switch which is mounted on the slider at the end of cut off stroke. Fault detection is performed by counting the number of steps taken to back off the cut off tool and compare the count with the number of steps it takes to reach the limit switch.

6.4.7 Part separator

During the machine operation, three types of metal pieces fall from the chuck. One is the chip from machining operation and the other two are the desired part and the ejected stub. The chip can be separated out using a grill arrangement. A rail was mounted on the cross slide to catch the desired part when the cut off is complete and channel the part to one bucket. This rail will not catch the stub because when the part is being ejected, the cross slide is at the full out position. In this case, the stub will fall to the catch area for the chips and removed at every cutting tool change.

6.4.8 Stopper

The function of the stopper is to position the raw material in the axis of lathe's rotation. The stopper must operate while the chuck and raw material are rotating at 1040 rpm. The stopper must also have three stopping position. One near the off tool to position material for end cut off for new raw material, one at the position where the chuck will clamp the material for machining operation, and one out of the way when machining begins. Of these, three position, only the position for material clamping must be held to the tolerance of ± 0.15 mm. This position control the length of the finished part. The stopper will be adapted from the lathe's tail stock. The screw of the tail stock will be driven by stepper motor for precise positioning. With the pitch of the tail stock screw being 16.0 mm per revolution and the stepper resolution is 200 steps per revolution, the positioning resolution is 16.0 / 200 = 0.08 mm. This resolution is well within the 0.15 mm required by part length tolerance. The drive for this application is not critical and the existing hybrid type Step-Syn motor type 80 two phase motor number 103-815-2 was tested with satisfactory result. This motor has holding torque of 21.6 kg-cm and running torque of 12 kg-cm for immediate start-stop operation at 60 rpm (200 steps per second). The fault detection method and the control method for the stopper is the same as the cross slide.

6.4.9 Controller

It can be seen later on in this document that in order to meet the cycle time of 40 seconds per piece, several things has to happen at the same time as well as the precise control of the stepper motor step rate and number of steps taken. Further more, fault detection of the machine must also be active at the same time. These requirements cannot be easily met with standard programmable logic controller (PLC) and microprocessor system must be used for the control purpose. The microprocessor selected for this application is the microprocessor designed specifically for control application manufactured by Intel. It is in the MCS 51 family which is particularly suitable for control purpose. The complete microprocessor board is manufactured in Thailand by Sila Research Inc. and this would guarantee spare part availability. The software for the controller is written in assembly language and embedded in the controller board using erasable programmable read only memory (EPROM) and the entire microprocessor board contain no moving part. The controller communicate with use via switches and lamps. A set of part dimension trim control is built in so that tool setting is not required when adjustment is necessary. More detail on the controller will be discussed in another section of this document.

7. STEP MOTOR DRIVE

Step motor used in this project is a hybrid type with four phases. The characteristic of step motor is that it can be moved in small steps (usually 200 steps per revolution) and capable of sudden start and stop operation. The speed of the stepper motor can be controlled by the frequency of phase excitation and the direction of rotation is controlled by the sequence of phase excitation. A step motor can be driven in one of three ways. Single phase excitation provide lowest power consumption and heat generation. Two step drive provide high operating torque. Two-one phase drive provide halve step motion In general, two phase drive is used except for positioning of the turning tool on the cross slide which require two-one phase drive to meet the positioning resolution requirement. Step motors are rated according to it's holding torque, phase current, and phase voltage. Torque-speed curve is supplied in the motors catalog.

The power supply for the stepper motor is usually of a voltage higher than the motor rating and the voltage to the motor is reduced by a set of series resistor. With this arrangement, torque at high step rate is improved due to better resistor - inductor ratio. Common supply voltage recommended by the factory is 24 volts but in practice, 13.8 volt supply is much more common and is selected for this system. The value of series resistor is calculated using the following equation:

$$R_s = R_m(V_s - V_q)/V_m - R_m$$

Where R_s = value of series resistor in ohm
R_m = phase resistance of stepper motor
V_s = supply voltage
V_q = voltage drop across drive transistor
V_m = motor voltage rating

Supply voltage of 13.8 volt is used because of availability. The voltage drop across the drive transistor is the saturation voltage which is normally about 0.8 volt. The power rating of the series resistor can be calculated from the equation:

$$P_s = I_m^2 R_s$$

Where P_s = power rating of series resistor in watt
I_m = current rating of the motor

It is common that the power rating of series resistor is in the order of 20 to 60 watts and the resistors used are commercially available adjustable power resistor with rating of 100 watt. The power dissipated by the resistor is quite large and cooling fan is required.

8 THE CONTROLLER

8.1 Function of Controller

As the name imply, a controller is used to control a process which in this case is the single purpose lathe. The controller monitor external events (called inputs) and base on those events, give commands to external actuators (called outputs). Inputs to the controller is user activated switches or switches that indicate position of drive systems. The outputs of this system compose of step motors, pneumatic actuator, hydraulic actuators, and indicator lamps. The control process will act according to pre programmed steps required by the process. These pre-programmed steps will be called operation chart. The description of the operation chart follows:

8.2 Operation Chart

The operation chart is a chart that shows when each of the system components need to be activated and for how long. For the single purpose lathe, there are 6 actuator involve which are the linear feeder, rotary feeder, chuck jaw, cross slide, turning slide, and stopper. Description of each of these components can be found in the 'Configuration' section. The chart has the time as the horizontal axis and actuator speed as the vertical axis. This chart is created to aid the programming process and do not show the conventional actuator displacement. The chart show 7 separate fields of action to process the raw into finished part. The description of each field follows:

The first field show actions required to feed the part. This involve pulling the linear feeder back, feeding part from the hopper via rotary feeder on to the rail, and pushing the part through the chuck opening. Before the part can be fed through the chuck opening, the stopper must move to the position just outside of the chuck opening and the turning tool move in to cutting position.

The second field is used to cut the face off a new raw material before part production begin. During this process, the turning tool will smooth out the side of the part before cutting it off. This is required because the raw material is cast iron with rough finish and cutting it directly with the cut off tool may cause tool breakage. While the turning tool is operating, the stopper begin to move back to prepare for next operation. When the turning process is finished, the cut off tool will move in to cut the face of the new material. After that, the cut off tool will move back to allow the material to be fed for further machining.

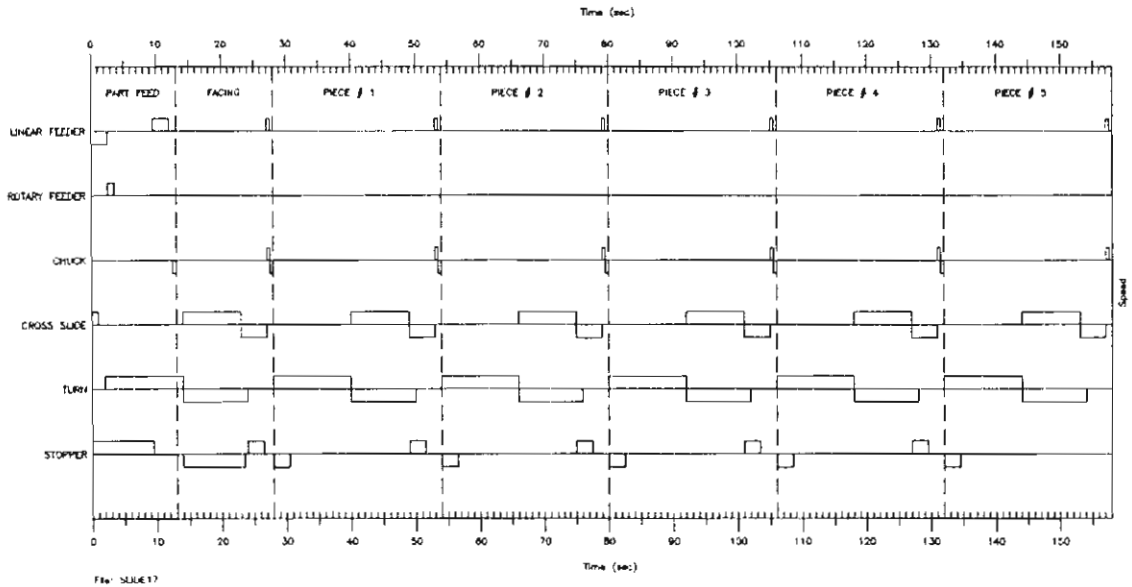
The last five fields are identical. For each of these field, one finished part will be produced. The process begin by turning operation followed by cut off operation. After the cut off operation is finished, the cut off tool is retracted, the stopper move in place, and the new material is fed by unclamping the chuck jaw while the material is being pushed by the linear feeder. After the material is in place, the chuck jaw clamps the new material and the next field of operation can begin.

8.3 Controller Board Configuration

The heart of the control unit is a microprocessor board selected from local manufacturer to guarantee spare part availability. The board used is ANT-32 made by Sila Research Inc. This particular microprocessor board use Intel 8032 as the microprocessor and two 8255

programmable peripheral interface (PPI) for input and output operation. This configuration allow up to 48 bits of input/output plus another 8 bits from the CPU port (P1.0 through P1.7). The board is configured with RS-232 communication link for future expansion and watch dog timer for automatic reset in case the CPU is locked up for any reason. Up to 32K of code memory can be programmed using EPROM. This particular company also produce several external boards such as stepper motor driver and relay boards which can be connected to the CPU board with supplied connecting cable.

OPERATION CHART



Instrumentation and Control Research Laboratory Institute for Science and Technology Research and Development	
Subject: Single Purpose Lathe Project: SUE Production Line File: SLIDE17	Designed by: Thawan Date: 26 Oct 96 Scale: 1mm=1sec Drawing No. 7

8.4 Inputs and Outputs Configuration

The single purpose lathe require several input and output for it to function properly. According to Sila convention. The First PPI is designated as U1 and the second PPI designated as U2 while the letter U stands for user port. For each user port there are three separate 8 bit ports called A, B, and C according to Intel convention. The I/O configuration list follows:

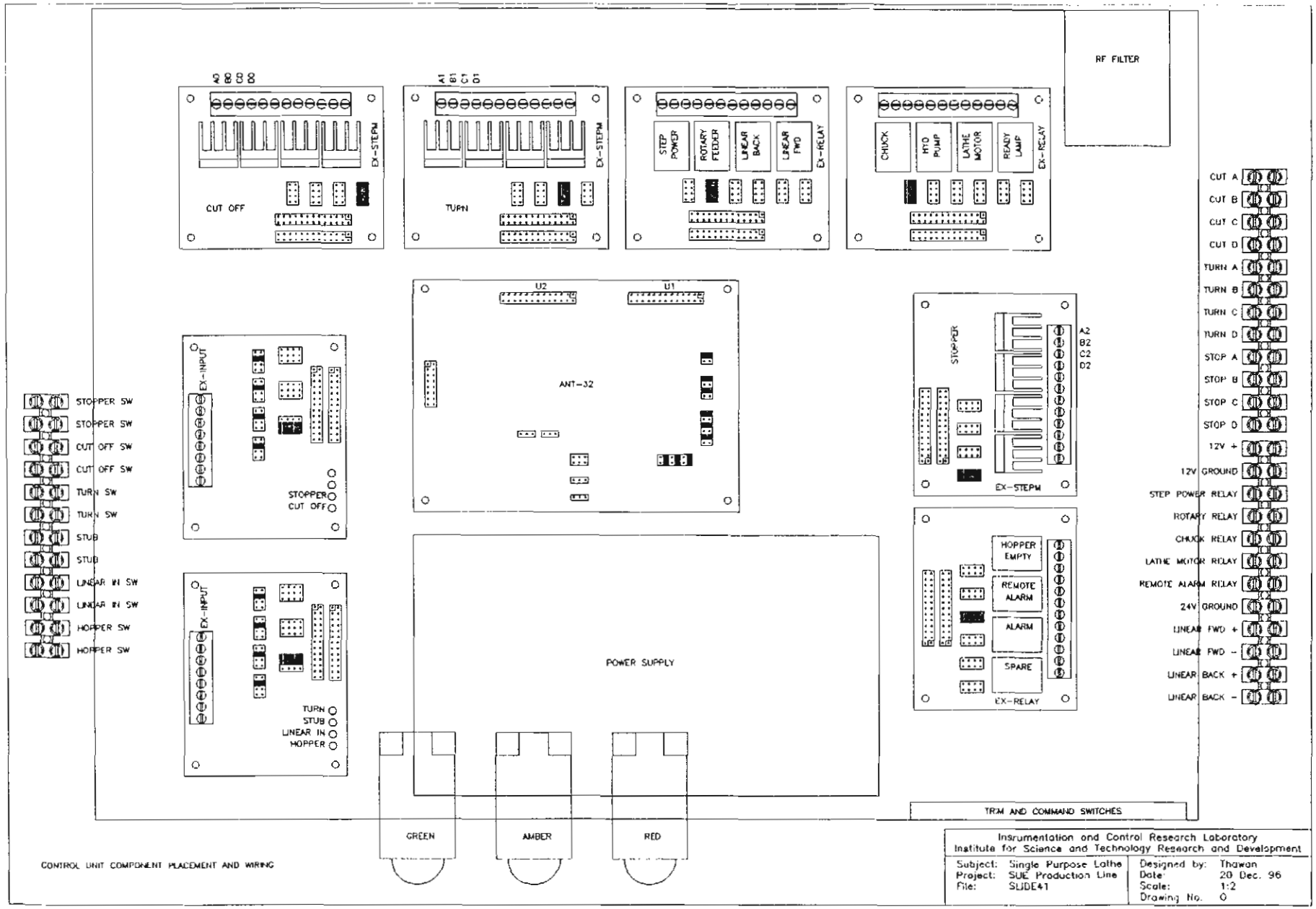
PORT	I/O	ACTIVE	DESCRIPTION
U1A0	O	LOW	MOTOR 0 (CUT) PHASE A
U1A1	O	LOW	MOTOR 0 (CUT) PHASE B
U1A2	O	LOW	MOTOR 0 (CUT) PHASE C
U1A3	O	LOW	MOTOR 0 (CUT) PHASE D
U1A4	O	LOW	MOTOR 1 (TURN) PHASE A
U1A5	O	LOW	MOTOR 1 (TURN) PHASE B
U1A6	O	LOW	MOTOR 1 (TURN) PHASE C
U1A7	O	LOW	MOTOR 1 (TURN) PHASE D
U1B0	O	HIGH	STEPPER MOTOR POWER SUPPLY
U1B1	O	HIGH	ROTARY FEEDER MOTOR ACTIVATION
U1B2	O	HIGH	LINEAR FEEDER BACK
U1B3	O	HIGH	LINEAR FEEDER FORWARD
U1B4	O	HIGH	CHUCK CONTROL
U1B5	O	HIGH	HYDRAULIC PUMP POWER
U1B6	O	HIGH	LATHE POWER CONTROL
U1B7	O	HIGH	READY LAMP
U1C0	I	HIGH	HOPPER EMPTY SWITCH
U1C1	I	LOW	LINEAR FEEDER FULL IN SWITCH
U1C2	I	LOW	LINEAR FEEDER STUB SWITCH
U1C3	I	LOW	TURNING SLIDE HOME SWITCH
U1C4	I	LOW	CUT OFF SLIDE HOME SWITCH
U1C7	I	LOW	SPARE
U2A0	O	LOW	MOTOR 2 (STOPPER) PHASE A
U2A1	O	LOW	MOTOR 2 (STOPPER) PHASE B
U2A2	O	LOW	MOTOR 2 (STOPPER) PHASE C
U2A3	O	LOW	MOTOR 2 (STOPPER) PHASE D
U2A4	O	HIGH	HOPPER EMPTY LAMP
U2A5	O	HIGH	REMOTE ALARM
U2A6	O	HIGH	SPARE
U2A7	O	HIGH	SPARE
U2B0	I	LOW	DIAMETER TRIM BIT 0
U2B1	I	LOW	DIAMETER TRIM BIT 1
U2B2	I	LOW	DIAMETER TRIM BIT 2
U2B3	I	LOW	DIAMETER TRIM BIT 3
U2B4	I	LOW	LENGTH TRIM BIT 0
U2B5	I	LOW	LENGTH TRIM BIT 1
U2B6	I	LOW	LENGTH TRIM BIT 2
U2B7	I	LOW	LENGTH TRIM BIT 3
U2C0	I	LOW	TEST CODE BIT 0
U2C1	I	LOW	TEST CODE BIT 1
U2C2	I	LOW	TEST CODE BIT 2
U2C3	I	LOW	TEST CODE BIT 3
U2C4	I	LOW	GO SWITCH
U2C5	I	LOW	STOP SWITCH
U2C6	I	LOW	EMERGENCY STOP SWITCH
U2C7	I	LOW	SPARE

8.5 Watch Dog Timer

In industrial control application, it is quite possible despite all the precautions exercised during design stage, that the CPU will lock up or get lost during it's operation. This may be due to improper hardware/software design, power surge, brown out of line power, intense electromagnetic radiation, sun spot activity, or whatever reason. When that happens, the hardware connected to the controller is out of control. The hardware could be the lathe motor which is rated at 3 hp. and sometime things can get dangerous. To prevent this problem, an electronic circuit is designed to cope with this kind of events. The circuit under discussion is called 'watch dog timer'. The timer will have to be activated (petted?) once every second to keep it happy. If the activation signal do not come regularly, it means that the CPU is not doing what it supposed to do and the watch dog timer will proceed to reset the CPU. Once reset, a good practice in controller programming is to have the CPU wait for a push button switch before it continue to work. The watch dog timer, in this case, is activate by complimenting the P1.7 bit regularly in the program loop.

8.6 Driving Multiple Step Motors

Looking at the operation chart, it can be seen that more than step motor need to work at the same time at different step rate and number of steps. The step motor used in this project is a four phase type. To make the motor turn, the phases must be exited in sequence. The step rate can be as high as 300 steps per second which means that the CPU must issue a command to the step motor driver 300 times in one second. If only one step motor is in operation at any given time, a simple software timing loop can be utilized to drive the motor. In this case, more than one step motor is working at the same time and while the motors are being operated, other control function is also required. A software loop for such operation would be hopelessly complex and the accuracy of motor speed is questionable. The approach taken in this project is to use the internal timer available in the CPU to operate the stepper motor in the background while other control function is being operated on in the foreground. In fact, two timers are operated simultaneously in the background so that the motors can run at different speed. The timer for MCS-51 family microprocessor run at 1/12 crystal speed and it operates as a counter counting number of pulses from the crystal divided by twelve When the timer reach the maximum value of FFFFh (65535) it will roll over to 0000h and at the same time generate an interrupt to the CPU if the interrupt enable register is properly programmed. When the CPU is interrupted, it will call the interrupt service subroutine and this interrupt service routine is used to issue commands to the stepper motor controller. To set the step rate of the motor, the timer register will be loaded with a value larger that zero during the interrupt service. This will make the timer fill up faster and hence increase the step rate. The value to load in to timer register can be calculated from the following equation:



CONTROL UNIT COMPONENT PLACEMENT AND WIRING

Insrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

Subject:	Single Purpose Lathe	Designed by:	Thawan
Project:	SUE Production Line	Date:	20 Dec. 96
File:	SLIDE41	Scale:	1:2
		Drawing No.	0

Reload value = round(FFFFh-f/(12*S)-T_o)

where f = Crystal frequency = 11,059,200 Hz
S = Step rate of motor (Hz)
T_o = Overhead time for reload operation (cycles) = 5

The two timers used to operate the stepper motor are assigned the same priority level so that when an interrupt is requested while the other is in progress, there will be slight delay for the first one to finish before the second routine can be executed. This causes miner speed variation of the stepper motor when two motors are being operated simultaneously.

8.7 Error Detection of Step Motor Drive

When using a step motor as a drive unit, it operates in an open loop mode. What this mean is that, when the motor is overloaded, it skip steps and the actual displacement of the actuator is not what has been programmed. This step skipping cannot be detected because no feed back information is available. The step skipping can be caused by the following events:

- The step motor is driven at excessively high step rate.
- High friction due to lack of lubrication of the slide.
- High friction due to improper adjustment of slide.
- Excessive force due to broken cutting tool insert.
- The tool is stalled because the chuck is not turning.
- Insufficient voltage is available to drive circuit.

Whatever the cause, the parts produced will most likely not be in proper tolerance and the condition must be corrected. It is not possible to detect the problem when the motor is in forward motion but when the motor reverse the direction to return to reference position (detected with limit switch) the number of backward steps can be counted and compare to the number of forward steps taken. If the two counts are differ more that a pre specified amount, skipping of steps has occurred and the operation will stop and the error indicator lamp will come on. Please note that the number of step count for the forward motion and backward motion will never be the same because the on and off position of the limit switch is not at the same position. Backlashes in the drive mechanism also contribute to this difference

To detect the reference position properly, the step count is monitored only after the switch contact has made the transition between on and off. If the limit switch malfunction, the motor can step forward forever or until something break. This condition can be prevented by programming in the maximum number of steps expected for limit switch stroke. If the number of steps exceed a pre specified value, it can be concluded that the limit switch has failed and the error lamp will come on.

8.8 Error Detection of Pneumatic Drive

The control unit is interfaced with the pneumatic system with a solenoid valve. When the valve is operated, the compressed air will rush in to the cylinder and cause the piston to move and perform the required task. The pneumatic actuator may not be able to perform it's task in the following circumstances:

- The supply air pressure is low or absent.
- The solenoid valve is stuck.
- Open circuitry between controller and valve.
- Air line is leaking.
- Mechanical binding of actuator.

The air cylinder is equipped with two reed switches. The reed switch is activated by magnetic material embedded in the piston and can detect the piston positions. The check of pneumatic device function is done by issuing a command for it to move and then monitor if the appropriate reed switch is activated within a specified time frame. If the reed switch is not activated properly, the operation stop, and the error lamp comes on.

8.9 Controller Software

The controller software is written in assembly language for speed and compactness. The software is tested on the actual machine using EPROM emulator. The final version of the software is programmed in to an EPROM and inserted in to the code memory socket of the micro controller board.

8.10 Controller Housing

The controller unit which include the CPU board, stepper motor drivers, relay boards, input board, and CPU power supply is housed in a personal computer housing. The housing is equipped with cooling fan and air filter to cope with the production line environment. The fan and filter system is designed such that the interior of the housing contains positive pressure when operating to keep out the dust and other contaminants. The stepper motor resistor pack and power relays are housed in a separate box to prevent excessive heat built up and electrical noise that could upset the CPU board.

9 TEST FUNCTIONS

The control system is designed with a rotary switch to enable individual testing of components of the single purpose lathe. This would allow easy maintenance, testing, trouble shooting, and adjustment of the lathe Each of the switch position is discussed in turn:

9.1 Position 0: Normal Operation

This is the normal run position of the lathe. The switch is kept at this position during normal operation.

9.2 Position 1: Single Piece Test

This is the single piece position. When this function is selected, the lathe will machine the part one at a time and then stop. The cycle repeat itself when the go switch is pressed. This function is used for quality control and tool setting. The part diameter and length can be adjusted using trim switches on the control unit.

9.3 Position 2: Turn Slide Test

This function is for turning slide test. After this function is selected, the go switch need to be pressed to initiate the test. The turning slide will move to the end of stroke. At this position,

the tip of the turning tool should be about 0.5 mm behind the edge of the cut off tool on the chuck side. When the go switch is pressed again, the turning tool will retract to full back position. This position should be about 26.5 mm from the edge of the cutting tool on the tail stock side. These positions can be adjusted by moving limit switches or limit switch actuator if applicable.

9.4 Position 3: Cross Slide Test

This function test the cross slide positioning mechanism. The cross slide has three critical positions. The first is against the limit switch which is the end of cut off position. The second is the turning position. The third is where cut off tool is about to make contact with raw material. For each time that the go switch is pressed, the cross slide will sequence through these positions in turn. The turning position can be adjusted by moving the limit switch and then fine tune with trim switch in the controller. The raw material contact switch is adjusted by moving the cut off tool in its holder with a part 21.30 mm diameter clamped in the chuck.

9.5 Position 4: Stopper Test

This position is used to adjust the tail stock position. When the go switch is pressed each time, the stopper will sequence between the full back position, halve way position, and full out position. The critical position for the stopper is the halve way position which control the finished part length. This position can be adjust by loosen the tail stock lock, install cylinder rod in the chuck so that the end of the rod is exactly 26.0 mm from the cut off tool edge on the tail stock side, and then operate the go switch. At this point, the step motor will push the stopper to halve way position. Since the tail stock is not locked, the entire tail stock structure will be pushed back to the appropriate position. The tail stock then can be locked in to place against the lathe bed. The trim adjustment in the control box can then be adjusted for fine tuning of the position.

9.6 Position 5: Linear Feeder Test

Position 5 is used to test the operation of the linear feeder. Pressing the go switch once in this test position will cause the linear feeder to go to full back position. At this position, the tip of the linear feeder should be cleared of the feeding rail at the rotary feeder. The position is not critical. Pressing the go switch again will cause the linear feeder to move to full out position. At the full out position, the tip of the linear feeder should be level with the chuck jaw. The stroke of the linear feeder is not adjustable and this test is provided for testing during fabrication and alignment of the feeder system.

9.7 Position 6: Rotary Feeder Test

This function is used to test the hopper and rotary feeder. Pressing the go switch once at this test position will cause the rotary feeder to feed one part in to the linear feeder rail.

9.8 Position 7: Chuck Test

This position is used to test the chuck operation. Pressing the go switch once will cause the chuck to open and pressing the switch again will cause the chuck to close. In the full open position, the chuck jaws should just be able to accommodate a 26 mm diameter cylinder.

9.9 Position 8: Lathe Motor Control

This function will test the on/off function of the lathe motor control.

10 ERROR CODES

The error code is displayed on a 20 characters liquid crystal display (LCD) module mounted on the front panel of the control unit. Normally, the error display is off and when the control unit detect any error in the machine operation, the error display will show a short error massage corresponding to the type of error that occur. During an error condition, the lathe motor and the hydraulic power unit will be turned off and the red alarm lamp as well as the remote alarm contact will be on. This section describe possible error associate with each error code and methods to test and fix the problem.

10.1 ERROR CODE 01 CROSS LIMIT

This error occur when the cross slide attempt to leave the home position. Initially, the contact of the limit switch attached to the cross slide is in close position and as the step motor turn the drive screw to move the cross slide, the switch will be released. If the switch is not released after a fixed number of steps, the error condition is detected. This error is caused by one of two things. One is the drive system malfunction and the other is the limit switch malfunction.

10.1.1 Drive Malfunction

The drive system malfunction can be caused by the following:

10.1.1.1 Loose coupling

The step motor drive action create quite a bit of torsional vibration in the related parts which can cause the set screw in the coupling to work itself loose. To check this, turn the coupling by hand and see if the associated slide move and some resistance is felt from the drive motor. If the slide does not move, the set screw on the slide side is loose. If the resistance of the step motor is not felt, the set screw on the motor side is loose. The set screw is assembled using Permatex thread lock compound, and if it was disassembled, the compound will have to be replaced. To do this, remove the lock screw using an allen wrench (L shaped wrench with hexagonal cross section) and clean it with wire brush. Put a drop of thread lock compound on the screw thread and tighten the screw.

10.1.1.2 Excessive slide friction

Excessive slide friction can be caused by improper adjustment of the slide clearance or lack of lubrication. Lubricate and adjust the slide according to lathe manufacture recommendation.

10.1.1.3 Drive motor malfunction

The drive motor function can be tested by moving test selector switch to position 2 and press the go switch. Pressing the go switch will cause the slide to alternately move forward and backward. If the motor do not turn smoothly, check the following:

a) The 12V DC power supply should have output of 12.5V or more while the drive motor is running. If the supply voltage is below this, replace the power supply.

b) The resistance of the resistor pack should be about 2.5 ohm. If the resistance is much higher than 3 ohm, replace the resistor pack.

c) Try replacing the CPU to stepper motor driver cable.

d) Try replacing the stepper driver board making sure that the jumpers are at the same positions for replacement board.

10.1.2 Limit Switch Malfunction

The limit switch can malfunction when operated with oil or water contamination. The following procedure will check out the limit switch operation.

a) With the controller power on, turn the coupling so that the slide move away from the limit switch in question.

b) Check the voltage across the limit switch terminal. With the switch released, the voltage should be 5V $\pm$ 0.5V. If the voltage is below the specified value, disconnect the wiring at limit switch and check switch resistance. Replace the limit switch if the resistance is not infinite.

c) Press the switch in, the voltage across the switch should be zero. If not, replace the switch.

d) With the switch pressed, the voltage across the switch connection terminals in the control box should be less than 0.1 V. If the voltage is higher than this, replace the switch wiring.

e) If the switch system pass all the tests and the system is still not operating correctly, try checking the voltage level at the 8255 PPI on the CPU board while operating the switch If the voltage level change between 4.5 V and 0.2 V, replace the 8255 chip.

f) After the tests, if the system still have problem, consult the software listing and change the appropriate hardware parameters. The problem may be caused by increase in backlash after long period of use.

10.2 ERROR CODE 02 CROSS STROKE

This error can occur when the cross slide return to home position and it take too few steps for the motor to accomplish this. The most likely cause for this error is the broken cutting tool. When the cutting tool is broken, the force required be drive mechanism is excessive and may cause the drive motor to skip steps which cause the number of steps backward to be less than the number of steps required to move forward.

10.3 ERROR CODE 03 CROSS NOT RET

This error occur when the cross slide return to home position and it take too many steps for the motor to accomplish this. The diagnostic procedure is the same as error code 1. If all

else fail, try to change step rate for tool movement in the software. Lower step rate will yield more motor torque and should remedy the problem.

10.4 ERROR CODE 04 TURN LIMIT

The error code 4 occur when the turning slide attempt to leave the home position but did not succeed. The diagnostic is the same as error code 1.

10.5 ERROR CODE 05 TURN STROKE

This error occur when the cross slide return to home position and it takes too few steps for the motor to accomplish this. The remedy is the same as error code 3.

10.6 ERROR CODE 06 TURN NOT RET

This error occur when the cross slide return to home position and it takes too many steps for the motor to accomplish this. The remedy is the same as error code 2.

10.7 ERROR CODE 07 STOPPER LIMIT

This code is associated with stopper drive. Look for mechanical binding and if non is found, try to trouble shoot the problem with procedure associated with error code 1.

10.8 ERROR CODE 08 STOP STROKE

This error occur when the stopper return to home position and it take too few steps for the motor to accomplish this. The remedy is the same as error code 3.

10.9 ERROR CODE 09 STOP NOT RET

This error occur when the stopper attempt to return to home position and failed.

10.10 ERROR CODE 10 CHUCK NOT CLR

This error could occur during lathe initialization when the liner feeder is operated and cannot reach it's full out position. This is caused by a piece of raw material which is still in the chuck when the lathe is turned on. Remove the raw material from the chuck and try again.

10.11 ERROR CODE 11 LINEAR NOT RE

This error code deal with linear feeder. The error is generated when a command is issued to the linear feeder and the reed switch mounted on the cylinder is not closed in a specified amount of time. This error is caused by:

a) Mechanical deformation of cylinder/shaft.
b) Binding of mechanical parts.
c) Air supply pressure is below 4 bar.
d) Pressure regulator is not properly adjusted.
e) Speed control valve is not properly adjusted.
f) Air leak in the system.
g) Reed switch malfunction.

h) Defective solenoid valve.
i) Defective drive relay.
j) Defective 24 VDC supply.

10.12 ERROR CODE 12 ROTARY PROB.

This error code is associated with rotary feeder malfunction. When the rotary feeder is operated and the linear feeder detect no raw material in the chuck, this error is generated.

10.13 ERROR CODE 13 HOPPER EMPTY

This error code is generated when the limit switch at the part hopper detect hopper empty condition

10.14 ERROR CODE 14 EMERGENCY ST

This error condition is generated when the lathe is stopped abnormally by pressing the emergency stop button.

10.15 ERROR CODE 15 STOPPED

This code is generated when the stop button is pressed. When this button is pressed, the lathe finish the piece of raw material before stopping.

10.16 ERROR CODE 16 COUNT ERROR

This error is generated when the lathe have machined five sliders from the raw material and the stub of the piece is not found by the stub detection switch. This error occur when the stopper setting is incorrect or the raw material is abnormally long.

11 HARDWARE PARAMETERS IN THE CONTROL SOFTWARE:

11.1 Overview

The program for the single purpose lathe is written in assembly language. This program must be translated in to machine code using a program called assembler and then written to a permanent memory called erasable programmable read only memory (EPROM). The EPROM is then plugged in to the microprocessor board of the single purpose lathe. The program contain hardware parameters which can be changed by the user to suit the production requirements. The hardware parameters include cut speed, move speed, forward stroke and reverse stroke of sliders. Changing these parameter is analogous to changing the setting of flow control valves and limit switch positions of hydraulically operated machines.

11.2 User Tool Kit

To enable the user to change the hardware parameter to suit the production requirement, a tool kit has been supplied for this purpose. The tool kit contain the following parts:

1. Lined suitcase to contain all the parts.
2. One master disk (3.5" IBM format) containing programs.
3. One work disk (3.5" IBM format) containing programs.

4. One EPROM programmer with power adapter.
5. One RS-232 cable for 9 pins connector.
6. One RS-232 cable for 25 pins connectors.
7. One EEPROM for program testing.
8. Three EPROM for working program.
9. Ten opaque stickers for EPROM software protection.
10. This document translated in to Thai.

Each of the tool kit parts will be discussed in turn in the following paragraphs:

11.2.1 Suitcase

The suitcase is a hard shelled suitcase lined with open cell foam rubber to protect the internal parts from shock, vibration, moisture, and temperature variations. The suitcase contain magnetic recording material and is not shielded against magnetic field. The suitcase should not be stored near strong magnetic field such as power transformers, speaker enclosures, or uninterruptable power supplies.

11.2.2 Master disk

The master disk is packed along with the work disk in a plastic container wrapped with aluminum foil to protect it from x-ray radiation at the airport while being transported from Chiang Mai. The master disk contain the original lathe control software listing as will as machine code in Intel hexadecimal format. It also contain operating system, editor, assembler, and communication software needed for changing hardware parameters. The disk is write protected and should be kept that way.

11.2.3 Work disk

The work disk is an identical copy of the master disk but is not write protected. It is used for making program modifications and if it become defective, another copy can be made from the master disk.

11.2.4 EPROM programmer

The EPROM programmer is an electronic device that take machine code from a microcomputer via series communication line and then write corresponding bit pattern in to an EPROM. Once the information is written, it is permanent and can only be erased by expose the EPROM to strong ultraviolet radiation for some amount of time depending on the light intensity.

11.2.5 RS-232 cable

The RS-232 cable is used to transfer information from a personal computer to the EPROM programmer. There are two standard type of connectors in use. One use 9 pin connector and the other use 25 pin connector. One cable is supplied for each type of connector.

11.2.6 EEPROM

One EEPROM is supplied in the kit for program testing. EEPROM is an abbreviation of electrically erasable programmable read only memory. The EEPROM is suitable for program

testing because it can be written many times by EPROM programmer without having to be erased first by ultraviolet light. This feature aid program testing but the information contained inside can be erased by power surge in the system and is not as rugged as the EPROM counter part. It is not advisable to use EEPROM to store program code in the microprocessor board for extended use.

11.2.7 EPROM

The EPROM is used to store program code the controller board. The EPROM can be written only once by the programmer. If it is desirable to rewrite an EPROM, it must be erased first by ultraviolet light through the crystal window on the chip. Once the EPROM is written, it should be protected from light by placing an opaque sticker on the crystal window.

11.2.8 Opaque sticker

The supplied opaque sticker is used to protect the code in the EPROM from being erased. The sticker should be place on the EPROM chip and cover the entire transparent window on the chip. Material like masking tape or paper sticker is not truly opaque and should not be used. Black electrical insulation tape can be used but the glue may run at elevated temperature.

11.2.9 User manual

The user manual in Thai is translated from part of the English language report. The report may be referred to for clarification.

11.3 Relations of Parameters

The hardware parameter is related to the lathe parameter. Each of the parameters are discussed in turn:

11.3.1 Pitch

The pitch for each slider is the linear distance that the slide travel when the drive screw is rotated one full revolution. The unit used is in millimeter per revolution.

11.3.2 Forward stroke

The forward stroke for each slider is the distance that the slider will move measuring from the position that the slide leave limit switch (switch contact begin to open) to the end of the travel required by the operation. Some slider such as the stopper may have several forward strokes.

11.3.3 Reverse stroke

The reverse stroke is the distance that the screw is expected to turn to bring the slider back in contact with limit switch at the point that the switch contact begin to close. The reverse stroke is longer than forward stroke because it will include backlash in the drive mechanism plus the difference between the position that the switch begin to open and switch begin to close. The closing and opening position of switch contact is always different due to histeresys built in to the switch mechanism.

11.3.4 Motor resolution

Motor resolution is the number of steps the motor must take to complete one full revolution. All the step motors used for the lathe is 200 steps per revolution but the drive method may differ. A one phase or two phase drive method will yield 200 steps per revolution but one-two phase drive will yield 400 steps per revolution.

11.3.5 Number of steps

The number of steps is the number of steps that the motor is required to make to achieve the desired stroke. The number of steps are calculated from all of the above parameters and can be written in equation form as

$$N = S/Pr$$

where N = number of steps required
S = stroke of slider in mm.
P = pitch of slider in mm/revolution
r = motor resolution in steps/revolution

The lathe being used is YAM-550 engine lathe with the following specifications:

Cross slide

pitch	4 mm/rev.
forward stroke	15 mm.
reverse stroke	16 mm.
turn stroke	3.35 mm.
motor resolution	400 steps/rev.

motor designation in software M0

Top slide

pitch	3 mm/rev.
forward stroke	30 mm.
reverse stroke	31 mm.
ready for facing	27 mm.
turn for facing	3 mm.
motor resolution	200 steps/rev.

motor designation in software M1

Stopper slide

pitch	4 mm/rev.
normal stroke	10 mm.
return from normal	12 mm.
facing stroke	37 mm.
return from facing	39 mm.
clear chuck stroke	40 mm.
return from clear	42 mm.
motor resolution	200 steps/rev.

motor designation in software M2

11.3.6 Timer reload value

The step motor speed is controlled by CPU internal timer. The timer normally begin the count at 0 and count up to 65535. When the terminal count is reached, the timer generate an interrupt which cause the CPU to turn the motor one step. At this point the CPU can reload the timer with a value larger than 0 to cause the timer to reach the terminal count earlier. The larger the reload value, the faster the step rate. The timer run at 1/12 of crystal frequency and the equation to calculate the reload value follows:

$$R = 65535 - f/12S - T_o$$

where
R = timer reload value
f = crystal frequency = 11,059,200 Hz
S = step rate (step/second or Hz)
T_o = software overhead to reset timer = 5 cycles

A table for reload value can be found in Appendix A.

11.3.7 Step rate

The step rate of the motor correspond to the tool speed required to machine a part. The tool speed is normally referred to as feed rate which has a unit of mm per chuck revolution. The step rate can be calculated from the following equation.

$$S = WFr/(60P)$$

where
S = step rate in Hz
W = chuck speed in rpm.
F = feed in mm/revolution
r = motor resolution in steps/revolution
P = drive screw pitch of slider

Once step rate is calculated, the appropriate reload value can be found on the timer reload table.

The torque-speed characteristic of a stepper motor is such that, when the step rate increase, the torque is reduced. This is the limiting condition for tool speed during tool movement without material removal. If the step rate is too high, the motor will skip steps during tool movement and cause error condition. The optimum tool speed is best determined experimentally since the drive assembly inertia and friction is difficult to calculate.

11.4 Constant Definitions

In the program listing under the heading of 'DEFINE HARDWARE PARAMETERS' contain parameters of the machine. These parameters can be changed to effect the way the single purpose lathe operates. The description if each parameter follows:

m0fstroke:
This constant refer to motor number 0 forward stroke. Motor number 0 is associated with the cross slide and the number that follow 'equ' is the number of steps to achieve this stroke. This is the number of steps that the drive motor will move to cut off the part from the raw

material after leaving limit switch. If the cutting tool do not quite reach the center of the part, this number can be increased. If the cutting tool move beyond the center of the part, the number may be decreased.

m0rstroke:
This constant is motor number 0 reverse stroke. This is the distance that the cross slide expected to move until it reach the limit switch. If the actual step required to reach switch is much larger than the specified number, it means that the slide is jammed in the reverse stroke and error condition will result. This problem may be caused by excessive friction on the slide or the step rate set to move the tool back is too high. If the motor take too few steps to reach the switch, it means that the slide is jammed in the forward stroke. The condition may be caused by defective cutting tool

m0tstroke:
This is the number of steps required for the cross slide to reach turning position. The number is the number of steps that the motor will turn beyond the limit switch position. This number is added to the diameter trim switch on the front panel before the program move the cross slide. At later stage in the lathe development, a micrometer is added to the lathe as the activator for the limit switch to ease the part diameter setting and this variable is no longer used for part diameter setting.

m0cutrelo:
This is timer reload value for cutting off the part. This number control cutting speed of the cut off tool. Higher reload value will cause the cut off rate to increase.

m0movrelo:
This is timer reload value for moving the cross slide back to limit switch position. If this value is set too high, the motor may skip step and cause error condition. If it is set too low, productivity will suffer.

m1fstroke:
The number is the number of steps required for the turning tool to complete the turning operation. The effect of the value setting is similar to m0fstroke.

m1fstroke1:
This stroke is used to prepare the turning tool before turning the head of a new part. It is used during facing operation of a new part.

m1fstroke2:
This is turning stroke used after the turning tool moved in to position by m1fstroke1. The purpose of this operation is to turn the head of a new part and get ready for facing operation.

m1rstoke:
This is the stroke used for returning the cutting tool to home position. The effect of this number is similar to m0rstroke.

m1cutrelo:
This is the timer reload value for turning The effect of this parameter is similar to m0cutrelo.

m1movrelo:

This is the timer reload value for returning the top slide to home position. The effect of the number is similar to m0movrelo

m2fstroke1:

This number is the number of steps required for the stopper to reach normal part stop position. It is referred to in other part of document as stopper position 1. The value from the front panel length trim switch is added to this value before the program start the execution of moving the stopper.

m2rstroke1:

This is the number of steps the stopper expected to move back to limit switch from position 1 described above.

m2fstroke2:

This is the stopper position 2 which is used to stop the raw material near the chuck for facing operation.

m2rstroke2:

This is the number of steps the stopper expected to move back to limit switch from position 2 described above.

m2fstroke3:

This is the stopper position 3 which is used to push the raw material in to the chuck. It is used during self test after power up when the control unit detect raw material in the lathe. This motion assure that the path of the cross slide is clear before testing.

m2rstroke3:

This is the number of steps the stopper expected to move back to limit switch from position 3 described above.

11.5 Reprogramming the Lathe

The following instruction is used to reprogram the lathe for different operating parameter. An IBM compatible computer with at least one serial port and one 1.4 Mb disk drive is required for the process

1. Connect the EPROM programmer to COM1 port of the computer using supplied RS-232 cable.
2. Power up the computer.
3. Plug in the EPROM programmer power adapter.
4. Inset the work disk in drive A:
5. Press [Alt][Ctrl][Del] to reset the computer.
6. When A>_ appear on the screen type edit[Enter]
7. The program listing will appear on the screen. Locate 'DEFINE HARDWARE PARAMETERS' section and make required alteration with arrow keys and backspace key
8. Press [Alt]FS to save the alteration.
9. Press [Alt]FQ to exit editor.
10. The assembler will be started automatically. Look for error massage and if there is non, proceed to step 11, otherwise, go back to step 6 to correct the problem.

11. Press SEL key on the EPROM programmer until the green lamp with a number corresponding to EPROM part number is lit.
12. Put a new EPROM in the green zero insertion force (ZIF) socket observing the orientation of the notch on the EPROM chip. Lock the chip securely using the lever on the ZIF socket.
13. Press FUNC key on the programmer until BLK lamp is lit.
14. Press ENT key and observe BLK lamp. If it flash, the EPROM is not blank and cannot be programmed. In this case, replace the EPROM and go back to step 13.
15. Press FUNC key on the programmer until DWL lamp is lit and press ENT.
16. At the computer type send[Enter] to send the machine code from the computer to the programmer.
17. The DWL lamp should flash for a few seconds during the data transfer and then remain on. If the lamp flash after the transfer, the communication line is bad or the serial port is not properly set. This is caused by some computer that do not allow a start up from floppy disk.
18. Press FUNC key until WR lamp is lit and press ENT.
19. If no lamp flash after a few seconds, the EPROM is ready, if a lamp flash, the EPROM selection in step 11 may not be correct.
20. Remove power from the control unit and replace the EPROM.
21. Power up the lathe and test the operation.

For testing of the operation and experimenting with parameters, use the supplied EEPROM and skip step 13 and 14. Replace the EEPROM with EPROM when everything is working right.

12. PROBLEMS ENCOUNTERED AND SOLUTIONS

The nature of a good research project is to find something new and suitable for the task at hand. For this reason, it is expected that unforeseen problems will present itself during the research and must be overcome. This section list the problems encountered and solutions used to solve them in chronological order.

The single purpose lathe under development is a complex network of computerized control system interfaced with mechanical and pneumatic feed system, hydraulic gripping system, step motor drive system and feedback systems of switches. During the development, many types of problem have occurred and solved. Some of the problems are trivial and should have been foreseen while some only become trivial after careful analysis. These problems are documented for future work and next generation of researchers working on machine tool retrofitting. Each of the problems along with causes and remedies are discussed in turn.

Raw Material Hopper Jamming:

Originally, the hopper for the raw material is designed as a triangular bucket with two sloping walls at forty-five degree. This configuration would work well with material of reasonable surface finish. The raw material used is cast iron caste with sand mold and has very irregular surface finish which cause the material to jam against each other when they converge at the bottom of the hopper and the feed operation become unreliable. The hopper is redesigned as a long rail that hold only one layer of the raw material to solve this problem.

Rotary Feeder Impact:

After the hopper is redesigned, the new hopper configuration causes all the raw material mass to sit on the rotary feeder and when the rotary feeder operates, large impact can be felt when a new piece fall in to the rotary feeder slot. The rotary feeder drum is machined to have gradual slope when receiving new part to reduce the impact to acceptable level.

Linear Feeder Impact:

During the lathe operation, the raw material is fed through the chuck center hole against the stopper before machining process begins. The original design is to push the material with pneumatic cylinder against the chuck, when the chuck opens, the air pressure will push the material against the stopper. The speed of the material feed will be controlled by the flow control valve attached to the air cylinder. As it happens, the air cylinder is near the end of stroke during the feed cycle and large volume of compressed air is stored behind the piston before the chuck open. When the chuck opens, the compressed air become decompressed and push the material against the stopper at high speed causing large impact. The flow control valve is not in control of the cylinder speed at this moment. The solution is to change the solenoid valve from one solenoid system to two solenoid system to enable the cylinder to unload and release all air pressure during the idle cycle and then push forward only when the chuck begins to open. This change require a new valve, alteration of the system wiring, and alteration of control software.

Linear Feeder Air Valve Problem:

The supply pressure to the system is regulated to control the air cylinder force on the raw material. Some of the time, the air valve do not operate properly even though the regulated pressure is well above the valve minimum operating pressure. After extensive testing and installation of a pressure gage behind the regulator, it was found that during the valve opening, the flow is large and cause the regulated pressure supplying the valve to drop below the valve operating pressure and proper opening of the valve cannot be achieved. The regulator has been changed to a large one and regulated pressure increased to rectify the problem.

Lathe Bore Adapter Length:

In the original design, the lathe bore adapter will reach just behind the chuck jaw to carry the raw material to the machining area. As it turn out, the chuck jaw is much thicker than expected and some binding can occur during the feeding of the last piece. This can be dangerous because during such time, the chuck is rotating at 1020 rpm. The lathe bore adapter was extended to the front of the chuck grip with slot machined on the end to enable the adapter to carry the part while allowing the chuck to grip the part at the same time.

Programmable Peripheral Interface Initialization:

The central processing unit of the control system perform most if it's input and output operation through a 8255 programmable peripheral interface (PPI) integrated circuit. The PPI seems not to function properly after power up but work some of the time after CPU reset. It was found that the PPI require some time after power up and after receiving programming command to work properly. This feature of the PPI chip is not documented anywhere and the source of the problem is found after consultation with an experienced PPI

user. A software delay routine is inserted in to the control program during power up and after PPI programming commands to solve the problem.

Stepper Motor Power Supply Overload During Power Up:

All three stepper motors are connected to the CPU via the PPI chip. During power up, a small delay is needed for the PPI to operate properly. During this delay period of about 0.1 second, all of the PPI output is in low state which is the state that turn on stepper motor phases. During this time, the power demand for all the motors far exceed the power supply capability and cause the power transistor of the power supply to be damaged. A relay is added to the system to turn on the stepper motor power supply after the PPI has been initialized to prevent this problem.

Assembler Problem:

During testing, the cross slide motor tends to turn on and off randomly. Examination of program code show no apparent explanation for the behavior. CPU boards has been changed with no success. An assistant has been assigned to examine the machine code against the assembly language coding as a last resort of debugging the system. It has been found that the carry flag of the CPU is referred to a symbol 'C' when operation is being done on the flag and during arithmetic operation but when referred to during program branching operation, the machine code produced by the assembler refers to bit number zero in the bit addressable area which control the cross slide motor. This conflict should have been reported as error by the assembler but unintended code was produced instead. When the symbol 'C' has been changed to 'CY' according to Intel convention for program branching operation, the problem was solved. The assembler used is a legally purchased SXA51 8051/8052 cross-assembler version 1.3 by Binary Technology Inc. Meriden, NH, USA.

Control Unit Interference:

The control unit was designed as a single enclosure which contain the CPU as well as power resistors and power relays to control the operation of the lathe. Some times the CPU lock up when some of the high power equipment is switched on and off. This problem probably cause by the current surge and the control unit is rewired with separate box containing the power equipment control.

Finished Part Taper:

The turning operation is done by the top slide which has a taper adjustment. The finished part shows about 0.02 mm taper when measured with micrometer. It is next to impossible to adjust the top slide normally to compensate for such fine taper so a screw adjustment mechanism is fitted to the slide for fine taper adjustment.

Inconsistency of part diameter:

During test runs, it was found that the part diameter vary from one another at random in 0.02 mm increments. Dial indicator is installed on the cross slide to verify the situation and found that the cross slide control lack consistency. Limit switch was tested and found to be repeatable within 0.005 mm and has no connection with the problem. Force of about 50 Kgf was exert on the cross slide tool post and no significant deflection was found. Step rate for the control motor was reduced from 800 steps per second to 400 steps per second and the

problem disappear. The lathe cross slide is programmed so that it slow down before hitting the limit switch.

Part out of round and has unpredictable taper:

Part manufactured by the lathe is out of round and the taper is not predictable. Some part is even smaller around the middle section. The hydraulic pressure to the chuck was increased from 30 bar to 38 bar (the chuck pressure limit is 40 bar) with small improvement. Closer inspection reveal that the chuck grip was worn more in the extremities that the middle. The soft jaw grip was replaced with a set of hard grip provided by SUE and the problem disappear. The hydraulic pressure is set back to 30 bar and the problem did not return.

Flying stub problem:

On some occasions, the ejected stub bounced back from the stopper or the turning tool while it is being ejected and fly toward the rotating chuck. The stub is then thrown out of the chuck with centrifugal force at some respectable velocity. This is deemed to be an operational hazard and a guard plate is installed in front of the chuck to prevent the problem.

Last piece inconsistency:

It was found that the fifth piece machined from each piece of raw material has dimensional variations. The variation take the from of out of round, taper, and diameter error. After numerous trials and monitor of cross slide movement as well as reconditioning of turn slide, the problem persist. During one test, the lathe make strange noise and was switched off immediately and it was found that the linear feeder cannot be retracted. The problem is traced to the tip of linear feeder which interfere with the chuck gripping mechanism. This occur after installation of the guard plate in front of the chuck which necessitate the relocation of the cut off tool. The linear feeder now have to move further to feed the part and interfere with the gripper jaw. The tip of the linear feeder was machined to smaller diameter and solve the problem.

The 'good' part went out of tolerance overnight:

Machined parts changed diameter when left unattended. The staff was asked to verify the collected data on the part diameter and come up with inconsistent results. The same part measured the previous day with the same micrometer just got smaller by 0.01 mm. The person who took the initial measurement was called in to make the measurement again and get the same strange result. The coefficient of expansion for cast iron is0.0000059/F and assuming that the just machined part is at 80 C and the ambient temperature in the morning is 20 C, calculation shows that the part can change its diameter up to 0.0136 mm from this temperature variation. SUE had advised that it is the policy of the plant to reduce the amount of dust and suggest that coolant could be used to solve this problem.

Part diameter change during lathe warm up:

It was observed that the part diameter changes consistently during the first 4 pieces of raw material. The change is about 0.015 mm. The expansion of turning tool may be responsible for this problem. The use of coolant also solved this problem.

The first piece is always larger than the last piece:

The first piece is always larger than the last piece of the raw material by about 0.005 mm. It is speculated that the last piece being machined is hotter than the first one. The use of cutter coolant also solve this problem.

Size measurement is not repeatable:

The same part measured at the same spot twice yield slightly different result. The difference is about 0.003 mm and may be due to smoothing out of surface roughness. The variation is too small to cause any serious problem and is documented for reference.

13. PROJECT EVALUATION

The retrofitted lathe for the first stage of slider production works but it failed occasionally. The mode of failure is almost always traced to electronic parts used in manufacturing of control components. It is a known fact that Thailand is the dump site of barely work electronic parts which has been rejected from various factories around the world. If you wish to see the above mentioned dump site, please tell the Bangkok taxi to drop you off at a place called Clong Thom.

The lathe can produce parts with good consistency and the test data is shown in the appendix. The goal of 30 second cycle time was not achieved due to lack of good stepper motor driver and some oversight during threshing phase and the possible cycle time for this lathe configuration is 40 seconds. This is of no great consequence because by the time that this project was approved, SUE has already plan their new production line and automated machinery ordered. By the time that the project is finished, the new production line is already in use. The old production line can use this lathe as the first stage machining for the slider and the line will be in use soon when SUE increase their production rate again. The lathe will provide the promised prevention of workforce and floor space increase when the production is geared up for 1.5 million units per year.

14. CONCLUSION AND SUGGESTIONS

The microcomputer controlled lathe was built, installed at SUE Udomsuk plant and tested. The development process was challenging and numerous problems have been solved to get the lathe working. Valuable experience has been gained by all parties involve and the know how on production technology of Thailand has went up a notch. For suggestions, if a servo motor drive and PLC controller was used for the lathe, it could have achieve the 30 seconds cycle time targeted with higher reliability. But then, the cost would have tripled.

References.

1. AB Sandvik Coromant LTD., Turning Tools Catalog. AB Sandvik Coromant LTD., Sandviken, Sweden, 1995, p 12.
2. E. Paul DeGarmo, Material and Processes in Manufacturing, Fifth Edition, Macmillan Publishing Co., INC. New York, 1979, p 517.
3. E. Paul DeGarmo, Material and Processes in Manufacturing, Fifth Edition, Macmillan Publishing Co., INC. New York, 1979, p 532.
4. Sanyo Denki Co., LTD, Stepping Motor Step-Syn Catalog No. 560- 1 88.8.2DB, Sanyo Denki Co., LTD., Japan, 1988, p 43.

