การออกแบบระบบ HMI ผ่านระบบโครงข่ายเครื่องควบคุมแบบโปรแกรมได้ด้วยภาษา JAVA

DESIGN OF HMI FOR PLC NETWORK USING JAVA

นายกิตติศักดิ์ พลันสังเกตุ
นายฉัตรชัย เรืองปรีชา
นายสุวัตร บัวเพชร

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมการวัดคุม
ภาควิชาวิศวกรรมการวัดคุม คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2548

DESIGN OF HMI FOR PLC NETWORK USING JAVA

KITTISAK PLANSANGKATE

CHATCHAI RUANGPREECHA

SUWAT BUAPATCH

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

BACHELOR OF ENGINEERING IN INSTRUMENTATION ENGINEERING

DEPARTMENT OF INSTRUMENTATION ENGINEERING

FACULTY OF ENGINEERING

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2005

ภาควิชาวิศวกรรมการวัดคุม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ใบรับรองปริญญานิพนธ์

หัวข้อปริญญานิพนธ์ การออกแบบระบบ HMI ผ่านระบบโครงข่ายเครื่องควบคุมแบบโปรแกรมได้ด้วยภาษา JAVA

DESIGN OF HMI FOR PLC NETWORK USING JAVA

นักศึกษาผู้จัดทำ	นายกิตติศักดิ์	พลันสังเกต	**รหัสประจำตัว**	46015433
	นายฉัตรชัย	เรืองปรีชา	**รหัสประจำตัว**	46015436
	นายสุวัตร	บัวเพชร	**รหัสประจำตัว**	46015474
ปริญญา	วิศวกรรมศาสตรบัณฑิต			
สาขาวิชา	วิศวกรรมการวัดคุม			
ปีการศึกษา	2548			

อาจารย์ผู้ควบคุมปริญญานิพนธ์	**ลายมือชื่อ**
รศ.สุพรรณ กุลพาณิชย์	

ภาควิชารับรองแล้ว

(รศ.ประสิทธิ์ จุลเสรีวงศ์)

หัวหน้าภาควิชาวิศวกรรมการวัดคุม

หัวข้อปริญญานิพนธ์	การออกแบบระบบ HMI ผ่านระบบโครงข่ายเครื่องควบคุมแบบโปรแกรมได้ด้วยภาษา JAVA DESIGN OF HMI FOR PLC NETWORK USING JAVA			
นักศึกษาผู้จัดทำ	นายกิตติศักดิ์	พลันสังเกตุ	รหัสประจำตัว	46015433
	นายฉัตรชัย	เรืองปรีชา	รหัสประจำตัว	46015436
	นายสุวัตร	บัวเพชร	รหัสประจำตัว	46015474
อาจารย์ที่ปรึกษา	รศ.สุพรรณ	กุลพาณิชย์		
ปีการศึกษา	2548			

บทคัดย่อ

ปริญญานิพนธ์ฉบับนี้มีวัตถุประสงค์ที่จะนำเสนอการออกแบบ HMI (Human Machine Interface) เพื่อใช้เป็นสื่อกลางในการเชื่อมโยงการทำงาน ระหว่างกระบวนการที่ถูกควบคุมโดยโครงข่ายของเครื่องควบคุมแบบโปรแกรมได้กับผู้ใช้งาน โดยแบ่งออกเป็น HMI สำหรับเครื่องคอมพิวเตอร์โดยใช้โปรแกรมภาษาจาวา และการออกแบบ HMI สำหรับเครื่องควบคุมแบบหน้าจอสัมผัส (Touch Screen) โดยใช้ NT Support Software ลักษณะของ HMI ที่ถูกออกแบบขึ้นจะเป็นโปรแกรมสำเร็จรูปที่ใช้สำหรับควบคุมกระบวนการทางอุตสาหกรรมที่ถูกควบคุมโดยโครงข่ายของเครื่องควบคุมแบบโปรแกรมได้ ซึ่งสามารถทำการเปลี่ยนแปลงสภาวะของกระบวนการตามการควบคุมของเครื่องควบคุมแบบโปรแกรมได้ นอกจากนั้นผู้ใช้งานยังสามารถที่จะปรับเปลี่ยนการทำงานของเครื่องควบคุมแบบโปรแกรมได้ผ่าน HMI นี้ได้ด้วย

Thesis Title	Design of HMI for PLC network using JAVA
Authors	Mr. Kittisak Plansangkate
	Mr. Chatchai Ruangpreecha
	Mr. Suwat Buapatch
Thesis Advisor	Assoc. Prof. Suphan Koonphanich
Year	2005

ABSTRACT

The Objective of this thesis is presentation of designing HMI (Human Machine Interface) used are middleware for operation interface between process controlled by PLC with user. The HMI to divided are designing HMI on computer by using JAVA programming and designing HMI on the touch screen by using NT Support Software. This HMI be able to control the industrial process controlled by PLC network. It be able to change status the industrial process controlled by PLC network. And user can control PLC through this HMI.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงได้ด้วยดี เพราะได้รับความเมตตาจาก รองศาสตราจารย์ สุพรรณ กุลพาณิชย์ และผู้ช่วยศาสตราจารย์ ทวีพล ซื่อสัตย์ ที่ได้ให้คำแนะนำกับผู้วิจัยตลอดมา อีกทั้งยังเอื้อเฟื้ออุปกรณ์และเครื่องมือต่าง ๆ ในการทำปริญญานิพนธ์นี้ ผู้วิจัยรู้สึกซาบซึ้งและขอกราบขอบพระคุณเป็นอย่างสูง หลายครั้งที่ประสบปัญหาต่างๆ มากมายเกี่ยวกับการเขียนโปรแกรม การทำงานกับเครื่องควบคุมแบบโปรแกรมได้ แต่ก็รับความช่วยเหลือจากอาจารย์ที่ได้คอยแนะนำแนวทางในการแก้ไขปัญหาและพัฒนาโครงงานโดยตลอด

ขอขอบพระคุณ อาจารย์ภาควิชาวิศวกรรมการวัดคุมทุกท่าน ที่ให้คำแนะนำอันเป็นประโยชน์ต่อการทำปริญญานิพนธ์ฉบับนี้

และที่ลืมเสียมิได้คือ ขอกราบขอบพระคุณ คุณพ่อ คุณแม่ อันเป็นที่รักยิ่ง ที่สนับสนุนและเป็นแรงบันดาลใจในการทำปริญญานิพนธ์ฉบับนี้

และสุดท้ายขอขอบพระคุณผู้ที่เกี่ยวข้องที่คอยให้ความช่วยเหลือต่างๆ จนทำให้โครงงานปริญญานิพนธ์เล่มนี้สำเร็จลุล่วงไปได้

คณะผู้จัดทำ

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญภาพ

ภาพที่ **หน้า**

สารบัญภาพ(ต่อ)

ภาพที่ หน้า

สารบัญตาราง

ตารางที่ **หน้า**

บทที่ 1
บทนำ

1.1 ความเป็นมาและเหตุจูงใจของการวิจัย

ปัจจุบันเทคโนโลยีทางด้านอุตสาหกรรมมีการพัฒนา และเปลี่ยนแปลงไปอย่างรวดเร็ว ดังนั้นระบบควบคุมที่ดีมีประสิทธิภาพ โดยเฉพาะอย่างยิ่งระบบควบคุมในโรงงานอัตโนมัติจึงนับว่าเป็นสิ่งที่จำเป็น เครื่องควบคุมแบบโปรแกรมได้เป็นเครื่องควบคุมที่ถูกนำมาประยุกต์ใช้งานในระบบโรงงานอัตโนมัติ ทั้งนี้เนื่องจากเครื่องควบคุมแต่ละชุดจะควบคุมเครื่องจักรกล หรือกระบวนการที่ถูกจัดวางในที่ต่างๆกัน จึงไม่สะดวกที่จะนำคอมพิวเตอร์มาทำการตรวจสอบสภาวะ หรือส่งค่าตัวแปรให้กับเครื่องควบคุมชุดต่อชุดเพราะจะทำให้ต้องสิ้นเปลืองค่าใช้จ่ายเป็นอย่างมาก และได้ประสิทธิภาพการทำงานที่ต่ำ แต่ถ้านำคอมพิวเตอร์เพียงเครื่องเดียวมาทำการควบคุมแล้วสามารถควบคุมการทำงาน โดยใช้ส่วนของการเชื่อมต่อกับโครงข่ายของเครื่องควบคุมจำนวนหลายๆ เครื่อง ก็จะอำนวยความสะดวกได้เป็นอย่างมาก

จึงเป็นเหตุผลให้มีการพัฒนาระบบ HMI ขึ้นมา เพื่อที่จะสามารถสั่งงานควบคุมเครื่องจักร หรือกระบวนการที่ถูกควบคุมโดยเครื่องควบคุมแบบโปรแกรมได้ ด้วยการนำข้อดีของการโปรแกรมด้วยภาษาจาวามาทำการประยุกต์ใช้ในการออกแบบระบบ HMI สำหรับโครงข่ายเครื่องควบคุมแบบโปรแกรมได้โดยใช้คอมพิวเตอร์เป็นตัวโฮสในควบคุมการทำงานผ่านโครงข่ายเครื่องควบคุม

1.2 วัตถุประสงค์ของปริญญานิพนธ์

1. เพื่อศึกษาการทำงานของเครื่องควบคุมแบบโปรแกรมได้ (PLC) ทั้งทางด้านทฤษฎี และ การประยุกต์ใช้งาน

2. เพื่อสร้างโปรแกรมที่สามารถจำลองกระบวนการ เป็นภาพกราฟิกบนหน้าจอคอมพิวเตอร์ (HMI) ซึ่งสามารถแสดงผลการเปลี่ยนแปลงที่เกิดจากการควบคุมของ PLC และสามารถสั่งงานเครื่องPLC ผ่านทางหน้าจอคอมพิวเตอร์ได้

1.3 ขอบเขตของปริญญานิพนธ์

1. เป็นโปรแกรมควบคุมกระบวนการบนหน้าจอของคอมพิวเตอร์ ที่สามารถใช้งานได้บนระบบปฏิบัติการวินโดวส์

2. เป็นโปรแกรมที่สามารถแสดงผลและควบคุมการทำงานจากคอมพิวเตอร์ได้

1.4 ขั้นตอนการศึกษา

1. ศึกษาโครงสร้าง และการใช้งานเครื่องควบคุมแบบโปรแกรมได้ (PLC) ในการควบคุมกระบวนการ
2. ศึกษาวิธีการติดต่อสื่อสารระหว่าง PLC กับคอมพิวเตอร์ (Host Link Communication)
3. ศึกษาโปรแกรมภาษาจาวา (Java programming)
4. ออกแบบและพัฒนากราฟิกที่ใช้ในการแสดงผลกระบวนการ และทำการเชื่อมต่อโปรแกรมที่ออกแบบกับเครื่อง PLC
5. ทดสอบโปรแกรมที่ได้ออกแบบกับกระบวนการ
6. วิเคราะห์ และสรุปผล และข้อเสนอแนะเพื่อเป็นแนวทางในการพัฒนาต่อไป

บทที่ 2
โครงสร้างและการใช้งานเครื่องควบคุมแบบโปรแกรมได้ (PLC)

2.1 กล่าวนำ

เมื่อปี พ.ศ. 2511 ในฝ่าย Hydromatic ของบริษัท General Motors ประเทศสหรัฐอเมริกา ได้คิดค้นอุปกรณ์ควบคุมแบบใหม่ เพื่อใช้ทดแทนวงจรไฟฟ้าแบบเดิมที่ใช้กันอยู่ในโรงงานอุตสาหกรรมของบริษัท และในปี พ.ศ.2512 PLC ได้ถูกผลิตขึ้นจำหน่ายในประเทศสหรัฐอเมริกาเป็นแห่งแรก ส่วนในประเทศญี่ปุ่น PLC ได้ถูกพัฒนาขึ้นมาภายหลังจากที่บริษัท ออมรอน (OMRON) ประเทศญี่ปุ่นได้สำเร็จในการผลิตโซลิด - สเตทรีเลย์ (Solid State Relay)ในปี พ.ศ.2508 หลังจากนั้น 5 ปี PLC ได้ถูกนำออกจำหน่ายสู่ท้องตลาดจนเป็นที่แพร่หลายในเวลาต่อมา ส่วน PLC นั้นมีชื่อเรียกแตกต่างกันแต่ละประเทศ เช่น PC (Programmable Controller)- อังกฤษ, PLC (Programmable Logic Controller) - USA, PBS(Programmable Binary System)- สแกนดิเนเวีย

PLC นั้นเป็นอุปกรณ์ชนิดโซลิด-สเตท (Solid State) ที่ทำงานแบบลอจิก (Logic Function) การออกแบบการทำงานของ PLC จะคล้ายกับหลักการทำงานของคอมพิวเตอร์ จากหลักการพื้นฐานแล้ว PLC จะประกอบด้วยอุปกรณ์ที่เรียกว่า Solid-State Digital Logic Elements เพื่อให้ได้งานและตัดสินใจแบบลอจิก PLC ใช้สำหรับควบคุมกระบวนการทำงานของเครื่องจักรและอุปกรณ์ในโรงงานอุตสาหกรรม ซึ่งการใช้ PLC สำหรับควบคุมเครื่องจักรหรืออุปกรณ์ต่างๆ ในโรงงานอุตสาหกรรม จะมีข้อได้เปรียบกว่าการใช้ระบบรีเลย์ (Relay) ซึ่งจำเป็นจะต้องเดินสายไฟฟ้า หรือเรียกว่า Hard-Wired ดังนั้นเมื่อมีความจำเป็นที่จำเป็นจะต้องเปลี่ยนกระบวนการผลิต หรือลำดับการทำงานใหม่ก็ต้องเดินสายไฟฟ้าใหม่ ซึ่งเสียเวลาและเสียค่าใช้จ่ายสูงแต่เมื่อเปลี่ยนมาใช้ PLC แล้ว การเปลี่ยนกระบวนการผลิตหรือลำดับการทำงานใหม่นั้นทำได้โดยการเปลี่ยนโปรแกรมใหม่เท่านั้น นอกจากนี้แล้ว PLC ยังใช้ระบบโซลิด-สเตท ซึ่งจะมีความน่าเชื่อถือมากกว่าระบบเดิม กินกระแสไฟฟ้าน้อยกว่า และสะดวก เมื่อต้องการขยายขั้นตอนการทำงานของเครื่องจักร

2.2 โครงสร้างและส่วนประกอบของ PLC

PLC เป็นอุปกรณ์ควบคุมชนิดหนึ่ง ที่นำมาแทนที่การควบคุมที่ใช้รีเลย์ ทำให้สะดวกขึ้น PLC เป็นอุปกรณ์ชนิด Solid State ที่ทำงานแบบลอจิก (Logic Function) การออกแบบการทำงานของ PLC จะคล้ายกับหลักการทำงานของคอมพิวเตอร์ ซึ่งจากหลักการพื้นฐานแล้ว PLC จะประกอบด้วยอุปกรณ์ที่เรียกว่า Solid-State Digital Elements เพื่อให้ทำงานและตัดสินใจแบบลอจิก PLC ใช้สำหรับควบคุมกระบวนการทำงานของเครื่องจักรและอุปกรณ์ในโรงงานอุตสาหกรรม

การใช้ PLC สำหรับควบคุมเครื่องจักรหรืออุปกรณ์ต่างๆ ในโรงงานอุตสาหกรรมจะมีข้อได้เปรียบกว่าการใช้ระบบของรีเลย์ซึ่งจำเป็นจะต้องเดินสายไฟ ฉะนั้นเมื่อมีความจำเป็นที่ต้องเปลี่ยนกระบวนการผลิต หรือลำดับการทำงานใหม่ ก็ต้องเดินสายไฟใหม่ ซึ่งเสียเวลาและเสียค่าใช้จ่ายสูง แต่เมื่อเปลี่ยนมาใช้ PLC แล้ว การเปลี่ยนกระบวนการผลิตหรือลำดับการทำงานใหม่นั้นทำได้โดยการเปลี่ยนโปรแกรมใหม่เท่านั้น นอกจากนี้แล้ว PLC ยังใช้ระบบโซลิด-สเตท ซึ่งน่าเชื่อถือกว่าเดิม การกินกระแสไฟฟ้าน้อยกว่า และสะดวกกว่าเมื่อต้องการขยายขั้นตอนการทำงานของเครื่องจักร จึงกล่าวได้ว่า PLC สามารถควบคุมเครื่องจักรได้ทุกชนิดอีกทั้งมีประสิทธิภาพสูงน่าเชื่อถือกว่าระบบควบคุมแบบเดิม เมื่อเปรียบเทียบกับระบบซีเคว้นซ์ (Sequence) หรือใช้การเดินสายไฟแบบเก่าแล้ว PLC มีข้อดีกว่าดังต่อไปนี้

- แก้ไข้ได้ง่าย
- ติดต่อกับระบบอื่นได้ง่าย
- ติดตั้งง่าย
- ลดการเดินสายไฟฟ้าควบคุม
- เนื้อที่ติดตั้งน้อยกว่า
- มีความน่าเชื่อถือสูงกว่า
- บำรุงรักษา และซ่อมแซมง่าย
- มีประสิทธิภาพการทำงานสูงกว่า

2.3 ส่วนประกอบของ PLC

PLC (Programmable Logic Controller) แบ่งออกได้เป็น 4 ส่วนด้วยกัน

2.3.1 ส่วนที่เป็นหน่วยประมวลผลกลาง (Central Processing Unit:CPU)

2.3.2 ส่วนที่เป็นหน่วยความจำ(Memory Unit)

2.3.3 ส่วนที่เป็นหน่วยรับข้อมูลและหน่วยส่งข้อมูล (Input /Output: I/O)

2.3.4 ส่วนที่เป็นอุปกรณ์ติดต่อภายนอก (Peripheral Devices)

2.3.1 หน่วยประมวลผลกลาง (CPU)

หน่วยประมวลผลกลางเป็นส่วนที่ทำหน้าที่ควบคุมการทำงานของ PLC โดยภายในจะประกอบด้วยวงจร Logic Gate ชนิดต่างๆ หลายชนิด และมี Microprocessor - based ใช้สำหรับแทนอุปกรณ์จำพวก รีเลย์ (Relay) เคาน์เตอร์ (Counter) ไทเมอร์ (timer) และซีเคว้นเซอร์ (Sequencer) เพื่อให้ผู้ใช้ได้ออกแบบใช้วงจรรีเลย์ แลดเดอร์ ลอจิก (Relay Ladder Logic) เข้าไปได้

หน่วยประมวลผลกลางจะรับข้อมูล (Read Input Data) จากอุปกรณ์ให้สัญญาณ (Sensing Device) ต่างๆ จากนั้นจะปฏิบัติการเก็บข้อมูล โดยใช้โปรแกรมจากหน่วยความจำ และส่งข้อมูลที่

เหมาะสมถูกต้องไปยังอุปกรณ์ควบคุม (Control Device) แหล่งของกระแสไฟฟ้าตรง (DC Current) สำหรับใช้สร้างแรงดันไฟฟ้าต่ำๆ (Low Level Voltage) ซึ่งใช้โดยโปรเซสเซอร์ (Processor) และ I/O Modules ซึ่งแหล่งจ่ายไฟนี้จะเก็บไว้ที่ หน่วยประมวลผลกลาง หรือแยกออกไปติดตั้งที่จุดอื่นก็ได้

การประมวลผลของหน่วยประมวลผลกลางจากโปรแกรม ทำได้โดยรับข้อมูลจากหน่วยอินพุตเข้ามาแล้วทำการประมวลผล แล้วจึงส่งข้อมูลที่ได้ออกไปยังเอาท์พุต จากนั้นก็วกกลับไปรับข้อมูลอินพุตเข้ามาอีก ทำซ้ำๆ ในลักษณะเช่นนี้ไปเรื่อยๆ การทำในลักษณะนี้เรียกว่า การสแกน (Scan Time)

นอกจากนี้ CPU ยังทำหน้าที่ รับส่งข้อมูลกับอุปกรณ์ติดต่อภายนอก (peripheral device) ตรวจเช็คตัวเองและหน่วยความจำ, ตรวจเช็คแหล่งจ่ายไฟฟ้า

2.3.2 หน่วยความจำ (Memory Unit)

หน่วยความจำเป็นองค์ประกอบหนึ่งที่สำคัญของระบบ เพราะใช้เป็นที่เก็บโปรแกรมและข้อมูลขนาดต่างๆ ของหน่วยความจำจะเป็นสิ่งกำหนดความสามารถของระบบ ระบบที่มีหน่วยความจำมากจะทำให้ผู้ใช้สามารถเขียนโปรแกรมที่มีความซับซ้อนได้มากขึ้นต่างๆ โดยขนาดของหน่วยความจำจะถูกแบ่งออกเป็นบิตข้อมูล (Data Bit) ภายในหน่วยความจำ 1 บิต ก็จะมีค่าสภาวะทางลอจิก 0 หรือ 1 แตกต่างกันแล้ว แต่คำสั่งหน่วยความจำของ PLC ประกอบด้วยหน่วยความจำชนิด RAM และ ROM หน่วยความจำชนิด RAM ทำหน้าที่ เก็บโปรแกรมของผู้ใช้และข้อมูลสำหรับใช้ในการปฏิบัติงานของ PLC ส่วน ROM ทำหน้าที่ เก็บโปรแกรมสำหรับใช้ในการปฏิบัติงานของ PLC ตามโปรแกรมของผู้ใช้ ROM (Read Only Memory) สามารถโปรแกรมได้แต่ลบไม่ได้ ถ้าชำรุดแล้วซ่อมไม่ได้

หน่วยความจำชนิดต่างๆ

1. RAM (Random Access Memory) RAM ทำหน้าที่เก็บโปรแกรมของผู้ใช้และข้อมูลที่ใช้ในการปฏิบัติงานของ PLC หน่วยความจำประเภทนี้จะมีแบตเตอรี่เล็กๆ ต่อไว้เพื่อใช้รักษาข้อมูลเมื่อเกิดไฟดับ การอ่านและเขียนโปรแกรมลงใน RAM ทำได้ง่ายมาก จึงเหมาะกับการใช้งานในระยะทดลองเครื่องที่มีการเปลี่ยนแปลงแก้ไขโปรแกรมบ่อยๆ

2. ROM ทำหน้าที่เก็บโปรแกรมสำหรับใช้ในการปฏิบัติงานของ PLC ตามโปรแกรมของผู้ใช้ หน่วยความจำแบบ ROM ยังสามารถแบ่งได้เป็น EPROM และ EEPROM

2.1 EPROM (Erasable Programmable Read Only Memory) หน่วยความจำชนิด EPROM นี้ จะต้องใช้เครื่องมือพิเศษในการเขียนโปรแกรม การลบโปรแกรมทำได้โดยใช้แสงอัลตราไวโอเลตหรือตากแดดร้อนๆ นานๆ มีข้อดีตรงที่โปรแกรมจะไม่สูญหายแม้ไฟดับ จึงเหมาะกับการใช้งานที่ไม่ต้องการเปลี่ยนโปรแกรม

2.2 EEPROM (Electrical Erasable Programmable Read Only Memory)
หน่วยความจำชนิดนี้ไม่ต้องใช้เครื่องมือพิเศษในการเขียนและโปรแกรม โดยใช้วิธีการทางไฟฟ้าเหมือนกับ RAM นอกจากนั้นไม่จำเป็นต้องมีแบตเตอรี่สำรองไฟเมื่อไฟดับ ราคาแพงกว่าแต่จะรวมคุณสมบัติที่ดีของ RAM และ EPROM เอาไว้

2.3.3 ส่วนของหน่วยรับข้อมูลและหน่วยส่งข้อมูล (I/O Unit)

หน่วยรับข้อมูล (Input) ทำหน้าที่รับสัญญาณจากอุปกรณ์ภายนอกที่เป็นสวิตช์ และตัวตรวจจับชนิดต่างๆ (Sensor) ของเครื่องจักรหรือกระบวนการ แล้วแปลงสัญญาณเป็น AC หรือ DC ที่เหมาะสมเพื่อส่งให้แก่หน่วยประมวลผลกลาง

สัญญาณอินพุตที่ดีจะต้องมีคุณสมบัติและหน้าที่ดังนี้

1. ทำให้สัญญาณเข้าได้ระดับที่เหมาะสมกับ PLC

2. การส่งสัญญาณระหว่างอินพุตกับ CPU จะติดต่อกันด้วยลำแสงซึ่งอาศัยอุปกรณ์ประเภทโฟโตทรานซิสเตอร์เพื่อต้องการแยกสัญญาณ (Isolate) ทางไฟฟ้าให้ออกจากกันเป็นการป้องกันไม่ให้ CPU เสียหายเมื่ออินพุตเกิดลัดวงจร

3. หน้าสัมผัสจะต้องไม่สั่นสะเทือน (Contact Chattering)

ในส่วนของหน่วยส่งข้อมูล (Output) จะทำหน้าที่ รับค่าสภาวะที่ได้จากการประมวลผลของหน่วยประมวลผลกลางแล้วนำค่าที่ได้นี้ไปขยายสัญญาณออกให้มีขนาดใหญ่ พอจะขับอุปกรณ์ภายนอก เช่น รีเลย์ โซลินอยด์ หลอดไฟ มอเตอร์ วาล์ว หรือปั๊ม เป็นต้น นอกจากนั้นแล้วยังทำหน้าที่แยกสัญญาณของหน่วยประมวลผลกลางออกจากอุปกรณ์เอาท์พุตเพื่อป้องกันความเสียหายที่อาจจะเกิดขึ้นได้ โดยปกติเอาท์พุตนี้จะสามารถขับโหลดได้ด้วยกระแสไฟฟ้าประมาณ 1-2 แอมแปร์ ถ้าโหลดต้องการกระแสมากกว่านี้จะต้องต่อเข้ากับอุปกรณ์ขับหรือขยายอีกที เช่น รีเลย์ โซลิด-สเตทรีเลย์ และคอนแทคเตอร์

2.3.4 อุปกรณ์ติดต่อภายนอก (Peripheral Devices)

ทำหน้าที่ ป้อนโปรแกรมของผู้ใช้ลงในหน่วยความจำของ PLC นอกจากนั้นแล้วยังทำหน้าที่ติดต่อระหว่างผู้ใช้กับ PLC เพื่อให้ผู้ใช้สามารถตรวจการปฏิบัติงานของ PLC และผลการควบคุมเครื่องจักรและกระบวนการตามโปรแกรมควบคุมที่ผู้ใช้เขียนขึ้นได้อีกด้วย หน้าที่ของอุปกรณ์ติดต่อภายนอกได้แก่

1. ป้อนโปรแกรมเข้าไปในหน่วยความจำของระบบ
2. ใช้ในการแก้ไขโปรแกรม
3. ใช้ในการพิมพ์โปรแกรม
4. ใช้แสดงสภาวะการควบคุม

Computer กับ PLC

การเชื่อมต่อกันระหว่างคอมพิวเตอร์กับ PLC นั้น ส่วนใหญ่นั้นจะใช้สาย RS-232C เป็นตัวเชื่อมต่อ และ PLC สามารถใช้ software ของคอมพิวเตอร์เพื่อทำหน้าที่ได้หลายๆ อย่าง เช่นการใช้ Software ในการป้อนโปรแกรม แก้โปรแกรม ดูการทำงานของโปรแกรม เป็นต้น ดังนั้น software ของแต่ละบริษัทจึงไม่เหมือนกันแต่มีจุดประสงค์การใช้งานที่ใกล้เคียงกัน

