การประยุกต์ใช้งานสมองกลฝังตัวสำหรับการออกแบบ PLC บนเครือข่ายอีเธอร์เน็ต

APPLYING EMBEDDED SYSTEM TO DESIGN PROGRAMMABLE LOGIC CONTROLLER BASED ON ETHERNET NETWORK

สามารถ เจิศเศรี

SAMART JERTSAREE

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชาวิศวกรรมการวัดคุม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ. 2552

KMITL-2009-EN-M-060-074

การประยุกต์ใช้งานสมองกลฝังตัวสำหรับการออกแบบ PLC บนเครือข่ายอีเธอร์เน็ต

Applying embedded system to design programmable logic controller based on Ethernet network

สามารถ เลิศเสรี

SAMART LERTSAREE

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชาวิศวกรรมการวัดคุม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ. 2552

KMITL 2009-EN-M-060-074

Applying embedded system to design programmable logic controller based on Ethernet network

SAMART LERTSAREE

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

MASTER OF ENGINEERING IN INSTRUMENTATION ENGINEERING

FACULTY OF ENGINEERING

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2009

KMITL 2009-EN-M-060-074

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ใบรับรองวิทยานิพนธ์

หัวข้อวิทยานิพนธ์ การประยุกต์ใช้งานสมองกลฝังตัวสำหรับการออกแบบ PLC บนเครือข่ายอีเธอร์เน็ต

Thesis Title Applying Embedded System to Design Programmable Logic Controller Based on Ethernet Network

นักศึกษา นายสามารถ เลิศเสรี

รหัสประจำตัว 47060604

ปริญญา วิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชา วิศวกรรมการวัดคุม

อาจารย์ที่ปรึกษาวิทยานิพนธ์ รศ.สุพรรณ กุลพาณิชย์

หมายเลขวิทยานิพนธ์ KMITL-2009-EN-M-060-074

คณะกรรมการสอบวิทยานิพนธ์		ลายมือชื่อ
รศ.วิริยะ	กองรัตน์	
ผศ.ทรงชัย	วีระทวีมาศ	
รศ.ดร.ประเสริฐ	ปิ่นปฐมรัฐ	
รศ.ดร.ทวีพล	ซื่อสัตย์	
รศ.สุพรรณ	กุลพาณิชย์	

วัน / เดือน/ ปี ที่สอบ วันพุธที่ 20 พฤษภาคม พ.ศ. 2552 เวลา 09.30-11.30 น.

สถานที่สอบ ณ อาคาร A ชั้น 5 ห้องประชุม 3

คณะวิศวกรรมศาสตร์ รับรองแล้ว

(รองศาสตราจารย์ ดร.กอบชัย เดชหาญ)

คณบดี คณะวิศวกรรมศาสตร์

วันที่ 20 พฤษภาคม พ.ศ. 2552

หัวข้อวิทยานิพนธ์	การประยุกต์ใช้งานสมองกลฝังตัวสำหรับการออกแบบ PLC บนเครือข่ายอีเธอร์เน็ต
นักศึกษา	สามารถ เลิศเสรี
รหัสนักศึกษา	47060604
ปริญญา	วิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชา	วิศวกรรมการวัดคุม
พ.ศ.	2552
อาจารย์ที่ปรึกษาวิทยานิพนธ์	รศ. สุพรรณ กุลพาณิชย์

บทคัดย่อ

บทความนี้กล่าวถึงการออกแบบและพัฒนา คอนโทรลเลอร์แบบสมองกลฝังตัว (Embedded Controller) ให้เป็นเครื่องควบคุม PLC ที่สามารถเชื่อมต่อกับโฮสคอมพิวเตอร์ผ่านระบบโครงข่ายแบบอีเธอร์เน็ต (Ethernet) การออกแบบแบ่งเป็นส่วนของฮาร์ดแวร์ และ ซอฟท์แวร์ โดยส่วนของฮาร์ดแวร์ได้เลือกเอาสมองกลฝังตัวตระกูล ARM7TDMI-S ที่ประมวลผลข้อมูลขนาด บิต เบอร์ 32 LPC2378 เป็นตัวควบคุมการทำงานหลักซึ่งประกอบด้วยหน่วยอินพุตเอาต์พุตแบบดิจิตอล ระบบสื่อสารข้อมูลที่เป็นทั้งแบบอนุกรมและขนาน หน่วยความจำภายในและภายนอก ในส่วนของซอฟท์แวร์ ได้ใช้ระบบปฏิบัติการเวลาจริง (Real time OS) ที่มีอยู่ในระบบสมองกลฝังตัวให้เป็นระบบปฏิบัติการของเครื่องควบคุม โปรแกรมตัวแปลงภาษา (Interpreter) เป็นส่วนหนึ่งของระบบปฏิบัติการที่แปลงจากไฟล์ตัวอักษร (Text File) ให้เป็นรหัสการทำงาน (Operating Code) ในรูปของไฟล์แบบไบนารี (Binary File) การพัฒนาระบบปฏิบัติการนี้อาศัยโปรแกรมตัวแปรภาษา C ตามมาตรฐาน ANSI (ANSI Standard C) การทดสอบประสิทธิภาพการทำงานของเครื่องควบคุมทำได้โดยการวัดเวลาในการสแกน (Scan Time) เมื่อมีการ Online และ Offline พอร์ทอีเธอร์เน็ต

Thesis Title	Applying embedded system to design programmable logic controller based on Ethernet network
Student	Samaer Lertsaree
Student ID.	47060604
Degree	Master of Engineering
Programme	Instrumentation Engineering
Year	2009
Thesis Advisor	Assoc. Prof. Suphan Gulphanich

ABSTRACT

This paper proposes the design and development of embedded controller to programmable logic controller (PLC) linked to host computer via Ethernet network. The design. hardware part, The hardware consists of the 32- bit LPC2378 that is a family of ARM7TDMI-S embedded system applied to be the main controller comprising of digital input and output, both serial and parallel communication, internal and external memories. The software part operate as in real-time Operating System (OS) based on embedded system. The Interpreter a part of OS generates code from text file to a binary file. The development uses ASCI C Complier program according to the efficiency of the PLC and network commutation was illustrated by measuring the scan time during online and off-line Ethernet port.

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จได้ด้วยดี โดยคำแนะนำและคำปรึกษาท่านอาจารย์สุพรรณ กุลพาณิชย์ ซึ่งเป็นอาจารย์ผู้ควบคุมวิทยานิพนธ์ ข้าพเจ้ารู้สึกซาบซึ้งในความอนุเคราะห์จากท่านอาจารย์ และขอขอบพระคุณเป็นอย่างสูง

ขอขอบคุณเพื่อนๆ พี่ๆ และน้องๆ ในภาควิชาวัดคุม คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ที่ได้ให้ความช่วยเหลือ คำแนะนำและกำลังใจตลอดมา

สุดท้ายนี้ข้าพเจ้าขอกราบขอบพระคุณ บิดา มารดา และครอบครัวข้าพเจ้าที่ให้การสนับสนุนในทุกๆเรื่อง และคอยให้กำลังใจตลอดมาทำให้ข้าพเจ้าสามารถทำวิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงได้ด้วยดี

ข้าพเจ้าหวังว่าวิทยานิพนธ์ฉบับนี้จะเป็นประโยชน์กับผู้ที่สนใจทุกท่าน

สามารถ เลิศเสรี

สารบัญ

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญตาราง

สารบัญตาราง(ต่อ)

ตาราง หน้า

สารบัญรูป

สารบัญรูป(ต่อ)

รูปที่ หน้า

สารบัญรูป(ต่อ)

รูปที่ หน้า

สารบัญรูป(ต่อ)

รูปที่ หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

การควบคุมเครื่องจักรกลและอุปกรณ์ขับทางไฟฟ้าในอดีตจะอาศัยแผงวงจรควบคุมแบบรีเลย์เป็นหลักเพื่อควบคุมการ เปิด-ปิด ให้เป็นไปตามเงื่อนไขที่ต้องการในปี ค.ศ. 1960 มีการค้นพบเครื่องควบคุม PLC "PROGRAMMABLE LOGIC CONTROLLER" โดยบริษัทเยนเนอร์รัลมอเตอร์ได้ออกแบบเครื่องควบคุมที่อาศัยไมโครคอนโทรเลอร์ พัฒนาให้เป็นเครื่องควบคุมอเนกประสงค์สำหรับควบคุมเครื่องจักรกลอุตสาหกรรมได้สำเร็จและได้รับความนิยมเรื่อยมาจนเป็นปัจจัยสำคัญในการพัฒนาอุตสาหกรรมทั้งแบบกึ่งอัตโนมัติและแบบอัตโนมัติ ตัวอย่างเช่น อุตสาหะกรรมประกอบรถยนต์ อุตสาหกรรมผลิตอาหารมนุษย์ และ สัตว์ อุตสาหกรรมเครื่องจักรกลเกษตร กล่าวได้ว่าแทบทุกอุตสาหกรรม เครื่องควบคุมนี้มีบทบาทอย่างมากในการเพิ่มประสิทธิภาพการผลิตให้กับผู้ประกอบการ สำหรับงานวิจัยนี้เป็นการออกแบบและพัฒนาเครื่องควบคุม PLC ขึ้นมาใช้งานเอง โดยมีต้นแบบมาจากงานวิจัย Design of programmable logic controllers and I/O Expansionsที่ออกแบบโดยมีโครสร้าง เป็นแบบไมโครโปรเซสเซอร์ Z84C11 ตระกูลเดียวกันกับ Z80 ประมวลผลข้อมูลขนาด 16 บิทที่มีสถาปัตยกรรมทางด้านฮาร์ดแวร์แบบดั้งเดิมมีเพียง CPU เท่านั้นส่วนประกอบต่างๆ และอุปกรณ์รอบข้าง เช่น หน่วยความจำ EPROM หน่วยความจำ RAM ส่วนขยายพอร์ทแบบ ดิจิตอล และ อนาล็อก ถ้าต้องการจำนวนหลายจุดจำเป็นต้องขยายจากภายนอกเอา จึงทำให้เป็นเครื่องควบคุมที่ออกแบบไว้มีขนาดใหญ่ และ มีความน่าเชื่อถือต่ำไม่ดีเท่าที่ควร จากปัญหาเหล่านี้จึงเป็นข้อจำกัดของ CPU และเทคโนโลยีที่มีอยู่ในขณะนั้น สำหรับงานวิจัยนี้ ตัวควบคุมเป็นแบบสมองกลฝังตัวเป็นไอซีตระกูล ARM7TDMI-S ที่ประมวลผลข้อมูลขนาด 32 บิต เบอร์ LPC2378 สถาปัตยกรรมภายในจะถูกควบรวมเข้าด้วยกันและรวมถึงพอร์ทที่ใช้ในการสื่อสารข้อมูลประเภทต่างๆ ก็ถูกนำมาบรรจุไว้ภายในตัวชิพไอซีเพียงตัวเดียว(LSI: Large Scale Integrated Circuit) ส่งผลดีคือระบบที่ออกแบบจะมีขนาดเล็กลง มีความเร็วในการทำงานและความน่าเชื่อถือสูง มีความหลากหลายในมิเดียที่ใช้ในการสื่อสารข้อมูลจึงทำให้ตัวควบคุมแบบสมองกลฝังตัวเป็นที่นิยมและมีบทบาทอย่างมากในการพัฒนางานวิจัย

1.2 ความมุ่งหมายและวัตถุประสงค์ของการศึกษา

วิทยานิพนธ์นี้มุ่งหวังในการศึกษาและออกแบบเพื่อประยุกต์เทคโนโลยีระบบสมองกลฝังตัว (Embedded System)มาออกแบบเป็นเครื่องควบคุม PLC ขนาดเล็ก ทั้งนี้เพราะระบบของเทคโนโลยีสมองกลฝังตัวมีขีดความสามารถสูงทางด้านฮาร์ทแวร์ เช่น พอร์ทอินพุต พอร์ทเอาต์พุต ระบบสื่อสารข้อมูลที่เป็นทั้งแบบอนุกรมและขนานในรูปแบบต่างๆ หน่วยความจำภายในเก็บผลลัพธ์การประมวลผล หน่วยความจำภายนอกแบบ SD card ใช้เก็บโปรแกรมการทำงานของผู้ใช้งาน สิ่งต่างๆเหล่านี้ถูกนำมารวมอยู่ในฮาร์ทแวร์ระบบสมองกลฝังตัวที่ทำงานร่วมกับซอฟท์แวร์โดยมีระบบปฏิบัติการที่ทำงานเป็นแบบมัลติแทสค์ (Multi Tasking) ที่บริการได้หลายงานพร้อมๆกัน และ เสร็จสิ้นงานในเวลาที่กำหนด(Real Time task) ระบบปฏิบัติการนี้มีการตอบสนองของ หน่วยประมวลผลกลางที่มีต่อฮาร์ดแวร์ เราเรียกระบบปฏิบัติการนี้ว่าระบบปฏิบัติการเวลาจริง (Real time Operating System) ดังนั้นเครื่องควบคุม PLC ที่ได้พัฒนามาจากเทคโนโลยีนี้จึงมีขีดความสามารถสูงไปด้วย ทั้งในเรื่องของความเร็วในการตอบสนอง ความสามารถในการควบคุมผ่านระบบเครือข่ายในรูปแบบต่างๆ จึงดูทันสมัยเมื่อเปรียบเทียบกับเครื่องควบคุม PLC แบบดั้งเดิม

1.3 สมมติฐานของการศึกษา

เครื่องควบคุม PLC ที่มีอยู่ในปัจจุบันส่วนใหญ่จะถูกนำเข้าตามต่างประเทศการออกแบบการทำงานของวงจรอิเลคทรอนิคส์ เป็นแบบ Embedded PLC ที่มีไมโครโปรเซสเซอร์แบบชิพเดียวเป็น CPU โดยพื้นฐานฮาร์ทแวร์จะประกอบด้วย หน่วยประมวลผลกลาง หน่วยอินพุท เอาท์พุทแบบดิจิตอล การสื่อสารข้อมูลเป็นแบบอนุกรมทางพอร์ท RS 232 หรือ พอร์ท USB สำหรับฮาร์ทแวร์ส่วนอื่นๆ จะเป็นส่วนขยายที่ต้องจัดหาเพิ่มเติม การทำงานของชุด CPU มีระบบปฏิบัติการเป็นแบบมัลติแทสค์ โปรแกรมการทำงานจะมีซอฟท์แวร์สำเร็จรูปทำงานบนคอมพิวเตอร์เป็นเครื่องมือช่วยเพื่ออำนวยความสะดวกให้กับผู้ใช้งาน ในการพัฒนาโปรแกรมการทำงานของ PLC

สำหรับงานวิจัยนี้ในส่วนของฮาร์ทแวร์ออกแบบเพื่อประยุกต์เทคโนโลยีระบบสมองกลฝังตัว (Embedded System) ไอซีตระกูล ARM7TDMI-S ที่ประมวลผลข้อมูลขนาด 32 บิต เบอร์ LPC2378 เป็นหน่วยประมวลผลกลางของเครื่องควบคุม PLC มีอินพุทและเอาท์พุทแบบดิจิตอลอย่างละ 16 จุด ฮาร์ทแวร์ออกแบบการสื่อสารข้อมูลแบบอนุกรมเป็น RS 232 จำนวน 2 พอร์ท USB 1 พอร์ท ระบบบัสแบบ Can bus และ Ethernet พอร์ท สำหรับ อนาล็อกอินพุทและเอาท์พุทจะใช้เป็นส่วนต่อขยาย โปรแกรมการทำงานจะเขียนโปรแกรมรูปแบบคำสั่ง IL จัดเก็บในหน่วยความจำ SD การ์ด ในรูปแบบไฟล์ตัวอักษร (.txt) จะใช้โปรแกรมสำเร็จรูปของไมโครซอฟท์วินโดว์ ได้แก่ Notepad หรือ โปรแกรมอิดิเตอร์อื่นๆ สร้างไฟล์ตัวอักษรขึ้นมาก่อนและจัดเก็บในหน่วยความจำ SD การ์ด

สร้างไฟล์ตัวอักษรขึ้นมาก่อนและจัดเก็บในหน่วยความจำ SD การ์ด การตรวจสอบสถานการทำงานในหน่วยความจำสามารถกระทำได้ผ่านทางพอร์ทอนุกรม หรือ ตรวจสอบผ่านระบบอินเทอร์เน็ตเว็บบราวเซอร์ผ่านทางพอร์ท Ethernet ในรูปแบบรีโมทการควบคุมระยะไกลผ่านเครือข่ายอีเทอร์เน็ตได้อีกทางหนึ่ง

1.4 ทฤษฎีหรือแนวคิดในการวิจัย

การออกแบบและพัฒนา ได้ศึกษาข้อมูลจากงานวิจัยที่เกี่ยวข้อง ในการออกแบบเครื่องควบคุม PLC ในงานวิจัยของ Gab Seon Rho เรื่อง Implementation of a RISC Microprocessor for programmable logic controllers ปี ค.ศ. 1995ว่าด้วย การออกแบบการทำงานของ PLC โดยใช้ FPGA ออกแบบโครสร้างทางฮาร์ดแวร์ของ PLC ขึ้นมา ข้อได้เปรียบของ การทำงานภายในเป็นการทำงานด้วยเงื่อนไขทางฮาร์ดแวร์ในรูปแบบของเกตนำมาจัดเรียงกันเป็นวงจรทำให้มีความเร็วในการประมวลผลสูง และอีกหนึ่งในงานวิจัยของ Suphan Gulpanich เรื่อง Design of programmable logic controllers and I/O Expansions ปี ค.ศ.2005 การออกแบบการทำงานของ PLC โดยให้มีการทำงานเงื่อนไขทางซอฟแวร์แทน ให้กับชุดไมโครคอนโทรลเลอร์ แต่งานวิจัยที่สองที่กล่าวถึง มีการประมวลผลกลางที่ล่าช้าทำให้ไม่สามารถตอบสนองได้ทันเวลา สำหรับในวิทยานิพนธ์นี้ออกแบบโดยใช้การทำงานเงื่อนไขทางซอฟแวร์เช่นเดียวกับงานวิจัยที่สอง ข้อดีของการออกแบบโดยซอฟแวร์ PLC บน ไมโครคอนโทรลเลอร์ จะมีจุดเด่นคือ สามารถใช้โครงสร้างทางฮาร์ดแวร์ ที่มีอยู่ในไมโครคอนโทรลเลอร์ เช่น USB พอร์ท, Can bus, Ethernet พอร์ท นำมาประยุกต์ใช้งานได้ทันที่โดยไม่จำเป็นต้องออกแบบโครงสร้างทางฮาร์ดแวร์ ขึ้นมาใหม่ทั้งหมด และในการแก้ไขทางซอฟแวร์ จะทำได้สะดวกกว่าการแก้ไขทางฮาร์ดแวร์ ส่วนในด้านความเร็วในการประมวลผลนั้นสามารถใช้ไมโครคอนโทรลเลอร์ที่มีความเร็วในการประมวลผลสูง เข้ามาแก้ไขปัญหาความล่าช้าของการทำงานได้

ไมโครคอนโทรลเลอร์แบบสมองกลฝังตัวขอเรียกว่าเป็นเครื่องควบคุมพื้นฐาน เพื่อพัฒนาให้เป็นเครื่องควบคุม PLC ขอเรียกว่าเป็นเครื่องควบคุมประยุกต์ ในระดับเครื่องควบคุมพื้นฐานใช้ไมโครคอนโทรลเลอร์ ตระกูล ARM7TDMI-S จำเป็นต้องอาศัยความรู้ความเข้าใจในหลักการทำงานของคอนโทรลเลอร์สมองกลฝังตัว มีการประมวลผลข้อมูลได้ทีละขนาด 32 บิต ใช้ไอซีเบอร์ LPC2378 เป็นตัวควบคุมการทำงานหลัก สำหรับ โปรแกรมหรือซอฟแวร์ที่ใช้เป็นเครื่องมือในการพัฒนาจะเป็น โปรแกรมตัวแปรภาษา C ตามมาตรฐาน ANSI (ANSI Standard C) ส่วนเครื่องควบคุมประยุกต์อย่าง PLC จำเป็นต้องมีความรู้ และหลักการทำงานของเครื่องควบคุม PLC เป็นอย่างดี เพื่อจะได้ออกแบบวงจรอิเลคทรอนิคส์และแผ่นวงจรพิมพ์ให้ระบบมีสัญญาณรบกวนจากภายนอกต่ำ เครื่องควบคุมมีเสถียรภาพการทำงานสูง

1.5 ขอบเขตการวิจัย

การออกแบบพัฒนา คอนโทรลเลอร์แบบสมองกลฝังตัว ให้เป็นเครื่องควบคุม PLC ที่สามารถเชื่อมต่อกับโฮสคอมพิวเตอร์ผ่านพอร์ท Ethernet ในระบบเครือข่ายแบบอินเทอร์เน็ตในรูปแบบเว็บบราวเซอร์ การออกแบบแบ่งเป็นส่วนของฮาร์ดแวร์ และ ซอฟท์แวร์บนระบบปฏิบัติการของเครื่องควบ PLC โดยมีโปรแกรม IL (Instruction List) เป็นโปรแกรมประยุกต์ของผู้ใช้งานที่ทำงานภายใต้ระบบปฏิบัติการเครื่องควบคุม PLC

1.6 ขั้นตอนของการศึกษา

วิทยานิพนธ์ฉบับนี้ได้แบ่งเนื้อหาออกเป็น 6 บทด้วยกันคือ

บทที่ 1 กล่าวถึงความเป็นมาของงานวิจัย ความมุ่งหมายและวัตถุประสงค์ สมมติฐาน ทฤษฎีที่ใช้ ขอบเขตของการวิจัย และ ขั้นตอนการศึกษา

บทที่ 2 กล่าวถึงทฤษฎีพื้นฐานของเทคโนโลยีสมองกลฝังตัว (Embedded Technology) โดยกล่าวถึงสถาปัตยกรรมพื้นฐานทางด้านฮาร์ดแวร์และซอฟท์แวร์ของสมองกลฝังตัว รวมถึงการติดต่อการสื่อสารข้อมูลและเทคโนโลยีระบบเครือข่าย

บทที่ 3 การออกแบบฮาร์ดแวร์ของเครื่องควบคุม PLC โครงสร้างทางฮาร์ดแวร์ การออกแบบ การสื่อสารข้อมูลในรูปแบบต่างๆและการออกแบบวงจรทางด้านอินพุต-เอาท์พุต แบบดิจิตอล และการทำงานส่วนอื่นๆอีกเป็นต้น

บทที่ 4 การออกแบบทางซอฟท์แวร์ของเครื่องควบคุม PLC กล่าวถึงหลักการทำงานทั่วไปของ PLC โครงสร้างการทำงานระบบปฏิบัติการ RTOS และการออกแบบฐานข้อมูลของชุดคำสั่ง PLC และชุดคำสั่งในการติดต่อสื่อสารข้อมูลแบบต่างๆ

บทที่ 5 หาสมรรถนะของ Embedded PLC โดยการทดสอบการทำงานโปรแกรมประยุกต์หาค่าสแกนไทม์หรือเวลาในการตอบสนองต่ออินพุทเอาท์พุทของเครื่องควบคุม การทดสอบประสิทธิภาพการทำงานของเครื่องควบคุม ทดสอบประสิทธิภาพของแสกนไทม์ตามจำนวนคำสั่งทำและวัดเวลาในการสแกนไทม์เมื่อมีการ Online และ Offline อีเธอร์เน็ต พอร์ท

บทที่ 6 สรุปผลการวิจัยและข้อเสนอแนะ

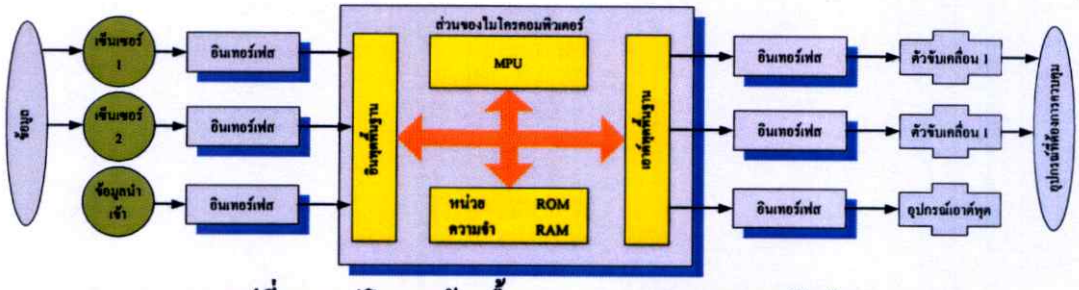
บทที่ 2
ทฤษฎีพื้นฐานและสถาปัตยกรรมของสมองกลฝังตัว

ในหัวข้อนี้จะกล่าวถึงทฤษฎีพื้นฐานต่างๆ ที่เกี่ยวข้องกับสมองกลฝังตัว เริ่มตั้งแต่คำจำกัดความและความหมายของสมองกลฝังตัว การประยุกต์เอาสมองกลฝังตัวไปใช้งานกับอุปกรณ์ควบคุมประเภทใดบ้าง สถาปัตยกรรมพื้นฐานทางด้านฮาร์ดแวร์ของสมองกลฝังตัวว่ามีส่วนประกอบสำคัญอะไรบ้างและมีความสำคัญอย่างไร เทคโนโลยีพื้นฐานทางด้านซอฟท์แวร์ของสมองกลฝังตัว เช่นการประมวลผลแบบเรียลไทม์ การอินเทอร์รัพต์ และการสื่อสารข้อมูลและเทคโนโลยีระบบโครงข่าย ของสมองกลฝังตัว

2.1 สถาปัตยกรรมพื้นฐานทางด้านฮาร์ดแวร์ของสมองกลฝังตัว

ระบบสมองกลฝังตัวเป็นระบบคอมพิวเตอร์ที่ออกแบบมาเพื่อควบคุมการทำงานของอุปกรณ์หรือเครื่องจักรต่างๆ โดยเชื่อมโยงหรือฝังตัวอยู่กับอุปกรณ์เหล่านั้น ซึ่งจะต้องมีการกำหนดคุณลักษณะทางด้านฮาร์ดแวร์และซอฟแวร์อย่างเหมาะสม ดังนั้น สำหรับอุปกรณ์หรือเครื่องจักรที่แตกต่างกัน ก็จะทำให้ข้อกำหนดเกี่ยวกับคุณลักษณะของระบบสมองกลฝังตัวนั้นแตกต่างกันออกไปด้วย ในส่วนของฮาร์ดแวร์ ผู้ออกแบบจะต้องวางโครงสร้างให้รองรับการทำงานของซอฟต์แวร์ได้ และควรจะมองเผื่อไปถึงการขยายขีดความสามารถของระบบต่อไปในอนาคตด้วย โดยควรจะวางโครงสร้างที่ง่ายต่อการขยายระบบ ถึงแม้ว่าจะมีความแตกต่างของระบบสมองกลฝังตัวตามสเกลของระบบที่ใช้ แต่โครงสร้างพื้นฐานจะคล้ายคลึงกัน ซึ่งในบทนี้จะอธิบายถึงองค์ประกอบของโครงสร้างพื้นฐานของระบบสมองกลฝังตัว

สถาปัตยกรรมพื้นฐาน (Basic architecture) ความต้องการของสถาปัตยกรรมของระบบสมองกลฝังตัวปกติจะแตกต่างกันไปตามลักษณะของการประยุกต์ใช้ซึ่งสามารถสรุปโครงสร้างพื้นฐานของฮารด์แวร์โดย (โครงสร้างของฮาร์ดแวร์) ได้ดังรูปที่ 2.1



รูปที่ 2.1 รูปโครงสร้างพื้นฐานของระบบสมองกลฝังตัว

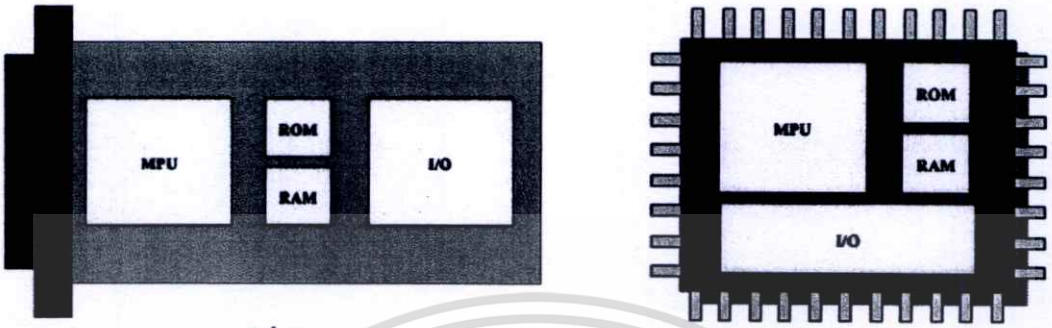
ส่วนแรกและเป็นหัวใจสำคัญของระบบ คือ ไมโครคอมพิวเตอร์ (microcomputer)เป็นส่วนที่มีหน่วยประมวลผลและหน่วยความจำซึ่งเป็นส่วนสำคัญที่สุดของระบบ ประกอบด้วย CPU หน่วยความจำรอมและแรม (ROM, RAM) อินพุตเอาต์พุตพื้นฐาน (basic I/O) ส่วนของ CPU เป็นส่วนที่จะกำหนดความเร็วในการประมวลผลและกำหนดความสามารถของระบบ ส่วน ROM เป็นหน่วยความจำสำหรับเก็บข้อมูลที่ไม่มีความจำเป็นต้องมีการเปลี่ยนแปลง เช่น โค๊ดของโปรแกรม (program code) และข้อมูลตัวแปรแบบกำหนดตายตัว ส่วน RAM เป็นหน่วยความจำสำหรับ CPU ให้ทำงานตามโปรแกรม เช่น สแตก เป็นต้น โดยจะทำการเก็บหรือเขียนทับค่าของตัวแปรที่ใช้ในงานระหว่างการทำงาน หรือเป็นบัฟเฟอร์สำหรับตัวแปรหรือข้อมูลต่างๆ โดยจะต้องมีขนาดไม่เล็กไปกว่าขนาดที่จำเป็นสำหรับการใช้งาน ส่วนอินพุตพื้นฐานรวมถึงการเชื่อมต่อเซ็นเซอร์ และอุปกรณ์อินพุตต่างๆเป็นส่วนรับข้อมูลและสัญญาณต่างๆของไมโครคอมพิวเตอร์ โดยจะส่งต่อข้อมูลเหล่านี้ในรูปของข้อมูลดิจิตอลให้กับ CPU เพื่อนำไปประมวลผลต่อ โดยทั่วไปเรียกว่า อินพุตพอร์ต โดยทั่วไประยะเวลาที่ CPU ทำการอ่านข้อมูลจากอินพุตพอร์ตจะสั้นมาก ในกรณีที่มีสัญญาณต่างๆ ที่จะต้องผ่านอินพุตพอร์ตจำนวนมาก จะต้องมีการสวิตช์ระหว่างการอ่านอินพุตพอร์ตแต่ละช่องสัญญาณ ซึ่งการปรับจังหวะเวลาให้เข้ากับจังหวะการทำงานของ CPU รวมทั้งการสวิตช์ระหว่างช่องสัญญาณต่างๆจะเป็นคุณสมบัติพื้นฐานของอินพุตพอร์ต

สำหรับเซ็นเซอร์เป็นอุปกรณ์ที่แปลงการเปลี่ยนแปลงเชิงกายภาพให้เป็นการเปลี่ยนแปลงเชิงสัญญาณไฟฟ้าอย่างไรก็ตาม ข้อมูลที่ส่งออกมาจากเซ็นเซอร์อาจเป็นค่าความเปลี่ยนแปลงของความต้านทาน หรือเป็นการเปลี่ยนแปลงของแรงดันไฟฟ้าระดับอ่อนๆ ซึ่งอินพุตพื้นฐานของไมโครคอมพิวเตอร์ไม่สามารถที่จะรับข้อมูลในลักษณะนี้ได้ จึงจำเป็นต้องมีอุปกรณ์ที่ทำหน้าที่เปลี่ยนสัญญาณจากรูปแบบของสัญญาณออกของเซ็นเซอร์ให้เป็นสัญญาณส่วนของอินพุตของไมโครคอมพิวเตอร์สามารถรับได้ วงจรไฟฟ้าทำหน้าที่นี้เรียกว่า วงจรอินเทอร์เฟส (Interfacing circuit) ซึ่งมีการพัฒนาขึ้นใช้งานอย่างแพร่หลาย

ส่วนของเอาต์พุตพื้นฐาน รวมถึงการเชื่อมต่อตัวขับเคลื่อน และอุปกรณ์เอาต์พุตต่างๆ เป็นช่องทางออกของสัญญาณที่เกิดจากผลการคำนวณของ CPU โดยมีการเปลี่ยนแปลงข้อมูลแบบดิจิตอลที่ได้นี้ให้เป็นสัญญาณทางกายภาพที่เหมาะสม เรียกว่า เอาต์พุตพอร์ต ในกรณีนี้ก็เช่นกัน ระยะเวลาที่ CPU ส่งข้อมูลออกไปยังเอาต์พุตพอร์ตนั้นเป็นช่วงเวลาสั้นๆ โดยอุปกรณ์เอาต์พุตจะต้องทำหน้าที่ในการเก็บข้อมูลนั้นเพื่อแปลงเป็นสัญญาณออกที่ต่อเนื่องและมีเสถียรภาพ นอกจากนี้ CPU เองยังต้องทำการติดต่อกับเอาต์พุตพอร์ตหลายช่องสัญญาณ ดังนั้นส่วนของเอาต์พุตพื้นฐานจึงต้องมีคุณสมบัติในการปรับจังหวะเวลาให้เหมาะสมกับการทำงานของ CPU และการสวิตช์ระหว่างเอาต์พุตพอร์ตหลายๆ ช่องสัญญาณเช่นเดียวกันกับในกรณีของอินพุตพื้นฐาน

ตัวขับเคลื่อนเป็นส่วนที่ทำหน้าที่แปลงการเปลี่ยนแปลงของสัญญาณไฟฟ้าให้เป็นการทำงานด้านกายภาพ ซึ่งเป็นส่วนสำคัญในการเชื่อมผลการคำนวณของ CPU กับโลกภายนอก ในการ

เชื่อมโยงสัญญาณออกของเอาต์พุตพอร์ตกับตัวขับเคลื่อนให้สามารถทำงานได้ตามที่ต้องการนั้น จำเป็นต้องมีวงจรเชื่อมต่อ ซึ่งในกรณีนี้วงจรเชื่อมต่อจะทำหน้าที่แปลงสัญญาณดิติตอลที่ออกจากเอาต์พุตพอร์ตให้เป็นกำลังไฟฟ้าสำหรับจ่ายให้กับตัวขับเคลื่อน



(ก)ไมโครคอมพิวเตอร์ที่เป็นบอร์ด (ข)ไมโครคอมพิวเตอร์แบบชิปเดียว

รูปที่ 2.2 รูปโครงสร้างของไมโครคอมพิวเตอร์

การวางโครงสร้างของระบบสมองกลฝังตัวสามารถทำได้หลายแบบ ดังรูปที่ 2.2 องค์ประกอบที่สำคัญ คือ การใช้ชิปไอซี โดยผู้ออกแบบอาจทำเป็นแผ่นลายวงจรพิมพ์ (print circuit board) ดังรูปที่ 2.2(ก) หรือวางโครงสร้างทั้งหมดภายในไมโครคอมพิวเตอร์แบบชิปเดียว (single chip microcomputer หรือ one chip microcomputer) ดังรูปที่ 2.2 (ข) หรือออกแบบในลักษณะผสมผสานการใช้งานของทั้งสองแบบ เช่น มีเพียงชิปหน่วยความจำ (RAM, ROM) ที่เพิ่มขึ้นมาเท่านั้น

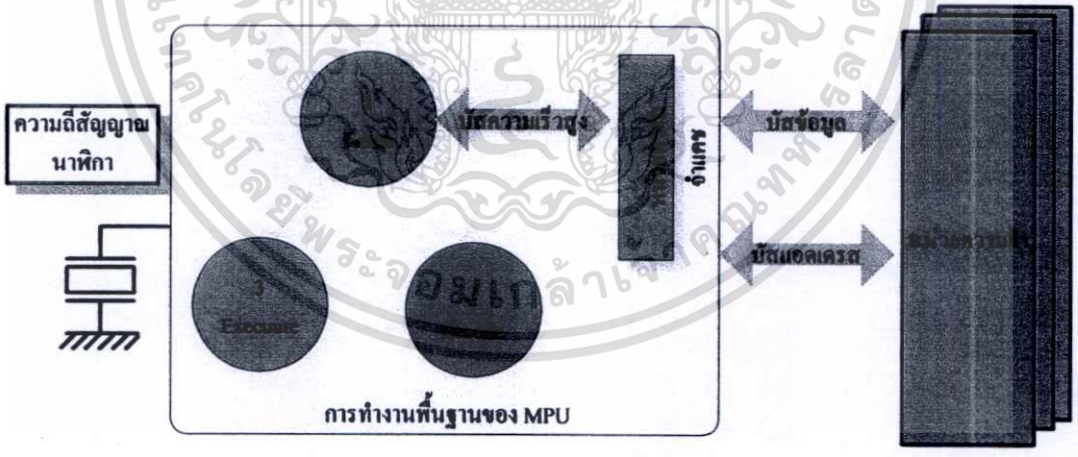
ในกรณีที่ต้องการไมโครคอมพิวเตอร์ที่มีความสามารถสูง (เช่น CPU ที่มีความเร็วสูงและฟังก์ชั่นการทำงานที่ซับซ้อนจะต้องการหน่วยความจำขนาดใหญ่ หรือต้องการอินพุตเอาต์พุตแบบพิเศษ) ก็อาจใช้โครงสร้างในลักษณะที่เป็นบอร์ด ซึ่งมีความเป็นอิสระในการออกแบบสูง หรือหากต้องการของที่ราคาถูก เหมาะกับการใช้ในผลิตภัณฑ์ที่ผลิตจำนวนมากๆ ก็อาจเลือกใช้ไมโครคอมพิวเตอร์แบบชิปเดียว ซึ่งในกรณีนี้เราสามารถเลือกไมโครคอมพิวเตอร์จากรุ่นต่างๆ ที่มีอยู่เพื่อให้เหมาะกับการใช้งานของเราได้ โดยสามารถเลือกได้ตั้งแต่ประเภทของ CPU ที่อยู่ภายในขนาดของหน่วยความจำ และประเภทของอินพุตเอาต์พุตพื้นฐานที่อยู่ภายในชิป เพื่อให้เหมาะสมกับลักษณะงานได้

โครงสร้างของระบบในรูปแบบต่างๆ ที่กล่าวมานั้น การที่จะเลือกเอารูปแบบใดรูปหนึ่งนั้น นอกจากจะพิจารณาจากความสามารถในการประมวลผลของระบบสมองกลฝังตัวที่จะพัฒนาแล้ว ยังต้องคำนึงถึงระยะเวลาในการพัฒนา ต้นทุนในการพัฒนา รวมถึงการผลิต บุคลากรที่จะเป็นผู้รับผิดชอบ และอุปกรณ์ต่างๆ ที่ต้องใช้ในการพัฒนาระบบด้วย และเนื่องจากในปัจจุบันขนาดของระบบเริ่มใหญ่ขึ้น ระยะเวลาในการพัฒนาก็จะต้องสั้นลง

รวมถึงต้นทุนในการพัฒนาที่จะต้องลดลงตลอดเวลา ทำให้จะต้องมีการออกแบบเพื่อให้สามารถนำส่วนต่างๆ ที่ได้เคยพัฒนาแล้วมาทำการใช้ซ้ำ รวมถึงจะต้องคำนึงถึงความอ่อนตัวของระบบในการขยายระบบในอนาคตอีกด้วย นอกจากนี้จำนวนชิปที่ใช้อยู่ในระบบสมองกลฝังตัวหนึ่งระบบก็ไม่ได้ถูกจำกัดอยู่แค่เพียงชิปเดียว บางครั้งการออกแบบระบบสมองกลฝังตัวที่ต้องการฟังก์ชั่นการทำงานที่ซับซ้อน หรือมีการประมวลผลมาก หรือฟังก์ชั่นระดับสูง ในการควบคุมก็อาจใช้ชิปไมโครคอมพิวเตอร์ชิปต่างๆ ที่ออกแบบมาเฉพาะสำหรับแต่ละงานรวมอยู่บนบอร์ดเดียวกันได้ เรียกการออกแบบนี้ว่า โครงสร้างแบบมัลติโพรเซสเซอร์ (Multiprocessor) นอกจากนี้ระบบที่ใหญ่กว่านั้นยังอาจใช้ไมโครคอมพิวเตอร์หลายตัวเชื่อมต่อกันโดยใช้ระบบ LAN แล้วทำการประมวลผลแบบกระจาย (distributed processing) รวมถึงการนำชิปหลายๆ ชิป มาประมวลผลการคำนวณ หรือการทำงานงานเดียวที่มีความซับซ้อนด้วยชิปหลาย ๆ ชิป เพื่อให้ได้กลุ่มของหน่วยประมวลผลที่มีความเร็วในการประมวลผลสูง หรือที่เรียกว่า การประมวลผลแบบขนาน (parallel processing)

2.1.1 หน่วยประมวลผลกลาง CPU

CPU ย่อมาจาก (Central processing Unit) เป็นส่วนที่หัวใจสำคัญของไมโครคอมพิวเตอร์ ในการกำหนดคุณสมบัติของระบบสมองกลฝังตัวนั้น การเลือก CPU เป็นปัจจัยสำคัญที่จะส่งผลให้ระบบมีความสามารถพอเพียงสำหรับการประมวลผลในการใช้งาน

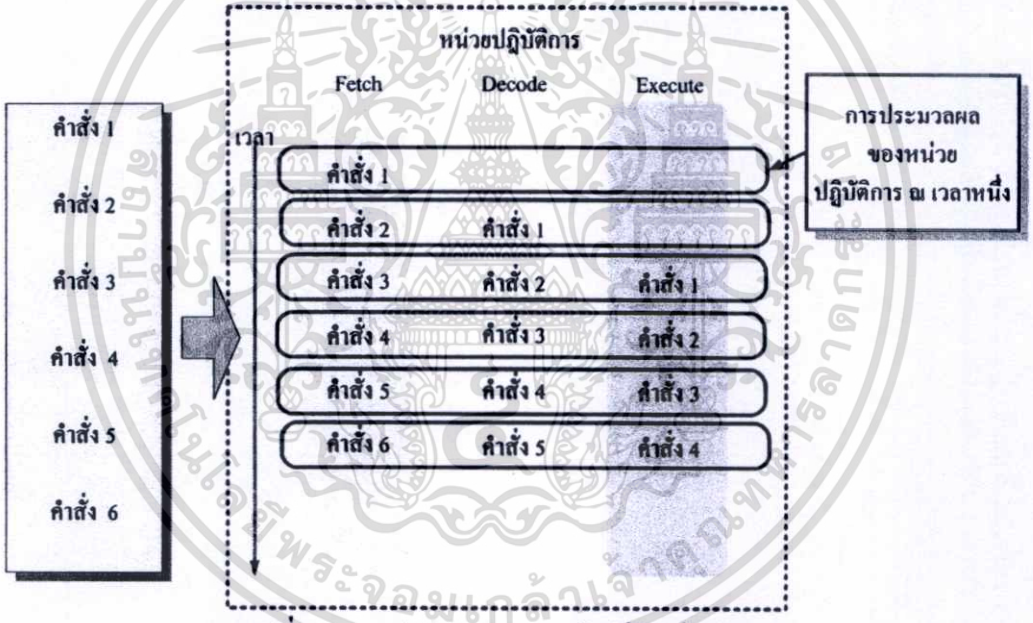


รูปที่ 2.3 รูปการทำงานพื้นฐานของ CPU

สำหรับคุณลักษณะสมบัติของ CPU ที่จะส่งผลกับความสามารถในการประมวลผล ได้แก่ จำนวนบิตของข้อมูลบนบัสข้อมูล ที่สามารถส่งไปได้บนบัสข้อมูลต่อคำสั่ง 1 คำสั่งของ CPU เช่น 8, 16 หรือ 32 บิต เป็นต้น ขนาดพื้นที่แอดเดรสของหน่วยความจำ (memory address space) ซึ่งเป็นตัวกำหนดว่า CPU สามารถเข้าถึงข้อมูลในหน่วยความจำเป็นปริมาณเท่าไร และความถี่สัญญาณนาฬิกา เป็นมาตรฐานในการวัดความเร็วในการประมวลผล เป็นต้น ดังแสดงดังรูปที่ 2.1

ปัจจุบัน CPU ส่วนมากมีการสังเคราะห์ความถี่สัญญาณนาฬิกาภายในตัวเอง ทำให้สามารถเพิ่มความถี่ในการประมวลผลให้สูงขึ้นได้มาก

ในส่วนของความกว้างบิตของบัสข้อมูล หากนำไปใช้ในการควบคุมโดยทำการ ON/OFF แต่ละบิตแค่ CPU ที่เป็น 8 บิตก็เพียงพอแล้ว แต่หากต้องการนำไปใช้ในการประมวลผลทางคณิตศาสตร์ที่ต้องการความละเอียดสูง ก็อาจใช้ CPU ที่มีความกว้างของบัสข้อมูลเป็น 32 บิต ส่วนการนำไปใช้ในการประมวลผลภาพนั้น นอกจากจะต้องมีพื้นที่ในหน่วยความจำขนาดใหญ่เพื่อเป็นที่เก็บข้อมูลแล้ว ความเร็วในการประมวลผลที่สูงก็เป็นองค์ประกอบที่จำเป็นด้วย ถ้าเป็นงานควบคุมตัวแปรที่มีการเปลี่ยนแปลงช้า เช่น อุณหภูมิ รวมถึงถ้าไม่จำเป็นต้องมีการบันทึกผลมากมายนัก ก็อาจใช้ CPU ที่ช้าลงก็ได้ จากการพิจารณาเช่นนี้ การเลือก CPU ให้เหมาะสมกับงานเพื่อใช้ในระบบสมองกลฝังตัวจะต้องสามารถตอบสนองต่อความต้องการ โดยอย่างน้อย CPU จะต้องมีคุณสมบัติเพียงพอที่จะสามารถทำงานนั้นได้ เป็นต้น



รูปที่ 2.4 การประมวลผลแบบไปป์ไลน์(pipeline)

การตอบสนองต่อความต้องการความสามารถในการประมวลผลที่สูงนั้น นอกจากจะเพิ่มความถี่สัญญาณนาฬิกาแล้ว การปรับเปลี่ยนวิธีการประมวลผลภายในของ CPU ก็มีผลกับความเร็วเช่นกัน โดยปกติแล้วการทำงานภายในของ CPU

จะเริ่มด้วยการอ่านคำสั่ง (fetch เป็นการอ่านคำสั่งจากหน่วยความจำเข้ามา) แล้วทำการแปลความหมาย (decode เป็นการตีความคำสั่ง) แล้วจึงทำตามคำสั่ง (execute เป็นการปฏิบัติตามคำสั่ง) ขั้นตอนทั้งสามนี้จะทำซ้ำกันไปเรื่อยๆ เป็นผลให้เกิดการประมวลผลของโปรแกรมทั้งชุด

การทำงานแบบไปป์ไลน์ (pipeline) จะมีส่วนช่วยให้การทำงานของยูนิตทั้งสามไม่ต้องมีการว่างการทำงานเนื่องจากต้องรอผลของส่วนอื่น โดยทำการประมวลผลของแต่ละส่วนไปพร้อมๆ

กันดังรูปที่ 2.4 วิธีนี้ทำให้มีความเร็วในการประมวลผลสูงขึ้น นอกจากนี้การเพิ่มความเร็วในการเคลื่อนย้ายข้อมูลระหว่าง CPU และหน่วยความจำก็สามารถช่วยให้ความเร็วในการประมวลผลสูงขึ้นได้เช่นกัน โดยนำหน่วยความจำที่มีความเร็วสูงที่เรียกว่า หน่วยความจำแคช วางไว้ระหว่าง CPU และหน่วยความจำหลัก เพื่อเพิ่มความเร็วในการอ่านคำสั่ง ซึ่งวิธีนี้เป็นวิธีที่ใช้กันอย่างแพร่หลาย อย่างไรก็ตาม วิธีนี้ไม่สามารถรองรับการทำงานของโปรแกรมที่มีการกระโดดหรือมีการทำงานแยกตามเงื่อนไขต่างๆ (branch) รวมทั้งมีจุดด้อยในเรื่องการประมาณการเวลาในการทำงานของ CPU นั้นทำได้ยาก เป็นต้น

นอกจากนี้ยังสามารถแบ่ง CPU ออกตามโครงสร้างภายในได้เป็น 2 ประเภทหลักๆ คือ ซิสก์ (CISC)และริสก์ (RISC) โดยโครงสร้างแบบ CISC (Complex Instruction Set Computer) เป็น CPU ที่ออกแบบมาให้มีคำสั่งในการทำงานที่ซับซ้อนจำนวนมาก เพื่อให้สามารถตอบสนองกับความต้องการในการโปรแกรมด้วยฟังก์ชั่นต่างๆ (เพื่อให้ใช้จำนวนคำสั่งน้อยลง) ในการทำงาน เป็นผลให้โครงสร้างภายในของ CPU มีความซับซ้อนสูง และใช้ความถี่สัญญาณนาฬิกาจำนวนหลายลูกในการปฏิบัติตามคำสั่ง นอกจากนี้ยังสามารถเข้าถึงหน่วยความจำเพื่ออ่านหรือเขียนค่าในการประมวลผลได้โดยตรง ไม่ต้องผ่านรีจิสเตอร์

ส่วน CPU ที่มีโครงสร้างแบบ RISC (Reduced Instruction Set Computer) ถูกออกแบบมาให้มีคำสั่งในการทำงานจำนวนน้อย และมีเพียงคำสั่งที่จำเป็นเท่านั้น ทำให้โครงสร้างภายในของ MPC ไม่ซับซ้อน และสามารถประมวลผลได้อย่างมีประสิทธิภาพมากขึ้น จากการที่คำสั่งในการทำงานเป็นคำสั่งง่ายๆ ทำให้สามารถทำงานได้อย่างเร็วเพียงแค่ไม่กี่สัญญาณนาฬิกา และสามารถใช้กับสัญญาณนาฬิกาที่ความถี่สูงมากๆ ได้ นอกจากนี้การทำงานยังถูกออกแบบมาให้ใช้รีจิสเตอร์ภายใน จึงต้องมีการถ่ายเทข้อมูลจากหน่วยความจำไปไว้ในรีจิสเตอร์ก่อน จึงจะประมวลผลได้ ทำให้ทำงานได้เร็วมาก

ในการสร้างโปรแกรมให้สามารถทำงานได้อย่างที่ต้องการจากคำสั่งพื้นฐานนั้น จำเป็นจะต้องมีหน่วยประมวลผลทางภาษาที่มีสมรรถนะสูง เช่น ตัวแปลภาษาหรือคอมไพเลอร์ และการพัฒนาของคอมไพเลอร์นี้ก็จะเป็นประโยชน์ต่อการใช้งาน CPU ที่เป็นแบบ CISC ทั้งนี้เนื่องจากสมัยก่อนเวลาที่ใช้ในการเข้าถึงหน่วยความจำยังช้าอยู่ ทำให้มีแนวคิดที่จะลดจำนวนครั้งของการอ่านข้อมูลจากหน่วยความจำ โดยการทำให้คำสั่งในการทำงานซับซ้อนขึ้น ต่อมาภายหลังได้มีการพัฒนาหน่วยความจำให้สามารถเข้าถึงได้เร็วขึ้น โครงสร้างแบบ RISC จึงเริ่มเป็นที่นิยมกันมากขึ้น

โดยทั่วไปผู้พัฒนาโปรแกรมนิยมใช้ภาษาระดับสูง เช่น ภาษาซี ในการเขียนโปรแกรม ดังนั้นส่วนที่จะมีผลต่อการพัฒนาระบบมาก ได้แก่ เครื่องมือที่ใช้ในการพัฒนา เช่น โปรแกรมแปลภาษาหรือคอมไพเลอร์ และเครื่องมือต่างๆ ที่ใช้ในการดีบัก รวมถึงเรียลไทม์โอเอสและมิดเดิลแวร์ต่างๆ ด้วย ดังนั้น การพิจารณาเลือก CPU ในการพัฒนาระบบสมองกลฝังตัว ก็ควรจะพิจารณา

เครื่องมือที่ใช้ในการพัฒนาร่วม เช่น คอมไพเลอร์ และ เครื่องมืออื่นๆ ด้วยว่ามีหรือไม่ และความสามารถเป็นอย่างไร

2.1.2 ระบบบัสตามมาตรฐาน ISO และ IEEE

บัส (Bus) สามารถแบ่งออกตามตำแหน่งที่ใช้งานได้เป็น 2 ประเภทหลักๆ ได้แก่ บัสภายใน (internal bus) และบัสภายนอก (external bus) บัสภายในเป็นกลุ่มของสายสัญญาณภายในไมโครคอนโทรลเลอร์ที่ใช้ร่วมกันสำหรับส่งผ่านข้อมูลระหว่าง CPU หน่วยความจำ และอินพุตเอาต์พุตพื้นฐาน บัสภายนอกเป็นกลุ่มของสายสัญญาณเพื่อส่งผ่านข้อมูลระหว่างไมโครคอมพิวเตอร์และอุปกรณ์ภายนอกต่างๆ และมีการแยกประเภทการใช้งานตามประเภทของอุปกรณ์ภายนอกและวัตถุประสงค์การใช้งาน

สำหรับมาตรฐานของบัส นอกจากที่ ISO และ IEEE กำหนดไว้แล้ว ยังมีที่ถูกกำหนดโดยผู้ผลิตรายต่างๆ ที่นำมาใช้กับผลิตภัณฑ์ของตนเองจนกลายเป็นมาตรฐานในตลาด (de facto standard) จะเห็นได้ว่ามาตรฐานของบัสมีการพัฒนาไปตามการเปลี่ยนแปลงของ CPU ที่เร็วขึ้นมีการประมวลผลที่หลากหลายขึ้น หน่วยความจำมีขนาดใหญ่ขึ้น และมีการประยุกต์ใช้งานที่หลากหลายขึ้น

สำหรับกลุ่มของสายสัญญาณที่เป็นบัสภายในสามารถแบ่งออกได้เป็น 3 ประเภทหลักๆ คือ แอดเดรสบัส (address bus) บัสข้อมูล (data bus) และบัสควบคุม (control bus) ในส่วนของแอดเดรสบัสจะใช้ในการระบุดีไวซ์หรืออุปกรณ์ และข้อมูลจะถูกแลกเปลี่ยนผ่านบัสข้อมูล เพื่อทำให้เกิดการระบุและแลกเปลี่ยนข้อมูล ทั้งหมดนี้ถูกกำหนดผ่านบัสควบคุม

จำนวนบิตของแอดเดรสบัสจะเป็นตัวกำหนดขนาดของพื้นที่แอดเดรสที่บัสนั้นสามารถอ้างอิงถึงได้ ซึ่งรวมไปถึงขนาดของโปรแกรมที่เก็บอยู่ในหน่วยความจำรอมเพื่อการประมวลผล ขนาดของแอดเดรสที่ใช้ระหว่างการทำงานในลักษณะของสะแตก พื้นที่ที่เป็นตัวแปร และบัฟเฟอร์ข้อมูลที่สามารถอ้างอิงถึงได้ ในส่วนของบัสข้อมูล ถ้าจำนวนบิตของบัสข้อมูลมาก ก็หมายถึงข้อมูลที่สามารถส่งได้ในแต่ละครั้งมีความขนาดใหญ่ขึ้น (เท่ากับจำนวนบิตของบัสข้อมูล) ซึ่งหากจำนวนบิตมาก ก็หมายความว่าสามารถส่งข้อมูลได้มากขึ้น และเร็วขึ้นนั่นเอง นอกจากนี้บาง CPU อาจมีจำนวนบิตที่ใช้ในการประมวลผลจริงกับขนาดของบัสข้อมูลที่แตกต่างกัน เนื่องจากข้อจำกัดเกี่ยวกับขา (pin) ของชิปไอซี เช่น บาง CPU มีการประมวลผลภายในหน่วยของ 16 บิต แต่มีบัสข้อมูลขนาดเพียง 8 บิต เป็นต้น

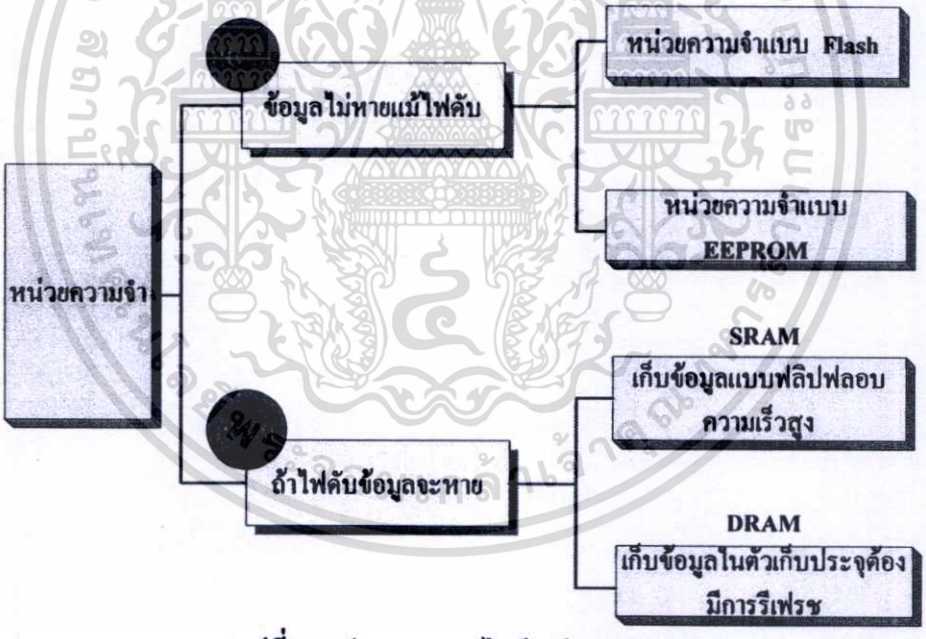
นอกจากนี้บางไมโครคอมพิวเตอร์ก็มีจำนวนบิตของบัสข้อมูลและบัสแอดเดรสมากขึ้น แต่มีข้อจำกัดด้านกายภาพของแพ็กเกจของชิปไอซี ทำให้ต้องมีการใช้ขาบางส่วนของบัสข้อมูลและบัสแอดเดรสร่วมกัน โดยทำการมัลติเพลกซ์ด้านเวลาโดยในช่วงต้นของไซเคิล บัสจะมีการทำงานและให้สัญญาณออกเป็นบัสแอดเดรสก่อนหลังจากนั้นจึงจะทำงานเป็นบัสข้อมูล

ซึ่งโดยทั่วไปผู้ออกแบบมักจะแนะนำให้ใช้ชิปไอซีที่เป็นตระกูลเดียวกันกับชิปไมโครคอมพิวเตอร์เพื่อสนับสนุนการทำงานของบัสทั้งสองนี้ อย่างไรก็ตาม ผู้ใช้งานก็สามารถเลือกชิปไอซีที่มีอยู่ทั่วไปได้ ในกรณีหลังนี้ผู้ใช้งานจะต้องใช้วงจรแลตช์ (latch) เพื่อรักษาสัญญาณที่เป็นแอดเดรสไว้ระหว่างทำงาน

สำหรับบัสภายนอกนั้นมีทั้งบัสทั่วไปและบัสแบบที่ระบุอุปกรณ์ที่จะมาต่อเชื่อมด้วย โดยบัสแต่ละประเภทจะมีการกำหนดคุณสมบัติทางไฟฟ้า (เช่น ระดับแรงดันไฟฟ้าของสัญญาณ เป็นต้น) ลักษณะสมบัติทางกายภาพ (เช่น โครงสร้างของจุดเชื่อมต่อหรือคอนเนกเตอร์ หรือขนาดที่เป็นมาตรฐานต่างๆ) โดยบัสต่างๆ ที่เป็นมาตรฐานนั้นก็จะมีชิปไอซีที่สามารถทำงานตามมาตรฐานนั้นๆ ซึ่งผู้ใช้งานก็จะต้องศึกษาวิธีใช้และการต่อเชื่อมกับ CPU เพื่อให้สามารถนำมาใช้งานได้อย่างถูกต้อง

2.1.3 หน่วยความจำ

หน่วยความจำเป็นที่สำหรับเก็บรักษาข้อมูลข่าวสาร ได้แก่ โปรแกรมและข้อมูล ที่อยู่ภายในไมโครคอมพิวเตอร์ สำหรับสมองกลฝังตัวซึ่งประกอบด้วยส่วนที่เป็นฮาร์ดแวร์และส่วนที่เป็นซอฟต์แวร์นั้น ที่เก็บซอฟต์แวร์ก็คือหน่วยความจำนั่นเอง



รูปที่ 2.5 ประเภทของไอซีหน่วยความจำ

หน่วยความจำที่เป็นสารกึ่งตัวนำแบ่งได้เป็น 2 ประเภทใหญ่ๆ ดังรูปที่ 2.5 ได้แก่ หน่วยความจำที่เก็บข้อมูลแบบไม่สามารถเปลี่ยนแปลงแก้ไขได้ (คือ ROM) และหน่วยความจำที่สามารถเปลี่ยนแปลงแก้ไขได้อย่างอิสระ รวมถึงการที่ข้อมูลอาจหายไปเนื่องจากไฟดับ (ก็คือ RAM) ในการใช้งานจริงเรามักจะใช้ ROM เพื่อจัดเก็บโค๊ดโปรแกรมที่ไม่มีการเปลี่ยนแปลง

และใช้ RAM เป็นหน่วยความจำที่เก็บค่าตัวแปรต่างๆ หรือสแตกที่มักมีการเปลี่ยนแปลงไปตามจังหวะการทำงานของระบบ โดยหน่วยความจำทั้งสองส่วนนี้ล้วนมีความจำเป็นสำหรับการทำงานของระบบสมองกลฝังตัวด้วยกันทั้งนั้น

2.1.4 หน่วยอินพุทเอาท์พุทพื้นฐาน

ในระบบสมองกลฝังตัวนั้น CPU จะรับข้อมูลข่าวสารจากภายนอกเข้ามาและทำการคำนวณและประมวลนั้นก็ที่จะส่งผลนั้นออกไปภายนอกอีกครั้งซึ่งจะมีการแลกเปลี่ยนข้อมูลระหว่าง CPU กับภายนอกเพื่อให้สามารถทำได้ตามที่กำหนด การแลกเปลี่ยนข้อมูลระหว่าง CPU กับภายนอกจำเป็นต้องมีส่วนของการเชื่อมต่ออินพุตเอาต์พุต (I/O interface) เข้ามาเกี่ยวข้องในการปรับระดับแรงดันของสัญญาณไฟฟ้า หรือจังหวะเวลาในการแลกเปลี่ยนข้อมูลระหว่างการเชื่อมต่อกันนี้ ส่วนของการเชื่อมต่ออินพุตเอาต์พุตนี้จะประกอบด้วยส่วนของซอฟต์แวร์และส่วนของฮาร์ดแวร์รวมกันขึ้นเป็นโครงสร้างที่ลักษณะต่างๆมากมาย ดังนี้ จำเป็นจะต้องมีความรู้ครอบคลุมทั้งสองส่วน ในบทนี้จะขอกล่าวถึงฮาร์ดแวร์ที่จำเป็นในการอินเทอร์เฟส ซึ่งได้แก่ I/O แบบขนาน ,I/O แบบอนุกรม,และอนาลือก I/O ซึ่งเป็น I/O พื้นฐานที่สำคัญ