Appendix A. Timer Reload Value Table

Hz	Reload	Hz	Reload	Hz	Reload	Hz	Reload	Hz	Reload	Hz	Reload
15	4090	65	51352	115	57516	165	59945	215	61243	265	62052
16	7930	66	51566	116	57585	166	59978	216	61263	266	62065
17	11318	67	51775	117	57653	167	60011	217	61283	267	62078
18	14330	68	51977	118	57720	168	60044	218	61302	268	62091
19	17025	69	52173	119	57785	169	60077	219	61322	269	62104
20	19450	70	52364	120	57850	170	60109	220	61341	270	62117
21	21644	71	52550	121	57913	171	60141	221	61360	271	62129
22	23639	72	52730	122	57976	172	60172	222	61379	272	62142
23	25460	73	52905	123	58037	173	60203	223	61397	273	62154
24	27130	74	53076	124	58098	174	60233	224	61416	274	62166
25	28666	75	53242	125	58157	175	60264	225	61434	275	62179
26	30084	76	53404	126	58216	176	60294	226	61452	276	62191
27	31397	77	53561	127	58273	177	60323	227	61470	277	62203
28	32616	78	53715	128	58330	178	60352	228	61488	278	62215
29	33751	79	53864	129	58386	179	60381	229	61506	279	62227
30	34810	80	54010	130	58441	180	60410	230	61523	280	62239
31	35801	81	54152	131	58495	181	60438	231	61540	281	62250
32	36730	82	54291	132	58548	182	60466	232	61558	282	62262
33	37603	83	54426	133	58601	183	60494	233	61575	283	62273
34	38424	84	54559	134	58652	184	60521	234	61592	284	62285
35	39199	85	54688	135	58703	185	60548	235	61608	285	62296
36	39930	86	54814	136	58754	186	60575	236	61625	286	62308
37	40622	87	54937	137	58803	187	60602	237	61641	287	62319
38	41277	88	55057	138	58852	188	60628	238	61658	288	62330
39	41899	89	55175	139	58900	189	60654	239	61674	289	62341
40	42490	90	55290	140	58947	190	60679	240	61690	290	62352
41	43052	91	55403	141	58994	191	60705	241	61706	291	62363
42	43587	92	55513	142	59040	192	60730	242	61722	292	62374
43	44097	93	55620	143	59085	193	60755	243	61737	293	62385
44	44585	94	55726	144	59130	194	60779	244	61753	294	62395
45	45050	95	55829	145	59174	195	60804	245	61768	295	62406
46	45495	96	55930	146	59218	196	60828	246	61784	296	62416
47	45921	97	56029	147	59261	197	60852	247	61799	297	62427
48	46330	98	56126	148	59303	198	60875	248	61814	298	62437
49	46722	99	56221	149	59345	199	60899	249	61829	299	62448
50	47098	100	56314	150	59386	200	60922	250	61844	300	62458
51	47459	101	56405	151	59427	201	60945	251	61858	301	62468
52	47807	102	56495	152	59467	202	60968	252	61873	302	62478
53	48141	103	56582	153	59506	203	60990	253	61887	303	62488
54	48463	104	56668	154	59546	204	61012	254	61902	304	62498
55	48774	105	56753	155	59584	205	61034	255	61916	305	62508
56	49073	106	56836	156	59622	206	61056	256	61930	306	62518
57	49362	107	56917	157	59660	207	61078	257	61944	307	62528
58	49640	108	56997	158	59697	208	61099	258	61958	308	62538
59	49910	109	57075	159	59734	209	61120	259	61972	309	62547
60	50170	110	57152	160	59770	210	61141	260	61985	310	62557
61	50422	111	57227	161	59806	211	61162	261	61999	311	62567
62	50665	112	57301	162	59841	212	61183	262	62012	312	62576
63	50901	113	57374	163	59876	213	61203	263	62026	313	62586
64	51130	114	57446	164	59910	214	61223	264	62039	314	62595

Appendix A. Timer Reload Value Table (continue)

Hz	Reload	Hz	Reload	Hz	Reload	Hz	Reload	Hz	Reload	Hz	Reload
315	62604	365	63005	415	63309	465	63548	515	63740	565	63899
316	62614	366	63012	416	63315	466	63552	516	63744	566	63902
317	62623	367	63019	417	63320	467	63557	517	63747	567	63905
318	62632	368	63026	418	63325	468	63561	518	63751	568	63907
319	62641	369	63032	419	63330	469	63565	519	63754	569	63910
320	62650	370	63039	420	63336	470	63569	520	63758	570	63913
321	62659	371	63046	421	63341	471	63573	521	63761	571	63916
322	62668	372	63053	422	63346	472	63577	522	63764	572	63919
323	62677	373	63059	423	63351	473	63582	523	63768	573	63922
324	62686	374	63066	424	63356	474	63586	524	63771	574	63924
325	62694	375	63072	425	63362	475	63590	525	63775	575	63927
326	62703	376	63079	426	63367	476	63594	526	63778	576	63930
327	62712	377	63085	427	63372	477	63598	527	63781	577	63933
328	62720	378	63092	428	63377	478	63602	528	63785	578	63936
329	62729	379	63098	429	63382	479	63606	529	63788	579	63938
330	62737	380	63105	430	63387	480	63610	530	63791	580	63941
331	62746	381	63111	431	63392	481	63614	531	63794	581	63944
332	62754	382	63117	432	63397	482	63618	532	63798	582	63946
333	62762	383	63124	433	63402	483	63622	533	63801	583	63949
334	62771	384	63130	434	63406	484	63626	534	63804	584	63952
335	62779	385	63136	435	63411	485	63630	535	63807	585	63955
336	62787	386	63142	436	63416	486	63634	536	63811	586	63957
337	62795	387	63149	437	63421	487	63638	537	63814	587	63960
338	62803	388	63155	438	63426	488	63641	538	63817	588	63963
339	62811	389	63161	439	63431	489	63645	539	63820	589	63965
340	62819	390	63167	440	63435	490	63649	540	63823	590	63968
341	62827	391	63173	441	63440	491	63653	541	63826	591	63971
342	62835	392	63179	442	63445	492	63657	542	63830	592	63973
343	62843	393	63185	443	63450	493	63661	543	63833	593	63976
344	62851	394	63191	444	63454	494	63664	544	63836	594	63978
345	62859	395	63197	445	63459	495	63668	545	63839	595	63981
346	62866	396	63203	446	63464	496	63672	546	63842	596	63984
347	62874	397	63209	447	63468	497	63676	547	63845	597	63986
348	62882	398	63214	448	63473	498	63679	548	63848	598	63989
349	62889	399	63220	449	63477	499	63683	549	63851	599	63991
350	62897	400	63226	450	63482	500	63687	550	63854	600	63994
351	62904	401	63232	451	63487	501	63690	551	63857	601	63997
352	62912	402	63237	452	63491	502	63694	552	63860	602	63999
353	62919	403	63243	453	63496	503	63698	553	63863	603	64002
354	62927	404	63249	454	63500	504	63701	554	63866	604	64004
355	62934	405	63254	455	63505	505	63705	555	63869	605	64007
356	62941	406	63260	456	63509	506	63709	556	63872	606	64009
357	62948	407	63266	457	63513	507	63712	557	63875	607	64012
358	62956	408	63271	458	63518	508	63716	558	63878	608	64014
359	62963	409	63277	459	63522	509	63719	559	63881	609	64017
360	62970	410	63282	460	63527	510	63723	560	63884	610	64019
361	62977	411	63288	461	63531	511	63726	561	63887	611	64022
362	62984	412	63293	462	63535	512	63730	562	63890	612	64024
363	62991	413	63299	463	63540	513	63734	563	63893	613	64027
364	62998	414	63304	464	63544	514	63737	564	63896	614	64029

Appendix A. Timer Reload Value Table (continue)

Hz	Reload	Hz	Reload	Hz	Reload	Hz	Reload	Hz	Reload	Hz	Reload
615	64031	665	64144	715	64241	765	64325	815	64399	865	64465
616	64034	666	64146	716	64243	766	64327	816	64401	866	64466
617	64036	667	64148	717	64245	767	64328	817	64402	867	64467
618	64039	668	64150	718	64246	768	64330	818	64403	868	64468
619	64041	669	64152	719	64248	769	64332	819	64405	869	64469
620	64044	670	64154	720	64250	770	64333	820	64406	870	64471
621	64046	671	64157	721	64252	771	64335	821	64407	871	64472
622	64048	672	64159	722	64254	772	64336	822	64409	872	64473
623	64051	673	64161	723	64255	773	64338	823	64410	873	64474
624	64053	674	64163	724	64257	774	64339	824	64412	874	64476
625	64055	675	64165	725	64259	775	64341	825	64413	875	64477
626	64058	676	64167	726	64261	776	64342	826	64414	876	64478
627	64060	677	64169	727	64262	777	64344	827	64416	877	64479
628	64062	678	64171	728	64264	778	64345	828	64417	878	64480
629	64065	679	64173	729	64266	779	64347	829	64418	879	64482
630	64067	680	64175	730	64268	780	64348	830	64420	880	64483
631	64069	681	64177	731	64269	781	64350	831	64421	881	64484
632	64072	682	64179	732	64271	782	64351	832	64422	882	64485
633	64074	683	64181	733	64273	783	64353	833	64424	883	64486
634	64076	684	64183	734	64274	784	64354	834	64425	884	64487
635	64079	685	64185	735	64276	785	64356	835	64426	885	64489
636	64081	686	64187	736	64278	786	64357	836	64428	886	64490
637	64083	687	64189	737	64280	787	64359	837	64429	887	64491
638	64085	688	64190	738	64281	788	64360	838	64430	888	64492
639	64088	689	64192	739	64283	789	64362	839	64432	889	64493
640	64090	690	64194	740	64285	790	64363	840	64433	890	64494
641	64092	691	64196	741	64286	791	64365	841	64434	891	64496
642	64094	692	64198	742	64288	792	64366	842	64435	892	64497
643	64097	693	64200	743	64290	793	64368	843	64437	893	64498
644	64099	694	64202	744	64291	794	64369	844	64438	894	64499
645	64101	695	64204	745	64293	795	64371	845	64439	895	64500
646	64103	696	64206	746	64295	796	64372	846	64441	896	64501
647	64106	697	64208	747	64296	797	64374	847	64442	897	64503
648	64108	698	64210	748	64298	798	64375	848	64443	898	64504
649	64110	699	64212	749	64300	799	64377	849	64444	899	64505
650	64112	700	64213	750	64301	800	64378	850	64446	900	64506
651	64114	701	64215	751	64303	801	64379	851	64447	901	64507
652	64117	702	64217	752	64304	802	64381	852	64448	902	64508
653	64119	703	64219	753	64306	803	64382	853	64450	903	64509
654	64121	704	64221	754	64308	804	64384	854	64451	904	64511
655	64123	705	64223	755	64309	805	64385	855	64452	905	64512
656	64125	706	64225	756	64311	806	64387	856	64453	906	64513
657	64127	707	64226	757	64313	807	64388	857	64455	907	64514
658	64129	708	64228	758	64314	808	64389	858	64456	908	64515
659	64132	709	64230	759	64316	809	64391	859	64457	909	64516
660	64134	710	64232	760	64317	810	64392	860	64458	910	64517
661	64136	711	64234	761	64319	811	64394	861	64460	911	64518
662	64138	712	64236	762	64321	812	64395	862	64461	912	64519
663	64140	713	64237	763	64322	813	64396	863	64462	913	64521
664	64142	714	64239	764	64324	814	64398	864	64463	914	64522

Appendix B. Firmware Program Listing.

```
;Program MTEST.ASM last update 28 Jan 1997
;
;PROGRAM DESCRIPTION
;
;This program is written to control a single purpose lathe which
;is a joint research effort between CMU, TDO, and SUE. The hardware
;used for this program is Sila Research ANT-32 micro controller board
;equipped with Intel 8032 microprocessor. The main control function
;is done in the foreground while stepper motor operation is carried out
;with 2 levels of timer interrupt. The assembler used is SXA51 8051/
;8052 cross-assembler version 1.3 by Binary Technology Inc. Meriden,
;NH. USA.
;
;In case of difficulties, the programmer can be reached at:
;
;  Thawan Sucharitakul
;  Department of Mechanical Engineering
;  Faculty of Engineering
;  Chiang Mai University, Chiang Mai 50200
;  THAILAND
;  Phone and Fax: (053) 210-320
;  E-Mail enagi001@chiangmai.ac.th
;
;HARDWARE CONFIGURATION
;
; 8255 PPI #1 at data memory F800h - F803h operate in mode 0
; 8255 PPI #2 at data memory FC00h - FC03h operate in mode 0
;
;  PORT   I/O ACTIVE  DESCRIPTION
;
;  U1A0   O  LOW    MOTOR0 (CUT) PHASE A
;  U1A1   O  LOW    MOTOR0 (CUT) PHASE B
;  U1A2   O  LOW    MOTOR0 (CUT) PHASE C
;  U1A3   O  LOW    MOTOR0 (CUT) PHASE D
;  U1A4   O  LOW    MOTOR1 (TURN) PHASE A
;  U1A5   O  LOW    MOTOR1 (TURN) PHASE B
;  U1A6   O  LOW    MOTOR1 (TURN) PHASE C
;  U1A7   O  LOW    MOTOR1 (TURN) PHASE D
;  U1B0   O  HIGH   STEPPER MOTOR POWER
;  U1B1   O  HIGH   ROTARY FEEDER MOTOR
;  U1B2   O  HIGH   LINEAR FEEDER BACK
;  U1B3   O  HIGH   LINEAR FEEDER FORWARD
;  U1B4   O  HIGH   CHUCK CONTROL
;  U1B5   O  HIGH   HYDRAULIC PUMP POWER
;  U1B6   O  HIGH   LATHE POWER CONTROL
;  U1B7   O  HIGH   READY LAMP
;  U1C0   I  HIGH   HOPPER EMPTY SWITCH
;  U1C1   I  LOW    LINEAR FEEDER FULL IN SWITCH
;  U1C2   I  LOW    LINEAR FEEDER STUB SWITCH
;  U1C3   I  LOW    TURNING SLIDE HOME SWITCH
;  U1C4   I  LOW    CUT OFF SLIDE HOME SWITCH
;  U1C5   I  LOW    STOPPER HOME SWITCH
;  U1C6   I  LOW    SPARE
```

Appendix B. Firmware Program Listing (continue).

```
;  U1C7   I  LOW   SPARE
;  U2A0   O  LOW   MOTOR2 (STOPPER) PHASE A
;  U2A1   O  LOW   MOTOR2 (STOPPER) PHASE B
;  U2A2   O  LOW   MOTOR2 (STOPPER) PHASE C
;  U2A3   O  LOW   MOTOR2 (STOPPER) PHASE D
;  U2A4   O  HIGH  HOPPER EMPTY LAMP
;  U2A5   O  HIGH  REMOTE ALARM
;  U2A6   O  HIGH  SPARE
;  U2A7   O  HIGH  SPARE
;  U2B0   I  LOW   DIAMETER TRIM BIT 0
;  U2B1   I  LOW   DIAMETER TRIM BIT 1
;  U2B2   I  LOW   DIAMETER TRIM BIT 2
;  U2B3   I  LOW   DIAMETER TRIM BIT 3
;  U2B4   I  LOW   LENGTH TRIM BIT 0
;  U2B5   I  LOW   LENGTH TRIM BIT 1
;  U2B6   I  LOW   LENGTH TRIM BIT 2
;  U2B7   I  LOW   LENGTH TRIM BIT 3
;  U2C0   I  LOW   TEST CODE BIT 0
;  U2C1   I  LOW   TEST CODE BIT 1
;  U2C2   I  LOW   TEST CODE BIT 2
;  U2C3   I  LOW   TEST CODE BIT 3
;  U2C4   I  LOW   GO SWITCH
;  U2C5   I  LOW   STOP SWITCH
;  U2C6   I  LOW   EMERGENCY STOP SWITCH
;  U2C7   I  LOW   SPARE
;  P1.7   O        WATCH DOG TIMER
;
;MEMORY MAP
;
;  Register banks          00 - 1F
;  Bit-addressable variables  20 - 2F
;  Byte variables          30 - 4F
;  Stack area              50 - 7F
;
;ERROR CODES
;
;  01 CROSS LIMIT
;     Cross slide did not leave limit switch in forward motion.
;     Possible causes: loose coupling, low or missing stepper supply,
;                      loose limit switch, limit switch malfunction,
;                      limit switch wiring short circuit, step motor
;                      wiring bad, step motor driver bad, resistor pack
;                      open circuit, slider jammed, chuck not turning.
;  02 CROSS STROKE
;     Cross slide forward stroke too short.
;     Possible causes: loose coupling, low stepper supply,
;                      slider jammed, chuck not turning.
;  03 CROSS NOT RET
;     Cross slide cannot reach home position.
;     Possible causes: loose coupling, low stepper supply,
;                      slider jammed, loose limit switch, limit
;                      switch malfunction, limit switch wiring
;                      open circuit, step rate set too high.
```

Appendix B. Firmware Program Listing (continue).

```
; 04 TURN LIMIT
;    Turn slide did not leave limit switch in forward motion.
;    Possible causes: loose coupling, low or missing stepper supply,
;                     loose limit switch, limit switch melfunction,
;                     limit switch wiring short circuit, step motor
;                     wiring bad, step motor driver bad, resistor pack
;                     open circuit, slider jammed, chuck not turning.
; 05 TURN STROKE
;    Turn slide forward stroke too short.
;    Possible causes: loose coupling, low stepper supply,
;                     slider jammed, chuck not turning.
; 06 TURN NOT RET
;    Turn slide cannot reach home position.
;    Possible causes: loose coupling, low stepper supply,
;                     slider jammed, loose limit switch, limit
;                     switch melfunction, limit switch wiring
;                     open circuit, step rate set too high.
; 07 STOPPER LIMIT
;    Stopper did not leave limit switch in forward motion.
;    Possible causes: loose coupling, low or missing stepper supply,
;                     loose limit switch, limit switch melfunction,
;                     limit switch wiring short circuit, step motor
;                     wiring bad, step motor driver bad, resistor pack
;                     open circuit, slider jammed, chuck did not open.
;                     stopper gear box not in proper position.
; 08 STOP STROKE
;    Stopper forward stroke too short.
;    Possible causes: loose coupling, low stepper supply,
;                     slider jammed, chuck did not open.
; 09 STOP NOT RET
;    Stopper cannot reach home position.
;    Possible causes: loose coupling, low stepper supply,
;                     loose limit switch, limit switch melfunction,
;                     limit switch wiring open circuit, step rate
;                     set too high, stopper gear box not in proper
;                     position.
; 10 CHUCK NOT CLR
;    Raw material remain in the chuck during self test.
;    Possible causes: Raw material remain in the chuck during self test,
;                     low or missing air supply, bad air solenoid valve,
;                     low or missing 24 volt supply, forward reed switch
;                     not in proper place or loose, reed switch melfunction,
;                     chuck did not open, linear slider jammed.
; 11 LINEAR NOT RE
;    Linear feeder cannot move back to full in position.
;    Possible causes: Low or missing air supply, bad air solenoid valve,
;                     low or missing 24 volt supply, back reed switch
;                     not in proper place or loose, reed switch melfunction,
;                     chuck did not open, linear slider jammed.
; 12 ROTARY PROB.
;    Rotary feeder problem, no part is fed when commanded.
;    Possible causes: Low or missing 12V supply, feeder jammed, short
;                     circuit in hopper switch or wiring, broken rotary
;                     feeder limit switch, bad rotary feeder wiring.
; 13 HOPPER EMPTY
;    There is no raw material in the hopper.
;    Possible causes: There is no raw material in the hopper, bad hopper
;                     switch, open circuit in hopper switch wiring.
```

```
;                material jammed in the upper part of the hopper.
;                limit switch arm bent
;  14 EMERGENCY ST
;     Emergency stop switch pressed, system shut down immediately.
;     Possible causes: Emergency stop switch pressed, emergency stop
;                switch melfunction, front panel circuit board
;                melfunction.
;  15 STOPPED
;     Stop switch pressed, system stop at end of machining cycle.
;          resume by pressing GO or press stop again for system
;          shut down.
;     Possible causes: Stop switch pressed, stop switch melfunction,
;                front panel circuit board melfunction.
;  16 COUNT ERROR
;     Stub sensor did not read stub after 5 cylinders was machined.
;
:DEFINE PORT ADDRESSE
   aport1  equ 0F800h     :Port A of 8255-1
   bport1  equ aport1+1   :Port B of 8255-1
   cport1  equ bport1+1   :Port C of 8255-1
   rport1  equ cport1+1   :Control register of 8255-1
   aport2  equ 0FC00h     :Port A of 8255-2
   bport2  equ aport2+1   :Port B of 8255-2
   cport2  equ bport2+1   :Port C of 8255-2
   rport2  equ cport2+1   :Control register of 8255-2
   wr_inst   equ  0FA00h  :Write instruction port of LCD
   rd_busy   equ  0FA01h  :Read busy instruction port of LCD
   wr_data   equ  0FA02h  :Write data port of LCD
   rd_data   equ  0FA03h  :Read data port of LCD

:DEFINE VARIABLES IN BIT ADDRESSABLE AREA (128 bit max)
   m0done  bit 00h        :Motor 0 done
   m0rev   bit m0done+1   :Motor 0 reverse
   m0tsel  bit m0rev+1    :Motor 0 timer select
   m1done  bit m0tsel+1   :Motor 1 done
   m1rev   bit m1done+1   :Motor 1 reverse
   m1tsel  bit m1rev+1    :Motor 1 timer select
   m2done  bit m1tsel+1   :Motor 2 done
   m2rev   bit m2done+1   :Motor 2 reverse
   m2tsel  bit m2rev+1    :Motor 2 timer select
   stopped bit m2tsel+1   :Stop request flag
   estop   bit stopped+1  :Emergency stop request flag
   stub    bit estop+1    :Stub detection flsg
   m0speed bit stub+1     :Cross slide speed up
   holdm0  bit m0speed+1  :Cross slide hold
   mes1    bit holdm0+1   :enable massage set 1
   m0dbo   bit mes1+1     :Motor 0 limit switch debounce

:DEFINE VARIABLES IN MEMORY (36 byte max, change SP if need more)
   m0sth   equ 30h        :Motor 0 step count high byte
   m0stl   equ m0sth+1    :Motor 0 step count low byte
   m0ph    equ m0stl+1    :Phase count motor 0
   m1sth   equ m0ph+1     :Motor 1 step count high byte
   m1stl   equ m1sth+1    :Motor 1 step count low byte
   m1ph    equ m1stl+1    :Phase count motor 1
   m2sth   equ m1ph+1     :Motor 2 step count high byte
   m2stl   equ m2sth+1    :Motor 2 step count low byte
   m2ph    equ m2stl+1    :Phase count motor 2
   relo0h  equ m2ph+1     :Timer0 reload high byte
   relo0l  equ relo0h+1   :Timer0 reload low byte
```

```
    relo1h  equ relo0l+1    ;Timer1 reload high byte
    relo1l  equ relo1h+1    ;Timer1 reload low byte
    error   equ relo1l+1    ;Error code
    phomeh  equ error+1     ;post home pos. high byte M0 only
    phomel  equ phomeh+1    ;post home pos. low byte
    pcount  equ phomel+1    ;Piece count during machining
    m0hold  equ pcount+1    ;Motor 0 hold period

;reload value = 0FFFFh - crystal frequency /(12*step rate) - 5
;FFFF = 65535, crystal frequency = 11,059,200 Hz
    m0fstroke  equ  1700   ;15mm / 4mm/rev * 400steps/rev =1500steps
    m0rstroke  equ  1800   ;16mm / 4mm/rev * 400steps/rev = 1600steps
    m0begincut equ  1400   ;begin cut at 11 mm from center
    m0lstroke  equ  600    ;3.35mm/4mm/rev*400steps/rev = 335 steps
    m0cutrelo  equ  60109  ;cut speed reload value 170 Hz
    m0movrelo  equ  63039  ;move speed reload value 370 Hz
    m0limrelo  equ  57850  ;Hit limit switch speed 120 Hz
    m1fstroke  equ  2100   ;30mm / 3mm/rev * 200steps/rev = 2000steps
    m1fstroke1 equ  1800   ;Ready for clean up for facing
    m1fstroke2 equ  300    ;Clean up for facing
    m1rstroke  equ  2150   ;33mm / 3mm/rev * 200steps/rev = 2200steps
;   m1cutrelo  equ  60109  ;cut speed reload value 170 Hz
    m1cutrelo  equ  58703  ;cut speed reload value 135 Hz
    m1movrelo  equ  61844  ;move speed reload value 250 Hz
    m2fstroke1 equ  494    ;10mm / 4mm/rev * 200steps/rev = 500 steps
    m2rstroke1 equ  580    ;12mm / 4mm/rev * 200steps/rev = 600 steps
    m2fstroke2 equ  1850   ;37mm / 4mm/rev * 200steps/rev = 1850steps
    m2rstroke2 equ  1950   ;39mm / 4mm/rev * 200steps/rev = 1950steps
    m2fstroke3 equ  2000   ;40mm / 4mm/rev * 200steps/rev = 2000steps
    m2rstroke3 equ  2100   ;42mm / 4mm/rev * 200steps/rev = 2100steps
    m2movrelo  equ  62117  ;move speed reload value 270 Hz

   org    00h             ;PC location after reset
    jmp    MAIN            ;go to main program

    org    0Bh             ;PC location for timer0 interrupt
    jmp    STEP0           ;go to timer0 interrupt service routine

    org    1Bh             ;PC location for timer1 interrupt
Appendix B. Firmware Program Listing (continue).

    jmp    STEP1           ;go to timer1 interrupt service routine

    org    100h            ;start location of main program
MAIN:           ;Main program start here
    mov    SP,#4Fh         ;set stack pointer for 64 byte stack area
    call   INI8255         ;Get 8255 port ready
    call   INIVARS         ;Initiate variables
    call   INICLOCK        ;Initiate timer interrupt
    call   INIT_LCD        ;Set up LCD display
    call   CHUCK_OPEN      ;Open chuck head
    call   STEP_POWER_ON   ;Turn on 12V supply
    call   WAITONE         ;Wait one second
    call   READY_LAMP_ON   ;Turn on ready lamp
MAINLOOP:
    call   READ_TEST       ;Read test switch
```

```
    jnz   MAINLOOP1      :Loop if no test
    call  RUN1
    jmp   MAINLOOP
  MAINLOOP1:
    call  TEST0          :Go to test routine
    jmp   MAINLOOP       :Loop again

INISYS:          :Initialize system and test components
    setb  mes1
    mov   a,#1
    call  DISPLAY
    call  ALARM_LAMP_OFF  :Turn off alarm lamp
    call  HYD_POWER_ON    :Power up hydraulic pump
    call  CHUCK_OPEN      :Open chuck grip
    call  WAITONE         :Wait two seconds for hydraulic pressure
    call  WAITONE
    call  LINEAR_F_BACK   :Move linear feeder full back
    call  LINEAR_OFF      :Unload linear feeder
    call  CHECK_LINEAR_B  :Check linear feeder back switch
    call  M0HOME          :Home cross slide
    call  WAITM0          :Wait until done
    call  CHECK_CROSS_H   :See if cross slide home properly
    call  M1HOME          :Home turn slide
    call  WAITM1          :Wait until done
    call  CHECK_TURN_H    :See if turn slide home properly
    call  M2HOMEPOS3      :Home stopper
    call  WAITM2          :Wait until done
    call  WAIT            :switch bounce?
    call  CHECK_STOP_H    :See if stopper home properly
    call  WAIT            :Wait 0.1 sec
    call  M2POS3          :Push stopper all the way in
    call  WAITM2          :Wait until done
    call  CHECK_STOP_F    :Check stopper forward operation
    call  WAIT            :Wait 1/10 sec
    call  M2HOMEPOS3      :Home stopper
    call  WAITM2          :Wait until done
    call  CHECK_STOP_R    :Check stopper operation
    call  M0CUT           :Move cross slide
    call  WAITM0          :Wait for cross slide done
    call  CHECK_CROSS_F   :Check cross slide forward motion
    call  WAIT            :Wait 1/10 sec before motor reversal
    call  M0HOME          :Move cross slide home
    call  WAITM0          :Wait for cross slide to home
    call  CHECK_CROSS_R   :Check cross slide operation
    call  M1TURN          :Move turn slide
    call  WAITM1          :Wait for turn slide to finish
    call  CHECK_TURN_F    :Check turn slide forward motion
    call  WAIT            :Wait 1/10 sec before motor reversal
    call  M1HOME          :Home turn slide
    call  WAITM1          :Wait for turn slide to home
    call  CHECK_TURN_R    :Check turn slide reverse motion
    call  M2POS1          :Move stopper in
    call  M0_OFF_LIMIT    :Move cross slide off limit sw.
    call  WAITM0          :Wait till done
    call  WAIT            :Wait 1/10 sec.
    call  M0TURN          :Move turning tool to turn position
    call  WAITM0          :Wait for cross slide
    call  CHECK_CROSS_H   :See if cross slide home properly
    call  WAITM2          :Wait for stopper
```

```
    call   CHECK_STOP_F   ;Check stopper forward operation
    call   LINEAR_F_FWD   ;Move linear feeder in
    call   LINEAR_OFF     ;Unload linear feeder
    mov    a,error        ;Check error
    jnz    INISYS_ERR     ;Chuck not clear, stop system
    call   WAITONE        ;Wait for linear feeder to unload
    call   LINEAR_F_BACK  ;Move linear feeder back
    mov    a,error        ;Check for problem
    jnz    INISYS_ERR     ;No air pressure, stop system
    setb   mes1
    mov    a,#15
    call   DISPLAY
    ret
  INISYS_ERR:             ;The jnz command cannot make long jump
    jmp    ALL_STOP       ;Stop all system and flash alarm

PART_FEED:     ;Feed new part and facing
    setb   mes1
    mov    a,#2
    call   DISPLAY
    call   READ_HOPPER    ;Check hopper switch
    jz     PART_FEED_A    ;Skip if not empty
    mov    error,#13
    jmp    HOPPER_EMPTY   ;Go to hopper empty condition
  PART_FEED_A:
    call   CHUCK_OPEN     ;Make sure chuck is open
    call   M2HOMEPOS1     ;Home stopper
    call   WAITM2         ;Wait for stopper
    call   CHECK_STOP_H   ;See if stopper home properly
    call   WAIT           ;Wait 1/10 sec before motor reversal
    call   M2POS2         ;Begin move stopper in
    call   M1TURN1        ;Move turning tool to ready position
    call   LINEAR_F_BACK  ;Move linear feeder back
    mov    a,error        ;Check for problem
    jnz    INISYS_ERR     ;No air pressure, stop system
    call   ROTARY_GO      ;Activate rotary feeder
    call   WAITONE        ;Wait one second
    call   WAITM2         ;Wait for stopper to finish moving
    call   CHECK_STOP_F   ;Stopper left limit switch?
    call   WAITM1         ;Wait for turning tool to finish
    call   CHECK_TURN_F   ;Turn slide left limit switch?
    call   LINEAR_F_FWD   ;Linear feeder in
    mov    a,error
    jnz    PART_FEED_B    ;PROBLEM!! part not in, rotary feeder problem
    mov    error,#12
    jmp    ALL_STOP

  PART_FEED_B:
    call   CHUCK_CLOSE    ;Close chuck
    call   WAITONE        ;Wait for chuck to close
    call   LINEAR_OFF     ;Unload linear feeder
    call   M2HOMEPOS2     ;Home stopper
    call   M1TURN2        ;Turn to get ready for facing
    call   WAITM1         ;Wait for turning done
    call   M0CUT          ;Cut off the face
    call   WAIT           ;Wait 1/10 sec before motor reversal
    call   M1HOME         ;Home turn slide
    call   WAITM0         ;Wait for cross slide
    call   CHECK_CROSS_F  ;Cross slide left limit switch?
    call   WAIT           ;Delay 1/10 sec
```

```
    call   M0TURN          :Position cross slide for turning
    call   WAITM1          :Wait for turn slide to finish
    call   CHECK_TURN_R    :Check turn slide reversal motion
    call   WAITM2          :Wait for stopper to finish
    call   CHECK_STOP_R    :Check stopper reversal motion
    call   WAIT            :Wait before stopper reversal
    call   M2POS1          :Move stopper in to place
    call   WAITM0          :Wait for cross slide
    call   CHECK_CROSS_H   :Check cross slide home?
    call   WAITM2          :Wait for stopper
    call   CHECK_STOP_F    :Stopper left limit switch?
    setb   mes1
    mov    a,#15
    call   DISPLAY
    ret

MAKE_SLIDE:    :Make slider part
    setb   mes1
    mov    a,#3
    call   DISPLAY
    call   FEED_AND_CHECK  :Feed material and check for stub
    call   M2HOMEPOS1      :Home stopper
    call   M1TURN          :Turn the part
    call   WAITM1          :Wait for turning to finish
    call   CHECK_TURN_F    :Turning slide off limit sw?
    call   CHECK_STOP_H    :Stopper home OK?
    call   M0CUT           :Begin cutting off
    call   WAIT            :Wait 1/10 sec before motor reversal
    call   M1HOME          :Home turning slide
    call   WAITM0          :Wait for cutting done
    call   CHECK_CROSS_F   :Cross slide left limit sw?
    call   WAIT            :Wait 1/10 sec before motor reversal
    call   M0TURN          :Return cross slide
    call   WAITM1          :Wait for turn slide to home
    call   CHECK_TURN_R    :Turn slide movement OK?
    call   M2POS1          :Move stopper in
    call   WAITM0          :Wait for cross slide
    call   CHECK_CROSS_H   :Cross slide movement OK?
    call   WAITM2          :Wait for stopper
    call   CHECK_STOP_F    :Stopper left sw OK?
    setb   mes1
    mov    a,#15
    call   DISPLAY
    ret

EJECT_STUB:    :Eject remaining stub
    setb   mes1
    mov    a,#4
    call   DISPLAY
    call   M0_OFF_LIMIT    :Move cross slide off limit switch
    call   M2HOMEPOS2      :Back off stopper
    call   WAITM0          :Wait for the operation
    call   WAITM2          :Wait for stopper to home
    call   WAIT            :switch bounce?
    call   WAIT
    call   CHECK_STOP_H    :Stopper home OK?
    call   LINEAR_FOR      :Push linear feeder to eject part
    call   CHUCK_OPEN      :Open chuck
    call   WAITONE         :Wait two seconds
    call   M0TURN          :Position turning tool
```

```
    call   WAITM0          :Wait for cross slide
    call   CHECK_CROSS_H   :Cross slide back?
    call   LINEAR_OFF      :Unload linear feeder
    setb   mes1
    mov    a,#15
    call   DISPLAY
    ret

FEED_AND_CHECK: :Feed part and check for stub (2 sec.)
    clr    EA              :Stop all interrupts
    clr    stub            :Clear stub bit
    call   LINEAR_FOR
    call   CHUCK_OPEN      :Open chuck
    mov    r2,#255         :Load loop counter for 0.85 sec.
  FEED_AND_CHECK3:
    mov    r1,#255         :Load loop counter
  FEED_AND_CHECK2:
    call   READ_STUB       :Read stub switch (9 ms)
    jnz    FEED_AND_CHECK1 :Stub detected?  (2 ms)
    setb   stub            :Set flag if so  (1 ms)
  FEED_AND_CHECK1:
    djnz   r1,FEED_AND_CHECK2 :Loop back (2 ms)
    djnz   r2,FEED_AND_CHECK3 :Loop back
    call   CHUCK_CLOSE     :Close the chuck
    call   WAITONE         :Wait for fully closed
    call   LINEAR_OFF
    setb   EA              :Restart interrupts
    ret

LINEAR_F_BACK:  :Pull linear feed back and check for problem
    mov    error,#0        :clear error code
    call   LINEAR_BACK     :Move linear feeder back
    mov    r3,#255         :reload loop counter for 1.7 sec
  LINEAR_F_BACK3:
    mov    r2,#200         :reload loop counter
  LINEAR_F_BACK2:
    mov    r1,#3           :reload loop counter (USE 2)
  LINEAR_F_BACK1:
    call   READ_LINEAR_IN  :Check back switch (9 ms)
    jz     LINEAR_F_BACK0  :Get out if switch reached (2 ms)
    djnz   r1,LINEAR_F_BACK1 :(2 ms) Total loop time = 13 ms
    djnz   r2,LINEAR_F_BACK2
    djnz   r3,LINEAR_F_BACK3
    mov    error,#11       :set error code if time expire
    mov    p1,error        :set error display
  LINEAR_F_BACK0:
    call   LINEAR_OFF      :Unload linear feeder
    ret

LINEAR_F_FWD.   :Linear forward feed and check for problem
    mov    error,#10       :preset error code
    call   LINEAR_FOR      :Move linear feeder forward
    mov    r3,#255         :reload loop counter for 1.7 sec
  LINEAR_F_FWD3:
    mov    r2,#255         :reload loop counter
  LINEAR_F_FWD2:
    mov    r1,#2           :reload loop counter (USE 2)
  LINEAR_F_FWD1:
    call   READ_STUB       :Check back switch (9 ms)
    jnz    LINEAR_F_FWD0   :clear error if switch reached (2 ms)
```

```
    mov     error,#0
  LINEAR_F_FWD0:
    djnz    r1,LINEAR_F_FWD1 ;(2 ms) Total loop time = 13 ms
    djnz    r2,LINEAR_F_FWD2
    djnz    r3,LINEAR_F_FWD3
    ret

 RUN1:          ;System run
    call    INISYS          ;Self test
  RUN1C:
    call    CHUCK_MOTOR_ON
    call    WAITONE
    call    PART_FEED
    mov     pcount,#6
  RUN1A:
    dec     pcount
    mov     a,pcount
    jnz     RUN1B
    setb    mes1
    mov     error,#14
    jmp     ALL_STOP
  RUN1B:
    call    MAKE_SLIDE
    jnb     stub,RUN1A
    call    EJECT_STUB
    call    HOPPER_LAMP_OFF
    call    LINEAR_F_BACK
    jmp     RUN1C
    ret

TEST0:          ;Test routine for system parts
    call    WAITGO          ;Wait for go switch pressed
    call    READ_TEST       ;read test routine number
    rl      a               ;Multiply a by 2
    mov     DPTR,#JUMPTAB   ;Point to jump table
    jmp     @a+DPTR         ;Jump to selected test routine

JUMPTAB:        ;Jump table for test routines
    ajmp    TEST00          ;Dummy routine
    ajmp    TEST1           ;Single piece test
    ajmp    TEST2           ;Turn slide test
    ajmp    TEST3           ;Cross slide test
    ajmp    TEST4           ;Stopper test
    ajmp    TEST5           ;Linear feeder test
    ajmp    TEST6           ;Rotary feeder test
    ajmp    TEST7           ;Chuck test
    ajmp    TEST8           ;Lathe motor test
    ajmp    TEST9           ;Tool set

TEST00:         ;Dummy subroutine
    ret

TEST1:          ;System cycle test
    call    INISYS          ;Self test
    call    WAITGO
  TEST1C:
    call    CHUCK_MOTOR_ON
    call    HOPPER_LAMP_ON
    call    WAITONE
    call    PART_FEED
```

```
    mov    pcount,#6
  TEST1A:
    dec    pcount
    mov    a,pcount
    jnz    TEST1B
    setb   mes1
    mov    error,#14
    jmp    ALL_STOP
  TEST1B:
    call   MAKE_SLIDE
    jnb    stub,TEST1A
    call   EJECT_STUB
    call   HOPPER_LAMP_OFF
    call   LINEAR_F_BACK
    call   CHUCK_MOTOR_OFF
    call   WAITGO
    jmp    TEST1C
    ret

TEST2:          ;Turning slide test
    call   ALARM_LAMP_OFF
    call   HOPPER_LAMP_ON
    call   M1HOME
    call   WAITM1
    call   WAIT
    call   M1TURN
    call   WAITM1
    call   WAIT
    call   M1HOME
    call   WAITM1
    call   WAIT
    call   HOPPER_LAMP_OFF
    call   CHECK_TURN_R
    mov    a,error
    jnz    TEST2_A
    ret
  TEST2_A:
    call   ALARM_LAMP_ON
    ret

TEST3:          ;Cross slide test
    call   ALARM_LAMP_OFF
    call   HOPPER_LAMP_ON
    call   M0HOME
    call   WAITM0
    call   WAIT
    call   M0CUT
    call   WAITM0
    call   WAIT
    call   M0TURN
    call   WAITM0
    call   HOPPER_LAMP_OFF
    call   CHECK_CROSS_H
    mov    a,error
    jnz    TEST3_A
    ret
  TEST3_A:
    call   ALARM_LAMP_ON
    ret
```

```
TEST4          :Stopper test
  call   ALARM_LAMP_OFF
  call   HOPPER_LAMP_ON
  call   M2HOMEPOS3
  call   WAITM2
  call   WAIT
  call   M2POS1
  call   WAITM2
  call   WAIT
  call   M2HOMEPOS1
  call   WAITM2
  call   WAIT
  call   HOPPER_LAMP_OFF
  call   CHECK_STOP_R
  mov    a.error
  jnz    TEST4_A
  ret
 TEST4_A:
  call   ALARM_LAMP_ON
  ret

TEST5:         :Linear feeder test
  call   ALARM_LAMP_OFF
  call   HOPPER_LAMP_ON
  call   LINEAR_F_FWD
  call   LINEAR_OFF
  mov    a.error
  jz     TEST5A
  call   ALARM_LAMP_ON
 TEST5A:
  call   WAITONE
  call   ALARM_LAMP_OFF
  call   LINEAR_F_BACK
  call   LINEAR_OFF
  mov    a.error
  jz     TEST5B
  call   ALARM_LAMP_ON
 TEST5B:
  call   HOPPER_LAMP_OFF
  ret

TEST6:         :Rotary feeder test
  call   HOPPER_LAMP_ON
  call   ROTARY_GO
  call   WAITONE
  call   HOPPER_LAMP_OFF
  ret

TEST7:         :Chuck test
  call   HOPPER_LAMP_ON
  call   CHUCK_CLOSE
  call   WAITONE
  call   WAITONE
  call   CHUCK_OPEN
  call   WAITONE
  call   WAITONE
  call   HOPPER_LAMP_OFF
  ret
```

```
TEST8:          ;Lathe motor test
   call   HOPPER_LAMP_ON
   call   CHUCK_MOTOR_ON
   call   WAITONE
   call   WAITONE
   call   CHUCK_MOTOR_OFF
   call   HOPPER_LAMP_OFF
   ret

TEST9:          ;Tool set position
   call   CHUCK_OPEN
   call   M0HOME
   call   WAITM0
   call   M1HOME
   call   WAITM1
   call   M2HOMEPOS3
   call   WAITM2
   call   M0CUT
   call   WAITM0
   call   M0TURN
   call   WAITM0
   call   M1SET
   call   WAITM1
   call   HOPPER_LAMP_ON
   call   WAITGO
   call   HOPPER_LAMP_OFF
   call   CHUCK_CLOSE
   call   WAITONE
   call   HOPPER_LAMP_ON
   call   WAITGO
   call   HOPPER_LAMP_OFF
   call   M2POS1
   call   WAITM2
   call   HOPPER_LAMP_ON
   call   WAITGO
   call   HOPPER_LAMP_OFF
   call   CHUCK_OPEN
   call   M2HOMEPOS1
   call   WAITM2
   ret

M0HOME:         ;Home cross slide to reference position (T0 4 sec)
   mov    m0sth,#high m0rstroke   ;Load step counter high byte
   mov    m0stl,#low m0rstroke    ;Load step counter low byte
   mov    relo0h,#high m0movrelo  ;Load timer reload high byte
   mov    relo0l,#low m0movrelo   ;Load timer reload low byte
   mov    phomel,#20      ;set post home variable for no reverse
   mov    phomeh,#0
   setb   m0rev           ;set reverse bit
   clr    m0tsel          ;select timer 0
   clr    m0done          ;Go
   ret

M0_OFF_LIMIT:   ;Move cross slide off limit switch (T0 0.84 sec)
   mov    m0sth,#0                ;Load step counter high byte
   mov    m0stl,#5                ;Load step counter low byte
   mov    relo0h,#high m0movrelo  ;Load timer reload high byte
   mov    relo0l,#low m0movrelo   ;Load timer reload low byte
   clr    m0rev           ;clear reverse bit
```

```
    clr    m0tsel          :select timer 0
    clr    m0done          :Go
    ret

M0CUT:          :Cut off operation (T0 8.82 sec)
    mov    m0sth,#high m0fstroke   :Load step counter high byte
    mov    m0stl,#low m0fstroke    :Load step counter low byte
    mov    relo0h,#high m0movrelo  :Load timer reload high byte
    mov    relo0l,#low m0movrelo   :Load timer reload low byte
    clr    m0rev           :clear reverse bit
    clr    m0tsel          :select timer 0
    clr    m0done          :Go
    ret

M0TURN:         :Move cross slide to turn position (T0 4.84 sec)
    mov    m0sth,#high m0rstroke   :Load step counter high byte
    mov    m0stl,#low m0rstroke    :Load step counter low byte
    mov    phomeh,#high m0tstroke
    mov    phomel,#low m0tstroke
    call   READ_DTRIM      :Read trim switch, result in a
    add    a,phomel        :add trim to step counter low byte
    mov    phomel,a        :save the value
    clr    a               :clear a register
    addc   a,phomeh        :add any carry to high byte
    mov    phomeh,a        :save the value
    mov    relo0h,#high m0movrelo  :Load timer reload high byte
    mov    relo0l,#low m0movrelo   :Load timer reload low byte
    setb   m0rev           :set reverse bit
    clr    m0tsel          :select timer 0
    clr    m0done          :Go
    ret

M1HOME:         :Home turn slide (T1 11.0 sec)
    mov    m1sth,#high m1rstroke   :Load step counter high byte
    mov    m1stl,#low m1rstroke    :Load step counter low byte
    mov    relo1h,#high m1movrelo  :Load timer reload high byte
    mov    relo1l,#low m1movrelo   :Load timer reload low byte
    setb   m1rev           :set reverse bit
    setb   m1tsel          :select timer 1
    clr    m1done          :Go
    ret

M1SET:          :Turn operation
    mov    m1sth,#2                :Load step counter high byte
    mov    m1stl,#0                :Load step counter low byte
    mov    relo1h,#high m1cutrelo  :Load timer reload high byte
    mov    relo1l,#low m1cutrelo   :Load timer reload low byte
    clr    m1rev           :clear reverse bit
    setb   m1tsel          :select timer 1
    clr    m1done          :Go
    ret

M1TURN:         :Turn operation (T1 11.8 sec)
    mov    m1sth,#high m1fstroke   :Load step counter high byte
    mov    m1stl,#low m1fstroke    :Load step counter low byte
    mov    relo1h,#high m1cutrelo  :Load timer reload high byte
    mov    relo1l,#low m1cutrelo   :Load timer reload low byte
    clr    m1rev           :clear reverse bit
    setb   m1tsel          :select timer 1
    clr    m1done          :Go
```

```
    ret

M1TURN1:        :Move turning tool to ready pos for facing (T0 10.6)
    mov     m1sth,#high m1fstroke1  :Load step counter high byte
    mov     m1stl,#low m1fstroke1   :Load step counter low byte
    mov     relo1h,#high m1movrelo  :Load timer reload high byte
    mov     relo1l,#low m1movrelo   :Load timer reload low byte
    clr     m1rev           :clear reverse bit
    clr     m1tsel          :select timer 0
    clr     m1done          :Go
    ret

M1TURN2:        :Turning tool to clean up for facing (T0 1.2 sec)
    mov     m1sth,#high m1fstroke2  :Load step counter high byte
    mov     m1stl,#low m1fstroke2   :Load step counter low byte
    mov     relo1h,#high m1cutrelo  :Load timer reload high byte
    mov     relo1l,#low m1cutrelo   :Load timer reload low byte
    clr     m1rev           :clear reverse bit
    clr     m1tsel          :select timer 0
    clr     m1done          :Go
    ret

M2HOMEPOS1:     :Home stopper from normal position (T0 2.4 sec)
    mov     m2sth,#high m2rstroke1  :Load step counter high byte
    mov     m2stl,#low m2rstroke1   :Load step counter low byte
    mov     relo0h,#high m2movrelo  :Load timer reload high byte
    mov     relo0l,#low m2movrelo   :Load timer reload low byte
    setb    m2rev           :set reverse bit
    clr     m2tsel          :select timer 0
    clr     m2done          :Go
    ret

M2HOMEPOS2:     :Home stopper from facing position (T1 7.8 sec)
    mov     m2sth,#high m2rstroke2  :Load step counter high byte
    mov     m2stl,#low m2rstroke2   :Load step counter low byte
    mov     relo1h,#high m2movrelo  :Load timer reload high byte
    mov     relo1l,#low m2movrelo   :Load timer reload low byte
    setb    m2rev           :set reverse bit
    setb    m2tsel          :select timer 1
    clr     m2done          :Go
    ret

M2HOMEPOS3:     :Home stopper from clear chuck position (T1 8.4)
    mov     m2sth,#high m2rstroke3  :Load step counter high byte
    mov     m2stl,#low m2rstroke3   :Load step counter low byte
    mov     relo1h,#high m2movrelo  :Load timer reload high byte
    mov     relo1l,#low m2movrelo   :Load timer reload low byte
    setb    m2rev           :set reverse bit
    setb    m2tsel          :select timer 1
    clr     m2done          :Go
    ret

M2POS1:         :Move stopper to normal position (T1 2.0 sec)
    mov     m2sth,#high m2fstroke1  :Load step counter high byte
    mov     m2stl,#low m2fstroke1   :Load step counter low byte
    call    READ_LTRIM      :Read trim switch, result in a
    add     a,m2stl         :add trim to step counter low byte
```

```
  mov   m2stl,a        ;save the value
  clr   a              ;clear a register
  addc  a,m2sth        ;add any carry to high byte
  mov   m2sth,a        ;save the value
  mov   relo1h,#high m2movrelo  ;Load timer reload high byte
  mov   relo1l,#low m2movrelo   ;Load timer reload low byte
  clr   m2rev          ;clear reverse bit
  setb  m2tsel         ;select timer 1
  clr   m2done         ;Go
  ret

M2POS2:        ;Move stopper to facing position (T1 7.4 sec)
  mov   m2sth,#high m2fstroke2  ;Load step counter high byte
  mov   m2stl,#low m2fstroke2   ;Load step counter low byte
  mov   relo1h,#high m2movrelo  ;Load timer reload high byte
  mov   relo1l,#low m2movrelo   ;Load timer reload low byte
  clr   m2rev          ;clear reverse bit
  setb  m2tsel         ;select timer 1
  clr   m2done         ;Go
  ret

M2POS3:        ;Move stopper to clear chuck of part (T1 8.0 sec)
  mov   m2sth,#high m2fstroke3  ;Load step counter high byte
  mov   m2stl,#low m2fstroke3   ;Load step counter low byte
  mov   relo1h,#high m2movrelo  ;Load timer reload high byte
  mov   relo1l,#low m2movrelo   ;Load timer reload low byte
  clr   m2rev          ;clear reverse bit
  setb  m2tsel         ;select timer 1
  clr   m2done         ;Go
  ret

STEP0:         ;Timer0 interrupt service routine
  clr   TR0            ;Stop timer
  mov   TL0,relo0l     ;LSB timer reload
  mov   TH0,relo0h     ;MSB timer reload
  setb  TR0            ;Run timer
  push  PSW            ;Save program status word
  push  DPL            ;Save lower byte of data pointer
  push  DPH            ;Save high byte of data pointer
  push  b              ;Save b register
  push  acc            ;Save accumulator
  call  READ_STOP      ;Read stop switch
  jnz   STEP07
  setb  stopped        ;If pressed, set flag
STEP07:
  call  READ_E_STOP    ;Read stop switch
  jnz   STEP08
  setb  estop          ;If pressed, set flag
  jmp   EMERGENCY_STOP
STEP08:
  jb    m0tsel,STEP02  ;This timer selected?
  jb    m0done,STEP02  ;Motor 0 done?
  jb    m0rev,STEP01   ;Reverse direction?
  call  M0FORWARD      ;Step forward
  jmp   STEP02         ;Done with motor 0
STEP01:
  call  M0REVERSE      ;Step back
STEP02:
```

```
    jb     m1tsel,STEP03   ;This timer selected?
    jb     m1done,STEP03   ;Motor 1 done?
    jb     m1rev,STEP04    ;Reverse direction?
    call   M1FORWARD       ;Step forward
    jmp    STEP03          ;Done with motor 1
  STEP04:
    call   M1REVERSE       ;Step back
  STEP03:
    jb     m2tsel,STEP05   ;This timer selected?
    jb     m2done,STEP05   ;Motor 2 done?
    jb     m2rev,STEP06    ;Reverse direction?
    call   M2FORWARD       ;Step forward
    jmp    STEP05          ;Done with motor 2
  STEP06:
    call   M2REVERSE       ;Step back
  STEP05:

  STEP0DONE:
    pop    acc             ;Restore accumulator
    pop    b               ;Restore b register
    pop    DPH             ;Restore data pointer hi byte
    pop    DPL             ;Restore data pointer lo byte
    pop    PSW             ;Restore program status word
    reti

STEP1           ;Timer1 interrupt service routine
    clr    TR1             ;Stop timer
    mov    TL1,#lo1l       ;LSB timer reload
    mov    TH1,#lo1h       ;MSB timer reload
    setb   TR1             ;Run timer
    push   PSW             ;Save program status word
    push   DPL             ;Save lower byte of data pointer
    push   DPH             ;Save high byte of data pointer
    push   b               ;Save b register
    push   acc             ;Save accumulator
    call   READ_STOP       ;Read stop switch
    jnz    STEP17
    setb   stopped         ;If pressed, set flag
  STEP17:
    call   READ_E_STOP     ;Read stop switch
    jnz    STEP18
    setb   estop           ;If pressed, set flag
    jmp    EMERGENCY_STOP
  STEP18:
    jnb    m0tsel,STEP12   ;This timer selected?
    jb     m0done,STEP12   ;Motor 0 done?
    jb     m0rev,STEP11    ;Reverse direction?
    call   M0FORWARD       ;Step forward
    jmp    STEP12          ;Done with motor 0
  STEP11:
    call   M0REVERSE       ;Step back
  STEP12:
    jnb    m1tsel,STEP13   ;This timer selected?
    jb     m1done,STEP13   ;Motor 1 done?
    jb     m1rev,STEP14    ;Reverse direction?
    call   M1FORWARD       ;Step forward
    jmp    STEP13          ;Done with motor 1
```

```
STEP14:
  call   M1REVERSE      :Step back
STEP13:
  jnb    m2tsel.STEP15  :This timer selected?
  jb     m2done.STEP15  :Motor 2 done?
  jb     m2rev.STEP16   :Reverse direction?
  call   M2FORWARD      :Step forward
  jmp    STEP15         .Done with motor 2
STEP16:
  call   M2REVERSE      :Step back
STEP15:
STEP1DONE:
  pop    acc            :Restore accumulator
  pop    b              :restore b register
  pop    DPH            :Restore data pionter hi byte
  pop    DPL            :Restore data pointer lo byte
  pop    PSW            :Restore program status word
  ret

M0FORWARD:      :Cross slide move forward
  mov    a,m0stl        :load lsb of step counter
  jnz    M0FORWARD0
  mov    a,m0sth        .load msb of step counter
  jnz    M0FORWARD0
  setb   m0done
  clr    m0speed
  mov    DPTR,#aport1   :point to motor port address
  movx   a,@DPTR        .read port value
  orl    a,#0Fh         :mask lower nibble
  movx   @DPTR,a        :send it to motor
  ret
M0FORWARD0:
  call   READ_CUT_HOME  :check limit switch
  jz     M0FORWARD2     :do not decrement step count if pressed
  mov    a,m0stl        :check for speed change point
  cjne   a,#low m0begincut,M0FORWARD3
  mov    a,m0sth
  cjne   a,#high m0begincut,M0FORWARD3
  mov    relo0h,#high m0cutrelo
  mov    relo0l,#low m0cutrelo
  setb   m0speed
M0FORWARD3:
  jnb    m0speed,M0FORWARD4
  clr    c              :Increase speed of cross slide
  mov    a,relo0l       :load lsb of step counter
  add    a,#1           :increment lsb
  mov    relo0l,a       :return the value
  mov    a,relo0h       :load msb
  addc   a,#0           :add carry flag if any
  mov    relo0h,a       :return value
M0FORWARD4:
  clr    c              .clear carry flag
  mov    a,m0stl        .load lsb of step counter
  subb   a,#1           :decrement lsb
  mov    m0stl,a        :return the value
  mov    a,m0sth        :load msb
```

```
  subb   a,#0           ;subtract carry flag if any
  mov    m0sth,a        ;return value
M0FORWARD2:
  inc    m0ph           ;increment phase counter
  mov    a,m0ph
  cjne   a,#8,M0FORWARD1
  mov    a,#0           ;reset phase counter
  mov    m0ph,a

M0FORWARD1:
  mov    DPTR,#STEPTAB12F ;point to phase table
  movc   a,@a+DPTR      ;read it
  mov    b,a            ;save phase pattern in b
  mov    DPTR,#aport1   ;point to motor port address
  movx   a,@DPTR        ;read port value
  anl    a,#0F0h        ;mask lower nibble
  orl    a,b            ;move bit pattern to lower nibble
  movx   @DPTR,a        ;send it to motor
  ret

M0REVERSE:    ;Cross slide move backward
  jb     holdm0,M0REVERSE4 ;check hold mode
  jb     m0dbo,M0REVERSE2
  call   READ_CUT_HOME  ;check limit switch
  jz     M0REVERSE2     ;go to post limit if pressed
  mov    a,m0stl        ;load lsb of step counter
  jnz    M0REVERSE0
  mov    a,m0sth        ;load msb of step counter
  jnz    M0REVERSE0
M0REVERSE2:             ;post limit movement
  call   TURN_POS_SW
  jz     M0REVERSE6
  setb   m0dbo          ;set debounce bit
  mov    relo0h,#high m0limrelo  ; *
  mov    relo0l,#low m0limrelo   ; *
  clr    c              ;clear carry flag
  mov    a,phomel       ;load lsb of step counter
  subb   a,#1           ;decrement lsb
  mov    phomel,a       ;return the value
  mov    a,phomeh       ;load msb
  subb   a,#0           ;subtract carry flag if any
  mov    phomeh,a       ;return value
  mov    a,phomel       ;load lsb of step counter
  jnz    M0REVERSE3
  mov    a,phomeh       ;load msb of step counter
  jnz    M0REVERSE3
M0REVERSE6:
  mov    m0hold,#10     ;Hold still after stop
  setb   holdm0
  ret
M0REVERSE4:
  dec    m0hold
  mov    a,m0hold
  jz     M0REVERSE5
  ret
```

```
M0REVERSE5:
  clr   m0dbo
  clr   holdm0
  setb  m0done
  mov   DPTR,#aport1   ;point to motor port address
  movx  a,@DPTR        ;read port value
  orl   a,#0Fh         ;mask lower nibble
  movx  @DPTR,a        ;send it to motor
  ret
M0REVERSE0:
  clr   c              ;clear carry flag
  mov   a,m0stl        ;load lsb of step counter
  subb  a,#1           ;decrement lsb
  mov   m0stl,a        ;return the value
  mov   a,m0sth        ;load msb
  subb  a,#0           ;subtract carry flag if any
  mov   m0sth,a        ;return value
M0REVERSE3:
  dec   m0ph           ;decrement phase counter
  mov   a,m0ph
  cjne  a,#0FFh,M0REVERSE1
  mov   a,#7           ;reset phase counter
  mov   m0ph,a
M0REVERSE1:
  mov   DPTR,#STEPTAB12F ;point to phase table
  movc  a,@a+DPTR      ;read it
  mov   b,a            ;save phase pattern in b
  mov   DPTR,#aport1   ;point to motor port address
  movx  a,@DPTR        ;read port value
  anl   a,#0F0h        ;mask lower nibble
  orl   a,b            ;move bit pattern to lower nibble
  movx  @DPTR,a        ;send it to motor
  ret

M1FORWARD:     ;Turn slide move forward
  mov   a,m1stl        ;load lsb of step counter
  jnz   M1FORWARD0
  mov   a,m1sth        ;load msb of step counter
  jnz   M1FORWARD0
  setb  m1done
  mov   DPTR,#aport1   ;point to motor port address
  movx  a,@DPTR        ;read port value
  orl   a,#0F0h        ;mask upper nibble
  movx  @DPTR,a        ;send it to motor
  ret
M1FORWARD0:
  call  READ_TURN_HOME ;check limit switch
  jz    M1FORWARD2     ;do not decrement step count if pressed
  clr   c              ;clear carry flag
  mov   a,m1stl        ;load lsb of step counter
  subb  a,#1           ;decrement lsb
  mov   m1stl,a        ;return the value
  mov   a,m1sth        ;load msb
  subb  a,#0           ;subtract carry flag if any
  mov   m1sth,a        ;return value
```

```
M1FORWARD2:
  inc    m1ph          ;increment phase counter
  mov    a,m1ph
  cjne   a,#4,M1FORWARD1
  mov    a,#0          ;reset phase counter
  mov    m1ph,a
M1FORWARD1:
  mov    DPTR,#STEPTAB2R ;point to phase table
  movc   a,@a+DPTR     ;read it
  swap   a             ;swap nibbles
  mov    b,a           ;save phase pattern in b
  mov    DPTR,#aport1  ;point to motor port address
  movx   a,@DPTR       ;read port value
  anl    a,#0Fh        ;mask upper nibble
  orl    a,b           ;move bit pattern to lower nibble
  movx   @DPTR,a       ;send it to motor
  ret

M1REVERSE:    ;Turn slide move backward
  call   READ_TURN_HOME ;check limit switch
  jz     M1REVERSE2     ;done if pressed
  mov    a,m1stl        ;load lsb of step counter
  jnz    M1REVERSE0
  mov    a,m1sth        ;load msb of step counter
  jnz    M1REVERSE0
M1REVERSE2:
  setb   m1done
  mov    DPTR,#aport1   ;point to motor port address
  movx   a,@DPTR        ;read port value
  orl    a,#0F0h        ;mask upper nibble
  movx   @DPTR,a        ;send it to motor
  ret
M1REVERSE0:
  clr    c              ;clear carry flag
  mov    a,m1stl        ;load lsb of step counter
  subb   a,#1           ;decrement lsb
  mov    m1stl,a        ;return the value
  mov    a,m1sth        ;load msb
  subb   a,#0           ;subtract carry flag if any
  mov    m1sth,a        ;return value
  dec    m1ph           ;decrement phase counter
  mov    a,m1ph
  cjne   a,#0FFh,M1REVERSE1
  mov    a,#3           ;reset phase counter
  mov    m1ph,a
M1REVERSE1:
  mov    DPTR,#STEPTAB2R ;point to phase table
  movc   a,@a+DPTR      ;read it
  swap   a              ;swap nibbles
  mov    b,a            ;save phase pattern in b
  mov    DPTR,#aport1   ;point to motor port address
  movx   a,@DPTR        ;read port value
  anl    a,#0Fh         ;mask upper nibble
  orl    a,b            ;move bit pattern to lower nibble
  movx   @DPTR,a        ;send it to motor
  ret
```

```
M2FORWARD:      :Stopper move forward
  mov    a,m2stl        :load lsb of step counter
  jnz    M2FORWARD0
  mov    a,m2sth        :load msb of step counter
  jnz    M2FORWARD0
  setb   m2done
  mov    DPTR,#aport2   :point to motor port address
  movx   a,@DPTR        :read port value
  orl    a,#0Fh         :mask lower nibble
  movx   @DPTR,a        :send it to motor
  ret
M2FORWARD0:
  call   READ_STOP_HOME :check limit switch
  jz     M2FORWARD2     :do not decrement step count if pressed
  clr    c              :clear carry flag
  mov    a,m2stl        :load lsb of step counter
  subb   a,#1           :decrement lsb
  mov    m2stl,a        :return the value
  mov    a,m2sth        :load msb
  subb   a,#0           :subtract carry flag if any
  mov    m2sth,a        :return value
M2FORWARD2:
  inc    m2ph           :increment phase counter
  mov    a,m2ph
  cjne   a,#4,M2FORWARD1
  mov    a,#0           :reset phase counter
  mov    m2ph,a
M2FORWARD1:
  mov    DPTR,#STEPTAB2R :point to phase table
  movc   a,@a+DPTR      :read it
  mov    b,a            :save phase pattern in b
  mov    DPTR,#aport2   :point to motor port address
  movx   a,@DPTR        :read port value
  anl    a,#0F0h        :mask lower nibble
  orl    a,b            :move bit pattern to lower nibble
  movx   @DPTR,a        :send it to motor
  ret

M2REVERSE:      :Stopper move reverse
  call   READ_STOP_HOME :check limit switch
  jz     M2REVERSE2     :done if pressed
  mov    a,m2stl        :load lsb of step counter
  jnz    M2REVERSE0
  mov    a,m2sth        :load msb of step counter
  jnz    M2REVERSE0
M2REVERSE2:
  setb   m2done
  mov    DPTR,#aport2   :point to motor port address
  movx   a,@DPTR        :read port value
  orl    a,#0Fh         :mask lower nibble
  movx   @DPTR,a        :send it to motor
  ret
M2REVERSE0:
  clr    c              :clear carry flag
  mov    a,m2stl        :load lsb of step counter
  subb   a,#1           :decrement lsb
  mov    m2stl,a        :return the value
  mov    a,m2sth        :load msb
  subb   a,#0           :subtract carry flag if any
```

```
    mov    m2sth,a          ;return value
    dec    m2ph             ;decrement phase counter
    mov    a,m2ph
    cjne   a,#0FFh,M2REVERSE1
    mov    a,#3             ;reset phase counter
    mov    m2ph,a
  M2REVERSE1:
    mov    DPTR,#STEPTAB2R ;point to phase table
    movc   a,@a+DPTR        ;read it
    mov    b,a              ;save phase pattern in b
    mov    DPTR,#aport2     ;point to motor port address
    movx   a,@DPTR          ;read port value
    anl    a,#0F0h          ;mask lower nibble
    orl    a,b              ;move bit pattern to lower nibble
    movx   @DPTR,a          ;send it to motor
    ret

INI8255:        ;Initiate 8255 PPI chip
    call   WAITONE          ;wait a bit
    mov    DPTR,#rport1     ;point to control register of 8255-1
    mov    a,#89h           ;A out, B out, C in
    movx   @DPTR,a          ;send program byte
    mov    DPTR,#rport2     ;point to control register of 8255-2
    mov    a,#8Bh           ;A out, B in, C in
    movx   @DPTR,a          ;send program byte
    call   WAIT             ;wait a bit
    mov    DPTR,#aport1     ;point to stepper motor port
    mov    a,#0FFh          ;Clear stepper port
    movx   @DPTR,a
    mov    DPTR,#aport2     ;point to stepper motor port
    mov    a,#0Fh           ;Clear stepper port
    movx   @DPTR,a
    ret

INIVARS:        ;Initiate system variables
    mov    m0ph,#0          ;Initiate phase counter motor 0
    mov    m1ph,#0          ;Initiate phase counter motor 1
    mov    m2ph,#0          ;Initiate phase counter motor 2
    setb   m0done           ;stop motor 0
    setb   m1done           ;stop motor 1
    setb   m2done           ;stop motor 2
    clr    holdm0           ;clear motor 0 hold bit
    clr    m0speed          ;clear motor 0 speed up bit
    clr    m0dbo            ;clear motor 0 switch debounce bit
    clr    m0rev            ;clear motor 0 reverse direction
    clr    m1rev            ;clear motor 1 reverse direction
    clr    m2rev            ;clear motor 2 reverse direction
    mov    relo0h,#0        ;clear high byte timer0 reload value
    mov    relo0l,#0        ;clear low byte timer0 reload value
    mov    relo1h,#0        ;clear high byte timer1 reload value
    mov    relo1l,#0        ;clear low byte timer1 reload value
    ret

INICLOCK:       ;Initiate timer interrupt
    clr    T0               ;Disable input to timer 0
    clr    T1               ;Disable input to timer 1
    mov    TMOD,#11h        ;Timer/counter 0&1 as timer mode 1
    setb   TR0              ;Run clock 0
    setb   TR1              ;Run clock 1
    setb   ET0              ;Enable timer 0 interrupt
```

```
    setb   ET1          ;Enable timer 1 interrupt
    setb   EA           ;Enable master interrupt switch
    ret

STEP_POWER_ON:  ;Turn on step motor power supply
    mov    DPTR,#bport1   ;point to stepper power control port
    movx   a,@DPTR        ;read port
    orl    a,#00000001b   ;set control bit
    movx   @DPTR,a        ;send control bit
    ret

STEP_POWER_OFF: ;Turn off step motor power supply
    mov    DPTR,#bport1   ;point to stepper power control port
    movx   a,@DPTR        ;read port
    anl    a,#11111110b   ;clear control bit
    movx   @DPTR,a        ;send control bit
    ret

HYD_POWER_ON:   ;Turn on hydraulic power unit
    mov    DPTR,#bport1   ;point to hydraulic power control port
    movx   a,@DPTR        ;read port
    orl    a,#00100000b   ;set control bit
    movx   @DPTR,a        ;send control bit
    ret

HYD_POWER_OFF:  ;Turn off hydraulic power unit
    mov    DPTR,#bport1   ;point to hydraulic power control port
    movx   a,@DPTR        ;read port
    anl    a,#11011111b   ;clear control bit
    movx   @DPTR,a        ;send control bit
    ret

CHUCK_MOTOR_ON: ;Turn on lathe motor
    mov    DPTR,#bport1   ;point to lathe motor control port
    movx   a,@DPTR        ;read port
    orl    a,#01000000b   ;set control bit
    movx   @DPTR,a        ;send control bit
    ret

CHUCK_MOTOR_OFF::Turn off lathe motor
    mov    DPTR,#bport1   ;point to lathe motor control port
    movx   a,@DPTR        ;read port
    anl    a,#10111111b   ;clear control bit
    movx   @DPTR,a        ;send control bit
    ret

CHUCK_OPEN:     ;Open chuck
    mov    DPTR,#bport1   ;point to chuck hydraulic control port
    movx   a,@DPTR        ;read port
    orl    a,#00010000b   ;set control bit
    movx   @DPTR,a        ;send control bit
    ret

CHUCK_CLOSE:    ;Close chuck
    mov    DPTR,#bport1   ;point to chuck hydraulic control port
    movx   a,@DPTR        ;read port
    anl    a,#11101111b   ;clear control bit
    movx   @DPTR,a        ;send control bit
    ret
```

```
LINEAR_FOR.    :Move linear feeder forward
   mov    DPTR,#bport1    ;point to linear feeder control port
   movx   a, @DPTR        ;read port
   anl    a,#11111011b    ;make sure linear back is off
   orl    a,#00001000b    ;set control bit of feeder
   movx   @DPTR,a         ;send control bit
   ret

LINEAR_BACK:   ;Move linear feeder back
   mov    DPTR,#bport1    ;point to linear feeder control port
   movx   a, @DPTR        ;read port
   anl    a,#11110111b    ;make sure linear forward is off
   orl    a,#00000100b    ;set control bit of feeder
   movx   @DPTR,a         ;send control bit
   ret
LINEAR_OFF:    ;Release air pressure in linear feeder
   mov    DPTR,#bport1    ;point to linear feeder control port
   movx   a, @DPTR        ;read port
   anl    a,#11110011b    ;linear forward & backward bits off
   movx   @DPTR,a         ;send control bits
   ret

ROTARY_GO:     ;Activate rotary feeder
   mov    DPTR,#bport1    ;point to rotary feeder control port
   movx   a, @DPTR        ;read port
   orl    a,#00000010b    ;set control bit
   movx   @DPTR,a         ;send control bit
   call   WAIT          ;delay a bit
   call   WAIT
   anl    a,#11111101b    ;clear control bit
   movx   @DPTR,a         ;send control bit
   ret

READY_LAMP_ON:  ;Turn on ready lamp
   mov    DPTR,#bport1    ;point to ready lamp control port
   movx   a, @DPTR        ;read port
   orl    a,#10000000b    ;set control bit
   movx   @DPTR,a         ;send control bit
   ret

READY_LAMP_OFF: ;Turn off ready lamp
   mov    DPTR,#bport1    ;point to ready lamp control port
   movx   a, @DPTR        ;read port
   anl    a,#01111111b    ;clear control bit
   movx   @DPTR,a         ;send control bit
   ret

HOPPER_LAMP_ON: ;Turn on hopper empty lamp
   mov    DPTR,#aport2    ;point to hopper lamp control port
   movx   a, @DPTR        ;read port
   orl    a,#00010000b    ;set control bit
   movx   @DPTR,a         ;send control bit
   ret

HOPPER_LAMP_OFF:;Turn off hopper empty lamp
   mov    DPTR,#aport2    ;point to hopper lamp control port
   movx   a, @DPTR        ;read port
   anl    a,#11101111b    ;clear control bit
   movx   @DPTR,a         ;send control bit
   ret
```

```
ALARM_LAMP_ON:  ;Turn on remote alarm
   mov    DPTR,#aport2    ;point to alarm lamp control port
   movx   a,@DPTR         ;read port
   orl    a,#01100000b    ;set control bit
   movx   @DPTR,a         ;send control bit
   ret

ALARM_LAMP_OFF: ;Turn off remote alarm
   mov    DPTR,#aport2    ;point to alarm lamp control port
   movx   a,@DPTR         ;read port
   anl    a,#10011111b    ;clear control bit
   movx   @DPTR,a         ;send control bit
   ret

READ_DTRIM:    ;Read diameter trim, result in acc
   mov    DPTR,#bport2    ;point to diameter trim port
   movx   a,@DPTR         ;read port
   orl    a,#0F0h         ;mask upper nibble
   cpl    a               ;invert bits
   ret

READ_LTRIM:    ;Read length trim, result in acc
   mov    DPTR,#bport2    ;point to length trim port
   movx   a,@DPTR         ;read port
   orl    a,#0Fh          ;mask lower nibble
   swap   a               ;swap nibbles
   cpl    a               ;invert bits
   ret

READ_TEST:     ;Read test code, result in acc
   mov    DPTR,#cport2    ;point to test switch port
   movx   a,@DPTR         ;read port
   orl    a,#0F0h         ;mask upper nibble
   cpl    a               ;invert bits
   mov    r0,a            ;Make sure a <= 9
   clr    c
   mov    a,#9
   subb   a,r0
   jbc    cy,READ_TEST0
   mov    a,r0
   ret
 READ_TEST0:
   clr    a
   ret

READ_GO:       ;Read Go switch, a=0 if pressed
   mov    DPTR,#cport2    ;point to go switch port
   movx   a,@DPTR         ;read port
   anl    a,#00010000b    ;If not pressed
   jnz    READ_GO1
   ret
 READ_GO1:
   mov    DPTR,#cport1    ;point to go switch port
   movx   a,@DPTR         ;read port
   anl    a,#10000000b    ;If not pressed
   ret

READ_STOP:     ;Read stop switch, a=0 if pressed
   mov    DPTR,#cport2    ;point to stop switch port
```

```
    movx   a, @DPTR         ;read port
    anl    a,#00100000b     ;a = 0 if pressed
    ret

READ_E_STOP:    ;Read emergency sw, a=0 if pressed
    mov    DPTR,#cport2     ;point to stop switch port
    movx   a, @DPTR         ;read port
    anl    a,#01000000b     ;a = 0 if pressed
    ret

READ_HOPPER:    ;Read hopper empty sw, a=1 if empty
    mov    DPTR,#cport1     ;point to hopper switch port
    movx   a, @DPTR         ;read port
    anl    a,#00000001b     ;a = 1 if empty
    ret

READ_LINEAR_IN: ;Read linear feed sw, a=0 if full in
    mov    DPTR,#cport1     ;point to linear in switch port
    movx   a, @DPTR         ;read port
    anl    a,#00000010b     ;a = 0 if full in
    ret

READ_STUB:      ;Read stub sw, a=0 if stub reached
    mov    DPTR,#cport1     ;point to stub switch port
    movx   a, @DPTR         ;read port
    anl    a,#00000100b     ;a = 0 if at stub
    ret

READ_TURN_HOME: ;Read turn slide home, a=0 for home
    mov    DPTR,#cport1     ;point to turn slide home switch port
    movx   a, @DPTR         ;read port
    anl    a,#00001000b     ;a = 0 if home
    ret

READ_CUT_HOME:  ;Read cut slide home, a=0 for home
    mov    DPTR,#cport1     ;point to cut slide home switch port
    movx   a, @DPTR         ;read port
    anl    a,#00010000b     ;a = 0 if home
    ret

READ_STOP_HOME: ;Read stopper home, a=0 for home
    mov    DPTR,#cport1     ;point to stop slide home switch port
    movx   a, @DPTR         ;read port
    anl    a,#00100000b     ;a = 0 if home
    ret

TURN_POS_SW: ;Read turn position sw, a=0 for pressed
    mov    DPTR,#cport1     ;point to turn position switch port
    movx   a, @DPTR         ;read port
    anl    a,#01000000b     ;a = 0 if home
    ret

WAITSW:         ;Wait for front panel switch
    call   READ_GO
    jz     WAITSW_DONE
    call   READ_STOP
    jnz    WAITSW
    setb   stopped
  WAITSW_DONE:
    ret
```

```
WAITGO:       ;Wait for GO switch to be pressed
  setb  mes1
  mov   a,#5
  call  DISPLAY
 WAITGO1:
  call  READ_GO       ;Read go switch
  jnz   WAITGO1       ;Wait around
  setb  mes1
  mov   a,#15
  call  DISPLAY
  ret

WAITM0:       ;Wait for motor 0 done
  jnb   m0done,WAITM0  ;wait until done
  ret

WAITM1:       ;Wait for motor 1 done
  jnb   m1done,WAITM1  ;wait until done
  ret

WAITM2:       ;Wait for motor 2 done
  jnb   m2done,WAITM2  ;wait until done
  ret

WAITONE:      ;Time delay one second
  mov   r3,#10
 WAITONE1:
  call  WAIT
  djnz  r3,WAITONE1
  ret

WAIT:         ;Time delay subroutine 100 millisec
  mov   r1,#196
 WAIT1:
  mov   r2,#0FFh
 WAIT2:
  djnz  r2,WAIT2
  djnz  r1,WAIT1
  ret

CHECK_CROSS_R:  ;Check cross slide reversal for error
  mov   a,m0sth       ;load step counter MSB
  jnz   CHECK_CROSS_R1  ;error if not zero
  mov   a,m0stl       ;load step counter LSB
  jz    CHECK_CROSS_R2  ;If zero, limit switch not reached
  clr   c             ;clear carry flag
  subb  a,#200        ;subtract 200
  jnb   cy,CHECK_CROSS_R1;error if more than 200
  mov   error,#0
  ret

 CHECK_CROSS_R1:      ;Slide reach limit too soon
  mov   error,#2
  jmp   ALL_STOP
 CHECK_CROSS_R2:      ;Limit not reached
  mov   error,#3
  jmp   ALL_STOP

CHECK_CROSS_F:  ;Check cross slide forward for error
```

```
    call   READ_CUT_HOME
    jz     CHECK_CROSS_F1
    mov    error,#0
    ret
  CHECK_CROSS_F1:
    mov    error,#1
    jmp    ALL_STOP

 CHECK_CROSS_H:  ;Check cross slide for proper homeing
    call   WAIT            ;Limit switch debounce
    call   READ_CUT_HOME
    jnz    CHECK_CROSS_H1
    mov    error,#0
    ret
  CHECK_CROSS_H1:
    mov    error,#3
    jmp    ALL_STOP

 CHECK_TURN_R:   ;Check turn slide reversal for error
```

