USER MANUAL UMAX060150
By: Erik Sasse
Date: Dec 18, 2018

High Performance Tilt Sensor

USER MANUAL

P/N: AX060150 Series

In Europe:
Axiomatic Technologies Oy
Hepolamminkatu 36 C 3
33720 TAMPERE - Finland
Tel. +358 3 3595 600
Fax. +358 3 3595 660
www.axiomatic.fi

In North America:
Axiomatic Technologies Corporation
5915 Wallace Street
Mississauga, ON Canada L4Z 1Z8
Tel. 1 905 602 9270
Fax. 1 905 602 9279
www.axiomatic.com

Contents

1 Introduction

The Tilt Sensor Controller, as the name implies, monitors tilt angles and turns ON or OFF outputs based on the status of these angles.
The controller has two ON/OFF outputs A and B with their respective indicator LED (Light Emitting Diode), which can be configured based on specific requirements of an application.

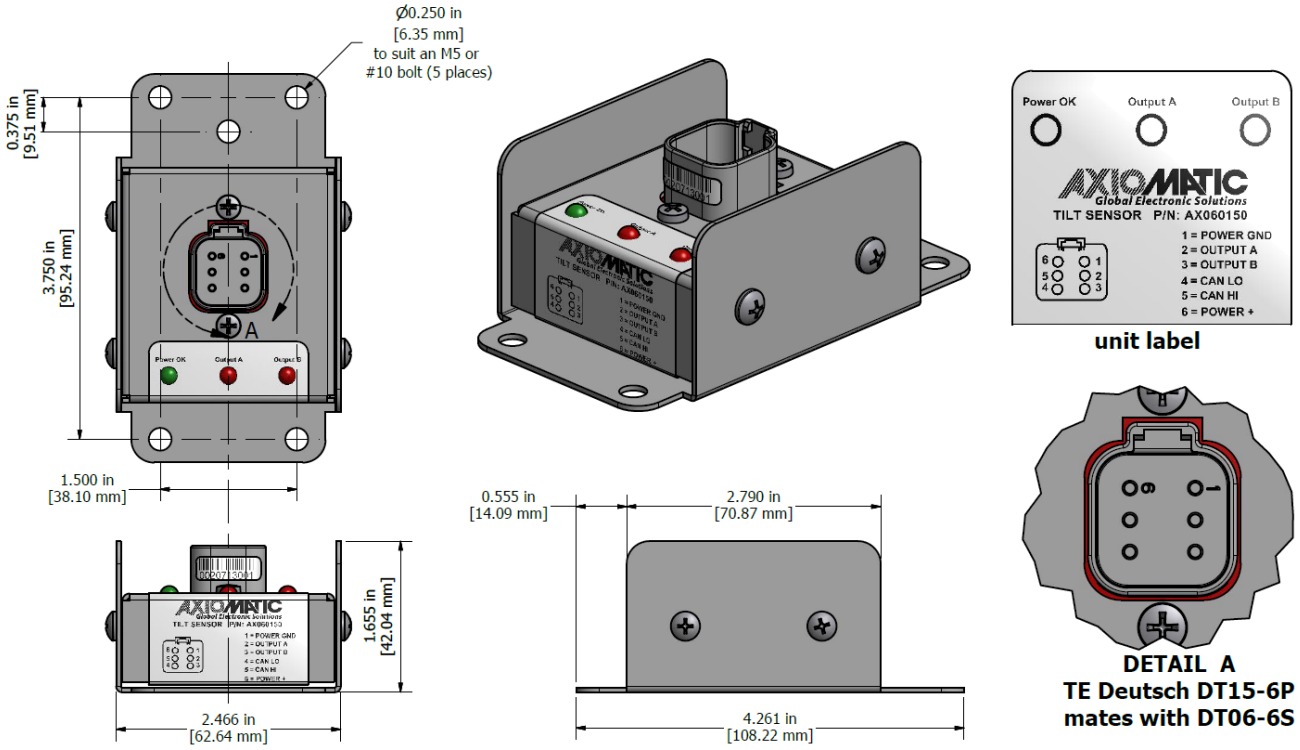
The parameters of the controller can be configured through CANopen and more directly through any CANopen Object View access software available from any CAN to USB converter vendors.
This user manual has some examples that refer to the ESD CAN/USB converter, but any other converter from a different manufacturer can be used.

2 Pinout

Pin Number	Description
1	Power – (Battery -)
2	Output A
3	Output B
4	CAN LO
5	CAN HI
6	Power + (Battery +)

Mating Connector Deutsch P/N: DT06-6S with Wedge lock W6S

3 Assembly Dimensions



4 CANopen Object Table

The object table is the complete list of the variables that can be access and modified to configure the controller for a specific application. It is important to have an idea of this table before explaining the modes of operation and how these variables affect the functionality of the controller.
This section in this manual is more for reference than actual explanation. Many of these variables and their functions will be explained in detail in the following sections.

index$_h$	*sub*	*obj.*	*variable*	*type*	*bytes*	*access*	*default value (profile)*	*Static value (flash)*	*Notes*
1000		VAR	co_device_type	U32	4	RO	0xXXXX019A	0x0002019A	
1001		VAR	co_error_register	U8	1	RO		0	
1002		VAR	manuf_status_register	U32	4	RO		0	
1003	0	ARRAY	pre_defined_error_field	U32	4	RW		0x0000	
	1			U32	4	RW		0x0000	
	2			U32	4	RW		0x0000	
	3			U32	4	RW		0x0000	
1009	0	ARRAY	co_manufacturer_hardware_version	U8	1	RO		3	
	1			U8	1	RO		'1'	
	2			U8	1	RO		'.'	
	3			U8	1	RO		'0'	
100A	0	ARRAY	co_manufacturer_software_version	U8	1	RO		3	
	1			U8	1	RO		'0'	
	2			U8	1	RO		'.'	
	3			U8	1	RO		'0'	
100C		VAR	guard_time	U16	2	RW			
100D		VAR	life_time_factor	U8	1	RW			
1010	0	ARRAY	co_store_params	U32	4	RO	1-127	6	
	1			U32	4	RW		1	All settings (0x 65 76 61 73 saves the parameters)
	2			U32	4	RW		1	Communication 1000-1FFF
	3			U32	4	RW		1	Application 6000-9FFF
	4			U32	4	RW		1	Manufacturer 2000-5FFF
1011	0	ARRAY	co_restore_params	U32	4	RO	1-127	6	
	1			U32	4	RW		1	All settings default (0x 64 61 6f 6c loads default parameters) Use object 1010 to save the parameters to eeprom.
	2			U32	4	RW		1	Communication 1000-1FFF

index$_h$	*sub*	*obj.*	*variable*	*type*	*bytes*	*access*	*default value (profile)*	*Static value (flash)*	*Notes*
	3			U32	4	RW		1	Application 6000-9FFF
	4			U32	4	RW		1	Manufacturer 2000-5FFF
1016	0	ARRAY	co_consumer_heartbeat_time	U32	4	RO	1-127	1	
	1			U32	4	RW	0	0	
	2			U32	4	RW		0	
	3			U32	4	RW		0	
	4			U32	4	RW		0	
1017		VAR	co_producer_heartbeat_time	U16	2	RW	0	0	
1018	0	RECORD	co_identity	U32	4	RO	1-4	4	
	1			U32	4	RO		0x00000055	Vendor ID
	2			U32	4	RO		0x00000000	Product code
	3			U32	4	RO		0x00010000	Revision number
	4			U32	4	RO		0x00000001	Serial number
1020	0	ARRAY	verify_configuration	U32	4	RO		2	
	1			U32	4	RO		0x12172018	Day/Month/Year
	2			U32	4	RO		0x00001801	Time, 24hour clock
1029	0	ARRAY	error_behaviour	U8	1	RW		2	
	1							1	Busoff, Lifeguarding Event, Heartbeat Event or Sync Error Occurred
	2							2	Accelerometer Sensor Error State
1400	0	RECORD	rpdo_comm_par[0]			RO		5	
	1				4	RW		0xC0000250	
	2				1	RO		0xFF	
	3				2	RW		0	
	4				1	RW		0	
	5				2	RW		0	
1401	0	RECORD	rpdo_comm_par[1]			RO		5	
	1				4	RW		0xC0000350	
	2				1	RO		0xFF	
	3				2	RW		0	
	4				1	RW		0	
	5				2	RW		0	
1402	0	RECORD	rpdo_comm_par[2]			RO		5	
	1				4	RW		0xC0000450	
	2				1	RO		0xFF	

