การใช้เพททริเนทส์ช่วยในการออกแบบวงจรแลดเดอร์

APPLIED PETRI NETS AID DESIGNING LADDER DIAGRAM

นายประเทือง ปัญญาคม

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมการวัดคุม

ภาควิชาวิศวกรรมการวัดคุม คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2545

APPLIED PETRI NETS AID DESIGNING LADDER DIAGRAM

PRATUANG PANYAKOM

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

BACHELOR OF ENGINEERING IN INSTRUMENTATION ENGINEERING

DEPARTMENT OF INSTRUMENTATION ENGINEERING

FACULTY OF ENGINEERING

KING MONGKUT'SINSTITUTE OF TECHNOLOGY LADKRABANG

2002

ภาควิชาวิศวกรรมการวัดคุม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ใบรับรองปริญญานิพนธ์

หัวข้อปริญญานิพนธ์ การใช้เพททริเนทส์ช่วยในการออกแบบวงจรแลดเดอร์

APPLIED PETRI NETS AID DESIGNING LADDER DIAGRAM

นักศึกษาผู้จัดทำ นายประเทือง ปัญญาคม

ปริญญา วิศวกรรมศาสตรบัณฑิต

สาขาวิชา วิศวกรรมการวัดคุม

ปีการศึกษา 2545

อาจารย์ผู้ควบคุมปริญญานิพนธ์	ลายมือชื่อ
ผศ. ทวีพล ซื่อสัตย์	

วัน/เดือน/ปี ที่สอบ วันอังคารที่ 25 มีนาคม พ.ศ. 2546

สถานที่สอบ ณ ห้องสอบปริญญานิพนธ์ ภาควิชาวิศวกรรมการวัดคุม

ภาควิชารับรองแล้ว

(ผศ.ประสิทธ์ จุลเสรีวงศ์)

หัวหน้าภาควิชาวิศวกรรมการวัดคุม

หัวข้อปริญญานิพนธ์	การใช้เพททริเนทส์ช่วยในการออกแบบวงจรแลดเดอร์
	APPLIED PETRI NETS AID DESIGNING LADDER DIAGRAM
นักศึกษาผู้จัดทำ	นายประเทือง ปัญญาคม
อาจารย์ที่ปรึกษา	ผศ.ทวีพล ซื่อสัตย์
ปีการศึกษา	2545

บทคัดย่อ

เครื่องควบคุมแบบตรรกะโปรแกรมได้ (PLC) ได้มีใช้กันอย่างกว้างขวางในวงการอุตสาหกรรม โดยมีมาตรฐานในการเขียนโปรแกรมคือ ICE 1131-3 ภาษาแลดเดอร์เป็นภาษาหนึ่งซึ่งได้รับความนิยมสูงที่สุด โดยมีลักษณะเหมือนกับวงจรไฟฟ้า ซึ่งไม่มีการจัดลำดับและขั้นตอนที่ชัดเจน จึงเป็นการยากในการทำความเข้าใจและพัฒนาโปรแกรม Petri net เป็นตัวแบบเชิงภาพที่สามารถอธิบายขั้นตอนการทำงานได้ง่ายต่อการทำความเข้าใจ ในปริญญานิพนธ์ฉบับนี้ได้นำเสนอวิธีการเขียนวงจรแลดเดอร์โดยการเขียนเงื่อนไขการทำงานให้อยู่ในรูปแบบของ Petri nets หลังจากนั้นจึงแปลงเป็นรูปแบบของวงจรแลดเดอร์ และในโครงงานนี้ได้ออกแบบโปรแกรมคอมพิวเตอร์ที่จะนำมาใช้ช่วยในการศึกษาเรื่องการเขียนวงจรแลดเดอร์ จาก Petri nets โดยสามารถที่จะเขียนโปรแกรมการควบคุมในรูปแบบของ Petri nets และใช้ควบคุมอุปกรณ์ภายนอกผ่านโมดุลอินพุตและโมดุลเอาท์พุต ของ PLC จากผลการทดลองนำวงจรแลดเดอร์ที่ได้มาโดยการแปลงมาจากรูปแบบของ Petri nets แล้วนำมาทำการป้อนโปรแกรมให้กับ PLC และการเขียนโปรแกรมควบคุมบนโปรแกรมที่สร้างขึ้น แสดงให้เห็นว่าสามารถใช้วงจรแลดเดอร์ที่ได้มาจากการใช้ทฤษฎีของ Petri nets ช่วย สามารถใช้ในการควบคุมระบบการทำงานได้เป็นอย่างดี

Thesis Title	Applied Petri nets aid Designing Ladder Diagram
Author	Mr.Pratuang Panyakom
Thesis Advisor	Asst.Prof.Taweepol Suesut
Year	2002

ABSTRACT

The Programmable logic Controller (PLC) is widely used in industrial by using the IEC 1131-3 programming language. The ladder diagram is the most popular language that similar the electrical schematic diagrams which does not have the step and sequence of the operation. Therefore it is difficult to understand the program structure and developing the system. The Petri nets is one of graphic models that can be explained the system operating easily and clearly. This thesis present the computer software that developed by Delphi to transform petri nets to Ladder diagram and implementation to control the process through PLC's input and output. This program can be applied to another applications on any PLC as well.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงได้ด้วยดีเพราะได้รับความเมตตาจากผู้ช่วยศาสตราจารย์ทวีพล ซื่อสัตย์ ที่ได้ให้คำแนะนำแก่ผู้วิจัยตลอดมาอีกทั้งยังเอื้อเฟื้ออุปกรณ์และเครื่องมือต่างๆ ในการทำปริญญานิพนธ์นี้ ผู้วิจัยรู้สึกซาบซึ้งและขอกราบขอบพระคุณเป็นอย่างสูง

ขอขอบพระคุณอาจารย์ภาควิชาวิศวกรรมการวัดคุมทุกท่านที่ได้ให้คำปรึกษา แนะนำ และประสิทธ์ประสาทวิชาความรู้ให้กับข้าพเจ้าตลอดมา จนสามารถทำปริญญานิพนธ์ฉบับนี้ได้สำเร็จ

และที่ลืมเสียมิได้ ขอกราบขอบพระคุณ บิดา มารดา อันเป็นที่รักยิ่ง และพี่ๆ ที่สนับสนุนช่วยเหลือ และเป็นแรงบัลดาลใจในการทำปริญญานิพนธ์ฉบับนี้

คุณค่าและประโยชน์อันพึงมีจากปริญญานิพนธ์ฉบับนี้ ผู้วิจัยขอมอบแด่ผู้มีพระคุณทุกท่าน

ผู้จัดทำ

สารบัญ

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

สารบัญรูป

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

บทที่ 1
บทนำ

1.1 ความเป็นมาและเหตุจูงใจของการวิจัย

ในการอธิบายการทำงานของเครื่องจักรกลโดยใช้คำพูดหรือลายลักษณ์อักษรนั้น แม้จะเป็นวิธีที่ง่าย แต่ไม่มีกฎเกณฑ์ที่แน่นอน และไม่สามารถที่จะถ่ายทอดรายละเอียดของการทำงานของเครื่องจักรกลให้ง่ายต่อการทำความเข้าใจได้ ซึ่งถ้าการควบคุมเครื่องจักรกลนั้นมีความยุ่งยาก และซับซ้อนมาก ๆ ถ้าอธิบายด้วยคำพูดก็คงจะต้องใช้เวลานานในการทำความเข้าใจ ดังนั้นโดยมากแล้วมักจะอธิบายการทำงานของเครื่องจักรกลในรูปของผังเวลาการทำงาน และโฟว์ชาร์ตเป็นส่วนใหญ่

ปัจจุบันการเขียนวงจรแลดเดอร์เพื่อใช้สำหรับโปรแกรมการทำงานต่างๆ ที่ต้องการให้กับเครื่องควบคุมแบบตรรกะที่โปรแกรมได้ (Programmable Logic Controller) มักจะเขียนด้วยวิธีการดัดแปลงมาจากวงจรรีเลย์ หรือเขียนตามผังเวลาการทำงานของเครื่องจักรกล ซึ่งทั้งสองวิธีดังกล่าวไม่มีรูปแบบขั้นตอนที่ชัดเจน โดยส่วนมากแล้วจะขึ้นอยู่กับประสบการณ์ของผู้ใช้งานเอง จึงต้องอาศัยผู้ชำนาญงานมาช่วยในการออกแบบโปรแกรมระบบควบคุม กรณีเกิดเหตุขัดข้องขึ้น หรือต้องการที่จะเปลี่ยนแปลงเงื่อนไขการทำงานของเครื่องจักรเพียงเล็กน้อย หรือในขั้นตอนการเขียนโปรแกรมนั้นเกิดปัญหาขึ้นก็จะประสบกับปัญหาในการตรวจสอบหาจุดบกพร่องของเครื่องจักรหรือจุดบกพร่องภายในวงจรแลดเดอร์ ทำให้เกิดความล่าช้าในการตรวจสอบหาจุดบกพร่องหรือข้อผิดพลาดที่เกิดขึ้น ทั้งนี้เพราะในรูปแบบของวงจรแลดเดอร์ไม่ได้แสดงลำดับขั้นตอนการทำงานของเครื่องจักรกลที่เราควบคุมอยู่ให้เห็นอย่างชัดเจน จากที่ได้ศึกษาทฤษฎีของ Petri nets. ซึ่งจะเป็นการจำลองกระบวนการของระบบการทำงานต่าง ๆ ให้อยู่ในลักษณะของรูปภาพแทนในส่วนต่าง ๆ ของระบบการทำงานที่สนใจ ซึ่งจะเป็นลำดับขั้นตอนของการทำงานที่เห็นได้ชัดเจน จึงช่วยให้ง่ายในการออกแบบโปรแกรมการควบคุมการทำงาน และแก้ไขการทำงานของเครื่องจักรกลได้ง่ายขึ้น และยังสามารถใช้เป็นเครื่องมือเพื่อช่วยในการวิเคราะห์การทำงานของระบบต่าง ๆ ได้ เช่น เกิดการหยุดนิ่งของระบบการทำงาน การรวนซ้ำ เป็นต้น

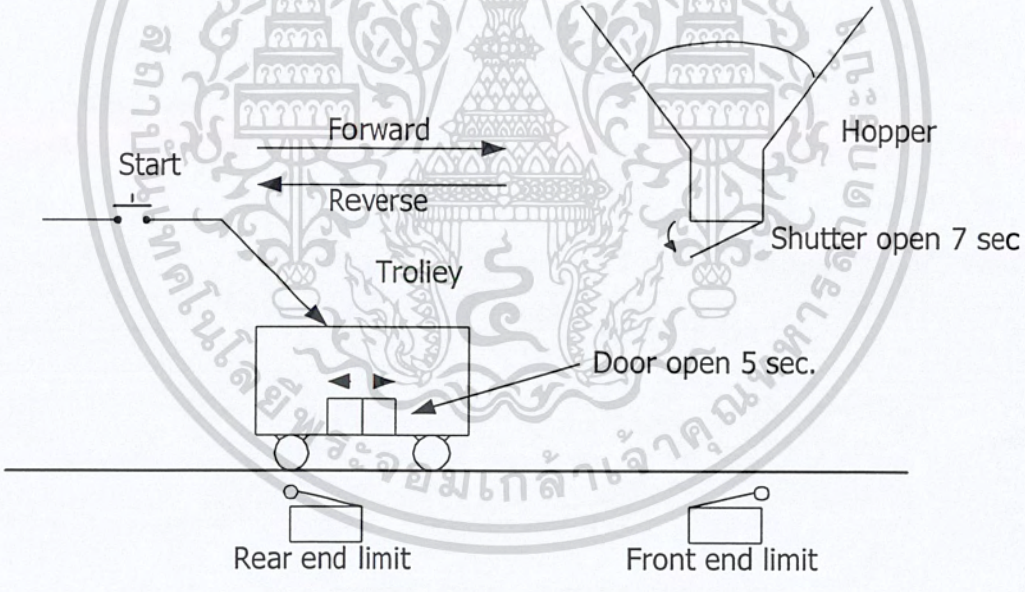
จากที่เราได้กล่าวถึงการอธิบายระบบการทำงานของเครื่องจักรกลในแบบต่าง ๆ มาแล้วนั้น เพื่อให้เห็นชัดเจนยิ่งขึ้นดังนั้น จึงยกตัวอย่างการอธิบายระบบการทำงานในรูปแบบต่าง ๆ เพื่อใช้ในการเปรียบเทียบ

ตัวอย่างที่ 1.1 ระบบขนถ่ายสินค้าด้วยรถเลื่อน

การอธิบายด้วยลายลักษณ์อักษร

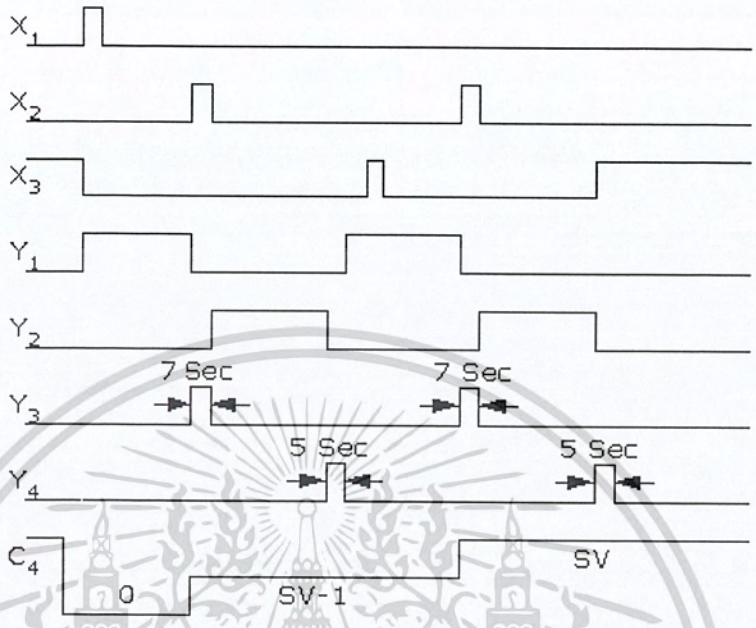
ลำดับการทำงานของเครื่องจักรมีเงื่อนไขการทำงานดังนี้

1. เมื่อกดสวิทช์เริ่มทำงานตัวรถบรรทุกของจะเลื่อนไปทางด้านขวามือและหยุดอยู่ในตำแหน่งใต้ถังเก็บของ
2. ฝาถังเก็บของเปิดออกเพื่อปล่อยสิ่งของออกมานาน 7 วินาที
3. หลังจากนั้นให้รถบรรทุกของกลับมาทางด้านซ้ายมือ แล้วเปิดประตูปล่อยของออกเป็นเวลานาน 5 วินาที
4. ตัวรถบรรทุกจะเลื่อนกลับไปรับสิ่งของ เพื่อขนถ่ายสินค้า 2 รอบแล้วหยุด



รูปที่ 1.1 โครงสร้างภายนอกของเครื่องจักร

การอธิบายโดยผังเวลาการทำงาน



รูปที่ 1.2 แสดงผังเวลาในการทำงาน

X1 คือ สวิตช์เริ่มต้น

X2 คือ ลิมิตสวิตช์ด้านขวา

X3 คือ ลิมิตสวิตช์ด้านซ้าย

Y1 คือ เอาท์พุตสั่งเลื่อนไปทางขวา

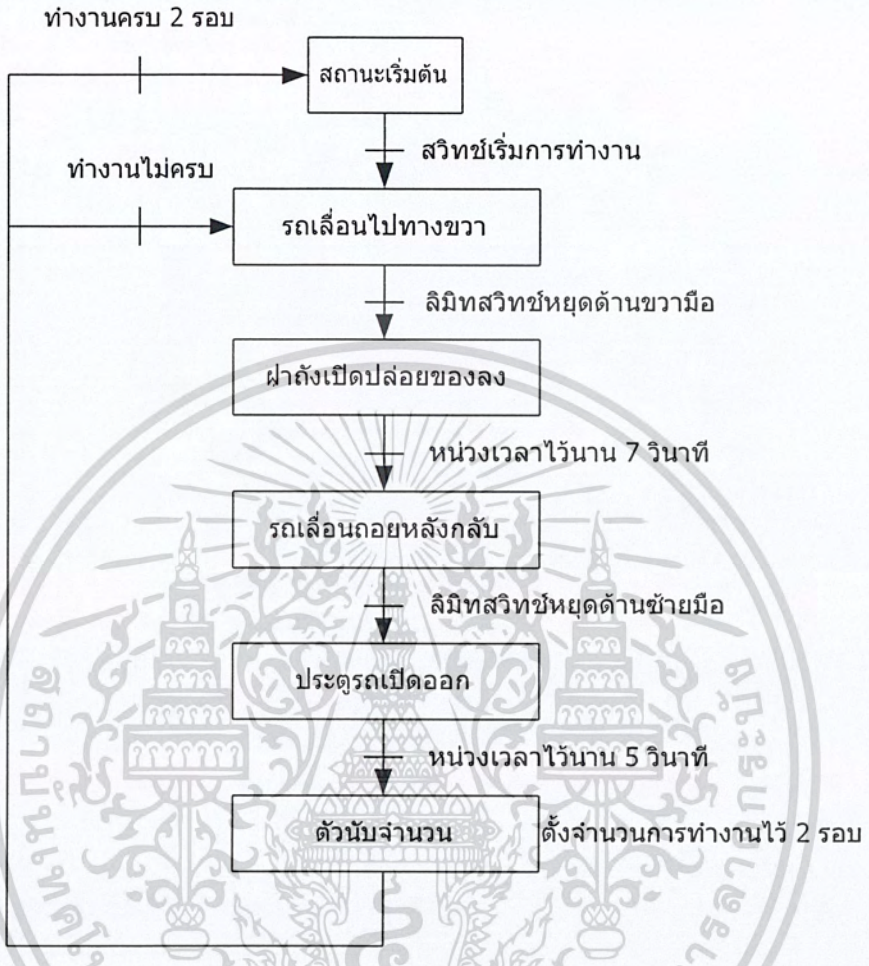
Y2 คือ เอาท์พุตสั่งเลื่อนไปทางซ้าย

Y3 คือ เอาท์พุตสั่งเปิดฝาถัง

Y4 คือ เอาท์พุตสั่งเปิดประตูรถ

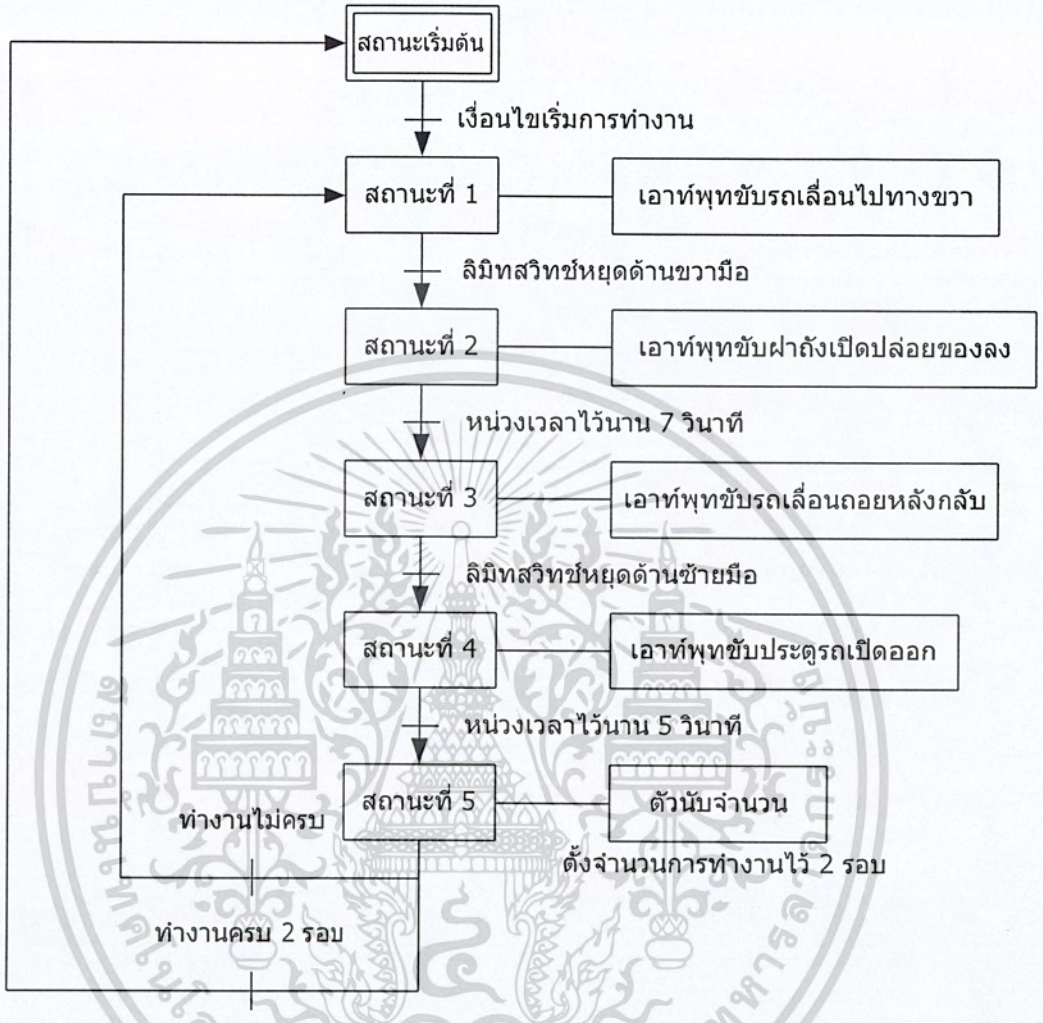
C4 คือ ตัวนับจำนวน(Counter)

การอธิบายโดยแผนผังการทำงาน



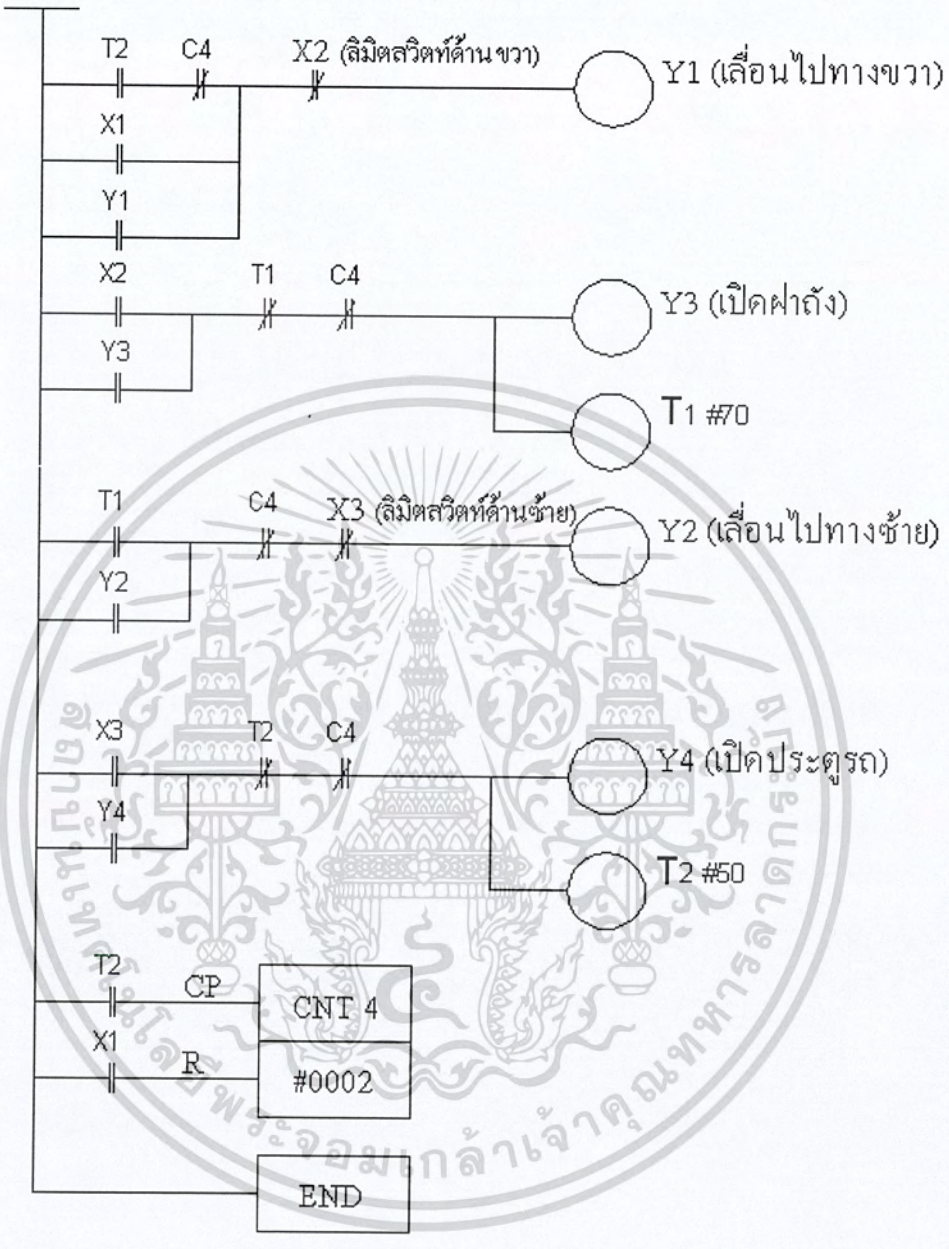
รูปที่ 1.3 แสดงแผนผังการทำงาน

การอธิบายโดย Grafcet



รูปที่ 1.4 รูปแบบของ Grafcet

การอธิบายโดยวงจรแลดเดอร์



รูปที่ 1.5 รูปแบบของวงจรแลดเดอร์

จากการอธิบายการทำงานของเครื่องจักรแบบต่าง ๆ นั้นจะเห็นได้ว่าในรูปแบบของวงจรแลดเดอร์ไม่มีลำดับขั้นตอนการทำงานของเครื่องจักรกลเลย ซึ่งแตกต่างกับรูปแบบของ Gratect ซึ่งเป็นทฤษฎีที่ได้ประยุกต์มาจาก Petri Nets. และมีลักษณะรูปแบบที่ช่วยให้เราเห็นภาพลำดับขั้นตอนการทำงานที่ชัดเจนและง่ายต่อการทำความเข้าใจ

1.2 วัตถุประสงค์ของปริญญานิพนธ์

ในปริญญานิพนธ์นี้จะเป็นการศึกษาเกี่ยวกับทฤษฎีของ Petri Nets. และวงจรแลดเดอร์ของเครื่องควบคุมแบบตรรกะที่โปรแกรมได้(Programmable Logic Controller) โดยมีวัตถุประสงค์ที่จะนำทฤษฎีของ Petri Nets. มาประยุกต์ใช้เพื่อช่วยในการเขียนโปรแกรมการควบคุมที่ง่ายขึ้น ลดเวลาและความยุ่งยากในการเขียนโปรแกรมที่ต้องการจะนำไปใช้ควบคุมเครื่องจักรกลให้ทำงานตามที่ต้องการ กล่าวคือ สามารถที่จะเขียนวงจรแลดเดอร์โดยการเขียนจากการอธิบายการทำงานของระบบการทำงานของเครื่องจักรกลในแบบต่าง ๆ ให้อยู่ในรูปแบบของ Petri Net หลังจากนั้น จึงแปลงจากรูปแบบของ Petri Nets ให้อยู่ในรูปแบบของวงจรแลดเดอร์เพื่อนำไปใช้ในการป้อนโปรแกรมให้กับเครื่องควบคุมแบบตรรกะที่โปรแกรมได้ต่อไป และเขียนโปรแกรมเพื่อให้ผู้ใช้งานเขียนโปรแกรมควบคุมอุปกรณ์ภายนอกและสามารถจำลองเหตุการณ์ในรูปแบบของ Grafcet

1.3 ขอบเขตของปริญญานิพนธ์

1) สามารถที่เขียนวงจรแลดเดอร์โดยการเขียนเป็น Petri Nets จากเงื่อนไขการทำงานแล้วทำการแปลงจาก Petri Net. มาเป็นวงจรแลดเดอร์ เพื่อใช้ในการโปรแกรมให้กับเครื่องควบคุมต่อไป
2) เขียนโปรแกรมเพื่อให้ผู้ใช้งานเขียนโปรแกรมในการรับค่าสถานะทางลอจิกจากอุปกรณ์ภายนอกผ่านทางอินพุตโมดุลของเครื่องควบคุม นำค่าที่ได้มาประมวลผลโดยโปรแกรมบนเครื่องคอมพิวเตอร์หลังจากนั้นส่งผลที่ได้จากการประมวลผลไปควบคุมอุปกรณ์ภายนอกผ่านทางเอาท์พุตของเครื่องควบคุม
3) สามารถตรวจสอบความผิดพลาดของโปรแกรมที่ผู้ใช้งานเขียนขึ้นมาได้
4) สามารถจำลองการทำงานของโปรแกรมที่ผู้ใช้งานเขียนขึ้นมาได้

บทที่ 2
ทฤษฎีและหลักการที่เกี่ยวข้อง

2.1 ทฤษฎีของเพททริเนทส์ (Petri net.)

