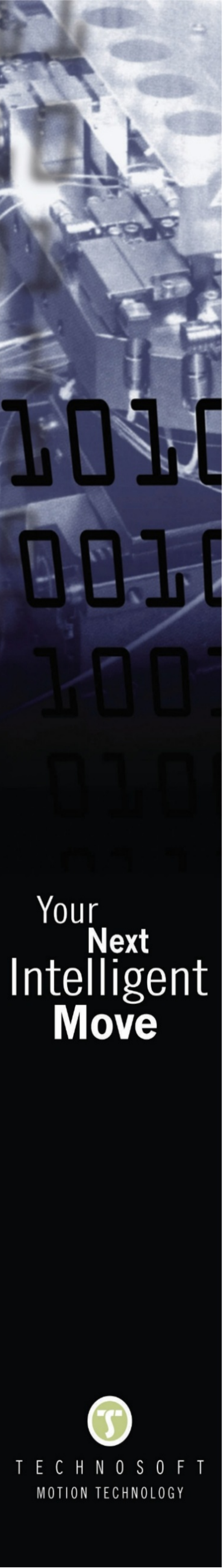


TECHNOSOFT

MOTION TECHNOLOGY

Custom homing modes implementation

Application Note

Easy Motion Studio II

Table of content

1. Application description

This application note outlines the steps to implement a custom homing routine that positions the load at the center of the working area.

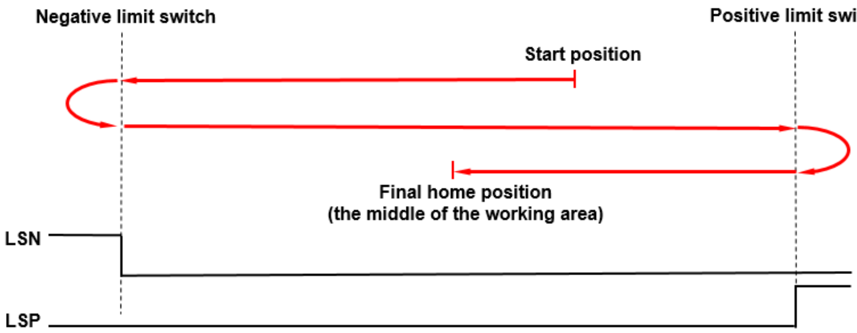
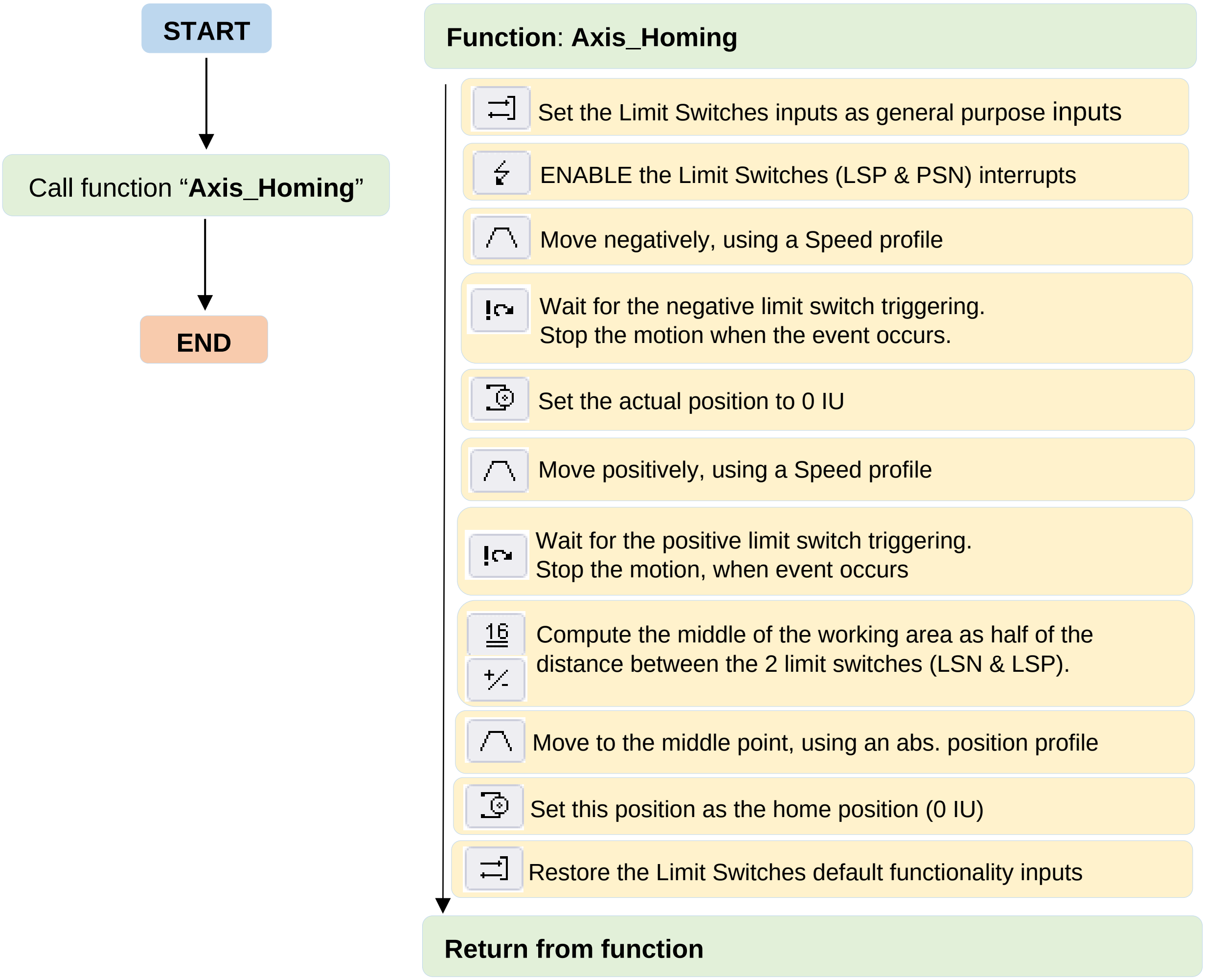


***Figure 1** - Homing steps*

The homing code will be stored in the drive memory and assumes the system is equipped with both positive and negative limit switch sensors.

2. Application flow chart



START

Call function “**Axis_Homing**”

END

Function: Axis_Homing

Set the Limit Switches inputs as general purpose inputs

ENABLE the Limit Switches (LSP & PSN) interrupts

Move negatively, using a Speed profile

Wait for the negative limit switch triggering.
Stop the motion when the event occurs.

Set the actual position to 0 IU

Move positively, using a Speed profile

Wait for the positive limit switch triggering.
Stop the motion, when event occurs

Compute the middle of the working area as half of the distance between the 2 limit switches (LSN & LSP).

