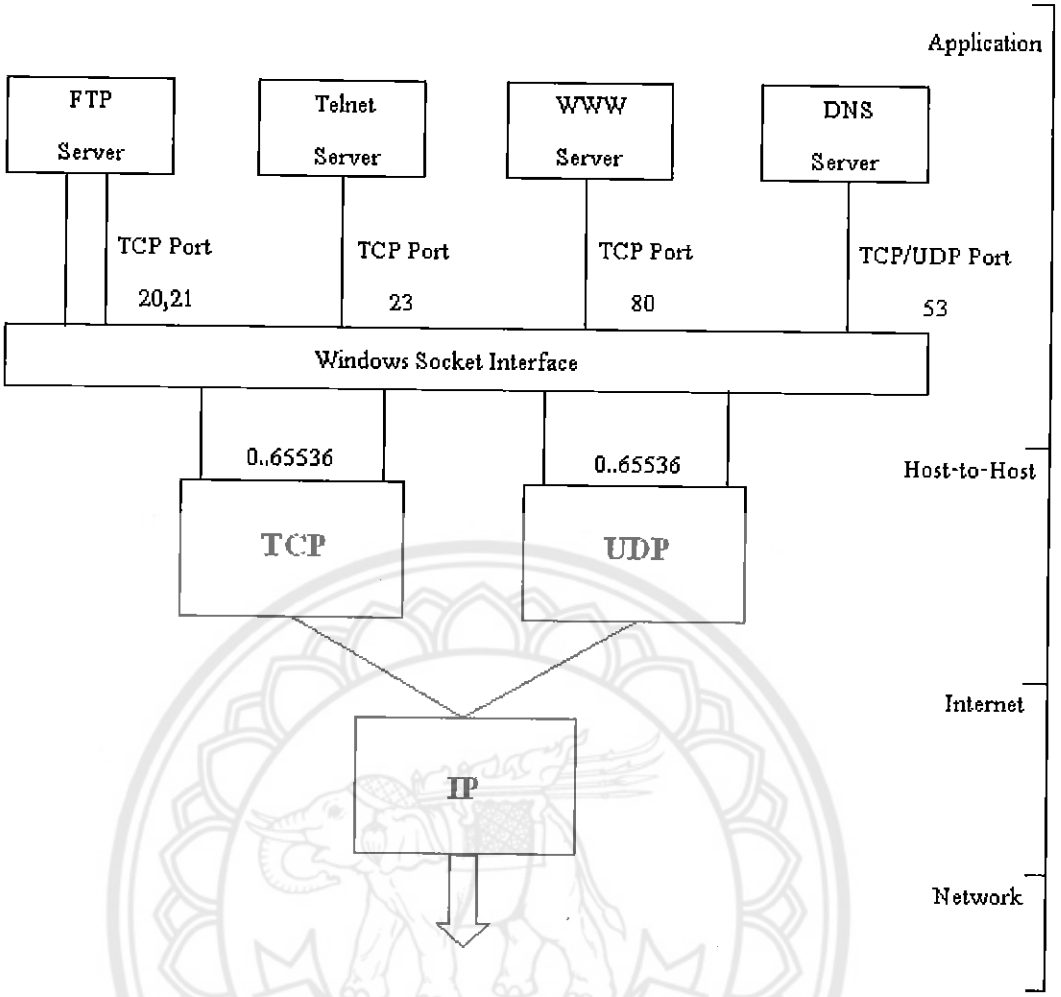
2. **UDP Destination Port Number (16 บิต)** : เป็นหมายเลขพอร์ตของทางฝั่งเครื่องรับที่ใช้ในการส่งผ่านข้อมูลไปยังแอพพลิเคชั่นที่ต้องการติดต่อด้วย

3. **UDP Checksum (16 บิต)** : เป็นข้อมูลที่ใช้ในการตรวจสอบข้อผิดพลาดของข้อมูล เครื่องทางฝั่งรับจะทำการคำนวณหมายเลขนี้ด้วยวิธีเดียวกัน แล้วเปรียบเทียบกับค่าที่ส่งมา ถ้าหมายเลขเท่ากันแสดงว่าไม่มีข้อผิดพลาดในข้อมูลส่วนหัว

4. **UDP Message Length (16 บิต)** : เป็นข้อมูลที่บอกความยาวของข้อมูลทั้งหมด ซึ่งจะเป็นข้อมูลที่ช่วยให้ทางฝ่ายรับทราบว่าข้อมูลควรมีขนาดเท่าใด

2.4.5 หมายเลขพอร์ต

คอมพิวเตอร์ที่ใช้โปรโตคอล TCP/IP ส่วนใหญ่จะมีแอพพลิเคชั่นหลายตัวที่ใช้โปรโตคอล TCP/IP ในการสื่อสารกับเครื่องอื่น ซึ่งโปรโตคอล TCP/IP จะจัดการส่งข้อมูลไปยังแอพพลิเคชั่นที่เหมาะสม เพื่อให้ TCP/IP สามารถรองรับแอพพลิเคชั่นหลายแอพพลิเชั่นในเครื่องเดียว จึงมีการใช้พอร์ตและซ็อกเก็ต เพื่อช่วยในการแยกแยะแอพพลิเคชันต่างๆ รูปที่ 2.14 แสดงการใช้พอร์ตในการรับส่งข้อมูลของแอพพลิเคชั่นประเภทต่างๆ



รูปที่ 2.14 พอร์ตและซ็อกเก็ต

แอพพลิเคชั่นแต่ละตัวจะรับส่งข้อมูลผ่านเครือข่ายจะใช้หมายเลขพอร์ตตั้งแต่ 0 ถึง 65,536 ดังนั้นเพื่อให้การรับส่งข้อมูลถูกต้อง แอพพลิเคชั่นที่รันในเครื่องเดียวกันจะต้องใช้หมายเลขพอร์ตที่ต่างกันเพื่อช่วยลดความสับสน แอพพลิเคชั่นที่นิยมใช้กันทั่วไปส่วนใหญ่จะถูกกำหนดให้ใช้หมายเลขพอร์ตใดพอร์ตหนึ่ง ซึ่งองค์กรที่ทำหน้าที่กำหนดหมายเลขนี้คือ IANA (Internet Assigned Numbers Authority) หมายเลขพอร์ตเหล่านี้จะถูกตีพิมพ์ใน RFC 1700 ซึ่งพอร์ตสำหรับแอพพลิเคชั่นที่นิยมใช้ทั่วไปแสดงในตารางที่ 2.2

ตารางที่ 2.2 หมายเลขพอร์ตของบางแอพพลิเคชั่น

พอร์ต	โปรโตคอล	แอพพลิเคชัน
20	TCP	FTP (Data)
21	TCP	FTP (Control)
23	TCP	Telnet
25	TCP	SMTP (อีเมล)
53	TCP/UDP	DNS (Domain Name System)
80	TCP	HTTP (เว็บเซิร์ฟเวอร์)
110	TCP	POP3 (อีเมล)
161	UDP	SNMP (Simple Network Management Protocol)

โปรโตคอล TCP/IP จะแยกแยะแอพพลิเคชั่นที่รันในแต่ละโฮสต์ โดยใช้ข้อมูล 3 ส่วนต่อไปนี้

1. หมายเลขไอพีของโฮสต์นั้น
2. ประเภทของโปรโตคอลชั้นทรานสปอร์ต : TCP หรือ UDP
3. หมายเลขพอร์ตที่แอพพลิเคชั่นนั้นใช้

ตาราง 2.2 แสดงหมายเลขพอร์ตที่ใช้โดยแอพพลิเคชั่นที่เป็นที่รู้จักโดยทั่วไป ส่วนเครื่องไคลเอนท์ที่เชื่อมต่อกับเซอร์ฟเวอร์จะใช้หมายเลขพอร์ตที่ต่างจากเซิร์ฟเวอร์ ซึ่งหมายเลขพอร์ตที่ใช้บนเครื่องไคลเอนท์ จะถูกจัดการโดยระบบปฏิบัติการของเครื่องนั้น กล่าวคือ ถ้าไคลเอนท์จะเชื่อมต่อกับเว็บเซิร์ฟเวอร์ หมายเลขพอร์ตของเซิร์ฟเวอร์ที่ใช้คือ พอร์ต 80 แต่เครื่องไคลเอนท์จะใช้หมายเลขพอร์ตอื่นที่ว่า

2.4.6 แอพพลิเคชั่นเลเยอร์

การทำงานของโปรโตคอลในชั้นนี้จะเป็นการเข้าใช้ทรัพยากรระยะไกล (Remote Access) และการแชร์การใช้ทรัพยากร (Resource Sharing) โปรโตคอลแอพพลิเคชั่นที่จัดอยู่ในชั้นนี้ได้แก่

1. HTTP (Hyper Text Transfer Protocol) : ใช้สำหรับการรับส่งไฟล์เว็บเพจระหว่างเว็บบราวเซอร์และเว็บเซิร์ฟเวอร์

2. SMTP (Simple Mail Transfer Protocol) : ใช้สำหรับการรับส่งอีเมลระหว่างเมลเซิร์ฟเวอร์

3. POP (Post Office Protocol) : ใช้สำหรับการดาวน์โหลดอีเมลจากเมลเซิร์ฟเวอร์

4. MAP (Internet Message Access Protocol) : ใช้สำหรับการดาวน์โหลดอีเมลจากเมลเซิร์ฟเวอร์

5. FTP (File Transfer Protocol) : ใช้สำหรับถ่ายโอนไฟล์ระหว่างโฮสต์

6. Telnet : ใช้สำหรับการล็อกอินเข้าใช้โฮสต์ระยะไกล

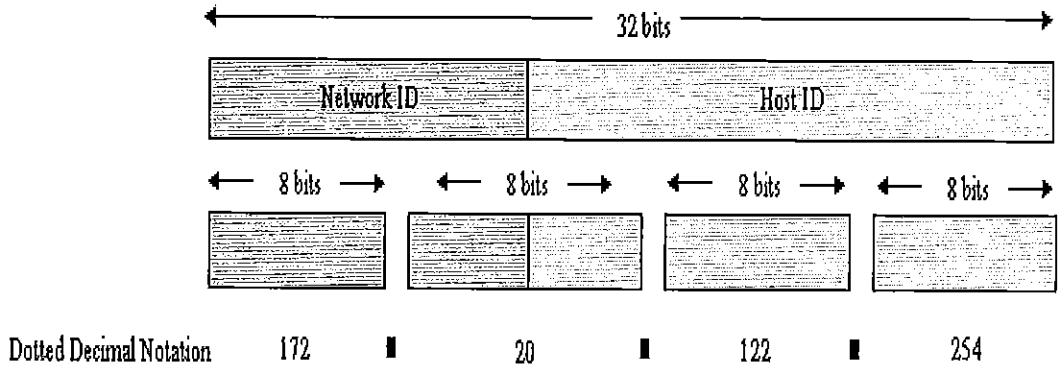
2.4.7 IP Addressing

หมายเลขไอพี (IP Address) คือเลขที่บอกที่อยู่เฉพาะของโหนดหรือโฮสต์ที่อยู่ในเครือข่าย รวมถึงคอมพิวเตอร์และเราท์เตอร์ (Router) ที่อยู่บนระบบเครือข่าย หมายเลขนี้จะเป็นที่อยู่เลเยอร์ที่ 3 หรือชั้นเครือข่ายหมายเลขไอพีของแต่ละเครื่องที่อยู่ในเครือข่ายเดียวกันจะต้องไม่ซ้ำกัน อย่างไรก็ตามโฮสต์หนึ่งอาจจะมีหมายเลขไอพีได้มากกว่าหนึ่งหมายเลขก็ได้ ซึ่งอาจจะมีประโยชน์ในการจัดการวงข่าย เช่น เราท์เตอร์ หรือเกตเวย์เป็นต้น

ปัจจุบันโปรโตคอล IP ที่ใช้งานอยู่ในเครือข่ายอินเตอร์เน็ตจะเป็นเวอร์ชัน 4 หรือเรียกสั้นๆ ว่า "IPv4" ซึ่งในเวอร์ชั่นนี้หมายเลขไอพีจะขนาด 32 บิต เนื่องจากเลขฐานสอง 32 บิตเป็นตัวเลขที่ยาวและยากต่อการจดจำ ดังนั้นเพื่อเป็นการง่าย หมายเลขไอพีจึงนิยมเขียนให้อยู่ในรูปแบบด็อตเดซิมอล (Dotted Decimal Notation) การเขียนให้อยู่ในรูปแบบนี้จะทำได้โดยการจัดเลขฐานสองเป็น 4 กลุ่มๆละ 8 บิต หลังจากนั้นให้แปลงเลขฐานสองของแต่ละกลุ่มให้เป็นเลขฐานสิบ เมื่อแปลงเสร็จแล้วให้เอาเลขทั้งสี่ตัวมารวมกันโดยใช้จุดเป็นตัวเชื่อม ตัวอย่างเช่น เลขไอพี 10101100 00010100 00000001 00011000 เขียนให้อยู่ในรูปแบบด็อตเดซิมอลได้เป็น 172.20.1.24 เป็นต้น ดังตัวอย่างนี้จะเห็นได้ว่าหมายเลขไอพีที่อยู่ในรูปด็อตเดซิมอลจะง่ายต่อการจำมากกว่าหมายเลขไอพีที่อยู่ในรูปเลขฐานสอง เนื่องจากหมายเลขไอพีที่เป็นเลขฐานสิบนี้เป็นการแปลงมาจากเลขฐานสอง 8 บิต ดังนั้นเลขฐานสิบแต่ละตัวจะอยู่ระหว่าง 0 ถึง 255 เพราะฉะนั้นหมายเลขไอพีที่ถูกต้องจะอยู่ระหว่าง 0.0.0.0 ถึง 255.255.255.255

2.4.7.1 ประเภทของหมายเลขไอพี

โปรโตคอลไอพีเวอร์ชัน 4 (IPv4) ที่ใช้อยู่ในปัจจุบันจะแบ่งหมายเลขไอพีออกเป็น 5 ประเภท (class) คือ A,B,C,D และ E โดยหมายเลขไอพีทั้ง 32 บิตจะถูกจัดให้เป็น 2 กลุ่มดังนี้คือ กลุ่มแรกจะเป็นตัวเลขที่ใช้บอกหมายเลขเครือข่าย (Network ID) และกลุ่มที่สองจะเป็นตัวเลขที่บอกหมายเลขโฮสต์ที่อยู่ในเครือข่ายนั้น ดังแสดงในรูปที่ 2.15 ข้างล่าง ส่วนรูปที่ 2.16 แสดงการจัดประเภทของหมายเลขไอพี



รูปที่ 2.15 การแบ่งส่วนของหมายเลขไอพีและด็อตเดซิมอลโนเตชัน

ข้อกำหนดที่ใช้ในการแบ่งประเภทของหมายเลขไอพีมีดังนี้

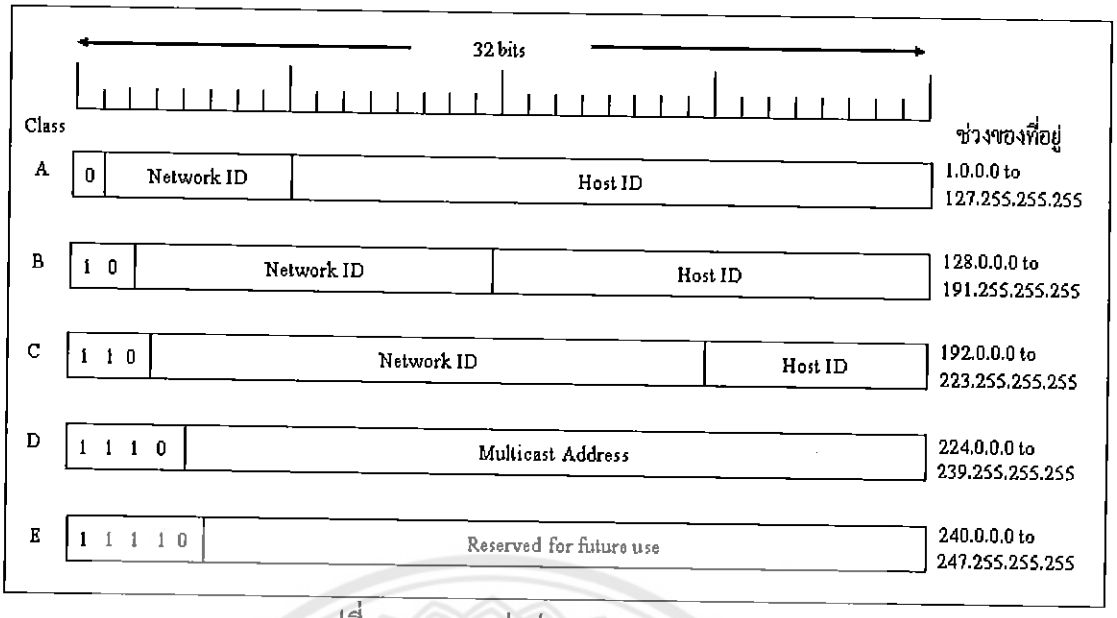
1. Class A : สำหรับหมายเลขไอพีประเภท A หรือคลาสเอนั้น บิตแรกจะเป็นเลข 0 เท่านั้น และส่วนที่บอกหมายเลขเครือข่าย (Network ID) คือ 8 บิตแรก ดังนั้นจะมีได้ทั้งหมด 126 เครือข่าย (หมายเลขเครือข่าย 0 จะไม่ใช้) ส่วนอีก 24 บิตที่เหลือจะเป็นเลขที่ใช้บอกหมายเลขโฮสต์ (Host ID) ดังนั้นในแต่ละเครือข่ายจะมีโฮสต์ทั้งหมด 16,777,124 เครื่อง (หมายเลข 0.0.0 และ 255.255.255 จะไม่ใช้) เนื่องจากเครือข่ายมีจำนวนน้อยมากเมื่อเทียบกับจำนวนโฮสต์ ฉะนั้นหมายเลขไอพีประเภทนี้จึงไม่เหมาะสำหรับเครือข่ายขนาดใหญ่ ซึ่งประกอบด้วยหลายเครือข่ายเชื่อมต่อกัน เช่น ระบบอินเตอร์เน็ต เพราะในการส่งข้อมูลระหว่างเครือข่ายนั้นเราท์เตอร์จะใช้เฉพาะหมายเลขเครือข่ายเท่านั้น ดังนั้นเครือข่ายประเภทนี้จึงเหมาะสำหรับเครือข่ายส่วนบุคคลมากกว่า

2. Class B : สำหรับหมายเลขไอพีประเภท B นั้นสองบิตแรกจะเป็น 10 เท่านั้น ส่วนหมายเลขเครือข่ายจะใช้ 16 บิตแรก ดังนั้นจะมีจำนวนเครือข่ายได้ทั้งหมด 16,382 เครือข่าย ส่วนอีก 16 บิตที่เหลือเป็นหมายเลขโฮสต์ ซึ่งจะทำให้ในแต่ละเครือข่ายมีจำนวนโฮสต์ได้ทั้งหมด 65,534 เครื่อง

3. Class C : สำหรับประเภท C นั้นจะมีบิตเริ่มต้นเป็น 110 และเมื่อรวมกับอีก 21 บิตต่อมาก็จะเป็นหมายเลขเครือข่าย ซึ่งจะได้ทั้งหมด 2,097,152 เครือข่าย ส่วน 8 บิตสุดท้ายเป็นหมายเลขโฮสต์ ซึ่งมีทั้งหมด 254 เครื่อง

4. Class D : ส่วนประเภทที่สี่คือ เลขไอพีขึ้นต้นด้วย 1110 ซึ่งจะเป็นเลขไอพีที่ใช้สำหรับการมัลติคาสต์ (Multicasting) หรือการส่งข้อมูลแบบมีโฮสต์ปลายทางหลายเครื่องแต่อาจจะอยู่คนละเครือข่ายกัน

5. ประเภทสุดท้าย คือเลขไอพีที่ขึ้นต้นด้วย 11110 เป็นหมายเลขที่สงวนไว้ใช้ในอนาคต หมายเลขเหล่านี้จะถูกกำหนดให้โดยศูนย์ข้อมูลเครือข่าย หรือ InterNIC (Internet Network Information Center)



รูปที่ 2.16 การแบ่งประเภทของ IP Address

จะเห็นได้ว่าหมายเลขไอพีแต่ละประเภทจะมีหลายหมายเลขที่สงวนไว้สำหรับกรณีพิเศษ เช่นหมายเลข 0.0.0.0 จะใช้โดยโฮสต์ขณะที่เริ่มเปิดเครื่อง (Boot up) เลขไอพีที่มีหมายเลขเครือข่ายเป็น 0 ทั้งหมดจะใช้อ้างถึงเครือข่ายที่โฮสต์นั้นอยู่ เช่น ถ้าโฮสต์หนึ่งมีเลขไอพีเป็น 172.20.1.24 ซึ่งจะจัดอยู่ในประเภท B โดยมีหมายเลขเครือข่ายเป็น 172.20 เมื่อโฮสต์นี้อ้างถึงหมายเลขไอพี 0.0.1.32 จะมีความหมายเช่นเดียวกับหมายเลข 172.20.1.32 เป็นต้น ส่วนหมายเลขไอพีที่ประกอบด้วยเลข 1 ทั้ง 32 บิตนั้น (255.255.255.255) จะถูกใช้สำหรับการส่งข้อมูลแบบแพร่กระจาย (Broadcast) ในเครือข่ายนั้นๆ ส่วนเลขไอพีที่มีหมายเลขโฮสต์เป็น 1 ทั้งหมดนั้น จะใช้สำหรับการส่งข้อมูลแบบแพร่กระจายไปยังเครือข่ายนั้น เช่น เลขไอพี 172.20.255.255 เป็นเลขที่ใช้สำหรับการส่งข้อมูลแบบแพร่กระจายไปยังโฮสต์ทุกเครื่องที่อยู่ในเครือข่าย 172.20 เป็นต้น ส่วนเลขหมาย 127.xx.yy.zz โดยเลข xx ,yy และ zz จะเป็นเลขอะไรก็ได้ (แต่โดยทั่วไปจะนิยมใช้เป็นเลข 127.0.0.1) นั้น จะใช้สำหรับการส่งข้อมูลไปยังตัวเอง (Loopback) ซึ่งข้อมูลจะไม่ถูกส่งออกไปในเครือข่ายแต่จะถูกโพรเซสภายในเครื่องเท่านั้น แต่โฮสต์นั้นจะถือเสมือนว่าข้อมูลนั้นได้รับผ่านเครือข่าย ประโยชน์ของการทำเช่นนี้ เช่น การทดสอบซอฟต์แวร์ที่ต้องส่งข้อมูลผ่านเครือข่าย แต่ไม่ต้องการส่งข้อมูลออกไปรบกวนระบบเครือข่ายจริง ทำให้ลดความคับคั่งในเครือข่ายลงได้

2.5 ไมโครคอนโทรลเลอร์ MCS – 51

ไมโครคอนโทรลเลอร์ (Microcontroller) เป็นชื่อของอุปกรณ์อิเล็กทรอนิกส์แบบหนึ่งที่บรรจุความสามารถมากมายไม่ว่าจะเป็นหน่วยประมวลผม หน่วยคำนวณทางคณิตศาสตร์และลอจิก วงจรรับสัญญาณอินพุต วงจรขับสัญญาณเอาต์พุต หน่วยความจำ วงจรกำเนิดสัญญาณนาฬิกา ทำให้

ไมโครคอนโทรลเลอร์สามารถนำไปประยุกต์ใช้งานแทนวงจรอิเล็กทรอนิกส์ที่ซับซ้อนได้เป็นอย่างดี โดยช่วยลดจำนวนของอุปกรณ์และขนาดของระบบลง ในขณะที่มีขีดความสามารถสูงขึ้นภายใต้งบประมาณที่เหมาะสม

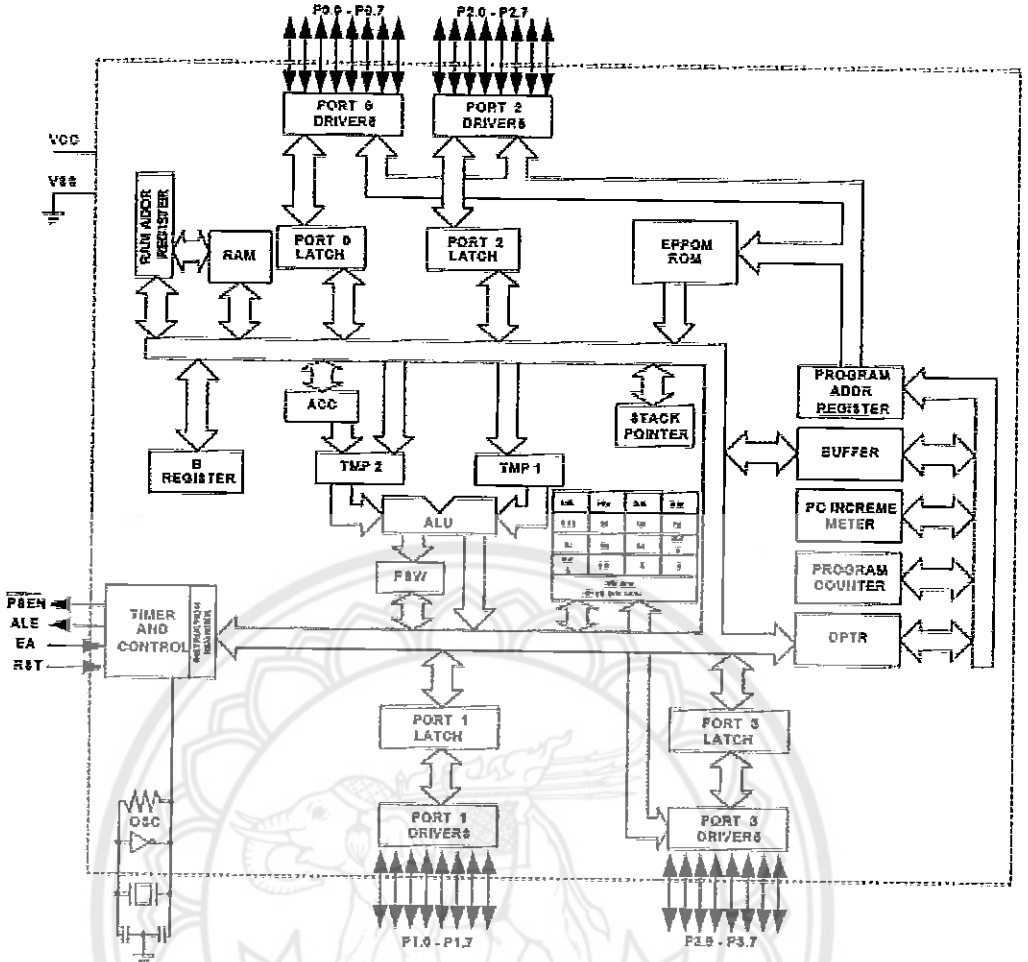
ไมโครคอนโทรลเลอร์มาจากคำสองคำ คือ "ไมโคร" (Micro) ซึ่งหมายถึง ไมโครโปรเซสเซอร์ (Microprocessor) ซึ่งเป็นอุปกรณ์ประมวลผลข้อมูลขนาดเล็ก ซึ่งภายในประกอบด้วยหน่วยประมวลผลกลางหรือซีพียู (CPU : Central Processing Unit) หน่วยคำนวณทางคณิตศาสตร์และลอจิก (ALU : Arithmetic Logic Unit) วงจรเชื่อมต่อหน่วยความจำ และวงจรเชื่อมต่อสัญญาณนาฬิกา อีกคำหนึ่งคือ คำว่า "คอนโทรลเลอร์" (Controller) หมายถึงอุปกรณ์ควบคุม ดังนั้นไมโครคอนโทรลเลอร์ จึงเป็นอุปกรณ์ที่ใช้ในการควบคุม โดยที่สามารถเขียนโปรแกรมเพื่อกำหนดรูปแบบการควบคุมได้อย่างอิสระ

2.5.1 คุณสมบัติทางเทคนิคของไมโครคอนโทรลเลอร์ ตระกูล MCS – 51 [7-8]

- เป็นไมโครคอนโทรลเลอร์ที่ใช้ซีพียูขนาด 8 บิต
- ภายในมีหน่วยความจำโปรแกรมเป็นแบบแฟลชสามารถลบและเขียนใหม่ได้
- หน่วยความจำข้อมูลพื้นฐานเป็นหน่วยความจำแบบแรม ในบางเบอร์จะมีหน่วยความจำแบบอีอีพรอมเพิ่มเติม
- ขาพอร์ตเป็นแบบ 2 ทิศทาง สามารถใช้งานได้เป็นทั้งอินพุตและเอาต์พุต
- มีวงจนสื่อสารอนุกรมแบบฟูลดูเพล็กซ์
- สามารถรองรับแหล่งกำเนิดอินเตอร์รัปต์ได้ 6 ประเภท
- สามารถขยายหน่วยความจำภายนอกเพิ่มเติมได้สูงสุด 64 กิโลไบต์
- มีวงจรกำเนิดสัญญาณนาฬิกาภายในชิป
- มีวงจรสื่อสารอนุกรมแบบ SPI
- มีวอตช์ด็อกไทเมอร์ในตัว