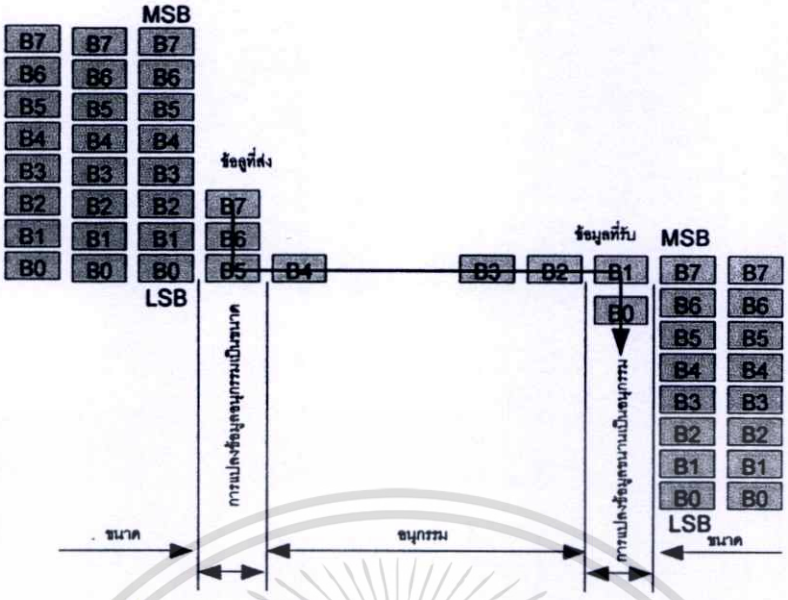
อินพุตเอาต์พุตแบบอนุกรม (Serial I/O)

อินพุตเอาต์พุตแบบอนุกรมเป็นการเชื่อมต่อโดยสัญญาณเส้นเดียวในรับส่งข้อมูล ซึ่งใช้สายสัญญาณจำนวนน้อย จึงเป็นที่นิยมใช้กัน โดยจะเรียกช่วงต่อสำหรับการเชื่อมต่อสายสัญญาณในการใช้งานอินพุตเอาต์พุตแบบอนุกรมนี้ว่า "พอร์ตอนุกรม (serial port)"

ข้อดีของการเชื่อมต่ออินพุตเอาต์พุตแบบอนุกรมนี้คือ ใช้จำนวนสัญญาณน้อย ทำให้ต้นทุนของระบบลดลง แต่มีข้อเสียคือ หากต้องการเพิ่มความเร็วในการรับส่งข้อมูล ก็จะต้องเพิ่มความถี่สัญญาณนาฬิกาขึ้นซึ่งปัจจุบันความถี่ของระบบที่ใช้กันสูงถึงหลายร้อย Mbps (Megabit per second : ล้านบิตต่อวินาที) ถือเป็นความถี่สูงมาก การรับส่งสัญญาณความถี่สูงจะต้องมีการออกแบบบอร์ดและการวางตำแหน่งสายสัญญาณที่ดีเพื่อไม่ให้รูปทรงของสัญญาณเสียไป

วงจรเชื่อมต่อสัญญาณของอินพุตเอาต์พุตแบบอนุกรมเป็นวงจรที่แปลงข้อมูลแบบขนานที่ส่งออกมากจาก CPU ให้กลายเป็นข้อมูลแบบอนุกรม โดยมีการแปลงทีละบิต เริ่มจากบิตล่างสุดคือ LSB (Least Significant Bit) หรือบิตบนสุด MSB (Most Significant Bit) ให้เป็นข้อมูลบนสายสัญญาณเพียงหนึ่งเส้นการประมวลผลลักษณะนี้เรียกว่า การแปลงข้อมูลขนานอนุกรม (parallel-sequential conversion)

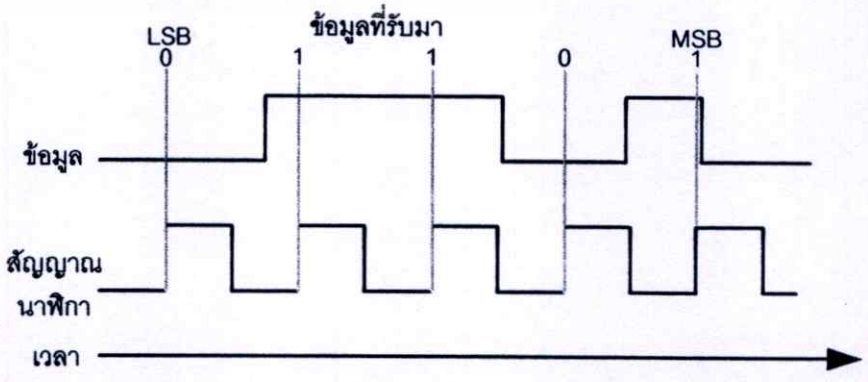
ด้านซ้ายของรูปที่ 2.6 วงจรสำหรับการเชื่อมต่อสัญญาณเข้าจะทำการแปลงข้อมูลทีละบิตที่ถูกส่งมายังอินพุตให้กลายเป็นข้อมูลแบบขนาน (เรียงกันให้ครบจำนวนบิตของข้อมูลแบบขนาน) เรียกการประมวลผลในลักษณะนี้ว่าการแปลงข้อมูลอนุกรมเป็นข้อมูลขนาน (sequential-parallel conversion) ด้านขวาของรูปที่ 2.6



รูปที่ 2.6 วงจรเชื่อมต่อสัญญาณอินพุตเอาต์พุตแบบอนุกรม

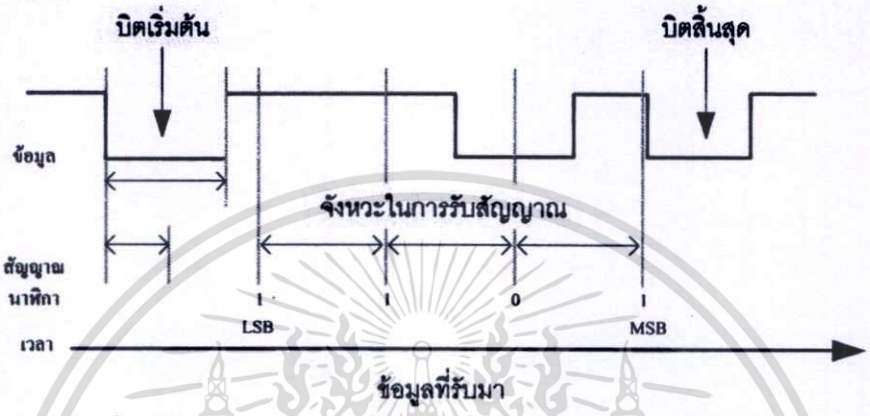
โดยทั่วไปแล้วการรับส่งข้อมูลแบบอนุกรมจะใช้สายสัญญาณสำหรับส่งและรับข้อมูลคนละเส้นแยกจากกัน แล้วส่งข้อมูลทีละบิต หากใช้กราวด์ร่วมกัน ก็จะใช้สายสัญญาณทั้งหมดเพียง 3 เส้น สำหรับการรับส่งข้อมูล นอกจากนี้อาจมีการใช้สายสัญญาณเพิ่มเพื่อทำการตรวจสอบสถานะของฝั่งรับและฝั่งส่ง รวมไปถึงอาจมีการกำหนดรูปแบบการรับส่งข้อมูลว่าเป็นแบบซิงโครนัส (synchronous) หรืออะซิงโครนัส (asynchronous) เป็นแบบ SDLC (synchronous data link control) หรือ HDLC (high-level data link control) ต่างๆเป็นต้น

การสื่อสารแบบอนุกรมอย่างซิงโครนัส (synchronous serial communication) จะส่งข้อมูลทีละบิตโดยอาจเริ่มจาก LSB (least significant bit) หรือ MSB (most significant bit) และส่งจนครบทั้งหมดวิธีในการซิงโครไนซ์สัญญาณจะต้องมีการตกลงกันระหว่างฝั่งรับและฝั่งส่งตั้งแต่แรก โดยวิธีพื้นฐานที่นิยมใช้กันในการส่งข้อมูลแบบซิงโครนัส ด้วยการใช้สัญญาณนาฬิกาเป็นตัวช่วยในการซิงโครไนซ์สัญญาณ โดยวิธีจะต้องใช้สายสัญญาณ 2 เส้นที่อิสระต่อกัน เส้นหนึ่งเป็นสัญญาณนาฬิกา และอีกเส้นหนึ่งเป็นข้อมูล ดังรูปที่ 2.7



รูปที่ 2.7 จังหวะเวลาของของการสื่อสารแบบอนุกรรมอย่างซิงโครนัส

การสื่อสารแบบอนุกรมอย่างอะโครซิงโครนัส (asynchronous serial communication) ฝั่งส่งและฝั่งรับจะสามารถแบ่งแยกข้อมูลออกจากสัญญาณที่ติดต่อสื่อสารกันได้อย่างถูกต้องโดยไม่จำเป็นต้องมีสายสัญญาณนาฬิกาอีกเส้นหนึ่งต่างหาก แต่จะต้องมีสัญญาณในส่วนที่จะบอกการเริ่มต้นของข้อมูล คือ บิตเริ่มต้น (start bit) และสัญญาณในส่วนที่จะบอกการสิ้นสุดของข้อมูล คือ บิตสิ้นสุด (stop bit) ดังรูปที่ 2.8

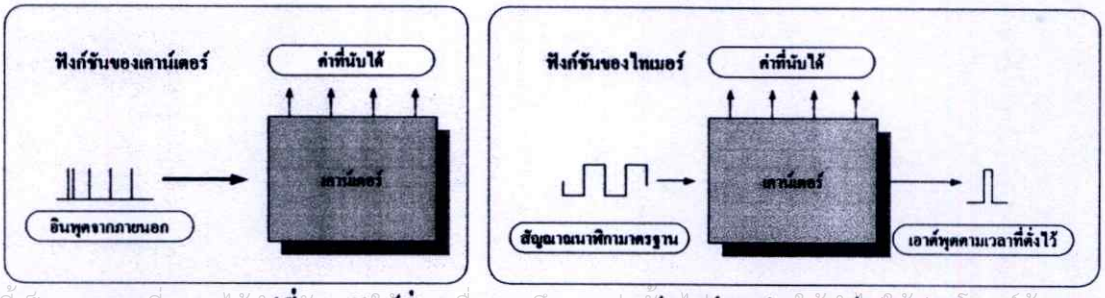


รูปที่ 2.8 จังหวะเวลาของการสื่อสารแบบอนุกรมอย่างอะซิงโครนัส

รูปแบบการอิมพลีเมนต์ช่องสัญญาณการสื่อสารที่ใช้ส่งข้อมูลอนุกรมอาจแตกต่างกันไปตามลักษณะเฉพาะเชิงไฟฟ้า หรือรูปร่างเชิงกล ปัจจุบันรูปแบบที่เป็นมาตรฐานได้แก่ มาตรฐาน RS-232C, RS-422, RS-485 และเคอร์เรนลูปแบบ 20 mA (20 mA current loop) เป็นต้น นอกจากนี้การส่งข้อมูลแบบไร้สายผ่านการเชื่อมต่อสัญญาณอินฟราเรด (IrDA) และมาตรฐานการส่งข้อมูลแบบ USB ก็เป็นที่นิยมกันในปัจจุบัน พอร์ตที่ใช้ในการสื่อสารแบบอนุกรมนิยมเรียกกันว่า พอร์ตคอม (COM port)

ตัวจับเวลา/ตัวนับ (Timer/Counter)

ตัวจับเวลาหรือไทเมอร์เป็นฟังก์ชั่นสำหรับสร้างสัญญาณที่สามารถกำหนดเวลาได้อย่างแม่นยำ ส่วนตัวนับหรือเคาน์เตอร์เป็นฟังก์ชั่นสำหรับนับจำนวนครั้งของการเปลี่ยนแปลงของสัญญาณอินพุต ทั้งสองฟังก์ชั่นจะมีโครงสร้างพื้นฐานที่เหมือนกันคือ ถูกสร้างมาจากวงจรนับเหมือนกัน ดังรูปที่ 2.9 ถ้าเรานำสัญญาณนาฬิกาที่รู้ค่าความถี่ที่แน่นอนมาป้อนให้กับอินพุตของวงจรนับ การนับพลส์ของสัญญาณนาฬิกาก็จะเทียบได้กับการจับเวลาที่แม่นยำ และเคาน์เตอร์ก็จะกลายเป็นไทเมอร์นั้นเอง



รูปที่ 2.9 ฟังก์ชั่นการทำงานของเคาน์เตอร์และไทเมอร์

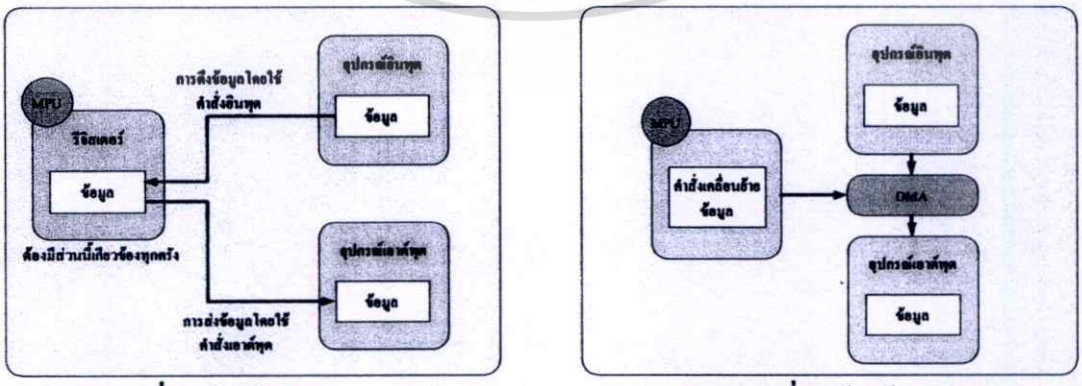
ฟังก์ชั่นของไทเมอร์สามารถสร้างขึ้นได้จากการใช้อินพุตเอาต์พุตแบบขนานร่วมกับการประมวลผลด้วยซอฟต์แวร์ ตัวอย่างเช่น หากเรามีรูทีนของโปรแกรม (program routine) ที่รู้เวลาการทำงานของมันอย่างแน่ชัดการรันรูทีนโปรแกรมนี้ซ้ำๆ ก็เปรียบเสมือนการนับเวลานั่นเอง กล่าวคือการนับจำนวนครั้งของการสั่งรูทีนให้ทำงานซ้ำจะทำให้เกิดไทเมอร์ ฟังก์ชั่นการทำงานของไทเมอร์ที่เกิดขึ้นจากวิธีการนี้ เรียกว่า ตัวจับเวลาชนิดซอฟต์แวร์หรือซอฟต์แวร์ไทเมอร์ (software timer) แต่การใช้งานซอฟต์แวร์ไทเมอร์จะทำให้ CPU ต้องทำงานตามขั้นตอนของโปรแกรมนั้นโดยไม่สามารถไปทำงานอื่นได้ ซึ่งถ้าหากมีอินเทอร์รัพต์เกิดขึ้น และ CPU ไปทำงานในการตอบสนองอินเทอร์รัพต์ เวลาก็ผิดเพี้ยนไป ดังนั้นจึงนิยมใช้ไอซีเฉพาะที่ทำหน้าที่เป็นไทเมอร์หรือเคาน์เตอร์ ซึ่งไมโครคอมพิวเตอร์แบบชิปเดียวที่ใช้กันในระบบฝังตัวปกติจะมีไทเมอร์และเคาน์เตอร์มาให้ด้วยโดยอาจจะมีมากกว่าหนึ่งตัว

2.1.5 เทคโนโลยีรอบข้างของ CPU (CPU Related Technology)

ภายในไมโครคอมพิวเตอร์ CPU จะเป็นศูนย์กลางการคำนวณและประมวลผลทั้งหมด ในหัวข้อนี้จะกล่าวถึงเทคโนโลยีที่เป็นส่วนสนับสนุนการทำงานเหล่านั้นซึ่งได้แก่ เทคโนโลยีอินเทอร์รัพต์เพื่อการประมวลผลแบบเรียลไทม์ เทคโนโลยี DMA (Direct Memory Access) สำหรับการเคลื่อนย้ายข้อมูลด้วยความเร็วสูง เทคโนโลยีหน่อยความจำแคช (cache memory technology) เพื่อการเข้าถึงข้อมูลในหน่วยความจำด้วยความเร็วสูง

และเทคโนโลยีหน่วยความจำเสมือน (virtual memory technology) ที่ช่วยให้โปรแกรมขนาดใหญ่สามารถทำงานได้ในระบบที่มีหน่วยความจำน้อย ซึ่งความรู้ในเทคโนโลยีต่างๆเหล่านี้จะช่วยให้ผู้พัฒนาสามารถเลือกประยุกต์ใช้กับงานของตนได้อย่างเหมาะสม

DMA (Direct Memory Access) เป็นการเคลื่อนย้ายข้อมูลระหว่างหน่วยความจำและอุปกรณ์อินพุตเอาต์พุตหรือระหว่างหน่วยความจำด้วยกันโดยใช้ฮาร์ดแวร์ควบคุมและไม่จำเป็นต้องใช้การประมวลด้วยโปรแกรมของ CPU ดังรูปที่ 2.10



(ก) การเคลื่อนย้ายข้อมูลระหว่างโปรแกรม (ข) การเคลื่อนย้ายด้วยวิธี DMA

รูปที่ 2.10 แสดงการเคลื่อนย้ายข้อมูลด้วยวิธี DMA

ภาษาทางคอมพิวเตอร์จะเรียกช่องทางที่ข้อมูลวิ่งผ่านว่า ช่องสัญญาณ (channel) และเรียกอุปกรณ์ฮาร์ดแวร์ที่ทำหน้าที่เคลื่อนย้ายข้อมูลโดยตรงจากหน่วยความจำโดยไม่ผ่าน CPU ว่า อุปกรณ์ช่องสัญญาณดีเอ็มเอ DMA channel หลังจากที่ DMA ได้รับคำสั่งเกี่ยวกับตำแหน่งและข้อมูลที่จะการเคลื่อนย้ายจาก CPU แล้ว DMA จะมีการทำงานที่เป็นอิสระจาก CPU และสามารถทำการเคลื่อนย้ายข้อมูล โดยระหว่างนั้น CPU จะไม่เข้ามายุ่งเกี่ยวกับขั้นตอนของ DMA การเคลื่อนย้ายข้อมูลด้วย DMA นี้จะมีความเร็วสูงกว่าที่ทำโดย CPU และเป็นการลดโหลด CPU ด้วย

2.2 เทคโนโลยีพื้นฐานทางด้านซอฟท์แวร์ของสมองกลฝังตัว

จุดประสงค์ของซอฟแวร์สมองกลฝังตัวก็คือ การเพิ่มขีดความสามารถหรือฟังก์ชั่นการทำงานของอุปกรณ์สมองกลฝังตัว หรืออุปกรณ์ที่ใส่เข้าไปในระบบที่ใช้เทคโนโลยีสมองกลฝังตัวลายประเภทและรูปแบบ ตั้งแต่อุปกรณ์ในดาวเทียม โดยปรกติอุปกรณ์สมองกลฝังตัวจะมีหลากหเครื่องบิน หรือในผลิตภัณฑ์ของผู้บริโภค เช่น กล้องดิจิตอล ระบบนำทางรถยนต์ เป็นต้น ไปจนถึงการควบคุมเครื่องใช้ไฟฟ้าต่างๆ เช่น หม้อหุงข้าว ตู้เย็น เป็นต้น สิ่งเหล่านี้จะมีอุปกรณ์สมองกลฝังตัวนั้นมีอยู่นับไม่ถ้วน

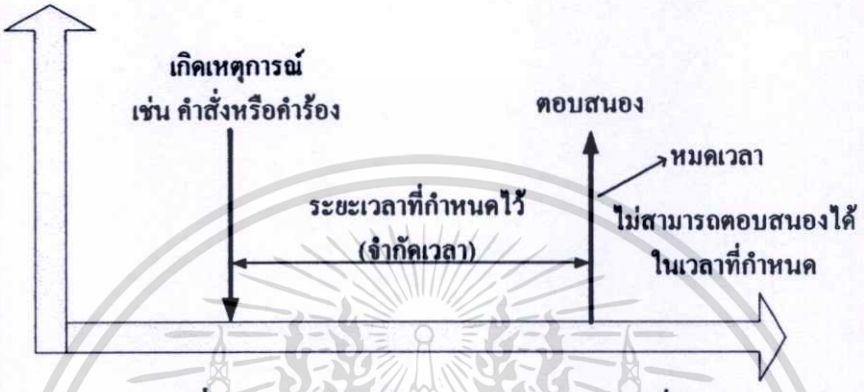
ยกเว้นอุปกรณ์สมองกลฝังตัวที่มีฟังช์ชั่นที่น้อยมากแล้ว อุปกรณ์สมองกลฝังตัวส่วนใหญ่จะเป็นระบบตอบสนองทันที หรือที่รู้จักกันในชื่อของ ระบบเรียลไทม์ (real-time system) ซึ่งจะรับสัญญานจากภายนอกและจะตอบสนองภายในเวลาที่กำหนด ระบบสมองกลฝังตัวส่วนใหญ่จะเป็นแบบเรียลไทม์ ซึ่งจะกลายเป็นลักษณะพิเศษอย่างหนึ่งของซอฟแวร์สมองกลฝังตัวระบบสมองกลฝังตัวโดยทั่วไปอุปกรณ์ที่ต่างกันมักต้องมีองค์ประกอบด้านฮาร์ดแวร์ของระบบแตกต่างกันออกไป เพราะอุปกรณ์เหล่านั้นจะมีรูปแบบและฟังก์ชั่นการใช้งานที่แตกต่างกันออกไปด้วย ดังนั้นซอฟแวร์สมองกลฝังตัวจึงถูกสร้างขึ้นบนพื้นฐานของฮาร์ดแวร์แพลตฟอร์มที่หลากหลายแตกต่างกันไปตามระบบ

2.2.1 การประมวลผลแบบเรียลไทม์ (Real-time processing)

โดยทั่วไประบบสมองกลฝังตัวจะรับคำร้องขอประมวลผลต่างๆ จากภายนอก ตัวอย่างของคำร้องขอประมวลผล ได้แก่ การเริ่มทำงานของอุปกรณ์ตามค่าที่อ่านได้จากเซ็นเซอร์ (sensor) การแสดงผลบนหน้าจอตามคำสั่งของผู้บังคับควบคุมเครื่องมือ การตอบสนองต่อข้อมูลที่กำลังสื่อสาร คำร้องขอประมวลผลต่างๆ ซึ่งส่วนใหญ่ต้องการความเป็นเรียลไทม์ หมายความว่าการประมวลผลต้องเสร็จสิ้นภายในเวลาที่กำหนด โดยต้องตอบสนองอะไรบางอย่าง ดังนั้น ฟังก์ชั่นในการสื่อสารระหว่างอุปกรณ์เหล่านี้จำเป็นต้องตอบสนองภายในเวลาที่จำกัด

การได้รับคำสั่งต่างๆ จากภายนอกโดยต้องการการตอบสนองจากระบบ เราเรียกสภาวะการตอบสนองว่า เหตุการณ์ (event) ในระบบเรียลไทม์ เมื่อเกิดเหตุการณ์อะไรบางอย่างขึ้นมาจะต้องตอบสนองต่อเหตุการณ์นั้นภายในเวลาที่กำหนด ดังแสดงดังรูปที่ 2.1.1

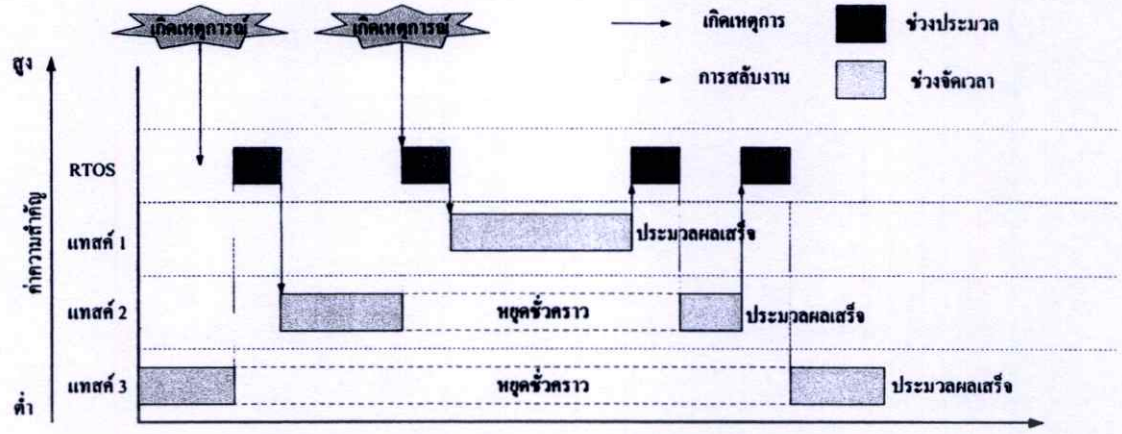
ระบบเรียลไทม์สามารถแบ่งออกเป็น 2 ประเภท คือ ระบบเรียลไทม์แบบอ่อน(soft real-time system) และแบบหลังจะเรียกว่า ระบบเรียลไทม์แบบแข็ง (hard real-time system)



รูปที่ 2.11 การร้องให้ตอบสนองภายในเวลาที่กำหนด

ในกรณีที่เกิดเหตุการณ์หลายเหตุการณ์พร้อมๆ กัน ระบบจะต้องสามารถตอบสนองต่อเหตุการณ์แต่ละเหตุการณ์ภายในเวลาที่กำหนด แต่โดยปรกติแล้วหน่วยประมวลผลไมโครจะมีเพียงชิ้นเดียว จึงทำให้ไม่สามารถตอบสนองต่อเหตุการณ์มากกว่าหนึ่งเหตุการณ์ในเวลาเดียวกันได้ ดังนั้นจึงจำเป็นต้องมีการให้ลำดับกับเหตุการณ์ที่เข้ามา เพื่อควบคุมให้เหตุการณ์ไหนเกิดขึ้นก่อนเหตุการณ์ไหนถูกประมวลผลหลัง โพรเซสการควบคุมแบบนี้เรียกว่า การจัดตารางเวลางาน (task scheduling) ตัวชี้วัดว่าโพรเซสไหนทำก่อน โพรเซสไหนทำหลังนั้น เรียกว่า ค่าความสำคัญหรือไพร์ออริตี้ (priority) เหตุการณ์ที่มีไพร์ออริตี้สูงนั้นจะถูกประมวลผลก่อน การให้ค่าความสำคัญหรือไพร์ออริตีเพื่อจัดลำดับให้ประมวลผลก่อนหรือหลังนี้ อาจทำให้เกิดกรณีที่เหตุการณ์ที่มีไพร์ออริตีต่ำนั้นไม่ถูกตอบสนองในเวลาที่กำหนด สถานการณ์เช่นนี้เรียกว่า หมดเวลาหรือไทม์เอาต์ (time out) หรือเลยเวลา หรือไทม์โอเวอร์ (time over)

ดังนั้น จุดประสงค์หลักของเรียลไทม์ก็คือ การตอบสนองเหตุการณ์หลายๆ เหตุการณ์ภายในเวลาที่กำหนด โดยสิ่งที่ให้กรอบการจัดตารางเวลางาน (scheduling framework) เพื่อทำให้เกิดระบบเรียลไทม์ก็คือ ระบบปฏิบัติการเรียลไทม์ หรือเรียลไทม์โอเอส (real-time OS) เรียกย่อๆ ว่า RTOS



รูปที่ 2.12 วิธีการพรีเอมชันตามเหตุการณ์

โดยระบบปฏิบัติการเรียลไทม์จะให้ค่าความสำคัญหรือไพร์อริตีแก่การประมวลผลเหตุการณ์ งานหรือแทสค์ ด้วยวิธีนี้ งานที่มีค่าความสำคัญน้อยก็จะถูกหยุดไว้ก่อน และระบบจะไปทำงานที่มีค่าความสำคัญมากกว่า ทำให้ระบบสามารถตอบสนองต่อเหตุการณ์ที่เร่งด่วนได้ และจะเป็นการลดความเสี่ยงของการตอบสนองไม่ทันตามเวลาที่กำหนดได้

การจัดตารางงานแบบนี้เรียกว่า การพรีเอมชันตามเหตุการณ์ (event-driven preemption) ดังแสดงดังรูปที่ 2.12 ค่าความสำคัญของงานนั้น จำเป็นต้องเรียงตามความเร่งด่วนของงาน ซึ่งเราจะไม่ยอมให้เคอร์เนลมาทำการเปลี่ยนลำดับค่าความสำคัญตามใจชอบได้

รูปแบบที่เราไม่ยอมให้คอร์เนลมาแก้ไขความสำคัญได้นั้น เรียกว่า วิธีจัดตารางเวลาแบบค่าความสำคัญตายตัว (fixed priority scheduling method)

ข้อควรระวังอันหนึ่งเกี่ยวกับระบบปฏิบัติการเรียลไทม์ก็คือ การกำหนดกรอบการทำงานของระบบปฏิบัติการเรียลไทม์ ซึ่งไม่ได้หมายความว่า กรอบการทำงานนี้จะรับประกันการตอบสนองภายในเวลาที่กำหนด ความเร็วของการประมวลผลก็จะแตกต่างกันไปตามเนื้อหาการประมวลผลและความสามารถของฮาร์ดแวร์ เช่น CPU เป็นต้น การทำให้การทำงานของระบบเป็นเรียลไทม์จริงๆ จำเป็นต้องพึ่งความสามารถในการออกแบบระบบหรือความสามารถในการโปรแกรมของผู้พัฒนาซอฟต์แวร์สมองกลฝังตัว โดยจะต้องคำนึงถึงขีดความสามารถของฮาร์ดแวร์ที่จะนำมาประกอบในการออกแบบ และสร้างระบบเรียลไทม์ภายใต้กรอบการทำงานของระบบปฏิบัติการเรียลไทม์

ตัวอย่างเช่น การใช้งานกรอบการทำงานของระบบปฏิบัติการเรียลไทม์นั้น เมื่อเลยเวลาที่กำหนดจะเกิดเหตุการณ์ที่เรียกว่า “เลยเวลาไทม์โอเวอร์” และ “หมดเวลาหรือไทม์เอาต์” และการจัดการเมื่อเกิดเหตุการณ์เหล่านี้ หรือการป้องกันไม่ให้เกิดไทม์โอเวอร์หรือไทม์เอาต์นั้นเป็นหน้าที่ของผู้ที่พัฒนาระบบที่ใช้ระบบปฏิบัติการเรียลไทม์

เรียลไทม์เคอร์เนลเป็นส่วนสนับสนุนฟังก์ชั่นพื้นฐานที่จะทำให้เกิดการประมวลผลแบบเรียลไทม์ เรียลไทม์เคอร์เนลจะทำหน้าที่ในการควบคุมจัดการ CPU เพื่อให้ง่ายต่อการสร้างระบบ

เรียลไทม์ เช่นเดียวกับเคอร์เนลของระบบอื่น เรียลไทม์เคอร์เนลประกอบด้วยส่วนจัดการทรัพยากรฮาร์ดแวร์พื้นฐาน เช่น หน่วยความจำการเข้าแทรกหรืออินเทอร์รัพต์ เป็นต้น นอกจากนี้ยังทำหน้าที่ให้ฟังก์ชั่นซิสเต็มคอล (system call) แก่แอพพลิเคชั่น โดยให้เซอร์วิสในการทำงานร่วมกันระหว่างแทสค์หรืองานต่างๆ ซึ่งทำให้เกิดเป็นระบบเรียลไทม์ขึ้น

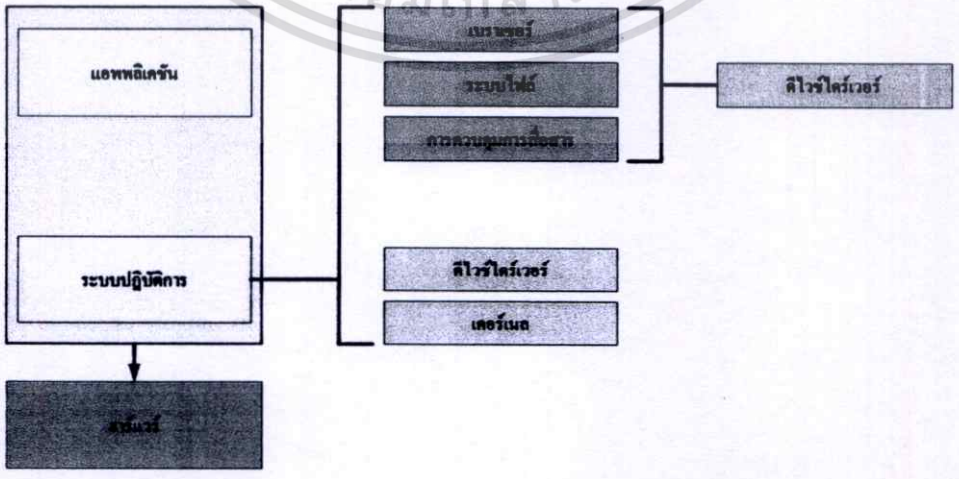
ในการสร้างระบบเรียลไทม์นั้น เราจำเป็นต้องทำความเข้าใจและแบ่งแยกการใช้งานให้เหมาะสมระหว่างฟังก์ชั่นเรียลไทม์ที่เป็นเซอร์วิสแบบแฝง (implicit) ซึ่งทำหน้าที่ในการจัดแบ่งการใช้งาน CPU และฟังก์ชั่นเรียลไทม์ที่เป็นเซอร์วิสแบบชัดเจน (explicit) ในรูปแบบของซิสเต็มคอล ไม่ว่าเราจะซื้อเรียลไทม์เคอร์เนลมาใช้ หรือสร้างเรียลไทม์เคอร์เนลขึ้นมาเองก็ตาม ถ้าเราไม่เข้าใจฟังก์ชั่นต่างๆ เหล่านี้อย่างถูกต้อง ก็จะเป็นอุปสรรคในการสร้างระบบเรียลไทม์ ดังนั้น ในบทนี้จึงขออธิบายรายละเอียดฟังก์ชั่นที่เรียลไทม์เคอร์เนลสนับสนุน

โดยในส่วนแรกจะอธิบายว่า การประมวลผลแบบเรียลไทม์นั้นสามารถทำให้เกิดขึ้นมาได้อย่างไร แล้วจึงอธิบายการอินเทอร์รัพต์ (interrupt) เหตุการณ์ (event) มัลติโปรแกรมมิ่ง (multiprogramming) รวมถึงอธิบายเรื่องเทคนิคในการเขียนโปรแกรมมิ่ง การทำงานของแทสค์ และตัวจัดการอินเทอร์รัพต์ภายใต้สิ่งแวดล้อมแบบมัลติโปรแกรมมิ่ง

2.1.2 ส่วนประกอบของซอฟต์แวร์สมองกลฝังตัว (Software components)

รูปที่ 2.13 แสดงส่วนประกอบโดยสังเขปของซอฟต์แวร์สมองกลฝังตัว โดยซอฟต์แวร์สมองกลฝังตัวก็เหมือนกับระบบทั่วไปที่สามารถแบ่งออกได้ 2 ประเภทใหญ่ๆ คือ

ระบบปฏิบัติการ (operating system) และโปรแกรมประยุกต์ (application program) โดยระบบปฏิบัติการนั้นจะเป็นซอฟต์แวร์ที่ใช้ร่วมกันในระบบต่างๆ และมีหน้าที่ให้กรอบการทำงานแก่ซอฟต์แวร์อื่นๆ ในขณะที่มันทำงานอยู่ ส่วนโปรแกรมประยุกต์หรือแอพพลิเคชันเป็นโปรแกรมที่ใช้กรอบการทำงานที่ระบบปฏิบัติการเสนอให้ เพื่อประมวลผลแก้ปัญหาใดปัญหาหนึ่ง



รูปที่ 2.13 แผนภาพส่วนประกอบซอฟต์แวร์

ส่วนดีไวซ์ไดร์เวอร์นั้นจะใช้เคอร์เนลเพื่อควบคุมอุปกรณ์รับเข้าและส่งออกข้อมูล (input & Output devices – I/O devices) เพื่อทำให้เกิดฟังก์ชั่นในการนำเข้าและส่งออกข้อมูลแก่แอพพลิเคชัน ดีไวซ์ไดร์เวอร์หรือฟังก์ชั่นบางส่วน บางครั้งเราจะเรียกว่า ระบบควบคุมอินพุตเอาต์พุต (Input & Output Control System – IOCS) หรือระบบอินพุตเอาต์พุตพื้นฐาน (Basic Input & Output System – BIOS) เป็นต้น หลังจากเขียนหรือสร้างระบบควบคุมอินพุตเอาต์พุตแล้ว ถ้าไม่มีการเปลี่ยนแปลงฮาร์ดแวร์ ก็จะสามารถใช้งานระบบควบคุมอินพุตเอาต์พุตนี้กับแอพพลิเคชันต่างๆ ได้

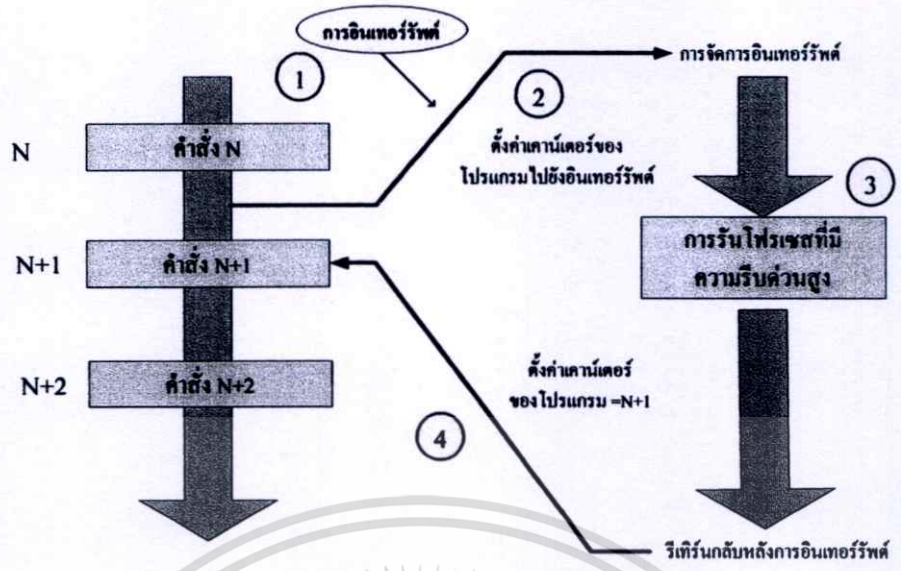
ระบบปฏิบัติการประกอบด้วยเคอร์เนล ดีไวซ์ไดร์เวอร์ และมิดเดิลแวร์ต่างๆ ดังแสดงในรูปที่ 2.13 เคอร์เนลจะทำหน้าที่ควบคุม ดูแล และรันแอพพลิเคชันต่างๆ อย่างที่ได้อธิบายไปแล้วว่าเรียลไทม์เคอร์เนล (real-time kernel) ด้วย และระบบปฏิบัติการที่ใช้เคอร์เนลชนิดนี้คือระบบปฏิบัติการเรียลไทม์

ระบบปฏิบัติการของระบบทั่วไปนั้น ส่วนใหญ่ดีไวซ์ไดร์เวอร์จะเป็นส่วนหนึ่งของระบบปฏิบัติการเลย แต่ถ้าเป็นซอฟต์แวร์สมองกลฝังตัว อุปกรณ์นั้นจะแตกต่างออกไปตามผลิตภัณฑ์ และการใช้งานอุปกรณ์ก็จะแตกต่างกันไปตามความต้องการที่แตกต่างกัน ดังนั้นจึงไม่สามารถที่จะใส่ฟังก์ชั่นของดีไวซ์ไดร์เวอร์แบบตายตัวลงไปในระบบปฏิบัติการได้

สำหรับระบบสมองกลฝังตัว เคอร์เนล ดีไวซ์ไดร์เวอร์ และมิดเดิลแวร์ จะถูกขายเป็นแพ็กเกจแยกตามฟังก์ชั่นของมัน หรือที่เรียกว่า ชิ้นส่วนซอฟต์แวร์ (software components) ในการพัฒนาระบบสมองกลฝังตัวนั้น ส่วนใหญ่ผู้ใช้แต่ละคนจะทำการประกอบชิ้นส่วนซอฟต์แวร์ต่างๆ เหล่านี้เพื่อสร้างระบบปฏิบัติการเฉพาะ สำหรับผู้ใช้คนนั้น แน่นอนว่ามีบ่อยครั้งที่ผู้ใช้นั้นต้องสร้างหรือเขียนชิ้นส่วนซอฟต์แวร์บางอย่างขึ้นมาใช้เอง

2.2.3 การอินเทอร์รัพต์

การอินเทอร์รัพต์หรือการเข้าแทรกเป็นฟังก์ชั่นที่ฮาร์ดแวร์ต้องสนับสนุน ซึ่งมีความสำคัญและจำเป็นมากในการสร้างระบบเรียลไทม์ โดยทั่วไปการทำงานของอุปกรณ์รอบข้างคอมพิวเตอร์ (computer peripherals) เช่น อุปกรณ์รับเข้าหรือส่งออกข้อมูล (I/O device) จะเป็นอิสระจาก CPU โดยความต้องการประมวลผลที่อุปกรณ์เหล่านี้มีต่อ CPU หรือ CPU มีต่ออุปกรณ์เหล่านี้อาจจะเกิดขึ้นเมื่อไรก็ได้ เพื่อตอบสนองความต้องการนี้จึงต้องเตรียมกลไกที่เรียกว่า การอินเทอร์รัพต์ เพื่อทำให้เครื่องสามารถตอบสนองต่อเหตุการณ์ที่เกิดขึ้นได้



รูปที่ 2.14 การหยุดการประมวลผลตามลำดับอันเนื่องมาจากการอินเทอร์รัพต์

การอินเทอร์รัพต์นั้นจะมีกรอบการทำงาน (frame work) ดังนี้ คือ (1) รับสัญญาณจากอุปกรณ์รอบ (2) เมื่อถึงตามกำหนดเวลาที่ตัวนับหรือเคาน์เตอร์ของโปรแกรม (program counter) ตั้งไว้ ก็จะหยุดการทำงานของโปรแกรมที่รันอยู่ชั่วคราว แล้ว (3) บังคับให้รันโปรแกรมที่ตอบสนองต่อสัญญาณที่รับจากอุปกรณ์นั้น แล้ว (4) กลับไปทำงานโปรแกรมเดิมที่ถูกให้หยุดชั่วคราว โดยกรอบการทำงานเช่นนี้ เมื่อมีคำร้องขอประมวลผลจากเครื่องรอบข้าง ก็จะสามารถตอบสนองได้ทันทีระบบเรียลไทม์เป็นระบบที่มีกรอบการทำงานที่ยอมให้ระบบตอบสนองต่อเหตุการณ์ที่มีความเร่งด่วนมากที่สุดก่อน โดยใช้วิธีการอินเทอร์รัพต์จากเหตุการณ์ภายนอก (external event) ที่เข้ามา ดังรูปที่ 2.14

ประเภทของการอินเทอร์รัพต์ (Types of interruption)

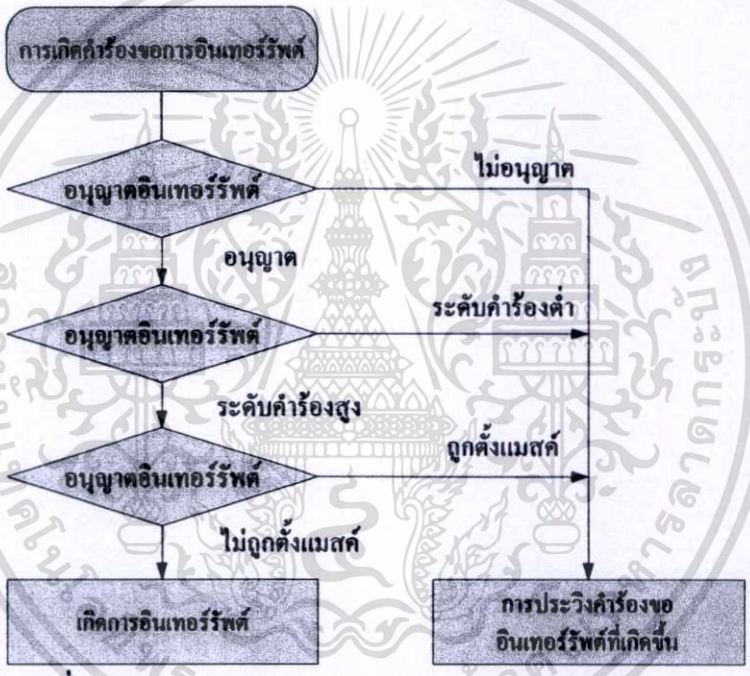
อินเทอร์รัพต์แบ่งออกได้เป็น 2 ประเภทใหญ่ๆ คือ อินเทอร์รัพต์ภายใน (internal interrupt) และอินเทอร์รัพต์ภายนอก (external interrupt) สำหรับอินเทอร์รัพต์ภายในยังแยกออกได้เป็น 2 ประเภท คือ อินเทอร์รัพต์คำสั่ง (command interrupt) และอินเทอร์รัพต์ยกเว้น (exception interrupt) โดยอินเทอร์รัพต์คำสั่งเกิดจากการรันคำสั่งประเภทคำสั่งซิสเต็มคอลหรือ คำสั่งแทรป (trap command) เป็นต้น ส่วนอินเทอร์รัพต์ ยกเว้นจะเกิดขึ้นเมื่อเรารันคำสั่งที่ยังไม่ได้นิยาม หรือการเข้าใช้แอดเดรสอย่างไม่ถูกต้อง ซึ่งเป็นกรณีที่ CPU ไม่สามารถทำงานต่อได้ โดยสรุปแล้วอินเทอร์รัพต์ภายในจะเกิดขึ้นเพื่อพบเหตุการณ์ที่ไม่ปรกติ หรือกรณีเรียกใช้ฟังก์ชั่นคอลเนล เป็นต้น

การอินเทอร์รัพต์ภายนอกคือ การเข้าแทรก CPU ด้วยปัจจัยภายนอก เช่นอุปกรณ์รอบข้าง โดยอินเทอร์รัพต์ภายนอกสามารถแบ่งออกได้เป็น 2 ประเภทคือ อินเทอร์รัพต์จากความผิดปรกติ (abnormal interrupt) และอินเทอร์รัพต์จากอินพุตเอาต์พุต (input & output interrupt) โดยอินเทอร์รัพต์จากความผิดปรกติจะเกิดขึ้นเมื่อมีสิ่งผิดปรกติ เช่น การผิดปรกติในระบบไฟ เป็นต้นสำหรับ

อินเทอร์รัพต์จากอินพุตเอาต์พุต เป็นต้นอินเทอร์รัพต์จากอินพุตเอาต์พุตนั้นเป็นอินเทอร์รัพต์ที่เกิดเพื่อแจ้งว่ามีเหตุการณ์ภายนอกเกิดขึ้น

กรอบการทำงานของอินเทอร์รัพต์ (Interrupt)

1. ขั้นตอนการรับอินเทอร์รัพต์ของ CPU ขณะที่ CPU รับคำสั่งหนึ่งคำสั่งแล้ว มันจะตรวจสอบระดับฮาร์ดแวร์ว่ามีคำร้องของอินเทอร์รัพต์จากอุปกรณ์ประเภทอินพุตเอาต์พุตหรือไม่ และอยู่ในสภาพที่ยอมให้เกิดอินเทอร์รัพต์หรือไม่ ถ้าในสภาพที่เหมาะสมก็จะยอมให้เกิดการอินเทอร์รัพต์ขึ้น รูปที่ 2.15 แสดงลำดับการตรวจสอบการอินเทอร์รัพต์เนื่องจาก CPU อย่างไรก็ตาม CPU ที่แตกต่างกันก็มีลักษณะการตรวจสอบที่แตกต่างกัน CPU บางประเภทอาจไม่มีการตรวจสอบอินเทอร์รัพต์บางอย่าง แต่ CPU บางประเภทอาจมีการตรวจสอบที่ละเอียดมาก



รูปที่ 2.15 แสดงลำดับการตรวจสอบการอินเทอร์รัพต์จาก CPU

การตรวจสอบการยอมรับอินเทอร์รัพต์ (Interrupt permission checking) ของ CPU จะอ้างอิงแฟลก์ซึ่งจะควบคุมสถานภาพของ CPU และอยู่ในตัวของ CPU นั้นเอง (PSW: Program Status Word หรือ flag register) โดยมีการเตรียมแฟลกอนุญาตอินเทอร์รัพต์ (interrupt permission flag) ไว้ในแฟลกรีจิสเตอร์นี้ถ้าไม่อนุญาตให้มีการอินเทอร์รัพต์ ก็จะไม่ยอมให้มีโพรเซสมาอินเทอร์รัพต์ และจะประวิงการอินเทอร์รัพต์ไว้จนกว่าจะได้รับอนุญาต

ในการตรวจสอบระดับของการอินเทอร์รัพต์ (Interrupt level checking) จะอ้างอิงถึงค่าระดับการอนุญาตต้องของ CPU โดยปรกติระดับการอินเทอร์รัพต์จะตั้งไว้ในฮาร์ดแวร์ของอุปกรณ์แต่ละชิ้น ถ้าอุปกรณ์นั้นต้องการอินเทอร์รัพต์ และมีระดับการอินเทอร์รัพต์สูงกว่าค่าระดับการอนุญาตที่ตั้งไว้ใน CPU ก็จะได้รับอนุญาตให้อินเทอร์รัพต์ได้

ในการตรวจสอบหรือหน้ากากของการอินเทอร์รัพต์ (Interrupt mask checking) จะอ้างอิงถึงแฟลกที่ถูกตั้งค่าเพื่อกำหนดการยอมรับอินเทอร์รัพต์จากอุปกรณ์แต่ละชิ้น เรียกว่า อินเทอร์รัพต์แมสค์ (interrupt mask) หรือแฟลกอินเทอร์รัพต์ โดยปรกติภายในโปรแกรมเราสามารถตั้งค่าแฟลกนี้ให้เป็น ON หรือ OFF เพื่อระบุว่ายอมให้เกิดการอินเทอร์รัพต์จากอุปกรณ์ชิ้นนั้นๆ หรือ ไม่ การตั้งค่าแฟลกไม่ให้ยอมรับการอินเทอร์รัพต์นั้นเรียกว่า การตั้งแมสค์ (masking) ให้กับการอินเทอร์รัพต์

2. กลไกของการเกิดอินเทอร์รัพต์หลังจากที่ CPU รับคำร้องของอินเทอร์รัพต์แล้ว ก็จะเกิดการอินเทอร์รัพต์ขึ้น ขั้นตอนการรับนี้แบ่งได้เป็น 2 กลุ่มใหญ่ตามประเภทของคอมพิวเตอร์ คือ CISC (Complex Instruction Set Computer) กับ RISC (Reduced Instruction Set Computer) การอินเทอร์รัพต์นั้นจะถูกกำกับหมายเลข เรียกว่า หมายเลขอินเทอร์รัพต์ (interrupt number)

โดยปรกติแล้ว CPU ประเภทซิสก์ (CISC) จะใช้ตารางเวกเตอร์ (Vector table) ในการจัดการกับอินเทอร์รัพต์ตารางเวกเตอร์ ซึ่งเป็นตารางที่เก็บแอดเดรสส่วนหน้าสุด (start address) หรือที่เรียกว่าอินเทอร์รัพต์เวกเตอร์ (interrupt vector) ของส่วนโปรแกรมรูทีน (routine) ที่ใช้จัดการกับอินเทอร์รัพต์แยกตามหมายเลข ตารางนี้จะถูกสร้างเตรียมไว้ล่วงหน้าด้วยโปรแกรม และเก็บอยู่ในหน่วยความจำบริเวณที่ CPU กำหนด และเมื่อมีการอินเทอร์รัพต์เกิดขึ้น CPU ก็จะไปดูที่ตารางเวกเตอร์นี้ เพื่อหาแอดเดรสของรูทีนที่จัดการกับอินเทอร์รัพต์ตามหมายเลขอินเทอร์รัพต์นั้น

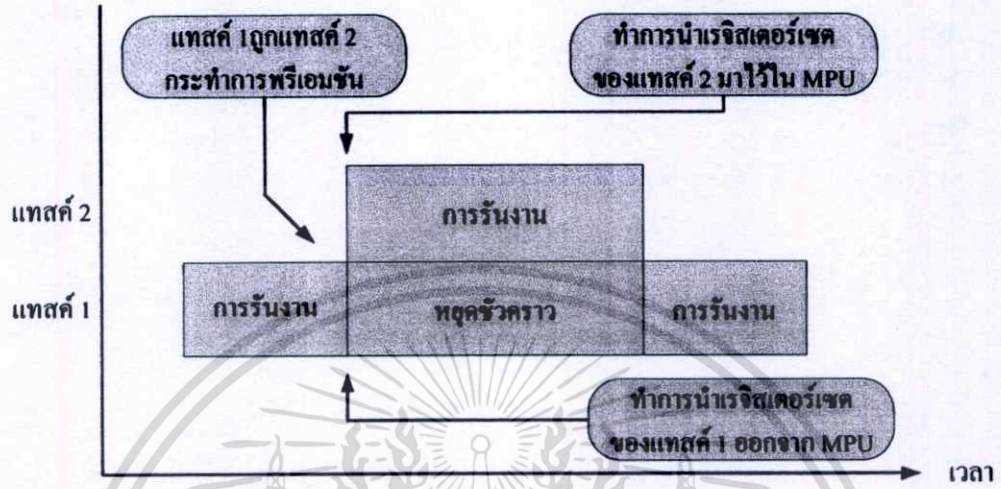
ในทางตรงข้าม CPU ประเภทริสก์ (RISC) จะเรียกว่าอินเทอร์รัพต์ว่า กรณียกเว้น (exception) และมักจะไม่มีตารางเวกเตอร์แยกตามประเภทอินเทอร์รัพต์โดยเฉพาะ แต่จะจัดกลุ่มของอินเทอร์รัพต์เป็นประเภทแล้วกำหนดแอดเดรสของรูทีนที่จัดการกับอินเทอร์รัพต์กลุ่มนั้นๆ และจะแจ้งหมายเลขอินเทอร์รัพต์ผ่านทางรีจิสเตอร์ที่กำหนด

2.2.4 การทำงานแบบมัลติโปรแกรมมิ่งหรือมัลติแทสค์

มัลติโปรแกรมมิ่ง (Multiprogramming) คือการที่เครื่องคอมพิวเตอร์รันโปรแกรมหลายๆ โปรแกรมในเวลาเดียวกัน โดยการวกกับไปและมา ระหว่างโปรแกรมตามตัวเหนี่ยวไก (trigger) บางอย่าง ภายใต้สิ่งแวดล้อมของมัลติโปรแกรมมิ่ง หน่วยของโปรแกรมที่วิ่งสลับไปมานั้นเรียกว่า "งานหรือแทสค์ (task) ดังนั้น มัลติโปรแกรมมีบางครั้งจึงถูกเรียกว่า มัลติแทสค์ (multitask) นอกจากนี้ ในมองมุมของการจัดการทรัพยากรจะเรียกแทสค์ว่า โพรเซสหรือเทรด (process or thread)" และมัลติโปรแกรมมิ่งจะถูกเรียกว่า มัลติโพรเซส (multiprocess) หรือมัลติเทรด (multithread)

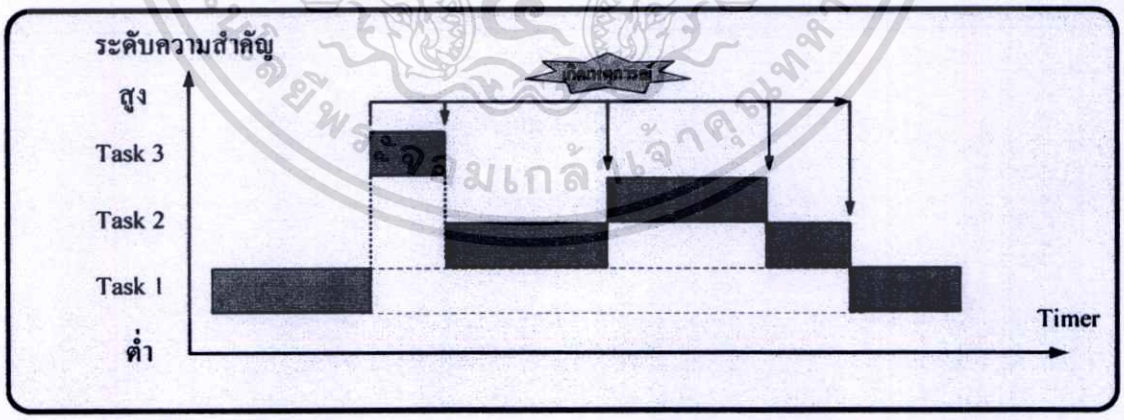
มีแทสค์บางแทสค์ที่หยุดทำงานเพราะแทสค์ตัวอื่น ปรากฏการณ์นี้เรียกว่า "พรีเอมชัน (preemption)" ภายใต้สิ่งแวดล้อมมัลติโปรแกรมมิ่งนั้น แทสค์สามารถถูกรีเอมชันขณะที่ทำงานอยู่ได้ โดยทั่วไปการไหลของการประมวลผล เรียกว่า คอนเท็กซ์หรือบริบท (context) การสลับไป-มาของคอนเท็กซ์ เรียกว่า การสวิตซ์ระหว่างคอนเท็กซ์ (context switching)

รูปที่ 2. 16 แสดงการนำรีจิสเตอร์ของ CPU (CPU register) ที่แทสค์ใช้อยู่ระหว่างทำงาน หรือที่เรียกว่าคอนแท็กซ์มาเก็บลงในสแตก (stack) เพื่อหยุดการทำงานชั่วคราวและนำคอนเท็กซ์กลับมาจากสแตกเพื่อทำงานต่อ



รูปที่ 2.16 การทำพรีเอมชันและการสลับคอนเท็กซ์

เคอร์เนลที่ให้สิ่งแวดล้อมมัลติโปรแกรมมิ่งนั้นจะแบ่งการทำงานโดยรวมทั้งหมดออกเป็นโฟรเซสหรือแทสค์ย่อยๆ แล้วทำการจัดการควบคุม และสั่งให้ทำงานการตัดสินว่างานหรือแทสค์ไหนต้องจัดการก่อนหรือหลัง เรียกว่า "การจัดตารางเวลางาน (task scheduling)" โปรแกรมที่จัดตารางเวลาการทำงาน เรียกว่า "ตัวจัดตารางเวลางาน (task scheduler)"

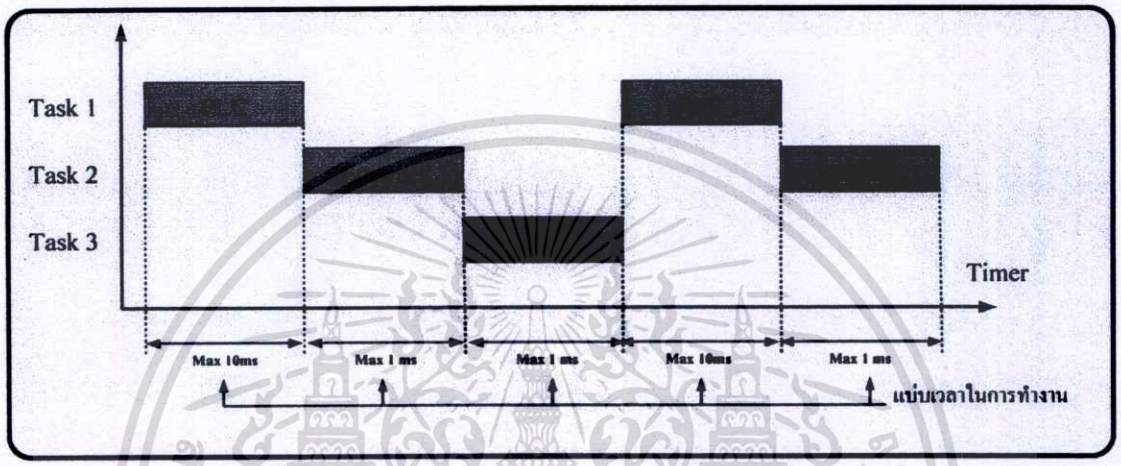


รูปที่ 2.17 วิธีการขับเคลื่อนด้วยเหตุการณ์

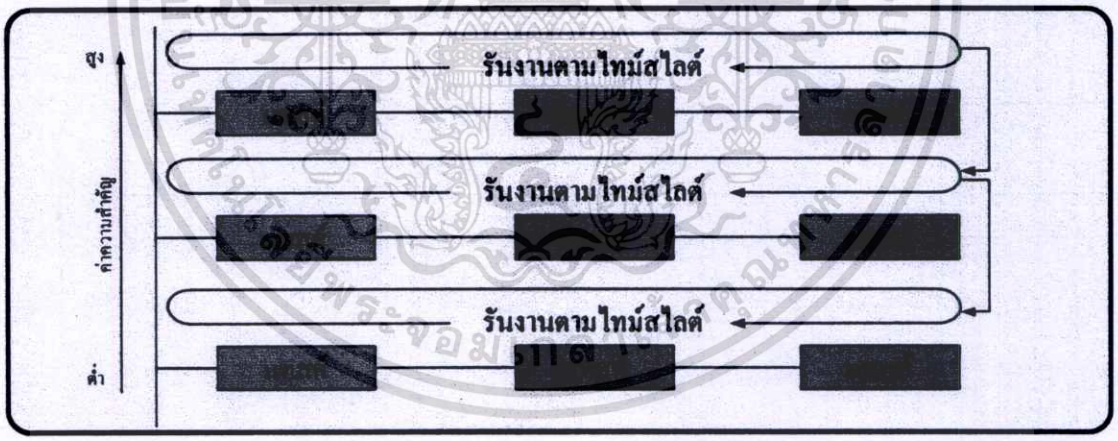
วิธีการจัดตารางเวลางาน (scheduling approach) แบ่งออกได้เป็น วิธีการขับเคลื่อนด้วยเหตุการณ์ (event – driven approach) และ วิธีการแบ่งเวลา (time sharing approach) รูปที่ 2.17แสดงวิธีการแบบขับเคลื่อนด้วยเหตุการณ์ ซึ่งจะมีการให้ค่าความสำคัญกับแทสค์แต่ละแทสค์ และแทสค์ที่มีค่าความสำคัญต่ำจะถูกแทสค์ที่มีค่าความสำคัญสูงกว่ากระทำการพรีเอมชั่น ทำให้แทสค์ที่มีค่า

