การควบคุมอุปกรณ์ไฟฟ้าผ่านทางอินเตอร์เน็ต

Appliance control via internet

โดย

นายนเรศ ศรีจาด

นายวิศรุต ราชแพทยาคม

นายสันติ เขียนศิริ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมโทรคมนาคม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2543

การควบคุมอุปกรณ์ไฟฟ้าผ่านทางอินเตอร์เน็ต

appliance control via Internet

โดย นายนเรศ ศรีจาด 41013015

นายวิศรุต ราชแพทยาคม 41013031

นายสันติ เขียนศิริ 41013034

อาจารย์ที่ปรึกษา อาจารย์สุรพล บุญจันทร์

อาจารย์นภัทร สระเอี่ยม

บทคัดย่อ

โครงงานนี้เสนอ การพัฒนาระบบควบคุมอุปกรณ์ไฟฟ้าผ่านทางอินเตอร์เน็ต หลักการทำงานโดยสรุปคือ ติดตั้งเว็บเซิร์ฟเวอร์ไว้บนเครื่องคอมพิวเตอร์ที่ใช้สำหรับควบคุมอุปกรณ์ไฟฟ้า ต่อเชื่อมตัวโซลิดสเตทรีเลย์ที่สื่อสารกับอุปกรณ์ไฟฟ้าต่างๆเข้ากับคอมพอร์ตและพัฒนาแอพพลิเคชั่นด้วยไมโครซอฟต์วิชวลเบสิกซึ่งทำหน้าที่เป็นซีจีไอแอพพลิเคชั่นที่ทำงานส่งข้อมูลคำสั่งออกสู่คอมพอร์ตเพื่อควบคุมอุปกรณ์ไฟฟ้าแล้วส่งสถานะในรูปแบบเอชทีเอ็มแอลคืนสู่เว็บเซิร์ฟเวอร์ เพื่อการแสดงผลที่เว็บบราวส์เซอร์

Abstract

This project is concerning about appliance control via Internet. The conclude concept of this work is the following: (1). Installing web server onto the PC using for controlling the appliance (2). Installing relay communication device to needed COM port. (3). Development of Visual Basic application to be CGI application for sending controlling commands to the appliance, and then, transfering the statuses as the format of HTML back to the web server to present the result on web browser.

ปริญญานิพนธ์ปีการศึกษา 2543

ภาควิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง **การควบคุมอุปกรณ์ไฟฟ้าผ่านทางอินเตอร์เน็ต**

Appliance control via Internet

ผู้จัดทำ

1. นายนเรศ ศรีจาด	**41013015**
2. นายวิศรุต ราชแพทยาคม	**41013031**
3. นายสันติ เขียนศิริ	**41013034**

..... อาจารย์ที่ปรึกษา

(อาจารย์สุรพล บุญจันทร์)

..... อาจารย์ที่ปรึกษา

(อาจารย์นภัทร สระเอี่ยม)

สารบัญ

สารบัญ

สารบัญรูปภาพ

หน้า

สารบัญรูปภาพ

หน้า

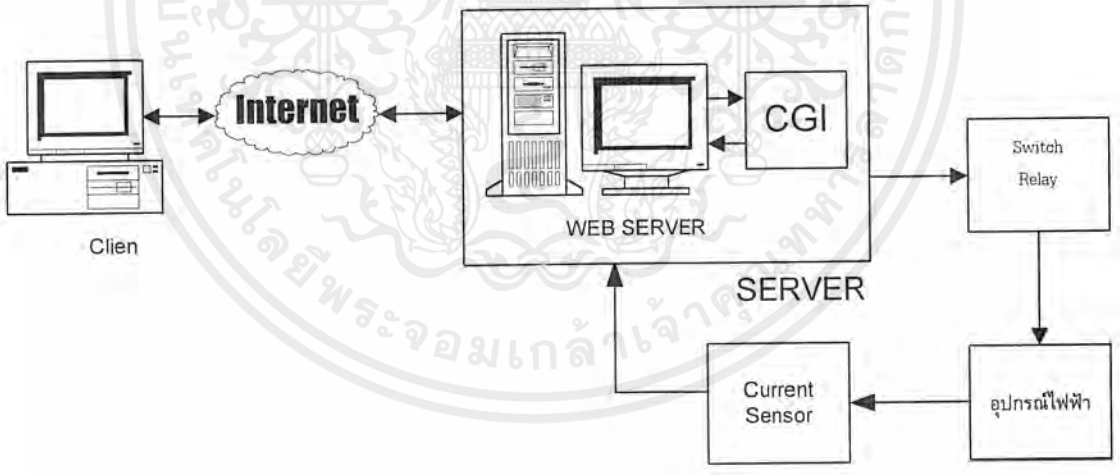
สารบัญตาราง

หน้า

บทที่ 1

บทนำ

ในปัจจุบันนี้เทคโนโลยีทางการสื่อสารได้พัฒนาทั้งรูปแบบและวิธีการไปอย่างรวดเร็วและมีความหลากหลายมากขึ้นซึ่งอินเตอร์เน็ตเองนั้นก็เป็นรูปแบบหนึ่งของการสื่อสารข้อมูลที่ได้รับความนิยมอย่างสูงในขณะนี้ ที่เห็นได้ชัดเจนคือ การควบคุมระยะไกล (Remote Control) ซึ่งเครือข่ายอินเตอร์เน็ตนั้นได้ครอบคลุมไปทั่วทุกมุมโลก ในขณะนี้ และจำนวนผู้ใช้งานก็มีอัตราเพิ่มขึ้นอย่างรวดเร็ว เราจึงใช้คุณสมบัติตรงนี้ของอินเตอร์เน็ตนำมาประยุกต์ใช้ในชีวิตประจำวันของเราโดยโครงงานนี้จะนำประโยชน์จากอินเตอร์เน็ต นำมาใช้ควบคุมอุปกรณ์ไฟฟ้าต่างๆซึ่งหลักการทำงานโดยสรุปคือติดตั้งเว็บเซิร์ฟเวอร์ไว้บนเครื่องคอมพิวเตอร์ที่ใช้สำหรับควบคุมอุปกรณ์ไฟฟ้า ต่อเชื่อมตัวรีเลย์ที่สื่อสารกับอุปกรณ์ไฟฟ้าต่างๆ เข้ากับคอมพอร์ต และพัฒนาแอพพลิเคชั่น ด้วยไมโครซอฟต์วิชวลเบสิกซึ่งทำหน้าที่เป็น ซีจีไอ แอพพลิเคชั่นที่ทำงานส่งข้อมูลคำสั่งออกสู่คอมพอร์ตเพื่อควบคุมอุปกรณ์ไฟฟ้าแล้วส่งสถานะในรูปแบบเอชทีเอ็มแอลคืนสู่เว็บเซิร์ฟเวอร์เพื่อการแสดงผลที่เว็บบราวส์เซอร์



รูปที่ 1.1 แสดงขอบข่ายของโครงงาน

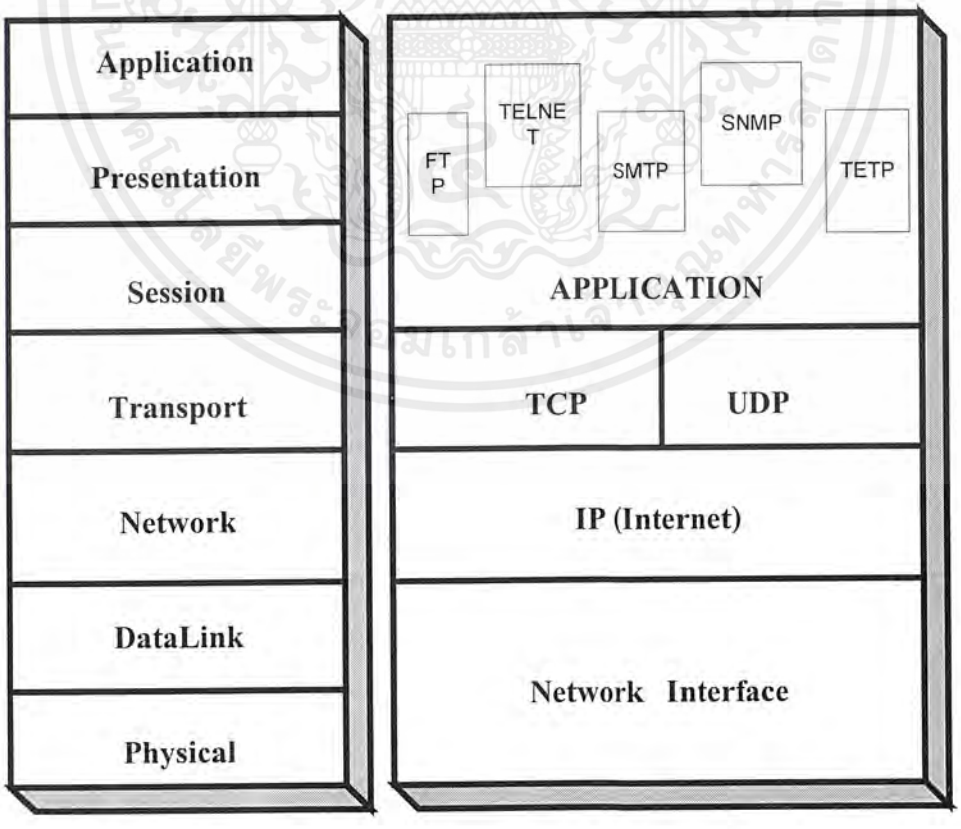
บทที่ 2

ทฤษฎีหรือหลักการ

2.1 อินเตอร์เน็ต

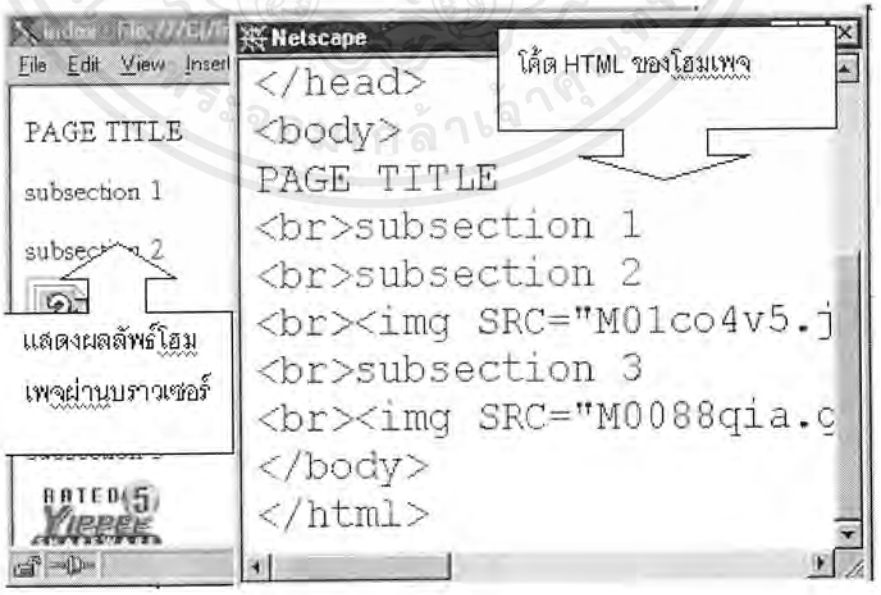
2.1.1 ความรู้ทั่วไปเกี่ยวกับอินเตอร์เน็ต

ในปัจจุบันเราคุ้นเคยกับคำว่าอินเตอร์เน็ตกันมากขึ้น เราสามารถติดต่อไปยังที่ใด ๆ ก็ได้ในโลกที่ไร้พรมแดนนี้ตามที่เราต้องการ และเช่นกัน เรายังสามารถติดต่อจากที่ใด ๆ ก็ได้ ในโลกที่เชื่อมต่อกับเครือข่ายอินเตอร์เน็ต ซึ่งแน่นอนว่าต้องมีวิธีการ หรือรายละเอียดของข้อกำหนดต่าง ๆ สำหรับการเชื่อมต่อหรือติดต่อสื่อสารของเครือข่ายอินเตอร์เน็ตด้วย ซึ่งเราเรียกวิธีการหรือข้อตกลงต่าง ๆ ของวิธีการติดต่อนี้ว่า โพรโตคอล (Protocol) โดยโพรโตคอลที่เป็นพื้นฐานสำหรับการเชื่อมโยงสื่อสารของเครือข่ายอินเตอร์เน็ตใช้ TCP/IP ย่อมาจาก "Transmission Control Protocol / Internet Protocol" โพรโตคอลนี้ถือเป็นโพรโตคอลมาตรฐาน ในการกำหนดรายละเอียดการทำงาน ทำให้สามารถเชื่อมโยงคอมพิวเตอร์ที่มีความแตกต่างกัน ให้สามารถทำงานร่วมกัน และใช้งานในระบบเครือข่ายอินเตอร์เน็ตได้



รูปที่ 2.1.1 ภาพเปรียบเทียบ TCP/IP กับ 7 เลเยอร์

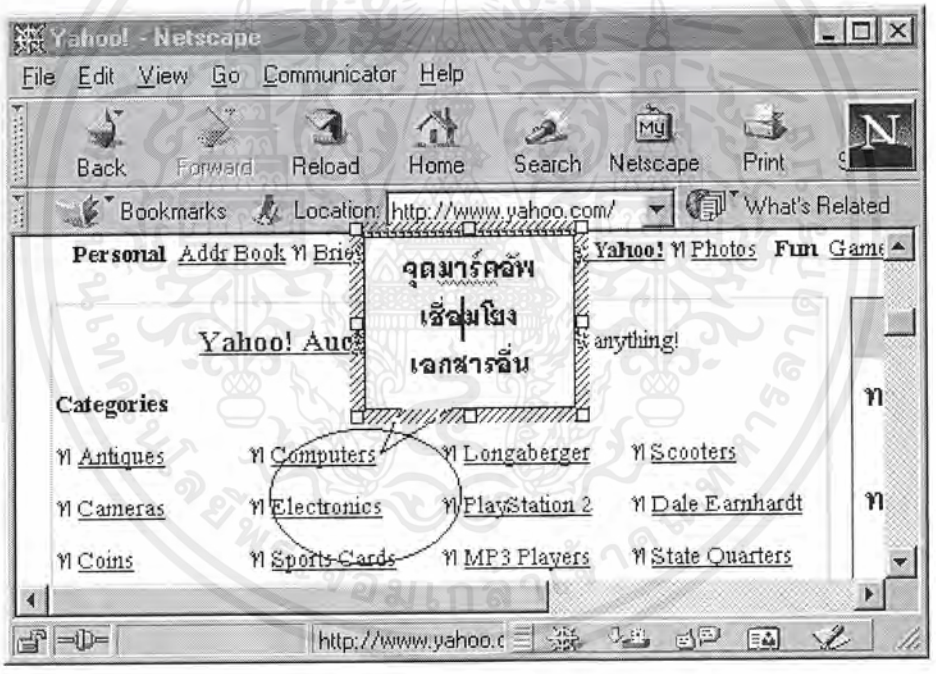
เราสามารถเปรียบเทียบการทำงาน TCP/IP กับ ตัวโอเอสไอเซเว่น-เลเยอร์ (Open Systems Interconnection 7 Layer) ซึ่งเป็นโมเดลอ้างอิงมาตรฐานสำหรับการศึกษาระบบเครือข่าย ที่ออกโดย ISO (International Standards Organization) ต่อมาเราจะมาทำความเข้าใจในรายละเอียดมากขึ้น เกี่ยวกับเครือข่ายเวิร์ลไวด์เว็บ (World Wide Web) หรือบางครั้งเรามักเรียกสั้น ๆ ว่า เว็บ เริ่มด้วยข้อมูลที่ใช้ในระบบเว็บเป็นข้อมูลในลักษณะ Interactive Hypermedia หรือกล่าวอีกอย่างว่า รูปแบบหรือเอกสารที่ใช้งานที่เรียกว่า ไฮเปอร์เท็ก (Hypertext) ซึ่งความหมายของไฮเปอร์เท็กคือ เอกสารที่สามารถ เชื่อมโยงกับเอกสารต่าง ๆ ที่มีความสัมพันธ์กันโดยภาษาที่ใช้เป็นข้อกำหนดในการสร้างเอกสารรูปแบบนี้คือภาษา HTML (Hypertext Markup Language) ภาษา HTML มีการกำหนดส่วนที่เรียกว่ามาร์คอัพ (Markup) หรือจุดที่จะเชื่อมโยง (Link) ส่วนของเอกสารต่าง ๆ ไปยังเอกสาร หรือแหล่งข้อมูลอื่น ๆ ซึ่งรายละเอียดจะได้กล่าวในหัวข้อต่อไป และวิธีการหรือรายละเอียด ข้อกำหนดในการ รับ –ส่ง ข้อมูลของระบบเว็บ จะอาศัยโพรโตคอล HTTP ซึ่งย่อมาจาก HyperText Transfer Protocol และตัวรูปแบบข้อมูลจะเรียกว่า ไฮเปอร์มีเดีย (Hypermedia) ทั้งนี้เพราะข้อมูลมีความหลากหลายในรูปแบบ การใช้งานของตัวข้อมูล ไม่ว่าจะเป็นเท็ก (Text), กราฟฟิก หรือรูปภาพ (Graphics, Images), เสียง (Audio), วีดีโอ (Video) และอื่น ๆ การแสดงผลหากเป็นการใช้งานแบบเดิมหรือในยุคแรก ๆ ของอินเตอร์เน็ตจะแสดงผลเป็น เท็กอย่างเดียว โดยใช้ Lynx ซึ่งเป็นคำสั่งใช้งานบนระบบปฏิบัติการยูนิกซ์ (UNIX) เป็นตัวค้นหาข้อมูลหรือเอกสาร (แบบเดียวกันกับบราวเซอร์) แต่ปัจจุบันสามารถใช้งานผ่านโปรแกรมจำพวก บราวเซอร์ (Browser) ที่มีขีดความสามารถสูง ทำให้สามารถใช้งาน ได้กับข้อมูล ที่มีรูปแบบที่หลากหลายได้ โดยการจัดรูปแบบการนำเสนอยังคงอาศัยภาษา HTML ในการกำหนด และสร้างตัวเอกสารตัวอย่างของบราวเซอร์โปรแกรม เช่น Netscape, IE, Opera เป็นต้น



รูปที่ 2.1.2 แสดงการทำงานของบราวเซอร์ และ HTML โค้ด

2.1.2 การสร้างโฮมเพจ ด้วยภาษา HTML

HTML เป็นส่วนหนึ่งของภาษา SGML (Standard Genneralized Markup Language) ที่มีโครงสร้างการใช้งานแบบง่าย ๆ อีกทั้งยังสามารถศึกษาและทำความเข้าใจได้ไม่ยาก โดยภาษา HTML มีไว้สำหรับใช้สร้างเอกสารแบบไฮเปอร์เท็ก ซึ่งรูปแบบของเอกสาร หรือข้อมูลส่วนต่าง ๆ ของเอกสารนั้นจะสนับสนุนทั้งที่เป็น ข้อความ, ภาพ, เสียง, วีดีโอ และอื่น ๆ โดยการใช้งานนั้นสามารถเชื่องโยงเอกสารหรือข้อมูลต่าง ๆ นี้ได้ โดยจุดที่แสดงการเชื่อมโยงนี้เองที่เราเรียกว่าการ มาร์คอัพ (มาร์คอัพ จะเป็นส่วนที่ตัวอักษรสีฟ้า หรืออื่น ๆ แล้วแต่กำหนด แต่จะเป็นจุดที่แตกต่างจากสีพื้นเอกสาร เพื่อใช้ในการเชื่อมโยงเอกสาร ผู้ใช้งานสามารถคลิ๊กจุดมาร์คอัพเพื่อกระโดดไปยังส่วนของข้อมูลในส่วนเชื่อมโยงที่ต้องการ)



รูปที่2.1.3 แสดงส่วนโฮมเพจ และมาร์คอัพ

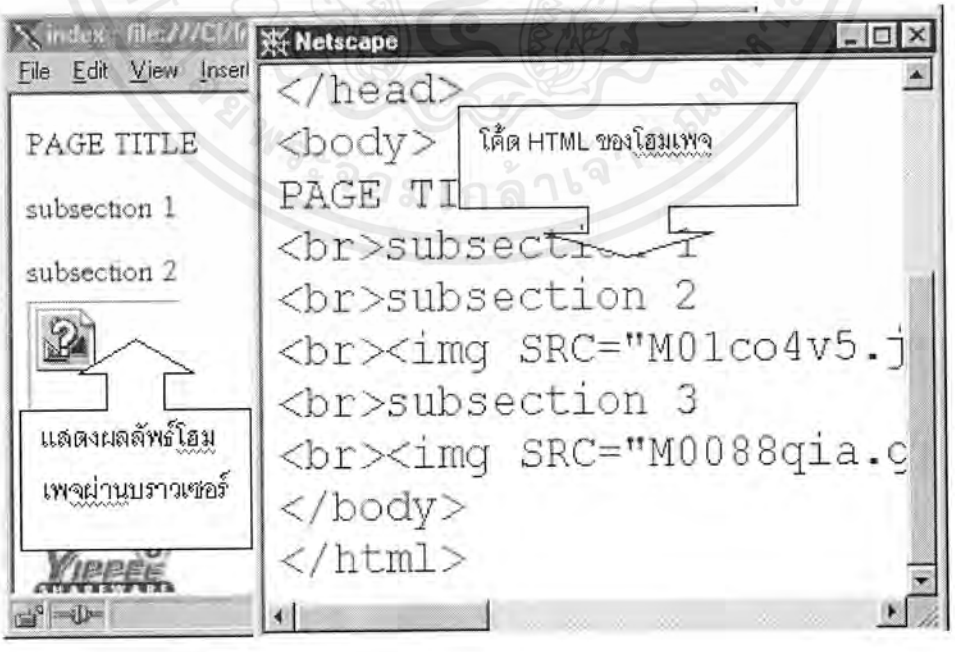
เอกสารที่สร้างด้วย HTML นี้เราเรียกว่า โฮมเพจ (Homepage) หรือเว็บเพจ (Webpage) หน้าแรกของโฮมเพจจะเป็นเสมือนอินเด็กซ์ หรือดัชนี (Index) ของรายละเอียดว่าไซต์นั้น ๆ มีรายละเอียดอะไรบ้างเกี่ยวกับเว็บเพจ ภายใต้ไซต์ นั้น ๆ ปกติแล้วโฮมเพจหรือเว็บเพจใด ๆ จะใช้นามสกุลของแฟ้มหรือไฟล์เป็น ".htm" หรือ".html" แต่ทั้งนี้ก็ไม่จำเป็นเสมอไปที่จะต้องใช้นามสกุลดังกล่าว เราสามารถปรับเปลี่ยน การติดตั้งค่าคอนฟิก (config) หรือยังมี นามสกุลแบบอื่น ๆ อีก เช่น .shtml สำหรับ SSI (Server Side Include) แต่

จุดนี้คิดว่าทำตามมาตรฐานที่มีอยู่แล้วจะดีที่สุดคือ ใช้นามสกุลของแฟ้ม สำหรับเอกสาร HTML เป็น ".htm" หรือ ".html" ชื่อแฟ้มที่ใช้กำหนดเป็นแฟ้มเริ่มต้น (ในกรณีผู้เยี่ยมชมโฮมเพจไม่ได้ระบุถึง ตัวแฟ้ม หรือ ชื่อของข้อมูล) จะเป็นชื่อ "index.html" หรือ "index.htm" เช่น URL คือ http://www.kmitl.ac.th แฟ้มที่ ถูกเลือกมาเปิดจะมีค่าเท่ากับการกำหนด URL เป็น http://www.kmitl.ac.th/index.htm หรือ http://www.kmitl.ac.th/index.html แต่ทั้งนี้ก็ยังสามารถเปลี่ยนชื่อจาก "index.html" หรือ "index.htm" เป็นอย่างอื่น ได้โดยการกำหนดค่าคอนฟิกของเว็บเซิร์ฟเวอร์

รายละเอียดพื้นฐานของภาษา HTML

-ชื่อแท็ก (Tag Name) การใช้งาน HTML จะมีคำสั่งต่าง ๆ ในการกำหนดรูปแบบการใช้งาน หรือ สร้างตัวเอกสาร ซึ่งคำสั่งต่าง ๆ นี้เรียกว่าแท็ก เช่น <HEAD>, <TITLE>, <BODY> เป็นต้นโดยการใช้งาน แท็กของ HTML นั้นมีการใช้งานสองรูปแบบใหญ่คือ

แบบแรก แท็กที่ใช้งานแล้วจำเป็นต้องมีการยกเลิก โดยจะอาศัยสัญลักษณ์ "/" ตามด้วยชื่อแท็ก ในการยุติการใช้งานแท็ก เช่น </HEAD>, </BODY> เป็นต้น จากตัวอย่างรูป 2.1.5 เป็นการทำรูปแบบตัว อักษรเข้ม โดยใช้แท็กชื่อ "<B>" ผลลัพธ์จะแสดงตัวอักษรเข้ม (Bold) ทั้งหมดตั้งแต่จุดเริ่มต้นคือ "<B>" และจุดจบ "</B>" โดยการกำหนดขอบเขตเป็นโครงสร้างเหมือน การเขียนโปรแกรม เช่น begin.....end หรือ {.....}



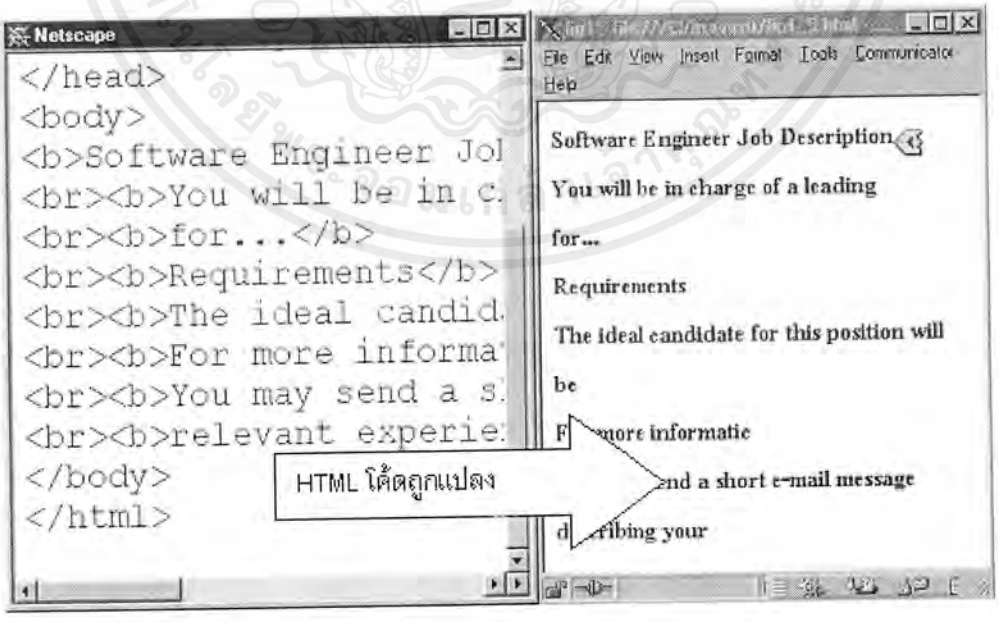
รูปที่ 2.1.4 แสดงการใช้งาน HTML แท็ก ทั้งสองแบบ

ส่วนแบบที่สอง คือการใช้งานแท็กบางตัวที่ไม่จำเป็นต้องยกเลิกการใช้งาน เช่น จากตัวอย่าง รูป 2.1.4 แสดงการใช้งาน Line Break คือจบบรรทัดให้ขึ้นบรรทัดใหม่ โดยชื่อแท็กคือ "
" จากภาพตัวอย่างจะเห็นว่าการสั่งขึ้นบรรทัดใหม่หนึ่งครั้ง ก็คือให้ทำงาน เพียงครั้งเดียว โดยไม่ต้องยกเลิกเหมือนกับแท็กในแบบแรก

- ส่วนของข้อมูล (Document Elements) คือส่วนข้อมูลที่ต้องการใช้งานของแท็ก เช่น <TITLE> A Sample HTML Document </TITLE> ซึ่งข้อความ "A Sample HTML Document" จะถือว่าเป็นค่าหรือส่วนของข้อมูล หรือเป็นส่วนของข้อมูลเอกสาร

- ส่วนถัดมาเรียกว่า "Element Attributes" ซึ่งคือส่วนที่เป็นค่าของการใช้งาน ที่ไม่ได้ถือว่าเป็นตัวเอกสารหรือข้อมูลเอกสาร แต่เป็นส่วนที่จำเป็นในการใช้งานเพื่ออ้างอิง ตัวอย่างเช่น <INPUT Type=text name="test"> หรือ <Frame src="menu.html" Name="menu"> จากตัวอย่าง อีลิเมนต์แอทริบิวคือจุดที่มีค่าเท่ากับ "test" หรือ "memu"

- รายละเอียดสุดท้ายของการใช้งาน HTML คือ การใช้งานแท็กหรือชื่อแท็กจะเป็น **"case insensitive"** *คือการใช้อักษรตัวใหญ่หรือตัวเล็กไม่มีความหมาย* ในการใช้งานแท็ก เช่น <Head> = <hEaD> = <HeaD> ถือว่ามีค่าเท่ากันคือใช้งานแท็กชื่อ <HEAD> หรือ <head> การแปลงภาษา HTML นั้นบราวเซอร์จะทำการแปลง โดยบราวเซอร์จะรับ HTML โค้ดของโฮมเพจนั้นมา แล้วทำการแปลงการทำงานที่กำหนดไว้ว่าต้องการรูปแบบอย่างไรทีลแท็ก



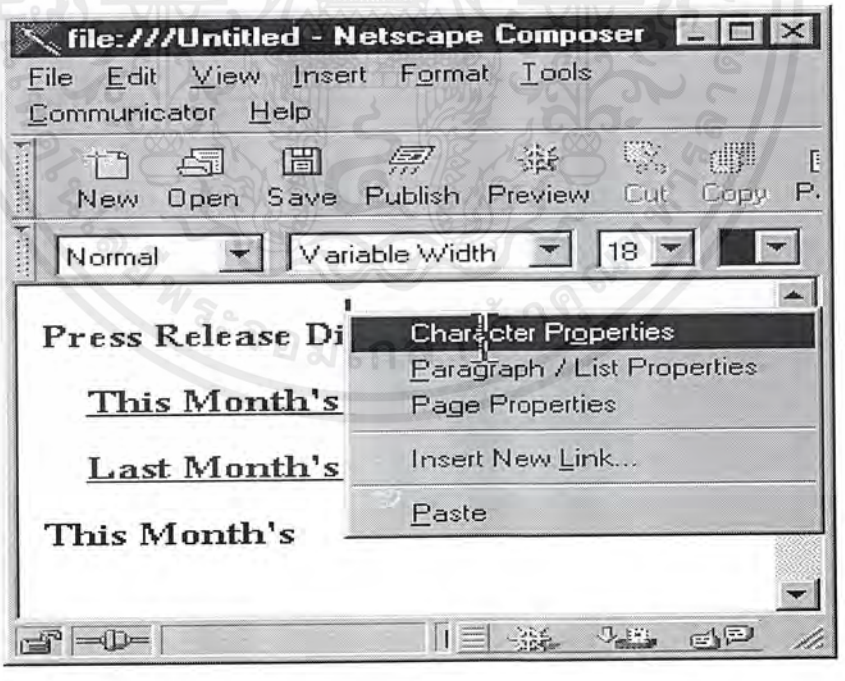
รูปที่ 2.1.5 แสดงแท็กถูกแปลงผลลัพธ์บนบราวเซอร์โปรแกรม

ต่อไปเราจะมาศึกษาเกี่ยวกับ HTML โดยมีรายละเอียดดังนี้

- การสร้างโฮมเพจ
- ส่วนสำคัญในการเขียน
- คำสั่งหรือแท็กของ HTML เบื้องต้นที่ควรทราบ

การสร้างโฮมเพจ

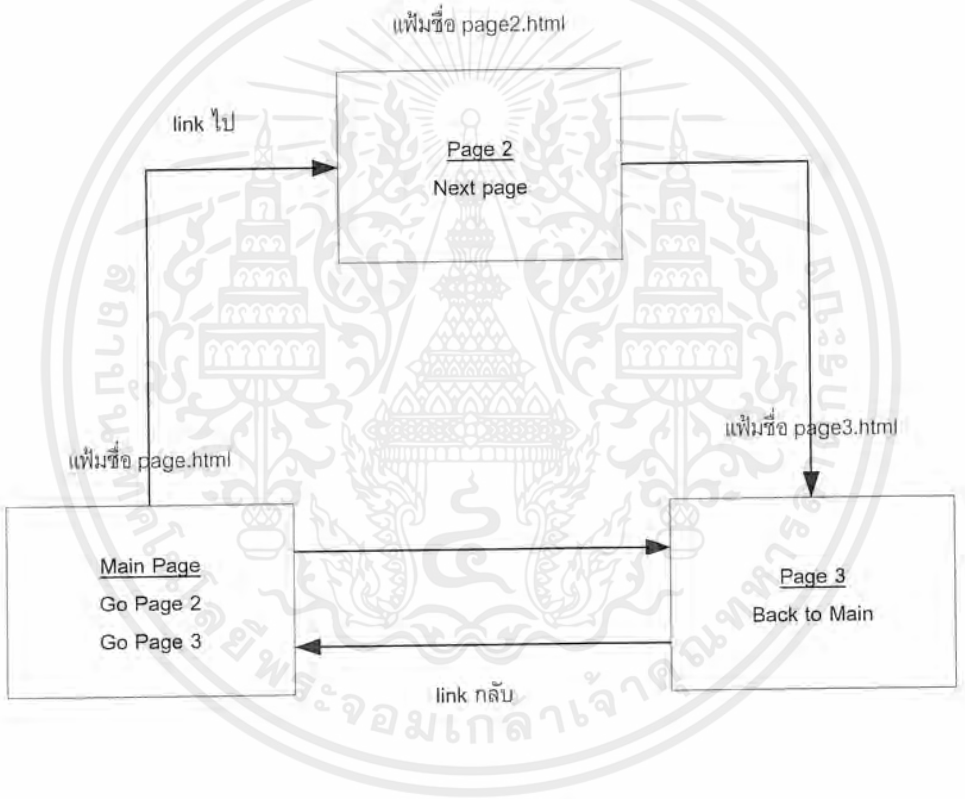
ก่อนหน้านี้เราทราบรายละเอียดพื้นฐานของภาษา HTML ไปแล้วว่ามีแท็กระบุในการกำหนดการใช้งานอย่างไร และแน่นอนว่าแฟ้ม HTML เรานั้นเป็นแท็กไฟล์ (Text File) ธรรมดา ดังนั้นเราสามารถใช้โปรแกรมประเภทอิดิเตอร์ทั่วไป เช่น โปรแกรม Edit ในดอส (Dos) หรือ Notepad ของ Windows ในการสร้างหรือปรับปรุงแฟ้มเอกสารนี้ได้ ซึ่งการยกตัวอย่างการสร้างโฮมเพจ ที่เราจะแสดงในหนังสือเล่มนี้ก็จะสร้างโดยใช้อิดิเตอร์โปรแกรมในการสร้าง เช่นกัน แต่ในความเป็นจริง แล้วเว็บมาสเตอร์ (web Master) หรือ ผู้ที่มีหน้าที่ดูและโฮมเพจ หรือเว็บเพจ มักจะอาศัยเครื่องมือในการช่วยสร้างโฮมเพจทั้งนี้เพราะจะง่าย และรวดเร็วซึ่งผู้เขียนก็ขอแนะนำให้ใช้เครื่องมือช่วยเช่นกัน แต่หากผู้เขียนโฮมเพจมีความรู้พื้นฐานของการเขียน HTML บ้างก็จะเป็นประโยชน์ต่อผู้พัฒนาแอพพลิเคชันเอง



รูปที่ 2.1.6 แสดงการใช้เครื่องมือช่วยสร้างโฮมเพจ

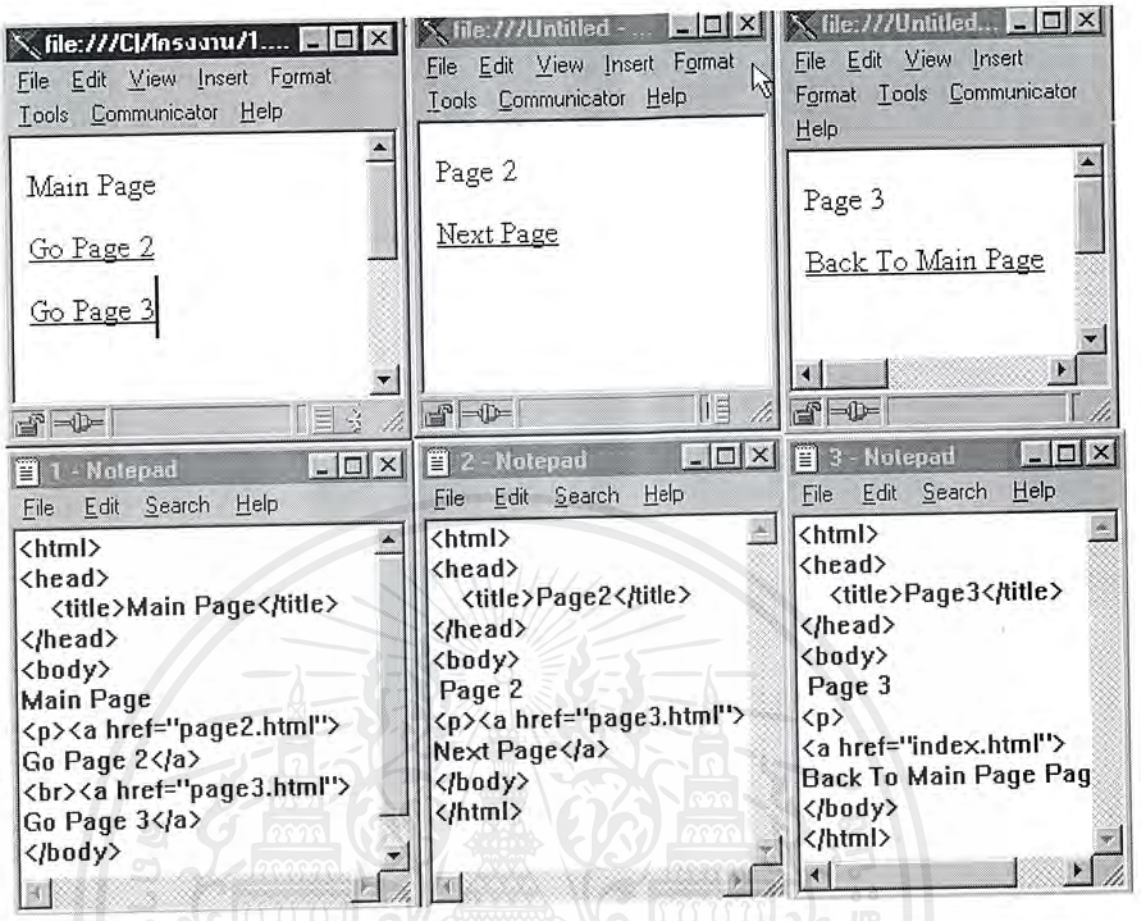
จากภาพเป็นตัวอย่างข้างต้นเป็นใช้โปรแกรม Netscape Composer ในการช่วยสร้างโฮมเพจซึ่งโปรแกรมที่ใช้หรือเครื่องมือช่วยเหล่านี้จะทำการแปลงการสร้างโฮมเพจ ที่ใช้งานลักษณะแบบ GUI (Graphics User Interface) ซึ่งจะเห็นภาพผลลัพท์ของโฮมเพจทันที และยังปรับแต่งได้ง่ายโดยไม่จำเป็นต้องทราบว่าโฮมเพจที่กำลังสร้าง หรือออกแบบประกอบด้วยแท็กใดบ้าง สำหรับประกอบการสร้าง จากนั้น เมื่อเราทำการจัดเก็บแฟ้มโฮมเพจดังกล่าวแล้ว เครื่องมือเหล่านี้จะทำการสร้าง HTML หรือ แปลงเป็น HTML โค้ด ให้อัตโนมัติ ก่อนที่เราจะเริ่มลงมือเขียนโฮมเพจนั้นสมมุติว่าเรามีรายละเอียดดังนี้