index$_h$	*sub*	*obj.*	*variable*	*type*	*bytes*	*access*	*default value (profile)*	*Static value (flash)*	*Notes*
	3				2	RW		0	
	4				1	RW		0	
	5				2	RW		0	
1403	0	RECORD	rpdo_comm_par[3]			RO		5	
	1				4	RW		0xC0000550	
	2				1	RO		0xFF	
	3				2	RW		0	
	4				1	RW		0	
	5				2	RW		0	
1600	0	RECORD	rpdo_map_par[0]			RO		4	
	1				4	RO		0x25000110	
	2				4	RO		0x25000210	
	3				4	RO		0x25000310	
	4				4	RO		0	
1601	0	RECORD	rpdo_map_par[1]			RO		4	
	1				4	RO		0	
	2				4	RO		0	
	3				4	RO		0	
	4				4	RO		0	
1602	0	RECORD	rpdo_map_par[2]			RO		4	
	1				4	RO		0	
	2				4	RO		0	
	3				4	RO		0	
	4				4	RO		0	
1603	0	RECORD	rpdo_map_par[3]			RO		4	
	1				4	RO		0	
	2				4	RO		0	
	3				4	RO		0	
	4				4	RO		0	
1800	0	RECORD	co_tpdo_comm_par[0]	U8	1	RO	2-5	5	
	1			U32	4	RW	0x180+nodeID	0xC0000000	PDO:n cob-id
	2			U8	1	RO		0xFE	Transmit type
	3			U16	2	RW		0	Inhibit time
	4			U8	1	RW		0	Reserved
	5			U16	2	RW		0	Event timer
1801	0	RECORD	co_tpdo_comm_par[1]	U8	1	RO	2-5	5	
	1			U32	4	RW	0x280+nodeID	0xC0000000	PDO:n cob-id

index$_h$	*sub*	*obj.*	*variable*	*type*	*bytes*	*access*	*default value (profile)*	*Static value (flash)*	*Notes*
	2			U8	1	RO		0xFE	Transmit type
	3			U16	2	RW		0	Inhibit time
	4			U8	1	RW		0	Reserved
	5			U16	2	RW		0	Event timer
1802	0	RECORD	co_tpdo_comm_par[2]	U8	1	RO	2-5	5	
	1			U32	4	RW	0x380+nodeID	0xC0000000	PDO:n cob-id
	2			U8	1	RO		0xFE	Transmit type
	3			U16	2	RW		0	Inhibit time
	4			U8	1	RW		0	Reserved
	5			U16	2	RW		0	Event timer
1803	0	RECORD	co_tpdo_comm_par[3]	U8	1	RO	2-5	5	
	1			U32	4	RW	0x480+nodeID	0xC0000000	PDO:n cob-id
	2			U8	1	RO		0xFE	Transmit type
	3			U16	2	RW		0	Inhibit time
	4			U8	1	RW		0	Reserved
	5			U16	2	RW		0	Event timer
1A00	0	RECORD	co_tpdo_map_par[0]	U8	1	RO		4	
	1			U32	4	RO		0x20010010	
	2			U32	4	RO		0x20020010	
	3			U32	4	RO		0x20030010	
	4			U32	4	RO		0	
1A01	0	RECORD	co_tpdo_map_par[1]	U8	1	RO		4	
	1			U32	4	RO		0x20010010	
	2			U32	4	RO		0x20020010	
	3			U32	4	RO		0x20030010	
	4			U32	4	RO		0	
1A02	0	RECORD	co_tpdo_map_par[2]	U8	1	RO		4	
	1			U32	4	RO		0x20010010	
	2			U32	4	RO		0x20020010	
	3			U32	4	RO		0x20030010	
	4			U32	4	RO		0	
1A03	0	RECORD	co_tpdo_map_par[3]	U8	1	RO		4	
	1			U32	4	RO		0x20010010	
	2			U32	4	RO		0x20020010	
	3			U32	4	RO		0x20030010	
	4			U32	4	RO		0	
2000		VAR	co_temperature	I8	1	RO		25	

index$_h$	*sub*	*obj.*	*variable*	*type*	*bytes*	*access*	*default value (profile)*	*Static value (flash)*	*Notes*
2001		VAR	co_angle_X	I16	2	RO		0	Includes the mounting offset object 5050. value range -900 to 900 (-90.0 – 90.0) degrees
2002		VAR	co_angle_Y	I16	2	RO		0	Includes the mounting offset object 5051.value range -900 to 900 (-90.0 – 90.0) degrees
2003		VAR	co_angle_360	I16	2	RO		0	Includes the mounting offset object 5052. value range 0 to 3599 (0 – 359.9) degrees
2004		VAR	co_Output_A_State	U16	2	RO		0	The current drive state of output A
2005		VAR	co_Output_B_State	U16	2	RO		0	The current drive state of output B
2006		VAR	co_angle_X_without_offset	I16	2	RO		0	Angle without mounting offset 5050 .value range -900 to 900 (-90.0 – 90.0) degrees
2007		VAR	co_angle_Y_without_offset	I16	2	RO		0	Angle without mounting offset 5051.value range -900 to 900 (-90.0 – 90.0) degrees
2008		VAR	co_angle_360_without_offset	I16	2	RO		0	Angle without mounting offset 5052. value range 0 to 3599 (0 – 359.9) degrees
2020		VAR	co_10ms	U16	2	RO		0	
2021		VAR	co_print_cmd	U32	4	RW		0	
2022		VAR	co_bootup_message_enabled	U16	2	RW		0	0 == message disabled, 1 -> message enabled
2023		VAR	co_calib_temp	I16	2	ROP		25	Protected variable
2030		VAR	can_slew_rate	U8	1	RW		0	
2032		VAR	sensor_input_filter_enable	U8	1	RW		1	
2033		VAR	input_filter_cutoff_frequency	U8	1	RW		1	
2040	0	ARRAY	axis_data_decimal_digits_fv	U8	1	RO		3	
	1			U16	2	RW		1000	
	2			U16	2	RW		1000	
	3			U16	2	RW		1000	
2041	0	ARRAY	axis_data_inverse_fv_data	U8	1	RO		3	
	1			U8	1	RW		0	
	2			U8	1	RW		0	
	3			U8	1	RW		0	
2050	0	ARRAY	axis_data_offset_fv	U8	4	RO		3	
	1			R32	4	RW		0	
	2			R32	4	RW		0	
	3			R32	4	RW		0	
2060	0	ARRAY	angle_data_decimal_digits_fv	U16	2	RO		3	
	1			U16	2	RW		1000	