ความสำคัญสูงทำงานก่อน วิธีการแบบทางตรงข้าม วิธีการแบ่งเวลาหรือที่เรียกว่า ไทม์แชริ่ง เป็นวิธีการจัดตารางเวลาที่มีจุดประสงค์ในการแบ่งปันคอมพิวเตอร์ 1 เครื่องในการใช้ร่วมกัน

ระหว่างผู้ใช้หลายคน ซึ่งเป็นวิธีการที่ใช้ในระบบปฏิบัติการ เช่น ยูนิกซ์ (UNIX) รูปที่ 2.18 แสดงวิธีการแบบแบ่งเวลา โดยวิธีการนี้จะสลับแทสค์ทุกช่วงเวลาขนาดเท่ากัน ซึ่งช่วงเวลาที่เท่ากันนั้นจะเรียกว่า ไทม์สไลด์ (time slide) หรือ (time quantum) วิธีการนี้จะไม่เหมาะสำหรับระบบเรียลไทม์



รูปที่ 2.18 วิธีการแบ่งเวลาการทำงาน



รูปที่ 2.19 วิธีจัดเวลาแบบวนรอบ

วิธีจัดตารางที่เป็นที่รู้กันมากสำหรับแบบแบ่งเวลาก็คือ วิธีการวนรอบ (Round robin method) รูปที่ 2.19 แสดงวิธีการวนรอบที่แบ่งแทสค์ออกเป็นกลุ่มกำหนดตามค่าความสำคัญของงาน และจะจัดการแทสค์ที่มีค่าความสำคัญสูงสุดก่อน แล้วค่อยทำแทสค์ที่มีความสำคัญต่ำกว่าตามลำดับ โดยแทสค์ในกลุ่มที่มีค่าความสำคัญเท่ากันจะทำงานในลักษณะของการแบ่งเป็นไทม์สไลด์ย่อยๆ แล้วทำตามขั้นตอนข้างต้น โดยเมื่อถึงจุดเวลาที่แทสค์ในกลุ่มนั้นกระทำเสร็จสิ้นแล้ว ก็จะดำเนินงานแทสค์ในกลุ่มที่มีความสำคัญต่ำลงมา

2.3 การสื่อสารข้อมูลและเทคโนโลยีระบบโครงข่าย

2.3.1 พื้นฐานและประเภทการสื่อสารข้อมูลแบบโครงข่าย

การที่คอมพิวเตอร์เครื่องหนึ่งจะส่งข้อมูลไปยังคอมพิวเตอร์อีกเครื่องหนึ่งได้นั้น จะต้องอาศัยกลไกหลายๆอย่างร่วมกันทำงานต่างหน้าที่กันและเชื่อมต่อเป็นเครือข่ายเข้าด้วยกัน ปัญหาที่เกิดขึ้นคือการเชื่อมต่อมีความแตกต่าง ระหว่างระบบและอุปกรณ์หรือเป็นผู้ผลิตคนละรายกัน ซึ่งเป็นสิ่งที่ทำให้การสร้างเครือข่ายเป็นเรื่องยากมาก เนื่องจากขาดมาตฐานกลางที่จำเป็นในการเชื่อมต่อ จึงได้เกิดหน่วยงานกำหนดมาตรฐานสากลขึ้นคือ International Standards Organization ขึ้นและทำการกำหนดโครงสร้างทั้งหมดที่จำเป็นต้องใช้ในการสื่อสารข้อมูลและเป็นระบบเปิดเพื่อให้ผู้ผลิตต่างๆสามารถแยกผลิตในส่วนที่ตัวเองถนัด แต่สามารถนำไปใช้ร่วมกันได้ ระบบเครือข่ายคอมพิวเตอร์สมัยใหม่จะถูกออกแบบให้มีโครงสร้างที่แน่นอน และเพื่อเป็นการลดความซับซ้อน ระบบเครือข่ายส่วนมากจึงแยกการทำงานออกเป็นชั้นๆ (layer)

รูปที่ 2.20 แบบจำลอง OSI 7 Layer Reference Model

โดยกำหนดหน้าที่ในแต่ละชั้นไว้อย่างชัดเจน แบบจำลองสำหรับอ้างอิงแบบ OSI (Open System Interconnection Reference Model) หรือที่นิยมเรียกกันทั่วไปว่า OSI Reference Model ของ ISO เป็นแบบจำลองที่ถูกเสนอและพัฒนาโดยองค์กร International Standard Organization (ISO) โดยจะบรรยายถึงโครงสร้างของสถาปัตยกรรมเครือข่ายในอุดมคติ ซึ่งระบบเครือข่ายที่เป็นไปตามสถาปัตยกรรมนี้จะเป็นระบบเครือข่ายแบบเปิด และอุปกรณ์ทางเครือข่ายจะสามารถติดต่อกันได้โดยไม่ขึ้นกับว่าเป็นอุปกรณ์ของผู้จำหน่ายรายใด

หน้าที่ของแต่ละ Layer

Layer7, Application Layer เป็นชั้นที่อยู่บนสุดของขบวนการรับส่งข้อมูล ทำหน้าที่ติดต่อกับผู้ใช้ โดยจะรับคำสั่งต่างๆจากผู้ใช้ส่งให้คอมพิวเตอร์แปลความหมาย และทำงานตามคำสั่งที่ได้รับในระดับโปรแกรมประยุกต์ เช่นแปลความหมายของการกดปุ่มเมาส์ให้เป็นคำสั่งในการก็อปปี้ไฟล์ หรือดึงข้อมูลมาแสดงผลบนหน้าจอเป็นต้น

Layer6, Presentation Layer เป็นชั้นที่ทำหน้าที่ตกลงกับคอมพิวเตอร์อีกด้านหนึ่งในชั้นเดียวกันว่า การรับส่งข้อมูลในระดับโปรแกรมประยุกต์จะมีขั้นตอนและข้อบังคับอย่างไร จุดประสงค์หลักของ Layer นี้คือ กำหนดรูปแบบของการสื่อสาร อย่างเช่น ASCII Text, EBCDIC, Binary และ JPEG รวมถึงการเข้ารหัส (Encryption) ก็รวมอยู่ใน Layer นี้ด้วย

Layer5, Session Layer เป็น Layer ที่ควบคุมการสื่อสารจากต้นทางไปยังปลายทางแบบ End to End และคอยควบคุมช่องทางการสื่อสารในกรณีที่มีหลายๆ โปรเซสต้องการรับส่งข้อมูลพร้อมๆกันบนเครื่องเดียวกัน (ทำงานคล้ายๆเป็นหน้าต่างคอยสลับเปิดให้ข้อมูลเข้าออกตามหมายเลขช่องหรือพอร์ทที่กำหนด) และยังให้อินเตอร์เฟซสำหรับ Application Layer ด้านบนในการควบคุมขั้นตอนการทำงานของ protocol ในระดับ transport/network เช่น socket ของ unix หรือ windows socket ใน windows ซึ่งได้ให้ Application Programming Interface (API) แก่ผู้พัฒนาซอฟแวร์ในระดับบนสำหรับการเขียนโปรแกรมเพื่อควบคุมการทำงานของ protocol TCP/IP ในระดับล่าง และทำหน้าที่ควบคุม "จังหวะ" ในการรับส่งข้อมูล ของทั้ง 2 ด้านให้มีความสอดคล้องกัน (synchronization) และกำหนดวิธีที่ใช้รับส่งข้อมูล เช่นอาจจะเป็นในลักษณะสลับกันส่ง (Half Duplex) หรือรับส่งไปพร้อมกันทั้ง 2 ด้าน (Full Duplex)

Layer 4, Transport Layer เป็น Layer ที่มีหน้าที่หลักในการแบ่งข้อมูลใน Layer บนให้พอเหมาะกับการจัดส่งไปใน Layer ล่าง ซึ่งการแบ่งข้อมูลนี้เรียกว่า Segmentation , ทำหน้าที่ประกอบรวมข้อมูลต่างๆที่ได้รับมาจาก Layer ล่าง และให้บริการตรวจสอบและแก้ไขปัญหาเมื่อเกิดข้อผิดพลาดขึ้นระหว่างการส่ง(error recovery) ทำหน้าที่ยืนยันว่าข้อมูลได้ถูกส่งไปถึงยังเครื่องปลายทางและได้รับข้อมูลถูกต้องเรียบร้อยแล้ว หน่วยของข้อมูลที่ถูกแบ่งแล้วนี้เรียกว่า Segment

Layer3,Network Layer เป็น Layer ที่มีหน้าที่หลักในการส่ง packet จากเครื่องต้นทางให้ไปถึงปลายทางด้วยความพยายามที่ดีที่สุด (best effort delivery) layer นี้จะกำหนดให้มีการตั้ง logical address ขึ้นมาเพื่อใช้ระบุตัวตน ตัวอย่างของ protocol นี้เช่น IP และ logical address ที่ใช้คือหมายเลข IP นั่นเอง layer นี้ส่วนใหญ่เกี่ยวข้องกับอุปกรณ์ฮาร์ดแวร์ซึ่งที่ทำงานอยู่บน Layer นี้คือ router นั่นเอง protocol ที่ทำงานใน layer นี้จะไม่ทราบว่า packet จริงๆแล้วไปถึงเครื่องปลายทางหรือไม่ หน้าที่ยืนยันว่าข้อมูลได้ไปถึงปลายทางจริงๆแล้วคือหน้าที่ของ Transport Layer นั่นเอง หน่วยของ layer นี้คือ packet

Layer2, Data Link Layer รับผิดชอบในการส่งข้อมูลบน network แต่ละประเภทเช่น Ethernet, Token ring, FDDI, หรือบน WAN ต่างๆ ดูแลเรื่องการห่อหุ้มข้อมูลจาก layer บนเช่น

packet IP ไว้ภายใน Frame และส่งจากต้นทางไปยังอุปกรณ์ตัวถัดไป layer นี้จะเข้าใจถึงกลไกและอัลกอริที่มรวมทั้ง format จอง frame ที่ต้องใช้ใน network ประเภทต่างๆเป็นอย่างดี ใน Network แบบ Ethernet layer นี้จะมีการระบุหมายเลข address ของเครื่อง/อุปกรณ์ต้นทางกับเครื่อง/อุปกรณ์ปลาทางด้วย hardware address ที่เรียกว่า MAC Address MAC Address เป็น address ที่ฝังมากับอุปกรณ์นั้นเลยไม่สามารถเปลี่ยนเองได้ MAC Address เป็นตัวเลขขนาด 6 byte, 3 byte แรกจะได้รับการจัดสรรโดยองค์กรกลาง IEEE ให้กับผู้ผลิตแต่ละราย ส่วนตัวเลข 3 byte หลังทางผู้ผลิตจะเป็นผู้กำหนดเอง หน่วยของ layer นี้คือ Frame ตัวอย่างของ protocol ในชั้นนี้คือ Ethernet, Token Ring, IEEE 802.3/202.2,Frame Relay, FDDI, HDLC,ATM เป็นต้น

Layer1, Physical Layer ใน Layer นี้เป็นการกล่าวถึงข้อกำหนดมาตรฐานคุณสมบัติทางกายภาพของฮาร์ดแวร์ที่ใช้เชื่อมต่อระหว่างคอมพิวเตอร์ทั้ง2ระบบ สัญญาณทางไฟฟ้าและการเชื่อมต่อต่างๆของสายเคเบิล,Connector ต่างๆ เช่นสายที่ใช้รับส่งข้อมูลเป็นแบบไหน ข้อต่อหรือปลั๊กที่ใช้มีมาตรฐานอย่างไร ใช้ไฟกี่โวลต์ ความเร็วในการรับส่งเป็นเท่าไร สัญญาณที่ใช้รับส่งข้อมูลมีมาตรฐานอย่างไร Layer1 นี้จะมองเห็นข้อมูลเป็นการรับ-ส่งที่ละ bit เรียงต่อกันไปโดยไม่มีการพิจารณาเรื่องความหมายของข้อมูลเลย การรับส่งจะเป็นในรูป 0 หรือ 1 หากการรับส่งข้อมูลมีปัญหาเนื่องจากฮาร์ดแวร์ เช่นสายขาดก็จะเป็นหน้าที่ของ Layer1 นี้ที่จะตรวจสอบและแจ้งข้อผิดพลาดนั้นให้ชั้นอื่นๆที่อยู่เหนือขึ้นไปทราบ หน่วยของ layer นี้คือ bits ตัวอย่างของ protocol ในชั้นนี้คือ CAT5,CAT6,RJ-45,EIA/TIA-232,V.35cable เป็นต้น

2.3.3 การสื่อสารระบบเครือข่ายแบบอีเทอร์เน็ต

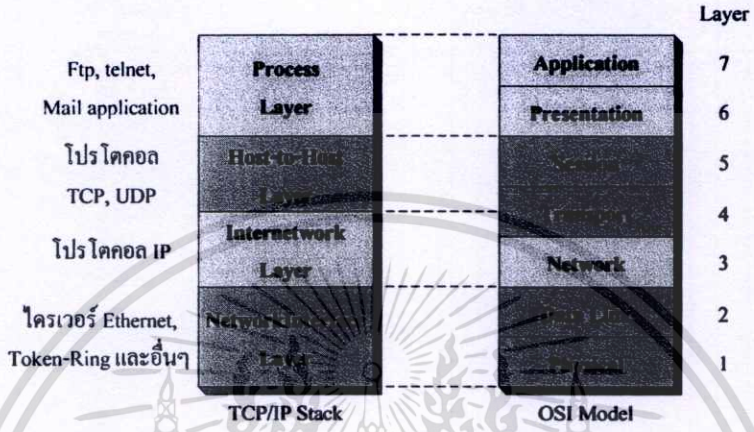
อีเทอร์เน็ต (Ethernet) เป็นเทคโนโลยีเครือข่ายคอมพิวเตอร์ที่เป็นฐานหลักของเทคโนโลยีสารสนเทศทั้งหมด เนื่องจากเป็นเทคโนโลยี LAN ที่ได้รับความนิยมมากที่สุด เทคโนโลยีนี้ได้ถูกพัฒนาและปรับปรุงภายใต้ความดูแลรับผิดชอบของ IEEE สิ่งสำคัญที่ได้รับการเปลี่ยนแปลงปรับปรุง คือ "ความเร็วในการรับส่งข้อมูล (Bandwidth)" โดยมีการปรับปรุงความเร็วจาก 10 Mbps เป็น 100 Mbps ซึ่งเรียก Ethernet นี้ว่า Fast Ethernet ซึ่งได้รับความนิยม

2.3.3.1 การสื่อสารผ่านโปรโตคอล TCP/IP

โปรโตคอล TCP/IP มีการจัดแบ่งกลไกการทำงานออกเป็นชั้นๆ หรือ layer เหมือนกับมาตรฐาน OSI Model และสามารถเทียบเคียงกับมาตรฐานของ OSI Model ได้ ซึ่งในแต่ละ layer ของโปรโตคอล TCP/IP จะประกอบด้วย

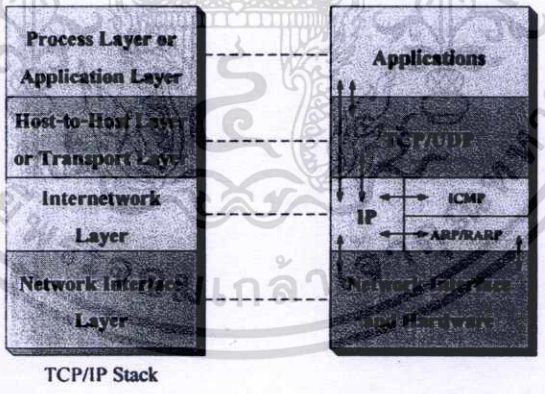
- Process layer หรือ Application Layer
- Host-to-Host layer หรือ Transport Layer
- Internetworking layer
- Network Interface layer

โดยเมื่อเทียบกับมาตรฐาน OSI model แล้วจะเป็นดังรูปที่ 2.21 ซึ่งเราจะเห็นว่าบาง Layer ของโปรโตคอล TCP/IP เทียบได้กับมาตรฐาน OSI model ถึงสอง Layer และบาง Layer ก็จะทำงานคาบเกี่ยวกับหลายๆ Layer model ตัวอย่างเช่น ในส่วน Network Interface layer ของโปรโตคอล TCP/IP จะเทียบได้กับการนำเอา Data Link layer และ Physical layer ของมาตรฐาน OSI model มารวมกัน เป็นต้น



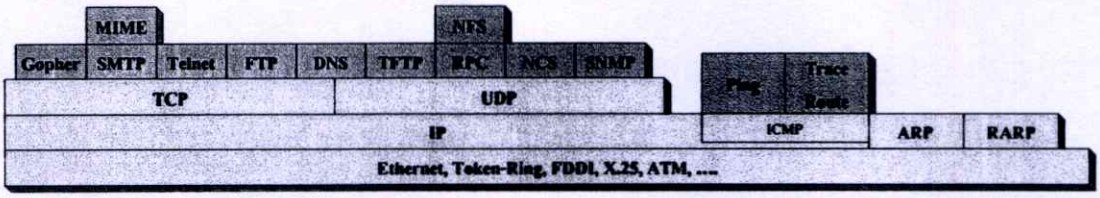
รูปที่ 2.21 แสดง TCP/IP stack เปรียบเทียบกับมาตรฐาน OSI

TCP/IP จะมีโปรโตคอลอื่นๆในชุดของ TCP/IP ร่วมทำงานอยู่ด้วย จึงทำให้เป็นที่มาของชื่อเรียก Protocol Stack เนื่องจากมีโปรโตคอลซ้อนทับกันอยู่เพื่อช่วยกันทำงานดังรูปต่อไปนี้



รูปที่ 2.22 แสดงความสัมพันธ์ระหว่างโปรโตคอลต่างๆ ของ TCP/IP

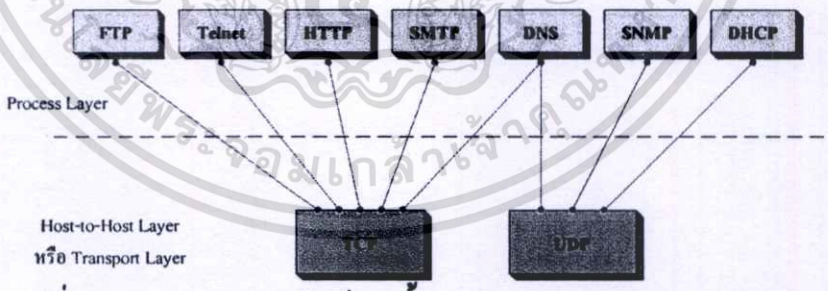
รูปจะเห็นได้ว่า มีโปรโตคอลในแต่ละระดับซ้อนทับกันอยู่หลายตัวด้วยกัน ซึ่งเราจะพูดกันในLayer นี้หากเป็น OSI Model จะมีข้อบังคับให้แต่ละชั้นติดต่อได้กับเฉพาะชั้นที่ติดกับตนเองเท่านั้น แต่สำหรับ TCP/IP Stack แล้วจะเห็นว่าบางชั้นสามารถละเลยหรือข้ามไปติดต่อกับชั้นอื่นที่ไม่ติดกับตนได้



รูปที่ 2.23 แสดงแอพพลิเคชั่นต่างๆใน TCP/IP Stack

จากรูปแสดงลำดับชั้นการทำงานของโปรโตคอล TCP/IP เทียบกับมาตรฐาน OSI model นั้นในชั้นบนสุดเรียกว่า Process layer ทำงาน หน้าที่เทียบได้กับ 2 Application layer และ Presentation layer ในชั้นนี้จะรอบรับการทำงานของแอพพลิเคชั่นต่างๆ ที่ทำงานเป็นโปรเซส อยู่ในเครื่องเซิร์พเวอร์ที่ให้บริการและเครื่องที่ขอใช้บริการ หรือไคลเอนต์ client ซึ่งจะติดต่อกับผ่านโปรโตคอลเฉพาะแอพพลิเคชั่นอีกทีหนึ่ง การทำงานของแอพพลิเคชั่นต่างๆจะอยู่ที่ Process layer นี้และมีการติดต่อกันตามแต่ละโปรโตคอลเฉพาะแล้วแต่แอพพลิเคชั่นทีใช้งาน จากการที่ Process layer ของ TCP/IP รองรับให้โปรโตคอลอื่นทำงานได้หลายโปรเซสและหลายโปรโตคอลได้พร้อมกันนั้น ทำให้ผู้ใช้สามารถเปิดโปรแกรมใช้งานได้หลายๆอย่างพร้อม

โปรโตคอลหลักๆที่ทำงานในProcess layer ซึ่งผู้ใช้มันจะคุ้นเคยกันดีได้แก่ FTP (File Transfer Protocol), Telnet, HTTP (Hyper Text Transfer Protocol) และ SMTP (Simple Mail Transfer protocol) นอกจากนี้ยังมีโปรโตคอลอื่นที่อยู่เบื้องหลัง ซึ่งทำงานโดยที่ผู้ใช้ไม่สามารถมองเห็นได้จากโปรแกรม



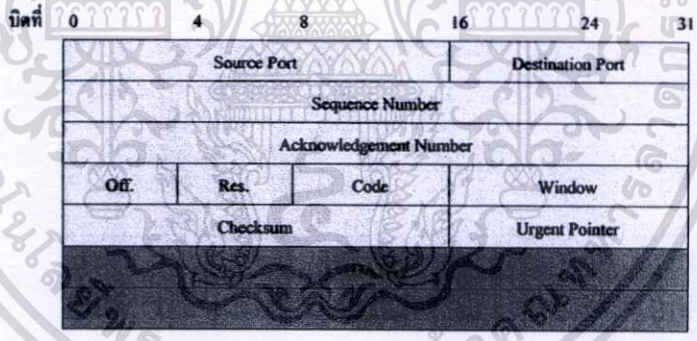
รูปที่ 2.24 แสดงความสำพันธ์ของชั้น Process layer และ Transport

รูปที่ 2.24 โปรเซสต่างๆที่เรียกใช้ Transport layer เพื่อส่งผ่านข้อมูลโดยอาศัย port ซึ่งในแต่ละโปรเซสจะเรียกใช้งานได้ port เฉพาะแตกต่างกัน ยกเว้น DNS ทีสามารถใช้งานได้ทั้ง TCP และ UDP โปรโตคอลตัว TCP และโปรโตคอล UDP จะมีแอพพลิเคชั่นเฉพาะเพื่อเรียกใช้งานแยกกันคือแอพพลิคเคชั่นที่ใช้โปรโตคอล FTP, Telnet, HTTP และ SMTP จะมีการส่งผ่านข้อมูลโดยเรียกใช้โปรโตคอล TCP ส่วนแอพพลิเคชั่นที่ใช้โปรโตคอล SNMP และ DHCP จะส่งผ่านข้อมูลโดยเรียกใช้โปรโตคอล UDP และสำหรับโปรโตคอล DNS นั้นจะสามารถเรียกใช้งานได้ทั้ง TCP และ UDP ดังรูป ซึ่ง

เหตุผลที่มีการเรียกใช้โปรโตคอล TCP และ UDP แตกต่างกัน ก็เนื่องจากวิธีการทำงานของทั้งสองโปรโตคอลต่างกันนั่นเอง

โปรโตคอล TCP (RFC 793)

โปรโตคอล TCP (Transmission Control Protocol) เป็นโปรโตคอลที่มีการรับส่งข้อมูลแบบ stream oriented protocol หมายความว่า การรับส่งข้อมูลจะไม่คำนึงถึงปริมาณข้อมูลที่จะส่งไป แต่จะแบ่งข้อมูลเป็นส่วนย่อยๆก่อน แล้วจึงจะส่งไปยังปลายทางอย่างต่อเนื่องเป็นลำดับข้อมูล ในกรณีที่ข้อมูลส่วนใดส่วนหนึ่งสูญหายไป ก็จะส่งข้อมูลส่วนนั้นใหม่อีกครั้ง สำหรับปลายทางก็จะทำหน้าที่จัดเรียงส่วนของข้อมูล datagram ใหม่ให้ต่อเนื่องกันและประกอบกลับเป็นข้อมูลทั้งหมดได้ ซึ่งจะแยกข้อมูลที่ไม่ถูกต้องออก ดังนั้นแอพพลิเคชั่นหรือโปรเซสใดที่อาศัยการส่งผ่านข้อมูลด้วยโปรโตคอล TCP จะต้องใช้หน่วยความจำและขนาดของช่องสัญญาณ (bandwidth) มากกว่า UDP ในระหว่างการรับส่งข้อมูลนี้ โปรโตคอล TCP จะเพิ่มขบวนการสอบทานข้อมูลเพื่อให้ข้อมูลมีความถูกต้องไม่ผิดพลาดไปจากเดิม โดยการส่งสัญญาณสอบทานข้อมูล (acknowledgement) และส่งข้อมูลให้ใหม่อีกครั้ง ถ้าปลายทางไม่ได้รับหรือเกิดความผิดพลาดขึ้น



รูปที่2.25 รูปแบบของ TCP packet

ความน่าเชื่อถือของการส่งผ่านข้อมูลโดยโปรโตคอล TCP จะมีมากกว่า แต่ก็ต้องอาศัยทรัพยากรของระบบมากกว่าในการทำงานเช่นกันรูปที่ 2.25 รูปแบบของ TCP packet จะเห็นว่ามีฟิลด์ Acknowledgement Number และข้อมูล Checksum รูปที่2.25 รูปแบบของ TCP packet จะเห็นว่ามีฟิลด์ Acknowledgement Number และข้อมูล Checksum

โปรโตคอล IP (RFC 791)

โปรโตคอลIP ทำหน้าที่ให้บริการส่งผ่านข้อมูลที่มาจาก Host-to-Host layer เพื่อส่งข้ามไปยังเครือข่ายใดๆได้อย่างถูกต้อง แม้ว่าจะมีเครือข่ายเชื่อมต่อกันอยู่ในอินเตอร์เน็ตเป็นล้านๆเครือข่ายก็ตาม เนื่องจากโปรโตคอล IP มีข้อมูลตำแหน่ง IP ปลายทางที่จะส่งข้อมูลไปให้

โดยทำงานร่วมกับอุปกรณ์ Router เพื่อส่งข้อมูลผ่านสวิทซ์ (switch) ไปยังปลายทาง โดยข้อมูลจะเดินทางไปยังเครือข่ายต่างๆผ่านสวิทซ์ไปเรื่อยๆจนกว่าจะถึงปลายทาง ตัววงจรผ่านหรือ switch นี้อาจเป็น Gateway หรือ Router ในระบบเครือข่ายก็ได้ ซึ่งในข้อมูลของโปรโตคอล IP จะมีข้อมูลของหมายเลย IP ให้เป็นหมายเลขฮาร์ดแวร์ประจำเครื่องที่ถูกต้องอีกทีหนึ่งด้วยโปรโตคอล ARP

กลไกของโปรโตคอล IP

ในการส่งผ่านข้อมูล หรือ IP datagram ไปยังเครือข่ายอินเตอร์เน็ตนั้น โปรโตคอล IP จะทำหน้าที่พิจารณาว่าปลายทางในการส่ง IP datagram นั้นจะเป็นภายในเครือข่ายของตนเองหรือจะต้องส่งข้อมูลข้ามเครือข่ายไปอีก โดยการพิจารณานี้โปรโตคอล IP จะตรวจสอบจากค่า IP Address ปลายทางว่าส่วนที่เป็นค่าหมายเลยเครือข่าย (network address) จะเหมือนกับค่าหมายเลยเครือข่ายของ IP Address ต้นทางหรือไม่ ถ้าค่าตรงกันแสดงว่าการส่งข้อมูลอยู่ภายในเครือข่ายเดียวกัน แต่ถ้าค่าต่างกัน แสดงว่าต้องส่งข้อมูลไปยังปลายทางที่อยู่คนละเครือข่ายกัน

- การส่งข้อมูลภายในเครือข่ายเดียวกัน มีกลไกดังนี้

1. โปรโตคอล IP จะเรียกใช้บริการโปรโตคอล ARP (Address Resolution Protocol) เพื่อแปลงหมายเลย IP ปลายทางให้เป็นค่าหมายเลขฮาร์ดแวร์ เช่น MAC address
2. เมื่อโปรโตคอล IP ได้รับค่าหมายเลยฮาร์ดแวร์แล้ว ก็จะส่งข้อมูลนั้นไปยังฮาร์ดแวร์ที่ระบุไว้

- การส่งข้อมูลข้ามเครือข่าย มีกลไกดังนี้

1. โปรโตคอล IP ตรวจสอบพบว่าหมายเลย IP Address ปลายทางอยู่คนละเครือข่ายกัน โดยโปรโตคอล IP จะอ่านค่า IP Address ของ Router เพื่อเตรียมส่งข้อมูลไปที่ Router แทน ซึ่งในที่นี้จะมีการกำหนดเป็น default router
2. โปรโตคอล IP จะเรียกใช้บริการโปรโตคอล ARP เพื่อแ ปลงค่า IP Address ของ Router ให้เป็นค่าหมายเลขฮาร์ดแวร์
3. โปรโตคอล IP ส่งข้อมูล IP datagram ไปยัง Router ที่กำหนดไว้

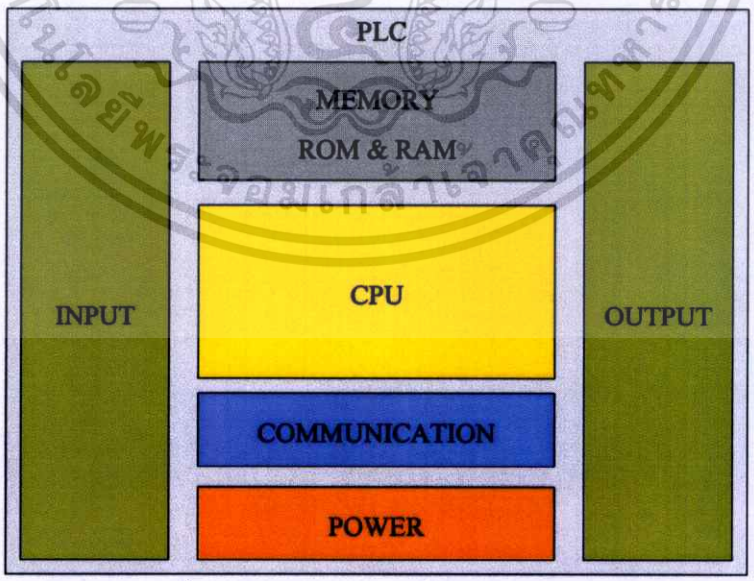
โปรโตคอล IP จะรู้ได้อย่างไรว่าเครือข่ายดังกล่าวมีการเชื่อมต่อ Router อยู่และมีค่า IP อะไร ซึ่งในเรื่องนี้ผู้ใช้จะต้องกำหนดค่าที่เรียกว่า default Router หรือ default Gateway เสียก่อน ว่ามีค่า IP Address อะไร โดยสามารถสอบถามได้จากผู้ดูแลระบบ

บทที่ 3
การออกแบบฮาร์ดแวร์ของเครื่องควบคุม PLC

สาระสำคัญของบทนี้จะอธิบายถึงขั้นตอนการดำเนินการวิจัยที่เกี่ยวข้องกับการออกแบบฮาร์ดแวร์ หลักการและเหตุผลที่ใช้ในการออกแบบ การเลือกใช้ CPU ที่เป็นที่เป็นตัวประมวลผลหลัก วงจรอิเลคทรอนิคส์ ส่วนต่างๆ เช่น วงจรภาคจ่ายไฟ วงจรในการติดต่อหน่วยความจำภายนอกแบบ SD Card วงจรในการติดต่อสื่อสารต่างๆ ของ PLC และขั้นตอนการออกแบบลายวงจรพิมพ์

3.1 การออกแบบทางฮาร์ดแวร์

โครงสร้างของ PLC ขนาดเล็กปัจจุบันจะเป็นอุปกรณ์สมองกลเช่นกันที่ออกแบบมาเฉพาะทางให้เป็นเครื่องควบคุม PLC ในส่วนของชุดพื้นฐานจะประกอบด้วย หน่วยประมวลผลกลาง หน่วยความจำ หน่วยรับข้อมูล หน่วยส่งข้อมูล และหน่วยป้อนโปรแกรม ส่วนประกอบทั้งหมดของ PLC จะรวมกันอยู่ภายในชุดพื้นฐาน ถ้าเป็น PLC ขนาดใหญ่สามารถแยกส่วนประกอบหลักที่จำเป็นต้องใช้ร่วมกับโมดูล ต่างๆ เช่น อินพุทโมดูล เอาท์พุทโมดูล อนาล็อกโมดูล เป็นต้น มีความต้องการใช้งานโมดูลใดก็จัดหาโมดูลนั้นมาประกอบร่วมด้วยทำให้ระบบมีความยืดหยุ่นสูง สำหรับการออกแบบ PLC ขนาดเล็กจะมีส่วนประกอบทางฮาร์ดแวร์ตามโครงสร้างดังรูปที่ 3.1



รูปที่3.1 โครงสร้างของเครื่องควบคุม PLC

3.1.1 หน่วยประมวลผลกลาง (CPU)

หน่วยประมวลผลของ PLC ทำหน้าที่ในการควบคุมการทำงานของระบบทั้งหมดโดยรับข้อมูลเข้ามาทางอินพุต เพื่อนำมาประมวลผลในโปรแกรมคำสั่ง PLC ของผู้ใช้งานแล้วนำข้อมูลที่ประมวลผลได้ส่งออกต่อไปยังเอาท์พุต หลังจากนั้นจะวนกลับมารับข้อมูลอินพุตเข้าชุดใหม่แล้วทำซ้ำในลักษณะเดิมเป็นอย่างนี้เรื่อยไปจนกว่าจะยกเลิกการทำงาน เวลาในการประมวล เร็วหรือช้าขึ้นอยู่กับขีดความสามารถของ CPU ดังนั้น CPU จึงเป็นชิ้นส่วนสำคัญที่สุดในการออกแบบเครื่อง PLC ในการออกแบบของงานวิจัยนี้เราจะเลือกใช้ ไมโครคอนโทรลเลอร์ ซึ่งเป็นชิบอุปกรณ์ควบคุมขนาดเล็ก กล่าวคือภายในไมโครคอนโทรลเลอร์ ได้รวมเอาซีพียู หน่วยความจำชั่วคราว (RAM) หน่วยความจำถาวร (ROM) และพอร์ตอินพุตเอาท์พุต ส่วนประกอบสำคัญทั้งหมดนี้นำมารวมเข้าไว้ด้วยกัน โดยบรรจุเข้าไว้ในตัวถังชิพเดียวกันทำให้ง่ายแก่การออกแบบ การเลือก CPU (Micro Controller Unit) เพื่อออกแบบให้เป็นเครื่องควบคุม PLC สามารถตั้งสมมุติฐานเกี่ยวกับคุณสมบัติของ CPU ไว้ดังนี้

1. ความเร็วของ CPU ต้องมีการทำงานที่ความเร็วสูง เนื่องจาก PLC ที่มาจากต่างประเทศ จะออกแบบไอซี CPU เฉพาะขึ้นมาใช้งานเป็น PLC โดยคำสั่งเป็นคำสั่งตรงที่ CPU นำไปประมวลผลได้ทันที คำสั่งจึงกระชับ สั้น และทำงานได้เร็ว แต่งานวิจัยนี้ประยุกต์เอาสมองกลฝังตัวมาทำเป็น PLC ใช้วิธีการทางซอฟท์แวร์เขียนกลุ่มคำสั่ง CPU หลายคำสั่งให้เป็นคำสั่ง PLC หนึ่งคำสั่ง การทำงานจึงอ้อมกว่าแบบแรกที่กล่าวมา นอกจากนี้ยังต้องบริการงานอื่นอีกเช่น การสื่อสารข้อมูลกับอุปกรณ์ภายนอกไปพร้อมๆกัน ไม่ว่าจะเป็นการบริการในส่วนของการสื่อสารทางพอร์ทอนุกรม RS 232 พอร์ท USB และ พอร์ท Ethernet อีกด้วยดังนั้นความเร็วของ CPU จึงต้องมีประสิทธิภาพมาก ความเร็วของ CPU นั้น มีหน่วยวัดเป็น มิปส์ (MIPS) ซึ่งเป็นหน่วยวัดความเร็วในการประมวลผลของซีพียูในคอมพิวเตอร์ระดับกลางขึ้นไป โดย 1 MIPS จะสามารถประมวลผลได้หนึ่งล้านคำสั่งต่อวินาที (Million Instruction Per Second) หน่วยวัดแบบ มิปส์ นี้จะมีความเที่ยงตรงกว่าหน่วยวัดความเร็วแบบ Clock Speed ที่ใช้ในไมโครคอมพิวเตอร์รุ่นก่อนๆอีกด้วย

2. จำนวนบิตในการประมวลผล จากโครงสร้างหน่วยความจำ ของPLC ส่วนใหญ่จะมีขนาดเป็น word ซึ่งในหนึ่ง word มีขนาด 16 บิต ดังนั้นถ้าเลือกใช้ CPU ขนาด 8 บิตจึงจำเป็นต้องประมวลผล 2 ครั้ง ถ้าเลือกใช้ขนาด 16 บิต หรือ 32 บิตก็จะประมวลผลเพียงครั้งเดียวทำให้เวลาในการปฏิบัติคำสั่งนั้นเร็วขึ้น

3. หน่วยความจำโปรแกรม ROM เป็นหน่วยความจำที่ใช้ในการเก็บระบบปฏิบัติการ ที่ควบคุมการทำงานทั้งหมด ของ PLC ซึ่งต้องสามารถปรับปรุงหรือ แก้ไขได้ จึงเลือกหน่วยความจำแบบ Flash Memory ให้มีขนาดใหญ่ เพียงพอที่จะเก็บ โปรแกรมทั้งหมด และมีพื้นที่เหลือพอที่จะพัฒนา หรือ ปรับปรุงโปรแกรมให้มีความสามารถมากขึ้นในอนาคตได้อีกด้วย

4. หน่วยความจำชั่วคราว RAM ทำหน้าที่พักข้อมูลในการประมวลผล และในการออกแบบ PLC จะต้องสร้างหน่วยความจำเสมือนของ PLC ด้วยจึงต้องคำนึงถึงขนาดของหน่วยความจำ RAM อีกเช่นกันถ้าหน่วยความจำภายใน CPU ไม่เพียงพออาจพิจารณาใส่หน่วยความจำเพิ่มเติมจากภายนอกได้

5. พอร์ตที่ติดต่อสื่อสาร มีอยู่ด้วยกันหลายรูปแบบ แต่ในงานวิจัยนี้ต้องการเน้นออกแบบให้สามารถเชื่อมต่อทาง Ethernet ได้ใช้วิธีการทางซอฟท์แวร์เขียนโปรแกรมบริหารการจัดการ TCP/IP Protocol หรืออีกวิธีหนึ่งคือวิธีทางฮาร์ดแวร์ โดยเลือก CPU ที่มีฮาร์ดแวร์ตัวจัดการ TCP/IP Protocol ภายในมาใช้งานแทนในงานวิจัยเลือกใช้แบบที่เป็นฮาร์ดแวร์นี้

นอกจากจะใช้พอร์ท Ethernet แล้วยังมีการติดต่อสื่อสารชนิดอื่นอีก เช่น RS-232, RS-485, USB ใช้เพื่อการติดต่อกับโปรแกรมระบบปฏิบัติการ หรือ CAN bus ใช้เพื่อการติดต่อสื่อสารระยะไกล เป็นต้น จะเห็นได้ว่าการบริการของ CPU ที่มีต่ออุปกรณ์ภายนอกนั้นมีจำนวนมากและต้องทำงานให้ทันเวลาที่กำหนดจึงต้องมีความเร็วในการทำงานที่สูงนั้นเอง

6. ความสามารถพิเศษอื่นๆ ที่อาจจะมีเสริมเพื่อเติม เช่น RTC (Real Time Clock) ใช้เป็นฐานเวลาหรือนาฬิกา ของ PLC, มีพอร์ท SPI หรือ SD bus ใช้ในการการติดต่อกับหน่วยความจำแบบ SD Card การพิจารณาจะอ้างอิงจากคุณสมบัติตามข้อมูลที่ได้กล่าวมาตามข้างต้น ถ้าดูจาก CPU ที่นิยมใช้ การออกแบบ ก็จะมีด้วยกันหลายประเภท ตามตารางที่ 3.1

ตาราง 3.1 เปรียบเทียบคุณสมบัติของ CPU ประเภทต่างๆ

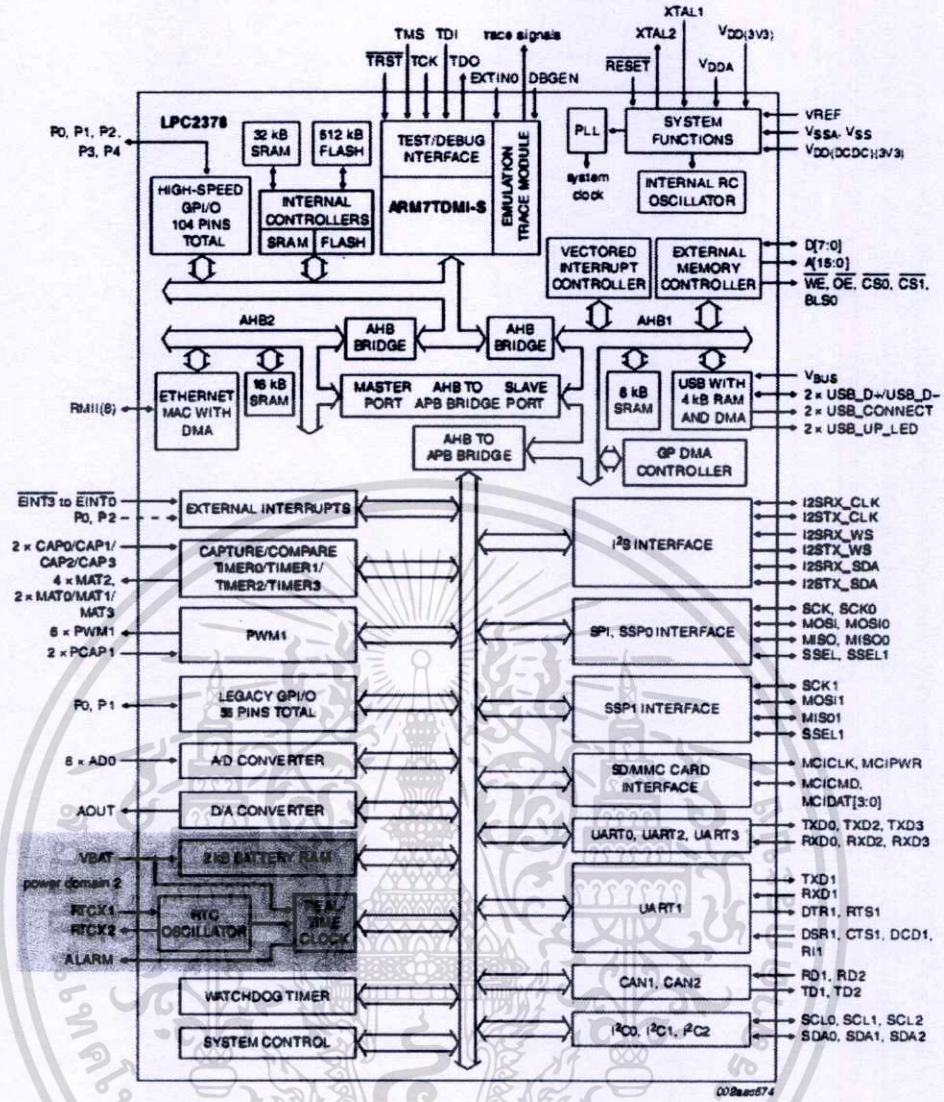
คุณสมบัติ	MCS51	AVR	PIC24	PIC32MX3	LPC2378
จำนวนบิต	8	8	16	32	32
ความเร็ว MHz (MIPS)	24MHz (2 MIPS)	16MHz (16 MIPS)	32MHz (16 MIPS)	80MHz (80 MIPS)	72MHz (62 MIPS)
หน่วยความจำแฟลช	64Kbyte	128Kbyte	128Kbyte	512Kbyte	512Kbyte
RAM	2 Kbyte	4 Kbyte	16 Kbyte	32 Kbyte	58 Kbyte
Ethernet	ซอฟแวร์	ซอฟแวร์	ซอฟแวร์	ซอฟแวร์	ฮาร์ดแวร์
พอร์ตที่ติดต่อสื่อ	RS-232, SPI	RS-232, SPI,	RS-232, SPI ,USB,CAN	RS-232, SPI ,USB,CAN	RS-232, SPI ,USB,CAN, SD BUS
อื่นๆ	-	EEPROM	RTC	RTC	RTC

จากตารางที่ 3.1 พิจารณาความสามารถของ CPU ทั้ง 5 ชนิดที่มีคุณสมบัติเด่นในการใช้งานที่แตกต่างกัน ถ้าพิจารณาคุณสมบัติด้านความเร็วการทำงาน MCS-51 ความเร็วค่อนข้างจะต่ำเพียงแค่ 2 MIPS จึงพิจารณาไว้เป็นทางเลือกสุดท้ายๆ ส่วนหน่วยประมวลผลลำดับถัดมาเป็นแบบ AVR ความเร็วอยู่ที่ 16 MIPS แต่สามารถประมวลผลได้เพียงครั้งละ 8 บิต ซึ่งไม่เหมาะกับการประมวลผลของ PLC ที่จะต้องปฏิบัติคำสั่งกับหน่วยความจำและคำสั่งในระดับ word คือ 16 บิต นั้นเองจึงไม่เหมาะ จึงพิจารณาไว้เป็นทางเลือกสุดท้ายๆ เช่นกัน ทำให้เหลือ CPU ขนาด 16 บิต และ 32 บิต ได้แก่ไอซีตระกูล PIC24F ,PIC32 และ ARM7 ถ้าดูด้านโครงสร้างการติดต่อสื่อสารและการเชื่อมต่อกับอุปกรณ์ภายนอกของ CPU ทั้งสามแบบมี RS-232, SPI, USB, RCT เช่นเดียวกัน แต่ในการเชื่อมต่อ Ethernet LPC2378 จะได้เปรียบเนื่องจากมีฮาร์ดแวร์ที่ทำหน้าที่จัดการเกี่ยวกับ Ethernet โดยตรงซึ่งมีข้อได้เปรียบกว่าไอซีตระกูล PIC24F ,PIC32 ที่ต้องใช้ซอฟต์แวร์ช่วยจัดการให้เกิดฟังค์ชั่นนี้

ถ้าเปรียบเทียบด้านความเร็วและหน่วยความจำแล้วไอซีตระกูล PIC24F จะมีหน่วยความจำ RAM เพียง 16 Kbyte ที่ใช้เก็บเป็นหน่วยความจำเสมือน และ เก็บสถานะของการประมวลผลของ PLC พิจารณาแล้วอาจจะไม่เพียงพอกับการใช้งาน และมีความเร็วต่ำกว่าจึงพิจารณาไว้เป็นทางเลือกท้ายๆ

เมื่อพิจารณาหรือเปรียบเทียบระหว่าง PIC32 และ LPC2378 ทั้งคู่มีหน่วยความจำที่โปรแกรมที่มีขนาดเท่ากัน หน่วยความจำ RAM ของ LPC2378 มีขนาดสูงกว่าคือ 58 Kbyte แต่ด้านการติดต่อสื่อสาร LPC2378 มีการเชื่อมต่อ Ethernet เร็วกว่าเพราะว่าเป็นฮาร์ดแวร์และมี SD BUS ที่ให้ติดต่อ SD Card ได้ในโหมด SD Bus ที่ให้มีความเร็วสูงกว่า PIC32 ที่ติดต่อในรูปแบบ SPI Bus ทำให้งานวิจัยนี้สนใจ และ เลือกเอาไอซีตระกูล ARM7TDMI-S เบอร์ LPC2378 เป็น CPU ของเครื่องควบคุม PLC

ในการออกแบบ PLC เราใช้ ไมโครคอนโทรลเลอร์ 32 ในตระกูล ARM7TDMI-S เบอร์ LPC2378 ของ NXP เป็นขนาด 144 ขา ซึ่งมีหน่วยความจำแบบแฟลชความจุ 512 กิโลไบต์ มีหน่วยความจำสำหรับประมวลผลขนาด 128 บิตเพื่อให้ช่วยประมวลผลขนาด 32บิตได้ด้วยความเร็วสูงทำงานกับความถี่สัญญาณนาฬิกาได้ถึง 72MHz หน่วยความจำแบบสแตติกแรม 58 กิโลไบต์ มีโมดูล USB,RCT, Ethernet, SD/MMC memory card interface, CAN controller, 2 Kbyte SRAM powered backup ซึ่ง จะกล่าวถึงการประยุกต์ให้งานในการออกแบบโครงสร้างของ PLC ตามโครงสร้างของ LPC2378 ดังรูปที่ 3.2 สามารถนำมาออกแบบตามโครงสร้างของ PLC ได้ดังตาราง 3.2



รูปที่ 3.2 โครงสร้าง CPU LPC2378

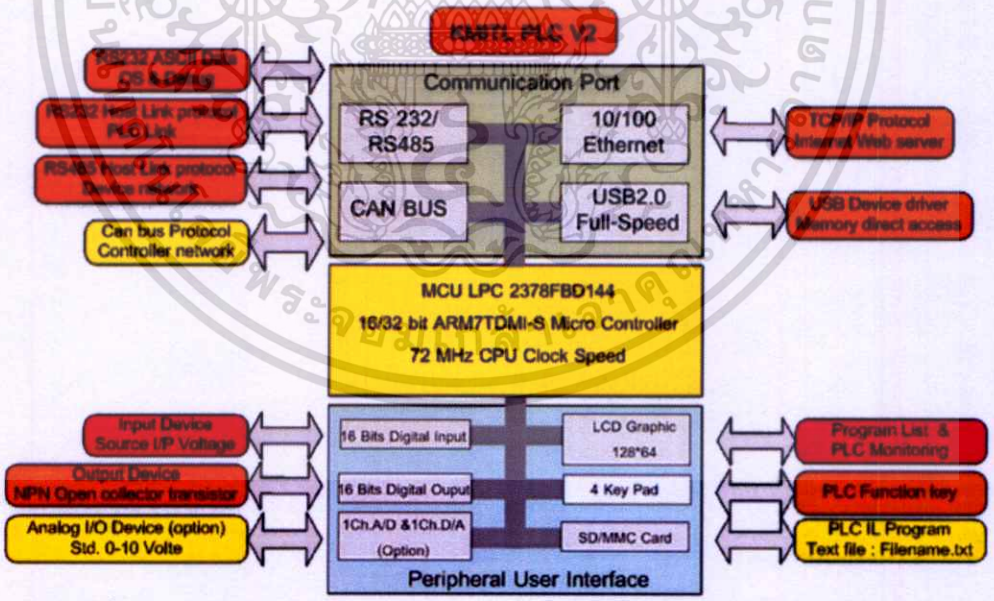
ตาราง 3.2 การออกแบบฮาร์ดแวร์ส่วนต่างของ LPC2378 ให้เป็น PLC

LPC2378	PLC
P0-P4 HIGH-SPEED GPI/O	อินพุต/เอาพุตท์
512 KB FLASH	ใช้เก็บ Operating system PLC
32 KB SRAM	ความจำของ PLC และ Operating system
2 KB BATTERY RAM	ใช้เก็บสถานะของ Holding relay (HR Area)
TIMER0	เป็นตัวกำหนดเวลา TIMER ภายใน PLC (TIM0- TIM256)
REALTIME CLOCK	ใช้เป็นฐานเวลา(นาฬิกา)และวันที่(ปฏิทิน)ของ PLC
SD/MMC CARD INTERFACE	เชื่อมต่อ SD/MMC memory card ในการเก็บโปรแกรม PLC
UART0	เป็น RS-232 ใช้โหลดโปรแกรม OS และ Debug
UART1	เป็น RS-485 ในการเชื่อมต่อระยะไกลระดับอุปกรณ์

ตาราง 3.2 (ต่อ) การออกแบบฮาร์ดแวร์ส่วนต่างของ LPC2378 ให้เป็น PLC

LPC2378	PLC
UART3	เป็น RS-232
CAN1	เป็น CAN Busในการเชื่อมต่อระยะไกลระดับควบคุมและอุปกรณ์
USB	เป็น USB2.0 Port อ่านเขียนหน่วยความจำ PLC โดยตรงใช้เป็นโฮสลิงค์ และ พีซีลิงค์
ETHERNET	ใช้เป็น Embedded Web Server ในการติดต่อกับ PLC
P0-P4 HIGH-SPEED GPI/O	ติดต่อกับผู้ใช้ ในส่วนแสดงผลLCD และคีย์แพ็ต

จากตารางที่ 3.2 เราจะเห็นได้ว่าส่วนโครงสร้างต่างๆของ LPC2378 ถูกนำมาประยุกต์ให้เป็น PLC ในส่วนประกอบที่สำคัญจัดเรียงชิ้นส่วนต่างๆเสียใหม่ให้ดูเป็นกลุ่มและประเภทตามฟังค์ชั่นการใช้งานของ PLC ดังรูปที่ 3.3 ซึ่งออกแบบให้เป็น PLC ที่มีขนาดเล็ก มี CPU LPC2378 ควบคุมงานทั้งหมดเพียงตัวเดียวเป็นหน่วยพื้นฐาน(Base Unit) ออกแบบให้ดิจิตอลอินพุท16 จุด ดิจิตอลเอาท์พุท 16 จุดส่วนที่ติดต่อกับผู้ใช้โดยผ่านทางจอแสดงผล LCD Graphic ขนาด 128x64 จุด รับคำสั่งในการติดต่อกับผู้ใช้



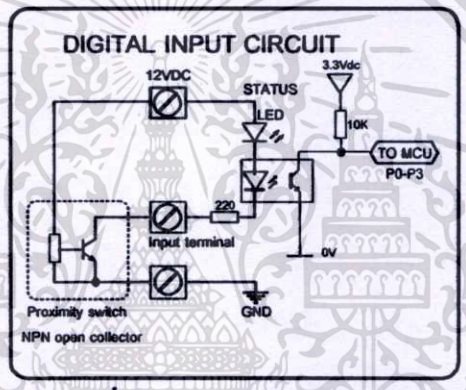
รูปที่ 3.3 บล๊อคไดอะแกมโครงสร้างของ PLCที่ออกแบบจาก CPU LPC2378

การทำงานของ CPU LPC2378 เป็นไมโครคอนโทลเลอร์ที่ใช้ CPU ARM7TDMI-S บรรจุในตัวถังแบบ LQFP 144 มีไฟเลี่ยง 3.3V ความถี่สัญญาฯนาฬิกาที่ใช้คือ 72MHz กำหนดความถี่

สัญญาณนาฬิกาด้วยกระบวนการ PLL จากคริสตอล 12MHz โดยใช้ PLL x 6 มี ระบบฐานเวลาจริง (RCT) ที่ใช้คริสตอลกำลังงานต่ำ เป็นตัวควบคุมฐานเวลา

โดยเฉพาะ จะถูกต่อกับ แบตเตอรี่ลิเธียมขนาด 3V ที่ใช้ร่วมกับ 2 KB BATTERY RAM ทำให้ วงจร RCT ทำงานได้ตลอดเวลาแม้ว่าแหล่งจ่ายไฟหลักจะไม่ได้ต่อใช้งานก็ตาม การดาวน์โหลดโปรแกรม ของ CPU ออกแบบวงจรให้ทำได้ 2 ทางคือ ผ่านทาง In-System Programming ดาวน์โหลดโดยใช้ซอฟแวร์ บูตโหลดเดอร์ภายใน ทำได้โดยการกดสวิทช์รีเซ็ตต่อเข้าที่ขา P2.7 ของ CPU โดยตรง การดาวน์โหลดอีกวิธีหนึ่งทำได้ โดยโหลดข้อมูลผ่านทางพอร์ท UART0 ที่ต่อให้เป็นการสื่อสารแบบ RS-232 สามารถดาวน์โหลดและดีบัก การทำงานระบบปฏิบัติการ OS ของ PLC ได้สะดวก

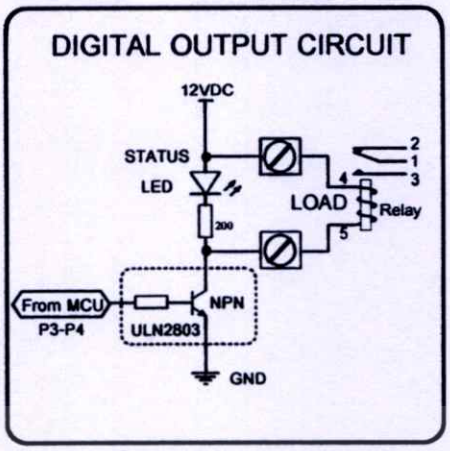
3.1.2 อินพุต และ เอาต์พุต แบบดิจิตอล



รูปที่ 3.4 วงจรภาคอินพุต

อินพุตอยู่ส่วนล่างของเมนบอร์ดซึ่งเป็น DC INPUT แบบ Opto Isolator ดังรูปที่3.4 แบบซอสขนาด 12V เมื่อมีสัญญาณ โลจิกต่ำกระทำเข้ามาทางอินพุต จะทำให้กระแสไฟฟ้าไหลผ่านตัวต้านทาน ผ่านอินพุตของ Opto Isolator ที่เป็น LED และ LED D15 ใช้แสดงสถานะการทำงานของอินพุต และ ส่งผลทำให้ Opto Isolator ทางเอาต์พุตที่เป็นทรานซิสเตอร์ชนิด NPN นำกระแสทำให้เกิดสถานะลอจิกต่ำ ต่อไปยัง CPU เพื่อรับค่าสถานะของอินพุตไปเก็บยังหน่วยความจำอินพุตต่อไป ในการออกแบบจะอาศัย พอร์ท P0 และ P3 ของ CPU LPC2378 เป็นวงจรทางด้านอินพุท

ในการออกแบบจะอาศัย พอร์ท P3 และ P4 ของ CPU LPC2378 เป็นพอร์ทวงจรขับทางด้านเอาท์พุทโดย อุปกรณ์ด้านเอาต์พุตชั้นแรกใช้เป็นทรานซิสเตอร์ชนิดNPN Open Collector แบบอาร์เรย์ ขนาด 8 จุด คือไอซีเบอร์ ULN2308 เป็นจำนวน 2 ตัว ดังรูปที่ 3.5 เพื่อให้เป็นชุดขับกำลังขนาดเล็กในชั้นแรกใช้เปิด/ปิดวงจรไฟฟ้ากระแสตรง ที่มีแรงดันขนาด 12VDC เพื่อนำเอาต์พุตนี้ไปต่อยังชุดขับชั้นที่สองได้แก่ รีเลย์หรือโซลิดสเตทรีเลย์ เพื่อให้มีกำลังขัดโหลดได้มากขึ้น และ อีกประการหนึ่งเพื่อแยกกราวด์ของระบบออกจาก กราวด์ของอุปกรณ์ภายนอกเป็นการป้องกันความเสียหายอันเนื่องมาจากมีการลัดวงจรจากอุปกรณ์หรือสายไฟฟ้าภายนอกนั้นเอง



รูปที่ 3.5 วงจรภาคเอาต์พุต

3.1.3 การสื่อสารข้อมูล

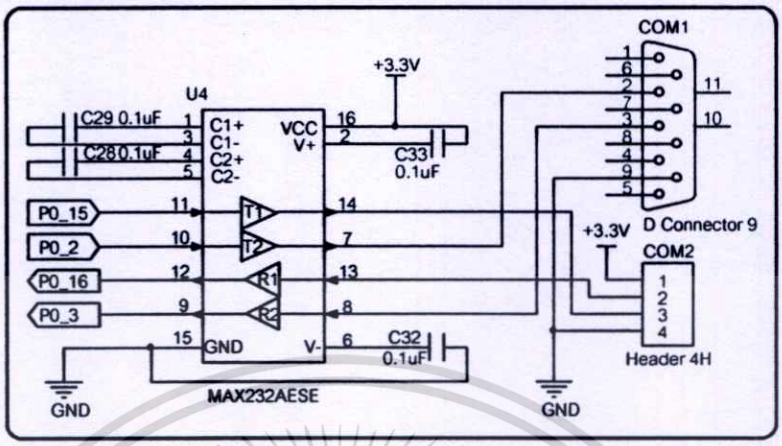
ตามโครงสร้างของ CPU LPC2378 จะมีโมดูลที่ใช้ในการติดต่อสื่อสารอยู่หลายประเภท เป็นระบบบัสที่ใช้ในการติดต่อข้อมูล เช่น I^2C BUS 2 ช่อง, SPI BUS 2 ช่อง, UART 4 ช่อง, CAN BUS 2 ช่อง, USB 1 ช่อง, ETHERNET 1 ช่อง สัญญาณ ซึ่งทั้งหมดเราสามารถเลือกให้มาใช้งานได้ตามความเหมาะสม I^2C BUS และ SPI BUS เป็นบัสที่นิยมใช้ในการติดต่อสื่อสารภายใน สำหรับการออกแบบ ฮาร์ดแวร์ของเครื่องควบคุม PLC ขอกล่าวเฉพาะส่วนที่ออกแบบใช้งานเท่านั้น ดังตารางที่ 3.3

ตารางที่ 3.3 ตารางการใช้งานโมดูลการติดต่อสื่อสารของ CPU LPC2378

ชนิด	การนำใช้งาน
UART0 RS-232	ใช้โหลดโปรแกรม OS และ Debug และใช้ในการติดต่อระยะใกล้ความเร็วต่ำ
UART1 RS-485	ใช้ในการติดต่อระยะไกลความเร็วต่ำ สามารถประยุกต์ใช้เป็น MODBUS ได้ในระดับอุปกรณ์ (Device Network)
UART3 RS-232	ใช้สื่อสารข้อมูลในรูปแบบ PC Link โดยเชื่อมต่อ PLC จำนวน 2 ชุด
CAN BUS	ใช้ในการติดต่อระยะไกลความเร็วสูง ในระดับเครื่องควบคุม (Controller Network)
USB BUS	ใช้ในการติดต่อระยะใกล้ความเร็วสูง 12Mbps ใช้ติดต่อหน่วยความจำ PLC โดยตรงในระดับฮาร์ดแวร์
ETHERNET	ใช้ในการติดต่อระยะไกลความเร็วสูง ใช้เป็น Embedded Web Server ในการติดต่อกับ PLC และสามารถอ่านเขียนหน่วยความจำ PLC ได้โดยตรง