- เราออกแบบเอกสารที่ต้องการสร้าง โดยมี 3 หน้า และมีการเชื่อมโยง ดังนี้



รูปที่ 2.1.7 ภาพแสดงการออกแบบการเชื่อมโยงโฮมเพจ

มีหน้าแรกของโฮมเพจ ชื่อ index.html และมีลิงค์ (Link) ชี้ไป หน้าสอง และหน้าสาม ตามลำดับ หน้าสองจะมีลิงค์ เชื่อมโยงไปยังหน้าสาม และหน้าสามมีลิงค์ชี้กลับไปยังหน้าแรกดังภาพ



รูปที่ 2.1.8 HTML โค้ด และโฮมเพจ ที่ออกแบบไว้

- สมมุติว่าเว็บเซิร์ฟเวอร์ ได้กำหนดไดเรกทอรีเริ่มต้นสำหรับโฮมเพจ หรือพาท (Path) สำหรับใช้เก็บโฮมเพจไว้ที่ "/var/lib/httpd/htdocs"/ และไซต์ของเราชื่อ "dream.world.net"

- เรากำหนดให้แฟ้มเริ่มต้นชื่อ "index.html" เราจะเข้าเรียกดูโฮมเพจโดยกำหนด URL ดังนี้ http://dream.world.net หรือ http://dream.world.net/index.html จากภาพ 2.1.8 เป็นการแสดงโฮมเพจ 3 หน้า ประกอบด้วย หน้า Main Page, page2 และ page3 ซึ่งสามารถเชื่อมโยงถึงกันได้โดยในตอนแรกเราจะเข้าเรียกดูโฮมเพจโดยกำหนด URL เป็น http://drea,/world.net/index.htmlหรือ"http://dream.world.net" (กำหนดให้หน้าแรกของแฟ้มชื่อ index.html) ซึ่งจะเข้าไปยังโฮมเพจหน้าของ Main Page ที่แฟ้มข้อมูลชื่อ index.html ภายในแฟ้ม index (ภาพซ้ายสุด) สามารถลิงค์ไปยัง page 2 และ page 3 ได้โดยการคลิ๊กไปที่ Go Page 2 เพื่อลิงค์ไปยังหน้า 2 และ คลิ๊ก Go Page 3 เพื่อลิงค์ไปยังหน้า 3 ในทางกลับกันหน้า 3 (ภาพขวาสุด) ก็สามารถลิงค์กลับไปยัง หน้าแรก Main Page ได้ โดยคลิ๊กที่ Back To Main Page ของหน้า 3

ส่วนสำคัญของการเขียน HTML

หัวข้อก่อนหน้านี้เราทราบว่าจะเขียนหรือสร้างโฮมเพจ ได้อย่างไร และเราสามารถสร้างเอกสารด้วยภาษา HTML ได้ ซึ่งรายละเอียดของการพัฒนาโฮมเพจนั้น มีรายละเอียดอีกมากในการสร้างเทคนิคต่างๆ ประกอบ โดยจะไม่ขออธิบายในหนังสือเล่มนี้ ต่อมาเรามาดูส่วนสำคัญหรือโครงสร้างของเอกสารที่ใช้ภาษา HTML สร้างโดยโครงสร้างจะแบ่งเป็น 2 ส่วนคือ ส่วนหัว (Header) และส่วนบอดี้ (Body)

1. ส่วนหัว (Header) ของเอกสารจะเป็นดังนี้

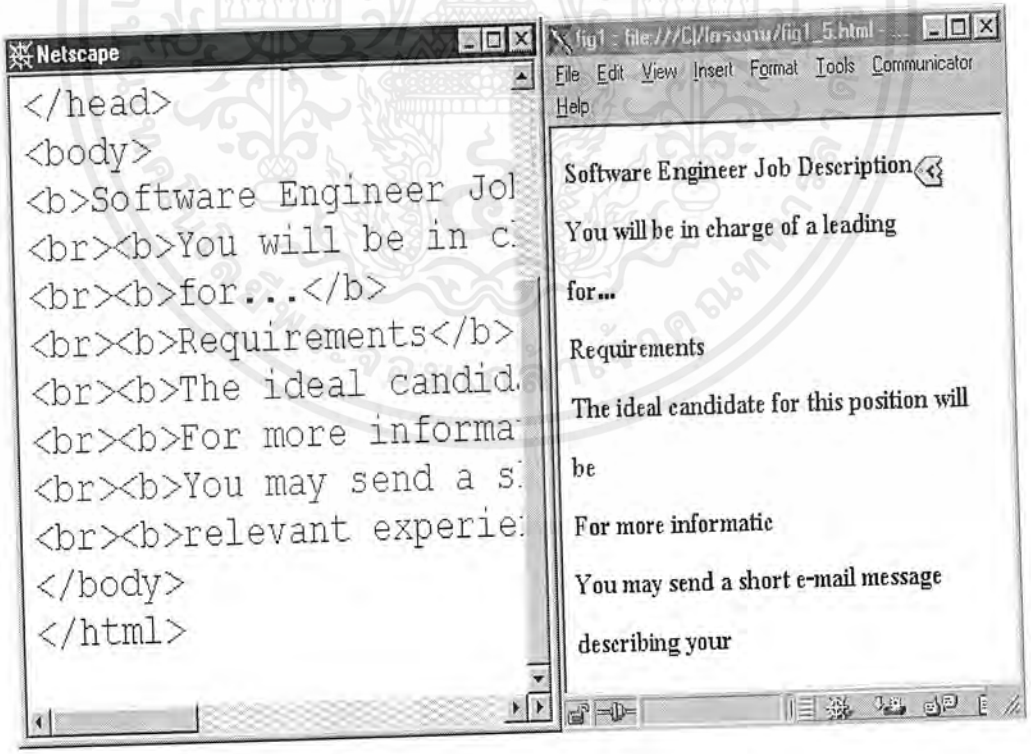
<HTML>

<HEAD>

<TITLE>....</TITLE> <! ------- ส่วนนี้แสดงบนวินโดว์บาร์ด้านบนสุดของบราวเซอร์ -------- >

</HEAD>

หมายเหตุ <!------ คือการคอมเมนท์ (Comment) โดยส่วนนี้ถือว่าไม่มีการแสดงผลออกบราวเซอร์



รูปที่ 2.1.9 แสดงส่วนเฮดเดอร์ และบอดี้เปรียบเทียบกับ HTML โค้ด

2. ส่วนบอดี้ (Body) ของเอกสารจะเป็นดังนี้

<BODY>

: :

: :

</BODY>

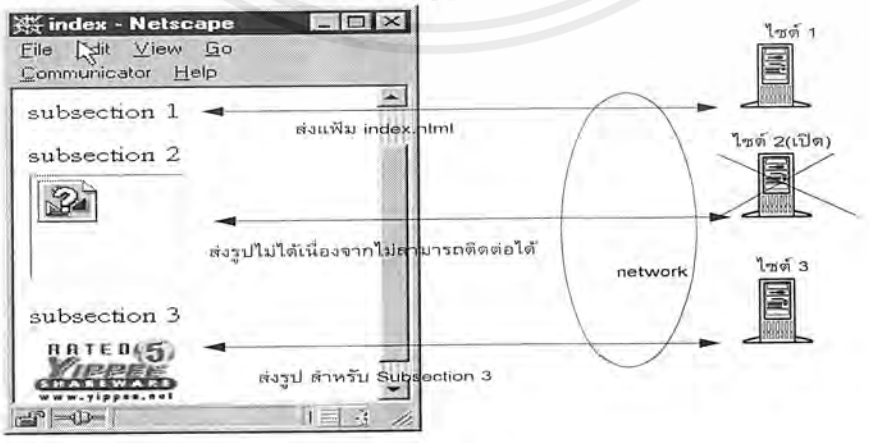
</HTML>

ในส่วนของบอดี้ จะเริ่มตั้งแต่แท็ก <BODY> จนกระทั้งถึง </BODY> รายละเอียดที่อยู่ภายใต้แท็กนี้จะเป็นส่วนของรายละเอียดที่ปรากฏเป็นโฮมเพจ

2.1.3 ความรู้ทั่วไปเกี่ยวกับระบบเครือข่ายเวิร์ลไวด์เว็บ

HTTP (HyperText Transfer Protocol)

ในตอนต้นบทเราได้กล่าวถึงคร่าว ๆ ว่า เครือข่ายเวิร์ลไวด์เว็บ จะเชื่อมโยงกัน โดยใช้ HTTP โพรโตคอล ซึ่ง HTTP โพรโตคอลนี้ เป็นโพรโตคอลเกี่ยวกับการจัดการเครือข่าย ที่ใช้ในการติดต่อสื่อสาร และรับส่งข้อมูล ภายใต้ระบบเว็บซึ่งคือ ไฮเปอร์เท็ก หรือเว็บเพจ นั่นเอง โดยรูปแบบจะเป็นแบบ "Connection Oriented" และการทำงานพื้นฐานจะมีรูปแบบเป็นลักษณะแบบ "Transaction Oriented" คือ จะอาศัยหลักการง่าย ๆ ของไคลเอ็นต์-เซิร์ฟเวอร์ (Client-Server) ในการร้องขอบริการ ซึ่งข้อมูลต่าง ๆ ที่เป็นส่วนประกอบของโฮมเพจที่ร้องขอบริการ เช่น ภาพ, เสียง หรืออื่น ๆ จะมีการเปิดการติดต่อใหม่เป็นอิสระแก่กัน ซึ่งจะอธิบายจากตัวอย่างภาพต่อไปนี้ เพื่อให้ง่ายในการทำความเข้าใจ



รูปที่ 2.1.10 แสดงการเปิดการติดต่อย่อยในการร้องขอโฮมเพจ

จากภาพ เป็นการแสดงถึง โฮมเพจ ที่มีการติดต่อขอข้อมูลจากไซต์อื่น ๆ 3 ไซต์ โดยที่ไคลเอ็นต์ (หรือบราวเซอร์) ทำการร้องขอ (request) ข้อมูลจากไซต์ที่แตกต่างกัน 3 ไซต์ ซึ่งตัวข้อมูลที่ ไคลเอ็นต์ (บราวเซอร์) ร้องขอข้อมูลในแต่ละไซต์และแต่ละงานนั้นจะมีอิสระแก่กัน เช่น

การร้องขอจากไซต์ 1 ไคลเอ็นต์ เปิดการติดต่อกับไซต์ที่ 1 (ติดต่อผ่านเว็บเซิร์ฟเวอร์) เพื่อทำการร้องขอข้อมูลที่ต้องการ ซึ่งเซิร์ฟเวอร์มี เว็บเซิร์ฟเวอร์ HTTPD (Daemon โปรแกรม) คอยให้บริการที่พอร์ท 80 แล้วเซิร์ฟเวอร์ ก็จะส่งข้อมูลกับ (reply) ตามที่ไคลเอ็นต์ขอมาหลังจากนั้นจึงปิดการติดต่อ

การร้องขอบริการจาก ไซต์ 2 ไคลเอ็นต์ ทำการร้องขอไปยังไซต์ 2 แต่ว่า ไซต์ 2 ปิดการให้บริการอยู่ จึงไม่สามารถติดต่อได้ หลังจากนั้นจึงปิดการติดต่อ

การร้องขอบริการจากไซต์ 3 ไคลเอ็นต์ ทำการร้องขอไปยัง ไซต์ 3 (ซึ่งมีการทำงาน เหมือนกับไซต์ 1 แต่แตกต่างกันตรงตัวข้อมูลที่ส่งกลับ) ไซต์ 3 สามารถให้บริการได้ เซิรฟ์เวอร์ก็จะส่งข้อมูลกลับมายังไคลเอ็นต์ ตามที่ไคลเอ็นต์ ตามที่ไคลเอ็นต์ได้ร้องขอ

สรุป จากการร้องขอข้อมูลของไคลเอ็นต์ ไปยังไซต์ 1, 2 และ 3 จะเห็นว่าข้อมูลที่ส่งกลับมาจากแต่ละไซต์ ไม่ขึ้นแก่กัน ทั้ง 3 ไซต์ ไซต์ 1 และ 3 เว็บเซิร์ฟเวอร์มีการส่งข้อมูลกลับมายังไคลเอ็นต์ได้ แต่ไซต์ 2 ไม่มีข้อมูลกลับมา ซึ่งหลักการที่สำคัญในการทำงานเป็นเรื่องของ การร้องขอของไคลเอ็นต์ และการตอบกลับของเซิร์ฟเวอร์ นั้น ใช้กลักการของ ไคลเอ็นต์ / เซิร์ฟเวอร์ทั้งนี้เพราะ HTTP ก็เป็นโพรโตคอล ที่ทำงานแบบ ไคลเอ็นต์ / เซิร์ฟเวอร์ และด้วยเทคนิคกับวิธีที่กล่าวข้างต้นจะเห็นว่ามีข้อดีคือทำให้งานแต่ละชิ้นเป็นอิสระต่อกันดังนั้น หากมีส่วนใดเสียหรือมีปัญหาในการติดต่อสื่อสารไม่ว่าจะด้วยสาเหตุใดก็ตามจะไม่กระทบแก่กัน เมื่อครู่เรากล่าวถึงการทำงานแบบไคลเอ็นต์ / เซิร์ฟเวอร์ ต่อมาเราจะมาทำความเข้าใจเพิ่มเติมเกี่ยวกับหลักการทำงานของไคลเอ็นต์ / เซิร์ฟเวอร์ซึ่งการทำงานมีสามส่วนหลักๆ คือ

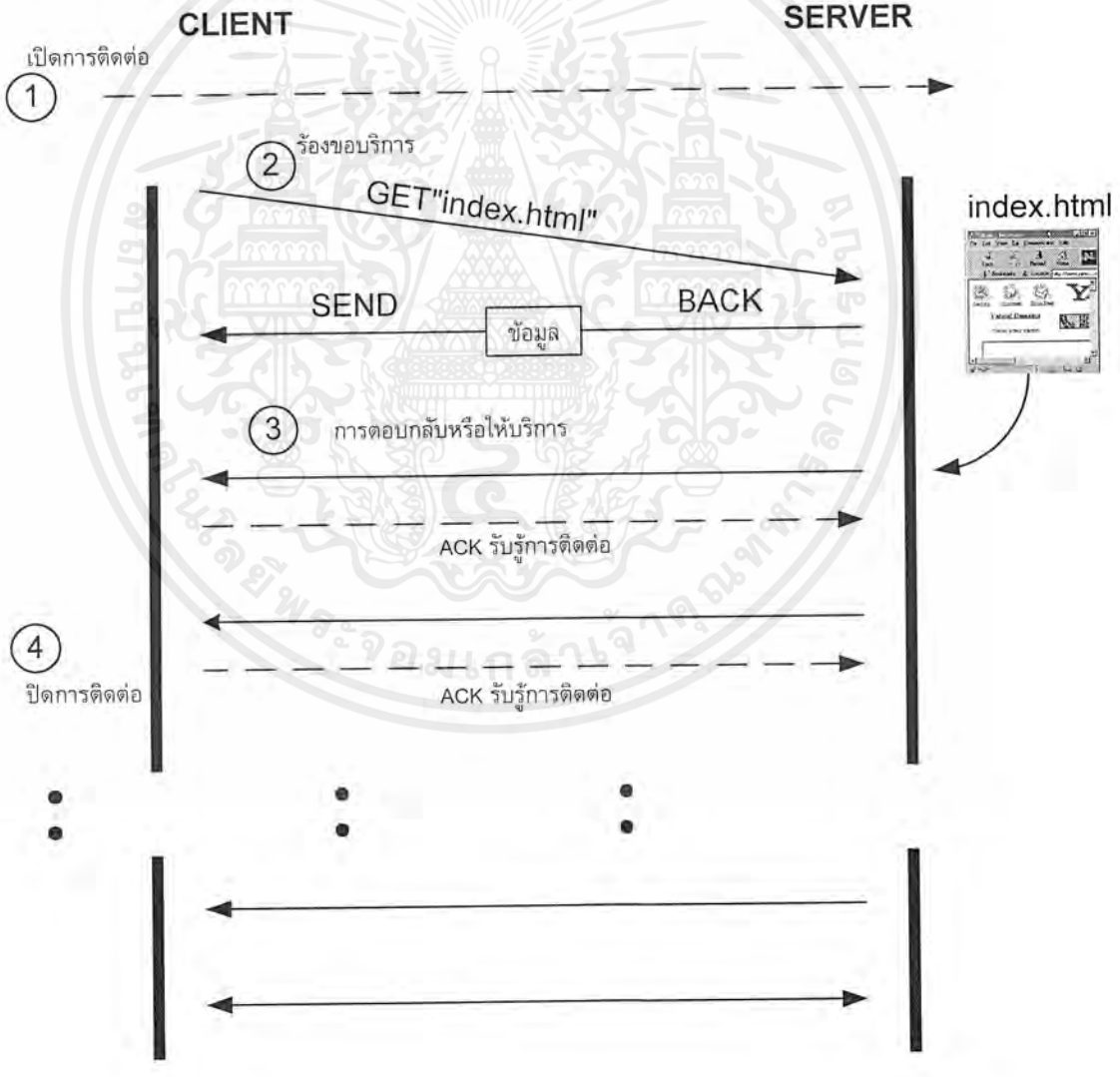
1. ไคลเอ็นต์ (Client) เป็นผู้ขอใช้บริการ
2. ระบบเครือข่าย (Network)
3. เซิร์ฟเวอร์ (Server) เป็นผู้ให้บริการแก่ไคลเอ็นต์

โดยมีขั้นตอนการทำงานดังนี้คือ

1. การสร้างการติดต่อระหว่าง ไคลเอ็นต์ และ เซิร์ฟเวอร์ (Establish Connection)

2. การร้องขอบริการ (Request) คือ ไคลเอนต์ ร้องขอบริการไปยังเซิร์ฟเวอร์
3. การตอบกลับหรือการให้บริการ (Reply) คือ การที่เซิร์ฟเวอร์ ให้บริการแล้วแต่ชนิดการร้องขอบริการของ ไคลเอ็นต์
4. การปิดการติดต่อ (Terminate) คือ การยกเลิกการทำงานกรณีที่การร้องขอ บริการ และการให้บริการเสร็จสิ้นสมบูรณ์

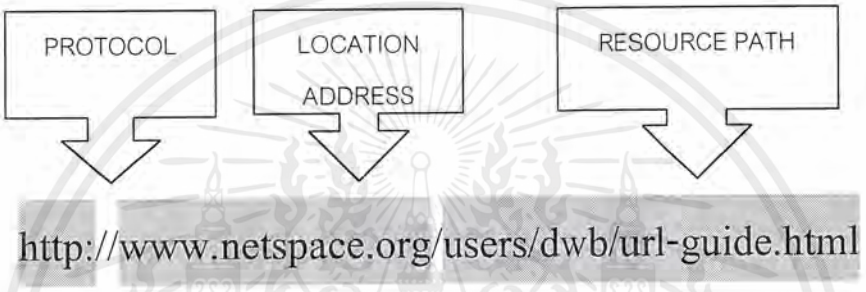
ในหัวข้อ 2 และ 3 บางครั้งจะมองรวมเป็นส่วนเดียวกันได้ คือการโต้ตอบ หรือ การรับ – ส่งข้อมูล หรือ การทำอินเตอร์แอกชัน (Interaction) แล้วแต่จะเรียก



รูปที่ 2.1.11 แสดงการทำงานของไคลเอ็นต์ / เซิร์ฟเวอร์

URL (Uniform Resource Locators)

เรามักได้ยินว่าการใช้งานอินเตอร์เน็ตต่าง ๆ จะสามารถใช้งานได้ภายใต้พื้นฐาน ของระบบเว็บ หรือที่นิยมเรียกว่า "Web-besed" ทั้งนี้ เพราะบราวเซอร์ โปรแกรม อาศัยหลักการของ URL ในการสร้างความหลากหลายของการใช้งานหรือขอบริการ ในรูปแบบต่าง ๆ ซึ่ง URL ก็คือการระบุถึงแหล่งข้อมูลที่ต้องการขอบริการ โดยมี โครงสร้างคร่าว ๆ ดังนี้

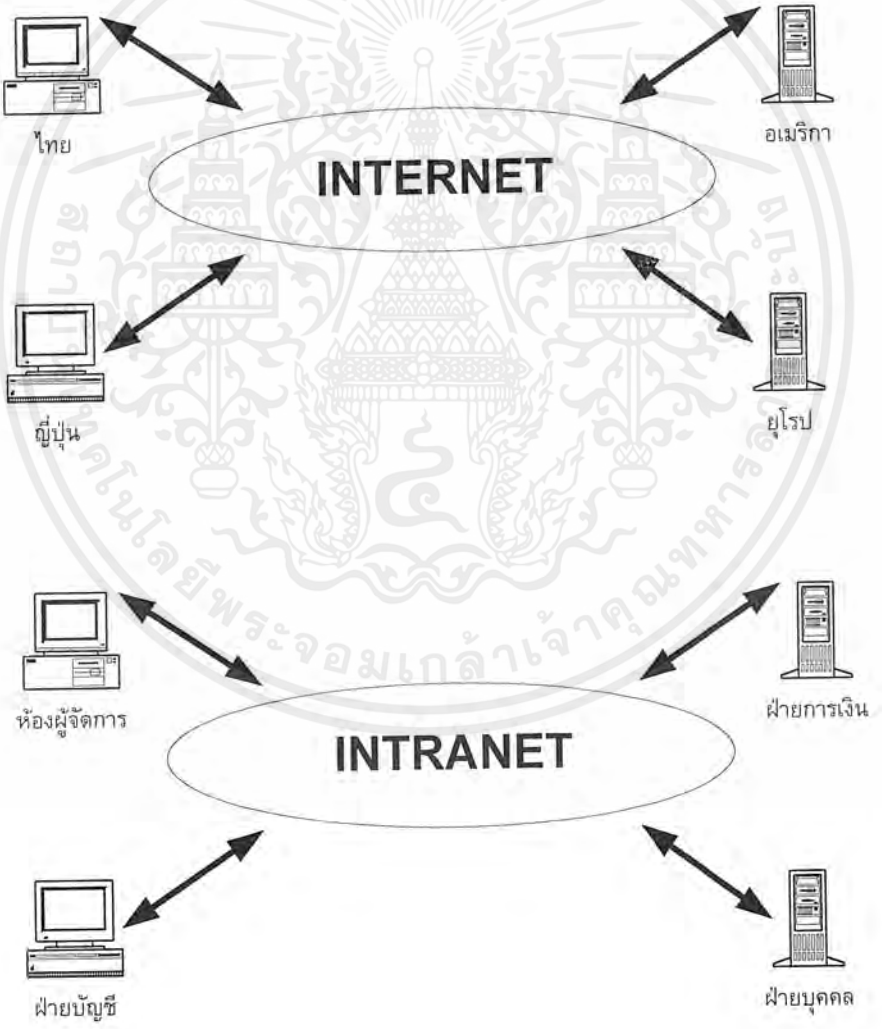


และจากโครงสร้างของ URL ดังกล่าวทำให้เราสามารถติดต่อตามไซต์ต่าง ๆ โดยระบุถึงที่อยู่ หรือ Location Address และกำหนดถึงพาท (Path) ที่อยู่ของข้อมูลที่ต้องการได้ และยังสามารถใช้ตัวอย่างการกำหนด URL ขอใช้บริการ อื่น ๆ นอกเหนือจาก HTTP FTP (File Transfer Protocol) ftp://chaokhun.kmitl.ac.th/pub E-Mail mailto:s8626072@kmitl.ac.th

2.1.4 CGI กับการพัฒนาแอพพลิเคชันใน Internet, Intranet และ Extranet

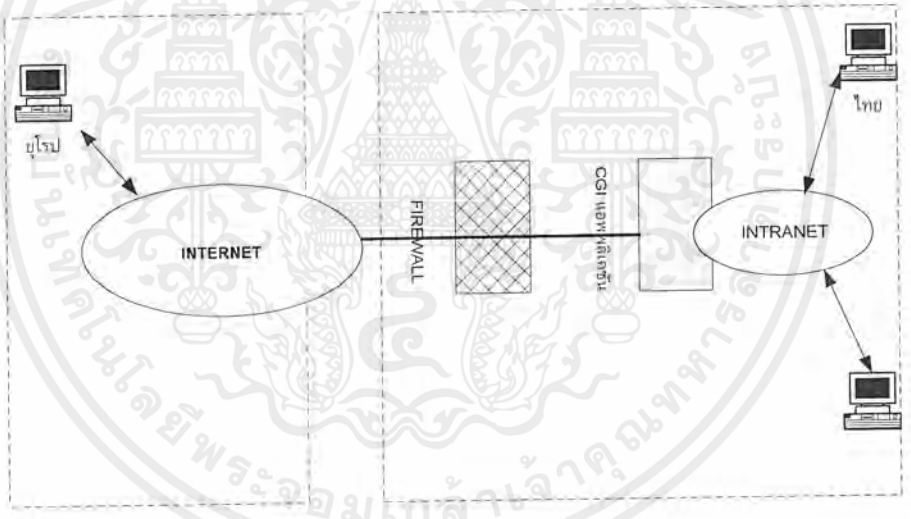
สำหรับวิธีการของ CGI ที่เรานำมาพัฒนาแอพพลิเคชันนั้น เราทราบอยู่แล้วว่าเป็น แอพพลิเคชันที่ใช้งานภายใต้ระบบเครือข่ายเวิร์ลไวด์เว็บ และแน่นอนว่าเครือข่ายเว็บเป็นส่วนหนึ่งของการอาศัยโครงสร้างของเครือข่ายอินเตอร์เน็ตเป็นพื้นฐานในการใช้งาน แต่ไม่ใช่ว่าทุก แอพพลิเคชันที่ประยุกต์ใช้งานบนเครือข่ายอินเตอร์เน็ต จะเป็นการทำงานภายใต้ระบบเว็บ เพียงแต่ว่าระบบเว็บจะได้รับความนิยมมากกว่าเพราะความสามารถของตัวบราวเซอร์โปรแกรมและอีกหลาย ๆเหตุผลที่สนับสนุน ดังนั้นไซต์ต่าง ๆ จึงพยายามพัฒนา CGI ในการติดตั้งกับแอพพลิเคชันอื่น ๆ ให้สามารถ ใช้งานผ่านระบบเว็บได้ และจุดนี้เองคือจุดเริ่มต้นสำหรับที่มาของ CGI แต่ในปัจจุบันวิธีการนี้ถูกนำไปพัฒนาแอพพลิเคชันอื่น ๆ จนแอพพลิเคชันหลาย ๆ แบบ ไม่ว่าจะเป็นระบบการจัดการงานขององค์กร, การบริหารองค์กร และอื่น ๆ ต่างก็

ถูกพัฒนาบนระบบเว็บไปเสียทุกแขนงจนเกิดการใช้งานในรูปแบบต่าง ๆ มากมายดังที่เห็นในปัจจุบัน ไม่ว่าจะเป็น การทำการประชุมทางไกล, การทำกระดานข่าว, การทำระบบจัดการเอกสาร, การติดต่อสื่อสารในรูปแบบต่าง ๆ เป็นต้น ทั้งนี้ก็เพราะ อาศัยข้อดีของระบบเครือข่ายในการช่วย เพิ่มประสิทธิภาพและเพื่อลดค่าใช้จ่ายนั่นเองและด้วยเหตุผลที่เห็นภาพได้ชัดเจนจากเครือข่ายอินเตอร์เน็ตนี่เองที่ทำให้หลายๆ องค์กร พยายามนำมาประยุกต์ใช้งานในองค์กรตนเองบ้างจึงเกิดระบบอินทราเน็ต(Intranet) ซึ่งระบบอินทราเน็ตนี้เกิดขึ้นโดยองค์กรเหล่านั้น นำระบบเครือข่ายอินเตอร์เน็ตมาประยุกต์เพื่อใช้งาน และพัฒนาแอพพลิเคชันใช้งานจนแอพพลิเคชันแบบเดิมหลายๆอย่างเริ่มโยกย้ายเข้าสู่การใช้งานในระบบอินทราเน็ตนี้ด้วย บวกกับยังมีแอพพลิเคชันเดิมที่เป็นพื้นฐานเดิมอยู่แล้วของอินเตอร์เน็ตอีกเช่น ระบบอีเมล์,ระบบเว็บบอร์ดหรือกระดานข่าวสาร เป็นต้น ทำให้อินทราเน็ตยิ่งเป็นทางออก สำหรับ การจัดองค์กรด้านไอทีที่เปลี่ยนแปลงเร็วอีกทั้งอินทราเน็ตมีทิศทางและแนวโน้มที่ชัดเจน และดีกว่าในทุกด้าน จากแนวทางเดิม



รูปที่ 2.1.12 ภาพแสดงรูปแบบเครือข่ายอินเตอร์เน็ตและอินทราเน็ต

และเมื่อเครือข่ายอินทราเน็ตเริ่มถูกใช้งานอย่างจริงจัง ข้อมูลและแอพพลิเคชันต่าง ๆ เริ่มมีมากขึ้นหลาย ๆ องค์กรก็เริ่มพยายามทำให้การใช้งานสามารถใช้งานผ่าน เครือข่ายอินเตอร์เน็ตหรือต้องการนำข้อมูลในอินทราเน็ตออกสู่อินเตอร์เน็ตบ้าง เพื่อประโยชน์หลาย ๆ อย่างเช่น ลดค่าใช้จ่าย (ในกรณีที่ต้องการใช้งานอินทราเน็ตจากที่ไหนก็ได้ในอินเตอร์เน็ต), หรือต้องการขยายองค์กรสู่ระดับสากล เป็นต้น ปัญหาด้านเครือข่ายคงไม่มีแน่ทั้งนี้เนื่องมาจากว่า ทั้งสองระบบอาศัย หลักการพื้นฐานด้านเทคนิคที่เหมือนกัน แต่ปัญหาด้านแอพพลิเคชัน คงหนีไม่พ้น ไม่ว่าจะเป็นเรื่องขอบเขตการใช้งาน, ระบบรักษาความปลอดภัยต่าง ๆ , สิทธิการใช้งาน, การจัดการตัวฐานข้อมูล โดยเฉพาะเรื่องของตัวข้อมูล แน่นอนว่าระบบอินทราเน็ตนั้นย่อมมีข้อมูลที่มากมายบางส่วนก็สำคัญยิ่งยวด ไม่สามารถเปิดเผยสู่ สาธารณะชนได้ หากจะกล่าวในมุมมองของโปรแกรมเมอร์ก็คือ บางตาราง (Table) มีบางฟิลด์ (Field) ที่ไม่สามารถนำออกสู่ภายนอกดังนั้นจึงเกิดการจัดระบบเอ็นทราเน็ต (Extranet) เกิดขึ้น

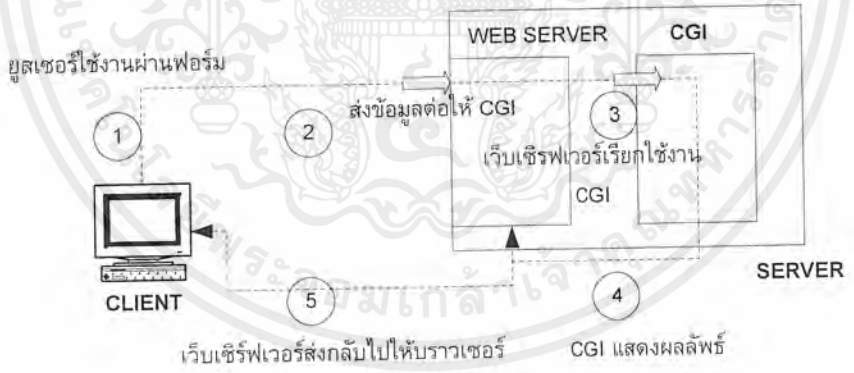


รูปที่ 2.1.13 ภาพแสดงการใช้ CGI ควบคุมงานข้อมูลที่เหมาะสม

เราสามารถพัฒนา CGI แอพพลิเคชันในการจัดการปัญหาดังกล่าวข้างต้นได้ หรือ อาจจะนำ CGI แอพพลิเคชันเดิมในอินทราเน็ตจัดการเรื่องขอบเขต การใช้งาน หรือ การจัดการว่าจะนำข้อมูลมากน้อยแค่ไหนสู่ภายนอก หรือสู่อินเตอร์เน็ต เช่น เราอาจจะทำ CGI โปรแกรมตรวจสอบ IP address หรือตรวจสอบว่าผู้ใช้กำลังใช้งานจากที่ใด ภายในองค์กร หรือจากภายนอก แล้วจัดการนำเสนอรูปแบบการใช้งานที่เหมาะสมให้ และเช่นกัน เราก็ยังสามารถใช้ CGI โปรแกรม ในการจัดการเกี่ยวกับข้อมูลที่ต้องการเก็บจากภายนอก ว่าต้องการมากน้อยแค่ไหนสำหรับองค์กร

2.1.5 Common Gateway Interface (CGI)

เมื่อเกิดระบบเครือข่ายเวิร์ลไวด์เว็บใช้งานจนเป็นที่นิยมดังเช่นปัจจุบัน หลาย ๆ ไซต์ (Web Site) เริ่มต้องการนำเสนอข้อมูลภายในองค์กร ที่เคยใช้งานกับโปรแกรมประยุกต์ของตนภายในองค์กรผ่านเว็บเพจ หรือ โฮมเพจ (HomePage) ของตน จึงเกิดปัญหาว่าจะสามารถทำอย่างไร ทั้งนี้เพราะทั้งสองแอพพลิเคชั่น อยู่คนละส่วนกัน และวิธีการทำงานก็แตกต่างกันอย่างสิ้นเชิง ทางออกคือการพัฒนาแอพพลิเคชัน ในลักษณะเหมือนโปรแกรมประยุกต์ที่องค์กรใช้งานอยู่ โดยอาศัยหลักการของ CGI ในการพัฒนา แต่นี่ยังเป็นเพียงแค่จุดเริ่มต้น ของความต้องการเท่านั้น เพราะปัจจุบันเราจะเห็นว่า มีแอพพลิเคชันหลากหลายรูปแบบบนระบบเว็บ เช่น การให้บริการส่งเพจ (Pager), การให้บริการค้นหางาน, การให้บริการความช่วยเหลือแบบออนไลน์ เป็นต้น ซึ่งแอพพลิเคชันเหล่านี้เกิดจากความต้องการที่หลากหลาย และต่างความคิด รวมไปถึงวิสัยทัศน์ของแต่ละคนในการที่คิดประยุกต์ และร่วมสร้างกิจกรรมต่าง ๆ บนระบบเว็บ จนทำให้การใช้งานระบบเว็บนี้กลายเป็นส่วนสำคัญหลักของเครือข่ายอินเตอร์เน็ตในปัจจุบันไปเสียแล้วและด้วยความสามารถของหลักการ CGI นี้เองทำให้หลาย ๆ องค์กรต้องการนำมาประยุกต์ใช้ในองค์กรจนเกิดคำที่ว่า "แอพพลิเคชันในอนาคต คือ แอพพลิเคชันที่ใช้งานผ่านบราวเซอร์ หรือใช้งานภายใต้พื้นฐานเว็บ (Web –based หรือเรียกว่า Web – based Application)"



รูปที่ 2.1.14 บราวเซอร์ติดต่อ CGI และรับผลลัพธ์จาก CGI ผ่านเว็บเซิร์ฟเวอร์

จากภาพ 2.1.14 เราจะมาให้ความหมายว่า อะไรคือCGI จริง ๆ แล้ว CGI ก็คือหลักการหรือวิธีการของการพัฒนาแอพพลิเคชัน ที่ทำหน้าที่เสมือนประตู (Gateway) เชื่อมโยงการติดต่อกับการทำงานอื่น ๆ เพื่อให้เกิดการทำงานที่หลากหลายในการใช้งาน โดยอาศัยพื้นฐานของระบบเว็บหรือจะกล่าวได้ว่าทำงานควบคู่กับเว็บเซิร์ฟเวอร์ เพราะบราวเซอร์ไม่สามารถติดต่อส่วนอื่น ๆ โดยตรงได้ เช่น จะติด

ต่อกับฐานข้อมูล เป็นต้น จำเป็นต้องติดต่อผ่านเว็บเซิร์ฟเวอร์ ไปยังส่วนของ CGI โดยเรามักเรียกว่า "CGI โปรแกรม" หรือ "CGI แอพพลิเคชัน" หรือ "เว็บแอพพลิเคชัน" ก็ได้ ด้วยเหตุนี้เองเราจึงเห็นว่า จริง ๆ แล้ว CGI แอพพลิเคชัน หรือ แอพพลิเคชัน ที่พัฒนาตามแนวทาง CGI เป็นแอพพลิเคชัน ประเภท เซิร์ฟเวอร์ แอพพลิเคชัน (Server Application) หรือแอพพลิเคชันที่ทำงานอยู่ที่ฝั่งเซิร์ฟเวอร์ โดยมี ส่วนที่ทำหน้าที่ติดต่อกับ ผู้ขอใช้บริการ หรือไคลเอ็นต์ (Client) คือเว็บเซิร์ฟเวอร์ และไคล เอ็นต์ใช้งานผ่านเว็บบราวเซอร์ ข้อดีของเซิร์ฟเวอร์แอพพลิเคชันที่เห็นได้ชัดคือ การปรับปรุงหรือ เปลี่ยนเวอร์ชันจะทำได้ง่ายโดยไม่ต้องแจกจ่ายให้ผู้ใช้งานทุกครั้งแต่สามารถดูแลปรับปรุงได้ที่ เซิร์ฟเวอร์โดยตรงพอมีวิธีการของ CGI เกิดขึ้นปัจจุบันเราจึงได้เห็นรูปแบบของโฮมเพจที่เปลี่ยนไป จากเดิมที่เคยเป็นแค่ "Static Hypermedia Document" คือเอกสารที่แสดงโดยไม่มีการเปลี่ยนแปลง ไปเป็นเอกสาร ที่สามารถเปลี่ยนแปลงรูปแบบได้ ตลอดจนเห็นเป็นโฮมเพจ ที่สามารถโต้ตอบ หรือ เป็น อินเตอร์แอกทีฟ (Interractive) เหมือนส่วนของอินเตอร์เฟส (Interface) ของ CGI แอพพลิเคชัน ที่แปรเปลี่ยนตลอดเหมือนกับการใช้งานโปรแกรมประยุกต์นั่นเอง

2.1.6 ไดนามิก โฮมเพจ (Dynamic Home Page)