index$_h$	*sub*	*obj.*	*variable*	*type*	*bytes*	*access*	*default value (profile)*	*Static value (flash)*	*Notes*
	2			U16	2	RW		1000	
	3			U16	2	RW		1000	
2061	0	ARRAY	angle_data_inverse_fv_data	U8	1	RO		3	
	1			U8	1	RW		0	
	2			U8	1	RW		0	
	3			U8	1	RW		0	
2062	0	ARRAY	angle_data_range	U8	1	RO		3	
	1			U8	1	RO		1	
	2			U8	1	RO		1	
	3			U8	1	RO		1	
2070	0	ARRAY	angle_data_offset_fv	U8	1	RO		3	
	1			R32	4	RW		0	
	2			R32	4	RW		0	
	3			R32	4	RW		0	
2100		VAR	cpAngleType	U8	1	RW		1	
2200		VAR	f_cpInitPitchAngle	R32	4	RW		0.0	
2210		VAR	f_cpInitRollAngle	R32	4	RW		0.0	
2220		VAR	b_cpInstalllMount	U8	1	RW		0	
2300		VAR	auto_null_command	U8	1	RW		0	
2400		VAR	bs_pitch_angle	R32	4	RO		0	
2401		VAR	bs_roll_angle	R32	4	RO		0	
2500	0	ARRAY	rpdo_received_pv	U8	1	RO		3	
	1			U16	2	RWM		0	
	2			U16	2	RWM		0	
	3			U16	2	RWM		0	
2501	0	ARRAY		U8	1	RO		3	
	1			U16	2	RW		1000	
	2			U16	2	RW		1000	
	3			U16	2	RW		1000	
2502		ARRAY	rpdo_received_offsef_pv	U8	1	RO		3	
				R32	4	RW		0	
				R32	4	RW		0	
				R32	4	RW		0	
4000	0	ARRAY	accelerometer_data_output	U16	2	RO		3	
	1			I16	2	RO		0	
	2			I16	2	RO		0	

index$_h$	*sub*	*obj.*	*variable*	*type*	*bytes*	*access*	*default value (profile)*	*Static value (flash)*	*Notes*
	3			I16	2	RO		0	
4010		VAR		U16	2	RO		0	
4020		VAR		U8	1	RO		0	
4030		VAR		I16	2	RO		0	
4040		VAR		I16	2	RO		0	
4050		VAR		I16	2	RO		0	
4060		VAR		I16	2	RO		0	
4070		VAR		R32	4	RW		0	
4080		VAR		U32	2	RW		0	
5050		VAR	co_mounting_offset_X	I16	2	RWP			X -axis mounting offset in degrees. Value range -800 to 800 (-80.0 ... 80.0) degrees. NOTE: Modules actual measuring range is always +-90 degrees despite this offset. So with 10 degrees offset (100) the module is able to measure 80 degrees to the minus direction and 100 degrees to plus direction when co_angle_X object is 0.
5051		VAR	co_mounting_offset_Y	I16	2	RWP			Y axis mounting offset in degrees. Value range -800 to 800 (-80.0 ... 80.0) degrees. NOTE: Modules actual measuring range is always +-90 degrees despite this offset. So with 10 degrees offset (100) the module is able to measure 80 degrees to the minus direction and 100 degrees to plus direction when co_angle_Y object is 0
5052		VAR	co_mounting_offset_360	I16	2	RWP			Mounting offset in degrees in 360 degrees mode Value range 0 to 3599 (0 ... 359.9). Doesn't affect to the modules actual measuring range
5060		VAR	co_set_mounting_offsets	U8	1	RWP		0	Writing 0x12 sets the real angles of X and Y axis (objects 2006 and 2007) as offsets to the objects 5050 and 5051. Writing 0x36 sets the real angle (object 2008) as 360 degrees mode offset to the object 5052.
6000		VAR	co_operation_mode	U16	2	RW		0x36	Tells are we in 1 axis 360 degrees mode(0x36) or in dual axis xy mode (0x12)
6002		VAR	co_output_A_boot_delay	U16	2	RWP		0	Protected variable (in 10 milliseconds, 0,10,20 ... 5000ms, 500 -> 5000ms))
6003		VAR	co_output_A_boot_delay_value	U16	2	RWP		0	Protected variable (active while booting ==1 or deactive == 0)
6004		VAR	co_output_B_boot_delay	U16	2	RWP		0	Protected variable

index$_h$	*sub*	*obj.*	*variable*	*type*	*bytes*	*access*	*default value (profile)*	*Static value (flash)*	*Notes*
6005		VAR	co_output_B_boot_delay_value	U16	2	RWP		0	Protected variable
6006		VAR	co_output_A_inverse_mode_on	U16	2	RWP		0	Protected variable (0 == not active, 0x23 == output is inverted)
6007		VAR	co_output_B_inverse_mode_on	U16	2	RWP		0	Protected variable (0 == not active, 0x23 == output is inverted)
6008		VAR	co_output_A_control_logic	U16	2	RWP		0x0F	Protected variable, Used only in XY mode, 0x0F -> output_A is on if both inclinometers agree ==LOGIC_AND, 0x0A -> output_A is on if one inclinometer states that == LOGIC_OR
6009		VAR	co_output_B_control_logic	U16	2	RWP		0x0F	Protected variable, Used only in XY mode, 0x0F -> output_B is on if both inclinometers agree ==LOGIC_AND, 0x0A -> output_B is on if one inclinometer states that == LOGIC_OR
600A		VAR	co_number_of_sectors_x	U16	2	RWP		4	Protected variable (2-4)
600B		VAR	co_number_of_sectors_y	U16	2	RWP		4	Protected variable. Used in XY mode.
6010	0	RECORD	sectors_A0_x	U16	2	ROP		4	max_subindex
	1			U16	2	RWP		0	filter_delay (in 10 miliseconds, 0,10,20 .. 5000ms, 500 -> 5000ms)
	2			U16	2	RWP		0	driver_value (1 or 0)
	3			I16	2	RWP		0	trigger_angle (80 = 8.0 degrees) value range 0...3599 in 360- mode. -900 ... 900 in xy mode. trigger_angle_A(n) < trigger_angleA(n+1)
	4			I16	2	RWP		0	Hysteresis (8 = 0.8 degrees)
6020	0	RECORD	sectors_A1_x	U16	2	ROP		4	max_subindex
	1			U16	2	RWP		0	filter_delay
	2			U16	2	RWP		0	driver_value
	3			I16	2	RWP		0	trigger_angle
	4			I16	2	RWP		0	hysteresis
6030	0	RECORD	sectors_A2_x	U16	2	ROP		4	max_subindex
	1			U16	2	RWP		0	filter_delay
	2			U16	2	RWP		0	driver_value
	3			I16	2	RWP		0	trigger_angle
	4			I16	2	RWP		0	hysteresis
6040	0	RECORD	sectors_A3_x	U16	2	ROP		4	max_subindex
	1			U16	2	RWP		0	filter_delay
	2			U16	2	RWP		0	driver_value
	3			I16	2	RWP		0	trigger_angle
	4			I16	2	RWP		0	hysteresis
6050	0	RECORD	sectors_B0_x	U16	2	ROP		4	max_subindex
	1			U16	2	RWP		0	filter_delay
	2			U16	2	RWP		0	driver_value