Appendix B. Firmware Program Listing (continue).

```
    mov    a,m1sth         ;load step counter MSB
    jnz    CHECK_TURN_R1   ;error if not zero
    mov    a,m1stl         ;load step counter LSB
    jz     CHECK_TURN_R2   ;If zero, limit sw not reached
    clr    c               ;clear carry flag
    subb   a,#200          ;subtract 200
    jnb    cy,CHECK_TURN_R1;error if more than 200
    mov    error,#0
    ret
  CHECK_TURN_R1:           ;Slide reach limit too soon
    mov    error,#5
    jmp    ALL_STOP
  CHECK_TURN_R2:           ;Limit not reached
    mov    error,#6
    jmp    ALL_STOP

 CHECK_TURN_F:  ;Check turn slide forward for error
    call   READ_TURN_HOME
    jz     CHECK_TURN_F1
    mov    error,#0
    ret
  CHECK_TURN_F1:
    mov    error,#4
    jmp    ALL_STOP

 CHECK_TURN_H:  ;Check turn slide for proper homeing
    call   WAIT            ;Limit switch debounce
    call   READ_TURN_HOME
    jnz    CHECK_TURN_H1
    mov    error,#0
    ret
  CHECK_TURN_H1:
    mov    error,#6
    jmp    ALL_STOP
```

```
CHECK_STOP_R:  :Check stopper slide reversal for error
   mov    a,m2sth        :load step counter MSB
   jnz    CHECK_STOP_R1  :error if not zero
   mov    a,m2stl        :load step counter LSB
   jz     CHECK_STOP_R2  :If zero, limit sw not reached
   clr    c              :clear carry flag
   subb   a,#200         :subtract 200
   jnb    cy,CHECK_STOP_R1:error if more than 200
   mov    error,#0
   ret
 CHECK_STOP_R1:          :Slide reach limit too soon
   mov    error,#8
   jmp    ALL_STOP
 CHECK_STOP_R2:          :Limit not reached
   mov    error,#9
   jmp    ALL_STOP

CHECK_STOP_F:  :Check stop slide forward for error
   call   READ_STOP_HOME
   jz     CHECK_STOP_F1
   mov    error,#0
   ret
 CHECK_STOP_F1:
   mov    error,#7
   jmp    ALL_STOP

CHECK_STOP_H:  :Check stopper for proper homing
   call   WAIT           :Limit switch debounce
   call   READ_STOP_HOME
   jnz    CHECK_STOP_H1
   mov    error,#0
   ret
 CHECK_STOP_H1:
   mov    error,#9
   jmp    ALL_STOP

CHECK_LINEAR_B: :Check linear feeder full back position
   mov    a,error
   jnz    ALL_STOP
   ret

HOPPER_EMPTY:   :Hopper empty condition
   call   HOPPER_LAMP_ON
   mov    a,error
   call   DISPLAY
   call   CHUCK_MOTOR_OFF :Turn off lathe motor
   call   STEP_POWER_OFF  :Turn off step motor power
   call   LINEAR_F_BACK   :Pull back linear feeder
   call   LINEAR_OFF      :Unload linear feeder
   call   CHUCK_OPEN      :Open chuck
 HOPPER_EMPTY_1:
   call   HOPPER_LAMP_OFF :Flash hopper lamp
   call   WAIT
   call   WAIT
   call   HOPPER_LAMP_ON
   call   WAIT
   call   WAIT
   call   WAIT
   call   WAIT
   jmp    HOPPER_EMPTY_1  :Loop forever, wait for reset
```

```
NORMAL_STOP:
   clr    stopped
   call   WAITGO
   ret

ALL_STOP:      :Fatal error, stop all system and flash error
   call   ALARM_LAMP_ON
:  call   READ_TEST      :Testing?
:  jz     ALL_STOP_2
:  ret                   :Return if testing

 ALL_STOP_2:
   call   CHUCK_MOTOR_OFF :Turn off lathe motor
   clr    EA
   mov    a,error
   call   DISPLAY
   call   STEP_POWER_OFF  :Turn off step motor power
   call   READY_LAMP_OFF  :Turn off ready lamp
   call   LINEAR_F_BACK   :Pull back linear feeder
   call   LINEAR_OFF      :Unload linear feeder
   call   CHUCK_OPEN      :Open chuck
 ALL_STOP_1:
   call   ALARM_LAMP_OFF  :Flash alarm lamp
   call   WAIT
   call   WAIT
   call   ALARM_LAMP_ON
   call   WAIT
   call   WAIT
   call   WAIT
   call   WAIT
   jmp    ALL_STOP_1      :Loop forever, wait for reset

EMERGENCY_STOP:
   call   ALARM_LAMP_ON
   clr    EA             :Disable all interrupt
   mov    a,#14          :Emergency stop massage
   call   DISPLAY
   call   STEP_POWER_OFF
   call   CHUCK_MOTOR_OFF
   call   WAITONE
   call   WAITONE
   call   CHUCK_OPEN
   call   WAITONE
   call   LINEAR_BACK
   call   WAITONE
   call   WAITONE
   call   LINEAR_OFF
 EMERGENCY_STOP1:
   call   ALARM_LAMP_OFF  :Flash alarm lamp
   call   WAIT
   call   WAIT
   call   ALARM_LAMP_ON
   call   WAIT
   call   WAIT
   call   WAIT
   call   WAIT
   jmp    EMERGENCY_STOP1 :Loop forever, wait for reset

STEPTAB2F:     :2 phase drive forward bit pattern
   db  00001100b
```

```
    db 00001001b
    db 00000011b
    db 00000110b

 STEPTAB2R:     ;2 phase drive reverse bit pattern
    db 00001100b
    db 00000110b
    db 00000011b
    db 00001001b

 STEPTAB12R.    ;1-2 phase drive reverse bit pattern
    db 00001100b
    db 00001110b
    db 00000110b
    db 00000111b
    db 00000011b
    db 00001011b
    db 00001001b
    db 00001101b

 STEPTAB12F:    ;1-2 phase drive forward bit pattern
    db 00001100b
    db 00001101b
    db 00001001b
    db 00001011b
    db 00000011b
    db 00000111b
    db 00000110b
    db 00001110b

;LCD Driver written by Somchai Pattana
;call  INIT_LCD to initialize LCD
;call  DISPLAY with massage number in a

delay:                  ;delay 15 millisec
        mov  r1,#15
 WAITLCD1:                  ;delay 1 millisec
        mov  r2,#226        ;[(12 x (1/11.0592 MHz.)] x [(4x226)+2]
 WAITLCD2:
        nop             ;= 983.0729 microsec
        nop
        djnz r2,WAITLCD2
        nop
        nop
        djnz r1,WAITLCD1    ;Loop 15 time
        ret

LCD_DONE:
        mov   r1,#210       ;1700 u sec max
 LCD_DONE1:
        mov  DPTR,#rd_busy  ;Point to LCD busy instruction
        movx a,@DPTR        ;Read port
        jnb  ACC.7,LCD_DONE2 ;Wait until bit 7 is zero
        djnz r1,LCD_DONE1
 LCD_DONE2:
        ret

ClrScr:
        call LCD_DONE       ;Wait until LCD done
        mov  DPTR,#wr_inst  ;Point to LCD write instruction
```

```
        mov  a,#01h         ;Set clear data to A
        movx @DPTR,a        ;Send clear screen data
        ret
Init_LCD:
        call delay
        mov  DPTR,#wr_inst  ;Point to LCD write instruction
        mov  a,#30h         ;8 bit interface set
        movx @DPTR,a        ;Send function set
        call delay
        mov  DPTR,#wr_inst  ;Point to LCD write instruction
        mov  a,#30h         ;8 bit interface set
        movx @DPTR,a        ;Send function set
        call delay
        mov  DPTR,#wr_inst  ;Point to LCD write instruction
        mov  a,#30h         ;8 bit interface set
        movx @DPTR,a        ;Send function set
        call LCD_DONE       ;Wait until LCD done
        mov  DPTR,#wr_inst  ;Point to LCD write instruction
        mov  a,#38h         ;8 bits data, 16 columns, 5x7 dots
        movx @DPTR,a        ;Send function set
        call LCD_DONE       ;Wait until LCD done
        mov  DPTR,#wr_inst  ;Point to LCD write instruction
        mov  a,#08h         ;Display off
        movx @DPTR,a        ;Send display control

        call ClrScr         ;Clear all display

        call LCD_DONE       ;Wait until LCD done
        mov  DPTR,#wr_inst  ;Point to LCD write instruction
        mov  a,#0Eh         ;Display on, cursor on, blink off
        movx @DPTR,a        ;Send display control
        ret

Display:
        mov  r6,a           ;Move data to R6
        call ClrScr         ;Clear display
        mov  a,r6           ;Move data to ACC
        jnz  SHOW           ;Return if code 0
        ret
 SHOW:
        dec  a              ;Match data to table
        swap a              ;Swap nibbles
        mov  r6,a           ;Move data to R6
        mov  r5,#80h        ;First DD RAM address
        mov  r7,#10h        ;Load number of character
 NEXT_CHR:
        call LCD_DONE       ;Wait until LCD done
        jb   mes1,NEXT_CHR1
        mov  DPTR,#MESSAGE  ;Point to string table
        jmp  NEXT_CHR2
      NEXT_CHR1:
        mov  DPTR,#MESSAGE1
      NEXT_CHR2:
        mov  a,r6           ;Move string position to ACC
        move a,@a+DPTR      ;Read ASCII from table
        mov  DPTR,#wr_data  ;Point to LCD write data port
        movx @DPTR,a        ;Send data
        inc  r5             ;Increase DD RAM address
        cjne r5,#88h,NEXT   ;Check last 8 digit
        mov  r5,#0C0h       ;Set the 8th digit DD RAM address
```

```
        call LCD_DONE       :Wait until LCD done
        mov  DPTR,#wr_inst  :Point to LCD write instruction
        mov  a,r5           :Set DD RAM address
        movx @DPTR,a        :Send function set
NEXT:
        inc  r6             :Increase string position
        djnz r7,NEXT_CHR    :Check last character
        clr  mes1
        ret

MESSAGE:                    :Table of error message
        db  '01 CROSS LIMIT  '
        db  '02 CROSS STROKE '
        db  '03 CROSS NOT RET'
        db  '04 TURN LIMIT   '
        db  '05 TURN STROKE  '
        db  '06 TURN NOT RET '
        db  '07 STOPPER LIMIT'
        db  '08 STOP STROKE  '
        db  '09 STOP NOT RET '
        db  '10 CHUCK NOT CLR'
        db  '11 LINEAR NOT RE'
        db  '12 ROTARY PROB. '
        db  '13 HOPPER EMPTY '
        db  '14 EMERGENCY ST '
        db  '15 STOPPED      '

MESSAGE1:                   :Table of display message
        db  'INITIALIZING    '
        db  'FEEDING NEW PART'
        db  'MACHINING SLIDE '
        db  'EJECTING STUB   '
        db  'READY--PRESS GO '
        db  '6               '
        db  '7               '
        db  '8               '
        db  '9               '
        db  '10              '
        db  '11              '
        db  '12              '
        db  '13              '
        db  '16 COUNT ERROR  '
        db  '                '

    end
```

Appendix C. Specifications.

The single purpose lathe is a manual lathe retrofitted for automated machining of the first stage slider production. The cycle time is 40 seconds per piece and the tool change is 120 pieces for turning and 200 pieces for cut off. The lathe being used is YAM-550 engine lathe with the following specifications:

Cross slide
- pitch 4 mm/rev.
- forward stroke 15 mm.
- reverse stroke 16 mm.
- turn stroke 3.35 mm.
- motor resolution 400 steps/rev.

Top slide
- pitch 3 mm/rev.
- forward stroke 30 mm.
- reverse stroke 31 mm.
- ready for facing 27 mm.
- turn for facing 3 mm.
- motor resolution 200 steps/rev.

Stopper slide
- pitch 4 mm/rev.
- normal stroke 10 mm.
- return from normal 12 mm.
- facing stroke 37 mm.
- return from facing 39 mm.
- clear chuck stroke 40 mm.
- return from clear 42 mm.
- motor resolution 200 steps/rev.

Appendix D. Cost Breakdown.

Description		Part No.	Vender	Quan.	@	Total	Delivery	
CONTROL UNIT								
Front control panel		custom	CCR		1	1,500	1,500	7 days
Microprocessor board		ANT-32	Sila	1	1,590	1,590	3 days	
EPROM	27128		CCR	1	200	200	in stock	
Stepper motor driver		EX-STEPM	Sila		3	650	1,950	3 days
Relay output board		EX-RELAY	Sila		3	650	1,950	3 days
Input board		custom	CCR		1	300	300	3 days
Power supply	5V3A,24V1A	POW-53	Sila		1	950	950	3 days
Power supply		12V35A	CCR	1	2,500	2,500		
Resistor pack		custom	CCR		3	300	900	3 days
Circuit box		#0	CCR		1	400	400	in stock
Circuit box		#1	CCR		1	450	450	in stock
Relay 24V Koeke	MB2		CCR	5	150	750	7 days	
Magnetic switch 10A		H10C	CCR		1	550	550	in stock
Pilot lamp			CCR		2	150	300	in stock
Misc hardware & assembly	custom		CCR	1	2,000	2,000	7 days	
DRIVE SYSTEM								
Step motor Step-Syn		103-815-2	Flexible	2	5,625	11,250	30 days	
Step motor Step-Syn		103-8920-26	Flexible	1	15,000	15,000	30 days	
Air cylinder 40X1200mm		SUE		1	4,500	4,500	30 days	
Solenoid valve		183-4E2-13	Flexible	1	4,200	4,200	30 days	
Muffler	KM-1		Flexible	1	402	402	30 days	
Pressure regulator		RTL6-02	Flexible	1	1,060	1,060	30 days	
Speed control		ESC6-PT 1/4-0	Flexible	1	290	290	30 days	
Hydraulic chuck system	N-206		Tzu Sheng	1	121,208	121,208	30 days	
Rotary feeder & hopper	custom		Choorat	1	10,000	10,000	15 days	
LATHE								
Engine lathe		YAM-550	SUE		1		0	in stock
Cut off tool post	custom		Choorat	1	2,000	2,000	7 days	
Cut off tool holder	ENTARN161H33		SUE		1		0	in stock
Cut off tool insert	CUTTER SLIDE		SUE	1		0	in stock	
Turn tool post		custom	Choorat	1	2,000	2,000	7 days	
Turn tool holder	CF3-3		SUE	1		0	in stock	
Turn tool insert	TNPA332		SUE	1		0	in stock	
Stopper tip		custom	Choorat	1	400	400	3 days	
Switch actuator	custom		Choorat	1	1,000	1,000	3 days	
Lathe bore adaptor		custom	Choorat	1	1,000	1,000	3 days	
Limit switch and bracket	custom		Choorat	3	700	2,100	7 days	
Motor mount and coupling	custom		Choorat	3	1,000	3,000	7 days	
Feed system bracket		custom	Choorat	1	3,000	3,000	7 days	
User tool kit		custom	CCR	1	10,000	10,000	7 days	
TOTAL						**208,700**		

Appendix E. Vender List.

CCR

หจก. คอมพิวเตอร์ คอนโทร โรโบติค มาโนช ประพันธ์ ผู้จัดการ
8/25 หมู่ 3 ถนนซุปเปอร์ไฮเวย์ ตำบลช้างเผือก อ. เมือง เชียงใหม่ 50300
โทร/แฟกซ์ (053) 217053 มือถือ (01) 9605787

Choorat

ชูรัตน์ ธารารักษ์ สถาบันเทคโนโลยีราชมงคล วิทยาเขตภาคพายัพ
คณะวิชาเครื่องกล แผนกวิชาช่างกลเกษตร
128 ถนนห้วยแก้ว อ. เมือง เชียงใหม่ 50000
โทร. (053) 213112, 213134, 221576 ต่อ 467

Flexible

Flexible Automation System Co. Ltd. สุนทร แซ่ลี้ Sales Engineer
263/6 หมู่ 4 ถนนลำพูน - ดอยศรีวิชัย ต. ป่าสัก อ. เมือง ลำพูน 51000
โทร. (053) 537518-9 แฟกซ์ (053) 537517

SUE

SANYO Universal Electric Public Company Ltd. Compressor Factory
องอาจ วีรชาติยานุกูล Engineer QI Section (IE) 19 สุขุมวิท ซอย 103 กรุงเทพ 10260
โทร. (02) 3980120 ต่อ 388 แฟกซ์ (02) 3980120 ต่อ 261

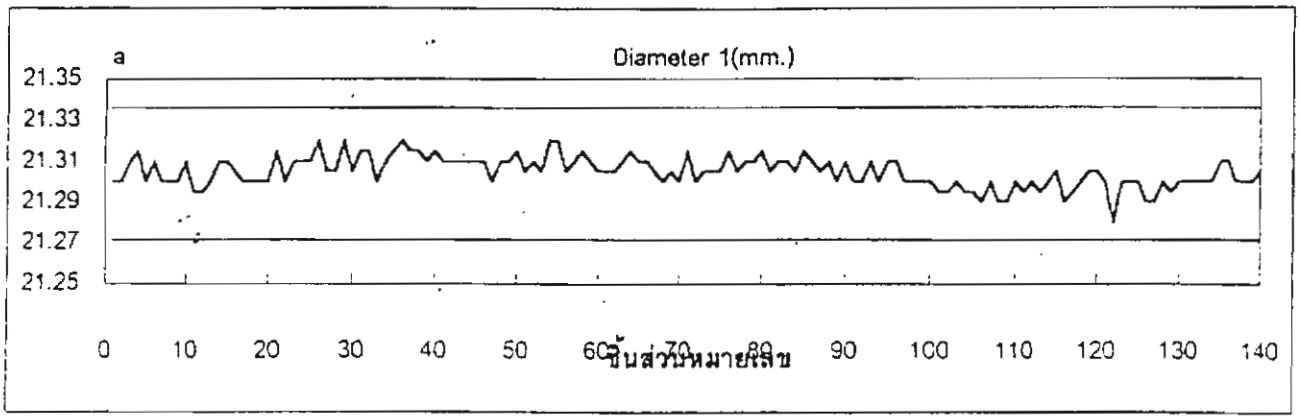
Sila

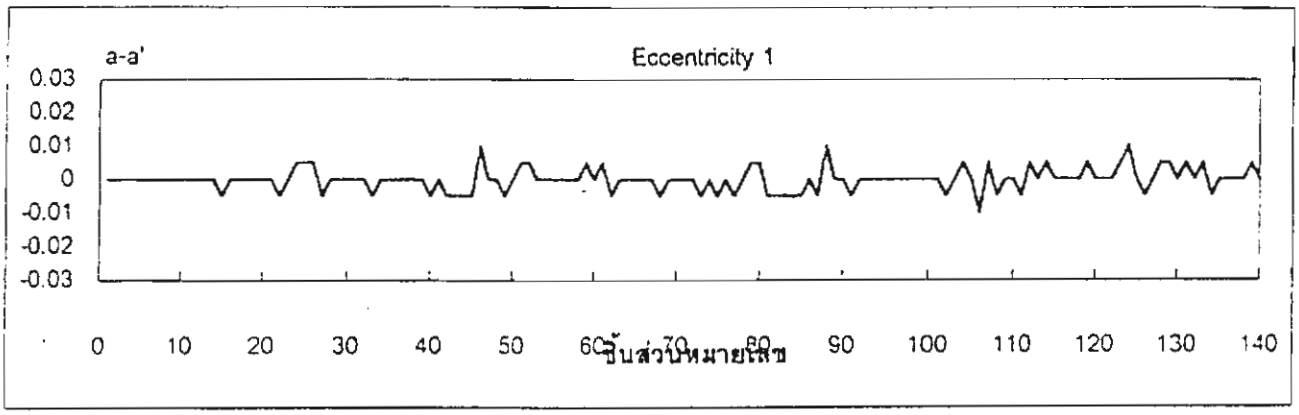
บริษัท ศิลารีเสิร์ช จำกัด
1108/41 ศูนย์การค้าพระโขนง ถนนสุขุมวิท กรุงเทพ 10110
โทร. (02) 7122850-2 แฟกซ์ (02) 3811447

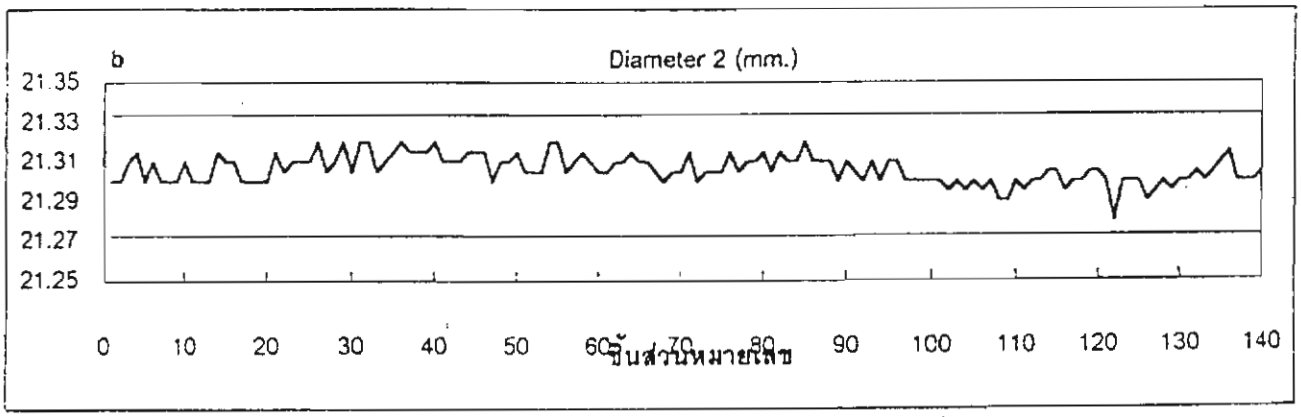
Tzu Cheng

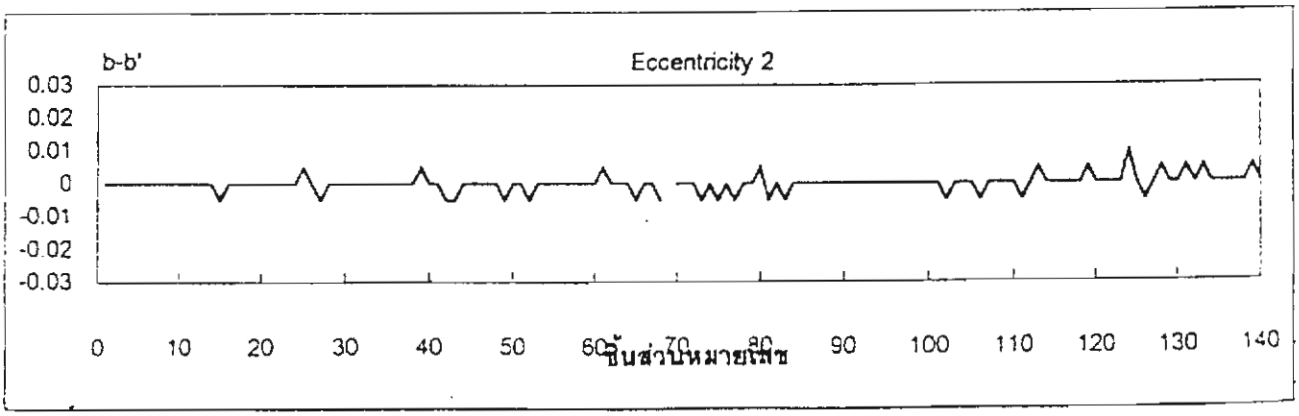
บริษัท จือเซิน (ไทยแลนด์) จำกัด ศักดา พงศาสนองกุล
7/194 หมู่ 14 บางนา-ตราด กม. 6.5 บางแก้ว-บางพลี สมุทรปราการ 10540
โทร. (02) 3163960-1, 3160873, 01-3535038 แฟกซ์ (02) 3160873

Appendix F. Test Result Run 1.

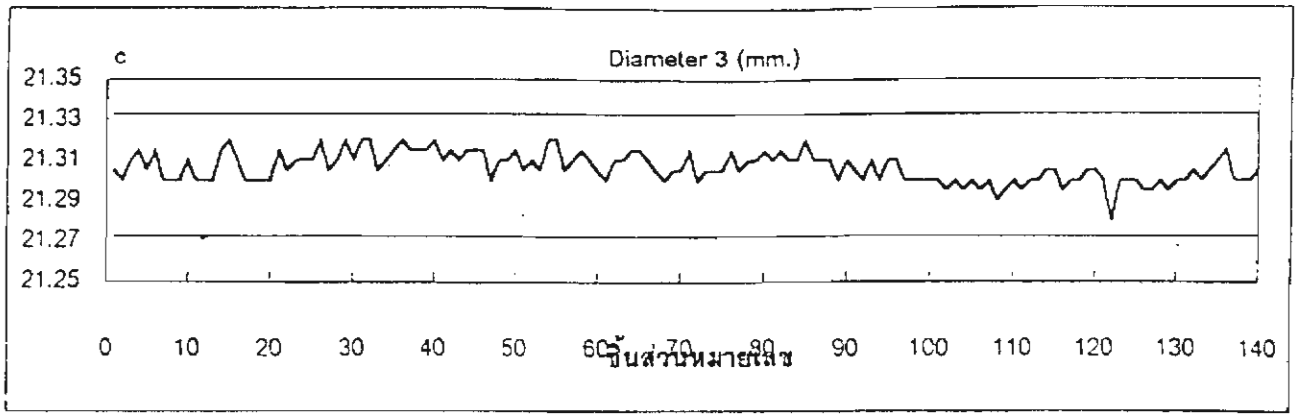


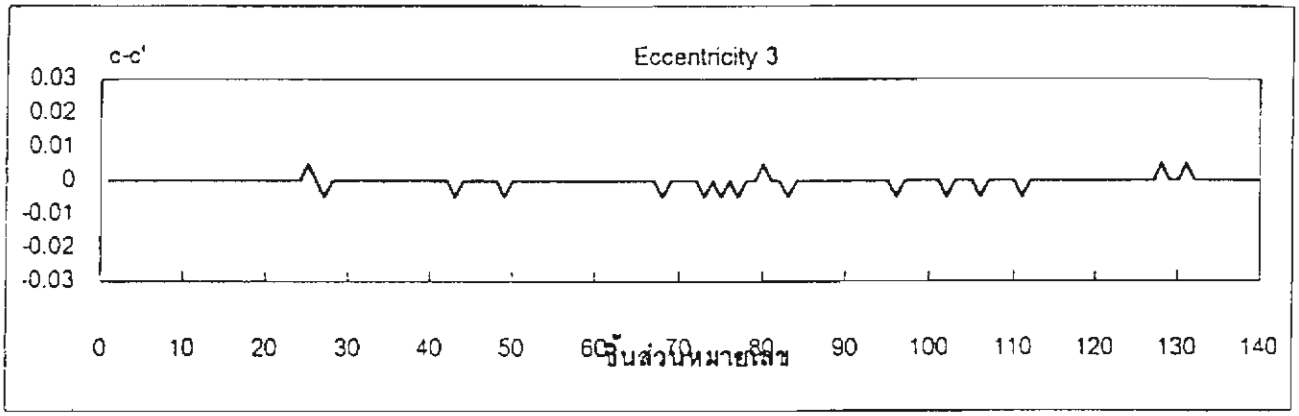


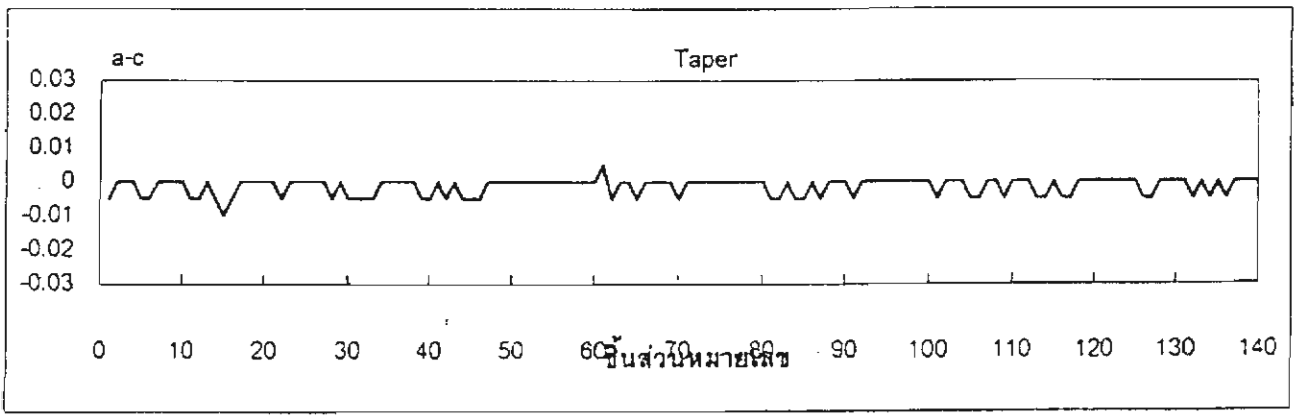


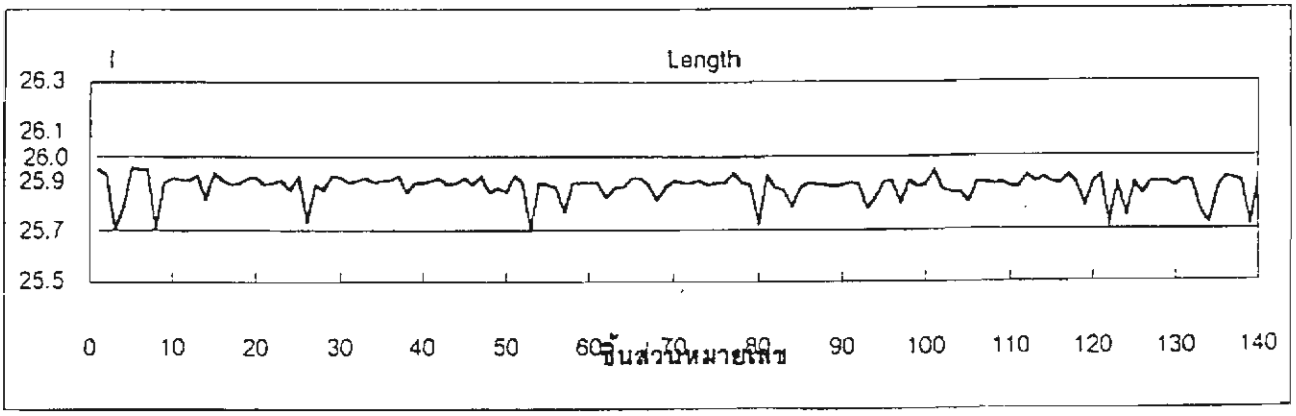


Appendix F. Test Result Run 1 (continue).

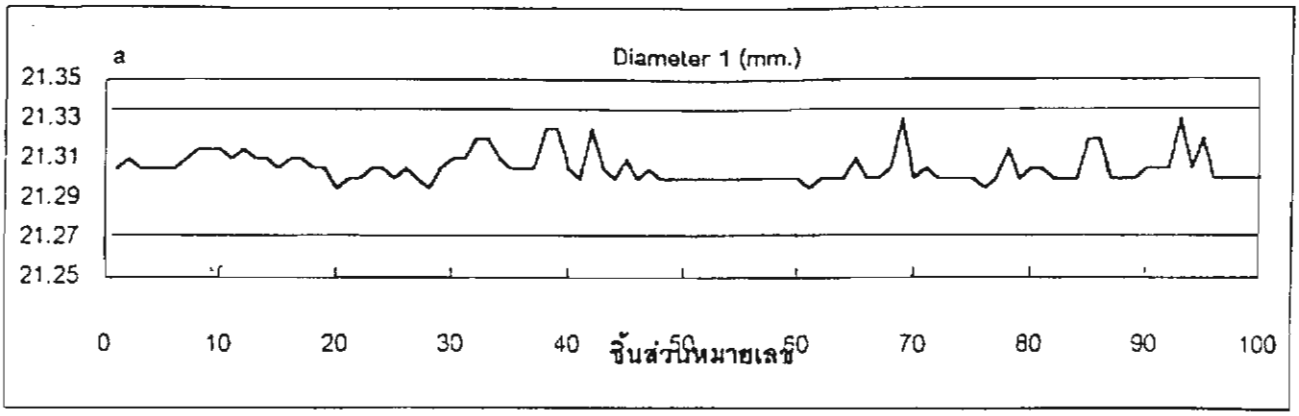


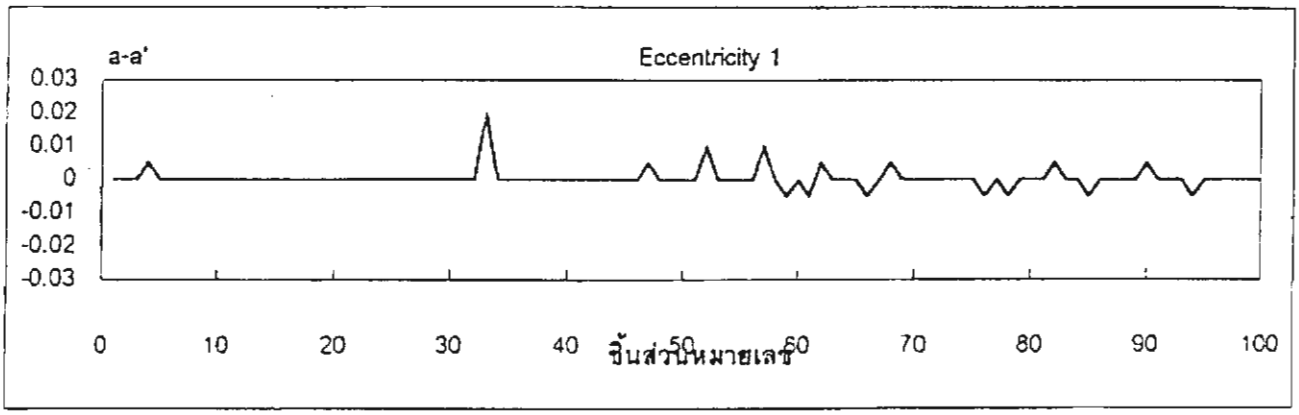


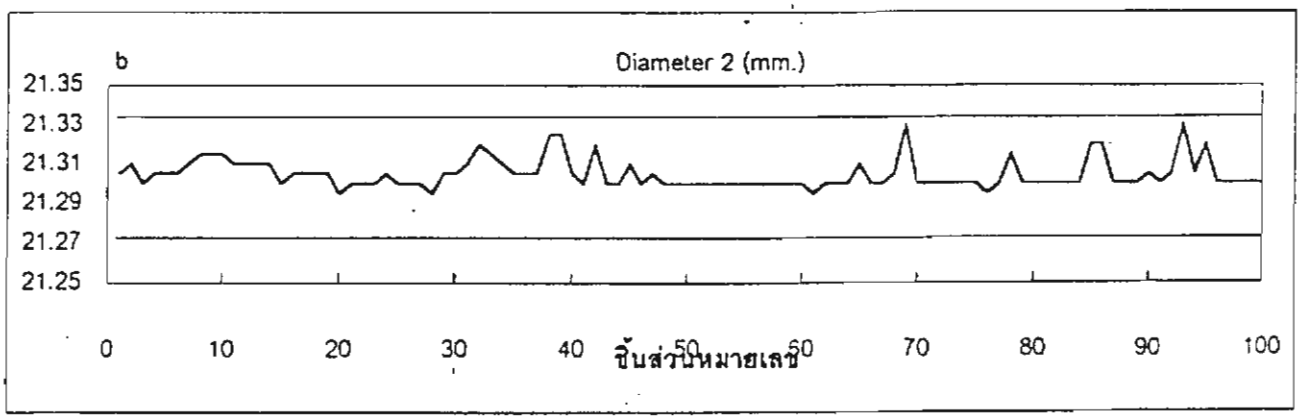


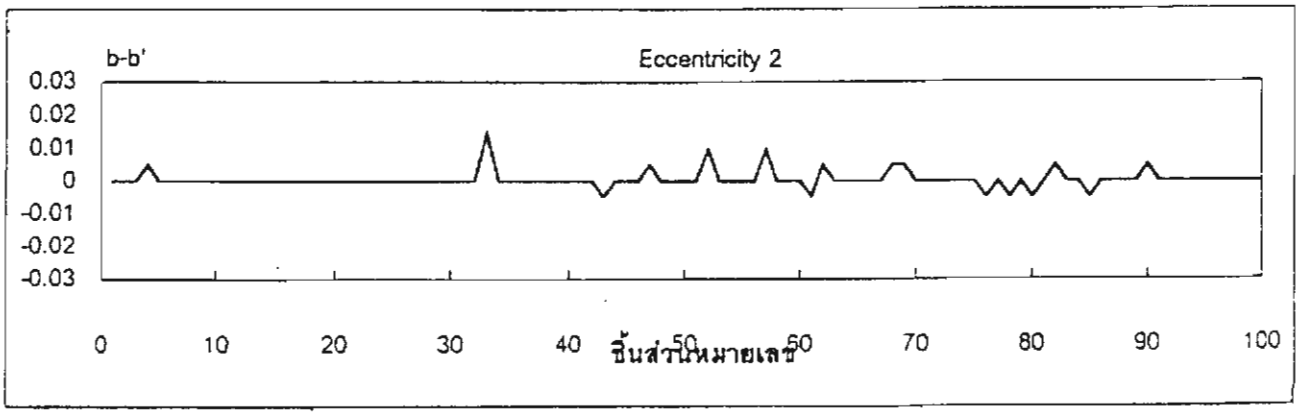


Appendix F. Test Result Run 2.

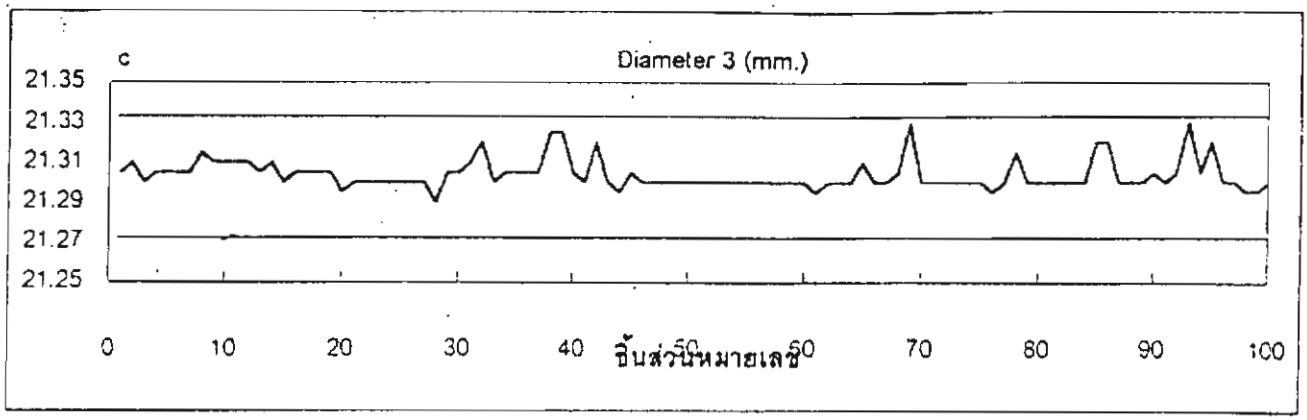


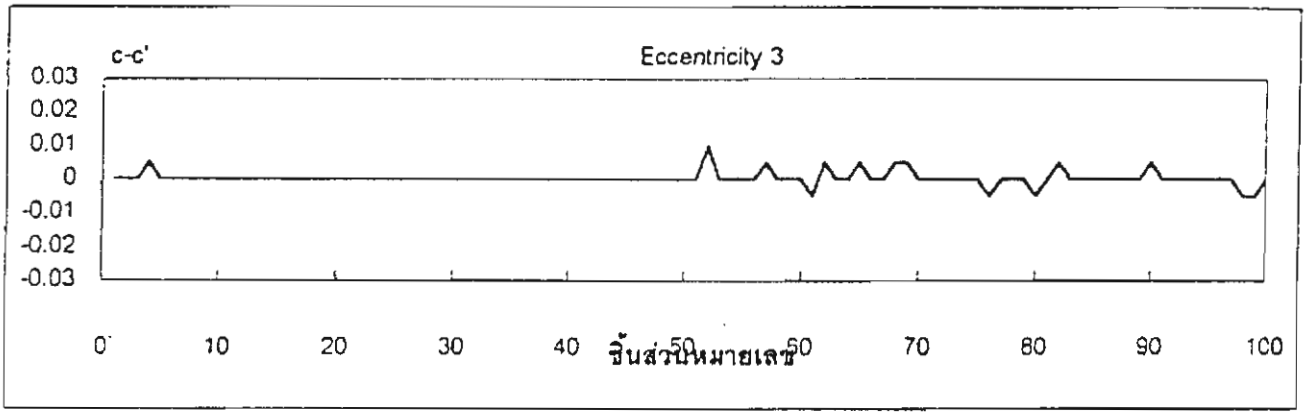


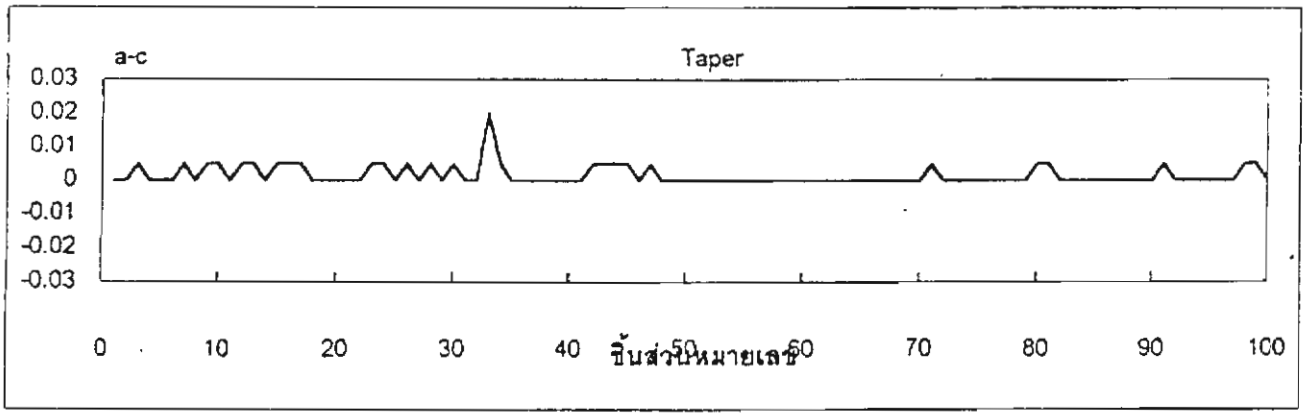


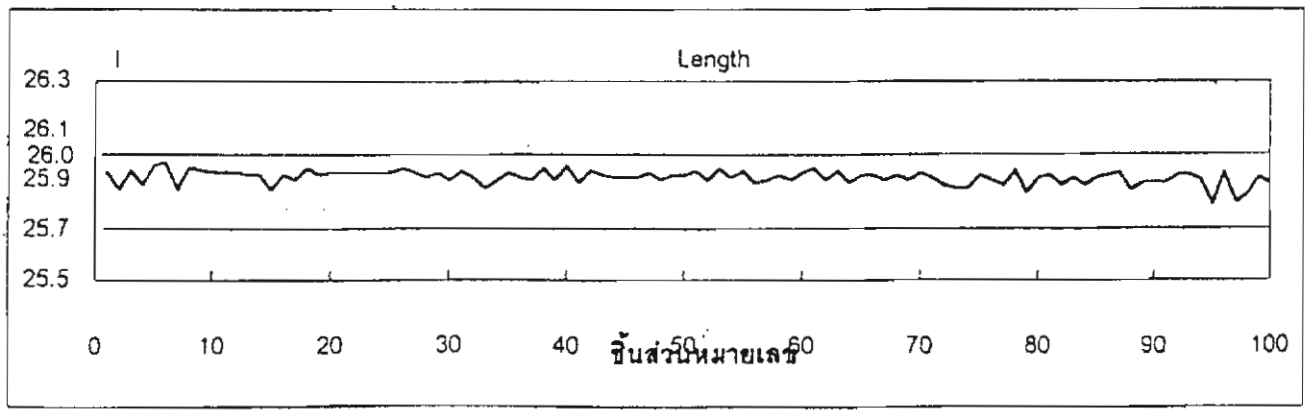


Appendix F. Test Result Run 2 (continue).

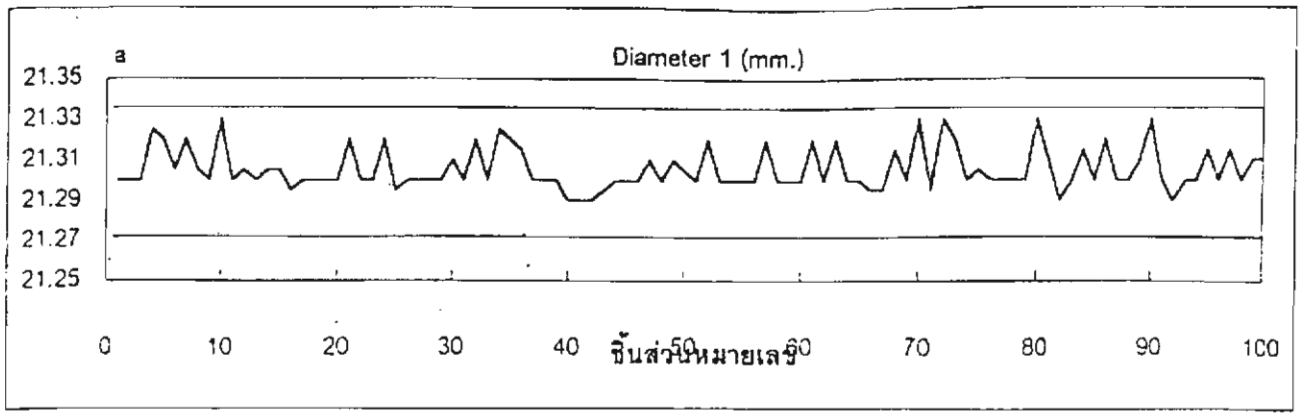


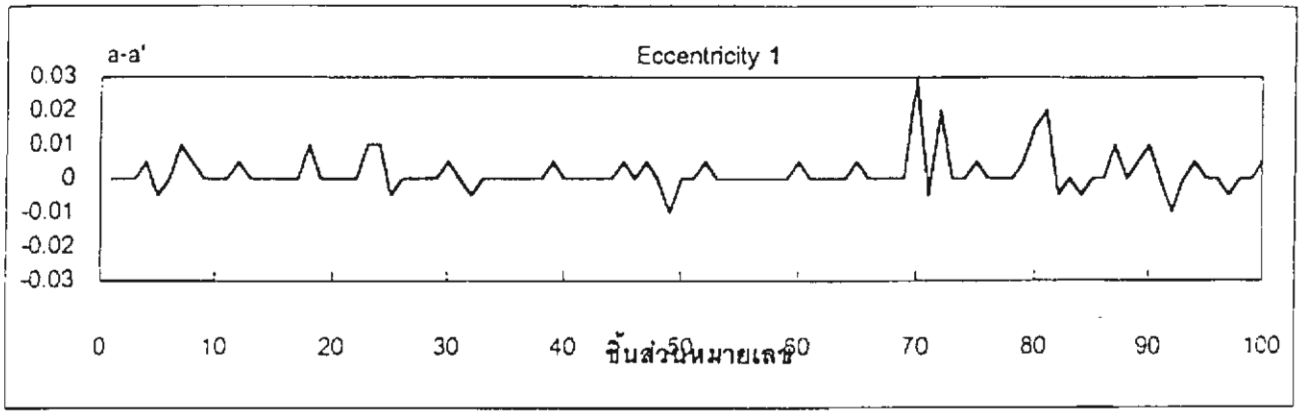


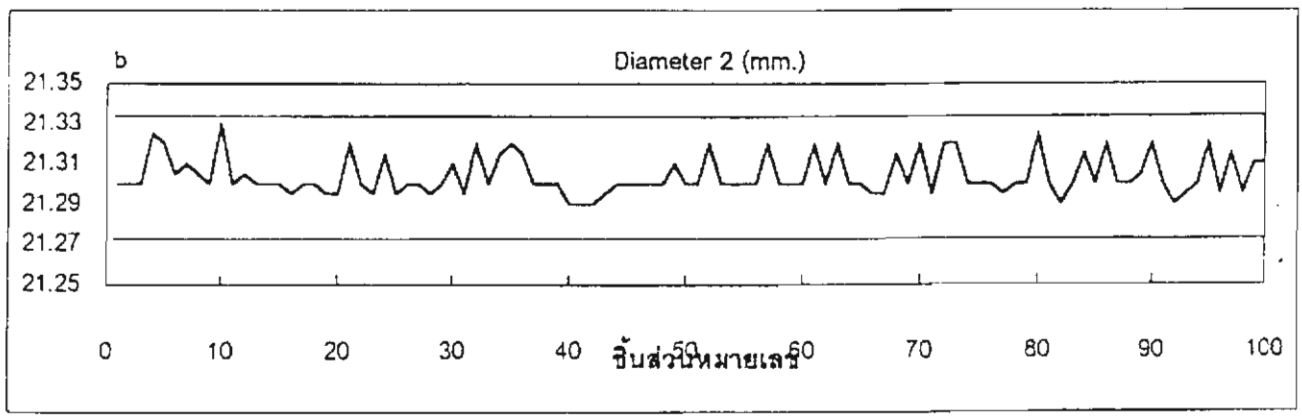


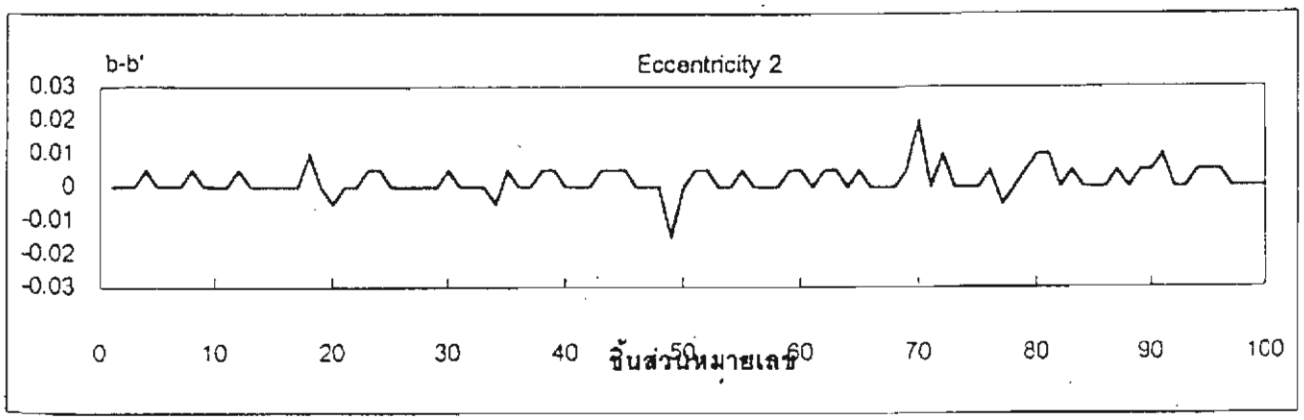


Appendix F. Test Result Run 3.