ทฤษฎีด้าน Petri netที่ยกมากล่าวนี้ได้มาจากการรวบรวม และศึกษาจากหนังสือ บทความ และเวปไซต์ ที่ได้มีอยู่ เราได้รวบรวมนำเนื้อหาบางส่วนที่เกี่ยวกับการทำปริญญานิพนธ์เพื่อเขียนเป็นทฤษฎีในส่วนนี้ด้วย

2.1.1 ส่วนประกอบของเพ่ททริเนทส์เบื้องต้น

จากที่ได้อธิบายไปแล้วว่า Petri net เป็นแบบจำลองทางด้านกราฟฟิกสำหรับระบบต่าง ๆ โดยการสร้างและใช้แบบจำลองแทนส่วนประกอบต่าง ๆ ในระบบที่เราสนใจ ดังนั้นแบบจำลองนี้จะประกอบด้วยส่วนต่าง ๆ และการใช้งานดังนี้

1. Places ใช้สัญลักษณ์เป็นวงกลม Places สามารถแบ่งเป็น

1.1 input places แทนเงื่อนไขเริ่มต้น (Pre Condition) สำหรับเหตุการณ์หนึ่ง ๆ

1.2 output places แทนเงื่อนไขที่ได้ (Post Condition) จากการทำเหตุการณ์หนึ่ง ๆ

ในแต่ละ Place จะมีพร๊อพเพอร์ตี้ อยู่ได้แก่

- Capacity ระบุจำนวนสูงสุดของ Token ที่ Place สามารถเก็บไว้ได้
- Marking ระบุจำนวน Token ที่ Place มีอยู่

2. Transitions ใช้สัญลักษณ์เป็นสี่เหลี่ยมด้านขนานหรือจตุรัสแทนเหตุการณ์ของระบบในแต่ละ Transition จะมีพร๊อพเพอร์ตี้อยู่ได้แก่

- Priority กำหนดความสำคัญของ Transition นั้น โดยค่า 0 จะมีค่าสูงสุด
- Probability กำหนดความน่าจะเป็นของการเลือก Transition นั้น โดยระบบ

3. Token ใช้สัญลักษณ์เป็นจุดสีดำซึ่งจะปรากฏเป็นจุดสีดำเราจะแทนเงื่อนไขที่จะพิจารณาด้วยสัญลักษณ์ของ Token ดังนั้น Token จะอยู่ใน Place ซึ่งจะหมายถึงมีเงื่อนไขตามจำนวนของ Tokens เข้ามาในเหตุการณ์ (Transition)

4. Arcs ใช้สัญลักษณ์เป็นเส้นที่เชื่อมต่อระหว่าง Transitions และ Places ซึ่งจะบอกทิศทางของเงื่อนไขกับเหตุการณ์ Arcs แบ่งเป็น 2 ประเภทตามทิศทาง

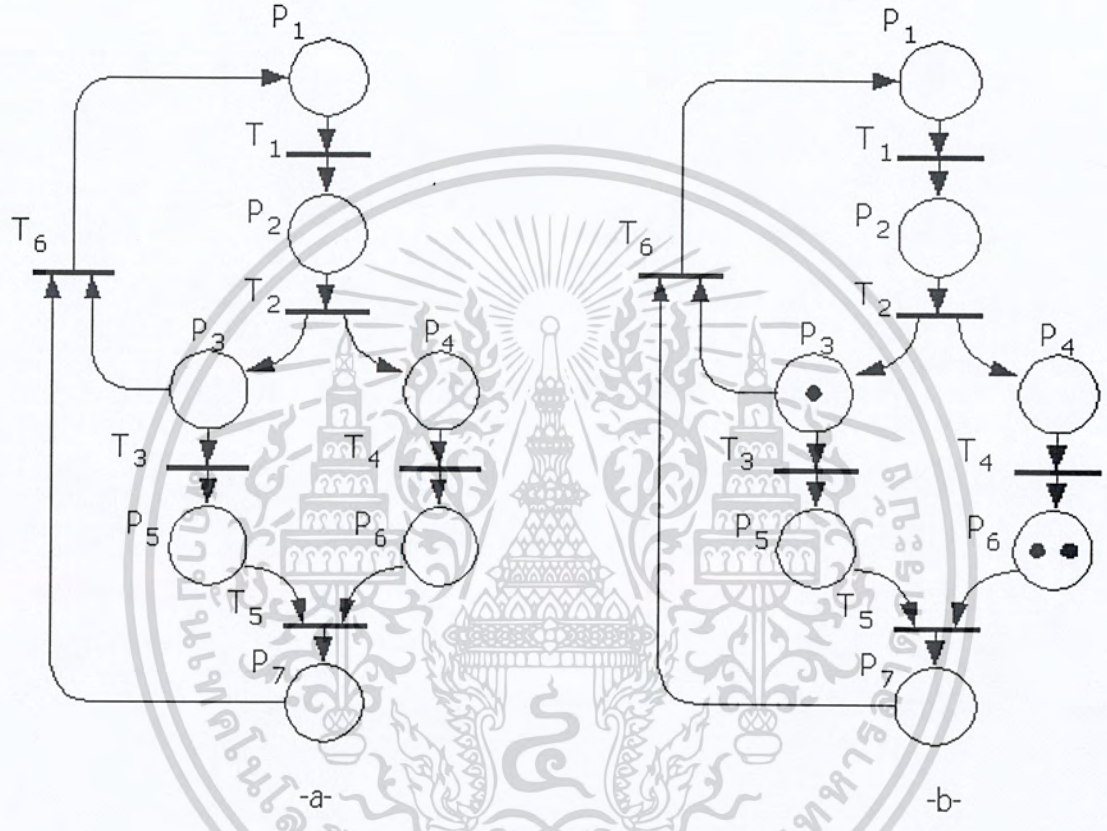
4.1 PreArcs มีทิศทางจาก Place ไป Transition

มี Arc Weight w(p,t) ระบุจำนวน Token ที่ต้องการสำหรับการผ่านในแต่ละครั้ง

4.2 PostArcs มีทิศทางจาก Transition ไป Place

มี Arc Weight w(t,p) ระบุจำนวน Token ที่จะเกิดขึ้นสำหรับการผ่าน Arc นี้ในแต่ละครั้ง

จากรูปที่ 2.1 Place จะแทนโดยรูปวงกลม Transition จะแทนด้วยบัสหรือกล่องสี่เหลี่ยม โดยที่ Place และ Transition จะเชื่อมต่อกันโดย arcs ซึ่งใน arcs นี้ก็จะมีอยู่ 2 รูปแบบ คือ arc ที่ต่อจาก Place ไป Transition และจาก Transition ไป Place



รูปที่ 2.1 a) Not marked Petri net. B) marked Petri net.

ในรูปที่ 2.1 จะมี 7 Place, 6 Transition และ 15 arc เราจะเรียกเซ็ตของ Place ว่า P. สำหรับตัวอย่างนี้จะมี P = {P1,P2,P3,P4,P5,P6,P7} เซ็ตของ Transition คือ T = {T1,T2,T3,T4,T5,T6} Place P 3 เรียกว่า Place ต้นทาง หรือ input Place ของ T3 เพราะเป็นการเชื่อมต่อเข้าโดยตรงจาก P3 ไป T3 Place P5 เป็น Place ปลายทางหรือ output Place ของ T3 เนื่องจากเป็น Place ที่ถูกเชื่อมจาก T3 ไป P5 และในทางเดียวกัน Place และ Transition อื่น ๆ ที่มีรูปแบบดังกล่าวก็สามารถที่จะเรียกได้เหมือนกัน ส่วนใน Transition ที่ไม่มี input Place เรียกว่า Source Transition และ Transition ที่ปราศจาก output Place จะเรียกว่า Sink Transition

2.1.2 การทำเครื่องหมาย (Marking)

ในรูปที่ 2.1b จะเป็นการแสดงถึงการ Marked ของ Petri net ซึ่งแต่ละ Placeจะบรรจุด้วยค่าจำนวนเต็มบวก ซึ่งเรียกว่า Token หรือ Marks ซึ่งจะบรรจุอยู่ใน place Pi เราจะสามารถเรียกว่า M(Pi) หรือ mi สำหรับในตัวอย่างในรูปที่ 2.1b นั้นเราจะได้ว่า m1 = m3 = 1 , m6 = 2 และ m2 = m4 = m5 = m7 = 0 ซึ่งข่ายของการ marking (M) จะสามารถกำหนดโดยเวคเตอร์ได้ดังนี้คือ

M = (m1,m2,m3,m4,m5,m6,m7) ซึ่งจากรูปที่ 2.1b ก็จะได้ว่า

M = (1,0.1,0,0,2,0) โดยที่ marking นี้สามารถเปลี่ยนแปลงได้โดยการ firing ของ Transition ซึ่งการ firing เราก็จะได้นำเสนอในหัวข้อต่อไป

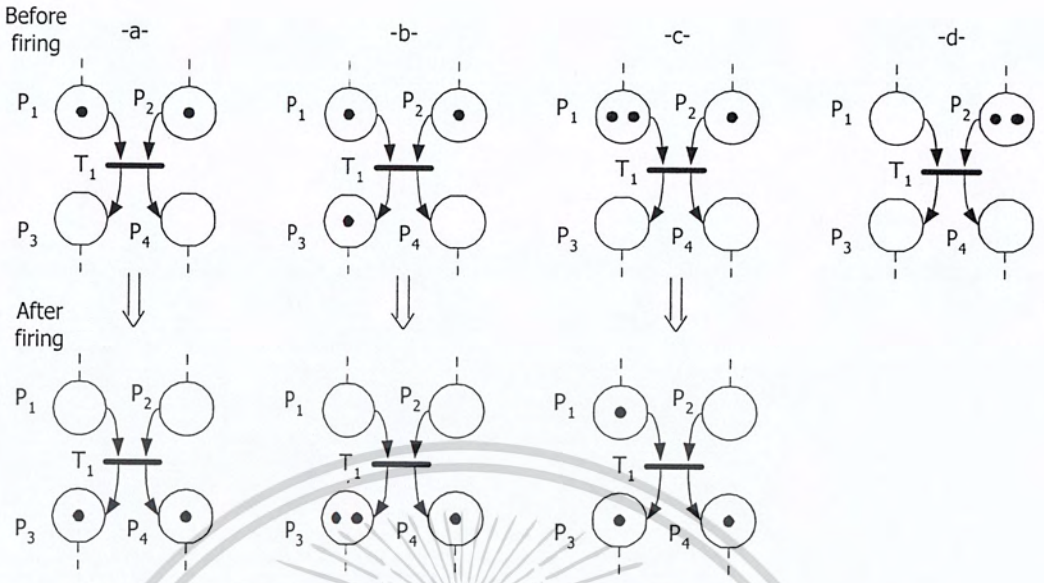
2.1.3 การส่งผ่าน Tokens ของทรานซิสชั่น (Firing of a Transition)

หมายความว่าการเคลื่อนที่และการเปลี่ยนแปลงของ Tokens ที่อยู่ใน Places แรกและจะเคลื่อนที่ไปสู่อีก Places อื่น โดยผ่าน Transition เทียบกับระบบได้ว่าการมีเงื่อนไขที่พร้อม (จำนวน token ใน places แรก) สำหรับการเกิดเหตุการณ์ (transition) และผลที่ได้จากการเกิดเหตุการณ์จะเป็นอย่างไร (จำนวน token ที่เกิดขึ้นใน place อื่น) นั้นมีเงื่อนไขอย่างไร

1. transition t จะบอกได้ว่า enable (พร้อมที่จะทำ firing) เมื่อทุก input place p ของ t มีจำนวน tokens อย่างน้อย w(p,t) เท่ากับ weight ของ arcs จาก p ไป t (Pre Arcs)
2. transition ที่ enable จะทำ firing หรือไม่ก็ได้
3. การ fire สำหรับ enable transition t จะย้าย token จำนวน w(p,t) สำหรับแต่ละ input place p ของ t และจะสร้าง tokens จำนวน w(t,p) ให้กับแต่ละ output place p ของ t ซึ่ง w(t,p) คือ weight ของแต่ละ arc จาก t ไป p

จากรูปที่ 2.2 a, 2.2b , 2.2c (ก่อนการfiring) มันจะ enabled เพราะในแต่ละ Place P1,P2 มี Token อย่างน้อยที่สุดเท่ากับหนึ่งทั้งนั้น ส่วนในรูปที่ 2.2 d นั้นไม่ enable เพราะ P1 ไม่มี Token

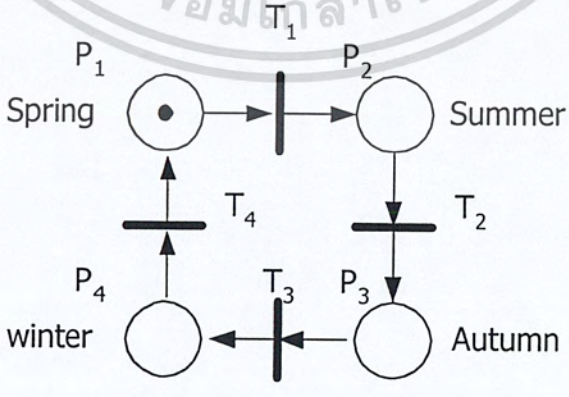
การ firing ของ Transition Tj จะประกอบไปด้วยการนำ Token จาก input place ไปยัง output place โดยจะต้องคำนึงถึงของกำหนดของ arc ที่ใช้ในการส่งผ่าน Transition และกฎการ enable ของ Transition ด้วย ซึ่งจะแสดงให้เห็นได้ดังรูปที่ 2.2



รูปที่ 2.2 Firing of Transition

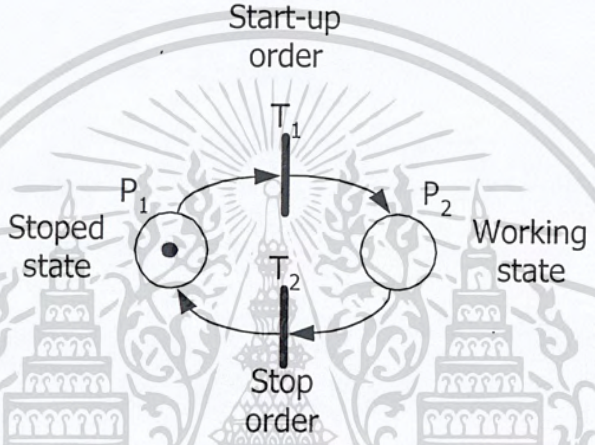
2.1.4 ระบบอัตโนมัติและระบบไม่อัตโนมัติ(Autonomous and non autonomous Petri net.)

จากรูปที่ 2.3 เป็นรูปของช่วงฤดูการณ์ซึ่งแต่ละ Place นั้นจะเป็นฤดูกาล และ ฤดูกาลนี้มันก็จะเปลี่ยนไปได้โดยการที่ รอให้ Transition enable ซึ่งมันจะเปลี่ยนจากฤดูกาลหนึ่งไปยังอีกฤดูกาล เมื่อเราพิจารณาพฤติกรรมของการเปลี่ยนแปลงฤดูกาลนี้ มันจะค่อยเปลี่ยนแปลงขึ้นและจะทำให้เกิด Transition enable ขึ้นและ firing ไปตามกาลเวลาทันทีซึ่งการเกิดเหตุการณ์ลักษณะนี้เราจะสามารถพูดได้ว่ามันเป็นพฤติกรรมของ Autonomous Petri net



รูปที่ 2.3 Autonomous Petri net

จากในรูปที่ 2.4 จะเป็นการแสดงวงรอบการทำงานของมอเตอร์ในการเริ่มต้นทำงานและหยุดทำงาน โดยจะเริ่มการทำงานจากการอยู่ที่ตำแหน่งหยุดการทำงานไปตำแหน่งเริ่มทำงาน และกลับมายังตำแหน่งหยุดการทำงานอีกครั้ง ในรูปที่ 2.4 นั้น สถานะเริ่มต้นนั้นจะมี Marking อยู่ที่สถานะหยุดการทำงานซึ่งมันจะเริ่มทำงานได้ก็ต่อเมื่อ Transition ของการสตาร์ท (T1) enable แต่ T1 จะ enable ได้นั้นจะต้องอาศัยการสั่งงานมาจากภายนอก ซึ่งพฤติกรรมแบบนี้จะเรียกได้ว่าเป็น non-autonomous Petri net



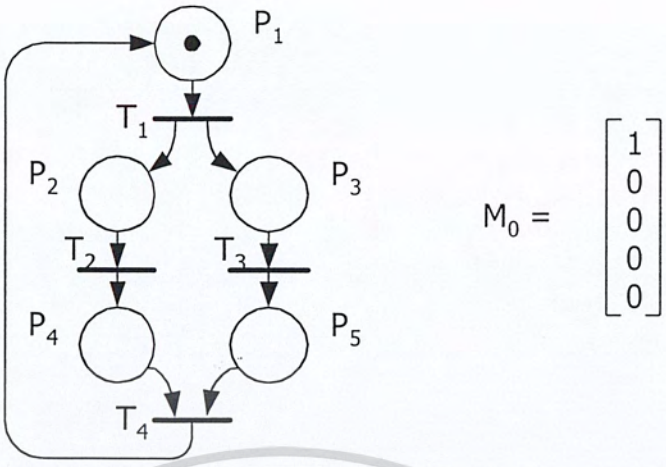
รูปที่ 2.4 non-autonomous Petri net

2.1.5 สัญลักษณ์และคำจำกัดความ

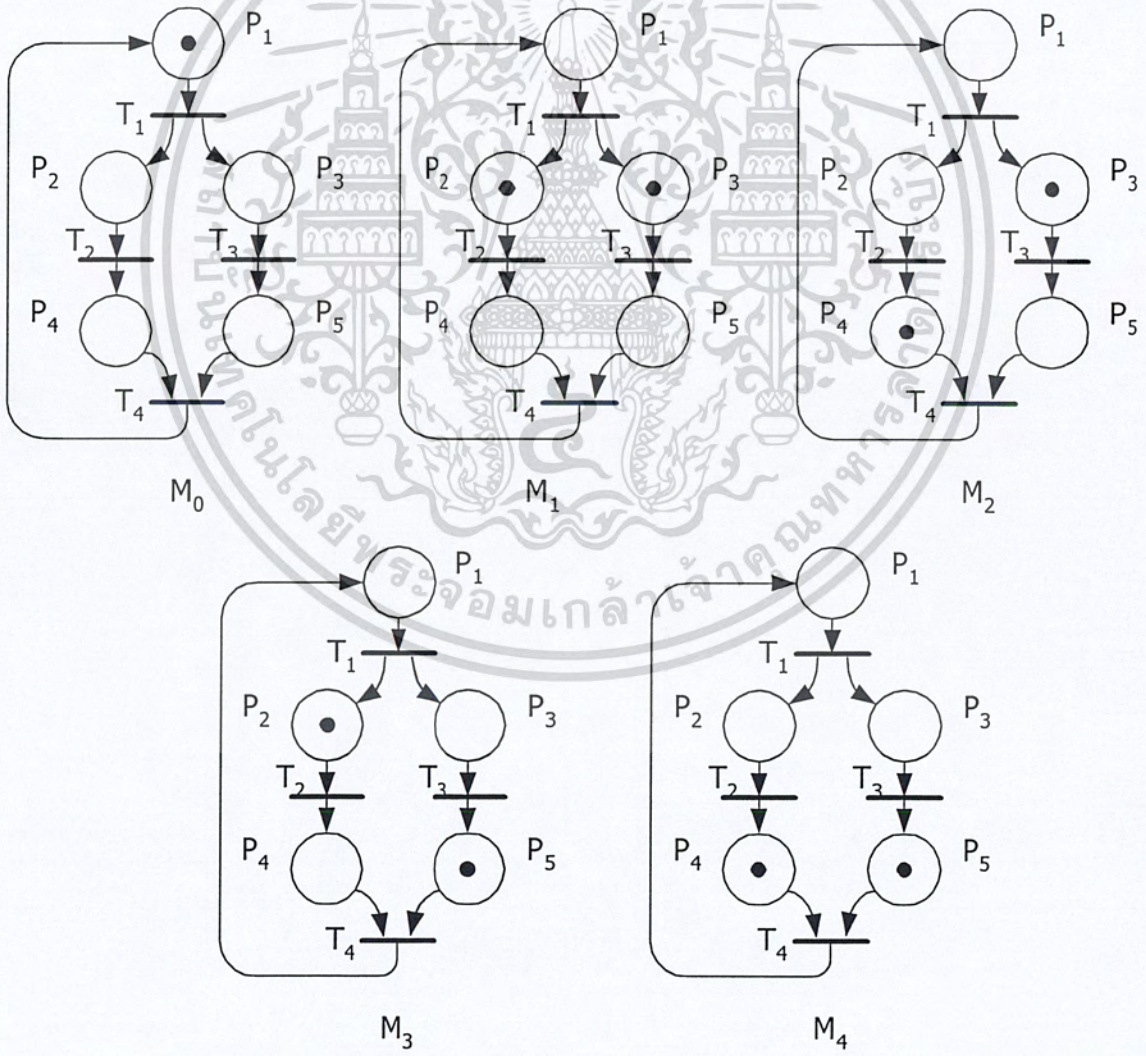
ในการที่เราใช้สัญลักษณ์และคำจำกัดความต่าง ๆ แทนนั้นจะทำให้ง่ายในการที่เราจะเขียนและอธิบาย Petri net

ในรูปที่ 2.5 เป็นการแสดงการ Mark ของ Petri net โดยกำหนดให้เขียนอยู่ในรูปของ Column Vector ซึ่งเราสามารถให้มันง่ายขึ้นโดยการ Transposed เราจะใช้สัญลักษณ์ [ ] แทนเมตริกซ์ และ ( ) แทน transposed

$$M_0 = (1,0,0,0,0) = [\,1\,0\,0\,0\,0\,]^T = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$



รูปที่ 2.5 แสดงสถานะเริ่มต้นของ Petri net



รูปที่ 2.6 แสดงการ Reachable marking

ที่ตำแหน่ง Mo จะสามารถ enable ที่ T_1 เพียงแห่งเดียวและถ้าเรา firing ก็จะได้ M_1 = (0,1,1,0,0) และจะใช้สัญลักษณ์เป็น

Mo [$T_1 > M_1$

ที่ตำแหน่ง M1 จะมี สอง Transition ที่ enable อยู่คือ T_2 และ T_3 ถ้าเรา firing แล้วก็จะได้ M_2 และ M_3 ซึ่งก็จะได้ (ในรูปที่ 2.6)

M_1 [$T_2 > M_2$ = (0,0,1,1,0)

M_1 [$T_2 > M_3$ = (0,0,0,1,1)

ที่ตำแหน่ง M_2 มี T_3 enable เพียง Transition เดียว และถ้า firing แล้วก็จะได้ M_2 [$T_3 > M_4$ = (0,0,0,1,1)

สำหรับที่ M_3 T_2 enable ตำแหน่งเดียว ถ้า firing ก็จะได้ M_4 ซึ่งเมื่อยู่ในตำแหน่ง M_4 แล้วนั้น T_4 ก็ enable เพียง Transition เดียวและเมื่อ firing T_4 แล้วนั้นก็จะได้ผลลัพท์เป็น Mo อีกครั้ง

ดังเป็นเหตุการณ์เช่นนี้แล้วสามารถเขียนได้ว่า *Mo = {Mo,M_1,M_2,M_3,M_4} และสามารถแสดงได้ดังรูปที่ 2.6

เมื่อเราเริ่มต้นที่ Mo ในขั้นแรก T_1 enable ต่อมา T_2 enable หลังจากการ firing ก็จะได้ M2 เราสามารถเขียนได้อีกแบบหนึ่ง คือ

Mo [$T_1T_2 > M_2$

T1T2 เป็นลำดับการ firing ซึ่งเขียนได้คือ

$S = T_1T_2$ และ Mo[$S>M_2$

2.1.6 คุณสมบัติที่สำคัญ (The essential characteristics)ของเพรททริเนทส์

Petri netเป็นสิ่งที่นำไปใช้ในการวิเคราะห์การทำงานของระบบ โดยเราจะต้องเรียนรู้ถึงคุณสมบัติของมัน ซึ่งในที่นี้จะได้กล่าวถึงเฉพาะคุณสมบัติที่เกี่ยวข้องกับการทำโครงงานนี้เท่านั้น

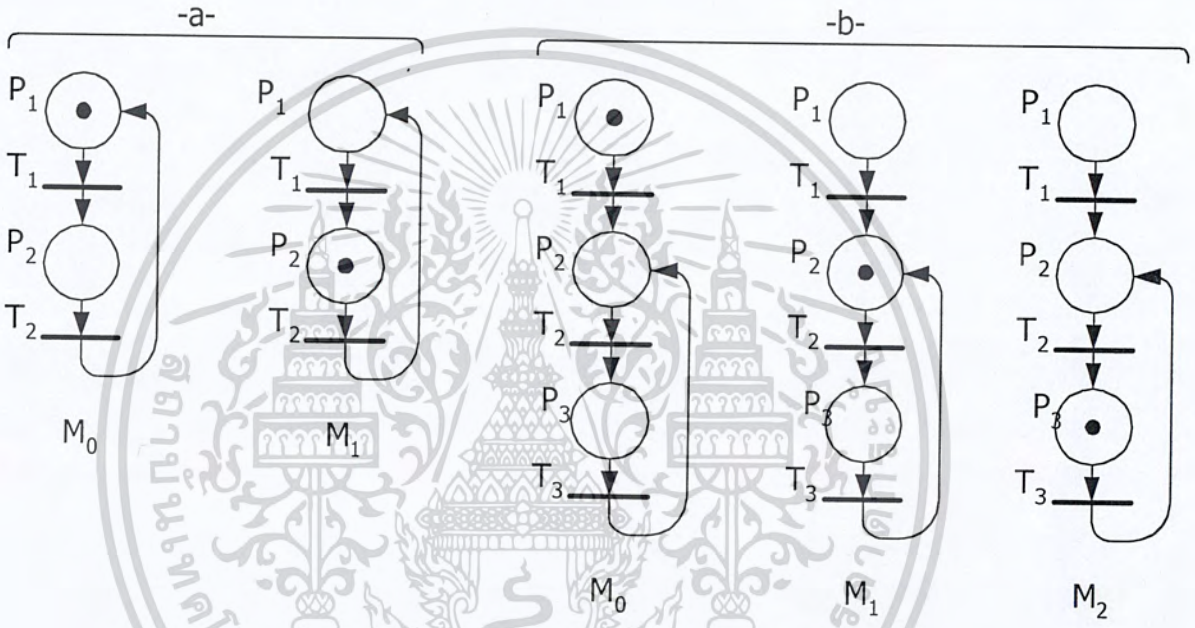
การคงสภาวะทำงานและอยู่นิ่งของระบบ (Livens and deadlock)

การเกิด marking ที่ค่อย ๆ เกิดขึ้นโดยการ fireable ของ Transitions เมื่อใน Petri netนั้น ไม่สามารถ fireable ต่อไปได้อีกในทุก ๆ ทางของ Petri netแล้วจะทำให้ระบบที่ทำการออกแบบมีความน่าจะเป็นสูง และค่อนข้างจะแน่นอนที่จะเกิดปัญหา

Transition Tj จะคงสภาวะทำงาน (live) สำหรับสถานะเริ่มต้นของ Marking Mo ถ้า MiE*Mo จะต้องผ่าน Tj ทุกครั้งและเหมือนเดิมตลอด Transition live นี้สามารถอธิบายดังรูปที่ 2.7 จากรูป 2.16 a จะเห็นว่าในลำดับการเข้าหา Mo จะประกอบด้วยการ firing อยู่ 2 ลำดับคือ T1,T2, Mo[T_1>M1 และ M_1[$T_2T_1>M_1$ และจะเป็นอย่างนี้เรื่อย ๆ เมื่อมีการ firing เข้าหา Mo จะเรียก T1,T2 ว่า live ส่วนในรูปที่ 2.7b จะเห็นได้ว่าจะไม่สามารถเข้าหา Mo ได้ ซึ่งเมื่อทำการ firing T1 ไปแล้ว

T_1 ไม่สามารถ enabled ได้อีก ดังนั้น T_1 จึงไม่ใช่ live Transition และ Petri netจะ Live ได้เมื่อทุก Transition ของ Petri netนั้น live

Transition เป็น quasi-live ถ้า Transition นั้นมีโอกาสที่จะ fired ดังในรูปที่ 2.7b ที่ T1 ซึ่งสามารถ fired หนึ่งครั้งก่อนที่จะไม่สามารถ enable ได้ตลอดไป ในรูปที่ 2.8a แสดงให้เห็นถึง quasi-live Petri net ในการ firing จากสถานะของการทำงานเริ่มต้น Mo. $Mo[T_1>(0,1,0,0,)$ และ $Mo[T_2T_3T_4>(0,0,1,0)$ ซึ่งเป็นการแย่งกัน firing ระหว่าง T_1,T_2 ($<P_1\{T_1,T_2\}$ ถ้า Transition ที่ firing ก่อนเป็น T_1 ก็จะไม่สามารถ firing Transition อื่นได้อีก