3.1.3.1 การสื่อสารข้อมูลแบบอนุกรมผ่านพอร์ท RS-232 และ RS-485

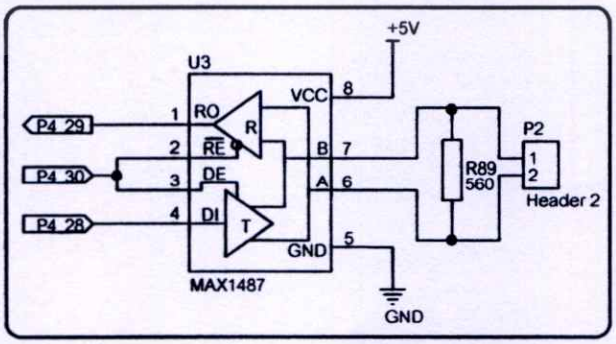
การสื่อสารข้อมูลแบบอนุกรมผ่านพอร์ท RS-232



รูปที่ 3.6 วงจรเชื่อมต่อ UART0, UART3 แบบ RS232

การสื่อสารทางพอร์ตอนุกรมเป็นการสื่อสารแบบ Full duplex คือสามารถรับและส่งได้ในเวลาเดียวกัน ในการออกแบบชุดขับรับส่งข้อมูลเราใช้ไอซี MAX3232 เพื่อแปลงสัญญาณลอจิกของ CPU ให้เป็นมาตรฐาน RS-232 โดยคอนเน็กเตอร์ DSUB 9 PIN ถูกต่อเข้ากับ UART0 นอกจากนี้แล้ว UART-0 ยังสามารถใช้งานเป็น ISP Download เพื่อการดาว์นโหลดข้อมูลแบบไฟล์ไบนารีให้กับ CPU ได้อีกด้วย โดยใช้งานร่วมกับสวิทช์ SW1 เพื่อการดาว์นโหลดไฟล์ และ SW2 ทำการรีเซ็ทให้กับ CPU เพื่อการเริ่มต้นทำงานใน Boot-Loader Mode เครื่องจะดาว์นโหลดข้อมูลแบบไฟล์ไบนารีให้กับ CPU เองแบบอัตโนมัติภายหลังที่กดสวิทช์รีเซ็ท ส่วน UART1 ถูกนำมาเพื่อใช้งานเป็น PC Link ติดต่อสื่อสารระหว่าง PLC จำนวน 2 ชุดในการออกแบบพอร์ทของ CPU ที่ถูกนำมาใช้งาน ได้แก่ พอร์ท P0.2 และ P0.3 เป็น UART0 และ พอร์ท P0.15 และ P0.16 เป็น UART1 ตามลำดับ

การสื่อสารข้อมูลแบบอนุกรมผ่านพอร์ท RS-485

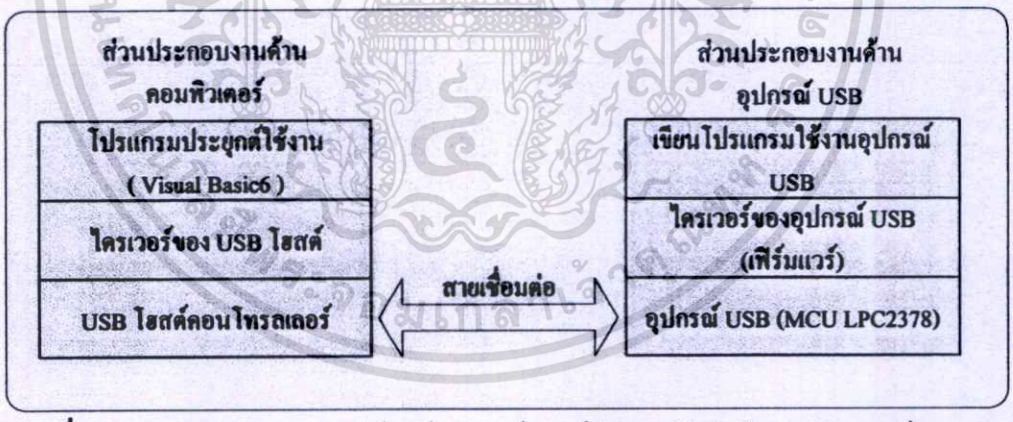


รูปที่ 3.7 วงจรเชื่อมต่อ UART2 แบบ RS-485

RS485 เป็นการรับส่งแบบ Half-Duplex การเขียนโปรแกรมการทำงานให้เป็นแบบโพลลิ่ง (Polling) คือจะกำหนดให้มีชุดทำงานหลักเรียกว่าชุด Master จำนวน 1 ชุดและที่เหลือในเครือข่ายให้เป็นชุด Slave อาจจะมีอยู่หลายๆชุดสูงสุดได้จำนวน 128 โหนด(Node) ชุด Master มีหน้าที่เพื่อคอยจัดคิวและจัดลำดับการสื่อสารในเครือข่ายให้กับชุด Slave โดยที่ Slave แต่ละชุดจะมี ตำแหน่งของตัวเอง ในขณะที่ชุด Master ต้องการจะสื่อสารกับ Slave ชุดใดก็สามารถส่งผ่านโปรโตคอลที่มีตำแหน่งกำกับในส่วนของซอฟท์แวร์โปรโตคอลสำหรับการสื่อสารผ่านRS 485 ยังไม่ได้มีการพัฒนา ส่วนทางด้านฮาร์ดแวร์ ในการออกแบบจะอาศัย พอร์ท P4.28 ถึง30 ของ CPU LPC2378 เป็นพอร์ท UART 2 แปลงเป็นพอร์ท RS485 ใช้ไอซีเบอร์ MAX487 สามารถขับวงจรได้มากสุดถึง 128 โหนดซึ่งเป็นแบบที่สามารถรองรับกับฮาร์ดแวร์ MODBUS ในรูปแบบ Modbus Protocol

3.1.3.2 การสื่อสารข้อมูลผ่านระบบบัสชนิด USB

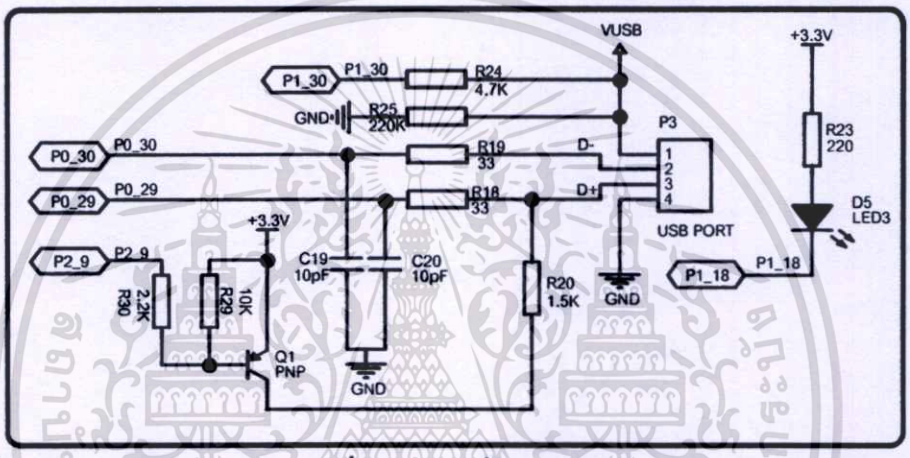
USB (Universal Serial Bus) เป็นบัสอนุกรมอเนกประสงค์ ที่ใช้ในการรับส่งข้อมูลระหว่างเครื่องแม่ข่าย (Host Computer) กับอุปกรณ์ประยุกต์อื่นๆ มีหลากหลายชนิดในงานวิจัยนี้ คือเครื่องควบคุม PLC สามารถเชื่อมต่อเป็นเครือข่ายของ PLC ได้สูงสุดถึง 127 โหนด ในการสื่อสารระคอมพิวเตอร์และเครื่องควบคุม PLC ผ่านระบบบัส USB จะต้องมีฮาร์ดแวร์และซอฟต์แวร์ที่เกี่ยวข้องที่ทำงานร่วมกัน ดังแสดงดังรูปที่ 3.8 โดยแบ่งออกเป็นสองส่วนคือ ส่วนของโฮสต์หรือคอมพิวเตอร์ และ ส่วนของอุปกรณ์หรือเครื่องควบคุมที่มี USB เป็นพอร์ทสื่อสาร



รูปที่ 3.8 แสดงการทำงานซอฟต์แวร์และฮาร์ดแวร์กับการรับส่งข้อมูลผ่าน พอร์ท USB

ทางส่วนของโฮสต์ประกอบด้วย USB Host Controller เป็นฮาร์ดแวร์สำเร็จที่มีอยู่ในชิปเซ็ตของเมนบอร์ดคอมพิวเตอร์ทำหน้าที่ควบคุมการทำงานของพอร์ท USB ส่วนที่สองคือ USB Host Driver เป็นซอฟต์แวร์ไดรเวอร์สำหรับอุปกรณ์ USB ที่จะต้องติดตั้งบนเครื่องคอมพิวเตอร์เพื่อเตรียมความพร้อมก่อนการใช้งาน โดยปกติบริษัทผู้ผลิตอุปกรณ์ USB จะเป็นผู้สร้างไดรเวอร์ให้ และส่วนสุดท้ายคือ แอปพลิเคชั่นหรือโปรแกรมที่เรียกใช้ไดรเวอร์เพื่อเชื่อมต่อกับอุปกรณ์ในที่นี้คือ PLC

ทางส่วนอุปกรณ์ USB แบ่งได้เป็น 3 ส่วนเช่นกัน ได้แก่ USB Device หรืออุปกรณ์ USB เป็นฮาร์ดแวร์ที่ช่วยในการติดต่อกับพอร์ท USB ในที่นี้ก็คือไมโครคอนโทรเลอร์ CPU LPC2378 ที่มีโปรแกรมการทำงานเป็น PLC ส่วนที่สองคือ USB Device Driver หรือไดร์เวอร์ของอุปกรณ์ USB เป็นซอฟต์แวร์ไดร์เวอร์ที่บรรจุอยู่ในหน่วยความจำโปรแกรมของไมโครคอนโทรลเลอร์ CPU LPC2378 และส่วนสุดท้ายคือ USB Device Application หรือ การประยุกต์ใช้งานอุปกรณ์ USB เป็นการเขียนโปรแกรมเพื่อส่งข้อมูลระหว่างหน่วยความจำภายในของ PLC ที่มีไมโครคอนโทรลเลอร์ LPC2378 เป็น CPU ผ่านทางพอร์ต USB ไปแสดงผลที่คอมพิวเตอร์ หรือในทางกลับกัน เพื่อรับข้อมูลจากคอมพิวเตอร์แล้วส่งไปยังหน่วยความจำภายในของ PLC

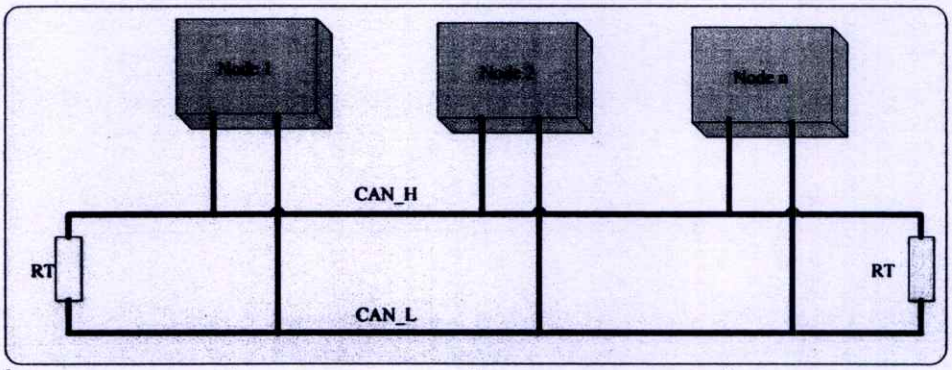


รูปที่ 3.9 วงจรเชื่อมต่อ USB

ขั้นตอนการทำงานของวงจรตามรูป ที่ 3.9 ขา P1.30 จะทำหน้าที่เป็น ขา Vusb ในการตรวจจับแรงดันไฟจากพอร์ท USB ที่ถูกต่อเข้ามาตรงกับตำแหน่งP3 ของ พอร์ท USB จากนั้นกระบวนการทางซอฟต์แวร์ จะสัญญาณมาที่ขา P2.9 มีความหมายคือ USB_CONNECT ที่ขา D+ ของ USB มีตัวต้านทานขนาด1.5 กิโลโอมห์ ต่อร่วมด้วยในวงจร เป็นการแจ้งสถานะให้โฮสรู้ว่าจะทำการติดต่อสื่อสารด้วยความเร็วฟูลสปีด และเมื่อกระบวกการเชื่อมต่อเสร็จสมบูรณ์ ขา P1.8 มีความหมายคือ USB_UP_LED จะแสดงสถานะการเชื่อมต่อได้เสร็จสมบูรณ์แล้ว

3.1.3.3 การสื่อสารข้อมูลผ่านระบบบัสชนิด CAN bus

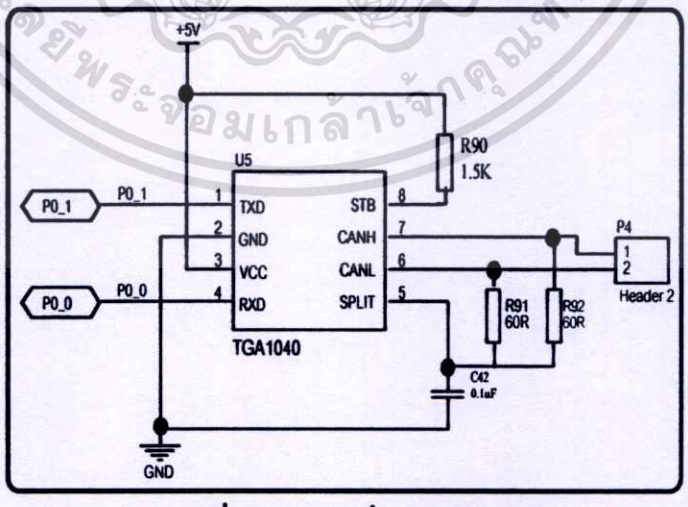
CAN ย่อมาจาก Controller Area Network หมายถึง การควบคุมพื้นที่ด้วยระบบเครือข่าย (การนำคอมพิวเตอร์ตั้งแต่ 2 เครื่องขึ้นไปมาเชื่อมต่อเข้าด้วยกันเพื่อใช้ข้อมูล โปรแกรมหรืออุปกรณ์บางอย่างร่วมกัน) ซึ่ง CAN คือ มาตรฐานการติดต่อสื่อสารแบบอนุกรม โดยการใช้สายตัวนำสัญญาณ2 เส้น ดังรูปที่ 3.10



รูปที่ 3.10 ลักษณะการเชื่อมเครือข่ายระดับเครื่องควบคุมของระบบสื่อสารแบบ CAN

คุณสมบัติของบัสแบบ CAN ที่ได้รับมาตรฐาน ISO 11898. มีที่มาและส่วนประกอบดังนี้

- มาตรฐาน ISO 11898 พัฒนาขึ้นโดย Robert Bosch GmbH
- โครงสร้างการทำงานมี 2 ชั้น คือ ชั้นกายภาพ (Physical Layer) และชั้นเชื่อมโยงข้อมูล (Data Link Layer)
- สื่อสารข้อมูลแบบ อะซิงโครนัส
- สร้างเครือข่ายแบบเรียลไทม์
- ใช้หลักการทำงานเป็น CSMA/CD ในการขอใช้บัส
- ไม่มีการกำหนดหมายเลขที่อยู่กับโหนด
- ส่งข้อมูลครั้งละ 0-8 ไบต์ ด้วยอัตราเร็วสูงสุด 1Mbps
- ต้นทุนต่ำ ใช้งานสะดวก บำรุงรักษาง่าย เป็นที่นิยมให้แพร่หลายในงานอุตสาหกรรม และระบบอิเล็กทรอนิกส์ในยานยนต์

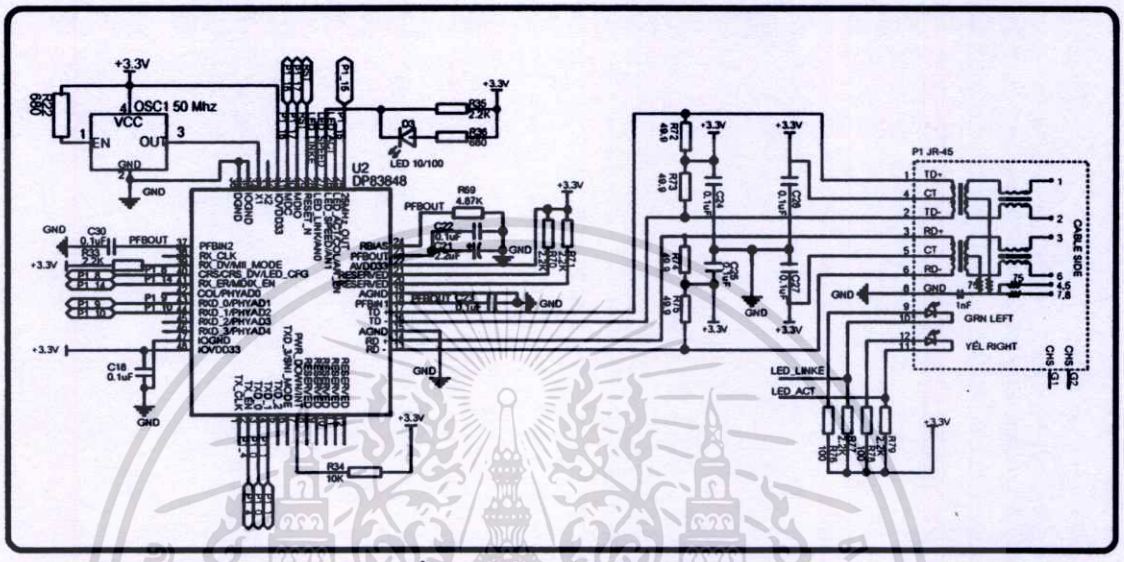


รูปที่ 3.11 วงจรเชื่อมต่อ CAN

ภายใน CPU LPC2378 จากรูปที่ 3.11 มี CAN controller 2 ช่องในการออกแบบถูกต่อออกมาใช้งาน 1 ช่อง คือ CAN1 ทางขา P0.0 และ P0.1

ออกแบบใช้งานรองรับการเชื่อมต่อระยะไกลระดับเครื่องควบคุม ผ่าน CAN BUS โดยออกแบบใช้ Chips Physical CAN ไอซีเบอร์ TGA1040 ของบริษัท NXP เป็นอุปกรณ์หลักในการนำมาออกแบบใช้งาน

3.1.3.4 การสื่อสารข้อมูลผ่านระบบโครงข่ายอินทาเน็ต



รูปที่ 3.12 วงจรเชื่อมต่อ Ethernet

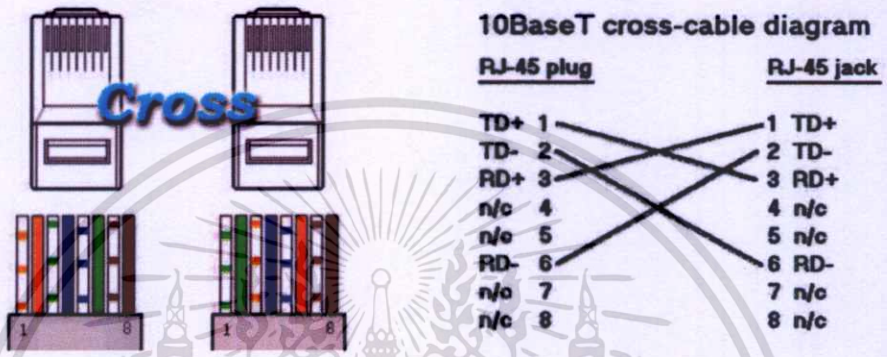
สำหรับการเชื่อมต่อกับเครือข่าย ระหว่างบอร์ดของเครื่องควบคุมสามารถทำได้โดยใช้พอร์ท Ethernet ในการออกแบบจะใช้ขั้วต่อที่รับส่งสัญญาณเข้าออกเป็นขั้วต่อมาตรฐาน แบบ RJ45 โดยวงจรส่วนที่ออกแบบไว้มีขาสัญญาณ P1[0,1,4,8,9,10,14,17] ที่มีคุณสมบัติตามมาตรฐาน Reduced Media Independent Interface PHY Interface – RMII ใช้ขาสัญญาณทั้งหมด 8 ขาที่ จำนวนขาสัญญาณ มีหน้าที่และความหมาย ดังตารางที่ 3.4

ตารางที่ 3.4 ตารางแสดงขาสัญญาณตามแบบ RMII

RMII Pin	# of Pins	Function
TX_EN	1	Transmit data enable
TXD[1:0]	2	Transmit data output
RXD[1:0]	2	Receive data input
CRS_DV	1	Carrier sense / Data valid
RX_ER	1	Receive error (optional, depending on application)
REF_CLK	1	Reference clock input

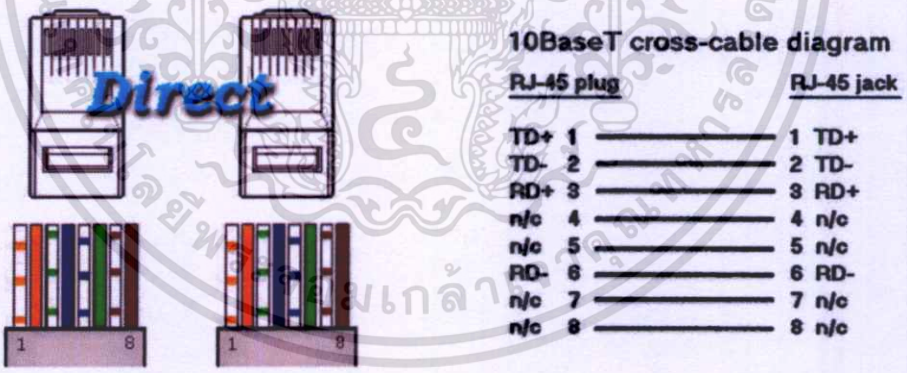
โดยใช้ Chips Physical Ethernet ไอซีเบอร์ DP83848 เป็นตัวขับสัญญาณในการเชื่อมต่อสายสัญญาณเพื่อใช้งานในระบบเครือข่ายสามารถเชื่อมต่อสายสัญญาณ Ethernet LAN ของบอร์ดเครื่องควบคุมเข้ากับระบบเครือข่ายได้ 2 แบบด้วยกัน คือการต่อสายแบบต่อตรง และ การต่อผ่าน Ethernet Hub เพื่อกระจ่ายสัญญาณได้หลายๆจุด

กรณีที่ 1 การต่อแบบต่อตรง เป็นการเชื่อมต่อระหว่างเคอมพิวเตอร์เข้ากับเครื่องควบคุมโดยตรง สายสัญญาณตัวนำจะต้องเข้าสายแบบไข้วกัน



รูปที่3.13 การต่อแบบต่อตรง

กรณีที่ 2 คือ การต่อผ่าน Ethernet Hub ระหว่างเครื่องคอมพิวเตอร์ที่ทำหน้าที่เป็นServer จะต้องเข้าสายสัญญาณให้ตรงกันทั้ง 8 สายสัญญาณ



รูปที่3.14 การต่อผ่าน Ethernet Hub

3.1.4 หน่วยความจำของ PLC

ประเภทหน่วยความจำที่ใช้งานในการออกแบบเครื่องควบคุม PLC แบ่งออกเป็น 3 ชนิด ตามลักษณะการใช้งานได้ดังนี้

- Secure Digital Memory Card/Multi Media Card
- Flash Program Memory (on-chip)
- Static RAM (on-chip)

Secure Digital Memory Card (SD Card)

เป็นหน่วยความจำแบบเขียนแล้วลบใหม่ได้ในปัจจุบันได้รับความนิยมมากโดยเฉพาะอุปกรณ์สารสนเทศสมัยใหม่ ด้วยคุณสมบัติ ขนาดเล็กความจุสูง ราคาถูก โดยในการออกแบบการติดต่อกับหน่วยความจำชนิดนี้ จะถูกเชื่อมต่อกับ CPU ผ่านทาง SD BUS ด้วยวามถี่สัญญาณความเร็ว 6 MHz เก็บข้อมูลในรูปแบบไฟล์ FAT16 ใช้เก็บค่าไฟล์ setup การตั้งค่าสถานะของ PLC และยังใช้เก็บโปรแกรมคำสั่ง ของ PLC ที่เป็นแบบ Text File และ Binary File โดย Text File จะใช้เก็บโปรแกรม PLC ที่เป็นคำสั่งของ PLC รูปแบบ IL (Instruction List) หรือ Mnemonic code และเมื่อเครื่องทำการ Compiler ออกมา เป็น Binary File ก็จะถูกจัดเก็บ ก่อนที่เครื่องจะนำไปใช้งานต่อไป

Flash Program Memory

เป็นหน่วยความจำชนิดแฟลชที่บรรจุอยู่ใน CPU LPC2378 มีความจุขนาด 512 Kbyteใช้สำหรับเก็บโปรแกรมระบบปฏิบัติการที่ควบคุมการทำงานของฮาร์ดแวร์ ทั้งหมดของ PLC ไม่ว่าจะเป็น หน่วยอินพุต/เอาต์พุต หน่วยแสดงผล LCD และหน่วยติดต่อสื่อสารประเภทต่างๆของ PLC

Static RAM

เป็นหน่วยความจำภายใน CPU LPC2378 มีขนาดความจุ 58 Kbyte โดยแบบออกเป็นหน่วยความจำชนิด Nonvolatile SRAM ที่แหล่งจ่ายไฟสำรองเลี้ยงอยู่มีความจุขนาด 2 Kbyte เพื่อเก็บ หรือ คงค่าสถานะไว้ในหน่วยความจำได้ตลอดเวลาถึงแม้จะไม่มีแหล่งจ่ายไฟหลักก็ตาม และอีกส่วนหนึ่งเป็นหน่วยความจำ Static RAMที่มีความจุขนาด 56 Kbyte หน่วยความจำส่วนนี้ไม่มีแหล่งจ่ายไฟสำรอง

3.1.4.1 หน่วยความจำควบคุมระบบปฏิบัติการ (OS)

จะเห็นได้ว่า CPU LPC2378 จะเป็นหน่วยความจำ Static RAM ขนาด 58 Kbyte โดยจัดแบ่งการใช้งาน ตามโปรแกรมส่วนต่างๆ สามารถจัดสรรพื้นที่หน่วยความจำโดยออกแบบตามโปรแกรมที่ใช้งานได้ ดังตารางที่ 1

ตารางที่ 3.5 แสดงการจัดแบบหน่วยความจำภายใน CPU LPC2378

โปรแกรมการทำงาน	ขนาดของหน่วยความจำ (Kbyte)
PLC Data Memory	8K
PLC Program Memory buffer	4K
Memory Card read buffer	2K
Ethernet buffer	16K
LCD display	1K
Nonvolatile SRAM	1K
Operating system	10K

PLC Memory Data เป็นหน่วยความจำในส่วน หน่วยความจำผู้ใช้งาน หน่วยความจำ PLC ที่ใช้เนื้อที่บนหน่วยความจำหลัก ของ CPU ใช้เป็นหน่วยความเสมือน เพื่อใช้ ในการประมวลผล และเก็บสถานะที่ได้จากการประมวลผล เช่น รีเลย์ภายในของ PLC รีเลย์ชนิดพิเศษ ตัวตั้งเวลา ตัวนับ พื้นที่ของหน่วยความจำส่วนนี้มีความจุขนาด 8 Kbyte ที่อยู่ในส่วนของ Static RAM จะได้กล่าวถึงในหัวข้อถัดไป

PLC Memory Program buffer เป็นหน่วยความจำชนิด Static RAM ทำหน้าที่เป็นที่พักข้อมูลโปรแกรมประยุกต์ PLC ขนาด 4 Kbyte(4096 byte) เก็บโปรกรม PLC ได้ 1024 คำสั่ง โดยในหนึ่งคำสั่งจะมีขนาด 4 byte ขั้นตอนการทำงานดังนี้ เครื่องควบคุมจะทำการโหลด Binary File โปรแกรม PLC จากหน่วยความจำชนิด Memory Card ไปจัดเก็บยัง PLC Memory Program buffer ที่เป็นหน่วยความจำชนิด Static RAM ที่มีความเร็วสูงกว่า ในกรณีที่โปรแกรม PLC มีขนาดมากกว่า1024 คำสั่ง ซึ่งมากกว่าขนาด PLC Memory Program buffer โปรแกรมจะทำการอ่านข้อมูลจาก Memory Card ขึ้นมาทำงานครั้งละ1024 คำสั่ง หลังจากทำงานเสร็จตามคำสั่ง ก็จะทำการอ่านโปรแกรมในส่วนที่เหลือขึ้นมาทำงานต่อไป จนกว่าจะจบโปรแกรม

Memory Card read buffer จะทำหน้าที่เป็นตัวพักข้อมูลและเป็น Memory ตัวแปรในการจัดการกับไฟล์ในรูปแบบของ FAT16 ในการติดต่อกับ Memory Card การทำงาน Memory Card จำเป็นต้องทำการอ่านข้อมูล หรือเขียนข้อมูลครั้งละ 1 บล็อก คือขนาด 512 Byte จึงต้องใช้ buffer ในการอ่านเขียนข้อมูล

ส่วนแสดงผล LCD จะมีหน่วยความจำพักข้อมูล ของขนาด 1 KByte สามารถคำนวณได้จาก LCD มีขนาดความยาว 128 จุด แสดงผลสีขาวและดำ ในการเก็บข้อมูลจะเก็บจุดละขนาด 1 บิตดังนั้นตลอดความยาว128 จุดจะต้องใช้หน่วยความจำในการเก็นแนวนอนเท่ากับ 128/8 = 16 Byte ส่วนความกว้าง LCD มีขนาดความกว้าง 64 แถว ดังนั้นต้องให้หน่วยความจำในการเก็บข้อมูลทั้งจอภาพเท่ากับ 16X64 Byte = 1024 Byte หรือขนาดความจุ 1K Byte

หน่วยความจำอีกส่วนหนึ่งได้แก่ Nonvolatile SRAM จากโครงสร้างของ PLC จะมีหน่วยความจำที่จำค่าสถานะไว้ถึงแม้ว่า แหล่งจ่ายไฟหลักจะดับลงก็ตาม เพราะจะได้รับพลังงานจากแหล่งจ่ายไฟสำรอง หรือ แบตเตอร์รีขนาดแรงดันไฟตรง 3V เข้ามาเป็นแหล่งจ่ายแทน หน่วยความจำที่จำค่าสถานะ ได้แก่ หน่วยความจำ HR (Holding relay) และ หน่วยความจำ DM (Data Memory) และยังใช้ร่วมกับสัญญาณนาฬิกาหรือ RCT ทำให้นาฬิกาสามารถทำงานได้ตลอดเวลา หน่วยความจำส่วนนี้มีความจุขนาด 2 K byte

3.1.5 อุปกรณ์อินพุต-เอาต์พุต เสริมการทำงานอื่นๆและแหล่งจ่ายไฟตรง

3.1.5.1 SD/MMC memory card

ความรู้เบื้องต้นเกี่ยวกับ SD การ์ด SD การ์ดเป็นหน่วยความจำแบบเขียนและลบใหม่ได้ชนิดหนึ่งที่ใช้เทคโนโลยีหน่วยความจำแบบแฟลช ชื่อเต็มคือ Secure Digital Card มีลักษณะการทำงานและการติดต่อคล้ายกับการ์ดหน่วยความจำแบบ MMC หรือ Multi Media Card หากแต่ใน SD การ์ดได้บรรจุส่วนการรักษาข้อมูลเข้าไปเพิ่มเติม ดังรูปที่ 3.18 แสดงไดอะแกรมการทำงานของ SD การ์ด จะเห็นว่า มีส่วนประกอบหลัก 2 ส่วนคือ โมดูลหน่วยความจำแบบแฟลชและตัวควบคุมการติดต่อกับ SD การ์ดสามารถกระทำผ่านบัส SD หรือบัส SPI

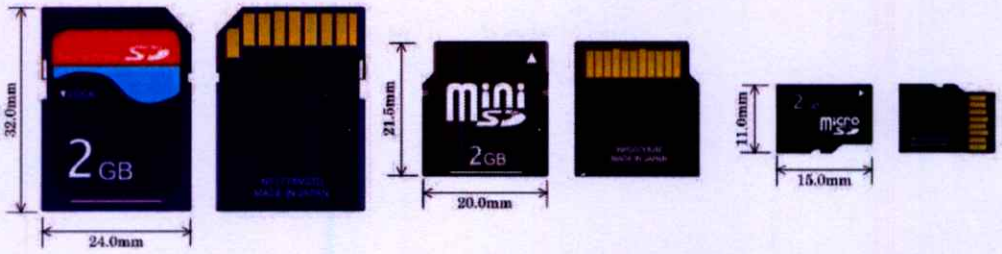
รูปที่ 3.15 แสดงไดอะแกรมการทำงานเบื้องต้นของ SD การ์ด

คุณสมบัติเด่นของ SD การ์ด

SD การ์ดเกิดขึ้นจากความร่วมมือของ 3 บริษัทคือ Matsushita Electric Industrial (MEI), SanDisk Corporation (SanDisk) และ Toshiba Corporation (Toshiba) มีการกำหนดคุณสมบัติต่างๆ รวมถึงมาตรฐานการติดต่อที่ชัดเจนภายใต้การกำกับดูแลโดย SD card Association

ในปัจจุบัน SD การ์ดได้รับความนิยมสูงมาก โดยเฉพาะในอุปกรณ์สารสนเทศสมัยใหม่ๆ ไม่ว่าจะเป็นกล้องดิจิตอล โทรศัพท์เคลื่อนที่ เครื่องเล่น MP3 เป็นต้น ทั้งนี้เนื่องจากการ์ดได้รับการออกแบบให้มีความโดดเด่นในทุกด้านที่หน่วยความจำชั้นดีพึงมี 5 ประการ ดังนี้

1.ขนาดเล็กและบาง (compact & slim)SD การ์ดรุ่นมาตรฐานมีขนาด 24 x 32 มิลลิเมตร และหนาเพียง 2.1 มิลลิเมตร ดังแสดงดังรูปที่ 3.16จะเห็นว่ามีขนาดเล็กและบางมาก จึงทำให้สามารถพกพาได้สะดวก นอกจากนั้นได้มีการพัฒนา SD การ์ด ให้มีขนาดเล็กลงไปอีกเรียกว่า mini SD การ์ด ซึ่งมีขนาดเพียง 20 x 21.5 มิลลิเมตร หนาเพียง 1.4 มิลลิเมตร ซึ่งเล็กและบางกว่ารุ่นมาตรฐานพอสมควรทีเดียว



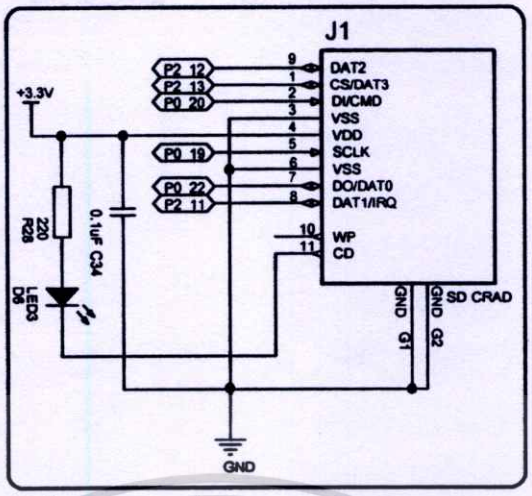
รูปที่ 3.16 แสดงขนาดของ SD การ์ด

2. ความจุสูง (large capacity)แม้ว่า SD การ์ดจะมีขนาดเล็กและบาง แต่นั่นไม่ใช่ข้อจำกัดในด้านความจุ SD การ์ด มีความจุสูงมากเมื่อเทียบกับขนาด โดยในยุคแรกๆของการพัฒนาจะมีความจุ 8MB และได้รับการพัฒนาให้มีความจุสูงขึ้นเป็น 16 และ 32 MB ภายใต้เทคโนโลยีไบนารีแนนด์ (Binary NAND technologies) จากนั้นมีการใช้เทคโนโลยีใหม่ที่เรียกว่า Multi Level Cell (MLC) NAND technologies ทำให้สามารถเพิ่มความจุเป็น 64, 128, 256, 512 MB และไปถึงระดับ GB (ในขณะที่ทำเอกสารนี้มีการพัฒนาถึง 4 GB แล้ว และมีแนวโน้มสูงมากที่จะเพิ่มความจุไปถึง 8 GB) เพื่อให้สามารถรองรับไฟล์ภาพ รวมไปถึงภาพยนต์คุณภาพสูงในระดับ DVD

3. อัตราเร็วในการถ่ายทอดข้อมูลสูง (high – speed data transfer) SD การ์ดมีอัตราการถ่ายทอดข้อมูลในแบบสายสัญญาณเดี่ยวผ่านทางขา DAT0 25 เมกะบิตต่อวินาที ถ้าหากถ่ายทอดข้อมูลแบบ 4 สาย อัตราเร็วจะเพิ่มขึ้นเป็น 100 เมกะบิตต่อวินาที หรือ 12 MB ต่อวินาที

คุณสมบัติทางเทคนิคที่สำคัญของ SD การ์ด

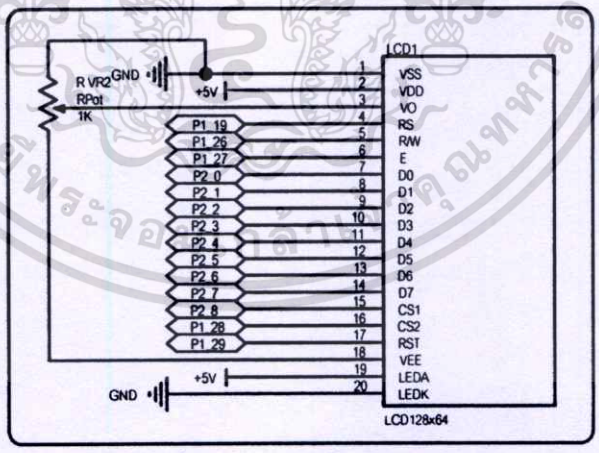
- สามารถเก็บข้อมูลได้ถึง 8 GB (ในขณะจัดทำเอกสารนี้)
- รองรับการติดต่อแบบหนึ่งสายสัญญาณ และแบบ 4 สายสัญญาณ รวมทั้งแบบบัส SPI
- สามารถป้องกันการคัดลอกข้อมูลลิขสิทธิ์ได้
- สามารถลบ – เขียนใหม่ในแต่ละเซกเตอร์ได้ 100,000 ครั้ง
- สามารถเก็บรักษาข้อมูลได้นานมากกว่า 10 ปี
- แรงดันใช้งาน กรณีกระทำคำสั่งสื่อสารเบื้องต้น (CMD0, CMD15, CMD56 และ ACMD41)ใช้ 2.0 ถึง 3.6 Vกรณีกระทำคำสั่งอื่นรวมถึงเข้าถึงหน่วยความจำใช้ 2.7 ถึง 3.6 V
- สามารถปรับสัญญาณนาฬิกาได้ในช่วง 0 ถึง 25 MHz
- อัตราเร็วในการถ่ายทอดข้อมูลสูงสุด 12.5MB ต่อวินาที (ในกรณีใช้การสื่อสารแบบ 4 สาย)
- สามารถติดต่อการ์ดได้พร้อมกันสูงสุด 10 แผ่น
- ป้องกันการเขียนด้วยสวิตช์
- สามารถกำหนดการป้องกันการเขียนได้ทางซอฟต์แวร์ ทั้งแบบชั่วคราวและถาวร
- มีสัญญาณแจ้งการถอดและใส่การ์ดกับซ็อกเก็ต



รูปที่ 3.17 วงจรเชื่อมต่อ SD/MMC memory card

ออกแบบวงจรให้ รองรับการเชื่อมต่อกับการ์ดหน่วยความจำแบบ SD Card และ MMC Card ใช้ในการเก็บข้อมูลโปรแกรมไฟล์ของ PLCโดยในส่วนนี้จะมี LED SD สำหรับแสดงสถานะของการทำงานของการ์ดหน่วยความจำ แหล่งจ่ายไฟของการ์ด หน่วยความจำ จะใช้แหล่งจ่ายไฟ 3.3 V ของบอร์ด โดยวงจรที่ใช้เชื่อมต่อ กับการ์ดหน่วยความจำทั้งหมดจะเลือกขาสัญญาณจาก CPU SD/MMC CARD Interface ซึ่งเป็นโครงสร้างภายในที่ออกแบบมาสำหรับเชื่อมต่อ SD/MMC CARD โดยเฉพาะทำให้สามารถรับส่งข้อมูลด้วยความเร็วสูงกว่าการเชื่อมต่อที่ใช้ใน SPI Bus

3.1.5.2 ส่วนแสดงผล LCD ขนาด 126x64 ตัวอักษร



รูปที่ 3.18 วงจรเชื่อมต่อ LCD

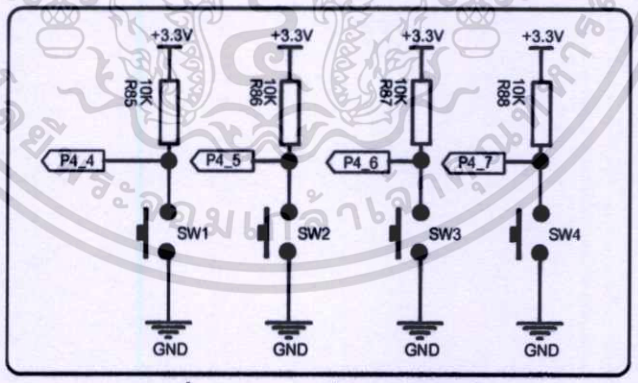
เป็นอุปกรณ์แสดงผลตัวอักษรหรือรูปภาพโดยมีส่วนแสดงผลเป็น DOT ขนาด 128 X 64 สามารถแสดงอักษรภาษาอังกฤษได้ 8 บรรทัด บรรทัดละ 21 ตัวอักษร หรือแสดงรูปภาพกราฟิก การทำงานขนาดไม่เกิน 128 X 64 จุด มีไฟส่องหลัง (Backlight) ช่วยให้เห็นได้ในที่มืดหน้าที่หลักของ LCD คือใช้ในการแสดงเมนู Configuration Program การทำงานของ PLC สามารถเลือกการ

ทำงานในรูปแบบต่างๆ และใช้แสดงโปรแกรมไฟล์ PLC ในรูปแบบ โปรแกรม IL ที่ทำงานตัวอย่างดังรูปที่ 3.19

รูปที่ 3.19 แสดงการทำงานของ LCD

วงจรเชื่อมต่อ LCD 64 x 128 ซึ่งมีไอซีเบอร์ KS0108 จำนวน2 ตัว เป็นตัวควบคุม เชื่อมต่อจากพอร์ท GPI/O ของ CPU LPC2378 โดยตรง โดยใช้ 8 บิทบัสข้อมูล ให้ขาสัญญาณควบคุมอีก 6 บิท การทำงานโดย CPU LPC2378 จะสร้างหน่วยความจำแสดงผลขึ้นมาภายใน ขนาด 1Kbyte ซึ่งมีขนาดเท่ากับจำนวนจุดภาพบนหน้าจอ เพื่อใช้เป็นที่พักข้อมูลแสดงผลภายในที่ทำงานได้เร็วกว่าหน่วยความจำใน LCD หลายเท่า จากนั้นจะทำการพล๊อตหรือแสดงจุดต่างๆ ลงในหน่วยความจำแสดงผล จากนั้นให้ทำการพล๊อตจุดต่างๆจนหมดหน้าจอแล้ว CPUจะทำการส่งข้อมูลทั้งหมดไปยัง LCD ในครั้งเดียว

3.1.5.3 ส่วนป้อนข้อมูลการทำงานแบบคีย์แพ็ต

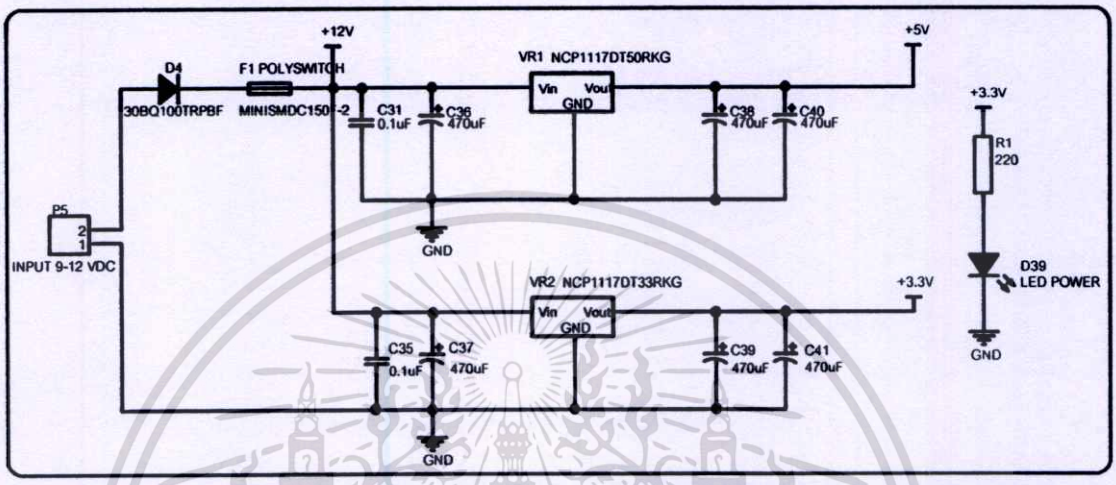


รูปที่ 3.20 วงจรเชื่อมต่อคีย์แพ็ต

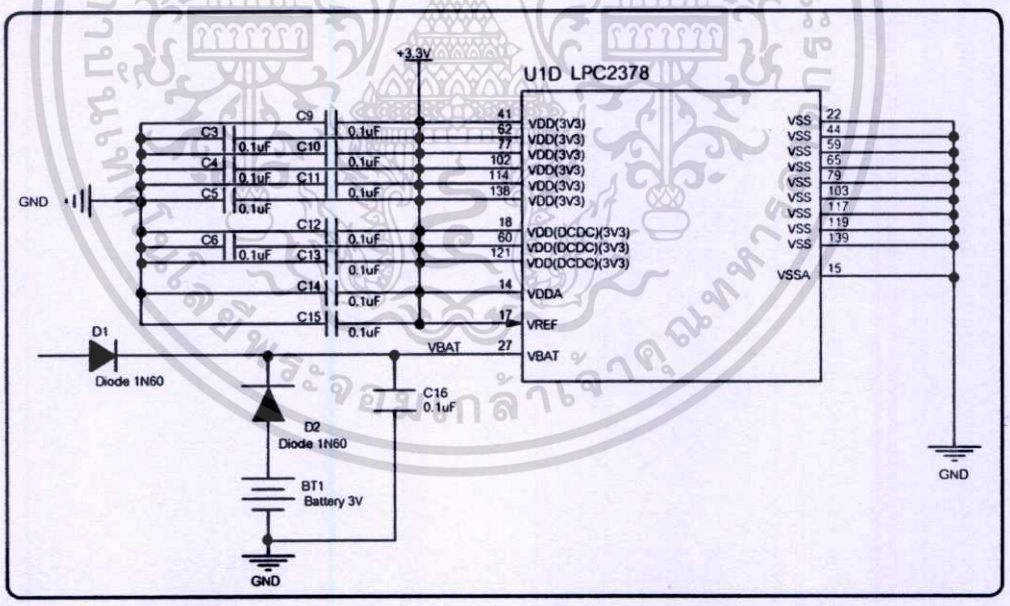
คีย์แพ็ต มีด้วยกันทั้งหมด 4 ปุ่ม คือ ESC, UP, DOWN, ENTERใช้งานร่วมกับ LCD ตามเมนูที่แสดงบนจอแสดงผล LCD กำหนดคีย์ ESC ใช้ในการออกจากเมนูนั้นคีย์ ENTER ใช้ในการตอบตกลงส่วนคีย์ UP, DOWN ใช้ในการเลือกเมนู หรือ ปรับเปลี่ยนค่าตัวเลขต่างๆ

การต่อวงจรสวิทช์ จะถูกต่อเข้าโดยตรงกับพอร์ท ของ CPU LPC2378 ที่พอร์ท P4.4, P4.5, P4.6, P4.7 ตามลำดับ โดยมีตัวต้านทานขนาด 10K Ohm พูลอัพอยู่ ทำให้โลจิกเป็นโลจิก HIGH และเมื่อทำการกดสวิทช์ จะทำให้ GPI/O ของ CPU จะมีสถานะเป็นโลจิก LOW

3.1.5.4 แหล่งจ่ายไฟ (POWER SUPPLY)



รูปที่ 3.21 ภาคจ่ายไฟ 12VDC, 5VDC และ 3.3VDC



รูปที่ 3.22 ภาคจ่ายไฟ 3 VDC สำหรับ RCT และ 2Kbyte RAM

ภาคจ่ายไฟภายใน ฮาร์ดแวร์ของเครื่องควบคุม PLCจะให้แรงดันไฟในการทำงานอยู่ด้วยกัน 4 ระดับ ตามตารางที่ 3.6

ตารางที่ 3.6 แสดงแหล่งจ่ายไฟของวงจรส่วนต่างๆ

แรงดันไฟ	อุปกรณ์ที่ใช้งาน
9-12VDC	อินพุตและเอาต์พุต
5VDC	RS-485,CAN,LCD
3.3VDC	RS-232, CPU, Chips Physical Ethernet, SD/MMC memory card, switch
3VDC	CPU(RCT และ 2Kbyte RAM)

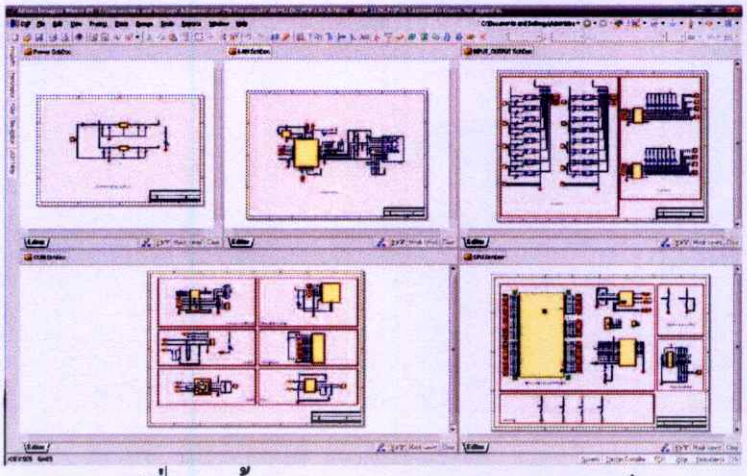
หลักการทำงานของวงจรภาคจ่ายไฟ จะเข้าบอร์ดมาทาง P5 ใช้แรงดันไปขนาด 9-12 VDC ขึ้นอยู่กับการใช้งานวงจรภาคเอาต์พุต ไฟที่เข้ามาจะผ่าน D4 ทำหน้าที่ป้องกันการจ่ายไฟกลับขั้วให้แก่วงจร และต่อมายังมี Poly Switch ทำหน้าที่เป็นOver current Protection Device ซึ่งทำหน้าที่เหมือนกับฟิวส์จะทำการตัดวงจรเมื่อกระแสไฟผ่านตัวมันเกิน 3A และจะทำการต่อใหม่เองโดยอัตโนมัติ และในจุดนี้จะเป็นจุดต่อ ไฟ 12 Vไปยัง วงจรเอาต์พุตเพื่อใช้งานอีกด้วย ส่วนของวงจรแหล่งจ่าไฟ 5 V แรงดันไฟเข้ามีขนาด 12 V จะผ่าน IC Regulators รักษาระดับแรงดันไฟ 5VDC ขนาด 1A Chip NCP1117DT50RKG และรักษาระดับแรงดันขนาด 3.3VDC 3A Chip NCP1117DT33RKG จากภาคจ่ายไฟที่ทำการเรียงวงจรแล้วก็ยังมีภาคจ่ายไฟสำหรับ RCT และ 2Kbyte RAM MEMORY BACK UP ที่ต้องทำงานตลอดเวลาเราจึงเลือกใช้ LITHIUM COIN BATTERY ขนาดแรงดัน 3V มาใช้เป็นแหล่งจ่ายไฟสำรอง ดังรูปที่ 3.22

3.2 การออกแบบแผงวงจรพิมพ์ของ PLC

จากหัวข้อที่ผ่านมา ได้ออกแบบโครงสร้าง ของวงจรเครื่องควบคุม PLC เอาไว้แล้ว ขั้นตอนต่อไปเป็นขั้นตอนของการออกแบบแผ่นวงจร PCB โดยอาศัยโปรแกรม Altium Designe 8 ซึ่งเป็นโปรแกรมที่ช่วยออกแบบงานและวงจรด้านอิเล็กทรอนิกส์ และ ยังสามารถช่วยในการออกแบบ เส้นลายวงจรโดยมีไลบารีของอุปกรณ์ให้เลือกใช้จำนวนมากจึงทำให้มีความสะดวกรวดเร็ว และ มีความคล่องตัวในการออกแบบ PCB มาก

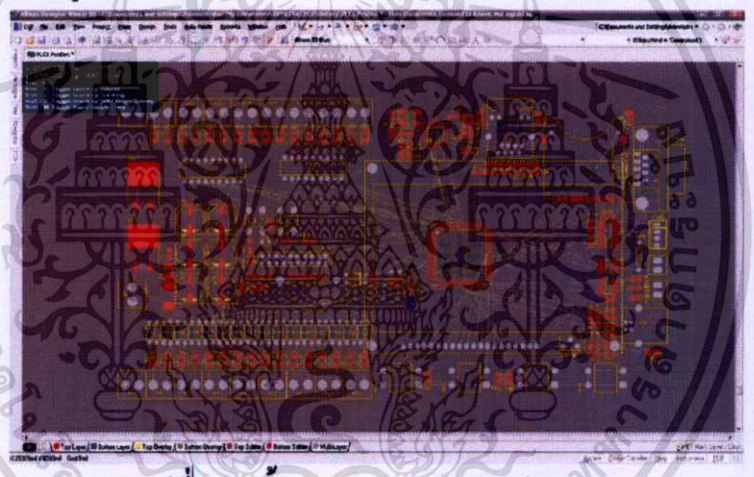
3.2.1 ขั้นตอนและโปรแกรมการออกแบบแผงวงจรพิมพ์

ในการออกแบบเริ่มต้นโดยการสร้าง PCB Project ขึ้นมาให้ชื่อว่า ARMPLC จากนั้นทำการสร้างไฟล์ Schematic ขึ้นมา เพื่อทำการวาดลายวงจร โดยจะแบ่งลายวงจรออกเป็น 5 ส่วนคือเพื่อให้ง่ายแก่การตรวจสอบและการออกแบบแบ่งส่วนงานออกเป็นส่วนวงจรดังนี้ CPU, INPUT OUTPUT, INTERFACE, ETHERNET, POWER ดังรูปที่ 3.23



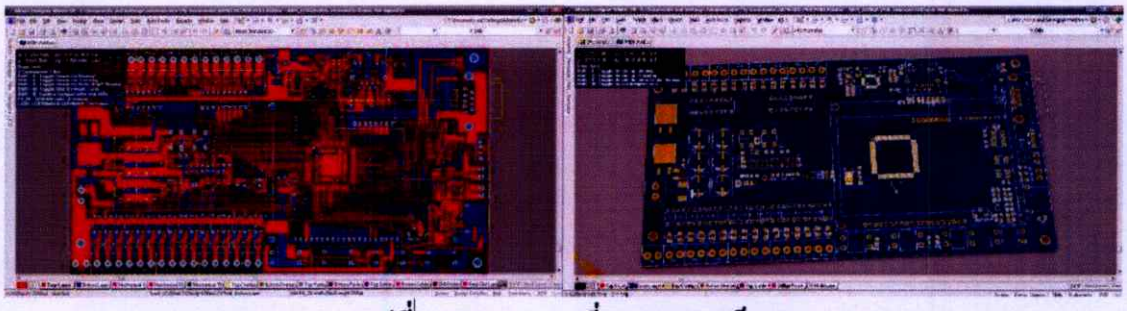
รูปที่ 3.23 ขั้นตอนการออกแบบลายวงจรพิมพ์

จากนั้นทำการกำหนด Footprint ให้กับอุปกรณ์ทุกตัว และทำการ Update PCB จะทำให้ได้ไฟล์ PCB ที่มี Nets เชื่อมโยงขาอุปกรณ์ต่างๆเข้าไว้ด้วยกัน ทำการจัดเรียงอุปกรณ์ต่างๆ ให้อยู่ในตำแหน่งที่ต้องการ ดังรูปที่ 3.24



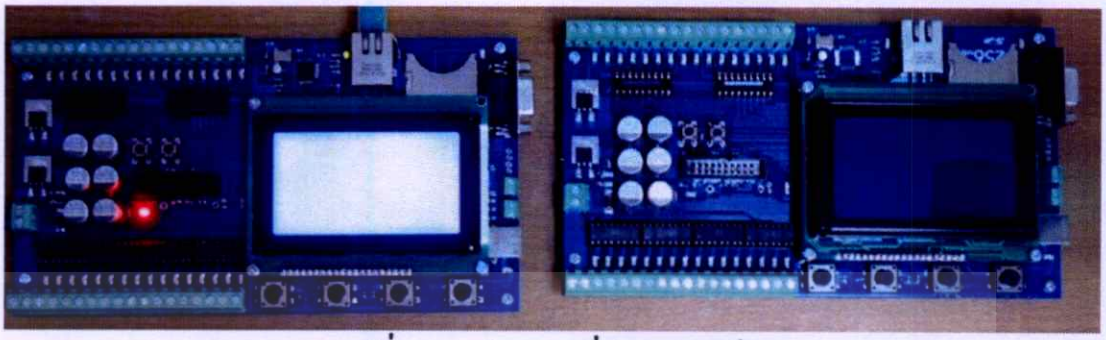
รูปที่ 3.24 ขั้นตอนการออกแบบแผ่น PCB

พอจัดพื้นที่การวางอุปกรณ์ได้แล้วทำการออกแบบ PCB ทั้ง 2 หน้าโดยการออกแบบด้วยตนเองเหตุผลที่ไม่ใช้การออกแบบลายวงจรแบบอัตโนมัติของโปรแกรมเพื่อทำ Auto Route เนื่องจากเหตุผลด้านความสวยและความเป็นระเบียบ ของ ลายเส้นทองแดงในแผ่งวงจร เมื่อทำการออกแบบลายวงจรเสร็จ จะได้ผลงานออกมา ดังรูปที่ 3.25



รูปที่ 3.25 ลายวงจรที่ออกแบบเสร็จ

ทำการส่งไฟล์ไปผลิตได้แผ่น PCB ออกมา ทำการลงอุปกรณ์ตามที่ออกแบบ ได้ออกมา ดังรูปที่ 3.26 สามารถดูลายเอียดเพิ่มเติมได้จากภาคผนวก ก



รูปที่ 3.26 แผ่น PCB ที่ประกอบเสร็จ

สาระสำคัญของบทนี้ได้อธิบายถึงขั้นตอนการดำเนินการวิจัยที่เกี่ยวข้องกับการออกแบบฮาร์ดแวร์การออกแบบวงจรอิเลคทรอนิคส์ ส่วนต่างๆ ของ PLC สำหรับในบทถัดไปจะได้กล่าวถึงความรู้ต่างๆที่เกี่ยวข้องกับเครื่องควบคุม PLC ทั้งในเรื่องหลักการทำงาน การเขียนโปรแกรมตามมาตรฐานสากล IEC 1131-3 มีได้กี่ภาษาและเขียนได้อย่างไร พร้อมตัวอย่าง และอื่นๆที่น่าสนใจเกี่ยวกับเครื่องควบคุมที่ได้ออกแบบ

บทที่ 4

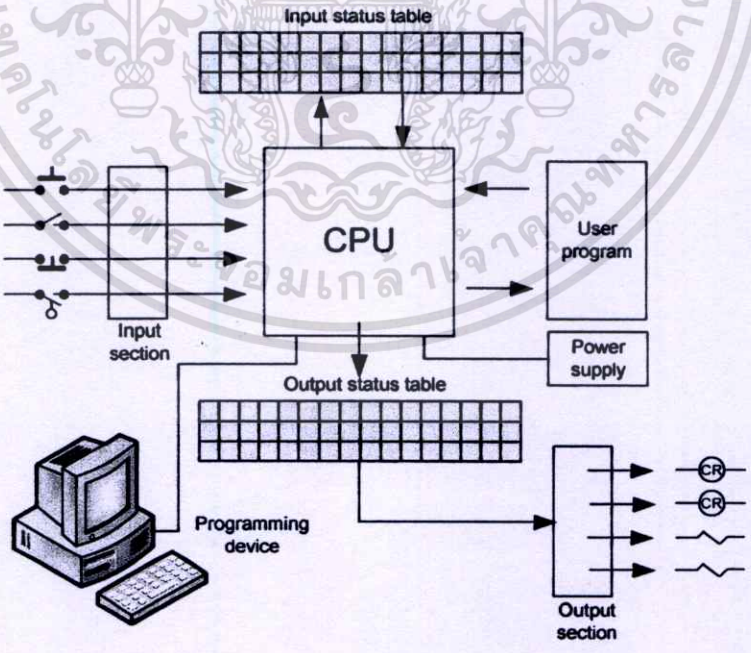
การออกแบบโปรแกรมการทำงานของเครื่องควบคุม PLC

สาระสำคัญของบทนี้จะอธิบายถึงทฤษฎีทั่วไป หลักการทำงานของเครื่องควบคุม ส่วนประกอบและหน่วยความจำPLC การเขียนโปรแกรมให้กับเครื่องควบคุมตามมาตรฐาน IEC 1131-3 และ ขั้นตอนการดำเนินการวิจัยที่เกี่ยวข้องกับการออกแบบตัวแปรภาษาจากโปรแกรม IL ที่เป็นไฟล์ตัวอักษรหรือ Text file ให้เป็น ไฟล์ข้อมูลไบนารี เพื่อป้อนให้กับตัวควบคุมเป็นไปตามโปรแกรมการทำงาน

4.1 หลักการทำงานทั่วไปของ PLC

ในบทนี้จะกล่าวถึงส่วนที่สำคัญทางด้านฮาร์ดแวร์และซอฟท์แวร์ของเครื่องควบคุมที่ได้ออกแบบไว้โดย โครงสร้างพื้นฐานของ PLC มีส่วนประกอบที่สำคัญ แบ่งออกเป็น 5 ส่วน ดังนี้