Move to the middle point, using an abs. position profile

Set this position as the home position (0 IU)

Restore the Limit Switches default functionality inputs

Return from function

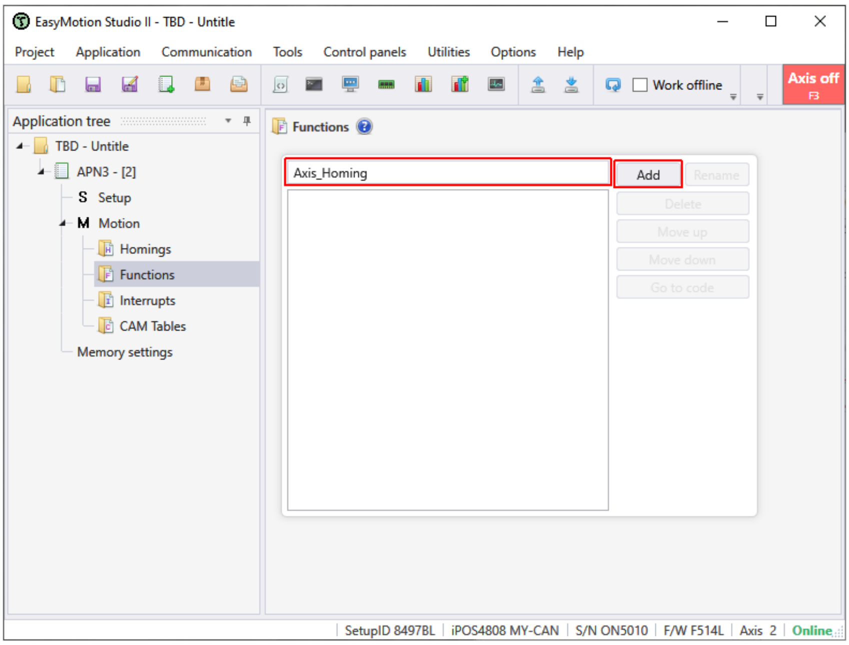
3. EasyMotion Studio II implementation

The application implementation in EasyMotion Studio II, includes two main parts:

1) The function (“Axis_Homing” in this case) that will include the homing procedure code;
2) The main program, in the “Motion” branch, which in this particular case will include only a function call instruction.

3.1 Homing procedure implementation using functions

The “Axis_Homing” function can be created using the wizard under the “Function” tree item.

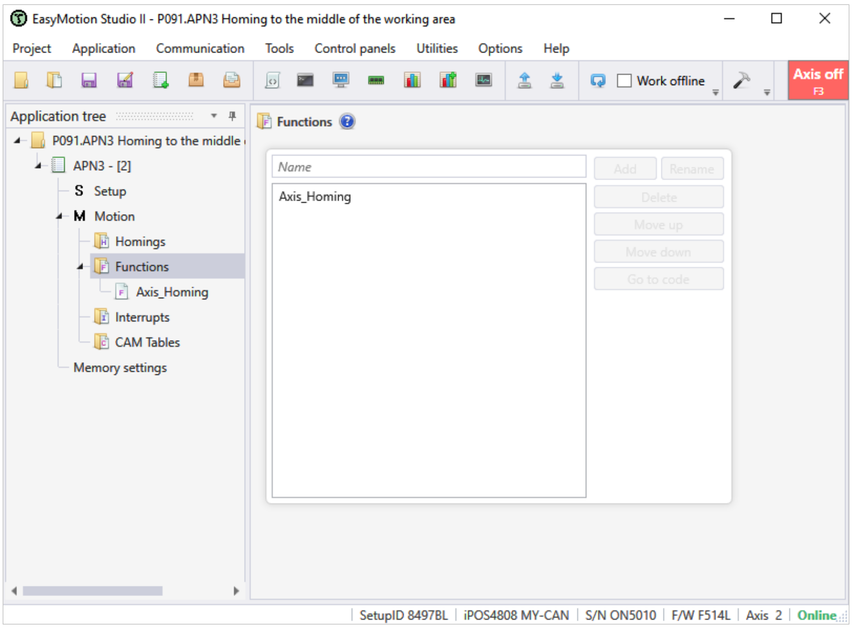


Axis_Homing

SetupID 8497BL | iPOS4808 MY-CAN | S/N ON5010 | F/W F514L | Axis 2 | Online

__Figure 2__ – Functions tree item options

Once added, the function will appear under the “Function” tree item.


EasyMotion Studio II - P091.APN3 Homing to the middle of the working area
Project Application Communication Tools Control panels Utilities Options Help
Work offline
Axis off F3
Application tree
P091.APN3 Homing to the middle
APN3 - [2]
S Setup
M Motion
Homings
Functions
Axis_Homing
Interrupts
CAM Tables
Memory settings
Functions
Name
Add
Rename
Delete
Move up
Move down
Go to code

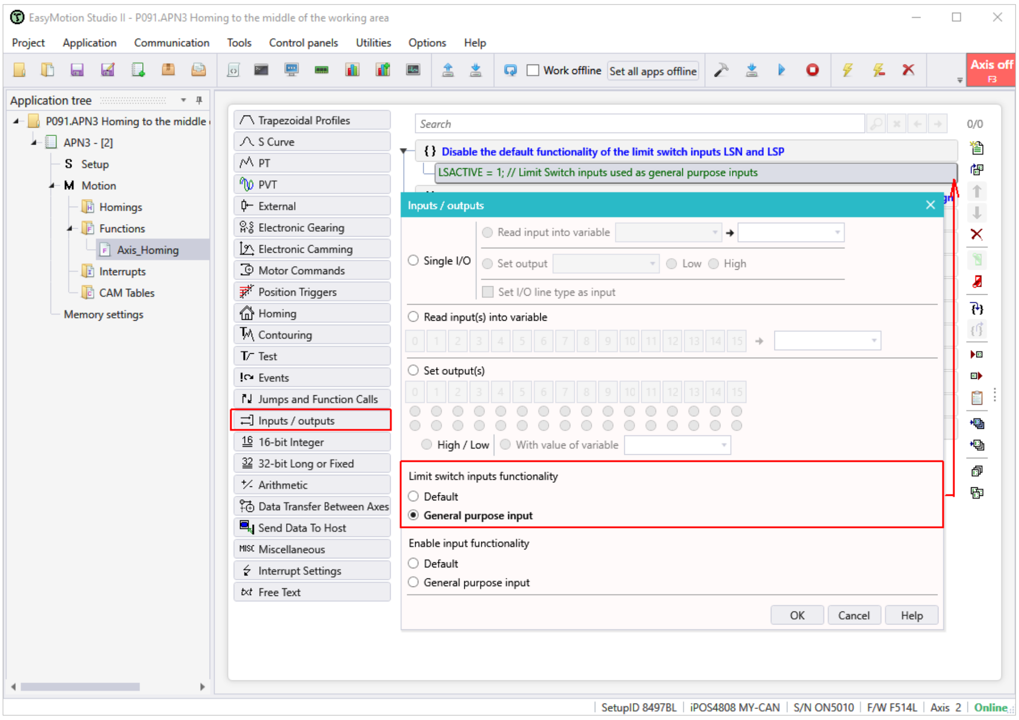
Axis_Homing

SetupID 8497BL | iPOS4808 MY-CAN | S/N ON5010 | F/W F514L | Axis 2 | Online

__Figure 3__ – Add the “Axis_Homing” function

Clicking on the function name will open its body, where the needed code can be added.