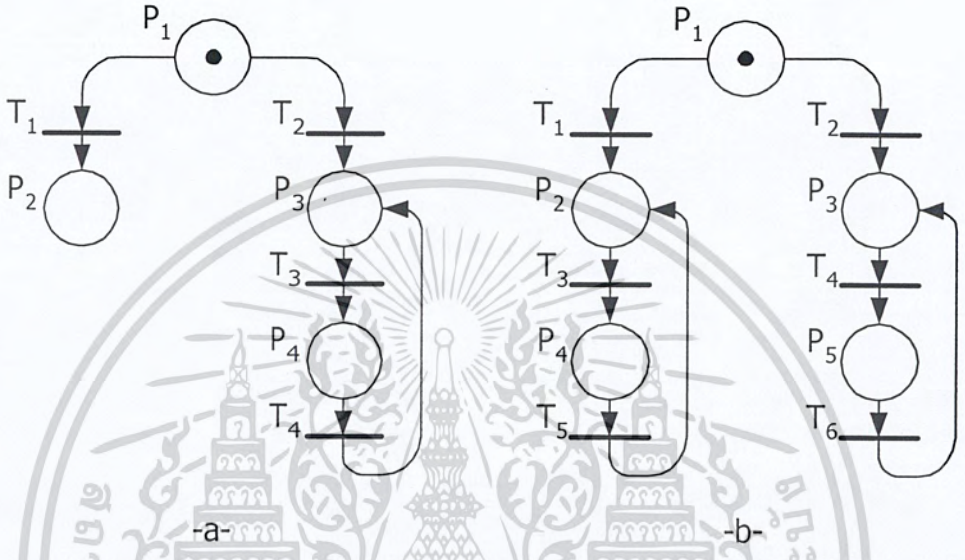


รูปที่ 2.7 แสดงการ Livens

Deadlock (หรือ sink state) เป็นปรากฏการณ์ที่ไม่สามารถที่จะ firing ต่อไปได้อีก ซึ่งเกือบจะถูกรูปแบบทุกระบบเมื่อเกิดสภาวะ deadlock และจะเป็นสิ่งที่แย่มากซึ่งก็จะกล่าวได้คือในระบบส่วนมากไม่ต้องการให้เกิด deadlock หรือกล่าวอีกนัยหนึ่ง deadlock คือ marking ที่ไม่สามารถจะ enabled ได้อีก จากในรูปที่ 2.8a เมื่อ firing T1 แล้วจะได้ผลลัพธ์ คือ M1 = (0,1,0,0) ซึ่ง Marking นี้คือ deadlock และPetri net จะสามารถพูดได้ว่าเป็น deadlock-free สำหรับสถานะเริ่มต้น Mo ถ้าเมื่อ firing ไปแล้วไม่สามารถเข้าหา Mo ได้โดยจะวนอยู่ที่เดิม

ในรูปที่ 2.8b เป็น deadlock free Petri net ถ้า firing T1 จะได้ Marking M1= (0,1,0,0,0) และ T3,T5 จะเป็น live Transition จากการ Marking นี้และเมื่อ firing T2 ก็จะได้ลักษณะเดียวกัน ซึ่งมันเรียกว่า deadlock free Petri nets

หมายเหตุ การเกิด livens และ deadlock นั้นจะขึ้นอยู่กับสถานะเริ่มต้นของการ Marking ด้วยเช่นถ้าสถานะเริ่มต้นของการ Marking ในรูปที่ 2.7a เป็นศูนย์ Mo = (0,0) นั้นจะเป็น deadlock และไม่มี Transition ใด enable เลย จากรูปที่ 2.8a จะเห็นได้ว่าเป็น quasi-live และจะเป็น deadlock เมื่อ Mo =(1,0,0,0) แต่ถ้า Mo = (0,0,1,0) จะไม่สามารถเรียกว่า live Petri net ได้ มันจะเป็น deadlock free แทน

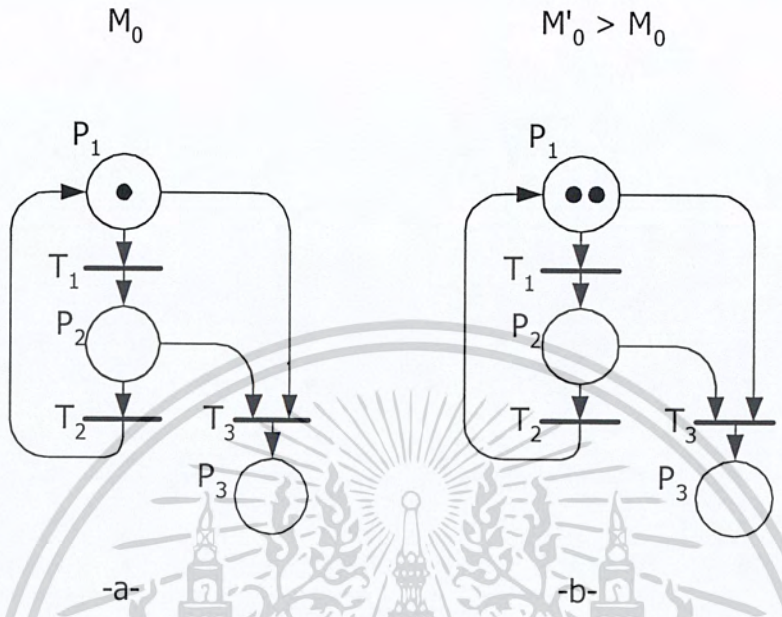


รูปที่ 2.8 Quasi live Petri nets. (a) With deadlock. (b) Deadlock-free

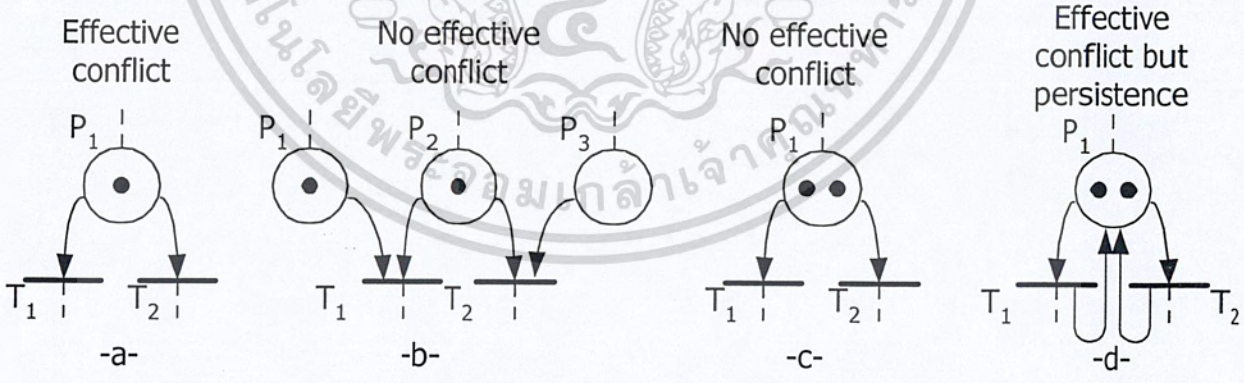
การต่อสู้ของเพททริเนทส์ (conflicts Petri nets)

Conflict ในโครงสร้างของระบบ conflict ก็คือเมื่อ Place หนึ่งเป็น input ของ Transition จำนวน 2 Transition หรือมากกว่านั้น ดังแสดงในรูปที่ 2.1 ที่ Place P3 เป็น input ของ Transition T3 และ T6 ซึ่งเมื่อ T3 และ T6 เกิดการ enable พร้อมกันนั้น จะทำให้ต้องตัดสินใจว่าจะ firing ในทางใดซึ่งสามารถเขียนได้เป็น $K = <P_1,\{T_1,T_2...\}>$

ผลของการแข่งขันกัน firing ของ Transition จะเป็นการคงอยู่ของโครงสร้าง(K) และการ Marking(M) เช่น จำนวนของ Tokens ใน Pi น้อยกว่า จำนวน output Transition ของ Pi ซึ่งจะ enabled โดย M



รูปที่ 2.9 (a) T1 และ T2 เป็น Live T3 ไม่เป็น quasi-live.
(b) T1 และ T2 ไม่เป็น live , T3 เป็น quasi-live.



รูปที่ 2.10 Conflict

ในรูปที่ 2.10a เรามีโครงสร้างการแก่งแย่ง K_1=<P1,{T_1,T_2}> Transition T_1 และ T_2 enable แต่ใน place มี Token เพียงตัวเดียว ซึ่งจะทำให้เกิดผลของการแย่งกัน firing ระหว่าง T_1 กับ T_2 คือ Ke = <P_1,{T_1,T_2},M>

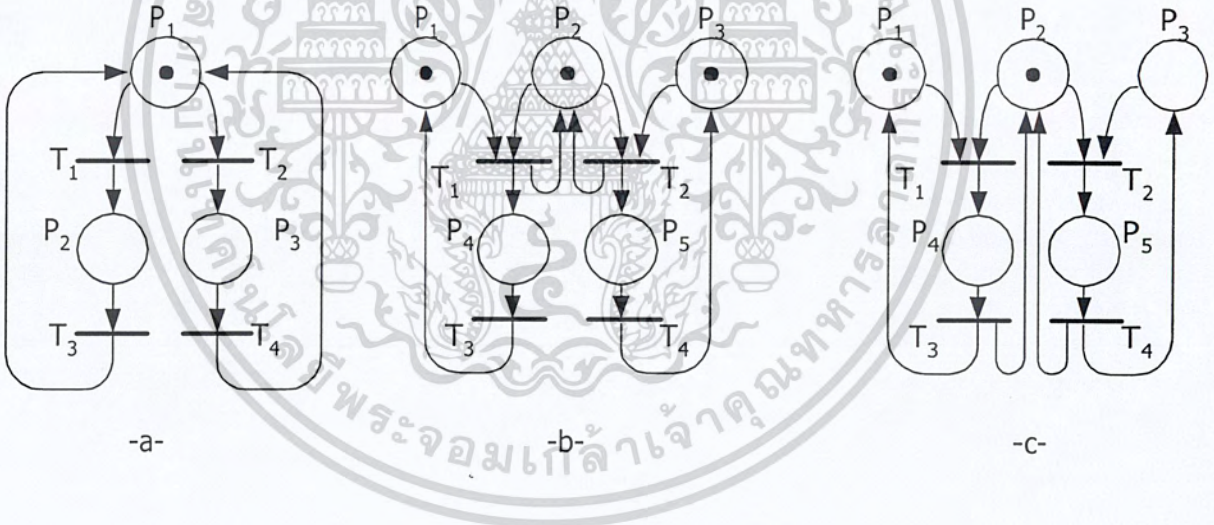
ในรูปที่ 2.10b จะมี T_1 enable เพียงตัวเดียว จึงไม่มีการแก่งแย่งกัน firing

ในรูปที่ 2.10c Transition T_1 และ T_2 enable ทั้งคู่ แต่ใน place มีจำนวน Token อยู่ 2 Token ดังนั้น Transition ทั้งสองจึงสามารถ firing พร้อมกันได้

ในรูปที่ 2.10d ถึงแม้ว่ามันจะเป็นการแย่งกันfiring ระหว่าง T_1 และ T_2 แต่จะมีโครงสร้างเฉพาะ ตามรูป จะไม่สามารถ firing พร้อมกันได้แต่เมื่อ firing ไปแล้ว จะมี Marking กลับมาดังเดิม จึงทำให้ firing ได้อีกซึ่งเหตุการณ์แบบนี้จะเรียกว่า persistent Petri net

ถ้าทุก Marking เป็นส่วนหนึ่งของการเข้าหา Mo(MiE*Mo) Petri nets นั้นจะปราศจากการแก่งแย่งเพื่อ firing Transition

หมายเหตุ Multiple firing คือการที่ Petri net สามารถ firing ไปพร้อมกันได้ และมีค่าเท่ากัน เช่น ในรูปที่ 2.6 ที่ M_1 จะเป็น Multiple firing ซึ่งจะลำดับการ firing เป็น $[T_2T_3]$ ซึ่งจะได้เป็น M_4 หรืออาจจะเขียนได้เป็น $M_1[[T_2T_3]> M4$



รูปที่ 2.11 Effective conflict and persistence for a Petri nets

2.1.7 กราฟของเครื่องหมาย และ กราฟแบบต้นไม้

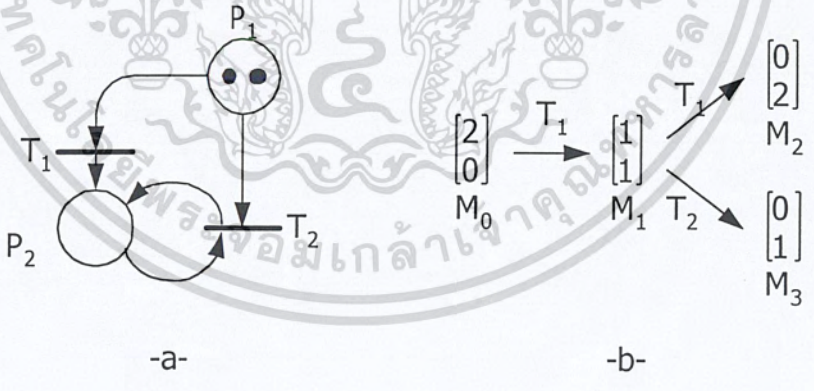
กราฟของเครื่องหมาย (Graph of markings)

Graph of markings สร้างขึ้นมาให้สอดคล้องกับการ firing ของ Transition ต่าง ๆ และเป็นเส้นทางการ firing ซึ่งจะเขียนขึ้นมาจากจำนวน Tokens ที่อยู่ใน Place ของแต่ละขั้นในการ firing

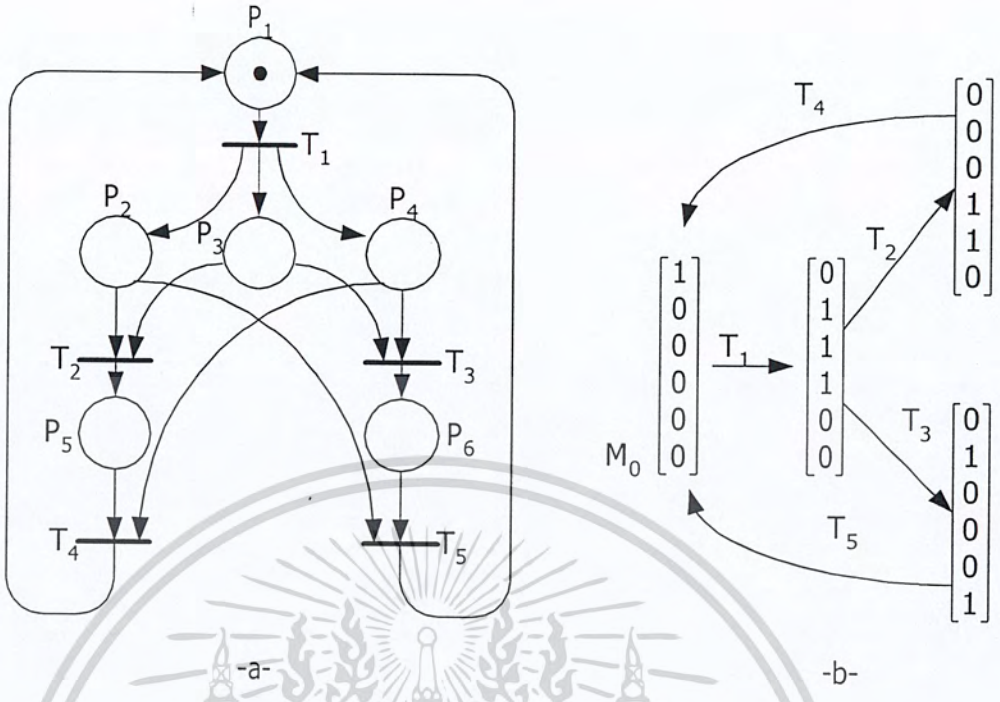
ในรูปที่ 2.12a คือ สถานะการทำงานของ Mo = (2,0) และในรูป 2.21b ก็คือ graph of marking ของรูปที่ 2.21a ซึ่งจะเห็นได้ว่าจาก Mo T_1 พร้อมที่จะทำงานถ้า firing ก็จะได้ M_1 เท่ากับ $M_1[T_1>M_2$, $M_1[T_2>M_3$ ซึ่งจะเห็นได้ว่าในรูปที่ 2.12a และ b มีความสัมพันธ์ในการ firing ของ Transition แต่ละ Transition ตรงกัน

Graph of markings สามารถนำไปใช้ประโยชน์ในการหาคุณสมบัติของ Petri net อย่างเช่น เราจะเห็นได้ว่า จากรูปที่ 2.12b เราจะรู้ว่ามันเกิด deadlock ขึ้นที่ M_2 และ M_3 ซึ่งมันมีขอบเขตไม่ใช่ live Petri nets

ในรูปที่ 2.13 เป็นการใช้ graph of marking หาคุณสมบัติของ Petri net จะสังเกตเห็นได้ว่า Petri net เป็น safe, live,reversible และ T_1 T_2 T_4 T_1 T_3 T_5 เป็นลำดับการ firing ที่กลับคืนได้



รูปที่ 2.12 ตัวอย่าง กราฟของ Marking

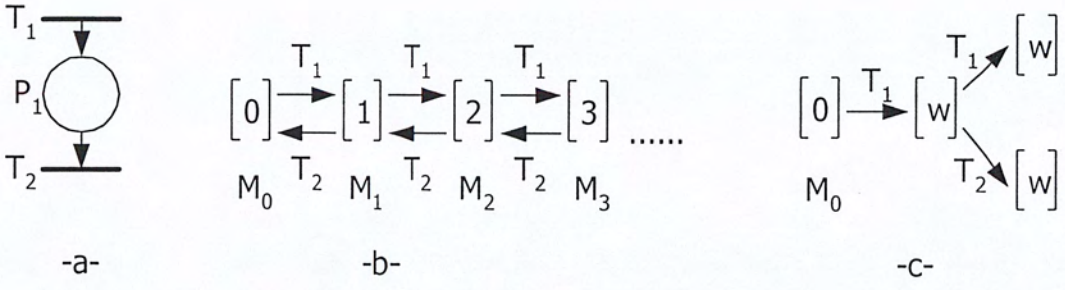


รูปที่ 2.13 ตัวอย่าง กราฟของ Marking

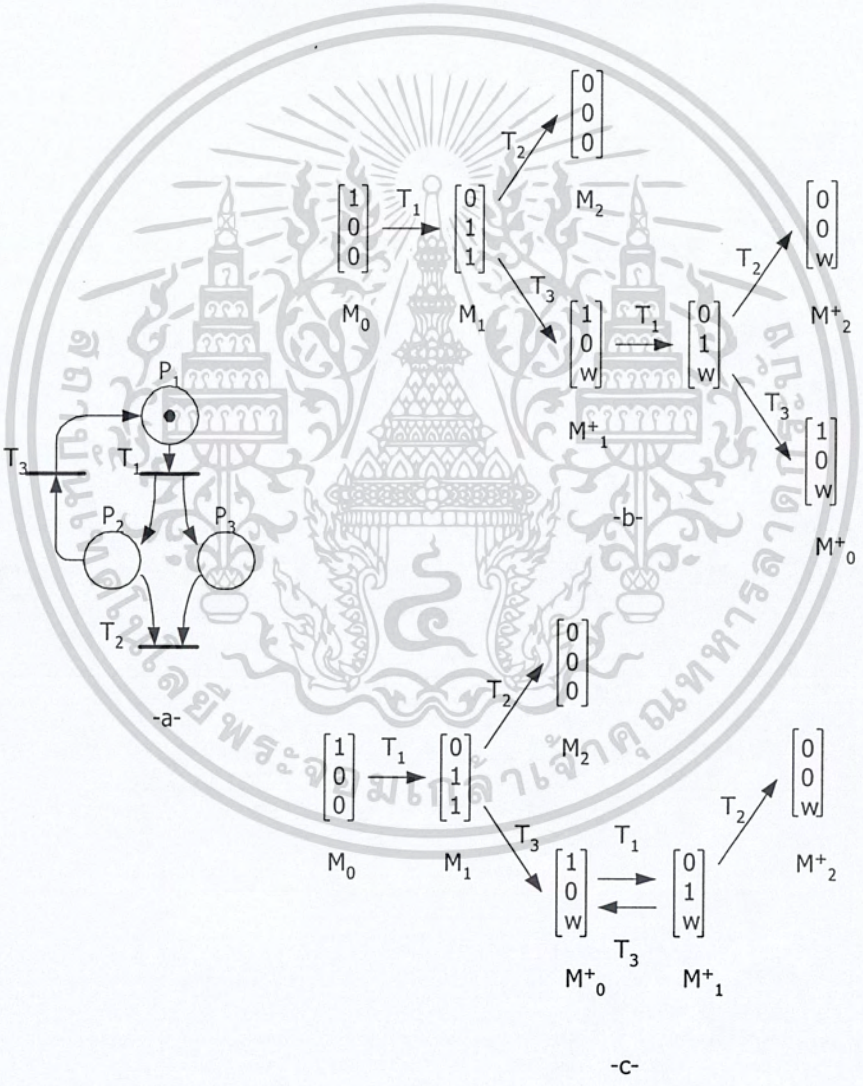
กราฟแบบต้นไม้ (Coverability graph and tree)

Coverability graph and tree เขียนขึ้นมาเพื่อรองรับการ firing ที่สร้าง Tokens ใน place ขึ้น ซึ่งมีจำนวนไม่จำกัด และมีลักษณะการ firing ซ้ำ ๆ กัน ดังเช่นในรูปที่ 2.14a ถ้า Transition T1 enabled และ fired ออกไปก็จะมีหนึ่ง Token ที่ P1 และ Transition T1 และ T2 ก็ยังคง enable อยู่ ถ้า T1 tired ก็จะมี Token ในP1 เป็นสอง Token แต่ถ้า T2 fired ก็จะทำให้ Tokens ลดลงและยังคงเป็นอย่างนี้ไปเรื่อย ๆ ในรูปที่ 2.14b เป็นการสร้าง graph of marking อย่างไรก็ตามแล้ว graph of marking นี่ก็ไม่จะสามารถที่จะแสดงความเป็นไปได้ของ marking ทั้งหมดได้เพราะการเข้าถึง Mo นั้นเป็นอนันต์ เพราะฉะนั้นเราจึงได้สร้างกราฟที่มีความสามารถที่จะแสดง marking ได้ครอบคลุมทั้งหมด

ในรูปที่ 2.14c ได้แสดง coverability tree(coverability graph and tree) ของรูป 2.23a โดยเริ่มที่ Mo = (O),Mo[T_1>M_1=(1) ซึ่ง M_1 covers Mo (M_1>Mo) และ T_1 สามารถกลับมา enable ได้อีก จำนวน Tokens ใน T1 สามารถเข้าถึง +k ซึ่งเราจะเรียกมันว่า(w) ซึ่งจะแสดงลักษณะของ marking ที่เป็นอนันต์ ซึ่งในรูป 1.26 จะมี Tokens ใน P_1 เป็น K+1 เมื่อ T_1 fired และ k-1 เมื่อ T_2 fired ซึ่งเราจะใช้ w นี้ เป็นตัวแทนสำหรับ place ที่มี Tokens มากขึ้นเป็นอนันต์ (แต่จะต้องไม่น้อยไปกว่าศูนย์)



รูปที่ 2.14 ตัวอย่างของ coverability tree



รูปที่ 2.15 coverability tree and coverability graph

Coverability tree จะมี Algorithm ในการสร้างดังนี้

ขั้นตอนที่ 1. จากสถานะเริ่มต้น Mo เขียน Mi (Marking ต่าง ๆ) ที่เกิดจากการ fired ของ Transition ที่ enable แต่ถ้ามีส่วนประกอบใดที่มากกว่าและสอดคล้องกับส่วนประกอบใน Mo ให้แทนด้วย w

ขั้นตอนที่ 2. สำหรับ M1(Marking ที่เกิดขึ้นใหม่) ซึ่งจะแยกได้ออกเป็นสองแบบคือ ขั้นตอน 2.1 หรือ ขั้นตอน 2.2

ขั้นตอน 2.1 ถ้า M ใด ๆ ที่เกิดขึ้นใหม่ มีค่าเท่ากับ M ที่ผ่านมา ก็แสดงว่าจะไม่มี Marking ต่อไปจะหยุดแค่นั้นและนำมาเชื่อมต่อเข้าด้วยกัน

ขั้นตอน 2.2 ถ้า M ที่เกิดขึ้นใหม่ไม่เท่ากับ M เดิม ก็จะสร้างตัวรับหรือจะพูดได้ว่าสร้าง Marking ใหม่ขึ้นมา และเขียนใหม่เช่นเดียวกันกับ step1 ถ้ามีส่วนประกอบใน M ใหม่ใด ที่มากกว่าและสอดคล้องกับส่วนประกอบใน M เดิมก็ให้แทนด้วย w

ตัวอย่าง (ในรูปที่ 2.15a และ b)

Step 1. $Mo[T_1>M_1$

Step 2.2 สำหรับ M_1

$M1[T_2>M_2$

$M1[T_3>(1,0,1)$ ซึ่ง $(1,0,1) > (1,0,0) = Mo$ เราจะเขียน

$M1[T_3>(1,0,w)\ M^+0$

Step 2.2 สำหรับ M_2 ไม่มี Transition ที่ enable ซึ่งมันคือ deadlock

Step 2.2 สำหรับ M^+0

$M^+1[T_2>(0,0,w) = M^+_2$

$M^+1[T_3>(1,0,w) = M^+o$

Step 2.2 สำหรับ M^+_2 ไม่มี Transition ที่ enable ซึ่งมันคือ deadlock

Step 2.1 สำหรับ M^+o เจอ M^+o ซึ่งได้รับการ fired จาก T_3 ซึ่งมีค่าเหมือนกันและเป็นการ fired ที่ซ้ำกัน $T_1\ T_3\ T_1\ T_3$ จาก Mo

2.2 ทฤษฎีของ Grafcet

ทฤษฎีของ Grafcet เป็นทฤษฎีที่ประยุกต์มาจากทฤษฎี Petri netซึ่งจะใช้ในการออกแบบควบคุมทางลอจิก ซึ่งมีค่าที่เป็นไปได้แค่เพียง “0” กับ “1” และจะดำเนินการด้านตรรกะ คือ การดำเนินการกับข้อมูลแล้วให้ค่าผลลัพธ์เป็น จริง หรือ ไม่จริง เท่านั้น โดยที่ “0” แทนเหตุการณ์ที่ไม่เป็นจริง หรือไม่ทำงาน ส่วน “1” จะแทนเหตุการณ์ที่เป็นจริงหรือทำงาน

การดำเนินการทางตรรกะ เช่น Not, And, Or, Xor

ตารางที่ 2.1 การดำเนินการทางตรรกะ Not

Input	Output
0	1
1	0

ตารางที่ 2.2 การดำเนินการทางตรรกะ And

Input	Input 2	Output
0	0	0
0	1	0
1	0	0
1	1	1

ตารางที่ 2.3 การดำเนินการทางตรรกะ Xor

Input 1	Input 2	Output
0	0	0
0	1	1
1	0	1
1	1	0

ตารางที่ 2.4 การดำเนินการทางตรรกะ Or

Input 1	Input 2	Output
0	0	0
0	1	1
1	0	1
1	1	1

2.2.1 สัญลักษณ์ที่ใช้

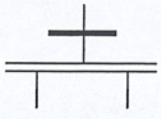
Place (Step) เริ่มต้น คือสถานะที่ใช้ในการเริ่มทำงานจะมี Logic = "1" เมื่อเริ่มทำงาน

Place (Step) คือ ขั้นการทำงานของโปรแกรม ซึ่งจะแสดงการทำงานของเครื่องจักรกลว่าตอนนี้จะให้มันทำงานอะไร

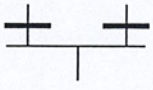
Transition มีลักษณะการทำงานเหมือนหน้าสัมผัส หรือสวิตช์ของวงจรไฟฟ้า

Arc คือ ใช้สำหรับต่อเชื่อมระหว่าง Place และ Transition

junction AND



Distribution AND



Junction OR

Distribution OR

2.2.2 เงื่อนไขของการทำงาน

1. Transition จะส่งผ่านสัญญาณออกไปเมื่อ Transition นั้น มีสถานะที่พร้อมที่จะทำงาน คือ มีสถานะเท่ากับ "1" และ Place ก่อนหน้าก็มีค่าเท่ากับ "1"
2. เมื่อ Transition หลาย ๆ ตัวมีสถานะพร้อมที่จะทำงานพร้อมกันมันก็จะทำงานไปพร้อมกัน
3. เมื่อ Transition ถัดไปจาก Step ไม่พร้อมที่จะทำงาน Step นั้นก็จะยังคงสภาพทำงานต่อไป

2.3 การติดต่อสื่อสาร

การติดต่อสื่อสารแบบ Host link เป็นการเชื่อมต่อระหว่าง PLC กับคอมพิวเตอร์ทั่วไปผ่านพอร์ตการสื่อสารของคอมพิวเตอร์ที่เรียกว่า COM 1: หรือ COM 2: ซึ่งเป็นพอร์ตการสื่อสารแบบอนุกรม เพื่อให้สามารถควบคุม PLC จากคอมพิวเตอร์ได้ สำหรับคอมพิวเตอร์ 1 เครื่อง สามารถต่อเข้ากับ PLC ได้จำนวนมาก โดยใช้การเชื่อมต่อ PLC หลาย ๆ ตัวเข้าด้วยกัน ซึ่งเรียกว่า PC Link ในการติดต่อแบบ Host link จะต้องผ่านอุปกรณ์ที่เรียกว่า Host link Unit