2.4 การทำงานของ PLC

เครื่องจักรที่ควบคุมด้วย PLC จะมีความสามารถเขียนโปรแกรมการทำงานของเครื่องจักรและมีความยืดหยุ่นในการเขียนโปรแกรม เช่น เปลี่ยนแปลงแก้ไขเพิ่มเติมก็สามารถกระทำได้ ซึ่งจะรวมถึงมี ไทเมอร์ (Timer) เคาท์เตอร์ (Counter) หรือฟังก์ชั่นคำสั่งพิเศษต่างๆ มากมาย เช่น MOV Data เพื่อที่จะได้ใช้ในงานควบคุมอุปกรณ์ภายนอก ไม่ว่าจะเป็น มอเตอร์ โซลีนอด์ย หรือหลอดไฟ เป็นต้น นอกจากนี้ยังมีการติดต่อสื่อสารระหว่าง PLC กับ Computer เพื่อแลกเปลี่ยนข้อมูลระหว่างกันหรืออาจจะติดต่อกับอุปกรณ์อื่น เช่น จอสัมผัส (Touch Screen) เพื่ออำนวยความสะดวกต่อสัญญาณอินพุตและสัญญาณเอาท์พุต ยิ่งไปกว่านั้นการติดต่อกับคอมพิวเตอร์เพื่อให้คอมพิวเตอร์เป็นตัวควบคุมอีกทีหนึ่ง ซึ่งจะทำให้ขีดความสามารถในการควบคุมนั้นสูงขึ้นอีกด้วย โดยการควบคุมนั้นสามารถทำได้ โดยการเขียนวงจรแลดเดอร์ (Ladder Diagram) ซึ่งแลดเดอร์ไดอะแกรมจัดเป็นภาษาสัญลักษณ์ที่สามารถดูตามโครงสร้าง แล้วเข้าใจการทำงาน แต่ที่เวลาที่ PLC จะทำงานจะอาศัยชุดคำสั่ง (Instructions) ทำงาน โดยวิธีการเขียนลงในส่วนหน่วยความจำ ข้อมูลในหน่วยความจำนั้นจะจัดเก็บเป็นรหัส (Code) ไม่สามารถจัดเก็บแลดเดอร์โดยตรงได้ ดังนั้นผู้ใช้จึงจำเป็นต้องเข้าใจชุดคำสั่งเพราะชุดคำสั่งนั้นก็แปลงภาษามาจากแลดเดอร์ไดอะแกรมนั่นเอง

วงจรตรรกะ (ลอจิก)

จากพื้นฐานความรู้หลักการของเลขฐานชนิดต่างๆแล้ว หลักการทำงานของ PLC ก็ยังใช้วงจรตรรกะ (ลอจิก) เพื่อให้เกิดสัญญาณเอาท์พุตที่มีเงื่อนไข (สัญญาณอินพุต) ชนิดต่างๆ หลักการของวงจรตรรกะ มีดังต่อไปนี้

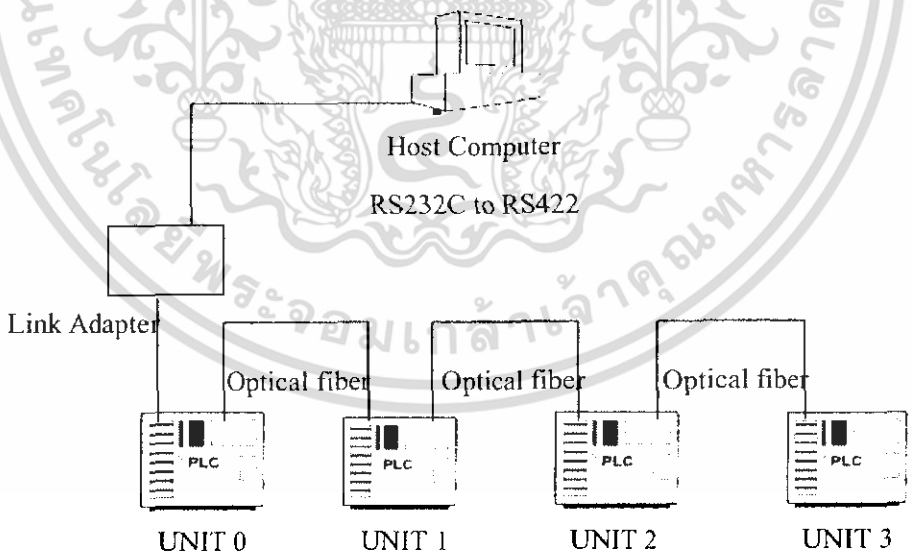
วงจรตรรกะหมายถึง วงจรไฟฟ้าที่ประกอบไปด้วย อุปกรณ์อิเล็กทรอนิกส์ หรือระบบรีเลย์ที่มีสัญญาณเพียง 2 ระดับ หรือ 2 สภาวะเท่านั้น PLC ใช้สัญญาณไฟฟ้า 2 ระดับแทน 2 เหตุการณ์ที่ต่างกัน เช่น การเปิดวาล์ว การปิด-เปิด สวิตช์ เป็นต้น วงจรตรรกะมี 2 ชนิดคือ แบบบวก (Positive Logic) แบบลบ (Negative Logic) ลอจิกบวก จะใช้สัญญาณไฟระดับสูงแทนสภาวะลอจิก “1” และใช้สัญญาณไฟฟ้าระดับต่ำแทนสภาวะลอจิก “0” ส่วนวงจรลอจิกแบบลบจะใช้

สัญญาณไฟฟ้าระดับต่ำแทนสภาวะลอจิก "1" และใช้สัญญาณไฟฟ้าระดับสูงแทนสภาวะลอจิก "0"

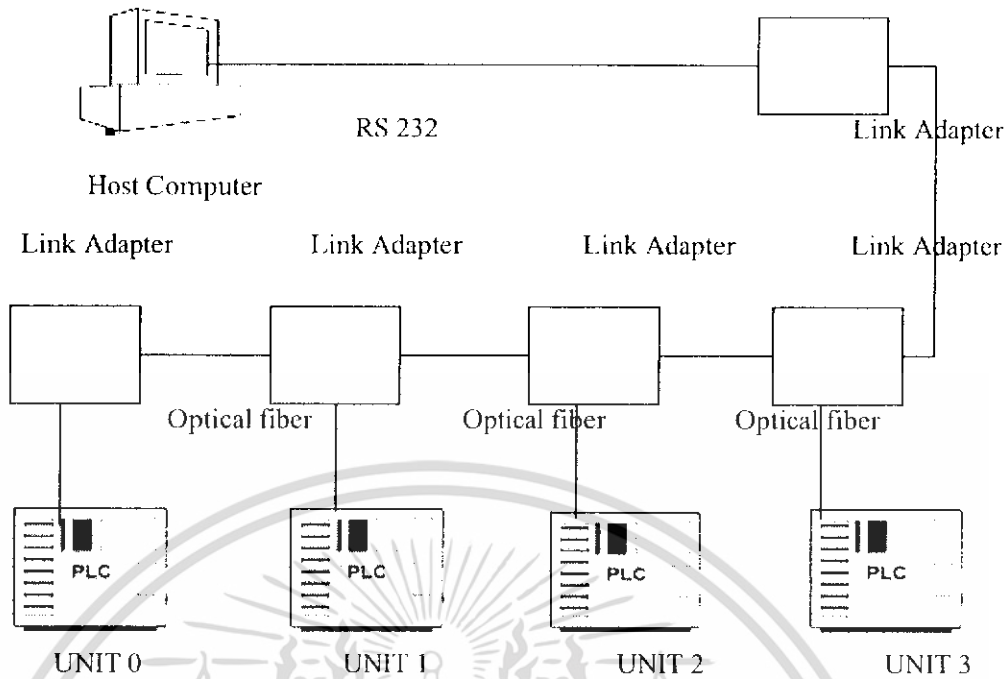
สภาวะทางลอจิก คือ สภาวะ "1" หรือ "0" ใช้แทนการทำงานของอุปกรณ์ที่เปลี่ยนแปลง 2 สภาวะ ระบบควบคุมที่ใช้ระบบรีเลย์ และ PLC จะนำเอาสภาวะของอุปกรณ์เหล่านี้มาปฏิบัติลอจิกด้วยกัน เพื่อให้เข้ากันกับเงื่อนไขการควบคุม ปฏิบัติการลอจิกประกอบด้วย AND OR และ NOT เพื่อทำให้สภาวะอินพุตต่างๆ เช่น A, B ทำให้เกิดเอาท์พุต Y เป็นต้น ส่วนพีชคณิตบูลีนมีไว้สำหรับอธิบายความสัมพันธ์ทางลอจิกเพื่อที่จะให้เข้าใจได้ง่ายขึ้น

2.5 การติดต่อแบบพีซีลิงค์(PC Link System)

เป็นการเชื่อมโยงสื่อสารข้อมูลในลักษณะของการแบ่งพื้นที่ ในการอ่านเขียนระหว่างเครื่องควบคุม PLC/PC แต่ละตัว เครื่องควบคุม PLC/PC แต่ละตัวสามารถรับรู้ข่าวสารซึ่งกันและกันได้ มาตรฐานในการสื่อสารข้อมูลชนิดนี้มักจะเป็นแบบ RS-422A หรือ RS 485 หรือรูปแบบการสื่อสารผ่านสายใยแก้วนำแสง (Fiber optic) หรือผ่านทางสายโคแอคเชียล หรือสายคู่ตีเกลียว โดยจะมีตัว Link Adapter เป็นตัวแปลงรูปแบบข้อมูลให้เป็นมาตรฐานตรงกัน อาจจะมีการเชื่อมโยงกับเครื่องคอมพิวเตอร์เข้ามาในระบบการลิงค์ เพื่อสนับสนุนการทำงานของเครื่องควบคุม PLC/PC ซึ่งจะเรียกว่าโฮสคอมพิวเตอร์ (Host Computer) มีลักษณะการต่อวงจรดังนี้



ภาพที่ 2.1 แสดงการต่อแบบ Serial link



ภาพที่ 2.2 แสดงการต่อแบบ Parallel link

ตัวอย่างการจัดพื้นที่ของเครื่องควบคุม PLC /PC กำหนดให้มีเครื่องควบคุมจำนวน 4 ชุด เชื่อมโยงในเครื่องควบคุมแต่ละเครื่อง จะมีพื้นที่ของรีเลย์ที่ทำหน้าที่ในการเชื่อมโยงข้อมูล (Link-Relay) ถ้ามีเครื่องควบคุมที่เชื่อมต่อในระบบมากพื้นที่ก็จะถูกแบ่งออกไปตามจำนวน เช่น 4 ชุด พื้นที่จะถูกแบ่งออกเป็น 4 ส่วน เช่นเครื่องควบคุมนี้มีพื้นที่ Link Relayจำนวน 64 word ก็จะถูกแบ่งเป็นส่วนละ 16word ส่วนแรกจะเป็นพื้นที่ของเครื่องควบคุมตัวที่ 1 ส่วนที่ 2 ก็จะเป็นพื้นที่ของเครื่องควบคุมตัวที่ 2 และตัวที่ 3 และตัวที่ 4 ตามลำดับ โดยพื้นที่ส่วนอื่นจะถูกกันไว้ไม่ให้สามารถเขียนข้อมูลลงไปได้ (Read Only) ยกเว้นพื้นที่ของตัวเอง ดังนั้นถ้า PLC/PC เครื่องที่ 1 ต้องการจะรับข้อมูลจากเครื่องที่ 3ก็สามารถอ่านได้ จากพื้นที่ของเครื่องควบคุมส่วนที่ 3 และถ้าต้องการส่งข้อมูลให้ก็เขียนลงในพื้นที่ของส่วนแรก แล้ว PLC ตัวที่ 3 จะทำการอ่านข้อมูลในส่วนแรก ก็จะสามารถทราบข้อมูลของเครื่องควบคุมPLCตัวที่1

PLC เครื่องที่ 1	PLC เครื่องที่ 2	PLC เครื่องที่ 3	PLC เครื่องที่ 4
LR 00 – LR 15	LR 00 – LR 15	LR 00 – LR 15	LR 00 – LR 15
LR 16 – LR 31	LR 16 – LR 31	LR 16 – LR 31	LR 16 – LR 31
LR 32 – LR 47	LR 32 – LR 47	LR 32 – LR 47	LR 32 – LR 47
LR 48 – LR 63	LR 48 – LR 63	LR 48 – LR 63	LR 48 – LR 63

ภาพที่ 2.3 แสดงการแบ่งพื้นที่สำหรับ PC Link System

การเชื่อมต่อ PLC เข้าด้วยกัน

โดยใช้หลักของ PC Link ต้องใช้หลักการเบื้องต้นดังต่อไปนี้

1. นำเครื่องควบคุมจำนวน 2 ชุดขึ้นไป ทำการต่อสายโคแอกเชียลเข้าที่ SYSMAC LINK โมดูล
2. ทำการตั้งหมายเลขประจำเครื่อง (Unit Number) ให้ต่างกันเริ่มต้นจาก 1, 2, 3, 4....16
3. ทำการตั้งค่าบน DIP SWITCHS ดังนี้

ตารางที่ 2.1 แสดงการตั้งค่าบน DIP SWITCHS

หมายเลข DIP SWITCHS	รายละเอียดในการกำหนดค่า	
	ON	OFF
1	Start Test	Stop test
2	Start Link	Stop Link
3	Operating On Level 0	Operating On Level 1
4	Not Use	Not Use

*หมายเหตุ---SW 1OFF

SW 2 ON

SW 3 ON

4. ทำการตั้งค่าในหน่วยความจำสำรอง Auxiliary Relay (AR) เพื่อกำหนดพื้นที่ของหน่วยความจำ ดังตารางข้างล่างนี้

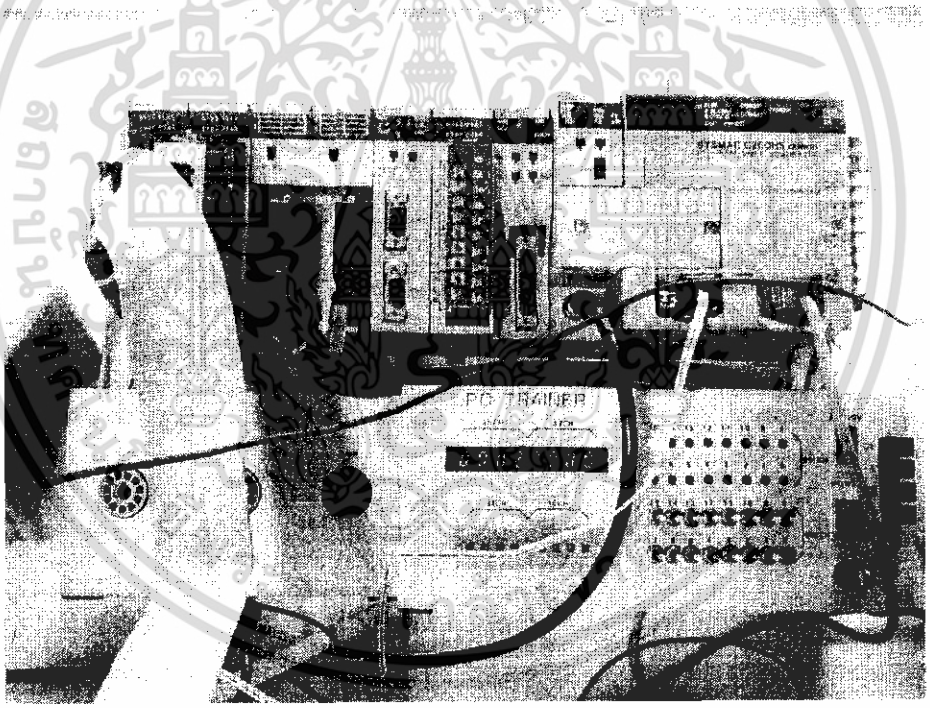
ตารางที่ 2.2 แสดงการตั้งค่าในหน่วยความจำสำรอง Auxiliary Relay (AR)

Operation Level 0		Operation Level 1		Memory Setting
AR 700	AR 701	AR 704	AR 705	
0	0	0	0	No meaning
1	0	1	0	LR area only
0	1	0	1	DM area only
1	1	1	1	LR and DM

ตารางที่ 2.3 แสดงการเลือกจำนวน WORD สำหรับเชื่อมโยงข้อมูล

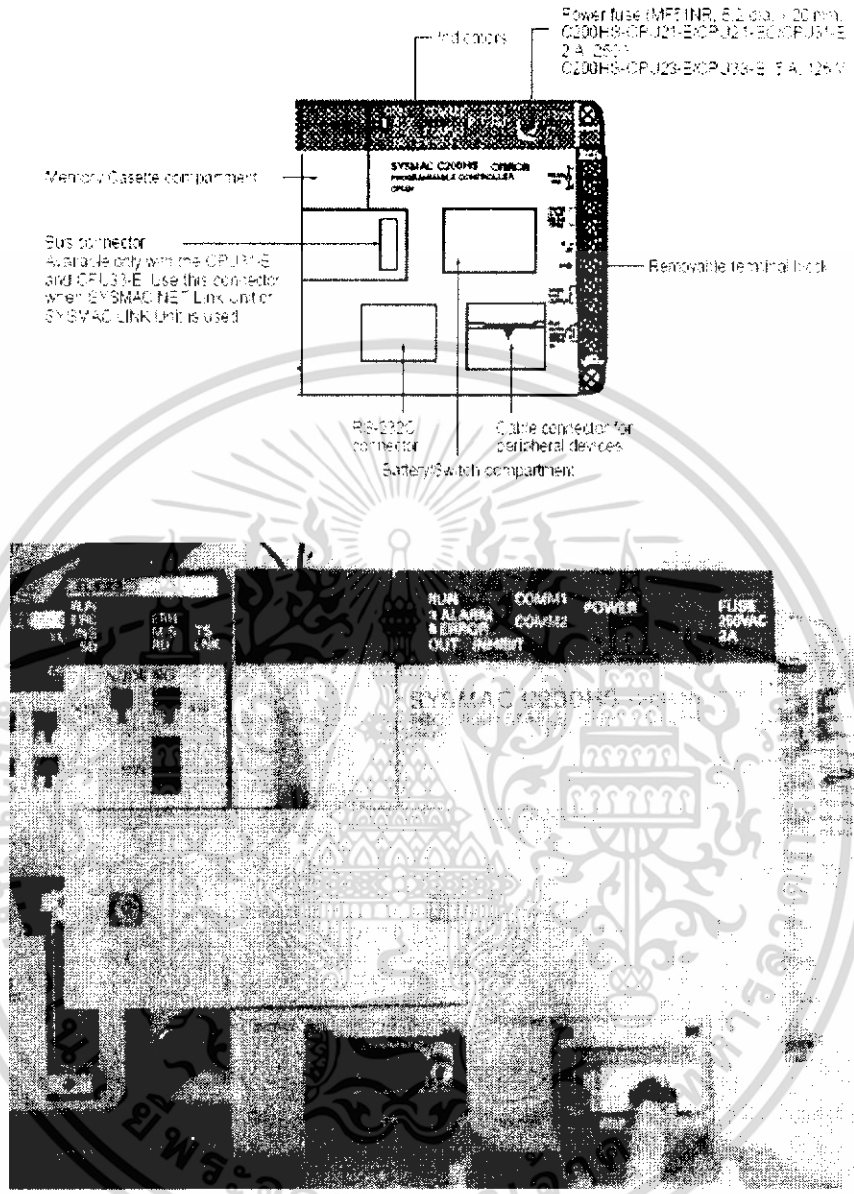
Operation Level 0		Operation Level 1		Word per Node		MAX node
AR 702	AR 703	AR 706	AR 707	LR	DM	
0	0	0	0	4	8	16
1	0	1	0	8	16	8
0	1	0	1	16	32	4
1	1	1	1	32	64	2

2.6 ข้อมูลของPLC รุ่น C 200 HS



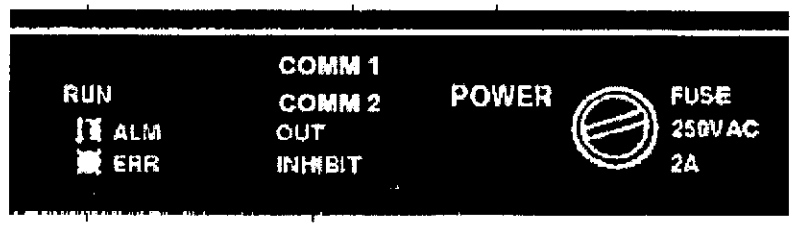
ภาพที่ 2.4 แสดง PLC รุ่น C200 HS

2.6.1 ส่วนประกอบของ CPU ของ C 200 HS



ภาพที่ 2.5 แสดง CPUของPLC รุ่น C200 HS

1. Indicator (ตัวชี้สถานะและไฟแสดงสถานะ) ต่างๆ



ภาพที่ 2.6 แสดงพื้นที่ส่วนของ Indicator

- Run ไฟสีเขียวจะติดเมื่อ CPU กำลังทำงานตามปกติ
- ALM ไฟสีส้มจะติดเมื่อเกิดความผิดพลาดของระบบขึ้น แต่ยังไม่ถึงกับขั้นที่ CPUหยุดการทำงาน
- ERR ไฟสีแดงจะติดเมื่อเกิดความผิดพลาดถึงขั้น CPU หยุดการทำงานแล้วจะทำให้ไฟการ RUN หยุด การทำงานและหน่วย OUTPUT ก็จะหยุดการทำงานด้วย
- COMM1 ไฟสีส้มจะติดก็ต่อเมื่ออุปกรณ์ภายนอกถูกต่อกับระบบและทำงาน
- COMM2 ไฟสีส้มจะติดก็ต่อเมื่อสาย RS-232 ถูกเชื่อมต่อระหว่าง PLC กับ อุปกรณ์อื่น
- Power ไฟสีเขียวจะติดเมื่อมีไฟจ่ายให้ PLC

2. Fuse เป็นฟิวส์ของเครื่อง CPU ของ PLC ช่วยป้องกันเมื่อเกิดความผิดพลาดทางวงจรขึ้นมา

3. Removable terminal block เป็นส่วนที่รับไฟฟ้ามาจากภายนอก หรือ เป็นส่วนที่ส่งสัญญาณต่างๆ ไปยังอุปกรณ์ภายนอก

4. Cable connector for peripheral devices เป็นส่วนที่ใช้เชื่อมต่อกับอุปกรณ์ภายนอก เช่น Programming Console

5. RS-232 Connector เป็นส่วนที่ใช้สำหรับเชื่อต่อกับอุปกรณ์อื่น โดยผ่านสาย RS-232.

2.6.2 ทางด้านการเปรียบเทียบระหว่าง Relay และ PLC ของ Omron

PLC นั้นจะใช้ระบบคำศัพท์ซึ่งต่างจาก Relay แต่ความหมายนั้นเหมือนกันซึ่ง ตามตาราง ด้านล่างนี้จะเปรียบเทียบให้เห็นเฉพาะส่วนสำคัญเท่านั้น

ตารางที่ 2.4 เปรียบเทียบความแตกต่างระหว่างศัพท์ของRelay และ PLC ของ Omron

Relay Term	Equivalent
Contact	Input or condition
Coil	Output or work bit
NO Relay	Normally open Condition
NC Relay	Normally close Condition

ซึ่งในความเป็นจริงแต่ละ term ไม่จำเป็นต้องมีความหมายตามด้านบนนี้เสมอไป เช่น Condition นั้นจะใช้ในการอธิบาย Ladder อย่างเดียวไม่สามารถใช้ในการเขียนได้

ตารางที่ 2.5 แสดงประสิทธิภาพ CPU ของ PLC รุ่น C200 HS

Function	C200HS					
	CPU01-E	CPU21-E	CPU31-E	CPU03-E	CPU23-E	CPU33-E
Built-in clock/calendar	Yes					
Error log	Yes[1]					
Data Trace	Yes					
Differential Monitor	Yes					
Expansion DM	32K words max.[2]					
General-use DM	6K words					
Ladder Program capacity	15.2K words max.[2]					
SR Area	SR 236 to SR 255 and SR 256 to SR 299					
New instructions: (See 1-8-3 Larger Instruction Set for a list of the 36 new instructions.)	Yes					
Network Instructions: NETWORK SEND - SEND(90) NETWORK RECEIVE - RECV(98)	No	No	Yes	No	No	Yes
Power Supply	AC			DC		

Note – 1. CPU C-200HS นี้สามารถบันทึกวันและเวลาที่เกิดปัญหาการรบกวนทาง Power

2. ในส่วนของ word ที่ 16K UM จะถูกติดตั้งในส่วน Expansion DM

2.6.3 โครงสร้างของพื้นที่ข้อมูล (DATA AREAS)

เมื่อมีการกำหนดพื้นที่ข้อมูลต่างๆ จะใช้ตัวย่อซึ่งมาจากศัพท์เต็มของแต่ละพื้นที่ เช่น IR หรือSR Areas อย่างไรก็ตามคำย่อของ IRหรือSR Areas จะถูกนำไปใช้บ่อยในการอธิบายความหมายต่างๆ ส่วน Data Areas อื่นๆ จะถูกสมมุติให้อยู่ในรูปของ IR หรือ SR Areas เพราะว่า IR และ SR Areas นั้นจะทำงานอยู่ตลอด

โดยในความเป็นจริงข้อมูลของData Areas นั้น จะอยู่ข้างในแต่พื้นที่ของ TC Areas จะออกแบบให้อยู่ในรูปของ address ซึ่งการระบุที่อยู่ของพื้นที่ TC นี้ จะอยู่ในรูปแบบของ Bitsและ Words พื้นที่ของ TC นี้จะประกอบไปด้วย TC Numbers และพื้นที่ของTC Areasนี้ จะถูกนำไปใช้ในพื้นที่พิเศษของพวก Timer และ Counter ส่วน Date Areas อื่นๆที่เหลือนอกจากนี้ เช่น IR,SR,HR,DM,ARและLR Areas ก็จะอยู่ในรูปของ words ต่างๆ และแต่ละ words นั้นจะประกอบด้วย 16 bits ซึ่งจะเริ่มต้นจาก 00 จนถึง 15 จากขวาไปซ้าย ซึ่งBit 00 จะถูกเรียกว่า Rightmost ส่วนทาง Bit 15 จะถูกเรียกว่า Leftmost

หลักการทำงานของ Bits และ Words

เมื่อบาง Bits หรือ Words ใน data areas ไม่ถูกนำไปใช้ในจุดที่ต้องการใช้งานพวกมันก็จะถูกนำไปใช้งานในจุดอื่นๆ Words และ Bits ที่นิยมนำไปใช้งานนี้จะเรียกว่า Work words และ Work Bits ซึ่งโดยส่วนมากเราจะไม่เรียกว่า Bits เฉยๆ แต่เราจะเรียกพวกมันว่า Work bits ซึ่งพวกมันจะถูกอธิบายในแต่ละพื้นที่ของมัน