The homing functionality implementation will follow the flow chart in chapter 2. The first step is to set the limit switches as general purpose inputs. This can be done through the "Inputs/outputs" wizard.



{ } Disable the default functionality of the limit switch inputs LSN and LSP

```
LSACTIVE = 1; // Limit Switch inputs used as general purpose inputs
```

Inputs / outputs

○ Single I/O
○ Read input into variable
○ Set output ○ Low ○ High
□ Set I/O line type as input
○ Read input(s) into variable
○ Set output(s)
○ High / Low ○ With value of variable

Limit switch inputs functionality
○ Default
● General purpose input

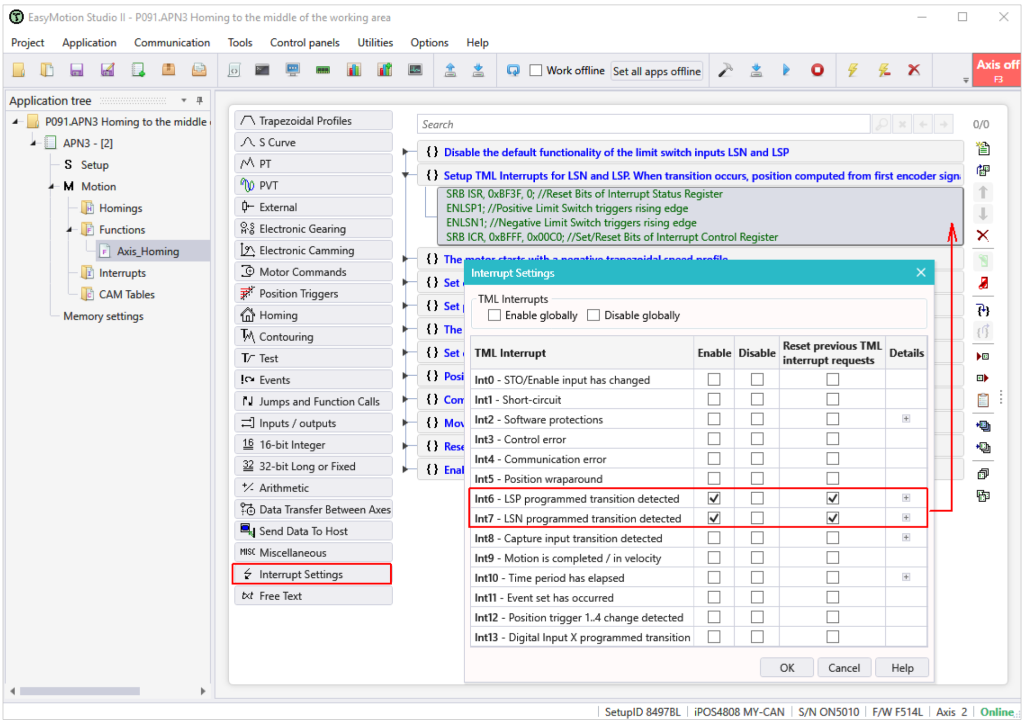
Enable input functionality
○ Default
○ General purpose input

OK Cancel Help

Figure 4 – Set the limit switches as general purpose inputs

Even the limit switches are set as general digital inputs, the associated interrupts can still be enabled. This feature will be used to capture the precise limit switches position.

The limit switches interrupts will be enabled and set through the "Interrupt Settings" wizard.



{ } Disable the default functionality of the limit switch inputs LSN and LSP

{ } Setup TML Interrupts for LSN and LSP. When transition occurs, position computed from first encoder sign

```
SRB ISR, 0xBF3F, 0; //Reset Bits of Interrupt Status Register
ENLSP1; //Positive Limit Switch triggers rising edge
ENLSN1; //Negative Limit Switch triggers rising edge
SRB ICR, 0xBFFF, 0x00C0; //Set/Reset Bits of Interrupt Control Register
```

{ } The motor starts with a negative trapezoidal speed profile

Interrupt Settings

TML Interrupts
□ Enable globally □ Disable globally

TML Interrupt	Enable	Disable	Reset previous TML interrupt requests	Details
Int0 - STO/Enable input has changed	□	□	□	
Int1 - Short-circuit	□	□	□	
Int2 - Software protections	□	□	□	+
Int3 - Control error	□	□	□	
Int4 - Communication error	□	□	□	
Int5 - Position wraparound	□	□	□	
Int6 - LSP programmed transition detected	☑	□	☑	+
Int7 - LSN programmed transition detected	☑	□	☑	+
Int8 - Capture input transition detected	□	□	□	+
Int9 - Motion is completed / in velocity	□	□	□	
Int10 - Time period has elapsed	□	□	□	+
Int11 - Event set has occurred	□	□	□	
Int12 - Position trigger 1..4 change detected	□	□	□	
Int13 - Digital Input X programmed transition	□	□	□	

