

TML loop with motion profiles and events

Application Note

Easy Motion Studio II

Your Next Intelligent **Move**



Table of contents

1. Application description	3
2. Application flow chart	3
3. EasyMotion Studio II implementation	3
3.1 Detailed description of the EasyMotion Studio II implementation	3

1. Application description

This application note outlines the procedure for constructing a loop utilizing a label and the "GOTO" TML instruction. For illustrative purposes, this loop incorporates two motion profiles, each triggered by distinct time events.

2. Application flow chart



Figure 1 – Application structure

3. EasyMotion Studio II implementation

This application program features an infinite loop that executes five rotations in the positive direction followed by five rotations in the negative direction. A one second pause is implemented between the two movements to ensure precise timing and control.

3.1 Detailed description of the EasyMotion Studio II implementation

The "Jumps and Function Calls" dialogue allows controlling the TML program flow through unconditional or conditional jumps and unconditional, conditional or cancelable calls of TML function. In this application example, it was used to create the infinite loop, using a label and a "GOTO" TML instruction.

		😡 🗆 Work offline Set all	apps offline 🥻 🏄 👔	0 /	1 12 7	K 👘	A	
ication tree 🔹 🔻								
P091.APN6 TML loop with motion profiles and events	Trapezoidal Profiles	Search					0/0	
TML loop with motion profiles and events - [1]	∧ S Curve	Motion Loop: //Define label	Motion Loop				7 12	
S Setup	W bi					_	一個	
M Motion	NO PVT	Jumps and Function Calls	•			×		
— 🚯 Homings	Ø⊢ External	O Go to	Address / Label					
- 🕑 Functions	유용 Electronic Gearing	⊖ ¢all	100 M		• (1 = 0)	≥ 0. 1		
CAM Tables	12 Electronic Camming	 Cancelable call 					-	
	Motor Commands							
Memory settings	Position Triggers	Return from function						
	🔂 Homing	Return from interrupt					F	
	¶A Contouring	Abort cancelable call						
	T/ [™] Test	Insert label Motion_Loop					H	
	In Events			OF	Coursel	Unite		
	I Jumps and Function Calls			UK	Cancer	netb	1	
	리 Inputs / outputs						-	
	15 16-bit integer							
	32-bit Long or Fixed							
	* Arithmetic						F	
	🔁 Data Transfer Between Axes						5	
	Send Data To Host							
	MISC Miscellaneous							
	🐓 Interrupt Settings							
	but Eron Taut							

Figure 2 – Creating Motion_Loop label

The GOTO instruction is used to create a loop by jumping back to the specified label, in this case, Motion_Loop. This ensures the continuous execution of the sequence of commands within the loop. This process is managed through the Jumps and Function Calls dialog, which allows the program to control the flow of execution and create repetitive cycles effectively.

	- 2 2 = 1 1	😡 🗌 Work offline Set al	ll apps offline 🥻 🙎 🕨 Օ	🐔 🛠 🚳 🚬	
ication tree 🔹 🔻	f				
Bernard State		Search			
TML loop with motion profiles and events - [1]	∧ S Curve	Motion_Loop: //Define labe	I Motion_Loop		
S Setup	MPT	GOTO Motion_Loop; //Bra	nch to Motion_Loop		
M Motion	NO PVT				
- 🔛 Homings 🛛 🖓 – External		interpretation cars			
- D Functions	유용 Electronic Gearing	Go to Address / Label Motion_Loop			
CAM Tables	Electronic Camming	Cancelable call	□ #	* 10 20	
	Motor Commands	Concellable can			
Memory settings	Position Triggers				
	Homing	Return from function			
	M Contouring	Return from interrupt			
	T∕~ Test	O Abort cancelable call			
	IC+ Events	O Insert label			
	N Jumps and Function Calls		ок	Cancel Help	
	Inputs / outputs				
	16 16-bit Integer				
	32 32-bit Long or Fixed				
	* Arithmetic				
	Po Data Transfer Between Axes				
	Send Data To Host				
	MISC Miscellaneous				
	🗧 Interrupt Settings				
	but Error Tout				

Figure 3 – Jump to Motion_Loop label

The "Motion – Trapezoidal Profiles" dialog allows the programming of a position or speed profile with a trapezoidal speed shape, resulting from limited acceleration. In this example, the "Motion – Trapezoidal Profiles" dialog is used to control the two movements.

When using a position profile, the load/motor is controlled in position mode. You can specify either an absolute position to reach or a relative position increment, along with the slew speed and acceleration rate. In relative mode, the target position can be calculated in two ways: standard or additive.

