



ปีการศึกษา 2533

เรื่อง

การควบคุมลิฟต์โดยใช้ไมโครโปรเซสเซอร์
(Microprocessor Elevator Controller)

โดย

นาย ทวีพงษ์ ลอยวัฒนาเรืองทิติ 30.1090

นาย เกอศกุน วีระสาสน์ 30.1082

อาจารย์ที่ปรึกษา

อาจารย์ ขนิษฐา แซ่ตั้ง

028832

16.ค.ค.2534

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญาโทบริหารการศึกษา 2533

ภาควิชา อีเล็คตรอนิกส์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหารลาดกระบัง

เรื่อง การควบคุมลิฟต์โดยใช้ไมโครโปรเซสเซอร์

(Microprocessor Elevator Controller)

ผู้จัดทำ

1. นาย ทวีพงษ์ ลอรัตนเรืองกิจ 30.1080

2. นาย เทอดคุณ วีระสาสน์ 30.1082

.....อาจารย์ที่ปรึกษา
(อาจารย์ วัฒนา แซ่ตั้ง)

028832

การควบคุมลิฟต์โดยไฟไหม้โครโปรเซสเซอร์

นาย ทวีพงษ์ ลอรัตนเรืองกิต 301080

นาย เทอดทูน ชีระสาสน์ 301082

อาจารย์ ชนิษฐา แซ่ตั้ง อาจารย์ที่ปรึกษา

ปีการศึกษา 2533

บทคัดย่อ

ปริญญานิพนธ์นี้ เป็นการพัฒนาชุดควบคุมสำหรับใช้ควบคุมการทำงานของลิฟต์ โดย
ทำการพัฒนาทั้งทางด้านฮาร์ดแวร์ และ ซอฟต์แวร์ ในส่วนของฮาร์ดแวร์นั้นจะประกอบด้วย
ซีพียูบอร์ด เมตริกซ์บอร์ด และ อินพุต เอาท์พุทบอร์ด ซีพียูที่ใช้เป็นซีพียูเบอร์ 8085 ซึ่งได้
ทำการออกแบบให้มีความคล่องตัว และ มีความยืดหยุ่นสูง สามารถที่จะทำการเพิ่มเติมการ
ควบคุมจำนวนชั้นของลิฟต์ได้โดยการเพิ่มแผงวงจรของอินพุต และ เอาท์พุทบอร์ด เข้าไปใน
ชุดควบคุมโดยไม่ต้องทำการเปลี่ยนแปลง หรือ แก้ไขฮาร์ดแวร์ในส่วนอื่น ๆ ทั้งนี้เนื่องจาก
ได้ออกแบบให้ส่วนของการควบคุมแอดเดรสของอินพุต และ เอาท์พุท แยกไปอยู่ที่อินพุต และ
เอาท์พุทบอร์ด ต่างจากวงจรของซีพียูบอร์ดทั่วไปที่มีวงจรเลือกแอดเดรสของอินพุต เอาท์พุท
อยู่บนบอร์ดเดียวกัน สำหรับในส่วน of ซอฟต์แวร์นั้นได้ออกแบบให้ลิฟต์มีการทำงานเป็นลิฟต์
โดยสารเดี่ยว แบบที่มีประตูเพียงด้านเดียว สามารถที่จะใช้ควบคุมลิฟต์ที่มีจำนวนชั้นสูงสุด
ได้ไม่เกิน 64 ชั้น และ สามารถที่จะดัดแปลงให้ใช้กับลิฟต์โดยสารเดี่ยว แบบที่มีประตูสอง
ด้าน และมีจำนวนชั้นสูงสุดไม่เกิน 32 ชั้นได้

Microprocessor Elevator Controller

Mr.Taweepong Lorrattanaruengkit 30.1080

Mr.Therdtoon Theerasasana 30.1082

Ms.Khanittha Saetang Advisor

Academic Year 1990

Abstract

This project presents the development of controller board for simplex elevator control. This development deals with the hardware design and software design. The hardware is separated into three parts, microprocessor board, matrix board and input output board. The hardware design of input and output board with address port controller make this system flexible and expandable. This system can be added more input and output boards without changing on microprocessor board. The software design in this project can be used for control the simplex 64 floors elevator which has only one door. And this software may be improved to control 32 floors-elevator which has dual door without the modification of hardware.

สารบัญ

1. บทนำ.....	1
2. ทฤษฎี.....	3
2.1 ประวัติความเป็นมาของลิฟต์.....	3
2.2 นิยามของลิฟต์และประเภทของลิฟต์.....	3
2.3 โครงสร้างและองค์ประกอบของลิฟต์.....	4
2.4 กลอุกรณ์ความปลอดภัย.....	5
2.5 กลอุกรณ์ของสัญญาณต่าง ๆ ที่ใช้ลิฟต์.....	7
2.6 เครื่องขับลิฟต์.....	8
2.7 การควบคุมลิฟต์.....	9
2.8 การทำงานของลิฟต์.....	10
2.9 ลิฟต์รีเลย์และลิฟต์ไมโครคอมพิวเตอร์.....	13
2.9.1 ลิฟต์รีเลย์.....	13
2.9.2 ลิฟต์ไมโครคอมพิวเตอร์.....	25
3. การออกแบบ.....	32
3.1 รายละเอียดของการทำงานในส่วนฮาร์ดแวร์.....	32
3.1.1 แผงวงจรดิจิทัล.....	32
3.1.2 แผงวงจรอินพุต และ เอาท์พุต.....	33
3.1.3 แผงวงจรเมตริกซ์.....	34
3.2 รายละเอียดของการทำงานในส่วนซอฟต์แวร์.....	38
4. การทดสอบ และ ผลการทดสอบ.....	58
5. สรุปผลการทดสอบ.....	60
ภาคผนวก	
กิตติกรรมประกาศ	
เอกสารอ้างอิง	

สารบัญรูปภาพ

รูปที่ 2.1	บล็อกไดอะแกรมของลิฟต์รีเลย์.....	14
รูปที่ 2.2 ถึง รูปที่ 2.15	แสดงการทำงานของลิฟต์รีเลย์.....	20-24
รูปที่ 2.16	บล็อกไดอะแกรมของลิฟต์ไมโครคอมพิวเตอรื.....	28
รูปที่ 2.17	การออกแบบระบบลิฟต์ของ KONE.....	29
รูปที่ 2.18	การสื่อสารของระบบลิฟต์ KONE.....	30
รูปที่ 2.19	การออกแบบระบบลิฟต์ของญี่ปุ่น.....	30
รูปที่ 3.1	วงจรของซีพียูบอร์ด.....	35
รูปที่ 3.2	วงจรของอินพุตและเอาต์พุตบอร์ด.....	36
รูปที่ 3.3	วงจรของเมมทริกซ์บอร์ด.....	37
รูปที่ 3.4	ผังงานของโปรแกรมหลัก.....	50
รูปที่ 3.5	ผังงานของโปรแกรมน้อยทั้งสถานะเริ่มต้น.....	51
รูปที่ 3.6	ผังงานของโปรแกรมน้อยการเข้าจอดและความคุมประตู.....	52
รูปที่ 3.7	ผังงานของโปรแกรมน้อยกำหนดทิศทางการเคลื่อนที่.....	53
รูปที่ 3.8	ผังงานของโปรแกรมน้อยลิฟต์เคลื่อนที่ขึ้น.....	54
รูปที่ 3.9	ผังงานของโปรแกรมน้อยลิฟต์เคลื่อนที่ลง.....	55
รูปที่ 3.10	ผังงานของโปรแกรมน้อยอ่านสัญญาณเรียกจากชั้นที่สูงกว่า.....	56
รูปที่ 3.11	ผังงานของโปรแกรมน้อยอ่านสัญญาณเรียกจากชั้นที่ต่ำกว่า.....	57
รูปที่ 3.12	ผังงานของโปรแกรมน้อยตรวจสอบการเข้าจอดที่ระดับชั้น.....	58

สารบัญตาราง

ตารางที่ 2.1	แสดงข้อเปรียบเทียบของเครื่องควบคุมลิฟต์แบบรีเลย์กับ เครื่องควบคุมลิฟต์แบบไมโครคอมพิวเตอร์.....	26
ตารางที่ 2.2	ข้อเปรียบเทียบของลิฟต์ไมโครคอมพิวเตอร์ที่ได้ศึกษา.....	31



1 - บทนำ

ในปัจจุบัน ประเทศไทย สามารถผลิตลิฟต์โดยสารขึ้นใช้เองภายในประเทศได้แล้ว แต่ในส่วนควบคุมลิฟต์ยังต้องใช่วางจรซึ่งประกอบด้วยรีเลย์อยู่ ในขณะที่ส่วนควบคุมลิฟต์จากต่างประเทศ ได้ใช้เทคโนโลยีทางด้านไมโครคอมพิวเตอร์ การที่ส่วนควบคุมลิฟต์มีการนำเอาไมโครโปรเซสเซอร์ เข้ามาใช้ในการควบคุมแทนรีเลย์ เพราะไมโครโปรเซสเซอร์ สามารถทำงานได้อย่างมีประสิทธิภาพมากกว่า และจะทำให้ขนาดของส่วนควบคุมเล็กลง ตลอดจนประหยัดพลังงานไฟฟ้ามากกว่า การซ่อมแซมและแก้ไข สามารถทำได้สะดวกและรวดเร็วกว่าเก่า ด้วยเหตุนี้ ไมโครโปรเซสเซอร์จึงมีบทบาทสำคัญในการนำมาใช้ควบคุมการทำงานของลิฟต์

ในขณะนี้กรุงเทพมหานคร และตามเมืองใหญ่ ๆ จะพบอาคารที่สูงกว่าสิบชั้นเป็นจำนวนมาก การควบคุมลิฟต์ที่สูงเกินกว่าสิบชั้นนั้น ถ้าใช้ส่วนควบคุมลิฟต์ซึ่งประกอบด้วยวางจรรีเลย์ ตัวควบคุมจะต้องใช้รีเลย์เป็นจำนวนมาก ทำให้เสียเวลาในการเดินสาย และต่อวงจรของรีเลย์เหล่านี้ ในขณะที่เครื่องควบคุมแบบที่เอาไมโครโปรเซสเซอร์เข้ามาใช้เพียงแต่เพิ่มแผงวงจรที่เป็นส่วนของอินพุต และเอาท์พุต เท่านั้น นอกจากนี้อาคารที่มีความสูงมากกว่าสิบชั้น จะมีความต้องการการใช้ลิฟต์มากกว่าหนึ่งตัวเสมอ ถ้ามีการออกแบบให้ลิฟต์แต่ละตัว เป็นลิฟต์ในระบบ simplex จะทำให้เกิดการสิ้นเปลืองพลังงานไฟฟ้า ทั้งนี้เนื่องจากผู้ใช้ลิฟต์ส่วนใหญ่จะมีการกดสัญญาณเรียกใช้ลิฟต์ทุกตัวในขณะเดียวกัน ทำให้ลิฟต์ทุกตัวเคลื่อนที่มายังชั้นที่มีการเรียก ลิฟต์ตัวที่มาถึงเร็วที่สุด จะรับผู้โดยสารในชั้นนั้นไป โดยจะทำให้เสียประโยชน์สำหรับลิฟต์ตัวอื่น ๆ เป็นการสิ้นเปลืองพลังงานไฟฟ้าโดยใช่เหตุ แต่ถ้าเป็นการออกแบบส่วนควบคุม โดยใช้ไมโครโปรเซสเซอร์ในระบบ duplex สามารถที่จะใช้แผงควบคุมที่หน้าประตูลิฟต์ในแต่ละชั้นเพียงแผงเดียว ซึ่งอาจจะควบคุมลิฟต์ที่ตัวก็ได้ เมื่อมีการเรียกใช้ลิฟต์ ไมโครโปรเซสเซอร์จะทำการประมวลผล และควบคุมลิฟต์ตัวที่อยู่ใกล้ที่สุด ให้ทำงานในขณะนั้น ซึ่งจะทำให้เกิดการประหยัดพลังงานไฟฟ้าเป็นอย่างมาก สำหรับลิฟต์ในระบบ duplex ถ้าใช่วางจรรูปแบบที่เป็นรีเลย์นั้น จะต้องออกแบบวงจรที่มีความซับซ้อนมากและจะทำให้ตัวควบคุมมีขนาดใหญ่ตามไปด้วย

สำหรับโครงงานนี้ ได้มีการนำเอาไมโครโปรเซสเซอร์ โดยมีการออกแบบให้ใช้ความถี่ในระบบ simplex ในกรณีที่ผลการทดสอบเป็นที่น่าพอใจ ก็จะได้ทำการออกแบบเพื่อให้สามารถใช้งานในระบบ duplex ต่อไป



2 - ทฤษฎี

2.1 ประวัติความเป็นมาของลิฟท์

การขนส่งในแนวตั้งมีมาตั้งแต่สมัยโรมัน หลักฐานยืนยันได้แก่ลูกโลก และรางในแนวตั้ง (Vertical Rails) ซึ่งเป็นองค์ประกอบของลิฟท์ในปัจจุบัน ในซีเบตได้มีการขนส่งคนขึ้นภูเขาโดยใช้ตะกร้าผูกเชือกคล้องผ่านลูกรอกแล้วใช้แรงคนหรือสัตว์ดึงตะกร้านั้นขึ้นไป ลิฟท์ได้ถือกำเนิดราวศตวรรษที่ 18 และในต้นศตวรรษที่ 19 ได้มีการนำพลังเครื่องจักรไอน้ำมาขับเคลื่อนลิฟท์ และต่อมาได้นำเครื่องจักรไฟฟ้ามาขับเคลื่อนลิฟท์ แต่การใช้ลิฟท์ในสมัยนั้นไม่นิยมนัก เพราะเกิดโศกนาฏกรรมขึ้นบ่อยเนื่องจากเชือกขาดและไม่สามารถหยุดลิฟท์ได้ จนกระทั่งในปี ค.ศ.1853 Elisha Graves Otis ได้ประดิษฐ์อุปกรณ์สำหรับความปลอดภัยของลิฟท์ขึ้น เพื่อป้องกันการตกอย่างอิสระเนื่องจากเชือกขาด อุปกรณ์สำหรับความปลอดภัยที่ Otis คิดขึ้นนี้ ได้รับการยอมรับในปี ค.ศ.1857 และในปีนั้น ลิฟท์โดยสารตัวแรกก็ได้ติดตั้งขึ้นที่ตึก E.V. Haughwout & Company ใน New York

2.2 นิยามของลิฟท์และประเภทของลิฟท์

ลิฟท์ (Lift) หมายถึง พาหนะที่ใช้สำหรับบรรทุกคนโดยสารหรือสิ่งของขึ้นลงในแนวตั้งมีรางบังคับ และวิ่งรับส่งระหว่างชั้นในตึกอาคารตั้งแต่ 2 ชั้นขึ้นไป ลิฟท์โดยสารโดยทั่วไปมี 2 ชนิด คือ

1. ลิฟท์โดยสาร (Passenger lift) เป็นลิฟท์ที่ใช้สำหรับบรรทุกคนโดยสารเป็นส่วนใหญ่ รวมทั้งลิฟท์ที่ใช้บรรทุกคนโดยสารและขนของเป็นครั้งคราว

2. ลิฟท์ขนของ (Freight lift) เป็นลิฟท์ที่ใช้สำหรับบรรทุกของเป็นส่วนใหญ่ โดยไม่มีผู้โดยสารขึ้นใด นอกจากพนักงานประจำลิฟท์ และคนงานที่ขนของออกจากลิฟท์

ลิฟท์ สามารถแบ่งออกเป็น 5 ประเภท ดังนี้

- 1) ลิฟท์ประเภท ก. เป็นลิฟท์ที่ออกแบบสำหรับขนส่งผู้โดยสาร
- 2) ลิฟท์ประเภท ข. เป็นลิฟท์ที่ออกแบบสำหรับขนส่งผู้โดยสารเป็นหลัก แต่อาจบรรทุกของได้ด้วย มีความแตกต่างที่สำคัญจากลิฟท์ประเภท ก. และ ค. คือการติดตั้งอุปกรณ์ภายในตัวลิฟท์

- 3) ลิฟท์ประเภท ค. ลิฟท์ที่ออกแบบสำหรับขนส่งเพียงคนใช้
- 4) ลิฟท์ประเภท ง. ลิฟท์ที่ออกแบบสำหรับบรรทุกของเป็นหลักแต่โดยทั่วไปจะมีผู้โดยสารร่วมไปด้วย (ลิฟท์ขนของ)
- 5) ลิฟท์ประเภท จ. ลิฟท์ที่ออกแบบสำหรับใช้ส่งของระหว่างชั้น ประกอบด้วยตัวลิฟท์ที่คนไม่สามารถเข้าไปได้ เนื่องจากมิติ และวิธีการสร้าง รังขนส่งระหว่างรางบังคับในแนวดิ่ง หรือแนวที่เอียงจากแนวดิ่งไม่เกิน 15 องศา (ลิฟท์ส่งของ)

2.3 โครงสร้างและองค์ประกอบของลิฟท์

โครงสร้างและองค์ประกอบของลิฟท์ มีรายละเอียดดังนี้

- ตัวลิฟท์ (Car) เป็นส่วนที่ใช้บรรทุกคนหรือสิ่งของ ซึ่งรวมทั้งพื้นตัวลิฟท์สำหรับห้องลิฟท์ และประตูลิฟท์
- ห้องลิฟท์ คือ โครงและส่วนประกอบที่เป็นเพดาน (ceiling) และผนังรอบ ๆ ตัวลิฟท์ ซึ่งประกอบติดอยู่บนพื้นตัวลิฟท์
- สำหรับ (Car frame) เป็นโครงซึ่งประกอบด้วยเหล็กคานบน เหล็กเสาข้าง และเหล็กคานล่าง ยึดกันเป็นสำหรับรองรับพื้นตัวลิฟท์ สำหรับนี้ จะมีตัวนำร่อง เครื่องนิรภัย ห่วงแขนสลิงหรือโซ่ และรอก ติดตั้งอยู่
- ชุดควบคุม (Controller) เป็นอุปกรณ์ หรือกลุ่มอุปกรณ์ ที่ใช้บังคับการทำงานของอุปกรณ์สำเร็จต่าง ๆ ตามที่กำหนดไว้
- รางบังคับ (Guide rail) คือ รางที่บังคับการขึ้นลงของตัวลิฟท์ หรือ น้ำหนักถ่วงตลอดแนวปล่องลิฟท์
- ปล่องลิฟท์ (Hoistway) เป็นส่วนของอาคาร ซึ่งออกแบบก่อสร้างไว้สำหรับติดตั้งลิฟท์ มีลักษณะเป็นปล่องทึบติดต่อกันระหว่างชั้นตลอดความสูงที่ลิฟท์วิ่งขึ้นลง รวมทั้งส่วนที่เป็นปล่องลิฟท์ขึ้นไปจนถึงใต้พื้นห้องเครื่อง หรือใต้พื้นหลังคา
- บ่อลิฟท์ (Pit) หมายถึง ส่วนของปล่องลิฟท์ที่อยู่ใต้ระดับพื้นตัวลิฟท์ จากชั้นล่างสุดไปจนถึงพื้นก้นปล่องลิฟท์

น้ำหนักถ่วง (Counterweight)	เป็นตุ้มน้ำหนักที่แขวน เพื่อถ่วงน้ำหนักของตัวลิฟต์ในการวิ่งขึ้นลง
โครงสร้างบนปล่องลิฟต์	หมายถึงส่วนโครงสร้างทั้งหมดรวมทั้งพื้นที่รองรับเครื่องลิฟต์ รอก และบริภัณฑ์ซึ่งติดตั้งอยู่บนปล่องลิฟต์
เครื่องกันปะทะ (Buffer)	เป็นอุปกรณ์ที่ใช้ลดการกระแทก และหยุดการเคลื่อนที่ของลิฟต์ หรือ น้ำหนักถ่วง เมื่อเคลื่อนที่ถึงเลขระดับต่ำสุด

2.4 กลยุทธ์ความปลอดภัย

ในระบบลิฟต์จะต้องมีกลยุทธ์ความปลอดภัย เพื่อคุ้มครองผู้โดยสารให้ปลอดภัยจากการเกิดอุบัติเหตุ กลยุทธ์ความปลอดภัยเหล่านี้ ได้แก่

กลยุทธ์การหยุด (Stopping Device)	เป็นลิฟท์ หรืออุปกรณ์อย่างอื่นที่ติดตั้งอยู่ในปล่องลิฟต์ในชั้นที่จอด ซึ่งเมื่อกดแล้วลิฟท์จะจอดที่ชั้นนั้น
กลยุทธ์การหยุดสุดท้าย (Final Terminal Stopping Device)	เป็นอุปกรณ์ที่ทำหน้าที่ตัดกระแสไฟฟ้าอัตโนมัติออกจากมอเตอร์ และชุดเบรก โดยอิสระ ไม่ขึ้นกับกลยุทธ์ฉุกเฉินลดความเร็ว อุปกรณ์นี้จะทำงานเมื่อตัวลิฟท์วิ่งเลยชั้นจอดแล้ว
กลยุทธ์การหยุดสุดท้ายปกติ (Normal Terminal Stopping Device)	คือ กลยุทธ์ซึ่งทำหน้าที่ลดความเร็ว และหยุดตัวลิฟท์เมื่อเข้าใกล้ปลายสุดของชั้นจอด ซึ่งทำงานเป็นอิสระ ไม่ขึ้นกับการทำงานของกลยุทธ์บังคับการทำงาน
กลยุทธ์ฉุกเฉินลดความเร็ว (Emergency Terminal Speed Limited Device)	เป็นอุปกรณ์ที่ลดความเร็วโดยอัตโนมัติ เมื่อตัวลิฟท์เข้าใกล้ชั้นจอดปลายสุด ซึ่งทำงานอิสระ ไม่ขึ้นกับกลยุทธ์บังคับการทำงาน และอุปกรณ์การหยุดสุดท้ายปกติ เพื่อให้ความเร็วตัวลิฟท์ลดลงตามต้องการ อุปกรณ์นี้จะทำงานเมื่อกลยุทธ์การหยุดสุดท้ายปกติไม่ทำงาน
กลยุทธ์เปิดและปิด (Parking Device)	เป็นอุปกรณ์ทางกลหรือไฟฟ้า ซึ่งมีหน้าที่ยอมให้ประตูปล่องลิฟท์เปิดได้จากภายนอก เมื่อตัวลิฟท์อยู่ในบริเวณเขตจอดของชั้นจอดนั้น และอุปกรณ์นี้จะใช้ปิดประตูด้วย

กลอุปกรณ์ล็อกประตูปล่องลิฟท์

เป็นอุปกรณ์ที่ทำให้ประตูปล่องลิฟท์อยู่ในตำแหน่งปิดอย่างมั่นคง และป้องกันการเปิดจากภายนอก ยกเว้นการเปิดในเงื่อนไขที่ระบุ การล็อกทางกล และทางไฟฟ้า หมายถึงการรวมอุปกรณ์ทางกล และทางไฟฟ้า ทั้งสองให้สัมพันธ์กัน แต่อิสระต่อกันและมีหน้าที่ดังนี้

- ป้องกันการทำงานของเครื่องลิฟท์เมื่อประตูปล่องลิฟท์ไม่เปิด
- ล็อกประตูปล่องลิฟท์ ให้อยู่ในตำแหน่งปิด และป้องกันการเปิดจากภายนอก ถ้าหากตัวลิฟท์ไม่อยู่ในบริเวณเขตจอดลิฟท์

กลอุปกรณ์ล็อกควบคุมของประตูปล่องลิฟท์ (Hoistway - door Interlock)

เป็นอุปกรณ์ที่มีหน้าที่ 2 อย่างสัมพันธ์ และขึ้นแก่กันดังต่อไปนี้

- ป้องกันไม่ให้เครื่องลิฟท์ทำงาน โดยอุปกรณ์ที่ใช้งานตามปกติ จนกว่าประตูปล่องลิฟท์จะล็อกอยู่ในตำแหน่งปิด
- ป้องกันไม่ให้ประตูปล่องลิฟท์เปิดจากภายนอก จนกว่าตัวลิฟท์จะอยู่ในบริเวณเขตจอด และหยุด หรือ กำลังจะหยุด

กลอุปกรณ์ล็อกเฉพาะทางกลของประตูปล่องลิฟท์
(Hoistway-gate Separate Mechanical Lock)

เป็นอุปกรณ์ล็อกทางกล ซึ่งหน้าที่ล็อกประตูปล่องลิฟท์ ให้อยู่ในตำแหน่งปิด หลังจากตัวลิฟท์เคลื่อนที่จากชั้นจอด และป้องกันไม่ให้ประตูเปิดจากภายนอกเมื่อตัวลิฟท์ไม่อยู่ในบริเวณเขตจอด

กลอุปกรณ์เปิด-ปิด ล็อกควบคุมชนิดแคมชั๊ปได้ (Hoistway door Interlock Retriving Cam Device)

เป็นอุปกรณ์ซึ่งประกอบด้วยแคม และส่วนประกอบที่ยกขึ้นลง หรือเข้าออกได้ สำหรับใช้บังคับอุปกรณ์ล็อกควบคุม โดยการทำงานเป็นอิสระ ไม่เกี่ยวข้องกับตัวเปิด-ปิดประตูด้วยกำลังของประตูลิฟท์ และประตูปล่องลิฟท์

2.5 กลอปรณ์ของสัญญาณต่าง ๆ ที่ใช้ในลิฟต์

ในระบบลิฟต์จำเป็นต้องมีการรับและส่งสัญญาณต่าง ๆ ซึ่งสัญญาณเหล่านี้จะต้องมีกลอปรณ์ในการรับหรือส่งสัญญาณ กลอปรณ์เหล่านี้ได้แก่

กลอปรณ์เรียกลิฟต์

หมายถึง ปุ่ม หรือ กลอปรณ์อย่างอื่นที่อยู่ภายในตัวลิฟต์ หรืออยู่ที่หน้าชั้นจอดของลิฟต์ ถ้าอยู่ในตัวลิฟต์ เรียกว่า Car Call ถ้าอยู่หน้าชั้นจอดลิฟต์ เรียกว่า Landing Call หรือ Hall Call กลอปรณ์นี้ทำหน้าที่ส่งสัญญาณจอดไปยังตัวลิฟต์เมื่อผู้โดยสารกดกลอปรณ์นี้

กลอปรณ์แสดงสัญญาณไฟไหม้
ในตัวลิฟต์

เป็นกลอปรณ์ที่จะให้สัญญาณเป็นแสงไฟในตัวลิฟต์ เมื่อลิฟต์เคลื่อนเข้าใกล้ชั้นจอดซึ่งมีสัญญาณเรียกลิฟต์บันทึกไว้ก่อนแล้ว

กลอปรณ์แสดงตำแหน่ง
(Position Indicator)

คืออุปกรณ์ซึ่งแสดงตำแหน่งของตัวลิฟต์ ถ้าติดตั้งอยู่หน้าชั้นจอดเรียกว่าอุปกรณ์แสดงตำแหน่งชั้น ถ้าติดตั้งอยู่ในตัวลิฟต์ เรียกว่า อุปกรณ์แสดงตำแหน่งตัวลิฟต์

กลอปรณ์แสดงให้ผู้โดยสารคอย

คืออุปกรณ์ที่แสดงไว้ที่ชั้นจอดเพื่อแสดงทิศทางของตัวลิฟต์ หรือ สัญญาณเรียกที่บันทึกไว้แต่ยังไม่ได้รับตอบ

กลอปรณ์กำหนดขึ้นที่จอด
(Dispatching Device)

เป็นกลอปรณ์ที่มีหน้าที่ในการทำงาน คือ แสดงสัญญาณเมื่อตัวลิฟต์จะออกจากที่จอด หรือ กระตุ้นกลไกเริ่มเคลื่อนที่ของลิฟต์ไปยังชั้นที่กำหนด

สวิตช์หยุดฉุกเฉิน

คือ อุปกรณ์ที่ติดตั้งอยู่ภายในตัวลิฟต์ เมื่อใช้แล้วจะตัดกำลังไฟฟ้าที่จ่ายให้กับมอเตอร์ที่ใช้ขับเคลื่อนลิฟต์ และหยุด

(Emergency Stop Switch)

การทำงานของลิฟต์

สวิตช์ไม่จอดรับ

เป็นสวิตช์ซึ่งเมื่อทำงานแล้วลิฟต์จะไม่จอดรับชั้นอื่นอีก

(Nonstop Switch)

2.6 เครื่องขับเคลื่อน

เครื่องขับเคลื่อน หมายถึง ตัวกำลังที่ให้พลังงานในการขับเคลื่อนตัวลิฟต์โดยทั่วไปจะเป็นเครื่องขับเคลื่อนขับเคลื่อนด้วยไฟฟ้า (Electric Driving Machine) ซึ่งมีมอเตอร์ ชุดเบรกและรอกขับเคลื่อน (Driving Sheave) หรือล้อแรง (Drum) พร้อมด้วยเฟือง

เครื่องขับเคลื่อนสามารถแบ่งเป็นประเภทได้ดังนี้

1. เครื่องลิฟต์ขับเคลื่อนโดยตรง (Direct-drive Machine) หมายถึง เครื่องลิฟต์ขับเคลื่อนด้วยมอเตอร์ไฟฟ้าที่ต่อโดยตรงกับรอกขับเคลื่อน ล้อแรง หรือเพลลา โดยไม่ใช้สายพาน หรือเฟืองกลาง แบ่งเป็น

ก) เครื่องลิฟต์ขับเคลื่อนด้วยเฟือง (Geared-Drive Machine) คือ เครื่องลิฟต์ที่ใช้กำลังจากมอเตอร์ไฟฟ้าผ่านเฟืองไปหมุนรอกขับเคลื่อน ล้อแรง หรือเพลลา

ข) เครื่องลิฟต์แรงความฝืด (Traction Machine) คือ เครื่องลิฟต์ที่ขับเคลื่อนตัวลิฟต์โดยความฝืดระหว่างลวดแขวนกับรอก ซึ่งมีทั้งแบบใช้เฟืองและไม่ใช้เฟืองต่อระหว่างมอเตอร์กับรอก

ค) เครื่องลิฟต์รอกกว้าน (Winding Drum Machine) คือ เครื่องลิฟต์ที่ใช้เฟืองขับเคลื่อนรอกกว้านลวดแขวนลิฟต์

ง) เครื่องลิฟต์เฟืองหนอน (Worm Geared Machine) คือ เครื่องลิฟต์ที่ใช้กำลังจากมอเตอร์ไฟฟ้า โดยผ่านเฟืองหนอนไปหมุนรอกขับเคลื่อนล้อแรง หรือเพลลา

2. เครื่องลิฟต์ขับเคลื่อนทางอ้อม (Indirect Drive Machine) คือ เครื่องลิฟต์ที่ขับเคลื่อนด้วยมอเตอร์ไฟฟ้าผ่านสายพานหรือโซ่ไปยังเฟือง เพื่อหมุนรอกขับเคลื่อนล้อแรง หรือเพลลา แบ่งออกเป็น