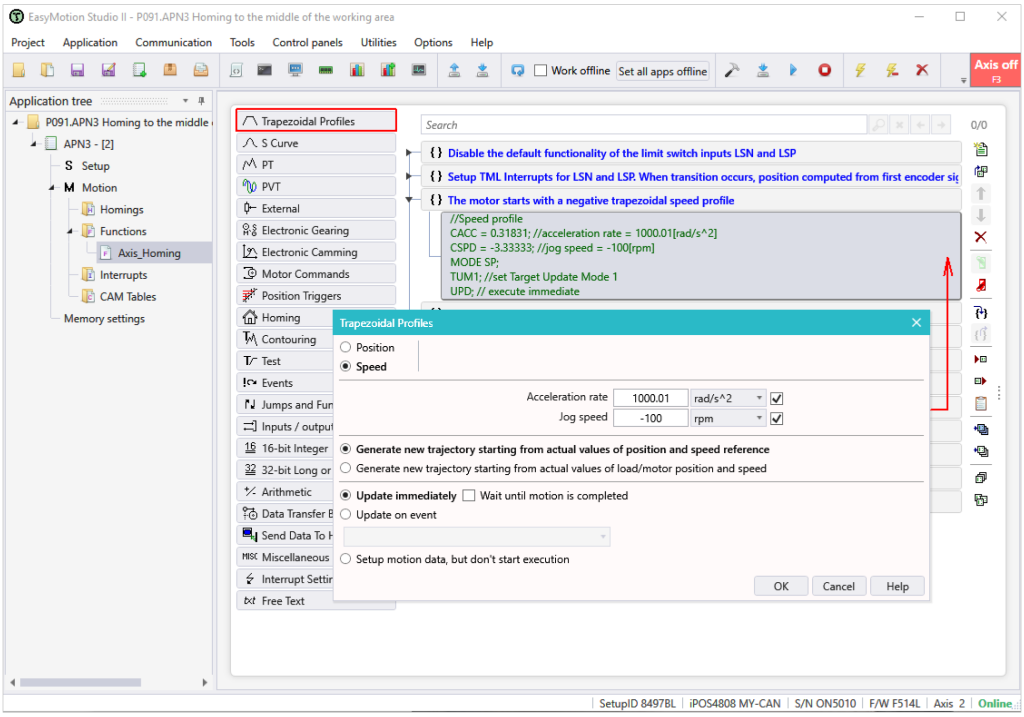
OK Cancel Help

Figure 5 – Enable and set the limit switches interrupts

The wizard is also allowing to set the LSN and LSP inputs transition (Low-High or High-Low) that will generate the associated interrupt.

The main idea behind this homing method is to move the load from one of the limits to the other, while measuring the distance. This allows computing the middle point coordinate.

The load can be moved using a position profile (with a position command higher than the max. possible distance between the 2 limit switches), without waiting for motion to be completed or using a speed profile. In this case, the axis is fist driven negatively, using a “Trapezoidal Profile – Speed”.


EasyMotion Studio II - P091.APN3 Homing to the middle of the working area

{ } Disable the default functionality of the limit switch inputs LSN and LSP

{ } Setup TML Interrupts for LSN and LSP. When transition occurs, position computed from first encoder sig

{ } The motor starts with a negative trapezoidal speed profile

```
//Speed profile
CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2]
CSPD = -3.33333; //jog speed = -100[rpm]
MODE SP;
TUM1; //set Target Update Mode 1
UPD; // execute immediate
```

Trapezoidal Profiles

○ Position
● Speed

Acceleration rate 1000.01 rad/s^2 ☑
Jog speed -100 rpm ☑

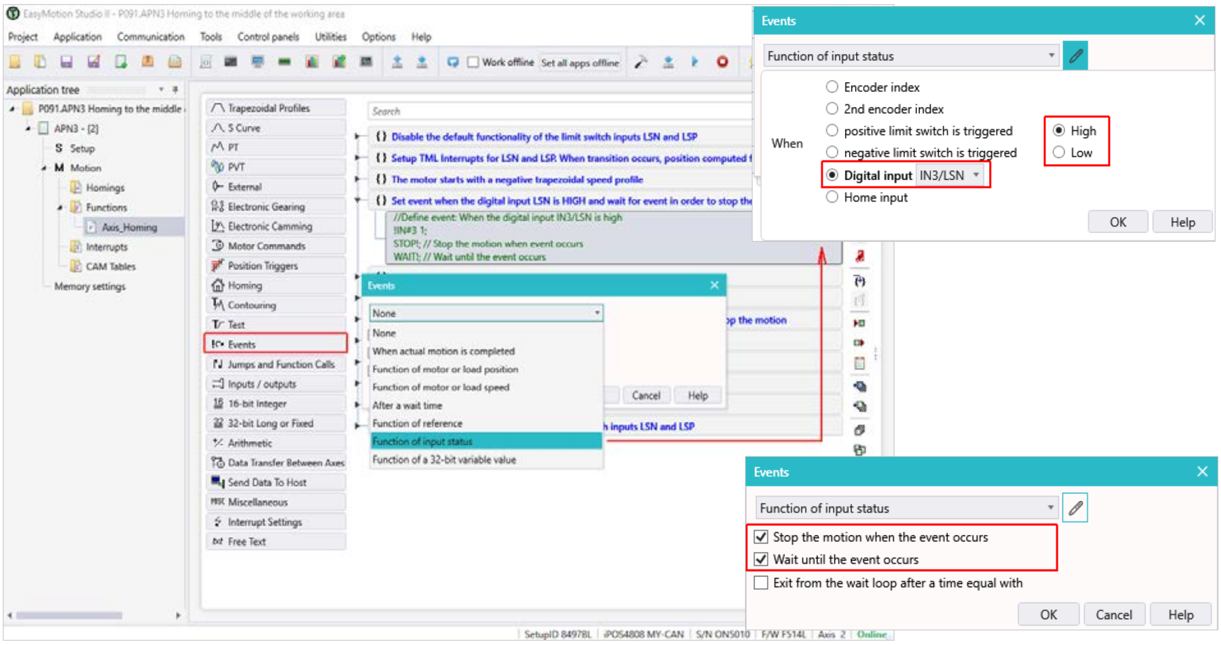
● Generate new trajectory starting from actual values of position and speed reference
○ Generate new trajectory starting from actual values of load/motor position and speed

● Update immediately □ Wait until motion is completed
○ Update on event

○ Setup motion data, but don't start execution

Figure 6 – Trapezoidal Profile - Speed wizard

The “Events” wizard was used to set the drive to detect when the negative limit switch digital input goes high and stop the movement.


{ } Disable the default functionality of the limit switch inputs LSN and LSP

{ } Setup TML Interrupts for LSN and LSP. When transition occurs, position computed f

{ } The motor starts with a negative trapezoidal speed profile

{ } Set event when the digital input LSN is HIGH and wait for event in order to stop the

```
//Define event: When the digital input IN3/LSN is high
!IN#3 1;
STOP!; // Stop the motion when event occurs
WAIT!; // Wait until the event occurs
```

Events

Function of input status

When
○ Encoder index
○ 2nd encoder index
○ positive limit switch is triggered
○ negative limit switch is triggered
● Digital input IN3/LSN
○ Home input

