Chapter 23 Tape Rewind – PID+

Toilet Paper Rewind Lab

The Tape Rewind Lab began with the idea of rolling up toilet paper. A bad idea. The stuff breaks with the least tug and the system used stepper motors, not bad but controlled only in discrete angle increments. The lab was designed, built and programmed. It worked kinda but not well enough to continue with the design.



The design of a two-axis motion system with feedback control between the two has been an objective of a lab experience for some time. This lab involved two stepper motors and a PID loop feedback between the two. Tension control is to be maintained. It is anticipated that many rolls of toilet paper will find the bottom of a waste can because of this lab.

While the stepper lab was successful to the point that a MS student received his degree from programming it, the design was not satisfactory. The numerous problems keeping this system operational was enough to cause the construction of the winder shown below.



Later Motor Speed Control Design This design uses two dc gear motors to transfer a cloth tape from one reel to another with a dancer roll in the middle. This design has the advantage of position and speed control of two motors and the PID control of the tension with the dancer roll.



This design was developed before the design in Ch. 21 – Single Axis Gear Motor Speed Control. This diagram uses 24 V inputs instead of the 5 V input for the encoder.





The wiring for the motor from the PLC and motor drive board is shown below:

Separate wiring acquires the POT information from the dancer arm. While this lab in all regards is a double of the previous Lab 21, there are several changes. First, the need for position control may be needed and is not added at present. The design of this lab pre-dates information for Lab 21 in that the 5 V interface was not known at that time. From this lab, that information was obtained and that lab was built. Also, this lab has several possible outcomes:

- 1. The second motor can run as a percent of the first motor and a percent of the POT.
- 2. The second motor can run independent of the first motor at a set ramp speed.
- 3. Both motors can run independent and be separately ramped.
- 4. The second motor can run as a direct follower of the POT only.

Any of these schemes are acceptable under certain criteria. Any could be used by industry.

Since this is a process, the inputs for starting and stopping the system plus each motor independently have been added. These are I0.6 and I0.7 for the motors.

PLC_2 [CPU 1	214C DC/DC	/DC]				🔍 Properties	🔄 Info	i Diagnostics	78
General	IO tags	Syst	em constants	Texts					
PROFINET inte	erface [X1]	~		Channel addr	ess:	10.0			
✓ DI 14/DQ 10				Input filt	ers:	20 microsec			•
General									
👻 Digital inp	uts				-				
Channe	elO	≡				Enable rising edge dete	ection		
Channe	el1			Eventname	:				
Channe	el2		Hardv	vare interrupt	:				
Channe	el3			Priority]		
Channe	el4						_		
Channe	el5								
Channe	el6	-							
Channe	el7								
Channe	el8	-				Enable falling edge det	ection		
Channe	el9			Event name	:				
Channe	el10		Hardv	vare interrupt					
Channe	el11			Priority	. —		1		
Channe	el12			rnonty	·				
Channe	el13								
 Digital out 	tputs								
Channe	elO					Enable pulse catch			
e1	1 a								

Configuration of Digital Inputs for high-speed pulse input:

Analog address of the POT when added to the design. The POT is not presently included in the program.

General I	O tags Syst	em constants	Texts					
Channel2 Channel3	~	//O addresses						
Channel4 Channel5		Input addresses						
Channel6 Channel7		Start address: 64 End address: 67 Organization block: (Automatic update)		64				
Channel8 Channel9				(Automatic update)				
I/O addresses ▼ AI 2	;		Process image:	Automatic update				
General	4							
 Analog inputs Channel0 	s							
Channel1 I/O addresses								
▼ AQ1 Signal boar	ď							
Catalog int	formation							

Configuration of the High Speed Counter:

General IO tags	Syste	em constants Texts			
I/O addresses • High speed counters (HSC)	^	> General			
✓ HSC1 General Function		Enable	nable this high speed counter		
Reset to initial values					
Event configuration		Project information			
Hardware inputs					
I/O addresses		Name: HSC	U		
 HSC2 		Comment			
General	1	comment.	<u>^</u>		
Function					
Reset to initial values	_				
Event configuration	=				
Hardware inputs					
I/O addresses					

General IO tags	System constants Texts	
I/O addresses	Eunction	
 High speed counters (HSC) 		
➡ HSC1		
General	Type of countin	g: Count
Function	Operating phase	e: Single phase
Reset to initial values	operating pros	
Event configuration		
Hardware inputs	Counting direction is specifie	ed
I/O addresses	D	y: Oser program (internal direction control)
	Initial counting direction	n: Count up
General		
Function	Frequency measuring period	d: sec 💌
Reset to initial values	=	··· [
Event configuration		
Hardware inputs		
I/O addresses		

I/O addresses	^	,	Hardware inputs				
 High speed counters (HSC) 		<u> </u>					
General			Clock generator input:	%10.0	100 kHz on-board input		
Function							
Reset to initial values							
Event configuration							
Hardware inputs			Direction input:				
I/O addresses			Direction input:				
✓ HSC2							
General		-					
Function							
Reset to initial values		-	Sync input:				
Event configuration							
Hardware inputs							
I/O addresses							

Configuration of the PWM Outputs:





Organization block: --- (Automatic update)

Process image: Automatic update

HSC3

HSC4
 HSC5
 HSC6

PTO1/PWM1
 General

Pulse generators (PTO/PWM)

Parameter assign...

Hardware outputs I/O addresses \equiv

....

....

Cyclic interrupt [OB30]		Reperties	🗓 Info 🔒 🗓 Diagnostics	
General Texts				
General Information	General			
Time stamps				
Compilation	Name:	Cyclic interrupt		
Protection	Constant name:	OB_Cyclic interrupt		
Attributes	Type:	OB		
Cyclic interrupt	Super-	C alla internet		
	Event class:	Cyclic Interrupt		
	Language:	LAD		-
	• Number:	30	\$	
		🔘 Manual		
	•	 Automatic 		
	Process image part number			
	PIP:	None		

General Texts	
General	Curlis interment
Information	
Time s <mark>t</mark> amps	
Compilation	Cyclic time (ms): 100
Protection	Phase offret (ms):
Attributes	These obset (his).
Cyclic interrupt	

The programming is next. There are many statements to be added to complete the actual program desired. The suggestions listed above may be used as a starting point for your program.











The watch table below was used in troubleshooting the proram above. There may be additions necessary as you complete the program.

Da	Dancer3_V16 PLC_2 [CPU 1214C DC/DC/DC] Watch and force tables Watch table_1										
Ý	# # 1 In 2, 2 2 m m m										
	i	Name	Address	Display format	Monitor value	Modify value	9	Comment			
1		"Motor_PWM_R"	%QW1000	DEC+/-		500	🗹 🔔				
2			%QD1002	DEC+/-		10000	🗹 📐				
з		"in"	%M10.0	Bool		TRUE	🗹 📐				
4		"out"	%M10.1	Bool							
5		"Out_R_To_Motor"	%MW22	DEC+/-		50	🗹 📐				
6		"In_Pulse_R"	%ID1000	Hex		16#0000_0000	🗹 🔺				
7		"In_Pulse_L"	%ID1004	Hex							
8		"SP_R"	%MD30	Floating-point nu		50.0	🗹 📐				
9		"PID_Mode_R"	%MW24	DEC+/-		3	🗹 📐				
10		"SP_L"	%MD42	Floating-point nu		50.0	🗹 📐				
11		"PID_Mode_L"	%MW52	DEC+/-		3	🗹 📐				
12		"test"	%MW62	DEC+/-		500	🗹 📐				



This work is licensed under a Creative Commons Attribution 4.0 International License.