2.6.4 ในส่วนของ Memory Areas

ตารางที่ 2.6 ประกอบด้วยชื่อ ขนาด และ ช่วงที่ใช้งานของแต่ละพื้นที่

Area	Size	Range	Comments
I/O Area	480 bits	IR 000 to IR 029	I/O words are allocated to the CPU Rack and Expansion I/O Racks by slot position.
Group-2 High-density I/O Unit and B7A Interface Unit Area	320 bits	IR 030 to IR 049	Allocated to Group-2 High-density I/O Units and to Group-2 B7A Interface Units 0 to 9.
SYSMAC BUS Area	800 bits	IR 050 to IR 099	Allocated to Remote I/O Slave Racks 0 to 4.[1]
Special I/O Unit Area	1,600 bits	IR 100 to IR 199	Allocated to Special I/O Units 0 to 9.
Optical I/O Unit and I/O Terminal Area	512 bits	IR 200 to IR 231	Allocated to Optical I/O Units and I/O Terminals.
Work Area 1	64 bits	IR 232 to IR 235	For use as work bits in the program.
Special Relay Area 1	312 bits	SR 23600 to SR 25507	Contains system clocks, flags, control bits, and status information.
Special Relay Area 2	704 bits	SR 250 to SR 299 (298 to 299 reserved by system)	Contains flags, control bits, and status information.
Macro Area	64 bits	SR 290 to SR 293	Inputs
	64 bits	SR 294 to SR 297	Outputs
Work Area 2	3,392 bits	IR 300 to IR 511	For use as work bits in the program.
Temporary Relay Area	8 bits	TR 00 to TR 07	Used to temporarily store and retrieve execution conditions when programming certain types of branching ladder diagrams.
Holding Relay Area	1,600 bits	HR 00 to HR 99	Used to store data and to retain the data values when the power to the PC is turned off.
Auxiliary Relay Area	448 bits	AR 00 to AR 27	Contains flags and bits for special functions. Retains status during power failure.
Link Relay Area	1,024 bits	LR 00 to LR 63	Used for data links in the PC Link System.[1]
Timer/Counter Area	512 counters/ timers	TC 000 to TC 511	Used to define timers and counters, and to access completion flags, PV, and SV. Interval timers 0 through 2 and high-speed counters 0 through 2 provided in separate area. TIM 000 through TIM 015 can be refreshed via interrupt processing as high-speed timers.
Data Memory Area	6,144 words	DM 0000 to DM 6143	Read/Write
	1,000 words	DM 0000 to DM 0999	Normal DM.
	1,000 words	DM 1000 to DM 1999	Special I/O Unit Area[2]
	4,000 words	DM 2000 to DM 5999	Normal DM.
	31 words	DM 6000 to DM 6030	History Log
	144 words)	DM 6100 to DM 6143	Link test area (reserved)
Fixed DM Area	512 words	DM 6144 to DM 6599	Fixed DM Area (read only)
	56 words	DM 6600 to DM 6655	PC Setup
Expansion DM Area	3,000 words max.	DM 7000 to DM 9999	Read only

Note – 1. การที่จะใช้พื้นที่ของ Word และ Bit ต่างๆ ตามตาราง ควรใช้ให้ตรงกับวัตถุประสงค์ที่ต้องการจะใช้งาน

2. DM7000-7999 นี้ สามารถที่จะถูกกำหนดให้เป็นพื้นที่พิเศษของ I/O Area

2.6.5 หลักการใช้งานเบื้องต้นของ PLC

จากเนื้อหาที่จะกล่าวต่อไปนี้ จะพูดถึงหลักการทำงานเป็นขั้นตอนรวมถึงการเขียนโปรแกรม หากคุณมี PC ที่สามารถเขียนโปรแกรมได้คุณจะต้องเริ่มการคิดเป็นขั้นตอนโดยทำงานเป็นขั้นๆ ตั้งแต่ Step ที่ 1 2 3 4.....10 หรือต่อไปเรื่อยๆ ซึ่งต่อไปนี้ จะพูดถึงขั้นตอนการคิดเป็น step

1. ตัดสินใจดูว่าเราต้องการที่จะทำอะไร ควบคุมระบบอะไร ใช้เวลาเท่าไหร่

2. ดูว่างานที่เราจะควบคุมต้องการใช้อะไรบ้าง ติดตั้ง Guide C 200HS ดูว่าการเชื่อมต่อต้องใช้อะไรบ้าง

3. ในการเขียนควรเขียนในกระดาษก่อน ควรกำหนด Input/Output ของอุปกรณ์ต่างๆ และตัดสินใจในการใช้ว่า I/O ตัวไหนใช้ บิตไหน หากระบบใช้ในการ Link เชื่อมต่อกับตัวอื่นให้ดูตำแหน่งพื้นที่ของการใช้พื้นที่ต่างๆ ในตาราง Memory area

4. ในการใช้สัญลักษณ์ต่างๆของ PLC ในการเขียน Ladder ควรจะเขียนให้ถูกหลักการซึ่งสามารถศึกษาหลักการเขียนได้จากหนังสือต่างๆ

5. ทำการป้อนข้อมูลลงในเครื่อง PC

6. ทำการ Debug โปรแกรม แล้วหากมีข้อผิดพลาดก็ทำการแก้ไขที่ตรงจุด

7. ทำการต่อสายจาก PLC ไปยังอุปกรณ์ต่างๆ

8. ทำการ Test โปรแกรมกับตัว simulator เพื่อทดลองโปรแกรมต่างๆ

9. บันทึกไฟล์ไว้ 2 ไฟล์เพื่ออาจมีการย้าย โปรแกรมไปติดตั้งยังอีกเครื่อง

บทที่ 3
การติดต่อสื่อสารและการส่งผ่านข้อมูล

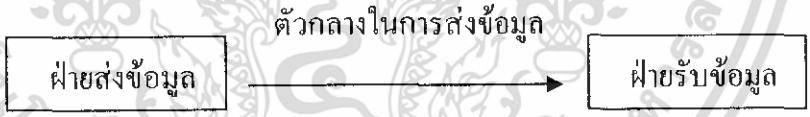
3.1 กล่าวนำ

การสื่อสารแบบ Host link เป็นการเชื่อมต่อระหว่าง เครื่อง PLC กับ คอมพิวเตอร์ทั่วไป โดยผ่านทางพอร์ตของคอมพิวเตอร์ (COM1: หรือ COM2:) ส่วนมากจะนิยมใช้มาตรฐานการสื่อสารข้อมูลแบบอนุกรมเพื่อให้สามารถควบคุม PLC จากคอมพิวเตอร์ได้ สำหรับคอมพิวเตอร์ 1 เครื่องสามารถต่อเข้ากับ PLC ได้จำนวนมาก โดยใช้การเชื่อมต่อหลายๆ ตัวเข้าด้วยกันเรียกว่า PC LINK ในการติดต่อแบบ Host Link จะต้องผ่านอุปกรณ์ที่เรียกว่า Host Link Unit ซึ่งจะต้องตั้งค่าต่างๆ ที่จำเป็นที่ใช้ในการติดต่อแบบ Host Link Unit SLK 23 ผ่านสวิตช์ของเครื่อง PLC

3.2 การสื่อสารข้อมูลทั่วไป

ส่วนประกอบเบื้องต้นในการสื่อสารข้อมูลแบ่งได้ออกเป็น 3 ส่วนคือ

1. ฝ่ายส่งข้อมูล (Transmitter)
2. ตัวกลางในการส่งผ่านข้อมูล (Medium)
3. ฝ่ายรับข้อมูล (Receiver)



ภาพที่ 3.1 แสดงส่วนประกอบหลักในการสื่อสารข้อมูล

3.3 ความรู้เบื้องต้นเกี่ยวกับพอร์ตอนุกรม

การเคลื่อนย้ายข้อมูลจากคอมพิวเตอร์ไปยังอุปกรณ์ต่อพ่วงภายนอก หรือคอมพิวเตอร์ด้วยกันมีด้วยกัน 2 แบบคือ การรับส่งข้อมูลแบบขนาน และการรับส่งแบบอนุกรม

การรับส่งข้อมูลแบบขนานเป็นการรับหรือส่งข้อมูลคราวละ 4 หรือ 8 บิต ในเวลาเดียวกัน ทำให้การรับและส่งข้อมูลมีความเร็วสูง ทว่าจำนวนรีจิสเตอร์ที่ใช้ในการถ่ายทอดข้อมูลต้องมีมากเท่ากับจำนวนบิตของข้อมูลที่ทำการถ่ายทอดด้วย นอกจากนี้ยังมีสายที่ใช้สำหรับควบคุมและตรวจสอบการรับส่งข้อมูลด้วยซึ่งอาจใช้มากเป็น 2 เท่าของจำนวนบิตของข้อมูล ข้อจำกัดของการถ่ายข้อมูลแบบขนานก็คือ ระยะทางในการถ่ายทอดข้อมูลโดยปกติอยู่ที่ 10-15 ฟุต

ในขณะที่การรับส่งข้อมูลแบบอนุกรมจะเป็นการรับส่งข้อมูลครั้งละ 1 บิต โดยมีรูปแบบการรับส่งเป็นมาตรฐาน ต้องมีการตรวจสอบความพร้อมในการรับส่งข้อมูลของตัวรับและตัวส่ง ซึ่งการรับส่งข้อมูลแบบอนุกรมนั้น จะมีข้อดีในเรื่องของจำนวนสายสัญญาณที่ใช้น้อยมาก และไม่แปรผันตามจำนวนบิตของข้อมูล ระยะทางในการส่งสูงกว่าแบบขนาน โดยปกติ พอร์ตอนุกรมแบบ RS-232 จะสามารถต่อสายได้ยาว 50 ฟุต โดยประมาณ

3.3.1 การสื่อสารข้อมูลแบบอนุกรม

เมื่อพิจารณาการส่งข้อมูลในแบบอนุกรมให้ดีจะพบว่า ปัญหาหนึ่งที่ควรจะเกิดขึ้นอยู่เสมอก็คือการตัดสินว่าข้อมูลที่ได้รับนั้นมีจุดเริ่มต้นที่ใด ดังนั้นจึงมีการกำหนดข้อตกลงในการสื่อสารขึ้นเพื่อแก้ปัญหาข้อตกลงดังกล่าวเราเรียกว่า โปรโตคอล (Protocol) ของการสื่อสารข้อมูลแบบอนุกรมสามารถแบ่งออกเป็น 2 ประเภทใหญ่ๆ คือ โปรโตคอลสำหรับการสื่อสารข้อมูลแบบซิงโครนัส (Synchronous) และโปรโตคอลสำหรับการสื่อสารข้อมูลแบบอะซิงโครนัส (Asynchronous) การสื่อสารข้อมูลแบบซิงโครนัสนั้นข้อมูลจะถูกส่งออกไปอย่างสม่ำเสมอ ช่วงเวลาระหว่างบิตและระหว่างเวิร์ดจะมีค่าเท่ากันเสมอ ดังนั้นในการสื่อสารข้อมูลอนุกรมในแบบซิงโครนัสจึงต้องมีสัญญาณเพิ่มเติมเพื่อกำกับการส่ง ว่าควรจะส่งเมื่อใด และควรจะหยุดเมื่อใด แบบซิงโครนัสจะเป็นระบบที่มีความเร็วสูง แต่ก็ยังต่ำกว่าการสื่อสารแบบขนาน

การสื่อสารข้อมูลแบบอะซิงโครนัสนี้เป็นหัวใจของการสื่อสารข้อมูลผ่านทางสายโทรศัพท์ในปัจจุบัน การสื่อสารแบบนี้ช่วงระยะเวลาระหว่างบิตจะมีค่าเท่ากันเช่นเดียวกับแบบซิงโครนัส แต่จะมีระยะห่างระหว่างเวิร์ดนั้นแตกต่างออกกันไปเป็นกี่ วินาที นาที ชั่วโมง เป็นต้น ได้ทั้งสิ้นขึ้นอยู่กับว่าฝ่ายรับสามารถรอได้หรือไม่เท่านั้น เมื่อไม่มีข้อกำหนดทางด้านระยะเวลาระหว่างเวิร์ดเพื่อแก้ปัญหานี้ จึงมีการกำหนดข้อตกลงเกี่ยวกับรูปแบบของข้อมูลที่จะส่งให้ทางผู้รับสามารถเข้าใจว่าจุดใดเป็นจุดเริ่มต้นของเวิร์ด

ข้อกำหนดดังกล่าวกำหนดให้แต่ละเวิร์ดจะต้องขึ้นต้นด้วยบิตที่เรียกว่า บิตเริ่มต้น (Start Bit) ซึ่งจะต้องมีข้อมูลที่เป็นลอจิก 0 เสมอ จากนั้นตามด้วยบิตข้อมูลที่ต้องการส่งซึ่งมีความยาว 5-8 บิต ถัดจากบิตข้อมูล ก็จะเป็นบิตแบบพาริตี้บิตซึ่งทำหน้าที่เป็นบิตสำหรับตรวจสอบความถูกต้องของข้อมูลที่ได้รับว่ามีความถูกต้องหรือไม่ บิตพาริตี้นี้จะมี 2 ประเภท คือ อีเวนพาริตี้ (Even Parity) ซึ่งจะกำหนดจำนวนบิตที่เป็นลอจิก 1ในสายข้อมูล แล้วได้เป็นจำนวนคู่ ส่วนออดพาริตี้ (Odd Parity) ซึ่งจะกำหนดจำนวนบิตที่เป็นลอจิก 1ในสายข้อมูล แล้วได้เป็นจำนวนคี่ ในการส่งข้อมูลบางครั้งอาจจะไม่มีการใช้บิตพาริตี้ก็ได้ ถ้าหากการสื่อสารในครั้งนั้นมีความน่าเชื่อถือสูง คือมีสัญญาณรบกวนต่ำเป็นการเพิ่มความเร็วในการสื่อสารได้ด้วย

บิตสุดท้ายในรูปแบบก็คือ บิตสิ้นสุดข้อมูล (Stop Bit) ในการส่งข้อมูลแบบอะซิงโครนัส (Asynchronous) นั้น บิตสิ้นสุดข้อมูลจะต้องถูกต่อเพิ่มท้ายข้อมูลทุกครั้งเพื่อแจ้งการสิ้นสุดข้อมูล

บิตสิ้นสุดข้อมูลสามารถกำหนดให้มี 1 หรือ 2 ก็ได้ แต่ต้องกำหนดให้ตรงกันทั้งในคอมพิวเตอร์และอุปกรณ์ที่สื่อสารด้วย บิตสิ้นสุดข้อมูลนี้ถูกกำหนดให้เป็นลอจิก 1 เสมอ ทั้งนี้เพื่อให้ระบบสามารถตรวจสอบบิตเริ่มต้น จากรูปแบบดังกล่าวจะเห็นว่าเรามีรูปแบบสำหรับการสื่อสารมากมาย เช่น 5E1 (5Data bit, Even Parity, 1 Stop Bit), 7E1 (7Data bit, Even Parity, 1 Stop Bit) และ 8N1 (8Data bit, Non Parity, 1 Stop Bit) เป็นต้น ในการใช้งานทั่วไปเรานิยมใช้อยู่ 2 รูปแบบ คือ 7E1และ 8N1 จะเลือกใช้รูปแบบใด ก็ขึ้นอยู่กับสภาพของสายส่งสัญญาณว่ามีสัญญาณรบกวนมากเพียงใด ถ้าหากสายส่งมีสัญญาณรบกวนมากก็ควรจะใช้ 7E1 แต่ถ้าสายส่งสัญญาณมีสภาพดีสัญญาณรบกวนต่ำการใช้ 8N1 จะเร็วกว่า เป็นต้น

ทั้งนี้จะต้องมีการตกลงกันล่วงหน้าระหว่างผู้รับ และผู้ส่งว่าจะใช้รูปแบบใดในการสื่อสาร ลักษณะของข้อมูลที่ถูกส่งออกไป

ภาพที่ 3.5 แสดงรูปแบบชุดข้อมูลมาตรฐานการสื่อสารอนุกรม

3.3.2 การส่งข้อมูลแบบซิมเพล็กซ์ และแบบดูเพล็กซ์

ในการสื่อสารไม่ว่าจะเป็นการสื่อสารข้อมูลหรือการสื่อสารทั่วไปนั้น ย่อมจะต้องประกอบด้วย ผู้รับ และผู้ส่ง ผู้รับในขณะนี้อาจจะเป็นผู้ส่งในอนาคตก็ได้ แต่มีบางกรณีที่เป็นผู้รับและผู้ส่งแน่นอนตายตัวอยู่ตลอดเวลา เช่น การสื่อสารข้อมูลระหว่างคอมพิวเตอร์ กับเครื่องพิมพ์ เป็นต้น การสื่อสารข้อมูลของอุปกรณ์ที่มี ผู้รับ และผู้ส่งตายตัวนั้น เราเรียกว่าการสื่อสารแบบซิมเพล็กซ์ กล่าวคือ การสื่อสารเป็นลักษณะทิศทางเดียวตลอดเวลาซึ่งจะมีที่ใช้ไม่มากนัก การสื่อสารโดยทั่วไปนั้นจะเป็นแบบดูเพล็กซ์ คือ มีทิศทางการสื่อสาร 2 ทิศทางทั้งไปและกลับ การสื่อสารในลักษณะดูเพล็กซ์นี้ ยังแบ่งออกเป็น 2 ชนิดคือ แบบฮาล์ฟดูเพล็ก (Half Duplex) นิยมเขียนย่อกันว่า HDX ซึ่งจะมีทิศทางการสื่อสารในลักษณะที่ผลัดกันเป็นผู้ส่ง และผู้รับพร้อมกันไป และแบบฟูลดูเพล็กซ์ (Full Duplex) นิยมเขียนย่อว่า FDX จะมีทิศทางการสื่อสารในลักษณะสัญญาณรับทิศทางหนึ่ง สัญญาณส่งอีกทิศทางหนึ่งหรือกล่าวได้อีกนัยว่า สัญญาณรับและส่งจะมีสายตัวนำสัญญาณแยกออกจากกันโดยเด็ดขาด

ภาพที่ 3.2 แสดงการส่งข้อมูลแบบทิศทางเดียว (Simplex)

ภาพที่ 3.3 แสดงการส่งข้อมูลแบบฮาล์ฟดูเพล็ก (Half Duplex)

ภาพที่ 3.4 แสดงการส่งข้อมูลแบบฟูลดูเพล็กซ์ (Full Duplex)

3.3.3 มาตรฐานการสื่อสารข้อมูลอนุกรม RS-232 C

มาตรฐานการสื่อสารข้อมูลแบบอนุกรมที่กำหนดโดย EIA (Electronics Industries Association) มาตรฐาน RS-232 C โดยตัวอักษร RS แทน " Recommend Standard " และ 232 แทนหมายเลขของมาตรฐาน ส่วนอักษร C แสดงให้รู้ว่ามาตรฐานนี้ได้รับการแก้ไขกี่ครั้ง ซึ่ง EIA เป็นสมาคมอุตสาหกรรมอิเล็กทรอนิกส์ของอเมริกา เป็นมาตรฐานอุตสาหกรรมที่ออกแบบมาเพื่อใช้ในการส่งข้อมูลแบบอะซิงโครนัส 2 ทิศทาง โดยแบ่งการเชื่อมต่อออกเป็น 2 ลักษณะคือ DTE (Data Terminal Equipment) และ DCE (Data Communication Equipment) ซึ่งโดยปกติ DTE จะต้องต่อเข้ากับ DCE เสมอ ข้อแตกต่างของอุปกรณ์ DTE และอุปกรณ์ DCE ก็คือคอนเน็คเตอร์ของ DTE จะเป็นตัวผู้ ส่วนของ DCE จะเป็นตัวเมีย ซึ่งพอร์ตอนุกรมของคอมพิวเตอร์ที่ใช้กันอยู่จะเป็นแบบ DTE เช่น การต่อเครื่องคอมพิวเตอร์ (อุปกรณ์ DTE) เข้ากับโมเด็ม (อุปกรณ์ DCE) เป็นต้น

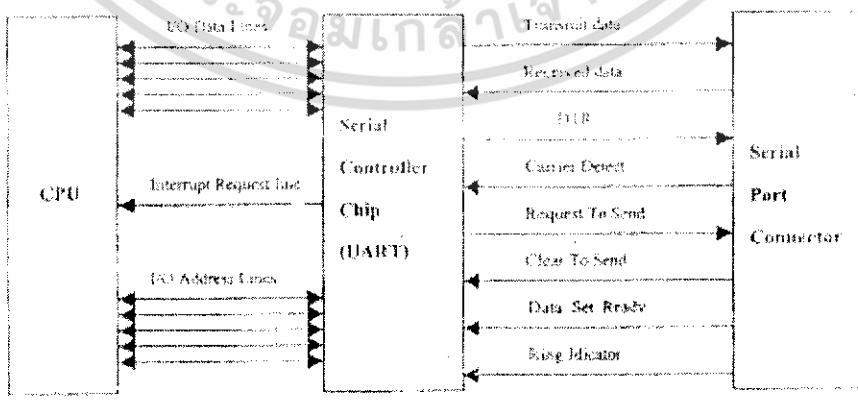
พอร์ตอนุกรม RS-232 จะเป็นพอร์ตของเครื่องคอมพิวเตอร์ ซึ่งเรามักจะเรียกว่าพอร์ต COM1: และ COM2: ในความเป็นจริงพอร์ตอนุกรมไม่ได้ถูกควบคุมจาก CPU บนเมนบอร์ด แต่การสื่อสารทั้งหมดจะถูกจัดการโดยชิพ UART (Universal Asynchronous Receiver/Transmitter) อีกทีหนึ่ง ซึ่งชิพ UART นี้จะทำหน้าที่ในการรับและส่งข้อมูลแบบอะซิงโครนัสนั่นเอง สำหรับการสื่อสารแบบอนุกรมบนคอมพิวเตอร์แล้ว UART ถือว่าเป็นหัวใจสำคัญของการสื่อสารแบบอนุกรม

หน้าที่หลักของ UART คือ แปลงข้อมูลที่อยู่ในรูปแบบขนานจากซีพียู ให้อยู่ในรูปแบบอนุกรมแบบอะซิงโครนัส แล้วทำการส่งออกไป และทำการแปลงสัญญาณอนุกรมแบบอะซิงโครนัสที่ป้อนเข้ามายัง UART ให้เป็นแบบขนานก่อนที่จะส่งเข้าซีพียู ซึ่งนอกจาก UART จะส่งข้อมูลไปยังซีพียูแล้ว ยังแจ้งถึงรายละเอียดอื่นๆ ของข้อมูลให้คอมพิวเตอร์รับทราบอีกด้วย

อาทิ อัตราเร็วในการรับส่งข้อมูลหรือบอดเรต,รูปแบบการส่งข้อมูล,ความผิดพลาดที่เกิดขึ้นระหว่างการส่งข้อมูล เช่น เฟรมข้อมูล, โอเวอร์รัน เป็นต้น

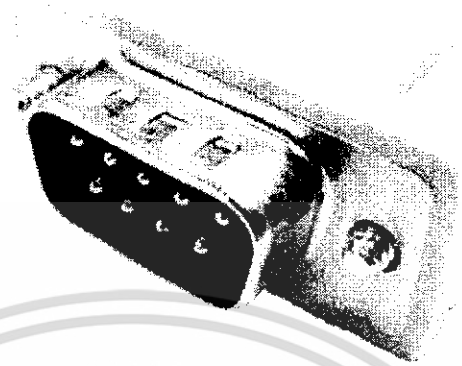
1. การส่งข้อมูล (Data Transmission):
   - รับตัวอักษรจากเครื่องคอมพิวเตอร์
   - แปลงตัวอักษรให้เป็นสายข้อมูลแบบบิต (เรียกว่าขบวนการ Serialization)
   - สร้างเฟรมข้อมูลโดยการเพิ่มบิตที่จำเป็นสำหรับการสื่อสาร และการตรวจสอบ เช่น Start Bit, Stop Bit, และ Parity Bit เป็นต้น
   - ส่งผ่านเฟรมข้อมูลที่สร้างขึ้นมาแล้วจากขั้นตอนที่ผ่านมา ด้วยความเร็วของโมเด็ม หรือพอร์ตอนุกรม (Baud Rate)
   - แสดงสถานะความพร้อมที่จะรับข้อมูลตัวอักษรถัดไปให้กับเครื่องคอมพิวเตอร์
2. การรับข้อมูล (Data Receiver):
   - รับตัวอักษรจากอินเทอร์เฟซ (Interface)
   - ตรวจสอบความถูกต้องของเฟรมข้อมูลตามมาตรฐานที่กำหนด โดยถ้าหากเฟรมข้อมูลมีรูปแบบไม่ถูกต้อง ก็จะมีการแจ้งความผิดพลาดทันที
   - ตรวจสอบความถูกต้องของพาริตีบิต
   - แปลงสายข้อมูลแบบบิตให้เป็นตัวอักษร
   - ส่งตัวอักษรให้กับเครื่องคอมพิวเตอร์
   - แสดงสถานะความพร้อมที่จะรับข้อมูลตัวอักษรตัวต่อไปให้กับอินเทอร์เฟซ

สำหรับการเชื่อมต่อสายสัญญาณต่างๆ ระหว่าง CPU บนเมนบอร์ดของเครื่องคอมพิวเตอร์ กับ พอร์ตอนุกรมนั้นจะต้องกระทำผ่านทางชิพ UART ซึ่งจะมีวิธีการเชื่อมต่อดังรูป



ภาพที่ 3.6 แสดงการเชื่อมต่อระหว่าง CPU ของเครื่องคอมพิวเตอร์ กับ พอร์ตอนุกรม

พอร์ตสื่อสารอนุกรม (Serial Communication) หรือเรามักจะเรียกว่า "Serial Port" ซึ่งเครื่องคอมพิวเตอร์โดยปกติจะมีพอร์ตชนิดนี้อยู่แล้ว 2 พอร์ตคือขนาด 9 ขา และ 25 ขา



ภาพที่ 3.7 แสดงรูปพอร์ตสื่อสารอนุกรมขนาด 9 ขา

ตารางที่ 3.1 แสดงขาสัญญาณต่างๆในการรับส่งข้อมูลแบบอนุกรม RS-232 C Standard

DB-9 pins	DB-25 pins	ชื่อของสายสัญญาณ
1	8	Carrier Detect: DCD
2	3	Received Data: RxD
3	2	Transmitted Data: TxD
4	20	Data Terminal Ready: DTR
5	7	Signal Ground: GND
6	6	Data Set Ready: DSR
7	4	Request to send: RTS
8	5	Clear to send: CTS
9	22	Ring Indictor: RI

ลักษณะสมบัติทางไฟฟ้าของการอินเทอร์เฟซแบบ RS-232 C

- ถูกออกแบบให้ใช้กับอุปกรณ์พวกสัญญาณ Discrete
- ใช้การอินเทอร์เฟซแบบ Unbalanced
- ใช้สายตัวนำในการนำสัญญาณ 1 เส้น และมีสายกราวด์รวมของทุกวงจรอีกหนึ่งเส้น
- อัตราเร็วในการส่งข้อมูลมีค่า < 20 กิโลบิทต่อวินาที (Kbps)
- ระยะทางสูงสุดที่ใช้ในการส่งข้อมูลมีค่า < 15 เมตร
- ทำให้เกิด Crosstalk ที่มีค่ามาก

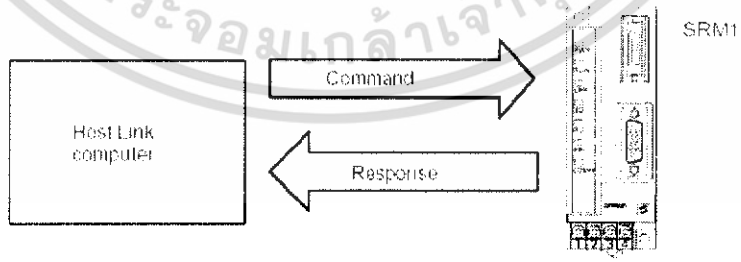
มาตรฐานของการรับส่งข้อมูลแบบอนุกรม RS-232 C นี้ได้กำหนดขึ้นเพื่อให้คอมพิวเตอร์หรืออุปกรณ์ต่อพ่วงแต่ละชนิดรับส่งข้อมูลกันได้เมื่อทำตามมาตรฐานนี้ โดยไม่สนใจว่าอุปกรณ์หรือคอมพิวเตอร์นั้นจะผลิตมาจากที่ใด

3.4 ข้อกำหนดในการสื่อสารระหว่าง PLC กับ คอมพิวเตอร์ (Host Link Communication)