2.5.2 โครงสร้างของ MCS – 51

ไมโครคอนโทรลเลอร์ตระกูล MCS – 51 มีด้วยกันหลายเบอร์ขึ้นกับโครงสร้างภายใน บางเบอร์จะมีหน่วยความจำภายในเป็นแบบ ROM บางเบอร์เป็นแบบ EPROM บางเบอร์มี RAM ภายใน 128 ไบต์ บางเบอร์มี 256 ไบต์ เป็นต้น รายละเอียดโครงสร้างหนักของไมโครคอนโทรลเลอร์ MCS – 51 แสดงดังรูปที่ 2.17



รูปที่ 2.17 โครงสร้างภายในของ MCS-51

2.5.3 การจัดขาต่าง ๆ ของ MCS - 51

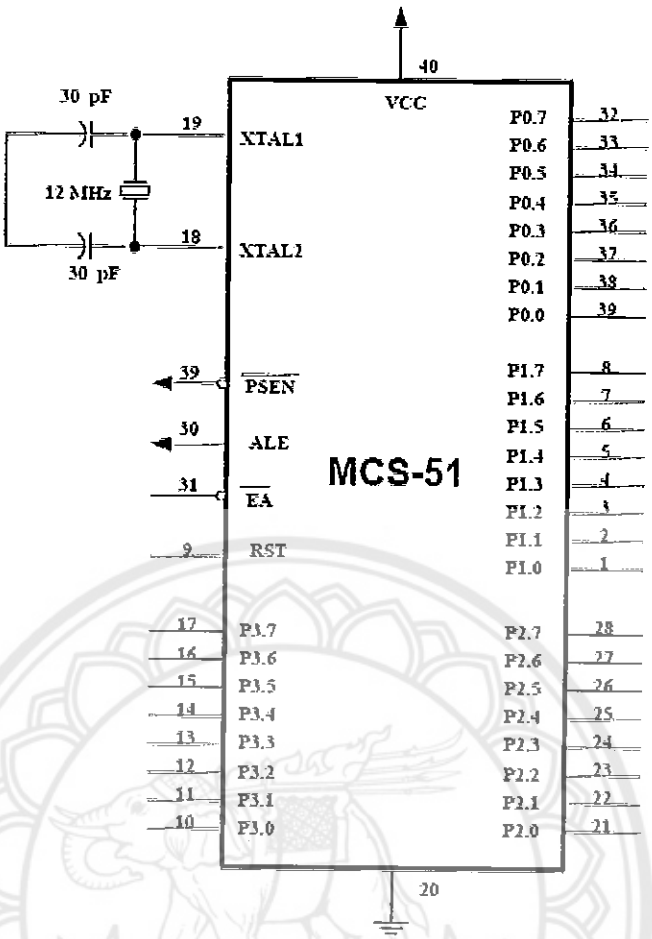
ไอซี ไมโครคอนโทรลเลอร์ MCS – 51 โครงสร้าง IC เป็นแบบ DIP มีขาทั้งหมด 40 ขา โดยขาต่าง ๆ จะใช้เป็นขาพอร์ตอินพุต, เอาต์พุต, ขาสัญญาณควบคุม, ขาตำแหน่งหน่วยความจำ และขาข้อมูลดังรูปที่ 2.18

ความหมายของขาต่าง ๆ มีดังนี้

1. พอร์ต 0 (Port 0) พอร์ต 0 ได้แก่ขาที่ 32 – 39 ของ MCS – 51 สามารถใช้เป็นอินพุตเอาต์พุตได้ นอกจากนี้ในการติดต่อกับหน่วยความจำภายนอกยังใช้เป็นขา Address Bus และ Data Bus อีกด้วย

2. พอร์ต 1 (Port 1) พอร์ต 1 ได้แก่ขาที่ 1 – 8 เป็นพอร์ต 8 บิต สามารถอ้างทีละบิตได้ คือ P1.0, P1.1,..., P1.7

3. พอร์ต 2 (Port 2) พอร์ต 2 ได้แก่ขาที่ 21 – 28 จะใช้งาน 2 หน้าที่ คือใช้เป็นพอร์ต 8 บิตกับใช้เป็นขาแอดเดรส 8 บิตในการอ้างหน่วยความจำภายนอก



รูปที่ 2.18 ขาต่างๆ ของไมโครคอนโทรลเลอร์

4. พอร์ต 3 (Port 3) พอร์ต 3 ได้แก่ขาที่ 10 – 17 จะใช้งานสองหน้าที่คือ เป็นพอร์ตอินพุตและเอาต์พุต และใช้เป็นขาควบคุมต่าง ๆ ดังตารางที่ 2.3

ตารางที่ 2.3 หน้าที่ของขาต่าง ๆ ในพอร์ต 3

บิต	ชื่อ	หน้าที่พิเศษ
P3.0	RXD	ใช้รับข้อมูลทางพอร์ตอนุกรม
P3.1	TXD	ใช้ส่งข้อมูลทางพอร์ตอนุกรม
P3.2	INT0	อินเทอร์รัปต์ภายนอกหมายเลข 0
P3.3	INT1	อินเทอร์รัปต์ภายนอกหมายเลข 1
P3.4	T0	ตัวจับเวลา/ตัวนับ ตัวที่ 0
P3.5	T1	ตัวจับเวลา/ตัวนับ ตัวที่ 1
P3.6	$\overline{WR}$	สัญญาณเขียนข้อมูลหน่วยความจำภายนอก
P3.7	$\overline{RD}$	สัญญาณอ่านข้อมูลหน่วยความจำภายนอก

5. PSEN (Program Store Enable) ขา PSEN เป็นขาที่ส่งสัญญาณออก ได้แก่ขา 29 ขานี้จะแอคทีฟเมื่อ MCS – 51 ต้องการอ่าน Code โปรแกรมภายนอก โดยปกติถ้าหน่วยความจำภายนอกเป็น EPROM ขา PSEN จะต่อกับขา Output Enable (OE) ของ EPROM

6. ALE (Address Latch Enable) เนื่องจากพอร์ต 0 สามารถใช้เป็นขาอ้างตำแหน่งและขาข้อมูล MCS – 51 จึงมีขา ALE ได้แก่ขา 30 Multiplex สัญญาณ Address bus ของ Port 0 ในการใช้งานระบบ MCS – 51 นั้นจำเป็นต้องมีอุปกรณ์มาต่อกับ Port 0 ที่ทำหน้าที่ Latch สัญญาณ Address Bus เมื่อ MCS – 51 ต้องการติดต่อกับหน่วยความจำภายนอก MCS – 51 จะส่งสัญญาณ Address Bus ออกมาก่อนทาง Port 0 จากนั้นจะส่งสัญญาณ ALE มา Latch อุปกรณ์ภายนอก ให้เก็บค่า Address Bus ของ Port 0 ไว้เพื่อใช้ Port เป็น Data Bus ต่อไป

7. EA (External Access) EA ได้แก่ขาที่ 31 ถ้าขานี้เป็นลอจิก "1" จะใช้เบอร์ 8051/8052 เพื่อบอกว่าให้อ่านโปรแกรมจากหน่วยความจำโปรแกรมภายใน แต่ถ้าเป็นลอจิก "0" จะบอกว่าให้ MCS – 51 ทำโปรแกรมโดยอ่านจากหน่วยความจำโปรแกรมภายนอก (ถ้าขา EA เป็น "0" ขา PSEN จะแอคทีฟ) ถ้าหากเป็นเบอร์ 8031 หรือ 8032 ขา EA จะเป็น "0" เสมอ เพราะว่าไม่มีโปรแกรมหน่วยความจำภายใน แต่ถ้าใช้เบอร์ 8051/8052 ซึ่งมีหน่วยความจำโปรแกรมภายในและให้ขา EA เป็น "0" ซึ่งจะ Disabled ROM ภายในและอ่านโปรแกรมจาก EPROM ภายนอกแทน

8. RST (Reset) ขา RST ได้แก่ขา 9 จะใช้ในการรีเซต MCS – 51 โดยจะให้ขานี้เป็นลอจิก "1" อย่างน้อย 2 Machine Cycles จึงจะรีเซตระบบได้

2.5.4 โครงสร้างของพอร์ตอินพุตเอาต์พุต (I/O Port Structure)

ขาของพอร์ตมีโครงสร้างเป็น Field – effect Transistor ต่ออยู่กับขาภายนอก และมีความต้านทานต่อ Pull – up อยู่สำหรับพอร์ต 1, 2, 3 แต่ถ้าเป็นพอร์ต 0 จะไม่มีตัวต้านทาน Pull – up ภายใน เพราะว่าต้องใช้เป็นขา Address Bus และ Data Bus

พอร์ตนี้สามารถใช้เป็นอินพุตเอาต์พุตกับอุปกรณ์ภายนอกได้ ในการอ่านข้อมูลจากพอร์ตจะอ่านได้สองแบบ คือ Read Latch และ Read Pin โดย Read Latch หมายถึงการอ่านข้อมูลที่ถูก Latch เอาไว้ เข้าสู่บัสภายในของ MCS – 51 เช่น การทำคำสั่ง CLP P1.5 แต่ถ้าเป็นการ Read Pin จะเป็นการใช้พอร์ตเป็นอินพุต โดยจะอ่านค่าจากขาของไอซีเข้าสู่บัสภายใน การอ่านแบบ Read Latch และ Read Pin จะมีสัญญาณมาควบคุมที่บัฟเฟอร์

2.5.5 พอร์ตอนุกรมของไมโครคอนโทรลเลอร์ MCS - 51

ไมโครคอนโทรลเลอร์ตระกูล MCS – 51 มีวงจรสื่อสารอนุกรมแบบฟูลดูเพล็กซ์ 1 ชุด (วงจรสื่อสารแบบฟูลดูเพล็กซ์ หมายถึง วงจรสื่อสารที่สามารถทำการรับและส่งข้อมูลในลักษณะ 2 ทิศทาง ได้ในเวลาเดียวกัน) โดยใช้ขาสัญญาณของพอร์ต 3 คือ ขา P3.0 เป็นขารับข้อมูลเข้าหรือ RxD และขา P3.1 เป็นขาส่งข้อมูลออก หรือ TxD โดยวงจรสื่อสารข้อมูลแบบอนุกรมของไมโครคอนโทรลเลอร์ตระกูล MCS – 51 แบบแฟลชเป็นแบบอซิงโครนัส ปกติแล้วพอร์ตอนุกรม

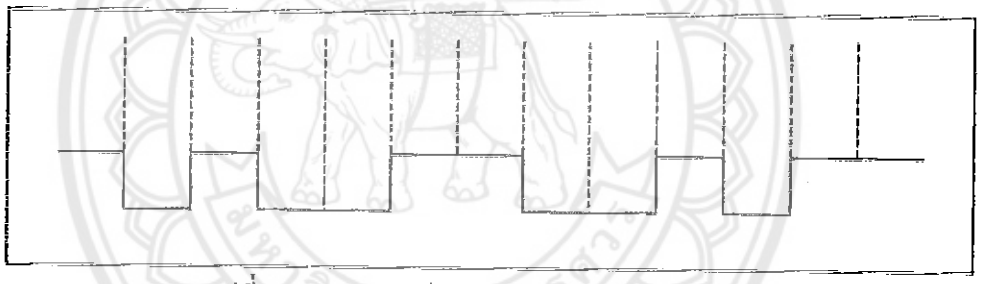
ของไมโครคอนโทรลเลอร์ MCS – 51 จะใช้ในการติดต่อสื่อสารกับพอร์ตอนุกรมของคอมพิวเตอร์ โดยใช้มาตรฐาน RS – 232 แต่ในปัจจุบันสามารถติดต่อกันในมาตรฐาน RS – 422 หรือ RS – 485 ได้แล้ว โดยใช้ไอซีพิเศษที่ทำหน้าที่ในการแปลงสัญญาณการสื่อสารดังกล่าว

2.5.5.1 การสื่อสารข้อมูลแบบอะซิงโครนัส

การสื่อสารข้อมูลแบบอะซิงโครนัสคือการรับและส่งข้อมูล โดยไม่จำเป็นต้องมีสัญญาณนาฬิการ่วมด้วย แต่จะให้การกำหนดอัตราเร็วในการรับและส่งข้อมูลให้มีค่าเท่ากัน เรียกอัตราเร็วนี้ว่า อัตราบอด หรือ บอดเรต (baud rate) มีหน่วยเป็น บิตต่อวินาที (bit per second : bps)

รูปแบบของข้อมูลที่ใช้ในการรับส่งแบบอะซิงโครนัสประกอบด้วย 4 ส่วนด้วยกันคือ

1. บิตเริ่มต้น (start bit) มีขนาด 1 บิต
2. บิตข้อมูลแบบอนุกรม มีขนาด 8 บิต
3. บิตตรวจสอบพาริตี้ (parity bit) มีขนาด 1 บิต หรือไม่มี
4. บิตปิดท้ายหรือบิตหยุด (stop bit) มีขนาด 1 บิต



รูปที่ 2.19 รูปแบบข้อมูลอนุกรมแบบอะซิงโครนัส

รูปที่ 2.19 แสดงรูปแบบของข้อมูล อนุกรมแบบอะซิงโครนัส เมื่อไม่มีการส่งข้อมูล ขา DATA จะมีสถานะลอจิก “1” เรียกสถานะนี้ว่า สถานะหยุดรอ (waiting stage) การเริ่มต้นส่งข้อมูลจะเริ่มจากการให้ขา DATA มีลอจิก “0” ด้วยช่วงระยะเวลา 1 บิต เรียกบิตนี้ว่า บิตเริ่มต้น (start bit) จากนั้นบิตข้อมูลจะถูกส่งออกไป โดยเริ่มจากบิตที่มีนัยสำคัญต่ำสุดหรือบิต LSB ก่อน ข้อมูลที่ต้องการส่งมีจำนวน 8 บิต ตามด้วย บิตพาริตี้ (parity bit) ซึ่งใช้ในการตรวจสอบความผิดพลาดที่เกิดขึ้นจากการส่งข้อมูล บิตสุดท้ายที่จะส่งคือบิตปิดท้ายหรือบิตหยุด (stop bit) โดยจะเป็นการทำให้ขา DATA มีสถานะลอจิก “1” อีกครั้งด้วยระยะเวลาอย่างน้อย 1 บิต, 5 บิต หรือ 2 บิต เพื่อเป็นการแสดงว่าสิ้นสุดข้อมูลแล้ว

อัตราความเร็วในการรับและส่งข้อมูลของการรับส่งข้อมูลแบบอะซิงโครนัสหรืออัตราการบอดหรือบอดเรตที่ใช้สำหรับพอร์ตอนุกรม RS – 232 มีด้วยกันหลายค่า ได้แก่ 110, 150, 300, 600, 1200, 2400, 2800 ค่าของจำนวนบิทที่สามารถส่งได้ใน 1 วินาที สมมติว่า ข้อมูลอนุกรมมีขนาด 8 บิต

ไม่มีการตรวจสอบพาริตี้ มีบิตเริ่มต้นที่ 1 บิต และบิตปิดท้าย 1 บิต ความยาวของข้อมูล 1 ไบต์ จะเท่ากับความยาว 10 บิต ถ้าใช้บอดเรตในการส่งข้อมูลเท่ากับ 9600 บิตต่อวินาที ก็สามารถรับส่งข้อมูลได้ด้วยความเร็ว 960 ไบต์ต่อวินาที

การตรวจสอบพาริตี้สามารถกำหนดให้เป็นแบบคี่ (odd), แบบคู่ (even) หรือไม่มีการตรวจสอบพาริตี้ก็ได้ การตรวจสอบแบบพาริตี้คี่ (หรือคู่) คือการนับจำนวนลอจิก "1" ของข้อมูลขนาด 1 ไบต์และของบิตพาริตี้ว่ามีจำนวนเป็นเลขคี่ (หรือคู่) หรือไม่ ถ้าใช่แสดงว่าข้อมูลที่ส่งมาถูกต้อง ยกตัวอย่าง ข้อมูลที่จะทำการส่งมีขนาด 8 บิต มีค่าเท่ากับ 99H หรือ 10011001B จะเห็นได้ว่าข้อมูลในไบต์นี้มีจำนวนลอจิก "1" จำนวน 4 ตัว ซึ่งเป็นเลขคู่ ดังนั้นถ้ากำหนดการตรวจสอบค่าพาริตี้แบบคู่ ค่าของบิตพาริตี้จะต้องมีลอจิกเป็น "0" แต่ถ้ากำหนดการตรวจสอบพาริตี้เป็นแบบคี่ ค่าของบิตพาริตี้จะต้องเป็น "1" เพื่อให้จำนวนลอจิก "1" ข้อมูล 1 ไบต์รวมทั้งในบิตพาริตี้มีจำนวนรวมเป็นคี่

บิตพาริตี้จะถูกสร้างขึ้นจากภาคส่งข้อมูลของ UART (Universal Asynchronous Receiver Transmitter: เป็นอุปกรณ์ที่ใช้ในการรับและส่งข้อมูลอนุกรม) ซึ่งทางภาครับจะต้องกำหนดคุณสมบัติการตรวจสอบพาริตี้ที่ตรงกันเอาไว้ว่าตรวจสอบพาริตี้คี่หรือพาริตี้คู่ จากนั้นภาครับของ UART จะทำการตรวจสอบค่าพาริตี้ที่เกิดขึ้นว่าเป็นคู่หรือเป็นคี่ โดยการนับจำนวนลอจิก 1 ทั้งหมดรวมทั้งบิตพาริตี้ด้วย ถ้ากำหนดพาริตี้ไว้เป็นคู่ แต่อ่านค่าตัวเลขในการนับออกมาได้ตัวเลขเป็นคี่ ทางภาครับจะแสดงข้อผิดพลาดออกมาให้ผู้ใช้ทราบ กระบวนการดังกล่าวเป็นวิธีการตรวจสอบความผิดพลาดที่เกิดขึ้นในการรับส่งข้อมูลที่ง่ายที่สุด แต่มันสามารถตรวจสอบได้เมื่อมีบิตข้อมูลที่ทำการรับส่งผิดพลาดเพียงบิตเดียวเท่านั้น ถ้าข้อมูลที่ทำการส่งบิตที่มีการผิดพลาดมากกว่า 1 บิต การตรวจสอบด้วยวิธีนี้จะไม่ได้ผล สำหรับการตั้งค่าพาริตี้บิตเป็น NONE นั้น ทั้งภาครับและภาคส่งจะไม่มีการตรวจสอบพาริตี้

2.5.5.2 รีจิสเตอร์ที่เกี่ยวข้องกับการทำงานของพอร์ตอนุกรมใน MCS - 51

ในการทำงานของวงจรพอร์ตอนุกรมในไมโครคอนโทรลเลอร์ MCS – 51 มีรีจิสเตอร์ที่ต้องเกี่ยวข้องอยู่ 2 ตัว มีรายละเอียดดังต่อไปนี้

1. รีจิสเตอร์บัฟเฟอร์ของพอร์ตอนุกรมหรือ SBUF (Serial Data Buffer Register)

มีแอดเดรสอยู่ที่ 99H ในพื้นที่ของรีจิสเตอร์ฟังก์ชันพิเศษหรือ SFR มีขนาด 8 บิต แบ่งเป็น 2 ส่วน คือ รีจิสเตอร์บัฟเฟอร์สำหรับส่งข้อมูล (Transmit buffer register) และรีจิสเตอร์บัฟเฟอร์สำหรับรับข้อมูล (receive buffer register) เมื่อมีการเขียนข้อมูลมายังรีจิสเตอร์ SBUF ข้อมูลนั้นจะถูกส่งต่อไปยังบัฟเฟอร์สำหรับส่งข้อมูล เพื่อส่งออกจากไมโครคอนโทรลเลอร์ผ่านทางขา TxD หรือ P3.1 ในกรณีที่มีการอ่านข้อมูลจากรีจิสเตอร์ด้วย SBUF ข้อมูลจะถูกส่งผ่านไปยังรีจิสเตอร์บัฟเฟอร์สำหรับรับข้อมูลเพื่อส่งต่อไปยังไมโครคอนโทรลเลอร์ต่อไป สำหรับการรับ

ข้อมูลอนุกรมจากภายนอกนั้นจะผ่านมาทางขา RxD หรือ P3.0 ของไมโครคอนโทรลเลอร์ MCS – 51 แบบแฟลช

2. รีจิสเตอร์ควบคุมการทำงานของพอร์ตอนุกรมหรือ SCON

(Serial Port Control Register)

เป็นรีจิสเตอร์ขนาด 8 บิต มีแอดเดรสอยู่ที่ 98H ในพื้นที่ของรีจิสเตอร์ SFR สามารถเข้าถึงได้ในระดับบิต มีรายละเอียดการทำงานดังนี้

บิต 7	บิต 6	บิต 5	บิต 4	บิต 3	บิต 2	บิต 1	บิต 0
SM0	SM1	SM2	SM3	SM4	SM5	SM6	SM7

รูปที่ 2.20 รีจิสเตอร์ควบคุมการทำงานของพอร์ตอนุกรม

SM0 – SM1 (Serial port mode bit 0 – 1) : ใช้ในการเลือกโหมดการทำงานของพอร์ตอนุกรมภายในไมโครคอนโทรลเลอร์

SM2 : ใช้ในการอีนาเบิลการสื่อสารในแบบมัลติโปรเซสเซอร์ (Multiprocessor) ในการทำงานของโหมด 2 และ 3 ของพอร์ตอนุกรมในไมโครคอนโทรลเลอร์ MCS – 51 ในโหมด 2 และ 3 ถ้าบิตนี้เป็น "1" บิต RI จะไม่แอกทีฟ ถ้าบิตที่ 9 ที่รับเข้ามาเป็น "0" (ข้อมูลบิตที่ 9 เก็บไว้ที่บิต RB8) ในการทำงานโหมด 1 ถ้าบิตนี้ถูกเซตบิต RI จะไม่แอกทีฟถ้ายังไม่รับบิตหยุด ส่วนในโหมด 0 บิตนี้ไม่มีการใช้งาน

REN (Enable serial reception) : ใช้ในการอีนาเบิลการรับรู้ข้อมูลพอร์ตอนุกรม ทำการเซตและเคลียร์ด้วยกระบวนการทางซอฟต์แวร์ ถ้าต้องการรับข้อมูลต้องเซตบิตนี้ให้เป็น "1"

TB8: ใช้สำหรับเก็บข้อมูลที่บิต 9 ที่ต้องการส่งออกไปขณะทำงานในโหมด 2 และ 3 ของพอร์ตอนุกรมในไมโครคอนโทรลเลอร์ MCS – 51 เซตและเคลียร์ด้วยกระบวนการทางซอฟต์แวร์

RB8: ใช้สำหรับข้อมูลบิตที่ 9 ที่เข้ามาขณะทำงานในโหมด 2 และ 3 ของพอร์ตอนุกรมในไมโครคอนโทรลเลอร์ MCS – 51 แต่ถ้าหากพอร์ตอนุกรมทำงานอยู่ในโหมด 1 และบิต SM2 เป็น "0" ข้อมูลที่บิต RB8 คือข้อมูลของบิตหยุด (stop bit) สำหรับการทำงานในโหมด 0 บิตนี้จะไม่ใช้งาน บิต RB8 นี้สามารถเซตและเคลียร์ด้วยกระบวนการทางซอฟต์แวร์

TI (Transmit Interrupt flag): ใช้ในการแสดงการเกิดอินเตอร์รัปต์เมื่อมีการส่งข้อมูลออกจากพอร์ตอนุกรมของไมโครคอนโทรลเลอร์ MCS – 51 สามารถเซตได้ด้วยกระบวนการทางฮาร์ดแวร์ เมื่อทำการส่งข้อมูลบิตที่ 8 เรียบร้อยแล้วในการทำงานโหมด 0 ส่วนการทำงานโหมดอื่นบิตนี้จะเซตเมื่อมีการเริ่มต้นส่งบิตหยุดออกไป การเคลียร์บิตนี้ต้องใช้กระบวนการทางซอฟต์แวร์เท่านั้น

RI (Receiver Interrupt flag): ใช้ในการแสดงการเกิดอินเตอร์รัปต์เมื่อมีการรับข้อมูลเข้าจากพอร์ตอนุกรมของโมโครคอนโทรลเลอร์ MCS - 51 สามารถเซตได้ด้วยกระบวนการทางฮาร์ดแวร์ เมื่อทำการรับข้อมูลบิตที่ 8 เรียบร้อยแล้วในการทำงานโหมด 0 ส่วนการทำงานโหมดอื่นบิตนี้จะเซตเมื่อมีการรับบิตหยุดของข้อมูลอนุกรมไปได้ครึ่งทางแล้ว ยกเว้นกรณีที่บิต SM2 มีการเซต บิตนี้จะเซตได้ก็ต่อเมื่อการรับบิตหยุดหรือบิตที่ 9 เกิดขึ้นอย่างสมบูรณ์แล้ว การเคลียร์บิตนี้ต้องใช้กระบวนการทางซอฟต์แวร์เท่านั้น

2.5.5.3 โหมดการทำงานของพอร์ตอนุกรมใน MCS - 51

พอร์ตอนุกรมในไมโครคอนโทรลเลอร์ MCS – 51 แบบแฟลชสามารถเลือกโหมดการทำงานได้ถึง 4 โหมด คือ

1. โหมด 0 เป็นการกำหนดให้พอร์ตอนุกรมทำงานในลักษณะชิฟต์รีจิสเตอร์
2. โหมด 1 เป็นการกำหนดให้เป็น UART ขนาด 8 บิต สามารถเลือกอัตราบอดได้
3. โหมด 2 เป็นการกำหนดให้เป็น UART ขนาด 9 บิต โดยมีอัตราบอดคงที่
4. โหมด 3 เป็นการกำหนดให้เป็น UART ขนาด 9 บิต สามารถเลือกอัตราบอดได้

การเลือกโหมดการทำงานของวงจรพอร์ตอนุกรมในไมโครคอนโทรลเลอร์ MCS – 51 กระทำได้โดยการกำหนดข้อมูลให้แก่บิต SM0 และ SM1 ในรีจิสเตอร์ SCON

2.5.5.4 อัตราบอดของพอร์ตอนุกรมในไมโครคอนโทรลเลอร์ MCS - 51

1. โหมด 0

อัตราบอดของโหมดนี้มีค่าคงที่ โดยสามารถคำนวณได้จากสูตร

อัตราบอดของโหมด 0 = ความถี่ของสัญญาณนาฬิกา/12 มีหน่วยเป็นบิตต่อวินาที

2. โหมด 1 และ 3

ทั้งสองโหมดนี้สามารถเลือกแหล่งกำเนิดอัตราบอดได้ 2 แหล่ง คือ จากอัตราโอเวอร์โฟลวของไทเมอร์ 1 และ 2 สำหรับอัตราบอดเมื่อใช้การโอเวอร์โฟลวของไทเมอร์ 1 จะต้องใช้ค่าของบิต SMOD ในรีจิสเตอร์ PCON มาพิจารณาประกอบด้วย โดยสามารถหาค่าอัตราบอดได้จาก

$$\text{อัตราบอด} = (2^{\text{ค่าของบิต SMOD}}/32) \times \text{อัตราโอเวอร์โฟลวของไทเมอร์}$$

ถ้าหากไทเมอร์ 1 ไม่ได้อินาเบิลการอินเตอร์รัปต์ไว้ สามารถคำนวณหาค่าอัตราบอดได้จาก

$$\text{อัตราบอด} = (2^{\text{ค่าในรีจิสเตอร์ SMOD}}/32) \times (\text{ความถี่สัญญาณนาฬิกา}/\{12\times[256 - (TH1)]\})$$

ตารางที่ 2.4 แสดงการกำหนดอัตราบอดโดยใช้ไทเมอร์ 1

กรณีที่ใช้ไทเมอร์ 2 ในการกำหนดอัตราบอด โดยกำหนดให้ไทเมอร์ 2 ทำงานในโหมดกำเนิดอัตราบอด (baud rate generator) สามารถคำนวณหาค่าอัตราบอดได้จาก

อัตราบอด = อัตราโอเวอร์โฟลวของไทเมอร์ 2/16 มีหน่วยเป็นบิตต่อวินาที

ตารางที่ 2.4 อัตราบอดของวงจรพอร์ตอนุกรมภายในไมโครคอนโทรลเลอร์ MCS – 51

อัตราบอด (บิตต่อวินาที:bps)	ความถี่สัญญาณนาฬิกา	SMOD	ไทเมอร์ 1		
			C/T	โหมด	ค่ารีโหลด
โหมด 0 : สูงสุด 1 MHz	12 MHz	x	x	x	x
โหมด 2 : สูงสุด 375k	12 MHz	1	x	x	x
โหมด 1, 3 : 62.5 K	12 MHz	1	0	2	FFH
19.2 K (19,20)	11.0592 MHz	1	0	2	FDH
9.6 K (9,600)	11.0592 MHz	0	0	2	FDH
4.8 K (4,800)	11.0592 MHz	0	0	2	FAH
2.4 K (2,400)	11.0592 MHz	0	0	2	F4H
1.2 K (1,200)	11.0592 MHz	0	0	2	E8H
137.5	11.0592 MHz	0	0	2	1DH
110	6 MHz	0	0	2	72H
110	12 MHz	0	0	1	FEEBH