● High
○ Low

Events

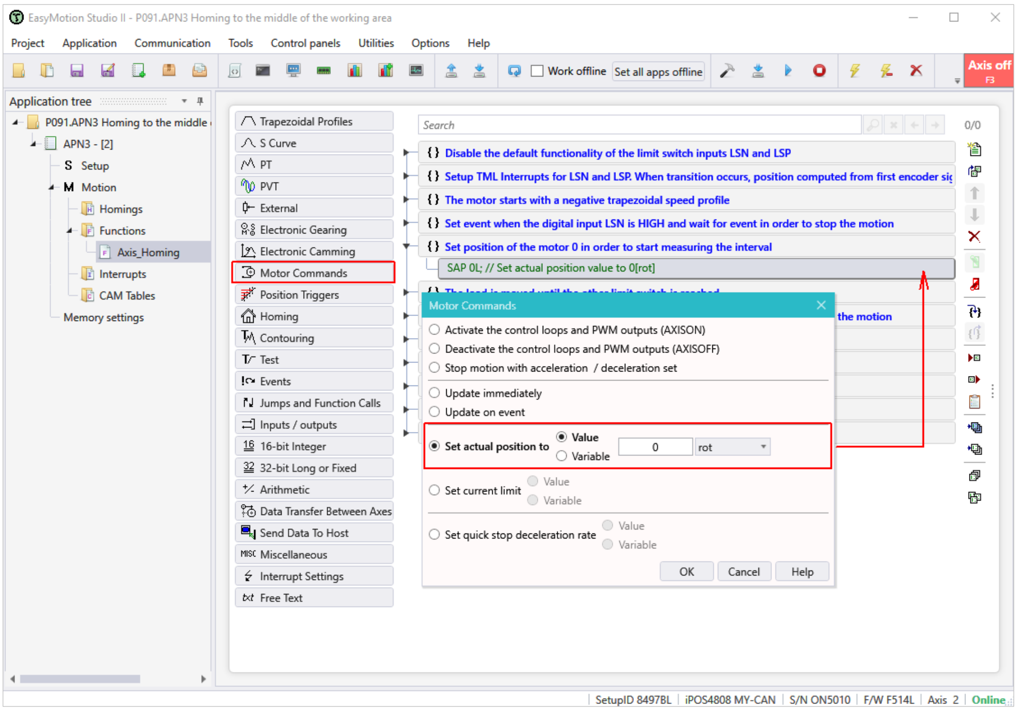
Function of input status

☑ Stop the motion when the event occurs
☑ Wait until the event occurs
□ Exit from the wait loop after a time equal with

Figure 7 – Events wizard

Remark More details on the wizard dialogues options are available in the EasyMotion Studio II help topics that can be open by clicking on the “Help” button available in all the wizard dialogues.

After reaching the negative limit, the actual position is set to 0 IU, using the “Motor Commands” wizard. This will make the middle point coordinate computing easier.



{ } Disable the default functionality of the limit switch inputs LSN and LSP

{ } Setup TML Interrupts for LSN and LSP. When transition occurs, position computed from first encoder sig

{ } The motor starts with a negative trapezoidal speed profile

{ } Set event when the digital input LSN is HIGH and wait for event in order to stop the motion

{ } Set position of the motor 0 in order to start measuring the interval

```
SAP 0L; // Set actual position value to 0[rot]
```

{ } The load is moved until the other limit switch is reached

Motor Commands

○ Activate the control loops and PWM outputs (AXISON)
○ Deactivate the control loops and PWM outputs (AXISOFF)
○ Stop motion with acceleration / deceleration set

○ Update immediately
○ Update on event

● Set actual position to ● Value ○ Variable 0 rot

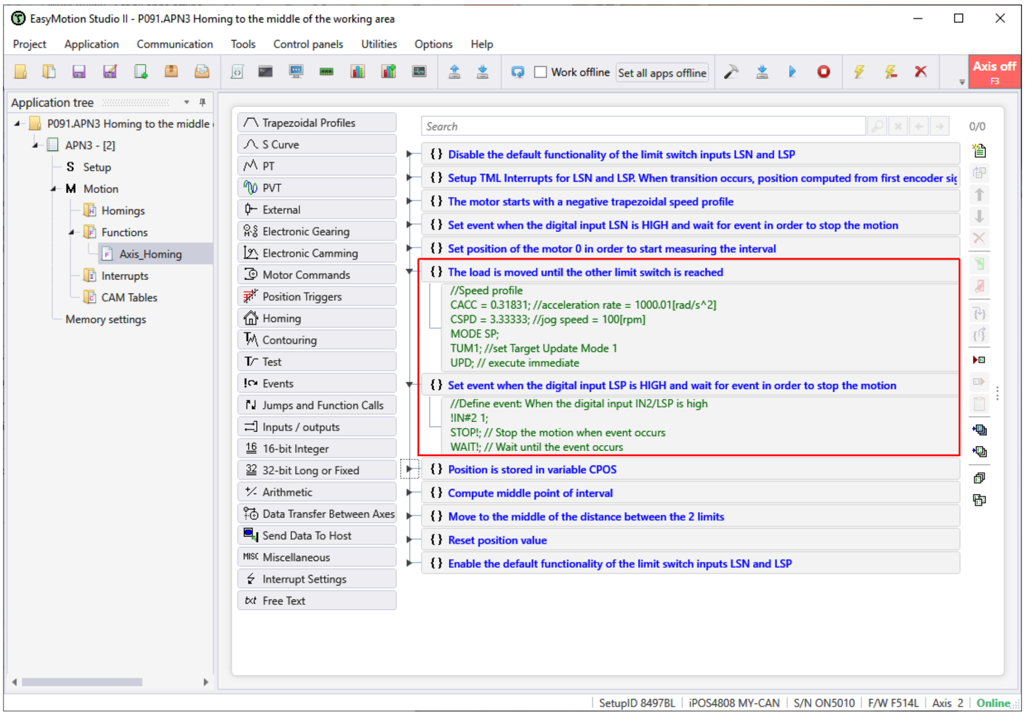
○ Set current limit ○ Value ○ Variable

○ Set quick stop deceleration rate ○ Value ○ Variable

OK Cancel Help

Figure 8 – Motor Commands wizard

The next step is to move positively until the positive limit switch is reached. This will be done using the “Trapezoidal Profile – Speed” and the “Events” wizard dialogues.



{ } Disable the default functionality of the limit switch inputs LSN and LSP

{ } Setup TML Interrupts for LSN and LSP. When transition occurs, position computed from first encoder sig

{ } The motor starts with a negative trapezoidal speed profile

{ } Set event when the digital input LSN is HIGH and wait for event in order to stop the motion

{ } Set position of the motor 0 in order to start measuring the interval

{ } The load is moved until the other limit switch is reached

```
//Speed profile
CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2]
CSPD = 3.33333; //jog speed = 100[rpm]
MODE SP;
TUM1; //set Target Update Mode 1
UPD; // execute immediate
```

{ } Set event when the digital input LSP is HIGH and wait for event in order to stop the motion

```
//Define event: When the digital input IN2/LSP is high
!IN#2 1;
STOP!; // Stop the motion when event occurs
WAIT!; // Wait until the event occurs
```