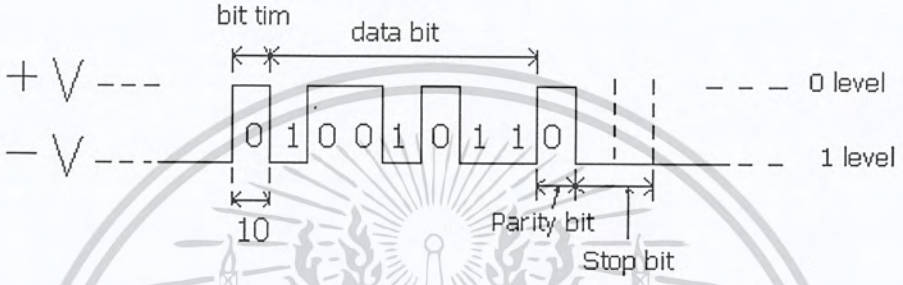
2.3.1 พอร์ตการสื่อสารแบบอนุกรม

ช่องการสื่อสารอนุกรมนี้สามารถเรียกได้หลายอย่าง เช่น Serial Port, RS-232, comport เครื่องคอมพิวเตอร์โดยปกติจะมีพอร์ตชนิดนี้อยู่แล้ว 2 พอร์ต คือ พอร์ตขนาด 9 ขา มีรูปร่างเหมือนสี่เหลี่ยมคางหมูมีเข็มยื่นออกมา 9 เข็ม เรียกหัวชนิดนี้ว่า "DB – 9 Connector Male Type" และอีกพอร์ตหนึ่ง คือพอร์ตขนาด 25 ขา มีลักษณะเหมือนกับแบบ 9 ขา ต่างกันตรงมีขามากกว่าเท่านั้น และเรียกว่า "DB – 25"

รูปที่ 2.16 ตำแหน่งของ DB – 9

ลักษณะที่สำคัญของพอร์ตอนุกรม คือ ข้อมูลจะส่งในลักษณะอนุกรม กล่าวคือ แทนที่จะต้องมีสายสัญญาณ 8 เส้น สำหรับการส่ง 1 ไบต์ ระบบนี้อาจใช้เพียง 2 เส้น หรือมากกว่า (อย่างน้อยต้องมีสายสัญญาณ 1 เส้น และ Ground 1 เส้น) โดยจะให้สัญญาณผ่านทีละบทเรียงต่อกันไปจนครบไบต์ ซึ่งลักษณะเช่นนี้มีข้อดี คือ จำนวนสายสัญญาณจะลดลงไปมาก แม้ความเร็วในการส่งจะน้อยลงก็ตาม โดยสัญญาณที่ส่งไปนี้จะส่งไปในลักษณะเป็น Asynchronous Serial Data Format

กล่าวคือ ข้อมูลที่ส่งจะประกอบด้วยสัญญาณ 2 ระดับ มาเรียงต่อกัน (เช่น + 5 V สำหรับระดับ “0” และ – 5 V สำหรับระดับ “1”) โดยก่อนจะเริ่มส่งข้อมูลระดับสัญญาณจะอยู่ที่ระดับ “1” จากนั้นก็จะมีสัญญาณเริ่มส่ง (Start Bit) ซึ่งเป็นสัญญาณระดับต่ำ “0” ส่งไปเพื่อบอกว่ากำลังจะส่งข้อมูลตามมา จาก Start Bit จะตามด้วยสัญญาณข้อมูลขนาด 8 บิท เรียงต่อกันไป จากนั้นก็จะมีสัญญาณบอกว่าจบการส่งข้อมูล (เรียกว่า Stop Bit) ซึ่งเป็นสัญญาณระดับสูง (“1”) เมื่อหมดชุดข้อมูลก็ปรับให้สัญญาณมาอยู่ที่ระดับ “1” เพื่อคอยรับ Start Bit ต่อไป



รูปที่ 2.17 ลักษณะสัญญาณขณะส่งผ่านพอร์ทอนุกรม

การส่งสัญญาณลักษณะที่ค่อนข้างช้า เพราะจำนวนบิทเรียงต่อกันอย่างอนุกรม โดยอัตราความเร็วของการส่ง เรียกว่า Boad Rate: จำนวนบิท/วินาที จะมีค่าอยู่ในช่วง 1200, 2400, 9600 จะเห็นว่าการส่งสัญญาณลักษณะนี้ 1 ไบต์ จะต้องใช้จำนวนบิทถึง 10 บิท (Start Bit + Data Bit + Stop Bit) เรียงต่อกันแบบอนุกรมกันไป จึงเป็นเหตุให้การส่งข้อมูลช้า

ในการส่งข้อมูลผ่านพอร์ตอนุกรมจะต้องคำนึงถึงสิ่งต่าง ๆ เช่น

- ต้องมีระดับสัญญาณตั้งแต่ +3 ถึง +25 โวลท์ (สำหรับการส่งสัญญาณโดยคอมพิวเตอร์ไม่มีปัญหา แต่ในกรณีรับสัญญาณเข้าอาจต้องมีระบบป้องกัน เพราะหากสัญญาณเกิน +25 โวลท์ อาจต่อให้เกิดความเสียหายได้
- ความเร็วในการส่ง (Baud Rate) ถ้าเป็นสายสัญญาณธรรมดาอาจส่งได้เร็วถึง 9600 บิท/วินาที แก่กรณีใช้ผ่านระบบโทรศัพท์ความเร็วจะลดลง
- จำนวนบิทสำหรับข้อมูลว่าใช้ระบบใด เช่น 5, 6, 7 หรือ 8 บิท/ตัวอักษร
- จำนวน Stop Bit มีขนาดเท่าใด เช่น 1 Stop Bit หรือ 2 Stop Bit
- จะต้องมี Parity Check หรือไม่ ฯลฯ

ตารางที่ 2.5 ตารางหน้าที่ของแต่ละขาของพอร์ตอนุกรม DB-9

ขา	หน้าที่
1	Data Carrier Detect
2	Receive Data
3	Transmit Data
4	Data Terminal Ready
5	Signal Ground
6	Data Set Ready
7	Request to Send
8	Clear to Send
9	Ring Indicator

ตารางที่ 2.6 ตารางหน้าที่ของแต่ละขาของพอร์ตอนุกรม แบบ DB-25

ขา	หน้าที่
1	ไม่ได้ใช้
2	Transmit Data
3	Receive Data
4	Request to Send
5	Clear to Send
6	Data Set Ready
7	Signal Ground
8	Data Carrier Detect
20	Data Terminal Ready
22	Ring Indicator
9-19, 21< 23-25	ไม่ได้ใช้

- ขา Data Carrier Detect : DCD หรืออาจเรียกว่า Carrier Detect : CD ขาที่จะแอกทีฟเมื่อมีการส่งสัญญาณพาหะจากอุปกรณ์สื่อสารข้อมูล เช่น โมเด็ม สำหรับการใช้งานปกติขานี้จะไม่ถูกใช้งานมากนัก
- ขา Receiver Data : หรือ RXD ขานี้เพื่อรับสัญญาณอนุกรมเข้ามายังคอมพิวเตอร์โดยนำข้อมูลที่อ่านได้ไปเก็บไว้ในรีจีสเตอร์บัฟเฟอร์
- ขา Transmitter Data หรือ TXD ขานี้ใช้เพื่อส่งข้อมูลอนุกรมออกมาจากคอมพิวเตอร์โดยในการนำข้อมูลที่เก็บอยู่ในบัฟเฟอร์ส่งข้อมูลออกไป
- ขา Data Terminal Ready : DTR เป็นขาเอาท์พุตที่ใช้ในการส่งสัญญาณข้อมูลออกจากคอมพิวเตอร์เพื่อใช้อุปกรณ์ปลายทางรับรู้ว่าต้องการติดต่อกับอุปกรณ์ปลายทางโดยที่ขา DTR นี้จะต้องเชื่อมต่อกับขา DSR ของอุปกรณ์ปลายทางและขา DTR ของอุปกรณ์ปลายทางจะต้องเชื่อมต่อกับขา DSR ของพอร์ตอนุกรมเข้าด้วยกัน และจะต้องเชื่อมต่อเข้ากับขา DCD ด้วยในกรณีที่โปรแกรมสื่อสารที่ใช้ในการตรวจสอบสัญญาณพาหะ
- ขา Signal Ground : GND เป็นขากราวด์ของสัญญาณ
- ขา Data Set Ready : DSR ขานี้ใช้คู่กับขา DTR เพื่อตรวจสอบการเชื่อมต่อระหว่างคอมพิวเตอร์กับอุปกรณ์ปลายทาง ซึ่งขา DSR นี้จะเป็นขาที่รับรู้ข้อมูลจากภายนอก
- Request to send : RTS เป็นขาเอาท์พุทสำหรับส่งสัญญาณร้องขอให้อุปกรณ์ปลายทางส่งข้อมูลมายังคอมพิวเตอร์ โดยขาที่รับสัญญาณขา RTS คือขา CTS ซึ่งในกรณีที่มีการเชื่อมต่อแบบ 3 สาย จะต้องเชื่อมต่อขา RTS และ CTS เข้าด้วยกันเพื่อให้การรับและส่งข้อมูลเกิดขึ้นได้ตลอดเวลา
- ขา Clear To Send : CTS เป็นขาอินพุททำหน้าที่รอรับสัญญาณที่ส่งเข้ามาเมื่อมีการส่งสัญญาณเข้ามา ขานี้ใช้ตรวจสอบอุปกรณ์ต่อพ่วงว่าพร้อมที่จะรับข้อมูลแล้วหรือยัง
- ขา Ring Tnd : cator : RT ใช้แสดงสถานะสัญญาณเรียกจากสายโทรศัพท์ปกติในการสื่อสารทั่วไป สายนี้จะไม่ถูกใช้งาน จะใช้งานก็ต่อเมื่อมีการเชื่อมต่อกับโมเด็มแล้วยังมีความต้องการตรวจสอบสัญญาณเรียกจากสายโทรศัพท์

3.3.2 การส่งสัญญาณเพื่อควบคุมการทำงานของ PLC

ในการส่งสัญญาณเพื่อควบคุม PLC จะส่งสัญญาณผ่านพอร์ตอนุกรมหรือ RS-232 ซึ่งจะอาศัยรูปแบบข้อตกลงในการสื่อสาร (Protocol) ของ PLC โดยทั่วไปจะเป็นการตอบโต้กันระหว่างเครื่องควบคุมกับอุปกรณ์ภายนอกอุปกรณ์ภายนอกในที่นี้คือ คอมพิวเตอร์ ซึ่ง คอมพิวเตอร์จะเป็นตัวส่งคำสั่งไปยัง PLC ในลักษณะตามรูปแบบของ Protocol จากนั้น PLC จะส่งสัญญาณตอบสนองกลับมายังคอมพิวเตอร์ในลักษณะรูปแบบของ Protocol เช่นกัน

ในการติดต่อสื่อสารของ PLC ยี่ห้อ OMRON นี้ รูปแบบข้อตกลงในการสื่อสาร (Protocol) เรียกว่า Host link Protocol ซึ่งมีรูปแบบการสื่อสารดังนี้

รูปแบบของคำสั่ง (Command Format)

@ เป็นเครื่องหมายที่ต้องมีเสมอสำหรับเริ่มต้นคำสั่ง

Node no

จะใช้สำหรับกำหนดว่าจะติดต่อกับ PLC หมายเลขอะไร (Nod) ในกรณีที่ PLC ที่เชื่อมต่ออยู่มีมากกว่า 1 เครื่องขึ้นไป ซึ่งหมายเลขของ PLC นี้จะสามารถตั้งได้ที่ตัว PLC เอง

Header Code

เป็นรหัสของคำสั่งว่าเราต้องการที่จะทำอะไรกับ PLC และทำในพื้นที่ใด เช่น ต้องการที่จะอ่านข้อมูลจากพื้นที่รีเลย์ภายในของ PLC ก็จะใช้รหัสคำสั่ง เป็นตัวอักษรพิมพ์ใหญ่ว่า “RR” หรือถ้าต้องการเขียนข้อมูลลงไปในพื้นที่รีเลย์ภายในของ PLC ก็จะใช้รหัสคำสั่งเป็นตัวอักษรพิมพ์ใหญ่ว่า “WR” เป็นต้น ซึ่งจะสามารถศึกษาได้จากคู่มือของ PLC

Text

เป็นส่วนของข้อมูลหรือกลุ่มคำสั่งที่ต้องการสั่ง เช่น ถ้าเป็นคำสั่งในการอ่านก็จะระบุตำแหน่งเริ่มต้นในการอ่านและจำนวนที่ต้องการอ่าน หรือถ้าเป็นคำสั่งเขียนก็จะระบุตำแหน่งเริ่มต้นในการเขียนข้อมูลและข้อมูลที่ต้องการเขียนลงไป เป็นต้น

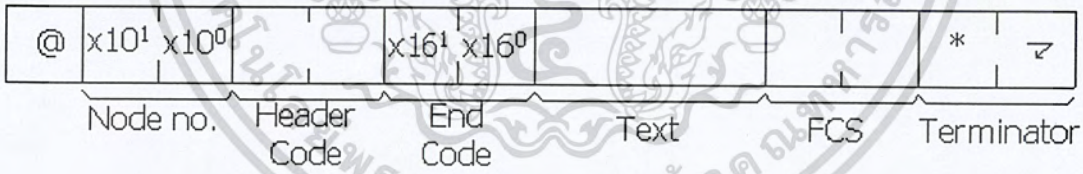
FCS (Frame Check Sequence Code)

เป็นรูปแบบของการตรวจสอบความผิดพลาดในการสื่อสารมี 2 byte โดยที่ FCS นี้จะได้จากการนำค่าเลขฐานสิบหกของรหัส ASCII แต่ละตัวแปลงเป็นรหัส BCD แล้วทำการทางตรรกะโดยการ Exclusive-Or (XOR) โดยเริ่มจากอักษร ๑ จนถึงตัวอักษรสุดท้ายของ Text หลังจากนั้นทำการแปลงกลับให้เป็นเลขฐานสิบหกอีกครั้งหนึ่งก็จะได้รหัสในการตรวจสอบข้อผิดพลาด ของ Host link Protocol ที่เรียกว่า FCS

Terminal

จะประกอบด้วย 2 ตัวอักษร คือ * ใช้สำหรับปิดท้ายคำสั่งเพื่อบอกว่าจบคำสั่งนี้แล้วแล้วก็จะตามด้วยรหัสของ Carriage return (CHR $ (13))

รูปแบบของการตอบสนอง (Response Format)



ในรูปแบบของการตอบสนองนี้จะมีส่วนที่แตกต่างจากรูปแบบคำสั่ง คือ

End Code

เป็นรหัสสถานะของการรับ-การส่งข้อมูล เช่น รหัส 00 เป็นรหัสที่บอกให้ทราบว่าการส่งข้อมูลเรียบร้อยดี หรือรหัส 14 คือรหัสบอกให้ทราบว่ารูปแบบของคำสั่งผิด เป็นต้น

Text

ในรูปแบบของการตอบสนองในส่วนนี้จะมีเฉพาะคำสั่งที่ใช้สำหรับอ่านเท่านั้น ซึ่งจะเป็นส่วนของข้อมูลที่อ่านได้

บทที่ 3
การใช้ Petri Nets ช่วยในการออกแบบวงจรแลดเดอร์ และการออกแบบโปรแกรม

เพื่อให้การตรวจสอบข้อบกพร่อง แก้ไขข้อผิดพลาด และเปลี่ยนแปลงเงื่อนไขการทำงานของวงจรแลดเดอร์ที่ใช้สำหรับควบคุมเครื่องจักรกลต่าง ๆ ง่ายขึ้นนั้น ควรที่จะมีรูปแบบในการเขียนวงจรแลดเดอร์ที่เป็นลำดับขั้นตอนไม่ซับซ้อน ไม่ใช่ว่านึกจะใส่อะไรก็ใส่เข้าไป จากที่ได้ศึกษาเกี่ยวกับทฤษฎีของ Petri Nets. และ Gratect ในบทที่ 2 ที่ผ่านมา ซึ่งจะได้นำความรู้เรื่องนี้มาประยุกต์ใช้ ช่วยในการออกแบบวงจรแลดเดอร์ แต่ในที่นี้ก็ไม่ใช่กฎเกณฑ์ตายตัวที่จะต้องเขียนวงจรแลดเดอร์ด้วยวิธีนี้เท่านั้น เพียงแต่วิธีนี้จะเป็นแนวทางในการออกแบบวงจรแลดเดอร์ที่ทำให้ผู้ออกแบบไม่สับสน และผู้ที่กลับมาแก้ไขก็จะสามารถศึกษาวงจรเก่าที่มีอยู่ได้ง่ายขึ้น และเพื่อให้เห็นผลการทำงานของวงจรที่ใช้ควบคุมเครื่องจักรกลได้ชัดเจนยิ่งขึ้น จึงได้เขียนโปรแกรมเพื่อใช้ในการตรวจสอบโปรแกรมที่ผู้ใช้งานเขียนขึ้นมาเพื่อใช้ในการควบคุมเครื่องจักร และจำลองสภาวะการทำงานต่าง ๆ ก่อนที่จะนำไปใช้งานจริง ซึ่งจะได้เขียนโปรแกรมอยู่บนพื้นฐานของทฤษฎีของ Petri Net. และ Grafcet และควบคุมอุปกรณ์ภายนอกผ่านหน่วยอินพุตและหน่วยเอาท์พุตของ PLC

3.1 การใช้ Petri Nets. ช่วยในการออกแบบวงจรแลดเดอร์

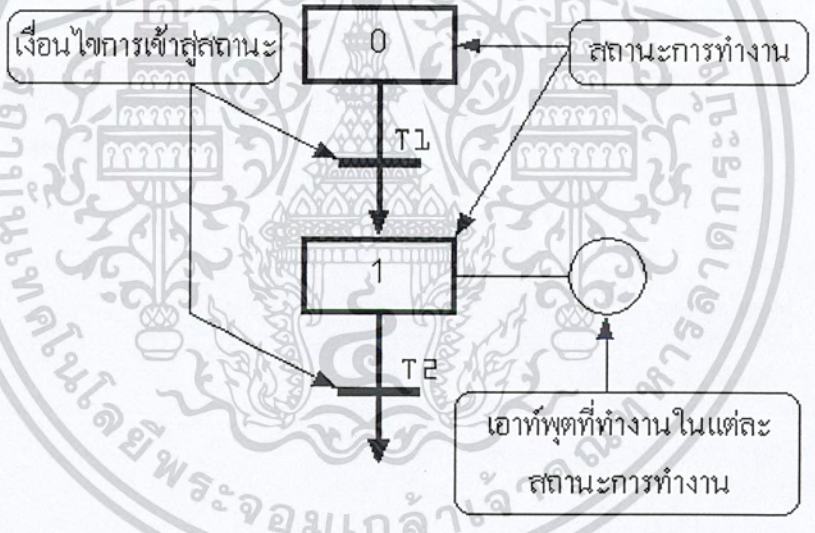
ในการใช้ petri Nets. ช่วยในการออกแบบวงจรแลดเดอร์นั้น พอที่จะสรุปเป็นขั้นตอนต่าง ๆ ได้ดังนี้ คือ

1. วิเคราะห์และทำความเข้าใจกับเงื่อนไขการทำงานของเครื่องจักรกลให้เข้าใจหลักการทำงาน (สมควรเขียนเป็นแผนผังเวลาออกมาเพื่อง่ายในการพิจารณาในขั้นตอนต่อไป)
2. พิจารณาสภาวะการทำงานของเอาท์พุตและอุปกรณ์ช่วยต่าง ๆ ที่ทำงานอยู่ในช่วงเวลาเดียวกัน ให้จัดอยู่ในสถานะการทำงานเดียวกัน
3. พิจารณาว่าเงื่อนไขอะไรที่ทำให้สถานะการทำงานเปลี่ยนแปลงไปจากสถานะของการทำงานใด และไปยังสถานะของการทำงานใด
4. เขียนวงจรการควบคุมออกมาโดยเริ่มจากสถานะการทำงานเริ่มต้นและเขียนตามลำดับการทำงานของสถานะต่าง ๆ โดยที่ระหว่างสถานะของการทำงานในแต่ละสถานะจะถูกขัดขวางโดยเงื่อนไขการเปลี่ยนแปลงสถานะ (ในขั้นตอนที่ 3)
5. ระบุอุปกรณ์ต่าง ๆ ที่จะทำงานในแต่ละสถานะการทำงานนั้นลงไป
6. แปลงจากรูปแบบของ Petri Nets. ให้เป็นวงจรแลดเดอร์

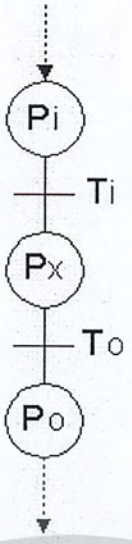
การแปลงจารรูปแบบของ Petri Nets. เป็นวงจรแลดเดอร์

เพื่อให้วงจรควบคุมที่เขียนขึ้นมานั้นสามารถใช้ได้กับ PLC ทั่วไป เราจึงจำเป็นที่จะต้องแปลงวงจรควบคุมที่เขียนขึ้นมาให้อยู่ในรูปแบบของวงจรแลดเดอร์เพื่อให้ PLC รับรู้ได้ โดยที่ในรูปแบบเพททริเนทส์จะเห็นได้ว่ามีส่วนที่สำคัญอยู่ 3 ส่วน คือ

1. เงื่อนไขการเข้าสู่สถานะการทำงาน ซึ่งเงื่อนไขของสถานะการทำงานนี้จะเป็นตัวกำหนดว่าจะให้วงจรนั้นทำงานอยู่ที่สภาวะการทำงานใด
2. สถานะการทำงาน เป็นลำดับการทำงานของแต่ละลำดับขั้นตอนของระบบการทำงาน โดยการทำงานนันจะขึ้นอยู่กับเงื่อนไขการเข้าสู่สถานะการทำงานเพื่อเข้าสู่การทำงานในแต่ละลำดับขั้นตอน
3. อุปกรณ์ที่จะทำงานในแต่ละสถานะการทำงาน ในส่วนนี้จะเป็นตัวบอกว่าในแต่ละสถานะการทำงานนั้นจะมีเอาท์พุตตัวใดบ้างที่ทำงาน



รูปที่ 3.1 ส่วนประกอบที่สำคัญของวงจรควบคุมในรูปแบบของเพททริเนทส์



รูปที่ 3.2 ส่วนประกอบที่สำคัญของวงจรควบคุมในรูปแบบของเพททริเนทส์

เมื่อ

Px = Place (สถานะการทำงาน) ใดๆ

Pi = Place input ของ Place (สถานะการทำงาน) ใดๆ

Po = Place output ของ Place (สถานะการทำงาน) ใดๆ

Ti = Transition input ของ Place (สถานะการทำงาน) ใดๆ

To = Transition output ของ Place (สถานะการทำงาน) ใดๆ

จากเงื่อนไขการส่งผ่าน Token ของ Transition เมื่อนำมาประยุกต์เพื่อใช้ในการออกแบบวงจรแลดเดอร์ พอที่จะสรุปได้ว่า ที่ Place ใดๆก็ตาม จะทำงานได้ก็ต่อเมื่อ Place input และ Transition input ของ Place นั้น มีสภาวะลอจิกเป็น "1" (Place input, Transition input พร้อมที่จะทำงาน และ Transition input ทำการส่งผ่าน Token จาก Place input ไปยัง Place ใดๆ) และที่ Place ใดๆ นี้จะหยุดทำงานเมื่อ Place output และ Transition output ของ Placeนั้น มีสภาวะลอจิกเป็น "1" (Transition output พร้อมที่จะทำงาน และทำการส่งผ่าน Token จาก Place ใดๆ ไปยัง Place output และหลังจากส่งผ่าน Token ไปแล้วจะไม่มีผลกับ Place ใด) ซึ่งสามารถที่จะเขียนเป็นสมการได้ดังนี้

$$Px = (Pi\ Ti + Px)(\overline{Po} + \overline{To})$$

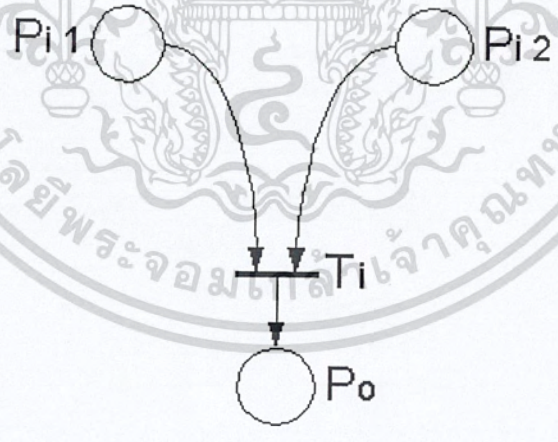
ยกเว้นที่ Place ใดๆ นั้นเป็นสถานะการทำงานเริ่มต้น (P0) ซึ่งจะมีผลของเงื่อนไขที่ว่า Token จะเป็น 1 เมื่อเริ่มทำงานดังนั้นจึงต้องเพิ่มส่วนนี้เข้ามาซึ่งจะได้สมการดังนี้

$$P0 = (Pi\ Ti\ +\ \overline{Pj}\ + P\ 0)(\ \overline{Po} + \overline{To})$$

เมื่อ J = 1 ถึง n (n = จำนวนสถานะการทำงาน)

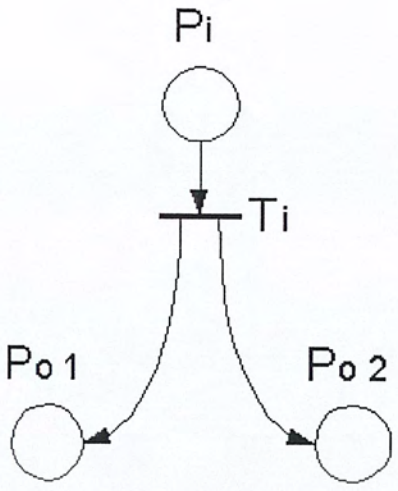
$Po = Pi\ Ti$

รูปที่ 3.3 Sample



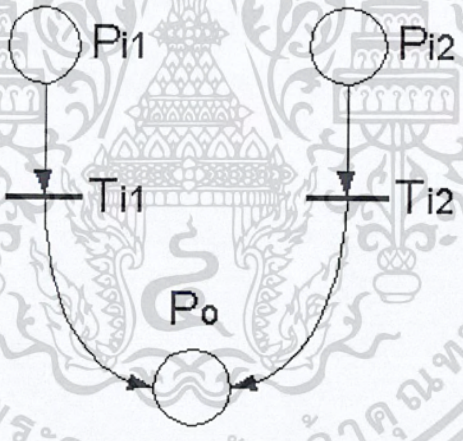
$Po = Pi1\ Pi2\ Ti$

รูปที่ 3.4 JunctionAND



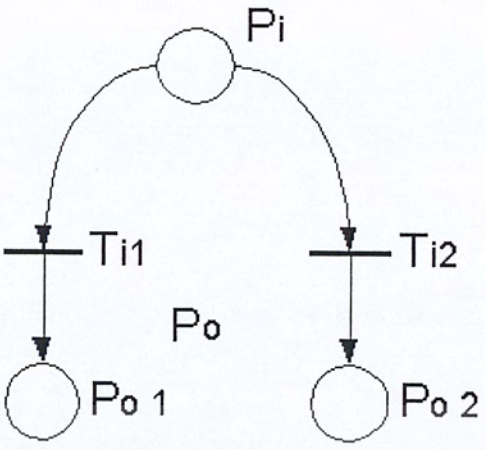
Po1 = Pi Ti
Po2 = P i Ti

รูปที่ 3.5 DistributionAND



Po = Pi1 Ti1 + P i2 Ti2

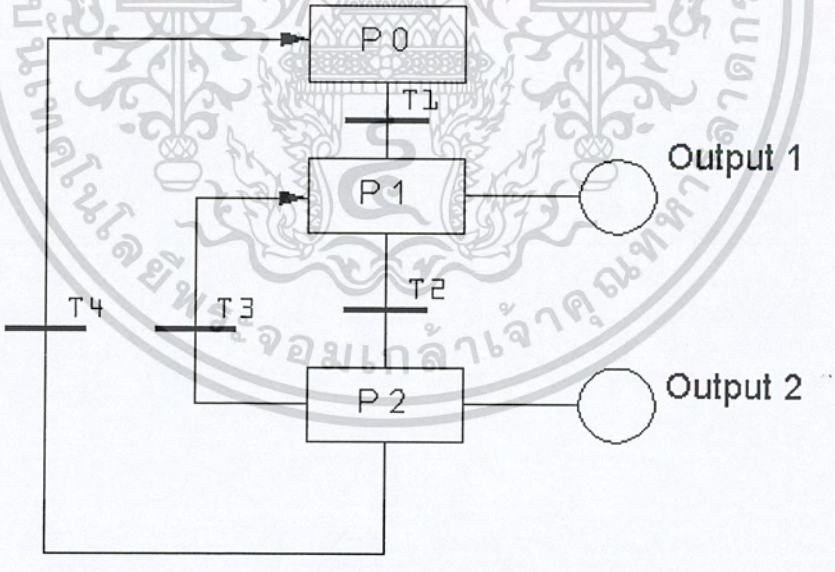
รูปที่ 3.6 JunctionOR



Po1 = Pi Ti1
Po2 = P i Ti2

รูปที่ 3.7 DistributionOR

ตัวอย่าง



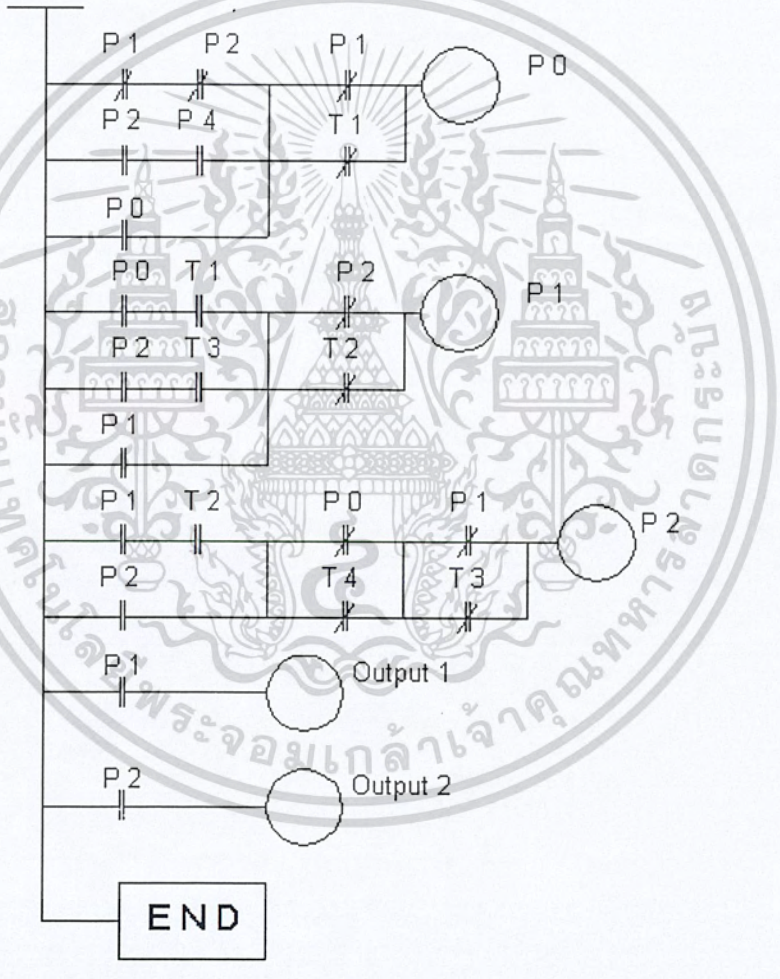
รูปที่ 3.8 ตัวอย่างของรูปแบบ Grafcet

$$P0 = (P2T4 + \overline{P1}\overline{P2} + P0)(\overline{P1} + \overline{T1})$$

$$P1 = (P0T1 + \overline{P2}\overline{T3} + P1)(\overline{P2} + \overline{T2})$$

$$P2 = (P1T2 + P2)(\overline{P2} + \overline{T3})(\overline{P2} + \overline{T4})$$