ก) ขับเคลื่อนด้วยสายพาน

ข) ขับเคลื่อนด้วยโซ่

ค) เครื่องลิฟต์ขับเคลื่อนด้วยเกลียว (Screw Machine) หมายถึง เครื่องลิฟต์ที่ขับเคลื่อนด้วยไฟฟ้าโดยมอเตอร์ไปหมุนแกนตั้งที่เป็นเกลียว เพื่อยกหรือยกเป็นเกลียวซึ่งยึดอยู่กับตัวลิฟต์

ง) เครื่องลิฟต์ขับเคลื่อนด้วยไฮดรอลิก (Hydraulic Machine) หมายถึง



เครื่องลิฟท์ที่ขับเคลื่อนด้วยไฟฟ้าโดยมอเตอร์ไปหมุนเครื่องสูบน้ำ (pump) ให้ทำการสูบน้ำขึ้นเข้าไปในกระบอกสูบเพื่อดันน้ำมันสู่ทำให้ตัวลิฟท์เคลื่อนที่

2.7 การควบคุมลิฟท์ (Lift Control)

การควบคุมลิฟท์ คือ ระบบควบคุมการเริ่มเคลื่อนที่ การหยุด ทิศทางของการหมุน ความเร่ง อัตราเร็ว และความหน่วงของส่วนที่เคลื่อนที่ แบ่งเป็น

1. ระบบควบคุมโดยสนามแม่เหล็กของเครื่องกำเนิดไฟฟ้า

(Generator Field Control)

คือ ระบบควบคุมซึ่งใช้วิธีการปรับแรงดันไฟฟ้าที่จ่ายให้กับมอเตอร์ที่ใช้ขับเคลื่อนลิฟท์ โดยการเปลี่ยนแปลงความเข้ม และ ทิศทางของสนามแม่เหล็กของเครื่องกำเนิดไฟฟ้า ระบบนี้ใช้กับลิฟท์ที่มีเครื่องกำเนิดไฟฟ้าของตัวเอง

2. ระบบควบคุมโดยแรงดันหลายค่า

(Multi-Voltage Control)

คือ ระบบควบคุมโดยการเปลี่ยนแรงดันที่ต่อเนื่องกันให้อาร์เมเจอร์ของมอเตอร์ที่ขับเคลื่อนลิฟท์ด้วยแรงดันต่าง ๆ ที่มีค่าคงที่ ซึ่งได้รับจากเครื่องกำเนิดไฟฟ้าที่มีคอมมิวเตเตอร์หลายชุดจ่ายให้กับลิฟท์ที่ติดตั้งเป็นกลุ่ม

3. ระบบควบคุมโดยรีโอสแตต

(Rheostatic Control)

คือ ระบบควบคุมซึ่งเปลี่ยนแปรค่าความต้านทานและ/หรือ ในวงจรสนามแม่เหล็ก

4. ระบบควบคุมโดยไฟฟ้ากระแสสลับอัตราเร็วเดียว

(Single speed Alternating Current Control)

คือ ระบบควบคุมที่ใช้มอเตอร์แบบเหนี่ยวนำ (Induction Motor) ขับเคลื่อนลิฟท์ด้วยอัตราเร็วเดียว

5. ระบบควบคุมโดยไฟฟ้ากระแสสลับสองอัตราเร็ว

(Two-Speed Alternating Current Control)

คือ ระบบควบคุมที่ใช้มอเตอร์แบบเหนี่ยวนำสองอัตราเร็วขับเคลื่อนลิฟท์ ซึ่งทำให้หมุนด้วยอัตราเร็วเชิงโคโรนัส 2 อัตราเร็ว โดยการต่อขดลวดของมอเตอร์ให้มีจำนวนขั้วต่างกัน

2.8 การทำงานของลิฟท์ (Lift Operation)

การทำงานของลิฟท์ หมายถึง วิธีบังคับระบบควบคุมให้ทำงานตามลักษณะการใช้งาน สามารถแบ่งได้เป็น 2 แบบ คือ

1. การทำงานอัตโนมัติ (Automatic Operation) การทำงานตั้งแต่ได้รับคำสั่งจนส่งผู้โดยสารหรือสิ่งบรรทุกถึงขึ้นจอดที่ต้องการ และ/หรือลิฟท์จะวิ่งไปจอดในชั้นที่กำหนด โดยการทำงานทุกขั้นตอนจะถูกควบคุมโดยเครื่องควบคุมลิฟท์ การควบคุมอัตโนมัติ แบ่งได้เป็น

ก) การทำงานอัตโนมัติของกลุ่มลิฟท์

(Group Automatic Operation)

คือ ลิฟท์ที่มีระบบการทำงานสัมพันธ์กันตั้งแต่สองตัวขึ้นไป ลิฟท์จะวิ่งและวิ่งและจอดรับส่งผู้โดยสารในทางเดียวกันโดยอัตโนมัติ ภายในตัวลิฟท์จะมีปุ่มกดสำหรับแต่ละชั้น ขึ้นละ 1 ปุ่ม และขึ้นจอดจะมีปุ่มกดสำหรับขึ้นลงอย่างละ 1 ปุ่ม ชั้นล่างสุดมีปุ่มกดขึ้นละ 1 ปุ่ม เมื่อปุ่มกดที่ขึ้นจอดถูกกด ลิฟท์ตัวใดที่อยู่ใกล้ชั้นนั้นที่สุด และกำลังวิ่งอยู่ในทิศทางเดียวกันกับที่ผู้โดยสารเรียก จะมาหยุดที่ชั้นนั้น

ข) การทำงานอัตโนมัติไม่แยกทิศทาง

(Nonselective Collective Automatic Operation)

คือ ลิฟท์ที่มีปุ่มกดสำหรับแต่ละชั้นภายในตัวลิฟท์ขึ้นละ 1 ปุ่ม และที่ขึ้นจอดขึ้นละ 1 ปุ่ม ลิฟท์จะหยุดรับส่งผู้โดยสารทุก ๆ ชั้นที่ปุ่มกดถูกกด แต่ผู้โดยสารที่เรียกลิฟท์จากภายนอกระหว่างทาง ไม่สามารถเลือกทิศทางตามความต้องการได้

ค) การทำงานอัตโนมัติแยกทิศทาง

(Selective Collective Automatic Operation)

คือ ลิฟต์ที่มีปุ่มกดสำหรับแต่ละชั้นภายในตัวลิฟต์ชั้นละ 1 ปุ่ม และที่ชั้นจอด มีปุ่มกดสำหรับเรียกลิฟต์ชั้นละ 2 ปุ่ม ทุกปุ่มที่กด จะถูกบันทึกการเรียกไว้ จนกว่าลิฟต์จะไปจอด ณ. ชั้นที่ถูกเรียกแล้วจึงจะหมดภาระ ลิฟต์จะหยุดรับส่งได้ทุกชั้นเมื่อมีผู้เรียกใช้จากภายนอกหรือภายในทั้งขึ้นและลง สำหรับชั้นจอดบนสุดหรือล่างสุด จะเรียกลิฟต์ได้ในกรณีที่ลิฟต์กำลังวิ่งมาในทิศทางที่เรียกเหล่านั้น

ง) การทำงานอัตโนมัติเฉพาะครั้ง

(Single Automatic Operation)

คือ ลิฟต์ที่มีปุ่มกดสำหรับแต่ละชั้นภายในตัวลิฟต์ชั้นละ 1 ปุ่ม และที่ชั้นจอดชั้นละ 1 ปุ่ม ผู้ใช้ลิฟต์ครั้งแรกสามารถใช้ลิฟต์จนออกจากลิฟต์แล้วลิฟต์จึงจะทำงานให้ผู้อื่นต่อไป ในขณะที่ลิฟต์กำลังทำงานถ้ากดปุ่มเรียกอีกก็จะไม่มีผลใด ๆ ทั้งสิ้น

2. การทำงานโดยสวิตช์ (Car Switch Operation) เป็นการบังคับลิฟต์ให้วิ่งไปในทิศทางที่พนักงานประจำลิฟต์บังคับ โดยใช้สวิตช์หรือปุ่มกดบังคับชนิดกดตลอดเวลาที่ต้องการให้ลิฟต์วิ่ง

ก) การทำงานโดยสวิตช์แบบจอดอัตโนมัติ

(Car switch Automatic Floor Stop Operation)

คือ การทำงานโดยสวิตช์ ซึ่งลิฟต์จอดโดยอัตโนมัติในบริเวณเขตจอดเมื่อพนักงานประจำลิฟต์ปลดสวิตช์หรือปล่อยปุ่มบังคับ

ข) การทำงานโดยการกดปุ่ม หรือ สวิตช์ตลอดเวลา

(Continuous Pressue Operation)

คือ การบังคับให้ลิฟต์วิ่ง โดยกดปุ่มหรือกดสวิตช์จากภายในตัวลิฟต์หรือชั้นจอด ลิฟต์จะหยุดทันทีเมื่อปลดสวิตช์หรือปล่อยปุ่มบังคับ

ค) การทำงานโดยการบันทึกสัญญาณล่วงหน้า

(Pre-Register Operation)

คือระบบการทำงานซึ่งสัญญาณเรียกลิฟท์ให้จอดได้รับการบันทึกไว้ล่วงหน้า โดยกดปุ่มในตัวลิฟท์หรือที่ขึ้นจอด ในระหว่างที่ลิฟท์วิ่งจะมีสัญญาณแสดงให้พนักงานประจำลิฟท์หยุดรับส่ง และลิฟท์จะจอดโดยอัตโนมัติ

ง) การทำงานโดยสัญญาณ

(Signal Operation)

คือ การทำงานโดยวิธีกดปุ่มหรือสวิตช์ในตัวลิฟท์ หรือปุ่มเรียกขึ้นลงที่ขึ้นจอด ซึ่งจะบันทึกสัญญาณบังคับลิฟท์ให้จอดสำหรับลิฟท์ตัวเดียวหรือกลุ่มลิฟท์สัญญาณบังคับให้ลิฟท์จอด ซึ่งเกิดจากการกดปุ่มในตัวลิฟท์จะ เป็นไปโดยอัตโนมัติตามลำดับที่ลิฟท์วิ่งผ่าน โดยไม่คำนึงถึงลำดับการกดก่อนหลังทั้งสองทิศทาง สัญญาณบังคับลิฟท์ให้จอดซึ่ง เกิดจากการกดปุ่มเรียกลิฟท์ขึ้นหรือลงที่ขึ้นจอด จะทำให้ลิฟท์ตัวแรกที่อยู่ใกล้ขึ้นจอดตามทิศทางเดียวกันจอดโดยอัตโนมัติ โดยไม่คำนึงถึงการเรียกก่อนหลัง การทำงานตามระบบนี้ตัวลิฟท์จะเริ่มทำงานได้โดยการกดปุ่ม หรือ กดสวิตช์เริ่มทำงาน (Starting Switch) ภายในตัวลิฟท์เท่านั้น

2.9 ลิฟท์รีเลย์และลิฟท์ไมโครคอมพิวเตอร์

2.9.1 ลิฟท์รีเลย์

ลิฟท์รีเลย์เป็นลิฟท์ที่ควบคุมใช้วงจรรีเลย์ในการควบคุมการทำงานของลิฟท์ โดยการทำงานเป็นไปตามลำดับขั้นตอนของการเปิดปิดหน้าสัมผัสของรีเลย์ แล้วนำผลของการเปิดปิดหน้าสัมผัสไปควบคุมวงจรคอนแทกเตอร์ที่ใช้ในการขับเคลื่อนหลักการทำงานของลิฟท์รีเลย์

ลิฟท์รีเลย์มีการทำงานดังรูปที่ 2.1 ซึ่งจะพบว่ารีเลย์ในเครื่องควบคุมแบ่งได้เป็นส่วนใหญ่ คือรีเลย์รับสัญญาณเข้า รีเลย์ประมวลผลและรีเลย์ให้สัญญาณคำสั่ง รีเลย์รับสัญญาณเข้าจะรับสัญญาณจากเซนเซอร์ (sensor) และปุ่มกด ซึ่งสัญญาณเหล่านี้ได้แก่ สัญญาณปุ่มกด สัญญาณจากตัวลิฟท์ สัญญาณจากปล่องลิฟท์ สัญญาณจากประตู และสัญญาณความปลอดภัย เมื่อเครื่องควบคุมรับสัญญาณเข้ามาแล้ว รีเลย์ประมวลผล จะทำการประมวลผลแล้วส่งผลที่ได้ไปยังรีเลย์ให้สัญญาณคำสั่ง สัญญาณคำสั่งนี้จะถูกส่งไปยังวงจรความปลอดภัย และหลอดไฟของปุ่มกด เพื่อทำการควบคุมการทำงานของมอเตอร์ขับเคลื่อนลิฟท์ มอเตอร์ประตู การทำงานของวงจรรีเลย์

ก. ชื่อเรียกและความหมายของรีเลย์

รีเลย์ในเครื่องควบคุมสามารถแบ่งตามหน้าที่ได้ดังนี้

รีเลย์ส่วนประตู

รีเลย์ A

รีเลย์ A₁

รีเลย์ C

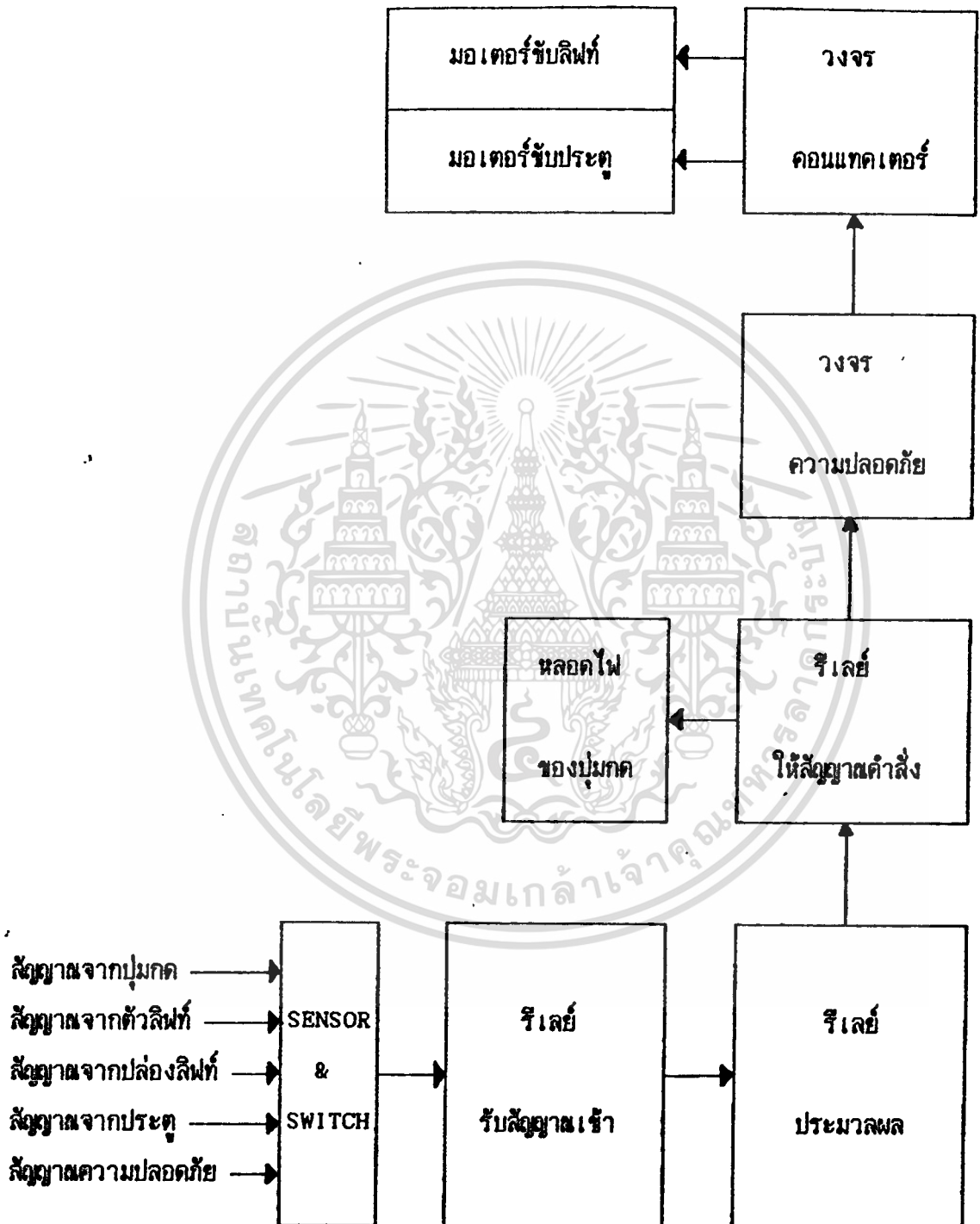
รีเลย์ C₁ , C₂

รีเลย์ RTP

ใช้สำหรับเปิดประตู รีเลย์ A จะติดในขณะที่ประตูเปิด เป็นรีเลย์ช่วยของรีเลย์ A

ใช้สำหรับเปิดประตู รีเลย์ C จะติดในขณะที่ประตูปิด เป็นรีเลย์ช่วยของ C (รีเลย์ C₁ มีเวลาหน่วง)

เป็นรีเลย์ที่มีเวลาหน่วงสำหรับเปิด-ปิดประตู เมื่อลิฟท์วิ่งจะติด ขณะที่ลิฟท์จอดปกติจะปล่อย ขณะที่ลิฟท์จอดปกติ และประตูเริ่มเปิด ถ้ากด Door Open หรือผลักเซนติว จะทำให้ RTP ติด และประตูจะเปิดกลับอีกครึ่งหนึ่ง



รูปที่ 2.1 บล็อกไดอะแกรมของลิฟท์รีเลย์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ไมโครสวิตช์ LA

เป็นไมโครสวิตช์ อยู่ในกล่องชุดมอเตอร์ประตูบนหลังคา
ลิฟท์ เมื่อเวลาประตูปิดจะต่อ เมื่อประตูเปิดสุดจะตัด

ไมโครสวิตช์ LC

อยู่ที่เดียวกับ LA เมื่อประตูเปิดจะต่อ และประตูปิดสุด
จะตัด

อินคัลเตอร์ ZP

เป็นอินคัลเตอร์บนหลังคาลิฟท์ อินคัลเตอร์นี้ เรียกอีกชื่อ
หนึ่งว่า DOOR ZONE ถ้าจุดเชื่อมต่อขึ้นจะต่อ (ตอนที่
อินคัลเตอร์อยู่ในรางตัดจุด)

รีเลย์ปุ่มกด

 $R_1 - R_n$

เป็นรีเลย์ปุ่มกดในตัวลิฟท์

 $U_1 - U_{n-1}$

เป็นรีเลย์ปุ่มกดหน้าขึ้นหาขึ้น

 $D_2 - D_n$

เป็นรีเลย์ปุ่มกดหน้าขึ้นหาลง

รีเลย์บอกตำแหน่งขึ้น

 $RS_1 - RS_n$

เป็นรีเลย์บอกตำแหน่งขึ้น ถ้าลิฟท์อยู่ขึ้นไหน รีเลย์ตัวนั้น
จะจุด เช่น จุดขึ้น 3 รีเลย์ RS_3 จะจุด

RPP

เป็นรีเลย์สำหรับช่วย HOLD รีเลย์บอกขึ้นคู่

RPD

เป็นรีเลย์สำหรับช่วย HOLD รีเลย์บอกขึ้นคู่

รีเลย์กำหนดทิศทาง

DU

รีเลย์กำหนดทิศทางลิฟท์วิ่งขึ้น

DUU

รีเลย์ช่วย DU

DD

รีเลย์กำหนดทิศทางลิฟท์วิ่งขึ้น

DDD

รีเลย์ช่วย DD

DC

ถ้ามีสัญญาณขึ้นหรือลง DC จะจุด

รีเลย์ SAFETY

PE

ถ้าสวิทช์ STOP บนหลังคา, ในตัวลิฟต์, สวิทช์ SAFETY LOCK และ FINAL SW. อยู่ในตำแหน่งปกติ PE จะติด (ปกติจะติด)

TG

เป็นรีเลย์สำหรับตัด SUPPLY ออก ในการลิฟท์วิ่งถึง FINAL SW.

รีเลย์ขับเคลื่อนมอเตอร์ลิฟท์

D

รีเลย์ขับเคลื่อนลิฟท์ลง

S'

รีเลย์ขับเคลื่อนลิฟท์ขึ้น

TA

รีเลย์ FAST SPEED

TB

รีเลย์ SLOW SPEED

E₁

รีเลย์ตัด RESISTOR ของ FAST SPEED

E₂

รีเลย์ตัด RESISTOR ของ SLOW SPEED

RA₂รีเลย์หน่วงเวลาลำดับรีเลย์ E₁

TAT

รีเลย์หน่วงเวลาลำดับรีเลย์ E₂

RK

RUNNING RELAY เมื่อลิฟท์วิ่ง RK จะติด

RKA, RKT

รีเลย์ช่วย RK โดยที่ RKT เป็นรีเลย์ที่มีการหน่วงเวลา

รีเลย์อื่น ๆ

RPR

ถ้า SAFETY SW. ต่อกปกติ และประตูตัวลิฟท์ปิดสนิท

RPR จะติด

RFC

เป็นรีเลย์ชนิด LIGHT RAY

RSM, RSMX

รีเลย์ NORMAL และรีเลย์ INSPECTION ปกติจะติด

TCC

รีเลย์ OVERLOAD

VM

รีเลย์หน่วงเวลา

F

คอยล์เบรก

๗. การทำงานของรีเลย์ MOTION CONTROL

ในภาวะปกติ รีเลย์ TG, PE, RSM, RSMX, RPP, RA, RA₂, TAT₁, กด สมมุติให้ลิฟท์อยู่ที่ชั้นที่ 2 ถ้ากดเรียกลิฟท์หน้าชั้น 3 ปุ่มลง ไฟจะมาจาก 301 ผ่านปุ่มกดมา ไคโอด มาคองส์ D₂ ผ่าน RS₂ ขา 12-4 ลง 0 ตามลูกศร ก. ในรูปที่ 2.2 ก) ทำ รีเลย์ D₂ กด เมื่อรีเลย์ D₂ กด ไฟจะผ่านมาจาก D₂ ขา 5-8 มา HOLD รีเลย์ D₂ ไว้ ตามลูกศร ข. รูปที่ 2.2. ก)

เมื่อ D₂ กดจะทำให้หน้าสัมผัส D₂ ขา 10-6 ต่อ ไฟจะไหลผ่านขา 10-6 มา RS₂ ขา 2 และ 3 แต่ผ่าน RS₂ ขา 11-3 ไม่ได้เพราะ RS₂ กดอยู่ทำให้ หน้าสัมผัสเปิด ไฟจะไหลผ่าน RS₂ ขา 2-10 ไปทำให้ DU, DU_B ทำงานตามรูปที่ 2.2 ข) เมื่อ DU, DU_B กด จะทำให้ DU, DU_B กดจะทำให้ DU ขา 8-12 ต่อ ทำให้ลูกศรขึ้นติด DU ขา 7-11 ต่อ ทำให้ DC กด ไฟจึงไหลผ่านหน้าสัมผัส DC 6-10 ผ่านไป RTP ขา 11-3, RA ขา 6-10, PE ขา 2 และ A ขา 4 ทำให้รีเลย์ C และ C₂ กด ตามรูปที่ 2.3 เมื่อรีเลย์ C กด จะทำให้หน้าสัมผัส C ขา 1 และขา 2 ต่อให้มอเตอร์ประตูกำลังทำงานตามรูป ที่ 2.4 เมื่อมอเตอร์ประตูกำลังประตูปิดสนิท GATE CONTACT ต่อ ทำให้รีเลย์ RPP กด ประตุนอกต่อ (รีเลย์ C จะหลุดเมื่อประตูปิดสนิท ไมโครสวิทช์ LC จะตัดก่อนประตูปิดสนิท 1/2" - 3/4") ไฟจะผ่านมาจาก 4 ผ่าน PAR ขา 4-12, หลักร 55, DU ขา 6-10 ผ่าน D ขา 2 ทำให้ S กดตามลูกศร ก รูปที่ 2.5 เมื่อ S กดแล้วหน้าสัมผัส S ขา 1 ต่อ ไฟผ่านมาจาก TAT₁ ขา 5-8 ทำให้ TAT กด ตามลูกศร ข รูปที่ 2.5 และไฟผ่านมาจาก RAR ขา 10-2, TB ขา 3 และ RSM ขา 9-5 ทำให้ TA กดตามลูกศร ค รูปที่ 2.5 และ TAT ขา 1 ต่อให้ RK กดตามลูกศร ง รูปที่ 2.5 เมื่อรีเลย์ S กด RK กด จะทำให้คองส์เบรค กด และ เบรคจะกางตามรูปที่ 2.6 มอเตอร์จะทำงานจนลิฟท์ให้เคลื่อนที่ ไฟจะผ่าน THERMAL OVERLOAD 380 V₂ 3 0 ผ่านหน้าสัมผัส S ขา 1 และ 2 ผ่านหน้าสัมผัส TA ขา 1 และ 2 ผ่าน RESISTOR E₁ เข้ามอเตอร์ทาง U, V, W ทำให้มอเตอร์หมุนตามรูป ที่ 2.7 เมื่อลิฟท์เคลื่อนตัวออกเล็กน้อย (ประมาณ 3-4 วินาที) รีเลย์ RA₂ จะปล่อย เพราะหน้าสัมผัส S เปิด ตามรูปที่ 2.8 และหน้าสัมผัส E₁ ขา 1, 2, 3 จะต่อคร่อม RESISTORS ของชุด START ตามรูปที่ 2.7

เมื่อลิฟต์วิ่ง FAST SPEED มาเกือบถึงชั้น 3 แคมตัด SLOW ช่างลิฟต์เข้า
 อินคัลเตอร์ชั้น 3 จะทำให้รีเลย์ RS3 และ RS3X คุดตามลูกศร ก รูปที่ 2.9 และหน้าสัมผัส
 RS3X ขา 9-5 จะต่อให้รีเลย์ RPD คุดตามลูกศร ข รูปที่ 2.9 ทำให้หน้าสัมผัส RPD ขา
 9-1 หลุด ทำให้ไม่มีไฟมาที่ RS2 รีเลย์ RS2 จึงปล่อยตามลูกศร ค. รูปที่ 2.9 ช่วงที่
 RPD คุด RPP จะไม่ปล่อยทันที เพราะรีเลย์ RPP มีเวลาหน่วงเล็กน้อย จะทำให้ไฟไหล
 ผ่านทาง RS3X ขา 10-6 มา D3 ขา 8-12 ผ่าน DDD ขา 1-9 มา RPP, RPD ขา
 10-6 ทำให้ PAR คุดตามลูกศร ก. รูปที่ 2.10 เมื่อรีเลย์ PAR คุด หน้าสัมผัส RAR ขา
 4-12 และขา 2-10 เปิด ทำให้ TA ปล่อยตามรูปที่ 2.11 หน้าสัมผัส TA ขา 3 จะต่อให้
 รีเลย์ TB คุดตามลูกศร ก รูปที่ 2.11 และหน้าสัมผัส TB ต่อให้มอเตอร์ชุด SLOW ตาม
 รูปที่ 2.7 เมื่อ TB คุด หน้าสัมผัส TB ขา 1 หลุดทำให้รีเลย์ TAT1 เริ่มปล่อย หน้าสัมผัส
 TAT1 หลุด TAT ปล่อย หน้าสัมผัส TAT ต่อให้ E2 คุด หน้าสัมผัสจะต่อให้ไฟผ่านหน้าสัมผัส
 E2 โดยไม่ผ่าน RESISTOR ของชุด SLOW ดังรูปที่ 2.7 ลิฟท์จะวิ่ง SLOW ไปจนถึงชั้น เมื่อ
 อินคัลเตอร์ตัดจอตเข้ารางตัดจอต จะทำให้หน้าสัมผัสของอินคัลเตอร์เปิด (อินคัลเตอร์ตัด
 จอตอยู่บนหลังคาลิฟท์) ดังรูปที่ 2.11 เมื่อหน้าสัมผัสเปิด ไฟไม่สามารถจะไหลผ่านไปยัง
 รีเลย์ต่าง ๆ ได้ จะทำให้รีเลย์ปล่อยและเบรกจะจับให้มอเตอร์หยุด ลิฟท์ก็จะจอดเสมอขึ้น
 เมื่อลิฟท์จอดเสมอขึ้นแล้ว อินคัลเตอร์บนหลังคา (Normally Open) อยู่ในตำแหน่ง
 กับตัวตัดจอตจะต่อ ทำให้รีเลย์ A คุดตามรูปที่ 2.12 เมื่อรีเลย์คุดจะทำให้มอเตอร์ประตูกำ
 งาน ประตูจะเปิดจนสุดแล้วไมโครสวิตช์ LA จะตัดทำให้รีเลย์ A ปล่อย

ค) การทำงานของรีเลย์ DOOR CONTROL

เมื่อลิฟท์วิ่งรีเลย์ RKT จะคุดทำให้น้ำสัมผัสขา 6-10 ต่อให้ไฟเข้ารีเลย์
 RTP คุดตามลูกศร ก รูปที่ 2.12 เมื่อลิฟท์วิ่ง SLOW จนไปเสมอขึ้น DOOR ZONE (ZP)
 จะต่อทำให้รีเลย์ A คุด และทำให้ประตูเปิดตามลูกศร ข. เมื่อประตูเปิดสุด DOOR
 OPEN LIMIT SW. (LA) จะตัด ทำให้รีเลย์ A ปล่อย แต่รีเลย์ RTP จะหน่วงเวลาไว้
 จนตัวเก็บประจุคายประจุหมด รีเลย์ RTP จึงจะปล่อย ทำให้น้ำสัมผัส RTP ขา 3-11 ต่อ
 ให้รีเลย์ C คุดตามลูกศร ค. ประตูเริ่มปิด เมื่อประตูปิดสุด DOOR CLOSE LIMIT SW.
 (LC) จะตัด ทำให้รีเลย์ C หลุด การกดปุ่ม DOOR CLOSE จะเป็นการตัดเวลาหน่วง
 ของรีเลย์ RTP ทำให้ประตูปิดทันที

ง) การทำงานของรีเลย์ SAFETY

ในกรณีที่วงจร SAFETY ไม่สมบูรณ์ หรือมีการ STOP SW. ในตัวลิฟท์ จะทำให้รีเลย์ PE ปลอย หน้าสัมผัสของ PE ขา 1 ต่อ ทำให้รีเลย์ A ถูตามลูกศร ง. ของรูปที่ 2.12 ประตูกำลังเปิดค้าง และประตูจะปิดไม่ได้ เพราะหน้าสัมผัส PE ขา 2 จะตัดไม่ให้ไฟเข้ารีเลย์ C