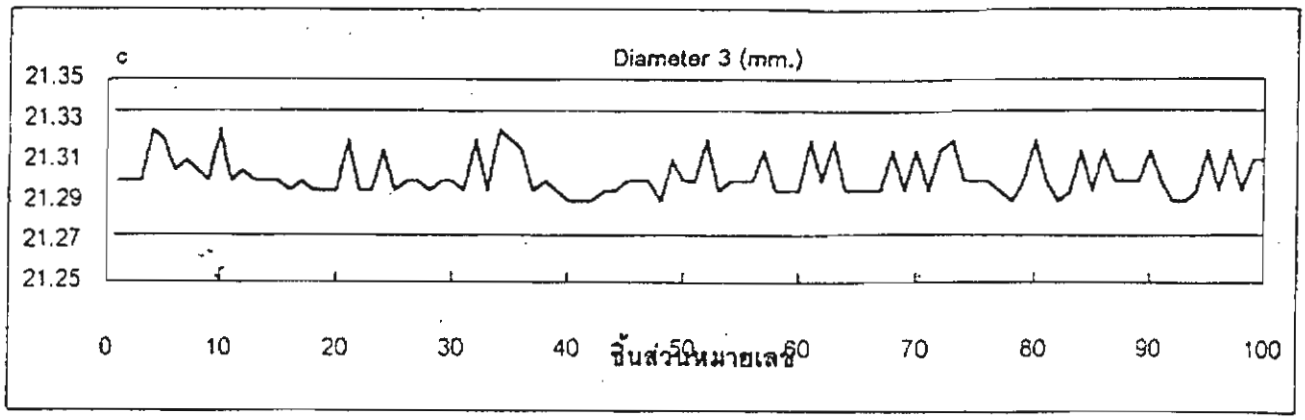


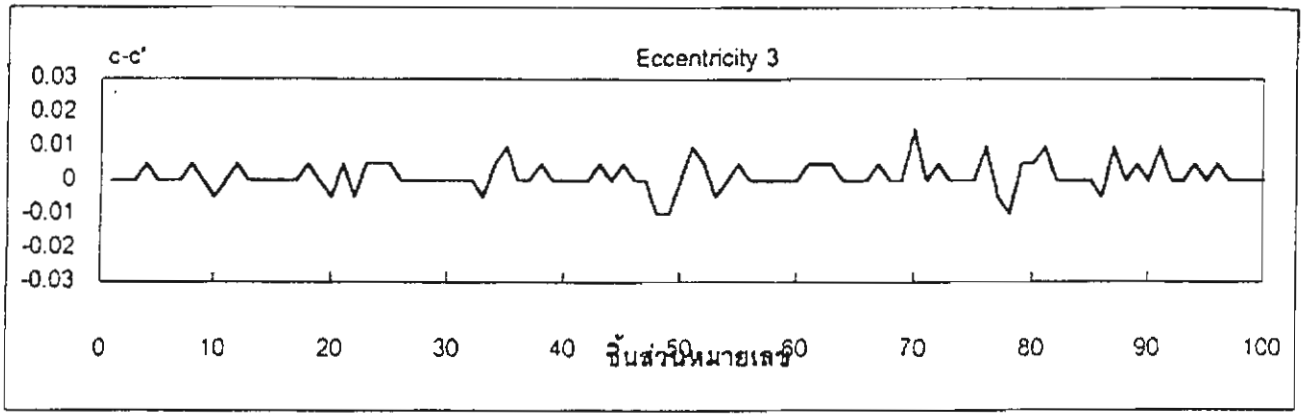


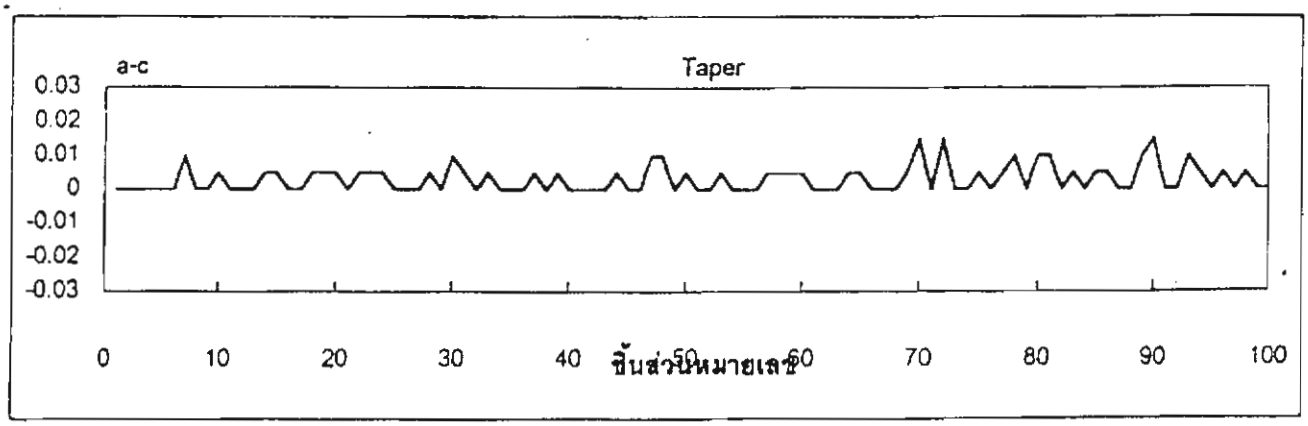


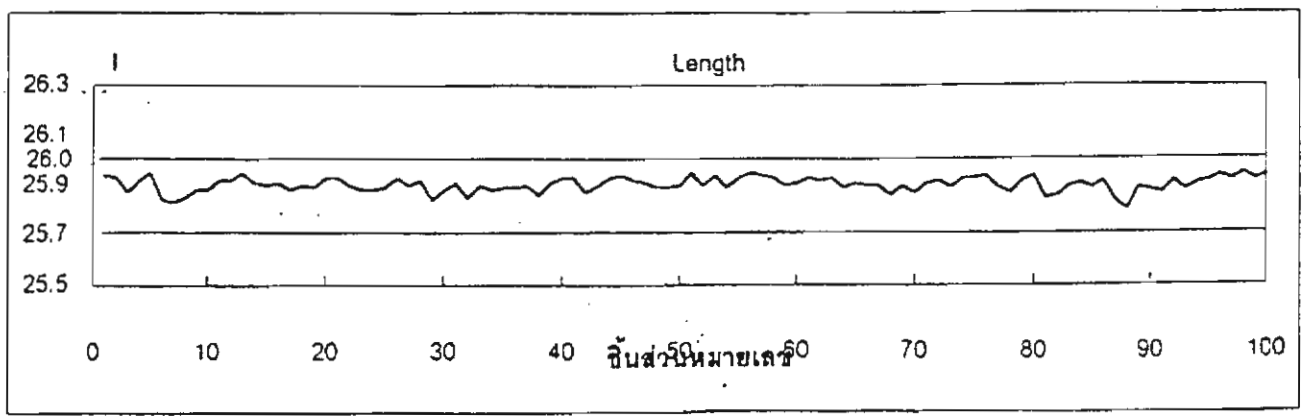


Appendix F. Test Result Run 3 (continue).

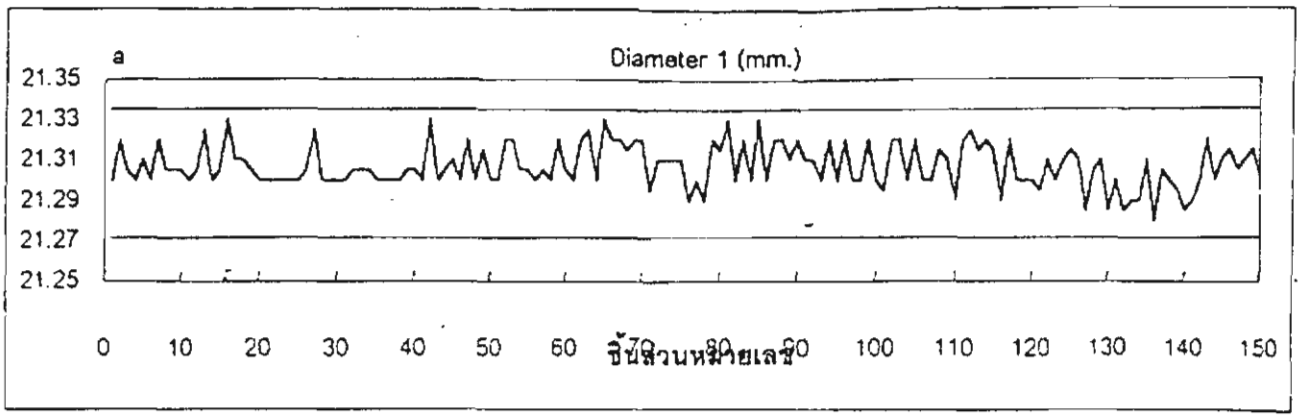


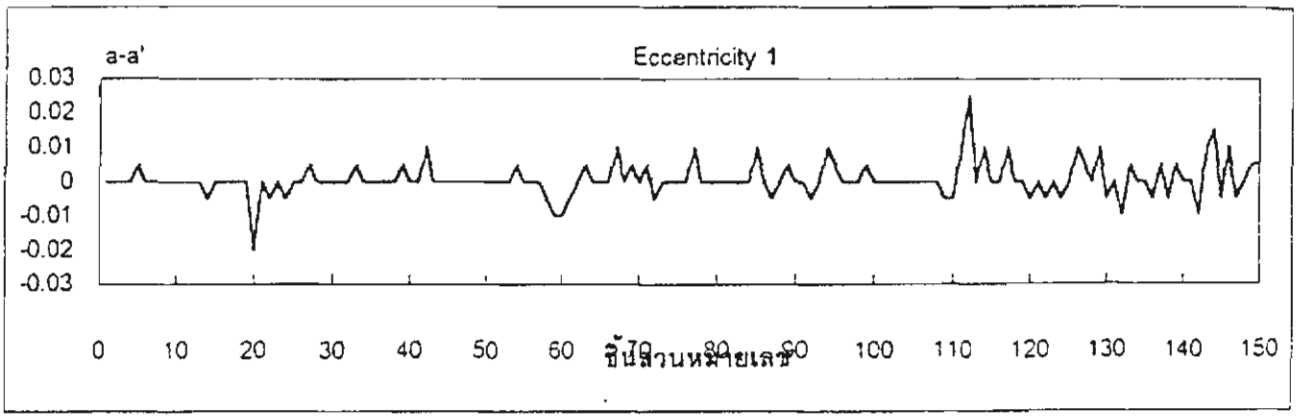


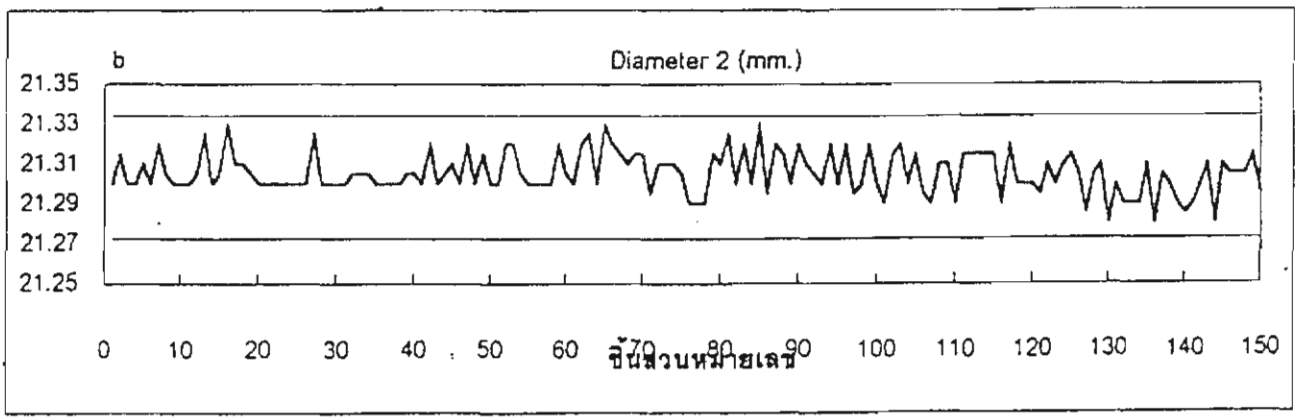


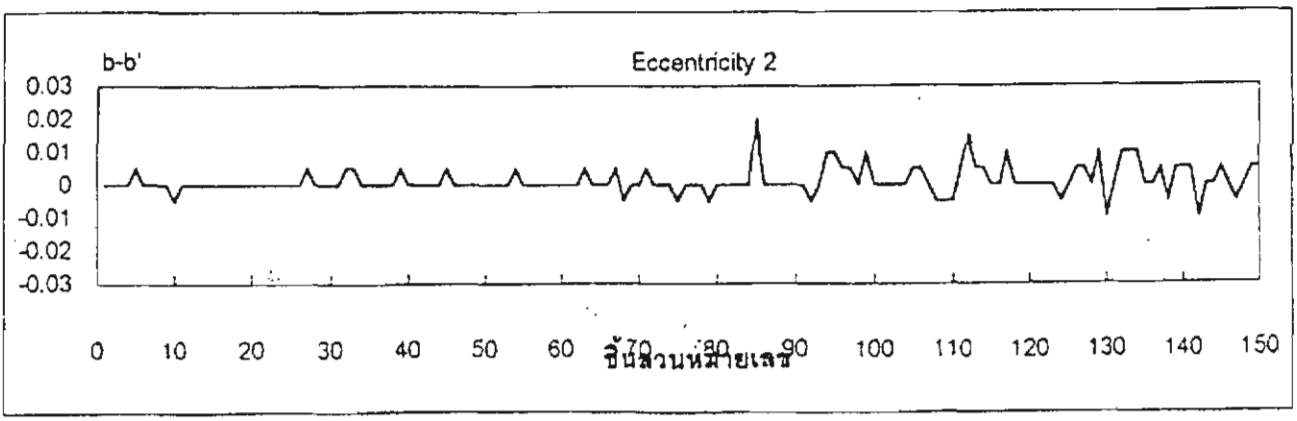


Appendix F. Test Result Run 4.

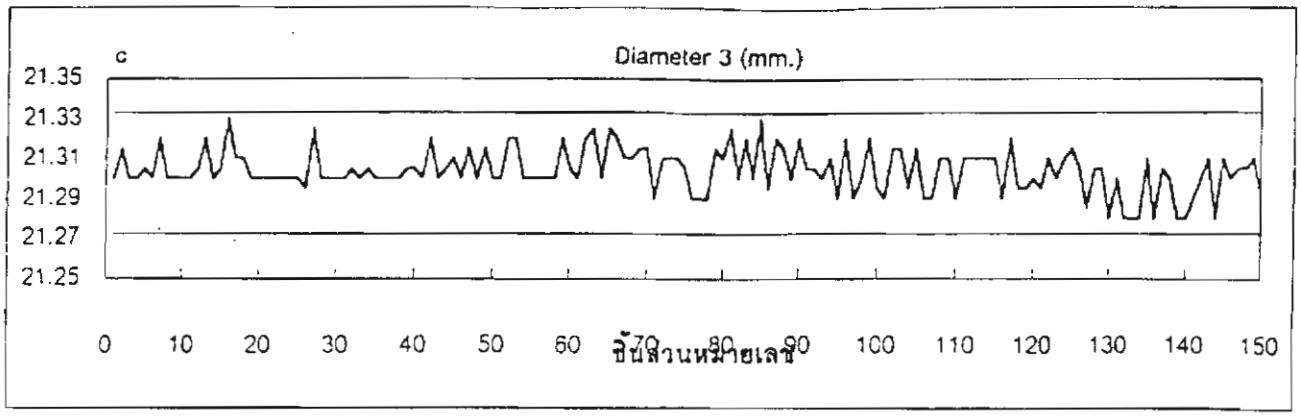


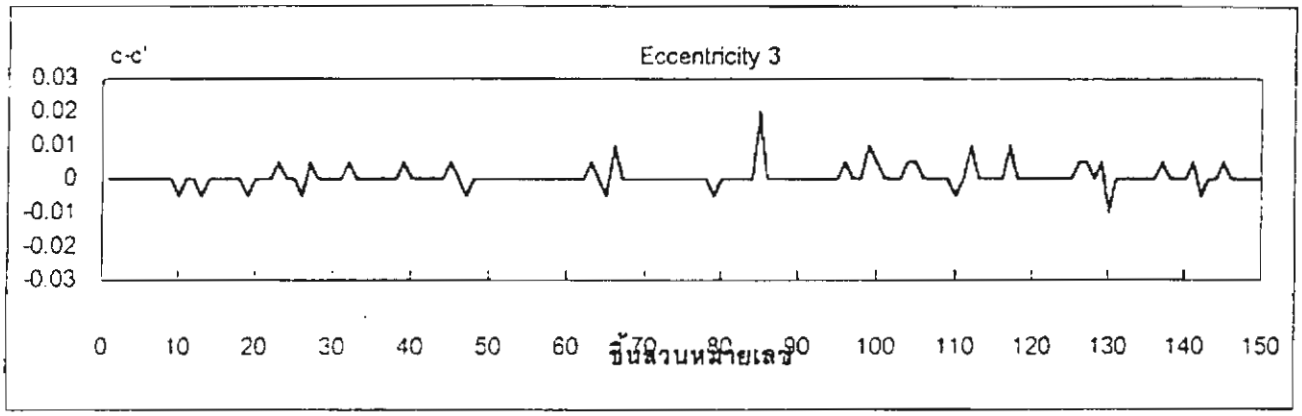


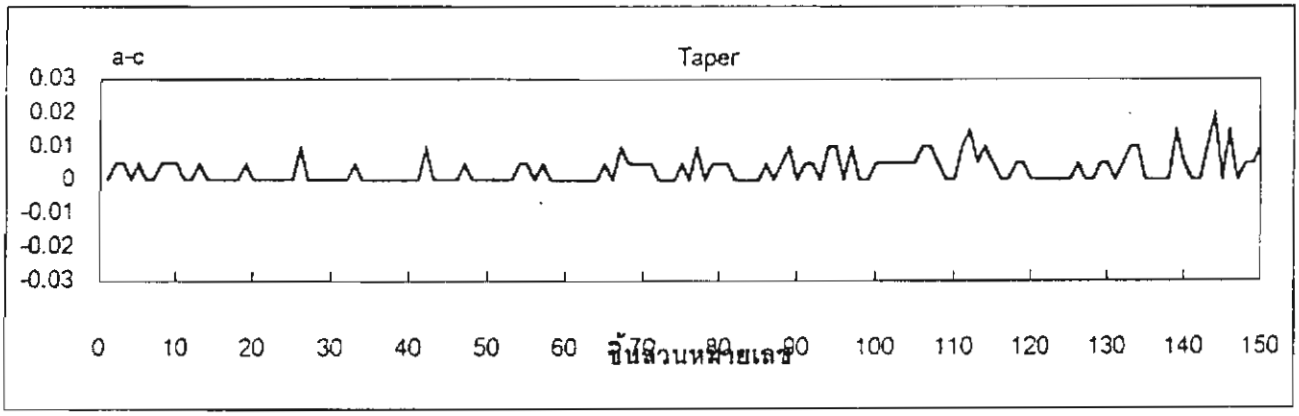


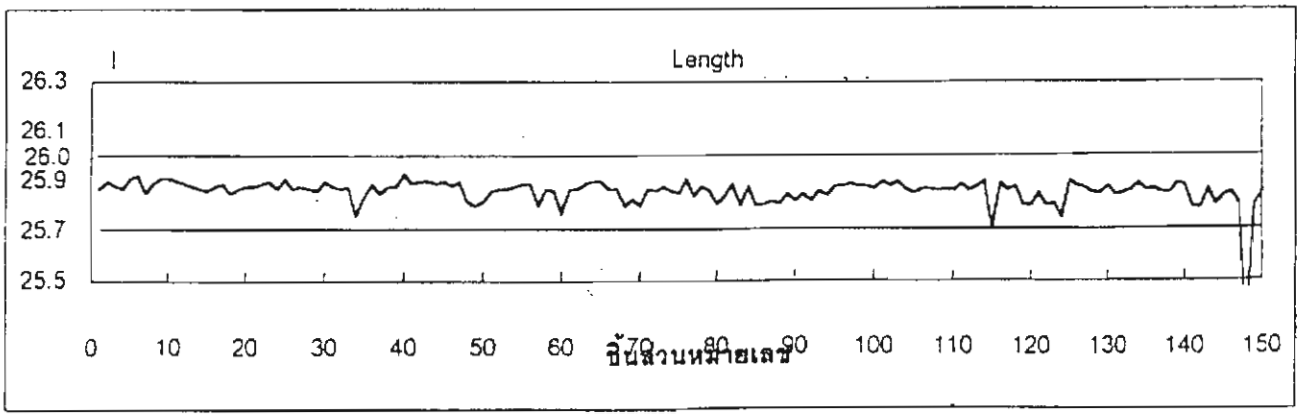


Appendix F. Test Result Run 4 (continue).



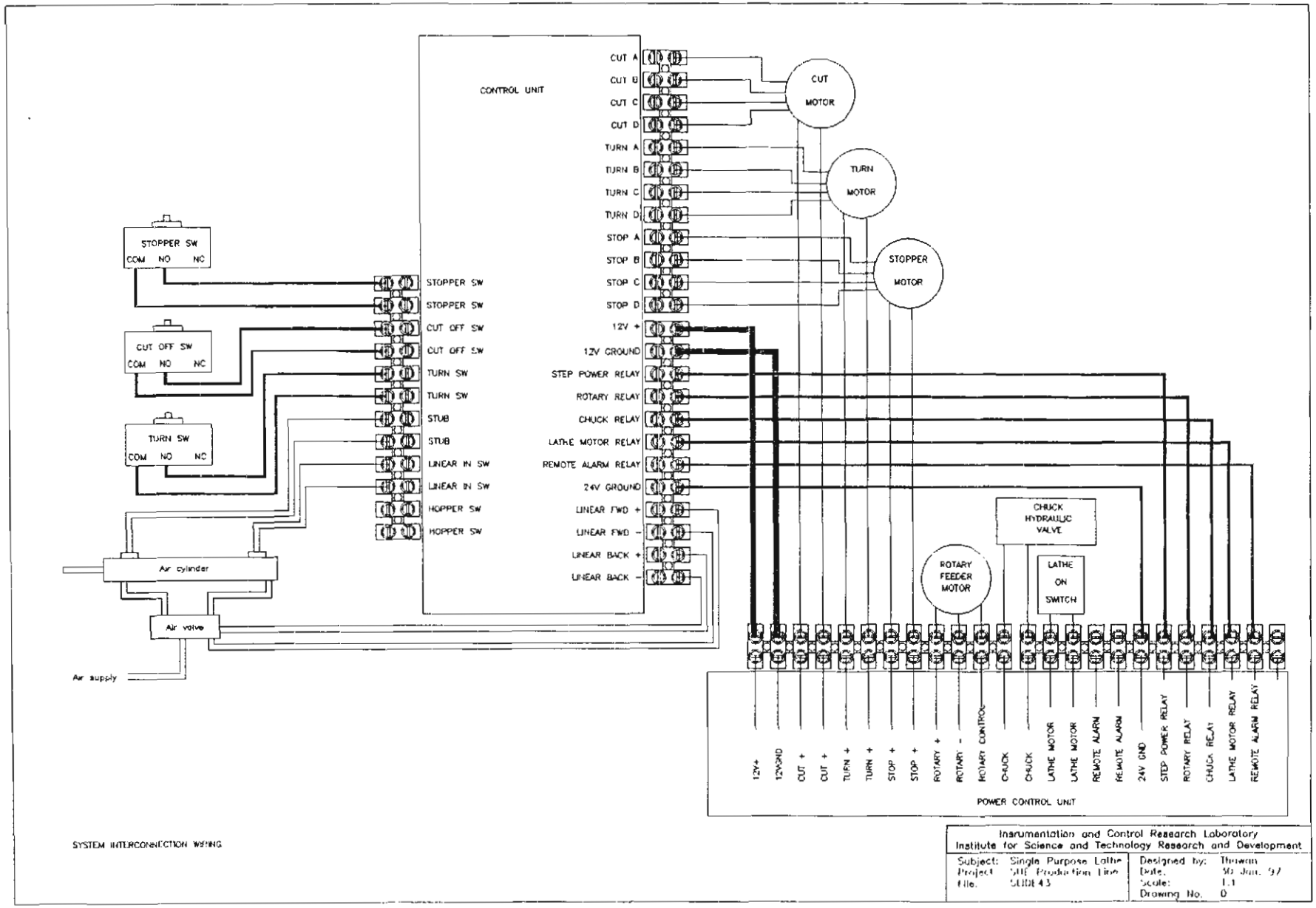






Appendix G. Drawings.

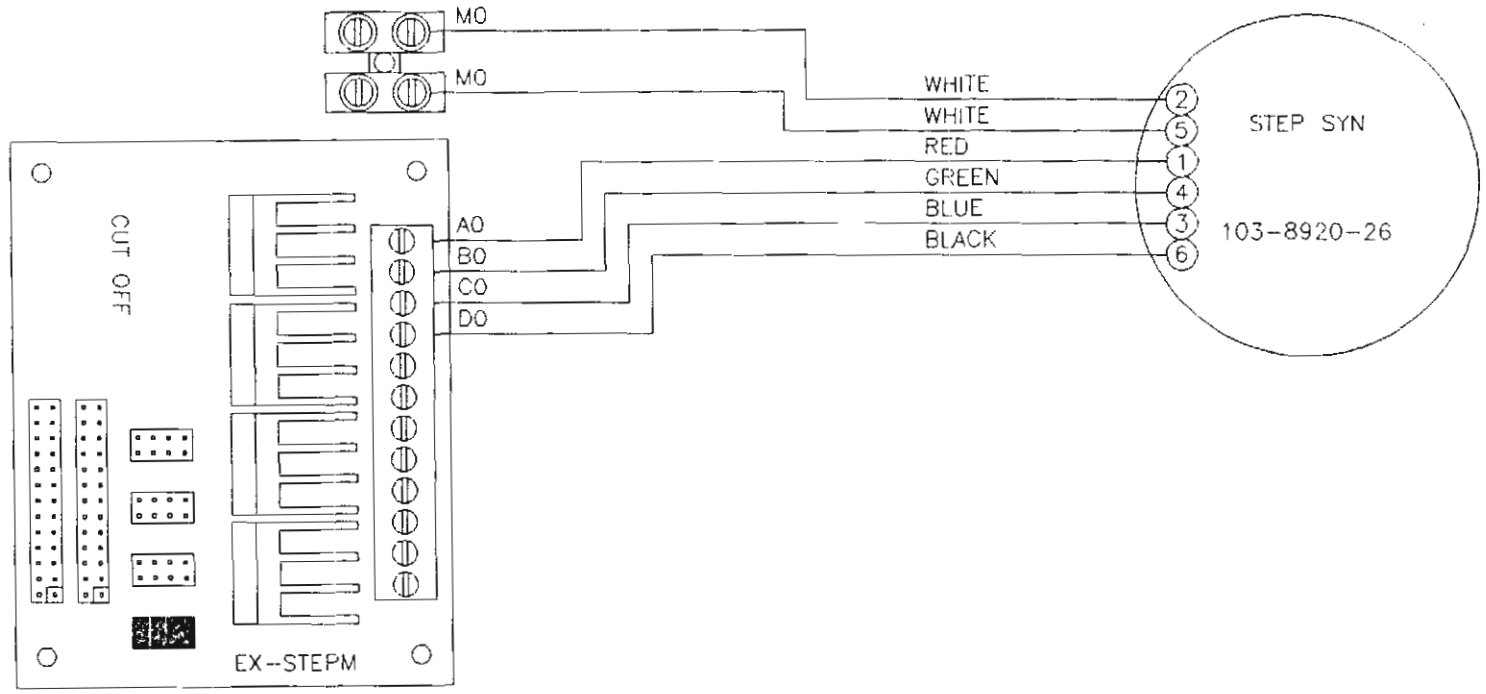
The technical drawings for the control system and mechanical parts are in the following pages:



SYSTEM INTERCONNECTION WIRING

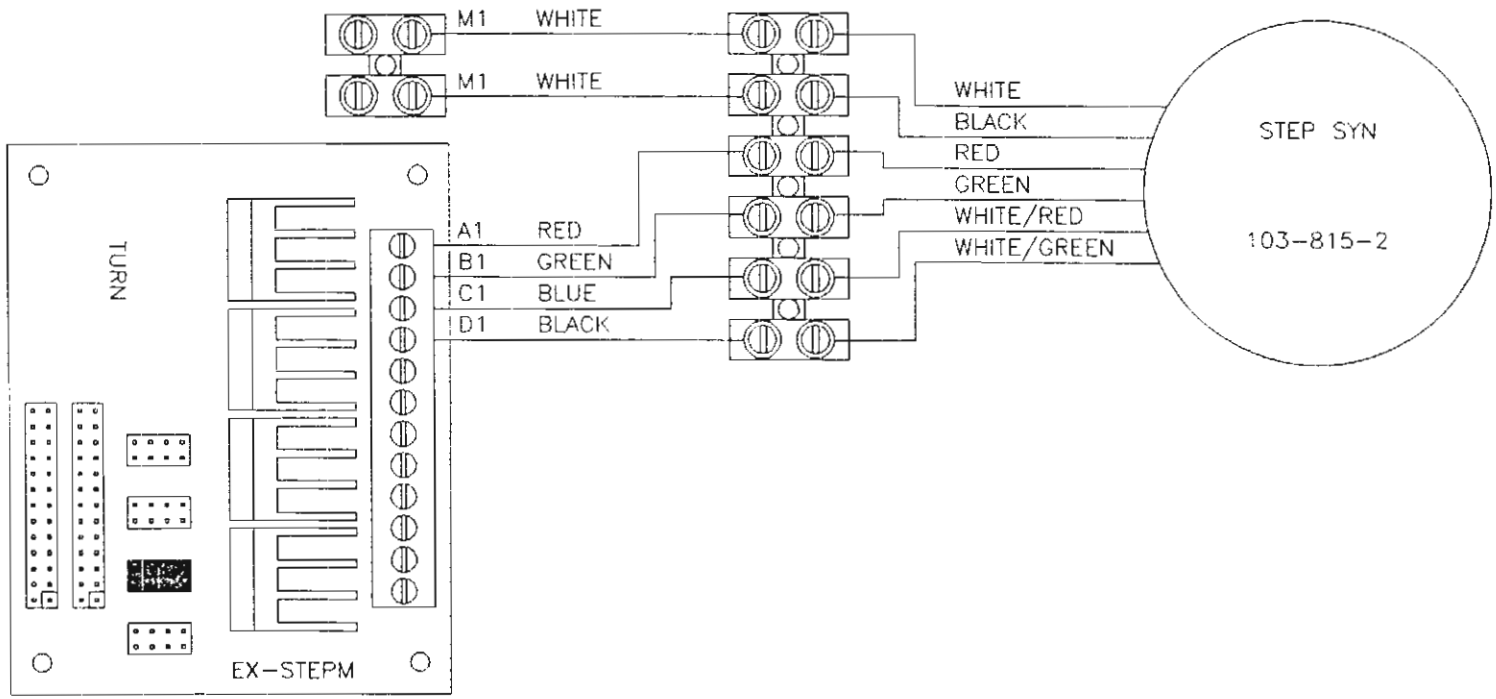
Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

Subject:	Single Purpose Lathe	Designed by:	Thiwon
Project:	SME Production Line	Date:	30 Jun. 97
File:	SLDR43	Scale:	1:1
		Drawing No.	0



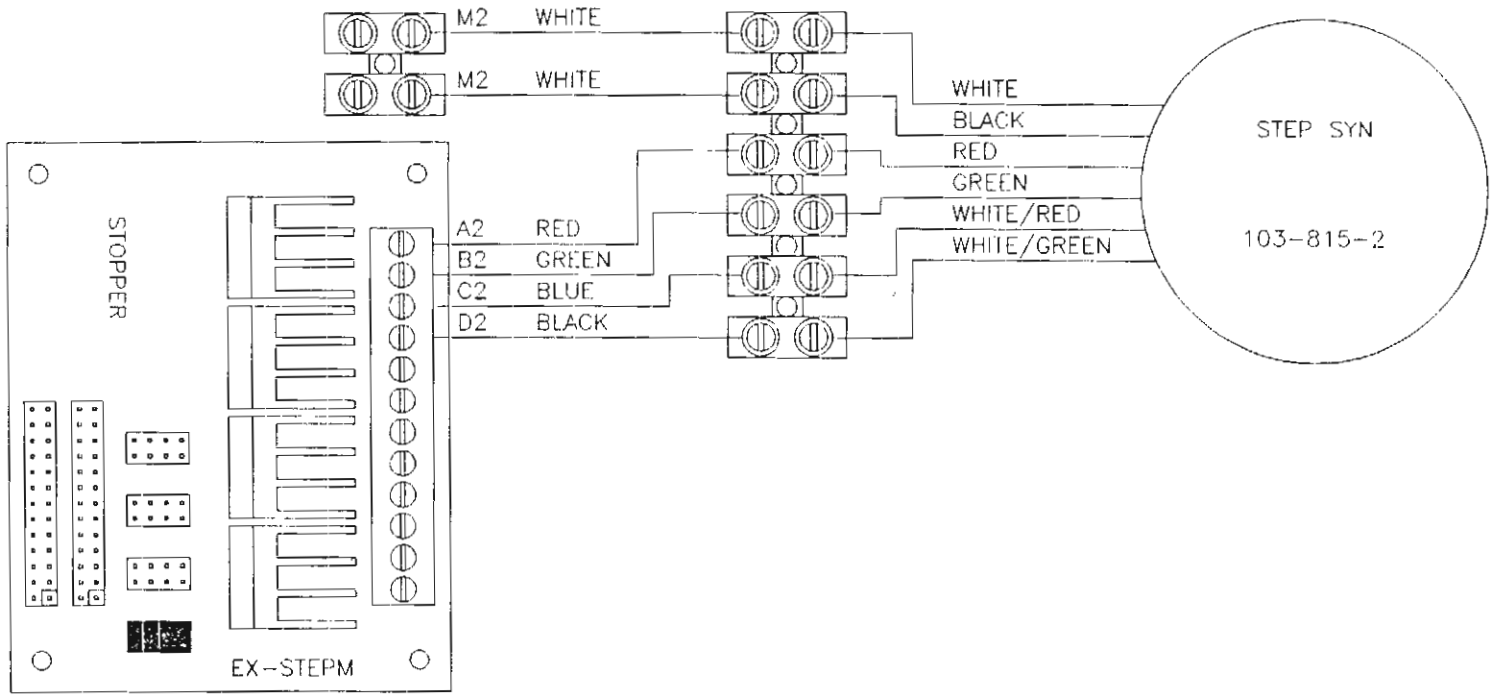
CROSS SLIDE MOTOR WIRING

Instrumentation and Control Research Laboratory Institute for Science and Technology Research and Development	
Subject: Single Purpose Lathe Project: SUE Production Line File: SLIDE31	Designed by: Thawon Date: 26 Oct 96 Scale: 1:1 Drawing No. 0



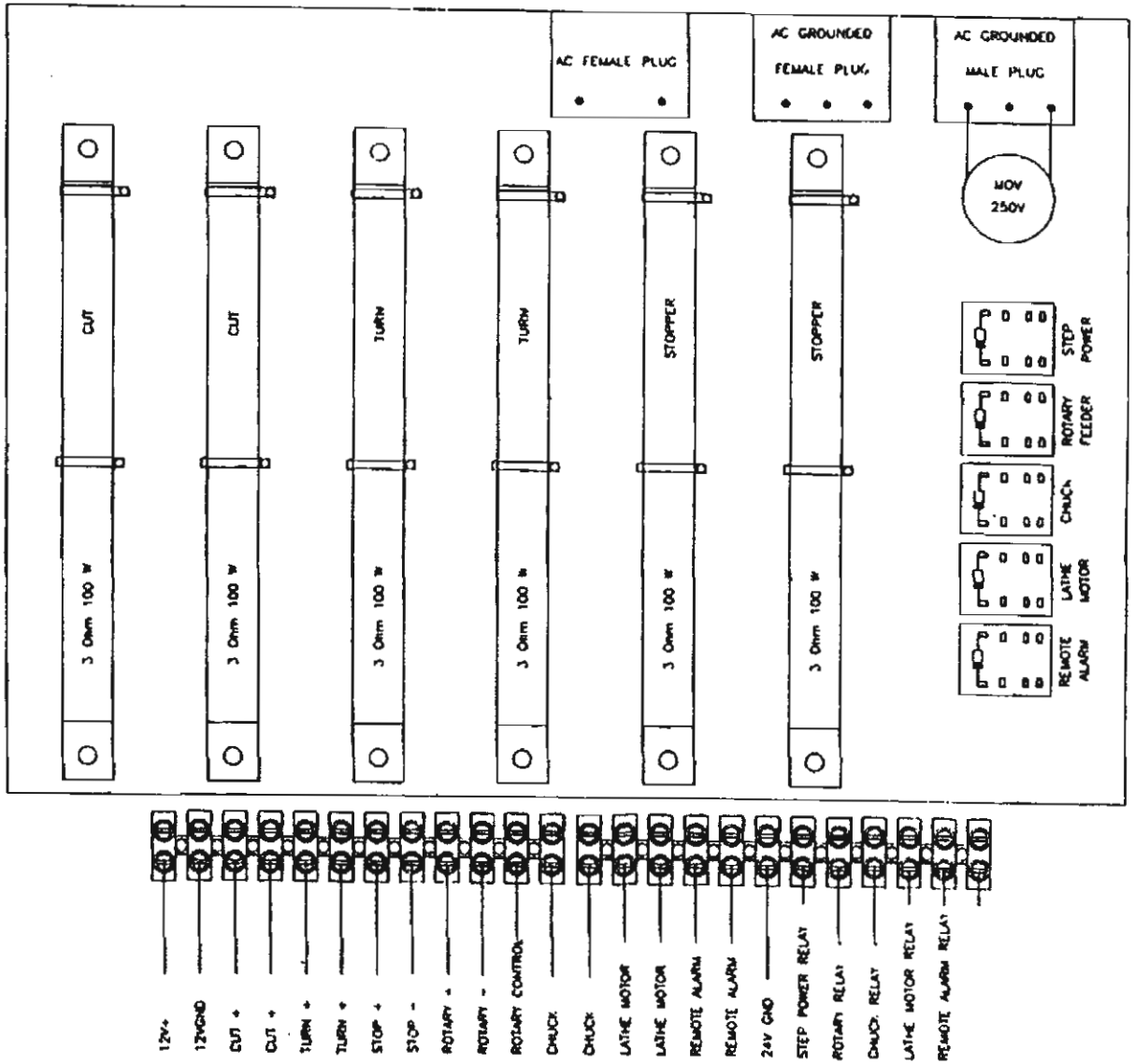
TURN SLIDE MOTOR WIRING

Instrumentation and Control Research Laboratory Institute for Science and Technology Research and Development	
Subject: Single Purpose Lathe Project: SUE Production Line File SLIDE32	Designed by: Thawan Date 26 Oct 96 Scale: 1:1 Drawing No 0



STOPPER MOTOR WIRING

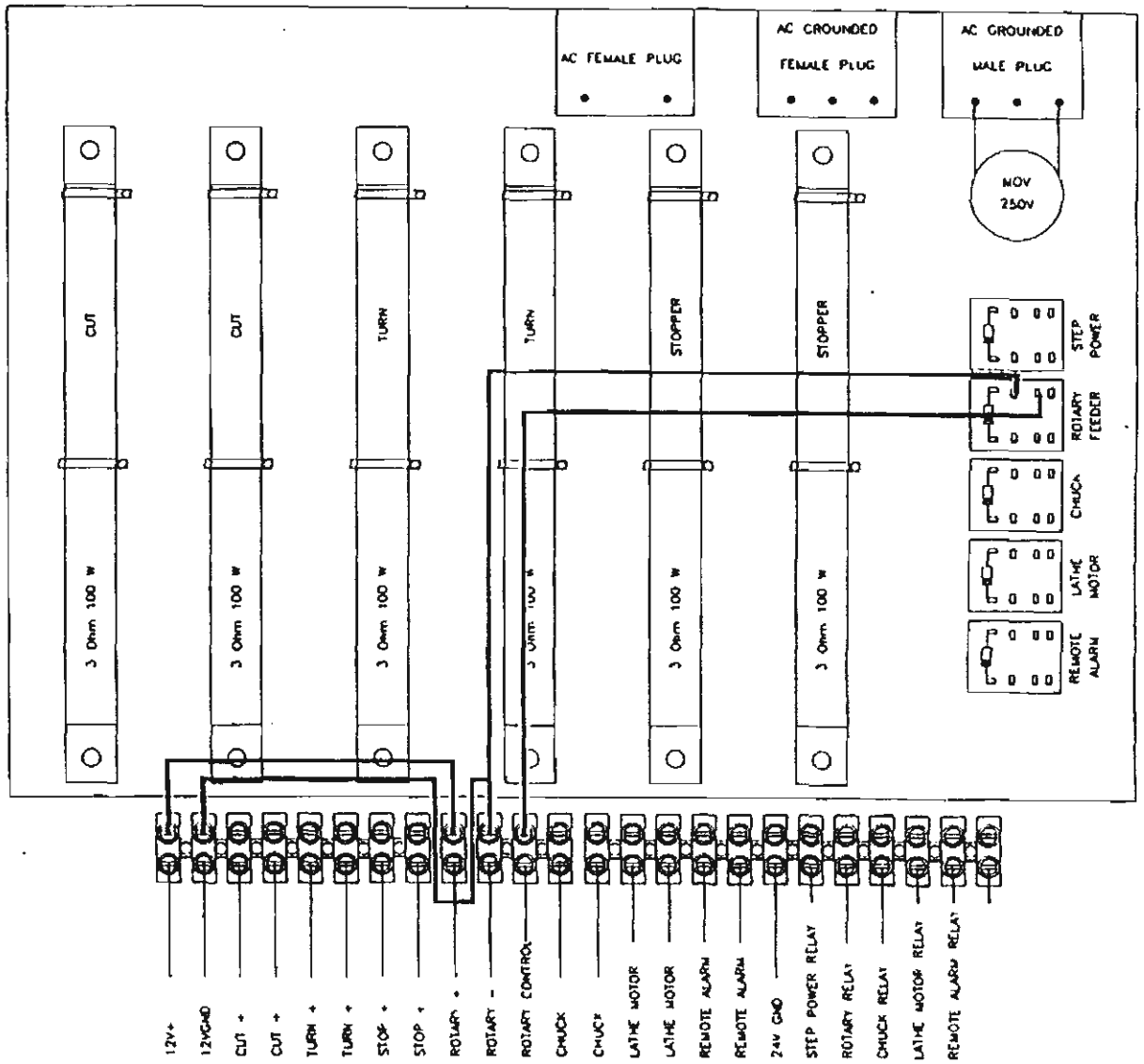
Instrumentation and Control Research Laboratory Institute for Science and Technology Research and Development	
Subject: Single Purpose Lathe Project: SUE Production Line File: SLIDE33	Designed by: Thawon Date: 26 Oct 96 Scale: 1:1 Drawing No: 0



POWER CONTROL UNIT

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

Subject:	Single Purpose Lathe,	Designed by:	Thawan
Project:	SUE Production Line	Date:	26 Oct. 95
File:	SLIDE42	Scale:	1:2
		Drawing No.	0

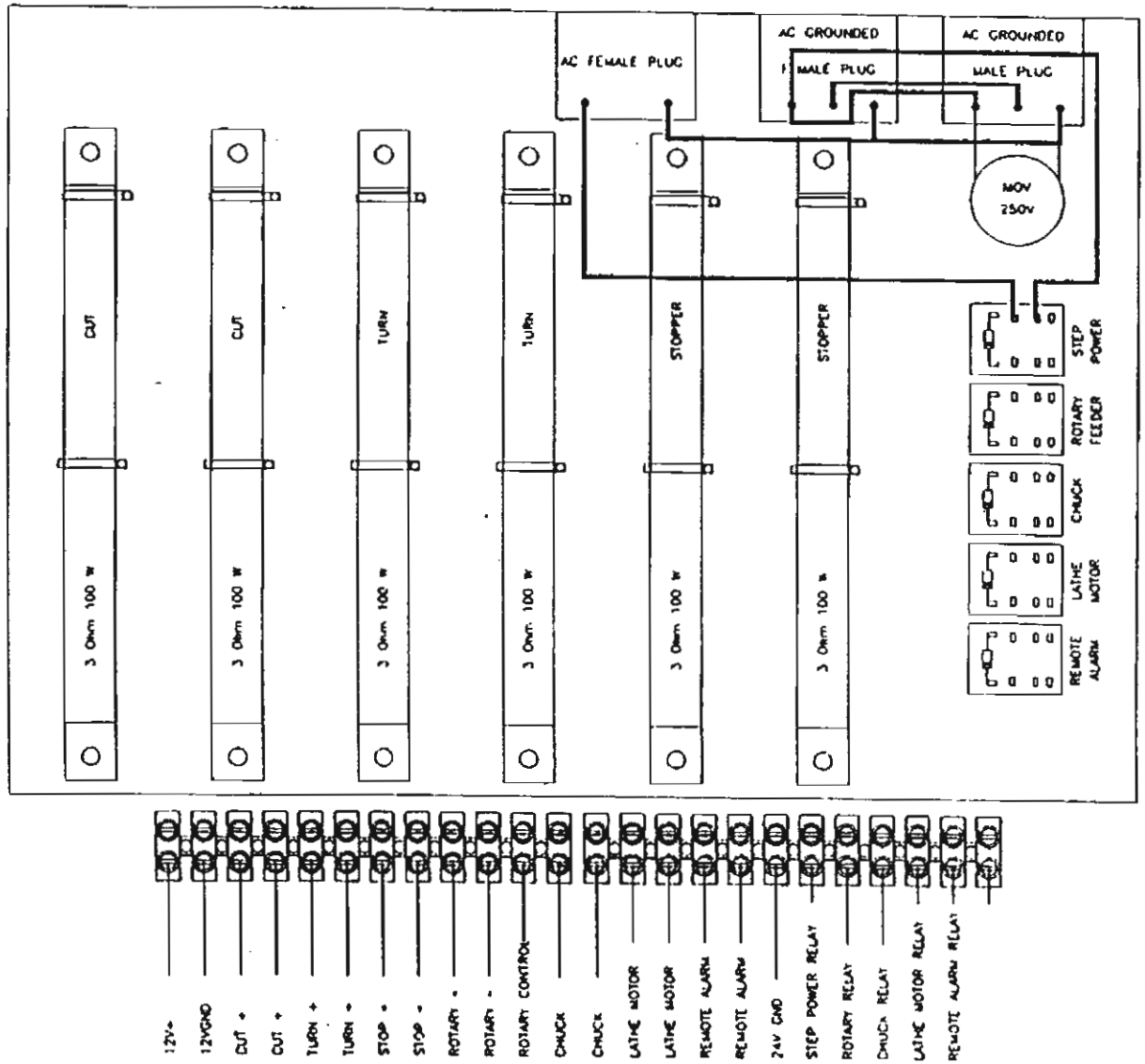


POWER CONTROL UNIT

12V

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

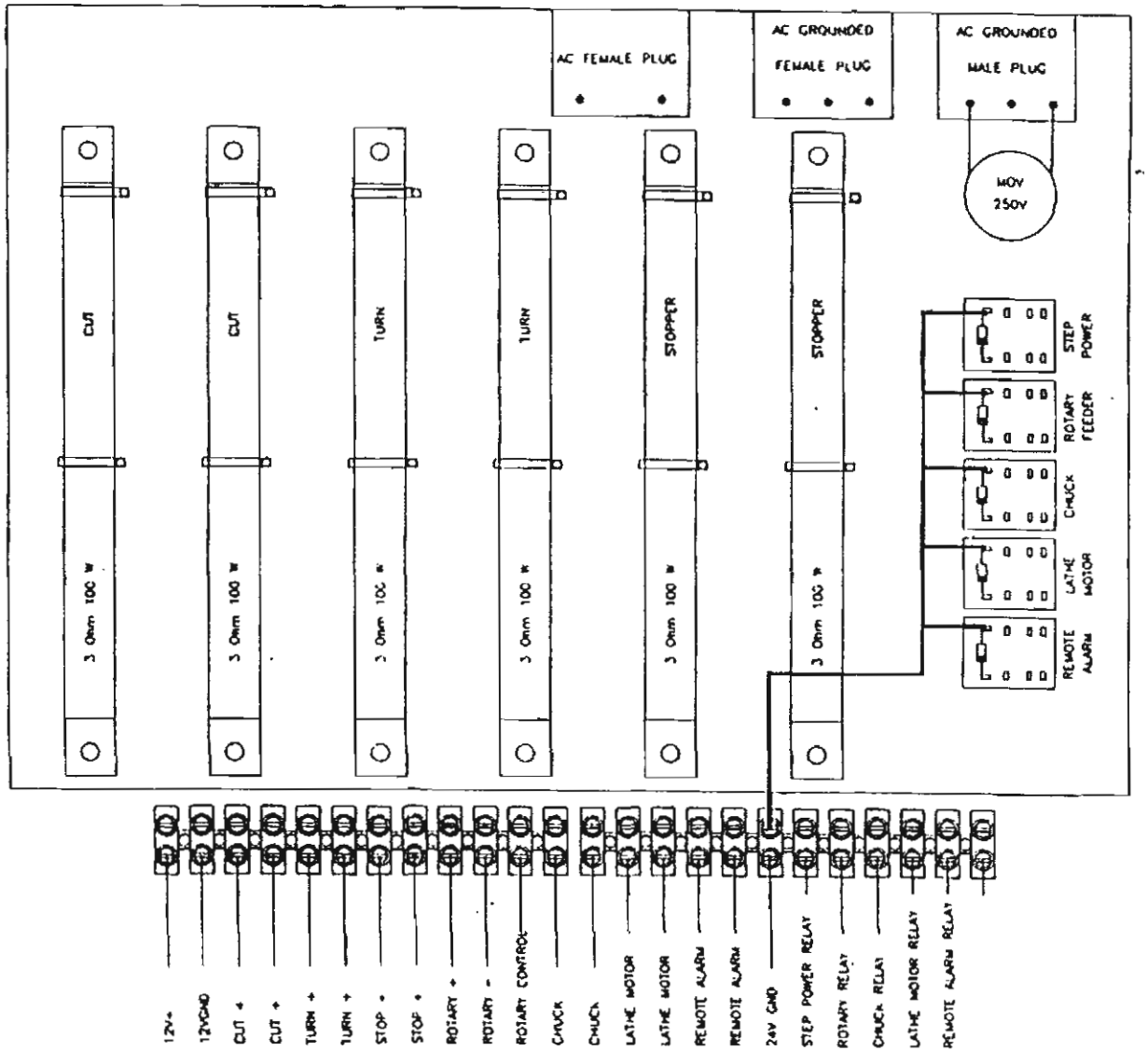
Subject: Single Purpose Lathe	Designed by: Thawon
Project: SUE Production Line	Date: 26 Oct 96
File: SLIDE42	Scale: 1:2
	Drawing No 0



POWER CONTROL UNIT

220V

Instrumentation and Control Research Laboratory Institute for Science and Technology Research and Development	
Subject: Single Purpose Lathe Project: SUE Production Line File: SLIDE42	Designed by: Ihawan Date: 26 Oct. 96 Scale: 1:2 Drawing No. 0

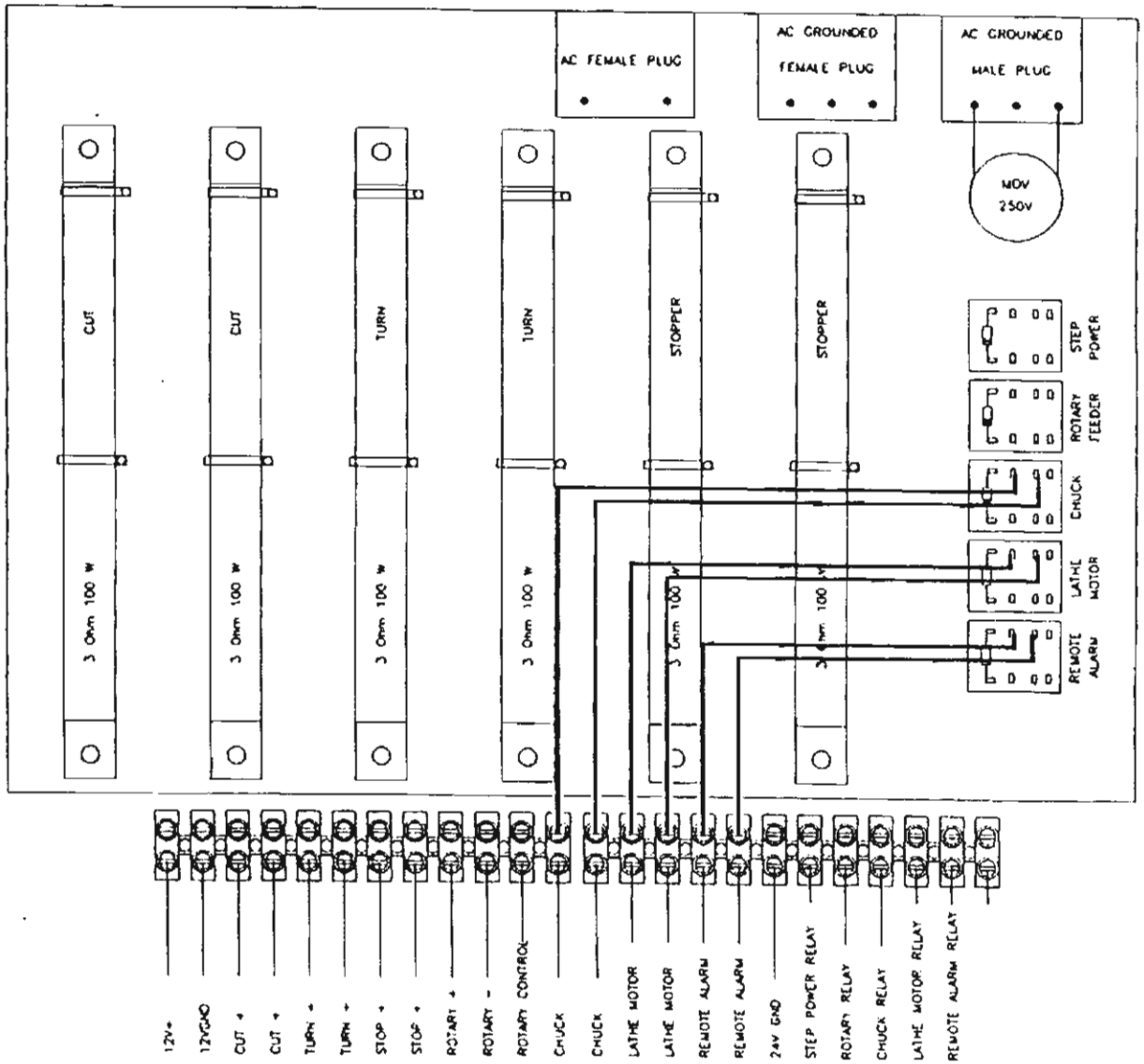


POWER CONTROL UNIT

24V

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

Subject:	Single Purpose Lathe	Designed by:	Thawon
Project:	SUE Production Line	Date:	26 Oct 95
File:	SLIDE42	Scale:	1:2
		Drawing No.	0

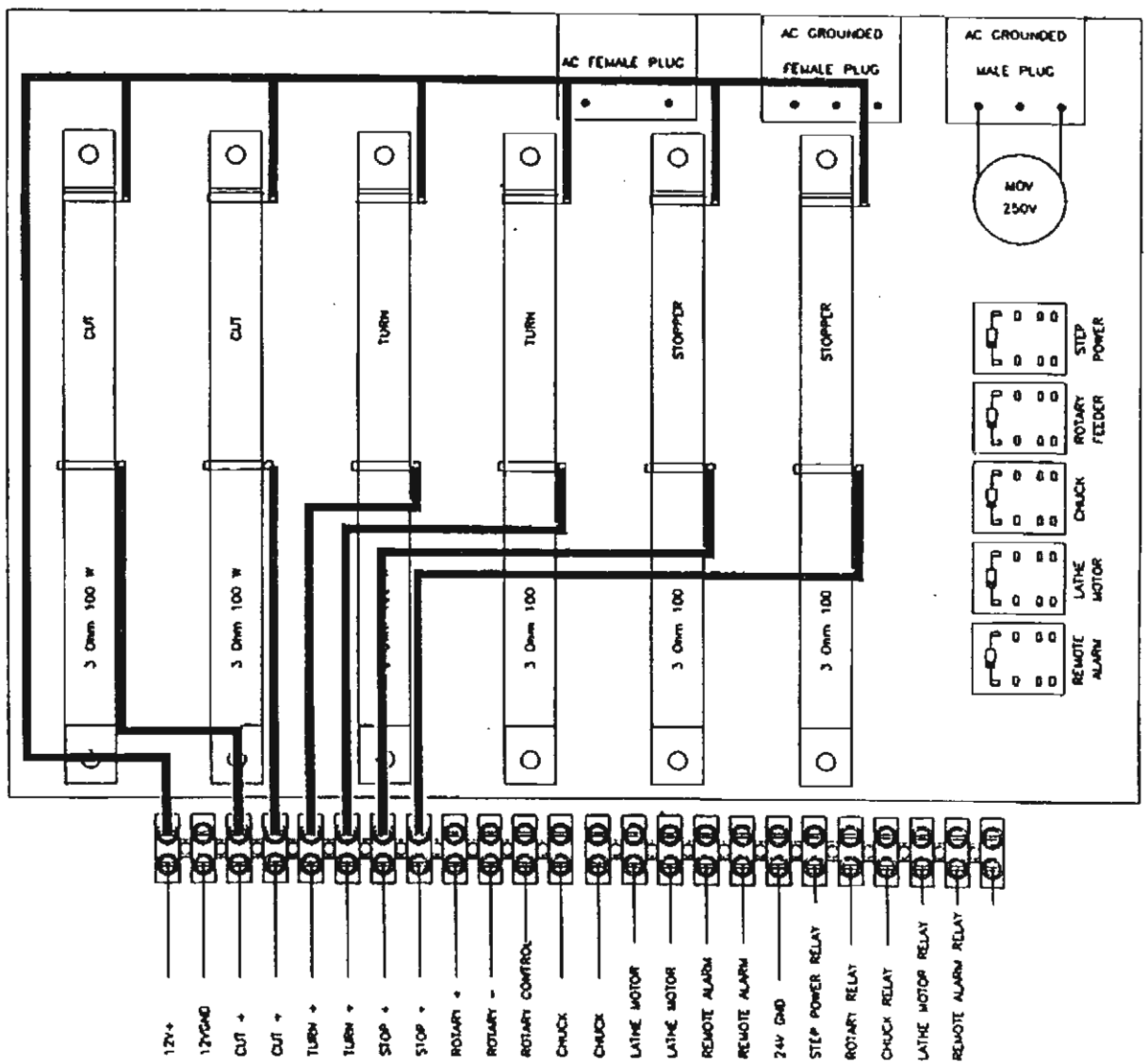


POWER CONTROL UNIT

CONTROL

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

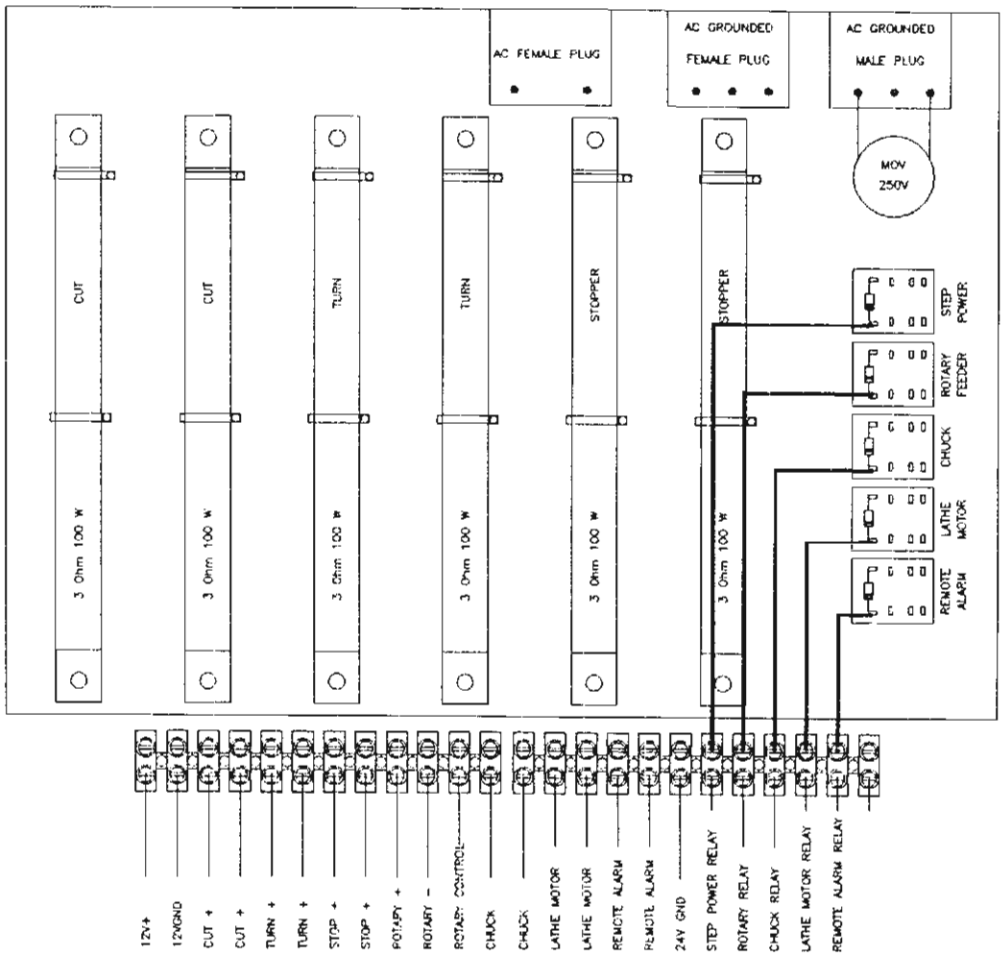
Subject	Single Purpose Lathe	Designed by:	Thawan
Project	SUE Production Line	Date	26 Oct. 98
File	SLIDE42	Scale	1:2
		Drawing No	0