เมื่อครู่ มีการพูดถึงโฮมเพจ ที่แสดงการใช้งานของCGI แอพพลิเคชัน ที่ติดต่อกับยูสเซอร์ สามารถโต้ตอบในการติดต่อทำงานได้ซึ่งส่วนนี้ก็คือ อินเตอร์เฟซ (User interface) ของ CGI โปรแกรม และรูปแบบที่มีการเปลี่ยนแปลงของอินเตอร์เฟซ ที่เป็นลักษณะเว็บเพจนี้เองเราเรียกว่า "ไดนามิกโฮมเพจหรือไดนามิกเว็บเพจ"แต่บางครั้งเอกสารแบบนี้อาจมีการเรียกว่า Virtual Document ทั้งนี้เพราะเอกสารเหล่านี้ไม่มีรูปแบบที่แน่นอนถ้าเรามองโปรแกรม ก็เหมือนว่า การ ทำงานของแต่ละโพรเซส (Process) ของโปรแกรมจะแสดงผลลัพธ์ข้อมูล แปรเปลี่ยนตามการปรับ ปรุงล่าสุดของข้อมูลหรือแล้วแต่ว่าตอนนี้อยู่ส่วนไหนของโปรแกรมต้องแสดงรายละเอียดอย่างไร ในโปรแกรม ตัวอย่างของไดนามิกโฮมเพจ ที่จะทำให้เข้าใจง่ายขึ้น ซึ่งรายละเอียด วัน-เดือน-ปี และเวลา จะถูกแสดงผ่านโฮมเพจโดยมีการเปลี่ยนแปลงรายละเอียดทุกครั้งที่มีการเข้ามาเยี่ยมชม หรือเมื่อโฮมเพจถูกเปิดดูการทำโฮมเพจแบบไดนามิกนี้มีหลายวิธีในการจัดการเช่น การใช้ CGI โปรแกรม หรือจะใช้ภาษาจาวา สคริปต์ (Java script) หรือ แอพเพล็ต (Java Applet) (ซึ่งในกรณี ของตัวอย่างนี้ เป็นจาวาแอพเพล็ต) หรือจะเป็นวิธีการของ SSI ก็ได้ (Server Side Includes) นอกจาก นี้ยังมีภาษาที่ยังไม่เป็นมาตรฐาน อีกหลายตัว ตัวอย่างการประยุกต์ใช้งาน เช่นการใช้ CGI ในการ แลกเปลี่ยนลิงค์ (Link Exchange) โดยมีวิธีการคือ เมื่อมีการแลกเปลี่ยน จะมีการระบุโลโก หรือ พื้น ที่รูปภาพในการโฆษณา เมื่อมีการเข้าเยี่ยมชมโฮมเพจ จะมีการแสดงโฆษณาซึ่งจะเปลี่ยนไปเรื่อย ๆ ตามลำดับการโฆษณา ประกอบภายในโฮมเพจหน้านั้น ๆ ทำให้รูปภาพในการโฆษณาจะแปรเปลี่ยน ไปเรื่อย ๆ หรืออาจจะเป็น โฆษณาตามไซต์ต่าง ๆ ที่มักจะเปลี่ยนไปเรื่อย ๆ หรือจะแสดงโฆษณาใน

หัวข้อตามที่ผู้ใช้งานต้องการตามไซต์ที่ให้บริการค้นหาข้อมูลมักทำกันคือเมื่อเราค้นหารายละเอียดเกี่ยวกับอะไรจะปรากฏข้อมูลโฆษณาในแนวทางเดียวกัน เป็นต้น

2.1.7 HTML กับ CGI

นั้นส่วนที่จำเป็นต้องทราบคือ ภาษา HTML ที่มีไว้สำหรับเขียนเว็บเพจ ทั้งนี้เพราะที่บราวเซอร์ จะสนับสนุน HTML ในการจัดการรูปแบบของไฮเปอร์เท็ก (Hyper Text) และจุดนี้เอง HTML ก็คือส่วนในการจัดการอินเตอร์เฟสของโปรแกรม หรือ CGI แอพพลิเคชันนั่นเอง แท็กของ HTML ที่สำคัญในการจัดการและใช้งานใน CGI แอพพลิเคชันคือ แท็กฟอร์ม (Form Tag)

ฟอร์มแท็กของ HTML

ฟอร์มแท็กของภาษา HTML นั้นนับว่ามีความสำคัญมากทั้งนี้เพราะ ถ้ามี CGI แอพพลิเคชันแต่ไม่รู้ว่าจะเรียกว่าใช้งานหรือจะส่งข้อมูลไปให้ได้อย่างไร คงไม่มีทางที่จะได้พัฒนา แอพพลิเคชันใช้งาน เราจะเห็นว่าบทบาทของ HTML ในการพัฒนา CGI แอพพลิเคชัน นั้นคือว่าส่วนที่ติดต่อกับผู้ใช้ระบบหรือยูสเซอร์ โดยตรงโดยข้อกำหนดของ HTMLระบุวิธีการใช้งานเรื่องCGI ตามมาตราฐานไว้แล้ว มีชื่อเรียกว่า "ฟอร์มแท็ก" หรือ "แท็กฟอร์ม" โดยมีรูปแบบ คือ "<FORM>" สำหรับเริ่มต้นการใช้งานฟอร์มและ </FORM> สำหรับสิ้นสุดการใช้งานฟอร์ม โดยรายละเอียด ของการใช้งานฟอร์มแท็กนี้ จะได้อธิบายอย่างละเอียดในบทต่อไปแต่ในบทนี้เป้าหมายคือต้องการให้ผู้ศึกษาได้เห็นภาพโดยรวมของการทำงานก่อน เริ่มด้วยฟอร์มของ HTML ทำให้ยูสเซอร์สามารถ ติดต่อผ่านบราวเซอร์โปรแกรมที่ฝั่งไคลเอ็นต์กับ CGI โปรแกรม ผ่านเว็บเซิร์ฟเวอร์ที่ฝั่งเซิร์ฟเวอร์ โดยลักษณะการทำงานนั้นจะเป็นในลักษณะ แบบโต้ตอบ (Interaction) แต่ในปัจจุบัน มีหลายภาษานอกเหนือจาก HTML ที่ทำให้เราสามารถทำงานกับแอพพลิเคชัน หรือติดต่อส่วนอื่น ๆ ที่ ฝั่งด้านเซิร์ฟเวอร์ได้ แต่เราจะไม่ขอกล่าวในโครงงานเล่มนี้ เพราะเราต้องการให้ศึกษา วิธีที่เป็นมาตราฐานเพื่อประโยชน์ในการศึกษาและใช้งานเราสามารถสร้างฟอร์มได้โดยอาศัยข้อกำหนดของ HTML ที่ระบุเกี่ยวกับฟอร์ม และการใช้งานซึ่งมีรูปแบบสำหรับการทำรูปแบบนำเสนอ ในแอพพลิเคชันได้หลากหลายพอเพียงสำหรับการพัฒนาแอพพลิเคชัน ดังตัวอย่างข้างต้น เราใช้ชนิดของอินพุท (Input) แบบต่าง ๆ สำหรับรับค่าข้อมูลจากยูสเซอร์ในหลาย ๆ รูปแบบเหมือนแอพพลิเคชันอื่นๆ เมื่อเราทราบวิธีการใช้งานฟอร์มแท็กสำหรับรับค่าข้อมูลจากยูสเซอร์แล้วจากนั้นเมื่อยูสเซอร์มีการเรียกใช้โปรแกรม หรือ CGI ซึ่งจะระบุไว้ในฟอร์มว่าจะเรียกใช้โปรแกรมใด โดยการซับมิท (Sumit) ซึ่งซับมิทจะเป็นรูปแบบการใช้งานชนิดหนึ่งของฟอร์ม เมื่อมีการกำหนด จะแสดงเป็นปุ่ม ดังรูปจะใช้ในชื่อ " Click Me" ขณะที่เรียกใช้ หรือมีการกดปุ่มซับมิทแล้วจะมีการส่งข้อมูลที่ยูสเซอร์ใส่ค่าผ่านฟอร์ม (ซึ่งข้อมูลจะถูกเปลี่ยนรูปแบบ หรือ เข้ารหัสในรูปแบบ ที่สามารถทำการจัดส่งได้) ไปให้ยัง

เว็บเซิร์ฟเวอร์และเว็บเซิร์ฟเวอร์จะเรียกใช้งาน CGI โปรแกรม และในตัว CGI โปรแกรมจะติดต่อใช้งานค่าต่าง ๆ นั้นอีกทีหนึ่ง จากนั้นก็จะประมวลผลการทำงาน ซึ่งแล้วแต่โค้ดหรือโปรแกรมจะกำหนดว่าทำงานอะไร เมื่อเสร็จแล้วจะส่งผลลัพธ์กลับมายังเว็บเซิร์ฟเวอร์เพื่อส่งให้บราวเซอร์ที่ไคลเอ็นต์ต่อไป

2.1.8 การทำงานของ HTTP โพรโตคอล

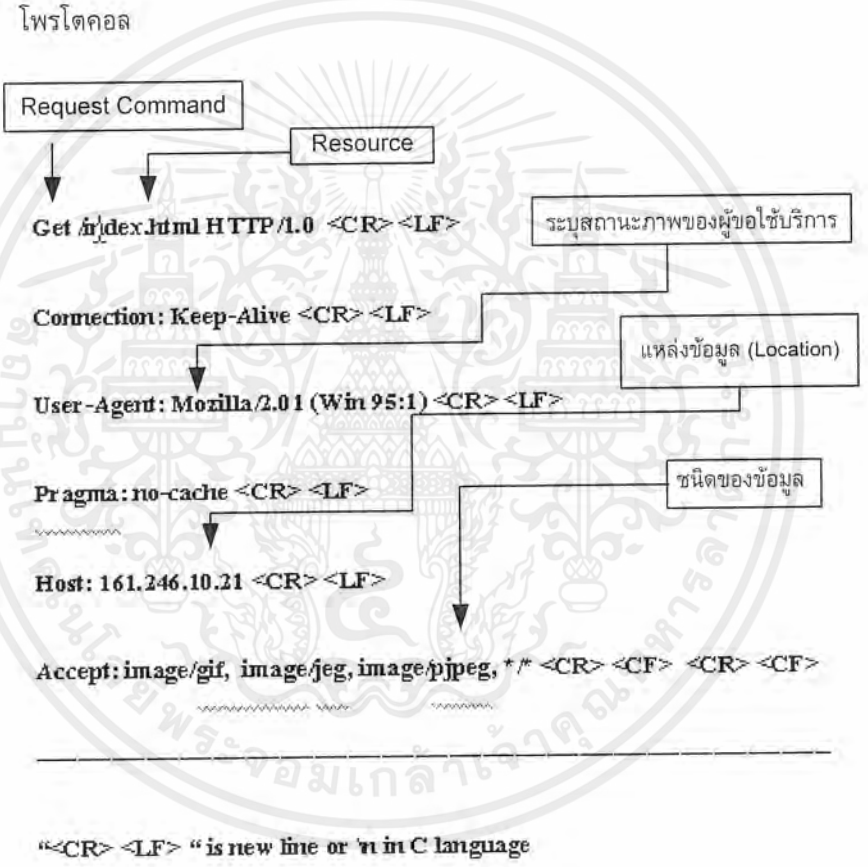
HTTP ย่อมาจาก HyperText Transfer Protocool เป็นข้อกำหนดหรือวิธีในการติดต่อสื่อสารหรือจัดการข้อมูลประเภทไฮเปอร์เท็ก หรือเรามักเรียกว่าไฮเปอร์มีเดีย ทั้งนี้เพราะ ตัวรูปแบบเอกสารเปลี่ยนไปจากเดิมที่เป็นเอกสารแบบข้อความ (text) เชื่อมโยงกันเป็นโครงข่าย แต่ปัจจุบันข้อมูลอาจเป็นได้ทั้งภาพ เสียง หรืออื่น ๆ (พวกมัลติมีเดีย Multimedia) ซึ่ง HTTP โพรโตคอลเป็นพื้นฐานในการใช้งาน หรือการทำงานสื่อสาร ของเครือข่ายเวิร์ลไวด์เว็บ โดยการทำงานของโพรโตคอล ดังกล่าวนี้มีส่วนต่าง ๆ ที่ควรทราบดังนี้

รูปที่ 2.1.15 แสดงการทำงานของ HTTP โพรโตคอล

จากภาพข้างต้น แสดงการทำงานของ HTTP โพรโตคอล ซึ่งการทำงาน จะแบ่งเป็นสามส่วนหลัก ๆ คือ ส่วนไคลเอ็นต์, เซิร์ฟเวอร์ และเครือข่าย ตามหลักการพื้นฐาน ของวิธีการไคลเอ็นต์-เซอร์ฟเวอร์ ฝั่งไคลเอ็นต์จะใช้งานบราวเซอร์โปรแกรมติดต่อกับเว็บเซิรฟ์เวอร์ซึ่งคือผู้ให้บริการเว็บโดยโปรแกรมดังกล่าว จะเรียกว่า HTTPD (ย่อมาจาก HyperText Transfer Protocol Daemon) ปกติแล้วจะทำงานที่ พอร์ท (Port) 80 เราทราบแล้วว่าหลักการของ ไคลเอ็นต์-เซิร์ฟเวอร์ คือมีตัวไคลเอ็นต์ เป็นตัวขอใช้บริการ และมีเซิร์ฟเวอร์ เป็นผู้ให้บริการโดยวิธีการทำงาน จะใช้วิธีการRequest หรือ การร้องขอและการResponse/Reply หรือการตอบสนองการขอบริการ ต่อไปเราจะมาดูว่าโพรโตคอล HTTP มีการ Request และ Response อย่างไร

การร้องขอบริการ (Request)

สำหรับ HTTP จะมีคำสั่งหลาย ๆ ในการจัดการ ดังนี้ OPTION, HEAD, PUT, DELETE, TRAGE GET และ POST แต่คำสั่งหลัก ๆ สำหรับผู้ขอใช้บริการ มักใช้บ่อยคือ GET, POST และ HEAD โดยรายละเอียดการใช้คำสั่ง สำหรับการร้องขอนั้น จะมีการระบุรายละเอียด ไว้ในตัวเอง HTTP โพรโตคอล



รูปที่2.1.16 แสดงตัวอย่างการร้องขอบริการของ HTTP

การร้องขอบริการจะมีความเกี่ยวกับ การระบุถึงสถานที่ที่ต้องการใช้บริการและรายละเอียดที่อยู่ของข้อมูลต่าง ๆ ตามหลักการของ URL (Uniform Resource Locators) ซึ่ง URL คือส่วนระบุเพิ่มเติมในการเข้าถึงข้อมูลโดยสามารถใช้วิธีการใช้งานหรือโพรโตคอลได้หลายวิธีดังนั้นข้อดีของURL

คือสามารถเข้าถึงแหล่งข้อมูลต่างๆได้ตามวิธีการที่ควรจะเป็นหรือเหมาะสมเช่น http://www.kmitl.ac.th/~s8626072/ fttp://s8626072@kmitl.ac.th นอกจากนี้ยังสามารถเรียกในลักษณะของแอพพลิเคชันอื่น ๆ ได้ เช่น mailto:s8626072@kmitl.ac.th เป็นต้น

จากการร้องขอบริการข้างต้นจะเป็นการเรียกขอโฮมเพจ ที่ชื่อ "index.html" โดยตรงนี้อาจจะระบุแบบ ส่วนขยาย URL ได้เช่น /~s862072/index.html เป็นต้น ส่วนสำคัญถัดมาของการร้องขอบริการ คือส่วนของ User Agent เป็นการระบุรายละเอียดเกี่ยวกับ สถานะภาพของไคลเอ็นต์ เช่น โปรแกรมที่ใช้งานเป็น บราวเซอร์ของ Netscape หรือ IE ใช้งานกับ Windows 95 เป็นต้นส่วนต่อมาคือโฮสต์ (Host) ซึ่งจะเป็นไซต์ปลายทาง และรายละเอียดนี้จะสัมพันธ์ กับส่วนเพิ่มเติมของ URL ดังเช่นจากตัวอย่างจะเป็นการระบุURLเท่ากับhttp://16.246.10.21/index.html"หรือ http://www.kmitl.ac.th/index.htmlในกรณีที่สองจะมีการนำค่าชื่อ โฮสต์ www.kmitl.ac.th " ไปแปลงเป็นค่า IP Address เท่ากับ"16.246.10.21" ค่าโฮสต์ = 161.246.10.21 และ GET จะได้ค่าเท่ากับ /index.html

การตอบกลับ/การให้บริการ (Response หรือ การ Reply)

ในการตอบกลับการให้บริการของ HTTP นั้นจะมีรูปแบบในการใช้งานเหมือนโพรโตคอล อื่น ๆ ตามหลักการของไคลเอ็นต์-เซิร์ฟเวอร์ทั่วไปคือจะมีการส่งค่ากลับนำด้วยหมายเลขซึ่งเรียกว่า "Response Tags Number หรือ Status Code" และจะตามด้วยรายละเอียดข้อความซึ่งอธิบายจากนั้นส่วนท้ายสุดที่อาจจะมีส่งตามคือตัวของข้อมูลจริง ๆ ในกรณีที่มีการขอข้อมูล เช่น จากการ ร้องขอ ข้างต้นเราจะได้รับข้อมูล ที่เว็บเซิร์ฟเวอร์ หรือ เซิร์ฟเวอร์ส่งกลับมาให้ดังนี้ หมายเลขการ Response จะเป็นค่ามาตรฐาน แต่ข้อความที่ตามมานั้นอาจจะไม่เหมือนกันก็ได้ทั้งนี้แล้วแต่ยี่ห้อของผลิตภัณฑ์ของเว็บเซิร์ฟเวอร์ ดังนั้นในการเขียนโปรแกรมประเภท ไคลเอ็นต์เซิร์ฟเวอร์ หรือหากเราจะเขียนบราวเซอร์โปรแกรม เราจะต้องอาศัยการตรวจสอบ การทำงานของไคลเอ็นต์-เซิร์ฟเวอร์จากหมายเลขดังกล่าวไม่ควรใช้ข้อความเพราะอาจจะผิดพลาดได้ เช่น

```
:          :          :
if ($response_number eq "200") {
:          :          :
}
```

ตัวเลขทุกหลักของ Response Tags Number ล้วน มีความหมาย เริ่มที่หลักแรก จะเป็นการแสดงความมีประสิทธิภาพของการร้องขอและตอบกลับว่าเป็นอย่างไร เช่น “200 OK” หรือ “500 Error” เป็นต้น ตัวเลขหลักที่สองจะแจ้งให้ทราบถึงชนิดของการทำงาน เช่นหากเกิด Error เป็นชนิดไหน และหลักสุดท้ายเป็นส่วนย่อยในประเภทของการทำงานนั้น ๆ เป็นการขยายในส่วนรายละเอียดของหลักที่สอง

ตาราง 2.2.1 แสดงรายละเอียดของหมายเลขสถานะภาพการทำงานของ HTTP โพรโตคอล ที่ควรทราบ

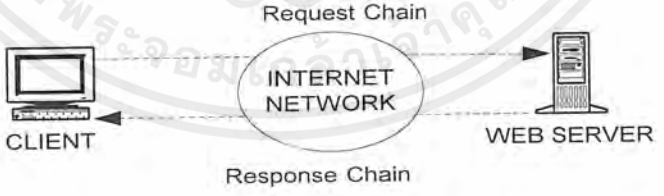
Response Tage Number	รายละเอียด
100	Continue
101	Switching
200	OK
201	Created
202	Accepted
203	Non – Autharitative information
204	No Content
205	Reset Content
206	Partial Content
300	Multiple Choices
301	Moved Permanently
302	Moved Temporarily
303	See Other
304	Not Modified
305	Use Proxy
400	Bad Request
401	Unauthorized
402	Payment Required
403	Forbidden
404	Not Found
405	Method Not Allowed
406	Not Acceptable
407	Proxy Authentication Required
408	Request Time-out
409	Conflict
410	Gone

411	Length Required
412	Precondition Failed
413	Request Entity Too Large
414	Request – URI Too large
415	Unsupported Media type
500	Internal Server Error
501	Not Implemented
502	Bad Gateway
503	Service Unavailable
504	Gateway Time - out
505	HTTP Version not supported

ตาราง2.2.1แสดงรายละเอียดของหมายเลขสถานะภาพการทำงานของ HTTP โพรโตคอลที่ควรทราบ (ต่อ)

ตอนนี้เราได้ทราบการร้องขอ และ การตอบสนอง แล้วซึ่งทั้งสองส่วนเป็นหลักการทำงานทั่ว ๆ ไป ของไคลเอ็นต์-เซิร์ฟเวอร์ ต่อมาเราจะมาดูส่วนของเครือข่ายหรือ Network ในระบบเครือข่ายเวิร์ลไวด์เว็บ มีการเชื่อมต่อระหว่างผู้ใช้บริการ และผู้ให้บริการได้ 2 ลักษณะ คือ

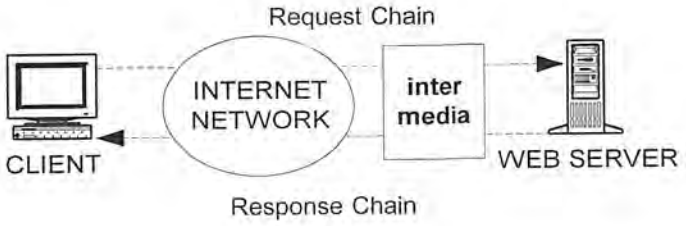
1. การเชื่อมต่อโดยตรง



รูปที่ 2.1.17 แสดงการเชื่อมต่อโดยตรง

ลักษณะการทำงาน คือผู้ขอใช้บริการจะติดต่อกับเซิร์ฟเวอร์หรือผู้ให้บริการโดยตรง ซึ่งส่วนของไคลเอ็นต์จะเรียกว่า User Agent โดยมีบราวเซอร์ทำหน้าที่นี้ให้ และส่วนเซริ์ฟเวอร์ จะเรียกว่า "Origin" จะทำงานกับบราวเซอร์หรือ User Agent นี้โดยตรง

2. การเชื่อมต่อผ่านตัวกลาง



รูปที่ 2.1.18 แสดงการเชื่อมต่อผ่านตัวกลาง

ลักษณะการติดต่อแบบนี้ ส่วนของ User Agent ไม่สามารถติดต่อกับ Origin ได้โดยตรง นั่นคือ ต้องติดต่อผ่านตัวกลางทุกครั้งที่มีการร้องขอบริการ และการตอบสนอง ก็จะต้องผ่านตัวกลาง เช่นกัน ดังนั้น การร้องขอ หรือการตอบสนอง จะมีลักษณะเป็นเหมือนลูกโซ่ โยงผ่านเป็นช่วง ๆ เรียกว่า Request Chain/ Response Chain

ประเภทของตัวกลาง (Intermedia) ตามข้อกำหนดของ HTTP มี 3 ประเภทคือ

1. Tunnel ทำหน้าที่เชื่อมต่อเท่านั้น อาจจะไว้เพื่อความประสงค์อะไรบางอย่าง แต่ตัวกลางนี้จะไม่มีหน้าที่หรืออำนาจในการเปลี่ยนข้อมูลที่วิ่งผ่าน

2. Proxy ส่วนนี้สามารถปรับปรุงรายละเอียดมีการประยุกต์ใช้งานได้ ทั้ง 2 ส่วนคือ ไคลเอ็นต์ และเซิร์ฟเวอร์

3. Gateway ส่วนนี้มักทำหน้าที่เชื่อมต่อ ในกรณีที่ไม่สามารถติดต่อ หรือใช้งานเชื่อมต่อ กับตัวเว็บเซิร์ฟเวอร์ได้โดยตรงตัวกลางที่เรามักพบบ่อยในการใช้งานคือ Proxy ซึ่งมักนำมาติดตั้ง Cache Server หรือ FireWall

2.1.9 การติดตั้งค่าคอนฟิก (Configuration)

ในหัวข้อต่อไปจะกล่าวถึงการทำงานของ CGI ว่ามีการทำงานอย่างไร แต่ก่อนที่เราจะศึกษานั้น เราจำเป็นต้องติดตั้งค่าของเว็บเซิร์ฟเวอร์ ให้สามารถทำงาน หรือใช้งาน CGI ก่อน ในโปรแกรมเว็บ

เซิร์ฟเวอร์ จะมีการแบ่งแฟ้มสำหรับการติดตั้งค่าหลัก ๆ เป็น 2 แฟ้มคือ "httpd.conf" และ "srm.conf" ซึ่งแฟ้มทั้งสองถือว่าเป็นพื้นฐานค่าคอนฟิกที่สำคัญ สำหรับ HTTP โพรโตคอลโดยเราจะมาดูการระบุหรือติดตั้งค่ารายละเอียดที่น่าสนใจ แต่ก่อนจะเริ่มต้องทำความเข้าใจก่อนว่ารูปแบบแฟ้มคอนฟิกนี้จะเป็นลักษณะเท็กไฟล์ธรรมดา โดยค่าที่ติดตั้ง จะมีการแสดง เป็นบรรทัด ๆ ไป หากไม่ต้องการให้ค่าคอนฟิกทำงานเราสามารถลบบรรทัดนั้นทิ้ง แต่จะให้ ดี เราไม่ควรทำแต่ควรใช้วิธีการรีมาร์ค (remark) หรือคอมเมนต์ (comment) ไว้โดยใช้สัญลักษณ์ " # " นำหน้าบรรทัดค่าคอนฟิกนั้น ๆ ซึ่งจะถือว่าค่าคอนฟิกนั้น ๆ ไม่ทำงาน เช่นเดียวกับการลบบรรทัดนั้น ๆ ทิ้งไป และทุกครั้งที่มีการเปลี่ยนแปลงค่าคอนฟิกแล้ว หากต้องการให้เว็บเซิร์ฟเวอร์ ทำงานตามค่าคอนฟิกนั้น ๆ เราจำเป็นต้องรับทั้งโปรแกรมเว็บเซิร์ฟเวอร์ใหม่ทุกครั้ง

2.1.10 การทำงานของ CGI

การทำงานของ CGI ก็ยังคงอาศัยหลักการพื้นฐานของ ไคลเอ็นต์-เซิร์ฟเวอร์ โดยเว็บเซิร์ฟเวอร์จะเป็นผู้ติดต่อขอใช้บริการและรอรับผลลัพธ์ของ CGI กลับมา แล้วส่งต่อให้กับยูสเซอร์ที่ใช้งานผ่านบราวเซอร์ที่ไคลเอ็นต์ลองศึกษาตัวอย่างต่อไปนี้ จะทำให้เห็นภาพการทำงาน ทั้งระบบได้ชัดเจนยิ่งขึ้น

- สมมุติยูสเซอร์ กำหนด URL สำหรับเรียกใช้ CGI โปรแกรม เป็น http://dream.world.net/cgi-bin/helloworld.cgi เมื่อเว็บเซิร์ฟเวอร์รับรายละเอียดมาตรวจสอบพบว่ามีการเรียก CGI หรือ Script Alias ที่กำหนดในคอนฟิกไฟล์ ก็จะทราบทันทีว่าจะต้องอ่านโปรแกรมหรือสคริปต์ที่ ไดเรกทอรี่ใด ตามที่กำหนดในคอนฟิก ซึ่งไฟล์ที่ใช้งานนั้น คือไฟล์ cgi โปแกรม

- หรือไม่เช่นนั้นอาจทำการตรวจสอบชนิดของไฟล์ ก็พบว่านามสกุลดังกล่าวเป็น "cgi" และพบในรายละเอียด คอนฟิกว่า เป็น CGI แอพพลิเคชัน แสดงว่า การเรียกหรือขอใช้บริการครั้งนี้เป็นการเรียกรัน CGI แอพพลิเคชัน เว็บเซิร์ฟจะทำการเรียก CGI ให้ทำงาน และจะรอรับผลลัพธ์ของการทำงานด้วย

- เมื่อโปรแกรมหรือ CGI สคริปต์ ทำงานเสร็จสิ้นก็จะส่งผลลัพธ์กลับมาให้เว็บเซิร์ฟเวอร์ จากนั้นเว็บเซิร์ฟเวอร์ก็จะทำการรับข้อมูลส่งกลับไปให้กับไคลเอ็นต์ การตอบหรือส่งผลลัพธ์ของ CGI ให้กับเว็บเซิร์ฟเวอร์ เพื่อคืนให้ไคลเอ็นต์นั้น จะมีรูปแบบของการส่ง โดยทุกครั้งที่ส่งกลับจะมีการระบุส่วนหัว (Header) ในการติดต่อสื่อสาร ว่าข้อมูลหรือรูปแบบข้อมูลที่ส่งตามมานั้น จะเป็นอย่างไร โดยวิธีการของ MIMEซึ่งวิธีการตอบหรือรูปแบบของการแจ้งส่วนหัวนี้จะมีอยู่ 2 แบบคือการตอบแบบเต็ม (Full Header) และ การตอบแบบย่อหรือแค่บางส่วน (Partial Header) ตัวอย่างเช่น

รูปแบบส่วนหัวแบบ Full Header

HTTP/1.0 200 OK

Date : Friday. 26-February-96 10:10:00 GMT

Server : Apache/1.2

MIME-version : 1.0

Content-Type : text/html

Content-length : 12345

<HTML>

: :

</HTML>

ตัวอย่างโปรแกรมสั่งพิมพ์รูปแบบผลลัพธ์แบบ Full Header

```
#!/usr/bin/perl
$server_protocol=$ENV {'SERVER_PROTOCOL'};
$server_software=$ENV {'SERVER_SOFTWARE'};
$request_file=$ENV{'DOCUMENT_ROOT'}, "/index.html";
        if(open(TEXT, ".$resuest_file))
        {$SIZE=stat ($request_file)[7];
         print "Content-length : $SIZE \n";
         close (TEXT);
        }
print "$server_protocol 200 OK " , "\n",
print "$SERVER SOFTWARE: $server_software " , "\n" ;
print "content-type:text/html \n\n" ;
print "<HTML> " , " \n" ;
print"<HEAD><TITLE> This is Full Header </TITLE></HEAD>
print "<BODY>","\n";
print "<H2>", I am so grad , "</H2>","\n";
print "<BODY></HTML>","\n";
exit(0) ;
```

รูปแบบส่วนหัวแบบ Partial Header

Content-type : text/html

<HTML>

: :

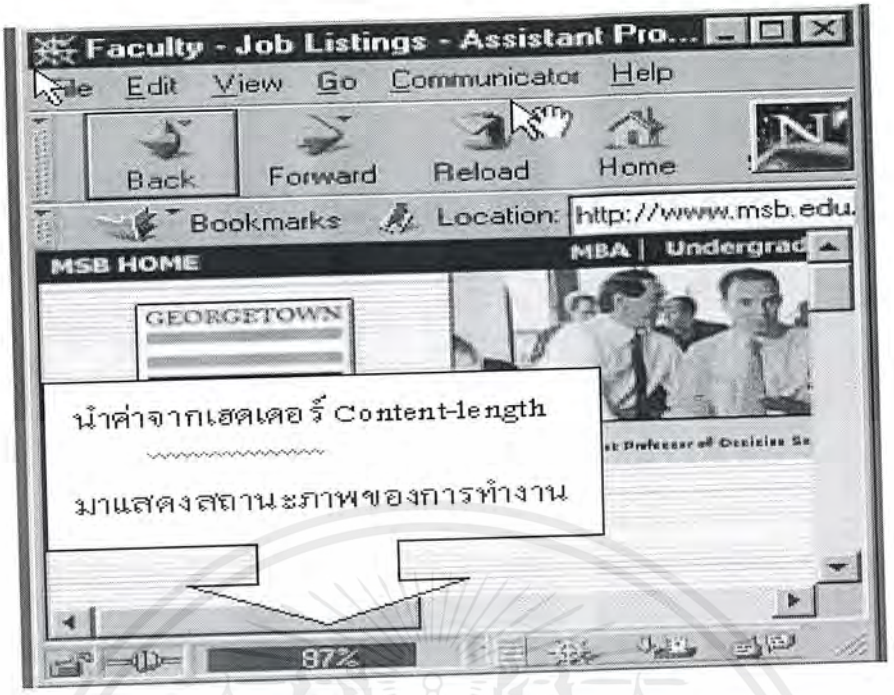
</HTML>

ในการใช้งานสำหรับ CGI แล้ว มักจะเขียนโปรแกรมกำหนด รูปแบบของผลลัพธ์ในโปรแกรมเป็นแบบ Partial Header ทั้งนี้เพราะง่าย และสะดวก อีกทั้งโดยทั่วไปแล้วถ้าแอพพลิเคชันไม่ได้ต้องการแสดงรูปแบบข้อมูลที่ซับซ้อน เช่น การแสดงผลแบบมัลติพาร์ท (Multipart) คือมีข้อมูลเป็นส่วนย่อย ๆ หลายส่วน เนื่องจากข้อมูลที่ส่งจะแบ่งเป็นหลาย ๆ ส่วนดังนั้นจึงจำเป็นต้องคอยบอกรายละเอียดแต่ละส่วนให้ชัดเจน แต่จะใช้เป็นแบบ Partial ก็ไม่ถือว่าผิด

ตัวอย่างโปรแกรมสั่งพิมพ์รูปแบบผลลัพธ์แบบ Partial Header

```
#!/usr/bin/pert
print “content-type:text/html \n\n” ;
print “<HTML> “ , “ \n” ;
print”<HEAD><TITLE> This is Partial Header </TITLE></HEAD>
print “<BODY>”,”\n”;
print “<H2>”, I am so grad , “</H2>”,”\n”;
print “<BODY></HTML>”,”\n”;
exit(0) ;
```

มีคำถามว่า แล้วการพิมพ์ เฮดเดอร์ สำหรับการแสดงผลลัพธ์ ใน HTTP โพรโตคอล ทั้งสองแบบมีความแตกต่างกันอย่างไร ในเมื่อ ในหลักการระบุไว้ว่า ไม่ว่าจะพิมพ์แบบใด ก็ไม่ถือว่าผิด แล้วทำไมถึงต้องแบบ Full Header ด้วย เพราะรู้สึกว่าจะต้องเสียเวลา และยุ่งยากยิ่งขึ้น ผู้เขียนขอชี้จากตัวอย่างให้เห็นภาพเพิ่มขึ้นสักหนึ่งตัวอย่างคือ ถ้าเราใช้บราวเซอร์ แล้วเคยได้สังเกต สถานะการทำงานของบราวเซอร์ ที่ด้านล่างบราวเซอร์ จะเห็นว่าบราวเซอร์มีการแจ้งสถานะภาพว่า ตอนนี้ ไฟล์ หรือ ข้อมูลที่กำลังรับกลับมาแสดงอยู่นั้น เหลืออีกกี่เปอร์เซนต์ นั่นแสดงว่าบราวเซอร์โปรแกรมต้องทราบว่า ข้อมูลหรือ Resource ที่กำลังรับมานั้นมีขนาดของไฟล์เท่าไหร่ จึงจะสามารถคำนวณ และ แสดงสถานะการรับข้อมูลได้ว่า รับมาแล้วกี่เปอร์เซนต์ ของขนาดข้อมูลทั้งหมดที่ส่งมาให้ ซึ่งจุดนี้เองที่ตัวบราวเซอร์โปรแกรม นำข้อมูลที่ได้จาก เฮดเดอร์ มาทำงาน



รูปที่ 2.1.19 ตัวอย่างการนำข้อมูลจากเฮดเดอร์ "Content-Length" มาใช้งาน

จากตัวอย่างที่ยกมาข้างต้นนี้ เป็นเพียงแค่ หนึ่งรูปแบบ ที่นำข้อมูลจาก เฮดเดอร์ ของการแสดงผลลัพธ์ มาใช้งาน และก่อนหน้านี้เราทราบว่าการทำงานของ HTTP จะเปิดการติดต่อเป็นรายการย่อย ๆ ตามงานแต่ละชิ้นทำให้รายละเอียด เฮดเดอร์ จึงแยกเป็นชิ้น ๆ ด้วย ดังนั้นเราจึงเห็นสถานะของการโหลดแฟ้มต่าง ๆ สลับไปมาแล้วแต่จำนวนชิ้นงานที่เปิดการติดต่อ

2.1.11 ภาษาที่ใช้ในการพัฒนา CGI โปรแกรม

มีหลายคนมักสงสัยว่า CGI โปรแกรม จะเขียนหรือสามารถพัฒนาด้วยภาษาอะไร หรือภาษาเพิร์ล (Pert) คือภาษาของ CGI ใช่หรือไม่ คำตอบคือ เพิร์ลเป็นภาษาแบบหนึ่งเช่นเดียวกับภาษา C หรือ C++ หรือ Pascal แล้วภาษาที่ใช้พัฒนา CGI นั้นคือภาษาอะไร คงต้องตอบว่าแล้วแต่แพลตฟอร์ม (Platform) ของเซิร์ฟเวอร์ที่สนับสนุนภาษานั้น ๆ เช่น ยูนิกซ์ ก็อาจจะสามารถใช้งานภาษา C/C++, Shell Script, Pert ส่วน NT ก็อาจใช้งานกับโปรแกรมที่เขียนด้วย Delphi, VB, C/C++ เป็นต้น อันนี้ก็แล้วแต่ว่ามีอะไรให้ใช้ และติดตั้งภาษาอะไรไว้บ้างภายใต้เซิร์ฟเวอร์ หรือสามารถรัน โปรแกรมที่เขียนด้วยภาษาอะไรได้บ้างภายใต้เซิร์ฟเวอร์นั้น ๆ แต่การเลือกว่าจะพัฒนาด้วยภาษาอะไรนั้นเราต้องดูรายละเอียดเพิ่มเติมด้วย เช่น ความง่ายในการจัดการ, ความสามารถในการใช้งาน, โปรแกรมหรือไลบรารีสำเร็จรูปที่สนับสนุนการพัฒนา และ อื่น ๆ อีกด้วยเช่น ความเป็นมาตรฐานของภาษา เพราะถ้าภาษานั้น ๆ เป็นมาตราฐานเรา ก็จะสามารถโยกย้าย หรือเปลี่ยนแปลงแพลตฟอร์ม ได้สะดวกในอนาคต สำหรับ CGI จุดที่ต้องสนใจอีกจุดคือการใช้งานกับตัวแปรค่าสภาพแวดล้อม (Environment Variable) ซึ่งภาษาที่เป็นมาตรฐานจะไม่มีปัญหาและจุดนี้นับว่าสำคัญเพราะตัวแปรเหล่านี้จำเป็นต้องใช้งานในการพัฒนา

โปรแกรมหรือ CGI แอพพลิเคชัน ภาษาที่นิยมสำหรับ CGI คือ C/C++, Apple Script, C Shell, Pert และ VB ต่อไปเราจะมาดูรายละเอียดภาษาแต่ละตัวว่ามีข้อดี – ข้อเสีย อย่างไร เพื่อช่วยสนับสนุนการตัดสินใจในการเลือกใช้ภาษาสำหรับพัฒนา CGI แอพพลิเคชัน

ภาษาต่าง ๆ ที่สามารถพัฒนา CGI แอพพลิเคชัน

-C/C++ ภาษา C/C++ สามารถพัฒนาบนระบบปฏิบัติการยูนิกซ์ (UNIX) , วินโดว์ (Window) และ แมคอินทอช (Macintosh) ได้ จึงเป็นภาษาที่มีความนิยมมาก แต่ว่า ภาษา C/C++ นั้นมี ลักษณะโครงสร้างของการเขียนโปรแกรมที่ค่อนข้างยาก เนื่องจากในตัวเองของภาษาเอง มีความเข้มงวดมาก และยังขาดความสามารถในเรื่อง pattern-matching ด้วย แม้ว่าจะสามารถเขียนในรูปแบบโมดูล และฟังก์ชัน ซึ่งช่วยทำงานได้สะดวกขึ้น อย่างไรก็ตาม CGI โปรแกรมที่เขียนด้วยภาษา C ก็สามารถเรียกใช้ตัวแปรค่าสภาพแวดล้อมได้ และก็มีการใช้งานได้ง่าย

-C Shell มักไม่ค่อยได้รับความนิยมในการพัฒนาโปรแกรม CGI เนื่องจากว่าสามารถ ใช้ได้เพียงระบบปฏิบัติการยูนิกซ์ (UNIX) อย่างเดียว, มีข้อจำกัดเข้มงวด

-Apple Script สามารถพัฒนาบนระบบปฏิบัติการแมคอินทอช (Macintosh) ได้เพียงอย่างเดียวเท่านั้น แต่มีข้อดีในเรื่องของการจัดการกับข้อมูลได้หลายหลาก เนื่องจากว่าตัว Apple Script สามารถอินเตอร์เฟส (interface) ได้ดีกับ Macintosh application หรือ งานที่อยู่บนเครื่อง Mac ซึ่งจะทำงานเกี่ยวกับฐานข้อมูลได้โดยตรงโดยใช้ Apple Event จึงสามารถพัฒนา โปรแกรมได้ไม่ยากนัก สิ่งสำคัญที่ Apple script เหมาะสำหรับ CGI แอพพลิเคชันคือ ง่ายต่อการใช้ syntax เนื่องจากเหมือนภาษาอังกฤษ และมีไลบารีสำหรับการออกแบบพัฒนาการเขียน CGI มากมาย

-VB สามารถพัฒนาได้บนระบบปฏิบัติการวินโดว์ (MS, Windows) เพียงอย่างเดียว แต่มีข้อดีคือง่ายต่อการใช้งาน

