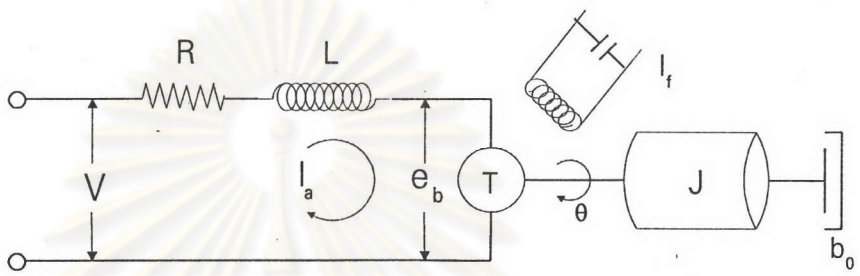
ภาคผนวก

ภาคผนวก ก

แบบจำลองทางคณิตศาสตร์ของมอเตอร์

มอเตอร์เป็นอุปกรณ์ทำหน้าที่เปลี่ยนพลังงานไฟฟ้าเป็นพลังงานกล ซึ่งนิยมใช้ในการควบคุมทั่ว ๆ ไป สำหรับโครงการวิทยานิพนธ์นี้ใช้มอเตอร์กระแสสลับแบบไม่ใช้แปรงถ่าน



รูปที่ 1 วงจรของมอเตอร์กระแสตรงแบบฟิลด์คงที่

กำหนดให้

R = ความต้านทานของอาร์มาเจอร์

L = อินดัคแต็นซ์ของอาร์มาเจอร์

I_a = กระแสอาร์มาเจอร์

V = โวลต์เตจที่ป้อนให้กับมอเตอร์

e_b = โวลต์เตจย้อนกลับ (BACK emf.)

θ = มุมที่มอเตอร์หมุน

T = แรงบิดที่ได้จากมอเตอร์

J = โมเมนต์ของแรงเฉื่อยของมอเตอร์และโหลด

b_O = สัมประสิทธ์ของวิสคอสแด้มป์ปิ้ง ของมอเตอร์

เนื่องจากมอเตอร์ที่ใช้เป็นแบบแม่เหล็กถาวร เส้นแรงของฟิลด์มีค่าคงที่ ดังนั้นอัตราส่วนระหว่างกระแสอาร์มาเจอร์และแรงบิดจะมีค่าคงที่ ตามสมการที่ 1

$$T = K_m I_a \qquad (1)$$

เมื่อ K_m = ค่าคงที่แรงบิดมอเตอร์ (MOTOR TORQUE CONSTANT) และในขณะที่มอเตอร์หมุนจะเกิดโวลท์เตจย้อนกลับ ซึ่งมีค่าเท่ากับ

$$e_b = K_b \frac{d\theta}{dt} \qquad (2)$$

เมื่อ K_b = ค่าคงที่ของโวท์เตจย้อนกลับของมอเตอร์ จากรูปที่ 1 เราจะได้ว่า

$$L\frac{dI_a}{dt} + RI_a + e_b = V \qquad (3)$$

จากสมการสมดุลย์ของแรงบิด

$$J\ddot{\theta} + b_0\dot{\theta} = T = K_m I_a \qquad (4)$$

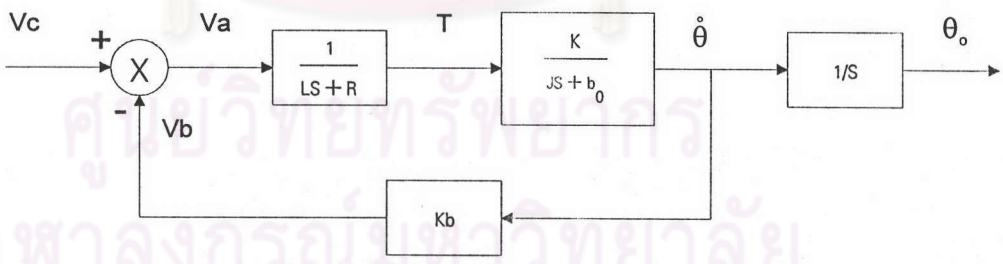
ถ้ากำหนดให้เงื่อนไขเริ่มต้นเป็นศูนย์ จากสมการที่ (2), (3) และ (4) ลาปลาสทรานฟอร์มของสมการข้างต้น เขียนได้เป็น

$$K_b S\Theta(S) = E_b(S) \qquad (5)$$

$$(LS + R)\, I(S) + E_b(S) = V(S) \qquad (6)$$

$$(JS^2 + BS)\Theta(S) = T(S) = K_m I_a(S) \qquad (7)$$

จากสมการที่ (5), (6) และ (7) เราสามารถเขียนบล็อกไดอะแกรม ดังแสดงในรูปที่ 2 และทำการยุบบล็อก จะได้ฟังก์ชั่นถ่ายโอน(TRANSFER FUNCTION) ตามสมการที่ (8)



รูปที่ 2 บล็อคไดอะแกรมของวงจร

$$\frac{\Theta(S)}{V(S)} = \frac{K_m}{S[LJS^2 + (Lb_0 + RJ)S + Rb_0 + K_m K_b]} \qquad (8)$$

ถ้าเราถือว่าค่า L มีค่าน้อยกว่าค่า R มากๆ และตัดทิ้งได้ จะได้ว่า

$$\frac{\Theta(S)}{V(S)} = \frac{K}{S(JS+B)} \qquad (9)$$

โดยกำหนดให้

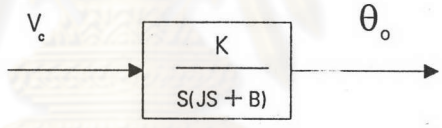
$$K = \frac{K_m}{R} \qquad (10)$$

= ค่าคงที่สมมูลมอเตอร์ หรือ ค่าคงที่มอเตอร์(MOTOR CONSTANT)

$$B = b_0 + \frac{K_m K_b}{R} \qquad (11)$$

= สัมประสิทธิ์วิสคอสแดมปิ้งสมมูล ของมอเตอร์(EQUIVALENT VISCOUS FRICTION COEFFICIENT)

ดังนั้นบล็อกไดอะแกรมของมอเตอร์ จึงเขียนอยู่ในรูป ค่าคงที่มอเตอร์ และสัมประสิทธิ์วิสคอสแดมปิ้งสมมูล ได้ดังรูปที่ 3



รูปที่ 3 บล็อกไดอะแกรมของมอเตอร์

ภาคผนวก ข

รายละเอียดอุปกรณ์ที่ใช้ในงานวิจัย

ตารางแสดงรายละเอียดอุปกรณ์ที่ใช้ในงานวิจัย

แกนเคลื่อนที่	ชื่อรุ่นอุปกรณ์	รายละเอียด	ผู้ผลิต	หมายเหตุ
X Y	ZXF-605 BL-2340	Brushless Servo Motor and Drive	PARKER	with positioner
X Y	∅25x10 ∅16x16	Ball Screw and Ball nut	STAR	accuracy grade 0.025 mm l300 mm
X Y	SIZE #45 SIZE#25	Ball Rail and Ball Nut	STAR	Preload 0.2 Co accuracy class H

30 BL SERVO DRIVES USER GUIDE

Brushless Motor/Drive Packages

The BL Series drives may be matched with motors in the Digiplan brushless range and supplied as ready-wired motor/drive packages. Details of the range of four motors (Types ML-1620, ML-2340, ML-3450 and ML-3475) are given in Table 4-2.

Type	Weights (including cable)	Rotor Inertia Kg-cm^2	Incremental Encoder Line Count
ML-1620	0.85Kg	0.056	500
ML-2340	2.1Kg	0.28	500
ML-3450	5.1Kg	1.6	1000
ML-3475	6.4Kg	2.4	1000

Table 4-2. Brushless Motor Data

รูปที่ 1 รายละเอียดของมอเตอร์ แกน Y

Chapter Objectives

This chapter is designed to function as a quick reference tool for system specifications.

BL Drive Specification

	BL30	BL75	BL150
Continuous Current	3.75A	7.5A	15A
Peak current	7.5A	15A	30A
DC bus Voltage	85V	150V	150V
AC Input Voltage: Nom.	61V	107V	107V
Max.	67V	118V	118V
Min.	24V	48V	48V
Weights *	0.8Kg	1.7Kg	2.8Kg
Motor Options	ML-1620, ML-2340	ML-2340, ML-3450, ML-1620	ML-3450, ML-3475
Power input	AC direct from mains transformer		
Control input	±10V analogue (torque or velocity)		
Reference outputs	±15V at 10mA		
Velocity feedback	Built-in incremental encoder		
Commutation method	4 bit absolute encoder		
Torque amplifier bandwidth	>2500Hz		
Switching frequency	20Khz		
Velocity amp gain	0.66/6.6/51/500/4800 set by links		
Torque amp gain	0.75/0.075AV	1.5/0.15AV	3/0.3 AV
Gain linearity	±4%		
Typ. input amp drift	10µV/°C		
Power dump current	9A @ 100V	16A @ 100V	16A @ 100V
Max. cont. dump power	40W		
Min recommended load inductance	0.5mH		
Jumper link settings	Input range, current limit, torque/vel. mode		
Potentiometer settings	Time constant, damping, balance, tach gain		
Diagnostic LED's (Front)	Power on, current limit, overtemperature, drive/motor fault		
Diagnostic LED's (Rear)	Power on, composite fault		
Dimensions	See Figure 3-1 (Chapter 3)		

* If the drive is supplied with the positioner option, add 0.5Kg to this weight.

Table 4-1. BL Servo Drives Specification

รูปที่ 1 (ต่อ) รายละเอียดของมอเตอร์ แกน Y

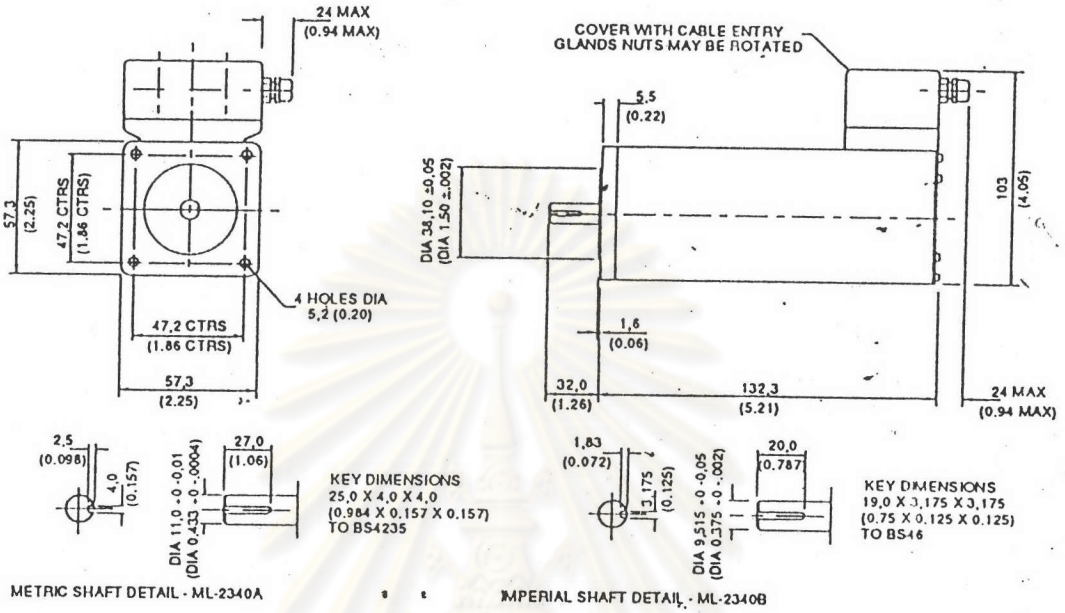
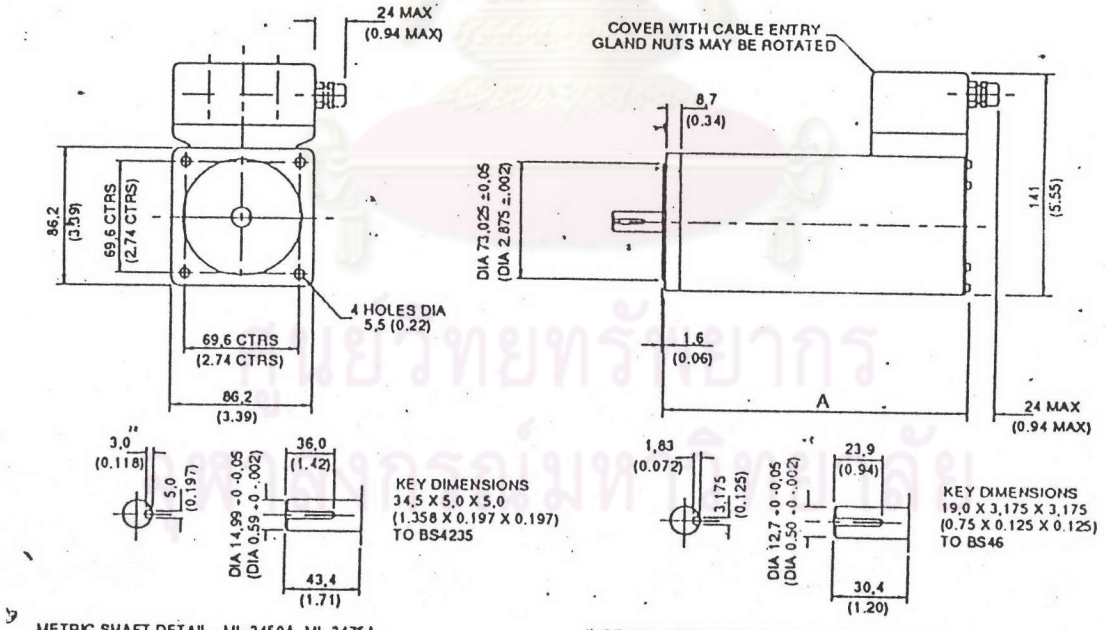


Figure 4-2. Motor Type ML-2340 Dimensions



Motor Type	3450	3475
Dimension A	177 (6.97)	217 (8.54)

Figure 4-3. Motor Type ML-3450 & ML-3475 Dimensions

รูปที่ 2 รายละเอียดของชุดขับมอเตอร์แกน Y

Motor/Drive Package Performance Data

The torque curves for the possible motor/drive combinations are shown in Figure 4-4.

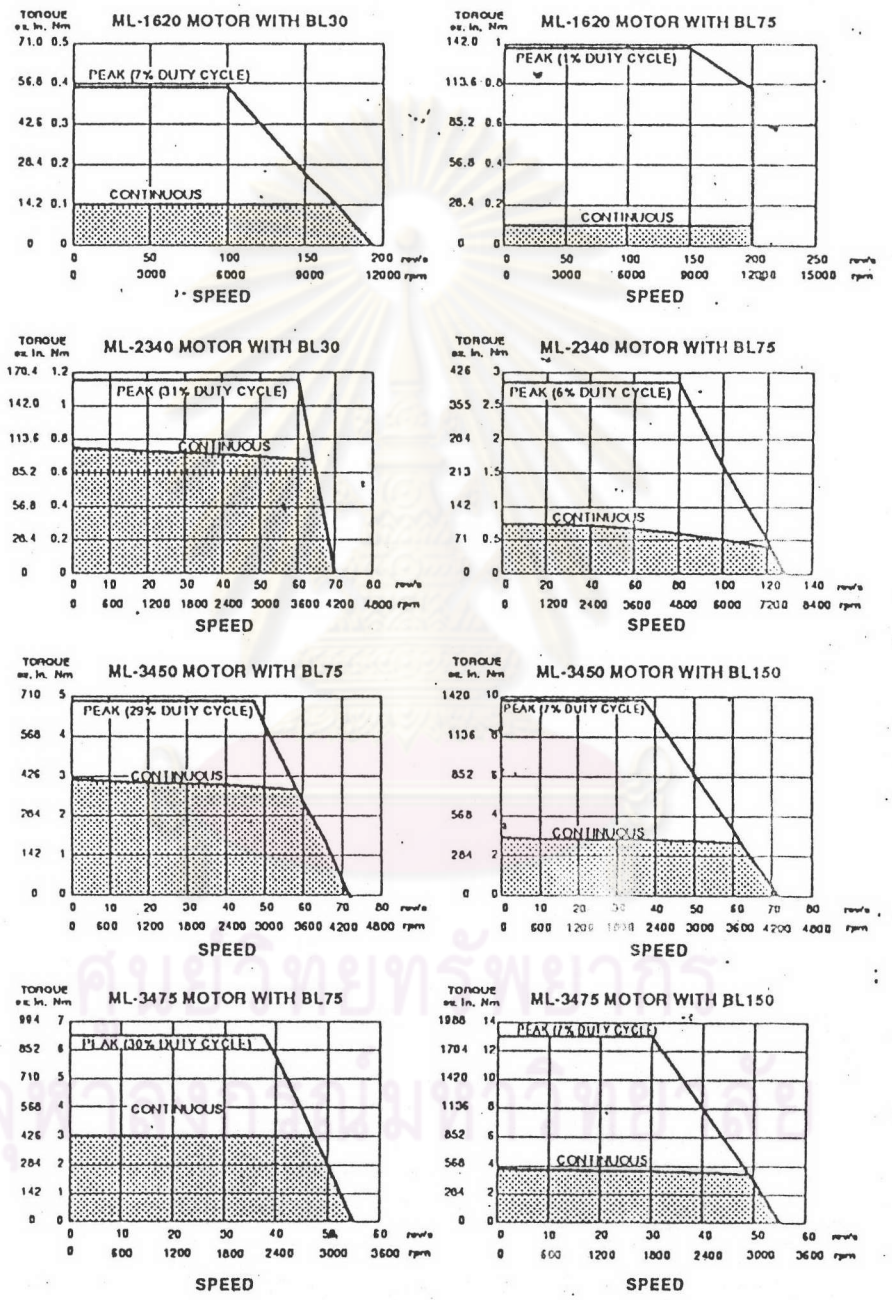


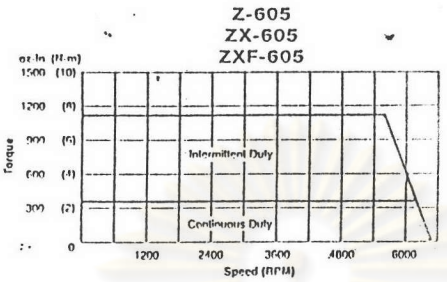
Figure 4-5. Motor/Drive Packages Torque Curves

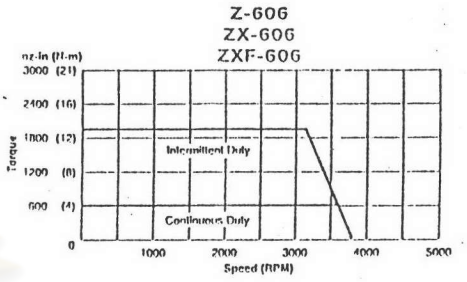
รูปที่ 2(ต่อ) รายละเอียดของมอเตอร์ แกน Y

Z, ZX & ZXF Series

Brushless Servo Systems

Torque/Speed Curves



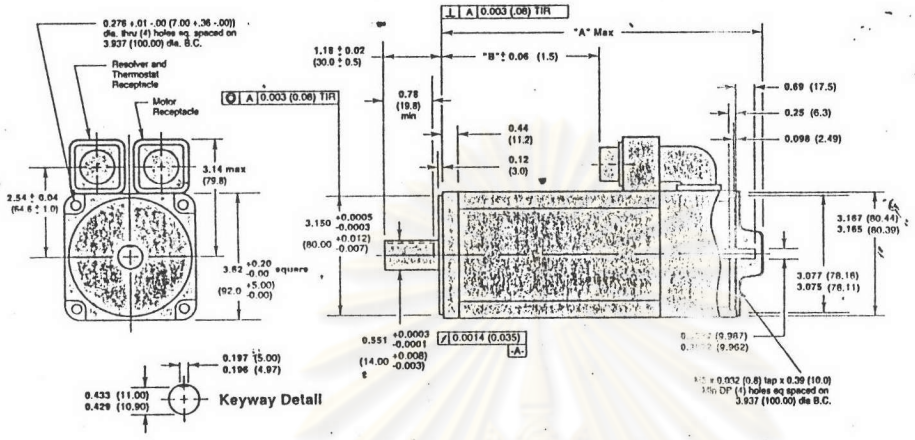


Technical Data		Z-605 ZX-605 ZXF-605	Z-606 ZX-606 ZXF-606	Z-610 ZX-610 ZXF-610	Z-620 ZX-620 ZXF-620	Z-630 ZX-630 ZXF-630	Z-635 ZX-635 ZXF-635	Z-640 ZX-640 ZXF-640	Z-910 ZX-910 ZXF-910	Z-920 ZX-920 ZXF-920	Z-930 ZX-930 ZXF-930	Z-940 ZX-940 ZXF-940
Continuous stall torque												
	oz-in	346	633	867	1,743	2,475	2,319	4,114	2,407	4,263	5,990	9,021
	lb-in	22	40	54	109	155	145	257	150	266	374	564
	Nm	2.44	4.47	6.12	12.31	17.48	16.38	29.05	17.0	30.1	42.3	63.7
Peak torque	oz-in	1,083	1,954	1,733	3,486	4,951	4,638	8,228	5,205	8,525	11,980	18,041
	lb-in	68	122	108	218	309	290	514	325	533	749	1,128
	Nm	7.65	13.80	12.24	24.62	34.96	32.75	58.10	35.4	61.5	84.6	127.5
Rated power	hp	2	2.1	4.2	5.6	5.4	5.4	5.9	9.6	10.4	11.0	11.1
	k Watts	1.49	1.57	3.13	4.18	4.03	4.24	4.40	7.2	7.8	8.2	8.3
Rated speed	rpm	6,200	3,600	7,000	3,700	2,500	3,000	1,600	5,000	3,150	2,300	1,500
	rps	103	60	117	62	42	50	27	83.3	52.5	38.3	25.0
Rated current (line)	A (rms)	5	5.3	14.1	14.1	14.1	14.1	14.1	27.2	27.7	28.3	28.3
Peak current (3.3 sec max)	A (rms)	16.6	17.2	28.2	28.2	28.2	28.2	28.2	56.6	56.6	56.6	56.6
Max cont AC input power (3 phase 240 VAC)	A (rms)	6	6	15	15	15	15	15	30	30	30	30
Rotor inertia	oz-in² (mass)	5.45	9.45	13.73	35.87	50.79	56.21	111.21	50.79	111.21	166.21	459.48
	oz-in-sec²	0.01	0.02	0.04	0.09	0.13	0.15	0.29	0.132	0.288	0.431	1.190
	kg m² x 10⁻⁴	99.6	172.9	251.2	656	929	1,028	2,034	929	2,034	3,040	8,404
Motor weight	lbs	10.0	14.0	17.0	29.0	32.0	37	51.0	32.0	57.0	65.0	112.0
	kg	4.5	6.4	7.7	13.2	14.5	16.8	23.2	15.0	26.0	29.0	51.0
Shipping weight	lbs	52.0	55.0	58.0	71.0	74.0	79.0	93.0	89.0	114.0	122.0	169.0
	kg	23.6	25.0	26.4	32.3	33.6	35.9	42.3	40.0	52.0	55.0	77.0

รูปที่ 3 รายละเอียดของมอเตอร์ แกน X

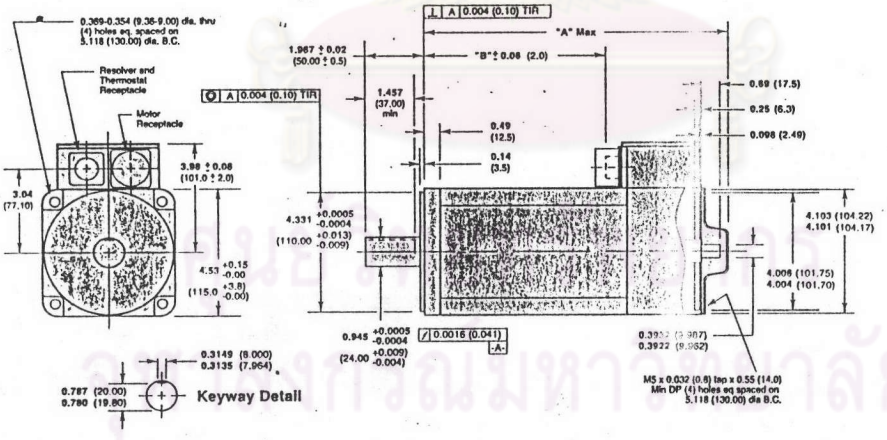
Dimensions (—) denotes millimeters

Z605
Z606
Z610



Model	A	B
Z605	9.30 (236.20)	4.87 (123.7)
Z606	10.86 (275.80)	6.42 (163.1)
Z610	12.42 (315.40)	7.23 (183.6)

Z620
Z630
Z910



Model	A	B
Z620	12.55 (318.8)	8.30 (210.9)
Z630	14.65 (372.1)	10.41 (264.3)
Z910	14.65 (372.1)	10.41 (264.3)

รูปที่ 3(ต่อ) รายละเอียดของมอเตอร์ แกน X

Right-hand thread

Ground ball race in nut

Interchangeable and combinable within modular series range

For housings refer to Table 16/17 on page 3.2.20

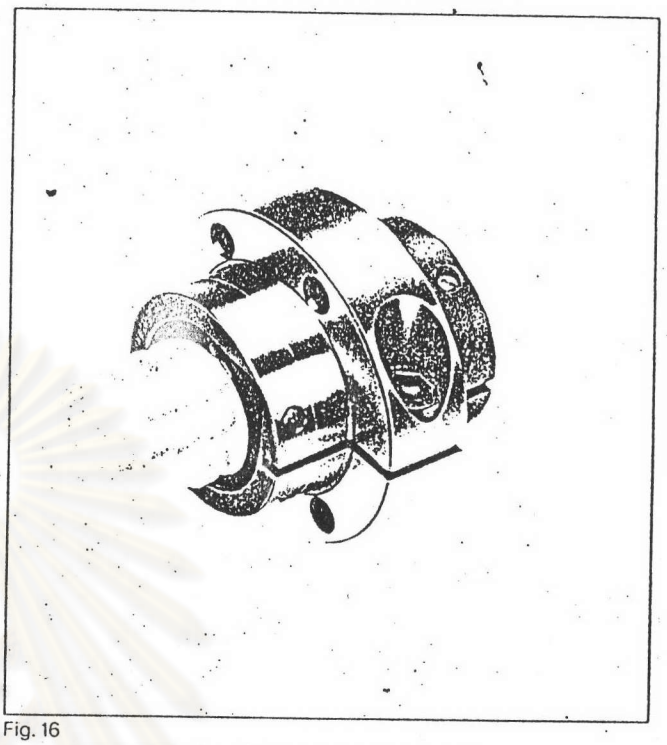
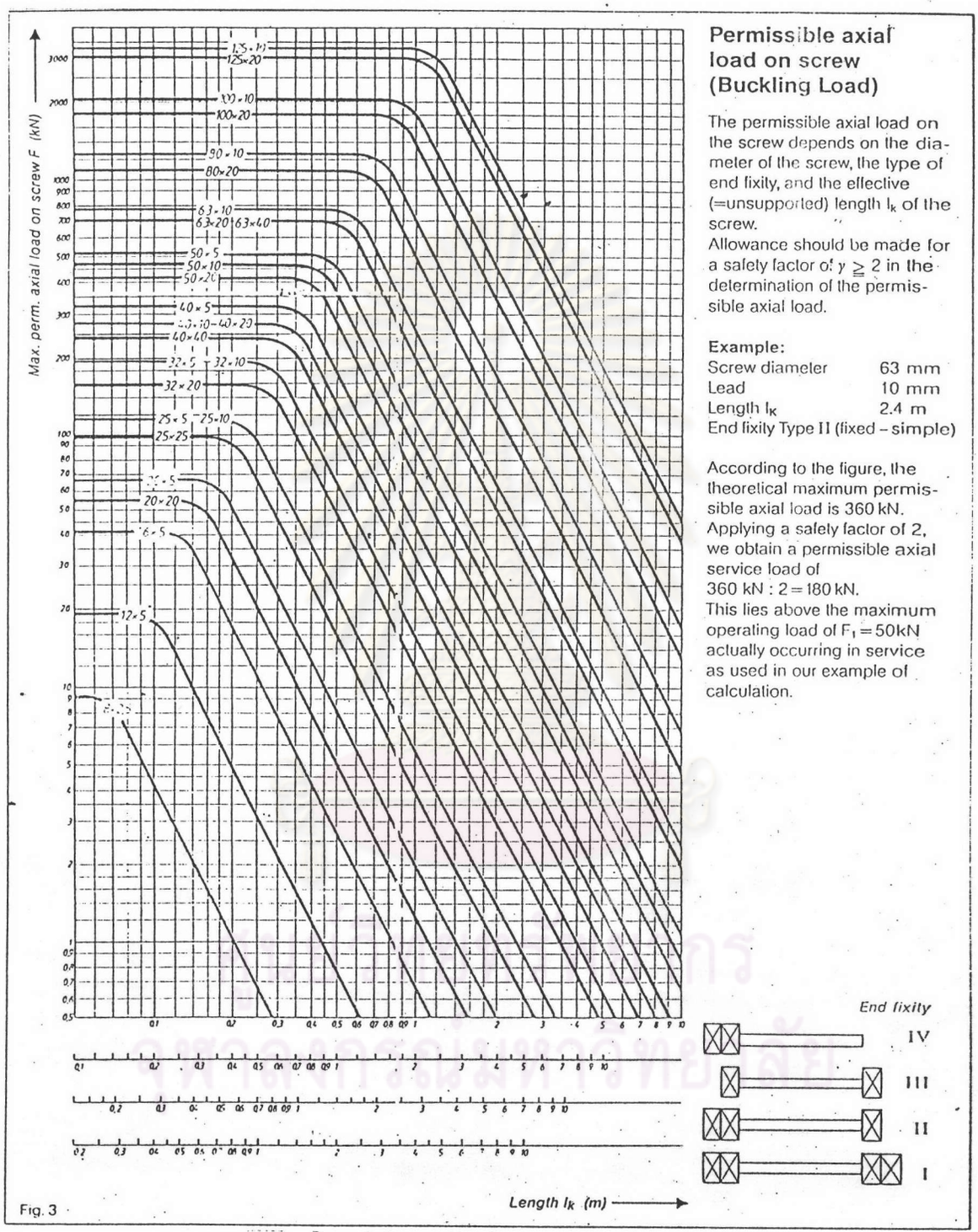