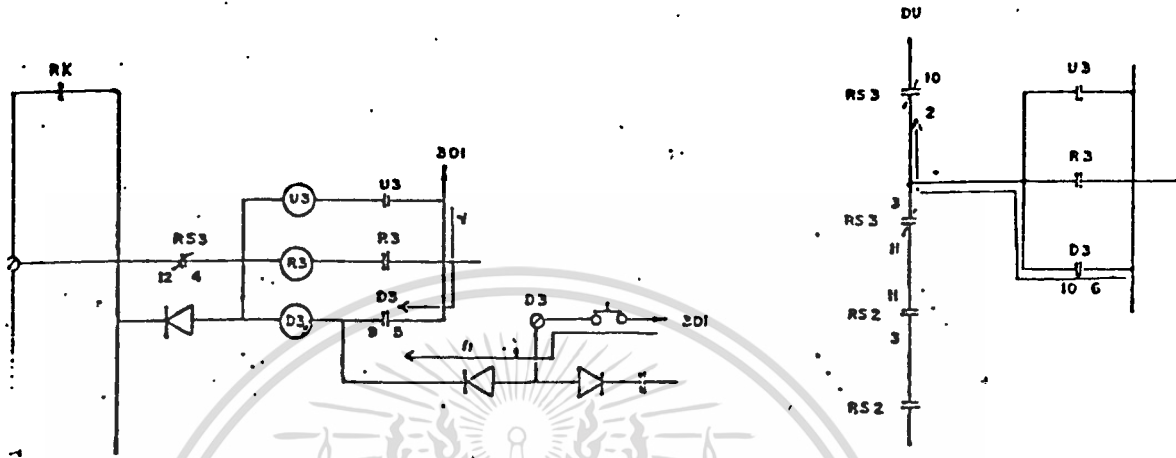
ถ้าลิฟท์จอดเสมอชั้น เมื่อมีน้ำหนักบรรทุกในลิฟท์เกิน 110 % ไมโครสวิตช์ OL จะต่อทำให้รีเลย์ TCC ถูตามรูปที่ 2.13 หน้าสัมผัส TCC ขา 6-10 จะต่อให้ RTP ถูตามลูกศร ก รูปที่ 2.14 และหน้าสัมผัส TCC ขา 7-11 จะต่อให้ OVERLOAD BUZZER ดังเพื่อให้คนรู้ว่าน้ำหนักเกิด พร้อมกันนั้นหน้าสัมผัส RTP ขา 5-8 จะต่อให้รีเลย์ A' ถู จะทำให้ประตูเปิดค้างตามลูกศร ข รูปที่ 2.14 และหน้าสัมผัส RTP ขา 3-11 จะตัดไม่ให้รีเลย์ C ถู

ในรูป 2.13 รีเลย์ RMS และ RSMX เป็นรีเลย์เกี่ยวกับการขับเคลื่อนลิฟท์เพื่อตรวจสอบลิฟท์ ถ้าลิฟท์วิ่งอัตโนมัติ รีเลย์ RSM และ RSMX จะถู และจะปล่อยเมื่อต้องการขับเคลื่อนลิฟท์โดยการ OFF สวิตช์ที่หลังคาลิฟท์ หรือในตัวลิฟท์ ส่วนรีเลย์ ZT จะเป็นรีเลย์ของพัดลมระบายความร้อนของมอเตอร์ขับเคลื่อนลิฟท์ ถ้ามีคนใช้ลิฟท์ รีเลย์ ZT จะถู ทำให้พัดลมทำงาน ถ้าไม่มีคนใช้ลิฟท์นาน ๆ รีเลย์ ZT จะปล่อย

ในรูป 15 ถ้าประตูไม่มีปัญหาแล้ว RA จะถูและจะปล่อยเมื่อประตูปิดแล้วไม่สามารถปิดได้สนิท เช่น เมื่อมีคนกดเรียกขึ้น หน้าสัมผัส DC ขา 5-8 จะต่อให้ไฟผ่าน RKA ขา 1-8 และ C ขา 4 ทำให้ C1 ถู และ C1 จะ Self HOLD ด้วยหน้าสัมผัส C1 ขา 12-8 เมื่อประตูเริ่มเปิด ถ้ามีสิ่งของมาขัดประตูทำให้ประตูปิดไม่สนิท ลิฟท์จะไม่สามารถวิ่งได้ ทั้งนี้เพราะหน้าสัมผัส RKA ขา 7-11 ไม่ต่อให้ RA เมื่อตัวเก็บประจุคายประจุจนหมดแล้ว รีเลย์ RA จะปล่อย และหน้าสัมผัส RA ขา 5-8 จะหลุด ทำให้ไม่มีไฟเลี้ยงไปที่ขั้ว 3-1 จะทำให้ปุ่มกดรีเลย์ขึ้นที่กดไว้ปล่อยหมด ทำให้รีเลย์ DC ปล่อยประตู จะเปิดกลับอีกครึ่งหนึ่ง ตามลูกศร ค. รูปที่ 2.14

รีเลย์ 3P จะเป็นรีเลย์เกี่ยวกับการที่ลิฟท์เข้า SLOW นานเกินกำหนดแล้วไม่ถึงชั้น (25-30 วินาที) ส่วนรีเลย์ 1P เป็นรีเลย์เกี่ยวกับลิฟท์เริ่ม START แต่เบรคไม่ยอมปล่อย ถ้าเกิดเหตุการณ์ใดเหตุการณ์หนึ่ง จะทำให้หน้าสัมผัสของ 3P หรือ

1P ทำให้รีเลย์ T ปลดอยู่ ดังในรูปที่ 2.15 แล้วหน้าสัมผัสของ T จะตัด SUPPLY ออก

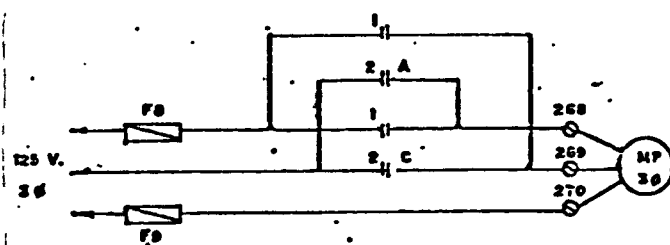


รูปที่ 2.2 ก)

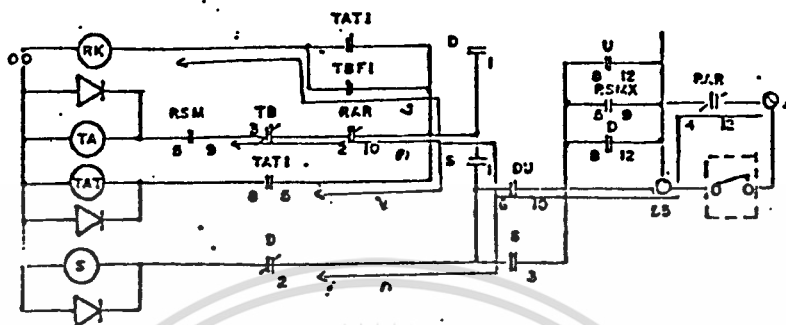
รูปที่ 2.2 ข)



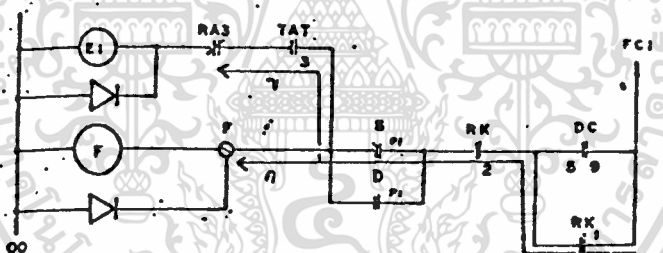
รูปที่ 2.3



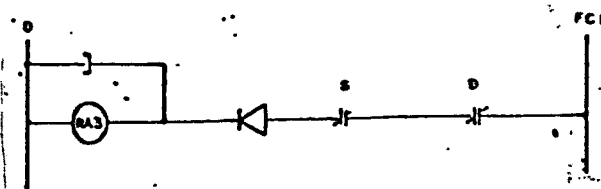
รูปที่ 2.4



รูปที่ 2.5

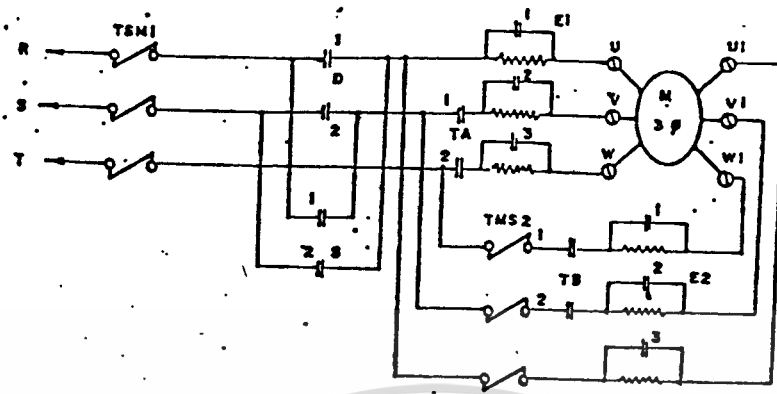


รูปที่ 2.6

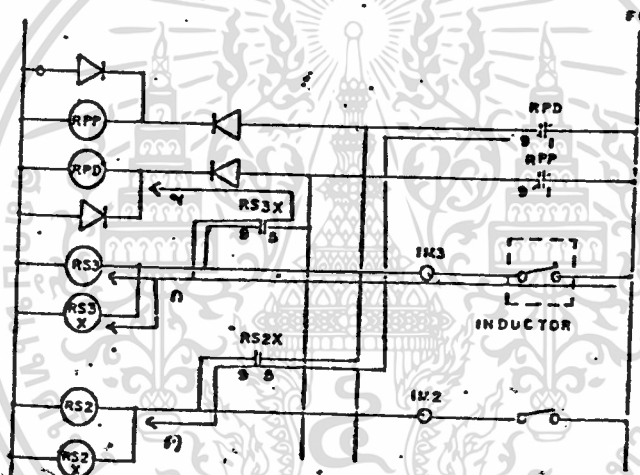


รูปที่ 2.7

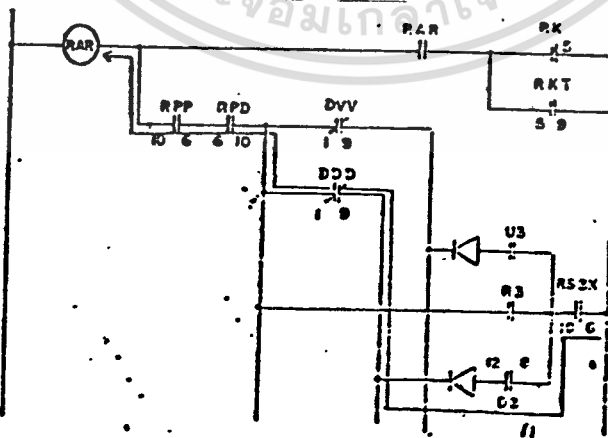
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



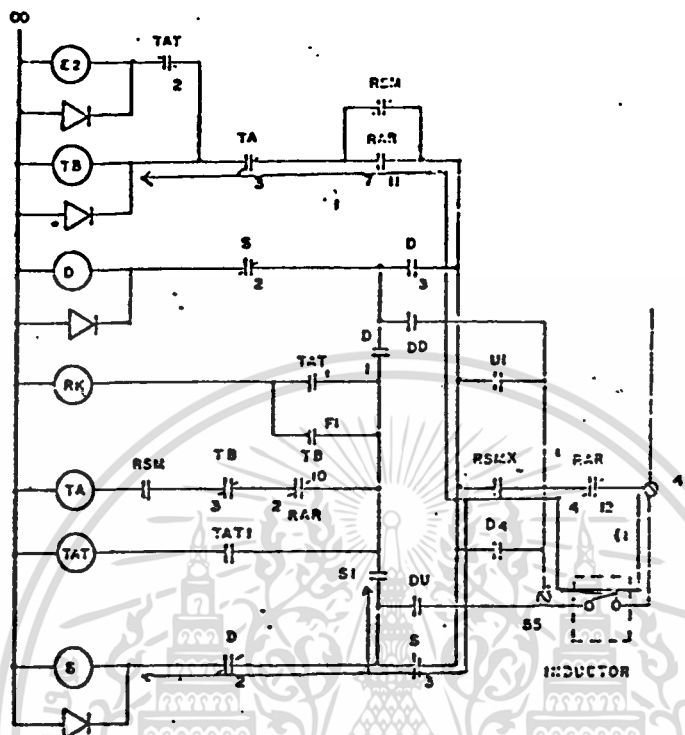
รูปที่ 2.8



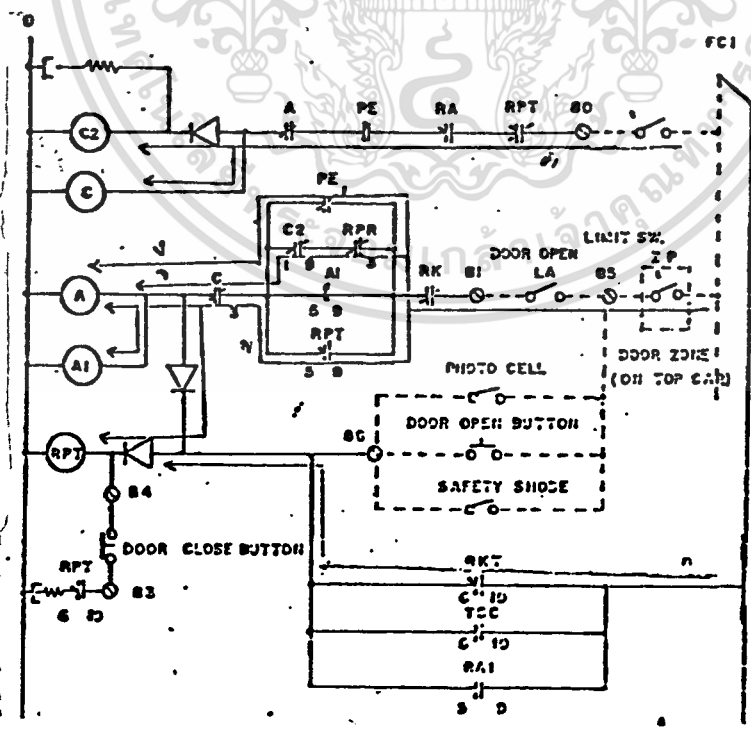
รูปที่ 2.9



รูปที่ 2.10

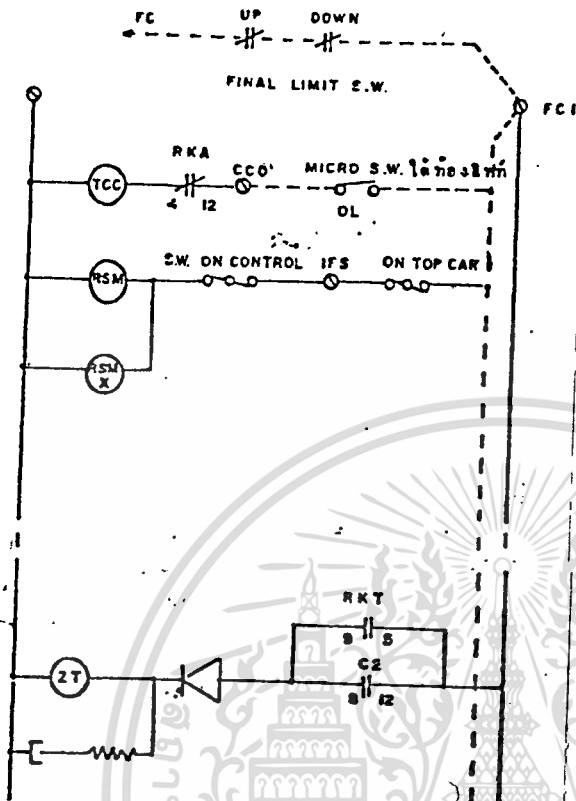


รูปที่ 2.11

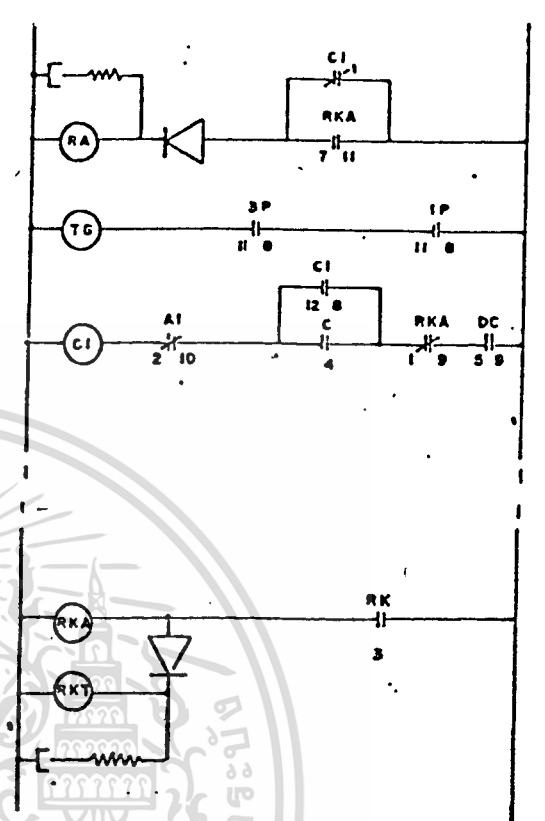


รูปที่ 2.12

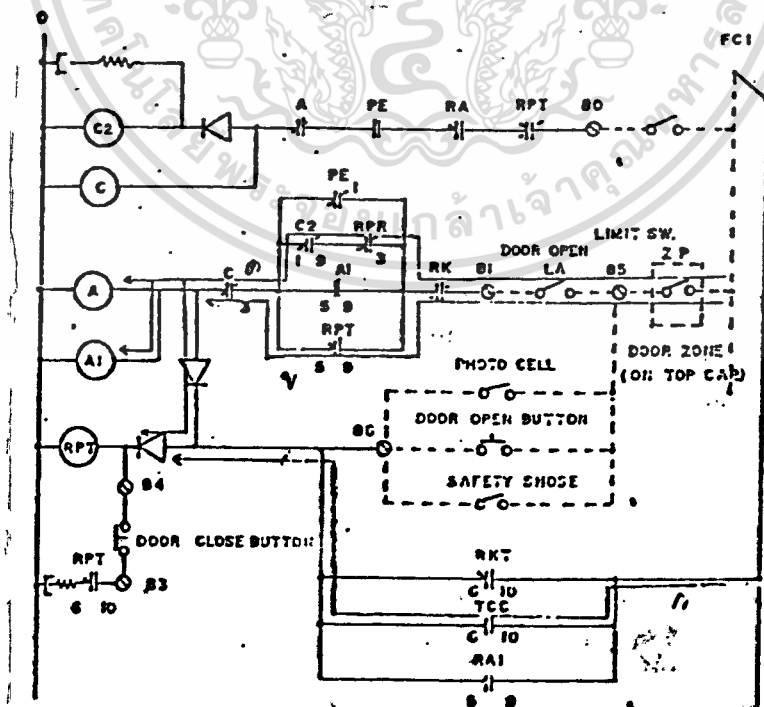
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.13



รูปที่ 2.15



รูปที่ 2.14

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.9.2 ลิฟท์ไมโครคอมพิวเตอร์

ลิฟท์ไมโครคอมพิวเตอร์ คือ ลิฟท์ที่ใช้ไมโครคอมพิวเตอร์มาใช้ในการควบคุมการทำงานของลิฟท์ การทำงานของลิฟท์สามารถเปลี่ยนแปลงได้ ด้วยการโปรแกรม โดยไม่ต้องแก้ไขฮาร์ดแวร์ เครื่องควบคุมของไมโครคอมพิวเตอร์นี้ นิยมใช้ในการควบคุมกลุ่มลิฟท์ การที่นำไมโครคอมพิวเตอร์มาใช้ในการควบคุมลิฟท์นั้น เพื่อ

1. สามารถปรับปรุงการบริการผู้ใช้ได้ง่าย
2. เพิ่มประสิทธิภาพในการใช้งาน และประหยัดพลังงาน
3. ลดขนาดของตู้ควบคุม
4. สามารถแก้ไข หรือ เปลี่ยนแปลงการทำงานของลิฟท์ได้ง่าย

ลิฟท์ไมโครคอมพิวเตอร์ มีข้อได้เปรียบกว่าลิฟท์รีเลย์ สามารถสรุปเป็นตารางเปรียบเทียบ ในตารางที่ 2.1

2.9.2.1 หลักการทำงานของลิฟท์ไมโครคอมพิวเตอร์

ลิฟท์ไมโครคอมพิวเตอร์ มีหลักการทำงาน ดังรูปที่ 2.16 ลิฟท์ไมโครคอมพิวเตอร์จะมีหน้าที่หลักอยู่ 3 อย่าง คือ การควบคุมการทำงานของลิฟท์ การควบคุมความเร็ว และการติดต่อสื่อสารกับลิฟท์อื่น ซึ่งแต่ละส่วนจะทำงานประสานกัน เพื่อควบคุมการทำงานของลิฟท์

รูปที่ 2.16 เมื่อเครื่องควบคุมรับสัญญาณอินพุตเข้ามาแล้ว จะทำการประมวลผลเพื่อควบคุมการทำงานของลิฟท์ และทำการแลกเปลี่ยนข้อมูลของสถานะการทำงานของลิฟท์ตัวอื่น เพื่อใช้ในการควบคุมกลุ่มลิฟท์ ในขณะเดียวกัน จะปรับความเร็วของลิฟท์ให้เหมาะสมกับการทำงาน เมื่อไมโครคอมพิวเตอร์ทำการประมวลผลเสร็จ ก็จะส่งคำสั่งที่ประมวลได้ ออกไปควบคุมการทำงานของลิฟท์

2.9.2.2 ลิฟท์ไมโครคอมพิวเตอร์ของต่างประเทศ

ก. ระบบลิฟท์ของ KONE

ลิฟท์นี้เป็นของประเทศฟินแลนด์ ได้ออกแบบเครื่องควบคุมลิฟท์ไมโครคอมพิวเตอร์ โดยเครื่องควบคุมแต่ละตัวมีโปรแกรมควบคุมลิฟท์ (CAR CONTROL PROGRAM) และโปรแกรมควบคุมกลุ่มลิฟท์ (Group Control Program) ดังนั้น เครื่องควบคุมลิฟท์

ตารางที่ 2.1

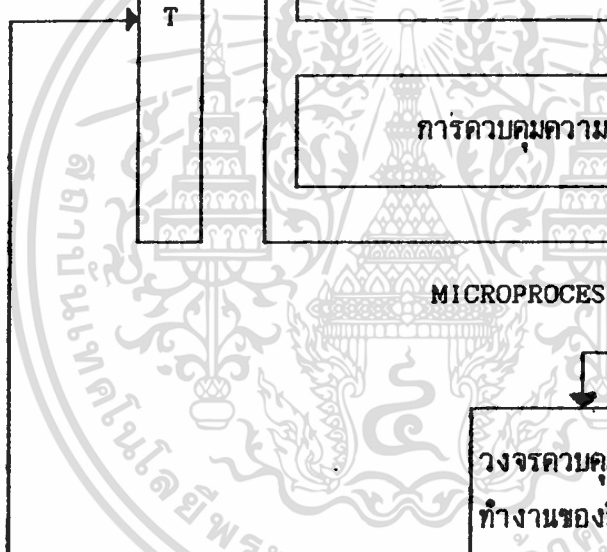
แสดงข้อเปรียบเทียบของเครื่องควบคุมลิฟต์แบบรีเลย์
กับเครื่องควบคุมลิฟต์แบบไมโครคอมพิวเตอร์

ข้อเปรียบเทียบ	เครื่องควบคุมแบบรีเลย์	เครื่องควบคุมแบบไมโครคอมพิวเตอร์
1.การทำงาน	ทำงานตามลำดับของวงจรรีเลย์ ถ้าจะแก้ไขการทำงาน จะต้องแก้ไขวงจรรีเลย์ซึ่งยุ่งยาก	ทำงานตามโปรแกรม ซึ่งสามารถแก้ไขสะดวก
2.ฟังก์ชันการทำงาน	มีน้อยและไม่สามารถทำฟังก์ชันซับซ้อนได้	สามารถทำฟังก์ชันที่ซับซ้อนได้ด้วยการโปรแกรม
3.การซ่อมบำรุง	ต้องอาศัยประสบการณ์ เพื่อตรวจดูว่ารีเลย์ตัวไหนเสียซึ่งเสียเวลามาก	สามารถดูจาก LED บอกสถานะการทำงาน และ DIAGNOSTIC PROGRAM
4.ความเร็วในการสวิตช์	การปิด-เปิด ของรีเลย์ต้องใช้หน้าสัมผัส ซึ่งการสวิตช์จะช้าและหน้าสัมผัสสึกได้	วงจรที่ใช้เป็นสารกึ่งตัวนำ สวิตช์ไม่มีหน้าสัมผัส การทำงานจึงเร็ว
5.การควบคุมกลุ่มลิฟต์ (Group Control)	เมื่อใช้ลิฟต์ตั้งแต่ 2 ตัวขึ้นไป วงจรก็จะมีขนาดใหญ่ขึ้น และเพิ่มความซับซ้อนมากยิ่งขึ้น	สามารถพัฒนาต่อให้ใช้ได้กับกลุ่มลิฟต์
6.ราคา	ถ้าจะให้ทำงานที่ซับซ้อน จำนวนรีเลย์จะมากขึ้นทำให้ราคาแพงขึ้น	ในงานที่ซับซ้อน ราคาจะถูกกว่า

ทุกตัว จึงสามารถทำหน้าที่ควบคุมกลุ่มลิฟต์ได้ การควบคุมกลุ่มลิฟต์แสดงในรูปที่ 2.17 คือ เครื่องควบคุมที่กำหนดเป็นตัวควบคุมกลุ่มลิฟต์ จะทำงานในโปรแกรมควบคุมกลุ่มลิฟต์ ในลักษณะคำสั่ง เรียกเครื่องควบคุมนี้ว่า MASTER ส่วนเครื่องควบคุมอื่น จะทำงานในโปรแกรมควบคุมลิฟต์ ในลักษณะรับสั่ง เรียกเครื่องควบคุมนี้ว่า SLAVE ในรูปที่ 2.17 สัญญาณปุ่มกดหน้าขึ้น จะต่อเข้ากับเครื่องควบคุมทุกตัว ทั้งนี้เพราะเครื่องควบคุมแต่ละตัวจะตรวจสอบความถูกต้องของกันและกัน ถ้าเครื่องควบคุมกลุ่มลิฟต์เสีย เครื่องควบคุมตัวอื่นสามารถทำหน้าที่ควบคุมกลุ่มลิฟต์แทนได้ทันที ส่วนการสื่อสารของเครื่องควบคุมทุกตัว จะต่อกันหมด ซึ่งมีข้อดีคือ ถ้าการสื่อสารระหว่างเครื่องควบคุมคู่ใด เกิดบกพร่อง ก็สามารถเปลี่ยนเส้นทางสื่อสารได้ เช่น ถ้าระหว่างเครื่องควบคุม 1 และ 2 เกิดการสื่อสารบกพร่อง ก็สามารถเปลี่ยนเส้นทางเป็นจาก 1 ไปหา 3 แล้วจึงไป 2 ได้