-Pert สามารถพัฒนาได้บนระบบปฏิบัติการยูนิกซ์ (UNIX), วินโดว์ (Windows), และรวมไปถึงแมคอินทอช (Macintosh) ด้วยเหตุนี้ ภาษาเพิร์ล จึงเป็นภาษาที่ใช้ในการเขียน CGI โปรแกรมอย่างแพร่หลายที่สุด เนื่องจากว่า เพิร์ล มีข้อดีมากมายที่ช่วยสนับสนุนในการเขียนโปรแกรม และปริญญานิพนธ์เล่มนี้ได้อ้างอิงการเขียน CGI ด้วย ภาษาเพิร์ลด้วย ข้อดีของการใช้ เพิร์ล ในการพัฒนา CGI แอพพลิเคชัน คือ มี pattern-matching, มีความสามารถเรื่อง portable และสามารถจัดการเรื่องสตริงได้ดี เขียนง่าย กระทัดรัด, เรียกคำสั่ง shell ได้ง่าย และ ยังมีการสร้างส่วนเพิ่มขยาย extension มากมาย เพื่อง่ายต่อการอินเตอร์เฟส กับฐานข้อมูล จากข้อดีเหล่านี้เอง เพิร์ล จึงมีความนิยมมากสำหรับเขียน CGI โปรแกรม

2.1.12 ระบบเครือข่ายคอมพิวเตอร์และอินเตอร์เน็ตเวิร์คกิ้ง (Internetworking)

ระบบเครือข่ายคอมพิวเตอร์ (Computer Network) คือระบบการเชื่อมต่อระหว่างระบบปลายทาง (EndSystem) ซึ่งระบบปลายทางเป็นระบบอิสระต่อกัน (Autonomous) ระบบปลายทางสามารถเป็นได้ตั้งแต่ไมโครคอมพิวเตอร์ไปจนกระทั่งถึงซุปเปอร์คอมพิวเตอร์ (Supercomputer) ขนาดใหญ่เพื่อจุดมุ่งหมายในการแลกเปลี่ยนข้อมูลและการแบ่งปันทรัพยากรของระบบ เช่น ไฟล์ (File), เครื่องพิมพ์ (Printer), โมเด็ม (Modem) ตลอดจนการให้บริการฐานข้อมูลร่วม (Sharing database)

อินเตอร์เน็ตเวิร์คกิ้ง หรืออินเตอร์เน็ต (internet) คือการเชื่อมต่อระบบเครือข่าย 2 เครือข่ายขึ้นไป ดังนั้น คอมพิวเตอร์บนระบบเครือข่ายหนึ่ง ก็สามารถติดต่อกับคอมพิวเตอร์บนระบบเครือข่ายอื่น ๆ ได้ องค์ประกอบของอินเตอร์เน็ต อินเตอร์เน็ตเป็นเครือข่ายคอมพิวเตอร์ชนิดหนึ่งที่ใช้โปรโตคอล TCP/IP (Transmission Control Protocal/Internet Protocal) เป็นมาตรฐานการทำงานของระบบ ดังนั้นถ้ามีเครือข่ายคอมพิวเตอร์ที่ใช้โปรโตคอล TCP/IP อยู่แล้วก็จะเป็นการสะดวกและง่ายต่อการเชื่อมต่อเข้ากับระบบอินเตอร์เน็ต ระบบการทำงานของเครือข่าย โปรโตคอล TCP/IP โดยเฉพาะสำหรับเครือข่ายอินเตอร์เน็ตนั้นจะแบ่งกลุ่มของแฟกเตอร์ หรือฟังก์ชั่นการทำงานออกเป็น 6 กลุ่มใหญ่ ๆ ซึ่งการติดตั้งอุปกรณ์ต่าง ๆ เองก็ต้องคำนึงถึงแฟกเตอร์ 6 กลุ่มด้วยกันคือ

ชนิดของสถานีระบบเครือข่าย TCP/IP ส่วนใหญ่จะประกอบด้วยเครื่องคอมพิวเตอร์ที่ทำหน้าที่แตกต่างกันอยู่สองชนิด คือ เครื่องที่ทำหน้าที่ให้บริการที่เรียกว่าโฮสต์ (Host) หรือเซิร์ฟเวอร์ (Server) และคอมพิวเตอร์สำหรับผู้ใช้ทั่วไปเรียกว่าเทอร์มินัล (Terminal) หรือ ไคลเอ็นต์ (Client) โดยที่เครื่องคอมพิวเตอร์ที่ทำหน้าที่เป็นสถานีให้บริการนั้นจะเป็นเครื่องที่คอยให้บริการแก่ผู้ใช้ในด้านต่าง ๆ ไม่ว่าจะเป็นแหล่งเก็บรวบรวมข้อมูล (Data Sharing) การให้บริการโปรแกรมประยุกต์ต่าง ๆ (Application) หรือการให้บริการการใช้งานระบบประมวลผลกลาง (CPU Time Sharing) เป็นต้น ดังนั้นคุณสมบัติโดยทั่วไปทั้งด้านฮาร์ดแวร์และด้านซอฟต์แวร์ของเครื่องโฮสต์หรือเซอร์ฟเวอร์จึงมีคุณสมบัติที่ดีกว่าเครื่องคอมพิวเตอร์สำหรับผู้ใช้มาก

ระบบไอพีแอดเดรส (IP Address)

การสื่อสารข้อมูลในระบบเครือข่ายคอมพิวเตอร์ เป็นการสื่อสารในลักษณะที่เฟรมข้อมูลของแต่ละการสื่อสารเป็นคนกำหนดเส้นทางที่สื่อสารเองคือเมื่อมีการขอติดต่อสื่อสารข้อมูลของเครื่องคอมพิวเตอร์คู่ใดเกิดขึ้น เครื่องคอมพิวเตอร์เครื่องนั้นก็จะทำการสร้างเฟรมข้อมูลขึ้นมาแล้วค่อยส่งออกไปในระบบเครือข่ายโดยที่เฟรมข้อมูลนั้นจะมีส่วนของแอดเดรสที่อยู่ในส่วนอ้างอิงทำกับการสื่อสาร (Header) ที่จะบอกว่า เฟรมข้อมูลนี่เป็นของเครื่องคอมพิวเตอร์เครื่องใดที่กำลังส่งและจะส่งไปยังเครื่องใด

ดังนั้นการที่เครื่องคอมพิวเตอร์ต่าง ๆ จะติดต่อสื่อสารกันในระบบเครือข่ายโปรโตคอล TCP/IP จะต้องมีการกำหนดค่าไอพีแอดเดรสให้แต่ละสถานีที่จะสื่อสารกันด้วย นอกเหนือจากค่า MAC Address ที่มีอยู่ในแต่ละเครื่อง เพราะค่าไอพีแอดเดรสนั้นจะเป็นค่าอ้างอิงในเฟรมข้อมูลที่สื่อสารในเครือข่าย ซึ่งจะมี 2 ชนิดคือ ไอพีแอดเดรสต้นทาง (Source IP Address) และไอพีแอดเดรสปลายทาง (Destination IP Address)

ระบบโปรโตคอลหาเส้นทาง (IP Routing Protocal)

ลักษณะการติดต่อสื่อสารกันระหว่างเครื่องคอมพิวเตอร์ใด ๆ ในระบบเครือข่ายอินเตอร์เน็ต นั้น โดยทั่วไปแล้วมี 2 ลักษณะคือ

1. การเชื่อมต่อภายในเครือข่ายท้องถิ่น (LAN)
2. การเชื่อมต่อระหว่างเครือข่ายท้องถิ่นหนึ่งกับเครือข่ายท้องถิ่นโดยอาจมีการบริการของระบบเครือข่ายระยะไกล (WAN) เข้ามาเกี่ยวข้องด้วย

ในการเชื่อมต่อจำเป็นต้องใช้อุปกรณ์ที่เรียกว่าเราเตอร์ (Router)โดยเราเตอร์จะเกี่ยวข้องกับระบบทำงานที่เรียกว่าโปรโตคอลหาเส้นทาง (Routing Protocal) ซึ่งจะทำหน้าที่ตรวจสอบและจัดการเกี่ยวกับเส้นทางในการสื่อสารข้อมูลทั้งหมดของระบบ

ระบบชื่อกลุ่ม (Domain Name System)

มีการออกแบบระบบชื่อของสถานีบริการต่าง ๆ บนเครือข่ายอินเทอร์เน็ตในรูปลักษณะตัวอักษร เพื่ออำนวยความสะดวกต่อการใช้งานของยูสเซอร์ ระบบ DNS เป็นระบบซอฟร์แวร์ที่ทำหน้าที่ในการจัดสรรและบริการในส่วนการเปรียบเทียบค่าระหว่างชื่อตัวอักษรกับค่าไอพีแอดเดรสของเครื่องสถานีต่าง ๆ บนอินเตอร์เน็ตโปรแกรมประยุกต์บนเครือข่ายอินแตอร์เน็ต (Application)

ระบบเครือข่ายอินเตอร์เน็ตเป็นระบบเครือข่ายขนาดใหญ่ที่มีการเชื่อมโยงกันทั่วโลก ดังนั้นการใช้งานโปรแกรมประยุกต์ต่าง ๆ บนระบบอินเทอร์เน็ตจึงมีลักษณะพิเศษแตกต่างจากการใช้งานบนระบบเครือข่ายท้องถิ่นทั่วไป คือจะมีโปรแกรมประยุกต์มากมายหลายชนิด เช่น ระบบจดหมายอิเลคทรอนิคส์ (E-mail) ระบบข่าวสารรวม (Usenet) ระบบอุโมงค์อินเทอร์เน็ต (Gopher) และระบบเครือข่ายใยแมงมุม (World-Wide-Web)เป็นต้นโดยที่แต่ละชนิดมีการใช้งานที่แตกต่างกันมากระบบความปลอดภัย (Security) มีหน้าที่ป้องกันไม่ให้มีการลักลอบเข้ามาใช้หรือทำลายข้อมูลที่สำคัญ

2.1.13 โปรโตคอล TCP/IP

ข้อแตกต่างระหว่างโปรโตคอล TCP/IP และรูปแบบของ OSI

-ลำดับการติดต่อสื่อสารของชั้นเลเยอร์ในรูปแบบ OSI นั้นจะกำหนดลำดับชั้นการสื่อสารที่เป็นลำดับขั้นตอนการติดต่อที่แน่นอนโดยเฉพาะการอินเตอร์เฟสระหว่างเลเยอร์ ซึ่งทำให้ระบบ OSI สามารถเป็นระบบเปิดสำหรับระบบเครือข่ายคอมพิวเตอร์ทั่วไป เพราะไม่ว่าจะมีการเปลี่ยนแปลงโปรโตคอลในเลเยอร์ชั้นใดก็ตามจะไม่มีผลกระทบต่อการสื่อสารเลเยอร์ชั้นถัดไป ในขณะที่ชุด โปรโตคอล TCP/IP จะไม่มีการกำหนดรูปแบบการติดต่อที่ตายตัว เพื่อให้ผู้ออกแบบสามารถออกแบบโครงสร้างของเครือข่ายได้อย่างอิสระ

-การติดต่อสื่อสารระหว่างเครือข่ายหรือการอินเทอร์เน็ตคือการติดต่อสื่อสารข้อมูลระหว่างระบบคอมพิวเตอร์ 2 ระบบที่ไม่สามารถติดต่อสื่อสารกันได้โดยผ่านทางเครือข่ายการสื่อสารข้อมูลเพียงเครือข่ายเดียวได้ ต้องอาศัยเครือข่ายตั้งแต่ 2 เครือข่ายขึ้นไปในการติดต่อสื่อสารกัน และเครือข่ายเหล่านี้อาจจะมีลักษณะของเครือข่ายที่ต่างกันได้ความแตกต่างในเรื่องของอินเตอร์เน็ตระหว่างชุดโปรโตคอล TCP/IP กับรูปแบบ OSI ก็คือในชุดโปรโตคอล TCP/IP จะใช้โปรโตคอลสำหรับอินเทอร์เน็ตที่เรียกว่าโปรโตคอล IP (Internet Protocal) ซึ่งในรูปแบบ OSI จะเรียกว่าโปรโตคอลสำหรับการอินเตอร์เน็ตว่าโปรโตคอลเน็ตเวิร์ค

-การบริการการเชื่อมต่อข่าวสาร (connection service) ในชุดโปรโตคอล TCP/IP นั้นจะมีการบริการการเชื่อมต่อการสื่อสารระหว่างต้นทางและปลายทาง 2 แบบ คือการบริการแบบคอนเนคชันเลส (Connectionless) และแบบ คอนเนคชันโอเรียนท์ (Connection-Oriented) ส่วนในรูปแบบ OSI จะให้ความสำคัญเฉพาะกับการบริการแบบ คอนเนคชันโอเรียนท์เท่านั้น

-โปรโตคอล ควบคุมการจัดการสื่อสารในชุดโปรโตคอล TCP/IP จะใช้โปรโตคอล TCP (Transmission Control Protocal) เป็นโปรโตคอล สำหรับควบคุมการสื่อสาร กำหนดตำแหน่งต้นและปลายทาง และอื่น ๆ กับข้อมูล ซึ่งในรูปแบบ OSI นั้นจะแบ่งแยกการควบคุมการสื่อสารออกจากกันโดยใช้โปรโตคอลเซสชั่นและโปรโตคอล ทรานสปอร์ตตามลำดับ

ลักษณะการติดต่อ แบ่งออกเป็น 2 ชนิดคือ

-คอนเนคชั่นโอเรียนท์ (Connection- Oriented) คือการติดต่อที่ต้องมีการเชื่อมโปรเซสที่จะทำการติดต่อก่อนที่จะมีการส่งหรือรับข้อมูล ซึ่งสามารถใช้คำว่าวงจรเสมือน(Virtual Circuit) เพราะว่าจะทำงานเสมือนมีวงจรต่ออยู่ระหว่างโปรเซส ถึงแม้ว่าข้อมูลนี้อาจจะผ่าน Packet-Switching Network บริการชนิดนี้ส่วนมากจะใช้ในกรณีที่มีข่าวสารต้องการมากกว่าหนึ่งข่าวสารดังนั้นสามารถแบ่งขั้นการทำงานออกเป็น

ขั้นการสร้างการติดต่อ (Connection Establishment)

ขั้น การส่งผ่านข้อมูล (Data Transfer)

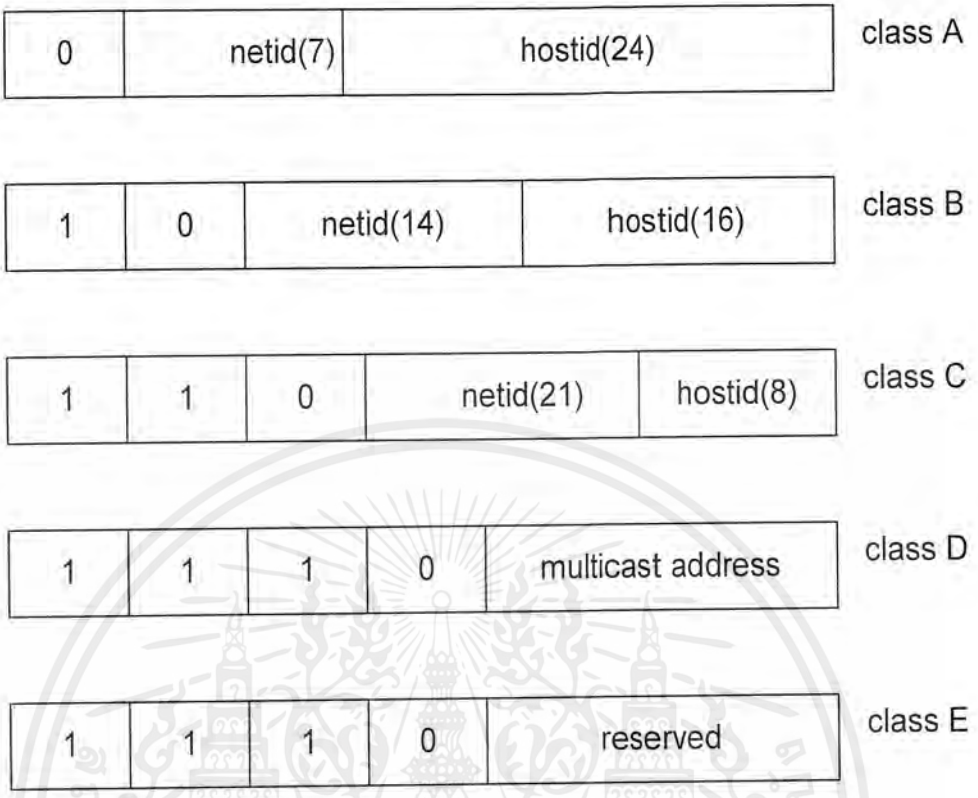
ขั้นยกเลิกการติดต่อ (Connection Termination)

- คอนเนคชั่นเลส (connectionless) หรือ ดาต้าแกรม (Datagram) คือจะไม่มีการสร้างการติดต่อและขั้นการยกเลิกการติดต่อ แต่จะมีขั้นการส่งผ่านข้อมูลเพียงอย่างเดียว โดยข้อมูลที่ซึ่งเรียกว่าดาต้าแกรมจะถูกส่งจากระบบหนึ่งอย่างเป็นอิสระโดยไม่ขึ้นอยู่กับดาต้าแกรมอื่น

-ไอพีแอดเดรส (IP Address) เป็นหัวใจสำคัญของโปรโตคอล IP เพราะเป็นแอดเดรสที่บ่งบอกถึงสถานีปลายทางจริง ๆ ภายในระบบเครือข่ายขนาดใหญ่ที่มีการเชื่อมต่อทั้ง LAN และ WAN ทำให้ผู้ใช้งานใช้ระบบเครือข่ายเสมือนเป็นระบบเครือข่ายเดียวกันได้ และมีวัตถุประสงค์ให้ผู้ใช้สามกำหนดแอดเดรสของสถานีได้ตามต้องการ ไอพีแอดเดรสมีขนาด 32 บิทแบ่งออกเป็น 5 ประเภท คือ A, B, C, D, E มีใช้งานโดยทั่วไปอยู่ 3 ประเภทแรก ส่วนแบบ D ใช้ในกรณีพิเศษ ส่วน E ถูกสำรองไว้ในอนาคต

ไอพีแต่ละประเภทมีหมายเลขที่ไม่ซ้ำกัน ซึ่งก็หมายความว่า ถ้ากำหนดสถานีใด ๆ ด้วยไอพีแอดเดรส หมายเลขของสถานีก็ไม่ซ้ำกันด้วย เมื่อดูแค่สามประเภทแรก จะพบว่าภายในไอพีแอดเดรสขนาด 32 บิทแบ่งออกเป็น 2 ส่วนย่อยคือ หมายเลขเครือข่าย (network identification) และหมายเลขสถานี (host identification)

ในการอ่านไอพีแอดเดรสจะไม่มองเป็นเลขฐานสอง แต่จะมองเป็นเลขฐานสิบแทน โดยการแบ่ง 32 บิทออกเป็น 4 ไบต์ ย่อย แต่ละไบต์แทนด้วยเลขฐานสิบ 1 ตัว เรียกว่า “dotted decimal” ดังนั้นเลขแต่ละตัวของไอพีแอดเดรสมีค่า 0-255 เช่น 10011110011011000000010011100010 เขียนในรูป dotted decimal ได้เป็น 158.108.4.226 ส่วนค่าหมายเลขเน็ตเวิร์กนั้นถ้าเป็นแบบ A จะต้องขึ้นต้นด้วย 0 แบบ B ต้องขึ้นต้นด้วย 10 และแบบ C ต้องขึ้นต้นด้วย 110

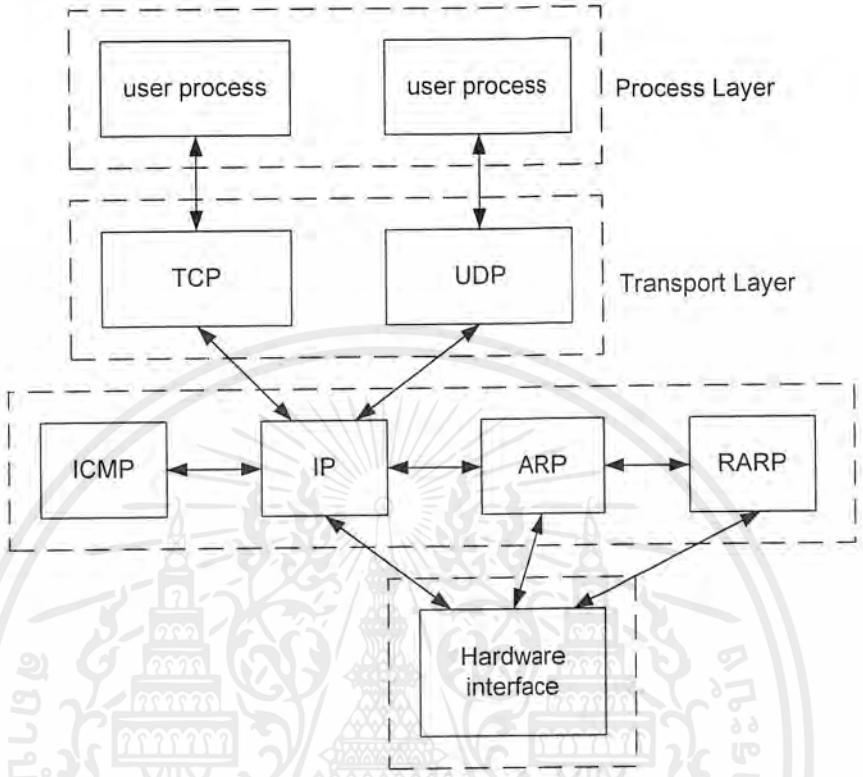


รูปที่ 2.1.20 แสดงประเภทของไอพีแอดเดรส

<u>โครงสร้างของชุดโปรโตคอล TCP/IP</u>

โครงสร้างของสถาปัตยกรรมของชุดโปรโตคอล TCP/IP นั้นแบ่งออกเป็น 3 ส่วนหลัก ๆ คือส่วนของกรรมวิธีปฏิบัติหรือโปรเซส (Process) โฮสต์ (Host) และเครือข่าย (Network) ในส่วนของโปรเซสก็ได้แก่แอปพลิเคชั่นหรือเอนทิลิตี้ที่ต้องการติดต่อสื่อสาร ทุกโปรเซสจะกระทำในเครื่องของโฮสต์ ซึ่งในแต่ละโฮสต์จะสามารถมีหลายเอนทิตี้ ได้พร้อมกัน การสื่อสารกันระหว่างเอนทิตี้ของโฮสต์เครื่องหนึ่งกับเอนทิตี้ของโฮสต์อีกเครื่องหนึ่ง หรือหลายเครื่องจะกระทำโดยผ่านทางเครือข่ายที่โฮสต์เชื่อมต่ออยู่

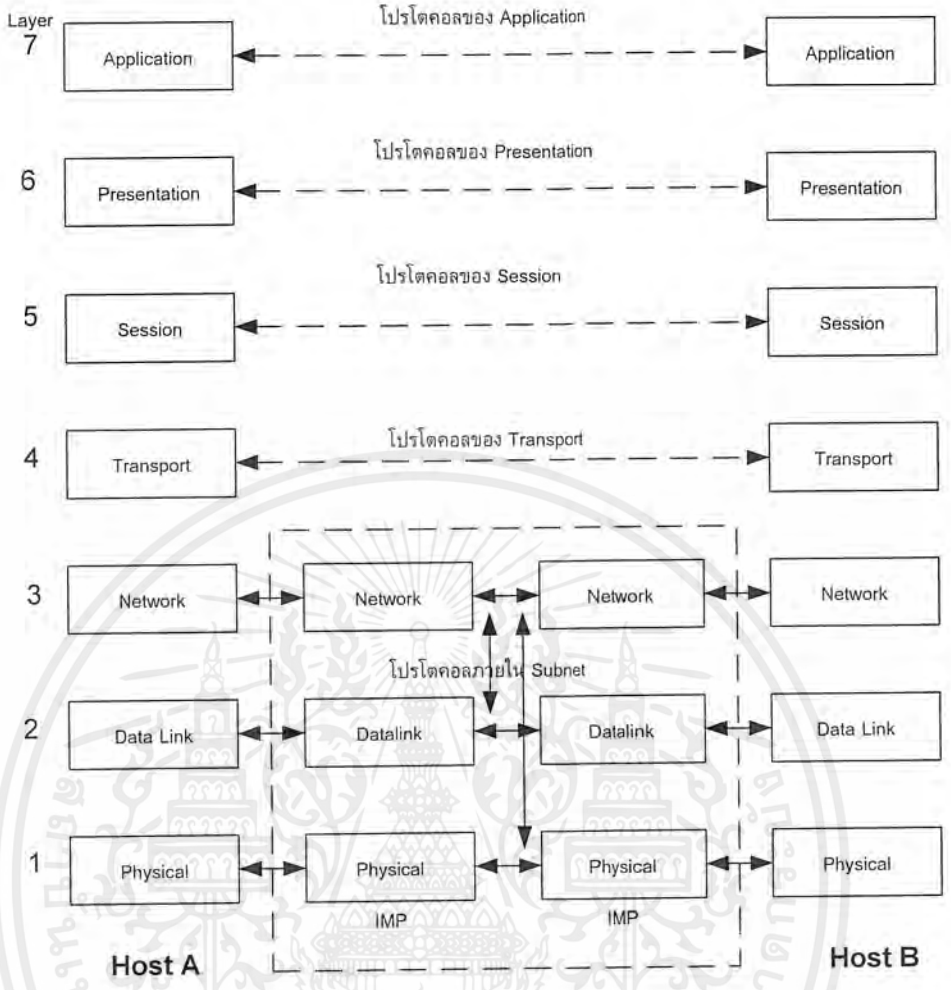
การทำงานที่สัมพันธ์กันระหว่างโปรเซส โฮสต์ และเครือข่ายของสถาปัตยกรรม TCP/IP ทำให้สามารถจัดรูปแบบของสถาปัตยกรรม TCP/IP ได้เป็น 4 ชั้น และสามารถกำหนดชนิดของโปรโตคอล ที่ทำงานในแต่ละชั้นได้ 4 โปรโตคอลเช่นกัน ดังที่ได้กล่าวมาแล้วว่าในชุดโปรโตคอล TCP/IP นั้นเอนทิตี้แต่ละชั้นอาจจะติดต่อสื่อสารข้อมูลโดยผ่านเอนทิตี้ในชั้นเดียวกัน หรือเอนทิตี้ในชั้นล่างลงไปซึ่งไม่จำเป็นต้องเป็นชั้นที่ติดกัน



รูปที่ 2.1.21 แสดงความสัมพันธ์ระหว่างชั้นต่าง ๆ ของอินเตอร์เน็ต

2.1.14 แบบจำลอง OSI (OSI model)

แบบจำลองOSI (Open System Interconnection model) เป็นรูปแบบของเครือข่ายที่ถูกเสนอขึ้นโดยองค์กร ISO(International Standard Organization)ในปี1983 ซึ่งมีจุดประสงค์ใหญ่เพื่อให้ระบบคอมพิวเตอร์ต่างๆ สามารถเชื่อมต่อเป็นระบบเดียวกันได้ไม่ว่าเครื่องนี้จะเป็นของบริษัทผู้ผลิตรายใด แต่ว่ารูปแบบในการเชื่อมต่อกันเป็นเครือข่ายนั้นจะปฏิบัติตามแนวทางเหมือนกันตามแบบจำลอง OSI นอกจากนี้แล้วในแต่ละชั้นของแบบจำลอง OSI ก็ต้องใช้โปรโตคอลในแบบเดียวกันด้วย ปัจจุบันแบบจำลอง OSI มีรูปแบบเป็นลำดับขั้นโปรโตคอล (protocal hierarchies) โดยแบ่งเป็นชั้นทั้งหมด 7 ชั้นด้วยกัน



รูปที่ 2.1.22 แสดง OSI โมเดลและการเชื่อมต่อ

ชั้นฟิสิคัล (physical layer)

ชั้นฟิสิคัลนี้เป็นชั้นที่เกี่ยวข้องกับการติดต่อสื่อสารในระดับเบื้องต้นที่เรียกว่าบิทดิบ จุดประสงค์ใหญ่ในการออกแบบหน้าที่ของชั้นคือเมื่อผู้ส่งบิทที่มีค่าเป็น 1 ออกไป ทางด้านผู้รับก็จะต้องรับค่าได้เป็น 1 เช่นเดียวกัน จากความต้องการดังกล่าวทำให้เกิดเป็นข้อกำหนดต่าง ๆ เช่น การกำหนดแรงดันไฟฟ้าที่มีค่าเป็น 1 และ 0 ช่วงเวลาในการส่งบิท 0 ที่ติดกันนั้นห่างกันได้น้อยแค่ไหน ความเป็นไปได้ในการส่งสัญญาณใน 2 ทิศทางโดยมีช่องสัญญาณเดียวกันและในช่วงเดียวกัน การเริ่มต้นการเชื่อมต่อและยกเลิกการเชื่อมต่อเมื่อการติดต่อสิ้นสุดลงว่าเป็นอย่างไร ตลอดจนจำนวนของพิน (pin) และหน้าที่ของแต่ละพินว่าเป็นอย่างไร จะเห็นว่าในชั้นนี้ต้องเกี่ยวข้องกับคุณสมบัติด้าน ไฟฟ้าเครื่องกลเป็นส่วนใหญ่

ชั้นดาต้าลิงค์ (data link layer)

หน้าที่หลักของชั้นดาต้าลิงค์คือการจัดส่งข้อมูลผ่านไปยังชั้นชั้นฟิสิคัล รวมทั้งจัดการเกี่ยวกับการตรวจจับความผิดพลาดและการแก้ไขความผิดพลาด ถ้าจะสรุปก็คือเมื่อมองจากชั้นที่อยู่เหนือชั้นดาต้าลิงค์ถัดขึ้นไปซึ่งก็คือชั้นเน็ตเวิร์กนั้น ตัวของชั้นเน็ตเวิร์กจะต้องมองเป็นว่าการส่งผ่านข้อมูลจากตัวเองไปยังชั้นเน็ตเวิร์กของคอมพิวเตอร์ตัวอื่นนั้นเป็นการส่งผ่านข้อมูลที่ถูกต้องสมบูรณ์ไม่มีข้อผิดพลาดเลย ซึ่งก็เป็นหน้าที่ของชั้นดาต้าลิงค์ที่ต้องให้บริการแก่ชั้นเน็ตเวิร์ก

เมื่อชั้นดาต้าลิงค์ได้รับข้อมูลที่ส่งผ่านชั้นเน็ตเวิร์กแล้ว ก็จะดำเนินการแบ่งข้อมูล จากนั้นจึงทำการส่งไปยังชั้นดาต้าลิงค์ของคอมพิวเตอร์ทางด้านรับ ด้านรับเมื่อได้รับเฟรมข้อมูลแล้วก็จะส่งเฟรมตอบรับ (acknowledgement frame) กลับไปยังเครื่องคอมพิวเตอร์ทางด้านส่ง หากพิจารณาว่าข้อมูลได้ถูกส่งเรียงกันไปโดยไม่สนใจว่าบิทที่ต่อกันนั้นจะประกอบกันเป็นเฟรมหรือไม่ ดังนั้นจะเป็นหน้าที่ของชั้นดาต้าลิงค์ที่จะต้องกำหนดขอบเขตและขนาดของเฟรมข้อมูลเอาเอง ซึ่งชั้นด้าต้าลิงค์จะทำโดยการแทรกบิทพิเศษเอาไว้ที่ต้นเฟรมและที่ท้ายเฟรม

ชั้นเน็ตเวิร์ก (network layer)

ชั้นเน็ตเวิร์กนี้มีหน้าที่หลักเกี่ยวกับการควบคุมการทำงานของซับเน็ต ซึ่งต้องกำหนดว่าแพ็กเกจใดจะถูกส่งไปยังเส้นทางใด (route) จากด้านส่งไปยังด้านรับ โดยเส้นทางนี้อาจจะถูกกำหนดออกมาตายตัวเลยระหว่างตัวประมวลผลเชื่อมต่อข่าวสาร เรียกว่า IMP (Interface Message Processors) กับ IMP อีกตัวหนึ่ง หรือการกำหนดแบบไดนามิค (dynamic) ซึ่งทำให้การติดต่อระหว่าง IMP แต่ละคู่นั้นจะใช้เส้นทางในการติดต่อแตกต่างกันทั้งนี้เพื่อลดปัญหาด้านทราฟฟิคในเครือข่ายชั้นทรานสปอร์ต(transport layer)

หน้าที่หลักของชั้นทรานสปอร์ต นี้คือการทำการรับข้อมูลจากชั้นเซสชัน จากนั้นทำการแยกข้อมูลออกเป็นหน่วยที่เล็กลง จากนั้นจึงทำการส่งหน่วยต่าง ๆ เหล่านี้ลงไปยังชั้นเน็ตเวิร์กและต้องกระทำการอันเป็นที่แน่ใจว่าหน่วยต่าง ๆ จะต้องส่งไปถึงจุดหมายอย่างถูกต้องตามลำดับ ในสภาวะปกตินั้นชั้นทราบสปอร์ตจะทำหน้าที่คอยสร้างเส้นทางในการติดต่อสื่อสารตามที่ชั้นเซสชันร้องขอมาเมื่อชั้นเซสชันต้องการส่งผ่านข้อมูล และทางชั้นทรานสปอร์ตก็ต้องการให้การไหลเวียนของข้อมูลเป็นอย่างรวดเร็ว ก็อาจจะสร้างเส้นทางขึ้นมาหลายเส้นทางแล้วแยกข้อมูลออกเป็นหน่วยย่อย ๆ เพื่อที่จะส่งข้อมูลแต่ละหน่วยแยกทางกันไป

ชั้นทรานสปอร์ตนั้นจะต้องกำหนดชนิดของบริการที่จะบริการให้แก่ชั้นเซสชัน ชนิดของการเชื่อมต่อในชั้นนี้เช่น แบบ ช่องสัญญาณปราศจากความผิดพลาดชนิดจุดต่อจุด (error free point-to-point channel) ซึ่งจะทำการส่งผ่านข้อมูลเรียงตามลำดับตามที่ถูกส่งออกมาจากต้นทาง

ชั้นเซสชัน (session layer)

ชั้นเซสชันนี้จะคอยทำหน้าที่ในการสร้างเซสชันระหว่างผู้ใช้บริการที่อยู่คนละเครื่องคอมพิวเตอร์กันให้เกิดการติดต่อสื่อสารกันได้ ความหมายคือ การที่จะยอมให้มีการลำเลียงข้อมูลเป็นลำดับตามมา เช่นในระบบโทรศัพท์ หน้าที่ในการบริการต่อผู้เรียกโดยส่งสัญญาณไปยังผู้รับ จนผู้รับรับขึ้นมาจนทำให้เกิดการสนทนาเริ่มต้นขึ้นได้ หน้าที่ดังกล่าวจะเป็นหน้าที่ของชั้นเซสชัน แต่หน้าที่ในการส่งสัญญาณแต่ละคำพูดจะเป็นหน้าที่ของชั้นทรานสปอร์ต

บริการอีกอย่างหนึ่งคือ การบริหาร โทเคน (token management) โดยโปรโตคอลบางตัวนั้นจะไม่อนุญาตให้เครื่องคอมพิวเตอร์ทั้งสองด้านที่ทำการติดต่อสื่อสารกันอยู่นั้นทำการปฏิบัติการที่เหมือนกันในเวลาเดียวกันได้ นอกจากนี้จังมีหน้าที่อีกอย่างหนึ่งคือการซิงโครนัสเครื่องคอมพิวเตอร์สองเครื่องเข้าด้วยกันเพื่อให้การติดต่อเป็นไปอย่างคล้องจองกัน

ชั้นพรีเซนเตชัน (presentation layer)

ชั้นพรีเซนเตชันนี้มีหน้าที่จัดการเกี่ยวกับวากยสัมพันธ์ (syntax) และรูปแบบต่าง ๆ ของข่าวสารที่จะถูกส่งจากเครื่องคอมพิวเตอร์ออกไป เช่น การเข้ารหัสข้อมูลให้อยู่ในรูปที่เข้าใจกันทั้งผู้รับและผู้ส่ง ตัวอย่างก็คือเมื่อชั้นนี้รับข้อความจากชั้นแอปพลิเคชันซึ่งเป็นข้อความมาแล้ว ก็จะต้องมาทำการเข้ารหัสอักขระ (character) แต่ละตัวในข้อความนั้นให้อยู่ในรูปที่ทางผู้รับรับแล้วสามารถแปลงกลับเป็นข้อความที่ถูกต้องได้เหมือนเดิม เช่น การเข้ารหัสข้อมูลแบบแอสกี เป็นต้น

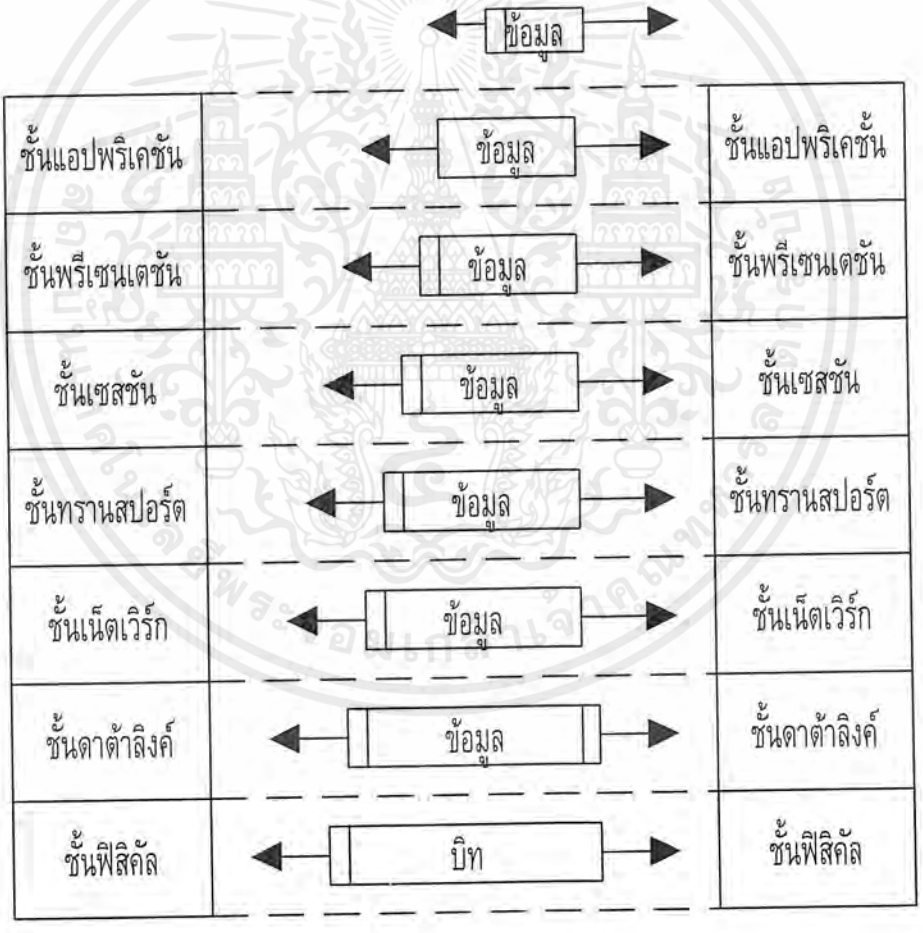
ชั้นแอปพลิเคชัน (application layer)

ชั้นแอปพลิเคชันนี้เป็นชั้นที่มีโปรโตคอลหลายหลากมากมาย โดยที่ชั้นนี้เป็นส่วนที่ติดต่อกับผู้ใช้โดยตรง ได้แก่ โฮลคอมพิวเตอร์ เทอร์มินัลหรือคอมพิวเตอร์ส่วนบุคคล เป็นต้น แอปพลิเคชันในชั้นนี้สามารถนำเข้าหรือออกจากระบบเครือข่ายได้โดยไม่ต้องสนใจว่าจะมีขั้นตอนการทำงานอย่างไร เพราะจะมีชั้นพรีเซนเตชัน รับผิดชอบอยู่แล้ว