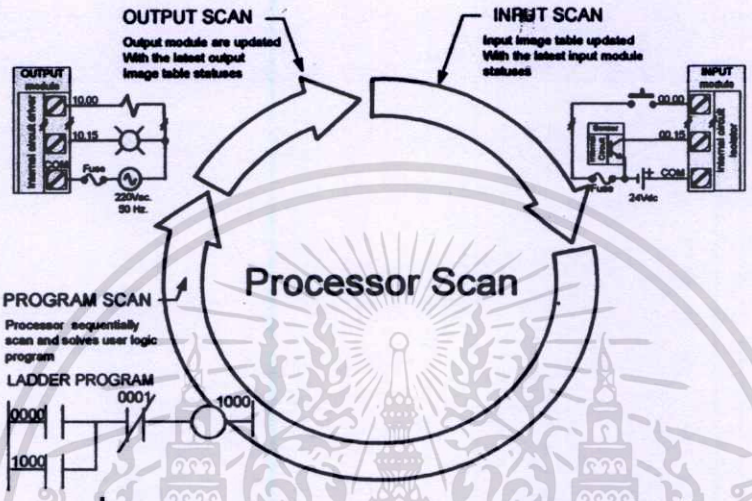
1. หน่วยประมวลผล (CENTRAL PROCESSING UNIT)
2. หน่วยความจำ (MEMORY UNIT)
3. หน่วยอินพุต/เอาต์พุต (INPUT/OUTPUT UNIT)
4. แหล่งจ่ายไฟ (POWER SUPPLY)
5. อุปกรณ์ต่อร่วม (PERIPHERAL DEVICES)



รูปที่ 4.1 แสดงส่วนประกอบฮาร์ดแวร์และอุปกรณ์ต่อร่วมของ PLC

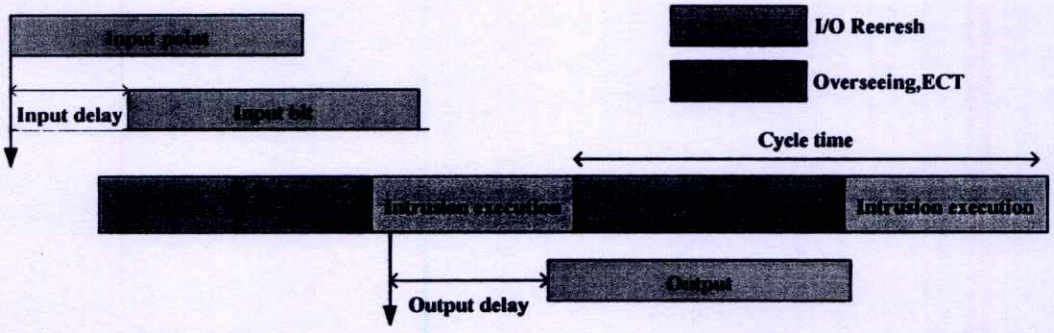
4.1.1 หน่วยประมวลผลกลาง (Central Processing Unit: CPU)

หน่วยประมวลผลกลางทำหน้าที่ในการควบคุมการทำงานของระบบทั้งหมด โดยรับข้อมูลจากหน่วยอินพุตเข้ามาทำการประมวลผลตามโปรแกรมคำสั่งผู้ใช้งาน แล้วนำผลลัพธ์ที่ได้ส่งออกไปหน่วยเอาต์พุต หลังจากนั้นก็จะเข้าไปวนกลับรับข้อมูลอินพุตเข้ามาใหม่อีกครั้ง แล้วจะทำซ้ำในลักษณะเดียวกันนี้ไปเรื่อยๆ จนกว่าจะหยุดการทำงานของ CPU ดังรูปที่ 4.2



รูปที่ 4.2 แสดงการทำงานของ CPU ในแต่ละแสกนไทม์

การทำงานของ CPU ในแต่ละรอบเวลาที่ใช้ในการทำงานเราเรียกว่า ไซเคิลไทม์ (Cycle Time) หรือบางครั้งเรียกอีกอย่างหนึ่งว่า การสแกน (Scan Time) ปัจจัยสำคัญที่มีผลทำให้ สแกนไทม์มีระยะเวลามากน้อยต่างกันมีผลมาจาก ความเร็วของหน่วยประมวลผลกลาง ขนาดของอินพุทเอาท์พุท ขนาดหน่วยความจำ และการเชื่อมอุปกรณ์ต่อร่วมอื่นๆของ PLC ผลตอบสนองทางเวลานี้จะใช้เป็นตัวชี้วัดขีดความสามารถของ CPU ว่าจะตอบสนองต่อการเปลี่ยนแปลงของอินพุตและเอาต์พุตได้เร็วมากน้อยเพียงใด ถ้า CPU มีความเร็วในการตอบสนองที่ครอบคลุมแล้วจะทำให้การควบคุมต่อระบบมีเสถียรภาพ และ สามารถควบคุมการทำงานอยู่ได้ แต่ในทางตรงกันข้ามถ้าความเร็วในการตอบสนองของ CPU ช้าและไม่ทันต่อการเปลี่ยนแปลงของข้อมูลอินพุต เอาท์พุตการควบคุมจะเกิดความผิดพลาดและส่งผลให้การควบคุมขาดความมีเสถียรภาพในทันที



รูปที่ 4.3 การสแกนที่มีผลตอบสนองต่ออินพุต/เอาต์พุต

คำอธิบายความหมายของคำในรูปที่ 4.3

การรับข้อมูลใหม่ของอินพุตและเอาท์พุต (I/O Refresh) หมายถึง ข้อมูลใหม่ของ อินพุตเอาต์พุตจะถูกทำให้เป็นข้อมูลปัจจุบันในทุกรอบการทำงานของโปรแกรม PLC

การตรวจตรา(Overseeing) หมายถึง ระบบตรวจสอบและกำหนดค่าเวลาสแกนไทม์ในแต่ละรอบการทำงานของ CPU

การปฏิบัติคำสั่ง (Instruction Execution) หมายถึงการปฏิบัติคำสั่งตามโปรแกรมผู้ใช้งานที่เขียนให้กับเครื่องควบคุม PLC

กระบวนการของ CPU (CPU Processing) หมายถึงกระบวนการหรือขั้นตอนการทำงานของ CPU

4.1.2 หน่วยความจำ PLC (Memory Unit)

สำหรับพีแอลซีโดยทั่วไปจะแบ่งหน่วยความจำส่วนนี้ออกเป็น 2 ส่วน คือ หน่วยความจำส่วนที่เก็บโปรแกรมของผู้ใช้งาน (User Program Memory) ในรูปแบบโปรแกรม IL (Instruction List) และหน่วยความจำที่เก็บข้อมูลได้จากการประมวลผล บางครั้งเรียกว่า หน่วยความจำเสมือน หน่วยความจำส่วนนี้มีหลากหลายชนิดแต่ละชนิดจะมีขนาดแตกต่างกันไปตามลักษณะการใช้งาน สำหรับบทนี้จะกล่าวถึงหน่วยความจำเสมือนของ PLC แบ่งออกเป็นพื้นที่ส่วนต่างๆของหน่วยความจำไว้ดังนี้

ตารางที่ 4.1 แสดงหน่วยความจำ PLC

Area	Size	Range (word)
Input bits	160 bit	IR000 to IR009
Output bits	160 bit	IR010 to IR019
Work relay (IR Area)	2000 bit	IR106 to IR231
Holding relay (HR Area) (Nonvolatile SRAM)	320 bit	HR 00 to HR 19
Auxiliary relay (AR Area)	256 bit	AR 00 to AR 15
Link relay (LR Area)	256 bit	LR 00 to LR 15
Temporary relay (TR Area)	8 bit	TR 0 to TR 7
Special relay (SR Area)	384 bit	SR232 to SR255
Timers	256 Points	TIM 000 to 256
Counters	256 Points	CNT 000 to 256
Data memory	800 Byte	DM000-DM399
Data memory (Nonvolatile SRAM)	224 Byte	DM400-DM512

โดยหน่วยความจำของ PLC ในแต่ละพื้นที่จะมีหน้าที่แตกต่างกันไปตามการใช้งานดังนี้

Internal Relay (IR) เป็นหน่วยความจำภายในส่วนนี้จะไม่มีการเก็บสถานะถ้าแหล่งจ่ายไฟหลักดับ ส่วนประกอบของหน่วยความจำนี้ได้แก่ หน่วยความจำอินพุต หน่วยความจำเอาต์พุต และหน่วยความจำภายในที่ไม่มีฮาร์ดแวร์รองรับ Work Area ก็ถือว่าเป็นหน่วยความจำภายใน เช่นกัน

ตารางที่ 4.2 พื้นที่หน่วยความจำในพื้นที่หน่วยความจำ IR

หน่วยความจำ IR	เวิร์ด	หน้าที่
Input word	IR000	อินพุตที่ถูกติดตั้งอยู่บนแผงวงจร 16 อินพุต
Input Expansion	IR001- IR009	อินพุตที่รองรับการขยายจำนวนจากภายนอก
Output word	IR010	เอาต์พุตที่ถูกติดตั้งอยู่บนแผงวงจร 16 อินพุต
Output Expansion	IR011- IR019	เอาต์พุตที่รองรับการขยายจำนวนจากภายนอก
Work relay word	IR106 to IR231	ใช้เป็น Work Bit ในการเขียนโปรแกรม

Special Relay (SR) เป็นหน่วยความจำภายในที่มีหน้าที่เฉพาะอย่าง และใช้เก็บสถานะของแฟลก (Flag) ที่เกิดจากการประมวลผลและใช้งานคำสั่งต่างๆ ของ PLC เช่นบิตควบคุมที่ใช้แสดงสถานะการทำงานของคำสั่งเปรียบเทียบ สัญญาณคล็อกพัลส์ (Clock Pulse) และ ค่าความผิดพลาดจากการประมวลผล ตัวอย่างการใช้งานพื้นที่หน่วยความจำในส่วนของ SR มีดังนี้

ตารางที่ 4.3 ตัวอย่างการใช้งานพื้นที่หน่วยความจำในส่วนของ SR

Words	Bits	Function
SR 254	00	1.0 min clock pulse (30 Seconds ON,30 Seconds OFF)
	01	0.2 seconds clock pulse (0.1 Seconds ON ,0.1 Seconds OFF)
SR 255	00	0.1 seconds clock pulse (0.05 Seconds ON ,0.05 Seconds OFF)
	01	0.2 seconds clock pulse (0.1 Seconds ON ,0.1 Seconds OFF)
	02	1.0 seconds clock pulse (0.5 Seconds ON ,0.5 Seconds OFF)
	04	Carry (CY) Flag
	05	Greater Than (GR) Flag
	06	Equals (EQ) Flag
	07	Less Than (LE) Flag
SR 253	13	Always ON Flag
	14	Always OFF Flag
	15	First Cycle Flag

SR 255.00-255.02 เป็นสัญญาณคล็อกพัลส์ สามารถนำสัญญาณเหล่านี้ไปควบคุมการทำงานของเอาต์พุตให้ปิด/เปิด (ON/OFF) เป็นจังหวะได้ตามความต้องการของผู้ออกแบบโปรแกรม

Holding Relay (HR) Area เป็นพื้นที่หน่วยความจำที่ใช้เก็บข้อมูลการใช้งานสามารถเรียกใช้ควบคู่กับคำสั่งระดับบิทและระดับเวิลด์ได้ไม่ต่างจากหน่วยความจำภายใน IR แต่ต่างตรงที่ว่ามีแหล่งจ่ายไฟสำรองทำให้ยังคงค่าสภาวะในหน่วยความจำอยู่ได้แม้แหล่งจ่ายไฟหลักจะดับไป ในการเขียนโปรแกรมแลดเดอร์ให้กับเครื่องควบคุมผู้ใช้สามารถเรียกใช้งานโดยการอ่านข้อมูลได้หลายครั้งไม่จำกัดจำนวนเพื่อให้เป็นไปตามเงื่อนไขของแกรมที่ผู้ออกแบบกำหนดไว้

Auxiliary Relay (AR) Area สำหรับพื้นที่หน่วยความจำส่วนนี้ใช้เก็บค่าเวลา, แฟลกสถานะต่างๆ และบิตควบคุมการทำงาน และรวมถึงหน่วยความจำที่ใช้เป็น RTC (Real Time Clock) ซึ่งมีปฏิทิน นาฬิกาที่แสดง เวลา/วัน/เดือน/ปีและวันที่ เป็นค่าเวลาปัจจุบันสามารถอ่านออกมาแล้วนำไปควบคุมเอาท์พุทให้ทำงานตามวันเวลาที่กำหนด เช่น ต้องการควบคุมการปิดเปิดปั้มน้ำช่วงเวลาเช้า 8.00น. – 9.00 น. แทนที่จะใช้ฟังก์ชั่น Timer/Counter หลายๆตัวมาต่อกันให้เป็นฟังค์ชั่นการควบคุมที่ต้องการ ก็เขียนโปรแกรมอ่านค่าเวลาจาก RTC ได้โดยตรงเพื่อนำมาเปรียบเทียบและควบคุมให้เอาท์พุททำงานและหยุดตามเวลาที่กำหนด การเขียนโปรแกรมร่วมกับ RTC จะได้ทั้งความเที่ยงตรง และ ความสะดวกต่อการเขียนโปรแกรม ทั้งนี้เพราะผู้ใช้งานไม่ต้องเขียนโปรแกรมฟังก์ชั่น Timer/Counter เพราะจะทำให้ซับซ้อนและยุ่งยากกว่ามาก จึงทำให้ฟังค์ชั่น RTC เป็นที่นิยมมากกว่าถ้าต้องการควบคุมเอาท์พุทให้ทำงานเปิดปิดตามเวลาที่กำหนด

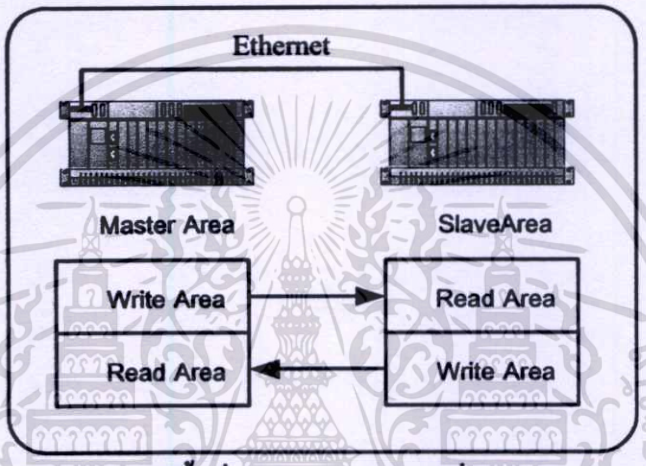
ตารางที่ 4.4 ตารางแสดงค่า Real Time Clock ของ PLC

Words	Bits	Function
AR 17	00 to 07	Minutes : 00 to 59
	08 to 15	Hours : 00 to 23 (24-hour system)
AR 18	00 to 07	Seconds : 00 to 59
	08 to 15	Minutes : 00 to 59
AR 19	00 to 07	Hours : 00 to 23 (24-hour system)
	08 to 15	Day of Month : 01 to 31 (Adjusted by month and for leap year)
AR 20	00 to 07	Month : 1 to 12
	08 to 15	Year : 00 to 99 (Rightmost two digits of year)
AR 21	00 to 07	Day of week : 00 to 06 (00 : Sunday,01 : Monday,02 : Tuesday, 03 : Wednesday, 04 : Thursday, 05 : Friday,06 : Saturday)
	08 to 15	Not used
	13	30 Seconds Compensation Bit

การกำหนดหน่วยความจำ RTC ของ PLC สามารถสังเกตได้จากโครงสร้างภายในของ CPU LPC2378 ดังรูปที่ 3.2 จะมี RTC อยู่ภายในอยู่แล้ว เพียงแต่เรียกหรืออ่านค่าเวลาจริงจาก RTC ภายใน CPU LPC2378 นำมาจัดเรียงเสียใหม่ในหน่วยความจำ AR ของ PLC เท่านั้นและในทุกหนึ่งสแกนไทม์หรือหนึ่งรอบการทำงานของ PLC ก็จะรีเฟรช หน่วยความจำ AR ที่เกี่ยวข้องให้มีข้อมูลเป็นปัจจุบันตามความเป็นจริง

Temporary Relay (TR) หน่วยความจำ TR ใช้ร่วมกับการเขียนโปรแกรม PLC ในกรณีที่มีการแยกสาขา หรือ จุดแยกการทำงานที่มี เงื่อนไขต่อท้าย โดยพื้นที่นี้ใช้กับคำสั่ง LD และ OUT

Link Relay (LR) หน่วยความจำ LR ใช้ในการแลกเปลี่ยนข้อมูลระหว่าง PLC จำนวน 2 เครื่องแบบเครื่องต่อเครื่องดังรูปที่ 4.4



รูปที่ 4.4 ตัวอย่างการใช้พื้นที่หน่วยความจำ LR เมื่อต่อPLC 2 ชุดเข้าด้วยกัน

จากรูปที่ 4.4 เป็นการต่อ PLC สองเครื่องเข้าด้วยกัน PLC เครื่องหนึ่งจะทำหน้าที่เป็น Master ส่วนอีกเครื่องหนึ่งเป็น Slave หน่วยความจำพื้นที่ LR ของPLC ทั้งสองจะใช้งานร่วมกันหมายความว่า เมื่อเขียนข้อมูลลงในหน่วยความจำ LR00-LR07 ของ PLC ที่เป็น Master จะทำให้ LR00-LR07 ของ PLC เครื่องที่เป็น Slave ถูกปรับปรุงหรือ Up Date ข้อมูลตามไปด้วย และในทางกลับกันเมื่อเขียนข้อมูลลงในหน่วยความจำ LR08-LR15 ของ PLC ที่เป็น Slave จะทำให้ LR08-LR15 ของ PLC เครื่องที่เป็น Master ถูกปรับปรุงหรือ Up Date ข้อมูลตามไปด้วยเช่นกันการทำงานนี้จะเป็นไปโดยอัตโนมัติควบคุมโดยระบบปฏิบัติการ เหมาะสมสำหรับระบบที่ต้องการสื่อสารข้อมูลเพื่อทำงานที่สอดคล้องร่วมกันเป็นหนึ่งเดียว หน่วยความจำ LR จะถูกจัดแบ่งในชุด Master และ Slave เพื่อบริหารข้อมูลดังนี้

ตารางที่ 4.5 ตัวอย่างการใช้งานพื้นที่หน่วยความจำ LR

Setting	LR 00 to LR 15
Master words	LR 00 to LR 07
Slave words	LR 08 to LR 15

หน่วยความจำตัวตั้งเวลา (Timer Area :TIM) ใช้สำหรับตั้งค่าเวลาการทำงานให้กับ PLC เพื่อควบคุมการทำงานของเอาท์พุท สำหรับ PLC ที่เป็นงานวิจัยนี้ได้ทำการแยกพื้นที่หน่วยความจำในส่วนของตัวตั้งเวลา และ ตัวนับออกจากกันโดยเด็ดขาด ดังนั้นจึงสามารถใช้ ตัวตั้งเวลา และ ตัวนับหมายเลขเดียวกันได้ การออกแบบฟังค์ชั่นการทำงานของตัวตั้งเวลา ประกอบด้วยหน่วยความจำ 2 ส่วนด้วยกันได้แก่ หน่วยความจำที่ทำหน้าที่เก็บค่าของเวลาที่ต้องการหน่วงมีขนาด 15 บิท และหน่วยความจำแสดงสถานะการทำงานของตัวตั้งเวลามีขนาด 1 บิท

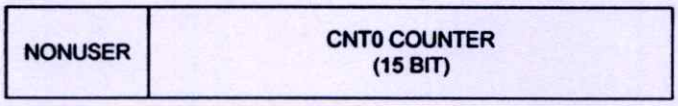
โครงสร้างหน่วยความจำ Timer

โครงสร้างของ FLAG Timer แสดงสถานการณ์ทำงานของ Timer

รูปที่ 4.5 โครงสร้างหน่วยความจำตัวตั้งเวลา

ในการตั้งค่าเวลาจะทำได้กับข้อมูล 15 บิต จะสามารถตั้งค่าหน่วยความจำได้ค่าตั้งแต่ 0-32767 และ 1 บิท จะทำหน้าที่เซ็ตข้อมูลให้เป็น 1 หรือ ON เมื่อต้องการให้ ตัวตั้งเวลานับถอยหลัง ฐานเวลาที่ใช้ในการนับลดลงทุกๆ 1 ms. ได้มาจากการสร้างสัญญาณอินเตอร์รัพท์ของ Timer 0 ภายในตัว CPU LPC2378 ที่ทำการอินเตอร์รัปต์ 100 ms เมื่อโปรแกรมเข้าสู่โปรแกรมอินเตอร์รัพท์ โปรแกรม จะทำการตรวจสอบ บิตที่ 15 ของตัวตั้งเวลาทั้ง 256 ช่องคือตั้งแต่ TIM0 จนถึงTIM255 ว่ามีบิตใดมีสถานะเป็นON บ้าง ถ้ามีแสดงว่าตัวตั้งเวลาในช่องนั้นๆมีการขอเรียกใช้งาน ก็จะทำการนับลดค่าในตัวตั้งเวลาช่องนั้นลง ถ้าค่าในตัวตั้งเวลา มีค่านับลดจนเท่ากับ 0 ก็จะทำการเซ็ตค่า แฟล็กแสดงสถานะตัวตั้งเวลา (TIMER FLAG) ให้เป็น 1หรือ ON แสดงว่าสถานะของตัวตั้งเวลาในช่องนั้นๆ นับครบตามเวลาที่กำหนดแล้ว

หน่วยความจำตัวนับ (Counter Area :CNT) เป็นฟังค์ชั่นที่ใช้เกี่ยวกับการนับ มีโครงสร้างการทำงานคล้ายกับตัวตั้งเวลา แต่มีหน่วยเป็นจำนวนครั้งการนับ โดยมีหน่วยความจำที่ทำหน้าที่เก็บค่า Counter และมีหน่วยความจำที่เก็บสถานะหรือแฟล็กของตัวตั้งเวลา



โครงสร้างหน่วยความจำ Counter

โครงสร้างของ FLAG Counter แสดงสถานการณ์ทำงานของ Counter

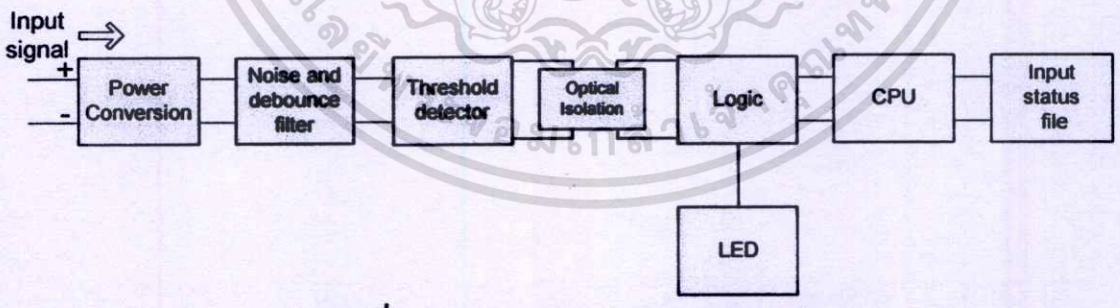
รูปที่ 4.6 โครงสร้างหน่วยความจำตัวนับ

ฟังค์ชั่น การทำงานของตัวตั้งเวลาเมื่อใช้คำสั่ง CNT จะมีอินพุต 2 อินพุต คือ สัญญาณนับพัลส์และสัญญาณรีเซ็ต การจับสัญญาณการนับและรีเซ็ทจะตรวจสอบสัญญาณที่ขอบขาขึ้น โดยจะทำการนับทุกๆขอบขาขึ้นของสัญญาณนับจนครบตามจำนวนที่ตั้งไว้ การนับเป็นแบบนับขึ้น ก็จะทำให้ แฟล็กของตัวนับ (CNT FALG) มีสถานะเป็น 1 หรือ ON และถ้าเมื่อใดมีสัญญาณรีเซ็ตเข้ามาก็จะทำให้ ค่าของตัวนับมีข้อมูลการนับเป็นศูนย์ในขณะเดียวกัน ค่าของแฟล็กของตัวนับ (CNT FALG) มีสถานะเป็น 0 หรือ OFF ด้วยเช่นกัน

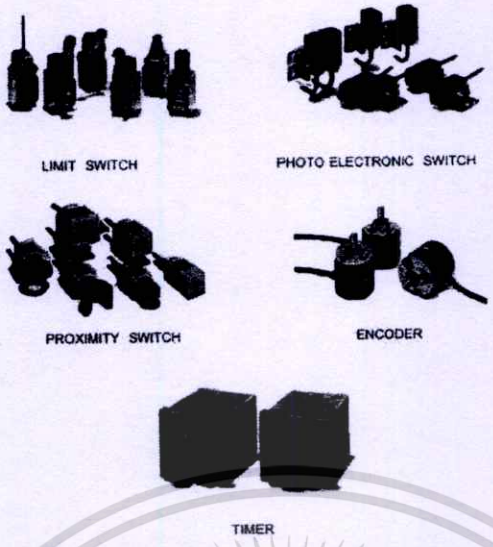
หน่วยความจำ DM (Data Memory) เป็นหน่วยความจำที่ใช้ในการเก็บข้อมูลจากการประมวลผลตามโปรแกรม มีความจุของหน่วยความจำขนาด 512 Word สามารถอ่าน (Read) และ เขียน (Write) ข้อมูลลงไปได้ และสามารถเปลี่ยนแปลงค่าขณะที่ PLC กำลังทำงานอยู่ได้ พื้นที่หน่วยความจำส่วนนี้ยังแบ่งออกเป็น 2 ส่วนคือ ส่วนที่อยู่ในหน่วยความจำ RAM DM000-DM399 และ Nonvolatile SRAM DM400-DM512 ที่จะเก็บข้อมูลได้ในขณะที่ไฟดับ

4.1.3หน่วยอินพุต/เอาต์พุต (Input/Output Unit)

หน่วยอินพุต (Input Unit) ทำหน้าที่รับสัญญาณจากอุปกรณ์ด้านอินพุต ภายนอก เช่น สวิตช์ และตัวตรวจจับต่างๆ แล้วแปลงชนิดของสัญญาณอินพุตภายนอกไม่ว่าจะเป็น AC, DC ให้เป็นสัญญาณที่เหมาะสมผ่านวงจรกรองสัญญาณ และ วงจรช่องผ่านสัญญาณ เพื่อส่งต่อเข้าไปให้กับหน่วยประมวลผลกลาง โดยเชื่อมต่อผ่าน Opto Couple ที่เชื่อมต่อลำแสงด้วยไดโอดเปล่งแสงและรับสัญญาณโดยโฟโต้ทรานซิสเตอร์ ทั้งนี้เพื่อเป็นการแยกสัญญาณ (Isolated) ทางไฟฟ้าให้ออกจากกันเป็นการป้องกันไม่ให้หน่วยประมวลผลได้รับความเสียหายเมื่อเกิดการลัดวงจร สำหรับหน่วยอินพุตที่กล่าวถึงนี้เป็นหน่วยอินพุตแบบดิจิตอล ดังรูปที่ 4.7 และรูปที่ 4.8 จะเป็นตัวอย่างอุปกรณ์ทางด้านอินพุตที่นำมาต่อร่วมด้วย



รูปที่ 4.7แสดงวงจรการทำงานของหน่วยอินพุต

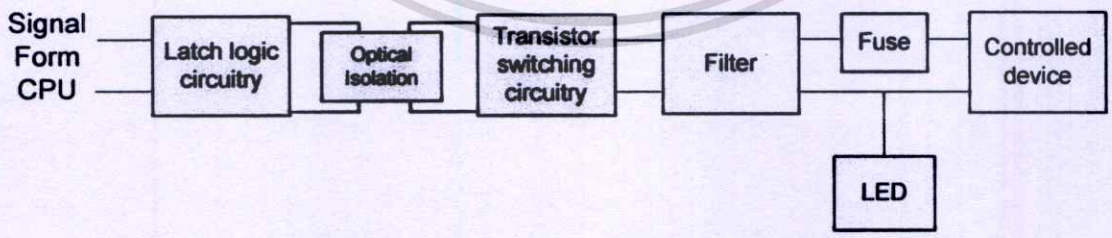


รูปที่ 4.8 แสดงตัวอย่างอุปกรณ์ทางด้านอินพุต

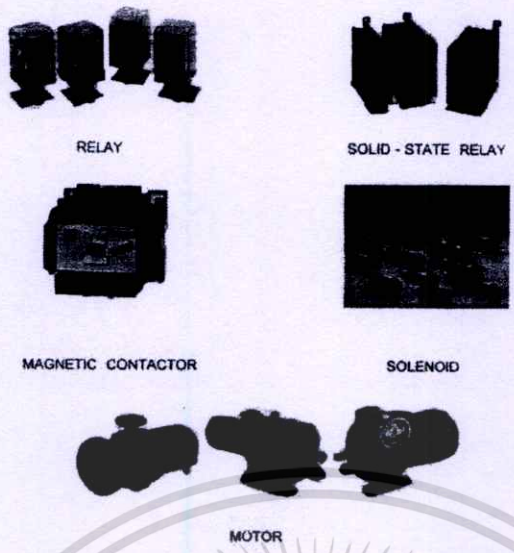
หน่วยเอาต์พุต (Output Unit) ทำหน้าที่ในการการรับค่าสภาวะ ที่ได้จากการประมวลผลของหน่วยประมวลผล ผ่านวงจรค้างค่าสภาวะ และวงจร Isolated จากนั้นนำค่าสภาวะไปควบคุมอุปกรณ์ชุดขับทางด้านเอาต์พุตภายนอก เช่น รีเลย์ โซลินอยด์ ดังรูปที่ 4.9 หน่วยเอาต์พุตของ PLC จะมีอยู่ด้วยกันหลายแบบให้ผู้ใช้เลือกใช้ และการเลือกใช้จะต้องเลือกให้เหมาะสมกับลักษณะการใช้งาน เช่น

เอาต์พุตแบบรีเลย์ ใช้งานกับไฟฟ้ากระแสสลับหรือไฟฟ้ากระแสตรง โดยปกติเอาต์พุตแบบรีเลย์สามารถขับโหลดด้วยกระแสประมาณ 2 แอมแปร์ ในกรณีที่โหลดต้องการกระแสใช้งานมากกว่านี้ผู้ใช้จะต้องนำไปต่อกับอุปกรณ์ขับหรือขยายอีกทีหนึ่ง เช่น รีเลย์โซลิดสเตทรีเลย์หรือแมกเนติกคอนแทคเตอร์

เอาต์พุตแบบทรานซิสเตอร์ ใช้งานที่มีการ เปิด-ปิด บ่อยๆ ของไฟฟ้ากระแสตรง เพราะมีความเร็วในการทำงานสูง ใช้ในการขับโหลดที่เป็นวงจรอิเล็กทรอนิกส์ เช่น การแสดงผล 7-SEGMENT หรือต่อกับไดร์เวอร์ เพื่อขับสเต็ปปิ้งมอเตอร์ เซอร์โวมอเตอร์ เป็นต้น



รูปที่ 4.9 แสดงวงจรการทำงานของหน่วยเอาท์พุต



รูปที่ 4.10 แสดงอุปกรณ์ทางด้านเอาต์พุต

4.1.4 แหล่งจ่ายกำลังไฟ (Power Supply)

ทำหน้าที่ในการแปลงระดับแรงสูงจากแหล่งจ่ายไฟภายนอกไม่ว่าจะเป็นแหล่งจ่ายไฟสลับ 100-240 Vac. 50 Hz. หรือ แหล่งจ่ายไฟตรง 24 Vdc. ให้เป็นแรงดันไฟตรงระดับต่ำ 5 Vdc. และ 3.3 Vdc. เพื่อจ่ายไฟให้แก่อุปกรณ์อิเล็กทรอนิกส์ภายในชุดหน่วยประมวลผลกลาง หน่วยอินพุต / เอาต์พุต หรืออุปกรณ์อื่นๆ ที่ประกอบกันเป็นวงจรการใช้งานของหน่วยประมวลผลกลาง การทำงานของวงจรแหล่งจ่ายไฟฟ้าเป็นแบบวงจรสวิตช์ชิ่ง ที่มีสัญญาณรบกวนต่ำสามารถจ่ายกระแสไฟให้กับโหลดหรือภาระได้คงที่ นอกจากนี้ยังมีแหล่งจ่ายไฟตรงแรงดัน 24 Vdc. นับว่าเป็นระดับแรงดันไฟที่ใช้กันมากในงานอุตสาหกรรม เพื่อจ่ายกำลังไฟฟ้าให้กับอุปกรณ์อินพุตภายนอกของ PLC ไม่ว่าจะเป็น สวิทช์ และ ตัวตรวจจับประเภทต่างๆ เป็นต้น

4.1.5 อุปกรณ์ต่อร่วม (Peripheral Devices)

เครื่องป้อนโปรแกรม (Programming Device) ในที่นี้คือคอมพิวเตอร์ที่ใช้งานร่วมกับโปรแกรมที่พัฒนาขึ้นบนภาษา C มีหน้าที่ป้อนโปรแกรม หรือ เขียน ปรับปรุง และ ตรวจสอบโปรแกรมผู้ใช้งานกับหน่วยความจำของเครื่องควบคุม PLC นอกจากนั้นแล้วยังทำหน้าที่ติดตามผลการควบคุมและการปฏิบัติงานของ PLC ว่าเป็นไปตามเงื่อนไขหรือความต้องการของผู้ใช้งานหรือไม่อีกด้วยดังรูปที่ 4.1

4.2 ภาษาในการเขียนโปรแกรมให้กับเครื่องควบคุม PLC ตามมาตรฐาน IEC 1131-3

มีองค์กรอิสระที่จัดตั้งขึ้นมีอยู่หลายกลุ่มที่กำหนดมาตรฐานประเภทต่างๆ แต่ในที่นี้ขอกล่าวเฉพาะ International Electrotechnical Commission (IEC) IEC1131-3 เป็นมาตรฐานทางด้านโปรแกรมให้กับเครื่องควบคุม PLC ถือกำเนิดขึ้นในปี คศ.1993 โดยวางมาตรฐานสำหรับเขียนโปรแกรมไว้ถึง 5 ภาษา นั้น

หมายความว่า เครื่องควบคุมใดที่อยู่ในมาตรฐานของ IEC 1131-3 จะสามารถเขียนโปรแกรมประยุกต์ได้ถึง 5 ภาษามีดังต่อไปนี้

- STRUCTURED TEXT (ST)
- INSTRUCTION LIST (IL)
- LADDER DIAGRAM (LD)
- FUNCTION BLOCK DIAGRAM (FBD)
- SEQUENTIAL FUNCTION CHART (SFC)

4.2.1 STRUCTURED TEXT (ST)

เป็นโปรแกรมภาษาโครงสร้างที่นักเขียนโปรแกรมบนคอมพิวเตอร์ ที่นักโปรแกรมเมอร์ ค่อนข้างจะคุ้นเคยกันดี เพราะเป็นโปรแกรมภาษาชั้นสูง (High Level) ที่ประกอบด้วยคำสั่ง เช่น IF...THEN , CASE , FOR , WHILE , REPEAT , UNTIL เหล่านี้เป็นต้น โปรแกรม ST ตามมาตรฐานแล้วเปรียบเทียบได้กับการเขียนโปรแกรมภาษา C นั้นเอง รูปแบบโดยทั่วไปของโปรแกรม STRUCTURED TEXT (ST) ประกอบด้วยสิ่งต่อไปนี้

```
PROGRAM ProgNAME
  VAR_INPUT
    (*list each input variable and its data type* )
  END_VAR
  VAR_OUTPUT
    (*list each output variable and its data type* )
  END_VAR
  VAR
    (*list each internal  variable and function block used within the program* )
  END_VAR
    (*main program body*)
END_PROGRAM
```

4.2.2 INSTRUCTION LIST (IL)

เป็นอีกภาษาหนึ่งที่ใช้ในการโปรแกรมให้กับเครื่องควบคุม PLC ในรูปแบบของคำสั่ง บูลลีน (Boolean Instruction) ซึ่งถือได้ว่าเป็นภาษาขั้นต่ำหรือล่างสุดของการเขียนโปรแกรมให้กับ PLC เมื่อเปรียบเทียบแล้วจะใกล้เคียงกับภาษาแอสเซมบลี (Assembly)ของไมโครโปรเซสเซอร์มากข้อได้เปรียบของภาษานี้คือความเร็วในการประมวลผลของชุดคำสั่งให้กับเครื่องควบคุมทำได้รวดเร็ว เพราะไม่ต้องเสียเวลาตีความมากแต่ก็มีข้อเสียตรงที่ความเข้าใจในความหมายของคำสั่งและโปรแกรมระหว่างเครื่องควบคุมกับผู้ใช้งานทำได้ยาก จึงทำให้การตีความหมายเมื่อเปรียบเทียบแล้วค่อนข้างจะด้อยกว่าภาษาอื่น

ที่อาศัยคอมพิวเตอร์แสดงการทำงานของโปรแกรมในรูปแบบกราฟฟิคจะสื่อความหมายให้ผู้ใช้งานให้เข้าใจได้ง่ายกว่า ภาษาเหล่านี้ได้แก่ แลดเดอร์ไดอะแกรม(Ladder Diagram) , ฟังค์ชั่นบล็อกไดอะแกรม (Function Block Diagram) และ ซีเค้วนฟังชั่นชาร์ท (Sequence Function chart) เป็นต้น สำหรับงานวิจัยนี้จะนำเสนอวิธีการเขียนโปรแกรมด้วยภาษานี้เป็นหลัก

ตารางที่ 4.6 ชุดคำสั่งของ Instruction List (IL)

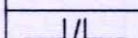
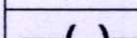
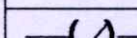
Instruction Operator	Modifiers	Operand Type	Description
LD	N	Any	Load operand into accumulator (e.g.LD%IX1)
ST	N	Any	Store accumulator into operand (e.g.ST%QX2)
S		Boolean	Set operand to 1 (TRUE)
R		Boolean	Set operand to 0 (FALSE)
AND	N and ( )	Boolean	Logical AND
OR	N and ( )	Boolean	Logical OR
XOR	N and ( )	Boolean	Exclusive OR
ADD	( )	Any	Addition
SUB	( )	Any	Subtraction
MUL	( )	Any	Multiplication
DIV	( )	Any	Division
GT	( )	Any	Greater than comparison operator
GE	( )	Any	Greater than or equal to comparison operator
EQ	( )	Any	Equal to comparison operator
LE	( )	Any	Less than or equal to comparison operator
LT	( )	Any	Less than comparison operator
RET	C,N		Return form function or function block

Modifiers are : N = NOT,( ) = deferred execution and C= Conditional execution. Instruction that can be used with any data type are said to be over loaded. The accumulator is a CPU data register

4.2.3 LADDER DIAGRAM (LD)

เป็นภาษาหนึ่งที่ใช้ในการโปรแกรมให้กับเครื่องควบคุม PLC และจัดได้ว่าเป็นที่นิยมใช้กันค่อนข้างมาก ทั้งนี้เพราะว่าการเขียนโปรแกรมแทบจะคล้ายกับการเขียนไดอะแกรมของ วงจรรีเลย์ ซึ่งเป็นที่คุ้นเคยกันดี หรือกล่าวได้อีกนัยหนึ่งคือ ผู้ที่มีความชำนาญการเขียนไดอะแกรมวงจรรีเลย์ จึงไม่ใช่เรื่องยากเลยที่จะเขียนโปรแกรมแลดเดอร์ อีกประการหนึ่งแลดเดอร์โปรแกรมนี้สามารถแปรความหมายกลับไปมา-กลับมา กับ Instruction List ได้สะดวกและตรงตัวมากกว่าภาษาอื่นๆ

ตารางที่ 4.7 ตัวอย่างชุดคำสั่งและสัญลักษณ์ของ Ladder Diagram

IEC Standard ladder graphical symbols	Description
	Normally open contact
	Normally close contact
	Coil : The coil set from the left-hand link
	Negative Coil : The negative coil is set to the opposite state to that of left-hand link

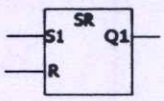
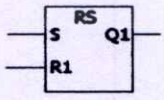
ตารางที่ 4.7(ต่อ) ตัวอย่างชุดคำสั่งและสัญลักษณ์ของ Ladder Diagram

IEC Standard ladder graphical symbols	Description
—(S)—	SET coil : The coil variable is set to the ON state when the left-hand link is ON. The coil remains ON (i.e. latched) until it is reset using the RESET coil
—(R)—	RESET coil: The coil variable is reset to OFF(i.e. cleared) when the left-hand link is ON. The coil remains OFF until it is set using the SET coil

4.2.4 FUNCTION BLOCK DIAGRAM (FBD)

เป็นอีกภาษาหนึ่งที่ใช้ในการโปรแกรมให้กับเครื่องควบคุม PLC นิยมกันมากในบริเวณแถบยุโรป จะสังเกตได้จากเครื่องควบคุมที่ผลิตจากแถบดังกล่าวจะต้องเขียนโปรแกรมได้ด้วย FBD อย่างน้อยหนึ่งภาษา ลักษณะของภาษาที่ใช้เขียนจะเป็นการนำเอาบล็อกที่มีฟังค์ชั่นการทำงานแบบต่างๆมาจับเรียงต่อกันจากซ้ายไปขวา ในรูปกราฟฟิค การจัดเรียงมีลักษณะคล้ายการออกแบบวงจรดิจิตอล ซึ่งทำให้ง่ายต่อการเขียน แก้ไข ตรวจสอบ โปรแกรมได้ไม่ต่างจาก Ladder diagram ดังนั้นความนิยมของโปรแกรมในแต่ละพื้นที่ ที่แตกต่างกันนั้นมีปัจจัยหลักมาจากความสามารถของโปรแกรมที่จะเอื้ออำนวยความสะดวก และ การเผยแพร่อย่างต่อเนื่องของเทคโนโลยีในท้องถิ่นนั้น

ตารางที่ 4.8 ตัวอย่าง ชุดคำสั่ง FUNCTION BLOCK DIAGRAM

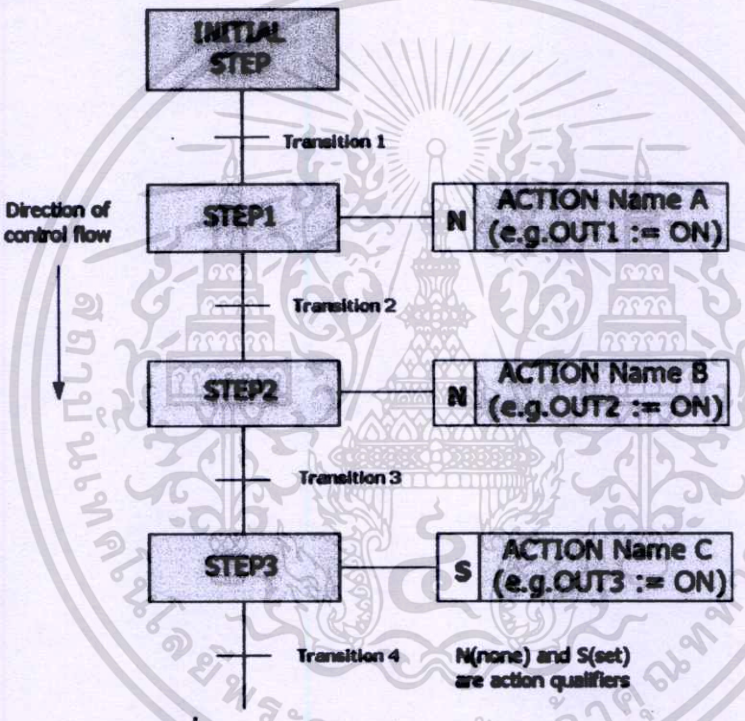
IEC Standard function block	Label description
SR bistable 	The SR bistable is a latch where the set (S1) input takes priority. S1 = Set input of type BOOL R = reset input of type BOOL Q1 = latch output of type BOOL
RS bistable 	The RS bistable is a latch where the set (R1) input takes priority. S = Set input of type BOOL R1 = reset input of type BOOL Q1 = latch output of type BOOL

4.2.5 SEQUENTIAL FUNCTION CHART (SFC)

เป็นภาษาที่ใช้เขียนโปรแกรมแบบกราฟฟิคให้เครื่องควบคุม PLC ที่ควบคุมเป็นแบบลำดับ ก่อนอื่นต้องทำความเข้าใจก่อนว่าการควบคุมแบบลำดับนั้นคืออะไร การควบคุมแบบลำดับนั้นเป็นกระบวนการทำงานที่มีขั้นตอนแน่นอนตายตัว โดยแบบกระบวนการหลักให้เป็นกระบวนการย่อยที่ A,B,...จนกระทั่งถึง

กระบวนการย่อยที่ N และการทำงานจะต้องไล่ลำดับขั้นก่อนและหลังกันลงมา กระบวนการที่ A จะเริ่มต้นด้วยเงื่อนไขที่ผู้ใช้กำหนดอาจจะเป็นการกดสวิทช์สตาร์ท เพื่อเข้าสู่กระบวนการ A (Transition1) เมื่อกระบวนการที่ A ทำงานให้มีการเปลี่ยนแปลงทางด้านเอาท์พุตจนเสร็จกระบวนการที่ A

จะมีเงื่อนไขของชุดตรวจจับบอกสถานะให้เปลี่ยนลำดับขั้น (Transition2) ซึ่งเป็นการจบกระบวนการที่ A พอดี แต่เป็นการเริ่มต้นกระบวนการที่ B เช่นกันกระบวนการที่ B ก็เริ่มทำงานให้มีการเปลี่ยนแปลงทางด้านเอาท์พุต จนเสร็จกระบวนการที่ B จะมีเงื่อนไขของชุดตรวจจับบอกสถานะให้เปลี่ยนลำดับขั้น (Transition3) ซึ่งเป็นการจบกระบวนการที่ B พอดี แต่เป็นการเริ่มต้นกระบวนการที่ C การทำงานจะอยู่ในลักษณะนี้จนกระทั่งถึงกระบวนการที่ N เป็นอันว่าเสร็จกระบวนการทำงานหลักและถือว่าจบกระบวนการพอดี



รูปที่ 4.11 แสดงการเขียน Sequential Function Chart

รูปที่ 4.11 เป็นการแสดงการเขียน Sequential Function Chart ในมาตรฐานของ IEC 1131-3 โปรแกรมภาษานี้ก็เป็นที่นิยมในแถบยุโรปเช่นกันจะเห็นได้ว่าแผนภูมิของโปรแกรมดูแล้วสื่อความหมายกับผู้ใช้งานได้ง่ายจนรู้สึกว่าการทำงานแบบลำดับนั้นเป็นเรื่องไม่ยากเลย

4.3 โปรแกรมการทำงานระบบปฏิบัติการ RTOS

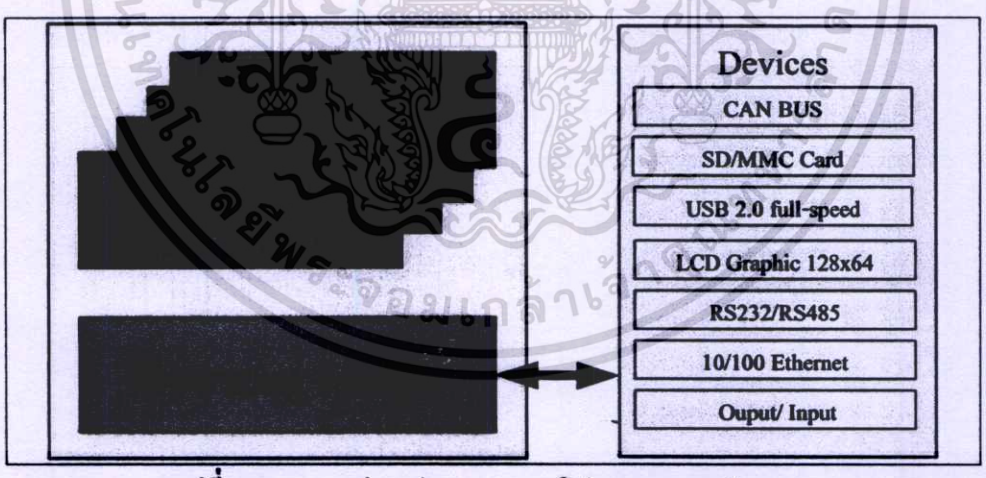
ระบบปฏิบัติการของเครื่องควบคุม PLC ทำหน้าที่บริการจัดการ โปรแกรมควบคุมการทำงานทุกภาคส่วนของฮาร์ดแวร์ให้สามารถทำงานสอดประสานกันชนิดสมบูรณ์แบบและมีประสิทธิภาพอย่างเป็นระบบ ในการพัฒนาซอร์ฟแวร์ระบบปฏิบัติการของเครื่องควบคุมจะอาศัยโปรแกรม Real View Real-Time Library

เป็นเครื่องมือช่วยในการพัฒนา โดยที่นี้เป็นโปรแกรมทำงานบนระบบปฏิบัติการ MS Windows ของตัวแปลภาษาแบบ C ซึ่งเป็นภาษาระดับสูง ตามมาตรฐาน (ANSI Standard C) ระบบปฏิบัติการของเครื่องควบคุมจะทำงานคล้ายกับระบบปฏิบัติการของคอมพิวเตอร์ ที่สามารถทำงานได้หลายๆโปรแกรมประยุกต์ในช่วงเวลาเดียวกัน เราเรียกระบบนี้ว่าระบบปฏิบัติการเวลาจริง (Real-time OS) หรือ มัลติทาสกิ้งเคอร์เนล (multitasking kernel) สามารถทำงานได้หลายๆ ทาสพร้อมกันและให้แล้วเสร็จในเวลาที่กำหนด

ตามที่เคยกล่าวมาแล้วในบทที่ 2 ตัวอย่างการทำงานของเครื่องควบคุมหลักที่มีระบบปฏิบัติการเวลาจริง แบ่งส่วนย่อยการทำงานที่เป็นอิสระต่อกันของการ ควบคุมโปรแกรมประยุกต์ผู้ใช้งาน ควบคุมการทำงาน Embedded Web Server ผ่านพอร์ท Ethernet และ การทำงานอื่นๆ ของเครื่องควบคุมหลักจะต้องทำงานให้สอดคล้องกันของโปรแกรมย่อยทั้งหมดร่วมเป็นหนึ่งเดียว และที่สำคัญจะต้องตอบสนองการเปลี่ยนแปลงข้อมูลเป็นแบบเรียลไทม์

ในการออกแบบโปรแกรมการทำงานของระบบสมองกลฝังตัวสามารถแบ่งโปรแกรมการทำงานของระบบปฏิบัติการที่ประกอบด้วยโปรแกรมย่อยดังนี้

- การออกแบบฐานข้อมูล และ ชุดคำสั่งของ PLC
- โปรแกรมตัวแปลงภาษา และ สร้างรหัสการทำงาน
- โปรแกรม Embedded Web Server ของ Ethernet
- โปรแกรมประยุกต์การใช้งาน



รูปที่ 4.12 แสดงส่วนประกอบของโปรแกรมการทำงาน

4.3.1 การออกแบบฐานข้อมูล ชุดคำสั่ง IL ที่สัมพันธ์กับโปรแกรม LD

ในงานวิจัยนี้ได้พัฒนาเอาระบบสมองกลฝังตัวมาประยุกต์ให้เป็นเครื่องควบคุม PLC ดังนั้นในการพัฒนาระบบปฏิบัติการจะอาศัยคอมพิวเตอร์และโปรแกรม RTX Real-Time Kernel ทำงานบน MS Windows เป็นเครื่องมือช่วยในการพัฒนา เครื่องมือดังกล่าวจะเป็นตัวแปลภาษา C (C compiler) ตามมาตรฐาน ANSI (ANSI Standard C) ให้เป็นภาษาเครื่องของซีพียูด้วย นอกจากนี้ระบบปฏิบัติการจะต้องรองรับ

โปรแกรมประยุกต์ผู้ใช้งานของเครื่องควบคุมที่ได้ออกแบบไว้ด้วยภาษา หรือ คำสั่ง Instruction List (IL) แต่ก่อนอื่นผู้ใช้งานจะต้องสร้างโปรแกรม Ladder Diagram (LD) แบบ Manual ขึ้นมาก่อน เพราะ LD เป็นโปรแกรมรูปภาพที่มนุษย์เข้าใจได้ง่าย จากนั้นค่อยแปลงเป็นคำสั่ง IL ซึ่งถือว่าเป็นภาษาระดับสูงกว่าภาษาเครื่อง(Machine Language) สำหรับเครื่องควบคุมซึ่งเป็นระบบสมองกลฝังตัวจะใช้ ภาษาแอสแซมบี้ล และ ภาษาเครื่อง เป็นตัวแปลภาษาสั่งการให้เครื่องควบคุมทำงานซึ่งถือว่าเป็นภาษาระดับล่างสุดที่เครื่องเข้าใจและพร้อมที่จะนำข้อมูลไปทำการประมวลผล

ดังนั้นผู้ใช้งานจำเป็นต้องอาศัยล่ามในที่นี้คือตัวแปลภาษาจึงจะสามารถสั่งการทำงานให้กับเครื่องควบคุมได้ตัวแปลภาษานี้มีหน้าที่แปลคำสั่ง IL ที่อยู่ในรูปไฟล์ตัวอักษรและแปลให้เป็นกลุ่มคำสั่งภาษาเครื่องที่อยู่ในรูปไฟล์ไบนารี ให้สามารถทำหน้าที่ตามที่ผู้ใช้งานต้องการ

การออกแบบฐานข้อมูลของ PLC จะอาศัยชุดคำสั่ง IL เป็นหลักโดยคำสั่งทั้งหมดมีอยู่ 49 คำสั่ง สามารถจัดแบ่งกลุ่มคำสั่งได้ดังนี้

- กลุ่มคำสั่งพื้นฐานระดับบิท
- กลุ่มคำสั่งควบคุมระดับบิท
- กลุ่มคำสั่งตัวตั้งเวลา ตัวนับ
- กลุ่มคำสั่งการเลื่อนข้อมูลระดับบิท
- กลุ่มคำสั่งการเคลื่อนย้ายและเปรียบเทียบเวิลด์ข้อมูล
- กลุ่มคำสั่งแปลงข้อมูลและคำนวนทางคณิตศาสตร์
- กลุ่มคำสั่งสื่อสารข้อมูล

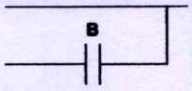
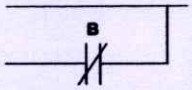
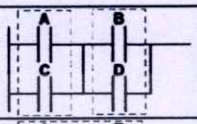
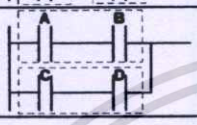
4.3.1.1 กลุ่มคำสั่งพื้นฐานระดับบิท

เป็นกลุ่มคำสั่งอ่านและร่วมถึงการปฏิบัติคำสั่งลอจิกระดับบิตในที่นี้จะกล่าวควบคู่กันไประหว่างคำสั่ง IL และโปรแกรม หรือ สัญญาลักษณ์ LD มีดังต่อไปนี้

ตารางที่ 4.9 กลุ่มคำสั่งพื้นฐานระดับบิต

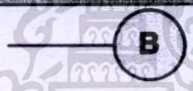
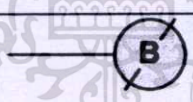
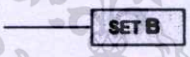
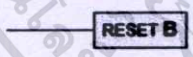
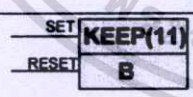
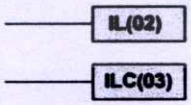
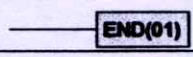
ลำดับ	คำสั่ง IL	สัญญาลักษณ์ LD	พื้นที่หน่วยความจำ	ความหมาย
1	LOAD-LD	B	B : IN BIT IR,SR,AR,HR,TC,LR	เริ่มต้นโปรแกรมคำสั่งอ่านสภาวะปัจจุบัน B รอการกระทำทางลอจิก
2	LOAD NOT – LD NOT	B	B : IN BIT IR,SR,AR,HR,TC,LR	เริ่มต้นโปรแกรมคำสั่งอ่านสภาวะตรงข้ามกับปัจจุบัน B รอการกระทำทางลอจิก
3	AND	B	B : IN BIT IR,SR,AR,HR,TC,LR	คำสั่งที่กระทำทางลอจิก AND กับค่าปัจจุบันของ B
4	AND NOT	B	B : IN BIT IR,SR,AR,HR,TC,LR	คำสั่งที่กระทำทางลอจิก AND กับค่าตรงข้ามกับปัจจุบันของ B

ตารางที่ 4.9 (ต่อ) กลุ่มคำสั่งพื้นฐานระดับบิต

ลำดับ	คำสั่ง IL	สัญญาลักษณ์ LD	พื้นที่หน่วยความจำ	ความหมาย
5	OR		B : IN BIT IR,SR,AR,HR,TC,LR	คำสั่งที่กระทำทางลอจิก OR กับค่าปัจจุบันของ B
6	OR NOT		B : IN BIT IR,SR,AR,HR,TC,LR	คำสั่งที่กระทำทางลอจิก OR กับค่าตรงข้ามกับปัจจุบันของ B
7	AND LD			คำสั่งกระทำลอจิก ANDในรูปแบบของกลุ่มคำสั่ง หรือ บล็อก
8	OR LD			คำสั่งกระทำลอจิก ORในรูปแบบของกลุ่มคำสั่ง หรือ บล็อก

4.3.1.2 กลุ่มคำสั่งควบคุมระดับบิต

ตารางที่ 4.10 กลุ่มคำสั่งควบคุมระดับบิต

ลำดับ	คำสั่ง IL	สัญญาลักษณ์ LD	พื้นที่หน่วยความจำ	ความหมาย
9	OUT		B : IN BIT IR,HR,LR	คำสั่งเขียนสภาวะปัจจุบันลงหน่วยความจำเพื่อจบใน Rung นั้นๆ
10	OUT NOT		B : IN BIT IR,HR,LR	คำสั่งเขียนสภาวะตรงข้ามปัจจุบันลงหน่วยความจำเพื่อจบใน Rung นั้นๆ
11	SET		B : IN BIT IR,HR,LR	คำสั่งเขียนเซ็ทหน่วยความจำบิท B ให้มีค่าเป็น "1"
12	RESET		B : IN BIT IR,HR,LR	คำสั่งเขียนรีเซ็ทหน่วยความจำบิท B ให้มีค่าเป็น "0"
13	KEEP(11)		B : IN BIT IR,HR,LR	คำสั่งเซ็ทและรีเซ็ทหน่วยความจำบิท B ให้มีค่าเป็น "1" และ "0" ตามลำดับ
14	IL(02) , ILC(03)			คำสั่งที่ใช้งานร่วมกันเพื่อกำหนดการควบคุมหลักและยกเลิกการควบคุมหลัก
15	DIFU(13)		B : IN BIT IR,HR,LR	คำสั่งเขียนที่ขอบขาขึ้น(Leading Edge) ลงหน่วยความจำ B
16	DIFD(14)		B : IN BIT IR,HR,LR	คำสั่งเขียนที่ขอบขาขึ้น(Tailing Edge) ลงหน่วยความจำ B
17	END(01)			คำสั่งจบการทำงานโปรแกรม

4.3.1.3 กลุ่มคำสั่งตัวตั้งเวลา ตัวนับ

ตารางที่ 4.11 กลุ่มคำสั่งตัวตั้งเวลา ตัวนับ

ลำดับ	คำสั่ง IL	สัญญาลักษณ์ LD	พื้นที่หน่วยความจำ	ความหมาย
18	Timer – TIM	TIM N SV	N(#): T number SV(word BCD): IR,HR,LR,#	Timer ตัวที่ N หน่วงเวลาแบบ (Delay ON) กำหนดคาบเวลาด้วย SV
19	Counter – CNT	SET RESET CNT N SV	N(#):C number SV(word BCD): IR,HR,LR,#	Counter ตัวที่ N แบบนับลง (Count down) กำหนดค่าการนับด้วย SV
20	Reverse Counter CNTR (12)	UP DOWN RESET CNTR(12) N SV	N(#):C number SV(word BCD): IR,HR,LR,#	Reverse Counter ตัวที่ N แบบนับขึ้น-ลง (Up-Down) กำหนดค่าการนับด้วย SV

4.3.1.4 บางส่วนกลุ่มคำสั่งการเลื่อนข้อมูลระดับบิต

ตารางที่ 4.12 บางส่วนกลุ่มคำสั่งการเลื่อนข้อมูลระดับบิต

ลำดับ	คำสั่ง IL	สัญญาลักษณ์ LD	พื้นที่หน่วยความจำ	ความหมาย
21	ASL(25)	ASL(25) Wd	Wd: Source wd IR,HR,LR	Arithmetic Shift Left with carry การเคลื่อนข้อมูลแบบบิททางซ้ายในเวิลด์ wd
22	ASR(26)	ASR(26) Wd	Wd: Source wd IR,HR,LR	Arithmetic Shift Right with carry การเคลื่อนข้อมูลแบบบิททางขวาในเวิลด์ wd
23	ROL(27)	ROL(27) Wd	Wd: Source wd IR,HR,LR	Rotate Left with carry การวนข้อมูลแบบบิททางซ้ายในเวิลด์ wd
24	ROR(28)	ROR(28) Wd	Wd: Source wd IR,HR,LR	Rotate Right with carry การวนข้อมูลแบบบิททางขวาในเวิลด์ wd

4.3.1.5 บางส่วนกลุ่มคำสั่งการเคลื่อนย้ายและเปรียบเทียบเวิลด์ข้อมูล

ตารางที่ 4.13 บางส่วนกลุ่มคำสั่งการเคลื่อนย้ายและเปรียบเทียบเวิลด์ข้อมูล

ลำดับ	คำสั่ง IL	สัญญาลักษณ์ LD	พื้นที่หน่วยความจำ	ความหมาย
25	MOVE-MOV(21)	MOV(21) Source wd Destination wd	S: Source wd IR,SR,AR,HR,LR,TC,DM,# D: Destination wd IR,HR,LR,DM	คำสั่งเวิลด์ทำการสำเนาข้อมูลจาก Source wd. ไปยัง Destination wd.
26	MOVE NOT-MVN(22)	MVN(22) Source wd Destination wd	S: Source wd IR,SR,AR,HR,LR,TC,DM,# D: Destination wd IR,HR,LR,DM	คำสั่งเวิลด์ทำการสำเนาข้อมูลแบบตรงข้ามของ Source wd. ไปยัง Destination wd.
27	COMPARE CMP(20)	CMP(20) CP1 CP2	CP1: Compare wd 1 IR,SR,AR,HR,LR,TC,DM,# CP1: Compare wd 2 IR,SR,AR,HR,LR,TC,DM,#	คำสั่งเวิลด์เปรียบเทียบข้อมูลระหว่าง CP1 และ CP2 แล้วมีผลต่อแฟลก >, < ,และ =