{ } Position is stored in variable CPOS

{ } Compute middle point of interval

{ } Move to the middle of the distance between the 2 limits

{ } Reset position value

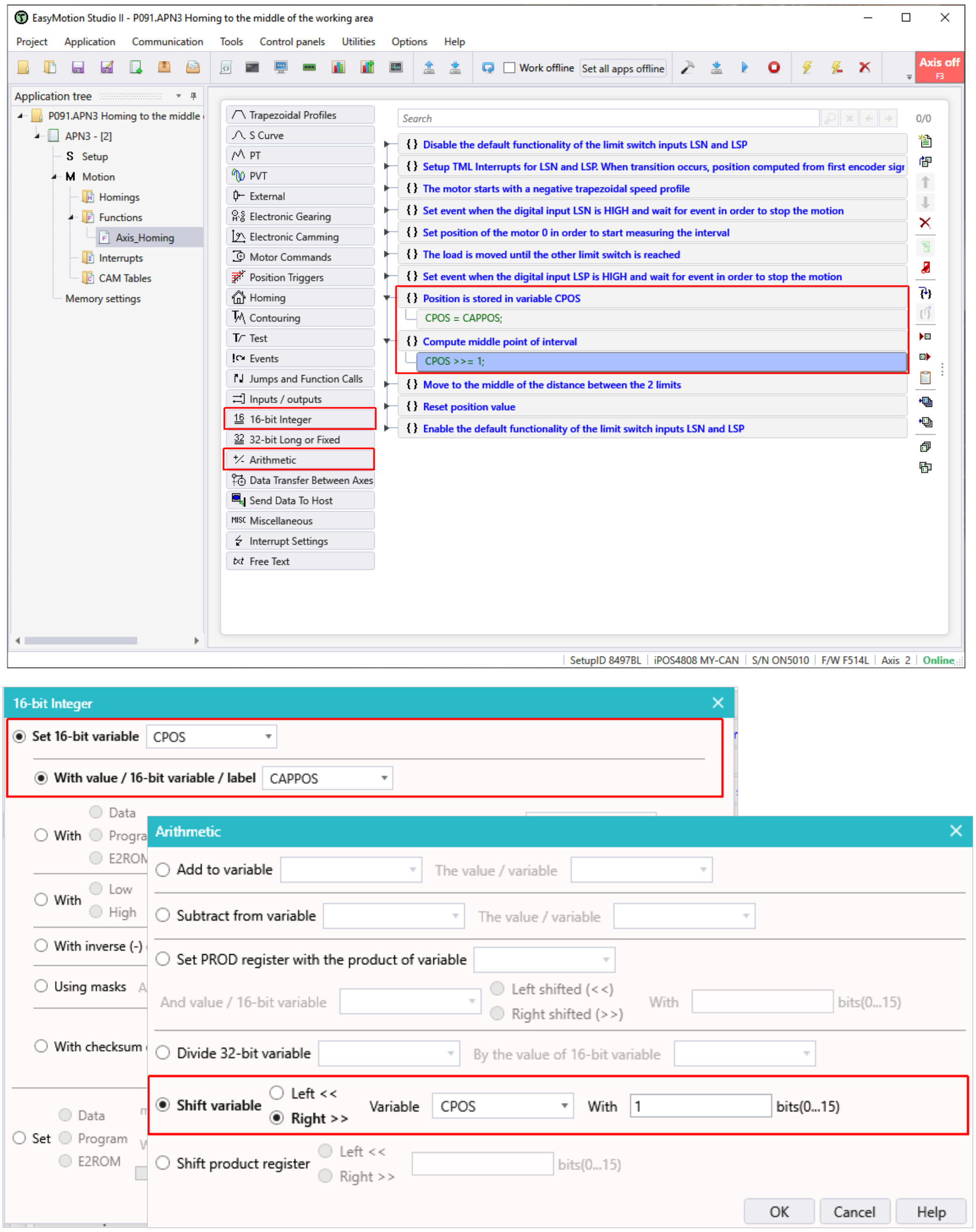
{ } Enable the default functionality of the limit switch inputs LSN and LSP

Figure 9 – Positive limit switch detection

Remark: By default, when a limit switch became active, if the associated interrupt is enabled, the drives automatically saves the encoder position indication in the “CAPPOS” variable, if the encoder is connected to Feedback #1 connector or in the “CAPPOS2” variable, if the encoder is connected to Feedback #2 connector.

In this application case, the “CAPPOS” variable will contain the exact distance (in encoder counts) between the 2 limit switches, because the position was set to 0 IU, when the negative limit was detected.

The middle point can be computed by dividing the value in “CAPPOS” to 2 or using a right shift operation with 1-bit, that is equivalent with a division to 2^1.



EasyMotion Studio II - P091.APN3 Homing to the middle of the working area

- { } Disable the default functionality of the limit switch inputs LSN and LSP
- { } Setup TML Interrupts for LSN and LSP. When transition occurs, position computed from first encoder sign
- { } The motor starts with a negative trapezoidal speed profile
- { } Set event when the digital input LSN is HIGH and wait for event in order to stop the motion
- { } Set position of the motor 0 in order to start measuring the interval
- { } The load is moved until the other limit switch is reached
- { } Set event when the digital input LSP is HIGH and wait for event in order to stop the motion
- { } Position is stored in variable CPOS
  - CPOS = CAPPOS;
- { } Compute middle point of interval
  - CPOS >>= 1;
- { } Move to the middle of the distance between the 2 limits
- { } Reset position value
- { } Enable the default functionality of the limit switch inputs LSN and LSP

16-bit Integer

● Set 16-bit variable: CPOS

● With value / 16-bit variable / label: CAPPOS

Arithmetic

● Shift variable ● Right >> Variable: CPOS With: 1 bits(0...15)

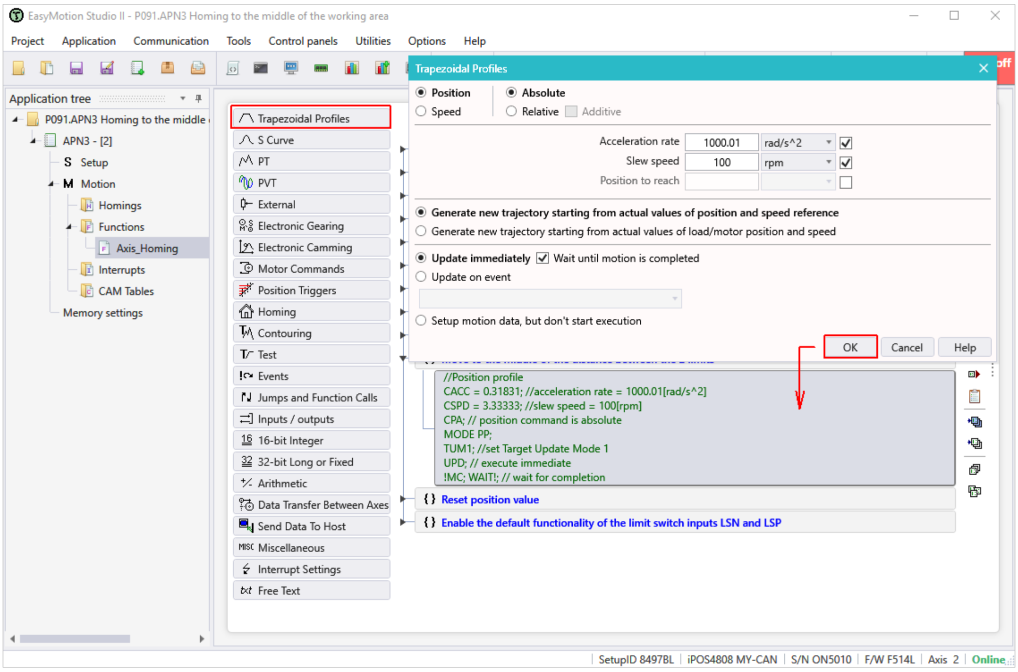
***Figure 10** – Middle point coordinate computing*

The implementation was done using the wizards for the “16-bit Integer” and “Arithmetic” operations.

