



การพัฒนาหุ่นยนต์เคลื่อนที่สองขา

DEVELOPMENT OF 3D BIPED ROBOT



โดย

นายภูสิทธิ์ รัตนปิยะสุนทร

นายอาทิตย์ ชีวะเกตุ

วัน เดือน ปี..... ๓๐ ก.ค. ๒๕๕๐
เลขทะเบียน..... ๐๖๔๙๕๖
เลขเรียกหนังสือ..... T 38065 ๓๕๗๗ ก

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมระบบควบคุม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2538

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ปริญญานิพนธ์ปีการศึกษา 2538

ภาควิชาวิศวกรรมระบบควบคุม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง

การพัฒนาหุ่นยนต์เคลื่อนที่สองขา

DEVELOPMENT OF 3D BIPED ROBOT

ผู้จัดทำ

นายภูสิทธิ์ รัตนปิยะสุนทร

นายอาทิตย์ ชีวะเกตุ


อาจารย์ที่ปรึกษา
(อาจารย์สุมิตร พนาอุดมทรัพย์)



การพัฒนาหุ่นยนต์เคลื่อนที่สองขา
DEVELOPMENT OF 3D BIPED ROBOT

โดย

นายภูสิทธิ์ รัตนปิยะสุนทร

รหัสประจำตัวนักศึกษา 35104322

นายอาทิตย์ ชีวะเกตุ

รหัสประจำตัวนักศึกษา 35104551

อาจารย์ที่ปรึกษา

อาจารย์สมิตร พนาอุดมทรัพย์

บทคัดย่อ

หุ่นยนต์ที่พัฒนาขึ้น มีส่วนประกอบ 3 ส่วนหลักคือ ส่วนยูสเซอร์อินเตอร์เฟซ บนไมโครคอมพิวเตอร์ ส่วนควบคุมที่เป็นไมโครคอนโทรลเลอร์ และกลไกขาหุ่นยนต์ โดยได้มีการพัฒนาระบบควบคุมที่ไม่ซับซ้อน ทำให้ได้ระบบหุ่นยนต์ที่มีการควบคุมแบบแมนนวล Open Loop รับคำสั่งจากผู้ใช้งานไมโครคอมพิวเตอร์ ที่สามารถ จัดท่าทางการเคลื่อนไหวได้ตาม ที่ผู้ใช้กำหนด

สำหรับโครงการนี้ ยังต้องได้รับการพัฒนาต่อไป เพื่อให้ได้ หุ่นยนต์ที่มี เสถียรภาพในการทรงตัวและความสามารถในการประมวลผลข้อมูลรวมทั้งพารามิเตอร์ที่จำเป็นในระบบควบคุมที่สมบูรณ์

ABSTRACT

This 3D biped prototype robot has 3 main subsystems, which are Windows-based Graphical-User-Interface on microcomputer, microcontroller-based control system and robot mechanics.

By developing a simple control system, we brought about a robot which has an open-loop manual control system, receiving commands from microcomputer which is able to order robot to pose or animate in any manner as user desire.

This project needs further development to obtain more sophisticated and stabilized robot control system for the robot to be balanced and able to process more complex parameters.

เอกสารนี้เป็นเอกสารลิขสิทธิ์ในสถาบันวิจัยเพื่อการพัฒนาประเทศไทย เมื่อผู้ใดที่เห็นประโยชน์ในการนำเอกสารนี้ไปใช้โดยไม่ผ่านการคัดค้านจากเจ้าของลิขสิทธิ์ หรือมีการแก้ไขเนื้อหาโดยไม่ได้รับอนุญาตจากเจ้าของลิขสิทธิ์ ถือว่าผิดกฎหมาย และต้องแจ้งเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลงได้ด้วยดีก็เพราะได้รับความเมตตาจากอาจารย์สุมิตร พนา
อุดมทรัพย์ อาจารย์ที่ปรึกษาที่ได้ให้ความกรุณาแนะนำแก่ผู้จัดทำตลอดมา ผู้จัดทำขอกราบ
ขอบพระคุณเป็นอย่างสูงมา ณ โอกาสนี้ด้วย และยังขอขอบคุณคณาจารย์ต่าง ๆ ที่ได้ประสิทธิ
ประสาทวิชาความรู้แก่ผู้จัดทำ

และต้องขอกราบขอบพระคุณคุณพ่อคุณแม่ของผู้จัดทำที่ได้อุปการะผู้จัดทำและยังเป็นผู้
ให้กำลังใจแก่ผู้จัดทำตลอดมา

ขอขอบคุณเพื่อน ๆ ที่ให้ความเป็นเพื่อนและให้กำลังใจแก่ผู้จัดทำ

ผู้จัดทำ

นายภูสิทธิ์

นายอาทิตย์

รัตนปิยะสุนทร

ชีวะเกตุ

สารบัญเรื่อง

เรื่อง	หน้า
บทที่ 1 บทนำ	
1.1 ขอบเขตของโครงการ	1-2
1.2 จุดประสงค์โครงการ	2-2
บทที่ 2 ทฤษฎีและหลักการเบื้องต้น	
2.1 ไมโครคอนโทรลเลอร์	3-8
2.2 การสื่อสารข้อมูลผ่านพอร์ตอนุกรม	8-9
2.3 ซีซีเซิร์ฟเวอร์	9-10
2.4 การควบคุมหุ่นยนต์	10-12
บทที่ 3 การออกแบบและการทดลอง	
3.1 การออกแบบส่วนไมโครคอนโทรลเลอร์	13-24
3.2 การออกแบบทางด้าน Mechanic ของส่วนขาหุ่นยนต์	25-26
3.3 การออกแบบส่วนคอมพิวเตอร์เพื่อติดต่อกับผู้ใช้	26-31
บทที่ 4 ผลการทดลอง	
4.1 การทดลองของไมโครคอนโทรลเลอร์	32-38
4.2 การทดลองทางด้านเครื่องคอมพิวเตอร์	39-42
บทที่ 5 สรุปและวิเคราะห์ผลการทดลอง	
5.1 สรุปผลการดำเนินงาน	43-43
5.2 ปัญหาที่เกิดขึ้นและแนวทางการแก้ไข	43-44
5.3 แนวทางการพัฒนาในอนาคต	44-44

สารบัญรูป

รูป	หน้า
1.1 ภาพแสดงบล็อกไดอะแกรมของระบบ	2
2.1 แสดงขาของ Z86E08	4
2.2 การจัดแบ่งรีจิสเตอร์ไฟล์	5
2.3 แสดงการจัดหน่วยความจำ	6
2.4 แสดงอินเตอร์รัพท์เวกเตอร์	7
2.5 แสดงบล็อกไดอะแกรมของแกนเทอร์/ไทม์เมอร์	8
2.6 คอนเน็คเตอร์ที่ใช้ในการเชื่อมต่อแบบ RS-232 (DB-25)	9
2.7 บล็อกไดอะแกรมของระบบดีซีเซอร์ไวโมเตอร์	10
2.8 แสดงหุ่นที่ควบคุมสองมิติ	10
3.1 วงจรแปลง RS-232 เป็น TTL	14
3.2 การส่งข้อมูลจากขาสัญญาณข้อมูลอนุกรมเข้า	15
3.3 สัญญาณพัลส์ที่ใช้ในการควบคุมมอเตอร์	16
3.4 ไฟล์ไดอะแกรมของโปรแกรม INTERRUPT.ASM	18
3.5 ไฟล์ไดอะแกรมของโปรแกรม DRIVER.ASM	19
3.6 ไฟล์ไดอะแกรมของโปรแกรม POSITION.ASM	20
3.7 ไฟล์ไดอะแกรมของโปรแกรม DRIVE.ASM	21
3.8 ไฟล์ไดอะแกรมของโปรแกรม SORT.ASM	22
3.9 ไฟล์ไดอะแกรมของโปรแกรม NEWDRIVE.ASM	23
3.10 ไฟล์ไดอะแกรมของโปรแกรม PREPARE.ASM	24
3.11 แสดงโครงสร้างหุ่นยนต์ที่ตั้งเป้าหมายไว้	25
3.12 แสดงโปรแกรมจัดลำดับท่าทางที่เสร็จสมบูรณ์	28
3.13 คำพารามิเตอร์ของพอร์ตสื่อสารที่ตั้งผ่าน Comm Object	30
4.1 แสดงลำดับสัญญาณจากพอร์ตอนุกรม 1	32
4.2 แสดงรูปสัญญาณจากชุดบังคับวิทยุที่เข้าสู่ดีซีเซอร์ไวโมเตอร์	33
4.3 แสดงสัญญาณควบคุมที่ได้จากโปรแกรม POSITION.ASM	34
4.4 แสดงความผิดพลาดที่เกิดขึ้นในโปรแกรม DRIVE.ASM	35

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5 แสดงสัญญาณควบคุมที่ได้จากโปรแกรม NEWDRIVE.ASM 36

 เมื่อยังไม่มีกรหน่วงสัญญาณ ‘0’

4.6 สัญญาณจาก NEWDRIVE.ASM เมื่อมีการหน่วงเวลาสัญญาณศูนย์โวลต์ 37

4.7 แสดงสัญญาณเมื่อค่าควบคุมเรียงลำดับเป็นกรณีที่ดีที่สุด 38

4.8 แสดงสัญญาณเมื่อค่าควบคุมเรียงลำดับเป็นกรณีที่แย่ที่สุด 38

4.9 Diagram ของ Data ขนาด 2 byte 39

4.10 แสดงการวางคอมพิวเตอรเพื่อทดสอบ 40

4.11 แสดง Output ที่ได้จากการทดสอบ 41

4.12 การ Send Animation 42



บทที่ 1

บทนำ

หุ่นยนต์ได้ถูกนำมาประยุกต์ใช้งานอย่างมากในทางอุตสาหกรรม ซึ่งโดยมากจะอยู่ในรูปของแขนกล หรือหุ่นที่เคลื่อนที่โดยอาศัยล้อเลื่อน การพัฒนาหุ่นยนต์ที่ใช้ในการเคลื่อนที่ เป็นอีกแนวการศึกษาหนึ่งที่เริ่มจะมีการศึกษามากขึ้น เนื่องจากมีจุดเด่นอยู่ที่ความสามารถในการเคลื่อนที่ ผ่านภูมิประเทศ และพื้นผิวได้หลายรูปแบบ ทั้งในแนวราบ ขึ้นบันได หรือพื้นเอียง

การพัฒนาหุ่นยนต์เคลื่อนที่สองขา จะมีการศึกษาในสามหัวข้อหลักๆ ได้แก่ [1] การเคลื่อนที่ของหุ่นยนต์ (Biomechanics) ซึ่งจะมีการศึกษาท่าทางการเดินของมนุษย์ โดยอาจได้จากการสังเกตและรวบรวมลักษณะท่าทางของขามนุษย์ซึ่งเป็นแม่แบบการเดินของหุ่น [2] หุ่นยนต์และการควบคุมหุ่นยนต์ (Robotics and Controls) มีการศึกษาด้านหุ่นยนต์และการควบคุมอุปกรณ์ที่ใช้ส่งถ่ายพลังงานได้แก่ ลักษณะทางกลไก และอุปกรณ์ส่งถ่ายพลังงาน และ [3] การสั่งการหุ่นยนต์ผ่านเครื่องมือต่างๆ เช่น คอมพิวเตอร์ หรือไมโครคอนโทรลเลอร์

1.1 ขอบเขตของโครงการ

โครงการหุ่นยนต์เคลื่อนที่สองขา ที่ทำขึ้นสามารถแบ่งออกเป็นระบบย่อยๆ คือ

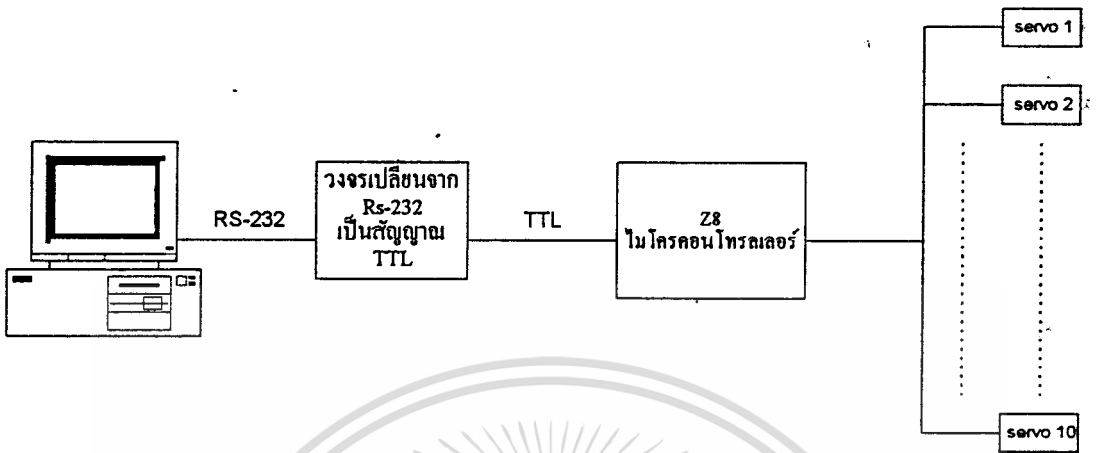
(1) ระบบคอมพิวเตอร์ที่จะเป็นระบบกำหนดการเคลื่อนไหวของขาของหุ่นยนต์และการติดต่อกับผู้ใช้ (User Interface)

(2) ส่วนไมโครคอนโทรลเลอร์ ซึ่งจะติดตั้งอยู่ที่ขาหุ่นยนต์มีหน้าที่รับข้อมูลจากคอมพิวเตอร์ แล้วส่งสัญญาณไปควบคุมดีซีเซอร์โวมอเตอร์ (DC Servo Motor) ตามข้อต่อของขาหุ่นยนต์ เพื่อควบคุมการเคลื่อนไหวให้เป็นไปตามคำสั่งที่ส่งมาจากคอมพิวเตอร์

(3) ระบบกลไกขาหุ่นยนต์

(4) ส่วนประกอบเพิ่มเติม ได้แก่ วงจรรับสัญญาณจากมาตรฐาน RS-232 จากคอมพิวเตอร์ แล้วแปลงให้เป็นสัญญาณ TTL เข้าสู่ไมโครคอนโทรลเลอร์ เพื่อใช้ในการควบคุมมอเตอร์ต่อไป

รายละเอียดทั้งหมดประกอบขึ้นเป็นระบบดังรูป



รูปที่ 1.1 ภาพแสดงบล็อกไดอะแกรมของระบบ

1.2 จุดประสงค์ของโครงการ

- (1) ศึกษาการเคลื่อนที่ของหุ่นยนต์ในลักษณะสองขา
- (2) ศึกษาการเชื่อมต่อและสั่งงานจากไมโครคอมพิวเตอร์ไปยังไมโครคอนโทรลเลอร์ เพื่อ

ควบคุมมอเตอร์

บทที่ 2

ทฤษฎีและหลักการเบื้องต้น

การสร้างหุ่นยนต์เคลื่อนที่สองขาขึ้น ได้มีการศึกษาในหัวข้อหลักดังนี้

1. ไมโครคอนโทรลเลอร์ Z8
2. การสื่อสารข้อมูลผ่านพอร์ตอนุกรม RS-232 ระหว่างไมโครคอมพิวเตอร์และไมโครคอนโทรลเลอร์
3. ดีซีเซอร์โวมอเตอร์
4. หุ่นยนต์

2.1 ไมโครคอนโทรลเลอร์

ไมโครคอนโทรลเลอร์ตระกูล Z8 ของ ซิล็อก (Zilog) เบอร์ Z86E08 OTR มีคุณสมบัติเหมาะสมที่จะใช้ในโครงงานนี้ เนื่องจากขนาดของ package เล็ก (18 pin) มีขาสัญญาณให้ใช้งานอย่างพอเพียง และมีเครื่องมือพัฒนาที่อำนวยความสะดวก คือชุด Z8 CCP Emulator

2.1.1 หน้าที่ของไมโครคอนโทรลเลอร์

ไมโครคอนโทรลเลอร์ มีหน้าที่หลักคือ

1. รับและส่งข้อมูลกับคอมพิวเตอร์แบบอนุกรม ในการรับข้อมูล จะรับจากคอมพิวเตอร์ไปบรรจุในบัฟเฟอร์ (Buffer) ที่เตรียมไว้สำหรับเก็บค่าเพื่อควบคุมตำแหน่ง ของมอเตอร์แต่ละตัว ส่วนการส่งข้อมูลไปยังคอมพิวเตอร์ จะเป็นข้อมูลตำแหน่งของระบบ เพื่อใช้ในการควบคุมการทรงตัว

2. หน้าที่ในการนำค่าตำแหน่งจากบัฟเฟอร์ส่งไปควบคุมมอเตอร์แต่ละตัว

2.1.2 ไมโครคอนโทรลเลอร์ Z86E08

2.1.2.1 คุณสมบัติโดยทั่วไปที่ได้นำมาใช้ประโยชน์

- (1) เป็นไมโครคอนโทรลเลอร์ 8 บิต (Bit) ชนิด CMOS
- (2) ลักษณะแพ็คเกจเป็นแบบ DIP มี 18 ขา
- (3) ขนาดแรงดันในการทำงาน (Operaitng Voltage) อยู่ในช่วง 4.0-9.5 โวลต์
- (4) กินไฟต่ำ ประมาณ 50 มิลลิวัตต์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

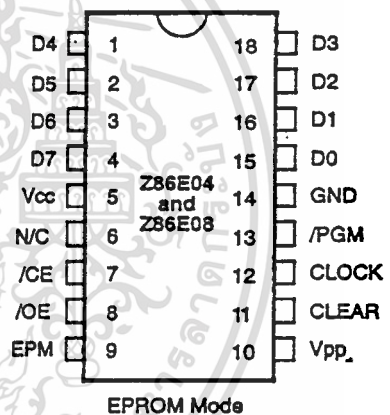
- (5) ขาสัญญาณอินพุต/เอาต์พุต 14 ขา
- (6) PROM ภายในขนาด 2 กิโลไบต์
- (7) RAM ภายในขนาด 128 ไบต์
- (8) เกานต์เตอร์/ไทม์เมอร์ (Counter/Timer) ที่สามารถโปรแกรมได้ขนาด 8 บิต 2 ตัวซึ่งมีพรีสเกลเลอร์ ขนาด 6 บิต
- (9) อินเตอร์รัพท์เวกเตอร์ (Interrupt Vector) 6 ตำแหน่ง จากแหล่งกำเนิดอินเตอร์รัพท์ (Interrupt Source) มีลำดับความสำคัญ (Priority) ที่กำหนดได้จากรีจิสเตอร์

2.1.2.2 รายละเอียดของขาต่างๆ

PIN DESCRIPTION

Table 1. EPROM Mode Pin Identification

Z86E04/ 08	EPROM Mode	Symbol	Function	Direction
Pin #				
1-4	D7-D4		Data 4, 5, 6, 7	In/Output
5	V _{cc}		Power Supply	
6	N/C		No Connection	
7	/CE		Chip Enable	Input
8	/OE		Output Enable	Input
9	EPM		EPROM Prog Mode	Input
10	V _{pp}		Prog Voltage	Input
11	Clear		Clear Clock	Input
12	Clock		Address	Input
13	/PGM		Prog Mode	Input
14	GND		Ground	
15-18	D3-D0		Data 0, 1, 2, 3	In/Output



รูปที่ 2.1 แสดงขาของ Z86E08

2.1.2.3 เวลาของคำสั่ง (Instruction Timing)

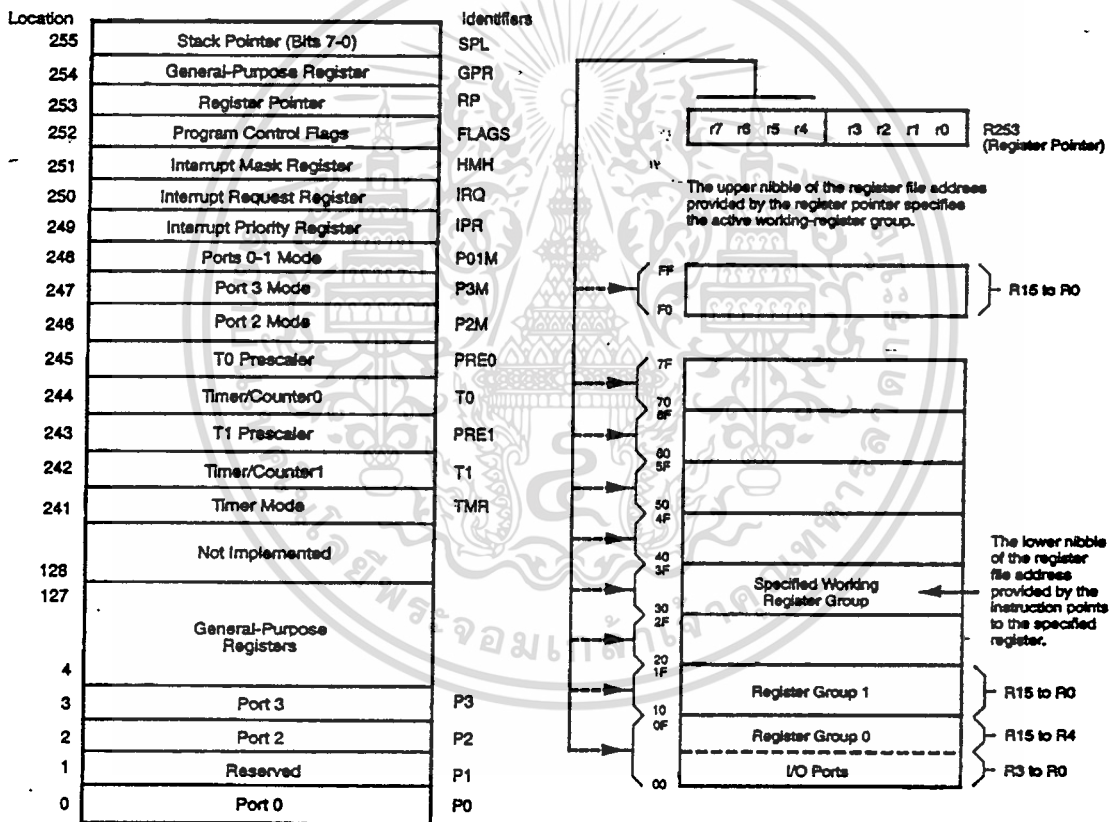
สถาปัตยกรรมของโปรเซสเซอร์ Z8 ใช้การทำไปป์ไลน์ (Instruction Pipe lining) ซึ่งจะทำให้ ไซเคิลเฟตช์ (fetch cycle) และ ไซเคิลปฏิบัติคำสั่ง (execute cycle) มีการซ้อนทับกัน (overlap) กัน แต่อย่างไรก็ตาม การคำนวณหาเวลาของคำสั่งแต่ละคำสั่ง จะทำโดยดูค่าไซเคิล (cycle) ที่ใช้ในตารางข้อมูลการใช้เวลาของคำสั่ง โดยที่ 1 ไซเคิลเท่ากับ ค่าความถี่ของสัญญาณนาฬิกาภายนอกหารสอง

2.1.2.4 การใช้งานหน่วยความจำ การใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แบ่งหน่วยความจำได้เป็น 2 ส่วน คือ

(1) รีจิสเตอร์ไฟล์ (Register File)

Z86E08 มีหน่วยความจำที่เป็นรีจิสเตอร์ทั้งหมด 256 ไบต์ ซึ่งถูกใช้งานเพียง 144 ไบต์ (ส่วนที่ไม่ได้ใช้งานจะสำรองไว้เพื่อใช้งานในอนาคต) ในพื้นที่รีจิสเตอร์ 144 ไบต์ นี้ จะเก็บรีจิสเตอร์ทั้งหมดของ Z86E08 ไม่ว่าจะเป็น อินพุต/เอาต์พุตพอร์ต (I/O Port), รีจิสเตอร์เพื่อการใช้งานทั่วไป (General Purpose Register), รีจิสเตอร์ประกอบ (Peripheral Register) หรือ รีจิสเตอร์ที่ใช้ในการควบคุม (Control Register) มีการจัดแบ่งตามรูป

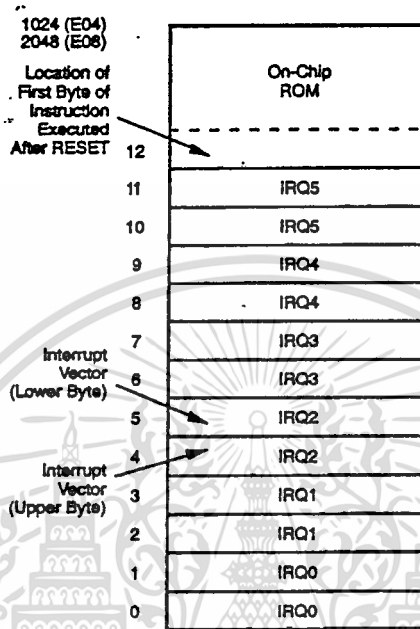


รูปที่ 2.2 การจัดแบ่งรีจิสเตอร์ไฟล์

(2) หน่วยความจำเก็บโปรแกรม (Program Memory)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ใน Z86E08 จะมี หน่วยความจำในตัวอยู่ 2 กิโลไบต์ มีการจัดหน่วยความจำดังรูปที่ ตามรูปที่ 2.3 ซึ่งจะเก็บตารางอินเตอร์รัพท์เวกเตอร์ ในการเขียนโปรแกรมการอินเตอร์รัพท์ ก็ต้องเข้ามาแก้ไขตารางอินเตอร์รัพท์เวกเตอร์ที่นี้



รูปที่ 2.3 แสดงการจัดหน่วยความจำ

2.1.2.5 อินพุต/เอาต์พุตพอร์ต (I/O Port)

Z86E08 มีพอร์ตให้ใช้งาน 3 พอร์ต แต่เนื่องจากข้อจำกัดทางฮาร์ดแวร์ จึงไม่สามารถใช้ทุกพอร์ต ได้อย่างอิสระ โดยจะมีขา 14 ขาที่เป็นพอร์ต โดยเป็น P2.0-P2.7 P3.0-P3.2 P0.0-P0.2 แต่ละบิตของทุกพอร์ตจะมีอินพุตรีจิสเตอร์ และเอาต์พุตรีจิสเตอร์แยกกันและสามารถกำหนดให้บิตเป็นอินพุตหรือเอาต์พุตได้ โดยผ่านรีจิสเตอร์ควบคุมพอร์ต P01M , P2M , P3M

Port 0 มีให้ใช้ตั้งแต่ P0.0-P0.2 รวม 3 บิต แบบสองทิศทาง (Bi-directional)

Port 2 P2.0-P2.7 เหมือน Port 0

Port 3 P3.0-P3.2 มีลักษณะพิเศษ คือ เป็น พอร์ตอินพุตได้อย่างเดียว ใช้เป็นส่วนรับอินเตอร์รัพท์จากภายนอก ซึ่งกำหนดได้ว่าจะอินเตอร์รัพท์ที่ขอบขาขึ้น (Rising Edge) หรือขอบขาลง (Falling Edge)

2.1.2.6 อินเตอร์รัพท์ (Interrupt)

แบ่งออกเป็นอินเทอร์รัพท์จากภายนอก (External Interrupt Source) ซึ่งรับเข้ามาทาง P3.0 P3.1 และ P3.2 และ อินเทอร์รัพท์จากภายใน (Internal Interrupt Source) ซึ่งเกิดจาก Timer ซึ่งจะมีเวกเตอร์ที่จัดสรรดังนี้

Name	Source	Vector Location	Comments
IRQ ₀	$\overline{\text{DAV}}_0$, IRQ ₀	0,1	External (P3 ₂), $\downarrow$ Edge Triggered
IRQ ₁	$\overline{\text{DAV}}_1$, IRQ ₁	2,3	External (P3 ₃), $\downarrow$ Edge Triggered
IRQ ₂	$\overline{\text{DAV}}_2$, IRQ ₂ , T _{IN}	4,5	External (P3 ₁), $\downarrow$ Edge Triggered
IRQ ₃	IRQ ₃	6,7	External (P3 ₀), $\downarrow$ Edge Triggered
	Serial In	6,7	Internal
IRQ ₄	T ₀	8,9	Internal
	Serial Out	8,9	Internal
IRQ ₅	T ₁	10,11	Internal

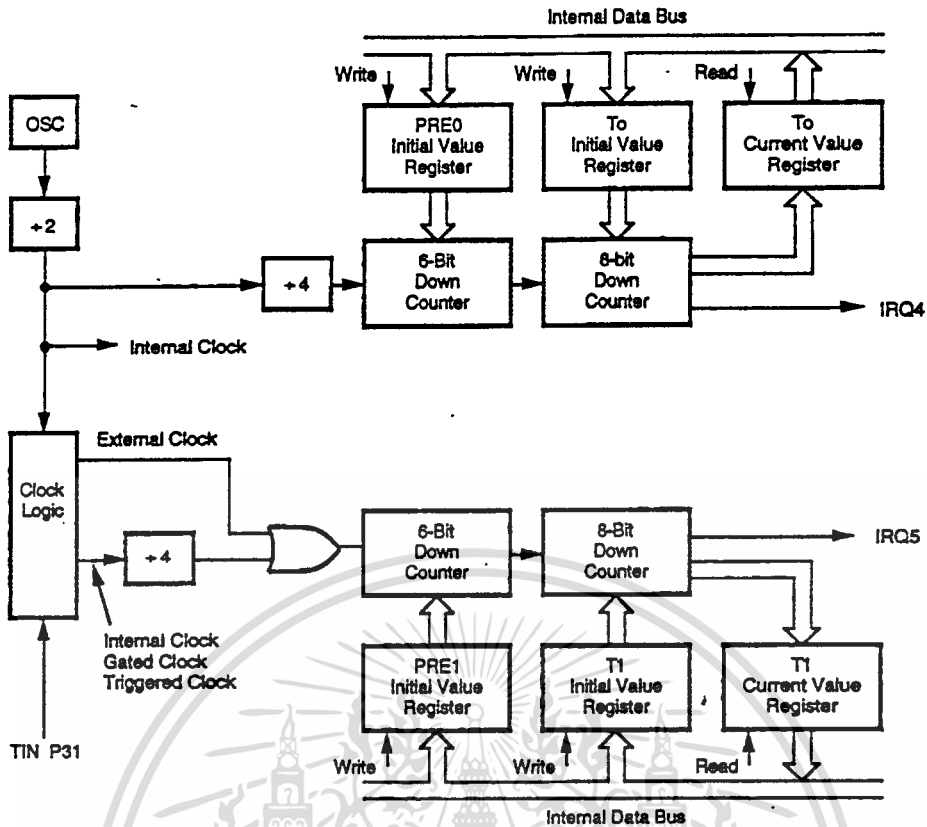
รูปที่ 2.4 แสดงอินเทอร์รัพท์เวกเตอร์

2.1.2.7 เคาท์เตอร์/ไทม์เมอร์ (Counter / Timer)

Z8 มี เคาท์เตอร์/ไทม์เมอร์ ขนาด 8 บิตอยู่ 2 ตัว แต่ละตัวมี 프리สเกลเลอร์ ขนาด 6 บิต เคาท์เตอร์/ไทม์เมอร์ ทั้งสองตัวนี้จะทำงานเป็นอิสระกับการทำงานของ Z8 จึงเหมาะสำหรับงานที่ต้องการความเที่ยงตรงทางด้านเวลาสูง

การทำงานของ เคาท์เตอร์/ไทม์เมอร์ สามารถทำงานได้ทั้งโหมดครั้งเดียวผ่าน(Single Pass) และโหมดต่อเนื่อง (Continuous mode) และค่าใน เคาท์เตอร์ หรือ 프리สเกลเลอร์ สามารถถูกเปลี่ยนแปลงได้ในขณะที่คาท์เตอร์/ไทม์เมอร์ กำลังทำงานอยู่

สัญญาณนาฬิกาที่มาขับ เคาท์เตอร์/ไทม์เมอร์ จะนำมาจากสัญญาณนาฬิกาภายในหารด้วย 4 ซึ่งจะไปยัง 프리สเกลเลอร์ ขนาด 6 บิต หรืออาจเป็นสัญญาณจากภายนอกมาขับ เคาท์เตอร์ตัวที่ 1 ทางพอร์ต 3 ดังแสดงในบล็อกไดอะแกรม



รูปที่ 2.5 แสดงบล็อกโคะแกรมของ เคาท์เตอร์/ไทม์เมอร์

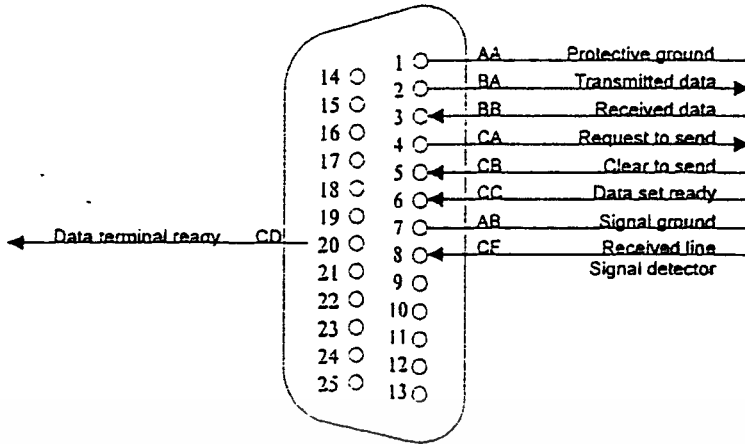
สำหรับไทม์เมอร์/เคาท์เตอร์ นี้เดิมทีจะใช้เป็นตัวสร้างบอดเรตแต่ได้เปลี่ยนไปใช้การวนลูปของโปรแกรม(ดังจะแสดงในส่วนของการออกแบบ)

2.2 การสื่อสารข้อมูลผ่านพอร์ตอนุกรม (RS-232)

เนื่องจาก Z86E08 เป็นโปรเซสเซอร์ ที่ถูกตัดความสามารถทางอินพุต/เอาต์พุตแบบอนุกรม ไป ทางผู้จัดทำโครงการจึงใช้การประยุกต์ใช้งาน อินพุต/เอาต์พุตพอร์ต มาพัฒนาโปรแกรมเพื่อรับ-ส่งแบบอนุกรมขึ้นเอง ดังจะได้กล่าวถึงต่อไปในเรื่องการออกแบบ

มาตรฐาน RS-232 เป็นวิธีการเชื่อมต่อคอมพิวเตอร์กับอุปกรณ์ต่างๆที่นิยมอย่างแพร่หลาย เป็นมาตรฐานที่รับรองโดย EIA:Electronic Industries Association ซึ่งกำหนดให้มี Data Terminal Equipment :DTE และ Data Communication Equipment : DCE

มาตรฐาน RS-232 กำหนดให้ใช้คอนเนคเตอร์ 25 พินที่เรียกกันว่า DB-25 Connector ซึ่งมีขาสัญญาณต่างๆตามรูปที่ 2.6



รูปที่ 2.6 แสดงคอนเน็คเตอร์ที่ใช้ในการเชื่อมต่อแบบ RS-232 (DB-25)

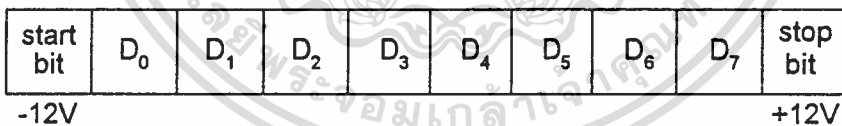
เมื่อทำการส่งข้อมูลแบบ 8 บิต , ไม่มีพาริตี (parity) , บิตสต็อบ 1 บิต ได้ลักษณะการส่งอนุกรม แบบ RS-232 คือ

- บิตสตาร์ท (start bit) คือ + 12 V
- บิตสต็อบ (stop bit) คือ -12 V
- ข้อมูลที่เป็น 8 บิต มีสถานะดังนี้