ถ้าหากกำหนดให้ไทเมอร์ 2 ทำงานในโหมดไทเมอร์หรือเคาน์เตอร์ตามปกติ สามารถคำนวณหาค่าอัตราบอดได้จาก

อัตราบอด = ความถี่สัญญาณนาฬิกา/(32 X (65536 – (RCAP2H,RCAP2L)))

โดยที่ (RCAP2H,RCAP2L) เป็นค่าของรีจิสเตอร์ RCAP2H และ RCAP2L มีขนาด 16 บิต ไม่คิดเครื่องหมาย

3. โหมด 2

ในโหมดนี้การกำหนดอัตราบอดจะขึ้นอยู่กับค่าของบิต SMOD ในรีจิสเตอร์ PCON ถ้า SMOD เป็น "0" อัตราบอดจะเท่ากับ 1/64 ของความถี่สัญญาณนาฬิกา ในกรณีที่ SMOD เป็น "1" อัตราบอดจะเท่ากับ 1/32 ของความถี่สัญญาณนาฬิกา สามารถแสดงเป็นสูตรคำนวณทางคณิตศาสตร์ได้ดังนี้

$$\text{อัตราบอด} = (2^{\text{ค่าของบิต SMOD}}/64) \times \text{ความถี่สัญญาณนาฬิกา}$$

2.5.5.5 การกำหนดค่าของไทเมอร์เพื่อเลือกอัตราบอด

ในการใช้งานพอร์ตอนุกรมของไมโครคอนโทรลเลอร์ MCS – 51 สิ่งที่ต้องให้ความสนใจมากที่สุดประการหนึ่งคือ อัตราการถ่ายทอดข้อมูล หรือ อัตราบอด ซึ่งการกำหนดอัตราบอดนั้นจะขึ้นอยู่กับค่าความถี่ของสัญญาณนาฬิกาเป็นหลัก สำหรับโหมดการทำงานของพอร์ตอนุกรมที่สามารถเลือกอัตราบอดได้อย่างอิสระคือโหมด 1 และ 3 โดยกำหนดได้จากอัตราการเกิด

โอเวอร์โฟลวของไทเมอร์ 1 หากไทเมอร์ 1 มีการเกิดโอเวอร์โฟลวในอัตราที่สูงมากอัตราบอดก็จะมีค่าสูงมากขึ้นตาม นั่นหมายความว่า อัตราในการถ่ายทอดข้อมูลจะสูงมาก สามารถถ่ายทอดข้อมูลได้อย่างรวดเร็ว ดังนั้นการกำหนดและโหมดการทำงานของไทเมอร์ 1 จึงเป็นสิ่งที่มีผลต่อการเปลี่ยนแปลงของอัตราบอดด้วย

ในการใช้ไทเมอร์ 1 เพื่อกำหนดอัตราบอดในโหมด 1 และ 3 ของพอร์ตอนุกรมจะต้องกำหนดไทเมอร์ ทำงานในโหมด 2 หรือโหมด 8 บิตแบบตั้งค่าการนับอัตโนมัติ ในกรณีการกำหนดค่ารีโหลดให้แก่รีจิสเตอร์ TH1 จะเป็นตัวแปรหลักที่ใช้ในการกำหนดอัตราบอดให้แก่พอร์ตอนุกรมของไมโครคอนโทรลเลอร์ MCS – 51

เริ่มต้นด้วยการเคลียร์บิต SMOD ซึ่งเป็นบิต 7 ของรีจิสเตอร์ PCON ให้เป็น “0” ค่าของการรีโหลดให้แก่ TH1 สามารถคำนวณได้จาก

TH1 = 256 - ((ค่าความถี่ของคริสตอล/384)/อัตราบอด)

แต่ถ้าบิต SMOD เกิดจากการรีเซต จะเป็นการอีนาเบิลการทวีคูณของอัตราบอด ดังนั้นการกำหนดค่าให้แก่ TH1 จึงต้องคำนวณจาก

TH1 = 256 – ((ค่าความถี่ของคริสตอล/192)/อัตราบอด)

ยกตัวอย่างถ้าหากในไมโครคอนโทรลเลอร์ AT89C51 ใช้คริสตอล 11.0592 MHz ต้องการกำหนดอัตราบอดของพอร์ตอนุกรมของไมโครคอนโทรลเลอร์ไว้ที่ 19,200 บิตต่อวินาที ในกรณีที่ไม่อีนาเบิลการทวีคูณของอัตราบอด ค่ารีโหลดของไมโครคอนโทรลเลอร์จะเท่ากับ

TH1 = 256 – ((ค่าความถี่ของคริสตอล/384)/อัตราบอด

= 256 – ZZ11059200/192)/19200)

= 256 – (57600/19200)

= 256 – 3 = 253

นำค่าของ TH1 ที่ได้ทำการแทนค่าคำนวณหาค่าอัตราบอดจะได้เท่ากับ 19,200 บิตต่อวินาที

สามารถสรุปขั้นตอนในการเลือกอัตราบอดโดยการกำหนดค่าของไทเมอร์ 1 ได้ดังนี้

1. กำหนดให้พอร์ตอนุกรมของไมโครคอนโทรลเลอร์ MCS – 51ทำงานในโหมด 1 หรือ 3

2. กำหนดให้ไทเมอร์ 1 ทำงานในโหมด 2 หรือโหมด 8 บิตตั้งค่าอัตโนมัติ

3. กำหนดข้อมูลให้แก่ TH1 เท่ากับ 253 เพื่อให้สามารถกำเนิดอัตราบอดได้ 19,200 บิตต่อวินาที ตามที่ต้องการ

4. ทำการเซตบิต SMOD ซึ่งเป็นบิต 7 ของรีจิสเตอร์ PCON เพื่ออีนาเบิลการทวีคูณอัตราบอด

2.5.5.6 การเขียนหรือส่งข้อมูลออกจากพอร์ตอนุกรม

ข้อมูลที่ต้องการส่งออกทุกค่าต้องนำไปเก็บไว้ในรีจิสเตอร์บัฟเฟอร์ของพอร์ตอนุกรม ซึ่งก็คือ รีจิสเตอร์ SBUF ดังตัวอย่าง

```
MOV SBUF ,#' A'
```

จากคำสั่งข้างต้นเป็นการส่งข้อมูลของตัวอักษร A ออกไปยังพอร์ตอนุกรมของไมโครคอนโทรลเลอร์ อย่างไรก็ตามก่อนที่จะส่งข้อมูลทุกครั้ง ต้องแน่ใจว่าบิต TI เคลียร์ หรือมีค่าเป็น "0" และเมื่อทำการส่งข้อมูลเรียบร้อยแล้ว ก็จะเกิดการเซตบิต TI เพื่อแจ้งให้ทราบ ดังตัวอย่างโปรแกรมต่อไปนี้

```
CLR TI          ; เคลียร์บิต TI เพื่อเตรียมการส่งข้อมูลออก
MOV SBUF, # 'A' ; ส่งข้อมูลของตัวอักษร A ไปยังพอร์ตอนุกรม
JNB TI, $       ; รอการเซตของบิต TI เพื่อแจ้งการส่งข้อมูลที่เสร็จสมบูรณ์
```

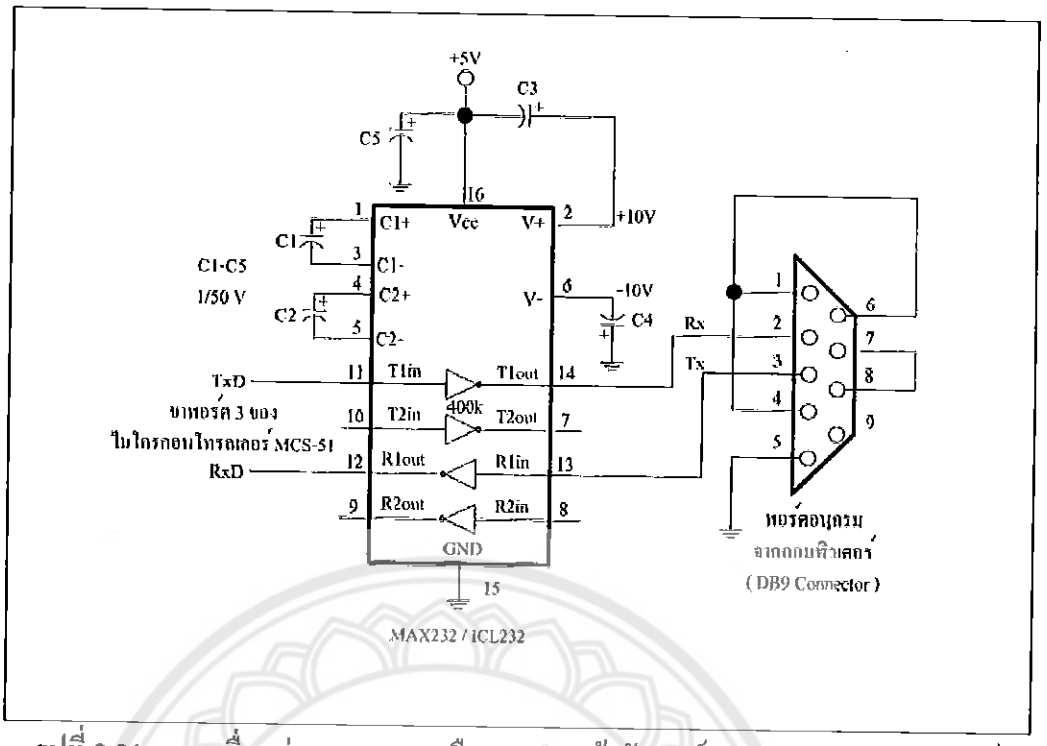
2.5.5.7 การอ่านหรือรับข้อมูลจากพอร์ตอนุกรม

การรับข้อมูลจากพอร์ตอนุกรมสามารถกระทำได้ง่ายมาก เพียงแค่ทำการตรวจสอบว่าบิต RI เกิดการเซตขึ้นหรือไม่ ถ้าพบว่ามีการเซตเกิดขึ้นแล้ว ให้ทำการอ่านค่าจากรีจิสเตอร์ SBUF โดยต้องทำการโอนย้ายข้อมูลผ่านทางแอกคิวมูลเลเตอร์หรือรีจิสเตอร์ A ดังตัวอย่าง

```
CLR RI          ; เคลียร์บิต RI เพื่อเตรียมการรับข้อมูล
JNB RI, $       ; รอคอยการเซตของบิต RI อันเป็นการแจ้งให้ทราบว่า การรับข้อมูล
                  เสร็จสมบูรณ์และมีข้อมูลที่เกิดขึ้นที่รีจิสเตอร์ SBUF
MOV A, SBUF     ; อ่านค่าจากรีจิสเตอร์ โดยการโอนย้ายข้อมูลผ่านทางรีจิสเตอร์ A
CLR RI          ; หลังจากทำการอ่านข้อมูลเรียบร้อยแล้ว ต้องทำการเคลียร์บิต RI
                  เสมอ
```

2.5.5.8 การเชื่อมต่อกับพอร์ตอนุกรมของคอมพิวเตอร์

การใช้งานวงจรพอร์ตอนุกรมของไมโครคอนโทรลเลอร์ MCS – 51 มักนิยมใช้ในการติดต่อเพื่อแลกเปลี่ยนข้อมูลกับคอมพิวเตอร์ผ่านทางพอร์ตอนุกรมในมาตรฐาน RSS – 232 เป็นส่วนใหญ่ แต่เนื่องจากระดับสัญญาณของพอร์ตอนุกรม RS – 232 มีระดับตั้งแต่ -12 ถึง 12 V ในขณะที่ระดับสัญญาณของไมโครคอนโทรลเลอร์ MCS – 51 อยู่ในระดับทีทีแอล ดังนั้นจึงไม่สามารถเชื่อมต่อพอร์ตอนุกรมของไมโครคอนโทรลเลอร์ MCS – 51 เข้ากับพอร์ตอนุกรมของคอมพิวเตอร์ได้โดยตรง จึงต้องอาศัยการเชื่อมต่อผ่านไอซีพิเศษที่ทำหน้าที่แปลงระดับสัญญาณ



รูปที่ 2.21 วงจรเชื่อมต่อ MAX232 หรือ ICL232 เข้ากับพอร์ตอนุกรมของคอมพิวเตอร์และไมโครคอนโทรลเลอร์

ไอซีที่ทำหน้าที่ในการแปลงระดับสัญญาณนี้จะต้องทำการแปลงข้อมูลส่งของไมโครคอนโทรลเลอร์ MCS – 51 จากระดับทีทีแอลไปเป็นระดับของ RS – 232 และทำการแปลงข้อมูลที่รับจากคอมพิวเตอร์จากระดับของ RS – 232 เป็นระดับทีทีแอลเพื่อให้สามารถถ่ายทอดไปยังไมโครคอนโทรลเลอร์ MCS – 51 ได้อย่างสมบูรณ์ ไอซีดังกล่าวมีด้วยกันหลายเบอร์จากผู้ผลิต อาทิ MAX232 [9] จาก MAXIM หรือ ICL232 จาก HARRIS เป็นต้น ในรูปที่ 2.21 แสดงส่วนวงจรของการต่อกับพอร์ตอนุกรมของคอมพิวเตอร์และไมโครคอนโทรลเลอร์

2.6 สเต็ปปิ้งมอเตอร์ (Stepping Motor) [10]

สเต็ปปิ้งมอเตอร์ คือมอเตอร์ชนิดหนึ่งที่ทำการหมุนเป็นขั้นๆ ตามเส้นรอบวงและหยุด แตกต่างกับมอเตอร์ทั่วๆไปซึ่งจะหมุนไปเรื่อยๆ ตลอดเวลา ซึ่งเราจะสามารถควบคุมการหมุนของสเต็ปปิ้งมอเตอร์ ได้อย่างละเอียดโดยคอมพิวเตอร์หรือไมโครคอนโทรลเลอร์ สเต็ปปิ้งมอเตอร์สามารถใช้ในงานละเอียดต่างๆ เช่น ควบคุมการหมุนกระดาษในพรินเตอร์หรือใช้ในการเคลื่อนที่หัวอ่านและเขียนของฟลอปปี้ดิสก์ เป็นต้น สเต็ปปิ้งมอเตอร์ ใช้งานลักษณะระบบเปิด (Open Loop System) คือ สเต็ปปิ้งมอเตอร์สามารถทำงานได้โดยไม่ต้องมีการ ป้อนค่าพารามิเตอร์กลับมา (Feed back) แต่งานที่ต้องการกำหนดตำแหน่งที่แน่นอน ก็จะต้องมีการป้อนกลับไปยังระบบและตัวบอกตำแหน่งว่าถูกต้องหรือผิดพลาดให้ทราบ

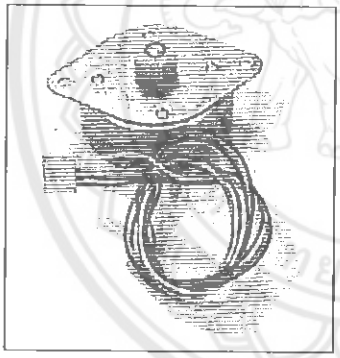
2.6.1 ลักษณะและโครงสร้างของสเต็ปปิ้งมอเตอร์

การพันขดลวดบนสเตเตอร์ของสเต็ปปิ้งมอเตอร์ มีการพันอยู่ด้วยกันกัน 2 แบบ คือ แบบไบโพลาร์ (Bipolar) กับแบบยูนิโพลาร์ (Unipolar)

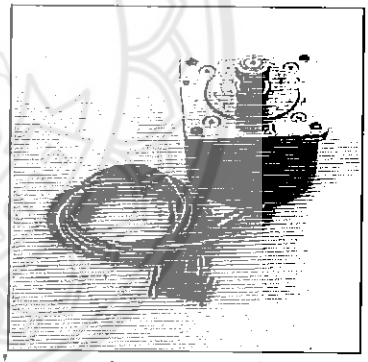
- แบบไบโพลาร์ (Bipolar)จะมีการพันขดลวดหนึ่งขด (จำนวนรอบแล้วแต่การใช้งาน) ในแต่ละขั้วแม่เหล็กของสเตเตอร์ โดยขั้วแม่เหล็กที่เกิดขึ้นที่สเตเตอร์จะถูกกำหนดโดยทิศทางการไหลของกระแสไฟฟ้า ซึ่งสามารถทำให้เกิดขั้วแม่เหล็กในทิศทางตรงกันข้ามได้ เพียงแค่การกลับทิศทางของการไหลของกระแสไฟฟ้า โดยการควบคุมของวงจรสวิตซ์ชิ่งให้กลับขั้วไฟฟ้า

- แบบยูนิโพลาร์ (Unipolar) แบบนี้จะมีการพันขดลวด 2 ขดบนแต่ละขั้วแม่เหล็กของสเตเตอร์ ทำให้แต่ละขดลวดเกิดขั้วแม่เหล็กในทิศทางตรงกันข้าม การกลับทิศทางขั้วแม่เหล็กทำได้โดยใช้วงจรสวิตซ์ชิ่งกลับขั้วไฟฟ้า

การพันขดลวดทั้ง 2 แบบที่กล่าวมาต่างกันคือ แบบยูนิโพลาร์ จะทำให้เกิดแรงบิดน้อยกว่าแบบไบโพลาร์ ความแตกต่างที่รูปร่างของสเต็ปปิ้งมอเตอร์ ทั้ง 2 แบบก็คือ สายไฟที่ต่อจากตัวสเต็ปปิ้งมอเตอร์ ซึ่งแบบไบโพลาร์จะมี 4 สายดังรูปที่ 2.22 ส่วนแบบยูนิโพลาร์จะมี 5 สายหรือ 6 สายดังรูปที่ 2.23



รูปที่ 2.22 สเต็ปปิงมอเตอร์แบบไบโพลาร์



รูปที่ 2.23 สเต็ปปิงมอเตอร์แบบยูนิโพลาร์

2.6.2 ชนิดของสเต็ปปิ้งมอเตอร์

2.6.2.1 แบบแม่เหล็กถาวร (PERMANENT MAGNET – PM) สเต็ปปิ้งมอเตอร์แบบ PM จะมีสเตเตอร์ (STATOR) ที่พันขดลวดไว้หลาย ๆ โพล โดยมีโรเตอร์ (ROTOR) เป็นรูปทรงกระบอกฟันเลื่อย และโรเตอร์ทำด้วยแม่เหล็กถาวร เพื่อป้อนไฟกระแสตรงให้กับขดสเตเตอร์ ทำให้เกิดแรงแม่เหล็กไฟฟ้าผลักต่อโรเตอร์และทำให้มอเตอร์หมุน มอเตอร์แบบ PM จะเกิดแรงฉุดยึดให้โรเตอร์หยุดอยู่กับที่ แม้จะไม่ได้ป้อนไฟเข้าขดลวด

2.6.2.2 แบบแปรค่ารีลักแตนซ์ (VARIABLE RELUCTANCE – VR) สเต็ปปิ้งมอเตอร์แบบ VR จะมีการหมุนโรเตอร์ได้อย่างอิสระ แม้จะไม่ได้จ่ายไฟให้โรเตอร์ทำจากสารเฟอร์โรแมกเนติกกำลังอ่อน มีลักษณะเป็นฟันเลื่อย รูปทรงกระบอก โดยจะมีความสัมพันธ์โดยตรง

กับจำนวนโพลในสเตเตอร์ แรงบิดที่เกิดขึ้นจะหมุนโรเตอร์ไปในเส้นทางของอำนาจแม่เหล็กที่มีค่ารีลักแตนซ์ต่ำที่สุด ตำแหน่งที่จะเกิดแน่นอนและมีเสถียรภาพแต่จะเกิดขึ้นได้หลาย ๆ จุด ดังนั้นเมื่อป้อนไฟเข้าขดลวดต่าง ๆ ในมอเตอร์แตกต่างขดกันไป ก็ทำให้มอเตอร์หมุนไปตำแหน่งต่าง ๆ กัน โรเตอร์ของ VR จะมีความเฉื่อยของโรเตอร์น้อย จึงมีความเร็วรอบสูงกว่ามอเตอร์แบบ PM

2.6.2.3 แบบผสม (HYBRID – H) สเต็ปปิ้งมอเตอร์แบบ H จะเป็นลูกผสมของ VR กับ PM โดยจะมีสเตเตอร์คล้ายกับที่ใช้ใน VR โรเตอร์มีหมวกหุ้มปลาย ซึ่งมีลักษณะของสารแม่เหล็กที่มีกำลังสูง โดยการควบคุมขนาดรูปร่างของหมวกแม่เหล็กอย่างดี จะทำให้ได้มุม การหมุนและจำนวนครั้งในการหมุนน้อยและแม่นยำ ข้อดีก็คือ ให้แรงบิดสูงและมีขนาดกะทัดรัด และให้แรงฉุดยึดโรเตอร์นิ่งกับที่ตอนไม่จ่ายไฟ

2.6.3 การกระตุ้นและควบคุมการหมุนของสเต็ปปิ้งมอเตอร์

การกระตุ้นและควบคุมการหมุนของมอเตอร์ให้เคลื่อนที่ไปแต่ละสเต็ปทำได้โดยจ่ายกำลังไฟฟ้าไปยังขดลวดแต่ละขดบนสเตเตอร์ ซึ่งต้องป้อนเป็นแบบซีเควนเชียล (Sequential) ในรูปแบบที่ถูกต้องด้วย สามารถแบ่งได้เป็น 3 รูปแบบคือ

2.6.3.1 แบบหนึ่งเฟสหรือแบบฟูลสเต็ป (Full step) เป็นการกระตุ้นโดยทำการกระตุ้นขดลวดทีละขดในเวลาหนึ่งไล่เรียงถัดกันไป ทำให้ในการกระตุ้นรูปแบบนี้จึงมีขดลวดเพียงขดลวดเดียวในเวลาหนึ่งที่ถูกกระตุ้นเท่านั้น วงจรแบบเวฟจึงมีราคาถูกและง่าย ขั้นตอนการทำงานต่าง ๆ แสดงดังตารางที่ 2.5

ตารางที่ 2.5 การควบคุมสเต็ปปิ้งมอเตอร์แบบหนึ่งเฟสหรือแบบฟูลสเต็ป (Full step)

สเต็ปที่	เฟสที่ 1	เฟสที่ 2	เฟสที่ 3	เฟสที่ 4
1	ทำงาน	-	-	-
2	-	ทำงาน	-	-
3	-	-	ทำงาน	-
4	-	-	-	ทำงาน

2.6.3.2 แบบ 2 เฟส เป็นการกระตุ้นคล้ายแบบหนึ่งเฟส แต่จะทำการกระตุ้นโดยจ่ายกำลังไฟฟ้าไปที่ขดลวด 2 ขด ที่อยู่ใกล้กันในเวลาเดียวกัน และเรียงถัดไปเช่นเดียวกับแบบเวฟ การกระตุ้นสเต็ปปิ้งมอเตอร์แบบนี้สามารถเพิ่มแรงบิดได้มากกว่าแบบเวฟ โรเตอร์จะเคลื่อนที่ด้วยแรงดึงอย่างเต็มแรง จาก 2 ขดลวดที่ถูกกระตุ้นพร้อมกัน และต่อไปด้วยแรงดึงจากอีก 2 ขดลวดถัดไป สำหรับข้อเสียคือต้องใช้กำลังไฟฟ้ามากขึ้น ขั้นตอนการทำงานต่าง ๆ แสดงดังตารางที่ 2.6

ตารางที่ 2.6 การควบคุมสเต็ปปิ้งมอเตอร์แบบ 2 เฟส

สเต็ปที่	เฟสที่ 1	เฟสที่ 2	เฟสที่ 3	เฟสที่ 4
1	ทำงาน	ทำงาน	-	-
2	-	ทำงาน	ทำงาน	-
3	-	-	ทำงาน	ทำงาน
4	ทำงาน	-	-	ทำงาน

2.6.3.3 แบบครึ่งสเต็ป เป็นรูปแบบการผสมผสานระหว่างการกระตุ้นแบบเวฟและแบบ 2 เฟส เพื่อเพิ่มจำนวนของสเต็ปต่อรอบอีกเท่าตัวหนึ่ง ในระบบนี้จะทำการกระตุ้นเรียงกันไปเป็นลำดับดังนี้ เริ่มจากขดลวดที่ 1, 1 และ 2, 2 และ 3, 3 และ 4, 4, 4 และ 1 แล้ววนกลับมายังขดลวดที่ 1 แรงบิดที่ได้จากการกระตุ้นแบบนี้จะเพิ่มมากขึ้นอีก เพราะช่วงสเต็ปมีระยะสั้นลง แต่ละสเต็ปเกิดแรงดึงจากขดลวด 2 ขดที่ถูกกระตุ้นพร้อมกัน ความถูกต้องของตำแหน่งมีเพิ่มมากขึ้น แต่ต้องพึงระวังเรื่อง เมื่อกระตุ้นให้ทำงานในรูปแบบนี้จะต้องหมุนถึง 2 สเต็ป จึงจะได้ระยะเท่ากับ 1 สเต็ปเต็มของการควบคุมใน 2 แบบแรก สำหรับแหล่งจ่ายไฟฟ้าต้องใช้ขนาดเท่ากับแบบ 2 เฟส เป็นอย่างน้อยจึงจะพอ ขั้นตอนการทำงานต่าง ๆ แสดงดังตารางที่ 2.7

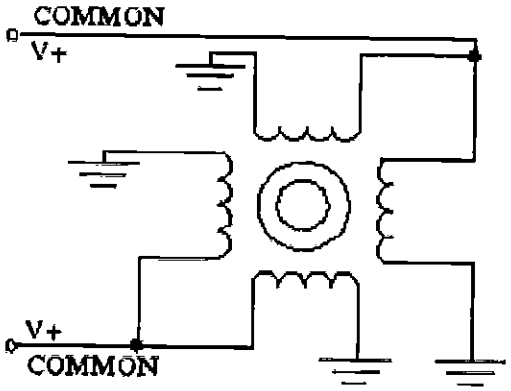
ตารางที่ 2.7 การควบคุมสเต็ปปิ้งมอเตอร์แบบครึ่งสเต็ป

สเต็ปที่	เฟสที่ 1	เฟสที่ 2	เฟสที่ 3	เฟสที่ 4
1	ทำงาน	-	-	-
2	ทำงาน	ทำงาน	-	-
3	-	ทำงาน	-	-
4	-	ทำงาน	ทำงาน	-
5	-	-	ทำงาน	-
6	-	-	ทำงาน	ทำงาน
7	-	-	-	ทำงาน
8	ทำงาน	-	-	ทำงาน

2.6.4 การตรวจสอบหาสาย COMMON และสาย GROUND ของ STEPPING แบบ PM (แบบแกนโรเตอร์เป็นแม่เหล็กถาวร)

โดยทั่วไป SP MOTOR แบบ PM จะมีอยู่ 2 ชนิด

1. ชนิดที่เป็น COMMON ภายนอก SP MOTOR แบบนี้มาสายอยู่ 6 เส้น ดังรูปที่ 2.24

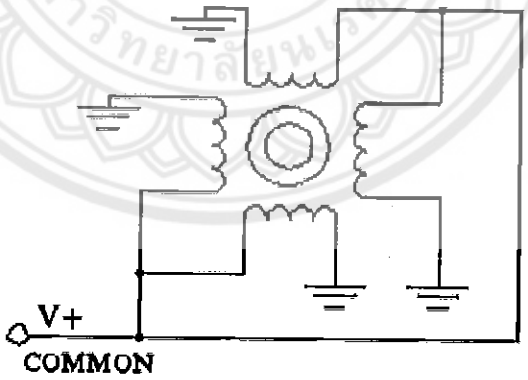


รูปที่ 2.24 วงจรเทียบเท่าของสเต็ปปิ้งมอเตอร์ ชนิดมีสาย 6 เส้น

จากรูปที่ 2.24 สายที่เป็น COMMON มี 2 สาย และสายที่เป็น GROUND มี 4 สาย

สาย COMMON 1 เส้น จะ DRIVE GROUND 2 เส้น ในการเช็คให้ใช้มิเตอร์วัดหาสายที่เป็น COMMON ก่อนโดยการตั้ง RANGE ของมิเตอร์ที่ R*1 จับที่สายทีละคู่ ถ้าหากวัดสาย COMMON เทียบกับสาย GROUND ได้ถูกต้อง ค่าความต้านทานที่อ่านได้จะน้อย แต่ถ้าวัดผิดสาย คือวัดสาย GROUND เทียบกับ GROUND ค่าความต้านทานที่อ่านได้จะสูงกว่า แต่ถ้าวัดสาย COMMON เทียบกับสาย GROUND ที่ไม่ใช่คู่กันแล้ว เข็มมิเตอร์ก็จะไม่กระดิก ให้ทดลองวัดเปรียบเทียบกันทีละคู่ ก็จะทราบว่าสายใดเป็นสาย COMMON สายใดเป็นสาย GROUND