2.1.15 การส่งข้อมูลในแบบจำลอง OSI

ในรูปที่ 2.5 แสดงตัวอย่างให้เห็นถึงการส่งผ่านข้อมูลโดยใช้เครือข่ายตามแบบจำลอง OSI โดยเริ่มจากข้อถูกป้อนจากผู้ใช้บริการเข้าไปยังชั้นแอบพลิเคชัน จากนั้นชั้นแอบพลิเคชันก็อาจจะเพิ่มส่วนที่เป็นข้อมูลสำหรับการควบคุมบางอย่างเพิ่มเติมเข้าไปยังข้อมูลเดิมที่ส่วนหัว จากนั้นจึงส่งผ่านต่อลงมายังชั้นพรีเซนเตชันเมื่อชั้นพรีเซนเตชันได้รับข้อมูลก็อาจจะแปลงข้อมูลให้อยู่ในรูปแบบมาตรฐาน เช่น แปลง

ข้อความที่อยู่ในรูปอักษร(text)ให้อยู่ในรูปแอสกี และอาจจะเพิ่มเติมส่วนหัวเข้าไปด้วย แล้วจึงส่งต่อมาให้ชั้นเซสชันนอกจากนี้ตัวของชั้นพรีเซนเตชันเองจะไม่สนใจด้วยว่าข้อมูลที่มันรับมาจากชั้นแอปพลิเคชันนั้นประกอบด้วยอะไรบ้างโดยจะมองข้อมูลที่ส่งออกมาเป็นเนื้อเดียวกันหมดไม่สนว่าอันไหนเป็นข้อมูลจริงอันไหนเป็นส่วนหัวที่เติมเข้ามาในทำนองเดียวกันกับชั้นบนข้อมูลจะถูกส่งผ่านลงมาเรื่อยๆจนถึงชั้นฟิสิคัลซึ่งเป็นชั้นที่จะกระทำให้เกิดการส่งข้อมูลที่แท้จริงเพื่อส่งผ่านไปถึงเครื่องคอมพิวเตอร์ตัวรับ แล้วข้อมูลก็จะถูกส่งย้อนขึ้นไปข้างบนกระบวนการก็จะกระทำย้อนกลับกับตอนที่ส่งมาจนถึงชั้นแอปพลิเคชัน ถึงแม้ว่าการส่งผ่านข้อมูลจริง ๆ นั้นจะกระทำในแนวตั้งลงมา แต่ก็สามารถมองได้เสมือนว่าแต่ละชั้นที่อยู่ระดับเดียวกันรับส่งข้อมูลกันได้โดยตรงตามแนวนอน แนวคิดนี้เป็นจุดประสงค์ในการจะทำให้การนำไปใช้ที่ชั้นต่าง ๆ ง่ายขึ้น โดยเสมือนว่าติดต่อกับชั้นในระดับเดียวกันของเครื่องคอมพิวเตอร์ตัวอื่น โดยไม่ต้องสนใจเลยว่าแท้จริงแล้วข้อมูลจะถูกส่งผ่านไปอย่างไร

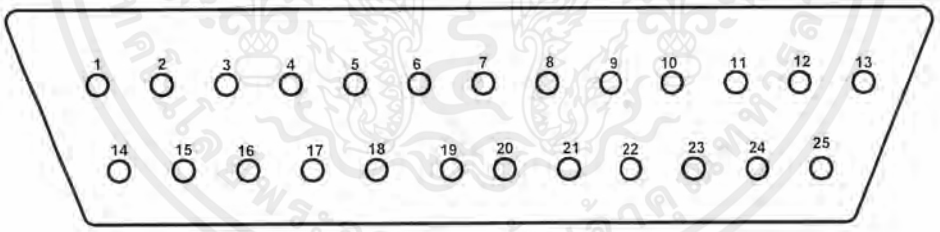


รูปที่ 2.1.23 แสดงการข้อมูลในเลย์เยอร์ต่าง ๆ

2.2 พอร์ตขนาน

พอร์ตขนานหรือพอร์ตเครื่องพิมพ์ของคอมพิวเตอร์มีขั้วต่อเป็นคอนเนคเตอร์แบบ D ขนาด 25 ขา (DB -25) มีการจัดขาสัญญาณดังรูปที่ 2.2.1 และมีหน้าที่การทำงานแสดงในตารางที่ 2.2.2 ภายในพอร์ตขนานประกอบด้วยรีจิสเตอร์พื้นฐานที่ใช้ในการรับและส่งข้อมูล 3 ตัวคือรีจิสเตอร์ DATA,STATUS และ CONTROL โดยแอดเดรสของรีจิสเตอร์ทั้งสามนี้จะมีตำแหน่งไล่เรียงกันไปตามลำดับขึ้นอยู่กับแอดเดรสของพอร์ตขนาน เช่น ที่พอร์ตขนาน LPT1 แอดเดรสของริจิสเตอร์ DATA อยู่ที่ & H 378 (&H เป็นตัวอักษรที่แสดงว่าข้อมูลนี้เป็นข้อมูลเลขฐานสิบหกเมื่อทำการเขียนโปรแกรมด้วย QBASIC และ VISUAL BASIC) ในขณะที่แอดเดรสของรีจิสเตอร์ STATUS จะขณะที่ &H379 และ แอดเตรสของรีจิสเตอร์ CONTROL จะอยู่ที่ &H37A โดยความสัมพันธ์ระหว่างแอดเดรสพอร์ตขนานกับแอดเดรสของริจิสเตอร์พื้นฐานทั้งสามตัวแสดงในตารางที่ 2.2.1

ตำแหน่งของพอร์ตขนานส่วนใหญ่จะมีตำแหน่งพอร์ตเริ่มต้นอยู่ที่ &H378 แต่ก็ไม่ใช่ตำแหน่งสำหรับคอมพิวเตอร์ทุกเครื่อง ดังนั้นวิธีการง่าย ๆ เพื่อตรวจสอบตำแหน่งของพอร์ตขนานที่ใช้งานอยู่สามารถทำได้โดยการใช้โปรแกรม DEBUG แสดงค่าของหน่วยความจำตำแหน่ง 0040:80 ซึ่งใช้สำหรับเก็บค่าแอดเดรสของพอร์ตขนาน LPT1, LPT2 และ LPT3 เอาไว้ดังตัวอย่างในกรอบแยกที่ I



รูปที่ 2.2.1 รูปร่างและตำแหน่งขาของคอนเนคเตอร์พอร์ตขนาน DB-25 ตัวเมีย (มองจากด้านหน้า)

กรอบแยกที่ I

```
C:\dos\debug
-d 0040 : 08
0040 : 0008 78 03 78 02 00 00 00 00
```

จากตัวอย่างข้างต้นพอร์ต LPT1 มีแอดเดรสอยู่ที่ &H378 ส่วนพอร์ต LPT 2 มีแอดเดรสอยู่ที่ &H278 สำหรับ LPT 3 และ LPT4 นั้นไม่ได้ถูกกำหนดไว้ นอกจากการใช้โปรแกรม DEBUG แล้วยังสามารถใช้โปรแกรม MSD.EXE ของ DOS เพื่อตรวจสอบได้อีกด้วย

ตารางที่ 2.2.2 หน้าที่การทำงานของขาต่างๆ ของพอร์ตขนาน

ขา	หน้าที่	พอร์ต		ทิศทาง
		ชื่อ	บิต	
1.	Strobe	Control	Co	เอาต์พุต
2.	Data บิต 0	Data	Do	เอาต์พุต
3.	Data บิต 1	Data	D1	เอาต์พุต
4.	Data บิต 2	Data	D2	เอาต์พุต
5.	Data บิต 3	Data	D3	เอาต์พุต
6.	Data บิต 4	Data	D4	เอาต์พุต
7.	Data บิต 5	Data	D5	เอาต์พุต
8.	Data บิต 6	Data	D6	เอาต์พุต
9.	Data บิต 7	Data	D7	เอาต์พุต
10.	Acknowledge	Status	S6	อินพุต
11.	Busy	Status	S7	อินพุต
12.	Out of Paper	Status	S5	อินพุต
13.	Select	Status	S4	อินพุต
14.	Line Feed	Control	C1	เอาต์พุต
15.	Error	Status	S3	อินพุต
16.	Initial	Control	C2	เอาต์พุต
17.	Select In	Control	C3	เอาต์พุต
18-25	Ground			

พอร์ตที่ทำหน้าที่เป็นเอาต์พุต และพอร์ตที่ทำหน้าที่เป็นอินพุต ดังมีรายละเอียดต่อไปนี้

พอร์ตเอาต์พุต

พอร์ตที่ทำหน้าที่เป็นพอร์ตเอาต์พุตคือ พอร์ต DATA และพอร์ต CONTROL สำหรับพอร์ต DATA ตำแหน่งบิต DO-D7 สามารถใช้งานเป็นเอาต์พุตได้ทั้งหมด ส่วนพอร์ต CONTROL มีบิตที่ทำหน้าที่เป็นเอาต์พุตเพียง 4 บิตคือ C0-C3 ส่วนตำแหน่งอื่น ๆ ไม่ใช้งานหรือถูกสงวนไว้ใช้กับงานอื่นการเขียนข้อมูลไปยังรีจิสเตอร์ DATA นั้นข้อมูลที่ส่งออกไปกับสถานะลอจิกที่ขาพอร์ต DATA จะตรงกัน แต่สำหรับพอร์ต CONTROL นั้นบิต C0, CI และ C3 จะกลับสถานะ โดยเมื่อป้อนค่าหนึ่งค่าใดไปยังรีจิสเตอร์ CONTROL ที่พอร์ต CONTROL จะทำการแปลงข้อมูลที่อยู่ในบิต C0, C1, C3 ให้มีสถานะตรงกันข้าม ดังนั้นเมื่อต้องการส่งข้อมูลออกไปยังพอร์ต CONTROL จะต้องมีการกลับค่าของข้อมูลในบิต C0, C1, และ C3ก่อนเสมอสำหรับโปรแกรมที่ใช้เขียน ข้อมูลไปยังพอร์ตขนานนั้น สามารถเขียนได้โดยใช้ภาษาระดับต่ำเช่น ภาษาแอสเซมบลี้ไปจนถึงภาษาระดับสูงเช่น ภาษาเบสิกเทอร์โบปาสคาล หรือเทอร์โบซี สำหรับตัวอย่างการส่งข้อมูลออกไปยังพอร์ตเอาต์พุตของพอร์ตขนานด้วยการเขียนโปรแกรมภาษาต่าง ๆ มีดังนี้

โปรแกรมภาษาแอสเซมบลี้

```
mov dx, 378h
mov a1, 01
out dx, a1
```

โปรแกรม QBASIC และ VISUAL BASIC (ต้องเพิ่มไฟล์ INP OUT.DLL เมื่อใช้โปรแกรม VISUAL BASIC)

```
BaseAddr = &h378
Bit = 01
Out BaseAddr, Bit
```

โปรแกรมเทอร์โบปาสคาล

```
BaseAddr := 378h
Bit := 01
port[BaseAddr] := Bit
```

โปรแกรมเทอร์โบซี

```
unsigned BaseAddr = 0X378 ; int Dataport ;
Dataport = outp (BaseAddr,0X01) ; return 0 ;
```

พอร์อินพุต

พอร์ตทำหน้าที่เป็นพอร์ตอินพุต คือ พอร์ต STATUS มีบิตที่ใช้งานเป็นอินพุตเพื่อรับข้อมูลจากภายนอก 5 บิตคือ S3 – S7 โดยบิต S7 นั้นมีการกลับสถานะอยู่ดังนั้นการอ่านค่าข้อมูลในบิต 7 จะได้ค่าออกมาเป็นค่าตรงกันข้ามตำแหน่งแอดเดรสของพอร์ต STATUS จะอยู่ถัดจากแอดเดรสของพอร์ต DATA หนึ่งตำแหน่ง ดังนั้นเพื่อเป็นการง่ายต่อการทำความเข้าใจ การเขียนโปรแกรมไปยังพอร์ต STATUS จึงนิยมระบุตำแหน่ง แอดเดรสเป็น BaseAddr+1 (ปกติมีค่าเท่ากับ &H378+1) สำหรับตัวอย่างการอ่านข้อมูลจากพอร์ตอินพุตด้วยการเขียนโปรแกรมภาษาต่าง ๆ มีดังนี้

โปรแกรมภาษาแอสเซมบลี้

```
mov dx, 378h
in a1, dx
```

โปรแกรม QBASIC และ VISUAL BASIC (ต้องเพิ่มไฟล์ INPOUT.DLL เมื่อใช้โปรแกรม VISUALBASIC

```
BaseAddr := &h378
Bit = In (BaseAddr+1)
```

โปรแกรมเทอร์โบปาสคาล

```
BaseAddr := &h378
Bit := port[BaseAddr+1]
```

โปรแกรมเทอร์โบซี

```
ussigned BaseAddr = 0X378 ; int Dataport :
Dataport = inp (BaseAddr+1) ; return 0 ;
```

พอร์ตสำหรับคอมพิวเตอร์รุ่นใหม่

สำหรับในคอมพิวเตอร์รุ่นใหม่ ๆ นั้น พอร์ต DATA สามารถใช้เป็นอินพุตได้ด้วย โดยจะมีบิตที่ทำหน้าที่กำหนดทิศทางของข้อมูลอยู่ที่พอร์ต CONTROL บิต 5 (C5) โดยถ้าบิตนี้เป็น " 0 " จะเป็นการกำหนดให้พอร์ตนี้เป็นเอาต์พุตถ้าบิตนี้เซตเป็น "1 " จะเป็นการดิสเอเบิลเอาต์พุต ทำให้สามารถอ่านสถานะลอจิกจากภายนอกได้ ซึ่งก็คือการทำหน้าที่เป็นอินพุตนั่นเอง

ตารางที่ 2.2.3 แสดงแอดเดรสของรีจิสเตอร์ของพอร์ตขนาน

พอร์ตขนาน	แอดเดรสของ DATA	แอดเดรสของ STATUS	แอดเดรสของ CONTROL
LPT1	&H378	&H379	&H37A
LPT2	&H3BC	&H3BD	&H3BE
LPT3	&H278	&H279	&H27A

บทที่ 3
การคำนวณและการสร้าง

3.1 โปรแกรมควบคุม

โปรแกรมควบคุมประกอบด้วย 3 ส่วนได้แก่

1. โปรแกรมเว็บเซิร์ฟเวอร์
2. โปรแกรมเขียนข้อมูล 1 ไบต์ ออกพอร์ตขนาน
3. โปรแกรมซีจีไอ

โปรแกรมเว็บเซิร์ฟเวอร์

เว็บเซิร์ฟเวอร์ในปัจจุบันมีมากมายหลายแบบทำงานบนระบบปฏิบัติการหลายตัว ตั้งแต่วินโดวส์เอ็นที, ยูนิกซ์, ลีนุกซ์และBeOS โดยแต่ละตัวจะเหมาะสมสำหรับงานแต่ละอย่างหรือบางตัวก็เหมาะกับงานทุกประเภท ซอฟต์แวร์ที่ใช้สร้างเว็บเซิร์ฟเวอร์บางตัวเป็นฟรีแวร์ที่เราสามารถนำมาใช้ได้ฟรีในขณะบางตัวเราต้องซื้อมาใช้แต่ยังคงสามารถดาวน์โหลดเวอร์ชันมาทดลองใช้ได้ในระยะเวลาหนึ่ง

เซิร์ฟเวอร์	จำนวนเซิร์ฟเวอร์ที่ใช้	ส่วนแบ่งตลาด
Apache	8,812,961	61.53%
Microsoft-IIS	3,017,373	21.07%
Netscape-Enterprise	1,002,631	7.00%
Zeus	265,870	1.86%
Rapidsite	256,129	1.79%
Thttpd	214,376	1.50%
WebSitePro	106,300	0.74%
Stronghold	85,586	0.60%
WebSTAR	78,496	0.55%
ConcentriHost-Ashurbanipal	43,780	0.31%

ตารางที่ 3.1 จากผลการสำรวจของ NetCraft ในเดือนเมษายน 2000 ที่ผ่านมา

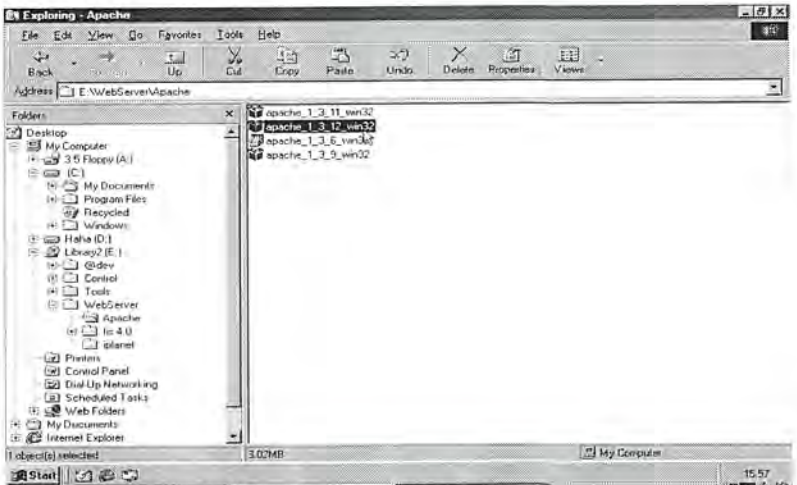
จะเห็นได้ว่าเซิร์ฟเวอร์ที่ได้รับความนิยมสูงสุดสามตัวแรก คือ Apache, IIS ของไมโครซอฟต์ และ Netscape-Enterprise ซึ่งขณะนี้กลายเป็น iPlanet ไปเรียบร้อยแล้ว(หลังจากร่วมมือกับซัน) ซึ่งในโครงงานนี้เราใช้ Apache เป็นเว็บเซิร์ฟเวอร์ เนื่องจากการทำงานที่หน้าเชื่อถือ สมรรถภาพที่โดดเด่น และยังแจกจ่ายซอร์สโค้ดฟรีสำหรับเวอร์ชัน 1.3.0 ที่เปิดตัวไปนั้นได้แสดงให้เห็นว่า Apache เวอร์ชันนี้สามารถทำงานได้อย่างเที่ยงตรงและเร็วที่สุดเท่าที่เคยเป็นมา และในเวอร์ชันนี้ Apache ได้ขยายเครือข่ายให้สามารถทำงานบนวินโดวส์ 95/98/NT

โครงสร้างการทำงานพื้นฐานของ Apache ถือกำเนิดมาจาก HTTPd server ของ NCSA ซึ่งแจกจ่ายฟรี ทำให้ Apache มีความสามารถที่โดดเด่นและความแข็งแกร่ง ดังจะเห็นได้จากความนิยมใช้ทั่วโลกที่มีผู้ใช้มากกว่าครึ่งหนึ่งใช้ Apache เป็นเว็บเซิร์ฟเวอร์ คุณสมบัติที่โดดเด่นของ Apache ได้แก่ การทำงานข้ามแพลตฟอร์ม, การสนับสนุนโปรโตคอล(HTTP/1.1), การทำงานเป็นโมดูล(API), ระบบรักษาความปลอดภัย Apache สามารถทำงานบนวินโดวส์(95/98/NT), OS/2 และ ยูนิกซ์หลายตัวที่ใช้กันอยู่เป็นส่วนใหญ่ โดยทำงานตรงตามมาตราฐาน HTTP/1.1 และสนับสนุน API และ ISAPI(ของวินโดวส์ NT) Apache ประกอบด้วยกลุ่มโมดูลหลักที่ทำหน้าที่จัดการงานทุกอย่างตั้งแต่การตรวจสอบความถูกต้องของผู้ใช้คุกกี้(cookies)ไปจนถึงการแก้ไข URL ที่ระบุผิด นอกจากนี้ยังมีโมดูลให้ทดลองใช้หรือให้ปรับแต่งตามต้องการ

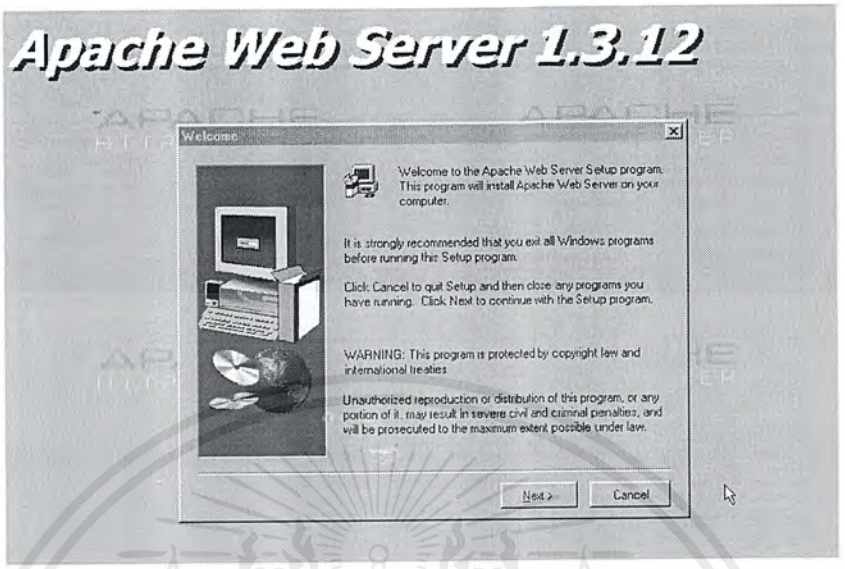
Apache ยังมีจุดเด่นในเรื่องระบบรักษาความปลอดภัย, สมรรถภาพในการทำงานสูงและความแข็งแกร่ง เว็บไซต์ที่มีผู้นิยมเข้าไปเยี่ยมชมมากที่สุดในโลกส่วนใหญ่จะใช้ Apache หรือเซิร์ฟเวอร์ที่มีรากฐานจาก Apache การแจกจ่ายซอร์ดโค้ดทั่วโลกทำให้โปรแกรมแก้ปัญหา(patch)สามารถกระจายได้อย่างรวดเร็วเช่นกัน และยังช่วยให้มีผู้คนทั่วโลกคอยติดตามคุณภาพซอฟต์แวร์และรายงานการข้อผิดพลาดที่เกิดขึ้น ก่อให้เกิดซอฟต์แวร์ที่มีเสถียรภาพและมีระบบความปลอดภัยที่รัดกุม ซึ่งสามารถแข่งขันกับซอฟต์แวร์ธุรกิจตัวอื่นที่ต้องเสียค่าใช้จ่ายในการจัดซื้อมาใช้ทั้งในด้านความเร็วและความสามารถ

วิธีการติดตั้งโปรแกรม Apache

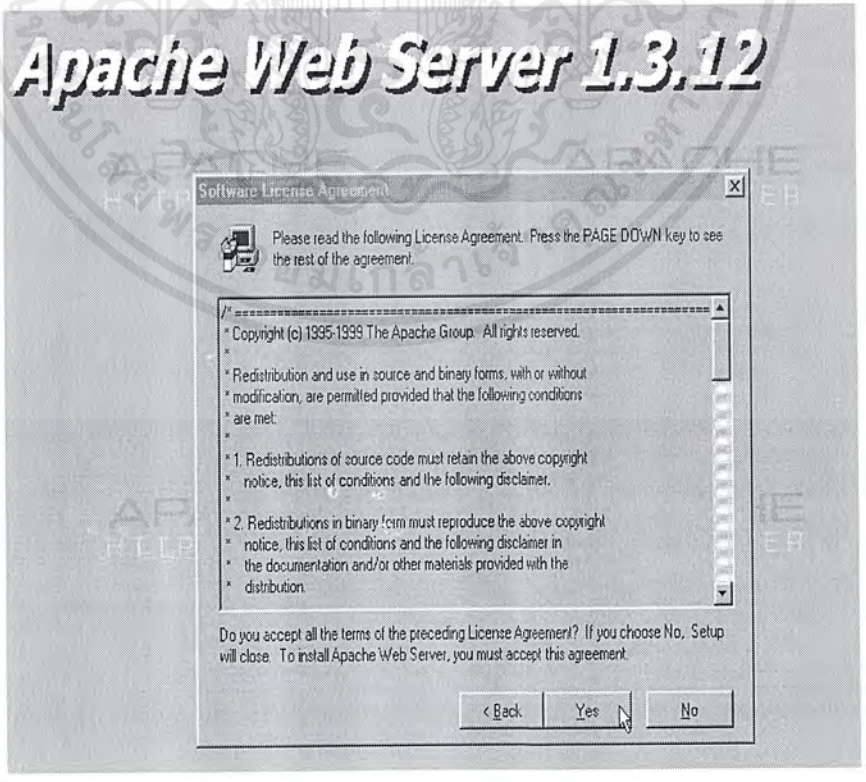
1. เลือก Apache เวอร์ชันที่ต้องการในโครงการนี้เลือกเวอร์ชัน 1.3.12 แล้วดับเบิ้ลคลิก



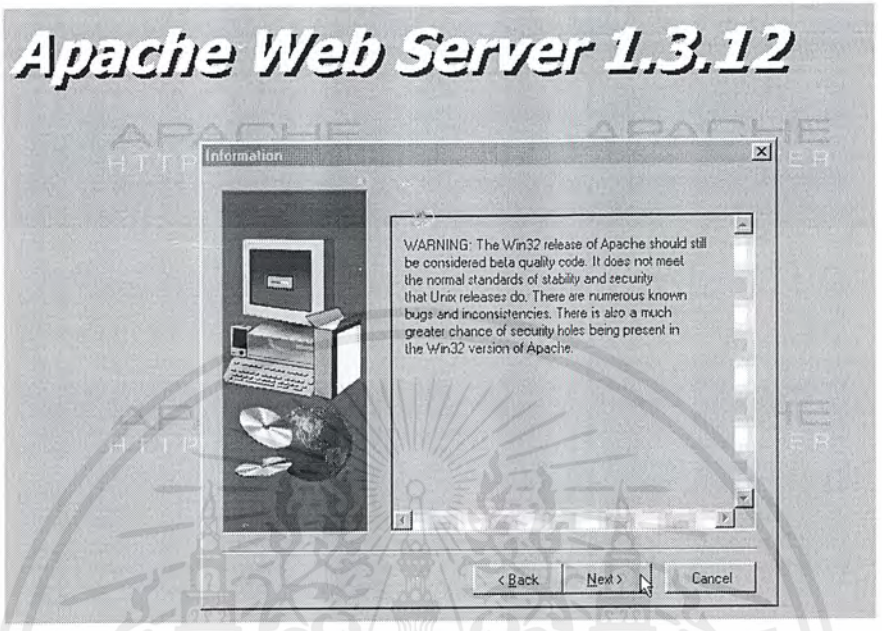
2. คลิก “ NEXT ”



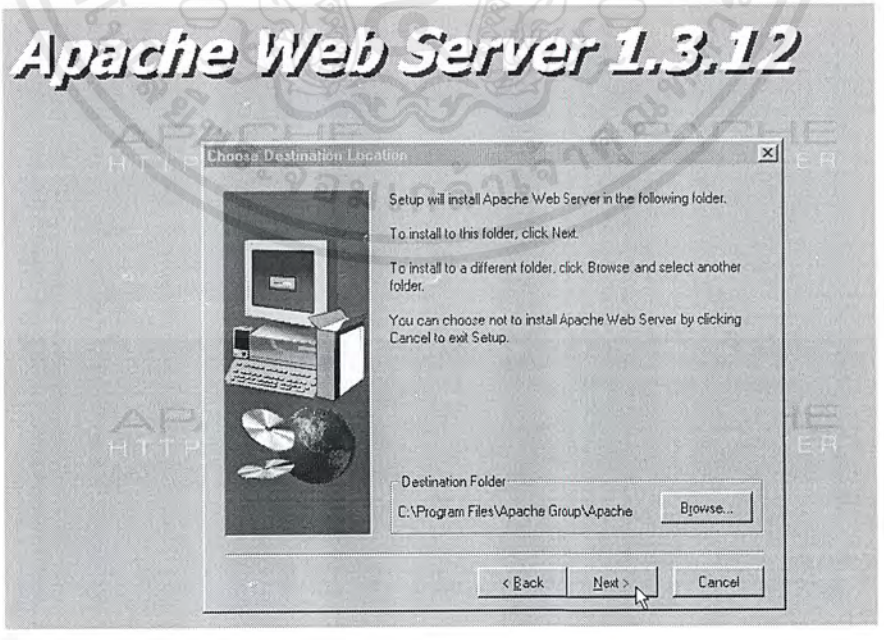
3. คลิก “ Yes ”



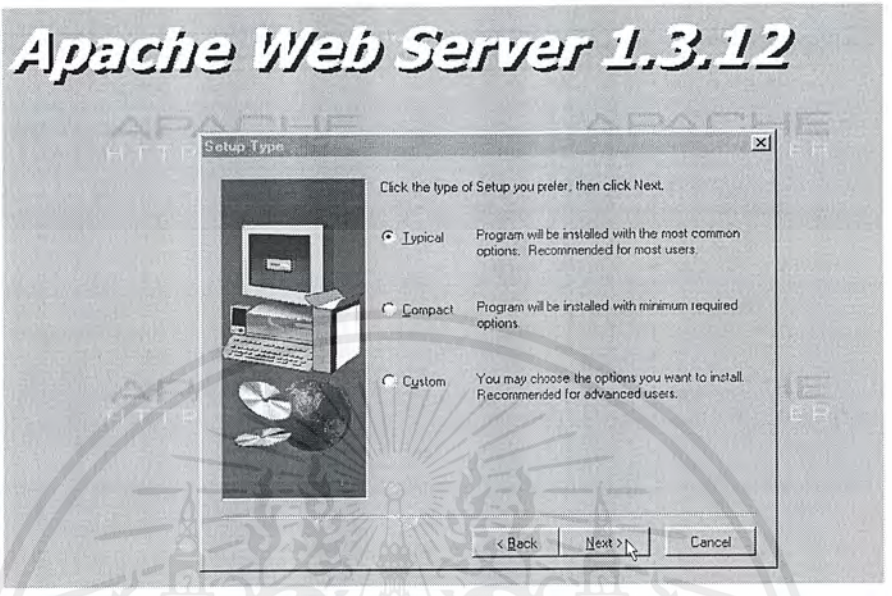
4. คลิก " NEXT "



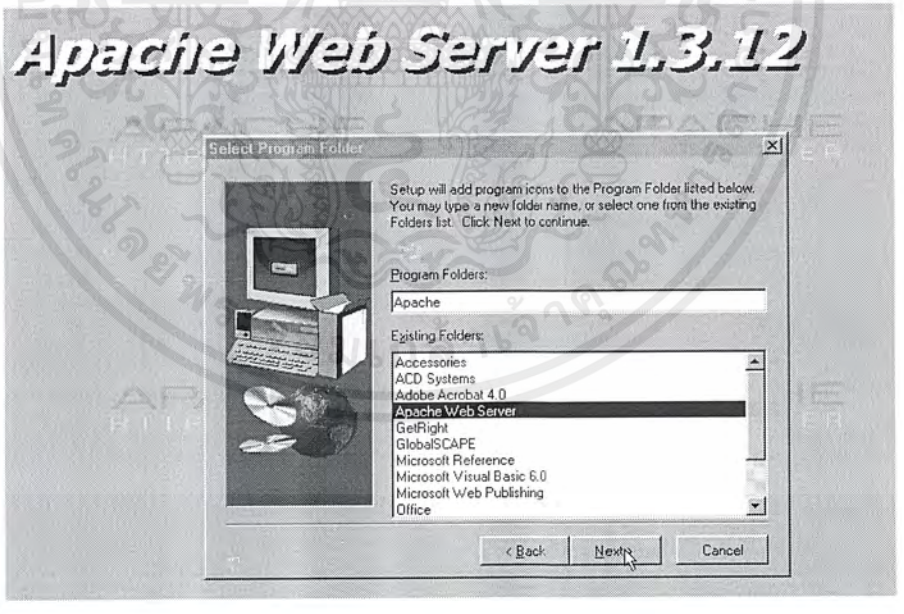
5. เลือกไดเร็กทอรี่ ที่จะติดตั้งแล้วคลิก " NEXT "



6. เลือกไดเร็กทอรี่ ที่จะติดตั้งแล้วคลิก " NEXT "



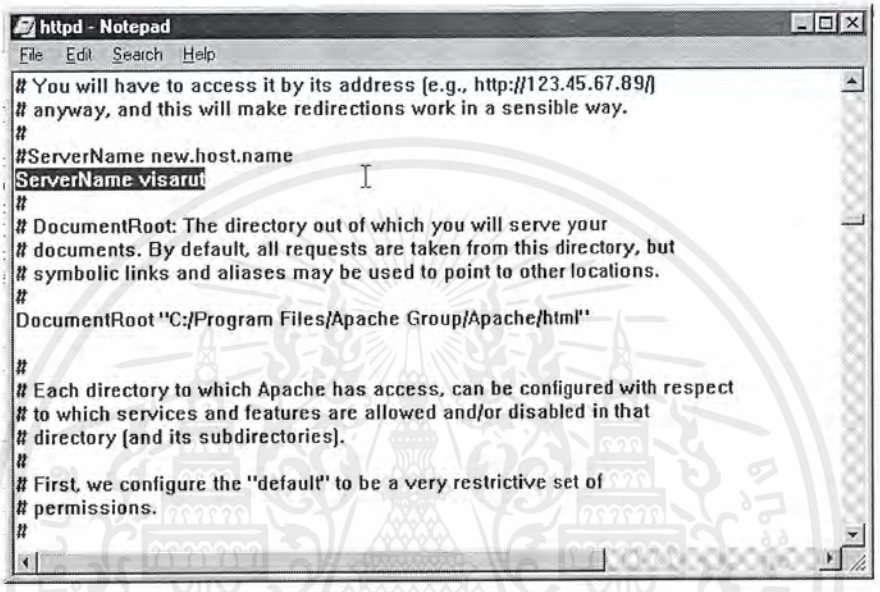
7. ตั้งชื่อโปรแกรมโฟวเดอร์แล้วคลิก " NEXT "



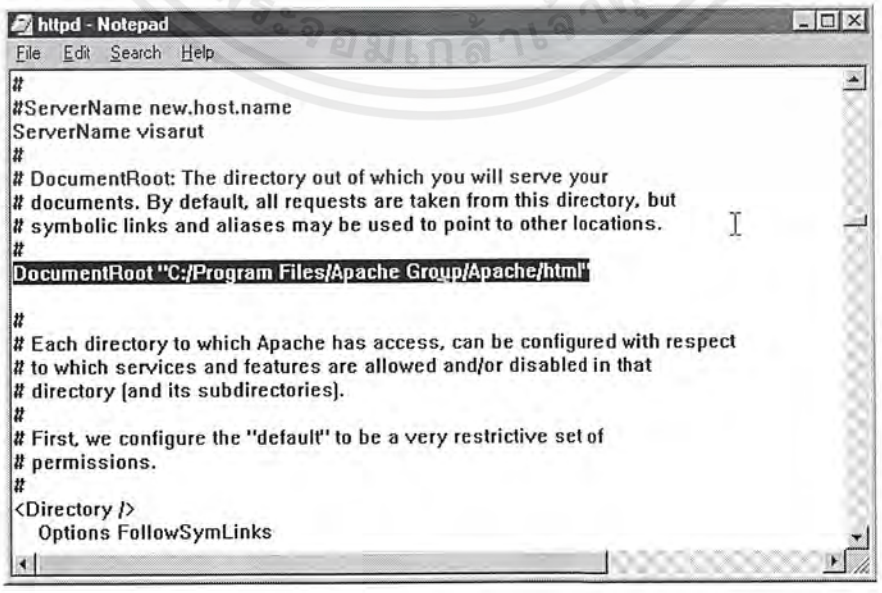
หลังจากติดตั้งโปรแกรม Apache เรียบร้อยแล้วเราจะทำการเซ็ตอัพค่า เพื่อให้สามารถรันโปรแกรมเว็บเซิร์ฟเวอร์ได้

การเซ็ตอัพค่า Apache

1. ไปที่ไฟล์ C:\Program Files\Apache Group\Apache\conf\httpd.conf แล้วใช้โปรแกรม Notepad เปิด จากนั้นตั้งชื่อเซิร์ฟเวอร์ในที่นี้เราตั้งชื่อว่า " visarut " ดังรูป



2. จากนั้นกำหนดไดเร็กทอรี่ที่เก็บไฟล์นามสกุล .html เมื่อรันเซิร์ฟเวอร์จะมาที่ไดเร็กทอรี่นี้เพื่อหาไฟล์ "index.html" ในที่นี้ใช้ไดเร็กทอรี่ "C:/Program Files/Apache Group/Apache/html"



หลังจากเซ็ตอัพค่า Apache แล้ว ก็เป็นการเสร็จสิ้นส่วนของการทำเครื่องคอมพิวเตอร์เป็นเว็บเซิร์ฟเวอร์

โปรแกรมเขียนข้อมูล 1 ไบต์ ออกพอร์ตขนาน

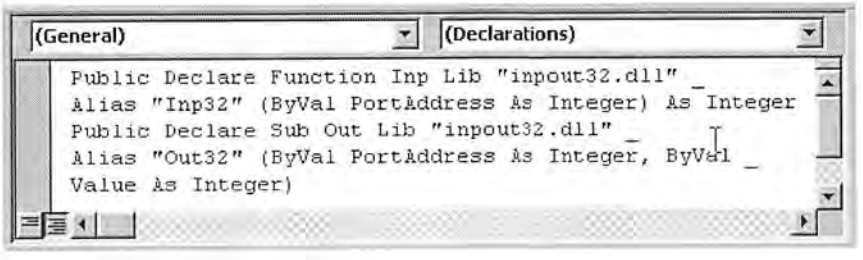
การใช้งาน Visual Basic เพื่อควบคุมพอร์ตต่างๆของคอมพิวเตอร์ซึ่งตามปกติใน Visual Basic ไม่มีคำสั่งเหล่านี้อยู่ จึงใช้วิธีการเรียกไฟล์ DLL ซึ่งเป็นไฟล์ที่เขียนขึ้นด้วยภาษาอื่นแล้วนำมาแทรกให้ Visual Basic ใช้งาน โดยไฟล์ DLL หรือ Dynamic Link Library นั้นสามารถรัน(Execute)ได้ แต่มันจะไม่โหลดลงไปในหน่วยความจำ จนกว่าจะมีโปรแกรมเรียกใช้มันอย่างน้อย 1 ฟังก์ชันใน DLL นั้นวินโดวส์จะรู้ของมันเองด้วยเหตุนี้จึงเรียกว่า Dynamic เมื่อวินโดวส์โหลดโปรแกรมประยุกต์ของ DLL ของมัน ก็จะเกิดการ Link เพื่อทำการเชื่อมต่อระหว่างโปรแกรมกับ DLL และโปรแกรมก็จะรันโดยมี DLL เป็นส่วนหนึ่งของมัน

ทุกโปรแกรมภายใต้วินโดวส์จะสามารถเรียกใช้ DLL โดยโหลดมาที่หน่วยความจำเพียงครั้งเดียว ทำให้ลดปริมาณการใช้หน่วยความจำลงได้ การแก้ไขเพื่อเพิ่มประสิทธิภาพการทำงาน สามารถกระทำเพียงที่เดียวที่โปรแกรม DLL ไม่จำเป็นต้องแก้ที่โปรแกรมหลัก