Fig. 16

Standard element, available at short notice

Lube hole		Load capacities (N)		Size	Reference number	Weights	
	Angle (°)	dyn.	stat.		screw	nut	screw
	α	C	C_0	$d_0 \times P$	(kg/m)	(kg/m)	(kg)
$3,9^{H11}$	90	3 000	2 900	8 × 2,5	1532-2-3004	0,32	0,060
M6	35	6 900	7 200	12 × 5	1532-4-6004	0,70	0,13
M6	37	11 900	15 100	16 × 5	1512-0-1004	1,26	0,27
M6	40	9 200	8 400	16 × 16	1512-0-6004	1,27	0,39
M6	34	12 300	19 100	20 × 5	1512-1-1004	2,06	0,33
M6	30	16 000	14 300	20 × 20	1512-1-7004	1,98	0,63
M6	30	13 300	25 200	25 × 5	1512-2-1004	3,33	0,44
M6	30	13 300	25 200	25 × 10	1512-2-4004	3,44	0,53
M6	42	18 700	20 600	25 × 25	1512-2-8004	3,16	1,16
M6	31	17 700	37 000	32 × 5	1512-3-1004	5,50	0,64
M6	282	20 800	46 800	32 × 10	1512-3-4004	5,71	0,86
M8 × 1	30	18 600	25 100	32 × 20	1512-3-7004	5,33	1,09
M8 × 1	30	18 600	25 100	32 × 32	1512-3-9004	5,49	1,39
M8 × 1	25	22 000	59 000	40 × 5	1512-4-1004	8,84	0,87
M8 × 1	33	43 000	76 600	40 × 10	1512-4-4004	8,21	1,53
M8 × 1	270	34 500	56 500	40 × 20	1512-4-7004	8,21	1,77
M8 × 1	41	43 600	52 700	40 × 40	1512-4-9004	8,21	3,77
M8 × 1	23	23 500	74 700	50 × 5	1512-5-1004	14,20	1,23
M8 × 1	29	62 100	153 800	50 × 10	1512-5-4004	13,40	2,44
M8 × 1	270	70 250	106 700	50 × 20	1512-5-7004	12,70	3,78
M8 × 1	30	51 500	68 600	50 × 40	1512-5-9004	12,70	4,36
M8 × 1	25	64 700	191 000	63 × 10	1512-6-4004	21,80	2,94
M8 × 1	260	72 200	133 700	63 × 20	1512-6-7004	21,00	4,16
M8 × 1	20	53 000	86 000	63 × 40	1512-6-9004	21,00	4,86
M8 × 1	23	78 500	268 000	80 × 10	1512-7-4004	35,70	4,20
M8 × 1	30	220 000	496 100	80 × 20	1512-7-7004	32,40	13,30

รูปที่ 4 รายละเอียดของบอลสกรู และ บอลนัต



Permissible axial load on screw (Buckling Load)

The permissible axial load on the screw depends on the diameter of the screw, the type of end fixity, and the effective (=unsupported) length l_k of the screw.
Allowance should be made for a safety factor of $y \geq 2$ in the determination of the permissible axial load.

Example:

Screw diameter	63 mm
Lead	10 mm
Length l_K	2.4 m

End fixity Type II (fixed – simple)

According to the figure, the theoretical maximum permissible axial load is 360 kN.
Applying a safety factor of 2, we obtain a permissible axial service load of
360 kN : 2 = 180 kN.
This lies above the maximum operating load of $F_1 = 50$kN actually occurring in service as used in our example of calculation.

Fig. 3

รูปที่ 2 ball screw permissible axial load

รูปที่ 4(ต่อ) รายละเอียดของบอลสกรู และ บอลนัต

Critical Speed

The critical speed depends on the diameter of the screw, the type of end fixity, and the free length l_n of the screw. No allowance may be made for guidance by the nut. The operating speed should not be greater than 80 % of the critical speed.

Example:
Screw diameter 63 mm
Length l_n 2.4 m
End fixity Type II (fixed – simple)

According to the figure, the critical speed is 1850 min^{-1}. The safe operating speed is thus 1850 min^{-1} x 0,8 = **1480 min^{-1}.**

The maximum operating speed in our example of calculation, $n_4 = 1000\ min^{-1}$, is thus below the safe operating speed.

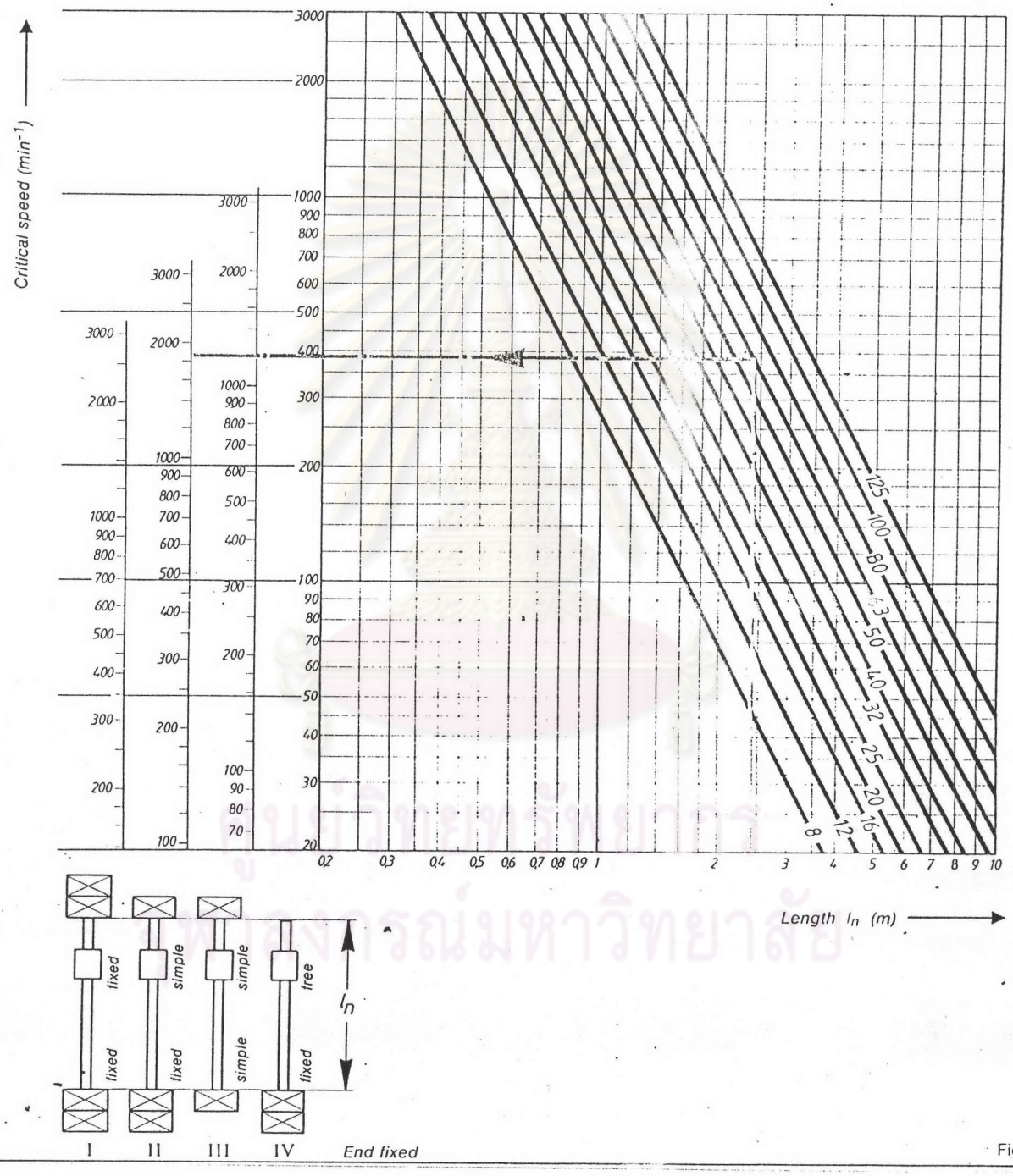


Fig. 2

รูปที่ 4(ต่อ) รายละเอียดของบอลสกรู และ บอลนัต

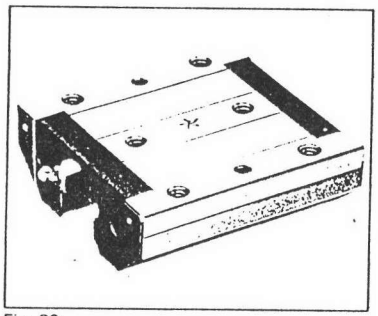
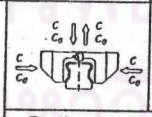


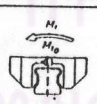
Fig. 20

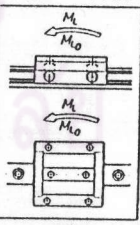
Size	Accuracy Class	Reference Number for Runner Block for Preload Class			
		up to 10 µm clearance	Preload 0,02 C	Preload 0,08 C	Preload 0,13 C
15	UP		1604-119-10	1604-129-10	1604-139-10
	SP		1604-111-10	1604-121-10	1604-131-10
	P		1604-112-10	1604-122-10	1604-132-10
	H	1604-193-10	1604-113-10	1604-123-10	
	N	1604-194-10	1604-114-10	1604-124-10	
25	UP		1604-219-10	1604-229-10	1604-239-10
	SP		1604-211-10	1604-221-10	1604-231-10
	P		1604-212-10	1604-222-10	1604-232-10
	H	1604-293-10	1604-213-10	1604-223-10	
	N	1604-294-10	1604-214-10	1604-224-10	
30	UP		1604-719-10	1604-729-10	1604-739-10
	SP		1604-711-10	1604-721-10	1604-731-10
	P		1604-712-10	1604-722-10	1604-732-10
	H	1604-793-10	1604-713-10	1604-723-10	
	N	1604-794-10	1604-714-10	1604-724-10	
35	UP		1604-319-10	1604-329-10	1604-339-10
	SP		1604-311-10	1604-321-10	1604-331-10
	P		1604-312-10	1604-322-10	1604-332-10
	H	1604-393-10	1604-313-10	1604-323-10	
	N	1604-394-10	1604-314-10	1604-324-10	
45	UP		1604-419-10	1604-429-10	1604-439-10
	SP		1604-411-10	1604-421-10	1604-431-10
	P		1604-412-10	1604-422-10	1604-432-10
	H	1604-493-10	1604-413-10	1604-423-10	
	N	1604-494-10	1604-414-10	1604-424-10	
55	UP		1604-519-10	1604-529-10	1604-539-10
	SP		1604-511-10	1604-521-10	1604-531-10
	P		1604-512-10	1604-522-10	1604-532-10
	H	1604-593-10	1604-513-10	1604-523-10	
	N	1604-594-10	1604-514-10	1604-524-10	

Table 9

Dimensions [mm]										Load Capacities [N]		All. Moments [Nm]				Weights	
N_4	E_5	E_6	S_4	N_5	T	T_1 min.	S_5	N_6	L max.	C dyn.	C_0 stat.	M_t dyn.	M_{t_0} stat.	M_L dyn.	M_{L_0} stat.	Runner Block [kg]	Rail [kg/m]
10	38	7,5	M4– 6 deep	4	60	10	4,4	10,8	3 000	5 400	11 000	52	105	30	100	0,25	1,4
18	57	5,5	M4– 6 deep	5,5	60	10	7	16	4 000	13 500	25 000	190	350	120	200	0,75	3,2
18	72	7	M4– 8 deep	6	80	12	9	17,5	4 000	19 100	34 400	330	570	190	300	1,7	5,0
21	82	7	M4– 8 deep	7	80	12	9	21	4 000	25 500	44 000	530	910	290	500	1,85	6,8
30	100	11	M5–10 deep	8	105	16	14	24	4 000	42 500	70 500	1 160	1 910	610	1 000	3,4	10,5
35	116	13	M6–10 deep	9	120	18	16	29,5	4 000	62 500	10 0000	2 000	3 200	1 070	1 700	5,65	16,2







For the purpose of comparison with the 50,000 m travel life sometimes applied for rail-type guideways, the figures for C, M_t and M_L in the table should be multiplied by 1.26.