index$_h$	*sub*	*obj.*	*variable*	*type*	*bytes*	*access*	*default value (profile)*	*Static value (flash)*	*Notes*
	3			I16	2	RWP		0	trigger_angle
	4			I16	2	RWP		0	hysteresis
6060	0	RECORD	sectors_B1_x	U16	2	ROP		4	max_subindex
	1			U16	2	RWP		0	filter_delay
	2			U16	2	RWP		0	driver_value
	3			I16	2	RWP		0	trigger_angle
	4			I16	2	RWP		0	hysteresis
6070	0	RECORD	sectors_B2_x	U16	2	ROP		4	max_subindex
	1			U16	2	RWP		0	filter_delay
	2			U16	2	RWP		0	driver_value
	3			I16	2	RWP		0	trigger_angle
	4			I16	2	RWP		0	hysteresis
6080	0	RECORD	sectors_B3_x	U16	2	ROP		4	max_subindex
	1			U16	2	RWP		0	filter_delay
	2			U16	2	RWP		0	driver_value
	3			I16	2	RWP		0	trigger_angle
	4			I16	2	RWP		0	hysteresis
6110	0	RECORD	sectors_A0_y	U16	2	ROP		4	max_subindex
	1			U16	2	RWP		0	filter_delay
	2			U16	2	RWP		0	driver_value
	3			I16	2	RWP		0	trigger_angle. Ignored in 360-mode. value range -900 ... 900
	4			I16	2	RWP		0	hysteresis
6120	0	RECORD	sectors_A1_y	U16	2	ROP		4	max_subindex
	1			U16	2	RWP		0	filter_delay
	2			U16	2	RWP		0	driver_value
	3			I16	2	RWP		0	trigger_angle
	4			I16	2	RWP		0	hysteresis
6130	0	RECORD	sectors_A2_y	U16	2	ROP		4	max_subindex
	2			U16	2	RWP		0	filter_delay
	3			U16	2	RWP		0	driver_value
	3			I16	2	RWP		0	trigger_angle
	4			I16	2	RWP		0	hysteresis
6140	0	RECORD	sectors_A3_y	U16	2	ROP		4	max_subindex
	2			U16	2	RWP		0	filter_delay
	3			U16	2	RWP		0	driver_value
	3			I16	2	RWP		0	trigger_angle
	4			I16	2	RWP		0	hysteresis
6150	0	RECORD	sectors_B0_y	U16	2	ROP		4	max_subindex
	2			U16	2	RWP		0	filter_delay
	3			U16	2	RWP		0	driver_value
	3			I16	2	RWP		0	trigger_angle
	4			I16	2	RWP		0	hysteresis

index$_h$	*sub*	*obj.*	*variable*	*type*	*bytes*	*access*	*default value (profile)*	*Static value (flash)*	*Notes*
6160	0	RECORD	sectors_B1_y	U16	2	ROP		4	max_subindex
	2			U16	2	RWP		0	filter_delay
	3			U16	2	RWP		0	driver_value
	3			I16	2	RWP		0	trigger_angle
	4			I16	2	RWP		0	hysteresis
6170	0	RECORD	sectors_B2_y	U16	2	ROP		4	max_subindex
	2			U16	2	RWP		0	filter_delay
	3			U16	2	RWP		0	driver_value
	3			I16	2	RWP		0	trigger_angle
	4			I16	2	RWP		0	hysteresis
6180	0	RECORD	sectors_B3_y	U16	2	ROP		4	max_subindex
	2			U16	2	RWP		0	filter_delay
	3			U16	2	RWP		0	driver_value
	3			I16	2	RWP		0	trigger_angle
	4			I16	2	RWP		0	hysteresis

17 Dec, 2018 Object Table

5 Accessing Table of Objects using an CANopen Object Viewer software

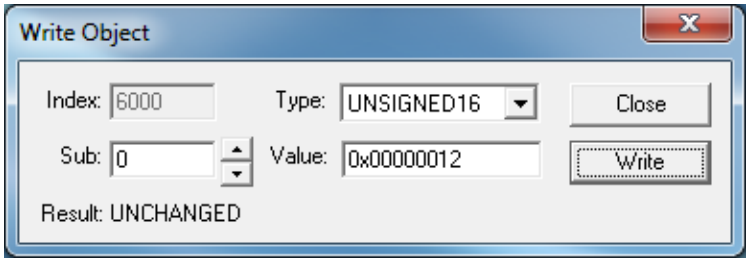
The following example documents how to view, access and modify the objects in the object table by using third party software from ESD.

Note: Any software with CANopen object view capabilities can be used.

a. Open the program **cobview.exe** located in the CAN esd bin folder and set the following parameters:
   Net: 0
   Baud: 125 kBits/s
   Module No:80 (the nodeID is 0x50 = 80 dec, that is the module number)
b. Click on Find Modules and notice the Hex value 0002019A, which is the static value of the object 1000 co_device_type (See object table for details)

Finding Module (Module No = 80 = NodeID)

c. To read an object in the object table, just enter the index number and click on Read. Once the object and its sub-index are displayed select the index that you want to change and click on Write.
   The Write Object window appears. Enter the new hex value and click on Write.

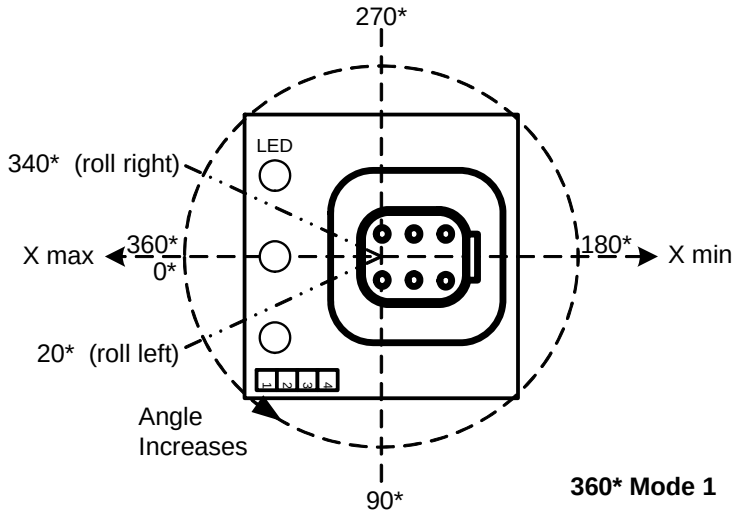


Click on Close to close the window.

6 Modes of Operation

The controller has two modes of operation. Mode 1 - 360° degrees and Mode 2 - XY

6.1 Mode 1 - 360° degrees



Mode 1 - 360 °Angle Displacement

The controller is to be mounted on its side facing the LEDs and connector as per figure above. The mid LED can be reference as 0° degrees.
Rolling the controller to the left increases the angle from 0° all the way up to 360°. Rolling the controller to the right decreases the angle from 360° all the way down to 0° (full rotation).
The Mode 1 - 360° uses only the X-axis to monitor the angle displacement but it has both Output A and Output B available to be used.
For example, if the measured angle is greater than 2°, Output A can be programmed to be turn OFF or ON and if the angle is greater than let's say 5°, Output B can be programmed to be turn OFF or ON.

Mode 1 has 4 sectors for each Output A and Output B that can be programmed giving it a total of 8 sectors. Not all applications need to have Output A and B actives. If that is the case, just activate Output A as your only output.

Before operating in this mode, the following controller's object variables must be configured as follow:

6.1.1 Setting 360° Operation Mode

The operation mode is controlled by the object index 6000 (co_operation_mode).
To set the operation mode to 360° set index 6000 (co_operation_mode) sub-index 0 = 0x00000036.
Example using ESD

a. Access the Index 6000 (co_operation_mode) and click on Read
b. Verify that the Hex value is 0x00000036 (54 dec).
c. If not, click on Write, the Write Object window opens. Enter the correct value and click on Write to change the object value. Click on Close to finish.

6.1.2 Setting up the 360° number of sectors

For Mode 1 only the sectors in the X-axis need to be setup. The number of sectors in X-axis is controlled by object index 600A (co_number_of_sectors_x).
To set the number of sectors between 1 and 4 for the X-axis, set index 600A (co_number_of_sectors_x) Sub-index 0 = number of sectors required.