จากสมการข้างต้นเราสามารถที่จะเขียนเป็นวงจรแลดเดอร์ได้ตามการกระทำทางลอจิกของอุปกรณ์หรือหน้าสัมผัสของตัวแปรในสมการข้างต้นนั้นได้เลย และจากสมการข้างต้นนั้นจะเขียนเป็นวงจรแลดเดอร์ได้ดังนี้ คือ



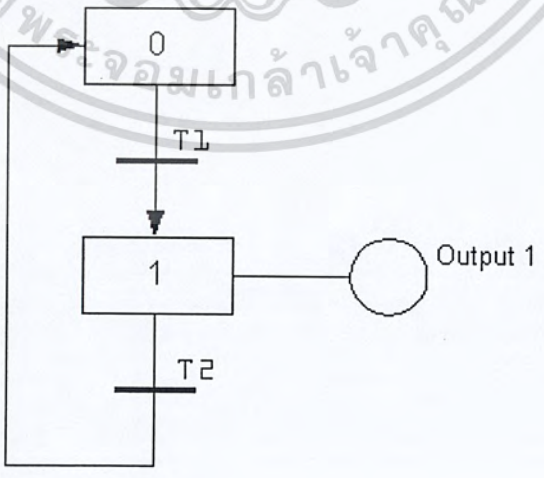
รูปที่ 3.9 ตัวอย่างของการแปลงจาก Petri net เป็นวงจรแลดเดอร์

ถ้าเราใช้วิธีการที่กล่าวมาในข้างต้นวงจรแลดเดอร์ที่ได้มานั้นจะยาวมากดั้งนั้นในกรณีที่เงื่อนไขการทำงานเป็นวงรอบที่แน่นอนเราสามารถที่จะใช้ฟังชั่นก์ของ PLC ที่เกี่ยวกับการเลื่อนสถานะการทำงาน เช่น ชีฟรีจิสเตอร์ (shift register) มาช่วยในการเปลี่ยนสถานะของการทำงานได้ ซึ่งในเงื่อนไขที่สามารถทำแบบนี้ได้นั้น จะแบ่งการทำงานออกได้เป็น 2 แบบ คือ

1. เงื่อนไขการทำงานแบบอนุกรม หรือ เงื่อนไขที่เป็นลำดับขั้นตอนการทำงานที่แน่นอน เช่น ทำงานจากกระบวนการที่ 1 หลังจากนั้นจะทำงานในกระบวนการที่ 2, 3, 4,... และจบการทำงาน
2. เงื่อนไขการทำงานแบบขนาน หรือ เงื่อนไขการทำงานที่เป็นแบบทางเลือกว่าจะทำงานในกระบวนการใด กระบวนการหนึ่ง ไม่เป็นลำดับขั้นตอนการทำงานที่แน่นอน เช่น เลือกที่จะทำงานในกระบวนการที่ 1 หรือ ทำงานในกระบวนการที่ 2 หลังจากนั้นให้จบการทำงาน

ในการแปลงจากรูปแบบของเพททริเนทส์ เป็นวงจรแลดเดอร์ ฟังชั่นก์ของ PLC ที่เกี่ยวกับการเลื่อนสถานะการทำงานอาจจะมีหลายฟังชั่น ขึ้นอยู่กับ PLC ที่ใช้ แต่ในวิธีที่จะกล่าวถึงต่อไปนี้จะใช้ ชีฟรีจิสเตอร์ (shift register) เป็นเครื่องมือที่ใช้ในการเปลี่ยนสถานะการทำงานในแต่ละสถานะการทำงาน และก็จะขึ้นอยู่กับรูปแบบของเงื่อนไขการทำงานด้วย ในการเขียนวงจรแลดเดอร์เราจะแยกออกเป็น 2 ส่วน ส่วนของการควบคุมสถานะการทำงาน และในส่วนของการควบคุมเอาท์พุตของระบบควบคุม

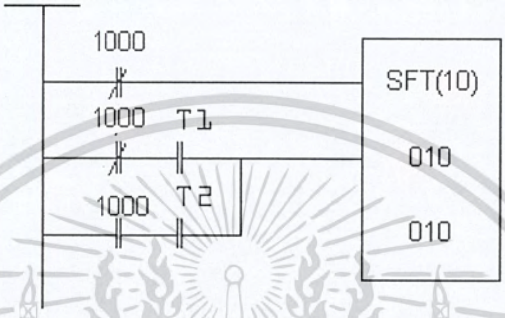
เงื่อนไขการทำงานแบบอนุกรม



รูปที่ 3.10 วงจรควบคุมในรูปแบบของเพททริเนทส์แบบอนุกรม

ส่วนของการควบคุมสถานะการทำงาน

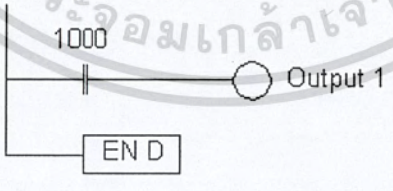
เงื่อนไขของการเข้าสู่สถานะการทำงานจะทำงานได้ก็ต่อเมื่อสถานะการทำงานที่อยู่ก่อนหน้ามันขึ้นไปมีสภาวะลอจิกเป็น 1 จากรูปที่ 3.11 ส่วนควบคุมสถานะการทำงานของวงจรควบคุมในรูปแบบของวงจรแลดเดอร์แบบอนุกรมจะเห็นได้ว่า T2 จะอยู่หลัง สถานะการทำงานที่ 1 ซึ่งจะถูกควบคุมการทำงานโดยพื้นที่ภายในของ PLC ตำแหน่งที่ 1000



รูปที่ 3.11 ส่วนควบคุมสถานะการทำงานของวงจรควบคุมในรูปแบบของวงจรแลดเดอร์แบบอนุกรม

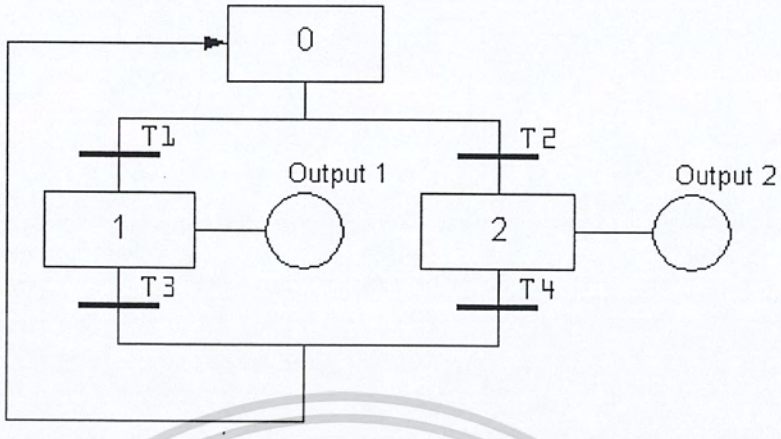
ส่วนของการควบคุมเอาท์พุตของระบบควบคุม

ในส่วนของเอาท์พุตที่จะนำไปใช้ควบคุมอุปกรณ์นั้นจะถูกควบคุมโดยพื้นที่ภายในของ PLC



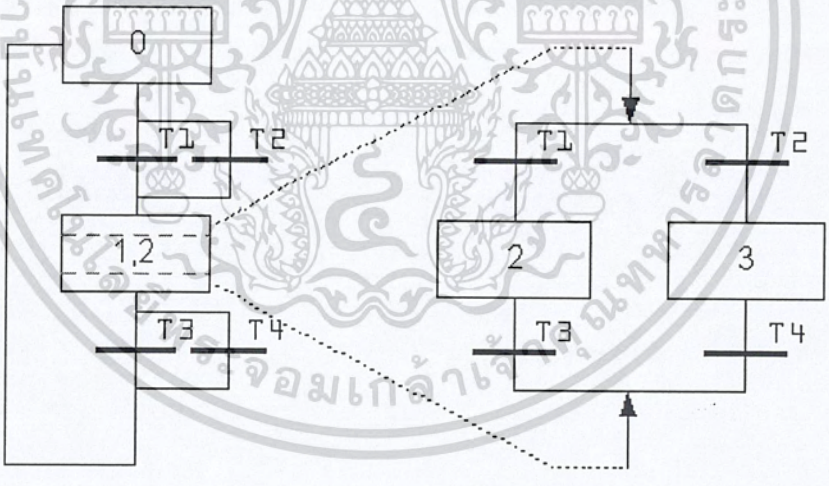
รูปที่ 3.12 ส่วนควบคุมเอาท์พุตของวงจรควบคุมในรูปแบบของวงจรแลดเดอร์แบบอนุกรม

เงื่อนไขการทำงานแบบขนาน



รูปที่ 3.13 วงจรควบคุมในรูปแบบของเพททริเนทส์แบบขนาน

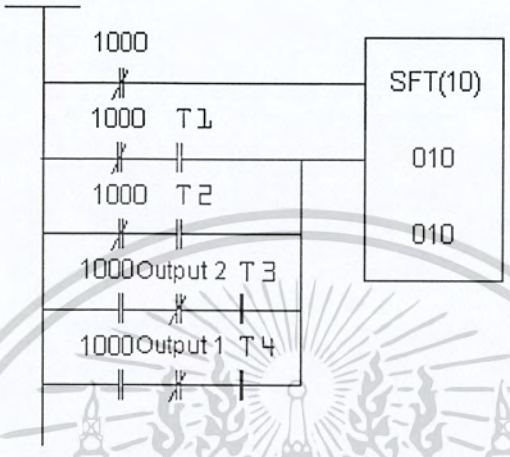
เมื่อรวมสถานะของการทำงานที่ขนานกันอยู่ให้เป็นสถานะการทำงานเดียวกันแล้วเราได้วงจรควบคุมในรูปแบบของเพททริเนทส์แบบอนุกรมดังนี้



รูปที่ 3.14 วงจรควบคุมในรูปแบบของเพททริเนทส์แบบขนานเมื่อรวมให้เป็นแบบอนุกรม

ส่วนของการควบคุมสถานะการทำงาน

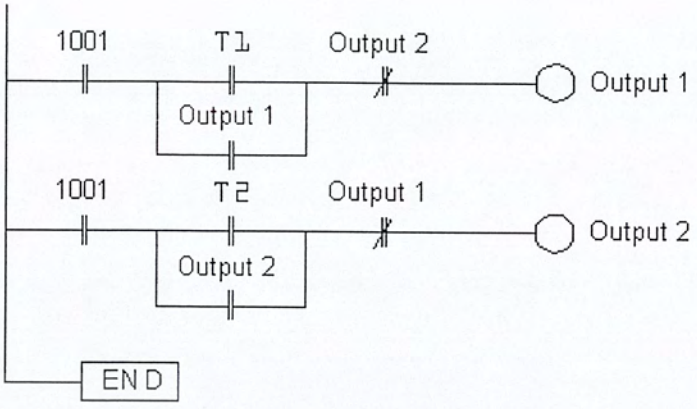
เมื่อรวมสถานะของการทำงานที่ขนานกันอยู่ให้เป็นสถานะการทำงานเดียวกันแล้วเราก็จะสามารถที่จะเขียนส่วนของการควบคุมสถานะของการทำงานได้เหมือนกับส่วนของการควบคุมสถานะของการทำงานในรูปแบบของเพททริเนทส์แบบอนุกรมได้



รูปที่ 3.15 ส่วนของการควบคุมสถานะการทำงานของวงจรควบคุมในรูปแบบของเพททริเนทส์แบบขนาน

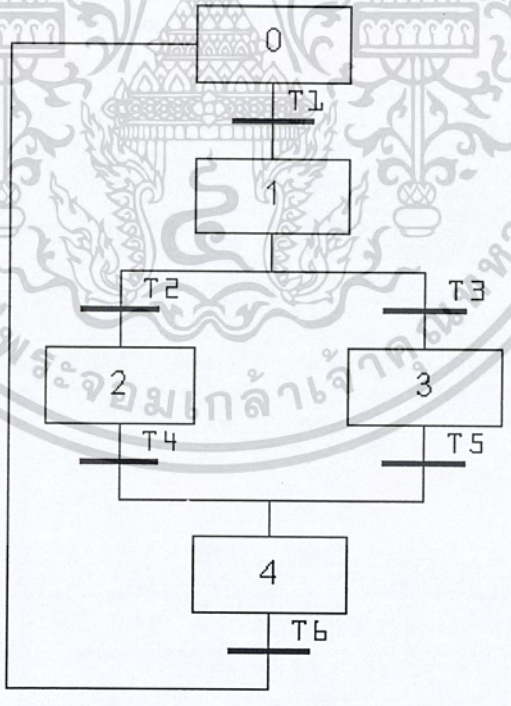
ส่วนของการควบคุมเอาท์พุตของระบบควบคุม

ถึงแม้ว่าเราจะรวมสถานะการทำงานที่ขนานกันอยู่ให้เป็นสถานะการทำงานเดียวกันแล้วแต่นั้นมันก็เป็นแค่การเปรียบเหมือนว่าเป็นสถานะการทำงานเดียวเพื่อใช้ในการเลื่อนสถานะของการทำงานเท่านั้น แต่ในส่วนของการควบคุมเอาท์พุตนั้นเราก็ยังต้องแยกกันอยู่โดยจะแยกกันหลังจากผ่านพื้นที่ภายในของ PLC ที่ใช้เป็นสถานะการทำงานแต่ละสถานะการทำงานแล้ว และในส่วนของเงื่อนไขของการเข้าสู่สถานะการทำงานเราจะต้องขนานด้วยหน้าสัมผัสของตัวเอาท์พุตเพื่อล็อคการทำงานให้อยู่ในสถานะการทำงานนั้นจนกว่าจะมีการเปลี่ยนแปลง ในกรณีที่ไม่มีหน้าสัมผัสของเอาท์พุตที่ใช้ได้ ให้สั่งออกเอาท์พุตที่พื้นที่ภายในของ PLC เพื่อใช้ช่วยในการล็อคสถานะของการทำงาน

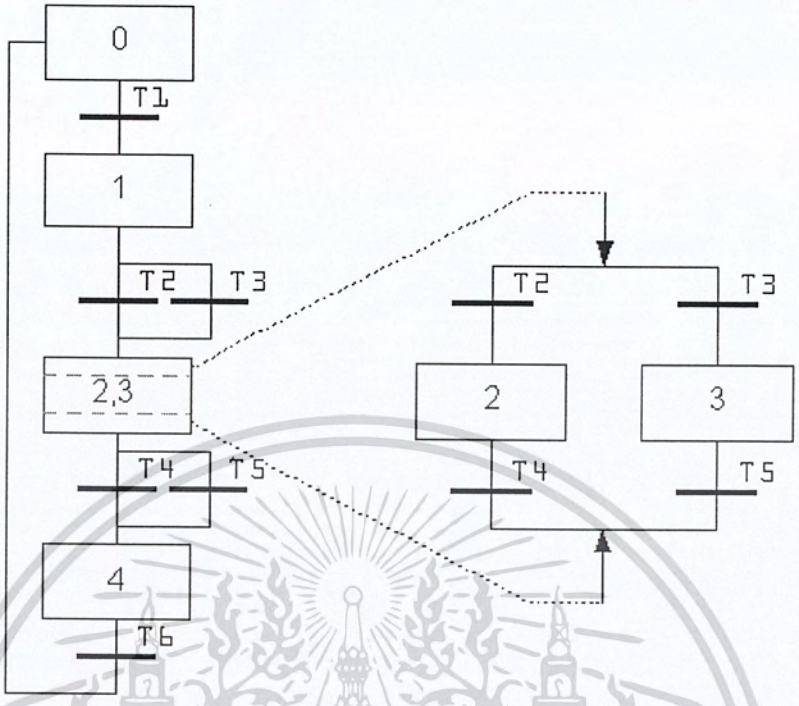


รูปที่ 3.16 ส่วนของการควบคุมเอาท์พุตของวงจรควบคุมในรูปแบบของเพททริเนทส์แบบขนาน

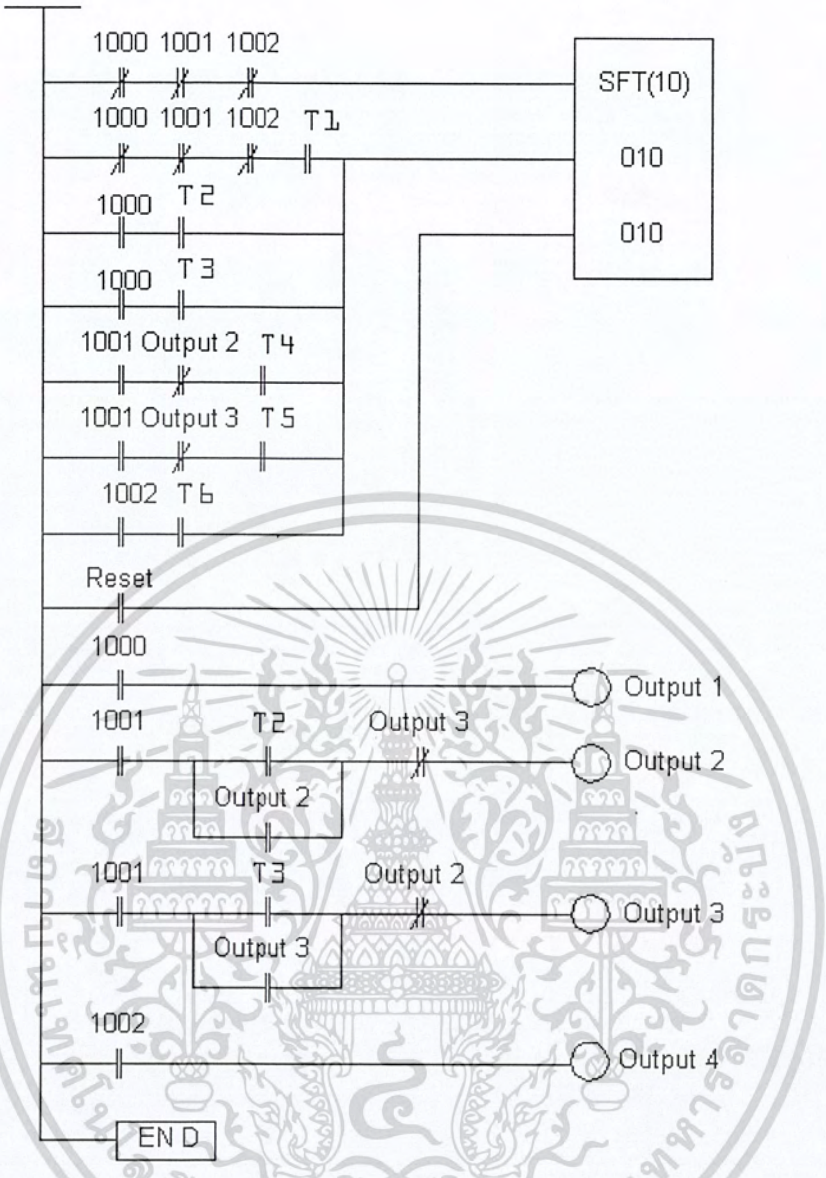
ตัวอย่างที่ 3.1 วงจรควบคุมในรูปแบบของเพททริเนทส์แบบผสม



รูปที่ 3.17 วงจรควบคุมในรูปแบบของเพททริเนทส์แบบผสม



รูปที่ 3.18 วงจรควบคุมในรูปแบบของเพททริเนทส์แบบผสมเมื่อรวมให้เป็นแบบอนุกรม



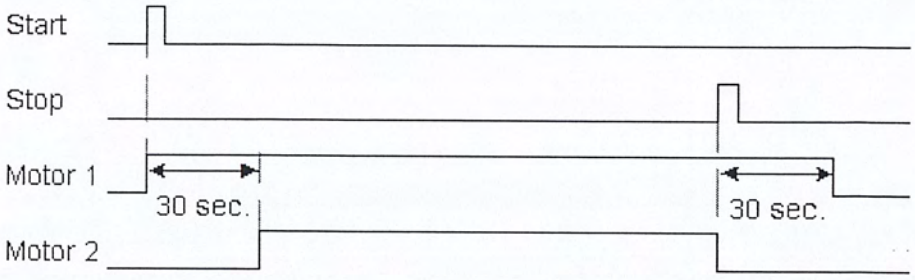
รูปที่ 3.19 วงจรควบคุมในรูปแบบของวงจรแลดเดอร์แบบผสม

ตัวอย่างที่ 3.2 การสตาร์ทมอเตอร์แบบเรียงลำดับจำนวน 2 ตัว

เงื่อนไขการทำงาน

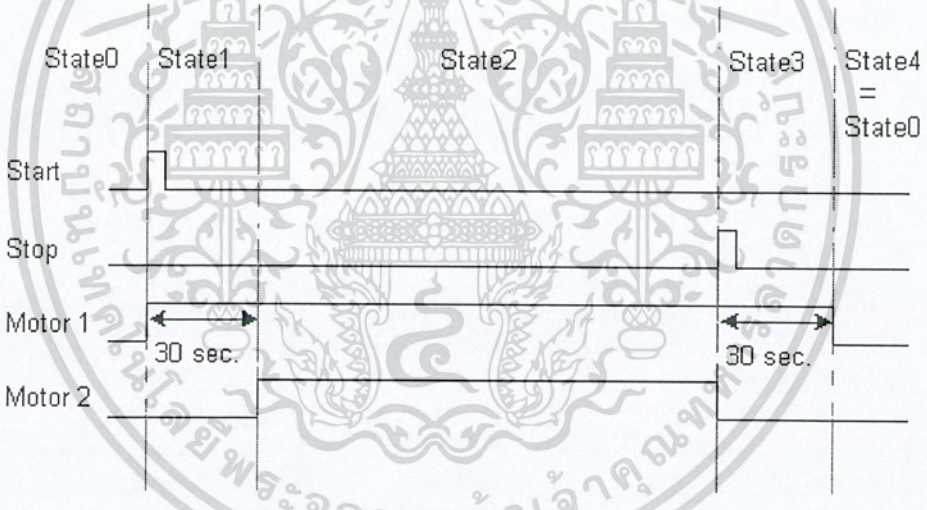
เมื่อกดสวิตช์สตาร์ทแล้วให้มอเตอร์ตัวที่หนึ่งทำงาน หลังจากนั้น 30 วินาทีให้มอเตอร์ตัวที่สองทำงาน เมื่อกดสวิทช์หยุดการทำงานให้มอเตอร์ตัวที่สองหยุดการทำงานก่อนหลังจากนั้นเป็นเวลา 30 วินาที ให้มอเตอร์ตัวที่หนึ่งหยุดการทำงาน

ขั้นตอนที่ 1 พิจารณาเงื่อนไขการทำงานต่าง ๆ ให้เข้าใจ



รูปที่ 3.20 แผนผังเวลาการทำงาน

ขั้นตอนที่ 2 พิจารณาที่เอาท์พุต เพื่อจัดสถานะการทำงาน



รูปที่ 3.21 การแบ่งสถานะการทำงาน

สถานะที่ 0 คือ สถานะเริ่มต้นไม่มีเอาท์พุตตัวใดทำงาน
สถานะที่ 1 มอเตอร์ตัวที่ 1 ทำงาน และเริ่มนับเวลา
สถานะที่ 2 มอเตอร์ตัวที่ 1 และมอเตอร์ตัวที่ 2 ทำงาน
สถานะที่ 3 มอเตอร์ตัวที่ 1 ทำงานและเริ่มนับเวลา
สถานะที่ 4 หยุดทำงานทั้งหมดกลับเข้าสู่สถานะเริ่มต้น

ขั้นตอนที่ 3 พิจารณาเงื่อนไขที่ทำให้สถานะการทำงานเปลี่ยนแปลง

จากรูปที่ 3.2 เมื่อเราพิจารณาจากเงื่อนไขการเปลี่ยนแปลงสถานะการทำงานต่าง ๆ จะได้สถานะของการทำงานดังนี้ คือ

เริ่มเข้าสถานะที่ 0 จะเป็นสัญญาณเริ่มทำงานโปรแกรม

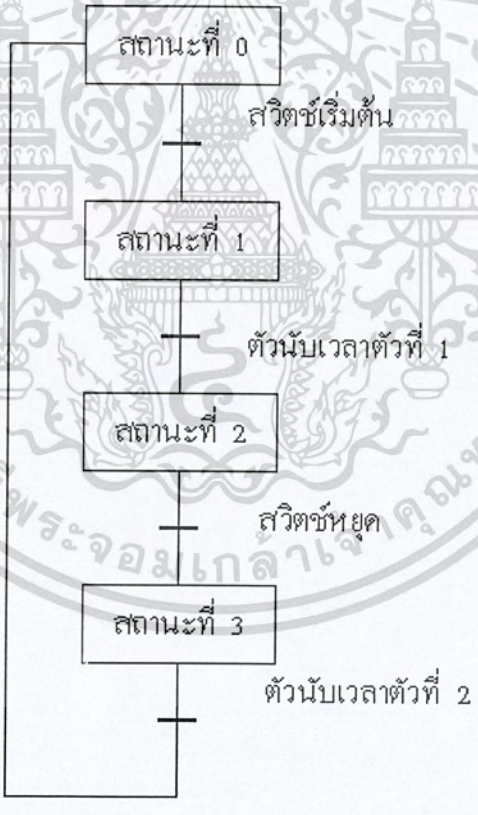
สถานะที่ 0 เป็นสถานะที่ 1 คือ สวิตช์เริ่มต้น