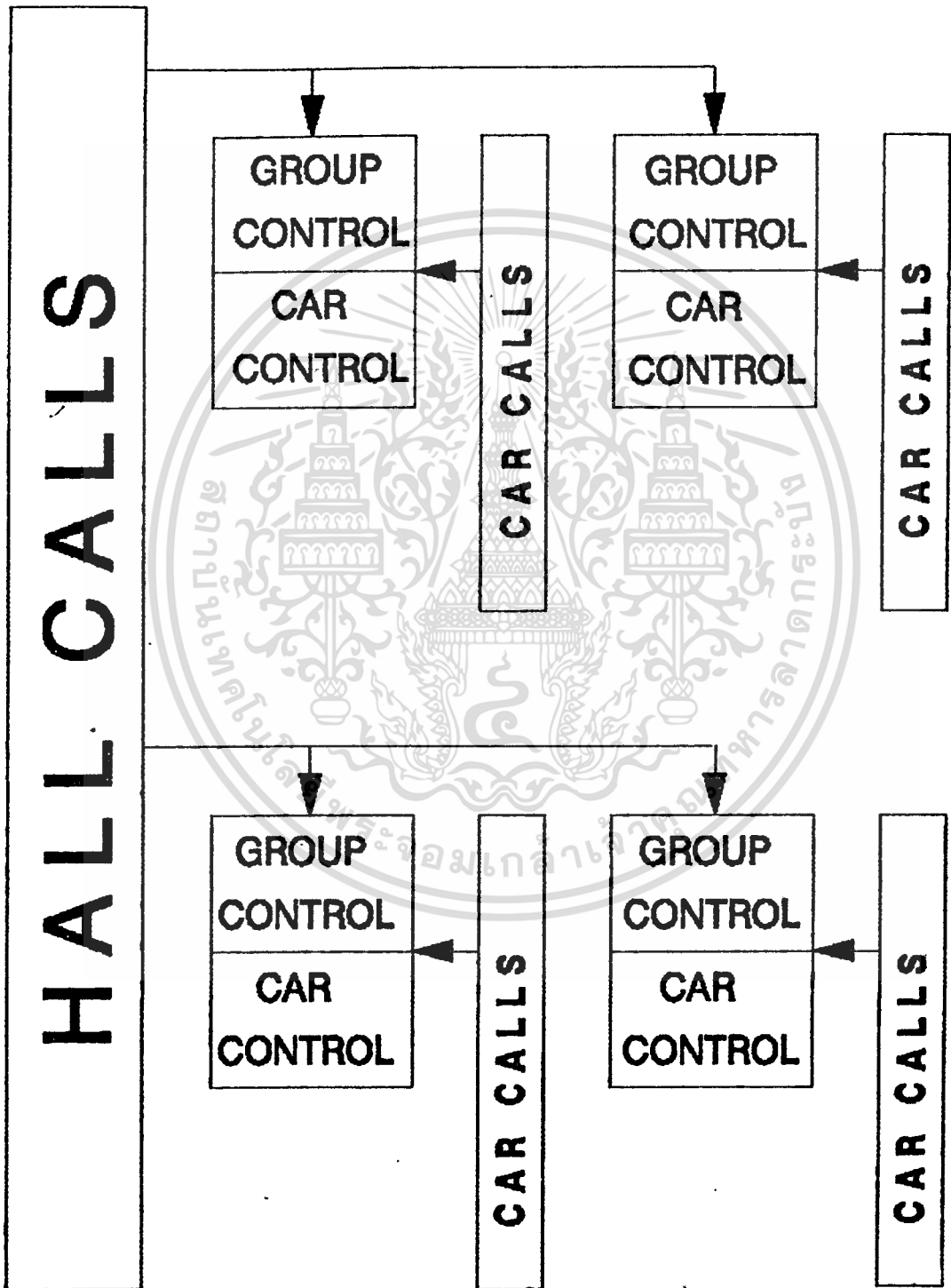
ข. ระบบลิฟต์ของญี่ปุ่น

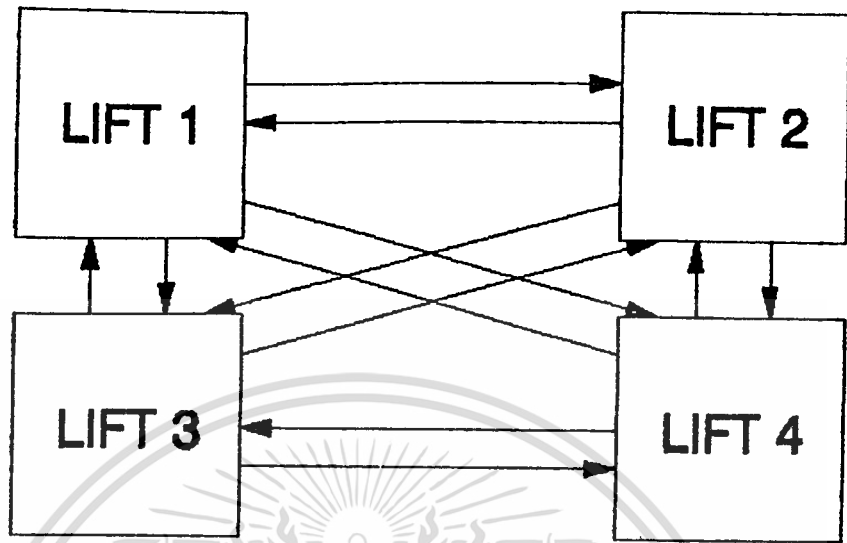
ลิฟต์ไมโครคอมพิวเตอร์ของญี่ปุ่น ที่ได้ศึกษาเป็นของบริษัทมิตซูบิชิ และ ฮิตาชิ ซึ่งทั้ง 2 บริษัท มีการออกแบบระบบลิฟต์ที่คล้ายกัน คือ จะมีเครื่องควบคุมกลุ่มลิฟต์ (Group Controller) เป็นตัวควบคุมการทำงานของกลุ่มลิฟต์ โดยเครื่องควบคุมลิฟต์ (Car Controller) แต่ละตัวจะรับคำสั่งจากเครื่องควบคุมกลุ่มลิฟต์ การออกแบบระบบแสดงในรูปที่ 2.18 ในการออกแบบระบบลิฟต์ของญี่ปุ่น เครื่องควบคุมทุกตัวจะมีไมโครคอมพิวเตอร์อยู่ 2 ชุด เป็นการสำรองซึ่งกันและกัน และ เป็นการป้องกันระบบล้มเหลวของเครื่องควบคุมลิฟต์ทุกตัว จะต่อเข้ากับเครื่องควบคุมกลุ่มลิฟต์ และสัญญาณปุ่มกดหน้าขึ้น จะต่อเข้ากับเครื่องควบคุมกลุ่มลิฟต์เท่านั้น การออกแบบในลักษณะนี้มีข้อเสีย คือ ถ้าการสื่อสารระหว่างลิฟต์ตัวใดตัวหนึ่งกับเครื่องควบคุมกลุ่มลิฟต์เกิดขัดข้องจนใช้การไม่ได้ ก็ไม่สามารถกำหนดเครื่องควบคุมกลุ่มลิฟต์ให้ทำหน้าที่ควบคุมกลุ่มลิฟต์ได้ ข้อเปรียบเทียบเครื่องควบคุมลิฟต์ไมโครคอมพิวเตอร์ แสดงในตาราง 2.2



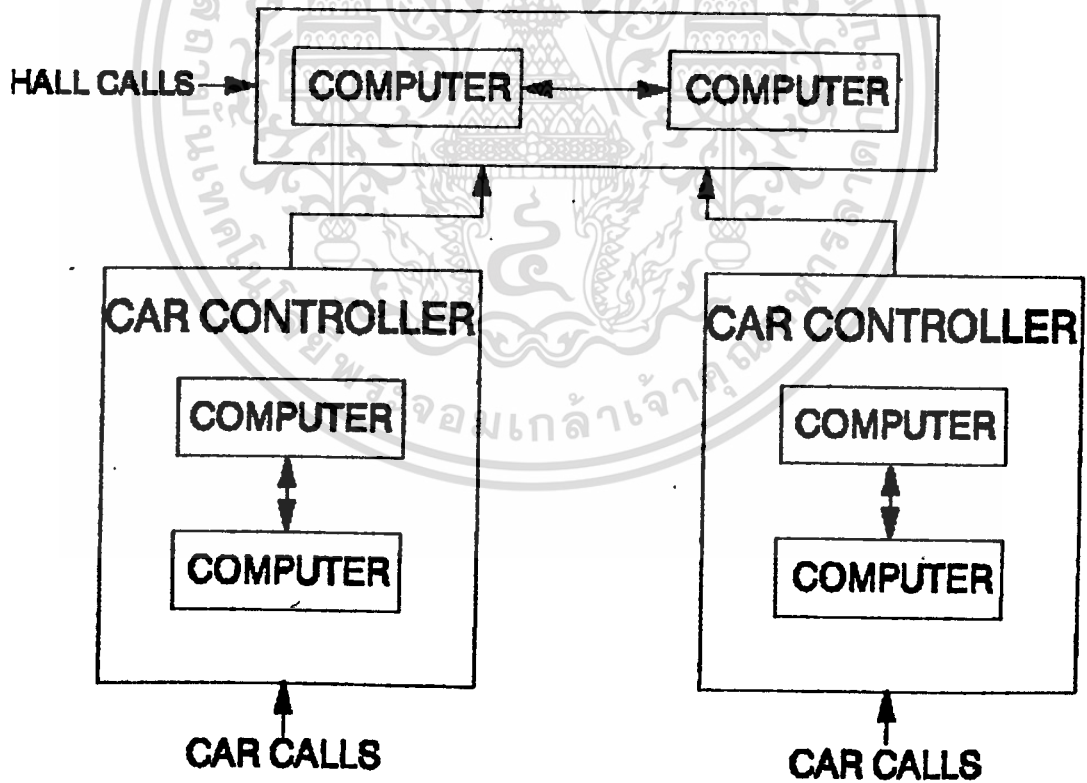
ไม่ว่ากรณีใดๆทั้งสิ้น **วันที่ 2-16 เดือนธันวาคมของปีงบประมาณ ๒๕๖๖ และต่อไปเป็นจำนวนสองสัปดาห์** ทุกครั้งที่มีการนำไปใช้

รูปที่ 2.16 บล็อกโคะแกรมของฟังก์ชันไมโครคอมพิวเตอร์





รูปที่ 2.18 การสื่อสารของระบบลิฟต์ KONE



รูปที่ 2.19 การออกแบบระบบลิฟต์ของ KONE

ตารางที่ 2.2 ข้อเปรียบเทียบของลิฟท์ไมโครคอมพิวเตอร์ที่ได้ศึกษา

ลักษณะ เปรียบเทียบ	MITSUBISHI	HITACHI	KONE
1. CPU	8086	6800	8080
2. GROUP SUPERVISORY	ให้การคำนวณ waiting time ตามหลักจิตวิทยา โดยจะไม่ให้เกินค่าที่กำหนด และให้ลิฟท์ที่ waiting time น้อยที่สุดเป็นตัวรับใช้ผู้โดยสาร	คำนวณ waiting time ของ Car แต่ละตัวแล้วเปรียบเทียบกัน ตัวไหนน้อยที่สุด ก็จะมีหมายให้ลิฟท์ตัวนั้นไป	แบ่ง Call เป็น 3 ชนิด คือ Normal Call, Priority Call และ TimeOut Call สถานะของ Call จะเปลี่ยนไปตามเวลาที่รอ ตามลำดับ
3. BACK-UP FUNCTION ON MULFUNCTION	ถ้า Group Controller เสีย ก็จะวิ่งไปจอดขึ้นใกล้ที่สุด ถ้า Car Controller เสีย Car ตัวนั้นจะถูกตัดออกจากกลุ่ม	Group Controller ใช้ Dual System ถ้าตัวหนึ่งเสีย อีกตัวหนึ่งก็สามารถทำงานได้ ถ้า Car Controller เสีย ก็จะตัดออกจากระบบ	ลิฟท์แต่ละชุดสามารถทำหน้าที่เป็น Master หรือ Slave ได้ในตัวเอง และมีการตรวจสอบระหว่างลิฟท์แต่ละตัว ถ้าลิฟท์ Master เสีย ลิฟท์ slave ที่ตรวจพบก่อน จะตั้งตนเป็น Master แล้วตัด Master ที่เสียออกจากระบบ
4. DIAGNOSTIC AND ANALYSIS DEVICE	มี	มี	มี
5. SPEED CONTROL	ใช้ระบบ Thyristor Leonard ซึ่งจะได้รับ Firing pulse มาจาก Signal Processor นอกจากนั้นมี Distance Pulse Detector เป็นตัวนับ Pulse ทำให้รู้ว่า Car อยู่ตำแหน่งไหนในปล่องลิฟท์	มี Digital Floor Controller เป็นตัวควบคุมตำแหน่งและความเร็วของ Car โดยที่ความเร็วของ Car นั้นสามารถควบคุมโดย Phase Shifter จะเป็นตัวกำหนด Firing Firing ให้ Thyristor	มีชุดควบคุมความเร็ว คือ TAC-3 ซึ่งเป็นชุดควบคุมความเร็ว โดย CPU จะออกคำสั่ง Motion Control มาที่ TAC-3 แล้วมีการป้อนสัญญาณกลับ CPU อีกครั้ง

3. การออกแบบ

3.1 รายละเอียดของการทำงานในส่วนฮาร์ดแวร์

ประกอบด้วยแผงควบคุม (Board) 3 ชนิด เสียบลงสล็อต (slot) ในตู้ควบคุม

1. แผงวงจรซีพียู (CPU Board)
2. แผงวงจรอินพุต และ เอาท์พุท (I/O Board)
3. แผงวงจรเมตริกซ์ (MATRIX Board)

3.1.1 แผงวงจรซีพียู

มีหน้าที่ และ การทำงานดังต่อไปนี้

- ส่งสัญญาณแอดเดรส และสัญญาณควบคุม (เช่น สัญญาณอ่าน, สัญญาณเขียน) ไปยังหน่วยอินพุต-เอาท์พุท รวมทั้งหน่วยความจำภายในแผงที่ CPU ต้องการทำการติดต่อด้วย เช่น ส่งสัญญาณแอดเดรส และ สัญญาณอ่าน ไปยังแผงอินพุต-เอาท์พุทที่รับสัญญาณกดเรียกลิฟท์ เพื่อทำการอ่านสัญญาณเรียกลิฟท์ ผ่านบัลลูนข้อมูล มาทำการประมวลผล
- อ่านสัญญาณจากแผงอินพุต-เอาท์พุท เช่น สัญญาณปุ่มกด, สัญญาณเรียกลิฟท์หน้าประตู, สัญญาณความปลอดภัย, สัญญาณน้ำหนักบรรทุกเกิน (overload), สัญญาณลิฟท์เข้าจอดพอดีระดับชั้น, สัญญาณขึ้นสูงสุด และ สัญญาณขึ้นต่ำสุด ซีพียูจะอ่านสัญญาณเหล่านี้ผ่านทางไอซีที่ทำหน้าที่เป็นบัลลูนเฟอ์ (Bus Buffer) แบบสองทิศทาง เบอร์ 74LS245 โดย ซีพียูจะส่งสัญญาณควบคุม ให้สัญญาณเหล่านี้ผ่านในทิศทางเข้าสู่คาตาบัส (Data Bus) แล้วนำสัญญาณมาทำการประมวลผล โดยอาศัยโปรแกรมซึ่งบรรจุอยู่ในหน่วยความจำแบบอีพรอม (EPROM) ไอซีเบอร์ 2764 (IC17) และไอซีที่เป็นหน่วยความจำแบบแรม (RAM , IC18)
- ส่งข้อมูลที่ประมวลผลได้ออกไปยังแผงอินพุต-เอาท์พุท ผ่านทางบัลลูนเฟอ์เช่นเดียวกัน แต่มีการส่งสัญญาณควบคุมให้ข้อมูลออกจากคาตาบัส สู่แผงอินพุต-เอาท์พุท
- เขียน และ อ่านข้อมูล กับหน่วยความจำ ไอซีเบอร์ 8085 ซึ่งทำหน้าที่เป็นซีพียู จะอ่านโปรแกรมที่ใช้ในการประมวลผลจากหน่วยความจำแบบอีพรอม ส่วนข้อมูลที่ให้ หรือได้จากการประมวลผล จะถูกเขียนไว้ และอ่านจาก หน่วยความจำชนิดแรม
- การที่ ซีพียู จะทำการติดต่อกับหน่วยอื่นนั้น จะต้องมีการอ้างแอดเดรสของหน่วยนั้น

ผ่านทางบัลแอตเดรล ส่งสัญญาณควบคุมผ่านบัลควบคุม แล้วส่งหรือรับข้อมูลผ่านทางบัลข้อมูล แต่เนื่องจากไอซี 8085 มีบัลแอตเดรล 16 เส้น คือ AD0-AD7 และ A8-A15 และใช้บัลแอตเดรล 8 บิตล่าง คือ AD0-AD7 เป็นบัลข้อมูลด้วย ดังนั้นจึงต้องมีการไอซี13 เบอร์ 74LS373 สำหรับแลทช์ (latch) แอตเดรล A0-A7 ไว้ ก่อนที่จะเปลี่ยนการใช้เส้น AD0-AD7 เป็นบัลข้อมูล

-ไอซี 11 เบอร์ 74LS257 เป็นดีโคเดอ์ (Decoder) ทำหน้าที่แยกสัญญาณอ่าน (RD) สัญญาณเขียน (WR), สัญญาณติดต่ออินพุต-เอาต์พุต หรือ หน่วยความจำ (IO/M) ออกเป็น สัญญาณอ่านจากหน่วยความจำ (MRD), สัญญาณเขียนสู่หน่วยความจำ (MW), สัญญาณอ่านจากหน่วยอินพุต-เอาต์พุต (IOR), สัญญาณเขียนสู่หน่วยอินพุต-เอาต์พุต (IOW) สัญญาณเหล่านี้จะถูกส่งผ่านบัลบัฟเฟอร์ (ไอซี เบอร์ 8216) ไปยังแผงหน่วยอินพุต-เอาต์พุต และแผงเมตริก เพื่อใช้เป็นสัญญาณควบคุม

3.1.2 แผงวงจรอินพุต และ เอาต์พุต

แผงหน่วยอินพุต-เอาต์พุตนี้ มีจำนวนหลายแผง ขึ้นอยู่กับจำนวนพอร์ (port) ของอินพุตและเอาต์พุต สำหรับแผงหน่วยอินพุต-เอาต์พุตที่ใช้ในโครงงานนี้เป็นแบบเอนกประสงค์ คือ สามารถใช้เป็นที่ทั้งอินพุต, เอาต์พุต หรือ ทั้งอินพุตและเอาต์พุตพอร์ต์ โดยเป็นพอร์ต์ขนาด 8 บิต

สำหรับแอตเดรลของแต่ละพอร์ต์ สามารถกำหนดได้โดยการตั้งค่าสวิตช์ S00-S07 เมื่อแผงซีพียู ต้องการติดต่อกับแผงหน่วยอินพุต-เอาต์พุต จะทำการส่งแอตเดรล ผ่านบัลแอตเดรล และสัญญาณควบคุมผ่านบัลควบคุม แต่ละพอร์ต์จะทำการตรวจสอบว่า แอตเดรลที่ส่งมานั้น ตรงกับค่าแอตเดรลที่ตั้งไว้โดย S00-S07 หรือไม่ ถ้าแอตเดรลตรงกัน แผงหน่วยอินพุต-เอาต์พุตนั้น จึงจะสามารถติดต่อกับแผงซีพียูได้

ค่าแอตเดรลที่ส่งมาจากแอตเดรลบัล จะถูกตรวจสอบโดย ไอซี 2 และ ไอซี 3 ซึ่งเป็น ไอซีเปรียบเทียบขนาด 4 บิต (4 bit Magnitude Comparator IC) ทำการตรวจสอบแอตเดรลทีละบิต ถ้าตรวจสอบแล้วพบว่า แอตเดรลถูกต้องตรงกันกับแอตเดรลของแผงทุกบิต พอร์ต์นั้นจึงจะอยู่ในลักษณะพร้อม (Enable)

ข้อมูลที่ส่งออกจากแผงซีพียู มายังแผงหน่วยอินพุต-เอาต์พุต ผ่านมาทางบัลข้อมูล

จะถูกส่งผ่าน ไอซี 15 เบอร์ 74LS373 ซึ่งเป็น ดี-แลทช์ ฟลิปฟลอป (D-Type Latched Flip-Flop) และผ่านไอซีที่เชื่อมผ่านทางแสง (opto IC) เพื่อเป็นการลดสัญญาณรบกวนด้วย

ข้อมูลที่ส่งมายังแผงหน่วยอินพุต-เอาต์พุต เพื่อรอการอ่านเข้าไปยังแผงซีพียู จะมี ไอซีที่ส่งสัญญาณเชื่อมผ่านทางแสง เพื่อเป็นการลดสัญญาณรบกวนเช่นเดียวกัน แผงซีพียูจะอ่านข้อมูล ผ่านบัลลูนข้อมูล โดยมี ไอซี 6 เป็น อินพุตบัลบัฟเฟอร์ สัญญาณข้อมูลอินพุตสามารถกำหนดให้มีการ แลทช์ หรือ ไม่แลทช์ข้อมูลที่ส่งเข้ามา ได้ด้วยการตั้งสวิตช์ S4 เหตุที่ต้องทำการแลทช์สัญญาณข้อมูลบางสัญญาณไว้ เนื่องจากสัญญาณเข้าบางสัญญาณมีลักษณะเป็นพัลส์ (Pulse) จึงต้องมีการคงสัญญาณนั้นไว้เพื่อรอการอ่านจากซีพียู

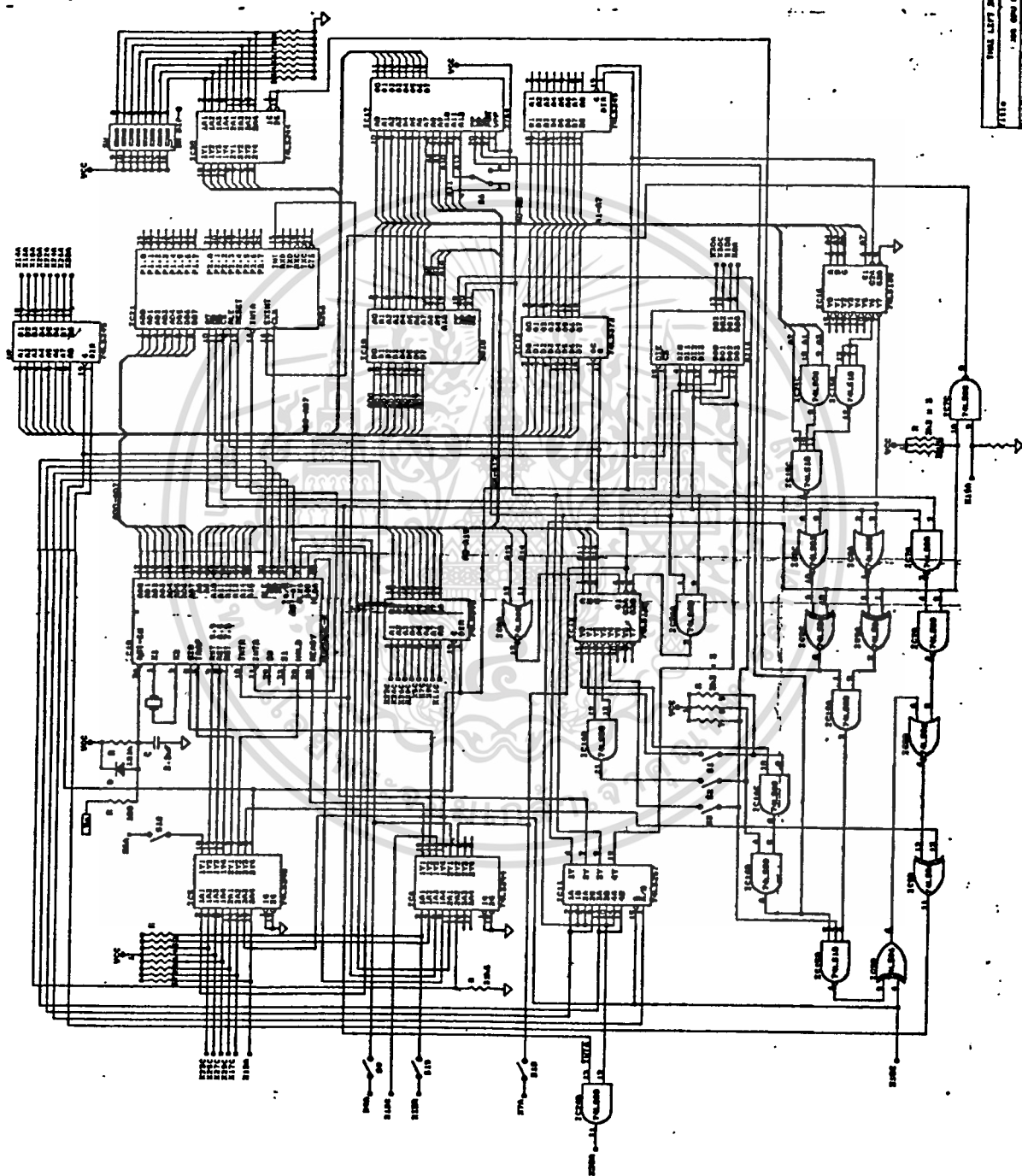
สัญญาณควบคุมจากแผงซีพียู จะร่วมกับสัญญาณเอาต์พุตจากการตรวจสอบค่าแอดเดรส ในการควบคุมแผงหน่วยอินพุต-เอาต์พุต เช่น สัญญาณควบคุมการอ่านจากหน่วยอินพุต-เอาต์พุต จะร่วมกับสัญญาณเอาต์พุตของการตรวจสอบแอดเดรส ในการควบคุมอินพุตบัลบัฟเฟอร์ จึงเห็นได้ว่า ซีพียูสามารถอ่านข้อมูลจากแผงอินพุต-เอาต์พุตได้ ถ้าซีพียูส่งแอดเดรสตรงกับแอดเดรสของพอร์ต และส่งสัญญาณควบคุมการอ่านมายังพอร์ต

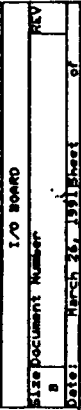
3.2.3 แผงวงจรเมตริกซ์

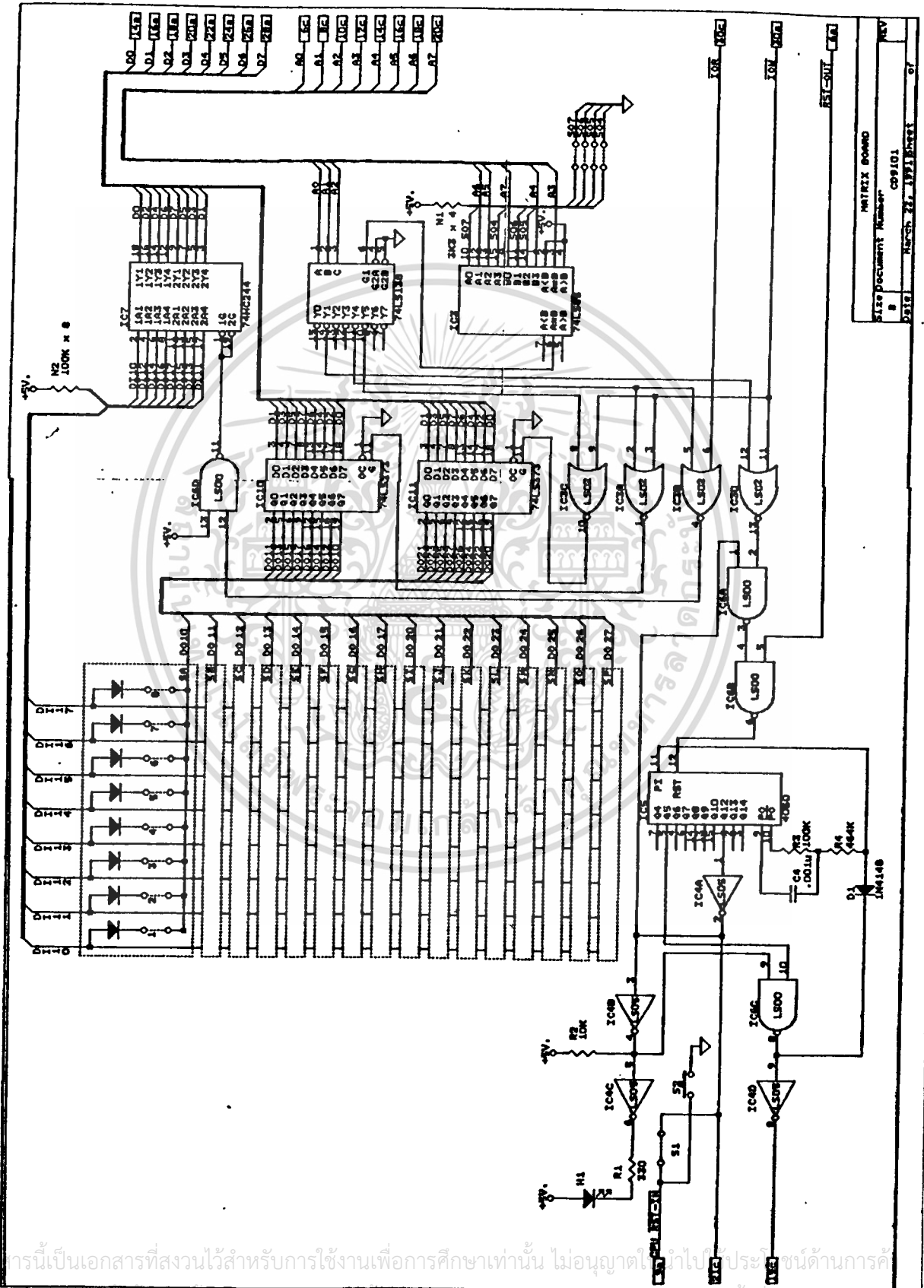
แอดเดรสของแผงเมตริกซ์ จะถูกตั้งค่าโดยสวิตช์ S04-S07 เป็นค่า 4 บิตสูง และตรวจสอบด้วยการเปรียบเทียบ โดยใช้ไอซี 3 ในลักษณะเช่นเดียวกับการตรวจสอบแอดเดรสของแผงหน่วยอินพุต-เอาต์พุต และมีไอซีดีโคเดอร์ เบอร์ 74LS138 ทำหน้าที่ถอดรหัสค่าแอดเดรส 3 บิตต่ำ

แผงเมตริกซ์ เป็นแผงสำหรับทำการอ่านข้อมูลตามค่าที่ทำการตั้งไว้ล่วงหน้า ข้อมูลที่จะทำการอ่านเข้าไปยังแผงซีพียู จะถูกต่อในลักษณะเมตริกซ์ โดยแต่ละแถวของเมตริกซ์ จะมีสวิตช์ 8 ตัว เป็นตัวตั้งค่า การเลือกค่าและอ่านค่าของข้อมูลที่ต้องการจะมีลำดับขั้นตอน คือ

1) ซีพียูจะส่งแอดเดรส และสัญญาณควบคุมการเขียนหน่วยอินพุต-เอาต์พุต เพื่อเป็นการรีนาเบิ้ล และส่งข้อมูลผ่านบัลลูนข้อมูล เพื่อเป็นการเลือกค่า (คือ แถวของเมตริกซ์) ที่ต้องการอ่าน โดยข้อมูล 8 บิตที่ส่งมา จะมีเพียงบิตเดียวที่แอดคิท (บิตมีค่าลอจิกเป็นศูนย์)







เป็นการเลือกแถวของเมตริก

2) ซีพียูส่งแอดเดรส และสัญญาณควบคุมการอ่านจากหน่วยอินพุต-เอาต์พุต เพื่อทำการอ่านค่าของข้อมูลขนาด 8 บิตที่เลือกไว้ โดยผ่านบัฟเฟอร์ เบอร์ 74LS244 เข้ามาทางบัลลูนข้อมูล

นอกจากนี้ แผงเมตริกยังมีส่วนสำคัญอีกส่วน เรียกว่า วงจรวอชด็อก (Watch-dog) คือในส่วนของ ไอซี 4060, ไอซี 4A, R3, C4, R4 วงจรนี้จะตั้งเวลาไว้ค่าหนึ่งสำหรับซีพียู ซึ่งซีพียูจะต้องทำการรีเซตวงจรนี้ทุกครั้งก่อนที่จะครบกำหนดเวลา แต่ถ้าซีพียูเกิดทำงานผิดปกติ ทำให้ไม่มีสัญญาณมาทำการรีเซตวงจรนี้ เมื่อครบกำหนดเวลาที่ตั้งไว้ จะมีสัญญาณเอาต์พุตซึ่งเป็นสัญญาณลอจิกศูนย์ (LOW) จากวงจรนี้ ไปทำการรีเซตซีพียู เพื่อให้ซีพียูกลับไปเริ่มต้นการทำงานใหม่

3.2 รายละเอียดของการทำงานในส่วนของซอฟต์แวร์

การทำงานในส่วนของซอฟต์แวร์ จะประกอบด้วย โปรแกรมหลัก และ โปรแกรมย่อยในส่วนต่าง ๆ สำหรับการทำงานในส่วนของโปรแกรมหลัก ในขณะที่เริ่มต้นจะมีการรีเซ็ตค่าของสถานะเริ่มต้นให้กับลิฟต์ โดยที่จะรีเซ็ตค่าของสัญญาณต่าง ๆ และ จะเข้าสู่ส่วนรอบการทำงานของลิฟต์ต่อไป

3.2.1 การทำงานของโปรแกรมหลัก (MAIN PROGRAM)

ภายหลังจากที่มีการรีเซ็ตระบบ จะเริ่มต้นการทำงานของโปรแกรมหลัก โดยจะเริ่มต้นตรวจสอบระบบทางฮาร์ดแวร์ จากนั้นจะทำการรีเซ็ต หรือ เคลียร์ ค่าเริ่มต้นต่าง ๆ รวมทั้งการรีเซ็ตค่าคงที่ให้กับตัวแปรที่ใช้ในโปรแกรมด้วย เมื่อทำการรีเซ็ตค่าต่าง ๆ เรียบร้อยแล้ว โปรแกรมหลักจะเรียกให้โปรแกรมย่อยตั้งสถานะเริ่มต้น (INITIALIZE) ทำงาน และ โปรแกรมย่อยการเข้าจอด และ ควบคุมประตู (STOP and DOOR) ทำงานต่อไปตามลำดับ

