1. Application description

This application note describes how to implement a homing routine, using a function that aligns the load to the middle of the working area.

The homing procedure follows the algorithm below:

- move negative until the negative limit switch is reached;
- stop and set the current position to 0 [IU];
- move positive until the positive limit switch is reached;
- move to the middle of the working area;
- set this position as the system home position.



Figure 1. Homing steps

2. Application flow chart





3. EasyMotion Studio implementation

🔁 Project		
へ へ М 🕦 🕩 📽 🖾 🗃 群	' 🕼 [씨, T/-] 📭 🔟 🎞 1일 3일 🎋 🍋 💐 MISC 🗲 641	1
🖃 🌇 Middle_Homing	Main	
Untitled Application	Call function Axis_Homing	the states
S Setup	1 CALL Axis_Homing; //Call function Axis_Homing	f#
M Motion		↓
Homing Modes		1
🖃 💼 Functions		~
Axis_Homing		\sim
🚺 Interrupts		{•}
CAM Tables		(1)
		8
1)	

Figure 3. Main section of the TML program



Figure 4. Function Edit

4. Detailed description of the EasyMotion Studio implementation

4.1. Motion section

The code in the "Motion" section was generated using the button marked with 1 in Figure 3. Clicking on this button, the "Jumps and Function Calls" dialogue will open.

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 functions. In this case, it was used to call the "Axis_Homing" function.

Jumps and Function	Calls				?	×
Goto Gotal	at address, label or at address set in variable ction rrupt call d:	Axis_Homing	is	O EQ O LT O GT	C NEQ C LEQ C GEQ	than O
	OK	Cancel Help				

Figure 5. How to call a TML function

4.3. Functions section

The "Functions" section allows creating, renaming or removing a function. In this case, it was used to create the "Axis_Homing" function that contains the functionality described in the first chapter.

Once a TML function is created, it will appear in the "Functions" section.

Middle_Homing Middl	Functions	
S Setup M Motion	Axis_Homing	Add Rename
In Homing Modes ☐ Functions 『 Axis_Homing Interrupts CAM Tables	Axis_Homing	

Figure 6. How to create a TML function

The code inside the "Axis_Homing" function was generated using the buttons marked with 1 to 6 in Figure 4. Clicking on those buttons, the following programming dialogues will open.

• The "Assignment and Data Transfer – 16 bit Integer Data" dialogue (4) allows different operations with the 16-bit integer variables / parameters / registers. Here it was used to set the LSACTIVE parameter value to 1. This is deactivating the default functionality of the hardware limit switches. For more details please check the" Drive special inputs - Limit Switches" application note.

Assignment & Data Transfer - 16 bit Integer Data	?	\times
Set 16-bit variable LSACTIVE		
With value / <u>1</u> 6 bit variable / label		
C With C grogram address set in pointer variable		

Figure 7. How to disable the default functionality of the limit switch inputs

• The "Interrupt Settings" dialogue (6) allows to activate and/or deactivate the TML (Technosoft Motion Language) interrupts. In this case, it was used to activate "int 6- LSP programmed transition detected" and "int 7- LSN programmed transition detected" interrupts routines.