การเขียนโปรแกรมส่งข้อมูล 1 ไบต์ ออกพอร์ตขนาน ด้วย Visual Basic

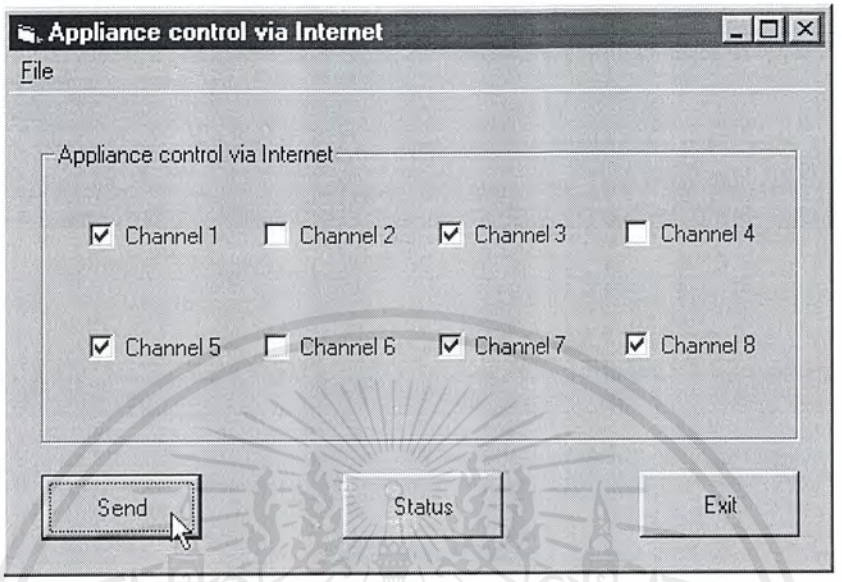
1. เพิ่มไฟล์ Inpout32.dll ลงในโปรแกรม

ในโครงงานเราใช้ไฟล์ Inpout32.dll ซึ่งจะมีคำสั่งการเขียนข้อมูลติดต่อพอร์ต คือ Out PortAddress, Value โดย PortAddress = ค่าของพอร์ต และ Value = ค่าของข้อมูลที่จะส่งออกพอร์ต ซึ่งในการทดลองใช้พอร์ตขนาน จะได้ค่า PortAddress = 378



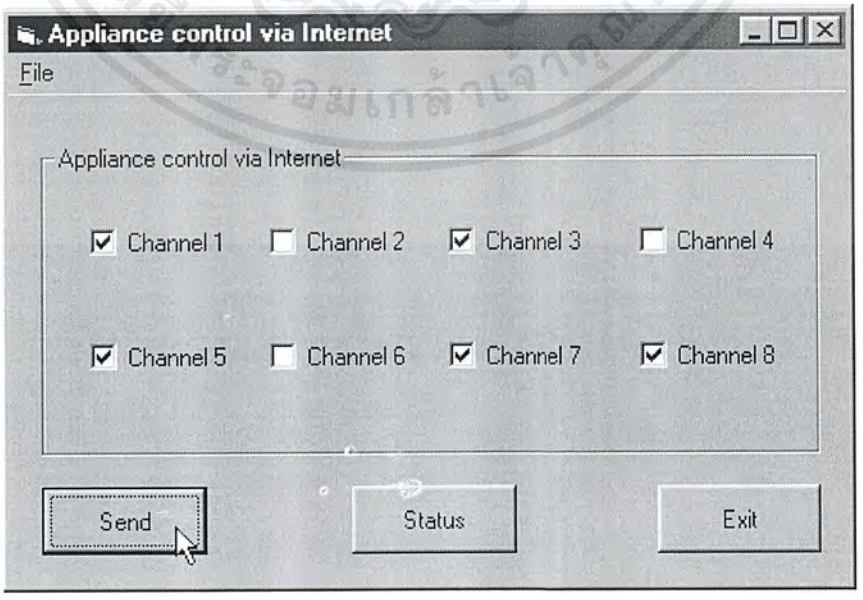
รูป3.1 แสดงการเพิ่มไฟล์ Inpout32.dll ลงในโปรแกรม

2. สร้างฟอร์มสำหรับการส่งข้อมูล

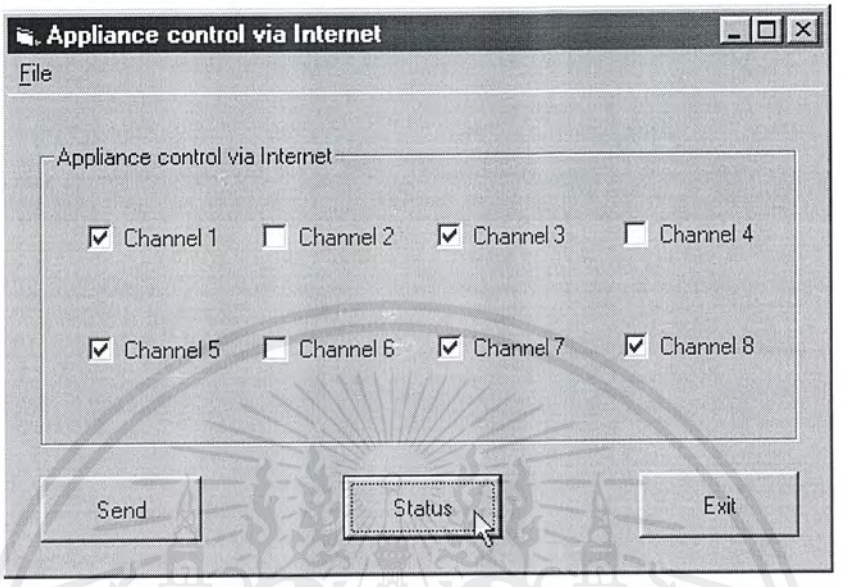


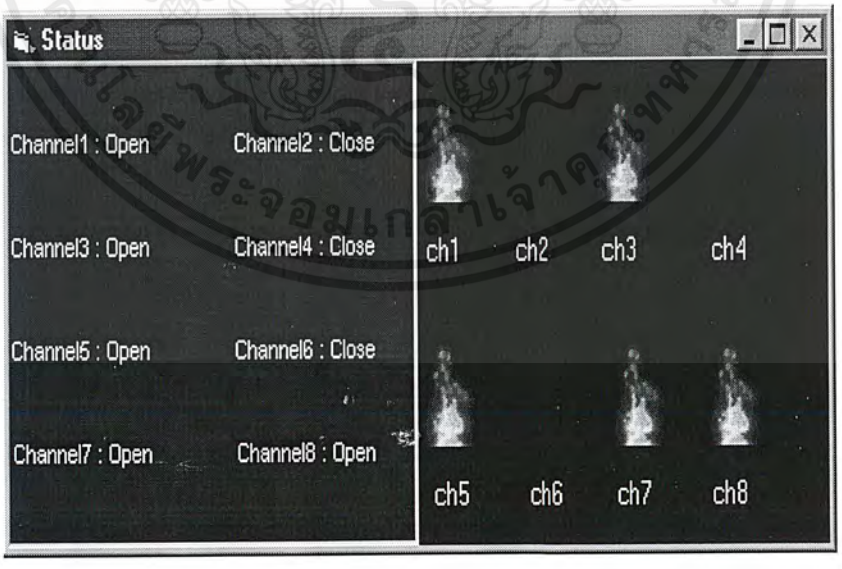
รูปที่ 3.2 แสดงฟอร์มการส่งข้อมูล

3. ทดลองการส่งข้อมูล ส่งข้อมูลออกช่อง 1 , 3 , 5 , 7 , 8 โดยเลือก CheckBox ที่ 1 , 3 , 5 , 7 , 8 แล้วกด Send



4. กดปุ่ม Status เพื่อดูสถานะ





รูปที่ 3.3 แสดงสถานะของข้อมูลที่ส่งไป

โปรแกรมซีจีไอ

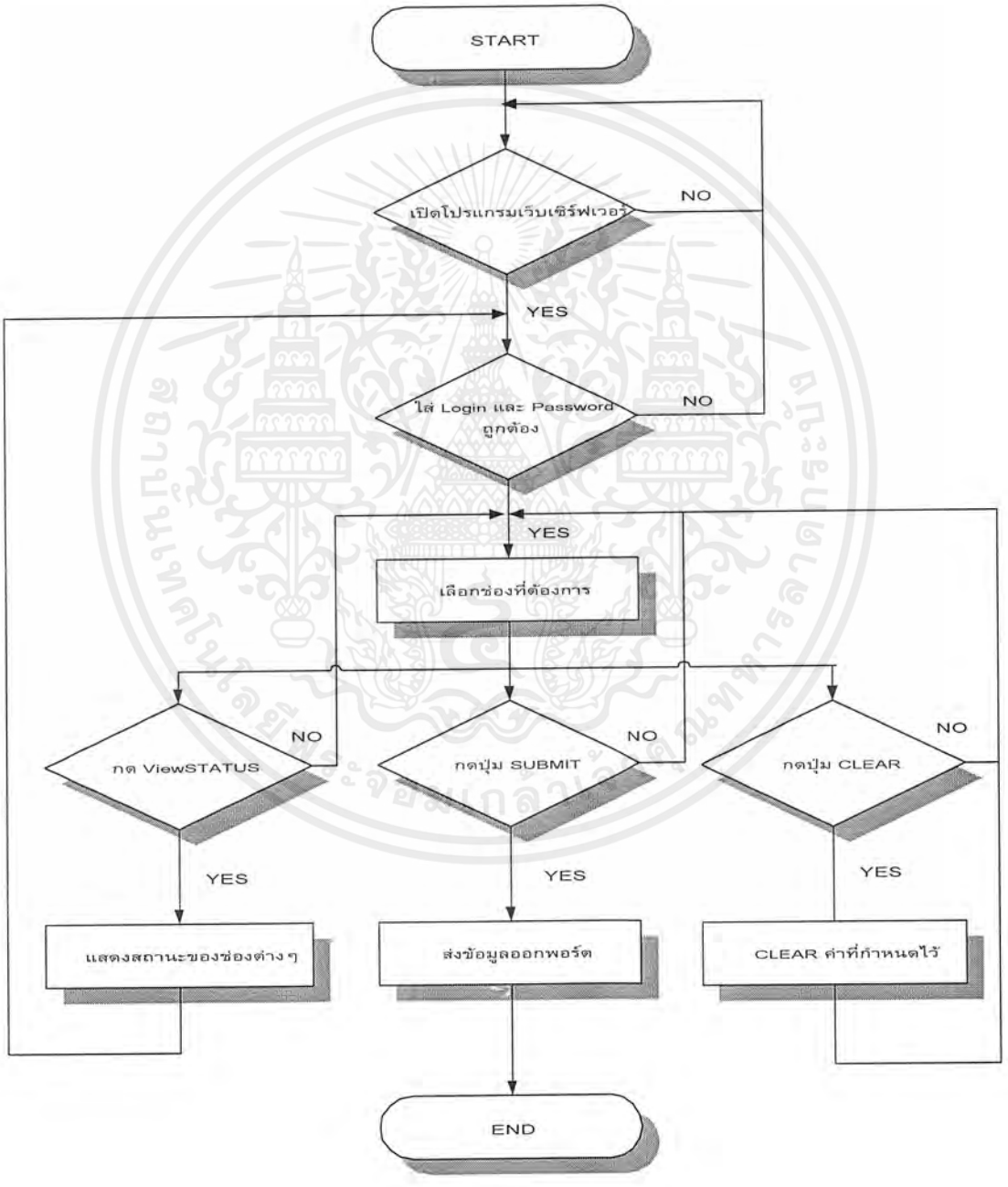
โปรแกรมซีจีไอเป็นโปรแกรมที่ถูกเรียกใช้งานจากเว็บเซิร์ฟเวอร์ ซึ่งมักจะมีขนาดเล็ก ทั้งนี้เพื่อจะสามารถทำงาน และตอบสนองไคลเอ็นท์ได้อย่างรวดเร็ว เว็บเซิร์ฟเวอร์ส่วนใหญ่จะเก็บโปรแกรมซีจีไอไว้ในไดเร็กทอรี่ cgi-bin

โปรแกรมซีจีไอสามารถสร้างได้จากภาษาโปรแกรมหลายภาษา ทั้งภาษา C, Pascal รวมทั้งภาษาสคริปต์ เช่น VBScript, Perl นอกจากนี้เรายังสามารถสร้างได้จากเครื่องมือพัฒนาแอพพลิเคชันแบบใหม่ เช่น Visual Basicหรือ Delphi ได้ด้วย ในโครงงานนี้เราใช้ Visual Basic สร้างเป็นโปรแกรมซีจีไอ

รายชื่อไฟล์ที่สำคัญ

1. Pass.exe เป็นซีจีไอที่คอมไพล์ จาก Visual Basic คอยรับ Login และ Password จากไคลเอ็นท์ แล้วแสดงสถานะในรูปแบบ HTML คืนให้ไคลเอ็นท์
2. Senddata.exe เป็นซีจีไอที่คอมไพล์ จาก Visual Basic คอยรับข้อมูลการควบคุมช่องของเครื่องใช้ไฟฟ้า พร้อมทั้งส่งข้อมูลออกพอร์ตขนาน แล้วแสดงสถานะในรูปแบบ HTML คืนให้ไคลเอ็นท์
3. Status.exe เป็นซีจีไอที่คอมไพล์ จาก Visual Basic เป็นโปรแกรมที่อ่านสถานะ แล้วแสดงสถานะในรูปแบบ HTML คืนให้ไคลเอ็นท์
4. Status.txt เป็นไฟล์ที่เก็บสถานะการส่งข้อมูล
5. Index.html เป็นไฟล์โฮมเพจหลัก ของเซิร์ฟเวอร์
6. pass.html เป็นไฟล์โฮมเพจที่รับ Login และ Password ของไคลเอ็นท์(เป็น HTML)
7. flash.html เป็นไพล์โฮมเพจที่รับ Login และ Password ของไคลเอ็นท์(เป็น flash)
8. Pass.swf เป็นไพล์ flash Movie ที่รับ Login และ Password ของไคลเอ็นท์
9. Checkbox.swf เป็นไพล์ flash Movie ที่ส่งข้อมูลการควบคุมช่องของเครื่องใช้ไฟฟ้า
10. Inpout32.dll เป็นไฟล์ที่ทำหน้าที่ทำให้ Visual Basic เชื่อมต่อกับพอร์ตของคอมพิวเตอร์ได้

รูปที่3.4 โฟลวชาร์ทแสดงขั้นตอนการทำงานของโครงงาน



3.2 ส่วนควบคุมอุปกรณ์ไฟฟ้า

3.2.1 วงจรควบคุมการเปิด-ปิดอุปกรณ์ไฟฟ้า

ทำหน้าที่ควบคุมการเปิด-ปิดอุปกรณ์ไฟฟ้า โดยรับคำสั่งการเปิด-ปิดอุปกรณ์ไฟฟ้า ในแต่ละช่อง จาก คอมพ์พอร์ต ของเซิร์ฟเวอร์

วงจรนี้ใช้ออฟโต้คัพเปลอร์ประเภทไดแอค (Moc3021) เป็นการเชื่อมโยงทางแสงเพื่อควบคุมไตรแอคอีกทีหนึ่ง มีวงจรดังรูป 3.4.1

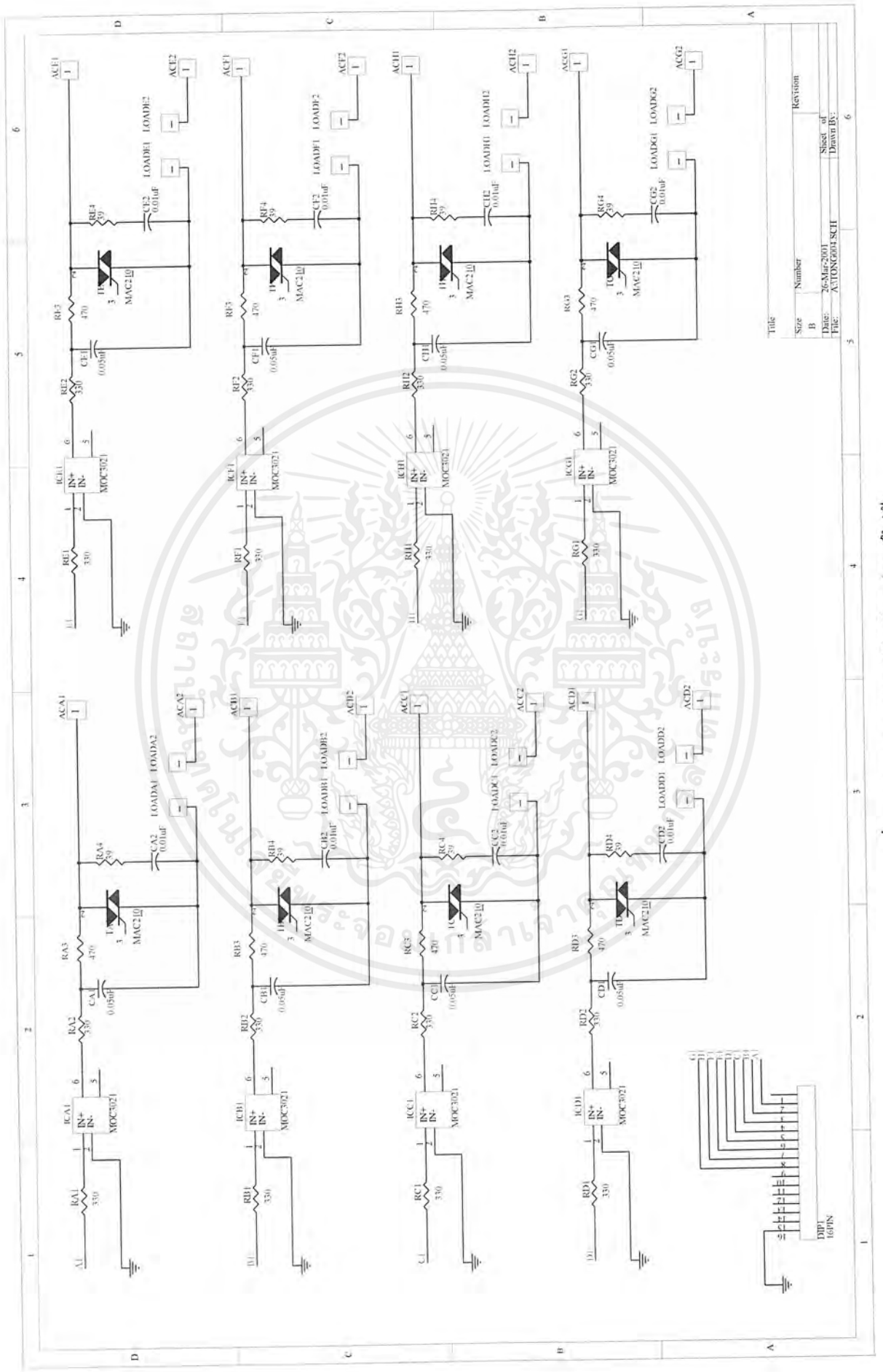
การเชื่อมต่อทางแสงมีข้อดีทำให้แรงดันไฟสูง (220 V) แยกกับแรงดันไฟต่ำ อย่างเด็ดขาด ทำให้เกิดความปลอดภัย ในแง่ของวงจรจะไม่เสียหายง่าย และในแง่ของการใช้งานจะปลอดภัยมาก ซึ่งโดยปกติที่ไม่ได้ใช้การเชื่อมโยงทางแสงจะมีไฟสูงส่วนหนึ่งต่ออยู่กับกราวนด์และอาจทำให้ย้อนกลับไปที่กราวนด์ของสัญญาณอินพุท ทำให้กราวนด์ทั้งระบบที่ต่อถึงกันอาจทำอันตรายได้เมื่อสัมผัสถูกตัวเครื่อง ไดแอคที่ออปโต้คัพเปลอร์ จะควบคุมที่ขาเกทของไตรแอค ให้ไตรแอคทำงานเมื่อมีสัญญาณมีอุปกรณ์ไฟฟ้าก็จะทำงาน

3.2.2 วงจรตรวจสอบสภาวะการทำงานของอุปกรณ์ไฟฟ้า

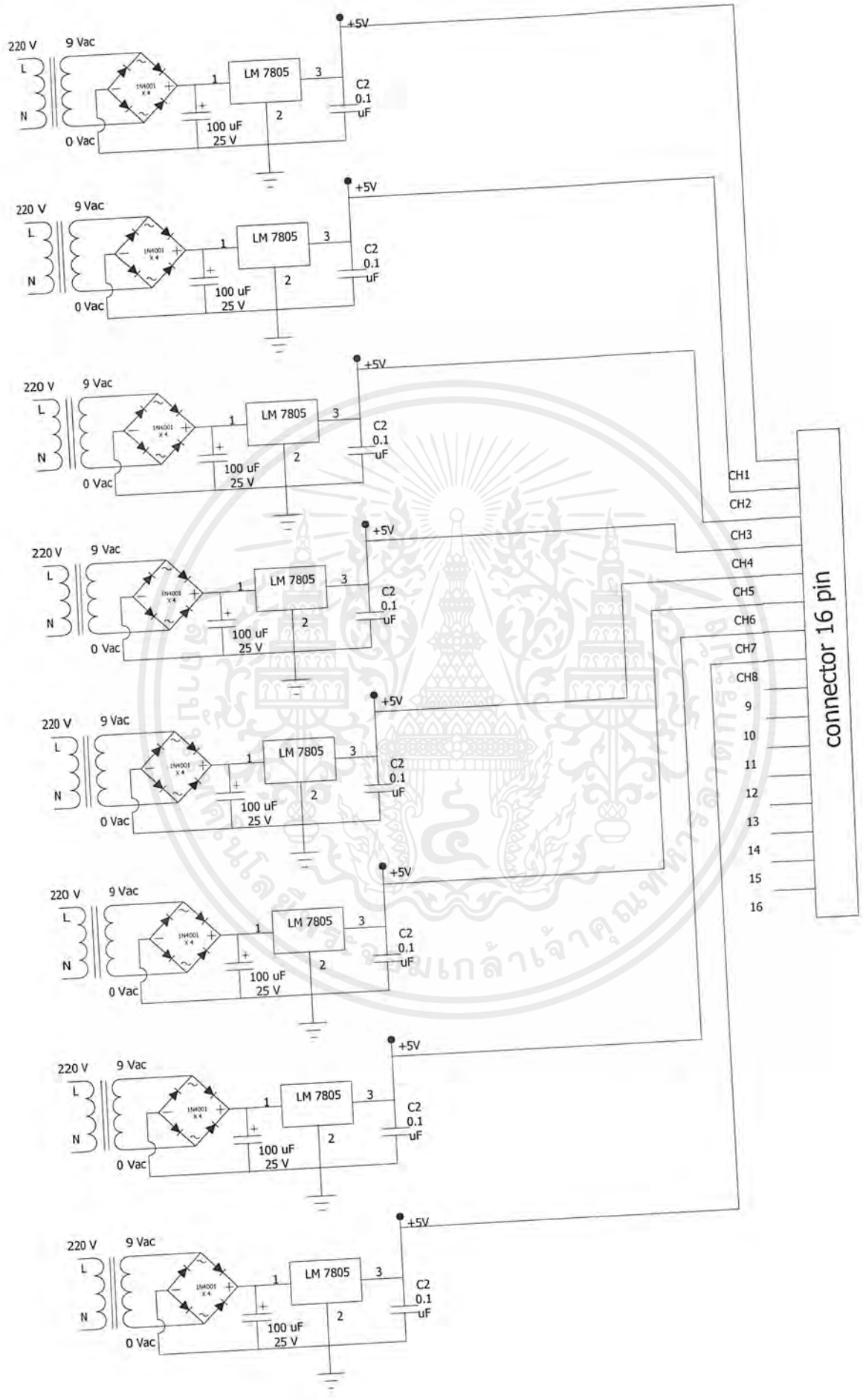
ทำหน้าที่ตรวจสอบสภาวะการทำงานของอุปกรณ์ไฟฟ้า แล้วส่งสัญญาณซึ่งเป็นสถานะการทำงานของอุปกรณ์ แต่ละช่องไปแสดงผลที่เวปเพจ เพื่อให้ผู้ใช้ได้ทราบสถานะการทำงานของอุปกรณ์ไฟฟ้าในแต่ละช่องเพื่อทำการสั่งการควบคุมต่อไป

เนื่องจากเราใช้ IC 8255 เป็นตัวรับค่าที่แสดงสถานะการทำงานของอุปกรณ์ไฟฟ้าโดยสัญญาณ 5 Vdc จะแสดงสถานะเปิด และแรงดัน 0 โวลต์แสดงสถานะปิดของอุปกรณ์ไฟฟ้า วงจรซึ่งแสดงดังรูปที่ 3.4.2 ใช้ ไอซีเบอร์ LM 7805 เป็นไอซีเรคกูเลเตอร์ แรงดันไฟฟ้า 5 โวลต์ ทางด้านขดไพรมารี่ของหม้อแปลงไฟฟ้า จะนำไปต่อขนานกับปลั๊กที่เชื่อมต่ออุปกรณ์ไฟฟ้า ในแต่ละช่องเมื่อผู้ใช้ส่งคำสั่งให้อุปกรณ์ในช่องเปิด ไตรแอกในวงจรควบคุมการเปิด-ปิด อุปกรณ์ไฟฟ้าก็จะทำงาน ทำให้อุปกรณ์ไฟฟ้าและหม้อแปลงไฟฟ้า ได้รับแรงดันไฟฟ้าครบ 1 เฟส อุปกรณ์ไฟฟ้าในช่องที่เชื่อมต่อนั้นก็จะทำงาน วงจรตรวจสอบสภาวะการทำงานของอุปกรณ์ไฟฟ้านี้ก็จะส่งสัญญาณไฟตรง 5 Vdc ไปยัง อินพุตพอร์ตของ 8255 เพื่อนำไปประมวลผล แล้วนำไปแสดงผลให้ผู้ใช้ได้ทราบสถานะการทำงานของอุปกรณ์ไฟฟ้า ในขณะที่ผู้ใช้กำลังจะส่งสัญญาณไปควบคุมอุปกรณ์ไฟฟ้า

วงจรตรวจจับกระแสโหลดมีดังรูป 3.4.2 ใช้หลักการทำงานของการเหนี่ยวนำกระแสที่ไหลผ่านไปยังอุปกรณ์ไฟฟ้า แรงดันที่ตกคร่อมที่ขดเซคคันดารี่ จะมีค่าน้อยมาก จึงต้องทำการขยายแรงดันนี้ให้มีขนาด 5 Vdc เพื่อส่งสัญญาณนี้ให้กลับเวปเซิร์ฟเวอร์ นำไปแสดงผลให้กับผู้ใช้เพื่อที่จะได้ทราบสถานะการทำงาน ของอุปกรณ์ไฟฟ้าในขณะนั้น



รูปที่ 3.4.1 วงจรควบคุมการปิดเปิดอุปกรณ์ไฟฟ้า



รูปที่ 3.4.2 วงจรตรวจสอบอุปกรณ์ไฟฟ้า

3.2.3 ISA CARD อเนกประสงค์

ในโครงงานเราได้ใช้ ISA CARD เพื่อทำการเพิ่มช่องสัญญาณในการควบคุม ซึ่งเพิ่มจากเดิม 8 ช่อง สัญญาณ เป็น 24 ช่องสัญญาณ และยังสามารถเป็นแนวทางนำไปประยุกต์ใช้งานได้

ลักษณะการทำงาน ISA CARD

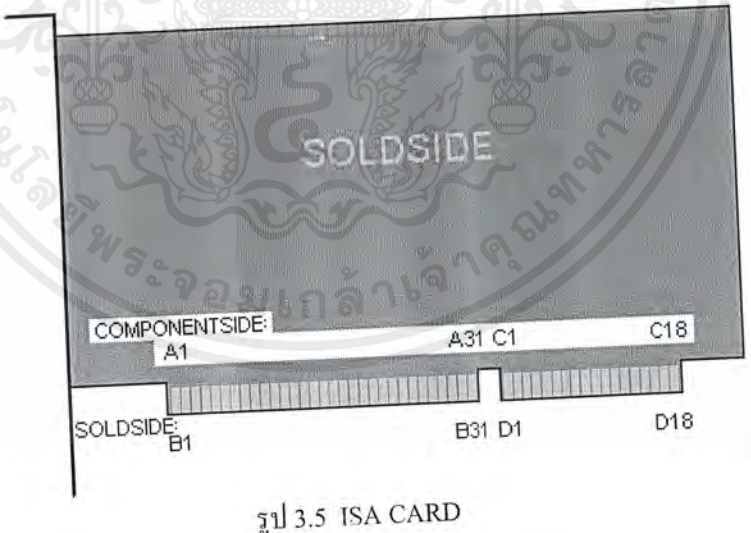
1. เป็น ISA card อเนกประสงค์โดยใช้หลักการ Mother-Daughter Board โดยจะแบ่งการทำงานเป็นสองส่วนคือส่วนที่เป็นส่วนหลัก (Mother Board) ทำหน้าที่ในส่วนของการเป็น Address Decodeซึ่งเป็นส่วนพื้นฐานในการทำงานเพื่อรองรับการเชื่อมต่อของDaughter board และส่วนย่อย(Daughter Board) คือส่วนที่เพิ่มเติม เพื่อการใช้งานเฉพาะทางผู้ใช้สามารถสร้างให้เหมาะสมกับงานที่ต้องการง่าย

2. เป็น ISA card ที่สามารถเลือก Address ทำงานได้ 8 ช่วงค่าโดยใช้ dip switch เป็นตัวเลือกเพื่อไม่ให้ค่า Address ซ้ำกับอุปกรณ์ตัวอื่นที่ติดตั้งไปแล้ว

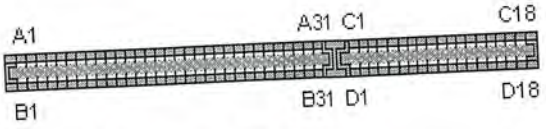
3. ในแต่ละช่วงค่าของ Base Address สามารถแบ่งเป็น Address ย่อยเพื่อติดต่อกับอุปกรณ์อื่น อีกได้ถึง 64 อุปกรณ์

4. มีราคาถูก เพราะผู้ใช้สามารถสร้างอุปกรณ์เพิ่มเติมให้เหมาะสมกับงานที่ต้องการได้จริง และสามารถนำไปประยุกต์ใช้งานได้หลากหลาย

5. ง่ายในการติดต่อ สามารถใช้การเขียนโปรแกรมควบคุม Hardware ได้โดยตรง



รูป 3.5 ISA CARD



รูป 3.6 ISA SLOT ON PC

ตารางที่ 3.2 แสดงขาสัญญาณต่างๆ

A1	/I/O CH CK	←	I/O channel check; active low=parity error
A2	D7	↔	Data bit 7
A3	D6	↔	Data bit 6
A4	D5	↔	Data bit 5
A5	D4	↔	Data bit 4
A6	D3	↔	Data bit 3
A7	D2	↔	Data bit 2
A8	D1	↔	Data bit 1
A9	D0	↔	Data bit 0
A10	I/O CH RDY	←	I/O Channel ready, pulled low to lengthen memory cycles
A11	AEN	→	Address enable; active high when DMA controls bus
A12	A19	→	Address bit 19
A13	A18	→	Address bit 18
A14	A17	→	Address bit 17
A15	A16	→	Address bit 16
A16	A15	→	Address bit 15
A17	A14	→	Address bit 14
A18	A13	→	Address bit 13
A19	A12	→	Address bit 12
A20	A11	→	Address bit 11
A21	A10	→	Address bit 10
A22	A9	→	Address bit 9
A23	A8	→	Address bit 8
A24	A7	→	Address bit 7
A25	A6	→	Address bit 6
A26	A5	→	Address bit 5
A27	A4	→	Address bit 4
A28	A3	→	Address bit 3
A29	A2	→	Address bit 2
A30	A1	→	Address bit 1
A31	A0	→	Address bit 0
B1	GND		GND
B2	RESET	→	Active high to reset or initialize system logic
B3	+5V		+5 VDC
B4	IRQ2	←	Interrupt Request 2

ตารางที่ 3.3 แสดงขาสัญญาณต่างๆ

B6	DRQ2	←	DMA Request 2
B7	-12VDC		-12 VDC
B8	/NOWS	←	No WaitState
B9	+12VDC		+12 VDC
B10	GND		Ground
B11	/SMEMW	→	System Memory Write
B12	/SMEMR	→	System Memory Read
B13	/IOW	→	I/O Write
B14	/IOR	→	I/O Read
B15	/DACK3	→	DMA Acknowledge 3
B16	DRQ3	←	DMA Request 3
B17	/DACK1	→	DMA Acknowledge 1
B18	DRQ1	←	DMA Request 1
B19	/REFRESH	↔	Refresh
B20	CLOCK	→	System Clock (67 ns, 8-8.33 MHz, 50% duty cycle)
B21	IRQ7	←	Interrupt Request 7
B22	IRQ6	←	Interrupt Request 6
B23	IRQ5	←	Interrupt Request 5
B24	IRQ4	←	Interrupt Request 4
B25	IRQ3	←	Interrupt Request 3
B26	/DACK2	→	DMA Acknowledge 2
B27	T/C	→	Terminal count; pulses high when DMA term. count
B28	ALE	→	Address Latch Enable
B29	+5V		+5 VDC
B30	OSC	→	High-speed Clock (70 ns, 14.31818 MHz, 50% duty
B31	GND		Ground
C1	SBHE	↔	System bus high enable (data available on SD8-15)
C2	LA23	↔	Address bit 23
C3	LA22	↔	Address bit 22
C4	LA21	↔	Address bit 21
C5	LA20	↔	Address bit 20
C6	LA18	↔	Address bit 19
C7	LA17	↔	Address bit 18
C8	LA16	↔	Address bit 17
C9	/MEMR	↔	Memory Read (Active on all memory read cycles)

ตารางที่ 3.4 แสดงขาสัญญาณต่างๆ

C10	/MEMW	↔	Memory Write (Active on all memory write cycles)
C11	SD08	↔	Data bit 8
C12	SD09	↔	Data bit 9
C13	SD10	↔	Data bit 10
C14	SD11	↔	Data bit 11
C15	SD12	↔	Data bit 12
C16	SD13	↔	Data bit 13
C17	SD14	↔	Data bit 14
C18	SD15	↔	Data bit 15
D1	/MEMCS	←	Memory 16-bit chip select (1 wait, 16-bit memory
D2	/IOCS16	←	I/O 16-bit chip select (1 wait, 16-bit I/O cycle)
D3	IRQ10	←	Interrupt Request 10
D4	IRQ11	←	Interrupt Request 11
D5	IRQ12	←	Interrupt Request 12
D6	IRQ15	←	Interrupt Request 15
D7	IRQ14	←	Interrupt Request 14
D8	/DACK0	→	DMA Acknowledge 0
D9	DRQ0	←	DMA Request 0
D10	/DACK5	→	DMA Acknowledge 5
D11	DRQ5	←	DMA Request 5
D12	/DACK6	→	DMA Acknowledge 6
D13	DRQ6	←	DMA Request 6
D14	/DACK7	→	DMA Acknowledge 7
D15	DRQ7	←	DMA Request 7
D16	+5 V		
D17	/MASTE	←	Used with DRQ to gain control of system
D18	GND		Ground

หน้าที่ของขาสัญญาณที่สำคัญ

RESET DRV(B2)
คือสัญญาณเอาต์พุต " 1 " เมื่อจ่ายไฟให้ระบบจะเป็น "0 "

IOR(B14)
คือสัญญาณเอาต์พุต " 0 " เพื่อแสดงว่าบัสไซเคิลของ I/O นี้เป็นการอ่านข้อมูล ที่แอสเดรสตรงกับบัสส่งข้อมูลมายังบัส

IOW(B13)
คือสัญญาณเอาต์พุต " 0 " เพื่อแสดงว่าบัสไซเคิลของ I/O นี้เป็นการเขียนข้อมูล ที่แอสเดรสตรงกับบัสส่งข้อมูลมายังบัส

SA0-SA19(Address Bus A12-A31)
คือสัญญาณเอาต์พุต 20 ขา ใช้สำหรับกำหนดแอสเดรส

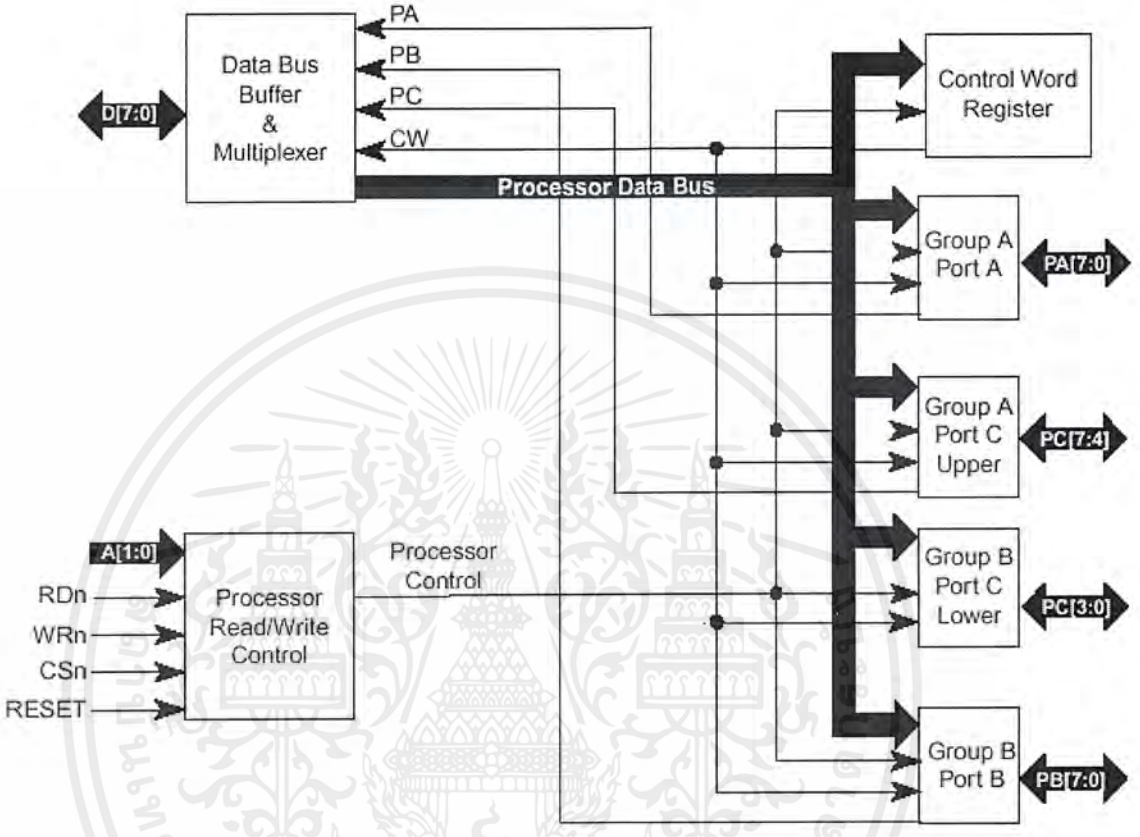
SD0-SD7
(Data Bus A2-A9) คือสัญญาณแบบ Bi-Directional ทำหน้าที่ส่งผ่านข้อมูล

การอินเตอร์เฟสเพื่อใช้งาน 8255 กับ ISA CRAD

คุณสมบัติของ 8255 A

1. มี Input / Out Put24 ขาที่สามารถโปรแกรมได้
2. ระดับสัญญาณเดียวกันกับ TTL
3. คุณสมบัติเหมือนกับไมโครโปรเซสเซอร์ตระกูลอินเทล
4. สามารถเซตและรีเซตบิตได้โดยตรง 8255A สามารถโปรแกรมให้เป็นอินพุตหรือเอาต์พุตก็ได้ สามารถกำหนดโหมดการทำงานได้ 3 โหมด คือโหมด 0, โหมด 1 และ โหมด 2

รูปที่3.7 บล็อกไดอะแกรม 8255



สามารถอธิบายหน้าที่การทำงานของแต่ละส่วนได้ดังต่อไปนี้

Data Bus Buffer

เป็นบัฟเฟอร์ 8 บิต สองทิศทาง 3 สถานะ (3-state) ใช้ในการอินเตอร์เฟสกับบัสข้อมูลของระบบ (system data bus) ข้อมูลจะถูกส่งหรือรับโดยทางบัฟเฟอร์ขึ้นอยู่กับการประมวลผลของพีซียูว่าให้ทำการส่งหรือรับข้อมูล