สถานะที่ 1 เป็นสถานะที่ 2 คือ ตัวนับเวลาตัวที่ 1

สถานะที่ 2 เป็นสถานะที่ 3 คือ สวิตช์หยุด

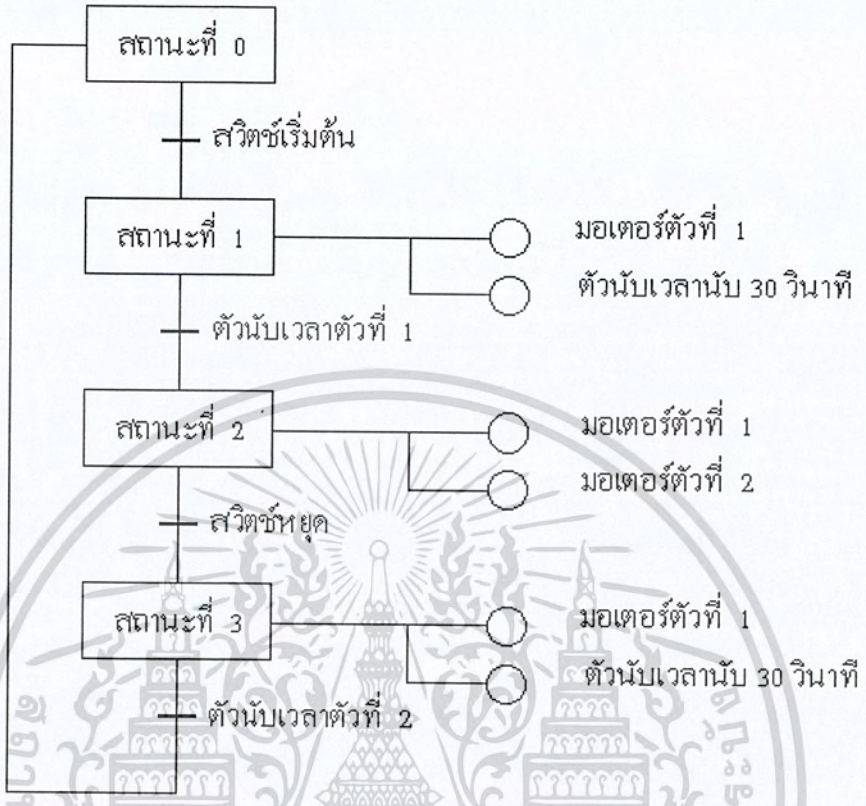
สถานะที่ 3 เป็นสถานะที่ 4 คือ ตัวนับเวลาตัวที่ 2

ขั้นตอนที่ 4 เขียนวงจรควบคุม



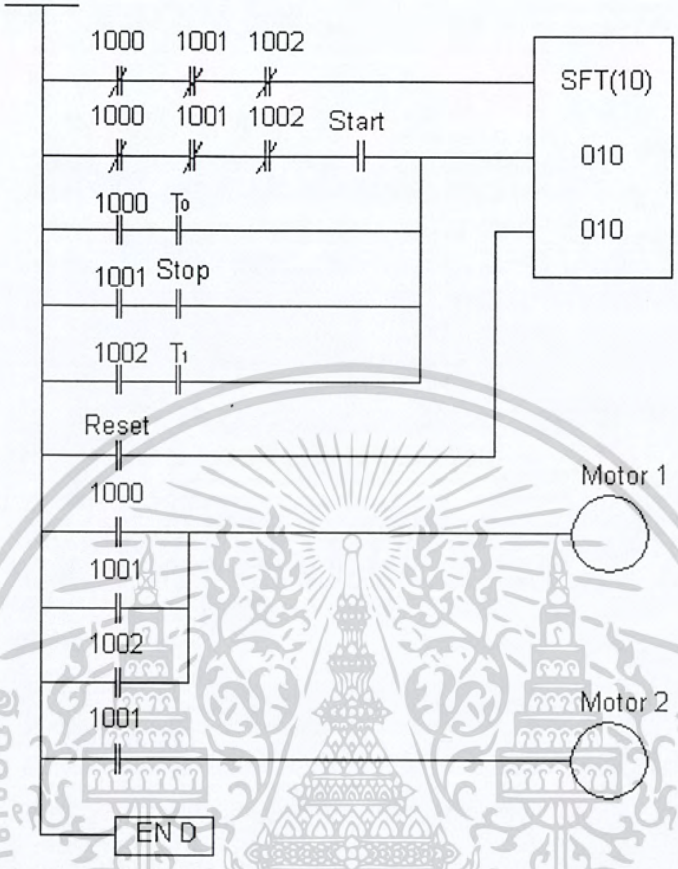
รูปที่ 3.22 วงรอบการควบคุม

ขั้นตอนที่ 5 ระบุ อุปกรณ์ต่าง ๆ ในแต่ละสถานะการทำงาน



รูปที่ 3.23 วงจรการควบคุมในรูปแบบของเพททริเนทส์

ขั้นตอนที่ 6 เปลี่ยนจากรูปแบบของ Petri Nets เป็นวงจรแลดเดอร์



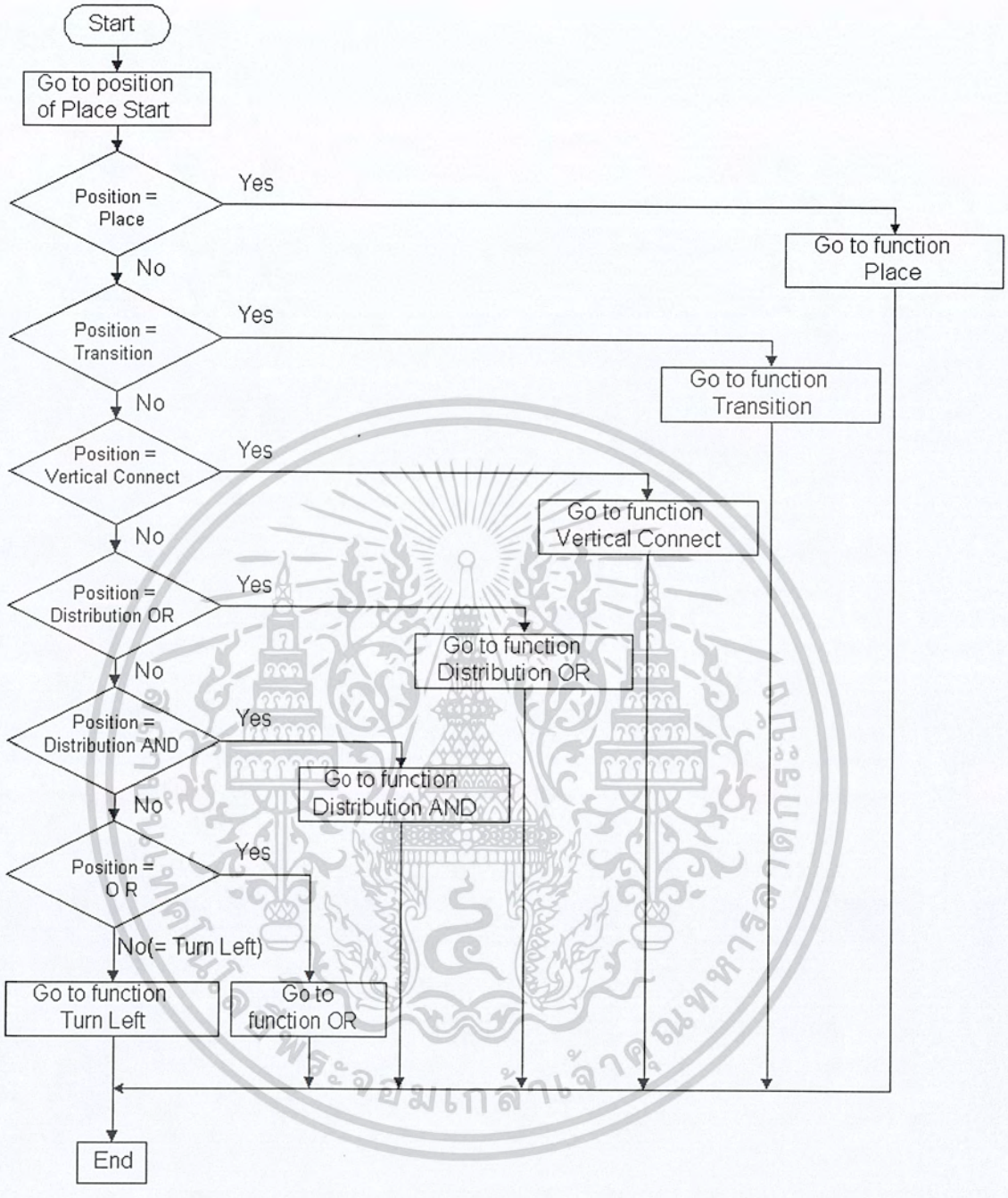
รูปที่ 3.24 วงจรการควบคุมในรูปแบบของวงจรแลดเดอร์

3.2 การออกแบบโปรแกรม

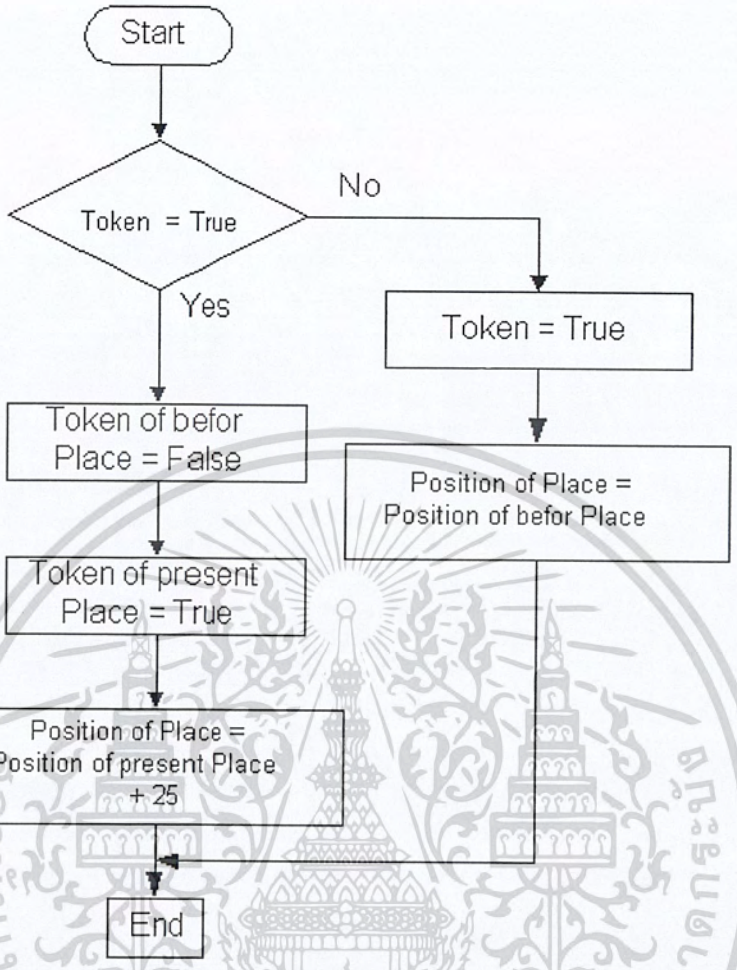
เพื่อที่จะแสดงว่าสามารถที่จะใช้ทฤษฎี Petri Net ในการควบคุมและจำลองการทำงานของเครื่องจักรกลต่าง ๆ ได้จริง ซึ่งได้เขียนโปรแกรมขึ้นมาเพื่อให้ผู้ใช้งานเขียนโปรแกรมควบคุมเครื่องจักรกล ตรวจสอบวงจรควบคุมและจำลองเหตุการณ์และเงื่อนไขต่าง ๆ ของเครื่องจักรกลที่จะเกิดขึ้นบนคอมพิวเตอร์โดยไม่จำเป็นต้องต่อเข้ากับอุปกรณ์จริงก่อนที่จะนำวงจรนี้ไปใช้ในการควบคุมเครื่องจักรกลจริง ๆ หรือจะต่อเข้ากับอุปกรณ์จริงเพื่อให้ควบคุมเครื่องจักรกลเลยก็ได้

ในการเขียนโปรแกรมนั้นได้ออกแบบโปรแกรมให้มีลักษณะเป็นฟังก์ชัน ซึ่งจะถูกเรียกขึ้นมาใช้เมื่อโปรแกรมตรวจสอบเจออุปกรณ์ต่าง ๆ และมีสมมติฐานว่าเวลาในการประมวลผลของโปรแกรมที่เขียนขึ้นมาในแต่ละบรรทัดนั้นน้อยมาก ๆ

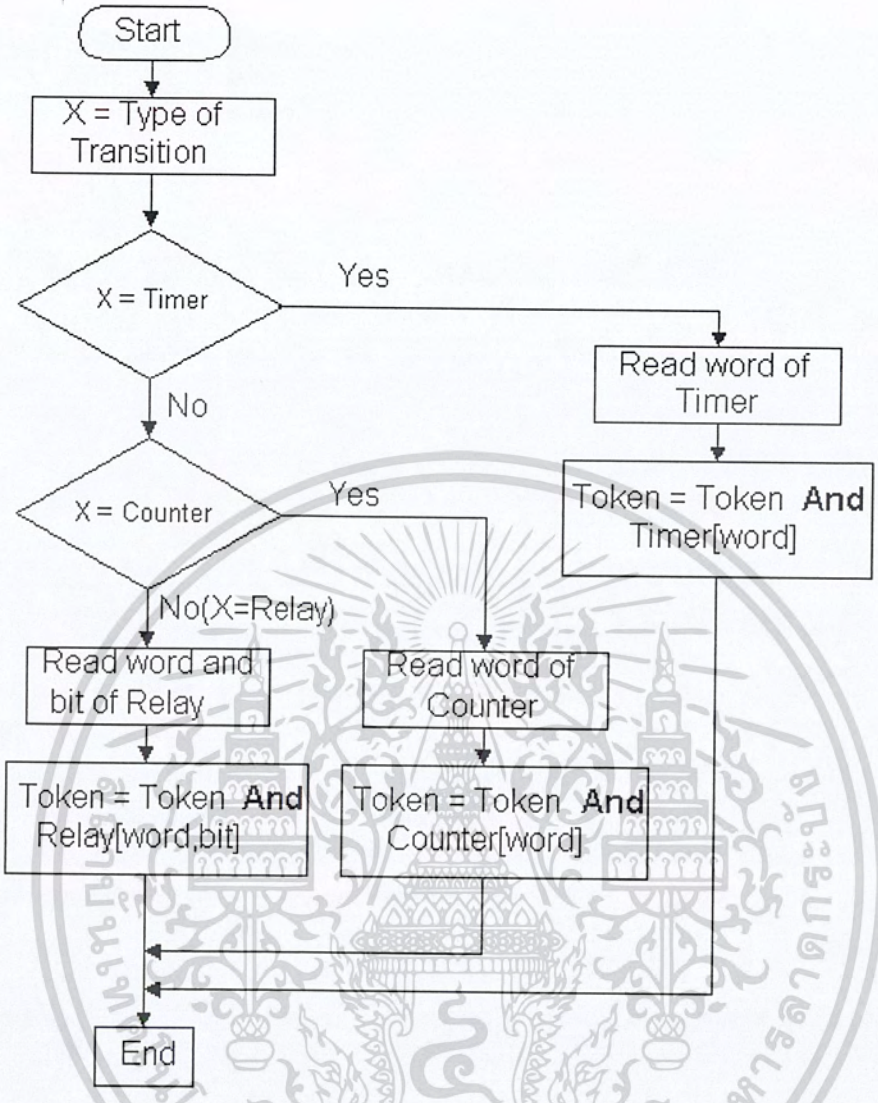
ในหัวข้อนี้จะแสดงโฟว์ชาร์ตที่สำคัญ ๆ ในการเขียนโปรแกรมนี้เท่านั้น



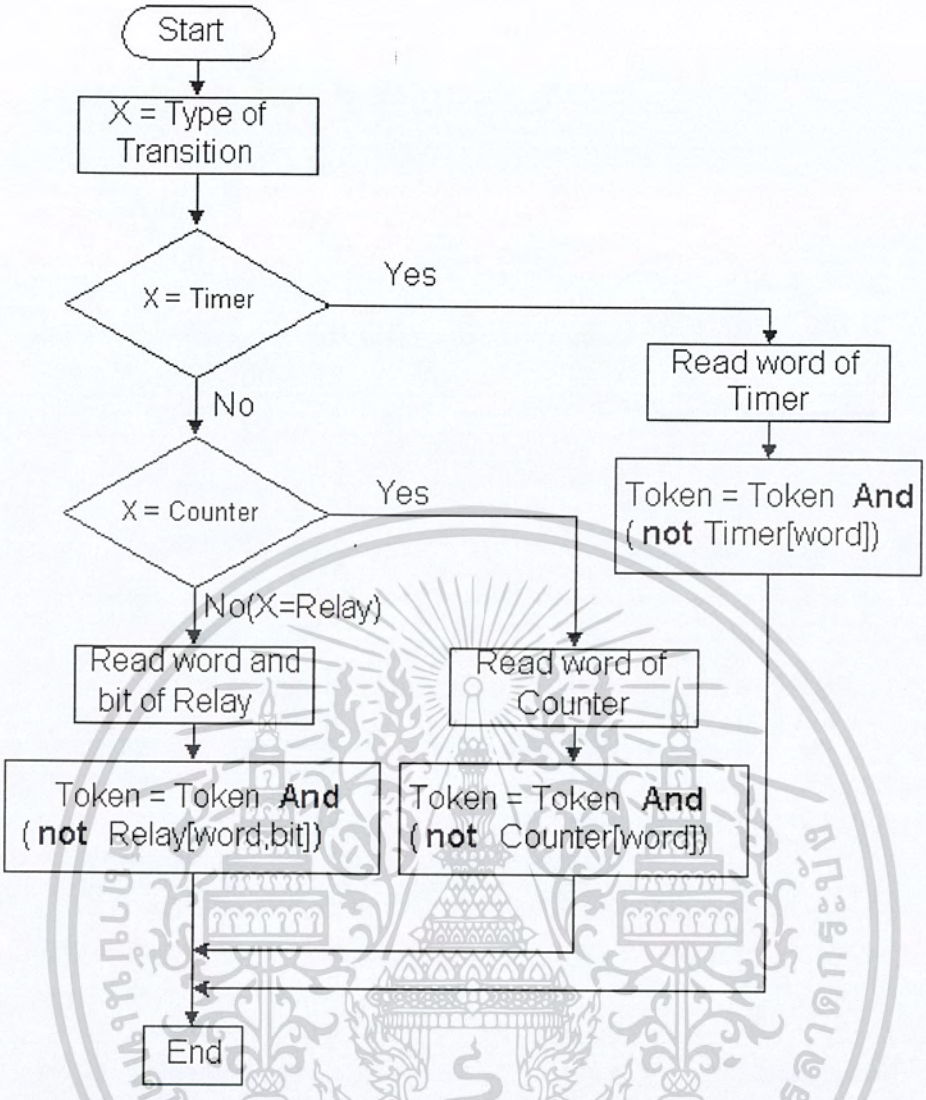
รูปที่ 3.25 โฟว์ชาร์ตแสดงการทำงานของฟังก์ชัน RUN



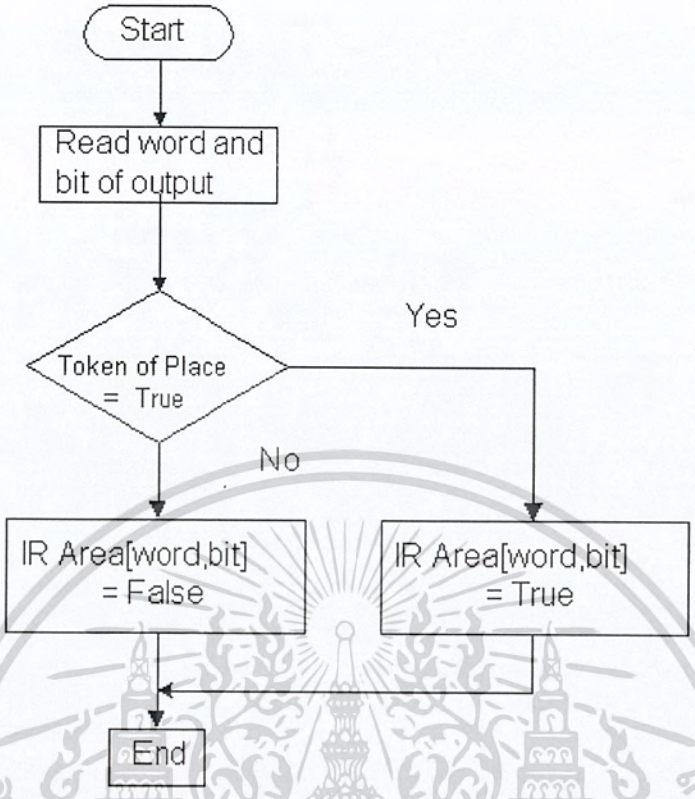
รูปที่ 3.26 โฟว์ชาร์ตแสดงการทำงานของฟังก์ชัน Place



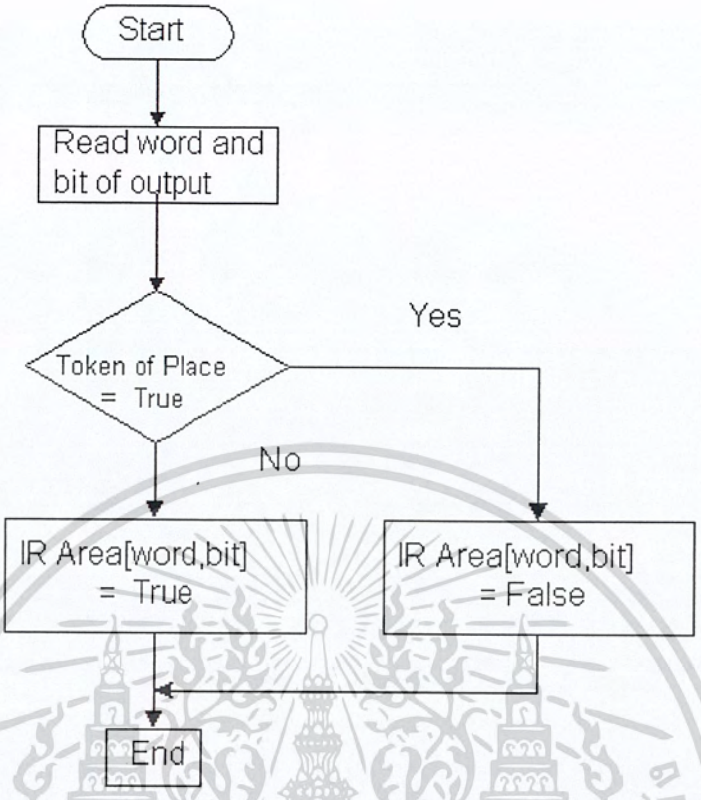
รูปที่ 3.27 โฟว์ชาร์ตแสดงการทำงานของฟังก์ชัน Transition



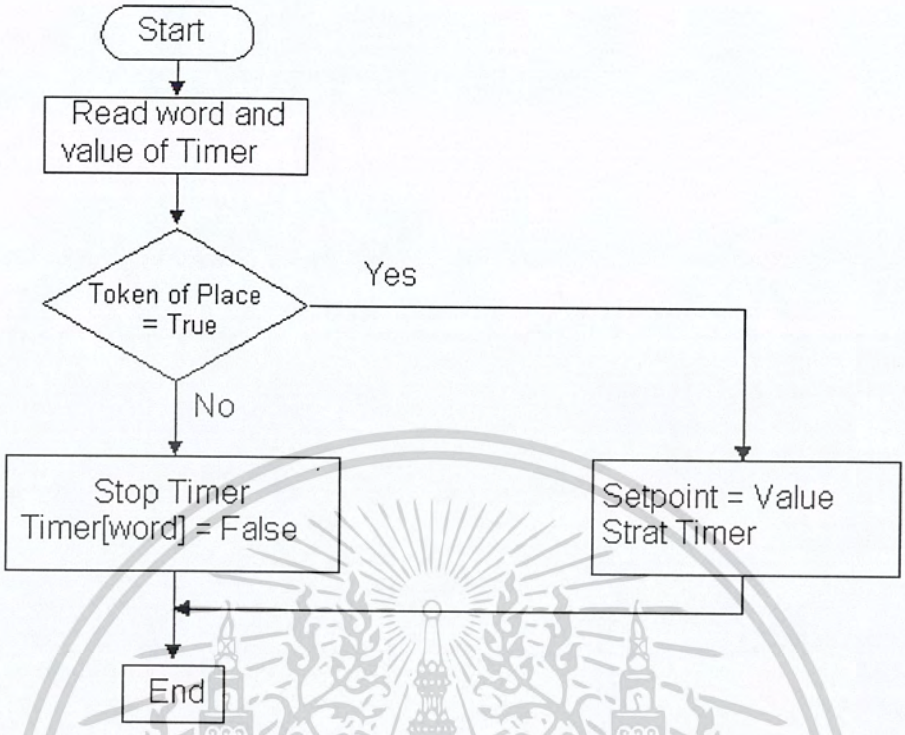
รูปที่ 3.28 โฟว์ชาร์ตแสดงการทำงานของฟังก์ชั่น Transition Not



รูปที่ 3.29 โฟว์ชาร์ตแสดงการทำงานของฟังก์ชั่น Output



รูปที่ 3.30 โฟว์ชาร์ตแสดงการทำงานของฟังก์ชัน Output Not



รูปที่ 3.31 โฟว์ชาร์ตแสดงการทำงานของฟังก์ชัน Timer

บทที่ 4
เกี่ยวกับโปรแกรม และการใช้งานโปรแกรม

โปรแกรมที่เราสร้างขึ้นมานี้เป็นโปรแกรมที่ใช้สำหรับประมวลผล เพื่อใช้ในการควบคุมอุปกรณ์ต่างๆ แบบ on – off Control โดยที่โปรแกรมนี้เขียนมาจากโปรแกรม Delphi และเขียนบนพื้นฐานของทฤษฎี Grafcet ซึ่งเป็นทฤษฎีที่ประยุกต์มาจาก ทฤษฎีของ Petri Nets และใช้สำหรับการควบคุมแบบ Logic Control

4.1 ความสามารถของโปรแกรม

โปรแกรมนี้ได้สร้างขึ้นมาบนพื้นฐานของทฤษฎี Petri Nets ซึ่งจะใช้สำหรับเขียนโปรแกรมควบคุมอุปกรณ์แบบ on – off Control โดยที่จะรับค่ามาจากหน่วยอินพุตโมดุล ของ PLC จากนั้นจะทำการประมวลผลตามโปรแกรมที่ผู้ใช้งานเขียนขึ้นมาแล้วส่งค่าไปยังเอาท์พุตโมดุลของ PLC เพื่อนำค่าสถานะทางลอจิกที่ได้นั้นไปควบคุมอุปกรณ์ที่ต้องการ และในโปรแกรมนี้ยังมีความสามารถที่จะช่วยตรวจสอบ deadlock ที่เกิดขึ้นในโปรแกรมที่ผู้ใช้งานเขียนขึ้นมา ซึ่งเป็นสาเหตุของปัญหาในการทำงานของระบบควบคุมต่างๆ และนอกจากนี้โปรแกรมนี้ยังสามารถที่จะจำลองเหตุการณ์ต่าง ๆ ของโปรแกรมที่ผู้ใช้งานเขียนขึ้นก่อนการใช้งานจริงได้บนคอมพิมเตอร์ได้โดยไม่จำเป็นต้องต่อเข้ากับ PLC ก็ได้

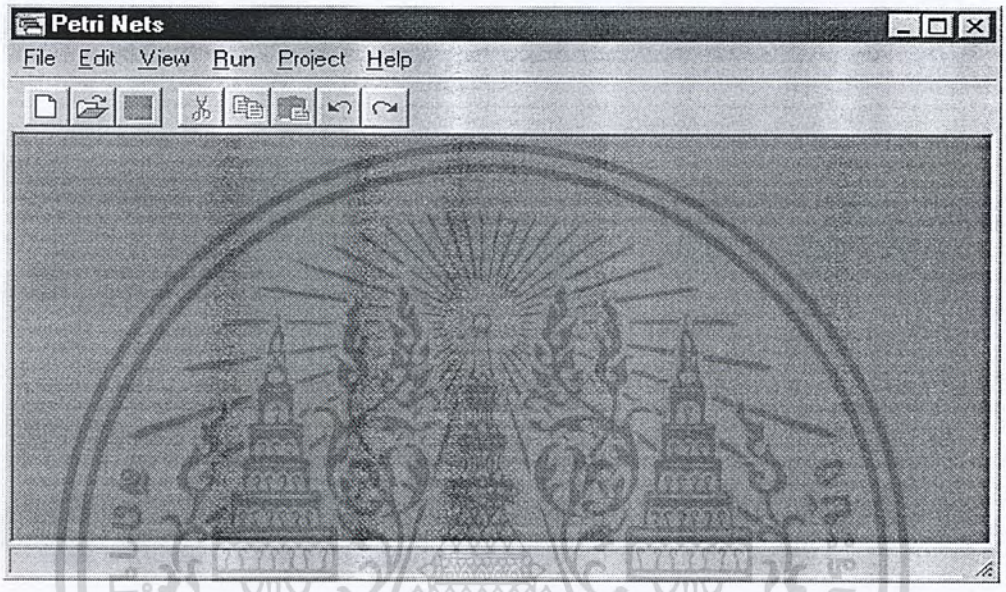
4.2 การติดตั้งใช้งาน

โปรแกรมนี้ได้ถูกสร้างขึ้นมาจากโปรแกรม Delphi ซึ่งเป็นเครื่องมือในการสร้างและพัฒนาโปรแกรมขึ้นมา อีกทั้งยังสามารถที่จะคอมไพล์โปรแกรมที่ผู้ใช้งาน สร้างขึ้นมาให้เป็นไฟล์นามสกุล .EXE ซึ่งผู้ใช้งานจะสามารถนำไฟล์ดังกล่าวเพียงไฟล์เดียวไปวันที่ PC เครื่องใดก็ได้ แต่ในโปรแกรมที่สร้างขึ้นมานี้จะต้องอ้างถึงไฟล์ข้อมูล ซึ่งเป็นรูปภาพต่าง ๆ ที่จะถูกเรียกขึ้นมาในขณะที่ผู้ใช้งานโปรแกรม เขียนโปรแกรมควบคุมขึ้นมา ดังนั้นจึงต้องนำไฟล์รูปภาพต่าง ๆ นี้ใส่เข้าไปด้วยโดยจะต้องใส่ให้ถูกตำแหน่งด้วยไม่เช่นนั้นโปรแกรมจะไม่สามารถทำงานได้ ซึ่งไฟล์ และตำแหน่ง ดังกล่าวก็คือให้คัดลอกแฟ้มข้อมูล ที่ชื่อว่า picture project ไปไว้ใน Drive C:\ ซึ่งจะเป็นที่สำหรับการอ้างถึงของโปรแกรม