The right shift operation was preferred here because it consumes fewer DSP resources than the division and there's no need to declare an extra 16-bit variable to be used only for this division.

After the shift operation is done, the “CPOS” variable will contain the absolute position of the middle point, so the load can be moved there, using an absolute position profile.

The absolute position profile can be inserted using the “Trapezoidal Profile” wizard dialogue.



Trapezoidal Profiles

● Position ○ Speed | ● Absolute ○ Relative □ Additive

Acceleration rate 1000.01 rad/s^2 ☑
Slew speed 100 rpm ☑
Position to reach □

● 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 ☑ Wait until motion is completed
○ Update on event
○ Setup motion data, but don't start execution

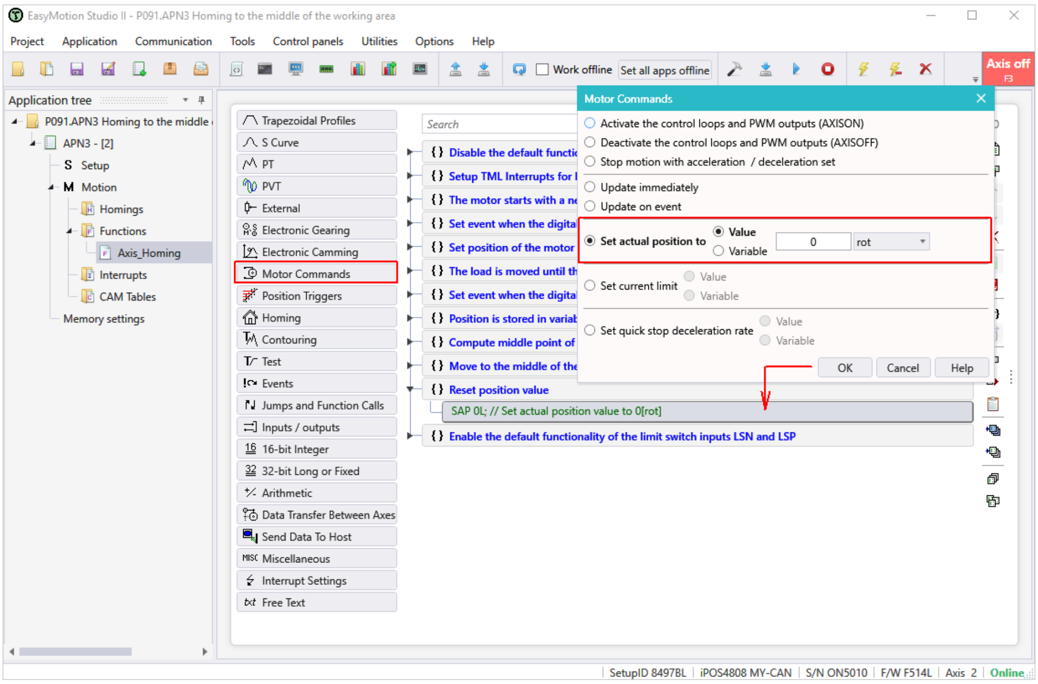
```
//Position profile
CACC = 0.31831; //acceleration rate = 1000.01[rad/s^2]
CSPD = 3.33333; //slew speed = 100[rpm]
CPA; // position command is absolute
MODE PP;
TUM1; //set Target Update Mode 1
UPD; // execute immediate
!MC; WAIT!; // wait for completion
```

{ } Reset position value
{ } Enable the default functionality of the limit switch inputs LSN and LSP

***Figure 11** – Absolute trapezoidal position profile*

<u>Remark</u> The “position to reach” field was disabled because the commanded position (the “CPOS” value) was computed in the previous step.

After the reaching the middle point (the home position), the actual position will be set to 0, using the “Motor commands” wizard.



{ } Disable the default functic
{ } Setup TML Interrupts for l
{ } The motor starts with a ne
{ } Set event when the digita
{ } Set position of the motor
{ } The load is moved until th
{ } Set event when the digita
{ } Position is stored in variab
{ } Compute middle point of
{ } Move to the middle of the
{ } Reset position value

```
SAP 0L; // Set actual position value to 0[rot]
```

{ } Enable the default functionality of the limit switch inputs LSN and LSP

Motor Commands

○ Activate the control loops and PWM outputs (AXISON)
○ Deactivate the control loops and PWM outputs (AXISOFF)
○ Stop motion with acceleration / deceleration set

