โปรแกรมสนับสนุนการเขียนโปรแกรมควบคุม PLC

LADDER SOFTWARE SUPPORT

นายพรเพชร กิจสิริสินชัย
นายสิทธิชัย ประสิทธิเวชชากูร
นายสุรเชษฎ์ ศิริลาภพานิช

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมการวัดคุม
ภาควิชาเทคโนโลยีการวัดคุมทางอุตสาหกรรม คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2543

LADDER SOFTWARE SUPPORT

Pornpeth Kijsirisinchai

Sitichai Prasitivedchakon

Surachet Sririlapanit

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT THE DEGREE OF

BACHELOR OF ENGINEERING IN INSTRUMENTATION ENGINEERING

DEPARTMENT OF INDUSTRIAL INSTRUMENTATION TECHNOLOGY

FACULTY OF ENGINEERING

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2000

ภาควิชาเทคโนโลยีการวัดคุมทางอุตสาหกรรม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ใบรับรองปริญญานิพนธ์

หัวข้อปริญญานิพนธ์ โปรแกรมสนับสนุนการเขียนโปรแกรมควบคุม PLC
LADDER SOFTWARE SUPPORT

นักศึกษาผู้จัดทำ

นายพรเพชร	กิจสิริสินชัย	รหัสประจำตัว	41012091
นายสิทธิชัย	ประสิทธิเวชชากูร	รหัสประจำตัว	41012107
นายสุรเชษฎ์	ศิริลาภพานิช	รหัสประจำตัว	41012112

ปริญญา วิศวกรรมศาสตรบัณฑิต

สาขาวิชา วิศวกรรมการวัดคุม

ปีการศึกษา 2543

อาจารย์ผู้ควบคุมปริญญานิพนธ์	ลายมือชื่อ
รศ. สุพรรณ กุลพาณิชย์ ผศ. วิริยะ กองรัตน์	

วัน/เดือน/ปี ที่สอบ วันจันทร์ที่ 9 เมษายน พ.ศ. 2544

สถานที่สอบ ณ. ห้องสอบปริญญานิพนธ์ ภาควิชาเทคโนโลยีการวัดคุมทางอุตสาหกรรม

ภาควิชารับรองแล้ว

(ผศ. ประสิทธ์ จุลเสรีวงศ์)

หัวหน้าภาควิชาฯ

หัวข้อปริญญานิพนธ์	โปรแกรมสนับสนุนการเขียนโปรแกรมควบคุม PLC LADDER SOFTWARE SUPPORT	
นักศึกษาผู้จัดทำ	นายพรเพชร	กิจสิริสินชัย
	นายสิทธิชัย	ประสิทธิเวชชากูร
	นายสุรเชษฎ์	ศิริลาภพานิช
อาจารย์ที่ปรึกษา	รศ. สุพรรณ	กุลพาณิชย์
	ผศ. วิริยะ	กองรัตน์
ปีการศึกษา	2543	

บทคัดย่อ

ปัจจุบันการควบคุมแบบอัตโนมัติ ในวงการอุตสาหกรรมได้รับความนิยมกันอย่างแพร่หลาย และเครื่องควบคุม PLC ก็เป็นหนึ่งในเครื่องควบคุมในจำนวนหลายๆ ประเภท ที่ได้นำมาประยุกต์ใช้งาน

ในงานวิจัยนี้ขอนำเสนอ การพัฒนาโปรแกรมบนคอมพิวเตอร์ เพื่อใช้เป็นเครื่องมือสนับสนุนใน การเขียนโปรแกรมให้กับเครื่องควบคุม PLC โปรแกรมนี้เรียกว่า โปรแกรมสนับสนุนการเขียนแลดเดอร์ (LADDER SOFTWARE SUPPORT) จะทำให้การเขียนโปรแกรมกับเครื่องควบคุม PLC สะดวก และรวดเร็วยิ่งขึ้น

Thesis Title	LADDER SOFTWARE SUPPORT
Authers	Mr. Pornpeth Kijsirisinchai
	Mr. Sitichai Prasitivedchakon
	Mr. Surachet Srirlapanit
Thesis Advisor	Assoc.Prof. Suphan Gulpanich
	Asst.Prof. Wiriya Kongrat
Year	2000

ABSTRACT

Currently the automatic control system are more widely used. So that various controllers including. Programmable logic Controller (PLC) are used to apply in that systems.

This paper presents Computer Software Support which is developed on Computer to be the supporting tool for PLC programming. This program is " LADDER SOFTWARE SUPPORT ". So that PLC program developing will be a more convenient and time saving process.

กิตติกรรมประกาศ

การจัดทำปริญญานิพนธ์ฉบับนี้สามารถสำเร็จลุล่วงด้วยดี เพราะได้รับความเมตตาจาก ท่าน รศ.สุพรรณ กุลพาณิชย์ ท่าน ผศ.วิริยะ กองรัตน์ ท่านอาจารย์ ทวีพล ซื่อสัตย์ และท่านอาจารย์ อาจินต์ น่วมสำราญ รวมไปถึงท่านอาจารย์ทุกๆ ท่านที่ไม่ได้เอ่ยนามในที่นี้ ซึ่งได้ให้คำปรึกษาและแนะนำคณะผู้จัดทำมาโดยตลอด คณะผู้จัดทำรู้สึกทราบซึ้งอย่างยิ่งในความอนุเคราะห์จากท่าน ที่ได้ประสิทธิ์ประสาทวิชาความรู้แก่ศิษย์ในการศึกษาระดับต่างๆ ผู้จัดทำขอกราบขอบพระคุณอย่างสูง

ขอกราบขอบพระคุณ บิดา มารดา และพี่น้อง ที่ให้โอกาสกับลูกสำหรับการศึกษาเล่าเรียนรวมทั้งให้กำลังใจต่างๆ ในการศึกษาตลอดมา

ขอขอบคุณ บริษัท แฟคทอรี่ คอนเซ้าท์แตนท์ จำกัด ที่ให้ความสนับสนุนอุปกรณ์และข้อมูลในการทำปริญญานิพนธ์และทำให้ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงไปด้วยดี

ขอขอบคุณ เพื่อน ๆ พี่ ๆ น้อง ๆ ทุกท่าน ที่เป็นกำลังใจและช่วยเหลือในการทำปริญญานิพนธ์ฉบับนี้มาโดยตลอด

ขออำนาจคุณพระศรีรัตนตรัยและสิ่งศักดิ์สิทธิ์ทั้งหลาย ช่วยดลบันดาลให้ทุกท่านที่กล่าวถึงนั้น ประสบความสำเร็จ มีความสุขความเจริญในหน้าที่การงานทุกประการ

คณะผู้จัดทำ

สารบัญ

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญตาราง

สารบัญรูป

สารบัญรูป(ต่อ)

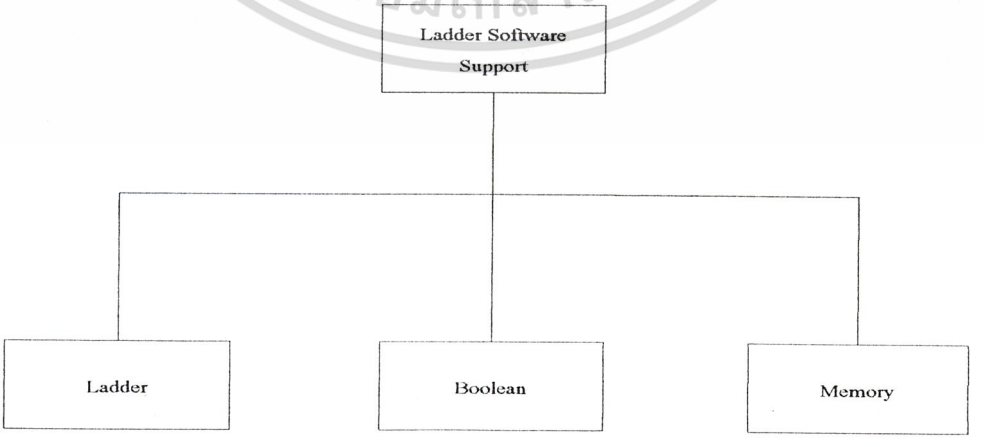
สารบัญรูป(ต่อ)

บทที่ 1
บทนำ

เนื่องจากในการพัฒนาระบบควบคุมแบบอัตโนมัติในปัจจุบัน ได้นำเอา PLC มาใช้ร่วมกันเพื่อควบคุมระบบต่างๆให้ทำงานได้โดยอัตโนมัติ เช่น ลิฟท์ สายพานลำเลียงสินค้า หรือ ระบบการผลิตต่างๆในโรงงานอุตสาหกรรม เป็นต้น ทำให้ผู้ควบคุมต้องเขียนโปรแกรมควบคุมเครื่อง PLC แต่ถ้าเขียนโปรแกรมจากเครื่อง PLC โดยตรง มักจะเกิดข้อผิดพลาดและแก้ไขโปรแกรมในภายหลังได้ยากอีกด้วย ทางผู้จัดทำจึงมีความคิดว่า ถ้าสามารถนำโปรแกรมที่เขียนลงในเครื่อง PLC มาเขียนบนคอมพิวเตอร์ก่อน แล้วจึงถ่ายโอนโปรแกรมลงเครื่องอีกทีหนึ่งก็จะเป็นการสะดวกแก่ผู้ใช้ PLC เพราะถ้าทำงานบนคอมพิวเตอร์จะสามารถดูโปรแกรมและคำสั่งต่างๆได้โดยง่าย และถ้าเขียนอยู่ในรูปแบบของภาพ Graphic แล้ว จะทำให้ผู้เขียนสามารถเข้าใจโปรแกรมที่ตนเขียนไว้ก่อนหน้านี้ รวมไปถึงการแก้ไขข้อบกพร่องของโปรแกรมของตนได้โดยง่าย ซึ่งการพัฒนาโปรแกรมขึ้นมานี้มีวัตถุประสงหลักดังนี้

1. ช่วยให้ผู้ใช้งานเครื่องควบคุม PLC สามารถเขียนโปรแกรมควบคุมได้อย่างสะดวกยิ่งขึ้น
2. ช่วยให้ผู้ใช้งานเครื่องควบคุม PLC สามารถเข้าใจวิธีการทำงานของโปรแกรมโดยอาศัยภาษา Ladder ได้
3. ช่วยประหยัดเวลาในการเขียนโปรแกรมเครื่องควบคุม PLC ได้

โครงงานนี้แบ่งการพัฒนาออกเป็น 3 ส่วน คือ ส่วนที่ 1 เป็นการเขียนโปรแกรมด้วยภาษา Ladder ในส่วนที่ 2 เป็นการเขียนโปรแกรมด้วยภาษา Boolean และส่วนสุดท้าย เป็นการแก้ไขค่าหน่วยความจำใน PLC ดังแสดงในรูป



รูปที่ 1.1 โครงสร้างโปรแกรม

บทที่ 2
ทฤษฎีและหลักการ

2.1 ที่มาและวิวัฒนาการของเครื่องควบคุมแบบตรรกที่โปรแกรมได้

การควบคุมแบบลำดับขั้น (Sequence Control) เป็นการควบคุมแบบหนึ่งที่ต้องการให้เครื่องจักรหรือระบบการทำงานตามช่วงเวลา ตามลำดับขั้นตอน ตลอดจนตามเงื่อนไขที่ได้กำหนดไว้โดยมีลักษณะของการควบคุมเป็นแบบ "ON" หรือ "OFF" การควบคุมแบบลำดับขั้นหรือแบบซีเควนซ์นี้จะพบ เห็นอยู่เสมอในงานอุตสาหกรรมแทบทุกชนิด ตัวอย่างของการควบคุมแบบนี้ได้แก่ ระบบควบคุมเครื่องจักร ระบบล้างรถอัตโนมัติ ระบบป้อนวัสดุ ระบบการการผสมวัสดุ ระบบควบคุมลิฟท์ เป็นต้น

ในอดีตที่ผ่านมาระบบควบคุมซีเควนซ์จะใช้อุปกรณ์ไฟฟ้าเชิงกล(Electromechanical Device) เช่น รีเลย์แม่เหล็กไฟฟ้า (Relay) ตัวตั้งเวลา (Timer) ตัวนับ (Counter) มาประกอบกันเป็นวงจรควบคุมเพื่อให้เครื่องจักรหรือระบบกระบวนการทำงานตามช่วงเวลาตามลำดับขั้นตอนตามเงื่อนไขที่วิศวกรหรือผู้ออกแบบระบบกำหนดได้อย่างไรก็ตามระบบควบคุมแบบซีเควนซ์ที่ใช้อุปกรณ์ไฟฟ้าเชิงกล ที่กล่าวมานี้มีข้อเสียมาก คือ ขนาดของวงจรควบคุมจะมีขนาดใหญ่สิ้นเปลืองทั้งเนื้อที่และพลังงานสูง ราคาแพง ไม่สามารถจะใช้กับระบบควบคุมที่มีขั้นตอนการทำงานยุ่งยากซับซ้อน เมื่อมีปัญหาเกิดขึ้นในวงจรควบคุมก็ตรวจสอบแก้ไขยาก การขยายระบบทำได้ยาก และที่สำคัญ คือถ้าต้องการเปลี่ยนแปลงลำดับขั้นตอนหรือเงื่อนไขของการทำงาน วงจรควบคุมแบบรีเลย์ที่มีอยู่เดิมจะเปลี่ยนแปลงเพื่อใช้กับงานใหม่ได้ยากต่อมาเมื่ออุปกรณ์สารกึ่งตัวนำ (Solid State Device) ได้แพร่หลาย และก้าวหน้าขึ้น จึงได้มีการนำเอาเกท (Gate) ต่างๆ มาประกอบกเป็นวงจรควบคุมแทนรีเลย์ และสามารถลดข้อเสียของระบบเดิมลงไปได้มาก เช่น วงจรควบคุมแทนรีเลย์ และสามารถลดข้อเสียการขยายระบบทำได้ง่ายขึ้น และยังสามารถต่อเข้ากับคอมพิวเตอร์ (Computer) เพื่อการเก็บข้อมูลและอื่นๆ ได้เป็นต้น แต่ก็ยังไม่สามารถจะลดข้อเสียบางอย่างที่สำคัญลงได้ นั่นคือการตรวจสอบแก้ไข เมื่อมีปัญหาก็ยังทำได้ยาก และไม่สามารถจะเปลี่ยนแปลงลำดับขั้นตอน หรือเงื่อนไขของการทำงาน จึงได้มีการสร้างเครื่องควบคุมที่สามารถกำหนดโปรแกรมการทำงาน (Programmable Controller) ขึ้นมาเพื่อให้นำมาประยุกต์ใช้กับระบบควบคุมแบบซีเควนซ์ได้ทุกระบบ โดยไม่ต้องเปลี่ยนแปลงแก้ไขวงจรควบคุมแต่อย่างใด

2.1.1 วิวัฒนาการของเครื่องควบคุมแบบตรรกที่โปรแกรมได้

PLC (Programmable Logic Controllers) และ PC(Programmable Controller) เป็นเครื่องควบคุมที่สามารถกำหนดโปรแกรมการทำงานได้ PLC/PC ที่ใช้ไมโครโปรเซสเซอร์ นอกจากจะใช้ในการควบคุมเครื่องจักรโดยทั่วไปแล้ว ยังมีการพัฒนาให้มีความสามารถและขอบเขตให้งานได้กว้างขวางขึ้น เช่น มีฟังก์ชันทางคณิตศาสตร์เพิ่มขึ้นทำให้การควบคุมเป็นได้ทั้งแบบ ON-OFF หรือแบบอนาล็อก (Analog) เช่น PID (Proportional +Integral +Derivative) สามารถต่อเข้ากับอุปกรณ์วัดและควบคุม (Instrumentation) อื่นๆ จำนวนของอินพุท / เอาท์พุท (Input/Output) ขยายให้มากขึ้นได้ หรือการใช้อินพุท/เอาท์พุทแบบรีโมท(Remote) เพื่อลดการเดินสาย การติดต่อสื่อสารข้อมูลกับคอมพิวเตอร์ ตลอดจนอุปกรณ์พิเศษอื่น ๆ เช่น เครื่องอ่านรหัสแถบเพื่อจำแนกหรือจัดทำฐานข้อมูล การเชื่อมโยงข้อมูลลักษณะโครงข่ายท้องถิ่น (LAN) เป็นต้น หน่วยความจำก็ขยายได้มาก ทำให้ใช้กับระบบกระบวนการใหญ่ๆได้ และมีการจัดการเกี่ยวกับข้อมูลได้ตามต้องการ

2.1.2 สรุปข้อดีของ PLC/PC

ข้อดีของ PLC/PC สามารถจำแนกออกได้เป็นข้อๆดังนี้

1. สิ้นเปลืองเนื้อที่น้อยเพราะมีขนาดเล็ก

2. สามารถจะใช้ควบคุมเครื่องจักรหรือระบบกระบวนการใดๆ ก็ได้ ถ้าเลือกขนาดของ PLC/PC ที่เหมาะสม

3. การเปลี่ยนลำดับขั้นตอนหรือเงื่อนไขของการทำงานก็ทำได้ตามต้องการ เพราะใช้หลักของการโปรแกรม

4. ตัวตั้งเวลาและตัวนับจะเป็นซอฟท์แวร์ ทำให้การกำหนดค่าต่างๆง่าย เปลี่ยนแปลงค่าได้ตลอดเวลาไม่ต้องมีฮาร์ดแวร์ร่วม และทำให้ราคาถูกลง

5. รีเลย์ภายใน(Internal relay) ก็เป็นซอฟท์แวร์เช่นเดียวกัน จึงลดค่าใช้จ่ายในการเดินสาย ลดฮาร์ดแวร์ และทำให้ขนาดเล็กลงด้วย

6. การติดตั้งทำ ได้ง่ายและสะดวก

7. ขยายระบบให้ใหญ่ขึ้นทำได้โดยง่าย

8. ราคาถูกกว่าระบบรีเลย์

9. ความน่าเชื่อถือ (Reliability) ดีเพราะเป็นอุปกรณ์สารกึ่งตัวนำ ไม่มีการเดินสายมาก ไม่มีปัญหาเกี่ยวกับหน้าสัมผัส (Contact) แบบรีเลย์

10. มีระบบตรวจสอบหาที่ผิดพลาดด้วยตัวเอง การตรวจสอบแก้ไขเมื่อมีปัญหาจึงทำได้เร็ว

11. ลดการเดินสายยาวๆ และลดค่าใช้จ่ายในการเดินสายได้ เพราะมีอินพุท/เอาท์พุทแบบรีโมท

12. การบำรุงรักษาทำได้ง่าย

13. เวลาในการทำงานเร็วกว่าระบบที่ใช้รีเลย์

14. มีฟังก์ชันทางคณิตศาสตร์ได้แก่ บวก ลบ คูณ หาร และอื่นๆ ทำให้สามารถใช้สำหรับการควบคุมแบบ ON-OFF หรือแบบอนาล็อก เช่น PIDได้

15. สามารถเชื่อมต่อกับอุปกรณ์วัด เช่น เทอร์โมคัปเปิล (Thermocouple) และอื่นๆได้นอกเหนือจากอุปกรณ์ตรวจวัดที่เป็นสวิทซ์ (Switch)

16. ต่อเข้ากับคอมพิวเตอร์เพื่อวัตถุประสงค์ใด ๆ เช่น การเก็บข้อมูลการกระจายการควบคุม(Distributed Control) เป็นต้น

17. การโปรแกรมทำได้หลายแบบ เช่น คำสั่งในรูปของแลดเดอร์ไดอะแกรม คำสั่งบูลีน (Boolean Instruction) คำสั่งในรูปบล็อค(Block Instruction)หรือคำสั่งภาษาเบสิค

18. ใช้ได้ในทุกสภาพแวดล้อมของงานอุตสาหกรรม

19. การโปรแกรมทำได้โดยใช้เครื่องป้อนโปรแกรม (Program Loader)หรือโปรแกรมลงบน CRT

จากที่กล่าวมานี้จะเห็นว่า PLC/PC เป็นเครื่องควบคุมที่มีประสิทธิภาพและมีประโยชน์อย่างยิ่งต่องานอุตสาหกรรมในปัจจุบันและในอนาคต

2.2 โครงสร้างและหลักการทำงานของเครื่องควบคุมแบบตรรกที่โปรแกรมได้

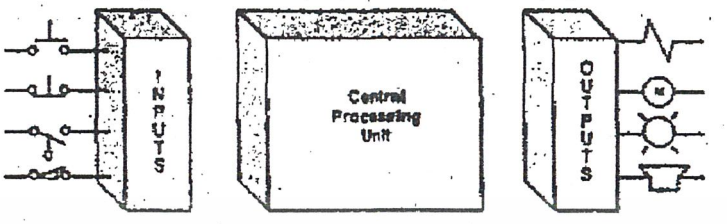
เครื่องควบคุมแบบตรรกที่โปรแกรมได้ หรือ PLC/PC เป็นอุปกรณ์สารกึ่งตัวนำ ที่ใช้สำหรับควบคุมเครื่องจักร หรือระบบกระบวนการให้ทำงานตามโปรแกรมคำสั่งของผู้ใช้ (User Program) และข้อมูลต่างๆ ที่ได้รับจากอินพุท/เอาท์พุทของ PLC/PC จะเป็นได้ทั้งการทำงานตามช่วงเวลา ตามลำดับขั้นตอนฟังก์ชันทางคณิตศาสตร์ และอื่นๆ PLC/PC มีส่วนประกอบสำคัญ 2 ส่วนคือ

1. หน่วยประมวลผลกลางหรือ CPU (Central Processing Unit)

2. หน่วยอินพุท/เอาท์พุท (Input/Output Unit)

นอกจากส่วนประกอบทั้งสองแล้ว PLC/PC ยังประกอบด้วยหน่วยป้อนโปรแกรม (Programming Unit) ผู้ใช้สามารถติดต่อกับ PLC/PC ยังประกอบด้วย หน่วยป้อนโปรแกรม หรือเปลี่ยนแปลงแก้ไขโปรแกรมคำสั่งตรวจสอบ สภาพการทำงานของเครื่องจักร หรือระบบกระบวนการ ตลอดจนตรวจสอบสภาพการทำงานของ PLC/PC ได้ทางหน่วยป้อนโปรแกรม เพื่อป้อนหรือเปลี่ยนแปลงแก้ไขโปรแกรม คำสั่งตรวจสอบสภาพการทำงานของเครื่องจักร หรือระบบ

กระบวนการ ตลอดจนตรวจสอบสภาพการทำงานของ PLC/PC รูปที่ 1 แสดงส่วนประกอบที่สำคัญทั้ง 2 ส่วนของ PLC/PC



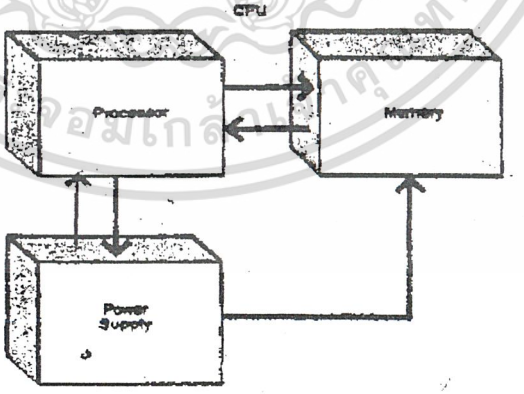
รูปที่ 2.1 บล็อคไดอะแกรมของ PLC/PC

2.2.1 หน่วยประมวลผลกลาง

หน่วยประมวลผลกลางหรือ CPU ประกอบด้วย 3 ส่วนคือ

1. หน่วยประมวลผล (Processor)
2. หน่วยความจำ (Memory)
3. หน่วยจ่ายกำลัง (Power Supply)

รูปที่ 2.2 แสดงส่วนประกอบของ CPU โดยทั่วไป CPU จะใช้ไมโครโปรเซสเซอร์ซึ่งเป็นไอซี (I.C.:Integrated Circuit) ที่มีความสามารถทั้งในการคำนวนทางคณิตศาสตร์ การจัดการข้อมูล และการตรวจสอบตัวเอง ในขณะที่วงจรควบคุมที่ใช้รีเลย์หรือไอซีพวกเกทต่างๆ ไม่สามารถจะทำได้ PLC/PC จึงมีข้อดีและประโยชน์มากกว่าระบบแบบเก่าอย่างเห็นได้ชัด

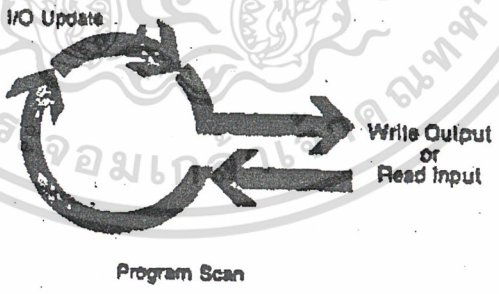


รูปที่ 2.2 บล็อคไดอะแกรมของ CPU

1. หน่วยประมวลผล

หน่วยประมวลผลของ CPU ทำหน้าที่ควบคุมและดูแลการทำงานของระบบทั้งหมด โดยรับข้อมูลจากหน่วยอินพุทมาทำการประมวลผลตามโปรแกรมคำสั่งของผู้ใช้ที่เก็บไว้ในหน่วยความจำ และส่งผลที่ได้ออกไปยังหน่วยเอาท์พุท ในปัจจุบันหน่วยประมวลผลของ CPU จะใช้ไมโครโปรเซสเซอร์ขนาดตั้งแต่ 4 บิท 8 บิท หรือ 16 บิท จำนวน 1 ตัว หรือหลายๆ ตัวมาทำงานร่วมกัน ในกรณีที่ใช้ไมโครโปรเซสเซอร์หลาย ๆตัวจะมีข้อดีคือ การทำงานของระบบจะเร็วขึ้นทำให้ PLC/PC มีขีดความสามารถสูงขึ้น เช่น การใช้ชุดเชื่อมต่อพิเศษที่มีไมโครโปรเซสเซอร์และหน่วยความจำภายในจะทำให้การทำงานควบคุมดียิ่งขึ้นและเป็นอิสระไม่ขึ้นกับ CPU ตัวอย่างก็คือชุดควบคุม PID ที่ใช้สำหรับควบคุมระบบแบบลูฟปิด (Closed-Loop Control) ซึ่งทำงานอิสระจาก CPU

การทำงานของ CPU 1 รอบ เริ่มจากการรับข้อมูลจากการรับข้อมูลจากหน่วยอินพุทเข้ามา ประมวลผลตามโปรแกรมคำสั่งของผู้ใช้ ส่งผลที่ได้ไปยังหน่วยเอาท์พุท ตรวจสอบการทำงานของระบบทั้งหมดและติดต่อกับผู้ใช้และส่งผลที่ได้ไปยังหน่วยเอาท์พุท ตรวจสอบการทำงานของระบบทั้งหมดและติดต่อกับผู้ใช้ เรียกว่าการ SCAN โดยทั่วไปเวลา SCAN ของ CPU จะใช้เวลาประมาณ 0.001 ถึง 0.1 วินาที เวลา SCAN จะอ้างอิงถึงขนาดของหน่วยความจำที่เก็บโปรแกรมของผู้ใช้เสมอ เช่น เวลา SCAN เท่ากับ 0.01 วินาทีต่อโปรแกรมคำสั่งขนาด 1 กิโลไบท์ (K-byte) อย่างไรก็ตามเวลา SCAN จะขึ้นอยู่กับองค์ประกอบอื่นๆด้วย เช่นการใช้ชุดอินพุท/เอาท์พุท แบบรีโมทจะทำให้เวลา SCAN ของ CPU เพิ่มขึ้น



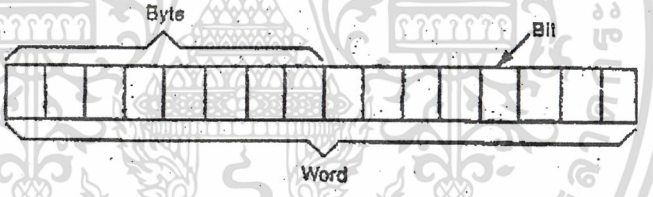
รูปที่ 2.3 การ SCANของ CPU

ช่วงเวลา SCAN ของ CPU จะทำให้ทราบถึงความสามารถในการตอบสนองต่อการเปลี่ยนแปลงของอินพุท/เอาท์พุท ของPLC/PC ว่ารวดเร็วเพียงไร เวลา SCAN จึงเป็นองค์ประกอบที่สำคัญอย่างหนึ่งในการเลือกใช้ PLC/PC ให้เหมาะสมกับงาน เช่น ถ้าต้องการที่จะตรวจสอบสัญญาณอินพุทที่เปลี่ยนค่าสภาวะถึง 2 ครั้งภายใน 0.008 วินาที การใช้PLC/PC ที่มีช่วงเวลา SCAN เท่ากับ 0.01 วินาที จะไม่สามารถตรวจสอบค่าสภาวะที่ถูกต้อง และอาจทำให้เครื่องจักร

หรือกระบวนการที่ควบคุมทำงานผิดพลาดได้ นอกจากนี้ หน่วยประมวลผลใน CPU ยังทำหน้าที่ในการติดต่อกับระบบย่อย (Subsystem) ตรวจสอบความผิดพลาดต่าง ๆเช่น การทำงานของหน่วยประมวลผล หน่วยความจำ หรือสภาพของ แบตเตอรี่จ่ายกำลังสำรอง เป็นต้น

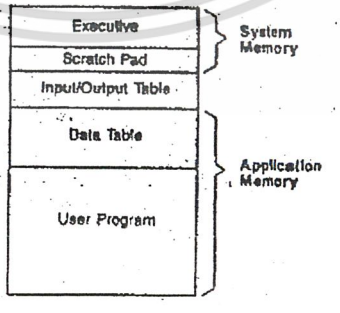
2. หน่วยความจำ

หน่วยความจำ เป็นที่เก็บโปรแกรมและข้อมูลที่หน่วยประมวลผลใช้ในการควบคุมการทำงานของ PLC/PC และ การทำงานตามโปรแกรมคำสั่งของผู้ใช้ ขนาดของหน่วยความจำ จะแบ่งออกเป็นบิทข้อมูลภายในหน่วยความจำ 1 บิท จะมีค่าสภาวะทาง ลอจิก “1” หรือ “0 ” ข้อมูลขนาด 8 บิท และ 16 บิท รวมกันเรียกว่าไบท์ (Byte) และ เวิร์ด(Word) ตามลำดับหน่วยความจำแบ่งออกได้เป็น 2 ประเภทใหญ่ๆ คือหน่วยความจำที่ข้อมูลสูญหายเมื่อไม่มีแหล่งจ่ายกำลัง (Volatile) และหน่วยความจำที่ข้อมูลคงอยู่แม้ไม่มีแหล่งจ่ายกำลัง (Non- Volatile) หรืออาจจะแบ่งเป็น 2 ชนิดใหญ่ๆ ตามคุณลักษณะคือ RAM (Random Access Memory) หรือ NOVRAM (Non-Volatile RAM) และ ROM(Read Only Memory) นอกจากนี้ ROM ยังจำแนกออกได้เป็น PROM (Programmable ROM) EPROM(Erasable Programmable ROM) EAROM (Electrically Alterable ROM) และ EEPROM (Electrically Erasable Programmable ROM)



รูปที่ 2.4 ข้อมูลเป็น บิท ไบท์ และเวิร์ด

หน่วยความจำของ PLC/PC แบ่งออกเป็น 4 ส่วน ตามรูปที่ 2.5 เพื่อใช้เก็บข้อมูลและโปรแกรมต่างๆคือ



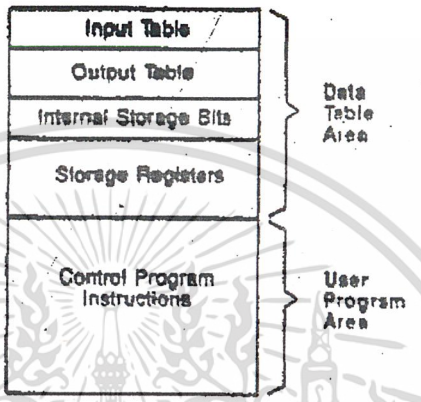
รูปที่ 2.5 หน่วยความจำของ PLC/PC

1. โปรแกรมจัดการ (Execute program) เป็นโปรแกรมที่ทำหน้าควบคุมดูแลและตรวจสอบการทำงานของ PLC/PC ทั้งหมด

2. ข้อมูลชั่วคราว (Processor Work Area หรือ Scratch pad) หน่วยความจำส่วนที่สองทำหน้าที่เก็บข้อมูลที่เกิดขึ้นระหว่าง PLC/PC ทำงานตามโปรแกรมจัดการและโปรแกรมคำสั่งของผู้ใช้

3. ตารางข้อมูล (Data Table) หน่วยความจำส่วนที่สามทำหน้าที่เก็บ ค่าของอินพุทเอาท์พุท และตัวแปรต่างๆ จากการทำงานตามโปรแกรมคำสั่ง ของผู้ใช้ ทั้งที่เป็นค่า สภาวะทางลอจิกและตัวเลข

4. หน่วยความจำสำหรับเก็บโปรแกรมคำสั่งของผู้ใช้ (User Program Memory)



รูปที่ 2.6 หน่วยความจำสำหรับผู้ใช้

หน่วยความจำของผู้ใช้จะแตกต่างกันตามขนาดของ PLC/PC ใน PLC หรือ PC ขนาดใหญ่ขนาดของหน่วยความจำดังกล่าวจะสามารถเปลี่ยนแปลงขนาดได้ตามต้องการ การเลือกใช้ PLC/PC จึงต้องคำนึงถึงขีดความสามารถและขีดจำกัดต่างๆประกอบด้วย เช่น ขนาดโปรแกรมคำสั่งของผู้ใช้จำนวนอินพุท/เอาท์พุท ที่จะขยายได้สูงสุด ขนาดของหน่วยความจำภายในที่ใช้เก็บข้อมูล และฟังก์ชันพิเศษต่างๆ เช่น รีเลย์ภายใน ตัวตั้งเวลา และตัวนับ

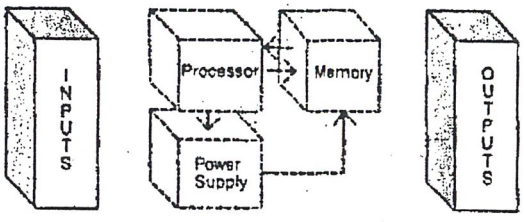
3. หน่วยจ่ายกำลัง

หน่วยจ่ายกำลังจะทำหน้าที่จ่ายและรักษาระดับแรงดันไฟฟ้ากระแสตรง (D.C Voltage) ให้กับหน่วยประมวลผล หน่วยความจำ และหน่วยอินพุท/เอาท์พุท ตามความต้องการ และทำหน้าที่เตือนให้หน่วยประมวลผลทราบเมื่อเกิดปัญหา

2.2.2 หน่วยอินพุท/เอาท์พุท

หน่วยอินพุททำหน้าที่เชื่อมต่อระหว่าง CPU กับอุปกรณ์ภายนอกโดยรับค่าสภาวะ หรือปริมาณทางกายภาพต่างๆ จากอุปกรณ์ตรวจวัด (Sensor) ของเครื่องจักรหรือกระบวนการ เช่น ลิมิตสวิทช์ (Limit Switch) พร็อกซิมิตี้สวิทช์ (Proximity Switch) ตำแหน่ง อุณหภูมิ ระดับ แรงดัน กระแสไฟ และอื่นๆ ส่งไปยัง CPU เพื่อประมวลผลตามโปรแกรมคำสั่งของผู้ใช้หน่วยเอาท์พุททำหน้าที่รับค่าสภาวะหรือคำสั่งควบคุมที่ได้จาก CPU เพื่อส่งไปควบคุมอุปกรณ์ภายในเครื่องจักร

หรือกระบวนการเช่น วาล์ว (Valve) มอเตอร์ (Motor) ปั๊ม (Pump) และอื่นๆให้ทำงานตามค่าสภาวะหรือคำสั่งที่ CPU ต้องการ



รูปที่ 2.7 บล็อคไดอะแกรมของหน่วยอินพุท/เอาท์พุท

ดังที่ได้กล่าวมาแล้วว่า ในช่วงแรก PLC/PC ถูกใช้แทนรีเลย์สำหรับการควบคุมแบบ ON-OFF หน่วยอินพุท/เอาท์พุทในขณะนั้นจึงเป็นแบบที่ใช้สำหรับค่าสภาวะลอจิก (Discrete Input / Output) เท่านั้น แต่ต่อมาเมื่อ PLC/PC ได้ถูกพัฒนาจนมีการควบคุมแบบอนาล็อกรวมอยู่ด้วย โดยใช้หน่วยอินพุท/เอาท์พุท ที่ถูกพัฒนาขึ้นเพื่อรับ-ส่งสัญญาณแบบอนาล็อคโดยเฉพาะ ปัจจุบันหน่วยอินพุทและเอาท์พุท ของ PLC/PC อาจแบ่งได้ดังนี้คือ

- อินพุท/เอาท์พุทแบบค่าสภาวะลอจิก
- อินพุท/เอาท์พุทแบบค่าตัวเลข (Numerical Data Input/Output)
- อินพุท/เอาท์พุทแบบหลายบิต
- อินพุท/เอาท์พุทแบบพิเศษ (Special Input/Output)

1. อินพุท/เอาท์พุทแบบค่าสภาวะลอจิก

อินพุท/เอาท์พุทแบบนี้จะใช้กันมากที่สุด ข้อมูลจะมีค่าสภาวะอยู่ 2 สภาวะคือ “ 1 ” หรือ “ ON ” หมายถึงสวิทซ์ หรือหน้าสัมผัสของรีเลย์แตะกัน และสภาวะ “ O ” หรือ “ OFF ” หมายถึงสวิทซ์หรือหน้าสัมผัสของรีเลย์เปิดออก

เนื่องจากระดับแรงดันไฟฟ้าทางด้านอินพุทมีหลายแบบ การเลือกใช้หน่วยอินพุท/เอาท์พุท แบบค่าสภาวะลอจิกจึงต้องเลือกใช้ให้ถูกต้อง ตารางที่ 2.1 แสดงให้เห็นตัวอย่างของอุปกรณ์อินพุท/เอาท์พุทแบบลอจิก

ตารางที่ 2.1 อุปกรณ์ อินพุท/เอาท์พุทแบบค่าสภาวะลอจิก

อุปกรณ์ทางด้านอินพุท	อุปกรณ์ทางด้านเอาท์พุท
Selector Switches	Alarms
Push Buttons	Control Relays
Photoelectric Eyes	Fans
Limit Switches	Lights
Circuit Breakers	Horns

2. อินพุท/เอาท์พุทแบบค่าตัวเลข

หน่วยอินพุท/เอาท์พุทแบบค่าตัวเลขนี้จะมีอยู่ใน PLC/PC ตั้งแต่ขนาดกลางที่มีฟังก์ชั่นการคำนวณทางคณิตศาสตร์รวมอยู่ด้วย โดยทั่วไปแล้ว หน่วยอินพุท/เอาท์พุทแบบค่าตัวเลขนี้จะแบ่งเป็น 2 ชนิดคือหน่วยอินพุท/เอาท์พุท สำหรับสัญญาณอนาล็อก และหน่วยอินพุท/เอาท์พุทสำหรับค่าสภาวะแบบหลายบิท

หน่วยอินพุทสำหรับสัญญาณอนาล็อกจะรับสัญญาณอินพุทจากอุปกรณ์ต่าง ๆ และหน่วยเอาท์พุทจะส่งสัญญาณอนาล็อกไปยังอุปกรณ์ต่างๆ ตาราง 2.2 แสดงถึงอุปกรณ์อินพุท/เอาท์พุท บางแบบ

ตารางที่ 2.2 อุปกรณ์อินพุท/เอาท์พุทแบบค่าตัวเลข

อุปกรณ์อินพุทแบบอนาล็อก	อุปกรณ์เอาท์พุทแบบอนาล็อก
Temperature Transducers	Analog Valves and Actuators
Pressure Transducers	Chart Recorders
Load Cell Transducers	Electric Motor Drives
Humidity Transducers	Analog Meters
Flow Transducers	

3. อินพุท/เอาท์พุทสำหรับค่าสภาวะแบบหลายบิท

หน่วยอินพุท/เอาท์พุทสำหรับค่าสภาวะแบบหลายบิท นี้จะทำงานเหมือนกับหน่วยอินพุท/เอาท์พุทแบบค่าสภาวะลอจิกทุกอย่าง ต่างกันก็แต่เพียงการรับ-ส่งข้อมูล ซึ่งสามารถทำได้ครั้งละหลายๆ บิทในเวลาเดียวกัน เช่น อินพุท/เอาท์พุทแบบ BCD (Binary Code Decimal) เป็นต้น ตารางที่ 2.3 แสดงถึงอุปกรณ์ที่ใช้ร่วมกับหน่วยอินพุท/เอาท์พุทสำหรับค่าสภาวะแบบหลายบิท

ตารางที่ 2.3 อุปกรณ์อินพุท/เอาท์พุทสำหรับค่าสภาวะแบบหลายบิท

อุปกรณ์อินพุทแบบหลายบิท	อุปกรณ์เอาท์พุทแบบหลายบิท
Thumbwheel Switches	Seven - Segment Displays
Bar Codes Readers	Intelligent Display

4. อินพุท/เอาท์พุทแบบพิเศษ

สำหรับ PC/PLC ตั้งแต่ขนาดกลางขึ้นไปจะสามารถนำเอาหน่วยอินพุท /เอาท์พุทแบบพิเศษมาต่อเพื่อใช้งานเฉพาะด้านได้ อินพุท/เอาท์พุทสำหรับงานเฉพาะด้านนั้นมีหลายอย่างดังต่อไปนี้

4.1 อินพุทที่เป็นเทอร์โมคัปเปิล (Thermocouple Input) หน่วยอินพุทสำหรับเทอร์โมคัปเปิลนี้จะทำหน้าที่ปรับระดับสัญญาณ (Signal Conditioning) ที่รับมาโดยตรงจากเทอร์โมคัปเปิลให้มีระดับที่เหมาะสม การทำงานของหน่วย อินพุทชนิดนี้จะเหมือนกับหน่วยอินพุทสัญญาณอนาล็อก จาก เทอร์โมคัปเปิลนี้จะถูกกรอง ขยาย และทำให้เป็นสัญญาณดิจิตอล จากนั้นถูกส่งไปยังหน่วยประมวลผลเพื่อการทำงานตามโปรแกรมคำสั่งของผู้ใช้

4.2 อินพุทที่เป็นพัลซ์หน่วยอินพุท สำหรับรับอินพุทที่เป็นพัลซ์นี้จะทำหน้าที่ตรวจรับสัญญาณที่มีการเปลี่ยนแปลงเร็วกว่า 1 ช่วงเวลาของการ SCAN ของ CPU

4.3 อินพุท/เอาท์พุทที่เป็นรหัส ASCII หน่วยอินพุท/เอาท์พุทแบบนี้ใช้สำหรับรับ-ส่งข้อมูลที่เป็นตัวอักษรใน ASCII

4.4 อินพุทที่เป็นสเตนเกจ (Strain Gage Input) หน่วยอินพุทสำหรับสัญญาณอนาล็อกจากสเตนเกจเหมือนกับหน่วยอินพุทสำหรับเทอร์โมคับเปิล แตกต่างกันที่วงจรปรับระดับสัญญาณซึ่งถูกออกแบบไว้รับสัญญาณจากสเตนเกจเท่านั้น

4.5 เอาท์พุทที่เป็นพัลซ์ (pulse output) หน่วยเอาท์พุทนี้ทำหน้าที่ส่งพัลซ์ไปควบคุมการทำงานของสเตปปิ้งมอเตอร์ โดยเฉพาะ ตำแหน่งของมอเตอร์จะถูกกำหนดจากจำนวนของพัลซ์เอาท์พุทที่ถูกนับไว้ก่อน และคำสั่งให้หมุนไปในทิศทางทวนเข็มนาฬิกาหรือตามเข็มนาฬิกา

4.6 หน่วยเชื่อมต่อเซอร์โว (Servo Interface) หน่วยเชื่อมต่อเซอร์โวเป็นหน่วยอินพุท/เอาท์พุทที่ถูกออกแบบขึ้นเพื่อตรวจสอบและควบคุมตำแหน่งของเครื่องจักรโดยเฉพาะ เช่น การควบคุมในระบบ NC (Numerical Control)

4.7 หน่วย APM (Axis Positioning Module) หน่วย APM เป็นหน่วยเชื่อมต่อของ PC ที่ได้รับการออกแบบขึ้นเพื่อใช้สำหรับตรวจสอบ และควบคุมตำแหน่งของเครื่องจักร เช่น เดียวกับหน่วยเชื่อมต่อเซอร์โว แต่สามารถจะใช้ร่วมกับการควบคุมแบบซีเควนซ์ได้โดยทีการทำงานที่แยกเป็นอิสระจาก CPU

4.8 หน่วย PID เป็นหน่วยอินพุท/เอาท์พุทที่ได้รับการออกแบบเพื่อใช้ควบคุมในระบบลูฟปิดที่ต้องการฟังก์ชัน การควบคุมแบบ Proportional ,Integral ,Derivative

4.9 หน่วยเชื่อมต่อโครงข่าย(Network Interface Module) หน่วยเชื่อมต่อนี้ถูกออกแบบขึ้นเพื่อเพิ่มสมรรถนะการทำงานของ PC ให้สูงขึ้น โดยช่วยให้ PC หลายๆ ระบบเชื่อมต่อกันและติดต่อกันได้ หรือติดต่อกับระบบคอมพิวเตอร์เพื่อใช้เป็นหน่วยควบคุมหลัก

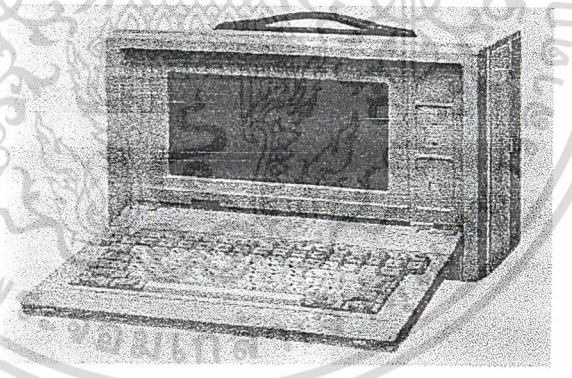
4.10 หน่วยอินพุท/ เอาท์พุทแบบรีโมท หน่วยอินพุท/ เอาท์พุทแบบรีโมท จะใช้กับPC ขนาดใหญ่ที่มี อินพุท / เอาท์พุท จำนวนมากๆ และมีระบบย่อยๆ หลายระบบ เพื่อทำหน้าที่ตรวจสอบสภาวะของอุปกรณ์จากจุดต่างๆ หลายจุดในเวลาเดียวกัน ภายใต้การควบคุมของ CPU

2.2.3 หน่วยป้อนโปรแกรม

หน่วยป้อนโปรแกรมของ PLC/PC ทำหน้าที่ติดต่อระหว่างผู้ใช้และ PLC/PC ทั้งระบบ คือ การป้อนโปรแกรมคำสั่งของผู้ใช้เข้าสู่หน่วยความจำของ CPU ตรวจสอบสภาวะการทำงาน รับ-ส่งข้อมูลระหว่าง CPU กับหน่วยป้อนโปรแกรม รับส่งข้อมูลระหว่าง CPU กับเทป หรืออื่นๆ ตามฟังก์ชั่นการใช้งานที่มีอยู่ หน่วยป้อนโปรแกรมของ PLC/PC แบ่งออกได้หลายชนิดคือ

1. เครื่องป้อนโปรแกรมแบบ CRT

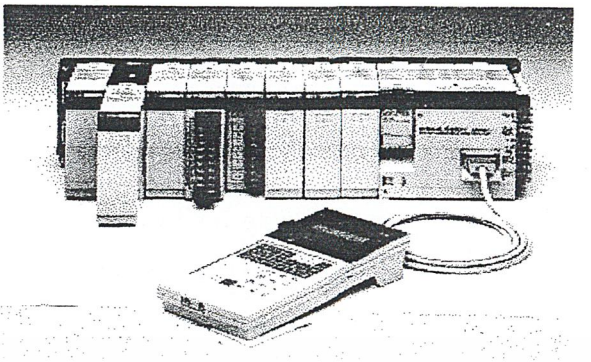
หน่วยป้อนโปรแกรม CRT ประกอบด้วยจอภาพและแป้นพิมพ์ เป็นเครื่องป้อนโปรแกรมแบบที่สะดวกที่สุด เครื่องป้อนโปรแกรมแบบนี้แบ่งออกเป็น 2 ชนิดคือ แบบดัมบ์ CRT (Dumb CRT) ซึ่งเป็นแบบที่ไม่มีหน่วยประมวลผลอยู่ภายในตัว การทำงานทุกอย่างจะถูกควบคุมจาก CPU ของ PLC/PC และแบบอินเทลลิเจนต์ CRT (Intelligent CRT) ซึ่งจะมีหน่วยประมวลอยู่ภายในตัวการทำงานเป็นอิสระจาก CPU จึงมีประสิทธิภาพดีกว่าแบบ ดัมบ์



รูปที่ 2.8 เครื่องป้อนโปรแกรมแบบ CRT

2. เครื่องป้อนโปรแกรมขนาดเล็ก (Mini Programmer)

เป็นเครื่องป้อนโปรแกรมขนาดเล็กที่สามารถพกติดตัวและเคลื่อนย้ายไปมาได้สะดวก แต่ประสิทธิภาพจะด้อยกว่าเครื่องป้อนโปรแกรมแบบ CRT ปกติแล้วจะใช้จอ LCD ในการแสดงผล เครื่องป้อนโปรแกรมแบบนี้จะแบ่งออกเป็น 2 ชนิดคือ แบบอินเทลลิเจนต์ และ แบบดัมบ์



รูปที่ 2.9 เครื่องป้อนโปรแกรมขนาดเล็กหรือพกพา

3. เครื่องป้อนโปรแกรมลงในหน่วยหน่วยความจำของ PLC/PC หรือ เทป

เครื่องป้อนโปรแกรมแบบนี้สามารถใช้ป้อนหรือแก้ไขโปรแกรมคำสั่งของผู้ใช้เข้าไปในหน่วยความจำของ PLC/PC ได้โดยตรง หรือใช้ในการรับ-ส่ง โปรแกรมและส่งข้อมูลต่าง ๆ ระหว่าง PLC/PC และเทป

4. เครื่องป้อนโปรแกรมลงบนหน่วยความจำ (Memory Burner)

เครื่องป้อนโปรแกรมแบบนี้ทำหน้าที่ถ่ายโปรแกรมที่มีอยู่ลงเก็บไว้ในหน่วยความจำแบบ ROM เพื่อให้โปรแกรมต่างๆ คงอยู่ตลอดเวลา เมื่อโปรแกรมต่างๆ ได้ถูกตรวจสอบและแก้ไขจนเรียบร้อยสมบูรณ์แล้ว

5. คอมพิวเตอร์

เครื่องควบคุม PLC/PC บางรุ่นได้ออกแบบให้ สามารถติดต่อกับคอมพิวเตอร์ได้เพื่อให้การโปรแกรมทุกอย่างสามารถทำบนคอมพิวเตอร์ได้

2.3 ลักษณะการโปรแกรมให้กับเครื่องควบคุมแบบตรรกที่โปรแกรมได้

ภาษาที่ใช้ในการเขียนโปรแกรมคำสั่งสำหรับ PLC/PC นั้นสามารถจะจำแนกออกได้เป็น 4 ภาษาคือ

1. ภาษาแลดเดอร์ไดอะแกรม
2. ภาษาคำสั่งบูลีน
3. ภาษาคำสั่งในรูปบล็อค
4. ภาษาระดับสูง

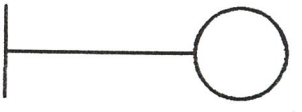
ภาษาแลดเดอร์ไดอะแกรมหรือคำสั่งบูลีนเป็นภาษาที่นิยมใช้และเข้าใจ ได้ง่ายสำหรับการเขียนโปรแกรมคำสั่งของ PLC/PC ในการควบคุมแบบซีเควนซ์ที่เป็น ON-OFF การเขียนโปรแกรมคำสั่งในภาษาแลดเดอร์ ไดอะแกรมหรือคำสั่งบูลีนจึงใช้กันมากใน PLC หรือ PC ทุก

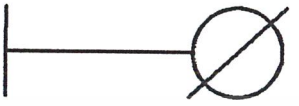
ขนาด ส่วนภาษาคำสั่งในรูปแบบบล็อก หรือ ภาษาระดับสูง เช่นภาษาซี ภาษาปาสคาล ภาษาฟอร์แทรน นั้นจะเหมาะสำหรับ PLC /PC ในการควบคุมแบบอนาล็อกการจัดการเกี่ยวกับข้อมูลการทำรายงานและอื่นๆ ซึ่งมักจะเป็น PLC ขนาดใหญ่ PLC แต่ละแบบอาจใช้คำสั่งเพียงภาษาเดียวหรือหลายภาษารวมกันในการเขียนโปรแกรมคำสั่งก็ได้ เช่นภาษาแลดเดอร์ไดอะแกรมร่วมกับคำสั่งบูลีนเพียงภาษาเดียวหรือใช้ภาษาแลดเดอร์ไดอะแกรมหรือคำสั่งใน รูปบล็อค ภาษาคำสั่งบูลีนร่วมกับคำสั่งในรูปบล็อคหรือภาษาคำสั่งบูลีน ร่วมกับภาษาระดับสูง เป็นต้น อย่างก็ตามในหัวข้อนี้จะกล่าวถึงการเขียนโปรแกรมคำสั่งของ PLC/PC โดยใช้ภาษาแลดเดอร์ไดอะแกรมและคำสั่งบูลีนเท่านั้น

2.3.1 โปรแกรมภาษาแลดเดอร์ไดอะแกรม

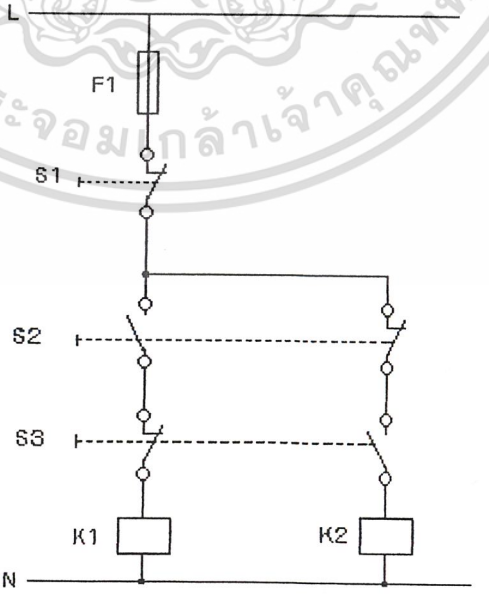
ดังที่ทราบกันแล้วว่า PLC/PC เป็นเครื่องควบคุมที่ใช้วิธีการโปรแกรม และใช้ซอฟท์แวร์แทนอุปกรณ์ต่างๆ เช่น รีเลย์ ตัวตั้งเวลา ตัวนับ ดังนั้นการเขียนโปรแกรมคำสั่งจึงใช้สัญลักษณ์พื้นฐานต่อไปนี้แทนอุปกรณ์ต่างๆ

ตารางที่ 2.4 แสดงถึงตัวอย่างของการใช้ฟังก์ชั่นลอจิกและสัญลักษณ์พื้นฐานในการเขียนโปรแกรมแลดเดอร์ของ PLC/PC

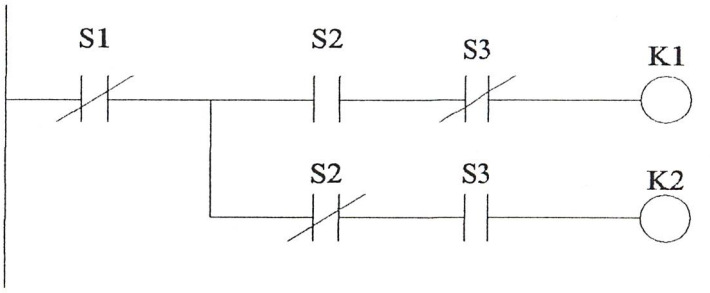
สัญลักษณ์	ความหมาย
	จะใช้แทนอุปกรณ์ทางด้านอินพุทต่างๆ ได้แก่ สวิทซ์ ไฟฟ้าหรือหน้าสัมผัสของรีเลย์ ตัวตั้งเวลา ตัวนับ หรือรีเลย์ภายในเป็นต้น โดยที่อุปกรณ์นั้นปกติจะมีค่าสภาวะเป็น “O”
	หน้าสัมผัสปกติเปิดหรือ N.O. จะใช้แทนอุปกรณ์ทางด้านอินพุทต่างๆ ได้แก่ สวิทซ์ ไฟฟ้าหรือหน้าสัมผัสของรีเลย์ ตัวตั้งเวลา ตัวนับ หรือรีเลย์ภายในเป็นต้น โดยที่อุปกรณ์นั้นปกติจะม่ค่าสภาวะเป็น “1” เอาท์พุทปกติไม่ทำงาน
	จะใช้แทนอุปกรณ์ทางด้านเอาท์พุทต่างๆ ได้แก่ หลอดไฟฟ้ามอเตอร์ไฟฟ้า ขดลวดของรีเลย์ ขดลวดโซลีนอยด์ (Solenoid) ตัวตั้งเวลา ตัวนับ รีเลย์ภายใน เป็นต้น โดยที่

	ปกติแล้วอุปกรณ์ทางด้านเอาท์พุทจะมีค่าสภาวะเป็น“0”
	เอาท์พุทปกติทำงาน จะใช้แทนอุปกรณ์ทางด้านเอาท์พุทต่างๆ ได้แก่ หลอดไฟฟ้า มอเตอร์ไฟฟ้า ขดลวดของรีเลย์ ขดลวดโซลีนอยด์ (Solenoid) ตัวตั้งเวลา ตัวนับ รีเลย์ภายใน เป็นต้น โดยที่ปกติแล้วอุปกรณ์ทางด้านเอาท์พุทจะมีค่าสภาวะเป็น “1”

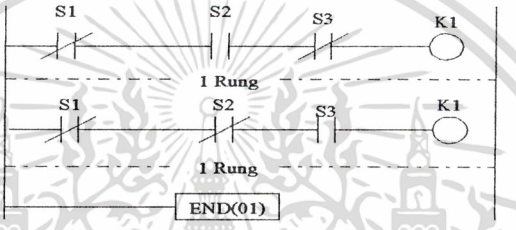
ภาษาแลดเดอร์ไดอะแกรม เป็นการโปรแกรมที่เข้าใจได้ง่าย เนื่องจากมีลักษณะคล้ายกับรีเลย์ไดอะแกรม จึงทำได้ง่าย ดังตัวอย่างแสดงให้เห็นความสัมพันธ์ระหว่าง รีเลย์ไดอะแกรมซึ่งเป็นวงจรควบคุมการกลับทางหมุนของมอเตอร์แบบ JOGGING ในรูปที่10 และเปลี่ยนมาใช้ PLC/PC ควบคุมโดยเขียนเป็นโปรแกรมแลดเดอร์ไดอะแกรม ในรปที่ 11 แลดเดอร์ ไดอะแกรม จะแบ่งออกเป็นส่วน ๆ แต่ละส่วนคือ เอาท์พุท 1 จุดหรือเอาท์พุทมากกว่า 1 จุด แต่เอาท์พุทเหล่านี้จะมีลอจิกเหมือนกันแต่ละส่วนเหล่านี้เรียกว่า “รังค์ (RUNG)” ดังในรูปที่ 12 เป็นโปรแกรมที่ให้ผลการควบคุมเหมือนเดิม แต่แยกออกเป็น 2 รังค์



รูปที่ 2.10 วงจรรีเลย์ควบคุมการหมุนกลับทางของมอเตอร์แบบ JOGGING



รูปที่ 2.11 แลดเดอร์ไดอะแกรมควบคุมการหมุนกลับทางของมอเตอร์แบบ JOGGING



รูปที่ 2.12 แสดงการแบ่งแลดเดอร์ไดอะแกรมออกเป็นรังค์

การโปรแกรมด้วยแลดเดอร์ไดอะแกรมจะสะดวกและมีประสิทธิภาพมากแต่ข้อเสียของการโปรแกรมแบบนี้คืออุปกรณ์ที่ใช้ในการเขียนโปรแกรมจะต้องมีหน่วยแสดงผลที่สามารถแสดงอักษรหรืออุปกรณ์ที่ใช้ในการโปรแกรมแบบนี้มักจะใช้จอ ซีอาร์ที่ (CRT : CATHODE RAY TUBE) แต่ปัจจุบันจะมีซอฟท์แวร์สนับสนุนการเขียนแลดเดอร์บนคอมพิวเตอร์และเชื่อมโย่งข้อมูลกับ PLC/PC ทางพอร์ทอนุกรมโดยคอมพิวเตอร์ทำหน้าที่ในการสร้าง และแก้ไขแลดเตอร์ไดอะแกรมแล้วทำการแปลเป็นชุดคำสั่งของเครื่องควบคุมนอกจากนั้นแล้วขณะที่เครื่องควบคุมทำงานอยู่ก็สามารถตรวจสอบการทำงานได้โดยการแสดงผลซึ่งจะแสดงเป็นแลดเดอร์ไดอะแกรมและเมื่ออินพุท หรือเอาท์พุทตัวใดมีการเปลี่ยนแปลง สภาวะก็จะมีการเปลี่ยนแปลงตามสภาวะของอินพุทหรือเอาท์พุทตัวนั้นๆ ด้วย

2.3.2 โปรแกรมภาษาคำสั่งแบบบูลีน

การโปรแกรมชนิดนี้มีลักษณะคล้ายกับสัญลักษณ์ของพีชคณิตบูลีนนอกจากจะเป็นสัญลักษณ์ ของพีชคณิตบูลีนแล้วก็ยังมีการใช้ฟังก์ชั่นพิเศษต่างๆ อีกมากเช่น ตัวนับ ตัวตัวเวลา เป็นต้น ภาษาคำสั่งบูลีน จะมีความสัมพันธ์กันกับ ภาษาแลดเดอร์ไดอะแกรม สามารถที่จะแปลความหมายถึงกันได้ดังตัวอย่างในรูปที่ 13 ซึ่งเป็นชุดคำสั่งบูลีนที่เขียนจากแลดเดอร์ไดอะแกรมในรูปที่ 12 การเขียนโปรแกรมเป็นแลดเดอร์ไดอะแกรมสามารถแสดงกระบวนการของโปรแกรมได้

ง่ายต่อการเข้าใจ แต่ไม่สะดวกในการป้อนลงบนเครื่องควบคุม จึงมีการแปลงจากแลดเดอร์ไดอะแกรม เป็นภาษาคำสั่งบูลีน เพื่อ ป้อนลงชุดรับข้อมูลแบบพกพาของเครื่องควบคุม PLC/PC ซึ่งจะกระทำได้สะดวกมากขึ้น

0000	LD NOT	S1
0001	AND	S2
0002	AND NOT	S3
0003	OUT	K1
0004	LD NOT	S1
0005	AND NOT	S2
0006	AND	S3
0007	OUT	K2
0008	END	

รูปที่ 2.13 โปรแกรมชุดคำสั่งบูลีน

2.4 รูปแบบของโปรโตคอล(Protocols Type)

ในการสื่อสารข้อมูลจะต้องมีกฎหรือข้อกำหนดในการสื่อสารข้อมูล หรือที่นิยมเรียกว่า โปรโตคอล (Protocols) ซึ่งเป็นส่วนที่จะกำหนดมาตรฐานในการควบคุม และจัดการระบบการสื่อสารข้อมูลสำหรับรายละเอียดที่กล่าวในที่นี้จะเกี่ยวข้อเฉพาะในส่วนของโปรโตคอลการควบคุมการเชื่อมโยงข้อมูล (Data Link Control Protocols หรือ DLCP) ซึ่งจัดการในส่วนของขั้นตอนและหลักการต่างๆคือโครงสร้างและรายละเอียดของข้อมูล วิธีในการสื่อสารข้อมูลการตรวจสอบแก้ไขความผิดพลาดของข้อมูล และขบวนการในการควบคุมการติดต่อสื่อสาร โดย DLCP แบ่งได้ตามโครงสร้างของข้อมูล 2 แบบ คือ Byte-Oriented Protocols และ Bit-Orientes Protocols

2.4.1 ไบท์โอเรียนโปรโตคอล (Byte-oriented peotocols)

โปรโตคอลแบบนี้เป็นโปรโตคอลที่การสื่อสารข้อมูล และการควบคุมการทำงานจะทำโดยใช้ลักษณะข้อมูลที่เป็นตัว (character) หรือ ไบท์ (byte) หรืออาจเรียกว่า Character Orientred Protocols ซึ่งแบ่งออกเป็น

1. อะซิงโครนัสโปรโตคอล (Asynchronous protocols)

โปรโตคอลในการสื่อสารข้อมูลนี้จะใช้การสื่อสารข้อมูลแบบ Half-Duplex ที่มีลักษณะการสื่อสารข้อมูลแบบอะเชิงโครนัส ซึ่งเป็นการสื่อสารข้อมูลแบบพื้นฐานที่ใช้งานมาเป็นเวลานานแล้ว จึงมีรายละเอียดและขั้นตอนในการสื่อสารข้อมูลที่ทำให้มีโอกาสเกิดความผิดพลาดได้น้อยยังมีข้อดีที่การสื่อสารข้อมูลแบบนี้มีในโครงสร้างการทำงานที่ง่าย อุปกรณ์ที่ใช้ในการสื่อสารข้อมูลก็ไม่สลับซับซ้อนและมีราคาถูก โปรโตคอลแบบนี้จึงเหมาะสำหรับใช้ในระบบขนาดเล็ก

2. ไบนารี่ซิงโครนัสโปรโตคอล (Binary synahronius protocols)

โปรโตคอลแบบนี้จะมีลักษณะการทำงานที่ใช้งานข้อมูลเป็นลักษณะไบท์ และยังคงใช้การสื่อสารข้อมูลแบบซิงโครนัส มีรายละเอียดและขั้นตอนในการสื่อสารข้อมูลที่ให้ความน่าเชื่อถือมากกว่า อีกทั้งยังสามารถใช้อัตราเร็วในการสื่อสารข้อมูลที่สูงกว่าโดยตัวอย่างจองการสื่อสารข้อมูลแบบนี้ที่ได้กำหนดเป็นมาตรฐานแล้วคือ การสื่อสารข้อมูลตามมาตรฐาน BSC (Binary Synchronous Communications) ซึ่งเป็นโปรโตคอลที่มีลักษณะของข้อมูลแบบไบท์ที่ได้รับความนิยมนำไปใช้งาน

2.4.2 บิทโอเรียนท์โปโตคอล (Bit-oriented protocols)

โปรโตคอลแบบนี้เป็นโปรโตคอลที่การสื่อสารข้อมูลและการควบคุมการทำงานจะทำโดยใช้ลักษณะข้อมูลที่เป็นบิทโดยมีตัวอย่างของการสื่อสารข้อมูลในลักษณะนี้ที่มีการกำหนดขึ้นเป็นมาตรฐานแล้ว คือ HDLC (High-level Data Link Control) โดยมีโครงสร้างของข้อมูลแบบซิงโครนัสเช่นเดียวกับ BSC แต่ต่างกันที่มีลักษณะของข้อมูลเป็นแบบบิท ซึ่งโปรโตคอลแบบนี้ข้อดีที่สามารถสื่อสารข้อมูลแบบ full-duplex ได้ทำให้การสื่อสารข้อมูลได้รวดเร็วกว่าแต่โปรโตคอลแบบนี้ก็มีรายละเอียดและโครงสร้างในการสื่อสารข้อมูลที่สลับซับซ้อนมากทำให้การควบคุมการทำงานทำได้ยากและต้องใช้อุปกรณ์ที่มีราคาสูง จึงไม่เหมาะที่จะนำไปใช้งานกับระบบขนาดเล็ก

2.5 การสื่อสารข้อมูลแบบอนุกรม

เครื่องควบคุมแบบ PLC /PC นอกจากจะใช้ในการควบคุมเครื่องจักรแล้ว ยังได้มีการพัฒนาขีดความสามารถในการสื่อสารข้อมูล การควบคุมระยะไกล การจัดการข้อมูล เพื่อเพิ่มประสิทธิภาพในการควบคุมให้กว้างมากยิ่งขึ้น โดยมีพื้นฐานการพัฒนามาจากการสื่อสารข้อมูลของคอมพิวเตอร์ นั่นเอง

2.5.1 การติดต่อสื่อสารข้อมูลทั่วไป

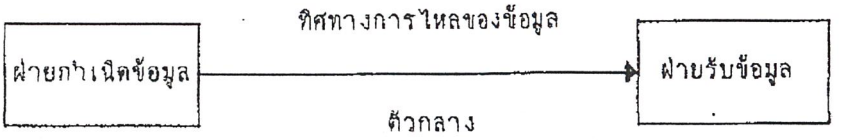
ส่วนประกอบเบื้องต้นในการสื่อสารข้อมูลแบ่งได้ออกเป็น 3 ส่วน คือ

1. ฝ่ายกำเนิดข้อมูล (Transmitter)

2. ตัวกลางในการส่งผ่านข้อมูล (Medium)

3. ฝ่ายรับข้อมูล (Receiver)

ซึ่งแต่ละส่วนมีความสัมพันธ์กันดังรูปที่ 3-1



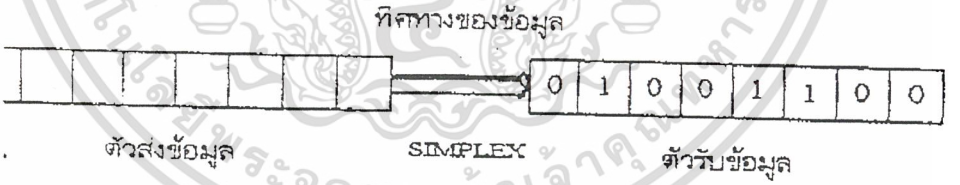
รูปที่ 2.14 แสดงความสัมพันธ์ของส่วนประกอบหลักในการสื่อสารข้อมูล

การจำแนกรูปแบบในการสื่อสารข้อมูลอาจจัดแบ่งได้หลายแบบ เช่น แบ่งตามชนิดของสัญญาณของข้อมูล คือ การสื่อสารข้อมูล อนาล็อก (Analog) และ การสื่อสารข้อมูลดิจิตอล (Digital)หรือแบ่งตามวิธีการส่งข้อมูล ในบทนี้จะกล่าวถึงการสื่อสารข้อมูลดิจิตอล ซึ่งเป็นข้อมูลชนิดที่เกี่ยวข้องกับเครื่องคอมพิวเตอร์และ เครื่องควบคุม PLC/PC

2.5.2 วิธีการส่งข้อมูล

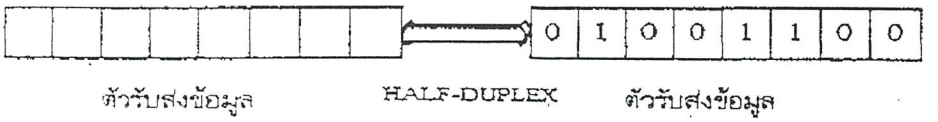
แบ่งได้ออกเป็น 3 แบบคือ

1. การส่งแบบทิศทางเดียว (Simplex) การส่งแบบนี้ทิศทางการส่งจะคงที่โดยกำหนดในครั้งแรกว่าฝ่ายใดเป็นฝ่ายส่ง ฝ่ายใดเป็นฝ่ายรับ การส่งแบบนี้แม้ไม่คล่องตัวแต่เหมาะสมในการใช้งานบางประเภท



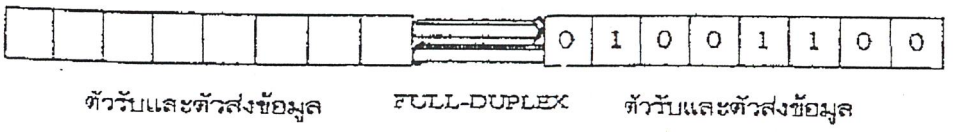
รูปที่ 2.15 การส่งแบบทิศทางเดียว (Simplex)

2. การส่งแบบฮาล์ฟดูเพล็กซ์ (Half Duplex) เป็นการส่งข้อมูลในทิศทางใดทิศทางหนึ่งในเวลาใดเวลาหนึ่ง คือ ทั้ง 2 สถานีสามารถผลัดกันส่งได้แต่จะส่งพร้อมๆกันไม่ได้



รูปที่ 2.16 การส่งแบบฮาล์ฟดูเพล็กซ์ (Half Duplex)

3. การส่งแบบสองทิศทาง(Full Duplex) เป็นการส่งข้อมูลในทิศทางใด ก็ได้ เวลาใดก็ได้ คือ สามารถส่งและรับข้อมูลในเวลาเดียวกันได้

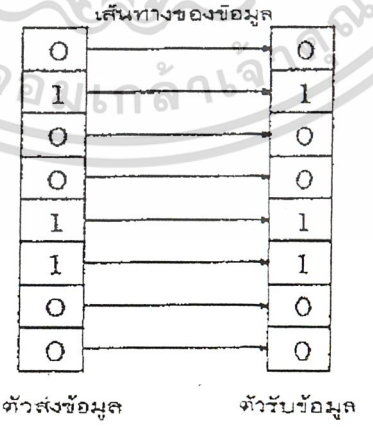


รูปที่ 2.17 การส่งแบบสองทิศทาง (Full Duplex)

วิธีการสื่อสารข้อมูลในการติดต่อสื่อสารข้อมูลของเครื่องคอมพิวเตอร์นั้นแบ่งเป็น 2 ชนิด

1. การสื่อสารข้อมูลแบบขนาน (Parallel)

เป็นการติดต่อสื่อสารโดยส่งข้อมูลออกเป็นครั้งละ 1 ไบต์ คือ ครั้งละ 8 บิท จากอุปกรณ์ส่งไปยังอุปกรณ์รับ ดังนั้นตัวกลางระหว่างเครื่องคอมพิวเตอร์ทั้งสองเครื่องจะต้องมีช่องทางให้ข้อมูลเดินทางอย่างน้อย 8 ช่องทาง โดยมากจะใช้เป็นสายขนานแต่เนื่องจากมีสัญญาณสูญหายไปกับอิมพิแดนซ์ของสาย ซึ่งมีความสัมพันธ์กับระยะทางของสาย ดังนั้นระยะทางระหว่างเครื่องสองเครื่อง ไม่ควรจะเกิน 100 ฟุต ทำให้การส่งแบบนี้ได้ระยะทางไม่ไกลนักแต่สามารถส่งข้อมูลได้รวดเร็วเพราะส่งออกมาพร้อมกันทีละ 8 บิท อุปกรณ์ที่ติดต่อแบบขนานกับคอมพิวเตอร์ เช่น เครื่อง พิมพ์ เป็นต้น



รูปที่ 2.18 การสื่อสารข้อมูลแบบขนาน

2. การติดต่อข้อมูลแบบกรม (Serial)

ในการติดต่อแบบนี้ ข้อมูลจะถูกส่งออกมา ทีละบิท ระหว่าง จุดส่งและจุดรับ ซึ่งการส่งข้อมูลแบบนี้จะช้ากว่าการส่งแบบขนาน แต่ใช้ตัวกลางในการสื่อสารเพียงช่องเดียวหรือสายเพียงคู่เดียว ทำให้ค่าใช้จ่ายสำหรับสื่อกลางถูกกว่าแบบขนาน การส่งแบบนี้ใช้สำหรับส่งในระยะทางไกลมากกว่า 100 ฟุต ข้อมูลจากจุดส่งจะต้องถูกเปลี่ยนให้เป็นอนุกรมก่อนแล้วค่อยทยอยส่งออก

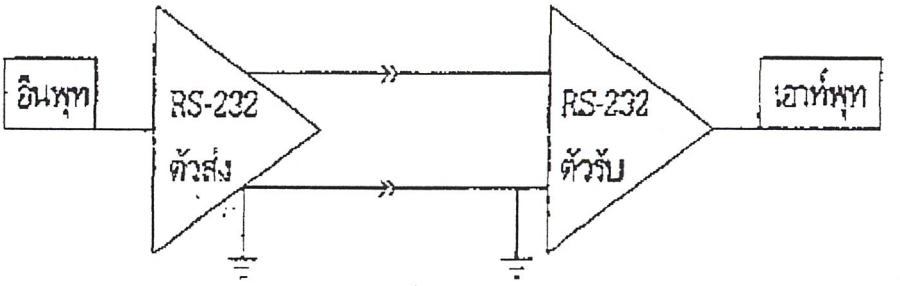
ไปทีละบิทไปยังจุดรับ ที่จุดรับจะต้องมีรูปแบบในการเปลี่ยนข้อมูลที่ถูกส่งมาให้เป็นแบบขนาน ในการแปลงต้องมีรูปแบบที่เหมาะสมเพื่อป้องกันการผิดพลาดของข้อมูล