Key Points:

- 1. Trapezoidal Profiles:
 - The "Motion Trapezoidal Profiles" dialog is used to create profiles where speed follows a trapezoidal shape due to limited acceleration.
- 2. Position Control:
 - The motor/load is controlled in position mode.
 - You can specify an absolute position to reach or a relative position increment.
- 3. Profile Parameters:
 - Define the slew speed and acceleration rate for the profile.
- 4. Relative Mode Calculation:
 - o In relative mode, the target position can be computed in two ways:
 - **Standard**: The new position is calculated based on the initial position.
 - Additive: The new position is calculated by adding the increment to the current position.

By following these guidelines, one can effectively use the "Motion – Trapezoidal Profiles" dialog to program precise and controlled motion profiles.

The configuration sets up a motion profile that will cause the motor to move 5 rotations in the positive direction using relative positioning.

roject Application Communication Tools Control panels Utilities C	Help			
	1 2 V C Work Onine (Set an approximate) Y 2 Y V Y			
DOI: 100/ This lass with motion profiles and sweets				
The loop with motion profiles and events	Search 0/			
Column And a profession events [1] A PT	Motion Loop: //Define label Motion Loop			
A M Mation	CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2]			
Homines Or External	CSPD = 16.66667; //siew speed = 500[/pm]			
P Functions S& Electronic Ge	CPOS = 10000C //position command = 5(rot) CPR: //position command is relative			
Internants Internants	MODE PP: 2			
CAM Tables D Motor Comm	UPD; // execute immediate			
Memory settings	IMC; WAIT: // wait for completion			
(Homing	GOTO Motion_Loop: //Branch to Motion_Loop			
TA Contouring	Trapezoidal Profiles X			
T/~ Test	Position Absolute			
IC+ Events	O Speed Relative Additive			
1 Jumps and F	alls Acceleration rate 1000.01 rad/s^2 -			
inputs / outp	Siew speed 500 rpm + 🗸			
15 16-bit Intege	Position increment 5 rot + 🖌			
32 32-bit Long	Generate new trajectory startion from actual values of position and speed reference			
*/ Arithmetic	Generate new trajectory starting from actual values of load/motor position and speed			
우ð Data Transfe				
Send Data To	Update immediately wait until motion is completed Update on event			
1985: Miscellaneou	C. WERE AN ANY ANY			
🗧 Interrupt Set	Setue motion data, but don't start execution			
txt Free Text				

Figure 4 – Inserting a trapezoidal position profile

The "Events" dialogue allows to define events. An event is a programmable condition, which once set, is monitored for occurrence.

The following actions can be connected to an event:

- stop the motion when the event occurs
- wait for the programmed event to occur
- exit from the wait loop after a set wait time

In the Events dialog, click on the dropdown menu to view the available event types and select "After a wait time." Once selected, a new field will appear where you can specify the wait duration. Enter the value 1 and choose s (seconds) from the time unit dropdown menu. Finally, click the "OK" button to confirm and apply the wait time event. This process inserts a one second wait time between the two motion profiles in the loop.



Figure 5 – Inserting a time event

In this example, the "Trapezoidal Profiles" dialog is used to configure and insert an additional motion profile that moves the motor 5 rotations in the negative direction using relative positioning.

	- i i i i i i i	🗊 🗌 Work offline Set all apps offline 🎤 🚊 🖡 🗿 🤗 🎉 🗙 🗊	A			
ication tree • 8						
P091,APN6 TML loop with motion profiles and events	A Trapezoidal Profiles	Search				
TML loop with motion profiles and events - [1]	∧ S Curve	Motion Loop: //Define lobal Mation Loop	0			
S Setup	M PT	/@netico.mofile				
M Motion	1 PVT	CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2]				
- 🕞 Homings	Ø− External	CSPD = 16.66667; //slew speed = 500[rpm] CPOS = 100001; //sneition command = 5(rpt]				
- Punctions	R& Electronic Gearing	lectronic Gearing CPR; //position command is relative				
- Interrupts	Electronic Camming	MODE PP: TUM: //wei Target Update Mode 1 UPD: // execute immediate IMC: WAIT: // wait for completion //Define event: After a wait time equal with value 1 s IRS 1000L: WAIT: // Wait until the event occurs //Position profile CACC: a 0.31831: //acceleration rate = 1000.0116ad/s^22				
CAM Tables	Motor Commands					
Memory settings	Position Triggers					
	A Homing					
	M Contouring					
	T/" Test					
	Ic+ Events	CSPD = 16.66667; //slew speed = 500[rpm]				
	N Jumps and Function Calls	CPOS = 10000E; //position command = 5[rot] — CPR: //position command is relative				
	리 Inputs / outputs	MODE PP:				
	16 16-bit Integer	TUM1: //set Target Update Mode 1 UPD: // execute immediate IMC: WATI: // wait for completion GOTO Motion_Loop: //Branch to Motion_Loop				
	32 32-bit Long or Fixed					
	* Arithmetic					
	2 Data Transfer Between Axes	Trapezoidal Profiles X				
	Send Data To Host	Position Absolute				
	MSC Miscellaneous	Speed Relative Additive				
		Acceleration esta				
	bit Free Text	Slew speed 500 mm x C2				
		Position increment 5 rot V				
		Generate new trajectory starting from actual values of position and speed reference Generate new trajectory starting from actual values of load/motor position and speed				
		Update immediately				
		Setup motion data, but don't start execution				