รูปที่ 5 รายละเอียดของชุดรองเลื่อน

ภาคผนวก ค

โแกรมควบคุมการทำงานแกน Y 4 แกน

```
;*********** Variable ***********

25000 @.x_acc
25000 @.y_acc
25000 @.z_acc
25000 @.w_acc
5000 @.x_vel          ; Homing velocity of Xaxis
5000 @.y_vel          ; Homing velocity of Yaxis
5000 @.z_vel          ; Homing velocity of Zaxis
5000 @.w_vel          ; Homing velocity of Waxis
25000 @.x_dacc
25000 @.y_dacc
25000 @.z_dacc
25000 @.w_dacc
80000 @.x_bra
80000 @.y_bra
80000 @.z_bra
80000 @.w_bra
100 @.x_bra_t
100 @.y_bra_t
100 @.z_bra_t
100 @.w_bra_t
-200000 @.x_ne_ne_lim
200000 @.x_po_ne_lim
-200000 @.y_ne_ne_lim
200000 @.y_po_ne_lim
-200000 @.z_ne_ne_lim
200000 @.z_po_ne_lim
-200000 @.w_ne_ne_lim
200000 @.w_po_ne_lim
1000 @.x_pro
800 @.y_pro
750 @.z_pro
1200 @.w_pro
3000 @.x_pro_l
1700 @.y_pro_l
2500 @.z_pro_l
3000 @.w_pro_l
15 @.x_pro_lz
15 @.y_pro_lz
20 @.z_pro_lz
15 @.w_pro_lz
80 @.x_dam
80 @.y_dam
80 @.z_dam
80 @.w_dam
46000 @.x_dam_d
52000 @.y_dam_d
45000 @.z_dam_d
46000 @.w_dam_d
0 @.x_oldacc
0 @.y_oldacc
0 @.z_oldacc
0 @.w_oldacc
0 @.x_oldcrus          ; Variable to store the Cruise Velocity of Xaxis
0 @.y_oldcrus          ; Variable to store the Cruise Velocity of Yaxis
0 @.z_oldcrus          ; Variable to store the Cruise Velocity of Zaxis
0 @.w_oldcrus          ; Variable to store the Cruise Velocity of Waxis
0 @.x_olddacc
0 @.y_olddacc
0 @.z_olddacc
0 @.w_olddacc
0 @.x_oldbra
0 @.y_oldbra
0 @.z_oldbra
0 @.w_oldbra
0 @.x_oldbra_t
```

```
0 @.y_oldbra_t
0 @.z_oldbra_t
0 @.w_oldbra_t
0 @.x_ne_ol_lim
0 @.x_po_ol_lim
0 @.y_ne_ol_lim
0 @.y_po_ol_lim
0 @.z_ne_ol_lim
0 @.z_po_ol_lim
0 @.w_ne_ol_lim
0 @.w_po_ol_lim
0 @.x_oldpro
0 @.y_oldpro
0 @.z_oldpro
0 @.w_oldpro
0 @.x_oldpro_l
0 @.y_oldpro_l
0 @.z_oldpro_l
0 @.w_oldpro_l
0 @.x_oldpro_lz
0 @.y_oldpro_lz
0 @.z_oldpro_lz
0 @.w_oldpro_lz
0 @.x_olddam
0 @.y_olddam
0 @.z_olddam
0 @.w_olddam
0 @.x_olddam_d
0 @.y_olddam_d
0 @.z_olddam_d
0 @.w_olddam_d
3000 @.x_bib        ; Variable to store the bottom address
3000 @.y_bib
3000 @.z_bib
3000 @.w_bib
-18750 @.x_bit       ; Variable to store the top address
-18750 @.y_bit       ; distance = 150 mm referance at home position
-18750 @.z_bit
-18750 @.w_bit
-12500 @.x_zml       ; Variable to store the address for trig ZXF to move
left
-12500 @.y_zml       ; distance = 100 mm referance at home position
-12500 @.z_zml
-12500 @.w_zml
-8000 @.x_zil      ; Variable to store the address for tast ZXF (left
center)
-8000 @.y_zil
-8000 @.z_zil
-8000 @.w_zil
-12500 @.x_zrt      ; Variable to store the address for trig ZXF to move
center
-12500 @.y_zrt
-12500 @.z_zrt
-12500 @.w_zrt
-8000 @.x_zis       ; Variable to store the address for test ZXF (center)
-8000 @.y_zis
-8000 @.z_zis
-8000 @.w_zis
0 @.x_c1          ; Variable to store the address for center1
0 @.y_c1
0 @.z_c1
0 @.w_c1
0 @.x_c2          ; Variable to store the address for center2
0 @.y_c2
0 @.z_c2
0 @.w_c2
0 @.y_check3      ; Variable to store the address for test work
0 @.z_check3
0 @.w_check3
0 @.y_check2
0 @.z_check2
0 @.w_check2
```

```
0 @.y_check1      ; Variable to store the address for test upper limit
0 @.z_check1
0 @.w_check1
0 @.x_offset
0 @.y_offset
0 @.z_offset
0 @.w_offset
20 @.x_error
0 @.xnum_axint          ; Variable to store the number of the axis interrupt
0 @.ynum_axint
0 @.znum_axint
0 @.wnum_axint
0 @.xpt_num
0 @.ypt_num
0 @.zpt_num
0 @.wpt_num
0 @.xflag_ax
0 @.yflag_ax
0 @.zflag_ax
0 @.wflag_ax
0 @.yflag_ck23
0 @.zflag_ck23
0 @.wflag_ck23
0 @.yflag_ck1
0 @.zflag_ck1
0 @.wflag_ck1
0 @.x_distance
0 @.y_distance
0 @.z_distance
0 @.w_distance
20 @.x_err
20 @.y_err
20 @.z_err
20 @.w_err
0 @.par_inzxf
1 @.mainaxis
50 @.mainmodel
0 @.mode_au_ma
0 @.ingroup1
0 @.oldvel
1 @.pa_rs_c1
2 @.pa_rs_c2
3 @.pa_rs_fu
4 @.pa_auto
5 @.pa_manual
6 @.pa_up
7 @.pa_down
8 @.pa_stop
9 @.pa_axisx
10 @.pa_axisy
11 @.pa_axisz
12 @.pa_axisw
13 @.pa_model50
14 @.pa_model150
15 @.pa_model200
2 @.pa_bmd
4 @.pa_zis
6 @.pa_zil
8 @.pa_zir
10 @.pa_ems
13 @.pa_zml
11 @.pa_zmr
9 @.pa_zrt
7 @.pa_zreset
5 @.pa_zc1
3 @.pa_zc2
1 @.pa_zems
0 @.allerror
1000 @.veljog
0 @.old_limneg
0 @.old_limpos
0 @.testgroup
```

```
50 @.offset_lim
0 @.checklim
0 @.testlimit
0 @.teststop

;****************************************
;Function Name:   emspro
Define emspro
;------------
                    Alloff                ;turn off motor
               .pa_zems 21 Qdigout        ;ems to ZXF
  .mode_au_ma 1 ==
 Iftrue
                1 16 Digout               ;x-y no ready
              Repeat
                8 Qdigin .pa_rs_fu !=     ;wait reset fault
              Endrpt
                 .pa_zreset 21 Qdigout
                 0 16 Digout              ;x-y ready
                  0 20 Digout             ;clear zxf error
                    200 Dwell             ;delay .2 msec
                  1 20 Digout
                   &$main Restart
 Else
              Repeat
                8 Qdigin .pa_rs_fu !=     ;wait reset fault
              Endrpt
                 .pa_zreset 21 Qdigout
                0 16 Digout               ;x-y ready
                 0 20 Digout              ;clear zxf error
                    200 Dwell             ;delay .2 msec
                 1 20 Digout
 Endif
End

;****************************************
;Function Name:   driveready
Define driveready
;----------------
                  Alloff                  ;turn off motor
               .pa_zems 21 Qdigout        ;ems to ZXF
                1 16 Digout               ;x-y no ready
             Repeat
                 5 Digin
             Endrpt
End

;****************************************
;Function Name:   inrlim
Define  inrlim
;---------------
   Xaxis
     &xintde &Nullptr 1 1 Dinisr
     &xintde &Nullptr 1 2 Dinisr
      1 Dinon
      2 Dinon
   Yaxis
     &yintde &Nullptr 1 1 Dinisr
     &yintde &Nullptr 1 2 Dinisr
      1 Dinon
      2 Dinon
   Zaxis
     &zintde &Nullptr 1 1 Dinisr
     &zintde &Nullptr 1 2 Dinisr
      1 Dinon
      2 Dinon
   Waxis
     &wintde &Nullptr 1 1 Dinisr
     &wintde &Nullptr 1 2 Dinisr
      1 Dinon
      2 Dinon
End
```

```
;****************************************
;Function Name:   xintde
Define  xintde
;-------------
     0 Axset
      Motoff
End

;****************************************
;Function Name:   yintde
Define  yintde
;-------------
     1 Axset
      Motoff
End

;****************************************
;Function Name:   zintde
Define  zintde
;-------------
     2 Axset
      Motoff
End

;****************************************
;Function Name:   wintde
Define  wintde
;-------------
     3 Axset
      Motoff
End

;****************************************
;Function Name:   limhome
Define  limhome
;---------------
   Xaxis
     &xintde &Nullptr 1 3 Dinisr
      3 Dinon
   Yaxis
     &Nullptr &yintde 1 3 Dinisr
      3 Dinon
   Zaxis
     &Nullptr &zintde 1 3 Dinisr
      3 Dinon
   Waxis
     &Nullptr &wintde 1 3 Dinisr
      3 Dinon
End

;****************************************
;Function Name:   inputzxf
Define  inputzxf
;---------------
              3 Dwell
             15 Qdigin @.par_inzxf
            .par_inzxf 14 Bitand @.par_inzxf
             .par_inzxf .pa_ems ==
        Iftrue
                ems_pro
        EndIf
End

;****************************************
;Function Name:   zmlaxint
Define  zmlaxint
;---------------
  0 1 1 Axintget @.xnum_axint
  1 1 1 Axintget @.ynum_axint
  2 1 1 Axintget @.znum_axint
  3 1 1 Axintget @.wnum_axint
```

```
      &xsetflag &Nullptr .x_offset .x_zml .xnum_axint Axintadd @.xpt_num
      &ysetflag &Nullptr .y_offset .y_zml .ynum_axint Axintadd @.ypt_num
      &zsetflag &Nullptr .z_offset .z_zml .znum_axint Axintadd @.zpt_num
      &wsetflag &Nullptr .w_offset .w_zml .wnum_axint Axintadd @.wpt_num
               .xnum_axint Axinton
               .ynum_axint Axinton
               .znum_axint Axinton
               .wnum_axint Axinton
End

;****************************************
;Function Name:   zilaxint
Define  zilaxint
;---------------
  0 1 1 Axintget @.xnum_axint
  1 1 1 Axintget @.ynum_axint
  2 1 1 Axintget @.znum_axint
  3 1 1 Axintget @.wnum_axint
     &Nullptr &xsetflag .x_offset .x_zil .xnum_axint Axintadd @.xpt_num
     &Nullptr &ysetflag .y_offset .y_zil .ynum_axint Axintadd @.ypt_num
     &Nullptr &zsetflag .z_offset .z_zil .znum_axint Axintadd @.zpt_num
     &Nullptr &wsetflag .w_offset .w_zil .wnum_axint Axintadd @.wpt_num
               .xnum_axint Axinton
               .ynum_axint Axinton
               .znum_axint Axinton
               .wnum_axint Axinton
End

;****************************************
;Function Name:   check3axint
Define  check3axint
;-----------------
  1 1 1 Axintget @.ynum_axint
  2 1 1 Axintget @.znum_axint
  3 1 1 Axintget @.wnum_axint
     &ycheck23 &ycheck23 .y_offset .y_check3 .ynum_axint Axintadd @.ypt_num
     &zcheck23 &zcheck23 .z_offset .z_check3 .znum_axint Axintadd @.zpt_num
     &wcheck23 &wcheck23 .w_offset .w_check3 .wnum_axint Axintadd @.wpt_num
               .ynum_axint Axinton
               .znum_axint Axinton
               .wnum_axint Axinton
End

;****************************************
;Function Name:   check2axint
Define  check2axint
;-----------------
  1 1 1 Axintget @.ynum_axint
  2 1 1 Axintget @.znum_axint
  3 1 1 Axintget @.wnum_axint
     &ycheck23 &ycheck23 .y_offset .y_check2 .ynum_axint Axintadd @.ypt_num
     &zcheck23 &zcheck23 .z_offset .z_check2 .znum_axint Axintadd @.zpt_num
     &wcheck23 &wcheck23 .w_offset .w_check2 .wnum_axint Axintadd @.wpt_num
               .ynum_axint Axinton
               .znum_axint Axinton
               .wnum_axint Axinton
End

;****************************************
;Function Name:   check1axint
Define  check1axint
;-----------------
  1 1 1 Axintget @.ynum_axint
  2 1 1 Axintget @.znum_axint
  3 1 1 Axintget @.wnum_axint
     &ycheck1 &ycheck1 .y_offset .y_check1 .ynum_axint Axintadd @.ypt_num
     &zcheck1 &zcheck1 .z_offset .z_check1 .znum_axint Axintadd @.zpt_num
     &wcheck1 &wcheck1 .w_offset .w_check1 .wnum_axint Axintadd @.wpt_num
               .ynum_axint Axinton
               .znum_axint Axinton
               .wnum_axint Axinton
End
```

```
;****************************************
;Function Name:   zrtaxint
Define  zrtaxint
;---------------
  0 1 1 Axintget @.xnum_axint
  1 1 1 Axintget @.ynum_axint
  2 1 1 Axintget @.znum_axint
  3 1 1 Axintget @.wnum_axint
     &xsetflag &Nullptr .x_offset .x_zrt .xnum_axint Axintadd @.xpt_num
     &ysetflag &Nullptr .y_offset .y_zrt .ynum_axint Axintadd @.ypt_num
     &zsetflag &Nullptr .z_offset .z_zrt .znum_axint Axintadd @.zpt_num
     &wsetflag &Nullptr .w_offset .w_zrt .wnum_axint Axintadd @.wpt_num
              .xnum_axint Axinton
              .ynum_axint Axinton
              .znum_axint Axinton
              .wnum_axint Axinton
End

;****************************************
;Function Name:   zisaxint
Define  zisaxint
;---------------
  0 1 1 Axintget @.xnum_axint
  1 1 1 Axintget @.ynum_axint
  2 1 1 Axintget @.znum_axint
  3 1 1 Axintget @.wnum_axint
     &Nullptr &xsetflag .x_offset .x_zis .xnum_axint Axintadd @.xpt_num
     &Nullptr &ysetflag .y_offset .y_zis .ynum_axint Axintadd @.ypt_num
     &Nullptr &zsetflag .z_offset .z_zis .znum_axint Axintadd @.zpt_num
     &Nullptr &wsetflag .w_offset .w_zis .wnum_axint Axintadd @.wpt_num
              .xnum_axint Axinton
              .ynum_axint Axinton
              .znum_axint Axinton
              .wnum_axint Axinton
End

;****************************************
;Function Name:   clr_axint
Define  clr_axint
;----------------
           .xnum_axint Axintoff
           .ynum_axint Axintoff
           .znum_axint Axintoff
           .wnum_axint Axintoff
             .xpt_num .xnum_axint Axintdel
             .ypt_num .ynum_axint Axintdel
             .zpt_num .znum_axint Axintdel
             .wpt_num .wnum_axint Axintdel
              .xnum_axint Axintfree
              .ynum_axint Axintfree
              .znum_axint Axintfree
              .wnum_axint Axintfree
End

;****************************************
;Function Name:   clr_chkint
Define  clr_chkint
;-----------------
           .ynum_axint Axintoff
           .znum_axint Axintoff
           .wnum_axint Axintoff
             .ypt_num .ynum_axint Axintdel
             .zpt_num .znum_axint Axintdel
             .wpt_num .wnum_axint Axintdel
              .ynum_axint Axintfree
              .znum_axint Axintfree
              .wnum_axint Axintfree
End

;****************************************
;Function Name:   xsetflag
```

```
Define  xsetflag
;---------------
1 @.xflag_ax
End

;****************************************
;Function Name:   ysetflag
Define  ysetflag
;---------------
1 @.yflag_ax
End

;****************************************
;Function Name:   zsetflag
Define  zsetflag
;---------------
1 @.zflag_ax
End

;****************************************
;Function Name:   wsetflag
Define  wsetflag
;---------------
1 @.wflag_ax
End

;****************************************
;Function Name:   ycheck23
Define  ycheck23
;---------------

            1 @.yflag_ck23
          0 Digin Iftrue
               1 @.allerror
              "No receive PT1.2 signal(Yaxis)" Pmsg
          Endif
End

;****************************************
;Function Name:   zcheck23
Define  zcheck23
;---------------

            1 @.zflag_ck23
          1 Digin Iftrue
               1 @.allerror
              "No receive PT1.3 signal(Zaxis)" Pmsg
          Endif
End

;****************************************
;Function Name:   wcheck23
Define  wcheck23
;---------------

            1 @.wflag_ck23
          2 Digin Iftrue
               1 @.allerror
              "No receive PT1.4 signal(Waxis)" Pmsg
          Endif
End

;****************************************
;Function Name:   ycheck1
Define  ycheck1
;--------------
            1 @.yflag_ck1
          0 Digin ! Iftrue
               1 @.allerror
              "Receive PT1.2 signal(Yaxis)" Pmsg
          Endif
End
```

```
;****************************************
;Function Name:   zcheck1
Define  zcheck1
;--------------
              1 @.zflag_ck1
           1 Digin ! Iftrue
                 1 @.allerror
              "Receive PT1.3 signal(Zaxis)" Pmsg
           Endif
End

;****************************************
;Function Name:   wcheck1
Define  wcheck1
;--------------
              1 @.wflag_ck1
           2 Digin ! Iftrue
                 1 @.allerror
              "Receive PT1.4 signal(Waxis)" Pmsg
           Endif
End

;****************************************
;Function Name:   clrflag
Define  clrflag
;--------------
  0 @.xflag_ax
   0 @.yflag_ax
    0 @.zflag_ax
     0 @.wflag_ax
End

;****************************************
;Function Name:   clrfcheck23
Define  clrfcheck23
;------------------
     0 @.yflag_ck23
      0 @.zflag_ck23
       0 @.wflag_ck23
End

;****************************************
;Function Name:   clrfcheck1

Define  clrfcheck1
;-----------------
     0 @.yflag_ck1
      0 @.zflag_ck1
       0 @.wflag_ck1
End

Define system
;--------------
          Init
         Reset
         swinit
    inrlim     ;Initial interrupt of Lim+ (input A) and Lim- (input B).
    "WAIT Drive Ready" Pmsg
        Repeat
            5 Digin !          ;wait drive ready
        Endrpt
    &Nullptr &driveready 5 Inputisr
             5 Inputon
    &emspro &Nullptr 4 Inputisr     ;ems interrupt
             4 Inputon
          newpar
           15 21 Qdigout
            Enable
          Xaxis Zero 211 Ctlgetpar Go
          Yaxis Zero 211 Ctlgetpar Go
          Zaxis Zero 211 Ctlgetpar Go
```

```
          Waxis Zero 211 Ctlgetpar Go

   "***** Start *****" Pmsg
     Repeat
      Xaxis 2 Din Yaxis 2 Din And Zaxis 2 Din Waxis 2 Din And And !
     Endrpt

   "***** Stop *****" Pmsg

      "Wait ZXF goto center (zis)" Pmsg
       .pa_zreset 21 Qdigout
         0 @.par_inzxf
    Repeat
       15 Digin !              ;wait zis signal
    Endrpt
     15 21 Qdigout       ;reset fault = 0
       limhome           ;initial lim Home
  Xaxis 210 Ctlgetpar Go
  Yaxis 210 Ctlgetpar Go
  Zaxis 210 Ctlgetpar Go
  Waxis 210 Ctlgetpar Go
   "Wait home switch" Pmsg
  Repeat
   Xaxis 3 Din ! Yaxis 3 Din Or Zaxis 3 Din Waxis 3 Din Or Or
  Endrpt
      500 Dwell
       Xaxis Zero
          3 Dinoff
       Yaxis Zero
          3 Dinoff
       Zaxis Zero
          3 Dinoff
       Waxis Zero
          3 Dinoff
            oldpar
              bibgo
 "move to bib" Pmsg
    Repeat
        Xaxis Read @.x_distance
     .x_distance .x_bib .x_err - >= .x_distance .x_bib .x_err + <= And
Iftrue
           1 @.xflag_ax
             Endif
        Yaxis Read @.y_distance
     .y_distance .y_bib .y_err - >= .y_distance .y_bib .y_err + <= And
Iftrue
           1 @.yflag_ax
             Endif
         Zaxis Read @.z_distance
     .z_distance .z_bib .z_err - >= .z_distance .z_bib .z_err + <= And
Iftrue
           1 @.zflag_ax
             Endif
       Waxis Read @.w_distance
     .w_distance .w_bib .w_err - >= .w_distance .w_bib .w_err + <= And
Iftrue
           1 @.wflag_ax
             Endif
    .xflag_ax .yflag_ax And .zflag_ax .wflag_ax And And !
    Endrpt
              500 Dwell
                clrflag
                  0 16 Digout        ;x-y ready
                  0 19 Digout        ;on home position
        "BEGIN START MAIN" Pmsg
             &Nullptr &stop_inchi 12 Inputisr
                      12 Inputon
                  $main
End

;***************************
;Function name:   oldpar
```

```
Define oldpar
;--------------
    Xaxis
       .x_oldacc 100 Ctlsetpar
       .x_oldcrus 101 Ctlsetpar
       .x_olddacc 102 Ctlsetpar
       .x_oldbra 103 Ctlsetpar
       .x_oldbra_t 104 Ctlsetpar
       .x_ne_ol_lim 211 Ctlsetpar
       .x_po_ol_lim 210 Ctlsetpar
       .x_oldpro 500 Ctlsetpar
       .x_oldpro_l 501 Ctlsetpar
       .x_oldpro_lz 502 Ctlsetpar
       .x_olddam 503 Ctlsetpar
       .x_olddam_d 504 Ctlsetpar
    Yaxis
        .y_oldacc 100 Ctlsetpar
        .y_oldcrus 101 Ctlsetpar
        .y_olddacc 102 Ctlsetpar
        .y_oldbra 103 Ctlsetpar
        .y_oldbra_t 104 Ctlsetpar
        .y_ne_ol_lim 211 Ctlsetpar
        .y_po_ol_lim 210 Ctlsetpar
        .y_oldpro 500 Ctlsetpar
        .y_oldpro_l 501 Ctlsetpar
        .y_oldpro_lz 502 Ctlsetpar
        .y_olddam 503 Ctlsetpar
        .y_olddam_d 504 Ctlsetpar
    Zaxis
         .z_oldacc 100 Ctlsetpar
         .z_oldcrus 101 Ctlsetpar
         .z_olddacc 102 Ctlsetpar
         .z_oldbra 103 Ctlsetpar
         .z_oldbra_t 104 Ctlsetpar
         .z_ne_ol_lim 211 Ctlsetpar
         .z_po_ol_lim 210 Ctlsetpar
         .z_oldpro 500 Ctlsetpar
         .z_oldpro_l 501 Ctlsetpar
         .z_oldpro_lz 502 Ctlsetpar
         .z_olddam 503 Ctlsetpar
         .z_olddam_d 504 Ctlsetpar
    Waxis
          .w_oldacc 100 Ctlsetpar
          .w_oldcrus 101 Ctlsetpar
          .w_olddacc 102 Ctlsetpar
          .w_oldbra 103 Ctlsetpar
          .w_oldbra_t 104 Ctlsetpar
          .w_ne_ol_lim 211 Ctlsetpar
          .w_po_ol_lim 210 Ctlsetpar
          .w_oldpro 500 Ctlsetpar
          .w_oldpro_l 501 Ctlsetpar
          .w_oldpro_lz 502 Ctlsetpar
          .w_olddam 503 Ctlsetpar
          .w_olddam_d 504 Ctlsetpar
End

;****************************
;Function name:    $main
Define $main
;------------------
         " ****** manual ****** " Pmsg
Repeat
      0 @.mode_au_ma          ;1 = auto ,0 = manual
        Repeat
                  7 Digin Iftrue
                      $gohome
                        Endif
                  12 Digin Iftrue
                       $maininchi
                         Endif
          8 Qdigin @.ingroup1
           .ingroup1 0 ==
```

```
        Endrpt
            10 Dwell
         8 Qdigin @.testgroup
       .ingroup1 .testgroup ==
  Iftrue
              .ingroup1 1 Print
        .ingroup1 .pa_rs_c1 == Iftrue
            $resetc1
      Else .ingroup1 .pa_rs_c2 == Iftrue
            $resetc2
      Else .ingroup1 .pa_up == Iftrue
            $up
      Else .ingroup1 .pa_down == Iftrue
            $down
      Else .ingroup1 .pa_axisx == Iftrue
          "set mainaxis = 1" Pmsg
            1 @.mainaxis
            .mainaxis 1 Print
      Else .ingroup1 .pa_axisy == Iftrue
          "set mainaxis = 2" Pmsg
            2 @.mainaxis
            .mainaxis 1 Print
      Else .ingroup1 .pa_axisz == Iftrue
             "set mainaxis = 3" Pmsg
            3 @.mainaxis
            .mainaxis 1 Print
      Else .ingroup1 .pa_axisw == Iftrue
            "set mainaxis = 4" Pmsg
            4 @.mainaxis
           .mainaxis 1 Print
      Else .ingroup1 .pa_model50 == Iftrue
            "set mainmodel = 50" Pmsg
            50 @.mainmodel
           .mainmodel 1 Print
      Else .ingroup1 .pa_model150 == Iftrue
          "set mainmodel = 150" Pmsg
            150 @.mainmodel
            .mainmodel 1 Print
      Else .ingroup1 .pa_model200 == Iftrue
             "set mainmodel = 200" Pmsg
            200 @.mainmodel
             .mainmodel 1 Print
      Else  .ingroup1 .pa_auto == Iftrue
             "set auto mode" Pmsg
              1 @.mode_au_ma          ;1 = auto ,0 = manual
                 $main_auto
         Endif Endif Endif Endif Endif Endif
         Endif Endif Endif Endif Endif Endif
  Endif
    1
Endrpt
End

;*****************************
;Function name:    $resetc1
Define $resetc1
;--------------
                  15 21 Qdigout
       " RESET CENTER 1 " Pmsg
         .mainmodel 50 == Iftrue
                  "model 50 c1" Pmsg
         Else .mainmodel 150 == Iftrue
                  "model 150 c1" Pmsg
         Else .mainmodel 200 == Iftrue
                  "model 200 c1" Pmsg
         Endif Endif Endif
                    savedata
                  2000 Dwell
               .pa_zc1 21 Qdigout
               "send .pa_zc1 to zxf" Pmsg
        Repeat
           8 Qdigin @.ingroup1
```

```
            .ingroup1 .pa_rs_c1 ==
          Endrpt
End

;****************************
;Function name:     $resetc2
Define $resetc2
;--------------
                  15 21 Qdigout
        "RESET CENTER 2 " Pmsg
            .mainmodel 50 == Iftrue
             "model 50 c2" Pmsg
          Else .mainmodel 150 == Iftrue
             "model 150 c2" Pmsg
          Else .mainmodel 200 == Iftrue
            "model 200 c2" Pmsg
          Endif Endif Endif
                   savedata
                    2000 Dwell
               .pa_zc2 21 Qdigout
            "send .pa_zc2 to zxf" Pmsg
        Repeat
            8 Qdigin @.ingroup1
            .ingroup1 .pa_rs_c2 ==
         Endrpt
End

;****************************
;Function name:    savedata
Define savedata
;--------------
 "run1.nvm" Ofl
    .x_mo50c1 1 Print " @.x_mo50c1" Pmsg
    .y_mo50c1 1 Print " @.y_mo50c1" Pmsg
    .z_mo50c1 1 Print " @.z_mo50c1" Pmsg
    .w_mo50c1 1 Print " @.w_mo50c1" Pmsg
      .x_mo150c1 1 Print " @.x_mo150c1" Pmsg
      .y_mo150c1 1 Print " @.y_mo150c1" Pmsg
      .z_mo150c1 1 Print " @.z_mo150c1" Pmsg
      .w_mo150c1 1 Print " @.w_mo150c1" Pmsg
        .x_mo200c1 1 Print " @.x_mo200c1" Pmsg
        .y_mo200c1 1 Print " @.y_mo200c1" Pmsg
        .z_mo200c1 1 Print " @.z_mo200c1" Pmsg
        .w_mo200c1 1 Print " @.w_mo200c1" Pmsg
    .x_mo50c2 1 Print " @.x_mo50c2" Pmsg
    .y_mo50c2 1 Print " @.y_mo50c2" Pmsg
    .z_mo50c2 1 Print " @.z_mo50c2" Pmsg
    .w_mo50c2 1 Print " @.w_mo50c2" Pmsg
      .x_mo150c2 1 Print " @.x_mo150c2" Pmsg
      .y_mo150c2 1 Print " @.y_mo150c2" Pmsg
      .z_mo150c2 1 Print " @.z_mo150c2" Pmsg
      .w_mo150c2 1 Print " @.w_mo150c2" Pmsg
        .x_mo200c2 1 Print " @.x_mo200c2" Pmsg
        .y_mo200c2 1 Print " @.y_mo200c2" Pmsg
        .z_mo200c2 1 Print " @.z_mo200c2" Pmsg
        .w_mo200c2 1 Print " @.w_mo200c2" Pmsg
  Oclose
End

;****************************************
;Function Name:    $up
Define  $up
;----------
        "MOVE UP " Pmsg
       .mainaxis 1 == Iftrue
            Xaxis
       Else .mainaxis 2 == Iftrue
            Yaxis
       Else .mainaxis 3 == Iftrue
            Zaxis
       Else .mainaxis 4 == Iftrue
            Waxis
```

```
        Endif Endif Endif Endif
             101 Ctlgetpar @.oldvel
               .veljog 101 Ctlsetpar
                    211 Ctlgetpar Go
             211 Ctlgetpar .offset_lim + @.checklim    ; offset_lim = 50
             Repeat
                "* UP *" Pmsg
                 Read .checklim <= @.testlimit
                 8 Qdigin .pa_stop == @.teststop
                  .testlimit .teststop Or Iftrue
                        Alloff
                       .oldvel 101 Ctlsetpar
                         Return
                  Endif
               1
             Endrpt
End

;****************************************
;Function Name:    $down
Define  $down
;------------
        "MOVE DOWN" Pmsg
       .mainaxis 1 == Iftrue
            Xaxis
       Else .mainaxis 2 == Iftrue
            Yaxis
       Else .mainaxis 3 == Iftrue
            Zaxis
       Else .mainaxis 4 == Iftrue
            Waxis
       Endif Endif Endif Endif
          101 Ctlgetpar @.oldvel
           .veljog 101 Ctlsetpar
                210 Ctlgetpar Go
           210 Ctlgetpar .offset_lim - @.checklim    ; offset_lim = 50
           Repeat
                "* DOWN *" Pmsg
               Read .checklim >= @.testlimit
               8 Qdigin .pa_stop == @.teststop
                .testlimit .teststop Or Iftrue
                      Alloff
                     .oldvel 101 Ctlsetpar
                       Return
                 Endif
             1
           Endrpt
End

;****************************************
;Function Name:    $gohome
Define  $gohome
;--------------
         Enable
              "***** Go home *****" Pmsg
               newpar
              15 21 Qdigout
               clrflag
             bitgo
          Repeat
        Xaxis Read @.x_distance
     .x_distance .x_bit .x_err - >= .x_distance .x_bit .x_err + <= And
Iftrue
           1 @.xflag_ax
              Endif
        Yaxis Read @.y_distance
     .y_distance .y_bit .y_err - >= .y_distance .y_bit .y_err + <= And
Iftrue
           1 @.yflag_ax
              Endif
        Zaxis Read @.z_distance
     .z_distance .z_bit .z_err - >= .z_distance .z_bit .z_err + <= And
```

```
Iftrue
           1 @.zflag_ax
              Endif
         Waxis Read @.w_distance
     .w_distance .w_bit .w_err - >= .w_distance .w_bit .w_err + <= And
Iftrue
           1 @.wflag_ax
              Endif
     .xflag_ax .yflag_ax And .zflag_ax .wflag_ax And And !
    Endrpt
      "Wait ZXF goto center (zis)" Pmsg
         15 21 Qdigout
          600 Dwell
          .pa_zrt 21 Qdigout
    Repeat
        15 Digin !          ;wait zis signal
    Endrpt
        15 21 Qdigout      ;reset fault = 0
                  clrflag
                bibgo
           "move to bib" Pmsg
   Repeat
        Xaxis Read @.x_distance
     .x_distance .x_bib .x_err - >= .x_distance .x_bib .x_err + <= And
Iftrue
           1 @.xflag_ax
              Endif