รูปที่ 2.19 การสื่อสารข้อมูลแบบอนุกรม

2.5.3 มาตรฐานในการสื่อสารข้อมูลแบบอนุกรม

2.5.3.1 การสื่อสารข้อมูลแบบอนุกรมตามมาตรฐาน RS-232C

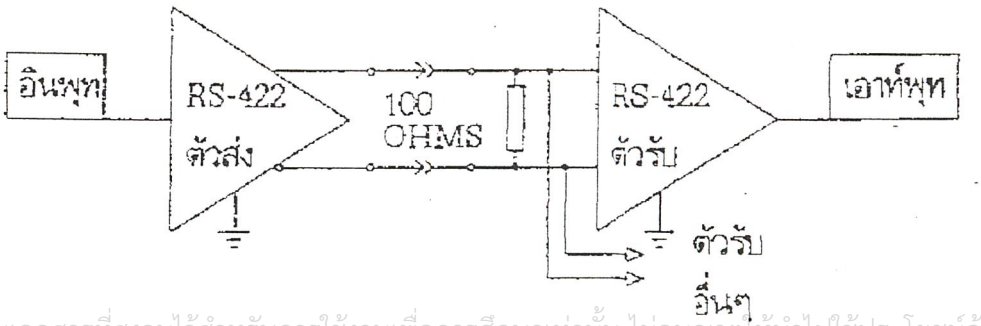
การสื่อสารข้อมูลแบบอนุกรมที่ใช้งานอยู่ในปัจจุบันนั้น ได้มีการกำหนดมาตรฐานการรับส่งข้อมูลไว้หลายแบบด้วยกัน แต่ที่ได้รับความนิยมนำมาใช้งานอย่างมาก คือ การสื่อสารข้อมูลแบบอนุกรมตามมาตรฐาน RS-232C และที่มาตรฐานนี้เป็นที่นิยมเนื่องจากเป็น ระบบการสื่อสารข้อมูลที่ใช้ในเครื่องไมโครคอมพิวเตอร์ IBM PC ซึ่งเป็นคอมพิวเตอร์ที่มีใช้อย่างแพร่หลายมากจากอดีตจนถึงปัจจุบันมาตรฐานการสื่อสารนี้ในการออกแบบเบื้องต้นได้ออกแบบ สำหรับการเชื่อมต่อกับเครื่องโมเด็ม (MODEM) ซึ่งเป็นอุปกรณ์ที่ใช้การสื่อสารข้อมูลระหว่างคอมพิวเตอร์ผ่านทางสายโทรศัพท์ ซึ่งทำให้อัตราการรับส่งข้อมูลถูกจำกัดให้มีค่าที่ค่อนข้างต่ำ มาตรฐาน RS-232C นี้ได้ออกแบบให้มีโครงสร้างการสื่อสารเป็นแบบจุดต่อจุดเท่านั้น โดยมีลักษณะสมบัติทางไฟฟ้าและทางกายภาพ ดังแสดงในตารางที่ 5 และรูปที่ 3-9



รูปที่ 2.20 แสดงโครงสร้างของการสื่อสารข้อมูลแบบอนุกรมตามมาตรฐาน RS-232C

2.5.3.2 การสื่อสารข้อมูลแบบอนุกรมมาตรฐาน RS-422A

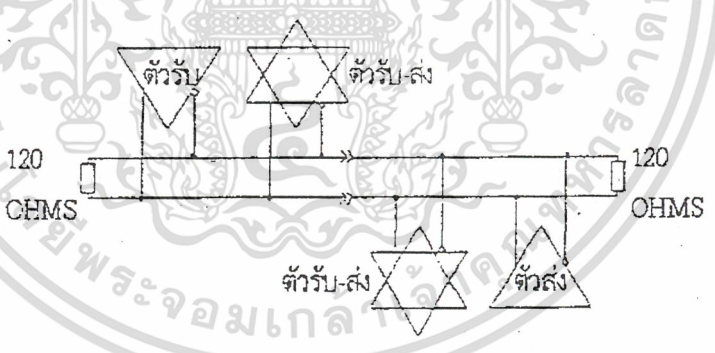
ในการออกแบบระบบสื่อสารข้อมูลจากที่ผ่านมาจนถึงปัจจุบัน ได้มีการพยายามที่จะออกแบบให้การสื่อสารข้อมูลได้รวดเร็วขึ้นและมีระยะในการสื่อสารที่มากขึ้นด้วย ซึ่งที่ผ่านมาการสื่อสารข้อมูลตามมาตรฐาน RS-232C ได้ออกแบบเพื่อใช้เชื่อมต่อกับโมเด็มเท่านั้น จึงไม่ได้คำนึงถึงความเร็วและระยะทางในการสื่อสาร ต่อมาได้มีมาตรฐานในการสื่อสารข้อมูลใหม่ที่ได้ออกแบบมาเพื่อรองรับความต้องการของผู้ใช้งานที่ต้องการให้การรับส่งข้อมูลได้ระยะทางไกลและรวดเร็ว มาตรฐานนี้คือ RS-422A ซึ่งการที่มาตรฐานสื่อสารนี้สามารถรับส่งข้อมูลได้ไกลและรวดเร็วขึ้นเนื่อง มาจากหลักการที่ใช้สัญญาณเป็นแบบดิฟเฟอเรนเชียลดังแสดงในรูปที่ 17 ซึ่งหลักการก็คือ สัญญาณที่รับส่งจะเป็นการเปรียบเทียบระหว่างสัญญาณ 2 เส้นเทียบกับ มาตรฐาน RS-232C ที่สัญญาณทุกสัญญาณจะเทียบกับกราวด์ ซึ่งในการสื่อสารในระยะทางไหลๆ แล้วสัญญาณจะถูกลดทอนไปและเมื่อสัญญาณถูกลดทอนถึงจุดๆ หนึ่งสัญญาณนั้นก็จะผิดพลาดไปจากความเป็นจริง ทำให้การรับส่งข้อมูลเกิดผิดพลาดขึ้น แต่สำหรับสัญญาณแบบดิฟเฟอเรนเชียลการลดทอนของสัญญาณก็จะไปลดทอนทั้งสองสายด้วยค่าที่เท่ากันหรือ ใกล้เคียงกันและความแตกต่างของสัญญาณระยะสัญญาณทั้ง 2 เส้น จากตัวส่งไปยังตัวรับก็ยังมีค่าเท่าเดิมหรือเปลี่ยนแปลงน้อย จึงทำให้ผลของการลดทอนต่อสัญญาณที่ระยะการสื่อสารที่ไกลมีผลต่อการ สื่อสารข้อมูลสูงกว่าพร้อมทั้งสามารถติดต่อตัวรับได้ถึง 10 ตัว ดังแสดงตารางเปรียบเทียบมาตรฐานการสื่อสารข้อมูลในตารางที่ 3-1



รูปที่ 2.21 แสดงโครงสร้างของการสื่อสารข้อมูลแบบอนุกรมตามมาตรฐาน RS-422A

2.5.3.3 การสื่อสารข้อมูลแบบอนุกรมตามมาตรฐาน RS-485

การสื่อสารข้อมูลตามมาตรฐานที่กล่าวมาข้างต้นคือ RS-232C นั้นเป็นมาตรฐานการสื่อสารข้อมูลในแบบที่ใช้สื่อสารระหว่างอุปกรณ์ต่ออุปกรณ์ หรือ จุดต่อจุด (Point-to-Point) ส่วน RS-422A นั้นเป็นมาตรฐานที่พัฒนามาจาก RS-232C ให้ได้ระยะทางไกลขึ้นและอัตราการสื่อสารเพื่อขึ้น แต่ก็ยังเป็นการสื่อสารข้อมูลจากอุปกรณ์หนึ่งตัวไปยังอุปกรณ์อื่นๆ ได้สูงสุด 10 ตัวเท่านั้น ไม่สามารถส่งย้อนกลับจากอุปกรณ์ตัวรับมาตัวส่งได้ หรือกล่าวได้ว่าการสื่อสารข้อมูลตามมาตรฐาน RS-422A นั้นเป็นการสื่อสารข้อมูลแบบ Simplex คือทิศทางของข้อมูลเป็นแบบทางเดียวตลอดเวลา ดังนั้นถ้าต้องการออกแบบระบบให้เป็นลักษณะโครงข่ายข้อมูลก็จะไม่สามารถทำได้ จึงมีการพัฒนามาตรฐานการสื่อสารข้องมูลขึ้นใหม่เพื่อรองรับความต้องการนี้ คือ มาตรฐาน RS-485 ซึ่งเป็นมาตรฐานที่อาศัยหลักการของสัญญาณแบบดิฟเฟอเรนเชียลเช่นเดียวกับมาตรฐาน RS-422A แต่สามารถสื่อสารข้อมูลได้ทั้ง 2 ทิศทางในสายสัญญาณเพียงคู่เดียว ซึ่งก็คือการสื่อสารข้อมูลแบบ Half-Duplex จากผลของการใช้สัญญาในลักษณะดิฟเฟอเรนเชียลนี้ ทำให้ระยะทางและความเร็วในการสื่อสารข้อมูลมีค่าสูง เช่นเดียวกับมาตรฐานการสื่อสารข้อมูล RS-422A แต่มาตรฐาน RS-485 สามารถที่จะสื่อสารระหว่างอุปกรณ์ทั้งการส่งของอุปกรณ์ได้สูงสุด 32 ตัว หรืออาจกล่าวได้ว่าการสื่อสารตามมาตรฐาน RS-485 เป็นการสื่อสารแบบหลายจุด (Multi point Communication) ดังแสดงค่าเปรียบเทียบในตารางที่ 3-1 และแสดงโครงสร้างในรูปที่ 3-11



รูปที่ 2.22 แสดงโครงสร้างของการสื่อสารข้อมูลแบบอนุกรมตามมาตรฐาน RS-485

ตารางที่ 2.5 แสดงค่าเปรียบคุณสมบัติทางไฟฟ้าของ RS-232 C , RS-422A , และ RS -485

พารามิเตอร์	RS-232-C	RS-433-A	RS-485
โหมดการทำงานจำนวนของตัวรับและส่งที่ยอมรับได้	Single-ended 1 ตัวส่ง 1 ตัวรับ	Differential 1 ตัวส่ง 10 ตัวรับ	Differential 32 ตัวส่ง 32 ตัวรับ
ความยาวของคู่สายสูงสุด	50 ฟุต	4000 ฟุต	4000 ฟุต
อัตราการส่งข้อมูลสูงสุด	20k bps	10 M bps	10 M bps

Maximum common mode voltage	2.5 V	6V - 2.5 V	12 V -7 V
Driver output	5V ต่ำสุด 15V สูงสุด	2V ต่ำสุด	1.5 V ต่ำสุด
Driver load (ohm)	3K ถึง 7K	100 ต่ำสุด	60 ต่ำสุด
Driver slew rate	30V/us สูงสุด	NA	NA
กระแสลิมิต เมื่อเอาท์พุทลัดวงจร	500mA ลัดวงจรกับ VCC หรือ GND	150 mA ลัดวงจรกับ GND	150 mA ลัดกับ GND 250 mA ลัดกับ 12V
ค่าความต้านทาน เอาท์พุทของตัวส่ง (โอหม์)	NA (power on) 300 (power off)	NA (power on) 60k (power off)	120k (power on) 120k (power off)
ค่าความต้านทานอินพุทตัวรับ	3k ถึง 7k ohm	4k ohm	12k ohm
ความไวของตัวรับ	3V	200 mV	200 mV

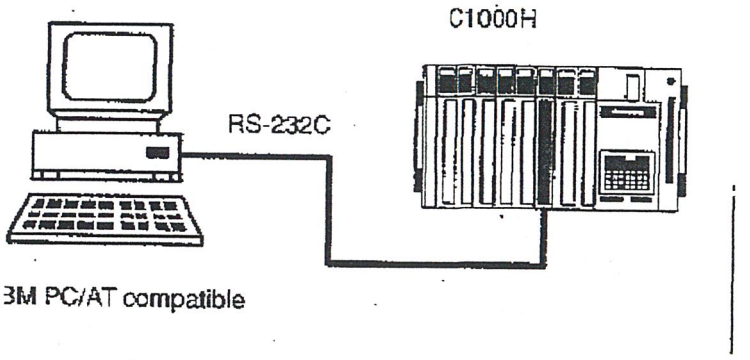
2.5.4 รูปแบบการสื่อสารข้อมูลของเครื่องควบคุมแบบตรรกที่โปรแกรมได้

การต่อวงจรในการติดต่อสื่อสารของเครื่องควบคุม PC/PLC อาจแบ่งออกได้ 3 แบบด้วยกันคือ

1. การต่อแบบจุดต่อจุด (Single link System)
2. การต่อแบบหลายจุด (Multi link System)
3. การต่อแบบระบบแลน (Local area Network

2.5.4.1 ระบบการติดต่อแบบจุดต่อจุด (Single link System)

ในการติดต่อสื่อสารข้อมูลแบบจุดต่อจุด จะใช้มาตรฐานการสื่อสารแบบ RS 232C ซึ่งเป็นการติดต่อในระยะทางที่ไม่ไกลมาก เช่น การติดต่อระหว่างเครื่องควบคุม PC/PLC กับเครื่องคอมพิวเตอร์ เพื่อทำการตรวจสอบสภาวะการทำงานหรือการควบคุมจากคอมพิวเตอร์ หรือสนับสนุนการทำงานอื่นๆ การติดต่อระหว่างเครื่องควบคุม PLC/PC กับอุปกรณ์อื่น เช่น โมเด็ม (MODEM) อุปกรณ์ควบคุมสัมผัสหน้าจอ (TOUCH SCREEEN) คีย์บอร์ด(Key board) เครื่องอ่านรหัสแถบ (Bar-Code Reader) เครื่องพิมพ์ เครื่องบันทึก (Recorder) เครื่องรายงานผล (Reporter) เป็นต้น

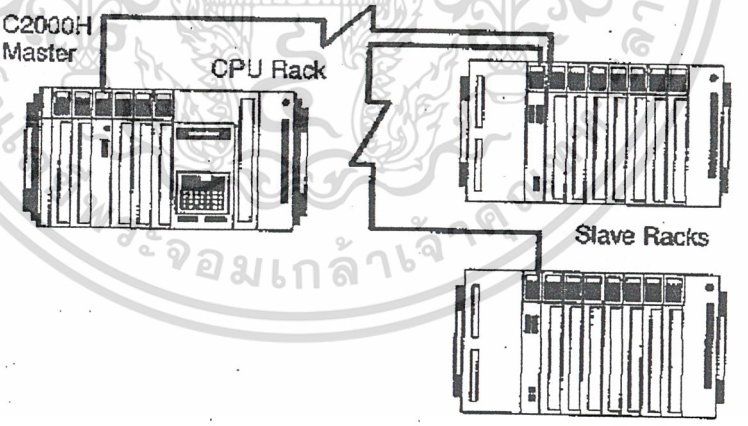


รูปที่ 2.23 ระบบการติดต่อแบบจุดต่อจุด (Single link System)

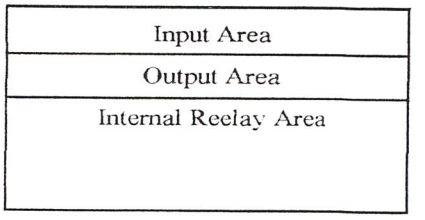
2.5.4.2 การต่อวงจรแบบหลายจุด (Multi point)

2.5.4.2.1 การเชื่อมโยงแบบอินพุทและเอาท์พุทระยะไกล (Remote I/O System)

เป็นการเชื่อมโยงสื่อสารข้อมูลในลักษณะที่ตัวประมวลผลของเครื่องควบคุมหลักกับปลายทางอุปกรณ์ที่ถูกควบคุมอยู่ไกลกัน และอาจมีมากกว่า 1 จุด ดังนั้นจึงมีความจำเป็นที่จะมีการเชื่อมโยงในระยะไกล ในการติดต่อลักษณะนี้จะใช้มาตรฐานการสื่อสารแบบ RS 485 บนตัวนำสองเส้น โดยที่ตัวควบคุมหลัก(Master) จะทำการรับอินพุททั้งจากพื้นที่ของตัวควบคุม และหน่วยระยะไกล(Slave) จากนั้นจะทำการประมวลผล แล้วส่งเอาท์พุทไปยังส่วนต่างๆ โดยที่อินพุทและเอาท์พุทของหน่วยระยะไกลจะจองพื้นที่ของรีเลย์ภายใน (Internal Relay) ของตัวควบคุมหลักโดยการเซ็ตระบบนั้นจะแตกต่างกันไปตามบริษัทผู้ผลิต



รูปที่ 2.24 การเชื่อมโยงแบบอินพุทและเอาท์พุทระยะไกล (Remote I/O System)



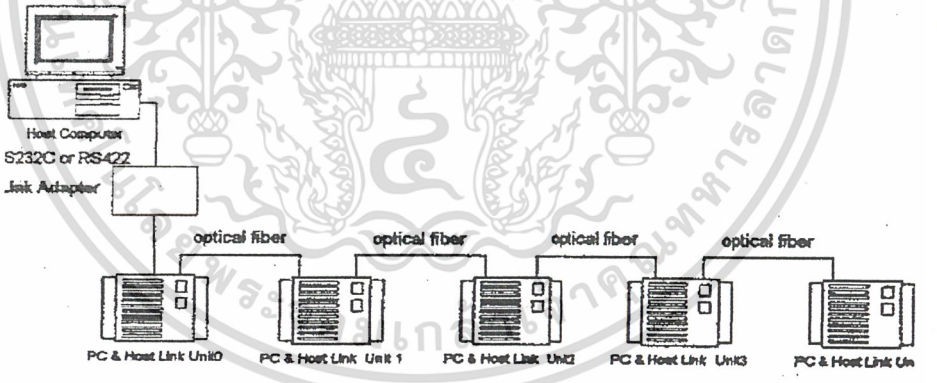
รูปที่ 2.25 แสดงการจำลองพื้นที่ของตัวควบคุมหลักขณะทำงานปกติ

Input Area
Output Area
Internal Relay Area
Input Remote Output Remote

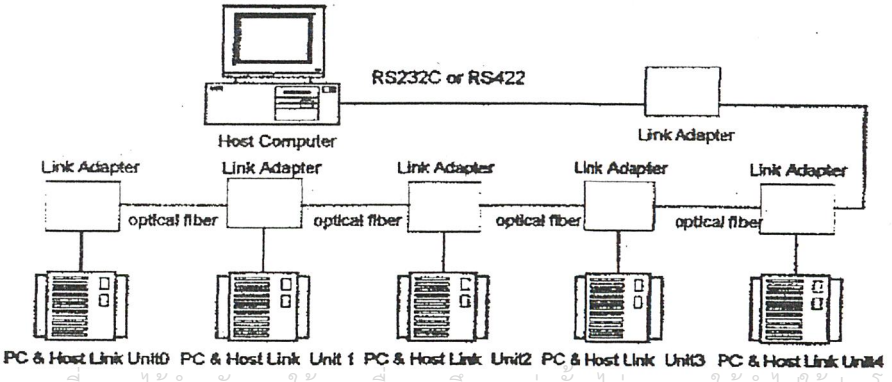
รูปที่ 2.26 แสดงการจำลองพื้นที่ของตัวควบคุมหลักขณะเมื่อมีการทำงานร่วมกับหน่วยอินพุทเอาท์พุทระยะไกล

2.5.4.2.2 การติดต่อแบบพีซีลิ้งค์ (PC Link System)

เป็นการเชื่อมโยงสื่อสารข้อมูลในลักษณะของการแบ่งพื้นที่ในการอ่านเขียนของ PLC/PCแต่ละตัว เครื่องควบคุม PLC/PC แต่ละตัวสามารถรับรู้ข่าวสารซึ่งกันและกันได้มาตรฐานในการสื่อสารข้อมูลชนิดนี้มักจะเป็นแบบ RS422A หรือเป็นรูปแบบการสื่อสารผ่าน สายใยแก้วนำแสง (Fiber optic) หรือผ่านสายโคแอคเชียล โดยจะมีตัว Link Adapter เป็นตัวแปลงรูปแบบข้อมูลให้เป็น มาตรฐานตรงกันอาจจะมีการเชื่อมโยงกับเครื่องคอมพิวเตอร์ เข้ามาในระบบการลิ้งค์ เพื่อสนับสนุนการทำงานของเครื่องควบคุม PLC/PC ซึ่งจะเรียกว่า โฮสคอมพิวเตอร์ (Host Computer) มีลักษณะการต่อวงจร ดังนี้

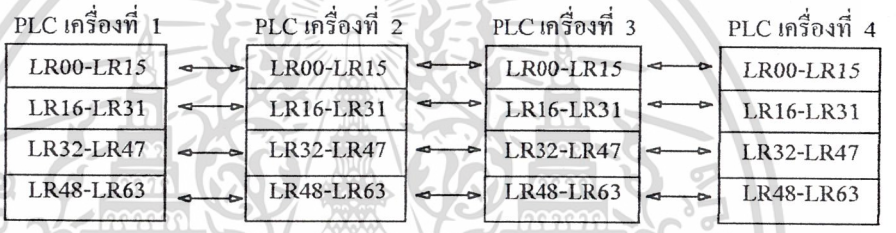


รูปที่ 2.27 การต่อแบบ Serial link



รูปที่ 2.28 การต่อแบบ Parallel link

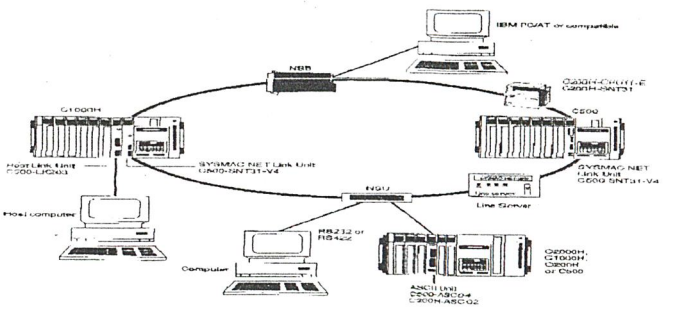
ตัวอย่างการจัดแบ่งพื้นที่ของเครื่องควบคุม PLC/PC กำหนดให้มีเครื่องควบคุมจำนวน 4 ชุด เชื่อมโยงในเครื่องควบคุมแต่ละเครื่องจะมีพื้นที่ของรีเลย์ที่ทำหน้าที่ในการเชื่อมโยงข้อมูล (Link–Relay) ถ้ามีเครื่องควบคุมที่เชื่อมต่อในระบบมากพื้นที่จะถูกแบ่งออกไปตามจำนวน เช่น 4 ชุด พื้นที่จะถูกแบ่งออกเป็น 4 ส่วน เช่น เครื่องควบคุมพื้นที่ของ ลิ้งค์รีเลย์จำนวน 64 Word ก็จะถูกแบ่งเป็นส่วนละ 16 Word ส่วนแรกจะเป็นพื้นที่ของเครื่องควบคุมตัวที่ 1 ส่วนที่สองจะเป็นของเครื่องควบคุมตัวที่ 2 และ 3 ตามลำดับ โดยพื้นที่ส่วนอื่นจะถูกกันไว้ไม่ให้สามารถเขียนข้อมูลลงไปได้ (Read Only) ยกเว้นพื้นที่ของตัวเอง ดังนั้นถ้า PLC/PC เครื่องที่ 1 ต้องการรับข้อมูลของ PLC เครื่องที่ 3 ก็สามารถอ่านได้จากพื้นที่ของเครื่องควบคุมส่วนที่สาม และถ้าต้องการส่งข้อมูลให้ก็เขียนลงในพื้นที่ของส่วนแรก แล้ว PLC ตัวที่ 3 จะทำการอ่านข้อมูลในส่วนแรก ก็จะสามารถทราบข้อมูลของเครื่องควบคุม PLC ตัวที่ 1



รูปที่ 2.29 การแบ่งพื้นที่สำหรับ PC Link System

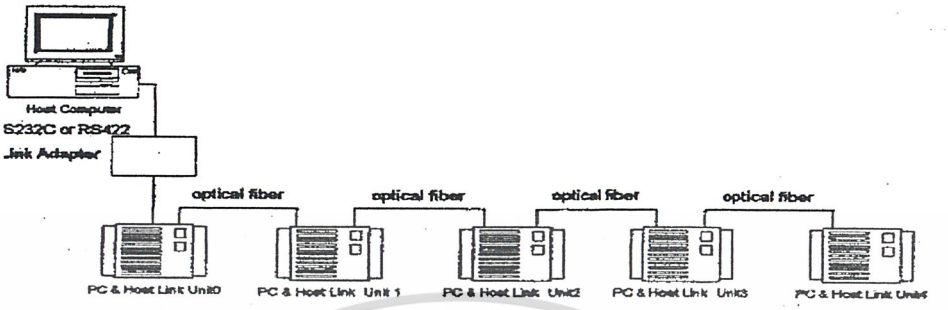
2.5.4.3 การต่อวงจรสื่อสารแบบเครือข่ายท้องถิ่น (Local area Network for PLC/PC)

การเชื่อมโยงในลักษณะของเครือข่ายท้องถิ่นนี้เป็นระบบขนาดใหญ่ประกอบด้วยอุปกรณ์ หลายชนิด และ หลายวงรอบด้วยกัน ซึ่งจะเป็นลักษณะเครือข่ายแบบวงแหวน (Ring Network) ในแต่ละวงรอบ สามารถที่จะเชื่อมโยงไปยังวงรอบอื่นๆได้ ซึ่งจะมีความซับซ้อนในระบบการติดต่อสื่อสารมากขึ้น รูปแบบสายนำสัญญาณจะมีทั้งสายใยแก้วนำแสง (Fiber optic) และสายตัวนำในแบบต่างๆ ในระบบโครงข่ายก็จะมีเครื่องคอมพิวเตอร์ส่วนบุคคลทำหน้าที่เป็นตัวจัดการข้อมูลเช่นเดียวกันกับเครื่องเซิฟเวอร์(Computer Server) บนโครงข่ายของคอมพิวเตอร์



รูปที่ 2.30 ระบบการติดต่อสื่อสารแบบเครือข่ายท้องถิ่น

สำหรับระบบโครงข่ายที่ได้ออกแบบในวิทยานิพนธ์นี้ ป็นการเชื่อมโยงแบบหลายจุดที่ใช้มาตรฐานการสื่อสารแบบ RS 485 โดยมีโฮสคอมพิวเตอร์เป็นตัวจัดการข้อมูลต่างๆให้กับเครื่องควบคุมที่เชื่อมต่อในระบบโครงข่ายดังรูปต่อไปนี้



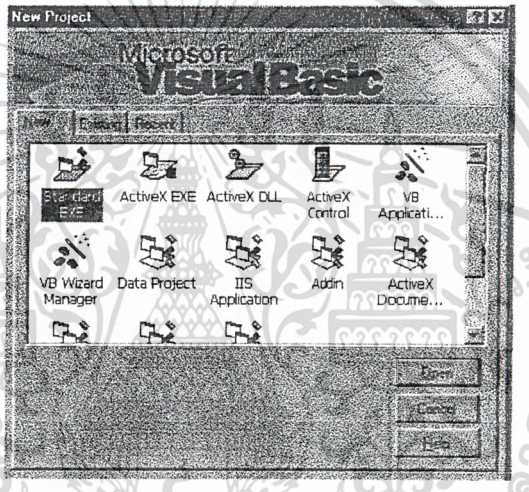
รูปที่ 2.31 แสดงระบบโครงข่ายแบบหลายจุดโดยใช้มาตรฐานการสื่อสาร RS 485

บทที่ 3
การเขียนโปรแกรมด้วย Visual Basic

3.1 การเขียนโปรแกรมด้วย Visual Basic ขั้นพื้นฐาน

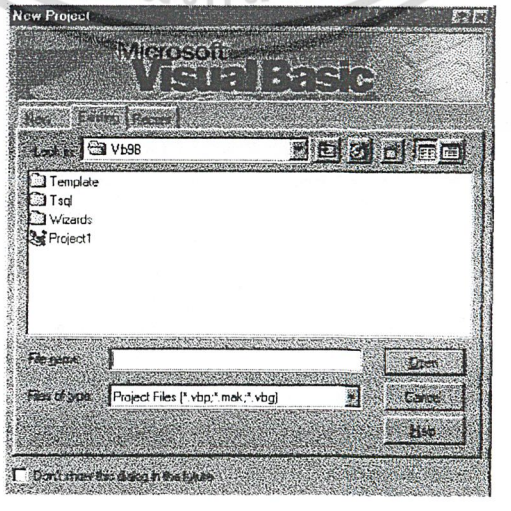
สิ่งแรกที่จะพบเมื่อเข้าสู่โปรแกรม Microsoft Visual Basic 6.0 ได้แก่ จอภาพ ที่ใช้สำหรับเปิด Project จะเป็นชื่อที่ใช้เรียกแทนระบบงานที่พัฒนาขึ้นด้วยการเขียนโปรแกรม Visual Basic ประกอบไปด้วย 3 Tab ดังนี้

1. Tab "New" เป็นจอภาพที่ประกอบไปด้วย Icon ต่างๆที่ใช้สำหรับเรียกใช้โปรเจคใหม่ขึ้นมาใช้งาน



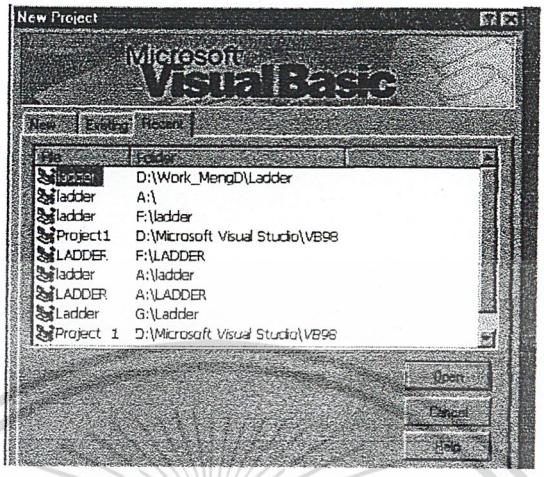
รูปที่ 3.1 แสดงส่วนของ Tab"New"

2. Tab "Existing" เป็นจอภาพสำหรับเรียกใช้ Project เดิมที่พัฒนาขึ้นแล้วมาเก็บไว้ใน Directory ต่างๆขึ้นมาใช้งาน



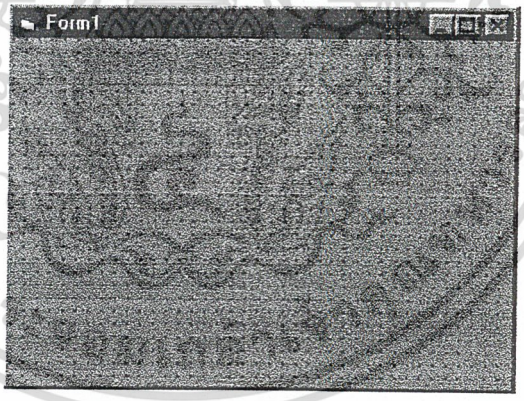
รูปที่ 3.2 แสดงส่วนของ Tab "Existing"

3. Tab "Recent" เป็นจอภาพที่แสดงประวัติของ Project ต่างๆที่เคยถูกเรียกขึ้นมาพัฒนา ส่วนประกอบของจอภาพ Visual Basic 6.0 มีดังนี้



รูปที่ 3.3 แสดง ส่วนของ Tab "Recent"

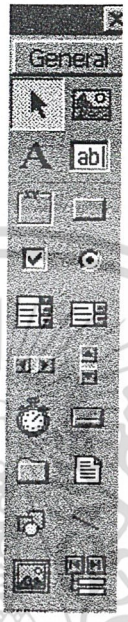
-Form เป็นส่วนที่ใช้สำหรับสร้างจอภาพของโปรแกรม โดยจะทำหน้าที่เป็นพื้น (Background) ของจอภาพ ทุกครั้งของการเปิด Project ใหม่ จะได้ฟอร์มเปล่า



รูปที่ 3.4 แสดง Form

-Toolbox เป็นแถบเครื่องมือที่ประกอบไปด้วย Icon ต่างๆ ซึ่งเรียกว่า "Control" ซึ่งจะใช้ร่วมกับ Form เพื่อสร้างจอภาพ Project แต่ละ Control จะใช้เป็นเครื่องมือที่ใช้สำหรับสร้างส่วนที่ใช้ติดต่อกับผู้ใช้ หรือที่เรียกว่า "User Interface" เช่นข้อความต่างๆ ช่องว่างสำหรับรับข้อมูลจากคีย์บอร์ดปุ่มต่างๆ ฯลฯ เป็นต้น และจะถูกนำไปใช้งานด้วย โดยการนำ Control ที่จะต้องการไปวางลงบน Form

Control แต่ละจะมีชื่อและหน้าที่ที่แตกต่างกันไป เมื่อต้องการดูชื่อของ Control ใดก็เพียงแต่เลื่อนเม้าท์ไปชี้ที่ Control นั้นชื่อของ Control จะปรากฏขึ้นให้เห็น



รูปที่ 3.5 แสดง Control บน Toolbox

-Toolbar เป็นแถบเครื่องมือที่ประกอบด้วย Icon ต่างๆดังรูป ซึ่ง Toolbar ทำหน้าที่เป็นผู้ช่วยในการพัฒนาโปรแกรม ซึ่งเมื่อเลื่อนเม้าท์ไปชี้ยัง Icon ใด ก็จะปรากฏชื่ออยู่ใต้ Icon นั้นแต่ละ Icon จะมีหน้าที่ต่างกันไป

รูปที่ 3.6 แสดง Toolbar

-Project Explorer Window เป็นส่วนสำหรับเรียก Form ต่างๆขึ้นมาแก้ไขในกรณีที่ Project ประกอบด้วย Form มากกว่า 1 Form

-Properties Window เป็นจอภาพที่ใช้สำหรับกำหนดคุณสมบัติ (Property) ให้กับ Form และ Object ต่างๆ ที่ปรากฏอยู่บน Form

- Form Layout Window ใช้สำหรับดูตำแหน่งของ Form บนจอภาพ ทำให้จัดตำแหน่งของ Form ได้สะดวกขึ้น

-Project โดยทั่วไปในงานหนึ่งๆมักประกอบไปด้วยหลายๆ จอภาพ ดังนั้นการพัฒนาโปรแกรม จึงนิยมแยกแต่ละจอภาพออกเป็นโปรแกรมเพื่อสะดวกต่อการแก้ไขตามหลักของการเขียนโปรแกรม แล้วจึงนำแต่ละโปรแกรมย่อยมาประกอบกันขึ้นเป็นระบบโดยการ Compile ไฟล์เหล่านั้นร่วมกันเป็น Executed Program (ไฟล์นามสกุล EXE) เพื่อนำไปใช้งาน

3.2 การ Run และเลิกงาน Project

ในการ Run Project ที่พัฒนาขึ้นด้วย Visual Basic สามารถ Run ทั้งโดยใช้ Interpreter และการใช้ Compiler กล่าวคือเราสามารถทดลอง Run สิ่งต่างๆ ที่เราจัดทำขึ้นไปพร้อม ๆ กับการแก้ไขโปรแกรมจนกระทั่งเสร็จสมบูรณ์ แล้วจึง Compileโปรแกรมให้อยู่ในรูปของ Executed Program เพื่อนำไปใช้ในงานได้เช่นเดียวกัน แต่ในเบื้องต้นนี้ จะขอกล่าวถึงส่วนที่เป็น Interpreter เป็นลำดับแรก

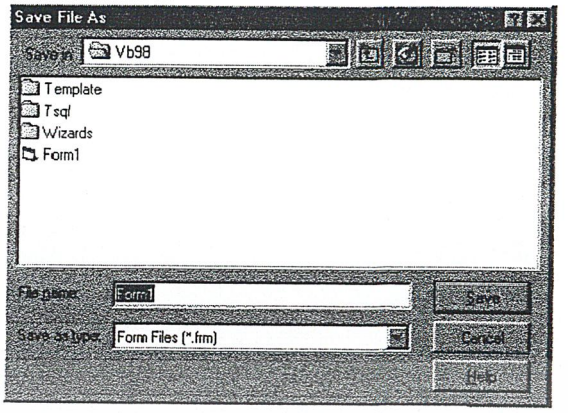
ตารางที่ 3.1 การ Run และเลิกงาน Project

ในการ Run Project ที่เราจัดทำขึ้นนั้นทำได้ 3 วิธี	สำหรับวิธีเลิกงาน Project ทำได้ 2 วิธี
วิธีที่ 1 Run โดยกด *F5* วิธีที่ 2 คลิกที่ Icon "Run"ใน Toolbar วิธีที่ 3 เลือกจากเมนู Run และ Start ตามลำดับ	วิธีที่ 1 คลิกที่ *Icon*"End"ในToolbar วิธีที่ 2 เลือกจากเมนู Run และ End ตามลำดับ

3.3 การบันทึก Form และ Project

ในการบันทึก Project จะต้องบันทึกทั้งส่วนของ Form และ Project โดย Form จะถูกบันทึกลงในไฟล์นามสกุล FRM ในขณะที่ Project จะถูกบันทึกลงในไฟล์นามสกุล VBPในการบันทึก Form ให้ทำดังนี้

1. คลิกที่ Icon "Save" ใน Toolbar จะปรากฏจอภาพดังรูป
2. ใส่ชื่อ File ที่จะใช้เก็บ Form ซึ่ง Visual Basic จะใช้ชื่อเริ่มต้นเป็นชื่อเดียวกับ Form เสมอดังนั้นจึงปรากฏชื่อ Form 1 มาให้ผู้อ่านสามารถเปลี่ยนแปลงชื่อได้ตามต้องการ จากนั้นให้คลิกที่ปุ่ม Save
3. จะปรากฏจอภาพสำหรับบันทึก Project ซึ่งก็เช่นเดียวกับ Form เป็นชื่อ Project 1 ให้เปลี่ยนชื่อตามต้องการแล้วคลิกที่ปุ่ม Save



รูปที่ 3.7 แสดงการบันทึก Form และ Project

3.4 ขั้นตอนในการพัฒนาโปรแกรมของ Visual Basic ประกอบไปด้วยขั้นตอนหลัก 2 ขั้นตอน

ขั้นตอนที่ 1 การสร้างจอภาพของโปรแกรม

ในขั้นตอนนี้จะนำ Form มาออกแบบเพื่อใช้ในการติดต่อกับผู้ใช้ หรือที่เรียกว่า การออกแบบ " User Interface" ในการพัฒนาโปรแกรมแบบเดิม แต่ในส่วนนี้ซึ่ง Visual Basic สามารถทำได้อย่างง่ายดายจากการพัฒนาโปรแกรมแล้ว เพียงแต่นำเอา Control ต่างๆ ในToolbox ที่ต้องการใช้งานมาวาดไว้บน Form

ขั้นตอนที่ 2 การเขียนโปรแกรม

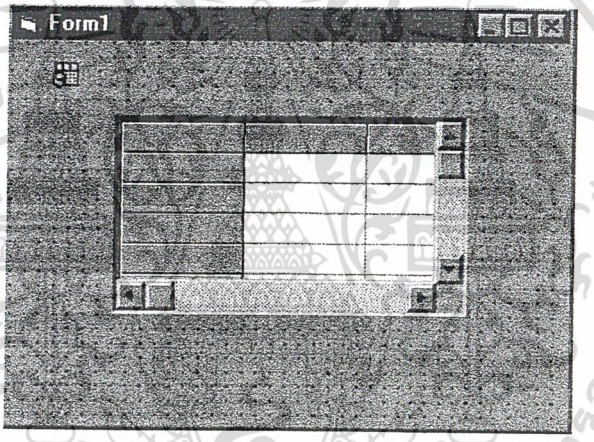
เมื่อวาง Control ต่างๆลงบน Form เป็นที่เรียบร้อยแล้ว (Control ต่างๆเมื่อถูกนำมาวางไว้บน Form จะเรียกว่า "Object") ขั้นตอนต่อมาได้แก่ การเขียนโปรแกรมเพื่อกำหนดการทำงานให้แต่ละ Object ภายในเหตุการณ์ต่างๆ (Even) ที่จะเกิดขึ้นกับจอภาพนั้นๆ

-Event-Driven Program กับ Visual Basic

แนวความคิดในกรเขียนโปรแกรมแบบ Event-Driven จะเปลี่ยนมาสนใจกับเหตุการณ์ (Event) ที่จะเกิดขึ้นในโปรแกรมมากกว่าการกำหนดขั้นตอนการทำงานของโปรแกรมในแบบเดิม เช่น ถ้ามีการ "เลื่อนเม้าท์" เกิดขึ้น จะให้โปรแกรมทำอย่างไร หรือมีการกดปุ่มที่ 1 ขึ้นจะให้โปรแกรมทำอย่างไรเป็นต้น อย่างไรก็ตาม ก็ยังต้องอาศัยแนวความคิดในการเขียนโปรแกรมแบบเดิมอยู่บ้าง เนื่องจากการกำหนดการทำงานให้กับแต่ละ Event ยังต้องกำหนดอย่างเป็นขั้นตอนอยู่ดี การเขียนโปรแกรมแบบ Event-Driven ใน Visual Basic จะเป็นการเขียนโปรแกรมให้กับ Object ต่างๆ ที่ปรากฏอยู่บน Form โดยจะพิจารณาว่าแต่ละ Object จะมี Event อะไรเกิดขึ้นบ้าง แล้วจึงเลือกเขียนโปรแกรมเฉพาะ Event นั้น

3.5 การใช้เอ็มเอสเฟลกกริด (MsFlexgride Control)

เอ็มเอสเฟลกกริด เป็นวัตถุควบคุมที่แสดงผลได้ค่อนข้างจะมีประสิทธิภาพ เมื่อเข้าสู่โปรแกรมแล้ว ถ้ามองไม่เห็นวัตถุควบคุมเอ็มเอสเฟลกกริดในกล่องเครื่องมือ ให้เพิ่มวัตถุควบคุมนี้เข้าไปใช้โดยใช้เมนู Project เอ็มเอสเฟลกกริด เป็นวัตถุควบคุมที่สามารถแสดงผลข้อมูลออกมาเป็นแถวและคอลัมน์ เช่นเดียวกับตารางทำงานของเอกซ์เซล ข้อมูลอาจเป็นข้อความ ตัวเลขหรือรูปภาพก็ได้ และยังมีความสามารถในการเชื่อมเซลล์เข้าด้วยกัน เรียงลำดับข้อมูลตามแนวคอลัมน์ได้อีกด้วย เมื่อนำเอ็มเอสเฟลกกริดมาวาดลงในแบบฟอร์ม จะมีแถวและคอลัมน์ดังเห็นในรูป การกำหนดจำนวนแถวและคอลัมน์ให้กับเอ็มเอสเฟลกกริด ทำได้โดยเปลี่ยนค่าตัวเลขของคุณสมบัติ Rows และ Cols ในหน้าต่างคุณสมบัติ (Properties) ได้ตามที่ต้องการหรือจะเขียนคำสั่งกำหนด (Code) ก็ได้



รูปที่ 3.8 ลักษณะของ MSFLEX GRID

การวาดเอ็มเอสเฟลกกริดต้องให้มีขนาดโตเท่ากับขนาดตารางที่จะแสดงออกมา จึงจะมองเห็นข้อความในตารางได้ทั้งหมดแต่ถ้าหากวาดเล็กจะไม่สามารถแสดงแถวและคอลัมน์ได้ทั้งหมดซึ่งจะมีแถบเลื่อนหน้าต่างตามแนวนอน และ/หรือ แนวตั้งเกิดขึ้นโดยอัตโนมัติ คอลัมน์ที่ 0 เรียกว่า Fixed Title Column สำหรับใส่ข้อความอธิบายแถว คอลัมน์นี้จะไม่เคลื่อนที่เมื่อเลื่อนแถบเลื่อนหน้าต่างตามแนวนอนแถวที่ 0 เรียกว่า Fixed Title Row สำหรับใส่ข้อความอธิบายคอลัมน์ แถวนี้จะไม่เคลื่อนที่เมื่อเลื่อนแถบเลื่อนหน้าต่างตามแนวตั้ง

3.6 การใช้ Control PictureClip

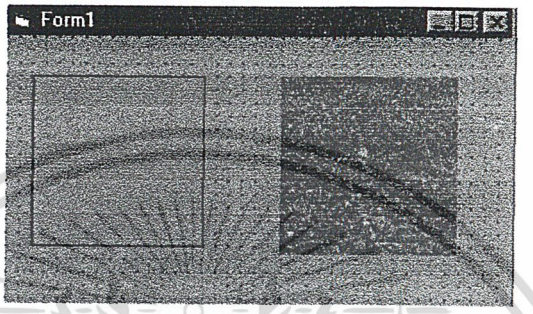
อีกส่วนหนึ่งที่ใช้ในการวางภาพในส่วนของการใช้งาน Control PictureClip โดยจะเป็นการนำภาพที่สร้างไว้หรือภาพที่มีอยู่มาใช้ Control PictureClipในการตัดภาพเป็นภาพย่อยๆ โดย

ต้องระบุจำนวน Column และจำนวน Row ก็จะได้ภาพย่อยๆเท่ากับจำนวน Column คูณกับRow โดยจะเก็บเป็น Array การเพิ่ม Control PictureClip ทำได้โดย

1. คลิกเมนู Project เลือกคำสั่ง Component... หรือคลิกขวาที่ Toolbox แล้วเลือกคำสั่ง Component...

2. เลือก Microsoft PictureClip Control 6.0

3. คลิก OK



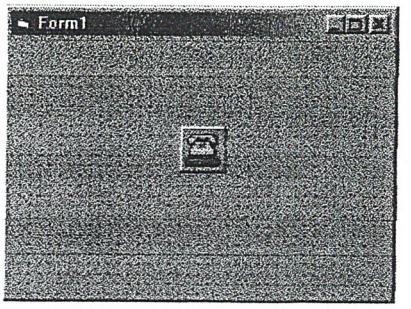
รูปที่ 3.9 ลักษณะของ Microsoft Picture Clip Control

3.7 การสื่อสารกับ Communication Port

Visual Basic ไม่สามารถจัดการกับ Hardware ได้โดยตรงเหมือนภาษา C หรือภาษาอื่นๆ อย่างไรก็ดี Visual Basic ได้เตรียม Tool และ Control ต่างๆ สำหรับจัดการกับ Hardware โดยเฉพาะไว้ให้ผู้เขียนโปรแกรมเท่าที่จำเป็น ในการทำงานกับพอร์ตสื่อสารบนเครื่อง PC ซึ่ง Visual Basic มี Control ชื่อ Microsoft Comm Control สำหรับการรับส่งข้อมูลทาง Port อนุกรม (Serial Port)

การใช้ MSComm Control

MSComm จัดเตรียมทางเลือกเอาไว้ 2 ทางเพื่อความสะดวกในการสื่อสารข้อมูล ทางแรกคือ การสื่อสารข้อมูลที่กระตุ้นด้วยเหตุการณ์ (event-driven communications) เป็นรูปแบบการใช้งานที่มีประสิทธิภาพมากสำหรับการตอบสนองแบบทันทีที่ทันใด เช่น เมื่อตัวอักษรถูกส่งมาที่พอร์ตอนุกรมหรือเกิดการเปลี่ยนแปลงที่ขา Data Carrier Detect (DCD) หรือขา Request To Send (RTS) เหตุการณ์ ONCOMM ของ MSComm จะสามารถตรวจจับสัญญาณนั้นได้ทันที ส่วนทางเลือกที่สองเป็นการคอยตรวจสอบค่าเหตุการณ์และความผิดพลาดที่เกิดขึ้นด้วยการดูค่าที่เปลี่ยนปแปลงภายในคุณสมบัติ CommEvent หลังจากให้โปรแกรมทำงานในฟังก์ชั่นต่างๆ ไปเรียบร้อยแล้ว ซึ่งวิธีนี้ใช้งานได้ดีในกรณีที่โปรแกรมขนาดเล็ก



รูปที่ 3.10 แสดงลักษณะ Mscomm

Control MSComm 1 ตัวสามารถควบคุมการทำงานของพอร์ตอนุกรมได้ 1 พอร์ต ถ้าในโปรแกรมที่ใช้งานต้องการติดต่อกับพอร์ตอนุกรมมากกว่า 1 พอร์ตจะต้องใช้ Control MSComm มากกว่า 1 ตัวเพื่อควบคุมพอร์ตอนุกรมในแต่ละพอร์ต แอดเดรสของพอร์ตอนุกรมและแอดเดรสของการเกิดอินเตอร์รัปต์สามารถเปลี่ยนแปลงได้จากการแก้ไขค่าที่ Control Panel

ถึงแม้ว่า คอนโทรล MSComm จะมีคุณสมบัติ (Property) มากมายหลากหลายตัว แต่สามารถทำความเข้าใจได้ไม่ยากดังนี้

CommPort

ใช้ในการกำหนดและอ่านค่าพอร์ตอนุกรมที่ติดต่ออยู่ (COM1,COM2,COM3,COM4)

รูปแบบการใช้งาน

object.CommPort[= value]

โดย Value เป็นค่าของพอร์ตอนุกรม ชนิดของข้อมูลเป็น Integer ค่า Value สามารถกำหนดได้ในช่วง 1-16 (ค่าเริ่มต้นกำหนดไว้ที่ 1) เมื่อมีการกำหนดค่าแล้วมีการเปิดพอร์ตโดยใช้คุณสมบัติ Port Open การเขียนโปรแกรมจึงจำเป็นต้องกำหนดตำแหน่งของพอร์ตอนุกรมก่อนที่ใช้คุณสมบัติ PortOpen

Setting

ใช้ในการกำหนดและอ่านค่าอัตราบอด,พาริตี้,จำนวนของบิตข้อมูล,จำนวนของบิตปิดท้าย

รูปแบบการใช้งาน

object.Setting [= value]

ค่า Value ชนิดข้อมูลเป็นแบบ String มีรูปแบบเป็น “BBB,P,D,S” โดย BBBB เป็นค่าอัตราบอด, P เป็นค่าพาริตี้, D เป็นจำนวนข้อมูล และ S เป็นจำนวนของบิตปิดท้าย ปกติแล้วค่านี้ถูกกำหนดไว้เป็น “9600,N,8,1”

PortOpen

ใช้ในการกำหนดและอ่านค่าสถานะของพอร์ตอนุกรม เพื่อเปิดและปิดพอร์ตอนุกรม

รูปแบบการใช้งาน

object.PortOpen [= value]

ค่า Value มีชนิดของข้อมูลเป็นแบบบูลีนคือ True กับ False โดย True หมายถึงการเปิดพอร์ตอนุกรมและ False หมายถึงการปิดพอร์ตอนุกรม สำหรับการปิดพอร์ตอนุกรมนั้นจะมีการเคลียร์บัฟเฟอร์รับข้อมูลและบัฟเฟอร์ส่งข้อมูลด้วย คอนโทรล MSComm จะปิดพอร์ตอนุกรมโดยอัตโนมัติเมื่อออกจากโปรแกรม ก่อนที่จะใช้คุณสมบัติ PortOpen ต้องตรวจสอบให้แน่ใจก่อนว่าคุณสมบัติ CommPort นั้นได้ทำการกำหนดตำแหน่งของพอร์ตอนุกรมไว้ถูกต้องหรือไม่

ถ้าคุณสมบัติ DTREnable หรือ RTSEnable ถูกกำหนดให้เป็น True ก่อนที่จะทำการเปิดพอร์ตค่าคุณสมบัติของ DTREnable หรือ RTSEnable จะถูกเซ็ตเป็น False หลังจากปิดพอร์ต แต่ถ้าเซ็ตเป็น False หลังจากปิดโปรแกรมแล้ว ค่าที่กำหนดไว้จะเป็นเท่าเดิม

การเปิด Serial Port

การเปิดพอร์ตสื่อสารด้วย MSComm Control สามารถทำได้ง่าย ๆ ด้วยการกำหนด Properties ดังนี้

```
MSComm1.ComPort = 4
MSComm1.Settings = “19200,N,8,1”
MSComm1.PortOpen = True
```

Input

อ่านค่าและลบค่าขบวนข้อมูลจากบัฟเฟอร์ภาครับ

รูปแบบการใช้งาน

object . input

คุณสมบัติ inputLen เป็นตัวกำหนดจำนวนของตัวอักษรที่จะอ่านโดยคุณสมบัติ Input การกำหนดค่าให้ inputLen เท่ากับ 0 เป็นการกำหนดให้คุณสมบัติ Input ทำการอ่านข้อมูลในบัฟเฟอร์รับข้อมูลทั้งหมด

คุณสมบัติ InputMode เป็นตัวกำหนดชนิดของข้อมูลที่คุณสมบัติ Input รับเข้ามา ถ้า InputMode ถูกกำหนดเป็น comInputModeText คุณสมบัติ Input จะส่งค่าข้อมูลกลับมาในรูปแบบ

ของข้อความชนิดชนิดข้อมูลเป็นแบบ Variant ถ้า InputMode กำหนดเป็น comInputModeBinary คุณสมบัติ Input จะส่งข้อมูลกลับมาในรูปของไบนารี่และชนิดข้อมูลเป็นแบบ Variant ในตัวอย่างโปรแกรมที่ 3 – 1 แสดงให้เห็นถึงวิธีในการรับข้อมูลจากบัฟเฟอร์รับข้อมูล

โปรแกรมที่ 3 – 1

```
Private Sub Command1_Click( )
Dim InString as String
‘ Retrieve all available data
        MSComm1.InputLen = 0
‘ Check for data .
        If MSComm1 . InBuffercount Then
‘ Read data .
            InString = MSComm1 . Input
        End If
End Sub
```

Output

ใช้ในการส่งขบวนของข้อมูลไปยังบัฟเฟอร์ส่งข้อมูล

รูปแบบการใช้งาน

object . Output [= value]

ค่า value เป็นค่าของตัวอักษรที่เขียนไปยังบัฟเฟอร์ส่งข้อมูล คุณสมบัติ Output สามารถใช้ในการส่งข้อมูลตัวอักษรหรือข้อมูลไบนารี่ก็ได้ โดยการส่งข้อมูลเป็นรูปแบบตัวอักษรจะต้องกำหนดข้อมูลเป็นแบบ Variant และมีข้อมูลภายในเป็นแบบ String สำหรับการส่งข้อมูลไบนารี่จะต้องกำหนดชนิดของข้อมูลเป็นแบบ Variant และมีข้อมูลภายในเป็นแบบ Byte ตัวอย่างโปรแกรมที่ 3 – 2 เป็นการส่งค่าที่ป้อนจากคีย์บอร์ดทุก ๆ ตัวไปยังพอร์ตอนุกรม

โปรแกรมที่ 3 – 2

```
Private Sub Form_KeyPress ( KeyAscii As Integer )
Dim Buffer as Variant
‘ Set and open port
        MSComm1 . CommPort = 1
        MSComm1 . PortOpen = True
        Buffer = Chr$ ( KeyAscii )
```

```
MSComm1 . Output = Buffer
End Sub
```

DTREnable

ใช้ในการอีนาเบิลขา Data Terminal Ready (DTR) โดยสัญญาณของขา DTR จะส่งจากคอมพิวเตอร์ไปยังโมเด็มเพื่อแสดงว่าคอมพิวเตอร์พร้อมที่จะรับข้อมูลแล้ว ชนิดของข้อมูลเป็นแบบ Boolean

รูปแบบการใช้งาน

object . DTREnable [= value]

ค่า value เป็นค่าสถานะ True หรือ False เพื่ออีนาเบิลหรือดิสเอเบิลขา DTR โดย

True หมายถึง อีนาเบิลขา DTR

False หมายถึง ดิสเอเบิลขา DTR (เป็นค่าปกติ)

เมื่อขา DTR ถูกกำหนดสถานะให้เป็น True ที่ขา DTR จะมีสถานะลอจิก "I" เมื่อทำการเปิดพอร์ตและจะมีสถานะเป็น "0" เมื่อมีการปิดพอร์ต เมื่อขา DTR ถูกกำหนดสถานะเป็น False ที่ขา DTR จะมีสถานะลอจิก "0" ตลอดเวลาไม่ว่าจะใช้คำสั่งเปิดพอร์ต หรือปิดพอร์ต

RTSEnable

ใช้เพื่ออีนาเบิลขา Request To Send (RTS) โดยขา RTS จะเป็นสัญญาณที่ส่งจากคอมพิวเตอร์ไปยังโมเด็มเพื่อร้องขอส่งข้อมูลชนิดของข้อมูลเป็นแบบ Boolean

รูปแบบการใช้งาน

object . RTSEnable [= value]

ค่า value เป็นค่าสถานะ True หรือหรือ False เพื่ออีนาเบิลหรือดิสเอเบิลขา RTS โดย

True หมายถึง อีนาเบิลขา RTS

False หมายถึง ดิสเอเบิลขา RTS (เป็นค่าปกติ)

เมื่อขา RTSEnable ถูกกำหนดให้เป็น True ขา RTS จะมีสถานะลอจิก "I" เมื่อเปิดพอร์ตและมีสถานะลอจิก "0" เมื่อปิดพอร์ต

EOFEnable

เป็นการกำหนดให้ MSComm รอสัญลักษณ์แสดงส่วนท้ายสุดของไฟล์ (End of file : EOF) ระหว่างการรับอินพุตเข้ามา ถ้าพบสัญลักษณ์ EOF ภาคอินพุตจะหยุดรับข้อมูล และเหตุ

การณ์ OnComm จะถูกกระตุ้นให้ทำงาน คุณสมบัติ CommEvent จะมีค่าเท่ากับ 7 หรือ ComEvEOF

รูปแบบการใช้งาน

object . EOFEnable [= value]

ค่า value เป็นค่าสถานะ True หรือหรือ False เพื่ออีนาเบิลหรือดิสเอเบิลการทำงานของเหตุการณ์ OnComm เมื่อตรวจพบสัญลักษณ์ EOF โดย

True หมายถึง เหตุการณ์ OnComm จะถูกกระตุ้นให้ทำงานด้วย EOF

False หมายถึง เหตุการณ์ OnComm จะไม่ถูกกระตุ้นให้ทำงานด้วย EOF (เป็นค่าปกติ)

เมื่อ EOFEnble กำหนดให้เป็น False ส่วนควบคุมจะไม่มีการตรวจสอบสัญลักษณ์ EOF

CTSHolding

ผู้ใช้งานสามารถตรวจสอบการทำงานของขา Clear To Send (CTS) ได้ว่ามีสถานะลอจิก "0" หรือ "1" โดยค่าที่อ่านได้จะเป็นบูลีน True และ False ถ้าค่า CTSHolding เป็น True ขา CTS จะสถานะลอจิกเป็น "1" ถ้าค่า CTSHolding เป็น False ขา CTS จะมีสถานะลอจิกเป็น "0"

รูปแบบการใช้งาน

object . CTSHolding

เมื่อขา CTS เป็นลอจิก "0" (CTSHolding = False) และเกิดไทม์เอาต์ คอนโทรล MSComm จะกำหนดให้คุณสมบัติ CommEvent มีค่าเป็น comEventCTSTO (Clear To Send Timeout) และกระตุ้นให้เกิดเหตุการณ์ OnComm

CDHolding

ผู้ใช้งานสามารถตรวจสอบการทำงานของขา Data Carrier Detect (DCD) ได้ว่ามีสถานะลอจิกเป็น "1" หรือ "0" โดยค่าที่อ่านได้จะเป็นบูลีน True และ False ถ้าค่า CDHolding เป็น True ขา DCD จะมีสถานะลอจิก "1" ถ้าค่า CDHolding เป็น False ขา DCD จะมีสถานะลอจิก "0"

รูปแบบการใช้งาน

object . CDHolding

เมื่อขา DCD มีลอจิก "1" (CDHolding = True) และเกิดไทม์เอาต์ คอนโทรล MSComm จะกำหนดให้คุณสมบัติ CommEvent มีค่าเป็น comEventCDTO (Carrier Detect Timeout Error) และกระตุ้นให้เกิดเหตุการณ์ OnComm

3.8 การใช้ MSComm เพื่อการติดต่อฮาร์ดแวร์

จะเห็นได้ว่าวิธีการที่จะอ่านค่าหรือเขียนค่าไปยังขาสถานะและขาควบคุมของพอร์ตอนุกรมสามารถทำได้ง่ายมากโดยใช้คำสั่งเหล่านี้

DTREnable	สำหรับสั่งให้ขา DTR มีลอจิก "0" หรือ "1"
RTSEnable	สำหรับสั่งให้ขา RTS มีลอจิก "0" หรือ "1"
CTSHolding	สำหรับอ่านค่าสถานะจากขา CTS ว่ามีลอจิก "0" หรือ "1"
CDHolding	สำหรับอ่านค่าสถานะจากขา CD ว่ามีลอจิก "0" หรือ "1"
DSRHolding	สำหรับอ่านค่าสถานะจากขา DSR ว่ามีลอจิก "0" หรือ "1"
Break	สำหรับการสั่งให้ขา Txd มีลอจิก "0" หรือ "1"

บทที่ 4
โครงสร้างข้อมูล

4.1 อาเรย์ เรคคอร์ด และพ้อยน์เตอร์ (ARRAYS RECORDS AND POINTERS)

4.1.1 อาเรย์แบบต่อเนื่อง (LINEAR ARRAYS)

อาเรย์แบบต่อเนื่องเป็นลิสท์ที่มีจำนวนอิลิเมนท์จำกัด โดยอิลิเมนท์ที่เก็บอยู่ในอาเรย์เป็นข้อมูลชนิดเดียวกัน ดังนี้

(1) อิลิเมนท์ต่างๆของอาเรย์จะถูกอ้างถึงโดยเซทของตัวชี้ (index set) ที่มีจำนวนเท่ากับช่องอาเรย์

(2) อิลิเมนท์ต่างๆของอาเรย์จะถูกเก็บในหน่วยความจำตำแหน่งที่ต่อเนื่องกัน จำนวนอิลิเมนท์ของอาเรย์ทั้งหมดเรียกว่า ความยาว (length) หรือขนาด (size) ของอาเรย์หากไม่มีการกำหนดเป็นอย่างอื่น เราจะสมมติว่าเซตของตัวชี้มีค่าเป็นเลขจำนวนเต็ม 1 , 2 ,...n โดยทั่วไปความยาวของอาเรย์จะได้จากสูตร

$$\text{Length} = UB - LB + 1$$

เมื่อ UB เป็นตัวชี้สูงสุด เรียกว่า upper bound และ LB เป็นค่าของตัวชี้ต่ำสุดเรียก lower bound ว่าของอาเรย์ ความยาวของอาเรย์เท่ากับ UN หาก LB = 1 อิลิเมนท์ต่างๆของอาเรย์ A อาจแสดงโดยตัวเลขซับสคริป

$$A_1 , A_2 , A_3 , \ldots\ldots\ldots , A_n$$

หรือใช้วงเล็บ (ใช้ใน FORTRAN , PL/1 และ BASIC)

$$A(1) , A(2) , \ldots\ldots\ldots , A(N)$$

หรือใช้วงเล็บใหญ่ (ใช้ใน PASCAL)

$$A[1], A[2], A[3],\ldots\ldots\ldots, A[N]$$

เราจะใช้ซับสคริปหรือวงเล็บใหญ่เป็นส่วนมาก โดยค่า K ที่อยู่ใน A[K] เรียกซับสคริปหรือตัวชี้ และ A[K] เรียกว่า ตัวแปรซับสคริป (Subscripted variable)

4.1.2 การเข้าถึงอาเรย์ในหน่วยความจำ : การชี้ตำแหน่ง (ACCESSINS LINEAR ARRAYS IN MEMORY ; INDEXING)

กำหนด LA เป็นอาเรย์ในหน่วยความจำของคอมพิวเตอร์ ตำแหน่งของหน่วยความจำในคอมพิวเตอร์จะต่อเนื่องกัน ดังแสดงให้เห็นดังรูป และเราสามารถเข้าถึงข้อมูลในทุกตำแหน่งได้

โดยตรงหากเราทราบตำแหน่ง ในหัวข้อนี้จะเป็นการกล่าวถึงวิธีการเข้าถึงข้อมูลที่อยู่ในอิลิเมนท์ต่างๆของอเรย์ LA

โดยใช้การกำหนดต่อไปนี้

LOC (LA[K]) = ตำแหน่งของอิลิเมนท์ LA[K] ของอาเรย์ LA

Base (LA) = ตำแหน่งของอิลิเมนทีของ LA

คอมพิวเตอร์เก็บเพียงตำแหน่ง Base (LA) ซึ่งเรียกว่า เป็นตำแหน่งเริ่มต้น (base address) ของ LA โดยการใช้ตำแหน่งเริ่มต้น ร่วมกับลักษณะของการจัดเก็บข้อมูลของอิลิเมนท์ต่างๆ ต่อเนื่องกัน คอมพิวเตอร์สามารถคำนวณ ตำแหน่งของอิลิเมนท์ใดๆ ของอาเรย์ LA ได้จากสูตร

$$LOC(LA[K]) = Base(LA) + w(K - lower\ bound)$$

เมื่อ w เป็นจำนวนช่องของหน่วยความจำต่อข้อมูล 1 อิลิเมนท์ของอาเรย์ LA เวลาที่ใช้คำนวณในแต่ละค่าของ K จะเท่ากัน เมื่อกำหนด K ให้สามารถคำนวณหาตำแหน่งและเข้าถึงข้อมูลของ LA[K] ได้โดยไม่ต้องผ่านอิลิเมนท์อื่นของ LA

รูปที่ 4.1 หน่วยความจำของคอมพิวเตอร์

4.1.3 การแทรกและการลบ (INSERTING AND DELETING)

ให้ A เป็นอาเรย์ที่อยู่ในหน่วยความจำของคอมพิวเตอร์ การแทรกเป็นการเพิ่มข้อความ 1 อิลิเมนท์เข้าใน A และการลบเป็นการเอาข้อมูล 1 อิลิเมนท์ ออกจากอาเรย์ A

การเพิ่มข้อมูลอีก 1 อิลิเมนท์ในส่วนท้ายของอาเรย์ทำได้ง่าย หากพื้นที่ของหน่วยความจำที่จัดเตรียมไว้ใหญ่พอ แต่ถ้าหากต้องการเพิ่มข้อมูลเข้าตรงกลางของอาเรย์ จะต้องมีการเคลื่อนย้ายอิลิเมนท์ครึ่งหนึ่งลงด้านล่าง เพื่อให้เกิดช่องว่างสำหรับอิลิเมนท์ใหม่

ในทำนองเดียวกัน การลบข้อมูล 1 อิลิเมนท์ที่อยู่ท้ายสุดของอาเรย์ก็ทำได้ง่าย แต่ถ้าต้องการลบข้อมูล 1 อิลิเมนท์ ที่อยู่ตรงกลางของอาเรย์จะต้องมีการย้ายข้อมูลของอิลิเมนท์ต่างๆ ที่อยู่ด้านล่างขึ้นมา

4.1.4 การเรียงลำดับ ; บับเบิ้ลซอร์ท (SORTING ; BUBBLE SORT)

กำหนด A เป็นลิสท์ของเลข n จำนวนการเรียงลำดับเป็นการจัดเรียงให้ข้อมูลอยู่ในลำดับต่อเนื่องกันเช่นเรียงจากน้อยไปมาก จะได้ความสัมพันธ์ข้อมูลที่เรียงแล้วดังนี้

$$A[1] < A[2] < A[3] < \ldots\ldots\ldots\ldots < A[N]$$

ตัวอย่างเช่น ค่าเริ่มต้นของ A เป็น

8, 4, 19, 2, 7, 13, 5, 16

หลังจากจัดเรียงแล้วจะเป็น

2, 4, 5, 7, 8, 13, 16, 19

การเรียงข้อมูลที่มีประสิทธิภาพ จะมีความซับซ้อนมาก ซึ่งมีวิธีการจัดเรียงข้อมูลที่แตกต่างกันมากมาย กำหนดลิสท์ของตัวเลข A[1], A[2] ..... A[n] อยู่ในหน่วยความจำ การจัดเรียงแบบบับเบิลซอร์ทจะเป็นดังนี้

Step1 เปรียบเทียบ A[1] กับ A[2] และเรียงตามลำดับ ได้ A[1] < A[2] แล้วจึงเปรียบเทียบ A[2] กับ A[3]แล้วจึงเรียงตามลำดับได้ A[2] < A[3] ต่อไป เปรียบเทียบ A[3] กับ A[4]แล้วจึงเรียงตามลำดับต่อไปได้ A[3] < A[4] แล้วจึงเรียงตามลำดับต่อไปได้ A[N-1] < A[N]

Step2 ทำซ้ำ Step1 แต่การเปรียบเทียบลดลง 1 ครั้ง จะหยุดเปรียบเทียบเมื่อมีการจัดเรียงระหว่าง A[N-2] และ A[N-1] แล้ว (การเปรียบเทียบจะมีจำนวน N-2 ครั้ง และค่ามากที่สุดอันดับ 2 จะอยู่ที่ A[N-1]

Step3 ทำซ้ำ Step1 แต่หยุดหลังจากเปรียบเทียบและจัดเรียงระหว่าง A[N-2] กับ A[N-3]

Step4 เป็นการเปรียบเทียบ A[1] กับ A[2] และจัดลำดับระหว่าง A[1] กับ A[2] หลังจาก n-1 สเตป ข้อมูลจะถูกจัดเรียงตามลำดับจากน้อยไปมาก

หมายเหตุ : คำจำกัดความที่กล่าวมา เป็นการจัดเรียงจากน้อยไปมากเท่านั้น ในการจัดเรียง อาจเป็นการจัดเรียงตัวเลขจากมากไปน้อยก็ได้ หรือเรียงลำดับตามตัวอักษรก็ได้โดยปกติ A เป็นไฟล์ของเรคคอร์ดและการจัดเรียง A จะหมายถึง การจัดเรียงเร็คคอร์ดของ A ตามค่าคีย์ที่กำหนดให้จัดเรียงกันตามลำดับ

4.1.5 การค้นหาข้อมูล; ลิเนียร์เซิรช (SEARCHING,LINEAR SEARCH)

เมื่อให้ DATTA ประกอบด้วยอิลิเมนท์ต่างๆ ที่เก็บอยู่ในหน่วยความจำและ ITEM เป็นข้อมูลที่กำหนดให้ การค้นหาเป็นการตำแหน่ง LOC ของ ITEM ที่อยู่ใน DATA หรือแสดงข้อ

ความให้ทราบหาก ITEM ไม่อยู่ใน DATA การค้นหาจะประสบความสำเร็จเมื่อมีข้อมูลอยู่ใน DATA และจะไม่ประสบความสำเร็จในกรณีอื่น

บ่อยครั้งที่อาจมีการเพิ่มรายการ ITEM ลงใน DATA หากไม่พบ ITEM หลังจากการค้นหา จึงมีการใช้การค้นหาและการแทรกข้อมูลร่วมกันแทนที่จะใช้ การค้นหาอย่างเดียวซึ่งอัลกอริทึมการค้นหาและการแทรกจะพิจารณาในส่วนของคำถาม

มีอัลกอริทึมสำหรับการค้นหาข้อมูลอยู่หลายแบบซึ่งการเลือกใช้แบบใดนั้นขึ้นอยู่กับการจัดโครงสร้างของการเก็บข้อมูลจะกล่าวเฉพาะอัลกอริทึมที่ง่ายคือ การค้นหาแบบลิเนียร์เซิรช และหัวข้อต่อไปจะเป็นการกล่าวถึง อัลกอริทึมที่รู้จักกันดี คือ ไบนารีเซิรช

ความซับซ้อนของอัลกอริทึมในการค้นหาข้อมูลจะวัดจากจำนวนของที่ทำการเปรียบเทียบข้อมูลที่ต้องการค้นหา ITEM ใน DATA เมื่อ DATA มี n อิลิเมนท์

กำหนด DATA เป็นอาเรย์ต่อเนื่องขนาด n อิลิเมนท์ การค้นหาข้อมูล ITEM จาก DATA ทำได้โดยการเปรียบเทียบ ITEM กับแต่ละอิลิเมนท์ของ DATA นั่นคือ เปรียบเทียบ DATA[1] = ITEM แล้วจึงไปเปรียบเทียบ DATA[2] = ITEM แล้วเปรียบเทียบต่อไปเรื่อยๆ ซึ่งวิธีนี้จะเข้าถึงข้อมูลในทุกอิลิเมนท์ของ DATA โดยวิธีนี้เรียกว่า การค้นหาแบบลิเนียร์เซิรช

เพื่อให้การทำงานง่ายขึ้น ในตอนต้นจะกำหนด ITEM ให้กับ DATA[N+1] ซึ่งเป็นตำแหน่งต่อจากท้ายสุดของ DATA ซึ่งจะได้

$$LOC = N + 1$$

เมื่อ LOC คือ ตำแหน่งที่ ITEM ปรากฏอยู่ใน DATA ตัวแรก เพื่อเป็นการแสดงว่าไม่มีข้อมูล ITEM อยู่ใน DATA จุดประสงค์ของการกำหนดลักษณะนี้ ก็เพื่อป้องกันการตรวจสอบซ้ำเมื่อมาถึงข้อมูลท้ายสุด ซึ่งวิธีการนี้จะพบข้อมูลทุกครั้ง

4.1.6 การค้นหาข้อมูล ; ไบนารีเซิรช (SEARCHING, BINARY SEARCH)

สมมติ DATA เป็นอาเรย์ที่มีอิลิเมนท์เรียงลำดับจากน้อยไปมาก หรือเรียงตามตัวอักษรแล้ว อัลกอริทึมการค้นหาแบบไบนารีเซิรชจะมีประสิทธิภาพสูงในการหา ITEM ใน DATA ก่อนที่จะไปดูอัลกอริทึมขอเสนอแนวความคิดของอัลกอริทึมที่คุ้นเคยในชีวิตประจำวันก่อน

สมมติ เราต้องการหาตำแหน่งของชื่อบางชื่อในสมุดโทรศัพท์ (หรือคำบางคำในพจนานุกรม) เราจะไม่ทำการการค้นหาแบบลิเนียร์ เราจะเริ่มทำการค้นหา โดยเปิดสมุดโทรศัพท์ไปที่ประมาณครึ่งหนึ่ง เพื่อดูว่าชื่อที่ต้องการอยู่ครึ่งใด แล้วจึงเปิดไปที่ครึ่งหนึ่งของซีกที่มีชื่ออยู่ ซึ่งจะได้เป็น ¼ ของสมุดโทรศัพท์ที่มีชื่ออยู่ แล้วจึงเปิดไปที่ครึ่งหนึ่งของ ¼ นั้นและเปิดเช่นนี้ไปเรื่อยจนพบตำแหน่งของชื่อที่ต้องการค้นหา

อัลกอริทึมของการค้นหาแบบไบนารีที่ใช้แบบอาเรย์ DATA จะมีการทำงานดังนี้ในแต่ละขั้นของการค้นหา ITEM จะลดจำนวนอิลิเมนท์ของจำนวนค้นหาลงที่ละเซกเมนท์

DATA [BEG], DATA[BEG + 1], DATA [BEG + 2],……..DATA[END]

BEG และ END แสดงตำแหน่งเริ่มต้น และตำแหน่งท้ายสุดของเซกเมนท์ที่พิจารณา อัลกอริทึมเปรียบเทียบ ITEM กับข้อมูลที่อยู่กึ่งกลางของ DATA คือ DATA [MID] ของเซกเมนท์ ซึ่ง MIDได้จาก

MID = INT ((BEG + END)/2)

(ใช้ INT (A) สำหรับเลขจำนวนเต็มของ A) ถ้า DATA [MID] = ITEM แล้วการค้นหาจะสำเร็จและได้ LOC = MID หากไม่พบ ITEM จะต้องไปหาเซกเมนท์ใหม่ของ DATA ซึ่งได้จากลำดับขั้นต่อไปนี้

(1) ถ้า ITEM < DATA [MID], แล้ว ITEM จะปรากฏอยู่บนครึ่งซ้ายของเซกเมนท์

DATA[BEG],DATA[BEG + 1],……..DATA[MID – 1]

ดังนั้นเรากำหนด END = MID – 1 และเริ่มหาใหม่อีก

(2) ถ้า ITEM > DATA[MID] แล้ว ITEM จะปรากฏอยู่ในครึ่งขวาของเซกเมนท์

DATA[MID + 1], DATA[MID + 2]……..., DATA [END]

ดังนั้นเรากำหนด BEG = MID + 1 และเริ่มหาใหม่อีก

ในตอนเริ่มต้นกำหนดการค้นหาตลอดย่านของอาเรย์ โดยให้ BEG = 1 และ END = n หรือ BEG = LB และ END = UB

END < BEG

ถ้า ITEMไม่อยู่ใน DATA จะได้หรืออาจจะใช้การกำหนด LOC = NULL เมื่อ NULL เป็นค่าที่อยู่นอกเหนือจากตัวชี้อาเรย์ของ DATA (ในกรณีทั่วไป เราจะใช้ NULL = 0)