2. ชนิดที่เป็น COMMON ภายใน SP MOTOR แบบนี้มีสายอยู่ 5 เส้น ดังรูปที่ 2.25



รูปที่ 2.25 วงจรเทียบเท่าของสเต็ปปิ้งมอเตอร์ ชนิดมีสาย 5 เส้น

จากรูปที่ 2.25 สายที่เป็น COMMON มี 1 สายและสายที่เป็น GROUND มี 4 สาย

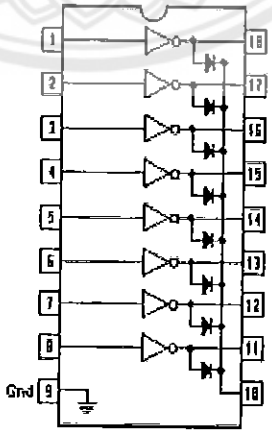
ในการวัดให้ทำแบบเดียวกับการวัด SP MOTOR ชนิด COMMON ภายนอก แตกต่างกันเพียงแบบ COMMON ภายใน สาย COMMON 1 เส้น DRIVE สาย GROUND 4 เส้น ดังนั้นหากสายเส้นใดเมื่อวัดเทียบกับสายเส้นอื่น แล้วมีค่าความต้านทานน้อยที่สุด สายเส้นนั้นเป็นสาย COMMON และที่เหลืออีก 4 เส้น จะเป็นสาย GROUND

2.6.5 การเรียงเฟสของ STEPPING MOTOR แบบ PM

เมื่อเราทราบว่าสายเส้นใดเป็นสาย COMMON แล้ว แต่เรายังไม่ทราบว่าสาย GROUND เส้นใดเป็นเฟสที่ 1 เฟสที่ 2 เฟสที่ 3 และเฟสที่ 4 ในการเรียงเฟสนั้นให้ใช้มิเตอร์วัดโดยนำ V+ เข้าที่สาย COMMON วัดเทียบกับสาย GROUND เส้นใดก็ได้ 1 เส้น จะทำให้แกนโรเตอร์เคลื่อนไปข้างหน้า 1 STEP เมื่อเปลี่ยนสาย GROUND เส้นแรกเป็นเส้นที่ 2 หาก MOTOR ไม่เคลื่อนที่ไปข้างหน้าแสดงว่าการเรียงเฟสไม่ถูกต้อง ให้วัดเทียบกับสาย GROUND เส้นใหม่ต่อไป หาก MOTOR เคลื่อนที่ไปข้างหน้าตามกัน ก็ให้วัดที่สาย GROUND เส้นต่อไปเรื่อย ๆ จะทำให้ทราบว่าสายเส้นใดเป็นเฟสแรก สายเส้นใดเป็นเฟสที่ 2 เฟสที่ 3 และเฟสที่ 4 การเรียงเฟสของ SP MOTOR แบบ PM ทั้งชนิดที่เป็น COMMON ภายนอกและชนิดที่เป็น COMMON ภายใน ใช้หลักการเดียวกัน

2.6.6 การเชื่อมต่อสเต็ปปิ้งมอเตอร์กับไอซีขับสเต็ปปิ้งมอเตอร์ ULN2803

เนื่องจากสเต็ปปิ้งมอเตอร์ต้องการใช้ ความต่างศักย์ไฟฟ้าถึง 12 โวลต์ แต่ไมโครโปรเซสเซอร์ MCS51 สามารถจ่ายความต่างศักย์ไฟฟ้าออกมาได้เพียง 5 โวลต์เท่านั้น และยังมีขนาดกระแสที่ต่ำด้วย ดังนั้นจึงจำเป็นต้องมีวงจรอินเตอร์เฟสเพื่อที่จะนำมาควบคุมการจ่ายกระแสไฟฟ้าให้กับสเต็ปปิ้งมอเตอร์ เพื่อจ่ายความต่างศักย์ไฟฟ้าเพิ่มขึ้นมาให้เพียงพอถึง 12 โวลต์ และมีกระแสไฟฟ้าที่เพียงพอ ในโครงงานนี้ได้เลือกใช้ไอซีขับสเต็ปปิ้งมอเตอร์ ULN2803 [11] ซึ่งไอซีเบอร์นี้มีลักษณะโครงสร้างภายในเป็นทราสซิสเตอร์ต่อกันแบบดาร์ลิงตันอยู่ทั้งหมดแปดชุดด้วยกัน ซึ่งทำให้ได้กระแสเอาต์พุตของแต่ละชุดสูงถึง 500 มิลลิแอมป์ และแรงดันเอาต์พุตสูงสุดที่ 50 โวลต์ ทำให้เพียงพอต่อการนำควบคุมการจ่ายกระแสไฟฟ้าให้กับสเต็ปปิ้งมอเตอร์ โครงสร้างของ ULN2803 แสดงดังรูปที่ 2.26

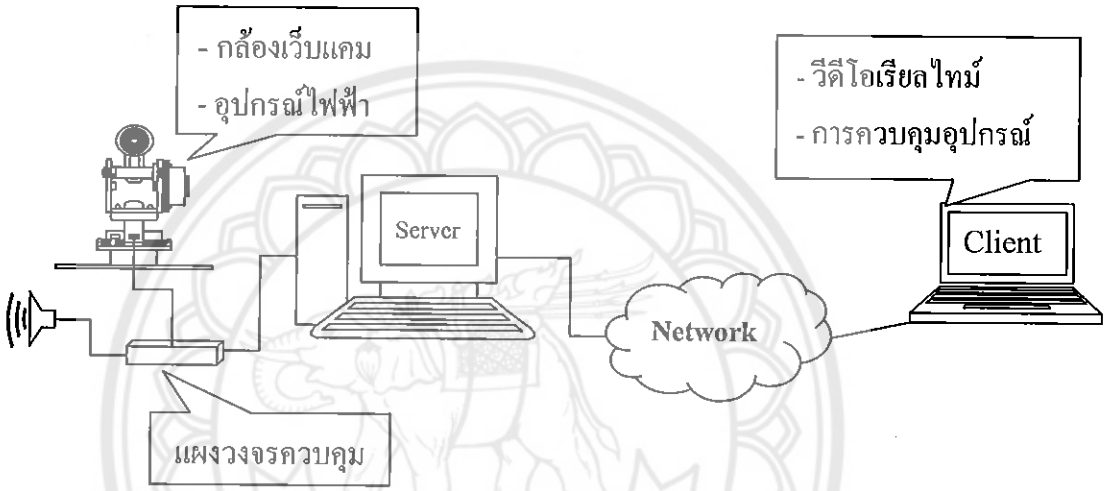


รูปที่ 2.26 โครงสร้างของ ULN2803

บทที่ 3
ขั้นตอนการดำเนินงาน

เมื่อมีการศึกษาทฤษฎีและวิธีการที่จำเป็นต้องใช้ในการพัฒนาโครงงานนี้แล้ว ขั้นตอนต่อไปก็จะเป็นส่วนของการออกแบบและพัฒนาโปรแกรม โดยมีลำดับขั้นตอนต่างๆ ดังนี้

3.1 การออกแบบภาพรวมของโครงงาน

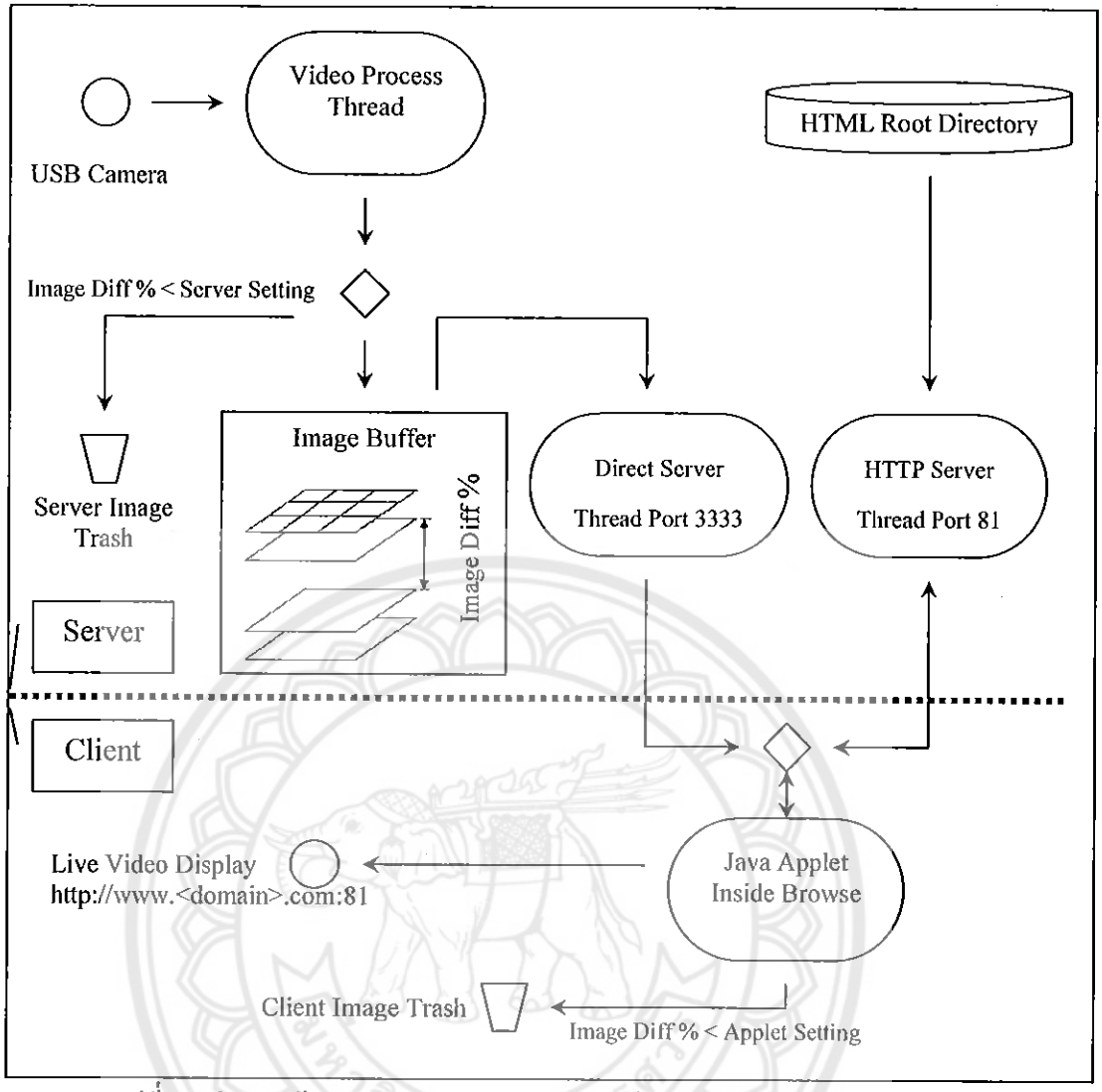


รูปที่ 3.1 ภาพรวมของโครงงาน

จากรูปที่ 3.1 เป็นภาพรวมของโครงงานการควบคุมอุปกรณ์ผ่านระบบเครือข่าย โดยที่สามารถแยกออกเป็น 2 ส่วนใหญ่ ๆ ได้คือ ส่วนเครื่องแม่ข่าย (Server) และส่วนเครื่องลูกข่าย (Client) ซึ่งอุปกรณ์ต่าง ๆ ที่จะทำการควบคุมจะอยู่ทางฝั่งเครื่องแม่ข่าย และคำสั่งควบคุมจะได้รับจากฝั่งเครื่องลูกข่ายพร้อมกันนั้นก็สามารถรับชมภาพวีดีโอเรียลไทม์ไปด้วย

3.2 การออกแบบเว็บเซิร์ฟเวอร์เพื่อให้บริการภาพวีดีโอแบบเรียลไทม์

โปรแกรมส่วนนี้จะเป็นส่วนที่มีการเขียนโปรแกรมเพื่อติดต่อรับภาพวีดีโอจากกล้องเว็บแคม (Web Cam) ซึ่งในโครงงานนี้ใช้เว็บแคมรุ่น "Logitech QuickCam Express" [12] และโปรแกรมจะคอยให้บริการภาพวีดีโอที่ได้รับจากกล้องเว็บแคมกับเครื่องลูกข่าย โดยการผ่านการขอบริการผ่านทางเว็บบราวเซอร์ โดยที่ส่วนแสดงผลจะแสดงผลโดยจาวาแอพเพล็ต [13] โดยลักษณะโครงสร้างการทำงานของระบบแสดงดังรูปที่ 3.2



รูปที่ 3.2 โครงสร้างการทำงานของระบบการให้บริการภาพวีดีโอแบบเรียลไทม์

จากรูปที่ 3.2 โครงสร้างการทำงานของระบบสามารถแบ่งออกเป็นส่วนๆ ได้ดังนี้

3.2.1 ส่วนการประมวลผลภาพที่ได้รับจากกล้องเว็บแคม

จากรูปโครงสร้างจะอยู่ในส่วน Video Process Thread ในส่วนนี้จะเป็นกระบวนการจัดการกับอุปกรณ์ในการรับภาพ ตั้งแต่ขั้นตอนการกำหนดค่าเริ่มต้นต่างๆ การโหลดไดร์เวอร์ การรับภาพ ไปจนถึงการจัดการกับรูปแบบข้อมูลที่รับเข้ามาและเปลี่ยนรูปแบบของข้อมูลเพื่อให้เหมาะสมต่อการส่งข้อมูลผ่านระบบเครือข่าย ซึ่งในกระบวนการต่างๆ นี้จะใช้ Java Media Framework API [5] เป็นไลบรารีที่มาช่วยในการเขียนโปรแกรม

3.2.2 ส่วนการเก็บข้อมูลภาพสำรอง (Image Buffer)

เมื่อกระบวนการประมวลผลภาพดำเนินการเสร็จแล้วก็จะส่งข้อมูลเก็บในหน่วยความจำเพื่อเป็นการเก็บข้อมูลที่ได้รับไว้ เหตุที่ต้องมีการเก็บข้อมูลลงที่พักข้อมูล (Buffer) ก็เป็นเพราะว่าในการแสดงข้อมูลในลักษณะที่เป็นสื่อผสม (Media) ที่มีลักษณะการเคลื่อนที่ หลักการก็คือภาพนิ่ง

หลายภาพที่มีความต่อเนื่องกันนำมาฉายต่อๆ กันในความถี่เท่าที่ตามนุษย์รับได้ ก็จะเห็นเป็นภาพที่มีการเคลื่อนไหวในรูปแบบวีดีโอ ดังนั้นจึงต้องมีการเก็บข้อมูลพักไว้เพื่อเป็นการสำรองภาพที่ได้รับจากการประมวลผลให้สามารถนำไปแสดงผลอย่างต่อเนื่องได้ทัน และที่สำคัญในการส่งสตรีมข้อมูลในลักษณะสื่อผสมนี้ผ่านทางระบบเครือข่ายจะต้องมีความต่อเนื่องของข้อมูลเป็นอย่างมาก เพราะว่าเส้นทางในการส่งอาจเป็นผลให้การส่งข้อมูลขาดความต่อเนื่องได้ แล้วภาพที่ได้รับชมอาจขาดความต่อเนื่อง ในส่วนของการสร้างพื้นที่สำรองข้อมูลนั้นจะมีการออกแบบให้มีขนาดไม่ใหญ่เกินไปเพื่อให้ไม่เปลืองเนื้อที่ของหน่วยความจำ และในการเก็บข้อมูลในแต่ละเฟรมนั้นจะมีการแบ่งเฟรมออกเป็นเก้าส่วนเพื่อให้ข้อมูลที่จะส่งไปมีขนาดเล็ก และก่อนที่จะมีการจัดเก็บข้อมูลในแต่ละเฟรมจะมีการเปรียบเทียบความแตกต่างของภาพที่รับเข้ามากับภาพที่ถูกเก็บอยู่แล้วว่ามีความแตกต่างเกินกว่าที่กำหนดไว้หรือไม่ ถ้าน้อยกว่าก็ไม่มีการเก็บเฟรมนั้นเข้ามา แต่ถ้ามากกว่าก็จะทำการเก็บเฟรมนั้นเข้ามาที่พื้นที่สำรองข้อมูล ที่ทำเช่นนี้ก็เป็นเพราะว่าหากไม่การเคลื่อนไหวก็ไม่จำเป็นต้องทำการเก็บภาพเพื่อที่จะส่งข้อมูลไปให้ส่วนแสดงผล

3.2.3 ส่วนกระบวนการส่งข้อมูลเพื่อนำไปแสดงผล

ส่วนนี้จะเริ่มทำงานเมื่อมีการร้องขอจากผู้ใช้ทางฝั่งเครื่องลูกข่ายเกิดขึ้น โดยที่จะทำการสร้างเส้นทางเพื่อที่จะส่งผ่านข้อมูลที่เก็บไว้ในพื้นที่สำรองข้อมูลไปให้กับส่วนแสดงผล โดยส่วนนี้แสดงในรูปที่ 3.2 คือ Direct Server Thread Port 3333 ซึ่งพอร์ตที่ทำการเปิดไว้เพื่อใช้เป็นช่องทางในการส่งข้อมูลคือ พอร์ต 3333 ที่มีการบอกว่าเป็นเทรด (Thread) ก็เพราะการเชื่อมต่อเข้ามาที่พอร์ตนี้สามารถเชื่อมต่อพร้อมกันได้หลายๆ ผู้ใช้ในเวลาเดียวกัน โดยโปรแกรมจะทำการจัดการสร้างเส้นทางให้กับผู้ใช้แต่ละคนที่เข้ามาได้อย่างสัมพันธ์กัน

3.2.4 ส่วนเว็บเซิร์ฟเวอร์ที่คอยให้บริการในการเรียกหน้าแรกของเว็บ

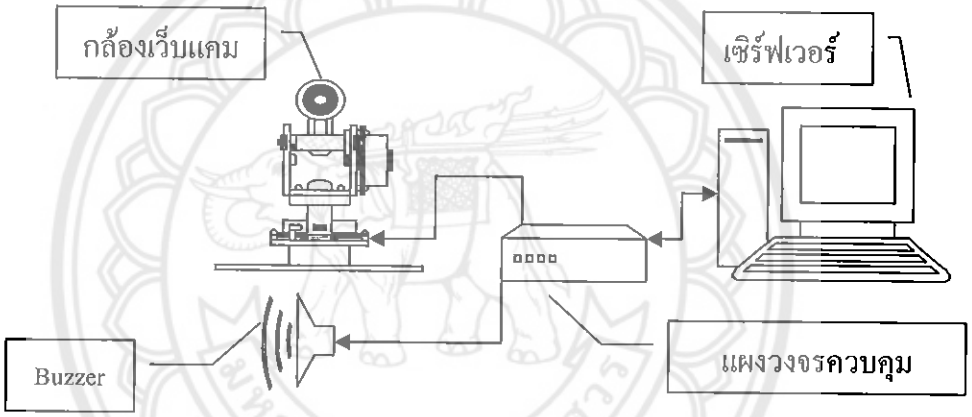
เป็นส่วนที่สร้างพื้นที่เพื่อเก็บไฟล์ index.html เพื่อให้ผู้ใช้สามารถเรียกใช้หน้าแรกผ่านทางเว็บบราวเซอร์ได้ ดังแสดงในรูปที่ 3.2 คือส่วน HTML ROOT Directory การทำงานในส่วนก็จะมีหน้าที่หลักที่เหมือนกับเว็บเซิร์ฟเวอร์ทั่วๆ ไป โดยพอร์ตที่เปิดบริการไว้คือพอร์ต 81 ใน HTML Root Directory นอกจากจะเก็บ ไฟล์ index.html ก็จะมีการเก็บไฟล์ .class ที่เป็นโปรแกรมส่วนของแอพเพล็ตที่แทรกอยู่ในหน้า index.html เพื่อเป็นส่วนที่จะจัดสร้างการเชื่อมต่อกับเครื่องแม่ข่ายผ่านทางพอร์ต 3333 เพื่อนำข้อมูลภาพในพื้นที่สำรองข้อมูล มาจัดการและทำการแสดงผลเป็นภาพวีดีโอ

3.2.5 ส่วนแสดงผล

ส่วนแสดงผลจะเป็นส่วนที่อยู่ทางฝั่งเครื่องลูกข่ายโดยในการเรียกใช้โปรแกรมจะกระทำการผ่านทางเว็บบราวเซอร์ โดยที่เว็บเว็บบราวเซอร์จะเป็นของค่ายไหนก็ได้ที่มีการติดตั้งปลั๊กอินของจาวารันไทม์ลงไป เพื่อทำการแสดงผลโปรแกรมที่เป็นจาวาแอพเพล็ต ในการเรียกใช้ก็จะทำการใส่แอดเดรสและเบอร์พอร์ตของเครื่องแม่ข่ายที่เปิดบริการอยู่ หน้า index.html ก็จะถูกเรียก

ขึ้นมาแสดงผล แล้วแอพเพล็ตที่แทรกอยู่ในหน้านี้ก็จะถูกรันขึ้นมาจากนั้นก็จะทำการสร้างการเชื่อมต่อกับเครื่องแม่ข่ายผ่านทางพอร์ต 3333 โดยอัตโนมัติ จากนั้นก็จะทำการแสดงผลสตรีมข้อมูลที่รับมาเป็นภาพวีดีโอที่ได้จากกล้องเว็บแคม ซึ่งในการแสดงผลจะมีการกำหนดค่าความแตกต่างของภาพที่รับมากับภาพที่แสดงผลไปแล้วไว้ หากภาพที่รับมาใหม่มีความแตกต่างของภาพน้อยกว่าที่กำหนดไว้นั่นหมายถึงภาพมีการเคลื่อนไหวที่เปลี่ยนไปน้อยมากก็จะทำการทิ้งข้อมูลนั้นไป และยังคงแสดงภาพนั้นค้างอยู่ จนกว่าจะมีการเคลื่อนไหวเกิดขึ้นจึงจะทำการรับภาพใหม่เข้ามาแสดงผล สาเหตุที่สร้างเงื่อนไขนี้ขึ้นก็เนื่องมาจากให้ลดปริมาณในการส่งข้อมูลและกระบวนการในการจัดการกับการแสดงผล

3.3 การออกแบบส่วนควบคุมการเปลี่ยนมุมของกล้องและเปิด-ปิดอุปกรณ์ไฟฟ้า



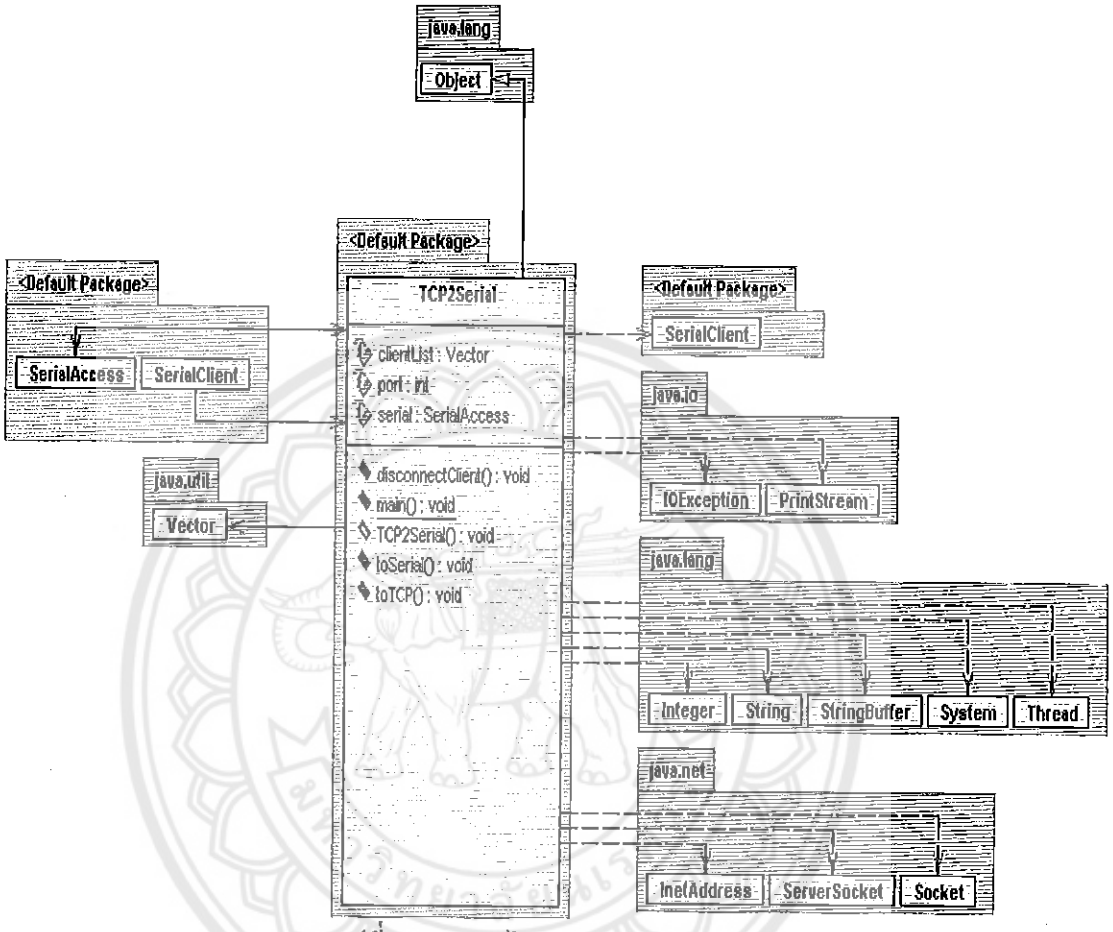
รูปที่ 3.3 ส่วนควบคุมการเปลี่ยนมุมของกล้องและเปิด-ปิดอุปกรณ์ไฟฟ้า

จากรูปที่ 3.3 เป็นภาพรวมการทำงานของส่วนควบคุมการหมุนของกล้องและเปิด-ปิดอุปกรณ์ไฟฟ้า โดยจะสามารถแบ่งออกได้เป็น 3 ส่วน คือ ส่วนโปรแกรมที่อยู่ในเครื่องคอมพิวเตอร์ ส่วนของแผงวงจรควบคุม และส่วนฮาร์ดแวร์ที่ควบคุมการหมุนของกล้อง ซึ่งสามารถแสดงรายละเอียดของแต่ละส่วนได้ดังนี้

3.3.1 การออกแบบโปรแกรมในส่วนเครื่องคอมพิวเตอร์

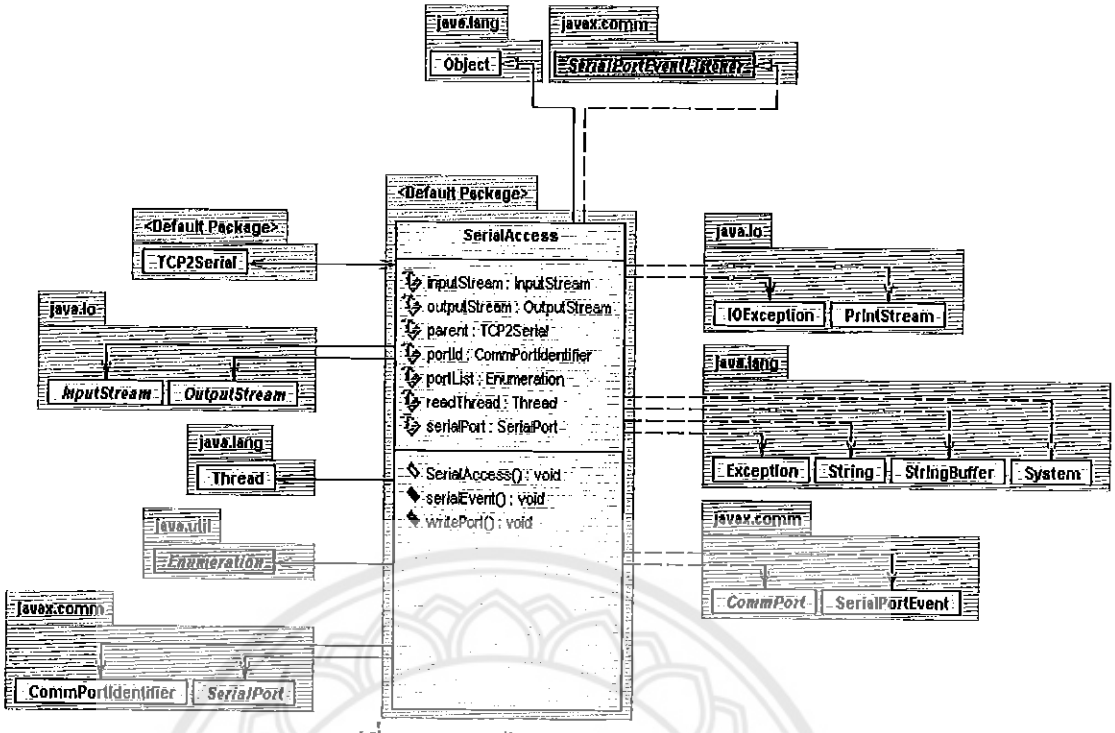
ซอฟต์แวร์ที่อยู่ในเครื่องคอมพิวเตอร์ เป็นโปรแกรมที่เขียนขึ้นด้วยภาษาจาวา โดยหน้าที่ของโปรแกรมส่วนนี้ มีหน้าที่หลักๆ สองอย่างคือ สร้างการเชื่อมต่อกับเครื่องลูกข่ายเพื่อรับคำสั่งที่จะทำการควบคุมการเปลี่ยนมุมของกล้องและเปิด-ปิดอุปกรณ์ไฟฟ้าผ่านโปรโตคอลทีซีพี และส่งข้อมูลที่ได้รับมาออกไปให้ไมโครคอนโทรลเลอร์ ผ่านทางคอมพอร์ต โดยโปรแกรมส่วนนี้ชื่อว่า TCP2Serial โดยแบ่งออกเป็นคลาสได้สามคลาส คือ