WARNING! It is a MUST to have 2 or more sectors for proper operation

Example using ESD

a. Set the Index to 600A (co_number_of_sectors_x) and click on Write
b. In Sub:0, enter the value 0x00000002 to set the number of sectors to two (2). Make sure the type is UNSIGNED16. Click on Write to send the new value. Click on Close to finish

6.1.3 Setting up Mode 1 360° sectors

The sectors are configured by the following variables:

Index	Sub Index	Variable Name	Notes
Output A sectors			
6010	0	sectors_A0_x	max_subindex
	1		filter_delay (in 10 miliseconds, 0,10,20 .. 5000ms, 500 -> 5000ms)
	2		driver_value (1 or 0)
	3		trigger_angle (80 = 8.0 degrees) value range 0...3599 trigger_angle_A(n) < trigger_angleA(n+1)
	4		Hysteresis (8 = 0.8 degrees)
6020	0	sectors_A1_x	max_subindex
	1		filter_delay
	2		driver_value
	3		trigger_angle
	4		hysteresis
6030		sectors_A2_x	max_subindex
			filter_delay
			driver_value
			trigger_angle
			hysteresis
6040		sectors_A3_x	max_subindex
			filter_delay
			driver_value
			trigger_angle
			hysteresis
Output B sectors			
6050	0	sectors_B0_x	max_subindex
	1		filter_delay (in 10 miliseconds, 0,10,20 .. 5000ms, 500 -> 5000ms)
	2		driver_value (1 or 0)

	3		*t*rigger_angle (80 = 8.0 degrees) value range 0...3599 trigger_angle_A(n) < trigger_angleA(n+1)
	4		Hysteresis (8 = 0.8 degrees)
6060	0	sectors_B1_x	max_subindex
	1		filter_delay
	2		driver_value
	3		trigger_angle
	4		hysteresis
6070		sectors_B2_x	max_subindex
			filter_delay
			driver_value
			trigger_angle
			hysteresis
6080		sectors_B3_x	max_subindex
			filter_delay
			driver_value
			trigger_angle
			hysteresis

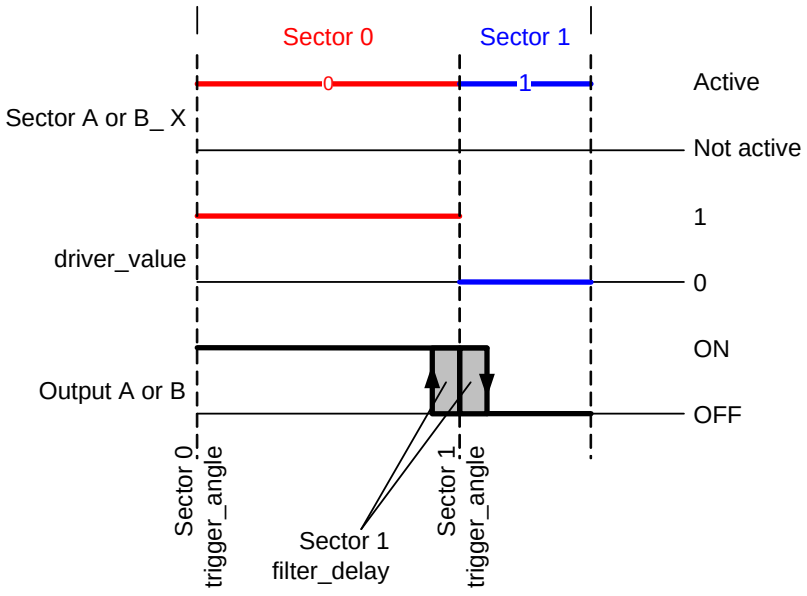
Note: A0_x, A1_x, A2_x and A3_x are Output A while B0_x, B1_x, B2_x, B3_x are Output B.

6.1.4 Filter Delay

The Filter Delay is the delay in milliseconds added to the selected sector before the driver_value is activated when the trigger_angle + hysteresis is reached.

Note: 100 decimal = 1 second or 1 decimal = 10 milliseconds.

This delay can be used in conjunction with the hysteresis to make sure the controller does not activate an output due to sudden movements, for example while driving machinery on unbalance terrain or acceleration.



Where Sector0 Trigger_Angle < Sector1 Trigger_Angle
And No hysteresis has been added

Filter Delay diagram

To set the filter_delay of any sector, select the Index of the sector required and write to sub-index 1 the value of the delay in hexadecimal format. So if a 500 ms delay is needed, convert 500 to hex = 01F4 and write that number to the sub-index 1.

WARNING! It is recommended NOT to have a delay greater than 500 ms. a delay greater than that may be too late to control heavy machinery.

6.1.5 Hysteresis

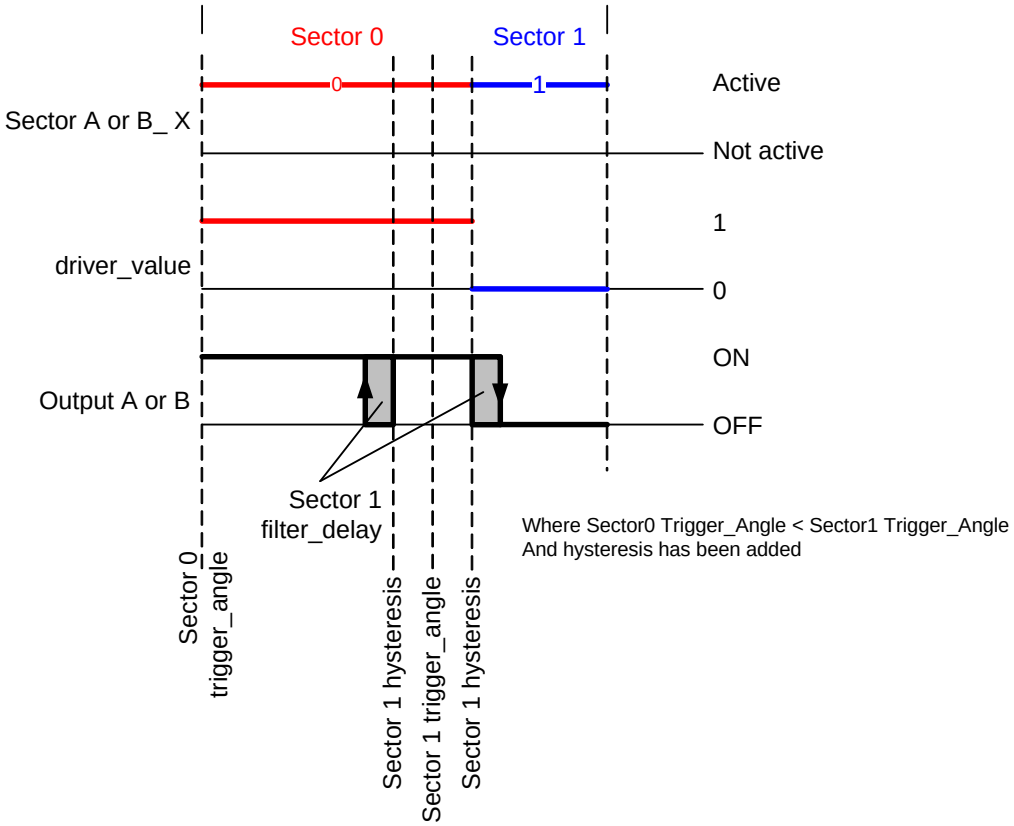
Hysteresis has almost the same functionality as the filter_delay. Instead, this is not a time variable, but an angle variable.
Basically:

driver_value = TRUE (ON or OFF) if measured angle > (trigger_angle + hysteresis)

Or

driver_value = TRUE (ON or OFF) if measured angle < (trigger_angle - hysteresis)

See the following figure for more details.