“1” คือ -12 โวลต์

“0” คือ +12 โวลต์

ลำดับการส่งข้อมูล 1 ไบต์ เป็นดังนี้



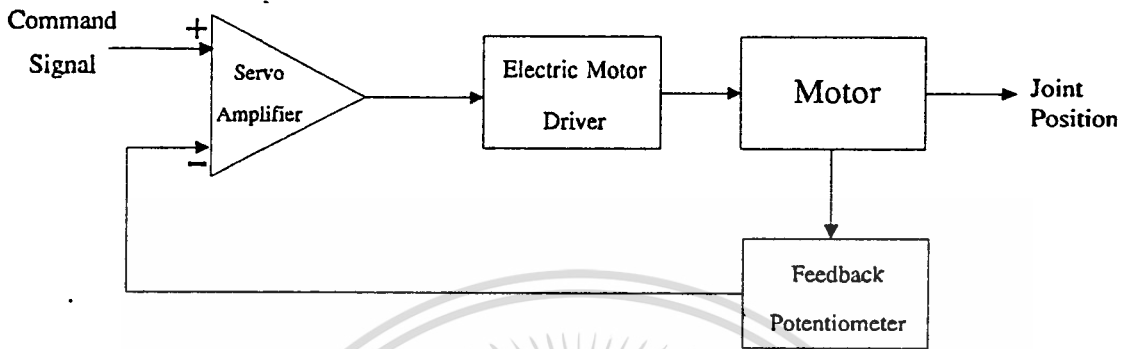
2.3 คีชีเซอร์โวมอเตอร์

คีชีเซอร์โวมอเตอร์ จะถูกมาใช้ในการหมุนข้อต่อต่างๆ โดยอาศัยการควบคุมตำแหน่งให้มอเตอร์แต่ละตัวหมุนไปยังตำแหน่งที่ต้องการ ซึ่งในการพัฒนาหุ่นยนต์เคลื่อนที่สองขา ได้ออกแบบให้มีข้อหมุนทั้งหมด 10 ข้อเพื่อให้ได้ลักษณะการเคลื่อนที่ที่ครบทุกอิริยาบถของขามนุษย์ ซึ่งจะต้องใช้คีชีเซอร์โวมอเตอร์ 10 ตัวเช่นกัน

การควบคุมตำแหน่ง จะทำได้โดยการเปรียบเทียบสัญญาณควบคุม กับสัญญาณป้อนกลับ (feedback) มาจากเอาต์พุต ความต่างของสัญญาณจะถูกแปลงเป็นแรงดันทางอนาล็อก (Analog Voltage) และส่งไปยัง ส่วนขยายของเซอร์โว (Analog servo amplifier) ที่เป็นตัวควบคุมการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทำงานของมอเตอร์ ซึ่งจะทำให้มอเตอร์หมุนไปในทิศทางที่ทำให้สัญญาณความผิดพลาด (Error Signal) เป็นศูนย์ก็จะหยุดหมุน สามารถอธิบายดังบล็อกไดอะแกรมง่ายๆดังรูป

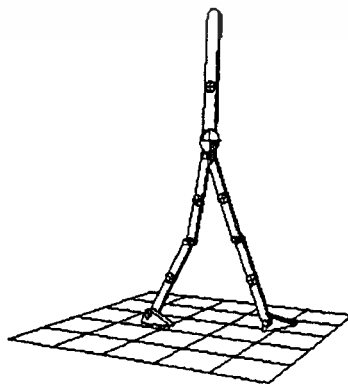


รูปที่ 2.7 แสดงบล็อกไดอะแกรมของระบบคิซีเซอร์โวมอเตอร์

จากการทดสอบมอเตอร์หลายชนิด ดังจะกล่าวในส่วนของการทดลองและออกแบบ ได้ตกลงเลือกใช้มอเตอร์วิทยุบังคับ ที่มาพร้อมกับตัวควบคุม (Controller) ในตัว ที่รับสัญญาณควบคุมแบบดิจิตอล 5 โวลต์โดยอาศัยความกว้างของสัญญาณซีกบวก ทำให้ไม่ต้องศึกษามอเตอร์และการควบคุมในระดับลึก

2.4 การควบคุมหุ่นยนต์

หุ่นยนต์เคลื่อนที่สองขานั้น ได้มีการแบ่งการควบคุมเป็นสองประเภทคือ การควบคุมสองมิติ (Planar) และการควบคุมสามมิติ (3D) ซึ่งการควบคุมสองมิติ จะควบคุมให้หุ่นมีให้ล้มไปข้าง



รูปที่ 2.8 แสดงหุ่นที่ควบคุมสองมิติ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน้าหรือข้างหลัง และให้ยืนตัวเองในแนวตั้งมิให้ล้มพับลงมา ส่วนการควบคุมสามมิติจะเพิ่มการควบคุมมิให้หุ่นยนต์ล้มในแนวด้านข้างด้วย

จำนวนส่วนประกอบ (links) ที่ประกอบขึ้นเป็นขาหุ่นยนต์ จะเป็นตัวกำหนด Degree Of Freedom (DOF) ยังมีจำนวนข้อต่อและส่วนประกอบมากขึ้น ก็จะทำให้มี Degree Of Freedom มากขึ้น ทำให้สามารถเคลื่อนไหวได้หลายท่าทางมากขึ้น และทำให้ต้องมีการควบคุมเสถียรภาพ มารองรับมากขึ้นเช่นกัน

เราสามารถแบ่งประเภทของการควบคุมเสถียรภาพออกได้เป็นสองวิธี ได้แก่ การควบคุมแบบแมนนวลจากผู้ควบคุม (Manual interactive user feedback control) และการควบคุมแบบอัตโนมัติ (Automatic Control) ซึ่งอธิบายได้ดังนี้

2.4.1 Manual Control

ผู้ควบคุมจะเป็นผู้กำหนดการเคลื่อนที่ของส่วนประกอบต่างๆโดยตรง (low level) โดยจะต้องสั่งการผ่านอุปกรณ์รับค่า การป้อนกลับก็จะหมายถึงการมองและการตัดสินใจของผู้ควบคุม คล้ายการเล่นวีดีโอเกมส์ ซึ่งต้องอาศัยความรวดเร็วในการรับรู้และตัดสินใจเพื่อไม่ให้เกิดสถานะไม่มีเสถียรภาพขึ้น ซึ่งจะตัดสินใจจากการรับรู้จุดศูนย์กลางแรงโน้มถ่วง (Center of Gravity), จุดโมเมนต์เป็นศูนย์ (Zero Moment Point)

ลำดับการควบคุมเหล่านี้ สามารถเรียบเรียงเป็นท่าทางในการเคลื่อนที่ และรวบรวมเป็นข้อมูลของท่าทางต่างๆ ซึ่งสามารถนำมาปรับแต่งเพื่อให้ได้ท่าทางที่เหมาะสมและมีเสถียรภาพมากขึ้น และสามารถนำมาวนซ้ำเพื่อให้ได้ลักษณะการเดินที่ต้องการได้

การควบคุมแบบแมนนวลนี้ ทำให้สามารถสร้างท่าทางที่ต้องการที่เป็นไปได้ทุกอย่างก็ได้ ทั้งเดิน, วิ่ง, หรือท่าทางที่สามารถแสดงเป็นลำดับได้ เช่นการกระโดดหรือดีดก้า

2.4.2 Automatic Control

การใช้การควบคุมอัตโนมัติก็เพื่อใช้ในการตัดสินใจว่าจะให้ส่วนประกอบต่างๆอยู่ในตำแหน่งใด ทำให้ผู้ควบคุมไม่ต้องตัดสินใจในระดับล่างในทุกๆช่วงเวลา โดยปล่อยให้ทำหน้าที่ของระบบควบคุมอัตโนมัติ ผู้ควบคุมจะทำเพียงกำหนดพารามิเตอร์ในระดับบน เช่น ความเร็ว, ทิศทาง หรือช่วงระยะเวลาก้าว เป็นต้น การควบคุมแบบอัตโนมัตินี้ จะทำให้หุ่นยนต์สามารถรักษาเสถียรภาพการทรงตัวได้ดีขึ้น เช่นการทรงตัวบนขาข้างเดียว

มีหลายวิธีการที่ใช้ในการควบคุมแบบอัตโนมัติ วิธีหนึ่งที่ดูง่ายก็คือ การแกว่งขาแบบธรรมชาติ (natural point) ซึ่งเป็นจุดที่ความเร็วเฉลี่ยของจุดศูนย์กลางแรงโน้มถ่วงมีค่าคงที่ หลังไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากการวางเท้าลงไปแล้วขาที่ยืนอยู่จะแกว่งมาข้างหลังเล็กน้อยเพื่อรักษาให้ความเร็วที่จุดศูนย์กลางแรงโน้มถ่วงมีค่าคงที่ การปรับให้ความเร็วที่จุดศูนย์กลางแรงโน้มถ่วงมีค่าคงที่ทำได้โดยการปรับระยะลงแตะพื้น (touchdown) จากจุดธรรมชาติ และการปรับการแกว่งขา

พารามิเตอร์ที่จะนำมาใช้ในการตัดสินใจในการควบคุมแบบอัตโนมัติ จะได้มาจากการควบคุมแบบป้อนกลับ ซึ่งต้องอาศัยการศึกษา ทั้งทางด้านระบบทางกล(mechanical system) ด้านการเคลื่อนที่ (motion) และด้านอุปกรณ์ส่งถ่ายพลังงาน



บทที่ 3

การออกแบบและการทดลอง

การออกแบบโครงงานสามารถแบ่งเป็นส่วนใหญ่ๆ ได้ 3 ส่วนคือ

1. การออกแบบส่วนไมโครคอนโทรลเลอร์
2. การออกแบบส่วนคอมพิวเตอร์เพื่อติดต่อกับผู้ใช้
3. การออกแบบส่วนขาคู่มือ

3.1 การออกแบบส่วนไมโครคอนโทรลเลอร์

3.1.1 การสร้างบอดเรต (Baud Rate)

เราต้องการให้มีบอดเรตที่ 9600 บิตต่อวินาที ซึ่งก็คือให้เกิดการสุ่ม (Sampling) ด้วยความเร็ว 9600 บิตต่อวินาที

$$\text{ช่วงเวลาระหว่างบิต} = \frac{1}{9600} = 0.104167 \times 10^{-3} \text{ วินาที} \quad \text{-----}(3.1)$$

$$\text{สัญญาณนาฬิกาภายในที่ใช้ในการปฏิบัติงาน} = \frac{1}{2(\text{External Clock})}$$

เนื่องจากเราใช้ออสซิลเลเตอร์ที่ 11.0592 เมกะเฮิร์ตซ์ เพราะฉะนั้น สัญญาณนาฬิกาภายในจะเป็น 5.5296 เมกะเฮิร์ตซ์

$$1 \text{ คาบ} = \frac{1}{5.5296 \times 10^6} = 0.1818449074074 \text{ ไมโครวินาที} \quad \text{-----}(3.2)$$

จากสมการที่ (3.1) และ (3.2) เทียบบัญญัติไครยงค์

$$0.180844907407 \times 10^{-6} \text{ วินาที} \quad \text{คือคาบของ 1 ไชเคิล}$$

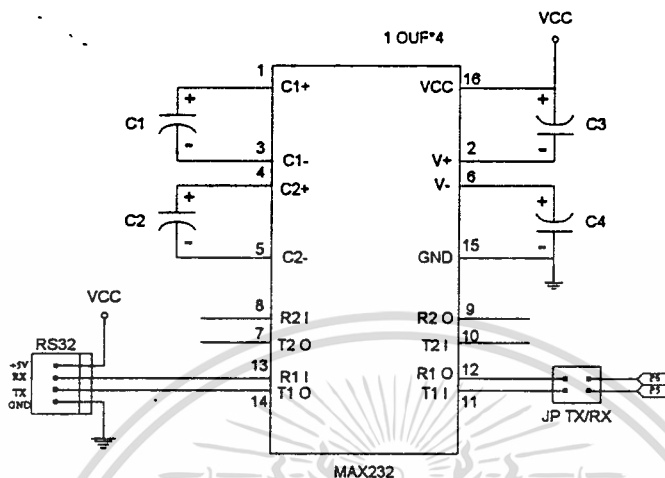
$$\text{ต้องการ } 0.104157 \times 10^{-3} \text{ วินาที} \quad \text{คือคาบของ 576 ไชเคิล} \quad \text{-----}(3.3)$$

เราจะใช้รูปทำการหาช่วงเวลาให้ได้ตามที่ต้องการดังนี้

Loop : DJNZ R, Loop

ซึ่งคำสั่ง DJNZ มีช่วงเวลาในการปฏิบัติงาน 12 ไชเคิล เพราะฉะนั้นค่าที่ใส่ใน รีจิสเตอร์ R เพื่อให้ได้ 576 ไชเคิล คือ 48 ก็จะได้การสุ่มข้อมูลที่ความเร็ว 9600 บิตต่อวินาที

3.1.2 การออกแบบวงจรแปลงจาก RS-232 เป็น TTL



รูปที่ 3.1 แสดงวงจรแปลง RS-232 เป็น TTL

ใช้ ICL232CPE เป็นไอซีที่ทำการแปลงระดับของศักดาตามมาตรฐาน RS-232 ให้อยู่ในช่วง 0-5 โวลต์ โดยจะมีการแปลง 12 โวลต์ ให้เป็น 0 โวลต์ หรือ “0” และ -12 โวลต์ ให้เป็น 5 โวลต์ หรือ “1”

ดังนั้นเอาต์พุตจากวงจรนี้จะได้

สตาร์ทบิต เป็น “0”

สตีออบิต เป็น “1”

ลำดับข้อมูลที่ออกมาจากวงจร จะมีลำดับดังนี้

start bit	D ₀	D ₁	D ₂	D ₃	D ₄	D ₅	D ₆	D ₇	stop bit
0 V									5 V

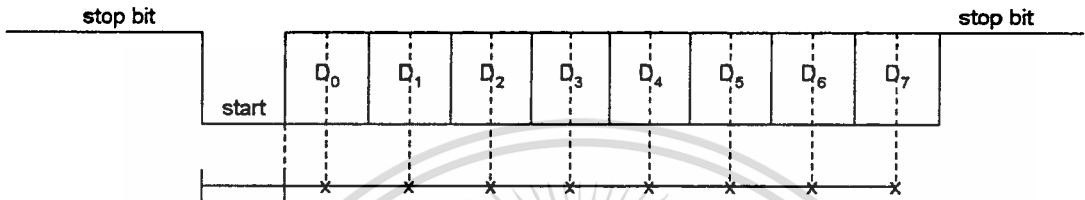
จากเอาต์พุตของวงจร จะสามารถใช้เป็นสัญญาณอินเตอร์รัพท์โดยใช้ขอบขาลงให้แก่ Z86E08 ได้ทันที โดยให้เข้าสู่ Z8 ทางพอร์ต 3.2 และใช้ IRQ0 เป็นอินเตอร์รัพท์เวกเตอร์

3.1.3 การรับข้อมูลแบบอนุกรม (Serial-In Interrupt)

จากเอาต์พุตจากวงจรแปลงสัญญาณ RS-232 เป็น TTL สถานะที่ไม่ได้ส่งข้อมูลจะมีลอจิก “1” เมื่อเริ่มส่งสตาร์ทบิต สัญญาณจะตกเป็นศูนย์ และจะทำให้เกิดการอินเตอร์รัพท์ IRQ 0

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรมย่อยตอบสนองอินเทอร์รัพท์ (Interrupt Service Routine) ต้องทำงานหน่วงเวลาไป เพื่อให้เกิดการสุ่มครั้งแรกที่กึ่งกลางบิตแรกที่เข้ามา ต้องใช้ รูป หน่วงด้วยค่าในรีจิสเตอร์ $R = 48 \times 1.5 = 72$ และจากนั้นก็สุ่มด้วยค่าบอดเรตที่ออกแบบไว้ในหัวข้อ 3.1.1 จนครบ 8 บิต แล้วจึงคืนการทำงานให้โปรแกรมอื่น



รูปที่ 3.2 แสดงการสุ่มข้อมูลจากขาสัญญาณข้อมูลอนุกรมเข้า

เขียนโปรแกรมสำหรับรับข้อมูลจากคอมพิวเตอร์ คือโปรแกรม INTERRUPT.ASM โดยมีการทำงานดังไฟล์ไคอะแกรมดังรูปที่ 3.4 โดยให้สัญญาณเข้าทางพอร์ต P3.2 เพื่อที่จะให้ขอบขา ลง (falling edge) ที่สแตร์ทบิตของลำดับสัญญาณที่ส่งมา เป็นตัวสร้างอินเทอร์รัพท์ IRQ 0 ให้ ไมโครคอนโทรลเลอร์ไปทำการสุ่มข้อมูลที่ขา P3.2 ด้วยอัตราเร็ว 9600 บิตต่อวินาที มาเก็บไว้ และประมวลผลข้อมูลที่เข้ามา เนื่องจากข้อมูลที่ส่งมาไม่ได้เป็นค่าตำแหน่งโดยตรง แต่จะมีข้อมูลอื่นๆ (overhead) ที่เป็นตัวบอกตำแหน่งของมอเตอร์ ดังนั้น โปรแกรมย่อยตอบสนองการอินเทอร์รัพท์ จะแยกข้อมูลออกจากไบต์ที่ส่งมาจากคอมพิวเตอร์ และเก็บลงสู่บัฟเฟอร์ในตำแหน่งที่ถูกต้อง ก่อนที่จะคืนการทำงานให้แก่โปรแกรมอื่นต่อไป

ส่วนโปรแกรมที่ส่งข้อมูล กำลังอยู่ในระหว่างการพัฒนา ซึ่งในระยะแรกจะยังไม่นำไปใช้ เพราะงานในระยะแรก จะเป็นการทดลองสั่งงานจากคอมพิวเตอร์เพื่อควบคุมชุดทดลองมอเตอร์

3.1.4 การคำนวณฐานเวลา 1 มิลลิวินาที เพื่อป้อนแก่เซอร์โวมอเตอร์

ใช้หลักการเดียวกันกับการสร้างบอดเรต แต่เนื่องจาก หากใช้รูปเดียว จะไม่สามารถหน่วงเวลาได้ถึง 1 มิลลิวินาที เพราะข้อจำกัดของค่ารีจิสเตอร์ที่มีขนาด 8 บิต จึงใช้รูปซ้อนโดยรูปวงใน หน่วงเวลา 0.5 มิลลิวินาที หรือ 500 ไมโครวินาที แล้ววนรูป 2 ครั้ง

คาบเวลา 0.1808449074074 ไมโครวินาที คือ มี 1 ไซเคิล

ต้องการเวลา 500 ไมโครวินาที ต้องใช้ 2764.8 ไซเคิล

กรณีที่ใช้รูป DJNZ ที่มี 12 ไซเคิล เพราะฉะนั้นต้องวนรูป $2764.8 / 12 = 230.4$ รอบ

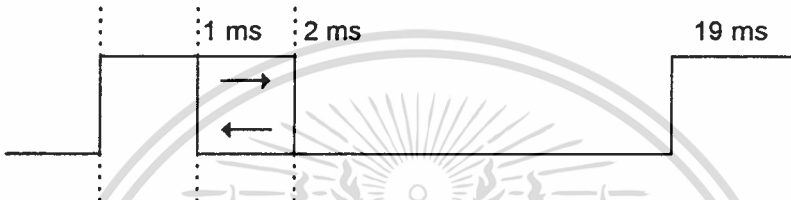
หรือ E6h

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เพราะฉะนั้น ถ้าต้องการให้หน่วยเวลา 1 มิลลิวินาที ต้องใช้รูปซ้อน $E6h \times 2$ ซึ่งเป็นฐานในการคำนวณสัญญาณพัลส์ที่ไปควบคุมคีย์เซอร์ไวมอเตอร์ต่อไป

3.1.5 การนำค่าตำแหน่งไปควบคุมมอเตอร์

ในระยะแรก ได้ทดลองเขียนโปรแกรม POSITION.ASM ที่มีลำดับการทำงานตามโฟลว์ไคอะแกรมรูปที่ 3.4 เพื่อสร้างสัญญาณพัลส์ ไปควบคุมตำแหน่งของเซอร์ไวมอเตอร์ สัญญาณพัลส์ที่ใช้ในการควบคุมตำแหน่งของมอเตอร์จะมีรูปแบบดังนี้



รูปที่ 3.3 ภาพแสดงสัญญาณพัลส์ที่ใช้ในการควบคุมมอเตอร์

คาบของสัญญาณพัลส์ จะมีค่า 19 มิลลิวินาที

ความกว้างของซิกบวคของสัญญาณพัลส์ จะเป็นตัวควบคุมตำแหน่งของมอเตอร์ โดยมีค่าอยู่ระหว่าง 1-2 มิลลิวินาที จะทำให้มอเตอร์มีตำแหน่งหมุนช้าที่สุด ไปจนถึงขวสุด

โปรแกรมที่ทดลองควบคุมตำแหน่งของมอเตอร์มี โปรแกรมย่อย one-pulse เป็นตัวสร้างสัญญาณพัลส์ขึ้นมา ที่มีคาบเวลาคงที่ 19 มิลลิวินาที ส่วนค่า “1” ของสัญญาณพัลส์ จะขึ้นอยู่ดับค่ารีจิสเตอร์ R5

3.1.6 โปรแกรมทดสอบการหมุนของมอเตอร์

โปรแกรม DRIVER.ASM เป็นโปรแกรมที่ทดลองหมุนมอเตอร์ให้มีการหมุนจากซ้ายสุดไปขวาสุด โดยโปรแกรมจะวนลูปเพื่อสร้างสัญญาณพัลส์ โดยอาศัยโปรแกรมย่อย one-pulse ดังแสดงในโฟลว์ไคอะแกรมรูปที่ 3.5

ทั้งโปรแกรม POSITION.ASM และ DRIVER.ASM ที่พัฒนาขึ้น ได้นำไปทดลองควบคุมมอเตอร์ โดยโปรแกรมหลักจะทำหน้าที่นำค่าตำแหน่งของมอเตอร์จากบัพเฟอร์ ส่งไปยังมอเตอร์แต่ละตัวผ่านพอร์ต P2.0-P2.7 และ P0.0-P0.7 รวม 10 ตัว หมุนเวียนไปตลอดเวลา ซึ่งไม่สามารถใช้ในการควบคุมได้ เนื่องจากการส่งค่าไปยังพอร์ตทั้งสองรวม 10 ขา ทำให้คาบเวลาของสัญญาณควบคุมมีมากเกินไปกว่า 19 มิลลิวินาที ทำให้แรงบิดของมอเตอร์ไม่สม่ำเสมอ จึงได้พัฒนาโปรแกรมส่งสัญญาณควบคุมขึ้นมาใหม่ คือโปรแกรม DRIVE.ASM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1.7 โปรแกรม DRIVE.ASM

มีกระบวนการทำงานดังไฟล์ไคอะแกรมที่ 3.7 ออกแบบให้แก้ไขจุดบกพร่องในโปรแกรม POSITION.ASM โดยคิดกระบวนการเปิด-ปิดสัญญาณควบคุมใหม่ อาศัยหลักการคือ เริ่มต้นจะเปิดทุกขาของพอร์ตเป็นลอจิก '1' แล้ววนลูปโปรแกรมเพื่อไล่ปิดขาพอร์ตให้เป็น '0' ตามลำดับจากน้อยไปมาก ซึ่งต้องอาศัยโมดูลการเรียงลำดับ ซึ่งพัฒนาขึ้นมาโดยมีกระบวนการคิดตามรูปไฟล์ไคอะแกรมที่ 3.8

จากการทำงานลักษณะนี้จะทำให้สามารถเปิดสัญญาณควบคุมมอเตอร์ได้พร้อมกัน และเรียงลำดับปิดตามลำดับน้อยไปมาก โดยอาศัยการนับขึ้นของรีจิสเตอร์เป็นตัวกำหนดว่าถึงเวลาปิดขาใดของพอร์ต

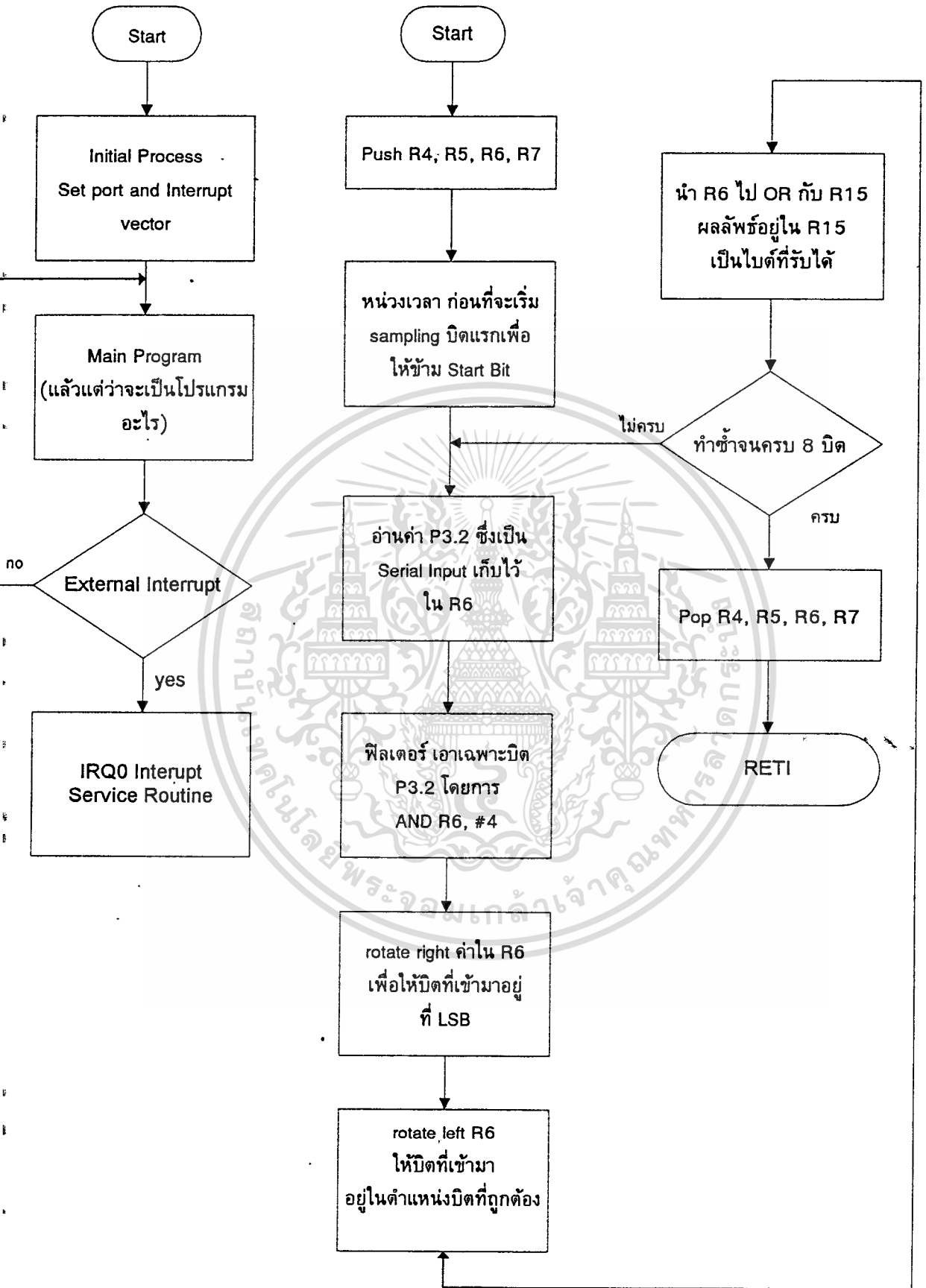
3.1.8 โปรแกรม NEWDRIVE.ASM

แม้ว่าจะพัฒนาโปรแกรม DRIVE.ASM ขึ้นมาแก้ไขจุดบกพร่องของโปรแกรม POSITION.ASM แล้วก็ตาม ก็ยังพบปัญหาในการควบคุมของ DRIVE.ASM ดังจะกล่าวถึงในเรื่องของการทดลอง จึงจำเป็นต้องพัฒนากระบวนการสร้างสัญญาณควบคุมขึ้นมาใหม่ ได้แก่ โปรแกรม NEWDRIVE.ASM

โปรแกรมนี้นี้มีกระบวนการทำงานตามไฟล์ไคอะแกรมที่ 3.9 มีหลักการทำงานคือจะต้องเรียงลำดับข้อมูล และโมดูลการเตรียมค่าสัญญาณควบคุม (คือโมดูล PREPARE ซึ่งคัดมาจากโปรแกรม PREPARE.ASM ดังมีลำดับการทำงานตามไฟล์ไคอะแกรมที่ 3.10) เพื่อรวบรวมมอเตอร์ที่มีค่าควบคุมเท่ากัน ให้ไมโครคอนโทรลเลอร์ทำการปิดสัญญาณควบคุมให้พร้อมกัน โดยแทนที่จะเรียงปิดขาสัญญาณทีละขา ก็จะปิดขาสัญญาณที่มีค่าควบคุมเท่ากัน ให้ปิดพร้อมกัน

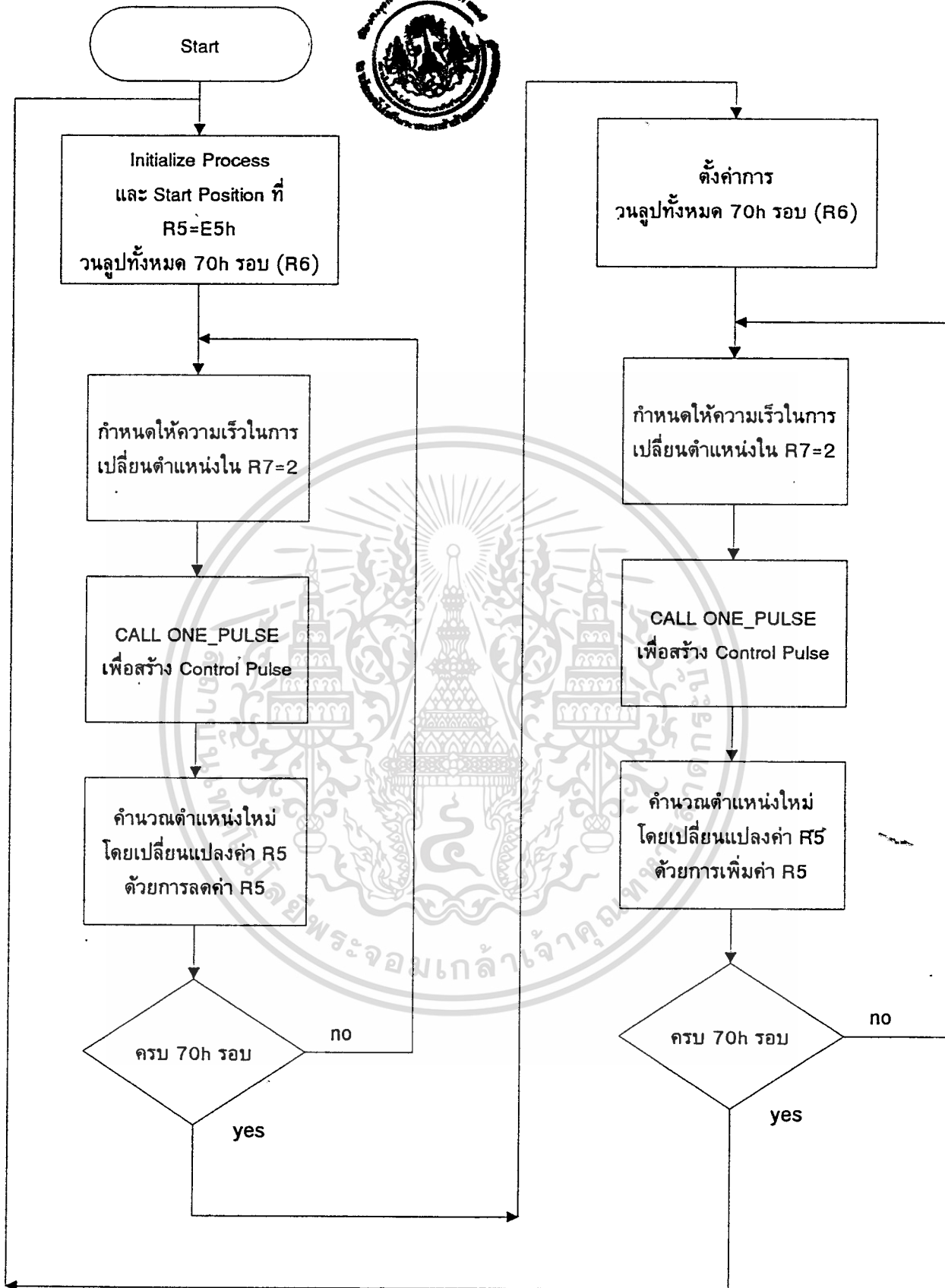
main program

IRQ0 Service Routine

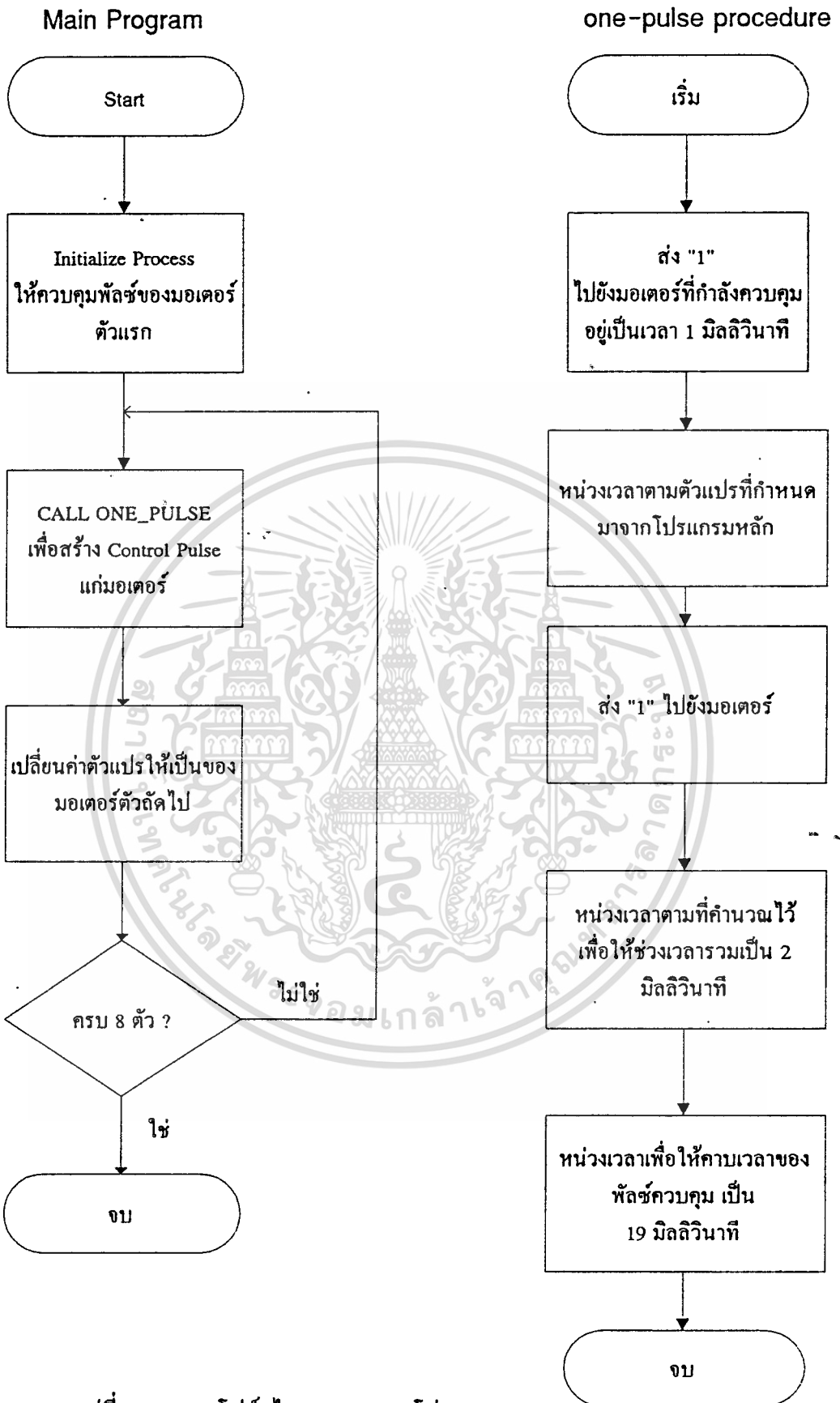


รูปที่ 3.4 แสดงโฟลว์ไคอะแกรมของโปรแกรม INTERRUPT.ASM

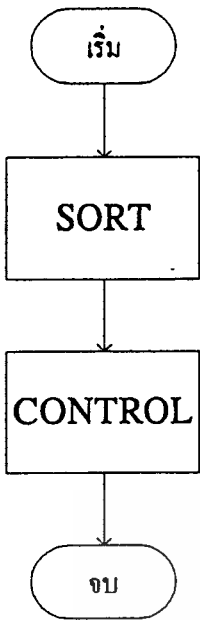
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.5 แสดงโฟลว์ไคอะแกรมของโปรแกรม DRIVER.ASM