1. คลาส TCP2Serial ทำหน้าเป็นคลาสหลักในการจัดการการทำงานรวมของโปรแกรมโดยจะเป็นตัวที่สร้างเซิร์ฟเวอร์ เพื่อรอการเชื่อมต่อจากเครื่องลูกข่าย และคอยจัดการในการส่งคำสั่งออกทางพอร์ตอนุกรมหรือส่งไปทีซีพีพอร์ต



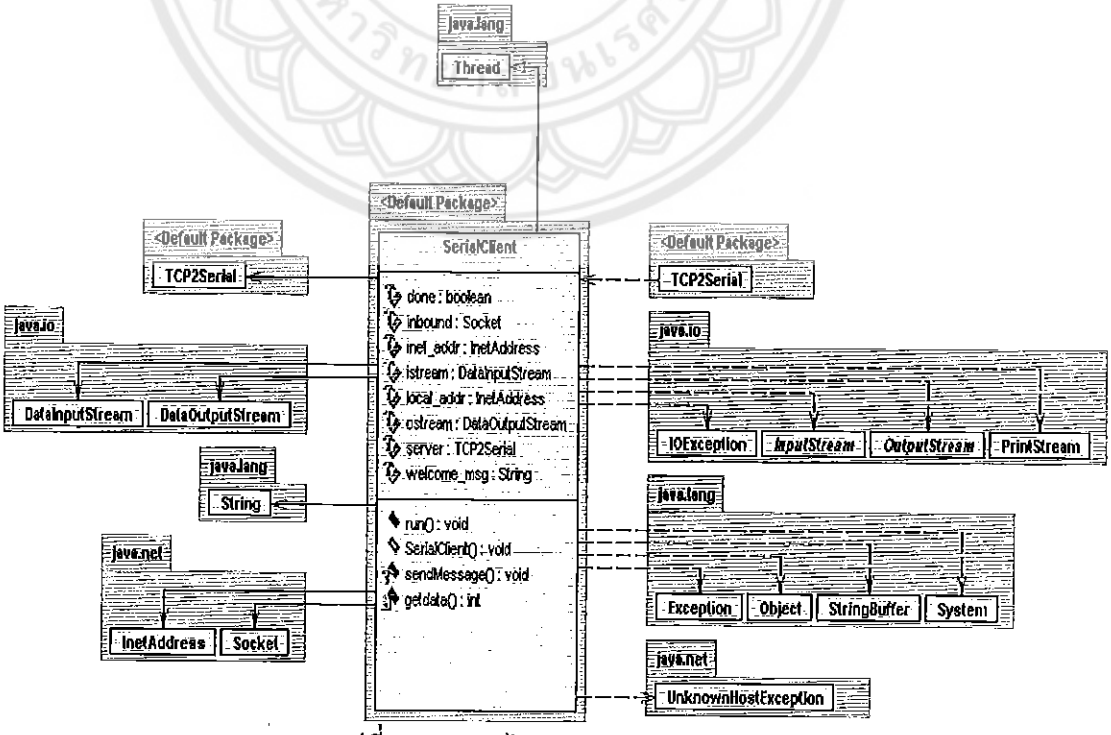
รูปที่ 3.4 คลาสไดอะแกรม TCP2Serial

2. คลาส SerialAccess ทำหน้าที่ในการติดต่อกับพอร์ตอนุกรม โดยจะมีเมธอดในการจัดการกับเหตุการณ์ของพอร์ตอนุกรม คือทำการเปิดพอร์ต กำหนดอัตราการในรับส่งข้อมูล (Baud Rate) กำหนดรูปแบบของข้อมูล และติดต่อรับ- ส่งข้อมูลผ่านพอร์ตอนุกรม ซึ่งแสดงคลาสไดอะแกรมดังรูปที่ 3.5



รูปที่ 3.5 คลาสไดอะแกรม SerialAccess

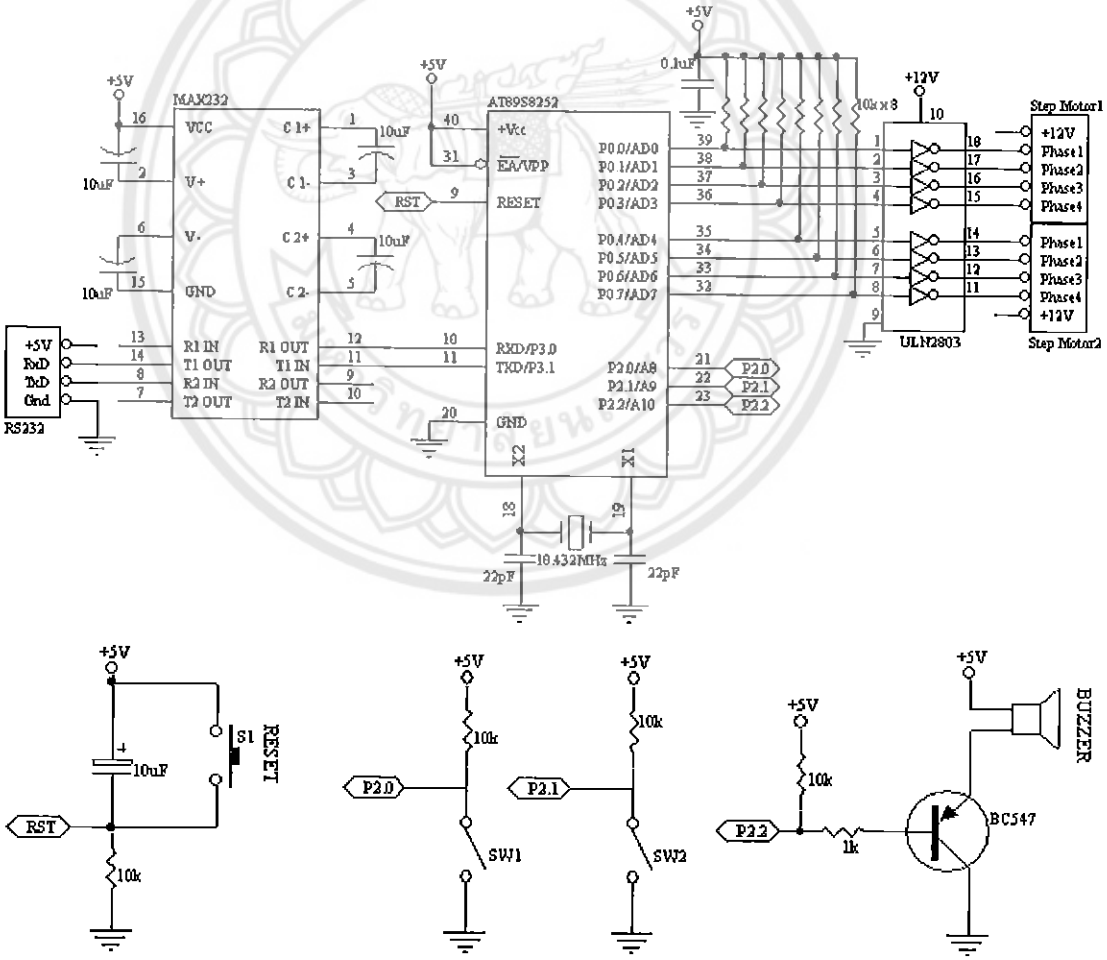
3. คลาส **SerialClient** ทำหน้าที่เป็นสะพาน (Bridge) ในการเชื่อมต่อทีซีพีพอร์ตเข้ากับพอร์ตอนุกรมเพื่อคอยจัดการคำสั่งที่รับเข้ามาทางทีซีพีพอร์ต แล้วส่งข้อมูลไปให้ SerialAccess จัดการส่งคำสั่งออกพอร์ตอนุกรม ซึ่งแสดงคลาสไดอะแกรมดังรูปที่ 3.6



รูปที่ 3.6 คลาสไดอะแกรม SerialClient

3.3.2 การออกแบบวงจรควบคุมการทำงาน

การออกแบบวงจรควบคุมการทำงานนั้น ประกอบด้วยวงจรของไมโครคอนโทรลเลอร์ MCS-51 (ซึ่งเลือกใช้ AT89S8252 ของบริษัท Atmel) [8] ที่ทำการโปรแกรมการควบคุมและสั่งงานสเต็ปปิ้งมอเตอร์ให้หมุนก้ม-เงย, หมุนซ้าย-ขวา และควบคุมการเปิด-ปิดลำโพง Buzzer วงจรไมโครคอนโทรลเลอร์ MCS-51 นี้จะทำการเชื่อมต่อกับวงจรไดร์เวอร์ ซึ่งทำหน้าที่ในการขยายกระแสในการควบคุมการทำงานของมอเตอร์ให้มีประสิทธิภาพสูงสุด การควบคุมสเต็ปมอเตอร์สิ่งสำคัญคือการจ่ายกระแสให้กับขดลวดของมอเตอร์ต้องคงที่เสมอ การออกแบบวงจรเรียงกระแสให้กับระบบจึงจำเป็นต้องมีวงจรขยายกระแสที่มีประสิทธิภาพสูง ในการออกแบบโครงงานนี้ได้ออกแบบวงจรเพาเวอร์ซัพพลายโดยมีไอซีเรกกูเลทเบอร์ 7805 [14] ในการควบคุมแรงดันให้กับไมโครคอนโทรลเลอร์ MCS-51 และมีไอซีเรกกูเลทเบอร์ 7812 [15] ในการควบคุมแรงดันให้กับวงจรไดร์เวอร์ ที่ทำหน้าที่ในการขยายกระแสในการควบคุมการทำงานของสั่งงานสเต็ปปิ้งมอเตอร์



รูปที่ 3.7 ส่วนประกอบของวงจรควบคุมการทำงาน

จากรูปที่ 3.7 ตัวไมโครคอนโทรลเลอร์ MCS-51 จะเป็นตัวควบคุมสั่งงานให้สเต็ปปิ้งมอเตอร์ทำการหมุนก้ม-เงย, หมุนซ้าย-ขวา และควบคุมการเปิด-ปิดลำโพง Buzzer โดยจะรับข้อมูลการสั่งงานจากวงจรเชื่อมต่อพอร์ตอนุกรมของคอมพิวเตอร์ซึ่งมีไอซี MAX232 [9] เป็นตัวทำหน้าที่ในการแปลงข้อมูลระดับ RS-232 ที่รับจากเครื่องคอมพิวเตอร์ไปเป็นระดับทีทีแอล ก่อนที่จะทำการส่งข้อมูลไปให้กับไมโครคอนโทรลเลอร์ MCS-51 เพื่อทำการประมวลผลต่อไป (เนื่องจาก MCS-51 ไม่เข้าใจสัญญาณ RS-232) หน้าที่อีกอย่างของ MAX232 คือแปลงข้อมูลการส่งของไมโครคอนโทรลเลอร์ MCS-51 จากระดับทีทีแอลไปเป็นระดับของ RS-232 เพื่อแสดงการทำงานบนเครื่องคอมพิวเตอร์

พอร์ต 2 ของไมโครคอนโทรลเลอร์ใช้ในการรับอินพุตจากสวิตซ์ 2 ตัว เพื่อจำกัดการหมุนขวา และเงย ซึ่งหน้าที่หลักของสวิตซ์ 2 ตัวนี้ จะใช้เป็นตัวช่วยในการปรับมุมของกล้องให้อยู่ในลักษณะหน้าตรงในขั้นตอนการเริ่มทำงาน จะเห็นได้ว่า P2.0 และ P2.1 จะต่อกับสถานะลอจิก "1" อยู่ตลอดเวลา เมื่อใดก็ตามที่มีอินพุตเข้ามาจากสวิตซ์ตัวใดตัวหนึ่ง สถานะที่พอร์ตของบิตที่ต่อสวิตซ์นั้นอยู่จะมีสถานะลอจิกเป็น "0" นั่นคือจะไม่สามารถให้สเต็ปปิ้งมอเตอร์หมุนในทิศทางนั้นได้ต่อ และจะใช้พอร์ต 0 ของไมโครคอนโทรลเลอร์ในการส่งข้อมูลการสั่งงานสเต็ปปิ้งมอเตอร์ไปให้กับไอซีไดร์สเต็ปปิ้งมอเตอร์ ULN2803 [11] เพื่อทำการขยายกระแสแล้วขับสเต็ปปิ้งมอเตอร์ต่อไป ส่วนพอร์ต P2.2 จะใช้เป็นเอาต์พุตในการควบคุมการเปิด-ปิดลำโพง Buzzers โดยที่จะใช้ทรานซิสเตอร์แบบพีเอ็นพีเบอร์ BC547 [16] ในการทำหน้าที่เป็นสวิตซ์ตัดต่อการทำงานของวงจร คือในสภาวะที่เป็นลอจิก "0" ลำโพงก็จะไม่มีเสียง และเมื่อได้รับลอจิก "1" ก็จะมีเสียง

เมื่อต้องการให้สเต็ปปิ้งมอเตอร์ที่ทำหน้าที่หมุนซ้าย-ขวาทำการหมุน [10] จะต้องส่งข้อมูล "1" ไล่เรียงไปตามลำดับจาก P0.0 ถึง P0.3 แล้ววนกลับมาที่ P0.0 ใหม่ หากต้องการให้สเต็ปปิ้งมอเตอร์หมุนกลับทิศทางก็ให้ส่งข้อมูลย้อนกลับ โดยเริ่มจาก P0.3 ก่อนแล้วสิ้นสุดรอบที่ P0.0 แล้ววนกลับมาที่ P0.3 ใหม่ ถ้าต้องการให้สเต็ปปิ้งมอเตอร์ที่ทำหน้าที่หมุนก้ม-เงย ทำการหมุน จะต้องส่งข้อมูล "1" ไล่เรียงไปตามลำดับทำนองเดียวกับสเต็ปปิ้งมอเตอร์ที่ทำหน้าที่หมุนซ้าย-ขวา แต่จะเรียงจาก P0.4 ถึง P0.7 แล้ววนกลับมาที่ P0.4 ใหม่ หากต้องการให้สเต็ปปิ้งมอเตอร์หมุนกลับทิศทางก็ให้ส่งข้อมูลย้อนกลับ โดยเริ่มจาก P0.7 ก่อนแล้วสิ้นสุดรอบที่ P0.4 แล้ววนกลับมาที่ P0.7 ใหม่ เมื่อไอซีไดร์สเต็ปปิ้งมอเตอร์ ULN2803 ได้รับข้อมูล "1" วงจรก็จะทำงาน ทำให้เกิดกระแส กระแสไฟฟ้าไหลผ่านขดลวดของมอเตอร์ที่ต่ออยู่ ส่งผลให้เกิดการเคลื่อนที่ของแกนมอเตอร์ขึ้น

ในการเขียนโปรแกรมกระตุ้นและควบคุมการหมุนของสเต็ปปิ้งมอเตอร์ให้เคลื่อนที่จะใช้แบบหนึ่งเฟสหรือแบบฟูลสเต็ป (Full Step) เพราะง่ายในการควบคุมสั่งงานและไม่ต้องใช้กำลังไฟมาก การเขียนโปรแกรมควบคุมสเต็ปปิ้งมอเตอร์จะใช้การรับข้อมูลจากพอร์ต RS-232 แล้วนำมาตรวจสอบ ซึ่งคำสั่งทั้งหมดแสดงดังตารางที่ 3.1 การสั่งงานควบคุมสเต็ปปิ้งมอเตอร์ในการหมุนซ้าย-ขวาและก้ม-เงยนั้น จะมีรูปการหมุนอยู่ 3 แบบซึ่งในการหมุนแต่ละแบบจะแตกต่างกันตรง

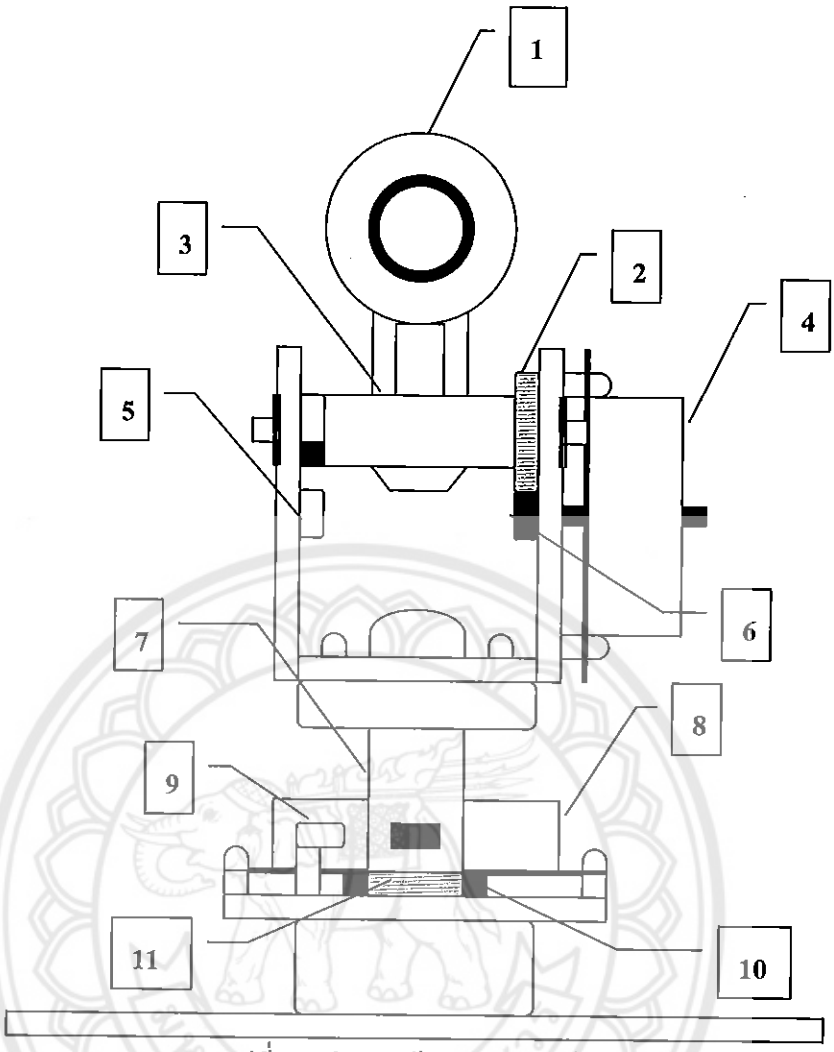
จำนวนของสเต็ปที่กระตุ้นให้กับสเต็ปปิ้งมอเตอร์ในการสั่งงานหนึ่งครั้ง มีผลทำให้อัตราการเปลี่ยนมุมกล้องในแต่ละแบบต่างกันในการสั่งงานหนึ่งครั้ง โดยที่สเต็ป 1 จะมีอัตราการเปลี่ยนมุมน้อยที่สุด และสเต็ป 3 จะมีอัตราการเปลี่ยนมุมมากที่สุด ส่วนคำสั่งควบคุมการเปิด-ปิด ลำโพง Buzzer จะใช้ M และ N ในการทำงานของโปรแกรมจะเริ่มจากการจัดตำแหน่งของกล้องให้อยู่ในแนวหน้าตรง โดยที่สเต็ปปิ้งมอเตอร์ที่ทำหน้าที่หมุนซ้าย-ขวาจะทำการหมุนไปทางขวาเรื่อยๆ ทีละสเต็ป จนกว่าสวิตซ์คุมเขตจำกัดการปรับมุมขวาถูกกดลง ก็จะหมุนกลับมาทางซ้าย 90 องศาเพื่อตั้งฐานกล้องให้อยู่ตำแหน่งหน้าตรง จากนั้นที่สเต็ปปิ้งมอเตอร์ที่ทำหน้าที่หมุนก้ม-เงย ก็จะหมุนขึ้นไปเรื่อยๆ ทีละสเต็ปจนกว่าสวิตซ์คุมเขตจำกัดการปรับมุมเงยจะถูกกดลง ก็จะหมุนลง 45 องศาเพื่อตั้งหน้ากล้องให้อยู่ในแนวตั้งตรง จากนั้นก็จะหยุดเพื่อรอคำสั่ง การตั้งหน้ากล้องนี้จะเกิดขึ้นทุกครั้งที่มีการรีเซตระบบ หรือเมื่อมีการเชื่อมต่อจากโปรแกรมฝั่งเครื่องลูกข่ายเข้ามา ซึ่งโปรแกรมฝั่งเครื่องลูกข่ายเมื่อทำการเชื่อมต่อได้สำเร็จจะส่ง "O" มาเป็นคำสั่งให้ระบบทำการจัดมุมกล้อง ส่วนในการควบคุมขอบเขตการหมุนซ้าย-ขวา และก้ม-เงย ในขั้นตอนการรับคำสั่งจากผู้ใช้จะใช้รีจิสเตอร์ 2 ตัวเป็นตัวจำกัดการหมุน โดยที่จะทำการกำหนดค่าให้กับรีจิสเตอร์ไว้ค่าหนึ่ง เมื่อหน้ากล้องตั้งอยู่ในตำแหน่งหน้าตรงค่าในรีจิสเตอร์นั้นจะมีค่าเป็นครึ่งหนึ่งของค่าที่กำหนดไว้ แล้วพอมีการหมุนทางขวาค่าก็จะลดลงเรื่อยๆ จนมีค่าเป็นศูนย์ก็จะไม่สามารถหมุนต่อไปได้ ในทางกลับกันเมื่อมีการหมุนทางซ้ายค่าก็จะเพิ่มขึ้นเรื่อยๆ จนมีค่าเท่ากับค่าที่กำหนดไว้ก็จะไม่สามารถหมุนต่อไปได้ รีจิสเตอร์ที่ควบคุมขอบเขตการหมุนก้ม-เงยก็ใช้หลักการเดียวกัน ต่างกันเพียงค่าขอบเขตที่กำหนดเท่านั้น

ตารางที่ 3.1 คำสั่งที่ใช้ควบคุมสเต็ปปิ้งมอเตอร์

คำสั่ง	สเต็ป 1	สเต็ป 2	สเต็ป 3
เงย	A	E	I
ก้ม	B	F	J
หมุนขวา	C	G	K
หมุนซ้าย	D	H	L

3.3.3 การออกแบบโครงสร้างของฐานกล้อง

ฐานกล้องสามารถเคลื่อนที่ได้ในลักษณะ หันซ้าย-ขวา และ ก้ม-เงย ทั้งหมดสี่ทิศทาง การทำงานจะถูกควบคุมโดยอุปกรณ์อิเล็กทรอนิกส์ และโปรแกรมของไมโครคอนโทรลเลอร์ โครงสร้างทั้งหมดของฐานกล้องทำจากอะลูมิเนียมและเหล็ก ซึ่งมีส่วนประกอบต่างๆ ดังนี้

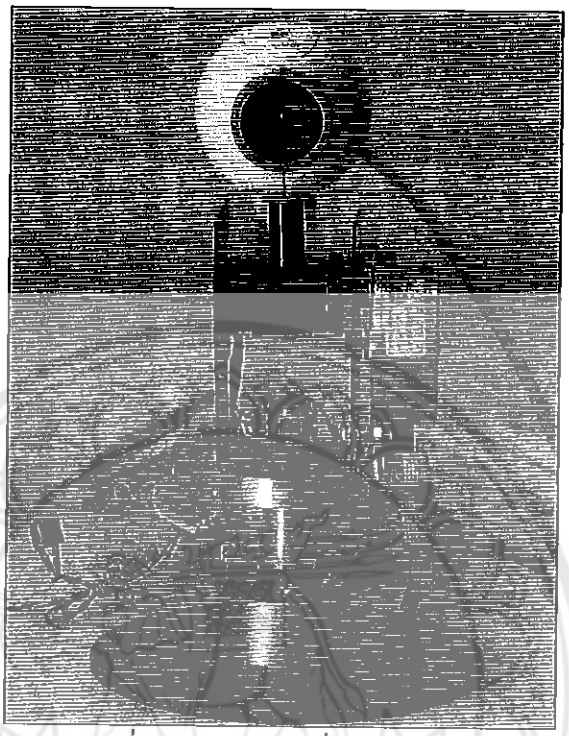


รูปที่ 3.8 โครงสร้างของฐานกล้อง

ส่วนประกอบต่างๆ มีดังนี้

1. กล้องเว็บแคม (Web Cam)
2. เฟืองแกนหมุนปรับมุมก้ม-เงย
3. แกนหมุนปรับมุมก้ม-เงย
4. สเต็ปปิ้งมอเตอร์ปรับมุมก้ม-เงย
5. สวิตซ์เซนเซอร์คุมเขตจำกัดการเงย
6. เฟืองสเต็ปปิ้งมอเตอร์ปรับมุมก้ม-เงย
7. แกนหมุนปรับมุมซ้าย-ขวา
8. สเต็ปปิ้งมอเตอร์ปรับมุมซ้าย-ขวา
9. สวิตซ์เซนเซอร์คุมเขตจำกัดการปรับมุมขวา
10. เฟืองสเต็ปปิ้งมอเตอร์ปรับมุมซ้าย-ขวา
11. เฟืองแกนหมุนปรับมุมซ้าย-ขวา

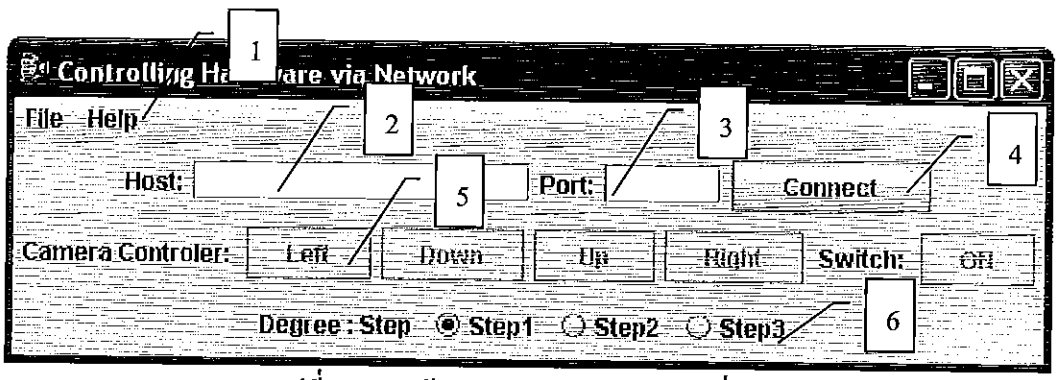
เมื่อทำการประกอบชิ้นส่วนต่างๆ เสร็จแล้วก็จะมีลักษณะดังรูปที่ 3.8 ซึ่งในแต่ละส่วนนั้นได้ออกแบบให้มีจุดยึดที่สามารถถอดออกได้สะดวก เพื่อประโยชน์ในการปรับปรุงประสิทธิภาพต่อไป และเมื่อมีการสร้างจนสำเร็จก็มีลักษณะตรงตามที่ออกแบบไว้ดังแสดงในรูปที่ 3.9



รูปที่ 3.9 ฐานกล้องเมื่อสร้างสำเร็จ

3.4 การออกแบบโปรแกรมส่วนเครื่องลูกข่าย

โปรแกรมในส่วนนี้จะเป็นโปรแกรมที่รันอยู่บนเครื่องลูกข่าย มีหน้าที่ในการสร้างการเชื่อมต่อกับโปรแกรม TCP2Serial บนเครื่องแม่ข่าย เพื่อทำการส่งคำสั่งควบคุมการหมุนของสเต็ปปิ้งมอเตอร์ในการเปลี่ยนมุมมองของกล้อง และเปิด-ปิด ลำโพง Buzzer ที่ได้รับจากผู้ใช้ โดยจะสร้างเป็นจาวาแอพพลิเคชั่นที่มีลักษณะเป็น GUI ดังแสดงในรูปที่ 3.10