4.3 หน้าจอและส่วนประกอบที่สำคัญ

4.3.1 ฟอร์มหลัก

ฟอร์มหลักนี้เป็นฟอร์มเริ่มต้นในการใช้งานโปรแกรมนี้ ซึ่งจะปรากฏขึ้นเมื่อเราเปิดโปรแกรมนี้ขึ้นมาใช้งาน ในขณะนี้จะยังไม่มีพื้นที่ในการเขียนโปรแกรม ซึ่งผู้ใช้งานจะต้องเลือกว่าจะสร้างงานใหม่หรือจะเปิดงานเก่าออกมาแก้ไขหรือใช้งานก็ได้



รูปที่ 4.1 ฟอร์มหลัก

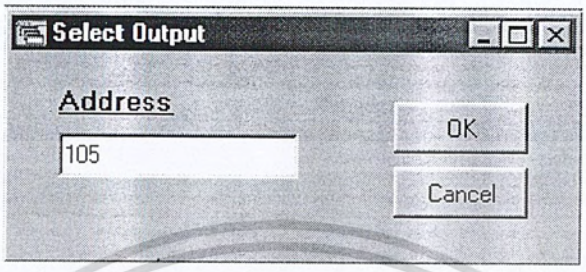
4.3.2 Tool box

Tool block เป็นฟอร์มที่จะบรรจุด้วยเครื่องมือต่าง ๆ ที่ใช้ในการเขียนโปรแกรมควบคุมที่มีไว้สำหรับผู้ใช้งานโปรแกรมนี้ ซึ่งในฟอร์ม Tool box จะประกอบด้วยเครื่องมือต่าง ๆ และหน้าที่ของเครื่องมือนั้น ดังนี้

รูปที่ 4.2 ฟอร์ม Tool box

4.3.3 ฟอร์ม Select output

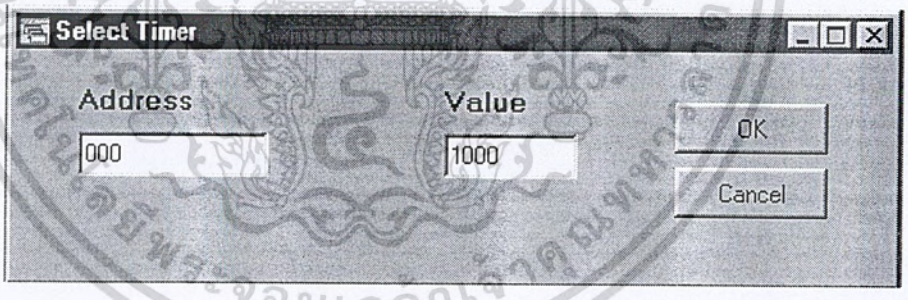
ฟอร์ม Select Output นี้จะเป็นฟอร์มที่มีไว้สำหรับผู้ใช้งานโปรแกรม เลือกตำแหน่งของเอาท์พุต ที่จะใช้งาน



รูปที่ 4.3 ฟอร์ม Select output

4.3.4 ฟอร์ม Select Timer

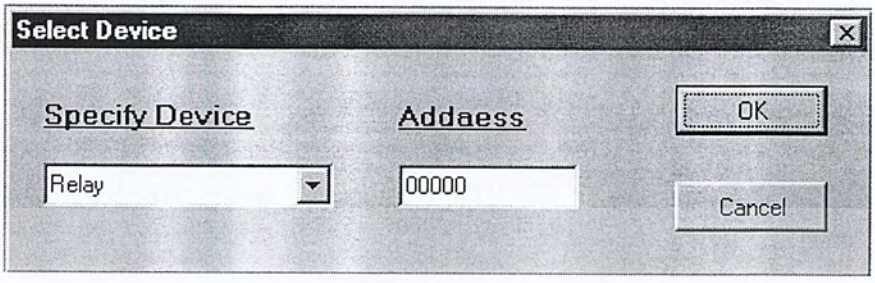
ฟอร์ม Select Timer นี้จะมีลักษณะคล้ายกับ ฟอร์ม Select Output โดยจะมีหน้าที่สำหรับให้ผู้ใช้งานเลือกตำแหน่งของตัวหน่วงเวลาที่จะใช้งานและค่าเวลาของ เวลาที่ต้องการหน่วงเวลาไว้



รูปที่ 4.4 ฟอร์ม Select Timer

4.3.5 ฟอร์ม Select Device

ในฟอร์ม Select Device นี้จะเป็นฟอร์มที่มีไว้สำหรับให้ผู้ใช้งานโปรแกรมนี้เลือกชนิดและตำแหน่งของหน้าสัมผัส (Transition) ที่จะใช้งานในการควบคุม



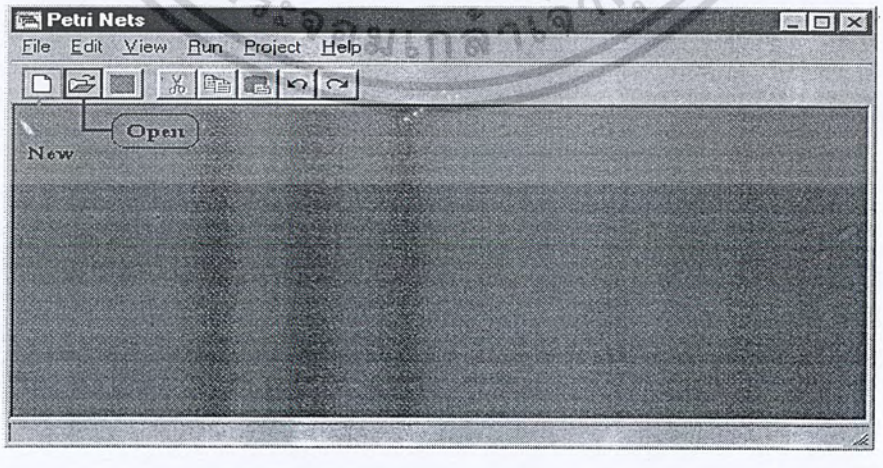
รูปที่ 4.5 ฟอร์ม Select Device

4.4 การใช้งานโปรแกรม

หลังจากที่เราได้ทราบถึงหน้าที่การทำงานและส่วนประกอบต่าง ๆ ที่สำคัญต่อการใช้งานโปรแกรมมาแล้ว ในหัวข้อนี้จะได้กล่าวถึงในเรื่องของการใช้งานโปรแกรม โดยการเขียนโปรแกรมมาควบคุมการทำงานของอุปกรณ์ภายนอกผ่าน PLC และการจำลองเหตุการณ์บน Computer

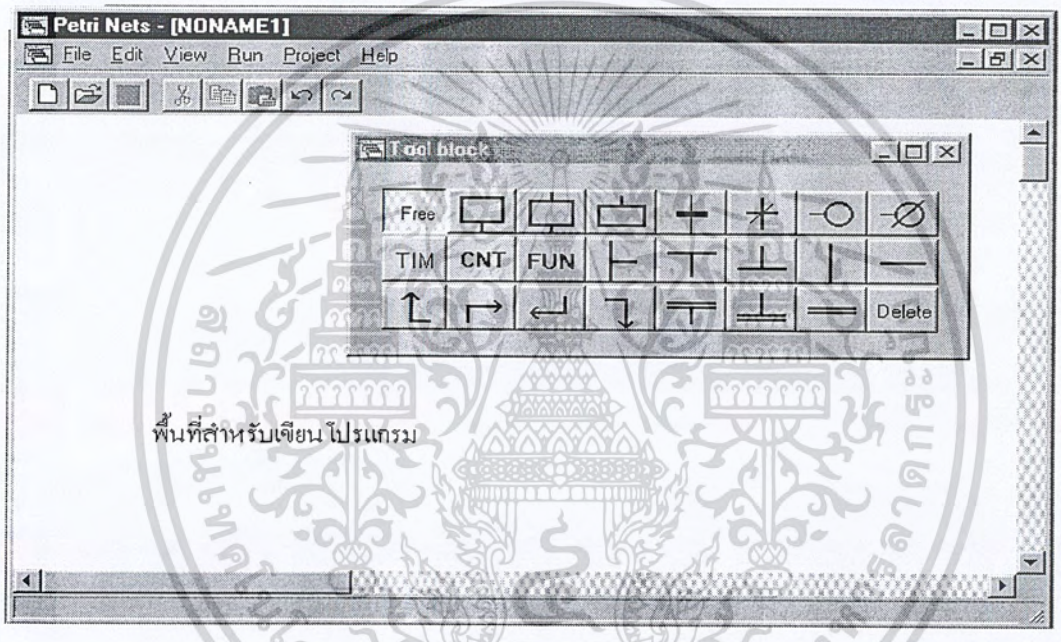
4.4.1 การเรียกโปรแกรมขึ้นมาใช้งาน

เริ่มต้นการใช้งานโดยการดับเบิ้ลคลิกที่ไอคอนของไฟล์โปรแกรม เพื่อเปิดโปรแกรมขึ้นมาใช้งาน



รูปที่ 4.6 ฟอร์มของการใช้งานเริ่มต้น

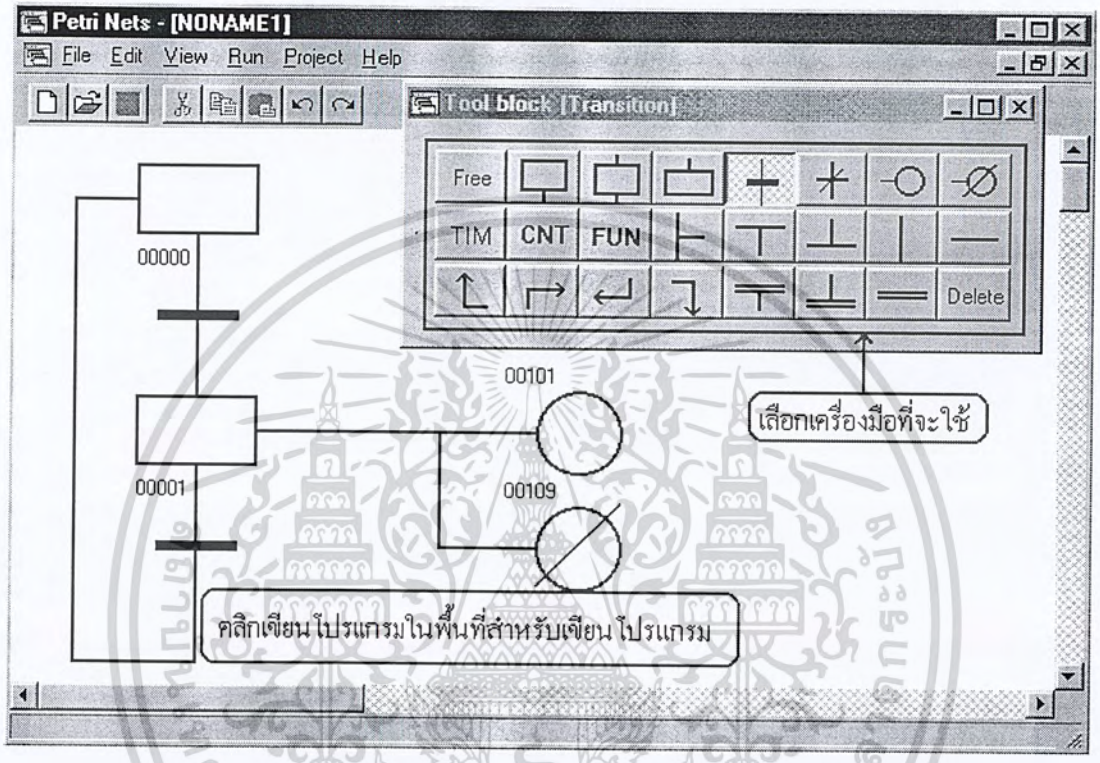
เราจะสังเกตุได้ว่าในฟอร์มเริ่มต้นเมื่อเปิดโปรแกรมขึ้นมานั้นจะยังไม่มีพื้นที่ที่จะใช้ในการเขียนหรือแก้ไข โปรแกรมควบคุมเครื่องจักรกลเลย ซึ่งเมื่อถึงขั้นตอนนี้แล้วให้ท่านเลือกว่าสร้างไฟล์ เพื่อใช้ในการเขียนโปรแกรมการควบคุมขึ้นมาใหม่ โดยการเลือกคลิกที่ปุ่ม New หรือจะเรียกจาก Menu File $\Longrightarrow$ New หรือว่าท่านจะเลือกที่จะเปิดโปรแกรมเก่าที่ได้เขียนไว้แล้วเพื่อแก้ไขหรือใช้โปรแกรมการควบคุมเก่าไปใช้เพื่อควบคุมเครื่องจักรได้โดยการคลิกที่ปุ่ม Open หรือจะเรียกจาก Menu File $\Longrightarrow$ Open



รูปที่ 4.7 ฟอร์มของโปรแกรมที่พร้อมใช้งาน

4.4.2 การเขียนโปรแกรมการควบคุม

ในการเขียนโปรแกรม หรือแก้ไขโปรแกรมการควบคุม มีหลักการคือ ใช้เมาส์คลิกที่ปุ่มเพื่อเลือกเครื่องมือต่าง ๆ ที่ต้องการใช้งาน ในฟอร์มของกล่องเครื่องมือ (Tool Box) แล้วคลิกในพื้นที่ที่ใช้ในการเขียนโปรแกรมการควบคุมตามที่ต้องการ



รูปที่ 4.8 แสดงถึงขั้นตอนการเขียนโปรแกรมการควบคุม

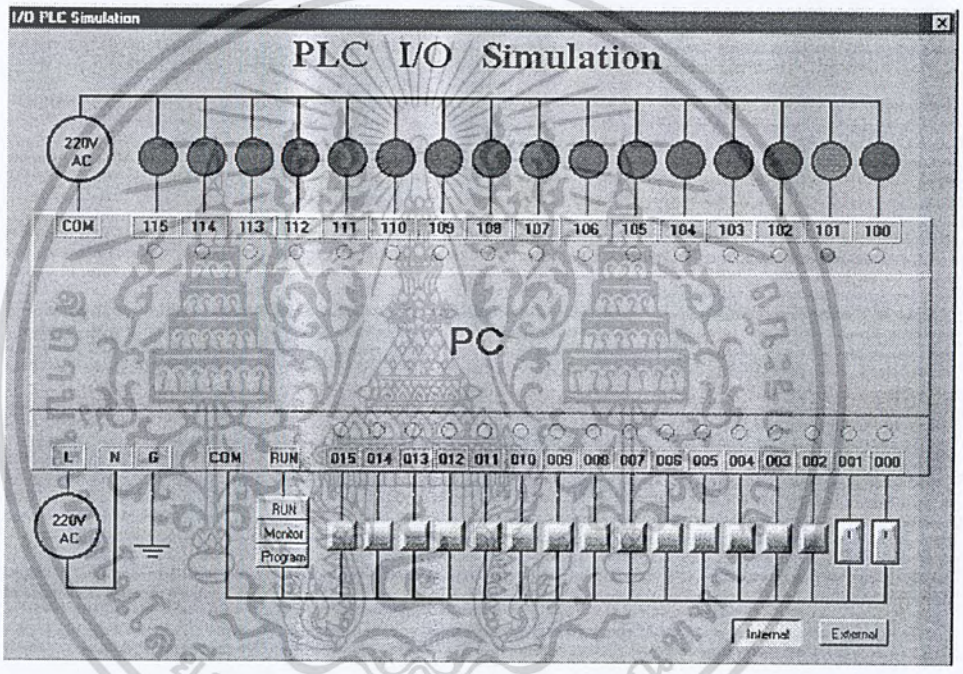
หากบนจอคอมพิวเตอร์ไม่ปรากฏฟอร์มของกล่องเครื่องมือ (Tool Box) ให้เรียกขึ้นมาโดยเลือกที่เมนู View ⟹ Show Tool Box

4.4.3 การจำลองสถานการณ์ (Simulation)

การจำลองสถานการณ์จะกระทำเพื่อจำลองเหตุการณ์ต่าง ๆ ที่เกิดขึ้น เพื่อทดสอบโปรแกรมที่เขียนขึ้นมาว่าเป็นไปตามเงื่อนไขที่ต้องการหรือไม่ จากการสั่งงานจากคอมพิวเตอร์เพื่อสร้างสัญญาณภายในคอมพิวเตอร์เท่านั้น นอกจากนี้อาจจะใช้เพื่อการเรียนรู้และฝึกฝนทักษะของการเขียนโปรแกรม ซึ่งจะสามารถกระทำได้ดังขั้นตอนต่อไปนี้

1. ทำการคอมไฟล์โปรแกรมควบคุมที่เราเขียนเสร็จและต้องการทดสอบโดยการเลือกที่เมนู Project ⟹ Compile

2. เลือกฟอร์มที่จะใช้ในการติดต่อระหว่างผู้ใช้งานโปรแกรมกับคอมพิวเตอร์โดยการเลือกที่ เมนู Simulation จากตัวอย่างนี้เราจะเลือกที่ฟอร์มของ I / O PLC simulation
3. คลิกที่ปุ่ม Internal ในฟอร์ม I / O PLC simulation เพื่อบอกว่าเราจะใช้ภายในเท่านั้น
4. สั่งโปรแกรมทำงานโดยการเลือกคลิกที่เมนู Run บนฟอร์มหลักของการใช้งาน
5. ใส่เหตุการณ์ของสวิทช์ต่าง ๆ บนฟอร์มของ I / O PLC Simulation



รูปที่ 4.9 ฟอร์ม I / O PLC Simulation

หมายเหตุ สวิตช์ในฟอร์มของฟอร์ม I / O PLC Simulation มี 2 ลักษณะ คือ

1. กดติด ปล่อยดับ
2. กดติด กดดับ

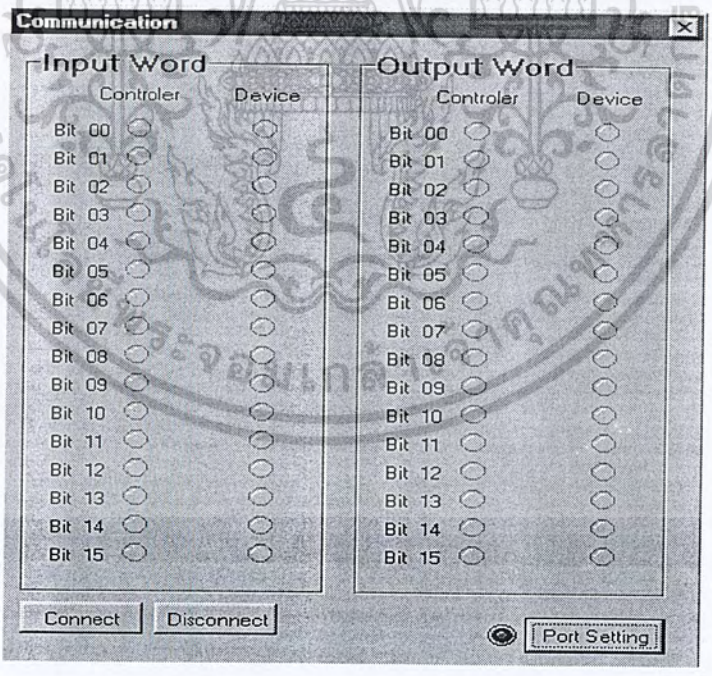
ซึ่งสามารถเปลี่ยนคุณลักษณะการทำงานของสวิตช์ ได้ดังนี้

1. คลิกขวาที่ปุ่มสวิตช์ที่ต้องการเปลี่ยนลักษณะการทำงาน
2. เลือกลักษณะการทำงานในฟอร์มที่แสดงขึ้นมา
3. เลือกกดที่ปุ่ม O.K. หากต้องการเปลี่ยน

4.4.4 การนำโปรแกรมไปควบคุมอุปกรณ์ภายนอก

การใช้โปรแกรมที่เขียนขึ้นมานี้ไปควบคุมอุปกรณ์ภายนอกให้ทำงานตามเงื่อนไข โดยการทำงานตามเงื่อนไข โดยการทำตามขั้นตอนดังต่อไปนี้

1. ต่อสายสัญญาณ PLC เพื่อใช้ในการส่งสัญญาณการควบคุมหน่วยอินพุตและหน่วย เอาท์พุต ของ PLC ซึ่งลักษณะการต่อนั้นได้กล่าวได้แล้วในบทที่ 2
2. หลังจากต่อสายสัญญาณเรียบร้อยแล้วเปิดเครื่อง PLC และเลือกสวิทช์ให้อยู่ในตำแหน่ง Program (เพราะเราจะใช้ PLC เป็นแค่ตัวส่งผ่านสัญญาณการควบคุมเท่านั้น)
3. ทำการคอมไฟล์โปรแกรมควบคุมที่เขียนขึ้นมาโดยการเลือกที่เมนู Project ⟹ Compile
4. เลือกฟอร์มที่จะใช้ในการแสดงผลการทำงานของโปรแกรม โดยการเลือกที่เมนู Simulation จากตัวอย่างนี้จะเลือกที่ฟอร์มของ I / O PLC Simulation
5. เลือก External ในฟอร์มของ I/O PLC Simulation เพื่อใช้ในการแสดงผลการควบคุม (ในกรณีนี้จะไม่สามารถสั่งวานบนคอมพิวเตอร์ได้)
6. สั่งงานให้คอมพิวเตอร์ติดต่อกับ PLC โดยการเลือกที่เมนู Communication หลังจากนั้นเลือกกดที่ปุ่ม connct บนฟอร์ม Communication
7. สั่งให้โปรแกรมทำงาน โดยการเลือกที่เมนู Ran บนฟอร์ม หลักของการใช้งาน

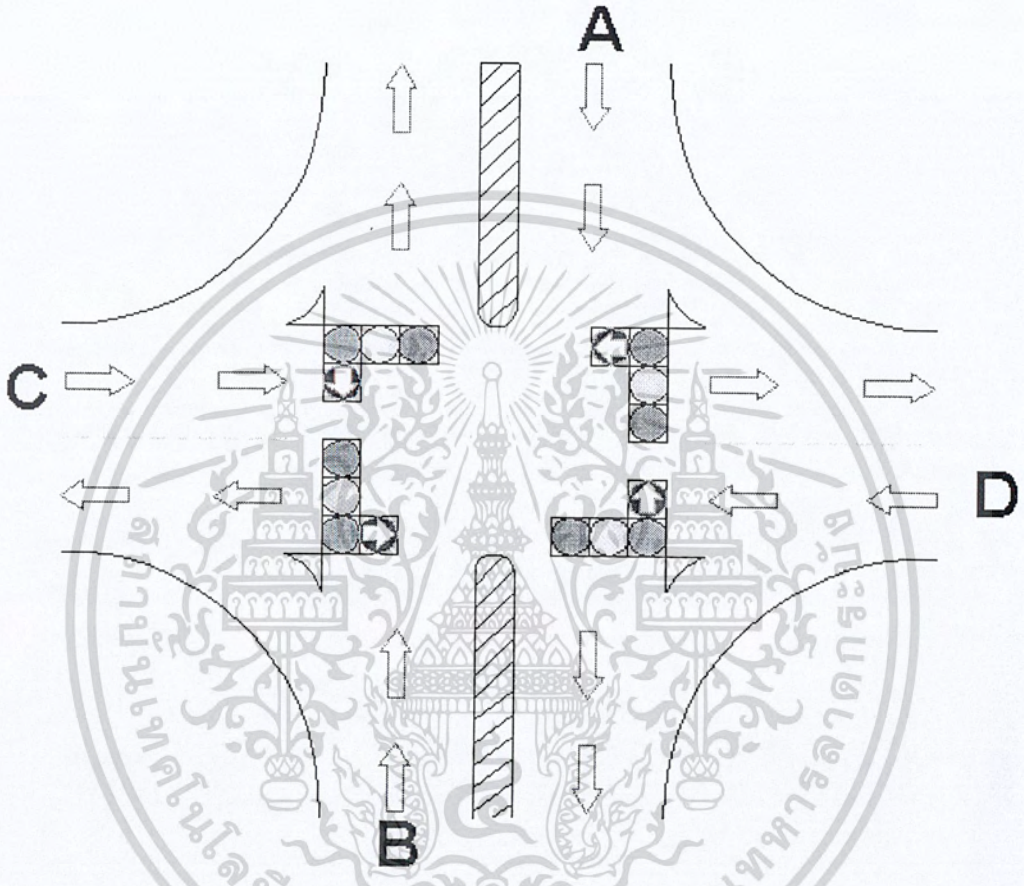


รูปที่ 4.10 ฟอร์มการติดต่อสื่อสาร

หมายเหตุ เราอาจจะใช้ฟอร์มของการติดต่อสื่อสารเพื่อดูการทำงานของโปรแกรมก็ได้

บทที่ 5
ตัวอย่างวงจรควบคุม

5.1 ไฟจราจร

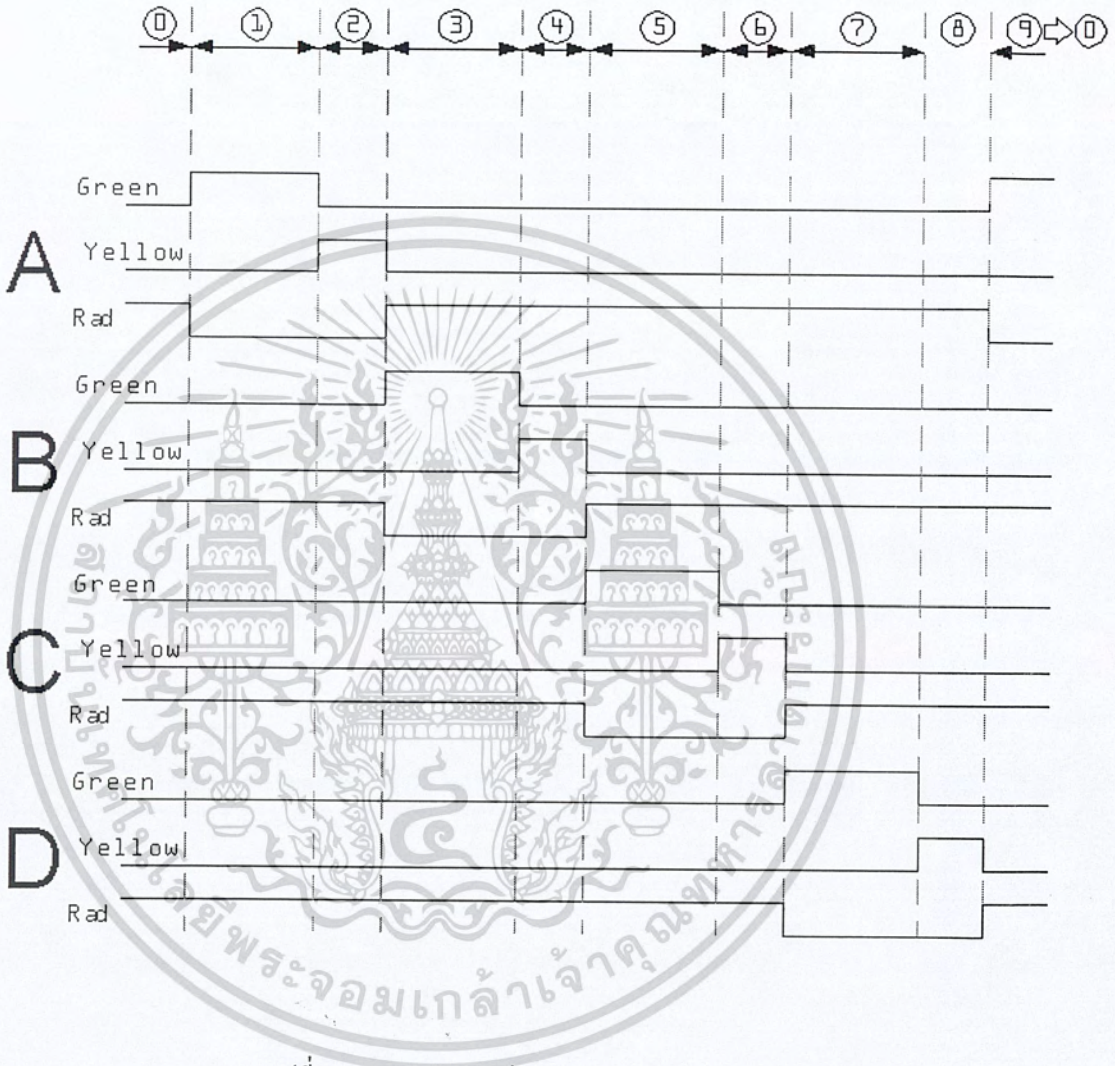


รูปที่ 5.1 ไฟจราจร

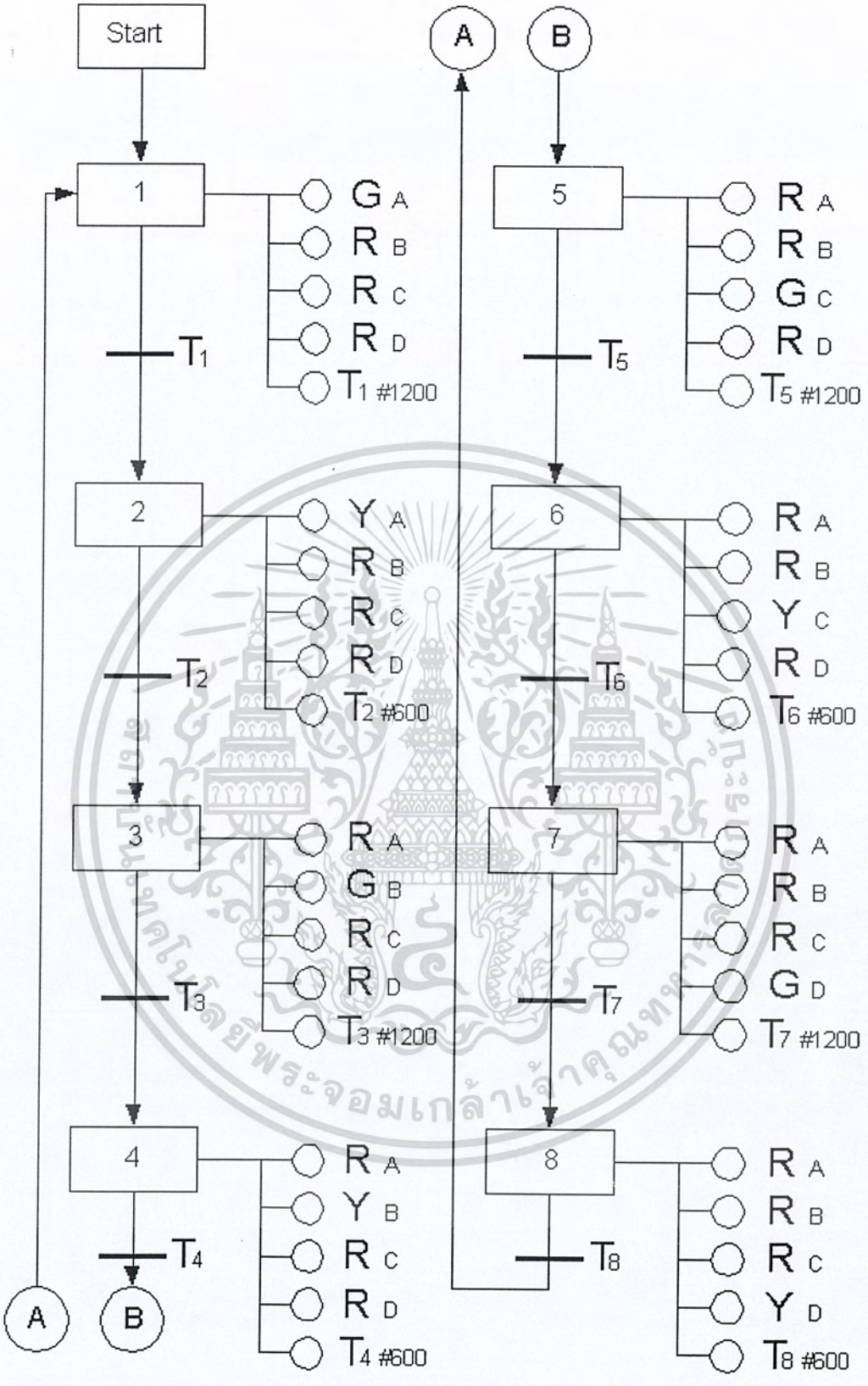
เงื่อนไขการทำงาน

1. เริ่มต้นการทำงานของสัญญาณไฟจราจร ปล่อยรถด้าน A ทางตรงและเลี้ยวขวานาน 3 นาที
2. หลังจากนั้น ปล่อยรถด้าน B ทางตรงและเลี้ยวขวานาน 3 นาที
3. หลังจากนั้น ปล่อยรถด้าน C ทางตรงและเลี้ยวขวานาน 3 นาที
4. หลังจากนั้น ปล่อยรถด้าน D ทางตรงและเลี้ยวขวานาน 3 นาที
5. ย้อนกลับมาเริ่มต้นการทำงานรอบใหม่