3.2.2 การทำงานของโปรแกรมย่อยตั้งสถานะเริ่มต้น (INITIALIZE)

โปรแกรมนี้นี้จะถูกเรียกใช้ภายหลังจากที่มีการเริ่มต้นใช้ลิฟต์ และเรียกใช้เพียงครั้ง

เดียว จนกว่าจะมีการรีเซ็ตระบบใหม่ สำหรับการทำงานของโปรแกรมในส่วนนี้ จะเริ่มต้นด้วยการตรวจสอบสัญญาณความปลอดภัย (Safety) กรณีที่ตรวจพบว่า สัญญาณนี้ต่ออยู่ก็จะทำงานต่อไป แต่ถ้าไม่พบสัญญาณจะยังคงตรวจสอบ โดยไม่ข้ามไปทำงานในส่วนอื่น จากนั้น จะทำการตรวจสอบสัญญาณใบธงของขึ้นสูงสุด หรือขึ้นล่างสุด ในกรณีที่ตรวจพบสัญญาณใดสัญญาณหนึ่ง จะทำการสั่งให้ลิฟท์วิ่งด้วยความเร็วต่ำ เข้าไปยังขึ้นสูงสุด หรือ ล่างสุด และจะทำการตรวจสัญญาณการเข้าจอดเสมอขึ้น (Levelling) และเมื่อตรวจพบจะส่งสัญญาณควบคุมให้ลิฟท์หยุดการเคลื่อนที่ จากนั้นจะเซตค่าของตัวนับขึ้นให้มีค่าเท่ากับขึ้นสูงสุด หรือ ขึ้นล่างสุด สำหรับในกรณีที่ในขณะที่เริ่มต้นตรวจไม่พบทั้งสัญญาณใบธงของขึ้นสูงสุด และ ขึ้นล่างสุด จะทำการสั่งให้ลิฟท์เคลื่อนที่ในทิศทางลงด้วยความเร็วสูง โดยมีการตรวจสอบสัญญาณใบธงของขึ้นล่างสุด เมื่อตรวจสอบพบจะเข้าจอดในชั้นล่างสุด และ ตั้งค่าของตัวนับขึ้นให้มีค่าเท่ากับชั้นล่างสุด

3.2.3 การทำงานของโปรแกรมย่อยการเข้าจอดและความคุมประตู (STOP and DOOR)

สำหรับการทำงานของโปรแกรมย่อย STOP and DOOR นี้ จะเป็นการทำงานในขณะที่ลิฟท์เข้าจอดที่ระดับชั้น โดยจะมีการเรียกใช้โปรแกรมย่อยนี้ ภายหลังจากที่มีการเซตค่าเริ่มต้นของระบบ ซึ่งการเซตค่าเริ่มต้นของระบบนี้ จะกระทำในขณะที่มีการเริ่มรีเซ็ตการทำงานของลิฟท์ในแต่ละครั้ง และโปรแกรมย่อยนี้จะถูกเรียกใช้ภายหลังจากที่ลิฟท์เข้าจอดที่ระดับชั้นพอดี ในขณะที่ลิฟท์เข้าจอดพอดีที่ระดับชั้นนั้น SAFETY SW. จะถูกตัด โปรแกรมย่อยนี้จะมีหน้าที่จัดการเกี่ยวกับการทำงานในฟังก์ชันที่เกี่ยวกับการเปิด-ปิดประตู และตรวจสอบสัญญาณความปลอดภัยต่าง ๆ เช่น การตรวจสอบ OVERLOAD, FULL LOAD เป็นต้น

ต่อไปจะกล่าวถึงรายละเอียดของการทำงานภายในโปรแกรมย่อยนี้ ในขณะที่เริ่มต้น เมื่อลิฟท์ทำการเข้าจอดได้พอดีที่ระดับชั้น โปรแกรมจะส่งสัญญาณควบคุมการเปิดประตู เพื่อให้ลิฟท์เปิดประตูดอก เมื่อประตูเปิดสุดแล้วจะมีสัญญาณตอบกลับมาว่า ในขณะนี้ประตูลิฟท์เปิดจนสุดแล้ว ในส่วนของ Timer จะเริ่มจับเวลา ต่อมาโปรแกรมจะเรียกให้โปรแกรมย่อย LOAD DIRECTION ทำงาน เพื่อเป็นการตรวจสอบสัญญาณของการเรียกใช้ลิฟท์ในชั้นที่สูงกว่าและชั้นที่ต่ำกว่า นอกจากนี้ในโปรแกรมย่อย LOAD DIRECTION ยังจะ

เป็นตัวกำหนดทิศทาง การเคลื่อนที่ของลิฟต์ต่อไปภายหลังจากที่ประตูลิฟต์ปิดสนิทแล้ว เมื่อกลับเข้ามาทำงานยังตัวโปรแกรมย่อย STOP and DOOR นี้ จะตรวจสอบสัญญาณ OVERLOAD เมื่อพบว่าเกิด OVERLOAD ขึ้นในขณะนั้น จะส่งสัญญาณแสดงผลภายในตัวลิฟต์และกลับไปควบคุมการเปิดประตูใหม่ จนกระทั่งตรวจไม่พบสัญญาณนี้แล้ว ต่อไปจะตรวจสอบสัญญาณจาก stop switch กรณีที่มีการกด stop switch โปรแกรมจะกลับไปทำงานยังฟังก์ชันของการควบคุมการเปิดประตู ซึ่งจะรวมไปถึงการตรวจสอบสัญญาณ Overload ด้วย จะยังคงทำงานซ้ำอยู่เช่นนี้จนกระทั่งตรวจไม่พบสัญญาณจาก stop switch จึงจะออกจากการทำงานในวงรอบดังกล่าว แล้วตรวจสอบสัญญาณ door open switch และสัญญาณการเรียกใช้ลิฟต์จากแผงควบคุมหน้าประตูหน้าชั้นที่ลิฟต์จอดอยู่ขณะนั้น เมื่อตรวจสอบพบสัญญาณดังกล่าว สัญญาณใดสัญญาณหนึ่ง จะกลับไปทำงานยังฟังก์ชันของการควบคุมการเปิดประตู ซึ่งจะเป็นการเซ็ต ให้ timer เริ่มจับเวลาใหม่ ในกรณีที่ตรวจสอบไม่พบทั้งสองสัญญาณ โปรแกรมจะตรวจสอบสัญญาณจาก door close switch ถ้ามีการกด door close switch หมายความว่า เป็นการเร่งให้ประตูลิฟต์ปิดอย่างรวดเร็ว โดยไม่พิจารณาเวลาที่ใช้ในการเปิดประตูจาก timer โปรแกรมก็จะทำการ รีเซ็ต door time แต่ถ้าตรวจสอบไม่พบสัญญาณจาก door close switch โปรแกรมจะตรวจสอบ door time จาก timer ว่าครบกำหนดเวลาในการเปิดประตูหรือยัง ถ้ายังไม่ครบก็จะกลับไปทำงานตั้งแต่การเรียกใช้โปรแกรมย่อย LOAD DIRECTION ตลอดจนตรวจสอบสัญญาณต่าง ๆ จนกระทั่งครบกำหนดเวลาในการเปิดประตู ก็ทำการ รีเซ็ต door time และส่งสัญญาณควบคุมการปิดประตู หลังจากนั้นจะตรวจสอบสัญญาณจาก SAFETY SW. สำหรับการทำงานของลิฟต์โดยทั่วไป ถ้าไม่มีกรณีผิดพลาดอื่น ๆ SAFETY SW. จะถูกต่อเข้าภายหลังจากประตูปิดสนิทแล้ว เมื่อตรวจสอบพบว่าหน้าสัมผัส SAFETY SW. ต่อเข้าด้วยกัน โปรแกรมจะตรวจสอบสัญญาณ FULL LOAD สำหรับกรณีที่ตรวจสอบพบว่าเกิด FULL LOAD ในขณะนั้น โปรแกรมจะไม่นำสัญญาณการเรียกใช้ลิฟต์จากแผงควบคุมหน้าประตูมาพิจารณา และจะรีเซ็ต สัญญาณลิฟต์เคลื่อนที่ขึ้น และสัญญาณลิฟต์เคลื่อนที่ลง หลังจากนั้น จะทำงานในช่วงของโปรแกรมย่อย LOAD DIRECTION เพื่อตรวจสอบสัญญาณการเคลื่อนที่ของลิฟต์ต่อไป ในขณะที่โปรแกรมย่อย LOAD DIRECTION ทำงานอยู่นี้จะไม่มี การตรวจสอบสัญญาณการเรียกใช้ลิฟต์จากแผงควบคุมหน้าประตูเลย แต่จะทำการตรวจสอบสัญญาณการ

เรียกใช้ลิฟต์จากแผงควบคุมภายในตัวลิฟต์ แล้วนำมาพิจารณาหาทิศทางในการเคลื่อนที่ของลิฟต์ต่อไป จากนั้นโปรแกรมก็จะตรวจสอบสัญญาณการเคลื่อนที่ของลิฟต์ ว่าจะมีทิศทางไปในทางใด แล้วก็จะส่งสัญญาณควบคุมไปยังส่วนขับเคลื่อนลิฟต์ให้ขับเคลื่อนลิฟต์ด้วยความเร็วสูงไปยังทิศทางที่ตรวจสอบพบ และทำงานต่อไปในโปรแกรมย่อย RUN-UP เมื่อตรวจสอบพบว่าลิฟต์มีการเคลื่อนที่ในทิศทางขึ้น หรือทำงานต่อไปในโปรแกรมย่อย RUN-DOWN เมื่อตรวจสอบพบว่าลิฟต์มีการเคลื่อนที่ในทิศทางลง สำหรับกรณีที่ตรวจสอบไม่พบทั้งสัญญาณลิฟต์เคลื่อนที่ขึ้น และสัญญาณลิฟต์เคลื่อนที่ลง โปรแกรมจะกลับไปทำงานตั้งแต่ส่วนที่มีการเรียกใช้โปรแกรมย่อย LOAD DIRECTION และทำการตรวจสอบสัญญาณตั้งแต่ต้น จนกว่าจะพบสัญญาณลิฟต์เคลื่อนที่ขึ้นหรือลง ก็จะจบการทำงานของโปรแกรมย่อยนี้

3.2.4 การทำงานของโปรแกรมย่อยกำหนดทิศทางการเคลื่อนที่ (LOAD DIRECTION)

โปรแกรมย่อย LOAD DIRECTION จะทำหน้าที่ตรวจสอบทิศทางในการเคลื่อนที่ของลิฟต์ว่า จะต้องมีการเคลื่อนที่ขึ้น หรือ ลง โดยที่จะทำการตรวจสอบจากสัญญาณลิฟต์เคลื่อนที่ขึ้น และสัญญาณลิฟต์เคลื่อนที่ลง นอกจากนี้ยังมีการตรวจสอบจากสัญญาณการเรียกใช้ลิฟต์ในชั้นที่สูงกว่า และสัญญาณการเรียกใช้ลิฟต์ในชั้นที่ต่ำกว่า โปรแกรมย่อยนี้ จะมีการถูกเรียกใช้ในขณะที่ลิฟต์เข้าจอดที่ระดับขึ้น และประตูลิฟต์เปิดจนสุดแล้ว และอาจมีการถูกเรียกใช้อีกครั้งในกรณีที่โปรแกรมตรวจสอบพบสัญญาณ FULL LOAD

เมื่อเริ่มต้นการทำงานของโปรแกรมย่อย LOAD DIRECTION นี้ จะทำการตรวจสอบสัญญาณลิฟต์ขึ้น (Direction UP : DU) เมื่อตรวจสอบพบสัญญาณ DU โปรแกรมจะทำการรีเซ็ต สัญญาณลิฟต์ลง (Direction DOWN : DD) ส่วนในกรณีที่ตรวจสอบไม่พบสัญญาณ DU โปรแกรม ก็จะทำการตรวจสอบสัญญาณ DD

ในการตรวจสอบสัญญาณ DD ถ้าพบว่ามีสัญญาณ DD โปรแกรม จะดำเนินการ รีเซ็ต สัญญาณ DU แต่ถ้าไม่มีสัญญาณ DD แสดงว่าสัญญาณการเรียกใช้ลิฟต์จากแผงควบคุมหน้าประตูหมคที่ขึ้นนี้ ซึ่งสัญญาณอ่านจากแผงควบคุมหน้าประตูของทุกชั้นจะถูก cancel ในขณะที่เริ่มเข้าสู่สัญญาณใบธงขาขึ้นหรือลงแล้ว ในกรณีนี้จะเป็นการให้ความสำคัญกับสัญญาณการควบคุมจากภายในลิฟต์มากกว่า โดยที่โปรแกรม จะทำการตรวจสอบเฉพาะสัญญาณจากแผงควบคุมภายในลิฟต์เท่านั้น หลังจากนั้น จะเข้าไปทำงานยังโปรแกรมย่อย

READ UPPER เพื่อทำการตรวจสอบว่ามีการเรียกใช้ลิฟต์จากชั้นที่สูงกว่าชั้นที่ลิฟต์จอดอยู่ในขณะนั้นหรือไม่ โดยจะตรวจสอบสัญญาณการเรียกใช้ลิฟต์จากแผงควบคุมภายในลิฟต์เท่านั้น เมื่อพบว่าการเรียกใช้ลิฟต์จากชั้นที่สูงกว่า โปรแกรมจะทำการรีเซ็ตสัญญาณลิฟต์เคลื่อนที่ลง และ รีเซ็ต สัญญาณลิฟต์เคลื่อนที่ขึ้น แต่ถ้าตรวจสอบไม่พบว่าการเรียกใช้ลิฟต์จากชั้นที่สูงกว่า จะทำงานต่อไปยังโปรแกรมย่อย READ LOWER เพื่อตรวจสอบว่ามีการเรียกใช้ลิฟต์จากชั้นที่ต่ำกว่าชั้นที่ลิฟต์จอดในขณะนั้นหรือไม่ ถ้ามีการเรียกใช้ลิฟต์จากชั้นที่ต่ำกว่า จะทำการรีเซ็ต สัญญาณลิฟต์เคลื่อนที่ขึ้น และ รีเซ็ตสัญญาณลิฟต์เคลื่อนที่ลง แต่ถ้าไม่พบสัญญาณการเรียกใช้ลิฟต์ในชั้นที่ต่ำกว่า จะออกจากการทำงานในโปรแกรมย่อยนี้ไป

3.3.5 การทำงานของโปรแกรมย่อยลิฟต์เคลื่อนที่ขึ้น (RUN UP)

การทำงานของโปรแกรมในการวิ่งขึ้น (RUN UP) ของลิฟต์ หลังจากที่ได้รับโปรแกรมย่อย STOP and DOOR ได้ประมวลผลสัญญาณแล้ว พบว่าต้องทำการส่งสัญญาณไปยังส่วนขับเคลื่อนลิฟต์ให้ทำการขับลิฟต์ขึ้นไปยังชั้นบน โปรแกรมจะทำการตรวจสอบสัญญาณชั้นบนสุด (Top floor signal) ถ้าพบสัญญาณ โปรแกรมจะส่งสัญญาณควบคุมไปยังส่วนขับเคลื่อนลิฟต์ ให้ขับลิฟต์ขึ้นด้วยความเร็วต่ำ จากนั้นจะทำการรีเซ็ตตัวนับขึ้น (counter) ให้มีค่าเท่ากับชั้นบนสุด และทำการรีเซ็ตสัญญาณลิฟต์ขึ้น (direction UP : DU) , สัญญาณลิฟต์ลง (direction DOWN : DD) ส่งสัญญาณแสดงสถานะขึ้น (counter) ไปยังส่วนแสดงผลภายในลิฟต์ และ ส่วนแสดงผลหน้าประตู และทำการตรวจสอบสัญญาณความปลอดภัย (safety switch) ในกรณีที่จะเริ่มต้นตรวจสอบพบว่าสถานะของลิฟท์ไม่ได้อยู่ที่ชั้นบนสุด โปรแกรมจะทำการตรวจสอบสัญญาณความปลอดภัยเลย เมื่อพบว่าสัญญาณความปลอดภัยถูกรีเซ็ตอยู่ จะทำการรีเซ็ตการเคลื่อนที่ของลิฟท์ทั้งในทิศทางขึ้นและลง แล้วจึงเริ่มต้นการทำงานตั้งแต่แรก แต่ถ้าพบว่าสัญญาณความปลอดภัยถูกรีเซ็ตอยู่ โปรแกรมจะสั่งให้ลิฟท์เคลื่อนที่ในทิศทางขึ้น จากนั้นก็จะทำการตรวจสอบสัญญาณที่อ่านได้จากใบธงขาขึ้น (UP Flag)

ถ้าอ่านพบสัญญาณใบธงขาขึ้น ก็จะทำให้การตรวจสอบสัญญาณสถานะของใบธงขาขึ้น (UP-Flag status) ว่าสัญญาณใบธงที่อ่านได้ถูกอ่านเป็นครั้งแรกหรือไม่ ถ้าเป็นใบธงที่ยังไม่เคย ถูกอ่าน โปรแกรม ก็จะเพิ่มค่า counter ขึ้น และส่งสัญญาณไปยังส่วนแสดงผลหน้าประตู และ ส่วนแสดงผลภายในลิฟต์ แล้วทำการรีเซ็ตสัญญาณสถานะของใบธง

ขาขึ้นเพื่อแสดงว่า ไบรชขาขึ้นของชิ้นนี้ถูกอ่านและทำการประมวลผลแล้ว ต่อมา โปรแกรม จะข้ามไปทำงานในฟังก์ชัน Fire man ซึ่งเป็นฟังก์ชันการทำงานของลิฟต์เมื่อเกิดเพลิงไหม้ จากนั้นจะทำการตรวจสอบสัญญาณเรียกใช้ลิฟต์ที่ยังขึ้นที่สูงกว่า (Higher call : HC) โดยพิจารณาจากแผงควบคุมภายในลิฟต์ และแผงควบคุมหน้าประตู โดยแผงควบคุมภายใน ลิฟต์จะพิจารณาจากหมายเลขชั้นที่สูงกว่าสถานะของ counter ในขณะนั้น สำหรับ แผงควบคุมหน้าประตู จะพิจารณาจากทิศทางการเรียกลิฟต์ทั้งการเรียกขึ้นและเรียกลง (Up call and Down call) ถ้าไม่พบสัญญาณการเรียกลิฟต์ โปรแกรมจะรีเซ็ต สัญญาณการอ่านจากแผงควบคุมหน้าประตูเพื่อเป็นการให้ความสำคัญกับสัญญาณจากแผงควบคุม ภายในลิฟต์ก่อน ถ้าพบสัญญาณ Higher call จะทำการตรวจสอบสัญญาณจุด ณ. ชั้นนั้น ซึ่งจะตรวจสอบจากแผงควบคุมภายในลิฟต์ และแผงควบคุมหน้าประตู สำหรับการตรวจสอบ จากแผงควบคุมภายในลิฟต์ จะตรวจสอบจากหมายเลขชั้นที่มีการกด ส่วนการตรวจสอบจาก แผงควบคุมหน้าประตู จะตรวจสอบจากทิศทางการเรียกลิฟต์ เฉพาะการเรียกขึ้น (Up-call) ถ้าพบสัญญาณใดสัญญาณหนึ่ง จะส่งสัญญาณควบคุมไปยังส่วนขับเคลื่อนลิฟต์ ให้ลิฟต์เคลื่อนด้วยความเร็วต่ำ (Slow speed) หลังจากนั้น โปรแกรม จะข้ามไปทำงานยังฟังก์ชัน การเข้าจอดให้พอดีระดับชั้น (Leveling process) เมื่อไม่พบสัญญาณการเข้าจอด ณ. ระดับชั้น จะทำการรีเซ็ตการเคลื่อนที่ของลิฟต์ ทั้งในทิศทางการเคลื่อนที่ขึ้นและลง และ กลับมาเริ่มตรวจสอบระดับชั้นสูงสุดใหม่

ในกรณีที่ลิฟต์กำลังเคลื่อนที่ขึ้น และตรวจไม่พบสัญญาณไบรชขาขึ้น โปรแกรมจะ รีเซ็ตสัญญาณแสดงสถานะการอ่านไบรชขาขึ้น และ ตรวจสอบความเร็วของลิฟต์ในขณะนั้น ถ้าพบว่าส่วนขับเคลื่อนลิฟต์กำลังขับเคลื่อนลิฟต์ด้วยความเร็วสูง โปรแกรม จะกลับไปเริ่มตรวจสอบ สัญญาณขึ้นบนสุด แต่ถ้าส่วนขับเคลื่อนลิฟต์กำลังขับเคลื่อนลิฟต์ด้วยความเร็วต่ำ (Slow speed) โปรแกรม จะเข้าสู่การทำงานในฟังก์ชันการเข้าจอดให้พอดีระดับชั้น (Leveling process)

ภายหลังจากตรวจพบสัญญาณไบรชขาขึ้น แล้วตรวจพบว่าสัญญาณแสดงสถานะการอ่าน ของไบรชขาขึ้นถูก เซ็ต อยู่ แสดงว่า สัญญาณไบรชขาขึ้นถูกอ่าน และประมวลผลผ่านไปแล้ว จะกระโดดไปทำการตรวจสอบความเร็วลิฟต์ ถ้าตรวจสอบพบว่าส่วนขับเคลื่อนลิฟต์กำลังขับเคลื่อนลิฟต์ด้วยความเร็วสูง โปรแกรมจะกลับไปเริ่มต้นโปรแกรมย่อย RUN UP ใหม่ ถ้า

พบว่าลิฟท์กำลังเคลื่อนที่ด้วยความเร็วต่ำ โปรแกรมจะไปยังฟังก์ชันการเข้าจอดให้พอดีระดับขึ้น

3.2.6 การทำงานของโปรแกรมย่อยลิฟท์เคลื่อนที่ลง (RUN DOWN)

การทำงานของโปรแกรมช่วงวิ่งลง (RUN DOWN) ของลิฟท์ หลังจากที่ได้ตัวโปรแกรมหลักทำการประมวลผลสัญญาณแล้ว กรณีที่พบว่าจะต้องส่งสัญญาณไปยังส่วนขับเคลื่อนลิฟท์ให้ทำการขับเคลื่อนลิฟท์ลงไปยังชั้นล่าง โปรแกรม จะทำการตรวจสอบสัญญาณขึ้นล่างสุด (BOTTOM-Floor Signal) ถ้าพบสัญญาณ โปรแกรม จะส่งสัญญาณควบคุมไปยังส่วนขับเคลื่อนลิฟท์ให้ขับเคลื่อนลิฟท์ลงด้วยความเร็วต่ำ จากนั้นจะทำการ เซ็ต ตัวนับขึ้น (counter) ให้มีค่าเท่ากับชั้นล่างสุด และทำการ รีเซ็ต สัญญาณลิฟท์ขึ้น (direction UP : DU) , สัญญาณลิฟท์ลง (direction DOWN : DD) ส่งสัญญาณแสดงสถานะขึ้น (counter) ไปยังส่วนแสดงผลภายในลิฟท์ และ ส่วนแสดงผลหน้าประตู และทำการตรวจสอบสัญญาณความปลอดภัย (safety switch) ในกรณีที่ขณะเริ่มต้นตรวจสอบพบว่าสถานะของลิฟท์ไม่ได้อยู่ที่ชั้นล่างสุด โปรแกรม จะทำการตรวจสอบสัญญาณความปลอดภัยเลย เมื่อพบว่าสัญญาณความปลอดภัยถูกเซ็ตอยู่ โปรแกรมจะทำการรีเซ็ตการเคลื่อนที่ของลิฟท์ทั้งในทิศทางขึ้น และ ลง แล้วจึงเริ่มต้นการทำงานตั้งแต่แรก แต่ถ้าพบว่าสัญญาณความปลอดภัยถูกรีเซ็ตอยู่ โปรแกรมย่อยนี้จะสั่งให้ลิฟท์เคลื่อนที่ในทิศทางลง จากนั้นจะทำการตรวจสอบสัญญาณที่อ่านได้จากใบธงขาลง (DOWN Flag)

ถ้าอ่านพบสัญญาณใบธงขาลง ก็จะทำให้การตรวจสอบสัญญาณสถานะของใบธงขาลง (DOWN-Flag status) ว่าสัญญาณใบธงที่อ่านได้ถูกอ่านเป็นครั้งแรกหรือไม่ ถ้าเป็นใบธง ที่ยังไม่เคยถูกอ่าน โปรแกรม ก็จะลดค่า counter ลง และส่งสัญญาณไปยังส่วนแสดงผลหน้าประตู และส่วนแสดงผลภายในลิฟท์ แล้วทำการ รีเซ็ต สัญญาณสถานะของใบธงขาลงเพื่อแสดงว่า ใบธงขาลงของชั้นนี้ได้ถูกอ่าน และทำการประมวลผลแล้ว ต่อมาโปรแกรม จะข้ามไปทำงานในฟังก์ชัน Fire man ซึ่งเป็นฟังก์ชันการทำงานของลิฟท์เมื่อเกิดเพลิงไหม้ จากนั้นจะทำการตรวจสอบสัญญาณเรียกใช้ลิฟท์ขึ้นที่ต่ำกว่า (LOWER - Call : LC) โดยพิจารณาจากแผงควบคุมภายในลิฟท์ และแผงควบคุมหน้าประตู โดยแผงควบคุมภายในลิฟท์จะพิจารณาจากหมายเลขขึ้นซึ่งต่ำกว่าสถานะของ counter ในขณะนั้น

สำหรับแผนควบคุมหน้าประตู จะพิจารณาจากทิศทาง การเรียกลิฟท์ทั้งการเรียกขึ้น และ เรียกลง (UP-Call & DOWN Call) ถ้าไม่พบสัญญาณการเรียกลิฟท์ โปรแกรม จะ รีเซ็ต สัญญาณการอ่านจากแผนควบคุมหน้าประตูเพื่อเป็นการให้ความสำคัญกับสัญญาณจาก แผนควบคุมภายในลิฟท์ก่อน ถ้าพบสัญญาณ LOWER Call จะทำการตรวจสอบสัญญาณจุด ณ. นั้นนั้น ซึ่งจะตรวจสอบจากแผนควบคุมภายในลิฟท์ และแผนควบคุมหน้าประตู สำหรับการ ตรวจสอบ จากแผนควบคุมภายในลิฟท์ จะตรวจสอบจากหมายเลขขึ้นที่มีการกด ส่วนการ ตรวจสอบจาก แผนควบคุมหน้าประตู จะตรวจสอบจากทิศทาง การเรียกลิฟท์ เฉพาะการ เรียกลง (DOWN Call) ถ้าพบสัญญาณใดสัญญาณหนึ่ง จะส่งสัญญาณควบคุมไปยังส่วนขับเคลื่อนลิฟท์ ให้ขับลิฟท์ ด้วยความเร็วต่ำ (slow speed) หลังจากนั้น โปรแกรม จะข้ามไปทำงานยังฟังก์ชัน การเข้าจอดให้พอดีระดับขึ้น (leveling process) เมื่อ ไม่พบสัญญาณการเข้าจอด ณ. ระดับขึ้น จะทำการรีเซ็ตการเคลื่อนที่ของลิฟท์ทั้งในทิศทาง การเคลื่อนที่ขึ้นและลง แล้วกลับมาเริ่มตรวจสอบระดับขึ้นต่ำสุดใหม่

ในกรณีที่ลิฟท์กำลังเคลื่อนที่ลง และตรวจไม่พบสัญญาณใบธงชาลง โปรแกรม จะ รีเซ็ต สัญญาณแสดงสถานะการอ่านใบธงชาลง และ ตรวจสอบความเร็วของลิฟท์ในขณะนั้น ถ้าพบว่าส่วนขับเคลื่อนลิฟท์กำลังขับลิฟท์ด้วยความเร็วสูง โปรแกรม จะกลับไปเริ่มตรวจสอบ สัญญาณขึ้นล่างสุด แต่ถ้าส่วนขับเคลื่อนลิฟท์กำลังขับลิฟท์ด้วยความเร็วต่ำ (slow speed) โปรแกรม จะเข้าสู่การทำงานในฟังก์ชันการเข้าจอดให้พอดีระดับขึ้น (leveling process)

ภายหลังจากตรวจพบสัญญาณใบธงชาลง แล้วตรวจพบว่าสัญญาณแสดงสถานะการอ่าน ของใบธงชาลงถูกเรียกอยู่ แสดงว่าสัญญาณใบธงชาลงถูกอ่าน และประมวลผลผ่านไปแล้ว จะกระโดดไปทำการตรวจสอบความเร็วลิฟท์ ถ้าตรวจสอบพบว่าส่วนขับเคลื่อนลิฟท์กำลังขับเคลื่อนลิฟท์ด้วยความเร็วสูง โปรแกรมจะกลับไปเริ่มต้นโปรแกรมย่อย RUN DOWN ใหม่ ถ้าพบว่าลิฟท์กำลังเคลื่อนที่ด้วยความเร็วต่ำ โปรแกรม จะไปยังฟังก์ชันการเข้าจอดให้พอดี ระดับขึ้น

3.2.7 การทำงานของโปรแกรมย่อยอ่านสัญญาณในชั้นที่สูงกว่าลิฟท์ขณะนั้น (READ UPPER)

โปรแกรมย่อยนี้จะถูกเรียกใช้สองกรณี คือ ขณะที่ลิฟท์กำลังเคลื่อนที่ขึ้น แล้วตรวจพบ

สัญญาณใบธงขาขึ้น (Up flag) และ ในขณะที่ลิฟต์จอดอยู่ที่ระดับชั้น กำลังทำงานในช่วงของโปรแกรมย่อย LOAD DIRECTION อยู่ โดยที่โปรแกรมย่อยนี้จะมีหน้าที่ตรวจสอบว่า มีสัญญาณการเรียกใช้ลิฟต์จากชั้นที่สูงกว่าชั้นที่ลิฟต์อยู่ในขณะนั้นหรือไม่ ซึ่งจะทำให้การตรวจสอบทั้งสัญญาณเรียกลิฟต์จากแผงควบคุมภายในลิฟต์ (Car call) และ สัญญาณเรียกลิฟต์จากแผงควบคุมหน้าประตู (Hall call) เมื่อมีการตรวจพบสัญญาณการเรียกใช้ลิฟต์จากชั้นที่สูงกว่า โปรแกรมย่อยนี้จะทำการเซตสัญญาณการเรียกใช้ลิฟต์จากชั้นที่สูงกว่า (Higher-call : HC) ในกรณีที่ตรวจสอบไม่พบโปรแกรมจะทำการเซตสัญญาณดังกล่าว