4.1.7 อาเรย์หลายมิติ (MULTIDIMENSIONAL ARRAYS)

อาเรย์ที่ได้กล่าวมาทั้งหมดเป็นอาเรย์แบบมิติเดียว เนื่องจากแต่ละอิลิเมนท์ในอาเรย์สามารถอ้างถึงได้โดยใช้ซับสคริปตัวเดียว โปรแกรมคอมพิวเตอร์หลายภาษายอมให้กำหนดอาเรย์หลายมิติได้ ซึ่งการอ้างถึงข้อมูลจะใช้ซับสคริปมากกว่า 1 ตัว อาเรย์หลายมิติก็มีการจัดเก็บเป็นบล็อกของหน่วยความจำเช่นกัน เราสามารถใช้ซับสคริปเพื่อกำหนดตำแหน่งติดต่อกับอิลิเมนท์ต่างๆได้โดยตรง

อาเรย์ 2 มิติ (Two – dimension Array)

อาเรย์ 2 มิติ A ขนาด m × n จะเก็บข้อมูลได้ mn อิลิเมนท์ โดยแต่ละอิลิเมนท์กำหนดโดยคู่ของเลขจำนวนเต็ม (เช่น J, K) ซึ่งเรียกว่า ซับสคริป มีคุณสมบัติดังนี้

$1 \leq J \leq m$ และ $1 \leq K \leq n$

อิลิเมนท์ของ A ที่มีซับสคริปแรกเป็น j และตัวที่สองเป็น k จะแสดงโดย

$A_{j,k}$ หรือ A [J,K]

เซตของอิลิเมนท์ที่อ้างอิงโดยใช้ซับสคริป 2 ตัว ในทางคณิตศาสตร์เรียกว่าเมตริก (matrices) ในทางธุรกิจเรียกว่าตาราง (tables) ดังนั้น บางครั้งจะเรียกอาเรย์ 2 มิติว่า เมตริกอาเรย์ (matrices arrays)

มาตรฐานการแสดงอาเรย์ 2 มิติ A ขนาด m × n โดยอิลิเมนท์ต่างๆของ A จะประกอบเป็นอาเรย์แบบสี่เหลี่ยมที่มีขนาด m โรว์ และ n คอลัมน์ซึ่งมี A [J,K] เป็นอิลิเมนท์ในโรว์ J และคอลัมน์ K (โรว์เป็นแถวแนวนอน และคอลัมน์เป็นแถวแนวตั้ง) แสดงอาเรย์ A ขนาด 3 โรว์ 4 คอลัมน์ แต่ละโรว์จะประกอบด้วย อิลิเมนท์ที่มีซับสคริปตัวแรกเหมือนกัน และแต่ละคอลัมน์ประกอบด้วยอิลิเมนท์ที่มีซับสคริปตัวที่ 2 เหมือนกัน

		Columns			
		1	2	3	4
Rows		A[1,1]	A[1,2]	A[1,3]	A[1,4]
		A[2,1]	A[2,2]	A[2,3]	A[2,4]
		A[3,1]	A[3,2]	A[3,3]	A[3,4]

รูปที่ 4.2 อาเรย์ 2 มิติ ขนาด 3X4

4.1.8 พ้อยเตอร์ ; อาเรย์ของพ้อยน์เตอร์ (POINTER ; POINTER ARRAYS)

ให้ DATA เป็นอาเรย์ใดๆ ตัวแปร P เรียกว่า พ้อยเตอร์ หาก P ชี้ไปยังอิลิเมนท์ใดๆ ใน DATA นั่นคือ P จะเก็บตำแหน่งของอิลิอเมนท์ใน DATA อาเรย์ PTR เรียกว่า อาเรย์ของพ้อยเตอร์ ถ้าหากอิลิเมนท์ของ PTR เป็นพ้อยเตอร์ พ้อยเตอร์และอาเรย์ของพ้อยเตอร์จะช่วยทำให้การกระทำกับข้อมูลง่ายขึ้น

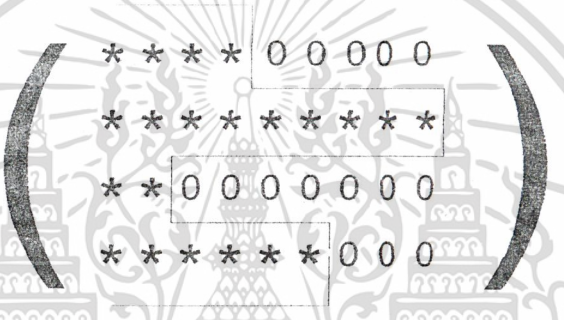
พิจารณาองค์กรหนึ่งซึ่งแบ่งออกเป็น 4 กลุ่ม โดยแต่ละกลุ่มมีรายชื่อเรียงกันตามตัวอักษร ซึ่งแสดงดังรูป 4.3 โดยมีสมาชิกทั้งหมด 21 คน แต่ละกลุ่มมี 4,9,2 และ 6 ตามลำดับ

Group 1	Group 2	Group 3	Group 4
Evans	Conrad	Davis	Baker
Harris	Felt	Segal	Cooper
Lewis	Glass		Ford
Shaw	Hill		Gray
	King		Jones
	Penn		Reed
	Silver		
	Troy		
	Wagner		

รูปที่ 4.3 กลุ่มสมาชิกข้อมูล

หากรายชื่อของสมาชิกเก็บอยู่ในหน่วยความจำตามลำดับกลุ่ม วิธีที่จะเก็บข้อมูลลงในหน่วยความจำวิธีหนึ่งคือ ใช้อาเรย์ 2 มิติ ขนาด 4 X n โดยแต่ละโรว์เป็นกลุ่ม หรือใช้อาเรย์ขนาด n X 4 เมื่อคอลัมน์เป็นกลุ่มการเก็บข้อมูล ในลักษณะของอาเรย์แบบนี้สามารถเข้าถึงข้อมูลในแต่ละกลุ่มได้ แต่จะต้องเสียพื้นที่ไปเมื่อกลุ่มใหญ่ขึ้นข้อมูลในรูปที่ 4.3 จะใช้อาเรย์ขนาด 4 X 9 หรือ 9 X 4 ในการเก็บข้อมูลเพียง 21ชื่อสำหรับรูป 4.4 เป็นการแสดงการเก็บข้อมูลของอาเรย์ขนาด 4X 9

* หมายถึง ตำแหน่งที่เก็บข้อมูล และ 0 หมายถึง ตำแหน่งที่ไม่ใช้ (อาเรย์ที่โรว์หรือคอลัมน์ เริ่มด้วยจำนวนข้อมูลที่แตกต่างกัน และจบด้วยตำแหน่งที่ไม่มีการเก็บข้อมูล เราเรียกว่า jagged)



รูปที่ 4.4 กลุ่มข้อมูลในหน่วยความจำ

4.2 การเรียงลำดับและการค้นหาข้อมูล (SORTING AND SEARCHING)

4.2.1 การเรียงลำดับ (SORTING)

ลิสท์ A มี n อิลิเมนท์ $A_1, A_2, \ldots, A_n$ เก็บอยู่ในหน่วยความจำ การเรียงลำดับข้อมูลใน A เป็นการกระทำเพื่อจัดเรียงข้อมูลใน A ตามลำดับเช่น จากน้อยไปมากซึ่งจะได้ความสัมพันธ์ของข้อมูลที่จัดเรียงแล้วเป็นดังนี้

$$A_1, A_2, A_3, \ldots, A_n$$

เนื่องจาก A มีจำนวน n อิลิเมนท์ดังนั้นการจัดเก็บข้อมูลใน A จะเป็นไปได้ n! แบบ ความซับซ้อนของอัลกอริทึมการเรียงลำดับ

ความซับซ้อนของอัลกอริทึมวัดจากเวลาที่ใช้ซึ่งอยู่ในรูปฟังก์ชั่นของ n ซึ่งเป็นจำนวนข้อมูลที่ถูกจัดเรียง อัลกอริทึมของการเรียงลำดับที่มี $A_1, A_2, \ldots A_n$ เป็นที่เก็บข้อมูลที่ต้องการเรียงลำดับ และ B เป็นตำแหน่งสำรอง จะมีการทำงานดังนี้

(1) เปรียบเทียบว่า $A_i < A_j$ หรือ $A_i < B$

(2) สลับที่ซึ่งเป็นการสลับค่าของ A_i กับ A_j หรือ A_i กับ B

(3) กำหนด B := A_I แล้วกำหนด A_J := B หรือ A_J := A_I

ปกติแล้วฟังก์ชั่นความซับซ้อนจะวัดจากจำนวนของการเปรียบเทียบ เนื่องจากจำนวนการกระทำอย่างอื่นจะเป็นแฟกเตอร์ที่คงที่ของจำนวนการเปรียบเทียบ

4.2.2 อินเซิรชชั่นซอร์ท (INSERTION SORT)

เมื่ออาเรย์ A มีขนาด n อิลิเมนท์ A [1], A [2],.....,A [N] เก็บอยู่ในหน่วยความจำ อัลกอริทึมของอินเซิรชชั่นซอร์ทจะสแกน A จาก A [1] ถึง A [N] แล้วแทรกแต่ละอิลิเมนท์ A [K] ลงในตำแหน่งที่เหมาะสมของอาเรย์ A [1], A [2],.....A [K – 1] ที่มีการจัดเรียงอยู่แล้ว ซึ่งมีการทำงานดังนี้

พาส 1 A [1] อยู่ที่จัดเรียงอยู่แล้ว

พาส 2 A [1] จะถูกแทรกก่อนหรือหลัง A [1] เพื่อทำให้ A [1], A[2] จัดเรียง

พาส 3 A [3] ถูกแทรกลงในตำแหน่งที่เหมาะสม เพื่อให้เกิดการจัดเรียงของ A [1], A [2], A [3]

พาส 4 A [4] ถูกแทรกลงในตำแหน่งที่เหมาะสม เพื่อทำให้เกิดการจัดเรียงของ A [1], A [2], A [3], A [4]

พาส N A [N] ถูกแทรกเพื่อทำให้เกิดการจัดเรียงของ A [1], A[2], A [3],....., A [N] อัลกอริทึมของการจัดเรียงลำดับแบบนี้ จะใช้กับกรณีที่ n มีค่าน้อย

การแทรก A [K] ลงในตำแหน่งที่เหมาะสมลงในอาเรย์ย่อยที่จัดเรียงแล้ว A [1] , A [2] ,...., A [K – 1] ทำโดยการเปรียบเทียบ A [K] กับ A [K – 1] เปรียบเทียบ A [K] กับ A [K – 2] เปรียบเทียบ A [K] กับ A [K – 3]และต่อไปเรื่อย ๆ จนกระทั่งพบอิลิเมนท์ A [J] ที่มี A [J] , A [K]แล้วย้ายอิลิเมนท์ A [K – 1], A [K – 2],....., A [J + 1] ไปข้างหน้า 1 ตำแหน่ง และแทรก A [K] ในตำแหน่งที่ J + 1

อัลกอริทึมจะง่ายขึ้น ถ้าอิลิเมนท์ A [J] มีค่า A [J] $\sqsupset$ A [K] เสมอ หากไม่เป็นไปตามนี้จะต้องตรวจสอบว่า เป็นการเปรียบเทียบ A [K] กับ A [1] หรือยังซึ่งการตรวจสอบนี้จะใช้การกำหนดอิลิเมนท์แสดงตัวสุดท้าย คือ A [0] = ∞ (หรือค่าที่น้อยมากๆ)

Pass	A[0]	A[1]	A[2]	A[3]	A[4]	A[5]	A[6]	A[7]	A[8]
K=1:	-∞	77	33	44	11	88	22	66	55
K=2:	-∞	77	33	44	11	88	22	66	55
K=3:	-∞	33	77	44	11	88	22	66	55
K=4:	-∞	33	44	77	11	88	22	66	55
K=5:	-∞	11	33	44	77	88	22	66	55
K=6:	-∞	11	33	44	77	88	22	66	55
K=7:	-∞	11	22	33	44	77	88	66	55
K=8:	-∞	11	22	33	44	66	77	88	55
Sorted:	-∞	11	22	33	44	55	66	77	88

รูปที่ 4.5 อินเซิรชชั่นซอร์ท (INSERTION SORT)

4.2.3 ซิเลคชั่นซอร์ท (SELECTION SORT)

อาเรย์ A ขนาด n อิลิเมนท์ A [1], A[2],....., A [N]เก็บอยู่ในหน่วยความจำ อัลกอริทึมซิเลคชั่นซอร์ทจะทำงานดังนี้ หาค่าน้อยสุดในลิสท์ แล้วใส่ลงในตำแหน่งแรก แล้วหาค่าน้อยที่สุดอันดับ 2 แล้วใส่ลงตำแหน่งที่ 2 แล้วทำต่อไปเรื่อยๆซึ่งจะเป็นดังนี้

พาส 1 หาตำแหน่ง LOC ของค่าน้อยที่สุดในลิสท์ของ N อิลิเมนท์

A [1], A[2],....., A [N] แล้วสลับ A [LOC] กับ A [1] จะทำให้ A [1] มีการเรียงลำดับ

พาส 2 หาตำแหน่ง LOC ของค่าน้อยที่สุดในลิสท์ย่อยของ N – 1 อิลิเมนท์

A [2], A [3],....., A [N] แล้วสลับ A [LOC] กับ A [2] ทำให้ A [1] , A [2] มีการจัดเรียงเนื่องจาก A [1] , A [2]

พาส 3 หาตำแหน่ง LOC ของค่าน้อยที่สุดในลิสท์ย่อยของ N – 2 อิลิเมนท์

A [3], A [4],....., A [N] แล้วสลับ A [LOC] กับ A [3] ทำให้ A [1], A [2], A [3] มีการจัดเรียงเนื่องจาก A [2] , A [3]

พาส N หาตำแหน่ง LOC ของค่าน้อยที่สุดของ A [N – 1] , A [N] แล้วสลับ A [LOC] กับ A [N – 1] ทำให้ A [1], A [2],....., A [N] มีการจัดเรียง

A จะมีการเรียงลำดับ เมื่อมีการทำงานผ่านไป N – 41 พาส

4.2.4 เมิรกกิ้ง (MERGINE)

A เป็นลิสท์ที่มีการเรียงลำดับข้อมูลขนาด r อิลิเมนท์ และ B เป็นลิสท์ที่มีการเรียงลำดับข้อมูลขนาด s อิลิเมนท์ การทำงานเพื่อรวมอิลิเมนท์ของ A และ B ลงในลิสท์ที่เรียงลำดับ C ที่มีขนาด n = r + s เรียกว่า เมิรกกิ้ง การรวมกันอาจใช้วิธีการวางอิลิเมนท์ของ B ต่อจากอิลิเมนท์ของ

A แล้วใช้วิธีการเรียงลำดับวิธีใดวิธีหนึ่ง ซึ่งวิธีนี้ไม่ได้ใช้ประโยชน์ของข้อมูลใน A และBที่จัดเรียงแล้วอัลกอรึทึมที่มีประสิทธิภาพในการรวมข้อมูล

สมมติมีคาร์ดที่จัดเรียงแล้ว 2 ชุด ต้องการรวมเข้าด้วยกัน ในแต่ละขั้นตอนคาร์ดที่อยู่หน้าสุดในแต่ละกลุ่มจะถูกเปรียบเทียบกัน และค่าที่น้อยกว่าจะถูกใส่ในกองที่รวม เมื่อกองใดหมดกองที่เหลืออยู่จะนำไปต่อท้ายกองใหม่ในทำนองเดียวกันหากมีนักศึกษา 2 แถวเรียงลำดับความสูงและต้องการรวมให้เป็นแถวเดียวกัน จะเริ่มจากการเปรียบเทียบนักเรียนที่อยู่หัวแถวของทั้งสองแล้วนำ ผู้ที่เตี้ยกว่าไปยืนในแถวใหม่ จนกระทั่งแถวใดแถวหนึ่งหมด จึงเอาอีกแถวที่เหลือไปต่อกับแถวใหม่

จากการทำงานที่ผ่านมา สามารถแปลงเป็นอัลกอรึทึม เพื่อเป็นการรวมข้อมูลของอาเรย์ A ขนาด r อิลิเมนท์และอาเรย์ B ขนาด s อิลิเมนท์ที่มีการจัดเรียงลำดับแล้วลงในอาเรย์ C ขนาด n = r + s อิลิเมนท์ โดยกำหนด NA และ NB เป็นตำแหน่งเริ่มต้นตามลำดับ และ PTR เป็นตำแหน่งของ C ที่จะเพิ่มข้อมูลในตอนเริ่มต้น กำหนด NA := 1 , NB : = 1 และ PTR := 1 ในแต่ละขั้นตอนของอัลกอรึทึมเปรียบเทียบ

A[NA] and B[NB]

แล้วกำหนดค่าที่ต่ำกว่าลงใน C [PTR] และเพิ่ม PTR โดยใช้ PTR := PYR + 1 แล้วเพิ่ม NA หรือ NB ขึ้น 1 โดยใช้ NA := NA + 1 หรือ NB := NB + 1 ขึ้นอยู่กับว่าค่าใน C มาจาก A หรือ B ถ้า NA > r แล้ว ค่าที่เหลือใน B จะถูกนำมาต่อใน C หรือถ้า NB > s แล้วค่าที่เหลือใน A จะถูกนำมาต่อใน C

4.2.5 เรดิกซ์ซอร์ท (RADIX SORT)

เรดิกซ์ซอร์ทเป็นวิธีการที่ทำการจัดเรียง ซึ่งอยู่ภายในชีวิตประจำวัน โดยเริ่มจากการเรียงรายชื่อตามตัวอักษร โดยการจัดเรียงจะเริ่มจากเรียงตัวอักษรแรกก่อน ซึ่งจะแบ่งออกเป็น 26 กลุ่ม โดยกลุ่มแรกจะมีอักษร A นำ กลุ่มที่ 2 มีอักษร B และต่อไปเรื่อยๆในขั้นที่ 2 ทำการจัดเรียงในแต่ละกลุ่มตามตัวอักษรที่ 2 ของชื่อ และทำต่อไปเรื่อยๆถ้าชื่อมีความยาว 12 ตัวอักษรต้องทำการจัดเรียง 12 ครั้ง

เรดิกซ์ซอร์ทเป็นวิธีการที่ใช้กับเครื่องเรียงบัตร (card sorter) เครื่องเรียงบัตรจะมีกล่องรับบัตร 13 กล่อง ซึ่งมีป้ายดังนี้

9 , 8 , 7 , 6 , 5 , 4 , 3 , 2 , 1 , 0 , 11 , 12 R (reject)

แต่ละกล่องนอกจาก R จะสัมพันธ์กับโรว์บนบัตร ซึ่งจะเจาะรูไว้ เลขฐานสิบ (เรดิกซ์คือ 10) จะถูกตอกเฉพาะใน 10 กล่องแรกของเครื่องเรียง เครื่องเรียงใช้การเรียงกลับของตัวเลข (reverse – digit sort on number) ดังนี้ สมมติเครื่องเรียงต้องการเรียงบัตร ที่ประกอบด้วย เลข 3

หลัก ที่มีการเจาะรูในคอลัมน์ 1 ถึง 3 ขั้นแรกจะเรียงจากหลักหน่วย ในพาสที่ 2 จะเรียงหลักสิบ และในพาสที่ 3 จะเรียงจากหลักร้อย

4.2.6 การค้นหาและแก้ไขข้อมูล (SEARCHING AND DATA MODIFICATION)

S เป็นกลุ่มข้อมูลที่เก็บอยู่ในหน่วยความจำ ซึ่งมีโครงสร้างของข้อมูลอย่างใดอย่างหนึ่ง การค้นหาข้อมูลเป็นการหาตำแหน่ง LOC ในหน่วยความจำของข้อมูล ITEM การที่ให้กำหนดให้หรือให้ข้อมูลบางอย่างที่ ITEM ไม่อยู่ใน S การค้นหาจะสำเร็จหรือไม่ขึ้นอยู่กับว่า ITEM มีอยู่ใน S หรือไม่ อัลกอริทึมการค้นหาข้อมูลจะขึ้นอยู่กับลักษณะโครงสร้างข้อมูลที่ S เก็บในหน่วยความจำ

การแก้ไขข้อมูล เป็นการทำงานเกี่ยวกับการแทรก การลบ และการเปลี่ยนแปลง (update)ข้อมูล แต่ในส่วนของหนังสือเล่มนี้จะเป็นการแทรกและการลบเท่านั้น การทำงานจะใกล้เคียงกับการค้นหาข้อมูล เนื่องจากเราค้นหาตำแหน่งของ ITEM ที่ต้องการลบออกก่อนที่จะทำการลบหรือต้องทำการค้นหาตำแหน่งที่เหมาะสมสำหรับแทรก ITEM การแทรกและการลบข้อมูลจะใช้เวลาการทำงานบางส่วนซึ่งขึ้นอยู่กับลักษณะโครงสร้างข้อมูลที่ใช้

โดยทั่วไปโครงสร้างของข้อมูลที่ค้นหาได้เร็ว และโครงสร้างข้อมูลที่แก้ไขได้เร็วจะไม่เป็นโครงสร้างชนิดเดียวกันซึ่งสามารถแสดงให้เห็นดังต่อไปนี้ เมื่อต้องการค้นหาและแก้ไขข้อมูลจากโครงสร้างข้อมูล 3 แบบ ที่ผ่านมาแล้ว

(1) อาเรย์ที่มีการจัดเรียง เราสามารถใช้ไบนารีเซิรชเพื่อหาตำแหน่ง LOC ของ ITEM ที่กำหนดให้โดยใช้เวลา 0 (log n) แต่แทรกหรือลบข้อมูลทำได้ช้ามาก เพราะว่าต้องการย้ายข้อมูลเมื่อมีการแทรกหรือลบ ซึ่งใช้เวลาเฉลี่ย n/2 = 0 (n) จะเห็นว่าอาเรย์ที่มีการจัดเรียงจะดีสำหรับการค้นหาข้อมูล แต่การแก้ไขข้อมูลทำได้ช้ามาก

(2) ลิงค์ลิสท์ การค้นหาข้อมูลจะใช้ลิเนียร์เซิรชในการหาตำแหน่ง LOC ของ ITEM ที่กำหนดให้ ซึ่งเสียเวลามากประมาณ 0 (n) แต่การแทรกและการลบทำได้เร็ว โดยเปลี่ยนพ้อยน์เตอร์เพียง 2 - 3 ตัว ดังนั้นลิงค์ลิสท์จะเหมาะสำหรับการแก้ไขข้อมูล เช่น การทำงานของการเวิร์ดโปรเซสซิ่ง

(3) ไบนารีเซิรชทรี โครงสร้างข้อมูลนี้รวมข้อดีของอาเรย์ที่มีการจัดเรียงและลิงค์ไว้ด้วยกัน นั่นคือ การค้นหาจะทำการค้นหาเฉพาะในพาท P ในทรี T เท่านั้นซึ่งใช้การเปรียบเทียบเฉลี่ย 0 (log n) และทรี T เก็บในหน่วยความจำในลักษณะของลิงค์ลิสท์ ดังนั้นการลบและการเพิ่มข้อมูลจะมีการเปลี่ยนพ้อยน์เตอร์เพียงไม่กี่ตัว

ข้อเสียของไบนารีเซิรชทรี คือความยาวของพาท P ในแต่ละพาทไม่เท่ากัน ทำให้การเปรียบเทียบใช้ 0 (n) แทน 0 (log n) ซึ่งจะทำให้การค้นหาใกล้เคียงกับลิเนียร์เซิรช

หมายเหตุ : worst – case ของไบนารีเซิรชทรีอาจจะขจัดออกไปได้โดยการใช้ ไบนารีเซิรชทรีที่มีความสมดุลย์สูง (hight – balanced binary search tree) ซึ่งจะต้องทำให้เกิดตามสมดุลย์ใหม่ทุกครั้งที่มีการแทรกหรือลบข้อมูล แต่อัลกอริทึมของการทำงานดังกล่าวมีความยุ่งยากจึงไม่ขอกล่าวในที่นี้

4.2.7 การค้นหาไฟล์ , การค้นหาพ้อยน์เตอร์

ไฟล์ F ประกอบด้วยเรคคอร์ด R_1, R_2, ...R_nเก็บอยู่ในหน่วยความจำ การค้นหา F เป็นการค้นหาตำแหน่ง LOC ของหน่วยความจำที่เก็บเรคคอร์ดที่มีค่าคีย์ฟิลด์ K ตรงกับค่าที่ต้องการ แนวทางการค้นหาทำได้โดยใช้อาเรย์ของพ้อยน์เตอร์ที่มีการจัดเรียงข้อมูลเพิ่มขึ้นมา 1 อาเรย์สำหรับเป็นอาเรย์ช่วยซึ่งจะทำให้ไบนารีเซิรชค้นหาตำแหน่ง LOC ได้เร็วขึ้น ในกรณีที่ต้องการเพิ่มหรือลบเรคคอร์ดให้มีความรวดเร็วขึ้น จะใช้ไบนารีเซิรชทรีช่วยแทนอาเรย์ช่วย ในกรณีอื่นๆ การค้นหาของไฟล์ F เป็นเช่นเดียวกับการค้นหาข้อมูลใน S ซึ่งได้พิจารณาไปแล้ว

4.2.8 แฮชชิ่ง (HASHING)

เวลาในการค้นหาข้อมูลของอัลกอริทึมต่างๆจะขึ้นอยู่กับจำนวนอิลิเมนท์ที่เก็บอยู่ ในหัวข้อนี้จะพิจารณาถึงเทคนิคการค้นหาข้อมูลที่เรียกว่า แฮชชิ่ง หรือ แฮดเครสชิ่งซึ่งไม่ขึ้นอยู่กับจำนวนอิลิเมนท์

การทำงานต่างๆที่ใช้กับการทำงานของแฮชชิ่ง จะเป็นระบบการจัดการกับไฟล์ โดยสมมติไฟล์ F ที่มี n เรคคอร์ด แต่ละเรคคอร์ดมีเซต K ของคีย์ซึ่งเป็นเอกลักษณ์ของเลขเรคคอร์ด และสมมติ F เก็บอยู่ในหน่วยความจำในลักษระของตาราง T ที่ใช้หน่วยความจำ m ตำแหน่ง และ L เป็นเซทของตำแหน่งหน่วยความจำใน T เพื่อความสะดวกเรากำหนดคีย์ใน K และตำแหน่งใน L เป็นเลขจำนวนเต็ม (วิธีการสามารถนำไปใช้กับเลขจำนวนเต็มฐานสอง หรือคีย์ที่เป็นสตริง เช่น ชื่อ ได้ เนื่องจากเราสามารถแทนสตริงด้วยเลขจำนวนจริงได้)

4.2.9 การเชื่อมต่อ (chaining)

การเชื่อมต่อเป็นการเก็บ 2 ตารางลงในหน่วยความจำ ใช้ตาราง T เก็บเรคคอร์ดของไฟล์ F เช่นที่ผ่านมา แต่ T ในส่วนนี้จะมีฟิลด์ LINK เพิ่มขึ้นมา ซึ่งฟิลด์นี้จะใช้เพื่อเชื่อมต่อเรคคอร์ดใน T ที่มีตำแหน่งแฮช H เหมือนกัน เพื่อสร้างเป็นลิงค์ลิสท์ อีกตารางหนึ่ง คือ ตารางตำแหน่งแฮชชื่อ LIST ทำหน้าที่เก็บพ้อยน์เตอร์ที่ชี้ลิงค์ลิสท์ใน T

เมื่อเรคคอร์ดใหม่ R ที่มีคีย์ k ถูกเพิ่มลงในไฟล์ F เราใส่ R ลงในตำแหน่งแรกที่ว่างของตาราง T และเพิ่ม R ให้กับลิงค์ลิสท์ด้วยค่าพ้อยน์เตอร์ LIST [H (k)] ถ้าลิงค์ลิสท์ของ

เรคคอร์ดไม่มีการเรียงลำดับจะใส่ R ลงในตำแหน่งเริ่มต้นของลิงค์ลิสท์ การค้นหาเรคคอร์ดหรือการลบเรคคอร์ดจะเหมือนกับการค้นหาหรือลบโหนดออกจากลิงค์ลิสท์

4.3 สแตก คิว รีเคอร์ชั่น (STACK QUEUE RECURSION)

4.3.1 สแตก (STACK)

สแตกเป็นลิสท์ของอิลิเมนท์ที่มีการเพิ่มหรือลบข้อมูลออก ได้ทางเดียวจากด้านบน (TOP)ของสแตก ข้อมูลที่นำออกจากสแตกจะเรียงตามลำดับแบบกลับทางกับลำดับของการเพิ่มเข้ามา

คำที่ใช้กับสแตก 2 คำ คือ

(1) Pushใช้สำหรับการเพิ่มอิลิเมนท์ลงในสแตก

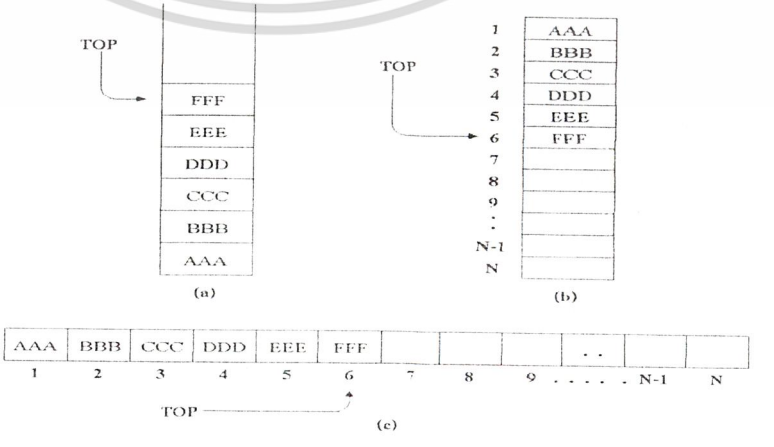
(2) Pop ใช้สำหรับการดึงอิลิเมนท์ออกจากสแตก

คำ 2 คำนี้จะใช้กับสแตกเท่านั้นไม่ใช้กับโครงสร้างข้อมูลชนิดอื่น

การทำงานของสแตก

สแตกมีการใช้สำหรับแสดงลำดับการประมวลผลข้อมูลเมื่อต้องการข้ามขั้นตอนบางขั้นตอนไปกระทำขั้นตอนอื่นให้จบก่อนแล้วจึงกลับมาทำขั้นตอนเดิมต่อดังตัวอย่างต่อไปนี้

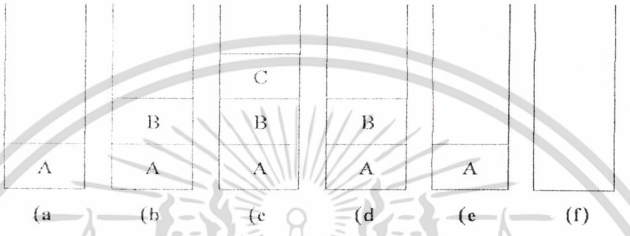
สมมติ เมื่อกำลังประมวลผลข้อมูล A อยู่เราต้องการข้ามไปประมวลผลข้อมูล B ให้สำเร็จก่อนแล้วนำข้อมูลมาใช้กับงาน A เราจะต้องเก็บข้อมูลของงาน A ลงในสแตกก่อนแล้วจึงข้ามไปทำงาน B ในขณะที่ทำงาน B อยู่เรามีความจำเป็นต้องการใช้การทำงานของ C ดังนั้นเราจะเก็บข้อมูลบางตัวของ B ลงในสแตกเหนือ A ดังแสดงในรูป แล้วเริ่มทำงาน C ถ้าหากต้องไปทำงาน D ในขณะที่ C ยังไม่สมบูรณ์ ก็ต้องเก็บข้อมูลของ C ลงสแตก เช่นเดียวกัน ดังแสดงในรูปที่ 4.7 แล้วจึงไปทำงาน D



รูปที่ 4.6 ไดอะแกรมของสแตก

เมื่องาน D ถูกทำเสร็จ ก็จะกลับมาทำงาน C ซึ่งอยู่บนสุดโดยดึงข้อมูลออกมาจากสแตก ได้ก่อนทำให้สแตกว่างลง 1 ช่อง ดังแสดงในรูป หลังจากงาน C สิ้นสุด ก็ดึงข้อมูลของ B จากสแตกออกมา ทำให้สแตกว่างลงอีกดังรูปและเมื่องาน B สิ้นสุด ข้อมูล A ที่อยู่ในสแตกจะถูกดึงออกมาจะทำให้สแตกว่างดังรูป และงาน A ถูกกระทำต่อไป

สังเกต สแตกจะมีการลำดับการทำงานอัตโนมัติในแต่ละขั้นตอนของการทำงานซึ่งตัวอย่างที่กล่าวมาจะเทียบได้กับการทำงานของโปรแกรมคอมพิวเตอร์ ซึ่งมี A เป็นโปรแกรมหลัก B, C, และ D เป็นโปรแกรมย่อยที่ถูกเรียกตามลำดับ

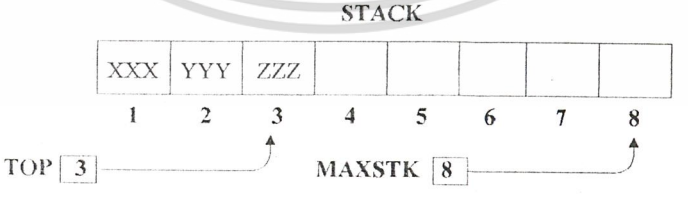


รูปที่ 4.7 โครงสร้างสแตก

4.3.2 การแสดงสแตกในรูปอาเรย์ (ARRAYS REPRESENTATION OF STACKS)

สแตกในคอมพิวเตอร์สามารถจะเก็บได้หลายรูปแบบ ปกติจะใช้จัดเก็บแบบลิสค์ทางเดียวหรืออาเรย์ หากไม่มีการกำหนดเป็นอย่างอื่น สแตกจะเก็บอยู่ในลักษณะของอาเรย์ STACK โดยมีตัวแปร TOP เก็บตำแหน่งบนสุดของสแตกและตัวแปร MAXSTK เป็นค่าสูงสุดที่จะเก็บลงในสแตกได้ ค่า TOP = 0 หรือ TOP = NULL หมายถึงสแตกว่าง

รูป 4.8 แสดงอาเรย์ของสแตก (อาเรย์อยู่ในแนวนอน) เนื่องจาก TOP = 3 สแตกมี 3 อิลิเมนท์ คือ XXX, YYY และ ZZZ ค่า MAXSTK = 8 จึงมีช่องว่างสำหรับเก็บข้อมูลอีก 5 รายการ



รูปที่ 4.8 สแตกในรูปอาเรย์

4.3.3 Minimizing Overflow

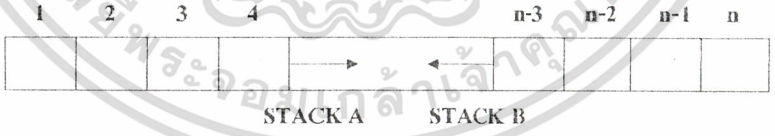
ความแตกต่างระหว่างอันเดอร์โฟลว์ และ โอเวอร์โฟลว์ ในการทำงานกับสแตก อันเดอร์โฟลว์ขึ้นอยู่กับอัลกอริทึมและข้อมูลที่กำหนดให้ ซึ่งผู้เขียนโปรแกรมไม่สามารถควบคุม

ได้ โอเวอร์โฟลว์ขึ้นอยู่กับผู้เขียนโปรแกรมว่าต้องการพื้นที่สำหรับสแตกเท่าใดหากพื้นที่ไม่พอจะเกิดโอเวอร์โฟลว์ได้

ทั่วไปแล้วจำนวนของอิลิเมนท์ในสแตกจะมากขึ้นหรือน้อยลงอยู่เสมอ ขึ้นอยู่กับข้อมูลที่ดึงออกหรือเพิ่มเข้าในสแตกหากมีการจองพื้นที่ของสแตกมาก จะทำให้โอกาสเกิดโอเวอร์โฟลว์ลดลง แต่ทำให้เปลืองพื้นที่ของหน่วยความจำ หากจองพื้นที่ของสแตกน้อยเกินไปจะทำให้มีโอกาสเกิดโอเวอร์โฟลว์เพิ่มขึ้นและต้องเสียเวลาในการแก้ปัญหาโอเวอร์โฟลว์ที่เกิดขึ้น มีเทคนิคหลายอย่างที่พัฒนาขึ้นมาใช้กับข้อมูลของสแตก ซึ่งการจองพื้นที่ของสแตกมากกว่า 1 สแตก อาจให้ประสิทธิภาพในการทำงานที่ดีกว่า เทคนิคเหล่านั้นอยู่นอกเหนือขอบเขตในส่วนนี้จะกล่าวดังตัวอย่างดังนี้

สมมุติอัลกอริทึมต้องการใช้ 2 สแตก คือ A และ B เราสามารถกำหนดอาเรย์ STACK A ที่มีจำนวน N_1 อิลิเมนท์ สำหรับสแตก A และอาเรย์ STACK B ที่มีจำนวน N_2 อิลิเมนท์สำหรับสแตก B เมื่อสแตก A มีมากกว่า N_1 อิลิเมนท์หรือสแตก B มีมากกว่า N_2 อิลิเมนท์จะเกิดโอเวอร์โฟลว์

ถ้าเรากำหนดอาเรย์เดียวคือ STACK ที่มีจำนวน $n = N_1 + N_2$ อิลิเมนท์สำหรับสแตก A และ B ดังแสดงในรูป 4.9 โดยกำหนด STACK [1] เป็นตำแหน่งเริ่มต้นของสแตก A และเพิ่มขึ้นไปทางขวา และกำหนด STACK [N] เป็นตำแหน่งเริ่มต้นของสแตก B และเพิ่มขึ้นไปทางซ้าย ในกรณีนี้จะเกิดโอเวอร์โฟลว์ เมื่อ A และ B รวมกันแล้วมีอิลิเมนท์มากกว่า $n = N_1 + N_2$ ซึ่งเทคนิคนี้จะทำให้อัตราการเกิดโอเวอร์โฟลว์ลดลงโดยไม่จำเป็นต้องเพิ่มจำนวนพื้นที่ของสแตกทั้งสอง การใช้งานในกรณีนี้การทำงานของ PUSH และ POP ต้องมีการดัดแปลง



รูปที่ 4.9 Minimizing Overflow

4.3.4 นิพจน์ทางคณิตศาสตร์ (ARITHEMATIC EXPRESSIONS)

เมื่อ Q เป็นนิพจน์ทางคณิตศาสตร์ที่มีตัวคงที่และตัวกระทำ ในหัวข้อนี้จะเป็นการแสดงอัลกอริทึมสำหรับหาค่าของ Q โดยใช้การกระทำอยู่ข้างหลัง (reverse Polish pustfix notation)ซึ่งเราจะสามารถแสดงให้เห็นถึงคุณสมบัติที่สำคัญของสแตกในอัลกอริทึมนี้

การทำงานในลักษณะของเลขฐานสองในนิพจน์ Q จะมีหลายระดับแตกต่างกัน เรากำหนดให้ลำดับการทำงานของเลขฐานสอง 5 แบบ มีลำดับความสำคัญดังนี้

ความสำคัญสูงสุด ยกกำลัง ($\uparrow$)

ความสำคัญรองลงมา	การคูณ (*) และการหาร (/)
ความสำคัญต่ำสุด	การบวก (+) และการลบ (-)

(การยกกำลังเราใช้สัญลักษณ์ของ BASIC) เพื่อให้ง่ายเราสมมติว่าทุกค่าใน Q เป็นเลขไม่มีเครื่องหมาย (ไม่มีค่าลบ) และไม่มีวงเล็บแยกนิพจน์ การทำงานของระดับเดียวกันจะทำจากซ้ายไปขวา(รูปแบบนี้ไม่เป็นมาตรฐาน เนื่องจากโปรแกรมคอมพิวเตอร์บางภาษายกกำลังจะทำจากขวาไปซ้าย)

-Polish Notation

ตัวกระทำทางคณิตศาสตร์ปกติจะอยู่ระหว่างตัวที่ถูกกระทำ 2 ตัว เช่น

A + B C – D E * F G / H

ที่เรียกว่าอินฟิกซ์ (infix notation) ซึ่งการใช้เครื่องหมายดังนี้จะแยกความแตกต่างของ

(A + B) * C และ A + (B * C)

โดยใช้วงเล็บหรือการกระทำตามลำดับดังที่อธิบายในตอนต้น ลำดับของการกระทำและการถูกกระทำในนิพจน์คณิตศาสตร์ปกติ จะไม่อยู่ในลำดับของการกระทำตามความสำคัญ Polish notation เกิดขึ้นหลังจากที่นักคณิตศาสตร์ชาวโปแลนด์ชื่อ Jan Lukasiewicz ใช้การแสดงเครื่องหมายการกระทำก่อนตัวที่ถูกกระทำ 2 ตัว เช่น

+ AB -CD *EF /GH

นิพจน์อินฟิกซ์ต่อไปนี้แปลงเป็น Polish notation ได้โดยใช้วงเล็บใหญ่ [ ] แยกตัวประกอบแต่ละส่วน

(A + B) * C = [+ AB] * C = * + ABC

A + (B * C) = A + [*BC] = + A*BC

(A + B) / (C - D) = [+ AB] / [- CD] = / + AB - CD

คุณลักษณะพื้นฐานของ Polish notation คือ การจัดเรียงการทำงานตามตำแหน่งของตัวกระทำและตัวถูกกระทำในนิพจน์ ซึ่งไม่จำเป็นต้องใช้นิพจน์ทางคณิตศาสตร์

Reverse Polish notation เป็นการแสดงนิพจน์ในลักษณะเดียวกัน แต่กลับเครื่องหมายการกระทำไว้หลังตัวถูกกระทำ เช่น

AB + CD - EF * GH /

ซึ่งการแสดงนิพจน์ในลักษระนี้ไม่มีกรใช้วงเล็บเช่นกัน การแสดงนิพจน์ในลักษณะนี้เรียกว่าโพสท์ฟิกซ์หรือซัฟฟิกซ์ (Postfix or Suffix) เมื่อพรีฟิกซ์เป็นเทอมที่ใช้สำหรับ Polish notation ที่ได้กล่าวไปแล้ว

ปกติคอมพิวเตอร์จะหาค่านิพจน์คณิตศาสตร์ที่เขียนในลักษณะของอินฟิกซ์ก่อนแล้วจึงทำขั้นที่สอง หาค่าของนิพจน์โพสท์ฟิกซ์ในแต่ละขั้นสแตกจะเป็นเครื่องมือที่สำคัญ

Symbol Scanned		STACK
(1)	5	5
(2)	6	5,6
(3)	2	5,6,2
(4)	+	5,8
(5)	*	40
(6)	12	40,12
(7)	4	40,12,4
(8)	/	40,3
(9)	–	37
(10)	)	

รูปที่ 4.10 Polish notation

-การเปลี่ยนนิพจน์อินฟิกซ์เป็นโพสฟิกซ์

เมื่อ Q เป็นนิพจน์คณิตศาสตร์ที่เขียนอยู่ในรูปอินฟิกซ์ ซึ่ง Q อาจมีวงเล็บเปิดและปิดประกอบอยู่ สมติว่าตัวกระทำใน Q ประกอบด้วย การยกกำลัง (↑), การคูณ (*), การหาร (/), การบวก (+) และการลบ (-) และลำดับความสำคัญมี 3 ลำดับ และการทำงานของตัวกระทำในลำดับเดียวกัน จะทำจากซ้ายไปขวา ถ้าไม่มีวงเล็บเป็นตัวแยก (กรณีนี้ไม่เป็นมาตรฐานเนื่องจากนิพจน์ทั่วไปอาจประกอบด้วย ตัวกระทำอื่นๆ และโปรแกรมคอมพิวเตอร์บางภาษา อาจมีการทำงานของการยกกำลังจากขวาไปซ้าย)

4.3.5 ควิกซอร์ทและการประยุกต์ใช้สแตก (QUICKSORT AND APPLICATION OF STACK)

กำหนด A เป็นลิสท์ของข้อมูลจำนวน n รายการ การเรียงลำดับข้อมูลใน A เป็นการจัดลำดับตำแหน่งของอิลิเมนท์ตามที่ต้องการซึ่งการจัดเรียงข้อมูลมีหลายวิธี ในหัวข้อนี้จะกล่าวถึงเฉพาะอัลกอริทึมการจัดเรียงแบบควิกซอร์ทเท่านั้น

ควิกซอร์ทเป็นอัลกอริทึมที่ทำการเรียงกลุ่มของข้อมูล โดยแบ่งข้อมูลออกเป็น 2 กลุ่มย่อย แล้วทำการจัดเรียง ซึ่งขั้นตอนการจัดแบ่งออกเป็นกลุ่มย่อยนั้น จะอธิบายโดยตัวอย่างเฉพาะดังนี้

สมมติ A ประกอบด้วยตัวเลข 12 จำนวน

(44) 33 11 55 77 90 40 60 99 22 88 (66)

ขั้นตอนของอัลกอริทึมควิกซอร์ทในการแบ่งเป็นกลุ่ม เริ่มจากการหาตำแหน่งท้ายสุดของเลขตัวหนึ่ง ในที่นี้ใช้ 44 เป็นตัวแรก ซึ่งการทำงานดังนี้ เริ่มจากการเปรียบเทียบกับ ค่าสุดท้าย

คือ 66 ทำงานจากขวาไปซ้าย ทีละตัว โดยการเปรียบเทียบจะหยุดเมื่อพบเลขตัวแรกที่ต่ำกว่า 44 ในที่นี้คือ 22 จะทำการสลับกันระหว่าง 44 กับ 22 ซึ่งจะได้ลิสท์เป็น

(22) 33 11 55 77 90 40 60 99 (44) 88 66

(สังเกตว่า 88 และ 66 ทางขวามือมากกว่า 44) เริ่มจากตำแหน่งของ 22 แล้วทำการเปรียบเทียบในทิศทางตรงกันข้าม คือ จากซ้ายไปขวา เปรียบเทียบ 44 กับ แต่ละค่าแล้วหยุดเมื่อพบจำนวนที่มากกว่า 44 ในที่นี้คือ 55 จะสลับตำแหน่งกับ 44 ได้ลิสท์

22 33 11 (44) 77 90 40 60 99 (55) 88 66

(ค่า 22, 33 และ 11 ที่อยู่ทางซ้ายมีค่าน้อยกว่า 44) ดราวนี้เริ่มจากตำแหน่ง 55 เริ่มทำการเปรียบจากทิศทางเริ่มต้นคือ ขวาไปซ้าย จนกว่าจะพบเลขจำนวนแรกที่ต่ำกว่า 44 ในที่นี้คือ 40 สลับตำแหน่ง 44 กับ 40 จะได้ลิสท์

22 33 11 (40) 77 90 (44) 60 99 55 88 66

(ค่าที่อยู่ทางขวามือของ 44 มากกว่า 44) เริ่มจากตำแหน่งของ 40 ทำการเปรียบเทียบจากซ้ายไปขวา เลขที่มีค่ามากกว่า 44 คือ 77 สลับตำแหน่งระหว่าง 44 และ 77 จะได้ลิสท์

22 33 11 40 (44) 90 (77) 60 99 66 55 88 66

(ค่าที่อยู่ทางซ้ายมือของ 44 มีค่าน้อยกว่า 44) เริ่มทำการเปรียบเทียบจากตำแหน่ง 77 จากขวาไปซ้าย เพื่อหาจำนวนที่น้อยกว่า 44 ซึ่งไม่พบค่าที่น้อยกว่า 44 หมายถึงว่า ตอนนี้ค่าทุกตัวได้ถูกเปรียบกับ 44 แล้ว และค่าที่มากกว่า 44 จะอยู่ทางขวามือ ส่วนค่าที่น้อยกว่า 44 จะอยู่ซ้ายมือ ดังนี้

22 33 11 40 (44) 90 77 60 99 55 88 66

ลิสท์ย่อยกลุ่มแรก ลิสท์ย่อยกลุ่มที่สอง

ค่า 44 อยู่ในตำแหน่งที่ถูกต้องการเรียงต่อไปจะทำกับข้อมูลในแต่ละกลุ่ม การกระทำขั้นตอนซ้ำในแต่ละกลุ่มย่อจะกระทำเมื่อมีข้อมูลในกลุ่มย่อยตั้งแต่ 2 อิลิเมนท์ขึ้นไป เนื่องจากเราจะทำได้ครั้งละกลุ่มเท่านั้น จึงจำเป็นต้องมีการเก็บข้อความของกลุ่มย่อยบางอย่างไว้ในสแตก 2 สแตก คือ LOWER และ UPPER ตำแหน่งแรกและตำแหน่งสุดท้ายของแต่ละกลุ่มเรียก "boundary value" จะถูกเก็บลงใน LOWER และ UPPER ตามลำดับ และการจัดเรียงจะกระทำกับกลุ่มข้อมูลย่อยที่ถูกดึงออกมาจากสแตกเท่านั้น

4.3.6 รีเคอร์ชั่น (RECURSION)

รีเคอร์ชั่นเป็นวิธีการที่สำคัญในวิทยาการคอมพิวเตอร์ซึ่งอัลกอริทึมหลายอย่างสามารถใช้วิธีการของ รีเคอร์ชั่นอธิบายได้ดี ในหัวข้อนี้จะเป็นการแสดงถึงประโยชน์ของรีเคอร์ชั่น

สมมติ P เป็นโปรซีเยอร์ที่มีสเตทเมนท์ Call ที่เรียกตัวเองหรือเรียกโปรซีเยอร์อื่นๆ ซึ่งอาจมีการส่งผลลัพธ์กลับมาที่โปรซีเยอร์ P ซึ่งเป็นโปรซีเยอร์หลัก เราเรียกโปรซีเยอร์ P ว่า รีเคอร์ซีฟโปรซีเยอร์ (Recursive procedure) เมื่อมีคุณสมบัติดังนี้

1. จะต้องมีกฎเกณฑ์บางอย่างซึ่งเรียกว่า Base criteria สำหรับโปรซีเยอร์ที่ไม่ทำให้มีการเรียกตัวเอง

2. ในแต่ละครั้งที่โปรซีเยอร์เรียกตัวเอง (ทางตรงหรือทางอ้อม) จะต้องเข้าใกล้กับกฎเกณฑ์ของ base criteria

รีเคอร์ซีฟโปรซีเยอร์ที่มีคุณสมบัติทั้งสองนี้เรียกว่า เป็นการกำหนดโปรซีเยอร์ที่ดี (well - defined) ในทำนองเดียวกัน ฟังก์ชั่นที่มีการเรียกตัวเองจะเรียกว่ารีเคอร์ซีฟดีไฟน์ (recursively defined) มีคุณสมบัติที่ป้องกันการวนรอบในตัวเองดังนี้

1. มีค่าอากิวเมนท์บางตัวเรียกว่า base values สำหรับฟังก์ชั่นเพื่อไม่ให้ทำการเรียกตัวเอง

2. ในแต่ละครั้งที่ฟังก์ชั่นเรียกตัวเอง ค่าอากิวเมนท์ของฟังก์ชั่นต้องเข้าใกล้กับของ base values

-อันดับไฟโบแนชชี่ (Fibanacci sequense)

อันดับของไฟโบไฟโบแนชชี่ (เขียนกำกับโดย $F_0, F_1, F_2, \ldots\ldots$) จะเป็นดังนี้

0, 1, 2, 3, 4, 5, 8, 13, 21, 34, 55, ...

ค่าของ $F_0 = 0$ และ $F_2 = 1$ และค่าของเทอมต่อๆ ไป จะเป็นผลรวมของ 2 เทอมที่อยู่ก่อนหน้าเช่น ค่าของ 2 เทอมของอันดับถัดไปจะเป็น

$34 + 55 = 89$ และ $55 + 89 = 144$

นิยามของการสร้างฟังก์ของอันดับจะเป็นดังนี้

(1) if $n = 0$ or $n = 1$, then $F_n = n$

(2) if $n > 2$ then $F_n = F_{n-2} + F_{n-1}$

นิยามนี้เป็นตัวอย่างการใช้งานรีเคอร์ซีฟอีกแบบหนึ่ง ซึ่งจะมีการเรียกตัวเอง เมื่อมีการใช้ F_{n-2} และ F_{n-1} และโดย (1) ค่า base value คือ 0 และ 1 และ (2) ค่าของ F_n กำหนดอยู่ในเทอมของค่าที่น้อยกว่า n ที่สูงกว่า base value

อัลกอริทึม ดีไวด์ และคองเคอร์ (Devide – and – Conquer Algolithms)

เมื่อเซต S ของข้อมูลที่อยู่ในการแก้ปัญหา P กำหนดให้ A เป็นอัลกอริทึม ซึ่งแยก S ออกเป็นส่วนย่อยๆ ดังนั้น การแก้ปัญหา P เกี่ยวกับค่าของ S จะกลายเป็นการแก้ปัญหา P สำหรับข้อมูลกลุ่มย่อยกลุ่มหนึ่งหรือมากกว่าแล้ว อัลกอริทึม A จะเรียกว่าเป็นอัลกอริทึม ดีไวด์ และคองเคอร์

อัลกอริทึม ดีไวด์และคองเคอร์ A อาจมองเป็นโปรซีเยอร์แบบรีเคอร์ซีฟก็ได้ โดยอัลกอริทึม A จะมีการนเรียกตัวเอง เมื่อมีการแบ่งข้อมูลออกเป็นกลุ่มย่อย ซึ่งค่าที่เป็น base criteria ของอัลกอริทึมคือกลุ่มของข้อมูลย่อยมีข้อมูลเพียง 1 อิลิเมนท์ เช่น การจัดเรียงข้อมูล หากมีข้อมูล 1 อิลิเมนท์ จะมีการจัดเรียงข้อมูลอยู่แล้ว และสำหรับการค้นหาข้อมูลในกลุ่มของข้อมูลที่มี 1 อิลิเมนท์ จะมีการเปรียบเทียบเพียงครั้งเดียว

-ฟังก์ชั่นแอคเคอร์มานน์ (Ackermanm Function)

ฟังก์ชั่นแอคเคอร์มานน์เป็นฟังก์ชั่นที่มี 2 อากิวเมนท์ แต่ละอากิวเมนท์เป็นเลขจำนวนเต็มบวก : 0, 1, 2, ... ซึ่งฟังก์ชั่นกำหนดได้ดังนี้

(1) if $m = 0$ then $A(m,n = n + 1)$

(2) if $m \neq 0$ but $n = 0$ then $A(m,n) = A(m - 1, 1)$

(3) if $m \neq 0$ but $n \neq 0$ then $A(m,n) = A(m - 1, A(m, n - 1))$

เป็นนิยามที่ใช้เรียกตัวเองใน (2) และ (3) สังเกต $A(m,n)$ จะมีค่าออกมาเมื่อ $m = 0$ เท่านั้น base criteria จะมีค่าเป็นคู่ดังนี้

$(0, 0), (0, 1), (0, 2), (0, 3), \ldots\ldots\ldots, (0, n), \ldots\ldots\ldots$

โดยทั่วไปแล้วฟังก์ชั่นแอคเคอร์มานน์มีความซับซ้อนมาก แต่มีความสำคัญในด้านของคณิตศาสตร์ลอจิก ซึ่งในส่วนนี้เป็นการแสดงให้เห็นถึงตัวอย่างของฟังก์ชั่นแบบรีเคอร์ซีฟเท่านั้น

บทที่ 5

โครงสร้างโปรแกรม LADDER SOFTWARE SUPPORT

5.1 การออกแบบทางด้านซอร์ฟแวร์

การออกแบบทางด้านซอร์ฟแวร์ หรือการออกแบบโปรแกรมระบบจัดการข้อตกลงในการติดต่อสื่อสารข้อมูล (PROTOCOLS) ข้อมูลที่ได้รับเข้ามาทางพอร์ทอนุกรมจะประกอบด้วยบล็อกคำสั่งแต่ละคำสั่ง เช่น การอ่านหรือการเขียนข้อมูลเป็นต้น ข้อมูลจะถูกเปลี่ยนจากอนุกรมให้เป็นแบบขนาน มาเก็บไว้ยังพื้นที่หน่วยความจำที่เตรียมไว้ (BUFFER) สำหรับบล็อกคำสั่งมีขนาดไม่เกิน 256 ไบท์ทันทีที่มีการส่งบล็อกคำสั่งเข้ามา PLC จะถูกอินเตอร์รัพให้มาตรวจสอบข้อมูลทีละ 1 ไบท์ โดย 3 ไบท์แรกจะเป็นหมายเลขประจำเครื่อง ถ้าตรงกันก็จะรับข้อมูลเข้ามาเก็บในบัฟเฟอร์ต่อไปแต่ถ้าไม่ตรงกันบัฟเฟอร์จะถูกลบออก และออกจากโปรแกรมเพื่อรอรับข้อมูลต่อไป ซึ่งถ้าข้อมูลตรงกันตามข้อตกลงในการติดต่อสื่อสารโดยมีการตรวจสอบแล้ว PLC ทำการถอดรหัสคำสั่ง และประมวลผลตามบล็อกคำสั่ง จากนั้นจะเริ่มส่งบล็อกตอบสนองกลับไป เมื่อได้รับสัญญาณ CARRIER RETURN (CR) เป็นการบอกให้ทราบว่าจบบล็อกคำสั่ง

5.2 ข้อตกลงในการสื่อสารข้อมูลที่ทำการออกแบบ

ชุดของข้อมูลในการสื่อสารจะถูกเรียกว่าบล็อก บล็อกของข้อมูลจะถูกส่งจากเครื่องคอมพิวเตอร์ ไปในระบบการเชื่อมต่อ ซึ่งจะเรียกว่า บล็อกคำสั่ง (COMMAND BLOCK) และบล็อกของข้อมูลที่ถูกส่งจากระบบ การเชื่อมต่อไปสู่ HOST จะเรียกว่า บล็อกตอบสนอง (RESPONSE BLOCK) ในระบบการเชื่อมต่อสื่อสารแบบหลายจุด แต่ละบล็อก ไม่ว่าจะเป็นบล็อกคำสั่ง หรือ บล็อกตอบสนองก็ตาม จะเริ่มต้นด้วย อักขระ “@” ตามด้วยตำแหน่งเฉพาะ (UNIT NUMBER) ตามด้วยคำสั่ง (HEADER) ข้อมูล (DATA) และอักขระ “*” สิ้นสุดด้วยรหัสกำกับบล็อก (FRAME CHECK SEQUENCE CODE : FCS) และรหัสปิดท้ายบล็อกที่เป็นอักขระ [CR]

@	X	X	X	X	DATA	*	X	X	CR
	ตำแหน่ง		คำสั่ง				FCS		ปิดท้าย

รูปที่ 5.1 แสดงรูปแบบของบล็อกตอบสนอง

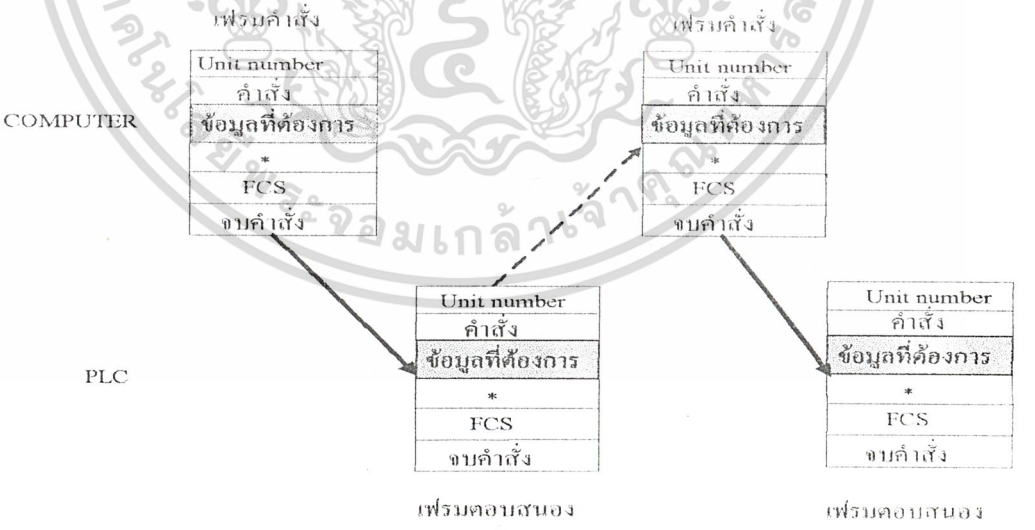
จำนวนอักขระในแต่ละบล็อกทั้งหมดจะต้องไม่เกิน 128 ตัว และช่วงของการคำนวณเพื่อหารหัสกำกับบล็อกจะอยู่ระหว่างอักขระเริ่มต้น (@)ไปจนถึงสิ้นสุด DATA ที่เป็น "*"

5.3 คำสั่งและคำตอบสนอง

ในขณะที่ทำการเชื่อมต่อเพื่อสื่อสารข้อมูล HOST สามารถที่จะทำการเฝ้ามองการดำเนินการและสามารถที่จะทำการควบคุมการทำงานของเครื่องควบคุม ถ้าอยู่ในสถานะการเฝ้ามองโฮสคอมพิวเตอร์จะต้องส่งคำสั่งไปถามตามชนิดของข้อมูลที่ต้องการ แต่ละเครื่องควบคุม หรือถ้าในสถานะการควบคุมก็สามารถที่จะส่งคำสั่งไปทำการเปลี่ยนแปลงค่าของข้อมูลที่อยู่ในหน่วยความจำโดยตรง เช่น ข้อมูลของอินพุท / เอาท์พุท เป็นต้น เวลาของการตอบสนองจะแปรไปขึ้นอยู่กับความเร็วในการส่งผ่านข้อมูล จำนวนของข้อมูล และเวลาในการสแกนของเครื่องควบคุม และ ถ้าเวลาในการสื่อสารมากขึ้นก็เป็นผลให้เวลาในการสแกนมากขึ้นตามไปด้วย ต่อไปจะเป็นคำสั่งต่างๆที่เป็นข้อกำหนด ที่ใช้ในการสื่อสาร

5.3.1 ลักษณะการรับ และส่งข้อมูล

การสื่อสารระหว่าง COMPUTER กับ PLC ทั้งส่ง และรับข้อมูล จะมีการส่ง เฟรม คำสั่งจาก COMPUTER ไปยัง PLC และPLC จะตอบกลับมา โดยในที่นี้อาจเป็นเฟรมของผลตอบสนองต่อคำสั่งหรือเป็นข้อมูลจาก PLC ส่งมาให้ COMPUTER ตามที่ ได้รับการร้องขอจากเฟรมคำสั่ง



รูปที่ 5.2 ลักษณะการรับส่งเฟรมคำสั่งและผลตอบสนอง

5.3.2 บล็อกคำสั่งประกาศ

บล็อกคำสั่งประกาศถูกส่งโดยโฮสไปเพื่อตรวจสอบว่ามีเครื่องควบคุมอยู่ในการเชื่อมต่อหรือในเครือข่ายหรือไม่

รูปแบบคำสั่ง

@	[Unit Number]	AT	*	[FCS]	[CR]

เช่น บล็อกคำสั่ง เป็น " @ 01AT*7E[CR] " หมายถึง โฮสส่งออกไปเพื่อ ประกาศถามถึงเครื่องควบคุมที่อยู่ในการเชื่อมต่อมีตำแหน่งเฉพาะที่ 01 หรือไม่ ถ้าเครื่อง ควบคุมมีจริงและได้รับคำประกาศ ดังกล่าวก็จะส่งคำตอบสนองออกไป ดังนี้

รูปแบบคำตอบสนอง

@	[Unit Number]	PC84	*	[FCS]	[CR]

เช่น บล็อกตอบสนองเป็น " @ 01PC84*74[CR] " หมายถึง เครื่องควบคุมตำแหน่งเฉพาะที่ 01 เท่านั้นที่ตอบสนองไป ซึ่งจะแสดงให้ HOST ทราบว่าเครื่องควบคุมที่ตำแหน่ง 01 ในระบบการเชื่อมต่อยังคงทำงานปกติ

รูปแบบคำสั่ง

@	[Unit Number]	RI	[ตำแหน่งเริ่มต้น]	[จำนวนข้อมูล]	*	[FCS]	[CR]

5.3.3 บล็อกคำสั่งอ่านพื้นที่ Input / Output / Internal Relay

เช่น ถ้าโฮสต้องการอ่านพื้นที่ของ I/ O หรือ Internal Relay ของเครื่องควบคุมที่วางอยู่ในระบบการเชื่อมต่อตำแหน่งเฉพาะที่ 02 ตำแหน่ง I/O ที่ 0010 มาจำนวน 5 ตำแหน่งก็สามารถจัดบล็อกได้ดังนี้ " @ 02RI00100005 "XX[CR] " (FCS XX : ASCII 2 digit)

รูปแบบคำตอบสนอง

@	[Unit Number]	RI	[ตำแหน่งเริ่มต้น]	[ข้อมูลXX(1)]	[ข้อมูลXX(..)]	[ข้อมูลXX(n)]	*	[FCS]	[CR]

เช่น ถ้าคำตอบสนองของเครื่องควบคุม PLC/PC ในตำแหน่ง ที่ 02 ตอบมาเป็น " @ 02RI0010234455566AB*XX[CR] " ก็หมายความว่า ตำแหน่งหน่วยความจำของ I/O หรือ Internal Relay ที่ 0010 มีข้อมูล 23 , 44 , 55 , 66 , AB [ฐาน 16] ตามลำดับ

5.3.4 บล็อกคำสั่งเขียนพื้นที่ Input / Output / Internal Relay

รูปแบบคำสั่ง

@	[Unit Number]	WI	[ตำแหน่งเริ่มต้น]	[ข้อมูลXX(1)]	[ข้อมูลXX(..)]	[ข้อมูลXX(n)]	*	[FCS]	[CR]

เช่น ต้องการเขียนข้อมูลลงในตำแหน่ง I/O หรือ Internal Relay ของเครื่องควบคุมที่ตำแหน่งเฉพาะ 03 ในตำแหน่งที่ 0010 ด้วยข้อมูล 12 , 34 , 56 , 78 , 9A จำนวน 5 ตำแหน่ง ก็สามารถจัดบล็อกคำสั่ง ได้ดังนี้ " @ 03WI0010123456789A*XX[CR] "

รูปแบบคำตอบสนอง

@	[Unit Number]	WI	[รหัสตอบสนอง]	*	[FCS]	[CR]

(XX : Response Code)

00 = Data Complete

08 = Data Error

และถ้าเรียบร้อยก็จะตอบออกมาเป็น " @ 03WI00*XX[CR] "

5.3.5 บล็อกคำสั่งอ่านค่าเป้าหมาย Timer

เพื่อสามารถให้โฮสรู้ค่าเป้าหมาย (Set Value) ที่ได้ตั้งไว้ที่ตัวเวลา

รูปแบบคำสั่ง

@	[Unit Number]	RT	[ตำแหน่งเริ่มต้น]	[จำนวน]	*	[FCS]	[CR]

เช่น ถ้าต้องการอ่านค่าเป้าหมายของตัวตั้งเวลาที่ 12 ของเครื่องควบคุมที่อยู่ในตำแหน่งเฉพาะที่04 สามารถจัดบล็อกคำสั่ง ได้ดังนี้ " @ 04RT00120001*XX[CR] "

รูปแบบคำตอบสนอง

@	[Unit Number]	RT	[ข้อมูลXX(..)]	[ข้อมูลXX(n)]	*	[FCS]	[CR]

และถ้าเครื่องควบคุมในตำแหน่งเฉพาะที่ 04 รับคำสั่งได้จากระบบเชื่อมต่อ ก็จะให้ค่าตอบสนองออกมาสมมติว่าค่าเป้าหมายของตัวตั้งเวลาตำแหน่งที่ 12 มีค่า #0150 และ Block Response เป็น " @ 04RT00120150*XX[CR] "

5.3.6 บล็อกคำสั่งอ่านค่าเป้าหมาย Counter

เพื่อสามารถให้โฮสรู้ค่าเป้าหมาย (Set Value) ที่ได้ตั้งไว้ที่ตัวตั้งนับ

รูปแบบคำสั่ง

@	[Unit Number]	RC	[ตำแหน่งเริ่มต้น]	[จำนวน]	*	[FCS]	[CR]

เช่น ถ้าต้องการอ่านค่าเป้าหมายของตัวนับที่ 10 ของเครื่องควบคุมที่อยู่ในตำแหน่งเฉพาะที่ 04 สามารถจัดบล็อกคำสั่ง ได้ดังนี้ " @ 04RT00100001*XX[CR] "

รูปแบบคำตอบสนอง

@	[Unit Number]	RC	[ตำแหน่งเริ่มต้น]	[ข้อมูลXX(..)]	[ข้อมูลXX(n)]	*	[FCS]	[CR]

และถ้าเครื่องควบคุม ในตำแหน่งเฉพาะที่ 04 รับคำสั่งได้จากระบบเชื่อมต่อก็จะให้คำตอบสนองออกมา สมมติว่าค่าเป้าหมายของตัวนับตำแหน่งที่ 10 มีค่า #8000 และ บล็อกตอบสนอง เป็น " @ 04RT00108000*XX[CR] "

5.3.7 บล็อกคำสั่งเขียนค่าเป้าหมาย Timer

เพื่อที่โฮสสามารถตั้งแก้ไขค่าเป้าหมายใหม่ของตัวตั้งเวลาแก่เครื่องควบคุมที่อยู่ในระบบเชื่อมต่อได้ตามต้องการ

รูปแบบคำสั่ง

@	[Unit Number]	WT	[ตำแหน่งเริ่มต้น]	[ข้อมูลXX(..)]	[ข้อมูลXX(n)]	*	[FCS]	[CR]

เช่น ถ้าต้องการตั้งค่าเป้าหมายแก่ตัวตั้งเวลาของเครื่องควบคุมที่วางในระบบเชื่อมต่อที่มีตำแหน่งเฉพาะที่ 05 และ ตัวตั้งเวลาตำแหน่งที่ 00 ด้วยค่าเวลา #0200 สามารถจัด บล็อกคำสั่งได้ดังนี้ " @ 05WT00000200*XX[CR] "

รูปแบบคำตอบสนอง

@	[Unit Number]	WI	[รหัสตอบสนองXX]	*	[FCS]	[CR]

(XX : Response Code)

00= Data Complete

08= Data Error

และถ้าเรียบร้อยก็จะตอบสนองออกมาเป็น " @ 05WT00*XX[CR] "

5.3.8 บล็อกคำสั่งเขียนค่าเป้าหมาย Counter

ลักษณะของข้อกำหนดจะเหมือนกับเขียนค่าเป้าหมายให้กับตัวตั้งเวลา ต่างกันที่ Header

รูปแบบคำสั่ง

@	[Unit Number]	WC	[ตำแหน่งเริ่มต้น]	[ข้อมูลXX(..)]	[ข้อมูลXX(n)]	*	[FCS]	[CR]

เช่น ถ้าต้องการตั้งค่าเป้าหมายแก่ตัวนับของเครื่องควบคุมที่วางในระบบเชื่อมต่อที่มีตำแหน่งเฉพาะที่ 05 และตัวตั้งเวลาตำแหน่งที่ 00 ด้วยค่านับ #0200 สามารถจัดบล็อกคำสั่งได้ดังนี้ " @ 05WC00000200*XX[CR] "

รูปแบบคำตอบสนอง

@	[Unit Number]	WC	[รหัสตอบสนองXX]	*	[FCS]	[CR]

(XX : Response Code)

00 = Data Complete

08 = Data Error

และถ้าเรียบร้อยก็จะตอบออกมาเป็น " @ 05WC00*XX[CR]"

5.3.9 บล็อกคำสั่งอ่านค่าปัจจุบัน Timer

เพื่อที่โฮสสามารถที่จะทำการเฝ้ามองความเป็นไปของค่าเวลาที่กำลังทำงานของตัวตั้งเวลาในเครื่องควบคุมตำแหน่งเฉพาะใดๆ ที่วางในระบบเชื่อมต่อได้

รูปแบบคำสั่ง

@	[Unit Number]	PT	[ตำแหน่งเริ่มต้น]	[จำนวน]	*	[FCS]	[CR]

เช่นถ้าต้องการอ่านค่าเป้าหมายของตัวตั้งเวลาที่ 12 และ 13 ของเครื่องควบคุมที่อยู่ในตำแหน่งเฉพาะที่ 04 สามารถจัดบล็อกคำสั่ง ได้ดังนี้ " @ 04PT00120001*XX[CR] "

รูปแบบคำตอบสนอง

@	[Unit Number]	PT	[ตำแหน่งเริ่มต้น]	[ข้อมูลXX(..)]	[ข้อมูลXX(n)]	*	[FCS]	[CR]

และถ้าเครื่องควบคุมในตำแหน่งเฉพาะที่ 04 รับคำสั่งได้จากระบบเชื่อมต่อ ก็จะให้คำตอบสนองออกมา สมมติว่าค่าเป้าหมายของตัวตั้งเวลาตำแหน่งที่ 12 มีค่า #0150 และ ตัวตั้งเวลาตำแหน่งที่ 13 มีค่า #0200 บล็อกตอบสนองเป็น "@ 04RT001201500200*XX[CR] "

รูปแบบคำสั่ง

@	[Unit Number]	PC	[ตำแหน่งเริ่มต้น]	[จำนวนXX]	*	[FCS]	[CR]

5.3.10 อ่านค่าปัจจุบัน Counter

และถ้าเครื่องควบคุมในตำแหน่งเฉพาะที่ 04 รับคำสั่งได้จากระบบเชื่อมต่อ ก็จะให้คำตอบสนองออกมา สมมติว่าค่าเป้าหมายของตัวนับตำแหน่งที่ 10 มีค่า #8000 และบล็อกตอบสนองเป็น " @ 04PT00108000*XX[CR] "

5.3.11 อ่านค่าจากพื้นที่โปรแกรม Up Load Program

เพื่อให้โฮสสามารถที่จะนำข้อมูลส่วนโปรแกรมบูลีนที่ผู้ใช้ได้ทำการโปรแกรมไว้ในหน่วยความจำนั้นขึ้นมาที่ โฮส เพื่อทำการเก็บรักษาไว้

รูปแบบคำสั่ง

@	[Unit Number]	UL	*	[FCS]	[CR]

เช่น ถ้าโฮสต้องการนำโปรแกรมส่วนบูลีนจาก เครื่องควบคุมที่มีตำแหน่งเฉพาะเป็น 01 มาจากจุดเริ่มโปรแกรมถึงคำสั่งสุดท้าย (END) มาสามารถจัดบล็อกคำสั่ง ได้เป็น " @ 01UL*72 [CR] " และเครื่องควบคุมที่ตำแหน่งเฉพาะ 01 จะให้คำตอบสนองออกมาเป็นชุดข้อมูลที่ละ 20 ไบท์ เริ่มที่ตำแหน่งในหน่วยความจำที่ 8000 และชุดต่อไปของข้อมูลจะมีตำแหน่งเริ่มต้นที่ สอดคล้องกับตำแหน่งที่ถูกถ่ายเทขึ้นมาด้วย หรือ จนกว่าจะพบคำสั่ง END ดังรูปแบบดังต่อไปนี้ ข้อสังเกตรูปแบบคำตอบสนองนั้นจะมีลักษณะเหมือนกับคำสั่งในการเขียนลงพื้นที่โปรแกรม ทั้งนี้ก็เพื่อที่จะให้สามารถนำชุดข้อมูลดังกล่าวนั้นเขียนลงไปใหม่ที่ส่วนโปรแกรมได้เลยโดยไม่ต้องเปลี่ยนแปลงรูปแบบ

รูปแบบคำตอบสนอง

@	[Unit Number]	UL	[ตำแหน่งเริ่มต้น8000]	[ข้อมูลXX(..)]	[ข้อมูลXX(20)]	*	[FCS]	[CR]

5.3.12 เขียนลงพื้นที่โปรแกรม Down Load Program

เพื่อให้โฮสสามารถทำการส่งผ่านข้อมูลส่วนโปรแกรมบูลีนเข้าไปในพื้นที่โปรแกรมของเครื่องควบคุมที่มีตำแหน่งเฉพาะนั้นๆ ได้

รูปแบบคำสั่ง

@	[Unit Number]	DL	[ตำแหน่งเริ่มต้น8000]	[ข้อมูลXX(..)]	[ข้อมูลXX(20)]	*	[FCS]	[CR]

ข้อมูลของโปรแกรมบูลีนที่ต้องการเขียนลงในหน่วยความจำส่วนโปรแกรมนั้น ก็จะต้องไม่มากกว่า 20 ไบท์ เช่นเดียวกัน และตำแหน่งที่ต้องการว่างลงในหน่วยความจำก็สามารถกำหนดได้ (ตำแหน่งในหน่วยความจำส่วนโปรแกรมบูลีนเริ่มที่ 8000)

รูปแบบคำตอบสนอง

@	[Unit Number]	DL	[รหัสตอบสนองXX]	*	[FCS]	[CR]

(XX : Response Code)