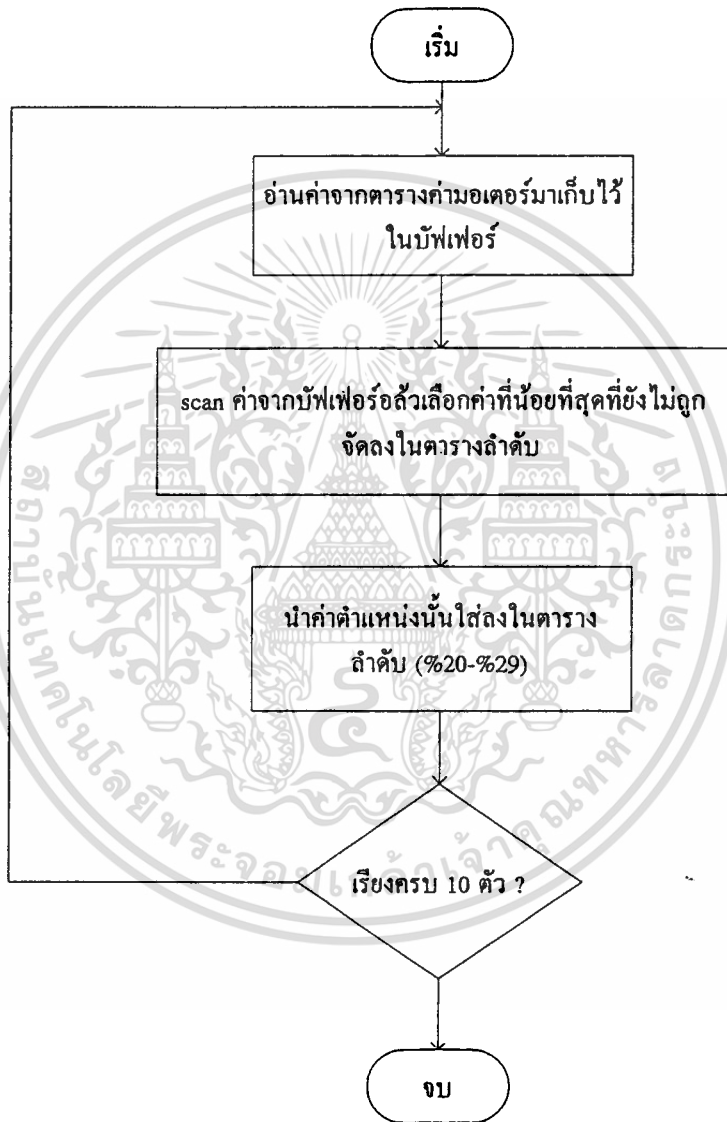


รูปที่ 3.6 แสดงโฟลว์ไดอะแกรมของโปรแกรม POSITION.ASM



รูปที่ 3.7 แสดงไฟล์ไคอะแกรมของโปรแกรม DRIVE.ASM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

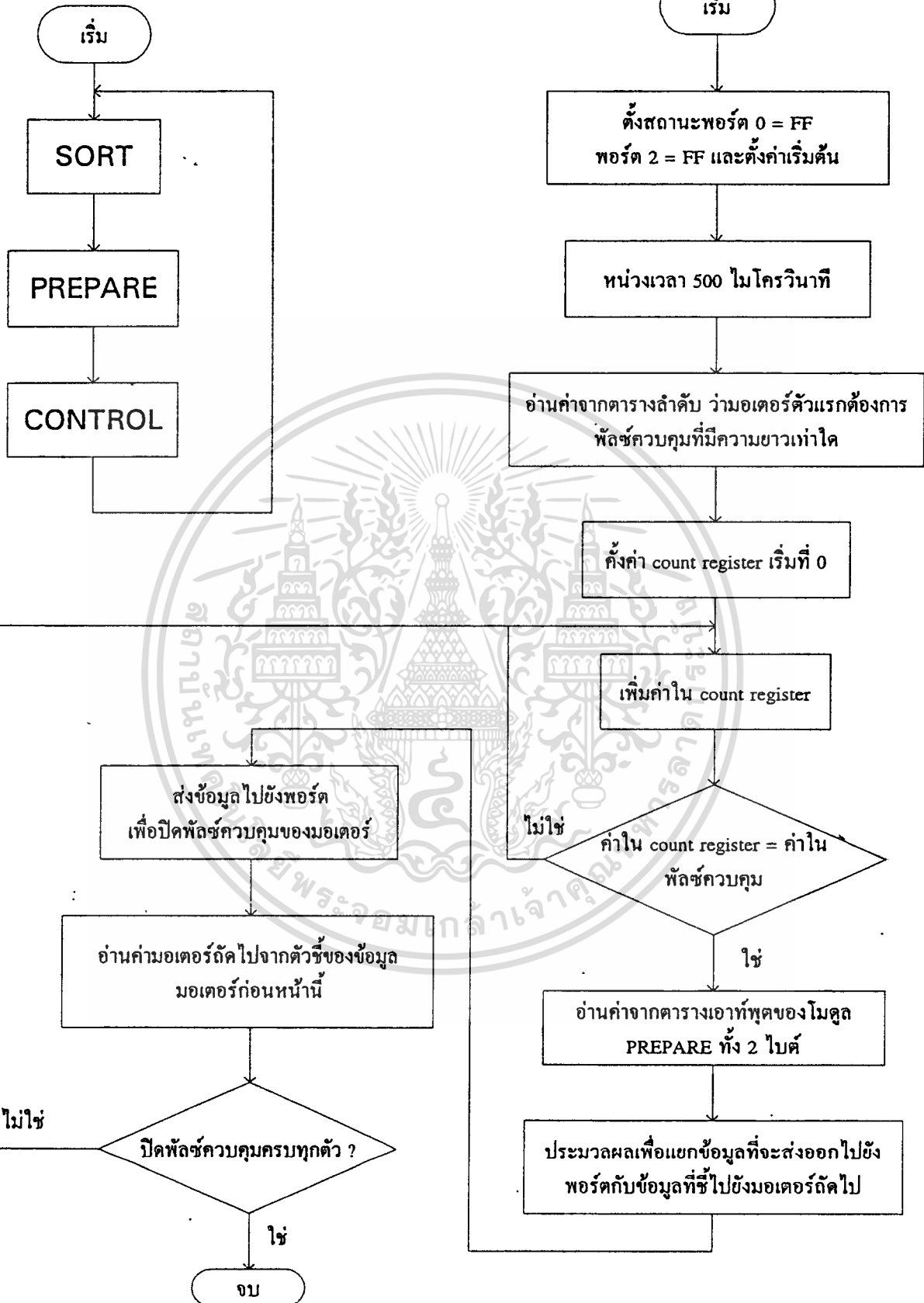


รูปที่ 3.8 แสดงโฟลว์ไคอะแกรมของโปรแกรม SORT.ASM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

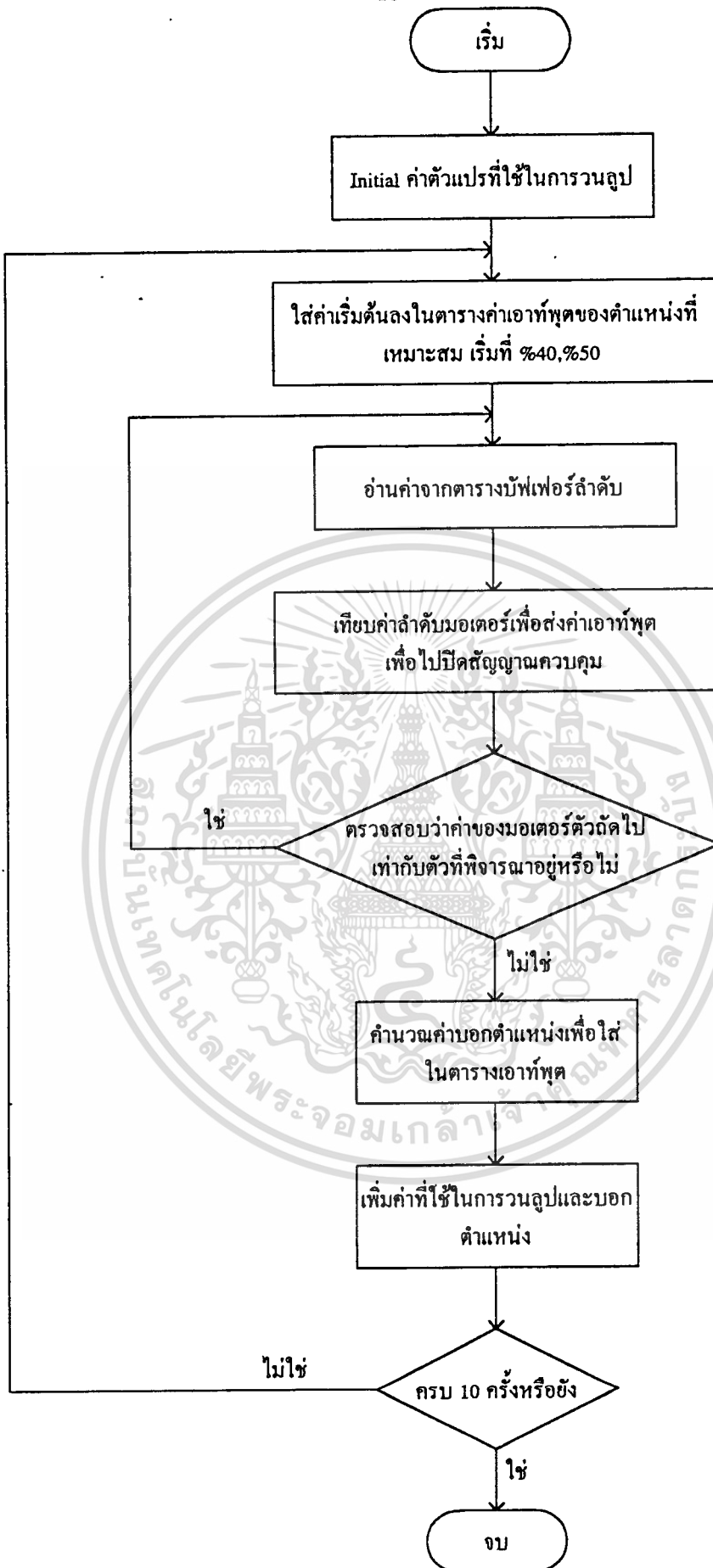
main program

control



รูปที่ 3.9 แสดงโฟลว์ไคอะแกรมของโปรแกรม NEWDRIVE.ASM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.10 แสดงโฟลว์ไคอะแกรมของโปรแกรม PREPARE.ASM

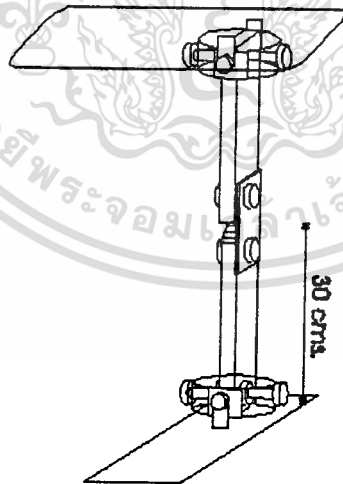
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2 การออกแบบทางด้านกลไกของส่วนขาหุ่นยนต์

การเคลื่อนที่ของมนุษย์เป็นลักษณะการเคลื่อนที่โดยใช้ขา และสามารถทรงตัวได้ด้วยขาเพียง 2 ข้างเท่านั้น หุ่นที่จำลองการเคลื่อนที่มาจากมนุษย์จะเรียกว่า Biped Walking Robot หมายถึง หุ่นที่เคลื่อนที่ด้วยขาเพียง 2 ข้างเท่านั้น

3.2.1 แนวทางการออกแบบ

เนื่องจากต้องการออกแบบให้หุ่นยนต์มีลักษณะการเคลื่อนที่ ที่มีการเลียนแบบมาจากขาของมนุษย์ ดังนั้น ตัวหุ่นจึงจำเป็นต้องมีทั้งหมด 2 ขา และแต่ละขามีจำนวนท่อน (Link) ทั้งหมดข้างละ 3 ท่อน ซึ่งก็คือ ขาท่อนบน ขาท่อนล่าง และเท้านั่นเอง ส่วนการออกแบบทางด้านข้อต่อ (Joint) นั้นเพื่อให้หุ่นสามารถทรงตัวได้ใน 3 มิติ คือทั้งระนาบด้านหน้า และระนาบด้านข้าง เราจึงกำหนดให้ที่ท่อนเอว หรือที่ส่วนต้นขา มีการหมุนได้ 2 แนวแกนคือ ด้านหน้าและด้านข้าง ต่อมาที่ส่วนเข่า กำหนดให้มีแกนหมุนได้ 1 แกน คือด้านหน้า และส่วนสุดท้ายคือข้อเท้าจะสามารถเคลื่อนที่ได้ 2 แนวแกนเช่นเดียวกับที่ต้นขา ซึ่งการเคลื่อนที่ในแนวแกนทางด้านข้างจะทำให้หุ่นสามารถชดเชยน้ำหนักได้เมื่อขาอีกข้างหนึ่งยกขึ้น และแนวแกนทางด้านหน้าจะทำให้หุ่นสามารถชดเชยโมเมนต์ที่เกิดจากขาที่ย่างก้าวไปข้างหน้าหรือข้างหลัง



รูปที่ 3.11 แสดงโครงสร้างหุ่นยนต์ที่ตั้งเป้าหมายไว้

3.2.2 การออกแบบในรายละเอียด

ทั้งหมดข้างต้นเป็นแนวทางคร่าวๆที่ออกแบบไว้ในตอนแรก ส่วนในการทำงานจริงนั้น เราไม่สามารถสร้าง ให้หุ่นเป็นไปตามโมเดลได้ เนื่องจากสาเหตุหลายๆประการด้วยกัน ไม่ว่าจะเป็น เอกสารเป็นเอกสารที่ลงวันเวลาที่ผ่านไปนานแล้วเพื่อให้ง่ายต่อการทำความเข้าใจ ไม่อย่างนั้น เราจะไม่สามารถสร้าง ไม่อย่างนั้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เป็น ข้อจำกัดทางด้านอุปกรณ์ คือ การเคลื่อนที่เป็นการเดินนั้น ตัวหุ่นต้องแบกรับน้ำหนักมอเตอร์ทั้งหมดรวมทั้งโครงสร้างอื่นๆ ไว้ด้วย ดังนั้นน้ำหนักของอุปกรณ์ต่างๆ จึงเป็นตัวแปรที่สำคัญมากในการออกแบบ

ในเบื้องต้น หุ่นต้นแบบตัวแรก ออกแบบให้ใช้ลูมินีซึมเป็นวัสดุหลัก แต่ปรากฏว่า ตัวหุ่นมีน้ำหนักมาก มากจนตัวมอเตอร์ที่กำหนดไว้ ไม่สามารถรับน้ำหนักได้ วัสดุต่อมาจึงเปลี่ยนมาใช้ไม้บัลซาร์ จากการทดสอบ ถึงแม้ไม้บัลซาร์จะมีน้ำหนักเบา แต่ตัวหุ่นไม่มีความคงทนพอ ไม่ว่าจะเป็นที่จุดเชื่อม และจุดที่รับแรงอื่นๆ จากการศึกษางานวิจัยด้านหุ่นยนต์จากต่างประเทศ วัสดุหนึ่งที่ได้รับคามนิยม และในที่สุดก็ได้รับเลือกให้เป็นโครงสร้างหลักของตัวหุ่น ซึ่งก็คืออะคริลิก เนื่องจากเป็นวัสดุที่มีความเหนียวและทนทาน รวมทั้งน้ำหนักเบา และยังสามารถได้ง่ายอีกด้วย

3.2.3 มิติและตำแหน่งของอุปกรณ์ต่างๆของหุ่น

แนวคิดคร่าวๆ ในการวางตำแหน่งมอเตอร์ก็คือ ตำแหน่งของมอเตอร์ควรวางให้อยู่ใกล้ส่วนต้นขามากที่สุด เพื่อลดโมเมนต์ที่เกิดจากน้ำหนักของตัวมอเตอร์เองที่มีต่อระบบรวม แต่อย่างไรก็ดี น้ำหนักของสิ่งที่อยู่ที่เอวของตัวหุ่นจะไม่มีผลต่อการทรงตัวเท่าไร เนื่องจาก น้ำหนักทั้งหมดที่เอวจะตกลงสู่ฝ่าเท้าทันที ส่วนน้ำหนักที่อยู่ที่ทำอีกข้างหนึ่ง จะเป็นส่วนที่ต้องการการการชดเชย ดังนั้นการออกแบบจึงออกแบบให้มอเตอร์ทั้งหมดอยู่ที่เอวของหุ่น และส่งแรงบิดทั้งหมดผ่านทางรอกและสายพานลงมายังส่วนต่างๆของหุ่น นั่นคือความพยายามที่จะให้หุ่นมีลักษณะที่สมบรูณ์ที่สุด แต่อย่างไรก็ดีในภายหลัง ก็ตัดสินใจใช้การต่อตรง (Direct Coupling) แทน เพื่อหลีกเลี่ยงปัญหาด้านวัสดุ เช่นรอก ซึ่งไม่สามารถหาขนาดที่ต้องการได้ ดังนั้นด้วยแนวคิดใหม่ ตัวเซอร์โวมอเตอร์ จะอยู่ตามข้อต่างๆ และส่งสัญญาณควบคุมมาทางสายจากทางส่วนเอว ที่เอวจะเป็นที่ตั้งของแหล่งจ่ายไฟและคอนโทรลเลอร์

3.3 การออกแบบส่วนคอมพิวเตอร์เพื่อติดต่อกับผู้ใช้

ทางด้านซอฟต์แวร์กำหนดไว้ให้เป็นโปรแกรมที่ทำงานบนไมโครซอฟต์วินโดวส์ และสื่อสารกับตัวไมโครคอนโทรลเลอร์ผ่านทางพอร์ต RS-232

โปรแกรมที่ทำการควบคุมบน PC จะทำหน้าที่หลักคือส่งตำแหน่งของเซอร์โวมอเตอร์ตัวต่างๆ มาให้กับตัวคอนโทรลเลอร์ โดยค่าตำแหน่งของเซอร์โวมอเตอร์ สามารถได้มาจากหลายวิธีด้วยกัน ไม่ว่าจะเป็นจากการรับอินพุต โดยตรงมาจากผู้ใช้ หรือจะเป็นการโปรแกรมเป็นลำดับ ไว้ก่อนและทยอยส่งมาตามเวลาที่เหมาะสม หรือจะเป็นการคำนวณมาจาก โมเดลทางคณิตศาสตร์ ซึ่งทำงาน

ร่วมกับการป้อนกลับ

เอกสารนี้เป็นเอกสารต้นฉบับงานวิจัยสำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

8.3.1 การควบคุมแบบแมนนวล

ขอบเขตของโครงการนี้ ครอบคลุมการสร้างระบบกลไกของตัวหุ่น การออกแบบตัวควบคุม และในส่วนบนไมโครคอมพิวเตอร์ จะครอบคลุมเฉพาะการควบคุมแบบแมนนวลเท่านั้น การควบคุมแบบอัตโนมัติจำเป็นต้องได้รับการพัฒนาต่อไป

โปรแกรมที่สร้างไว้มีสองโปรแกรมด้วยกัน โปรแกรมแรกคือโปรแกรม JBENCH.EXE ทำหน้าที่รับคำสั่งจากผู้ใช้งานและส่งให้ระบบคอนโทรลเลอร์แบบเรียลไทม์ (Realtime) เพื่อควบคุมเซอร์โวมอเตอร์แต่ละตัว แบบทันทีทันใด ซึ่งมีประโยชน์ในการทดสอบระบบและทดลองตำแหน่งของเซอร์โวเท่านั้น ไม่มีส่วนช่วยใดๆ ในการเคลื่อนที่ของตัวหุ่น

โปรแกรมที่สอง (JBIPED.EXE) เป็นโปรแกรมที่ยังคงต้องการการควบคุมและกำหนดค่าต่างๆ จากมนุษย์ แต่ได้รับการพัฒนาไปจากโปรแกรมแรกมาก นั่นคือ ผู้ใช้สามารถกำหนดได้ว่า ต้องการให้หุ่นมีลักษณะเป็นอย่างไร โดยจัดการกับภาพเหมือนกับจัดการกับหุ่นยนต์จริงๆ ซึ่งเรียกการควบคุมแบบนี้ว่า Virtual Model Control นอกจากนั้นผู้ใช้งานยังสามารถกำหนดได้ว่า ต้องการให้มีการเคลื่อนไหวเป็นอย่างไร โดยกำหนดเป็นเฟรม (Frame) ผู้ใช้ต้องจัดทำทางของหุ่นยนต์โดยมีพื้นฐานอยู่บนเฟรม หลังจากนั้นโปรแกรมจะจัดการทยอยส่งข้อมูลแต่ละเฟรมและ ข้อมูลของเซอร์โวแต่ละตัว ไปยังคอนโทรลเลอร์ ที่เวลาที่ถูกต้อง ผู้ใช้ยังสามารถจัดเก็บข้อมูลเหล่านั้นเป็นไฟล์ไว้ได้ เพื่อนำกลับมาเรียกใช้ในภายหลัง โดยไฟล์มีนามสกุลเป็น .PDA (Path-Draft Animation)

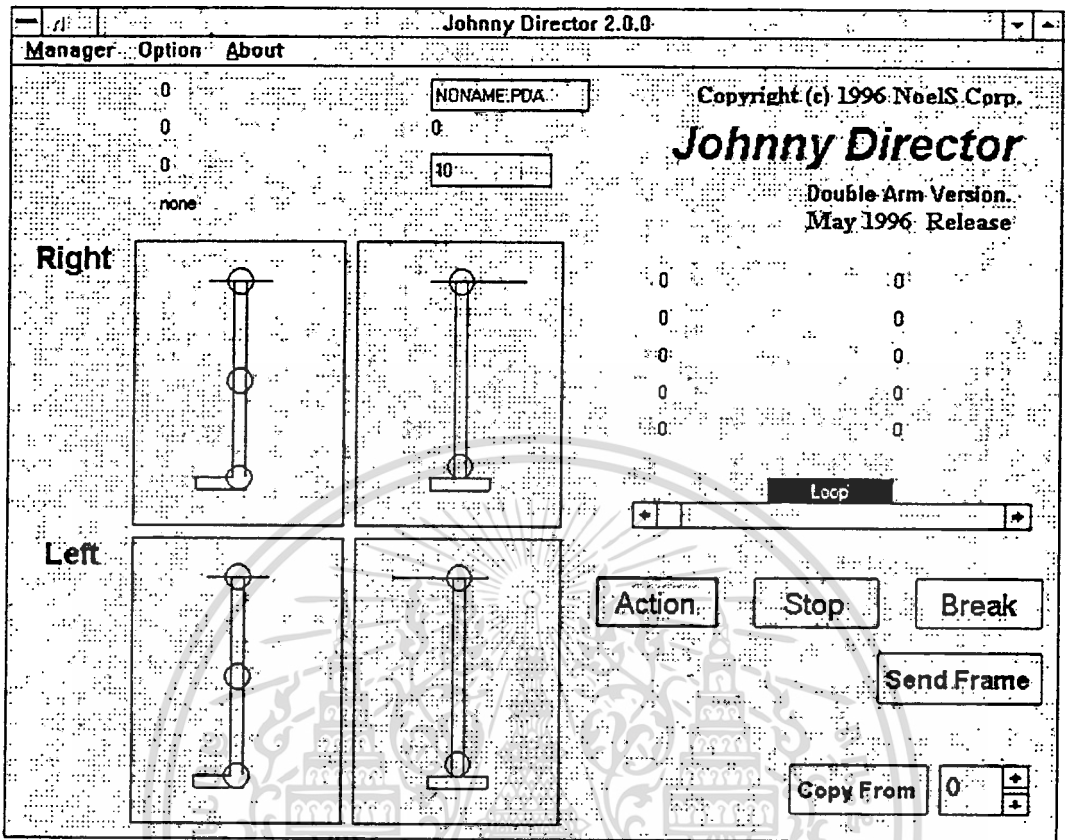
อีกสิ่งหนึ่งที่พัฒนาไปจากโปรแกรมเดิม นั่นคือ ผู้ใช้สามารถเลือกพอร์ตที่ต้องการสื่อสารด้วยได้ภายในโปรแกรม และยังสามารถควบคุมพารามิเตอร์ (Parameter) เกี่ยวกับพอร์ตได้เกือบทุกพารามิเตอร์

8.3.2 โครงสร้างของโปรแกรมควบคุม

เนื่องจากการเขียนโปรแกรมบนไมโครซอฟต์วินโดวส์ เป็นการเขียนโปรแกรมที่ขึ้นกับเหตุการณ์ (Event-Driven) ซึ่งโปรแกรมจะมีลักษณะการไหลตามเหตุการณ์ต่างๆ ที่เกิดขึ้น ไม่ว่าจะเป็นคำสั่งจากผู้ใช้งาน หรือจากอุปกรณ์ต่างๆ ในที่นี้จะพูดถึงแต่โปรแกรม JBIPED.EXE โปรแกรมนี้จะประกอบด้วยส่วนประกอบสำคัญๆ 4 ส่วนด้วยกัน คือ

- ส่วนการจัดทำทางหุ่น
- การจัดการ Animation
- การจัดการไฟล์

เอกสารนี้เป็นเอกสารที่สงวนลิขสิทธิ์โดยบริษัท RS-232 ศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.12 แสดงโปรแกรมจัดลำดับท่าทางที่เสร็จสมบูรณ์

3.3.2.1 ส่วนการจัดท่าทางหุ่น

โปรแกรมจะแสดงภาพโครงสร้างของหุ่นยนต์ตามรูปร่างที่เป็นอยู่จริง ผู้ใช้จะเลือกว่าต้องการจะขยับส่วนไหน โดยโปรแกรมต้องทำหน้าที่ตรวจสอบว่าผู้ใช้งานกำลังสัมผัสกับส่วนไหนของหุ่นอยู่ โดยตรวจสอบจากตำแหน่งที่ผู้ใช้คลิก และนำไปหาระยะห่างจากเส้นที่เป็นคาดว่าจะ เป็นจุดสัมผัส และกำหนดให้สามารถมีความผิดพลาดได้เล็กน้อย เมื่อสามารถระบุได้แล้วผู้ใช้ ต้องการจะเคลื่อนไหวส่วนไหน หลังจากนั้นก็ตรวจสอบการลาก (Drag) ของผู้ใช้ นำตำแหน่งที่ได้ ไปคำนวณความชันของเส้นต่างๆ และเปลี่ยนแปลง (update) ข้อมูลของทุกองค์ประกอบ และวาดภาพเพื่อให้ตรงกับความเป็นจริงมากที่สุด จากนั้นก็ส่งข้อมูลที่มีการเปลี่ยนแปลงจากเดิมออกสู่ RS-232

3.3.2.2 การจัดการท่าทางการเคลื่อนไหว (Animation)

การจัดการท่าทางการเคลื่อนไหวจะเป็นการควบคุมลำดับการเคลื่อนไหวของหุ่น โดย โปรแกรมจะแบ่งการเคลื่อนไหวที่เป็น เฟรมซึ่งผู้ใช้สามารถกำหนดได้ว่าต้องการมีจำนวนเฟรม ทั้งหมดกี่เฟรม ไม่ว่าจะกี่เฟรมก็ตาม อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หมดเท่าไร และระยะเวลาทั้งหมดที่ใช้เดินทางจากเฟรมแรก ไปถึงเฟรมสุดท้ายใช้เวลาเท่าไร ส่วนทางด้านโครงสร้างข้อมูลที่เลือกใช้กับการจัดการเฟรมนั้น เลือกใช้โครงสร้างข้อมูลแบบอาร์เรย์ (Array) เนื่องจากมีความเหมาะสมที่สุด เพราะเฟรมก็คืออาร์เรย์ นั่นเอง

จากลำดับการทำงานทั้งหมด โปรแกรมจะแบ่งออกเป็น 3 ส่วนด้วยกัน นั่นคือส่วนแรก เป็นส่วนที่เริ่มการเคลื่อนไหว คือเริ่มจากท่าหยุดนิ่งไปสู่ท่าเดินต่อเนื่อง ช่วงต่อมาจะเป็นช่วงที่วิ่ง ลูบจึงเป็นท่าที่วนซ้ำๆ สลับขาทั้งสองใช้ในการเดินตรง ส่วนสุดท้ายคือส่วนที่เป็นท่าจบนั่นเอง โปรแกรมจะสามารถกำหนดได้ว่าต้องการให้หุ่นทำงานวนเป็นลูบหรือไม่ ถ้ากำหนดให้เป็นการวน ลูบ โปรแกรมก็จะทำการวนในช่วงกลางให้จนกว่าผู้ใช้จะกดปุ่มให้หยุดเดิน หลังจากนั้น โปรแกรมก็จะทำงานต่อในช่วงที่ 3 จนหยุด แต่ถ้าไม่มีการวนลูบ โปรแกรมก็จะทำงานเพียงครั้งเดียว รอบเดียวและหยุดทำงาน

3.3.2.3 การจัดการไฟล์

เนื่องจากในภาษา Visual Basic ได้กำหนด ไวยากรณ์ภาษาเพิ่มเติมจากภาษา BASIC เดิม ในการจัดการกับไฟล์ ดังนั้นการจัดการ ไฟล์ที่ได้จึงแตกต่างไปจากเดิม เนื่องจากกำหนดให้ข้อมูลที่ ใช้มีนามสกุล .PDA ดังนั้นการเปิดไฟล์จึงกำหนดให้เป็นแบบไบนารีและจัดเก็บข้อมูลเป็นเรคคอร์ด (Record) รูปแบบของคำสั่งที่ใช้เปิดไฟล์เป็นดังนี้

```
Text3.Text = Comon2.FileName
```

```
Open Comon2.FileName For Random As 2 Len = 32
```