Figure 6 – Inserting a trapezoidal position profile

Following this motion, another wait event of 1 second will be defined, similar to the one shown in Figure 5.

upplication tree • 0 Implication tree • 0 Implicatin tree • 0 <			🔹 📮 🗋 Work offline Set all apps offline 🥻 🔹 🕨 🛛 🦸 💈 🤽 🛪 🔢	Ax			
P093 APN6 TML loop with motion profiles and events ↑ Trapezoidal Profiles Status ↑ Scurve Null loop with motion profiles and events - (1) ↑ Scurve Null loop with motion profiles and events - (1) ↑ Scurve Null loop with motion profiles and events - (1) ↑ Scurve Null loop with motion profiles and events - (1) ↑ Scurve Null loop with motion profiles and events - (1) ↑ Scurve Null loop with motion profiles and events - (1) ↑ Scurve Null loop with motion profiles and events - (1) ↑ Scurve Null loop with motion profiles and events - (1) ↑ Null loop with motion profiles and events - (1) Null loop with motion profiles and events - (1) ↑ Scurve Null loop with motion profiles and events - (1) Null loop with motion profiles and events - (1) ↑ Scurve Null loop with motion profiles and events - (1) Null loop with motion profiles and events - (1) ↑ Null loop with motion profiles and events - (1) Null loop with motion profiles and events - (1) Null loop with fragges (1) Null loop with fragges (1) Null loop with motion profiles and event loccurs Null loop of fixed	plication tree	. #					
 TML loop with motion profiles and events - [1] A S Curre M Muipion Puri External External	P091.APN6 TML loop with motion profiles and ender P091.APN6 TML loop with motion profiles and profiles	ents 🔿 Trapezoidal Profiles	Search	0/0			
S Stup // A PT M Motion My PVT E Monings My Extra and the system of	 TML loop with motion profiles and events - i 	1] A S Curve	Motion Loop: //Define label Motion Loop	12			
M Motion [®] p vr CACC = 0.31837: //acceleration rate = 100001[rad(s ⁴ /2) [®] Interrupts [®] External [©] External [®] M Motion [®] External [©] External [®] Functions [®] External [©] External [®] Motion: [®] External [©] Dotto: [®] CAM Tables: [®] Motion: [®] Motion [®] Motion: [®] Motion: [®] Motion [®] Motion: [®] Motion: [®] Motion [®] Motion:	- S Setup	M PT	//Position profile				
Promings Protectional Provide Functions Provide Provide	 M Motion 	N PVT	CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2] CSPD = 16.66667; //dew speed = 500[rbm]	t			
	- 🕒 Homings	0- External	CPOS = 10000L; //position command = 5[rot]	ŧ			
Interrupts Image: Sectronic Camming TUM*I: //sectronic Mediate Image: Sectronic Camming Motor Commands UPD: // vecute immediate Memory settings Position Triggers /// Contouring // Contouring // Contouring // Contouring // Test CACC = 0.31831: //acceleration rate = 1000.01[rad/s*2] CSPD = 16.68667: //acceleration rate = 0000.01[rad/s*2] CSPD = 16.68667: //acceleration rate = 0000.01[rad/s*2] CSPD = 16.68667: //acceleration rate = 0000.01[rad/s*2] CSPD = 16.68667: //acceleration rate = 0000.01[rad/s*2] CSPD = 16.68667: //acceleration rate = 0000.01[rad/s*2] CSPD = 16.68667: //acceleration rate = 0000.01[rad/s*2] CSPD = 16.68667: //acceleration rate = 0000.01[rad/s*2] CSPD = 16.68667: //acceleration rate = 0000.01[rad/s*2] CSPD = 16.68667: //acceleration rate = 0000.01[rad/s*2] CSPD = 16.68667: //acceleration rate = 0000.01[rad/s*2] CSPD = 16.68667: //acceleration rate = 0000.01[rad/s*2] CSPD = 16.68667: //acceleration rate = 0000.01[rad/s*2] CSPD = 16.68667: //acceleration rate = 0000.01[rad/s*2] CSPD = 16.68667: //acceleration rate = 0000.01[rad/s*2] CSPD = 16.68667: //acceleration rate = 0000.01[rad/s*2] CSPD = 16.68667: //acceleration rate = 0000.01[rad/s*2] CSPD = 16.68667: //acceleration rate = 0000.01[rad/s*2] CSPD = 16.68667: //acceleration rate = 0000000 <tr< td=""><td>- 🔛 Functions</td><td>R Electronic Gearing</td><td> CPR: //position command is relative MODE PP. </td><td>×</td></tr<>	- 🔛 Functions	R Electronic Gearing	 CPR: //position command is relative MODE PP. 	×			
[™] CAM Tables [™] Position Triggers [™] Contouring [™] Position Triggers	- 👔 Interrupts	Electronic Camming	TUM1: //set Target Update Mode 1	18			
Memory settings	CAM Tables	Motor Commands	UPD; // execute immediate IMC: WAIT: // wait for completion				
Image: Section 2014 Image: Section 2014 Image: Sectin 2014 Image: Sectin 2014	Memory settings //Defi Memory settings //Defi Moming	//Define event: After a wait time equal with value 1 s	7				