         Yaxis Read @.y_distance
     .y_distance .y_bib .y_err - >= .y_distance .y_bib .y_err + <= And
Iftrue
           1 @.yflag_ax
              Endif
          Zaxis Read @.z_distance
     .z_distance .z_bib .z_err - >= .z_distance .z_bib .z_err + <= And
Iftrue
           1 @.zflag_ax
              Endif
         Waxis Read @.w_distance
     .w_distance .w_bib .w_err - >= .w_distance .w_bib .w_err + <= And
Iftrue
           1 @.wflag_ax
              Endif
    .xflag_ax .yflag_ax And .zflag_ax .wflag_ax And And !
    Endrpt
              oldpar
                clrflag
                  0 16 Digout          ;x-y ready
                  0 19 Digout          ;on home position
              500 Dwell
          Repeat
              7 Digin
          Endrpt
End

;****************************************
;Function Name:   $main_auto
Define $main_auto
;----------------
              " ****** auto ******" Pmsg
            0 @.par_inchi
                findc1c2
                 findcheck
 Repeat
      "WAIT AUTO OR MANUAL" Pmsg
           Repeat
                 "WAIT start tranfer" Pmsg
                    1 @.mode_au_ma
                       3 Digin Iftrue
                            $auto
                              Endif
             8 Qdigin @.ingroup1
                .ingroup1 0 ==
```

```
            Endrpt
           20 Dwell
          8 Qdigin @.testgroup
        .ingroup1 .testgroup ==
        Iftrue
             .ingroup1 .pa_manual == Iftrue
                    0 @.mode_au_ma          ;1 = auto ,0 = manual
                      Return
              Endif
        Endif
       1
 Endrpt
End

;****************************************
;Function Name:   $auto
Define $auto
;-----------
            "***** main auto *****" Pmsg
                 1 @.par_inchi
                 0 @.allerror
                 1 17 Digout         ;off finish T.F
                 1 19 Digout         ;off home position
                 15 21 Qdigout  ;clr out for zxf
                   clrflag
                  clrfcheck1
                   clrfcheck23
              "WAIT ZXF center" Pmsg
                    bitgo
               "WAIT receive bit signal" Pmsg
    Repeat
        Xaxis Read @.x_distance
     .x_distance .x_bit .x_err - >= .x_distance .x_bit .x_err + <= And
Iftrue
           1 @.xflag_ax
              Endif
        Yaxis Read @.y_distance
     .y_distance .y_bit .y_err - >= .y_distance .y_bit .y_err + <= And
Iftrue
           1 @.yflag_ax
              Endif
        Zaxis Read @.z_distance
     .z_distance .z_bit .z_err - >= .z_distance .z_bit .z_err + <= And
Iftrue
           1 @.zflag_ax
              Endif
        Waxis Read @.w_distance
     .w_distance .w_bit .w_err - >= .w_distance .w_bit .w_err + <= And
Iftrue
           1 @.wflag_ax
              Endif
     .xflag_ax .yflag_ax And .zflag_ax .wflag_ax And And !
    Endrpt
                  2 @.par_inchi
                   clrflag
          .pa_zml 21 Qdigout
     "Wait receive zil signal" Pmsg
          Repeat
           15 Digin !
          Endrpt
                15 21 Qdigout ; clr out for zxf
              0 18 Digout        ;on solinoid
                c1go              ;move to center1
                check3axint
               3 @.par_inchi
          "WAIT receive check3 signal" Pmsg
 Repeat
  .yflag_ck23 .zflag_ck23 And .wflag_ck23 And !
 Endrpt
              clr_chkint
              .allerror Iftrue
                 "Error in CHECK3 position" Pmsg
```

```
                    emspro
              Endif
            0 @.allerror
              "WAIT receive c1 signal" Pmsg
 Repeat
         Xaxis Read @.x_distance
    .x_distance .x_c1 .x_err - >= .x_distance .x_c1 .x_err + <= And Iftrue
          1 @.xflag_ax
            Endif
       Yaxis Read @.y_distance
    .y_distance .y_c1 .y_err - >= .y_distance .y_c1 .y_err + <= And Iftrue
          1 @.yflag_ax
            Endif
       Zaxis Read @.z_distance
    .z_distance .z_c1 .z_err - >= .z_distance .z_c1 .z_err + <= And Iftrue
          1 @.zflag_ax
            Endif
       Waxis Read @.w_distance
    .w_distance .w_c1 .w_err - >= .w_distance .w_c1 .w_err + <= And Iftrue
          1 @.wflag_ax
            Endif
 .xflag_ax .yflag_ax And .zflag_ax .wflag_ax And And !
 Endrpt
                4 @.par_inchi
                 clrflag
                 clrfcheck23
            "WAIT pressure switch" Pmsg
           Repeat
               6 Digin !
           Endrpt
               &Nullptr &emspro 6 Inputisr
                    6 Inputon
                  bitgo
            "WAIT bit signal" Pmsg
 Repeat
         Xaxis Read @.x_distance
    .x_distance .x_bit .x_err - >= .x_distance .x_bit .x_err + <= And
Iftrue
          1 @.xflag_ax
            Endif
       Yaxis Read @.y_distance
    .y_distance .y_bit .y_err - >= .y_distance .y_bit .y_err + <= And
Iftrue
          1 @.yflag_ax
            Endif
       Zaxis Read @.z_distance
    .z_distance .z_bit .z_err - >= .z_distance .z_bit .z_err + <= And
Iftrue
          1 @.zflag_ax
            Endif
       Waxis Read @.w_distance
    .w_distance .w_bit .w_err - >= .w_distance .w_bit .w_err + <= And
Iftrue
          1 @.wflag_ax
            Endif
 .xflag_ax .yflag_ax And .zflag_ax .wflag_ax And And !
 Endrpt
              5 @.par_inchi
                clrflag
              .pa_zmr 21 Qdigout    ;ZXF move right
           "WAIT zir (z in right center) signal" Pmsg
           ;  0 @.par_inzxf
        Repeat
           15 Digin !        ;wait zir
        Endrpt
              6 @.par_inchi
             15 21 Qdigout  ;clr out for zxf
           "WAIT bmd signal" Pmsg
                c2go
          "WAIT receive c2 signal" Pmsg
 Repeat
        Xaxis Read @.x_distance
```

```
      .x_distance .x_c2 .x_err - >= .x_distance .x_c2 .x_err + <= And Iftrue
            1 @.xflag_ax
               Endif
         Yaxis Read @.y_distance
      .y_distance .y_c2 .y_err - >= .y_distance .y_c2 .y_err + <= And Iftrue
            1 @.yflag_ax
               Endif
         Zaxis Read @.z_distance
      .z_distance .z_c2 .z_err - >= .z_distance .z_c2 .z_err + <= And Iftrue
            1 @.zflag_ax
               Endif
         Waxis Read @.w_distance
      .w_distance .w_c2 .w_err - >= .w_distance .w_c2 .w_err + <= And Iftrue
            1 @.wflag_ax
               Endif
   .xflag_ax .yflag_ax And .zflag_ax .wflag_ax And And !
   Endrpt
               " c2 allready " Pmsg
                7 @.par_inchi
                   6 Inputoff
                    clrflag
                  1 18 Digout              ;off solinoid
                   500 Dwell               ;delay .5 sec
                      check2axint
                        bitgo
                 "WAIT check2 signal" Pmsg
   Repeat
    .yflag_ck23 .zflag_ck23 And .wflag_ck23 And !
   Endrpt
                     clr_chkint
                  .allerror Iftrue
                      "Error in CHECK2 position" Pmsg
                         emspro
                  Endif
                      check1axint
                    0 @.allerror
                "WAIT check1 signal" Pmsg
   Repeat
     .yflag_ck1 .zflag_ck1 And .wflag_ck1 And !
  Endrpt
                     clr_chkint
                  .allerror Iftrue
                      "Error in CHECK1 position" Pmsg
                         emspro
                  Endif
                    0 @.allerror
                "WAIT receive bit signal" Pmsg
   Repeat
            Xaxis Read @.x_distance
      .x_distance .x_bit .x_err - >= .x_distance .x_bit .x_err + <= And
Iftrue
            1 @.xflag_ax
               Endif
         Yaxis Read @.y_distance
      .y_distance .y_bit .y_err - >= .y_distance .y_bit .y_err + <= And
Iftrue
            1 @.yflag_ax
               Endif
         Zaxis Read @.z_distance
      .z_distance .z_bit .z_err - >= .z_distance .z_bit .z_err + <= And
Iftrue
            1 @.zflag_ax
               Endif
         Waxis Read @.w_distance
      .w_distance .w_bit .w_err - >= .w_distance .w_bit .w_err + <= And
Iftrue
            1 @.wflag_ax
               Endif
   .xflag_ax .yflag_ax And .zflag_ax .wflag_ax And And !
   Endrpt
                8 @.par_inchi
                    clrflag
```

```
                    clrfcheck1
                    clrfcheck23
                .pa_zrt 21 Qdigout
     "Wait receive zis signal" Pmsg
           Repeat
            15 Digin !
           Endrpt
                15 21 Qdigout     ;clr out for zxf
                  bibgo              ;move to start point
               9 @.par_inchi
            "WAIT receive bib signal" Pmsg
  Repeat
        Xaxis Read @.x_distance
     .x_distance .x_bib .x_err - >= .x_distance .x_bib .x_err + <= And
Iftrue
           1 @.xflag_ax
              Endif
        Yaxis Read @.y_distance
     .y_distance .y_bib .y_err - >= .y_distance .y_bib .y_err + <= And
Iftrue
           1 @.yflag_ax
              Endif
        Zaxis Read @.z_distance
     .z_distance .z_bib .z_err - >= .z_distance .z_bib .z_err + <= And
Iftrue
           1 @.zflag_ax
              Endif
        Waxis Read @.w_distance
     .w_distance .w_bib .w_err - >= .w_distance .w_bib .w_err + <= And
Iftrue
           1 @.wflag_ax
              Endif
        .xflag_ax .yflag_ax And .zflag_ax .wflag_ax And And !
  Endrpt
                clrflag
                 0 17 Digout        ;on finish T.F
                 0 19 Digout        ;on home position
End

;****************************************
;Function Name:   $maininchi
Define $maininchi
;---------------

             " ******** maininchi ********" Pmsg
  .mode_au_ma 0 ==
 Iftrue
                  clrflag
                       Xaxis Read @.x_distance
     .x_distance .x_bib .x_err - >= .x_distance .x_bib .x_err + <= And
Iftrue
           1 @.xflag_ax
              Endif
        Yaxis Read @.y_distance
     .y_distance .y_bib .y_err - >= .y_distance .y_bib .y_err + <= And
Iftrue
           1 @.yflag_ax
              Endif
        Zaxis Read @.z_distance
     .z_distance .z_bib .z_err - >= .z_distance .z_bib .z_err + <= And
Iftrue
           1 @.zflag_ax
              Endif
        Waxis Read @.w_distance
     .w_distance .w_bib .w_err - >= .w_distance .w_bib .w_err + <= And
Iftrue
           1 @.wflag_ax
              Endif
         .xflag_ax .yflag_ax And .zflag_ax .wflag_ax And And ! @.testflag
                       .testflag
              Iftrue
                  clrflag
```

```
                    Return
                Endif
                0 @.par_inchi
  Endif
                 findc1c2
       Repeat
                 1 17 Digout          ;off finish T.F
                 1 19 Digout          ;off home position
                 15 21 Qdigout        ;clr out for zxf
                   clrflag
                0 @.par_inzxf
             .par_inchi 1 <= Iftrue
                    inchi_1
            Endif
             .par_inchi 2 <= Iftrue
                      inchi_2
                   0 18 Digout     ;on solinoid
                    500 Dwell
            Endif
             .par_inchi 3 <= Iftrue
                      inchi_3
            Endif
             .par_inchi 4 <= Iftrue
                      inchi_4
            Endif
             .par_inchi 5 <= Iftrue
                      inchi_5
            Endif
             .par_inchi 6 <= Iftrue
                      inchi_6
                   1 18 Digout           ;off solinoid
                   500 Dwell            ;delay .5 sec
            Endif
             .par_inchi 7 <= Iftrue
                      inchi_7
            Endif
             .par_inchi 8 <= Iftrue
                      inchi_8
            Endif
             .par_inchi 9 <= Iftrue
                      inchi_9
                   0 17 Digout          ;on finish T.F
                   0 19 Digout          ;on home position
            Endif
                 1 @.par_inchi
                   500 Dwell
             1
      Endrpt
End

;*************************************
;Function Name:   inchi_1
Define inchi_1
;-------------
                 1 @.par_inchi   ;step 1 move to bit
                  bitgo
              "WAIT receive bit signal" Pmsg
    Repeat
            " .inchi_1 " Pmsg
        Xaxis Read @.x_distance
     .x_distance .x_bit .x_err - >= .x_distance .x_bit .x_err + <= And
Iftrue
           1 @.xflag_ax
              Endif
        Yaxis Read @.y_distance
     .y_distance .y_bit .y_err - >= .y_distance .y_bit .y_err + <= And
Iftrue
           1 @.yflag_ax
              Endif
        Zaxis Read @.z_distance
     .z_distance .z_bit .z_err - >= .z_distance .z_bit .z_err + <= And
Iftrue
```

```
            1 @.zflag_ax
               Endif
          Waxis Read @.w_distance
      .w_distance .w_bit .w_err - >= .w_distance .w_bit .w_err + <= And
Iftrue
            1 @.wflag_ax
               Endif
      .xflag_ax .yflag_ax And .zflag_ax .wflag_ax And And !
     Endrpt
                  clrflag
End

;****************************************
;Function Name:   inchi_2
Define inchi_2
;-------------
             2 @.par_inchi
            .pa_zml 21 Qdigout      ;zxf move left
      "Wait receive zil(z in left center) signal" Pmsg
          Repeat
              " .inchi_2 " Pmsg
                15 Digin !
          Endrpt
                15 21 Qdigout      ; clr out for zxf
End

;****************************************
;Function Name:   inchi_3
Define inchi_3
;-------------
               3 @.par_inchi
                  c1go                ;move to center1
               " WAIT receive c1 signal" Pmsg
  Repeat
       " .inchi_3 " Pmsg
          Xaxis Read @.x_distance
      .x_distance .x_c1 .x_err - >= .x_distance .x_c1 .x_err + <= And Iftrue
            1 @.xflag_ax
               Endif
          Yaxis Read @.y_distance
      .y_distance .y_c1 .y_err - >= .y_distance .y_c1 .y_err + <= And Iftrue
            1 @.yflag_ax
               Endif
          Zaxis Read @.z_distance
      .z_distance .z_c1 .z_err - >= .z_distance .z_c1 .z_err + <= And Iftrue
            1 @.zflag_ax
               Endif
          Waxis Read @.w_distance
      .w_distance .w_c1 .w_err - >= .w_distance .w_c1 .w_err + <= And Iftrue
            1 @.wflag_ax
               Endif
  .xflag_ax .yflag_ax And .zflag_ax .wflag_ax And And !
  Endrpt
                  clrflag
End

;****************************************
;Function Name:   inchi_4
Define inchi_4
;-------------
               4 @.par_inchi
                  bitgo
               "WAIT bit signal" Pmsg
  Repeat
            " .inchi_4 " Pmsg
           Xaxis Read @.x_distance
      .x_distance .x_bit .x_err - >= .x_distance .x_bit .x_err + <= And
Iftrue
            1 @.xflag_ax
               Endif
          Yaxis Read @.y_distance
      .y_distance .y_bit .y_err - >= .y_distance .y_bit .y_err + <= And
```

```
Iftrue
           1 @.yflag_ax
              Endif
         Zaxis Read @.z_distance
     .z_distance .z_bit .z_err - >= .z_distance .z_bit .z_err + <= And
Iftrue
           1 @.zflag_ax
              Endif
         Waxis Read @.w_distance
     .w_distance .w_bit .w_err - >= .w_distance .w_bit .w_err + <= And
Iftrue
           1 @.wflag_ax
              Endif
  .xflag_ax .yflag_ax And .zflag_ax .wflag_ax And And !
  Endrpt
                  clrflag
End

;****************************************
;Function Name:   inchi_5
Define inchi_5
;-------------
                5 @.par_inchi
                  .pa_zmr 21 Qdigout    ;ZXF move right
               "WAIT zir (z in right center) signal" Pmsg
               ;  0 @.par_inzxf
          Repeat
                " .inchi_5 " Pmsg
            15 Digin !
           Endrpt
                15 21 Qdigout  ;clr out for zxf
End

;****************************************
;Function Name:   inchi_6
Define inchi_6
;-------------
                6 @.par_inchi
                     c2go
             "WAIT receive c2 signal" Pmsg
  Repeat
        " .inchi_6 " Pmsg
          Xaxis Read @.x_distance
     .x_distance .x_c2 .x_err - >= .x_distance .x_c2 .x_err + <= And Iftrue
           1 @.xflag_ax
              Endif
        Yaxis Read @.y_distance
     .y_distance .y_c2 .y_err - >= .y_distance .y_c2 .y_err + <= And Iftrue
           1 @.yflag_ax
              Endif
        Zaxis Read @.z_distance
     .z_distance .z_c2 .z_err - >= .z_distance .z_c2 .z_err + <= And Iftrue
           1 @.zflag_ax
              Endif
        Waxis Read @.w_distance
     .w_distance .w_c2 .w_err - >= .w_distance .w_c2 .w_err + <= And Iftrue
           1 @.wflag_ax
              Endif
  .xflag_ax .yflag_ax And .zflag_ax .wflag_ax And And !
  Endrpt
              " c2 allready " Pmsg
                  clrflag
End

;****************************************
;Function Name:   inchi_7
Define inchi_7
;-------------
                  7 @.par_inchi
                     bitgo
             "WAIT receive bit signal" Pmsg
  Repeat
```

```
       " .inchi_7 " Pmsg
            Xaxis Read @.x_distance
     .x_distance .x_bit .x_err - >= .x_distance .x_bit .x_err + <= And
Iftrue
           1 @.xflag_ax
              Endif
         Yaxis Read @.y_distance
     .y_distance .y_bit .y_err - >= .y_distance .y_bit .y_err + <= And
Iftrue
           1 @.yflag_ax
              Endif
         Zaxis Read @.z_distance
     .z_distance .z_bit .z_err - >= .z_distance .z_bit .z_err + <= And
Iftrue
           1 @.zflag_ax
              Endif
         Waxis Read @.w_distance
     .w_distance .w_bit .w_err - >= .w_distance .w_bit .w_err + <= And
Iftrue
           1 @.wflag_ax
              Endif
  .xflag_ax .yflag_ax And .zflag_ax .wflag_ax And And !
  Endrpt
                   clrflag
End

;****************************************
;Function Name:   inchi_8
Define inchi_8
;-------------
               8 @.par_inchi
            .pa_zrt 21 Qdigout     ;zxf move return
          " wait receive zis signal " Pmsg
                Repeat
              " .inchi_8 " Pmsg
                 15 Digin !
              Endrpt
           15 21 Qdigout
End

;****************************************
;Function Name:   inchi_9
Define inchi_9
;-------------
              9 @.par_inchi
               bibgo              ;move to start point
                 "WAIT receive bib signal" Pmsg
  Repeat
          " .inchi_9 " Pmsg
       Xaxis Read @.x_distance
     .x_distance .x_bib .x_err - >= .x_distance .x_bib .x_err + <= And
Iftrue
           1 @.xflag_ax
              Endif
         Yaxis Read @.y_distance
     .y_distance .y_bib .y_err - >= .y_distance .y_bib .y_err + <= And
Iftrue
           1 @.yflag_ax
              Endif
         Zaxis Read @.z_distance
     .z_distance .z_bib .z_err - >= .z_distance .z_bib .z_err + <= And
Iftrue
           1 @.zflag_ax
              Endif
         Waxis Read @.w_distance
     .w_distance .w_bib .w_err - >= .w_distance .w_bib .w_err + <= And
Iftrue
           1 @.wflag_ax
              Endif
        .xflag_ax .yflag_ax And .zflag_ax .wflag_ax And And !
  Endrpt
                   clrflag
```

```
End

;****************************************
;Function Name:   findc1c2
Define  findc1c2
;---------------
             .mainmodel 50 == Iftrue
                    .x_mo50c1  @.x_c1
                    .y_mo50c1  @.y_c1
                    .z_mo50c1  @.z_c1
                    .w_mo50c1  @.w_c1
                       .x_mo50c2  @.x_c2
                       .y_mo50c2  @.y_c2
                       .z_mo50c2  @.z_c2
                       .w_mo50c2  @.w_c2
       Else  .mainmodel 150 == Iftrue
                    .x_mo150c1  @.x_c1
                    .y_mo150c1  @.y_c1
                    .z_mo150c1  @.z_c1
                    .w_mo150c1  @.w_c1
                       .x_mo150c2  @.x_c2
                       .y_mo150c2  @.y_c2
                       .z_mo150c2  @.z_c2
                       .w_mo150c2  @.w_c2
       Else  .mainmodel 200 == Iftrue
                    .x_mo200c1  @.x_c1
                    .y_mo200c1  @.y_c1
                    .z_mo200c1  @.z_c1
                    .w_mo200c1  @.w_c1
                       .x_mo200c2  @.x_c2
                       .y_mo200c2  @.y_c2
                       .z_mo200c2  @.z_c2
                       .w_mo200c2  @.w_c2
       Endif Endif Endif
End

;****************************************
;Function Name:   findcheck
Define  findcheck
;----------------
       .y_c1 2500 - @.y_check3
       .z_c1 2500 - @.z_check3
       .w_c1 2500 - @.w_check3
          .y_c2 2000 - @.y_check2
          .z_c2 2000 - @.z_check2
          .w_c2 2000 - @.w_check2
             .y_c2 3100 - @.y_check1
             .z_c2 3100 - @.z_check1
             .w_c2 3100 - @.w_check1
End

;****************************************
;Function Name:   bitgo
Define bitgo
;-----------
              Xaxis .x_bit Go
              Yaxis .y_bit Go
              Zaxis .z_bit Go
              Waxis .w_bit Go
End

;****************************************
;Function Name:   c1go
Define c1go
;----------
              Xaxis .x_c1 Go
              Yaxis .y_c1 Go
              Zaxis .z_c1 Go
              Waxis .w_c1 Go
End

;****************************************
```

```
;Function Name:   bibgo
Define bibgo
;-----------
              Xaxis .x_bib Go
              Yaxis .y_bib Go
              Zaxis .z_bib Go
              Waxis .w_bib Go
End

;*****************************************
;Function Name:   c2go
Define c2go
;----------
              Xaxis .x_c2 Go
              Yaxis .y_c2 Go
              Zaxis .z_c2 Go
              Waxis .w_c2 Go
End


;*****************************************
;Function Name:   stop_inchi
Define stop_inchi
;----------------
                  Alloff                ;turn off motor
              .pa_zems 21 Qdigout      ;ems to ZXF
      Repeat
        12 Digin
         Iftrue
                15 21 Qdigout  ;clr out for zxf
                       500 Dwell
                     .par_inchi 1 == Iftrue
                       bitgo                    ;move to bit
                      ".par_inchi 1(bitgo)" Pmsg
                Else .par_inchi 2 == Iftrue
                       .pa_zml 21 Qdigout      ;zxf move left
                       ".par_inchi 2(.pa_zml)" Pmsg
                Else .par_inchi 3 == Iftrue
                       c1go                     ;move to center1
                       ".par_inchi 3(c1go)" Pmsg
                Else .par_inchi 4 == Iftrue
                       bitgo                    ;move to bit
                        ".par_inchi 4(bitgo)" Pmsg
                Else .par_inchi 5 == Iftrue
                       .pa_zmr 21 Qdigout      ;zxf move right
                        ".par_inchi 5(pa_zmr)" Pmsg
                Else .par_inchi 6 == Iftrue
                       c2go                     ;move to center2
                        ".par_inchi 6(c2go)" Pmsg
                Else .par_inchi 7 == Iftrue
                       bitgo                    ;move to bit
                        ".par_inchi 7(bitgo)" Pmsg
                Else .par_inchi 8 == Iftrue
                       .pa_zrt 21 Qdigout      ;zxf move return
                         500 Dwell
                        ".par_inchi 8(.pa_zrt)" Pmsg
                Else .par_inchi 9 == Iftrue
                       bibgo                    ;move to start point
                        ".par_inchi 9(bibgo)" Pmsg
                Endif Endif Endif Endif
                Endif Endif Endif Endif Endif
                       Return
         Else
              8 Qdigin @.ingroup1
          .ingroup1 .pa_manual == Iftrue
                 &$main Restart
                   Endif
         Endif
          1
      Endrpt
End
```

```
;****************************************
;Function Name:   swinit
Define  swinit
;-------------
   0 Ioset
   0 0 Iodir          ;Set I/O 0-7 to be input
   0 1 Iodir          ;Set I/O 8-15 to be input
   1 2 Iodir          ;Set I/O 16-23 to be output
   255 16 Odigout     ;Clear I/O 16-23
End

;****************************************
;Function Name:   newpar
Define  newpar
;-------------
Xaxis
100 Ctlgetpar @.x_oldacc
101 Ctlgetpar @.x_oldcrus     ; Limit and Cruise Velocity of Xaxis.
102 Ctlgetpar @.x_olddacc
103 Ctlgetpar @.x_oldbra
104 Ctlgetpar @.x_oldbra_t
211 Ctlgetpar @.x_ne_ol_lim
210 Ctlgetpar @.x_po_ol_lim
500 Ctlgetpar @.x_oldpro
501 Ctlgetpar @.x_oldpro_l
502 Ctlgetpar @.x_oldpro_lz
503 Ctlgetpar @.x_olddam
504 Ctlgetpar @.x_olddam_d
.x_acc 100 Ctlsetpar
.x_vel 101 Ctlsetpar          ; Lower the Cruise Velocity of Xaxis to the
                              ; homing speed
.x_dacc 102 Ctlsetpar
.x_bra 103 Ctlsetpar
.x_bra_t 104 Ctlsetpar
.x_ne_ne_lim 211 Ctlsetpar
.x_po_ne_lim 210 Ctlsetpar
.x_pro 500 Ctlsetpar
.x_pro_l 501 Ctlsetpar
.x_pro_lz 502 Ctlsetpar
.x_dam 503 Ctlsetpar
.x_dam_d 504 Ctlsetpar

Yaxis
100 Ctlgetpar @.y_oldacc
101 Ctlgetpar @.y_oldcrus     ; Limit and Cruise Velocity of Yaxis.
102 Ctlgetpar @.y_olddacc
103 Ctlgetpar @.y_oldbra
104 Ctlgetpar @.y_oldbra_t
211 Ctlgetpar @.y_ne_ol_lim
210 Ctlgetpar @.y_po_ol_lim
500 Ctlgetpar @.y_oldpro
501 Ctlgetpar @.y_oldpro_l
502 Ctlgetpar @.y_oldpro_lz
503 Ctlgetpar @.y_olddam
504 Ctlgetpar @.y_olddam_d

.y_acc 100 Ctlsetpar
.y_vel 101 Ctlsetpar          ; Lower the Cruise Velocity of Yaxis to the
                              ; homing speed
.y_dacc 102 Ctlsetpar
.y_bra 103 Ctlsetpar
.y_bra_t 104 Ctlsetpar
.y_ne_ne_lim 211 Ctlsetpar
.y_po_ne_lim 210 Ctlsetpar
.y_pro 500 Ctlsetpar
.y_pro_l 501 Ctlsetpar
.y_pro_lz 502 Ctlsetpar
.y_dam 503 Ctlsetpar
.y_dam_d 504 Ctlsetpar

Zaxis
100 Ctlgetpar @.z_oldacc
```

```
101 Ctlgetpar @.z_oldcrus     ; Limit and Cruise Velocity of Zaxis.
102 Ctlgetpar @.z_olddacc
103 Ctlgetpar @.z_oldbra
104 Ctlgetpar @.z_oldbra_t
211 Ctlgetpar @.z_ne_ol_lim
210 Ctlgetpar @.z_po_ol_lim
500 Ctlgetpar @.z_oldpro
501 Ctlgetpar @.z_oldpro_l
502 Ctlgetpar @.z_oldpro_lz
503 Ctlgetpar @.z_olddam
504 Ctlgetpar @.z_olddam_d

.z_acc 100 Ctlsetpar
.z_vel 101 Ctlsetpar          ; Lower the Cruise Velocity of Zaxis to the
                              ; homing speed
.z_dacc 102 Ctlsetpar
.z_bra 103 Ctlsetpar
.z_bra_t 104 Ctlsetpar
.z_ne_ne_lim 211 Ctlsetpar
.z_po_ne_lim 210 Ctlsetpar
.z_pro 500 Ctlsetpar
.z_pro_l 501 Ctlsetpar
.z_pro_lz 502 Ctlsetpar
.z_dam 503 Ctlsetpar
.z_dam_d 504 Ctlsetpar

Waxis
100 Ctlgetpar @.w_oldacc
101 Ctlgetpar @.w_oldcrus     ; Limit and Cruise Velocity of Waxis.
102 Ctlgetpar @.w_olddacc
103 Ctlgetpar @.w_oldbra
104 Ctlgetpar @.w_oldbra_t
211 Ctlgetpar @.w_ne_ol_lim
210 Ctlgetpar @.w_po_ol_lim
500 Ctlgetpar @.w_oldpro
501 Ctlgetpar @.w_oldpro_l
502 Ctlgetpar @.w_oldpro_lz
503 Ctlgetpar @.w_olddam
504 Ctlgetpar @.w_olddam_d

.w_acc 100 Ctlsetpar
.w_vel 101 Ctlsetpar          ; Lower the Cruise Velocity of Waxis to the
                              ; homing speed
.w_dacc 102 Ctlsetpar
.w_bra 103 Ctlsetpar
.w_bra_t 104 Ctlsetpar
.w_ne_ne_lim 211 Ctlsetpar
.w_po_ne_lim 210 Ctlsetpar
.w_pro 500 Ctlsetpar
.w_pro_l 501 Ctlsetpar
.w_pro_lz 502 Ctlsetpar
.w_dam 503 Ctlsetpar
.w_dam_d 504 Ctlsetpar
End

;*************************************

"run1.nvm" Ifl
 &system Restart
```