การติดต่อสื่อสารระหว่าง PLC กับคอมพิวเตอร์สามารถทำได้โดย การใช้งานในส่วนของ Host Link Communication ซึ่งถูกพัฒนาโดย OMRON เพื่อใช้ในการติดต่อสื่อสารแบบอนุกรมระหว่าง PLC กับโฮสต์คอมพิวเตอร์ผ่านทางพอร์ตอนุกรมตามมาตรฐาน RS-232 C รูปแบบการจัดเรียงข้อมูลเป็นชุดคำสั่ง (Command Block) และชุดตอบสนอง (Response Block) ที่ใช้ใน Host Link นี้จะเป็นชุดคำสั่งที่ PLC สามารถเข้าใจและทำงานตามได้ โดยกลุ่มของข้อมูลที่ส่งในครั้งหนึ่งๆ เรียกว่า เฟรม (Frame) ซึ่งใน 1 เฟรมสามารถบรรจุตัวอักษรสูงสุด 122 ตัวอักษร

3.4.1 ข้อตกลงในการสื่อสารของเครื่องควบคุม PLC/PC

รูปแบบของข้อตกลงในการสื่อสารของเครื่องควบคุม PLC/PC โดยทั่วไป มีรูปแบบเป็นไบท์โอเรียนโปรโตคอล (Byte-oriented protocol) แบบอะซิงโครนัสโปรโตคอล (Asynchronous protocol) จะเป็นลักษณะการถามตอบกันระหว่างเครื่องควบคุมกับอุปกรณ์ภายนอก ซึ่งอุปกรณ์ภายนอกมักเป็นฝ่ายถามก่อน โดยการส่งบล็อกคำสั่ง(Command Block) ออกไป จากนั้นเครื่องควบคุมจะทำการตรวจสอบแล้วจึงส่งบล็อกตอบสนองกลับมา (Response Block) ในการวิจัยได้นำโปรโตคอลของออมรอนมาเป็นตัวอย่าง สำหรับการสื่อสารข้อมูลกันระหว่างเครื่องคอมพิวเตอร์กับเครื่องควบคุม PLC/PC



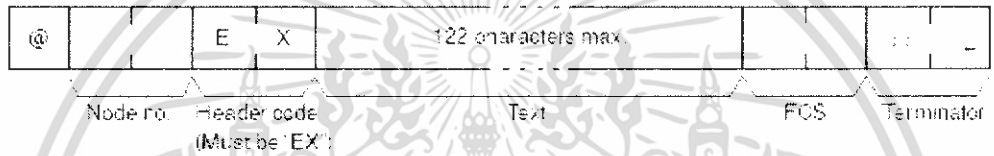
ภาพที่ 3.8 แสดงการติดต่อระหว่างคอมพิวเตอร์กับเครื่องควบคุม PLC/PC

3.4.1.1 รูปแบบชุดคำสั่ง (Command Format)

ในการสื่อสารข้อมูลจะต้องมีกฎหรือข้อกำหนดในการสื่อสารข้อมูล หรือที่เรียกว่า โปรโตคอล (Protocol) ซึ่งจะเป็นส่วนที่กำหนดมาตรฐานในการควบคุมและจัดการระบบการสื่อสารข้อมูล ลักษณะของบล็อกของเครื่องควบคุมPLC/PC แต่ละผู้ผลิตจะแตกต่างกันออกไปแต่จะมีพื้นฐานเดียวกัน

@	หมายเลขเครื่อง	HEADER	TEXT	FCS	*

ภาพที่ 3.9 แสดงรูปแบบของชุดข้อมูล (Block)



ภาพที่ 3.10 แสดงรูปแบบของชุดคำสั่ง

จากรูปแบบของชุดคำสั่งที่ส่งจากเครื่องคอมพิวเตอร์ไปยัง PLC สามารถอธิบายส่วนต่างๆได้ดังนี้

1. หมายเลขเครื่อง (Node No.) ในการเชื่อมต่อที่เป็นเครือข่ายแบบหลายๆ จุดนั้นเครื่องควบคุมที่เชื่อมต่ออยู่ในระบบจะมีมากกว่า 1 เครื่อง ดังนั้นจะต้องมีการกำหนดหมายเลขให้กับเครื่องควบคุม เช่น @03 คือ เครื่องควบคุมหมายเลขที่ 03

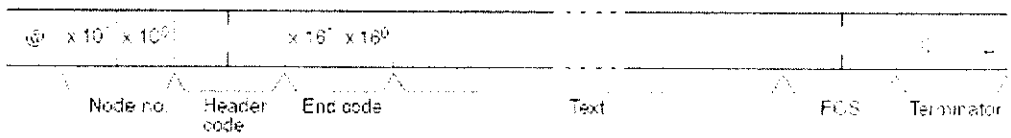
2. ส่วน HEADER COCE เป็นส่วนของคำสั่งหลักที่จะกำหนด ว่าต้องการกระทำกับข้อมูลส่วนใดซึ่งจะเป็นตัวอักษรย่อภาษาอังกฤษเช่น ต้องการอ่านข้อมูลอินพุตจากเครื่องควบคุมจะใช้ตัวอักษรพิมพ์ใหญ่ว่า “RR” หรือถ้าต้องการเขียนข้อมูลให้กับเครื่องควบคุมใช้ “WR” เป็นต้น ซึ่งสามารถดูรายละเอียดในภาคผนวก

3. ส่วน TEXT เป็นส่วนของข้อมูล เช่น ค่าที่อ่านได้จากอินพุต หรือค่าที่ต้องการเขียนลงในพื้นที่ต่างๆ

4. ส่วนของ FCS เป็นส่วนของการควบคุมความผิดพลาดของข้อมูล ซึ่งได้จากการคำนวณ

5. ส่วนของ TERMINAL เป็นส่วนที่ปิดท้ายบอกให้ทราบว่าจบเบล็อก และมักจะติดตามด้วยรหัส Carrier Return (CR)

3.4.1.2 รูปแบบชุดผลตอบสนอง (Response Block)



ภาพที่ 3.11 แสดงรูปแบบของชุดผลตอบสนอง

รูปแบบของชุดผลตอบสนอง (Response Block) ซึ่งส่วนประกอบจะเหมือนกับ Command Block แต่จะแตกต่างกันตรงที่บล็อกผลตอบสนองนั้นจะมีส่วนของ End code ซึ่งใช้แสดงสถานะในการสื่อสารและบอกความผิดพลาดที่เกิดขึ้นในการสื่อสาร ในกรณีที่ไม่มีความผิดพลาดเกิดขึ้น End code จะเท่ากับ 00 แต่ถ้า End code ไม่เท่ากับ 00 แสดงว่าการสื่อสารมีความผิดพลาด ซึ่งสามารถตรวจสอบสาเหตุของความผิดพลาดได้จากตารางแสดงค่า End code กับสาเหตุของความผิดพลาดในภาคผนวก เช่น End code แสดงตัวเลข 18 แสดงว่าเกิดความผิดพลาดในเรื่องขนาดความยาวของข้อมูลเกิน เป็นต้น

ตัวอย่างของรูปแบบชุดคำสั่งและชุดตอบสนอง

@	10	RR	00040003	46	* CR

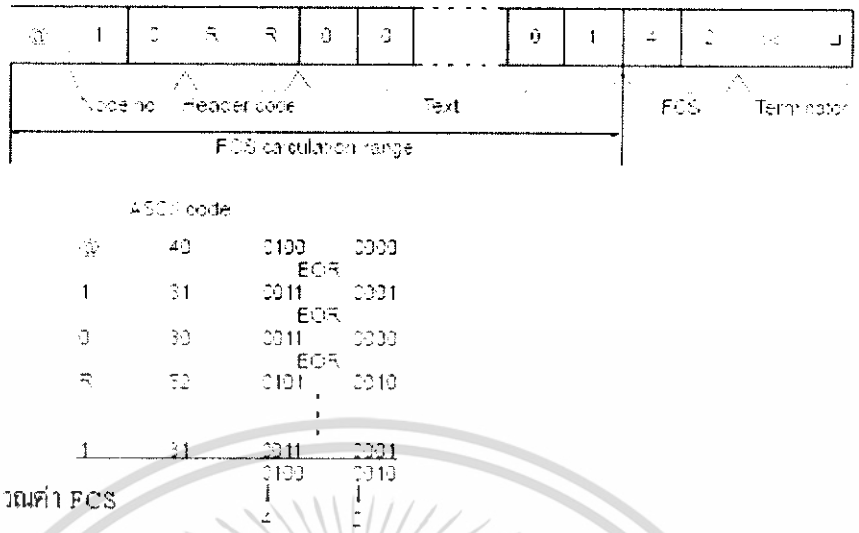
@	10	RR	00	AB59 6324 7831	47	* CR

ภาพที่ 3.12 แสดงตัวอย่างรูปแบบของชุดคำสั่งและชุดตอบสนอง

3.4.1.3 การคำนวณ FCS

FCS หรือ Frame Check Sequence เป็นสิ่งที่ใช้ในการตรวจสอบความผิดพลาดของการสื่อสาร ซึ่งจะทำหน้าที่คล้ายกับ พาริตี้บิตในการสื่อสารแบบอนุกรม ในการรับส่งข้อมูลระหว่างเครื่องคอมพิวเตอร์กับ PLC ทุกครั้ง จะต้องมีการคำนวณ FCS เพื่อใส่ให้กับชุดข้อมูลที่จะส่งออกไป หรือนำ FCS ที่คำนวณได้จากชุดข้อมูลรับเข้ามามาเปรียบเทียบกับค่า FCS ในชุดข้อมูลรับที่เข้ามาว่าตรงกันหรือไม่

หลักการนำข้อมูลในชุดข้อมูล โดยเริ่มจาก @ จนถึงตัวอักษรตัวสุดท้ายของ TEXT ข้อมูล 8 บิต เป็น 2 ตัวอักษร เป็นรหัสข้อมูล ASCII แล้วจึงนำมาทำการ EXCLUSIVE-OR (XOR) กันจะได้ค่า FCS



ภาพที่ 3.13 แสดงการคำนวณหา FCS

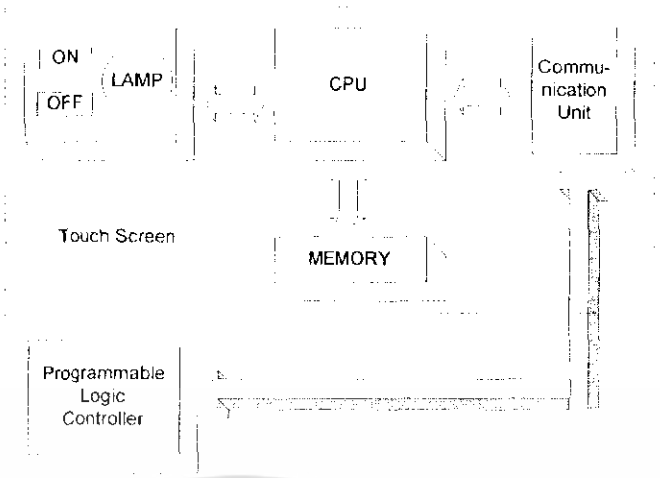
3.5 แผงควบคุมสัมผัสหน้าจอที่โปรแกรมได้ (Touch Screen)

ในการออกแบบระบบควบคุมให้กับเครื่องจักรในงานอุตสาหกรรมโดยทั่วไป จะประกอบด้วยอุปกรณ์ทางด้านอินพุต ได้แก่ ตัวตรวจจับ และตัวส่งสัญญาณ (Sensor and transducer) และ อุปกรณ์ทางด้านเอาท์พุต ได้แก่ รีเลย์ แมคเนติกคอนแทคเตอร์ อินเวอร์เตอร์ มอเตอร์ โซลินอล์ยวาล์ว เป็นต้น เครื่องควบคุมแบบโปรแกรมได้และส่วนที่เชื่อมโยงกับผู้ใช้งาน (Human Machine Interface) ซึ่งเป็นส่วนที่รับและส่งสัญญาณเพื่อแสดงผล และสั่งการควบคุม

แผงควบคุมสัมผัสหน้าจอเป็นอุปกรณ์ที่สามารถโปรแกรมรูปแบบในการแสดงผล และการควบคุมบนหน้าจอได้หลายๆ หน้าจอ โดยทำการออกแบบบนคอมพิวเตอร์และทำงานร่วมกับเครื่องควบคุมแบบโปรแกรมได้ โดยอาศัยการติดต่อสื่อสารผ่านพอร์ตอนุกรมทำให้ลดปัญหาในเรื่องของสายสัญญาณ อีกทั้งถูกออกแบบมาให้ทนต่อสภาพแวดล้อมในโรงงานอุตสาหกรรม

3.5.1 โครงสร้างของแผงควบคุมสัมผัสหน้าจอ

ประกอบด้วยแผ่นฟิล์มพิเศษทับซ้อนกันเมื่อออกแรงกด ก็จะส่งผลให้ความต้านทานของเซลล์ที่ถูกกดทับเปลี่ยนไป ซึ่งจะต้องสอดคล้องกับการโปรแกรมเซลล์ที่ทำหน้าที่เป็น สวิทช์ หลังจากนั้นหน่วยประมวลผลของแผงควบคุมสัมผัสหน้าจอก็จะส่งข้อมูลไปยังหน่วยติดต่อสื่อสารเพื่อเชื่อมโยงการทำงานกับอุปกรณ์ภายนอก ดังนั้นการโปรแกรมบนเครื่องควบคุมจะต้องสอดคล้องกันกับหน้าจอที่ทำการออกแบบด้วย



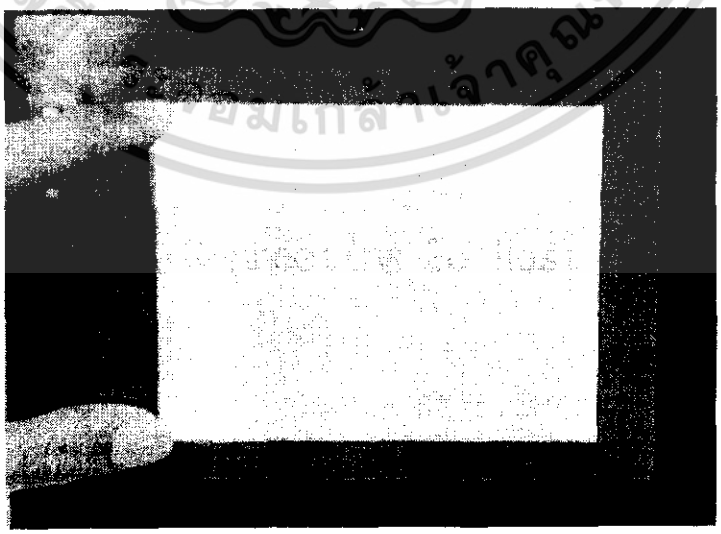
ภาพที่ 3.14 แสดงรูปโครงสร้างของแผงควบคุมสัมผัสหน้าจอ

3.5.2 ขั้นตอนการใช้งาน

- เขียนโปรแกรมเพื่อออกแบบหน้าจอของแผงควบคุมสัมผัสหน้าจอ ด้วยโปรแกรม NT Support Tool บนเครื่องคอมพิวเตอร์
- ส่งข้อมูลของโปรแกรมหน้าจอผ่านพอร์ตอนุกรม RS 232C จากคอมพิวเตอร์ ไปยังแผงควบคุมสัมผัสหน้าจอ
- ทำการเชื่อมต่อระหว่างเครื่องควบคุมที่โปรแกรมได้กับแผงควบคุมสัมผัสหน้าจอ

3.5.3 การตั้งค่าของแผงควบคุมสัมผัสหน้าจอ

การเข้าสู่ระบบโดยทำการ กดที่มุมบนซ้ายและล่างซ้ายของแผงควบคุมสัมผัสหน้าจอพร้อมกันทั้งสองจุด



ภาพที่ 3.15 แสดงรูปการเข้าสู่เมนูระบบ

ในส่วนของเมนูระบบ (System menu) ประกอบไปด้วย ปุ่มกดสัมผัสที่จะเรียกไปยังโหมดการทำงานต่างๆ คือ

1. Transmit Mode จะเป็นการส่งโปรแกรมหน้าจอจากคอมพิวเตอร์ ไปยังแผงควบคุมสัมผัสหน้าจอ หรือการนำโปรแกรมที่มีอยู่ในหน่วยความจำของแผงควบคุมสัมผัสมาทำการแก้ไข

2. Maintenance Mode จะเป็นโหมดของการตั้งค่าต่างๆ เมื่อเรียกเข้ามาก็จะพบปุ่มกดที่ใช้เรียกไปยังเมนูอื่นอีก ในที่นี้จะกล่าวถึงเฉพาะเมนูที่มีความจำเป็นเท่านั้น

- เมนู I/O Check ใช้ในการตรวจสอบสภาพการใช้งาน ของพอร์ตติดต่อสื่อสารและความพร้อมขององค์ประกอบในแผงควบคุมสัมผัสหน้าจอ

- เมนู PT Setting ใช้ดูการตรวจสอบการกำหนดค่า ให้กับพื้นที่ของหน่วยความจำพิเศษ ใน PT Notify area และ PT Control area

- เมนู Init. Memory ใช้สำหรับล้างข้อมูลในหน่วยความจำ ของแผงควบคุมสัมผัสหน้าจอ Memory Switch

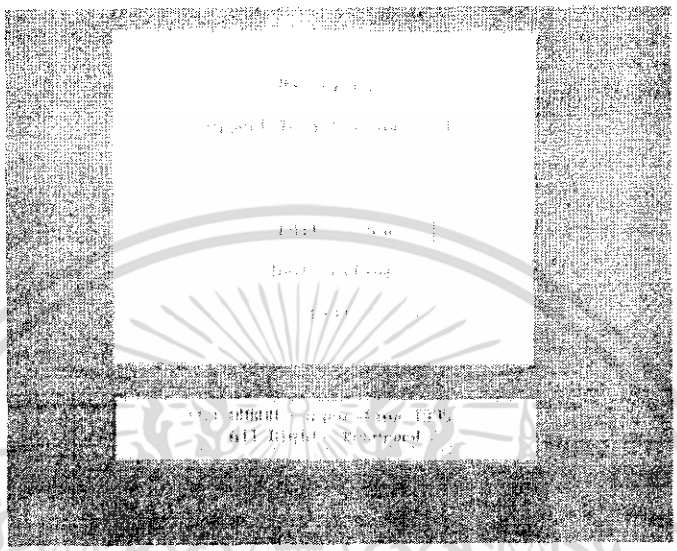
- เมนู Memory Switch ใช้ในการตั้งค่าการใช้งานอุปกรณ์และฟังก์ชั่นต่างๆ

ภาพที่ 3.16 แสดงรูปเมนูระบบ (System Menu)

3.5.4 การใช้งานโปรแกรมออกแบบหน้าจอ

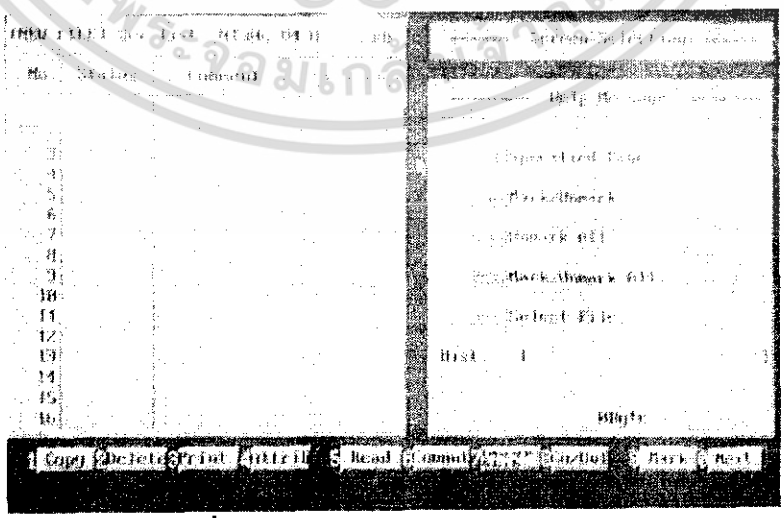
โปรแกรมที่ใช้ในการออกแบบหน้าจอนี้ ได้ถูกออกแบบมาเพื่อใช้งานบนระบบปฏิบัติการ DOS อย่างไรก็ตามจำเป็นต้องเขียนโปรแกรมแลดเดอร์ลงบนเครื่องควบคุมแบบโปรแกรมได้ควบคู่กันไปด้วย เพื่อแสดงถึงความสัมพันธ์กันของทั้งสองส่วน

1. เรียกใช้งานโปรแกรม NT Support tool จากหน้าจอคอมพิวเตอร์ จะปรากฏหน้าต่างหลักซึ่งจะประกอบไปด้วย 2 หน้าที่คือ Edit Screen หน้าที่ใช้ในการออกแบบ และ Tool Setting หน้าที่ใช้ในการกำหนดค่าให้ตรงกับรุ่นของแผงควบคุมสัมผัสหน้าจอ ในหน้านี้จะต้องทำการกำหนดค่าเช่น ชนิด ขนาดหน่วยความจำ พอร์ตอนุกรม ซึ่งรุ่นที่ใช้คือรุ่น NT 30 C



ภาพที่ 3.17 แสดงรูปหน้าหลัก

2. ทำการบันทึกค่าที่ตั้งไ ว้แล้วโปรแกรมจะเรียกกลับสู่หน้าหลัก ต่อไปเลือกหน้า Edit Screen จะพบหน้าของการเลือกแฟ้มข้อมูลหน้าจอ File Selection ถ้าต้องการสร้างใหม่ให้เข้าไปเลือกที่ Creation of New File เพื่อการสร้างหน้าจอใหม่ แต่ถ้าต้องการแก้ไขข้อมูลเดิมก็เข้าไปเลือกที่ File ข้อมูลนั้นได้เลย



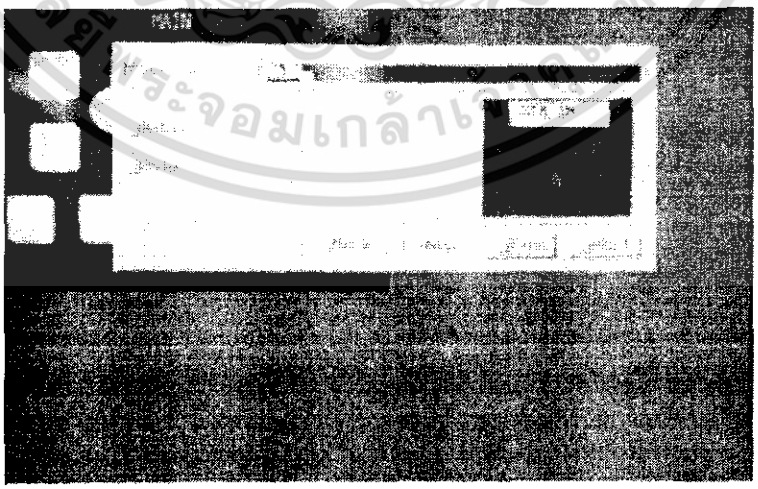
ภาพที่ 3.18 แสดงรูปหน้าจอ File Selection

3. การสร้างแฟ้มข้อมูลใหม่ โดยเลือกที่ Creation of New File จะพบหน้า Screen Selection ซึ่งเป็นหน้าที่ใช้ออกแบบสำหรับแต่ละหน้า ในแต่ละรุ่นจะมีหน่วยความจำในการจัดเก็บหน้าจอได้ไม่เท่ากัน

4. การสร้างหน้าจอโดยเลือกที่ Screen หมายเลข 1 ก็จะปรากฏหน้าจอ Screen Editor เพื่อออกแบบหน้าจอ โดยด้านล่างของหน้าจอนี้จะเป็นคำสั่งต่างๆ ที่ใช้ในการสร้างกราฟิกหน้าจอ ซึ่งมีอยู่หลายคำสั่งด้วยกันดังนี้

ภาพที่ 3.19 แสดงรูปหน้าจอ Screen Editor

- FIX DISP (Fix display) เป็นการสร้างข้อความ หรือรูปภาพประกอบ



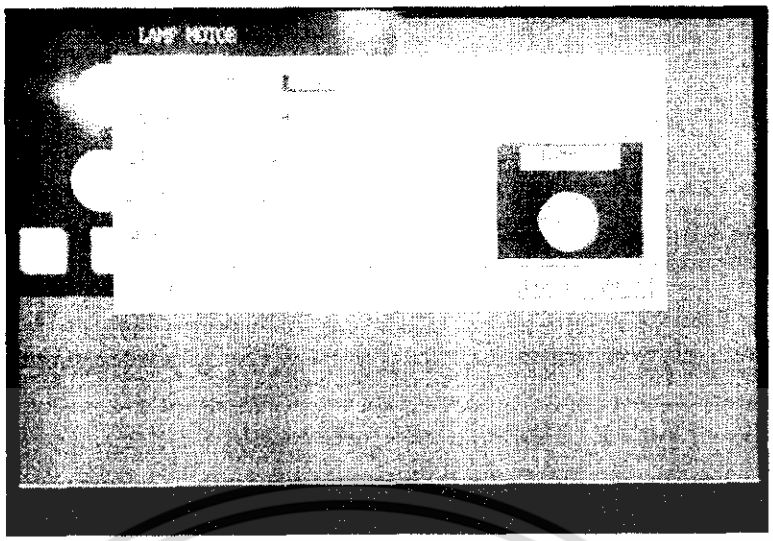
ภาพที่ 3.20 แสดงรูปการสร้างข้อความ

- TOUCH SW (Touch switch) เป็นการสร้างสวิทช์ ซึ่งแต่หน้าจอสามารถสร้างได้หลายตัวโดยเริ่มต้นด้วยการเลือกไปที่ช่องแรกของตาราง TOUCH SW ก็มีแสดงรายละเอียดต่างๆ จากนั้นเลือกประเภทสวิทช์ ซึ่งมีหลายฟังก์ชั่น ได้แก่ Switch Screen ใช้สำหรับการเรียกไปยังหน้าอื่นๆ และ In Notice function ซึ่งเป็นสวิทช์ที่ใช้สำหรับการสั่งงานบิตต่างๆ ซึ่งจะอ้างอิงกับหน่วยความจำระดับบิตบนเครื่องควบคุมแบบโปรแกรมได้

จากนั้นเป็นการใส่ข้อมูลระดับบิตให้กับตัวสวิทช์ และสามารถใส่ข้อความประกอบได้ เปลี่ยนรูปร่างตามแบบที่มีอยู่

ภาพที่ 3.21 แสดงรูปแบบการใส่ข้อมูลให้กับ TOUCH SWITCH

- LAMP เป็นส่วนของเอาท์พุตเพื่อใช้แสดงผลระดับบิต เริ่มต้นโดยการเลือกไปที่ช่องแรกของตาราง LAMP ก็จะแสดงรายละเอียดต่างๆ จากนั้นก็ใส่ข้อมูลระดับบิตให้กับตัวแสดงผล LAMP และ สามารถใส่ข้อความประกอบได้ เปลี่ยนรูปร่างตามแบบที่มีอยู่

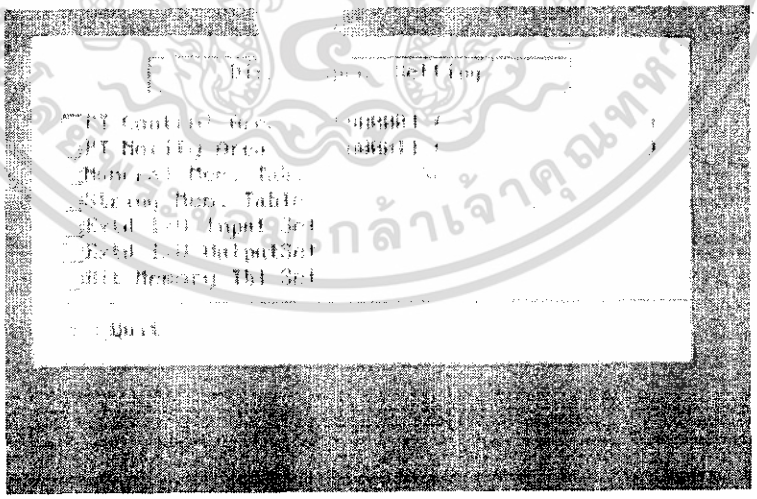


ภาพที่ 3.22 แสดงรูปการใส่ข้อมูลให้กับ LAMP

5. การกำหนดพื้นที่ของ PT Area ซึ่งเป็นพื้นที่หน่วยความจำระดับเวิร์ด ในเครื่องควบคุมที่โปรแกรมได้ สำหรับอ้างอิงหมายเลขหน้าของแผงควบคุมสัมผัสหน้าจอ