โดยในแต่ละเรคคอร์ด มีโครงสร้างข้อมูลเป็นฟิลด์ (Field) ต่างๆดังนี้

```
Private Type degrec
```

```
    d1 As Single
```

```
    d2 As Single
```

```
    d3 As Single
```

```
    d4 As Single
```

```
    d5 As Single
```

```
    d6 As Single
```

```
    d7 As Single
```

```
    d8 As Single
```

```
End Type
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตัวแปรแต่ละตัวจะเป็นชนิดซิงเกิล (Single) ซึ่งก็คือซิงเกิลพรีซิชั่น (Single Precision) เป็นเลขทศนิยม ใช้เก็บตำแหน่งมุมของเซอร์โวมอเตอร์แต่ละตัวในแต่ละขณะ กำหนดให้เป็นมุมแบบสัมบูรณ์ (Absolute) วัดจากระบบมุมทางคณิตศาสตร์ โดยมีช่วงการจัดเก็บคือ 0 ถึง 360 องศา เทียบกับเฟรมอ้างอิงของหุ่น (Inertial Frame of Reference) การจัดเก็บจะทำการเป็นเรคคอร์ด โดยแต่ละเรคคอร์ดจะแทนเฟรม การเคลื่อนที่ 1 เฟรม และคำสั่งที่ใช้ในการจัดเก็บและเรียกคืนคือ

Get #2, Record, BufferVariable

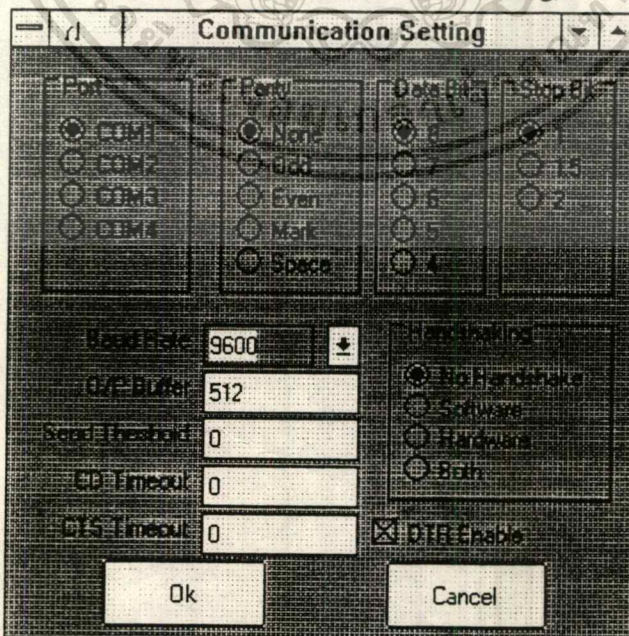
Put #2,Record, BufferVariable

ดังนั้นขนาดของไฟล์ผลลัพธ์ที่ได้จึงแปรไปตามจำนวนเฟรมที่กำหนดไว้ในระบบ

3.3.2.4 ระบบการสื่อสารผ่าน RS-232

การสื่อสารบนไมโครซอฟต์วินโดวส์ นั้นมีการจัดระบบทรัพยากรไว้อย่างเป็นระบบ นั่นคือการแบ่งปันกันใช้งานทรัพยากรต้องชัดเจน และแนวทางการเรียกใช้งานพอร์ตอนุกรม มี 2 แนวทางด้วยกันคือ การใช้ผ่านทางโครงสร้างภาษาเดิมของภาษา BASIC กับการใช้งานผ่านทางออปเจกต์การสื่อสาร (Communication Object) ของ Visual Basic

ในโครงงานนี้เลือกใช้วิธีหลังคือการใช้งานผ่านทางออปเจกต์การสื่อสาร เนื่องจากมีความเข้ากันได้กับไมโครซอฟต์วินโดวส์มากกว่า นอกจากนี้ยังสามารถกำหนดพารามิเตอร์ที่จำเป็นอื่นๆ ได้อีกมากมาย เช่น เอาท์พุทบัฟเฟอร์ (Output Buffer), อินพุทบัฟเฟอร์ (Input Buffer), เซนดเทรชโฮลด์ (Send Threshold) และระบบการทำแฮนด์เชกกิ้ง (handshaking)



รูปที่ 3.13 ค่าพารามิเตอร์ของพอร์ตสื่อสารที่ตั้งผ่านออปเจกต์การสื่อสาร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การเรียกใช้งานออปเจกต์ ตัวนี้ จะเรียกผ่านทางออปเจกต์ MSComm โดยมีการใช้งานในโปรแกรมดังนี้

MSComm.Commport	ใช้เลือกพอร์ตที่ต้องการทำงานด้วย
MSComm.Settings	ใช้กำหนดค่าพารามิเตอร์หลักๆ เช่น บอดเรต , พาริตี, บิตข้อมูล และบิตสิ้นสุดข้อมูล (Stop Bit)
MSComm.OutBuffer	ใช้กำหนดขนาดของเอาท์พุทบัฟเฟอร์
MSComm.SThreshold	ใช้กำหนดขนาดของเซนต์เทรสโฮลด์
MSComm.DTREnable	ใช้เลือกว่าจะใช้สัญญาณ DTR หรือไม่
MSComm.Handshaking	ใช้เลือกรูปแบบการทำแฮนด์เชกคิงว่าจะจะเป็นแบบฮาร์ดแวร์, ซอฟต์แวร์ หรือไม่มีการทำแฮนด์เชกคิง
MSComm.PortOpen	ใช้เปิดและปิดพอร์ตอนุกรม เพื่อเริ่มต้นการใช้งานและปิดการใช้งาน รวมถึงการฟลัช (Flush) ข้อมูลที่เหลือออกไปยังพอร์ตด้วย

ด้วยการเรียกใช้งานออปเจกต์และวิธีการ (Method) ของออปเจกต์สื่อสารตัวนี้ เราสามารถที่ทำงานที่ต้องการให้ดูด้วยได้ ไม่ว่าจะเป็นการ ตั้งค่าต่างๆให้กับระบบการสื่อสาร การเปิดและปิดพอร์ต และการส่งและรับค่าไปสู่พอร์ต

บทที่ 4

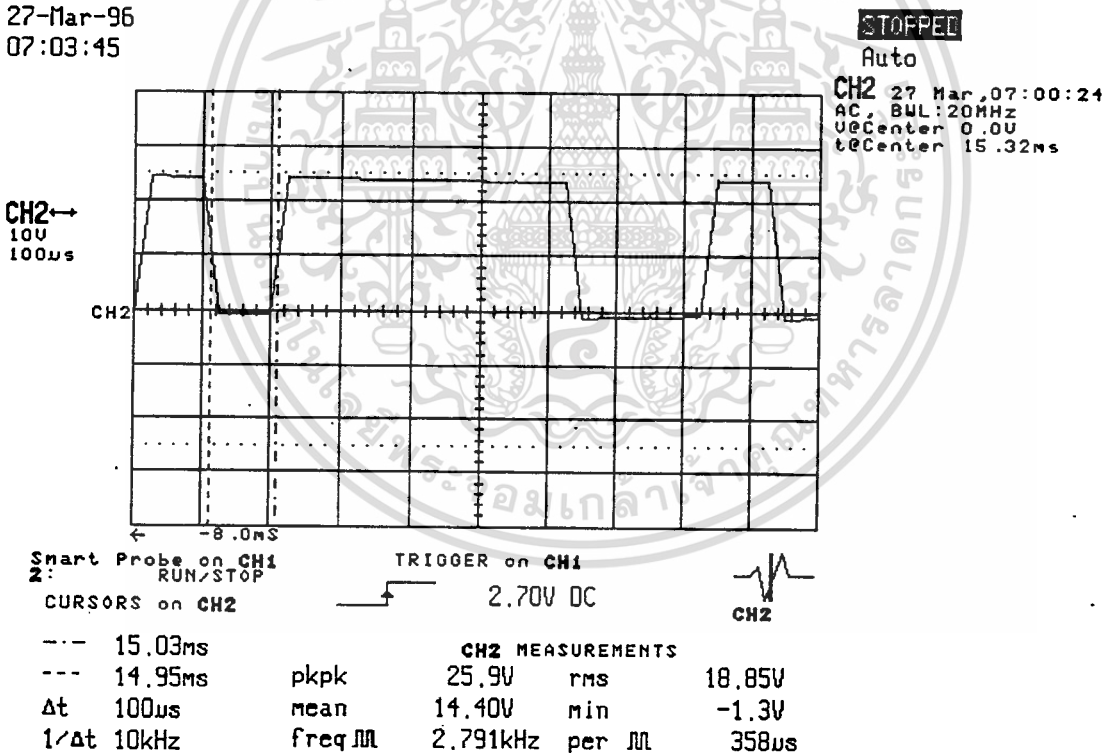
ผลการทดลอง

4.1 การทดลองของไมโครคอนโทรลเลอร์

4.1.1 การทดลองการรับข้อมูลอนุกรมผ่านพอร์ต RS-232

(1) ผลการวัดสัญญาณจากเอาต์พุตของพอร์ตอนุกรม RS-232 เมื่อส่งข้อมูล 'A' ออกมา ด้วยความเร็ว 9600 บิตต่อวินาที จะเห็นได้ว่าระดับสัญญาณแรงดันขดถึงขดมีค่า 25.9 โวลต์ และ 1 บิตที่ส่งออกมา มีความกว้างของสัญญาณประมาณ 100 ไมโครวินาที และมีบิตเริ่มต้น และ บิตสิ้นสุดอยู่ที่หัวและท้ายของลำดับสัญญาณดังรูปที่ 4.1

27-Mar-96
07:03:45



รูปที่ 4.1 แสดงลำดับสัญญาณจากพอร์ตอนุกรม 1

(2) ผลการรันโปรแกรม INTERRUPT.ASM ซึ่งมีการทดสอบการเรียก IRQ0 ได้ค่ารหัส ASCII ของข้อมูลที่ส่งออกจากเครื่องคอมพิวเตอร์ผ่านโปรแกรมเทอร์มินอล เก็บไว้ในรีจิสเตอร์

R15

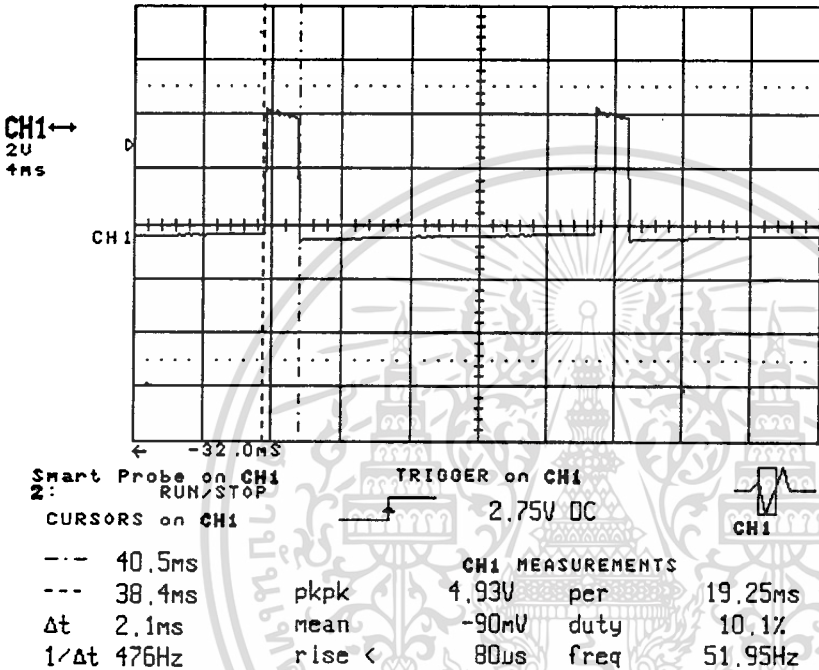
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.1.2 การทดลองวัดค่าสัญญาณควบคุมมอเตอร์ที่ส่งออกจากชุดส่งวิทยุบังคับ ได้ต่อชุด บังคับวิทยุ ซึ่งประกอบด้วย ภาควง ภาควัฒ และคิซีเซอร์ไวโมเตอร์ ทำการจับสัญญาณของ สัญญาณควบคุมที่ถูกส่งเข้าสู่คิซีเซอร์ไวโมเตอร์ ได้สัญญาณดังรูปที่ 4.2

22-Mar-96 Recalibration Suggested (SYSTEM CONFIG Menu)
23:26:37

RUN 
Auto

CH1 22 Mar, 23:26:23
AC, BWL: 20MHz
V@Center -128mV
t@Center 50.9ms



รูปที่ 4.2 แสดงรูปสัญญาณจากชุดบังคับวิทยุที่เข้าสู่คิซีเซอร์ไวโมเตอร์

จากการจับสัญญาณจากชุดบังคับวิทยุ เมื่อปรับคันบังคับให้มอเตอร์หมุนบิดซ้ายสุด และ ขวาสุด ได้ความกว้างของสัญญาณลอจิก '1' เปลี่ยนแปลงจาก 1 มิลลิวินาที ถึง 2 มิลลิวินาที ซึ่งจะ ได้มุมการกวาดของมอเตอร์ประมาณ 120 องศา

4.1.3 การทดสอบสัญญาณควบคุมที่สร้างโดยโปรแกรมบนไมโครคอนโทรลเลอร์

(1) เอ้าท์พุทที่ได้จากโปรแกรม POSITION.ASM สามารถปรับเปลี่ยนความกว้างของพัลส์ ได้โดยการปรับค่า R5 และปรับรูปการหน่วงเวลาซิกลอป เมื่อทดลองไคร่วมอเตอร์ด้วยค่าต่างๆพบว่าสามารถควบคุมตำแหน่งได้ สัญญาณที่ได้จากโปรแกรม POSITION.ASM เมื่อส่งค่า 00 เป็นค่า ควบคุม ทำให้ได้สัญญาณควบคุมดังแสดงในรูปที่ 4.3 ซึ่งมีคาบเวลา 18.57 มิลลิวินาที

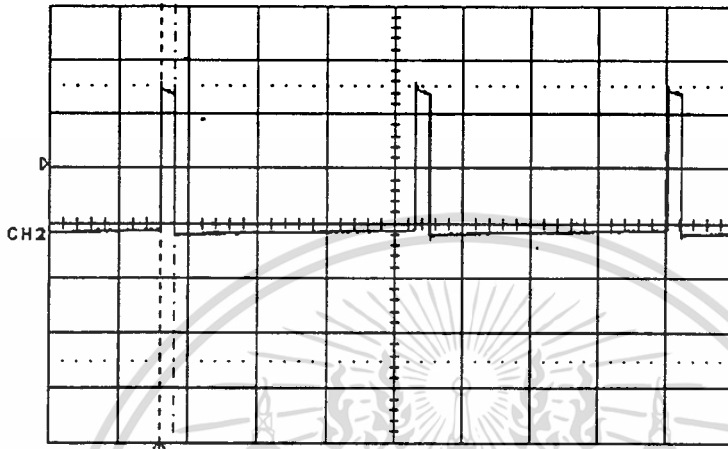
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

23-Mar-96
03:15:57

RUN ●
Auto

CH2
2V
5ms

CH1
2V
5ms



CH1 23 Mar, 03:15:43
AC, BUL: 20MHz
V@Center 0.00
t@Center 17.00ms

CH2 23 Mar, 03:15:43
AC, BUL: 20MHz
V@Center 0.00
t@Center 17.00ms

Smart Probe on CH1
2: RUN/STOP
CURSORS on CH2

TRIGGER on CH2
2.15V DC

--- 1.00ms
--- 0.0s
Δt 1.00ms
1/Δt 1.00kHz

CH1 MEASUREMENTS
pkpk 5.82V rms 1.261V
mean 13mV mean 13mV
per 18.570ms mean 13mV

รูปที่ 4.3 แสดงสัญญาณควบคุมที่ได้จากโปรแกรม POSITION.ASM

(2) เมื่อปรับปรุงโปรแกรม POSITION.ASM เป็นโปรแกรม DRIVER.ASM ให้ควบคุมมอเตอร์พร้อมๆกัน 8 ตัว (ใช้พอร์ต 2 ทั้ง 8 ขา) ดังกล่าวถึงในหัวข้อที่ 3.1.6 พบว่ามอเตอร์มีแรงบิดตกลง เนื่องมาจากการควบคุมสัญญาณควบคุมมอเตอร์ ใช้วิธีควบคุมทีละตัวเรียงไปจากตัวแรก P2.0 จนถึงตัวสุดท้ายที่ P2.7 ทำให้คาบเวลาของสัญญาณที่ควบคุมมอเตอร์แต่ละตัวมากขึ้นเป็น 8 เท่าของการควบคุมทีละตัว สรุปว่าโปรแกรม DRIVER.ASM ยังไม่สามารถใช้ในการควบคุมมอเตอร์พร้อมๆกันได้ในทางปฏิบัติ

ทางแก้ไขที่ได้คิดไว้มีอยู่สองแนวทางก็คือ การแก้ไขทางฮาร์ดแวร์ซึ่งหมายถึงการออกแบบวงจรพิเศษที่จะใช้ในการสร้างสัญญาณควบคุมโดยรับคำสั่งจากไมโครคอนโทรลเลอร์แล้วให้เอาต์พุตที่มีสัญญาณช็อบกวตามอินพุตจากไมโครคอนโทรลเลอร์ และสร้างสัญญาณเพิ่มเติมในช่วงที่ไมโครคอนโทรลเลอร์ไปควบคุมมอเตอร์ตัวอื่นๆ

อีกแนวทางหนึ่งคือการแก้ไขที่ซอฟต์แวร์ โดยหาลำดับการทำงานใหม่ให้มีประสิทธิภาพยิ่งขึ้น ซึ่งไม่ต้องใช้อุปกรณ์เพิ่มเติมพิเศษ และในที่สุดก็ได้เลือกวิธีการปรับปรุงทางซอฟต์แวร์นี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

(3) การพัฒนาโปรแกรม DRIVE.ASM ดังที่กล่าวถึงในหัวข้อ 3.1.7 ทำให้สามารถควบคุมมอเตอร์ได้พร้อมๆกัน 8 ตัวโดยที่กำจัดปัญหาที่เกิดขึ้นในโปรแกรม DRIVE.ASM เนื่องจากควบคุมสัญญาณทุกตัวพร้อมๆกัน แต่จากผลการทดลองพบว่า เมื่อมอเตอร์มีค่าควบคุมซ้ำกันหลายตัว ค่าสัญญาณควบคุมที่ได้จากพอร์ตไม่เท่ากัน ดังรูปที่ 4.4

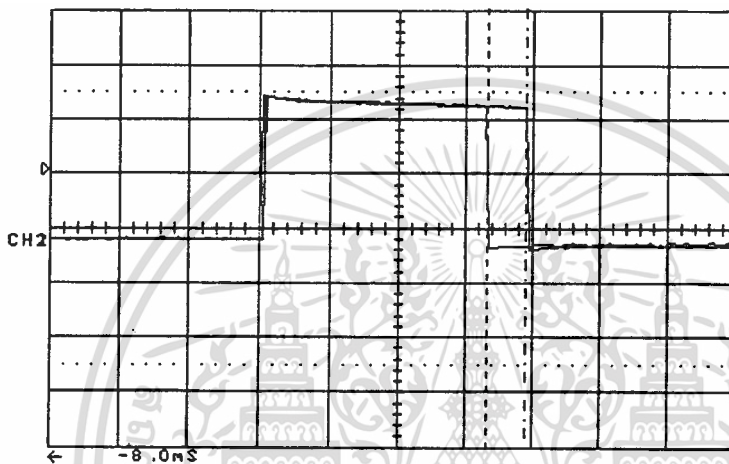
23-Mar-96 Recalibration Suggested (SYSTEM CONFIG Menu)
03:25:54

RUN ■

Auto

CH2←
2V
500ms

CH1←
2V
500ms



CH1 23 Mar, 03:25:40
AC, BUL: 20MHz
VCenter 0.0V
tCenter 20.35ms

CH2 23 Mar, 03:25:40
AC, BUL: 20MHz
VCenter 0.0V
tCenter 20.32ms

Smart Probe on CH1
2: RUN/STOP
CURSORS on CH1

TRIGGER on CH2
2.15V DC

CH1

-- 21.30ms
--- 21.03ms
Δt 300ms
1/Δt 3.6kHz

CH1 MEASUREMENTS			
pkpk	5.70V	rms	2.912V
mean	1.478V	mean	1.478V
per	--?--	mean	1.478V

รูปที่ 4.4 แสดงความผิดพลาดที่เกิดขึ้นในโปรแกรม DRIVE.ASM

จากการทดลอง เป็นการให้ค่าควบคุมเท่ากันในมอเตอร์ที่ลำดับติดกัน พบว่ามีผลต่างความกว้างสัญญาณบวกถึง 300 ไมโครวินาที และจะมากขึ้นเมื่อมีมอเตอร์ที่มีค่าควบคุมซ้ำกันหลายตัวมากขึ้น จึงเป็นปัญหาที่ต้องออกแบบกระบวนการสร้างสัญญาณควบคุมใหม่

(4) โปรแกรม NEWDRIVE.ASM ได้ถูกพัฒนาขึ้นเพื่อแก้ปัญหาของสัญญาณควบคุม โดยอาศัยการประมวลผลข้อมูลค่าควบคุมก่อน และเตรียมข้อมูลที่จะถูกส่งไปยังพอร์ตอีกครั้งหนึ่ง โดยไม่ได้อาศัยค่าควบคุมในการควบคุมการปิดสัญญาณควบคุมโดยตรงอีกต่อไป

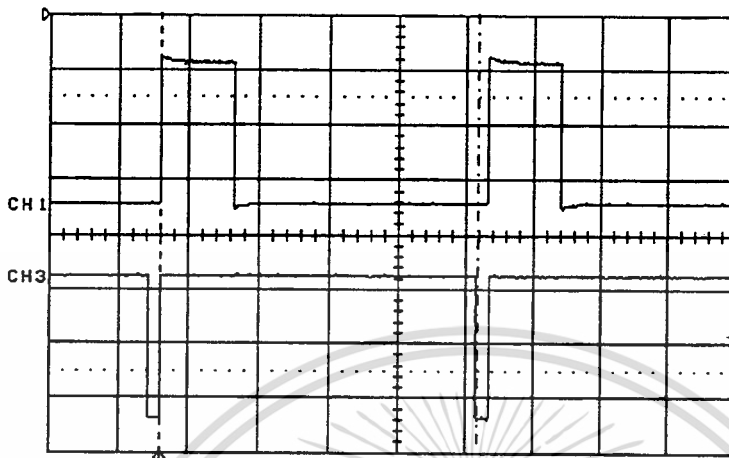
โปรแกรมในระยะแรก ยังไม่มีการหน่วงเวลาในซีกลบของสัญญาณเนื่องจากการดูความกว้างและสัญญาณที่ได้ ซึ่งได้ผลดังรูปที่ 4.5

25-Mar-96
06:27:47

RUN ■
Auto

CH3
2V
500ms

CH1
2V
500ms



CH1 25 Mar, 06:27:32
DC, BUL: Full
V@Center -1.22V
t@Center 1.700ms
CH3 25 Mar, 06:27:32
DC, BUL: Full
V@Center 6.66V
t@Center 1.700ms

Smart Probe on CH1
2: RUN/STOP

TRIGGER on CH1

CURSORS on CH1

-- 2.288ms

--- 0.0s

Δt 2.288ms

1/Δt 437.2Hz

pkpk

mean

freq

6.80V DC

CH1 MEASUREMENTS

5.63V rms 2.336V

1.011V min -320mV

421.5Hz per 2.3725ms

รูปที่ 4.5 แสดงสัญญาณควบคุมที่ได้จากโปรแกรม NEWDRIVE.ASM เมื่อยังไม่มีการหน่วงสัญญาณ '0'

เมื่อให้ค่าควบคุมเป็น '00' จะได้ความกว้างสัญญาณบวกเป็น 525 ไมโครวินาที และเมื่อให้ค่าควบคุมเป็น 'FF' จะได้ความกว้างสัญญาณบวกเป็น 2.3725 มิลลิวินาที ซึ่งเป็นสัญญาณบวกที่สามารถใช้ในการควบคุมตำแหน่งของเซอร์โวได้

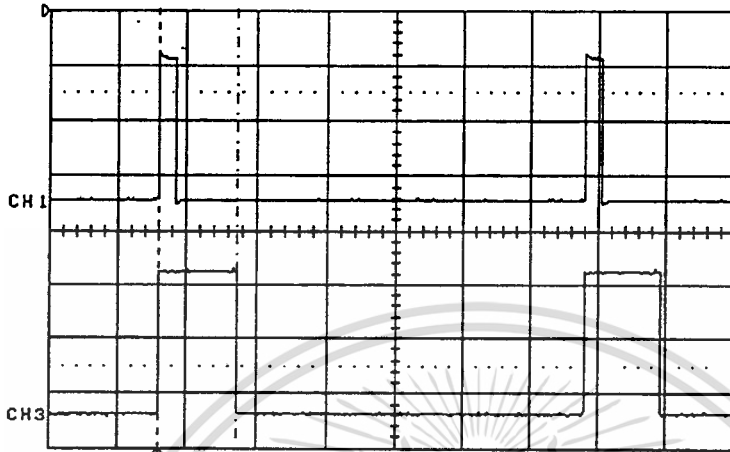
จากนั้นได้ทดลองใส่ลูปหน่วงเวลาซิกลอป ซึ่งจะได้สัญญาณดังรูป 4.6 จะเห็นว่าสัญญาณที่ได้มีคาบเพียง 12.40 มิลลิวินาที เพราะจากการทดลองพบว่า ความกว้างของสัญญาณที่เป็นศูนย์ไม่มีผลต่อตำแหน่งและแรงบิดของดีซีเซอร์โวมอเตอร์ จึงไม่จำเป็นต้องคำนวณให้สัญญาณที่เป็นศูนย์มีความกว้างที่แม่นยำที่ค่าใดค่าหนึ่ง และจากจุดนี้เอง ที่ทำให้โปรแกรม NEWDRIVE.ASM ที่สมบูรณ์จะตัดลูปหน่วงเวลาสัญญาณศูนย์โวลต์ทิ้ง และจะใช้เวลาในการทำคำสั่งของโมดูล SORT และ PREPARE เป็นช่วงที่มีสัญญาณเป็นศูนย์

25-Mar-96
06:29:34

RUN ■
Auto

CH3
2V
2ms

CH1
2V
2ms



CH1 25 Mar, 06:29:20
DC, BUL: Full
V@Center -1.22V
t@Center 6.80ms

CH3 25 Mar, 06:29:20
DC, BUL: Full
V@Center 6.66V
t@Center 6.80ms

Smart Probe on CH1
2: RUN/STOP
CURSORS on CH1

TRIGGER on CH1
6.80V DC

-- 2.25ms
--- 0.0s
Δt 2.25ms
1/Δt 444Hz

CH1 MEASUREMENTS			
pkpk	5.50V	rms	1.171V
mean	198mV	min	-256mV
freq	80.65Hz	per	12.400ms

รูปที่ 4.6 สัญญาณจาก NEWDRIVE.ASM เมื่อมีการหน่วงเวลาสัญญาณศูนย์โวลต์

(5) ผลจากโปรแกรม NEWDRIVE.ASM ที่สมบูรณ์ ที่จะทำงานโมดูล SORT และ PREPARE ในช่วงสัญญาณศูนย์ ซึ่งจะให้ความกว้างของช่วงนี้ไม่แน่นอน ขึ้นอยู่กับว่าข้อมูลที่เข้ามาทำการเรียงลำดับใน SORT เป็นอย่างไร ถ้าเป็นกรณีที่ดีที่สุด (Best Case) คือค่าควบคุมเรียงตามลำดับจากน้อยไปมากอยู่แล้ว ก็จะได้คาบสัญญาณรวมประมาณ 7 มิลลิวินาทีที่แสดงในรูปที่ 4.7 ส่วนกรณีที่แย่ที่สุด (Worst Case) คือค่าควบคุมเรียงลำดับจากมากไปหาน้อย ก็จะได้คาบสัญญาณรวมเป็น 9.67 มิลลิวินาทีที่แสดงรูปที่ 4.8

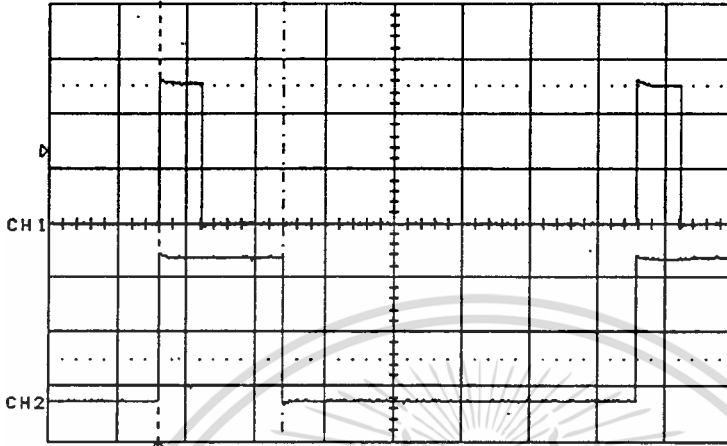
จากการวิเคราะห์โมดูลการเรียงลำดับ โดยการแยกมารันเป็นโปรแกรม พบว่าเวลาในการเรียงลำดับข้อมูลกรณีที่ดีที่สุดจะอยู่ที่ 5.2 มิลลิวินาที และกรณีที่แย่ที่สุดจะใช้เวลา 7.4 มิลลิวินาที ส่วนโมดูลการเตรียมค่าข้อมูลจากค่าควบคุม จะใช้เวลาทุกกรณี ประมาณ 750 ไมโครวินาที

27-Mar-96
02:59:20

RUN ■
Auto

CH1
2V
1ms

CH2
2V
1ms



CH1 27 Mar, 02:59:06
DC, BUL: Full
V@Center 0.0V
t@Center 3.400ms

CH2 27 Mar, 02:59:06
DC, BUL: Full
V@Center 6.53V
t@Center 3.400ms

Smart Probe on CH1
2: RUN/STOP

TRIGGER on CH1

Cursors on CH1

--- 1.800ms

--- 0.0s

Δt 1.800ms

1/Δt 556Hz

2.70V DC

CH1 MEASUREMENTS

pkpk	5.63V	rms	1.830V
mean	608mV	min	-256mV
freq	143.1Hz	per	6.9900ms

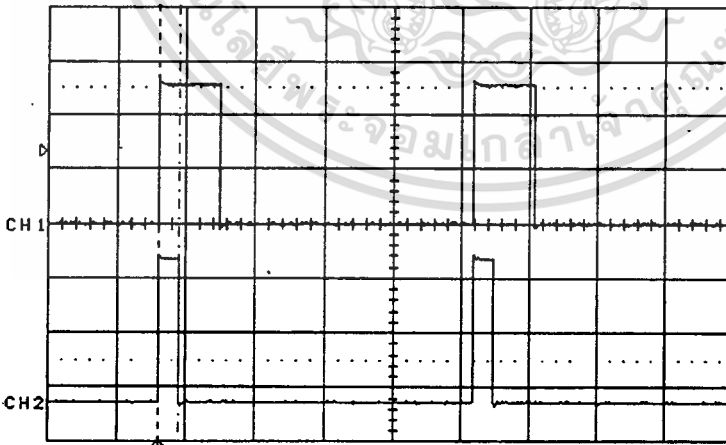
รูปที่ 4.7 แสดงสัญญาณเมื่อค่าควบคุมเรียงลำดับเป็นกรณีที่ดีที่สุด

27-Mar-96
03:46:49

RUN ■
Auto

CH1
2V
2ms

CH2
2V
2ms



CH1 27 Mar, 03:46:35
DC, BUL: Full
V@Center 0.0V
t@Center 6.80ms

CH2 27 Mar, 03:46:35
DC, BUL: Full
V@Center 6.53V
t@Center 6.80ms

Smart Probe on CH1
2: RUN/STOP

TRIGGER on CH1

Cursors on CH1

--- 600μs

--- 0.0s

Δt 600μs

1/Δt 1.67kHz

2.70V DC

CH1 MEASUREMENTS

pkpk	5.50V	rms	2.170V
mean	896mV	min	-256mV
freq	109.3Hz	per	9.150ms

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
รูปที่ 4.8 แสดงสัญญาณเมื่อค่าควบคุมเรียงลำดับเป็นกรณีที่แย่ที่สุด
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้คัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2 การทดลองทางด้านการคอมพิวเตอร์

การทดลองทางด้านการคอมพิวเตอร์ จะเป็นการทดสอบสัญญาณเอาต์พุตที่ออกมาจาก RS-232 ให้ได้สัญญาณที่ถูกต้อง เพื่อนำไปสื่อสารกับไมโครคอนโทรลเลอร์ Z8 อีกที การสื่อสารด้วยมาตรฐาน RS-232 ระหว่าง Z8 กับ ไมโครคอมพิวเตอร์จะเชื่อมต่อด้วยมาตรฐาน 3-Wired Economic Mode ซึ่งหมายความว่า ที่ตัวพอร์ตอนุกรมจะต้องทำการป้อนกลับ (Self-Feedback) ด้วยตัวเอง หรืออาจจะยกเลิกการทำแฮนด์เชกकि่ง ที่ตัวซอฟต์แวร์เลขก็ได้

4.2.1 การส่งข้อมูลระหว่างคอมพิวเตอร์กับไมโครคอนโทรลเลอร์

รูปแบบการส่งข้อมูลที่กำหนดไว้ในสำหรับการสื่อสาร ก็คือมีข้อมูลชุดละ 2 ไบต์ด้วยกัน โดยข้อมูลทั้ง 16 บิต จะเป็นเนื้อที่สำหรับส่งข้อมูลเกี่ยวกับรหัส (ID) ของตัวเซอร์โว และตำแหน่งของเซอร์โวที่ต้องการ โดยกำหนดให้รหัสมีขนาด 4 บิต ซึ่งทำให้สามารถระบุถึงตัวเซอร์โวได้ทั้งหมด 16 ตัวด้วยกัน กำหนดให้ตัวข้อมูล มีขนาด 8 บิต ซึ่งทำให้ตัวเซอร์โวสามารถมีตำแหน่งได้ทั้งหมด 256 ตำแหน่งในช่วงกวาดทั้งหมด ซึ่งที่ความละเอียดนี้เพียงพอต่อการใช้งาน รูปแบบของข้อมูลที่ทำกรสื่อสารกับตัว Z8 มีรูปแบบและตำแหน่งดังภาพ

โครงสร้างของไบต์แรก

0	X	X	A3	A2	A1	A0	D7
---	---	---	----	----	----	----	----

โครงสร้างของไบต์ที่สอง

1	D6	D5	D4	D3	D2	D1	D0
---	----	----	----	----	----	----	----

- เมื่อ A0-A3 เป็นรหัสของเซอร์โวมอเตอร์
D0-D7 เป็นตำแหน่งมุมของเซอร์โวมอเตอร์
X หมายถึง ไม่สนใจบิตนั้น
0,1 เป็นรหัสที่แยกความแตกต่างระหว่างไบต์แรกและไบต์ที่สอง

รูปที่ 4.9 แสดงไคอะแกรมของข้อมูลขนาด 2 ไบต์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2.2 การทดสอบการสื่อสาร

เนื่องจากเอ้าท์พุท นี้จะถูกนำไปใช้งานโดยตรงโดย ZX ดังนั้นเราสามารถ ทดสอบค่าที่โปรแกรมหลักส่งออกมาได้หลายวิธีด้วยกัน ไม่ว่าจะเป็นการตรวจจับจากออสซิลโลสโคป หรือจะเป็นการเขียนโปรแกรมเพื่ออ่านค่าเข้าทางพอร์ตอนุกรม และตรวจสอบค่าที่อ่านได้ ซึ่งในที่นี้เราจะใช้วิธีการเขียนโปรแกรมที่เครื่องคอมพิวเตอร์อีกเครื่องหนึ่ง และสื่อสารกับตัวเครื่องคอมพิวเตอร์หลัก



รูปที่ 4.10 แสดงการวางคอมพิวเตอร์เพื่อทดสอบ

4.2.3 การเขียนโปรแกรมเพื่อทดสอบ

โปรแกรมที่ใช้ในการทดสอบในช่วงแรกจะใช้ โปรแกรมสื่อสารประเภทเช่น โปรแกรม Terminal ของวินโดวส์ หรือโปรแกรม Procomm เพื่อใช้ในการตรวจสอบการทำงานของโปรแกรมว่าสามารถรับส่งข้อมูลได้ถูกต้องหรือไม่ ทั้งขนาดและตัวข้อมูล เมื่อตรวจสอบได้แล้วว่าโปรแกรมที่ใช้ทดสอบมีความน่าเชื่อถือพอที่จะใช้อ้างอิงได้จึงนำโปรแกรมนั้นมาทดสอบกับเครื่องควบคุมหุ่นอีกที

สำหรับอัตราเร็วที่ใช้ตั้งค่าใน โปรแกรมสื่อสารก็คือ อัตราการส่งข้อมูลที่ 9600 บิตต่อวินาที และมีข้อมูลขนาด 8 บิต ไม่มีการตรวจสอบพาริตี และมี 1 สตอปบิต ไม่มีการตรวจสอบสัญญาณแฮนด์เชกกิ้ง ซึ่งก็คือ 9600,n,8,1 นั่นเอง โปรแกรมที่จะนำมาทดสอบจะเขียนด้วยภาษา BASIC เพื่อความสะดวกในการทำงาน เนื่องจากเป็นภาษาที่ง่ายต่อการเขียนโปรแกรมขนาดสั้นๆ โปรแกรมสำหรับทดสอบที่เขียนขึ้นก็จะเป็นดังนี้

```
cls
open "com1:9600,n,8,1,ds,bin" for input as #1
while(1)
input #1,a$
print len(a$),asc(left$(a$,1)),asc(right$(a$,1))
wend
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรมนี้จะแสดงขนาดของข้อมูลที่เข้ามาที่ขณะใดๆ และข้อมูลแต่ละตัวที่เข้ามา ในช่วงแรกๆ โปรแกรมควบคุมหุ่นยนต์มีปัญหาในการสื่อสารมาก เนื่องจากข้อมูลที่สั่งให้ส่งออกมาไม่ยอมปรากฏอยู่ที่พอร์ตอนุกรม เนื่องจากค้างอยู่ในบัฟเฟอร์ซึ่งยังไม่สามารถตรวจสอบได้ว่า ค้างอยู่ในบัฟเฟอร์ ส่วนใดระหว่างบัฟเฟอร์ของระบบปฏิบัติการ หรือบัฟเฟอร์ของชิป UART (ในโครงงานนี้ใช้เบอร์ 16550AF) อย่างไรก็ตาม การปรับพารามิเตอร์ของพอร์ตการสื่อสารใหม่ ก็ทำให้การสื่อสารเป็นไปได้ในที่สุด โดยเริ่มมีข้อมูลออกมาที่พอร์ตอนุกรม พารามิเตอร์ที่ปรับใหม่ก็คือให้เซนต์เทรสโฮลด์มีค่าเป็น 1 หรือ มากกว่า



รูปที่ 4.11 แสดงผลลัพธ์ที่ได้จากการทดสอบ

หลังจากแก้ปัญหาแรกไปได้แล้ว ปัญหาต่อมาก็คือ ข้อมูลที่ส่งออกมา มีลักษณะเป็น รหัส ASCII นั่นคือ ทุกข้อมูลที่ส่งออกมาจะมีอักขรพิเศษ Carriage Return และ Line Feed ผสมออกมาด้วยทุกครั้ง ซึ่งในภายหลังได้เปลี่ยนรูปแบบการสื่อสารให้เป็นโหมด Binary และปัญหานี้ก็หมดไป ต่อมาปัญหาสุดท้ายก็คือ เมื่อทำการพล็อตข้อมูลทั้งหมดออกจากบัฟเฟอร์ แล้วข้อมูลตัวสุดท้ายจะเสียไป กลายเป็นอักขรแปลกๆ ไม่ว่าจะเปลี่ยนขนาดของชุดข้อมูลเป็นเท่าไรก็ตาม นอกจากนี้ยังมีปัญหาข้อมูลทับกัน (Overrun) ที่บัฟเฟอร์ของ UART อีก จากการทดสอบหลายๆครั้ง ก็สรุปได้ว่าเป็นปัญหาภายในออปเจกต์สื่อสารเอง ดังนั้นวิธีแก้ปัญหาคือที่ที่สุดก็คือ เปลี่ยนวิธีการสื่อสาร

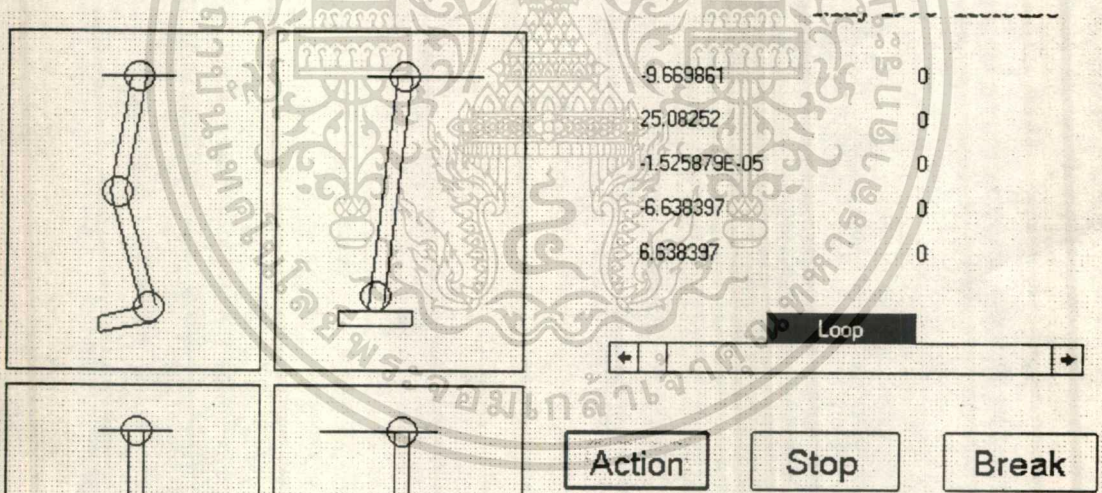
จากการใช้ออปเจกต์สื่อสาร ไปเป็นการสื่อสารผ่านภาษามาตรฐานของ BASIC เองคือใช้รูปแบบภาษาเป็น

```
open "com2" for output as#1
print#1,chr$(data1)+chr$(data2)
close#1
```