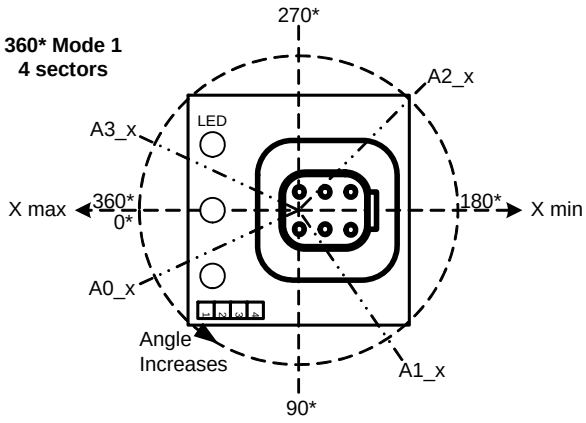


Hysteresis diagram

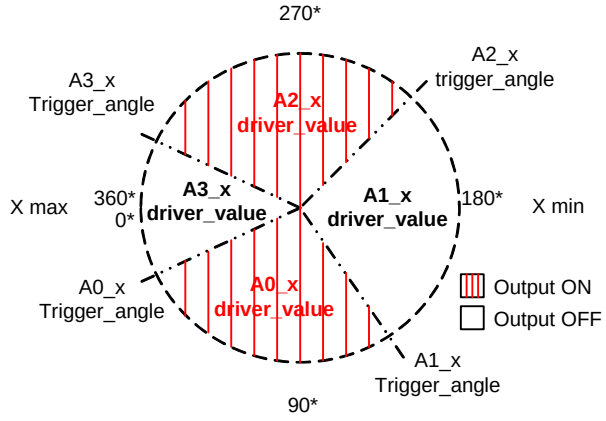
IMPORTANT NOTE: The sector hysteresis value cannot be greater than the sector trigger_angle. Failure to do so may cause the controller to misbehave.

6.1.6 Mode 1 - 360° Driver Value and Trigger Angle

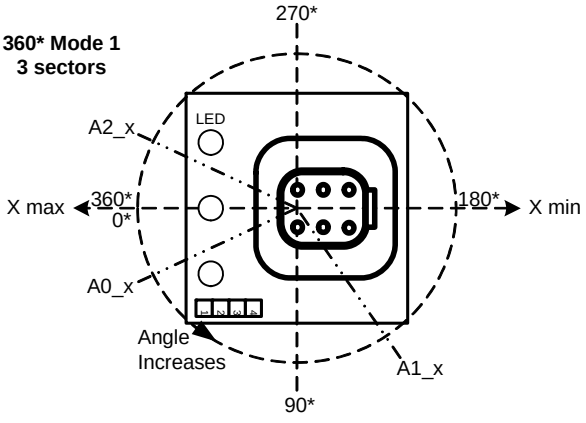
The effect of the driver_value and trigger_angle variables on the controller is dependent on the number of sectors established. See the examples bellow on Output A for a better understanding on all sectors.



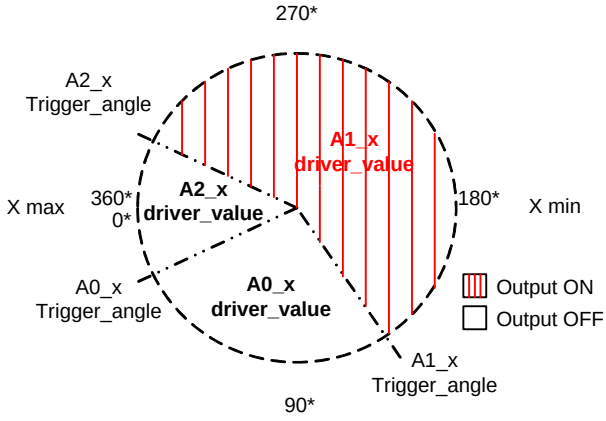
For example If,
A0_x driver_value = 1 (ON)
A1_x driver_value = 0 (OFF)
A2_x driver_value = 1 (ON)
A3_x driver_value = 0 (OFF)
Then the 360* mode 1 operation is as follow:



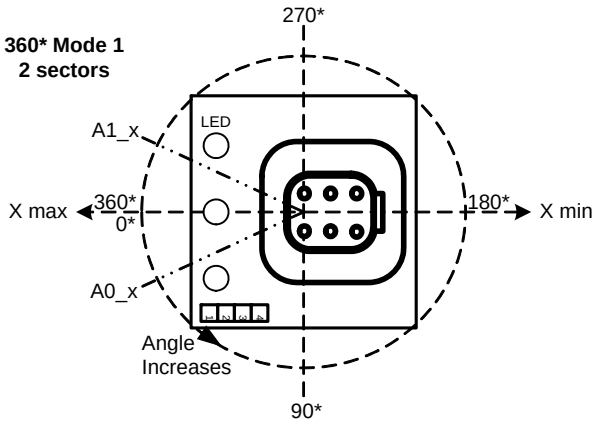
Functionality:
When the angle surpasses A0_x trigger_angle the output changes state to A0_x driver_value = ON until the next sector A1_x trigger_angle is reached.
Once A1_x trigger_angle is surpassed output will change state to the new A1_x driver_value = OFF, until the next A2_x trigger_angle is reached.
Once A2_x trigger_angle is surpassed output will change state to the new A2_x driver_value = ON, until the next A3_x trigger_angle is reached.
Once A3_x trigger_angle is surpassed output will change state to the new A3_x driver_value, until the next A0_x trigger_value is reached.
The cycle then repeats.



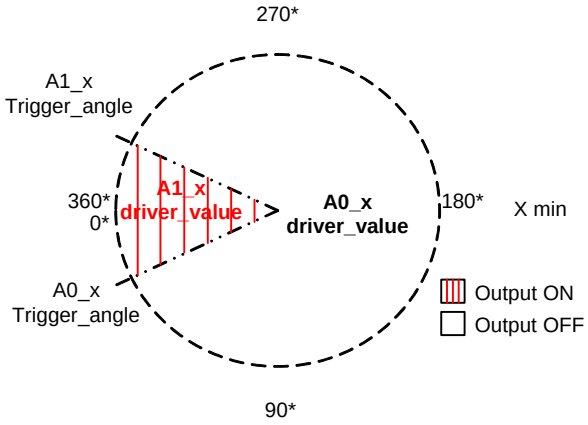
For example If,
A0_x driver_value = 0 (OFF)
A1_x driver_value = 1 (ON)
A2_x driver_value = 0 (OFF)
Then the 360* mode 1 operation is as follow:



Functionality:
When the angle surpasses A0_x trigger_angle the output changes state to A0_x driver_value = OFF until the next sector A1_x trigger_angle is reached.
Once A1_x trigger_angle is surpassed output will change state to the new A1_x driver_value = ON, until the next A2_x trigger_angle is reached.
Once A2_x trigger_angle is surpassed output will change state to the new A2_x driver_value = OFF, until the next A0_x trigger_angle is reached.
The cycle then repeats.



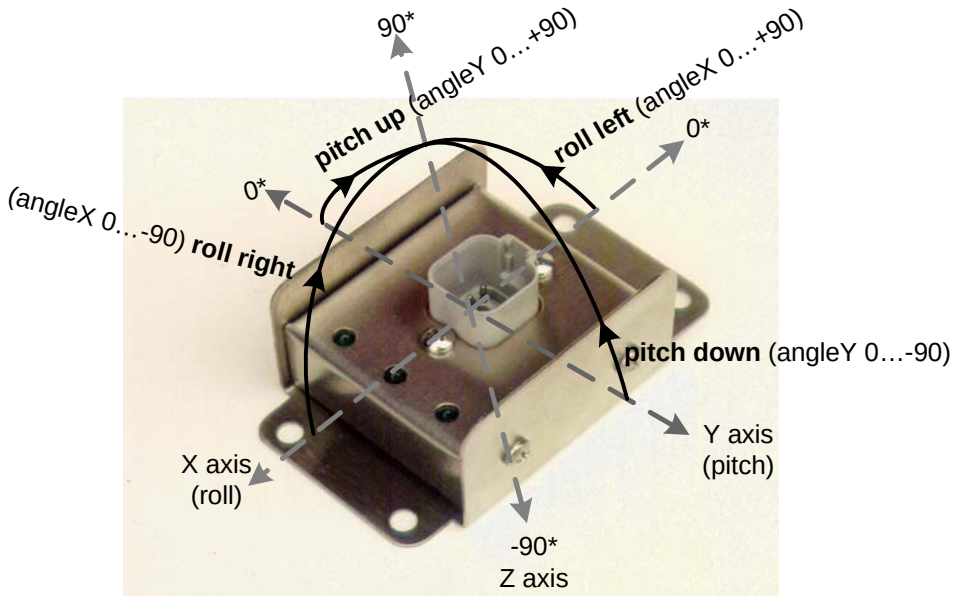
For example If,
A0_x driver_value = 0 (OFF)
A1_x driver_value = 1 (ON)
Then the 360* mode 1 operation is as follow:



Functionality:
When the angle surpasses A0_x trigger_angle the output changes state to A0_x driver_value = OFF until the next sector A1_x trigger_angle is reached.
Once A1_x trigger_angle is surpassed output will change state to the new A1_x driver_value = ON, until the next A0_x trigger_angle is reached.
The cycle then repeats.