00 = Data Complete

08 = Data Error

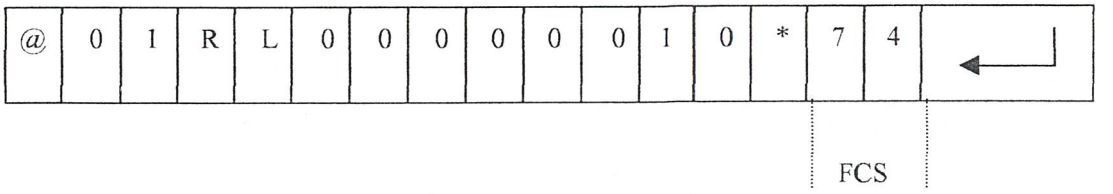
และถ้าเรียบร้อยก็จะตอบออกมาเป็น " @ 01DL00*XX[CR]"

5.3.13 การคำนวณหาค่า FCS (Frame Check Sequence)

FCS หรือ Frame Check Sequence เป็นสิ่งที่ใช้ในการตรวจสอบความผิดพลาดของการสื่อสาร เป็นการเปลี่ยนข้อมูล 8 บิต เป็น 2 ตัวอักษรข้อมูลASCII แล้วนำมาทำการ EXCLUSIVE - OR (XOR) โดยเริ่มจาก @ จนถึงตัวอักษรตัวสุดท้ายของ TEXT

อักขระ	รหัส(ASCII BINARY)	(HEX)	XOR (HEX)
@	0100 0000	[40]	
0	0011 0000	[30]	112 [70]
1	0011 0001	[31]	65 [41]
R	0101 0010	[52]	19 [13]
L	0100 1100	[4C]	95 [5F]
0	0011 0000	[30]	111 [6F]
0	0011 0000	[30]	95[5F]
0	0011 0000	[30]	111[6F]
0	0011 0000	[30]	95 [5F]
0	0011 0000	[30]	111 [6F]
0	0011 0000	[30]	95 [5F]
1	0011 0001	[31]	110 [6E]
0	0011 0000	[30]	94 [5E]
*	0010 1010	[2A]	116 [74]

จากการคำนวณที่ได้จะสร้างเป็นบล็อกได้ดังนี้



รูปที่ 5.3 แสดงรูปแบบของบล็อกและการคำนวณ FCS

5.4 ชุดคำสั่ง PLC

ชุดคำสั่งจะต้องทำการเก็บรหัสชุดคำสั่งและข้อมูลตำแหน่งตารางข้อมูลส่วนฟิลด์ CODE ใช้กำหนดรหัสของคำสั่งที่ผู้ใช้ทำการโปรแกรมขึ้น เพื่อให้เครื่องควบคุมทำงานตามฟังก์ชันการควบคุม คำสั่งที่ได้สร้างขึ้นมามีทั้งหมด 103 คำสั่งดังรายละเอียดต่อไปนี้

ตารางที่ 5.1 แสดงตารางคำสั่งที่ได้ออกแบบไว้

รหัส	คำสั่ง	ความหมาย
00	NOP	-
01	END	จบโปรแกรม
02	STC	เซตค่า Carry Flag
03	CTC	รีเซตค่า Carry Flag
04	ANDLD	รวมคำสั่งแบบ AND เมื่อคำสั่งแต่ละกลุ่มมากกว่า 1 คำสั่ง
05	ORLD	รวมคำสั่งแบบ OR เมื่อคำสั่งแต่ละกลุ่มมากกว่า 1 คำสั่ง
06	IL	ควบคุม Output ทั้งหมด
07	ILC	ยกเลิกการควบคุม Output ทั้งหมด
08	JMP	กระโดดข้ามไปที่คำสั่ง JME
09	JME	เป็นการสิ้นสุดการกระโดด
40	LD	นำค่าสภาวะจากตารางข้อมูลมาเป็นผลลัพธ์
41	AND	กระทำลอจิก AND กับสภาวะที่นำจากตารางข้อมูล
42	OR	กระทำลอจิก OR กับสภาวะที่นำจากตารางข้อมูล
43	OUT	นำผลลัพธ์ทางลอจิกและเก็บไว้ในตารางข้อมูล
44	LD NOT	นำค่าสภาวะตรงข้ามจากตารางข้อมูลมาเป็นผลลัพธ์
45	AND NOT	กระทำลอจิก AND กับสภาวะตรงข้ามที่นำจากตารางข้อมูล
46	OR NOT	กระทำลอจิก OR กับสภาวะตรงข้ามที่นำจากตารางข้อมูล

47	OUT NOT	นำผลลัพธ์ทางลอจิกและทำตรงข้ามเก็บไว้ในตารางข้อมูล
48	KEEP	เก็บค่าสภาวะลอจิก
49	DIFU	สร้างพัลส์สัญญาณขอบขาขึ้น
4A	DIFD	สร้างพัลส์สัญญาณขอบขาลง
4B	LD HR	นำค่าสภาวะตรงข้ามจากตารางข้อมูล HR มาเป็นผลลัพธ์
4C	AND HR	กระทำลอจิก AND กับสภาวะที่นำจากตารางข้อมูล HR
4D	OR HR	กระทำลอจิก OR กับสภาวะที่นำจากตารางข้อมูล HR
4E	OUT HR	นำผลลัพธ์ทางลอจิกและเก็บไว้ในตารางข้อมูล HR
4F	LD NOT HR	นำค่าสภาวะตรงข้ามจากตารางข้อมูล HR มาเป็นผลลัพธ์
50	AND NOT HR	กระทำลอจิก AND กับสภาวะตรงข้ามที่นำจากตารางข้อมูล HR
51	OR NOT HR	กระทำลอจิก OR กับสภาวะตรงข้ามที่นำจากตารางข้อมูล HR
52	OUT NOT HR	นำผลลัพธ์ทางลอจิกและทำตรงข้ามเก็บไว้ในตารางข้อมูล HR
53	KEEP HR	เก็บค่าสภาวะลอจิกลงตาราง HR
54	DIFU HR	สร้างพัลส์สัญญาณขอบขาขึ้นตาราง HR
55	DIFD HR	สร้างพัลส์สัญญาณขอบขาลงตาราง HR
56	LD LR	นำค่าสภาวะตรงข้ามจากตารางข้อมูล LR มาเป็นผลลัพธ์
57	AND LR	กระทำลอจิก AND กับสภาวะที่นำจากตารางข้อมูล LR
58	OR LR	กระทำลอจิก OR กับสภาวะที่นำจากตารางข้อมูล LR
59	OUT LR	นำผลลัพธ์ทางลอจิกและเก็บไว้ในตารางข้อมูล LR
5A	LD NOT LR	นำค่าสภาวะตรงข้ามจากตารางข้อมูล LR มาเป็นผลลัพธ์
5B	AND NOT LR	กระทำลอจิก AND กับสภาวะตรงข้ามที่นำจากตารางข้อมูล LR
5C	OR NOT LR	กระทำลอจิก OR กับสภาวะตรงข้ามที่นำจากตารางข้อมูล LR
5D	OUT NOT LR	นำผลลัพธ์ทางลอจิกและทำตรงข้ามเก็บไว้ในตารางข้อมูล LR
5E	KEEP LR	เก็บค่าสภาวะลอจิกลงตาราง LR
5F	DIFU LR	สร้างพัลส์สัญญาณขอบขาขึ้นตาราง LR
60	DIFD LR	สร้างพัลส์สัญญาณขอบขาลงตาราง LR
61	LD TIM	นำค่าสภาวะจากตัวตั้งเวลามาเป็นผลลัพธ์
62	LD CNT	นำค่าสภาวะจากตัวนับมาเป็นผลลัพธ์
63	LD NOT TIM	นำค่าสภาวะตรงข้ามจากตัวตั้งเวลามาเป็นผลลัพธ์

64	LD NOT CNT	นำค่าสภาวะตรงข้ามจากตัวนับมาเป็นผลลัพธ์
65	AND TIM	กระทำลอจิก AND กับสภาวะที่นำจากตัวตั้งเวลา
66	AND CNT	กระทำลอจิก AND กับสภาวะที่นำจากตัวนับ
67	AND NOT TIM	กระทำลอจิก AND กับสภาวะตรงข้ามที่นำจากตัวตั้งเวลา
68	AND NOT CNT	กระทำลอจิก AND กับสภาวะตรงข้ามที่นำจากตัวนับ
69	OR TIM	กระทำลอจิก OR กับสภาวะที่ที่นำจากตัวตั้งเวลา
6A	OR CNT	กระทำลอจิก OR กับสภาวะที่ตรงข้ามจากตัวนับ
6B	OR NOT TIM	กระทำลอจิก OR กับสภาวะตรงข้ามที่นำจากตัวตั้งเวลา
6C	OR NOT CNT	กระทำลอจิก OR กับสภาวะที่ตรงข้ามที่ตัวนับจากตัวนับ
6D	COM	กลับค่าสภาวะ ใน Word
6E	INC	เพิ่มค่าสภาวะใน Word
6F	DEC	ลดค่าสภาวะใน Word
70	ASL	เลื่อนข้อมูลทางซ้าย ภายใน Word จาก Bits(00) ไป Bits(15)
71	ASR	เลื่อนข้อมูลทางขวา ภายใน Word จาก Bits(15) ไป Bits(00)
72	ROL	วนรอบข้อมูลทางซ้าย ภายใน Word จาก Bits(00) ไป Bits(15)
73	ROR	วนรอบข้อมูลทางขวา ภายใน Word จาก Bits(15) ไป Bits(00)
74	DATA B	ข้อมูลอันดับที่ 2
75	DATA C	ข้อมูลอันดับที่ 3
76	DATA TIM	ข้อมูลTIMER
77	DATA CNT	ข้อมูลCOUNTER
78	DATA #	ข้อมูลแบบค่าคงที่
79	MSG	แสดงตัวอักษรหน้าจอ LCD
80	SFT	เลื่อนข้อมูลทางซ้ายระดับBits ระหว่าง Word
81	WSFT	เลื่อนข้อมูลระดับ Word
82	MOV	สำเนาข้อมูลแบบ Word
83	MOV NOT	สำเนาข้อมูลที่ตรงกันข้ามแบบ Word
84	CMP	เปรียบเทียบข้อมูลระดับระดับ Bits ของ 2 Word
85	BIN	แปลงข้อมูล BCD ไปเป็น Binary แบบ Word
86	BCD	แปลงข้อมูล Binary ไปเป็น BCD แบบ Word

87	RDI	อ่านข้อมูลจาก PLC Unit อื่นๆ
88	WRO	เขียนข้อมูลไป PLC Unit อื่นๆ
89	SLD	เลื่อนข้อมูลทางซ้ายระดับ Digits ระหว่าง Word
8A	SRD	เลื่อนข้อมูลทางขวาระดับ Digits ระหว่าง Word
8B	SFTR	เลื่อนข้อมูลระดับBits ระหว่าง Word ควบคุมทิศทางได้
8C	TKY	แปลงค่า KEY Binary เป็น BCD
8D	7SEG	แปลงค่า 7 Segment เป็น BCD ระดับ Digits ของ Word
8E	TIM	ตัวตั้งเวลา
8F	CNT	ตัวนับ
C0	ADD	บวกค่าระหว่าง Word ของข้อมูล BCD
C1	SUB	ลบค่าระหว่าง Word ของข้อมูล BCD
C2	MUL	คูณค่าระหว่าง Word ของข้อมูล BCD
C3	DIV	หารค่าระหว่าง Word ของข้อมูล BCD
C4	ANDW	การ AND ข้อมูลระหว่าง Word
C5	ORW	การ OR ข้อมูลระหว่าง Word
C6	XORW	การ XOR ข้อมูลระหว่าง Word
C7	XNRW	การ XOR NOT ข้อมูลระหว่าง Word
C8	MLPX	คำสั่งการแปลง1 Digit ต่อ 1 Bit ของ Word (4 to 16 Decoder)
C9	ADB	บวกค่าระหว่าง Word ของข้อมูลระดับ Bits
CA	SBB	ลบค่าระหว่าง Word ของข้อมูลระดับ Bits
CB	MLB	คูณค่าระหว่าง Word ของข้อมูลระดับ Bits
CC	DVB	หารค่าระหว่าง Word ของข้อมูลระดับ Bits
CD	MOVB	สำเนาข้อมูลแบบ Bits
CE	MOVD	สำเนาข้อมูลแบบ Digits
CF	ASC	แปลงค่า BCD เป็นรหัส ASCII
D0	SDEC	แปลงค่าระดับ Digits ของ Word เป็นค่า 7 Segment
D1	TXD	ส่งค่าข้อมูล ไป PLC อื่นๆ ในวงลูปเครือข่าย
D2	RXD	รับค่าข้อมูล ไป PLC อื่นๆ ในวงลูปเครือข่าย

เมื่อผู้ใช้โปรแกรมคำสั่งบูลีนจะได้รหัสที่นำไปเก็บไว้ในหน่วยความจำในพื้นที่โปรแกรมผู้ใช้ดังนี้

ตารางที่ 5.2 การจัดเก็บรหัสจากโปรแกรมคำสั่งบูลีน

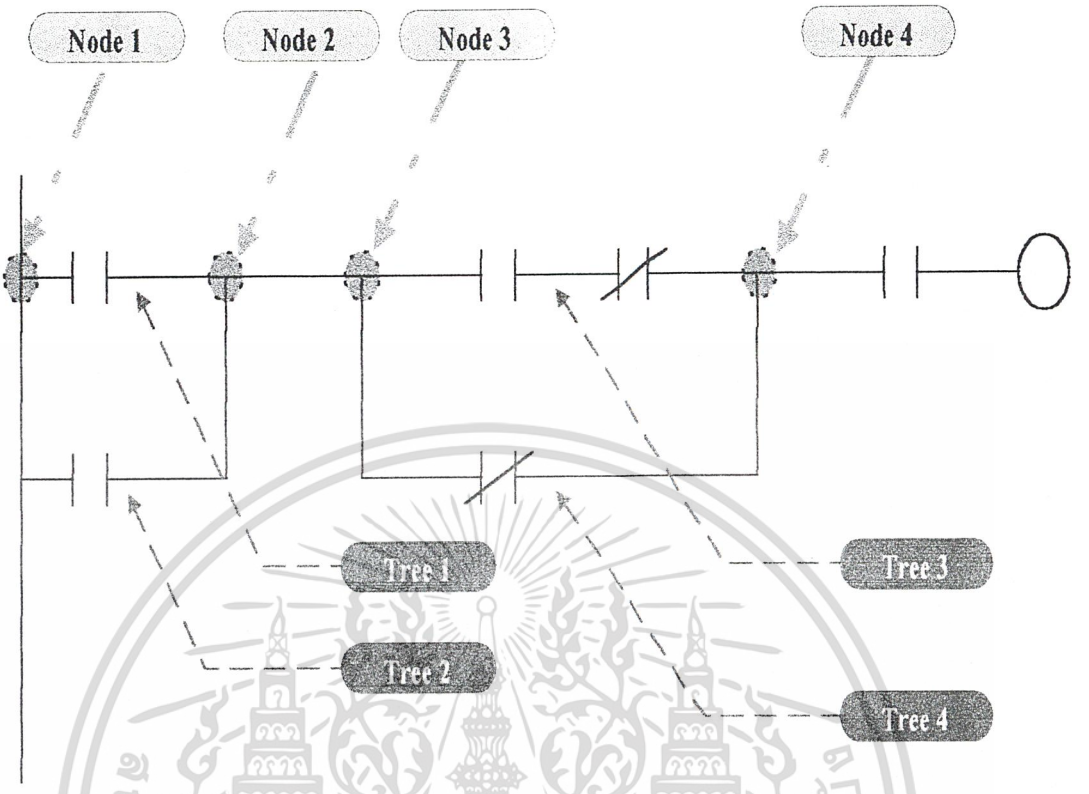
Address	Instruction	Operand	CODE	WORD	BIT
0000	LD NOT HR	0000	90	00	00
0001	AND	0105	85	01	05
0002	OR NOT	2007	99	20	07
0003	OUT	0700	9E	07	00
0004	END		01	00	00

5.5 เทคนิคการแปลงภาพ LADDER

เนื่องจากการแปลงภาพ Ladder ค่อนข้างยุ่งยาก และไม่แน่นอน โดยขึ้นกับผู้ใช้เป็นหลัก ดังนั้นจึงต้อง อาศัยหลักของโครงสร้างข้อมูลเข้ามาเกี่ยวข้องด้วย โดยโครงสร้างข้อมูลที่นำมาใช้ประกอบด้วย

1. ทรี (TREE)
2. โหนด (NODE)
3. สแตก (STACK)

ทั้ง 3 ส่วนนี้มีความสำคัญในการแปลงมากเนื่องจาก ในภาพของ Ladder มีรูปแบบการเชื่อมต่อระหว่างชุดคำสั่งที่สามารถแยกออกเหมือนกับ รากของต้นไม้ ซึ่งในที่นี้ถ้าเทียบกับโครงสร้างของข้อมูล ก็คือ ทรี (TREE) ส่วนบริเวณ ณ จุดเชื่อมต่อจะเป็นส่วนของโหนด (NODE) และ สำหรับการที่โปรแกรมจะสามารถแปลงคำสั่งให้ได้อย่างถูกต้องจึงจำเป็นต้อง อาศัยหลักของสแตก (STACK) เข้าช่วยอีกด้วย เพราะการเชื่อมต่อจะมีอยู่หลายตำแหน่งทำให้ต้องมีการจดจำตำแหน่งของ NODE ต่างๆ อยู่ตลอดเวลาที่ทำการแปลงภาพ Ladder



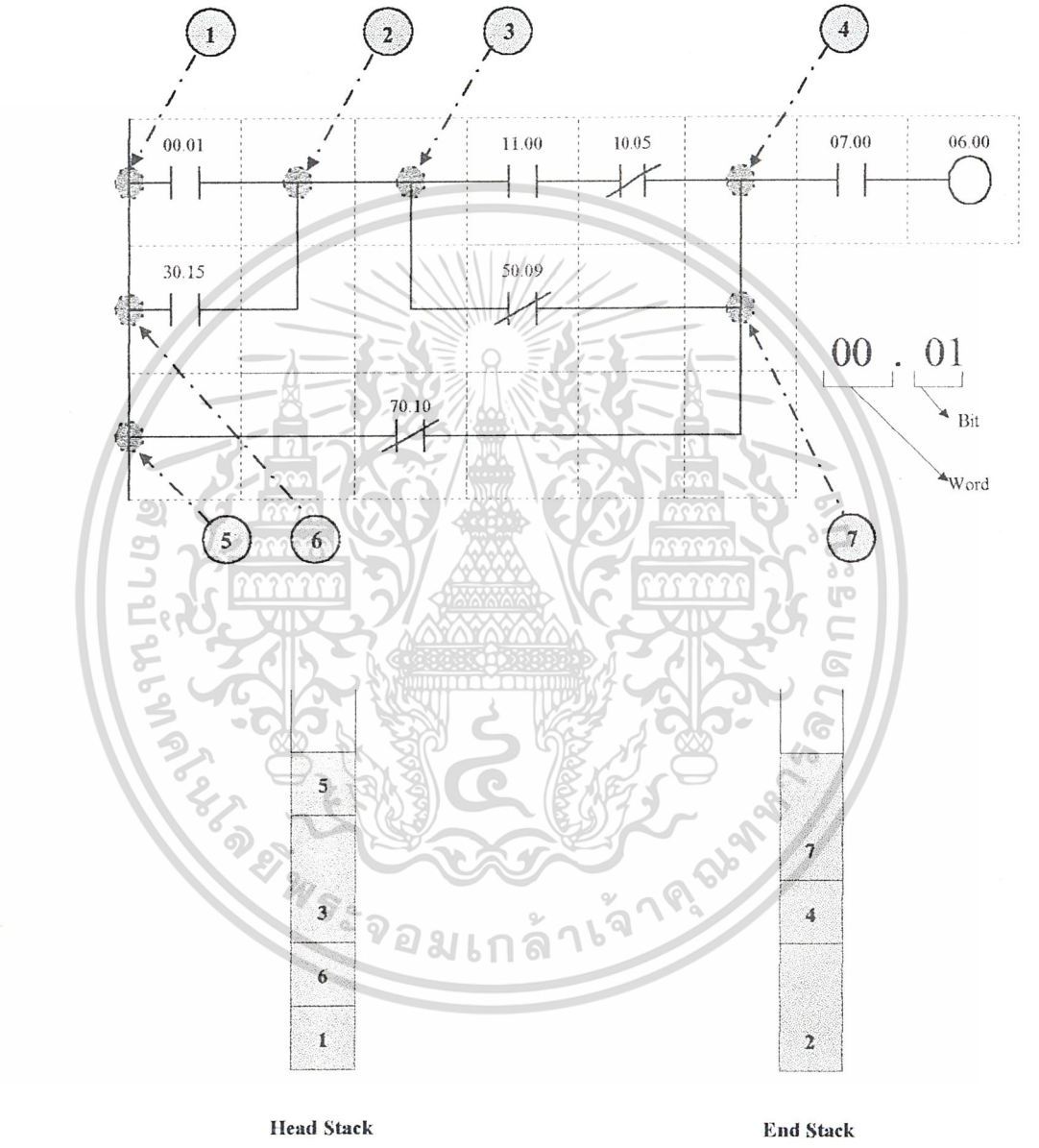
รูปที่ 5.4 แสดงการเชื่อมต่อและจุดที่ใช้แปลง LADDER

รูปที่ 5.5 แสดง STACK ที่เก็บค่า NODE ใน LADDER

ในขั้นตอนการแปลงคำสั่งนั้น จะเริ่มจากการจดจำตำแหน่งของ NODE เป็นอันดับแรกเสียก่อน โดยในที่นี้จะเก็บลง STACK ซึ่งจะเก็บอยู่ด้วยกัน 2 จุด คือ NODE เริ่มต้น ซึ่งจะอยู่ทางซ้ายของจุดแยกให้เก็บลง HEAD STACK ส่วนอีก NODE ที่อยู่ทางขวาจะเก็บลง END STACK

ดังรูป หลังจากนั้นจึงทำการแปลงภาพเป็นคำสั่งตามรูปภาพที่อยู่ใน TREE ซึ่งมีจุดเริ่มต้นการแปลงที่ NODE ใน Head Stack และสิ้นสุดการแปลงที่ NODE ใน End Stack ดังตัวอย่างต่อไปนี้

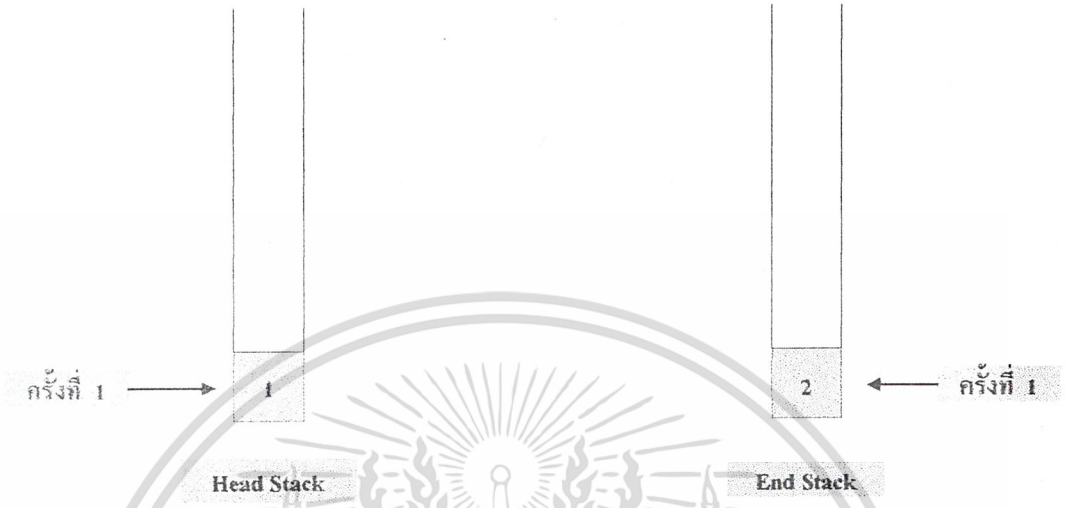
<u>ตัวอย่าง</u>



รูปที่ 5.6 ตัวอย่างการแปลงภาพ LADDER เป็นคำสั่ง

มีขั้นตอนการแปลง Ladder ดังนี้

1. เก็บค่า NODE 1 ลง Head Stack และ เก็บค่า NODE 2 ลง End Stack



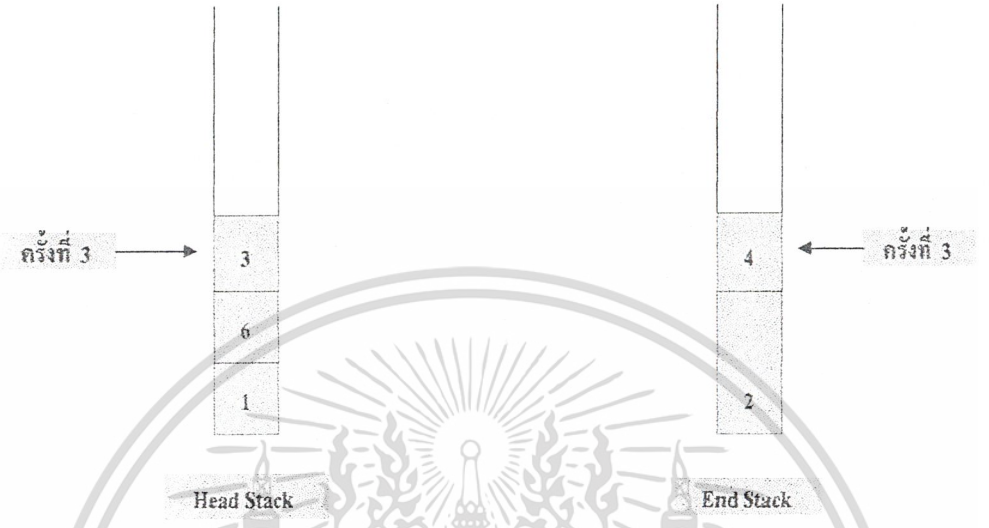
รูปที่ 5.7 เก็บค่า STACK ครั้งที่1

2. ทำการแปลง LD(00.01)
3. เก็บค่า NODE 6 ลง Head Stack

รูปที่ 5.8 เก็บค่า STACK ครั้งที่2

4. ทำการแปลง Function 1 = LD(00.01) OR(30.15)

5. เก็บค่า NODE 3 ลง Head Stack และ เก็บค่า NODE 4 ลง End Stack

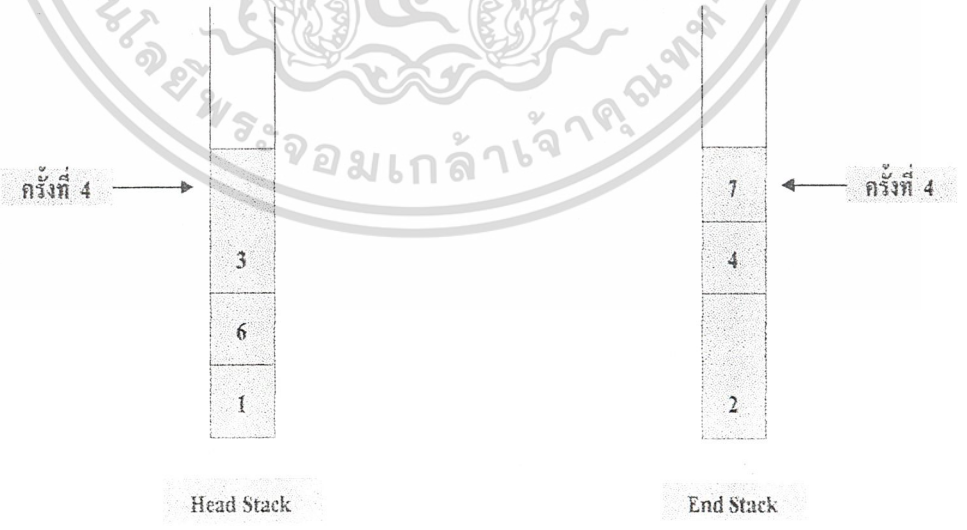


รูปที่ 5.9 เก็บค่า STACK ครั้งที่3

6. ทำการแปลง LD(11.00) ANDNOT(10.05)

7. เก็บค่า NODE 7 ลง End Stack

8. ทำการแปลง Function 2 =LD(11.00)ANDNOT(10.05)ORNOT(50.09)

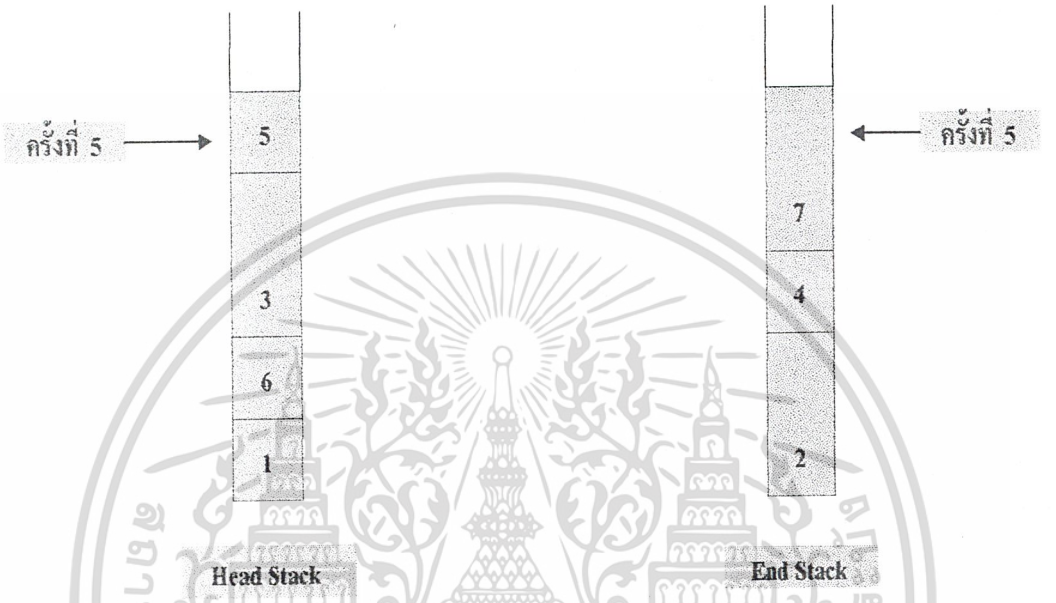


รูปที่ 5.10 เก็บค่า STACK ครั้งที่4

9. ทำการรวมคำสั่ง

Function1+Function2 = LD(00.01) OR(30.15) LD(11.00) ANDNOT(10.05) ORNOT(50.09) ANDLD

10. เก็บค่า NODE 5 ลง Head Stack



รูปที่ 5.11 เก็บค่า STACK ครั้งที่5

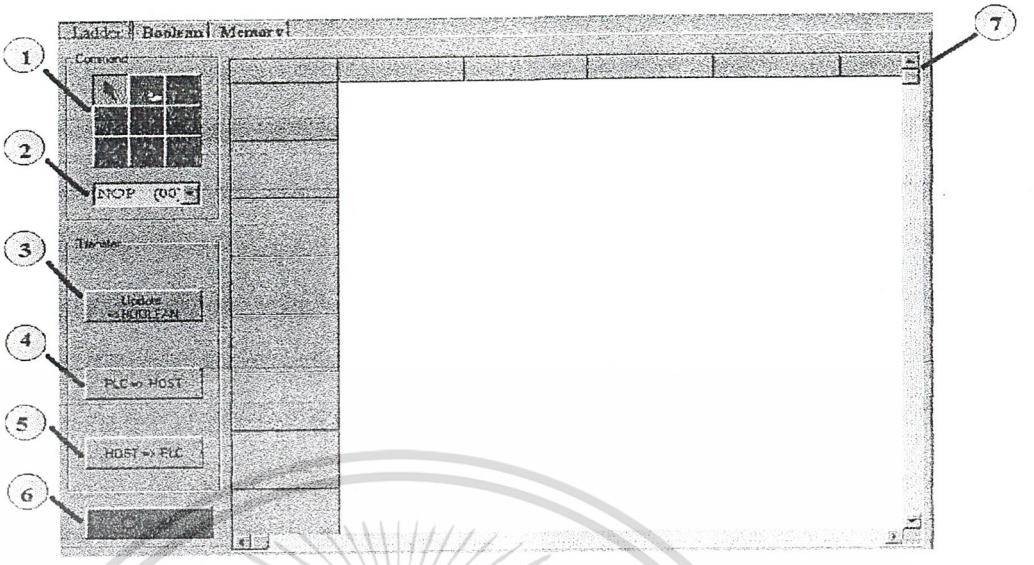
11. ทำการแปลง (Function1+Function2) ORNOT(70.10)

12. รวมคำสั่ง (Function1+Function2) ORNOT(70.10) AND(07.00) OUT(06.00)

5.6 การใช้งานโปรแกรม Ladder Software Support

5.6.1 โปรแกรมส่วนของ Ladder

หมายเลข I - เป็น Instruction ที่ใช้เพื่อเรียกคำสั่งทาง Logic และ Special Function ขึ้นมาเพื่อกำหนดโปรแกรมในรูปของ คำสั่ง ladder เมื่อผสมผสานคีย์คำสั่งเหล่านี้เข้าด้วยกัน แล้วจะสามารถกำหนดลักษณะของ Boolean Instruction ซึ่งนอกเหนือจาก Basic Logic Instruction แล้ว Special Function สามารถกำหนดรูปแบบได้มากกว่า 50 Instruction จะเห็นได้ว่าเราสามารถทำการควบคุมได้อย่างสมบูรณ์ และมีลูกเล่นมากมายตามแต่จินตนาการที่จะเกิดขึ้น เพื่อให้การควบคุมเป็นไปอย่างมีประสิทธิภาพมากที่สุด



รูปที่ 5.12 ภาพส่วนของ Ladder

หมายเลข 2 - เป็นฟังก์ชั่นพิเศษ ได้แก่

1. กลุ่มคำสั่งควบคุมหน่วยความจำแบบบิท ได้แก่ IL,ILC,KEEP,DIFU,DIFD
2. กลุ่มคำสั่งการย้ายข้อมูลแบบ BIT,DIGIT และ WORD ได้แก่ MOVB,MOVD,MOV,MVN
3. กลุ่มคำสั่งการเลื่อนข้อมูลแบบ BIT,DIGIT และ WORD ได้แก่ SFT,SFTR,WSFT,ASL,ASR,ROL,ROR,SLD,SRD
4. กลุ่มคำสั่งการคำนวณทางคณิตศาสตร์แบบ BIT ได้แก่ ADB,SBB,MLB,DVB
5. กลุ่มคำสั่งการคำนวณทางคณิตศาสตร์แบบ BCD ได้แก่ ADD,SUB,MUL,DIV,ROOT,INC,DEC,STC,CLC
6. กลุ่มคำสั่งการคำนวณแบบ HEX ได้แก่ CMP,COM
7. กลุ่มคำสั่งลอจิกแบบ WORD ได้แก่ ANDW,ORW,XORW,XNRW
8. กลุ่มคำสั่งการส่ง – รับข้อมูล ASCII ทางพอร์ท RS 323 C

หมายเลข 3 - เป็นปุ่ม Update Boolean จะใช้ในกรณีที่มีการป้อน Boolean มาเป็นคำสั่ง Ladder หรือมีการเปลี่ยนแปลงแก้ไขในส่วนของ Boolean

หมายเลข 4 - แสดงการติดต่อสื่อสารที่ใช้ในการตรวจสอบสถานะการทำงาน ทั้งระยะไกล และระยะใกล้ ซึ่งปุ่มนี้เป็นการติดต่อระหว่าง PLC ไป Host link

หมายเลข 5 - แสดงการติดต่อสื่อสารที่ใช้ในการตรวจสอบสถานะการทำงาน ทั้งระยะไกล และระยะใกล้ ซึ่งปุ่มนี้เป็นการติดต่อระหว่าง Host link ไป PLC

หมายเลข 6 - ปุ่ม Clear มีไว้สำหรับลบสถานะการแสดงผลเดิมของค่าข้อมูลแบบลอจิก, ฟังก์ชั่นพิเศษ,ค่า Data memory ข้อมูลทั้งหมดในหน้าจอนั้นๆ

หมายเลข 7 - หน้าจอแสดงผลค่าสภาวะทางลอจิก,ค่าฟังก์ชั่นพิเศษ,ค่า memory และลำดับ Contact relay,input,output อื่นๆ

5.6.2 โปรแกรมส่วนของBoolean

หมายเลข I - ใช้ในส่วนการแสดงผลก่อนป้อนคำสั่ง Add ลงในหน้าจอ ตัวอย่าง คำสั่งพื้นฐานแบบลอจิก เช่น LD,AND,OR,TIM,CNT,NOT,ANDNOT,ORNOT,OUTNOT และ คำสั่งพิเศษ เช่น

1. กลุ่มคำสั่งควบคุมหน่วยความจำแบบบิท ได้แก่ IL,ILC,KEEP,DIFU,DIFD
2. กลุ่มคำสั่งการย้ายข้อมูลแบบ BIT,DIGIT และ WORD ได้แก่ MOVB,MOVD,MOV, MVN
3. กลุ่มคำสั่งการเลื่อนข้อมูลแบบ BIT,DIGIT และ WORD ได้แก่ SFT,SFTR,WSFT, ASL,ASR,ROL,ROR,SLD,SRD
4. กลุ่มคำสั่งการคำนวณทางคณิตศาสตร์แบบ BIT ได้แก่ ADB,SBB,MLB,DVB
5. กลุ่มคำสั่งการคำนวณทางคณิตศาสตร์แบบ BCD ได้แก่ ADD,SUB,MUL,DIV, ROOT,INC,DEC,STC,CLC
6. กลุ่มคำสั่งการคำนวณแบบ HEX ได้แก่ CMP,COM
7. กลุ่มคำสั่งลอจิกแบบ WORD ได้แก่ ANDW,ORW,XORW,XNRW
8. กลุ่มคำสั่งการส่ง – รับข้อมูล ASCII ทางพอร์ท RS 323 C

หมายเลข 2 - แสดงตำแหน่ง DATA ในส่วนของ memory ซึ่งประกอบด้วย

- รีเลย์ภายใน(Internal Relay) จำนวน 77 เวิลด์ CH000…CH077 หรือ 1,232 บิท
- รีเลย์พิเศษ(Special Relay) จำนวน 2 เวิลด์ CH078,CH079
- หน่วยความจำ(Data Memory) จำนวน 1000 เวิลด์ DM000…DM1000
- โฮลดิ้งรีเลย์(Holding Relay) จำนวน 1,000 เวิลด์ หรือ 1,600
  เริ่มจาก HR000-HR099 รวม 100 CH
- ลิงค์รีเลย์(Link Relay) จำนวน 64 เวิลด์ หรือ 1,024 บิท เริ่มจาก LR000-LR063 รวม 64 CH

หมายเลข 3 - แสดงคำสั่งพื้นฐานแบบลอจิก เช่น LD,AND,OR,TIM,CNT,NOT, ANDNOT,ORNOT,OUTNOT

หมายเลข 4 - แสดงกลุ่มคำสั่งพิเศษ ได้แก่

1. กลุ่มคำสั่งควบคุมหน่วยความจำแบบบิท ได้แก่ IL,ILC,KEEP,DIFU,DIFD

2. กลุ่มคำสั่งการย้ายข้อมูลแบบ BIT,DIGIT และ WORD ได้แก่ MOVB,MOVD, MOV,MVN

3. กลุ่มคำสั่งการเลื่อนข้อมูลแบบ BIT,DIGIT และ WORD ได้แก่ SFT,SFTR, WSFT, ASL,ASR,ROL,ROR,SLD,SRD

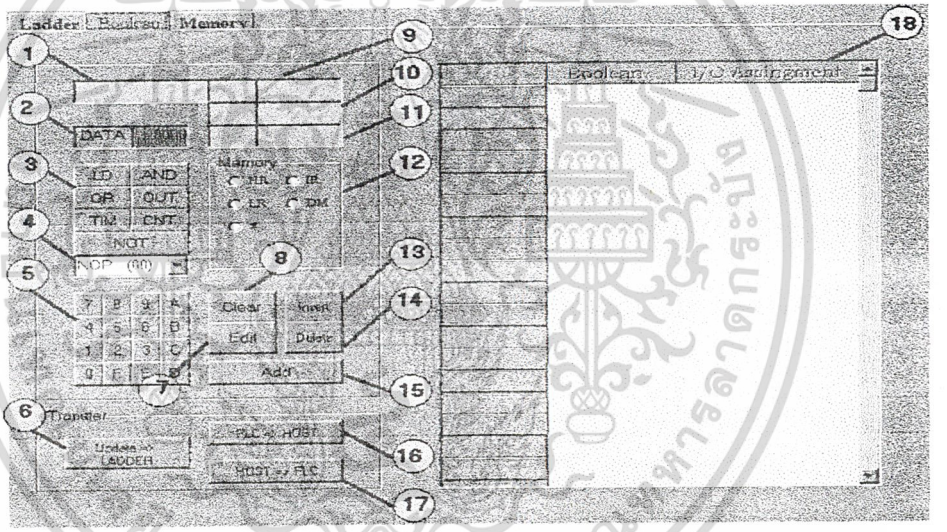
4. กลุ่มคำสั่งการคำนวณทางคณิตศาสตร์แบบ BIT ได้แก่ ADB,SBB,MLB,DVB

5. กลุ่มคำสั่งการคำนวณทางคณิตศาสตร์แบบ BCD ได้แก่ ADD,SUB,MUL,DIV, ROOT,INC,DEC,STC,CLC

6. กลุ่มคำสั่งการคำนวณแบบ HEX ได้แก่ CMP,COM

7. กลุ่มคำสั่งลอจิกแบบ WORD ได้แก่ ANDW,ORW,XORW,XNRW

8. กลุ่มคำสั่งการส่ง – รับข้อมูล ASCII ทางพอร์ท RS 323 C



รูปที่ 5.13 ภาพส่วนของ Boolean

หมายเลข 5 - ปุ่มป้อนค่าตำแหน่งของ Data ในส่วน memory สามารถแสดงค่าได้ตั้งแต่ตำแหน่ง 0000-FFFF

หมายเลข 6 - ปุ่ม Update Ladder จะใช้ในกรณีที่มีการ Ladder มาเป็นคำสั่ง Boolean หรือมีการเปลี่ยนแปลงแก้ไขในส่วนของ Ladder

หมายเลข 7 - ปุ่ม Edit มีไว้สำหรับป้อนหรือแก้ไขค่าข้อมูลแบบลอจิกและฟังก์ชั่นพิเศษเพิ่มเติมลงไปในหน้าจอ

หมายเลข 8 - ปุ่ม Clear มีไว้สำหรับลบสถานะการแสดงผลเดิมของค่าข้อมูลแบบลอจิก, ฟังก์ชั่นพิเศษ,ค่า Data memory ข้อมูลทั้งหมดในหน้าจอนั้นๆ

หมายเลข 9 - แสดงค่า Data,จำนวนตัวเลขของ Contact,รีเลย์ ของ I/O

หมายเลข 10 - แสดงค่าตำแหน่ง ตัวเลขในส่วนของ memory เงื่อนไขเพิ่มเติมของคำสั่ง เช่น ค่าเวลาของ Timer,ค่าการนับของ Counter

หมายเลข 11 - แสดงค่าเงื่อนไขเพิ่มเติมของคำสั่ง กรณีมีการป้อนค่า 3 ค่า

หมายเลข 12 - แสดงค่าในส่วนของ memory ซึ่งประกอบด้วย

- รีเลย์ภายใน(Internal Relay) จำนวน 77 เวิลด์ CH000...CH077 หรือ 1,232 บิท
- รีเลย์พิเศษ(Special Relay) จำนวน 2 เวิลด์ CH078,CH079
- หน่วยความจำ(Data Memory) จำนวน 1000 เวิลด์ DM000...DM1000
- โฮลดิ้งรีเลย์(Holding Relay) จำนวน 1,000 เวิลด์ หรือ 1,600
  เริ่มจาก HR000-HR099 รวม 100 CH
- ลิงค์รีเลย์(Link Relay) จำนวน 64 เวิลด์ หรือ 1,024 บิท เริ่มจาก LR000-LR063 รวม 64 CH

หมายเลข 13 - เป็นปุ่มแทรกค่า Instruction ที่ต้องการเข้าไปในโปรแกรม

หมายเลข 14 - เป็นปุ่มที่ใช้ในการลบค่า Instruction ที่ต้องการการออกจากโปรแกรม

หมายเลข 15 - เป็นปุ่ม ป้อนค่าฟังก์ชั่นทางลอจิก ฟังก์ชั่นพิเศษ ค่าตำแหน่งของ Contact รีเลย์เข้าไปสู่หน้าจอ

หมายเลข 16 - แสดงการติดต่อสื่อสารที่ใช้ในการตรวจสอบสถานะการทำงาน ทั้งระยะไกล และระยะใกล้ ซึ่งปุ่มนี้เป็นการติดต่อระหว่าง PLC ไป Host link

หมายเลข 17 - แสดงการติดต่อสื่อสารที่ใช้ในการตรวจสอบสถานะการทำงาน ทั้งระยะไกล และระยะใกล้ ซึ่งปุ่มนี้เป็นการติดต่อระหว่าง Host link ไป PLC

หมายเลข 18 - หน้าจอแสดงผลค่าสภาวะทางลอจิก,ค่าฟังก์ชั่นพิเศษ,ค่า memory และลำดับ Contact relay,input,output อื่นๆ

5.6.3 โปรแกรมส่วนของ Memory

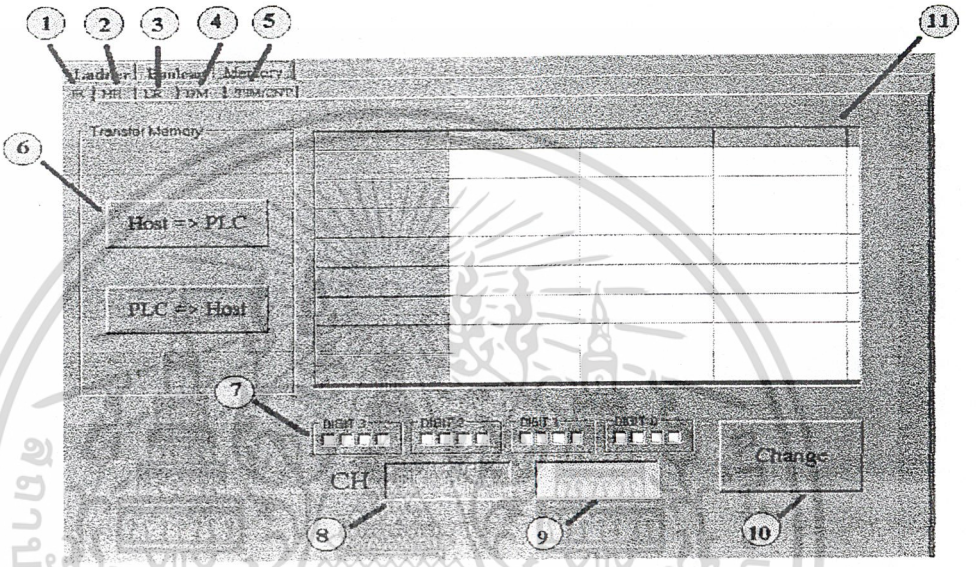
หมายแลข 1 - IR (Internal relay) เป็นส่วนแสดงผลของ รีเลย์ภายใน จำนวน 77 word CH000...CH077 หรือ 1,232 bit

หมายแลข 2 - HR (Holding Relay) เป็นส่วนแสดงผลของ รีเลย์แบบคงค่าสภาวะไว้ โดยเริ่มจากตำแหน่ง HR 000 จนถึง HR 915

หมายแลข 3 - LR (Link relay) เป็นส่วนแสดงผลของ รีเลย์เชื่อมต่อใช้ในการสื่อสารข้อมูลในเครือข่าย จำนวน 64 เวิร์ด หรือ 1,024 บิท

หมายแลข 4 - DM (Data memory) เป็นส่วนแสดงผลของหน่วยความจำข้อมูล โดยในการเก็บข้อมูลจะยังคงค่าสภาวะของข้อมูลเอาไว้ถึงแม้ว่าแหล่งจ่ายไฟจะหายไป เริ่มจาก DM000 – DM999 รวม 1000 CH

หมายแลข 5 - TIM/CNT เป็นตัวตั้งเวลาและตัวนับจะถูกให้ใช้ในพื้นที่เดียวกันในการเก็บข้อมูลมี 1000 CHและในแต่ละหน้าของ IR ,HR, LR, DM, TIM/CNT จะมีฟังก์ชั่นการทำงานที่คล้ายกันดังนี้



รูปที่ 5.14 ภาพส่วนของ Memory

หมายเลข 6 - Tranfer memory เป็นส่วนของข้อตกลงในการสื่อสารอย่างง่ายแบ่งเป็น

- Host to PLC เป็นการติดต่อสื่อสารอย่างง่ายกับเครื่องควบคุม PLC
- PLC to Host เป็นการติดต่อสื่อสารอย่างง่ายระหว่างเครื่องควบคุม PLC เครื่องคอมพิวเตอร์

หมายเลข 7 - Digit เป็นส่วนของการแสดงผลของหน่วยความจำในรูปแบบ Binary 16 Bit จาก Chanel

หมายเลข 8 - Chanel แสดงผลแบบตำแหน่งในรูปแบบของ Row และ Column ของหน่วยความจำ

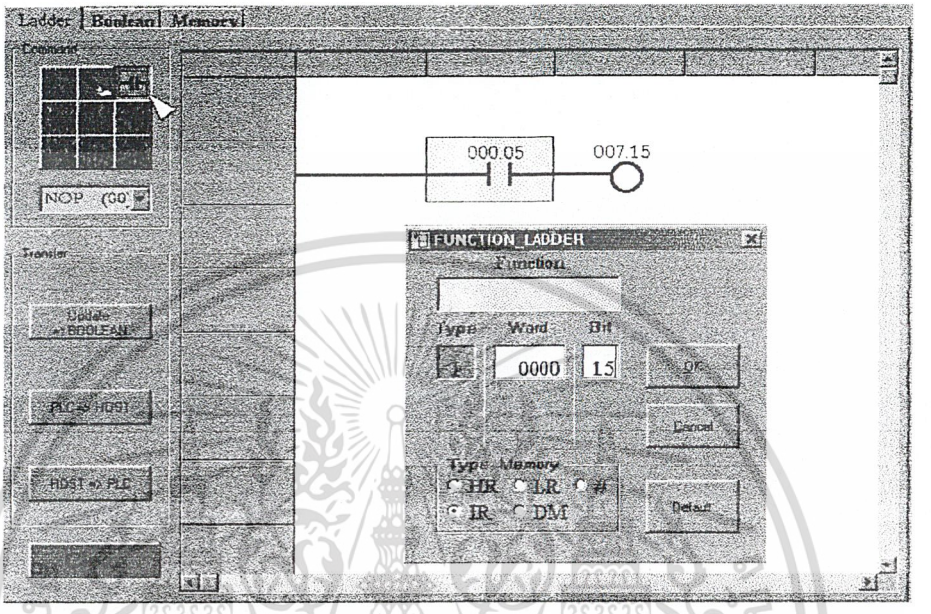
หมายเลข 9 - เป็นส่วนแสดงผลของเลขฐาน 16 ที่ส่งค่ามาจาก หน้าจอ Grid

หมายเลข 10 - ปุ่ม Change ใช้แก้ไขค่า เลขฐาน 16 จากช่องText ไปสู่ช่องGrid

หมายเลข 11 - หน้าจอแสดงผลค่าสภาวะของเลขฐาน 16 ที่ส่งเข้ามาจากหน่วยความจำของเครื่อง PLC

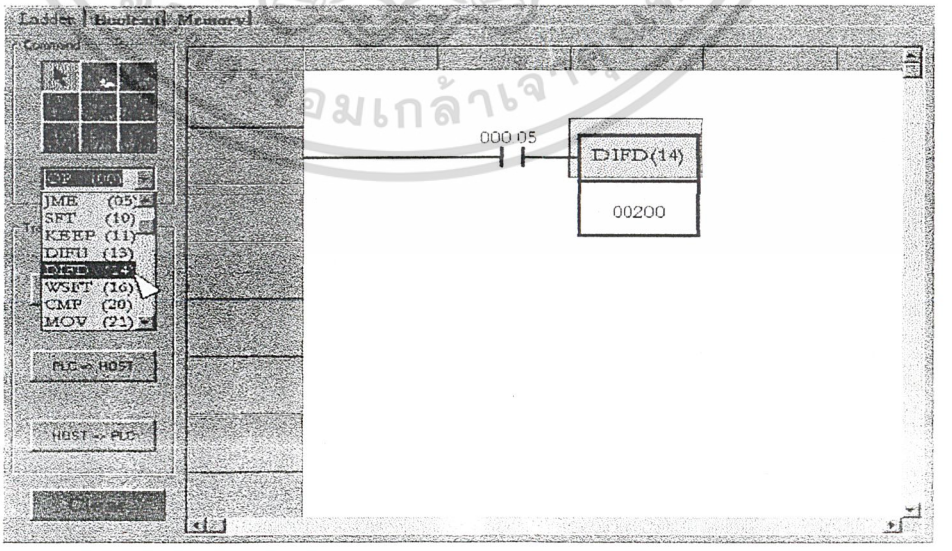
5.6.4 การใช้งานโปรแกรม

5.6.4.1 คลิกปุ่มคำสั่ง ในที่นี่จะใช้เป็นปุ่ม LD เพื่อป้อนค่าลง ตาราง Ladder หลังจากนั้นจึงไปคลิกตาราง Ladder เพื่อวางคำสั่ง LD ดังรูป



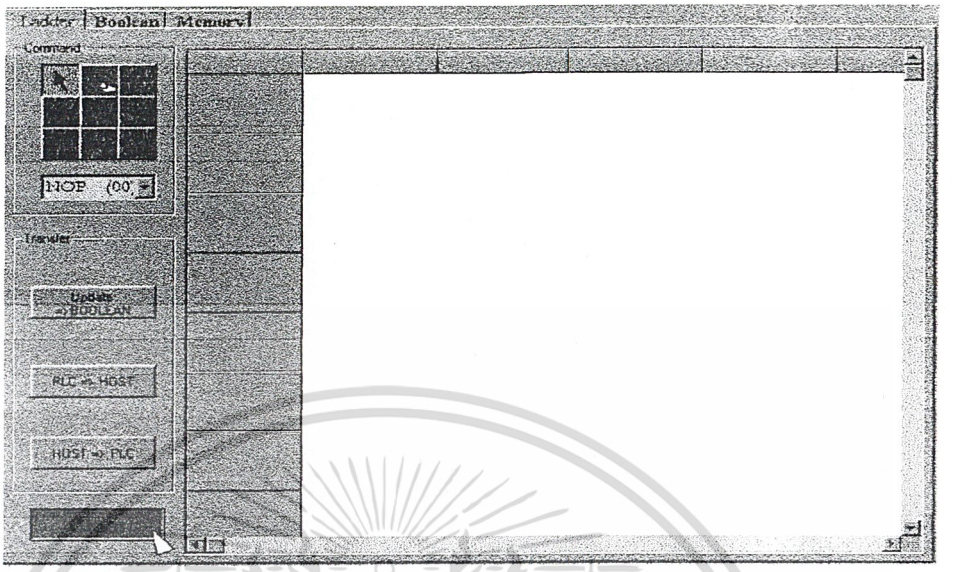
รูปที่ 5.15 การป้อนค่าลงตาราง Ladder

5.6.4.2 คลิกปุ่มคำสั่ง ในที่นี่จะใช้เป็นปุ่มเลือก FUCTION เพื่อป้อนค่าลง ตาราง Ladder หลังจากนั้นจึงไปคลิกตาราง Ladder เพื่อวาง Function "DIFD(14)" ดังรูป



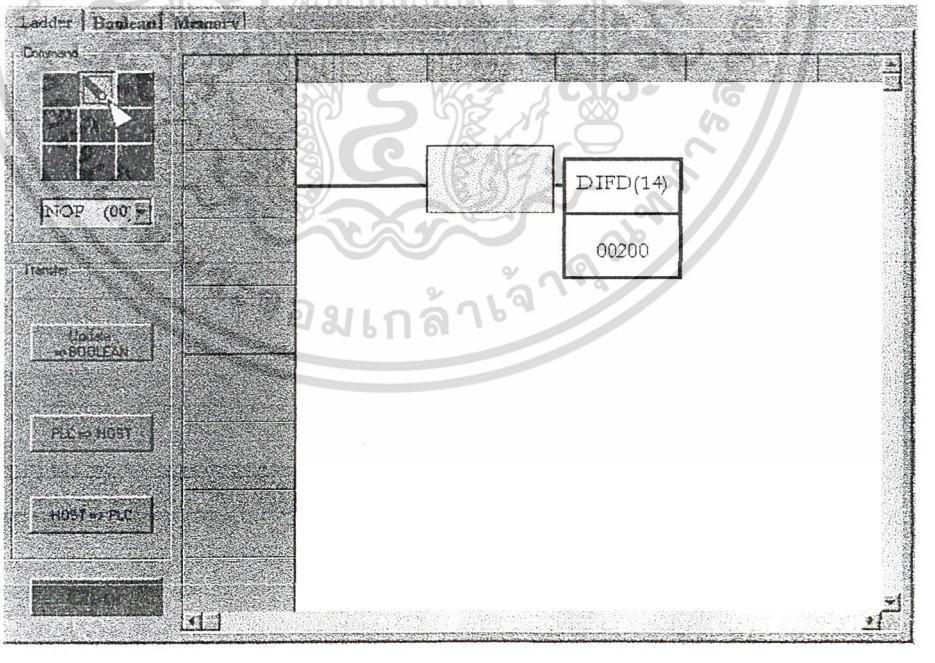
รูปที่ 5.16 การป้อน function ลงตาราง Ladder

5.6.4.3 คลิกปุ่มคำสั่ง Clear เพื่อลบค่าต่างๆใน ตาราง Ladder ดังรูป



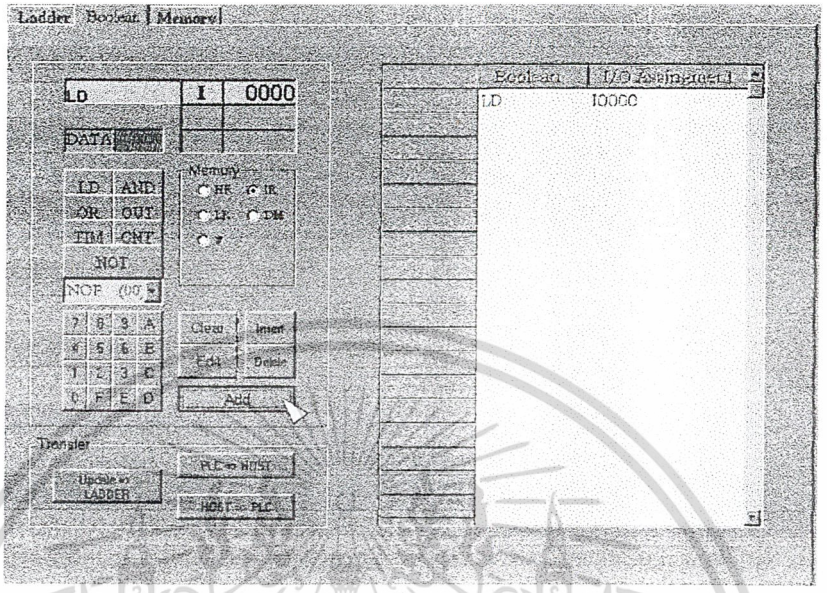
รูปที่ 5.17 การ Clear ค่าในตาราง Ladder

5.6.4.4 คลิกปุ่ม Delete แล้วเลือกช่องในตาราง Ladder เพื่อลบค่าหรือ Function ที่ต้องการใน ตาราง Ladder ดังรูป



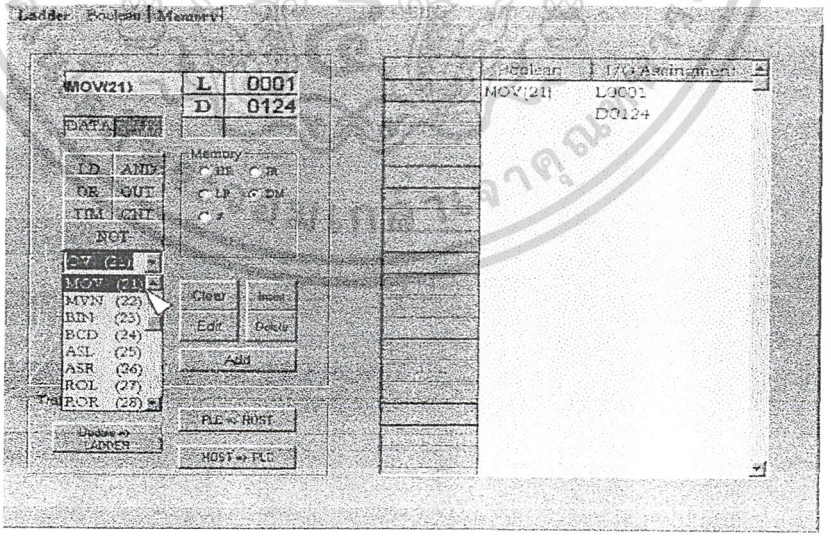
รูปที่ 5.18 การลบค่าในตาราง Ladder

5.6.4.5 คลิกปุ่มคำสั่ง Add เพื่อใส่คำสั่งหรือ Function ที่ต้องการใน ตาราง Boolean ดังรูป



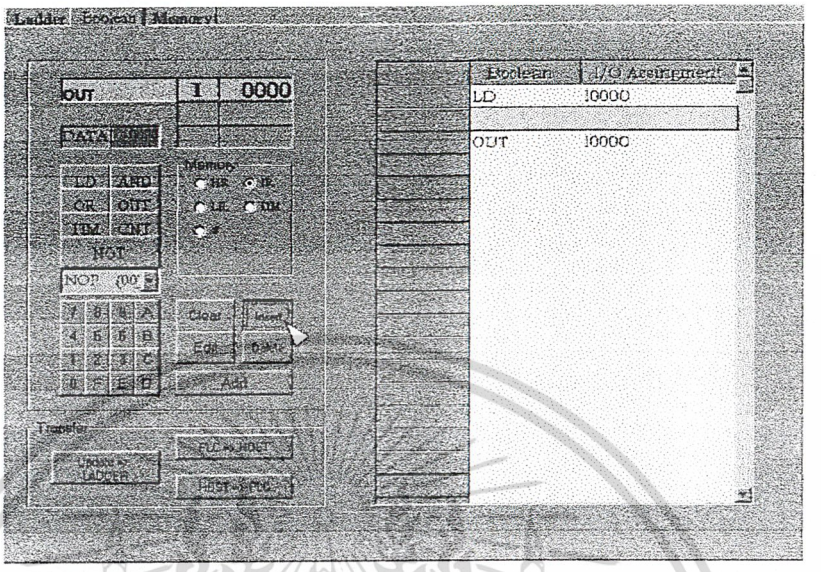
รูปที่ 5.19 การใส่คำสั่งในตาราง Boolean

5.6.4.6 คลิกปุ่มเลือก FUNCTION เพื่อเลือก Function ที่ต้องการแล้วจึงนำไปวางลงในตาราง Boolean ดังรูป



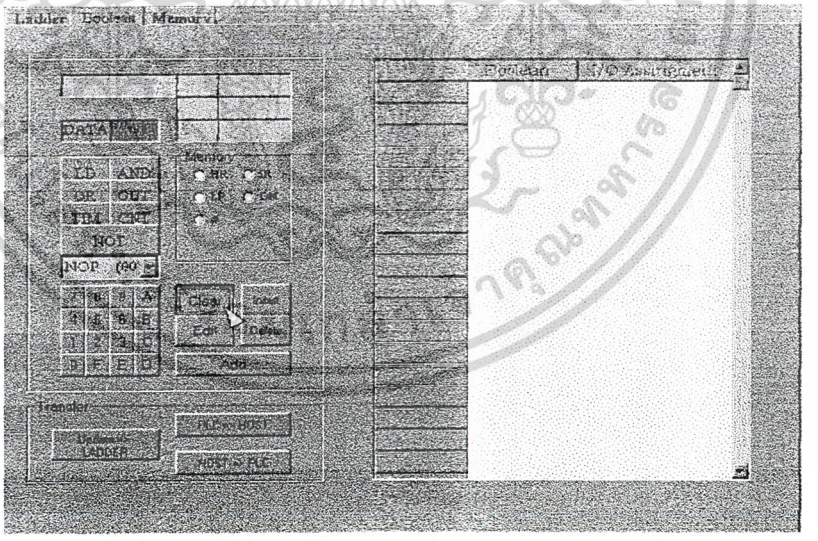
รูปที่ 5.20 การเลือก Function ลงในตาราง Boolean

5.6.4.7 คลิกบรรทัดที่ต้องการในตาราง Boolean แล้วจึงกดปุ่ม Insert ก็จะแทรกบรรทัดใน ตาราง Boolean ได้ ดังรูป



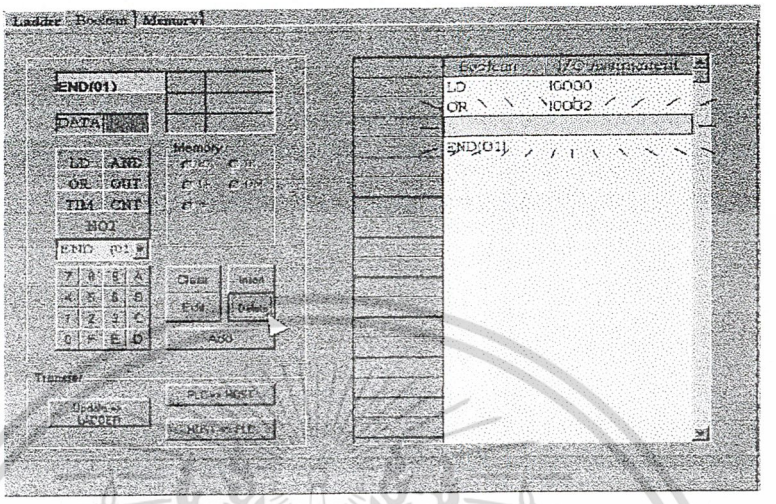
รูปที่ 5.21 การแทรกบรรทัดลงในตาราง Boolean

5.6.4.8 คลิกปุ่ม Clear จะเป็นการ Clear ค่าต่างๆ ใน ตาราง Boolean ดังรูป



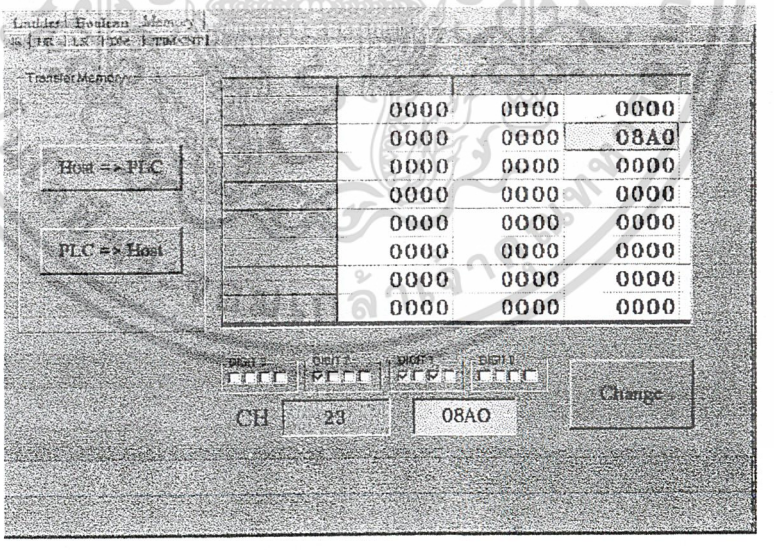
รูปที่ 5.22 การ Clear ค่าต่างๆในตาราง Boolean

5.6.4.9 คลิกบรรทัดที่ต้องการในตาราง Boolean แล้วจึงกดปุ่ม Insert ก็จะแทรกบรรทัดใน ตาราง Boolean ได้ ดังรูป



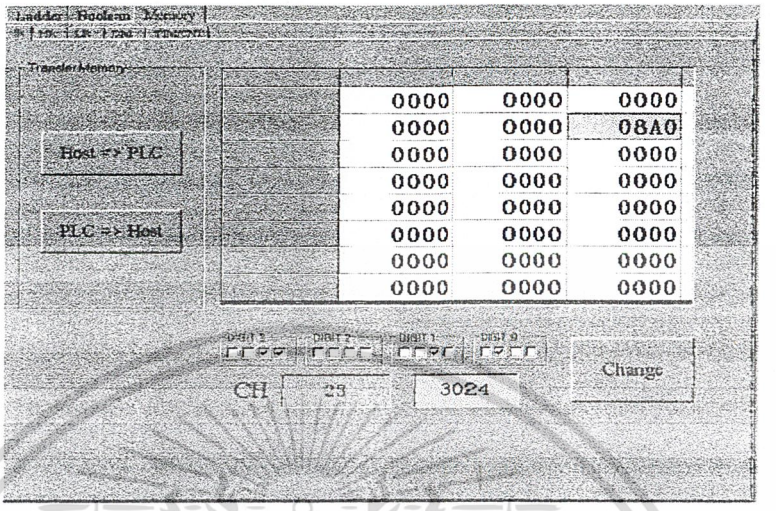
รูปที่ 5.23 การลบบรรทัดในตาราง Boolean

5.6.4.10 ในตาราง Memory เลือกช่องที่ต้องการ ค่าต่างๆก็จะไปปรากฏในช่อง Channel และ ช่องใส่ค่าหน่วยความจำ ดังรูป



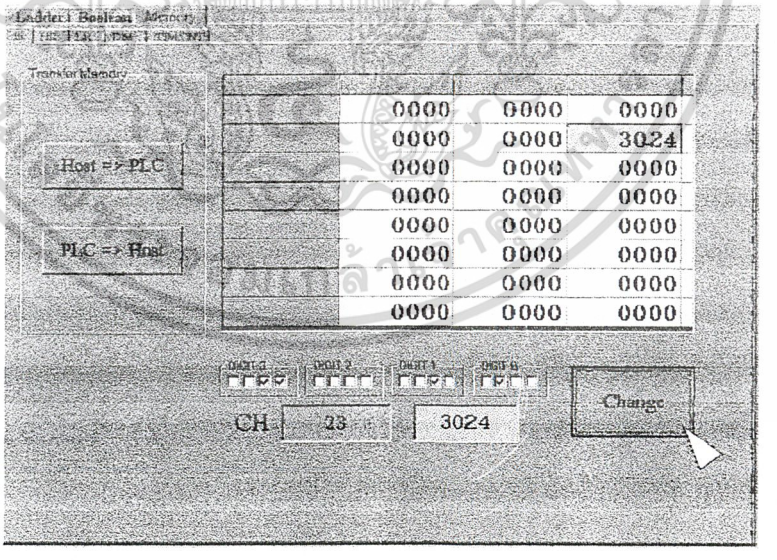
รูปที่ 5.24 การเลือกช่องในตาราง Memory

5.6.4.11 แก้ไขค่าหน่วยความจำแบบ bits โดยเช็คถูกที่ช่องของแต่ละ Digit ในตาราง Memory ดังรูป



รูปที่ 5.25 การแก้ไขค่าหน่วยความจำ

5.6.4.12 แก้ไขค่าหน่วยความจำแบบ bits โดยเช็คถูกที่ช่องของแต่ละ Digit ในตาราง Memory ดังรูป



รูปที่ 5.26 การเปลี่ยนค่าหน่วยความจำ

บทที่ 6
สรุปและวิจารณ์

จากโครงงานนี้ ซึ่งมีส่วนประกอบทั้งหมดเป็นโปรแกรม โดยภายในโปรแกรมจะแบ่งเป็น 3 ส่วนหลักๆ คือ

1. ส่วนของ LADDER ทำหน้าที่แปลงภาพ Graphic Ladder ให้กลายเป็นคำสั่งของเครื่อง PLC
2. ส่วนของ BOOLEAN ทำหน้าที่แปลงคำสั่งในตารางให้กลายเป็นรหัสภาษาเครื่อง PLC
3. ส่วนของ MEMORY ทำหน้าที่แก้ไขหน่วยความจำใน เครื่อง PLC

ในการเขียนโปรแกรมนี้จะใช้ภาษา Virsual Basic เวอร์ชัน 6 เป็นภาษาหลักในการพัฒนา สำหรับการเขียนโปรแกรมนั้นจะมีความยุ่งยากและแตกต่างกันไปในแต่ละส่วน สำหรับในส่วนของ LADDER จะมีความยุ่งยากในการแปลงเนื่องจากว่าผู้ใช้งานโปรแกรม สามารถที่จะเขียนโปรแกรมได้อย่างอิสระเพราะเป็นภาพ Graphic ทำให้เวลาที่แปลงภาพมาเป็นคำสั่งของโปรแกรมเกิดความยุ่งยากมาก ซึ่งรูปแบบของภาพนั้นไม่แน่นอนขึ้นอยู่กับผู้ใช้เป็นหลัก

ในการพัฒนาของส่วน LADDER ต้องทำพัฒนาหลายครั้งเนื่องจากเกิดปัญหาในการแปลง LADDER ซึ่งมีลำดับขั้นตอนดังนี้

1. ในครั้งแรกใช้วิธีการอ่านคำสั่งเรียงจากซ้ายไปขวาซึ่งสามารถอ่านคำสั่งได้ แต่เกิดปัญหาขึ้นในการอ่านคำสั่งที่ต่อขนานอยู่ เช่น คำสั่ง OR ทำให้ไม่สามารถอ่านได้
2. ครั้งที่ 2 ใช้วิธีของ Node ซึ่งเป็นวิธีของโครงสร้างข้อมูล โดยจะอ่านคำสั่ง ณ จุดต่อที่เป็นคำสั่ง OR ได้ แต่เมื่อมีคำสั่งซ้อนหรือขนานกันหลายชั้นก็ไม่สามารถอ่านได้
3. ครั้งที่ 3 ใช้วิธีของ Stack เข้าช่วย โดยจะเก็บค่า Node ต่างๆไว้ด้วย ซึ่งจะเก็บค่า Node เริ่มต้น และNode จุดปลาย เพื่อที่จะใช้ในการอ่านคำสั่ง หลังจากนั้นจึงทำการแปลงคำสั่ง ซึ่งวิธีนี้จะทำให้สามารถอ่านคำสั่งที่ซับซ้อนได้

สำหรับส่วนของ BOOLEAN จะมีความยุ่งยากที่ต่างไปจาก LADDER เนื่องจากในส่วนนี้จะใช้เป็นคำสั่งทั้งหมด จึงทำให้ปัญหาส่วนใหญ่อยู่ที่ปุ่มคำสั่ง ซึ่งในที่นี้ได้แก่ ปุ่ม ADD , ปุ่ม DELETE และ ปุ่ม INSERT นอกจากนั้นก็ยังมีส่วนของโปรแกรมย่อย ที่จะเขียนคำสั่งต่างๆลงในตาราง BOOLEAN อีกด้วย ซึ่งมีปัญหาดังต่อไปนี้

1. ปุ่มคำสั่ง ADD จะเป็นตัวจำแนกคำสั่งต่างๆ โดยดูว่าคำสั่งเหล่านั้นเป็นคำสั่ง 1 ,2

หรือ 3 บรรทัด หลังจากที่ส่งคำสั่งไปที่ Grid Boolean แล้ว เซลล์ใน Grid จะขยายตัวโดยอัตโนมัติ ปัญหาที่เกิดกับ ปุ่ม ADD อยู่ที่ ต้องทราบว่าคำสั่งที่ส่งไปที่ Grid Boolean นั้นมีจำนวนข้อมูลเท่าไร ไม่เช่นนั้นแล้วจะส่งข้อมูลผิดพลาด

2. ปุ่ม DELETE จะลบคำสั่งและลดจำนวนเซลล์ใน Grid boolean โดยอัตโนมัติ การลบบรรทัดคำสั่งมีความยุ่งยากมาก เนื่องจากต้องรู้ว่าคำสั่งที่ลบนั้นมีจำนวนกี่บรรทัดจึงจะลบได้อย่างถูกต้อง และถ้าบรรทัดใดว่างก็ต้องตรวจสอบว่ามีจำนวนบรรทัดเท่าไรที่คงเหลือไว้ใน Grid Boolean จึงจะลบได้อย่างถูกต้อง

3. ปุ่ม INSERT จะทำการแทรกบรรทัดแต่ก็จะทำการเช็คบรรทัดคำสั่งด้วยโดยไม่ให้เกินจำนวนบรรทัดคำสั่งที่กำหนดไว้