○ Update immediately
○ Update on event

● Set actual position to ● Value ○ Variable 0 rot

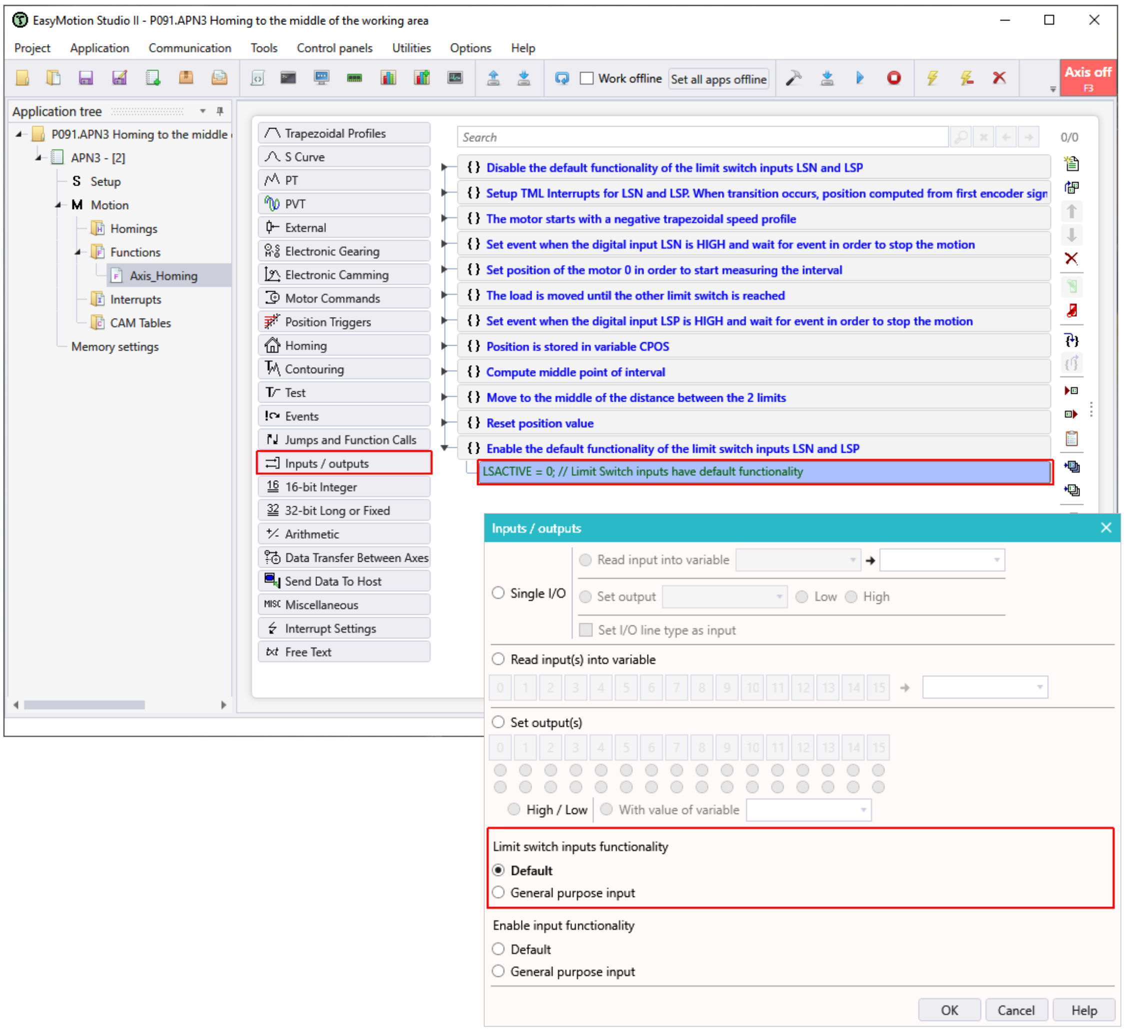
○ Set current limit ○ Value ○ Variable

○ Set quick stop deceleration rate ○ Value ○ Variable

***Figure 12** – Set the middle point as home position (0 IU)*

At this point, the load is placed in the middle of the working area, and the system can be operated using absolute position commands.

Before returning from the function, the limit switches default functionality will be restored using the option in the “Inputs/outputs” wizard dialogue.



{ } Disable the default functionality of the limit switch inputs LSN and LSP
{ } Setup TML Interrupts for LSN and LSP. When transition occurs, position computed from first encoder sign
{ } The motor starts with a negative trapezoidal speed profile
{ } Set event when the digital input LSN is HIGH and wait for event in order to stop the motion
{ } Set position of the motor 0 in order to start measuring the interval
{ } The load is moved until the other limit switch is reached
{ } Set event when the digital input LSP is HIGH and wait for event in order to stop the motion
{ } Position is stored in variable CPOS
{ } Compute middle point of interval
{ } Move to the middle of the distance between the 2 limits
{ } Reset position value
{ } Enable the default functionality of the limit switch inputs LSN and LSP

```
LSACTIVE = 0; // Limit Switch inputs have default functionality
```

Limit switch inputs functionality
● Default
○ General purpose input

Enable input functionality
○ Default
○ General purpose input

***Figure 13** – Limit switches default functionality restoring*

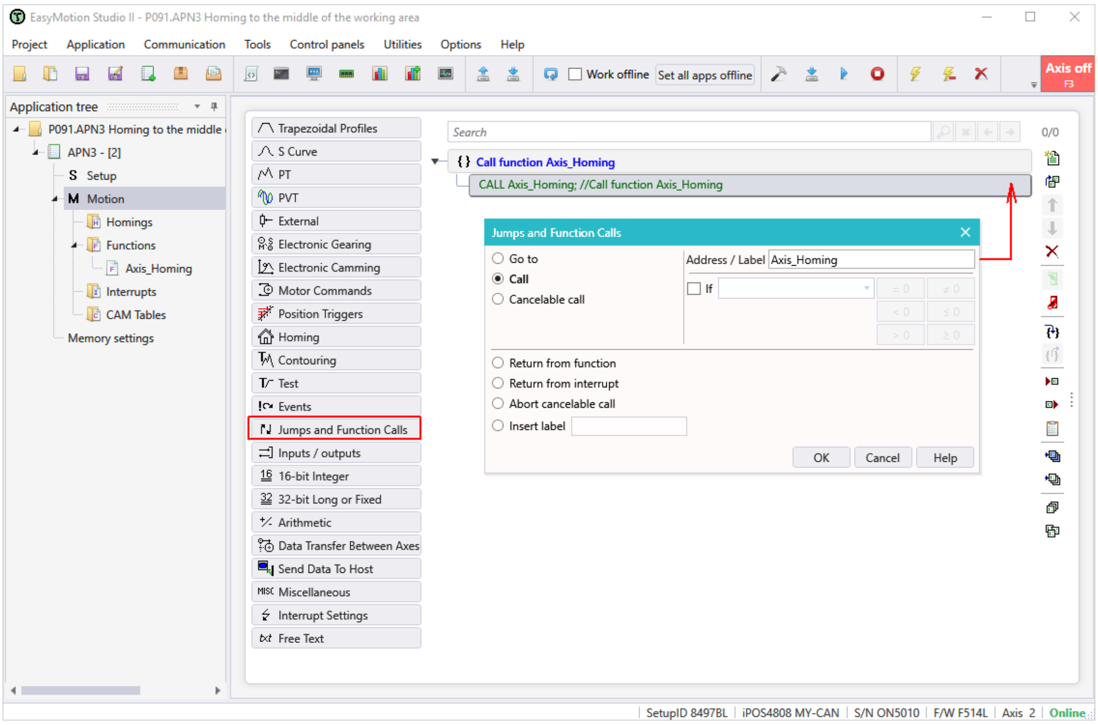
The main purpose of the “IN2/LSP” and “IN3/LSN” digital inputs is to allow connecting some limit switch sensors that will prevent the load from moving outside the safe are.

By default, when one of the limit switches became active, the drive stops the motor (using a quick stop profile), sets the correspondent bits in the MER error register and executes the code inside the correspondent limit switch interrupt routine (if it’s active on the respective transition).

A detailed description of the limit switches functionality can be found in the “Drive special inputs - Limit switches” application note.

3.2 Main motion program

In this application note case, the “Motion” section will contain only an instruction that will trigger the “Axis_Homing” function execution. This instruction can be generated using the “Call” option in the “Jumps and Function Calls” wizard dialogue.



EasyMotion Studio II - P091.APN3 Homing to the middle of the working area

```
{ } Call function Axis_Homing
    CALL Axis_Homing; //Call function Axis_Homing
```

Jumps and Function Calls

○ Go to
● Call
○ Cancelable call

Address / Label: Axis_Homing
□ If

○ Return from function
○ Return from interrupt
○ Abort cancelable call
○ Insert label

Figure 14 *- How to call a TML function*

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.

For more details about the functions call and usage, see the “Functions calling from a master” application note.

The “Axis_Homing” function can be called any time the homing procedure execution is required.