โปรแกรมย่อย Read Upper นี้ จะมีรายละเอียดของการทำงานดังต่อไปนี้ คือ ในขณะที่เริ่มต้นการทำงานนั้น โปรแกรมจะทำการตรวจสอบระดับชั้นของลิฟต์ในขณะนั้นว่า ลิฟต์อยู่ที่ชั้นบนสุด หรือไม่ ถ้าตรวจสอบพบว่าลิฟต์อยู่ในชั้นบนสุด โปรแกรมก็จะรีเซ็ตสัญญาณการเรียกใช้ลิฟต์จากชั้นที่สูงกว่าทันที แต่ถ้าตรวจสอบพบว่าลิฟต์ไม่ได้อยู่ที่ชั้นบนสุด จะเริ่มทำการตรวจสอบสัญญาณเรียกลิฟต์ จากแผงควบคุมภายในลิฟต์ของชั้นที่สูงกว่าชั้นที่ลิฟต์อยู่ในขณะนั้นหนึ่งชั้น ถ้าพบว่ามีสัญญาณเรียกลิฟต์ โปรแกรมจะเซตสัญญาณการเรียกใช้ลิฟต์จากชั้นที่สูงกว่า ถ้าไม่พบจะไปทำการตรวจสอบสถานะของสัญญาณการอ่านสัญญาณการเรียกใช้ลิฟต์จากแผงควบคุมหน้าประตู ถ้ามีสถานะถูกรีเซ็ตอยู่ โปรแกรมจะไม่ตรวจสอบสัญญาณการเรียกใช้ลิฟต์จากแผงควบคุมหน้าประตูเลย โดยจะข้ามไปทำงานที่ท้าย Loop เลย แต่ถ้าสถานะของสัญญาณนั้นถูกเซต โปรแกรมจะทำการตรวจสอบสัญญาณการเรียกใช้ลิฟต์จากแผงควบคุมหน้าประตูในทิศทางขึ้น (Up Hall Call) เมื่อพบว่ามีสัญญาณก็จะเซตสัญญาณการเรียกใช้ลิฟต์จากชั้นที่สูงกว่า แล้วข้ามไปทำงานที่ท้าย Loop เมื่อตรวจสอบไม่พบสัญญาณ โปรแกรมจะตรวจสอบสัญญาณการเรียกใช้ลิฟต์จากแผงควบคุมหน้าประตูในทิศทางลง (Down Hall Call) ถ้าตรวจพบสัญญาณนี้ จะเซตสัญญาณการเรียกลิฟต์จากชั้นที่สูงกว่า แล้วข้ามไปทำงานที่ท้าย loop แต่ถ้าตรวจไม่พบสัญญาณนี้ จุดต่อไปจะเป็นจุดสุดท้ายของการทำงานใน loop นี้ โดยจะมีการตรวจสอบเงื่อนไขสองข้อ ข้อแรก คือ โปรแกรมทำการตรวจสอบสัญญาณจนถึงชั้นบนสุดแล้วหรือยัง ถ้ายังก็จะเพิ่มขึ้นที่จะต้องทำการตรวจสอบขึ้นไปอีกหนึ่งชั้น แล้วทำงานซ้ำใน loop เดิมอีก แต่ถ้าโปรแกรมตรวจสอบจนถึงชั้นบนสุดแล้ว ก็จะออกจากการทำงานใน loop นี้ สำหรับข้อที่สองนั้น คือ โปรแกรมจะทำการตรวจสอบว่าสัญญาณการเรียกใช้ลิฟต์จากชั้นที่สูงกว่า ถูกเซตแล้วหรือยัง ถ้าถูกเซต

แล้ว ก็จะออกจากการทำงานใน loop นี้ไป แต่ถ้ายังไม่ถูกเซตจะกลับไปทำงานซ้ำที่ต้น Loop จนกระทั่งมีการตรวจสอบจนถึงขั้นบนสุด หรือ มีการเซตค่าของสัญญาณการเรียกใช้ลิฟต์จากชั้นที่สูงกว่า ซึ่งก็จะจบการทำงานของโปรแกรมย่อยนี้

3.2.8 การทำงานของโปรแกรมย่อยอ่านสัญญาณในชั้นที่ต่ำกว่าลิฟต์ขณะนั้น (READ LOWER)

การทำงานของโปรแกรมย่อยนี้ จะคล้ายกับการทำงานของโปรแกรมย่อย Read Upper จะต่างกันก็ตรงที่โปรแกรมย่อยนี้ มีหน้าที่ในการตรวจว่ามีการเรียกใช้ลิฟต์จากชั้นที่ต่ำกว่าชั้นที่ลิฟต์กำลังอยู่ในขณะนั้นหรือไม่ ถ้ามีการเรียกใช้ลิฟต์ในชั้นที่ต่ำกว่า ไม่ว่าจะ เป็นสัญญาณการเรียกใช้ลิฟต์จากแผงควบคุมหน้าประตู หรือ สัญญาณการเรียกใช้ลิฟต์จากแผงควบคุมภายในลิฟต์ โดยจะทำการเซตสัญญาณการเรียกใช้ลิฟต์ในชั้นที่ต่ำกว่า (Lower call signal : LC) เมื่อตรวจพบว่ามี การเรียกใช้ลิฟต์ในชั้นที่ต่ำกว่า และ ทำการรีเซตสัญญาณนี้ เมื่อไม่พบการเรียกใช้ลิฟต์จากชั้นที่ต่ำกว่า สำหรับการเรียกใช้โปรแกรมย่อยนี้ จะมีการเรียกใช้จากสองที่ คือ จากขณะที่ลิฟต์กำลังเคลื่อนที่ลงด้วยความเร็วสูง ผ่านเข้าสู่โพงชาลง (Down Flag) และ จากขณะที่ลิฟต์กำลังจอดอยู่ที่ระดับชั้น โดยกำลังทำงานอยู่ในช่วงของโปรแกรมย่อย Load Direction รายละเอียดการทำงานของโปรแกรมย่อยนี้ จะเริ่มต้นโดยมีการตรวจสอบว่าระดับชั้นที่ลิฟต์กำลังเคลื่อนที่เข้าไปนั้น หรือ ระดับชั้นที่ลิฟต์จอดอยู่นั้น เป็นระดับชั้นต่ำสุดหรือไม่ ถ้าพบว่าเป็นชั้นล่างสุด โปรแกรมในส่วนนี้จะรีเซตสัญญาณการเรียกใช้ลิฟต์ในชั้นที่ต่ำกว่า และ จบการทำงานของโปรแกรมย่อยนี้ เมื่อตรวจสอบพบว่าไม่ใช่ชั้นล่างสุด จะเริ่มทำการตรวจสอบสัญญาณการเรียกใช้ลิฟต์จากแผงควบคุมภายในลิฟต์ของชั้นที่ต่ำกว่าชั้นที่ลิฟต์อยู่ในขณะนั้นหนึ่งชั้น ถ้าพบว่ามีสัญญาณการเรียกใช้ลิฟต์ โปรแกรมก็จะทำการเซตสัญญาณการเรียกใช้ลิฟต์จากชั้นที่ต่ำกว่า ถ้าไม่พบจะไปทำการตรวจสอบสถานะของสัญญาณอ่านสัญญาณการเรียกใช้ลิฟต์ จากแผงควบคุมหน้าประตู ถ้ามีสถานะถูกรีเซตอยู่ โปรแกรมจะไม่ตรวจสอบสัญญาณการเรียกใช้ลิฟต์จากแผงควบคุมหน้าประตูเลย โดยจะข้ามไปทำงานที่ท้ายวงรอบของการทำงานนี้เลย แต่ถ้าสถานะของสัญญาณนั้นถูกเซตโปรแกรม จะทำการตรวจสอบสัญญาณการเรียกใช้ลิฟต์ จากแผงควบคุมหน้าประตูในทิศทางลง (Down hall call) เมื่อพบว่ามีสัญญาณก็จะเซตสัญญาณการเรียกใช้ลิฟต์จากชั้นที่ต่ำกว่า แล้วจะข้ามไปทำงานที่ท้ายวงรอบของการทำงานที่มีการ

ตรวจสอบเงื่อนไขเลย เมื่อตรวจสอบแล้วไม่พบสัญญาณ โปรแกรมจะตรวจสอบสัญญาณการเรียกใช้ลิฟต์จากแผงควบคุมหน้าประตูในทิศทางขึ้น (Up hall call) ถ้าตรวจพบสัญญาณนี้ จะเช็คสัญญาณการเรียกใช้ลิฟต์จากชั้นที่ต่ำกว่า แล้วข้ามไปทำงานที่ท้ายของวงรอบการทำงานเลย แต่ถ้าตรวจสอบไม่พบสัญญาณนี้ โปรแกรมจะทำการตรวจสอบเงื่อนไขข้อ โดยที่ข้อแรก คือ โปรแกรมทำการตรวจสอบสัญญาณจนถึงชั้นล่างสุดแล้วหรือยัง ถ้ายังก็จะลดชั้นที่จะต้องทำการตรวจสอบลงไปอีกชั้นหนึ่ง แล้วทำงานซ้ำในวงรอบเดิมอีก แต่ถ้าโปรแกรมตรวจสอบจน ถึงชั้นล่างสุดแล้ว ก็จะออกจากการทำงานในวงรอบนี้ สำหรับการตรวจสอบเงื่อนไขข้อที่สองนั้น คือ โปรแกรมจะทำการตรวจสอบว่าสัญญาณเรียกใช้ลิฟต์จากชั้นที่ต่ำกว่าถูกเช็ค จะกลับไปทำงานซ้ำที่ต้นของวงรอบการทำงานนี้ จนกระทั่งมีการตรวจสอบจนถึงชั้นล่างสุด หรือมีการเช็คค่าของสัญญาณการเรียกใช้ลิฟต์จากชั้นที่ต่ำกว่า และเมื่อตรวจตรงกับเงื่อนไขใดเงื่อนไขหนึ่งแล้ว จะจบการทำงานของโปรแกรมน้อยนี้กลับไปเข้าไปทำงานต่อยังโปรแกรมที่เรียกใช้โปรแกรมน้อยนี้ต่อไป

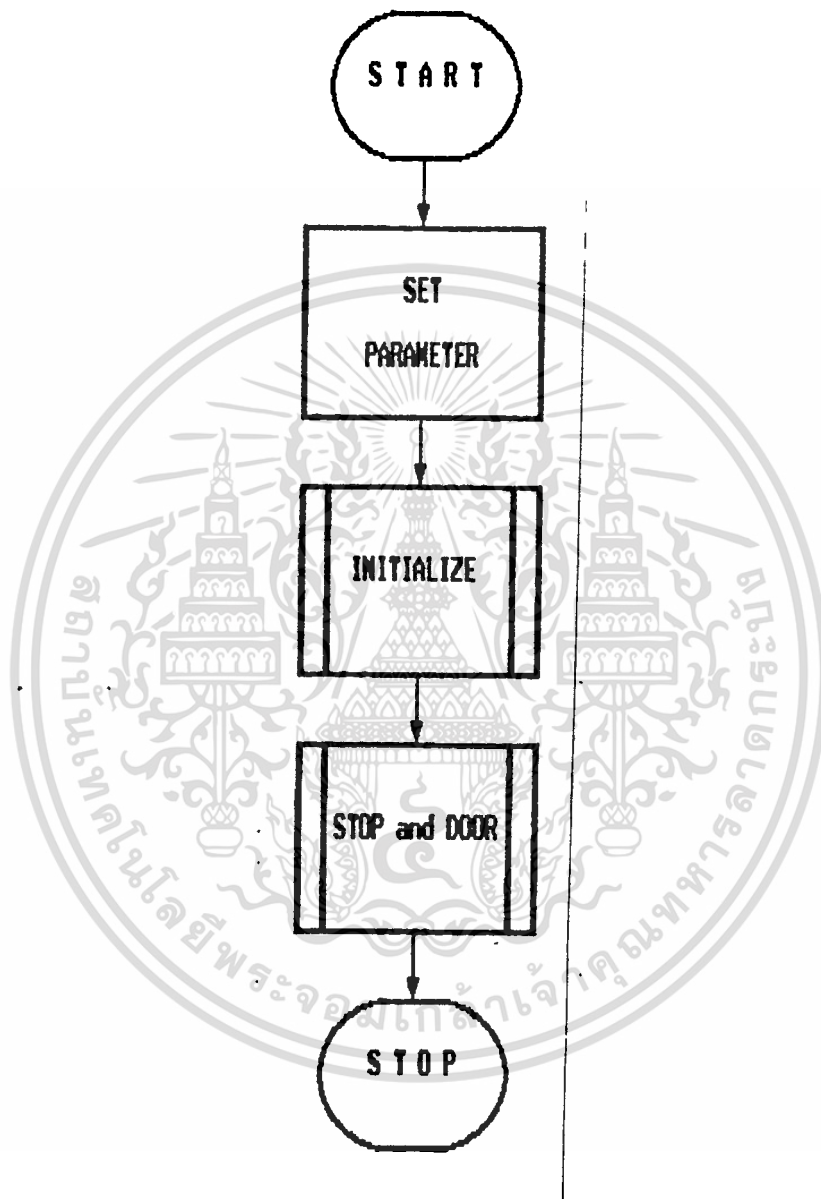
3.2.8 การทำงานของโปรแกรมน้อยตรวจสอบการเข้าจอดที่ระดับขึ้น (CHECK STOP)

ในส่วนของโปรแกรมน้อย Check Stop นี้ จะทำหน้าที่ในขณะที่ลิฟต์กำลังวิ่งทั้งในทิศทางขึ้น และ ในทิศทางลง โดยโปรแกรมน้อยนี้จะทำงานในกรณีทีลิฟต์ตรวจพบสัญญาณโบธงในแต่ละชั้น ไม่ว่าจะเป็นสัญญาณโบธงขาขึ้น (Up Flag) หรือ สัญญาณโบธงขาลง (Down Flag) เมื่อโปรแกรมตรวจพบสัญญาณโบธง จะทำงานในฟังก์ชันต่าง ๆ แล้วจะเข้ามาทำงานในช่วงของโปรแกรมน้อยนี้ โดยมีจุดประสงค์เพื่อที่จะตรวจสอบว่าจะต้องทำการเข้าจอดในชั้นนี้หรือไม่ ในกรณีที่ตรวจสอบพบว่าจะต้องจอดในชั้นนี้ โปรแกรมจะส่งสัญญาณไปยังส่วนขับเคลื่อนลิฟต์ ให้ขับเคลื่อนลิฟต์ด้วยความเร็วต่ำ (Slow speed) จนกระทั่งทำการเข้าจอดต่อไป สำหรับในกรณีที่ตรวจสอบพบว่าไม่ต้องเข้าจอดในชั้นนี้ โปรแกรมจะส่งสัญญาณไปยังส่วนขับเคลื่อนลิฟต์ ให้ขับเคลื่อนลิฟต์ด้วยความเร็วสูง (Fast speed)

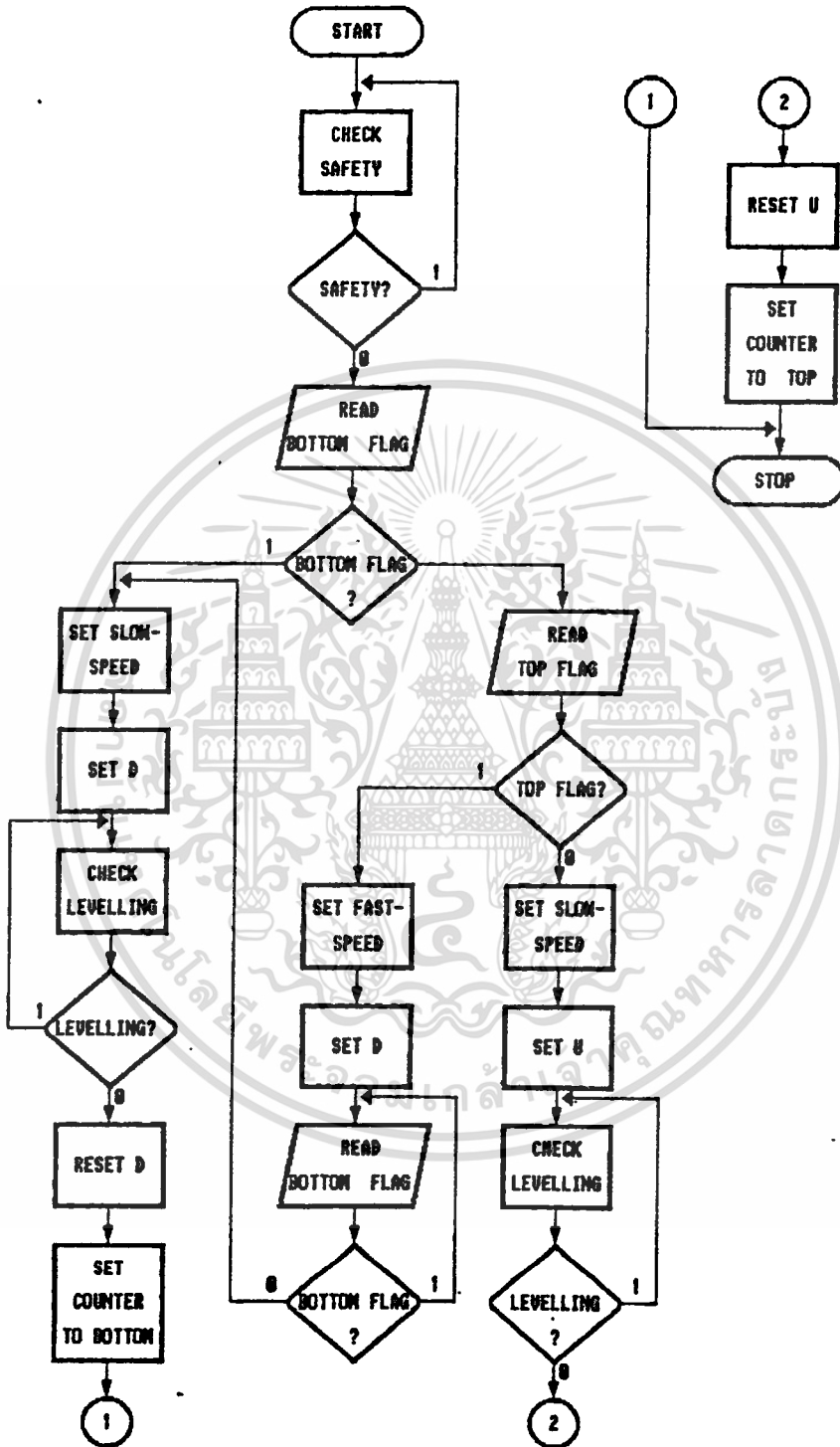
สำหรับรายละเอียดของการทำงานของโปรแกรมน้อยนี้ก็คือ ภายหลังจากที่ลิฟต์เข้าสู่ช่วงของสัญญาณโบธงแล้ว จะมีการเพิ่มค่าของตัวนับขึ้น โปรแกรมน้อยนี้จะอ่านสัญญาณที่มีการกดเรียกจากแผงควบคุมภายในลิฟต์ (สัญญาณ Car Call) ของชั้นที่ลิฟต์กำลังจะเข้าถึงในขณะนั้น ซึ่งสามารถที่จะตรวจสอบระดับขึ้นได้จากตัวนับชั้นนั่นเอง เมื่อมีการตรวจ

พบสัญญาณเรียกจากแผงควบคุมภายในลิฟท์ โปรแกรมจะรีเซ็ตสัญญาณการเข้าจอด (Stop-Signal) ที่ระดับชั้นนั้น ถ้าโปรแกรมตรวจไม่พบสัญญาณการเรียกจากแผงควบคุมภายในลิฟท์ จะต้องมีการตรวจสอบค่าของสัญญาณอ่านสัญญาณการเรียกจากแผงควบคุมที่อยู่หน้าประตู (Up and Down hall call) ถ้ามีการรีเซ็ตสัญญาณก่อนหน้านี้แล้ว แสดงว่าจะไม่มีการนำเอาสัญญาณเรียกจากหน้าประตูเข้ามาพิจารณา โปรแกรมจะรีเซ็ตสัญญาณการเข้าจอดที่ชั้นนั้น แต่ถ้าสถานะของสัญญาณดังกล่าวถูกยึดอยู่ นั่นคือจะต้องนำเอาสัญญาณเรียกจากหน้าประตูมาพิจารณาในการเข้าจอดของลิฟท์ โดยที่จะต้องตรวจสอบทิศทางการเคลื่อนที่ของลิฟท์ในขณะนั้นว่า มีทิศทางการเคลื่อนที่ขึ้น หรือ ลง ซึ่งสามารถที่จะตรวจสอบได้จากสัญญาณลิฟท์เคลื่อนที่ขึ้น หรือลงนั่นเอง (Direction up signal or Direction down Signal) สำหรับกรณีที่ลิฟท์กำลังเคลื่อนที่ขึ้น โปรแกรมจะตรวจสอบเฉพาะสัญญาณเรียกขึ้น (Up hall call) เท่านั้น ถ้าตรวจพบก็จะทำการรีเซ็ตสัญญาณการเข้าจอด และ เมื่อตรวจไม่พบก็หมายความว่าไม่ต้องเข้าจอดที่ชั้นนี้ โปรแกรมจะรีเซ็ตสัญญาณการเข้าจอดที่ชั้นนี้ ส่วนกรณีที่ลิฟท์กำลังเคลื่อนที่ในทิศลงนั้น จะมีการตรวจสอบสัญญาณเรียกลง จากแผงควบคุมหน้าประตู (Down hall Call) เมื่อตรวจพบก็จะรีเซ็ตสัญญาณการเข้าจอด และ เมื่อตรวจไม่พบก็จะรีเซ็ตสัญญาณการเข้าจอด เช่นเดียวกับกรณีที่ลิฟท์กำลังเคลื่อนที่ขึ้น

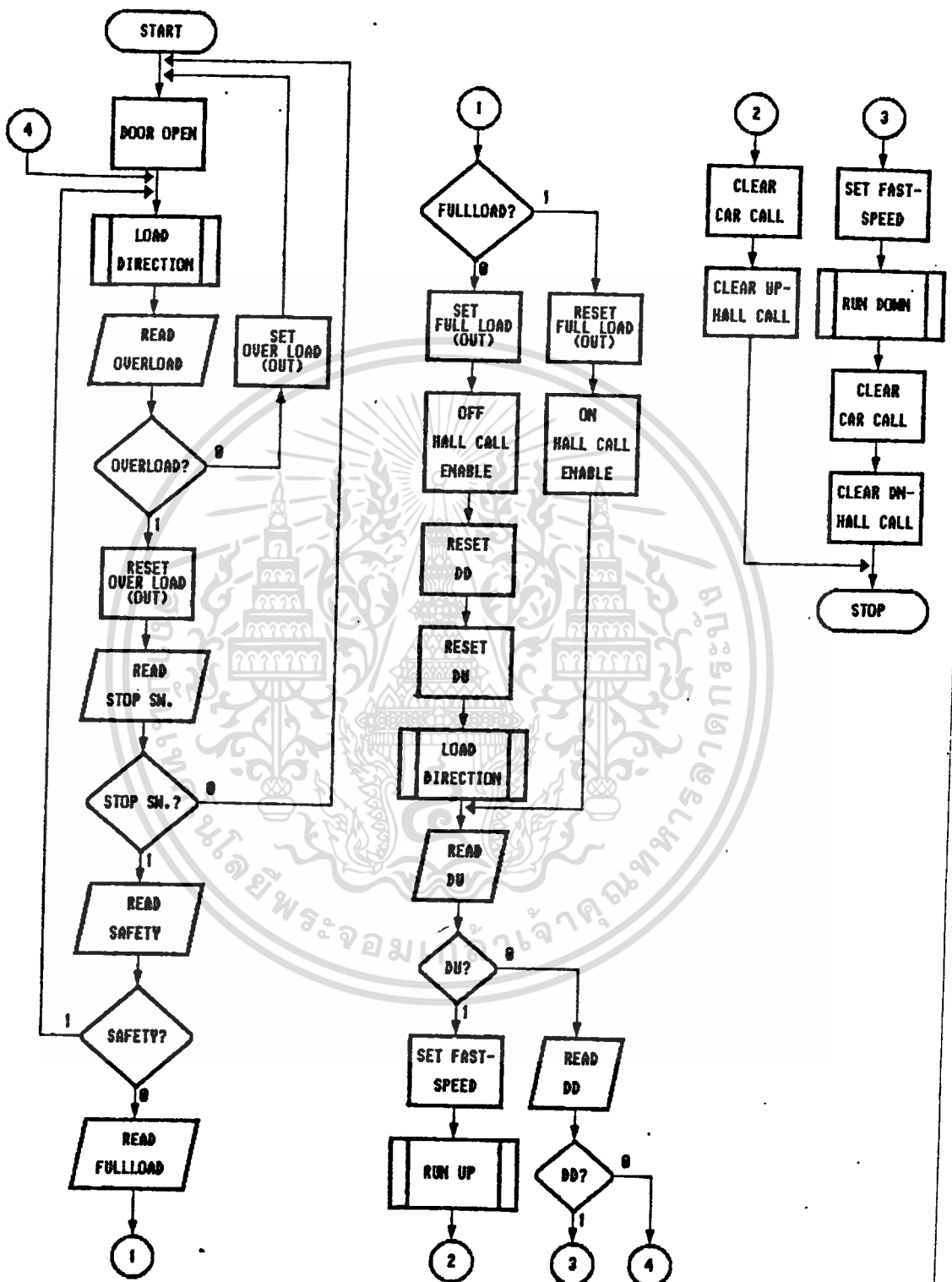
สำหรับในกรณีที่มีการทำงานในโปรแกรมน้อยนี้ โดยมีผลการตรวจสอบไม่พบสัญญาณเรียกจากแผงควบคุมภายในลิฟท์ และ สัญญาณทิศทางการเคลื่อนที่ขึ้น , สัญญาณการเคลื่อนที่ลง มีสถานะถูกยึดอยู่ ดังที่ได้กล่าวไว้ในตอนต้นแล้วว่า โปรแกรมย่อยนี้ จะถูกเรียกใช้ในขณะที่ลิฟท์กำลังเคลื่อนที่ขึ้น หรือ ลง ด้วยความเร็วสูง ในกรณีที่ผลการตรวจสอบดังกล่าวแสดงว่าเกิดความผิดพลาดขึ้น ไม่ว่าจะเป็นสาเหตุมาจากสิ่งใด ๆ ก็ตาม โปรแกรมจะสั่งให้ลิฟท์เข้าจอดในชั้นนั้นด้วย ซึ่งก็จะรีเซ็ตสัญญาณการเข้าจอดที่ชั้นนี้



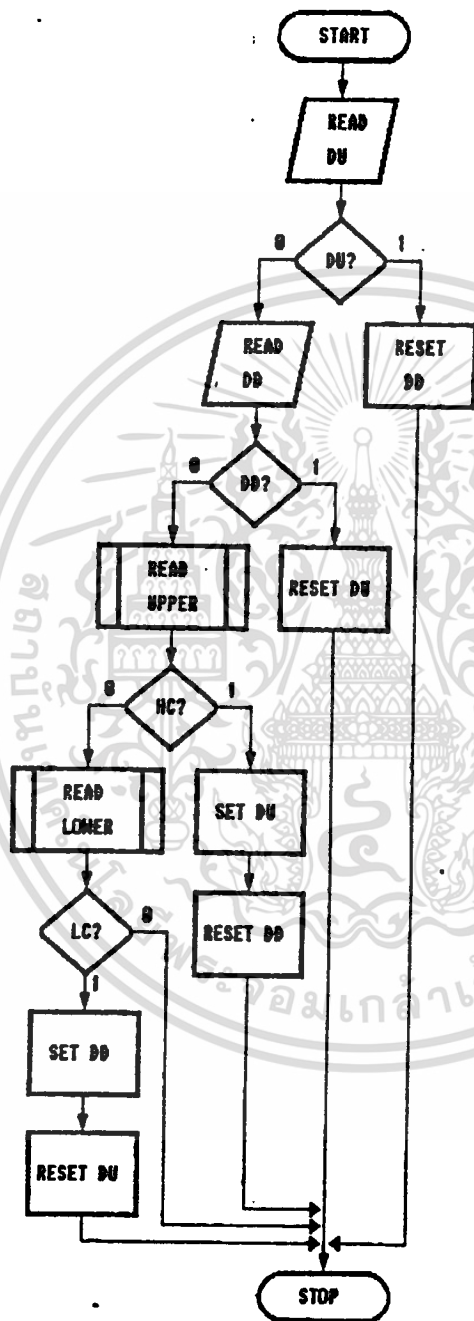
รูปที่ 3.4 ผลงานของโปรแกรมหลัก



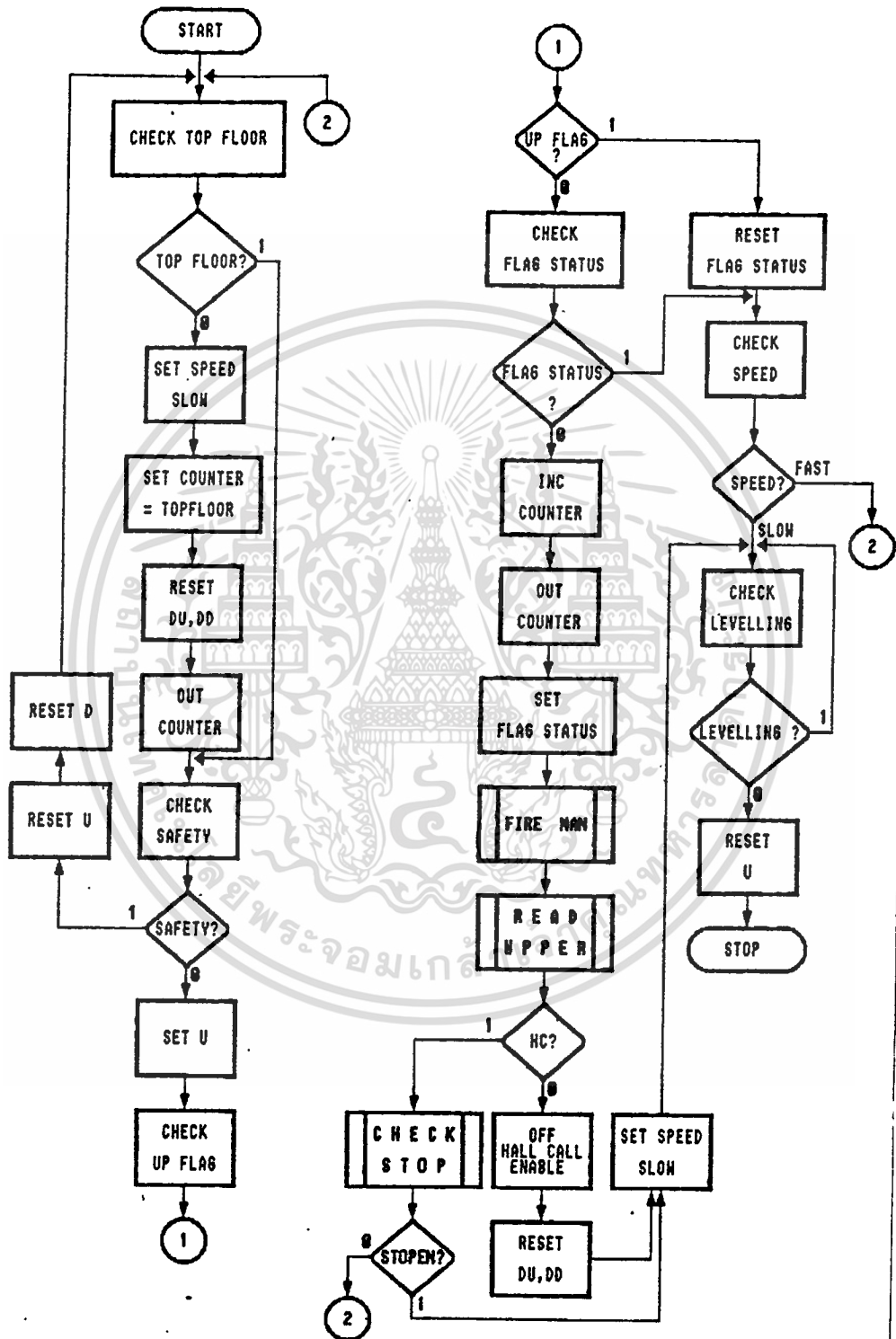
รูปที่ 3.5 ผังงานของโปรแกรมยกสิ่งของขึ้นลง