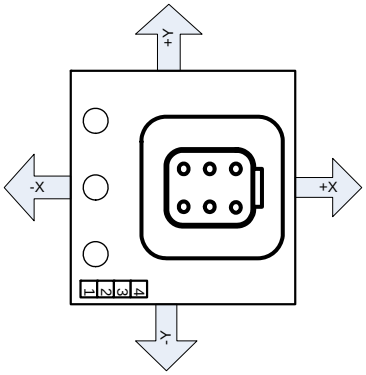
The same logic and functionality applies for Output B, just by changing the letter A for B in the sector name

6.2 Mode 2 - XY



Mode 2 - XY Angle Displacement

The controller is to be mounted on its back with the connector pointing to the sky as per figure above. Visualizing an X and Y co-ordinates, the LED point to X axis negative. See figure bellow for a better understanding.



Mode 2 – XY. X and Y axes orientation

Unlike Mode 1 - 360°, Mode 2 – XY, monitors both X and Y axes and activates or de-activates Output A and(or) B, based on the status of the axes.
This mode only measures from 0° to +70° and from 0° to -70° for both X and Y axes.

- Rolling the controller to the left (tilt downwards toward -X) increases the X angle from 0° all the way to +90°.
- Rolling the controller to the right (tilt downwards toward +X) decreases the X angle from 0° all the way -90°.
- Pitching the controller up (tilt downwards toward -Y) increases the Y angle from 0° all the way to +90°.
- Pitching the controller down (tilt downwards toward +Y) decreases the Y angle from 0° all the way to -90°.

Mode 2 has 4 sectors for each axis, for both Output A and Output B that can be programmed giving it a total of 16 sectors.
4 sectors X axis Output A

4 sectors Y axis Output A
4 sectors X axis Output B
4 sectors Y axis Output B

Not all applications need to have Output A and B actives. If that is the case, just activate Output A as your only output and the 8 sectors (if necessary) for X and Y axis.

Before operating in this mode, the following controller's object variables must be configured as follow:

6.2.1 Setting Operation Mode XY

The operation mode is controlled by the object index 6000 (co_operation_mode).
To set the operation mode to XY set index 6000 (co_operation_mode) sub-index 0 = 0x00000012.
Example using ESD

a. Access the Index 6000 (co_operation_mode) and click on Read
b. Verify that the Hex value is 0x00000012 (18 dec).
c. If not, click on Write, the Write Object window opens. Enter the correct value and click on Write to change the object value. Click on Close to finish.

6.2.2 Setting up the number of XY sectors

For Mode 2 both sectors X and Y axis need to be setup.
The number of sectors in X-axis is controlled by object index 600A (co_number_of_sectors_x).
The number of sectors in Y-axis is controlled by object index 600B (co_number_of_sectors_y).

To set the number of sectors between 1 and 4 for the X-axis, set index 600A (co_number_of_sectors_x) Sub-index 0 = number of sectors required.
To set the number of sectors between 1 and 4 for the Y-axis, set index 600B (co_number_of_sectors_y) Sub-index 0 = number of sectors required.

Note: Wherever the number of sectors set for X-axis will be the same for both Output A and Output B. The same applies for the Y-axis

WARNING! It is a MUST to have 2 or more sectors for proper operation

Example using ESD

a. Set the Index to 600A (co_number_of_sectors_x) and click on Write
b. In Sub:0, enter the value 0x00000002 to set the number of sectors to two (2). Make sure the type is UNSIGNED16. Click on Write to send the new value. Click on Close to finish

6.2.3 Setting up the Mode 2 XY sectors

The sectors are configured by the following variables:

Index	Sub Index	Variable Name	Notes
Output A X sectors			
6010	0	sectors_A0_x	max_subindex

	1		filter_delay (in 10 miliseconds, 0,10,20 .. 5000ms, 500 -> 5000ms)
	2		driver_value (1 or 0)
	3		*t*rigger_angle (80 = 8.0 degrees) value range 0...3599 trigger_angle_A(n) < trigger_angleA(n+1)
	4		Hysteresis (8 = 0.8 degrees)
6020	0	sectors_A1_x	max_subindex
	1		filter_delay
	2		driver_value
	3		trigger_angle
	4		hysteresis
6030		sectors_A2_x	max_subindex
			filter_delay
			driver_value
			trigger_angle
			hysteresis
6040		sectors_A3_x	max_subindex
			filter_delay
			driver_value
			trigger_angle
			hysteresis
Output B X sectors			
6050	0	sectors_B0_x	max_subindex
	1		filter_delay (in 10 miliseconds, 0,10,20 .. 5000ms, 500 -> 5000ms)
	2		driver_value (1 or 0)
	3		*t*rigger_angle (80 = 8.0 degrees) value range 0...3599 trigger_angle_A(n) < trigger_angleA(n+1)
	4		Hysteresis (8 = 0.8 degrees)
6060	0	sectors_B1_x	max_subindex
	1		filter_delay
	2		driver_value
	3		trigger_angle
	4		hysteresis
6070		sectors_B2_x	max_subindex
			filter_delay
			driver_value
			trigger_angle
			hysteresis
6080		sectors_B3_x	max_subindex
			filter_delay
			driver_value
			trigger_angle
			hysteresis
Output A Y sectors			
6110	0	sectors_A0_y	max_subindex
	1		filter_delay (in 10 miliseconds, 0,10,20 .. 5000ms, 500 -> 5000ms)
	2		driver_value (1 or 0)

	3		*t*rigger_angle (80 = 8.0 degrees) value range 0...3599 trigger_angle_A(n) < trigger_angleA(n+1)
	4		Hysteresis (8 = 0.8 degrees)
6120	0	sectors_A1_y	max_subindex
	1		filter_delay
	2		driver_value
	3		trigger_angle
	4		hysteresis
6130		sectors_A2_y	max_subindex
			filter_delay
			driver_value
			trigger_angle
			hysteresis
6140		sectors_A3_y	max_subindex
			filter_delay
			driver_value
			trigger_angle
			hysteresis
Output B Y sectors			
6150	0	sectors_B0_y	max_subindex
	1		filter_delay (in 10 miliseconds, 0,10,20 .. 5000ms, 500 -> 5000ms)
	2		driver_value (1 or 0)
	3		*t*rigger_angle (80 = 8.0 degrees) value range 0...3599 trigger_angle_A(n) < trigger_angleA(n+1)
	4		Hysteresis (8 = 0.8 degrees)
6160	0	sectors_B1_y	max_subindex
	1		filter_delay
	2		driver_value
	3		trigger_angle
	4		hysteresis
6170		sectors_B2_y	max_subindex
			filter_delay
			driver_value
			trigger_angle
			hysteresis
6180		sectors_B3_y	max_subindex
			filter_delay
			driver_value
			trigger_angle
			hysteresis

6.2.4 Filter Delay

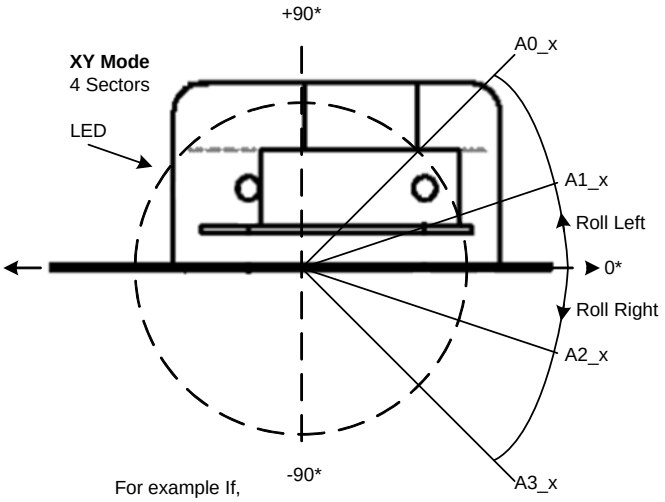
Same as Mode 1 - 360°. Read section 6.1.4 for all details.