Read/Write and Control Logic

ทำหน้าที่จัดการเกี่ยวกับการส่งผ่านข้อมูลและ control หรือ status ทั้งภายในและภายนอก ส่วนนี้จะรับอินพุตจากบัสแอดเดรส และบัสควบคุมของซีพียูแล้วมาถอดรหัสในการควบคุมจากภายนอก ดังต่อไปนี้

Chip Select(CS)

เมื่อขานี้ได้รับลอจิก "0" จะทำให้ 8255A ต่อเข้ากับระบบบัสของซีพียู เพื่อให้ซีพียูอ่านหรือเขียนข้อมูลผ่านพอร์ตได้

Read(RD)

เป็นขาสัญญาณอินพุตที่ต้องมาจากซีพียู เมื่อสัญญาณขานี้มีลอจิก "0 " และ CS เป็น "0" ด้วยซีพียูจะทำการอ่านข้อมูลจากบัสข้อมูลของ 8255A

Write (WR)

ขาสัญญาณการเขียนจะแอกทีฟเมื่อ CS เป็น "0" และ WR เป็น "0" โดยสัญญาณนี้จะถูกส่งมาจากซีพียู สามารถเขียนข้อมูลบนบัสข้อมูลของ 8255A ได้

Port Select 0 and Port Select 1

สัญญาณอินพุตสองขานี้จะต้องสัมพันธ์กับสัญญาณ RD , WR และ CS เพื่อใช้ในการเลือกพอร์ตใช้งาน รายละเอียดแสดงดังตาราง

Reset (RESET)

แอกทีฟ "1" สำหรับขานี้ เมื่อมีลอจิก "1" เข้ามาจะทำให้เกิดการเคลียร์รีจิสเตอร์ควบคุม (control register) และทุกพอร์ตจะถูกเซตสู่โหมดอินพุต

Group A and Group B controls

แต่ละบล็อกจะรับคำสั่งจาก read/write control logic และรับ control word จากบัสข้อมูลภายใน เพื่อใช้ในการควบคุมพอร์ต

- Control Group A – จะควบคุมพอร์ต A และพอร์ต C บน (PC_4-PC_7)

- Control Group B – จะควบคุมพอร์ต B และพอร์ต C ล่าง (PC_0-PC_3)

Port A, B and C 8255A

ประกอบด้วยพอร์ต 8 บิต 3 พอร์ต (พอร์ต A , B และ C) ซึ่งพอร์ตทั้ง 3 นี้สามารถกำหนดรูปแบบการใช้งานได้ โดยอาศัยซอฟแวร์ช่วยจัดการ

A1	A0	RD	WR	CS	กระบวนการทางอินพุต (อ่าน)
0	0	0	1	0	PORT A - DATA BUS
0	1	0	1	0	PORT B - DATA BUS
1	0	0	1	0	PORT C - DATA BUS
					การบวนการทางเอาต์พุต (เขียน)
0	0	1	0	0	DATA BUS - PORT A
0	1	1	0	0	DATA BUS - PORT B
1	0	1	0	0	DATA BUS - PORT C
1	1	1	0	0	DATA BUS - CONTROL
					ฟังก์ชันยกเลิกการทำงาน
X	X	X	X	1	DATA BUS - 3 - STAGE
1	1	0	1	0	ELEGE CONDITION
X	X	1	1	0	DATA BUS - STATE

ตารางที่ 3.5 รายละเอียดการเลือกใช้พอร์ต

การทำงานของวงจร

โครงงานนี้ใช้ไอซี 8255A 2 ตัว สายสัญญาณอินพุตและเอาต์พุตทั้ง 24 เส้น ของ 8255A มีระดับสสัญญาณ TTL จึงเป็นการง่ายในการใช้ 8255A อินเตอร์เฟสกับวงจรลอจิกอื่น ๆ รวมทั้ง CMOS ไอซี 74LS138 นำมาใช้ร่วมกับ 8255A เพื่อให้ในการถอดรหัสตำแหน่งควบคุม ISA CARD ทำงาน การจัดขาของ 8255A แอดเดรสปกติของวงจรนี้จะถูกตั้งไว้ที่ 200H และ 220H

การทำงานหลักสามารถแบ่งออกได้เป็น 2 ส่วนใหญ่ ๆ คือ

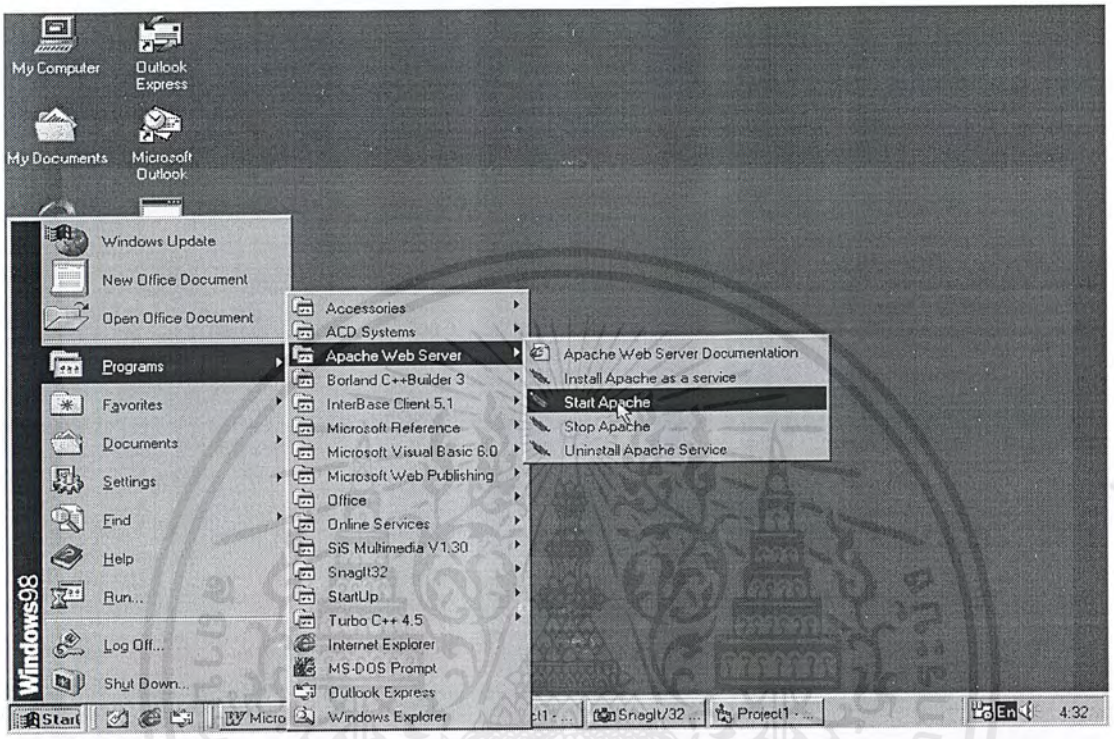
1. ฮาร์ดแวร์ คือ ตัวการ์ด I/O ที่กำลังกล่าวถึง

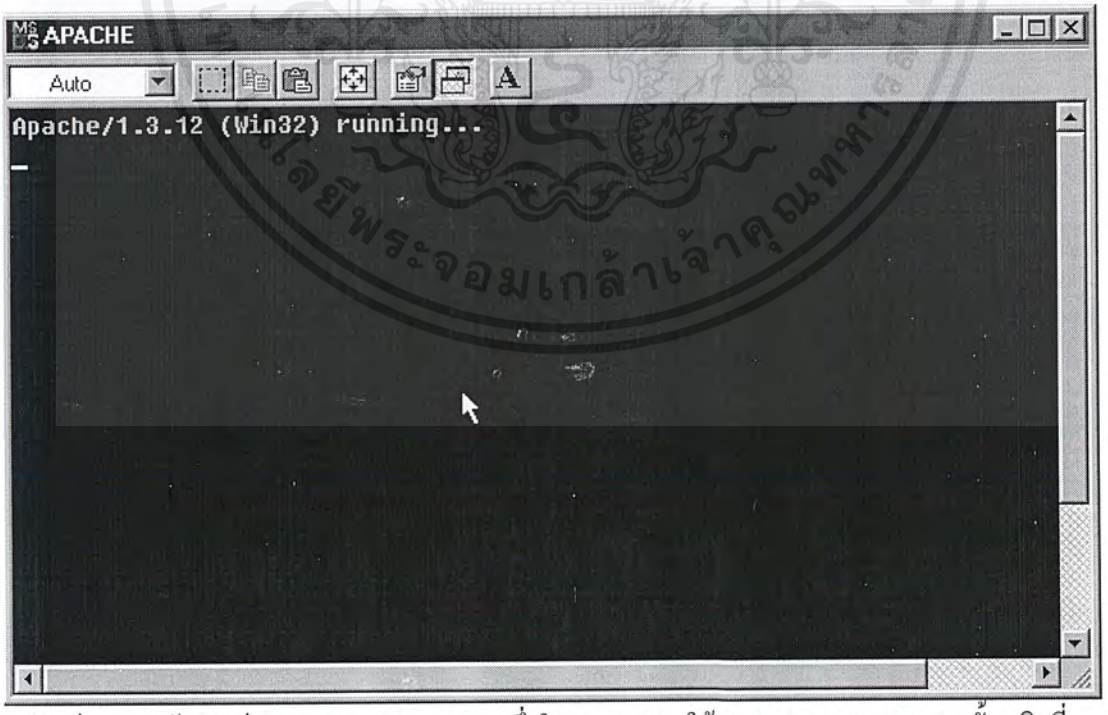
2. ซอฟแวร์ คือ โปรแกรมที่จะมาควบคุมการทำงานของการ์ด ในโครงงานนี้ใช้ Visual Basic ในการเขียนโปรแกรม

บทที่ 4
การทดลองและผลการทดลอง

การทำงานของโครงงาน

1. เปิดโปรแกรมเว็บเซิร์ฟเวอร์ เพื่อรอการร้องข้อของไคลเอ็นท์

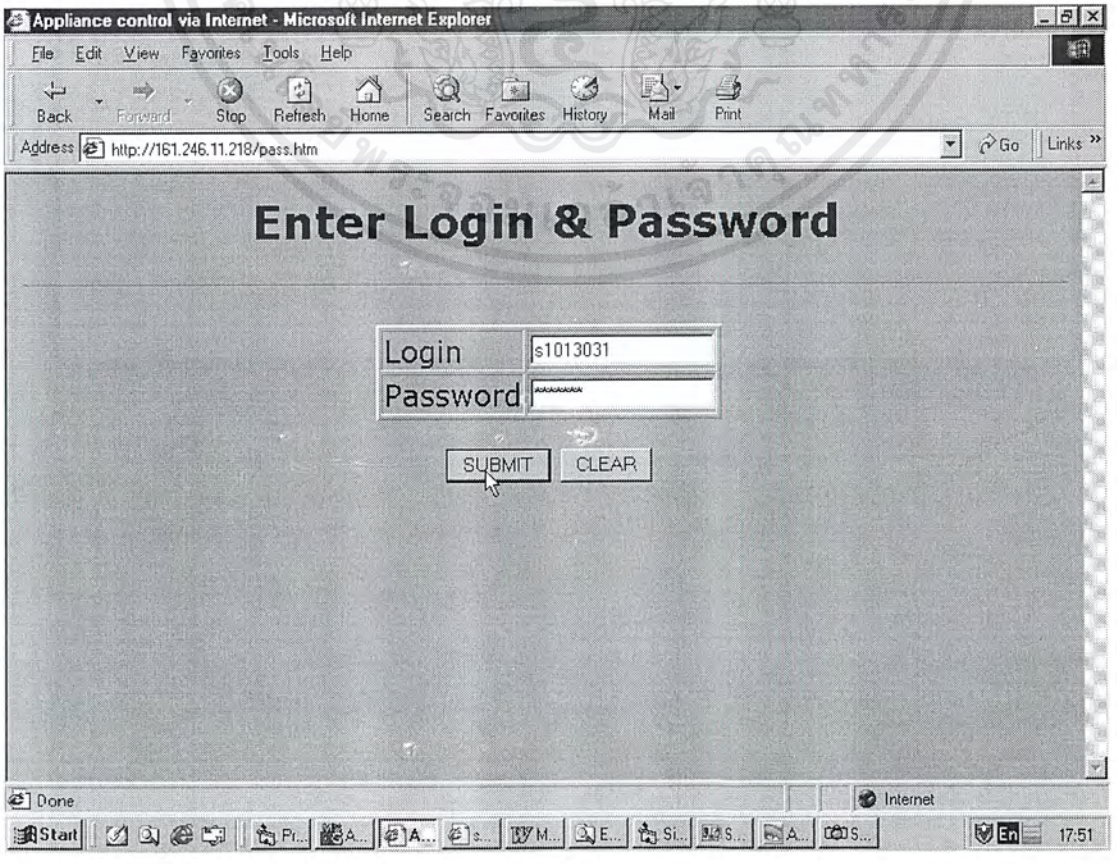




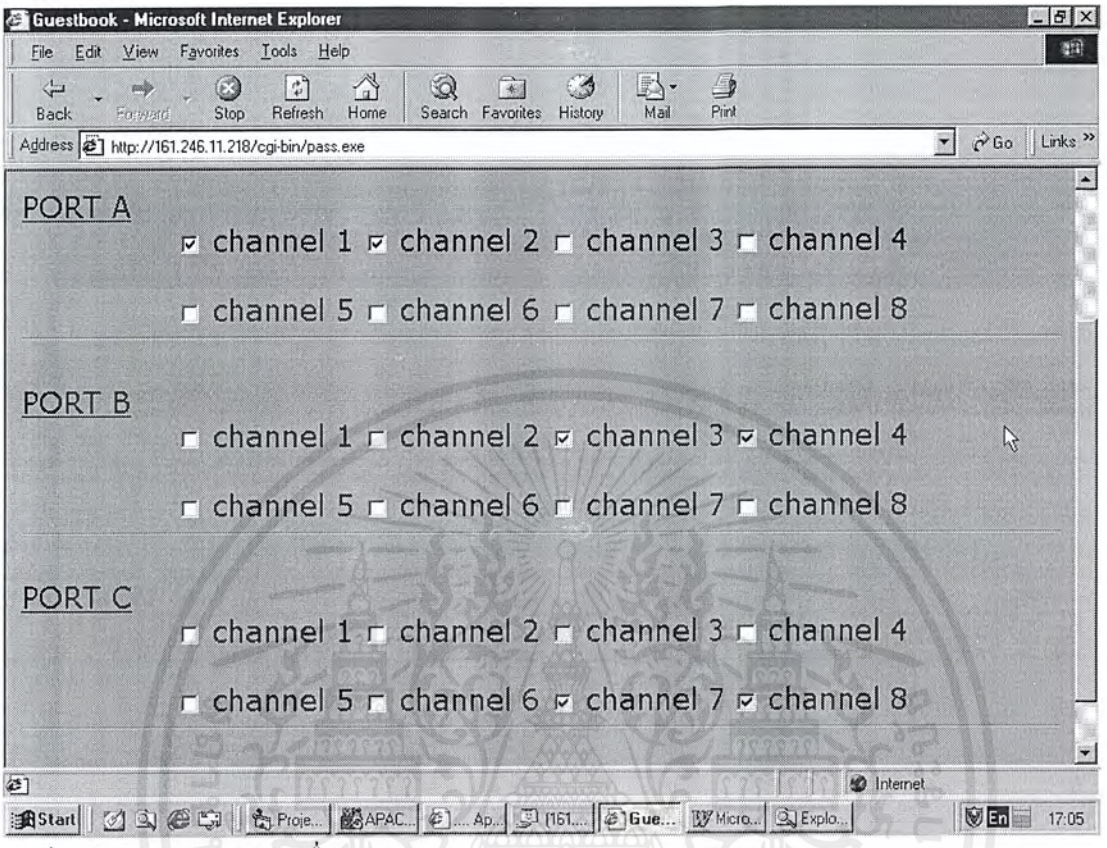
2. เปิดเว็บเพจ แล้วพิมพ์ http://161.246.11.218 ซึ่งในการทดลองใช้ IP=161.246.11.218 จากนั้นคลิกที่

Nonflash

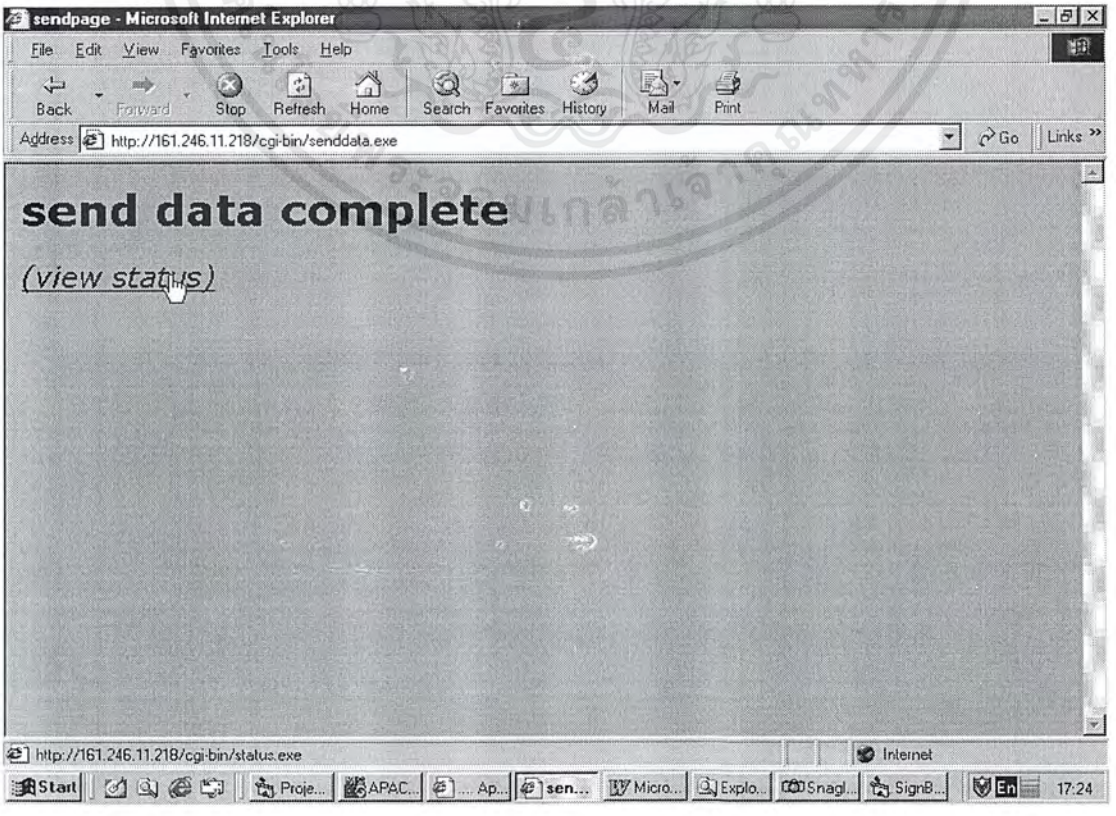
3. ใส่ Login และ Password แล้วกด SUBMIT



4. เลือกช่องที่ต้องการเปิดในที่นี้จะเปิดช่องที่ **PORT A** ส่ง 1, 2 **PORT B** ส่ง 3, 4 **PORT C** ส่ง 7,8 แล้ว กด **SUBMIT**



5. แล้วคลิก VIEWSTATUS เพื่อดูสถานะ



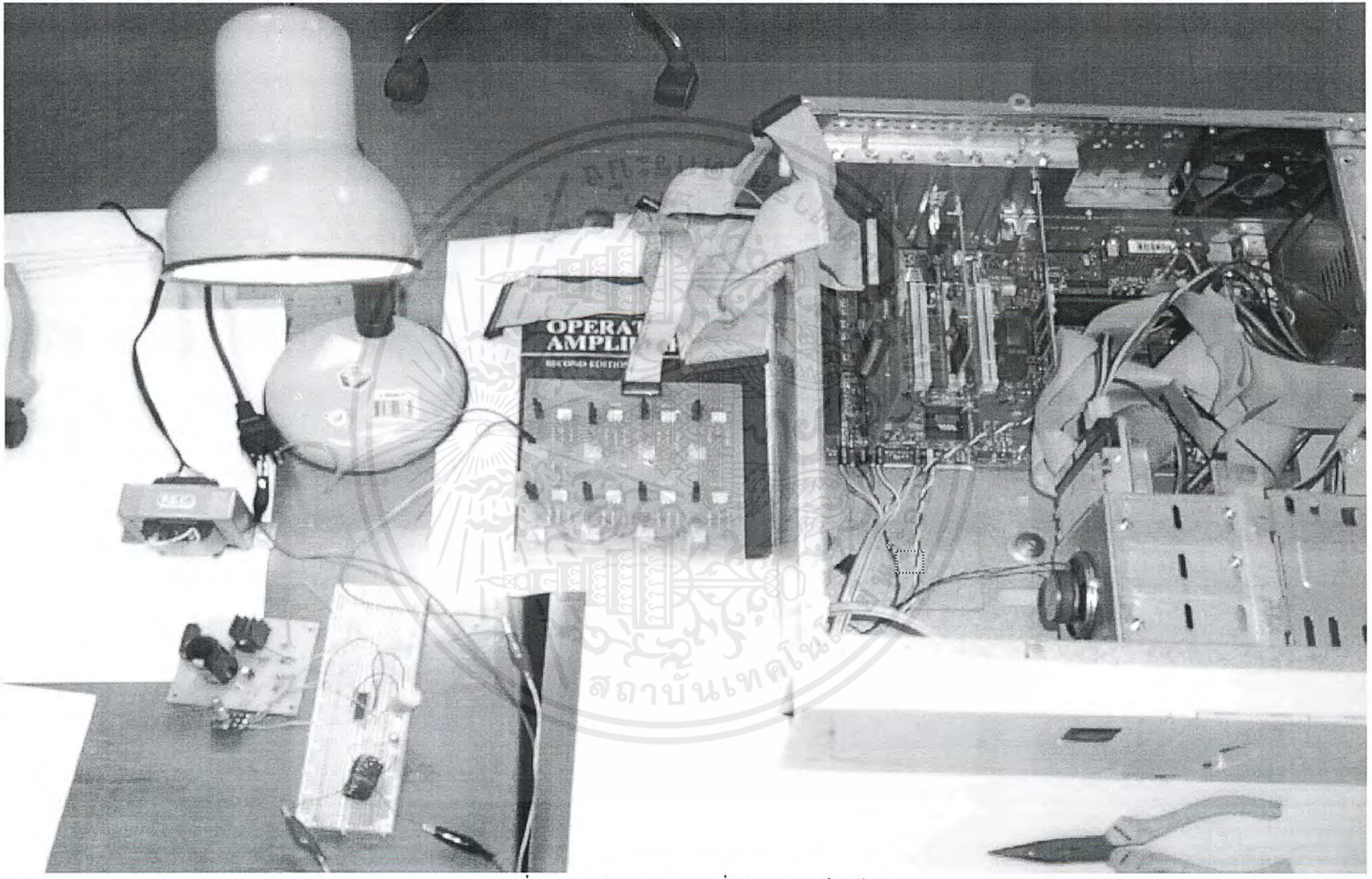
Appliance control via Internet

PORT A
Channel 1:open
Channel 2:open
Channel 3:close
Channel 4:close
Channel 5:close
Channel 6:close
Channel 7:close
Channel 8:close

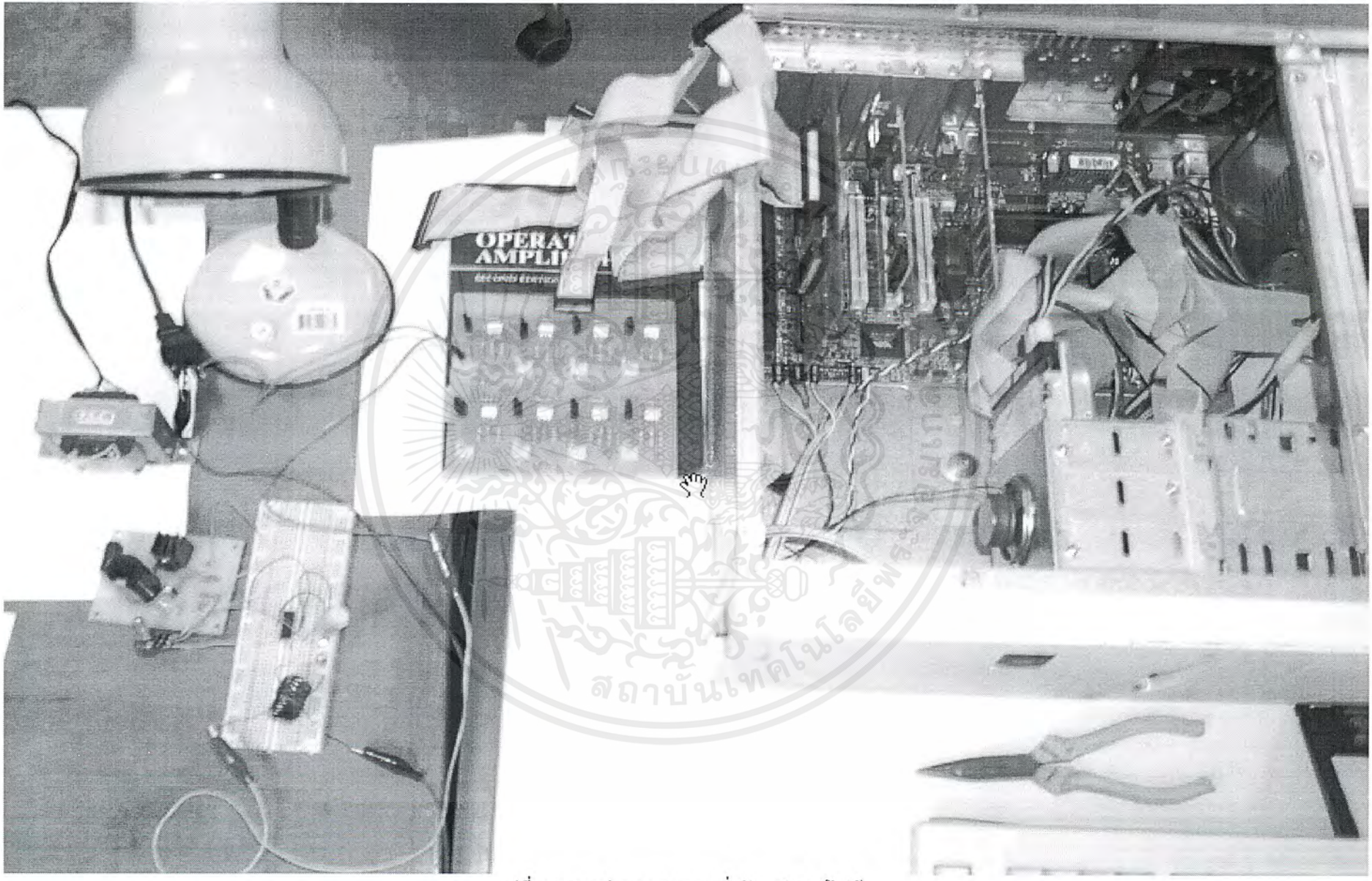
PORT B
Channel 1:close
Channel 2:close
Channel 3:open
Channel 4:open
Channel 5:close
Channel 6:close
Channel 7:close
Channel 8:close

PORT C
Channel 1:close
Channel 2:close
Channel 3:close
Channel 4:close
Channel 5:close
Channel 6:close
Channel 7:open
Channel 8:open

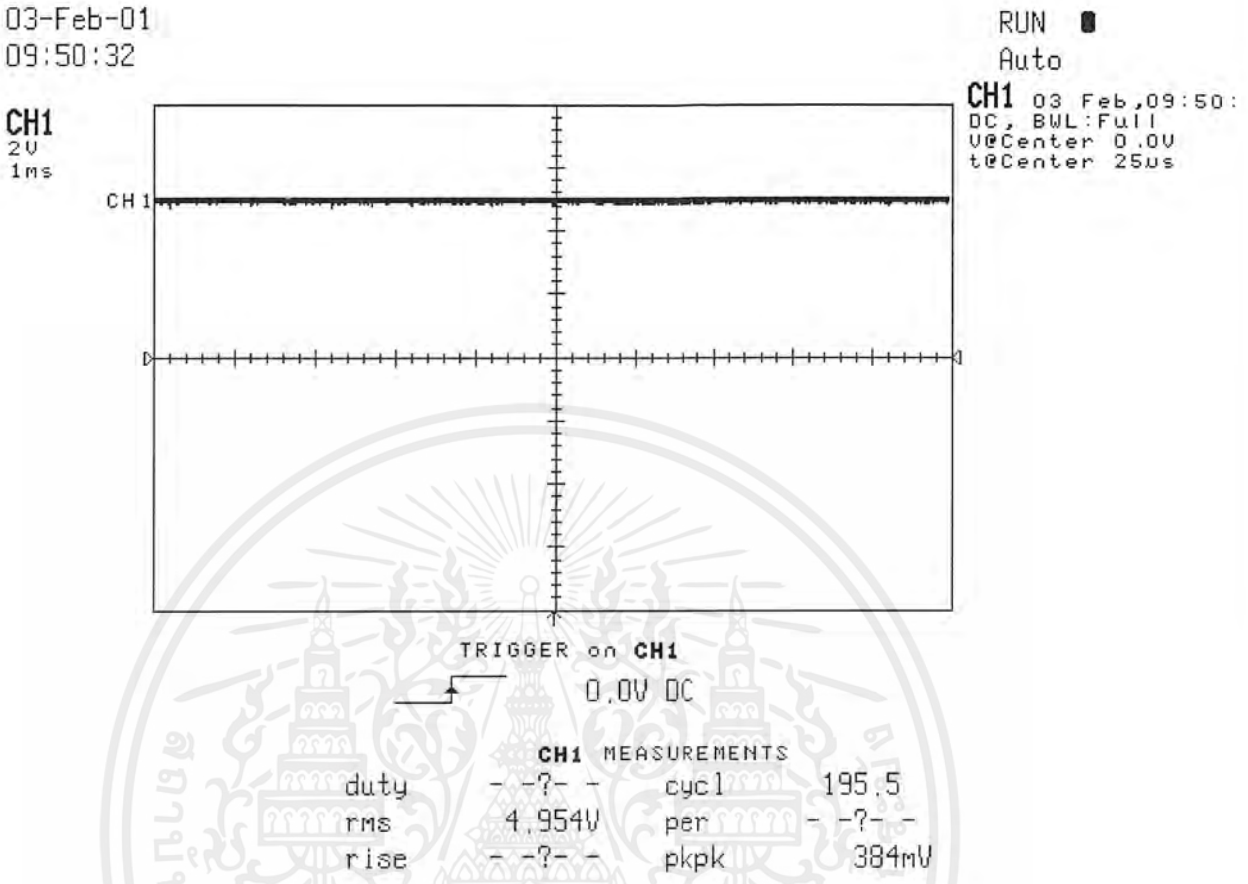
HOME



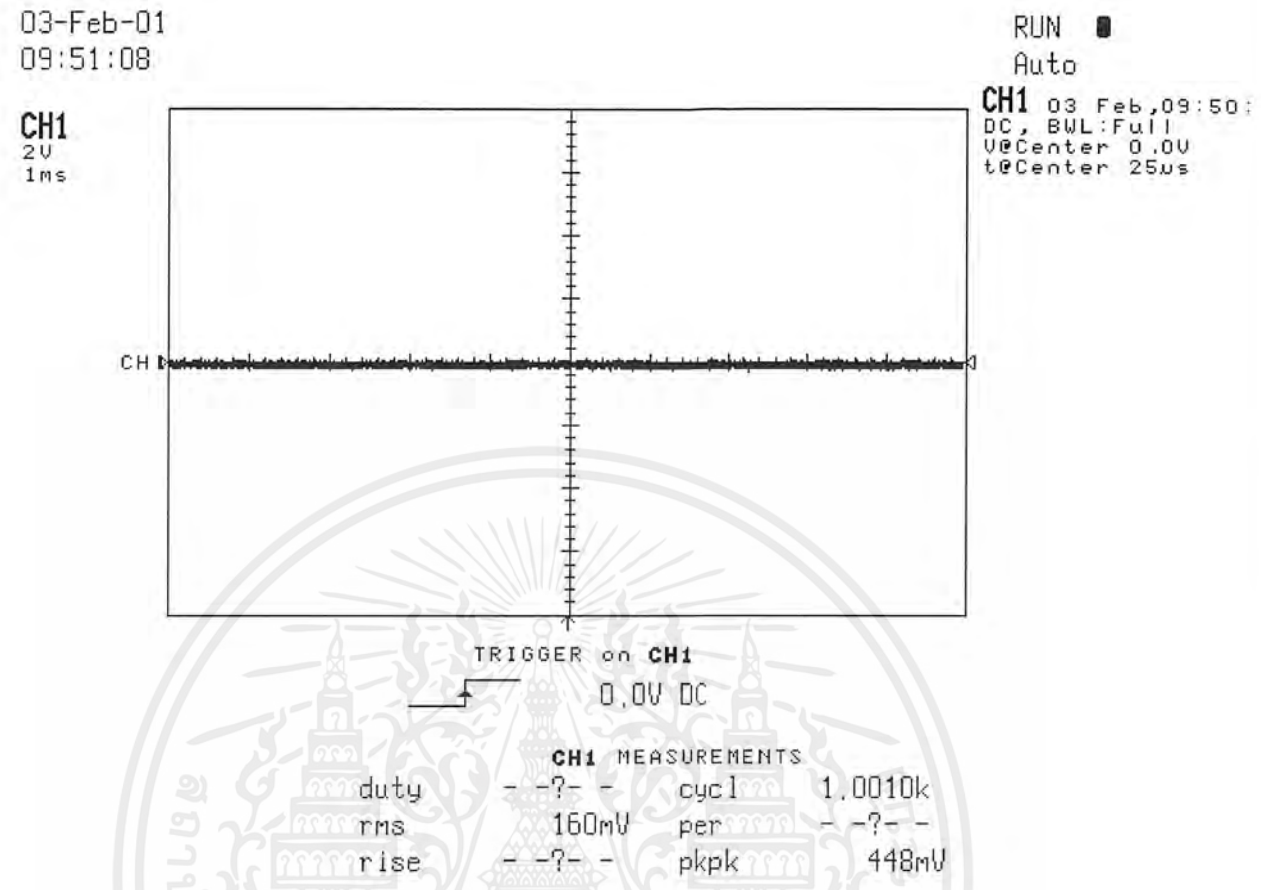
รูปที่ 4.1 ภาพถ่ายการทดลองสั่งเปิดอุปกรณ์ไฟฟ้า



รูปที่ 4.2 ภาพถ่ายการทดลองสั่งปิดอุปกรณ์ไฟฟ้า



รูปที่ 4.3 แสดงสัญญาณขณะเปิดอุปกรณ์ไฟฟ้า ก่อนส่งไปยัง SERVER



รูปที่ 4.4 แสดงสัญญาณขณะปิดอุปกรณ์ไฟฟ้า ก่อนส่งไปยัง SERVER

บทที่ 5

บทวิจารณ์และสรุป

5.1 สรุป

การใช้งานการควบคุมอุปกรณ์ไฟฟ้าผ่านทางอินเตอร์เน็ตนี้ จะเริ่มต้นโดยเปิดเครื่อง ที่เป็นเซิร์ฟเวอร์ ซึ่งต้องเปิดไว้ตลอดเวลา เพื่อรอรับคำสั่งให้ปิด-เปิด อุปกรณ์ไฟฟ้า โดยวงจรควบคุมอุปกรณ์ไฟฟ้าและวงจรตรวจสอบสถานะการทำงานของอุปกรณ์ไฟฟ้าจะถูกเชื่อมต่อกับเครื่องเซิร์ฟเวอร์ เมื่อผู้ใช้ต้องการใช้งานจะต้องทำการเปิด เวปเพจ และเวปเพจจะแสดงสถานะการทำงานของอุปกรณ์ไฟฟ้าแต่ละตัวให้ผู้ใช้ทราบ

จากนั้นผู้ใช้จะทำการสั่งเปิด-ปิด อุปกรณ์ไฟฟ้าที่ต้องการผ่านเวปเพจ เครื่องเซิร์ฟเวอร์จะส่งสัญญาณไปควบคุมอุปกรณ์ไฟฟ้าให้ทำงานตามคำสั่งที่ได้รับมาจากผู้ใช้งานเมื่ออุปกรณ์ไฟฟ้าทำงานวงจรตรวจสอบการทำงานของอุปกรณ์ไฟฟ้า จะส่งสัญญาณไปยังเซิร์ฟเวอร์ เวปเพจก็จะแสดงผล สถานะปัจจุบันของอุปกรณ์ไฟฟ้าให้ผู้ใช้ทราบ

หลังจากการศึกษาและทดลองการควบคุมอุปกรณ์ไฟฟ้าผ่านทางอินเตอร์เน็ตแล้ว ผลปรากฏว่าสามารถควบคุมการเปิดปิดอุปกรณ์ไฟฟ้า เป็นที่น่าพอใจ แต่ก็ยังเกิดปัญหาคือ สถานะการทำงานของอุปกรณ์ไฟฟ้าที่แสดงผลที่เวปเพจให้กับผู้ใช้ได้ทราบ จะเป็นการตรวจสอบการทำงานของไตรแอกมากกว่า เพราะเมื่อไตรแอกทำงานแรงดันไฟฟ้าที่ปลั๊ก ซึ่งเชื่อมต่ออยู่กับอุปกรณ์ไฟฟ้าและวงจรตรวจสอบสถานะการทำงาน ก็จะได้รับแรงดันไฟฟ้าครบ 1 เฟส ซึ่งแสดงว่าไตรแอกในช่องที่เราควบคุมทำงาน หากอุปกรณ์ไฟฟ้าที่เราต้องการควบคุมในช่องนั้นเสียวงจรตรวจสอบสถานะการทำงานก็ยังทำงานอยู่ และส่งสถานะไปยังเครื่อง เซิร์ฟเวอร์ ว่าอุปกรณ์ไฟฟ้าตัวที่ต่ออยู่นั้น เปิดใช้งานอยู่ ซึ่งความจริงอุปกรณ์ไฟฟ้านั้นชำรุด และไม่ได้เปิดใช้งาน การแก้ไขคือต้องมีการตรวจสอบ การทำงานของอุปกรณ์ไฟฟ้า แต่ละตัว ให้อยู่ในสภาพปกติ พร้อมใช้งานอยู่เสมอ และควรที่จะสร้างวงจรที่สามารถตรวจสอบการทำงานของอุปกรณ์ไฟฟ้า จากกระแสที่จ่ายไปให้กับอุปกรณ์ไฟฟ้าตัวนั้นให้ทำงาน ซึ่งจะได้สภาวะการทำงาน ที่แท้จริง และทำให้ทราบได้ว่า อุปกรณ์ไฟฟ้าตัวใดบ้างที่ชำรุด เพื่อจะได้แก้ไขหรือนำเอาอุปกรณ์มาเปลี่ยนต่อไป