POWER CONTROL Unit

R

Instrumentation and Control Research Laboratory Institute for Science and Technology Research and Development			
Subject:	Single Purpose Lathe	Designed by.	Thawon
Project.	SUE Production Line	Date:	26 Oct. 96
File	SLIDE42	Scale.	1:2
		Drawing No	0

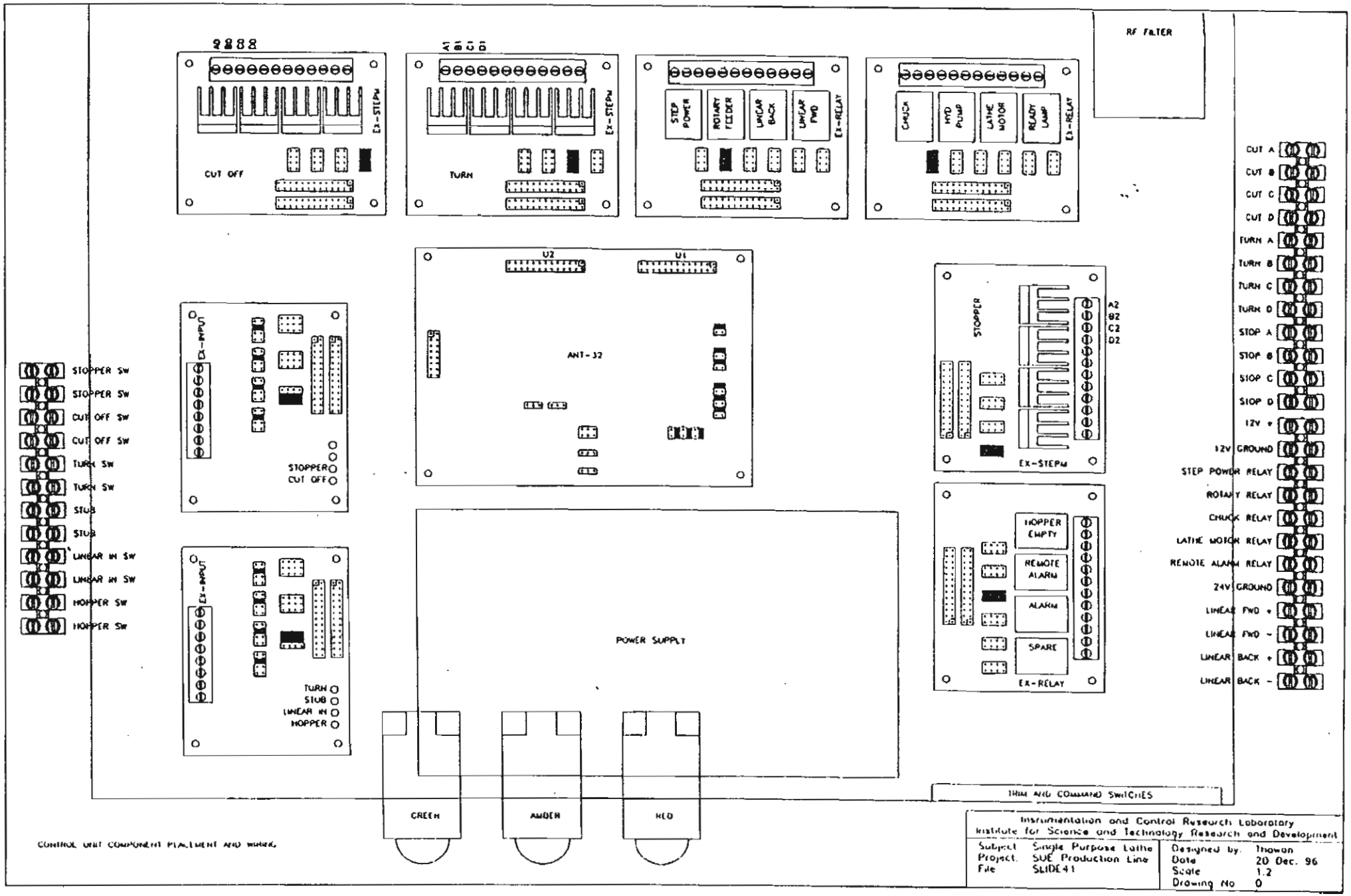


POWER CONTROL UNIT

RELAY

Insrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

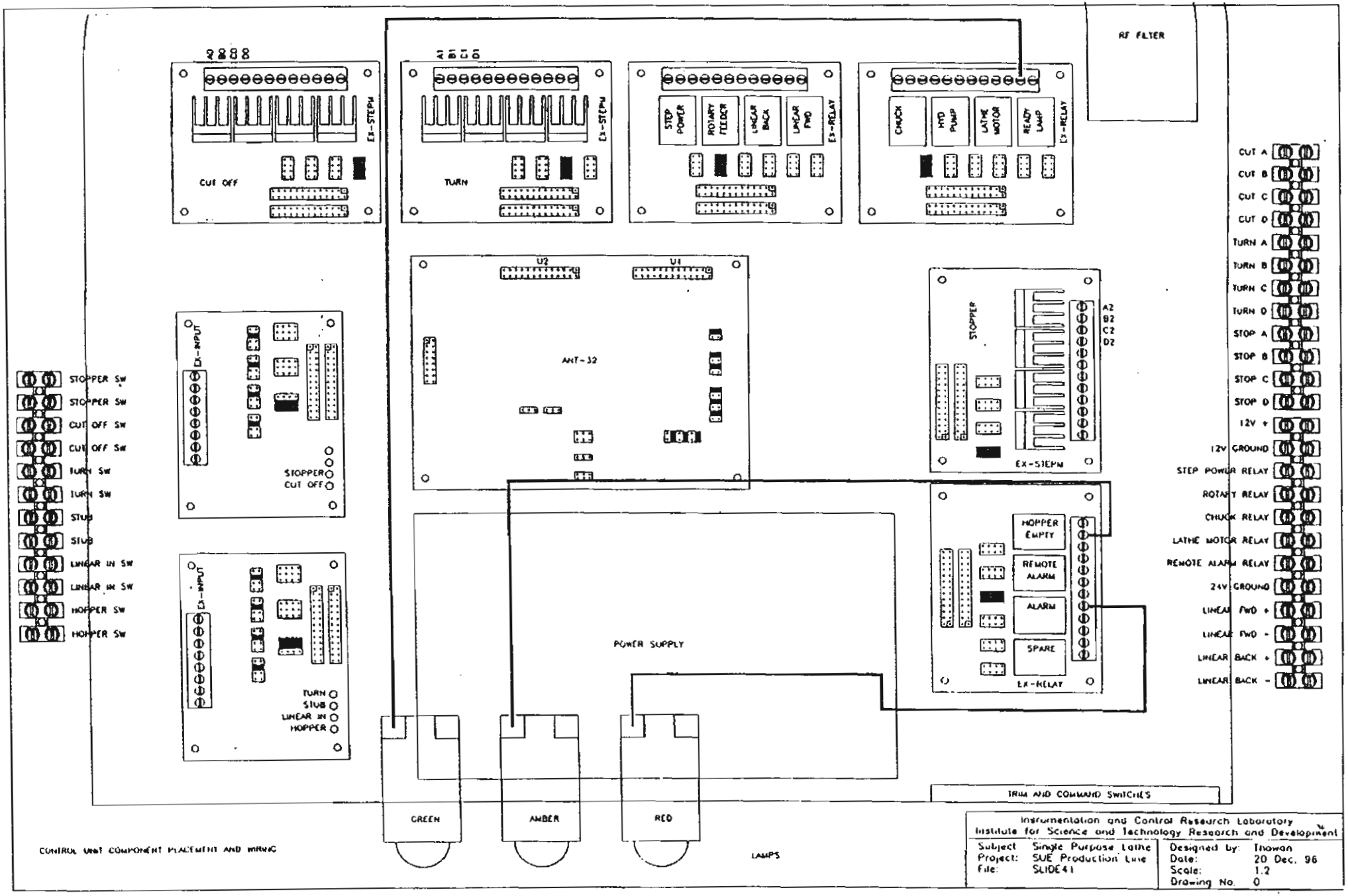
Subject:	Single Purpose Lathe	Designed by:	Thawan
Project	SUE Production Line	Date:	26 Oct 96
File:	SLIDE42	Scale:	1:2
		Drawing No.	0



CONTROL UNIT COMPONENT PLACEMENT AND WIRING

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

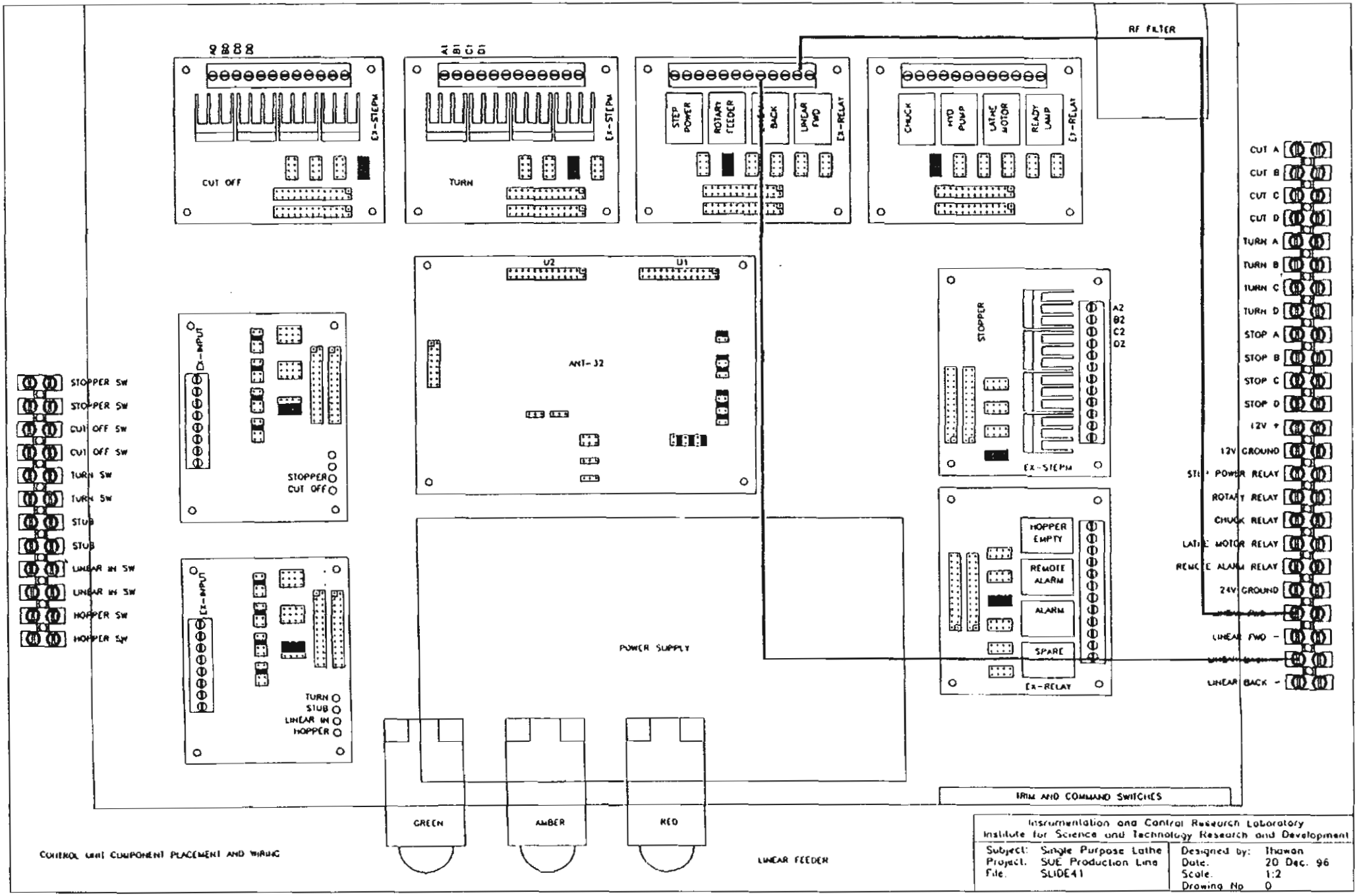
Subject	Single Purpose Lathe	Designed by	Thowan
Project	SUE Production Line	Date	20 Dec. 96
File	SLIDE41	Scale	1.2
		Drawing No	0



CONTROL UNIT COMPONENT PLACEMENT AND WIRING

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

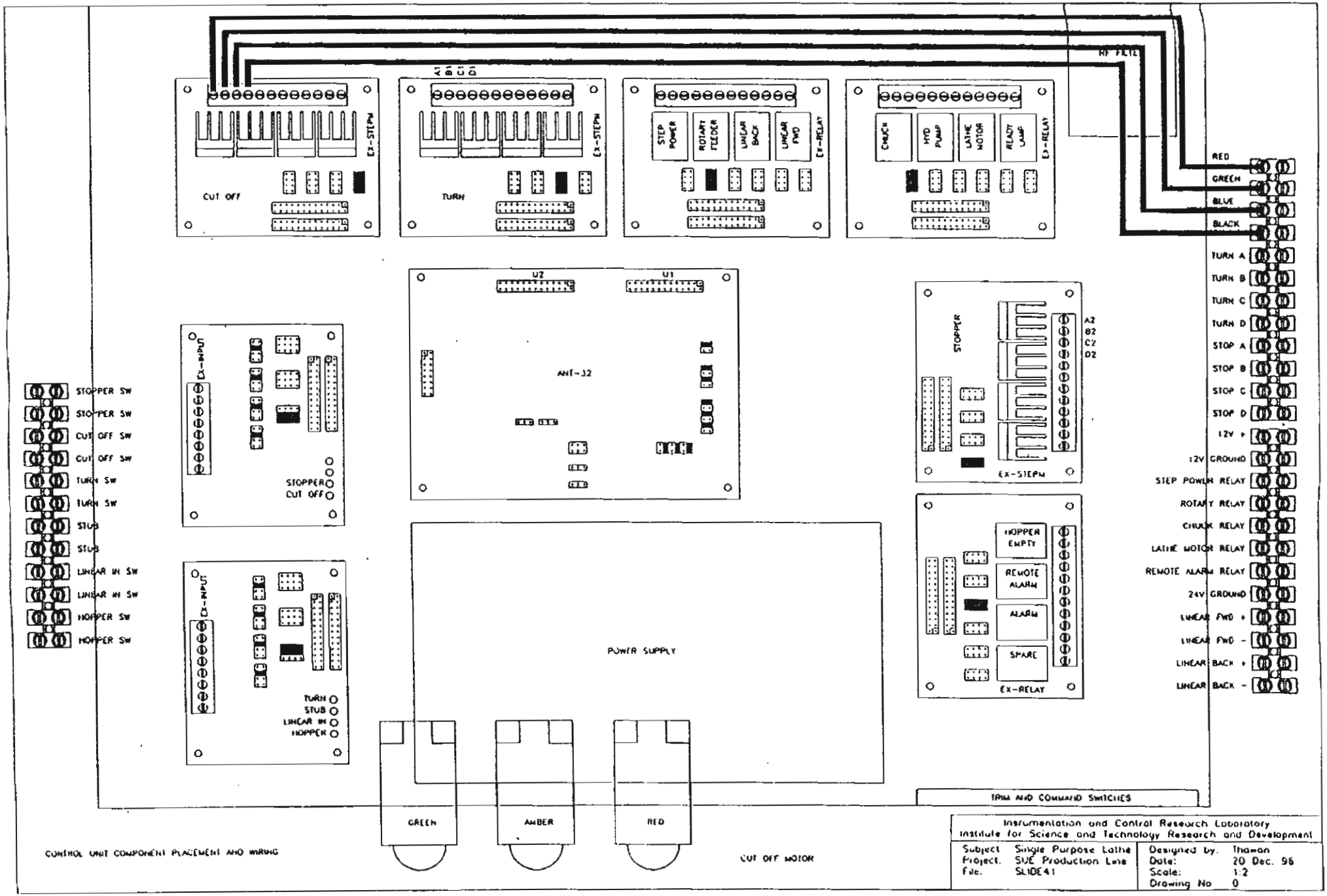
Subject	Single Purpose Lathe	Designed by:	Thawan
Project:	SUE Production Line	Date:	20 Dec. 96
File:	SLIDE41	Scale:	1:2
		Drawing No.	0



CONTROL unit COMPONENT PLACEMENT AND WIRING

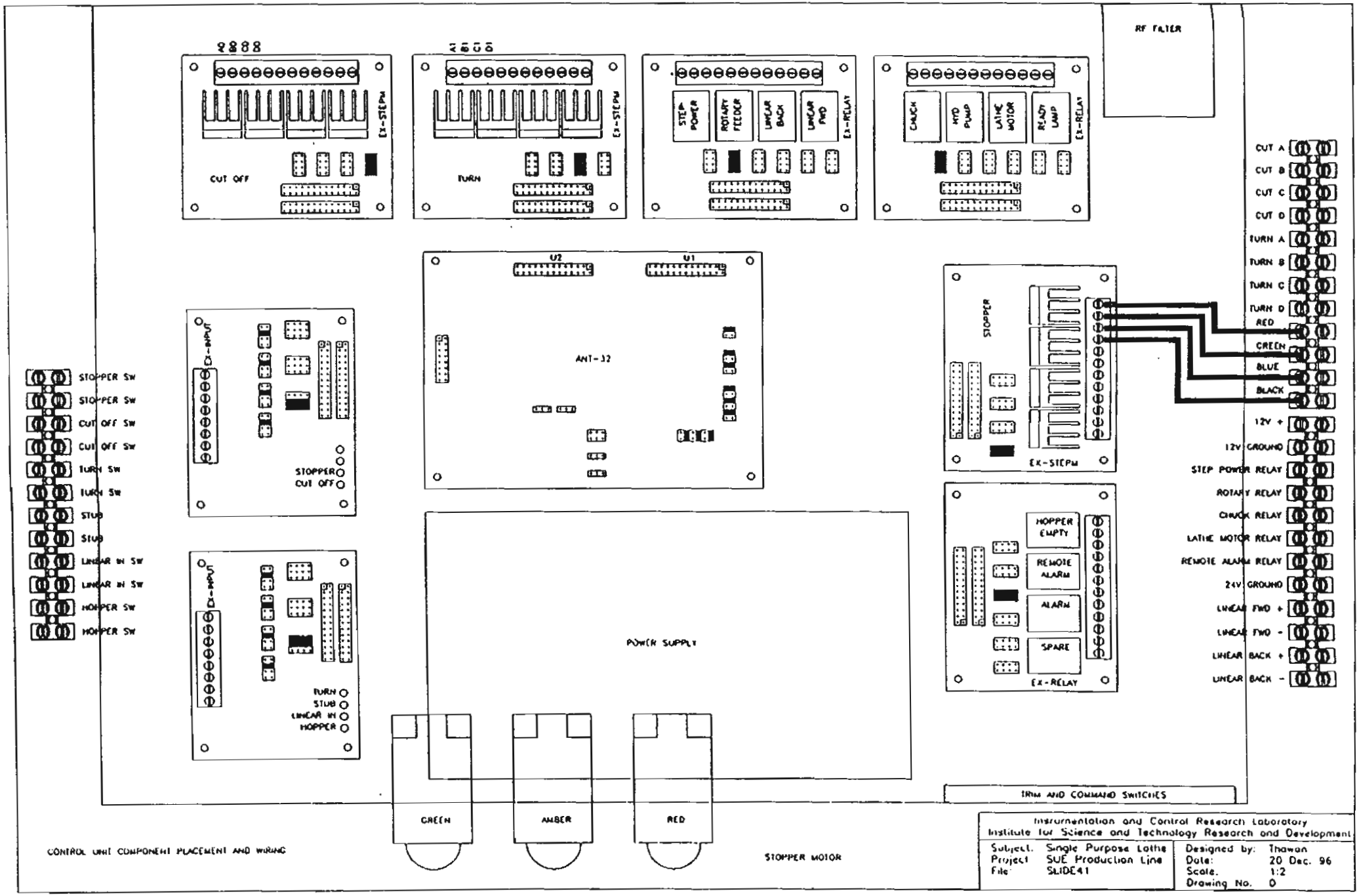
Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

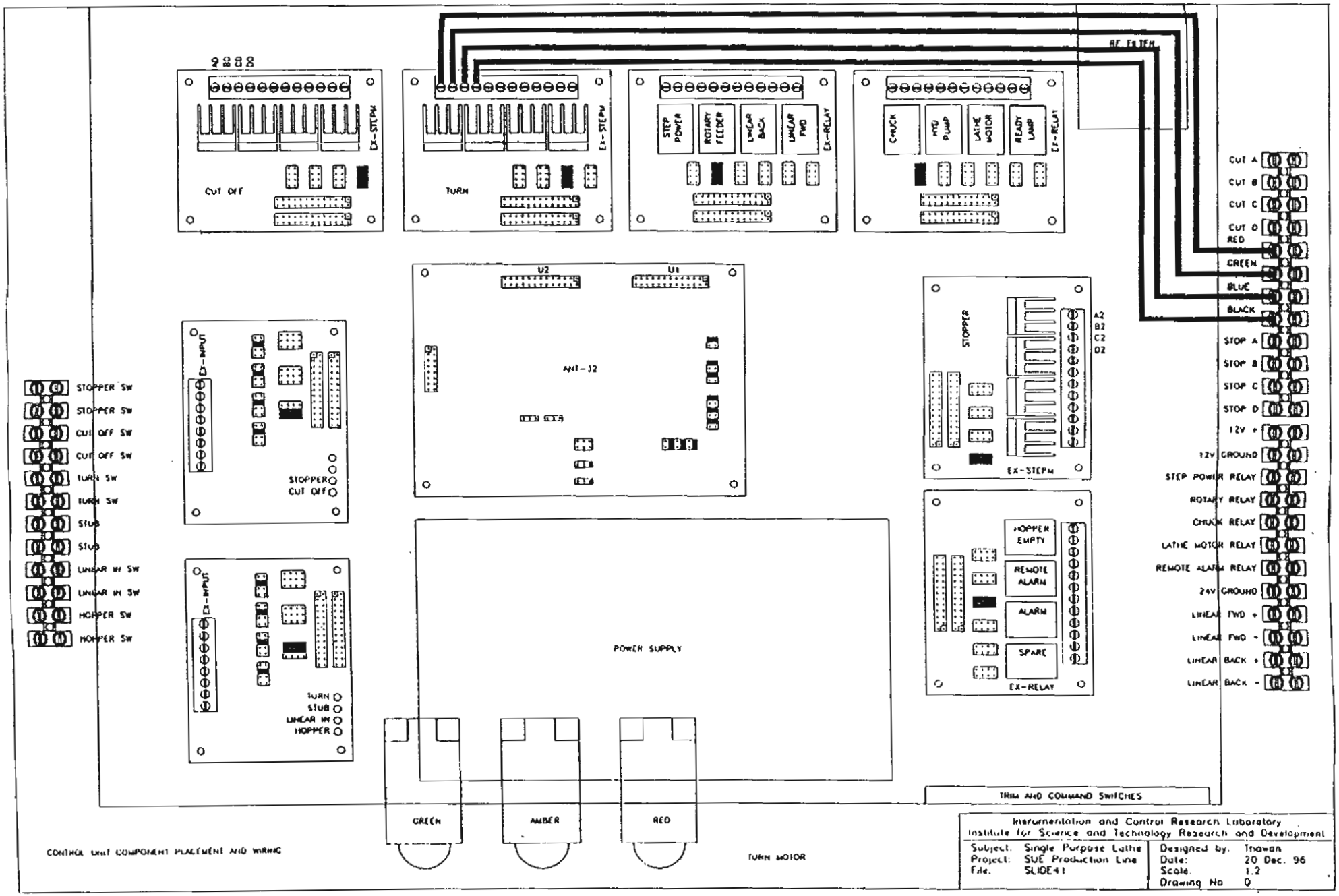
Subject:	Single Purpose Lathe	Designed by:	Thawan
Project:	SUE Production Line	Date:	20 Dec. 96
File:	SLIDE41	Scale:	1:2
		Drawing No	0

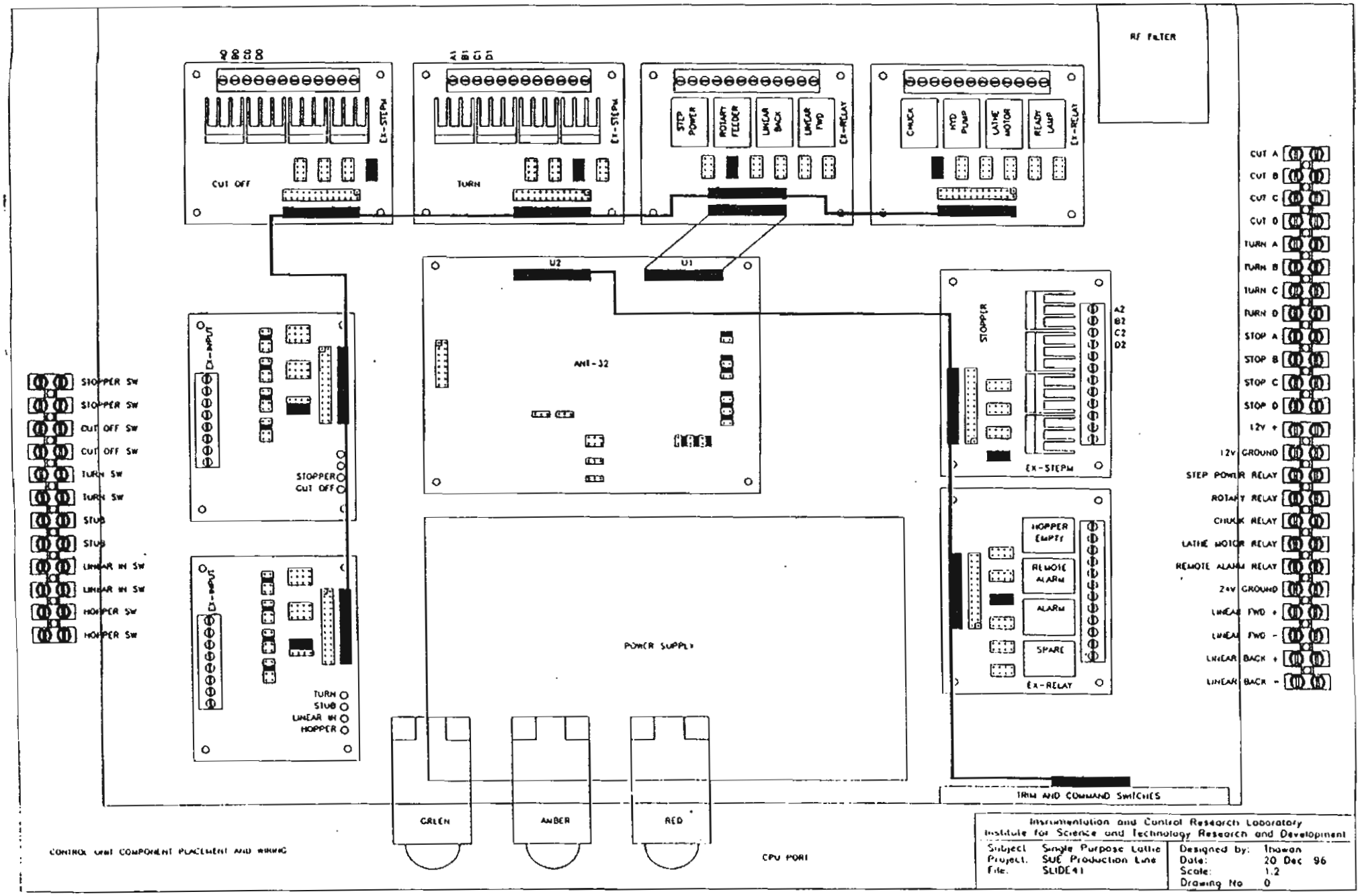


Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

Subject	Single Purpose Lathe	Designed by.	Thawon
Project.	SUE Production Line	Date:	20 Dec. 96
File.	SLIDE41	Scale:	1:2
		Drawing No	0


RF FILTER
EX-STEPM
CUT OFF
TURN
EX-STEPM
STEP POWER
ROTARY FEEDER
LINEAR BACK
LINEAR FWD
EX-RELAY
CHUCK
HYD PUMP
LATHE MOTOR
READY LAMP
EX-RELAY
CUT A
CUT B
CUT C
CUT D
TURN A
TURN B
TURN C
TURN D
RED
GREEN
BLUE
BLACK
12V +
12V GROUND
STEP POWER RELAY
ROTARY RELAY
CHUCK RELAY
LATHE MOTOR RELAY
REMOTE ALARM RELAY
24V GROUND
LINEAR FWD +
LINEAR FWD -
LINEAR BACK +
LINEAR BACK -
STOPPER SW
STOPPER SW
CUT OFF SW
CUT OFF SW
TURN SW
TURN SW
STUB
STUB
LINEAR IN SW
LINEAR IN SW
HOPPER SW
HOPPER SW
EX-INPUT
STOPPER
CUT OFF
EX-INPUT
TURN
STUB
LINEAR IN
HOPPER
U2
U1
ANT-32
STOPPER
EX-STEPM
HOPPER EMPTY
REMOTE ALARM
ALARM
SPARE
EX-RELAY
POWER SUPPLY
IRM AND COMMAND SWITCHES
GREEN
AMBER
RED
STOPPER MOTOR
CONTROL UNIT COMPONENT PLACEMENT AND WIRING
Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development
Subject. Single Purpose Lathe
Project SUE Production Line
File SLIDE41
Designed by: Thawan
Date: 20 Dec. 96
Scale. 1:2
Drawing No. 0

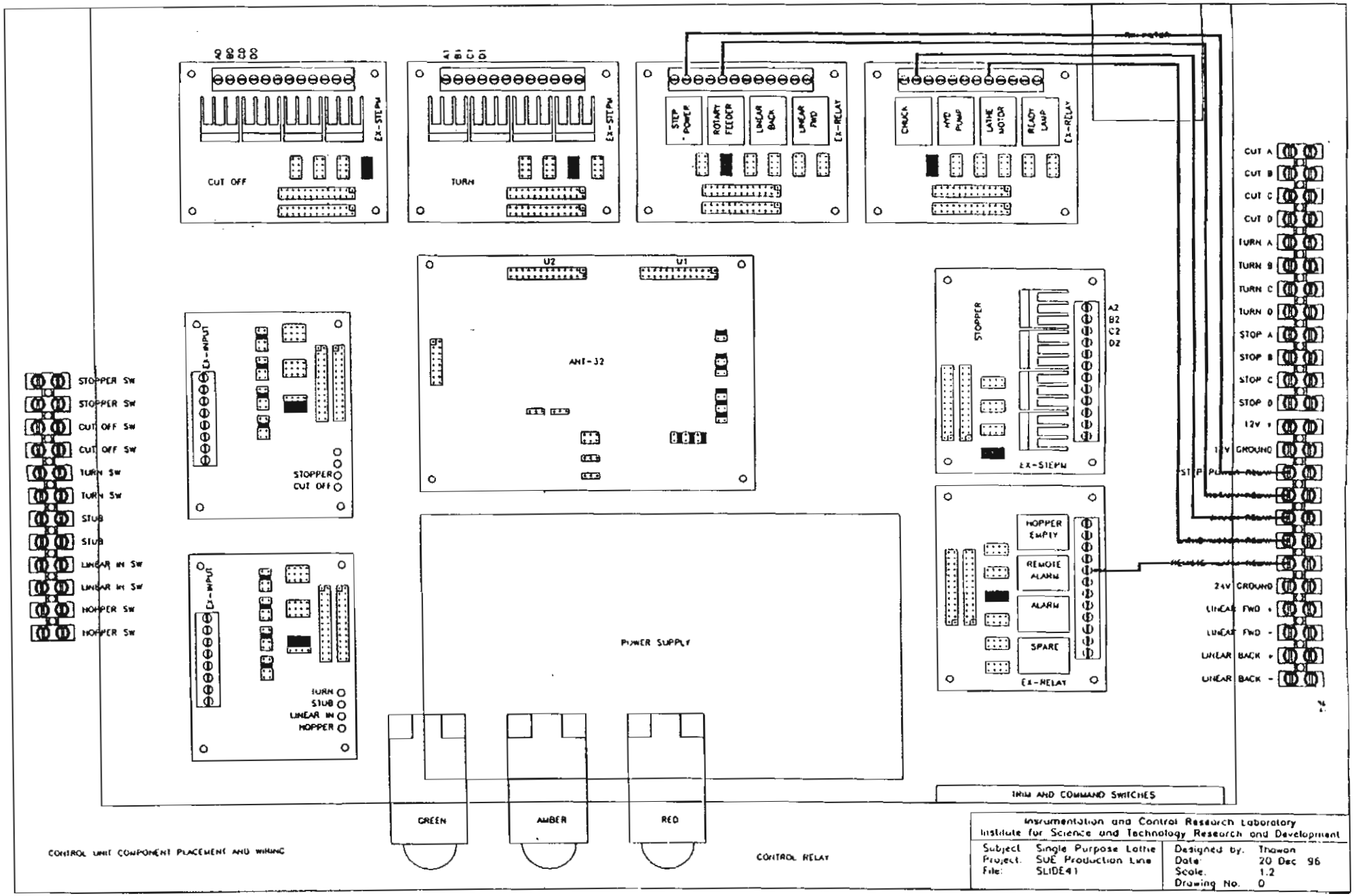

EX-STEPM
CUT OFF
TURN
EX-RELAY
STEP POWER
ROTARY FEEDER
LINEAR BACK
LINEAR FWD
CHUCK
HYD PUMP
LATHE MOTOR
READY LAMP
U2
U1
ANT-J2
EX-INPUT
STOPPER
CUT OFF
TURN
STUB
LINEAR IN
HOPPER
POWER SUPPLY
HOPPER EMPTY
REMOTE ALARM
ALARM
SPARE
A2
B2
C2
D2
CUT A
CUT B
CUT C
CUT D
RED
GREEN
BLUE
BLACK
STOP A
STOP B
STOP C
STOP D
12V +
12V GROUND
STEP POWER RELAY
ROTARY RELAY
CHUCK RELAY
LATHE MOTOR RELAY
REMOTE ALARM RELAY
24V GROUND
LINEAR FWD +
LINEAR FWD -
LINEAR BACK +
LINEAR BACK -
STOPPER SW
STOPPER SW
CUT OFF SW
CUT OFF SW
TURN SW
TURN SW
STUB
STUB
LINEAR IN SW
LINEAR IN SW
HOPPER SW
HOPPER SW
GREEN
AMBER
RED
TURN MOTOR
TRIM AND COMMAND SWITCHES
CONTROL UNIT COMPONENT PLACEMENT AND WIRING
Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development
Subject: Single Purpose Lathe
Project: SUE Production Line
File: SLIDE41
Designed by: Thawan
Date: 20 Dec. 96
Scale: 1:2
Drawing No 0



CONTROL UNIT COMPONENT PLACEMENT AND WIRING

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

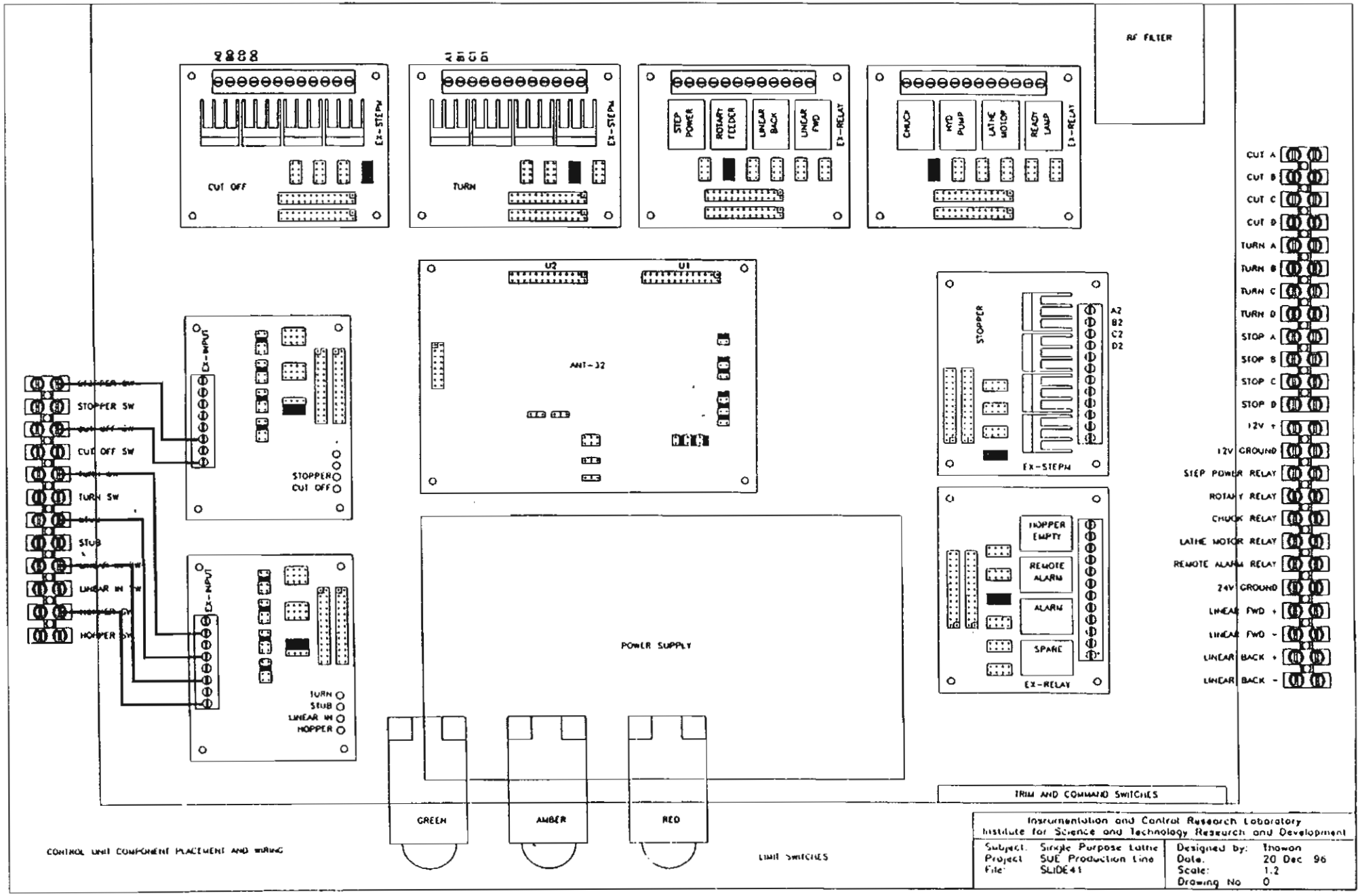
Subject:	Single Purpose Lathe	Designed by:	Thawan
Project:	SUE Production Line	Date:	20 Dec 96
File:	SLIDE41	Scale:	1:2
		Drawing No	0



CONTROL UNIT COMPONENT PLACEMENT AND WIRING

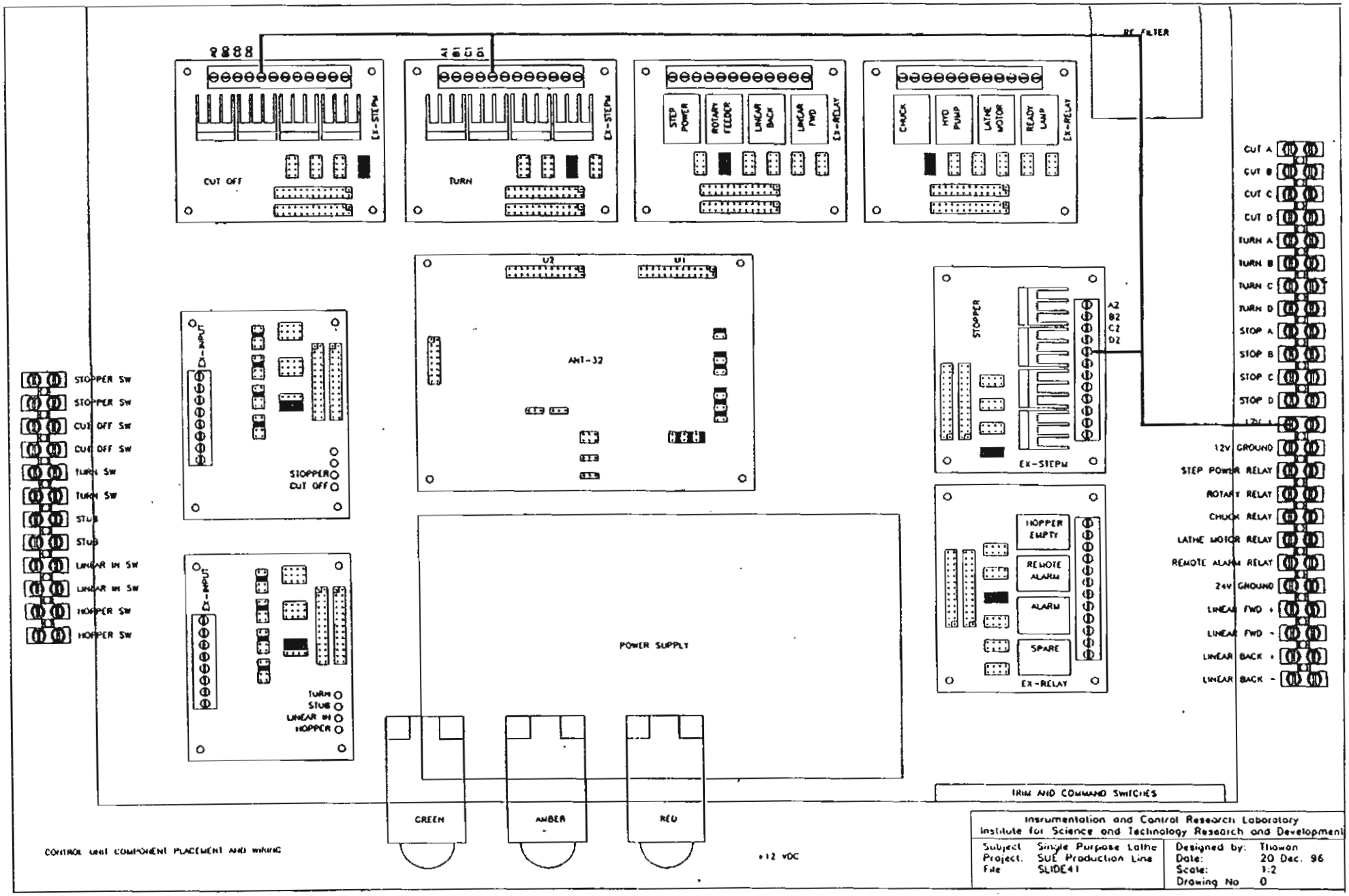
Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

Subject	Single Purpose Lathe	Designed by.	Thawon
Project.	SUE Production Line	Date	20 Dec 96
File:	SLIDE41	Scale.	1:2
		Drawing No.	0



Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

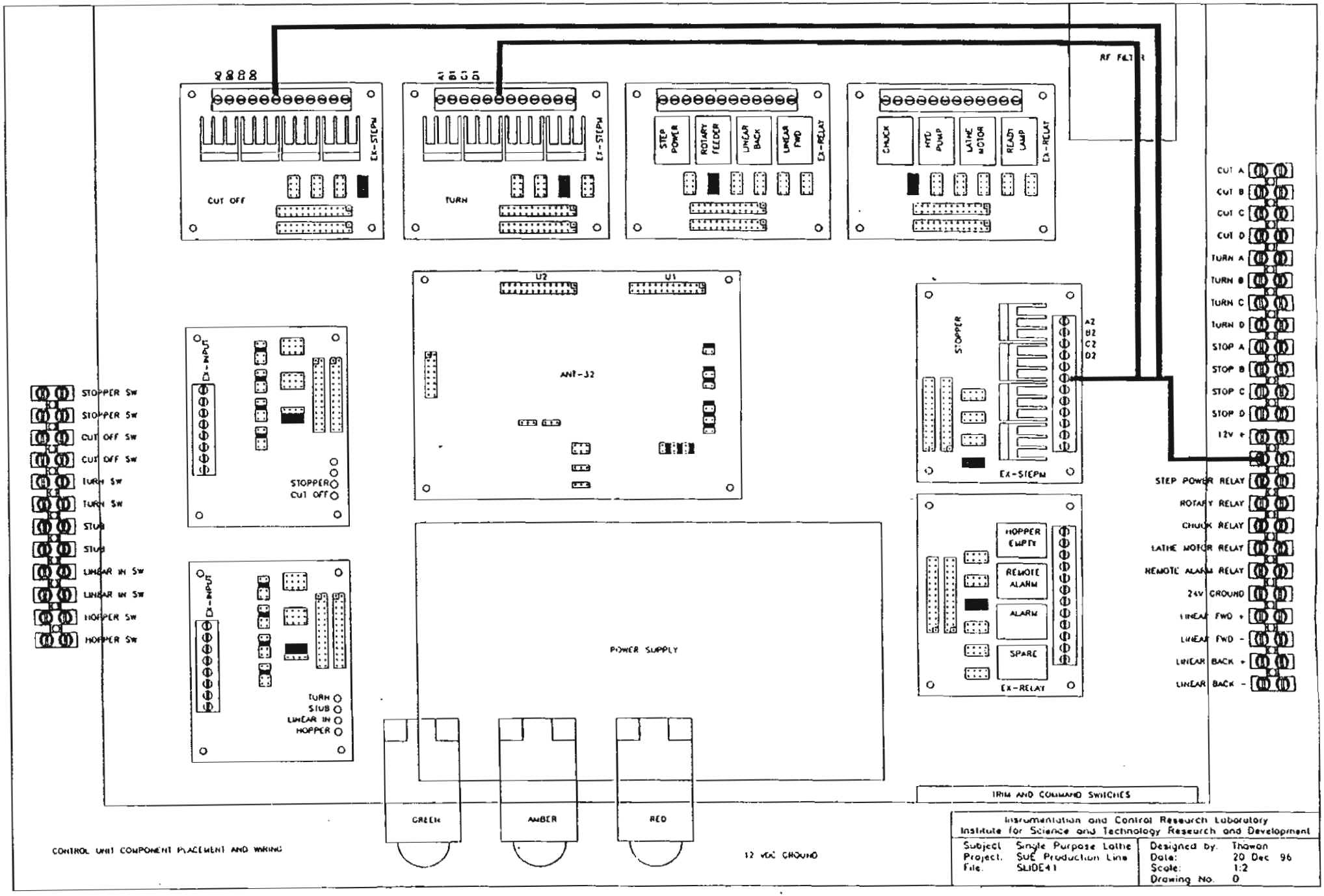
Subject:	Single Purpose Lathe	Designed by:	Thawon
Project:	SUE Production Line	Date:	20 Dec 96
File:	SLIDE41	Scale:	1:2
		Drawing No:	0



CONTROL unit COMPONENT PLACEMENT AND WIRING

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

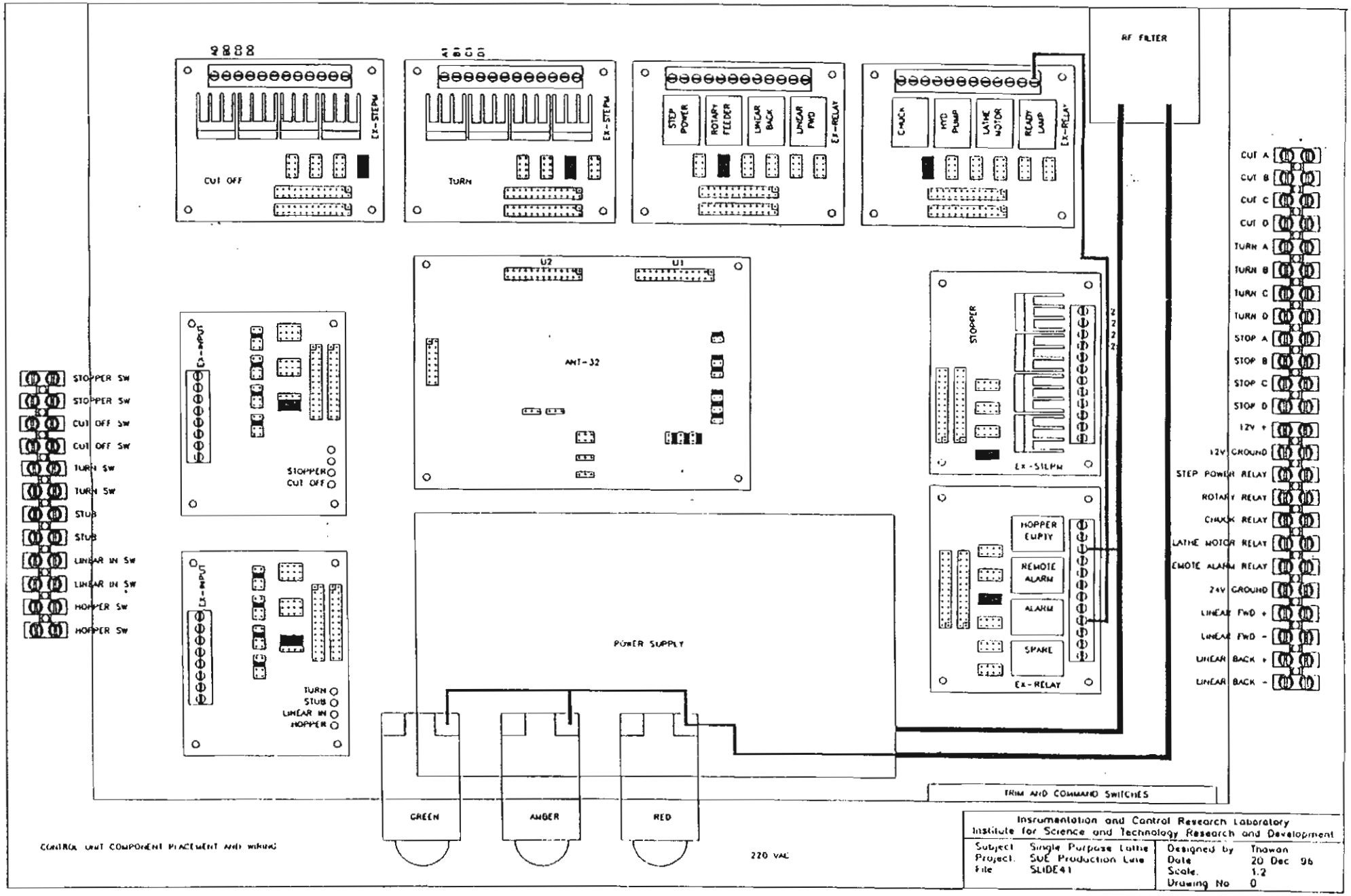
Subject	Single Purpose Lathe	Designed by:	Thawon
Project	SUE Production Line	Date:	20 Dec. 96
File	SLIDE41	Scale:	1:2
		Drawing No	0



CONTROL UNIT COMPONENT PLACEMENT AND WIRING

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

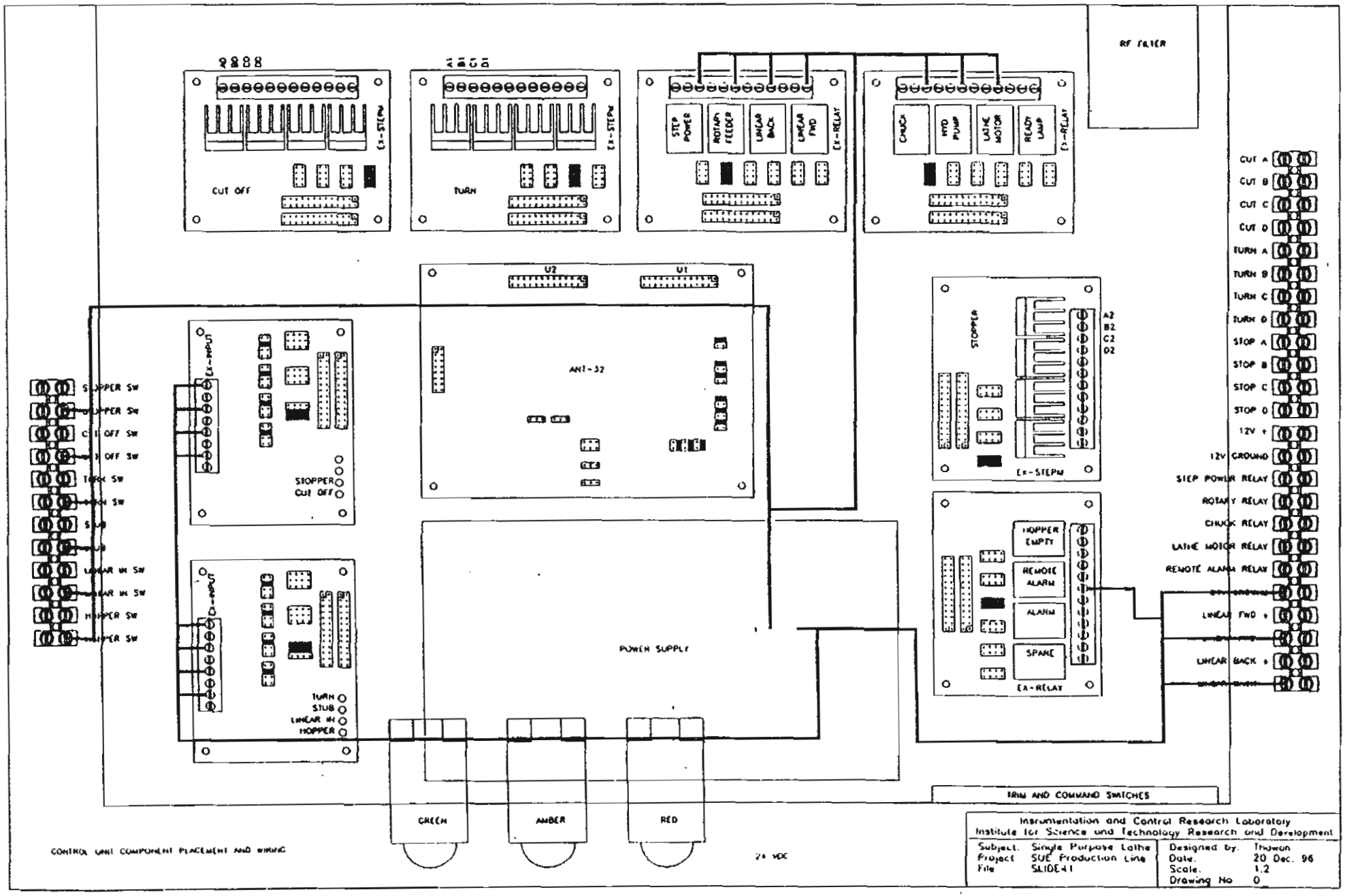
Subject:	Single Purpose Lathe	Designed by:	Thawon
Project:	SUE Production Line	Date:	20 Dec 96
File:	SLIDE41	Scale:	1:2
		Drawing No.	0



CONTROL UNIT COMPONENT PLACEMENT AND WIRING

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

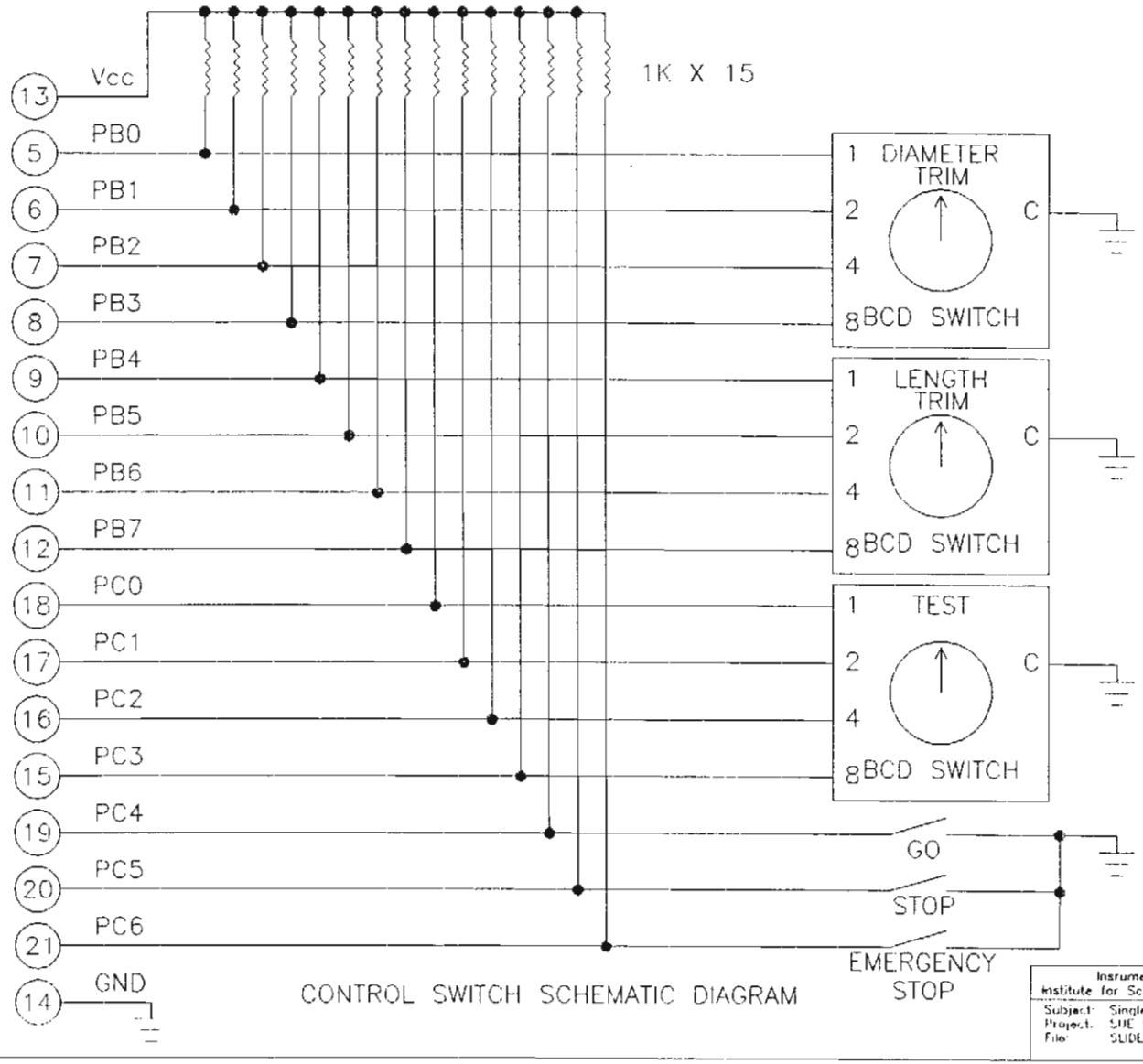
Subject	Single Purpose Lathe	Designed by	Thawan
Project	SUE Production Line	Date	20 Dec 96
File	SLIDE41	Scale	1:2
		Drawing No	0



CONTROL UNIT COMPONENT PLACEMENT AND WIRING

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

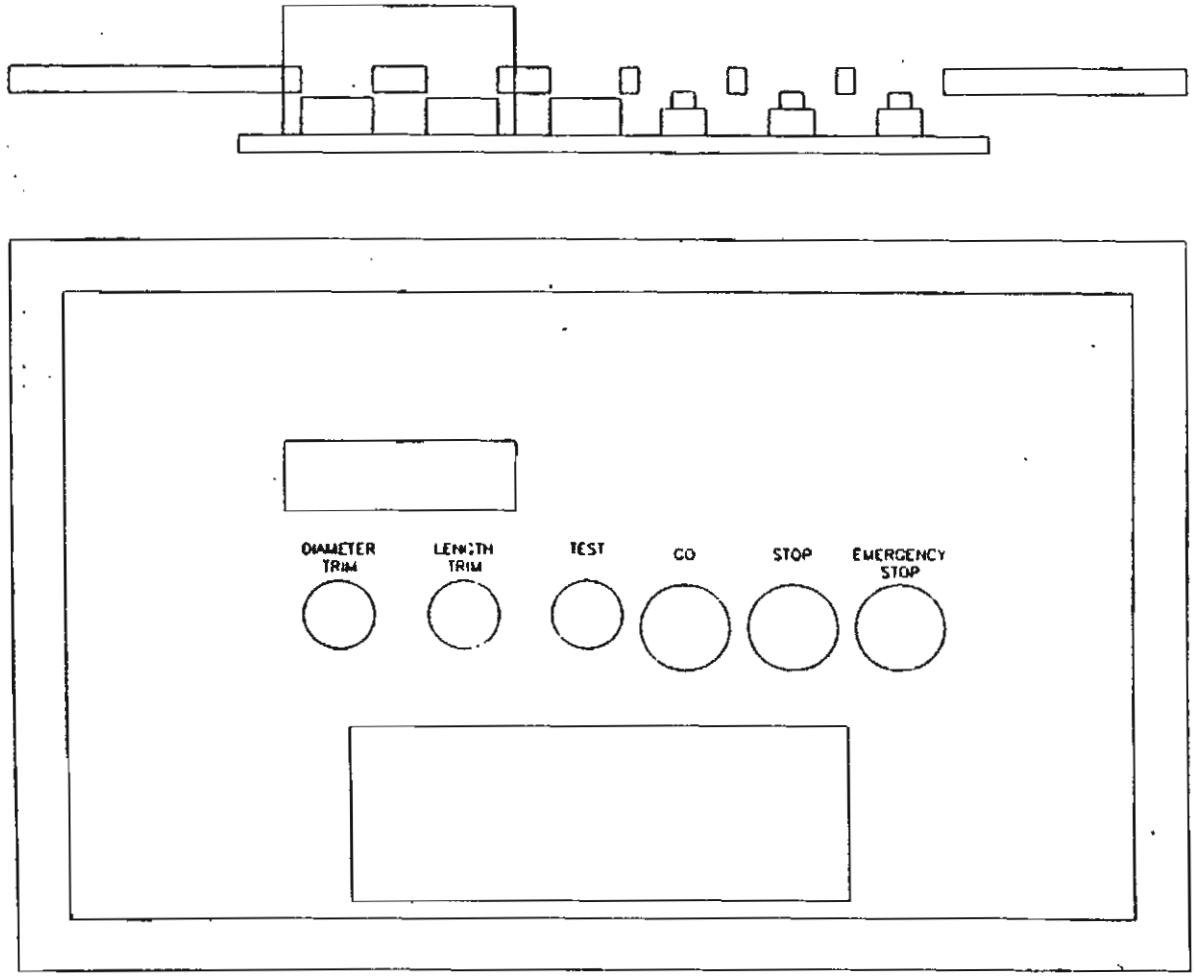
Subject.	Single Purpose Lathe	Designed by.	Thawon
Project	SUE Production Line	Date.	20 Dec. 96
File	SLIDE41	Scale.	1:2
		Drawing No	0