4.2.4 การทดสอบโปรแกรมทำการเคลื่อนไหว

ทางด้านการใช้งานโปรแกรมหลัก เมื่อมีลำดับท่าทางที่ต้องการแล้ว ไม่ว่าจะเป็นจากการจัดเอง หรือจะเรียกจากไฟล์ที่ทำไว้ก่อนหน้านั้นก็ตาม จะมีทางเลือก ที่จะส่งค่าออกพอร์ตอยู่ 2 ทางด้วยกัน ก็คือการกดปุ่ม 'Action' เพื่อให้แสดงตามลำดับท่าทาง โปรแกรมจะส่งข้อมูลออกไปทั้งหมดเป็นจำนวน

จำนวนเฟรม * จำนวนเซอร์โว * 2 ไบต์



รูปที่ 4.12 การส่งท่าทางเคลื่อนไหว

อีกวิธีที่สามารถทำได้ ก็คือ การส่งไปทีละเฟรม โดยการกดปุ่ม 'Send Frame' โปรแกรมจะส่งข้อมูลทั้งหมดที่มีของเฟรมนั้นไปยังพอร์ตอนุกรม ซึ่งมีประโยชน์ในการแก้ไขผิดพลาด (Debug) และตรวจสอบในแต่ละท่าทางว่ามีลักษณะตามที่ต้องการหรือไม่ และจำนวนข้อมูลที่ส่งออกไปทั้งหมดจะเท่ากับ

จำนวนเซอร์โว * 2 ไบต์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5

สรุปและวิเคราะห์ผลการทดลอง

5.1 สรุปผลการดำเนินงาน

การศึกษาและทดลองเท่าที่ผ่านมา แบ่งออกได้เป็นสองส่วนหลักๆคือ ส่วนของไมโครคอมพิวเตอร์ และส่วนของไมโครคอนโทรลเลอร์ ซึ่งได้มีการพัฒนาขึ้นเป็นลำดับ ด้านไมโครคอนโทรลเลอร์ได้ทดลองถึงส่วนของการควบคุมมอเตอร์อย่างสมบูรณ์โดยยังไม่ได้รวมส่วนของการรับข้อมูลจากไมโครคอมพิวเตอร์ซึ่งกำลังพัฒนาอยู่ ในส่วนของไมโครคอมพิวเตอร์ ได้เขียนโปรแกรมติดต่อกับผู้ใช้แบบกราฟิกเพื่อรับคำสั่งในการเคลื่อนไหวก โดยอาศัยการควบคุมแบบแมนนวล ซึ่งผู้ใช้งานจะจัดลำดับท่าทาง และสามารถบันทึกท่าทางที่ต้องการ เพื่อส่งสัญญาณไปยังไมโครคอนโทรลเลอร์ให้ควบคุมมอเตอร์ได้

5.2 ปัญหาที่เกิดขึ้นและแนวทางการแก้ไข

ปัญหาหลักในการดำเนินงานที่เกิดขึ้น พอจะแจกแจงได้ดังต่อไปนี้

5.2.1 ความเร็วในการทำงานของไมโครคอนโทรลเลอร์ อาจไม่เพียงพอกับข้อมูลที่ต้องติดต่อกับไมโครคอมพิวเตอร์ และการควบคุมทิศทางมอเตอร์ แก้ไขในเบื้องต้นโดยการเปลี่ยนตัวกำเนิดสัญญาณป้อนแกไมโครคอนโทรลเลอร์จาก 8 เมกะเฮิร์ตซ์เป็น 12 เมกะเฮิร์ตซ์ แต่คาดว่าเมื่อมีการพัฒนาหุ่นยนต์นี้ให้สมบูรณ์ขึ้น จะต้องมีการผ่านข้อมูลมากกว่านี้มาก ความเร็วของไมโครคอนโทรลเลอร์อาจไม่เพียงพอ ซึ่งอาจต้องแก้ไขโดยการเชื่อมต่อหุ่นยนต์เข้ากับคอมพิวเตอร์ความเร็วสูงโดยตรง ผ่านพอร์ตสื่อสารที่ออกแบบเฉพาะ

5.2.2 ปัญหาข้อจำกัดของชุดคำสั่งของ Z8 ที่มีน้อยและจำกัด ต้องอาศัยชุดคำสั่งมากในการปฏิบัติงานที่ซับซ้อนขึ้น ประกอบกับความเร็วในการทำงานที่อาจไม่เพียงพออยู่แล้ว ทำให้การพัฒนาซอฟต์แวร์บนไมโครคอนโทรลเลอร์ต้องมีการปรับแต่งให้มีประสิทธิภาพสูงที่สุด

5.2.3 ไมโครคอนโทรลเลอร์ Z8 ที่เลือกมาใช้งาน ไม่มีความสามารถในการรับส่งข้อมูลแบบอนุกรม ทำให้ต้องพัฒนาส่วนการรับ-ส่งข้อมูลขึ้นเอง ซึ่งต้องใช้คำสั่งในการปฏิบัติงานมากขึ้น และกินเวลาการปฏิบัติงานส่วนอื่นๆของไมโครคอนโทรลเลอร์

5.2.4 ระบบส่งข้อมูลผ่านทาง UART ยังได้ผลไม่เป็นที่น่าพอใจเนื่องจากข้อมูลยังมีการทับกันของข้อมูลบ้างในบางโอกาส ซึ่งเป็นข้อผิดพลาดที่เกิดจากระบบคอมพิวเตอร์ และจำเป็นต้องได้รับการตรวจสอบและเช็คข้อผิดพลาดต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.2.5 โปรแกรมที่ทำงานบนไมโครซอฟต์วินโดวส์ ยังมีความเร็วไม่เป็นที่น่าพอใจ และการทำงานพร้อมกันหลายงาน (Multitasking) ก็ยังไม่ดีนัก ทำให้บางครั้งการกระจายงานไม่ทั่วถึงทางแกก็คือหาระบบปฏิบัติการและภาษาใหม่ที่มีประสิทธิภาพดีขึ้น

5.3 แนวทางการพัฒนาในอนาคต

การพัฒนาที่ได้ทำขึ้นมา เป็นพื้นฐานและจุดเริ่มต้นก้าวหนึ่งในการศึกษาและวิจัยเกี่ยวกับหุ่นยนต์เคลื่อนที่สองขา ซึ่งสามารถพัฒนาเพิ่มเติมได้หลายๆส่วน เช่นพัฒนาระบบควบคุมอัตโนมัติเพื่อให้หุ่นยนต์สามารถทรงตัวและเคลื่อนที่สองขาได้โดยสามารถทนต่อการรบกวนจากสภาพแวดล้อมภายนอก เช่นแรงที่มากกระทำทำให้หุ่นยนต์เกิดการเปลี่ยนแปลงตำแหน่งหรือล้มลง ซึ่งต้องอาศัยการศึกษาลึกลงไปถึงเรื่องการวิเคราะห์เสถียรภาพในการทรงตัวของหุ่นยนต์ และการสร้างโมเดลคณิตศาสตร์เพื่อช่วยในการควบคุมและการจำลองระบบให้อยู่ในรูปคณิตศาสตร์ เพื่อใช้ในการคำนวณหาเสถียรภาพต่อไป

ส่วนของการติดต่อกับผู้ใช้ก็สามารถพัฒนาให้ดีขึ้นได้ เนื่องจากส่วนติดต่อกับผู้ใช้ที่ได้พัฒนาขึ้นมาได้ ได้ออกแบบเพื่อทำหน้าที่พื้นฐานเพื่อการทดสอบและพัฒนาเท่านั้น ซึ่งสามารถพัฒนาได้เช่น การทำอินเตอร์เฟซสามมิติ หรือการเพิ่มส่วนการวิเคราะห์เสถียรภาพจากท่าทางของหุ่นยนต์ที่จำลองขึ้นในไมโครคอมพิวเตอร์

สำหรับโปรแกรมที่ได้รับการปรับปรุงข้อผิดพลาดล่าสุด สามารถดาวน์โหลดได้ผ่านอินเทอร์เน็ต URL <http://www.kmitl.ac.th/~s5104322>



ภาคผนวก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

;*****
; SERIAL.ASM
; TEST RECEIVING SERIAL-IN DATA FROM COMPUTER AT 9,600 BPS
;*****
.ORG 0H
.BYTE %01, %00 ;Interrupt Vector of IRQ0

.ORG 0CH

LD IPR, %14 ;SET INTERRUPT PRIORITY REGISTER

LD P2M, #0 ;SET PORT 2 AS OUTPUT PORT
LD P3M, #1 ;SET PULL UP

LD P01M, %04
LD IMR, %81 ;ENABLE INTERRUPTS AND IRQ0
EI

LOOP: ;Main Program is just generate square wave on P2.0

XOR P2, #1
JP LOOP

.ORG 0100H
; IRQ 0 Interrupt Service Routine
; To Service Serial Input Interrupt Via P3.2
; Input: from RS-232 START_BIT-D0-D1-D2-D3-D4-D5-D6-D7-STOP_BIT
; at 9600 bps
; This routine will collect data from P3.2 and rearrange to
; D7-D6-D5-D4-D3-D2-D1-D0
; *****
DI

PUSH R4
PUSH R5
PUSH R6
PUSH R7

; ***** DELAY BEFORE GET FIRST BIT
LD R4, #3
OUTTER_LOOP:
LD R5, #20
INNER_LOOP:
DJNZ R5, INNER_LOOP
DJNZ R4, OUTTER_LOOP
; ***** WE ARE IN THE MIDDLE OF FIRST BIT AFTER START BIT, LET'S START
; ***** SAMPLING!!!!
LD R15, #0 ; CLEAR ACCUMULATER REGISTER R15
LD R7, #1 ; R7 COUNT NUMBER OF FINISHED BIT
; BE INCREMENTED EVERY LOOP

GET_DATA:
LD R6, P3 ; R6 IS THE BUFFER OF PORT 3.2
LD R7, P2, #128
LD P2, #0
AND R6, #4

LD R4, #2 ; R4 ARE NOT IN USE, USE IT FOR LOOPING
MOVE_TO_DO_POSITION:
RR R6
DJNZ R4, MOVE_TO_DO_POSITION

PUSH R7 ; WE GONNA ROTATE INCOMING BIT IN R6
ROTATE: ; TO THE PROPER POSITION BY THE
RL R6 ; VALUE TO ROTATE IN R7
DJNZ R7, ROTATE

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

RR R6 ; COMPENSATE RIGHT 1 POSITION
POP R7

OR R15,R6 ; MIX TOGETHER

;DELAY
LD R4,#2
OUT_LOOP: LD R5,#13 ; AFTER COMPENSATE WITH THE TIME SPENT
; WITH ABOUT 10 INSTRUCTIONS
; WE REDUCE R5 TO 20
IN_LOOP: DJNZ R5,IN_LOOP
DJNZ R4,OUT_LOOP

INC R7
CP R7, #9
JP Z, FINISH_8_BIT ;IF FINISHED 8 BITS THEN JUMP OUT OF LOOP
JP GET_DATA ;ELSE LOOP TO GET NEXT BIT

FINISH_8_BIT: POP R7
POP R6
POP R5
POP R4
LD IRQ,#0 ;CLEAR INTERRUPT REQUEST BEFORE
;ENABLE AGAIN (PREVENT RE-INTERUPT)

EI
IRET

.END

```



```

; *****
; newdrive.ASM
; New algorithm TO PRODUCE CONTROL pulse
; to control 10 motors simultaneously
; *****
.ORG 0CH

ld    imr, #%0
ld    spl, #%90

LD    P2M, #0      ;SET PORT 2 AS OUTPUT PORT
LD    P3M, #1      ;SET PULL UP
LD    P01M, #%04

LOOP:
CALL  SORT
CALL  PREPARE
CALL  CONTROL
JP    LOOP

CONTROL: ; *****PROCEDURE*****
;GENERATE CONTROL PULSE OF 10 MOTORS SIMULTANEOUSLY
;READ POSITION FROM TABLE stored at %40, %50
;use register bank 1
;R11 is the next value to close control pulse

LD    P2, #%FF      ;SET ALL PORTS TO '1'
ld    p0, %ff
ld    rp, %20

LD    R12, #0        ;R12 IS USED FOR COUNTER
ld    r10, #0        ;times of sending to port

LD    R14, %E6       ;'E6' IS THE VALUE THAT MAKE LOOP = 0.5 MS
HIGH: DJNZ R14, HIGH ;0.5 ms

;*****
LD    R11, R0
ADD   R11, %30        ;ADDRESS 30h-39h IS TABLE OF MOTORS
LD    R11, @R11       ;GET MOTOR VALUE TO R11

INC_R121:
CP    R11, R12        ;WHEN VALUE IN R12 IS INCed AND REACH R11
JP    Z, CLOSE_MOTOR1
INC   R12
JP    INC_R121

CLOSE_MOTOR1:
ld    r13, %50        ;mask bytes before sending to p0
ld    r14, r13        ;r14 to tell show many steps to jump
and   r13, %3
and   r14, %33c       ;only d1,d0 are to send
ld    r15, #2         ;only d5,d4,d3,d2 are index to jump
;loop 2 times
rorl: rr    r14
djnz  r15, rorl
ld    p2, %40        ;send byte to p2
ld    p0, r13        ;send byte (2 bit) to p0

add   r10, r14        ;next value to stop pulse
ld    r11, r10
add   r11, %20
ld    r11, @r11
add   r11, %30
ld    r11, @r11

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

INC_R122:
  CP      R11,R12      ;WHEN VALUE IN R12 IS INCed AND REACH R11
  JP      Z, CLOSE_MOTOR2
  inc     r12
  JP      INC_R122

```

```

CLOSE_MOTOR2:
  ld      r13,%51      ;mask bytes before sending to p0
  ld      r14,r13      ;r14 to tell show many steps to jump
  and     r13,%3       ;only d1,d0 are to send
  and     r14,%3c      ;only d5,d4,d3,d2 are index to jump
  ld      r15,#2       ;loop 2 times
ror2:  rr      r14
  djnz    r15,ror2
  ld      p2,%41      ;send byte to p2
  ld      p0,r13      ;send byte (2 bit) to p0

  add     r10,r14      ;next value to stop pulse
  ld      r11,r10
  add     r11,%20
  ld      r11,@r11
  add     r11,%30
  ld      r11,@r11

```

```

INC_R123:
  CP      R11,R12      ;WHEN VALUE IN R12 IS INCed AND REACH R11
  JP      Z, CLOSE_MOTOR3
  inc     r12
  JP      INC_R123

```

```

CLOSE_MOTOR3:
  ld      r13,%52      ;mask bytes before sending to p0
  ld      r14,r13      ;r14 to tell show many steps to jump
  and     r13,%3       ;only d1,d0 are to send
  and     r14,%3c      ;only d5,d4,d3,d2 are index to jump
  ld      r15,#2       ;loop 2 times
ror3:  rr      r14
  djnz    r15,ror3
  ld      p2,%42      ;send byte to p2
  ld      p0,r13      ;send byte (2 bit) to p0

  add     r10,r14      ;next value to stop pulse
  ld      r11,r10
  add     r11,%20
  ld      r11,@r11
  add     r11,%30
  ld      r11,@r11

```

```

INC_R124:
  CP      R11,R12      ;WHEN VALUE IN R12 IS INCed AND REACH R11
  JP      Z, CLOSE_MOTOR4
  inc     r12
  JP      INC_R124

```

```

CLOSE_MOTOR4:
  ld      r13,%53      ;mask bytes before sending to p0
  ld      r14,r13      ;r14 to tell show many steps to jump
  and     r13,%3       ;only d1,d0 are to send
  and     r14,%3c      ;only d5,d4,d3,d2 are index to jump
  ld      r15,#2       ;loop 2 times
ror4:  rr      r14
  djnz    r15,ror4
  ld      p2,%43      ;send byte to p2
  ld      p0,r13      ;send byte (2 bit) to p0

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

add    r10,r14          ;next value to stop pulse
ld     r11,r10
add    r11,#%20
ld     r11,@r11
add    r11,#%30
ld     r11,@r11

```

```

INC_R125:
CP      R11,R12          ;WHEN VALUE IN R12 IS INCed AND REACH R11
JP      Z, CLOSE_MOTOR5
inc     r12
JP      INC_R125

```

```

CLOSE_MOTOR5:
ld      r13,%54          ;mask bytes before sending to p0
ld      r14,r13          ;r14 to tell show many steps to jump
and     r13,#%3          ;only d1,d0 are to send
and     r14,#%3c         ;only d5,d4,d3,d2 are index to jump
ld      r15,#2           ;loop 2 times
ror5:   rr      r14
djnz    r15,ror5
ld      p2,%44           ;send byte to p2
ld      p0,r13          ;send byte (2 bit) to p0

add     r10,r14          ;next value to stop pulse
ld     r11,r10
add     r11,#%20
ld     r11,@r11
add     r11,#%30
ld     r11,@r11

```

```

INC_R126:
CP      R11,R12          ;WHEN VALUE IN R12 IS INCed AND REACH R11
JP      Z, CLOSE_MOTOR6
inc     r12
JP      INC_R126

```

```

CLOSE_MOTOR6:
ld      r13,%55          ;mask bytes before sending to p0
ld      r14,r13          ;r14 to tell show many steps to jump
and     r13,#%3          ;only d1,d0 are to send
and     r14,#%3c         ;only d5,d4,d3,d2 are index to jump
ld      r15,#2           ;loop 2 times
ror6:   rr      r14
djnz    r15,ror6
ld      p2,%45           ;send byte to p2
ld      p0,r13          ;send byte (2 bit) to p0

add     r10,r14          ;next value to stop pulse
ld     r11,r10
add     r11,#%20
ld     r11,@r11
add     r11,#%30
ld     r11,@r11

```

```

INC_R127:
CP      R11,R12          ;WHEN VALUE IN R12 IS INCed AND REACH R11
JP      Z, CLOSE_MOTOR7
inc     r12
JP      INC_R127

```

```

CLOSE_MOTOR7:
ld      r13,%56          ;mask bytes before sending to p0
ld      r14,r13          ;r14 to tell show many steps to jump
and     r13,#%3          ;only d1,d0 are to send

```

เอกสารนี้เป็นเอกสารสงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

        and    r14,#%3c          ;only d5,d4,d3,d2 are index to jump
        ld     r15,#2           ;loop 2 times
ror7:   rr     r14
        djnz   r15,ror7
        ld     p2,%46           ;send byte to p2
        ld     p0,r13           ;send byte (2 bit) to p0

        add    r10,r14          ;next value to stop pulse
        ld     r11,r10
        add    r11,#%20
        ld     r11,@r11
        add    r11,#%30
        ld     r11,@r11

INC_R128:
        CP     R11,R12          ;WHEN VALUE IN R12 IS INCed AND REACH R11
        JP     Z, CLOSE_MOTOR8
        inc    r12
        JP     INC_R128

CLOSE_MOTOR8:
        ld     r13,%57          ;mask bytes before sending to p0
        ld     r14,r13          ;r14 to tell show many steps to jump
        and    r13,#%3          ;only d1,d0 are to send
        and    r14,#%3c        ;only d5,d4,d3,d2 are index to jump
        ld     r15,#2          ;loop 2 times
ror8:   rr     r14
        djnz   r15,ror8
        ld     p2,%47          ;send byte to p2
        ld     p0,r13          ;send byte (2 bit) to p0

        add    r10,r14          ;next value to stop pulse
        ld     r11,r10
        add    r11,#%20
        ld     r11,@r11
        add    r11,#%30
        ld     r11,@r11

INC_R129:
        CP     R11,R12          ;WHEN VALUE IN R12 IS INCed AND REACH R11
        JP     Z, CLOSE_MOTOR9
        inc    r12
        JP     INC_R129

CLOSE_MOTOR9:
        ld     r13,%58          ;mask bytes before sending to p0
        ld     r14,r13          ;r14 to tell show many steps to jump
        and    r13,#%3          ;only d1,d0 are to send
        and    r14,#%3c        ;only d5,d4,d3,d2 are index to jump
        ld     r15,#2          ;loop 2 times
ror9:   rr     r14
        djnz   r15,ror9
        ld     p2,%48          ;send byte to p2
        ld     p0,r13          ;send byte (2 bit) to p0

        add    r10,r14          ;next value to stop pulse
        ld     r11,r10
        add    r11,#%20
        ld     r11,@r11
        add    r11,#%30
        ld     r11,@r11

INC_R1210:
        CP     R11,R12          ;WHEN VALUE IN R12 IS INCed AND REACH R11
        JP     Z, CLOSE_MOTOR10
        inc    r12

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

JP INC_R1210

CLOSE_MOTOR10:

```
ld r13,%59 ;mask bytes before sending to p0
ld r14,r13 ;r14 to tell show many steps to jump
and r13,#%3 ;only d1,d0 are to send
and r14,#%3c ;only d5,d4,d3,d2 are index to jump
ld r15,#2 ;loop 2 times
rorl0: rr r14
djnz r15,rorl0
ld p2,%49 ;send byte to p2
ld p0,r13 ;send byte (2 bit) to p0

add r10,r14 ;next value to stop pulse
ld r11,r10
add r11,#%20
ld r11,@r11
add r11,#%30
ld r11,@r11
```

```
; LD R13, #20 ;
;LOOP_UNTIL_20_MS:
; LD R14, #%E6
; LOOP_1MS:
; DJNZ R14, LOOP_1MS
; DJNZ R13, LOOP_UNTIL_20_MS
```

RET

SORT:

```
LD RP, #%30

ld r0,%60
ld r1,%61
ld r2,%62
ld r3,%63
ld r4,%64
ld r5,%65
ld r6,%66
ld r7,%67
ld r8,%68
ld r9,%69

ld rp,%20
```

```
; R10 TO R15 ARE USED AS GENERAL PURPOSE
; WHICH IS THE SAME BUNCH IN DRIVING MODULE
; R10 COUNTS HOW MANY MOTORS HAS BEEN SORTED
; R11 POINT TO POSITION THAT IS CONSIDERING
; R12 KEEP VALUE WHICH IS PREVIOUSLY SORTED, START FROM %FF
; R13 IS THE ABSOLUTE ADDRESS OF THE POSITION THAT IS CONSIDERING
```

LD R10,#0

NEXT_DIGIT:

```
LD R12,%FF ;START FROM FF
LD %3A,#0
LD R11,#0
```

REPEAT UNTIL 10:

```
LD R13,%30
ADD R13,R11 ;GET ADDRESS OF CONSIDERING POINT
CP R12,@R13 ;
JP ULT, NOT_LESS_THAN ;NOT WHAT WE WANT, SKIP TO NEXT VALUE

LD R15,%20 ;START CHECK&PUT ROUTINE
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

        CP      R10,#0
        JP      NE, LOOK_AGAIN
        JP      PUT

LOOK_AGAIN:
        LD      R14,@R15
        CP      R14,R11
        JP      EQ, DONT_PUT

PUT:
        INC     R15

        LD      R14,R15 ;R14 IS TEMP REG, TO HELP LOOPING
        SUB     R14,#%20
        CP      R10,R14
        JP      UGT, LOOK_AGAIN

        LD      R14,#%20 ;PUT IN INDEX TABLE, POINTED BY R14
        ADD     R14,R10 ;CALCULATE PROPER POSITION IN TABLE
        LD      @R14, R11 ;KEEP INDEX TO TABLE
        LD      %3A,#1 ;SET STATUS FLAG THAT WE FOUND THE NUMBER
                        ; WHICH IS LESS THAN

DONT_PUT:

NOT_LESS_THAN: ;FETCH SORTED NUMBER TO R12
        CP      %3A,#0
        JP      EQ, FIRST_IN_LOOP

NORMAL_FETCH:
        LD      R13,#%30
        LD      R14,R10
        ADD     R14,#%20
        ADD     R13,@R14

FETCH:
        LD      R12,@R13
        JP      NEXT_VALUE

FIRST_IN_LOOP:
        LD      R12,#%FF

NEXT_VALUE:
        INC     R11 ;GOTO THE NEXT VALUE
        CP      R11,#9
        JP      ULE, REPEAT_UNTIL_10
        LD      R11,#00
        INC     R10
        CP      R10,#9
        JP      ULE,NEXT_DIGIT

RET

;*****
PREPARE:
        ld      rp,#%20

        ld      r10,#0 ;unequal index (how many unequal number)
        ld      r12,#0 ;absolute index (how many pass)
        ld      r14,#0 ;to count how many equal number in loop

        ld      r15,#%fff
        ld      %40(R10),r15
        ld      r15,#%03
        ld      %50(R10),r15

loop_uneq:

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

loop_eq:
    ld    r11,#%20          ;considering index
    add   r11,r12
    ld    r11,@r11          ;get index from buffer to R11

```

```

    cp    r11,#5
    jp    uge, not4
    cp    r11,#0
    jp    ne, not0
    ld    r15,%40(r10)
    xor   r15,#1
    ld    %40(r10), r15
    jp    finish1

```

```

not0:   cp    r11,#1
        jp    ne, not1
        ld    r15,%40(r10)
        xor   r15,#2
        ld    %40(r10), r15
        jp    finish1

```

```

not1:   cp    r11,#2
        jp    ne, not2
        ld    r15,%40(r10)
        xor   r15,#4
        ld    %40(r10), r15
        jp    finish1

```

```

not2:   cp    r11,#3
        jp    ne, not3
        ld    r15,%40(r10)
        xor   r15,#8
        ld    %40(r10), r15
        jp    finish1

```

```

not3:   cp    r11,#4
        jp    ne, not4
        ld    r15,%40(r10)
        xor   r15,#16
        ld    %40(r10), r15
        jp    finish1

```

```

not4:   cp    r11,#5
        jp    ne, not5
        ld    r15,%40(r10)
        xor   r15,#32
        ld    %40(r10), r15
        jp    finish1

```

```

not5:   cp    r11,#6
        jp    ne, not6
        ld    r15,%40(r10)
        xor   r15,#64
        ld    %40(r10), r15
        jp    finish1

```

```

not6:   cp    r11,#7
        jp    ne, not7
        ld    r15,%40(r10)
        xor   r15,#128
        ld    %40(r10), r15
        jp    finish1

```

```

not7:   cp    r11,#8
        jp    ne, not8
        ld    r15,%50(r10)
        and   r15,#3
        xor   r15,#1
        ld    %50(r10), r15
        jp    finish1

```

```

not8:   ld    r15,%50(r10)
        and   r15,#3
        xor   r15,#2
        ld    %50(r10), r15

```

เอกสารนี้เป็นเอกสารสำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆทั้งสิ้น ยี่สิบปีให้หลังเมื่อเทคโนโลยีเปลี่ยนแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

finish1:

    cp        r12,#9
    jp        uge,finish

    inc       r14
    ld        r15,r12
    add       r15,#%20
    ld        r15,@r15
    add       r15,#%30
    ld        r15,@r15

    ld        r13,r12
    inc       r13
    add       r13,#%20
    ld        r13,@r13
    add       r13,#%30
    ld        r13,@r13

    cp        r13,r15
    jp        ne,next_uneq
    inc       r12
    jp        loop_eq

next_uneq:

    ld        r15,#2
rotate:
    rl        r14
    djnz      r15,rotate

    ld        r15,%50(r10)
    and       r15,#3
    xor       r15,r14
    ld        %50(r10),r15

    ld        r15,%50(r10)
    and       r15,#3
    ld        r13,%40(r10)
    inc       r10
    ld        %50(r10), r15
    ld        %40(r10), r13

    inc       r12
    ld        r14,#0
    jp        loop_uneq

finish:
    sub       r12,r10
    jp        z,jump_finish
    ld        r15,#0

put_zero:
    inc       r10
    ld        %40(r10),r15
    ld        %50(r10),r15
    djnz      r12,put_zero

jump_finish:
    ret

.END

```

```

; *****
; DRIVE.ASM
; TEST ONE PULSE PROCEDURE TO PRODUCE CONTROL
; PULSE IN ORDER TO CONTROL 10 SERVO POSITION SIMULTANEOUSLY
; *****
.ORG 0CH

LD      P2M, #0          ;SET PORT 2 AS OUTPUT PORT
LD      P3M, #1          ;SET PULL UP
LD      P01M, #%04
LD      P2, #0

LOOP:
CALL    SORT
CALL    CONTROL
JP      LOOP

CONTROL: ;*****PROCEDURE*****
;GENERATE CONTROL PULSE OF 8 MOTORS SIMULTANEOUSLY
;READ POSITION FROM TABLE
;R13, R14 USE FOR LOOPING

LD      RP, #%20          ;USE REGS BANK 2, R0-R9 IS INDEX
;R10-R15 IS GENERAL PURPOSE

LD      P2, #%FF          ;SET ALL PORTS TO '1'
LD      R12, 0            ;R12 IS USED FOR COUNTER

; LD      R13, #1
;R13_LOOP:
; LD      R14, #01        ;'E6' IS THE VALUE THAT MAKE LOOP = 0.5 MS
;HIGH:
; DJNZ    R14, HIGH
; DJNZ    R13, R13_LOOP   ;TOTALLY 1.0 MS

;*****
LD      R10, R0
ADD     R10, #30          ;ADDRESS 30h-39h IS TABLE OF MOTORS
LD      R11, @R10         ;GET MOTOR VALUE TO R11
LD      R15, R0           ;R12 POINT TO MOTOR No.

INC_R121:
CP      R11, R12          ;WHEN VALUE IN R12 IS INCed AND REACH R11
JP      Z, CLOSE_MOTOR1
INC     R12
JP      INC_R121

CLOSE_MOTOR1:
CP      R15, #8
JP      NE, SEVEN1
XOR     P2, #128
SEVEN1: CP      R15, #7
JP      NE, SIX1
XOR     P2, #64
SIX1:   CP      R15, #6
JP      NE, FIVE1
XOR     P2, #32
FIVE1:  CP      R15, #5
JP      NE, FOUR1
XOR     P2, #16
FOUR1:  CP      R15, #4
JP      NE, THREE1
XOR     P2, #8
THREE1: CP      R15, #3
JP      NE, TWO1
XOR     P2, #4
TWO1:   CP      R15, #2
JP      NE, ONE1
XOR     P2, #2
ONE1:   CP      R15, #1
JP      NE, NOT_MATCH1
XOR     P2, #1

```

NOT_MATCH1:

;*****

```
LD      R10,R1
ADD     R10,#%30      ;ADDRESS 30h-39h IS TABLE OF MOTORS
LD      R11,@R10      ;GET MOTOR VALUE TO R11
LD      R15,R1        ;R12 POINT TO MOTOR No.
```

INC_R122:

```
CP      R11,R12      ;WHEN VALUE IN R12 IS INCed AND REACH R11
JP      Z, CLOSE_MOTOR2
INC     R12
JP      INC_R122
```

CLOSE_MOTOR2:

```
CP      R15,#8
JP      NE,SEVEN2
XOR     P2,#128
SEVEN2: CP      R15,#7
JP      NE,SIX2
XOR     P2,#64
SIX2:   CP      R15,#6
JP      NE,FIVE2
XOR     P2,#32
FIVE2:  CP      R15,#5
JP      NE,FOUR2
XOR     P2,#16
FOUR2:  CP      R15,#4
JP      NE,THREE2
XOR     P2,#8
THREE2: CP      R15,#3
JP      NE,TWO2
XOR     P2,#4
TWO2:   CP      R15,#2
JP      NE,ONE2
XOR     P2,#2
ONE2:   CP      R15,#1
JP      NE,NOT_MATCH2
XOR     P2,#1
```

NOT_MATCH2:

;*****

```
LD      R10,R2
ADD     R10,#%30      ;ADDRESS 30h-39h IS TABLE OF MOTORS
LD      R11,@R10      ;GET MOTOR VALUE TO R11
LD      R15,R2        ;R12 POINT TO MOTOR No.
```

INC_R123:

```
CP      R11,R12      ;WHEN VALUE IN R12 IS INCed AND REACH R11
JP      Z, CLOSE_MOTOR3
INC     R12
JP      INC_R123
```

CLOSE_MOTOR3:

```
CP      R15,#8
JP      NE,SEVEN3
XOR     P2,#128
SEVEN3: CP      R15,#7
JP      NE,SIX3
XOR     P2,#64
SIX3:   CP      R15,#6
JP      NE,FIVE3
XOR     P2,#32
FIVE3:  CP      R15,#5
JP      NE,FOUR3
XOR     P2,#16
FOUR3:  CP      R15,#4
JP      NE,THREE3
XOR     P2,#8
THREE3: CP      R15,#3
```

เอกสารนี้เป็นเอกสารสำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่าการใดๆ ทั้งสิ้น หากมีข้อผิดพลาดหรือต้องการแจ้งถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

        JP      NE,TWO3
        XOR     P2,#4
TWO3:   CP      R15,#2
        JP      NE,ONE3
        XOR     P2,#2
ONE3:   CP      R15,#1
        JP      NE,NOT_MATCH3
        XOR     P2,#1

```

NOT_MATCH3:

;*****

```

        LD      R10,R3
        ADD     R10,#30          ;ADDRESS 30h-39h IS TABLE OF MOTORS
        LD      R11,@R10        ;GET MOTOR VALUE TO R11
        LD      R15,R3          ;R12 POINT TO MOTOR No.

```

INC_R124:

```

        CP      R11,R12          ;WHEN VALUE IN R12 IS INCed AND REACH R11
        JP      Z, CLOSE_MOTOR4
        INC     R12
        JP      INC_R124

```

CLOSE_MOTOR4:

```

        CP      R15,#8
        JP      NE,SEVEN4
        XOR     P2,#128
SEVEN4: CP      R15,#7
        JP      NE,SIX4
        XOR     P2,#64
SIX4:   CP      R15,#6
        JP      NE,FIVE4
        XOR     P2,#32
FIVE4:  CP      R15,#5
        JP      NE,FOUR4
        XOR     P2,#16
FOUR4:  CP      R15,#4
        JP      NE,THREE4
        XOR     P2,#8
THREE4: CP      R15,#3
        JP      NE,TWO4
        XOR     P2,#4
TWO4:   CP      R15,#2
        JP      NE,ONE4
        XOR     P2,#2
ONE4:   CP      R15,#1
        JP      NE,NOT_MATCH4
        XOR     P2,#1

```

NOT_MATCH4:

;*****

```

        LD      R10,R4
        ADD     R10,#30          ;ADDRESS 30h-39h IS TABLE OF MOTORS
        LD      R11,@R10        ;GET MOTOR VALUE TO R11
        LD      R15,R4          ;R12 POINT TO MOTOR No.

```

INC_R125:

```

        CP      R11,R12          ;WHEN VALUE IN R12 IS INCed AND REACH R11
        JP      Z, CLOSE_MOTOR5
        INC     R12
        JP      INC_R125

```

CLOSE_MOTOR5:

```

        CP      R15,#8
        JP      NE,SEVEN5
        XOR     P2,#128
SEVEN5: CP      R15,#7
        JP      NE,SIX5
        XOR     P2,#64
SIX5:   CP      R15,#6

```

```

JP      NE,FIVE5
XOR     P2,#32
FIVE5:  CP      R15,#5
JP      NE,FOUR5
XOR     P2,#16
FOUR5:  CP      R15,#4
JP      NE,THREE5
XOR     P2,#8
THREE5: CP      R15,#3
JP      NE,TWO5
XOR     P2,#4
TWO5:   CP      R15,#2
JP      NE,ONE5
XOR     P2,#2
ONE5:   CP      R15,#1
JP      NE,NOT_MATCH5
XOR     P2,#1

```

NOT_MATCH5:

;*****