ภาคผนวก ง

โปรแกรมควบคุมการทำงาน เซอร์โวมอเตอร์แกน X

```
1XE100
1XD100

MN                  ; ABSOLUTE MODE
MPA                 ;POWER ON EXCECUTE SEQUENCE
SSJ1
SSH1
OSI1
SN20
XQ1
LD0
O000                ; RESET OUTPUT

CPP50               ; TUNING PAARAMETER
CPI10
CTG60
CPD60
CVP30
CVI0
CVF60
;SEQUENCE SELECT
IN1J                ; JOG CW
IN2K                ; JOG CCW
IN3B                ; ZML  001     RF  100
IN4B                ; ZMR  010      RC1 101
IN5B                ; ZTR  011      RC2 110
IN7D                ; STOP MOTION (EMS)

JA25                ; INIT JOG MODE
```

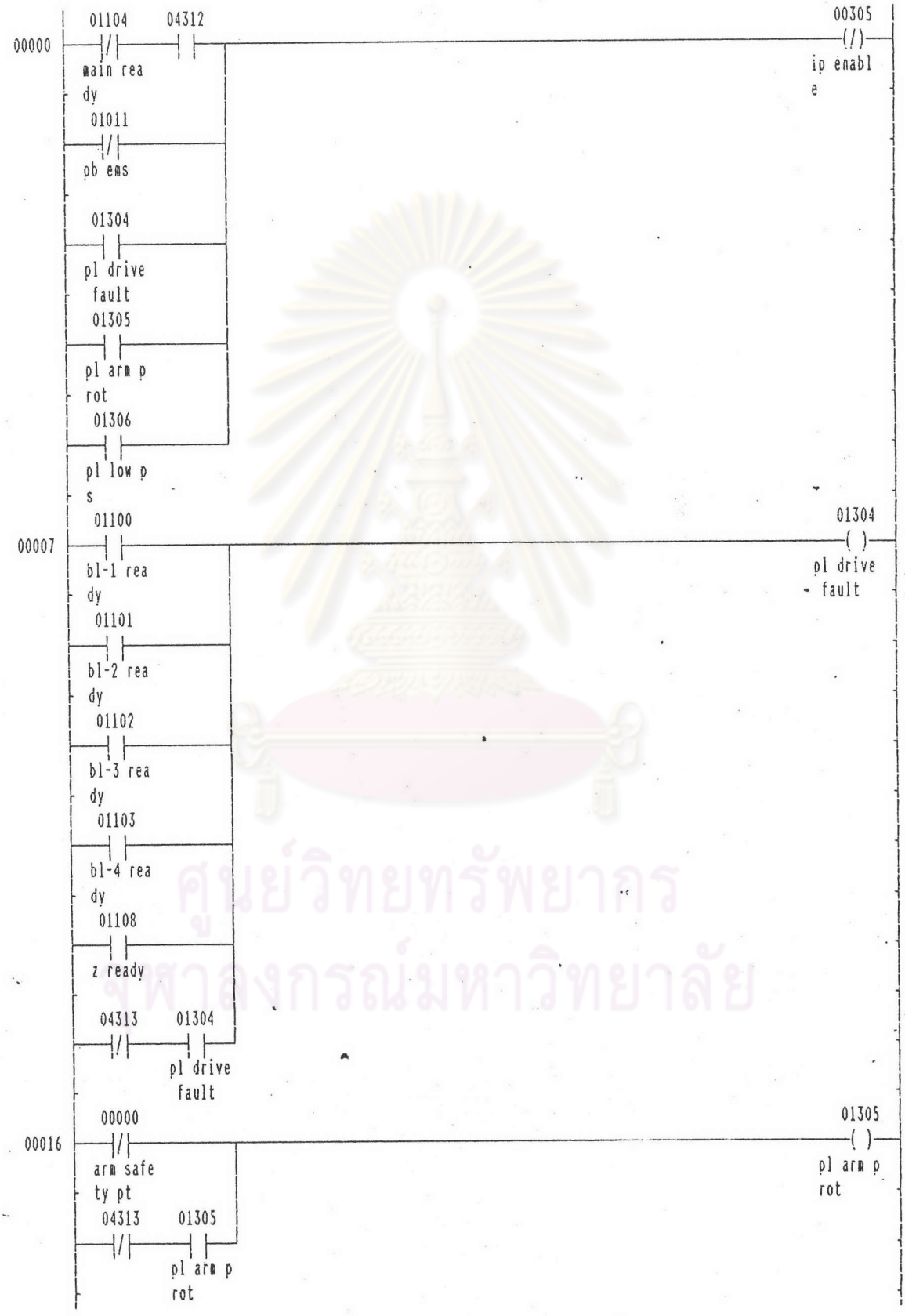
```
JAD25
OSE1
JVL3



INL1
OUTL1
OUT1F                    ; FAULT

1XT

;*******************************************
1XE16                    ;FIND HOME
1XD16

O000                     ; RESET OUTPUT

OSB1
OSG1
OSH1
GHA10
GHAD12
GHF2
GH-5
PZ                       ; RESET HOME

D5000
G
PZ

O001
```

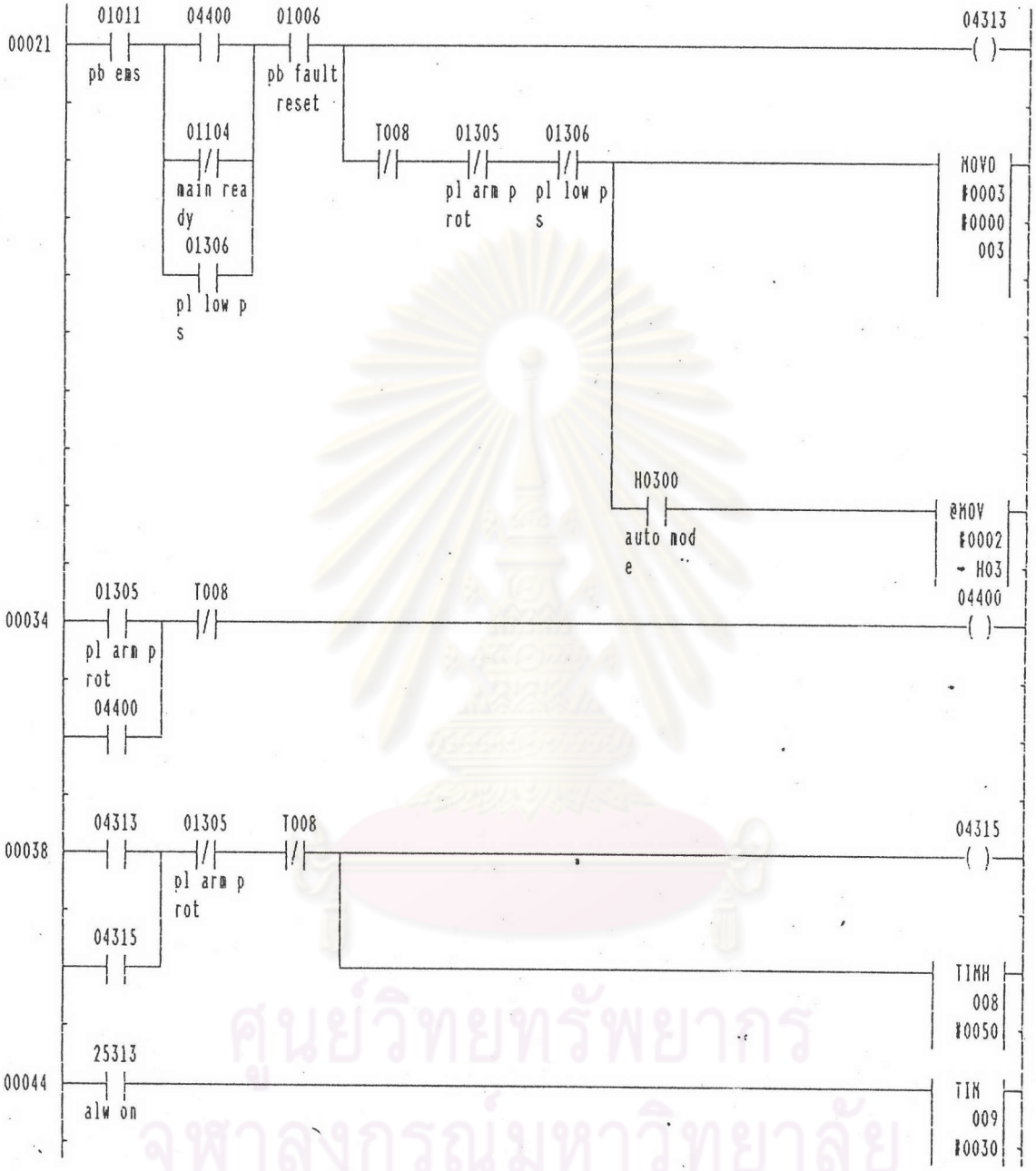
```
T0.3
O000

1XT

;*********************************************
1XE4                    ; ZML
1XD4


A120
AD110
V80
D(VAR5)                 ;-137500
G                       ; GD1
O001
T0.1
O000

1XT

;*********************************************
1XE8                    ; ZMR
1XD8

A120
AD110
V80
D(VAR6)                 ;137500
G
O001
T0.1
O000
```