CONTROL SWITCH SCHEMATIC DIAGRAM

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

Subject:	Single Purpose Lathe	Designed by	Thawan
Project:	SDE Production Line	Date:	15 Jun 97
File:	SLIDE40A	Scale:	1:1
		Drawing No:	0

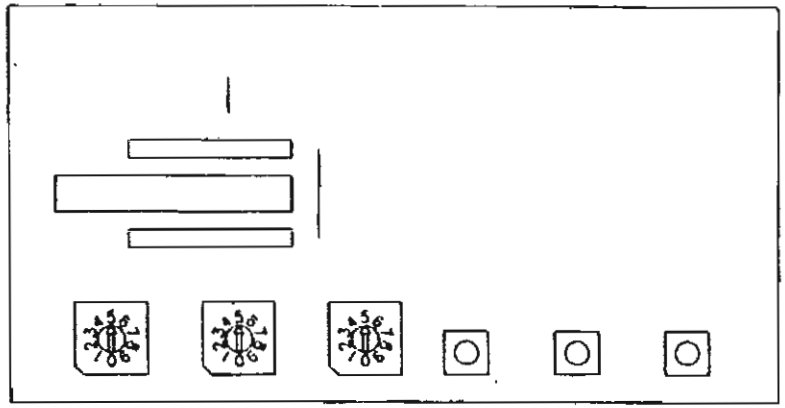


CONTROL SWITCH

FRONT PANNAL

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

Subject:	Single Purpose Lathe	Designed by:	Thawon
Project:	SUE Production Line	Date:	26 Oct. 98
File:	SLIDE40	Scale:	1:1
		Drawing No	0

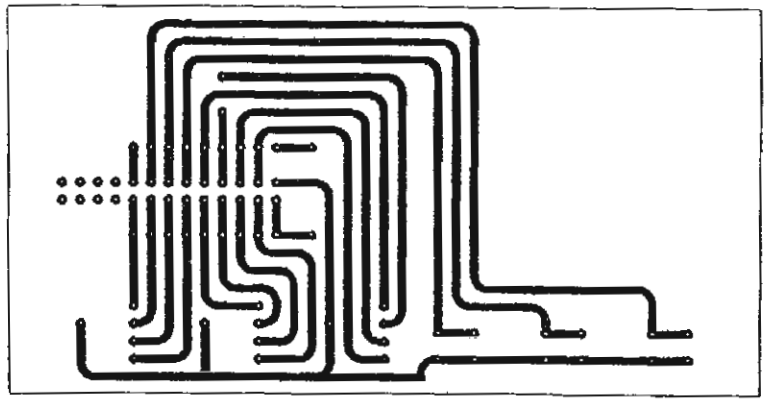


CONTROL SWITCH

PART LAYOUT

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

Subject:	Single Purpose Lathe	Designed by:	Thawan
Project:	SUE Production Line	Date:	26 Oct. 96
File:	SLIDE40	Scale:	1:1
		Drawing No.	0



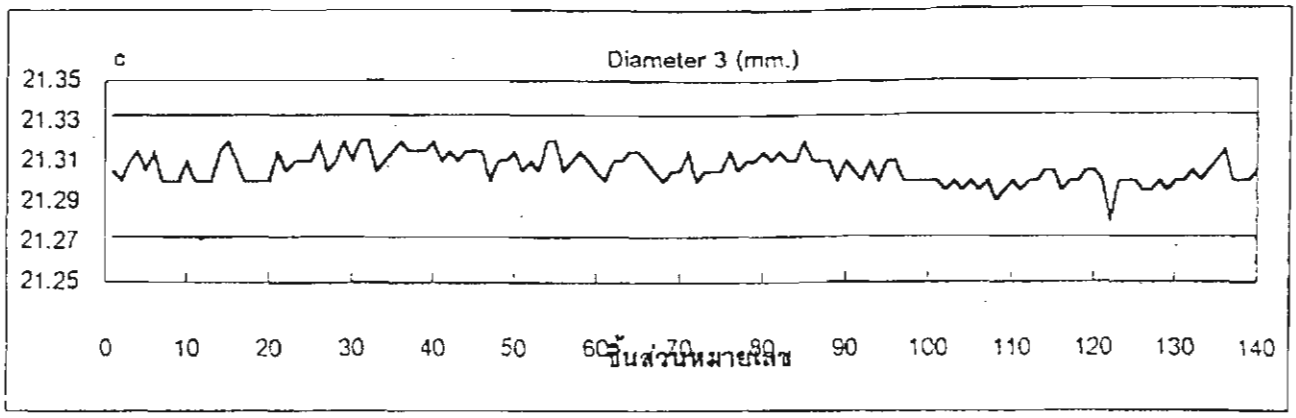
CONTROL SWITCH

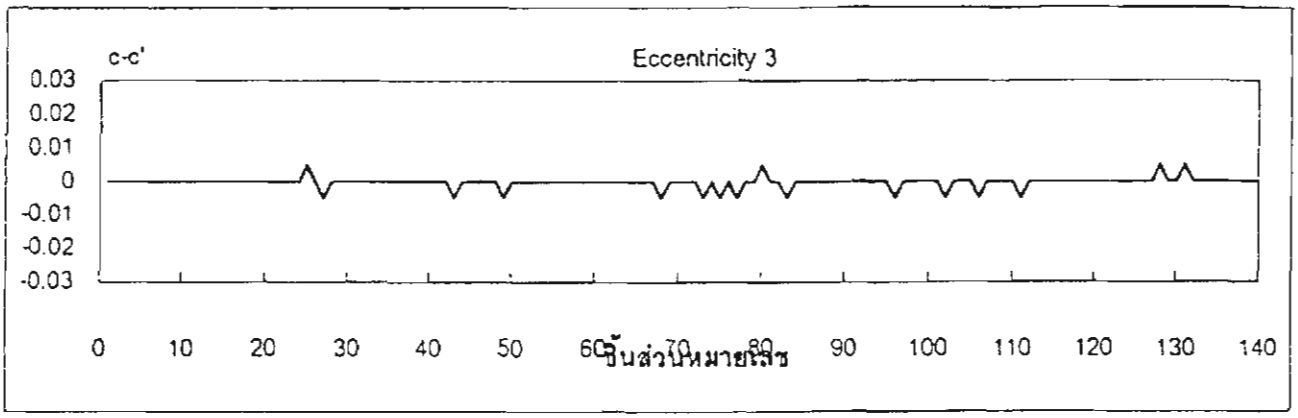
PCB

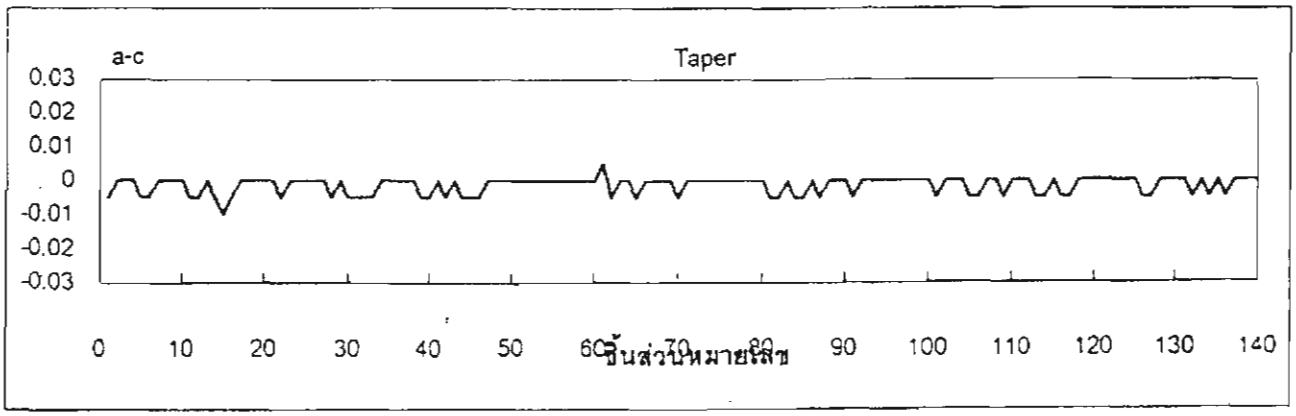
Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

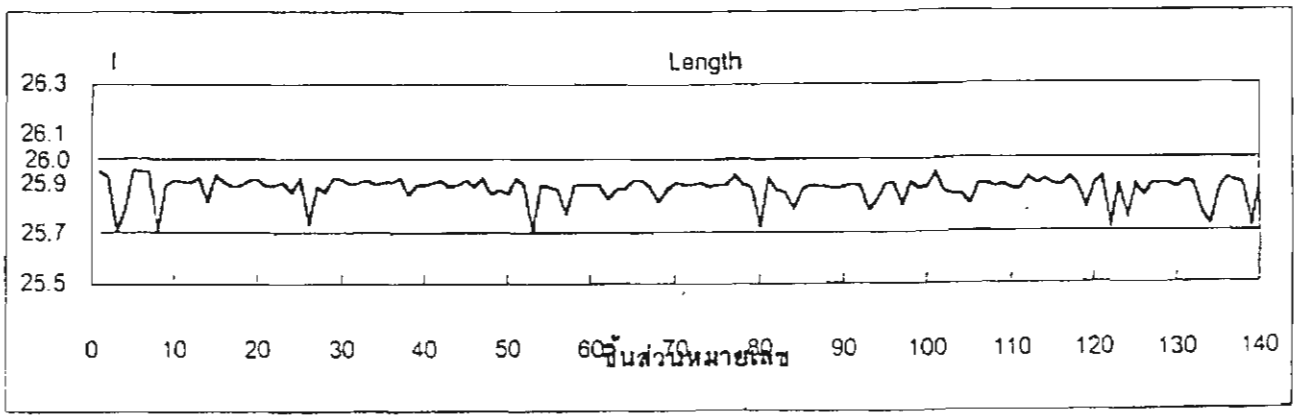
Subject	Single Purpose Lathe	Designed by:	Thawan
Project	SUE Production Line	Date:	26 Oct. 96
File:	SLIDE40	Scale:	1:1
		Drawing No.	0

Appendix F. Test Result Run 1 (continue).

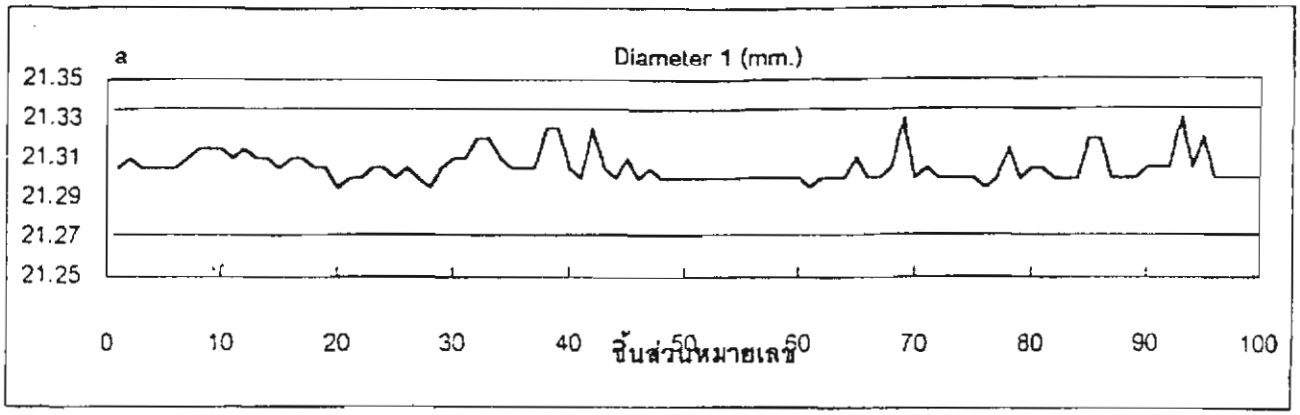


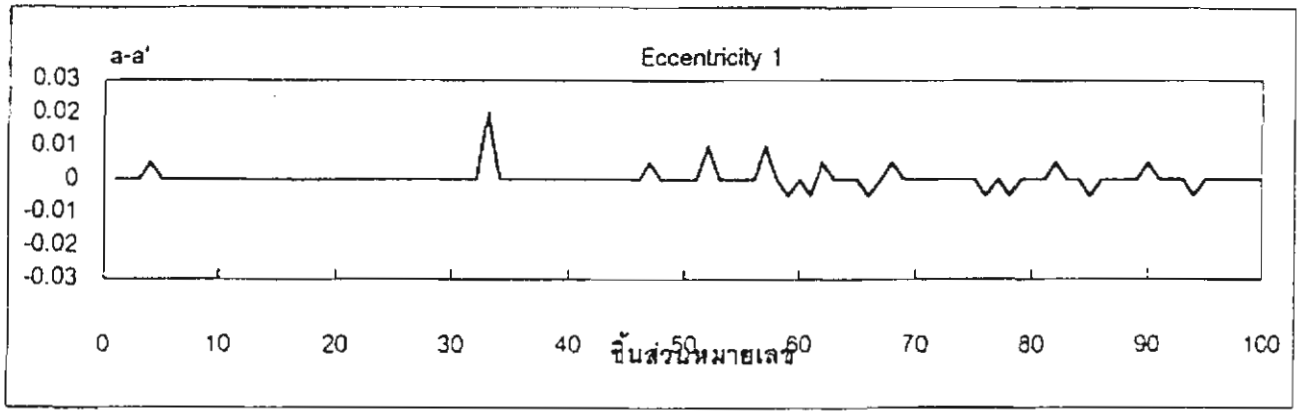


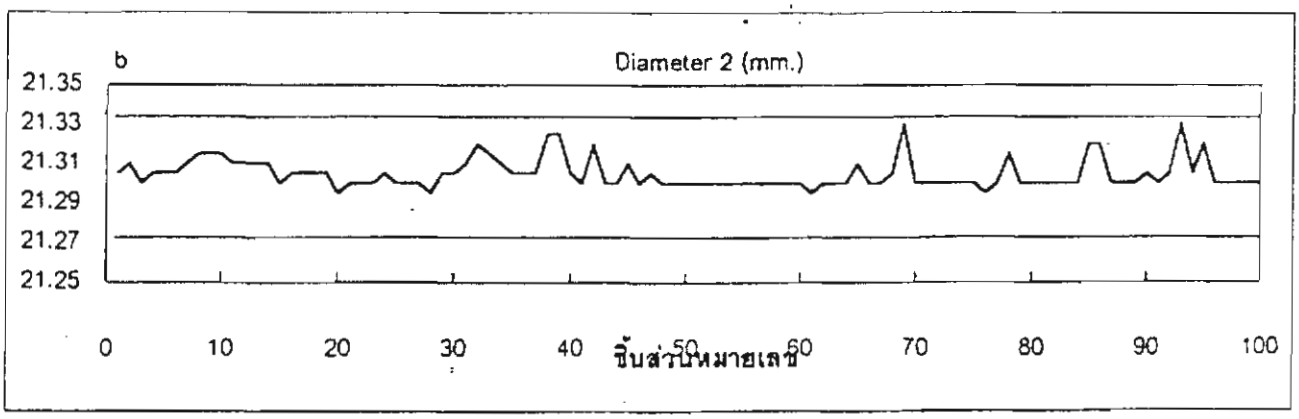


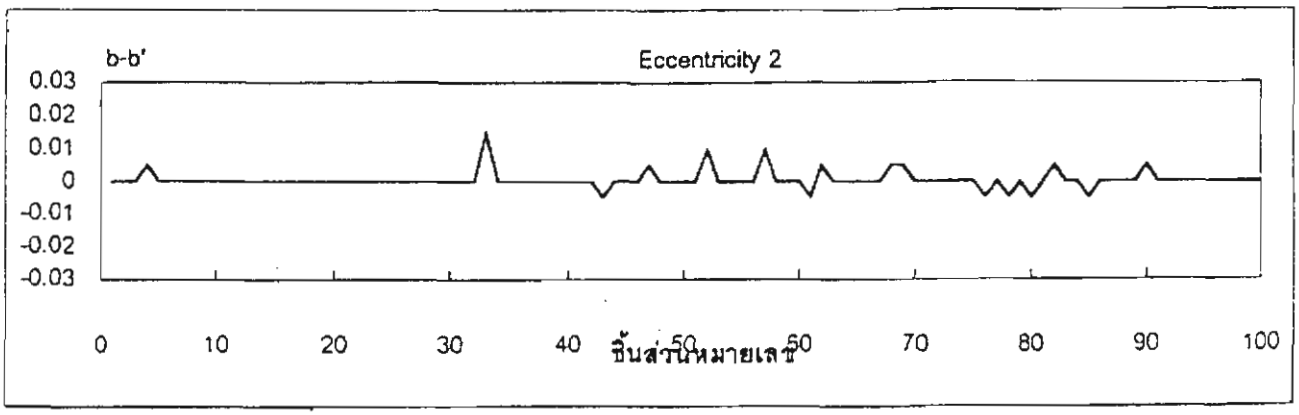


Appendix F. Test Result Run 2.

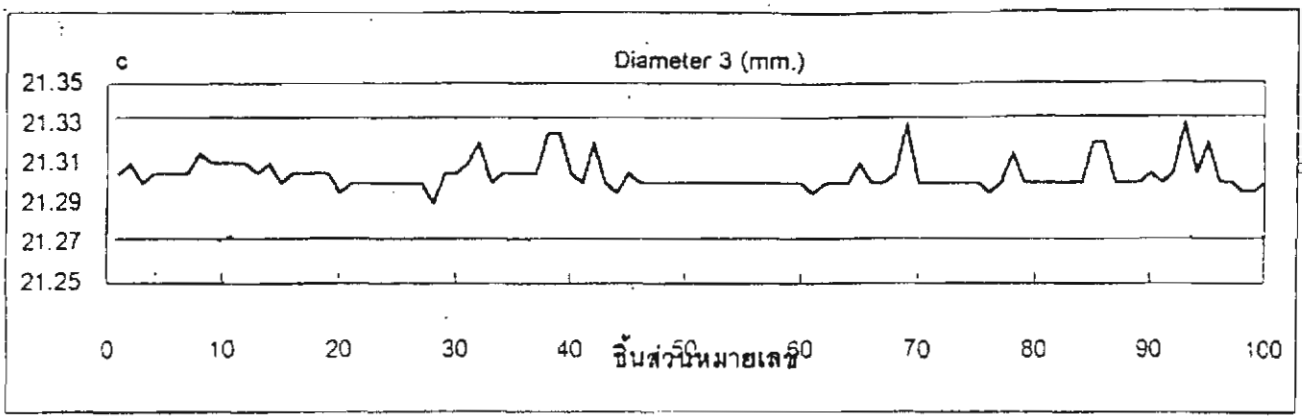


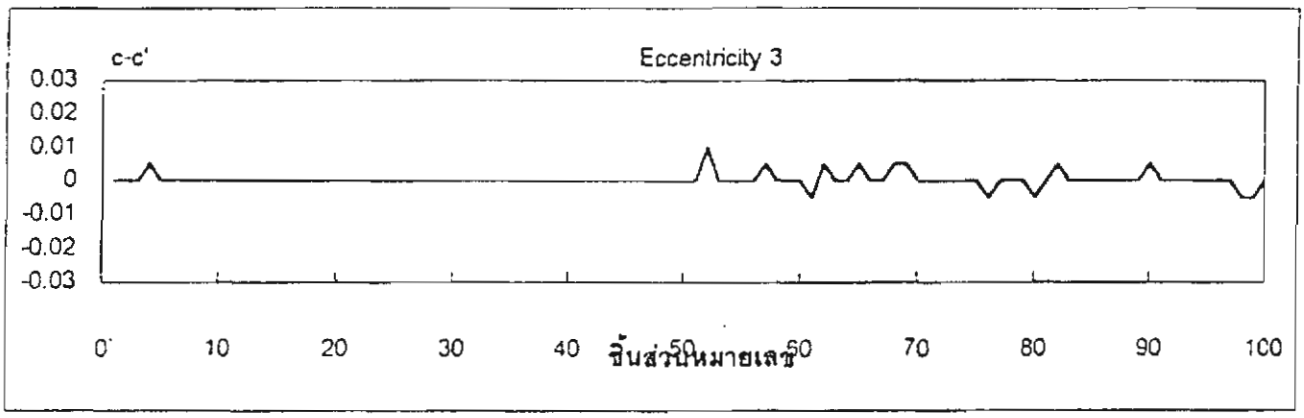


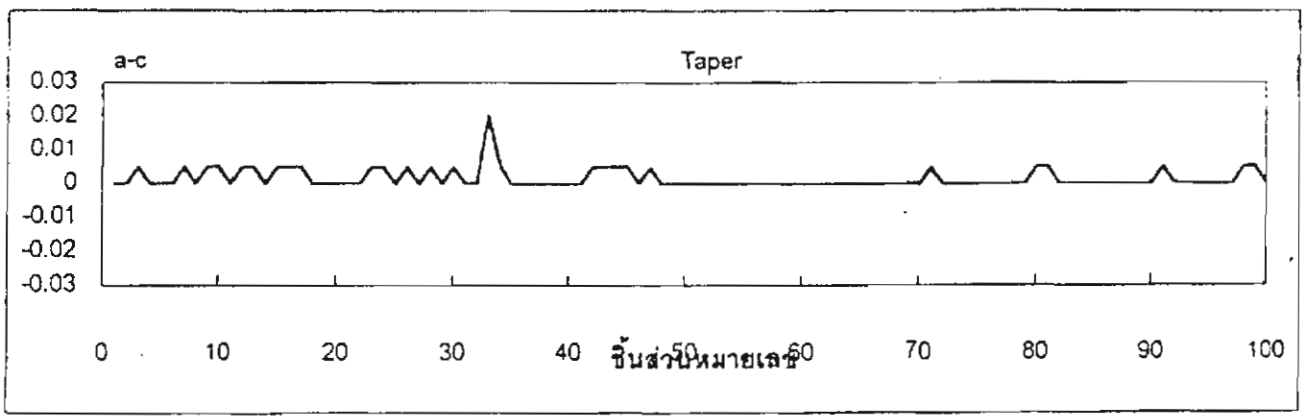


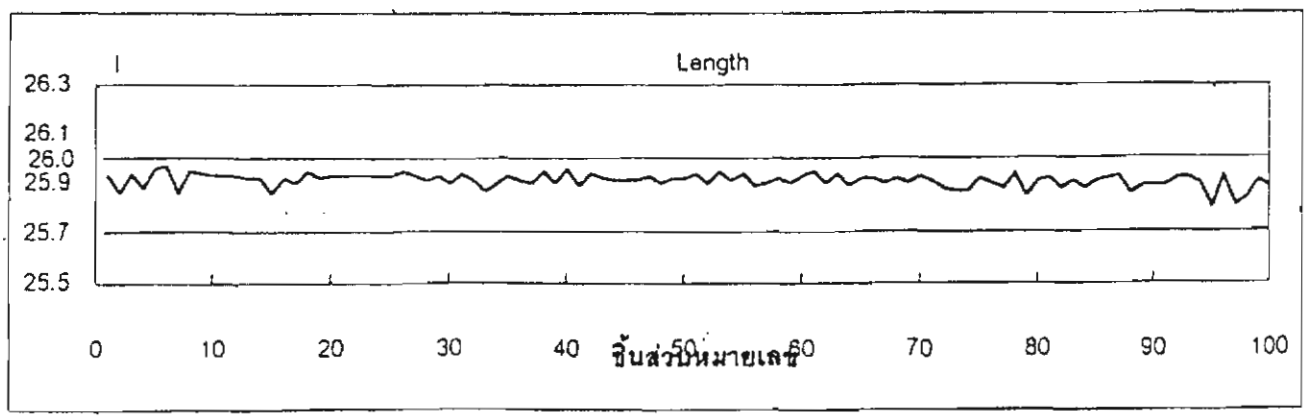


Appendix F. Test Result Run 2 (continue).

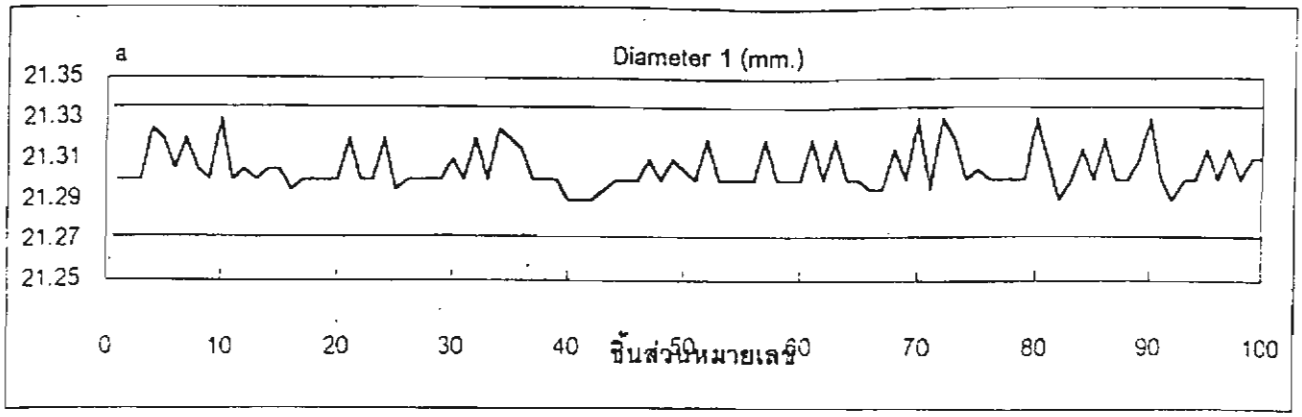


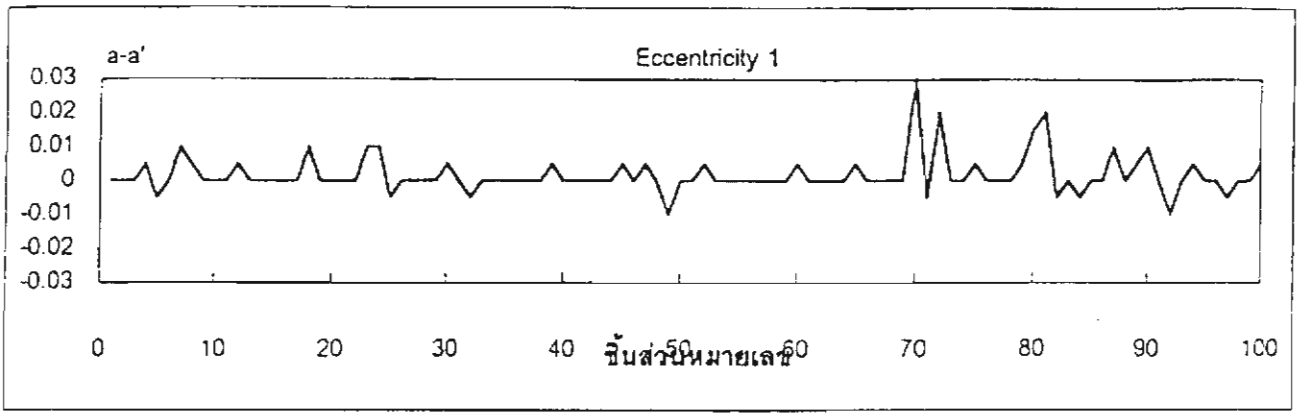


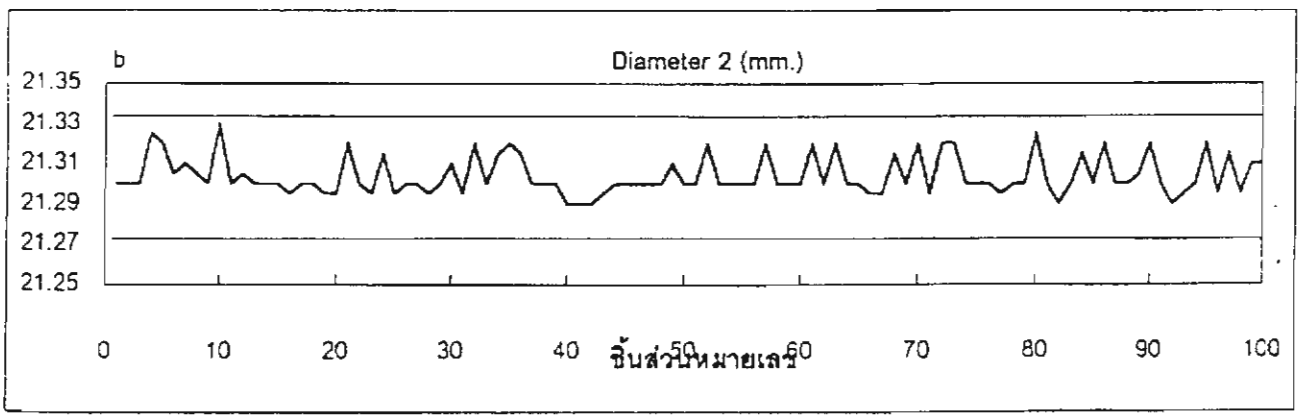


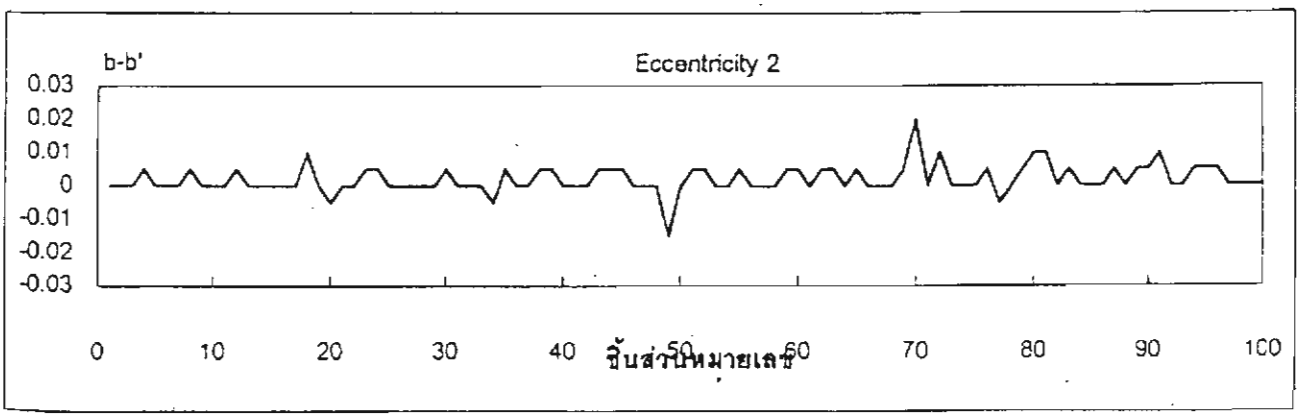


Appendix F. Test Result Run 3.

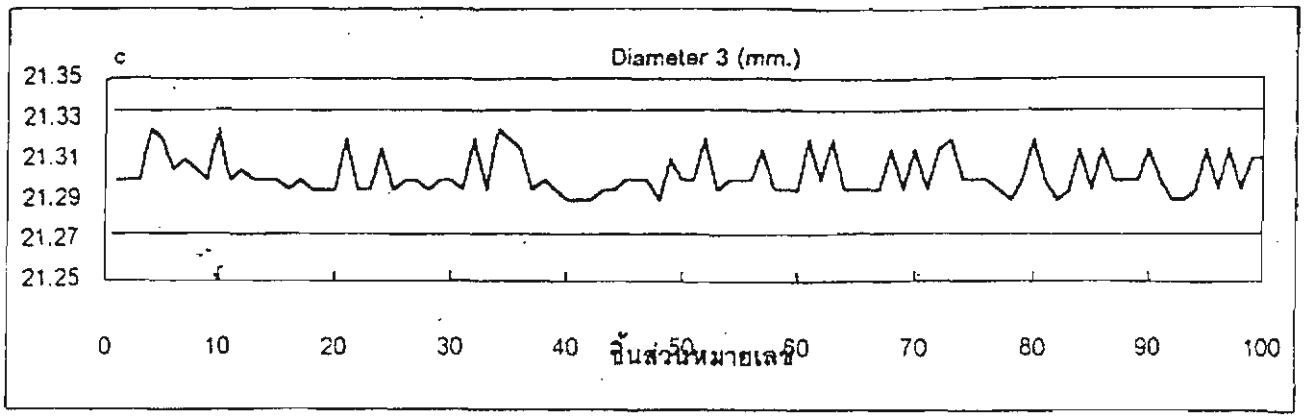


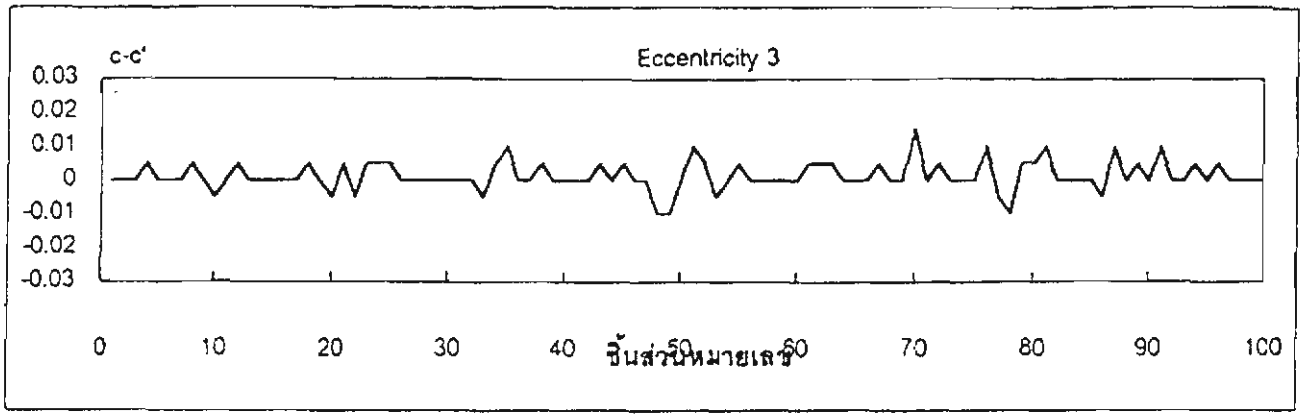


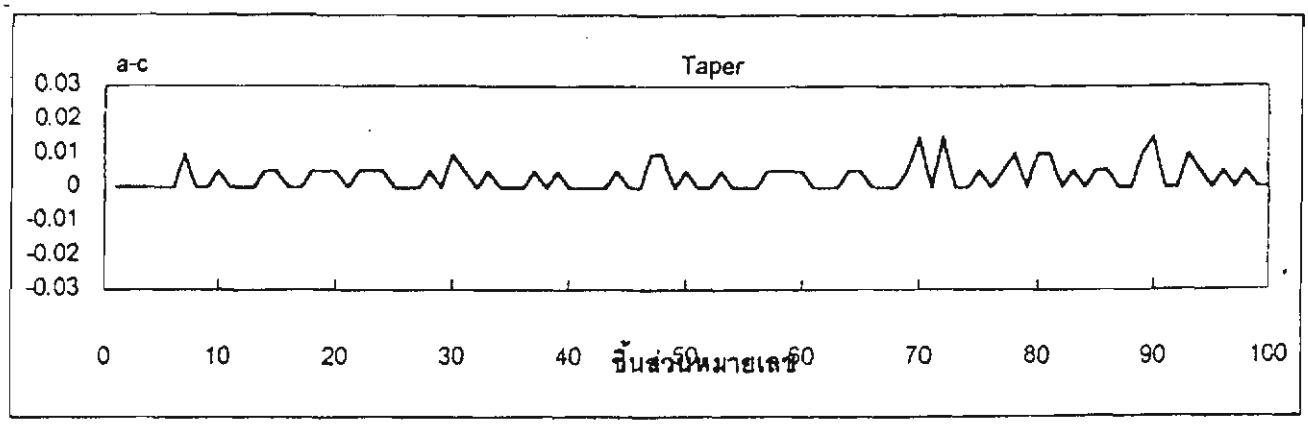


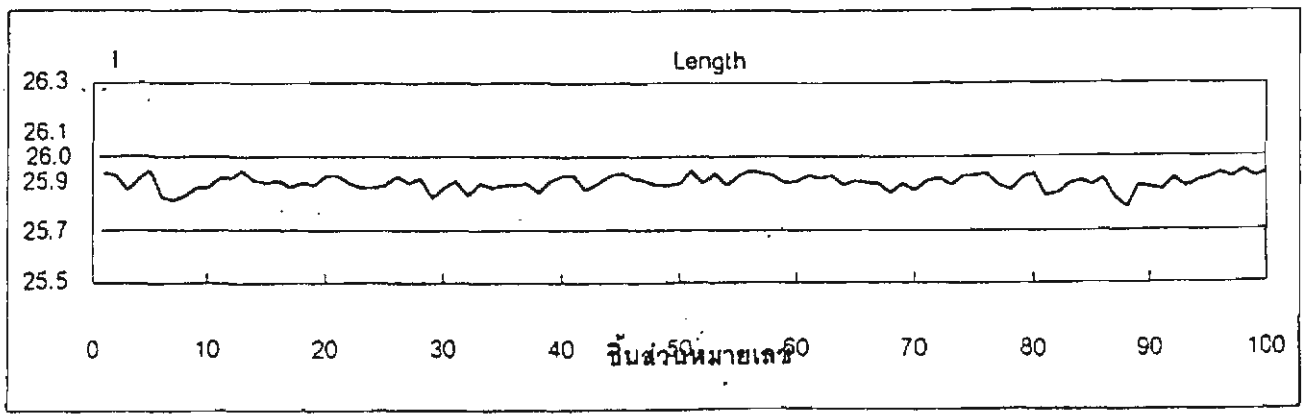


Appendix F. Test Result Run 3 (continue).

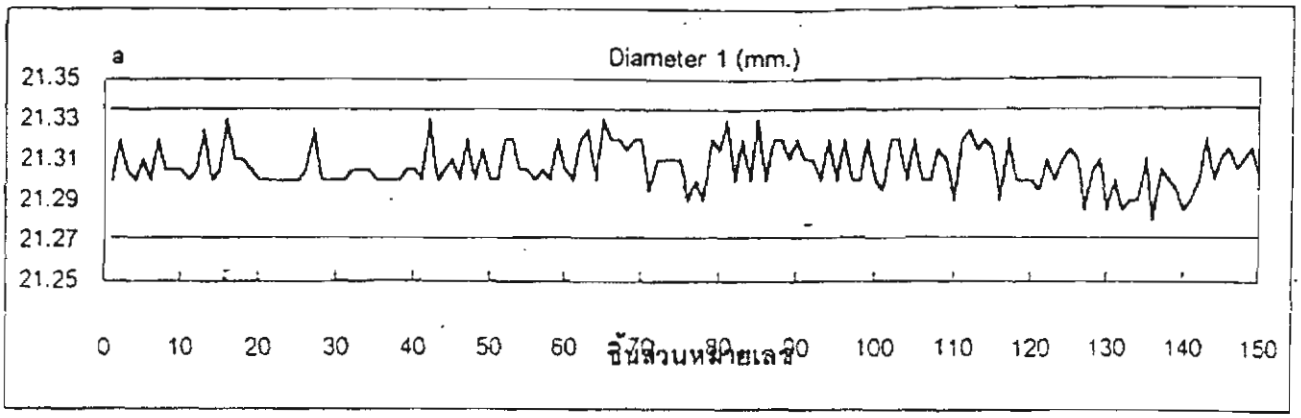


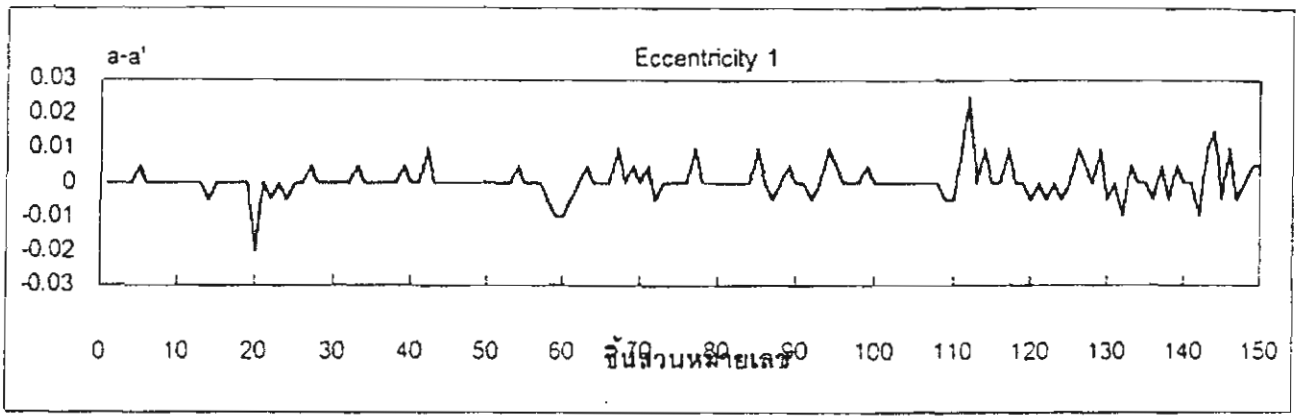


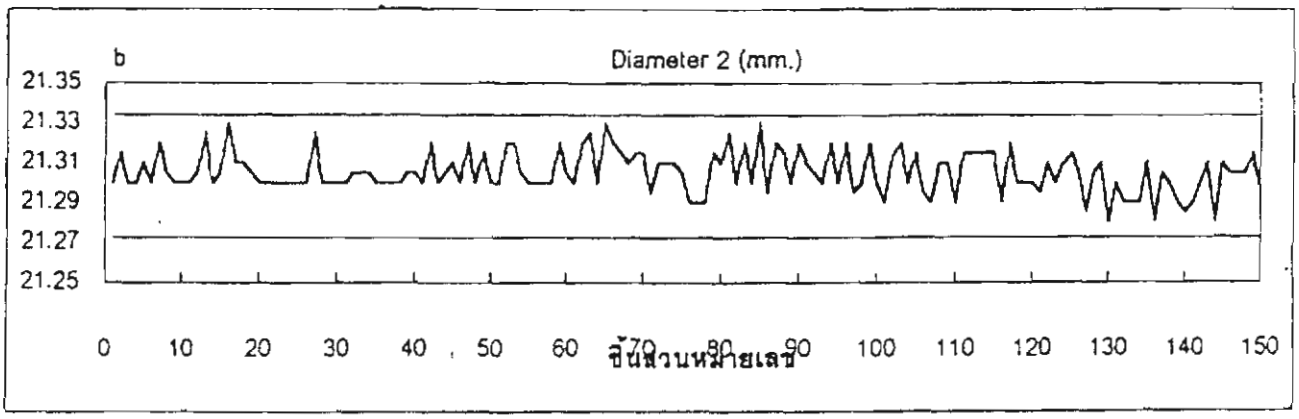


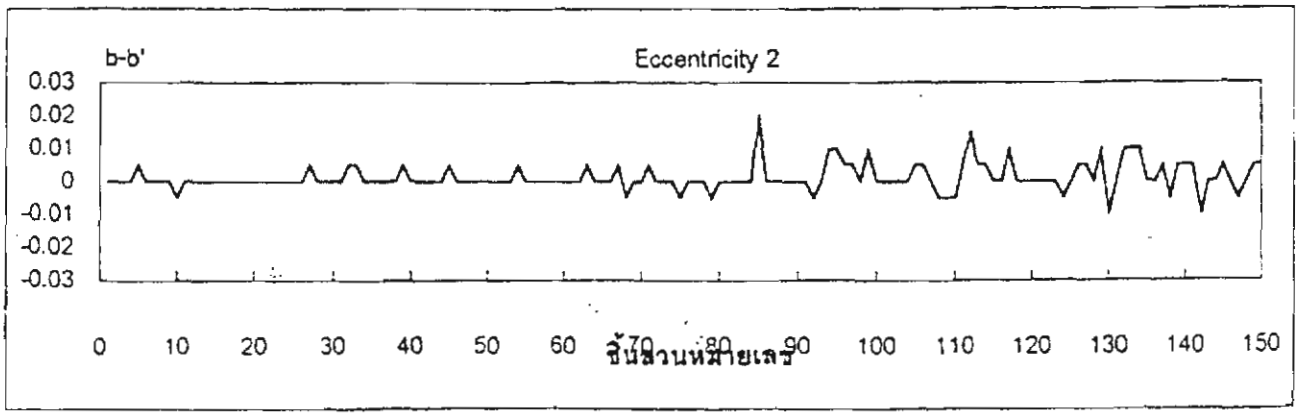


Appendix F. Test Result Run 4.

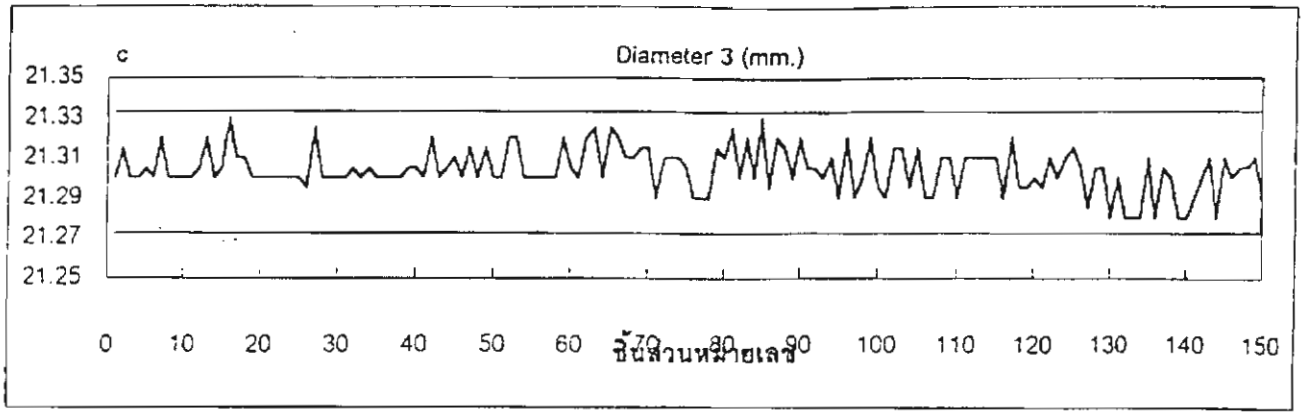


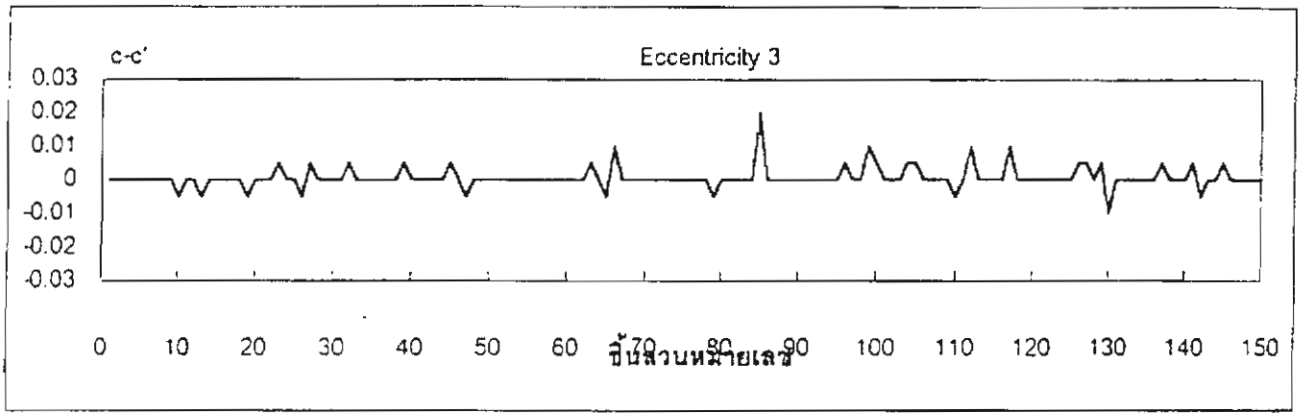


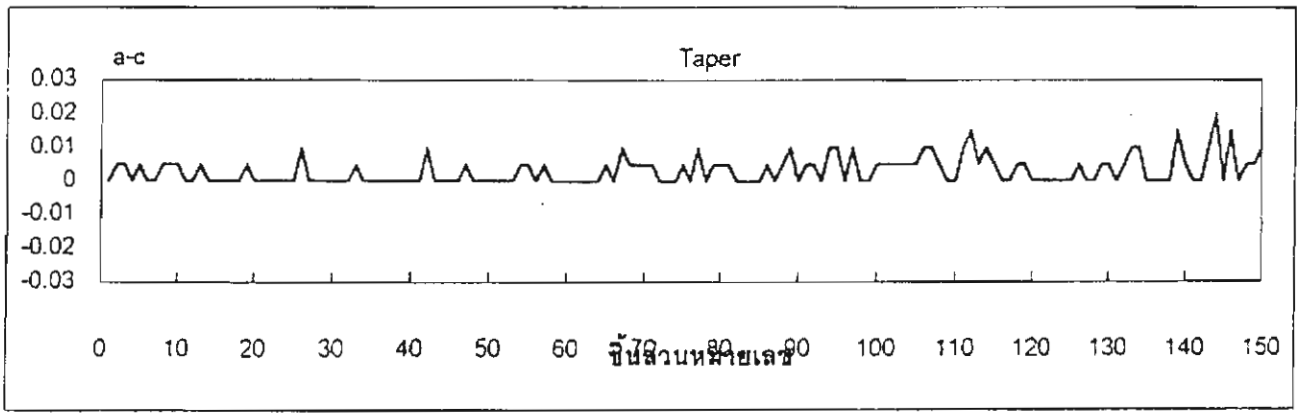


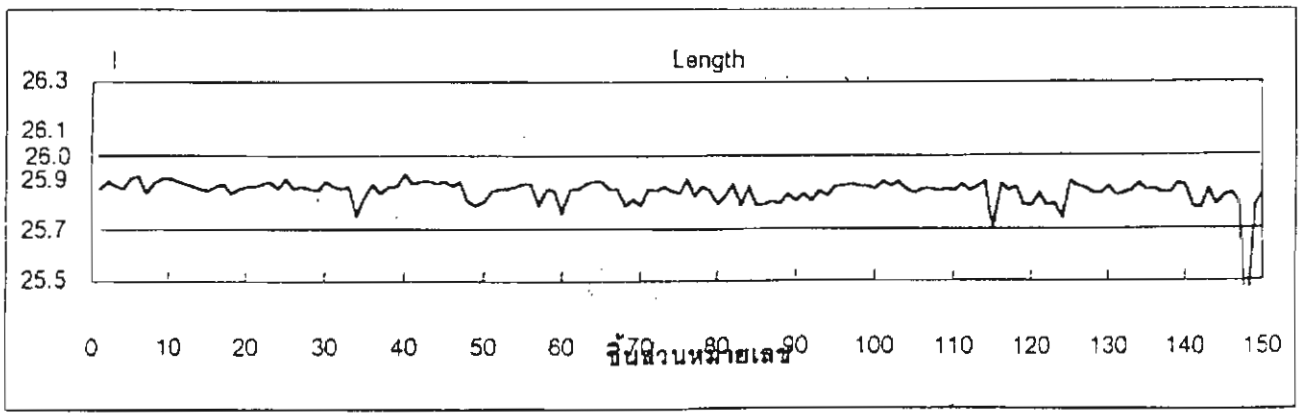


Appendix F. Test Result Run 4 (continue).



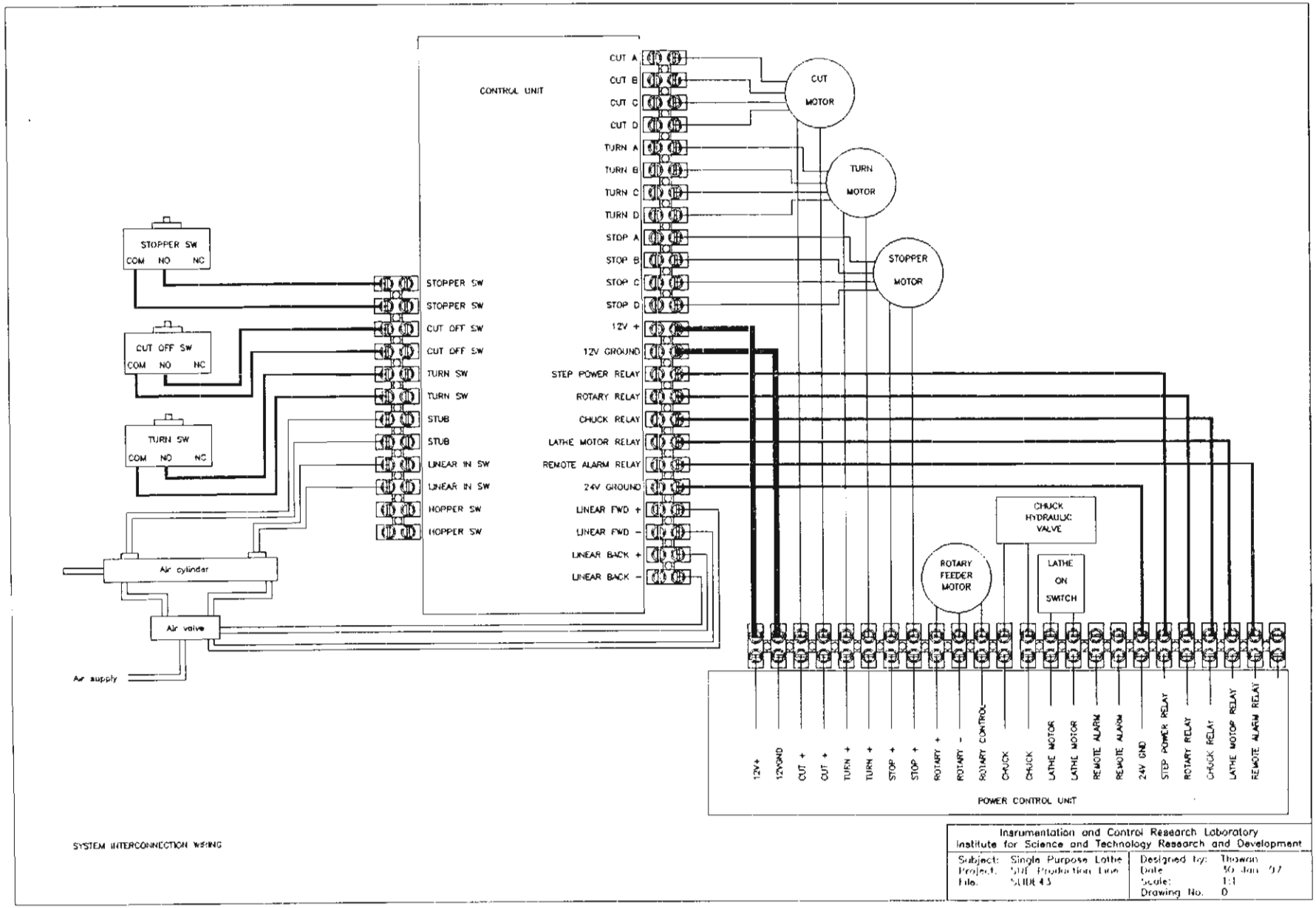






Appendix G. Drawings.