ในส่วนสุดท้ายของ MEMORY จะมีความสำคัญอยู่ที่การแก้ไขค่าในหน่วยจำแบบ Bits โดยอัตโนมัติซึ่งการทำเช่นนี้จะเป็นการสะดวกแก่ผู้ใช้งาน แต่ในส่วนของการเขียนโปรแกรมจะมีความยุ่งยากพอสมควรเนื่องจากต้องเขียนโปรแกรมให้สามารถแก้ไขค่าในแต่ละ Bits ให้ได้อย่างอัตโนมัติ

โปรแกรมนี้สามารถนำไปประยุกต์ใช้งานได้หลากหลายกับเครื่อง PLC ซึ่งอาจจะนำไปใช้ในการเขียนโปรแกรมควบคุมลิฟท์ โปรแกรมควบคุมสายพานลำเลียง ฯลฯ นอกจากนี้แล้วสามารถนำโปรแกรมใน PLC มาแก้ไขได้โดยง่ายอีกด้วย

บรรณานุกรม

กฤษฎา ใจเย็น , ชัยวัฒน์ ลิ้มพรจิตวิไล , "เรียนรู้การเชื่อมต่อ PC กับอุปกรณ์ภายนอกผ่านพอร์ท อนุกรม serial port interfaceing starter book", บริษัท อินโนเวตีฟ เอ็กเพอริเมนต์ จำกัด, 117 หน้า,2542

กิตติ ภักดีวัฒนะกุล , จำลอง ครูอุตสาหะ , "Visual Basic 6 ฉบับโปรแกรมเมอร์", บริษัท ดวงกมล สมัย จำกัด, 621 หน้า,2542

คู่มือการใช้งาน PLC C200HS , บริษัท ออมรอน(ประเทศไทย) จำกัด,501 หน้า,2537

ชัยวุฒิ จันมา , "การเขียนโปรแกรมด้วย Visual Basic 6.0", บริษัท ซีเอ็ดยูเคชั่น จำกัด(มหาชน), 380 หน้า,2540

ทวีชัย ภูรีทิพย์ , "ไขปัญหา RS-232", บริษัท ซีเอ็ดยูเคชั่น จำกัด(มหาชน), 240 หน้า,2538

ธนพล ฉันจรัสวิชัย , "การทำรายงานบนเครื่องพิมพ์ด้วย Visual Basic", บริษัท เอส.พี.ซี.บุ๊คส์ จำกัด,221 หน้า,2542

พุฒิพงษ์ นาคะปัท , "การเขียน GAME บนวินโดส์ ด้วย Visual Basic 6.0", บริษัท ซีเอ็ดยูเคชั่น จำกัด(มหาชน),352 หน้า,2542

ศูนย์ภาษาคอมพิวเตอร์ , "การเขียนโปรแกรมภาษาเบสิก", สำนักพิมพ์ฟิสิกส์เซ็นเตอร์, 461 หน้า, 2531

อุดม จีนประดับ , ดร.สมคิด เรืองธนะสกุลไทย , "โครงสร้างข้อมูล", บริษัท แมคกรอ-ฮิล อินเตอร์เนชั่นแนล เอ็นเตอร์ไพร์ส,อิงค์ จำกัด,417 หน้า,2540

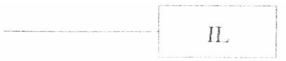
MARK SPENIK , ANDREW J. INDOVINA , PIERRE BOUTQUIN , DAVID JUNG ,"VISUAL BASIC 6 INTERACTIVE COURSE", Macmillan Computer Publishing,1131 pages, 1998

ภาคผนวก

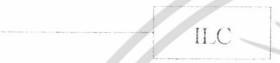
รูปแบบคำสั่ง LADDER

Fun (01) [3 byte]

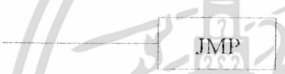
Fun (02) [3 byte]



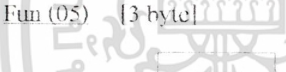
Fun (03) [3 byte]



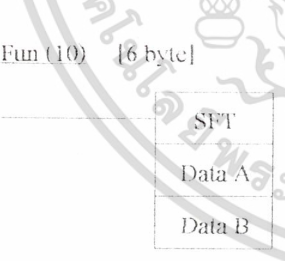
Fun (04) [3 byte]



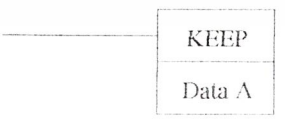
Fun (05) [3 byte]



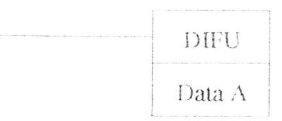
Fun (10) [6 byte]



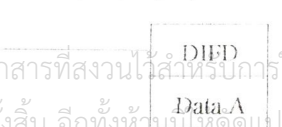
Fun (11) [3 byte]



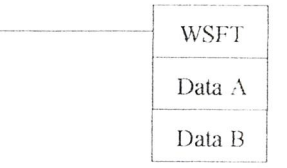
Fun (13) [3 byte]



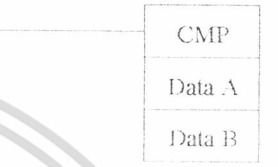
Fun (14) [3 byte]



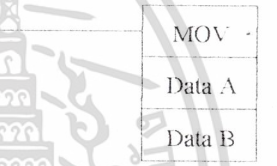
Fun (16) [6 byte]



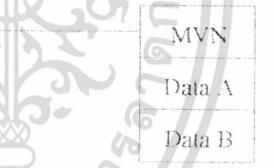
Fun (20) [6 byte]



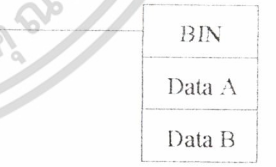
Fun (21) [6 byte]



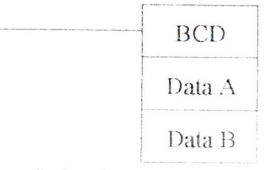
Fun (22) [6 byte]



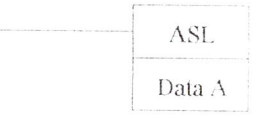
Fun (23) [6 byte]



Fun (24) [6 byte]

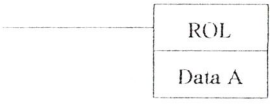


Fun (25) [3 byte]



Fun (26) [3 byte]

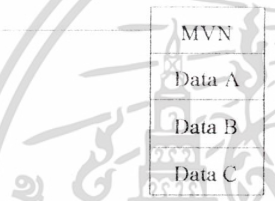
Fun (27) [3 byte]



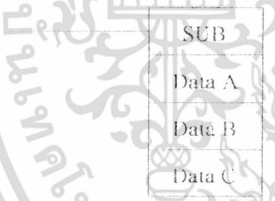
Fun (28) [3 byte]

Fun (29) [3 byte]

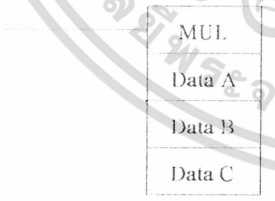
Fun (30) [9 byte]



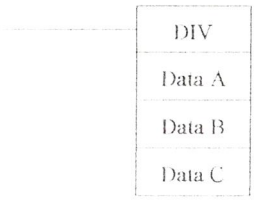
Fun (31) [9 byte]



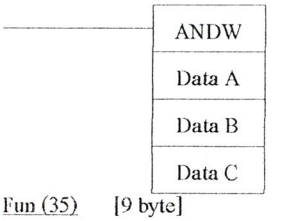
Fun (32) [9 byte]



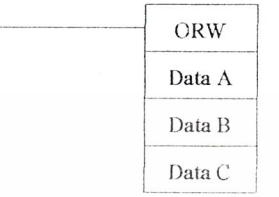
Fun (33) [9 byte]



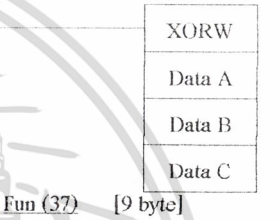
Fun (34) [9 byte]



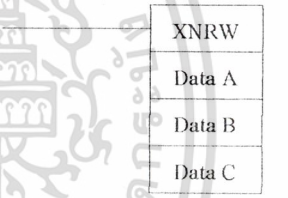
Fun (35) [9 byte]



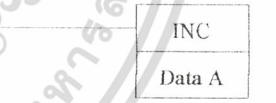
Fun (36) [9 byte]



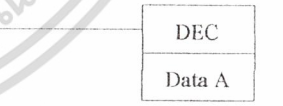
Fun (37) [9 byte]



Fun (38) [3 byte]

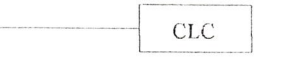


Fun (39) [3 byte]

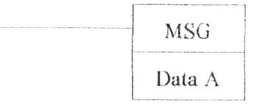


Fun (40) [3 byte]

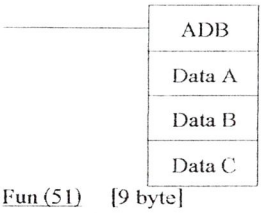
Fun (41) [3 byte]



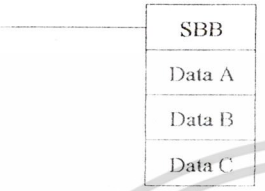
Fun (47) [3 byte]



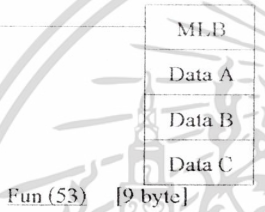
Fun (50) [9 byte]



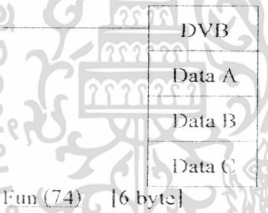
Fun (51) [9 byte]



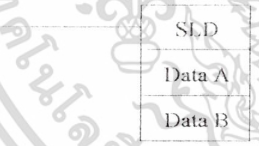
Fun (52) [9 byte]



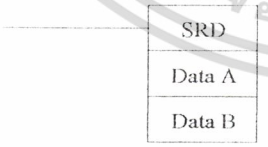
Fun (53) [9 byte]



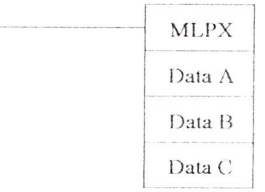
Fun (74) [6 byte]



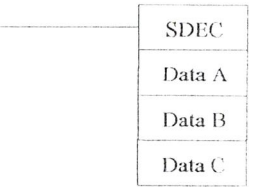
Fun (75) [6 byte]



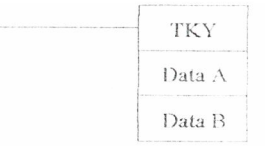
Fun (76) [9 byte]



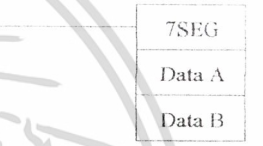
Fun (78) [9 byte]



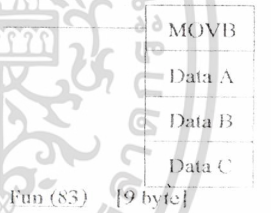
Fun (79) [6 byte]



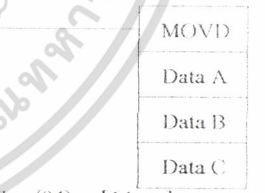
Fun (80) [6 byte]



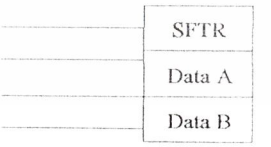
Fun (82) [9 byte]



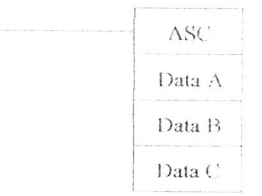
Fun (83) [9 byte]



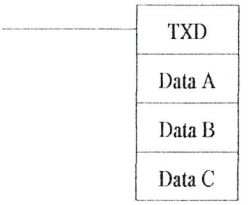
Fun (84) [6 byte]



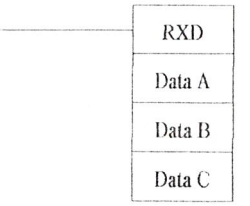
Fun (86) [9 byte]



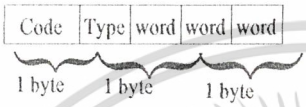
Fun (87) [9 byte]



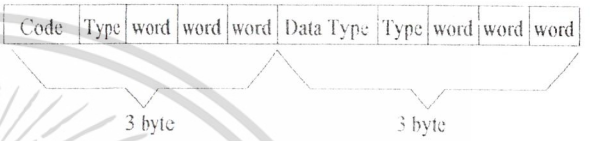
Fun (88) [9 byte]



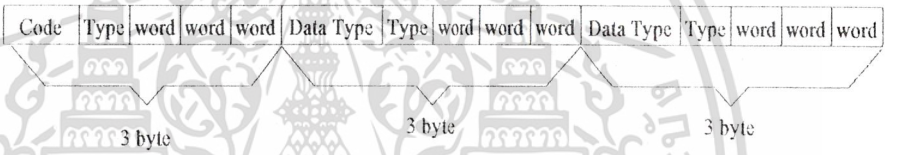
3 byte



6 byte



9 byte



Type

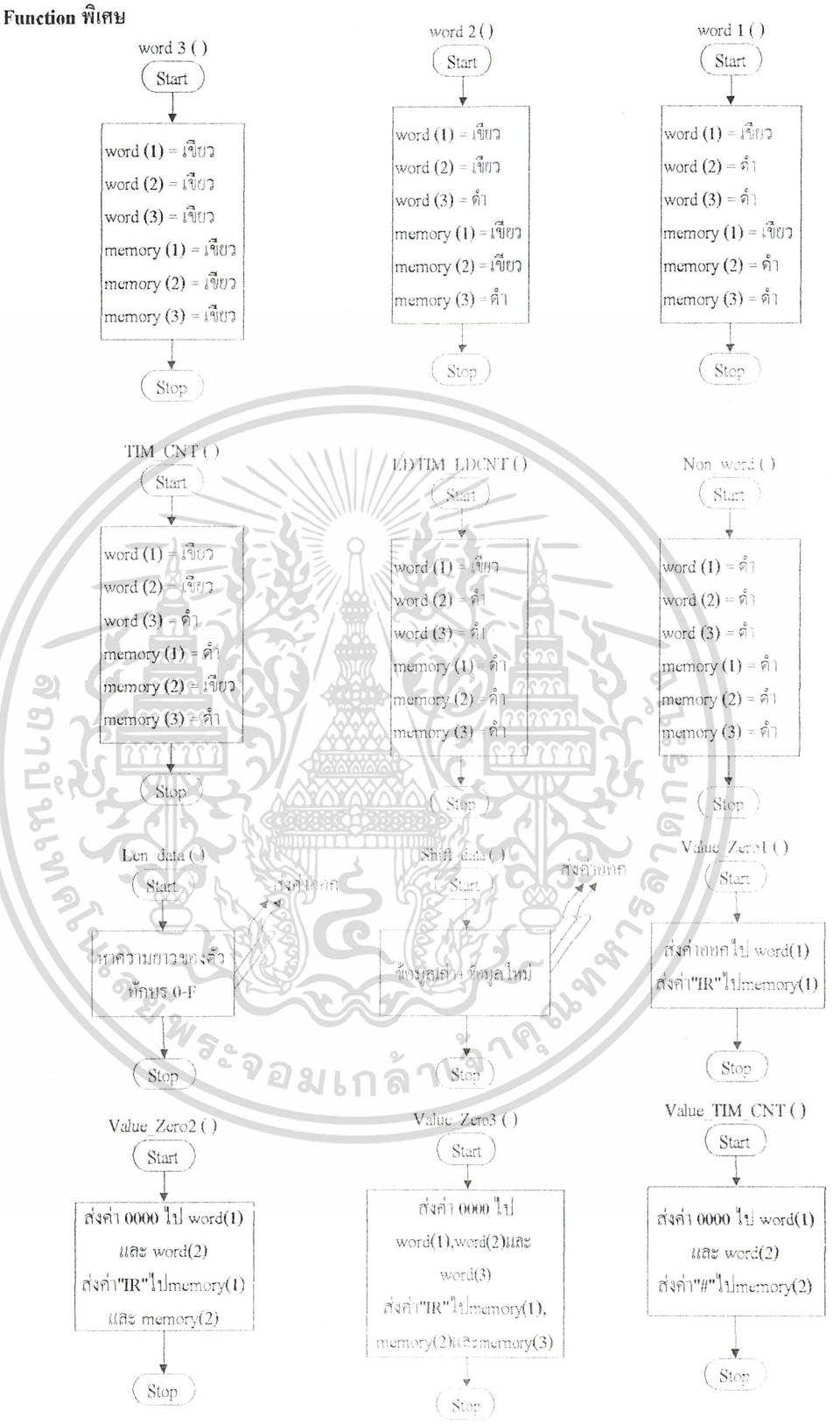
- A = IR
- B = DM
- C = LR
- D = HR
- E = T/C
- F = #

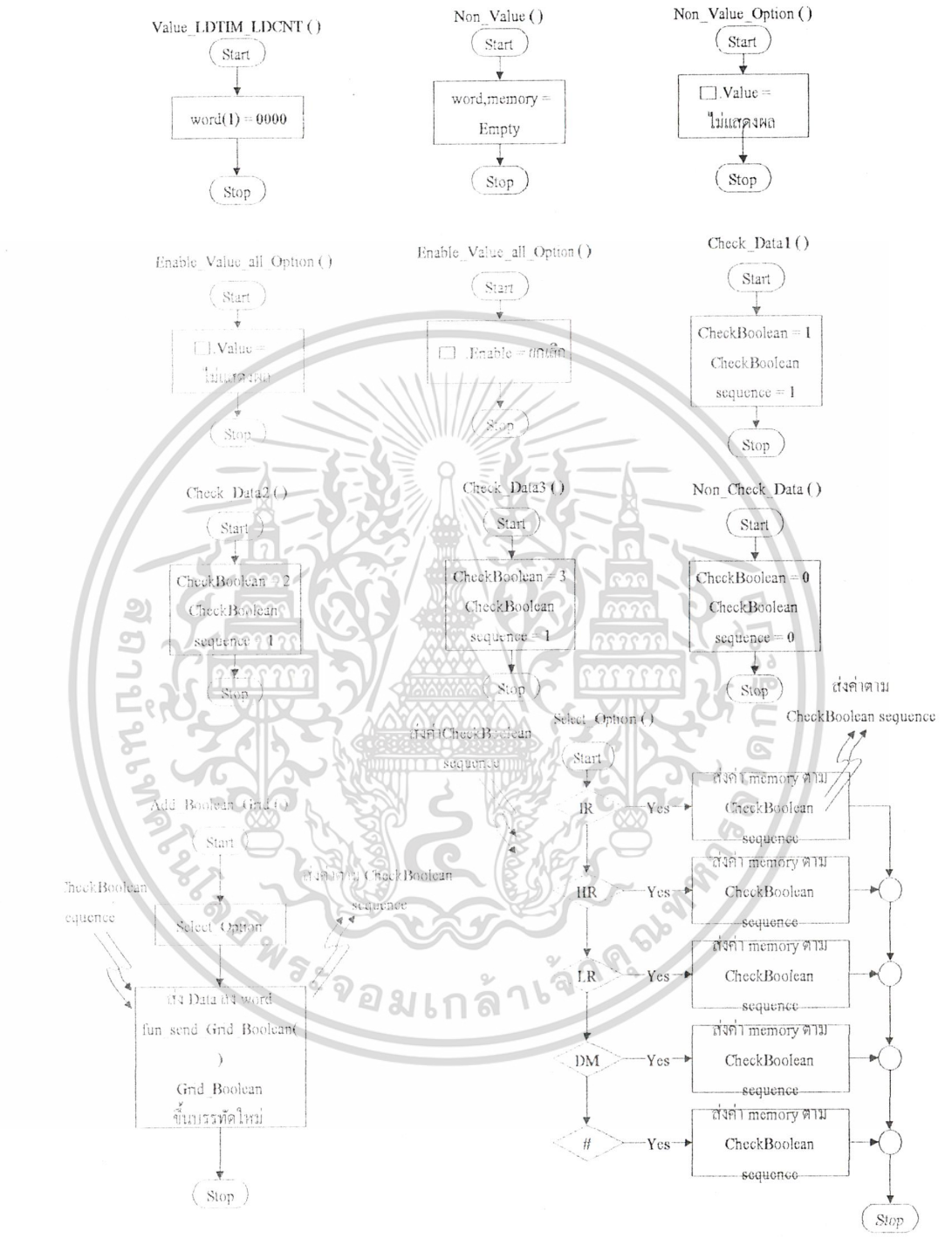
Data Type

- Data B = 74
- Data C = 75
- Data TIM = 76
- Data CNT = 77
- Data # = 78

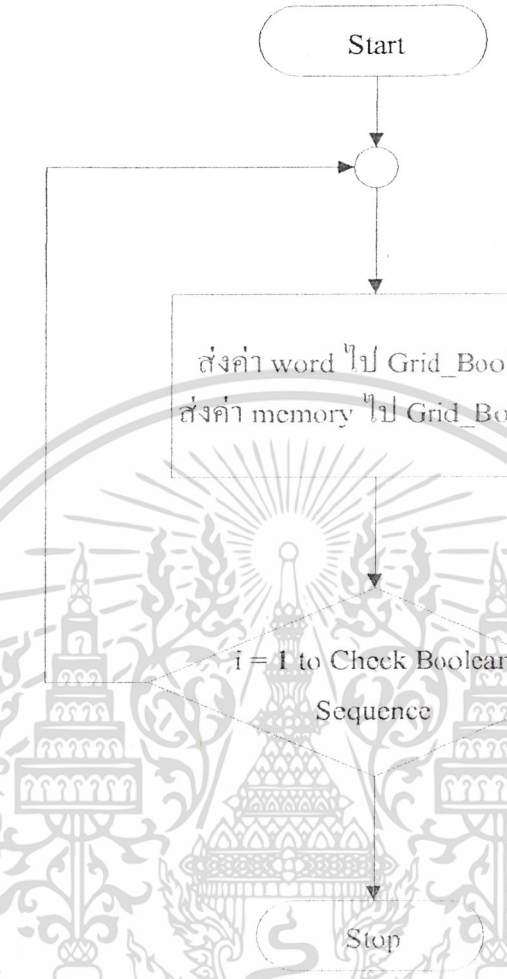
โครงสร้างโปรแกรม

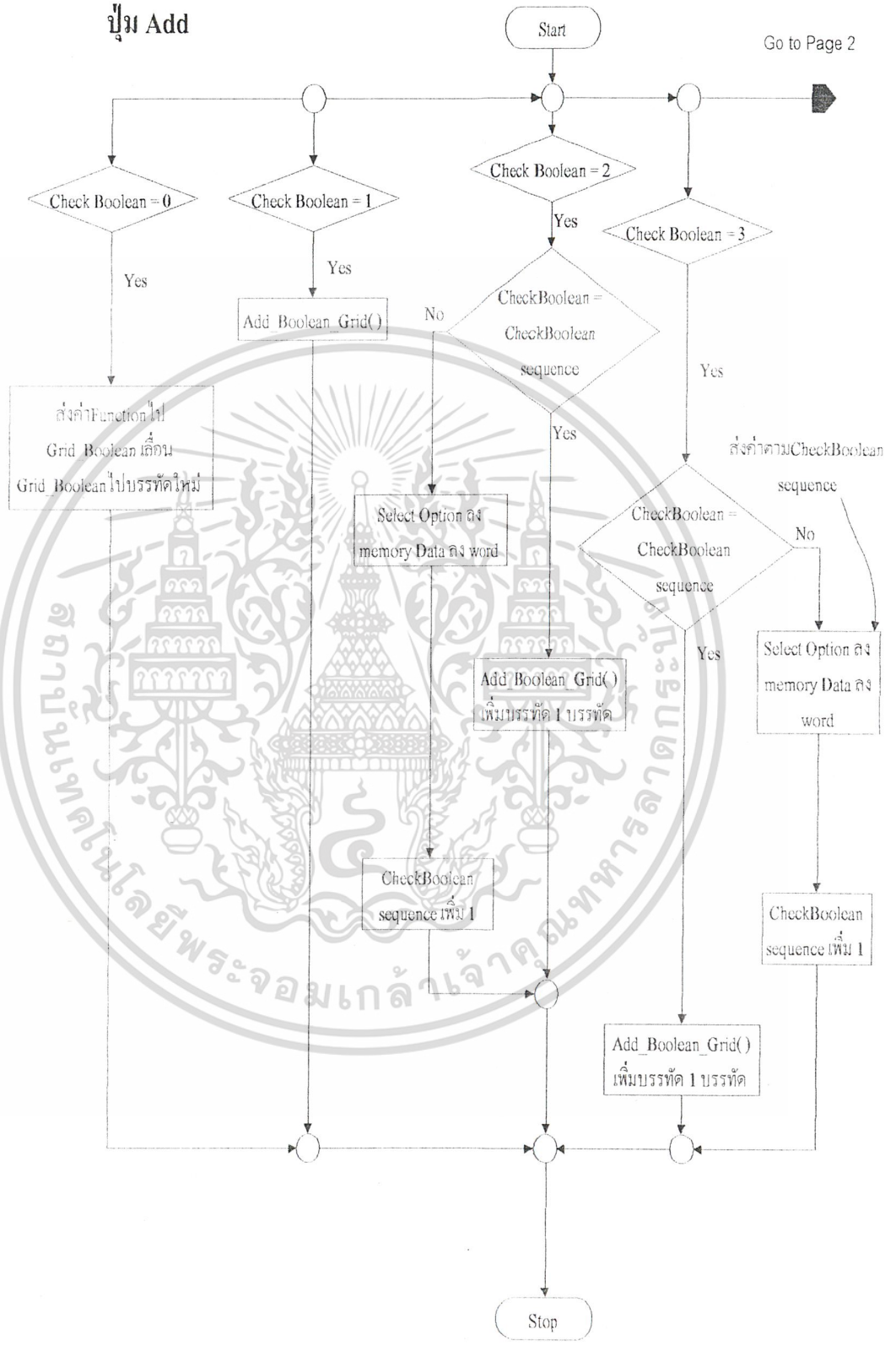
Function พิเศษ

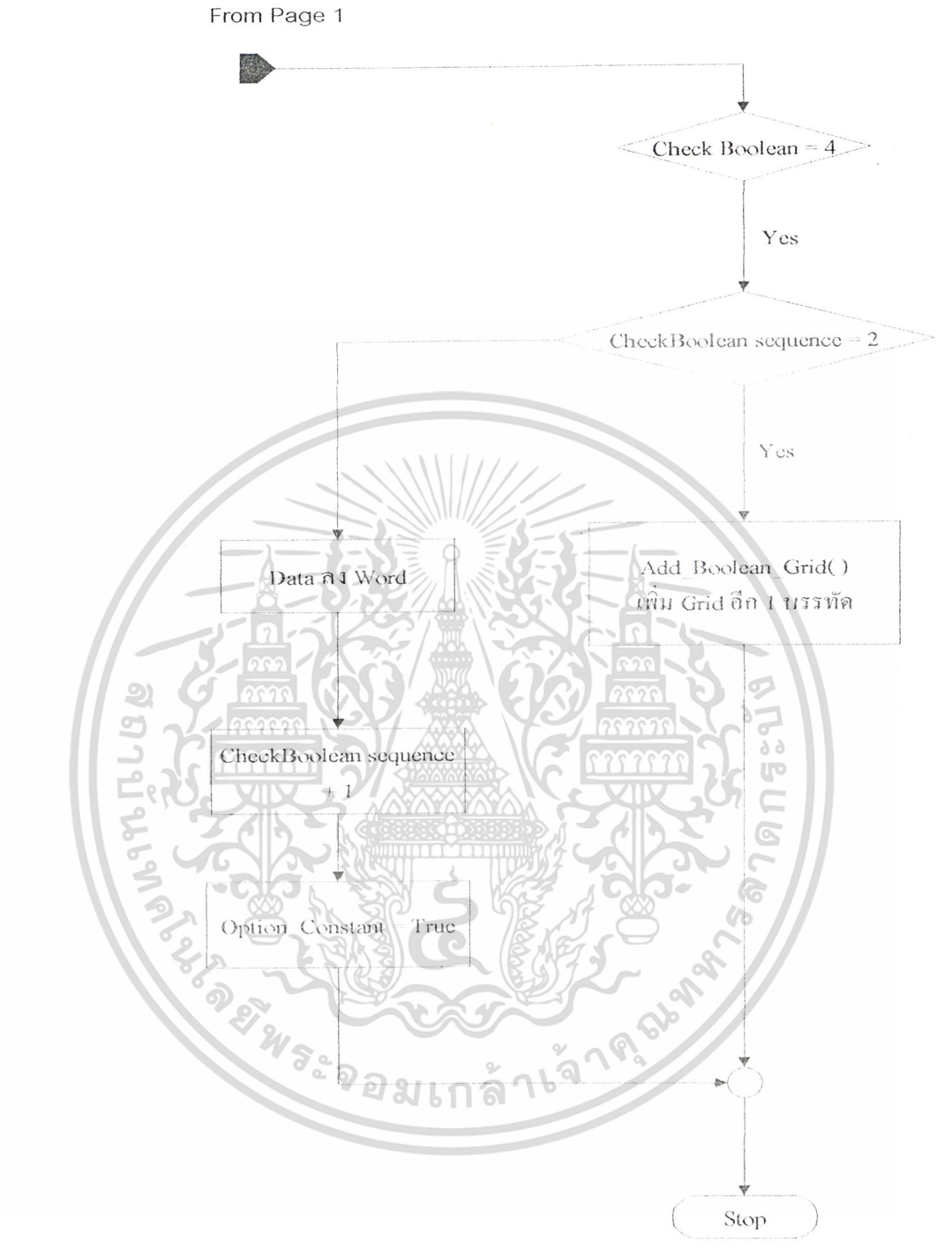


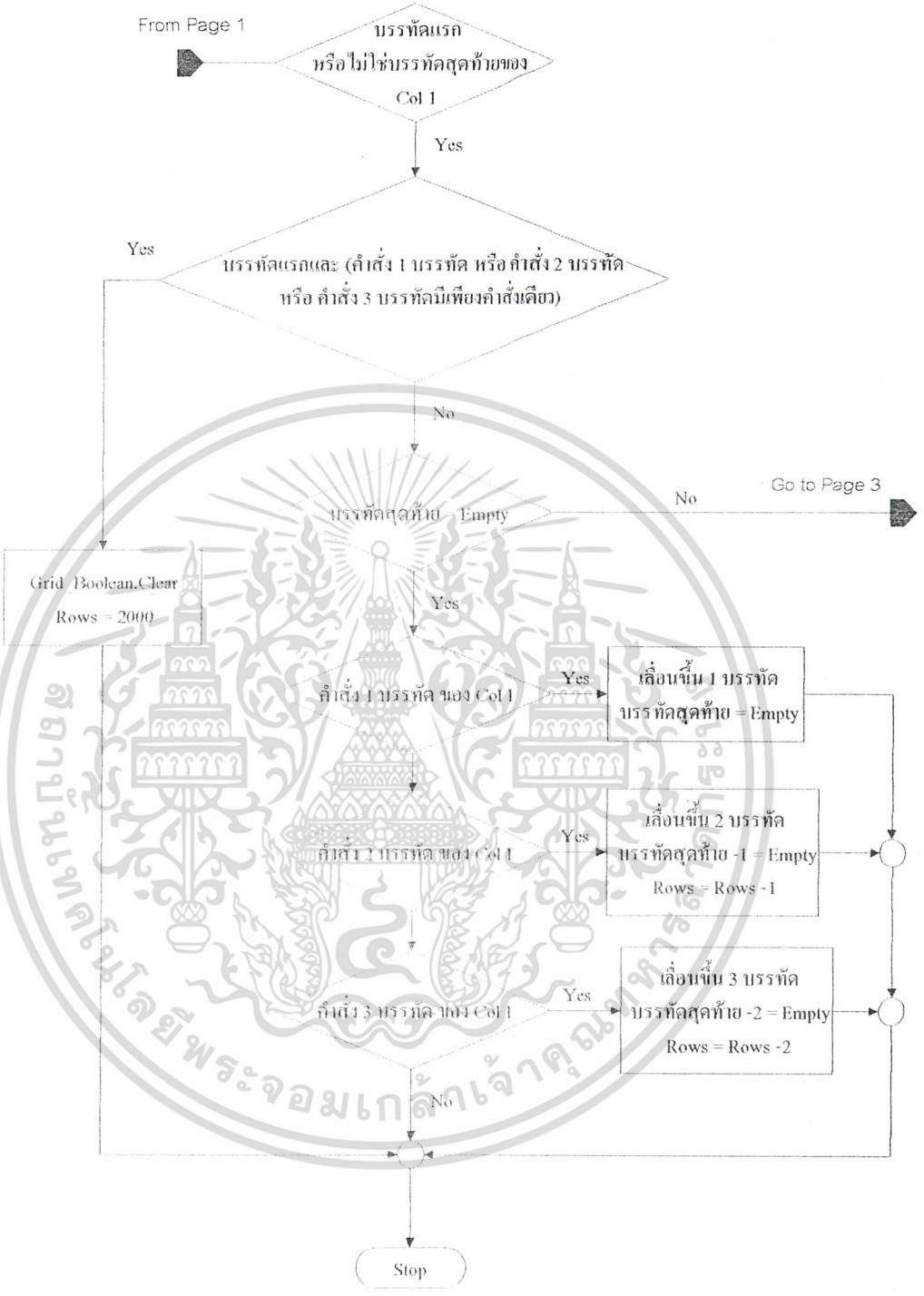


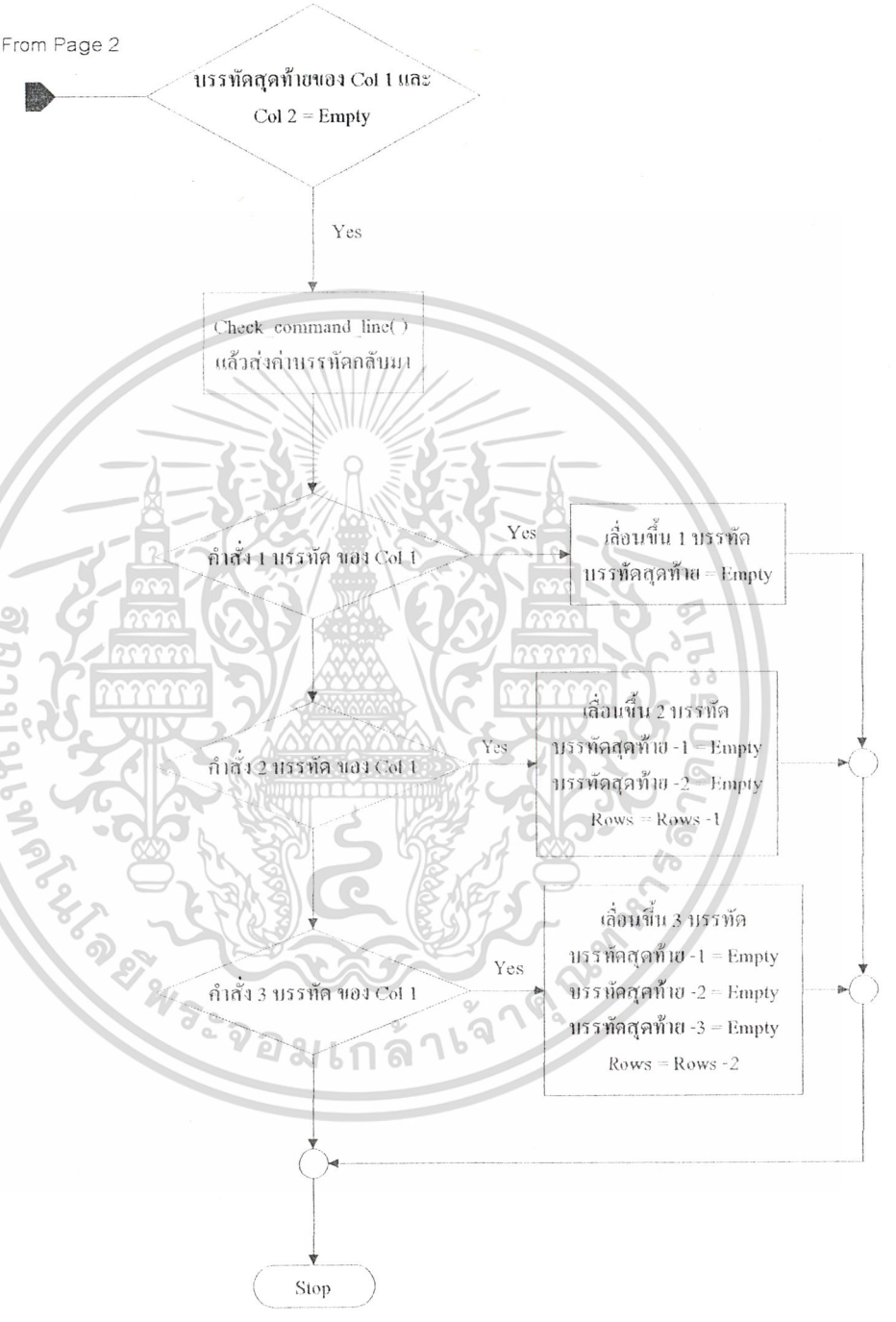
fun_send_Grid_Boolean ( )



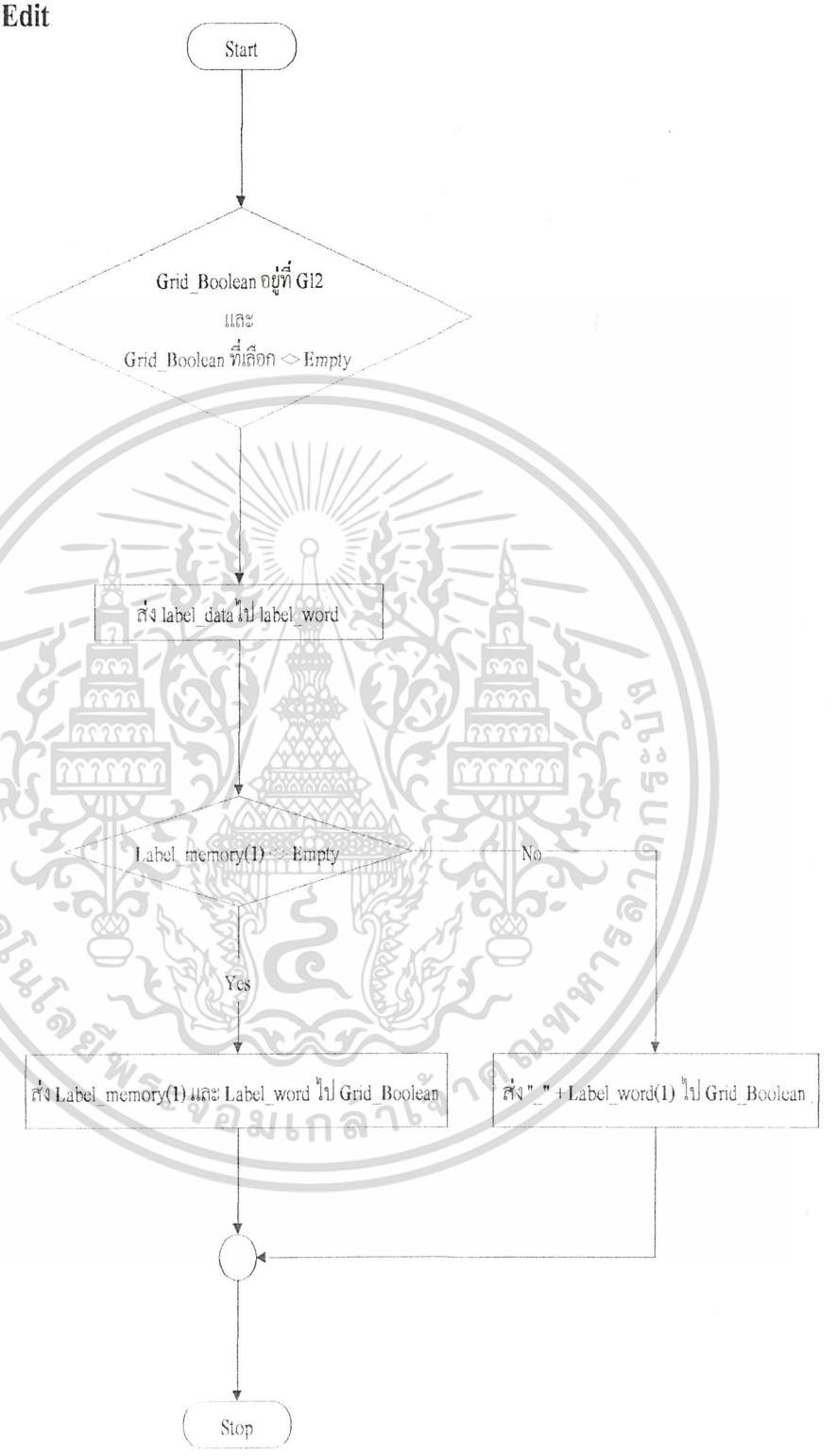






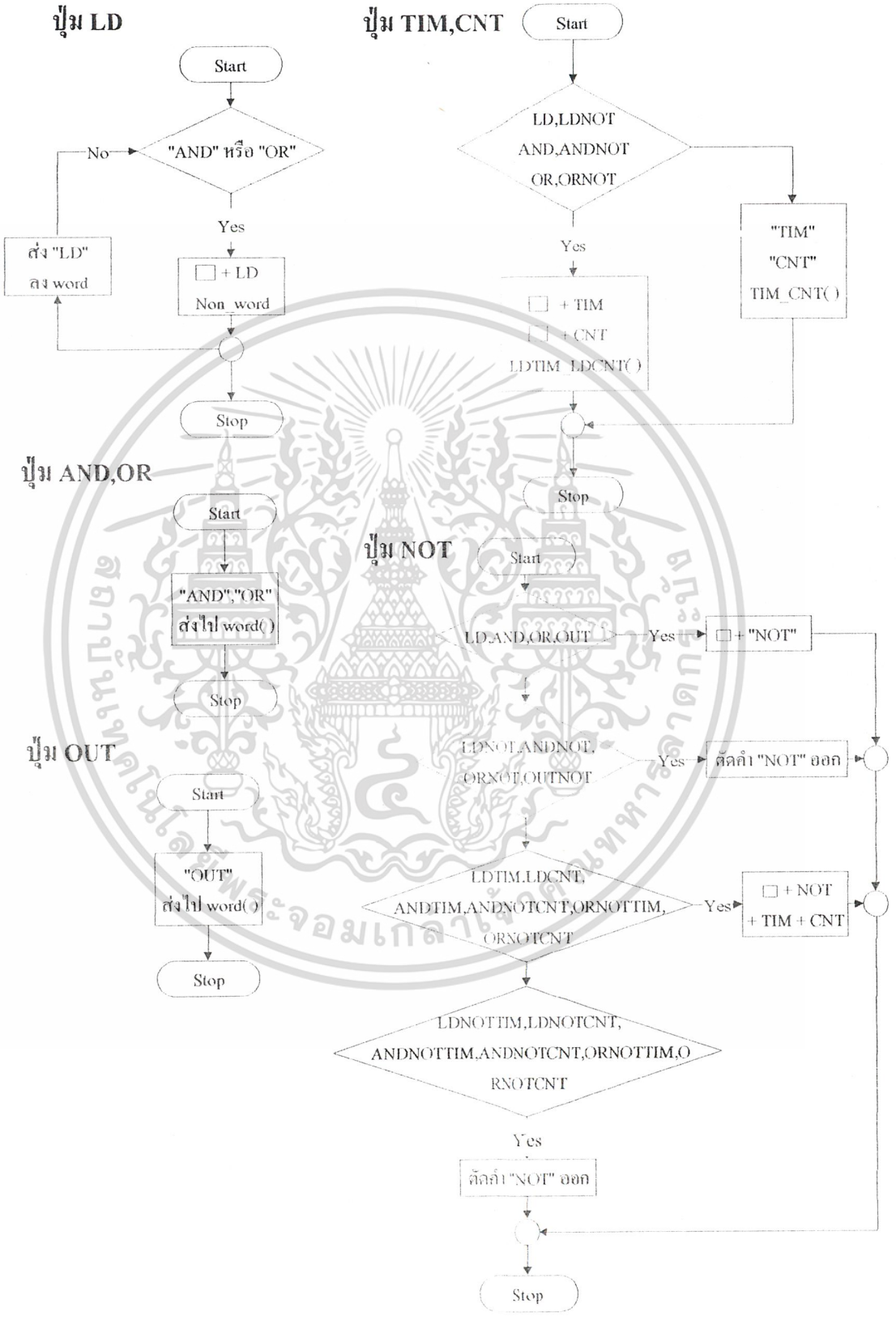


ปุ่ม Edit

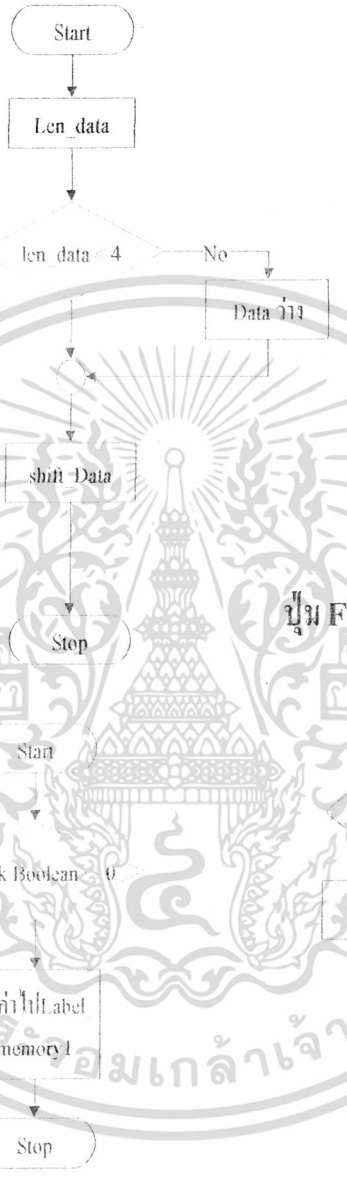


ปุ่ม Insert

□ = คำสั่ง



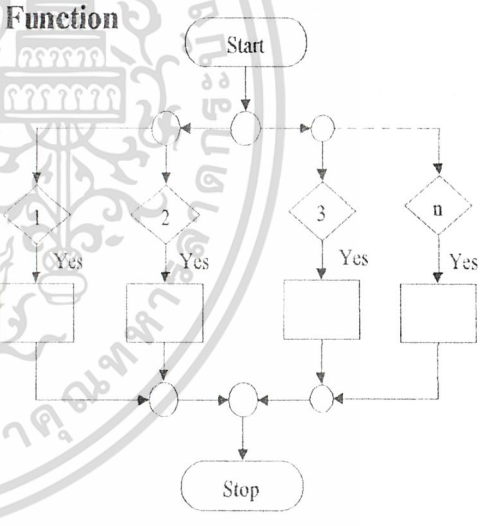
ปุ่ม ตัวเลข



ปุ่ม Option

IR,LR,HR,DM,#

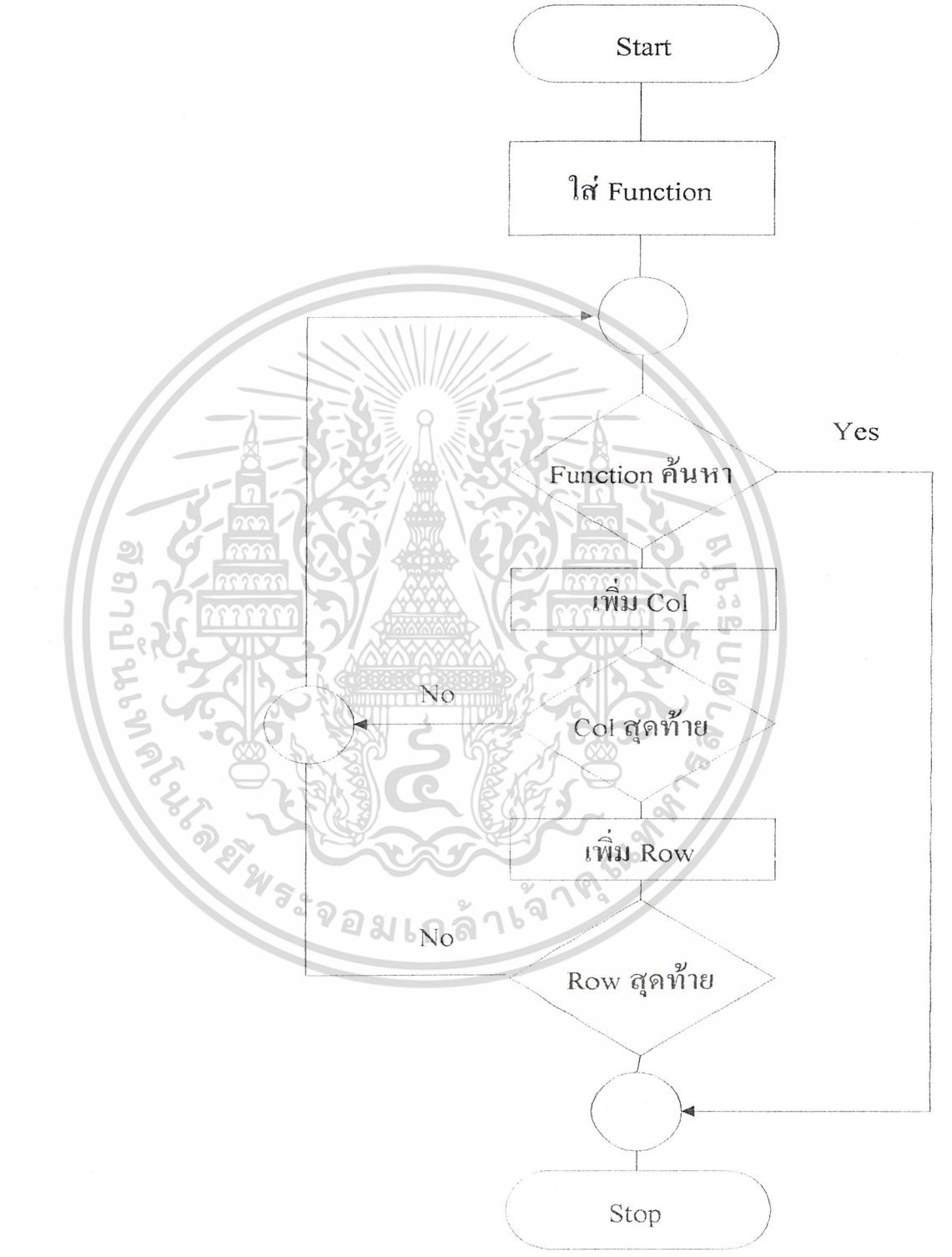
ปุ่ม Function



ปุ่ม Clear

Find_Boolean()

Find_Ladder()