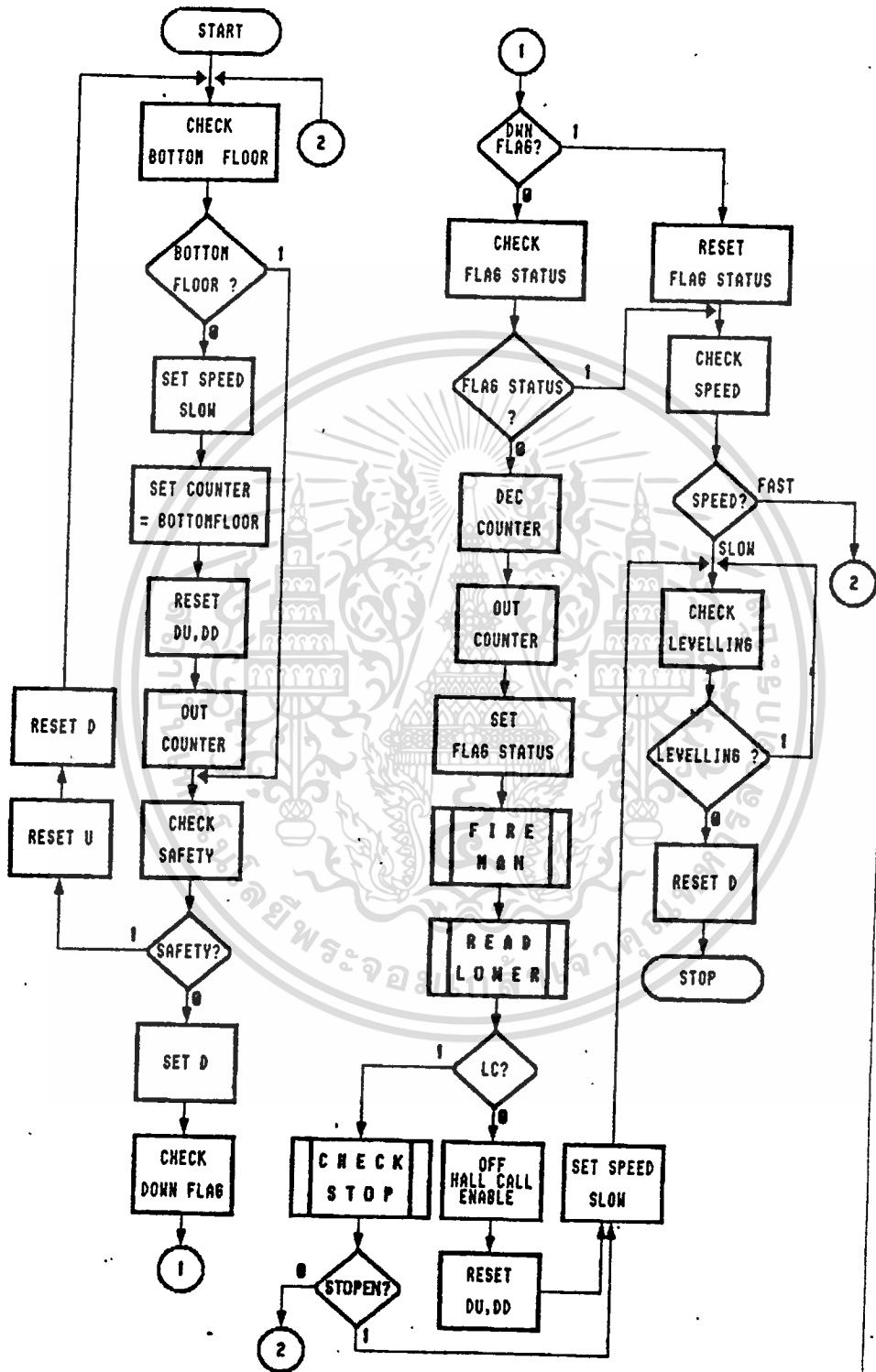
รูปที่ 3.6 ผลงานของโปรแกรมย่อยการเข้าจอดและความปลอดภัย



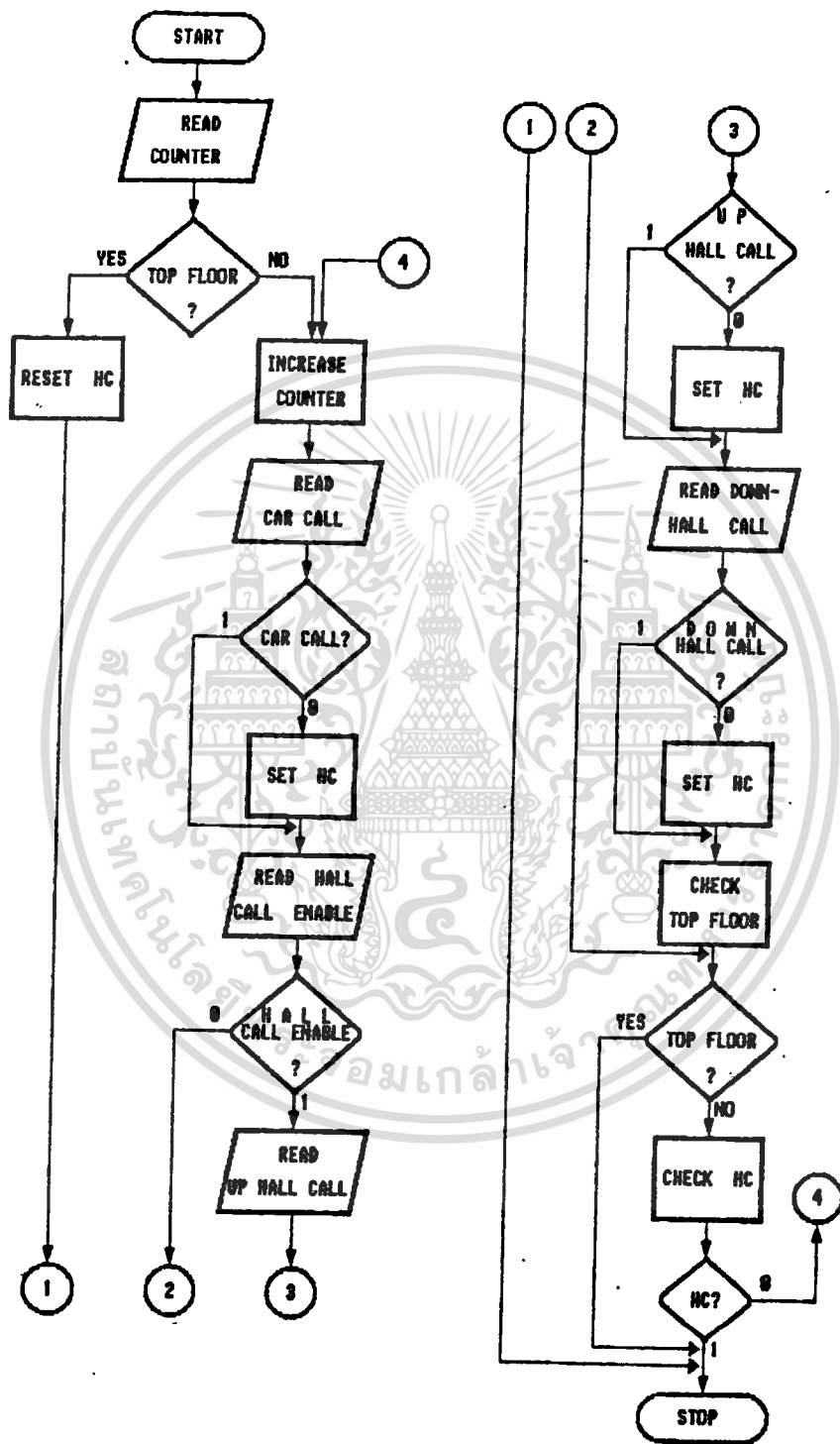
รูปที่ 3.7 ผังงานของโปรแกรมย่อยกำหนดทิศทาง การเคลื่อนที่



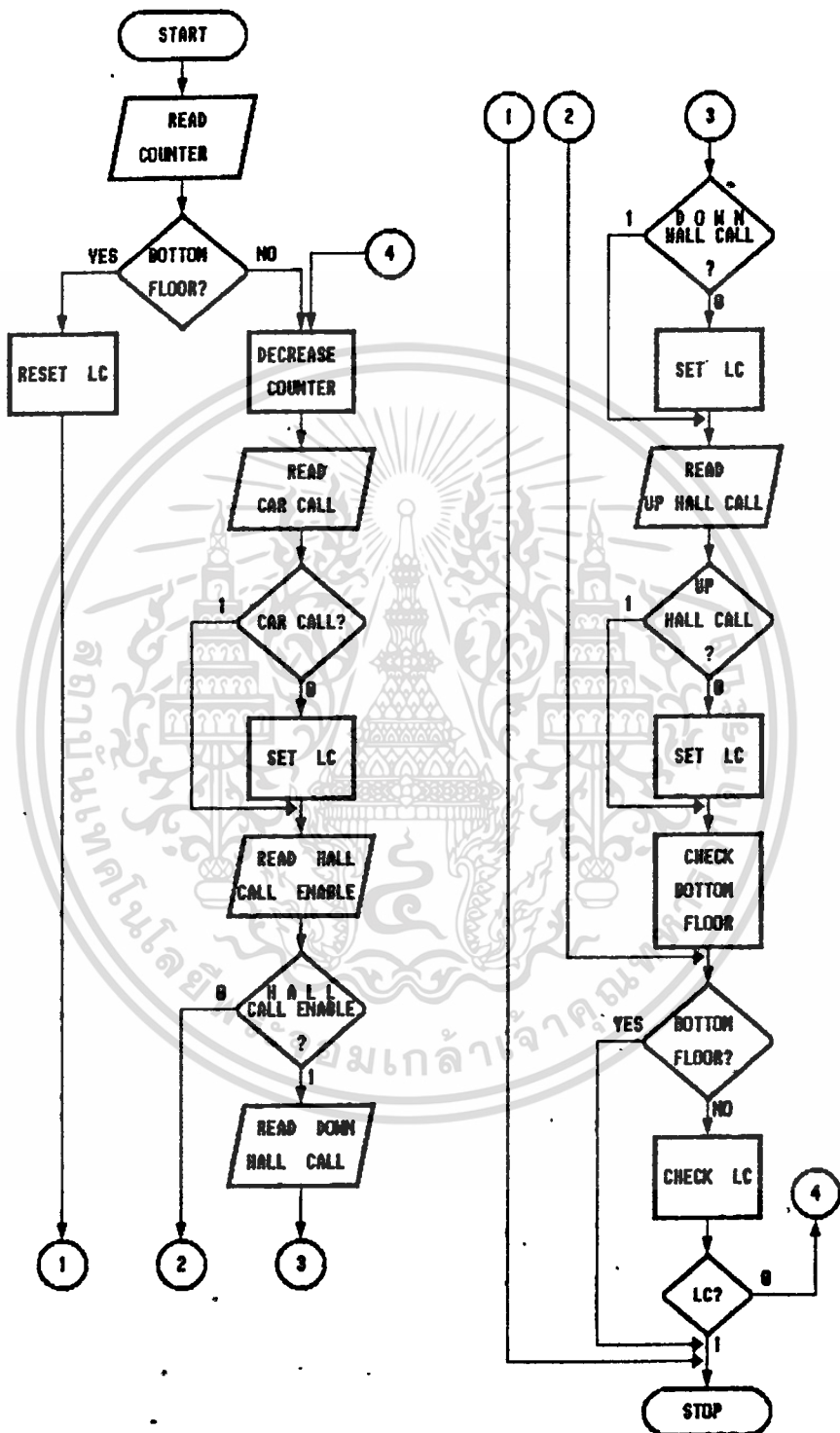
รูปที่ 3.8 ผังงานของโปรแกรมยอกลิฟต์เคลื่อนที่ขึ้น



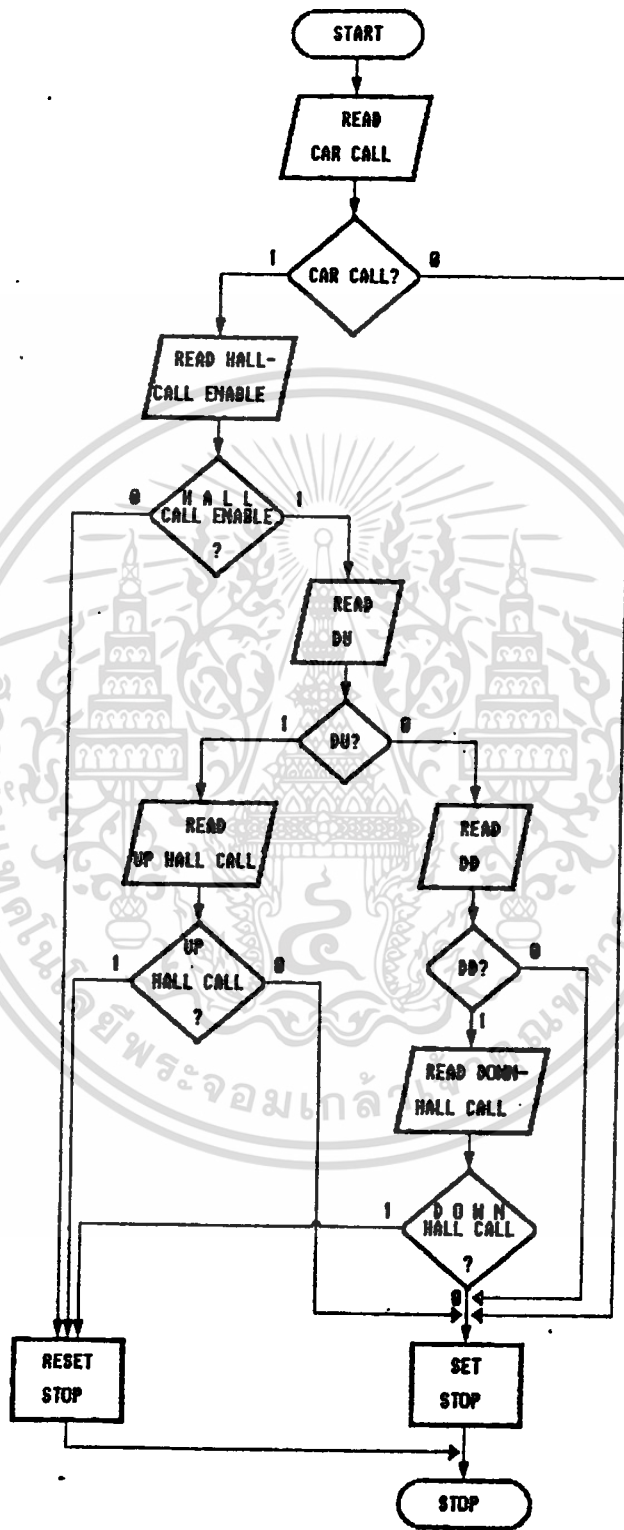
รูปที่ 3.9 ฟังก์ชันของโปรแกรมย่อยลิฟต์เคลื่อนที่ลง



รูปที่ 3.10 ฟังก์ชันของโปรแกรมอ่านสัญญาณเรียกจากพื้นที่สูงกว่า



รูปที่ 3.11 ผลงานของโปรแกรมย่อยอ่านสัญญาณเรียกจากชั้นที่ต่ำกว่า



รูปที่ 3.12 ฟังก์ชันของโปรแกรมย่อยตรวจสอบการเข้าจอดที่ระดับขึ้น

4. การทดสอบและผลการทดสอบ

เนื่องจากทดสอบ ไม่สามารถนำลิฟท์ที่จะใช้งานจริง มาใช้ทดสอบได้ ดังนั้น จึงต้องมีการสร้างลิฟท์จำลองขึ้นเพื่อใช้ทดสอบ โดยอาศัยการแสดงผลของแอลอีดี (LED) แทนการขับเคลื่อน และการทำงานของลิฟท์ แล้วนำผลที่ได้มาวิเคราะห์เปรียบเทียบกับการทำงานจริง ส่วนสัญญาณที่ป้อนให้กับแผงควบคุมต่าง ๆ ก็ใช้สัญญาณเปรียบเทียบแทนการใช้สัญญาณป้อนเข้าที่ใช้กับงานจริง โดยลิฟท์จำลองที่ใช้ทดสอบมีเพียง 3 ขึ้น

จากการทดสอบแผงที่ใช้ในการควบคุมลิฟท์แผงต่าง ๆ โดยใช้ชุดลิฟท์จำลอง พบว่าแผงควบคุมซึ่งใช้ในการควบคุมลิฟท์ สามารถควบคุมการทำงานในฟังก์ชันต่าง ๆ ของลิฟท์ได้ คือ สามารถนำสัญญาณเข้าต่าง ๆ มาประมวลผล และส่งสัญญาณออกเพื่อใช้ในการขับเคลื่อนลิฟท์ให้วิ่งขึ้น หรือ ลงในทิศทางที่ถูกต้อง สามารถหยุดจอดได้ตรงตามระดับขึ้นและทำงานในฟังก์ชันรักษาความปลอดภัย และฟังก์ชันอื่น ๆ ได้

5. สรุปผลการทดสอบ

ผลการนำเครื่องควบคุมลิฟต์ไมโครโปรเซสเซอร์ มาแทนที่เครื่องควบคุมลิฟต์แบบรีเลย์ สามารถสรุปได้ว่า มีข้อดี ข้อเสียดังต่อไปนี้

ข้อดี

1. ความสะดวกในการเปลี่ยนแปลงฟังก์ชันการทำงานของลิฟต์ จากที่ทราบมาแล้วว่า เครื่องควบคุมลิฟต์แบบรีเลย์ การเปลี่ยนแปลงฟังก์ชันการทำงานของลิฟต์จะต้องทำการแก้ไขวงจรรีเลย์ ซึ่งเป็นการแก้ไขฮาร์ดแวร์ของเครื่องควบคุม จึงไม่สะดวก แต่เครื่องควบคุมลิฟต์แบบไมโครโปรเซสเซอร์ การเปลี่ยนแปลงฟังก์ชันการทำงานของลิฟต์สามารถทำได้โดยการแก้ไขที่ซอฟต์แวร์ โดยไม่ต้องแก้ไขฮาร์ดแวร์ของเครื่องควบคุมเลย การเปลี่ยนแปลงแก้ไขจึงทำได้สะดวกกว่า

2. ความสะดวกในการดูแลรักษาและซ่อมบำรุง เครื่องควบคุมแบบไมโครโปรเซสเซอร์ จะมีขนาดเล็กกว่า จึงสามารถดูแลรักษาได้ง่าย ลักษณะของวงจรก็เป็นบอร์ดแยกกันแต่ละวงจร การซ่อมบำรุงจึงทำได้ง่ายโดยมีบอร์ดสำรองไว้เปลี่ยน นอกจากนี้การซ่อมบำรุงยังสามารถดูได้จากแอลอีดี บอกสถานะบนแผงอินเตอร์เฟส และทดสอบการทำงานของลิฟต์ได้ด้วยปุ่มกดทดสอบบนแผงอินเตอร์เฟส ซึ่งปุ่มกดทดสอบเหล่านี้ได้ต่อขนานกับปุ่มกดในตัวลิฟต์ และปุ่มกดหน้าขึ้นของลิฟต์จริง จึงเป็นการสะดวกที่ช่างลิฟต์ไม่ต้องลงไปกดปุ่มกดของลิฟต์จริง

ข้อเสีย

เมื่อเปรียบเทียบการควบคุมทั้งสองแบบ แทบจะไม่พบเลยว่าการควบคุมแบบไมโครโปรเซสเซอร์ จะมีข้อดีน้อยกว่าแบบควบคุมโดยใช้รีเลย์ นอกจากความยุ่งยากในการคำนวณออกแบบในครั้งแรกเริ่มต้นเท่านั้น



```

;
; PROGRAM : LIFT CONTROLLER
;
; BY : MR. THERDTON THEERASASANA ,KMITL
;

```

000000

CPU "8085.TBL"

;RAM ADDRESS DEFINITION

```

00000050 = BEGIN: EQU 50H
00002000 = HALLEN: EQU 2000H
00002001 = STOPEN: EQU 2001H
00002002 = FLAGST: EQU 2002H
00002003 = HICALL: EQU 2003H
00002004 = LOCALL: EQU 2004H
00002005 = TOPFLR: EQU 2005H

```

;I/O PORT ADDRESS DEFINITION

```

00000064 = TOPDIP: EQU 100
00000065 = COUNT: EQU 101
00000066 = PORTX1: EQU 102
00000067 = PORTX2: EQU 103
00000068 = PORTX3: EQU 104
000000E8 = CARCLL: EQU 232
000000F0 = DNHALL: EQU 240
000000FB = UPHALL: EQU 248

```

000000

; ORG 0

000000 C35000

JMP BEGIN

000050

ORG BEGIN

```

;
; MAIN PROGRAM
;

```

```

000050 210020 LXI H,HALLEN
000053 3600 MVI M,0
000055 210120 LXI H,STOPEN
000058 3600 MVI M,0
00005A 210220 LXI H,FLAGST
00005D 3600 MVI M,0
00005F 210320 LXI H,HICALL
000062 3600 MVI M,0
000064 210420 LXI H,LOCALL
000067 3600 MVI M,0
000069 210520 LXI H,TOPFLR
00006C 3600 MVI M,0
00006E CD7E00 CALL INIT

```

```

000071 3E00      MVI A,00000000B
000073 D365      OUT COUNT
000075 DB64      IN TOPDIP
000077 210520    LXI H,TOPFLR
00007A 77        MOV M,A
00007B CD7A02    CALL STD00R

```

SUBPROGRAM : INITIALIZE

```

00007E          INIT:
00007E DB66      IN PORTX1          ;READ SAFETY SWITCH
000080 47        MOV B,A           ;PORT : PORTX1
000081 3E02      MVI A,2           ;BIT : 2
000083 CD3705    CALL BITTST
000086 C27E00    JNZ INIT
000089 DB67      IN PORTX2          ;READ BOTTOM FLOOR FLAG
00008B 47        MOV B,A           ;PORT : PORTX2
00008C 3E06      MVI A,6           ;BIT : 6
00008E CD3705    CALL BITTST
000091 CAA300    JZ INIT2
000094 3EE7      MVI A,11100111B   ;SET FAST SPEED AND DOWN
000096 D368      OUT PORTX3
000098 DB67      INIT1: IN PORTX2   ;READ BOTTOM FLOOR FLAG
00009A 47        MOV B,A           ;PORT : PORTX2
00009B 3E06      MVI A,6           ;BIT : 6
00009D CD3705    CALL BITTST
0000A0 C29B00    JNZ INIT1
0000A3 3EEB      INIT2: MVI A,11101011B ;SET SLOW SPEED AND DOWN
0000A5 D368      OUT PORTX3
0000A7 DB67      INIT3: IN PORTX2   ;READ LEVELLING
0000A9 47        MOV B,A           ;PORT : PORTX2
0000AA 3E03      MVI A,3           ;BIT : 3
0000AC CD3705    CALL BITTST
0000AF C2A700    JNZ INIT3
0000B2 3EFF      MVI A,11111111B   ;RESET OUTPUT PORT : PORTX3
0000B4 D368      OUT PORTX3        ;TO STOP RUNNING DOWN
0000B6 C9        RET

```

SUBPROGRAM : READ LOWER

```

0000B7          READLO:
0000B7 DB65      IN COUNT          ;READ COUNTER PORT : COUNT
0000B9 E63F      ANI 00111111B     ;MASK BIT 6 AND 7 : DD AND DU
0000BB 4F        MOV C,A
0000BC FE00      CPI 0             ;COMPARE WITH BOTTOM FLOOR
0000BE C2C900    JNZ READL1
0000C1 210420    LXI H,LOCALL      ;POINT HL TO LOWER CALL ADDR.
0000C4 3600      MVI M,0           ;RESET LOWER CALL

```

0000C6 C32F01	JMP READL5	
0000C9 0D	READL1: DCR C	;COUNTER = COUNTER - 1
0000CA 79	MOV A,C	;CALCULATE OFFSET
0000CB 1F	RAR	
0000CC 1F	RAR	
0000CD 1F	RAR	
0000CE C6E8	ADI CARCLL	;ADD OFFSET TO CAR CALL PORT ADDR.
0000D0 320A05	STA PORTIN	
0000D3 CD0905	CALL INPUT	
0000D6 47	MOV B,A	
0000D7 79	MOV A,C	
0000DB E607	ANI 00000111B	
0000DA CD3705	CALL BITTST	;READ CAR CALL
0000DD C2E500	JNZ READL2	
0000E0 210420	LXI H,LOCALL	;POINT HL TO LOWER CALL ADDR.
0000E3 3601	MVI M,1	;SET LOWER CALL
0000E5 210020	READL2: LXI H,HALLEN	;POINT HL TO HALL CALL ENABLE ADDR.
0000E8 7E	MOV A,M	;CHECK HALL CALL ENABLE
0000E9 CA2201	JZ READL4	
0000EC 79	MOV A,C	;CALCULATE OFFSET
0000ED 1F	RAR	
0000EE 1F	RAR	
0000EF 1F	RAR	
0000F0 C6F8	ADI UPHALL	;ADD OFFSET TO UP HALL CALL PORT ADDR.
0000F2 320A05	STA PORTIN	
0000F5 CD0905	CALL INPUT	
0000F8 47	MOV B,A	
0000F9 79	MOV A,C	
0000FA E607	ANI 00000111B	
0000FC CD3705	CALL BITTST	;READ UP HALL CALL
0000FF C20701	JNZ READL3	
000102 210420	LXI H,LOCALL	;POINT HL TO LOWER CALL ADDR.
000105 3601	MVI M,1	;SET LOWER CALL
000107 4F	READL3: MOV C,A	;CALCULATE OFFSET
000108 1F	RAR	
000109 1F	RAR	
00010A 1F	RAR	
00010B C6F0	ADI DNHALL	;ADD OFFSET TO DOWN HALL CALL PORT ADDR
00010D 320A05	STA PORTIN	
000110 CD0905	CALL INPUT	
000113 47	MOV B,A	
000114 79	MOV A,C	
000115 E607	ANI 00000111B	
000117 CD3705	CALL BITTST	;READ DOWN HALL CALL
00011A C22201	JNZ READL4	
00011D 210420	LXI H,LOCALL	;POINT HL TO LOWER CALL ADDR.
000120 3601	MVI M,1	;SET LOWER CALL
000122 79	READL4: MOV A,C	
000123 FE00	CPI 0	;COMPARE WITH BOTTOM FLOOR
000125 CA2F01	JZ READL5	

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

00012B 210420      LXI H,LOCALL      ;POINT HL TO LOWER CALL ADDR.
00012B 7E          MOV A,M
00012C CAC900      JZ READL1
00012F C9          READL5: RET

```

```

;
; SUBPROGRAM : READ UPPER
;

```

```

000130      READUP:
000130 DB65      IN COUNT      ;READ COUNTER PORT : COUNT
000132 E63F      ANI 00111111B  ;MASK BIT 6 AND 7 : DD AND DU
000134 4F        MOV C,A
000135 210520     LXI H,TOFFLR   ;POINT HL TO TOP FLOOR ADDR.
000138 BE        CMP M          ;COMPARE COUNTER WITH TOP FLOOR
000139 C24401     JNZ READU1
00013C 210320     LXI H,HICALL   ;POINT HL TO HIGHER CALL ADDR.
00013F 3600      MVI M,0        ;RESET HIGHER CALL
000141 C3AC01     JMP READU5
000144 0C        READU1: INR C    ;COUNTER = COUNTER + 1
000145 79        MOV A,C        ;CALCULATE OFFSET
000146 1F        RAR
000147 1F        RAR
000148 1F        RAR
000149 C6EB      ADI CARCLL      ;ADD OFFSET TO CAR CALL PORT ADDR.
00014B 320A05     STA PORTIN
00014E CD0905     CALL INPUT
000151 47        MOV B,A
000152 79        MOV A,C
000153 E607      ANI 00000111B
000155 CD3705     CALL BITTST    ;READ CAR CALL
000158 C26001     JNZ READU2
00015B 210320     LXI H,HICALL   ;POINT HL TO HIGHER CALL ADDR.
00015E 3601      MVI M,1        ;SET HIGHER CALL
000160 210020     READU2: LXI H,HALLN  ;POINT HL TO HALL CALL ENABLE ADDR.
000163 7E        MOV A,M        ;CHECK HALL CALL ENABLE
000164 CA9D01     JZ READU4
000167 79        MOV A,C        ;CALCULATE OFFSET
000168 1F        RAR
000169 1F        RAR
00016A 1F        RAR
00016B C6FB      ADI UPHALL      ;ADD OFFSET TO UP HALL CALL PORT ADDR.
00016D 320A05     STA PORTIN
000170 CD0905     CALL INPUT
000173 47        MOV B,A
000174 79        MOV A,C
000175 E607      ANI 00000111B
000177 CD3705     CALL BITTST    ;READ UP HALL CALL
00017A C28201     JNZ READU3
00017D 210320     LXI H,HICALL   ;POINT HL TO HIGHER CALL ADDR.
000180 3601      MVI M,1        ;SET HIGHER CALL

```