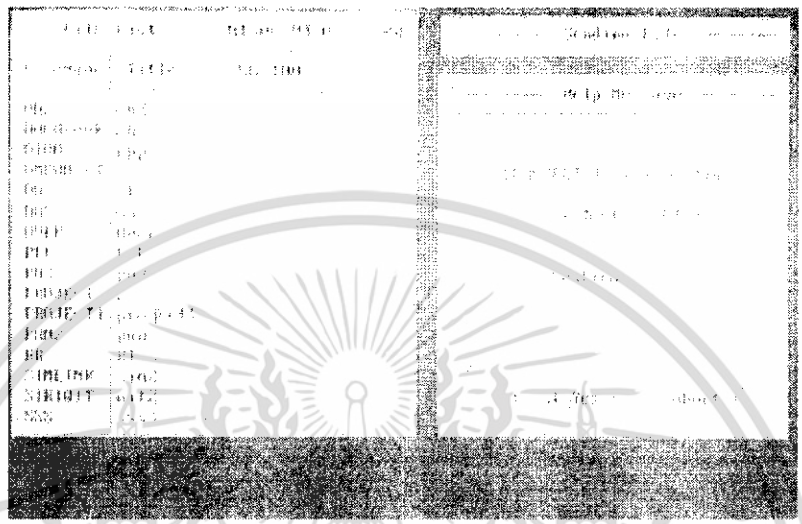
- PT Control Area คือ พื้นที่สำหรับควบคุมการเปลี่ยนหน้า จากเครื่องควบคุมแบบโปรแกรมได้

- PT Notify Area คือ พื้นที่สำหรับแสดงการเปลี่ยนหน้าของแผงควบคุม โดยสามารถดูได้จากเครื่องควบคุมแบบโปรแกรมได้



ภาพที่ 3.23 แสดงรูปการกำหนดพื้นที่ของ PT Area

6. เมื่อเสร็จแล้วให้ทำการบันทึกแฟ้มข้อมูล และตั้งชื่อของงาน การส่งข้อมูลหน้าจอไปสู่แผงควบคุมสัมผัสหน้าจอ ทำการตรวจสอบสายเชื่อมโยงข้อมูลและพอร์ตอนุกรมให้ถูกต้อง ตัวแผงควบคุมสัมผัสหน้าจอจะต้องอยู่ใน Transmit Mode การส่งข้อมูลไปโปรแกรมจะต้องอยู่หน้าหลักของ File Selection กดที่ปุ่ม TMX หรือกดปุ่ม F6 เพื่อส่งข้อมูล



ภาพที่ 3.24 แสดงรูปการเชื่อมโยงข้อมูลผ่านพอร์ตอนุกรม

บทที่ 4
การโปรแกรมด้วยภาษา JAVA

4.1 กล่าวนำ

Java ถูกพัฒนาขึ้นโดยบริษัท Sun Microsystems ซึ่งมีความเป็นมาเริ่มจากในช่วง 1990s บริษัท Sun Microsystems ได้สร้างโปรแกรมภาษาขึ้นมาใหม่ ชื่อว่า " Oak " หลังจากนั้นอีกไม่กี่ปีต่อมา Sun ก็มีความพร้อมที่จะผลักดันให้ Oak ถูกนำมาใช้ในธุรกิจอย่างจริงจังเสียที แต่ชื่อ " Oak" ยังไม่ผ่านการพิจารณาและเมื่อเสนอชื่ออื่นๆ ขึ้นมาก็ยังไม่ผ่าน จนกระทั่งมีผู้เสนอชื่อ จาวา(Java) ซึ่งทุกคนเห็นพ้องต้องกัน และตั้งแต่นั้นมา Java ก็ถูกนำมาใช้เป็นชื่อของโปรแกรมแบบใหม่นี้ทันที โดยที่อีกความหมายหนึ่งก็เป็นชื่อของกาแฟชนิดหนึ่งหรือเป็นชื่อของเกาะ "ชวา" ก็ได้ แต่เนื่องจาก Oak เป็นชื่อภายในที่ถูกนำไปใช้โดยสินค้าอื่นของ Sun แล้ว ดังนั้นชื่อนี้จึงไม่เหมาะสมเท่าที่ควร

ในปี 1995 Oak จึงถูกกำหนดชื่อใหม่เป็น "จาวา (Java)" โดยมีคุณสมบัติที่สำคัญคือเป็นภาษาโปรแกรมที่สามารถนำมาพัฒนา Application บนเว็บได้อย่างเหมาะสม หลังจากนั้นในปีเดียวกัน Sun ก็ได้ออกชุดเครื่องมือสำหรับพัฒนา Java เริ่มแรกขึ้นมาชื่อว่า Java Development Kit (JDK) โดยอนุญาตให้ผู้พัฒนาทั่วโลกสามารถดาวน์โหลดได้ฟรีผ่านทาง Internet ส่งผลให้มีการเริ่มใช้ Java ขึ้นมาอย่างเป็นทางการครั้งแรก

4.2 ข้อดีของ JAVA

ภาษาโปรแกรมอื่นๆ ที่แต่เดิมนั้นไม่ได้เป็นภาษาเชิงวัตถุ แต่ปัจจุบันก็หันมาสนับสนุนลักษณะของเชิงวัตถุ เช่น Visual Basic ที่ได้เพิ่มเติมส่วนของเชิงวัตถุเข้าไปภายในโปรแกรม และโปรแกรมอื่นๆ ก็ได้ปรับปรุงหลักการให้อิงคุณสมบัติทาง Object ให้มากขึ้น แม้แต่ ASP ซึ่งเป็นผู้นำทางธุรกิจซอฟต์แวร์แบบ client/server ก็ได้พัฒนา SAP R/3 Release 4 และ ABAP object ฯลฯ โดยผลิตภัณฑ์ต่างๆ เหล่านี้ของ SAP ก็ใช้หลักการเชิงวัตถุเช่นกัน

ในปัจจุบันจัดว่า Java เป็นโปรแกรมภาษาที่มีความเป็นศิลปะ (state – of – the – art programming language) เพราะ Java มีลักษณะที่น่าสนใจมากมาย เช่น Simple, เชิงวัตถุ, Distributed, Robust, Secure, Architecture, Neutral, Portable, Interpreted, High Performance และ Multithreading ซึ่งมีรายละเอียดดังต่อไปนี้

- **Simple** ภาษา Java กำจัดส่วนที่มีความซับซ้อนในภาษา C++ ออกไป เช่น multiple inheritance, automatic type conversion, pointer และ memory management ของ C++ ทำให้ภาษา Java ใช้งานง่าย และมีความซับซ้อนน้อยลงกว่า C++ มาก

- **OOP** ส่วน C++ เป็นภาษาผสม คือเขียนได้ทั้งแบบ Procedural และ OOP ที่สนับสนุนหลักการเชิงวัตถุ แต่ Java จะบังคับให้เขียนโปรแกรมได้ด้วยหลักการเชิงวัตถุเท่านั้น จึงถือได้ว่า Java เป็นภาษาที่สามารถแสดงลักษณะเชิงวัตถุได้อย่างชัดเจน เช่นเดียวกับ ภาษา Smalltalk

- **Distributed** ในด้าน network ภาษา Java จะสนับสนุน TCP/IP protocol เช่น HTTP, FTP ฯลฯ รวมทั้งมีส่วนในการสนับสนุนการเข้าถึง URL ด้วย

- **Robust** ภาษา Java เป็นภาษาที่มีความคงทน เนื่องจากภาษา Java มีระบบป้องกันและสามารถกำจัดความผิดเล็กๆ น้อย ๆ ที่อาจจะเกิดขึ้นในระบบ โดยไม่ต้องหยุดการทำงานของโปรแกรมด้วยสาเหตุเพียงเล็กน้อยเหล่านี้ นอกจากนี้ Java ยังไม่สนับสนุนการทำงานโดยใช้ pointer เพราะ pointer เป็นสาเหตุที่ทำให้เกิดความผิดพลาดได้ง่าย (pointer มีใช้ในภาษา C++)

- **Secure** ภาษา Java ได้รับการพัฒนาให้เป็น ภาษาสำหรับการดำเนินงานบนระบบเครือข่าย (Network) ดังนั้น ภาษา Java จึงได้รับการสร้างมาให้ยึดหลักความปลอดภัยของข้อมูลบนระบบ network เป็นสำคัญ

- **Architecture Neutral** ภาษา Java สามารถทำงานได้ บนทุกๆ ระบบ (Platform) เนื่องจาก Java จะมีการแปลภาษา 2 ขั้นตอน คือการ compile และ interpret โดยเมื่อภาษา Java ได้รับการ compile ออกมาเป็น ภาษาของ JVM แล้ว (Java Virtual Machine หรือ Byte code โดยเรียกว่า Byte code เพราะทุกคำสั่งของ JVM มีขนาดเพียง 1 byte เท่านั้น) จะสามารถนำโค้ดที่ได้มาไปทำงานบนเครื่องใดๆ ที่มีชุดคำสั่งของ JVM ติดตั้งอยู่ ดังนั้น Java จึงเป็นภาษาที่ไม่ขึ้นกับระบบใดๆ

- **Portable** ในภาษา C และ C++ จะทำงานขึ้นกับระบบควบคุม (OS) ซึ่งต่างกับภาษา Java ที่ไม่ยึดติดกับระบบควบคุมใดๆ ยกตัวอย่างเช่น int ใน C และ C++ ต้องการหน่วยความจำ 16 bits, 32 bits หรือขนาดอื่นๆ ขึ้นกับระบบปฏิบัติการ แต่สำหรับภาษา Java จะใช้หน่วยความจำ 32 bits เสมอ

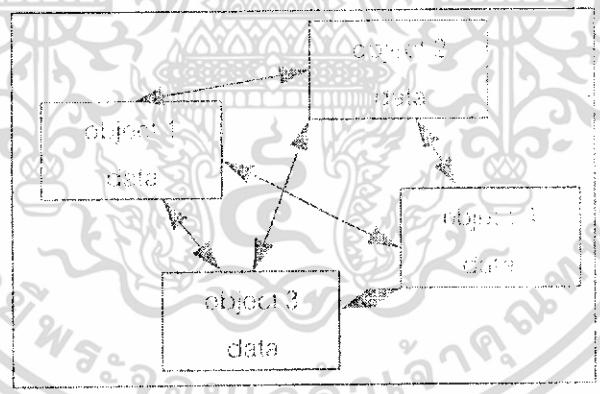
- **High – Performance** ในภาษา Java จะอนุญาต ให้เขียนโค้ด Java เพื่อเรียกใช้งานโปรแกรมภาษาอื่นได้ด้วย เช่น ภาษา C++ นอกจากนี้ในการประมวลผลด้วย JVM ไปเป็นชุดคำสั่งที่สามารถทำงานได้กับเครื่องคอมพิวเตอร์ใดๆ นั้น จะใช้เวลาในการประมวลผลน้อยกว่าเวลาที่ใช้ในการ compile ภาษาอื่น

- **Multithreading** เป็นการอนุญาตให้โปรแกรมมากกว่าหนึ่งโปรแกรม สามารถทำงาน ณ เวลาเดียวกันได้ ทำให้ระบบทำงานแบบ real – time ได้มากขึ้น ซึ่งในภาษาโปรแกรมอื่นๆ การทำ Multithreading นั้นจะมีความซับซ้อนมาก แต่ในภาษา Java จะสามารถจัดการกับ

Multithreading ได้อย่างง่ายดายภาษา Java ได้รับการออกแบบมาให้มีความใกล้เคียงกับภาษา C++ แต่จะได้เปรียบมากกว่า เนื่องจากภาษา C++ จะไม่มีข้อดีในส่วนที่ Java มี และเหตุผลที่ภาษา Javaได้รับความนิยมอยู่ในขณะนี้ ก็อาจจะเป็นไปได้ว่าเนื่องมาจากความนิยมของ C และ C++ ด้วย เพราะผู้ที่เป็น Programmer ทางด้าน C และ C++ จะสามารถใช้ภาษา Java ได้อย่างง่ายดาย โดยไม่จำเป็นต้องทำการศึกษารูปแบบการเขียนโปรแกรมใหม่มากมายนัก

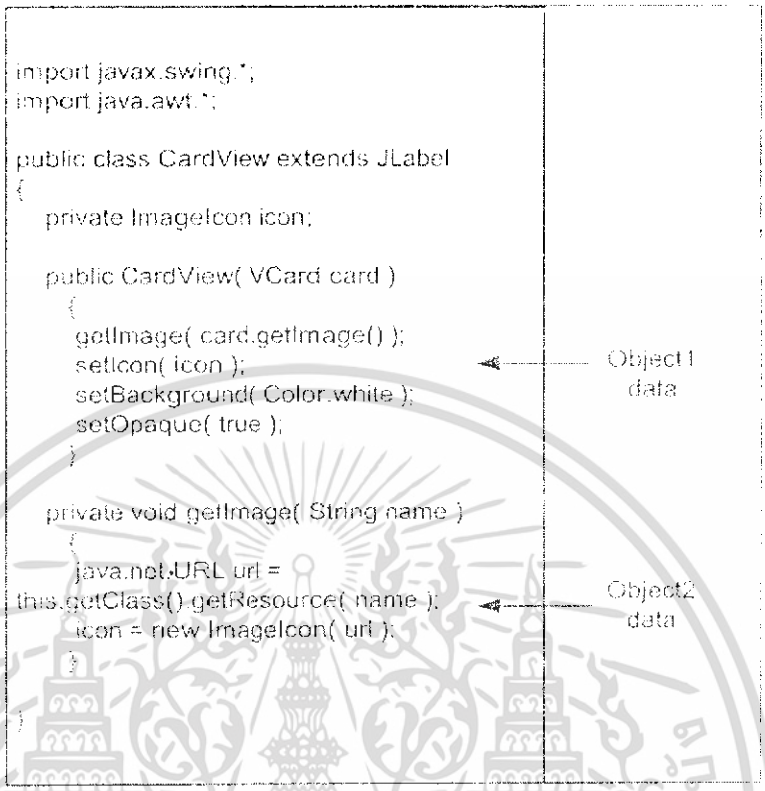
4.3 Object – Oriented - Programming (การโปรแกรมเชิงวัตถุ - OOP)

การโปรแกรมเชิงวัตถุเป็นวิธีการเขียนโปรแกรมแบบใหม่ ซึ่งจะเรียกโปรแกรมย่อยที่ทำหน้าที่เฉพาะว่า Object (Component) และยังได้รวม data เข้าเป็นส่วนของ Object ด้วย นอกจากนี้โปรแกรมเชิงวัตถุยังมีโครงสร้างที่พิเศษ นอกเหนือจาก procedure programming หรือ module programming คือการทำงานของแต่ละ Object จะไม่สิ้นสุดลง เมื่อกระบวนการทำงานของโปรแกรมจบการทำงานบน Object นั้นแล้ว ซึ่ง Object ที่อยู่ในโปรแกรมเหล่านี้จะพร้อมเสมอในการเริ่มทำงานต่อจากตำแหน่งเดิมเมื่อเกิดการเรียกใช้งาน Object อีกครั้ง ซึ่งแตกต่างกับการเขียนโปรแกรมในแบบอื่นๆ ที่โปรแกรมย่อยจะต้องเริ่มการทำงานใหม่เมื่อถูกเรียกใช้อีกครั้ง ดังนั้นเทคนิคนี้จึงช่วยลดปัญหาที่จะเกิดขึ้นจากการเขียนโปรแกรมด้วยวิธีอื่นๆ ได้



ภาพที่ 4.1 แสดง Object - Oriented - Programming โดยแต่ละ Object สามารถติดต่อกันได้โดยการส่ง message ไปที่ Object อื่น

ตัวอย่างโปรแกรม

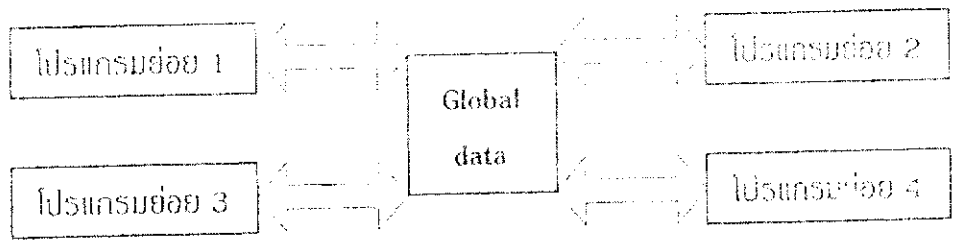


ภาพที่ 4.2 แสดงตัวอย่างโปรแกรม

ความแตกต่างระหว่าง Procedural Programming และ OOP

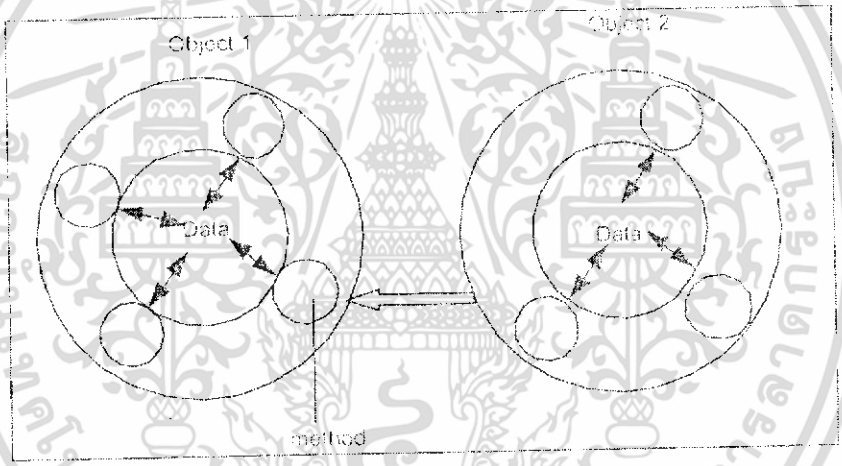
วิธีการเขียนโปรแกรมที่กล่าวมาข้างต้น สามารถแบ่งออกได้เป็น 2 แนวคิดหรือ 2 วิธีการในการพัฒนาซอฟต์แวร์ได้แก่ วิธีการทาง Procedural Programming และวิธีการเชิงวัตถุ programming ซึ่งมีความแตกต่างกันดังนี้

โครงสร้างของโปรแกรมแนวคิดแบบ Procedural Programming ภายในโครงสร้างของโปรแกรมจะแบ่งเป็น 2 ส่วนคือ ส่วนของโปรแกรมหลัก (main program) ที่มี data เป็นส่วนประกอบ และส่วนของโปรแกรมย่อย (procedures หรือ function) โดย data ที่ประกาศใช้อยู่ภายในโปรแกรมหลักนั้น จะถูกเรียกโดยโปรแกรมย่อยต่างๆ ที่อยู่ภายในโปรแกรม ลักษณะของ data ที่มีการประกาศใช้ทั่วทั้งโปรแกรมนั้นเรียกว่า เป็นการประกาศใช้แบบ global ดังรูป



ภาพที่ 4.3 แสดง global

แนวคิดแบบ OOP ดังรูป data จะได้รับการประกาศใช้เฉพาะภายใน object แต่ละobject เท่านั้น (จะไม่ประกาศเป็น global) ซึ่ง object อื่นสามารถเข้าถึง data ได้ โดยผ่าน method ของ object ที่เป็นเจ้าของ data เท่านั้น



ภาพที่ 4.4 แสดง object to object

จะเห็นได้ว่าโครงสร้างตามแนวคิดเชิงวัตถุ จะดีกว่าแบบ Procedural Programming เนื่องจากการประกาศใช้ data แบบ global จะมีผลเสียต่อการควบคุมการเปลี่ยนแปลง data ของโปรแกรม ซึ่งยากต่อการแก้ไขโปรแกรมในภายหลัง

ช่วงชีวิต (Life Time)

ใน Procedural Programming ช่วงชีวิตของโปรแกรมย่อยนั้น จะเท่ากับช่วงเวลาการทำงานของโปรแกรมที่ผ่านเข้ามาและออกจากโปรแกรมย่อย ดังนั้นเมื่อโปรแกรมหลักดำเนินงานผ่านโปรแกรมย่อยไปแล้วค่าที่อยู่ในโปรแกรมย่อยนั้นจะกลับคืนสู่สภาพเดิมเหมือนขณะโปรแกรมย่อยอยู่ในสภาพเริ่มต้น

ใน OOP ช่วงชีวิตของ object จะเริ่มตั้งแต่โปรแกรมเริ่มทำงาน จนกระทั่งสิ้นสุดการทำงานหรือเมื่อ object ถูกตัดขาดออกจากโปรแกรม ซึ่ง object จะถูกตัดขาดจากโปรแกรมเมื่อไม่มี object นั้นอีก (ไม่มีคำสั่งจาก object ใดๆ ที่เรียกใช้ object นั้นอีก)

4.4 Java 2 SDK

การใช้ Java และสร้างโปรแกรมจาก Java จำเป็นต้องใช้ชุด kit ในการพัฒนา ดังนั้น Sun จึงได้เตรียมชุด kit ไว้ให้กับผู้พัฒนาโปรแกรม คือ "Java 2 Platform System Development Kit (J2SDK)" มีลักษณะการทำงานในแบบ Command Line ซึ่งทำในดอส นอกจากนี้ยังมีอีกหลายบริษัทที่ได้พัฒนาเครื่องมือสำหรับสร้าง Java Application ที่มีการทำงานในแบบกราฟิก เพื่อให้ใช้งานร่วมกับ J2SDK ได้ง่าย เช่น โปรแกรม JBuilder หรือ โปรแกรม IBM Visual Age for Java เป็นต้น

เวอร์ชันแรกของ J2SDK คือ J2SDK V1.0 ตามมาด้วยเวอร์ชัน 1.0.2 ซึ่งเพิ่มคุณสมบัติต่างๆจาก v1.0 หลายอย่างหลังจากนั้น จึงออก J2SDK v1.1 ตามมาด้วย v1.18 และ v1.2 , v1.2.2 เรื่อยมา จนมาถึง J2SDK v1.4.2 ในปัจจุบัน (เดือนกันยายน พ.ศ.2546)

สำหรับขนาดของ J2SDK v1.4.1 จะมีขนาด 35.9 Mb ใน SDK จะมีเครื่องมือและ Library มาตรฐานของ Java ที่ใช้สร้าง Applet และ Application การเขียนโปรแกรมแบบ Server – sides หรือ Servlets การใช้ J2SE Platform อย่างเดียวไม่เพียงพอจะต้องดาวน์โหลด Java Platform อื่นเพิ่มเติม คือ Java 2 Enterprise Edition (J2EE) ถ้าต้องการพัฒนาโปรแกรมเกี่ยวกับ Middlets หรือ Wireless ซึ่งเป็นการทำงานเกี่ยวกับการติดต่อแบบไร้สาย จะต้องใช้ Java 2 Micro Edition (J2ME)

ภายใน SDK จะมีคอมไพเลอร์ (Compiler) ที่ใช้สำหรับแปล Source Code ให้เป็นไฟล์ Byte Code ที่สามารถทำงานได้ทั้งแบบที่เป็น Java Application และ Java Applet ในส่วนของ SDK นอกจากจะมีเครื่องมือที่ใช้ในการ Compile และ Interpret แล้ว ยังมี JRE และกลุ่มของ Libraries ต่างๆ ที่ใช้ในโปรแกรมด้วยทั้ง 2 ส่วนนี้จะเป็นตัวกำหนด Platform ของ Java ซึ่งส่วนมากการเปลี่ยนเวอร์ชันของ Java จะเปลี่ยนที่องค์ประกอบของ Libraries ต่างๆ เหล่านี้

เครื่องมือของ Java SDK ภายใน SDK จะมีเครื่องมือทั้งหมด 23 ชุด แต่ที่จะกล่าวถึงในขั้นนี้มี 8 ชุด ได้แก่

ตารางที่ 4.1 แสดงเครื่องมือของ Java SDK

เครื่องมือ	หน้าที่
javac	คำสั่งของคอมไพเลอร์ที่ใช้เปลี่ยนจาก Source Code เป็น Byte Code
java	คำสั่งของตัวแปลภาษาที่สั่งให้ Application ทำงาน โดยการอ่านจาก Byte Code สำหรับใช้ใน Java Application
appletviewer	ใช้สำหรับสั่งให้ Java Applet ทำงานเพื่อทดสอบก่อนการนำไปใช้งานจริงผ่านบราวเซอร์
javadoc	เป็นเครื่องมือที่ใช้อธิบายโปรแกรม Java โดยสร้างเอกสาร HTML จากโปรแกรมJava
jdb	เป็นตัวดีบักเกอร์ของ Java ที่ใช้สำหรับตรวจสอบการทำงานของโปรแกรม โดยสั่งให้โปรแกรมหยุดการทำงานตามขั้นตอนที่โปรแกรมเมอร์กำหนด เพื่อตรวจสอบตัวแปร และดูลำดับในการทำงานของชุดคำสั่ง
javap	เป็นตัว Java Disasambler ซึ่งใช้ทดสอบ Byte Code ที่เกิดขึ้นแล้วแสดงข้อมูล (Method หรือ Function) ของ Byte Code นั้น ถูกเรียกใช้ในกรณีที่มีปัญหาในเรื่องของการเรียกประมวลผลโปรแกรม
jar	คือ ตัวจัดการเกี่ยวกับไฟล์ Jar Archive (JAR) โดยการรวมและบีบอัดไฟล์หลายๆ ไฟล์ รวมถึงระดับไดเร็กทอรี่ให้เป็นไฟล์เดียว เพื่อความสะดวกในการใช้งาน
javah	คือ ตัวสร้างไฟล์ Header โดยจะถูกใช้เมื่อมีการทำงานร่วมกับโปรแกรม C หรือC++ ทำให้เรียกใช้ไฟล์ Java ได้
rmic	ใช้ร่วมกับ Remote Method Invocation (RMI) เพื่อสร้างคลาสในหารประมวลผลแบบกระจาย

4.5 Compilation และ Interpretation

ในการเขียนโปรแกรมโดยทั่วไปจะต้องมีกระบวนการและเครื่องมือสำหรับสร้างโปรแกรมขึ้นมา ซึ่งเครื่องมือพื้นฐานที่ใช้ในโปรแกรมทั่วไป ได้แก่ Editor, Compiler และ Interpreter โดยหลังจากสร้างโปรแกรมขึ้นมาแล้ว การนำโปรแกรมที่ได้ไปใช้งานจะต้องผ่านวิธีการคอมไพล์ (Compilation) หรือแปลภาษา (Interpretation) อย่างใดอย่างหนึ่ง เพื่อนำโปรแกรมนั้นไปแปลเป็นภาษาที่เครื่องเข้าใจ (Machine Language) ได้ ซึ่งแต่ละวิธีก็มีข้อดีและข้อเสียต่างกันออกไปดังนี้

- วิธีการคอมไพล์ (Compilation)