โปรแกรมแปลงตาราง
Grid_Boolean เป็นคำสั่ง
ฐาน16

```
Dim checkmemory As Byte ' check sequence in text type memory
Private Sub tim_cnt_ladder_place()
mainprogram.Grid_Ladder.row = mainprogram.Grid_Ladder.row + 1
Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell(4)
mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = Text_ladder_word(1).Text
mainprogram.Grid_Ladder.CellAlignment = 4
If mainprogram.Button_ladder_Tim.Value = True Then
   Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col, 31)
'code TIMER in grid ladder
ElseIf mainprogram.Button_ladder_Cnt.Value = True Then
   Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col, 31)
'code COUNTER in grid ladder
End If
mainprogram.Grid_Ladder.row = mainprogram.Grid_Ladder.row + 1
Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell(10)
mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "#" + Text_ladder_word(2).Text
mainprogram.Grid_Ladder.CellAlignment = 4
If mainprogram.Button_ladder_Tim.Value = True Then
   Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col, 32)
'code TIMER in grid ladder
ElseIf mainprogram.Button_ladder_Cnt.Value = True Then
   Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col, 32)
'code COUNTER in grid ladder
End If
End Sub
-----------------------------------------------------------------------------------------------------------
Private Sub ladder1_place()
mainprogram.Grid_Ladder.row = mainprogram.Grid_Ladder.row + 1
Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell(10)
```

```
mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = Text_ladder_memory(2).Text + Text_ladder_word(2).Text
mainprogram.Grid_Ladder.CellAlignment = 4
End Sub
```

```
Private Sub ladder2_place()
mainprogram.Grid_Ladder.row = mainprogram.Grid_Ladder.row + 1
Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell(4)
mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = Text_ladder_memory(1).Text + Text_ladder_word(1).Text
mainprogram.Grid_Ladder.CellAlignment = 4
mainprogram.Grid_Ladder.row = mainprogram.Grid_Ladder.row + 1
Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell(10)
mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = Text_ladder_memory(2).Text + Text_ladder_word(2).Text
mainprogram.Grid_Ladder.CellAlignment = 4
End Sub
```

```
Private Sub ladder3_place()
mainprogram.Grid_Ladder.row = mainprogram.Grid_Ladder.row + 1
Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell(4)
mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = Text_ladder_memory(1).Text + Text_ladder_word(1).Text
mainprogram.Grid_Ladder.CellAlignment = 4
mainprogram.Grid_Ladder.row = mainprogram.Grid_Ladder.row + 1
Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell(7)
mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = Text_ladder_memory(2).Text + Text_ladder_word(2).Text
mainprogram.Grid_Ladder.CellAlignment = 4
mainprogram.Grid_Ladder.row = mainprogram.Grid_Ladder.row + 1
Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell(10)
```

```
mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = Text_ladder_memory(3).Text + Text_ladder_word(3).Text
mainprogram.Grid_Ladder.CellAlignment = 4
End Sub
---
Private Sub button_ladder_ok_Click()
If mainprogram.Button_ladder_LD.Value = True Then 'button ld
  Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder1.GraphicCell(1)
  mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = Text_ladder_memory(1).Text + Text_ladder_word(1).Text +
"." + Text_ladder_bit(1).Text
  mainprogram.Grid_Ladder.CellAlignment = 3
  Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H40) 'code Input in grid ladder
ElseIf mainprogram.Button_ladder_Out.Value = True Then 'button out
  Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder3.GraphicCell(5)
  mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = Text_ladder_memory(1).Text + Text_ladder_word(1).Text +
"." + Text_ladder_bit(1).Text
  mainprogram.Grid_Ladder.CellAlignment = 3
  Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H43) 'code Output in grid ladder
ElseIf mainprogram.Button_ladder_Tim.Value = True Then 'button tim
  Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell(1)
  mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "TIM"
  mainprogram.Grid_Ladder.CellAlignment = 4
  Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H8E) 'code TIMER in grid ladder
  Call tim_cnt_ladder_place
ElseIf mainprogram.Button_ladder_Cnt.Value = True Then 'button cnt
  Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell(1)
```

```
    mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
  mainprogram.Grid_Ladder.col) = "CNT"
    mainprogram.Grid_Ladder.CellAlignment = 4
    Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
  &H8F) 'code COUNTER in grid ladder
    Call tim_cnt_ladder_place
  ElseIf mainprogram.checkladder = True Then
    If mainprogram.Combo_ladder.ListIndex = 1 Then
      Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(0)
      mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
  mainprogram.Grid_Ladder.col) = "END(01)"
      mainprogram.Grid_Ladder.CellAlignment = 4
      Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
  &H1) 'code FUNCTION END(01) in grid ladder
    ElseIf mainprogram.Combo_ladder.ListIndex = 2 Then
      Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(0)
      mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
  mainprogram.Grid_Ladder.col) = "IL(02)"
      mainprogram.Grid_Ladder.CellAlignment = 4
      Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
  &H6) 'code FUNCTION IL(02) in grid ladder
    ElseIf mainprogram.Combo_ladder.ListIndex = 3 Then
      Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(0)
      mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
  mainprogram.Grid_Ladder.col) = "ILC(03)"
      mainprogram.Grid_Ladder.CellAlignment = 4
      Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
  &H7) 'code FUNCTION ILC(03) in grid ladder
    ElseIf mainprogram.Combo_ladder.ListIndex = 4 Then
```

```
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(0)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "JMP(04)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H8) 'code FUNCTION JMP(04) in grid ladder
    ElseIf mainprogram.Combo_ladder.ListIndex = 5 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(0)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "JME(05)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H8) 'code FUNCTION JME(05) in grid ladder
    ElseIf mainprogram.Combo_ladder.ListIndex = 6 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "SFT(10)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H80) 'code FUNCTION SFT(10) in grid ladder
        Call ladder2_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 7 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "KEEP(11)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H48) 'code FUNCTION KEEP(11) in grid ladder
```

```
        Call ladder1_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 8 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "DIFU(13)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H49) 'code FUNCTION DIFU(13) in grid ladder
        Call ladder1_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 9 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "DIFD(14)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H4A) 'code FUNCTION DIFD(14) in grid ladder
        Call ladder1_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 10 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "WSFT(16)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H81) 'code FUNCTION WSFT(16) in grid ladder
        Call ladder2_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 11 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
```

```
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "CMP(20)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H81) 'code FUNCTION CMP(20) in grid ladder
        Call ladder2_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 12 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "MOV(21)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H82) 'code FUNCTION MOV(21) in grid ladder
        Call ladder2_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 13 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "MVN(22)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H82) 'code FUNCTION MVN(22) in grid ladder
        Call ladder2_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 14 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "BIN(23)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H85) 'code FUNCTION BIN(23) in grid ladder
```

```
        Call ladder2_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 15 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "BCD(24)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H86) 'code FUNCTION BCD(24) in grid ladder
        Call ladder2_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 16 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "ASL(25)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H70) 'code FUNCTION ASL(25) in grid ladder
        Call ladder1_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 17 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "ASR(26)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H71) 'code FUNCTION ASR(26) in grid ladder
        Call ladder1_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 18 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
```

```
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
 mainprogram.Grid_Ladder.col) = "ROL(27)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
 &H72) 'code FUNCTION ROL(27) in grid ladder
        Call ladder1_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 19 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "ROR(28)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H73) 'code FUNCTION ROR(28) in grid ladder
        Call ladder1_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 20 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "COM(29)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H6D) 'code FUNCTION COM(29) in grid ladder
        Call ladder1_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 21 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "ADD(30)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HC0) 'code FUNCTION ADD(30) in grid ladder
```

```
        Call ladder3_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 22 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "SUB(31)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HC1) 'code FUNCTION SUB(31) in grid ladder
        Call ladder3_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 23 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "MUL(32)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HC2) 'code FUNCTION MUL(32) in grid ladder
        Call ladder3_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 24 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "DIV(33)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HC3) 'code FUNCTION DIV(33) in grid ladder
        Call ladder3_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 25 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
```

```
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
 mainprogram.Grid_Ladder.col) = "ANDW(34)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
 &HC4) 'code FUNCTION ANDW(34) in grid ladder
        Call ladder3_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 26 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "ORW(35)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HC5) 'code FUNCTION ORW(35) in grid ladder
        Call ladder3_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 27 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "XORW(36)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HC6) 'code FUNCTION XORW(36) in grid ladder
        Call ladder3_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 28 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "XNRW(37)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HC7) 'code FUNCTION XNRW(37) in grid ladder
```

```
        Call ladder3_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 29 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "INC(38)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HCE) 'code FUNCTION INC(38) in grid ladder
        Call ladder1_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 30 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "DEC(39)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HCF) 'code FUNCTION DEC(39) in grid ladder
        Call ladder1_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 31 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(0)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "STC(40)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H2) 'code FUNCTION STC(40) in grid ladder
    ElseIf mainprogram.Combo_ladder.ListIndex = 32 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(0)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "CLC(41)"
```

```
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H2) 'code FUNCTION CLC(41) in grid ladder
    ElseIf mainprogram.Combo_ladder.ListIndex = 33 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "MSG(47)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H79) 'code FUNCTION MSG(47) in grid ladder
        Call ladder1_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 34 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "ADB(50)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HC9) 'code FUNCTION AEB(50) in grid ladder
        Call ladder3_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 35 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "SBB(51)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HCA) 'code FUNCTION SBB(51) in grid ladder
        Call ladder3_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 36 Then
```

```
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "MLB(52)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HCB) 'code FUNCTION MLB(52) in grid ladder
        Call ladder3_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 37 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "DVB(53)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HCC) 'code FUNCTION DVB(53) in grid ladder
        Call ladder3_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 38 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "RDI(70)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H87) 'code FUNCTION RDI(70) in grid ladder
        Call ladder2_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 39 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "WRO(71)"
        mainprogram.Grid_Ladder.CellAlignment = 4
```

```
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H88) 'code FUNCTION WRO(71) in grid ladder
        Call ladder2_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 40 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "SLD(74)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H89) 'code FUNCTION SLD(74) in grid ladder
        Call ladder2_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 41 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "SRD(75)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H8A) 'code FUNCTION SRD(75) in grid ladder
        Call ladder2_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 42 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "MLPX(76)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HC8) 'code FUNCTION MLPX(76) in grid ladder
        Call ladder3_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 43 Then
```

```
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "SDEC(78)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HD0) 'code FUNCTION SDEC(78) in grid ladder
        Call ladder2_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 44 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "TKY(79)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H8C) 'code FUNCTION TKY(79) in grid ladder
        Call ladder2_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 45 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "7SEG(80)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HD0) 'code FUNCTION 7SEG(80) in grid ladder
        Call ladder2_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 46 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "MOVB(82)"
        mainprogram.Grid_Ladder.CellAlignment = 4
```

```
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HCD) 'code FUNCTION MOVB(82) in grid ladder
        Call ladder3_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 47 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "MOVD(83)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HCE) 'code FUNCTION MOVD(83) in grid ladder
        Call ladder3_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 48 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(2)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "SFTR(84)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&H8B) 'code FUNCTION SFTR(84) in grid ladder
        mainprogram.Grid_Ladder.row = mainprogram.Grid_Ladder.row + 1
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(5)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = Text_ladder_memory(1).Text + Text_ladder_word(1).Text
        mainprogram.Grid_Ladder.CellAlignment = 4
        mainprogram.Grid_Ladder.row = mainprogram.Grid_Ladder.row + 1
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(8)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = Text_ladder_memory(2).Text + Text_ladder_word(2).Text
        mainprogram.Grid_Ladder.CellAlignment = 4
```

```
        mainprogram.Grid_Ladder.row = mainprogram.Grid_Ladder.row + 1
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(11)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = Text_ladder_memory(3).Text + Text_ladder_word(3).Text
        mainprogram.Grid_Ladder.CellAlignment = 4
    ElseIf mainprogram.Combo_ladder.ListIndex = 49 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "ASC(86)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HCF) 'code FUNCTION ASC(86) in grid ladder
        Call ladder3_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 50 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "TXD(87)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HD1) 'code FUNCTION TXD(87) in grid ladder
        Call ladder3_place
    ElseIf mainprogram.Combo_ladder.ListIndex = 51 Then
        Set mainprogram.Grid_Ladder.CellPicture = mainprogram.pictureclip_ladder4.GraphicCell
(1)
        mainprogram.Grid_Ladder.TextMatrix(mainprogram.Grid_Ladder.row,
mainprogram.Grid_Ladder.col) = "RXD(88)"
        mainprogram.Grid_Ladder.CellAlignment = 4
        Call mainprogram.scan(mainprogram.Grid_Ladder.row, mainprogram.Grid_Ladder.col,
&HD2) 'code FUNCTION RXD(88) in grid ladder
```

```
        Call ladder3_place
    End If
End If
function_ladder.Hide
End Sub
```

```
Private Sub Command_ladder_cancel_Click()
function_ladder.Hide
mainprogram.checkladder = False
End Sub
```

```
Private Sub Command_ladder_default_Click()
Text_ladder_memory(1).Text = "I"
Text_ladder_word(1).Text = "000"
Text_ladder_bit(1).Text = "00"
Text_ladder_memory(2).Text = "I"
Text_ladder_word(2).Text = "000"
Text_ladder_bit(2).Text = "00"
Text_ladder_memory(3).Text = "I"
Text_ladder_word(3).Text = "000"
Text_ladder_bit(3).Text = "00"
End Sub
```

```
Private Sub Form_Load()
Text_ladder_memory(1).Text = "I"
Text_ladder_word(1).Text = "000"
Text_ladder_bit(1).Text = "00"
Text_ladder_memory(2).Text = "I"
Text_ladder_word(2).Text = "000"
Text_ladder_bit(2).Text = "00"
Text_ladder_memory(3).Text = "I"
Text_ladder_word(3).Text = "000"
```

```
Text_ladder_bit(3).Text = "00"
checkmemory = 1
End Sub
---
Private Sub Form_Unload(Cancel As Integer)
mainprogram.checkladder = False
End Sub
---
Private Sub Option_function_ladder_cnt_Click()
Text_ladder_memory(checkmemory).Text = "C"
End Sub
---
Private Sub Option_function_ladder_constant_Click()
Text_ladder_memory(checkmemory).Text = "#"
End Sub
---
Private Sub Option_function_ladder_dm_Click()
Text_ladder_memory(checkmemory).Text = "D"
End Sub
---
Private Sub Option_function_ladder_hr_Click()
Text_ladder_memory(checkmemory).Text = "H"
End Sub
---
Private Sub Option_function_ladder_ir_Click()
Text_ladder_memory(checkmemory).Text = "I"
End Sub
---
Private Sub Option_function_ladder_lr_Click()
Text_ladder_memory(checkmemory).Text = "L"
End Sub
```

```
Private Sub Option_function_ladder_tim_Click()
Text_ladder_memory(checkmemory).Text = "T"
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub Text_ladder_bit_Change(Index As Integer)
Text_ladder_bit(Index).SelStart = Len(Text_ladder_bit(Index).Text)
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub Text_ladder_memory_Click(Index As Integer)
checkmemory = Index
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub Text_ladder_memory_KeyUp(Index As Integer, KeyCode As Integer, Shift As
Integer)
If UCase(Text_ladder_memory(Index).Text) = "H" Then
   Option_function_ladder_hr.Value = True
   Text_ladder_memory(Index).Text = "H"
ElseIf UCase(Text_ladder_memory(Index).Text) = "I" Then
   Option_function_ladder_ir.Value = True
   Text_ladder_memory(Index).Text = "I"
ElseIf UCase(Text_ladder_memory(Index).Text) = "L" Then
   Option_function_ladder_lr.Value = True
   Text_ladder_memory(Index).Text = "L"
ElseIf UCase(Text_ladder_memory(Index).Text) = "D" Then
   Option_function_ladder_dm.Value = True
   Text_ladder_memory(Index).Text = "D"
ElseIf UCase(Text_ladder_memory(Index).Text) = "#" Then
   Option_function_ladder_constant.Value = True
   Text_ladder_memory(Index).Text = "#"
ElseIf (UCase(Text_ladder_memory(1).Text) = "T") And (Option_function_ladder_tim.Enabled
= True) Then
   Option_function_ladder_tim.Value = True
```

```
    Text_ladder_memory(1).Text = "T"
ElseIf (UCase(Text_ladder_memory(1).Text) = "C") And (Option_function_ladder_cnt.Enabled
= True) Then
    Option_function_ladder_cnt.Value = True
    Text_ladder_memory(Index).Text = "C"
ElseIf Text_ladder_memory(Index).Text <> Empty Then
    Text_ladder_memory(Index).Text = Empty
End If
Text_ladder_memory(Index).SelStart = 1
End Sub
-------------------------------------------------------------------------------------------

Private Sub Text_ladder_word_Change(Index As Integer)
Text_ladder_word(Index).SelStart = Len(Text_ladder_word(Index).Text)
End Sub
-------------------------------------------------------------------------------------------
```

```
Dim booleancheck As Boolean
Dim textfind As String
---------------------------------------------------------------------------------------------------------------
Private Sub button_boolean_cancel_Click()
find_booleann.Visible = False
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub button_boolean_find_Click()
i = 1
Do
  If booleancheck = False Then
    If mainprogram.Grid_Boolean.TextMatrix(i, 1) = Text_find_boolean_function.Text Then
      If mainprogram.Grid_Boolean.TextMatrix(i, 2) = (Text_find_boolean_type.Text &
Text_find_boolean_word.Text) Then
        mainprogram.Grid_Boolean.row = i
        mainprogram.Grid_Boolean.col = 0
        mainprogram.Grid_Boolean.CellBackColor = &HFF&
      End If
    End If
  ElseIf booleancheck = True Then
    If mainprogram.Grid_Boolean.TextMatrix(i, 1) = textfind Then
      mainprogram.Grid_Boolean.row = i
      mainprogram.Grid_Boolean.col = 0
      mainprogram.Grid_Boolean.CellBackColor = &HFF&
    End If
  End If
  i = i + 1
Loop While i < mainprogram.Grid_Boolean.Rows - 1
mainprogram.checkfixcolor = True
End Sub
---------------------------------------------------------------------------------------------------------------
```

```
Private Sub Text_find_boolean_function_Change()
Text_find_boolean_function.Text = UCase(Text_find_boolean_function.Text)
Text_find_boolean_function.SelStart = Len(Text_find_boolean_function.Text)
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub Text_find_boolean_function_Click()
booleancheck = False
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub Text_find_boolean_type_Change()
Text_find_boolean_type.Text = UCase(Text_find_boolean_type.Text)
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub Text_find_boolean_type_Click()
booleancheck = False
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub Text_find_boolean_word_Click()
booleancheck = False
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub Combo_find_boolean_Click()
If Combo_find_boolean.ListIndex = 1 Then
   textfind = "END(01)"
ElseIf Combo_find_boolean.ListIndex = 2 Then
   textfind = "IL(02)"
ElseIf Combo_find_boolean.ListIndex = 3 Then
   textfind = "ILC(03)"
ElseIf Combo_find_boolean.ListIndex = 4 Then
   textfind = "JMP(04)"
ElseIf Combo_find_boolean.ListIndex = 5 Then
   textfind = "JME(05)"
```

```
ElseIf Combo_find_boolean.ListIndex = 6 Then
   textfind = "SFT(10)"
ElseIf Combo_find_boolean.ListIndex = 7 Then
   textfind = "KEEP(11)"
ElseIf Combo_find_boolean.ListIndex = 8 Then
   textfind = "DIFU(13)"
ElseIf Combo_find_boolean.ListIndex = 9 Then
   textfind = "DIFD(14)"
ElseIf Combo_find_boolean.ListIndex = 10 Then
   textfind = "WSFT(16)"
ElseIf Combo_find_boolean.ListIndex = 11 Then
   textfind = "CMP(20)"
ElseIf Combo_find_boolean.ListIndex = 12 Then
   textfind = "MOV(21)"
ElseIf Combo_find_boolean.ListIndex = 13 Then
   textfind = "MVN(22)"
ElseIf Combo_find_boolean.ListIndex = 14 Then
   textfind = "BIN(23)"
ElseIf Combo_find_boolean.ListIndex = 15 Then
   textfind = "BCD(24)"
ElseIf Combo_find_boolean.ListIndex = 16 Then
   textfind = "ASL(25)"
ElseIf Combo_find_boolean.ListIndex = 17 Then
   textfind = "ASR(26)"
ElseIf Combo_find_boolean.ListIndex = 18 Then
   textfind = "ROL(27)"
ElseIf Combo_find_boolean.ListIndex = 19 Then
   textfind = "ROR(28)"
ElseIf Combo_find_boolean.ListIndex = 20 Then
   textfind = "COM(29)"
ElseIf Combo_find_boolean.ListIndex = 21 Then
   textfind = "Add(30)"
```

```
ElseIf Combo_find_boolean.ListIndex = 22 Then
   textfind = "SUB(31)"
ElseIf Combo_find_boolean.ListIndex = 23 Then
   textfind = "MUL(32)"
ElseIf Combo_find_boolean.ListIndex = 24 Then
   textfind = "DIV(33)"
ElseIf Combo_find_boolean.ListIndex = 25 Then
   textfind = "ANDW(34)"
ElseIf Combo_find_boolean.ListIndex = 26 Then
   textfind = "ORW(35)"
ElseIf Combo_find_boolean.ListIndex = 27 Then
   textfind = "XORW(36)"
ElseIf Combo_find_boolean.ListIndex = 28 Then
   textfind = "XNRW(37)"
ElseIf Combo_find_boolean.ListIndex = 29 Then
   textfind = "INC(38)"
ElseIf Combo_find_boolean.ListIndex = 30 Then
   textfind = "DEC(39)"
ElseIf Combo_find_boolean.ListIndex = 31 Then
   textfind = "STC(40)"
ElseIf Combo_find_boolean.ListIndex = 32 Then
   textfind = "CLC(41)"
ElseIf Combo_find_boolean.ListIndex = 33 Then
   textfind = "MSG(47)"
ElseIf Combo_find_boolean.ListIndex = 34 Then
   textfind = "ADB(50)"
ElseIf Combo_find_boolean.ListIndex = 35 Then
   textfind = "SBB(51)"
ElseIf Combo_find_boolean.ListIndex = 36 Then
   textfind = "MLB(52)"
ElseIf Combo_find_boolean.ListIndex = 37 Then
   textfind = "DVB(53)"
```

```
ElseIf Combo_find_boolean.ListIndex = 38 Then
   textfind = "RDI(70)"
ElseIf Combo_find_boolean.ListIndex = 39 Then
   textfind = "WRO(71)"
ElseIf Combo_find_boolean.ListIndex = 40 Then
   textfind = "SLD(74)"
ElseIf Combo_find_boolean.ListIndex = 41 Then
   textfind = "SRD(75)"
ElseIf Combo_find_boolean.ListIndex = 42 Then
   textfind = "MLPX(76)"
ElseIf Combo_find_boolean.ListIndex = 43 Then
   textfind = "SDEC(78)"
ElseIf Combo_find_boolean.ListIndex = 44 Then
   textfind = "TKY(79)"
ElseIf Combo_find_boolean.ListIndex = 45 Then
   textfind = "7SEG(80)"
ElseIf Combo_find_boolean.ListIndex = 46 Then
   textfind = "MOVB(82)"
ElseIf Combo_find_boolean.ListIndex = 47 Then
   textfind = "MOVD(83)"
ElseIf Combo_find_boolean.ListIndex = 48 Then
   textfind = "SFTR(84)"
ElseIf Combo_find_boolean.ListIndex = 49 Then
   textfind = "ASC(86)"
ElseIf Combo_find_boolean.ListIndex = 50 Then
   textfind = "TXD(87)"
ElseIf Combo_find_boolean.ListIndex = 51 Then
   textfind = "RXD(88)"
End If
booleancheck = True
End Sub
```

```
Private Sub setting_cancel_Click()
communication_setting.Visible = False
End Sub
```

```
Private Sub setting_default_Click()
value_unit.Text = "0"
value_baud.Text = "9600"
value_data_bit.Text = "8"
value_parity_bit.Text = "none"
value_stop_bit.Text = "1"
value_com_port.Text = "1"
End Sub
```

```
Private Sub setting_ok_Click()
mainprogram.MSComm1.CommPort = value_com_port.Text
mainprogram.MSComm1.Settings = value_baud.Text + "," + Mid(value_parity_bit.Text, 1, 1) + "," +
value_data_bit.Text + "," + value_stop_bit.Text
communication_setting.Visible = False
End Sub
```

```
Dim lenfun As Integer 'lenght of data label
Dim databoolean As String 'data of tab boolean
Dim checkboolean As Integer 'check and count of grid_boolean
Dim checkbooleansequence As Integer 'sequence of label_word
Dim checkboolean_temp As Integer 'check and count of grid_boolean and temp
Dim checkbooleansequence_temp As Integer 'sequence of label_word and temp
Dim grid_row As Integer 'row position of grid boolean
Dim grid_col As Integer 'col position of grid boolean
Dim grid_rows As Integer 'rows number of grid boolean
Dim numbercommandline As Integer ' set number command line of model PLC
Dim codecommand(1 To 6000) As String 'HEX code of program PLC
Public checkfixcolor As Boolean ' check color when find is RED.
Public checkladder As Boolean 'check function ladder
Public rowladder As Integer 'Keep Value row of Grid_ladder
Public colladder As Integer 'Keep Value col of Grid_ladder
Dim bit_0 As Byte 'bit 0 of digit in memory
Dim bit_1 As Byte 'bit 1 of digit in memory
Dim bit_2 As Byte 'bit 2 of digit in memory
Dim bit_3 As Byte 'bit 3 of digit in memory
Dim rowscan As Integer 'keep row of gird ladder
Dim colscan As Integer 'keep col of gird ladder
'Dim checkscan As Boolean
Dim scanmap(0 To 51, 0 To 31) As Byte ' map for scan ladder code
Dim text_command As String ' text code recieve from PLC to host
Dim uploadcommand As String
```

```
Public Function scan(row As Integer, col As Integer, valuecode As Integer) 'keep code outside form
    scanmap(row, col) = valuecode
End Function
```

```
Private Function Fcs(word As String) As String ' Calculate FCS
Dim Length As Byte
Dim Cal As Integer
Dim i As Byte
For i = 1 To Len(word)
   Cal = Cal Xor Asc(Mid(word, i, 1))
Next i
Fcs = Cal
End Function
```

```
Private Function receive_command() 'receive command PLC to host
Do
   While MSComm1.InBufferCount = 0
   Wend
   text_command = text_command & MSComm1.Input
Loop Until Mid(text_command, Len(text_command), 1) = vbLf
End Function
```

```
Private Function delay_set(delay_value As Byte) 'minimum value = 10 mSec(0.01)
Start_Time = Timer
Do
Loop Until (Timer - Start_Time) > delay_value
End Function
```

```
Private Sub delay_base() 'base timer for send Program
For j = 1 To 500 ' Range 100-1000
Next j
End Sub
```

```
Private Sub word1() 'set color one word
   Label_word(1).BackColor = &HC0FFC0
   Label_word(2).BackColor = &H8000000F
```

```
    Label_word(3).BackColor = &H8000000F
    Label_memory(1).BackColor = &HC0FFC0
    Label_memory(2).BackColor = &H8000000F
    Label_memory(3).BackColor = &H8000000F
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub word2() 'set color two word
    Label_word(1).BackColor = &HC0FFC0
    Label_word(2).BackColor = &HC0FFC0
    Label_word(3).BackColor = &H8000000F
    Label_memory(1).BackColor = &HC0FFC0
    Label_memory(2).BackColor = &HC0FFC0
    Label_memory(3).BackColor = &H8000000F
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub word3() 'set color three word
    Label_word(1).BackColor = &HC0FFC0
    Label_word(2).BackColor = &HC0FFC0
    Label_word(3).BackColor = &HC0FFC0
    Label_memory(1).BackColor = &HC0FFC0
    Label_memory(2).BackColor = &HC0FFC0
    Label_memory(3).BackColor = &HC0FFC0
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub tim_cnt() 'set color timer and counter
    Label_word(1).BackColor = &HC0FFC0
    Label_word(2).BackColor = &HC0FFC0
    Label_word(3).BackColor = &H8000000F
    Label_memory(1).BackColor = &H8000000F
    Label_memory(2).BackColor = &HC0FFC0
    Label_memory(3).BackColor = &H8000000F
End Sub
```

```
Private Sub ldtim_ldcnt() 'set color LD timer and LD counter
    Label_word(1).BackColor = &HC0FFC0
    Label_word(2).BackColor = &H8000000F
    Label_word(3).BackColor = &H8000000F
    Label_memory(1).BackColor = &H8000000F
    Label_memory(2).BackColor = &H8000000F
    Label_memory(3).BackColor = &H8000000F
End Sub
```

```
Private Sub non_word() 'set color non word
    Label_word(1).BackColor = &H8000000F
    Label_word(2).BackColor = &H8000000F
    Label_word(3).BackColor = &H8000000F
    Label_memory(1).BackColor = &H8000000F
    Label_memory(2).BackColor = &H8000000F
    Label_memory(3).BackColor = &H8000000F
End Sub
```

```
Private Sub value_zero1()  'set caption one word
    Label_word(1).Caption = "0000"
    Label_word(2).Caption = Empty
    Label_word(3).Caption = Empty
    Label_memory(1).Caption = "I"
    Label_memory(2).Caption = Empty
    Label_memory(3).Caption = Empty
End Sub
```

```
Private Sub value_zero2() 'set caption two word
    Label_word(1).Caption = "0000"
    Label_word(2).Caption = "0000"
    Label_word(3).Caption = Empty
    Label_memory(1).Caption = "I"
```

```
    Label_memory(2).Caption = "I"
    Label_memory(3).Caption = Empty
End Sub
```

```
Private Sub value_zero3() 'set caption three word
    Label_word(1).Caption = "0000"
    Label_word(2).Caption = "0000"
    Label_word(3).Caption = "0000"
    Label_memory(1).Caption = "I"
    Label_memory(2).Caption = "I"
    Label_memory(3).Caption = "I"
End Sub
```

```
Private Sub value_tim_cnt() 'set caption timer and counter
    Label_word(1).Caption = "000"
    Label_word(2).Caption = "000"
    Label_word(3).Caption = Empty
    Label_memory(1).Caption = Empty
    Label_memory(2).Caption = "#"
    Label_memory(3).Caption = Empty
End Sub
```

```
Private Sub value_ldtim_ldcnt() 'set caption LD timer and LD counter
    Label_word(1).Caption = "000"
    Label_word(2).Caption = Empty
    Label_word(3).Caption = Empty
    Label_memory(1).Caption = Empty
    Label_memory(2).Caption = Empty
    Label_memory(3).Caption = Empty
End Sub
```

```
Private Sub non_value() 'set caption non word
    Label_word(1).Caption = Empty
    Label_word(2).Caption = Empty
    Label_word(3).Caption = Empty
    Label_memory(1).Caption = Empty
    Label_memory(2).Caption = Empty
    Label_memory(3).Caption = Empty
End Sub
```

```
Private Sub non_value_option() 'set non value of bottom option
    Option_hr.Value = False
    Option_ir.Value = False
    Option_lr.Value = False
    Option_dm.Value = False
    Option_constant.Value = False
End Sub
```

```
Private Sub disable_value_all_option() 'set disable all  of bottom option
    Option_hr.Enabled = False
    Option_ir.Enabled = False
    Option_lr.Enabled = False
    Option_dm.Enabled = False
    Option_constant.Enabled = False
End Sub
```

```
Private Sub enable_value_all_option() 'set enable all  of bottom option
    Option_hr.Enabled = True
    Option_ir.Enabled = True
    Option_lr.Enabled = True
    Option_dm.Enabled = True
    Option_constant.Enabled = True
End Sub
```

```
Private Sub check_data1() 'set one command line
    checkboolean = 1
    checkboolean_temp = checkboolean
    checkbooleansequence = 1
    checkbooleansequence_temp = checkbooleansequence
End Sub
-----------------------------------------------------------------------------------------------------------------------
Private Sub check_data2() 'set two command line
    checkboolean = 2
    checkboolean_temp = checkboolean
    checkbooleansequence = 1
    checkbooleansequence_temp = checkbooleansequence
End Sub
-----------------------------------------------------------------------------------------------------------------------
Private Sub check_data3() 'set three command line
    checkboolean = 3
    checkboolean_temp = checkboolean
    checkbooleansequence = 1
    checkbooleansequence_temp = checkbooleansequence
End Sub
-----------------------------------------------------------------------------------------------------------------------
Private Sub check_data4() 'set TIM/CNT command line
    checkboolean = 4
    checkboolean_temp = checkboolean
    checkbooleansequence = 1
    checkbooleansequence_temp = checkbooleansequence
End Sub
-----------------------------------------------------------------------------------------------------------------------
Private Sub non_check_data() 'set one command line when non valve
    checkboolean = 0
    checkboolean_temp = checkboolean
    checkbooleansequence = 0
```

```
    checkbooleansequence_temp = checkbooleansequence
End Sub
---------------------------------------------------------------------------------------------------------------------
Private Sub check_command_line() ' checking and display number command line Boolean
For i = 1 To Grid_Boolean.Rows - 1
    If Grid_Boolean.TextMatrix(i, 1) <> Empty Then
        numberline = numberline + 1
        Grid_Boolean.TextMatrix(i, 0) = numberline
    End If
Next i
End Sub
---------------------------------------------------------------------------------------------------------------------
Private Sub clear_command_line() ' clear number command line Boolean
For i = 1 To Grid_Boolean.Rows - 1
    Grid_Boolean.TextMatrix(i, 0) = Empty
Next i
End Sub
---------------------------------------------------------------------------------------------------------------------
Private Sub show_value_ladder1()
function_ladder.Text_ladder_memory(1).Visible = True
function_ladder.Text_ladder_word(1).Visible = True
function_ladder.Text_ladder_bit(1).Visible = True
function_ladder.Text_ladder_memory(2).Visible = False
function_ladder.Text_ladder_word(2).Visible = False
function_ladder.Text_ladder_bit(2).Visible = False
function_ladder.Text_ladder_memory(3).Visible = False
function_ladder.Text_ladder_word(3).Visible = False
function_ladder.Text_ladder_bit(3).Visible = False
function_ladder.Label_ladder_type.Visible = True
function_ladder.Label_ladder_word.Visible = True
function_ladder.Label_ladder_bit.Visible = True
End Sub
```

```
Private Sub show_value_ladder2()
function_ladder.Text_ladder_memory(1).Visible = True
function_ladder.Text_ladder_word(1).Visible = True
function_ladder.Text_ladder_bit(1).Visible = True
function_ladder.Text_ladder_memory(2).Visible = True
function_ladder.Text_ladder_word(2).Visible = True
function_ladder.Text_ladder_bit(2).Visible = True
function_ladder.Text_ladder_memory(3).Visible = False
function_ladder.Text_ladder_word(3).Visible = False
function_ladder.Text_ladder_bit(3).Visible = False
function_ladder.Label_ladder_type.Visible = True
function_ladder.Label_ladder_word.Visible = True
function_ladder.Label_ladder_bit.Visible = True
End Sub
```

```
Private Sub show_value_ladder3()
function_ladder.Text_ladder_memory(1).Visible = True
function_ladder.Text_ladder_word(1).Visible = True
function_ladder.Text_ladder_bit(1).Visible = True
function_ladder.Text_ladder_memory(2).Visible = True
function_ladder.Text_ladder_word(2).Visible = True
function_ladder.Text_ladder_bit(2).Visible = True
function_ladder.Text_ladder_memory(3).Visible = True
function_ladder.Text_ladder_word(3).Visible = True
function_ladder.Text_ladder_bit(3).Visible = True
function_ladder.Label_ladder_type.Visible = True
function_ladder.Label_ladder_word.Visible = True
function_ladder.Label_ladder_bit.Visible = True
End Sub
```

```
Private Sub show_nonvalue_ladder()
function_ladder.Text_ladder_memory(1).Visible = False
```

```
function_ladder.Text_ladder_word(1).Visible = False
function_ladder.Text_ladder_bit(1).Visible = False
function_ladder.Text_ladder_memory(2).Visible = False
function_ladder.Text_ladder_word(2).Visible = False
function_ladder.Text_ladder_bit(2).Visible = False
function_ladder.Text_ladder_memory(3).Visible = False
function_ladder.Text_ladder_word(3).Visible = False
function_ladder.Text_ladder_bit(3).Visible = False
function_ladder.Label_ladder_type.Visible = False
function_ladder.Label_ladder_word.Visible = False
function_ladder.Label_ladder_bit.Visible = False
End Sub
```

```
Private Sub show_value_ladder_tim_cnt()
function_ladder.Text_ladder_memory(1).Visible = False
function_ladder.Text_ladder_word(1).Visible = True
function_ladder.Text_ladder_bit(1).Visible = False
function_ladder.Text_ladder_memory(2).Visible = False
function_ladder.Text_ladder_word(2).Visible = True
function_ladder.Text_ladder_bit(2).Visible = False
function_ladder.Text_ladder_memory(3).Visible = False
function_ladder.Text_ladder_word(3).Visible = False
function_ladder.Text_ladder_bit(3).Visible = False
function_ladder.Label_ladder_type.Visible = False
function_ladder.Label_ladder_bit.Visible = False
End Sub
```

```
Private Sub option_type_memory_ladder_all_true()
function_ladder.Option_function_ladder_hr.Enabled = True
function_ladder.Option_function_ladder_ir.Enabled = True
function_ladder.Option_function_ladder_lr.Enabled = True
function_ladder.Option_function_ladder_dm.Enabled = True
```

```
function_ladder.Option_function_ladder_constant.Enabled = True
function_ladder.Option_function_ladder_ir.Value = True
End Sub
---
Private Sub option_type_memory_ladder_all_false()
function_ladder.Option_function_ladder_hr.Enabled = False
function_ladder.Option_function_ladder_ir.Enabled = False
function_ladder.Option_function_ladder_lr.Enabled = False
function_ladder.Option_function_ladder_dm.Enabled = False
function_ladder.Option_function_ladder_constant.Enabled = False
function_ladder.Option_function_ladder_hr.Value = False
function_ladder.Option_function_ladder_ir.Value = False
function_ladder.Option_function_ladder_lr.Value = False
function_ladder.Option_function_ladder_dm.Value = False
function_ladder.Option_function_ladder_constant.Value = False
function_ladder.Option_function_ladder_ir.Value = False
End Sub
---
Private Function code_program1(code_number As Integer, line_number As Integer) 'code one line bit
type
codecommand(code_number + 1) = Chr(Val("&H" + Mid(Grid_Boolean.TextMatrix(line_number, 2),
2, 2)))
codecommand(code_number + 2) = Chr(Val("&H" + Mid(Grid_Boolean.TextMatrix(line_number, 2),
4, 2)))
If (Len(Grid_Boolean.TextMatrix(line_number, 1)) > 3) Then ' When is TIM or CNT
   If((Mid(Grid_Boolean.TextMatrix(line_number, 1), Len(Grid_Boolean.TextMatrix(line_number, 1))
- 2, 3) = "TIM") Or (Mid(Grid_Boolean.TextMatrix(line_number, 1), Len(Grid_Boolean.TextMatrix
(line_number, 1)) - 2, 3) = "CNT")) Then
      codecommand(code_number + 1) = Chr(Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number, 2), 2, 1)))
      codecommand(code_number + 2) = Chr(Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number, 2), 3, 2)))
```

```
    End If
End If
End Function
```

```
Private Function code_program1_word(code_number As Integer, line_number As Integer) 'code one
line follow type
    Select Case Mid(Grid_Boolean.TextMatrix(line_number, 2), 1, 1)
        Case "I"
            codecommand(code_number + 1) = Chr(&HA0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number, 2), 3, 1)))
        Case "D"
            codecommand(code_number + 1) = Chr(&HB0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number, 2), 3, 1)))
        Case "L"
            codecommand(code_number + 1) = Chr(&HC0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number, 2), 3, 1)))
        Case "H"
            codecommand(code_number + 1) = Chr(&HD0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number, 2), 3, 1)))
        Case "#"
            codecommand(code_number + 1) = Chr(&HF0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number, 2), 3, 1)))
    End Select
    codecommand(code_number + 2) = Chr(Val("&H" + Mid(Grid_Boolean.TextMatrix(line_number,
2), 4, 2)))
    End If
End Function
```

```
Private Function code_program2_word(code_number As Integer, line_number As Integer) 'code two
line follow type
    Select Case Mid(Grid_Boolean.TextMatrix(line_number, 2), 1, 1)
        Case "I"
```

```
        codecommand(code_number + 1) = Chr(&HA0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number, 2), 3, 1)))
      Case "D"
        codecommand(code_number + 1) = Chr(&HB0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number, 2), 3, 1)))
      Case "L"
        codecommand(code_number + 1) = Chr(&HC0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number, 2), 3, 1)))
      Case "H"
        codecommand(code_number + 1) = Chr(&HD0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number, 2), 3, 1)))
      Case "#"
        codecommand(code_number + 1) = Chr(&HF0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number, 2), 3, 1)))
    End Select
    codecommand(code_number + 2) = Chr(Val("&H" + Mid(Grid_Boolean.TextMatrix(line_number,
2), 4, 2)))
    codecommand(code_number + 3) = Chr(&H74)
    Select Case Mid(Grid_Boolean.TextMatrix(line_number + 1, 2), 1, 1)
      Case "I"
        codecommand(code_number + 4) = Chr(&HA0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number + 1, 2), 3, 1)))
      Case "D"
        codecommand(code_number + 4) = Chr(&HB0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number + 1, 2), 3, 1)))
      Case "L"
        codecommand(code_number + 4) = Chr(&HC0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number + 1, 2), 3, 1)))
      Case "H"
        codecommand(code_number + 4) = Chr(&HD0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number + 1, 2), 3, 1)))
      Case "#"
```

```
        codecommand(code_number + 4) = Chr(&HF0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number + 1, 2), 3, 1)))
    End Select
    codecommand(code_number + 5) = Chr(Val("&H" + Mid(Grid_Boolean.TextMatrix(line_number +
1, 2), 4, 2)))
End Function
-----------------------------------------------------------------------------------------------------------------------
Private Function code_program3_word(code_number As Integer, line_number As Integer) 'code three
line follow type
    Select Case Mid(Grid_Boolean.TextMatrix(line_number, 2), 1, 1)
        Case "I"
            codecommand(code_number + 1) = Chr(&HA0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number, 2), 3, 1)))
        Case "D"
            codecommand(code_number + 1) = Chr(&HB0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number, 2), 3, 1)))
        Case "L"
            codecommand(code_number + 1) = Chr(&HC0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number, 2), 3, 1)))
        Case "H"
            codecommand(code_number + 1) = Chr(&HD0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number, 2), 3, 1)))
        Case "#"
            codecommand(code_number + 1) = Chr(&HF0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number, 2), 3, 1)))
    End Select
    codecommand(code_number + 2) = Chr(Val("&H" + Mid(Grid_Boolean.TextMatrix(line_number,
2), 4, 2)))
    codecommand(code_number + 3) = Chr(&H74)
    Select Case Mid(Grid_Boolean.TextMatrix(line_number + 1, 2), 1, 1)
        Case "I"
```

```
        codecommand(code_number + 4) = Chr(&HA0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number + 1, 2), 3, 1)))
      Case "D"
        codecommand(code_number + 4) = Chr(&HB0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number + 1, 2), 3, 1)))
      Case "L"
        codecommand(code_number + 4) = Chr(&HC0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number + 1, 2), 3, 1)))
      Case "H"
        codecommand(code_number + 4) = Chr(&HD0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number + 1, 2), 3, 1)))
      Case "#"
        codecommand(code_number + 4) = Chr(&HF0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number + 1, 2), 3, 1)))
    End Select
    codecommand(code_number + 5) = Chr(Val("&H" + Mid(Grid_Boolean.TextMatrix(line_number +
1, 2), 4, 2)))
    codecommand(code_number + 6) = Chr(&H75)
    Select Case Mid(Grid_Boolean.TextMatrix(line_number + 2, 2), 1, 1)
      Case "I"
        codecommand(code_number + 7) = Chr(&HA0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number + 2, 2), 3, 1)))
      Case "D"
        codecommand(code_number + 7) = Chr(&HB0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number + 2, 2), 3, 1)))
      Case "L"
        codecommand(code_number + 7) = Chr(&HC0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number + 2, 2), 3, 1)))
      Case "H"
        codecommand(code_number + 7) = Chr(&HD0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number + 2, 2), 3, 1)))
      Case "#"
```

```
        codecommand(code_number + 7) = Chr(&HF0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number + 2, 2), 3, 1)))
    End Select
    codecommand(code_number + 8) = Chr(Val("&H" + Mid(Grid_Boolean.TextMatrix(line_number +
2, 2), 4, 2)))
End Function
```

```
Private Function code_program_non_value(code_number As Integer) 'code one line non type
    codecommand(code_number + 1) = Chr(&H0)
    codecommand(code_number + 2) = Chr(&H0)
End Function

Private Function code_program_tim_cnt(code_number As Integer, line_number As Integer) 'code timer
and counter
    codecommand(code_number + 1) = Chr(Val("&H" + Mid(Grid_Boolean.TextMatrix(line_number,
2), 2, 1)))
    codecommand(code_number + 2) = Chr(Val("&H" + Mid(Grid_Boolean.TextMatrix(line_number,
2), 3, 2)))
    If Grid_Boolean.TextMatrix(line_number, 1) = "TIM" Then
        codecommand(code_number + 3) = Chr(&H76)
    Elself Grid_Boolean.TextMatrix(line_number, 1) = "CNT" Then
        codecommand(code_number + 3) = Chr(&H77)
    End If
    codecommand(code_number + 4) = Chr(&HF0 + Val("&H" + Mid(Grid_Boolean.TextMatrix
(line_number + 1, 2), 2, 1)))
    codecommand(code_number + 5) = Chr(Val("&H" + Mid(Grid_Boolean.TextMatrix(line_number +
1, 2), 3, 2)))
End Function
```

```
Private Function checking_end() As Boolean ' check command Function END(01)is TRUE when have
it.
    checking_end = False
```

```
For i = 1 To Grid_Boolean.Rows - 1
   If (Grid_Boolean.TextMatrix(i, 1) <> Empty) And (Grid_Boolean.TextMatrix(i, 1) = "END(01)")
Then
      checking_end = True
   End If
Next i
End Function
```

```
Private Function Hex_code_grid_boolean() As String 'compiler convert command to Hex code
Dim number_pointer_code As Integer
Dim i As Integer
number_pointer_code = 1
For j = 1 To 6000
   codecommand(j) = &H0
Next j
Do
   i = i + 1
   If Grid_Boolean.TextMatrix(i, 1) <> Empty Then
      If (Grid_Boolean.TextMatrix(i, 1) = "NOP(00)") Or ((Grid_Boolean.TextMatrix(i, 1) = Empty)
And (Grid_Boolean.TextMatrix(i, 2) = Empty)) Then 'code 00
         codecommand(number_pointer_code) = Chr(&H0)
         code_program_non_value (number_pointer_code)
         number_pointer_code = number_pointer_code + 3
      ElseIf Grid_Boolean.TextMatrix(i, 1) = "END(01)" Then 'code 01
         codecommand(number_pointer_code) = Chr(&H1)
         code_program_non_value (number_pointer_code)
         number_pointer_code = number_pointer_code + 3
      ElseIf Grid_Boolean.TextMatrix(i, 1) = "STC(40)" Then 'code 02
         codecommand(number_pointer_code) = Chr(&H2)
         code_program_non_value (number_pointer_code)
         number_pointer_code = number_pointer_code + 3
      ElseIf Grid_Boolean.TextMatrix(i, 1) = "CTC(41)" Then 'code 03
```

```
        codecommand(number_pointer_code) = Chr(&H3)
        code_program_non_value (number_pointer_code)
        number_pointer_code = number_pointer_code + 3
    ElseIf Grid_Boolean.TextMatrix(i, 1) = "ANDLD" Then 'code 04
        codecommand(number_pointer_code) = Chr(&H4)
        code_program_non_value (number_pointer_code)
        number_pointer_code = number_pointer_code + 3
    ElseIf Grid_Boolean.TextMatrix(i, 1) = "ORLD" Then 'code 05
        codecommand(number_pointer_code) = Chr(&H5)
        code_program_non_value (number_pointer_code)
        number_pointer_code = number_pointer_code + 3
    ElseIf Grid_Boolean.TextMatrix(i, 1) = "IL(02)" Then 'code 06
        codecommand(number_pointer_code) = Chr(&H6)
        code_program_non_value (number_pointer_code)
        number_pointer_code = number_pointer_code + 3
    ElseIf Grid_Boolean.TextMatrix(i, 1) = "ILC(03)" Then 'code 07
        codecommand(number_pointer_code) = Chr(&H7)
        code_program_non_value (number_pointer_code)
        number_pointer_code = number_pointer_code + 3
    ElseIf Grid_Boolean.TextMatrix(i, 1) = "JMP(04)" Then 'code 08
        codecommand(number_pointer_code) = Chr(&H8)
        code_program_non_value (number_pointer_code)
        number_pointer_code = number_pointer_code + 3
    ElseIf Grid_Boolean.TextMatrix(i, 1) = "JME(05)" Then 'code 09
        codecommand(number_pointer_code) = Chr(&H9)
        code_program_non_value (number_pointer_code)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "LD") And (Mid(Grid_Boolean.TextMatrix(i, 2), 1, 1) =
"I") Then 'code 40
        codecommand(number_pointer_code) = Chr(&H40)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
```

```
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "AND") And (Mid(Grid_Boolean.TextMatrix(i, 2), 1, 1)
= "I") Then 'code 41
        codecommand(number_pointer_code) = Chr(&H41)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "OR") And (Mid(Grid_Boolean.TextMatrix(i, 2), 1, 1) =
"I") Then 'code 42
        codecommand(number_pointer_code) = Chr(&H42)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "OUT") And (Mid(Grid_Boolean.TextMatrix(i, 2), 1, 1) =
"I") Then 'code 43
        codecommand(number_pointer_code) = Chr(&H43)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "LDNOT") And (Mid(Grid_Boolean.TextMatrix(i, 2), 1,
1) = "I") Then 'code 44
        codecommand(number_pointer_code) = Chr(&H44)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "ANDNOT") And (Mid(Grid_Boolean.TextMatrix(i, 2),
1, 1) = "I") Then 'code 45
        codecommand(number_pointer_code) = Chr(&H45)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "ORNOT") And (Mid(Grid_Boolean.TextMatrix(i, 2), 1,
1) = "I") Then 'code 46
        codecommand(number_pointer_code) = Chr(&H46)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "OUTNOT") And (Mid(Grid_Boolean.TextMatrix(i, 2),
1, 1) = "I") Then 'code 47
```

```
        codecommand(number_pointer_code) = Chr(&H47)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "KEEP(11)") And (Mid(Grid_Boolean.TextMatrix(i, 2),
1, 1) = "I") Then 'code 48
        codecommand(number_pointer_code) = Chr(&H48)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "DIFU(13)") And (Mid(Grid_Boolean.TextMatrix(i, 2),
1, 1) = "I") Then 'code 49
        codecommand(number_pointer_code) = Chr(&H49)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "DIFD(14)") And (Mid(Grid_Boolean.TextMatrix(i, 2),
1, 1) = "I") Then 'code 4A
        codecommand(number_pointer_code) = Chr(&H4A)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "LD") And (Mid(Grid_Boolean.TextMatrix(i, 2), 1, 1) =
"H") Then 'code 4B
        codecommand(number_pointer_code) = Chr(&H4B)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "AND") And (Mid(Grid_Boolean.TextMatrix(i, 2), 1, 1)
= "H") Then 'code 4C
        codecommand(number_pointer_code) = Chr(&H4C)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "OR") And (Mid(Grid_Boolean.TextMatrix(i, 2), 1, 1) =
"H") Then 'code 4D
        codecommand(number_pointer_code) = Chr(&H4D)
        Call code_program1(number_pointer_code, i)
```

```
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "OUT") And (Mid(Grid_Boolean.TextMatrix(i, 2), 1, 1) =
"H") Then 'code 4E
        codecommand(number_pointer_code) = Chr(&H4E)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "LDNOT") And (Mid(Grid_Boolean.TextMatrix(i, 2), 1,
1) = "H") Then 'code 4F
        codecommand(number_pointer_code) = Chr(&H4F)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "ANDNOT") And (Mid(Grid_Boolean.TextMatrix(i, 2),
1, 1) = "H") Then 'code 50
        codecommand(number_pointer_code) = Chr(&H50)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "ORNOT") And (Mid(Grid_Boolean.TextMatrix(i, 2), 1,
1) = "H") Then 'code 51
        codecommand(number_pointer_code) = Chr(&H51)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "OUTNOT") And (Mid(Grid_Boolean.TextMatrix(i, 2),
1, 1) = "H") Then 'code 52
        codecommand(number_pointer_code) = Chr(&H52)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "KEEP(11)") And (Mid(Grid_Boolean.TextMatrix(i, 2),
1, 1) = "H") Then 'code 53
        codecommand(number_pointer_code) = Chr(&H53)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
```

```
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "DIFU(13)") And (Mid(Grid_Boolean.TextMatrix(i, 2),
1, 1) = "H") Then 'code 54
        codecommand(number_pointer_code) = Chr(&H54)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "DIFD(14)") And (Mid(Grid_Boolean.TextMatrix(i, 2),
1, 1) = "H") Then 'code 55
        codecommand(number_pointer_code) = Chr(&H55)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "LD") And (Mid(Grid_Boolean.TextMatrix(i, 2), 1, 1) =
"L") Then 'code 56
        codecommand(number_pointer_code) = Chr(&H56)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "AND") And (Mid(Grid_Boolean.TextMatrix(i, 2), 1, 1)
= "L") Then 'code 57
        codecommand(number_pointer_code) = Chr(&H57)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "OR") And (Mid(Grid_Boolean.TextMatrix(i, 2), 1, 1) =
"L") Then 'code 58
        codecommand(number_pointer_code) = Chr(&H58)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "OUT") And (Mid(Grid_Boolean.TextMatrix(i, 2), 1, 1) =
"L") Then 'code 59
        codecommand(number_pointer_code) = Chr(&H59)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "LDNOT") And (Mid(Grid_Boolean.TextMatrix(i, 2), 1,
1) = "L") Then 'code 5A
```

```
        codecommand(number_pointer_code) = Chr(&H5A)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "ANDNOT") And (Mid(Grid_Boolean.TextMatrix(i, 2),
1, 1) = "L") Then 'code 5B
        codecommand(number_pointer_code) = Chr(&H5B)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "ORNOT") And (Mid(Grid_Boolean.TextMatrix(i, 2), 1,
1) = "L") Then 'code 5C
        codecommand(number_pointer_code) = Chr(&H5C)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "OUTNOT") And (Mid(Grid_Boolean.TextMatrix(i, 2),
1, 1) = "L") Then 'code 5D
        codecommand(number_pointer_code) = Chr(&H5D)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "KEEP(11)") And (Mid(Grid_Boolean.TextMatrix(i, 2),
1, 1) = "L") Then 'code 5E
        codecommand(number_pointer_code) = Chr(&H5E)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "DIFU(13)") And (Mid(Grid_Boolean.TextMatrix(i, 2),
1, 1) = "L") Then 'code 5F
        codecommand(number_pointer_code) = Chr(&H5F)
        Call code_program1(number_pointer_code, i)
        number_pointer_code = number_pointer_code + 3
    ElseIf (Grid_Boolean.TextMatrix(i, 1) = "DIFD(14)") And (Mid(Grid_Boolean.TextMatrix(i, 2),
1, 1) = "L") Then 'code 60
        codecommand(number_pointer_code) = Chr(&H60)
        Call code_program1(number_pointer_code, i)
```

```
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "LDTIM" Then 'code 61
    codecommand(number_pointer_code) = Chr(&H61)
    Call code_program1(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "LDCNT" Then 'code 62
    codecommand(number_pointer_code) = Chr(&H62)
    Call code_program1(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "LDNOTTIM" Then 'code 63
    codecommand(number_pointer_code) = Chr(&H63)
    Call code_program1(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "LDNOTCNT" Then 'code 64
    codecommand(number_pointer_code) = Chr(&H64)
    Call code_program1(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "ANDTIM" Then 'code 65
    codecommand(number_pointer_code) = Chr(&H65)
    Call code_program1(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "ANDCNT" Then 'code 66
    codecommand(number_pointer_code) = Chr(&H66)
    Call code_program1(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "ANDNOTTIM" Then 'code 67
    codecommand(number_pointer_code) = Chr(&H67)
    Call code_program1(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "ANDNOTCNT" Then 'code 68
    codecommand(number_pointer_code) = Chr(&H68)
    Call code_program1(number_pointer_code, i)
```

```
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "ORTIM" Then 'code 69
    codecommand(number_pointer_code) = Chr(&H69)
    Call code_program1(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "ORCNT" Then 'code 6A
    codecommand(number_pointer_code) = Chr(&H6A)
    Call code_program1(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "ORNOTTIM" Then 'code 6B
    codecommand(number_pointer_code) = Chr(&H6B)
    Call code_program1(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "ORNOTCNT" Then 'code 6C
    codecommand(number_pointer_code) = Chr(&H6C)
    Call code_program1(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "COM(29)" Then 'code 6D
    codecommand(number_pointer_code) = Chr(&H6D)
    Call code_program1_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "INC(38)" Then 'code 6E
    codecommand(number_pointer_code) = Chr(&H6E)
    Call code_program1_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "DEC(39)" Then 'code 6F
    codecommand(number_pointer_code) = Chr(&H6F)
    Call code_program1_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "ASL(25)" Then 'code 70
    codecommand(number_pointer_code) = Chr(&H70)
    Call code_program1_word(number_pointer_code, i)
```

```
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "ASR(26)" Then 'code 71
    codecommand(number_pointer_code) = Chr(&H71)
    Call code_program1_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "ROL(27)" Then 'code 72
    codecommand(number_pointer_code) = Chr(&H72)
    Call code_program1_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "ROR(28)" Then 'code 73
    codecommand(number_pointer_code) = Chr(&H73)
    Call code_program1_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "MSG(47)" Then 'code 79
    codecommand(number_pointer_code) = Chr(&H79)
    Call code_program1_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 3
ElseIf Grid_Boolean.TextMatrix(i, 1) = "SFT(10)" Then 'code 80
    codecommand(number_pointer_code) = Chr(&H80)
    Call code_program2_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 6
ElseIf Grid_Boolean.TextMatrix(i, 1) = "WSFT(16)" Then 'code 81
    codecommand(number_pointer_code) = Chr(&H81)
    Call code_program2_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 6
ElseIf Grid_Boolean.TextMatrix(i, 1) = "MOV(21)" Then 'code 82
    codecommand(number_pointer_code) = Chr(&H82)
    Call code_program2_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 6
ElseIf Grid_Boolean.TextMatrix(i, 1) = "MVN(22)" Then 'code 83
    codecommand(number_pointer_code) = Chr(&H83)
    Call code_program2_word(number_pointer_code, i)
```

```
    number_pointer_code = number_pointer_code + 6
ElseIf Grid_Boolean.TextMatrix(i, 1) = "CMP(20)" Then 'code 84
    codecommand(number_pointer_code) = Chr(&H84)
    Call code_program2_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 6
ElseIf Grid_Boolean.TextMatrix(i, 1) = "BIN(23)" Then 'code 85
    codecommand(number_pointer_code) = Chr(&H85)
    Call code_program2_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 6
ElseIf Grid_Boolean.TextMatrix(i, 1) = "BCD(24)" Then 'code 86
    codecommand(number_pointer_code) = Chr(&H86)
    Call code_program2_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 6
ElseIf Grid_Boolean.TextMatrix(i, 1) = "RDI(70)" Then 'code 87
    codecommand(number_pointer_code) = Chr(&H87)
    Call code_program2_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 6
ElseIf Grid_Boolean.TextMatrix(i, 1) = "WRO(71)" Then 'code 88
    codecommand(number_pointer_code) = Chr(&H88)
    Call code_program2_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 6
ElseIf Grid_Boolean.TextMatrix(i, 1) = "SLD(74)" Then 'code 89
    codecommand(number_pointer_code) = Chr(&H89)
    Call code_program2_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 6
ElseIf Grid_Boolean.TextMatrix(i, 1) = "SRD(75)" Then 'code 8A
    codecommand(number_pointer_code) = Chr(&H8A)
    Call code_program2_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 6
ElseIf Grid_Boolean.TextMatrix(i, 1) = "SFTR(84)" Then 'code 8B
    codecommand(number_pointer_code) = Chr(&H8B)
    Call code_program2_word(number_pointer_code, i)
```

```
    number_pointer_code = number_pointer_code + 6
ElseIf Grid_Boolean.TextMatrix(i, 1) = "TKY(79)" Then 'code 8C
    codecommand(number_pointer_code) = Chr(&H8C)
    Call code_program2_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 6
ElseIf Grid_Boolean.TextMatrix(i, 1) = "7SEG(80)" Then 'code 8D
    codecommand(number_pointer_code) = Chr(&H8D)
    Call code_program2_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 6
ElseIf Grid_Boolean.TextMatrix(i, 1) = "TIM" Then 'code 8E
    codecommand(number_pointer_code) = Chr(&H8E)
    Call code_program_tim_cnt(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 6
ElseIf Grid_Boolean.TextMatrix(i, 1) = "CNT" Then 'code 8F
    codecommand(number_pointer_code) = Chr(&H8F)
    Call code_program_tim_cnt(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 6
ElseIf Grid_Boolean.TextMatrix(i, 1) = "ADD(30)" Then 'code C0
    codecommand(number_pointer_code) = Chr(&HC0)
    Call code_program3_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 9
ElseIf Grid_Boolean.TextMatrix(i, 1) = "SUB(31)" Then 'code C1
    codecommand(number_pointer_code) = Chr(&HC1)
    Call code_program3_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 9
ElseIf Grid_Boolean.TextMatrix(i, 1) = "MUL(32)" Then 'code C2
    codecommand(number_pointer_code) = Chr(&HC2)
    Call code_program3_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 9
ElseIf Grid_Boolean.TextMatrix(i, 1) = "DIV(33)" Then 'code C3
    codecommand(number_pointer_code) = Chr(&HC3)
    Call code_program3_word(number_pointer_code, i)
```

```
    number_pointer_code = number_pointer_code + 9
ElseIf Grid_Boolean.TextMatrix(i, 1) = "ANDW(34)" Then 'code 34
    codecommand(number_pointer_code) = Chr(&HC4)
    Call code_program3_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 9
ElseIf Grid_Boolean.TextMatrix(i, 1) = "ORW(35)" Then 'code 35
    codecommand(number_pointer_code) = Chr(&HC5)
    Call code_program3_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 9
ElseIf Grid_Boolean.TextMatrix(i, 1) = "XORW(36)" Then 'code 36
    codecommand(number_pointer_code) = Chr(&HC6)
    Call code_program3_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 9
ElseIf Grid_Boolean.TextMatrix(i, 1) = "XNRW(37)" Then 'code 37
    codecommand(number_pointer_code) = Chr(&HC7)
    Call code_program3_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 9
ElseIf Grid_Boolean.TextMatrix(i, 1) = "MLPX(76)" Then 'code 38
    codecommand(number_pointer_code) = Chr(&HC8)
    Call code_program3_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 9
ElseIf Grid_Boolean.TextMatrix(i, 1) = "ADB(50)" Then  'code C9
    codecommand(number_pointer_code) = Chr(&HC9)
    Call code_program3_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 9
ElseIf Grid_Boolean.TextMatrix(i, 1) = "SBB(51)" Then 'code  &HCA
    codecommand(number_pointer_code) = Chr(&HCA)
    Call code_program3_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 9
ElseIf Grid_Boolean.TextMatrix(i, 1) = "MLB(52)" Then 'code  &HCB
    codecommand(number_pointer_code) = Chr(&HCB)
    Call code_program3_word(number_pointer_code, i)
```

```
    number_pointer_code = number_pointer_code + 9
  ElseIf Grid_Boolean.TextMatrix(i, 1) = "DVB(53)" Then 'code  &HCC
    codecommand(number_pointer_code) = Chr(&HCC)
    Call code_program3_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 9
  ElseIf Grid_Boolean.TextMatrix(i, 1) = "MOVB(82)" Then 'code  &HCD
    codecommand(number_pointer_code) = Chr(&HCD)
    Call code_program3_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 9
  ElseIf Grid_Boolean.TextMatrix(i, 1) = "MOVD(83)" Then 'code  &HCE
    codecommand(number_pointer_code) = Chr(&HCE)
    Call code_program3_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 9
  ElseIf Grid_Boolean.TextMatrix(i, 1) = "ASC(86)" Then 'code  &HCF
    codecommand(number_pointer_code) = Chr(&HCF)
    Call code_program3_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 9
  ElseIf Grid_Boolean.TextMatrix(i, 1) = "SDEC(78)" Then 'code  &HCD0
    codecommand(number_pointer_code) = Chr(&HD0)
    Call code_program3_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 9
  ElseIf Grid_Boolean.TextMatrix(i, 1) = "TXD(87)" Then 'code  &HD1
    codecommand(number_pointer_code) = Chr(&HD1)
    Call code_program3_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 9
  ElseIf Grid_Boolean.TextMatrix(i, 1) = "RXD(88)" Then 'code  &HD2
    codecommand(number_pointer_code) = Chr(&HD2)
    Call code_program3_word(number_pointer_code, i)
    number_pointer_code = number_pointer_code + 9
  End If
End If
Loop Until (i > Grid_Boolean.Rows - 2) Or (Grid_Boolean.TextMatrix(i, 1) = "END(01)")
```

```
End Function
---------------------------------------------------------------------------------------------------------------
Private Function value_check_bit(bit As Integer, value_bit As String) As Integer ' bit check of
component check
Select Case value_bit
  Case "0"
    If bit = 0 Then
      value_check_bit = Unchecked
    ElseIf bit = 1 Then
      value_check_bit = Unchecked
    ElseIf bit = 2 Then
      value_check_bit = Unchecked
    ElseIf bit = 3 Then
      value_check_bit = Unchecked
    End If
  Case "1"
    If bit = 0 Then
      value_check_bit = Checked
    ElseIf bit = 1 Then
      value_check_bit = Unchecked
    ElseIf bit = 2 Then
      value_check_bit = Unchecked
    ElseIf bit = 3 Then
      value_check_bit = Unchecked
    End If
  Case "2"
    If bit = 0 Then
      value_check_bit = Unchecked
    ElseIf bit = 1 Then
      value_check_bit = Checked
    ElseIf bit = 2 Then
      value_check_bit = Unchecked
```

```
    ElseIf bit = 3 Then
      value_check_bit = Unchecked
    End If
  Case "3"
    If bit = 0 Then
      value_check_bit = Checked
    ElseIf bit = 1 Then
      value_check_bit = Checked
    ElseIf bit = 2 Then
      value_check_bit = Unchecked
    ElseIf bit = 3 Then
      value_check_bit = Unchecked
    End If
  Case "4"
    If bit = 0 Then
      value_check_bit = Unchecked
    ElseIf bit = 1 Then
      value_check_bit = Unchecked
    ElseIf bit = 2 Then
      value_check_bit = Checked
    ElseIf bit = 3 Then
      value_check_bit = Unchecked
    End If
  Case "5"
    If bit = 0 Then
      value_check_bit = Checked
    ElseIf bit = 1 Then
      value_check_bit = Unchecked
    ElseIf bit = 2 Then
      value_check_bit = Checked
    ElseIf bit = 3 Then
      value_check_bit = Unchecked
```

```
    End If
Case "6"
    If bit = 0 Then
        value_check_bit = Unchecked
    ElseIf bit = 1 Then
        value_check_bit = Checked
    ElseIf bit = 2 Then
        value_check_bit = Checked
    ElseIf bit = 3 Then
        value_check_bit = Unchecked
    End If
Case "7"
    If bit = 0 Then
        value_check_bit = Checked
    ElseIf bit = 1 Then
        value_check_bit = Checked
    ElseIf bit = 2 Then
        value_check_bit = Checked
    ElseIf bit = 3 Then
        value_check_bit = Unchecked
    End If
Case "8"
    If bit = 0 Then
        value_check_bit = Unchecked
    ElseIf bit = 1 Then
        value_check_bit = Unchecked
    ElseIf bit = 2 Then
        value_check_bit = Unchecked
    ElseIf bit = 3 Then
        value_check_bit = Checked
    End If
Case "9"
```

```
    If bit = 0 Then
      value_check_bit = Checked
    ElseIf bit = 1 Then
      value_check_bit = Unchecked
    ElseIf bit = 2 Then
      value_check_bit = Unchecked
    ElseIf bit = 3 Then
      value_check_bit = Checked
    End If
  Case "A"
    If bit = 0 Then
      value_check_bit = Unchecked
    ElseIf bit = 1 Then
      value_check_bit = Checked
    ElseIf bit = 2 Then
      value_check_bit = Unchecked
    ElseIf bit = 3 Then
      value_check_bit = Checked
    End If
  Case "B"
    If bit = 0 Then
      value_check_bit = Checked
    ElseIf bit = 1 Then
      value_check_bit = Checked
    ElseIf bit = 2 Then
      value_check_bit = Unchecked
    ElseIf bit = 3 Then
      value_check_bit = Checked
    End If
  Case "C"
    If bit = 0 Then
      value_check_bit = Unchecked
```

```
    ElseIf bit = 1 Then
      value_check_bit = Unchecked
    ElseIf bit = 2 Then
      value_check_bit = Checked
    ElseIf bit = 3 Then
      value_check_bit = Checked
    End If
  Case "D"
    If bit = 0 Then
      value_check_bit = Checked
    ElseIf bit = 1 Then
      value_check_bit = Unchecked
    ElseIf bit = 2 Then
      value_check_bit = Checked
    ElseIf bit = 3 Then
      value_check_bit = Checked
    End If
  Case "E"
    If bit = 0 Then
      value_check_bit = Unchecked
    ElseIf bit = 1 Then
      value_check_bit = Checked
    ElseIf bit = 2 Then
      value_check_bit = Checked
    ElseIf bit = 3 Then
      value_check_bit = Checked
    End If
  Case "F"
    If bit = 0 Then
      value_check_bit = Checked
    ElseIf bit = 1 Then
      value_check_bit = Checked
```

```
    ElseIf bit = 2 Then
      value_check_bit = Checked
    ElseIf bit = 3 Then
      value_check_bit = Checked
    End If
End Select
End Function
```

```
Private Function value_ir_check_digit(digit As Integer, value_digit As String) ' check value digit of ir
memory
If digit = 0 Then
  Check_ir_bit(0).Value = value_check_bit(0, value_digit)
  Check_ir_bit(1).Value = value_check_bit(1, value_digit)
  Check_ir_bit(2).Value = value_check_bit(2, value_digit)
  Check_ir_bit(3).Value = value_check_bit(3, value_digit)
ElseIf digit = 1 Then
  Check_ir_bit(4).Value = value_check_bit(0, value_digit)
  Check_ir_bit(5).Value = value_check_bit(1, value_digit)
  Check_ir_bit(6).Value = value_check_bit(2, value_digit)
  Check_ir_bit(7).Value = value_check_bit(3, value_digit)
ElseIf digit = 2 Then
  Check_ir_bit(8).Value = value_check_bit(0, value_digit)
  Check_ir_bit(9).Value = value_check_bit(1, value_digit)
  Check_ir_bit(10).Value = value_check_bit(2, value_digit)
  Check_ir_bit(11).Value = value_check_bit(3, value_digit)
ElseIf digit = 3 Then
  Check_ir_bit(12).Value = value_check_bit(0, value_digit)
  Check_ir_bit(13).Value = value_check_bit(1, value_digit)
  Check_ir_bit(14).Value = value_check_bit(2, value_digit)
  Check_ir_bit(15).Value = value_check_bit(3, value_digit)
End If
End Function
```

```
Private Function value_hr_check_digit(digit As Integer, value_digit As String) ' check value digit of hr
memory
If digit = 0 Then
   Check_hr_bit(0).Value = value_check_bit(0, value_digit)
   Check_hr_bit(1).Value = value_check_bit(1, value_digit)
   Check_hr_bit(2).Value = value_check_bit(2, value_digit)
   Check_hr_bit(3).Value = value_check_bit(3, value_digit)
ElseIf digit = 1 Then
   Check_hr_bit(4).Value = value_check_bit(0, value_digit)
   Check_hr_bit(5).Value = value_check_bit(1, value_digit)
   Check_hr_bit(6).Value = value_check_bit(2, value_digit)
   Check_hr_bit(7).Value = value_check_bit(3, value_digit)
ElseIf digit = 2 Then
   Check_hr_bit(8).Value = value_check_bit(0, value_digit)
   Check_hr_bit(9).Value = value_check_bit(1, value_digit)
   Check_hr_bit(10).Value = value_check_bit(2, value_digit)
   Check_hr_bit(11).Value = value_check_bit(3, value_digit)
ElseIf digit = 3 Then
   Check_hr_bit(12).Value = value_check_bit(0, value_digit)
   Check_hr_bit(13).Value = value_check_bit(1, value_digit)
   Check_hr_bit(14).Value = value_check_bit(2, value_digit)
   Check_hr_bit(15).Value = value_check_bit(3, value_digit)
End If
End Function
```

```
Private Function value_lr_check_digit(digit As Integer, value_digit As String)  ' check value digit of lr
memory
If digit = 0 Then
   Check_lr_bit(0).Value = value_check_bit(0, value_digit)
   Check_lr_bit(1).Value = value_check_bit(1, value_digit)
   Check_lr_bit(2).Value = value_check_bit(2, value_digit)
   Check_lr_bit(3).Value = value_check_bit(3, value_digit)
```

```
ElseIf digit = 1 Then
    Check_lr_bit(4).Value = value_check_bit(0, value_digit)
    Check_lr_bit(5).Value = value_check_bit(1, value_digit)
    Check_lr_bit(6).Value = value_check_bit(2, value_digit)
    Check_lr_bit(7).Value = value_check_bit(3, value_digit)
ElseIf digit = 2 Then
    Check_lr_bit(8).Value = value_check_bit(0, value_digit)
    Check_lr_bit(9).Value = value_check_bit(1, value_digit)