Globally enable TML Interrupts Globally disable TML Interrupts	Enable/Disable	TMI. Internuota	Reset previous TML		1
TML Interrupts	Enable	Disable	interrupt requests		
int0 - Enable input has changed		F	E		
int1 - Shert-circuit	r -	Ê	E I		
Int2 - Software protections	E.	Г	E I	Details	
Int3 - Control error	Ē	E F	i i		1
Int4 - Communication error	F	Г	F I		1
Int5 - Position wraparound	E	F	Ē		1
Int6 - LSP programmed transition detected	×	F	X	Details	
Int7 - LSN programmed transition detected	×	Г	×	Details	
Int8 - Capture input transition detected	Г	Г	C 1	Details	T
Int9 - Motion is completed / in velocity	٣	Г			1
int10 - Time period has elapsed	1	Г	E L	Details	1
Int11 - Event set has occurred	E C	Г	Г		1
Int12 - Position trigger 1. 4 change detected	1	E	E		1
			w 1		
OK	Ca	nceli	lep		
OK witch Positive	Ca X	ncel i	iop		×
ok witch Positive erate limit switch interrupt on	Ca	Limit Swite	ielp	on	×
OK witch Positive reate limit switch interrupt on fit transition low -> high fit transition high -> low	Ca X	Limit Swite General C	tep	on	×
witch Positive witch Positive reate limit switch interrupt on transition low -> high transition high >> low encoder signals is stored in variable position computed from second enc at is stored in variable CAPPOS2.	Led from CAPPOS coder	Limit Swite Generat © When th first enc aignals i	hep h Negative e limit switch interrupt - transition low -> high transition high -> low ansition occurs, positi oder signals is stored in stored in variable CA	on computed f n variable CAF cond encode PPOS2	rom PPOS

Figure 8. Interrupt Settings dialogue

<u>**Remark**</u>: The limit switches interrupts were enabled to activate the position capture when the limit switches became active.

• The "Motion – Trapezoidal Profiles" dialogue (1) allows to program a position or speed profile. It was used to move the motor in the negative direction, as the first part of the homing procedure requires (move negative until the negative limit switch became active).

Motion - Trapezoidal Prot	files			?	Х
Acceleration rate =	C Position Speed 1000 -100	C Absolute C Relative C Additive	Execute © Immediate © On event	Then wai motion is completer Change Ev Edit Eve	: until d vent
େ Gen valu C Gen valu	erate new traject es of position and erate new traject es of load/motor	ory starting from actual speed reference ory starting from actual position and speed Cancel	C Setup motion don't start ex	n data, but kecution	

Figure 9. How to set a negative motion profile

• By default, when a limit switch became active, the drive stops the movement using a quick stop profile. In this case, the default functionality of the limit switches was deactivated, so a stopping mechanism is required, when the negative limit switch is reached, due to the above motion profile.

The stopping mechanism mentioned above, was implemented using the "Events" dialogue (3), that was set to wait until the negative limit switch became active and then, to stop the motor.

	Events			?	×	
	Set Event Change Even Edit Event	None			Ŷ	
	♥ Stop the mol ♥ Wait until the ♥ Exit from	tion when the ever e event occurs the wait loop after OK	nt occurs a time equal wit Cancel	h	Ţ	
Event Type		? ×	Event - Fu	nction of Input Status		? ×
When actual a Function of me Function of me After a wait tim Function of re Function of re Function of re Function of re Function of re	motion is completed ofor or load position ofor or load speed e lerence put status 22-bit variable value	Cancel Help	When C	2nd encoder index positive limit switch is triggered negative limit switch is triggered digital input INSALSN home input	goes	i nigh C low

Figure 10. Setting an event on the IN(3)/LSN digital input

• Once the negative limit switch is reached, the motion stops and the actual position is set to 0, using the "Motion - Motor Commands" dialogue (2).

Motion - Motor Commands	×
C Activate the control loops and PWM outputs (AXISDN) C Deactivate the control loops and PWM outputs (AXISDFF) C Stop motion with acceleration / deceleration set C Immediate update C Update on event Set position: C Set actival position to C Value	-
C variable	-

Figure 11. How to set position value

• The "Motion – Trapezoidal Profiles" dialogue (1) is used again, to reverse the motor and go positively until the positive limit switch is reached.

Motion - Trapezoidal Profiles	? ×
C Position C Absolute C Belative C Belative C Additive	Execute Then wait until motion is completed C On event Change Event Edit Event Edit Event
✓ Jog speed = 100 rpm ▼	
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 OK Cancel H	C Setup motion data, but don't start execution

Figure 12. How to set a positive motion profile

• The positive limit switch detection is done using the "Events" dialogue (3). This time it is set to wait until the positive limit switch became active and then to stop the motion.