Image: Section profile //Position profile Image: Section profile //Position profile Image: Section profile CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2] Image: Section profile CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2] Image: Section profile CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2] Image: Section profile CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2] Image: Section profile CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2] Image: Section profile CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2] Image: Section profile CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2] Image: Section profile CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2] Image: Section profile CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2] Image: Section profile CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2] Image: Section profile CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2] Image: Section profile CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2] Image: Section profile CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2] Image: Section profile CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2] Image: Section profile CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2] Image: Section profile CACC = 0.31831; //acceleration rate = 1000		G Homing	IRT 1000L: WAIT: // Wait until the event occurs	14			
CACC = 0.31831 //scceleration rate = 1000.01frad/s^2] CACC = 0.31831 //scceleration rate = 1000.01frad/scceleration rate = 1000.01frad/s^2] CACC = 0.31831		//Position profile					
IV Events CFOS = 10000L: //position command = 5(rot) IV Jumps and Function Calls CFPC //position command = 5(rot) III Inputs / outputs MODE PP; III 10puts / outputs TUME: //position command is relative III 10puts / outputs MODE PP; IIII 10puts / Mode PP; MODE PP; IIII 10puts		CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2] CSPD = 16.66667; //slew speed = 500[rpm] CPDS = 10000L; //position command = 51ot] — CPPL //position command is relative MODP PD-					
CPR: //position command is relative CPR: //position command is relative MODE PP: TUMT: //set Target Update Mode 1 UPD; // execute immediate MOC EPP: UPD; // execute immediate MOC WATE: // was time equal with value 1 s WATE: // was unit immediate							
13 11-bit / outputs 13 12-bit Integer 14 10-bit / execute immediate 15 23-bit Long or Fixed 15 10-bit / execute immediate 15 23-bit Long or Fixed 16 10-bit / execute immediate 17 Arithmetic 17 10-bit a for small time equal with value 1 s 17 10-bit a for small time equal with value 1 s 17 10-bit a for small time equal with value 1 s 17 10-bit a for small time equal with value 1 s 17 10-bit a for small time equal with value 1 s 17 10-bit a for small time equal with value 1 s 18 10-bit a for small time equal with value 1 s 19 10-bit a for small time equal with value 1 s 10 10-bit a for small time equal with value 1 s 11 10-bit a for small time equal with value 1 s 12 10-bit a for small time equal with value 1 s 13 10-bit a for small time equal with value 1 s 14 10-bit a for small time equal with value 1 s 15 10-bit a for small time equal with value 1 s 16 10-bit a for small time equal with value 1 s 17 10-bit a for small time equal with value 1 s 18 10-bit a for small time equal with value 1 s <t< td=""></t<>							
UDD: // recerct immediate // Define event. After a wait time equal with value 1 s IRT 1000L; // Define event. After a wait time equal with value 1 s IRT 1000L UDD: // Recerct immediate UDD: // Recerct		TUM1; //set Target Update Mode 1	2				
Soft Conjection (Constant Constant) Anithmetic Anithmetic Obsta Transfer Between Axes GOTO Motion_Loop: //Branch to Motion_Loop Miscellaneous Constant Con		32 32-bit Long or Event	UPD; // execute immediate IMC: WAITI: // wait for completion	~			
Image: State Transfer Between Axes Iff 10002 Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: State Transfer Between Axes Image: S		* Arithmetic	//Define event: After a wait time equal with value 1 s				
GOTO Motion_Loop: //Branch to Motion_Loop Miscellaneous Coverts Coverts	* Arithmetic ?3 Data Trace	27 Data Transfer Between Av	Ases WAITS // Wait until the event occurs	Ð			
Miscellaneous Country		Send Data To Host	GOTO Motion_Loop; //Branch to Motion_Loop				
I MARKED AND A COMPANY AND		MSC Miscellaneous					
		Interrupt Settings	Livens				
but Free Text After a wait time * 2		bit Free Text	After a wait time •				
After a wait time equal with Variable			After a wait time equal with Variable OK Help eip				

Figure 7 – Inserting a time event