เป็นการสร้าง Executable Code หรือโค้ดที่นำไปใช้งานได้ จาก Source Code โดยสิ่งที่เรียกว่า " คอมไพเลอร์ (Compiler)" วิธีนี้เกิดจากการที่คอมไพเลอร์จะทำการแปลโปรแกรมทีเดียวทั้งโปรแกรมซึ่งจะได้ Executable Code หรือ ไฟล์ .exe ขึ้นมา ดังนั้นคอมไพเลอร์จึงมีโอกาสวิเคราะห์ได้ทั้งโปรแกรม ทำให้การแปลภาษามีประสิทธิภาพ รวดเร็ว และมีขนาดไม่ใหญ่เกินไป ส่วนข้อเสียคือ ขาดความยืดหยุ่น เนื่องจากการแปลภาษาด้วยวิธีการแปลเพียงครั้งเดียวนี้จะได้ผลผลิตออกมาเป็นไฟล์ .exe ของโปรแกรมทั้งโปรแกรม ดังนั้นในระหว่างที่โปรแกรมทำงานจึงไม่สามารถเปลี่ยนแปลงส่วนใดของโค้ดได้เลย

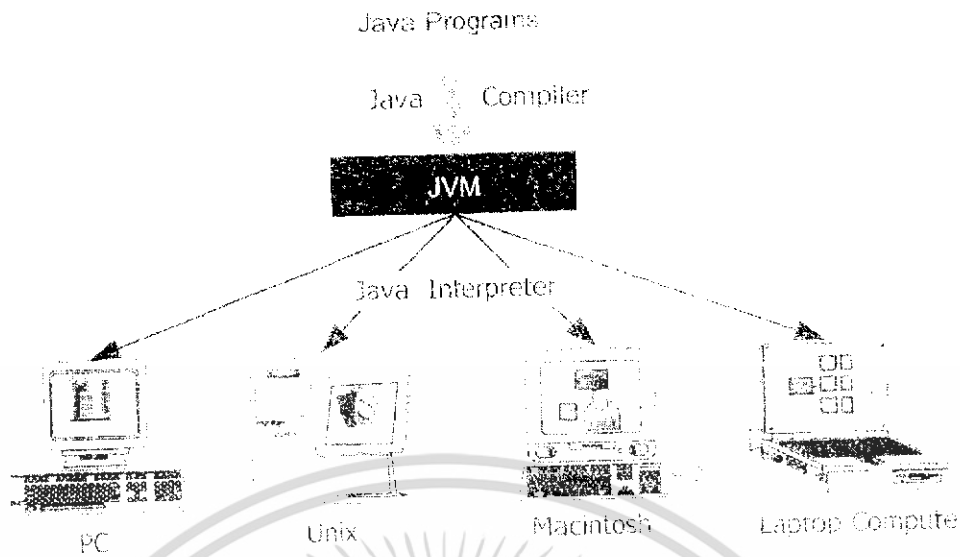
- วิธีการแปลภาษา (Interpretation)

วิธีนี้ใช้เครื่องมือที่เรียกว่า " ตัวแปลภาษา (Interpreter) " แทนคอมไพเลอร์เป็นวิธีที่แตกต่างจากวิธีการคอมไพล์คือ ใช้การอ่านและแปลงโค้ดทีละบรรทัดแทน โดยเมื่อแปลงบรรทัดหนึ่งเสร็จแล้วจึงค่อยกลับมาอ่านบรรทัดต่อไปและแปลงโค้ด ทำอย่างนี้ไปเรื่อยๆ จนจบโปรแกรม จากจุดนี้จะพบว่าวิธีนี้มีข้อดีคือ ทำได้ง่ายกว่าการคอมไพล์ทีเดียวทั้งโปรแกรมและมีความยืดหยุ่นในการเขียนโปรแกรมมากกว่าวิธีการคอมไพล์ เนื่องจากในขณะแปลถ้าเกิด Error ที่จุดใดตัวแปลภาษาก็จะฟ้องและอนุญาตให้ผู้พัฒนาสามารถทำการแก้ไขได้ทันที จากนั้นจึงทำงานในบรรทัดต่อไป แต่มีข้อเสียคือ เนื่องจากต้องทำทีละบรรทัดจึงส่งผลให้ทำงานได้ช้ากว่าวิธีการคอมไพล์

โดยทั่วไปภาษาโปรแกรมจะเลือกใช้วิธีการใดวิธีการหนึ่ง เช่น ภาษา C, Pascal ใช้วิธีการคอมไพล์ ส่วน Basic, smalltalk ใช้วิธีการแปลภาษาซึ่งรูปแบบการทำงานเช่นนี้ จะทำให้เกิดผลเสียคือ โปรแกรมขาดความสามารถในการนำไปใช้กับระบบหรือเครื่องที่ต่าง Platform กัน โดยหากต้องการนำโปรแกรมหนึ่งไปใช้กับเครื่องอื่นๆ ก็ต้องแก้ไข Source Code ใหม่ทำให้เสียเวลาเป็นอย่างมาก

4.6 Java Virtual Machine (JVM)

จากปัญหาที่เกิดขึ้นกับความต้องการที่สามารถนำโค้ดไปใช้งานได้อย่างไม่มีข้อจำกัด ไม่ว่าจะเป็นเครื่องใด หรือระบบปฏิบัติการแบบใดก็ตาม จึงได้มีการพัฒนาทำให้เกิดแนวความคิดของ Java Virtual Machine (JVM) ขึ้นมามีลักษณะดังรูป



ภาพที่ 4.5 แสดง Java Virtual Machine

JVM (Java Virtual Machine) เป็นกลไกเสมือน ซึ่งสร้างขึ้นโดยตัวแปลภาษา (Interpreter) ของ Java โดยมีขั้นตอนการทำงาน คือ เริ่มแรกนำ Source Code ที่อยู่ในรูปแบบของโปรแกรม Java มาผ่านการคอมไพล์โดยตัวคอมไพเลอร์ที่อยู่ในเครื่องที่สร้างโปรแกรมของ Java ซึ่งจะได้ผลลัพธ์คือ โปรแกรมที่ JVM สามารถเข้าใจได้ (ไฟล์ Byte Code) หลังจากนั้นจึงทำการเรียกใช้ตัวแปลภาษา (Interpreter) เพื่อสั่งให้โปรแกรมทำงาน โดยในระหว่างแปลภาษาตัวนี้ Interpreter ก็จะสร้าง JVM ขึ้นมา เพื่อนำโค้ดที่ได้จากการคอมไพล์มาเข้ากระบวนการที่สร้างขึ้น ซึ่งตัว Interpreter ไม่จำเป็นต้องอยู่ในเครื่องเดียวกับ Source Code ที่สร้างขึ้นก็ได้ โดยในการใช้งานถ้าต้องการเรียกใช้งานโปรแกรม Java ในเครื่องใดๆ ก็จะสามารถนำตัว Interpreter ไปติดตั้งไว้ที่เครื่องที่ต้องการได้ทันที โดยไม่ขึ้นกับชนิดของเครื่องหรือระบบปฏิบัติการใดๆ ดังนั้นภาษา Java จึงทำงานได้เร็วและไม่ขึ้นกับระบบ เพราะว่าระบบทั่วๆไปจะลงโปรแกรม Java Interpreter ไว้ และการคอมไพล์ถูกแยกออกจากการ Execution นอกจากนี้การออกแบบคำสั่งของ JVM จะมีความใกล้เคียงกับของหน่วยประมวลผลทั่วไป ซึ่งทำให้ขั้นตอนการทำ Interpretation ทำได้ง่าย และรวดเร็วอีกด้วย

4.7 Java Runtime Environment (JRE)

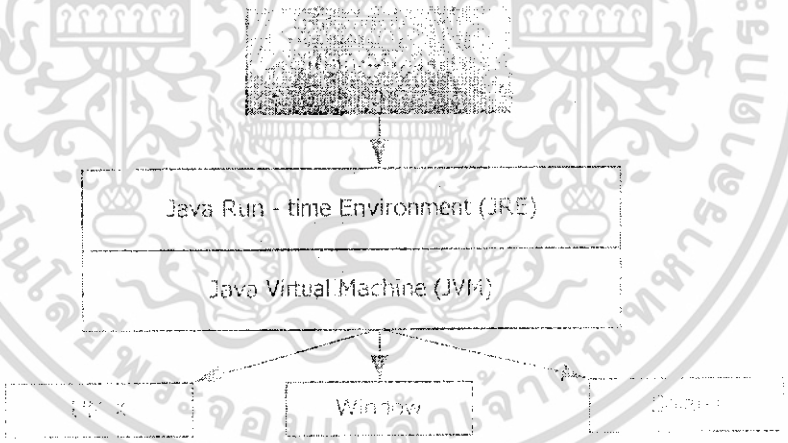
Java ไม่สามารถทำงานตามลำพังได้แต่จะต้องอาศัยสภาวะแวดล้อมที่สนับสนุนการทำงานด้านต่างๆ เพิ่มเติม เช่น การทำงานของ Java แบบ Applet ที่ต้องอาศัยเว็บบราวเซอร์ ในการแสดงผล เป็นต้น

Java ได้จัดเตรียมองค์ประกอบต่างๆ ที่ใช้สำหรับสนับสนุนการทำงานของโปรแกรมที่สร้างขึ้นอยู่ใน "Java Runtime Environment (JRE)"

JRE ประกอบด้วยไฟล์ที่จำเป็นต้องใช้ในการทำงานของ Java เช่น ไฟล์ที่บรรจุ Package ต่างๆ ไฟล์ใช้สำหรับแสดงรูปแบบตัวอักษร เป็นต้น โดยไฟล์ต่างๆ เหล่านี้จะถูกติดตั้งพร้อมกับ J2SDK ตั้งแต่ v.1.1 ในการพัฒนาโปรแกรมด้วยภาษา Java

JRE ถือว่าเป็นหัวใจสำคัญที่ขาดไม่ได้ โดยผู้ที่จะนำโปรแกรมที่พัฒนาขึ้นด้วยภาษา Java ไปทำงานได้นั้น ควรติดตั้ง JRE เวอร์ชันเดียวกันหรือสูงกว่า JRE ของผู้ที่สร้างโปรแกรมนั้นขึ้นมา

ลักษณะการทำงานของ JRE



ภาพที่ 4.6 แสดงลำดับการทำงานของการ Run โปรแกรม Java

จากรูปแสดงลำดับการทำงานของสภาวะแวดล้อมของ Java อย่างคร่าวๆ โดยในขั้นแรกโปรแกรมที่สร้างขึ้นจาก Java จะต้องผ่านกระบวนการของ JRE ก่อน ซึ่งเป็นการนำองค์ประกอบที่จำเป็นสำหรับเรียกโปรแกรมทำงานมาใช้ จากนั้นจึงเข้าไปทำงานที่ส่วนของ JVM ซึ่งเป็นการสร้างกลไกเสมือนที่มีการทำงานดังที่อธิบายไว้ในหัวข้อที่ผ่านมา สำหรับเครื่องที่ใช้แสดงผลอาจจะใช้ระบบปฏิบัติการ UNIX, Windows หรือ Macintosh ก็ได้

4.8 Java Class Libraries

โปรแกรม Java ประกอบขึ้นด้วยส่วนต่างๆ ที่เรียกว่า " คลาส (Class) " ภายในคลาสประกอบด้วยส่วนการทำงานที่เรียกว่า "เมธอด (Method)" ซึ่งจะทำหน้าที่แสดงผลของการทำงานและส่งข้อมูลกลับเมื่อทำงานนั้นๆ เสร็จ ***ในการสร้างโปรแกรมจะใช้วิธีเขียนคลาสแยกออกเป็น 1 ไฟล์ต่างหาก หรือเขียนทุกคลาสอยู่ในไฟล์เดียวกันก็ได้*** แต่โปรแกรมเมอร์ส่วนใหญ่นิยมเขียนให้ทุกคลาสเก็บอยู่ในไฟล์เดียวกันเพื่อความสะดวกในการคอมไพล์และทดสอบโปรแกรม นอกจากนี้ Java ยังมีคลาสต่างๆ ให้โปรแกรมสามารถเรียกใช้งานได้ทันทีไม่จำเป็นต้องเขียนเอง โดยคลาสเหล่านี้จะถูกจัดเป็นกลุ่มๆ ภายใน Java Class Libraries ซึ่งกลุ่มของ Class Libraries เหล่านี้ เรียกว่า "Java APIs (Application Programming Interfaces)" ในการเรียนรู้โปรแกรม Java ทั้ง 2 แบบนี้ โดยแบบแรกคือ การเรียนรู้การสร้างคลาสด้วยตนเองเพื่อให้โปรแกรมสามารถทำงานได้ตามที่ต้องการ ส่วนแบบที่ 2 คือ การเรียนรู้การนำคลาสต่างๆ ที่อยู่ใน Libraries ของโปรแกรม Java มาใช้งานให้ตรงตามที่ต้องการ ซึ่ง Libraries เหล่านี้ถูกจัดเตรียมมาให้พร้อมกับการลง JDK แล้ว

4.9 การสร้างโปรแกรม Java แบบ Application และ Applet

โปรแกรม Java โดยเริ่มแรกสร้างได้ 2 แบบคือ

- **แบบที่ 1** Java Applet เป็นโปรแกรม Java ที่ใช้ทำงานร่วมกับบราวเซอร์และภาษาHTMI. ซึ่งเป็นสภาวะแวดล้อมในการทำงานบนเว็บเพจ
- **แบบที่ 2** Java Application สามารถเรียกใช้งานได้ทันที โดยไม่ต้องใช้เว็บบราวเซอร์

การสร้างโปรแกรม Java แบบ Application มีขั้นตอนดังนี้

1. ในขั้นแรกจะต้องมีเครื่องมือสำหรับการเขียน และ Run โปรแกรม ดังนี้

- **Editor Tools** หรือ เครื่องมือที่ใช้สำหรับเขียนโปรแกรม เช่น Notepad, Editplus เป็นต้น ปัจจุบันเครื่องมือที่ใช้สำหรับเขียนโปรแกรม Java มีอยู่มากมายซึ่งมีอุปกรณ์ช่วยเหลือต่างๆ รวมทั้ง J2SDK ที่มีมาพร้อมกับตัวโปรแกรมไม่ต้องดาวน์โหลดเพิ่มเติม

- **J2SDK (Java 2 Software Development Kit)** เป็นเครื่องมือที่มีความจำเป็นสำหรับการเขียนโปรแกรม Java เป็นอย่างมาก เนื่องจากภายในได้บรรจุเครื่องมือที่จำเป็นสำหรับการเขียนโปรแกรมไว้ เช่น JVM, JRE, คอมไพเลอร์ (Compiler), และตัวแปลภาษา (Interpreter) เป็นต้น

2. ติดตั้ง J2SDK และ Set ค่าต่าง ๆ ให้เรียบร้อย

3. เริ่มต้นเขียนโปรแกรมโดยใช้ Editor และบันทึกไฟล์ โดยใช้นามสกุล .java ซึ่งไฟล์ที่ได้นี้ จะใช้คำสั่ง javadoc.exe จะทำให้ได้ไฟล์ HTML ที่อธิบายส่วนประกอบต่างๆ ของไฟล์ Java นั้น

4. คอมไพล์โปรแกรมโดยใช้คำสั่ง javac.exe จะได้ไฟล์ที่มีนามสกุล .class

5. หลังจากได้ไฟล์ .class แล้วอาจจะใช้ javah.exe เพื่อสร้างไฟล์ Header ของ C/C++ ก็ได้

6. เมื่อต้องการให้โปรแกรมทำงานจึงเรียกใช้ java.exe เพื่อ Run โปรแกรม ซึ่งเครื่องที่ใช้ Run โปรแกรมนี้อาจจะเป็นระบบหรือ Platform ที่ต่างจากเครื่องที่ใช้สร้าง Code ก็ได้ ถ้าติดตั้ง JVM ไว้

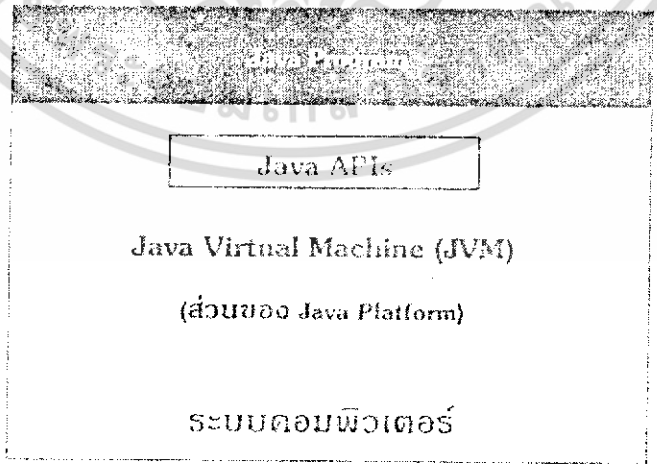
7. ในขณะที่โปรแกรมกำลังทำงานอาจจะเรียกใช้ jdb.exe เพื่อทำการแก้ไขโปรแกรมได้

4.10 Java Platform

Platform คือ สภาวะแวดล้อมการทำงานบนฮาร์ดแวร์และบนระบบปฏิบัติการ (Operating System) ที่เฉพาะเจาะจงซึ่งมีความเกี่ยวข้องกับการทำงานของแอพพลิเคชั่น โดย Platform ที่รู้จักกันดี ได้แก่ Window 2000, Linux, Solaris และ Mac Os Platform ซึ่ง Platform ต่างๆ เหล่านี้เป็นการทำงานร่วมกันของทั้งฮาร์ดแวร์และระบบปฏิบัติการ แต่ Platform ของ Java นั้นจะไม่ขึ้นอยู่กับทั้งฮาร์ดแวร์และระบบปฏิบัติการเลย ทำให้โปรแกรมที่พัฒนาด้วยภาษา Java สามารถทำงานอยู่บน Platform อื่นๆ ได้

Platform ของ Java มี 2 ส่วนได้แก่

- Java Programming Interface (API)
- Java Virtual Machine (JVM)



ภาพที่ 4.7 แสดง Java Platform

Platform ของ Java ยึดหลักการทำงานในส่วนของ Network และแนวความคิดที่ว่าซอฟต์แวร์โดยทั่วไป ควรจะทำงานกับเครื่องคอมพิวเตอร์ที่ต่าง Platform กันได้เป็นสำคัญ จึงทำให้ ***โปรแกรมที่ถูกเขียนด้วยภาษา Java สามารถทำงานได้แม้ Run บนระบบที่ต่าง Platform กันก็ตาม*** สิ่งที่ทำให้ Java มีความสามารถเช่นนี้ก็คือ JVM กล่าวคือ แต่ละ Platform จะมี JVM สำหรับ Platform นั้นๆ อยู่แล้ว เมื่อโปรแกรมถูกเขียนขึ้นมาและผ่านการคอมไพล์ แล้วนำไป Run (Interpret) บนเครื่องที่ติดตั้ง JVM อยู่ก็จะสามารถใช้งานโปรแกรม Java ได้นั่นเอง

สำหรับ Platform ต่างๆ ในภาษา Java มีดังนี้

- Java 2 Platform, Standard Edition (J2SE)
- Java 2 Platform, Enterprise Edition (J2EE)
- Java 2 Platform, Micro Edition (J2ME)

Java Platform ที่เป็นพื้นฐานของการสร้างโปรแกรม Java ก็คือ J2SE

4.10.1 Java 2 Platform, Standard Edition (J2SE)

J2SE Platform คือ เครื่องมือมาตรฐานเริ่มต้นสำหรับโปรแกรมเมอร์ที่ทำงานในฝั่ง Client มีมาพร้อมกับชุดติดตั้งของ J2SDK โดย J2SE นี้ถูกสร้างขึ้นมาจากพื้นฐานของการทำงานที่สำคัญในแง่ของความเร็ว ความปลอดภัย และมีหน้าที่เตรียมเครื่องมือสำหรับจัดการหน้าที่ต่างๆ ให้เหมาะสมในส่วนที่เป็นโปรแกรมในฝั่งของ Client สำหรับผู้ใช้ทั่วไป นอกจากนี้ J2SE ยังช่วยให้การใช้งาน Application บนเว็บเพจ เมื่อมีการร้องขอจากผู้ใช้ทำได้อย่างง่ายและรวดเร็วอีกด้วย

4.10.2 Java 2 Platform, Enterprise Edition (J2EE)

J2EE เป็น Java Platform ที่เป็นเครื่องมือมาตรฐานสำหรับสร้าง Application แบบ Multitier ซึ่งเป็นรูปแบบของ Application ประเภทเครือข่ายแบบหนึ่ง สิ่งที่ J2EE จัดเตรียมไว้ให้นั้นมีข้อได้เปรียบกว่า J2SE มากมาย โดยเฉพาะในเรื่องของการทำงานแบบเครือข่าย เช่น การเพิ่มส่วนสนับสนุนอย่างเต็มที่ให้กับองค์ประกอบต่างๆ ของ Java Beans , Java Servlets APIs และอื่นๆ ที่เกี่ยวข้อง

4.10.3 Java 2 Platform, Micro Edition (J2ME)

J2ME ถูกสร้างขึ้นเพื่อใช้ในการพัฒนา Application ในเทคโนโลยีแบบไร้สาย เช่น Palm, Screen Phone เป็นต้น โดย J2ME จะจัดเตรียมเครื่องมือสำหรับการคอมไพล์ การแก้ไข และ APIs ต่างๆ ที่นำไปใช้ในอุปกรณ์แต่ละชนิด นับว่าเป็น Platform สำหรับยุคใหม่ของคอมพิวเตอร์

4.11 ชนิดของโปรแกรม Java

ปัจจุบันโปรแกรม Java แบ่งออกเป็น 2 ชนิดคือ Java Application (หรือเรียกสั้นๆ ว่า "Application") และ Java Applet (หรือเรียกสั้นๆ ว่า "Applet") ซึ่งทั้ง 2 ชนิดนำไปใช้งานแตกต่างกันดังนี้

- **Java Application** เป็นการนำ Java มาเขียนเป็นโปรแกรมที่สามารถนำมาใช้งานได้อย่างอิสระ (Stand Alone Program) เหมือนกับการเขียนโปรแกรมภาษาระดับสูงอื่นๆ เช่น C++, Pascal, Cobol เป็นต้น นอกจากนี้ยังสามารถนำ Application ที่พัฒนาด้วยภาษา Java ไปใช้งานกับคอมพิวเตอร์ต่างแพลตฟอร์ม (Platform) กันได้ ไม่ว่าจะเป็น PC, Macintosh หรืออื่นๆ โดยไม่ต้องคอมไพล์หรือเขียนโปรแกรมใหม่

- **Java Applet** เป็นการนำ Java มาเขียนเป็นโปรแกรมเช่นเดียวกัน แต่ไม่สามารถเรียกใช้ตามลำพังเหมือนกับ Application ได้ จะต้องนำไปใส่ไว้ในเอกสาร HTML (เพื่อทำให้ Web Page ทำงานได้ดียิ่งขึ้น) แล้วใช้โปรแกรม Web Browser (เช่น Netscape, Internet Explorer) หรือใช้ Utilities ของ Java ชื่อ Applet Viewer เพื่อดูผลลัพธ์

แม้ไม่ว่าจะเป็น Application หรือ Applet ก็ตาม จะต้องมีการแปลงโค้ดโปรแกรมต้นฉบับที่เรียกว่า "Source Code" หรือ "Source Program" ไปเป็นรหัสที่คอมพิวเตอร์เข้าใจก่อน (ซึ่งเรียกว่า "Machine Code" หรือ "Object Code") ขั้นตอนในการแปลงเป็นดังนี้

1. สร้าง Source Code หรือ Source Program ขึ้นมาก่อน เมื่อสร้างเสร็จแล้วให้บันทึกนามสกุลเป็น .java

2. นำไฟล์ที่ได้จากข้อ 1 มาแปลงเป็น "Byte Code" ด้วยวิธีการที่เรียกว่า "คอมไพล์ (Compile)" เมื่อผ่านการคอมไพล์แล้วจะได้ไฟล์ใหม่มา 1 ไฟล์ โดยมีชื่อเดิมแต่มีนามสกุลเป็น .class

ถ้าเป็น Application

3. นำไฟล์ที่ได้จากข้อ 2. มาเรียกใช้ ขั้นตอนนี้จะเรียกว่า "อินเตอร์เพร็ต (Interpret)" โดยจะเรียกใช้ที่คอมพิวเตอร์ต่างแพลตฟอร์ม (Platform) ได้ ไม่ว่าจะเป็น PC, Macintosh หรืออื่นๆ

ถ้าเป็น Applet

4. สร้างเอกสาร HTML ขึ้นมา

5. นำชื่อไฟล์ที่ได้จากข้อ 2 ใส่ลงไปในเอกสาร HTML โดยใช้คำสั่งให้ถูกต้อง

6. เรียกดูผลลัพธ์ผ่านทาง Web Browser หรือ Applet Viewer

4.12 โครงสร้างของ Application

ในการเขียนโปรแกรมด้วยภาษา Java เพื่อสร้าง Application นั้นจะแบ่งโปรแกรมการทำงานออกเป็นคลาส (Class) โดยภายในหนึ่งโปรแกรมจะต้องมีคลาสอย่างน้อยหนึ่งคลาสเสมอ ผู้เขียนโปรแกรมสามารถกำหนดรูปแบบการเขียนโปรแกรมได้อย่างอิสระ ซึ่งอาจเขียนในรูปแบบที่แตกต่างกันไป เช่น เขียนการทำงานทั้งหมดไว้ในคลาสเดียวโดยไม่มีการสร้างออบเจ็กต์เลย หรือจะแบ่งการทำงานออกเป็นหลายๆ คลาส แล้วสร้างออบเจ็กต์เพื่อเรียกใช้เมธอดในการเข้าถึงคุณสมบัติบางตัว และในการเขียนคลาสแต่ละคลาสนั้น บางทีอาจเขียนให้ 1 คลาสอยู่ใน 1 ไฟล์ หรือบางทีอาจเขียนหลายๆ คลาสอยู่ในไฟล์เดียวกันก็ได้ เป็นต้น

หมายเหตุ

- **โปรแกรม (Program)** หมายถึง ชุดคำสั่งที่เขียนเรียงกันเป็นลำดับขั้นตอนเพื่อสั่งให้คอมพิวเตอร์ทำงานตามที่ต้องการ

- **แอปพลิเคชั่น (Application)** หมายถึง โปรแกรมที่สร้างขึ้นมาเพื่อใช้งานเฉพาะด้าน ซึ่งภายในแอปพลิเคชั่นหนึ่งๆอาจจะประกอบด้วยโปรแกรมหลายๆ โปรแกรมประสานการทำงานเข้าด้วยกันเพื่อให้บรรลุวัตถุประสงค์ของงานเฉพาะด้านนั้นๆ สำหรับโครงสร้างการเขียนโปรแกรมสามารถเขียนได้ 2 รูปแบบดังนี้

- **รูปแบบที่ 1** การเขียนโปรแกรมสำหรับภาษา Java นั้น จะเขียนโปรแกรมเป็นคลาส ซึ่งแต่ละคลาสจะประกอบไปด้วยเมธอด (Method) ตัวแปรและคำสั่งต่างๆ ที่จำเป็นต่อการใช้งานภายในโปรแกรมหนึ่งอาจจะประกอบด้วยคลาสมากกว่า 1 คลาสก็ได้ ขึ้นอยู่กับผู้ออกแบบโปรแกรม และเมื่อต้องการใช้งานคลาสที่สร้างขึ้นมาจะต้องสร้างออบเจ็กต์จากคลาสดังกล่าวขึ้นมาก่อน แล้วจึงค่อยนำออบเจ็กต์ที่สร้างขึ้นไปใช้งานตามที่ต้องการ