รูปที่ 3.10 หน้าจอของโปรแกรมส่วนเครื่องลูกข่าย

ส่วนประกอบต่างๆ ของหน้าจอของโปรแกรม ดังนี้

1. เมนูการทำงานของโปรแกรมซึ่งจะสามารถสั่งปิดโปรแกรมหรือสั่งเชื่อมต่อได้ผ่านทางเมนูย่อยในเมนู File และมีส่วนคำแนะนำการใช้งานโปรแกรมในเมนู Help

2. ช่องสำหรับใส่แอดเดรสของเครื่องแม่ข่ายซึ่งสามารถใส่เป็นหมายเลขไอพีของเครื่องหรือชื่อโดเมนก็ได้ เพื่อเป็นการบอกที่อยู่ของเครื่องแม่ข่ายที่จะทำการเชื่อมต่อ

3. ช่องสำหรับใส่เบอร์พอร์ตของเครื่องแม่ข่ายที่เปิดบริการไว้ เพื่อเป็นการติดต่อกับเครื่องแม่ข่ายในการส่งข้อมูลผ่านทางพอร์ตนี้

4. ปุ่มสำหรับสั่งการเริ่มการเชื่อมต่อกับเครื่องแม่ข่ายตามแอดเดรสและพอร์ตที่ใส่ไว้ และเมื่อทำการเชื่อมต่อได้สำเร็จปุ่มนี้จะทำหน้าที่กลับจากเดิมคือจะเป็นปุ่มที่สั่งตัดการเชื่อมต่อ

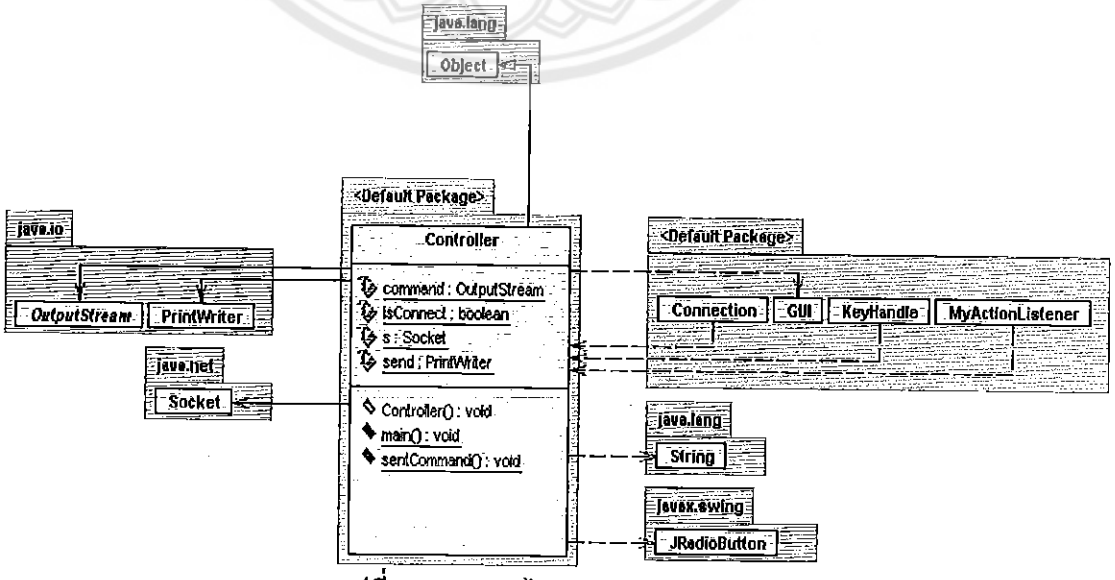
5. ปุ่มสำหรับควบคุมการปรับมุมของกล้องและเปิด-ปิดลำโพง Buzzer ปุ่มนี้จะสามารถรับการสั่งงานจากผู้ใช้ได้ก็ต่อเมื่อทำการเชื่อมต่อได้สำเร็จเท่านั้น โดยที่ปุ่มควบคุมการหมุนของกล้องสามารถรับคำสั่งจากเมาส์หรือคีย์ขึ้น-ลง ซ้าย-ขวา บนคีย์บอร์ดแทนได้

6. ช่องสำหรับเลือกอัตราการเปลี่ยนมุมกล้องต่อการสั่งงานหนึ่งครั้ง ทำการเลือกรูปแบบก่อนทำการสั่งงานเพื่อกำหนดอัตราการเปลี่ยนมุม

ส่วนประกอบของโปรแกรมแบ่งย่อยออกเป็นทั้งหมด 4 คลาส ซึ่งแต่ละคลาสก็มีหน้าที่ต่างๆ กันออกไป ซึ่งรายละเอียดของแต่ละคลาสแสดงได้ดังนี้

3.4.1 คลาส Controller

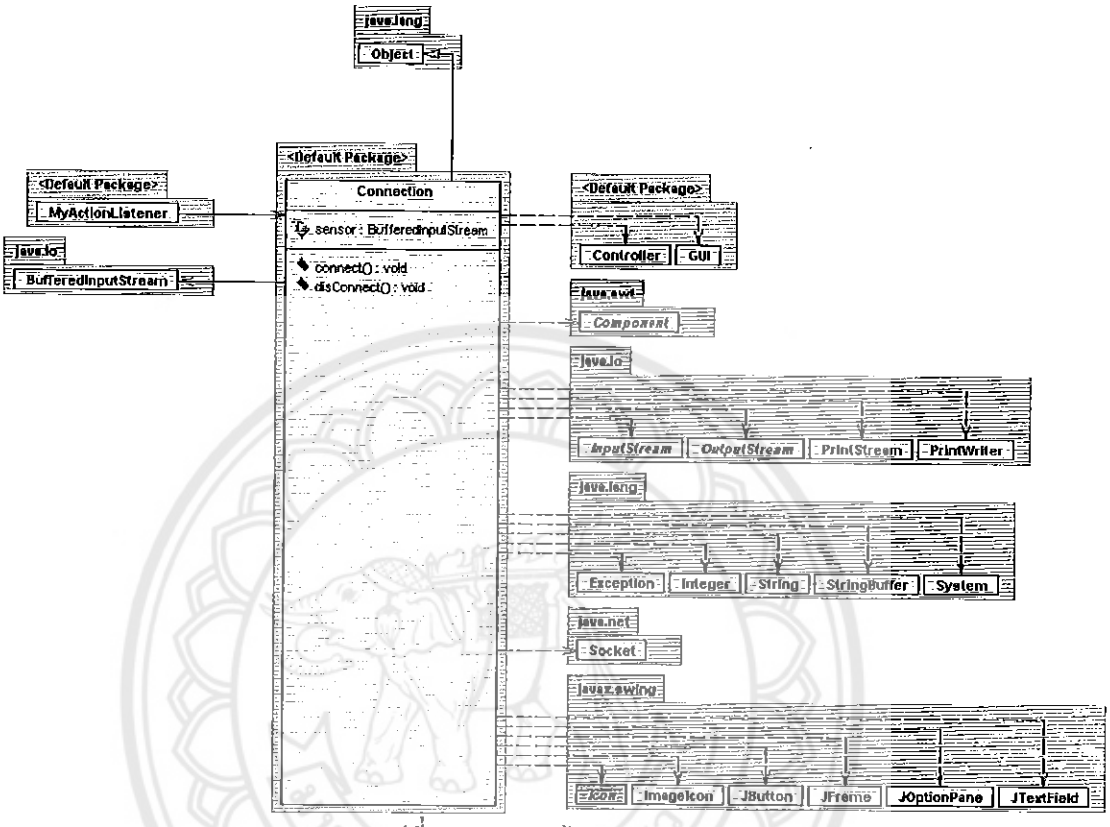
ทำหน้าที่เป็นคลาสหลักในการทำงานของโปรแกรม และเป็นตัวที่คอยจัดการกับคำสั่งที่รับมาจากผู้ใช้แล้วทำการส่งไปให้โปรแกรม TCP2Serial ที่รันอยู่บนเครื่องแม่ข่าย ซึ่งคลาสไดอะแกรมแสดงดังรูปที่ 3.11



รูปที่ 3.11 คลาสไดอะแกรม Controller

3.4.2 คลาส Connection

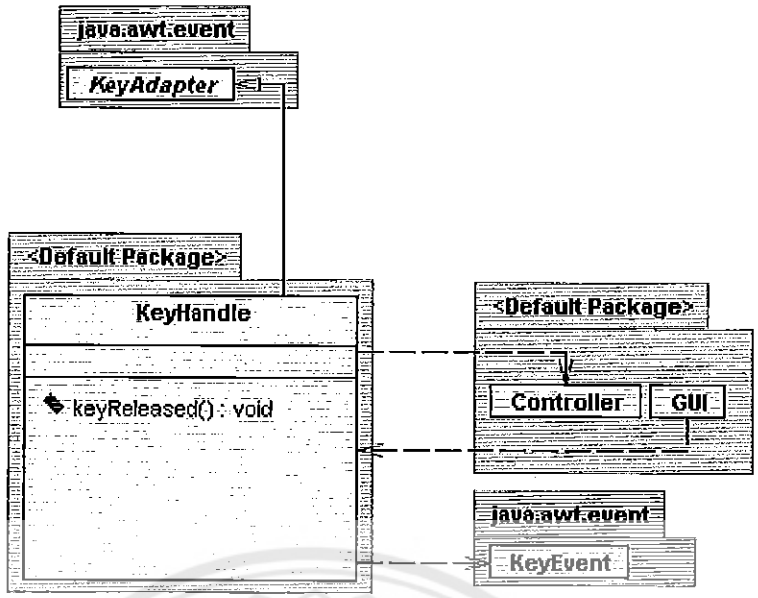
ทำหน้าที่ในการสร้างการเชื่อมต่อหรือตัดการเชื่อมต่อกับเครื่องแม่ข่าย ซึ่งคลาสไดอะแกรมแสดงดังรูปที่ 3.12



รูปที่ 3.12 คลาสไดอะแกรม Connection

3.4.3 คลาส KeyHandle

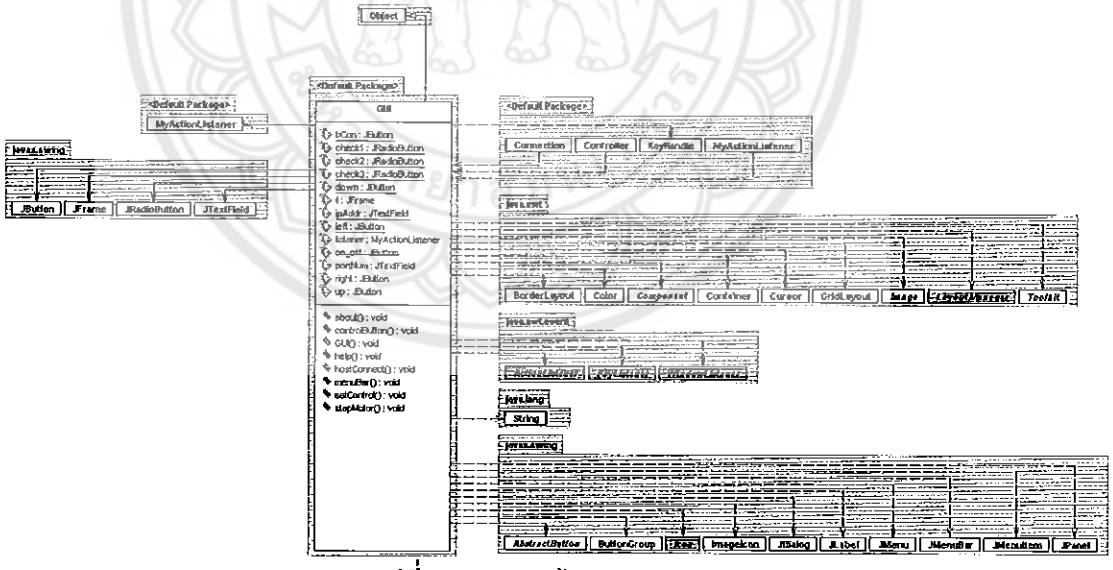
ทำหน้าที่ในการจัดการกับเหตุการณ์ในการกดคีย์บอร์ด เพื่อตรวจสอบการกดคีย์ขึ้น-ลง ซ้าย-ขวาบนคีย์บอร์ด จากนั้นส่งไปให้คลาส Controller ในการส่งคำสั่งที่ได้รับจากคีย์ที่กดไปให้เครื่องแม่ข่าย ซึ่งคลาสไดอะแกรมแสดงดังรูปที่ 3.13



รูปที่ 3.13 คลาสไดอะแกรม KeyHandle

3.4.4 คลาส GUI

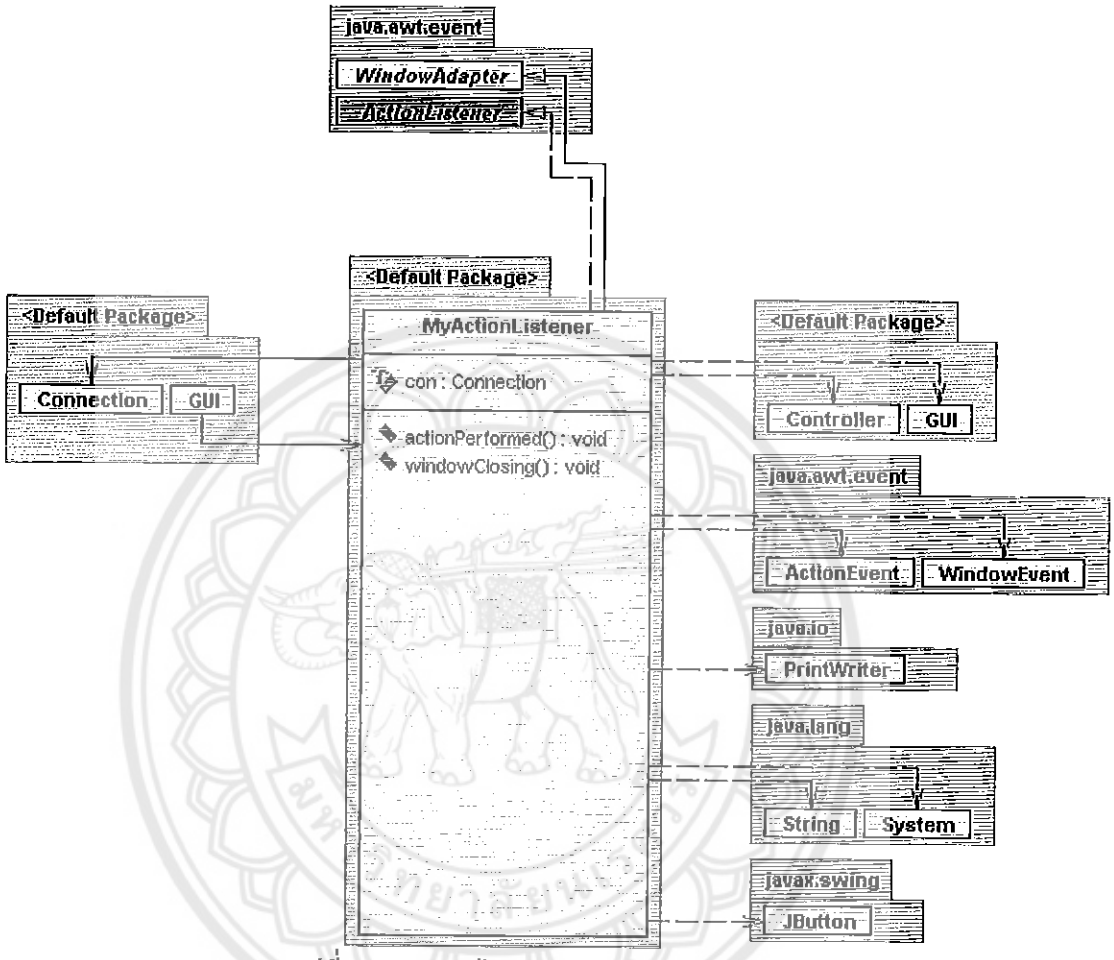
ทำหน้าที่ในการจัดการรูปแบบหน้าจอของโปรแกรม โดยที่จะรวมองค์ประกอบต่างๆ ของหน้าจอของโปรแกรมอยู่ที่คลาสนี้ ซึ่งคลาสไดอะแกรมแสดงดังรูปที่ 3.14



รูปที่ 3.14 คลาสไดอะแกรม GUI

3.4.5 คลาส MyActionListener

ทำหน้าที่ในการคอยจัดการกับเหตุการณ์ที่เกิดขึ้นบนหน้าจอของโปรแกรมทั้งหมด เช่น การคลิกเมาส์ที่ปุ่มแล้วให้เกิดกระบวนการอะไรขึ้น เป็นต้น ซึ่งคลาสไดอะแกรมแสดงดังรูปที่ 3.15

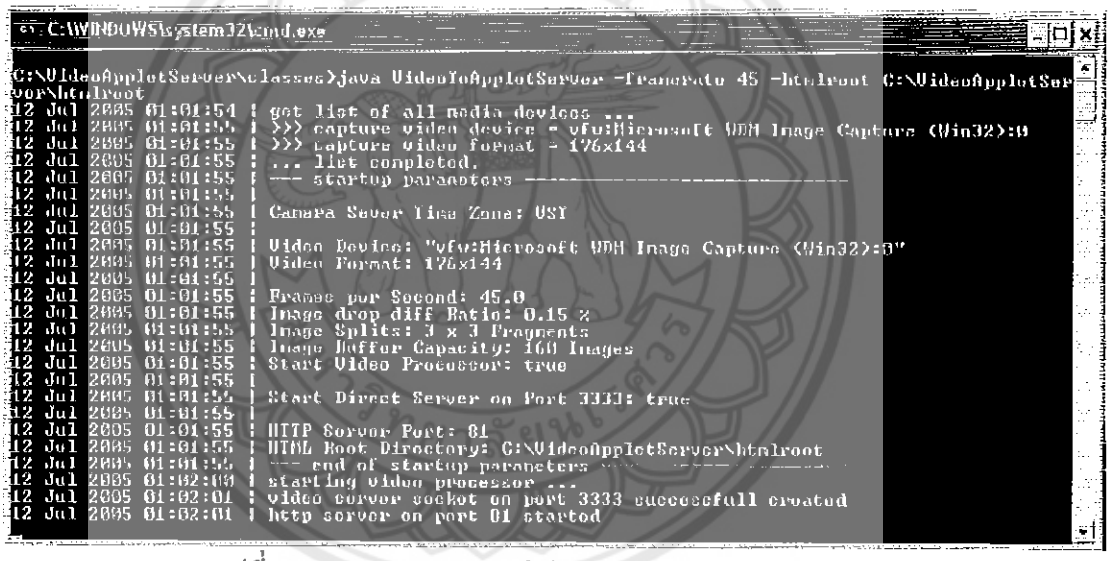


รูปที่ 3.15 คลาสไดอะแกรม MyActionListener

บทที่ 4
ผลการติดตั้งใช้งาน

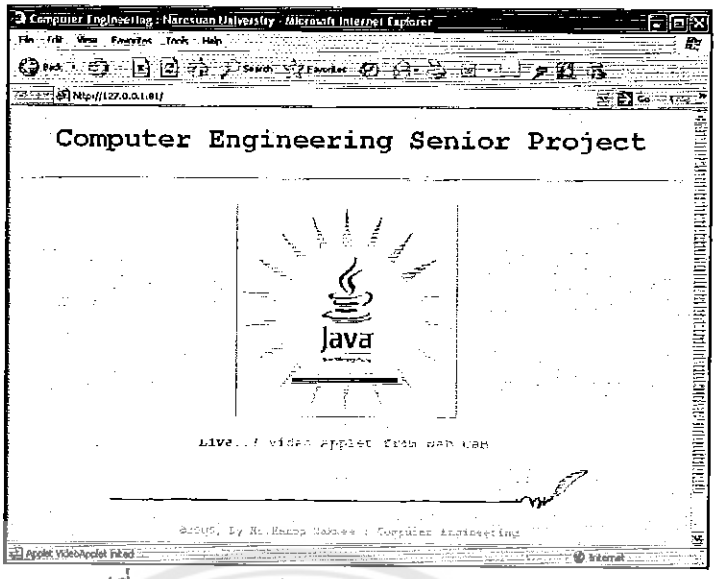
4.1 การให้บริการรับชมภาพวีดีโอแบบเรียลไทม์

เมื่อเครื่องแม่ข่ายรันโปรแกรม VideoToAppletServer ขึ้นแล้ว และตัวของโปรแกรมได้ทำการกำหนดค่าเริ่มต้นต่าง ๆ ตามที่ได้กำหนดไว้อย่างถูกต้องแล้ว ซึ่งผลการรันโปรแกรมแสดงได้ดังรูปที่ 4.1 เครื่องแม่ข่ายก็พร้อมที่จะรับภาพจากกล้องเว็บแคม ที่ต่ออยู่ทางพอร์ตยูเอสบี พร้อมกันนั้นพร้อมกันก็มีกระบวนการจัดการกับภาพที่รับเข้ามาและแปลงรูปแบบของภาพเพื่อเตรียมส่งให้กับเครื่องลูกข่ายที่ขอบริการเข้ามา และสร้างส่วน HTML Server เพื่อเป็นส่วนให้ผู้ใช้เรียกใช้งานส่วนแสดงผลผ่านทางเว็บบราวเซอร์



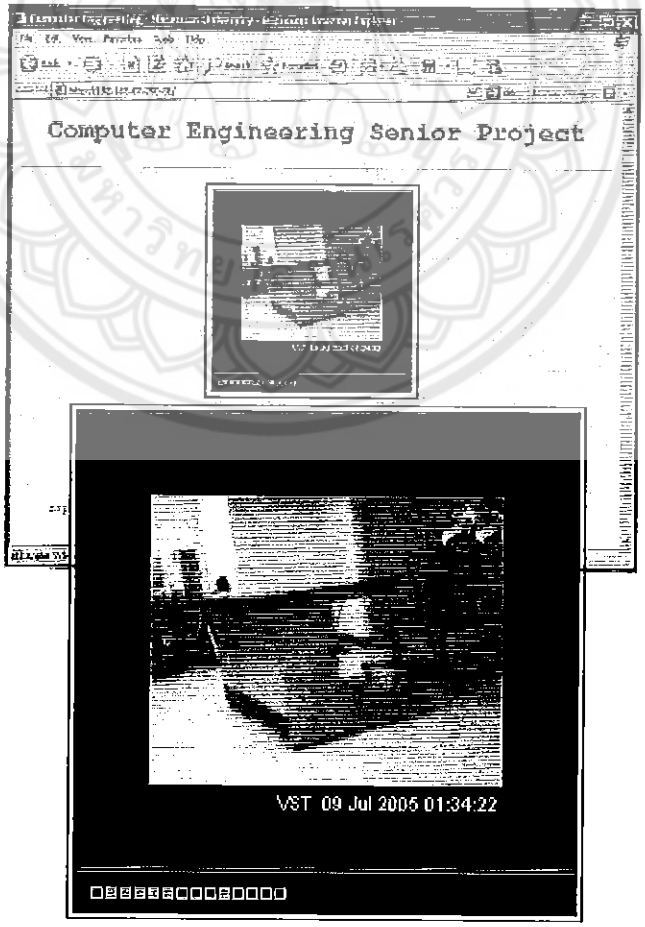
รูปที่ 4.1 ผลการทำงานของโปรแกรม VideoToAppletServer

การรับชมภาพวีดีโอสามารถทดสอบได้โดย เปิดเว็บบราวเซอร์ของเครื่องแม่ข่าย และใส่ Address: http://127.0.0.1:81 ผลที่ได้คือจะเป็นการเรียกหน้าเว็บเพจที่อยู่ใน HTML ROOT ที่สร้างไว้ขึ้นมาแสดงผล โดยการแสดงผลภาพวีดีโอจะแสดงผ่านจาวาแอพเพล็ต ที่ถูกแทรกไว้ในเว็บเพจ เมื่อหน้าหลักถูกเรียกขึ้น ก็จะมีการเรียก Java Runtime Enviroment (JRE) เพื่อใช้เป็นปลั๊กอิน ในการแสดงผลจาวาแอพเพล็ตดังแสดงในรูปที่ 4.2



รูปที่ 4.2 ผลการเรียก Java Runtime Environment

เมื่อทำการเรียกใช้ JRE เสร็จแล้วโปรแกรมก็จะเชื่อมต่อกับเครื่องแม่ข่ายผ่านทางพอร์ตเบอร์ 3333 ที่ใช้ส่งข้อมูลภาพ เพื่อนำภาพที่ได้รับจากกล้องเว็บแคมมาแสดงผล ซึ่งผลที่ได้แสดงดังรูปที่ 4.3

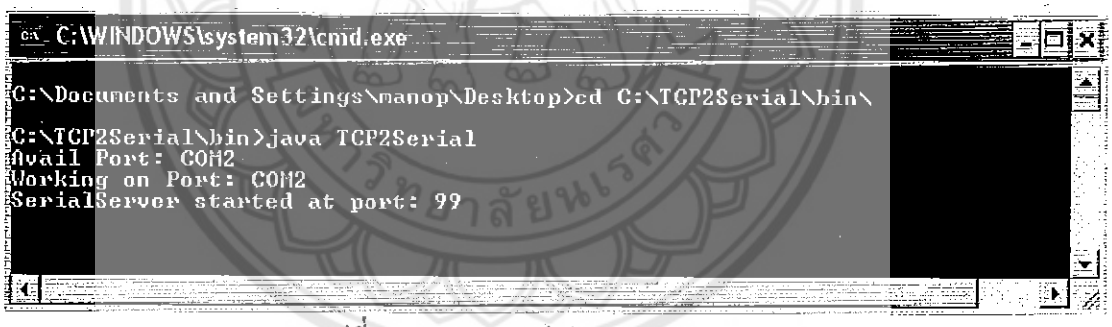


รูปที่ 4.3 ผลการแสดงภาพวีดีโอ

การเข้าใช้บริการรับชมภาพของผู้ใช้ทางเครื่องลูกข่ายก็เพียงใส่แอดเดรสของเครื่องแม่ข่ายและเบอร์พอร์ตที่เครื่องแม่ข่ายเปิดให้บริการไว้ เช่น เครื่องแม่ข่ายมี IP Address 192.168.67.250 ก็เรียกใช้ที่ http://192.168.67.250:81 การแสดงผลที่ได้ก็จะมีลักษณะเหมือนกับการทดสอบที่เครื่องแม่ข่าย แต่ภาพที่ได้รับอาจมีคุณภาพในการแสดงผลที่แย่กว่า อันเนื่องมาจากความเร็วของระบบเครือข่ายในการรับส่งข้อมูลอาจช้า ทำให้เกิดความไม่ต่อเนื่องของภาพขึ้น และในการที่กำหนดค่าความแตกต่างของภาพที่จะทำการแสดงไว้ ผลที่ได้คือถ้าไม่มีการเคลื่อนไหวผ่านทางหน้ากล้องภาพที่กล้องรับได้ก็จะเป็นภาพเดิม ทำให้ไม่มีความแตกต่างของภาพเกิดขึ้น ผลก็คือส่วนแสดงผลก็จะแสดงภาพล่าสุดค้างไว้

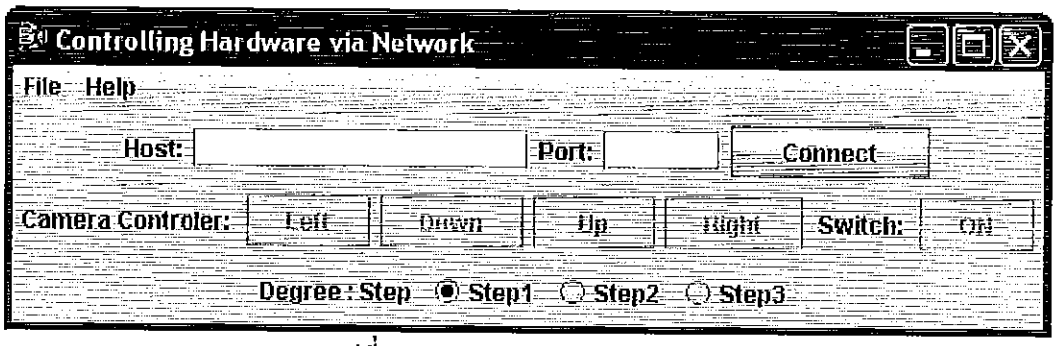
4.2 การควบคุมการหมุนของกล้องและเปิดปิดอุปกรณ์ไฟฟ้า

เมื่อเครื่องแม่ข่ายรันโปรแกรม TCP2Serial ขึ้น โดยทำการกำหนดค่าคอมพอร์ตที่ใช้งานสื่อสารกับไมโครคอนโทรลเลอร์ และเบอร์พอร์ตทีซีพีที่จะเปิดให้บริการกับเครื่องแม่ข่ายในการเข้าเชื่อมต่อแล้ว ซึ่งผลการรันของโปรแกรมแสดงได้ดังรูปที่ 4.4 โปรแกรมก็จะรอการเชื่อมต่อจากเครื่องลูกข่ายจากโปรแกรม Controller เพื่อรับคำสั่งส่งไปให้ไมโครคอนโทรลเลอร์เพื่อตีความหมายและปฏิบัติตามคำสั่งที่ได้รับต่อไป