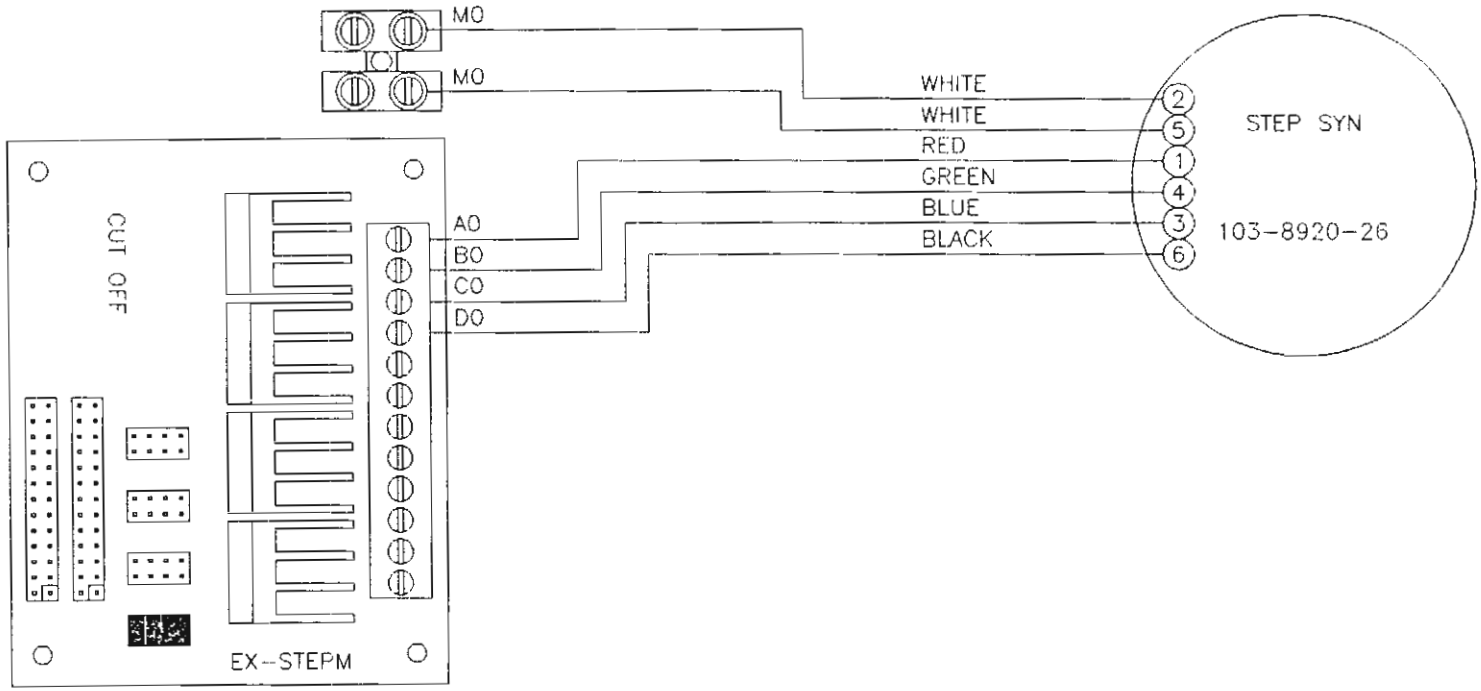
The technical drawings for the control system and mechanical parts are in the following pages:



SYSTEM INTERCONNECTION WIRING

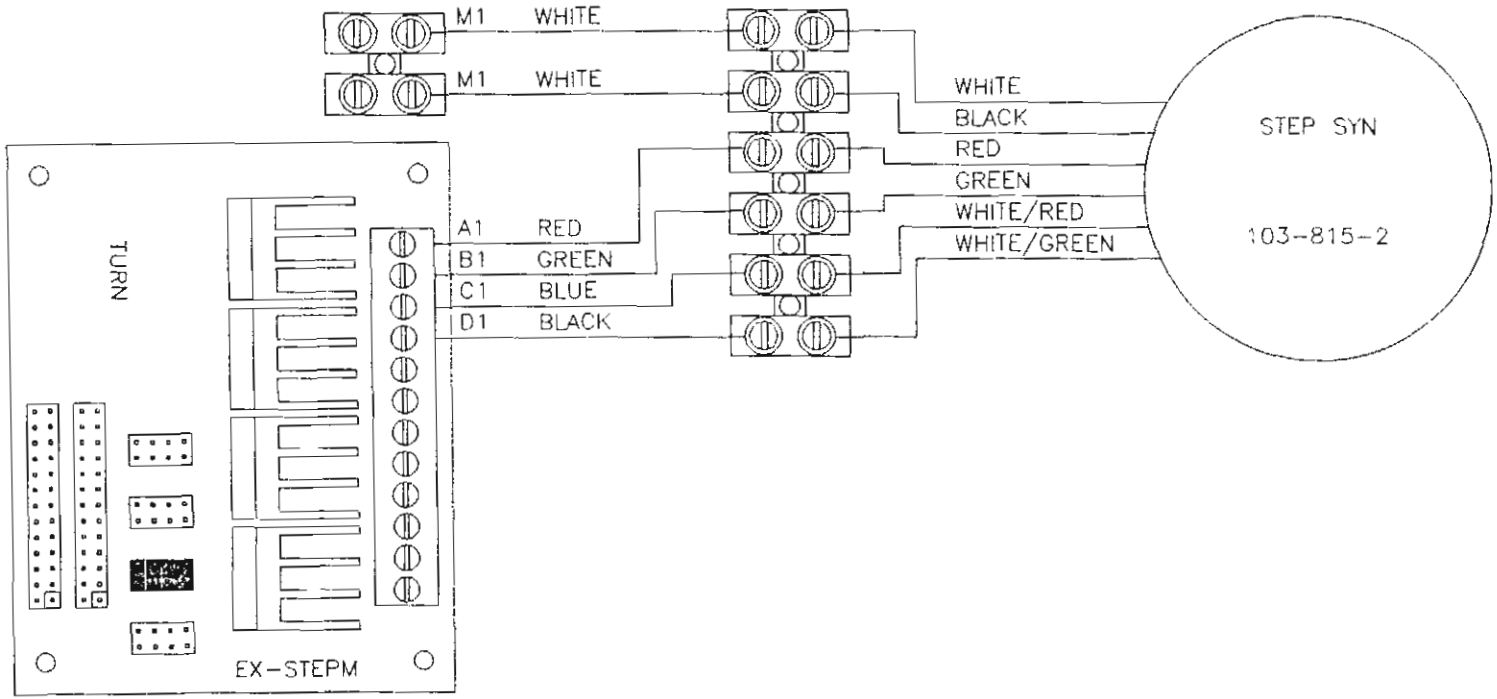
Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

Subject: Single Purpose Lathe	Designed by: Thawan
Project: SPL Production Line	Date: 30 Jan '97
File: SLIDE43	Scale: 1:1
	Drawing No. 0



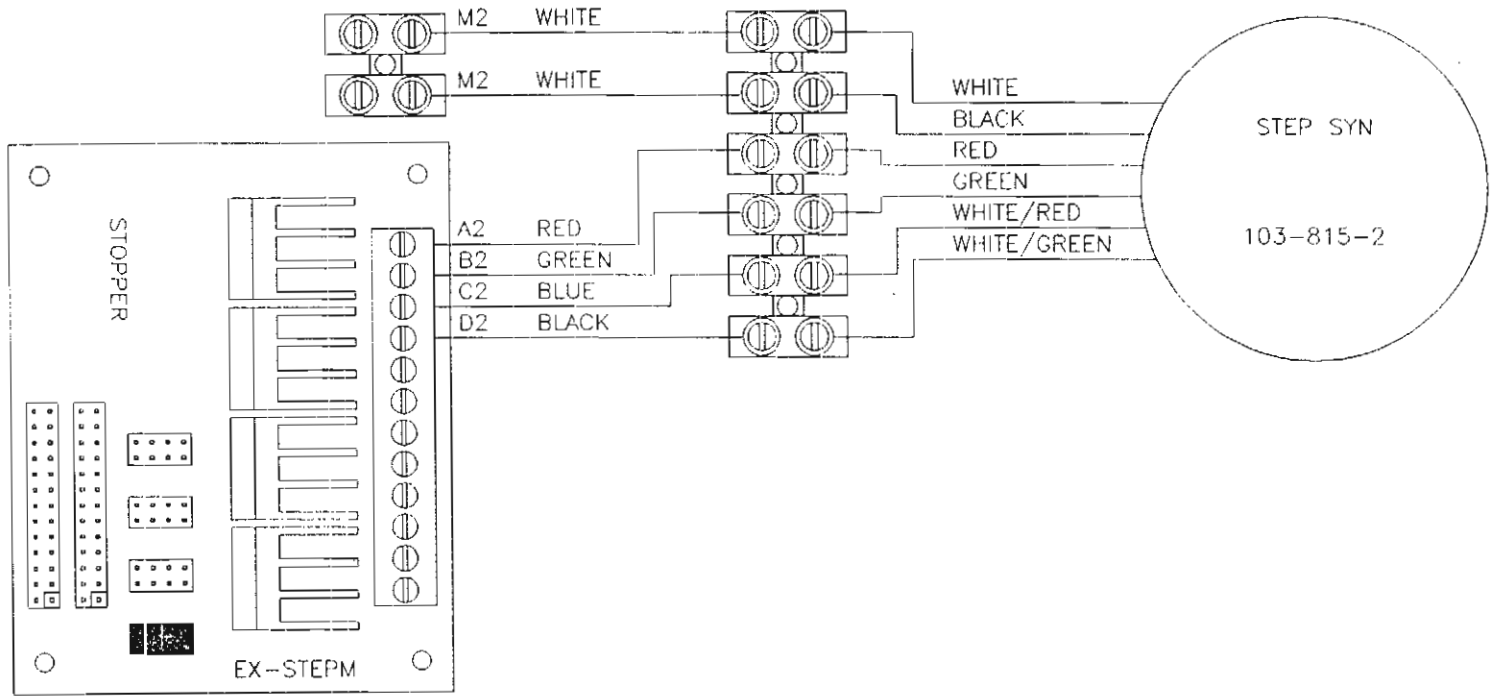
CROSS SLIDE MOTOR WIRING

Instrumentation and Control Research Laboratory Institute for Science and Technology Research and Development	
Subject: Single Purpose Lathe Project: SUF Production Line File: SLIDE31	Designed by: Thawan Date: 28 Oct. 96 Scale: 1:1 Drawing No 0



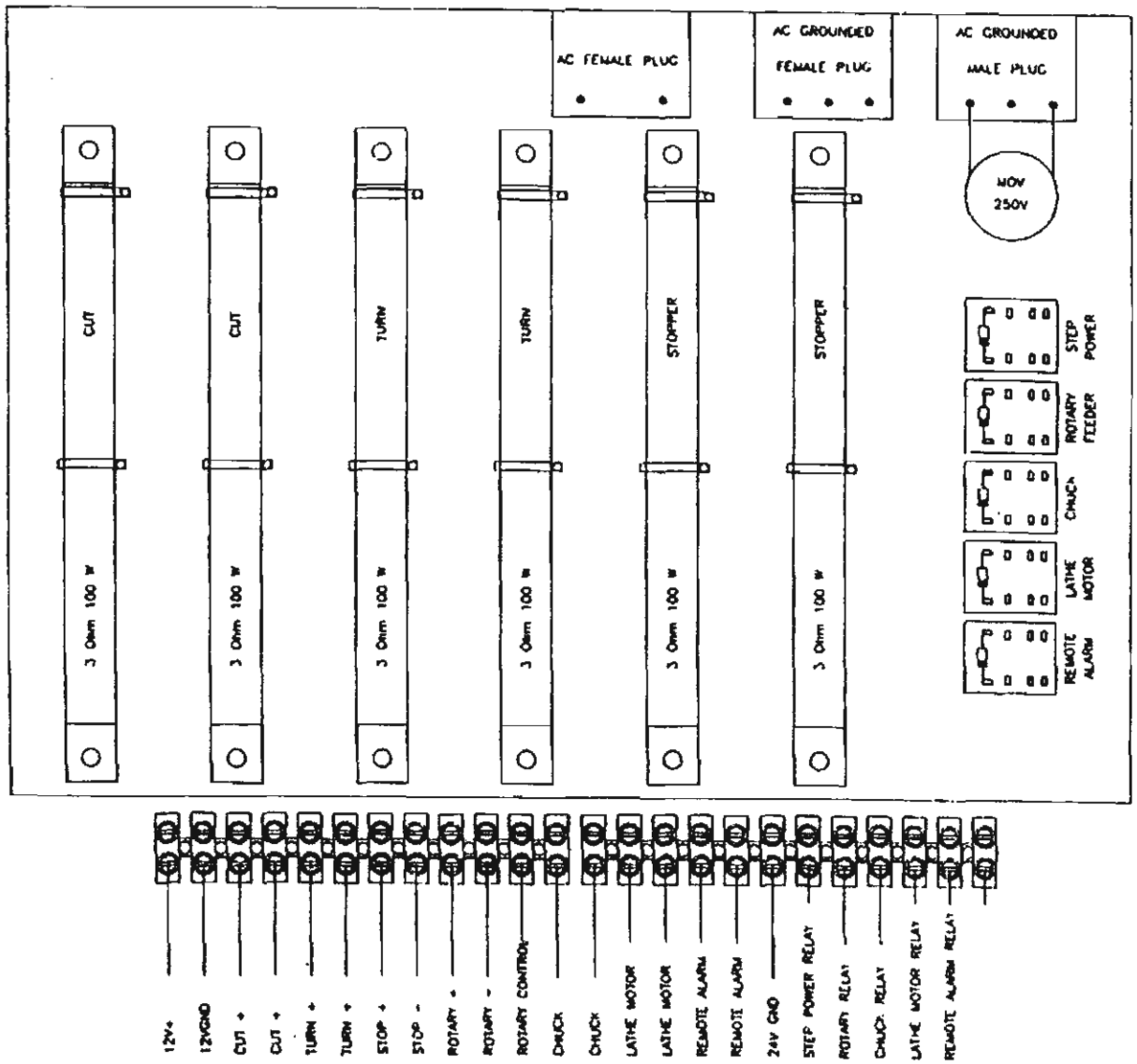
TURN SLIDE MOTOR WIRING

Instrumentation and Control Research Laboratory Institute for Science and Technology Research and Development			
Subject:	Single Purpose Lathe	Designed by:	Thawan
Project:	SUF Production Line	Date	26 Oct 96
File:	SLIDE32	Scale:	1:1
		Drawing No	0



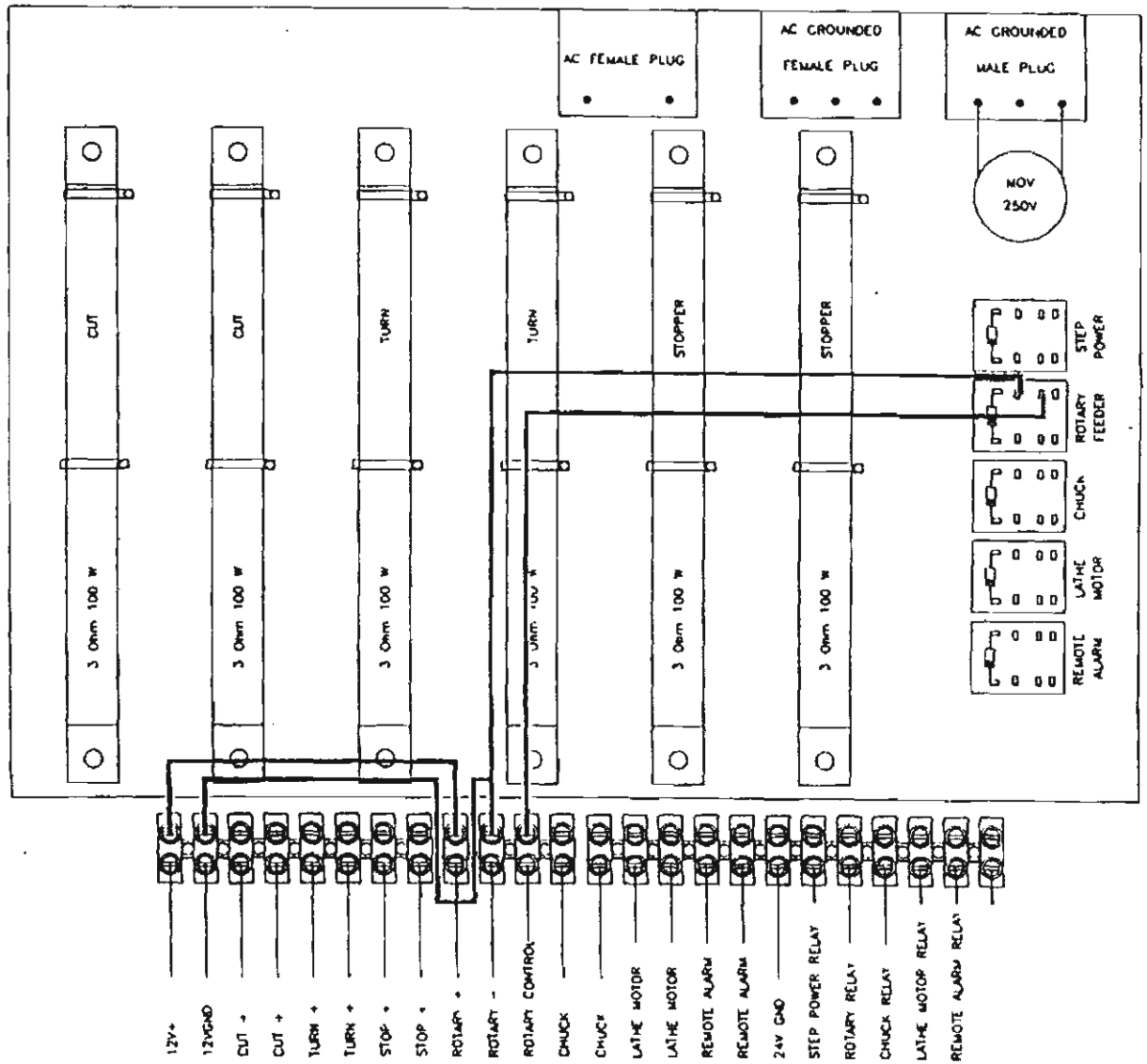
STOPPER MOTOR WIRING

Instrumentation and Control Research Laboratory Institute for Science and Technology Research and Development	
Subject: Single Purpose Lathe Project: SUE Production Line File: SLIDE33	Designed by: Thawon Date: 26 Oct 96 Scale: 1:1 Drawing No: 0



POWER CONTROL UNIT

Instrumentation and Control Research Laboratory Institute for Science and Technology Research and Development			
Subject:	Single Purpose Lathe	Designed by:	Thawon
Project:	SUE Production Line	Date:	26 Oct. 96
File:	SLIDE42	Scale:	1:2
		Drawing No.	0

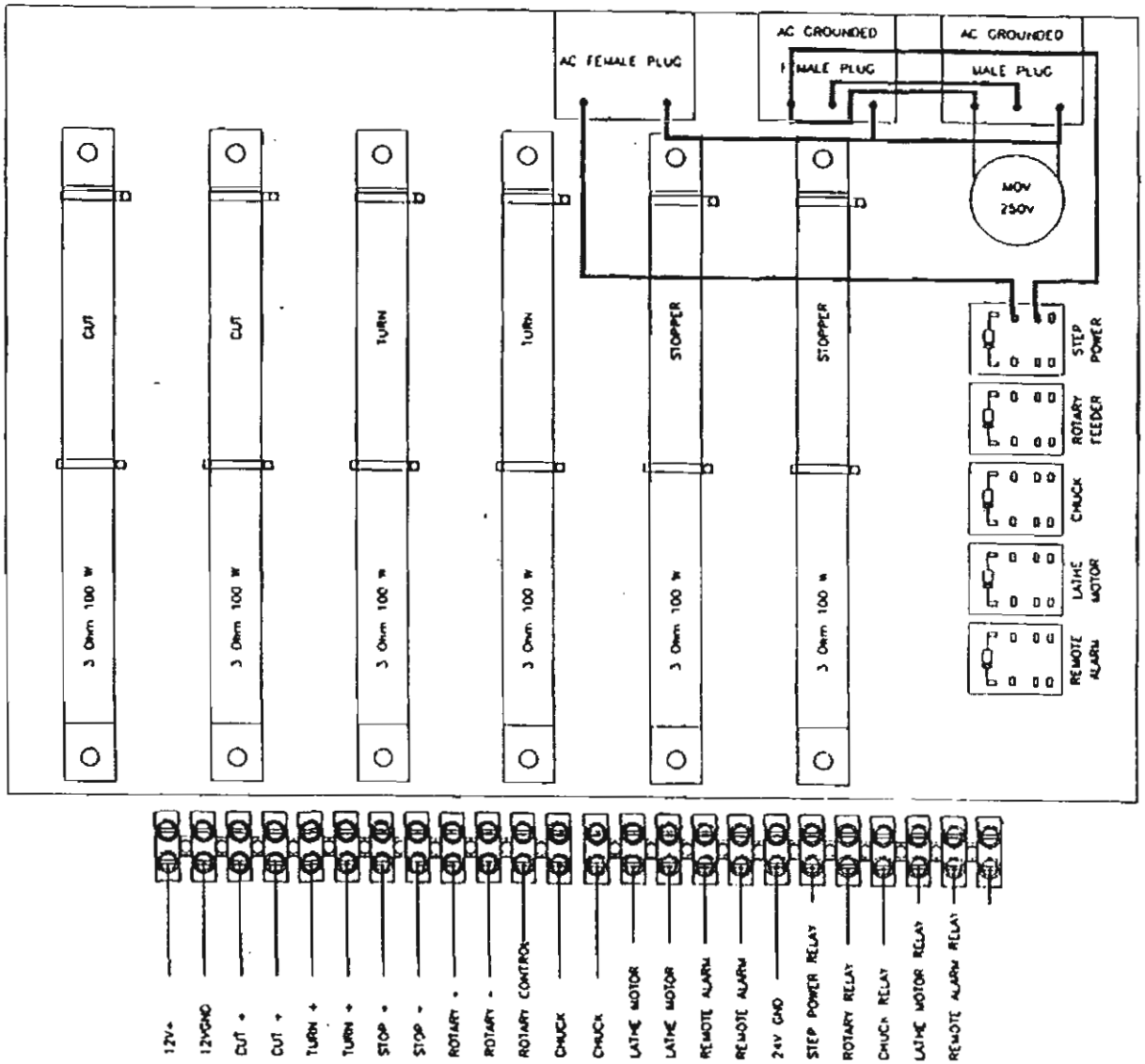


POWER CONTROL UNIT

12V

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

Subject:	Single Purpose Lathe	Designed by:	Thawon
Project:	SUE Production Line	Date:	26 Oct 96
File:	SLIDE42	Scale:	1:2
		Drawing No	0

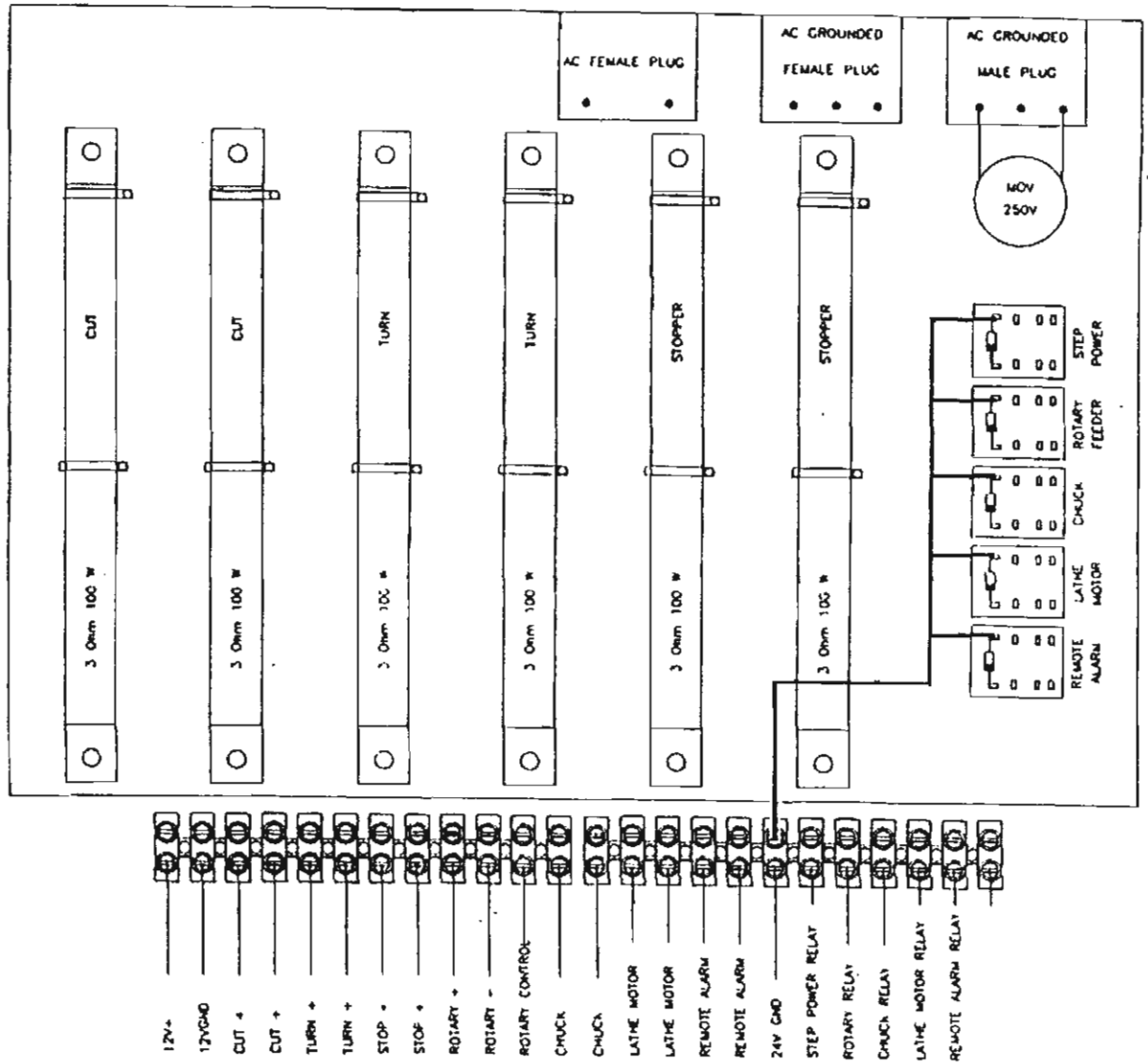


POWER CONTROL Unit

220v

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

Subject:	Single Purpose Lathe	Designed by:	Ihawan
Project:	SUE Production Line	Date	26 Oct 96
File:	SLIDE42	Scale:	1:2
		Drawing No.	0

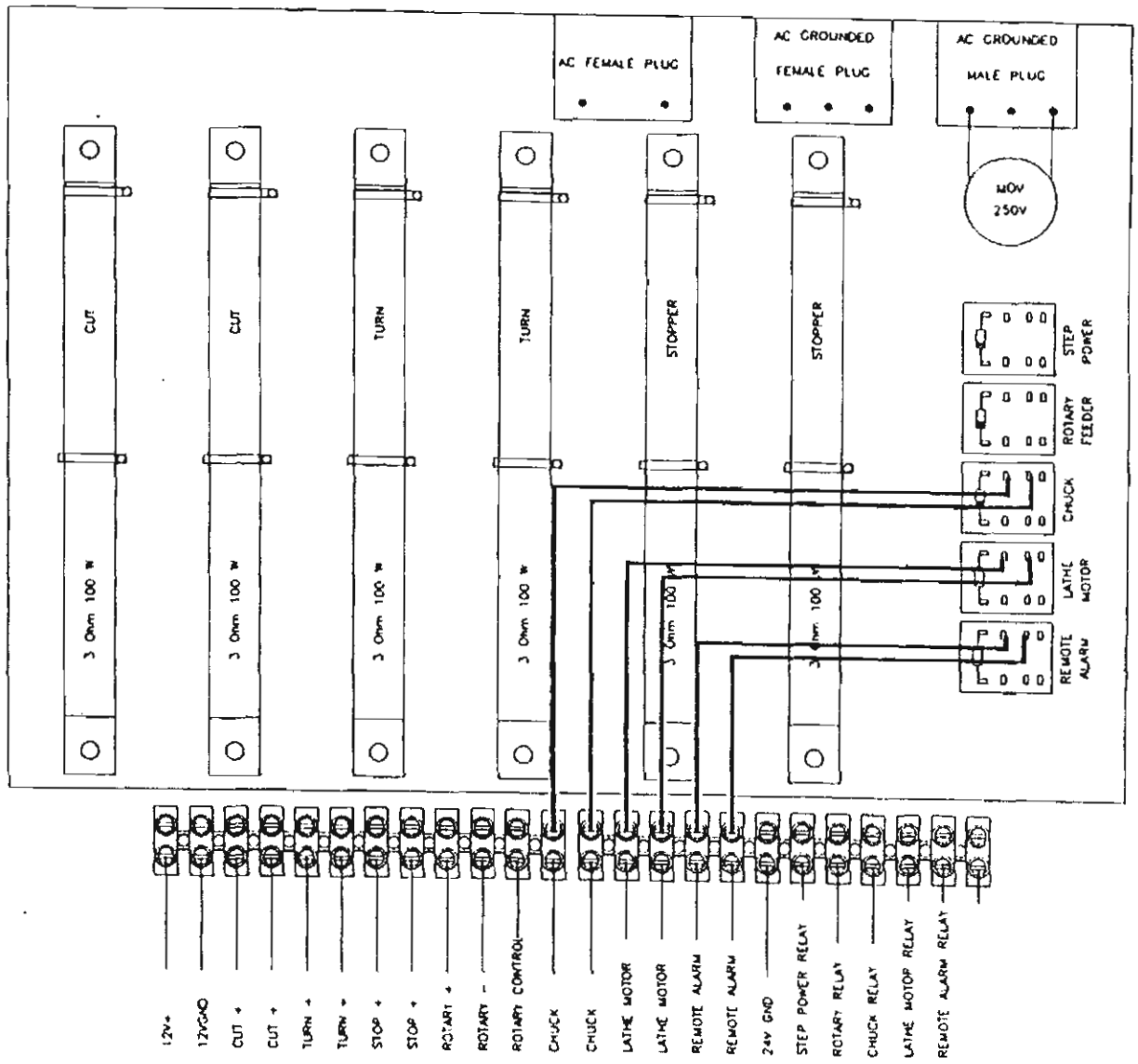


POWER CONTROL Unit

24V

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

Subject:	Single Purpose Lathe	Designed by:	Thawon
Project:	SUE Production Line	Date:	26 Oct 96
File:	SLIDE42	Scale:	1:2
		Drawing No.	0

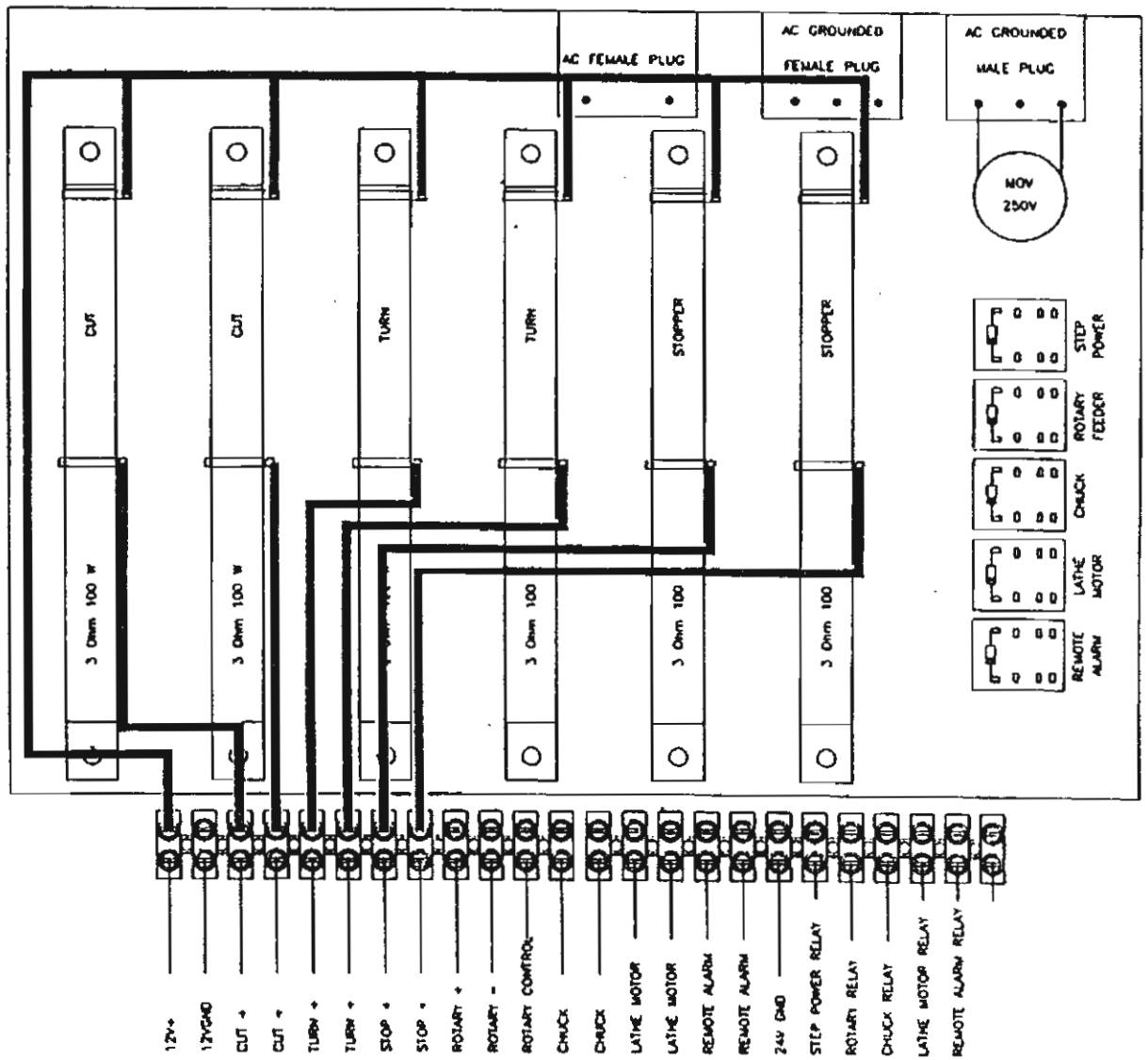


POWER CONTROL Unit

CONTROL

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

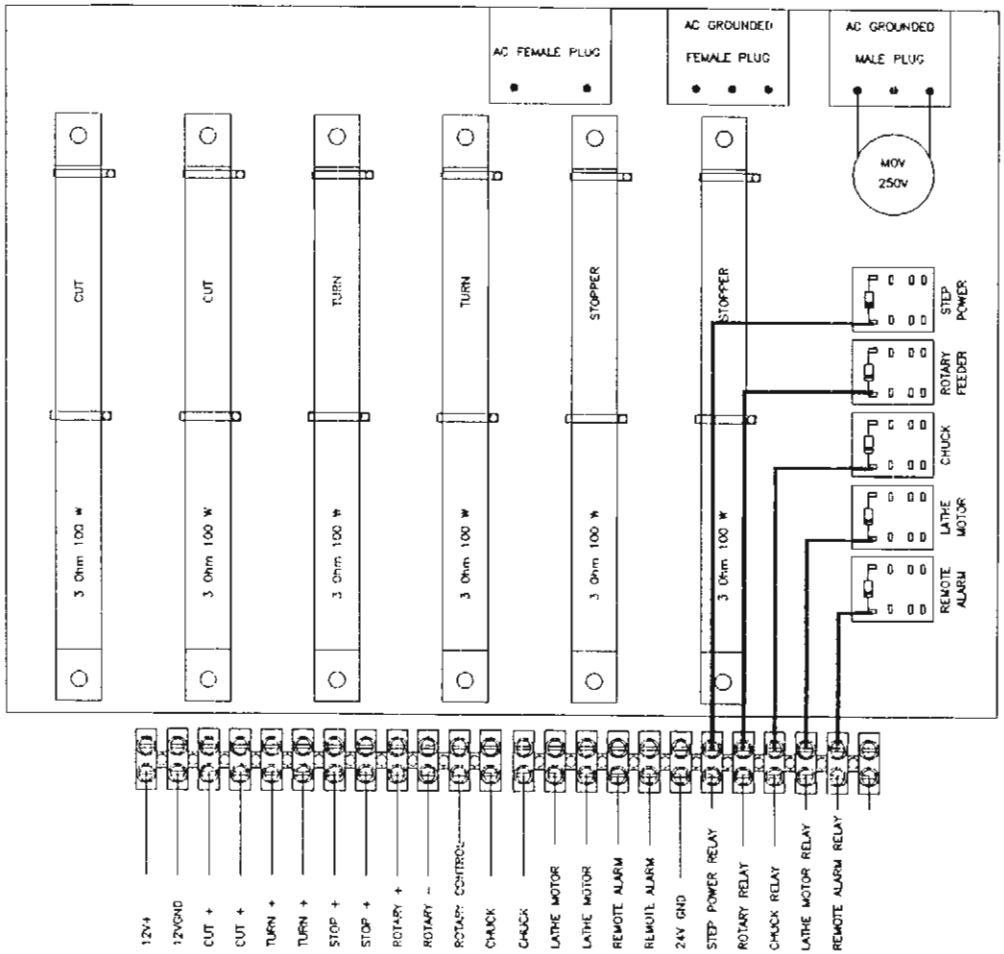
Subject	Single Purpose Lathe	Designed by:	Thawan
Project	SUE Production Line	Date	26 Oct 96
File	SLIDE42	Scale:	1:2
		Drawing No	0



POWER CONTROL UNIT

R

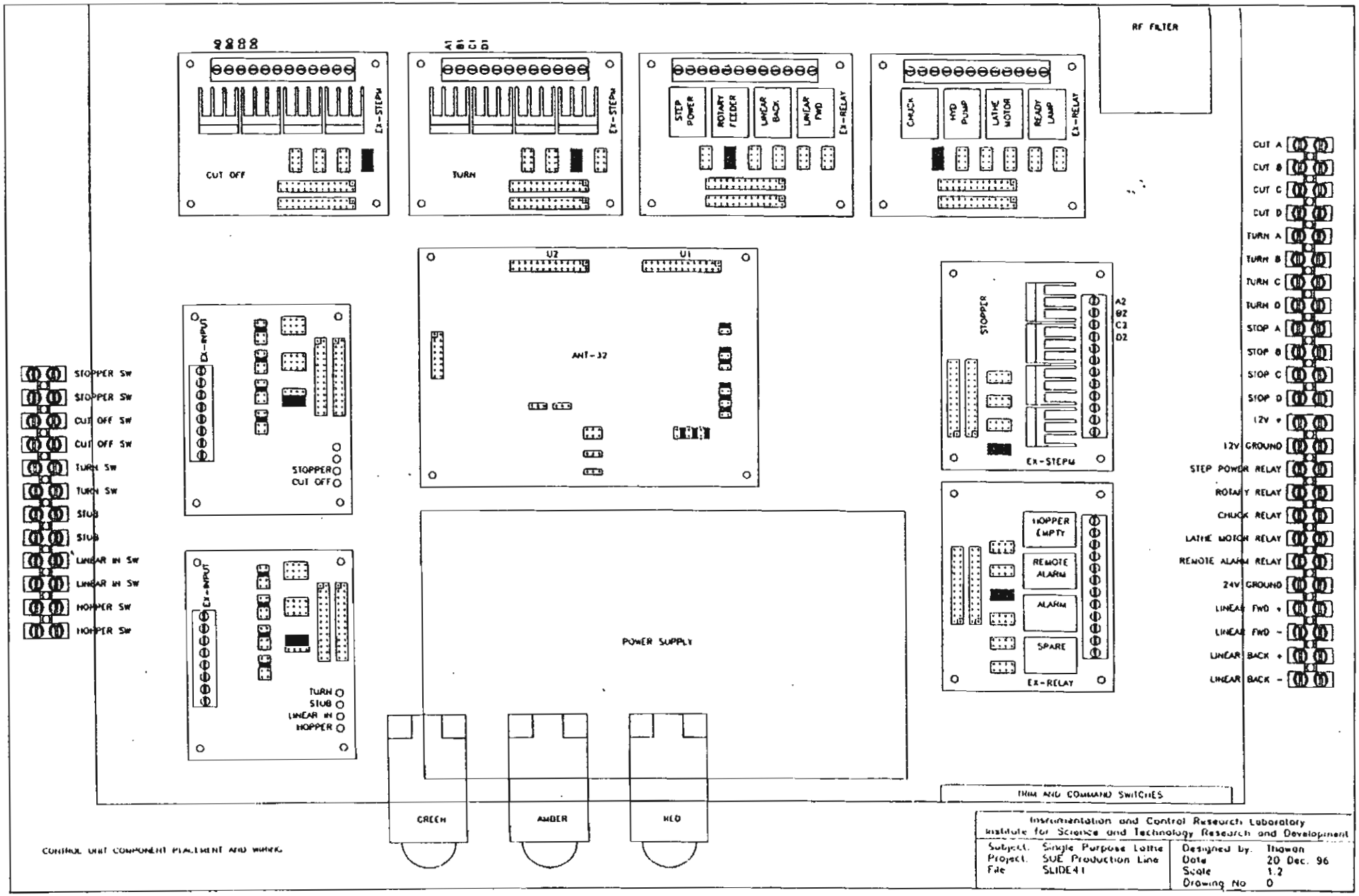
Instrumentation and Control Research Laboratory Institute for Science and Technology Research and Development	
Subject: Single Purpose Lathe Project: SUE Production Line File: SLIDE42	Designed by: Thawan Date: 26 Oct. 96 Scale: 1:2 Drawing No: 0



POWER CONTROL UNIT

RELAY

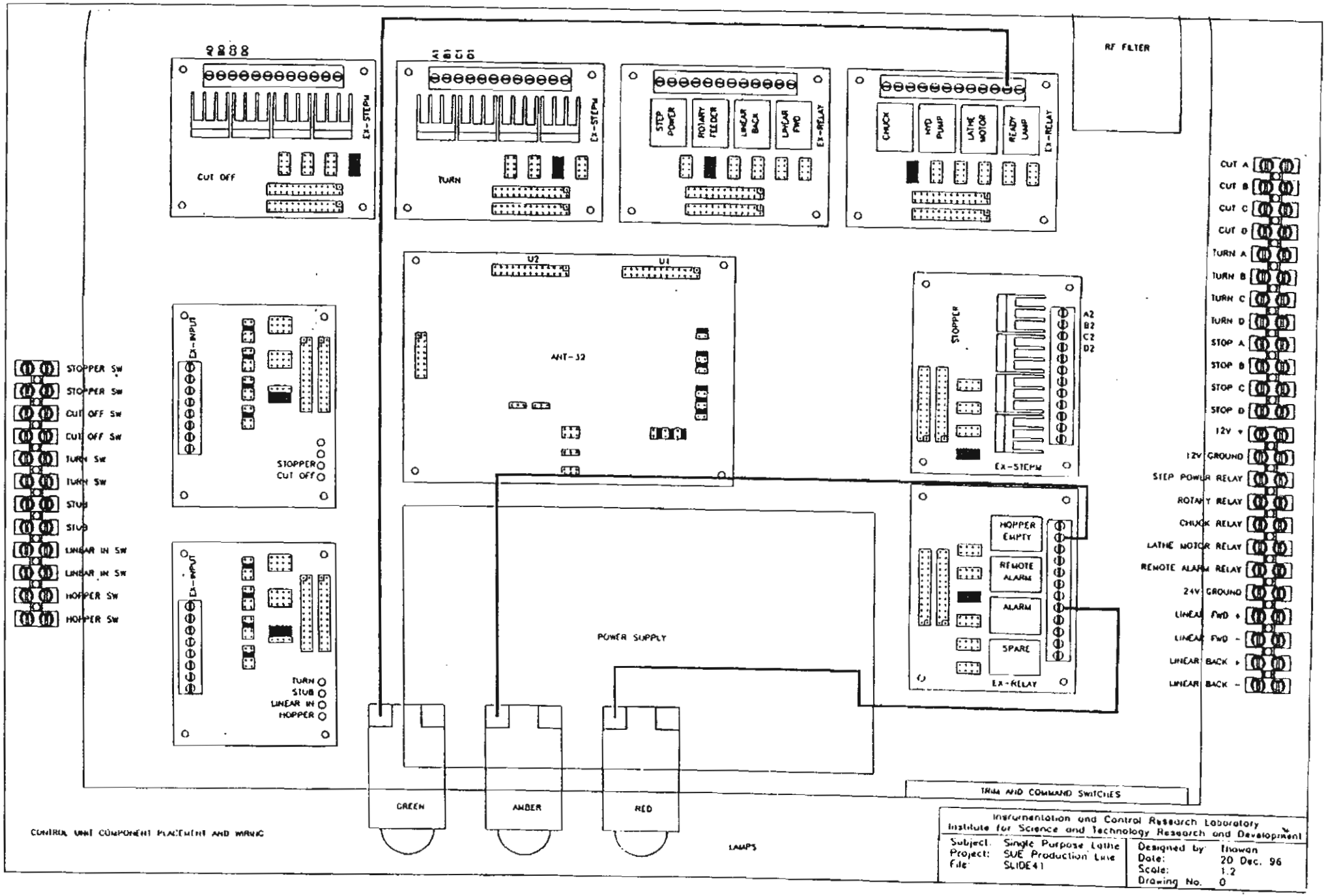
Instrumentation and Control Research Laboratory Institute for Science and Technology Research and Development	
Subject: Single Purpose Lathe Project: SUE Production Line File: SLIDE42	Designed by: Thawan Date: 26 Oct. 96 Scale: 1:2 Drawing No. 0



CONTROL UNIT COMPONENT PLACEMENT AND WIRING

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

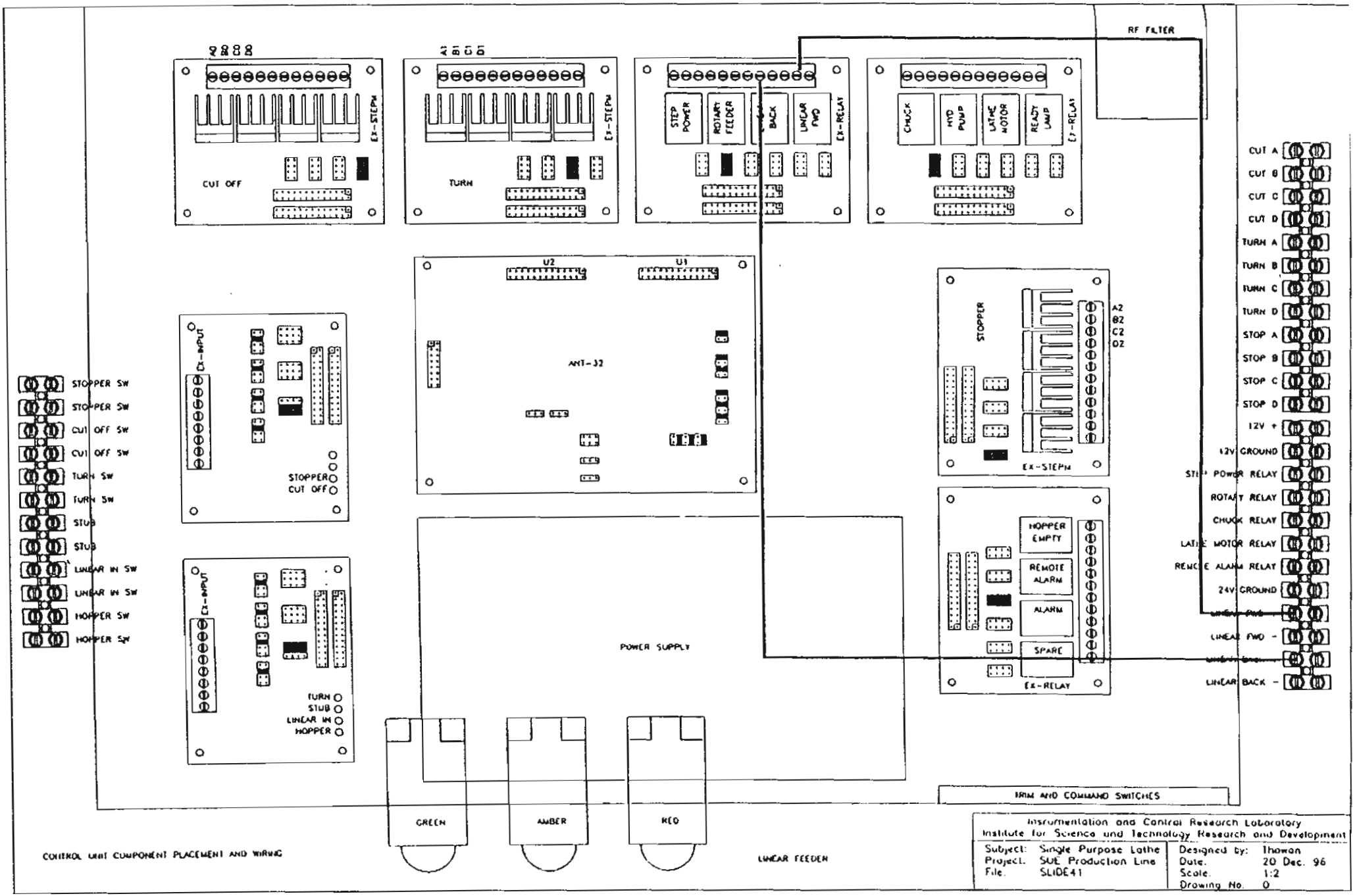
Subject	Single Purpose Lathe	Designed by	Thawan
Project	SUE Production Line	Date	20 Dec. 96
File	SLIDE41	Scale	1:2
		Drawing No	0



CONTROL UNIT COMPONENT PLACEMENT AND WIRING

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

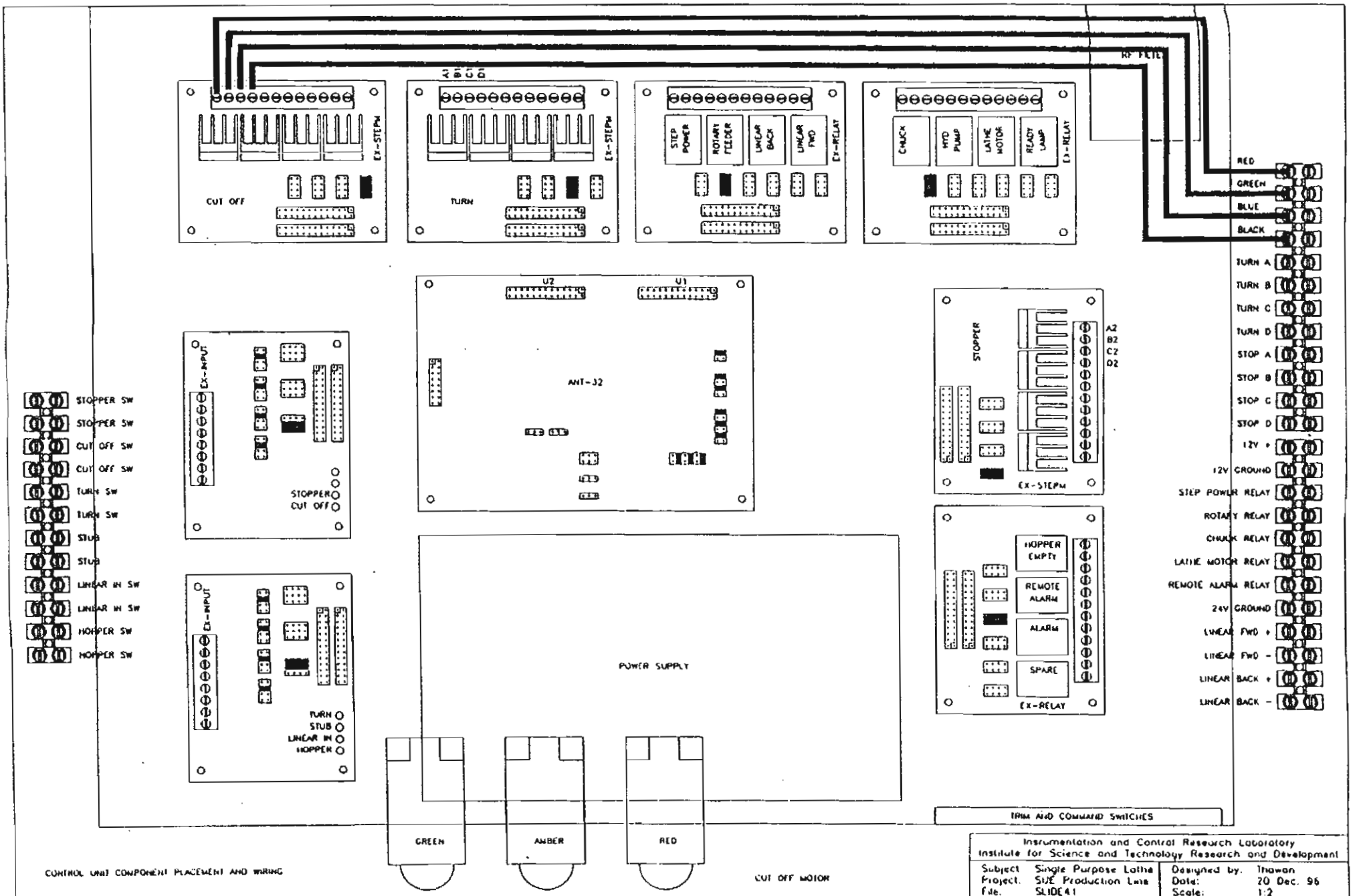
Subject:	Single Purpose Lathe	Designed by:	Ihsan
Project:	SUE Production Line	Date:	20 Dec. 96
File:	SLIDE41	Scale:	1:2
		Drawing No.	0



CONTROL UNIT COMPONENT PLACEMENT AND WIRING

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

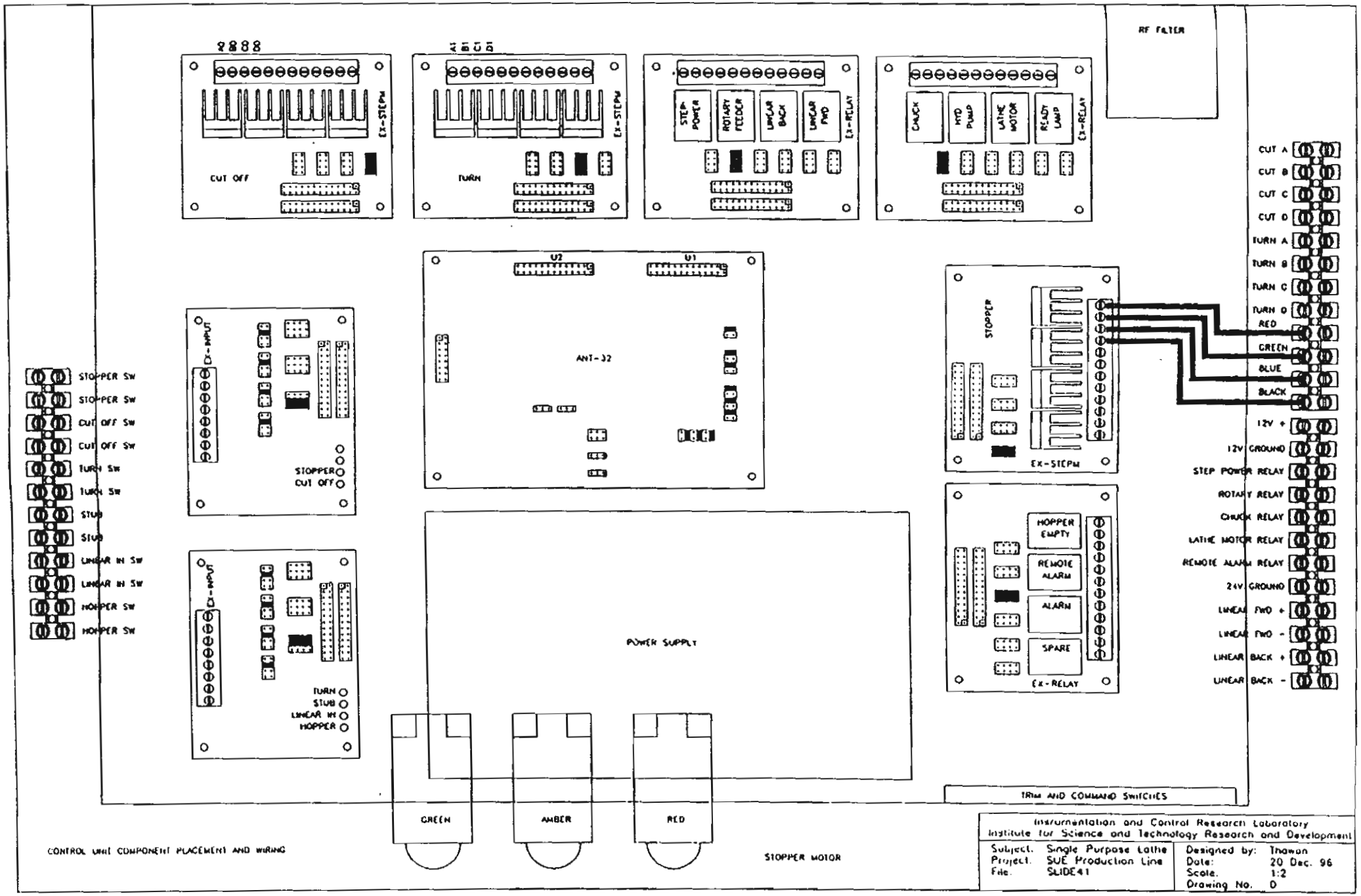
Subject:	Single Purpose Lathe	Designed by:	Thowan
Project:	SUE Production Line	Date:	20 Dec. 96
File:	SLIDE41	Scale:	1:2
		Drawing No.	0

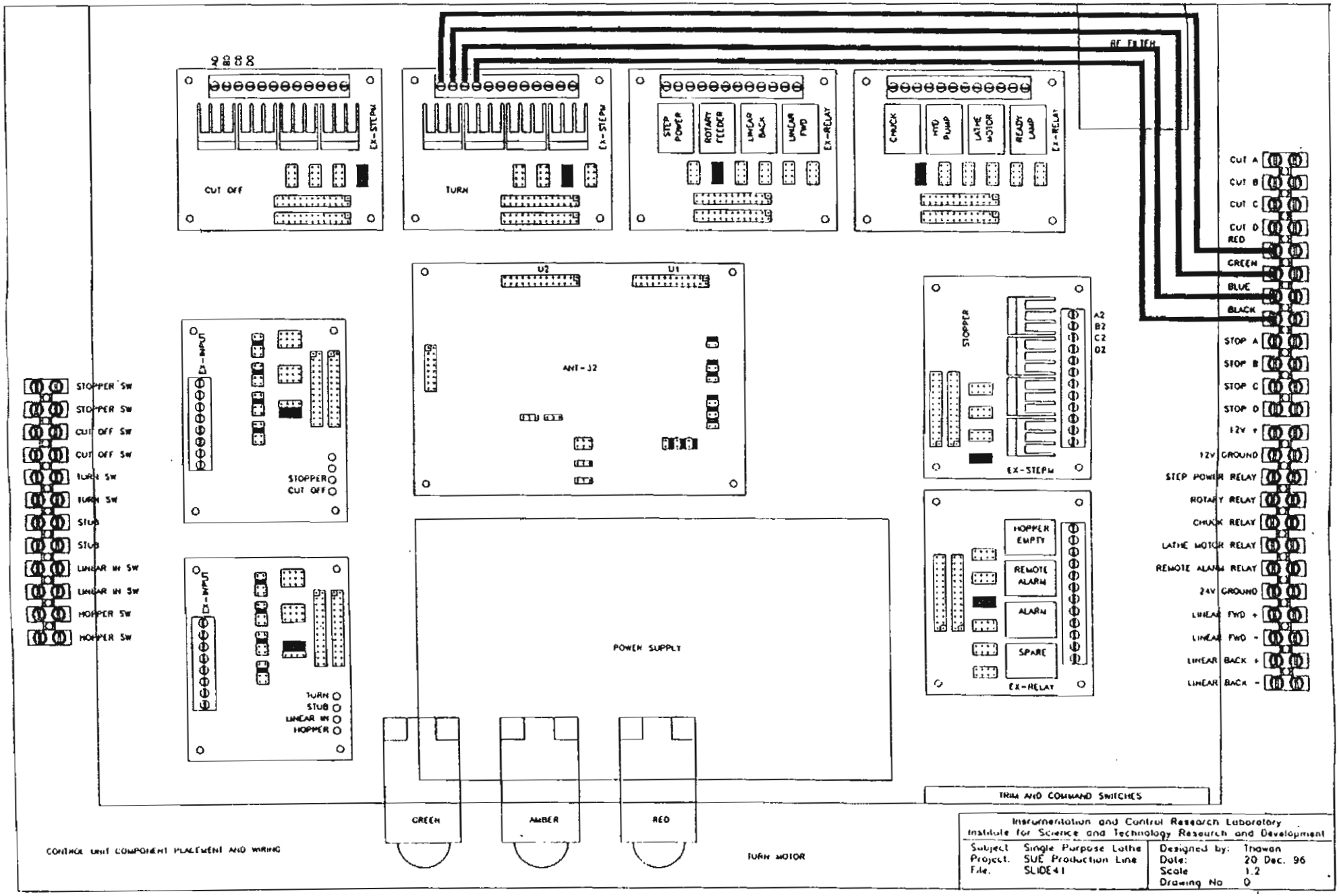


CONTROL UNIT COMPONENT PLACEMENT AND WIRING

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

Subject	Single Purpose Lathe	Designed by:	Thawan
Project	SME Production Line	Date:	20 Dec. 96
File:	SLIDE41	Scale:	1:2
		Drawing No	0

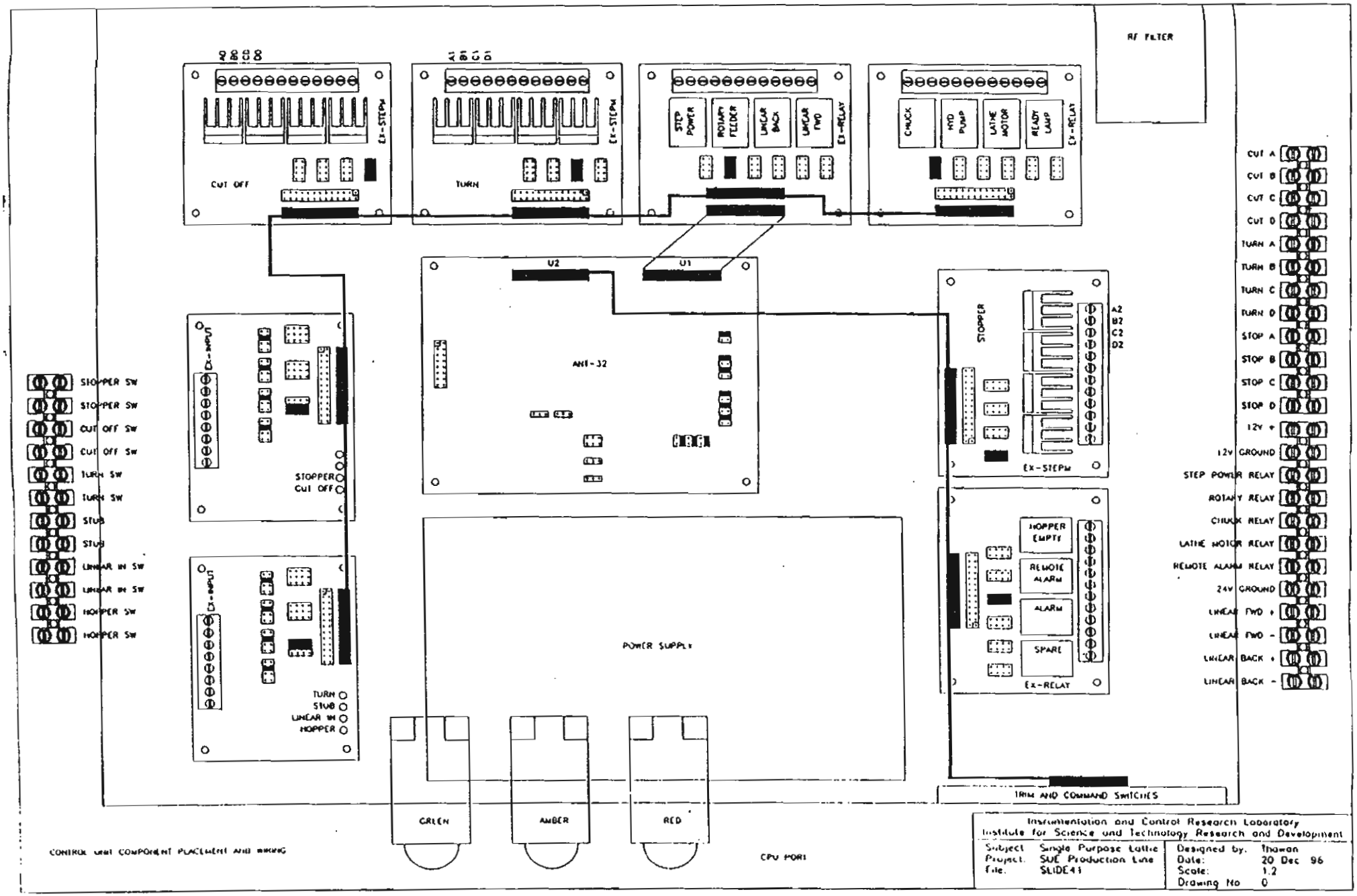

RF FILTER
CUT OFF
TURN
EX-STEPM
EX-RELAY
STEP POWER
ROTARY FEEDER
LINEAR BACK
LINEAR FWD
CHUCK
HYD PUMP
LATHE MOTOR
READY LAMP
ANT-32
U2
U1
DI-INPUT
STOPPER
CUT OFF
TURN
STUB
LINEAR IN
HOPPER
HOPPER EMPTY
REMOTE ALARM
ALARM
SPARE
POWER SUPPLY
GREEN
AMBER
RED
STOPPER SW
STOPPER SW
CUT OFF SW
CUT OFF SW
TURN SW
TURN SW
STUB
STUB
LINEAR IN SW
LINEAR IN SW
HOPPER SW
HOPPER SW
CUT A
CUT B
CUT C
CUT D
TURN A
TURN B
TURN C
TURN D
RED
GREEN
BLUE
BLACK
12V +
12V GROUND
STEP POWER RELAY
ROTARY RELAY
CHUCK RELAY
LATHE MOTOR RELAY
REMOTE ALARM RELAY
24V GROUND
LINEAR FWD +
LINEAR FWD -
LINEAR BACK +
LINEAR BACK -
TRIM AND COMMAND SWITCHES
CONTROL UNIT COMPONENT PLACEMENT AND WIRING
STOPPER MOTOR
Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development
Subject. Single Purpose Lathe
Project. SUE Production Line
File. SLIDE41
Designed by: Thawon
Date: 20 Dec. 96
Scale. 1:2
Drawing No. 0