```

000182 79          READU3: MOV A,C          ;CALCULATE OFFSET
000183 1F          RAR
000184 1F          RAR
000185 1F          RAR
000186 C6F0        ADI DNHALL              ;ADD OFFSET TO DOWN HALL CALL ADDR.
000188 320A05      STA PORTIN
00018B CD0905      CALL INPUT
00018E 47          MOV B,A
00018F 79          MOV A,C
000190 E607        ANI 00000111B
000192 CD3705      CALL BITTST            ;READ DOWN HALL CALL
000195 C29D01      JNZ READU4
000198 210320      LXI H,HICALL           ;POINT HL TO HIGHER CALL ADDR.
00019B 3601        MVI M,1               ;SET HIGHER CALL
00019D 79          READU4: MOV A,C
00019E 210520      LXI H,TOPLR
0001A1 BE          CMP M                 ;COMPARE COUNTER TO TOP FLOOR
0001A2 CAAC01      JZ READU5
0001A5 210320      LXI H,HICALL           ;POINT HL TO HIGHER CALL PORT ADDR.
0001AB 7E          MOV A,M               ;CHECK HIGHER CALL
0001A9 C24401      JNZ READU1
0001AC C9          READU5: RET

```

SUBPROGRAM : CHECK STOP

```

0001AD          CHKSTP:
0001AD DB65        IN COUNT              ;READ COUNTER PORT : COUNT
0001AF 57          MOV D,A
0001B0 E63F        ANI 00111111B         ;MASK BIT 6 AND 7 : DD AND DU
0001B2 4F          MOV C,A
0001B3 1F          RAR                   ;CALCULATE OFFSET
0001B4 1F          RAR
0001B5 1F          RAR
0001B6 C6E8        ADI CARCLL            ;ADD OFFSET TO CAR CALL PORT ADDR.
0001B8 320A05      STA PORTIN
0001BB CD0905      CALL INPUT
0001BE 47          MOV B,A
0001BF 79          MOV A,C
0001C0 E607        ANI 00000111B
0001C2 CD3705      CALL BITTST            ;READ CAR CALL
0001C5 CA1B02      JZ CHKST3
0001C8 210020      LXI H,HALLN           ;POINT HL TO HALL CALL ENABLE
0001CB 7E          MOV A,M               ;CHECK HALL CALL ENABLE
0001CC CA1302      JZ CHKST2
0001CF 42          MOV B,D               ;READ DIRECTION UP BIT
0001D0 3E07        MVI A,7               ;PORT : COUNT
0001D2 CD3705      CALL BITTST            ;BIT : 7
0001D5 CAF101      JZ CHKST1
0001D8 79          MOV A,C               ;CALCULATE OFFSET

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

0001D9 1F      RAR
0001DA 1F      RAR
0001DB 1F      RAR
0001DC C6F8    ADI UPHALL          ;ADD OFFSET TO UP HALL CALL PORT ADDR.
0001DE 320A05  STA PORTIN
0001E1 CD0905  CALL INPUT
0001E4 47      MOV B,A
0001E5 79      MOV A,C
0001E6 E607    ANI 00000111B
0001EB CD3705  CALL BITTST          ;READ UP HALL CALL
0001EB C21302  JNZ CHKST2
0001EE C31B02  JMP CHKST3
0001F1 42      CHKST1: MOV B,D          ;READ DIRECTION DOWN BIT
0001F2 3E06    MVI A,6          ;PORT : COUNT
0001F4 CD3705  CALL BITTST          ;BIT : 6
0001F7 CA1B02  JZ CHKST3
0001FA 79      MOV A,C          ;CALCULATE OFFSET
0001FB 1F      RAR
0001FC 1F      RAR
0001FD 1F      RAR
0001FE C6F0    ADI DNHALL          ;ADD OFFSET TO DOWN HALL CALL PORT ADDR.
000200 320A05  STA PORTIN
000203 CD0905  CALL INPUT
000206 47      MOV B,A
000207 79      MOV A,C
000208 E607    ANI 00000111B
00020A CD3705  CALL BITTST          ;READ DOWN HALL CALL
00020D C21302  JNZ CHKST2
000210 C31B02  JMP CHKST3
000213 210120  CHKST2: LXI H,STOPEN      ;POINT HL TO STOP ENABLE ADDR.
000216 3600    MVI M,0          ;RESET STOP ENABLE
000218 C32002  JMP CHKST4
00021B 210120  CHKST3: LXI H,STOPEN      ;POINT HL TO STOP ENABLE ADDR.
00021E 3601    MVI M,1          ;SET STOP ENABLE
000220 C9      CHKST4: RET

```

```

;
; SUBPROGRAM : LOAD DIRECTION
;

```

```

000221 DIRECT:
000221 DB65    IN COUNT          ;READ COUNTER PORT : COUNT
000223 47      MOV B,A          ;READ DIRECTION UP BIT
000224 3E07    MVI A,7          ;PORT : COUNT
000226 CD3705  CALL BITTST          ;BIT : 7
000229 CA3602  JZ DIREC1
00022C 3E06    MVI A,6          ;RESET DIRECTION DOWN BIT
00022E CD2305  CALL BITCLR
000231 D365    OUT COUNT
000233 C37902  JMP DIREC4
000236 3E06    DIREC1: MVI A,6          ;READ DIRECTION DOWN BIT

```

```

000238 CD3705      CALL BITTST      ;PORT : COUNT , BIT : 6
00023B CA4802      JZ  DIREC2
00023E 3E07        MVI  A,7      ;RESET DIRECTION UP BIT
000240 CD2305      CALL BITCLR
000243 D365        OUT  COUNT
000245 C37902      JMP  DIREC4
000248 CD3001      DIREC2: CALL READUP      ;SUBPROGRAM : READ UPPER
00024B 210320      LXI  H,HICALL      ;POINT HL TO HIGHER CALL ADDR.
00024E 7E          MOV  A,M      ;CHECK HIGHER CALL
00024F CA6202      JZ  DIREC3
000252 3E07        MVI  A,7      ;SET DIRECTION UP BIT
000254 CD0F05      CALL BITSET
000257 47          MOV  B,A
000258 3E06        MVI  A,6      ;RESET DIRECTION DOWN BIT
00025A CD2305      CALL BITCLR
00025D D365        OUT  COUNT
00025F C37902      JMP  DIREC4
000262 CDB700      DIREC3: CALL READLO      ;SUBPROGRAM : READ LOWER
000265 210420      LXI  H,LOCALL      ;POINT HL TO LOWER CALL ADDR.
000268 7E          MOV  A,M      ;CHECK LOWER CALL
000269 CA7902      JZ  DIREC4
00026C 3E07        MVI  A,7      ;RESET DIRECTION UP BIT
00026E CD2305      CALL BITCLR
000271 47          MOV  B,A
000272 3E06        MVI  A,6      ;SET DIRECTION DOWN BIT
000274 CD0F05      CALL BITSET
000277 D365        OUT  COUNT
000279 C9          DIREC4: RET

```

SUBPROGRAM : STOP & DOOR

```

00027A      STD00R:
00027A 3EFD        MVI  A,11111101B      ;SEND DOOR OPEN PULSE
00027C D368        OUT  PORTX3
00027E CDF604      CALL  DELAY      ;PULSE WIDTH 2 SECOND
000281 3EFF        MVI  A,11111111B
000283 D368        OUT  PORTX3
000285 CD2102      STD001: CALL DIRECT      ;SUBPROGRAM : LOAD DIRECTION
000288 DB66        IN   PORTX1      ;READ OVERLOAD BIT
00028A 47          MOV  B,A      ;PORT : PORTX1
00028B 3E07        MVI  A,7      ;BIT : 7
00028D CD3705      CALL BITTST
000290 C29A02      JNZ  STD002
000293 3E7F        MVI  A,01111111B      ;SET OVERLOAD OUTPUT
000295 D368        OUT  PORTX3      ;PORT : PORTX3 , BIT : 7
000297 C37A02      JMP  STD00R
00029A 3EFF        STD002: MVI  A,11111111B      ;CLEAR OVERLOAD OUTPUT
00029C D368        OUT  PORTX3
00029E DB66        IN   PORTX1      ;READ STOP SWITCH

```

0002A0 47	MOV B,A	;PORT : PORTX1
0002A1 3E01	MVI A,1	;BIT : 1
0002A3 CD3705	CALL BITTST	
0002A6 CA7A02	JZ STDOOR	
0002A9 DB66	IN PORTX1	;READ SAFETY BIT
0002AB 47	MOV B,A	;PORT : PORTX1
0002AC 3E02	MVI A,2	;BIT : 2
0002AE CD3705	CALL BITTST	
0002B1 C2B502	JNZ STDD01	
0002B4 DB66	IN PORTX1	;READ FULLLOAD BIT
0002B6 47	MOV B,A	;PORT : PORTX1
0002B7 3E06	MVI A,6	;BIT : 6
0002B9 CD3705	CALL BITTST	
0002BC CACB02	JZ STDD03	
0002BF 3EFF	MVI A,11111111B	;RESET FULLLOAD OUTPUT
0002C1 D368	OUT PORTX3	;PORT : PORTX3 , BIT : 6
0002C3 210020	LXI H,HALLCN	;POINT HL TO HALL CALL ENABLE
0002C6 3601	MVI M,1	;SET HALL CALL ENABLE
0002C8 C3E702	JMP STDD04	
0002CB 3EBF	STDD03: MVI A,10111111B	;SET FULLLOAD OUTPUT
0002CD D368	OUT PORTX3	;PORT : PORTX3 , BIT 6
0002CF 210020	LXI H,HALLCN	;POINT HL TO HALL CALL ENABLE
0002D2 3600	MVI M,0	;RESET HALL CALL ENABLE
0002D4 DB65	IN COUNT	;READ COUNTER PORT : COUNT
0002D6 47	MOV B,A	
0002D7 3E07	MVI A,7	
0002D9 CD2305	CALL BITCLR	;RESET DIRECTION UP BIT
0002DC 47	MOV B,A	
0002DD 3E06	MVI A,6	
0002DF CD2305	CALL BITCLR	;RESET DIRECTION DOWN BIT
0002E2 D365	OUT COUNT	
0002E4 CD2102	CALL DIRECT	
0002E7 DB65	STDD04: IN COUNT	;READ COUNTER PORT : COUNT
0002E9 47	MOV B,A	;READ DIRECTION UP BIT
0002EA 3E07	MVI A,7	;PORT : COUNT
0002EC CD3705	CALL BITTST	;BIT : 7
0002EF CA3C03	JZ STDD05	
0002F2 3E7B	MVI A,1111011B	;SET FAST SPEED BIT
0002F4 D368	OUT PORTX3	;PORT : PORTX3 , BIT : 3
0002F6 CDBF03	CALL RUNUP	;SUBPROGRAM : RUN UP
0002F9 DB65	IN COUNT	;READ COUNTER PORT : COUNT
0002FB 4F	MOV C,A	
0002FC E63F	ANI 00111111B	;CALCULATE OFFSET
0002FE 1F	RAR	
0002FF 1F	RAR	
000300 1F	RAR	
000301 C6E8	ADI CARCLL	;ADD OFFSET TO CARCALL PORT ADDR.
000303 57	MOV D,A	
000304 320A05	STA PORTIN	
000307 CD0905	CALL INPUT	

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

00030A 47	MOV B,A	
00030B 79	MOV A,C	
00030C E607	ANI 00000111B	
00030E CD0F05	CALL BITSET	;CLEAR CAR CALL
000311 47	MOV B,A	
000312 7A	MOV A,D	
000313 320D05	STA PORTOT	
000316 78	MOV A,B	
000317 CD0C05	CALL OUTPUT	
00031A 79	MOV A,C	;CALCULATE OFFSET
00031B E61F	ANI 0011111B	
00031D 1F	RAR	
00031E 1F	RAR	
00031F 1F	RAR	
000320 C6FB	ADI UPHALL	;ADD OFFSET TO UP HALL CALL PORT ADDR.
000322 57	MOV D,A	
000323 320A05	STA PORTIN	
000326 CD0905	CALL INPUT	
000329 47	MOV B,A	
00032A 79	MOV A,C	
00032B E607	ANI 00000111B	
00032D CD0F05	CALL BITSET	;CLEAR UP HALL CALL
000330 47	MOV B,A	
000331 7A	MOV A,D	
000332 320D05	STA PORTOT	
000335 78	MOV A,B	
000336 CD0C05	CALL OUTPUT	
000339 C38E03	JMP STD006	
00033C DB65	STD005: IN COUNT	;READ COUNTER PORT : COUNT
00033E 47	MOV B,A	;READ DIRECTION DOWN BIT
00033F 3E06	MVI A,6	;PORT : COUNT
000341 CD3705	CALL BITTST	;BIT : 6
000344 CA7A02	JZ STD00R	
000347 3EF7	MVI A,11110111B	;SET FAST SPEED
000349 D368	OUT PORTX3	;PORT : PORTX3 , BIT : 3
00034B CD4E04	CALL RUNDN	;SUBPROGRAM : RUN DOWN
00034E DB65	IN COUNT	;READ COUNTER PORT : COUNT
000350 4F	MOV C,A	
000351 E63F	ANI 00111111B	;CALCULATE OFFSET
000353 1F	RAR	
000354 1F	RAR	
000355 1F	RAR	
000356 C6E8	ADI CARCLL	;ADD OFFSET TO CARCALL PORT ADDR.
000358 57	MOV D,A	
000359 320A05	STA PORTIN	
00035C CD0905	CALL INPUT	
00035F 47	MOV B,A	
000360 79	MOV A,C	
000361 E607	ANI 00000111B	
000363 CD0F05	CALL BITSET	;CLEAR CAR CALL BIT

```

000366 47      MOV B,A
000367 7A      MOV A,D
000368 320D05   STA PORTOT
00036B 78      MOV A,B
00036C CD0C05   CALL OUTPUT
00036F 79      MOV A,C                      ;CALCULATE OFFSET
000370 E63F     ANI 00111111B
000372 1F      RAR
000373 1F      RAR
000374 1F      RAR
000375 E6F0     ADI DNHALL                      ;ADD OFFSET TO DOWN HALL CALL
000377 57      MOV D,A
000378 320A05   STA PORTIN
00037B CD0905   CALL INPUT
00037E 47      MOV B,A
00037F 79      MOV A,C
000380 E607     ANI 000000111B
000382 CD0F05   CALL BITSET                      ;CLEAR DOWN HALL CALL BIT
000385 47      MOV B,A
000386 7A      MOV A,D
000387 320D05   STA PORTOT
00038A 78      MOV A,B
00038B CD0C05   CALL OUTPUT
00038E C9      STD006: RET

```

SUBPROGRAM : RUN UP

```

00038F      RUNUP:
00038F DB67     IN PORTX2                      ;READ TOP FLOOR FLAG
000391 47      MOV B,A                      ;PORT : PORTX2
000392 3E07     MVI A,7                      ;BIT : 7
000394 CD3705   CALL BITTST
000397 C2BE03   JNZ RUNUP1
00039A DB68     IN PORTX3
00039C 47      MOV B,A                      ;RESET SPEED FAST BIT
00039D 3E03     MVI A,3                      ;PORT : PORTX3
00039F CD0F05   CALL BITSET                      ;BIT : 3
0003A2 47      MOV B,A                      ;SET SPEED SLOW BIT
0003A3 3E02     MVI A,2                      ;PORT : PORTX3
0003A5 CD2305   CALL BITCLR                      ;BIT : 2
0003AB D368     OUT PORTX3
0003AA DB64     IN TOPDIP                      ;READ TOP FLOOR DIP SWITCH
0003AC 210520   LXI H,TOPFLR                  ;POINT HL TO TOP FLOOR ADDR.
0003AF 77      MOV M,A
0003B0 47      MOV B,A                      ;CLEAR DIRECTION DOWN BIT
0003B1 3E06     MVI A,6                      ;PORT : PORTX3
0003B3 CD0F05   CALL BITSET                      ;BIT : 6
0003B6 47      MOV B,A                      ;CLEAR DIRECTION UP BIT
0003B7 3E07     MVI A,7                      ;PORT : PORTX3

```

0003B9 CD0F05	CALL BITSET	;BIT : 7
0003BC D365	OUT COUNT	
0003BE DB66	RUNUP1: IN PORTX1	;READ SAFETY BIT
0003C0 47	MOV B,A	;PORT : PORTX1
0003C1 3E02	MVI A,2	;BIT : 2
0003C3 CD3705	CALL BITTST	
0003C6 CAD003	JZ RUNUP2	
0003C9 DB68	IN PORTX3	;RESET UP RUNNING
0003CB F630	ORI 00110000B	;RESET DOWN RUNNING
0003CD C38F03	JMP RUNUP	
0003D0 DB68	RUNUP2: IN PORTX3	
0003D2 47	MOV B,A	
0003D3 3E05	MVI A,5	
0003D5 CD2305	CALL BITCLR	;SET UP RUNNING
0003D8 47	MOV B,A	
0003D9 3E04	MVI A,4	
0003DB CD0F05	CALL BITSET	;RESET DOWN RUNNING
0003DE D368	OUT PORTX3	
0003E0 DB67	IN PORTX2	;READ UP FLAG BIT
0003E2 47	MOV B,A	;PORT : PORTX2
0003E3 3E05	MVI A,5	;BIT : 5
0003E5 CD3705	CALL BITTST	
0003E8 C22E04	JNZ RUNUP5	
0003EB 210220	LXI H,FLAGST	;POINT HL TO FLAG STATUS ADDR.
0003EE 7E	MOV A,M	;CHECK FLAG STATUS
0003EF C23304	JNZ RUNUP6	
0003F2 DB65	IN COUNT	
0003F4 3C	INR A	;INCREASE COUNTER
0003F5 D365	OUT COUNT	
0003F7 3601	MVI M,1	;SET FLAG STATUS
0003F9 CD3001	CALL READUP	
0003FC 210320	LXI H,HICALL	;POINT HL TO HIGHER CALL ADDR.
0003FF 7E	MOV A,M	;CHECK HIGHER CALL
000400 CA1004	JZ RUNUP3	
000403 CDAD01	CALL CHKSTP	;SUBPROGRAM : CHECK STOP
000406 210120	LXI H,STOPEN	;POINT HL TO STOP ENABLE ADDR.
000409 7E	MOV A,M	;CHECK STOP ENABLE
00040A C8BF03	JZ RUNUP	
00040D C31B04	JMP RUNUP4	
000410 210020	RUNUP3: LXI H,HALLEN	;POINT HL TO HALL CALL ENABLE ADDR.
000413 3600	MVI M,0	;RESET HALL CALL ENABLE
000415 DB65	IN COUNT	;RESET DIRECTION DOWN : DD
000417 E63F	ANI 00111111B	;RESET DIRECTION UP : DU
000419 D365	OUT COUNT	
00041B DB68	RUNUP4: IN PORTX3	;SET SLOW SPEED BIT
00041D 47	MOV B,A	;PORT : PORTX3
00041E 3E02	MVI A,2	;BIT : 2
000420 CD2305	CALL BITCLR	
000423 47	MOV B,A	;RESET FAST SPEED BIT
000424 3E03	MVI A,3	;PORT : PORTX3

```

000426 CD0F05      CALL BITSET                ;BIT : 3
000429 D368        OUT PORTX3
00042B C33E04      JMP RUNUP7
00042E 210220      RUNUP5: LXI H,FLAGST        ;POINT HL TO FLAG STATUS ADDR.
000431 3600        MVI M,0                    ;RESET FLAG STATUS
000433 DB68        RUNUP6: IN PORTX3           ;READ FAST SPEED BIT
000435 47          MOV B,A                    ;PORT : PORTX3
000436 3E03        MVI A,3                    ;BIT : 3
000438 CD3705      CALL BITTST
00043B C8BF03      JZ RUNUP
00043E DB67        RUNUP7: IN PORTX2           ;READ LEVELLING BIT
000440 47          MOV B,A                    ;PORT : PORTX2
000441 3E03        MVI A,3                    ;BIT : 3
000443 CD3705      CALL BITTST
000446 C23E04      JNZ RUNUP7
000449 3EFF        MVI A,11111111B           ;RESET UP RUNNING
00044B D368        OUT PORTX3
00044D C9          RET

```

SUBPROGRAM : RUN DOWN

```

00044E      RUNDN:
00044E DB67      IN PORTX2                    ;READ BOTTOM FLOOR FLAG BIT
000450 47        MOV B,A                      ;PORT : PORTX2
000451 3E06      MVI A,6                      ;BIT : 6
000453 CD3705    CALL BITTST
000456 C26D04    JNZ RUNDN1
000459 DB68      IN PORTX3
00045B 47        MOV B,A                      ;RESET FAST SPEED BIT
00045C 3E03      MVI A,3                      ;PORT : PORTX3
00045E CD0F05    CALL BITSET                  ;BIT : 3
000461 47        MOV B,A                      ;SET SLOW SPEED BIT
000462 3E02      MVI A,2                      ;PORT : PORTX3
000464 CD2305    CALL BITCLR                  ;BIT : 2
000467 D368      OUT PORTX3
000469 3E00      MVI A,00000000B              ;SET COUNTER TO BOTTOM FLOOR
00046B D365      OUT COUNT                    ;AND RESET DU AND DD
00046D DB66      RUNDN1: IN PORTX1             ;READ SAFETY BIT
00046F 47        MOV B,A                      ;PORT : PORTX1
000470 3E02      MVI A,2                      ;BIT : 2
000472 CD3705    CALL BITTST
000475 CA7F04    JZ RUNDN2
000478 DB68      IN PORTX3                    ;RESET UP RUNNING
00047A F630      ORI 00110000B               ;RESET DOWN RUNNING
00047C C34E04    JMP RUNDN
00047F DB68      RUNDN2: IN PORTX3
000481 47        MOV B,A
000482 3E05      MVI A,5
000484 CD0F05    CALL BITSET                ;RESET UP RUNNING

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

000487 47	MOV B,A	
000488 3E04	MVI A,4	
00048A CD2305	CALL BITCLR	;SET DOWN RUNNING
00048D D368	OUT PORTX3	
00048F DB67	IN PORTX2	;READ DOWN FLAG BIT
000491 47	MOV B,A	;PORT : PORTX2
000492 3E04	MVI A,4	;BIT : 4
000494 CD3705	CALL BITTST	
000497 C2D604	JNZ RUNDN5	
00049A 210220	LXI H,FLAGST	;POINT HL TO FLAG STATUS BIT
00049D 7E	MOV A,M	;CHECK FLAG STATUS
00049E C2DB04	JNZ RUNDN6	
0004A1 DB65	IN COUNT	
0004A3 3D	DCR A	;DECREASE COUNTER
0004A4 D365	OUT COUNT	
0004A6 3601	MVI M,1	;SET FLAG STATUS BIT
0004A8 CDB700	CALL READLO	;SUBPROGRAM : READ LOWER
0004AB 210420	LXI H,LOCALL	;POINT HL TO LOWER CALL ADDR.
0004AE 7E	MOV A,M	;CHECK LOWER CALL
0004AF CAB804	JZ RUNDN3	
0004B2 CDAD01	CALL CHKSTP	;SUBPROGRAM : CHECK STOP
0004B5 C3C304	JMP RUNDN4	
0004B8 210020	RUNDN3: LXI H,HALLEN	;POINT HL TO HALL CALL ENABLE ADDR.
0004BB 3600	MVI M,0	;RESET HALL CALL ENABLE
0004BD DB65	IN COUNT	;RESET DIRECTION DOWN : DD
0004BF E63F	ANI 00111111B	;RESET DIRECTION UP : DU
0004C1 D365	OUT COUNT	
0004C3 DB68	RUNDN4: IN PORTX3	;SET SLOW SPEED BIT
0004C5 47	MOV B,A	;PORT : PORTX3
0004C6 3E02	MVI A,2	;BIT : 2
0004C8 CD2305	CALL BITCLR	
0004CB 47	MOV B,A	;RESET FAST SPEED BIT
0004CC 3E03	MVI A,3	;PORT : PORTX3
0004CE CD0F05	CALL BITSET	;BIT : 3
0004D1 D368	OUT PORTX3	
0004D3 C3E604	JMP RUNDN7	
0004D6 210220	RUNDN5: LXI H,FLAGST	;POINT HL TO FLAG STATUS ADDR.
0004D9 3600	MVI M,0	;RESET FLAG STATUS
0004DB DB68	RUNDN6: IN PORTX3	;READ FAST SPEED BIT
0004DD 47	MOV B,A	;PORT : PORTX3
0004DE 3E03	MVI A,3	;BIT : 3
0004E0 CD3705	CALL BITTST	
0004E3 CA4E04	JZ RUNDN	
0004E6 DB67	RUNDN7: IN PORTX2	;READ LEVELLING BIT
0004EB 47	MOV B,A	;PORT : PORTX2
0004E9 3E03	MVI A,3	;BIT : 3
0004EB CD3705	CALL BITTST	
0004EE C2E604	JNZ RUNDN7	
0004F1 3EFF	MVI A,11111111B	;RESET DOWN RUNNING
0004F3 D368	OUT PORTX3	

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

0004F5 C9

RET

SUBPROGRAM : DELAY

0004F6 3E14 DELAY: MVI A,20
0004F8 0664 DELAY1: MVI B,100
0004FA 0EB3 DELAY2: MVI C,131
0004FC 0D DELAY3: DCR C
0004FD C2FC04 JNZ DELAY3
000500 05 DCR B
000501 C2FA04 JNZ DELAY2
000504 3D DCR A
000505 C2FB04 JNZ DELAY1
00050B C9 RET

SUBPROGRAM : INPUT PORT

000509 INPUT:
0000050A = PORTIN: EQU INPUT+1
000509 DB00 IN 0
00050B C9 RET

SUBPROGRAM : OUTPUT PORT

00050C OUTPUT:
0000050D = PORTOT: EQU OUTPUT+1
00050C D300 OUT 0
00050E C9 RET

SUBPROGRAM : BIT SET

00050F BITSET:
00050F E607 ANI 00000111B
000511 4F MOV C,A
000512 7B MOV A,B
000513 211B05 LXI H,BITSE1
000516 0600 MVI B,0
000518 09 DAD B
000519 B6 DRA M
00051A C9 RET

00051B 01 BITSE1: DFB 00000001B
00051C 02 DFB 00000010B
00051D 04 DFB 00000100B
00051E 08 DFB 00001000B
00051F 10 DFB 00010000B

000520 20	DFB 00100000B
000521 40	DFB 01000000B
000522 80	DFB 10000000B

SUBPROGRAM : BIT CLEAR

```

000523      BITCLR:
000523 E607      ANI 00000111B
000525 4F        MOV C,A
000526 78        MOV A,B
000527 212F05    LXI H,BITCL1
00052A 0600      MVI B,0
00052C 09        DAD B
00052D A6        ANA M
00052E C9        RET
  
```

```

00052F FE      BITCL1: DFB 11111110B
000530 FD      DFB 11111101B
000531 FB      DFB 11111011B
000532 F7      DFB 11110111B
000533 EF      DFB 11101111B
000534 DF      DFB 11011111B
000535 BF      DFB 10111111B
000536 7F      DFB 01111111B
  
```

SUBPROGRAM : BIT TEST

```

000537      BITTST:
000537 E607      ANI 00000111B
000539 4F        MOV C,A
00053A 78        MOV A,B
00053B 214305    LXI H,BITTS1
00053E 0600      MVI B,0
000540 09        DAD B
000541 A6        ANA M
000542 C9        RET
  
```

```

000543 01      BITTS1: DFB 00000001B
000544 02      DFB 00000010B
000545 04      DFB 00000100B
000546 08      DFB 00001000B
000547 10      DFB 00010000B
000548 20      DFB 00100000B
000549 40      DFB 01000000B
00054A 80      DFB 10000000B
  
```

000000

END

00000050	BEGIN	0000052F	BITCL1	00000523	BITCLR
0000051B	BITSE1	0000050F	BITSET	00000543	BITTS1
00000537	BITTST	000000EB	CARCLL	000001F1	CHKST1
00000213	CHKST2	0000021B	CHKST3	00000220	CHKST4
000001AD	CHKSTP	00000065	CQUNT	000004F6	DELAY
000004F8	DELAY1	000004FA	DELAY2	000004FC	DELAY3
00000236	DIREC1	00000248	DIREC2	00000262	DIREC3
00000279	DIREC4	00000221	DIRECT	000000F0	DNHALL
00002002	FLAGST	00002000	HALLEN	00002003	HICALL
0000007E	INIT	00000098	INIT1	000000A3	INIT2
000000A7	INIT3	00000509	INPUT	00002004	LOCALL
0000050C	OUTPUT	0000050A	PORTIN	0000050D	PORTOT
00000066	PORTX1	00000067	PORTX2	00000068	PORTX3
000000C9	READL1	000000E5	READL2	00000107	READL3
00000122	READL4	0000012F	READL5	000000B7	READL0
00000144	READU1	00000160	READU2	00000182	READU3
0000019D	READU4	000001AC	READU5	00000130	READUP
0000044E	RUNDN	0000046D	RUNDN1	0000047F	RUNDN2
000004B8	RUNDN3	000004C3	RUNDN4	000004D6	RUNDN5
000004DB	RUNDN6	000004E6	RUNDN7	0000038F	RUNUP
000003BE	RUNUP1	000003D0	RUNUP2	00000410	RUNUP3
0000041B	RUNUP4	0000042E	RUNUP5	00000433	RUNUP6
0000043E	RUNUP7	00000285	STD001	0000029A	STD002
000002CB	STD003	000002E7	STD004	0000033C	STD005
0000038E	STD006	0000027A	STD00R	00002001	STOPEN
00000064	TOPD1P	00002005	TOPFLR	000000FB	UPHALL

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้ สำเร็จได้ด้วยความช่วยเหลือจาก อาจารย์ ชนิษฐา แซ่ตั้ง อาจารย์ประจำภาควิชาอิเล็กทรอนิกส์ ที่ได้ให้คำแนะนำที่เป็นประโยชน์ และเป็นอาจารย์ที่ปรึกษา กลุ่มข้าพเจ้าจึงขอกราบขอบพระคุณมา ณ.ที่นี้ นอกจากนี้ขอขอบพระคุณ คุณยุทธ ดวงทอง ผู้จัดการบริษัท ไทยลิฟท์อินดัสทรีส์ จำกัด และ ม.ล.วัชรพงศ์ เกษมสันต์ ที่ได้ให้ความช่วยเหลือ คัดคว้า ทดสอบการทำงาน และให้คำปรึกษาแนะนำในการทำ ปริญญานิพนธ์ครั้งนี้

สุดท้ายนี้กลุ่มข้าพเจ้า ขอขอบคุณผู้ที่ให้ความช่วยเหลืออื่น ๆ ซึ่งไม่ได้เอ่ยนาม มา ณ.ที่นี้ด้วย



เอกสารอ้างอิง

John Uffenbeck

Microcomputer and Microprocessors

America : Prentice-Hall international , 1985

William Kleitz

Digital and Microprocessor Funda-

mental New Jersey : Prentice-Hall international , 1990

Lance A. Leventhal

8085A/8085 Assemble Language Pro-

gramming California : Adam Osborne Associates incorporated , 1978

ซีเอ็ดยุคเคชั่น

คู่มือ ไอซี CMOS 4000 Series กรุงเทพฯ :

บริษัท ซีเอ็ดยุคเคชั่น จำกัด , 1988

ซีเอ็ดยุคเคชั่น

คู่มือ/เทียบเบอร์ ไอซี TTL กรุงเทพฯ :

บริษัท ซีเอ็ดยุคเคชั่น จำกัด , 1989