    Check_lr_bit(10).Value = value_check_bit(2, value_digit)
    Check_lr_bit(11).Value = value_check_bit(3, value_digit)
ElseIf digit = 3 Then
    Check_lr_bit(12).Value = value_check_bit(0, value_digit)
    Check_lr_bit(13).Value = value_check_bit(1, value_digit)
    Check_lr_bit(14).Value = value_check_bit(2, value_digit)
    Check_lr_bit(15).Value = value_check_bit(3, value_digit)
End If
End Function
-----------------------------------------------------------------------------------------------------------
Private Function value_dm_check_digit(digit As Integer, value_digit As String) ' check value digit of
dm memory
If digit = 0 Then
    Check_dm_bit(0).Value = value_check_bit(0, value_digit)
    Check_dm_bit(1).Value = value_check_bit(1, value_digit)
    Check_dm_bit(2).Value = value_check_bit(2, value_digit)
    Check_dm_bit(3).Value = value_check_bit(3, value_digit)
ElseIf digit = 1 Then
    Check_dm_bit(4).Value = value_check_bit(0, value_digit)
    Check_dm_bit(5).Value = value_check_bit(1, value_digit)
    Check_dm_bit(6).Value = value_check_bit(2, value_digit)
    Check_dm_bit(7).Value = value_check_bit(3, value_digit)
ElseIf digit = 2 Then
    Check_dm_bit(8).Value = value_check_bit(0, value_digit)
```

```
    Check_dm_bit(9).Value = value_check_bit(1, value_digit)
    Check_dm_bit(10).Value = value_check_bit(2, value_digit)
    Check_dm_bit(11).Value = value_check_bit(3, value_digit)
ElseIf digit = 3 Then
    Check_dm_bit(12).Value = value_check_bit(0, value_digit)
    Check_dm_bit(13).Value = value_check_bit(1, value_digit)
    Check_dm_bit(14).Value = value_check_bit(2, value_digit)
    Check_dm_bit(15).Value = value_check_bit(3, value_digit)
End If
End Function
-----------------------------------------------------------------------------------------------------------
Private Function value_tc_check_digit(digit As Integer, value_digit As String)  ' check value digit of tc
memory
If digit = 0 Then
    Check_tc_bit(0).Value = value_check_bit(0, value_digit)
    Check_tc_bit(1).Value = value_check_bit(1, value_digit)
    Check_tc_bit(2).Value = value_check_bit(2, value_digit)
    Check_tc_bit(3).Value = value_check_bit(3, value_digit)
ElseIf digit = 1 Then
    Check_tc_bit(4).Value = value_check_bit(0, value_digit)
    Check_tc_bit(5).Value = value_check_bit(1, value_digit)
    Check_tc_bit(6).Value = value_check_bit(2, value_digit)
    Check_tc_bit(7).Value = value_check_bit(3, value_digit)
ElseIf digit = 2 Then
    Check_tc_bit(8).Value = value_check_bit(0, value_digit)
    Check_tc_bit(9).Value = value_check_bit(1, value_digit)
    Check_tc_bit(10).Value = value_check_bit(2, value_digit)
    Check_tc_bit(11).Value = value_check_bit(3, value_digit)
ElseIf digit = 3 Then
    Check_tc_bit(12).Value = value_check_bit(0, value_digit)
    Check_tc_bit(13).Value = value_check_bit(1, value_digit)
    Check_tc_bit(14).Value = value_check_bit(2, value_digit)
```

```
    Check_tc_bit(15).Value = value_check_bit(3, value_digit)
End If
End Function
---------------------------------------------------------------------------------------------------------------
Private Function nonvalue_ir_check_digit(digit As Integer) 'clear value of ir memory
If digit = 0 Then
    Check_ir_bit(0).Value = Unchecked
    Check_ir_bit(1).Value = Unchecked
    Check_ir_bit(2).Value = Unchecked
    Check_ir_bit(3).Value = Unchecked
ElseIf digit = 1 Then
    Check_ir_bit(4).Value = Unchecked
    Check_ir_bit(5).Value = Unchecked
    Check_ir_bit(6).Value = Unchecked
    Check_ir_bit(7).Value = Unchecked
ElseIf digit = 2 Then
    Check_ir_bit(8).Value = Unchecked
    Check_ir_bit(9).Value = Unchecked
    Check_ir_bit(10).Value = Unchecked
    Check_ir_bit(11).Value = Unchecked
ElseIf digit = 3 Then
    Check_ir_bit(12).Value = Unchecked
    Check_ir_bit(13).Value = Unchecked
    Check_ir_bit(14).Value = Unchecked
    Check_ir_bit(15).Value = Unchecked
End If
End Function
---------------------------------------------------------------------------------------------------------------
Private Function nonvalue_hr_check_digit(digit As Integer) 'clear value of hr memory
If digit = 0 Then
    Check_hr_bit(0).Value = Unchecked
    Check_hr_bit(1).Value = Unchecked
```

```
    Check_hr_bit(2).Value = Unchecked
    Check_hr_bit(3).Value = Unchecked
ElseIf digit = 1 Then
    Check_hr_bit(4).Value = Unchecked
    Check_hr_bit(5).Value = Unchecked
    Check_hr_bit(6).Value = Unchecked
    Check_hr_bit(7).Value = Unchecked
ElseIf digit = 2 Then
    Check_hr_bit(8).Value = Unchecked
    Check_hr_bit(9).Value = Unchecked
    Check_hr_bit(10).Value = Unchecked
    Check_hr_bit(11).Value = Unchecked
ElseIf digit = 3 Then
    Check_hr_bit(12).Value = Unchecked
    Check_hr_bit(13).Value = Unchecked
    Check_hr_bit(14).Value = Unchecked
    Check_hr_bit(15).Value = Unchecked
End If
End Function
-----------------------------------------------------------------------------------------------------------
Private Function nonvalue_lr_check_digit(digit As Integer) 'clear value of lr memory
If digit = 0 Then
    Check_lr_bit(0).Value = Unchecked
    Check_lr_bit(1).Value = Unchecked
    Check_lr_bit(2).Value = Unchecked
    Check_lr_bit(3).Value = Unchecked
ElseIf digit = 1 Then
    Check_lr_bit(4).Value = Unchecked
    Check_lr_bit(5).Value = Unchecked
    Check_lr_bit(6).Value = Unchecked
    Check_lr_bit(7).Value = Unchecked
ElseIf digit = 2 Then
```

```
    Check_lr_bit(8).Value = Unchecked
    Check_lr_bit(9).Value = Unchecked
    Check_lr_bit(10).Value = Unchecked
    Check_lr_bit(11).Value = Unchecked
ElseIf digit = 3 Then
    Check_lr_bit(12).Value = Unchecked
    Check_lr_bit(13).Value = Unchecked
    Check_lr_bit(14).Value = Unchecked
    Check_lr_bit(15).Value = Unchecked
End If
End Function
-----------------------------------------------------------------------------------------------------------
Private Function nonvalue_dm_check_digit(digit As Integer) 'clear value of dm memory
If digit = 0 Then
    Check_dm_bit(0).Value = Unchecked
    Check_dm_bit(1).Value = Unchecked
    Check_dm_bit(2).Value = Unchecked
    Check_dm_bit(3).Value = Unchecked
ElseIf digit = 1 Then
    Check_dm_bit(4).Value = Unchecked
    Check_dm_bit(5).Value = Unchecked
    Check_dm_bit(6).Value = Unchecked
    Check_dm_bit(7).Value = Unchecked
ElseIf digit = 2 Then
    Check_dm_bit(8).Value = Unchecked
    Check_dm_bit(9).Value = Unchecked
    Check_dm_bit(10).Value = Unchecked
    Check_dm_bit(11).Value = Unchecked
ElseIf digit = 3 Then
    Check_dm_bit(12).Value = Unchecked
    Check_dm_bit(13).Value = Unchecked
    Check_dm_bit(14).Value = Unchecked
```

```
    Check_dm_bit(15).Value = Unchecked
End If
End Function
-----------------------------------------------------------------------------------------------------------------
Private Function nonvalue_tc_check_digit(digit As Integer) 'clear value of tc memory
If digit = 0 Then
    Check_tc_bit(0).Value = Unchecked
    Check_tc_bit(1).Value = Unchecked
    Check_tc_bit(2).Value = Unchecked
    Check_tc_bit(3).Value = Unchecked
ElseIf digit = 1 Then
    Check_tc_bit(4).Value = Unchecked
    Check_tc_bit(5).Value = Unchecked
    Check_tc_bit(6).Value = Unchecked
    Check_tc_bit(7).Value = Unchecked
ElseIf digit = 2 Then
    Check_tc_bit(8).Value = Unchecked
    Check_tc_bit(9).Value = Unchecked
    Check_tc_bit(10).Value = Unchecked
    Check_tc_bit(11).Value = Unchecked
ElseIf digit = 3 Then
    Check_tc_bit(12).Value = Unchecked
    Check_tc_bit(13).Value = Unchecked
    Check_tc_bit(14).Value = Unchecked
    Check_tc_bit(15).Value = Unchecked
End If
End Function
-----------------------------------------------------------------------------------------------------------------
Private Function check_bit_digit(digit_check As Integer, bit_check As Byte) As Byte ' check bit in digit
of memory
If digit_check = 0 Then
    If bit_check = 0 Then
```

```
        check_bit_digit = 0
    ElseIf bit_check = 1 Then
        check_bit_digit = 1
    ElseIf bit_check = 2 Then
        check_bit_digit = 2
    ElseIf bit_check = 3 Then
        check_bit_digit = 3
    End If
ElseIf digit_check = 1 Then
    If bit_check = 0 Then
        check_bit_digit = 4
    ElseIf bit_check = 1 Then
        check_bit_digit = 5
    ElseIf bit_check = 2 Then
        check_bit_digit = 6
    ElseIf bit_check = 3 Then
        check_bit_digit = 7
    End If
ElseIf digit_check = 2 Then
    If bit_check = 0 Then
        check_bit_digit = 8
    ElseIf bit_check = 1 Then
        check_bit_digit = 9
    ElseIf bit_check = 2 Then
        check_bit_digit = 10
    ElseIf bit_check = 3 Then
        check_bit_digit = 11
    End If
ElseIf digit_check = 3 Then
    If bit_check = 0 Then
        check_bit_digit = 12
    ElseIf bit_check = 1 Then
```

```
      check_bit_digit = 13
    ElseIf bit_check = 2 Then
      check_bit_digit = 14
   ElseIf bit_check = 3 Then
      check_bit_digit = 15
   End If
End If
End Function
```

```
Private Function check_bit_ir_convert_number(digit As Integer) As String ' convert bit ir  to number of
memory
bit_0 = check_bit_digit(digit, 0)
bit_1 = check_bit_digit(digit, 1)
bit_2 = check_bit_digit(digit, 2)
bit_3 = check_bit_digit(digit, 3)
If(Check_ir_bit(bit_0).Value = Unchecked) And (Check_ir_bit(bit_1).Value = Unchecked) And
(Check_ir_bit(bit_2).Value = Unchecked) And (Check_ir_bit(bit_3).Value = Unchecked) Then
   check_bit_ir_convert_number = "0"
ElseIf(Check_ir_bit(bit_0).Value = Checked) And (Check_ir_bit(bit_1).Value = Unchecked) And
(Check_ir_bit(bit_2).Value = Unchecked) And (Check_ir_bit(bit_3).Value = Unchecked) Then
   check_bit_ir_convert_number = "1"
ElseIf(Check_ir_bit(bit_0).Value = Unchecked) And (Check_ir_bit(bit_1).Value = Checked) And
(Check_ir_bit(bit_2).Value = Unchecked) And (Check_ir_bit(bit_3).Value = Unchecked) Then
   check_bit_ir_convert_number = "2"
ElseIf(Check_ir_bit(bit_0).Value = Checked) And (Check_ir_bit(bit_1).Value = Checked) And
(Check_ir_bit(bit_2).Value = Unchecked) And (Check_ir_bit(bit_3).Value = Unchecked) Then
   check_bit_ir_convert_number = "3"
ElseIf(Check_ir_bit(bit_0).Value = Unchecked) And (Check_ir_bit(bit_1).Value = Unchecked) And
(Check_ir_bit(bit_2).Value = Checked) And (Check_ir_bit(bit_3).Value = Unchecked) Then
   check_bit_ir_convert_number = "4"
ElseIf(Check_ir_bit(bit_0).Value = Checked) And (Check_ir_bit(bit_1).Value = Unchecked) And
(Check_ir_bit(bit_2).Value = Checked) And (Check_ir_bit(bit_3).Value = Unchecked) Then
```

```
    check_bit_ir_convert_number = "5"
ElseIf (Check_ir_bit(bit_0).Value = Unchecked) And (Check_ir_bit(bit_1).Value = Checked) And
(Check_ir_bit(bit_2).Value = Checked) And (Check_ir_bit(bit_3).Value = Unchecked) Then
    check_bit_ir_convert_number = "6"
ElseIf (Check_ir_bit(bit_0).Value = Checked) And (Check_ir_bit(bit_1).Value = Checked) And
(Check_ir_bit(bit_2).Value = Checked) And (Check_ir_bit(bit_3).Value = Unchecked) Then
    check_bit_ir_convert_number = "7"
ElseIf (Check_ir_bit(bit_0).Value = Unchecked) And (Check_ir_bit(bit_1).Value = Unchecked) And
(Check_ir_bit(bit_2).Value = Unchecked) And (Check_ir_bit(bit_3).Value = Checked) Then
    check_bit_ir_convert_number = "8"
ElseIf (Check_ir_bit(bit_0).Value = Checked) And (Check_ir_bit(bit_1).Value = Unchecked) And
(Check_ir_bit(bit_2).Value = Unchecked) And (Check_ir_bit(bit_3).Value = Checked) Then
    check_bit_ir_convert_number = "9"
ElseIf (Check_ir_bit(bit_0).Value = Unchecked) And (Check_ir_bit(bit_1).Value = Checked) And
(Check_ir_bit(bit_2).Value = Unchecked) And (Check_ir_bit(bit_3).Value = Checked) Then
    check_bit_ir_convert_number = "A"
ElseIf (Check_ir_bit(bit_0).Value = Checked) And (Check_ir_bit(bit_1).Value = Checked) And
(Check_ir_bit(bit_2).Value = Unchecked) And (Check_ir_bit(bit_3).Value = Checked) Then
    check_bit_ir_convert_number = "B"
ElseIf (Check_ir_bit(bit_0).Value = Unchecked) And (Check_ir_bit(bit_1).Value = Unchecked) And
(Check_ir_bit(bit_2).Value = Checked) And (Check_ir_bit(bit_3).Value = Checked) Then
    check_bit_ir_convert_number = "C"
ElseIf (Check_ir_bit(bit_0).Value = Checked) And (Check_ir_bit(bit_1).Value = Unchecked) And
(Check_ir_bit(bit_2).Value = Checked) And (Check_ir_bit(bit_3).Value = Checked) Then
    check_bit_ir_convert_number = "D"
ElseIf (Check_ir_bit(bit_0).Value = Unchecked) And (Check_ir_bit(bit_1).Value = Checked) And
(Check_ir_bit(bit_2).Value = Checked) And (Check_ir_bit(bit_3).Value = Checked) Then
    check_bit_ir_convert_number = "E"
ElseIf (Check_ir_bit(bit_0).Value = Checked) And (Check_ir_bit(bit_1).Value = Checked) And
(Check_ir_bit(bit_2).Value = Checked) And (Check_ir_bit(bit_3).Value = Checked) Then
    check_bit_ir_convert_number = "F"
End If
```

```
End Function
---------------------------------------------------------------------------------------------------------------------
Private Function check_bit_hr_convert_number(digit As Integer) As String ' convert bit hr  to number
of memory
bit_0 = check_bit_digit(digit, 0)
bit_1 = check_bit_digit(digit, 1)
bit_2 = check_bit_digit(digit, 2)
bit_3 = check_bit_digit(digit, 3)
If (Check_hr_bit(bit_0).Value = Unchecked) And (Check_hr_bit(bit_1).Value = Unchecked) And
(Check_hr_bit(bit_2).Value = Unchecked) And (Check_hr_bit(bit_3).Value = Unchecked) Then
   check_bit_hr_convert_number = "0"
ElseIf (Check_hr_bit(bit_0).Value = Checked) And (Check_hr_bit(bit_1).Value = Unchecked) And
(Check_hr_bit(bit_2).Value = Unchecked) And (Check_hr_bit(bit_3).Value = Unchecked) Then
   check_bit_hr_convert_number = "1"
ElseIf (Check_hr_bit(bit_0).Value = Unchecked) And (Check_hr_bit(bit_1).Value = Checked) And
(Check_hr_bit(bit_2).Value = Unchecked) And (Check_hr_bit(bit_3).Value = Unchecked) Then
   check_bit_hr_convert_number = "2"
ElseIf (Check_hr_bit(bit_0).Value = Checked) And (Check_hr_bit(bit_1).Value = Checked) And
(Check_hr_bit(bit_2).Value = Unchecked) And (Check_hr_bit(bit_3).Value = Unchecked) Then
   check_bit_hr_convert_number = "3"
ElseIf (Check_hr_bit(bit_0).Value = Unchecked) And (Check_hr_bit(bit_1).Value = Unchecked) And
(Check_hr_bit(bit_2).Value = Checked) And (Check_hr_bit(bit_3).Value = Unchecked) Then
   check_bit_hr_convert_number = "4"
ElseIf (Check_hr_bit(bit_0).Value = Checked) And (Check_hr_bit(bit_1).Value = Unchecked) And
(Check_hr_bit(bit_2).Value = Checked) And (Check_hr_bit(bit_3).Value = Unchecked) Then
   check_bit_hr_convert_number = "5"
ElseIf (Check_hr_bit(bit_0).Value = Unchecked) And (Check_hr_bit(bit_1).Value = Checked) And
(Check_hr_bit(bit_2).Value = Checked) And (Check_hr_bit(bit_3).Value = Unchecked) Then
   check_bit_hr_convert_number = "6"
ElseIf (Check_hr_bit(bit_0).Value = Checked) And (Check_hr_bit(bit_1).Value = Checked) And
(Check_hr_bit(bit_2).Value = Checked) And (Check_hr_bit(bit_3).Value = Unchecked) Then
   check_bit_hr_convert_number = "7"
```

```
ElseIf (Check_hr_bit(bit_0).Value = Unchecked) And (Check_hr_bit(bit_1).Value = Unchecked) And
(Check_hr_bit(bit_2).Value = Unchecked) And (Check_hr_bit(bit_3).Value = Checked) Then
   check_bit_hr_convert_number = "8"
ElseIf (Check_hr_bit(bit_0).Value = Checked) And (Check_hr_bit(bit_1).Value = Unchecked) And
(Check_hr_bit(bit_2).Value = Unchecked) And (Check_hr_bit(bit_3).Value = Checked) Then
   check_bit_hr_convert_number = "9"
ElseIf (Check_hr_bit(bit_0).Value = Unchecked) And (Check_hr_bit(bit_1).Value = Checked) And
(Check_hr_bit(bit_2).Value = Unchecked) And (Check_hr_bit(bit_3).Value = Checked) Then
   check_bit_hr_convert_number = "A"
ElseIf (Check_hr_bit(bit_0).Value = Checked) And (Check_hr_bit(bit_1).Value = Checked) And
(Check_hr_bit(bit_2).Value = Unchecked) And (Check_hr_bit(bit_3).Value = Checked) Then
   check_bit_hr_convert_number = "B"
ElseIf (Check_hr_bit(bit_0).Value = Unchecked) And (Check_hr_bit(bit_1).Value = Unchecked) And
(Check_hr_bit(bit_2).Value = Checked) And (Check_hr_bit(bit_3).Value = Checked) Then
   check_bit_hr_convert_number = "C"
ElseIf (Check_hr_bit(bit_0).Value = Checked) And (Check_hr_bit(bit_1).Value = Unchecked) And
(Check_hr_bit(bit_2).Value = Checked) And (Check_hr_bit(bit_3).Value = Checked) Then
   check_bit_hr_convert_number = "D"
ElseIf (Check_hr_bit(bit_0).Value = Unchecked) And (Check_hr_bit(bit_1).Value = Checked) And
(Check_hr_bit(bit_2).Value = Checked) And (Check_hr_bit(bit_3).Value = Checked) Then
   check_bit_hr_convert_number = "E"
ElseIf (Check_hr_bit(bit_0).Value = Checked) And (Check_hr_bit(bit_1).Value = Checked) And
(Check_hr_bit(bit_2).Value = Checked) And (Check_hr_bit(bit_3).Value = Checked) Then
   check_bit_hr_convert_number = "F"
End If
End Function

Private Function check_bit_lr_convert_number(digit As Integer) As String ' convert bit lr  to number of
memory
bit_0 = check_bit_digit(digit, 0)
bit_1 = check_bit_digit(digit, 1)
bit_2 = check_bit_digit(digit, 2)
```

```
bit_3 = check_bit_digit(digit, 3)
If (Check_lr_bit(bit_0).Value = Unchecked) And (Check_lr_bit(bit_1).Value = Unchecked) And
(Check_lr_bit(bit_2).Value = Unchecked) And (Check_lr_bit(bit_3).Value = Unchecked) Then
   check_bit_lr_convert_number = "0"
ElseIf (Check_lr_bit(bit_0).Value = Checked) And (Check_lr_bit(bit_1).Value = Unchecked) And
(Check_lr_bit(bit_2).Value = Unchecked) And (Check_lr_bit(bit_3).Value = Unchecked) Then
   check_bit_lr_convert_number = "1"
ElseIf (Check_lr_bit(bit_0).Value = Unchecked) And (Check_lr_bit(bit_1).Value = Checked) And
(Check_lr_bit(bit_2).Value = Unchecked) And (Check_lr_bit(bit_3).Value = Unchecked) Then
   check_bit_lr_convert_number = "2"
ElseIf (Check_lr_bit(bit_0).Value = Checked) And (Check_lr_bit(bit_1).Value = Checked) And
(Check_lr_bit(bit_2).Value = Unchecked) And (Check_lr_bit(bit_3).Value = Unchecked) Then
   check_bit_lr_convert_number = "3"
ElseIf (Check_lr_bit(bit_0).Value = Unchecked) And (Check_lr_bit(bit_1).Value = Unchecked) And
(Check_lr_bit(bit_2).Value = Checked) And (Check_lr_bit(bit_3).Value = Unchecked) Then
   check_bit_lr_convert_number = "4"
ElseIf (Check_lr_bit(bit_0).Value = Checked) And (Check_lr_bit(bit_1).Value = Unchecked) And
(Check_lr_bit(bit_2).Value = Checked) And (Check_lr_bit(bit_3).Value = Unchecked) Then
   check_bit_lr_convert_number = "5"
ElseIf (Check_lr_bit(bit_0).Value = Unchecked) And (Check_lr_bit(bit_1).Value = Checked) And
(Check_lr_bit(bit_2).Value = Checked) And (Check_lr_bit(bit_3).Value = Unchecked) Then
   check_bit_lr_convert_number = "6"
ElseIf (Check_lr_bit(bit_0).Value = Checked) And (Check_lr_bit(bit_1).Value = Checked) And
(Check_lr_bit(bit_2).Value = Checked) And (Check_lr_bit(bit_3).Value = Unchecked) Then
   check_bit_lr_convert_number = "7"
ElseIf (Check_lr_bit(bit_0).Value = Unchecked) And (Check_lr_bit(bit_1).Value = Unchecked) And
(Check_lr_bit(bit_2).Value = Unchecked) And (Check_lr_bit(bit_3).Value = Checked) Then
   check_bit_lr_convert_number = "8"
ElseIf (Check_lr_bit(bit_0).Value = Checked) And (Check_lr_bit(bit_1).Value = Unchecked) And
(Check_lr_bit(bit_2).Value = Unchecked) And (Check_lr_bit(bit_3).Value = Checked) Then
   check_bit_lr_convert_number = "9"
```

```
ElseIf (Check_lr_bit(bit_0).Value = Unchecked) And (Check_lr_bit(bit_1).Value = Checked) And
(Check_lr_bit(bit_2).Value = Unchecked) And (Check_lr_bit(bit_3).Value = Checked) Then
   check_bit_lr_convert_number = "A"
ElseIf (Check_lr_bit(bit_0).Value = Checked) And (Check_lr_bit(bit_1).Value = Checked) And
(Check_lr_bit(bit_2).Value = Unchecked) And (Check_lr_bit(bit_3).Value = Checked) Then
   check_bit_lr_convert_number = "B"
ElseIf (Check_lr_bit(bit_0).Value = Unchecked) And (Check_lr_bit(bit_1).Value = Unchecked) And
(Check_lr_bit(bit_2).Value = Checked) And (Check_lr_bit(bit_3).Value = Checked) Then
   check_bit_lr_convert_number = "C"
ElseIf (Check_lr_bit(bit_0).Value = Checked) And (Check_lr_bit(bit_1).Value = Unchecked) And
(Check_lr_bit(bit_2).Value = Checked) And (Check_lr_bit(bit_3).Value = Checked) Then
   check_bit_lr_convert_number = "D"
ElseIf (Check_lr_bit(bit_0).Value = Unchecked) And (Check_lr_bit(bit_1).Value = Checked) And
(Check_lr_bit(bit_2).Value = Checked) And (Check_lr_bit(bit_3).Value = Checked) Then
   check_bit_lr_convert_number = "E"
ElseIf (Check_lr_bit(bit_0).Value = Checked) And (Check_lr_bit(bit_1).Value = Checked) And
(Check_lr_bit(bit_2).Value = Checked) And (Check_lr_bit(bit_3).Value = Checked) Then
   check_bit_lr_convert_number = "F"
End If
End Function
-------------------------------------------------------------------------------------------------------------
Private Function check_bit_dm_convert_number(digit As Integer) As String ' convert bit dm  to number
of memory
bit_0 = check_bit_digit(digit, 0)
bit_1 = check_bit_digit(digit, 1)
bit_2 = check_bit_digit(digit, 2)
bit_3 = check_bit_digit(digit, 3)
If (Check_dm_bit(bit_0).Value = Unchecked) And (Check_dm_bit(bit_1).Value = Unchecked) And
(Check_dm_bit(bit_2).Value = Unchecked) And (Check_dm_bit(bit_3).Value = Unchecked) Then
   check_bit_dm_convert_number = "0"
ElseIf (Check_dm_bit(bit_0).Value = Checked) And (Check_dm_bit(bit_1).Value = Unchecked) And
(Check_dm_bit(bit_2).Value = Unchecked) And (Check_dm_bit(bit_3).Value = Unchecked) Then
```

```
    check_bit_dm_convert_number = "1"
ElseIf (Check_dm_bit(bit_0).Value = Unchecked) And (Check_dm_bit(bit_1).Value = Checked) And
(Check_dm_bit(bit_2).Value = Unchecked) And (Check_dm_bit(bit_3).Value = Unchecked) Then
    check_bit_dm_convert_number = "2"
ElseIf (Check_dm_bit(bit_0).Value = Checked) And (Check_dm_bit(bit_1).Value = Checked) And
(Check_dm_bit(bit_2).Value = Unchecked) And (Check_dm_bit(bit_3).Value = Unchecked) Then
    check_bit_dm_convert_number = "3"
ElseIf (Check_dm_bit(bit_0).Value = Unchecked) And (Check_dm_bit(bit_1).Value = Unchecked) And
(Check_dm_bit(bit_2).Value = Checked) And (Check_dm_bit(bit_3).Value = Unchecked) Then
    check_bit_dm_convert_number = "4"
ElseIf (Check_dm_bit(bit_0).Value = Checked) And (Check_dm_bit(bit_1).Value = Unchecked) And
(Check_dm_bit(bit_2).Value = Checked) And (Check_dm_bit(bit_3).Value = Unchecked) Then
    check_bit_dm_convert_number = "5"
ElseIf (Check_dm_bit(bit_0).Value = Unchecked) And (Check_dm_bit(bit_1).Value = Checked) And
(Check_dm_bit(bit_2).Value = Checked) And (Check_dm_bit(bit_3).Value = Unchecked) Then
    check_bit_dm_convert_number = "6"
ElseIf (Check_dm_bit(bit_0).Value = Checked) And (Check_dm_bit(bit_1).Value = Checked) And
(Check_dm_bit(bit_2).Value = Checked) And (Check_dm_bit(bit_3).Value = Unchecked) Then
    check_bit_dm_convert_number = "7"
ElseIf (Check_dm_bit(bit_0).Value = Unchecked) And (Check_dm_bit(bit_1).Value = Unchecked) And
(Check_dm_bit(bit_2).Value = Unchecked) And (Check_dm_bit(bit_3).Value = Checked) Then
    check_bit_dm_convert_number = "8"
ElseIf (Check_dm_bit(bit_0).Value = Checked) And (Check_dm_bit(bit_1).Value = Unchecked) And
(Check_dm_bit(bit_2).Value = Unchecked) And (Check_dm_bit(bit_3).Value = Checked) Then
    check_bit_dm_convert_number = "9"
ElseIf (Check_dm_bit(bit_0).Value = Unchecked) And (Check_dm_bit(bit_1).Value = Checked) And
(Check_dm_bit(bit_2).Value = Unchecked) And (Check_dm_bit(bit_3).Value = Checked) Then
    check_bit_dm_convert_number = "A"
ElseIf (Check_dm_bit(bit_0).Value = Checked) And (Check_dm_bit(bit_1).Value = Checked) And
(Check_dm_bit(bit_2).Value = Unchecked) And (Check_dm_bit(bit_3).Value = Checked) Then
    check_bit_dm_convert_number = "B"
```

```
ElseIf (Check_dm_bit(bit_0).Value = Unchecked) And (Check_dm_bit(bit_1).Value = Unchecked) And
(Check_dm_bit(bit_2).Value = Checked) And (Check_dm_bit(bit_3).Value = Checked) Then
   check_bit_dm_convert_number = "C"
ElseIf (Check_dm_bit(bit_0).Value = Checked) And (Check_dm_bit(bit_1).Value = Unchecked) And
(Check_dm_bit(bit_2).Value = Checked) And (Check_dm_bit(bit_3).Value = Checked) Then
   check_bit_dm_convert_number = "D"
ElseIf (Check_dm_bit(bit_0).Value = Unchecked) And (Check_dm_bit(bit_1).Value = Checked) And
(Check_dm_bit(bit_2).Value = Checked) And (Check_dm_bit(bit_3).Value = Checked) Then
   check_bit_dm_convert_number = "E"
ElseIf (Check_dm_bit(bit_0).Value = Checked) And (Check_dm_bit(bit_1).Value = Checked) And
(Check_dm_bit(bit_2).Value = Checked) And (Check_dm_bit(bit_3).Value = Checked) Then
   check_bit_dm_convert_number = "F"
End If
End Function
-----------------------------------------------------------------------------------------------------------
Private Function check_bit_tc_convert_number(digit As Integer) As String ' convert bit tc  to number of
memory
bit_0 = check_bit_digit(digit, 0)
bit_1 = check_bit_digit(digit, 1)
bit_2 = check_bit_digit(digit, 2)
bit_3 = check_bit_digit(digit, 3)
If (Check_tc_bit(bit_0).Value = Unchecked) And (Check_tc_bit(bit_1).Value = Unchecked) And
(Check_tc_bit(bit_2).Value = Unchecked) And (Check_tc_bit(bit_3).Value = Unchecked) Then
   check_bit_tc_convert_number = "0"
ElseIf (Check_tc_bit(bit_0).Value = Checked) And (Check_tc_bit(bit_1).Value = Unchecked) And
(Check_tc_bit(bit_2).Value = Unchecked) And (Check_tc_bit(bit_3).Value = Unchecked) Then
   check_bit_tc_convert_number = "1"
ElseIf (Check_tc_bit(bit_0).Value = Unchecked) And (Check_tc_bit(bit_1).Value = Checked) And
(Check_tc_bit(bit_2).Value = Unchecked) And (Check_tc_bit(bit_3).Value = Unchecked) Then
   check_bit_tc_convert_number = "2"
ElseIf (Check_tc_bit(bit_0).Value = Checked) And (Check_tc_bit(bit_1).Value = Checked) And
   (Check_tc_bit(bit_2).Value = Unchecked) And (Check_tc_bit(bit_3).Value = Unchecked) Then
```

```
    check_bit_tc_convert_number = "3"
ElseIf (Check_tc_bit(bit_0).Value = Unchecked) And (Check_tc_bit(bit_1).Value = Unchecked) And
(Check_tc_bit(bit_2).Value = Checked) And (Check_tc_bit(bit_3).Value = Unchecked) Then
    check_bit_tc_convert_number = "4"
ElseIf (Check_tc_bit(bit_0).Value = Checked) And (Check_tc_bit(bit_1).Value = Unchecked) And
(Check_tc_bit(bit_2).Value = Checked) And (Check_tc_bit(bit_3).Value = Unchecked) Then
    check_bit_tc_convert_number = "5"
ElseIf (Check_tc_bit(bit_0).Value = Unchecked) And (Check_tc_bit(bit_1).Value = Checked) And
(Check_tc_bit(bit_2).Value = Checked) And (Check_tc_bit(bit_3).Value = Unchecked) Then
    check_bit_tc_convert_number = "6"
ElseIf (Check_tc_bit(bit_0).Value = Checked) And (Check_tc_bit(bit_1).Value = Checked) And
(Check_tc_bit(bit_2).Value = Checked) And (Check_tc_bit(bit_3).Value = Unchecked) Then
    check_bit_tc_convert_number = "7"
ElseIf (Check_tc_bit(bit_0).Value = Unchecked) And (Check_tc_bit(bit_1).Value = Unchecked) And
(Check_tc_bit(bit_2).Value = Unchecked) And (Check_tc_bit(bit_3).Value = Checked) Then
    check_bit_tc_convert_number = "8"
ElseIf (Check_tc_bit(bit_0).Value = Checked) And (Check_tc_bit(bit_1).Value = Unchecked) And
(Check_tc_bit(bit_2).Value = Unchecked) And (Check_tc_bit(bit_3).Value = Checked) Then
    check_bit_tc_convert_number = "9"
ElseIf (Check_tc_bit(bit_0).Value = Unchecked) And (Check_tc_bit(bit_1).Value = Checked) And
(Check_tc_bit(bit_2).Value = Unchecked) And (Check_tc_bit(bit_3).Value = Checked) Then
    check_bit_tc_convert_number = "A"
ElseIf (Check_tc_bit(bit_0).Value = Checked) And (Check_tc_bit(bit_1).Value = Checked) And
(Check_tc_bit(bit_2).Value = Unchecked) And (Check_tc_bit(bit_3).Value = Checked) Then
    check_bit_tc_convert_number = "B"
ElseIf (Check_tc_bit(bit_0).Value = Unchecked) And (Check_tc_bit(bit_1).Value = Unchecked) And
(Check_tc_bit(bit_2).Value = Checked) And (Check_tc_bit(bit_3).Value = Checked) Then
    check_bit_tc_convert_number = "C"
ElseIf (Check_tc_bit(bit_0).Value = Checked) And (Check_tc_bit(bit_1).Value = Unchecked) And
(Check_tc_bit(bit_2).Value = Checked) And (Check_tc_bit(bit_3).Value = Checked) Then
    check_bit_tc_convert_number = "D"
```

```
ElseIf (Check_tc_bit(bit_0).Value = Unchecked) And (Check_tc_bit(bit_1).Value = Checked) And
(Check_tc_bit(bit_2).Value = Checked) And (Check_tc_bit(bit_3).Value = Checked) Then
   check_bit_tc_convert_number = "E"
ElseIf (Check_tc_bit(bit_0).Value = Checked) And (Check_tc_bit(bit_1).Value = Checked) And
(Check_tc_bit(bit_2).Value = Checked) And (Check_tc_bit(bit_3).Value = Checked) Then
   check_bit_tc_convert_number = "F"
End If
End Function
-------------------------------------------------------------------------------------------------------------

Private Function Lenght_data(data As String) As Integer 'calculate lenght of data
   Lenght_data = Len(data)
End Function
-------------------------------------------------------------------------------------------------------------

Private Function Shift_data(dataold As String, datanew As String) As String 'add data of word
   Shift_data = dataold + datanew
End Function
-------------------------------------------------------------------------------------------------------------

Private Function Add_boolean_grid(checking As Integer) 'add function and valve to
label_fun,label_word
   If Label_fun.Caption = "TIM" Or Label_fun.Caption = "CNT" Or Label_fun.Caption = "LDTIM" Or
Label_fun.Caption = "ANDTIM" Or Label_fun.Caption = "ORTIM" Or Label_fun.Caption =
"LDCNT" Or Label_fun.Caption = "ANDCNT" Or Label_fun.Caption = "ORCNT" Or
Label_fun.Caption = "LDNOTTIM" Or Label_fun.Caption = "ANDNOTTIM" Or Label_fun.Caption =
"ORNOTTIM" Or Label_fun.Caption = "LDNOTCNT" Or Label_fun.Caption = "ANDNOTCNT" Or
Label_fun.Caption = "ORNOTCNT" Then
      If Len(Label_data.Caption) = 1 Then
         Label_word(checking).Caption = "00" + Label_data.Caption
      ElseIf Len(Label_data.Caption) = 2 Then
         Label_word(checking).Caption = "0" + Label_data.Caption
      ElseIf Len(Label_data.Caption) = 3 Then
         Label_word(checking).Caption = Label_data.Caption
      Else
```

```
      Label_word(checking).Caption = Mid(Label_data.Caption, 2, 3)
    End If
  Else
    If Len(Label_data.Caption) = 1 Then
      Label_word(checking).Caption = "000" + Label_data.Caption
    ElseIf Len(Label_data.Caption) = 2 Then
      Label_word(checking).Caption = "00" + Label_data.Caption
    ElseIf Len(Label_data.Caption) = 3 Then
      Label_word(checking).Caption = "0" + Label_data.Caption
    Else
      Label_word(checking).Caption = Label_data.Caption
    End If
  End If
  Fun_send_grid_boolean (checking)
  clear_command_line
  check_command_line
End Function
-----------------------------------------------------------------------------------------------------------------------
Private Function Select_option(sequence As Integer) 'select option
If Option_ir.Value = True Then
  Label_memory(sequence) = "I"
ElseIf Option_hr.Value = True Then
  Label_memory(sequence) = "H"
ElseIf Option_lr.Value = True Then
  Label_memory(sequence) = "L"
ElseIf Option_dm.Value = True Then
  Label_memory(sequence) = "D"
ElseIf Option_constant.Value = True Then
  Label_memory(sequence) = "#"
End If
End Function
```

```
Private Function Fun_send_grid_boolean(sequence As Integer) 'send label_fun,label_word to
grid_boolean
Dim number_gridboolean_row As Integer 'check row command line
Dim number_gridboolean_row_upper As Integer 'check Upper row command line
Dim number_gridboolean_row_lower As Integer 'check Lower row command line
Dim i As Integer
  If sequence = 1 Then 'One command line
    Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) = Label_fun.Caption
    If Label_fun.Caption = "LDTIM" Or Label_fun.Caption = "ANDTIM" Or Label_fun.Caption =
"ORTIM" Or Label_fun.Caption = "LDCNT" Or Label_fun.Caption = "ANDCNT" Or
Label_fun.Caption = "ORCNT" Or Label_fun.Caption = "LDNOTTIM" Or Label_fun.Caption =
"ANDNOTTIM" Or Label_fun.Caption = "ORNOTTIM" Or Label_fun.Caption = "LDNOTCNT" Or
Label_fun.Caption = "ANDNOTCNT" Or Label_fun.Caption = "ORNOTCNT" Then
      If Label_word(sequence).Caption > 99 Then
        Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = Str(Label_word(sequence).Caption)
      ElseIf Label_word(sequence).Caption > 9 Then
        Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = " 0" + Trim(Str(Label_word
(sequence).Caption))
      ElseIf Label_word(sequence).Caption > 0 Then
        Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = " 00" + Trim(Str(Label_word
(sequence).Caption))
      ElseIf Label_word(sequence).Caption = 0 Then
        Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = " 000"
      End If
    Else
      Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = Label_memory(sequence).Caption +
Label_word(sequence).Caption
    End If
    If (Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 1) = Empty) And
(Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 2) = Empty) Then
      Grid_Boolean.row = Grid_Boolean.row + 1
    ElseIf Grid_Boolean.row <> Grid_Boolean.Rows - 1 Then
```

```
        If ((Grid_Boolean.TextMatrix(Grid_Boolean.row - 1, 1) <> Empty) Or
(Grid_Boolean.TextMatrix(Grid_Boolean.row - 1, 2) <> Empty)) And ((Grid_Boolean.TextMatrix
(Grid_Boolean.row + 1, 1) = Empty) And (Grid_Boolean.TextMatrix(Grid_Boolean.row + 1, 2) =
Empty)) And ((Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 1) <> Empty) Or
(Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 2) <> Empty)) Then
            Grid_Boolean.row = Grid_Boolean.row + 1
          End If
        End If
        Grid_Boolean.col = 1
        Grid_Boolean.CellAlignment = 1
        Grid_Boolean.SetFocus
      ElseIf sequence = 2 Then 'Two command line
        If (Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) = Empty) And (Grid_Boolean.TextMatrix
(Grid_Boolean.row, 2) = Empty) And (Grid_Boolean.row <> Grid_Boolean.Rows - 1) Then
          Do Until (Grid_Boolean.TextMatrix(Grid_Boolean.row + i, 1) <> Empty) Or
(Grid_Boolean.row + i = Grid_Boolean.Rows - 1)
            i = i + 1
            number_gridboolean_row = Grid_Boolean.row + i
          Loop
          For i = 1 To Grid_Boolean.Rows - 1 - number_gridboolean_row
            Grid_Boolean.TextMatrix(Grid_Boolean.Rows - i, 1) = Grid_Boolean.TextMatrix
(Grid_Boolean.Rows - i - 1, 1)
            Grid_Boolean.TextMatrix(Grid_Boolean.Rows - i, 2) = Grid_Boolean.TextMatrix
(Grid_Boolean.Rows - i - 1, 2)
          Next i
          Grid_Boolean.TextMatrix(number_gridboolean_row, 1) = Empty
          Grid_Boolean.TextMatrix(number_gridboolean_row, 2) = Empty
        End If
        Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) = Label_fun.Caption
        For i = 1 To sequence 'send command follow of sequence
          If Label_fun.Caption = "TIM" Or Label_fun.Caption = "CNT" Then
            If i = 1 Then
```

```
            If Label_word(i).Caption > 99 Then
              Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = Str(Label_word(i).Caption)
            ElseIf Label_word(i).Caption > 9 Then
              Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = " 0" + Trim(Str(Label_word
(i).Caption))
            ElseIf Label_word(i).Caption > 0 Then
              Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = " 00" + Trim(Str(Label_word
(i).Caption))
            ElseIf Label_word(i).Caption = 0 Then
              Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = " 000"
            End If
          Else
            Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = Label_memory(i).Caption +
Label_word(i).Caption
          End If
        Else
          Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = Label_memory(i).Caption + Label_word
(i).Caption
        End If
        If (Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 1) = Empty) And
(Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 2) = Empty) Then
          Grid_Boolean.row = Grid_Boolean.row + 1
        ElseIf Grid_Boolean.row <> Grid_Boolean.Rows - 1 Then
          If ((Grid_Boolean.TextMatrix(Grid_Boolean.row - 1, 1) <> Empty) Or
(Grid_Boolean.TextMatrix(Grid_Boolean.row - 1, 2) <> Empty)) And ((Grid_Boolean.TextMatrix
(Grid_Boolean.row + 1, 1) = Empty) And (Grid_Boolean.TextMatrix(Grid_Boolean.row + 1, 2) =
Empty)) And ((Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 1) <> Empty) Or
(Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 2) <> Empty)) Then
            Grid_Boolean.row = Grid_Boolean.row + 1
          End If
        End If
        Grid_Boolean.col = 1
```

```
        Grid_Boolean.CellAlignment = 1
    Next i
    Grid_Boolean.SetFocus
  ElseIf sequence = 3 Then 'Three command line
    If (Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) = Empty) And (Grid_Boolean.TextMatrix
(Grid_Boolean.row, 2) = Empty) And (Grid_Boolean.row <> Grid_Boolean.Rows - 1) Then
      Do Until (Grid_Boolean.TextMatrix(Grid_Boolean.row + i, 1) <> Empty) Or
(Grid_Boolean.row + i = Grid_Boolean.Rows - 1) 'Check Upper row of Grid_boolean
        i = i + 1
        number_gridboolean_row_upper = Grid_Boolean.row + i
      Loop
      i = 0
      If (Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 1) = Empty) And
(Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 2) = Empty) Then
        Do Until ((Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1 - i, 1) <> Empty) Or
(Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1 - i, 2) <> Empty)) 'Check Lower row of
Grid_boolean
          number_gridboolean_row_lower = Grid_Boolean.Rows - 2 - i
          i = i + 1
        Loop
      End If
      For i = 1 To number_gridboolean_row_lower - number_gridboolean_row_upper + 1
        Grid_Boolean.TextMatrix(number_gridboolean_row_lower + 3 - i, 1) =
Grid_Boolean.TextMatrix(number_gridboolean_row_lower - i + 1, 1)
        Grid_Boolean.TextMatrix(number_gridboolean_row_lower + 3 - i, 2) =
Grid_Boolean.TextMatrix(number_gridboolean_row_lower - i + 1, 2)
      Next i
      Grid_Boolean.TextMatrix(number_gridboolean_row_upper, 1) = Empty
      Grid_Boolean.TextMatrix(number_gridboolean_row_upper, 2) = Empty
      If (number_gridboolean_row_upper <> Grid_Boolean.Rows - 1) Then
        Grid_Boolean.TextMatrix(number_gridboolean_row_upper + 1, 1) = Empty
        Grid_Boolean.TextMatrix(number_gridboolean_row_upper + 1, 2) = Empty
```

```
        End If
      End If
      Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) = Label_fun.Caption
      For i = 1 To sequence 'send command follow of sequence
        Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = Label_memory(i).Caption + Label_word
(i).Caption
        If ((Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 1) = Empty) And
(Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 2) = Empty)) And ((Grid_Boolean.Rows - 1 -
Grid_Boolean.row) > 2) Then
          Grid_Boolean.row = Grid_Boolean.row + 1
        ElseIf ((Grid_Boolean.Rows - 1 - Grid_Boolean.row) <= 2) And (Grid_Boolean.row <>
Grid_Boolean.Rows - 1) Then
          Grid_Boolean.row = Grid_Boolean.row + 1
        ElseIf Grid_Boolean.row <> Grid_Boolean.Rows - 1 Then
          If ((Grid_Boolean.TextMatrix(Grid_Boolean.row - 1, 1) <> Empty) Or
(Grid_Boolean.TextMatrix(Grid_Boolean.row - 1, 2) <> Empty)) Or ((Grid_Boolean.TextMatrix
(Grid_Boolean.row + 1, 1) = Empty) And (Grid_Boolean.TextMatrix(Grid_Boolean.row + 1, 2) =
Empty)) And ((Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 1) <> Empty) Or
(Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 2) <> Empty)) Then
            Grid_Boolean.row = Grid_Boolean.row + 1
          End If
        End If
        Grid_Boolean.col = 1
        Grid_Boolean.CellAlignment = 1
      Next i
    End If
    Grid_Boolean.SetFocus
    checkboolean = checkboolean_temp
    checkbooleansequence = checkbooleansequence_temp
End Function
```

```
Private Sub botton_delete_Click() 'delete command
Dim number_gridboolean_row_upper As Integer
Dim number_gridboolean_row_lower As Integer
Dim i As Integer
  If (Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) = Empty) And (Grid_Boolean.TextMatrix
(Grid_Boolean.row, 2) = Empty) Then 'check command line is empty
    If Grid_Boolean.row <> Grid_Boolean.Rows - 1 Then ' not last command line
      If (Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 1) <> Empty) Or
(Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 2) <> Empty) Then 'command line not empty
        For i = Grid_Boolean.row To Grid_Boolean.Rows - 2
          Grid_Boolean.TextMatrix(i, 1) = Grid_Boolean.TextMatrix(i + 1, 1)
          Grid_Boolean.TextMatrix(i, 2) = Grid_Boolean.TextMatrix(i + 1, 2)
        Next i
        Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 1) = Empty
        Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 2) = Empty
      Else 'last command line is empty
        Do Until (Grid_Boolean.TextMatrix(Grid_Boolean.row + i, 1) <> Empty) Or
(Grid_Boolean.row + i = Grid_Boolean.Rows - 1) 'Check Upper row of Grid_boolean
          i = i + 1
          number_gridboolean_row_upper = Grid_Boolean.row + i
        Loop
        i = 0
        Do Until ((Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1 - i, 1) <> Empty) Or
(Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1 - i, 2) <> Empty)) 'Check Lower row of
Grid_boolean
          number_gridboolean_row_lower = Grid_Boolean.Rows - 2 - i
          i = i + 1
        Loop
        If number_gridboolean_row_upper <> Grid_Boolean.Rows - 1 Then
          For i = Grid_Boolean.row To number_gridboolean_row_lower
            Grid_Boolean.TextMatrix(i, 1) = Grid_Boolean.TextMatrix(i + 1, 1)
            Grid_Boolean.TextMatrix(i, 2) = Grid_Boolean.TextMatrix(i + 1, 2)
```

```
            Next i
            Grid_Boolean.TextMatrix(number_gridboolean_row_lower, 1) = Empty
            Grid_Boolean.TextMatrix(number_gridboolean_row_lower, 2) = Empty
            End If
          End If
        End If
      ElseIf Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) <> Empty Then 'check command line
        If (Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) <> Empty) And (Grid_Boolean.row =
Grid_Boolean.Rows - 1) Then 'one command line is last
          Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) = Empty
          Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = Empty
        ElseIf (Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) <> Empty) And
(Grid_Boolean.TextMatrix(Grid_Boolean.row + 1, 1) = Empty) And (Grid_Boolean.TextMatrix
(Grid_Boolean.row + 1, 2) = Empty) Then ' one command line is not last and next step is empty

          Do
            i = i + 1
            number_gridboolean_row_upper = Grid_Boolean.row + i
          Loop Until (Grid_Boolean.TextMatrix(Grid_Boolean.row + i, 1) <> Empty) Or
(Grid_Boolean.row + i = Grid_Boolean.Rows - 1) 'Check Upper row of Grid_boolean
          i = 0
          Do
            i = i + 1
            number_gridboolean_row_lower = Grid_Boolean.Rows - i
          Loop Until ((Grid_Boolean.TextMatrix(Grid_Boolean.Rows - i, 1) <> Empty) Or
(Grid_Boolean.TextMatrix(Grid_Boolean.Rows - i, 2) <> Empty)) 'Check Lower row of Grid_boolean
          For i = number_gridboolean_row_upper To number_gridboolean_row_lower
            Grid_Boolean.TextMatrix(i - 1, 1) = Grid_Boolean.TextMatrix(i, 1)
            Grid_Boolean.TextMatrix(i - 1, 2) = Grid_Boolean.TextMatrix(i, 2)
          Next i
          Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) = Empty
          Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = Empty
```

```
        Grid_Boolean.TextMatrix(number_gridboolean_row_lower, 1) = Empty
        Grid_Boolean.TextMatrix(number_gridboolean_row_lower, 2) = Empty
    ElseIf (Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) <> Empty) And
(Grid_Boolean.TextMatrix(Grid_Boolean.row + 1, 1) <> Empty) Then ' one command line is not last
and next step is not empty
        Do
          i = i + 1
          number_gridboolean_row_lower = Grid_Boolean.Rows - i
        Loop Until ((Grid_Boolean.TextMatrix(Grid_Boolean.Rows - i, 1) <> Empty) Or
(Grid_Boolean.TextMatrix(Grid_Boolean.Rows - i, 2) <> Empty)) 'Check Lower row of Grid_boolean
        For i = Grid_Boolean.row + 1 To number_gridboolean_row_lower
          Grid_Boolean.TextMatrix(i - 1, 1) = Grid_Boolean.TextMatrix(i, 1)
          Grid_Boolean.TextMatrix(i - 1, 2) = Grid_Boolean.TextMatrix(i, 2)
        Next i
        Grid_Boolean.TextMatrix(number_gridboolean_row_lower, 1) = Empty
        Grid_Boolean.TextMatrix(number_gridboolean_row_lower, 2) = Empty
    ElseIf (Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) <> Empty) And (Grid_Boolean.row =
Grid_Boolean.Rows - 2) Then  'two command line is last
        Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) = Empty
        Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = Empty
        Grid_Boolean.Rows = Grid_Boolean.Rows - 1
    ElseIf (Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) <> Empty) And
(Grid_Boolean.TextMatrix(Grid_Boolean.row + 1, 1) = Empty) And (Grid_Boolean.TextMatrix
(Grid_Boolean.row + 1, 2) <> Empty) And (Grid_Boolean.TextMatrix(Grid_Boolean.row + 2, 1) =
Empty) And (Grid_Boolean.TextMatrix(Grid_Boolean.row + 2, 2) = Empty) Then ' two command line
is not last and next step is empty
        Do
          i = i + 1
          number_gridboolean_row_upper = Grid_Boolean.row + i
        Loop Until (Grid_Boolean.TextMatrix(Grid_Boolean.row + i, 1) <> Empty) Or
(Grid_Boolean.row + i = Grid_Boolean.Rows - 1) 'Check Upper row of Grid_boolean
        i = 0
```

```
        Do
          i = i + 1
          number_gridboolean_row_lower = Grid_Boolean.Rows - i
        Loop Until ((Grid_Boolean.TextMatrix(Grid_Boolean.Rows - i, 1) <> Empty) Or
(Grid_Boolean.TextMatrix(Grid_Boolean.Rows - i, 2) <> Empty)) 'Check Lower row of Grid_boolean

        For i = number_gridboolean_row_upper To number_gridboolean_row_lower
          Grid_Boolean.TextMatrix(i - 2, 1) = Grid_Boolean.TextMatrix(i, 1)
          Grid_Boolean.TextMatrix(i - 2, 2) = Grid_Boolean.TextMatrix(i, 2)
        Next i
        Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) = Empty
        Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = Empty
        If (number_gridboolean_row_upper - Grid_Boolean.row - 1) > 2 Then
          Grid_Boolean.TextMatrix(Grid_Boolean.row + 1, 1) = Empty
          Grid_Boolean.TextMatrix(Grid_Boolean.row + 1, 2) = Empty
        End If
        Grid_Boolean.TextMatrix(number_gridboolean_row_lower, 1) = Empty
        Grid_Boolean.TextMatrix(number_gridboolean_row_lower, 2) = Empty
        Grid_Boolean.TextMatrix(number_gridboolean_row_lower - 1, 1) = Empty
        Grid_Boolean.TextMatrix(number_gridboolean_row_lower - 1, 2) = Empty
        Grid_Boolean.Rows = Grid_Boolean.Rows - 1
      ElseIf (Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) <> Empty) And
(Grid_Boolean.TextMatrix(Grid_Boolean.row + 1, 1) = Empty) And (Grid_Boolean.TextMatrix
(Grid_Boolean.row + 1, 2) <> Empty) And (Grid_Boolean.TextMatrix(Grid_Boolean.row + 2, 1) <>
Empty) Then ' two command line is not last and next step is not empty
        Do
          i = i + 1
          number_gridboolean_row_lower = Grid_Boolean.Rows - i
        Loop Until ((Grid_Boolean.TextMatrix(Grid_Boolean.Rows - i, 1) <> Empty) Or
(Grid_Boolean.TextMatrix(Grid_Boolean.Rows - i, 2) <> Empty)) 'Check Lower row of Grid_boolean
        For i = Grid_Boolean.row + 2 To number_gridboolean_row_lower
          Grid_Boolean.TextMatrix(i - 2, 1) = Grid_Boolean.TextMatrix(i, 1)
```

```
        Grid_Boolean.TextMatrix(i - 2, 2) = Grid_Boolean.TextMatrix(i, 2)
      Next i
      Grid_Boolean.TextMatrix(number_gridboolean_row_lower, 1) = Empty
      Grid_Boolean.TextMatrix(number_gridboolean_row_lower, 2) = Empty
      Grid_Boolean.TextMatrix(number_gridboolean_row_lower - 1, 1) = Empty
      Grid_Boolean.TextMatrix(number_gridboolean_row_lower - 1, 2) = Empty
      Grid_Boolean.Rows = Grid_Boolean.Rows - 1
ElseIf (Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) <> Empty) And (Grid_Boolean.row =
Grid_Boolean.Rows - 3) Then  'three command line is last
      Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) = Empty
      Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = Empty
      Grid_Boolean.Rows = Grid_Boolean.Rows - 2
    ElseIf (Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) <> Empty) And
(Grid_Boolean.TextMatrix(Grid_Boolean.row + 1, 1) = Empty) And (Grid_Boolean.TextMatrix
(Grid_Boolean.row + 1, 2) <> Empty) And (Grid_Boolean.TextMatrix(Grid_Boolean.row + 2, 1) =
Empty) And (Grid_Boolean.TextMatrix(Grid_Boolean.row + 2, 2) <> Empty) And
(Grid_Boolean.TextMatrix(Grid_Boolean.row + 3, 1) = Empty) And (Grid_Boolean.TextMatrix
(Grid_Boolean.row + 3, 2) = Empty) Then ' three command line is not last and next step is empty
      Do
        i = i + 1
        number_gridboolean_row_upper = Grid_Boolean.row + i
      Loop Until (Grid_Boolean.TextMatrix(Grid_Boolean.row + i, 1) <> Empty) Or
(Grid_Boolean.row + i = Grid_Boolean.Rows - 1) 'Check Upper row of Grid_boolean
      i = 0
      Do
        i = i + 1
        number_gridboolean_row_lower = Grid_Boolean.Rows - i
      Loop Until ((Grid_Boolean.TextMatrix(Grid_Boolean.Rows - i, 1) <> Empty) Or
(Grid_Boolean.TextMatrix(Grid_Boolean.Rows - i, 2) <> Empty)) 'Check Lower row of Grid_boolean
      For i = number_gridboolean_row_upper To number_gridboolean_row_lower
        Grid_Boolean.TextMatrix(i - 3, 1) = Grid_Boolean.TextMatrix(i, 1)
```

```
        Grid_Boolean.TextMatrix(i - 3, 2) = Grid_Boolean.TextMatrix(i, 2)
      Next i
      Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) = Empty
      Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = Empty
      If (number_gridboolean_row_upper - Grid_Boolean.row - 1) > 3 Then
        Grid_Boolean.TextMatrix(Grid_Boolean.row + 1, 1) = Empty
        Grid_Boolean.TextMatrix(Grid_Boolean.row + 1, 2) = Empty
        Grid_Boolean.TextMatrix(Grid_Boolean.row + 2, 1) = Empty
        Grid_Boolean.TextMatrix(Grid_Boolean.row + 2, 2) = Empty
      End If
      Grid_Boolean.TextMatrix(number_gridboolean_row_lower, 1) = Empty
      Grid_Boolean.TextMatrix(number_gridboolean_row_lower, 2) = Empty
      Grid_Boolean.TextMatrix(number_gridboolean_row_lower - 1, 1) = Empty
      Grid_Boolean.TextMatrix(number_gridboolean_row_lower - 1, 2) = Empty
      Grid_Boolean.TextMatrix(number_gridboolean_row_lower - 2, 1) = Empty
      Grid_Boolean.TextMatrix(number_gridboolean_row_lower - 2, 2) = Empty
      Grid_Boolean.Rows = Grid_Boolean.Rows - 2
    ElseIf (Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) <> Empty) And
(Grid_Boolean.TextMatrix(Grid_Boolean.row + 1, 1) = Empty) And (Grid_Boolean.TextMatrix
(Grid_Boolean.row + 1, 2) <> Empty) And (Grid_Boolean.TextMatrix(Grid_Boolean.row + 2, 1) =
Empty) And (Grid_Boolean.TextMatrix(Grid_Boolean.row + 2, 2) <> Empty) And
(Grid_Boolean.TextMatrix(Grid_Boolean.row + 3, 1) <> Empty) Then ' three command line is not last
and next step is not empty
      Do
        i = i + 1
        number_gridboolean_row_lower = Grid_Boolean.Rows - i
      Loop Until ((Grid_Boolean.TextMatrix(Grid_Boolean.Rows - i, 1) <> Empty) Or
(Grid_Boolean.TextMatrix(Grid_Boolean.Rows - i, 2) <> Empty)) 'Check Lower row of Grid_boolean
      For i = Grid_Boolean.row + 3 To number_gridboolean_row_lower
        Grid_Boolean.TextMatrix(i - 3, 1) = Grid_Boolean.TextMatrix(i, 1)
        Grid_Boolean.TextMatrix(i - 3, 2) = Grid_Boolean.TextMatrix(i, 2)
      Next i
```

```
        Grid_Boolean.TextMatrix(number_gridboolean_row_lower, 1) = Empty
        Grid_Boolean.TextMatrix(number_gridboolean_row_lower, 2) = Empty
        Grid_Boolean.TextMatrix(number_gridboolean_row_lower - 1, 1) = Empty
        Grid_Boolean.TextMatrix(number_gridboolean_row_lower - 1, 2) = Empty
        Grid_Boolean.TextMatrix(number_gridboolean_row_lower - 2, 1) = Empty
        Grid_Boolean.TextMatrix(number_gridboolean_row_lower - 2, 2) = Empty
        Grid_Boolean.Rows = Grid_Boolean.Rows - 2
      End If
    Grid_Boolean.SetFocus
    End If
    clear_command_line
    check_command_line
End Sub
-------------------------------------------------------------------------------------------------------------------

Private Sub botton_edit_Click() 'edit command
    If (Grid_Boolean.col = 2) And (Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) <> Empty) Then
      If Len(Grid_Boolean.TextMatrix(Grid_Boolean.row, 2)) = 4 Then
        If Len(Label_data.Caption) = 1 Then
          Label_word(1).Caption = "00" + Label_data.Caption
        ElseIf Len(Label_data.Caption) = 2 Then
          Label_word(1).Caption = "0" + Label_data.Caption
        ElseIf Len(Label_data.Caption) = 3 Then
          Label_word(1).Caption = Label_data.Caption
        Else
          Label_word(1).Caption = Mid(Label_data.Caption, 2, 3)
        End If
        If Mid(Grid_Boolean.TextMatrix(Grid_Boolean.row, 2), 1, 1) = "#" Then
          Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = "#" + Label_word(1).Caption
        ElseIf Mid(Grid_Boolean.TextMatrix(Grid_Boolean.row, 2), 1, 1) = " " Then
          Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = " " + Label_word(1).Caption
        End If
      ElseIf Len(Grid_Boolean.TextMatrix(Grid_Boolean.row, 2)) = 5 Then
```

```
        If Len(Label_data.Caption) = 1 Then
           Label_word(1).Caption = "000" + Label_data.Caption
        ElseIf Len(Label_data.Caption) = 2 Then
           Label_word(1).Caption = "00" + Label_data.Caption
        ElseIf Len(Label_data.Caption) = 3 Then
           Label_word(1).Caption = "0" + Label_data.Caption
        Else
           Label_word(1).Caption = Label_data.Caption
        End If
        Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = Label_memory(1).Caption + Label_word
(1).Caption
      End If
    Grid_Boolean.SetFocus
    End If
End Sub
```

```
Private Sub botton_insert_Click() 'insert command
If (Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) <> Empty) Then
   If Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 2) = Empty Then
      i = Grid_Boolean.Rows - 1
      Do
        Grid_Boolean.TextMatrix(i, 1) = Grid_Boolean.TextMatrix(i - 1, 1)
        Grid_Boolean.TextMatrix(i, 2) = Grid_Boolean.TextMatrix(i - 1, 2)
        i = i - 1
      Loop Until i < Grid_Boolean.row
      Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) = Empty
      Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = Empty
    End If
 If Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 1) = Empty Then
    Grid_Boolean.row = Grid_Boolean.row + 1
 End If
    Grid_Boolean.col = 1
```

```
    Grid_Boolean.SetFocus
End If
clear_command_line
check_command_line
End Sub
```

```
Private Sub botton_ladder_clear_Click()
For i = 1 To 50
    For j = 1 To 30
        Grid_Ladder.TextMatrix(i, j) = Empty
        scanmap(i, j) = Empty
        Grid_Ladder.row = i
        Grid_Ladder.col = j
        Set Grid_Ladder.CellPicture = pictureclip_ladder3.GraphicCell(0)
    Next j
Next i
End Sub
```

```
Private Sub Button_ladder_Cnt_Click() 'CNT command of Ladder
function_ladder.Label_ladder_function.Caption = "Counter"
Call show_value_ladder_tim_cnt
function_ladder.Option_function_ladder_tim.Enabled = False
function_ladder.Option_function_ladder_cnt.Enabled = False
function_ladder.Option_function_ladder_tim.Value = False
function_ladder.Option_function_ladder_cnt.Value = False
Call option_type_memory_ladder_all_false
End Sub
```

```
Private Sub Button_ladder_delete_Click()
checkladder = False
End Sub
```

```
Private Sub Button_ladder_LD_Click() 'LD command of Ladder
function_ladder.Label_ladder_function.Caption = "Input"
Call show_value_ladder1
function_ladder.Option_function_ladder_tim.Enabled = True
function_ladder.Option_function_ladder_cnt.Enabled = True
function_ladder.Option_function_ladder_tim.Value = False
function_ladder.Option_function_ladder_cnt.Value = False
Call option_type_memory_ladder_all_true
End Sub
```

```
Private Sub Button_ladder_Out_Click() 'OUT command of Ladder
function_ladder.Label_ladder_function.Caption = "Output"
Call show_value_ladder1
function_ladder.Option_function_ladder_tim.Enabled = False
function_ladder.Option_function_ladder_cnt.Enabled = False
function_ladder.Option_function_ladder_tim.Value = False
function_ladder.Option_function_ladder_cnt.Value = False
Call option_type_memory_ladder_all_true
End Sub
```

```
Private Sub Button_ladder_Tim_Click() 'TIM command of Ladder
function_ladder.Label_ladder_function.Caption = "Timer"
Call show_value_ladder_tim_cnt
function_ladder.Option_function_ladder_tim.Enabled = False
function_ladder.Option_function_ladder_cnt.Enabled = False
function_ladder.Option_function_ladder_tim.Value = False
function_ladder.Option_function_ladder_cnt.Value = False
Call option_type_memory_ladder_all_false
End Sub
```

```
Private Sub Combo_ladder_Click()
function_ladder.Option_function_ladder_tim.Enabled = False
```

```
function_ladder.Option_function_ladder_cnt.Enabled = False
function_ladder.Option_function_ladder_tim.Value = False
function_ladder.Option_function_ladder_cnt.Value = False
If Combo_ladder.ListIndex = 0 Then
   function_ladder.Label_ladder_function.Caption = "NOP(01)"
   Call show_nonvalue_ladder
   Call option_type_memory_ladder_all_false
ElseIf Combo_ladder.ListIndex = 1 Then
   function_ladder.Label_ladder_function.Caption = "END(01)"
   Call show_nonvalue_ladder
   Call option_type_memory_ladder_all_false
ElseIf Combo_ladder.ListIndex = 2 Then
   function_ladder.Label_ladder_function.Caption = "IL(02)"
   Call show_nonvalue_ladder
   Call option_type_memory_ladder_all_false
ElseIf Combo_ladder.ListIndex = 3 Then
   function_ladder.Label_ladder_function.Caption = "ILC(03)"
   Call show_nonvalue_ladder
   Call option_type_memory_ladder_all_false
ElseIf Combo_ladder.ListIndex = 4 Then
   function_ladder.Label_ladder_function.Caption = "JMP(04)"
   Call show_nonvalue_ladder
   Call option_type_memory_ladder_all_false
ElseIf Combo_ladder.ListIndex = 5 Then
   function_ladder.Label_ladder_function.Caption = "JME(05)"
   Call show_nonvalue_ladder
   Call option_type_memory_ladder_all_false
ElseIf Combo_ladder.ListIndex = 6 Then
   function_ladder.Label_ladder_function.Caption = "SFT(10)"
   Call show_value_ladder2
   Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 7 Then
```

```
    function_ladder.Label_ladder_function.Caption = "KEEP(11)"
    Call show_value_ladder1
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 8 Then
    function_ladder.Label_ladder_function.Caption = "DIFU(13)"
    Call show_value_ladder1
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 9 Then
    function_ladder.Label_ladder_function.Caption = "DIFD(14)"
    Call show_value_ladder1
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 10 Then
    function_ladder.Label_ladder_function.Caption = "WSFT(16)"
    Call show_value_ladder2
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 11 Then
    function_ladder.Label_ladder_function.Caption = "CMP(20)"
    Call show_value_ladder2
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 12 Then
    function_ladder.Label_ladder_function.Caption = "MOV(21)"
    Call show_value_ladder2
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 13 Then
    function_ladder.Label_ladder_function.Caption = "MVN(22)"
    Call show_value_ladder2
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 14 Then
    function_ladder.Label_ladder_function.Caption = "BIN(23)"
    Call show_value_ladder2
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 15 Then
```

```
    function_ladder.Label_ladder_function.Caption = "BCD(24)"
    Call show_value_ladder2
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 16 Then
    function_ladder.Label_ladder_function.Caption = "ASL(25)"
    Call show_value_ladder1
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 17 Then
    function_ladder.Label_ladder_function.Caption = "ASR(26)"
    Call show_value_ladder1
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 18 Then
    function_ladder.Label_ladder_function.Caption = "ROL(27)"
    Call show_value_ladder1
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 19 Then
    function_ladder.Label_ladder_function.Caption = "ROR(28)"
    Call show_value_ladder1
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 20 Then
    function_ladder.Label_ladder_function.Caption = "COM(29)"
    Call show_value_ladder1
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 21 Then
    function_ladder.Label_ladder_function.Caption = "ADD(30)"
    Call show_value_ladder3
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 22 Then
    function_ladder.Label_ladder_function.Caption = "SUB(31)"
    Call show_value_ladder3
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 23 Then
```

```
    function_ladder.Label_ladder_function.Caption = "MUL(32)"
    Call show_value_ladder3
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 24 Then
    function_ladder.Label_ladder_function.Caption = "DIV(33)"
    Call show_value_ladder3
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 25 Then
    function_ladder.Label_ladder_function.Caption = "ANDW(34)"
    Call show_value_ladder3
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 26 Then
    function_ladder.Label_ladder_function.Caption = "ORW(35)"
    Call show_value_ladder3
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 27 Then
    function_ladder.Label_ladder_function.Caption = "XORW(36)"
    Call show_value_ladder3
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 28 Then
    function_ladder.Label_ladder_function.Caption = "XNRW(37)"
    Call show_value_ladder3
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 29 Then
    function_ladder.Label_ladder_function.Caption = "INC(38)"
    Call show_value_ladder1
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 30 Then
    function_ladder.Label_ladder_function.Caption = "DEC(39)"
    Call show_value_ladder1
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 31 Then
```

```
    function_ladder.Label_ladder_function.Caption = "STC(40)"
    Call show_nonvalue_ladder
    Call option_type_memory_ladder_all_false
ElseIf Combo_ladder.ListIndex = 32 Then
    function_ladder.Label_ladder_function.Caption = "CLC(41)"
    Call show_nonvalue_ladder
    Call option_type_memory_ladder_all_false
ElseIf Combo_ladder.ListIndex = 33 Then
    function_ladder.Label_ladder_function.Caption = "MSG(47)"
    Call show_value_ladder1
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 34 Then
    function_ladder.Label_ladder_function.Caption = "ADB(50)"
    Call show_value_ladder3
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 35 Then
    function_ladder.Label_ladder_function.Caption = "SBB(51)"
    Call show_value_ladder3
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 36 Then
    function_ladder.Label_ladder_function.Caption = "MLB(52)"
    Call show_value_ladder3
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 37 Then
    function_ladder.Label_ladder_function.Caption = "DVB(53)"
    Call show_value_ladder3
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 38 Then
    function_ladder.Label_ladder_function.Caption = "RDI(70)"
    Call show_value_ladder2
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 39 Then
```

```
    function_ladder.Label_ladder_function.Caption = "WRO(71)"
    Call show_value_ladder2
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 40 Then
    function_ladder.Label_ladder_function.Caption = "SLD(74)"
    Call show_value_ladder2
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 41 Then
    function_ladder.Label_ladder_function.Caption = "SRD(75)"
    Call show_value_ladder2
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 42 Then
    function_ladder.Label_ladder_function.Caption = "MLPX(76)"
    Call show_value_ladder3
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 43 Then
    function_ladder.Label_ladder_function.Caption = "SDEC(78)"
    Call show_value_ladder3
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 44 Then
    function_ladder.Label_ladder_function.Caption = "TKY(79)"
    Call show_value_ladder2
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 45 Then
    function_ladder.Label_ladder_function.Caption = "7SEG(80)"
    Call show_value_ladder2
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 46 Then
    function_ladder.Label_ladder_function.Caption = "MOVB(82)"
    Call show_value_ladder3
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 47 Then
```

```
    function_ladder.Label_ladder_function.Caption = "MOVD(83)"
    Call show_value_ladder3
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 48 Then
    function_ladder.Label_ladder_function.Caption = "SFTR(84)"
    Call show_value_ladder2
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 49 Then
    function_ladder.Label_ladder_function.Caption = "ASC(86)"
    Call show_value_ladder3
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 50 Then
    function_ladder.Label_ladder_function.Caption = "TXD(87)"
    Call show_value_ladder3
    Call option_type_memory_ladder_all_true
ElseIf Combo_ladder.ListIndex = 51 Then
    function_ladder.Label_ladder_function.Caption = "RXD(88)"
    Call show_value_ladder3
    Call option_type_memory_ladder_all_true
End If
checkladder = True
End Sub
-----------------------------------------------------------------------------------------------------------------------

Private Sub Form_Load() 'set valve intitial of PROPGRAM
    'Me.Left = GetSetting(App.Title, "Settings", "MainLeft", 1000)
    'Me.Top = GetSetting(App.Title, "Settings", "MainTop", 1000)
    'Me.Width = GetSetting(App.Title, "Settings", "MainWidth", 6500)
    'Me.Height = GetSetting(App.Title, "Settings", "MainHeight", 6500)
    Grid_Boolean.ColWidth(1) = 1650
    Grid_Boolean.ColWidth(2) = 2500
    Grid_Boolean.CellAlignment = 1
    Grid_Boolean.TextMatrix(0, 1) = "  Boolean"
```

```
Grid_Boolean.TextMatrix(0, 2) = "  I/O Assingment"
Grid_Boolean.row = 1
Grid_Boolean.col = 1
checkboolean = 0
checkbooleansequence = 1
Label_data.Caption = "0000"
numbercommandline = 2001
Grid_Boolean.Rows = 2001
For i = 1 To 50
   Grid_Ladder.RowHeight(i) = 900
   For j = 1 To 30
      Grid_Ladder.ColWidth(j) = 1700
   Next j
Next i
For i = 1 To 8
grid_memory_ir.RowHeight(0) = 300
grid_memory_ir.ColWidth(0) = 900
   For j = 1 To 10
      grid_memory_ir.ColWidth(j) = 900
      grid_memory_ir.TextMatrix(0, j) = j - 1
      grid_memory_ir.TextMatrix(i, j) = "0000"
   Next j
   grid_memory_ir.TextMatrix(i, 0) = i - 1 & "x"
Next i
For i = 1 To 10
grid_memory_hr.RowHeight(0) = 300
grid_memory_hr.ColWidth(0) = 900
   For j = 1 To 10
      grid_memory_hr.ColWidth(j) = 900
      grid_memory_hr.TextMatrix(0, j) = j - 1
      grid_memory_hr.TextMatrix(i, j) = "0000"
   Next j
```

```
    grid_memory_hr.TextMatrix(i, 0) = i - 1 & "x"
Next i
For i = 1 To 7
grid_memory_lr.RowHeight(0) = 300
grid_memory_lr.ColWidth(0) = 900
    For j = 1 To 10
        grid_memory_lr.ColWidth(j) = 900
        grid_memory_lr.TextMatrix(0, j) = j - 1
        grid_memory_lr.TextMatrix(i, j) = "0000"
    Next j
    grid_memory_lr.TextMatrix(i, 0) = i - 1 & "x"
Next i
For i = 1 To 100
grid_memory_dm.RowHeight(0) = 300
grid_memory_dm.ColWidth(0) = 900
    For j = 1 To 10
        grid_memory_dm.ColWidth(j) = 900
        grid_memory_dm.TextMatrix(0, j) = j - 1
        grid_memory_dm.TextMatrix(i, j) = "0000"
    Next j
    grid_memory_dm.TextMatrix(i, 0) = i - 1 & "x"
Next i
For i = 1 To 26
grid_memory_tc.RowHeight(0) = 300
grid_memory_tc.ColWidth(0) = 900
    For j = 1 To 10
        grid_memory_tc.ColWidth(j) = 900
        grid_memory_tc.TextMatrix(0, j) = j - 1
        grid_memory_tc.TextMatrix(i, j) = "0000"
    Next j
    grid_memory_tc.TextMatrix(i, 0) = i - 1 & "x"
Next i
```

```
    Load function_ladder
    Load find_booleann
    Load find_ladder
    Load communication_setting
    Button_ladder_point.Value = True
    checkscan = False
    checkladder = False
End Sub
```

```
Private Sub Form_Unload(Cancel As Integer) 'Unload Program when close.
Unload function_ladder
Unload communication_setting
Unload find_booleann
Unload find_ladder
End Sub
```

```
Private Sub botton_add_Click() 'add function
If ((Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) = Empty) And (Grid_Boolean.TextMatrix
(Grid_Boolean.row, 2) = Empty)) And ((Grid_Boolean.TextMatrix(Grid_Boolean.row - 1, 1) <>
Empty) Or (Grid_Boolean.TextMatrix(Grid_Boolean.row - 1, 2) <> Empty)) Then
   If checkboolean = 0 Then
      Grid_Boolean.TextMatrix(Grid_Boolean.row, 1) = Label_fun.Caption
      Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) = Empty
      If (Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 1) = Empty) And
(Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 2) = Empty) Then
         Grid_Boolean.row = Grid_Boolean.row + 1
      ElseIf Grid_Boolean.row <> Grid_Boolean.Rows - 1 Then
         If ((Grid_Boolean.TextMatrix(Grid_Boolean.row - 1, 1) <> Empty) Or
(Grid_Boolean.TextMatrix(Grid_Boolean.row - 1, 2) <> Empty)) And ((Grid_Boolean.TextMatrix
(Grid_Boolean.row + 1, 1) = Empty) And (Grid_Boolean.TextMatrix(Grid_Boolean.row + 1, 2) =
Empty)) And ((Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 1) <> Empty) Or
(Grid_Boolean.TextMatrix(Grid_Boolean.Rows - 1, 2) <> Empty)) Then
```

```
        Grid_Boolean.row = Grid_Boolean.row + 1
      End If
    End If
    Grid_Boolean.col = 1
    Grid_Boolean.SetFocus
    check_command_line
  ElseIf checkboolean = 1 Then 'one command line
    Add_boolean_grid (checkbooleansequence)
    Label_word(1).BackColor = &HFF&
    If Label_memory(1).BackColor = &HC0FFC0 Then
      Label_memory(1).BackColor = &HFF&
    End If
    Timer_1.Enabled = True
  ElseIf checkboolean = 2 Then 'two command line
    If checkboolean = checkbooleansequence Then
      Grid_Boolean.Rows = Grid_Boolean.Rows + 1
      Add_boolean_grid (checkbooleansequence)
      Label_memory(2).BackColor = &HFF&
      Label_word(2).BackColor = &HFF&
      Timer_2.Enabled = True
    ElseIf checkboolean <> checkbooleansequence Then
      Select_option (checkbooleansequence)
      If Len(Label_data.Caption) = 1 Then
        Label_word(checkbooleansequence).Caption = "000" + Label_data.Caption
      ElseIf Len(Label_data.Caption) = 2 Then
        Label_word(checkbooleansequence).Caption = "00" + Label_data.Caption
      ElseIf Len(Label_data.Caption) = 3 Then
        Label_word(checkbooleansequence).Caption = "0" + Label_data.Caption
      Else
        Label_word(checkbooleansequence).Caption = Label_data.Caption
      End If
      Label_memory(1).BackColor = &HFF&
```

```
        Label_word(1).BackColor = &HFF&
        Timer_1.Enabled = True
        checkbooleansequence = checkbooleansequence + 1
      End If
    ElseIf checkboolean = 3 Then 'three command line
      If checkboolean = checkbooleansequence Then
        Grid_Boolean.Rows = Grid_Boolean.Rows + 2
        Add_boolean_grid (checkbooleansequence)
        Label_memory(3).BackColor = &HFF&
        Label_word(3).BackColor = &HFF&
        Timer_3.Enabled = True
      ElseIf checkboolean <> checkbooleansequence Then
        Select_option (checkbooleansequence)
        If Len(Label_data.Caption) = 1 Then
          Label_word(checkbooleansequence).Caption = "000" + Label_data.Caption
        ElseIf Len(Label_data.Caption) = 2 Then
          Label_word(checkbooleansequence).Caption = "00" + Label_data.Caption
        ElseIf Len(Label_data.Caption) = 3 Then
          Label_word(checkbooleansequence).Caption = "0" + Label_data.Caption
        Else
          Label_word(checkbooleansequence).Caption = Label_data.Caption
        End If
        If checkbooleansequence = 1 Then
          Label_memory(1).BackColor = &HFF&
          Label_word(1).BackColor = &HFF&
          Timer_1.Enabled = True
        ElseIf checkbooleansequence = 2 Then
          Label_memory(2).BackColor = &HFF&
          Label_word(2).BackColor = &HFF&
          Timer_2.Enabled = True
        End If
        checkbooleansequence = checkbooleansequence + 1
```

```
    End If
  ElseIf checkboolean = 4 Then 'tim/cnt command line
    If checkbooleansequence = 2 Then
      Grid_Boolean.Rows = Grid_Boolean.Rows + 1
      Add_boolean_grid (checkbooleansequence)
      Label_memory(2).BackColor = &HFF&
      Label_word(2).BackColor = &HFF&
      Timer_2.Enabled = True
    ElseIf checkboolean <> checkbooleansequence Then
      If Len(Label_data.Caption) = 1 Then
        Label_word(checkbooleansequence).Caption = "00" + Label_data.Caption
      ElseIf Len(Label_data.Caption) = 2 Then
        Label_word(checkbooleansequence).Caption = "0" + Label_data.Caption
      ElseIf Len(Label_data.Caption) = 3 Then
        Label_word(checkbooleansequence).Caption = Label_data.Caption
      End If
    Label_word(1).BackColor = &HFF&
    Timer_1.Enabled = True
    checkbooleansequence = checkbooleansequence + 1
    End If
  End If
End If
End Sub
```

```
Private Sub botton_cnt_Click() 'cnt command line
Label_fun.BackColor = &HC0FFC0
If Label_fun.Caption = "LD" Or Label_fun.Caption = "AND" Or Label_fun.Caption = "OR" Or
Label_fun.Caption = "LDNOT" Or Label_fun.Caption = "ANDNOT" Or Label_fun.Caption =
"ORNOT" Then
  Label_fun.Caption = Label_fun.Caption + "CNT"
  Call ldtim_ldcnt
  Call value_ldtim_ldcnt
```

```
    Call disable_value_all_option
    Call non_value_option
    Call check_data1
    Label_data.Caption = "000"
Else
    Label_fun.Caption = "CNT"
    Call tim_cnt
    Call value_tim_cnt
    Call disable_value_all_option
    Call non_value_option
    Call check_data4
    Label_data.Caption = "000"
End If
End Sub
-----------------------------------------------------------------------------------------------------------------
Private Sub botton_ld_Click() 'ld command line
Label_fun.BackColor = &HC0FFC0
If Label_fun.Caption = "AND" Or Label_fun.Caption = "OR" Then
    Label_fun.Caption = Label_fun.Caption + "LD"
    Call non_word
    Call non_value
    Call disable_value_all_option
    Call non_value_option
    Call non_check_data
    Label_data.Caption = Empty
Else
    Label_fun.Caption = "LD"
    Call word1
    Call value_zero1
    Call enable_value_all_option
    Option_ir.Value = True
    Call check_data1
```

```
    Label_data.Caption = "0000"
    Label_memory(2).Caption = Empty
    Label_memory(3).Caption = Empty
End If
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub botton_not_Click() ' not command line
If Label_fun.Caption = "LD" Or Label_fun.Caption = "AND" Or Label_fun.Caption = "OR" Or
Label_fun.Caption = "OUT" Then
    Label_fun.Caption = Label_fun.Caption + "NOT"
    Call word1
    Call value_zero1
    Call enable_value_all_option
    Option_ir.Value = True
    Call check_data1
    Label_data.Caption = "0000"
ElseIf Label_fun.Caption = "LDNOT" Or Label_fun.Caption = "ANDNOT" Or Label_fun.Caption =
"ORNOT" Or Label_fun.Caption = "OUTNOT" Then
    Label_fun.Caption = Mid(Label_fun.Caption, 1, Len(Label_fun.Caption) - 3)
    Call word1
    Call value_zero1
    Call enable_value_all_option
    Option_ir.Value = True
    Call check_data1
    Label_data.Caption = "0000"
ElseIf Label_fun.Caption = "LDTIM" Or Label_fun.Caption = "ANDTIM" Or Label_fun.Caption =
"ORTIM" Or Label_fun.Caption = "LDCNT" Or Label_fun.Caption = "ANDCNT" Or
Label_fun.Caption = "ORCNT" Then
    lenfun = Len(Label_fun.Caption)
    Label_fun.Caption = Mid(Label_fun.Caption, 1, lenfun - 3) + "NOT" + Mid(Label_fun.Caption,
lenfun - 2, 3)
    Call word1
```

```
    Label_memory(1).BackColor = &H8000000F
    Call value_zero1
    Label_word(1).Caption = "000"
    Label_memory(1).Caption = Empty
    Call disable_value_all_option
    Call non_value_option
    Call check_data1
ElseIf Label_fun.Caption = "LDNOTTIM" Or Label_fun.Caption = "ANDNOTTIM" Or
Label_fun.Caption = "ORNOTTIM" Or Label_fun.Caption = "LDNOTCNT" Or Label_fun.Caption =
"ANDNOTCNT" Or Label_fun.Caption = "ORNOTCNT" Then
    lenfun = Len(Label_fun.Caption)
    Label_fun.Caption = Mid(Label_fun.Caption, 1, lenfun - 6) + Mid(Label_fun.Caption, lenfun - 2,
lenfun)
    Call word1
    Label_memory(1).BackColor = &H8000000F
    Call value_zero1
    Label_word(1).Caption = "000"
    Label_memory(1).Caption = Empty
    Call disable_value_all_option
    Call non_value_option
    Call check_data1
End If
End Sub
```

```
Private Sub botton_tim_Click() 'tim command line
Label_fun.BackColor = &HC0FFC0
If Label_fun.Caption = "LD" Or Label_fun.Caption = "AND" Or Label_fun.Caption = "OR" Or
Label_fun.Caption = "LDNOT" Or Label_fun.Caption = "ANDNOT" Or Label_fun.Caption =
"ORNOT" Then
    Label_fun.Caption = Label_fun.Caption + "TIM"
    Call ldtim_ldcnt
    Call value_ldtim_ldcnt
```

```
    Call disable_value_all_option
    Call non_value_option
    Call check_data1
    Label_data.Caption = "000"
Else
    Label_fun.Caption = "TIM"
    Call tim_cnt
    Call value_tim_cnt
    Call disable_value_all_option
    Call non_value_option
    Call check_data4
    Label_data.Caption = "000"
End If
End Sub
-----------------------------------------------------------------------------------------------------------