4.3.1.6 บางส่วนกลุ่มคำสั่งแปลงข้อมูลและคำนวณทางคณิตศาสตร์

ตารางที่ 4.14 บางส่วนกลุ่มคำสั่งแปลงข้อมูลและคำนวณทางคณิตศาสตร์

ลำดับ	คำสั่ง IL	สัญญาลักษณ์ LD	พื้นที่หน่วยความจำ	ความหมาย
28	BCD to BIN BIN(23)	BIN(23) Source wd Destination wd	S: Source wd IR,SR,AR,HR,LR,TC,DM,#	คำสั่งเวิลด์ทำการแปลงข้อมูลจาก Source wd.(BCD)ไปเก็บที่ Destination wd.(Hex)
			D: Destination wd IR,HR,LR,DM	
29	BIN to BCD BCD(24)	BCD(24) Source wd Destination wd	S: Source wd IR,SR,AR,HR,LR,TC,DM,#	คำสั่งเวิลด์ทำการแปลงข้อมูลจาก Source wd.(Hex)ไปเก็บที่ Destination wd.(BCD)
			D: Destination wd IR,HR,LR,DM	
30	BCD ADD ADD(30)	ADD(30) Au Ad R	Au: Augend wd(BCD) IR,SR,AR,HR,LR,TC,DM,#	คำสั่งบวกเลขฐาน 10 พร้อมตัวทด Au + Ad + Cy = CY & R
			Ad: Addend wd(BCD) IR,SR,AR,HR,LR,TC,DM,#	
			R: Result wd(BCD) IR,HR,LR,DM	
31	BCD SUB SUB(31)	SUB(31) Mi Su R	Mi: Minuend wd(BCD) IR,SR,AR,HR,LR,TC,DM,#	คำสั่งลบเลขฐาน 10 พร้อมตัวยืม Mi - Su - Cy = CY & R
			Su: Subtrahend wd(BCD) IR,SR,AR,HR,LR,TC,DM,#	
			R: Result wd(BCD) IR,HR,LR,DM	
32	BCD MUL MUL(32)	MUL(32) Md Mr R	Md: Multiplicaton wd(BCD) IR,SR,AR,HR,LR,TC,DM,#	คำสั่งคูณเลขฐาน 10 ผลเก็บที่ R & R +1 Md * Mr = R+1 & R
			Mr: Multiplier wd(BCD) IR,SR,AR,HR,LR,TC,DM,#	
			R: First Result wd(BCD) IR,HR,LR,DM	
33	BCD DIV DIV(33)	DIV(33) Dd Dr R	Dd: Dividend wd(BCD) IR,SR,AR,HR,LR,TC,DM,#	คำสั่งบวกเลขฐาน 10 ผลเก็บที่ R & R+1 Dd / Dr = R+1 & R
			Dr: Divisor wd(BCD) IR,SR,AR,HR,LR,TC,DM,#	
			R: First Result wd(BCD) IR,HR,LR,DM	

4.3.1.7 บางส่วนกลุ่มคำสั่งสื่อสารข้อมูล

ตารางที่ 4.15 บางส่วนกลุ่มคำสั่งสื่อสารข้อมูล

ลำดับ	คำสั่ง IL	สัญญาลักษณ์ LD	พื้นที่หน่วยความจำ	ความหมาย
34	RECEIVE RXD(47)	RXD(47) D C N	D: Destination wd IR,SR,AR,HR,LR,TC,DM	คำสั่งเวิลด์อ่านข้อมูลจาก Ethernet Buffer แล้วเขียนที่ Destination wd.จำนวน N byte
			C: Control wd #	
			N : Number of byte IR,SR,AR,HR,LR,TC,DM,#	
35	TRANSMIT TXD(48)	TXD(48) S C N	S: Source wd IR,SR,AR,HR,LR,TC,DM	คำสั่งเวิลด์อ่านข้อมูลจาก Source wd จำนวน N byte แล้วส่งไปที่ Ethernet Buffer
			C: Control wd #	
			N : Number of byte IR,SR,AR,HR,LR,TC,DM,#	

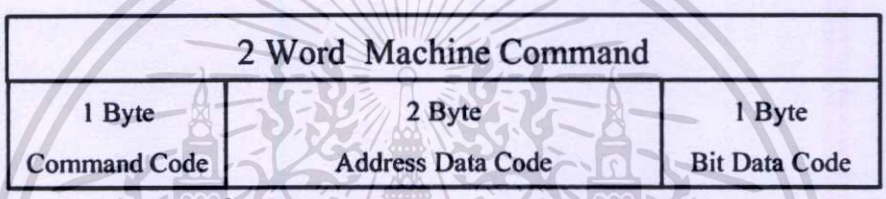
4.4 โปรแกรมสร้างรหัสเครื่อง PLC (Generator machine code)

โปรแกรมสร้างรหัสเครื่อง PLC (Generator machine code) เป็นส่วนสำคัญทำหน้าที่แปลงคำสั่งโปรแกรม PLC ในรูปคำสั่ง Instruction List : IL ตามมาตรฐาน IEC 1131-3ให้เป็นข้อมูลรหัสเครื่อง(Machine Code) เริ่มตั้งแต่การเขียนโปรแกรม IL ด้วยคอมพิวเตอร์อาศัยอิดีเตอร์ Notepad บน ไมโครซอร์ฟ Windows แล้วจัดเก็บข้อมูลในรูปไฟล์ตัวอักษรที่หน่วยความจำ SD Card

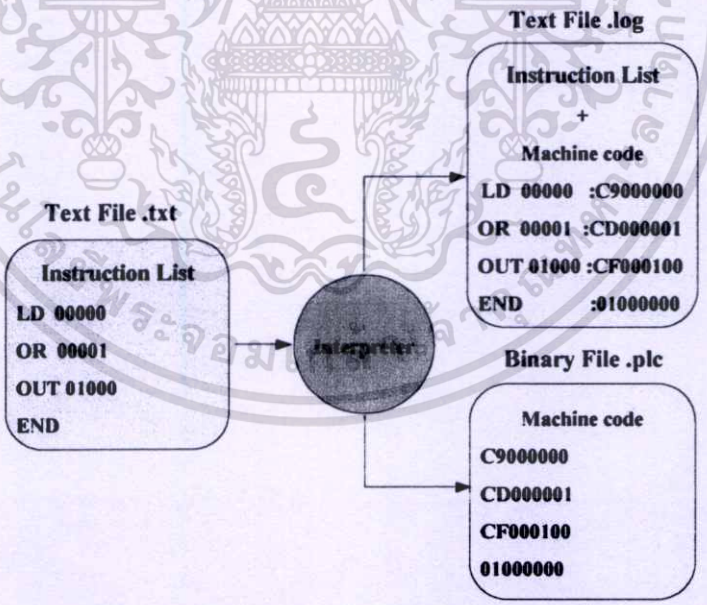
โปรแกรมจะทำการอ่านไฟล์ข้อมูลทีละบรรทัดแล้วเปรียบเทียบคำสั่งต่อคำสั่งเพื่อทำการแปลงข้อมูลคำสั่ง และ ข้อมูลโอเปอร์แลนทั้งหมดให้เป็นรหัส ตามรูปแบบรูปที่ 4.13 และลำดับขั้นการแปลงตามรูปที่ 4.15 โดยแยกจัดเก็บเป็น 2 ไฟล์ ด้วยกัน ตามรูปที่ 4.14 ได้แก่ไฟล์ข้อมูลแบบไบนารี หรือ ข้อมูลรหัสเครื่อง ที่ PLC สามารถเข้าใจเพื่อนำไปประมวลผลได้ทันที กับไฟล์ข้อมูลแบบตัวอักษร TextFile.log เพื่อใช้รายงานความผิดพลาดของโปรแกรมที่ผ่านขั้นตอนการทำงานของโปรแกรมสร้างรหัสเครื่องมาแล้ว

4.4.1 ส่วนประกอบคำสั่งรหัสเครื่อง

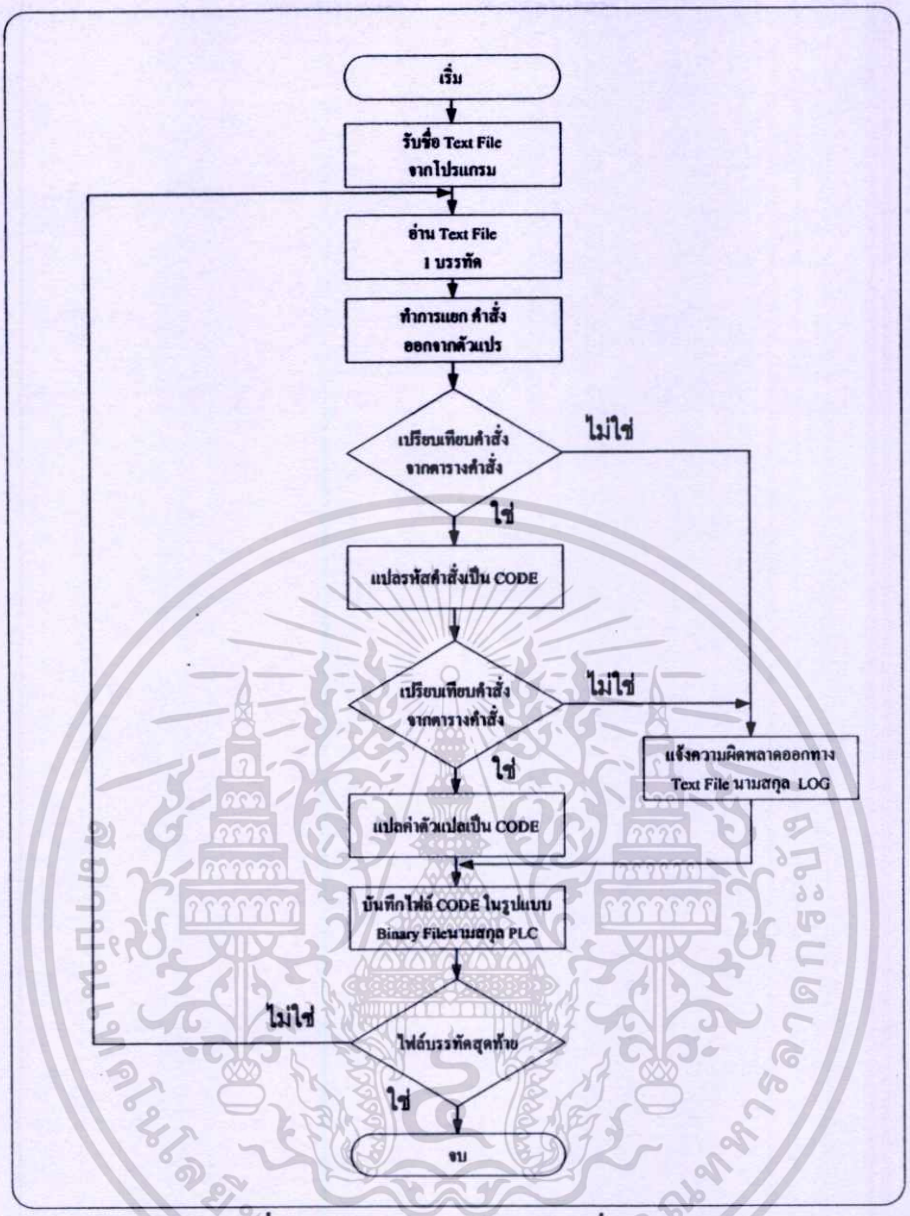
คำสั่งILหนึ่งๆที่อ่านมา จะถูกสร้างข้อมูลขึ้นมาใหม่มีขนาด 2 เวิลด์หรือขนาด 4 ไบท์ ประกอบด้วย รหัสคำสั่ง 1 ไบท์ แอดเดรสข้อมูล 2 ไบท์ และบิทข้อมูล 1 ไบท์ เพื่อให้กำหนดรายละเอียด ในคำสั่งนั้น ดังรูปที่ 4.10



รูปที่ 4.13 แสดงส่วนประกอบรหัสคำสั่งเครื่อง



รูปที่ 4.14 แสดงการจัดเก็บไฟล์ข้อมูลรหัสเครื่อง



รูปที่ 4.15 โปรแกรมสร้างรหัสเครื่อง PLC

บางส่วนคำสั่ง IL รวมถึงการสร้างรหัสคำสั่ง เวลาและจำนวนคล็อกในการปฏิบัติคำสั่ง มาแสดงให้ตามตารางที่ 4.16

ตารางที่ 4.16 บางส่วน คำสั่ง IL การสร้างรหัสคำสั่ง เวลาและจำนวนคล็อกในการปฏิบัติคำสั่ง

ลำดับ	Instruction List (IL)	รูปแบบโปรแกรม C	รหัสเครื่อง	เวลาการปฏิบัติคำสั่ง (μsec)	จำนวน (Clock) ของ LPC2378

1	LD [bit]	case 201 : // LD
		shift_buf();
		B0= inbit(CODE_2);
		P0= inbit_P(CODE_2);
			C900XXXYH	4	288

2	LD NOT [bit]	case 202 : //LD-NOT
		shift_buf();
		B0 = ~inbit(CODE_2);
		P0 = ~inbit_P(CODE_2);
			CA00XXXYH	4	288

ตารางที่ 4.16(ต่อ) บางส่วน คำสั่ง IL การสร้างรหัสคำสั่ง เวลาและจำนวนคล็อกในการปฏิบัติคำสั่ง

ลำดับ	Instruction List (IL)	รูปแบบโปรแกรม C	รหัสเครื่อง	เวลาการปฏิบัติคำสั่ง (μsec)	จำนวน (Clock) ของ LPC2378

3	AND [bit]	case 203 : //AND
		BF = inbit(CODE_2);
		PF= inbit_P(CODE_2);
		B0 = B0 & BF;
		P0 = P0 & PF;
			CB00XXXYH	2	144

4	OUT [bit]	case 207 : //OUT
		outbit(CODE_2,B0);
		outbit_P(CODE_2,P0);
			CF00XXXYH	3	216

5	DIFU [bit]	case 13 : //DIFU
		if(B0 & (~P0))
		outbit(CODE_2,1);
		else
		outbit(CODE_2,0);
			0C00XXXYH	3	216

6	DIFD [bit]	case 14 : //DIFD
		if((~B0) & P0)
		outbit(CODE_2,1);
		else
		outbit(CODE_2,0);
			0D00XXXYH	3	216

7	CMP [CP1,CP2]	case 20 : //CMP
		CODE_2= get_code_data(CODE_2);
		CODE_3= get_code_data(CODE_3);
		if(CODE_2>CODE_3)
		outbit(SR_GR,1);
		else
		outbit(SR_GR,0);
		if(CODE_2==CODE_3)
		outbit(SR_EQ,1);
		else
		outbit(SR_EQ,0);
		if(CODE_2<CODE_3)
		outbit(SR_LE,1);
		else
		outbit(SR_LE,0);
			140RXXXXH
			000RXXXXH
				2	144

8	MOV [S , D]	case 21 : //MOV
		if(CODE_2 > 0xFFFF)
		IR[CODE_3] =
		CODE_2&0xFFFF;
		else
		IR[CODE_3] =IR[CODE_2];
			150RXXXXH
			000XXXXXH
				1	72

10	ADD [Au, Ad, R]	case 30 : //ADD<<
		CODE_2=get_code_data(CODE_2);
		CODE_3=get_code_data(CODE_3);
		IR[CODE_4]
		=CODE_2+CODE_3+inbit(SR_CY);
		if(IR[CODE_4]>9999)
		outbit(SR_CY,1);
		IR[CODE_4] =
		bcd_bin(IR[CODE_4]);
			1E0RXXXXH
			000RXXXXH
			0000XXXXH
				1	72

12	TIM [N,SV]	case 211 : //TIM
		CODE_3=get_code_data(CODE_3);
		if(B0)
		IR[TIM+CODE_2] |= TIMIF_RUN;
		else
		IR[TIM+CODE_2] =CODE_3;
			D3RXXXXH
			0000XXXXH
				2	144

ตารางที่ 4.16 (ต่อ) บางส่วน คำสั่ง IL การสร้างรหัสคำสั่ง เวลาและจำนวนคล็อกในการปฏิบัติคำสั่ง

ลำดับ	Instruction List (IL)	รูปแบบโปรแกรม C	รหัสเครื่อง	เวลาการปฏิบัติคำสั่ง (μsec)	จำนวน (Clock) ของ LPC2378

13	CNT [N,SV]	case 212 : //CNT
		CODE_3=get_code_data(CODE_3);
		if(B1 & (~P1))
		if(IR[CNT+CODE_2] < CODE_3)
		IR[CNT+CODE_2]++;
		if(IR[CNT+CODE_2] == CODE_3)
		outbit((CNT_IF<<4)+CODE_2,1);
		else
		outbit((CNT_IF<<4)+CODE_2,1);
		if(B0)
		IR[CNT+CODE_2]=0;
		outbit((CNT_IF<<4)+CODE_2,0);
			D4RXXXXH
			0000XXXXH
				2	144

14	CMP [F , S]	case 20 : //CMP
		{ if(at_check()){
		CODE_2= get_code_data(CODE_2);
		CODE_3= get_code_data(CODE_3);
		if(CODE_2>CODE_3)
		outbit(SR_GR,1);
		else
		outbit(SR_GR,0);
		if(CODE_2==CODE_3)
		outbit(SR_EQ,1);
		else
		outbit(SR_EQ,0);
		if(CODE_2<CODE_3)
		outbit(SR_LE,1);
		else
		outbit(SR_LE,0);
		}
		break;
		}			

15	END	case 1 : //END
		PC = 0;
			01000000H	1	72

ตารางที่ 4.16 ตารางคำสั่ง LI การสร้างรหัสคำสั่งโดยจะให้วิธีการแบ่งคำสั่งออกเป็นแต่ละชุดคำสั่ง โดยในหนึ่ง step คำสั่งในในคำสั่ง LI จะสามารถแปลงเป็นคำสั่งการทำงานในภาษา C ได้หลายคำสั่ง โดยจะแบ่งเป็นชุด ดังตารางที่ 4.16 ในการทำงานนั้นสำสั่งจะถูกเลือกขึ้นมาทำงานตามลำดับตามโปรแกรมประยุกต์ใช้งาน โดยจะอ้างอิงการทำงานจาก รหัสคำสั่งดังรูปที่ 4.13

4.4.2 การอ้างอิงตำแหน่งหน่วยความจำ IR

การเรียกใช้งานพื้นที่หน่วยความจำแต่ละส่วน จะอ้างอิงจากหมายเลขด้วยตัวเลข 5 หลัก หลักหน่วย และหลักสิบ ตัวเลขสองทางขวาแสดงถึงบิต ตัวเลขที่เป็นไปได้คือ 00 ถึง 16 ส่วนหลักร้อยหลักพัน และ หลักหมื่นแสดงถึงเวิลด์ใน หน่วยความจำ IR สรุปว่า ถ้าต้องการอ้างอิงหน่วยความจำ IR แบบเวิลด์ (Wd.) ให้ใช้ตัวเลข 3 หลัก และถ้าต้องการอ้างอิงหน่วยความจำIR บิทให้ใช้ตัวเลข 5 หลัก

รูปที่ 4.16 การอ้างอิงตำแหน่งหน่วยความจำแบบบิทในพื้นที่ IR

เช่นอ้างอิงตำแหน่งหน่วยความจำแบบบิทตำแหน่งที่ 00215 หมายถึงต้องการเรียกใช้งานพื้นที่หน่วยความจำภายใน ใน Wd. ที่ 002 บิตที่ 15 เป็นต้นสำหรับพื้นที่ความจำหน่วยอื่นๆนอกเหนือจาก IR และ SR แล้วในการกำหนดแอสเดรสจะต้องใส่ชื่อ ตัวย่อของพื้นที่ส่วนนั้นก่อนเช่น หน่วยความจำ DM Wd. ที่ 0 ต้องกำหนดเป็น DM000 เป็นต้น

ตารางที่ 4.17 ตัวอย่างการอ้างอิงหน่วยความจำPLC ทั้งแบบเวิลด์ และ บิต

หน่วยความจำ	ตัวย่อ	ช่วงที่สามารถอ้างอิง	
		Word	Bit
หน่วยความจำอินพุท เอาท์พุท ภายใน	IR	000-231	00-15
หน่วยความจำชนิดพิเศษ	SR	232-255	
หน่วยความจำคงค่าสภาวะ	HR	00-19	00-15
หน่วยความจำ ออคซีเลียรี่	AR	17-21	00-15
หน่วยความจำลิงค์รีเลย์	LR	00-15	00-15
หน่วยความจำตัวตั้งเวลา	TIM	000-256	ไม่อ้างอิง
หน่วยความจำตัวนับ	CNT	000-256	ไม่อ้างอิง
หน่วยความจำดาต้าเมมโมรี่	DM	000-512	ไม่อ้างอิง

4.4.3 ข้อกำหนดในการเขียนโปรแกรมคำสั่ง IL

1) คำสั่งต้องเขียนด้วยตัวพิมพ์ใหญ่ทั้งหมด

2) เมื่อเขียนคำสั่ง แล้วตามด้วยเว้นวรรค เช่น LD 01001 ความหมาย โหลดค่า ในWd. 010 บิตที่01 การอ้างอิงตำแหน่งหน่วยความจำภายใน IR ไม่จำเป็นต้องเขียน IR ให้ใส่ตัวเลขหน่วยความจำได้เลย สำหรับพื้นที่อื่นๆ ให้ตามด้วยตัวย่อของพื้นที่ก่อนจากนั้นตามด้วยตัวเลขเลข 2 แสดงถึงบิตที่ต้องการใช้งาน

เช่น LD 01001 (01001 หมายถึง รีเลย์ IR ใน Word ที่ 010 ในตำแหน่งบิตที่ 01)

LD HR0108 (HR0108 หมายถึง รีเลย์ HR ใน Word ที่ 01 ในตำแหน่งบิตที่ 08)

3) การใช้คำสั่งระดับเวิลด์โอเปอร์แลนที่กะทำ ต้องอ้างอิงแบบ Wd. ด้วยเช่นกัน

เช่น ADD(40) IR000

TIM0

DM000 (นำค่าใน Wd. IR000 ไป บวกกับข้อมูลปัจจุบัน (PV.)ใน TIM00 ผลลัพธ์เก็บที่ DM000)

4) ตัวตั้งเวลากำหนดค่าคงที่ ทำได้โดยการใส่เครื่องหมาย # นำหน้าค่าตัวเลขนั้น จะแสดงถึงค่าคงที่

เช่น TIM 0

#100 (เป็นค่าคงที่ 100 * ฐานเวลา0.1วินาที = 10 วินาที)

5) ในคำสั่งระดับเวิลด์ที่มีโอเปอร์แลนมากกว่า 1 Wd.หลังจากที่ใส่โอเปอร์แลนชุดที่หนึ่งไปแล้ว ให้ขึ้นบรรทัดใหม่แล้วตามด้วยโอเปอร์แลนชุดที่สอง และ ขึ้นบรรทัดใหม่ตามด้วยโอเปอร์แลนชุดที่สามตามลำดับ

เช่น XORW DM000

#1234

DM001 (นำค่าใน DM000 ไป XOR กับ 1234 แล้วนำไปเก็บที่ DM001)

6) ในหมวดทำสั่งที่ต้องการให้ทำงานเพียงแค่ 1 scan time (อ้าอิงตาม OMRON) ให้ใส่เครื่องหมาย @ ก่อนหน้าคำสั่งนั้นๆ

เช่น LD 00001

@INC 010 (เมื่อมีสัญญาณโลจิก 1 เข้ามาให้เพิ่มค่าข้อมูลใน IR010 ขึ้นมาจากเดิม 1)

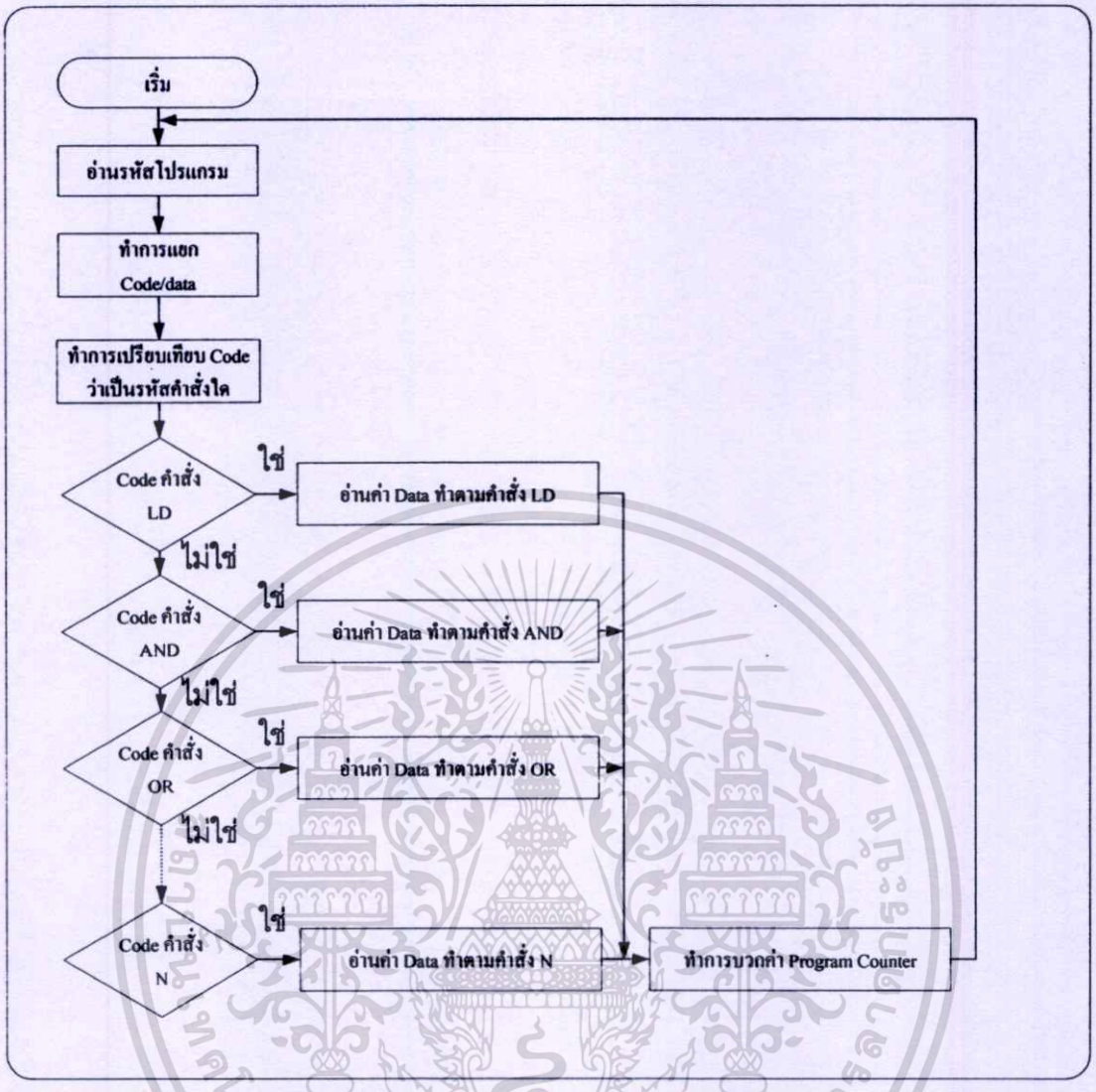
LD 00002

INC 011 (เมื่อมีสัญญาณโลจิก 1 เข้ามา ให้เพิ่มค่า ใน IR011 ทุกๆ scan time)

7) เมื่อเขียนคำสั่งเสร็จให้ตามด้วยคำสั่ง END เสมอ

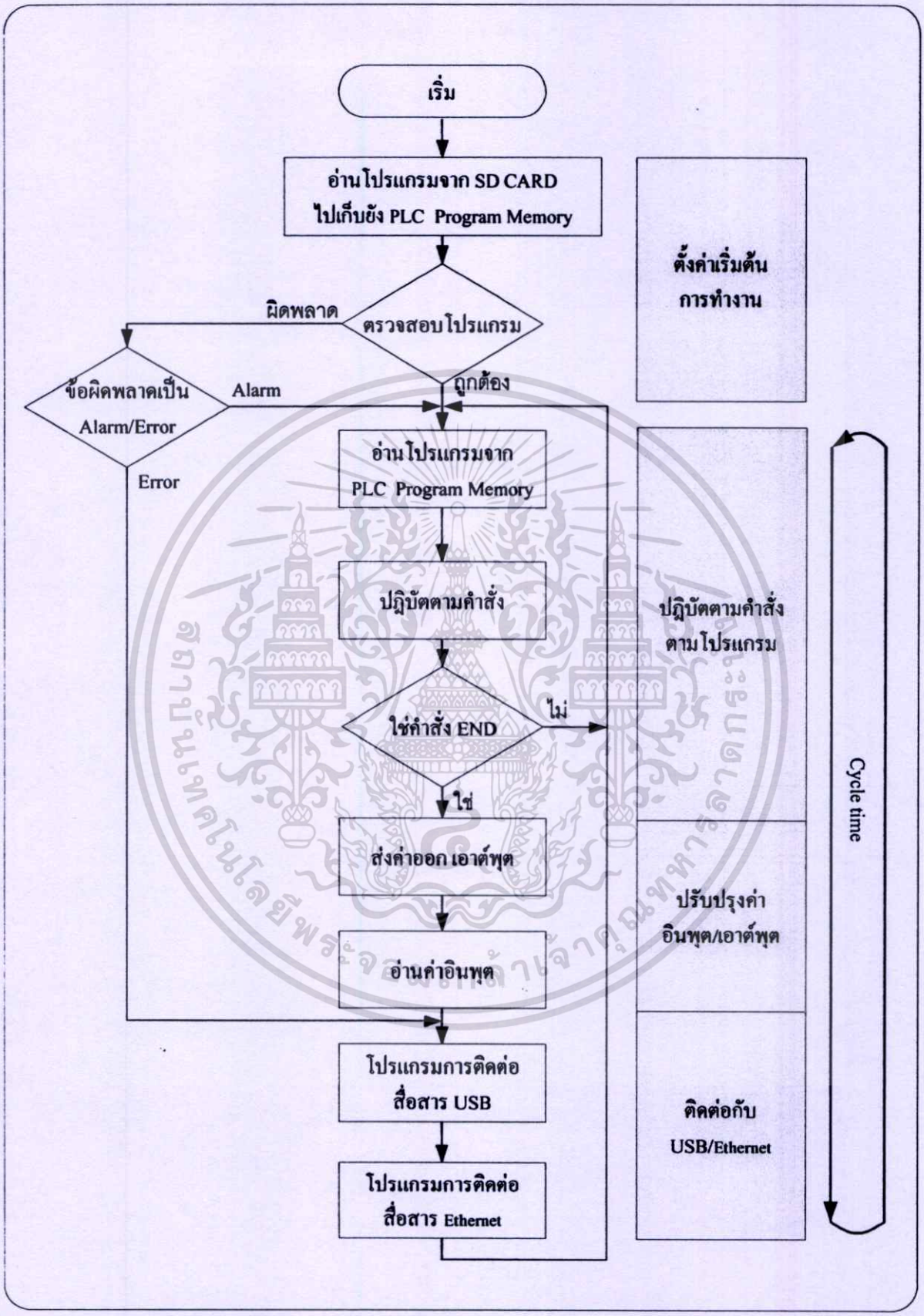
4.4.4 การแปลความหมายคำสั่ง และ ระบบปฏิบัติการของ PLC

การแปลความหมายคำสั่ง PLC ทำได้ 2 วิธี วิธีแรกแปลความหมายโปรแกรมทั้งหมดให้เป็น เลขฐาน 16 แล้วคอยนำไปประมวลผลพร้อมกันที่เดียว ส่วนวิธีที่สองการแปลความหมายที่ละบรรทัดขณะเดียวกันก็ประมวลผลการทำงานตามโปรแกรมไปด้วยคำสั่งต่อคำสั่งจนจบโปรแกรม สำหรับงานวิจัยนี้ใช้การแปลความหมายแบบวิธีแรกเพราะสามารถทำงานได้รวดเร็วกว่าวิธีที่สองการตรวจสอบและติดตามการทำงานโปรแกรมก็ทำได้สะดวกกว่า ขั้นตอนการทำงานของการแปลความหมาย ดังรูปที่ 4.17



รูปที่ 4.17 รูปแผนภูมิการแปลความหมายคำสั่งPLC

ระบบปฏิบัติการทำงานของ PLC เริ่มด้วยโปรแกรมจะทำการอ่านค่าจากหน่วยความจำ SD/MMC memory card เข้ามาเก็บยังหน่วยความจำ PLC ที่เป็น Program buffer โปรแกรมจะทำการตรวจสอบ ความถูกต้องของไฟล์ข้อมูลอีกครั้ง จากนั้นจะทำการอ่านคำสั่งแรกเข้ามา 1 ครั้ง จำนวน 2 เวิลด์แบบเลขฐาน 16 ทำการแปลความหมายและประมวลผลตามคำสั่งจัดเก็บข้อมูลลงหน่วยความจำ แล้ววนกลับเพื่ออ่านคำสั่งลำดับถัดมา เข้าสู่กระบวนการเหมือนคำสั่งแรกที่ได้กล่าวไปจนกว่าจะถึงคำสั่ง END โปรแกรมจะทำการส่งข้อมูลที่ประมวลผลทั้งหมดไปยังหน่วยความจำเอาท์พุต และ เอาท์พุตที่ขับอุปกรณ์ภายนอก จากนั้นอ่านข้อมูลจากหน่วยความจำอินพุตที่รับสถานะมาจากอุปกรณ์ภายนอกเข้ามาอีกครั้ง รวมถึงการรับส่งข้อมูลผ่านพอร์ท USB และ Ethernet ด้วย



รูปที่ 4.18 รูปแผนภูมิการทำงานของระบบปฏิบัติการ PLC

4.5 ชุดคำสั่งในการสื่อสารแบบอนุกรมผ่านพอร์ท USB

ในการสื่อสารข้อมูลระหว่าง PLC กับ เครื่องคอมพิวเตอร์ PCที่ทำหน้าที่เป็นโฮสต์ ผ่านทางพอร์ท USB นั้นเป็นเทคนิคที่ผู้ใช้สามารถ อ่านและเขียนข้อมูลกับหน่วยความจำเสมือนของ PLC ได้โดยตรง ทั้งนี้เพื่อความสะดวกรวดเร็ว และ มีความเร็วในการสื่อสารข้อมูลเป็นแบบความเร็วสูงเพื่อประโยชน์ในการทำระบบเฝ้ามอง และติดตามการทำงาน รวมถึงการอาศัยคอมพิวเตอร์ควบคุม PLC แบบรีโมทได้เช่นกัน

ในการทดสอบเขียนโปรแกรมติดต่อสื่อสารระหว่างโฮสต์คอมพิวเตอร์ PC กับโมดูล USB จะอาศัยไฟล์ตัวอย่าง และ ฟังก์ชั่นภาษา C ด้วยโปรแกรมพัฒนาของ Keil ในหัวข้อ USBHID – Human Interface ทำให้การเขียนโปรแกรมติดต่อสื่อสารกับ USB ทำได้สะดวกยิ่งขึ้น

สิ่งที่เป็นความรู้พื้นฐานเกี่ยวกับ HID คลาส

1. การเปลี่ยนแปลงของข้อมูลที่เกิดขึ้นในโครงสร้างการทำงานจะเรียกว่า รายงานหรือรีพอร์ต (report) อุปกรณ์ควบคุมหลักของอุปกรณ์ USB จะต้องสามารถรองรับรูปแบบของข้อมูลที่เป็นไปตามข้อกำหนดของ HID รีพอร์ต โดยโฮสต์จะรับส่งข้อมูลด้วยการส่งและร้องขอรีพอร์ตในการควบคุมหรือมีลักษณะเป็นการถ่ายทอดสัญญาณแบบอินเตอร์รัปต์ อย่างไรก็ตามรูปแบบของรีพอร์ตนี้ยังไม่มีข้อสรุปชัดเจนแต่ก็มีความอ่อนตัวสูง ทำให้สามารถรองรับกับข้อมูลได้ทุกรูปแบบ

2. อุปกรณ์ USB สามารถส่งข้อมูลกลับไปยังโฮสต์ (ซึ่งส่วนใหญ่คือ คอมพิวเตอร์) ได้ตลอดเวลาโดยไม่สามารถคาดหมายล่วงหน้า ดังนั้นจึงเป็นหน้าที่ของโฮสต์เองที่ต้องคอยวนตรวจสอบอยู่ตลอดเวลาเพื่อให้สามารถรองรับกับการทำงานของอุปกรณ์ได้ทันท่วงที

3. ในการถ่ายทอดสัญญาณหรือข้อมูลบนบัสของ USB ไม่สามารถทราบถึงอัตราการส่งผ่านข้อมูลได้อย่างแน่นอน ระบบจะพยายามทำให้เกิดอัตราการส่งผ่านข้อมูลเร็วที่สุดเท่าที่จะเป็นไปได้

4. ในการถ่ายทอดสัญญาณหรือข้อมูลสูงสุดจะถูกจำกัดไว้ตามประเภทของอุปกรณ์ ซึ่งมีด้วยกัน 3 แบบ คือความเร็วต่ำ,ความเร็วเต็มที่และความเร็วสูง โดยโฮสต์สามารถกำหนดคาบเวลาในการทำงานสูงสุดแยกกันไปตามประเภทของอุปกรณ์ โดย

4.1 สำหรับอุปกรณ์ความเร็วต่ำ โฮสต์จะกำหนดคาบเวลาในการทำงาน 1 งานสูงสุดไม่เกิน 10 มิลลิวินาที ทำให้อัตราการถ่ายทอดสัญญาณเกิดขึ้นได้สูงสุดไม่เกิน 800 ไบต์ต่อวินาที

4.2 สำหรับอุปกรณ์ความเร็วเต็มที่ โฮสต์กำหนดคาบเวลาในการทำงาน 1 งานสูงสุดไม่เกิน 1 มิลลิวินาที ทำให้อัตราการถ่ายทอดสัญญาณเกิดขึ้นได้สูงสุดไม่เกิน 8,000 ไบต์ต่อวินาที

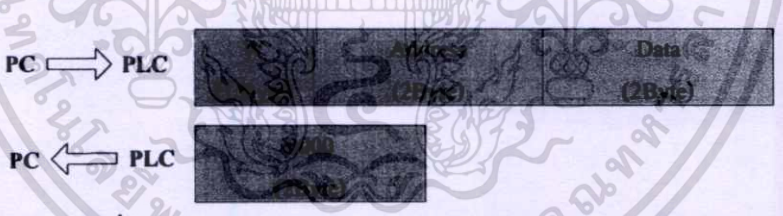
4.3 สำหรับอุปกรณ์ความเร็วสูง โฮสต์กำหนดคาบเวลาในการทำงาน 3 งานสูงสุดไม่เกิน 125 ไมโครวินาที อัตราการถ่ายทอดสัญญาณเกิดขึ้นได้สูงสุดไม่เกิน 24.576 MB ต่อวินาที

5. ในระบบปฏิบัติการวินโดวส์ที่ต่ำกว่าวินโดวส์ 98 SE จะไม่รองรับการถ่ายทอดสัญญาณแบบอินเตอร์รัปต์ ดังนั้นการติดต่อจากโฮสต์ไปยังอุปกรณ์จะต้องใช้การถ่ายทอดสัญญาณควบคุมมาช่วยแทน

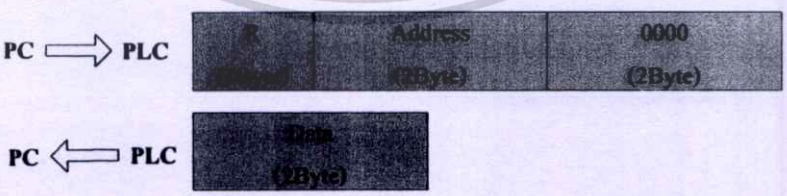
ในส่วนของไมโครคอนโทรลเลอร์ LPC2378 ได้กำหนดคุณสมบัติส่วนของฮาร์ดแวร์ที่เกี่ยวข้องกับโมดูล USB ดังนี้

- กำหนดการเชื่อมต่อ USB แบบ USB Link Mode และใช้พลังงานจากบัส (Bus Power) 400 mA
- กำหนดขนาดแพ็คเก็ตของเอ็นด์พอยต์เป็น 64 ไบต์
- กำหนดคลาสของอุปกรณ์เป็นแบบ HID โดยใช้ Non Protocol มี 1 อินเตอร์เฟส (Interface Number 0)
- ใช้การส่งถ่ายข้อมูลแบบอินเตอร์รัปต์และกำหนดค่าเวลาสแกนเป็น 1 มิลลิวินาที
- กำหนดค่า Manufacture Name เป็น "PLC32"
- กำหนดค่า Product Name เป็น "PLC32"
- กำหนดค่า Product Serial Number เป็น "PLCUSB V10"
- กำหนดใช้งาน End Point-0 สำหรับตั้งค่าและรับข้อมูล ส่วน End Point-1 ใช้สำหรับส่งข้อมูล
- Output Report กำหนดให้ มีขนาด 5 ไบต์/8 บิต (ไม่รวม Report ID)Byte-0 เป็นบอกว่าต้องการอ่านหรือเขียนหน่วยความจำของ Byte 1-2 บอกถึงแอสเดรสของหน่วยความจำบน PLC Byte 3-4 ค่าที่ต้องการเขียนลงในแอสเดรสที่ต้องการ

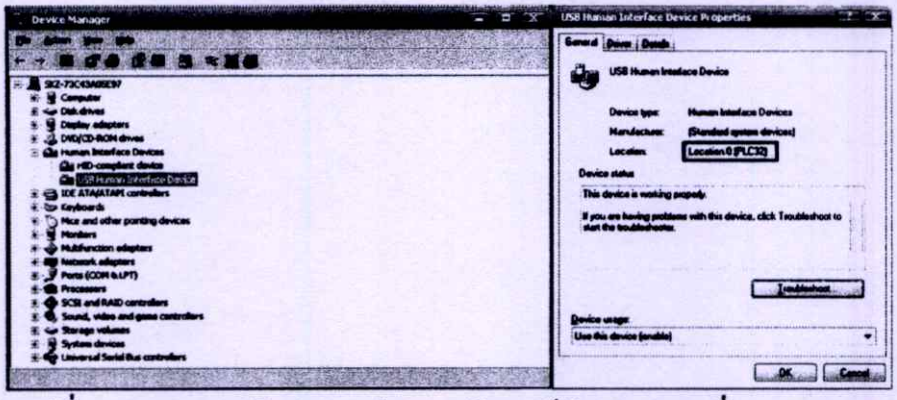
การรับส่งข้อมูลและคำสั่งจะใช้ HID Report โดยมีการกำหนดรูปแบบของการรายงานผลไว้ดังนี้ Input Report กำหนดให้ มีขนาด 2 ไบต์/8 บิต (ไม่รวม Report ID) เป็นค่าของหน่วยความจำภายในที่ต้องการอ่าน



รูปที่ 4.19 แสดงคำสั่งการเขียนข้อมูลลงในหน่วยความจำของ PLC ผ่านทาง USB Port



รูปที่ 4.20 แสดงคำสั่งการอ่านข้อมูลจากหน่วยความจำของ PLC ผ่านทาง USB Port

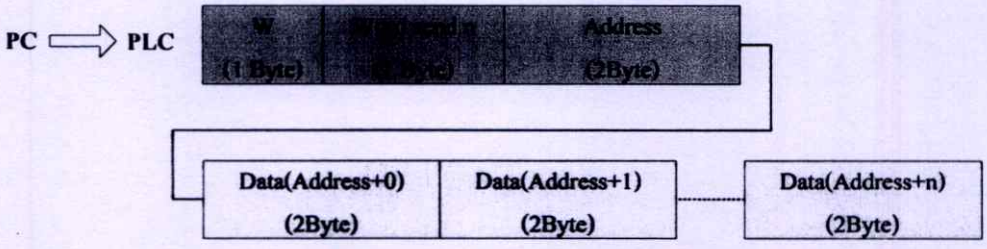


รูปที่ 4.21 แสดงการติดต่อระหว่างคอมพิวเตอร์กับ PLC การเชื่อม USBHID

4.6 ชุดคำสั่งในการสื่อสารผ่านโครงข่ายแบบ Internet

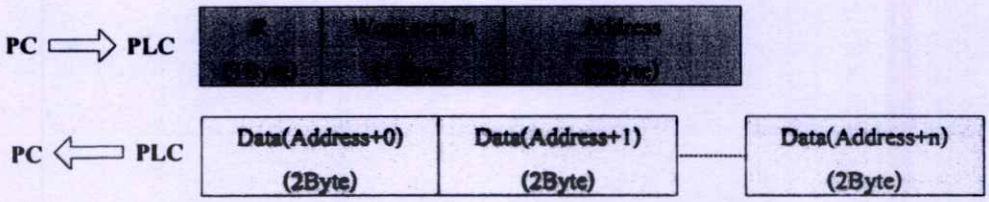
ระบบเครือข่าย Internet ที่ใช้โปรโตคอลมาตรฐานชื่อ TCP/IP ในการสื่อสารผ่านระบบเพื่อติดต่อกับ PLC โปรโตคอล TCP/IP ข้อดีของโปโตคอล TCP/IP ก็คือในการกำหนดเส้นทางสำหรับการรับส่ง ที่สามารถเลือกเส้นทางในการรับส่งข้อมูลได้อย่างอัตโนมัติหากถ้าเกิดเส้นทาง บางเส้นทางเสียหาย ระบบกลไกในการกำหนดเส้นทางสำหรับการรับส่งข้อมูลของโปรโตคอล TCP/IP ก็จะเลือกเส้นทางให้เหมาะสมถูกต้องให้สามารถรับส่งข้อมูลได้นั้น

ในการเขียนโปรแกรมเพื่อใช้ในระบบเครือข่าย(Network)จุดหลักๆของระบบ จะแบ่งฝ่ายที่ต้องติดต่อรับส่งข้อมูลระหว่างกันออกเป็น 2 ส่วน คือ แม่ข่าย(Server)และลูกข่าย(Client) ซึ่งในการใช้งานจริงอาจมีส่วนประกอบอื่นๆอีก แต่เราจะขอไม่กล่าวถึง Server จะเป็นส่วนการทำงานของ PLC ทำหน้าที่ Server โดย Client ในการอ่านเขียนข้อมูลในหน่วยความจำของ PLC โดยที่ทั้ง Server และ Client ต่างก็จะต้องมีตำแหน่งที่อยู่(IP Address),ช่องทางการติดต่อ(Port) โดยทั้งสองฝ่าย จะสามารถติดต่อถึงกันได้จะต้องอยู่ในช่องทางเดียวกัน ซึ่งเราสามารถกำหนดหมายเลขของ Port ได้คือ Host Port ของ PLC คือ Port 5000



รูปที่ 4.22 แสดงคำสั่งการเขียนข้อมูลลงในหน่วยความจำของ PLC ผ่านทาง Ethernet

รูปที่ 4.19 การเขียนข้อมูลลงในหน่วยความจำ เริ่มต้นด้วยตัวอักษร W เป็นการบอกต้องการเขียนหน่วยความจำ ตามด้วย จำนวนหน่วยความจำที่ต้องการส่งขนาด 1 Byte 1 – 255 word ตามด้วย Address เริ่มต้นของบล็อกข้อมูลที่จะทำการส่งไป ขนาด 2 byte จากนั้นตามด้วยข้อมูล ตั้งแต่ Address เริ่มต้นจนถึง Address สุดท้าย

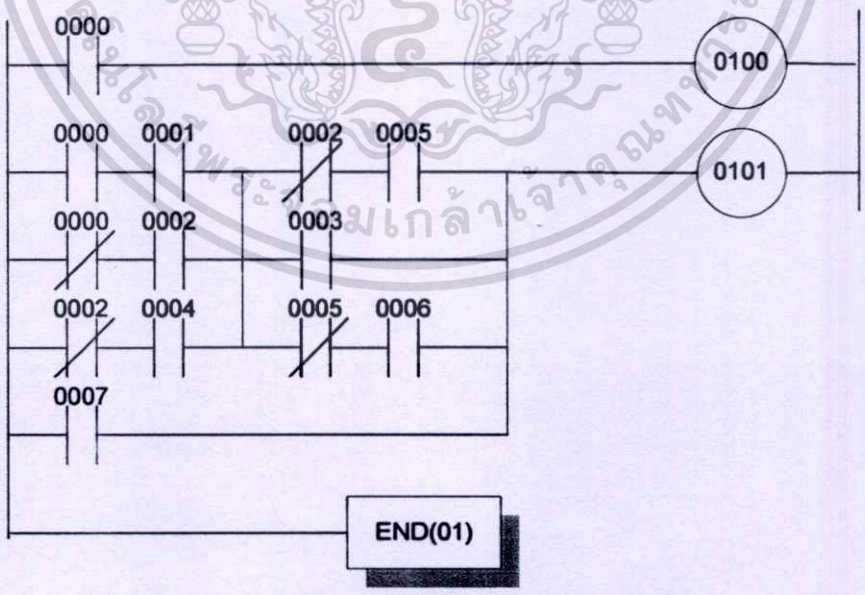


รูปที่ 4.23 แสดงคำสั่งการอ่านข้อมูลในหน่วยความจำของ PLC ผ่านทาง Ethernet

รูปที่ 4.23 การเขียนข้อมูลลงในหน่วยความจำ เริ่มต้นด้วยตัวอักษร R เป็นการบอกต้องการเขียนหน่วยความจำ ตามด้วย จำนวนหน่วยความจำที่ต้องการอ่านขนาด 1 Byte 1 – 255 word ตามด้วย Address เริ่มต้นของบล็อกข้อมูลที่จะทำการส่งไป ขนาด 2 byte เมื่อ PLC รับทราบถึงคำสั่งนั้น ก็จะตอบกลับมาด้วยข้อมูล ตั้งแต่ Address เริ่มต้นจนถึง Address สุดท้าย

4.7 โปรแกรมประยุกต์การใช้งาน

ตัวอย่าง โปรแกรมประยุกต์การใช้งานที่ 1 เพื่อทำการแปลคำสั่ง LD ให้เป็นคำสั่ง IL และ คำสั่งเครื่อง



รูปที่ 4.24 ตัวอย่างโปรแกรม LD ที่ประกอบคำสั่งพื้นฐาน

ตัวอย่าง นำเอา โปรแกรมคำสั่ง LD มาแปลงให้เป็นคำสั่ง IL

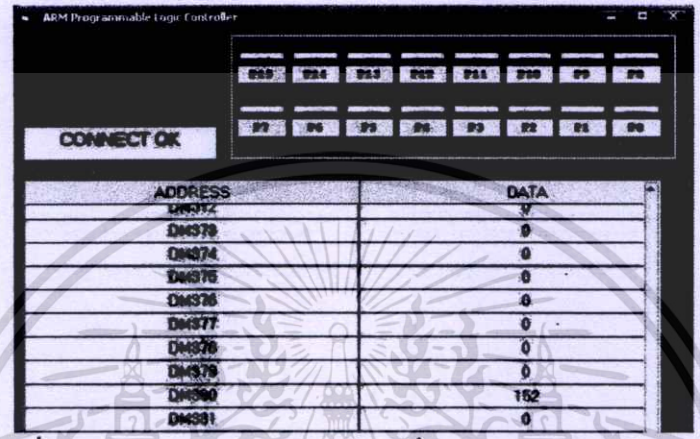
ตารางที่ 4.18 แสดงตัวอย่างโปรแกรมผู้ใช้งาน ที่ 1

ตำแหน่ง	คำสั่ง IL	รหัสคำสั่ง	เวลาการปฏิบัติคำสั่ง (µsec.)	จำนวน พัลส์ที่ใช้ ของMCU LPC2378 = 72 * เวลาการปฏิบัติคำสั่ง (µsec.)
0000	LD 0000	C9000000	4	288
0001	OUT 1000	CF000A00	3	216
0002	LD 0000	C9000000	4	288
0003	AND 0001	CB000001	2	144
0004	LD NOT 0000	CA000000	4	288
0005	AND 0002	CB000002	2	144
0006	OR LD	D1000000	4	288
0007	LD NOT 0002	CA000002	4	288
0008	AND 0004	CB000004	2	144
0009	OR LD	D1000000	4	288
0010	LD NOT 0002	CA000002	4	288
0011	AND 0005	CB000005	2	144
0012	OR 0003	CD000003	2	144
0013	LD NOT 0005	CA000005	4	288
0014	AND 0006	CB000006	2	144
0015	OR LD	D1000000	4	288
0016	AND LD	D2000000	4	288
0017	OR 0007	CD000007	2	144
0018	OUT 1001	CF0000A1	3	216
0019	END	01000000	65	4,680
	I/O refreshing		26	1,872
	Cycle Time		151 µsec.	11,088 clock pulse

ขั้นตอนในการเขียนโปรแกรม IL ให้กับ เครื่องควบคุม PLC

- เขียนโปรแกรม Ladder diagram (LD) ดังรูปที่ 4.24
- แปลงโปรแกรม จาก LD ให้เป็น Instruction List (IL)

- ย้าย SD card จากซ็อคเก็ตบนคอมพิวเตอร์ไปยังซ็อคเก็ตบน PLC จากนั้น กดรีเซ็ทสวิทช์บน PLC จะเป็นการ Down Load และสร้างโค๊ตภาษาเครื่องเก็บที่หน่วยความจำ SD card อีกครั้งในไฟล์ข้อมูล Filename.log
- PLC จะทำการรันโปรแกรมไฟล์ Filename.plc ที่เก็บในรูปแบบไฟล์ข้อมูลไบนารี
- ผลการแปลงโปรแกรมจัดเก็บในตารางที่ 4.18 และผลของสแกนไทม์ ดังรูปที่ 4.25



รูปที่ 4.25 ตรวจสอบเวลาแสกนไทม์ที่ DM 380 ผ่านพอร์ท USB

สาระสำคัญของบทนี้ได้อธิบายถึงขั้นตอนการออกแบบโปรแกรมการทำงานหรือระบบปฏิบัติการของเครื่องควบคุม PLC ที่ประกอบด้วยโปรแกรมการสร้างโค๊ตคำสั่ง โปรแกรมโฮสต์คอมพิวเตอร์เพื่อทำงานร่วมกับ PLC โปรแกรมสื่อสารข้อมูลผ่านพอร์ท Ethernet และ โปรแกรม IL ประยุกต์ของผู้ใช้งาน สำหรับในบทถัดไปจะได้กล่าวถึงการทดสอบการทำงานของ PLC ที่มีโปรแกรมทดสอบที่หลากหลายและทำการ Online และ Offline พอร์ท Ethernet เมื่อมีการเชื่อมต่อ PLC 2 ชุด เพื่อทดสอบสมรรถนะของ PLCเมื่อมีการรับ-ส่งข้อมูลผ่านทางพอร์ท และ เปรียบเทียบผลการทดสอบโดยการวัดสัญญาณด้วยเครื่อง Oscilloscope เปรียบเทียบกันทั้งกรณีที่ On-Offline

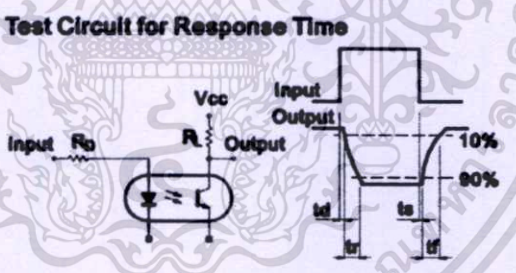
บทที่ 5

การหาค่าแสกนไทม์หรือหาสมรรถนะของ Embedded PLC

ในบทนี้จะกล่าวถึงการหาค่าการหาค่า แสกนไทม์หรือหาสมรรถนะของ Embedded PLC โดยใช้การทดลองโดยใช้เครื่องมือวัดทางไฟฟ้า และใช้ซอฟแวร์เป็นตัวช่วยในการวัดคาบเวลาในวัดแสกนไทม์ การทำงานของ PLC การทดลองจะถูกแบ่งออกเป็น 4 ส่วน ความเร็วของอินพุต/เอาต์พุตของ PLC, การหาค่าการวัดค่าแสกนไทม์ของ PLC, สมรรถนะการติดต่อสื่อสารผ่าน USB, สมรรถนะการติดต่อสื่อสารผ่าน Ethernet

5.1 ความเร็วของอินพุต/เอาต์พุตของ PLC

ปัจจัยที่มีผลสำคัญต่อการในการทำงานของในในการทำงานของภาคอินพุตจากการออกแบบในบทที่ 3 รูปที่ 3.4 ภาคอินพุตที่เราเลือกใช้เป็นแบบ OPTO ISOLATOR เบอร์ LTV-847 ใน ตระกูล LTV-817 Series ของบริษัท LITEON ในเพ็กเก็จ 1 ตัวมีขนาด 4 ช่องสัญญาณ ความเร็วในการทำงานของ OPTO ISOLATOR เบอร์ LTV-847 สามารถดูได้จาก ดาต้าชีส ของบริษัท LITEON ความเร็วในการทำงานของ เบอร์ LTV-847 หน้า 3 ตามรูปที่ 5.1 และหน้า 5 รูปที่ 5.2 วงจรที่อธิบายการทดสอบ



รูปที่ 5.1 วงจรการทดสอบ Response Time

Electrical/Optical Characteristics (Ta=25℃)

	Parameter	Symbol	Min.	Typ.	Max.	Unit	Conditions
Input	Forward Voltage	V_F	–	1.2	1.4	V	I_F=20mA
	Reverse Current	I_R	–	–	10	μA	V_R=4V
	Terminal Capacitance	C_t	–	30	250	pF	V=0, f=1KHz
Output	Collector Dark Current	I_{CEO}	–	–	100	nA	V_{CE}=20V
	Collector-Emitter Breakdown Voltage	BV_{CEO}	35	–	–	V	I_C=0.1mA
	Emitter-Collector Breakdown Voltage	BV_{ECO}	6	–	–	V	I_E=10 μ A
Transfer Characteristics	*Current Transfer Ratio	CTR	50	–	600	%	I_F=5mA, V_{CE}=5V R_{BE}=∞
	Collector Current	I_C	2.5	–	30	mA	
	Collector-emitter Saturation Voltage	$V_{CE(sat)}$	–	0.1	0.2	V	I_F=20mA, I_C=1mA
	Isolation Resistance	R_{ISO}	5×10^{10}	10^{11}	–	Ω	DC500V, 40~60% R.H.
	Floating Capacitance	C_f	–	0.6	1.0	pF	V=0, f=1MHz
	Cut-off Frequency	f_c	–	80	–	KHz	V_{CE}=5V, I_C=2mA R_L=100 Ω, -3dB
	Response Time (Rise)	t_r	–	4	18	μs	V_{CE}=2V, I_C=2mA
	Response Time (Fall)	t_f	–	3	18	μs	R_L=100 Ω

รูปที่ 5.2 ตาราง Electrical/Optical Characteristics ของ LTV-847

จากรูปจะเห็นได้ว่ามี Response Time Rise อยู่ที่ 4 μS และ Response Time Fall มีค่าสูงสุดอยู่ที่ 18 μS ดังนั้นในการคำนวณจึงใช้ค่ากลางคือ 11 μS ใช้ความเร็วนี้เป็นตัวอ้างอิงความเร็วภาคอินพุต

ภาคเอาต์พุตจากการออกแบบในบทที่ 3 รูปที่ 3.5 ใช้เป็นแบบ แบบทรานซิสเตอร์สวิทช์ NPN Open Collector แบบอาร์เรย์ เบอร์ ULN2308A ของบริษัท SGS-THOMSON MICROELECTRONICS ความเร็วในการทำงานของ เบอร์ ULN2308A จาก ดาต้าชีส ของบริษัทผู้ผลิตหน้า 3 ดังรูปที่ 5.3

ELECTRICAL CHARACTERISTICS (T_{amb} = 25°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit	Fig.
$I_{I(off)}$	Input Current	T_{amb} = 70°C, I_C = 500μA	50	65		μA	4
$V_{I(on)}$	Input Voltage	V_{CE} = 2 V for ULN2802A I_C = 300mA			13	V	5
		for ULN2803A I_C = 200mA			2.4	V	
		I_C = 250mA			2.7	V	
		I_C = 300mA			3	V	
		for ULN2804A I_C = 125mA			5	V	
		I_C = 200mA			6	V	
		I_C = 275mA			7	V	
		I_C = 350mA			8	V	
		for ULN2805A I_C = 350mA			2.4	V	
h_{FE}	DC Forward Current Gain	for ULN2801A V_{CE} = 2V, I_C = 350mA	1000			–	2
C_I	Input Capacitance			15	25	pF	–
t_{PLH}	Turn-on Delay Time	0.5 V_I to 0.5 V_O		0.25	1	μs	–
t_{PHL}	Turn-off Delay Time	0.5 V_I to 0.5 V_O		0.25	1	μs	–
I_R	Clamp Diode Leakage Current	V_R = 50V			50	μA	6
		T_{amb} = 70°C, V_R = 50V			100	μA	6
V_F	Clamp Diode Forward Voltage	I_F = 350mA		1.7	2	V	7

รูปที่ 5.3 ตาราง Electrical Characteristics ของ ULN2803

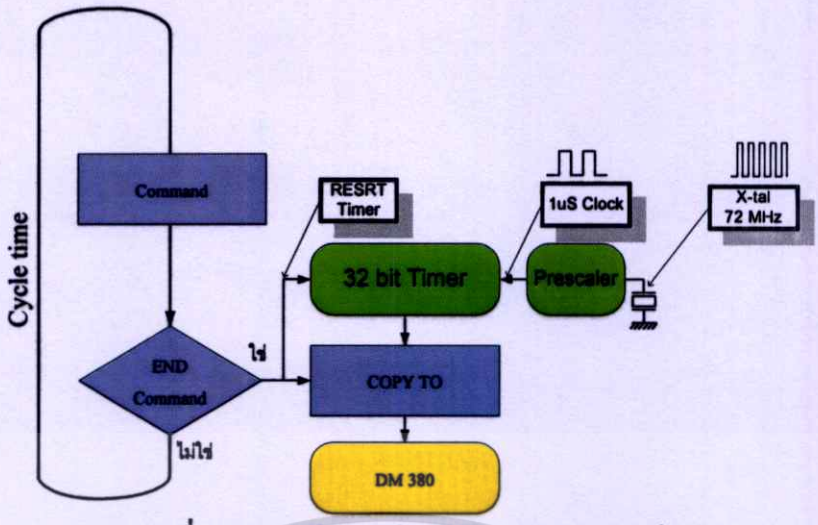
จากรูปจะเห็นได้ว่ามี Turn – ON และ Turn – OFF Delay Time มีค่าสูงสุดอยู่ที่ 1 μS ใช้ความเร็วนี้เป็นตัวอ้างอิงความเร็วภาคเอาต์พุต

สรุป จากดาต้าชีสของอุปกรณ์ด้านอินพุตและเอาต์พุต ที่เราเลือกใช้ในการออกแบบ มีความเร็วอยู่ที่ อินพุต 11 μS และภาคเอาต์พุตอยู่ที่ 1 μS

5.2 ความเร็วในการทำงานของโปรแกรม PLC

การวัดความเร็วของการหาค่าแสกนไทม์ของ PLC เป็นการวัดเป็นการวัดความเร็วในการประมวลผลคำสั่ง โดยเริ่มวัดคาบเวลาตั้งแต่รับค่าอินพุตเข้า PLC ทำการประมวลผลตามโปรแกรมคำสั่ง PLC ส่งค่าเอาต์พุตผลที่ได้ออกมาเป็นการจบคาบเวลาในหนึ่งแสกนไทม์ ในการทดลองนี้เราสามารถวัดคาบเวลาการแสกนไทม์ ของ PLC ได้ 1 วิธี