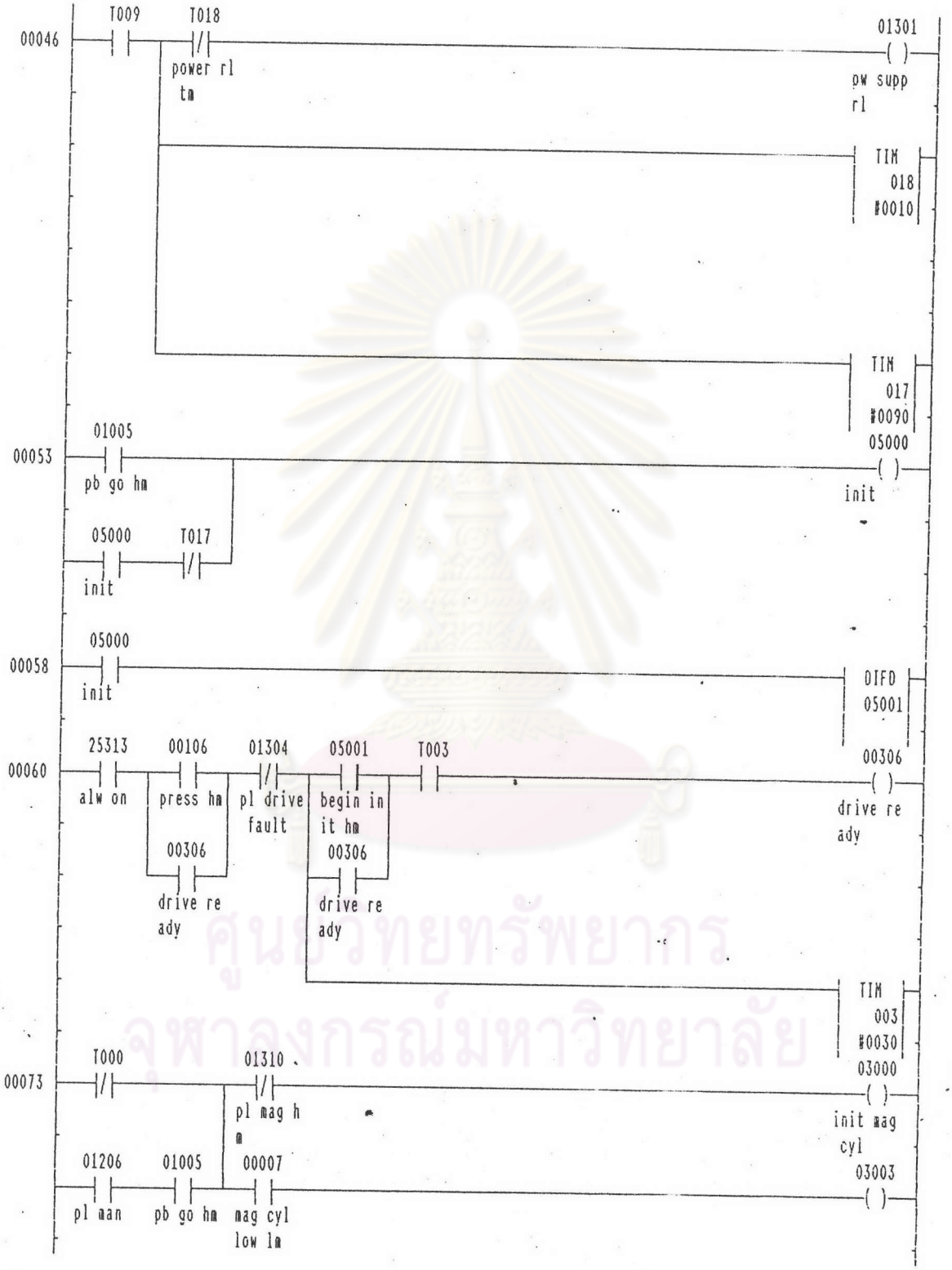
```
1XT

;*******************************************
1XE12                   ; ZRT
1XD12

A120
AD110
V80
D-3000
G

O001
T0.1
O000

1XT

;*******************************************
1XE20                   ;RESET C1   LEFT
1XD20

1VAR5=POS

1XT

;*******************************************
1XE24                   ;RESET C2   RIGHT
1XD24

1VAR6=POS
```

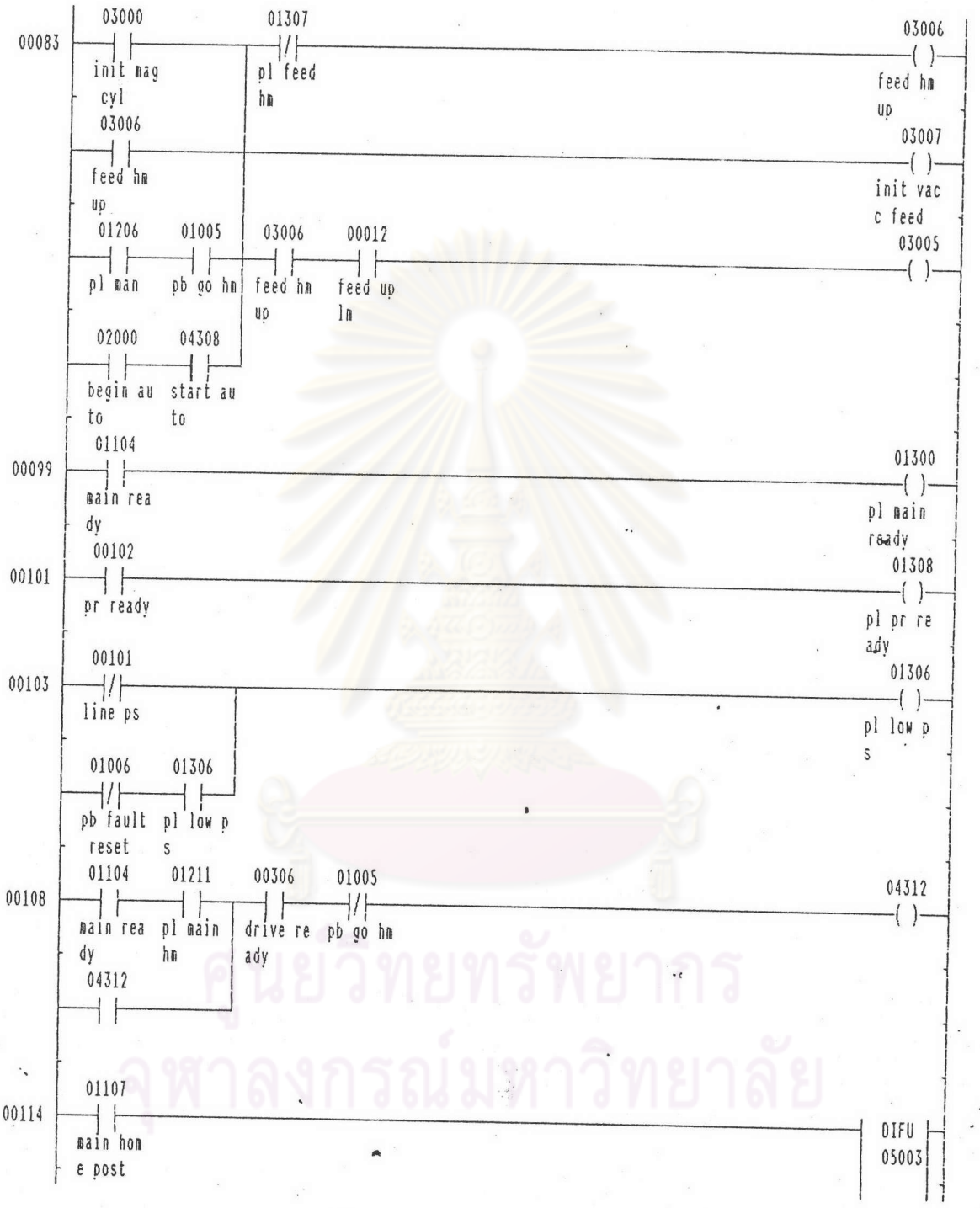
```
1XT


;*********************************************
1XE99                    ; DEFAULT  VALUE
1XD99

1VAR1=5000
1VAR5=-130000
1VAR6=137500

1XT

XT
```

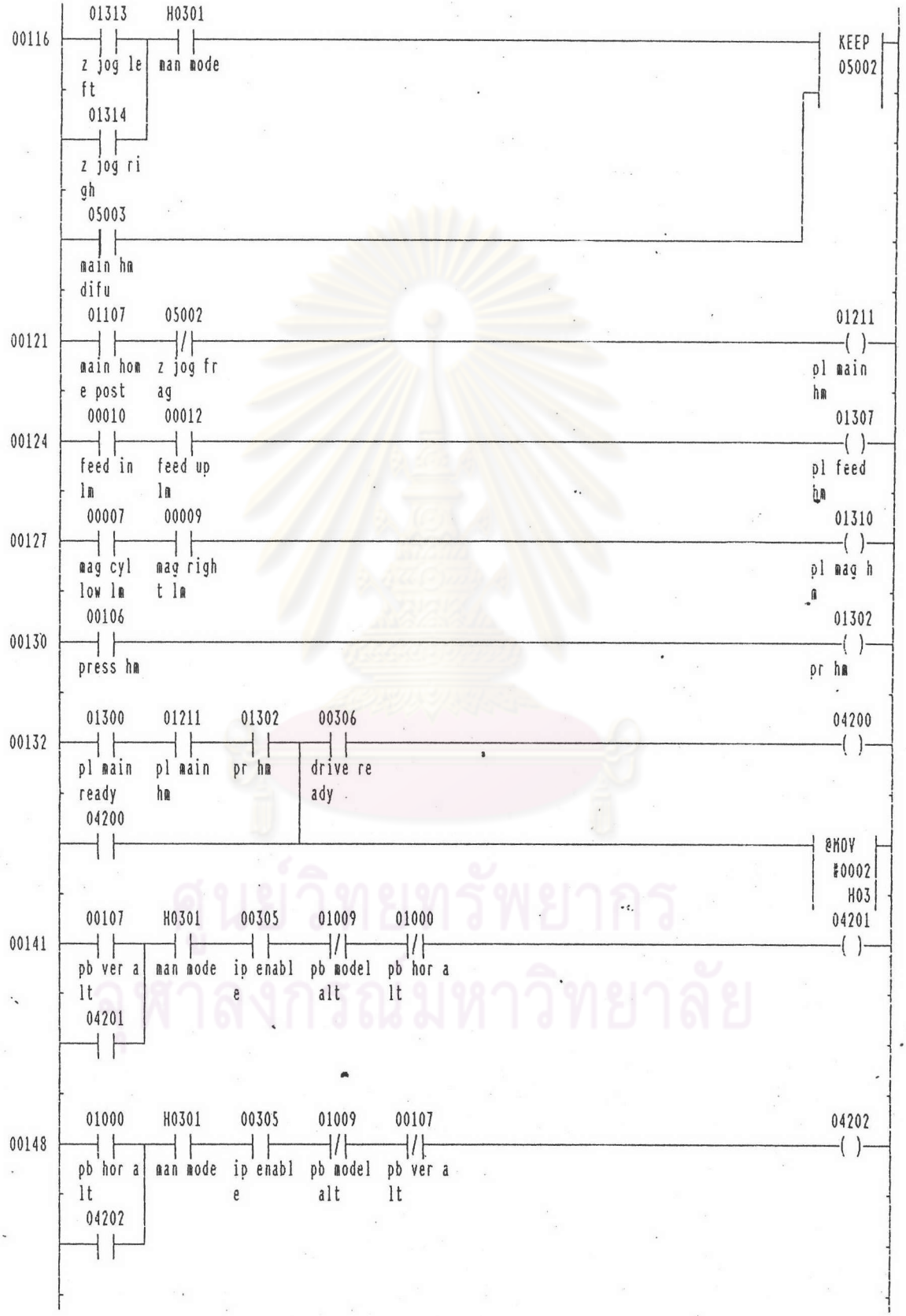
โปรแกรมการควบคุมการทำงานของโปรแกรมมาเบิล ลอจิคอล คอนโทรลเลอร์

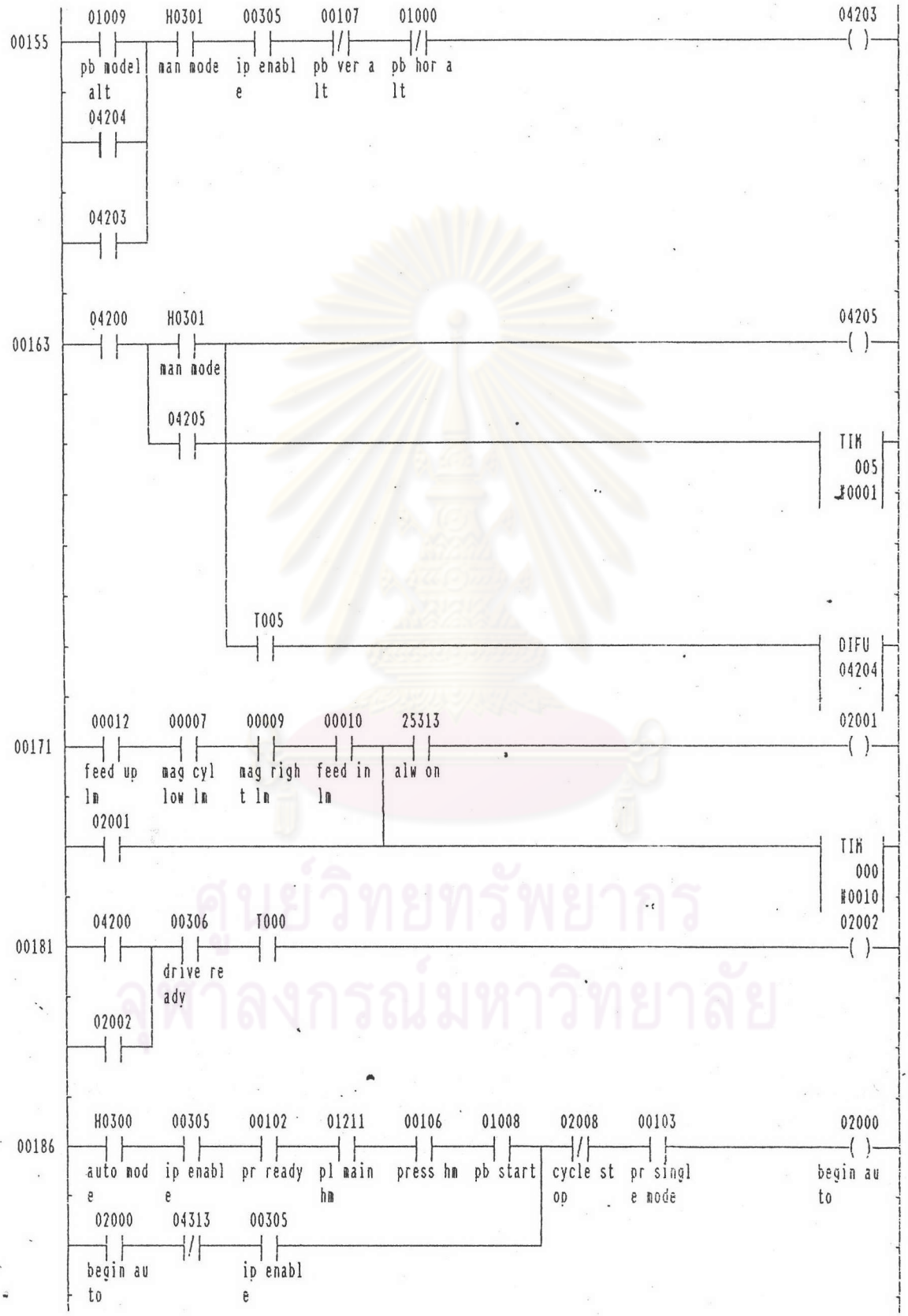


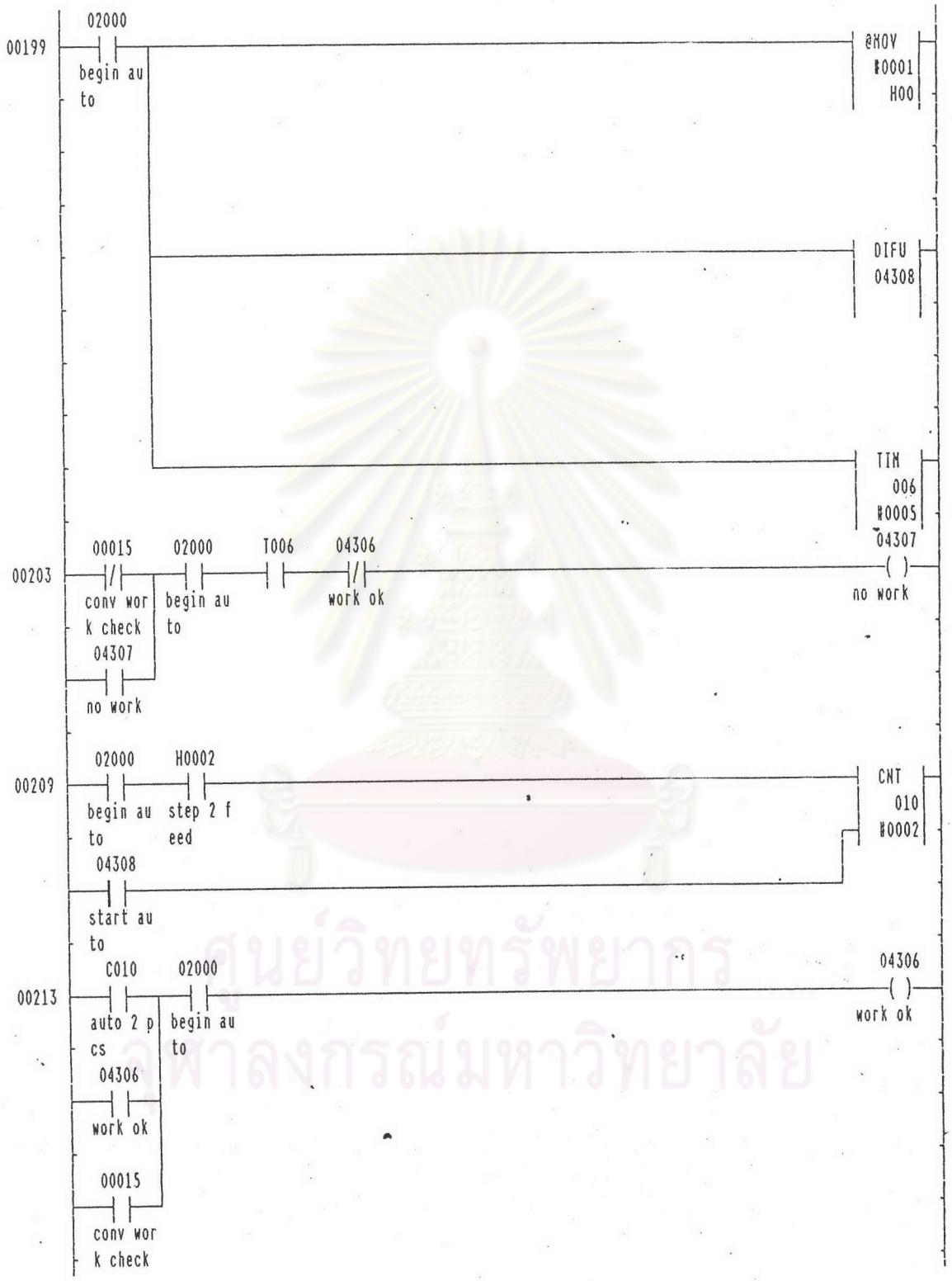


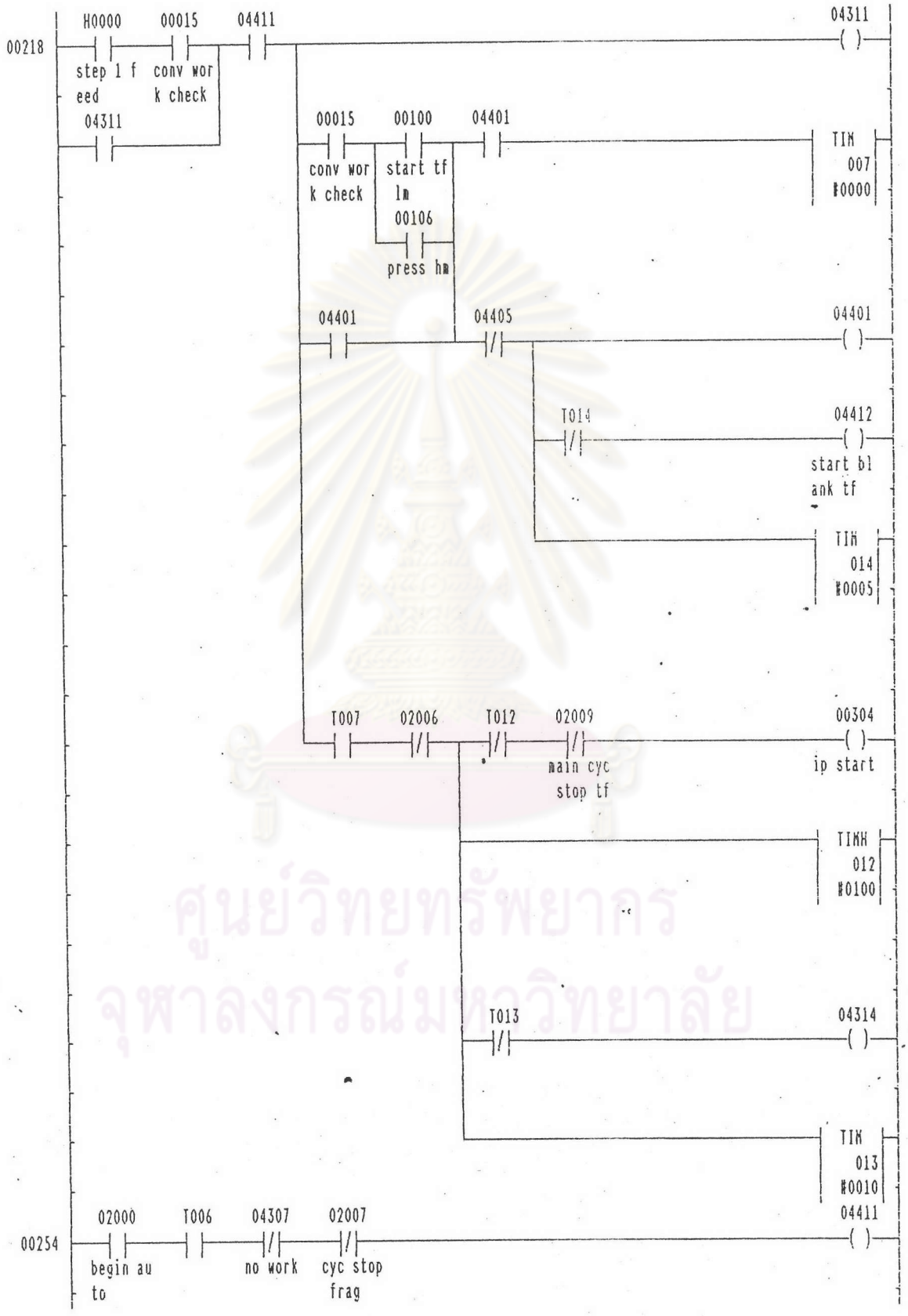


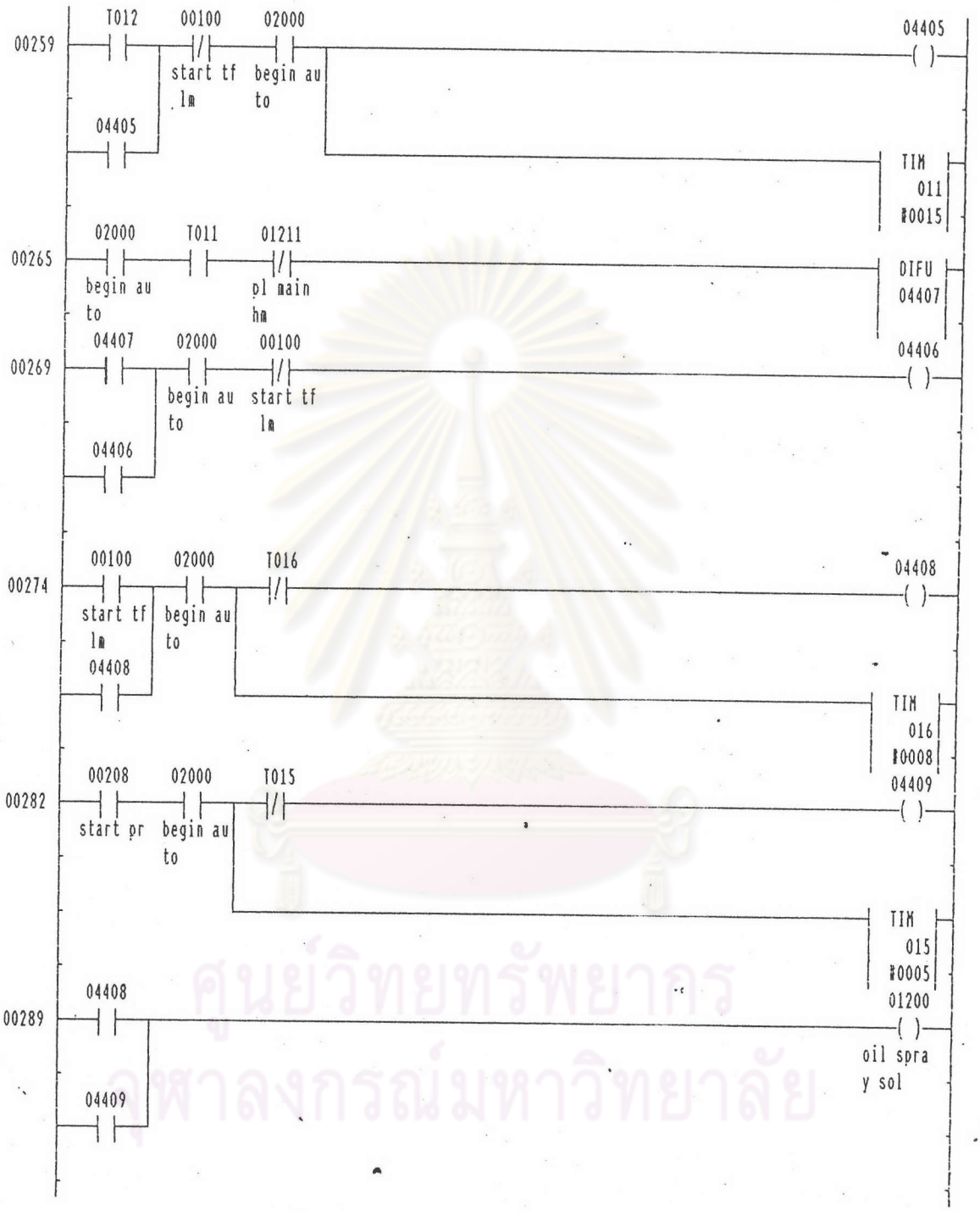


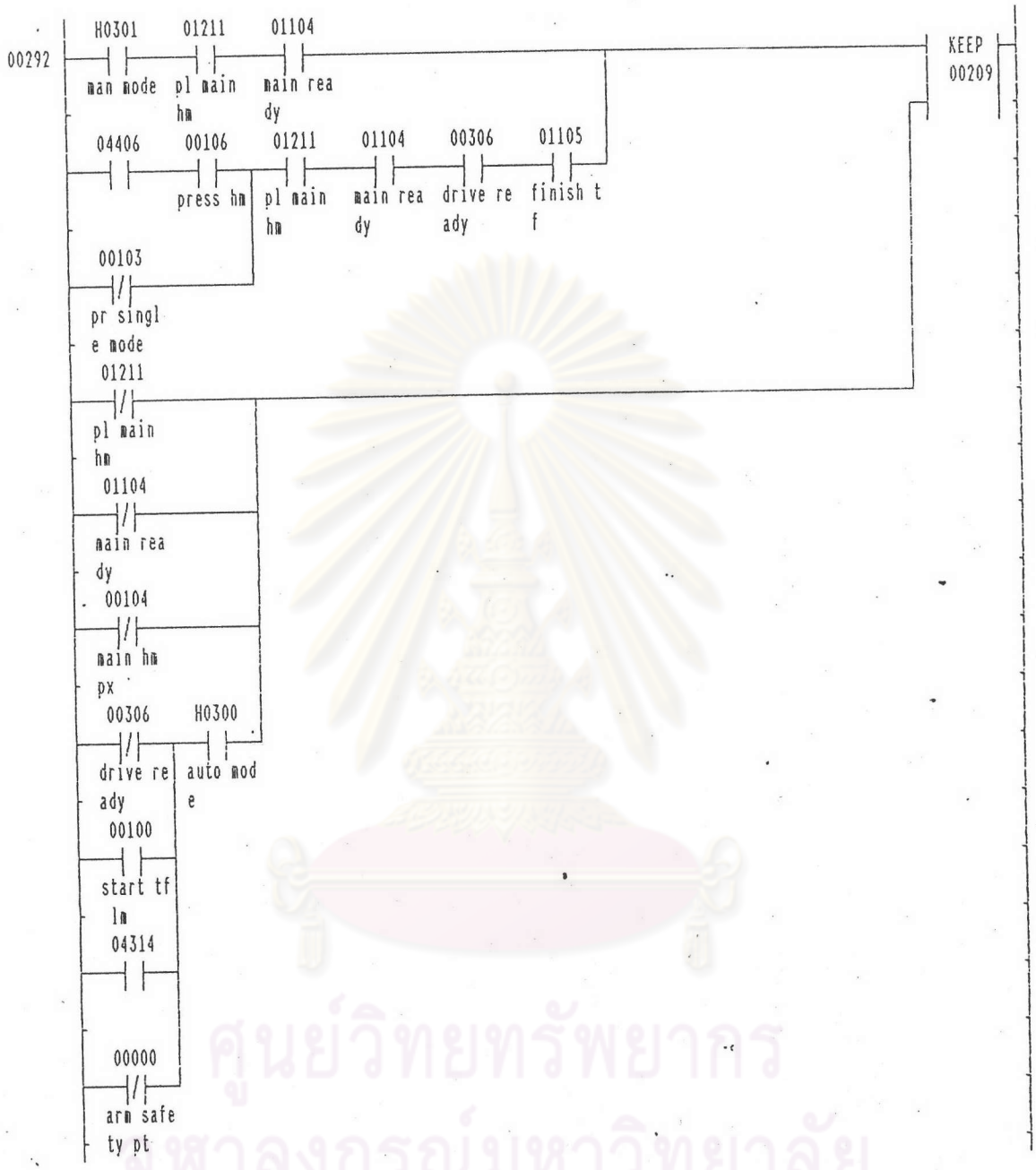

00116
01313
z jog le ft
01314
z jog ri gh
H0301
man mode
KEEP
05002
05003
main hm difu
00121
01107
main hom e post
05002
z jog fr ag
01211
pl main hm
00124
00010
feed in lm
00012
feed up lm
01307
pl feed hm
00127
00007
mag cyl low lm
00009
mag righ t lm
01310
pl mag h m
00130
00106
press hm
01302
pr hm
00132
01300
pl main ready
01211
pl main hm
01302
pr hm
00306
drive re ady
04200
MOV
#0002
H03
00141
00107
pb ver a lt
H0301
man mode
00305
ip enabl e
01009
pb model alt
01000
pb hor a lt
04201
00148
01000
pb hor a lt
H0301
man mode
00305
ip enabl e
01009
pb model alt
00107
pb ver a lt
04202