Private Sub botton_clear_Click() 'clear Grid_boolean
    Grid_Boolean.Clear
    Grid_Boolean.TextMatrix(0, 1) = "   Boolean"
    Grid_Boolean.TextMatrix(0, 2) = "  I/O Assingment"
    Grid_Boolean.Rows = numbercommandline 'Fix number rows = 2000
End Sub
-----------------------------------------------------------------------------------------------------------
Private Sub botton_0_Click() 'botton zero of boolean
databoolean = Label_data.Caption
IfLenght_data(databoolean) > 3 Then
    Label_data.Caption = "0"
ElseIf (Lenght_data(databoolean) = 3) And (Label_fun.Caption = "TIM" Or Label_fun.Caption =
"CNT" Or Label_fun.Caption = "LDTIM" Or Label_fun.Caption = "ANDTIM" Or Label_fun.Caption =
"ORTIM" Or Label_fun.Caption = "LDCNT" Or Label_fun.Caption = "ANDCNT" Or
Label_fun.Caption = "ORCNT" Or Label_fun.Caption = "LDNOTTIM" Or Label_fun.Caption =
"ANDNOTTIM" Or Label_fun.Caption = "ORNOTTIM" Or Label_fun.Caption = "LDNOTCNT" Or
Label_fun.Caption = "ANDNOTCNT" Or Label_fun.Caption = "ORNOTCNT") Then
```

```
   Label_data.Caption = "0"
Else
   Label_data.Caption = Shift_data(databoolean, "0")
End If
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub botton_1_Click() 'botton one of boolean
databoolean = Label_data.Caption
If Lenght_data(databoolean) > 3 Then
   Label_data.Caption = "1"
ElseIf (Lenght_data(databoolean) = 3) And (Label_fun.Caption = "TIM" Or Label_fun.Caption =
"CNT" Or Label_fun.Caption = "LDTIM" Or Label_fun.Caption = "ANDTIM" Or Label_fun.Caption =
"ORTIM" Or Label_fun.Caption = "LDCNT" Or Label_fun.Caption = "ANDCNT" Or
Label_fun.Caption = "ORCNT" Or Label_fun.Caption = "LDNOTTIM" Or Label_fun.Caption =
"ANDNOTTIM" Or Label_fun.Caption = "ORNOTTIM" Or Label_fun.Caption = "LDNOTCNT" Or
Label_fun.Caption = "ANDNOTCNT" Or Label_fun.Caption = "ORNOTCNT") Then
   Label_data.Caption = "1"
Else
   Label_data.Caption = Shift_data(databoolean, "1")
End If
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub botton_2_Click() 'botton two of boolean
databoolean = Label_data.Caption
If Lenght_data(databoolean) > 3 Then
   Label_data.Caption = "2"
ElseIf (Lenght_data(databoolean) = 3) And (Label_fun.Caption = "TIM" Or Label_fun.Caption =
"CNT" Or Label_fun.Caption = "LDTIM" Or Label_fun.Caption = "ANDTIM" Or Label_fun.Caption =
"ORTIM" Or Label_fun.Caption = "LDCNT" Or Label_fun.Caption = "ANDCNT" Or
Label_fun.Caption = "ORCNT" Or Label_fun.Caption = "LDNOTTIM" Or Label_fun.Caption =
"ANDNOTTIM" Or Label_fun.Caption = "ORNOTTIM" Or Label_fun.Caption = "LDNOTCNT" Or
Label_fun.Caption = "ANDNOTCNT" Or Label_fun.Caption = "ORNOTCNT") Then
```

```
   Label_data.Caption = "2"
Else
   Label_data.Caption = Shift_data(databoolean, "2")
End If
End Sub
```

```
Private Sub botton_3_Click() 'botton three of boolean
databoolean = Label_data.Caption
If Lenght_data(databoolean) > 3 Then
   Label_data.Caption = "3"
ElseIf (Lenght_data(databoolean) = 3) And (Label_fun.Caption = "TIM" Or Label_fun.Caption =
"CNT" Or Label_fun.Caption = "LDTIM" Or Label_fun.Caption = "ANDTIM" Or Label_fun.Caption =
"ORTIM" Or Label_fun.Caption = "LDCNT" Or Label_fun.Caption = "ANDCNT" Or
Label_fun.Caption = "ORCNT" Or Label_fun.Caption = "LDNOTTIM" Or Label_fun.Caption =
"ANDNOTTIM" Or Label_fun.Caption = "ORNOTTIM" Or Label_fun.Caption = "LDNOTCNT" Or
Label_fun.Caption = "ANDNOTCNT" Or Label_fun.Caption = "ORNOTCNT") Then
   Label_data.Caption = "3"
Else
   Label_data.Caption = Shift_data(databoolean, "3")
End If
End Sub
```

```
Private Sub botton_4_Click() 'botton four of boolean
databoolean = Label_data.Caption
If Lenght_data(databoolean) > 3 Then
   Label_data.Caption = "4"
ElseIf (Lenght_data(databoolean) = 3) And (Label_fun.Caption = "TIM" Or Label_fun.Caption =
"CNT" Or Label_fun.Caption = "LDTIM" Or Label_fun.Caption = "ANDTIM" Or Label_fun.Caption =
"ORTIM" Or Label_fun.Caption = "LDCNT" Or Label_fun.Caption = "ANDCNT" Or
Label_fun.Caption = "ORCNT" Or Label_fun.Caption = "LDNOTTIM" Or Label_fun.Caption =
"ANDNOTTIM" Or Label_fun.Caption = "ORNOTTIM" Or Label_fun.Caption = "LDNOTCNT" Or
Label_fun.Caption = "ANDNOTCNT" Or Label_fun.Caption = "ORNOTCNT") Then
```

```
   Label_data.Caption = "4"
Else
   Label_data.Caption = Shift_data(databoolean, "4")
End If
End Sub
```

```
Private Sub botton_5_Click() 'botton five of boolean
databoolean = Label_data.Caption
If Lenght_data(databoolean) > 3 Then
   Label_data.Caption = "5"
ElseIf (Lenght_data(databoolean) = 3) And (Label_fun.Caption = "TIM" Or Label_fun.Caption =
"CNT" Or Label_fun.Caption = "LDTIM" Or Label_fun.Caption = "ANDTIM" Or Label_fun.Caption =
"ORTIM" Or Label_fun.Caption = "LDCNT" Or Label_fun.Caption = "ANDCNT" Or
Label_fun.Caption = "ORCNT" Or Label_fun.Caption = "LDNOTTIM" Or Label_fun.Caption =
"ANDNOTTIM" Or Label_fun.Caption = "ORNOTTIM" Or Label_fun.Caption = "LDNOTCNT" Or
Label_fun.Caption = "ANDNOTCNT" Or Label_fun.Caption = "ORNOTCNT") Then
   Label_data.Caption = "5"
Else
   Label_data.Caption = Shift_data(databoolean, "5")
End If
End Sub
```

```
Private Sub botton_6_Click() 'botton six of boolean
databoolean = Label_data.Caption
If Lenght_data(databoolean) > 3 Then
   Label_data.Caption = "6"
ElseIf (Lenght_data(databoolean) = 3) And (Label_fun.Caption = "TIM" Or Label_fun.Caption =
"CNT" Or Label_fun.Caption = "LDTIM" Or Label_fun.Caption = "ANDTIM" Or Label_fun.Caption =
"ORTIM" Or Label_fun.Caption = "LDCNT" Or Label_fun.Caption = "ANDCNT" Or
Label_fun.Caption = "ORCNT" Or Label_fun.Caption = "LDNOTTIM" Or Label_fun.Caption =
"ANDNOTTIM" Or Label_fun.Caption = "ORNOTTIM" Or Label_fun.Caption = "LDNOTCNT" Or
Label_fun.Caption = "ANDNOTCNT" Or Label_fun.Caption = "ORNOTCNT") Then
```

```
   Label_data.Caption = "6"
Else
   Label_data.Caption = Shift_data(databoolean, "6")
End If
End Sub

Private Sub botton_7_Click() 'botton seven of boolean
databoolean = Label_data.Caption
If Lenght_data(databoolean) > 3 Then
   Label_data.Caption = "7"
ElseIf (Lenght_data(databoolean) = 3) And (Label_fun.Caption = "TIM" Or Label_fun.Caption =
"CNT" Or Label_fun.Caption = "LDTIM" Or Label_fun.Caption = "ANDTIM" Or Label_fun.Caption =
"ORTIM" Or Label_fun.Caption = "LDCNT" Or Label_fun.Caption = "ANDCNT" Or
Label_fun.Caption = "ORCNT" Or Label_fun.Caption = "LDNOTTIM" Or Label_fun.Caption =
"ANDNOTTIM" Or Label_fun.Caption = "ORNOTTIM" Or Label_fun.Caption = "LDNOTCNT" Or
Label_fun.Caption = "ANDNOTCNT" Or Label_fun.Caption = "ORNOTCNT") Then
   Label_data.Caption = "7"
Else
   Label_data.Caption = Shift_data(databoolean, "7")
End If
End Sub
-----------------------------------------------------------------------------------------------------------------------
Private Sub botton_8_Click() 'botton eight of boolean
databoolean = Label_data.Caption
If Lenght_data(databoolean) > 3 Then
   Label_data.Caption = "8"
ElseIf (Lenght_data(databoolean) = 3) And (Label_fun.Caption = "TIM" Or Label_fun.Caption =
"CNT" Or Label_fun.Caption = "LDTIM" Or Label_fun.Caption = "ANDTIM" Or Label_fun.Caption =
"ORTIM" Or Label_fun.Caption = "LDCNT" Or Label_fun.Caption = "ANDCNT" Or
Label_fun.Caption = "ORCNT" Or Label_fun.Caption = "LDNOTTIM" Or Label_fun.Caption =
"ANDNOTTIM" Or Label_fun.Caption = "ORNOTTIM" Or Label_fun.Caption = "LDNOTCNT" Or
Label_fun.Caption = "ANDNOTCNT" Or Label_fun.Caption = "ORNOTCNT") Then
```

```
   Label_data.Caption = "8"
Else
   Label_data.Caption = Shift_data(databoolean, "8")
End If
End Sub
```

```
Private Sub botton_9_Click() 'botton nine of boolean
databoolean = Label_data.Caption
If Lenght_data(databoolean) > 3 Then
   Label_data.Caption = "9"
ElseIf (Lenght_data(databoolean) = 3) And (Label_fun.Caption = "TIM" Or Label_fun.Caption =
"CNT" Or Label_fun.Caption = "LDTIM" Or Label_fun.Caption = "ANDTIM" Or Label_fun.Caption =
"ORTIM" Or Label_fun.Caption = "LDCNT" Or Label_fun.Caption = "ANDCNT" Or
Label_fun.Caption = "ORCNT" Or Label_fun.Caption = "LDNOTTIM" Or Label_fun.Caption =
"ANDNOTTIM" Or Label_fun.Caption = "ORNOTTIM" Or Label_fun.Caption = "LDNOTCNT" Or
Label_fun.Caption = "ANDNOTCNT" Or Label_fun.Caption = "ORNOTCNT") Then
   Label_data.Caption = "9"
Else
   Label_data.Caption = Shift_data(databoolean, "9")
End If
End Sub
```

```
Private Sub botton_a_Click() 'botton ten of boolean
databoolean = Label_data.Caption
If Lenght_data(databoolean) > 3 Then
   Label_data.Caption = "A"
ElseIf (Lenght_data(databoolean) = 3) And (Label_fun.Caption = "TIM" Or Label_fun.Caption =
"CNT" Or Label_fun.Caption = "LDTIM" Or Label_fun.Caption = "ANDTIM" Or Label_fun.Caption =
"ORTIM" Or Label_fun.Caption = "LDCNT" Or Label_fun.Caption = "ANDCNT" Or
Label_fun.Caption = "ORCNT" Or Label_fun.Caption = "LDNOTTIM" Or Label_fun.Caption =
"ANDNOTTIM" Or Label_fun.Caption = "ORNOTTIM" Or Label_fun.Caption = "LDNOTCNT" Or
Label_fun.Caption = "ANDNOTCNT" Or Label_fun.Caption = "ORNOTCNT") Then
```

```
  Label_data.Caption = "A"
Else
  Label_data.Caption = Shift_data(databoolean, "A")
End If
End Sub
```

```
Private Sub botton_b_Click() 'botton eleven of boolean
databoolean = Label_data.Caption
If Lenght_data(databoolean) > 3 Then
  Label_data.Caption = "B"
ElseIf (Lenght_data(databoolean) = 3) And (Label_fun.Caption = "TIM" Or Label_fun.Caption =
"CNT" Or Label_fun.Caption = "LDTIM" Or Label_fun.Caption = "ANDTIM" Or Label_fun.Caption =
"ORTIM" Or Label_fun.Caption = "LDCNT" Or Label_fun.Caption = "ANDCNT" Or
Label_fun.Caption = "ORCNT" Or Label_fun.Caption = "LDNOTTIM" Or Label_fun.Caption =
"ANDNOTTIM" Or Label_fun.Caption = "ORNOTTIM" Or Label_fun.Caption = "LDNOTCNT" Or
Label_fun.Caption = "ANDNOTCNT" Or Label_fun.Caption = "ORNOTCNT") Then
  Label_data.Caption = "B"
Else
  Label_data.Caption = Shift_data(databoolean, "B")
End If
End Sub
```

```
Private Sub botton_c_Click() 'botton twelve of boolean
databoolean = Label_data.Caption
If Lenght_data(databoolean) > 3 Then
  Label_data.Caption = "C"
ElseIf (Lenght_data(databoolean) = 3) And (Label_fun.Caption = "TIM" Or Label_fun.Caption =
"CNT" Or Label_fun.Caption = "LDTIM" Or Label_fun.Caption = "ANDTIM" Or Label_fun.Caption =
"ORTIM" Or Label_fun.Caption = "LDCNT" Or Label_fun.Caption = "ANDCNT" Or
Label_fun.Caption = "ORCNT" Or Label_fun.Caption = "LDNOTTIM" Or Label_fun.Caption =
"ANDNOTTIM" Or Label_fun.Caption = "ORNOTTIM" Or Label_fun.Caption = "LDNOTCNT" Or
Label_fun.Caption = "ANDNOTCNT" Or Label_fun.Caption = "ORNOTCNT") Then
```

```
  Label_data.Caption = "C"
Else
  Label_data.Caption = Shift_data(databoolean, "C")
End If
End Sub
```

```
Private Sub botton_d_Click() 'botton thirteen of boolean
databoolean = Label_data.Caption
If Lenght_data(databoolean) > 3 Then
  Label_data.Caption = "D"
ElseIf (Lenght_data(databoolean) = 3) And (Label_fun.Caption = "TIM" Or Label_fun.Caption =
"CNT" Or Label_fun.Caption = "LDTIM" Or Label_fun.Caption = "ANDTIM" Or Label_fun.Caption =
"ORTIM" Or Label_fun.Caption = "LDCNT" Or Label_fun.Caption = "ANDCNT" Or
Label_fun.Caption = "ORCNT" Or Label_fun.Caption = "LDNOTTIM" Or Label_fun.Caption =
"ANDNOTTIM" Or Label_fun.Caption = "ORNOTTIM" Or Label_fun.Caption = "LDNOTCNT" Or
Label_fun.Caption = "ANDNOTCNT" Or Label_fun.Caption = "ORNOTCNT") Then
  Label_data.Caption = "D"
Else
  Label_data.Caption = Shift_data(databoolean, "D")
End If
End Sub
```

```
Private Sub botton_e_Click() 'botton fourteen of boolean
databoolean = Label_data.Caption
If Lenght_data(databoolean) > 3 Then
  Label_data.Caption = "E"
ElseIf (Lenght_data(databoolean) = 3) And (Label_fun.Caption = "TIM" Or Label_fun.Caption =
"CNT" Or Label_fun.Caption = "LDTIM" Or Label_fun.Caption = "ANDTIM" Or Label_fun.Caption =
"ORTIM" Or Label_fun.Caption = "LDCNT" Or Label_fun.Caption = "ANDCNT" Or
Label_fun.Caption = "ORCNT" Or Label_fun.Caption = "LDNOTTIM" Or Label_fun.Caption =
"ANDNOTTIM" Or Label_fun.Caption = "ORNOTTIM" Or Label_fun.Caption = "LDNOTCNT" Or
Label_fun.Caption = "ANDNOTCNT" Or Label_fun.Caption = "ORNOTCNT") Then
```

```
    Label_data.Caption = "E"
Else
    Label_data.Caption = Shift_data(databoolean, "E")
End If
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub botton_f_Click() 'botton fifteen of boolean
databoolean = Label_data.Caption
If Lenght_data(databoolean) > 3 Then
    Label_data.Caption = "F"
ElseIf (Lenght_data(databoolean) = 3) And (Label_fun.Caption = "TIM" Or Label_fun.Caption =
"CNT" Or Label_fun.Caption = "LDTIM" Or Label_fun.Caption = "ANDTIM" Or Label_fun.Caption =
"ORTIM" Or Label_fun.Caption = "LDCNT" Or Label_fun.Caption = "ANDCNT" Or
Label_fun.Caption = "ORCNT" Or Label_fun.Caption = "LDNOTTIM" Or Label_fun.Caption =
"ANDNOTTIM" Or Label_fun.Caption = "ORNOTTIM" Or Label_fun.Caption = "LDNOTCNT" Or
Label_fun.Caption = "ANDNOTCNT" Or Label_fun.Caption = "ORNOTCNT") Then
    Label_data.Caption = "F"
Else
    Label_data.Caption = Shift_data(databoolean, "F")
End If
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub combo_fun_Click() 'combo select function
Label_fun.BackColor = &HC0FFC0
If combo_fun.ListIndex = 0 Then 'function 00
    Label_fun.Caption = "NOP(00)"
    Call non_word
    Call non_value
    Call non_value_option
    Call disable_value_all_option
    Call non_check_data
    Label_data.Caption = Empty
```

```
ElseIf combo_fun.ListIndex = 1 Then 'function 01
    Label_fun.Caption = "END(01)"
    Call non_word
    Call non_value
    Call non_value_option
    Call disable_value_all_option
    Call non_check_data
    Label_data.Caption = Empty
ElseIf combo_fun.ListIndex = 2 Then 'function 02
    Label_fun.Caption = "IL(02)"
    Call non_word
    Call non_value
    Call non_value_option
    Call disable_value_all_option
    Call non_check_data
    Label_data.Caption = Empty
ElseIf combo_fun.ListIndex = 3 Then 'function 03
    Label_fun.Caption = "ILC(03)"
    Call non_word
    Call non_value
    Call non_value_option
    Call disable_value_all_option
    Call non_check_data
    Label_data.Caption = Empty
ElseIf combo_fun.ListIndex = 4 Then 'function 04
    Label_fun.Caption = "JMP(04)"
    Call non_word
    Call non_value
    Call non_value_option
    Call disable_value_all_option
    Call non_check_data
    Label_data.Caption = Empty
```

```
ElseIf combo_fun.ListIndex = 5 Then 'function 05
    Label_fun.Caption = "JME(05)"
    Call non_word
    Call non_value
    Call non_value_option
    Call disable_value_all_option
    Call non_check_data
    Label_data.Caption = Empty
ElseIf combo_fun.ListIndex = 6 Then 'function 10
    Label_fun.Caption = "SFT(10)"
    Call word2
    Call value_zero2
    Call enable_value_all_option
    Option_ir.Value = True
    Call check_data2
    Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 7 Then 'function 11
    Label_fun.Caption = "KEEP(11)"
    Call word1
    Call value_zero1
    Call enable_value_all_option
    Option_ir.Value = True
    Call check_data1
    Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 8 Then 'function 13
    Label_fun.Caption = "DIFU(13)"
    Call word1
    Call value_zero1
    Call enable_value_all_option
    Option_ir.Value = True
    Call check_data1
    Label_data.Caption = "0000"
```

```
ElseIf combo_fun.ListIndex = 9 Then 'function 14
   Label_fun.Caption = "DIFD(14)"
   Call word1
   Call value_zero1
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data1
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 10 Then 'function 16
   Label_fun.Caption = "WSFT(16)"
   Call word2
   Call value_zero2
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data2
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 11 Then 'function 20
   Label_fun.Caption = "CMP(20)"
   Call word2
   Call value_zero2
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data2
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 12 Then 'function 21
   Label_fun.Caption = "MOV(21)"
   Call word2
   Call value_zero2
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data2
   Label_data.Caption = "0000"
```

```
ElseIf combo_fun.ListIndex = 13 Then 'function 22
   Label_fun.Caption = "MVN(22)"
   Call word2
   Call value_zero2
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data2
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 14 Then 'function 23
   Label_fun.Caption = "BIN(23)"
   Call word2
   Call value_zero2
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data2
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 15 Then 'function 24
   Label_fun.Caption = "BCD(24)"
   Call word2
   Call value_zero2
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data2
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 16 Then 'function 25
   Label_fun.Caption = "ASL(25)"
   Call word1
   Call value_zero1
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data1
   Label_data.Caption = "0000"
```

```
ElseIf combo_fun.ListIndex = 17 Then 'function 26
   Label_fun.Caption = "ASR(26)"
   Call word1
   Call value_zero1
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data1
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 18 Then 'function 27
   Label_fun.Caption = "ROL(27)"
   Call word1
   Call value_zero1
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data1
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 19 Then 'function 28
   Label_fun.Caption = "ROR(28)"
   Call word1
   Call value_zero1
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data1
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 20 Then 'function 29
   Label_fun.Caption = "COM(29)"
   Call word1
   Call value_zero1
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data1
   Label_data.Caption = "0000"
```

```
ElseIf combo_fun.ListIndex = 21 Then 'function 30
   Label_fun.Caption = "ADD(30)"
   Call word3
   Call value_zero3
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data3
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 22 Then 'function 31
   Label_fun.Caption = "SUB(31)"
   Call word3
   Call value_zero3
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data3
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 23 Then 'function 32
   Label_fun.Caption = "MUL(32)"
   Call word3
   Call value_zero3
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data3
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 24 Then 'function 33
   Label_fun.Caption = "DIV(33)"
   Call word3
   Call value_zero3
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data3
   Label_data.Caption = "0000"
```

```
ElseIf combo_fun.ListIndex = 25 Then 'function 34
   Label_fun.Caption = "ANDW(34)"
   Call word3
   Call value_zero3
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data3
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 26 Then 'function 35
   Label_fun.Caption = "ORW(35)"
   Call word3
   Call value_zero3
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data3
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 27 Then 'function 36
   Label_fun.Caption = "XORW(36)"
   Call word3
   Call value_zero3
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data3
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 28 Then 'function 37
   Label_fun.Caption = "XNRW(37)"
   Call word3
   Call value_zero3
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data3
   Label_data.Caption = "0000"
```

```
ElseIf combo_fun.ListIndex = 29 Then 'function 38
   Label_fun.Caption = "INC(38)"
   Call word1
   Call value_zero1
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data1
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 30 Then 'function 39
   Label_fun.Caption = "DEC(39)"
   Call word1
   Call value_zero1
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data1
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 31 Then 'function 40
   Label_fun.Caption = "STC(40)"
   Call non_word
   Call non_value
   Call non_value_option
   Call disable_value_all_option
   Call non_check_data
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 32 Then 'function 41
   Label_fun.Caption = "CLC(41)"
   Call non_word
   Call non_value
   Call non_value_option
   Call disable_value_all_option
   Call non_check_data
   Label_data.Caption = "0000"
```

```
ElseIf combo_fun.ListIndex = 33 Then 'function 47
   Label_fun.Caption = "MSG(47)"
   Call word1
   Call value_zero1
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data1
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 34 Then 'function 50
   Label_fun.Caption = "ADB(50)"
   Call word3
   Call value_zero3
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data3
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 35 Then 'function 51
   Label_fun.Caption = "SBB(51)"
   Call word3
   Call value_zero3
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data3
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 36 Then 'function 52
   Label_fun.Caption = "MLB(52)"
   Call word3
   Call value_zero3
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data3
   Label_data.Caption = "0000"
```

```
ElseIf combo_fun.ListIndex = 37 Then 'function 53
    Label_fun.Caption = "DVB(53)"
    Call word3
    Call value_zero3
    Call enable_value_all_option
    Option_ir.Value = True
    Call check_data3
    Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 38 Then 'function 70
    Label_fun.Caption = "RDI(70)"
    Call word2
    Call value_zero2
    Call enable_value_all_option
    Option_ir.Value = True
    Call check_data2
    Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 39 Then 'function 71
    Label_fun.Caption = "WRO(71)"
    Call word2
    Call value_zero2
    Call enable_value_all_option
    Option_ir.Value = True
    Call check_data2
    Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 40 Then 'function 74
    Label_fun.Caption = "SLD(74)"
    Call word2
    Call value_zero2
    Call enable_value_all_option
    Option_ir.Value = True
    Call check_data2
    Label_data.Caption = "0000"
```

```
ElseIf combo_fun.ListIndex = 41 Then 'function 75
    Label_fun.Caption = "SRD(75)"
    Call word2
    Call value_zero2
    Call enable_value_all_option
    Option_ir.Value = True
    Call check_data2
    Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 42 Then 'function 76
    Label_fun.Caption = "MLPX(76)"
    Call word3
    Call value_zero3
    Call enable_value_all_option
    Option_ir.Value = True
    Call check_data3
    Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 43 Then 'function 78
    Label_fun.Caption = "SDEC(78)"
    Call word3
    Call value_zero3
    Call enable_value_all_option
    Option_ir.Value = True
    Call check_data3
    Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 44 Then 'function 79
    Label_fun.Caption = "TKY(79)"
    Call word2
    Call value_zero2
    Call enable_value_all_option
    Option_ir.Value = True
    Call check_data2
    Label_data.Caption = "0000"
```

```
ElseIf combo_fun.ListIndex = 45 Then 'function 80
   Label_fun.Caption = "7SEG(80)"
   Call word2
   Call value_zero2
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data2
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 46 Then 'function 82
   Label_fun.Caption = "MOVB(82)"
   Call word3
   Call value_zero3
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data3
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 47 Then 'function 83
   Label_fun.Caption = "MOVD(83)"
   Call word3
   Call value_zero3
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data3
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 48 Then 'function 84
   Label_fun.Caption = "SFTR(84)"
   Call word2
   Call value_zero2
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data2
   Label_data.Caption = "0000"
```

```
ElseIf combo_fun.ListIndex = 49 Then 'function 86
   Label_fun.Caption = "ASC(86)"
   Call word3
   Call value_zero3
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data3
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 50 Then 'function 87
   Label_fun.Caption = "TXD(87)"
   Call word3
   Call value_zero3
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data3
   Label_data.Caption = "0000"
ElseIf combo_fun.ListIndex = 51 Then 'function 88
   Label_fun.Caption = "RXD(88)"
   Call word3
   Call value_zero3
   Call enable_value_all_option
   Option_ir.Value = True
   Call check_data3
   Label_data.Caption = "0000"
End If
End Sub
```

```
Private Sub botton_and_Click() 'and command line
Label_fun.BackColor = &HC0FFC0
Label_fun.Caption = "AND"
Call word1
Call value_zero1
```

```
Call enable_value_all_option
Option_ir.Value = True
Call check_data1
Label_data.Caption = "0000"
End Sub
```

```
Private Sub botton_or_Click() 'or command line
Label_fun.BackColor = &HC0FFC0
Label_fun.Caption = "OR"
Call word1
Call value_zero1
Call enable_value_all_option
Option_ir.Value = True
Call check_data1
Label_data.Caption = "0000"
End Sub
```

```
Private Sub botton_out_Click() 'out command line
Label_fun.BackColor = &HC0FFC0
Label_fun.Caption = "OUT"
Call word1
Call value_zero1
Call enable_value_all_option
Option_ir.Value = True
Call check_data1
Label_data.Caption = "0000"
End Sub
```

```
Private Sub Grid_Boolean_LostFocus() 'Check Grid_boolean when Lost Focus and Find it.
Dim row As Integer
Dim col As Integer
If checkfixcolor = True Then
```

```
    checkfixcolor = False
    row = Grid_Boolean.row
    col = Grid_Boolean.col
    For i = 1 To Grid_Boolean.Rows - 1
      mainprogram.Grid_Boolean.row = i
      mainprogram.Grid_Boolean.col = 0
      mainprogram.Grid_Boolean.CellBackColor = &HC0C0C0
    Next i
    Grid_Boolean.row = row
    Grid_Boolean.col = col
End If
End Sub
---------------------------------------------------------------------------------------------------------
Private Sub Grid_Boolean_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
'Check Grid_boolean when edit
If (Grid_Boolean.col = 2) And (Grid_Boolean.TextMatrix(Grid_Boolean.row, 2) <> Empty) Then
    Call non_word
    Call non_value
    Label_fun.Caption = Empty
    Label_fun.BackColor = &H8000000F
    Label_word(1).BackColor = &HFF0000
    If Len(Grid_Boolean.TextMatrix(Grid_Boolean.row, 2)) = 5 Then
      Call enable_value_all_option
      Label_memory(1).BackColor = &HFF0000
      If Mid(Grid_Boolean.TextMatrix(Grid_Boolean.row, 2), 1, 1) = "H" Then
        Option_hr.Value = True
      ElseIf Mid(Grid_Boolean.TextMatrix(Grid_Boolean.row, 2), 1, 1) = "I" Then
        Option_ir.Value = True
      ElseIf Mid(Grid_Boolean.TextMatrix(Grid_Boolean.row, 2), 1, 1) = "L" Then
        Option_lr.Value = True
      ElseIf Mid(Grid_Boolean.TextMatrix(Grid_Boolean.row, 2), 1, 1) = "D" Then
        Option_dm.Value = True
```

```
    ElseIf Mid(Grid_Boolean.TextMatrix(Grid_Boolean.row, 2), 1, 1) = "#" Then
      Option_constant.Value = True
    End If
      Label_memory(1).Caption = Mid(Grid_Boolean.TextMatrix(Grid_Boolean.row, 2), 1, 1)
      Label_word(1).Caption = Mid(Grid_Boolean.TextMatrix(Grid_Boolean.row, 2), 2, 4)
  ElseIf Len(Grid_Boolean.TextMatrix(Grid_Boolean.row, 2)) < 5 Then
    Call disable_value_all_option
    Call non_value_option
    If Mid(Grid_Boolean.TextMatrix(Grid_Boolean.row, 2), 1, 1) = "#" Then
      Label_memory(1).BackColor = &HFF0000
      Label_memory(1).Caption = "#"
      Label_word(1).Caption = Mid(Grid_Boolean.TextMatrix(Grid_Boolean.row, 2), 2, 3)
    ElseIf Mid(Grid_Boolean.TextMatrix(Grid_Boolean.row, 2), 1, 1) = " " Then
      Label_word(1).Caption = Mid(Grid_Boolean.TextMatrix(Grid_Boolean.row, 2), 2, 3)
    End If
  End If
End If
End Sub
```

```
Private Sub Grid_Ladder_Click()
rowscan = Grid_Ladder.row
colscan = Grid_Ladder.col
If Button_ladder_Vline.Value = True Then ' scan picture
  If ((scanmap(rowscan, colscan - 1) = 12) Or (scanmap(rowscan, colscan - 1) = 1) Or (scanmap
(rowscan, colscan - 1) = 2) Or (scanmap(rowscan, colscan - 1) = 5) Or (scanmap(rowscan, colscan - 1)
= 6) Or (scanmap(rowscan, colscan - 1) = 7) Or (scanmap(rowscan, colscan - 1) = 8)) And (scanmap
(rowscan, colscan + 1) = Empty) And (scanmap(rowscan - 1, colscan) = Empty) And ((scanmap
(rowscan + 1, colscan) = 15) Or (scanmap(rowscan + 1, colscan) = 1) Or (scanmap(rowscan + 1,
colscan) = 2) Or (scanmap(rowscan + 1, colscan) = 3) Or (scanmap(rowscan + 1, colscan) = 4) Or
(scanmap(rowscan + 1, colscan) = 5) Or (scanmap(rowscan + 1, colscan) = 6)) Then 'upper right conner
    Set Grid_Ladder.CellPicture = pictureclip_ladder1.GraphicCell(2)
    scanmap(Grid_Ladder.row, Grid_Ladder.col) = 9
```

```
ElseIf (scanmap(rowscan, colscan - 1) = Empty) And ((scanmap(rowscan, colscan + 1) = 12) Or (scanmap(rowscan, colscan + 1) = 2) Or (scanmap(rowscan, colscan + 1) = 3) Or (scanmap(rowscan, colscan + 1) = 4) Or (scanmap(rowscan, colscan + 1) = 5) Or (scanmap(rowscan, colscan + 1) = 8) Or (scanmap(rowscan, colscan + 1) = 9)) And (scanmap(rowscan - 1, colscan) = Empty) And ((scanmap (rowscan + 1, colscan) = 15) Or (scanmap(rowscan + 1, colscan) = 1) Or (scanmap(rowscan + 1, colscan) = 2) Or (scanmap(rowscan + 1, colscan) = 3) Or (scanmap(rowscan + 1, colscan) = 4) Or (scanmap(rowscan + 1, colscan) = 5) Or (scanmap(rowscan + 1, colscan) = 6)) Then 'upper left conner
    Set Grid_Ladder.CellPicture = pictureclip_ladder1.GraphicCell(0)
    scanmap(Grid_Ladder.row, Grid_Ladder.col) = 7
ElseIf ((scanmap(rowscan, colscan - 1) = 12) Or (scanmap(rowscan, colscan - 1) = 1) Or (scanmap (rowscan, colscan - 1) = 2) Or (scanmap(rowscan, colscan - 1) = 5) Or (scanmap(rowscan, colscan - 1) = 6) Or (scanmap(rowscan, colscan - 1) = 7) Or (scanmap(rowscan, colscan - 1) = 8)) And (scanmap (rowscan, colscan + 1) = Empty) And ((scanmap(rowscan - 1, colscan) = 15) Or (scanmap(rowscan - 1, colscan) = 4) Or (scanmap(rowscan - 1, colscan) = 5) Or (scanmap(rowscan - 1, colscan) = 6) Or (scanmap(rowscan - 1, colscan) = 7) Or (scanmap(rowscan - 1, colscan) = 8) Or (scanmap(rowscan - 1, colscan) = 9)) And (scanmap(rowscan + 1, colscan) = Empty) Then 'lower left conner
    Set Grid_Ladder.CellPicture = pictureclip_ladder1.GraphicCell(8)
    scanmap(Grid_Ladder.row, Grid_Ladder.col) = 3
ElseIf (scanmap(rowscan, colscan - 1) = Empty) And ((scanmap(rowscan, colscan + 1) = 12) Or (scanmap(rowscan, colscan + 1) = 2) Or (scanmap(rowscan, colscan + 1) = 3) Or (scanmap(rowscan, colscan + 1) = 4) Or (scanmap(rowscan, colscan + 1) = 5) Or (scanmap(rowscan, colscan + 1) = 8) Or (scanmap(rowscan, colscan + 1) = 9)) And ((scanmap(rowscan - 1, colscan) = 15) Or (scanmap(rowscan - 1, colscan) = 4) Or (scanmap(rowscan - 1, colscan) = 5) Or (scanmap(rowscan - 1, colscan) = 6) Or (scanmap(rowscan - 1, colscan) = 7) Or (scanmap(rowscan - 1, colscan) = 8) Or (scanmap(rowscan - 1, colscan) = 9)) And (scanmap(rowscan + 1, colscan) = Empty) Then 'lower right conner
    Set Grid_Ladder.CellPicture = pictureclip_ladder1.GraphicCell(6)
    scanmap(Grid_Ladder.row, Grid_Ladder.col) = 1
ElseIf ((scanmap(rowscan, colscan - 1) = 12) Or (scanmap(rowscan, colscan - 1) = 1) Or (scanmap (rowscan, colscan - 1) = 2) Or (scanmap(rowscan, colscan - 1) = 5) Or (scanmap(rowscan, colscan - 1) = 6) Or (scanmap(rowscan, colscan - 1) = 7) Or (scanmap(rowscan, colscan - 1) = 8)) And (scanmap (rowscan, colscan + 1) = Empty) And ((scanmap(rowscan - 1, colscan) = 15) Or (scanmap(rowscan - 1, colscan) = 5) Or (scanmap(rowscan - 1, colscan) = 6) Or (scanmap(rowscan - 1, colscan) = 7) Or
```

```
(scanmap(rowscan - 1, colscan) = 8) Or (scanmap(rowscan - 1, colscan) = 9)) And ((scanmap(rowscan
+ 1, colscan) = 15) Or (scanmap(rowscan + 1, colscan) = 1) Or (scanmap(rowscan + 1, colscan) = 2) Or
(scanmap(rowscan + 1, colscan) = 3) Or (scanmap(rowscan + 1, colscan) = 5) Or (scanmap(rowscan +
1, colscan) = 6)) Then 'mid right conner
    Set Grid_Ladder.CellPicture = pictureclip_ladder1.GraphicCell(5)
    scanmap(Grid_Ladder.row, Grid_Ladder.col) = 4
  ElseIf (scanmap(rowscan, colscan - 1) = Empty) And ((scanmap(rowscan, colscan + 1) = 12) Or
(scanmap(rowscan, colscan + 1) = 2) Or (scanmap(rowscan, colscan + 1) = 3) Or (scanmap(rowscan,
colscan + 1) = 4) Or (scanmap(rowscan, colscan + 1) = 5) Or (scanmap(rowscan, colscan + 1) = 8) Or
(scanmap(rowscan, colscan + 1) = 9)) And ((scanmap(rowscan - 1, colscan) = 15) Or (scanmap(rowscan
- 1, colscan) = 4) Or (scanmap(rowscan - 1, colscan) = 5) Or (scanmap(rowscan - 1, colscan) = 7) Or
(scanmap(rowscan - 1, colscan) = 8) Or (scanmap(rowscan - 1, colscan) = 9)) And ((scanmap(rowscan
+ 1, colscan) = 15) Or (scanmap(rowscan + 1, colscan) = 1) Or (scanmap(rowscan + 1, colscan) = 2) Or
(scanmap(rowscan + 1, colscan) = 3) Or (scanmap(rowscan + 1, colscan) = 4) Or (scanmap(rowscan +
1, colscan) = 5)) Then 'mid left conner
    Set Grid_Ladder.CellPicture = pictureclip_ladder1.GraphicCell(3)
    scanmap(Grid_Ladder.row, Grid_Ladder.col) = 6
  ElseIf ((scanmap(rowscan, colscan - 1) = 12) Or (scanmap(rowscan, colscan - 1) = 1) Or (scanmap
(rowscan, colscan - 1) = 2) Or (scanmap(rowscan, colscan - 1) = 5) Or (scanmap(rowscan, colscan - 1)
= 6) Or (scanmap(rowscan, colscan - 1) = 7) Or (scanmap(rowscan, colscan - 1) = 8)) And ((scanmap
(rowscan, colscan + 1) = 12) Or (scanmap(rowscan, colscan + 1) = 2) Or (scanmap(rowscan, colscan +
1) = 3) Or (scanmap(rowscan, colscan + 1) = 4) Or (scanmap(rowscan, colscan + 1) = 5) Or (scanmap
(rowscan, colscan + 1) = 8) Or (scanmap(rowscan, colscan + 1) = 9)) And (scanmap(rowscan - 1,
colscan) = Empty) And ((scanmap(rowscan + 1, colscan) = 15) Or (scanmap(rowscan + 1, colscan) = 1)
Or (scanmap(rowscan + 1, colscan) = 2) Or (scanmap(rowscan + 1, colscan) = 3) Or (scanmap(rowscan
+ 1, colscan) = 4) Or (scanmap(rowscan + 1, colscan) = 5) Or (scanmap(rowscan + 1, colscan) = 6))
Then 'mid upper conner
    Set Grid_Ladder.CellPicture = pictureclip_ladder2.GraphicCell(1)
    scanmap(Grid_Ladder.row, Grid_Ladder.col) = 8
  ElseIf ((scanmap(rowscan, colscan - 1) = 12) Or (scanmap(rowscan, colscan - 1) = 1) Or (scanmap
(rowscan, colscan - 1) = 2) Or (scanmap(rowscan, colscan - 1) = 5) Or (scanmap(rowscan, colscan - 1)
= 6) Or (scanmap(rowscan, colscan - 1) = 7) Or (scanmap(rowscan, colscan - 1) = 8)) And ((scanmap
```

```
(rowscan, colscan + 1) = 12) Or (scanmap(rowscan, colscan + 1) = 2) Or (scanmap(rowscan, colscan +
1) = 3) Or (scanmap(rowscan, colscan + 1) = 4) Or (scanmap(rowscan, colscan + 1) = 5) Or (scanmap
(rowscan, colscan + 1) = 8) Or (scanmap(rowscan, colscan + 1) = 9)) And ((scanmap(rowscan - 1,
colscan) = 15) Or (scanmap(rowscan - 1, colscan) = 4) Or (scanmap(rowscan - 1, colscan) = 5) Or
(scanmap(rowscan - 1, colscan) = 6) Or (scanmap(rowscan - 1, colscan) = 7) Or (scanmap(rowscan - 1,
colscan) = 8) Or (scanmap(rowscan - 1, colscan) = 9)) And (scanmap(rowscan + 1, colscan) = Empty)
Then 'mid lower conner
    Set Grid_Ladder.CellPicture = pictureclip_ladder2.GraphicCell(7)
    scanmap(Grid_Ladder.row, Grid_Ladder.col) = 2
  ElseIf ((scanmap(rowscan, colscan - 1) = 12) Or (scanmap(rowscan, colscan - 1) = 1) Or (scanmap
(rowscan, colscan - 1) = 2) Or (scanmap(rowscan, colscan - 1) = 6) Or (scanmap(rowscan, colscan - 1)
= 7) Or (scanmap(rowscan, colscan - 1) = 8)) And ((scanmap(rowscan, colscan + 1) = 12) Or (scanmap
(rowscan, colscan + 1) = 2) Or (scanmap(rowscan, colscan + 1) = 3) Or (scanmap(rowscan, colscan + 1)
= 4) Or (scanmap(rowscan, colscan + 1) = 8) Or (scanmap(rowscan, colscan + 1) = 9)) And ((scanmap
(rowscan - 1, colscan) = 15) Or (scanmap(rowscan - 1, colscan) = 4) Or (scanmap(rowscan - 1, colscan)
= 6) Or (scanmap(rowscan - 1, colscan) = 7) Or (scanmap(rowscan - 1, colscan) = 8) Or (scanmap
(rowscan - 1, colscan) = 9)) And ((scanmap(rowscan + 1, colscan) = 15) Or (scanmap(rowscan + 1,
colscan) = 1) Or (scanmap(rowscan + 1, colscan) = 2) Or (scanmap(rowscan + 1, colscan) = 3) Or
(scanmap(rowscan + 1, colscan) = 4) Or (scanmap(rowscan + 1, colscan) = 6)) Then 'The cross
    Set Grid_Ladder.CellPicture = pictureclip_ladder1.GraphicCell(4)
    scanmap(Grid_Ladder.row, Grid_Ladder.col) = 5
  Else 'insert Vertical line in grid ladder
    Set Grid_Ladder.CellPicture = pictureclip_ladder2.GraphicCell(4)
    scanmap(Grid_Ladder.row, Grid_Ladder.col) = 15
  End If
ElseIf Button_ladder_Hline.Value = True Then ' insert Horizontal line in grid ladder
  Set Grid_Ladder.CellPicture = pictureclip_ladder1.GraphicCell(7)
  scanmap(Grid_Ladder.row, Grid_Ladder.col) = 12
ElseIf Button_ladder_delete.Value = True Then ' Delete cell in grid ladder
  Set Grid_Ladder.CellPicture = pictureclip_ladder3.GraphicCell(0)
  scanmap(Grid_Ladder.row, Grid_Ladder.col) = Empty
  Grid_Ladder.TextMatrix(Grid_Ladder.row, Grid_Ladder.col) = Empty
```

```
End If
End Sub
---------------------------------------------------------------------------------------------------------
Private Sub Grid_Ladder_LostFocus() '
If function_ladder.Visible = False Then
   Button_ladder_point.Value = True
End If
End Sub
---------------------------------------------------------------------------------------------------------
Private Sub Grid_Ladder_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
'rowladder = Grid_Ladder.row
'colladder = Grid_Ladder.col
If (Button_ladder_LD.Value = True) Or (Button_ladder_Out.Value = True) Or
(Button_ladder_Tim.Value = True) Or (Button_ladder_Cnt.Value = True) Or (checkladder = True)
Then
   function_ladder.Visible = True
End If
End Sub
---------------------------------------------------------------------------------------------------------
Private Sub menu_bar_communication_model_oem_Click() 'set command line 2000 line of OEM
Model
numbercommandline = 2001
Grid_Boolean.Rows = numbercommandline
End Sub
---------------------------------------------------------------------------------------------------------
Private Sub menu_bar_communication_model_professional_Click() 'set command line 2000 line of
PROFESSIONAL Model
numbercommandline = 2001
Grid_Boolean.Rows = numbercommandline
End Sub
```

```
Private Sub menu_bar_communication_model_training_Click() 'set command line 2000 line of
TRAINING Model
numbercommandline = 501
Grid_Boolean.Rows = numbercommandline
End Sub
```

```
Private Sub menu_bar_communication_setting_default_Click() 'set  communication default
MSComm1.CommPort = 1
MSComm1.Settings = "9600,n,8,1"
End Sub
```

```
Private Sub menu_bar_communication_setting_manual_Click() 'set communication manual
communication_setting.Visible = True
End Sub
```

```
Private Sub menu_bar_edit_find_Click() 'Check Type Find
If tab_type.Tab = 0 Then
   find_ladder.Visible = True
ElseIf tab_type.Tab = 1 Then
   find_booleann.Visible = True
End If
End Sub
```

```
Private Sub menu_bar_mode_Offline_Click()
MSComm1.PortOpen = False
End Sub
```

```
Private Sub menu_bar_mode_Online_Click()
MSComm1.PortOpen = True
End Sub
```

```
Private Sub menu_bar_transfer_computer_plc_all_Click() 'Transfer Computer to PLC
checking_end
For i = 1 To 6000
   codecommand(i) = &H0
Next i
Hex_code_grid_boolean
MSComm1.Output = "@00ST*" + Fcs("@00ST*") + vbCr
Call receive_command
MSComm1.Output = "@00DL2000*" + Fcs("@00DL2000*") + vbCr
Call delay_set(1)
Do
'For i = 1 To 2000
   num = num + 1
   MSComm1.Output = codecommand(num)
   Call delay_base
'Next i
Loop Until num = 2000 'MSComm1.OutBufferCount = 0
Call delay_set(0.1)
MSComm1.Output = "@00RU*" + Fcs("@00RU*") + vbCr
End Sub
```

```
Private Sub Option_constant_Click()
If checkbooleansequence <> 0 Then
   Label_memory(checkbooleansequence).Caption = "#"
End If
End Sub
```

```
Private Sub Option_dm_Click()
If checkbooleansequence <> 0 Then
   Label_memory(checkbooleansequence).Caption = "D"
End If
End Sub
```

```
Private Sub Option_hr_Click()
If checkbooleansequence <> 0 Then
   Label_memory(checkbooleansequence).Caption = "H"
End If
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub Option_ir_Click()
If checkbooleansequence <> 0 Then
   Label_memory(checkbooleansequence).Caption = "I"
End If
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub Option_lr_Click()
If checkbooleansequence <> 0 Then
   Label_memory(checkbooleansequence).Caption = "L"
End If
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub tbToolBar_ButtonClick(ByVal Button As MSComctlLib.Button)
'gird_ladder(1, 1) = tbToolBar.Button
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub tbToolBar_ButtonMenuClick(ByVal ButtonMenu As MSComctlLib.ButtonMenu)
'gird_ladder(1, 1) = tbToolBar.Index
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub tbToolBar_Click()
'tbToolBar.Buttons.Item.
'tbToolBar.Align
'tbToolBar.Buttons
'tbToolBar.Controls
'tbToolBar.Customize
```

```
'tbToolBar.DataBindings
'tbToolBar.HotImageList
'tbToolBar.
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub Timer_1_Timer()
If Label_memory(1).BackColor = &HFF& Then
   Label_memory(1).BackColor = &HC0FFC0
End If
Label_word(1).BackColor = &HC0FFC0
Timer_1.Enabled = False
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub Timer_2_Timer()
Label_memory(2).BackColor = &HC0FFC0
Label_word(2).BackColor = &HC0FFC0
Timer_2.Enabled = False
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub Timer_3_Timer()
Label_memory(3).BackColor = &HC0FFC0
Label_word(3).BackColor = &HC0FFC0
Timer_3.Enabled = False
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub grid_memory_ir_EnterCell() 'show value ir memory in text box
If grid_memory_ir.row = 1 Then
   Label_ch_number_ir.Caption = grid_memory_ir.col - 1
ElseIf grid_memory_ir.row <> 0 Then
   Label_ch_number_ir.Caption = ((grid_memory_ir.row - 1) * 10) + grid_memory_ir.col - 1
End If
text_memory_ir.Text = grid_memory_ir.TextMatrix(grid_memory_ir.row, grid_memory_ir.col)
```

```
Call value_ir_check_digit(0, Mid(text_memory_ir.Text, 4, 1))
Call value_ir_check_digit(1, Mid(text_memory_ir.Text, 3, 1))
Call value_ir_check_digit(2, Mid(text_memory_ir.Text, 2, 1))
Call value_ir_check_digit(3, Mid(text_memory_ir.Text, 1, 1))
End Sub
```

```
Private Sub grid_memory_hr_EnterCell() 'show value hr memory in text box
If grid_memory_hr.row = 1 Then
   Label_ch_number_hr.Caption = grid_memory_hr.col - 1
ElseIf grid_memory_hr.row <> 0 Then
   Label_ch_number_hr.Caption = ((grid_memory_hr.row - 1) * 10) + grid_memory_hr.col - 1
End If
text_memory_hr.Text = grid_memory_hr.TextMatrix(grid_memory_hr.row, grid_memory_hr.col)
Call value_hr_check_digit(0, Mid(text_memory_hr.Text, 4, 1))
Call value_hr_check_digit(1, Mid(text_memory_hr.Text, 3, 1))
Call value_hr_check_digit(2, Mid(text_memory_hr.Text, 2, 1))
Call value_hr_check_digit(3, Mid(text_memory_hr.Text, 1, 1))
End Sub
```

```
Private Sub grid_memory_lr_EnterCell() 'show value lr memory in text box
If grid_memory_lr.row = 1 Then
   Label_ch_number_lr.Caption = grid_memory_lr.col - 1
ElseIf grid_memory_lr.row <> 0 Then
   Label_ch_number_lr.Caption = ((grid_memory_lr.row - 1) * 10) + grid_memory_lr.col - 1
End If
text_memory_lr.Text = grid_memory_lr.TextMatrix(grid_memory_lr.row, grid_memory_lr.col)
Call value_lr_check_digit(0, Mid(text_memory_lr.Text, 4, 1))
Call value_lr_check_digit(1, Mid(text_memory_lr.Text, 3, 1))
Call value_lr_check_digit(2, Mid(text_memory_lr.Text, 2, 1))
Call value_lr_check_digit(3, Mid(text_memory_lr.Text, 1, 1))
End Sub
```

```
Private Sub grid_memory_dm_EnterCell() 'show value dm memory in text box
If grid_memory_dm.row = 1 Then
   Label_ch_number_dm.Caption = grid_memory_dm.col - 1
ElseIf grid_memory_dm.row <> 0 Then
   Label_ch_number_dm.Caption = ((grid_memory_dm.row - 1) * 10) + grid_memory_dm.col - 1
End If
text_memory_dm.Text = grid_memory_dm.TextMatrix(grid_memory_dm.row, grid_memory_dm.col)
Call value_dm_check_digit(0, Mid(text_memory_dm.Text, 4, 1))
Call value_dm_check_digit(1, Mid(text_memory_dm.Text, 3, 1))
Call value_dm_check_digit(2, Mid(text_memory_dm.Text, 2, 1))
Call value_dm_check_digit(3, Mid(text_memory_dm.Text, 1, 1))
End Sub
```

```
Private Sub grid_memory_tc_EnterCell() 'show value tc memory in text box
If grid_memory_tc.row = 1 Then
   Label_ch_number_tc.Caption = grid_memory_tc.col - 1
ElseIf grid_memory_tc.row <> 0 Then
   Label_ch_number_tc.Caption = ((grid_memory_tc.row - 1) * 10) + grid_memory_tc.col - 1
End If
text_memory_tc.Text = grid_memory_tc.TextMatrix(grid_memory_tc.row, grid_memory_tc.col)
Call value_tc_check_digit(0, Mid(text_memory_tc.Text, 4, 1))
Call value_tc_check_digit(1, Mid(text_memory_tc.Text, 3, 1))
Call value_tc_check_digit(2, Mid(text_memory_tc.Text, 2, 1))
Call value_tc_check_digit(3, Mid(text_memory_tc.Text, 1, 1))
End Sub
```

```
Private Sub change_ir_Click() 'change value ir memory to grid memory
If Len(text_memory_ir.Text) = 1 Then
   grid_memory_ir.TextMatrix(grid_memory_ir.row, grid_memory_ir.col) = "000" +
text_memory_ir.Text
ElseIf Len(text_memory_ir.Text) = 2 Then
```

```
   grid_memory_ir.TextMatrix(grid_memory_ir.row, grid_memory_ir.col) = "00" +
text_memory_ir.Text
ElseIf Len(text_memory_ir.Text) = 3 Then
   grid_memory_ir.TextMatrix(grid_memory_ir.row, grid_memory_ir.col) = "0" +
text_memory_ir.Text
ElseIf Len(text_memory_ir.Text) <> 0 Then
   grid_memory_ir.TextMatrix(grid_memory_ir.row, grid_memory_ir.col) = text_memory_ir.Text
End If
grid_memory_ir.CellAlignment = 7
End Sub
```

```
Private Sub change_hr_Click() 'change value hr memory to grid memory
If Len(text_memory_hr.Text) = 1 Then
   grid_memory_hr.TextMatrix(grid_memory_hr.row, grid_memory_hr.col) = "000" +
text_memory_hr.Text
ElseIf Len(text_memory_hr.Text) = 2 Then
   grid_memory_hr.TextMatrix(grid_memory_hr.row, grid_memory_hr.col) = "00" +
text_memory_hr.Text
ElseIf Len(text_memory_hr.Text) = 3 Then
   grid_memory_hr.TextMatrix(grid_memory_hr.row, grid_memory_hr.col) = "0" +
text_memory_hr.Text
ElseIf Len(text_memory_hr.Text) <> 0 Then
   grid_memory_hr.TextMatrix(grid_memory_hr.row, grid_memory_hr.col) = text_memory_hr.Text
End If
grid_memory_hr.CellAlignment = 7
End Sub
```

```
Private Sub change_lr_Click() 'change value lr memory to grid memory
If Len(text_memory_lr.Text) = 1 Then
   grid_memory_lr.TextMatrix(grid_memory_lr.row, grid_memory_lr.col) = "000" +
text_memory_lr.Text
ElseIf Len(text_memory_lr.Text) = 2 Then
```

```
    grid_memory_lr.TextMatrix(grid_memory_lr.row, grid_memory_lr.col) = "00" +
 text_memory_lr.Text
 ElseIf Len(text_memory_lr.Text) = 3 Then
    grid_memory_lr.TextMatrix(grid_memory_lr.row, grid_memory_lr.col) = "0" +
 text_memory_lr.Text
 ElseIf Len(text_memory_lr.Text) <> 0 Then
    grid_memory_lr.TextMatrix(grid_memory_lr.row, grid_memory_lr.col) = text_memory_lr.Text
 End If
 grid_memory_lr.CellAlignment = 7
 End Sub
 ---------------------------------------------------------------------------------------------------------------------

 Private Sub change_dm_Click() 'change value dm memory to grid memory
 If Len(text_memory_dm.Text) = 1 Then
    grid_memory_dm.TextMatrix(grid_memory_dm.row, grid_memory_dm.col) = "000" +
 text_memory_dm.Text
 ElseIf Len(text_memory_dm.Text) = 2 Then
    grid_memory_dm.TextMatrix(grid_memory_dm.row, grid_memory_dm.col) = "00" +
 text_memory_dm.Text
 ElseIf Len(text_memory_dm.Text) = 3 Then
    grid_memory_dm.TextMatrix(grid_memory_dm.row, grid_memory_dm.col) = "0" +
 text_memory_dm.Text
 ElseIf Len(text_memory_dm.Text) <> 0 Then
    grid_memory_dm.TextMatrix(grid_memory_dm.row, grid_memory_dm.col) =
 text_memory_dm.Text
 End If
 grid_memory_dm.CellAlignment = 7
 End Sub
 ---------------------------------------------------------------------------------------------------------------------

 Private Sub change_tc_Click() 'change value tc memory to grid memory
 If Len(text_memory_tc.Text) = 1 Then
    grid_memory_tc.TextMatrix(grid_memory_tc.row, grid_memory_tc.col) = "000" +
 text_memory_tc.Text
```

```
ElseIf Len(text_memory_tc.Text) = 2 Then
   grid_memory_tc.TextMatrix(grid_memory_tc.row, grid_memory_tc.col) = "00" +
text_memory_tc.Text
ElseIf Len(text_memory_tc.Text) = 3 Then
   grid_memory_tc.TextMatrix(grid_memory_tc.row, grid_memory_tc.col) = "0" +
text_memory_tc.Text
ElseIf Len(text_memory_tc.Text) <> 0 Then
   grid_memory_tc.TextMatrix(grid_memory_tc.row, grid_memory_tc.col) = text_memory_tc.Text
End If
grid_memory_tc.CellAlignment = 7
End Sub
```

```
Private Sub text_memory_ir_Change()
text_memory_ir.Text = UCase(text_memory_ir.Text)
text_memory_ir.SelStart = Len(text_memory_ir.Text)
If text_memory_ir.Text <> Empty Then
   If (Mid(text_memory_ir.Text, Len(text_memory_ir.Text), 1) <> 1) And (Mid(text_memory_ir.Text,
Len(text_memory_ir.Text), 1) <> 2) And (Mid(text_memory_ir.Text, Len(text_memory_ir.Text), 1) <>
3) And (Mid(text_memory_ir.Text, Len(text_memory_ir.Text), 1) <> 4) And
(Mid(text_memory_ir.Text, Len(text_memory_ir.Text), 1) <> 5) And (Mid(text_memory_ir.Text, Len
(text_memory_ir.Text), 1) <> 6) And (Mid(text_memory_ir.Text, Len(text_memory_ir.Text), 1) <> 7)
And (Mid(text_memory_ir.Text, Len(text_memory_ir.Text), 1) <> 8) And (Mid(text_memory_ir.Text,
Len(text_memory_ir.Text), 1) <> 9) And (Mid(text_memory_ir.Text, Len(text_memory_ir.Text), 1) <>
"A") And (Mid(text_memory_ir.Text, Len(text_memory_ir.Text), 1) <> "B") And (Mid
(text_memory_ir.Text, Len(text_memory_ir.Text), 1) <> "C") And (Mid(text_memory_ir.Text, Len
(text_memory_ir.Text), 1) <> "D") And (Mid(text_memory_ir.Text, Len(text_memory_ir.Text), 1) <>
"E") And (Mid(text_memory_ir.Text, Len(text_memory_ir.Text), 1) <> "F") Then
      If (Len(text_memory_ir.Text) = 1) And (text_memory_ir.Text <> "0") Then
            text_memory_ir.Text = Empty
      ElseIf (Len(text_memory_ir.Text) > 1) And (Mid(text_memory_ir.Text,
Len(text_memory_ir.Text), 1) <> "0") Then
            text_memory_ir.Text = Mid(text_memory_ir.Text, 1, Len(text_memory_ir.Text) - 1)
```

```
        End If
      End If
    End If
    If Len(text_memory_ir.Text) = 0 Then
      Call nonvalue_ir_check_digit(0)
    ElseIf Len(text_memory_ir.Text) = 1 Then
      Call value_ir_check_digit(0, text_memory_ir.Text)
      Call nonvalue_ir_check_digit(1)
      Call nonvalue_ir_check_digit(2)
      Call nonvalue_ir_check_digit(3)
    ElseIf Len(text_memory_ir.Text) = 2 Then
      Call value_ir_check_digit(0, Mid(text_memory_ir.Text, 2, 1))
      Call value_ir_check_digit(1, Mid(text_memory_ir.Text, 1, 1))
      Call nonvalue_ir_check_digit(2)
      Call nonvalue_ir_check_digit(3)
    ElseIf Len(text_memory_ir.Text) = 3 Then
      Call value_ir_check_digit(0, Mid(text_memory_ir.Text, 3, 1))
      Call value_ir_check_digit(1, Mid(text_memory_ir.Text, 2, 1))
      Call value_ir_check_digit(2, Mid(text_memory_ir.Text, 1, 1))
      Call nonvalue_ir_check_digit(3)
    ElseIf Len(text_memory_ir.Text) = 4 Then
      Call value_ir_check_digit(0, Mid(text_memory_ir.Text, 4, 1))
      Call value_ir_check_digit(1, Mid(text_memory_ir.Text, 3, 1))
      Call value_ir_check_digit(2, Mid(text_memory_ir.Text, 2, 1))
      Call value_ir_check_digit(3, Mid(text_memory_ir.Text, 1, 1))
    End If
    End Sub
    ---------------------------------------------------------------------------------------------------------------------
    Private Sub text_memory_hr_Change()
    text_memory_hr.Text = UCase(text_memory_hr.Text)
    text_memory_hr.SelStart = Len(text_memory_hr.Text)
    If text_memory_hr.Text <> Empty Then
```

```
If (Mid(text_memory_hr.Text, Len(text_memory_hr.Text), 1) <> 1) And (Mid(text_memory_hr.Text,
Len(text_memory_hr.Text), 1) <> 2) And (Mid(text_memory_hr.Text, Len(text_memory_hr.Text), 1)
<> 3) And (Mid(text_memory_hr.Text, Len(text_memory_hr.Text), 1) <> 4) And (Mid
(text_memory_hr.Text, Len(text_memory_hr.Text), 1) <> 5) And (Mid(text_memory_hr.Text, Len
(text_memory_hr.Text), 1) <> 6) And (Mid(text_memory_hr.Text, Len(text_memory_hr.Text), 1) <> 7)
And (Mid(text_memory_hr.Text, Len(text_memory_hr.Text), 1) <> 8) And
(Mid(text_memory_hr.Text, Len(text_memory_hr.Text), 1) <> 9) And (Mid(text_memory_hr.Text, Len
(text_memory_hr.Text), 1) <> "A") And (Mid(text_memory_hr.Text, Len(text_memory_hr.Text), 1) <>
"B") And (Mid(text_memory_hr.Text, Len(text_memory_hr.Text), 1) <> "C") And (Mid
(text_memory_hr.Text, Len(text_memory_hr.Text), 1) <> "D") And (Mid(text_memory_hr.Text, Len
(text_memory_hr.Text), 1) <> "E") And (Mid(text_memory_hr.Text, Len(text_memory_hr.Text), 1) <>
"F") Then
    If (Len(text_memory_hr.Text) = 1) And (text_memory_hr.Text <> "0") Then
        text_memory_hr.Text = Empty
    ElseIf (Len(text_memory_hr.Text) > 1) And (Mid(text_memory_hr.Text,
Len(text_memory_hr.Text), 1) <> "0") Then
        text_memory_hr.Text = Mid(text_memory_hr.Text, 1, Len(text_memory_hr.Text) - 1)
    End If
  End If
End If
If Len(text_memory_hr.Text) = 0 Then
  Call nonvalue_hr_check_digit(0)
ElseIf Len(text_memory_hr.Text) = 1 Then
  Call value_hr_check_digit(0, text_memory_hr.Text)
  Call nonvalue_hr_check_digit(1)
  Call nonvalue_hr_check_digit(2)
  Call nonvalue_hr_check_digit(3)
ElseIf Len(text_memory_hr.Text) = 2 Then
  Call value_hr_check_digit(0, Mid(text_memory_hr.Text, 2, 1))
  Call value_hr_check_digit(1, Mid(text_memory_hr.Text, 1, 1))
  Call nonvalue_hr_check_digit(2)
  Call nonvalue_hr_check_digit(3)
```

```
ElseIf Len(text_memory_hr.Text) = 3 Then
   Call value_hr_check_digit(0, Mid(text_memory_hr.Text, 3, 1))
   Call value_hr_check_digit(1, Mid(text_memory_hr.Text, 2, 1))
   Call value_hr_check_digit(2, Mid(text_memory_hr.Text, 1, 1))
   Call nonvalue_hr_check_digit(3)
ElseIf Len(text_memory_hr.Text) = 4 Then
   Call value_hr_check_digit(0, Mid(text_memory_hr.Text, 4, 1))
   Call value_hr_check_digit(1, Mid(text_memory_hr.Text, 3, 1))
   Call value_hr_check_digit(2, Mid(text_memory_hr.Text, 2, 1))
   Call value_hr_check_digit(3, Mid(text_memory_hr.Text, 1, 1))
End If
End Sub
-----------------------------------------------------------------------------------------------------------

Private Sub text_memory_lr_Change()
text_memory_lr.Text = UCase(text_memory_lr.Text)
text_memory_lr.SelStart = Len(text_memory_lr.Text)
If text_memory_lr.Text <> Empty Then
   If (Mid(text_memory_lr.Text, Len(text_memory_lr.Text), 1) <> 1) And (Mid(text_memory_lr.Text,
Len(text_memory_lr.Text), 1) <> 2) And (Mid(text_memory_lr.Text, Len(text_memory_lr.Text), 1) <>
3) And (Mid(text_memory_lr.Text, Len(text_memory_lr.Text), 1) <> 4) And
(Mid(text_memory_lr.Text, Len(text_memory_lr.Text), 1) <> 5) And (Mid(text_memory_lr.Text, Len
(text_memory_lr.Text), 1) <> 6) And (Mid(text_memory_lr.Text, Len(text_memory_lr.Text), 1) <> 7)
And (Mid(text_memory_lr.Text, Len(text_memory_lr.Text), 1) <> 8) And (Mid(text_memory_lr.Text,
Len(text_memory_lr.Text), 1) <> 9) And (Mid(text_memory_lr.Text, Len(text_memory_lr.Text), 1) <>
"A") And (Mid(text_memory_lr.Text, Len(text_memory_lr.Text), 1) <> "B") And (Mid
(text_memory_lr.Text, Len(text_memory_lr.Text), 1) <> "C") And (Mid(text_memory_lr.Text, Len
(text_memory_lr.Text), 1) <> "D") And (Mid(text_memory_lr.Text, Len(text_memory_lr.Text), 1) <>
"E") And (Mid(text_memory_lr.Text, Len(text_memory_lr.Text), 1) <> "F") Then
      If (Len(text_memory_lr.Text) = 1) And (text_memory_lr.Text <> "0") Then
            text_memory_lr.Text = Empty
      ElseIf (Len(text_memory_lr.Text) > 1) And (Mid(text_memory_lr.Text,
Len(text_memory_lr.Text), 1) <> "0") Then
```

```
            text_memory_lr.Text = Mid(text_memory_lr.Text, 1, Len(text_memory_lr.Text) - 1)
        End If
    End If
End If
If Len(text_memory_lr.Text) = 0 Then
    Call nonvalue_lr_check_digit(0)
ElseIf Len(text_memory_lr.Text) = 1 Then
    Call value_lr_check_digit(0, text_memory_lr.Text)
    Call nonvalue_lr_check_digit(1)
    Call nonvalue_lr_check_digit(2)
    Call nonvalue_lr_check_digit(3)
ElseIf Len(text_memory_lr.Text) = 2 Then
    Call value_lr_check_digit(0, Mid(text_memory_lr.Text, 2, 1))
    Call value_lr_check_digit(1, Mid(text_memory_lr.Text, 1, 1))
    Call nonvalue_lr_check_digit(2)
    Call nonvalue_lr_check_digit(3)
ElseIf Len(text_memory_lr.Text) = 3 Then
    Call value_lr_check_digit(0, Mid(text_memory_lr.Text, 3, 1))
    Call value_lr_check_digit(1, Mid(text_memory_lr.Text, 2, 1))
    Call value_lr_check_digit(2, Mid(text_memory_lr.Text, 1, 1))
    Call nonvalue_lr_check_digit(3)
ElseIf Len(text_memory_lr.Text) = 4 Then
    Call value_lr_check_digit(0, Mid(text_memory_lr.Text, 4, 1))
    Call value_lr_check_digit(1, Mid(text_memory_lr.Text, 3, 1))
    Call value_lr_check_digit(2, Mid(text_memory_lr.Text, 2, 1))
    Call value_lr_check_digit(3, Mid(text_memory_lr.Text, 1, 1))
End If
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub text_memory_dm_Change()
text_memory_dm.Text = UCase(text_memory_dm.Text)
text_memory_dm.SelStart = Len(text_memory_dm.Text)
```

```
If text_memory_dm.Text <> Empty Then
   If (Mid(text_memory_dm.Text, Len(text_memory_dm.Text), 1) <> 1) And
(Mid(text_memory_dm.Text, Len(text_memory_dm.Text), 1) <> 2) And (Mid(text_memory_dm.Text,
Len(text_memory_dm.Text), 1) <> 3) And (Mid(text_memory_dm.Text, Len(text_memory_dm.Text),
1) <> 4) And (Mid(text_memory_dm.Text, Len(text_memory_dm.Text), 1) <> 5) And (Mid
(text_memory_dm.Text, Len(text_memory_dm.Text), 1) <> 6) And (Mid(text_memory_dm.Text, Len
(text_memory_dm.Text), 1) <> 7) And (Mid(text_memory_dm.Text, Len(text_memory_dm.Text), 1)
<> 8) And (Mid(text_memory_dm.Text, Len(text_memory_dm.Text), 1) <> 9) And (Mid
(text_memory_dm.Text, Len(text_memory_dm.Text), 1) <> "A") And (Mid(text_memory_dm.Text,
Len(text_memory_dm.Text), 1) <> "B") And (Mid(text_memory_dm.Text,
Len(text_memory_dm.Text), 1) <> "C") And (Mid(text_memory_dm.Text,
Len(text_memory_dm.Text), 1) <> "D") And (Mid(text_memory_dm.Text,
Len(text_memory_dm.Text), 1) <> "E") And (Mid(text_memory_dm.Text,
Len(text_memory_dm.Text), 1) <> "F") Then
      If (Len(text_memory_dm.Text) = 1) And (text_memory_dm.Text <> "0") Then
         text_memory_dm.Text = Empty
      ElseIf (Len(text_memory_dm.Text) > 1) And (Mid(text_memory_dm.Text,
Len(text_memory_dm.Text), 1) <> "0") Then
         text_memory_dm.Text = Mid(text_memory_dm.Text, 1, Len(text_memory_dm.Text) - 1)
      End If
   End If
End If
If Len(text_memory_dm.Text) = 0 Then
   Call nonvalue_dm_check_digit(0)
ElseIf Len(text_memory_dm.Text) = 1 Then
   Call value_dm_check_digit(0, text_memory_dm.Text)
   Call nonvalue_dm_check_digit(1)
   Call nonvalue_dm_check_digit(2)
   Call nonvalue_dm_check_digit(3)
ElseIf Len(text_memory_dm.Text) = 2 Then
   Call value_dm_check_digit(0, Mid(text_memory_dm.Text, 2, 1))
   Call value_dm_check_digit(1, Mid(text_memory_dm.Text, 1, 1))
```

```
    Call nonvalue_dm_check_digit(2)
    Call nonvalue_dm_check_digit(3)
ElseIf Len(text_memory_dm.Text) = 3 Then
    Call value_dm_check_digit(0, Mid(text_memory_dm.Text, 3, 1))
    Call value_dm_check_digit(1, Mid(text_memory_dm.Text, 2, 1))
    Call value_dm_check_digit(2, Mid(text_memory_dm.Text, 1, 1))
    Call nonvalue_dm_check_digit(3)
ElseIf Len(text_memory_dm.Text) = 4 Then
    Call value_dm_check_digit(0, Mid(text_memory_dm.Text, 4, 1))
    Call value_dm_check_digit(1, Mid(text_memory_dm.Text, 3, 1))
    Call value_dm_check_digit(2, Mid(text_memory_dm.Text, 2, 1))
    Call value_dm_check_digit(3, Mid(text_memory_dm.Text, 1, 1))
End If
End Sub
-----------------------------------------------------------------------------------------------------------------------
Private Sub text_memory_tc_Change()
text_memory_tc.Text = UCase(text_memory_tc.Text)
text_memory_tc.SelStart = Len(text_memory_tc.Text)
If text_memory_tc.Text <> Empty Then
    If (Mid(text_memory_tc.Text, Len(text_memory_tc.Text), 1) <> 1) And (Mid(text_memory_tc.Text,
Len(text_memory_tc.Text), 1) <> 2) And (Mid(text_memory_tc.Text, Len(text_memory_tc.Text), 1)
<> 3) And (Mid(text_memory_tc.Text, Len(text_memory_tc.Text), 1) <> 4) And (Mid
(text_memory_tc.Text, Len(text_memory_tc.Text), 1) <> 5) And (Mid(text_memory_tc.Text, Len
(text_memory_tc.Text), 1) <> 6) And (Mid(text_memory_tc.Text, Len(text_memory_tc.Text), 1) <> 7)
And (Mid(text_memory_tc.Text, Len(text_memory_tc.Text), 1) <> 8) And (Mid(text_memory_tc.Text,
Len(text_memory_tc.Text), 1) <> 9) And (Mid(text_memory_tc.Text, Len(text_memory_tc.Text), 1)
<> "A") And (Mid(text_memory_tc.Text, Len(text_memory_tc.Text), 1) <> "B") And (Mid
(text_memory_tc.Text, Len(text_memory_tc.Text), 1) <> "C") And (Mid(text_memory_tc.Text, Len
(text_memory_tc.Text), 1) <> "D") And (Mid(text_memory_tc.Text, Len(text_memory_tc.Text), 1) <>
"E") And (Mid(text_memory_tc.Text, Len(text_memory_tc.Text), 1) <> "F") Then
        If (Len(text_memory_tc.Text) = 1) And (text_memory_tc.Text <> "0") Then
            text_memory_tc.Text = Empty
```

```
        ElseIf (Len(text_memory_tc.Text) > 1) And (Mid(text_memory_tc.Text,
Len(text_memory_tc.Text), 1) <> "0") Then
            text_memory_tc.Text = Mid(text_memory_tc.Text, 1, Len(text_memory_tc.Text) - 1)
        End If
    End If
End If
If Len(text_memory_tc.Text) = 0 Then
    Call nonvalue_tc_check_digit(0)
ElseIf Len(text_memory_tc.Text) = 1 Then
    Call value_tc_check_digit(0, text_memory_tc.Text)
    Call nonvalue_tc_check_digit(1)
    Call nonvalue_tc_check_digit(2)
    Call nonvalue_tc_check_digit(3)
ElseIf Len(text_memory_tc.Text) = 2 Then
    Call value_tc_check_digit(0, Mid(text_memory_tc.Text, 2, 1))
    Call value_tc_check_digit(1, Mid(text_memory_tc.Text, 1, 1))
    Call nonvalue_tc_check_digit(2)
    Call nonvalue_tc_check_digit(3)
ElseIf Len(text_memory_tc.Text) = 3 Then
    Call value_tc_check_digit(0, Mid(text_memory_tc.Text, 3, 1))
    Call value_tc_check_digit(1, Mid(text_memory_tc.Text, 2, 1))
    Call value_tc_check_digit(2, Mid(text_memory_tc.Text, 1, 1))
    Call nonvalue_tc_check_digit(3)
ElseIf Len(text_memory_tc.Text) = 4 Then
    Call value_tc_check_digit(0, Mid(text_memory_tc.Text, 4, 1))
    Call value_tc_check_digit(1, Mid(text_memory_tc.Text, 3, 1))
    Call value_tc_check_digit(2, Mid(text_memory_tc.Text, 2, 1))
    Call value_tc_check_digit(3, Mid(text_memory_tc.Text, 1, 1))
End If
End Sub
```

```
Private Sub text_memory_ir_Click()
text_memory_ir.SelStart = Len(text_memory_ir.Text)
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub text_memory_hr_Click()
text_memory_hr.SelStart = Len(text_memory_hr.Text)
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub text_memory_lr_Click()
text_memory_lr.SelStart = Len(text_memory_lr.Text)
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub text_memory_dm_Click()
text_memory_dm.SelStart = Len(text_memory_dm.Text)
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub text_memory_tc_Click()
text_memory_tc.SelStart = Len(text_memory_tc.Text)
End Sub
---------------------------------------------------------------------------------------------------------------
Private Sub Check_ir_bit_Click(Index As Integer)
If (Check_ir_bit(Index).Index = 0) Or (Check_ir_bit(Index).Index = 1) Or (Check_ir_bit(Index).Index =
2) Or (Check_ir_bit(Index).Index = 3) Then
   If Len(text_memory_ir.Text) <= 1 Then
      text_memory_ir.Text = check_bit_ir_convert_number(0)
   ElseIf Len(text_memory_ir.Text) > 1 Then
      text_memory_ir.Text = Mid(text_memory_ir.Text, 1, Len(text_memory_ir.Text) - 1) +
check_bit_ir_convert_number(0)
   End If
ElseIf (Check_ir_bit(Index).Index = 4) Or (Check_ir_bit(Index).Index = 5) Or (Check_ir_bit
(Index).Index = 6) Or (Check_ir_bit(Index).Index = 7) Then
   If Len(text_memory_ir.Text) = 0 Then
```

```
        text_memory_ir.Text = check_bit_ir_convert_number(1) + "0"
    ElseIf Len(text_memory_ir.Text) = 1 Then
        text_memory_ir.Text = check_bit_ir_convert_number(1) + text_memory_ir.Text
    ElseIf Len(text_memory_ir.Text) = 2 Then
        text_memory_ir.Text = check_bit_ir_convert_number(1) + Mid(text_memory_ir.Text, 2, 1)
    ElseIf Len(text_memory_ir.Text) > 2 Then
        text_memory_ir.Text = Mid(text_memory_ir.Text, 1, Len(text_memory_ir.Text) - 2) +
check_bit_ir_convert_number(1) + Mid(text_memory_ir.Text, Len(text_memory_ir.Text), 1)
    End If
ElseIf (Check_ir_bit(Index).Index = 8) Or (Check_ir_bit(Index).Index = 9) Or (Check_ir_bit
(Index).Index = 10) Or (Check_ir_bit(Index).Index = 11) Then
    If Len(text_memory_ir.Text) = 0 Then
        text_memory_ir.Text = check_bit_ir_convert_number(2) + "00"
    ElseIf Len(text_memory_ir.Text) = 1 Then
        text_memory_ir.Text = check_bit_ir_convert_number(2) + "0" + text_memory_ir.Text
    ElseIf Len(text_memory_ir.Text) = 2 Then
        text_memory_ir.Text = check_bit_ir_convert_number(2) + text_memory_ir.Text
    ElseIf Len(text_memory_ir.Text) = 3 Then
        text_memory_ir.Text = check_bit_ir_convert_number(2) + Mid(text_memory_ir.Text, 2, 2)
    ElseIf Len(text_memory_ir.Text) = 4 Then
        text_memory_ir.Text = Mid(text_memory_ir.Text, 1, 1) + check_bit_ir_convert_number(2) + Mid
(text_memory_ir.Text, 3, 2)
    End If
ElseIf (Check_ir_bit(Index).Index = 12) Or (Check_ir_bit(Index).Index = 13) Or (Check_ir_bit
(Index).Index = 14) Or (Check_ir_bit(Index).Index = 15) Then
    If Len(text_memory_ir.Text) = 0 Then
        text_memory_ir.Text = check_bit_ir_convert_number(3) + "000"
    ElseIf Len(text_memory_ir.Text) = 1 Then
        text_memory_ir.Text = check_bit_ir_convert_number(3) + "00" + text_memory_ir.Text
    ElseIf Len(text_memory_ir.Text) = 2 Then
        text_memory_ir.Text = check_bit_ir_convert_number(3) + "0" + text_memory_ir.Text
    ElseIf Len(text_memory_ir.Text) = 3 Then
```

```
    text_memory_ir.Text = check_bit_ir_convert_number(3) + text_memory_ir.Text
  ElseIf Len(text_memory_ir.Text) = 4 Then
    text_memory_ir.Text = check_bit_ir_convert_number(3) + Mid(text_memory_ir.Text, 2, 3)
  End If
End If
End Sub
```

```
Private Sub Check_hr_bit_Click(Index As Integer)
If (Check_hr_bit(Index).Index = 0) Or (Check_hr_bit(Index).Index = 1) Or (Check_hr_bit(Index).Index
= 2) Or (Check_hr_bit(Index).Index = 3) Then
  If Len(text_memory_hr.Text) <= 1 Then
    text_memory_hr.Text = check_bit_hr_convert_number(0)
  ElseIf Len(text_memory_hr.Text) > 1 Then
    text_memory_hr.Text = Mid(text_memory_hr.Text, 1, Len(text_memory_hr.Text) - 1) +
check_bit_hr_convert_number(0)
  End If
ElseIf (Check_hr_bit(Index).Index = 4) Or (Check_hr_bit(Index).Index = 5) Or (Check_hr_bit
(Index).Index = 6) Or (Check_hr_bit(Index).Index = 7) Then
  If Len(text_memory_hr.Text) = 0 Then
    text_memory_hr.Text = check_bit_hr_convert_number(1) + "0"
  ElseIf Len(text_memory_hr.Text) = 1 Then
    text_memory_hr.Text = check_bit_hr_convert_number(1) + text_memory_hr.Text
  ElseIf Len(text_memory_hr.Text) = 2 Then
    text_memory_hr.Text = check_bit_hr_convert_number(1) + Mid(text_memory_hr.Text, 2, 1)
  ElseIf Len(text_memory_hr.Text) > 2 Then
    text_memory_hr.Text = Mid(text_memory_hr.Text, 1, Len(text_memory_hr.Text) - 2) +
check_bit_hr_convert_number(1) + Mid(text_memory_hr.Text, Len(text_memory_hr.Text), 1)
  End If
ElseIf (Check_hr_bit(Index).Index = 8) Or (Check_hr_bit(Index).Index = 9) Or (Check_hr_bit
(Index).Index = 10) Or (Check_hr_bit(Index).Index = 11) Then
  If Len(text_memory_hr.Text) = 0 Then
    text_memory_hr.Text = check_bit_hr_convert_number(2) + "00"
```

```
    ElseIf Len(text_memory_hr.Text) = 1 Then
      text_memory_hr.Text = check_bit_hr_convert_number(2) + "0" + text_memory_hr.Text
    ElseIf Len(text_memory_hr.Text) = 2 Then
      text_memory_hr.Text = check_bit_hr_convert_number(2) + text_memory_hr.Text
    ElseIf Len(text_memory_hr.Text) = 3 Then
      text_memory_hr.Text = check_bit_hr_convert_number(2) + Mid(text_memory_hr.Text, 2, 2)
    ElseIf Len(text_memory_hr.Text) = 4 Then
      text_memory_hr.Text = Mid(text_memory_hr.Text, 1, 1) + check_bit_hr_convert_number(2) +
Mid(text_memory_hr.Text, 3, 2)
    End If
ElseIf (Check_hr_bit(Index).Index = 12) Or (Check_hr_bit(Index).Index = 13) Or (Check_hr_bit
(Index).Index = 14) Or (Check_hr_bit(Index).Index = 15) Then
    If Len(text_memory_hr.Text) = 0 Then
      text_memory_hr.Text = check_bit_hr_convert_number(3) + "000"
    ElseIf Len(text_memory_hr.Text) = 1 Then
      text_memory_hr.Text = check_bit_hr_convert_number(3) + "00" + text_memory_hr.Text
    ElseIf Len(text_memory_hr.Text) = 2 Then
      text_memory_hr.Text = check_bit_hr_convert_number(3) + "0" + text_memory_hr.Text
    ElseIf Len(text_memory_hr.Text) = 3 Then
      text_memory_hr.Text = check_bit_hr_convert_number(3) + text_memory_hr.Text
    ElseIf Len(text_memory_hr.Text) = 4 Then
      text_memory_hr.Text = check_bit_hr_convert_number(3) + Mid(text_memory_hr.Text, 2, 3)
    End If
End If
End Sub
```

```
Private Sub Check_lr_bit_Click(Index As Integer)
If (Check_lr_bit(Index).Index = 0) Or (Check_lr_bit(Index).Index = 1) Or (Check_lr_bit(Index).Index =
2) Or (Check_lr_bit(Index).Index = 3) Then
    If Len(text_memory_lr.Text) <= 1 Then
      text_memory_lr.Text = check_bit_lr_convert_number(0)
    ElseIf Len(text_memory_lr.Text) > 1 Then
```

```
        text_memory_lr.Text = Mid(text_memory_lr.Text, 1, Len(text_memory_lr.Text) - 1) +
check_bit_lr_convert_number(0)
    End If
ElseIf (Check_lr_bit(Index).Index = 4) Or (Check_lr_bit(Index).Index = 5) Or (Check_lr_bit
(Index).Index = 6) Or (Check_lr_bit(Index).Index = 7) Then
    If Len(text_memory_lr.Text) = 0 Then
        text_memory_lr.Text = check_bit_lr_convert_number(1) + "0"
    ElseIf Len(text_memory_lr.Text) = 1 Then
        text_memory_lr.Text = check_bit_lr_convert_number(1) + text_memory_lr.Text
    ElseIf Len(text_memory_lr.Text) = 2 Then
        text_memory_lr.Text = check_bit_lr_convert_number(1) + Mid(text_memory_lr.Text, 2, 1)
    ElseIf Len(text_memory_lr.Text) > 2 Then
        text_memory_lr.Text = Mid(text_memory_lr.Text, 1, Len(text_memory_lr.Text) - 2) +
check_bit_lr_convert_number(1) + Mid(text_memory_lr.Text, Len(text_memory_lr.Text), 1)
    End If
ElseIf (Check_lr_bit(Index).Index = 8) Or (Check_lr_bit(Index).Index = 9) Or (Check_lr_bit
(Index).Index = 10) Or (Check_lr_bit(Index).Index = 11) Then
    If Len(text_memory_lr.Text) = 0 Then
        text_memory_lr.Text = check_bit_lr_convert_number(2) + "00"
    ElseIf Len(text_memory_lr.Text) = 1 Then
        text_memory_lr.Text = check_bit_lr_convert_number(2) + "0" + text_memory_lr.Text
    ElseIf Len(text_memory_lr.Text) = 2 Then
        text_memory_lr.Text = check_bit_lr_convert_number(2) + text_memory_lr.Text
    ElseIf Len(text_memory_lr.Text) = 3 Then
        text_memory_lr.Text = check_bit_lr_convert_number(2) + Mid(text_memory_lr.Text, 2, 2)
    ElseIf Len(text_memory_lr.Text) = 4 Then
        text_memory_lr.Text = Mid(text_memory_lr.Text, 1, 1) + check_bit_lr_convert_number(2) + Mid
(text_memory_lr.Text, 3, 2)
    End If
ElseIf (Check_lr_bit(Index).Index = 12) Or (Check_lr_bit(Index).Index = 13) Or (Check_lr_bit
(Index).Index = 14) Or (Check_lr_bit(Index).Index = 15) Then
    If Len(text_memory_lr.Text) = 0 Then
```

```
        text_memory_lr.Text = check_bit_lr_convert_number(3) + "000"
    ElseIf Len(text_memory_lr.Text) = 1 Then
        text_memory_lr.Text = check_bit_lr_convert_number(3) + "00" + text_memory_lr.Text
    ElseIf Len(text_memory_lr.Text) = 2 Then
        text_memory_lr.Text = check_bit_lr_convert_number(3) + "0" + text_memory_lr.Text
    ElseIf Len(text_memory_lr.Text) = 3 Then
        text_memory_lr.Text = check_bit_lr_convert_number(3) + text_memory_lr.Text
    ElseIf Len(text_memory_lr.Text) = 4 Then
        text_memory_lr.Text = check_bit_lr_convert_number(3) + Mid(text_memory_lr.Text, 2, 3)
    End If
End If
End Sub
-----------------------------------------------------------------------------------------------------------
Private Sub Check_dm_bit_Click(Index As Integer)
If (Check_dm_bit(Index).Index = 0) Or (Check_dm_bit(Index).Index = 1) Or (Check_dm_bit
(Index).Index = 2) Or (Check_dm_bit(Index).Index = 3) Then
    If Len(text_memory_dm.Text) <= 1 Then
        text_memory_dm.Text = check_bit_dm_convert_number(0)
    ElseIf Len(text_memory_dm.Text) > 1 Then
        text_memory_dm.Text = Mid(text_memory_dm.Text, 1, Len(text_memory_dm.Text) - 1) +
check_bit_dm_convert_number(0)
    End If
ElseIf (Check_dm_bit(Index).Index = 4) Or (Check_dm_bit(Index).Index = 5) Or (Check_dm_bit
(Index).Index = 6) Or (Check_dm_bit(Index).Index = 7) Then
    If Len(text_memory_dm.Text) = 0 Then
        text_memory_dm.Text = check_bit_dm_convert_number(1) + "0"
    ElseIf Len(text_memory_dm.Text) = 1 Then
        text_memory_dm.Text = check_bit_dm_convert_number(1) + text_memory_dm.Text
    ElseIf Len(text_memory_dm.Text) = 2 Then
        text_memory_dm.Text = check_bit_dm_convert_number(1) + Mid(text_memory_dm.Text, 2, 1)
    ElseIf Len(text_memory_dm.Text) > 2 Then
```

```
        text_memory_dm.Text = Mid(text_memory_dm.Text, 1, Len(text_memory_dm.Text) - 2) +
 check_bit_dm_convert_number(1) + Mid(text_memory_dm.Text, Len(text_memory_dm.Text), 1)
    End If
 ElseIf (Check_dm_bit(Index).Index = 8) Or (Check_dm_bit(Index).Index = 9) Or (Check_dm_bit
 (Index).Index = 10) Or (Check_dm_bit(Index).Index = 11) Then
    If Len(text_memory_dm.Text) = 0 Then
       text_memory_dm.Text = check_bit_dm_convert_number(2) + "00"
    ElseIf Len(text_memory_dm.Text) = 1 Then
       text_memory_dm.Text = check_bit_dm_convert_number(2) + "0" + text_memory_dm.Text
    ElseIf Len(text_memory_dm.Text) = 2 Then
       text_memory_dm.Text = check_bit_dm_convert_number(2) + text_memory_dm.Text
    ElseIf Len(text_memory_dm.Text) = 3 Then
       text_memory_dm.Text = check_bit_dm_convert_number(2) + Mid(text_memory_dm.Text, 2, 2)
    ElseIf Len(text_memory_dm.Text) = 4 Then
       text_memory_dm.Text = Mid(text_memory_dm.Text, 1, 1) + check_bit_dm_convert_number(2) +
 Mid(text_memory_dm.Text, 3, 2)
    End If
 ElseIf (Check_dm_bit(Index).Index = 12) Or (Check_dm_bit(Index).Index = 13) Or (Check_dm_bit
 (Index).Index = 14) Or (Check_dm_bit(Index).Index = 15) Then
    If Len(text_memory_dm.Text) = 0 Then
       text_memory_dm.Text = check_bit_dm_convert_number(3) + "000"
    ElseIf Len(text_memory_dm.Text) = 1 Then
       text_memory_dm.Text = check_bit_dm_convert_number(3) + "00" + text_memory_dm.Text
    ElseIf Len(text_memory_dm.Text) = 2 Then
       text_memory_dm.Text = check_bit_dm_convert_number(3) + "0" + text_memory_dm.Text
    ElseIf Len(text_memory_dm.Text) = 3 Then
       text_memory_dm.Text = check_bit_dm_convert_number(3) + text_memory_dm.Text
    ElseIf Len(text_memory_dm.Text) = 4 Then
       text_memory_dm.Text = check_bit_dm_convert_number(3) + Mid(text_memory_dm.Text, 2, 3)
    End If
 End If
 End Sub
```

```
Private Sub Check_tc_bit_Click(Index As Integer)
If (Check_tc_bit(Index).Index = 0) Or (Check_tc_bit(Index).Index = 1) Or (Check_tc_bit(Index).Index
= 2) Or (Check_tc_bit(Index).Index = 3) Then
   If Len(text_memory_tc.Text) <= 1 Then
      text_memory_tc.Text = check_bit_tc_convert_number(0)
   ElseIf Len(text_memory_tc.Text) > 1 Then
      text_memory_tc.Text = Mid(text_memory_tc.Text, 1, Len(text_memory_tc.Text) - 1) +
check_bit_tc_convert_number(0)
   End If
ElseIf (Check_tc_bit(Index).Index = 4) Or (Check_tc_bit(Index).Index = 5) Or (Check_tc_bit
(Index).Index = 6) Or (Check_tc_bit(Index).Index = 7) Then
   If Len(text_memory_tc.Text) = 0 Then
      text_memory_tc.Text = check_bit_tc_convert_number(1) + "0"
   ElseIf Len(text_memory_tc.Text) = 1 Then
      text_memory_tc.Text = check_bit_tc_convert_number(1) + text_memory_tc.Text
   ElseIf Len(text_memory_tc.Text) = 2 Then
      text_memory_tc.Text = check_bit_tc_convert_number(1) + Mid(text_memory_tc.Text, 2, 1)
   ElseIf Len(text_memory_tc.Text) > 2 Then
      text_memory_tc.Text = Mid(text_memory_tc.Text, 1, Len(text_memory_tc.Text) - 2) +
check_bit_tc_convert_number(1) + Mid(text_memory_tc.Text, Len(text_memory_tc.Text), 1)
   End If
ElseIf (Check_tc_bit(Index).Index = 8) Or (Check_tc_bit(Index).Index = 9) Or (Check_tc_bit
(Index).Index = 10) Or (Check_tc_bit(Index).Index = 11) Then
   If Len(text_memory_tc.Text) = 0 Then
      text_memory_tc.Text = check_bit_tc_convert_number(2) + "00"
   ElseIf Len(text_memory_tc.Text) = 1 Then
      text_memory_tc.Text = check_bit_tc_convert_number(2) + "0" + text_memory_tc.Text
   ElseIf Len(text_memory_tc.Text) = 2 Then
      text_memory_tc.Text = check_bit_tc_convert_number(2) + text_memory_tc.Text
   ElseIf Len(text_memory_tc.Text) = 3 Then
      text_memory_tc.Text = check_bit_tc_convert_number(2) + Mid(text_memory_tc.Text, 2, 2)
   ElseIf Len(text_memory_tc.Text) = 4 Then
```

```
    text_memory_tc.Text = Mid(text_memory_tc.Text, 1, 1) + check_bit_tc_convert_number(2) + Mid
(text_memory_tc.Text, 3, 2)
  End If
ElseIf (Check_tc_bit(Index).Index = 12) Or (Check_tc_bit(Index).Index = 13) Or (Check_tc_bit
(Index).Index = 14) Or (Check_tc_bit(Index).Index = 15) Then
  If Len(text_memory_tc.Text) = 0 Then
    text_memory_tc.Text = check_bit_tc_convert_number(3) + "000"
  ElseIf Len(text_memory_tc.Text) = 1 Then
    text_memory_tc.Text = check_bit_tc_convert_number(3) + "00" + text_memory_tc.Text
  ElseIf Len(text_memory_tc.Text) = 2 Then
    text_memory_tc.Text = check_bit_tc_convert_number(3) + "0" + text_memory_tc.Text
  ElseIf Len(text_memory_tc.Text) = 3 Then
    text_memory_tc.Text = check_bit_tc_convert_number(3) + text_memory_tc.Text
  ElseIf Len(text_memory_tc.Text) = 4 Then
    text_memory_tc.Text = check_bit_tc_convert_number(3) + Mid(text_memory_tc.Text, 2, 3)
  End If
End If
End Sub
```