CONTROL UNIT COMPONENT PLACEMENT AND WIRING

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

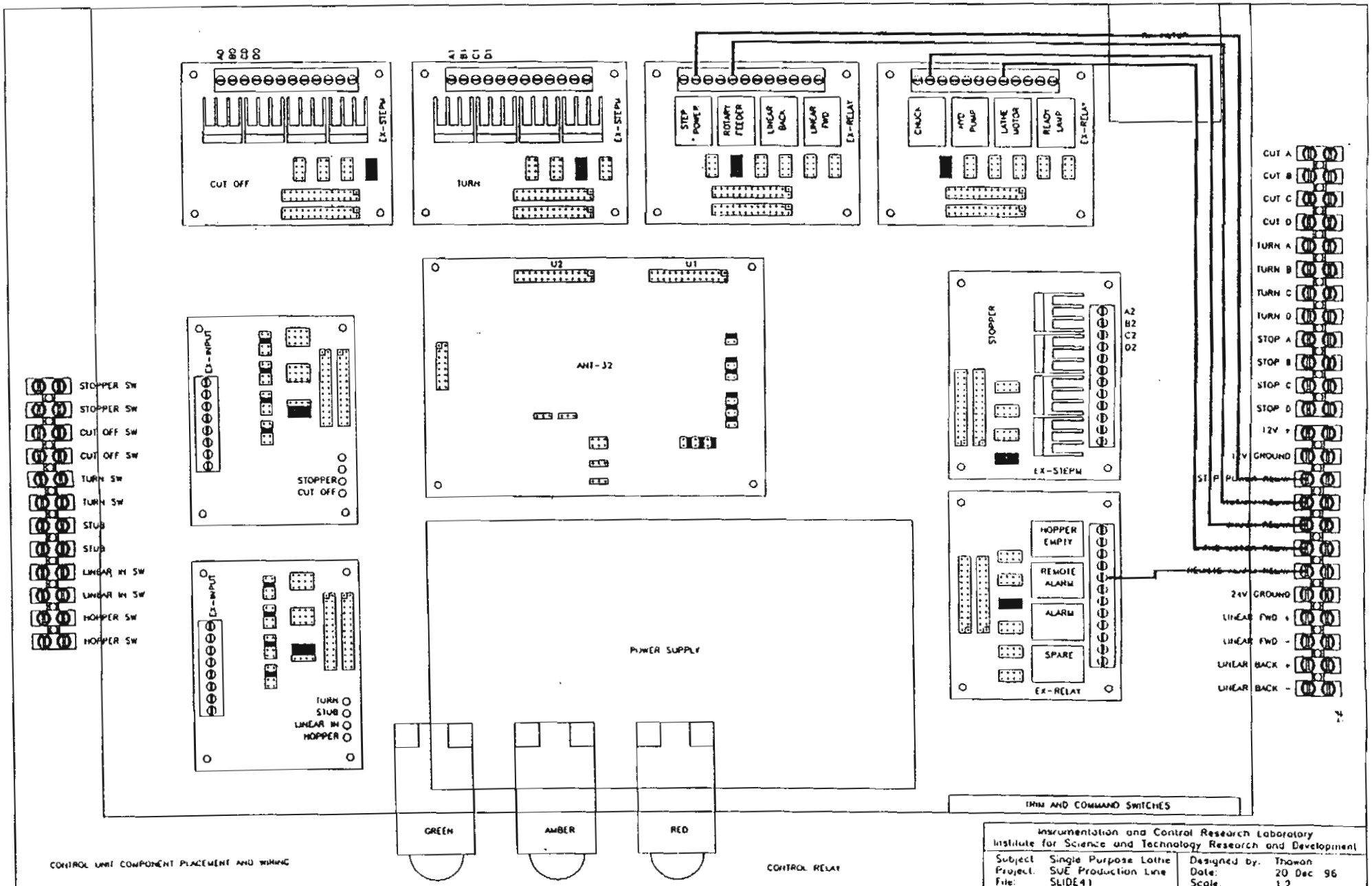
Subject	Single Purpose Lathe	Designed by:	Thawan
Project.	SUE Production Line	Date:	20 Dec. 96
File.	SLIDE41	Scale	1.2
		Drawing No	0



CONTROL UNIT COMPONENT PLACEMENT AND WIRING

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

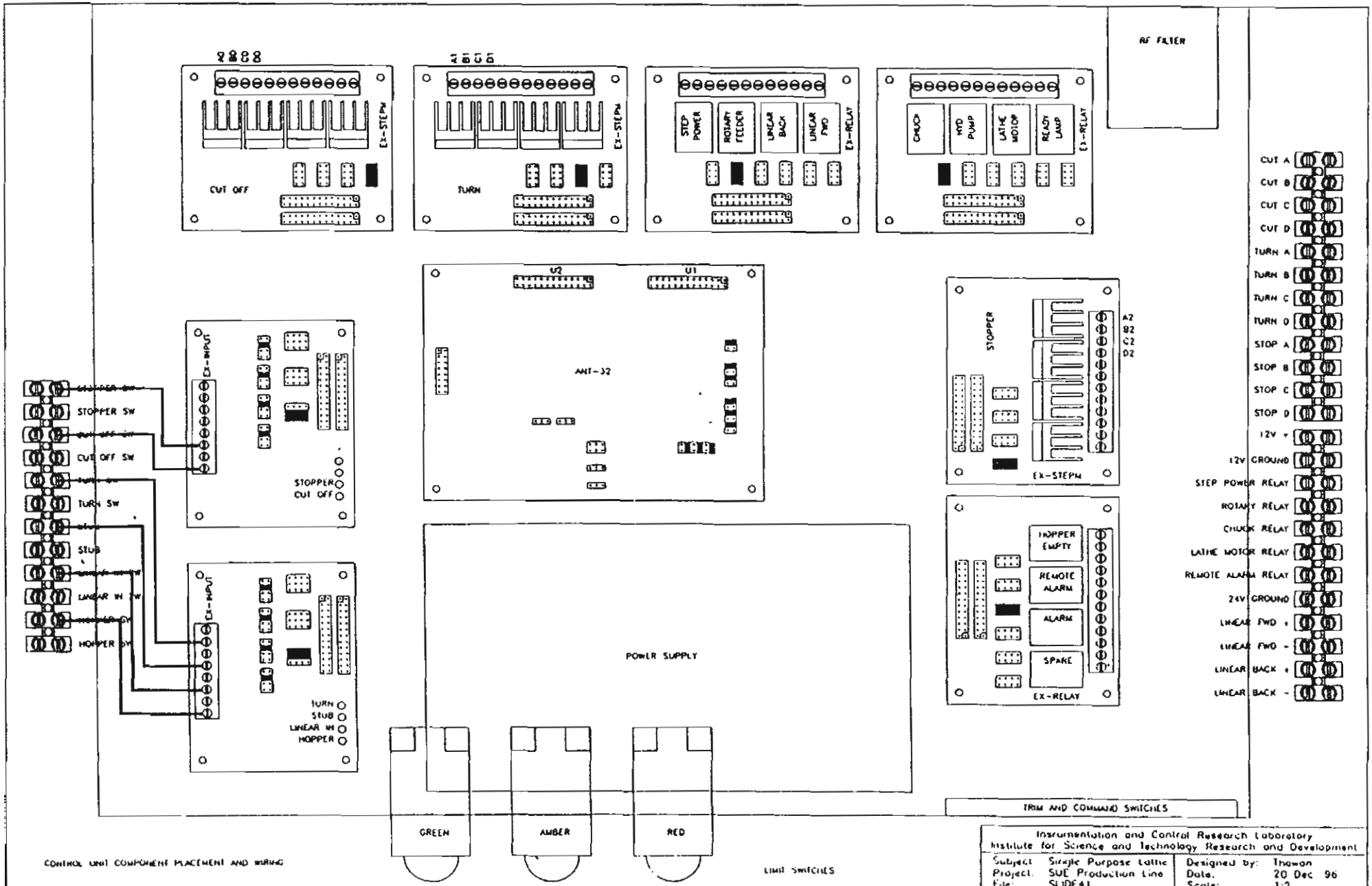
Subject	Single Purpose Lathe	Designed by:	Thawan
Project:	SUE Production Line	Date:	20 Dec 96
File:	SLIDE41	Scale:	1:2
		Drawing No	0

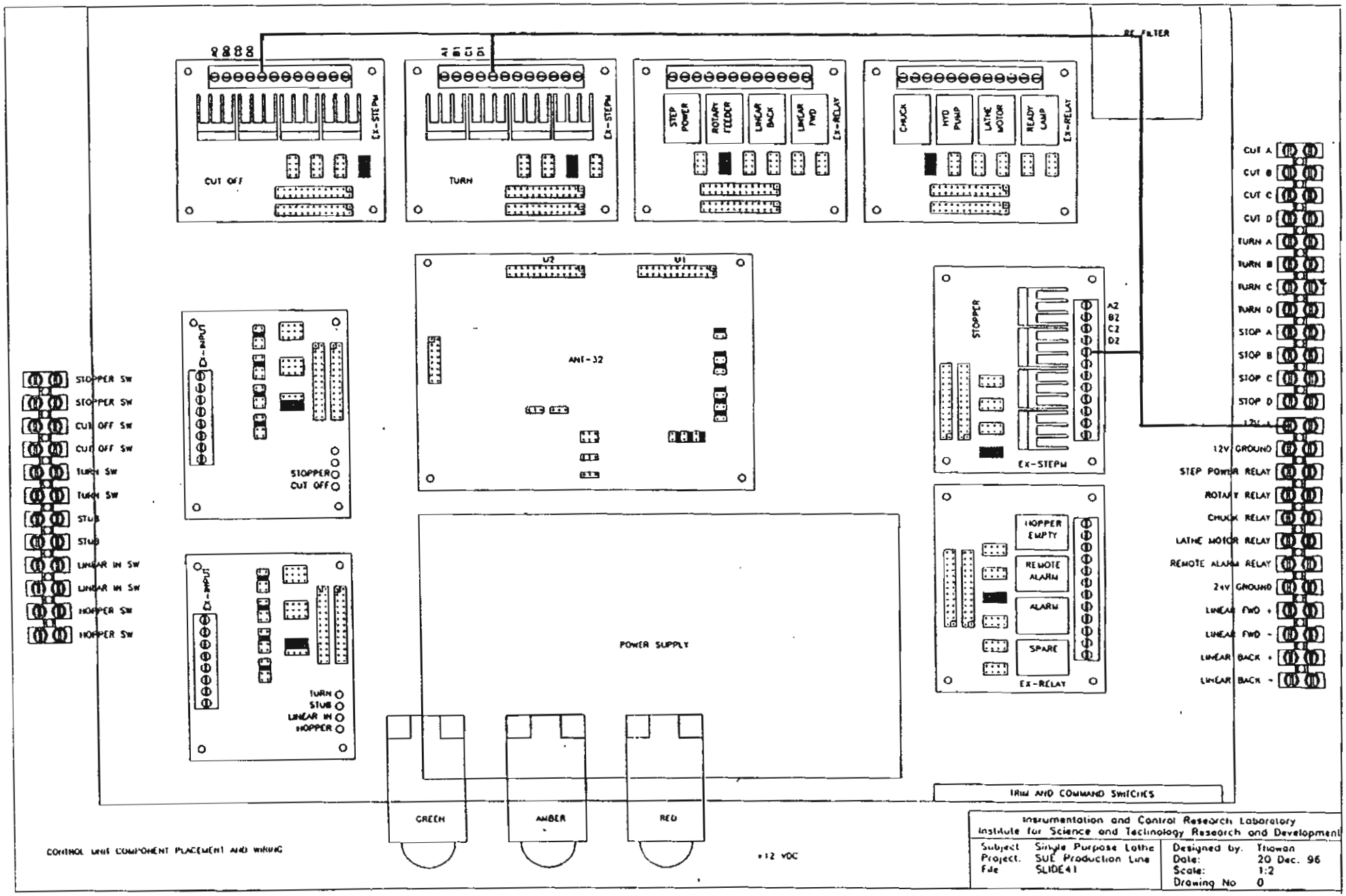


CONTROL UNIT COMPONENT PLACEMENT AND WIRING

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

Subject	Single Purpose Lathe	Designed by:	Thawon
Project	SUE Production Line	Date:	20 Dec 96
File:	SLIDE41	Scale:	1:2
		Drawing No.	0

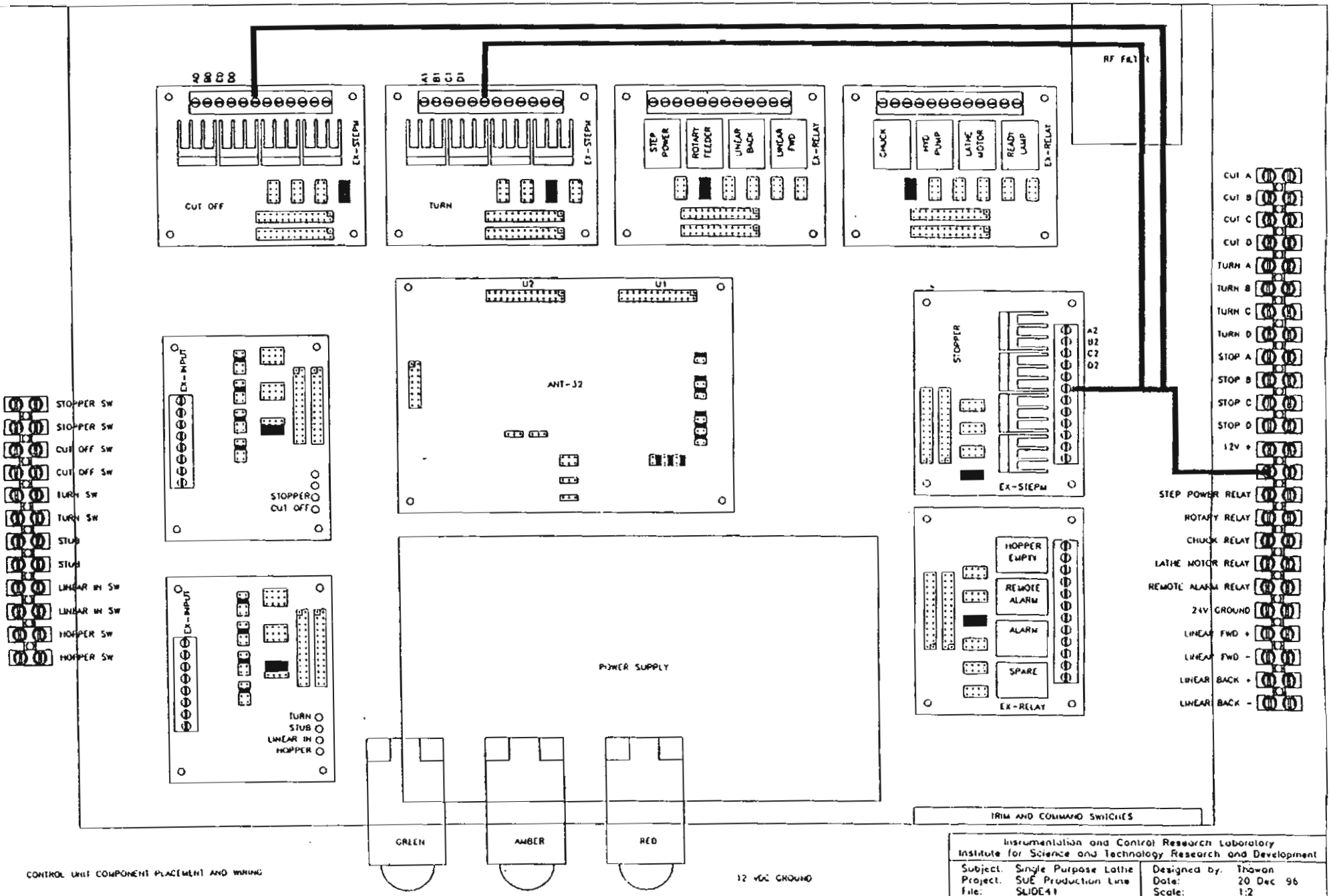

EX-STEPM
CUT OFF
TURN
EX-RELAY
STEP POWER
ROTARY FEEDER
LINEAR BACK
LINEAR FWD
CHUCK
HYD PUMP
LATHE MOTOR
READY LAMP
RF FILTER
CUT A
CUT B
CUT C
CUT D
TURN A
TURN B
TURN C
TURN D
STOP A
STOP B
STOP C
STOP D
12V +
12V GROUND
STEP POWER RELAY
ROTARY RELAY
CHUCK RELAY
LATHE MOTOR RELAY
REMOTE ALARM RELAY
24V GROUND
LINEAR FWD +
LINEAR FWD -
LINEAR BACK +
LINEAR BACK -
EX-INPUT
STOPPER SW
CUT OFF SW
TURN SW
STUB
LINEAR IN SW
HOPPER SW
STOPPER
CUT OFF
TURN
STUB
LINEAR IN
HOPPER
ANT-32
V2
V1
STOPPER
A2
B2
C2
D2
HOPPER EMPTY
REMOTE ALARM
ALARM
SPARE
POWER SUPPLY
TRIM AND COMMAND SWITCHES
GREEN
AMBER
RED
LIMIT SWITCHES
CONTROL UNIT COMPONENT PLACEMENT AND WIRING
Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development
Subject: Single Purpose Lathe
Project: SUE Production Line
File: SLIDE41
Designed by: Thawon
Date: 20 Dec 96
Scale: 1:2
Drawing No 0



CONTROL UNIT COMPONENT PLACEMENT AND WIRING

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

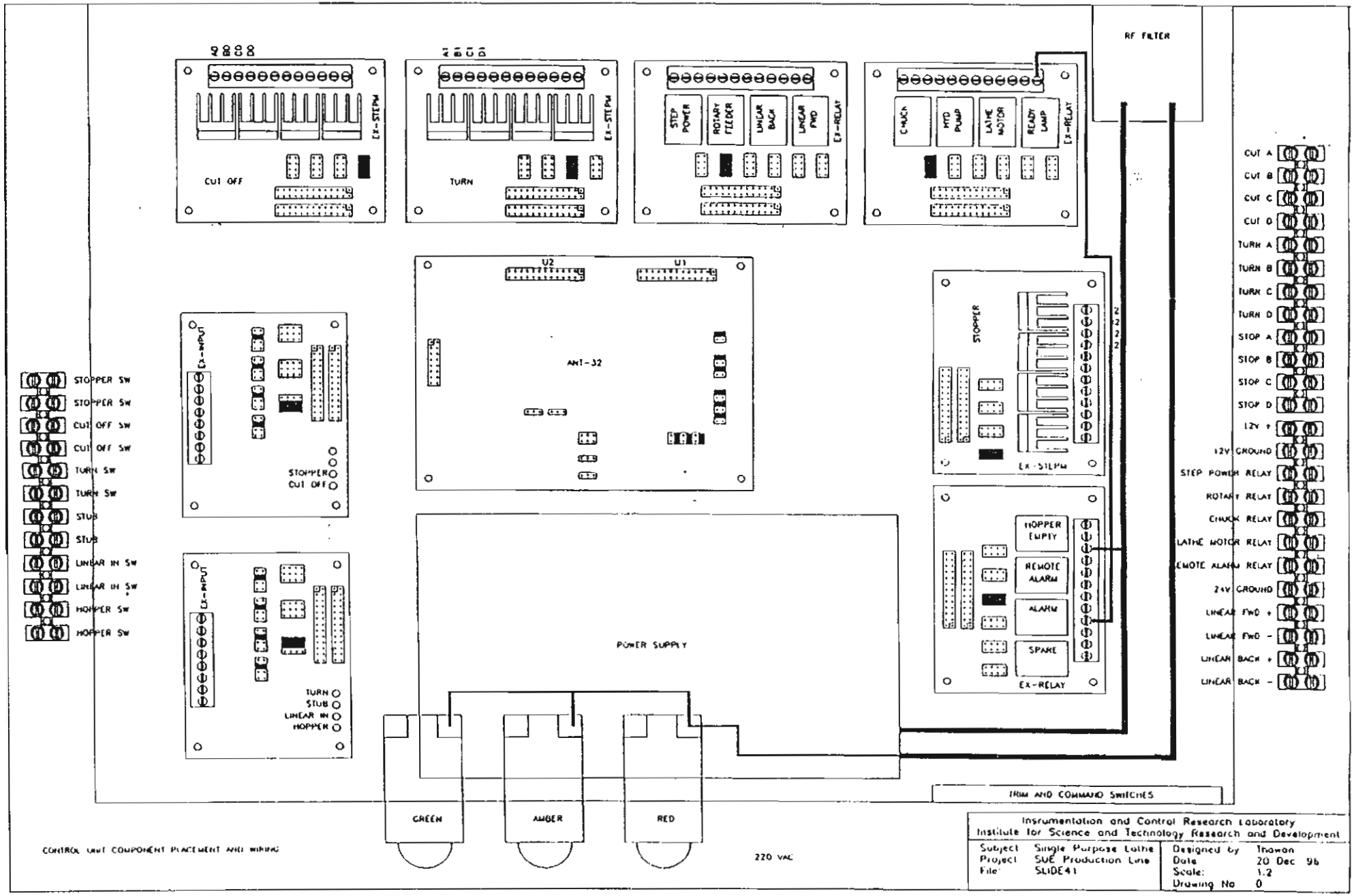
Subject	Simple Purpose Lathe	Designed by:	Yhowan
Project	SUE Production Line	Date:	20 Dec. 96
File	SLIDE41	Scale:	1:2
		Drawing No	0



CONTROL UNIT COMPONENT PLACEMENT AND WIRING

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

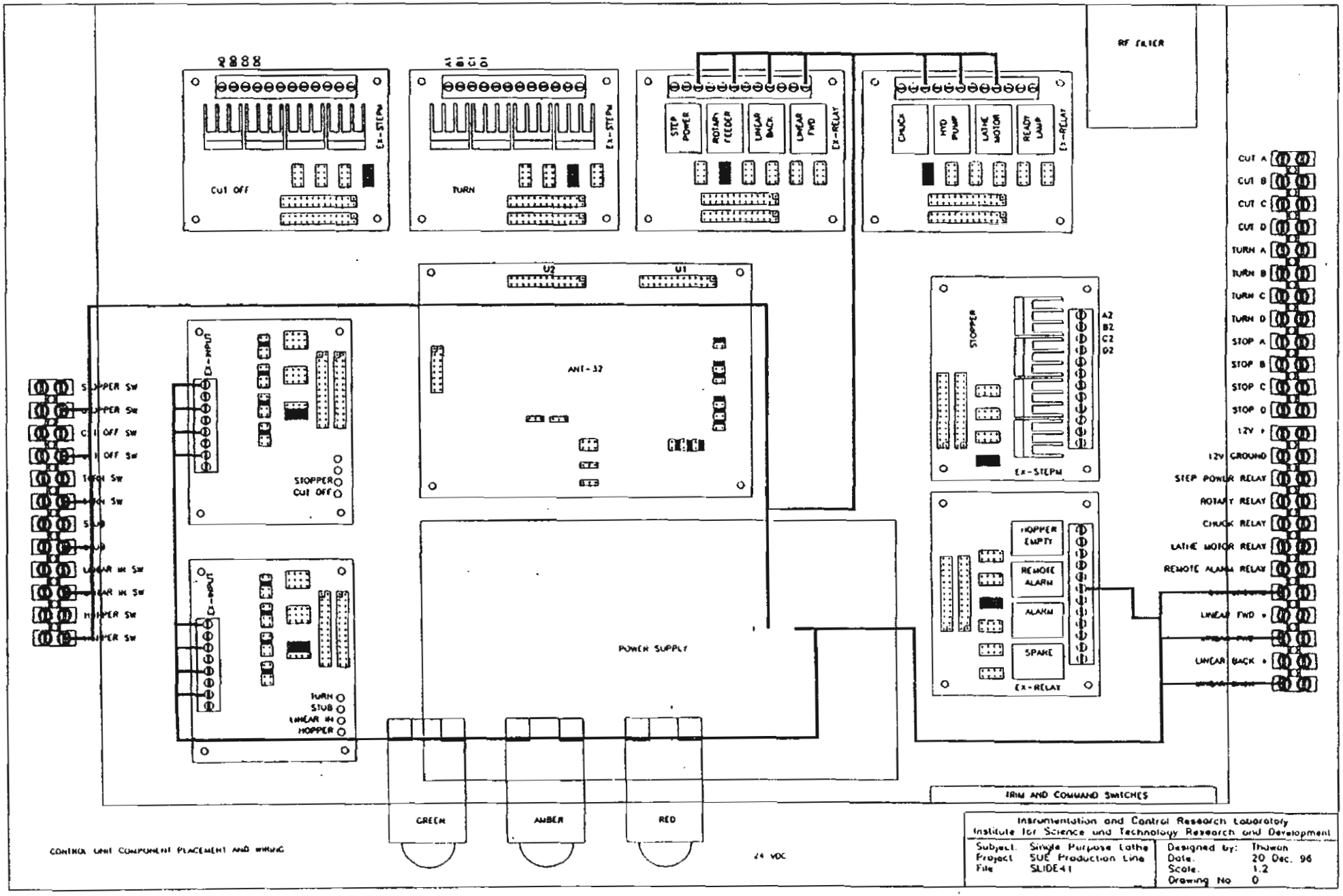
Subject:	Single Purpose Lathe	Designed by:	Thawon
Project:	SUE Production Line	Date:	20 Dec 96
File:	SLIDE41	Scale:	1:2
		Drawing No	0

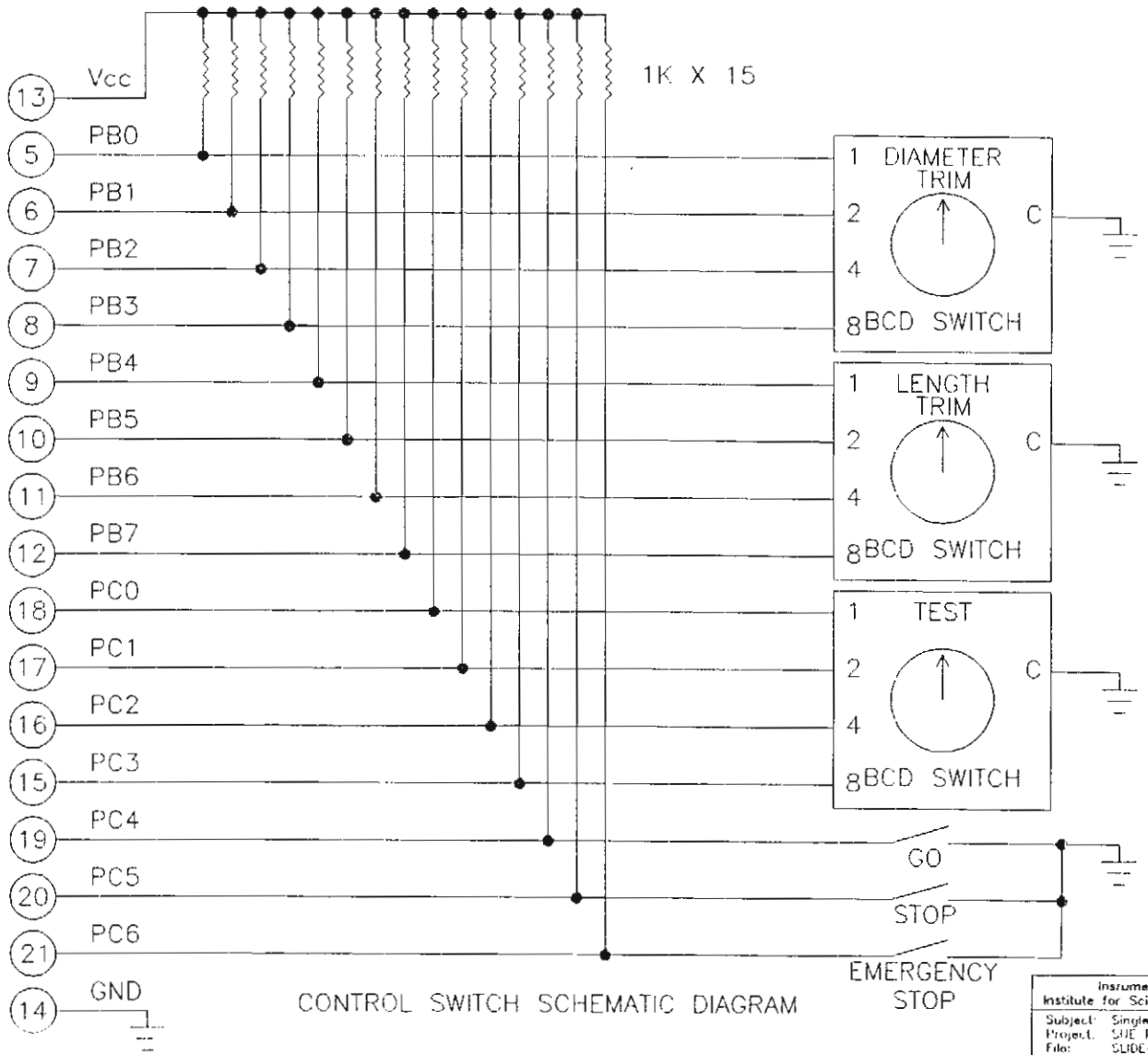


CONTROL UNIT COMPONENT PLACEMENT AND WIRING

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

Subject	Single Purpose Lathe	Designed by	Thawan
Project	SUE Production Line	Date	20 Dec 96
File	SLIDE41	Scale:	1:2
		Drawing No	0

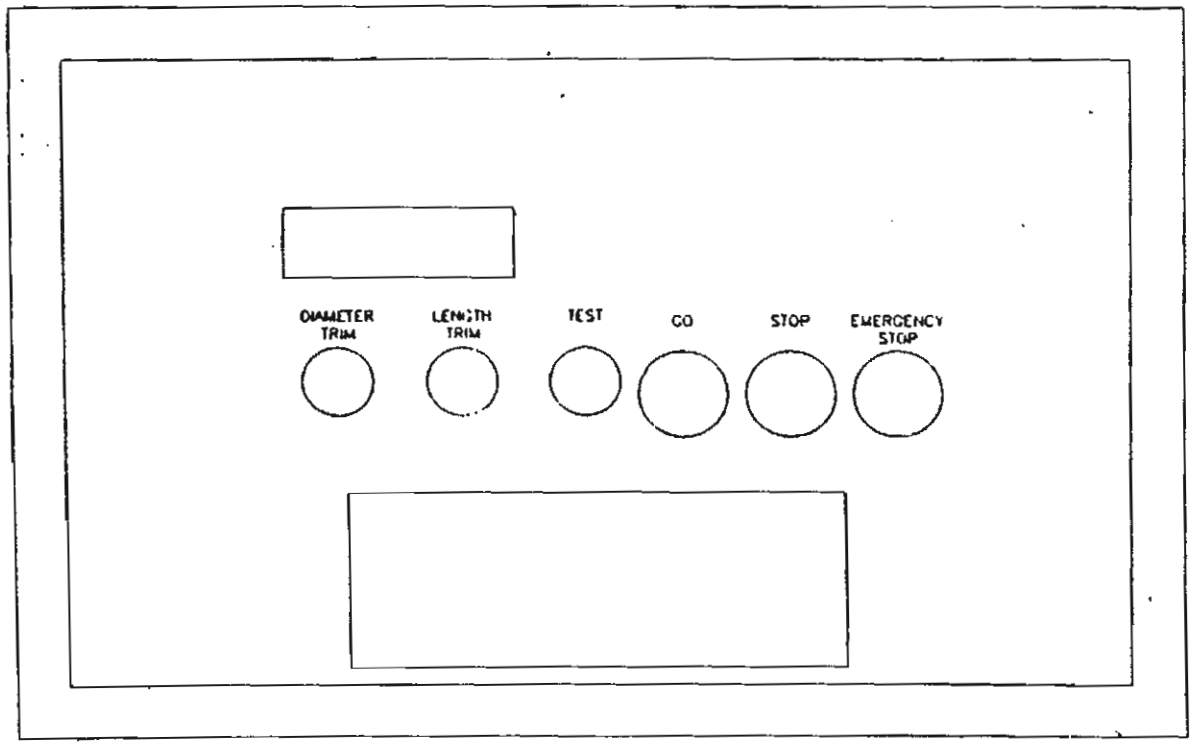

RF FILTER
EX-STEPM
CUT OFF
TURN
EX-RELAY
STEP POWER
ROTARY FEEDER
LINEAR BACK
LINEAR FWD
CHUCK
HYD PUMP
LATHE MOTOR
READY LAMP
U2
U1
ANT-32
EX-INPUT
STOPPER
CUT OFF
TURN
STUB
LINEAR IN
HOPPER
POWER SUPPLY
HOPPER EMPTY
REMOTE ALARM
ALARM
SPARE
A2
B2
C2
D2
CUT A
CUT B
CUT C
CUT D
TURN A
TURN B
TURN C
TURN D
STOP A
STOP B
STOP C
STOP D
12V +
12V GROUND
STEP POWER RELAY
ROTARY RELAY
CHUCK RELAY
LATHE MOTOR RELAY
REMOTE ALARM RELAY
LINEAR FWD +
LINEAR BACK +
TRIM AND COMMAND SWITCHES
GREEN
AMBER
RED
24 VDC
CONTROL UNIT COMPONENT PLACEMENT AND WIRING
Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development
Subject. Single Purpose Lathe
Project SUE Production Line
File SLIDE41
Designed by: Thawan
Date. 20 Dec. 96
Scale. 1:2
Drawing No 0



CONTROL SWITCH SCHEMATIC DIAGRAM

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

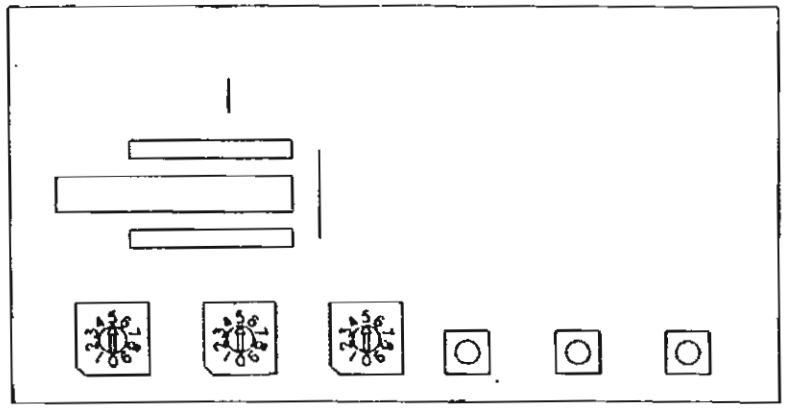
Subject:	Single Purpose Lathe	Designed by:	Thawan
Project:	SME Production Line	Date:	15 Jan 97
File:	SLIDE40A	Scale:	1:1
		Drawing No.	0



CONTROL SWITCH

FRONT PANNAL

Instrumentation and Control Research Laboratory Institute for Science and Technology Research and Development			
Subject.	Single Purpose Lathe	Designed by:	Thawan
Project.	SUE Production Line	Date:	26 Oct. 98
File.	SLIDE40	Scale:	1:1
		Drawing No	0

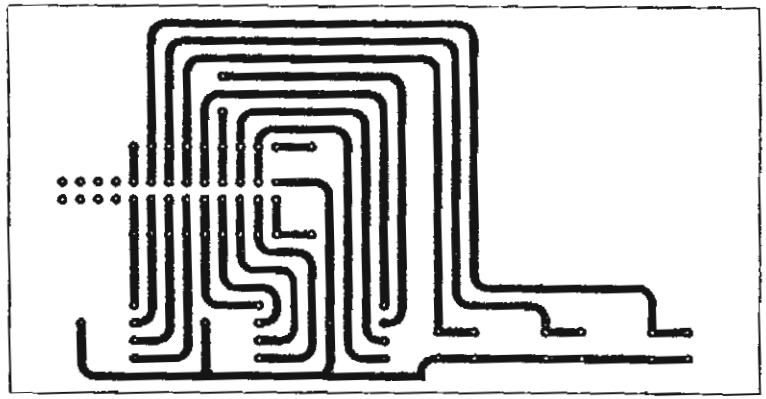


CONTROL SWITCH

PART LAYOUT

Instrumentation and Control Research Laboratory
Institute for Science and Technology Research and Development

Subject:	Single Purpose Lathe	Designed by:	Thawon
Project:	SUE Production Line	Date:	26 Oct. 96
File:	SUDE40	Scale:	1:1
		Drawing No.	0



CONTROL SWITCH

PCB

Instrumentation and Control Research Laboratory Institute for Science and Technology Research and Development			
Subject	Single Purpose Lathe	Designed by:	Thawan
Project	SUE Production Line	Date:	26 Oct. 96
File:	SLIDE40	Scale:	1:1
		Drawing No.	0

โครงการ Transfer Line

TRF 6/3/TRFLINE

โครงการ Transfer Line
TRF 6/3/TRFLINE

สรุปโครงการ

โครงการ Transfer Line เป็นโครงการปรับปรุงกระบวนการผลิตสไลด์เดอร์ ซึ่งเป็นชิ้นส่วนหนึ่งในคอมเพรสเซอร์ โดยนำเอาสถานีงานต่างๆ 3 สถานีคือ สถานีเจาะ สถานีลบคม และสถานีคว้านมารวมกันเป็นเครื่องจักรชุดเดียว มีกลไกขนถ่ายชิ้นงานระหว่างสถานีแทนการใช้พนักงานในการสร้างเครื่อง Transfer Line บางส่วนก็ใช้เครื่องจักรเก่าของโรงงานมาปรับปรุงใหม่

ทางด้านการดำเนินงาน บริษัท ซันโยฯ เป็นผู้ออกแบบและจัดสร้างระบบ Transfer Line ทั้งหมด โดยมีสำนักงานเทคโนโลยี สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหารลาดกระบัง และมหาวิทยาลัยเชียงใหม่เป็นผู้ให้คำปรึกษาในการออกแบบ ผลที่ได้จากโครงการ ช่วยให้ประสิทธิภาพของกระบวนการผลิตสูงขึ้น ใช้เนื้อที่ในการผลิตน้อยลง และลดค่าใช้จ่ายในการผลิตชิ้นงาน

โครงการนี้เป็นการออกแบบ และสร้างเครื่องจักรใหม่ โดยนำวัสดุอุปกรณ์ที่มีอยู่มาสร้างเครื่องต้นแบบ การออกแบบ และสร้างเครื่องต้นแบบไม่ได้ลอกเลียนเครื่องจักรที่มีขายอยู่แล้ว แต่เป็นการออกแบบเครื่องจักรโดยการดัดแปลงกลไกการทำงานของเครื่องจักรในโรงงานในรูปแบบต่างๆ มาประยุกต์ให้เหมาะสมกับแต่ละขั้นตอนของการทำชิ้นงาน เครื่องต้นแบบที่ได้พัฒนาขึ้นมายังจำเป็นต้องแก้ไขปรับปรุงในรายละเอียดเพื่อให้มีความแม่นยำสูงกว่าที่เป็นอยู่ จากนั้นจึงจะเข้าไปทดแทนในสายการผลิตจริง ความลำบากของการสร้างเครื่องต้นแบบคือ การสร้างชิ้นส่วนที่นำมาประกอบต้องใช้ช่างฝีมือที่มีความเชี่ยวชาญมาก แม้กระนั้นผลงานที่ออกมาก็ยังไม่ได้ตามข้อกำหนด ผลจากการค้นคว้าวิจัยได้เพิ่มขีดความสามารถของทีมงานในด้านการออกแบบ และการสร้างเครื่องจักรขึ้นใช้เอง พื้นฐานทางด้านเทคโนโลยีที่ได้จากโครงการนี้เป็นการเริ่มต้นสำหรับการพึ่งตนเองทางด้านเทคโนโลยี SUE ยังจะทำการพัฒนาและปรับปรุงต่อไปให้สามารถทำงานได้สมบูรณ์ยิ่งขึ้น

บทนำ

โครงการ Transfer Line เป็นการปรับปรุงกระบวนการผลิตชิ้นส่วนสไลด์เดอร์ของคอมเพรสเซอร์ให้มีประสิทธิภาพสูงขึ้น ใช้เนื้อที่ในการผลิตน้อยลง เพิ่มความปลอดภัยในการปฏิบัติงาน และลดค่าใช้จ่ายในการผลิตชิ้นงาน โดยการรวมสถานีงานที่แยกกันอยู่ 3 สถานี มารวมกันอยู่ในเครื่องจักรเพียงเครื่องเดียว และทำงานเป็นแบบอัตโนมัติ

ในการผลิตสไลด์เดอร์มีขั้นตอนการผลิตมากถึง 42 ขั้นตอนด้วยกัน และมี 11 ขั้นตอนที่อยู่ในส่วนของการผลิตด้วยเครื่อง Transfer Line ที่จัดสร้างขึ้น แต่เดิมนั้นต้องใช้เครื่องหลายเครื่องในการผลิตสไลด์เดอร์ แบ่งเป็นเครื่องเจาะ 5 เครื่อง (สถานีที่ 1) เครื่องลบคม 1 เครื่อง (สถานีที่ 2) และเครื่องคว้าน 2 เครื่อง (สถานีที่ 3) ใช้พนักงานในการผลิต 5 คน มีกำลังการผลิตชิ้นงานรวม 2,500 ชิ้น/กะ หลังจากที่พัฒนาเป็นเครื่อง Transfer Line แล้ว จะกลายเป็นเครื่องจักรเพียงเครื่องเดียว ใช้พนักงานดูแล 1 คน และมีอัตราการผลิตชิ้นงาน 3,500 ชิ้น/กะ

ระบบการผลิตของเครื่อง Transfer Line เป็นระบบทำงานแบบอัตโนมัติทั้งหมด ควบคุมการทำงานด้วย Programable Logic Controller (PLC) โดยประกอบด้วยส่วนสำคัญต่างๆ 4 ส่วนคือ

1) แขนกล (Transfer Arm) ทำหน้าที่ขนถ่ายชิ้นงานระหว่างสถานี
2) เครื่องเจาะ (Drilling Machine) ประกอบด้วยหัวเจาะ 6 หัว
3) เครื่องลบคม (Chamfering Machine) ประกอบด้วยหัวลบคม 12 หัว
4) เครื่องคว้าน (Boring Machine) ประกอบด้วยหัวคว้าน 6 หัว

ในการทำงานของเครื่องจะเป็นการผลิตชิ้นงานครั้งละ 6 ชิ้น โดยเคลื่อนไปตามสถานีต่างๆ ซึ่งมีแขนกลที่ทำงานเป็นแบบอัตโนมัติ ชิ้นงานที่ออกจากเครื่อง TRANSFER LINE จะส่งต่อไปยังกระบวนการอื่นอีก 13 ขั้นตอนก็จะได้สไลด์เดอร์ที่สมบูรณ์

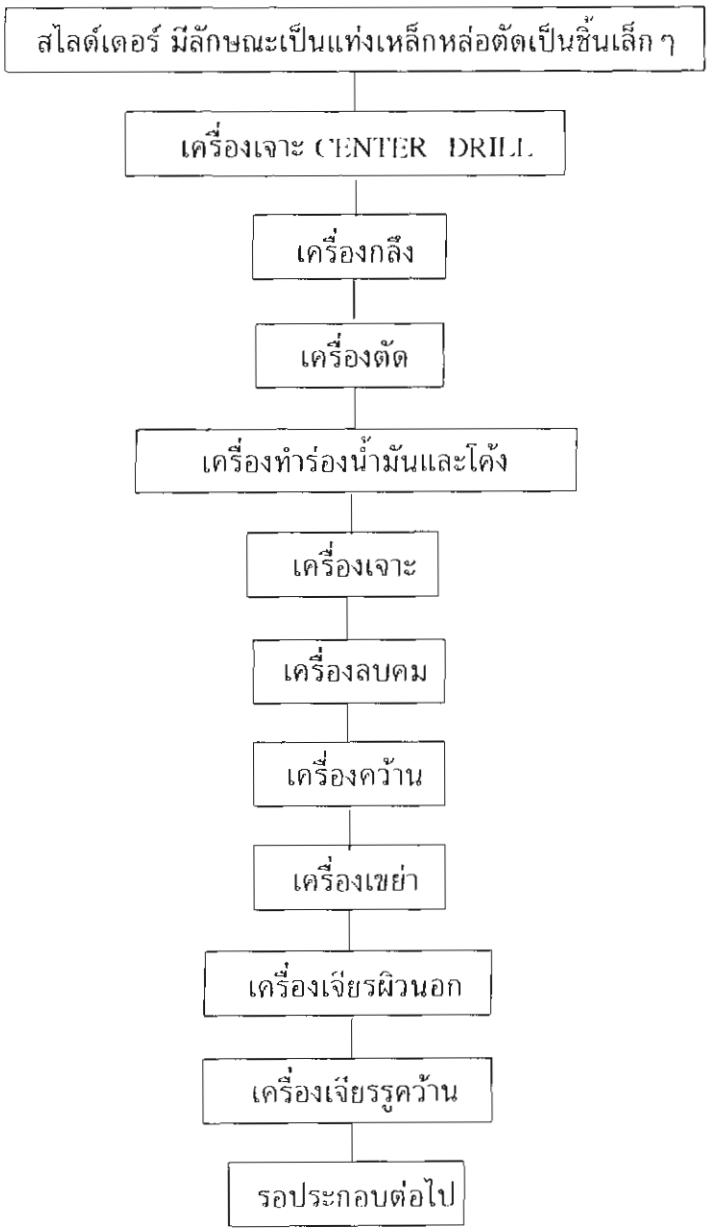
สารบัญ

หน้า

1. หลักการทำงาน

1.1 การผลิตชิ้นงาน สไลด์เดอร์ อย่างสมบูรณ์

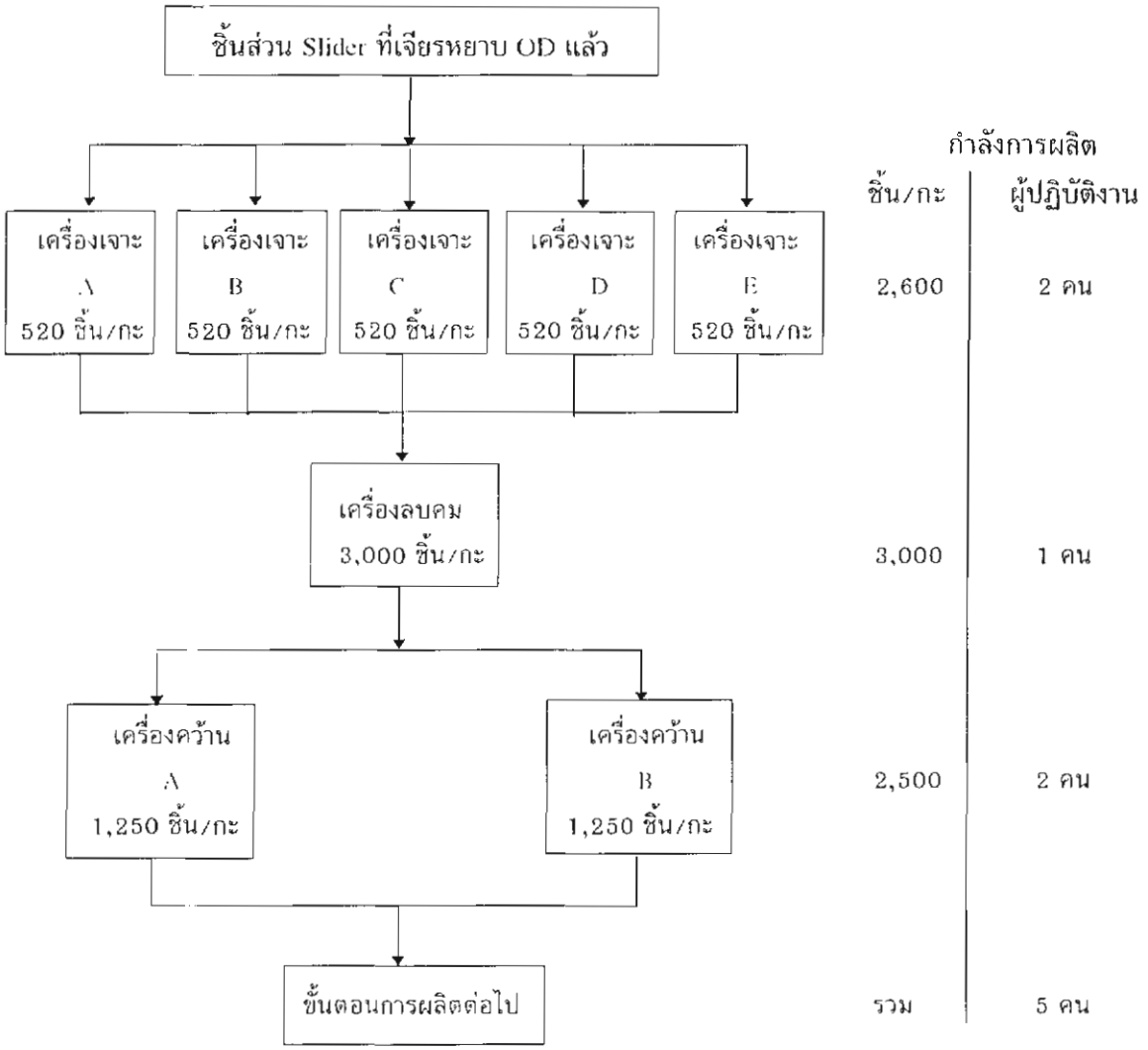
กระบวนการผลิตชิ้นส่วนสไลด์เดอร์ของคอมเพรสเซอร์ในตู้เย็นซันโย มีทั้งหมด 42 ขั้นตอนด้วยกัน โดยเริ่มจากนำแท่งเหล็กหล่อมากลึงปอกผิว และตัดเป็นชิ้นเล็ก ๆ ตามขนาดที่กำหนด จากนั้นจึงนำมากลึงร่องน้ำมันและโค้งที่หัวท้าย ซึ่งสไลด์เดอร์ที่ได้จะเป็นชิ้นงานที่จะนำไปเข้าชุด Transfer Line ที่จัดสร้างขึ้น ซึ่งมีการทำงาน 11 ขั้นตอนด้วยกัน เมื่อชิ้นงานออกจากชุด Transfer Line แล้วก็จะเข้ากระบวนการต่าง ๆ อีก 13 ขั้นตอน จะได้สไลด์เดอร์ที่สมบูรณ์ แผนภาพการผลิตชิ้นส่วน Slider ดังแสดงในรูปที่ 1.1



รูปที่ 1.1 แผนภาพแสดงการผลิตสไลด์เดอร์ที่สมบูรณ์

1.2 การผลิตชิ้นงาน สไลด์เดอร์ ในส่วนของการเจาะ, การลบคม และการคว้านในปัจจุบัน

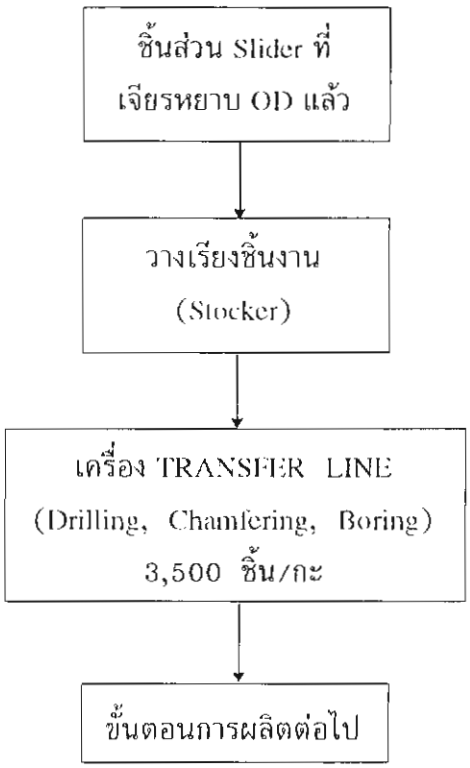
กระบวนการทำงานของเครื่องเจาะ เครื่องลบคม และเครื่องคว้าน มีขั้นตอนการทำงานดังรายละเอียดที่แสดงในรูปที่ 1.2



รูปที่ 1.2 แผนภูมิแสดงกระบวนการเจาะรู ลบคมรู และคว้านรูเจาะชิ้นส่วนสไลด์เดอร์ในปัจจุบัน

ปัญหาในการทำงานของระบบเก่า คือ ปัญหาคุณภาพชิ้นงานไม่ได้ตามมาตรฐาน ซึ่งขนาดรูของชิ้นงานต้องมีความถูกต้องแม่นยำสูงมาก

กระบวนการเจาะรู ลบคมรูเจาะ และคว้านรูเจาะ ในปัจจุบันใช้เครื่องจักร 3 สถานีด้วยกัน คือเครื่องเจาะ (5 เครื่อง) กำลังผลิต 520 ชิ้น/กะ/เครื่อง เครื่องลบคม (1 เครื่อง) กำลังการผลิต 3,000 ชิ้น/กะ/เครื่อง และเครื่องคว้านรูเจาะ (2 เครื่อง) กำลังการผลิต 1,250 ชิ้น/กะ /เครื่อง การทำงานเป็นแบบ Manual ใช้ Operator ทั้งหมดจับชิ้นงานเข้าและออกจาก Jig ของเครื่องจำนวน 5 คน โดยการทำงาน 2 กะ ซึ่งกระบวนการ 3 ขั้นตอนนี้ทำให้เกิดของเสียจากการผลิตคิดเป็น 50% ของจำนวนของเสียในสายการผลิตสไลด์เดอร์ทั้งสายการผลิต ซึ่งมีของเสียอยู่ประมาณ 2% แนวทางในการแก้ไขคือ การพัฒนาเครื่องจักรให้มีความแม่นยำสูงเพื่อเพิ่มประสิทธิภาพการผลิตของระบบโดยรวม กระบวนการทำงานของเครื่องจักรที่พัฒนาขึ้นใหม่นี้ประกอบด้วย การเจาะ การลบคม และการคว้านอัตโนมัติ ดังแสดงในรูปที่ 1.3



รูปที่ 1.3 แผนภูมิแสดงกระบวนการเจาะรู ลบคมรูเจาะ และคว้านรูเจาะชิ้นส่วนสไลด์เดอร์ ในระบบ Transfer Line ที่วิจัยและจัดสร้างขึ้น

1.3 การผลิตชิ้นงานสไลด์เดอร์จากเครื่อง TRANSFER LINE

จากการที่บริษัท ซันโย ยูนิเวอร์แซล อีเล็คทริค มีความต้องการขยายกำลังการผลิต จึงจำเป็นต้องเพิ่มประสิทธิภาพในการทำงานของเครื่องจักรดังนั้นจึงได้คิดริเริ่มในการวิจัยเครื่องจักรที่เป็นระบบอัตโนมัติสมบูรณ์แบบขึ้นเป็นระบบ Transfer Line ซึ่งมีกำลังการผลิตประมาณ 3,500 ชิ้น/กะ

การทำงานของชุด Transfer Line ที่ได้ออกแบบขึ้นมาใหม่นี้เป็นการทำงานแบบอัตโนมัติที่สมบูรณ์แบบ ทั้งการทำงานของแต่ละสถานีและการขนย้ายชิ้นงานระหว่างสถานีการทำงาน

ชิ้นงานที่ Stocker จะถูกมือจับ ของชุดขนย้ายชิ้นงาน (Transfer Arm) จับชิ้นงานมาไว้ที่ตำแหน่งเจาะทีละ 6 ชิ้นงาน แล้วเริ่มทำการเจาะ มือจับที่ประกอบอยู่กับรางเลื่อนที่อยู่ด้านบนของชุดเครื่องจักรจะทำการจับชิ้นงานแล้วเคลื่อนย้ายชิ้นงานจากตำแหน่งของชุดเจาะไปยังชุดลบคมชิ้นงาน

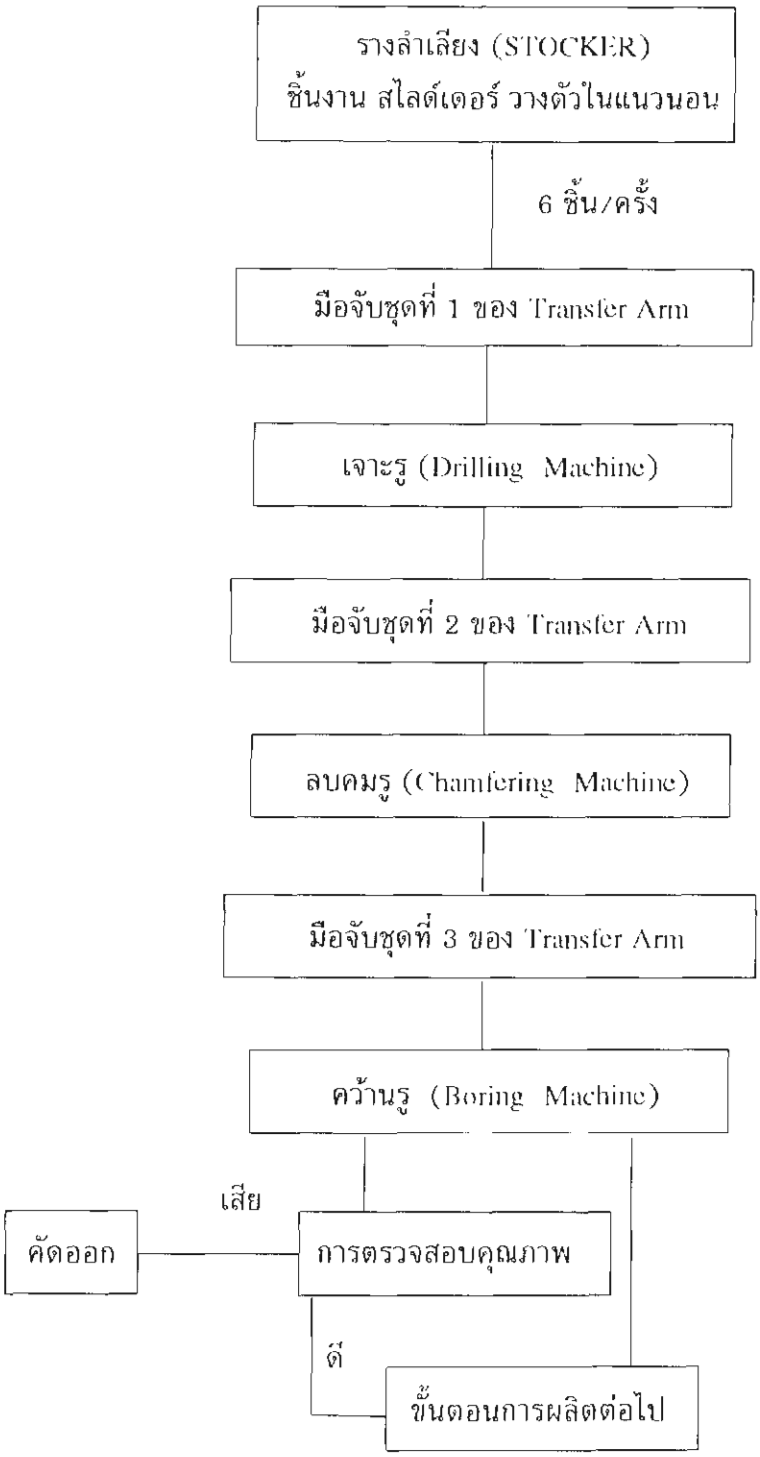
หลังจากกระบวนการเจาะเสร็จสิ้นลง มือจับจะจับและย้ายชิ้นงานจากตำแหน่งเจาะไปยังตำแหน่งลบคม พร้อมกับหยิบชิ้นงานใหม่เข้ามาที่ตำแหน่งเจาะด้วย ชุดหัวลบคมมี 2 ชุด ในการลบคม หัวลบคมจะเคลื่อนที่เข้าไปหาชิ้นงานทั้ง 2 ด้านของรู เพื่อลบคมที่เกิดจากการเจาะ

เมื่อชิ้นงานผ่านกระบวนการลบคมแล้ว มือจับจะเคลื่อนที่ไปจับชิ้นงานเพื่อเคลื่อนย้ายไปยังตำแหน่งคว้าน จังหวะนี้มือจับอีกชุดก็จะจับชิ้นงานจากชุดเจาะมาที่ Jig ของชุดลบคมพร้อม ๆ กันไปด้วย

การทำงานทั้งหมดเป็นการทำงานในลักษณะต่อเนื่อง โดยมีการควบคุมจากเครื่อง PLC ในลักษณะของ Open Loop Mode

ส่วนขั้นตอนการทำงานของเครื่อง TRANFER LINE แสดงในรูปที่ 1.4

ขั้นตอนการทำงานของเครื่อง Transfer Line



รูปที่ 1.4 แสดงขั้นตอนการทำงานของเครื่อง Transfer Line

1.4 กระบวนการทำงานของเครื่อง TRANSFER LINE

ก. กระบวนการป้อนชิ้นงานเข้าสู่เครื่อง Transfer Line

วัตถุประสงค์ :

เพื่อทำการป้อนชิ้นงานสไลด์เดอร์ครั้งละ 6 ชิ้นเข้าสู่กระบวนการเจาะ

กระบวนการ :

ชิ้นงานสไลด์เดอร์ที่บรรจุอยู่ภายใน STOCKER ทั้ง 6 ชุด เป็นชิ้นงานที่พร้อมเข้าสู่กระบวนการผลิต ลักษณะของ STOCKER เป็นแมกกาซีนในแนวตั้ง มีมุมเอียงประมาณ 15 องศา เพื่อให้ชิ้นงานสไลด์เดอร์เคลื่อนที่ออกจากแมกกาซีนได้ ชิ้นงานสไลด์เดอร์ที่จะถูกลำเลียงออกมาในแนวนอน ซึ่งสายลำเลียงจะมีทั้งหมด 6 สายการลำเลียง และชิ้นงานสไลด์เดอร์ทั้ง 6 ชิ้นจากสายลำเลียง จะถูกจับด้วยมือจับทั้ง 6 ชุดของ Transfer Arm ตำแหน่งที่ชิ้นงานสไลด์เดอร์ถูกจับคือบริเวณหัว-ท้ายของชิ้นงาน เพื่อเข้าสู่กระบวนการเจาะ (Drilling) ในขั้นตอนต่อไป ลักษณะของชิ้นงานสไลด์เดอร์เป็นเหล็กหล่อตันทรงกระบอกขนาดเส้นผ่าศูนย์กลาง 21.3 mm ความยาว 25 mm.

รูปที่ 1.5 แสดงชุด STOCKER