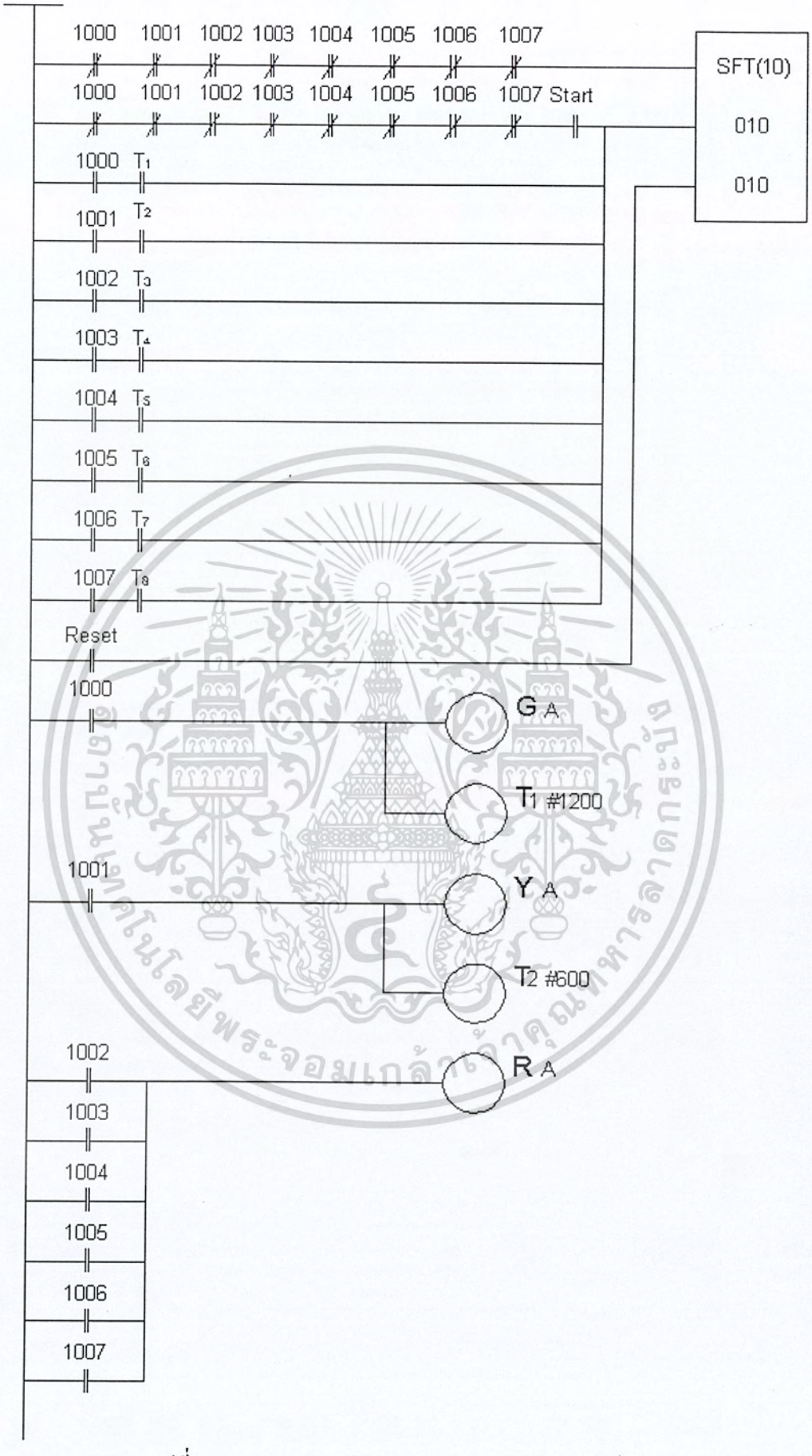
6. ช่วงระหว่างการเปลี่ยนสัญญาณไฟจราจรจากไฟเขียวเป็นไฟแดงให้เตือนด้วยไฟเหลือง นาน 2 นาที
7. รถทุกคันที่เลี้ยวซ้ายวิ่งผ่านตลอดเวลา



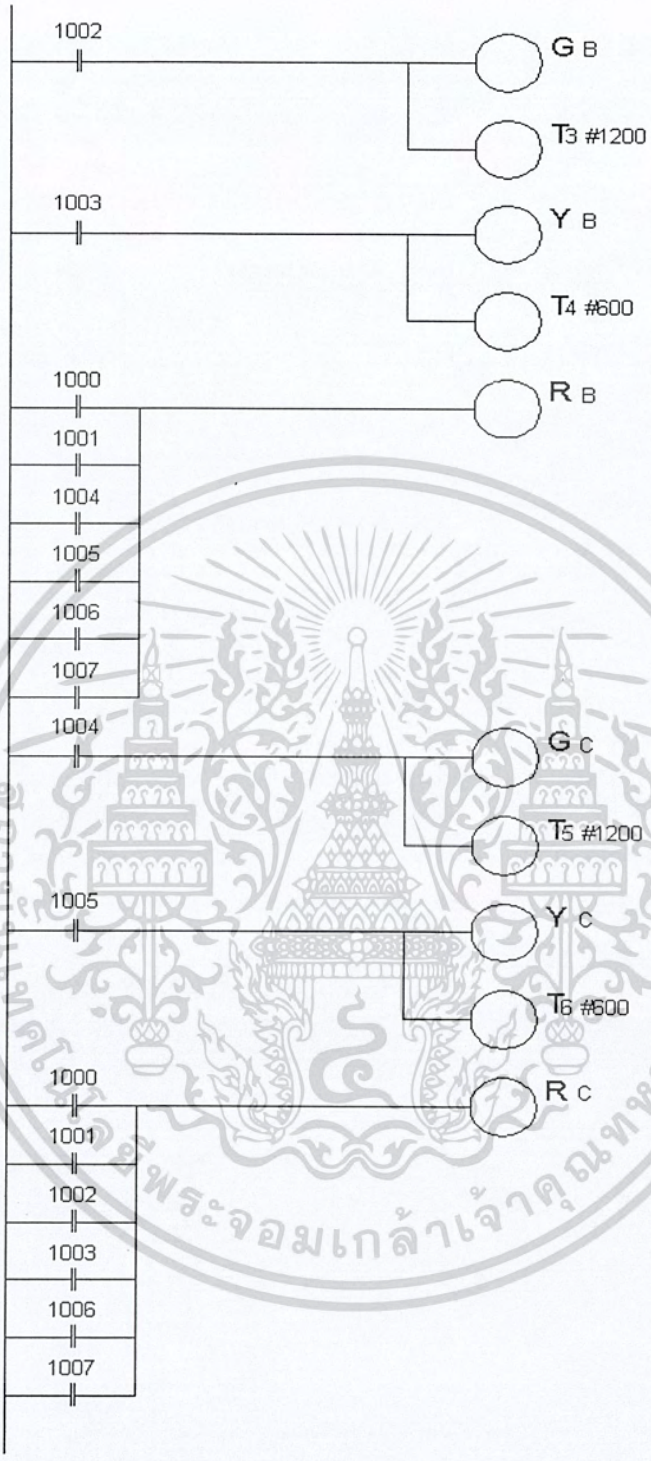
รูปที่ 5.2 แผนผังเวลาไฟจราจร



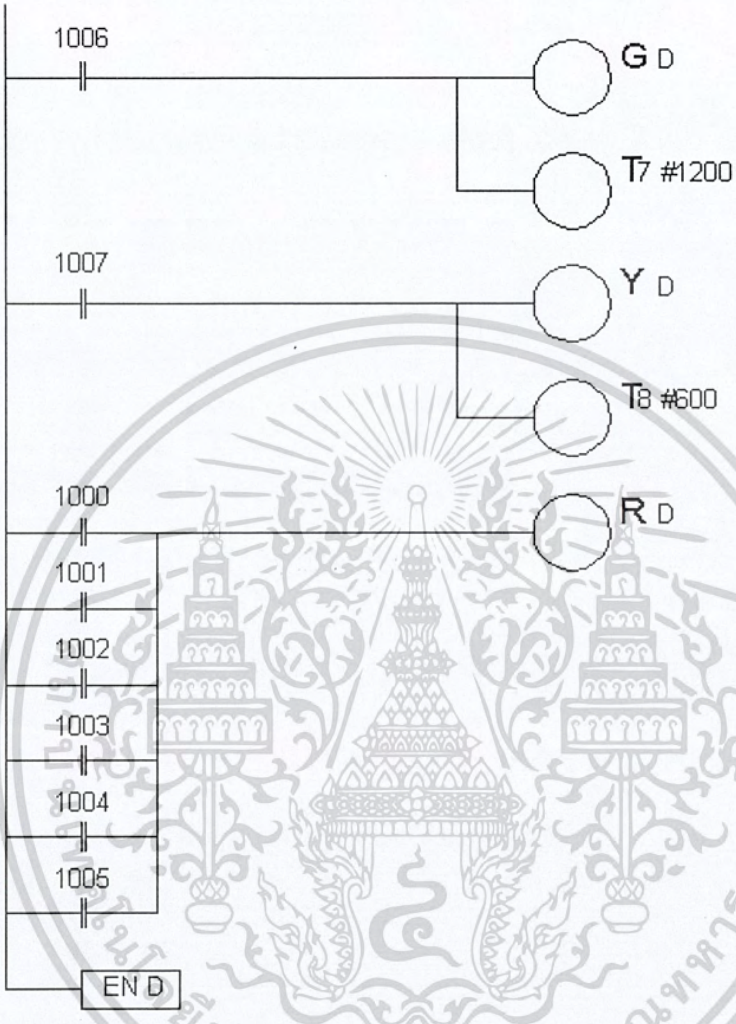
รูปที่ 5.3 วงจรควบคุมไฟจราจรในรูปแบบของเพททริเนทส์



รูปที่ 5.4 วงจรควบคุมในรูปแบบของวงจรแลดเดอร์

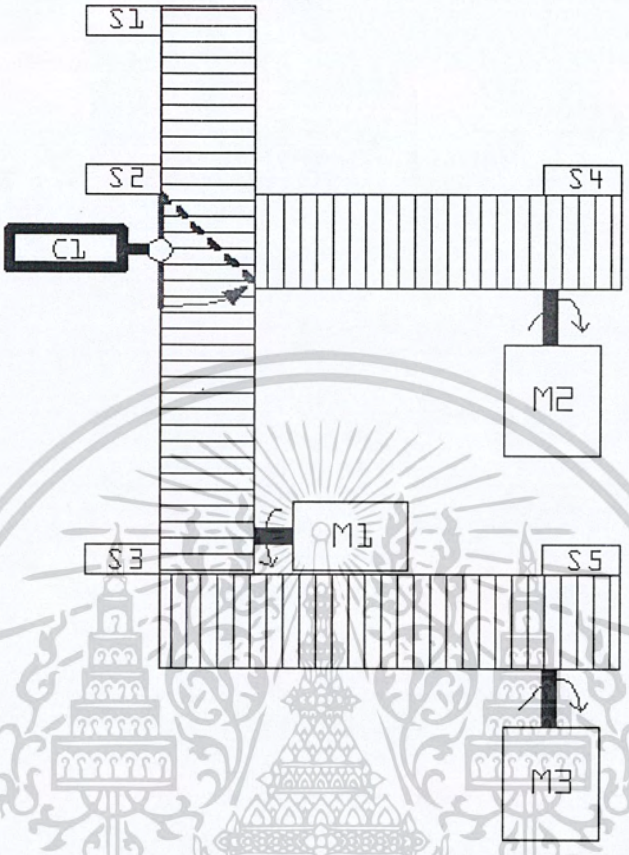


รูปที่ 5.4 ต่อ วงจรควบคุมในรูปแบบของวงจรแลดเดอร์



รูปที่ 5.4 ต่อ วงจรควบคุมในรูปแบบของวงจรแลดเดอร์

5.2 ระบบคัดแยกโลหะ



รูปที่ 5.5 ระบบคัดแยกโลหะ

S1, S3, S5 คือ เซนเซอร์แบบคาปาซิทีฟ

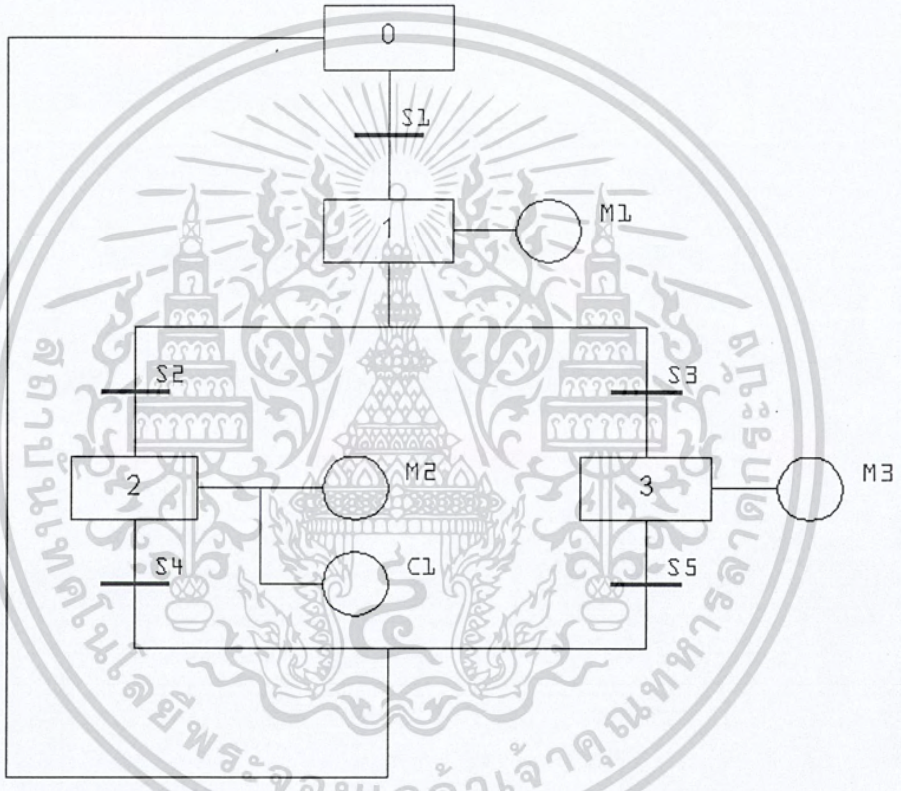
S2, S3 คือ เซนเซอร์แบบอินดักทีฟ

M1, M2, M3 คือ มอเตอร์ขับสายพานลำเลียง

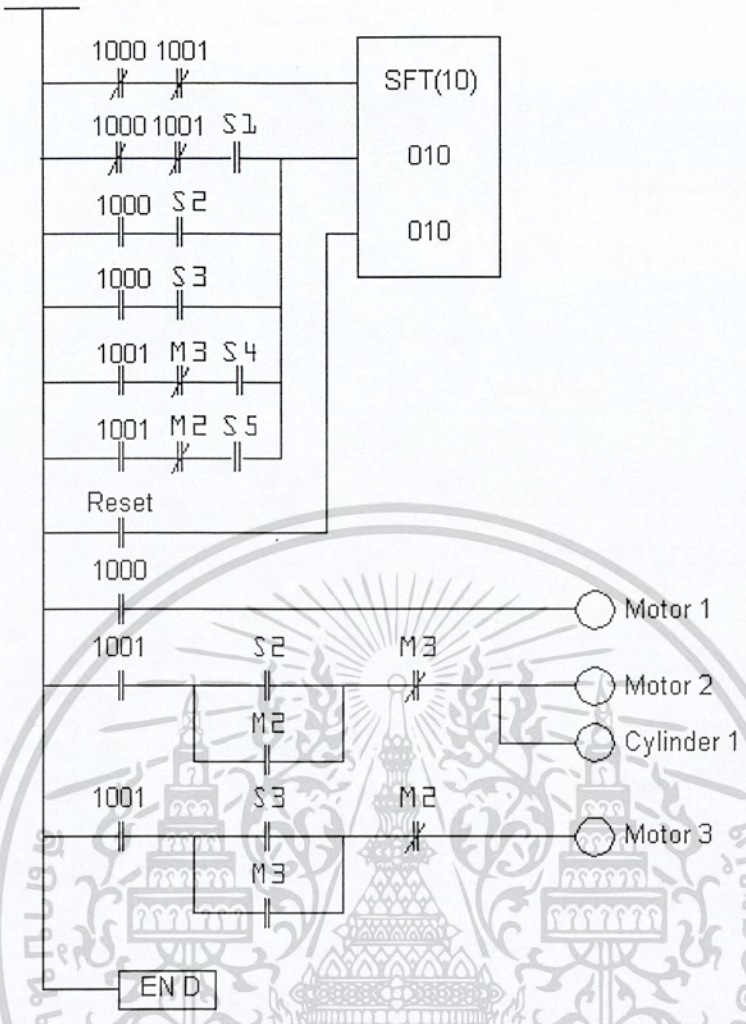
C1 คือ กระบอกสูบสำหรับผลักชิ้นงาน

เงื่อนไขการทำงาน

1. M1 ทำงานเมื่อ S1 ตรวจจับได้ว่ามีชิ้นงานเข้ามา
2. C1 , M2 ทำงานเมื่อ S2 ตรวจจับได้ว่าชิ้นงานนั้นเป็นโลหะ
3. M3 ทำงานเมื่อ S3 ตรวจจับได้ว่ามีชิ้นงานเข้ามา (ผ่าน S2 มาได้แสดงว่าเป็นอโลหะ)
4. M2 หยุดทำงานเมื่อ S4 ตรวจจับได้ว่ามีชิ้นงานไหลผ่านไปแล้ว
5. M3 หยุดทำงานเมื่อ S5 ตรวจจับได้ว่ามีชิ้นงานไหลผ่านไปแล้ว



รูปที่ 5.6 วงจรควบคุมระบบคัดแยกโลหะในรูปแบบของเพททริเนทส์



รูปที่ 5.7 วงจรควบคุมระบบคัดแยกโลหะในรูปแบบของวงจรแลดเดอร์

บทที่ 6
บทวิจารณ์และสรุป

6.1 สรุปผลงาน

โดยสรุปแล้วผลที่ได้จากการศึกษาเรื่องเพทททริเนทส์ และความสามารถของโปรแกรมที่ได้เขียนขึ้นมาพอที่จะสรุปได้ดังนี้

1. เขียนเงื่อนไขการทำงานของระบบการควบคุมการทำงานต่าง ๆ ให้อยู่ในรูปแบบของแพททริเนทส์ และสามารถแปลงรูปแบบของเพททริเนทส์ให้เป็นรูปแบบของวงจรแลดเดอร์เพื่อใช้สำหรับการโปรแกรมการทำงานให้กับ PLC เพื่อใช้ในการควบคุมเครื่องจักรกลและอุปกรณ์ต่าง ๆ ภายนอกและสามารถที่จะใช้ตรวจสอบข้อบกพร่องของการทำงานของระบบควบคุมเครื่องจักรที่ได้ออกแบบไว้ได้อีกด้วย
2. สามารถป้อนโปรแกรมในรูปแบบของ Grafcet ที่เขียนขึ้นมาลงบนโปรแกรมที่ได้เขียนขึ้นมารองรับเพื่อใช้ในการควบคุมเครื่องจักรกลต่อไป โดยในจะทำการควบคุมผ่านอินพุตโมดุลและเอาท์พุตโมดุล ของ PLC ยี่ห้อออมร่อน
3. ในส่วนของโปรแกรมนั้นนอกจากจะสามารถควบคุมเครื่องจักรกลได้แล้วยังสามารถที่จะจำลองเหตุการณ์ต่าง ๆ บนคอมพิวเตอร์เพื่อตรวจสอบก่อนการไปใช้งานจริงได้อีกด้วย อีกทั้งสามารถตรวจสอบหา deadlock ของโปรแกรม ซึ่งก่อให้เกิดปัญหาในการควบคุมได้อีกด้วย
4. สามารถที่จะเก็บบันทึกโปรแกรมที่เขียนขึ้นมาไว้ได้ ดังนั้นการแก้ไขโปรแกรมจะสามารถทำได้ง่าย โดยการจัดเก็บโปรแกรมที่เขียนขึ้นมานี้จะจัดเก็บในรูปแบบของตัวอักษร

6.2 แนวทางการพัฒนาต่อ

เนื่องจากแพททริเนทส์ยังมีส่วนที่ให้ศึกษาต่อไปอีกมากมายนัก ดังนั้นโปรแกรมที่สร้างขึ้นมานี้สามารถพัฒนาต่อไปได้อีก

1. โปรแกรมนี้สามารถทำงานในการควบคุมได้ในระดับ basic control เท่านั้น ยังไม่สามารถนำเรื่องของความน่าจะเป็นของแต่ละ arc เข้ามาเกี่ยวข้องได้ ซึ่งถ้าพัฒนาเกี่ยวกับความน่าจะเป็นของแต่ละ arc ที่ใช้ในการส่งผ่าน Token ได้ก็จะทำให้นำไปสู่การควบคุมแบบอนาล็อคได้ในที่สุด

2. เมื่อผู้ใช้งานเขียนโปรแกรม และทำการจำลองหรือรันโปรแกรมที่ได้เขียนขึ้นมาแล้วนั้นสมควรที่จะมี Timing diagram เพื่อใช้ในการตรวจสอบและวิเคราะห์การทำงานของโปรแกรมการทำงานที่ผู้ใช้งานเขียนขึ้นมา
3. ในโปรแกรมนั้นเขียนอยู่ในรูปของแพททริเนทส์ เพื่อนำไปใช้ในการป้อนโปรแกรมให้กับ PLC ให้ง่ายขึ้นควรมีฟังก์ชั่นในการแปลงจากแพททริเนทส์ เป็น Statement List และการประมวลผลยังช้าอยู่เนื่องด้วยในฟังก์ชั่นการรันนั้นทำงานเป็นลูปในรูปแบบของภาษาปาสคาลอยู่ ซึ่งมีช่วงเวลา scan time ที่นานจึงทำให้ผู้ตอบสนองช้า ดังนั้นจึงควรปรับปรุงในเรื่องนี้

6.3 สรุปผลงานโดยรวม

ตั้งแต่เริ่มทำงานจนถึงปัจจุบันที่ได้สร้างแอพพลิเคชั่นในรูปแบบของโปรแกรมทำให้เราได้ความรู้ความสามารถดังนี้

1. ความรู้ด้านแพททริเนทส์ ได้เรียนรู้การจำลองระบบด้วยโมเดลของแพททริเนทส์ ทฤษฎีการทำงาน ศึกษาพฤติกรรมของระบบจากโมเดลที่จำลองขึ้นมารวมถึงแนวทางในการนำเพททริเนทส์ไปใช้ในการวิเคราะห์ตรวจสอบระบบการทำงานหรือระบบการผลิตของเครื่องจักรต่าง ๆ
2. ความรู้ทางด้านการเขียนโปรแกรมด้วย โปรแกรมเดลไฟ ทักษะการเขียนโปรแกรมแบบวิชวล การเขียนโปรแกรมเชิงวัตถุ (Object Oriented Program) ซึ่งมาจากพื้นฐานของภาษาปาสคาล
3. ความรู้ทางด้านเขียนภาพเคลื่อนไหว โดยใช้โปรแกรม 3D Studio max เพื่อใช้จำลองการทำงานของเครื่องจักร
4. ได้ตระหนักและเรียนรู้ถึงการวางแผนงานวิเคราะห์ปัญหาก่อนที่จะลงมือทำงานจริงซึ่งเป็นสิ่งสำคัญมากในการทำงานให้มีประสิทธิภาพ
5. ความรู้ทางด้านการติดต่อสื่อสารระหว่างเครื่องควบคุมแบบตรรกะที่โปรแกรมได้กับคอมพิวเตอร์
6. ได้ฝึกทักษะในการแก้ปัญหาที่เกิดขึ้นในขณะทำงาน

บรรณานุกรม

1. นิพนธ์ เรืองวิริยะนันท์ "เอกสารประกอบการสอน วิชาโปรแกรมเมเบิ้ลคอนโทรลเลอร์" แผนกวิชาช่างไฟฟ้ากำลัง วิทยาเขตตาก
2. ก่อกิจ วิระอาชากุล , บุญชัย เอี่ยมเมตตา "แพททริเนทส์ Simulation Tool" ปริญญานิพนธ์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง , 2540
3. สัจจะ จรัสรุ่งรวีวร, จักรพงษ์ สุขประเสริฐ "เริ่มต้นอย่างมืออาชีพด้วย Delphi" สำนักพิมพ์ อินโฟเพรส 200 หมู่ 4 ห้อง 508 อาคารจัสมินอินเตอร์เนชันแนลทาวเวอร์ ชั้น 5 ถ.แจ้งวัฒนะ อ.ปากเกร็ด จ.นนทบุรี , 2543
4. Alan A. Desrochers , Robert Y. Al-Jaar "Application of แพททริเนทส์ in Manufacturing System" The institute of Electrical and Electronic Engineers , Inc , New York

   Rene David , Hassane Alla "แพททริเนทส์ and Grafcet" University Press, cambridge,1992