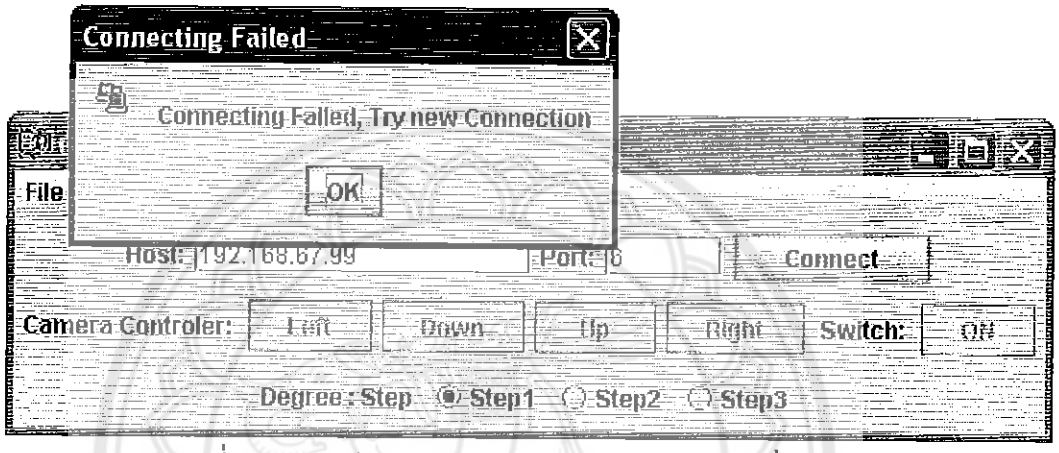


รูปที่ 4.4 ผลการรันโปรแกรม TCP2Serial

เมื่อเครื่องลูกข่ายรันโปรแกรม Controller ซึ่งผลการรันแสดงดังรูปที่ 4.5 เมื่อจะทำการเชื่อมต่อ ขั้นตอนแรกคือทำการใส่หมายเลขไอพีแอดเดรสหรือยูอาร์แอล (URL) และเบอร์พอร์ตให้ตรงกับที่เครื่องแม่ข่ายเปิดให้บริการไว้ แล้วทำการเชื่อมต่อ ถ้าการเชื่อมต่อไม่สามารถทำได้สำเร็จก็จะมีการแจ้งเตือนจากโปรแกรมดังแสดงในรูปที่ 4.6

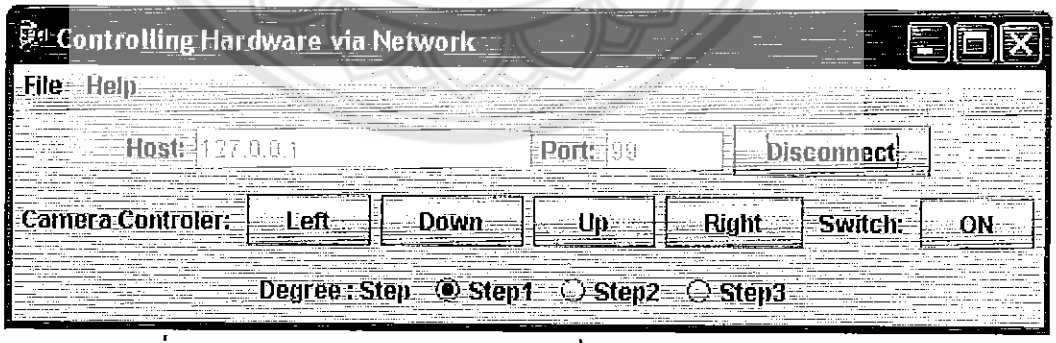


รูปที่ 4.5 ผลการรันโปรแกรม Controller



รูปที่ 4.6 การแจ้งจากโปรแกรมว่าไม่สามารถทำการเชื่อมต่อได้

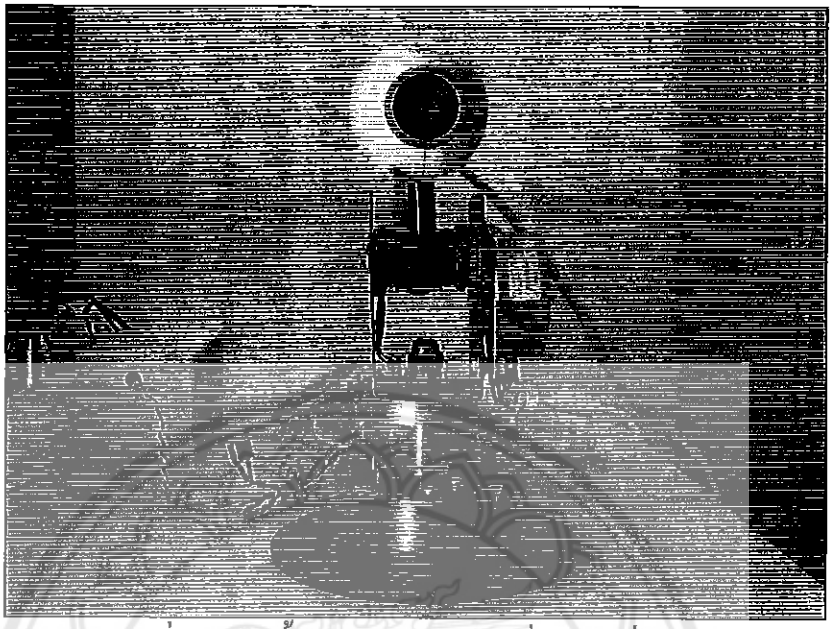
ถ้าการเชื่อมต่อสามารถทำได้สำเร็จ ปุ่มสำหรับใช้รับคำสั่งควบคุมจากผู้ใช้ก็จะสามารถรับคำสั่งได้ ดังแสดงรูปที่ 4.7



รูปที่ 4.7 โปรแกรม Controller สามารถเชื่อมต่อกับ TCP2Serial ได้สำเร็จ

โปรแกรม TCP2Serial ก็จะแสดงหมายเลข IP Address ของเครื่องที่ทำการเชื่อมต่อเข้ามา เมื่อมีการเชื่อมต่อสำเร็จโปรแกรม Controller ก็จะส่งคำสั่งเป็น ASCII ตัว "O" เมื่อโปรแกรม TCP2Serial ได้รับแล้วก็จะส่งข้อความที่ได้รับไปให้ไมโครคอนโทรลเลอร์ ผ่านทางพอร์ตอนุกรม

เพื่อให้ไมโครคอนโทรลเลอร์นำไปประมวลผลว่าต้องเริ่มทำการปรับตำแหน่งของกล้องให้อยู่ในตำแหน่งตั้งตรงไปทางด้านหน้า ดังแสดงในรูปที่ 4.8



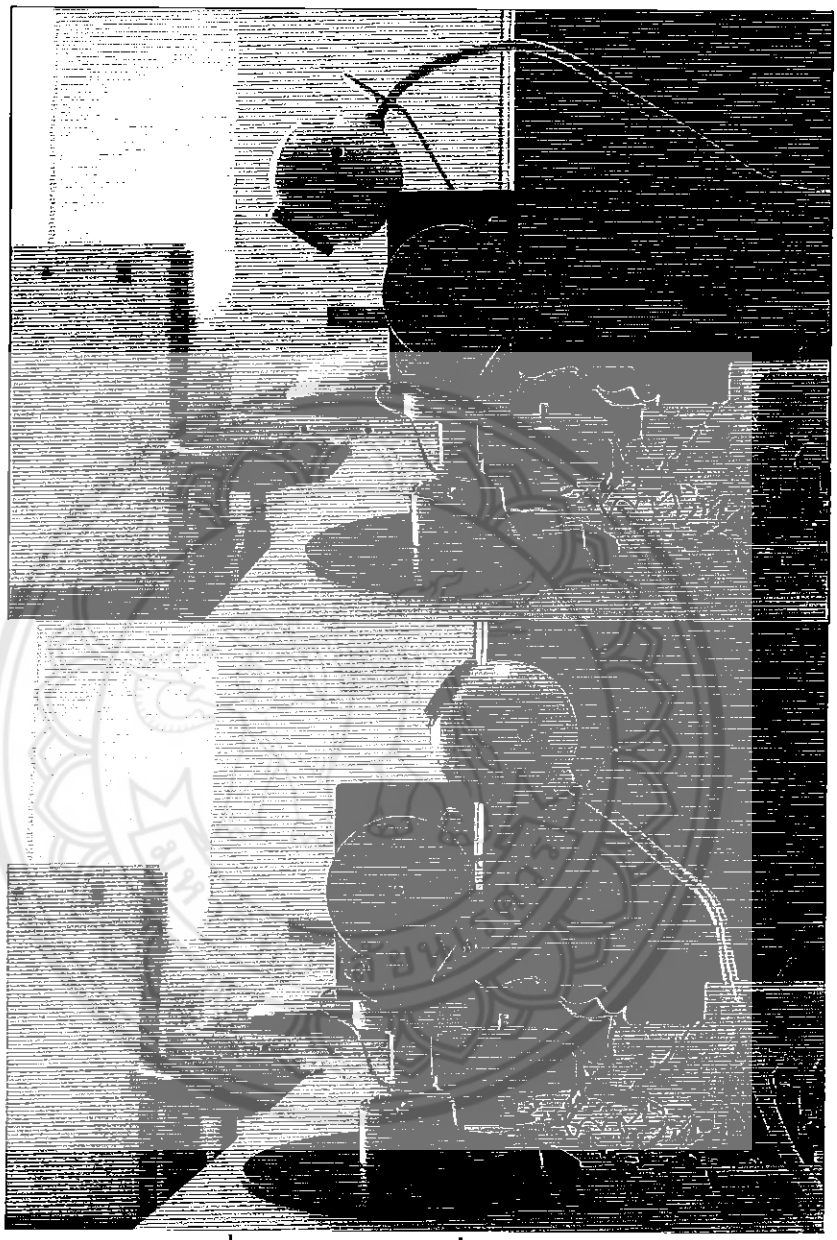
รูปที่ 4.8 การตั้งหน้าตรงของกล้องเมื่อมีการเชื่อมต่อ

เมื่อสั่งงานโดยการสั่งให้มีการเปลี่ยนมุมไปทางซ้าย-ขวา หรือ ขึ้น-ลง ผลที่ได้คือไมโครคอนโทรลเลอร์จะรับคำสั่งที่ได้มาจากการสั่งงาน และนำมาปฏิบัติตามความหมายของแต่ละคำสั่งที่กำหนดเป็นข้อตกลงไว้ คือจะสั่งให้สเต็ปปิ้งมอเตอร์หมุนเพื่อเปลี่ยนมุมกล้องตามคำสั่งที่ได้รับอย่างถูกต้อง โดยที่อัตราการเปลี่ยนไปของมุมกล้องต่อการสั่งงานหนึ่งครั้งจะเป็นไปตามค่า Degree : Step ซึ่งมุมการเปลี่ยนไปของแต่ละสเต็ปแสดงในตารางที่ 4.1 โดยการเปลี่ยนมุมในแนวซ้าย-ขวา จะอยู่ใน 180 องศา โดยที่หน้ากล้องตั้งตรงแล้วจะสามารถหมุนทางซ้ายได้ 90 องศา และขวาอีก 90 องศา ส่วนการหมุนขึ้น-ลง ก็จะทำมุมได้ 90 องศา โดยที่หน้ากล้องตั้งตรงแล้วจะสามารถหมุนขึ้นได้ 45 องศา และลงอีก 45 องศา ตัวอย่างการเปลี่ยนมุมของกล้องแสดงดังรูปที่ 4.9

ตารางที่ 4.1 อัตราการเปลี่ยนมุมของกล้อง

สเต็ป	องศา : การสั่งงาน 1 ครั้ง
1	18
2	30
3	45

ส่วนควบคุมการเปิด-ปิด อุปกรณ์ไฟฟ้าที่จำลองโดยใช้ Buzzer นั้น เมื่อรับคำสั่ง “ON” จากโปรแกรม Controller แล้ว Buzer ก็จะมีเสียงดังขึ้นและเมื่อได้รับคำสั่ง “OFF” เสียงก็จะเงียบ



รูปที่ 4.9 ตัวอย่างการเปลี่ยนมุมของกล้อง

บทที่ 5
บทวิจารณ์และสรุป

5.1 บทวิจารณ์

โครงงานการควบคุมอุปกรณ์ผ่านระบบเครือข่าย เมื่อเริ่มการพัฒนาและติดตั้งใช้งานแล้ว ได้พบปัญหาหลาย ๆ ที่ด้วยกัน ดังนี้

5.1.1 การส่งภาพวีดีโอ (Video) ในลักษณะที่ต้องการให้เป็นเรียลไทม์ (Real time) นั้น ค่อนข้างจะมีปัญหาอยู่ที่ระบบเครือข่าย ถ้าระบบเครือข่ายที่มีความเร็วในการส่งผ่านข้อมูลต่ำ ภาพที่ได้รับชมก็จะมีการกระตุกทำให้การรับชมมีปัญหา ภาพที่แสดงจึงต้องมีขนาดเล็กเพื่อแก้ปัญหาตรงจุดนี้

5.1.2 การสร้างโปรแกรมส่วนติดต่อกับผู้ใช้ (GUI) โดยใช้ภาษาจาวานั้นจะจัดรูปแบบยาก ถ้าไม่มีการใช้เครื่องมือ (IDE) ที่มีลักษณะเป็น Visual Programming

5.1.3 ส่วนฮาร์ดแวร์ (Hardware) ควบคุมการเปลี่ยนมุมมองของกล้องนั้น มีการทำงานที่ไม่ค่อยนุ่มนวล เนื่องมาจากสเต็ปปิ้งมอเตอร์ (Stepping Motor) และชุดเฟืองต่าง ๆ ยังทำงานได้ไม่สัมพันธ์กัน

5.1.4 การควบคุมสั่งงานของส่วนไมโครคอลโทรลเลอร์ (Microcontroller) กับการควบคุมการหมุนของสเต็ปปิ้งมอเตอร์ (Stepping Motor) ยังเกิดปัญหาจากสัญญาณรบกวน (noise) ทำให้การทำงานยังเกิดข้อผิดพลาด

5.2 สรุป

โครงงานการควบคุมอุปกรณ์ผ่านระบบเครือข่าย เป็นโครงงานที่นำประโยชน์ของระบบเครือข่ายคอมพิวเตอร์ มาประยุกต์ใช้ในการติดต่อสื่อสารและควบคุมอุปกรณ์จากระยะไกล ซึ่งในการพัฒนาโครงงานก็ได้มีทั้งส่วนที่เป็นซอฟต์แวร์ (Software) และฮาร์ดแวร์ (Hardware) แยกหน้าที่กันทำงานตามหน้าที่ของแต่ละส่วน ซึ่งได้ผลตามวัตถุประสงค์ที่ตั้งไว้ ดังนี้

5.2.1 ส่วนที่เป็นเครื่องแม่ข่าย (Server) ที่คอยให้บริการกับเครื่องลูกข่ายในการรับชมภาพวีดีโอแบบเรียลไทม์ (Real time) จากกล้องเว็บแคม ผ่านทางเว็บบราวเซอร์ (Web Browser) สามารถทำงานได้

5.2.2 ส่วนควบคุมการเปลี่ยนมุมมองของกล้องเว็บแคมที่เป็นส่วนของฮาร์ดแวร์ โดยใช้สเต็ปปิ้งมอเตอร์ (Stepping Motor) ในการควบคุมสามารถเปลี่ยนมุมมองได้สองมุมมอง คือ ซ้าย-ขวา และขึ้นลง โดยที่ ซ้าย-ขวา สามารถหมุนได้ 180 องศา ส่วนการหมุน ขึ้น-ลง สามารถหมุนได้ 90 องศา และส่วนการเปิด-ปิดอุปกรณ์ไฟฟ้าได้ใช้ Buzzer ในการแสดงการเปิด-ปิด

5.2.3 โปรแกรมส่วนเครื่องแม่ข่าย (Server) ที่ทำหน้าที่เป็นตัวกลางในการเชื่อมต่อชุดควบคุมฮาร์ดแวร์กับส่วนการสั่งงานจากเครื่องลูกข่ายสามารถติดต่อกันและส่งชุดคำสั่งในการควบคุมการเปลี่ยนมุมมองของกล้องเว็บแคม และเปิด-ปิดอุปกรณ์ไฟฟ้าได้

5.2.4 โปรแกรมส่วนเครื่องลูกข่าย (Client) ที่สร้างเป็น GUI (Graphic User Interface) ที่ให้ผู้ใช้ป้อนคำสั่งในการสั่งงานอุปกรณ์สามารถใช้งานได้ง่าย และส่งคำสั่งที่ใช้เป็นคำสั่งในการควบคุมอุปกรณ์ไฟฟ้าไปที่เครื่องแม่ข่าย (Server) ได้

5.3 ข้อเสนอแนะสำหรับการพัฒนาในอนาคต

5.3.1 เพิ่มความสามารถในการหมุนกล้องให้สามารถควบคุมการหมุนในระยะ ซ้าย-ขวา ได้ 0-360 องศา เพื่อเพิ่มมุมมองในการมองภาพ

5.3.2 สร้างส่วนการหมุนของกล้องให้มีความนุ่มนวลเพิ่มมากขึ้น โดยการเปลี่ยนสเต็ปปิ้งมอเตอร์ (Stepping Motor) หรือเปลี่ยนอัตราการทดเฟืองใหม่

5.3.3 ในส่วนของการรับชมภาพควรเพิ่มให้สามารถรับฟังเสียงได้ด้วย

5.3.4 เพิ่มส่วนที่สามารถบันทึกภาพและเสียง ให้เป็นไฟล์วีดีโอ ทางฝั่งเครื่องลูกข่าย (Client) ได้

5.3.5 เพิ่มการเชื่อมต่อกับอุปกรณ์อื่นๆ เพื่อควบคุมสั่งงานจากระยะไกลได้อีกตามความเหมาะสม

เอกสารอ้างอิง

[1] Shivaram H. Mysore and Rinaldo Di Giorgio. "Java gets serial support with the new javax.comm package." [Online]. Available: http://www.javaworld.com/javaworld/jw-05-1998/jw-05-javadev.html. 1998.

[2] ดร.วีระศักดิ์ ซึงถาวร. **JAVA PROGRAMMING Volume I.** กรุงเทพฯ : ซีเอ็ดยูเคชั่น. 2546.

[3] ดร.วีระศักดิ์ ซึงถาวร. **JAVA PROGRAMMING Volume III.** กรุงเทพฯ : ซีเอ็ดยูเคชั่น. 2547.

[4] Elliotte Rusty Harold. **Java Network Programming.** Second Edition. USA : O'Reilly & Associate, Inc. 2000.

[5] Budi Kurniawan. "Program multimedia with JMF, Part 1." [Online]. Available: http://www.javaworld.com/jw-04-2001/jw-0406-jmf1.html. 2001.

[6] จตุชัย แพงจันทร์. เจาะระบบ Network ฉบับสมบูรณ์. นนทบุรี : ไอดีซี. 2546.

[7] ผศ.ธีระวัฒน์ ประกอบผล. การประยุกต์ใช้งานไมโครคอนโทรลเลอร์. ฉบับปรับปรุง. กรุงเทพฯ : ส.ส.ท. . 2546.

[8] Atmel Corporation. "8-bit Microcontroller with 8K Bytes Flash AT89S8252." [Online]. Available: www.atmel.com/atmel/acrobat/doc0401.pdf.

[9] Texas Instruments Incorporated. "MAX232 Datasheet." [Online]. Available: http://focus.ti.com/lit/ds/symlink/max232.pdf.

[10] Thaiio. "ความรู้เบื้องต้นและ หลักการทำงาน Step Motor." [Online]. Available: http://www.thaiio.com/Hardware-cgi/hardware.cgi?0008.

[11] SGS-THOMSON Microclectronics. "Datasheet ULN2803." [Online]. Available: http://www.ucapps.de/midio128/uln2803.pdf.

[12] Logitech Inc. "QuickCam® Express." [Online]. Available: http://www.logitech.com/index.cfm/products/details/AU/EN,CRID=2204,CONTENTID=5037.

[13] ดร.วีระศักดิ์ ซึงถาวร. **JAVA PROGRAMMING Volume II.** กรุงเทพฯ : ซีเอ็ดยูเคชั่น. 2547.

[14] National Semiconductor Corporation. "LM78M05 Product Folder." [Online]. Available: http://www.national.com/pf/LM/LM78M05.html.

[15] National Semiconductor Corporation. "LM78M12 Product Folder." [Online]. Available: http://www.national.com/pf/LM/LM78M12.html.

[16] Philips Electronics. "Product description BC546; BC547; NPN general purpose Transistors." [Online]. Available: http://www.semiconductors.philips.com/pip/BC547.html.

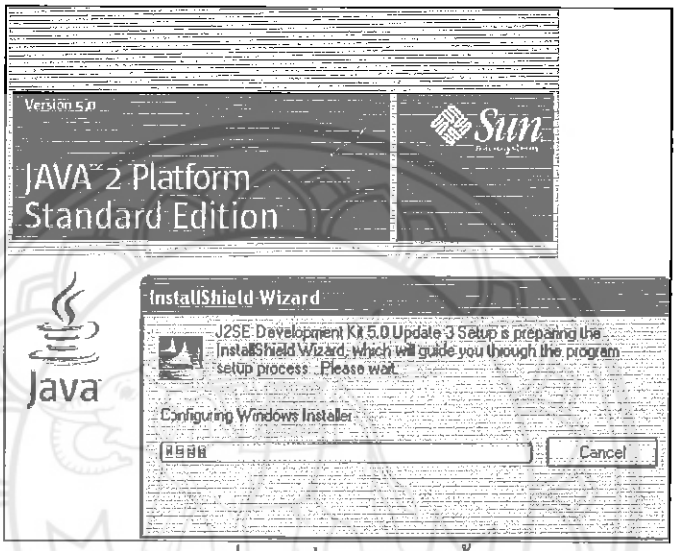
ภาคผนวก

ภาคผนวก ก

การติดตั้งตัวแปลภาษาจาวา J2SE 5.0

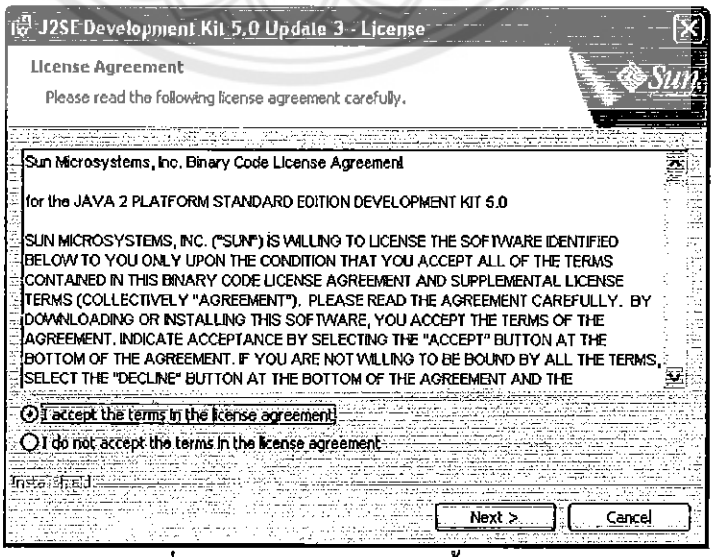
1. การติดตั้งตัวแปลภาษาจาวา J2SE 5.0

1.1 ให้ดับเบิลคลิกที่ไฟล์ jdk-1_5_0_03-windows-i586-p.exe ซึ่งเป็น JDK ที่ดาวน์โหลดมาจะมีหน้าจอปรากฏขึ้นมาดังรูป



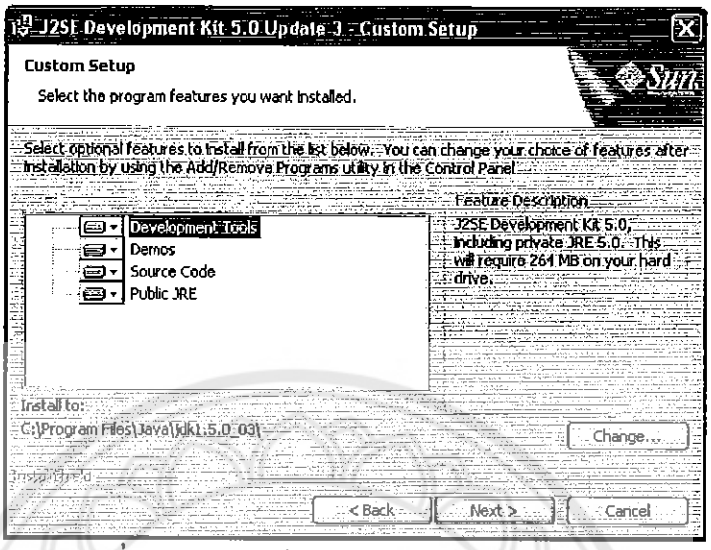
รูปที่ 1.1 เริ่มต้นการติดตั้ง

1.2 จะเข้าสู่หน้าจอซึ่งเป็นขอตกลงในการติดตั้งโปรแกรม ให้เลือก "I accept the terms in the license agreement" แล้วคลิก Next



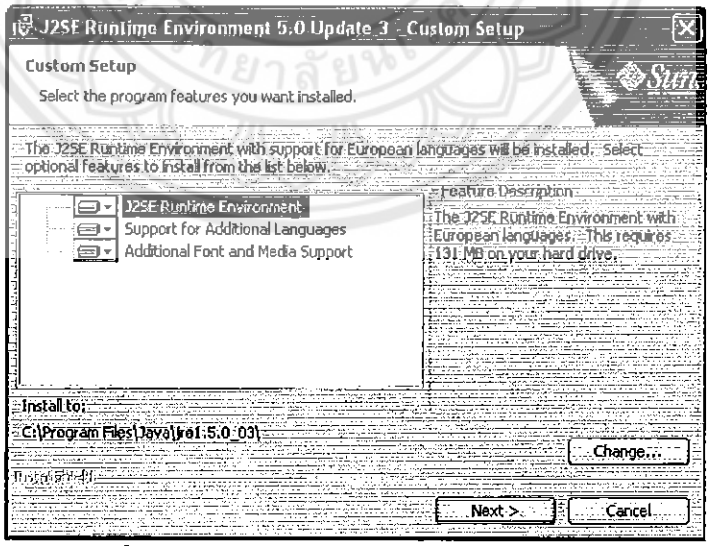
รูปที่ 1.2 ข้อตกลงในการติดตั้งโปรแกรม

1.3 จะมีองค์ประกอบที่ให้เลือกติดตั้งได้ทั้งหมด 4 ตัวด้วยกัน จากนั้นเลือกไดเรกทรอรีที่ต้องการทำการติดตั้ง ในที่นี้ให้เลือกลงที่ C:\Java จากนั้นคลิก Next



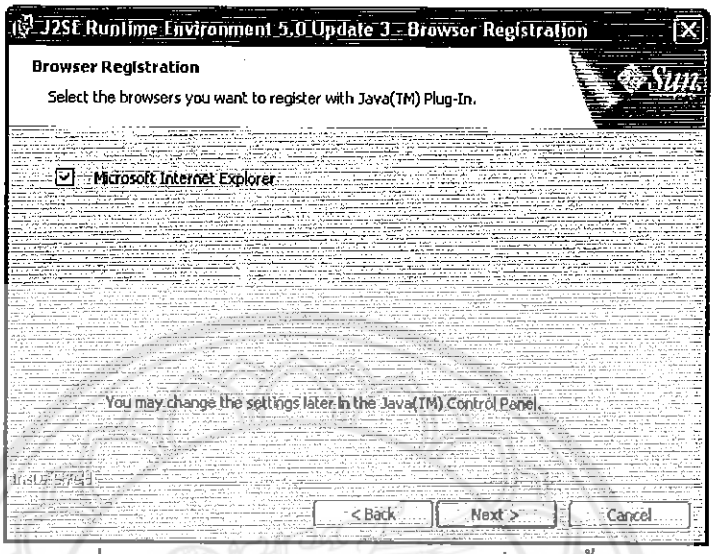
รูปที่ 1.3 เลือกองค์ประกอบและไดเรกทรอรี JDK

1.4 JDK จะทำการติดตั้งลงในเครื่อง ระหว่างนั้นจะมีวินโดว์ J2SE Runtime Environment 5.0 Update 3 แสดงขึ้นมา ให้เปลี่ยนไดเรกทรอรีไปที่ C:\Java จากนั้นคลิก Next เพื่อทำการติดตั้ง JRE ลงในเครื่อง