```

LD      R10,R5
ADD     R10,#30      ;ADDRESS 30h-39h IS TABLE OF MOTORS
LD      R11,@R10     ;GET MOTOR VALUE TO R11
LD      R15,R5       ;R12 POINT TO MOTOR No.

```

INC_R126:

```

CP      R11,R12      ;WHEN VALUE IN R12 IS INCed AND REACH R11
JP      Z,CLOSE_MOTOR6
INC     R12
JP      INC_R126

```

CLOSE_MOTOR6:

```

CP      R15,#8
JP      NE,SEVEN6
XOR     P2,#128
SEVEN6: CP      R15,#7
JP      NE,SIX6
XOR     P2,#64
SIX6:   CP      R15,#6
JP      NE,FIVE6
XOR     P2,#32
FIVE6:  CP      R15,#5
JP      NE,FOUR6
XOR     P2,#16
FOUR6:  CP      R15,#4
JP      NE,THREE6
XOR     P2,#8
THREE6: CP      R15,#3
JP      NE,TWO6
XOR     P2,#4
TWO6:   CP      R15,#2
JP      NE,ONE6
XOR     P2,#2
ONE6:   CP      R15,#1
JP      NE,NOT_MATCH6
XOR     P2,#1

```

NOT_MATCH6:

;*****

```

LD      R10,R6
ADD     R10,#30      ;ADDRESS 30h-39h IS TABLE OF MOTORS
LD      R11,@R10     ;GET MOTOR VALUE TO R11
LD      R15,R6       ;R12 POINT TO MOTOR No.

```

INC_R127:

```

CP      R11,R12      ;WHEN VALUE IN R12 IS INCed AND REACH R11

```

เอกสารนี้เป็นเอกสารสงวนลิขสิทธิ์ของมหาวิทยาลัยเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
 ไม่ควรนำเอกสารนี้ไปใช้ในการศึกษาหาความรู้โดยไม่ได้รับอนุญาตจากทางมหาวิทยาลัย
 ไม่ควรนำเอกสารนี้ไปใช้ในการเผยแพร่โดยไม่ได้รับอนุญาตจากทางมหาวิทยาลัย

```

CLOSE_MOTOR7:
  CP      R15,#8
  JP      NE,SEVEN7
  XOR     P2,#128
SEVEN7: CP      R15,#7
  JP      NE,SIX7
  XOR     P2,#64
SIX7:    CP      R15,#6
  JP      NE,FIVE7
  XOR     P2,#32
FIVE7:   CP      R15,#5
  JP      NE,FOUR7
  XOR     P2,#16
FOUR7:   CP      R15,#4
  JP      NE,THREE7
  XOR     P2,#8
THREE7:  CP      R15,#3
  JP      NE,TWO7
  XOR     P2,#4
TWO7:    CP      R15,#2
  JP      NE,ONE7
  XOR     P2,#2
ONE7:    CP      R15,#1
  JP      NE,NOT_MATCH7
  XOR     P2,#1

```

NOT_MATCH7:

;*****

```

  LD      R10,R7
  ADD     R10,#30      ;ADDRESS 30h-39h IS TABLE OF MOTORS
  LD      R11,@R10     ;GET MOTOR VALUE TO R11
  LD      R15,R7       ;R12 POINT TO MOTOR No.

```

INC_R128:

```

  CP      R11,R12      ;WHEN VALUE IN R12 IS INCed AND REACH R11
  JP      Z,CLOSE_MOTORS8
  INC     R12
  JP      INC_R128

```

CLOSE_MOTOR8:

```

  CP      R15,#8
  JP      NE,SEVEN8
  XOR     P2,#128
SEVEN8: CP      R15,#7
  JP      NE,SIX8
  XOR     P2,#64
SIX8:    CP      R15,#6
  JP      NE,FIVE8
  XOR     P2,#32
FIVE8:   CP      R15,#5
  JP      NE,FOUR8
  XOR     P2,#16
FOUR8:   CP      R15,#4
  JP      NE,THREE8
  XOR     P2,#8
THREE8:  CP      R15,#3
  JP      NE,TWO8
  XOR     P2,#4
TWO8:    CP      R15,#2
  JP      NE,ONE8
  XOR     P2,#2
ONE8:    CP      R15,#1
  JP      NE,NOT_MATCH8
  XOR     P2,#1

```

NOT_MATCH8:

;*****

```

  LD      R10,R8
  ADD     R10,#30      ;ADDRESS 30h-39h IS TABLE OF MOTORS
  LD      R11,@R10     ;GET MOTOR VALUE TO R11
  LD      R15,R8       ;R12 POINT TO MOTOR No.

```

เอกสารนี้เป็นเอกสารสำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
 ไม่ว่าการคัดลอกหรือการดัดแปลงเอกสารทุกครั้งที่มีการนำไปใช้

```

INC_R129:
  CP      R11,R12      ;WHEN VALUE IN R12 IS INCed AND REACH R11
  JP      Z, CLOSE_MOTOR9
  INC     R12
  JP      INC_R129

```

```

CLOSE_MOTOR9:
  CP      R15,#8
  JP      NE,SEVEN9
  XOR     P2,#128
SEVEN9:   CP      R15,#7
  JP      NE,SIX9
  XOR     P2,#64
SIX9:     CP      R15,#6
  JP      NE,FIVE9
  XOR     P2,#32
FIVE9:    CP      R15,#5
  JP      NE,FOUR9
  XOR     P2,#16
FOUR9:    CP      R15,#4
  JP      NE,THREE9
  XOR     P2,#8
THREE9:   CP      R15,#3
  JP      NE,TWO9
  XOR     P2,#4
TWO9:     CP      R15,#2
  JP      NE,ONE9
  XOR     P2,#2
ONE9:     CP      R15,#1
  JP      NE,NOT_MATCH9
  XOR     P2,#1

```

NOT_MATCH9:

```

;*****
LD      R10,R9
ADD     R10,#30      ;ADDRESS 30h-39h IS TABLE OF MOTORS
LD      R11,@R10     ;GET MOTOR VALUE TO R11
LD      R15,R9       ;R12 POINT TO MOTOR No.

```

```

INC_R1210:
  CP      R11,R12      ;WHEN VALUE IN R12 IS INCed AND REACH R11
  JP      Z, CLOSE_MOTOR10
  INC     R12
  JP      INC_R1210

```

```

CLOSE_MOTOR10:
  CP      R15,#8
  JP      NE,SEVEN10
  XOR     P2,#128
SEVEN10:  CP      R15,#7
  JP      NE,SIX10
  XOR     P2,#64
SIX10:    CP      R15,#6
  JP      NE,FIVE10
  XOR     P2,#32
FIVE10:   CP      R15,#5
  JP      NE,FOUR10
  XOR     P2,#16
FOUR10:   CP      R15,#4
  JP      NE,THREE10
  XOR     P2,#8
THREE10:  CP      R15,#3
  JP      NE,TWO10
  XOR     P2,#4
TWO10:    CP      R15,#2
  JP      NE,ONE10
  XOR     P2,#2
ONE10:    CP      R15,#1
  JP      NE,NOT_MATCH10
  XOR     P2,#1

```

NOT_MATCH10:

เป็นเอกสารที่สงวนลิขสิทธิ์สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่าในรูปแบบใดๆทั้งสิ้น ห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

        LD      R13, #20                ;(FORMER 34)17 ms LEFT TO DELAY, =34*0.5 CYCLE
LOOP_UNTIL_20_MS:
        LD      R14, #%E6
LOOP_1MS:
        DJNZ    R14, LOOP_1MS
        DJNZ    R13, LOOP_UNTIL_20_MS

        LD      RP, #%00

        RET

```

```

SORT:
        LD      RP, #20
; R10 TO R15 ARE USED AS GENERAL PURPOSE
; WHICH IS THE SAME BUNCH IN DRIVING MODULE
; R10 COUNTS HOW MANY MOTORS HAS BEEN SORTED
; R11 POINT TO POSITION THAT IS CONSIDERING
; R12 KEEP VALUE WHICH IS PREVIOUSLY SORTED, START FROM %FF
; R13 IS THE ABSOLUTE ADDRESS OF THE POSITION THAT IS CONSIDERING

```

```

        LD      R10, #0
NEXT_DIGIT:
        LD      R12, #%FF                ;START FROM FF
        LD      %3A, #0
        LD      R11, #0

```

```

REPEAT_UNTIL_10:
        LD      R13, #20
        ADD     R13, R11                ;GET ADDRESS OF CONSIDERING POINT
        CP      R12, @R13                ;
        JP      ULT, NOT_LESS_THAN        ;NOT WHAT WE WANT, SKIP TO NEXT VALUE

        LD      R15, #20                ;START CHECK&PUT ROUTINE
;
        CP      R10, #0
        JP      NE, LOOK_AGAIN
        JP      PUT

```

```

LOOK_AGAIN:
        LD      R14, @R15
        CP      R14, R11
        JP      EQ, DONT_PUT
PUT:
        INC     R15
;
        LD      R14, R15                ;R14 IS TEMP REG, TO HELP LOOPING
        SUB     R14, #20
        CP      R10, R14
        JP      UGT, LOOK_AGAIN
;
        LD      R14, #20                ;PUT IN INDEX TABLE, POINTED BY R14
        ADD     R14, R10                ;CALCULATE PROPER POSITION IN TABLE
        LD      @R14, R11                ;KEEP INDEX TO TABLE
        LD      %3A, #1                ;SET STATUS FLAG THAT WE FOUND THE NUMBER
; WHICH IS LESS THAN
;

```

```

DONT_PUT:
; CP      R10, #0                ;FOR FIRST TIME, IN CASE THAT
; JP      NE, NOT_FIRST          ;THE FIRST VALUE IS 00
; CP      R11, #0
; JP      NE, OUT
; JP      PUT                    ;WE HAVE TO PUT IT BECAUSE IT IS
;                                ;THE FIRST VALUE

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

```

NOT_LESS_THAN:
        CP      %3A, #0                ;FETCH SORTED NUMBER TO R12
        JP      EQ, FIRST_IN_LOOP

```

ลิขสิทธิ์ทั้งหมดมีให้ดัดแปลงแก้ไขโดยไม่เสียค่าใช้จ่ายของเอกสารทุกครั้งที่มีการนำไปใช้

NORMAL FETCH:

```
LD    R13, #30
LD    R14, R10
ADD   R14, #20
ADD   R13, @R14
```

FETCH:

```
LD    R12, @R13
JP    NEXT_VALUE
```

FIRST IN_LOOP:

```
LD    R12, #FF
```

NEXT_VALUE:

```
INC    R11                      ;GOTO THE NEXT VALUE
CP     R11, #9
JP     ULE, REPEAT_UNTIL_10
LD     R11, #00
INC    R10
CP     R10, #10
JP     ULE, NEXT_DIGIT

RET
```

.END



```
; *****
; POSITION.ASM
; TEST ONE PULSE PROCEDURE TO PRODUCE CONTROL
; PULSE IN ORDER TO CONTROL SERVO POSITION
; *****
```

```
.ORG 0CH
LD P2M, #0 ;SET PORT 2 AS OUTPUT PORT
LD P3M, #1 ;SET PULL UP
LD P01M, #04
LD RP, #10 ;USE PAGE 1 OF REGISTER FILE
```

```
LOOP: CALL ONE_PULSE ;LOOP FOR ONE POSITION FOR R7 TIMES
```

```
INC R8
SUB R5, #5

CP R5, #5
JP NE, NOT_RESET_POSITION
LD R5, #5
```

```
NOT_RESET_POSITION:
CP R8, #8
JP LT, NOT_RESET_MOTOR_NO
LD R8, #1
NOT_RESET_MOTOR_NO:
JP LOOP
```

```
ONE_PULSE: ;*****PROCEDURE*****
;GENERATE CONTROL PULSE TO CONTROL SERVO POSITION
; INPUT R5=DELAY TIME
; R8=PIN OF PORT 2
```

```
PUSH R4
PUSH R6
```

```
CP R8, #8
JP NE, SEVEN
XOR P2, #128
SEVEN: CP R8, #7
JP NE, SIX
XOR P2, #64
SIX: CP R8, #6
JP NE, FIVE
XOR P2, #32
FIVE: CP R8, #5
JP NE, FOUR
XOR P2, #16
FOUR: CP R8, #4
JP NE, THREE
XOR P2, #8
THREE: CP R8, #3
JP NE, TWO
XOR P2, #4
TWO: CP R8, #2
JP NE, ONE
XOR P2, #2
ONE: CP R8, #1
JP NE, NOT_MATCH
XOR P2, #1
```

```
NOT_MATCH:
```

```
LD R6, #2
```

```
R6_LOOP: LD R4, #E6 ;'E6' IS THE VALUE THAT MAKE LOOP = 0.5 MS
```

HIGH:
เอกสารนี้เป็นDJNZสารที่R4, HIGHการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

        DJNZ     R6, R6_LOOP                ;TOTALLY 1.0 MS

        LD       R6, #2                    ;THE ADDITIONAL PART DUE TO R5
ADDITIONAL_HIGH:
        LD       R4, R5                    ;ANOTHER 1 MS
HIGH_STATUS:
        DJNZ     R4, HIGH_STATUS
        DJNZ     R6, ADDITIONAL_HIGH

        CP       R8, #8
        JP       NE, SEVEN2
        XOR      P2, #128
SEVEN2:  CP       R8, #7
        JP       NE, SIX2
        XOR      P2, #64
SIX2:   CP       R8, #6
        JP       NE, FIVE2
        XOR      P2, #32
FIVE2:  CP       R8, #5
        JP       NE, FOUR2
        XOR      P2, #16
FOUR2:  CP       R8, #4
        JP       NE, THREE2
        XOR      P2, #8
THREE2: CP       R8, #3
        JP       NE, TWO2
        XOR      P2, #4
TWO2:   CP       R8, #2
        JP       NE, ONE2
        XOR      P2, #2
ONE2:   CP       R8, #1
        JP       NE, NOT_MATCH2
        XOR      P2, #1
NOT_MATCH2:

        LD       R6, #2
ADDITIONAL_LOW:
        LD       R4, #0x06                ;CALCULATE THE LOW STATUS TIME
        SUB      R4, R5
LOW_STATUS:
        DJNZ     R4, LOW_STATUS
        DJNZ     R6, ADDITIONAL_LOW

        LD       R6, #34                  ;17 mS LEFT TO DELAY, =34*0.5 CYCLE
LOOP_UNTIL_20_MS:
        LD       R4, #0x06
LOOP_1MS:
        DJNZ     R4, LOOP_1MS
        DJNZ     R6, LOOP_UNTIL_20_MS

        POP      R6
        POP      R4

        RET

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
;*****
; DRIVER.ASM
; TEST DRIVING SERVO MOTOR WITH THE CONSTANTANEOUS CHANGE
; OF CONTROL PULSE FROM LOWEST VALUE TO HIGHEST VALUE BACK AND
; FORTH OF CONTROL REGISTER
;*****
```

```
.ORG 0CH
```

```
LD      P2M, #0.          ;SET PORT 2 AS OUTPUT PORT
LD      P3M, #1          ;SET PULL UP

LD      P01M, #%04
LD      P2, #0
```

```
LOOP:
```

```
LD      R6, #%70          ;TEST LOOP 50h POSITION
LD      R5, #%E5          ;START POSITION
```

```
R6_POSITION:              ;*****FORWARD DIRECTION*****
LD      R7, #%2          ;DELAY TIME PER ONE POSITION
```

```
ONE_POSITION:
CALL    ONE_PULSE          ;LOOP FOR ONE POSITION FOR R7 TIMES
DJNZ    R7, ONE_POSITION
SUB     R5, #2
DJNZ    R6, R6_POSITION
```

```
;*****REVERSE DIRECTION*****
```

```
LD      R6, #%70
REV_TEN_POSITION:
LD      R7, #2          ;DELAY TIME PER ONE POSITION
```

```
REV_ONE_POSITION:
CALL    ONE_PULSE          ;LOOP FOR ONE POSITION FOR R7 TIMES
DJNZ    R7, REV_ONE_POSITION
```

```
ADD     R5, #2
DJNZ    R6, REV_TEN_POSITION
```

```
JP      LOOP
```

```
ONE_PULSE:                ;*****PROCEDURE*****
; INPUT:  R5 = DELAY TIME RANGE FROM 1-E6h
; OUTPUT: ONE PULSE WITH PERIOD=19 MS
; LOGIC '1' DEPENDS ON VALUE IN R5
; LOGIC '0' ELSEWHERE (CALCULATE FROM R5
; TO FIT 19 MS)
```

```
PUSH    R4
PUSH    R6
```

```
XOR     P2, #1          ;ACTIVE P2.0
LD      R6, #2          ;MAKE THE 2xLOOP = 1.0 MS
```

```
R6_LOOP:
LD      R4, #E6          ;'E6' IS THE VALUE THAT MAKE LOOP = 0.5 MS
```

```
HIGH:
DJNZ    R4, HIGH
DJNZ    R6, R6_LOOP      ;TOTALLY 1.0 MS
```

```
LD      R6, #2          ;THE ADDITIONAL PART DUE TO R5
ADDITIONAL_HIGH:
LD      R4, R5          ;ANOTHER 1 MS
```

```
HIGH_STATUS:
DJNZ    R4, HIGH_STATUS
DJNZ    R6, ADDITIONAL_HIGH
```

```
XOR     P2, #1          ; TOGGLE P2.0
```

```
LD      R6, #2
ADDITIONAL_LOW:
LD      R4, #E6          ;CALCULATE THE LOW STATUS TIME
SUB     R4, R5
```

เอกสารนี้เป็นเอกสารสงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
LOW_STATUS:
    DJNZ     R4, LOW_STATUS
    DJNZ     R6, ADDITIONAL_LOW

    LD       R6, #17                                ;17 mS LEFT TO DELAY, =34*0.5 CYCLE

LOOP_UNTIL_20_MS:
    LD       R4, #0E6
LOOP_1MS:
    DJNZ     R4, LOOP_1MS
    DJNZ     R6, LOOP_UNTIL_20_MS

    POP      R6
    POP      R4
    RET

.END
```



```

;*****
; DRIVER.ASM
; TEST DRIVING SERVO MOTOR WITH THE CONSTANTANEOUS CHANGE
; OF CONTROL PULSE FROM LOWEST VALUE TO HIGHEST VALUE BACK AND
; FORTH OF CONTROL REGISTER
;*****

.ORG 0CH

LD      P2M, #0          ;SET PORT 2 AS OUTPUT PORT
LD      P3M, #1          ;SET PULL UP

LD      P01M, #%04
LD      P2, #0

LOOP:
LD      R6, #%70          ;TEST LOOP 50h POSITION
LD      R5, #%E5          ;START POSITION

R6_POSITION:
LD      R7, #2            ;*****FORWARD DIRECTION*****
;DELAY TIME PER ONE POSITION
ONE_POSITION:
CALL    ONE_PULSE         ;LOOP FOR ONE POSITION FOR R7 TIMES
DJNZ    R7, ONE_POSITION
SUB     R5, #2
DJNZ    R6, R6_POSITION

;*****REVERSE DIRECTION*****
LD      R6, #%70
REV_TEN_POSITION:
LD      R7, #2            ;DELAY TIME PER ONE POSITION
REV_ONE_POSITION:
CALL    ONE_PULSE         ;LOOP FOR ONE POSITION FOR R7 TIMES
DJNZ    R7, REV_ONE_POSITION
ADD     R5, #2
DJNZ    R6, REV_TEN_POSITION
JP      LOOP

ONE_PULSE:
;*****PROCEDURE*****
; INPUT:  R5 = DELAY TIME RANGE FROM 1-E6h
; OUTPUT: ONE PULSE WITH PERIOD=19 MS
; LOGIC '1' DEPENDS ON VALUE IN R5
; LOGIC '0' ELSEWHERE (CALCULATE FROM R5
; TO FIT 19 MS)

PUSH    R4
PUSH    R6

XOR     P2, #1            ;ACTIVE P2.0
LD      R6, #2            ;MAKE THE 2xLOOP = 1.0 MS
R6_LOOP:
LD      R4, #%E6          ;'E6' IS THE VALUE THAT MAKE LOOP = 0.5 MS
HIGH:
DJNZ    R4, HIGH
DJNZ    R6, R6_LOOP        ;TOTALLY 1.0 MS

LD      R6, #2            ;THE ADDITIONAL PART DUE TO R5
ADDITIONAL_HIGH:
LD      R4, R5            ;ANOTHER 1 MS
HIGH STATUS:
DJNZ    R4, HIGH STATUS
DJNZ    R6, ADDITIONAL_HIGH

XOR     P2, #1            ; TOGGLE P2.0
LD      R6, #2

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

ADDITIONAL_LOW:
LD      R4, #E6          ;CALCULATE THE LOW STATUS TIME
SUB     R4, R5
LOW_STATUS:
DJNZ    R4, LOW_STATUS
DJNZ    R6, ADDITIONAL_LOW

LD      R6, #17          ;17 mS LEFT TO DELAY, =34*0.5 CYCLE

LOOP_UNTIL_20_MS:
LD      R4, #E6
LOOP_1MS:
DJNZ    R4, LOOP_1MS
DJNZ    R6, LOOP_UNTIL_20_MS

POP     R6
POP     R4
RET
ND

```



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
;*****
;PREPARE.ASM
; THIS IS ISOLATED PROGRAM TO TEST PREPARE MODULE IN NEWDRIVE.ASM
; WE CAN MEASURE TIME SPENT BY THIS MODULE BY ACTIVATE-DEACTIVATE
; P2 IN THE BEGINNING AND ENDING
;*****
```

```
.ORG 0CH
```

```
LD P2M, #0 ;SET PORT 2 AS OUTPUT PORT
LD P3M, #1 ;SET PULL UP
LD P01M, #%04
```

```
start:
```

```
LD P2, #%00
ld rp, #%20

ld r10, #0 ;unequal index (how many unequal number)
ld r12, #0 ;absolute index (how many pass)
ld r14, #0 ;to count how many equal number in loop

ld r15, #fff
ld %40(R10), r15
ld r15, #03
ld %50(R10), r15
```

```
loop_uneq:
```

```
loop_eq:
```

```
ld r11, #%20 ;considering index
add r11, r12
ld r11, @r11 ;get index from buffer to R11
```

```
cp r11, #5
jp uge, not4
cp r11, #0
jp ne, not0
ld r15, %40(r10)
xor r15, #1
ld %40(r10), r15
jp finish1
```

```
not0: cp r11, #1
jp ne, not1
ld r15, %40(r10)
xor r15, #2
ld %40(r10), r15
jp finish1
```

```
not1: cp r11, #2
jp ne, not2
ld r15, %40(r10)
xor r15, #4
ld %40(r10), r15
jp finish1
```

```
not2: cp r11, #3
jp ne, not3
ld r15, %40(r10)
xor r15, #8
ld %40(r10), r15
jp finish1
```

```
not3: cp r11, #4
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

    jp      ne, not4
    ld      r15, %40(r10)
    xor     r15, #16
    ld      %40(r10), r15
    jp      finish1
not4:    cp      r11, #5
    jp      ne, not5
    ld      r15, %40(r10)
    xor     r15, #32
    ld      %40(r10), r15
    jp      finish1
not5:    cp      r11, #6
    jp      ne, not6
    ld      r15, %40(r10)
    xor     r15, #64
    ld      %40(r10), r15
    jp      finish1
not6:    cp      r11, #7
    jp      ne, not7
    ld      r15, %40(r10)
    xor     r15, #128
    ld      %40(r10), r15
    jp      finish1
not7:    cp      r11, #8
    jp      ne, not8
    ld      r15, %50(r10)
    and     r15, #3
    xor     r15, #1
    ld      %50(r10), r15
    jp      finish1
not8:    ld      r15, %50(r10)
    and     r15, #3
    xor     r15, #2
    ld      %50(r10), r15

finish1:
    cp      r12, #9
    jp      uge, finish

    inc     r14
    ld      r15, r12
    add     r15, #20
    ld      r15, @r15
    add     r15, #30
    ld      r15, @r15

    ld      r13, r12
    inc     r13
    add     r13, #20
    ld      r13, @r13
    add     r13, #30
    ld      r13, @r13

    cp      r13, r15
    jp      ne, next_uneq
    inc     r12
    jp      loop_eq

next_uneq:
    ld      r15, #2

rotate:
    rl
    djnz    r15, rotate

```

```

ld      r15,%50(r10)
and     r15,#3
xor     r15,r14
ld      %50(r10),r15

ld      r15,%50(r10)
and     r15,#3
ld      r13,%40(r10)
inc     r10
ld      %50(r10), r15
ld      %40(r10), r13

inc     r12
ld      r14,#0
jp      loop_uneq

finish:
sub     r12,r10
jp      z,jump_start
ld      r15,#0
put_zero:
inc     r10
ld      %40(r10),r15
ld      %50(r10),r15
djnz    r12,put_zero

jump_start:
LD      P2,#%FF
jp      start

.end

```



```
;*****
;SORT.ASM
; ISOLATED VERSION OF MODULE SORT IN NEWDRIVE.ASM
; WE CAN MEASURE TIME CONSUMPTION BY ACTIVATE-DEACT PORT 2
; AT THE BEGINNING-ENDING POINT OF PROGRAM
;*****
```

```
.ORG 0CH
```

```
LD P2M, #0 ;SET PORT 2 AS OUTPUT PORT
LD P3M, #1 ;SET PULL UP
LD P01M, #%04
```

```
START: ;start sorting
```

```
LD P2, #0
```

```
LD RP, #%30
; First, move values from table at %60-%69 to buffers at %20-%29
ld r0, %60
ld r1, %61
ld r2, %62
ld r3, %63
ld r4, %64
ld r5, %65
ld r6, %66
ld r7, %67
ld r8, %68
ld r9, %69
```

```
; R10 TO R15 ARE USED AS GENERAL PURPOSE
; WHICH IS THE SAME BUNCH IN DRIVING MODULE
; R10 COUNTS HOW MANY MOTORS HAS BEEN SORTED
; R11 POINT TO POSITION THAT IS CONSIDERING
; R12 KEEP VALUE WHICH IS PREVIOUSLY SORTED, START FROM %FF
; R13 IS THE ABSOLUTE ADDRESS OF THE POSITION THAT IS CONSIDERING
```

```
ld rp, %20
LD R10, #0
```

```
NEXT_DIGIT:
```

```
LD R12, %FF ;START FROM FF
LD %3A, #0
LD R11, #0
```

```
REPEAT_UNTIL_10:
```

```
LD R13, %30
ADD R13, R11 ;GET ADDRESS OF CONSIDERING POINT
CP R12, @R13 ;
JP ULT, NOT_LESS_THAN ;NOT WHAT WE WANT, SKIP TO NEXT VALUE
```

```
LD R15, %20 ;START CHECK&PUT ROUTINE
;
```

```
CP R10, #0
JP NE, LOOK_AGAIN
JP PUT
```

```
LOOK_AGAIN:
```

```
LD R14, @R15 ;
CP R14, R11 ;
JP EQ, DONT_PUT ;
```

```
PUT: ;
INC R15 ;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

LD      R14,R15      ;
SUB     R14,#%20     ;R14 IS TEMP REG, TO HELP LOOPING
CP      R10,R14      ;
JP      UGT, LOOK_AGAIN ;
LD      R14,#%20     ;PUT IN INDEX TABLE, POINTED BY R14
ADD     R14,R10      ;CALCULATE PROPER POSITION IN TABLE
LD      @R14, R11    ;KEEP INDEX TO TABLE
LD      %3A,#1       ;SET STATUS FLAG THAT WE FOUND THE NUMBER
                        ; WHICH IS LESS THAN
DONT_PUT:            ;
NOT_LESS_THAN:      ;FETCH SORTED NUMBER TO R12
CP      %3A,#0
JP      EQ, FIRST_IN_LOOP

NORMAL_FETCH:
LD      R13,#%30
LD      R14,R10
ADD     R14,#%20
ADD     R13,@R14

FETCH:
LD      R12,@R13
JP      NEXT_VALUE

FIRST_IN_LOOP:
LD      R12,#%FF

NEXT_VALUE:
INC     R11           ;GOTO THE NEXT VALUE
CP      R11,#9
JP      ULE, REPEAT_UNTIL_10
LD      R11,#00
INC     R10
CP      R10,#09
JP      ULE,NEXT_DIGIT
ld      p2,%ff
JP      START

```

.END

```

VERSION 4.00
Begin VB.Form Form1
    BackColor       = &H00FF8080&
    BorderStyle     = 3 'Fixed Double
    Caption         = "Johnny Director 2.0.0"
    ClientHeight    = 7470
    ClientLeft      = 1380
    ClientTop       = 1290
    ClientWidth     = 10230
    Height          = 8160
    Left           = 1320
    LinkTopic       = "Form1"
    ScaleHeight     = 498
    ScaleMode       = 3 'Pixel
    ScaleWidth      = 682
    Top            = 660
    Width          = 10350
Begin VB.CommandButton Command5
    Caption         = "Send Frame"
BeginProperty Font
    name           = "Arial"
    charset        = 1
    weight         = 700
    size          = 12
    underline      = 0 'False
    italic         = 0 'False
    strikethrough  = 0 'False
EndProperty
    Height         = 495
    Left          = 8520
    TabIndex      = 55
    Top           = 5640
    Width         = 1575
End
Begin VB.TextBox Text3
    Height         = 315
    Left          = 4080
    TabIndex      = 54
    Text          = "NONAME.PDA"
    Top           = 120
    Width         = 1575
End
Begin VB.CommandButton Command4
    Caption         = "Break"
BeginProperty Font
    name           = "Arial"
    charset        = 1
    weight         = 400
    size          = 14.25
    underline      = 0 'False
    italic         = 0 'False
    strikethrough  = 0 'False
EndProperty
    Height         = 495
    Left          = 8880
    TabIndex      = 52
    Top           = 4920
    Width         = 1215
End
Begin VB.Timer Timer1
    Enabled        = 0 'False
    Left          = 6240
    Top           = 5880
End
Begin VB.TextBox Text2
    Height         = 315
    Left          = 4080

```

```

TabIndex      = 51
Text          = "10"
Top           = 840
Width         = 1215
End
Begin VB.TextBox Text1
  BeginProperty Font
    name        = "MS Sans Serif"
    charset     = 1
    weight      = 700
    size        = 12
    underline   = 0 'False
    italic      = 0 'False
    strikethrough = 0 'False
  EndProperty
  Height       = 495
  Left         = 9120
  TabIndex     = 50
  Text         = "0"
  Top          = 6720
  Width        = 615
End
Begin VB.VScrollBar VScroll1
  Height       = 495
  Left         = 9720
  Max          = 0
  Min          = 10
  TabIndex     = 49
  Top          = 6720
  Width        = 255
End
Begin VB.CommandButton Command3
  Caption      = "Copy From"
  BeginProperty Font
    name        = "Arial"
    charset     = 1
    weight      = 700
    size        = 11.25
    underline   = 0 'False
    italic      = 0 'False
    strikethrough = 0 'False
  EndProperty
  Height       = 495
  Left         = 7680
  TabIndex     = 48
  Top          = 6720
  Width        = 1335
End
Begin VB.PictureBox Picture4
  Height       = 2775
  Left         = 3360
  MousePointer = 13 'Custom
  ScaleHeight  = -183
  ScaleMode    = 0 'User
  ScaleTop     = 183
  ScaleWidth   = 135
  TabIndex     = 45
  Top          = 4560
  Width        = 2055
  Begin VB.Line Line31
    X1          = 72
    X2          = 72
    Y1          = 159
    Y2          = 31
  End
  Begin VB.Line Line30
    X1          = 64

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

X2          = 64
Y1          = 159
Y2          = 31
End
Begin VB.Line Line29
X1          = 24
X2          = 88
Y1          = 159
Y2          = 159
End
Begin VB.Line Line28
X1          = 48
X2          = 88
Y1          = 31
Y2          = 31
End
Begin VB.Line Line27
X1          = 48
X2          = 88
Y1          = 23
Y2          = 23
End
Begin VB.Line Line26
X1          = 48
X2          = 48
Y1          = 23
Y2          = 31
End
Begin VB.Line Line25
X1          = 88
X2          = 88
Y1          = 23
Y2          = 31
End
Begin VB.Shape Shape9
FillColor   = &H00FFFFC0&
FillStyle   = 0 'Solid
Height      = 255
Left        = 840
Shape       = 3 'Circle
Top         = 2040
Width       = 375
End
Begin VB.Shape Shape10
FillColor   = &H00FFFFC0&
FillStyle   = 0 'Solid
Height      = 255
Left        = 840
Shape       = 3 'Circle
Top         = 240
Width       = 375
End
End
Begin VB.PictureBox Picture3
Height      = 2775
Left        = 1200
MousePointer = 13 'Custom
ScaleHeight = -183
ScaleMode   = 0 'User
ScaleTop    = 183
ScaleWidth  = 135
TabIndex    = 44
Top         = 4560
Width       = 2055
Begin VB.Line Line24
X1          = 48
X2          = 88

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

        Y1          = 159
        Y2          = 159
    End
    Begin VB.Line Line23
        X1          = 64
        X2          = 64
        Y1          = 159
        Y2          = 95
    End
    Begin VB.Line Line22
        X1          = 64
        X2          = 64
        Y1          = 95
        Y2          = 31
    End
    Begin VB.Line Line21
        X1          = 64
        X2          = 40
        Y1          = 31
        Y2          = 31
    End
    Begin VB.Line Line20
        X1          = 72
        X2          = 72
        Y1          = 159
        Y2          = 95
    End
    Begin VB.Line Line19
        X1          = 72
        X2          = 72
        Y1          = 95
        Y2          = 31
    End
    Begin VB.Shape Shape8
        FillColor    = &H00FFFFC0&
        FillStyle    = 0 'Solid
        Height       = 255
        Left         = 840
        Shape        = 3 'Circle
        Top          = 1200
        Width        = 375
    End
    Begin VB.Shape Shape7
        FillColor    = &H00FFFFC0&
        FillStyle    = 0 'Solid
        Height       = 255
        Left         = 840
        Shape        = 3 'Circle
        Top          = 2160
        Width        = 375
    End
    Begin VB.Line Line18
        X1          = 72
        X2          = 40
        Y1          = 23
        Y2          = 23
    End
    Begin VB.Line Line15
        X1          = 40
        X2          = 40
        Y1          = 31
        Y2          = 23
    End
    Begin VB.Shape Shape6
        FillColor    = &H00FFFFC0&
        FillStyle    = 0 'Solid
        Height       = 255

```

```

        Left      = 840
        Shape     = 3 'Circle
        Top       = 240
        Width     = 375
    End
End
Begin VB.HScrollBar HScroll1
    Height = 255
    Left = 6120
    Max = 10
    TabIndex = 30
    Top = 4200
    Width = 3855
End
Begin VB.PictureBox Picture2
    Height = 2775
    Left = 3360
    MousePointer = 13 'Custom
    ScaleHeight = -183
    ScaleMode = 0 'User
    ScaleTop = 183
    ScaleWidth = 135
    TabIndex = 29
    Top = 1680
    Width = 2055
    Begin VB.Line Line14
        X1 = 88
        X2 = 88
        Y1 = 23
        Y2 = 31
    End
    Begin VB.Line Line12
        X1 = 48
        X2 = 48
        Y1 = 23
        Y2 = 31
    End
    Begin VB.Line Line11
        X1 = 48
        X2 = 88
        Y1 = 23
        Y2 = 23
    End
    End
    Begin VB.Line Line2
        X1 = 48
        X2 = 88
        Y1 = 31
        Y2 = 31
    End
    End
    Begin VB.Line Line17
        X1 = 48
        X2 = 112
        Y1 = 159
        Y2 = 159
    End
    End
    Begin VB.Line Line16
        X1 = 64
        X2 = 64
        Y1 = 159
        Y2 = 31
    End
    End
    Begin VB.Line Line13
        X1 = 72
        X2 = 72
        Y1 = 159
        Y2 = 31
    End
    End
End