- **รูปแบบที่ 2** เป็นการเขียนโปรแกรมในรูปแบบเก่า โดยจะรวมการทำงานทั้งหมดไว้ภายในไฟล์เดียวกัน รวมถึงไม่มีการสร้างออบเจ็กต์ขึ้นมาใช้งานภายในโปรแกรมเลย (1 โปรแกรมมี 1 คลาส) สำหรับในภาษา Java สามารถเขียนโปรแกรมแบบไม่ต้องสร้างออบเจ็กต์ก็ได้ แต่ไม่เป็นที่นิยมเนื่องจากความยืดหยุ่นของโปรแกรมมีน้อยกว่าการเขียนแบบสร้างออบเจ็กต์

ข้อควรจำ

1. ภายใน 1 โปรแกรมสามารถมีคลาสได้มากกว่า 1 คลาส ซึ่งจะเขียนให้ทุกคลาสอยู่ภายใน 1 ไฟล์ หรือจะเขียนแยกให้แต่ละคลาสอยู่ต่างไฟล์กันก็ได้

2. หากภายใน 1 ไฟล์มีมากกว่า 1 คลาส สามารถเรียงคลาสใดอยู่ก่อน คลาสใดอยู่หลังก็ได้

3. ภายในคลาสสามารถที่จะวางเมธอดไว้ตรงไหนก็ได้ โดยไม่มีข้อจำกัดว่าจะต้องวางเมธอดใดไว้ก่อนเมธอดใดไว้หลัง แต่จะวางเมธอดไว้นอกคลาสไม่ได้เด็ดขาด

4. ชื่อของคลาส เมธอด และตัวแปร จะต้องระวังเรื่อง ตัวพิมพ์เล็ก และตัวพิมพ์ใหญ่ด้วย เพราะตัวแปลภาษา Java จะตีความว่าเป็นคนละตัวกัน เช่น someApp จะไม่เหมือนกับ SomeApp

5. ภายในคลาสหนึ่ง จะต้องมีเครื่องหมายปีกกาเปิด และปิด "{ }" เพื่อบอกให้รู้ว่าเป็นจุดเริ่มต้นและสิ้นสุดของคลาส เมธอด และลูปคำสั่งต่างๆ เช่น ลูป "do...while" เป็นต้น และวงเล็บปีกกานี้จะวางอยู่บรรทัดเดียวกับชื่อคลาส ชื่อเมธอด หรือคำสั่งก็ได้หรือจะอยู่ต่างบรรทัดกันก็ได้

6. แต่ละคำสั่งที่ใช้ภายในคลาส (ยกเว้นบรรทัด Header ของคลาสและเมธอด) จะต้องปิดท้ายด้วยเครื่องหมาย Semicolon "; " เสมอ เช่น theApp.run();

7. ในการประกาศเมธอด "main ()" ควรวางไว้ที่ด้านบนสุด หรือด้านล่างสุดของโปรแกรม เพื่อให้สังเกตได้ง่าย เพราะทุกโปรแกรมจะต้องเริ่มทำงานที่เมธอด "main ()" ก่อนเสมอ

8. เมื่อต้องการเรียกใช้ (Run) โปรแกรม จะต้องเรียกใช้ชื่อคลาสที่มีเมธอด "main ()" อยู่เท่านั้น

4.13 การสร้างและเรียกใช้โปรแกรม

ในการสร้างโปรแกรมด้วยภาษา Java ชนิด Application สามารถที่จะสร้างได้ทั้งสองแบบ คือ แบบสร้างออบเจ็กต์ และไม่สร้างออบเจ็กต์ ขั้นตอนในการสร้าง Java Application มีดังนี้

1. ป้อนรายละเอียดต่างๆ ของโปรแกรม หรือที่เรียกว่า"SourceCode"ลงไป โดยใช้โปรแกรม Editor ตัวใดก็ได้ จากนั้นบันทึกไฟล์นี้เก็บไว้โดยกำหนดนามสกุลเป็น .java ไฟล์นี้เรียกว่า "Source File" เช่น Hello.java เป็นต้น

2. นำ Source File มาคอมไพล์ (Compile) ที่ DOS - Prompt โดยใช้คำสั่ง "javac" ซึ่งเป็นคอมไพเลอร์ (Compiler) ของภาษา Java แล้วตามด้วยชื่อ Source File ที่กำหนดไว้ในข้อ 1 ถ้าไม่มีข้อผิดพลาดใดๆ เกิดขึ้นในโปรแกรม (Syntax Error) จะได้ผลลัพธ์เป็นไฟล์ใหม่เพิ่มมาอีก 1 ไฟล์ โดยจะมีนามสกุลเป็น .class ทั้งนี้ไฟล์ที่ได้จะอยู่ในรูปของ "Byte Code" (ไบต์โค้ด)

```
D:\Java Code>javac Hello.java
```

จากคำสั่งข้างต้น แยกออกได้เป็น

- **D:\Java Code>** คือ DOS Prompt
- **javac** คือ คำสั่ง "javac"
- **Hello.java** คือ Source File

3. รันโปรแกรมโดยใช้คำสั่ง "java" ซึ่งเป็น Interpreter ของภาษา Java แล้วตามด้วยชื่อไฟล์ผลลัพธ์ในข้อ 2 แต่ไม่ต้องใส่นามสกุล .class ลงไปด้วย มิฉะนั้นจะเกิด Error ขึ้น

```
D:\Java Code>java Hello
Hello World
```

จากคำสั่งข้างต้น แยกออกได้เป็น

- **D:\Java Code>** คือ DOS Prompt
- **Java** คือ คำสั่ง java
- **Hello** คือ Class File
- **Hello World** คือ ผลลัพธ์

4.14 GUI

ในการใช้งานโปรแกรมทั่วไป เรามักจะเห็นหน้าตาของโปรแกรมที่พยายามแสดงจอภาพต่างๆให้เราเข้าใจขั้นตอนการทำงานต่างๆ ได้อย่างชัดเจน เช่น การป้อนข้อมูล ก็จะมีช่องว่างให้กรอกข้อมูลได้ไม่เกินความยาวที่มีให้ และถ้าใส่ข้อมูลที่ไม่ตรงกับที่กำหนด (เช่น ให้ป้อนตัวเลข แต่ใส่เป็นตัวอักษรมา) ก็จะแสดงจอภาพ (dialogbox) มาแจ้งแก่ผู้ใช้ว่า ป้อนข้อมูลผิดพลาด ให้ป้อนใหม่จนกว่าจะถูกต้อง เมื่อป้อนเสร็จก็อาจจะคลิกปุ่มๆ หนึ่งเพื่อใช้ทำงานยังขั้นตอนต่อไปลักษณะจอภาพที่เห็นนี้เรียกว่า "User Interface" คือ User สามารถติดต่อกับโปรแกรมได้ โดยผ่านทางจอภาพ

แต่ในปัจจุบันเทคโนโลยีได้ก้าวไปไกลมาก การพัฒนาด้าน User Interface ก็มีมากขึ้นจนเราสามารถนำเอา graphic มาช่วยในการออกแบบจอภาพให้สวยงาม และสื่อสารกับ user ได้อย่างมีประสิทธิภาพมากขึ้น (เข้าใจได้ง่ายยิ่งขึ้น) เช่น การแสดงด้วย Icon รูปภาพ หรืออื่นๆ ที่เหมือนจริงมากยิ่งขึ้นลักษณะเช่นนี้ เรียกว่า "Graphic User Interface" หรือ GUI

ดังนั้นในการเขียนโปรแกรม GUI (GUI programming) จึงเป็นการเขียนโปรแกรมที่พยายามให้มนุษย์ (ผู้ใช้) สื่อสาร และทำงานกับโปรแกรมได้อย่างมีประสิทธิภาพมากยิ่งขึ้นด้วยการเข้าใจ interface หน้าที่ ขั้นตอนการทำงานและข้อความต่างๆ ของ interface ได้ง่ายขึ้น

4.15 AWT

AWT หรือ Abstract Windowing Toolkit เป็นเครื่องมือพื้นฐานในการเขียนโปรแกรมด้วย GUI ใน Java โดยในการเขียนโปรแกรมด้วย GUI นั้น จะต้องประกอบไปด้วย Components, Containers และ Layout Managers

- AWT Components

AWT Components เป็น Class ที่ Java ได้สร้างมาให้ซึ่งสามารถใช้งานได้เลย (building block) ประกอบไปด้วย class ต่อไปนี้ Component, Container, Button, Canvas, CheckBox, Choice, Dialog, Frame, Label, List, Menu, MenuBar, MenuItem, Panel, Scrollbar, TextArea, Text Component และ Text field ซึ่ง Class ต่างๆ นี้จัดเก็บอยู่ใน Package "java.awt"

- AWT Containers

Containers จัดเป็น Component อย่างหนึ่งที่สามารถนำ Component อื่นๆ มาใส่เอาไว้ได้ (ใช้จัดเก็บ Components) ประกอบไปด้วย class ต่อไปนี้ Applet, Container, Dialog, File Dialog, Frame, Panel, ScrollPane และ Window

- Layout Managers

ใช้สำหรับจัดแสดง Container ออกทางจอภาพ (ใช้จัดเรียง Containers) ประกอบไปด้วย class ต่อไปนี้ Border Layout, Flow Layout, Card Layout, Grid Layout และ Gridbag Layout

4.16 Swing

เนื่องจาก AWT เมื่อนำมาใช้ในการเขียน GUI program นั้น จะมีปัญหาบางอย่างเกิดขึ้น เช่น สิ้นเปลืองหน่วยความจำของคอมพิวเตอร์ เป็นต้น ทำให้มีการพัฒนา package ชื่อ "swing" ขึ้นมาเพื่อช่วยสนับสนุนการทำงานให้ดีขึ้น แต่ไม่ได้นำมาใช้งานแทน AWT เพราะยังคงใช้ AWT อยู่และ Swing เองก็ยังคงต้อง extends บาง class จาก AWT มาใช้งานด้วย

Swing เป็น API (Application Programming Interface) ตัวหนึ่งใน Java Foundation class (JFC) เป็นส่วนหนึ่งของ JDK โดย API ใน JFC ประกอบด้วย Accessibility, Drag – and - Drop, Java2D, Swing และ Input Method Framework

ดังนั้น AWT และ Swing จะใช้งานร่วมกัน โดยนำสิ่งที่ดีของทั้ง 2 package มาใช้สำหรับ Component ของ AWT เมื่อเทียบเท่ากับ Swing แสดงได้ดังตารางต่อไปนี้

ตารางที่ 4.2 แสดงความสัมพันธ์ ระหว่าง AWT และ Swing

AWT Component	Swing JFC
Applet	JApplet
Component	JComponent
Button	JButton
Canvas	ไม่มี
Check Box	JCheckBox
Dialog	JDialog
Frame	JFrame
Label	JLabel
List	JList
Menu	JMenu
Menu Bar	JMenuBar
Menu Item	JMenuItem
Panel	JPanel
Scrollbar	JScrollbar
Text Area	JTextArea
Text Component	JTextComponent
Text Field	JTextField

และ Swing JFC (Components และ Containers) ที่ไม่มีใน AWTได้แก่ Box, JColorChooser, JComboBox, JDeskTopPane, JEditorPane, JInternalFrame, JLayeredPane, JRadioButton, JSeparator, JSlider, JSpiltPane, JTable, JTextPane, JToggleButton, JToolbar, JToolTip, JTree และ JviewPort

บทที่ 5

ผลการทดลอง

5.1 ขั้นตอนการออกแบบ HMI (Human Machine Interface) สำหรับกระบวนการจำลองการควบคุมระดับน้ำ

- พิจารณาการทำงานหลักของกระบวนการ
    1. วาล์วควบคุมระดับน้ำเข้าถัง a (Va)
    2. วาล์วควบคุมระดับน้ำออกถัง a (Vc)
    3. วาล์วควบคุมระดับน้ำเข้าถัง b (Vb)
    4. วาล์วควบคุมระดับน้ำออกถัง b (Vd)
    5. วาล์วควบคุมระดับน้ำออกถัง c (Vc)
- ออกแบบภาพกราฟิกและแสดงสัญลักษณ์ของการควบคุมกระบวนการ
    1. เลือกรูปภาพของกระบวนการมาเป็นกราฟิกเพื่อง่ายในการใช้งาน
    2. ใช้ปุ่มกดแบบ Button สำหรับการควบคุมการทำงานและใช้ชื่อของปุ่มกดเพื่อบอกหน้าที่การทำงาน
- พิจารณาการควบคุมการทำงานของกระบวนการ
    1. การควบคุมแบบอัตโนมัติ
    2. การควบคุมด้วยมือ
    3. เริ่มการทำงานของกระบวนการ
    4. หยุดการทำงานของกระบวนการ
- พิจารณาสภาวะการของควบคุมการทำงานของกระบวนการ
    1. วาล์วควบคุมระดับน้ำเข้าถัง a ทำงาน (Va ON)
    2. วาล์วควบคุมระดับน้ำเข้าถัง a หยุดทำงาน (Va OFF)
    3. วาล์วควบคุมระดับน้ำเข้าถัง b ทำงาน (Vb ON)
    4. วาล์วควบคุมระดับน้ำเข้าถัง b หยุดทำงาน (Vb OFF)
    5. วาล์วควบคุมระดับน้ำเข้าถัง c ทำงาน (Vc ON)
    6. วาล์วควบคุมระดับน้ำเข้าถัง c หยุดทำงาน (Vc OFF)
    7. วาล์วควบคุมระดับน้ำเข้าถัง d ทำงาน (Vd ON)
    8. วาล์วควบคุมระดับน้ำเข้าถัง d หยุดทำงาน (Vd OFF)
    9. วาล์วควบคุมระดับน้ำเข้าถัง e ทำงาน (Ve ON)
    10. วาล์วควบคุมระดับน้ำเข้าถัง e หยุดทำงาน (Ve OFF)

11. การควบคุมแบบอัตโนมัติ (AUTO)

12. การควบคุมด้วยมือ (MANUAL)

13. เริ่มการทำงานของกระบวนการ (START)

14. หยุดการทำงานของกระบวนการ (STOP)

- จัดเลย์เอาท์สำหรับจัดวางภาพกราฟิกและปุ่มกดต่างๆ ให้ง่ายต่อการใช้งาน
- ใส่เงื่อนไขการทำงานของส่วนต่างๆ ตามสภาวะการควบคุมการทำงาน
- ทดลองการทำงานตามเงื่อนไขต่างๆ และสังเกตการทำงานของกระบวนการ

5.2 ขั้นตอนการออกแบบ HMI (Human Machine Interface) สำหรับกระบวนการจำลองการควบคุมการทำงานของมอเตอร์

- พิจารณาการทำงานหลักของกระบวนการ

1. มอเตอร์เริ่มทำงาน
2. มอเตอร์หยุดทำงาน

- ออกแบบภาพกราฟิก และแสดงสัญลักษณ์ของการควบคุมกระบวนการ

1. เลือกรูปภาพของกระบวนการมาเป็นกราฟิกเพื่อง่ายในการใช้งาน
2. ใช้ปุ่มกดแบบ Button สำหรับการควบคุมการทำงาน และใช้ชื่อของปุ่มกดเพื่อบอกหน้าที่การทำงาน

- พิจารณาการควบคุมการทำงานของกระบวนการ

1. เริ่มการทำงานของกระบวนการ
2. หยุดการทำงานของกระบวนการ

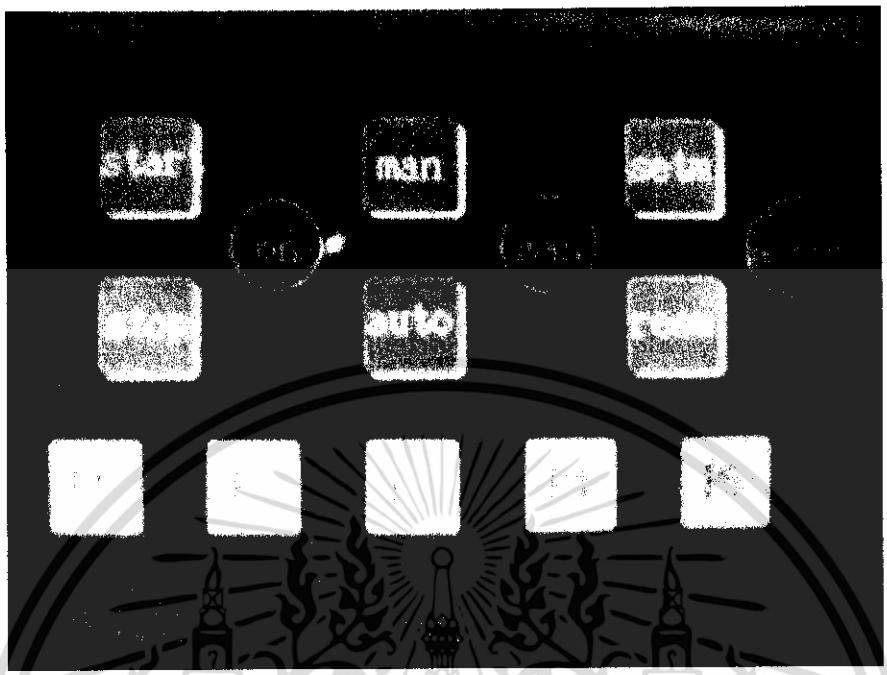
- พิจารณาสภาวะการของควบคุมการทำงานของกระบวนการ

1. เริ่มการทำงานของกระบวนการ (START)
2. หยุดการทำงานของกระบวนการ (STOP)

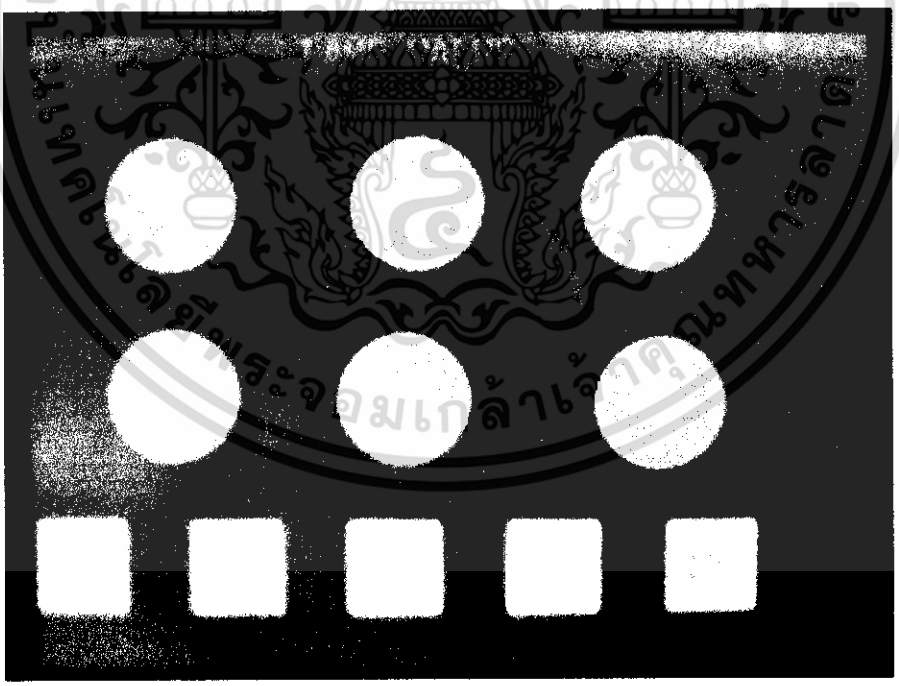
- จัดเลยเอาท์สำหรับจัดวางภาพกราฟิกและปุ่มกดต่างๆ ให้ง่ายต่อการใช้งาน
- ใส่เงื่อนไขการทำงานของส่วนต่างๆ ตามสภาวะการควบคุมการทำงาน
- ทดลองการทำงานตามเงื่อนไขต่างๆ และสังเกตการทำงานของกระบวนการ

5.3 การทดลองใช้งาน HMI ผ่านโครงข่ายเครื่องควบคุมแบบโปรแกรมได้

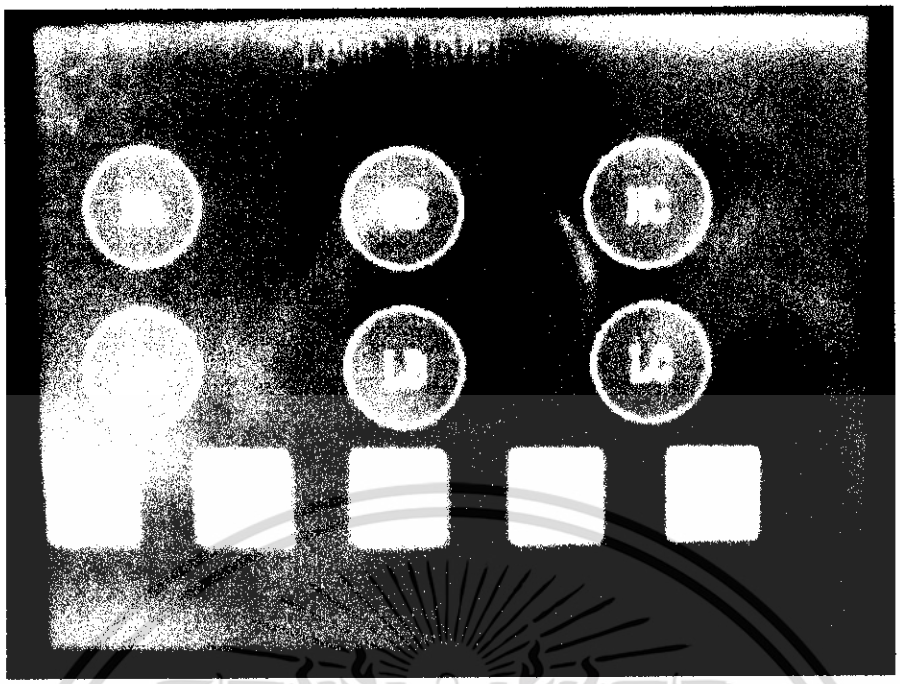
1. การใช้งาน HMI บนเครื่องควบคุมแบบหน้าจอสัมผัส



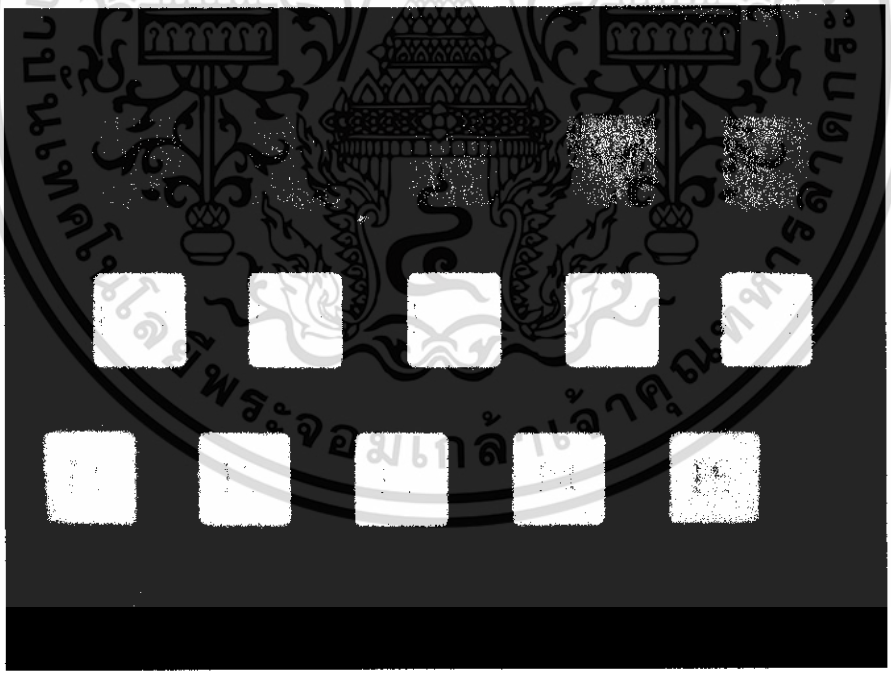
ภาพที่ 5.1 แสดงรูปการใช้งาน HMI บนเครื่องควบคุมแบบหน้าจอสัมผัสในหน้าที่ 1



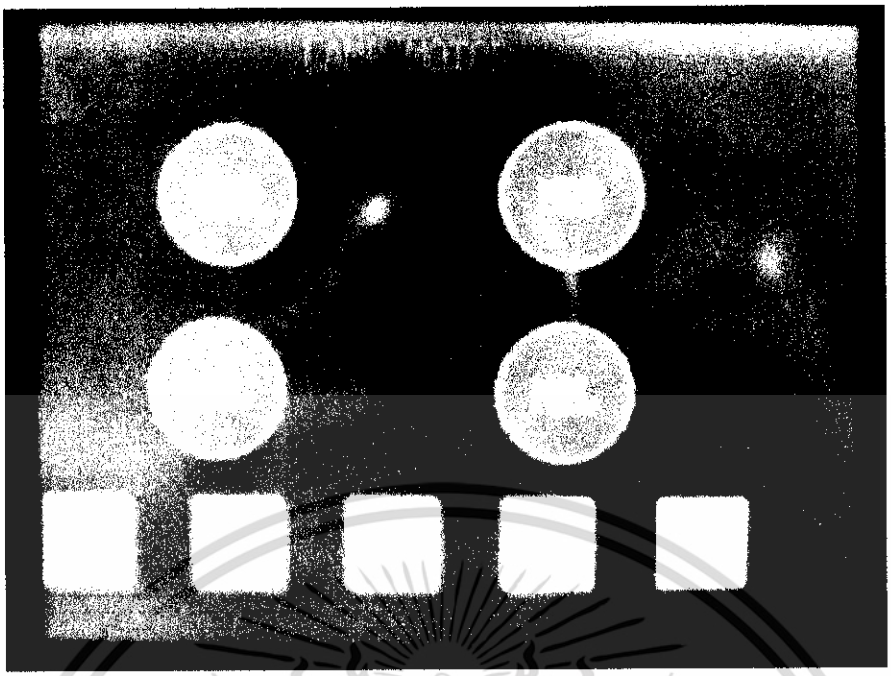
ภาพที่ 5.2 แสดงรูปการใช้งาน HMI บนเครื่องควบคุมแบบหน้าจอสัมผัสในหน้าที่ 2