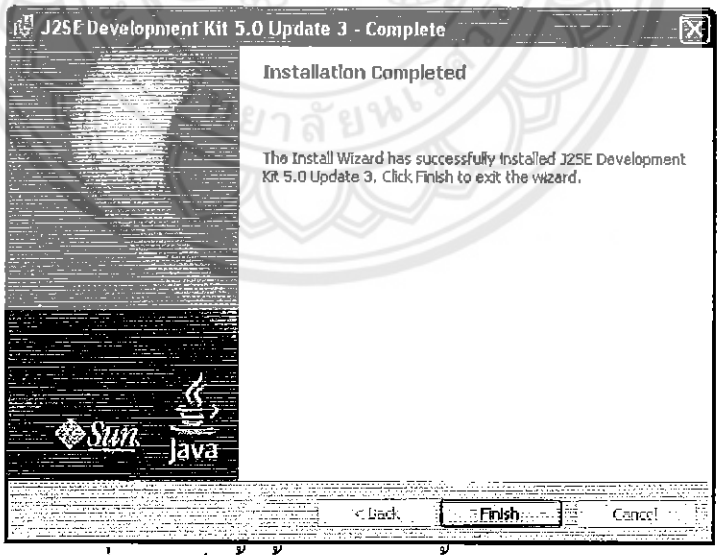
รูปที่ 1.4 เลือกองค์ประกอบและไดเรกทรอรี JRE

1.5 ทำการเลือกโปรแกรมเว็บบราวเซอร์ที่จะให้ติดตั้งตัว Java Plug-in เข้าไป เพื่อที่โปรแกรมเว็บบราวเซอร์นั้นจะสามารถเรียกรันแอพเพล็ตโดยใช้ JRE ของ J2SE 5.0 Update 3 ได้ เสร็จแล้วคลิก Next



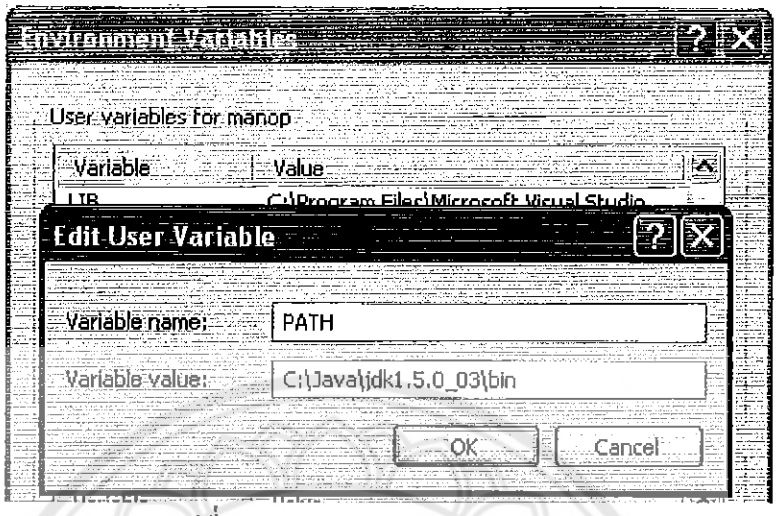
รูปที่ 1.5 เลือกโปรแกรมเว็บบราวเซอร์ที่จะติดตั้ง JRE

1.6 เมื่อ JRE ติดตั้งเสร็จแล้ว ให้คลิก Finish ก็เสร็จสิ้นขั้นตอนการติดตั้งตัวแปลภาษาจาวา



รูปที่ 1.6 เสร็จสิ้นขั้นตอนการติดตั้งตัวแปลภาษาจาวา

1.7 ทำการเซ็ต environment เพื่อความสะดวกในการเรียกใช้ตัวแปลภาษาจาวา โดยการกำหนดค่าของตัวแปร path เป็น C:\Java\Jdk1.5.0_03\bin



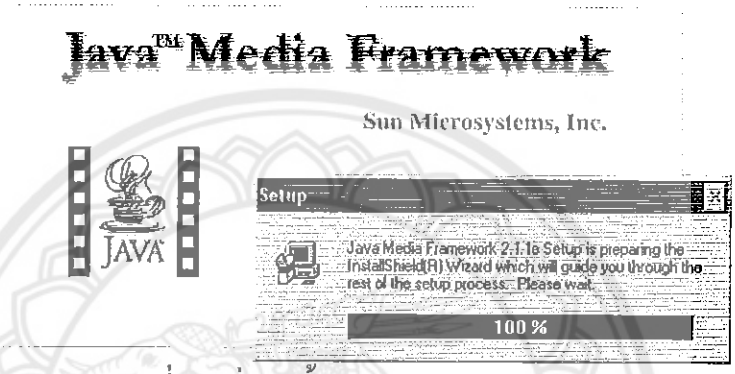
รูปที่ 1.7 การกำหนดค่าของตัวแปร path

ภาคผนวก ข

การติดตั้ง Java Media Framework

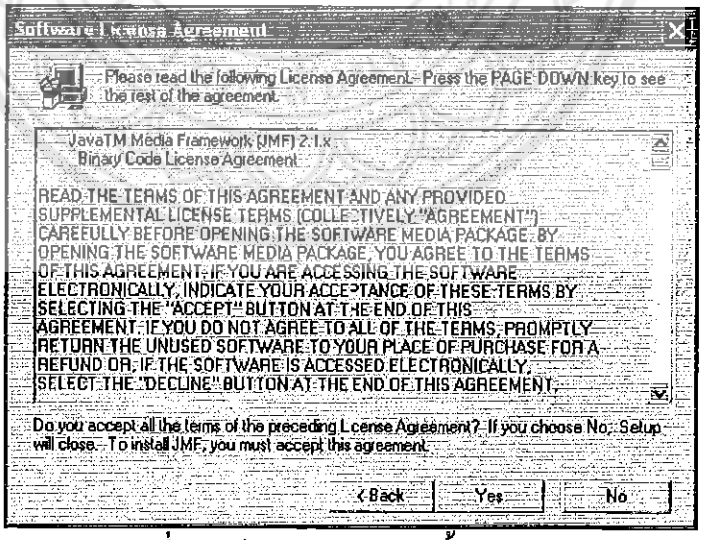
1. การติดตั้ง Java Media Framework

1.1 ให้ดับเบิลคลิกที่ไฟล์ jmf-2_1_1e-windows-i586.exe เพื่อเป็นการเริ่มติดตั้ง Java Media Framework



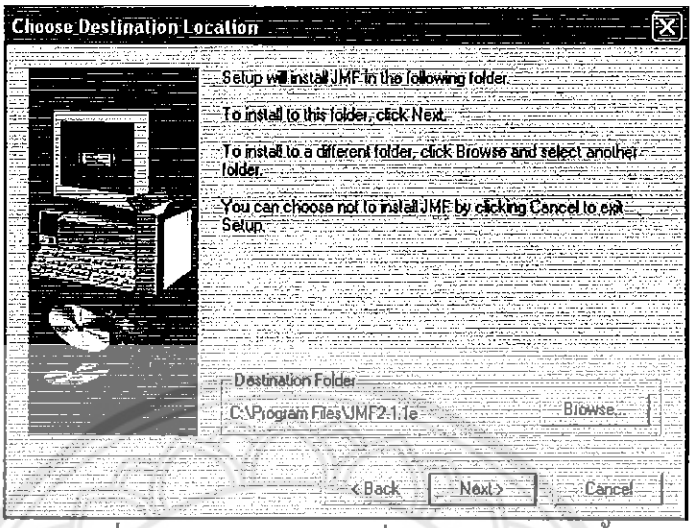
รูปที่ 1.1 เริ่มติดตั้ง Java Media Framework

1.2 จะเข้าสู่หน้าจอซึ่งเป็นขอตกลงในการติดตั้งโปรแกรม ให้เลือก Yes



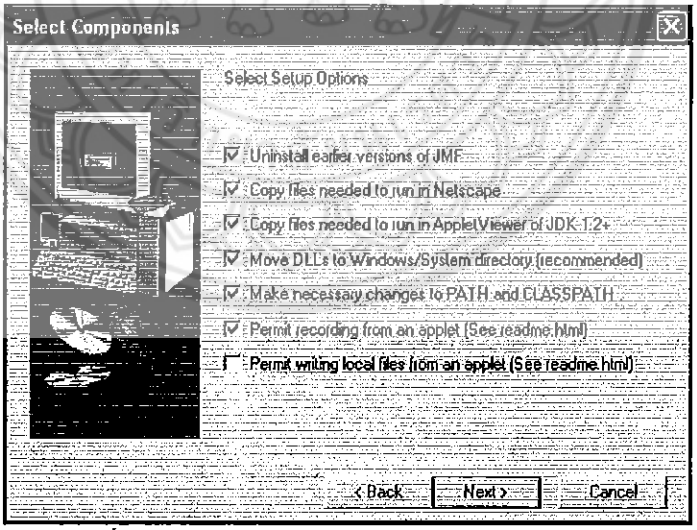
รูปที่ 1.2 ข้อตกลงในการติดตั้งโปรแกรม

1.3 เลือกไดเรกทรอรีที่ต้องการทำการติดตั้ง ในที่นี้ให้เลือกลงตามที่กำหนดมา จากนั้นคลิก Next



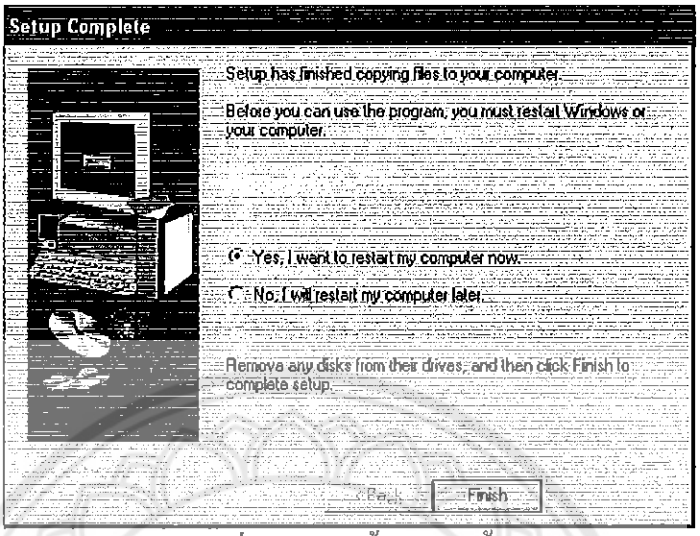
รูปที่ 1.3 เลือกไดเรกทรอรีที่ต้องการทำการติดตั้ง

1.4 เลือกออบชั่นตามรูปที่ 1.4 เพื่อเลือกคอมโพเนนท์ที่จำเป็นในการใช้งาน จากนั้นคลิก Next



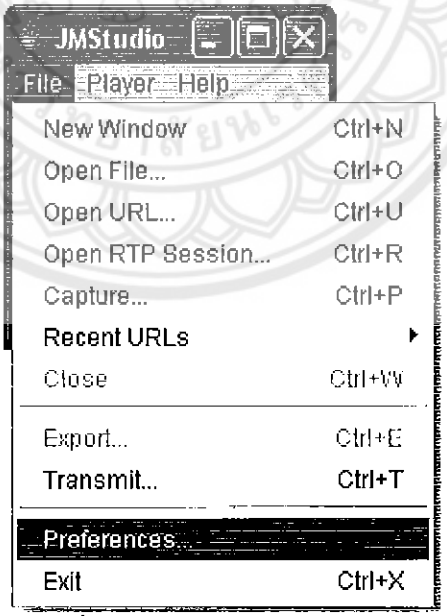
รูปที่ 1.4 เลือกคอมโพเนนท์ที่จำเป็นในการใช้งาน

1.5 เมื่อขั้นตอนการติดตั้งคอมโพเนนท์ต่างๆ เสร็จสมบูรณ์ ก่อนที่จะใช้งานโปรแกรมได้ ต้องมีการรีสตาร์ทเครื่องใหม่ ให้เลือกดังรูปที่ 1.5 จากนั้นคลิก Finish เครื่องก็จะทำการรีสตาร์ทเครื่องใหม่



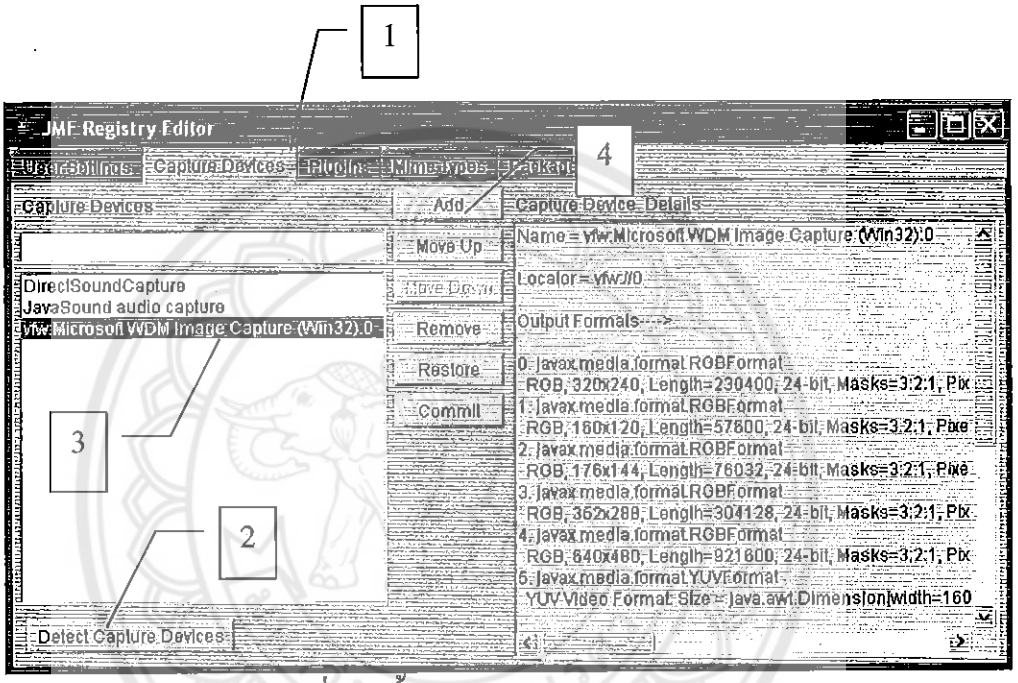
รูปที่ 1.5 เสร็จสิ้นการติดตั้ง

1.6 เมื่อเครื่องทำการรีสตาร์ทเสร็จแล้ว ให้รันโปรแกรม JMStudio ที่ได้ติดตั้งลงไปเพื่อทำการ Registry โปรแกรม เมื่อรันโปรแกรมขึ้นมาแล้วให้ไปที่เมนู File->Preferences ดังรูปที่ 1.6



รูปที่ 1.6 การ Registry โปรแกรม

เมื่อคลิกที่ Preferences แล้วให้เลือกที่ Capture Devices [1] จากนั้นก่อนทำขั้นตอนต่อไป ต้องทำการต่อกล้องเว็บแคมเข้ากับพอร์ตยูเอสบีพร้อมกับลงไดร์เวอร์เสร็จสมบูรณ์แล้ว เมื่อดำเนินการเสร็จเรียบร้อยแล้วก็คลิกที่ Detect Capture Device [2] เพื่อทำการค้นหาอุปกรณ์รับภาพ-เสียงต่างๆ เมื่อพบแล้วก็จะแสดงชื่อของอุปกรณ์นั้นตามรูปแบบการเรียกของ Java Media Framework ซึ่งชื่อของ "Logitech QuickCam Express" ที่ใช้ในโครงงานก็แสดงใน [3] จากนั้นก็คลิกที่ Add [4] แล้วก็ปิดโปรแกรม ก็เสร็จสิ้นขั้นตอนการติดตั้ง Java Media Framework

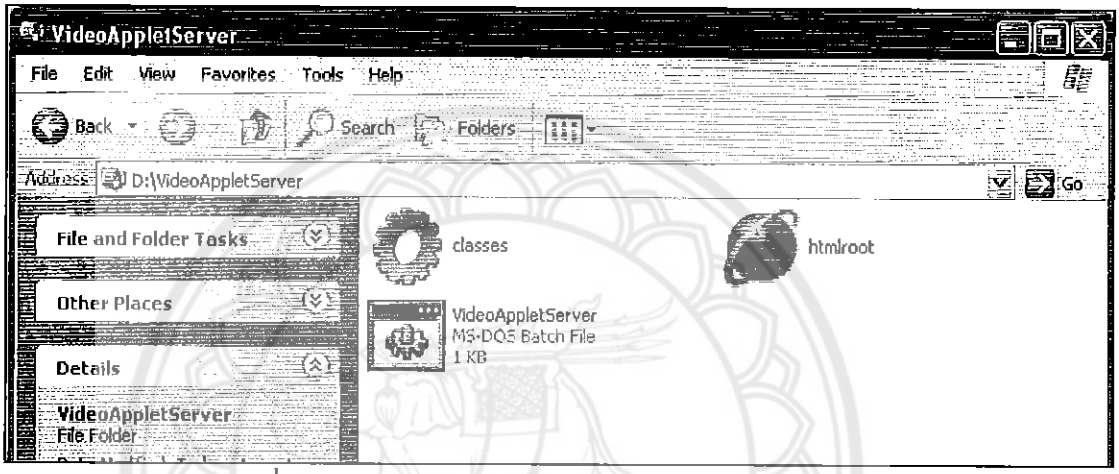


รูปที่ 1.7 ขั้นตอนการ Registry โปรแกรม

ภาคผนวก ค
การติดตั้งใช้งานโปรแกรม

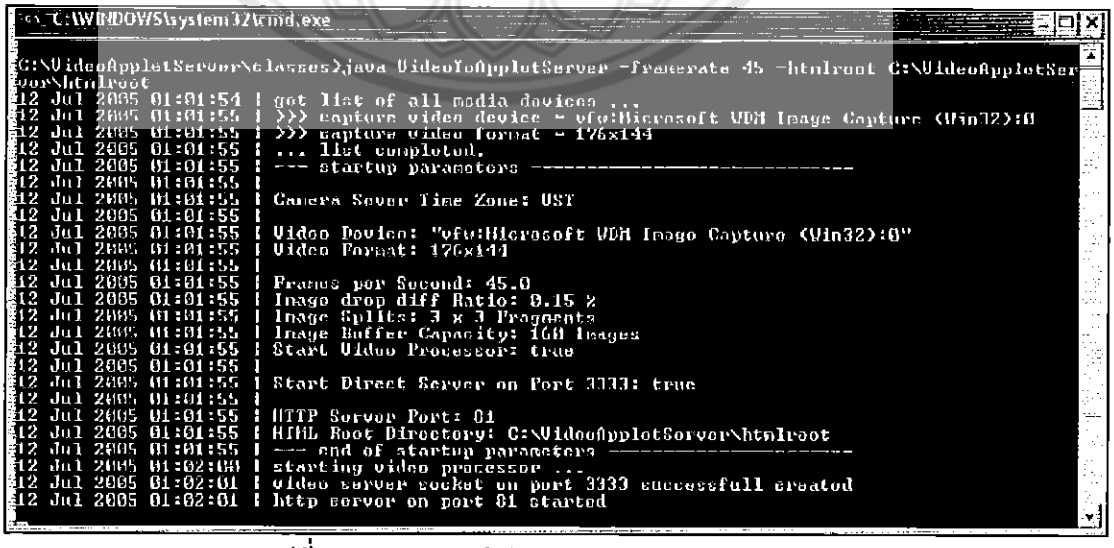
1. การติดตั้งโปรแกรมส่วนเว็บเซิร์ฟเวอร์เพื่อให้บริการภาพวีดีโอแบบเรียลไทม์

1.1 คัดลอกโฟลเดอร์ VideoAppletServer ซึ่งมีโฟลเดอร์ classes , htmlroot และ VideoAppletServer.bat อยู่ไปไว้ที่ไดร์ฟซี



รูปที่ 1.1 ไดเรกทรอรีเก็บโปรแกรม VideoToAppletServer

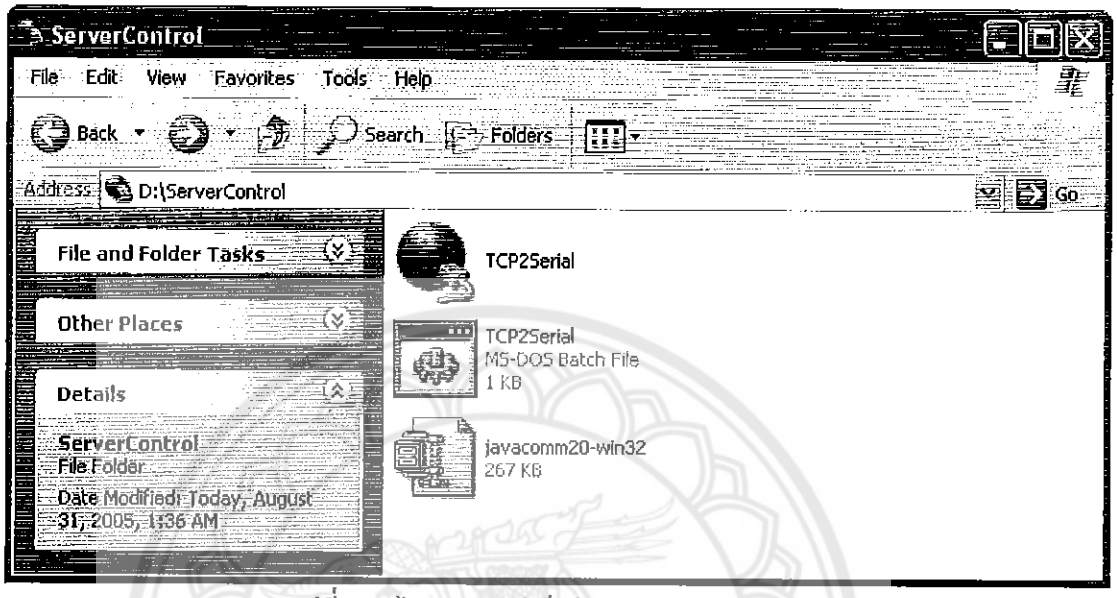
1.2 เมื่อทำการคัดลอกเสร็จแล้ว ดับเบิลคลิกที่ VideoAppletServer.bat เพื่อทำการรันโปรแกรม VideoToAppletServer



รูปที่ 1.2 ผลการรันโปรแกรม VideoToAppletServer

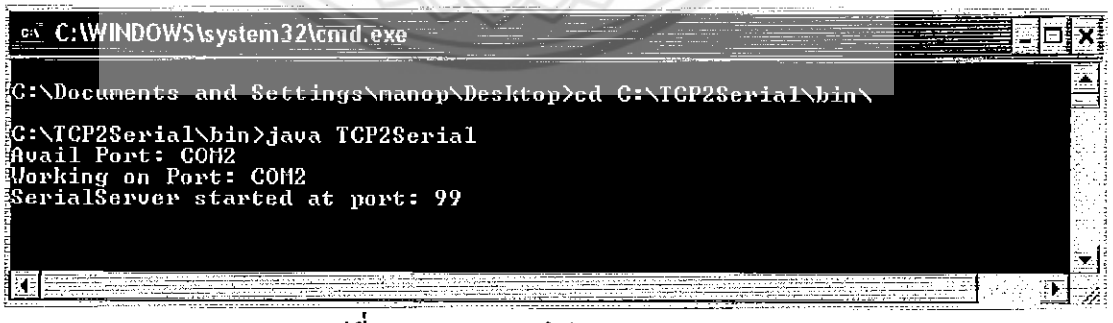
2. การติดตั้งโปรแกรมส่วนควบคุมการเปลี่ยนมุมของกล้องและเปิด-ปิดอุปกรณ์ไฟฟ้า

2.1 คัดลอกโฟลเดอร์ ServerControl ซึ่งมีโฟลเดอร์ TCP2Serial, javacomm20-win32.zib และ TCP2Serial.bat อยู่ไปไว้ที่ไดร์ฟซี



รูปที่ 2.1 ไดเรกทรอรีเก็บโปรแกรม TCP2Serail

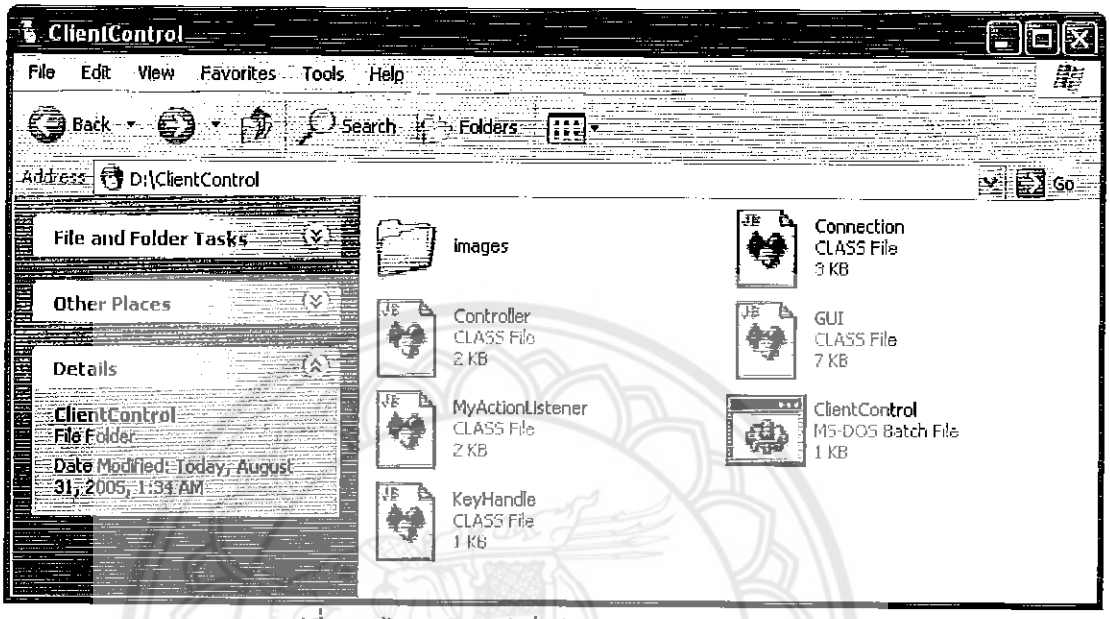
2.2 จากนั้นทำการขยายไฟล์ javacomm20-win32.zib ไปไว้ที่ C:\Java หลังจากนั้นทำการคัดลอก comm.jar และ javax.comm.properties ไปไว้ใน C:\Java\jdk1.5.0_03\lib อีกส่วนหนึ่งคือคัดลอก win32com.dll ไปไว้ใน C:\Java\jdk1.5.0_03\lib จากนั้นก็ทำการรันโปรแกรมโดยดับเบิลคลิกที่ TCP2Serial.bat



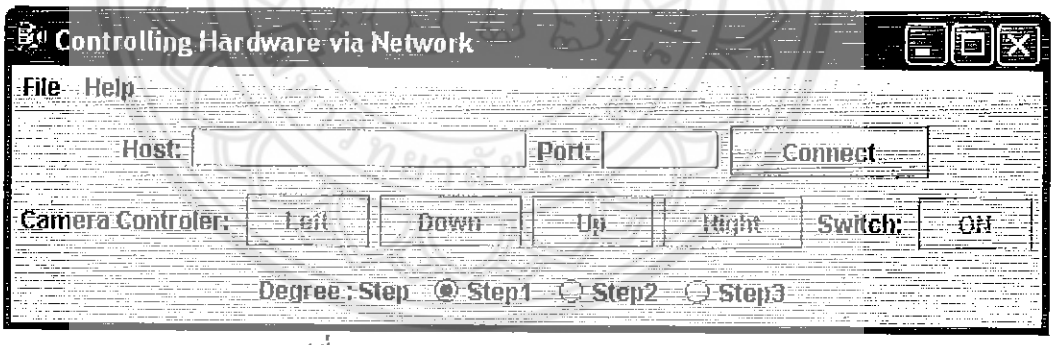
รูปที่ 2.2 ผลการรันโปรแกรม TCP2Serial

3. การติดตั้งโปรแกรมส่วนเครื่องลูกข่าย

3.1 ทำการคัดลอกโฟลเดอร์ ClientControl ไปไว้ที่ไดร์ฟซี จากนั้นดับเบิลคลิกที่ ClientControl.bat เพื่อการรันโปรแกรม



รูปที่ 3.1 ไดเรกทรอรีเก็บโปรแกรม ClientControl



รูปที่ 3.2 ผลการรันโปรแกรม ClientControl

ประวัติผู้เขียนโครงงาน

ชื่อ นายมานพ นาคมี

ภูมิลำเนา 82/1 ม.2 ต.บ้านซ่าน อ.ศรีสำโรง จ.สุโขทัย

ประวัติการศึกษา

- จบการศึกษาระดับมัธยมศึกษาจากโรงเรียนบ้านไร่พิทยาคม
- ปัจจุบันกำลังศึกษาระดับปริญญาตรีชั้นปีที่ 4
  สาขาวิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยนเรศวร

E-mail: cpenu_ix@hotmail.com, manop.com@gmail.com