```

```

Begin VB.Shape Shape5
    FillColor      = &H00FFFFC0&
    FillStyle      = 0 'Solid
    Height         = 255
    Left           = 840
    Shape          = 3 'Circle
    Top            = 2040
    Width          = 375
End
Begin VB.Shape Shape3
    FillColor      = &H00FFFFC0&
    FillStyle      = 0 'Solid
    Height         = 255
    Left           = 840
    Shape          = 3 'Circle
    Top            = 240
    Width          = 375
End
End
Begin VB.CommandButton Command2
    Caption        = "Stop"
    BeginProperty Font
        name        = "Arial"
        charset     = 1
        weight      = 400
        size        = 14.25
        underline   = 0 'False
        italic      = 0 'False
        strikethrough = 0 'False
    EndProperty
    Height         = 495
    Left           = 7320
    TabIndex       = 5
    Top            = 4920
    Width          = 1215
End
Begin VB.CommandButton Command1
    Caption        = "Action"
    BeginProperty Font
        name        = "Arial"
        charset     = 1
        weight      = 400
        size        = 14.25
        underline   = 0 'False
        italic      = 0 'False
        strikethrough = 0 'False
    EndProperty
    Height         = 495
    Left           = 5760
    TabIndex       = 4
    Top            = 4920
    Width          = 1215
End
Begin VB.PictureBox Picture1
    Height         = 2775
    Left           = 1200
    MousePointer   = 13 'Custom
    ScaleHeight    = -183
    ScaleMode      = 0 'User
    ScaleTop       = 183
    ScaleWidth     = 135
    TabIndex       = 14
    Top            = 1680
    Width          = 2055
End
Begin VB.Line Line10
    X1             = 40
    X2             = 40

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

Y1          = 31
Y2          = 23
End
Begin VB.Line Line9
X1          = 72
X2          = 40
Y1          = 23
Y2          = 23
End
Begin VB.Line Line8
X1          = 72
X2          = 72
Y1          = 95
Y2          = 31
End
Begin VB.Line Line7
X1          = 72
X2          = 72
Y1          = 159
Y2          = 95
End
Begin VB.Line Line6
X1          = 64
X2          = 40
Y1          = 31
Y2          = 31
End
Begin VB.Line Line5
X1          = 64
X2          = 64
Y1          = 95
Y2          = 31
End
Begin VB.Line Line4
X1          = 64
X2          = 64
Y1          = 159
Y2          = 95
End
Begin VB.Line Line3
X1          = 48
X2          = 88
Y1          = 159
Y2          = 159
End
Begin VB.Shape Shape4
FillColor   = &H00FFFFC0&
FillStyle   = 0 'Solid
Height      = 255
Left        = 840
Shape       = 3 'Circle
Top         = 240
Width       = 375
End
Begin VB.Shape Shape1
FillColor   = &H00FFFFC0&
FillStyle   = 0 'Solid
Height      = 255
Left        = 840
Shape       = 3 'Circle
Top         = 1200
Width       = 375
End
Begin VB.Shape Shape2
FillColor   = &H00FFFFC0&
FillStyle   = 0 'Solid
Height      = 255

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

    Left      = 840
    Shape     = 3 'Circle
    Top       = 2160
    Width     = 375
End
End
Begin MSCommLib.MSComm MSComm1
    Left      = 7680
    Top       = 5880
    _version  = 65536
    _extentx  = 847
    _extenty  = 847
    _stockprops = 0
    cdtimeout = 0
    commport  = 1
    ctsttimeout = 0
    dsrtimeout = 0
    dtrenable = -1 'True
    handshaking = 0
    inbuffersize = 1024
    inputlen  = 0
    interval  = 1000
    nulldiscard = 0 'False
    outbuffersize = 512
    parityreplace = "?"
    rthreshold = 0
    rtstenable = 0 'False
    settings  = "9600,N,8,1"
    sthreshold = 0
End
Begin MSComDlg.CommonDialog Comon2
    Left      = 7200
    Top       = 5880
    _version  = 65536
    _extentx  = 847
    _extenty  = 847
    _stockprops = 0
    cancelerror = -1 'True
    defaulttext = ".pda"
    dialogtitle = "Open Animation"
    filter      = "Animation Files (*.pda)|*.pda"
    flags       = 6148
End
Begin MSComDlg.CommonDialog Comon1
    Left      = 6720
    Top       = 5880
    _version  = 65536
    _extentx  = 847
    _extenty  = 847
    _stockprops = 0
    cancelerror = -1 'True
    defaulttext = ".pda"
    dialogtitle = "Save Animation"
    filter      = "Animation Files (*.pda)|*.pda"
    flags       = 43014
End
Begin VB.Label Label20
    Alignment  = 2 'Center
    BackColor  = &H00FF0000&
    BorderStyle = 1 'Fixed Single
    Caption    = "Loop"
    ForeColor  = &H00FFFFFF&
    Height     = 495
    Left       = 7440
    TabIndex   = 53
    Top        = 3960
    Width      = 1215

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

End
Begin VB.Label Label43
    Alignment       = 2    'Center
    BackStyle       = 0    'Transparent
    Caption         = "Left"
    BeginProperty Font
        name        = "Arial"
        charset     = 1
        weight      = 700
        size        = 14.25
        underline   = 0    'False
        italic      = 0    'False
        strikethrough = 0    'False
    EndProperty
    Height          = 375
    Left            = 0
    TabIndex        = 47
    Top             = 4560
    Width           = 1215
End
Begin VB.Label Label42
    Alignment       = 2    'Center
    BackStyle       = 0    'Transparent
    Caption         = "Right"
    BeginProperty Font
        name        = "Arial"
        charset     = 1
        weight      = 700
        size        = 14.25
        underline   = 0    'False
        italic      = 0    'False
        strikethrough = 0    'False
    EndProperty
    Height          = 375
    Left            = 0
    TabIndex        = 46
    Top             = 1680
    Width           = 1215
End
Begin VB.Label Label41
    Caption         = "0"
    Height          = 255
    Left            = 8640
    TabIndex        = 43
    Top             = 3360
    Width           = 1215
End
Begin VB.Label Label40
    Caption         = "0"
    Height          = 255
    Left            = 8640
    TabIndex        = 42
    Top             = 3000
    Width           = 1215
End
Begin VB.Label Label39
    Caption         = "0"
    Height          = 255
    Left            = 8640
    TabIndex        = 41
    Top             = 2640
    Width           = 1215
End
Begin VB.Label Label38
    Caption         = "0"
    Height          = 255
    Left            = 8640

```

```

    TabIndex      = 40
    Top           = 2280
    Width        = 1215
End
Begin VB.Label Label137
    Caption       = "0"
    Height        = 255
    Left          = 8640
    TabIndex      = 39
    Top           = 1920
    Width         = 1215
End
Begin VB.Label Label136
    Alignment     = 1 'Right Justify
    BackColor     = &H00FFFFFF&
    BackStyle     = 0 'Transparent
    Caption       = "Servo 10"
    ForeColor     = &H00FFFFFF&
    Height        = 255
    Left          = 7800
    TabIndex      = 38
    Top           = 3360
    Width         = 735
End
Begin VB.Label Label135
    Alignment     = 1 'Right Justify
    BackStyle     = 0 'Transparent
    Caption       = "Servo 9"
    ForeColor     = &H00FFFFFF&
    Height        = 255
    Left          = 7800
    TabIndex      = 37
    Top           = 3000
    Width         = 735
End
Begin VB.Label Label134
    Alignment     = 1 'Right Justify
    BackStyle     = 0 'Transparent
    Caption       = "Servo 8"
    ForeColor     = &H00FFFFFF&
    Height        = 255
    Left          = 7800
    TabIndex      = 36
    Top           = 2640
    Width         = 735
End
Begin VB.Label Label133
    Alignment     = 1 'Right Justify
    BackStyle     = 0 'Transparent
    Caption       = "Servo 7"
    ForeColor     = &H00FFFFFF&
    Height        = 255
    Left          = 7800
    TabIndex      = 35
    Top           = 2280
    Width         = 735
End
Begin VB.Label Label132
    Alignment     = 1 'Right Justify
    BackStyle     = 0 'Transparent
    Caption       = "Servo 6"
    ForeColor     = &H00FFFFFF&
    Height        = 255
    Left          = 7800
    TabIndex      = 34
    Top           = 1920
    Width         = 735

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

End
Begin VB.Label Label31
    Alignment       = 2 'Center
    BackStyle       = 0 'Transparent
    Caption         = "Time Line"
    ForeColor       = &H00FFFFFF&
    Height          = 255
    Left            = 7800
    TabIndex        = 33
    Top             = 4560
    Width           = 735
End
Begin VB.Label Label30
    Alignment       = 1 'Right Justify
    BackStyle       = 0 'Transparent
    Caption         = "End"
    ForeColor       = &H00FFFFFF&
    Height          = 255
    Left            = 9240
    TabIndex        = 32
    Top             = 4560
    Width           = 735
End
Begin VB.Label Label29
    BackStyle       = 0 'Transparent
    Caption         = "Start"
    ForeColor       = &H00FFFFFF&
    Height          = 255
    Left            = 6120
    TabIndex        = 31
    Top             = 4560
    Width           = 735
End
Begin VB.Label Label28
    BackStyle       = 0 'Transparent
    Caption         = "Show Time"
    ForeColor       = &H00FFFFFF&
    Height          = 255
    Left            = 3120
    TabIndex        = 28
    Top             = 840
    Width           = 855
End
Begin VB.Label Label27
    Alignment       = 1 'Right Justify
    BackStyle       = 0 'Transparent
    Caption         = "Time : "
    ForeColor       = &H00FFFFFF&
    Height          = 255
    Left            = 3240
    TabIndex        = 27
    Top             = 480
    Width           = 735
End
Begin VB.Label Label26
    BackStyle       = 0 'Transparent
    Caption         = "Filename:"
    ForeColor       = &H00FFFFFF&
    Height          = 255
    Left            = 3240
    TabIndex        = 26
    Top             = 120
    Width           = 735
End
Begin VB.Label Label25
    Alignment       = 1 'Right Justify
    BackColor       = &H00FFFFFF&

```

```

BackStyle      = 0 'Transparent
Caption        = "Servo 5"
ForeColor      = &H00FFFFFF&
Height         = 255
Left           = 5520
TabIndex       = 25
Top            = 3360
Width          = 735
End
Begin VB.Label Label24
Alignment      = 1 'Right Justify
BackStyle      = 0 'Transparent
Caption        = "Servo 4"
ForeColor      = &H00FFFFFF&
Height         = 255
Left           = 5520
TabIndex       = 24
Top            = 3000
Width          = 735
End
Begin VB.Label Label23
Alignment      = 1 'Right Justify
BackStyle      = 0 'Transparent
Caption        = "Servo 3"
ForeColor      = &H00FFFFFF&
Height         = 255
Left           = 5520
TabIndex       = 23
Top            = 2640
Width          = 735
End
Begin VB.Label Label22
Alignment      = 1 'Right Justify
BackStyle      = 0 'Transparent
Caption        = "Servo 2"
ForeColor      = &H00FFFFFF&
Height         = 255
Left           = 5520
TabIndex       = 22
Top            = 2280
Width          = 735
End
Begin VB.Label Label21
Alignment      = 1 'Right Justify
BackStyle      = 0 'Transparent
Caption        = "Servo 1"
ForeColor      = &H00FFFFFF&
Height         = 255
Left           = 5520
TabIndex       = 21
Top            = 1920
Width          = 735
End
Begin VB.Label Label19
Caption        = "0"
Height         = 255
Left           = 4080
TabIndex       = 20
Top            = 480
Width          = 1215
End
Begin VB.Label Label17
Caption        = "0"
Height         = 255
Left           = 6360
TabIndex       = 19
Top            = 3360

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

        Width          = 1215
End
Begin VB.Label Label16
    Caption          = "0"
    Height           = 255
    Left              = 6360
    TabIndex          = 18
    Top               = 3000
    Width             = 1215
End
Begin VB.Label Label15
    Caption          = "0"
    Height           = 255
    Left              = 6360
    TabIndex          = 17
    Top               = 2640
    Width             = 1215
End
Begin VB.Label Label14
    Caption          = "0"
    Height           = 255
    Left              = 6360
    TabIndex          = 16
    Top               = 2280
    Width             = 1215
End
Begin VB.Label Label13
    Caption          = "0"
    Height           = 255
    Left              = 6360
    TabIndex          = 15
    Top               = 1920
    Width             = 1215
End
Begin VB.Label Label12
    Caption          = "0"
    Height           = 255
    Left              = 1440
    TabIndex          = 13
    Top               = 840
    Width             = 1215
End
Begin VB.Label Label11
    Alignment         = 1 'Right Justify
    BackStyle         = 0 'Transparent
    Caption           = "Degree Absolute"
    ForeColor         = &H00FFFFFF&
    Height            = 255
    Left              = 120
    TabIndex          = 12
    Top               = 840
    Width             = 1215
End
Begin VB.Label Label10
    Alignment         = 1 'Right Justify
    BackStyle         = 0 'Transparent
    Caption           = "Robot Part"
    ForeColor         = &H00FFFFFF&
    Height            = 255
    Left              = 120
    TabIndex          = 11
    Top               = 1200
    Width             = 1215
End
Begin VB.Label Label9
    Alignment         = 1 'Right Justify
    BackStyle         = 0 'Transparent

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

Caption      = "Y - Position"
ForeColor    = &H00FFFFFF&
Height       = 255
Left         = 120
TabIndex     = 10
Top          = 480
Width        = 1215
End
Begin VB.Label Label8
Alignment    = 1 'Right Justify
BackStyle    = 0 'Transparent
Caption      = "X - Position"
ForeColor    = &H00FFFFFF&
Height       = 255
Left         = 120
TabIndex     = 9
Top          = 120
Width        = 1215
End
Begin VB.Label Label7
Caption      = "none"
Height       = 255
Left         = 1440
TabIndex     = 8
Top          = 1200
Width        = 1215
End
Begin VB.Label Label6
Caption      = "0"
Height       = 255
Left         = 1440
TabIndex     = 7
Top          = 480
Width        = 1215
End
Begin VB.Label Label5
Caption      = "0"
Height       = 255
Left         = 1440
TabIndex     = 6
Top          = 120
Width        = 1215
End
Begin VB.Label Label4
BackStyle    = 0 'Transparent
Caption      = "May 1996 Release"
BeginProperty Font
    name      = "Times New Roman"
    charset   = 1
    weight    = 700
    size      = 12
    underline  = 0 'False
    italic     = 0 'False
    strikethrough = 0 'False
EndProperty
Height       = 375
Left         = 7800
TabIndex     = 3
Top          = 1320
Width        = 2055
End
Begin VB.Label Label3
BackStyle    = 0 'Transparent
Caption      = "Copyright (c) 1996 Noels Corp."
BeginProperty Font
    name      = "Times New Roman"
    charset   = 1

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

        weight      = 700
        size        = 12
        underline   = 0 'False
        italic      = 0 'False
        strikethrough = 0 'False
    EndProperty
    Height          = 255
    Left            = 6720
    TabIndex        = 2
    Top             = 120
    Width           = 3135
End
Begin VB.Line Line1
    X1              = 500
    X2              = 500
    Y1              = 10969.866
    Y2              = 10970.866
End
Begin VB.Label Label2
    BackStyle       = 0 'Transparent
    Caption         = "Double Arm Version."
    BeginProperty Font
        name        = "Arial"
        charset     = 1
        weight      = 700
        size        = 9.75
        underline   = 0 'False
        italic      = 0 'False
        strikethrough = 0 'False
    EndProperty
    Height          = 255
    Left            = 7800
    TabIndex        = 1
    Top             = 1080
    Width           = 2055
End
Begin VB.Label Label1
    BackStyle       = 0 'Transparent
    Caption         = "Johnny Director"
    BeginProperty Font
        name        = "Arial"
        charset     = 1
        weight      = 700
        size        = 21.75
        underline   = 0 'False
        italic      = -1 'True
        strikethrough = 0 'False
    EndProperty
    Height          = 495
    Left            = 6480
    TabIndex        = 0
    Top             = 480
    Width           = 3495
End
Begin VB.Menu menu1
    Caption         = "&Manager"
    Begin VB.Menu menu2
        Caption     = "N&ew Action"
    End
    Begin VB.Menu menu3
        Caption     = "O&pen Action"
    End
    Begin VB.Menu menu4
        Caption     = "&Save Action"
    End
End
Begin VB.Menu menu10

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

Caption          = "Option"
Begin VB.Menu menu11
Caption          = "Cycle Action"
End
Begin VB.Menu menu12
Caption          = "Communication Setting"
End
End
Begin VB.Menu menu13
Caption          = "&About"
End
End
Attribute VB_Name = "Form1"
Attribute VB_Creatable = False
Attribute VB_Exposed = False
'Defind error deviation from line,user still catch the arm

Const errormax = 5
Const errordeg = 90
Const maxframe = 15 'Related to the range property in slide bar of Timeline.

Public mousenable As Boolean
Public filename1 As String
Dim Shared deg4(maxframe) As Single, deg5(maxframe) As Single, deg6(maxframe) As Single
Dim Shared deg21(maxframe) As Single, deg22(maxframe) As Single, deg23(maxframe) As Single
Dim Shared deg16(maxframe) As Single, deg30(maxframe) As Single
Public linetouch As Integer
Public finished As Boolean, delayfinished As Boolean, breakaction As Boolean
Private Type degrec
d1 As Single
d2 As Single
d3 As Single
d4 As Single
d5 As Single
d6 As Single
d7 As Single
d8 As Single
End Type

Private Sub Command1_Click()
delaytime = (Val(Text2.Text) / maxframe) * 1000
Timer1.Interval = Int(delaytime)
finished = False
breakaction = False
'Action the Starting Process
For frm = 0 To Int(maxframe / 3)
HScroll1.Value = frm
Label19.Caption = frm
LegUpdate1
LegUpdate2
LegUpdate3
LegUpdate4
SendFrame (frm)
delay
If (breakaction = True) Then Exit For
If (finished = True) Then Exit For
Next
If breakaction = True Then Exit Sub
'Action the Walking Process
Do
For frm = Int(maxframe / 3) To Int(maxframe * (2 / 3))
HScroll1.Value = frm
Label19.Caption = frm
LegUpdate1
LegUpdate2
LegUpdate3
LegUpdate4

```

```

        SendFrame (frm)
        delay
        If (breakaction = True) Then Exit For
        'If (finished = True) Then Exit For
    Next
Loop Until (finished = True Or menu1.Checked = False)
If breakaction = True Then Exit Sub

'Action the Stopping Process
For frm = Int(maxframe * (2 / 3)) To maxframe
    HScroll1.Value = frm
    Label19.Caption = frm
    LegUpdate1
    LegUpdate2
    LegUpdate3
    LegUpdate4
    SendFrame (frm)
    delay
    If (breakaction = True) Then Exit For
Next
If breakaction = True Then Exit Sub
HScroll1.Value = 0
End Sub
Private Sub delay()
    Timer1.Enabled = True
    delayfinished = False
    Do While (delayfinished = False)
        DoEvents ' Yield to other processes.
    Loop
    Timer1.Enabled = False
End Sub

Private Sub Command2_Click()
    finished = True
End Sub

Private Sub Command3_Click()
    deg4(HScroll1.Value) = deg4(VScroll1.Value)
    deg5(HScroll1.Value) = deg5(VScroll1.Value)
    deg6(HScroll1.Value) = deg6(VScroll1.Value)
    deg23(HScroll1.Value) = deg23(VScroll1.Value)
    deg22(HScroll1.Value) = deg22(VScroll1.Value)
    deg21(HScroll1.Value) = deg21(VScroll1.Value)
    deg16(HScroll1.Value) = deg16(VScroll1.Value)
    deg30(HScroll1.Value) = deg30(VScroll1.Value)
    LegUpdate1
    LegUpdate2
    LegUpdate3
    LegUpdate4
End Sub

Private Sub Command4_Click()
    breakaction = True
End Sub

Private Sub Command5_Click()
    SendFrame (HScroll1.Value)
End Sub

Private Sub Form_Load()

    'Set the Time Line Range Related to the maxframe.
    HScroll1.Max = maxframe
    VScroll1.Min = maxframe ' To reverse Scrolling direction
    'Setting The Pre-Running Value of All Degree.
    Timer1.Enabled = False

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
menu2 Click
Unload Form2
```

```
End Sub
```

```
Private Sub HScroll11_Change()
    Label19.Caption = HScroll11.Value
    LegUpdate1
    LegUpdate2
    LegUpdate3
    LegUpdate4
End Sub
```

```
End Sub
```

```
Private Sub menu11_Click()
    menu11.Checked = Not (menu11.Checked)
End Sub
```

```
Private Sub menu12_Click()
    Form1.Enabled = False
    Form2.Visible = True
    While (Form1.Enabled = False)
        DoEvents
    Wend
    Unload Form2
End Sub
```

```
Private Sub menu2_Click()
    For frm = 0 To maxframe
        deg4(frm) = 270
        deg5(frm) = 270
        deg6(frm) = 180
        deg23(frm) = 270
        deg22(frm) = 270
        deg21(frm) = 180
        deg16(frm) = 270
        deg30(frm) = 270
    Next frm
    HScroll11.Value = 0
    Text3.Text = "NONAME.PDA"
    LegUpdate1
    LegUpdate2
    LegUpdate3
    LegUpdate4
End Sub
```

```
Private Sub menu3_Click()
    Dim rectemp As degrec
    On Error Resume Next
    Comon2.Action = 1
```

```
If (Err.Number = 32755) Then
    Err.Clear
    Beep
```

```
Else
    Text3.Text = Comon2.FileTitle
    Open Comon2.FileName For Random As 2 Len = 32
    For i = 0 To maxframe
        Get #2, i + 1, rectemp
        deg4(i) = rectemp.d1
        deg5(i) = rectemp.d2
        deg6(i) = rectemp.d3
        deg21(i) = rectemp.d4
        deg22(i) = rectemp.d5
        deg23(i) = rectemp.d6
        deg16(i) = rectemp.d7
        deg30(i) = rectemp.d8
    Next i
End Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

Next
Close #2
HScroll11.Value = 0
LegUpdate1
LegUpdate2
LegUpdate3
LegUpdate4
End If
On Error GoTo 0
End Sub

Private Sub menu4_Click()
Dim rectemp As degrec

Comon1.FileName = .Text3.Text
On Error Resume Next
Comon1.Action = 2
If (Err.Number = 32755) Then
    Err.Clear
    Beep
Else
    Text3.Text = Comon1.FileTitle
    Open Comon1.FileName For Random As 2 Len = 32
    For i = 0 To maxframe
        rectemp.d1 = deg4(i)
        rectemp.d2 = deg5(i)
        rectemp.d3 = deg6(i)
        rectemp.d4 = deg21(i)
        rectemp.d5 = deg22(i)
        rectemp.d6 = deg23(i)
        rectemp.d7 = deg16(i)
        rectemp.d8 = deg30(i)
        Put #2, i + 1, rectemp
    Next
End If
On Error GoTo 0
Close #2
End Sub

Private Sub Picture1_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
Label5.Caption = X
Label6.Caption = Y
'Determine Which part is being moved.

'Start with part#1;Upper leg.
mousenable = True 'Enable MouseMove Function to Processing
line4touche = False 'Line#4 Touch Checker of All Point
line5touche = False ' #5
line6touche = False ' #6
'Determine Slope
dy4 = Int(Line4.Y1 - Line4.Y2)
dx4 = Int(Line4.X1 - Line4.X2)
If (Line4.X1 > Line4.X2) Then
    minx4 = Line4.X2
    maxx4 = Line4.X1
Else
    minx4 = Line4.X1
    maxx4 = Line4.X2
End If
If (Line4.Y1 > Line4.Y2) Then
    miny4 = Line4.Y2
    maxy4 = Line4.Y1
Else
    miny4 = Line4.Y1
    maxy4 = Line4.Y2
End If

```

```

For dx = -errormax To errormax
For dy = -errormax To errormax
    line4touch = False ' Line #4 Touch Checker of Single Loop

'X-range Check
    If (X + dx) >= minx4 And (X + dx) <= maxx4 Then
        line4touch = True
    Else
        line4touch = False
    End If
'Y-range Check
    If (Y + dy) >= miny4 And (Y + dy) <= maxy4 Then
        line4touch = line4touch And True
    Else
        line4touch = False
    End If

    line4touche = line4touche Or line4touch
Next dy
Next dx

'Distance Check
    d = Abs((dx4 * -(Y - Line4.Y1)) + (dy4 * (X - Line4.X1))) / Sqr(dx4 ^ 2 + dy4 ^ 2)
    Label16.Caption = d
    If d < errormax Then
        line4touche = line4touche And True
    Else
        line4touche = False
    End If

'continue with lower leg
If Not (line4touche) Then
    dy5 = Int(Line5.Y1 - Line5.Y2)
    dx5 = Int(Line5.X1 - Line5.X2)
    If (Line5.X1 > Line5.X2) Then
        minx5 = Line5.X2
        maxx5 = Line5.X1
    Else
        minx5 = Line5.X1
        maxx5 = Line5.X2
    End If
    If (Line5.Y1 > Line5.Y2) Then
        miny5 = Line5.Y2
        maxy5 = Line5.Y1
    Else
        miny5 = Line5.Y1
        maxy5 = Line5.Y2
    End If

For dx = -errormax To errormax
For dy = -errormax To errormax
    line5touch = False
    'X-range Check
    If (X + dx) >= minx5 And (X + dx) <= maxx5 Then
        line5touch = True
    Else
        line5touch = False
    End If
    'Y-range Check
    If (Y + dy) >= miny5 And (Y + dy) <= maxy5 Then
        line5touch = line5touch And True
    Else
        line5touch = False
    End If

```

```

        line5touche = line5touche Or line5touch
    Next dy
    Next dx
    d = Abs((dx5 * -(Y - Line5.Y1)) + (dy5 * (X - Line5.X1))) / Sqr(dx5 ^ 2 + dy5 ^ 2)
    Label16.Caption = d
    If d < errormax Then
        line5touche = line5touche And True
    Else
        line5touche = False
    End If
End If

'and the foot
If Not (line4touche Or line5touche) Then
    dy6 = Int(Line6.Y1 - Line6.Y2)
    dx6 = Int(Line6.X1 - Line6.X2)
    If (Line6.X1 > Line6.X2) Then
        minx6 = Line6.X2
        maxx6 = Line6.X1
    Else
        minx6 = Line6.X1
        maxx6 = Line6.X2
    End If
    If (Line6.Y1 > Line6.Y2) Then
        miny6 = Line6.Y2
        maxy6 = Line6.Y1
    Else
        miny6 = Line6.Y1
        maxy6 = Line6.Y2
    End If

    For dx = -errormax To errormax
    For dy = -errormax To errormax
        line6touch = False
        'X-range Check
        If (X + dx) >= minx6 And (X + dx) <= maxx6 Then
            line6touch = True
        Else
            line6touch = False
        End If
        'Y-range Check
        If (Y + dy) >= miny6 And (Y + dy) <= maxy6 Then
            line6touch = line6touch And True
        Else
            line6touch = False
        End If
        line6touche = line6touche Or line6touch
    Next dy
    Next dx
    d = Abs((dx6 * -(Y - Line6.Y1)) + (dy6 * (X - Line6.X1))) / Sqr(dx6 ^ 2 + dy6 ^ 2)
    Label16.Caption = d
    If d < errormax Then
        line6touche = line6touche And True
    Else
        line6touche = False
    End If
End If

'Display and Set Variable for MouseMove Method
If line4touche Then linetouch = 4
If line5touche Then linetouch = 5
If line6touche Then linetouch = 6
If Not (line4touche Or line5touche Or line6touche) Then Label7.Caption = "None": linetouc

End Sub

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

Private Sub Picture1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
Static olddeg4
Static olddeg5
'Private deg4x As Single, deg5x As Single, deg6x As Single
'Check Whether Mouse button still pressed or not
If (mousenable) Then
    Label5.Caption = X
    Label6.Caption = Y
    'Port Value from Motion Frame Table
    deg4x = deg4(HScroll11.Value)
    deg5x = deg5(HScroll11.Value)
    deg6x = deg6(HScroll11.Value)

    'Case Moving Upperleg. (deg4) static
    If (linetouch = 4) Then
        Label7.Caption = "Upper leg Right"
        dy4 = (Y - Line4.Y1)
        dx4 = (X - Line4.X1)
        olddeg4 = deg4x
        If dx4 <> 0 Then
            deg4x = ((180 / (22 / 7)) * Atn(dy4 / dx4))
        Else
            deg4x = 0
        End If
        'Process degree to reference system.
        If X > Line4.X1 Then 'Positive Quadrant
            deg4x = 90 + deg4x
            If (Abs(deg4x) > errordeg) Then deg4x = errordeg
        End If
        If X < Line4.X1 Then 'Negative Quadrant
            deg4x = deg4x - 90
            If (Abs(deg4x) > errordeg) Then deg4x = -errordeg
        End If
        deg4x = deg4x + 270
        'Label13.Caption = deg4x - 270
        Label12.Caption = deg4x
        'Change degree of other part due to upper part change
        deg5x = deg5x + (deg4x - olddeg4)
        deg6x = deg6x + (deg4x - olddeg4)
    End If

    'Case Moving Lowerleg; Line#5 (deg5)
    If (linetouch = 5) Then
        Label7.Caption = "Lower leg Right"
        dy5 = (Y - Line5.Y1)
        dx5 = (X - Line5.X1)
        olddeg5 = deg5x
        If dx5 <> 0 Then
            deg5x = ((180 / (22 / 7)) * Atn(dy5 / dx5))
        Else
            deg5x = 0
        End If
        'Process degree to reference system.
        If X > Line5.X1 Then 'Positive Quadrant
            deg5x = 90 + deg5x
        End If
        If X < Line5.X1 Then 'Negative Quadrant
            deg5x = deg5x - 90
        End If
        'Range Check
        deg5x = deg5x + 270
        If Abs(deg5x - deg4x) > errordeg Then
            If (deg5x > deg4x) Then
                deg5x = deg4x + errordeg
            Else
                deg5x = deg4x - errordeg
            End If
        End If
    End If
End Sub