5.2 บทวิจารณ์

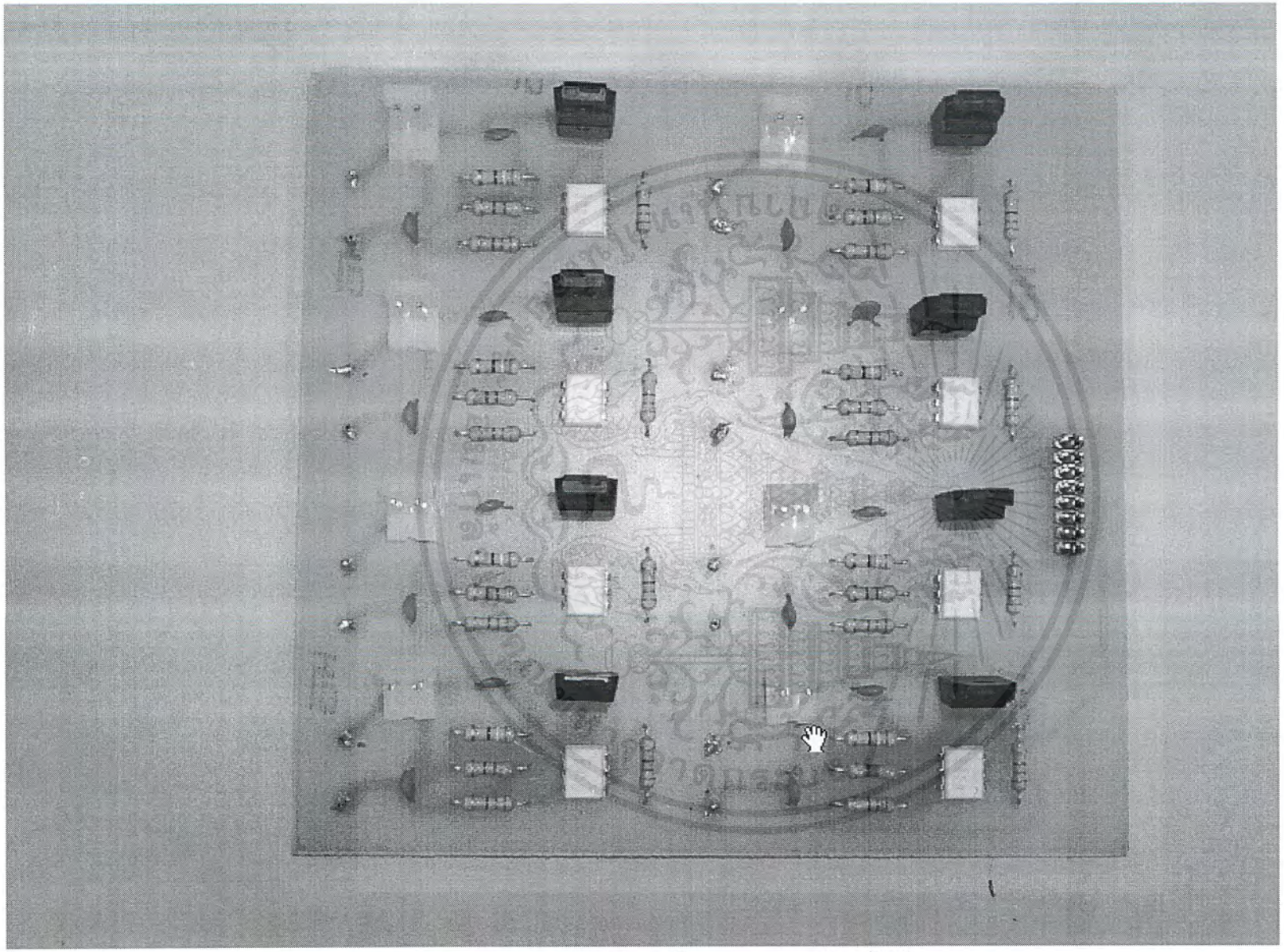
การศึกษาในการทำโครงงานนี้ มีข้อดีคือทำให้เรียนรู้หลักการของเครือข่ายอินเตอร์เน็ต และการประยุกต์ใช้งานซึ่งปัจจุบัน อินเตอร์เน็ตเป็นที่นิยมใช้งานกันอย่างแพร่หลาย

ข้อเสียจะเห็นว่าการควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเตอร์เน็ตนี้ จะเกิดปัญหากรณีที่เครือข่ายอินเตอร์เน็ตเกิดขัดข้อง จะทำให้ไม่สามารถต่อใช้งานได้ หากเราต้องการสั่งการควบคุมอุปกรณ์ไฟฟ้าในขณะนั้น

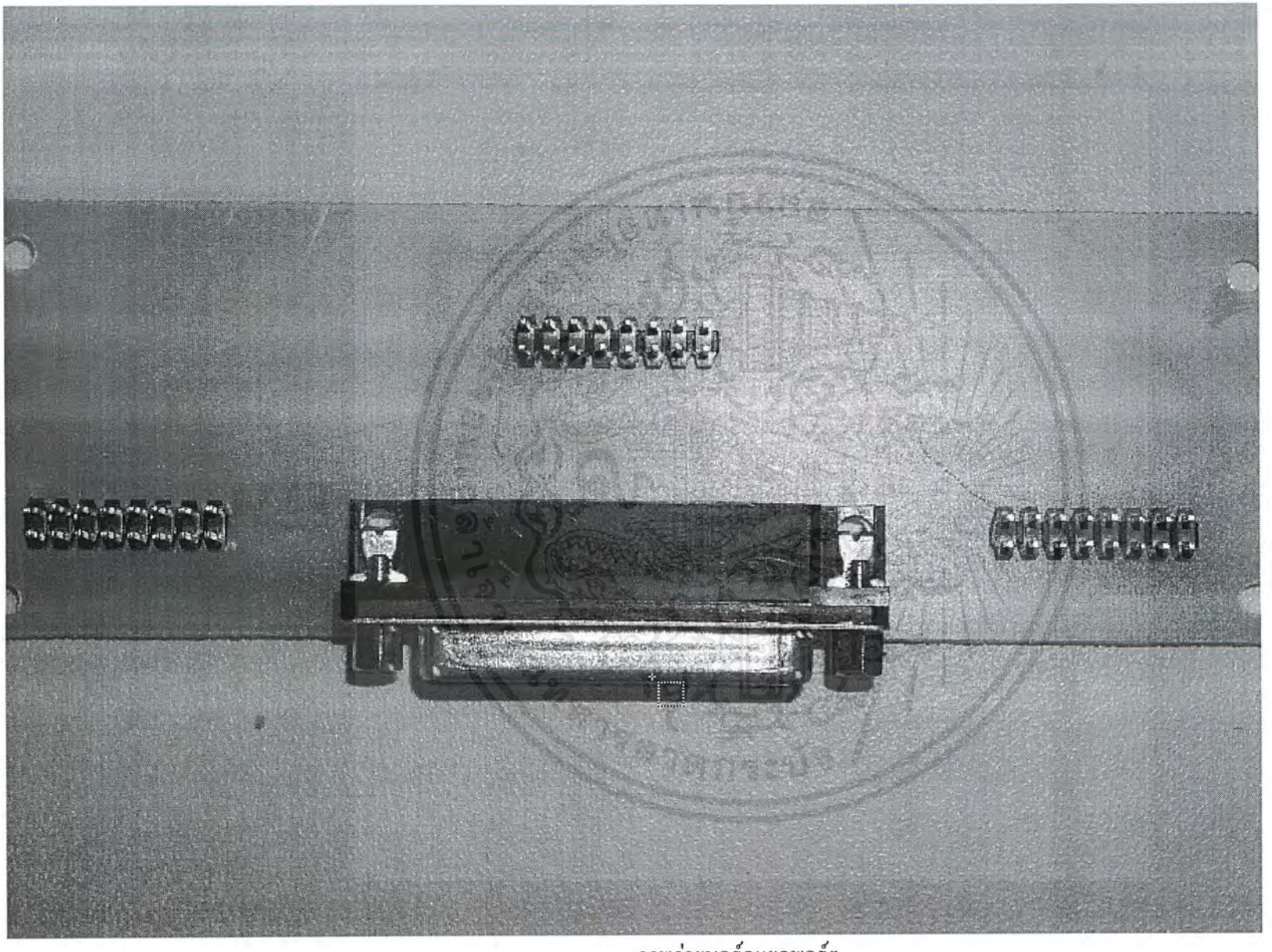
ชิ้นงานจากโครงงานนี้สามารถนำไปใช้งานได้จริง แต่ประสิทธิภาพยังไม่ดีเท่าที่ควรในการแสดงสถานะการทำงานของอุปกรณ์ไฟฟ้ากับมายังเวปเพจให้ผู้ใช้งานทราบ ในกรณีที่อุปกรณ์ไฟฟ้า ตัวที่เราต้องการควบคุมชำรุด ผู้ใช้จะไม่ทราบสถานะการทำงานจริงของอุปกรณ์ไฟฟ้าตัวนั้นได้ แต่ถ้าเราเพิ่มส่วนของวงจรตรวจสอบกระแส ที่จ่ายให้กับอุปกรณ์ไฟฟ้า ผู้ใช้ก็จะทราบสถานะการทำงานจริงของอุปกรณ์ไฟฟ้าตัวนั้นและทราบว่าอุปกรณ์ตัวใดชำรุดบ้าง เพื่อทำการแก้ไขหรือเปลี่ยนต่อไป

ภาคผนวก

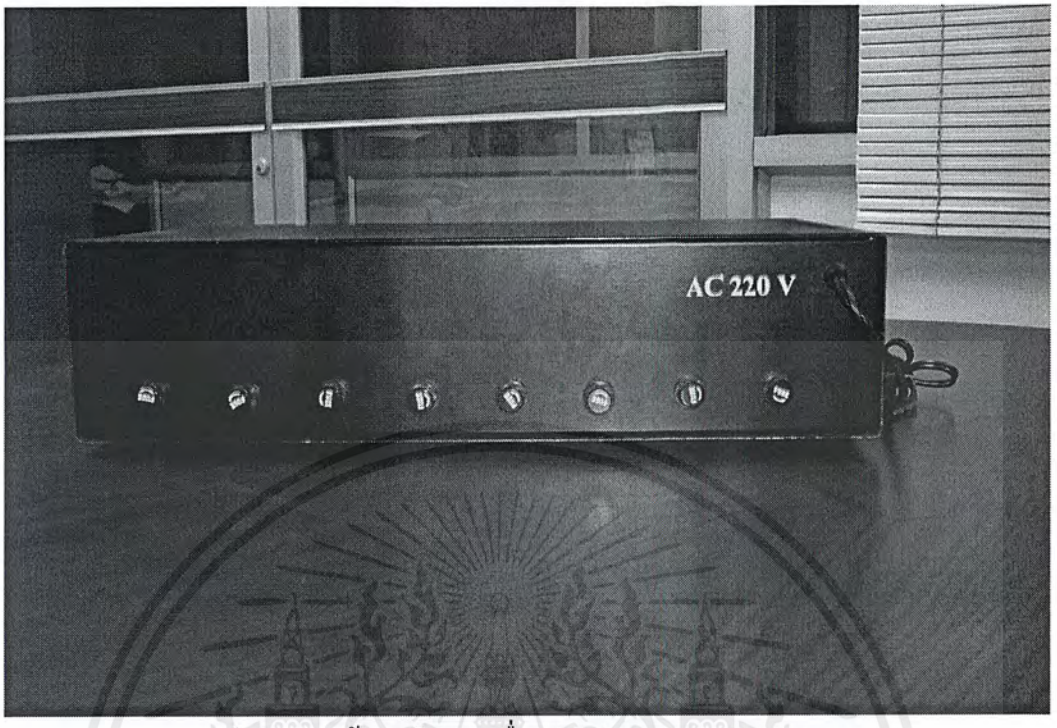
เครื่องคอมพิวเตอร์ขนาดเล็กมาก



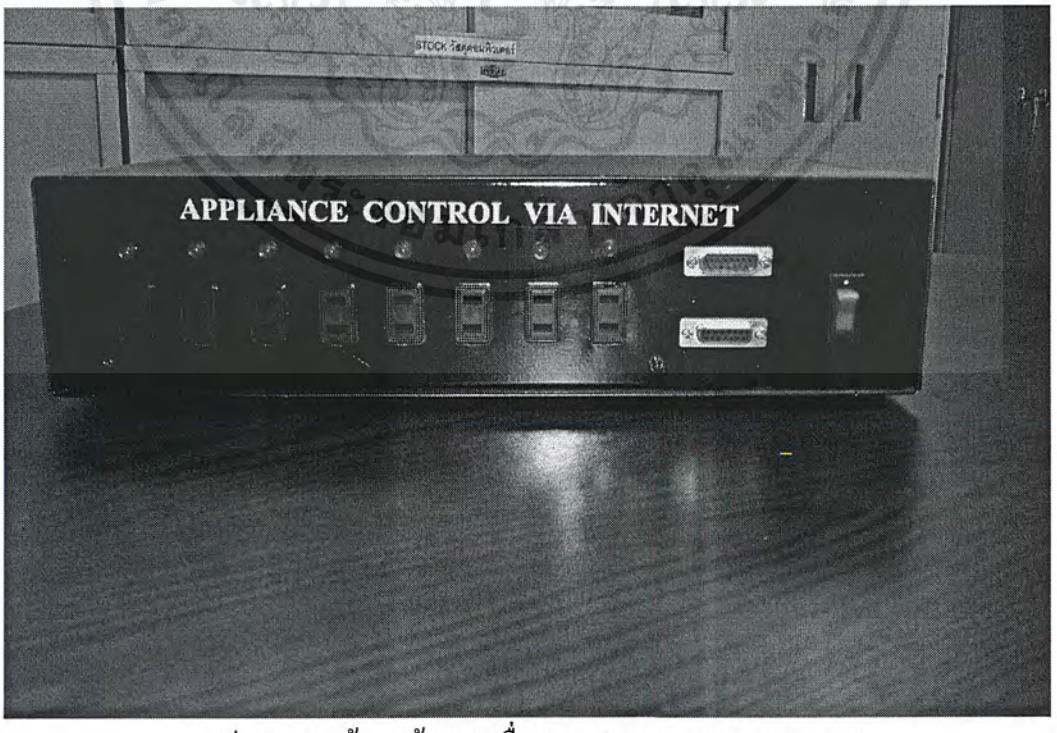
ภาพถ่ายบอร์ดวงจรควบคุมอุปกรณ์ไฟฟ้า



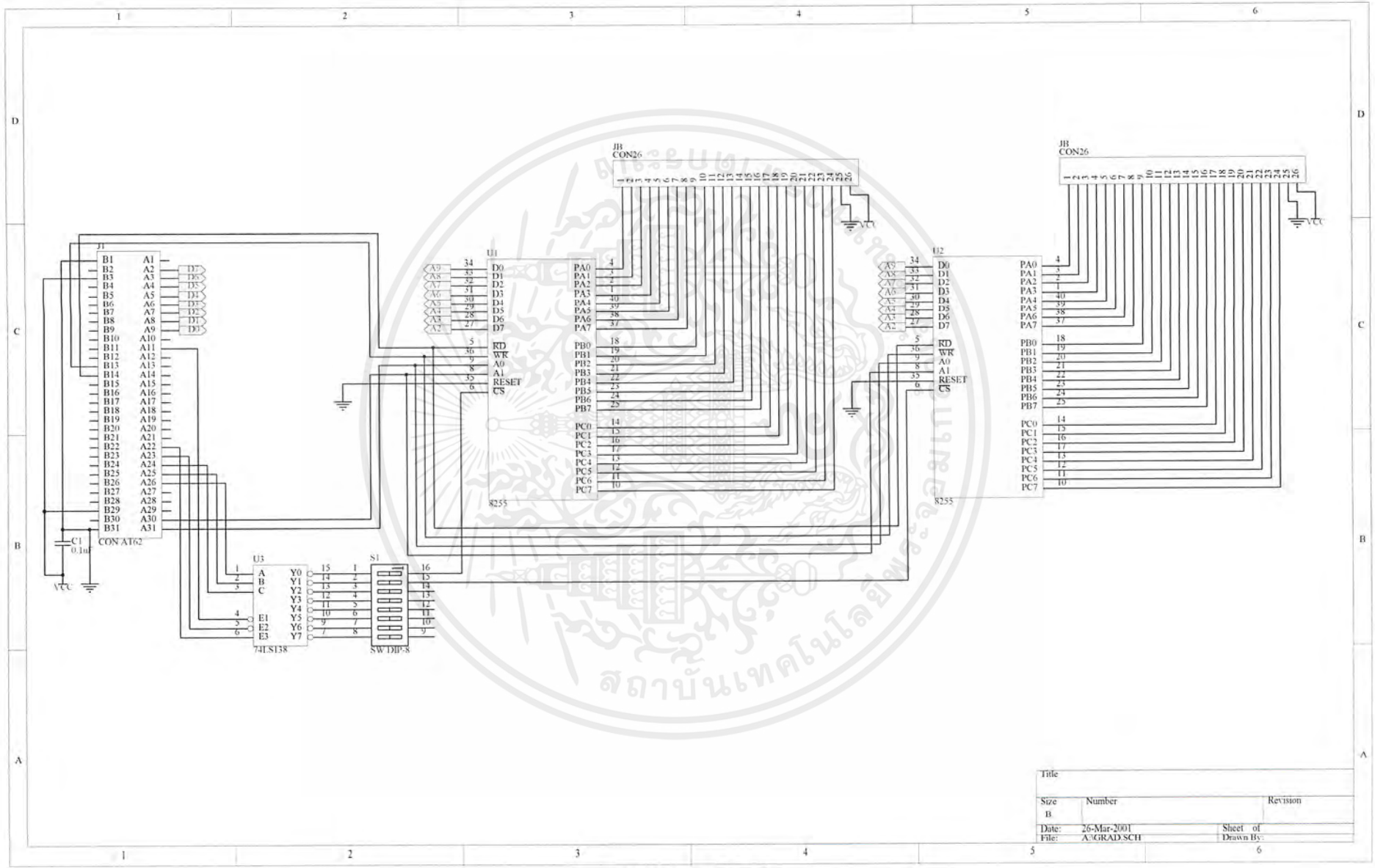
ภาพถ่ายบอร์ดแยกพอร์ต



รูปแสดงทางด้านหลังของเครื่อง Appliance control via internet



รูปแสดงทางด้านหน้าของเครื่อง Appliance control via internet



รูป วงจร ISA CARD

MAC210A8FP, MAC210A10FP

Triacs

Silicon Bidirectional Thyristors

Designed primarily for full-wave ac control applications, such as light dimmers, motor controls, heating controls and power supplies; or wherever full-wave silicon gate controlled solid-state devices are needed. Triac type thyristors switch from a blocking to a conducting state for either polarity of applied main terminal voltage with positive or negative gate triggering.

- Blocking Voltage to 800 Volts
- All Diffused and Glass Passivated Junctions for Greater Parameter Uniformity and Stability
- Small, Rugged, Thermowatt Construction for Low Thermal Resistance, High Heat Dissipation and Durability
- Gate Triggering Guaranteed in Four Modes
- Ⓡ Indicates UL Registered — File #E69369
- Device Marking: Logo, Device Type, e.g., MAC210A8FP, Date Code

MAXIMUM RATINGS (T_J = 25°C unless otherwise noted)

Rating	Symbol	Value	Unit
Peak Repetitive Off–State Voltage(1) (T_J = –40 to +125°C, Sine Wave, 50 to 60 Hz, Gate Open) MAC210A8FP MAC210A10FP	V_{DRM}, V_{RRM}	 600 800	Volts
On-State RMS Current (T_C = +70°C)(2) Full Cycle Sine Wave 50 to 60 Hz	$I_{T(RMS)}$	10	Amps
Peak Non–repetitive Surge Current (One Full Cycle Sine Wave, 60 Hz, T_C = +70°C) Preceded and followed by rated current	I_{TSM}	100	Amps
Circuit Fusing Consideration (t = 8.3 ms)	I^2t	40	A^2s
Peak Gate Power (T_C = +70°C, Pulse Width = 10 μs)	P_{GM}	20	Watts
Average Gate Power (T_C = +70°C, t = 8.3 ms)	$P_{G(AV)}$	0.35	Watt
Peak Gate Current (T_C = +70°C, Pulse Width = 10 μsec)	I_{GM}	2.0	Amps
RMS Isolation Voltage (T_A = 25°C, Relative Humidity ≤ 20%) (Ⓡ)	$V_{(ISO)}$	1500	Volts
Operating Junction Temperature Range	T_J	–40 to +125	°C
Storage Temperature Range	T_{stg}	–40 to +150	°C

(1) V_{DRM} and V_{RRM} for all types can be applied on a continuous basis. Blocking voltages shall not be tested with a constant current source such that the voltage ratings of the devices are exceeded.

(2) The case temperature reference point for all T_C measurements is a point on the center lead of the package as close as possible to the plastic body.

ON Semiconductor

http://onsemi.com

ISOLATED TRIAC (Ⓡ)
10 AMPERES RMS
600 thru 800 VOLTS

ISOLATED TO–220 Full Pack
CASE 221C
STYLE 3

PIN ASSIGNMENT	
1	Main Terminal 1
2	Main Terminal 2
3	Gate

ORDERING INFORMATION

Device	Package	Shipping
MAC210A8FP	ISOLATED TO220FP	500/Box
MAC210A10FP	ISOLATED TO220FP	500/Box

THERMAL CHARACTERISTICS

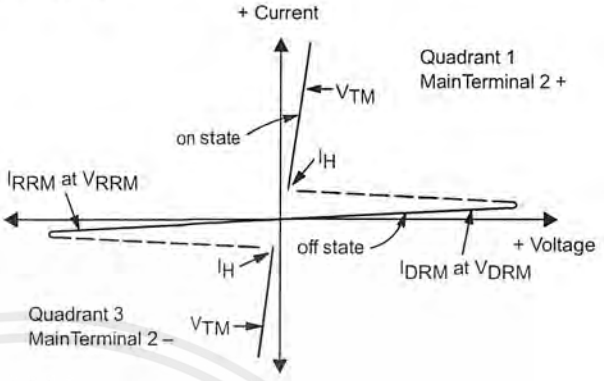
Characteristic	Symbol	Max	Unit
Thermal Resistance, Junction to Case	$R_{\theta JC}$	2.2	°C/W
Thermal Resistance, Case to Sink	$R_{\theta CS}$	2.2 (typ)	°C/W
Thermal Resistance, Junction to Ambient	$R_{\theta JA}$	60	°C/W
Maximum Lead Temperature for Soldering Purposes 1/8″ from Case for 10 Seconds	T_L	260	°C

ELECTRICAL CHARACTERISTICS (T_C = 25°C unless otherwise noted; Electricals apply in both directions)

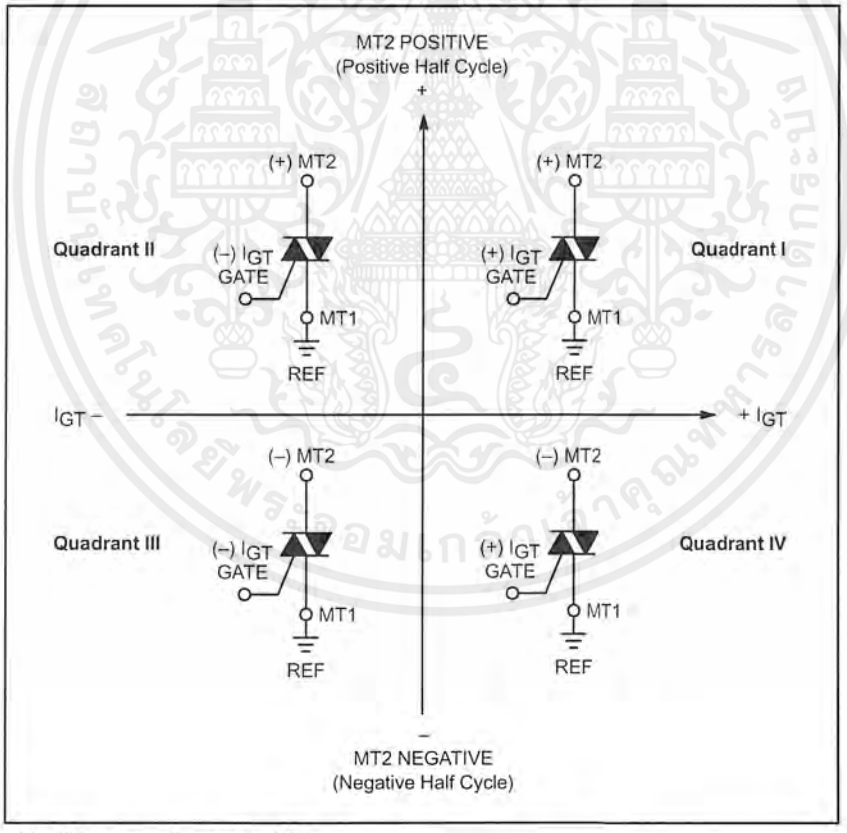
Characteristic		Symbol	Min	Typ	Max	Unit
OFF CHARACTERISTICS						
Peak Repetitive Blocking Current (V_D = Rated V_{DRM}, V_{RRM}; Gate Open)	T_J = 25°C	I_{DRM}, I_{RRM}	—	—	10	μA
	T_J = +125°C		—	—	2.0	mA
ON CHARACTERISTICS						
Peak On-State Voltage (I_{TM} = ±14 A Peak; Pulse Width = 1 to 2 ms, Duty Cycle ≤ 2%)		V_{TM}	—	1.2	1.65	Volts
Gate Trigger Current (Continuous dc) (Main Terminal Voltage = 12 Vdc, R_L = 100 Ohms)		I_{GT}				mA
MT2(+), G(+)			—	12	50	
MT2(+), G(–)			—	12	50	
MT2(–), G(–)			—	20	50	
MT2(–), G(+)			—	35	75	
Gate Trigger Voltage (Continuous dc) (Main Terminal Voltage = 12 Vdc, R_L = 100 Ohms)		V_{GT}				Volts
MT2(+), G(+)			—	0.9	2.0	
MT2(+), G(–)			—	0.9	2.0	
MT2(–), G(–)			—	1.1	2.0	
MT2(–), G(+)			—	1.4	2.5	
Gate Non–Trigger Voltage (Continuous dc) (Main Terminal Voltage = 12 V, R_L = 100 Ω, T_J = +125°C) All Four Quadrants		V_{GD}	0.2	—	—	Volts
Holding Current (Main Terminal Voltage = 12 Vdc, Gate Open, Initiating Current = ±200 mA)		I_H	—	6.0	50	mA
Turn-On Time (Rated V_{DRM}, I_{TM} = 14 A, I_{GT} = 120 mA, Rise Time = 0.1 μs, Pulse Width = 2 μs)		t_{gt}	—	1.5	—	μs
DYNAMIC CHARACTERISTICS						
Critical Rate of Rise of Commutation Voltage (V_D = Rated V_{DRM}, I_{TM} = 14 A, Commutating di/dt = 5.0 A/ms, Gate Unenergized, T_C = +70°C)		$dv/dt_{(c)}$	—	5.0	—	V/μs
Critical Rate of Rise of Off–State Voltage (V_D = Rated V_{DRM}, Exponential Voltage Rise, Gate Open, T_C = +70°C)		dv/dt	—	100	—	V/μs

Voltage Current Characteristic of Triacs (Bidirectional Device)

Symbol	Parameter
V_{DRM}	Peak Repetitive Forward Off State Voltage
I_{DRM}	Peak Forward Blocking Current
V_{RRM}	Peak Repetitive Reverse Off State Voltage
I_{RRM}	Peak Reverse Blocking Current
V_{TM}	Maximum On State Voltage
I_H	Holding Current



Quadrant Definitions for a Triac



All polarities are referenced to MT1.
With in–phase signals (using standard AC lines) quadrants I and III are used.

TYPICAL CHARACTERISTICS

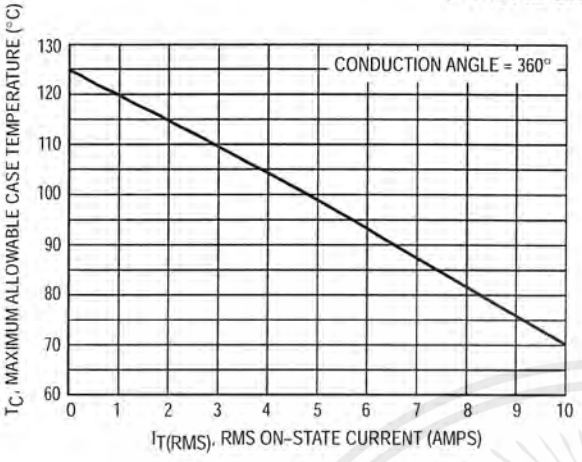


Figure 1. Current Derating

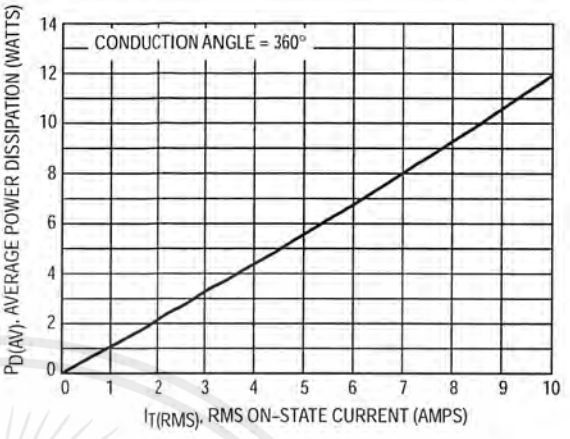


Figure 2. Power Dissipation

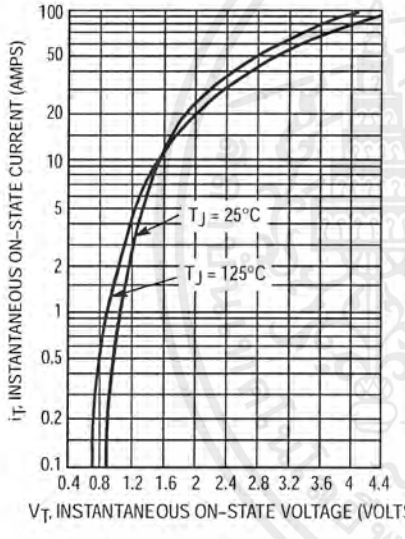


Figure 3. Maximum On–State Characteristics

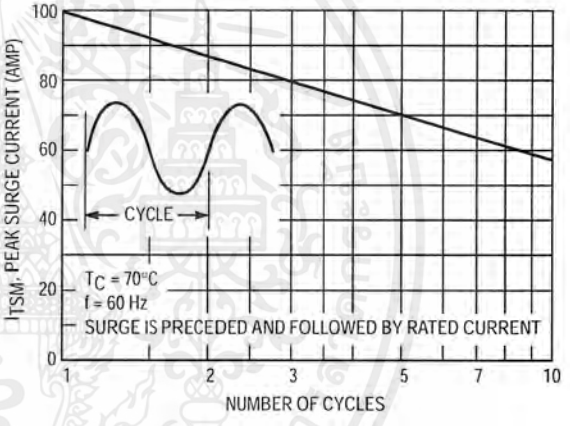


Figure 4. Maximum Nonrepetitive Surge Current

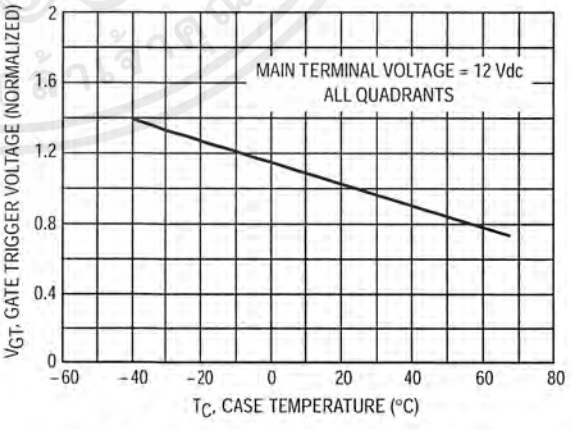


Figure 5. Typical Gate Trigger Voltage

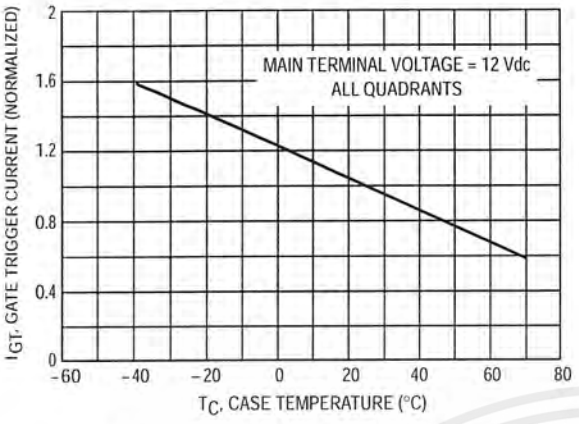


Figure 6. Typical Gate Trigger Current

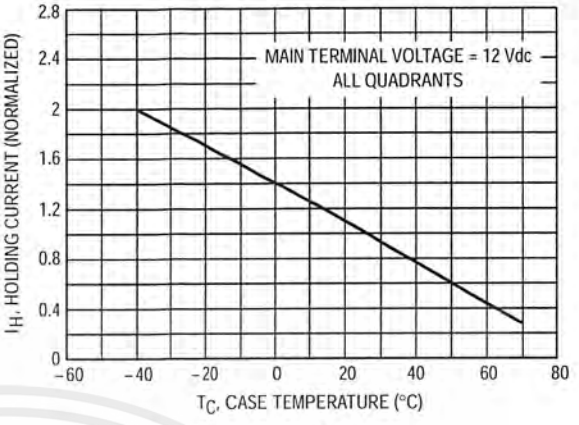


Figure 7. Typical Holding Current

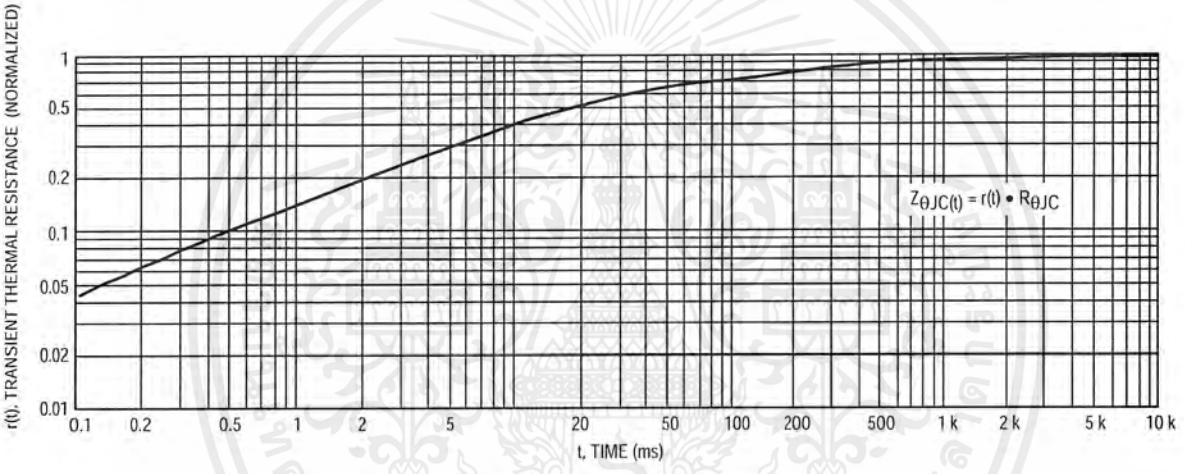
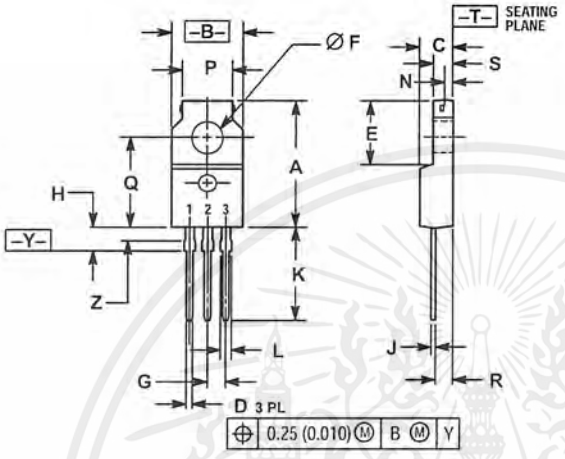


Figure 8. Thermal Response

PACKAGE DIMENSIONS

ISOLATED TO–220 Full Pack
CASE 221C–02
ISSUE C



NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: INCH.
3. LEAD DIMENSIONS UNCONTROLLED WITHIN DIMENSION Z.

	INCHES		MILLIMETERS	
DIM	MIN	MAX	MIN	MAX
A	0.680	0.700	17.28	17.78
B	0.388	0.408	9.86	10.36
C	0.175	0.195	4.45	4.95
D	0.025	0.040	0.64	1.01
E	0.340	0.355	8.64	9.01
F	0.140	0.150	3.56	3.81
G	0.100 BSC		2.54 BSC	
H	0.110	0.155	2.80	3.93
J	0.018	0.028	0.46	0.71
K	0.500	0.550	12.70	13.97
L	0.045	0.070	1.15	1.77
N	0.049	—	1.25	—
P	0.270	0.290	6.86	7.36
Q	0.480	0.500	12.20	12.70
R	0.090	0.120	2.29	3.04
S	0.105	0.115	2.67	2.92
Z	0.070	0.090	1.78	2.28

STYLE 3:
PIN 1. MT 1
2. MT 2
3. GATE

Notes

PUBLICATION ORDERING INFORMATION

NORTH AMERICA Literature Fulfillment:
Literature Distribution Center for ON Semiconductor
P.O. Box 5163, Denver, Colorado 80217 USA
Phone: 303–675–2175 or 800–344–3860 Toll Free USA/Canada
Fax: 303–675–2176 or 800–344–3867 Toll Free USA/Canada
Email: ONlit@hibbertco.com
Fax Response Line: 303–675–2167 or 800–344–3810 Toll Free USA/Canada

N. American Technical Support: 800–282–9855 Toll Free USA/Canada

EUROPE: LDC for ON Semiconductor – European Support
German Phone: (+1) 303–308–7140 (M–F 1:00pm to 5:00pm Munich Time)
Email: ONlit–german@hibbertco.com
French Phone: (+1) 303–308–7141 (M–F 1:00pm to 5:00pm Toulouse Time)
Email: ONlit–french@hibbertco.com
English Phone: (+1) 303–308–7142 (M–F 12:00pm to 5:00pm UK Time)
Email: ONlit@hibbertco.com

EUROPEAN TOLL–FREE ACCESS*: 00–800–4422–3781
*Available from Germany, France, Italy, England, Ireland

CENTRAL/SOUTH AMERICA:
Spanish Phone: 303–308–7143 (Mon–Fri 8:00am to 5:00pm MST)
Email: ONlit–spanish@hibbertco.com

ASIA/PACIFIC: LDC for ON Semiconductor – Asia Support
Phone: 303–675–2121 (Tue–Fri 9:00am to 1:00pm, Hong Kong Time)
Toll Free from Hong Kong & Singapore:
001–800–4422–3781
Email: ONlit–asia@hibbertco.com

JAPAN: ON Semiconductor, Japan Customer Focus Center
4–32–1 Nishi–Gotanda, Shinagawa–ku, Tokyo, Japan 141–8549
Phone: 81–3–5740–2745
Email: r14525@onsemi.com

ON Semiconductor Website: http://onsemi.com

For additional information, please contact your local
Sales Representative.

กิตติกรรมประกาศ

โครงงานฉบับนี้สามารถสำเร็จลุล่วงไปได้ด้วยดี โดยอาศัยความกรุณาของบุคคลหลายๆ ท่านได้แก่ อาจารย์สุรพล บุญจันทร์ , อาจารย์ นภัทร สระเอี่ยม, คุณกมลเนตร ระลึกชาติ, คุณดวงอาทิตย์ ศรีมูล คุณกิตติ อนุศิษฎ์วิวัฒน์ ที่ให้ยืมเครื่องสแกนเนอร์ เครื่องคอมพิวเตอร์ และ เครื่องพิมพ์ รวมทั้งบิดา - มารดา ที่คอยให้กำลังใจตลอดเวลา และเพื่อนๆ ที่ให้คำปรึกษา และแนะนำตลอดช่วยแก้ปัญหาต่างๆ ที่ในการทำงานและการจัดทำโครงงานฉบับนี้ ผู้จัดทำต้องขอขอบคุณทุกๆ ท่านที่ได้กล่าวนามมาเป็นอย่างสูง

เอกสารอ้างอิง

1. ฉลองชัย จงประเสริฐ และ วรรวิภา ท่าพระนา "CGI /WEB Programming การพัฒนาโปรแกรมใช้งานบน เครือข่ายอินเตอร์เน็ต" บริษัท ซีเอ็ดยูเคชั่น จำกัด (มหาชน)
2. "วารสารเซมิคอนดัคเตอร์" ฉบับที่ 187 เดือนสิงหาคม 2541