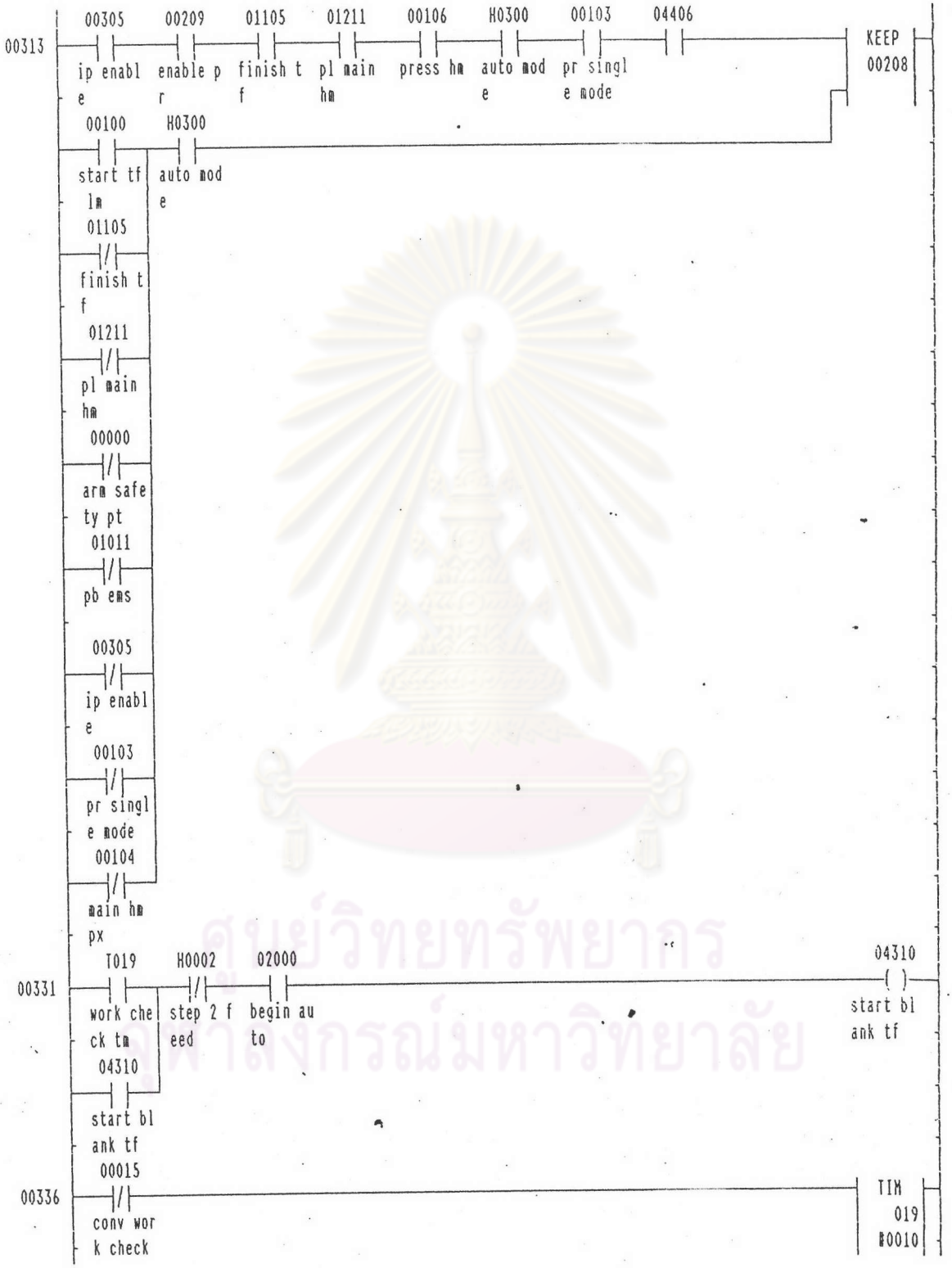


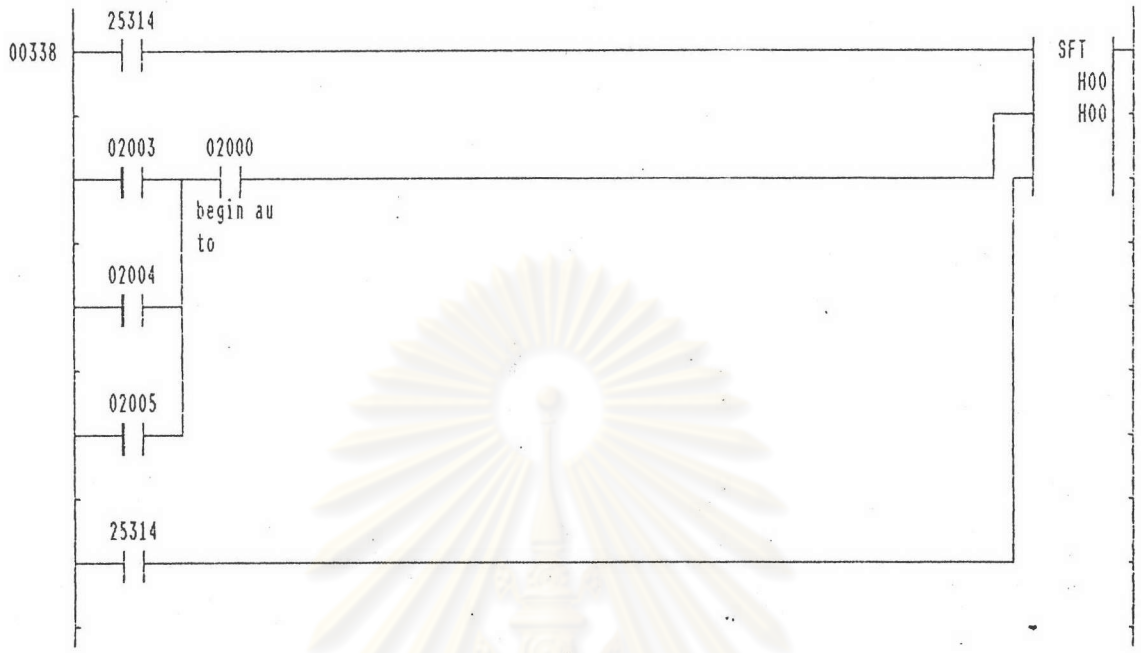


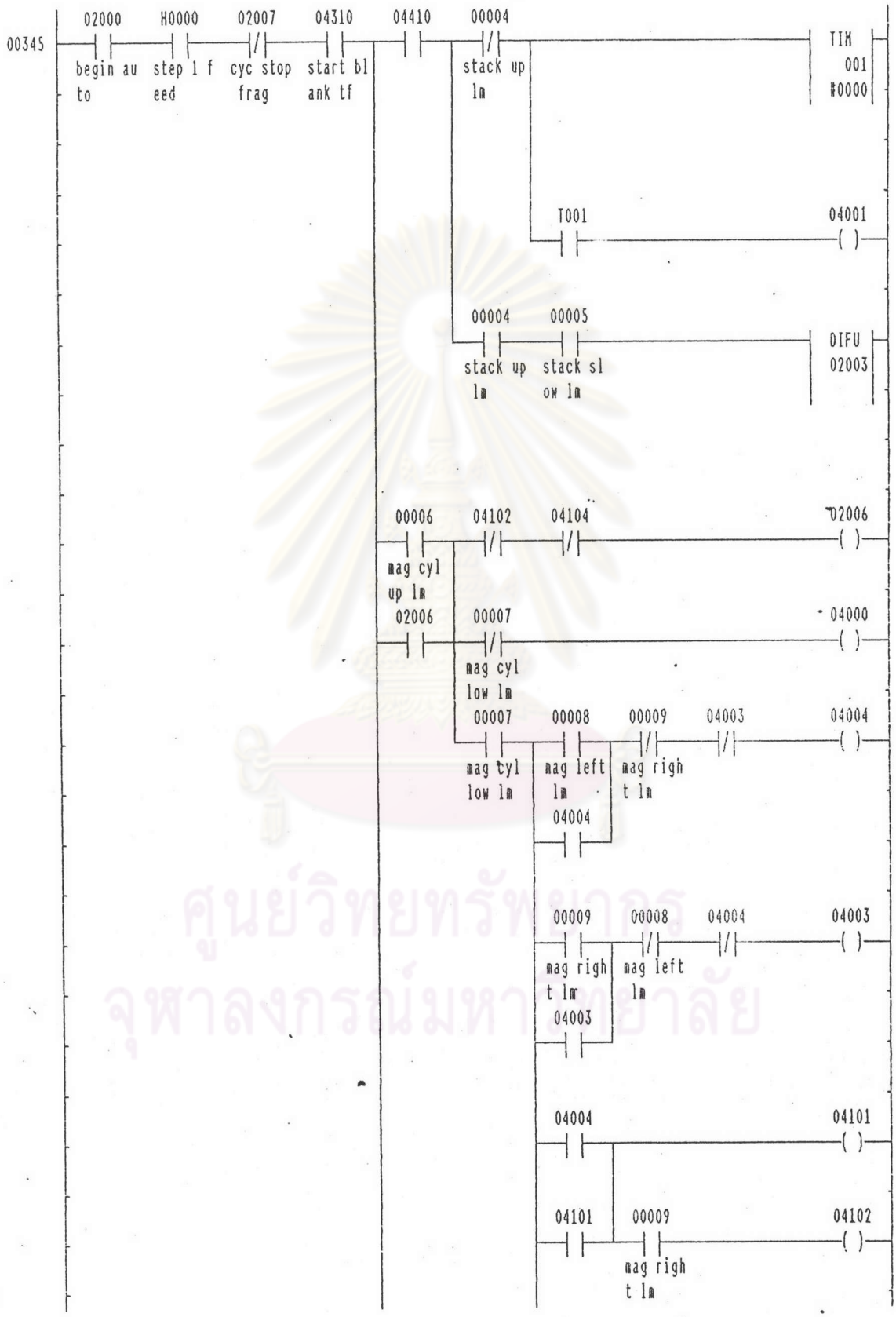


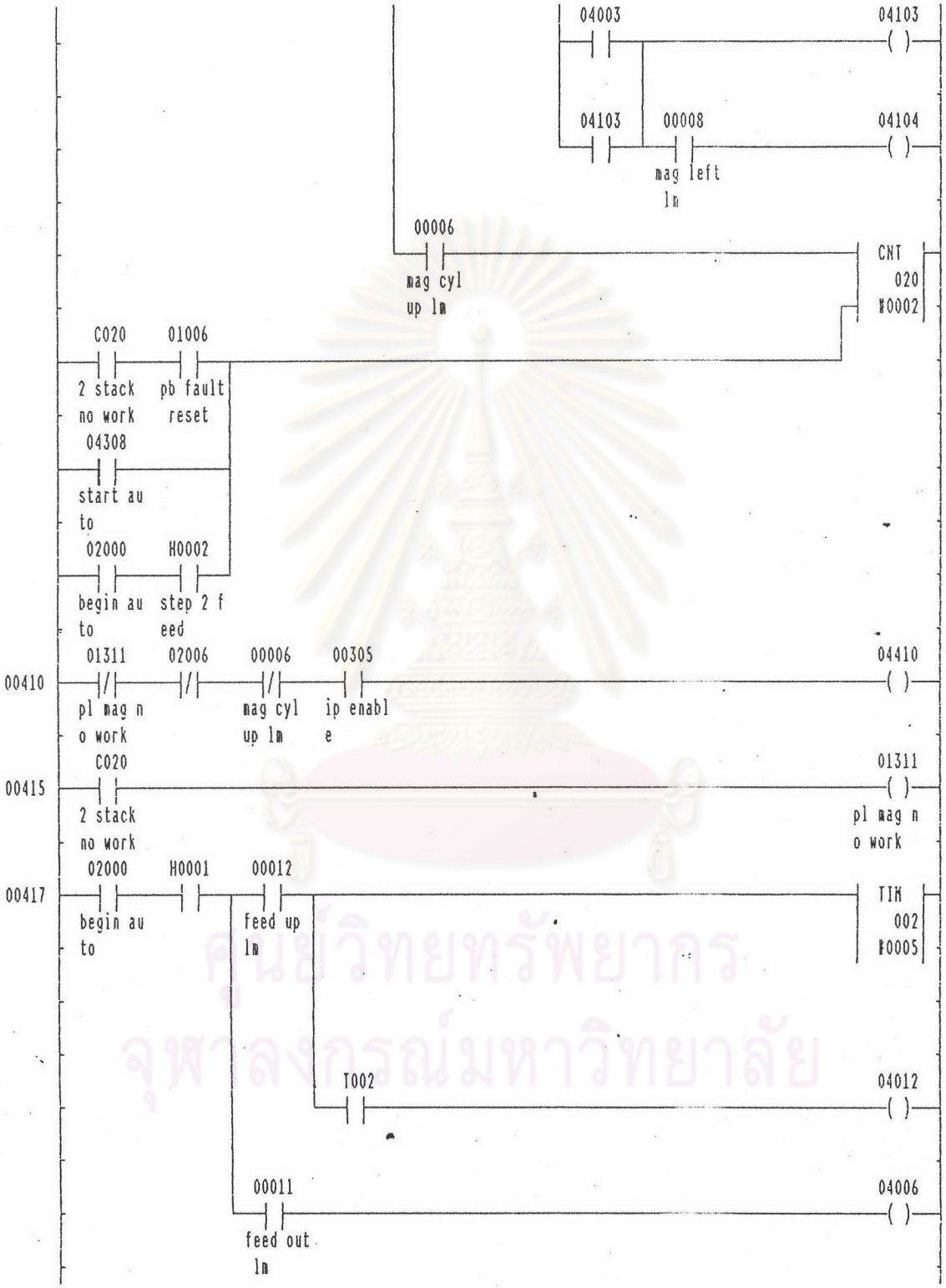


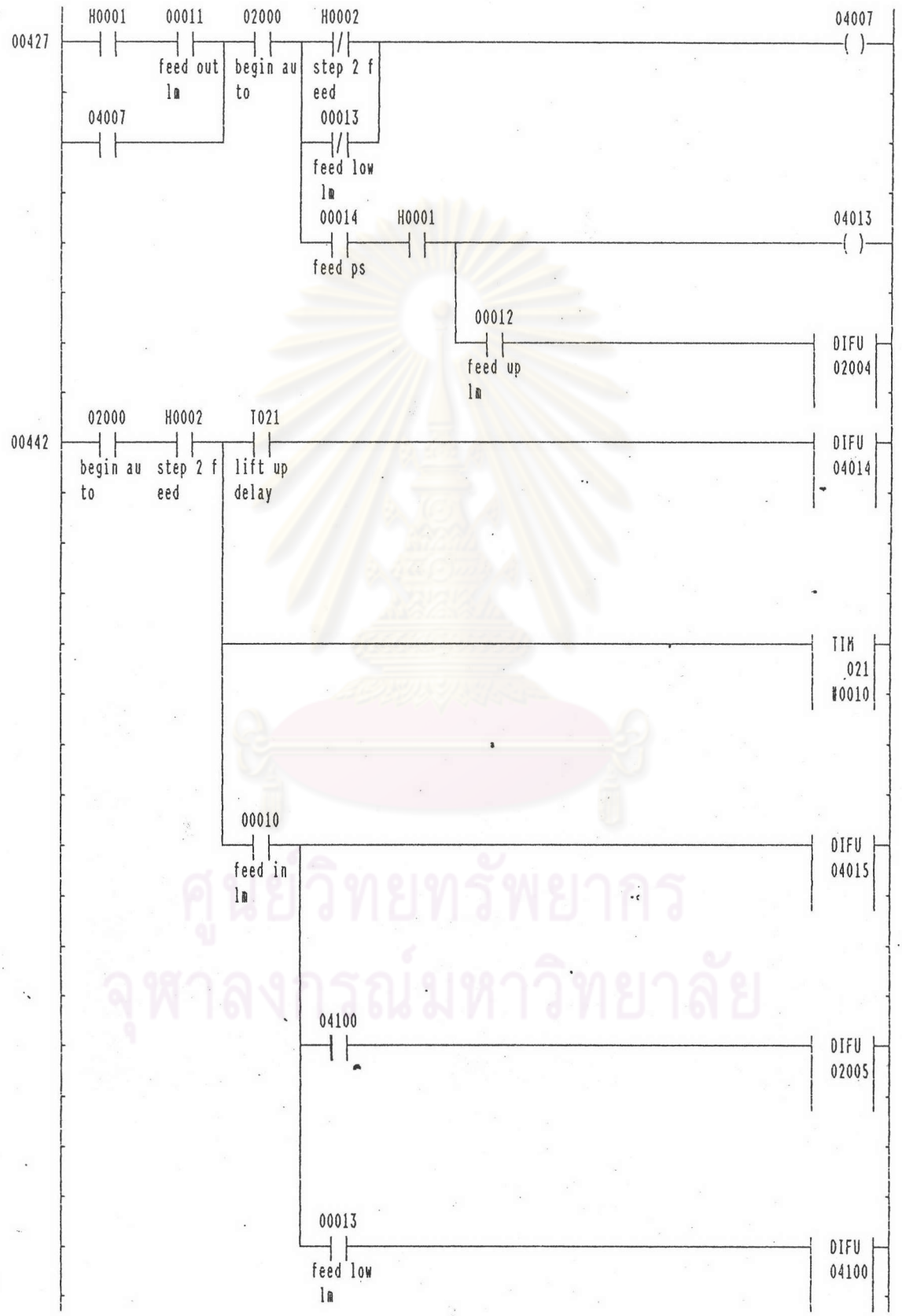


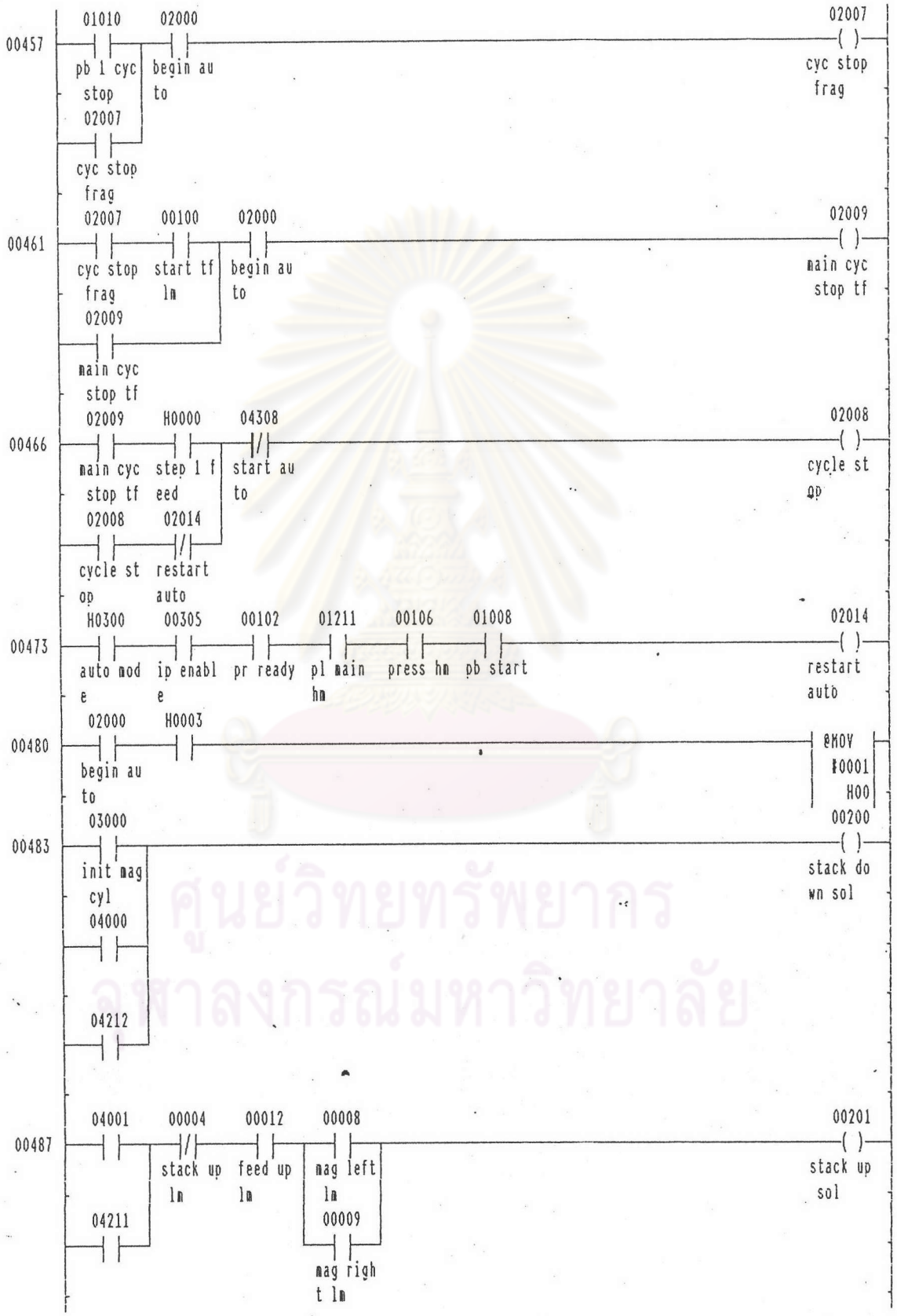


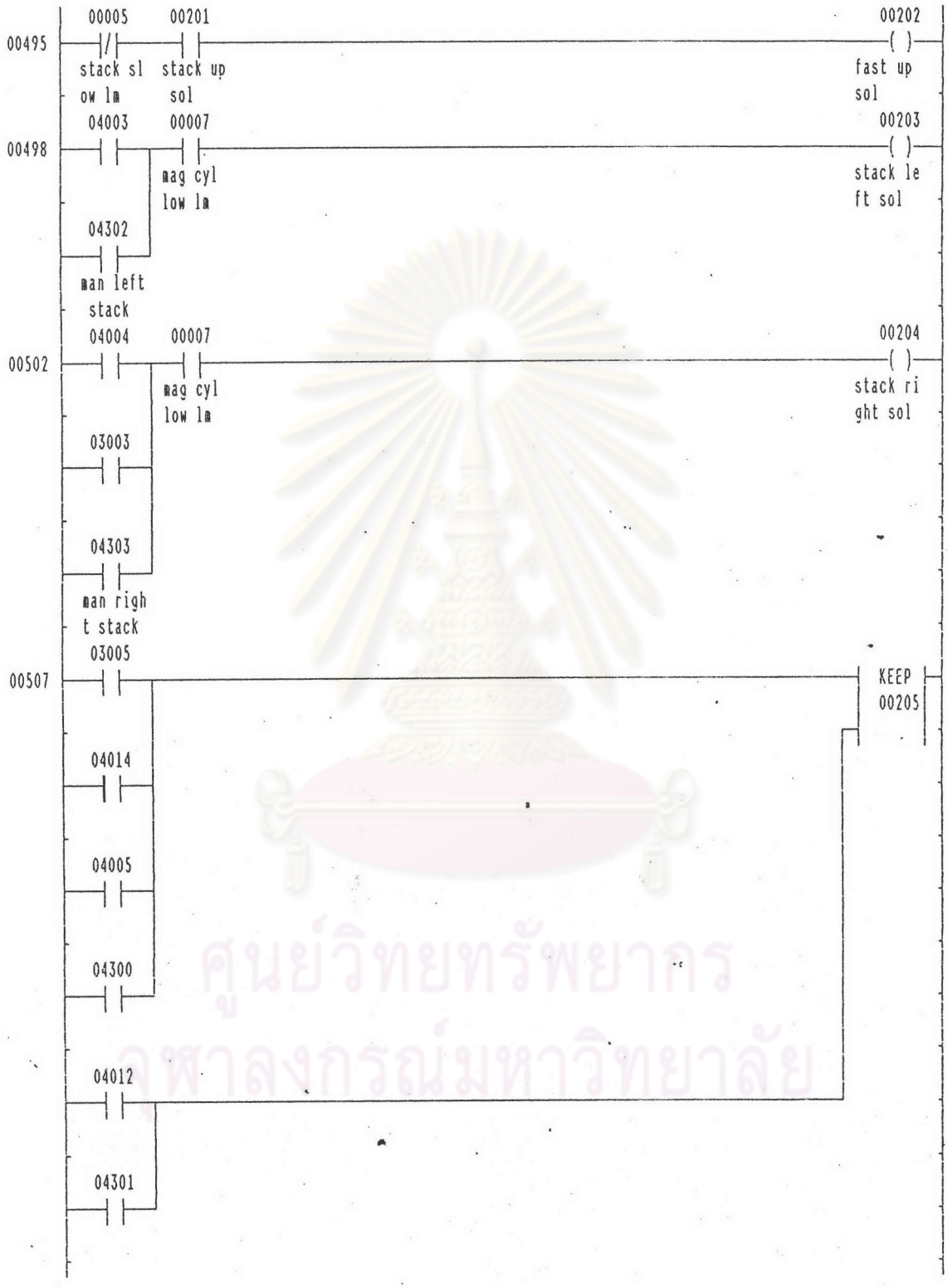


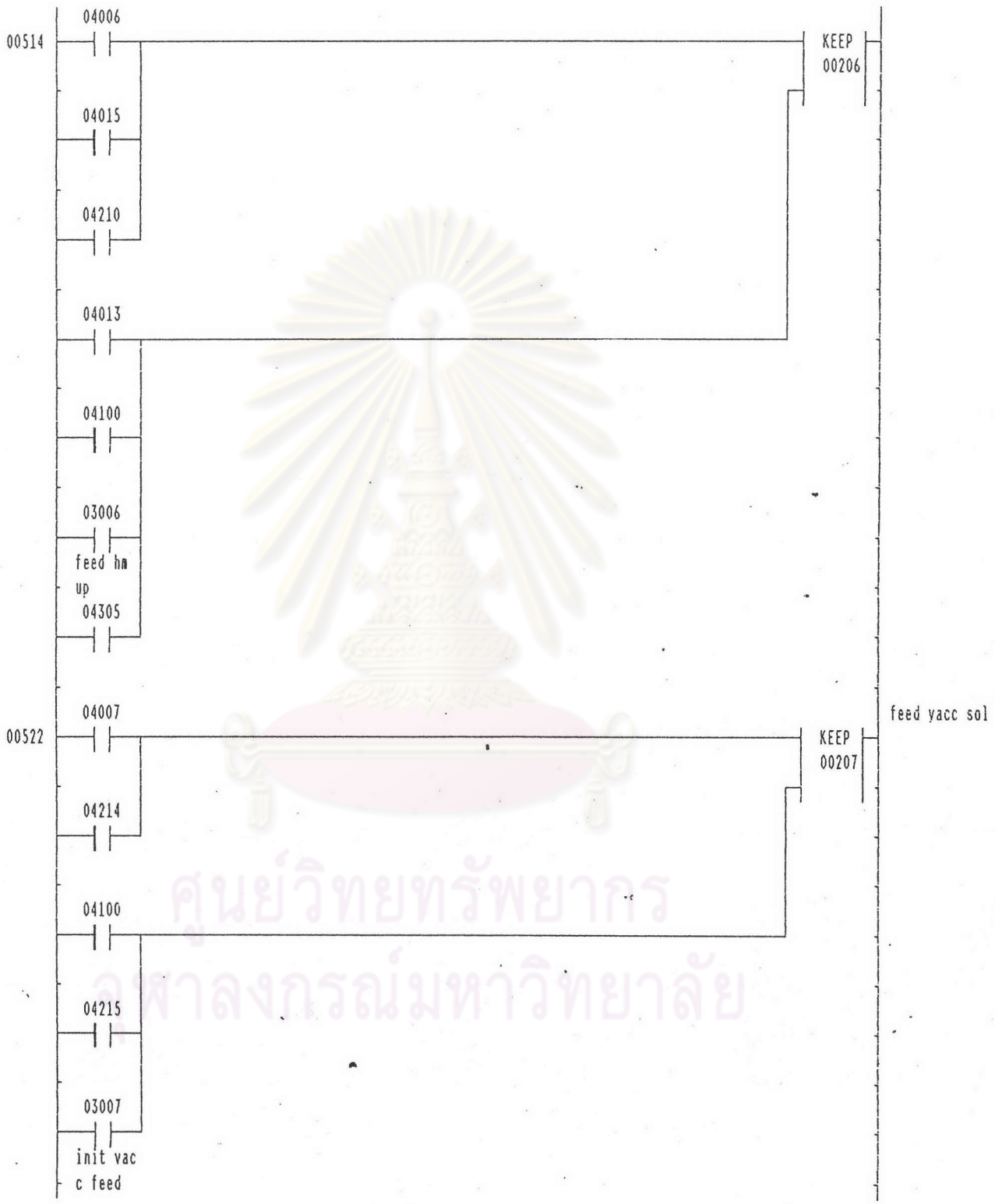


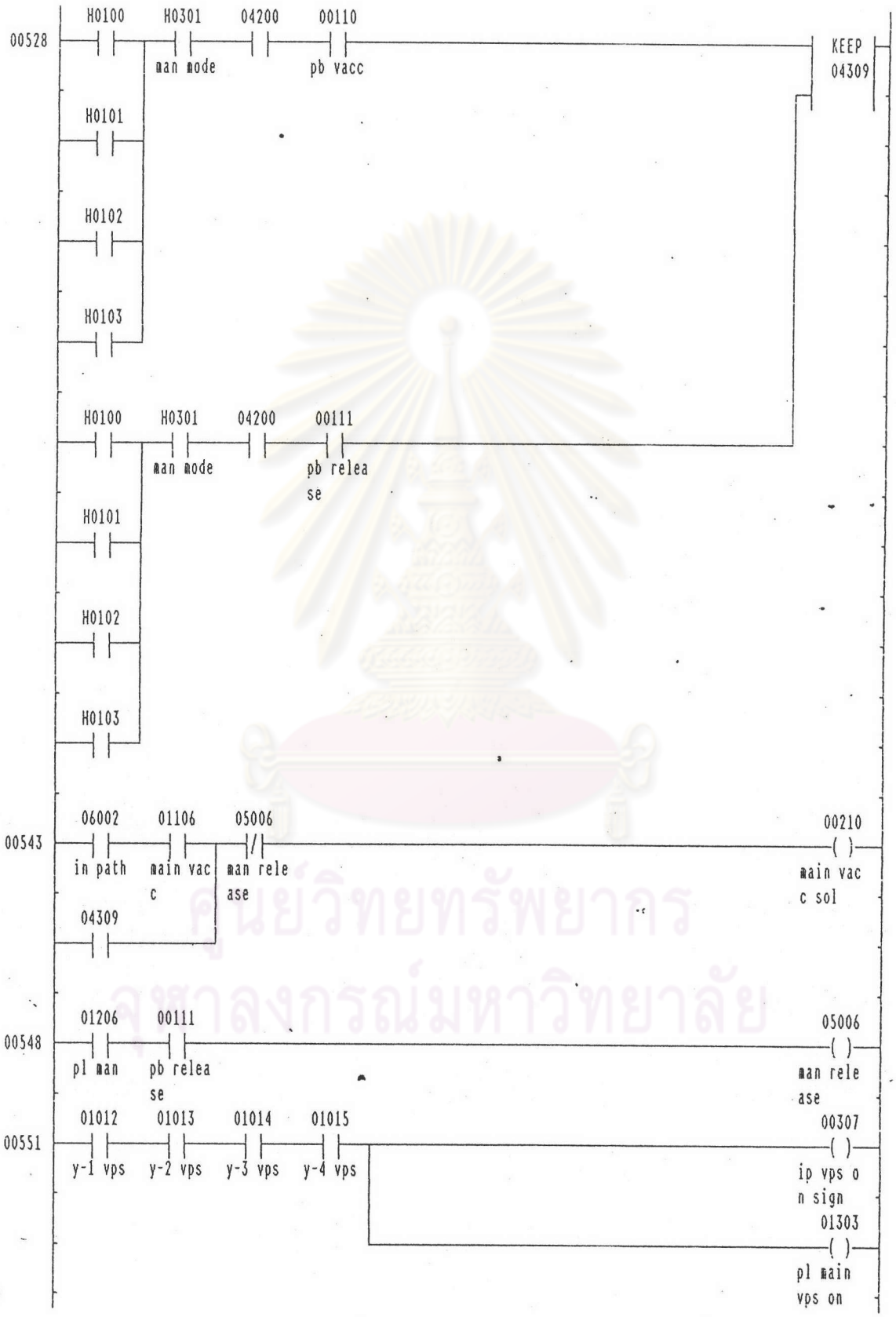


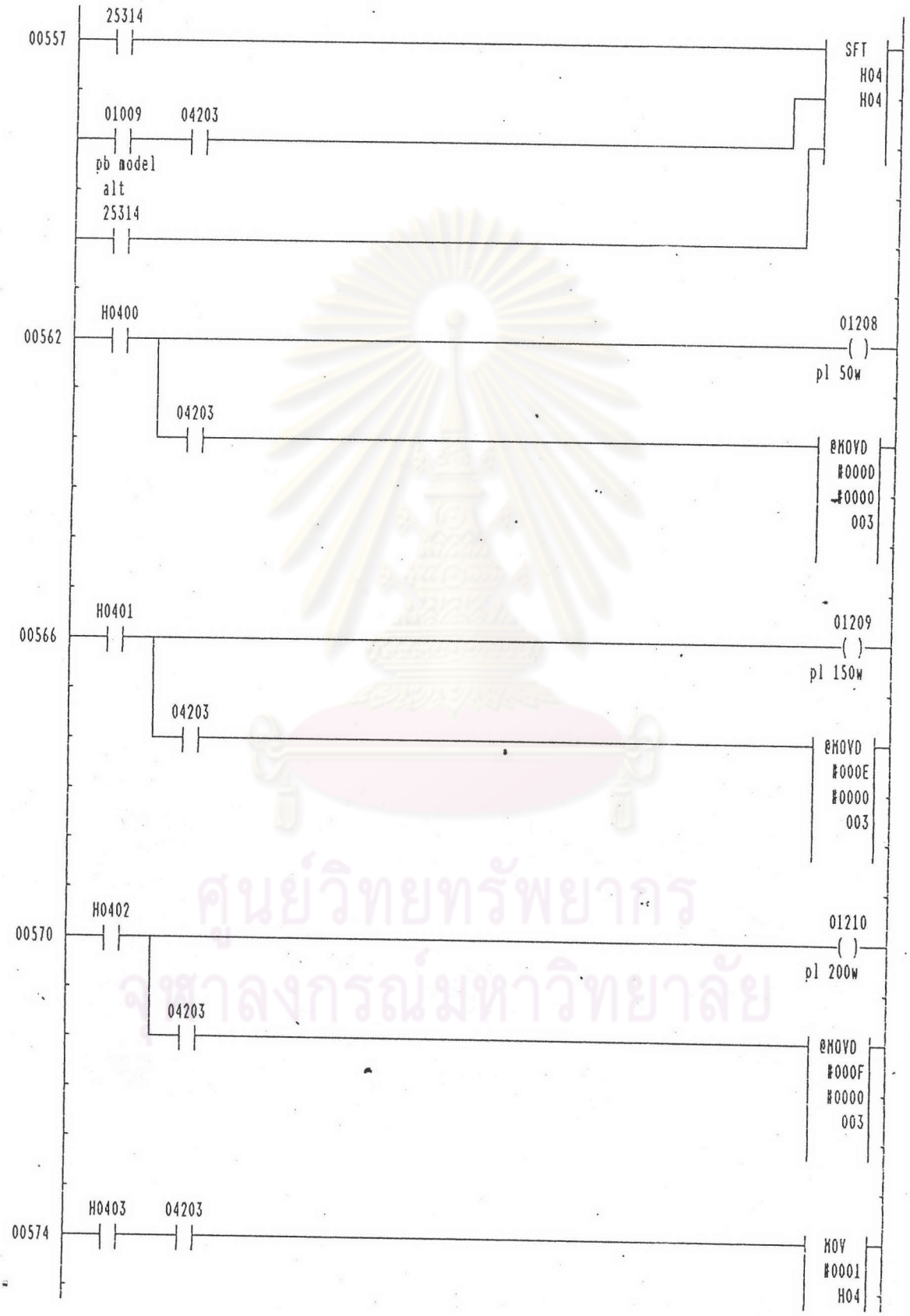


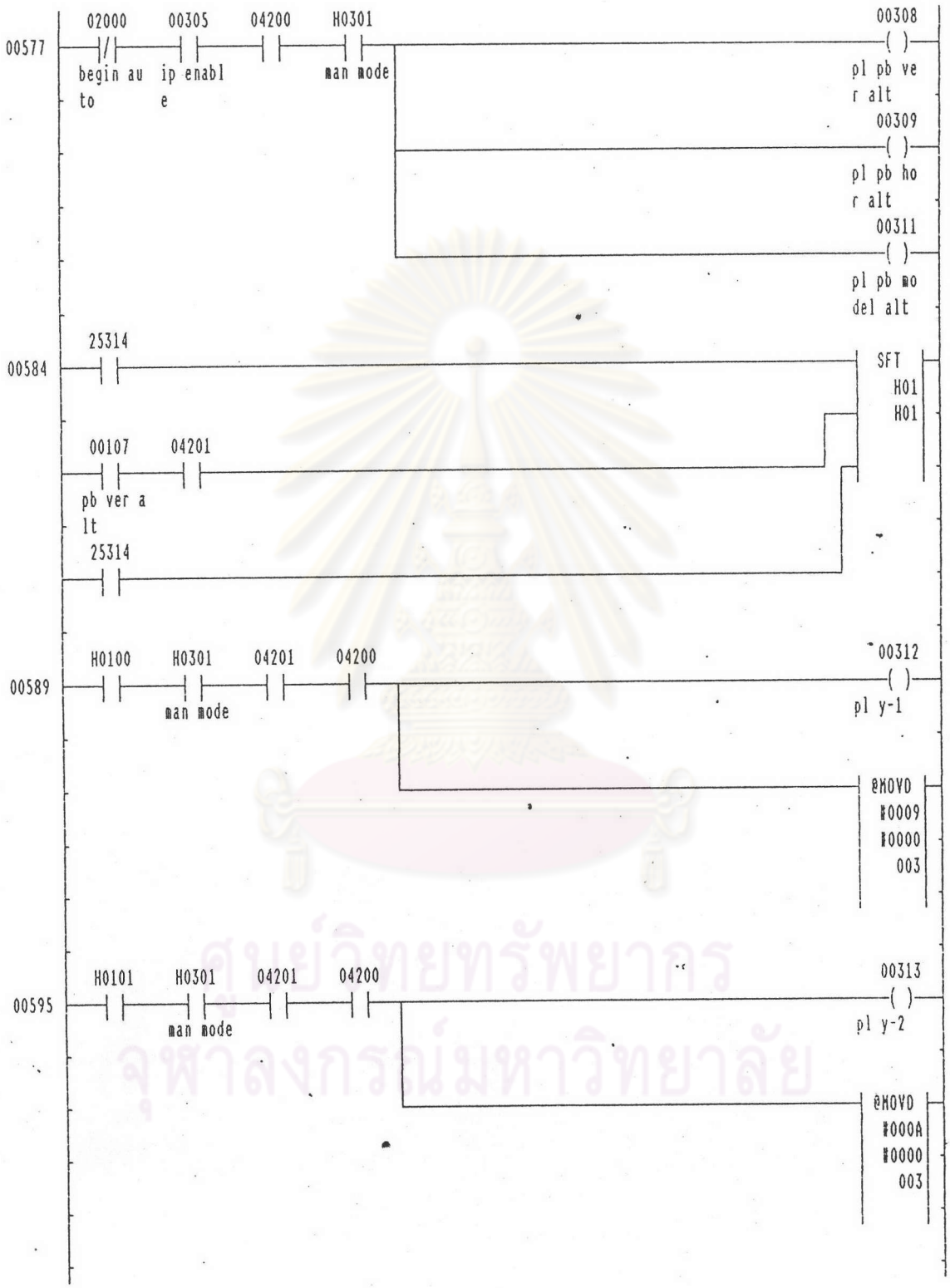


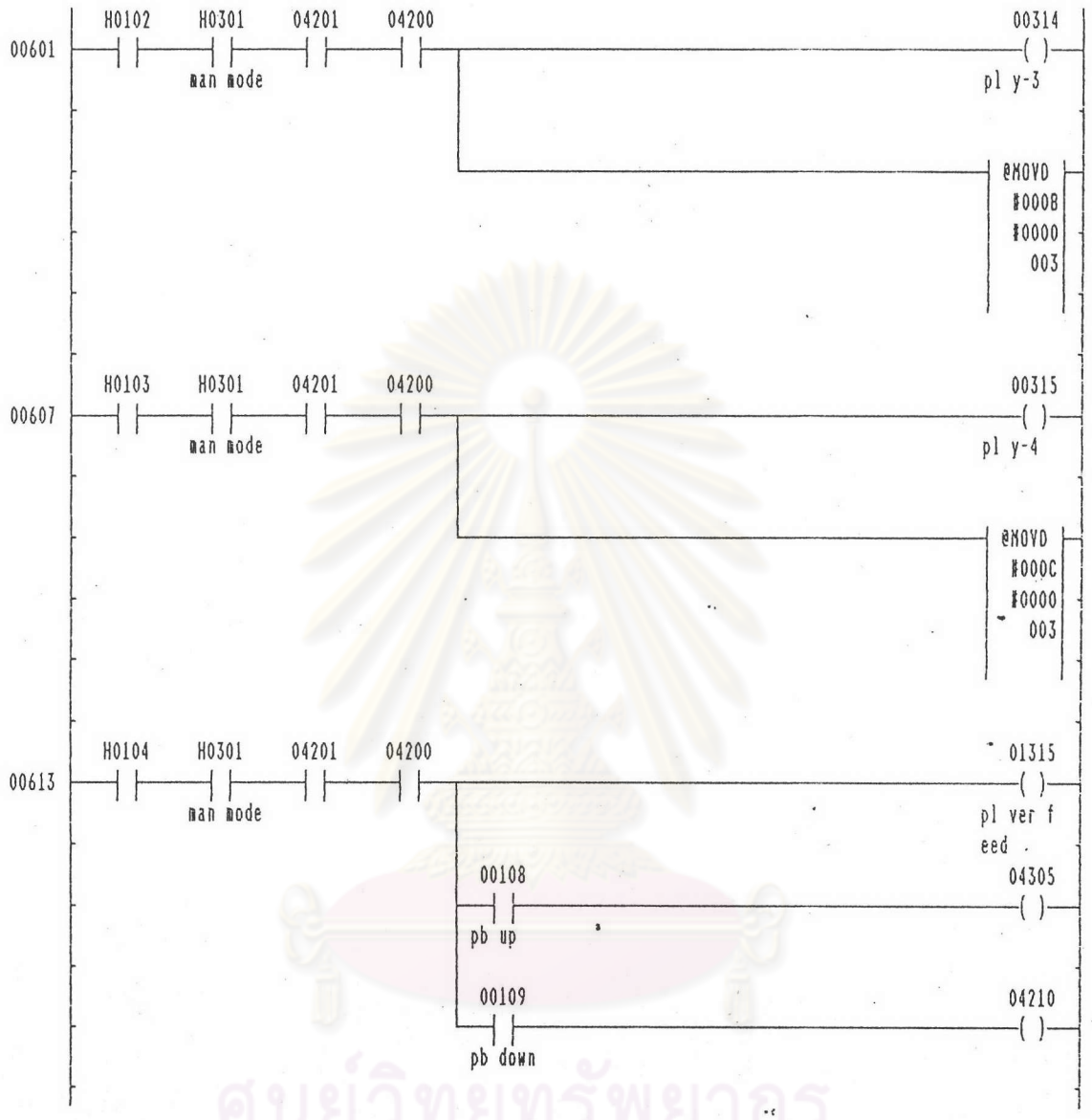


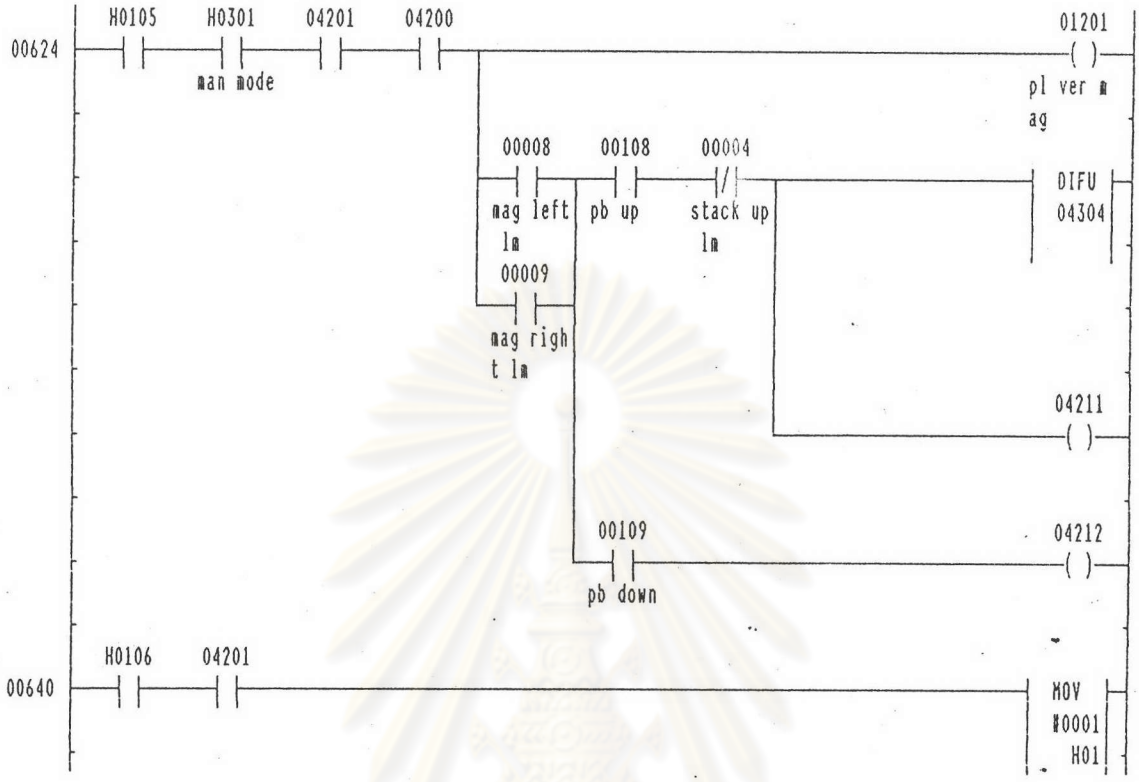


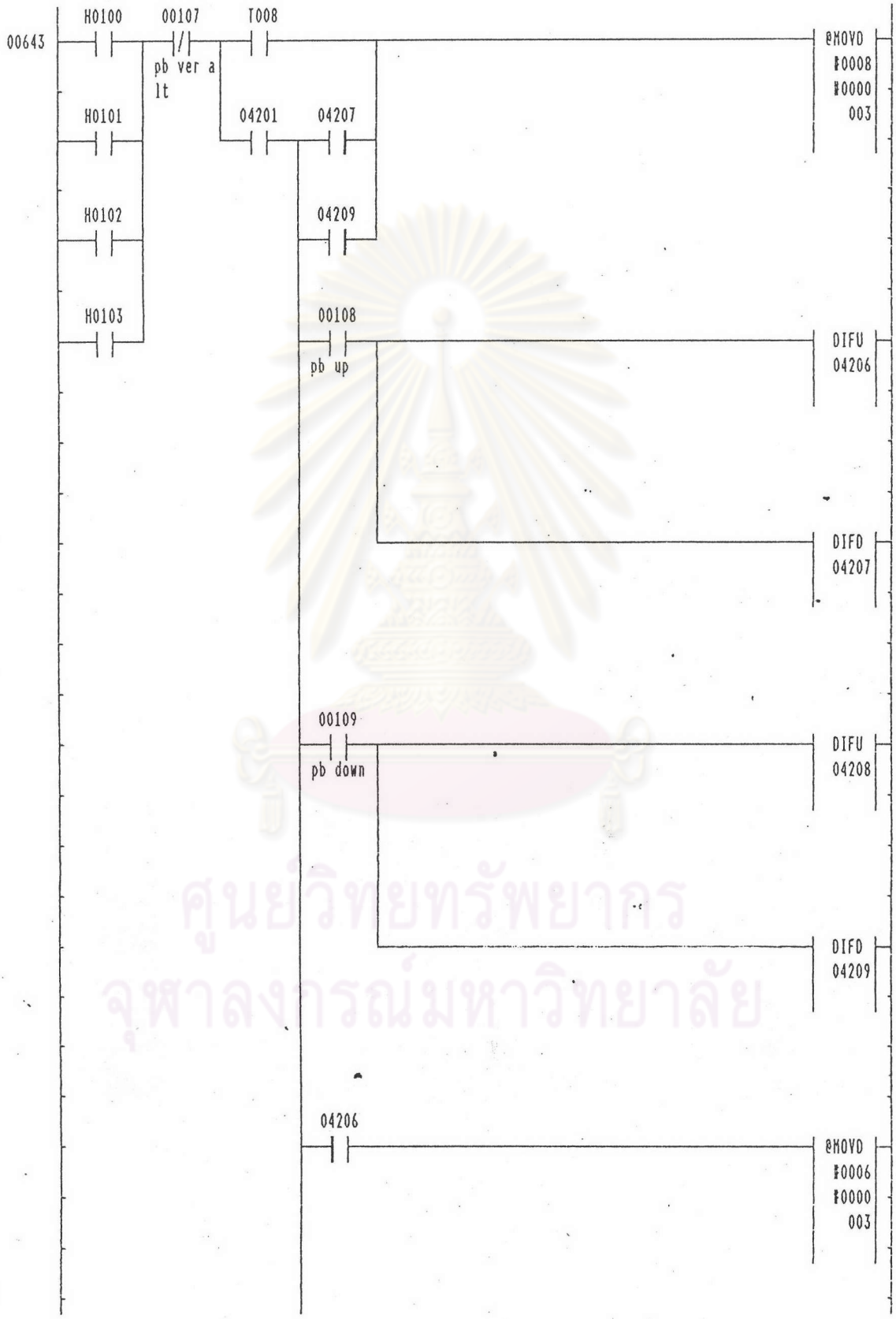


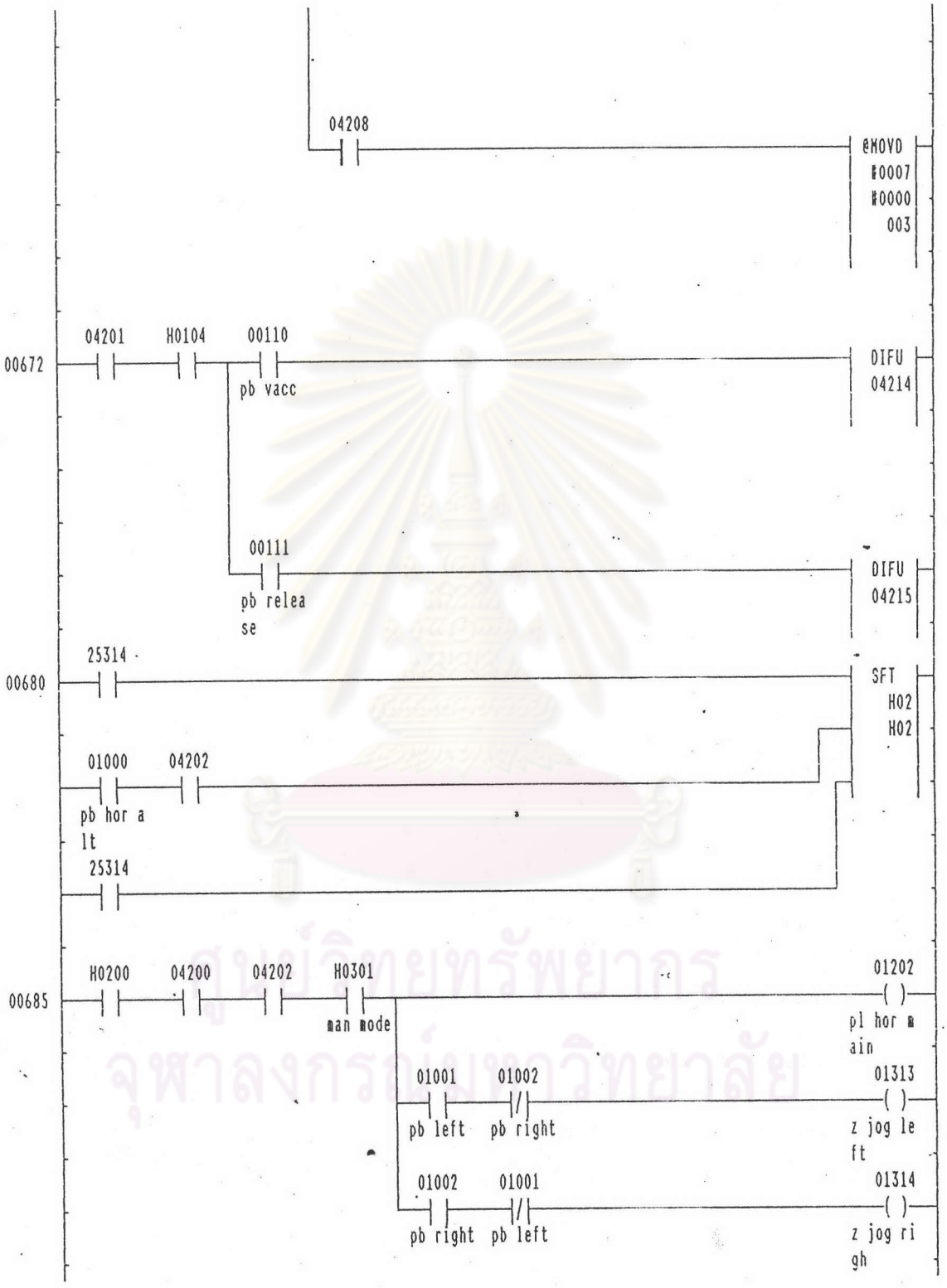


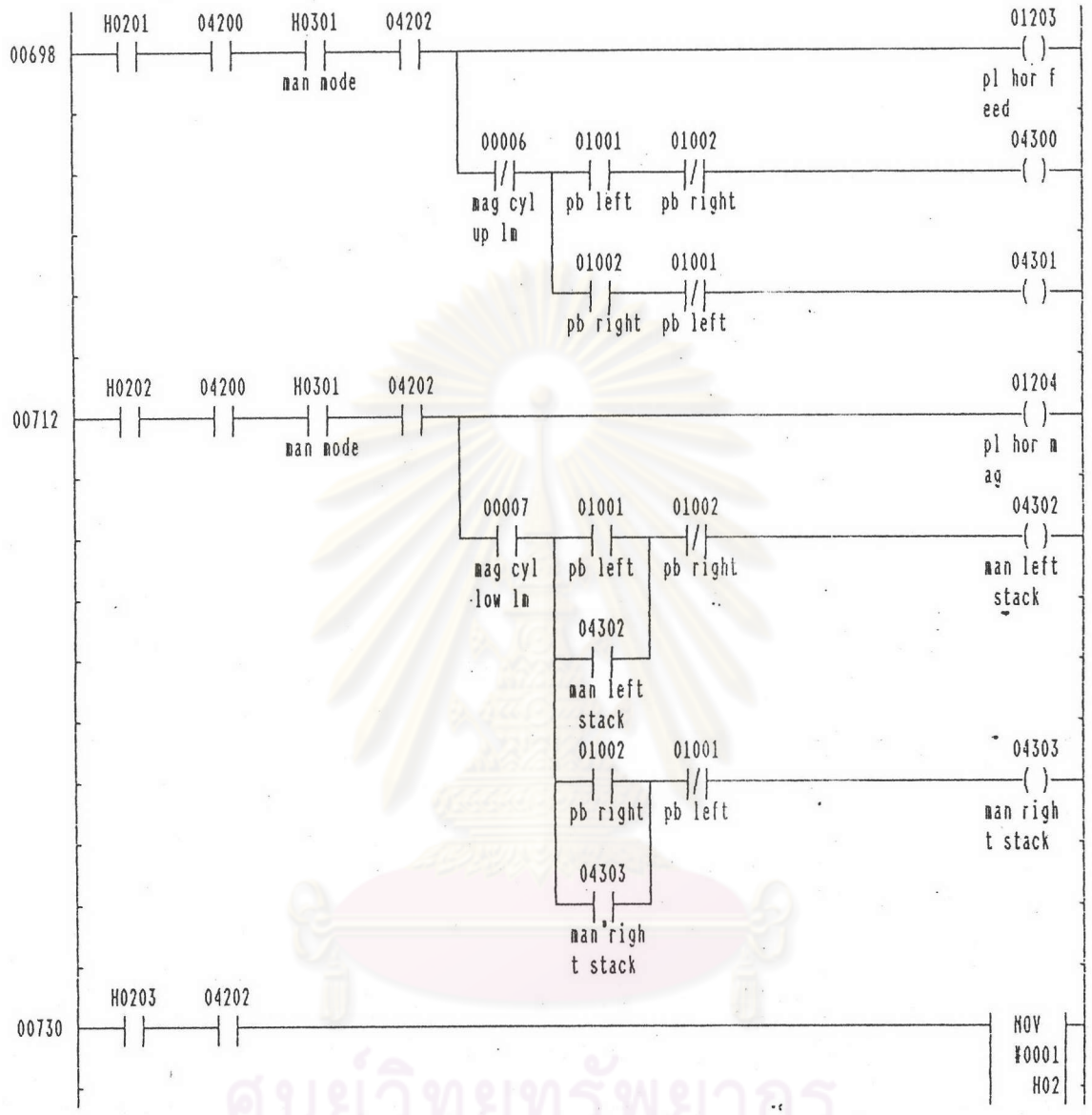


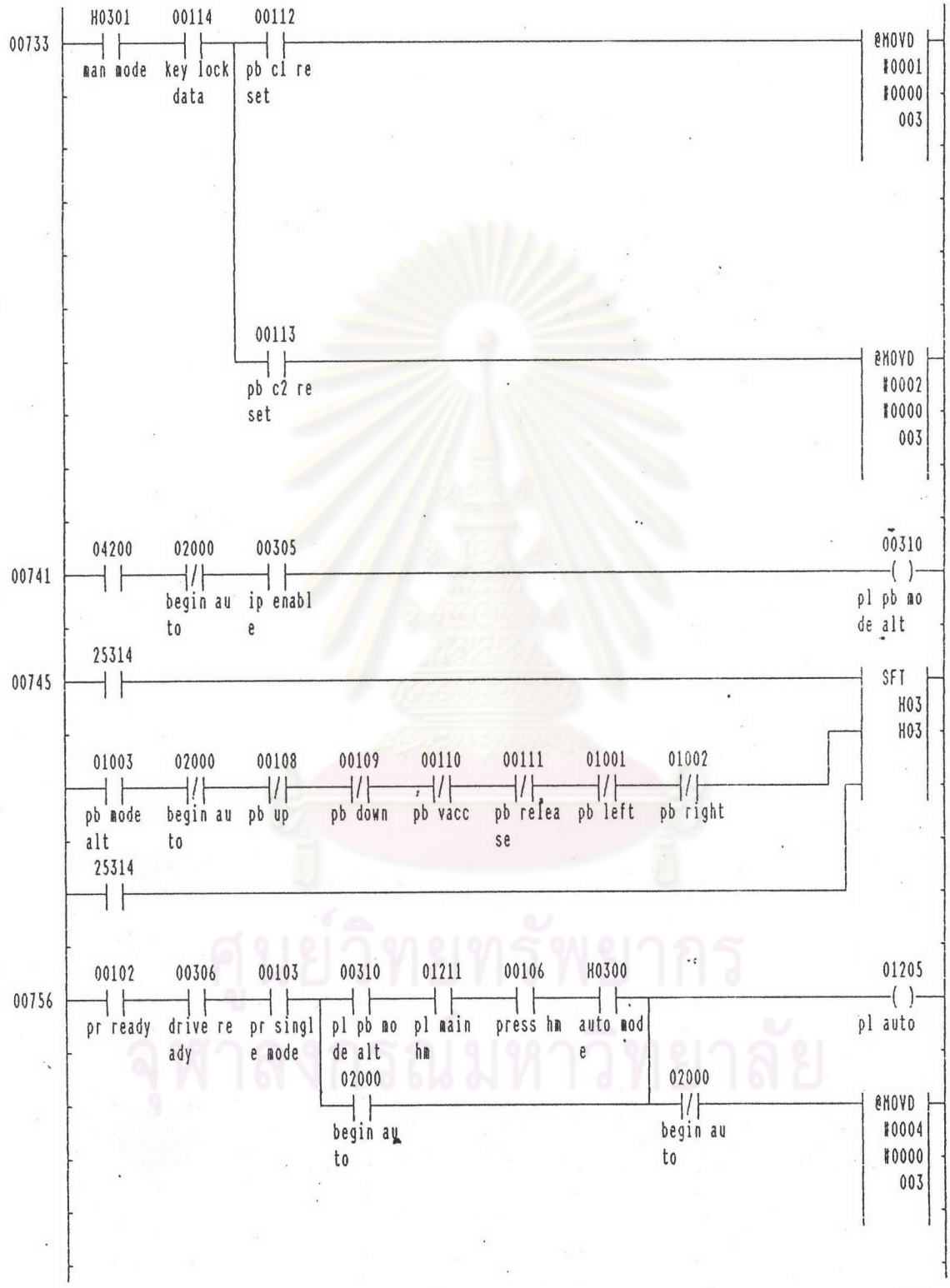


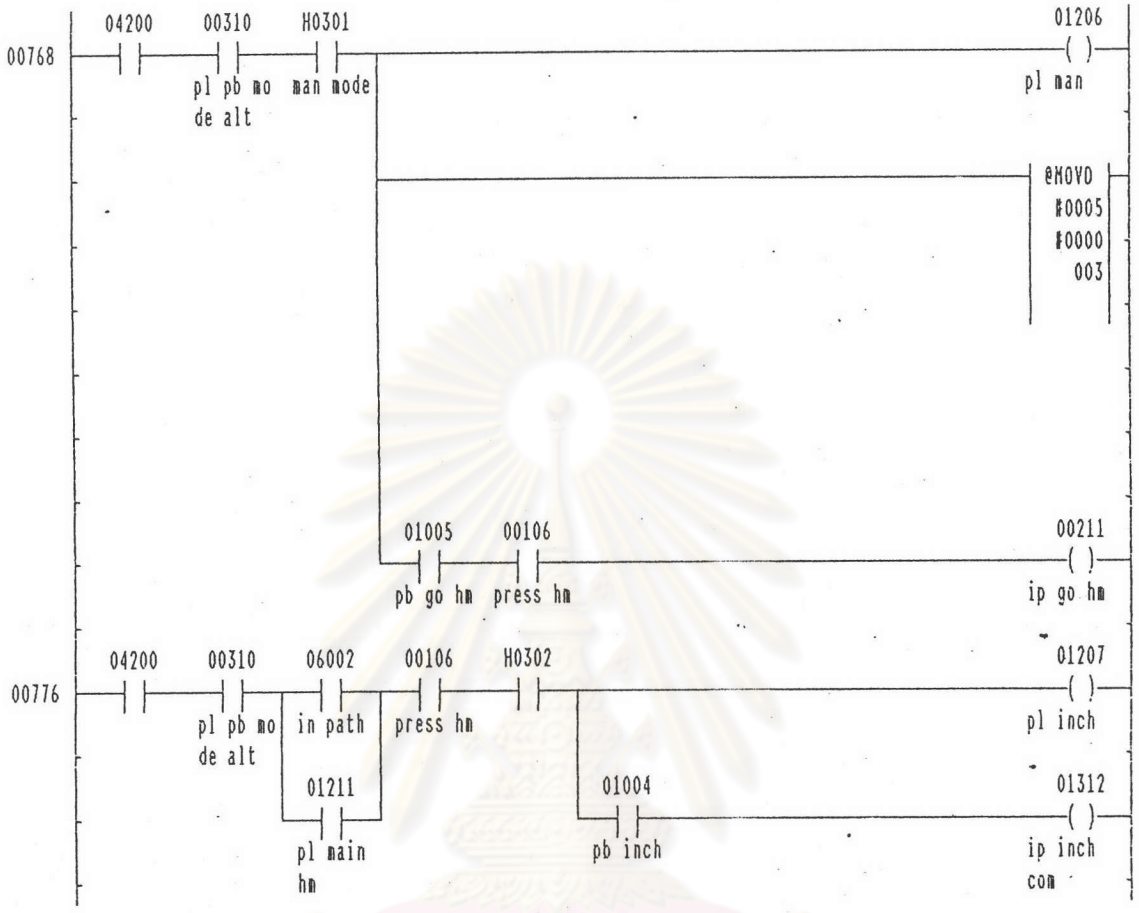


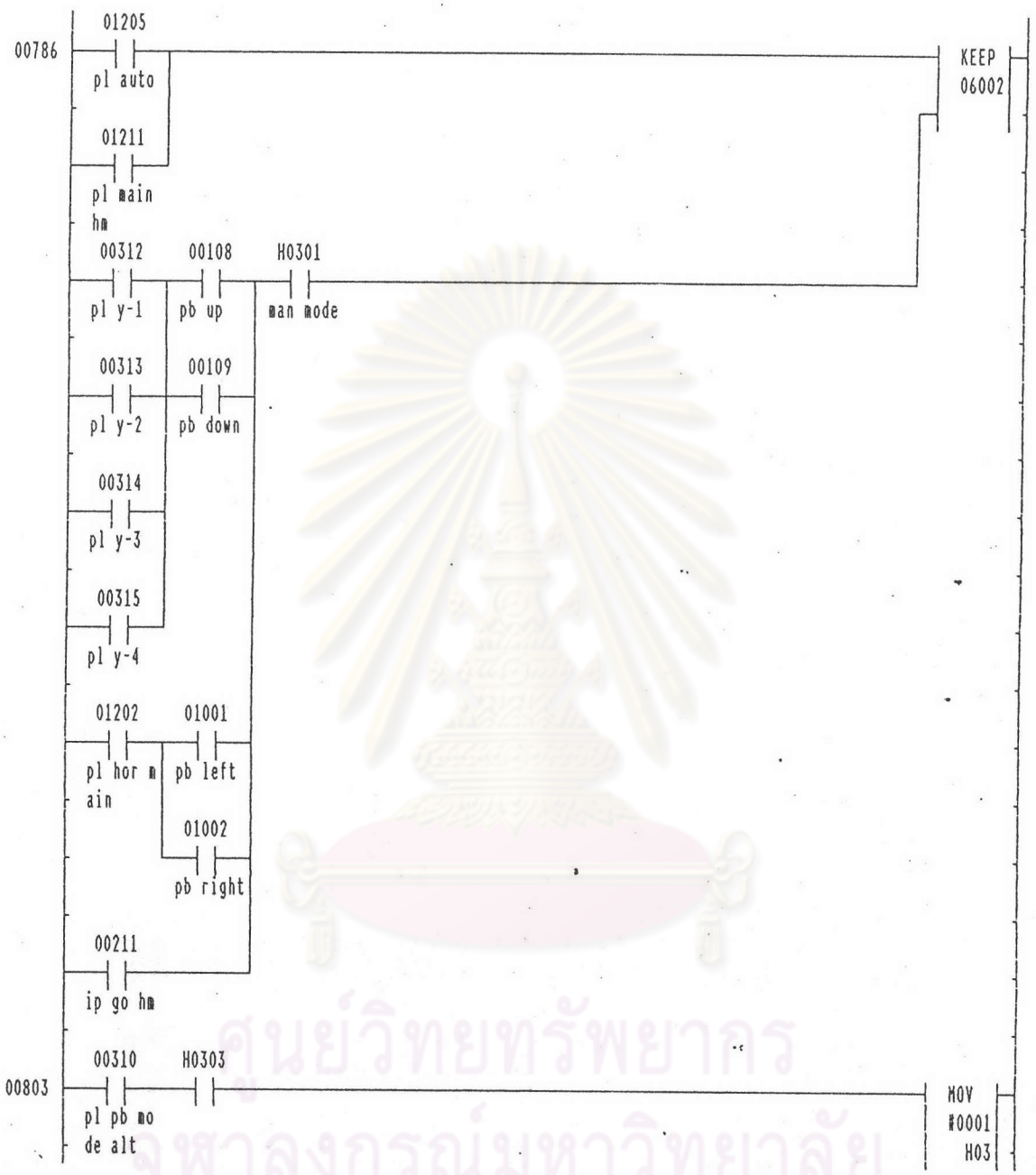


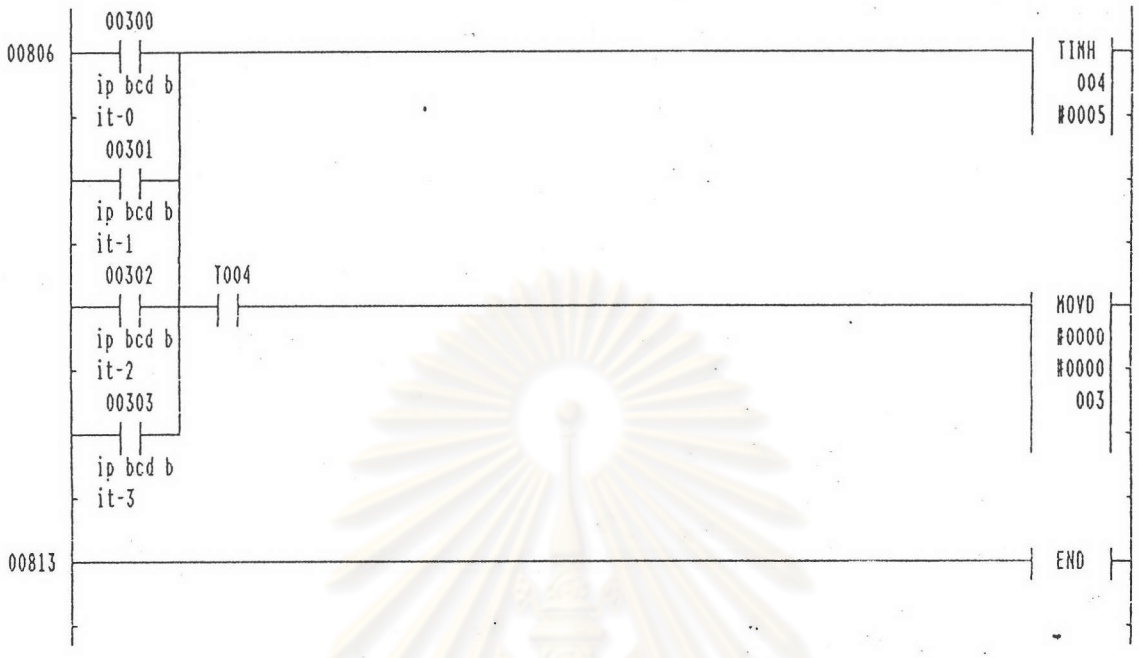





00733
H0301
man mode
00114
key lock data
00112
pb c1 reset
@MOVD
#0001
#0000
003
00113
pb c2 reset
@MOVD
#0002
#0000
003
00741
04200
02000
begin auto
00305
ip enable
00310
( )
p1 pb mode alt
00745
25314
SFT
H03
H03
01003
pb mode alt
02000
begin auto
00108
pb up
00109
pb down
00110
pb vacc
00111
pb release
01001
pb left
01002
pb right
25314
00756
00102
pr ready
00306
drive ready
00103
pr single mode
00310
p1 pb mode alt
01211
p1 main hm
00106
press hm
H0300
auto mode
01205
( )
p1 auto
02000
begin auto
02000
begin auto
@MOVD
#0004
#0000
003



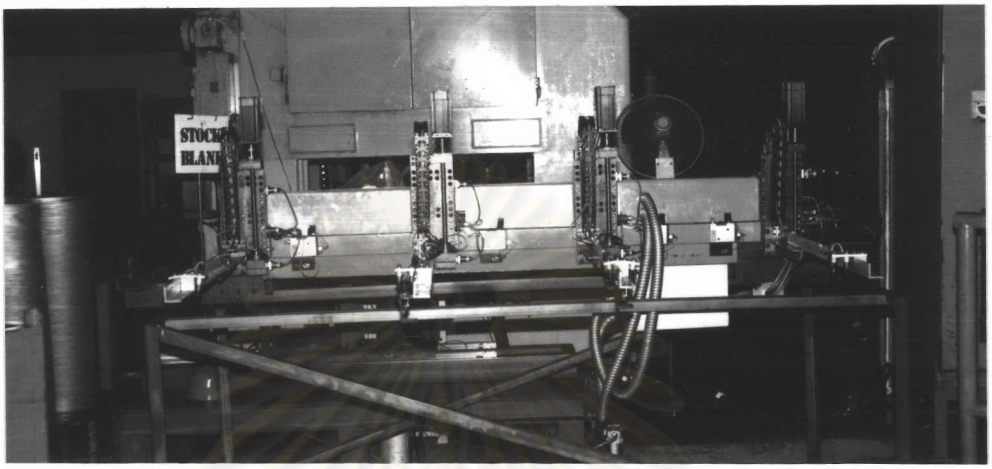