Events Set Event Change E Edit Event	vent None			?	×
I Stop the Vait uni Exit fr	motion when the event il the event occurs rom the wait loop after a OK	occurs time equal wit Cancel	h Help	(
Event Type None When actual motion is completed Function of motor or load speed After a wait time Function of reference Function of reference Function of input status Function of a2bit values Note: Status Note:	f X OK Cancel Help	When C	encoder index 2nd encoder index positive limit switch is trigger negative limit switch is trigger digital input (N27LSP home input	ed red	goes C high C low Help

Figure 13. Setting an event on the IN(2)/LSP digital input

• The "Assignment and Data Transfer – 16 bit Integer Data" dialogue (4) is used to set the command position with the captured position (when the positive limit switch was reached).

Assignment & Dat	a Transfer - 16	bit Integer Data	?	Х
 Set 16-bit value 	ariable CPOS			
With vi	alue / 16 bit varia	able / label CAPPOS		
C With	C data C program C E2ROM	memory contents, located at address set in pointer variable		
C With	C low C high	part of 32-bit variable		
C With th	e inverse (-) of v	ariable		
O Using A	AND mask	h and OR mask h		
C With cl of data	necksum located in C	data memory between address program E2ROM and	h h	
C Set C C	data addre program E2ROM	nory contents, located at ess set in pointer variable with value/variable then increment the pointer variable		
	OK	Cancel Help		

Figure 14. Position stored in variable CPOS

<u>Remark</u>: If the limit switches interrupts are active, then each time one of the limits is reached the drive saves automatically the actual position in the CAPPOS internal variable.

• Since the positon was set to 0 IU when the negative limit switch was reached, the position captured now represents the exact distance (in encoder counts) between the two limit switches. So, the middle point can be computed by dividing this value to 2.

At the processor level, the division is done through repetitive subtractions. Since the captured position needs to be divided to half, the shifting operations can be also used. The shift to the right, with 1 bit is equivalent with the division with 2¹ and it is also executed faster than the division. That is why the shifting operation is used instead the division.

The shifting operations are available in the "Arithmetic Operations" dialog (5).

Arithmetic Operations	?	×
C Add to variable the value/variable		
C Subtract from variable the value/variable		
C Set PROD register with the product of variable		
and value/16 bit variable C Left << C Right>>	bits (015)	
C Divide 32 bit variable by the value of 16 bit variable		
C Shift C Left << C Right >> product register by bits (015)		
OK Cancel Help		

Figure 15. Compute the distance to the middle point

• Once the middle point coordinate is computed, the load is moved there using the "Motion – Trapezoidal Profiles" dialogue (1).

Motion - Trapezoidal Profiles	? ×
C Position C Position C Speed C Relative Acceleration rate = 1000 Irad/s ² 2 Slew speed = 100 Irpm Position to reach =	Iditive
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 OK Cancel	C Setup motion data, but don't start execution

Figure 16. Position Profile

<u>Remark</u>: The profile above is absolute, because the value computed above represents the position to reach (the absolute position) not the position increment.

• After the load reaches the middle of the working area, the respective position is set to 0 and represents the system homing point. This operation was done using the "Motion - Motor Commands" dialogue (2).

Motion - Motor Commands	×
C Activate the control loops and PWM outputs (AXISON)	
C Deactivate the control loops and PWM outputs (AXISOFF)	
C Stop motion with acceleration / deceleration set	
C Immediate update	
C Update on event	
Set position: © Set actual position to © variable	
C Set quick stop deceleration rate to C value	-
0K Cancel Help	

Figure 17. Reset position value

• Before returning from function, the "Assignment and Data Transfer – 16 bit Integer Data" dialogue (4) is used to restore the limit switches default functionality by setting the LSACTIVE variable value to 0 (for reference, see Figure 7).