วิธีที่ 1 วัดโดยใช้ซอฟแวร์และ Timer ที่เป็นโครสร้างภายใน MCU LPC2378 เป็นตัวจับเวลาการทำงาน มีการทำงานดังรูปที่ 5.4

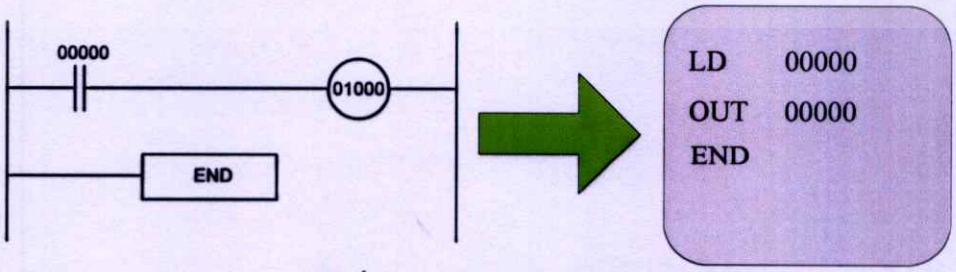


รูปที่ 5.4 การใช้โครสร้างภายในวัดแสกนไทม์

รูปที่ 5.4 การทำงาน ให้ Prescaler ภายใน Timer ทำการหารความถี่ให้ได้ 1MHz หรือสัญญาณนาฬิกาขนาด 1 μS เป็นความถีนับให้แก่ Timer ในขณะที่ PLC ทำงาน Timer จะทำการจับเวลาโดยอัตโนมัติ เมื่อโปรแกรมทำงานถึงคำสั่ง END ซึ่งเป็นเวลาเท่ากับหนึ่งแสกนไทม์ โปรแกรมจะทำการคัดลอก ค่าในเวลา Timer ที่วัดได้ ไปยังหน่วยความจำ PLC Data memory Address 380 จากนั้น จะทำการ Reset Timer เพื่อทำการวัดเวลาในรอบถัดไป และค่าเวลาของแสกนไทม์ ก็จะถูกเก็บใน Data memory DM380 มีหน่วยการวัดเป็น μS

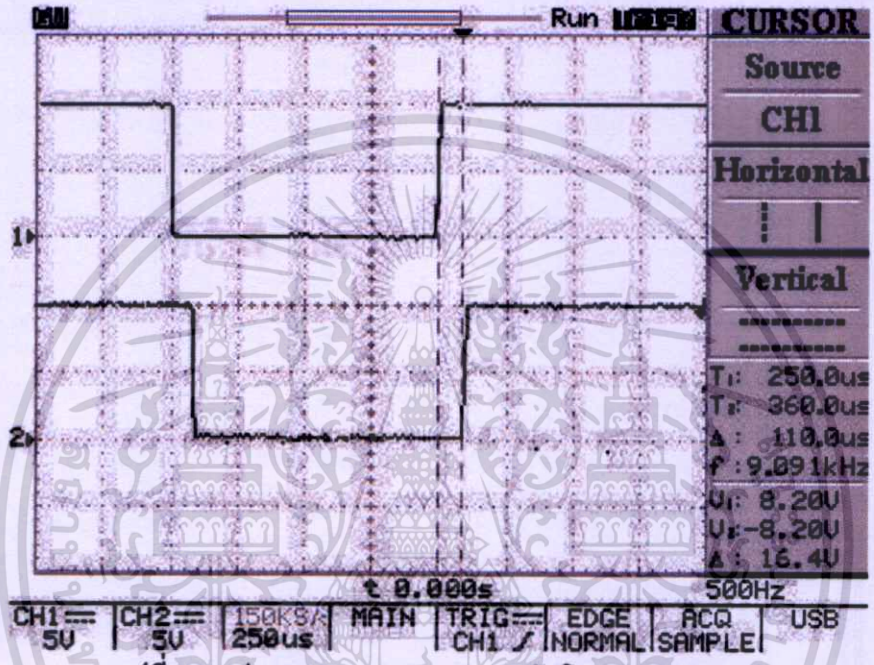
วิธีที่ 2 การวัดแสกนไทม์ จะทำการทดลองโดย ใช้เครื่องมือวัดทางไฟฟ้า Oscilloscope ในการวัดค่าเวลา โดยหาค่าเวลาโดยการจ่ายสัญญาณเข้าที่อินพุต แล้วทำการวัดเวลาจากอินพุตผ่านการประมวลผล แล้วส่งค่าออกเอาต์พุต การวัดแบบนี้ต้องโปรแกรมให้ PLC รับค่าจากอินพุตแล้วประมวลผลแล้วส่งออกเอาต์พุตทันที เป็นโปรแกรมที่ใช้ทดสอบวัดวัดค่าเวลาแสกนไทม์

การทดลองที่ 1 วัดความเร็วแสกนไทม์ ของเครื่องควบคุมแบบ PLC ทดลองโดยใช้ Pulse generator ให้กำเนิดความถี่ขนาด 500 Hz ป้อนเข้าอินพุต PLC อินพุตหมายเลข 00000 และและส่งข้อมูลออกไปทางเอาต์พุตหมายเลข 01000 ตามแลดร์เดอร์โปรแกรม ดังรูปที่ 5.5



รูปที่ 5.5 โปรแกรมการทดลองที่ 1

รูปที่ 5.6 รูปเครื่องมือที่ใช้ในการทดลอง



รูปที่ 5.7 รูปผลการทดลองทดลอง 1 วัดโดย Oscilloscope

ทำการทดลองโดยใช้เครื่องมือวัดทางไฟฟ้าโดย Oscilloscope จากผลการทดลองจากรูปที่ 5.7 สัญญาณ Oscilloscope

ช่องที่ 1 เป็นช่อง สัญญาณอินพุต มีความถี่ ขนาด 500Hz จากรูปอยู่ในส่วนบน ส่วนสัญญาณทางเอาต์พุตจะอยู่ด้านล่างช่องสัญญาณที่ 2 จากรูปจะเห็นได้ว่าสัญญาณเอาต์พุต สัญญาจะล้าหลังสัญญาณอินพุตอยู่ จากค่าที่อ่านได้จาก Oscilloscope คือ 110μS เวลานี้คือ

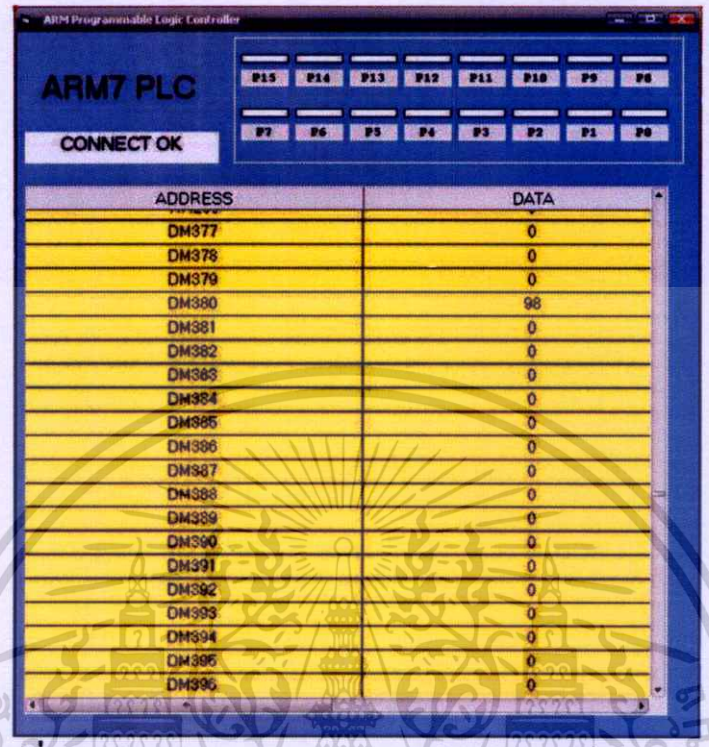
เวลาแสกนไทม์: Delay Time อินพุต + เวลาการประมวลผล + Delay Time เอาต์พุต = 110μS

เวลาแสกนไทม์: 11 μS + เวลาการประมวลผล + 1 μS = 110μS

เวลาการประมวลผล : 110μS - 11 μS - 1 μS = 98 μS

ทำการทดลองวัดความเร็วแสกนไทม์การทำงานของโปรแกรม โดยใช้ซอฟแวร์และ Timer เราสามารถดูการทำงานภายในหน่วยความจำ PLC ผ่านทาง USB Port ด้วยโปรแกรมที่พัฒนาด้วย Microsoft Visual Basic 6.0 ทำหน้าที่ในการอ่านหน่วยความจำ PLC ทั้งหมดขึ้นมาแสดงผล

และ ผ่านโปรแกรมแสดงแสกนไทม์ ทำงานในแต่ละรอบ หน่วยเวลาเป็นไมโครเซ็ก (μSec.) และจัดเก็บไว้ที่ หน่วยความจำ ตำแหน่ง DM380



รูปที่ 5.8 รูปผลการทดลองทดลอง 1 วัดโดย ซอฟแวร์และ Timer

ผลการทดลองนี้ใช้เวลาในประมวลผล 98 μSec ดังรูปที่ 5.8 ค่าที่ได้นี้ไม่ไม่รวมเวลา Delay Time อินพุต/เอาต์พุต

เวลาแสกนไทม์: Delay Time อินพุต + เวลาการประมวลผล + Delay Time เอาต์พุต

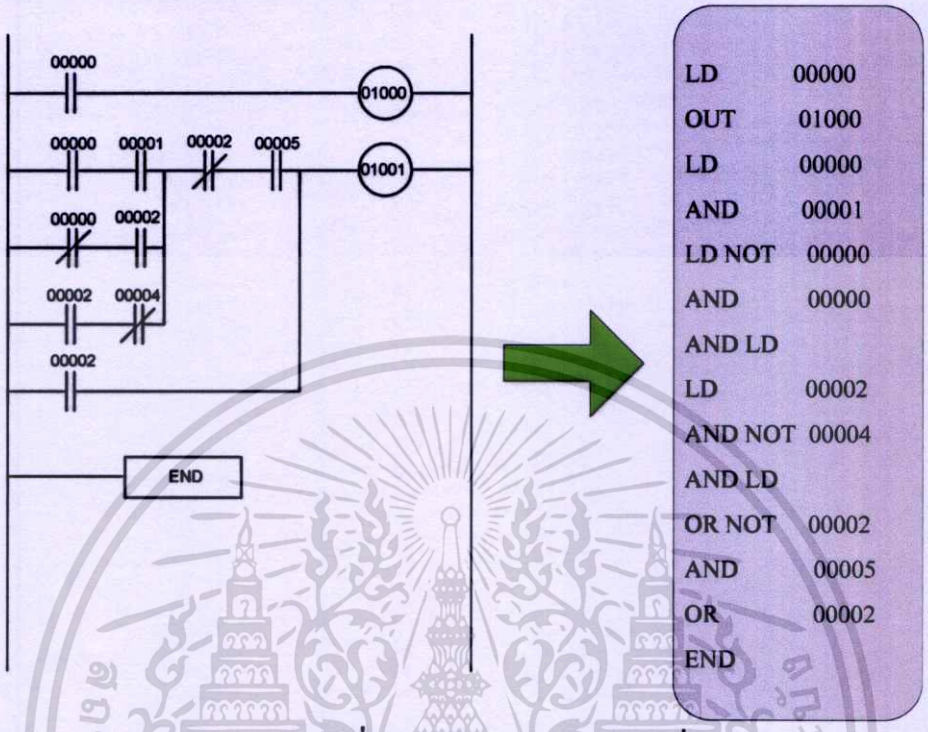
เวลาแสกนไทม์: 18 μS + 98 μS + 1 μS = 117μS

สรุปผลการทดลองที่ 1 การวัดค่าแสกนไทม์ ทั้ง 2 วิธี ให้ค่าออกมาใกล้เคียงกันคือมีผลต่างอยู่ที่ 7 μS ตามตารางที่ 5.1 มีปัจจัยที่ทำให้ค่าที่วัดได้แตกต่างกันอาจจะเกิดจากอุปกรณ์ OPTO ISOLATOR ที่ ดูจากค้าต้าชีส ดังรูปที่ 5.3ในขั้นต้น จะมีเวลา Delay Time ที่ให้มาอยู่ตังแต่ 3-18 μS ทำให้อาจเป็นปัจจัยทำให้ค่าที่อ่านได้มีความแตกต่างกันเล็กน้อย

เวลา / วัดด้วย	Delay Time อินพุต	เวลาการประมวลผล	Delay Time เอาต์พุต	แสกนไทม์
เครื่องมือวัด	1 μS	98 μS	11 μS	110 μS
ซอฟแวร์	1 μS	98 μS	11 μS	110 μS
ผลต่าง	0 μS	0 μS	0 μS	0 μS

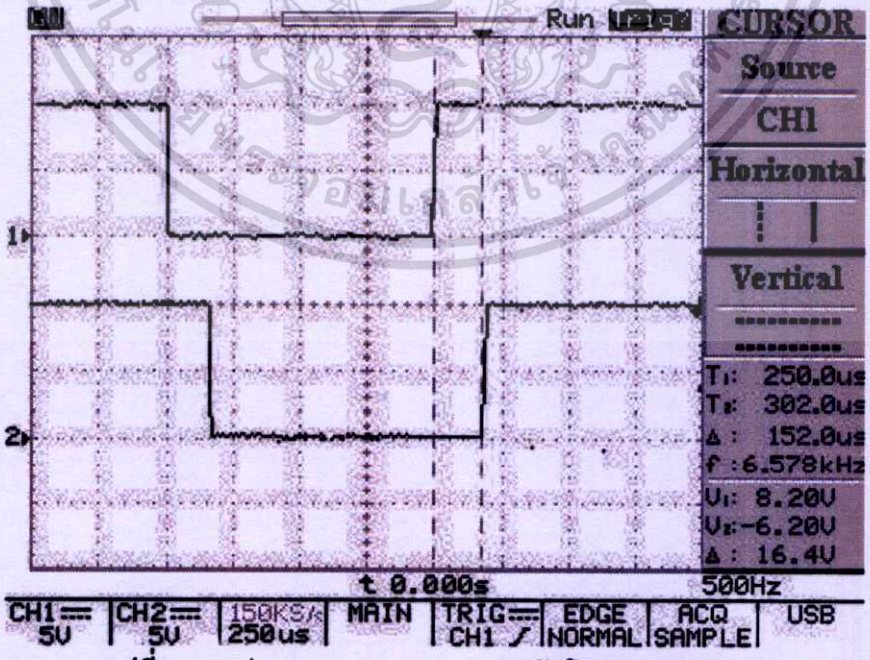
ตารางที่ 5.1 ผลการทดลองที่ 1 การหาค่าแสกนไทม์ ของ PLC

การทดลองที่ 2 วัดความเร็วแสกนไทม์ ของเครื่องควบคุมแบบ PLC โดยเพิ่มความความยาวของโปรแกรม ทำการทดลองเช่นเดียวกับการทดลองที่ 1



รูปที่ 5.9 โปรแกรมการทดลองที่ 1

การทดลองที่ 2 วัดความเร็วแสกนไทม์ขั้นพื้นฐานที่ใช้ในการประมวลผลในการปรับปรุงค่าหน่วยความจำภายใน PLC อ่านค่าอินพุตเอาต์พุต



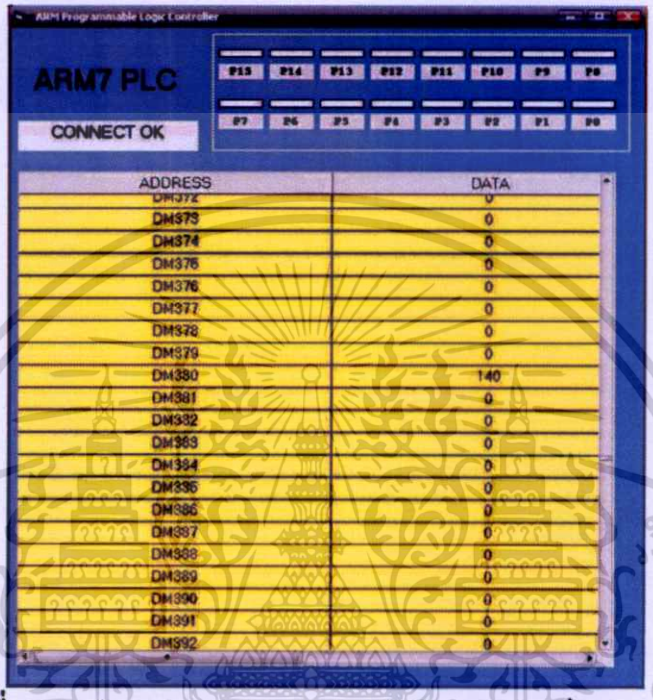
รูปที่ 5.10 รูปผลการทดลองทดลอง 2 วัดโดย Oscilloscope

จากรูปที่ 5.9 จะเห็นได้ว่าสัญญาณเอาต์พุต สัญญาจะล้าหลังสัญญาณอินพุตอยู่ จากค่าที่อ่านได้จาก Oscilloscope คือ 152µS เวลานี้คือ

เวลาแสกนไทม์: Delay Time อินพุต + เวลาการประมวลผล + Delay Time เอาต์พุต = 152µS

เวลาแสกนไทม์: 11 µS + เวลาการประมวลผล + 1 µS = 152µS

เวลาการประมวลผล : 152µS - 11 µS - 1 µS = 140 µS



รูปที่ 5.11 รูปผลการทดลองทดลอง 2 วัดโดย ซอฟแวร์และ Timer

ผลการทดลองนี้ใช้เวลาในประมวลผล 140 µSec ตามรูปที่ 5.11 ค่าที่ได้นี้ไม่ไม่รวมเวลา Delay Time อินพุต/เอาต์พุต

เวลาแสกนไทม์: Delay Time อินพุต + เวลาการประมวลผล + Delay Time เอาต์พุต

เวลาแสกนไทม์: 11 µS + 140 µS + 1 µS = 152µS

สรุปผลการทดลองที่ 2 จะมีเวลาการประมวลที่สูงขึ้นตามโปรแกรมคำสั่งที่มากขึ้น เป็นผลให้เวลาในการ ค่าแสกนไทม์สูงขึ้นตามจากการทดลองทั้ง 2 ครั้งเวลาในการประมวลผลจากการวัดจากการทดลองทัง 2 ครั้ง

เวลา / วัดด้วย	Delay Time อินพุต	เวลาการประมวลผล	Delay Time เอาต์พุต	แสกนไทม์
เครื่องมือวัด	1 µS	140 µS	11 µS	152 µS
ซอฟแวร์	1 µS	140µS	11 µS	152µS
ผลต่าง	0 µS	0 µS	0 µS	0 µS

ตารางที่ 5.2 ผลการทดลองที่ 2 การหาค่าแสกนไทม์ ของ PLC

จากผลการทดลองที่ 1 และ 2 จะเห็นได้ว่า ค่าที่วัดได้จากทั้ง เครื่องมือวัด Oscilloscope และซอฟแวร์ นั้นมีค่าที่เท่ากัน ทำให้โปรแกรมที่ออกแบบขึ้นมาเพื่อการใช้วัดมีความน่าเชื่อถือในการใช้วัดแสกนไทม์ ของ PLC ได้

5.3 การทดสอบประสิทธิภาพของแสกนไทม์ตามจำนวนคำสั่ง

การทดลองที่3 การทดสอบประสิทธิภาพการทำงานของ PLC โปรแกรมที่ใช้ทดสอบเป็นโปรแกรมผู้ใช้งานที่มีความจุคำสั่งผู้ใช้งานได้มากถึง 16 Kstep โดยการจำลองคำสั่งให้เพิ่มจากน้อยไปหามากเพิ่มครั้งละ 0.5 Kstep มีสัดส่วนคำสั่งระดับบิต 70% และคำสั่งระดับเวิลด์ 30% สัดส่วนนี้ได้มีการวิจัยมาแล้วว่าเหมาะสมกับการทดสอบผลการทดสอบ เริ่มต้นที่คำสั่ง 10 step, 0.5 , 1, 1.5, 2 ,....16 Kstep แล้วบันทึกเวลาการสแกนของ PLC โดยเวลาการสแกนจะอ้างอิงจาก ฐานเวลา 1 μsec ดังรูปที่ 5.4 เพื่อใช้ตรวจจับเวลาในแต่ละคาบที่ซีพียูทำงานครบ 1 รอบว่านานเท่าใด จากนั้นค่อยนำผลมาแสดงยังหน่วย ความจำ DM0380 ของ PLC โดยสามารถดูค่าได้จาก ทาง USB Port และ Embedded Web Server

ตารางที่ 5.3 คำสั่งและเวลาในการปฏิบัติงานคำสั่งบางส่วน

บางส่วนคำสั่งระดับบิท		บางส่วนคำสั่งระดับเวิลด์	
คำสั่ง IL	เวลาปฏิบัติ (μsec)	คำสั่ง IL	เวลาปฏิบัติ (μsec)
LD	4	CMP S,D	2
LD NOT	4	MOV S,D	2
AND	2	ADD S,A,R	1
OR	2	SUB S,S,R	1
OUT	3	MUL S,M,R	1

จากตารางที่ 5.3 แสดงบางส่วนของคำสั่ง IL ระดับบิทและเวิลด์ จะเห็นว่าเวลาการปฏิบัติคำสั่งระดับบิทจะมากกว่าเป็นเพราะการพัฒนาโปรแกรมบน ARM7TDMI-S ไม่ค่อยสนับสนุนคำสั่งในระดับบิทจำเป็นต้องเอาคำสั่งเวิลด์มาประยุกต์ใช้แทน เวลาการปฏิบัติคำสั่งมีหน่วยเป็น μsec การทดสอบ

โปรแกรมที่ใช้ในการทดลองตามตารางที่ 5.4 เป็นโปรแกรมขนาด 10 step ที่มีสัดส่วนของโปรแกรมคำสั่งระดับบิต และคำสั่งระดับเวิลด์ 70% และ 30% ตามลำดับ การทดลองจะใช้ชุดคำสั่งดังกล่าวทำการทดลองโดยใช้คำสั่งชุดเดิมเพิ่มคำสั่งชุดเดิมเพิ่มเข้าไปครั้งละ 500 step คำสั่ง จากนั้นจะที่การวัด

แสกนไทม์ทั้งแบบมีการ เปิดการใช้งาน Online Ethernet port และ Offline Ethernet port ในขณะ Offline จะดูค่า แสกนไทม์ ผ่านทาง USB port และขณะ Online จะดูค่าผ่านทาง Embedded Web Server

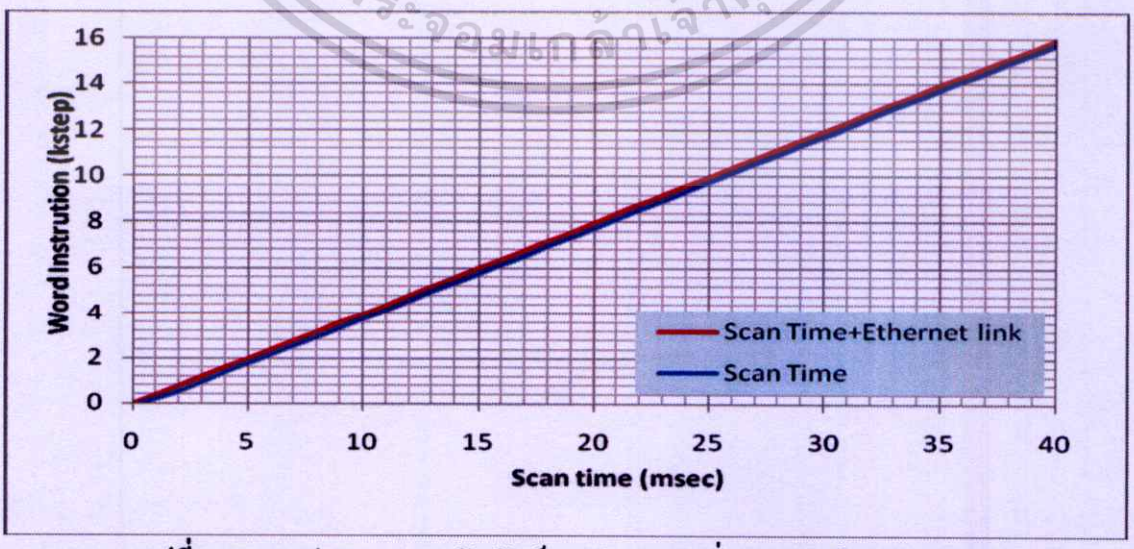
ตารางที่ 5.4 โปรแกรมที่ใช้ทดสอบประสิทธิภาพของแสกนไทม์ตามจำนวนคำสั่ง

ตำแหน่ง	คำสั่ง IL	เวลาในการทำงาน
0000	LD 00000	4
0001	OUT 01000	3
0002	LD 00001	4
0003	AND 00002	2
0004	OR 00003	2
0005	OUT 01000	3
0006	LD 00000	4
0007	ADD DM000 DM001 DM002	1
0008	SUB DM0002 DM0001 DM0003	1
0009	SUB DM0004 DM0005 DM0006	1
0010	END	91

รูปที่ 5.12 การอ่านแสกนไทม์ ผ่านทางโปรแกรม Internet Explorer

ตารางที่ 5.5 จำนวนคำสั่งกับเวลาการแสกนไทม์ที่ Online และ Offline พอร์ท Ethernet

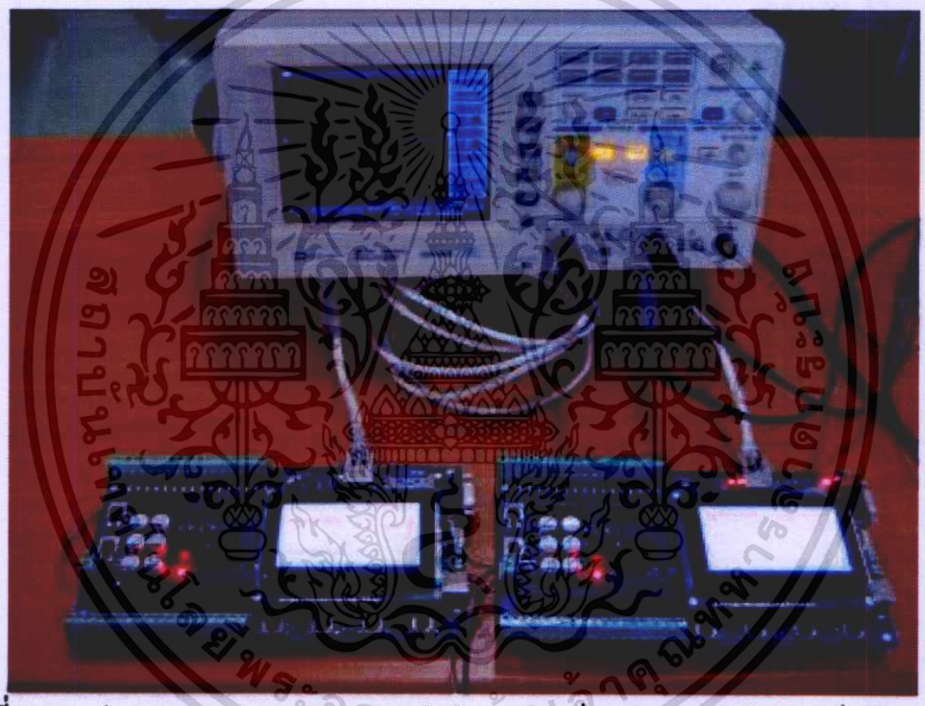
จำนวนสเต็ป (Kstep)	Online (msec)	Offline (msec)	จำนวนสเต็ป (Kstep)	Online (msec)	Offline (msec)
0.01	0.389	0.116	8.5	21.878	21.341
0.5	1.878	1.341	9.0	23.128	22.591
1.0	3.128	2.591	9.5	24.378	23.841
1.5	4.378	3.841	10.0	25.628	25.091
2.0	5.628	5.091	10.5	26.878	26.341
2.5	6.878	6.341	11.0	28.128	27.591
3.0	8.128	7.591	11.5	29.378	28.841
3.5	9.378	8.841	12.0	30.628	30.091
4.0	10.628	10.091	12.5	31.878	31.341
4.5	11.878	11.341	13.0	33.128	32.591
5.0	13.128	12.591	13.5	34.378	33.841
5.5	14.378	13.841	14.0	35.628	35.091
6.0	15.628	15.091	14.5	36.878	36.341
6.5	16.878	16.341	15.0	38.128	37.591
7.0	18.128	17.591	15.5	39.378	38.841
7.5	19.378	18.841	16.0	40.628	40.091
8.0	20.628	20.091			



รูปที่ 5.13 กราฟแสดงความสัมพันธ์ของจำนวนคำสั่งและเวลาในการแสกน

ผลการทดลองที่ 3 จากตารางที่ 5.5 เป็นผลการทดลองจำนวนโปรแกรมที่ 0.5 Kstep ใช้เวลาในการแสกน 1.578 msec และ1.341 msec และจำนวนโปรแกรมที่16 Kstepใช้เวลาในการแสกน 40.328 msec และ 40.091 msec เมื่อทำการOnline และ Offline พอร์ท Ethernet ตามลำดับ สามารถนำมาแสดงเป็นกราฟความสัมพันธ์ดังรูปที่ 5.13 ในการทดลองวัดค่าเวลาในการแสกน ขณะมีการติดต่อรับส่งข้อมูลทาง Ethernet นั้นจะทำให้เวลาเพิ่มขึ้นจากเดิมอีก 273 μSec ซึ่งนับว่าน้อยมากเมื่อเทียบกับโปรแกรมการทำงานทั้งหมดของ PLC

5.4 การทดสอบเวลาการตอบสนองต่ออินพุต/เอาต์พุต ของ PLC ผ่านพอร์ท Ethernet (I/O Response Time)

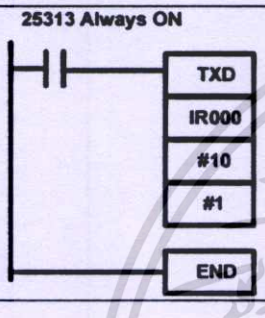
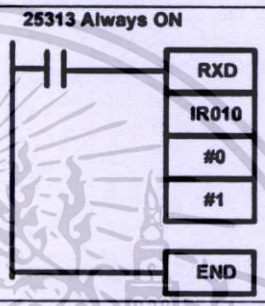


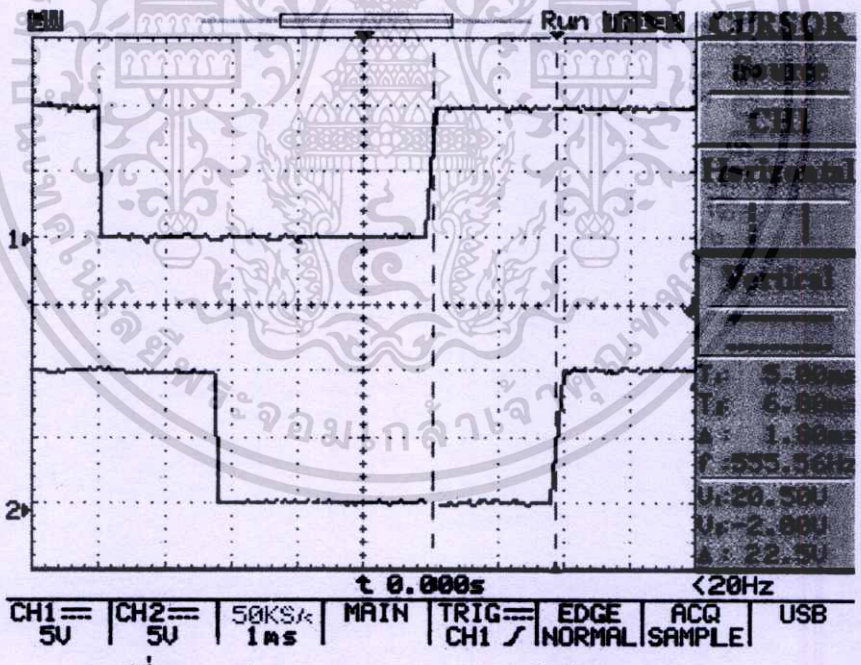
รูปที่ 5.14 รูปการทดสอบการตอบสนองต่ออินพุต/เอาต์พุต ของ PLC ผ่านพอร์ท Ethernet

การทดลองที่ 4 การทดสอบเวลาการตอบสนองต่ออินพุต/เอาต์พุต ของ PLC ผ่านพอร์ท Ethernet ทดสอบโดยใช้ PLC จำนวน 2 ชุดต่อผ่านพอร์ท Ethernet ตามรูปที่ 5.14 และเขียนโปรแกรมด้วยคำสั่ง IL ตามตารางที่ 5.6 โดยให้ชุดที่ 1โหนดหมายเลข #01 หรือ IP ADDRESS:192.168.1.1 เป็นชุดรับสัญญาณอินพุตจากสวิทช์ลงหน่วยความจำอินพุทแล้วเขียนโปรแกรมส่งข้อมูลด้วยคำสั่ง TXD ส่งข้อมูลผ่าน พอร์ท Ethernet ในทางกลับกันเขียนโปรแกรมรับข้อมูลด้วยคำสั่ง RXD ให้กับ PLC ชุดที่ 2 โหนดหมายเลข #10 หรือ IP ADDRESS:192.168.1.10 เพื่อรับข้อมูลจากพอร์ท Ethernet ส่งมายังหน่วยความจำเอาท์พุทและส่งข้อมูลออกไปอุปกรณ์ภายนอกทางฝั่งอินพุตของ PLC ชุดที่ 1 ป้อนสัญญาณพัลส์ความถี่ 100Hz หรือขนาด 10 msec

เป็นความถี่ที่เครื่องควบคุมตอบสนองได้ทัน แล้วใช้ออสซิลโลสโคป(Oscilloscope) ทำการเปรียบเทียบระหว่างสัญญานอินพุตของ PLC ชุดที่ 1 ด้วย OSC. Ch1 และ เอาต์พุตของ PLC ชุดที่ 2 ด้วย OSC. Ch2 ตามลำดับ

ตารางที่ 5.6 ตารางโปรแกรม PLC การทดลองที่ 4

PLCชุดที่ 1 Unit #01 โปรแกรมส่งข้อมูลอินพุตผ่าน พอร์ทEthernet ขนาด 1 word		PLC ชุดที่ 2 Unit #10 โปรแกรมรับข้อมูลเอาต์พุตผ่าน พอร์ทEthernet ขนาด 1 word	
โปรแกรม LD	โปรแกรม IL	โปรแกรม LD	โปรแกรม IL
	LD 25313 TX 000 #10 #01 END		LD 25313 TX 010 #01 #01 END

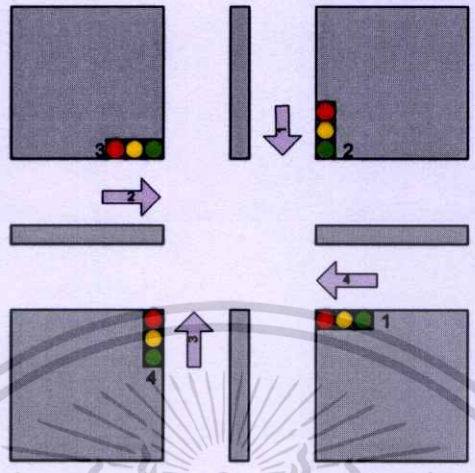


รูปที่ 5.14 รูปผลการทดลองทดลอง 4 วัดโดย Oscilloscope

จากผลการทดลองที่ 4 ตามที่รูป 5.14 เมื่อเปรียบเทียบอินพุตจาก PLC ชุดที่1 และเอาต์พุตจาก PLC ชุดที่ 2 สัญญาณเอาต์พุตจะล้าหลังตามหลังสัญญาณอินพุตอยู่ 1.80 msec.

5.5 การประยุกต์ใช้ PLC ควบคุมไฟจราจร

การทดลองที่ 5 การทดลองประยุกต์ใช้งาน PLC ในการควบคุมไฟจราจรโดยมีสี่แยกจราจร ดังรูปที่ 5.16 โดยจำลองให้เป็นไฟจราจรสี่แยกขนาดใหญ่ปล่อยรถที่ละทาง ตามลำดับ 1 ถึง 4 การ



รูปที่ 5.16 รูปแผนผังไฟสัญญาณจราจรสี่แยก

ขั้นตอนการทำงาน

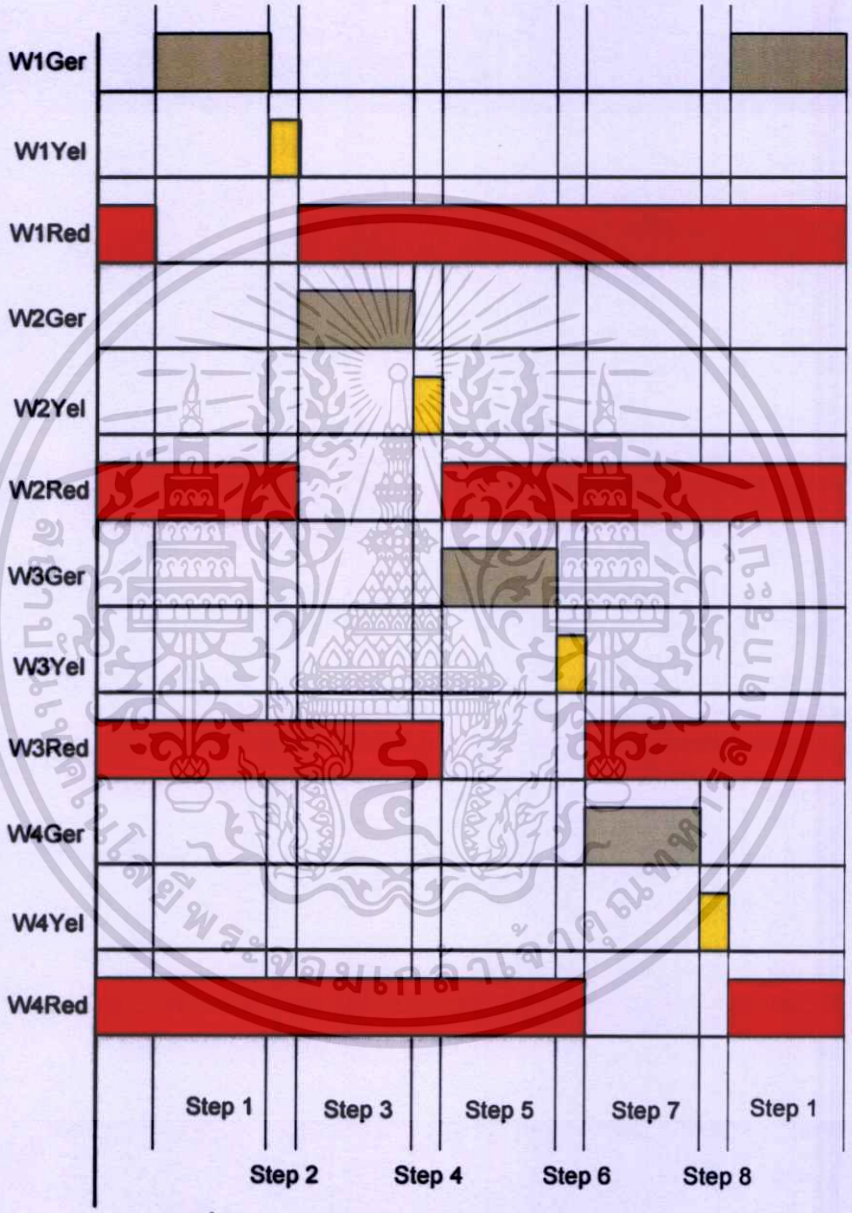
เป็นการควบคุมสัญญาณไฟสำหรับรถที่แล่นผ่านสี่แยก เริ่มจากปล่อยรถในทางที่ 1 เป็นเวลา 120 วินาที ด้วยสัญญาณจราจร ไฟสีเขียว 115 วินาที และ ทำการเตือนด้วยสัญญาณจราจร สีเหลือง 5 วินาที่ จากนั้น ให้สัญญาณหยุดรถด้วย สัญญาณไปจราจรสี่แดง แล้วจึงทำการปล่อยรถในช่องการจราจรต่อไป ทำเช่นนี้จนครบ 4 ช่องสัญญาณจราจร ตามไทม์มิ่งไดอะแกรม ดังรูปที่ 5.16

ตารางที่ 5.7 การกำหนดอินพุต/เอาต์พุต ของไฟจราจร

ตำแหน่ง	ความหมาย
01000	W1Ger เอาต์พุตไฟสัญญาณไฟเขียว จุดที่1
01001	W1Yel เอาต์พุตไฟสัญญาณไฟเขียว จุดที่1
01002	W1Red เอาต์พุตไฟสัญญาณไฟเขียว จุดที่1
01003	W2Ger เอาต์พุตไฟสัญญาณไฟเขียว จุดที่2
01004	W2Yel เอาต์พุตไฟสัญญาณไฟเขียว จุดที่2
01005	W2Red เอาต์พุตไฟสัญญาณไฟเขียว จุดที่2
01006	W3Ger เอาต์พุตไฟสัญญาณไฟเขียว จุดที่3
01007	W3Yel เอาต์พุตไฟสัญญาณไฟเขียว จุดที่3
01008	W3Red เอาต์พุตไฟสัญญาณไฟเขียว จุดที่3

ตารางที่ 5.7 (ต่อ) การกำหนดอินพุต/เอาต์พุต ของไฟจราจร

01009	W4Ger เอาต์พุตไฟสัญญาณไฟเขียว จุดที่4
01010	W4Yel เอาต์พุตไฟสัญญาณไฟเขียว จุดที่4
01011	W4Red เอาต์พุตไฟสัญญาณไฟเขียว จุดที่4



รูปที่ 5.17 ไทม์มิ่งไดอะแกรม ของไฟจราจร

ตารางที่ 5.8 โปรแกรมควบคุมไฟจราจร

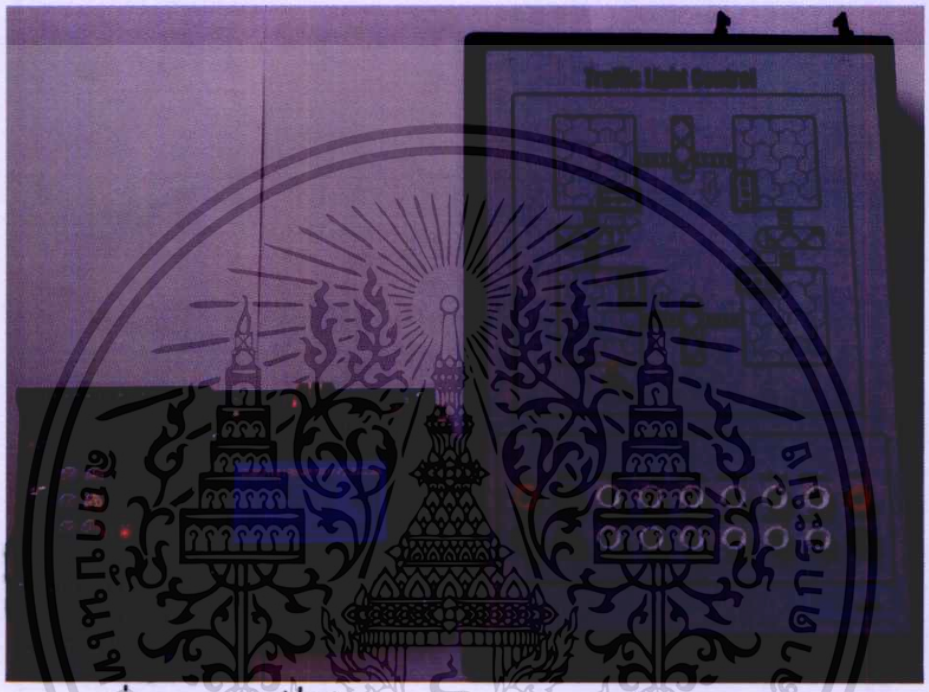
ตำแหน่ง	คำสั่ง IL	รหัสคำสั่งที่ผ่านแปลคำสั่ง
0000	LD 20000	:C9000C80
0001	TIM0 #1150	D3000000 :0001047E
0002	LD TIM0	:C9000580
0003	AND NOT TIM001	:CC000581
0004	OUT 20000	:CF000C80
0005	LD TIM0	:C9000580
0006	OR 20000	:CD000C80
0007	TIM1 #50	D3000001 :00010032
0008	LD TIM0	:C9000580
0009	OR TIM1	:CD000581
0010	INC DM000	:26003000
0011	LD 25313	:C9000FDD
0012	CMP DM000 #000	:14003000 :00010000
0013	LD 25506	C9000FF6
0014	MOV #2337 IR10	:15010921 :0000000A
0015	CMP DM000 #001	:14003000 :00010001
0016	LD 25506	:C9000FF6
0017	MOV #2338 IR10	:15010922 :0000000A
0018	CMP DM000 #002	:14003000 :00010002
0019	LD 25506	:C9000FF6

ตารางที่ 5.8 (ต่อ) โปรแกรมควบคุมไฟจราจร

ตำแหน่ง	คำสั่ง IL	รหัสคำสั่งที่ผ่านแปลคำสั่ง
0020	MOV #2348 IR10	:1501092C :0000000A
0021	CMP DM000 #003	:14003000 :00010003
0022	LD 25506	:C9000FF6
0023	MOV #2324 IR10	:15010914 :0000000A
0024	CMP DM000 #004	:14003000 :00010004
0025	LD 25506	:C9000FF6
0026	MOV #2244 IR10	:150108C4 :0000000A
0027	CMP DM000 #005	:14003000 :00010005
0027	LD 25506	:C9000FF6
0029	MOV #2212 IR10	:150108A4 :0000000A
0030	CMP DM000 #006	:14003000 :00010006
0031	LD 25506	:C9000FF6
0032	MOV #804 IR10	150108A4 :0000000A
0033	CMP DM000 #007	:14003000 :00010007
0034	LD 25506	:C9000FF6
0035	MOV #1316 IR10	150108A4 :0000000A
0033	CMP DM000 #008	:14003000 :00010008

ตารางที่ 5.8(ต่อ) โปรแกรมควบคุมไฟจราจร

ตำแหน่ง	คำสั่ง IL	รหัสคำสังที่ผ่านแปลคำสั่ง
0034	LD 25506	:C9000FF6
0035	@MOV #0000 DM000	:150108A4 :00003000
0036	END	:01000000



รูปที่ 5.18 อุปกรณ์ที่ใช้ทดสอบโปรแกรมควบคุมไฟจราจร ของ PLC

จากโปรแกรมตารางที่ 5.8 นำมาทดลองในโมเดล ตามอุปกรณ์ รูปที่ 5.18 ทำการทดสอบการทำงาน PLC ทำงานเป็นไปตามไทม์มิ่งไดอะแกรม ของไฟจราจร ตามที่กำหนด ตามรูปที่ 5.17 เวลาแสกนไทม์ของ PLC ที่ใช้ไปคือ 273 µSec

บทที่ 6
สรุปผลการวิจัยและข้อเสนอแนะ

ปัจจุบันนี้ PLC ได้เข้ามามีบทบาทในการควบคุมการทำงานของเครื่องจักรในอุตสาหกรรมมากขึ้น และจำเป็นอย่างยิ่งต่อการพัฒนาเทคโนโลยีทางการผลิต ซึ่ง PLC ที่ใช้งานกันอยู่ในปัจจุบันส่วนใหญ่เป็น PLC ที่นำเข้ามาจากต่างประเทศ ที่มีราคาข้อนข้างสูง และในการใช้งานอาจจำเป็นต้องเพิ่มค่าใช้จ่าย ในส่วนเของเครื่องโปรแกรม (Programming console) หรือโปรแกรมในการพัฒนาในส่วนของโปรแกรม PLC มาด้วยทำให้เพิ่มค่าใช้จ่ายในกระบวนการผลิตมากขึ้นและ

ในวิทยานิพนธ์ฉบับนี้ได้นำเสนอการใช้เทคโนโลยีด้านสมองกลฝังตัว โดยใช้ไมโครคอนโทรลเลอร์ 32 บิต ในตระกูล ARM7TDMI-S เบอร์ LPC2378 ซึ่งมีโครงสร้างภายในเอื้อต่อการนำมาออกแบบเป็นเครื่องควบคุม PLC ได้อย่างลงตัว และเนื่องจาก PLC สามารถทำงานแปลงคำสั่ง IL ในรูปแบบไฟล์ตัวอักษร ให้ทำงานได้ทันทีจึงลดความยุ่งยากในการใช้เครื่องป้อนโปรแกรม (Programming console) และสะดวกในการใช้งานอย่างมาก ที่สำคัญยังสามารถเชื่อมต่อบนเครือข่าย Ethernet ให้ทำงานได้ทั้งโหมดการปรับปรุง แก้ไข โปรแกรม และโหมดการลิงค์สัญญาณเครื่องควบคุมเข้าด้วยกัน จากการทดลองผลที่ได้รับเป็นที่พอใจในระดับหนึ่ง และสามารถปรับปรุงให้มีประสิทธิภาพ และ มีเสถียรภาพมากขึ้นทั้งในด้านการเขียนโปรแกรมผู้ใช้งาน การสื่อสารข้อมูล และการควบคุมการทำงาน นอกจากนี้ชุดคำสั่งของ PLC ที่มีอยู่ขณะนี้มีเพียง 49 คำสั่งสามารถใช้งานได้ในระดับหนึ่งเช่นกัน อาจจะไม่ครอบคลุมการประยุกต์ใช้งานในทุกรูปแบบ ดังนั้นสามารถจะพัฒนาต่อให้มีชุดคำสั่งที่หลากหลายมากขึ้นไปอีกทั้ง ทางด้านการสื่อสารข้อมูลซอฟท์แวร์ สามารถเพิ่มเติมโปรโตคอลคำสั่ง ที่ใช้ในการสื่อสารข้อมูลทั้งในส่วนพอร์ทอนุกรม และ ส่วนของ Can bus จะทำให้เครื่องควบคุมในงานวิจัยนี้มีความสมบูรณ์มากยิ่งขึ้น

บรรณานุกรม

[1] ประชา พฤกษ์ประเสริฐ,"สร้างเว็บและเพิ่มลูกเล่นด้วย HTML & XHTML",ซัคเซส มีเดีย, บจก. , 2552

[2] ณรงค์ ตันชีวะวงศ์, "ระบบ PLC "ส่งเสริมเทคโนโลยี (ไทย-ญี่ปุ่น) ,2541

[3] ดร.ธนารักษ์ ธีระมั่นคง, "เทคโนโลยีสมองกลฝังตัว Embedded Technology", ส่งเสริมเทคโนโลยี(ไทย-ญี่ปุ่น),2549

[4] ทีมงาน SCM INSTITUTION, "คล่องแคล่ว PLC", พีดีเอแม็กซ์, 2548

[5] ธีรศิลป์ ทุมวิภาต , "เรียนรู้ PLC ขั้นต้นด้วยตนเอง ,พิมพ์ลักษณ์ กรุงเทพฯ", ซีเอ็ดยูเคชั่น, 2545

[6] ธีรศิลป์ ทุมวิภาต, สุภาพร จำปาทอง, "เรียนรู้ PLC ขั้นกลางด้วยตนเอง",ซีเอ็ดยูเคชั่น ,2547

[7] นคร ภักดีชาติ , อรรถพล บุญยะโภคา , โอภาส ศิริครรชิตถาวร , ชัยวัฒน์ ลิ้มพรจิตรวิไล,คู่มือ "ทดลองไมโครคอนโทรลเลอร์ 32 บิต ตระกูล ARM7 เบื้องต้นฉบับ LPC2148", โรงพิมพ์วัชรินสาสน์รัชดา,2549

[8] บริษัท ซีเมนส์ จำกัด และทีมงาน SCMi, "คล่องแคล่ว PLC 2 ตอน การใช้งาน Siemens S7-200 ระดับกลาง",สมาร์ทโฟน,2550

[9] สุวัฒน์ ปุณณชัยยะ, ต้น ตัณฑ์สุทธิวงศ์, สุพจน์ ปุณณชัยยะ, "เปิดโลก TCP/IP และโปรโตคอลของอินเตอร์เน็ต",โปรวิชั่น,2550

[10] เสนีย์ เกียรติสุนทร, "หลักการทำงานและเทคนิคการประยุกต์ใช้งาน PLC", ซีเอ็ดยูเคชั่น, 2548

[11] Gab Seon Rho ,Kyeong-hoon Koo and Naehyuck "Chang, Implementation of a RISC Microprocessor for programmable logic controllers", Microprocessors Microsystems. Vol. 19 , pp 599-671,1992

[12] Kim,Jomg-il,Park,J and Kwon,W H',"Architecture of a ladder solving processor for [1] ProgrammableControllers", Microprocessors Microsystems. Vol. 16,pp 369-379,1992

[13] NXP, UM10211, "LPC2378 User manual", [Online]. Available : http://www.standardics.nxp.com/support/documents/microcontrollers/?scope=LPC2000

[14] Philip J, Koopman, Jr, "Beyond the Embedded Web Server" ,pp 143, 1996

[15] "Programmable Controllers-Part 3: Programming Languages" ,IEC 1131-3,pp 15,1993

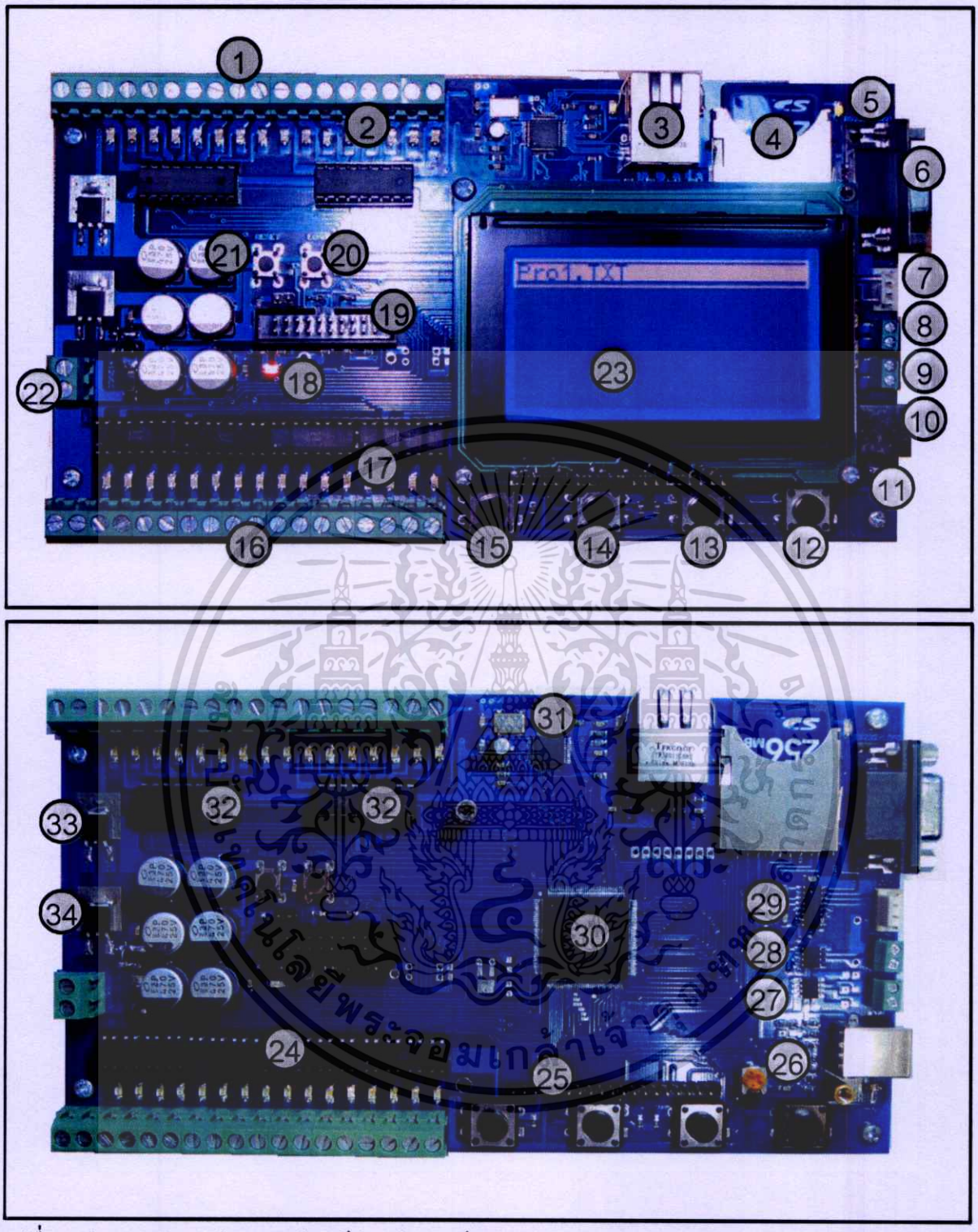
[16] Sergio Scaglia, "The Embedded Internet: TCP/IP Basics, Implementation and Applications 1ED (H)" , Pearson Education Indochaina Ltd.,1999

[17] Suphan Gulpanich, Ajin Numsomran,Prapas Roengruen,Viriya Kongratana and Kitti Tirasesth, "Design of programmable logic controllers and I/O Expansions", ICCAS2005 June 2-5, KINTEX,Gyeonggi-Do,Korea, pp1107-1111,2005

ภาคผนวก

ภาคผนวก ก.