```

```

        End If
    End If

    'degtemp = deg5      ' Shift to it's real position
    Label12.Caption = deg5x      'Absolute to origin
    'Label14.Caption = deg5x - deg4x      'Relative to prior link
    deg6x = deg5x + (deg5x - olddeg5)
End If
'Case Moving foot; Line#6 (deg6)
If (linetouch = 6) Then
    Label7.Caption = "Foot Right"
    dy6 = (Y - Line6.Y1)
    dx6 = (X - Line6.X1)
    If dx6 <> 0 Then
        deg6x = ((180 / (22 / 7)) * Atn(dy6 / dx6))
    Else
        deg6x = 0
    End If
End If

'Process degree to reference system.
If X > Line6.X1 Then 'Positive Quadrant
    deg6x = 90 + deg6x
End If
If X < Line6.X1 Then 'Negative Quadrant
    deg6x = deg6x - 90
End If
deg6x = deg6x + 270 'Convert to Real Math Co-or
'Range Check of Line 6
If Abs((deg6x + 90) - deg5x) > errordeg Then
    If ((deg6x + 90) > deg5x) Then
        deg6x = deg5x + errordeg - 90
    Else
        deg6x = deg5x - errordeg - 90
    End If
End If
' Label15.Caption = deg6x - deg5x + 90 'Relative to prior link
Label12.Caption = deg6x      'Absolute degree to origin
End If
'Remapping back to the Frame Table
deg4(HScroll11.Value) = deg4x
deg5(HScroll11.Value) = deg5x
deg6(HScroll11.Value) = deg6x

'Drawing Entire leg due to movement.
If Not (linetouch = 0) Then
    LegUpdate1
End If
End If
End Sub

Private Sub Picture1_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    mousenable = False
End Sub

Private Sub Picture2_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
    Label5.Caption = X
    Label6.Caption = Y
    'Determine Which part is being moved.

    'Start with part#1;Upper leg.
    mousenable = True      'Enable MouseMove Function to Processing
    line16touche = False 'Line#16 Touch Checker of All Point
    'Determine Slope
    dy16 = Int(Line16.Y1 - Line16.Y2)
    dx16 = Int(Line16.X1 - Line16.X2)
    If (Line16.X1 > Line16.X2) Then

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

minx16 = Line16.X2
maxx16 = Line16.X1
Else
minx16 = Line16.X1
maxx16 = Line16.X2
End If
If (Line16.Y1 > Line16.Y2) Then
miny16 = Line16.Y2
maxy16 = Line16.Y1
Else
miny16 = Line16.Y1
maxy16 = Line16.Y2
End If

For dx = -errormax To errormax
For dy = -errormax To errormax
line16touch = False ' Line #16 Touch Checker of Single Loop

'X-range Check
If (X + dx) >= minx16 And (X + dx) <= maxx16 Then
line16touch = True
Else
line16touch = False
End If

'Y-range Check
If (Y + dy) >= miny16 And (Y + dy) <= maxy16 Then
line16touch = line16touch And True
Else
line16touch = False
End If

line16touche = line16touche Or line16touch
Next dy
Next dx

'Distance Check
d = Abs((dx16 * -(Y - Line16.Y1)) + (dy16 * (X - Line16.X1))) / Sqr(dx16 ^ 2 + dy16 ^ 2)
Label16.Caption = d
If d < errormax Then
line16touche = line16touche And True
Else
line16touche = False
End If

'Display and Set Variable for MouseMove Method
If line16touche Then linetouch = 16
If Not (line16touche) Then Label7.Caption = "None": linetouch = 0

End Sub

Private Sub Picture2_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
'Check Whether Mouse_button still pressed or not
Private deg16x As Single
If (mousenable) Then
Label5.Caption = X
Label6.Caption = Y

' Mapping from Frame Table
deg16x = deg16(HScroll11.Value)
'Case Moving wrist. (deg16) static
If (linetouch = 16) Then
Label7.Caption = "Wrist Right"
dy16 = (Y - Line16.Y1)
dx16 = (X - Line16.X1)
If dx16 <> 0 Then
deg16x = ((180 / (22 / 7)) * Atn(dy16 / dx16))
Else

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

        deg16x = 0
    End If
    'Process degree to reference system..
    If X > Line16.X1 Then 'Positive Quadrant
        deg16x = 90 + deg16x
        If (Abs(deg16x) > errordeg) Then deg16x = errordeg
    End If
    If X < Line16.X1 Then 'Negative Quadrant
        deg16x = deg16x - 90
        If (Abs(deg16x) > errordeg) Then deg16x = -errordeg
    End If
    deg16x = deg16x + 270
    'Label16.Caption = deg16x - 270
    'Label17.Caption = -(deg16x - 270)
    Label12.Caption = deg16x
End If
'Remapping back to Frame Table
deg16(HScroll11.Value) = deg16x
'Drawing Entire leg due to movement.
If Not (linetouch = 0) Then
    LegUpdate2
End If
End If

End Sub

Private Sub Picture2_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    mousenable = False
End Sub

Private Sub Picture3_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
    Label5.Caption = X
    Label6.Caption = Y
    'Determine Which part is being moved.

    'Start with part#1;Upper leg.
    mousenable = True 'Enable MouseMove Function to Processing
    line4touche = False 'Line#4 Touch Checker of All Point
    line5touche = False ' #5
    line6touche = False ' #6
    'Determine Slope
    dy4 = Int(Line23.Y1 - Line23.Y2)
    dx4 = Int(Line23.X1 - Line23.X2)
    If (Line23.X1 > Line23.X2) Then
        minx4 = Line23.X2
        maxx4 = Line23.X1
    Else
        minx4 = Line23.X1
        maxx4 = Line23.X2
    End If
    If (Line23.Y1 > Line23.Y2) Then
        miny4 = Line23.Y2
        maxy4 = Line23.Y1
    Else
        miny4 = Line23.Y1
        maxy4 = Line23.Y2
    End If

    For dx = -errormax To errormax
    For dy = -errormax To errormax
        line4touch = False ' Line #4 Touch Checker of Single Loop

    'X-range Check
    If (X + dx) >= minx4 And (X + dx) <= maxx4 Then
        line4touch = True
    Else

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

        line4touch = False
    End If
'Y-range Check
    If (Y + dy) >= miny4 And (Y + dy) <= maxy4 Then
        line4touch = line4touch And True
    Else
        line4touch = False
    End If

    line4touche = line4touche Or line4touch
Next dy
Next dx
'Distance Check
    d = Abs((dx4 * -(Y - Line23.Y1)) + (dy4 * (X - Line23.X1))) / Sqr(dx4 ^ 2 + dy4 ^ 2)
    ' Label16.Caption = d For Debugging only ....
    If d < errormax Then
        line4touche = line4touche And True
    Else
        line4touche = False
    End If

'continue with lower leg
If Not (line4touche) Then
    dy5 = Int(Line22.Y1 - Line22.Y2)
    dx5 = Int(Line22.X1 - Line22.X2)
    If (Line22.X1 > Line22.X2) Then
        minx5 = Line22.X2
        maxx5 = Line22.X1
    Else
        minx5 = Line22.X1
        maxx5 = Line22.X2
    End If
    If (Line22.Y1 > Line22.Y2) Then
        miny5 = Line22.Y2
        maxy5 = Line22.Y1
    Else
        miny5 = Line22.Y1
        maxy5 = Line22.Y2
    End If

    For dx = -errormax To errormax
    For dy = -errormax To errormax
        line5touch = False
        'X-range Check
        If (X + dx) >= minx5 And (X + dx) <= maxx5 Then
            line5touch = True
        Else
            line5touch = False
        End If
        'Y-range Check
        If (Y + dy) >= miny5 And (Y + dy) <= maxy5 Then
            line5touch = line5touch And True
        Else
            line5touch = False
        End If

        line5touche = line5touche Or line5touch
    Next dy
    Next dx
    d = Abs((dx5 * -(Y - Line22.Y1)) + (dy5 * (X - Line22.X1))) / Sqr(dx5 ^ 2 + dy5 ^ 2)
    ' Label16.Caption = d
    If d < errormax Then
        line5touche = line5touche And True
    Else
        line5touche = False

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

End If

End If

'and the foot
If Not (line4touche Or line5touche) Then
    dy6 = Int(Line21.Y1 - Line21.Y2)
    dx6 = Int(Line21.X1 - Line21.X2)
    If (Line21.X1 > Line21.X2) Then
        minx6 = Line21.X2
        maxx6 = Line21.X1
    Else
        minx6 = Line21.X1
        maxx6 = Line21.X2
    End If
    If (Line21.Y1 > Line21.Y2) Then
        miny6 = Line21.Y2
        maxy6 = Line21.Y1
    Else
        miny6 = Line21.Y1
        maxy6 = Line21.Y2
    End If

    For dx = -errormax To errormax
    For dy = -errormax To errormax
        line6touch = False
        'X-range Check
        If (X + dx) >= minx6 And (X + dx) <= maxx6 Then
            line6touch = True
        Else
            line6touch = False
        End If
        'Y-range Check
        If (Y + dy) >= miny6 And (Y + dy) <= maxy6 Then
            line6touch = line6touch And True
        Else
            line6touch = False
        End If
        line6touche = line6touche Or line6touch
    Next dy
    Next dx
    d = Abs((dx6 * -(Y - Line21.Y1)) + (dy6 * (X - Line21.X1))) / Sqr(dx6 ^ 2 + dy6 ^ 2)
    Label16.Caption = d
    If d < errormax Then
        line6touche = line6touche And True
    Else
        line6touche = False
    End If
End If

End If
'Display and Set Variable for MouseMove Method
If line4touche Then linetouch = 4
If line5touche Then linetouch = 5
If line6touche Then linetouch = 6
If Not (line4touche Or line5touche Or line6touche) Then Label7.Caption = "None": linetouc

End Sub

Private Sub Picture3_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
'Private deg21x As Single, deg22x As Single, deg23x As Single

Static olddeg23
Static olddeg22
'Check Whether Mouse button still pressed or not
If (mousenable) Then
    Label5.Caption = X

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

label6.Caption = Y

'Mapping from Frame Table.
deg21x = deg21(HScroll11.Value)
deg22x = deg22(HScroll11.Value)
deg23x = deg23(HScroll11.Value)

'Case Moving Upperleg.Left (deg23) static
If (linetouch = 4) Then
    Label7.Caption = "Upper leg Left"
    dy4 = (Y - Line23.Y1)
    dx4 = (X - Line23.X1)
    olddeg23 = deg23x
    If dx4 <> 0 Then
        deg23x = ((180 / (22 / 7)) * Atn(dy4 / dx4))
    Else
        deg23x = 0
    End If
'Process degree to reference system.
If X > Line23.X1 Then 'Positive Quadrant
    deg23x = 90 + deg23x
    If (Abs(deg23x) > errordeg) Then deg23x = errordeg
End If
If X < Line23.X1 Then 'Negative Quadrant
    deg23x = deg23x - 90
    If (Abs(deg23x) > errordeg) Then deg23x = -errordeg
End If
deg23x = deg23x + 270
' Label37.Caption = deg23x - 270
Label12.Caption = deg23x
'Change degree of other part due to upper part change
deg22x = deg22x + (deg23x - olddeg23)
deg21x = deg21x + (deg23x - olddeg23)
End If

'Case Moving Lowerleg.Left Line#23 (deg23)
If (linetouch = 5) Then
    Label7.Caption = "Lower leg Left"
    dy5 = (Y - Line22.Y1)
    dx5 = (X - Line22.X1)
    olddeg22 = deg22x
    If dx5 <> 0 Then
        deg22x = ((180 / (22 / 7)) * Atn(dy5 / dx5))
    Else
        deg22x = 0
    End If
'Process degree to reference system.
If X > Line22.X1 Then 'Positive Quadrant
    deg22x = 90 + deg22x
End If
If X < Line22.X1 Then 'Negative Quadrant
    deg22x = deg22x - 90
End If
deg22x = deg22x + 270
'Range Check
If Abs(deg22x - deg23x) > errordeg Then
    If (deg22x > deg23x) Then
        deg22x = deg23x + errordeg
    Else
        deg22x = deg23x - errordeg
    End If
End If

Label12.Caption = deg22x 'Absolute to origin
'Label38.Caption = deg22x - deg23x 'Relative to prior link
deg21x = deg21x + (deg22x - olddeg22)
End If

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

'Case Moving foot.Left Line#21 (deg21)
If (linetouch = 6) Then
    Label7.Caption = "Foot Left"
    dy6 = (Y - Line21.Y1)
    dx6 = (X - Line21.X1)
    If dx6 <> 0 Then
        deg21x = ((180 / (22 / 7)) * Atn(dy6 / dx6))
    Else
        deg21x = 0
    End If

'Process degree to reference system.
If X > Line21.X1 Then 'Positive Quadrant
    deg21x = 90 + deg21x
End If
If X < Line21.X1 Then 'Negative Quadrant
    deg21x = deg21x - 90
End If
deg21x = deg21x + 270
'Range Check of Line 6
If Abs((deg21x + 90) - deg22x) > errordeg Then
    If ((deg21x + 90) > deg22x) Then
        deg21x = deg22x + errordeg - 90
    Else
        deg21x = deg22x - errordeg - 90
    End If
End If

'Label39.Caption = deg21x - deg22x + 90 'Relative to prior link
Label12.Caption = deg21x 'Absolute degree to origin
End If
'Remapping back to Frame Table
deg21(HScroll11.Value) = deg21x
deg22(HScroll11.Value) = deg22x
deg23(HScroll11.Value) = deg23x

'Drawing Entire leg due to movement.
If Not (linetouch = 0) Then
    LegUpdate3
End If
End If

End Sub

Private Sub Picture3_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    mousenable = False
End Sub

Private Sub Picture4_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
    Label5.Caption = X
    Label6.Caption = Y
    'Determine Which part is being moved.

    'Start with part#1;Upper leg.
    mousenable = True 'Enable MouseMove Function to Processing
    line16touche = False 'Line#16 Touch Checker of All Point
    'Determine Slope
    dy16 = Int(Line30.Y1 - Line30.Y2)
    dx16 = Int(Line30.X1 - Line30.X2)
    If (Line30.X1 > Line30.X2) Then
        minx16 = Line30.X2
        maxx16 = Line30.X1
    Else
        minx16 = Line30.X1
        maxx16 = Line30.X2
    End If
    If (Line30.Y1 > Line30.Y2) Then

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

        miny16 = Line30.Y2
        maxy16 = Line30.Y1
Else
        miny16 = Line30.Y1
        maxy16 = Line30.Y2
End If

For dx = -errormax To errormax
For dy = -errormax To errormax
        line16touch = False ' Line #16 Touch Checker of Single Loop

'X-range Check
        If (X + dx) >= minx16 And (X + dx) <= maxx16 Then
                line16touch = True
        Else
                line16touch = False
        End If

'Y-range Check
        If (Y + dy) >= miny16 And (Y + dy) <= maxy16 Then
                line16touch = line16touch And True
        Else
                line16touch = False
        End If

        line16touche = line16touche Or line16touch
Next dy
Next dx

'Distance Check
        d = Abs((dx16 * -(Y - Line16.Y1)) + (dy16 * (X - Line16.X1))) / Sqr(dx16 ^ 2 + dy16 ^ 2)
        Label16.Caption = d
        If d < errormax Then
                line16touche = line16touche And True
        Else
                line16touche = False
        End If

'Display and Set Variable for MouseMove Method
If line16touche Then linetouch = 16
If Not (line16touche) Then Label7.Caption = "None": linetouch = 0

End Sub

Private Sub Picture4_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
'Private deg30x As Single
'Check Whether Mouse button still pressed or not
If (mousenable) Then
        Label5.Caption = X
        Label6.Caption = Y

        deg30x = deg30(HScroll11.Value)
'Case Moving wrist. (deg16) static
If (linetouch = 16) Then
        Label7.Caption = "Wrist Left"
        dy16 = (Y - Line30.Y1)
        dx16 = (X - Line30.X1)
        If dx16 <> 0 Then
                deg30x = ((180 / (22 / 7)) * Atn(dy16 / dx16))
        Else
                deg30x = 0
        End If
'Process degree to reference system.
        If X > Line30.X1 Then 'Positive Quadrant
                deg30x = 90 + deg30x
                If (Abs(deg30x) > errordeg) Then deg30x = errordeg
        End If
        If X < Line30.X1 Then 'Negative Quadrant

```

```

        deg30x = deg30x - 90
        If (Abs(deg30x) > errordeg) Then deg30x = -errordeg
    End If
    deg30x = deg30x + 270
    ' Label40.Caption = deg30x - 270
    'Label41.Caption = -(deg30x - 270)
    Label12.Caption = deg30x
End If
'Remapping back to frame table
deg30(HScroll11.Value) = deg30x
'Drawing Entire leg due to movement.
If Not (linetouch = 0) Then
    LegUpdate4
End If
End If

End Sub

Private Sub Picture4_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    mousenable = False
End Sub

Private Sub LegUpdate1()
    Static olddeg4 As Single, olddeg5 As Single, olddeg6 As Single

    'Drawing The Front Right Leg (Picture 1).-----
    If (deg4(HScroll11.Value) <> olddeg4) Then
        Label13.Caption = deg4(HScroll11.Value) - 270
        degtemp = deg4(HScroll11.Value) ' Shift to it's real position
        degtemp = (((22 / 7) / 180) * degtemp) 'convert back to radius
        'draw upperleg
        Line4.X2 = Line4.X1 + 64 * Cos(degtemp)
        Line4.Y2 = Line4.Y1 + 64 * Sin(degtemp)
        Line7.X2 = Line4.X2 + 8
        Line7.Y2 = Line4.Y2
        Shape1.Top = Line4.Y2 + 8
        Shape1.Left = Line4.X2 - 8
        olddeg4 = deg4(HScroll11.Value)
    End If

    If (deg5(HScroll11.Value) <> olddeg5) Then
        Label14.Caption = deg5(HScroll11.Value) - deg4(HScroll11.Value)
        'draw lowerleg with same angle deg5
        degtemp = deg5(HScroll11.Value)
        degtemp = (((22 / 7) / 180) * degtemp) 'convert back to radius
        Line5.X1 = Line4.X2
        Line5.Y1 = Line4.Y2
        Line8.X1 = Line7.X2
        Line8.Y1 = Line7.Y2
        Line5.X2 = Line5.X1 + 64 * Cos(degtemp)
        Line5.Y2 = Line5.Y1 + 64 * Sin(degtemp)
        Line8.X2 = Line5.X2 + 8
        Line8.Y2 = Line5.Y2
        Shape2.Top = Line5.Y2 + 8
        Shape2.Left = Line5.X2 - 8
        olddeg5 = deg5(HScroll11.Value)
    End If

    If (olddeg6 <> deg6(HScroll11.Value)) Then
        Label15.Caption = deg6(HScroll11.Value) - deg5(HScroll11.Value) + 90
        'draw foot with same angle deg6
        degtemp = deg6(HScroll11.Value)
        degtemp = (((22 / 7) / 180) * degtemp) 'convert back to radius
        Line6.X1 = Line5.X2
        Line6.Y1 = Line5.Y2
        Line9.X1 = Line8.X2
        Line9.Y1 = Line8.Y2 - 8
    End If

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งยังมีให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

Line6.X2 = Line6.X1 + 24 * Cos(degtemp)
Line6.Y2 = Line6.Y1 + 24 * Sin(degtemp)
Line9.X2 = Line6.X2
Line9.Y2 = Line6.Y2 - 8
Line10.X1 = Line6.X2
Line10.Y1 = Line6.Y2
Line10.X2 = Line9.X2
Line10.Y2 = Line9.Y2
olddeg6 = deg6(HScroll11.Value)
End If
End Sub
Private Sub LegUpdate3()
Static olddeg23 As Single, olddeg22 As Single, olddeg21 As Single
' Drawing the Front Left Leg (Picture 3).-----
If (olddeg23 <> deg23(HScroll11.Value)) Then
Label37.Caption = deg23(HScroll11.Value) - 270
degtemp = deg23(HScroll11.Value) ' Shift to it's real position
degtemp = (((22 / 7) / 180) * degtemp) 'convert back to radius
'draw upperleg
Line23.X2 = Line23.X1 + 64 * Cos(degtemp)
Line23.Y2 = Line23.Y1 + 64 * Sin(degtemp)
Line20.X2 = Line23.X2 + 8
Line20.Y2 = Line23.Y2
Shape8.Top = Line23.Y2 + 8
Shape8.Left = Line23.X2 - 8
olddeg23 = deg23(HScroll11.Value)
End If
If (olddeg22 <> deg22(HScroll11.Value)) Then
Label38.Caption = deg22(HScroll11.Value) - deg23(HScroll11.Value)
'draw lowerleg with same angle deg22
degtemp = deg22(HScroll11.Value)
degtemp = (((22 / 7) / 180) * degtemp) 'convert back to radius
Line22.X1 = Line23.X2
Line22.Y1 = Line23.Y2
Line19.X1 = Line20.X2
Line19.Y1 = Line20.Y2
Line22.X2 = Line22.X1 + 64 * Cos(degtemp)
Line22.Y2 = Line22.Y1 + 64 * Sin(degtemp)
Line19.X2 = Line22.X2 + 8
Line19.Y2 = Line22.Y2
Shape7.Top = Line22.Y2 + 8
Shape7.Left = Line22.X2 - 8
olddeg22 = deg22(HScroll11.Value)
End If
If (olddeg21 <> deg21(HScroll11.Value)) Then
Label39.Caption = deg21(HScroll11.Value) - deg22(HScroll11.Value) + 90
'draw foot with same angle deg21
degtemp = deg21(HScroll11.Value)
degtemp = (((22 / 7) / 180) * degtemp) 'convert back to radius
Line21.X1 = Line22.X2
Line21.Y1 = Line22.Y2
Line18.X1 = Line19.X2
Line18.Y1 = Line19.Y2 - 8
Line21.X2 = Line21.X1 + 24 * Cos(degtemp)
Line21.Y2 = Line21.Y1 + 24 * Sin(degtemp)
Line18.X2 = Line21.X2
Line18.Y2 = Line21.Y2 - 8
Line15.X1 = Line21.X2
Line15.Y1 = Line21.Y2
Line15.X2 = Line18.X2
Line15.Y2 = Line18.Y2
olddeg21 = deg21(HScroll11.Value)
End If
End Sub
Private Sub LegUpdate2()
Static olddeg16 As Single
'Drawing Side Right (Picture 2) -----

```

```

If (olddeg16 <> deg16(HScroll11.Value)) Then
    Label16.Caption = deg16(HScroll11.Value) - 270
    Label17.Caption = -(deg16(HScroll11.Value) - 270)
    degtemp = deg16(HScroll11.Value) ' Shift to it's real position
    degtemp = (((22 / 7) / 180) * degtemp) 'convert back to radius
'draw upperleg
    Line16.X2 = Line16.X1 + 128 * Cos(degtemp)
    Line16.Y2 = Line16.Y1 + 128 * Sin(degtemp)
    Line13.X2 = Line16.X2 + 8
    Line13.Y2 = Line16.Y2
    Shape5.Top = Line16.Y2 + 16
    Shape5.Left = Line16.X2 - 8
    Line2.X1 = Line16.X2 - 16
    Line2.Y1 = Line16.Y2
    Line2.X2 = Line2.X1 + 40
    Line2.Y2 = Line2.Y1
    Line12.X1 = Line2.X1
    Line12.Y1 = Line2.Y1
    Line12.X2 = Line12.X1
    Line12.Y2 = Line12.Y1 - 8
    Line11.X1 = Line12.X2
    Line11.Y1 = Line12.Y2
    Line11.X2 = Line11.X1 + 40
    Line11.Y2 = Line11.Y1
    Line14.X2 = Line11.X2
    Line14.Y2 = Line11.Y2
    Line14.X1 = Line14.X2
    Line14.Y1 = Line14.Y2 + 8
    olddeg16 = deg16(HScroll11.Value)
End If
End Sub
Private Sub LegUpdate4()
Static olddeg30 As Single
'Drawing Side Left (Picture 4) -----
If (olddeg30 <> deg30(HScroll11.Value)) Then
    Label40.Caption = deg30(HScroll11.Value) - 270
    Label41.Caption = -(deg30(HScroll11.Value) - 270)
    degtemp = deg30(HScroll11.Value) ' Shift to it's real position
    degtemp = (((22 / 7) / 180) * degtemp) 'convert back to radius
'draw upperleg
    Line30.X2 = Line30.X1 + 128 * Cos(degtemp)
    Line30.Y2 = Line30.Y1 + 128 * Sin(degtemp)
    Line31.X2 = Line30.X2 + 8
    Line31.Y2 = Line30.Y2
    Shape9.Top = Line30.Y2 + 16
    Shape9.Left = Line30.X2 - 8
    Line28.X1 = Line30.X2 - 16
    Line28.Y1 = Line30.Y2
    Line28.X2 = Line28.X1 + 40
    Line28.Y2 = Line28.Y1
    Line26.X1 = Line28.X1
    Line26.Y1 = Line28.Y1
    Line26.X2 = Line26.X1
    Line26.Y2 = Line26.Y1 - 8
    Line27.X1 = Line26.X2
    Line27.Y1 = Line26.Y2
    Line27.X2 = Line27.X1 + 40
    Line27.Y2 = Line27.Y1
    Line25.X2 = Line27.X2
    Line25.Y2 = Line27.Y2
    Line25.X1 = Line25.X2
    Line25.Y1 = Line25.Y2 + 8
    olddeg30 = deg30(HScroll11.Value)
End If
' Finished -----
End Sub

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

Private Sub Text1_Change()
temp = Val(Text1.Text)
If temp < 0 Then Text1.Text = "0": temp = 0
If temp > maxframe Then Text1.Text = maxframe: temp = maxframe
VScroll1.Value = temp
End Sub

```

```

Private Sub Text3_Change()
Comon1.FileName = Text3.Text
End Sub

```

```

Private Sub Timer1_Timer()
delayfinished = True
End Sub

```

```

Private Sub VScroll1_Change()
Text1.Text = VScroll1.Value
End Sub

```

```

Sub SendFrame(frame As Integer)
Dim databyte As Byte
'MSComm1.PortOpen = True
Open "com2" For Output As 1
'send motor1
datasing = deg4(frame) - 270
databyte = Int((datasing + 90) * (255 / 180))
SendServ 0, databyte
'send motor2
datasing = deg5(frame) - deg4(frame)
databyte = Int((datasing + 90) * (255 / 180))
SendServ 1, databyte
'send motor3
datasing = deg6(frame) - deg5(frame) + 90
databyte = Int((datasing + 90) * (255 / 180))
SendServ 2, databyte
'send motor4
datasing = deg16(frame) - 270
databyte = Int((datasing + 90) * (255 / 180))
SendServ 3, databyte
'send motor5
datasing = -(deg16(frame) - 270)
databyte = Int((datasing + 90) * (255 / 180))
SendServ 4, databyte
'send motor6
datasing = deg23(frame) - 270
databyte = Int((datasing + 90) * (255 / 180))
SendServ 5, databyte
'send motor7
datasing = deg22(frame) - deg23(frame)
databyte = Int((datasing + 90) * (255 / 180))
SendServ 6, databyte
'send motor8
datasing = deg21(frame) - deg22(frame) + 90
databyte = Int((datasing + 90) * (255 / 180))
SendServ 7, databyte
'send motor9
datasing = deg30(frame) - 270
databyte = Int((datasing + 90) * (255 / 180))
SendServ 8, databyte
'send motor10
datasing = -(deg30(frame) - 270)
databyte = Int((datasing + 90) * (255 / 180))
SendServ 9, databyte
'finished fetch frame to Z8
'MSComm1.PortOpen = False

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Close #1
End Sub
Sub SendServ(id As Byte, dat As Byte)
    Dim temp1 As Byte
    Dim temp2 As Byte
    temp1 = (id * 2) Or ((dat And 128) / 128)
    temp2 = (dat Or 128)
    Print #1, Chr$(temp1) + Chr$(temp2)
    'MSComm1.Output = Chr$(temp1) + Chr$(temp2)
End Sub
```



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

VERSION 4.00
Begin VB.Form Form2
    BackColor      = &H00C0C000&
    BorderStyle    = 1 'Fixed Single
    Caption        = "Communication Setting"
    ClientHeight   = 4425
    ClientLeft     = 5610
    ClientTop      = 1920
    ClientWidth    = 4725
    Height         = 4830
    Left          = 5550
    LinkTopic      = "Form2"
    ScaleHeight    = 4425
    ScaleWidth     = 4725
    Top           = 1575
    Width         = 4845
Begin VB.Frame Frame4
    BackColor      = &H00C0C000&
    Caption        = "Parity"
    Height         = 1695
    Left          = 1560
    TabIndex      = 27
    Top           = 120
    Width         = 1095
Begin VB.OptionButton Option3
    BackColor      = &H00C0C000&
    Caption        = "Space"
    Height         = 255
    Index         = 4
    Left          = 120
    TabIndex      = 36
    Top           = 1320
    Width         = 855
End
Begin VB.OptionButton Option3
    BackColor      = &H00C0C000&
    Caption        = "Mark"
    Height         = 255
    Index         = 3
    Left          = 120
    TabIndex      = 31
    Top           = 1080
    Width         = 855
End
Begin VB.OptionButton Option3
    BackColor      = &H00C0C000&
    Caption        = "Even"
    Height         = 255
    Index         = 2
    Left          = 120
    TabIndex      = 30
    Top           = 840
    Width         = 855
End
Begin VB.OptionButton Option3
    BackColor      = &H00C0C000&
    Caption        = "Odd"
    Height         = 255
    Index         = 1
    Left          = 120
    TabIndex      = 29
    Top           = 600
    Width         = 855
End
Begin VB.OptionButton Option3
    BackColor      = &H00C0C000&
    Caption        = "None"

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

        Height      = 255
        Index       = 0
        Left        = 120
        TabIndex    = 28
        Top         = 360
        Value       = -1 'True
        Width       = 855
    End
End
Begin VB.CommandButton Command2
    Caption       = "Cancel"
    Height        = 495
    Left         = 2880
    TabIndex     = 26
    Top          = 3840
    Width        = 1215
End
Begin VB.CommandButton Command1
    Caption       = "Ok"
    Height        = 495
    Left         = 720
    TabIndex     = 25
    Top          = 3840
    Width        = 1215
End
Begin VB.CheckBox Check1
    BackColor     = &H00C0C000&
    Caption       = "DTR Enable"
    Height        = 255
    Left         = 2760
    TabIndex     = 24
    Top          = 3480
    Value        = 1 'Checked
    Width        = 1695
End
Begin VB.TextBox Text4
    Height        = 315
    Left         = 1440
    TabIndex     = 22
    Text         = "0"
    Top          = 3480
    Width        = 1215
End
Begin VB.TextBox Text3
    Height        = 315
    Left         = 1440
    TabIndex     = 20
    Text         = "0"
    Top          = 3120
    Width        = 1215
End
Begin VB.TextBox Text2
    Height        = 315
    Left         = 1440
    TabIndex     = 18
    Text         = "1"
    Top          = 2760
    Width        = 1215
End
Begin VB.TextBox Text1
    Height        = 315
    Left         = 1440
    TabIndex     = 16
    Text         = "512"
    Top          = 2400
    Width        = 1215
End
End

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

Begin VB.Frame Frame5
    BackColor      = &H00C0C000&
    Caption        = "Handshaking"
    Height         = 1455
    Left           = 2880
    TabIndex       = 13
    Top            = 1920
    Width          = 1695
    Begin VB.OptionButton Option5
        BackColor    = &H00C0C000&
        Caption       = "Both"
        Height        = 255
        Index         = 3
        Left          = 120
        TabIndex      = 38
        Top           = 1080
        Width         = 1455
    End
    Begin VB.OptionButton Option5
        BackColor    = &H00C0C000&
        Caption       = "Hardware"
        Height        = 255
        Index         = 2
        Left          = 120
        TabIndex      = 37
        Top           = 840
        Width         = 1455
    End
    Begin VB.OptionButton Option5
        BackColor    = &H00C0C000&
        Caption       = "No Handshake"
        Height        = 255
        Index         = 0
        Left          = 120
        TabIndex      = 15
        Top           = 360
        Value         = -1 'True
        Width         = 1455
    End
    Begin VB.OptionButton Option5
        BackColor    = &H00C0C000&
        Caption       = "Software"
        Height        = 255
        Index         = 1
        Left          = 120
        TabIndex      = 14
        Top           = 600
        Width         = 1455
    End
End
Begin VB.Frame Frame3
    BackColor      = &H00C0C000&
    Caption        = "Stop Bit"
    Height         = 1695
    Left           = 3720
    TabIndex       = 9
    Top            = 120
    Width          = 855
    Begin VB.OptionButton Option4
        BackColor    = &H00C0C000&
        Caption       = "2"
        Height        = 255
        Index         = 2
        Left          = 120
        TabIndex      = 35
        Top           = 840
        Width         = 495
    End
End

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

End
Begin VB.OptionButton Option4
    BackColor      = &H00C0C000&
    Caption        = "1.5"
    Height         = 255
    Index         = 1
    Left           = 120
    TabIndex       = 11
    Top            = 600
    Width          = 495
End
Begin VB.OptionButton Option4
    BackColor      = &H00C0C000&
    Caption        = "1"
    Height         = 255
    Index         = 0
    Left           = 120
    TabIndex       = 10
    Top            = 360
    Value          = -1 'True
    Width          = 495
End
End
Begin VB.Frame Frame2
    BackColor      = &H00C0C000&
    Caption        = "Data Bit"
    Height         = 1695
    Left           = 2760
    TabIndex       = 6
    Top            = 120
    Width          = 855
Begin VB.OptionButton Option2
    BackColor      = &H00C0C000&
    Caption        = "4"
    Height         = 255
    Index         = 4
    Left           = 120
    TabIndex       = 34
    Top            = 1320
    Width          = 495
End
Begin VB.OptionButton Option2
    BackColor      = &H00C0C000&
    Caption        = "5"
    Height         = 255
    Index         = 3
    Left           = 120
    TabIndex       = 33
    Top            = 1080
    Width          = 495
End
Begin VB.OptionButton Option2
    BackColor      = &H00C0C000&
    Caption        = "6"
    Height         = 255
    Index         = 2
    Left           = 120
    TabIndex       = 32
    Top            = 840
    Width          = 495
End
Begin VB.OptionButton Option2
    BackColor      = &H00C0C000&
    Caption        = "8"
    Height         = 255
    Index         = 0
    Left           = 120

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

        TabIndex      =      8
        Top           =      360
        Value         =      -1 'True
        Width         =      495
    End
    Begin VB.OptionButton Option2
        BackColor      =      &H00C0C000&
        Caption        =      "7"
        Height         =      255
        Index          =      1
        Left           =      120
        TabIndex       =      7
        Top            =      600
        Width          =      495
    End
End
Begin VB.Frame Frame1
    BackColor      =      &H00C0C000&
    Caption        =      "Port"
    Height         =      1695
    Left           =      240
    TabIndex       =      1
    Top            =      120
    Width          =      1095
    Begin VB.OptionButton Option1
        BackColor      =      &H00C0C000&
        Caption        =      "COM1"
        Height         =      255
        Index          =      0
        Left           =      120
        TabIndex       =      5
        Top            =      360
        Value          =      -1 'True
        Width          =      855
    End
    Begin VB.OptionButton Option1
        BackColor      =      &H00C0C000&
        Caption        =      "COM2"
        Height         =      255
        Index          =      1
        Left           =      120
        TabIndex       =      4
        Top            =      600
        Width          =      855
    End
    Begin VB.OptionButton Option1
        BackColor      =      &H00C0C000&
        Caption        =      "COM3"
        Height         =      255
        Index          =      2
        Left           =      120
        TabIndex       =      3
        Top            =      840
        Width          =      855
    End
    Begin VB.OptionButton Option1
        BackColor      =      &H00C0C000&
        Caption        =      "COM4"
        Height         =      255
        Index          =      3
        Left           =      120
        TabIndex       =      2
        Top            =      1080
        Width          =      855
    End
End
Begin VB.ComboBox Combo1

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

        BackColor      =    &H00C0C000&
        Height         =    300
        Left           =    1440
        TabIndex       =    0
        Text           =    "9600"
        Top            =    2040
        Width          =    1215
    End
    Begin VB.Label Label5
        Alignment       =    1 'Right Justify
        BackStyle       =    0 'Transparent
        Caption         =    "CTS Timeout"
        Height          =    255
        Left            =    240
        TabIndex       =    23
        Top             =    3480
        Width           =    1095
    End
    Begin VB.Label Label4
        Alignment       =    1 'Right Justify
        BackStyle       =    0 'Transparent
        Caption         =    "CD Timeout"
        Height          =    255
        Left            =    240
        TabIndex       =    21
        Top             =    3120
        Width           =    1095
    End
    Begin VB.Label Label3
        Alignment       =    1 'Right Justify
        BackStyle       =    0 'Transparent
        Caption         =    "Send Theshold"
        Height          =    255
        Left            =    240
        TabIndex       =    19
        Top             =    2760
        Width           =    1095
    End
    Begin VB.Label Label2
        Alignment       =    1 'Right Justify
        BackStyle       =    0 'Transparent
        Caption         =    "O/P Buffer"
        Height          =    255
        Left            =    480
        TabIndex       =    17
        Top             =    2400
        Width           =    855
    End
    Begin VB.Label Label1
        Alignment       =    1 'Right Justify
        BackStyle       =    0 'Transparent
        Caption         =    "Baud Rate"
        Height          =    255
        Left            =    480
        TabIndex       =    12
        Top             =    2040
        Width           =    855
    End
End
Attribute VB_Name = "Form2"
Attribute VB_Creatable = False
Attribute VB_Exposed = False
Public par As String, dbit As String, stpbit As String
Public brate As String, comindx As Integer, hndshk As Integer

Private Sub Check1_Click()
    Select Case (Check1.Value)

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

Case 0
    DTR1 = False
Case 1
    DTR1 = True
End Select
End Sub

Private Sub Comb1_Change()
brat = Comb1.Text
'Form1.MSComm1.Settings = brate + "," + par + "," + dbit + "," + stpbit
End Sub

Private Sub Command1_Click()
Form1.MSComm1.Settings = brate + "," + par + "," + dbit + "," + stpbit
Form1.MSComm1.CommPort = comindx + 1
Form1.MSComm1.Handshaking = hndshk
Form1.MSComm1.OutBufferSize = Val(Text1.Text)
Form1.MSComm1.SThreshold = Val(Text2.Text)
Form1.MSComm1.CDTimeout = Val(Text3.Text)
Form1.MSComm1.CTSTimeout = Val(Text4.Text)
Form1.MSComm1.RTSEnable = DTR1
Form2.Visible = False
Form1.Enabled = True
Form1.SetFocus
End Sub

Private Sub Command2_Click()
Form2.Visible = False
Form1.Enabled = True
Form1.SetFocus
End Sub

Private Sub Form_Load()
Comb1.AddItem "1200"
Comb1.AddItem "2400"
Comb1.AddItem "4800"
Comb1.AddItem "9600"
Comb1.AddItem "14400"
Comb1.AddItem "19200"
a$ = Form1.MSComm1.Settings
i = InStr(a$, ",")
brate = Left$(a$, i - 1)
par = Mid$(a$, i + 1, 1)
dbit = Mid$(a$, i + 3, 1)
stpbit = Right$(a$, Len(a$) - (i + 4))
Comb1.Text = brate
Option1(Form1.MSComm1.CommPort - 1) = True
comindx = Form1.MSComm1.CommPort - 1
Option2(8 - Val(dbit)) = True
Select Case par
Case "N"
    i = 0
Case "O"
    i = 1
Case "E"
    i = 2
Case "M"
    i = 3
Case "S"
    i = 4
End Select
Option3(i) = True
Select Case stpbit
Case "1"
    i = 0
Case "1.5"
    i = 1

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

Case "2"
    i = 2
End Select
Option4(i) = True
Option5(Form1.MSComm1.Handshaking) = True
hndshk = Form1.MSComm1.Handshaking
Text1.Text = Form1.MSComm1.OutBufferSize
Text2.Text = Form1.MSComm1.SThreshold
Text3.Text = Form1.MSComm1.CDTimeout
Text4.Text = Form1.MSComm1.CTSTimeout
If (Form1.MSComm1.DTREnable) Then
    Check1.Value = 1: DTR1 = True
Else
    Check1.Value = 0: DTR1 = False
End If

End Sub

Private Sub Option1_Click(Index As Integer)
    'Form1.MSComm1.CommPort = Index + 1
    comindx = Index
End Sub

Private Sub Option3_Click(Index As Integer)
    Select Case Index
    Case 0
        par = "N"
    Case 1
        par = "O"
    Case 2
        par = "E"
    Case 3
        par = "M"
    Case 4
        par = "S"
    End Select
    'Form1.MSComm1.Settings = brate + "," + par + "," + dbit + "," + stpbit

End Sub

Private Sub Option2_Click(Index As Integer)
    Select Case Index
    Case 0
        dbit = "8"
    Case 1
        dbit = "7"
    Case 2
        dbit = "6"
    Case 3
        dbit = "5"
    Case 4
        dbit = "4"
    End Select
    'Form1.MSComm1.Settings = brate + "," + par + "," + dbit + "," + stpbit

End Sub

Private Sub Option4_Click(Index As Integer)
    Select Case Index
    Case 0
        stpbit = "1"
    Case 1
        stpbit = "1.5"
    Case 2
        stpbit = "2"
    End Select
    'Form1.MSComm1.Settings = brate + "," + par + "," + dbit + "," + stpbit
End Sub

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

Private Sub Option5_Click(Index As Integer)
'Form1.MSComm1.HandShaking = Index
hndshk = Index
End Sub

Private Sub Text1_Change()
'Form1.MSComm1.OutBufferSize = Val(Text1.Text)
End Sub

Private Sub Text2_Change()
'Form1.MSComm1.STThreshold = Val(Text2.Text)
End Sub

Private Sub Text3_Change()
'Form1.MSComm1.CDTimeout = Val(Text3.Text)
End Sub

Private Sub Text4_Change()
'Form1.MSComm1.CTSTimeout = Val(Text4.Text)
End Sub

```



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้