ภาพที่ 5.3 แสดงรูปการใช้งาน HMI บนเครื่องควบคุมแบบหน้าจอสัมผัสหน้าที่ 3

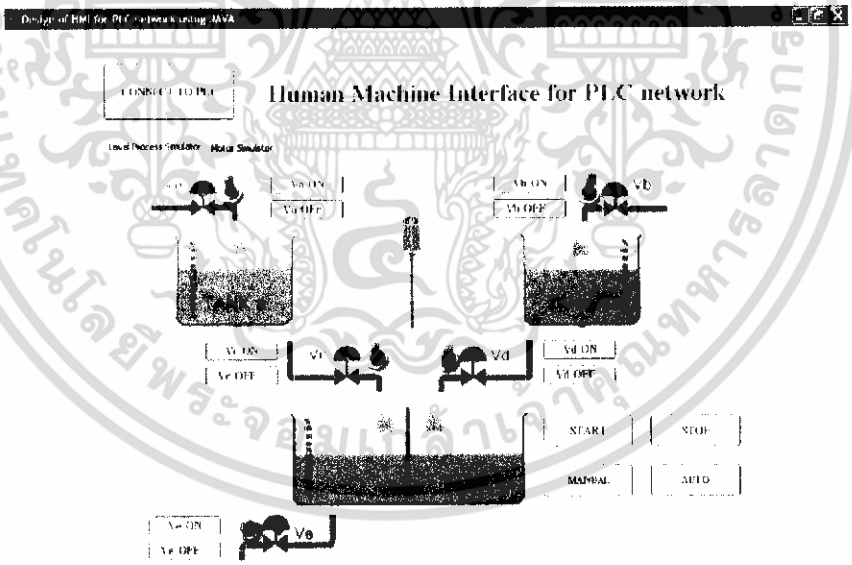


ภาพที่ 5.4 แสดงรูปการใช้งาน HMI บนเครื่องควบคุมแบบหน้าจอสัมผัสในหน้าที่ 4

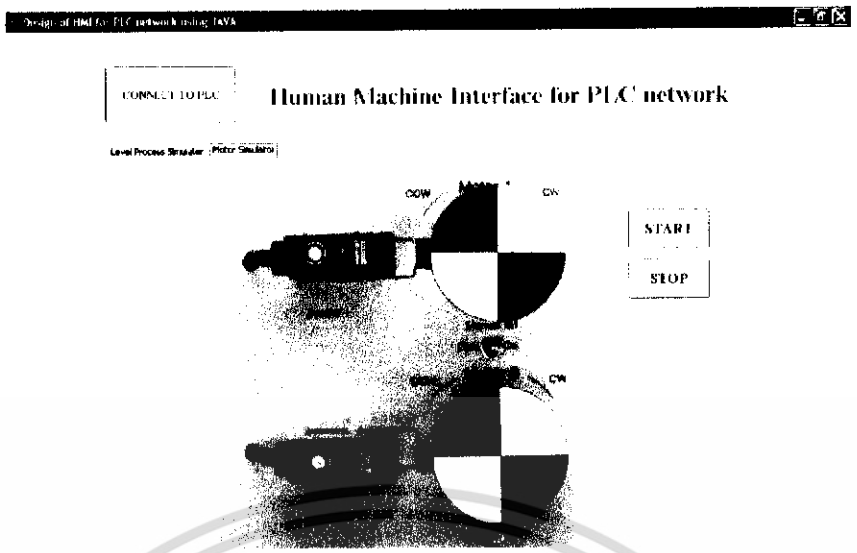


ภาพที่ 5.5 แสดงรูปการใช้งาน HMI บนเครื่องควบคุมแบบหน้าจอสัมผัสในหน้าที่ 5

2. การใช้งาน HMI บนเครื่องคอมพิวเตอร์



ภาพที่ 5.6 แสดงรูปการใช้งาน HMI บนเครื่องคอมพิวเตอร์สำหรับกระบวนการจำลองการควบคุมระดับน้ำ



ภาพที่ 5.7 แสดงรูปการใช้งาน HMI บนเครื่องคอมพิวเตอร์สำหรับกระบวนการจำลองการทำงานของมอเตอร์

ภาพที่ 5.8 แสดงรูปการเชื่อมต่อ HMI ผ่านโครงข่ายเครื่องควบคุมแบบโปรแกรมได้

บทที่ 6
สรุปผลการวิจัยและข้อเสนอแนะ

6.1 บทสรุป

จากการศึกษาและใช้งานได้ผลเป็นที่น่าพอใจในระดับหนึ่ง และมีแนวโน้มที่จะพัฒนาประสิทธิภาพของโปรแกรมให้สูงขึ้นต่อไป เพื่อความสะดวกในการควบคุมการทำงานของระบบที่ถูกควบคุมโดยเครื่องควบคุมแบบโปรแกรมได้ เช่น การพัฒนาซอฟท์แวร์ให้สามารถทำงานร่วมกับระบบฐานข้อมูล เพื่อจัดเก็บข้อมูลการทำงานของระบบควบคุม

6.2 ข้อเสนอแนะและแนวทางการวิจัย

6.2.1 แนวทางในการพัฒนาต่อ

จากแนวทางการออกแบบ HMI (Human Machine Interface) ผ่านโครงข่ายเครื่องควบคุมแบบโปรแกรมได้ สามารถนำไปประยุกต์ใช้และพัฒนาการควบคุมกระบวนการทางอุตสาหกรรมอื่นๆ ที่ถูกควบคุมโดยเครื่องควบคุมแบบโปรแกรมได้ และสามารถนำประยุกต์ใช้งานกับเครื่องควบคุมแบบโปรแกรมได้ที่รองรับข้อตกลงในการสื่อสารข้อมูลแบบ Host Link Communication อีกทั้งยังสามารถประยุกต์ใช้ในการเก็บข้อมูลต่างๆ รวมทั้งสถานะการทำงานของกระบวนการทางอุตสาหกรรมในรูปแบบของฐานข้อมูล (Data Acquisition) ตลอดจนการพัฒนาเป็นระบบ SCADA

6.2.2 ข้อจำกัดของโครงงาน

- ข้อจำกัดเรื่องข้อตกลงในการติดต่อสื่อสารแบบ Host Link Communication เนื่องจากข้อตกลงในการการติดต่อสื่อสารแบบ Host Link Communication นั้นยอมให้สามารถทำการควบคุมการทำงานของเครื่องควบคุมแบบโปรแกรมได้ที่ทำการเชื่อมต่ออยู่กับเครื่องคอมพิวเตอร์เท่านั้น ไม่สามารถทำการควบคุมการทำงานของเครื่องควบคุมแบบโปรแกรมได้ตัวอื่นๆ รวมถึงข้อจำกัดในเรื่องของการสื่อสารข้อมูลแบบ Host Link Communication ที่ระบุให้เป็นการสื่อสารแบบอนุกรมผ่านพอร์ต RS-232 C โดยกำหนดให้ค่าพารามิเตอร์ต่างๆ เป็นดังนี้คือ อัตราบอร์ดเท่ากับ 9600 บิตเริ่มต้นเท่ากับ 1 บิตข้อมูลเท่ากับ 7 บิตสุดท้ายเท่ากับ 2 และพาริตี้บิตเป็น EVEN เท่านั้น อีกทั้งข้อยังมีข้อจำกัดในเรื่องของความรวดเร็วในการสื่อสารข้อมูลเนื่องจากการสื่อสารผ่านพอร์ตอนุกรมนั้นเป็นเป็นการสื่อสาร

แบบไม่สัมพันธ์ (Asyncronous) แนวทางการพัฒนาต่อจากโครงงานนี้ จึงไม่เหมาะสำหรับนำไปใช้กับกระบวนการที่ต้องการการสื่อสารข้อมูลที่ต้องการความรวดเร็วสูง

- ข้อจำกัดเรื่องการโปรแกรมภาษาจาวา เนื่องจากการโปรแกรมด้วยภาษาจาวาสำหรับการส่งผ่านข้อมูลทางพอร์ตอนุกรมยังเป็นความรู้ที่ถูกจำกัดอยู่เพียงในวงแคบ ทั้งในประเทศไทยและต่างประเทศ ยังไม่มีตำราเกี่ยวกับการโปรแกรมภาษาจาวาสำหรับการส่งผ่านข้อมูลทางพอร์ตอนุกรมเป็นที่แพร่หลาย (ตำราในประเทศไทยไม่มีเลย) ทำให้เป็นอุปสรรคอย่างมากในการพัฒนาโครงงาน

บรรณานุกรม

1. OMRON, "SYSMAC Programmable Controller C200HS"
2. Craig Peacock, "Interfacing the serial / RS232 Port V5.0"
3. OMRON, "Programmable Terminals"
4. กิตติ ภักดีวัฒนะกุล, " JAVA ฉบับโปรแกรมเมอร์" , พิมพ์ครั้งที่ 1, สำนักพิมพ์ เคทีพี- คอมพ์ แอนด์ คอนซัลท์ จำกัด
5. กิตติ ภักดีวัฒนะกุล, "คัมภีร์ JAVAเล่ม1" , พิมพ์ครั้งที่ 1, สำนักพิมพ์ เคทีพี คอมพ์ แอนด์ คอน ซัลท์ จำกัด
6. ดร.วีระศักดิ์ ซึงถาวร, " JAVA PROGRAMMING VOLUME I ", บริษัท ซีเอ็ดยูเคชั่น จำกัด (มหาชน)
7. ดร.วีระศักดิ์ ซึงถาวร, " JAVA PROGRAMMING VOLUME II" , บริษัท ซีเอ็ดยูเคชั่น จำกัด (มหาชน)
8. ดร.วีระศักดิ์ ซึงถาวร, " JAVA PROGRAMMING VOLUME III" , บริษัท ซีเอ็ดยูเคชั่น จำกัด (มหาชน)
9. ผศ. ไสว พงศ์สวัสดิ์, ผศ. ทวีพล ซื่อสัตย์,"ปฏิบัติการวิศวกรรมการวัดคุม 3", ภาควิชาวิศวกรรมการวัดคุม คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง. กรุงเทพ : พิมพ์ครั้งที่ 1

ภาคผนวก

ตารางแสดงความหมายของ End Code ในการติดต่อสื่อสารแบบ Host Link Communication

End code	Contents	Probable cause	Corrective measures
00	Normal completion	---	---
01	Not executable in RUN mode	The command that was sent cannot be executed when the PC is in RUN mode.	Check the relation between the command and the PC mode.
02	Not executable in MONITOR mode	The command that was sent cannot be executed when the PC is in MONITOR mode.	
0B	Not executable in PROGRAM mode	The command that was sent cannot be executed when the PC is in PROGRAM mode.	This code is not presently being used.
13	FCS error	The FCS is wrong. Either the FCS calculation is mistaken or there is adverse influence from noise.	Check the FCS calculation method. If there was influence from noise, transfer the command again.
14	Format error	The command format is wrong.	Check the format and transfer the command again.
15	Entry number data error	The areas for reading and writing are wrong.	Correct the areas and transfer the command again.
16	Command not supported	The specified command does not exist.	Check the command code.
18	Frame length error	The maximum frame length was exceeded.	Divide the command into multiple frames.
19	Not executable	Items to read not registered for composite command (QQ).	Execute QQ to register items to read before attempting batch read.
23	User memory write-protected	Pin 1 on C200HS DIP switch is ON.	Turn OFF pin 1 to execute.
A3	Aborted due to FCS error in transmit data	The error was generated while a command extending over more than one frame was being executed. Note: The data up to that point has already been written to the appropriate area of the CPU.	Check the command data and try the transfer again.
A4	Aborted due to format error in transmit data		
A5	Aborted due to entry number data error in transmit data		
A8	Aborted due to frame length error in transmit data		
Other	---	Influence from noise was received.	Transfer the command again.

ภาคผนวก แสดง Header Code ในการติดต่อสื่อสารแบบ Host Link Communication

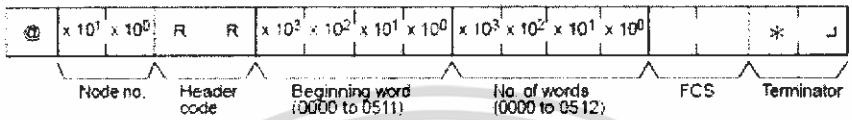
11-3 Host Link Commands

This section explains the commands that can be issued from the host computer to the PC.

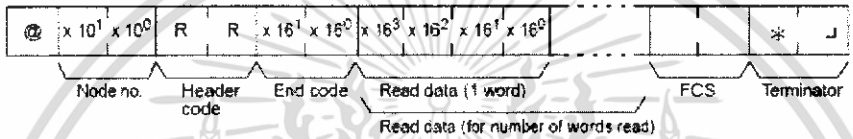
11-3-1 IR/SR AREA READ — RR

Reads the contents of the specified number of IR and SR words, starting from the specified word.

Command Format



Response Format



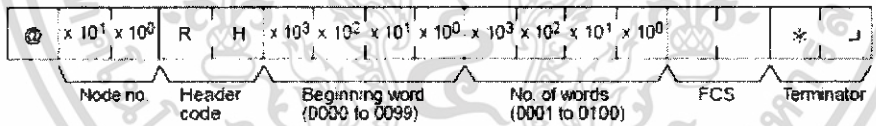
Parameters

Read Data (Response)
The contents of the number of words specified by the command are returned in hexadecimal as a response. The words are returned in order, starting with the specified beginning word.

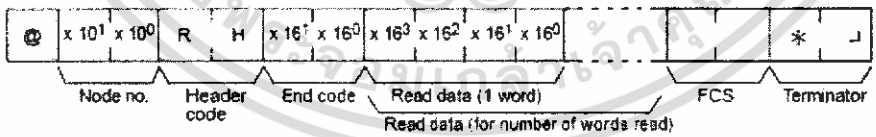
11-3-3 HR AREA READ — RH

Reads the contents of the specified number of HR words, starting from the specified word.

Command Format



Response Format



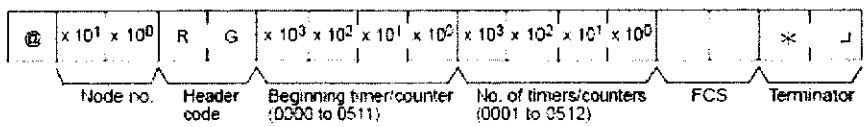
Parameters

Read Data (Response)
The contents of the number of words specified by the command are returned in hexadecimal as a response. The words are returned in order, starting with the specified beginning word.

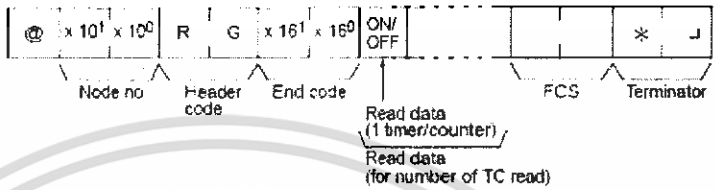
11-3-5 TC STATUS READ — RG

Reads the status of the Completion Flags of the specified number of timers/counters, starting from the specified timer/counter.

Command Format



Response Format



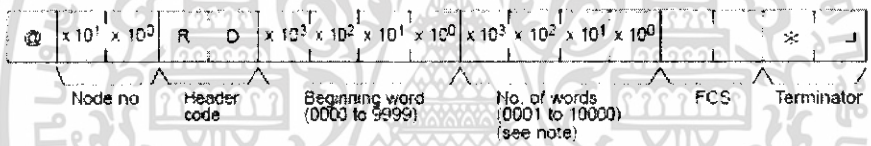
Parameters

Read Data (Response)
The status of the number of Completion Flags specified by the command is returned as a response. "1" indicates that the Completion Flag is ON.

11-3-6 DM AREA READ — RD

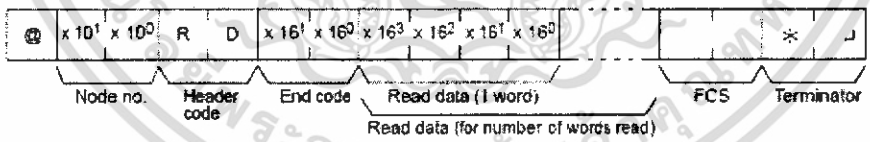
Reads the contents of the specified number of DM words, starting from the specified word.

Command Format



Note
1. If 10,000 words have to be read, specify the number of words to be read as 0000.
2. DM 6656 to DM 6999 do not exist. An error will not, however, result if you try to read these words. Instead, "0000" will be returned as a response.

Response Format



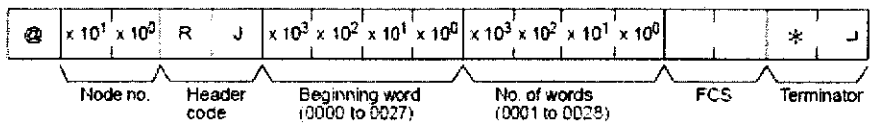
Parameters

Read Data (Response)
The contents of the number of words specified by the command are returned in hexadecimal as a response. The words are returned in order, starting with the specified beginning word.

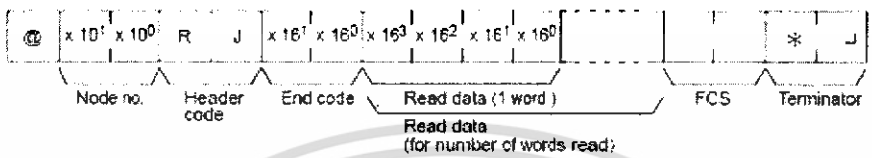
11-3-7 AR AREA READ — RJ

Reads the contents of the specified number of AR words, starting from the specified word.

Command Format



Response Format



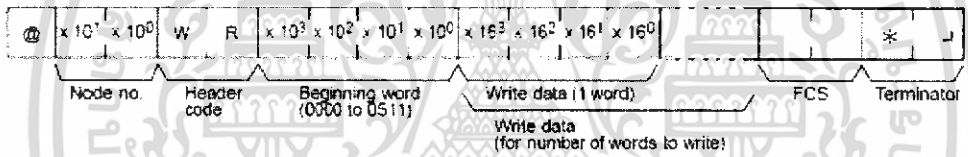
Parameters

Read Data (Response)
The contents of the number of words specified by the command are returned in hexadecimal as a response. The words are returned in order, starting with the specified beginning word.

11-3-8 IR/SR AREA WRITE — WR

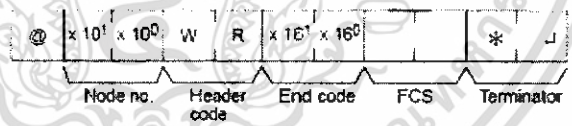
Writes data to the IR and SR areas, starting from the specified word. Writing is done word by word.

Command Format



Note Data cannot be written to words 253 to 255. If there is an attempt to write to these words, no error will result, but nothing will be written to these words.

Response Format



Parameters

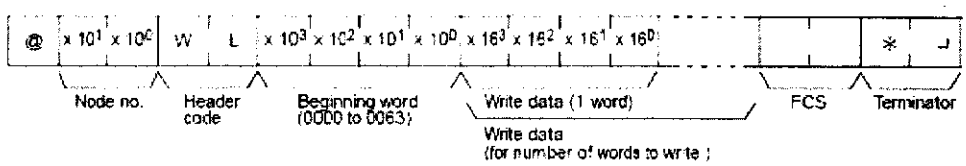
Write Data (Command)
Specify in order the contents of the number of words to be written to the IR or SR area in hexadecimal, starting with the specified beginning word.

Note If data is specified for writing which exceeds the allowable range, an error will be generated and the writing operation will not be executed. If, for example, 511 is specified as the beginning word for writing,and two words of data are specified, then 0512 will become the last word for writing data, and the command will not be executed because SR 512 is beyond the writeable range.

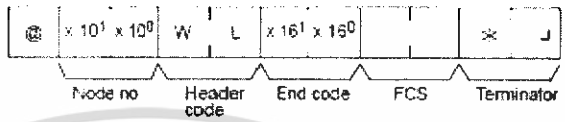
11-3-9 LR AREA WRITE — WL

Writes data to the LR area, starting from the specified word. Writing is done word by word.

Command Format



Response Format



Parameters

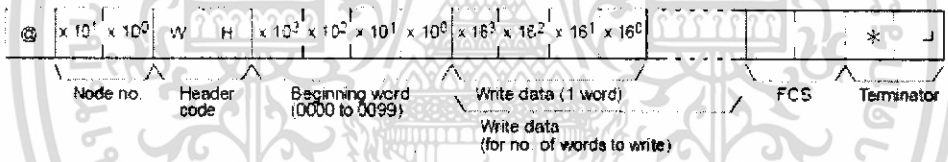
Write Data (Command)
Specify in order the contents of the number of words to be written to the LR area in hexadecimal, starting with the specified beginning word.

Note If data is specified for writing which exceeds the allowable range, an error will be generated and the writing operation will not be executed. If, for example, 60 is specified as the beginning word for writing and five words of data are specified, then 64 will become the last word for writing data, and the command will not be executed because LR 64 is beyond area boundary.

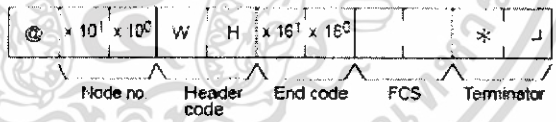
11-3-10 HR AREA WRITE — WH

Writes data to the HR area, starting from the specified word. Writing is done word by word.

Command Format



Response Format



Parameters

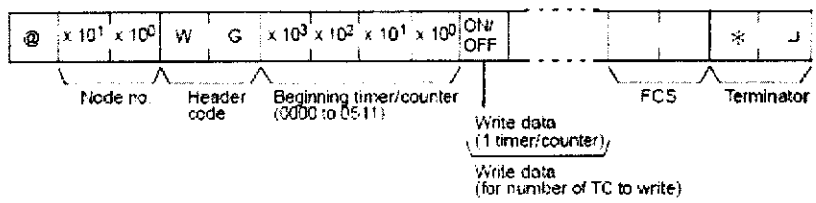
Write Data (Command)
Specify in order the contents of the number of words to be written to the HR area in hexadecimal, starting with the specified beginning word.

Note If data is specified for writing which exceeds the allowable range, an error will be generated and the writing operation will not be executed. If, for example, 98 is specified as the beginning word for writing, and three words of data are specified, then 100 will become the last word for writing data, and the command will not be executed because HR 100 is beyond area boundary.

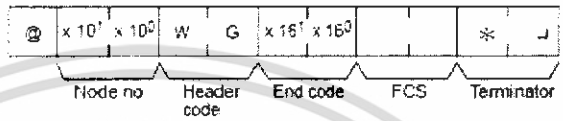
11-3-12 TC STATUS WRITE — WG

Writes the status of the Completion Flags for timers and counters in the TC area, starting from the specified timer/counter (number). Writing is done number by number.

Command Format



Response Format



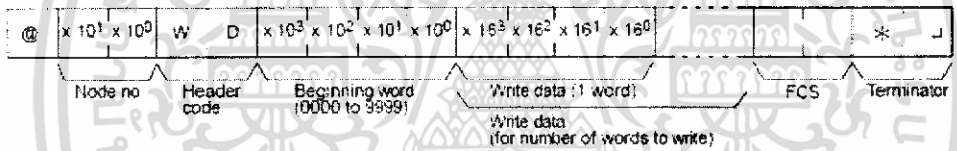
Parameters

Write Data (Command)
Specify the status of the Completion Flags, for the number of timers/counters to be written, in order (from the beginning word) as ON (i.e., "1") or OFF (i.e., "0"). When a Completion Flag is ON, it indicates that the time or count is up.

11-3-13 DM AREA WRITE — WD

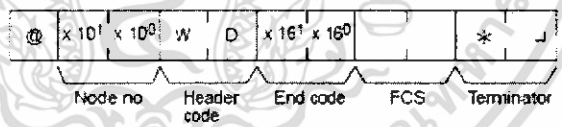
Writes data to the DM area, starting from the specified word. Writing is done word by word.

Command Format



Note DM 6656 to DM 6999 do not exist. An error will not, however, result if you try to write to these words.

Response Format



Parameters

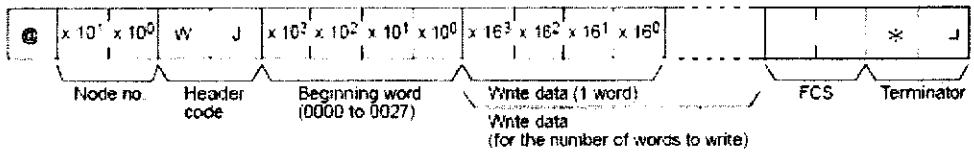
Write Data (Command)
Specify in order the contents of the number of words to be written to the DM area in hexadecimal, starting with the specified beginning word.

Note
1. If data is specified for writing which exceeds the allowable range, an error will be generated and the writing operation will not be executed. If, for example, 9998 is specified as the beginning word for writing, and three words of data are specified, then 10000 will become the last word for writing data, and the command will not be executed because DM 10000 is beyond the writeable range.
2. Be careful about the configuration of the DM area, as it varies depending on the CPU model.

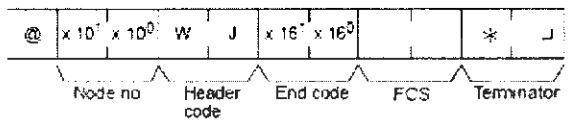
11-3-14 AR AREA WRITE — WJ

Writes data to the AR area, starting from the specified word. Writing is done word by word.

Command Format



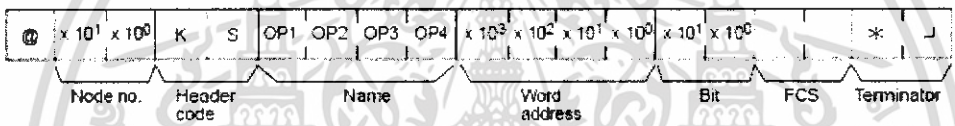
Response Format



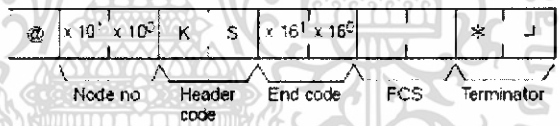
11-3-24 FORCED SET — KS

Force sets a bit in the IR, SR, LR, HR, AR, or TC area. Once a bit has been forced set or reset, that status will be retained until FORCED SET/RESET CANCEL (KC) is transmitted.

Command Format



Response Format



Parameters

Name, Word address, Bit (Command)
In "Name", specify the area (i.e., IR, SR, LR, HR, AR, or TC) that is to be forced set. Specify the name in four characters. In "Word address", specify the address of the word, and in "Bit" the number of the bit that is to be forced set.

Name				Classification	Word address setting range	Bit
OP1	OP2	OP3	OP4			
C	I	O	(S)	IR or SR	0000 to 0511	00 to 15 (decimal)
L	R	(S)	(S)	LR	0000 to 0063	
H	R	(S)	(S)	HR	0000 to 0099	
A	R	(S)	(S)	AR	0000 to 0027	
T	I	M	(S)	Completion Flag (timer)	0000 to 0511	Always 00.
T	I	M	H	Completion Flag (high-speed timer)		
C	N	T	(S)	Completion Flag (counter)		
C	N	T	R	Completion Flag (reversible counter)		
T	T	I	M	Completion Flag (totalizing timer)		

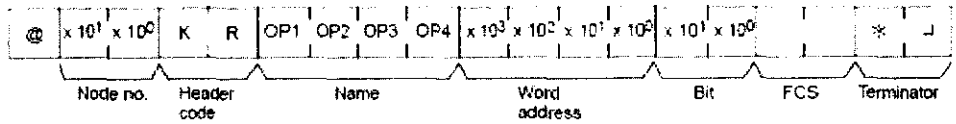
(S): Space

Note
1. The area specified under "Name" must be in four characters. Fill any gaps with spaces to make a total of four characters.
2. Words 253 to 255 cannot be set when the CIO Area is specified.

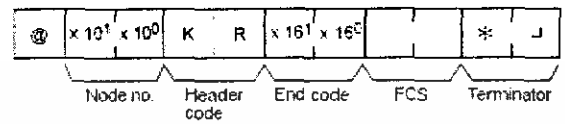
11-3-25 FORCED RESET — KR

Force resets a bit in an IR, SR, LR, HR, AR, or TC area. Once a bit has been forced set or reset, that status will be retained until FORCED SET/RESET CANCEL (KC) is transmitted.

Command Format



Response Format



Parameters

Name, Word address, Bit (Command)

In "Name", specify the area (i.e., IR, SR, LR, HR, AR, or TC) that is to be forced reset. Specify the name in four characters. In "Word address", specify the address of the word, and in "Bit" the number of the bit that is to be forced reset.

Name				Classification	Word address setting range	Bit
OP1	OP2	OP3	OP4			
C	I	O	(S)	IR or SR	0000 to 0511	00 to 15 (decimal)
L	R	(S)	(S)	LR	0000 to 0063	
H	R	(S)	(S)	HR	0000 to 0099	
A	R	(S)	(S)	AR	0000 to 0027	
T	I	M	(S)	Completion Flag (timer)	0000 to 0511	Always 00.
T	I	M	H	Completion Flag (high-speed timer)		
C	N	T	(S)	Completion Flag (counter)		
C	N	T	R	Completion Flag (reversible counter)		
T	T	I	M	Completion Flag (totalizing timer)		

(S): Space