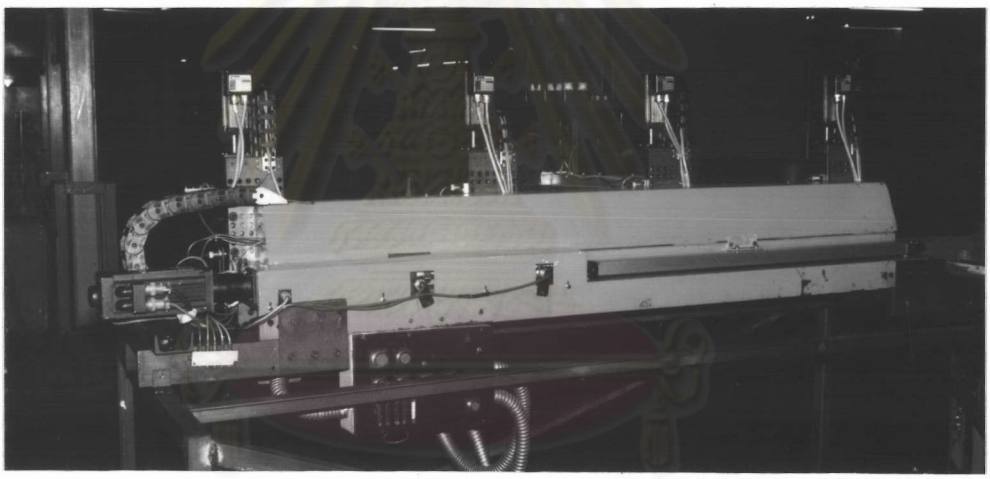


END

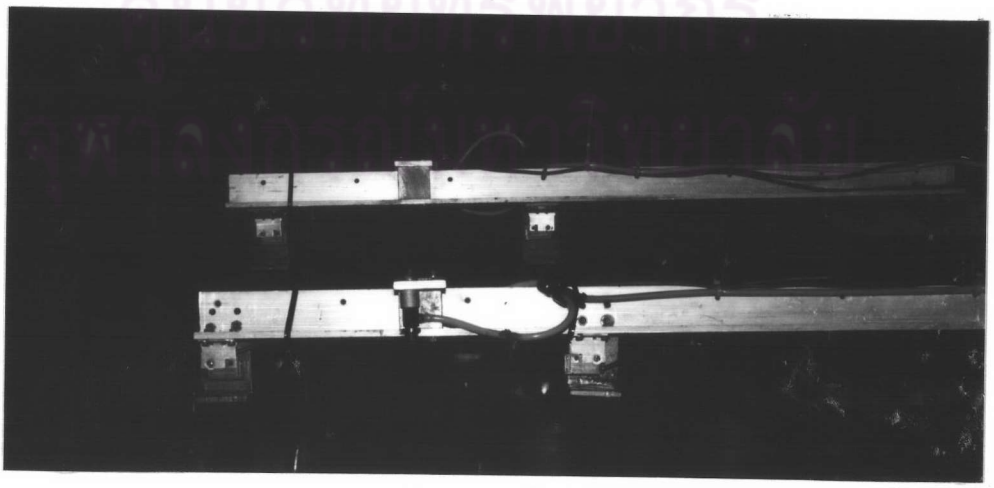
ภาคผนวก ฉ

รูปถ่าย และภาพประกอบเครื่องป้อนชิ้นงานอัตโนมัติ

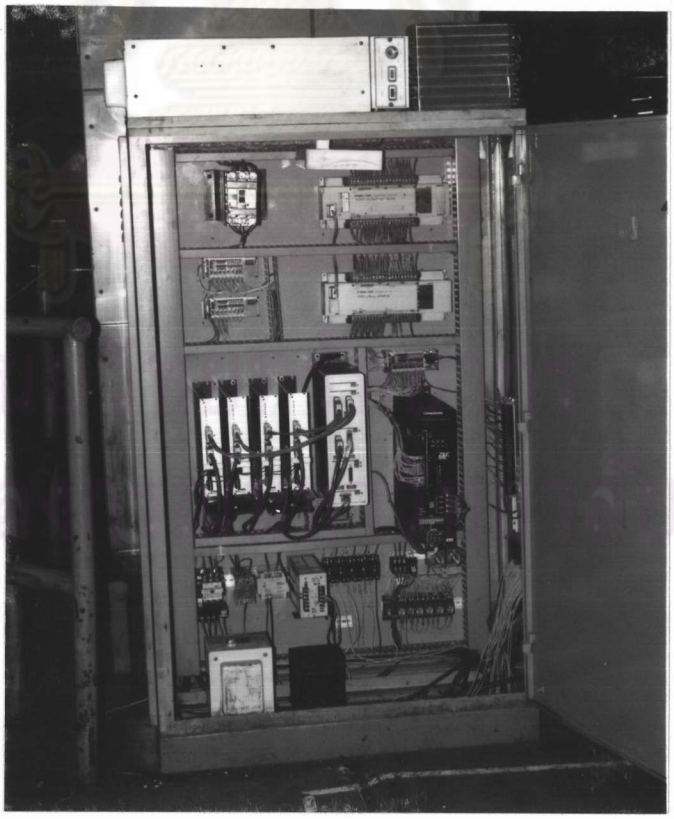




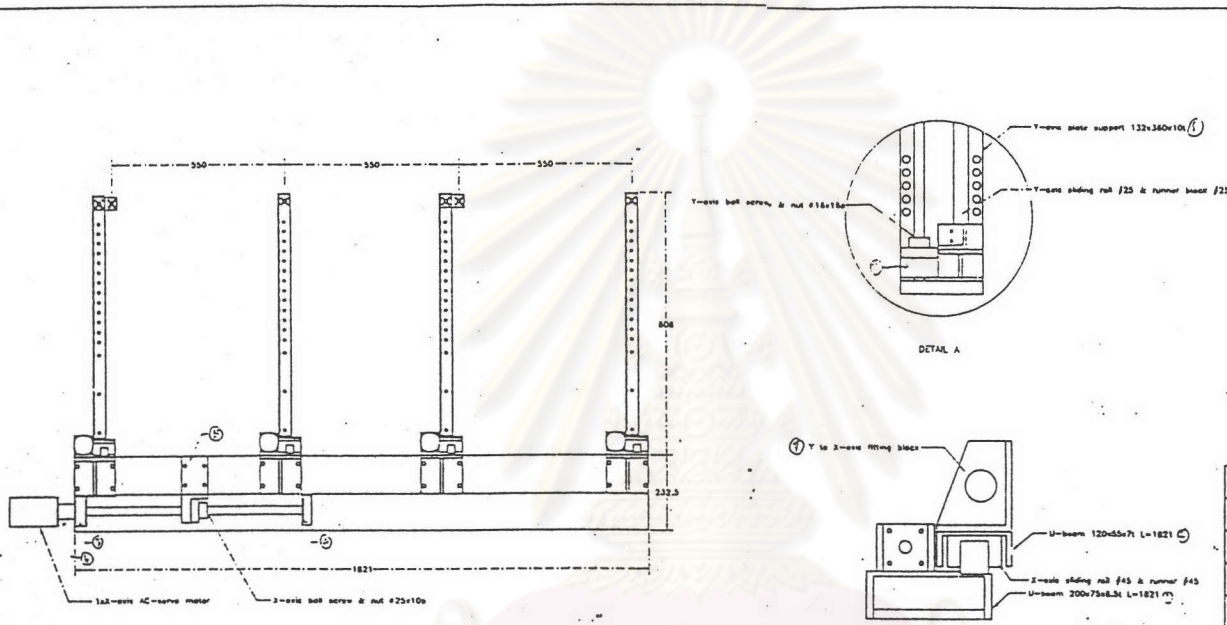
รูปที่ ,2 รูปด้านหน้า และด้านหลังเครื่องป้อนชิ้นงานอัตโนมัติ



รูปที่ 3 แสดงชุดจับชิ้นงาน และสวิทซ์ลำแสงตรวจสอบความสูงชิ้นงาน



รูปที่ 4, 5 รูปแสดงตู้ควบคุม เครื่องป้อนชิ้นงานอัตโนมัติ



No.	ASS'Y DWG.No.	Q'TY
14	ARM	1
13	ARM SUPPORT	1
12	Y- ROUTING HOUSING	1
11	Y- SLIDING BLOCK	4
10	LOWER BLOCK	2
9	UPPER BLOCK	4
8	PLATE SUPPORT	4
7	FITTING	2
6	X- ROUTING	1
5	X- SLIDING BLOCK	2
4	X- SUPPORT 2	1
3	X- SUPPORT 1	1
2	U- BEAM 2	1
1	U- BEAM 1	1

COPY TO	Q'TY

SCALE

DESIGN

DRAWN

CHECK

APPROVAL

MATERIAL SK4 Q'TY

TOLERANCES JIS B 0405 MEDIUM GRADE –

FIRST MADE FOR | MODEL PRESS 350 S. | PART NAME MAIN UNIT ASS'Y | DWG. No. SM 0647-1

DATE | REVISION | CHECKED | R No

SANYO UNIVERSAL ELECTRIC CO.,LTD. DEPARTMENT OF COMPRESSOR FACTORY

ประวัติผู้เขียน

นาย องอาจ วีรชาติยานุกูล เกิดวันที่ 22 มีนาคม พ.ศ. 2513 ที่อำเภอ บัวใหญ่ จังหวัดนครราชสีมา สำเร็จการศึกษา ปริญญาตรีวิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมเครื่องกล ภาควิชา วิศวกรรมเครื่องกล คณะวิศวกรรมศาสตร์ มหาวิทยาลัยขอนแก่น ในปีการศึกษา 2534 แล้วเข้าศึกษาต่อในหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต ที่จุฬาลงกรณ์มหาวิทยาลัยต่อ เมื่อ พ.ศ. 2534 ปัจจุบันทำงานประจำอยู่ที่บริษัท ซันโย ยูนิเวอร์แซล อิเลคทริค จำกัด(มหาชน) เขตประเวศ กรุงเทพมหานคร