โครงสร้างของ PLC ที่ออกแบบ



รูปที่ 1 รูปแสดง ตำแหน่งของอุปกรณ์ต่างๆในบอร์ด KMITL PLC V2

หมายเลข 1 คือ TERMINAL BLOCK เอาต์พุต

หมายเลข 2 คือ LED แสดงการทำงานของเอาต์พุต

หมายเลข 3 คือ ขั้วต่อสัญญาณ Ethernet LAN แบบ RJ45

หมายเลข 4 คือ ช่องเสียบการ์ดหน่วยความจำสามารถใช้ได้กับ SD Card และ MMC Card

หมายเลข 5 คือ LED แสดงสถานะของ แหล่งจ่ายไฟของการ์ดหน่วยความจำ SD/MMC

หมายเลข 6 คือ ขั้วต่อ UART-0(RS232) สำหรับใช้งาน และ Download Hex File ให้ CPU

หมายเลข 7 คือ ขั้วต่อ UART-1(RS232) สำหรับใช้งาน

หมายเลข 8 คือ ขั้วต่อ UART-3(RS485) สำหรับใช้งาน

หมายเลข 9 คือ ขั้วต่อ CAN-1(CAN BUS) สำหรับใช้งาน

หมายเลข 10 คือ ขั้วต่อ USB สำหรับเชื่อมต่อกับ USB Hub รุ่น 2.0

หมายเลข 11 คือ LED แสดงค่าสถานะของการทำงานและการเชื่อมต่อของ USB

หมายเลข 12 คือ สวิตช์ Enter สำหรับควบคุม

หมายเลข 13 คือ สวิตช์ Down สำหรับควบคุม

หมายเลข 14 คือ สวิตช์ Up สำหรับควบคุม

หมายเลข 15 คือ สวิตช์ Back สำหรับควบคุม

หมายเลข 16 คือ TERMINAL BLOCK อินพุต

หมายเลข 17 คือ LED แสดงการทำงานของอินพุต

หมายเลข 18 คือ LED Power สำหรับแสดงสถานะของ Power Supply

หมายเลข 19 คือ ขั้วต่อ JTAG ARM สำหรับ Debug แบบ Real Time

หมายเลข 20 คือ สวิตช์ Download สำหรับ Download โปรแกรมผ่านทาง

หมายเลข 21 คือ สวิตช์ RESET สำหรับ RESET การทำงานของ MCU

หมายเลข 22 คือ ขั้วต่อไฟเลี้ยงวงจร +12V จากภายนอก

หมายเลข 23 คือ LCD Graphic Displays ขนาด 128 x 64 doc

หมายเลข 24 คือ IC OPTO ISOLATOR เบอร์ LTV-847 ภาคอินพุต

หมายเลข 25 คือ ขั้วต่อ LCD Graphic

หมายเลข 26 คือ VR สำหรับปรับค่าความสว่างให้ LCD Graphic

หมายเลข 27 คือ IC Line Driver ของ CAN Transmit และ เบอร์ 75176

หมายเลข 28 คือ IC Line Driver ของ RS485 Transmit

หมายเลข 29 คือ IC Line Driver ของ RS232 Transmit

หมายเลข 30 คือ MCU เบอร์ LPC2378 (144Pin LQFP)

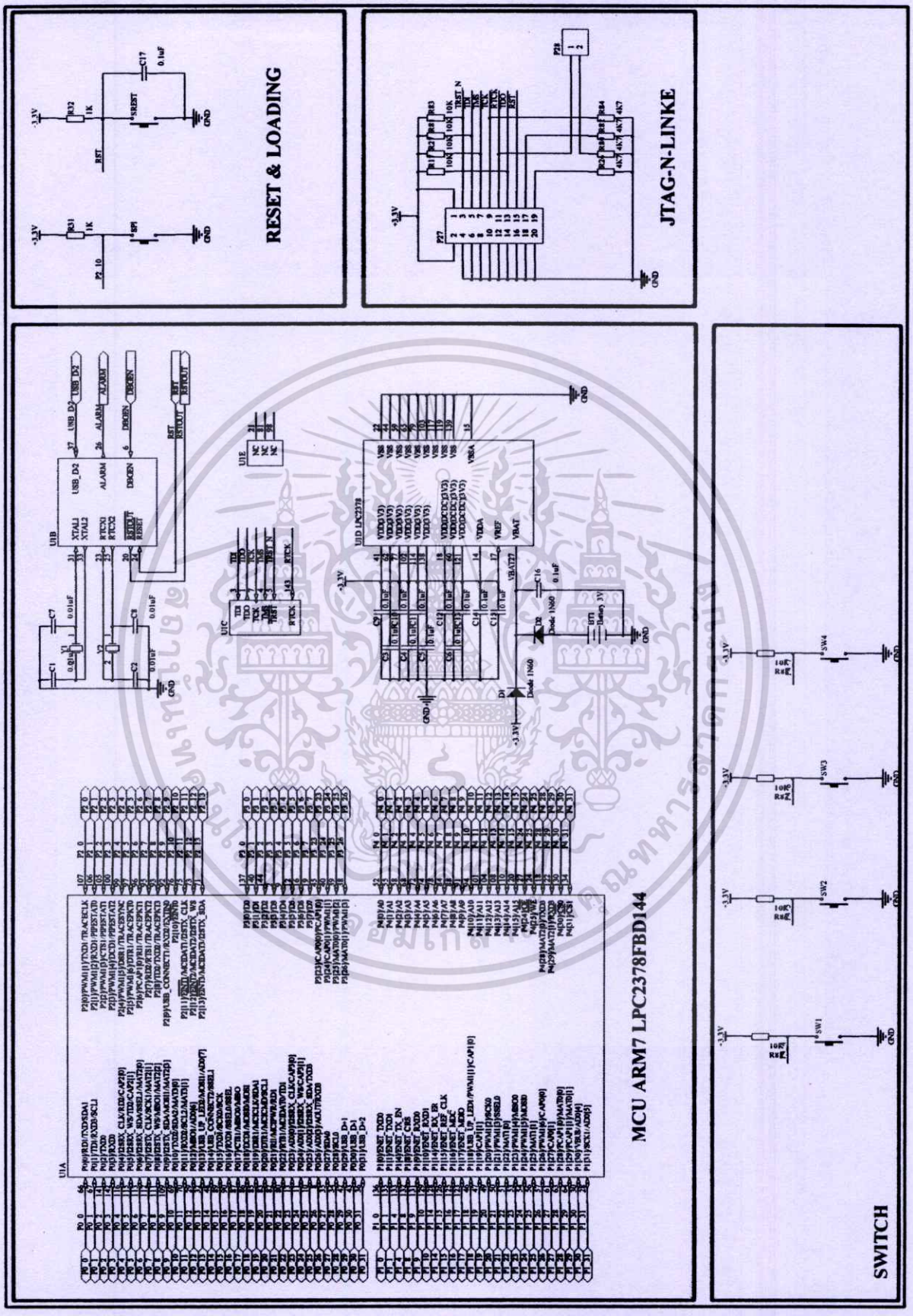
หมายเลข 31 คือ IC Physical Ethernet Driver เบอร์ DP83848

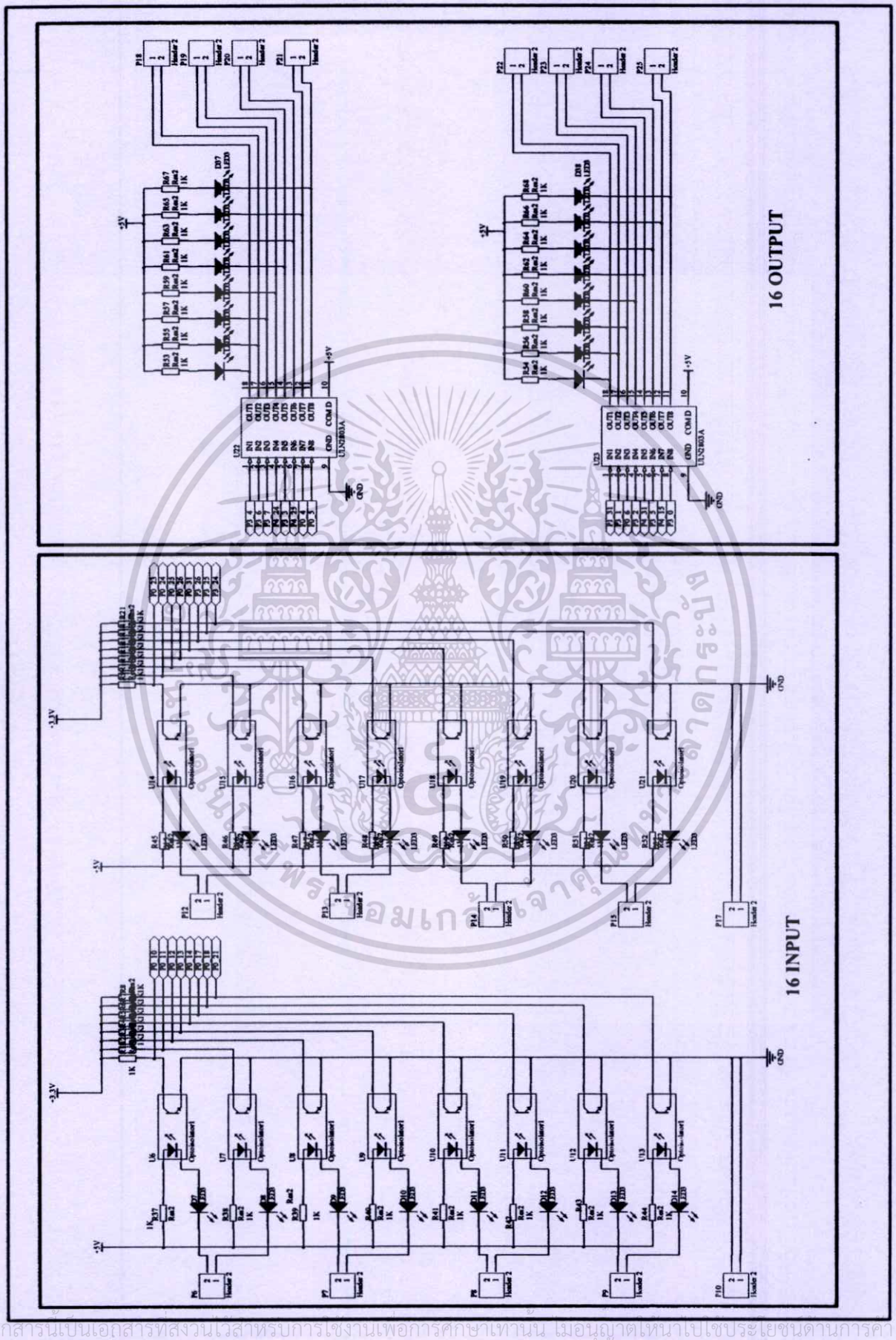
หมายเลข 32 คือ IC Power Driver NPN Open Collector เอาต์พุตเบอร์ ULN2308A

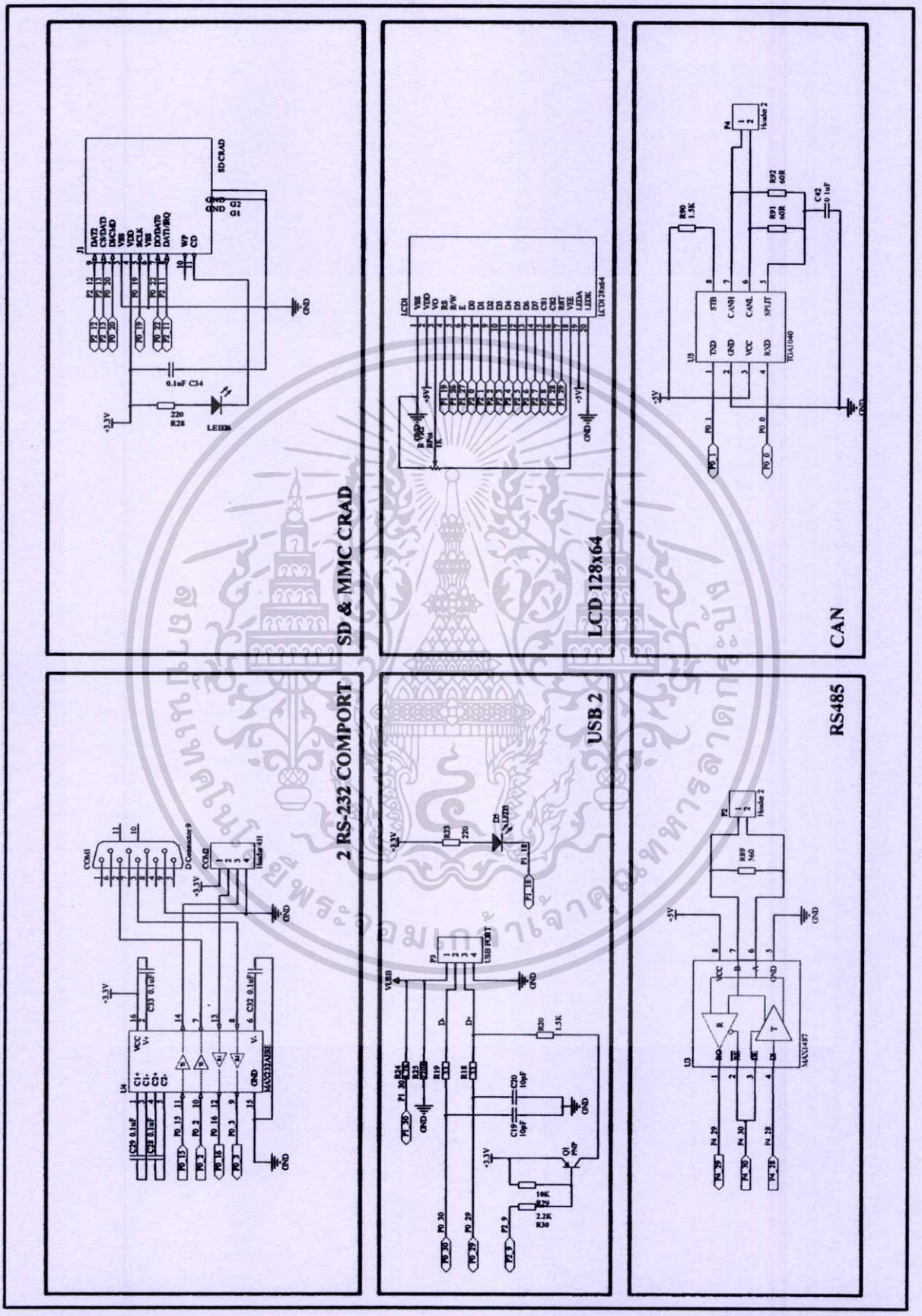
หมายเลข 33 คือ IC REGULATOR 5V 1A เบอร์ NCP1117DT50

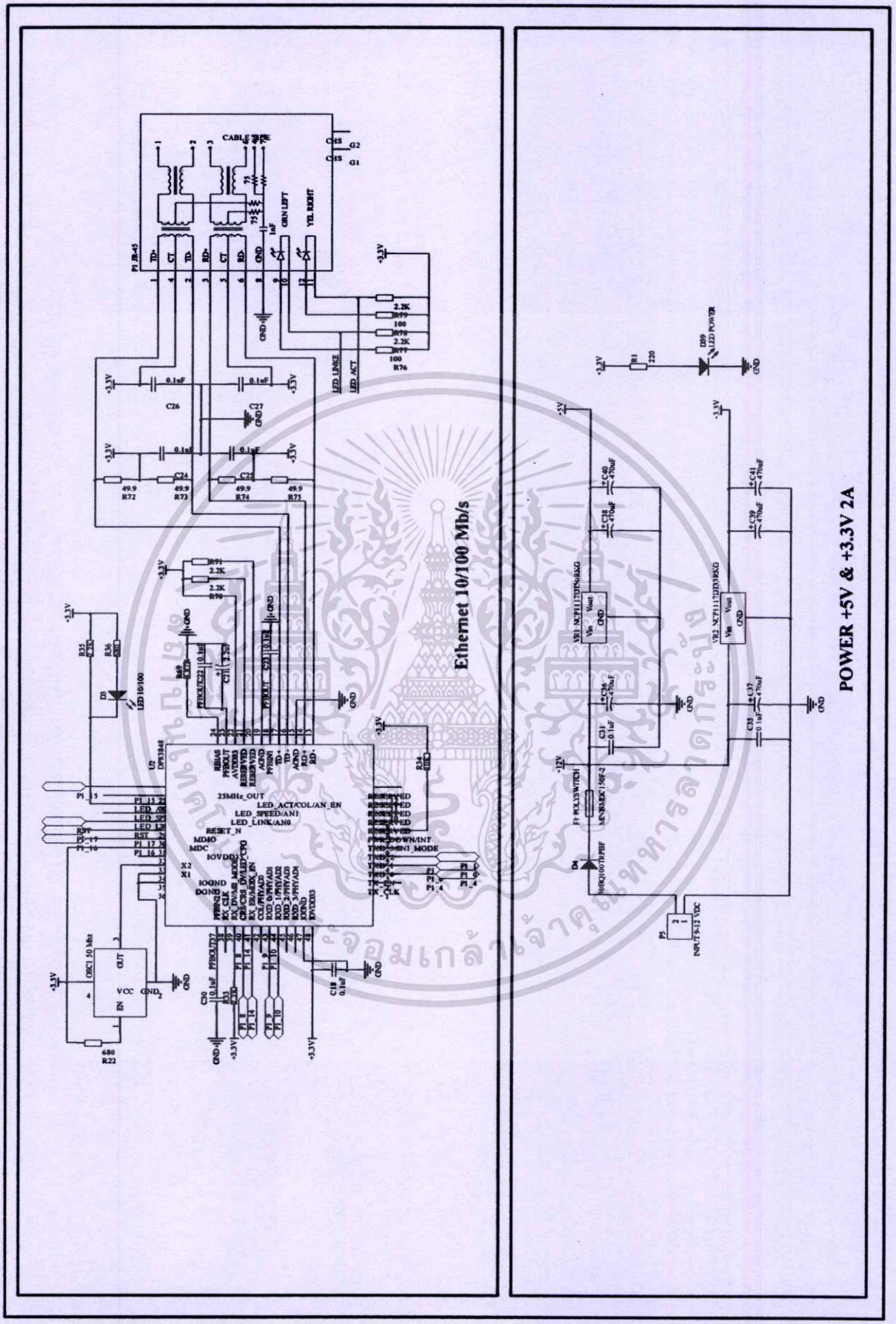
หมายเลข 34 คือ IC REGULATOR 3.3V 3A เบอร์ FAN1587AD33X

วงจรของ KMITL PLC V2

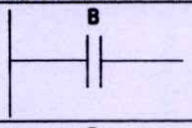
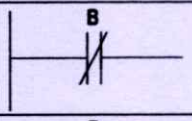
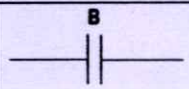
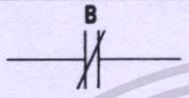
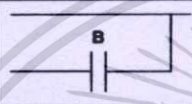
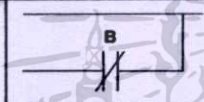
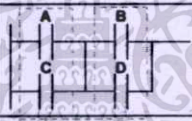
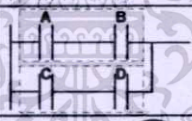
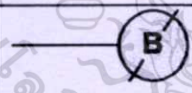
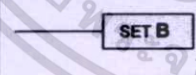
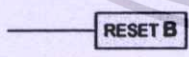
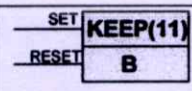
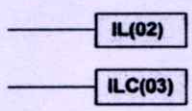
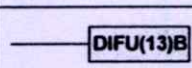








ตารางคำสั่งและการปฏิบัติงาน

ลำดับ	คำสั่ง IL	สัญญาลักษณ์ LD	พื้นที่หน่วยความจำ	ความหมาย
1	LOAD-LD		B : IN BIT IR,SR,AR,HR,TC,LR	เริ่มต้นโปรแกรมคำสั่งอ่านสภาวะปัจจุบัน B รอการกระทำทางลอจิก
2	LOAD NOT – LD NOT		B : IN BIT IR,SR,AR,HR,TC,LR	เริ่มต้นโปรแกรมคำสั่งอ่านสภาวะตรงข้ามกับปัจจุบัน B รอการกระทำทางลอจิก
3	AND		B : IN BIT IR,SR,AR,HR,TC,LR	คำสั่งที่กระทำทางลอจิก AND กับค่าปัจจุบันของ B
4	AND NOT		B : IN BIT IR,SR,AR,HR,TC,LR	คำสั่งที่กระทำทางลอจิก AND กับค่าตรงข้ามกับปัจจุบันของ B
5	OR		B : IN BIT IR,SR,AR,HR,TC,LR	คำสั่งที่กระทำทางลอจิก OR กับค่าปัจจุบันของ B
6	OR NOT		B : IN BIT IR,SR,AR,HR,TC,LR	คำสั่งที่กระทำทางลอจิก OR กับค่าตรงข้ามกับปัจจุบันของ B
7	AND LD			คำสั่งกระทำลอจิก ANDในรูปแบบของกลุ่มคำสั่ง หรือ บล็อก
8	OR LD			คำสั่งกระทำลอจิก ORในรูปแบบของกลุ่มคำสั่ง หรือ บล็อก
9	OUT		B : IN BIT IR,HR,LR	คำสั่งเขียนสภาวะปัจจุบันลงหน่วยความจำเพื่อจบใน Rung นั้นๆ
10	OUT NOT		B : IN BIT IR,HR,LR	คำสั่งเขียนสภาวะตรงข้ามปัจจุบันลงหน่วยความจำเพื่อจบใน Rung นั้นๆ
11	SET		B : IN BIT IR,HR,LR	คำสั่งเขียนเซ็ทหน่วยความจำบิท B ให้มีค่าเป็น "1"
12	RESET		B : IN BIT IR,HR,LR	คำสั่งเขียนรีเซ็ทหน่วยความจำบิท B ให้มีค่าเป็น "0"
13	KEEP		B : IN BIT IR,HR,LR	คำสั่งเซ็ทและรีเซ็ทหน่วยความจำบิท B ให้มีค่าเป็น "1" และ "0" ตามลำดับ
14	IL, ILC			คำสั่งที่ใช้งานร่วมกันเพื่อกำหนดการควบคุมหลักและยกเลิกการควบคุมหลัก
15	DIFU		B : IN BIT IR,HR,LR	คำสั่งเขียนที่ขอบขาขึ้น(Leading Edge)ลงหน่วยความจำ B

ลำดับ	คำสั่ง IL	สัญญาลักษณ์ LD	พื้นที่หน่วยความจำ	ความหมาย
16	DIFD	DIFD(14)B	B : IN BIT IR,HR,LR	คำสั่งเขียนที่ขอบขาขึ้น(Tailing Edge) ลงหน่วยความจำ B
17	END	END(01)		คำสั่งจบการทำงานโปรแกรม
18	Timer – TIM	TIM N SV	N(#): T number SV(word BCD): IR,HR,LR,#	Timer ตัวที่ N หน่วงเวลาแบบ (Delay ON) กำหนดคาบเวลาด้วย SV
19	Counter – CNT	SET RESET CNT N SV	N(#):C number SV(word BCD): IR,HR,LR,#	Counter ตัวที่ N แบบนับลง (Count down) กำหนดค่าการนับด้วย SV
20	Reverse Counter CNTR	UP DOWN RESET CNTR(12)N SV	N(#):C number SV(word BCD): IR,HR,LR,#	Reverse Counter ตัวที่ N แบบนับขึ้น-ลง (Up-Down) กำหนดค่าการนับด้วย SV
21	ASL	ASL(25) Wd	Wd: Source wd IR,HR,LR	Arithmetic Shift Left with carry การเคลื่อนข้อมูลแบบบิททางซ้ายในเวิลด์ wd
22	ASR	ASR(26) Wd	Wd: Source wd IR,HR,LR	Arithmetic Shift Right with carry การเคลื่อนข้อมูลแบบบิททางขวาในเวิลด์ wd
23	ROL	ROL(27) Wd	Wd: Source wd IR,HR,LR	Rotate Left with carry การวนข้อมูลแบบบิททางซ้ายในเวิลด์ wd
24	ROR	ROR(28) Wd	Wd: Source wd IR,HR,LR	Rotate Right with carry การวนข้อมูลแบบบิททางขวาในเวิลด์ wd
25	MOVE-MOV	MOV(21) Source wd Destination wd	S: Source wd IR,SR,AR,HR,LR,TC,DM,# D: Destination wd IR,HR,LR,DM	คำสั่งเวิลด์ทำการสำเนาข้อมูลจาก Source wd. ไปยัง Destination wd.
26	MOVE NOT-MVN	MVN(22) Source wd Destination wd	S: Source wd IR,SR,AR,HR,LR,TC,DM,# D: Destination wd IR,HR,LR,DM	คำสั่งเวิลด์ทำการสำเนาข้อมูลแบบตรงข้ามของ Source wd. ไปยัง Destination wd.
27	COMPARE CMP	CMP(20) CP1 CP2	CP1: Compare wd 1 IR,SR,AR,HR,LR,TC,DM,# CP1: Compare wd 2 IR,SR,AR,HR,LR,TC,DM,#	คำสั่งเวิลด์เปรียบเทียบข้อมูลระหว่าง CP1 และ CP2 แล้วมีผลต่อแฟลก >, < ,และ =
28	BCD to BIN BIN	BIN(23) Source wd Destination wd	S: Source wd IR,SR,AR,HR,LR,TC,DM,# D: Destination wd IR,HR,LR,DM	คำสั่งเวิลด์ทำการแปลงข้อมูลจาก Source wd.(BCD)ไปเก็บที่ Destination wd.(Hex)
29	BIN to BCD BCD	BCD(24) Source wd Destination wd	S: Source wd IR,SR,AR,HR,LR,TC,DM,# D: Destination wd IR,HR,LR,DM	คำสั่งเวิลด์ทำการแปลงข้อมูลจาก Source wd.(Hex)ไปเก็บที่ Destination wd.(BCD)
30	BCD ADD ADD	ADD(30) Au Ad R	Au: Augend wd(BCD) IR,SR,AR,HR,LR,TC,DM,# Ad: Addend wd(BCD) IR,SR,AR,HR,LR,TC,DM,# R: Result wd(BCD) IR,HR,LR,DM	คำสั่งบวกเลขฐาน 10 พร้อมตัวทด Au + Ad + Cy = CY & R

ลำดับ	คำสั่ง IL	สัญญาลักษณ์ LD	พื้นที่หน่วยความจำ	ความหมาย
31	BCD SUB SUB	SUB(31) Mi Su R	Mi: Minuend wd(BCD) IR,SR,AR,HR,LR,TC,DM,# Su: Subtrahend wd(BCD) IR,SR,AR,HR,LR,TC,DM,# R: Result wd(BCD) IR,HR,LR,DM	คำสั่งลบเลขฐาน 10 พร้อมตัวยืม Mi - Su - Cy = CY & R
32	BCD MUL MUL	MUL(32) Md Mr R	Md: Multipication wd(BCD) IR,SR,AR,HR,LR,TC,DM,# Mr: Multiplier wd(BCD) IR,SR,AR,HR,LR,TC,DM,# R: First Result wd(BCD) IR,HR,LR,DM	คำสั่งคูณเลขฐาน 10 ผลเก็บที่ R & R +1 Md * Mr = R+1 & R
33	BCD DIV DIV	DIV(33) Dd Dr R	Dd: Dividend wd(BCD) IR,SR,AR,HR,LR,TC,DM,# Dr: Divisor wd(BCD) IR,SR,AR,HR,LR,TC,DM,# R: First Result wd(BCD) IR,HR,LR,DM	คำสั่งบวกเลขฐาน 10 ผลเก็บที่ R & R+1 Dd / Dr = R+1 & R
34	RECEIVE RXD	RXD(47) D C N	D: Destination wd IR,SR,AR,HR,LR,TC,DM C: Control wd # N : Number of byte IR,SR,AR,HR,LR,TC,DM,#	คำสั่งเวิลด์อ่านข้อมูลจาก Ethernet Buffer แล้วเขียนที่ Destination wd.จำนวน N byte
35	TRANSMIT TXD	TXD(48) S C N	S: Source wd IR,SR,AR,HR,LR,TC,DM C: Control wd # N : Number of byte IR,SR,AR,HR,LR,TC,DM,#	คำสั่งเวิลด์อ่านข้อมูลจาก Source wd จำนวน N byte แล้วส่งไปที่ Ethernet Buffer

ตารางรหัสคำสั่งและเวลาในการปฏิบัติงานของคำสั่ง

ลำดับ	คำสั่ง IL	รูปแบบคำสั่ง	รหัสคำสั่ง	เวลาในการทำงาน (μsec)
1	LD	LD [bit]	C900XXXY	4
2	LD NOT	LD NOT [bit]	CA00XXXY	4
3	AND	AND [bit]	CB00XXXYH	2
4	AND NOT	AND-NOT [bit]	CC00XXXYH	2
5	OR	OR [bit]	CD00XXXYH	2
6	OR NOT	OR NOT [bit]	CE00XXXYH	2
7	AND LD	AND LD [bit]	D200XXXYH	2
8	OR LD	OR-LD [bit]	D100XXXYH	2
9	OUT	OUT [bit]	CF00XXXYH	3
10	OUT NOT	OUT-NOT [bit]	D000XXXYH	3
11	SET	SET [bit]	D300XXXYH	1
12	RESET	RESET [bit]	D400XXXYH	1
13	KEEP	KEEP [bit]	0B00XXXYH	1

ลำดับ	คำสั่ง IL	รูปแบบคำสั่ง	รหัสคำสั่ง	เวลาในการทำงาน (μsec)
14	IL,	IL [bit]	0200XXXYH	1-6
	ILC	ILC	03000000H	
15	DIFU	DIFU [bit]	0C00XXXYH	3
16	DIFD	DIFD [bit]	0D00XXXYH	3
17	END	END	01000000H	65
18	TIM	TIM [TIM Number] [Set Value]	D3RXXXXH 0000XXXXH	2
19	CNT	CNT [CNT Number] [Set Value]	D4RXXXXH 0000XXXXH	2
20	CNTR	CNTR [CNT Number] [Set Value]	1C RXXXXH 0000XXXXH	2
21	ASL	ASL [Source word]	1900XXXXH	1
22	ASR	ASR [Source word]	1A00XXXXH	1
23	ROL	ROL [Source word]	1A00XXXXH	1
24	ROR	ROR [Source word]	1C00XXXXH	1
25	MOV	MOV [Source word] [Destination]	150RXXXXH 000XXXXXH	1
26	MVN	MOV [Source word] [Destination]	150RXXXXH 000XXXXXH	1
27	CMP	CMP [First Compare] [Second Compaer]	140RXXXXH 000RXXXXH	2
28	BIN	BIN [Source word] [Result word]	170RXXXXH 000RXXXXH	1
29	BCD	BCD [Source word] [Result word]	180RXXXXH 000RXXXXH	1
30	ADD	ADD [Source word] [Add word] [Result word]	1E0RXXXXH 000RXXXXH 0000XXXXH	1

ลำดับ	คำสั่ง IL	รูปแบบคำสั่ง	รหัสคำสั่ง	เวลาในการทำงาน (μsec)
31	SUB	SUB [Source word] [Add word] [Result word]	1F0RXXXXH 000RXXXXH 0000XXXXH	1
32	MUL	MUL [Source word] [Add word] [Result word]	200RXXXXH 000RXXXXH 0000XXXXH	1
33	DIV	DIV [Source word] [Add word] [Result word]	210RXXXXH 000RXXXXH 0000XXXXH	1
34	RXD	RXD [Destination] [Node] [N byte]	800RXXXXH 000RXXXXH 0000XXXXH	2
35	TXD	TXD [Source word] [Node] [N byte]	800RXXXXH 000000XXH 000000XXH	2

ภาคผนวก ข.

ผลงานวิจัยที่ได้รับการตีพิมพ์เผยแพร่

1.สามารถ เลิศเสรี, สุพรรณ กุลพาณิชย์,"วิสวสารลาดกระบัง", ปีที่ 26 ฉบับที่ 2 เดือน มิถุนายน 2552

วิศวสารลาดกระบัง
Ladkrabang Engineering Journal
คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง กรุงเทพฯ 10520
Faculty of Engineering, King Mongkut's Institute of Technology Ladkrabang,Bangkok 10520

วันที่ 30 เมษายน 2552

เลขที่อ้างอิง 1300

เรื่อง การตอบรับบทความ

เรียน คุณสามารถ เลิศเสรี สุพรรณ กุลพาณิชย์

ตามที่ท่านได้ส่งบทความเรื่อง การประยุกต์ใช้งานสมองกลฝังตัวสำหรับการออกแบบ PLC บนเครือข่ายอีเธอร์เน็ต (Applying Embedded System to Design Programmable Logic Controller Based on Ethernet Network) มาให้พิจารณาเพื่อลงตีพิมพ์ในวารสารวิศวสารลาดกระบัง บัดนี้ ผู้ทรงคุณวุฒิได้ทำการพิจารณาแล้วเห็นว่า ยอมรับตีพิมพ์ได้ โดยจะตีพิมพ์ในปีที่ 26 ฉบับที่ 2 เดือน มิถุนายน 2552

จึงเรียนมาเพื่อทราบ

(รศ.ดร.อิสระชัย งามหรู)
หัวหน้ากองบรรณาธิการ

การประยุกต์ใช้งานสมองกลฝังตัวสำหรับการออกแบบ PLC บนเครือข่ายอีเธอร์เน็ต

Applying embedded system to design programmable logic controller based on Ethernet network

สามารถ เลิศเสรี สุพรรณ กุลพาณิชย์

สาขาวิชาวิศวกรรมการวัดคุม คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

บทคัดย่อ

บทความนี้กล่าวถึงการออกแบบและพัฒนา คอนโทรลเลอร์แบบสมองกลฝังตัว(Embedded Controller) ให้เป็นเครื่องควบคุม PLC ที่สามารถเชื่อมต่อกับโฮสคอมพิวเตอร์ผ่านระบบโครงข่ายแบบอีเธอร์เน็ต (Ethernet) การออกแบบแบ่งเป็นส่วนของฮาร์ดแวร์ และ ซอฟท์แวร์ โดยส่วนของฮาร์ดแวร์ได้เลือกเอาสมองกลฝังตัวตระกูล ARM7TDMI-S ที่ประมวลผลข้อมูลขนาด 32 บิต เบอร์ LPC2378 เป็นตัวควบคุมการทำงานหลักซึ่งประกอบด้วยหน่วยอินพุตเอาต์พุตแบบดิจิตอล ระบบสื่อสารข้อมูลที่เป็นทั้งแบบอนุกรมและขนาน หน่วยความจำภายในและภายนอก ในส่วนของซอฟท์แวร์ ได้ใช้ระบบปฏิบัติการเวลาจริง (Real time OS) ที่มีอยู่ในระบบสมองกลฝังตัวให้เป็นระบบปฏิบัติการของเครื่องควบคุม โปรแกรมตัวแปลงภาษา (Interpreter) เป็นส่วนหนึ่งของระบบปฏิบัติการที่แปลงจากไฟล์ตัวอักษร (Text File) ให้เป็นรหัสการทำงาน (Operating Code) ในรูปของไฟล์แบบไบนารี (Binary File) การพัฒนาระบบปฏิบัติการนี้อาศัยโปรแกรมตัวแปรภาษา C ตามมาตรฐาน ANSI (ANSI Standard C) การทดสอบประสิทธิภาพการทำงานของเครื่องควบคุมทำได้โดยการวัดเวลาในการสแกน (Scan Time) เมื่อมีการ Online และ Offline พอร์ทอีเธอร์เน็ต

คำหลัก: สมองกลฝังตัว, อีเธอร์เน็ต, ระบบปฏิบัติการเวลาจริง, โปรแกรมตัวแปลงภาษา, เวลาในการสแกน

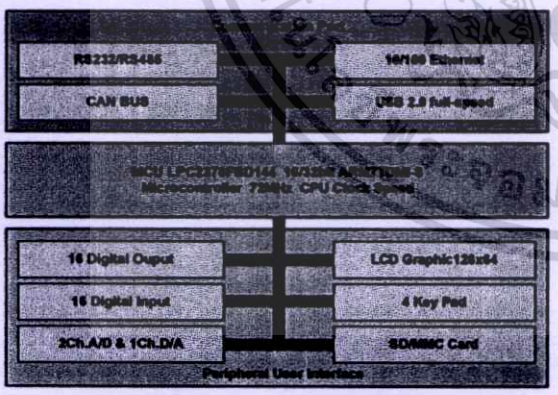
Abstract

This paper proposes the designation and development of embedded controller to programmable logic controller (PLC) linked to host computer via Ethernet network. The designation consisted of Hardware and Software parts. Hardware part, the 32 bit microcontroller (LPC2378) that is a family of ARM7TDMI-S embedded system has been applied to be the main controller comprising of digital input and output, both serial and parallel communication, internal and external memories. Software part uses the real-time Operating System (OS) based on embedded system, developed to be the OS of controller. The interpreter is a part of OS, generates code from text file to a binary file. The development uses C Complier program according to ANSI Standard. Efficiency of the controller measured by scanned time when on-line and off-line Ethernet port has been illustrated

Keywords: Embedded controller, Ethernet, Real time OS, Interpreter, Scan time

1.บทนำ

ระบบสมองกลฝังตัว(Embedded system) จะมีตัวควบคุมไมโครโปรเซสเซอร์แบบชิพเดียว(LSI : Large Scale Integrated Circuit) ที่เป็นส่วนประกอบสำคัญทางด้านฮาร์ดแวร์ การออกแบบ เริ่มตั้งแต่ การเลือกใช้หน่วยประมวลผลกลาง (Micro Processor Unit หรือ MPU) เป็นหน่วยประมวลผลที่มีความสามารถในการคำนวณและประมวลผลได้ครั้งละ 32 บิต ทำงานที่ความถี่สูงถึง 72 MHz ส่งผลให้เวลาในการปฏิบัติคำสั่ง(Execution Time) สามารถตอบสนองได้รวดเร็วและมีประสิทธิภาพมาก ลำดับต่อมา หน่วยเชื่อมต่ออุปกรณ์ภายนอก ถูกออกแบบให้รองรับส่วนที่เป็นข้อมูลทางเข้าโดยตรงไม่ว่าจะเป็นข้อมูลอินพุตเอาต์พุตแบบดิจิตอลหรืออนาล็อกที่เชื่อมต่อกับตัวตรวจจับ(Sensor)และตัวแปลงสัญญาณ (Transducer) หรือ จะเป็นอุปกรณ์ที่ใช้อำนวยความสะดวก เช่นจอภาพ LCD คีย์แพ็ต ป้อนข้อมูล หรือ หน่วยความจำ SD/MMC Card ทั้งหมดนี้เป็นหน่วยเชื่อมต่ออุปกรณ์ภายนอก หน่วยสื่อสารข้อมูลเป็นอีกหน่วยหนึ่งที่ออกแบบไว้ให้สามารถสื่อสารข้อมูลได้หลายช่องทางเช่น UART ,Can bus, Usb และ Ethernet port การออกแบบ PLC มีโครงสร้างตามรูปที่ 1

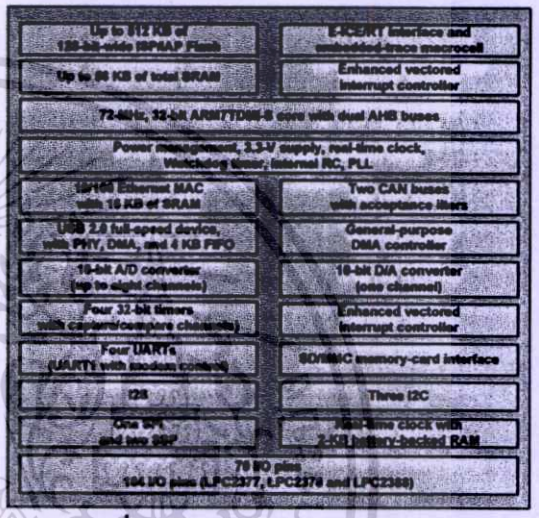


รูปที่ 1 บล็อคไดอะแกมโครงสร้างของ PLC

สิ่งต่างๆเหล่านี้มีอยู่ในระบบสมองกลฝังตัว ทำให้ความสามารถในการทำงานไม่ต่างไปจากเครื่องคอมพิวเตอร์ แต่มีขนาดเล็กกว่าหลายเท่า อีกสิ่งที่อยากจะกล่าวถึงคือความสามารถ และสถาปัตยกรรมทางด้านซอฟท์แวร์ ที่ทำงานเป็นแบบมัลติแทสค์ (Multi Tasking) สามารถบริการงานได้หลายงานพร้อมๆกัน ซึ่งเป็นระบบปฏิบัติการเวลาจริง (Real time Operating System) ดังนั้นระบบสมองกลฝังตัวจึงได้ถูกนำมาประยุกต์ใช้เป็นเครื่องควบคุมอัตโนมัติให้กับอุปกรณ์ไฟฟ้าต่างๆมากมายหลายชนิด สำหรับบทความนี้ก็เช่นกันได้นำระบบสมองกลฝังตัวมาประยุกต์ใช้เป็นเครื่องควบคุม PLC เพื่อควบคุมเครื่องจักรกลในงานอุตสาหกรรม

2 . โครงสร้างของ PLC

2.1หน่วยประมวลผลกลาง



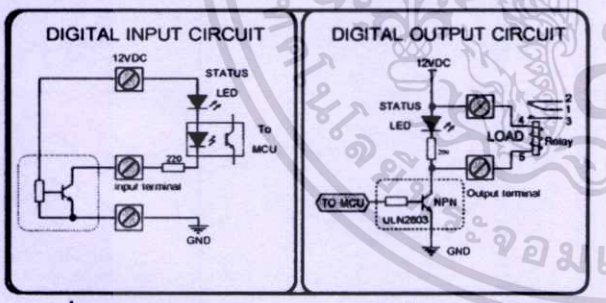
รูปที่ 2[1] โครงสร้าง MCU LPC2378

หน่วยประมวลผลกลาง (Central processing unit) หรือที่นิยมเรียกย่อ ๆ ว่า ซีพียู (CPU) เป็นหน่วยที่ทำหน้าที่เปรียบเสมือนสมองของระบบ โดยมีระบบปฏิบัติการเริ่มต้นตั้งแต่ โปรแกรมอ่านข้อมูลจาก อินพุททำหน้าที่รับข้อมูลเข้า จากนั้นนำข้อมูลที่ได้ ไปทำการ ประมวลผลคำนวณ ตามโปรแกรมประยุกต์ของผู้ใช้งานหรือโปรแกรม PLC แล้วจึงส่งข้อมูลผลลัพธ์ที่ได้ไปที่เอาท์พุท การทำงานจะวนรอบลักษณะเช่นนี้ต่อเนื่องตลอดเวลา และเวลาที่ใช้ในหนึ่งรอบการทำงานจะเรียกว่า เวลาในการสแกน (Scan Time)[2] ซึ่งจะเป็นตัวชี้วัดขีดความสามารถหรือประสิทธิภาพของ เครื่องควบคุม PLC ว่าสามารถตอบสนองต่อการเปลี่ยนแปลงสถานะของอินพุทเอาท์พุทได้เร็วเพียงใด จากข้อกำหนดต่างๆเหล่านี้ จึงได้ออกแบบเครื่องควบคุม PLC ให้ใช้ ไมโครคอนโทรลเลอร์ 32 บิต ในตระกูล ARM7TDMI-S เบอร์ LPC2378 ของ NXP มีโครง สร้างภายในตามรูปที่ 2 เป็นซีพียู ขนาด 32 บิต 144 ขา ทำงานที่ความถี่สัญญาณนาฬิกา 72 MHz

ซึ่งมีหน่วยความจำแบบแฟลชความจุ 512 กิโลไบท์ เก็บ โปรแกรมส่วนระบบปฏิบัติการ หน่วยความจำแบบสแตติกแรม 58 กิโลไบท์ใช้จัดเก็บหน่วยความจำต่างๆ PLC นอกจากนี้ยัง มีโมดูล USB, RTC, Ethernet, SD/MMC memory card, CAN controller, SRAM powered backup 2 กิโลไบท์ จากสถาปัตยกรรมที่มีอยู่อย่างครบถ้วน จึงถูกเลือกให้เป็นซีพียู ของงานวิจัยนี้ การควบคุมสามารถกระทำได้ทั้งการทำงานเป็นแบบอิสระ (Stand alone) หรือการทำงานเป็นแบบเครือข่าย (Network) ได้อย่างสอดคล้องและลงตัว

2.2 หน่วยอินพุต/เอาต์พุต

หน่วยอินพุตเอาต์พุตดิจิตอลมีขนาด 16 อินพุต/16 เอาต์พุต เนื่องจาก MCU มีขาอินพุต-เอาต์พุต อยู่จำนวนมากจึงสามารถต่อเข้าโดยตรงเพื่อให้เป็น อินพุตเอาต์พุตของเครื่องควบคุม PLC อุปกรณ์ด้านอินพุต จะทำหน้าที่รับสัญญาณจาก ภายนอก เช่น สวิตช์ และตัวตรวจจับต่างๆ ซึ่งอินพุตจะเป็น Sink input [2] ตามรูปที่ 3 สำหรับส่วนของชุดเชื่อมต่อสัญญาให้กับตัวควบคุมจะอาศัยออปโตคลัปเปิล (Opto-couple) เป็นสื่อเชื่อมสัญญาแบบแสงทำให้ไม่มีการเชื่อมต่อทางไฟฟ้าโดยตรง (Isolate)



รูปที่ 3 รูปแสดงวงจรอินพุต/วงจรเอาต์พุตแบบดิจิตอล

ด้านเอาต์พุตแบบดิจิตอล จะทำหน้าที่ ขับสัญญาณที่สั่งการมาจากหน่วยประมวลผลกลางออกไปยังอุปกรณ์ทางด้านเอาท์พุต ในที่นี้จะออกแบบให้เป็นทรานซิสเตอร์เอาท์พุตชนิด NPN แบบอาร์เรย์ Open Collector เบอร์ ULN2308 วงจรตามรูปที่ 3 โดยมี รีเลย์ เป็นภาระ ในการขับเพื่อเปิด/ปิดวงจรไฟฟ้ากระแสตรงอีกชั้นหนึ่งและเพื่อแยกกราวด์ของระบบออกจากกัน

3. การจัดวางเม็มโมรีของ PLC

3.1 หน่วยความจำแรม(Random Access Memory)

เป็นหน่วยความจำที่สามารถ อ่าน และเขียน ซ้ำได้หลายครั้ง ภายใน MCU LPC2378 มีขนาดความจุ 58 กิโลไบท์ ซึ่งมีขนาดเพียงพอที่จะจัดให้เป็นหน่วยความจำเพื่อใช้งานของหน่วยความจำ PLC ตามตารางที่ 1 มีขนาดขนาด 8 กิโลไบท์ เพื่อกำหนดให้เป็นหน่วยความจำ จำลอง (Visual Memory) สำหรับเครื่องควบคุม PLC ดังนั้นหน่วยความจำส่วนนี้จึงเรียกว่า หน่วยความจำ PLC โดยแบ่งตามชนิดใช้งาน และการทำงาน แสดงในตารางที่ 1

Area	Size	Range(word)
Input bits	160 bit	IR000 to IR009
Output bits	160 bit	IR010 to IR019
Work relay (IR Area)	2000 bit	IR106 to IR231
Holding relay (HR Area) * (Nonvolatile RAM)	320 bit	HR 00 to HR 19
Auxiliary relay (AR Area)	256 bit	AR 00 to AR 15
Link relay (LR Area)	256 bit	LR 00 to LR 15
Temporary relay (TR Area)	8 bit	TR 0 to TR 7
Special relay (SR Area)	384 bit	SR232 to SR255
Timers	256 Points	TIM 000 to 256
Counters	256 Points	CNT 000 to 256
Data memory	800 Byte	DM000-DM399
Data memory * (Nonvolatile RAM)	224 Byte	DM400-DM512

* หมายเหตุ Nonvolatile RAM เป็นหน่วยความจำที่คงค่าข้อมูลไว้ได้แม้ไฟดับ

ตารางที่ 1 แสดงหน่วยความจำ PLC[7]

3.2 หน่วยความจำ SD การ์ด (Secure Digital Memory Card)

SD การ์ด เป็นหน่วยความจำที่สามารถ อ่าน และเขียนซ้ำใหม่ได้ตลอดเวลาที่ผู้ใช้งานต้องการจึงมีความยืดหยุ่นสูงในการที่จะนำมาจัดเก็บโปรแกรมประยุกต์ของเครื่องควบคุม PLC การออกแบบจะอาศัยบัสข้อมูลแบบ SD เป็นช่องทางสื่อสารร่วมกับหน่วยประมวลผล จัดเก็บข้อมูลในรูปแบบ FAT16 ข้อมูลที่จัดเก็บส่วนแรกได้แก่โปรแกรมประยุกต์ผู้ใช้งานที่อยู่ในรูปแบบไฟล์ตัวอักษร (Text File) ผู้ใช้งานสร้างได้จากโปรแกรมอิดิเตอร์ทั่วไป เช่น โปรแกรม Notepad ที่มีอยู่ใน MS. Windows เป็นต้น และส่วนที่สองโปรแกรมโค๊ตคำสั่ง(Mechanic Code)ที่ผ่านการแปลงความหมายมาแล้วจัดเก็บในรูปแบบไฟล์ไบนารี

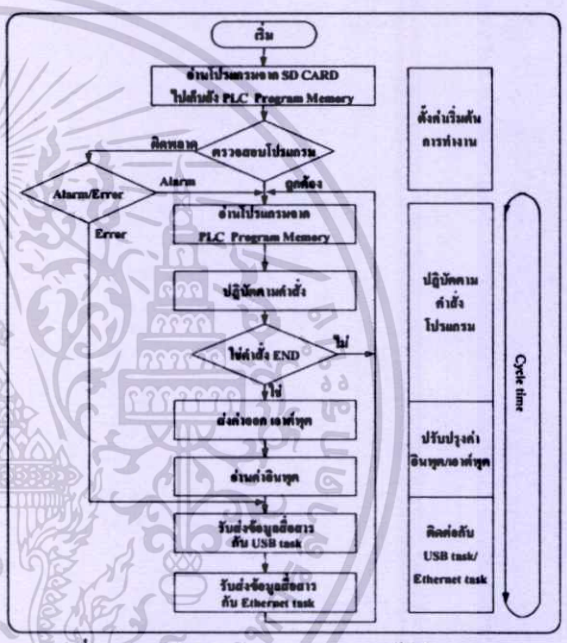
(Binary File) ที่หน่วยประมวลผลการสามารถอ่านและนำไปประมวลผลต่อได้ทันที

4.การติดต่อสื่อสารแบบ Ethernet

ในการติดต่อสื่อสารระหว่างโฮสคอมพิวเตอร์กับเครื่องควบคุมผ่านเครือข่ายแบบ Ethernet ชุดประมวลผลกลาง LPC2378 ตามรูปที่ 2 จะมี Ethernet MAC (Media Access Controller) [3] อยู่ภายในตัว นอกจากนี้ยังมีฮาร์ดแวร์ ตัวควบคุม DMA (Direct Memory Access) เพื่อการโอนถ่ายข้อมูลจากหน่วยความจำโดยตรง ทำให้ลด ภาระการทำงานของหน่วยประมวลผลในการเชื่อมต่อ Ethernet ลงได้ มีความเร็วในการสื่อสารมีระดับ Fast Ethernet 100 เมกะบิตต่อวินาที ผ่านสายสัญญาณชนิด UPT ทางพอร์ท RJ-45 ที่ติดตั้งอยู่บนแผงวงจรหลัก ไปยังระบบ Ethernet โดยมีโปรโตคอลแบบ TCP/IP และ UTP/IP เป็นพื้นฐานในการรับส่งข้อมูล และการสื่อสารรูปแบบ Embedded Web Server ที่เป็นอีกส่วนหนึ่งของระบบปฏิบัติการ ถูกพัฒนาด้วยภาษา HTML ทำให้เครื่องควบคุม PLC สื่อสารผ่านโครงข่าย Ethernet ได้อีกช่องทางหนึ่งโดยอาศัยโปรแกรมเว็บเบราเซอร์ (Web Browser) เพียงแค่กรอกตำแหน่ง IP address ลงไปก็จะสามารถ อ่าน/เขียนหน่วยความจำ PLC, Up-Down load โปรแกรมประยุกต์และควบคุม PLC ในระยะไกลผ่านโปรแกรม เว็บเบราเซอร์ ได้ทันทีทำให้สะดวกรวดเร็ว จึงเป็นจุดเด่นของงานวิจัยชิ้นนี้ทำให้ลดความยุ่งยากในการพัฒนาโปรแกรมบนคอมพิวเตอร์เพื่อควบคุมงานประยุกต์เฉพาะอย่างของเครื่องควบคุม PLC

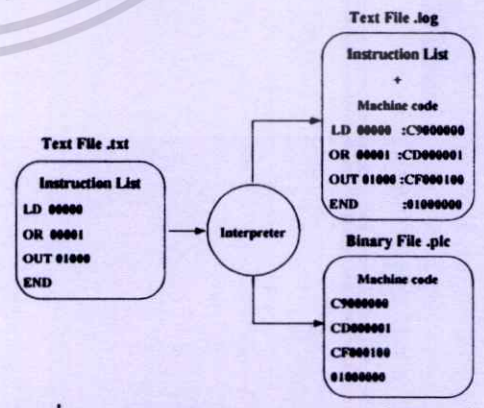
5.โปรแกรมการทำงานของ PLC

โปรแกรมระบบปฏิบัติการจะทำงานเป็นแบบมัลติแทสค์ (Multi Tasking) ที่บริการได้หลายงานพร้อมๆกัน ระบบปฏิบัติการนี้มีจะมีการตอบสนองของ หน่วยประมวลผลกลางที่มีต่อโปรแกรมผู้ใช้งานที่มีผลตอบสนองต่ออินพุทเอาท์พุท ต้องทำงานไปพร้อมกันกับการรับส่งข้อมูลผ่านทางพอร์ท Ethernet พอร์ท USB และ อ่านเขียนข้อมูลกับหน่วยความจำ SD และอื่นๆทั้งหมด เราเรียกระบบปฏิบัติการนี้ว่า ระบบปฏิบัติการเวลาจริง (Real time Operating System :RTOS) สำหรับ แทสค์(task) โปรแกรม PLC ตามแผนภูมิการทำงานดังรูปที่ 4 จะเริ่มจากโปรแกรมจะทำการอ่านไฟล์ข้อมูลแบบไบนารีไฟล์จากหน่วยความจำ SD Card มาเก็บยังหน่วยความจำโปรแกรมของ PLC จากนั้นจะอ่านโปรแกรมคำสั่งและประมวลผลตั้งแต่คำสั่งแรกจนถึงคำสั่งสุดท้ายแล้ว จึงส่งข้อมูลผลลัพธ์ที่ได้จากการประมวลผลไปยังหน่วยความจำเอาท์พุท และ สู่เอาต์พุตภายนอก จากนั้นอ่านข้อมูลอินพุตจากภายนอกเข้ามาสู่หน่วยความจำอินพุท พร้อมกับรับส่งข้อมูลให้กับหน่วยสื่อสารข้อมูล Ethernet แทสค์(task) และ USB แทสค์(task) เป็นการทำงานครบหนึ่งรอบ และจะเริ่มต้นรอบใหม่อีกครั้งในทำลักษณะเช่นนี้เรื่อยไปจนกว่าจะหยุดการทำงาน[4]



รูปที่ 4 แผนภูมิการทำงานของแทสค์โปรแกรม PLC

5.1 อินเทอร์พรีเตอร์ (Interpreter)



รูปที่ 5 แสดงการทำงานของอินเทอร์พรีเตอร์

โปรแกรมอินเทอร์พรีเตอร์ (Interpreter) เป็นส่วนสำคัญมีหน้าที่ทำการแปลงคำสั่งโปรแกรม PLC

ในรูปแบบ คำสั่ง Instruction List : IL ตามมาตรฐาน IEC 1131-3[5] โปรแกรมจะจัดเก็บข้อมูลในรูปแบบไฟล์ตัวอักษร ในหน่วยความจำ SD Card แล้วทำการแปลงโปรแกรมทั้งหมดให้ได้ผลลัพธ์จัดเก็บได้ออกเป็น 2 ไฟล์ ตามรูปที่ 5 ได้แก่ไฟล์ข้อมูลแบบไบนารี หรือข้อมูลรหัสเครื่อง (Machine Code) ที่ PLC สามารถเข้าใจแล้วนำไปใช้งานได้ทันที ส่วนไฟล์ข้อมูลแบบตัวอักษรอีกไฟล์หนึ่ง (Textfile.log) เพื่อใช้รายงานความผิดพลาดของโปรแกรมที่ผ่านการทำงานของอินเทอร์พรีเตอร์มาแล้ว

6.การทดลอง

รูปที่ 6 อุปกรณ์และการทดสอบประสิทธิภาพของ PLC

ในงานวิจัยนี้สามารถทดสอบประสิทธิภาพการทำงานได้ 2 การทดลองโดยแบ่งการทดลองที่ 6.1 ใช้เครื่องควบคุม PLC เพียงชุดเดียว ส่วนการทดลองที่ 6.2 จะใช้เครื่องควบคุมจำนวน 2 ชุด

6.1 การทดสอบประสิทธิภาพของแสกนไทม์ตามจำนวนคำสั่ง

การทดสอบประสิทธิภาพการทำงานของ PLC โปรแกรมที่ใช้ทดสอบเป็นโปรแกรมผู้ใช้งานที่มีความจุคำสั่งผู้ใช้งานได้มากถึง 16 Kstep โดยการจำลองคำสั่งให้เพิ่มจากน้อยไปหามากเพิ่มครั้งละ 0.5 Kstep มีสัดส่วนคำสั่งระดับบิต 70% และคำสั่งระดับเวิลด์ 30% สัดส่วนนี้ได้มีการวิจัยมาแล้วว่าเหมาะสมกับการทดสอบ[6] ผลการทดสอบ เริ่มต้นที่คำสั่ง 10 step, 0.5 , 1, 1.5, 2 ,....16 Kstep แล้วบันทึกเวลาการสแกนของ PLC โดยเวลาการสแกนจะอ้างอิงจาก ฐานเวลา 1 µsec ที่สร้างไว้ภายในหน่วยความจำตัวตั้งเวลาเพื่อใช้ตรวจจับเวลาในแต่ละคาบที่ซีพียูทำงานครบ 1 รอบว่านานเท่าใด จากนั้นค่อยนำผลมาแสดงยังหน่วย ความจำ DM0380 ของ PLC

จากตารางที่ 2 แสดงบางส่วนของคำสั่ง IL ระดับบิทและเวิลด์ จะเห็นว่าเวลาการปฏิบัติคำสั่งระดับบิทจะมากกว่าเป็นเพราะการพัฒนาโปรแกรมบน ARM7TDMI-S ไม่ค่อยสนับสนุนคำสั่งในระดับบิทจำเป็นต้องเอาคำสั่งเวิลด์มาประยุกต์ใช้แทน เวลาการปฏิบัติคำสั่งมีหน่วยเป็น µsec

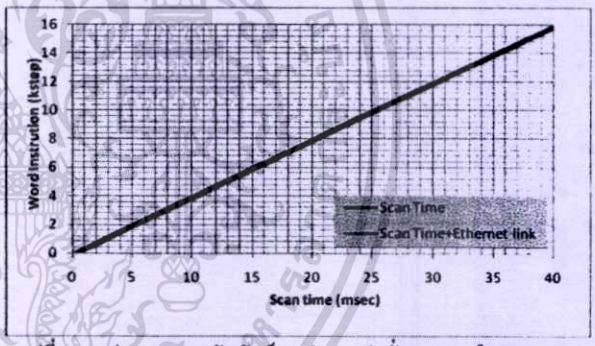
บางส่วนคำสั่งระดับบิท		บางส่วนคำสั่งระดับเวิลด์	
คำสั่ง IL	เวลาปฏิบัติ(µsec)	คำสั่ง IL	เวลาปฏิบัติ(µsec)
LD	4	CMP S,D	2
LD NOT	4	MOV S,D	2
AND	2	ADD S,A,R	1
OR	2	SUB S,S,R	1
OUT	3	MUL S,M,R	1

ตารางที่ 2

บางส่วนจำนวนคำสั่งกับเวลาการสแกนที่ Online และ Offline พอร์ท Ethernet

จำนวนสเต็ป(Kstep)	Online (msec)	Offline(msec)
0.01	0.389	0.116
0.5	1.578	1.341
4	10.328	10.091
8	20.328	20.091
12	30.328	30.091
16	40.328	40.091

ตารางที่ 3



รูปที่ 7 กราฟแสดงความสัมพันธ์ของจำนวนคำสั่งและเวลาในการสแกน

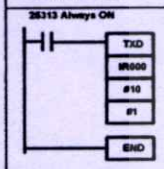
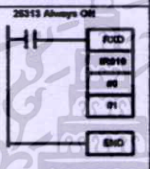
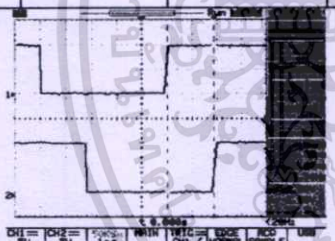
จากตารางที่ 3 เป็นผลการทดลองจำนวนโปรแกรมที่ 0.5 Kstep ใช้เวลาในการสแกน 1.578 msec และ1.341 msec และจำนวนโปรแกรมที่ 16 Kstep ใช้เวลาในการสแกน 40.328 msec และ 40.091 msec เมื่อทำการ Online และ Offline พอร์ท Ethernet ตามลำดับ ในการทดลองวัดค่าเวลาในการสแกน ขณะมีการติดต่อรับส่งข้อมูลทาง Ethernet นั้นจะทำให้เวลาเพิ่มขึ้นจากเดิมอีก 273 µSec ซึ่งนับว่าน้อยมากเมื่อเทียบกับโปรแกรมการทำงานทั้งหมดของ PLC

6.2 การทดสอบเวลาการตอบสนองต่ออินพุต/เอาต์พุต ของ PLC ผ่านพอร์ท Ethernet (I/O Response Time)

ทดสอบโดยใช้ PLC จำนวน 2 ชุดต่อผ่านพอร์ท Ethernet ตามรูปที่ 6 และเขียนโปรแกรมด้วยคำสั่ง IL ตามตารางที่ 4 โดยให้ชุดที่ 1 โหนดหมายเลข #01 เป็นชุดรับ

สัญญาณอินพุตจากสวิทช์ลงหน่วยความจำอินพุทแล้วเขียนโปรแกรมส่งข้อมูลด้วยคำสั่ง TXD ส่งข้อมูลผ่าน พอร์ท Ethernet ในทางกลับกันเขียนโปรแกรมรับข้อมูลด้วยคำสั่ง RXD ให้กับ PLC ชุดที่ 2 โหนดหมายเลข #10 เพื่อรับข้อมูลจากพอร์ท Ethernet ส่งมายังหน่วยความจำเอาท์พุทและส่งข้อมูลออกไปอุปกรณ์ภายนอก ทางฝั่งอินพุตของ PLC ชุดที่ 1 ป้อนสัญญาณพัลส์ความถี่ 100Hz หรือขนาด 10 msec

เป็นความถี่ที่เครื่องควบคุมตอบสนองได้ทัน แล้วใช้ออสซิลโลสโคป(Oscilloscope) ทำการเปรียบเทียบระหว่างสัญญานอินพุตของ PLC ชุดที่ 1 ด้วย OSC. Ch1 และเอาต์พุตของ PLC ชุดที่ 2 ด้วย OSC. Ch2 ตามลำดับ [7]

PLCชุดที่ 1 Unit #01 โปรแกรมส่งข้อมูลอินพุตผ่านพอร์ทEthernet ขนาด 1 word		PLC ชุดที่ 2 Unit #10 โปรแกรมรับข้อมูลเอาต์พุตผ่านพอร์ทEthernet ขนาด 1 word	
โปรแกรม LD	โปรแกรม IL	โปรแกรม LD	โปรแกรม IL
	LD 25313 TXD – (A) 000 (B) #10 (C) #01 END		LD 25313 RXD – (A) 010 (B) #00 (C) #01 END
			
Ch1 Input PLC1		Ch2 Output PLC2 Delay time	
10 msec.		1.80 msec.	

ตารางที่ 4 ตารางโปรแกรม PLC ทดสอบที่ 6.2

จากผลการทดลองตารางที่ 4 เมื่อเปรียบเทียบอินพุตจาก PLC ชุดที่1 และเอาต์พุตจาก PLC ชุดที่2 สัญญาณเอาต์พุตจะล้าหลังตามหลังสัญญาณอินพุตอยู่ 1.80 msec.

7. สรุป

ในบทความนี้ได้นำเสนอการใช้เทคโนโลยีด้านสมองกลฝังตัว โดยใช้ ไมโครคอนโทรลเลอร์ 32 บิต ในตระกูล ARM7TDMI-S เบอร์ LPC2378 ซึ่งมีโครงสร้างภายในเอื้อต่อการนำมาออกแบบเป็นเครื่องควบคุม PLC ได้อย่างลงตัว และเนื่องจาก PLC สามารถทำงานแปลงคำสั่ง IL ในรูปแบบไฟล์ตัวอักษร ให้ทำงานได้ทันทีจึงลดความยุ่งยากในการใช้เครื่องป้อนโปรแกรม (Programming console) และสะดวกในการใช้งานอย่างมาก ที่สำคัญยังสามารถเชื่อมต่อบนเครือข่าย Ethernet ให้ทำงานได้ทั้งโหมดการปรับปรุง แก้ไข โปรแกรม และโหมดการลิงค์สัญญาณเครื่องควบคุมเข้าด้วยกัน จากการทดลองผลที่ได้รับเป็นที่พอใจในระดับหนึ่ง และสามารถปรับปรุงให้มีประสิทธิภาพ และ มีเสถียรภาพมากขึ้นทั้งในด้านการเขียนโปรแกรมผู้ใช้งาน การสื่อสารข้อมูล และการควบคุมการทำงาน นอกจากนี้ชุดคำสั่งของ PLC ที่มีอยู่ขณะนี้มีเพียง 49 คำสั่งสามารถใช้งานได้ในระดับหนึ่งเช่นกัน อาจจะไม่ครอบคลุมการประยุกต์ใช้งานในทุกรูปแบบ ดังนั้นสามารถจะพัฒนาต่อให้มีชุดคำสั่งที่หลากหลายมากขึ้นไป อีกทั้ง ทางด้านการสื่อสารข้อมูลซอฟท์แวร์ สามารถเพิ่มเติมโปรโตคอลคำสั่ง ที่ใช้ในการสื่อสารข้อมูลทั้งในส่วนพอร์ทอนุกรม และ ส่วนของ Can bus จะทำให้เครื่องควบคุมในงานวิจัยนี้มีความสมบูรณ์มากยิ่งขึ้น

อ้างอิง

[1] www.nxp.com, UM10211, LPC2378 User manual,

[2] Warnock, G Programmable Controllers: Operation and Application,Prentice-Hall,Englewood Cliffs NJ,pp35,1998

[3] Philip J, Koopman, Jr, Beyond the Embedded Web Server ,pp 143, 1996

[4] Kim,Jomg-il,Park,J and Kwon,W H'Architecture of a ladder solving processor for Programmable Controllers, Microprocessors Microsystems. Vol. 16,pp 369-379,1992

[5] Programmable Controllers-Part 3: Programming Languages ,IEC 1131-3,pp 15,1993

[6] Gab Seon Rho ,Kyeong-hoon Koo and Naehyuck Chang, Implementation of a RISC microprocessor for programmable logic controllers, Microprocessors Microsystems. Vol. 19 , pp 599-671,1992

[7] Suphan Gulpanich, Ajin Numsomran,Prapas Roengruen,Viriya Kongratana and Kitti Tirasesth, Desing of programmable logic controllers and I/O Expansions, ICCAS2005 June 2-5, KINTEX,Gyeonggi-Do,Korea, pp1107-1111,2005

ประวัติผู้เขียน

ชื่อ–นามสกุล	นายสามารถ เลิศเสรี
วัน เดือน ปีเกิด	3 กันยายน 2525
ที่อยู่	109 ซอย.ประชานุกูล 1 ถนน.ประชานุกูล เขต บางซื่อ จังหวัด กรุงเทพมหานคร 10180
ประวัติการศึกษา	2547 อุตสาหกรรมศาสตรบัณฑิต ภาควิชาเครื่องมือวัดและควบคุม สถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ
ความชำนาญเฉพาะด้าน	1.) ระบบสมองกลฝังตัว 2.) การประยุกต์ใช้งานไมโครคอนโทรลเลอร์ 3.) ระบบปฏิบัติการเวลาจริง (RTOS) 4.) ระบบเครือข่ายคอมพิวเตอร์