6.2.5 Hysteresis

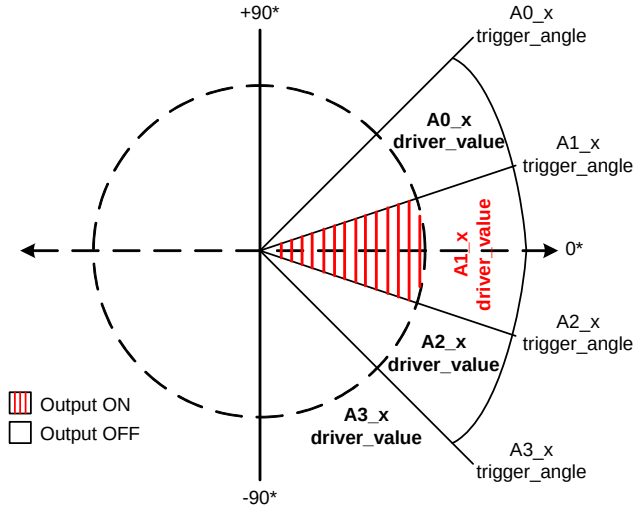
Same as Mode 1 - 360°. Read section 6.1.5 for all details.

6.2.5.1 6.2.6 Mode 2 XY Driver Value and Trigger Angle

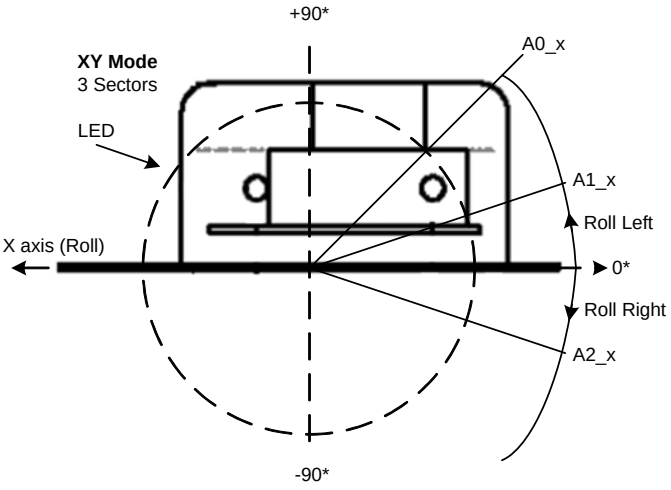
X axis (roll)



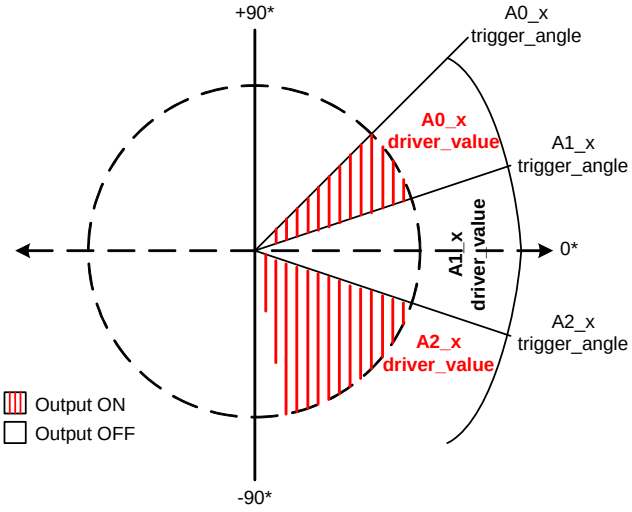
For example If,
A0_x driver_value = 0 (OFF)
A1_x driver_value = 1 (ON)
A2_x driver_value = 0 (OFF)
A3_x driver_value = 0 (OFF)
Then the XY mode 1 operation is as follow:



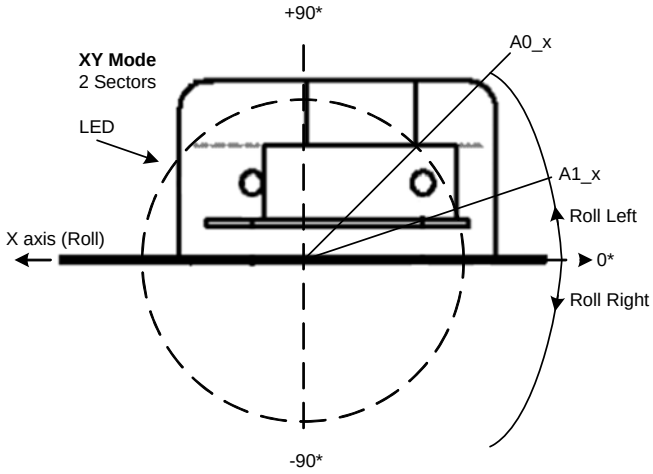
Functionality:
When the angle surpasses A0_x trigger_angle the output changes state to A0_x driver_value = OFF until the next sector A1_x trigger_angle is reached.
Once A1_x trigger_angle is surpassed output will change state to the new A1_x driver_value = ON, until the next A2_x trigger_angle is reached.
Once A2_x trigger_angle is surpassed output will change state to the new A2_x driver_value = OFF, until the next A3_x trigger_angle is reached.
Once A3_x trigger_angle is surpassed output will change state to the new A3_x driver_value = OFF, until the angle reaches -70* degrees.
When going back to zero, any angle that is within a sector will drive the respective output according to the driver_value controlling that sector.



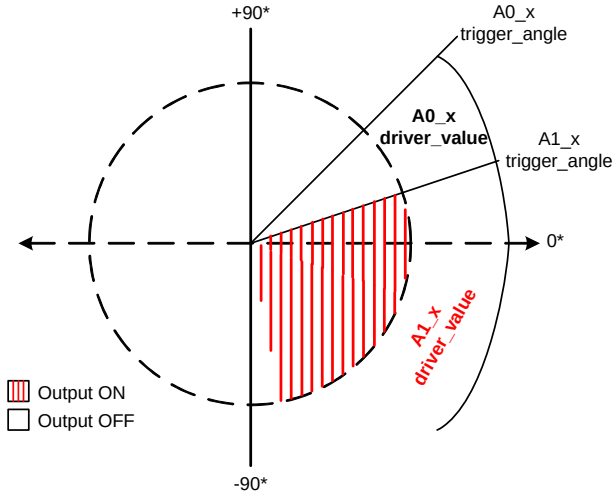
For example If,
A0_x driver_value = 1 (ON)
A1_x driver_value = 0 (OFF)
A2_x driver_value = 1 (ON)
Then the XY mode 1 operation is as follow:



Functionality:
When the angle surpasses A0_x trigger_angle the output changes state to A0_x driver_value = ON until the next sector A1_x trigger_angle is reached.
Once A1_x trigger_angle is surpassed output will change state to the new A1_x driver_value = OFF, until the next A2_x trigger_angle is reached.
Once A2_x trigger_angle is surpassed output will change state to the new A2_x driver_value = ON, until the angle reaches -70* degrees.
When going back to zero, any angle that is within a sector will drive the respective output according to the driver_value controlling that sector.



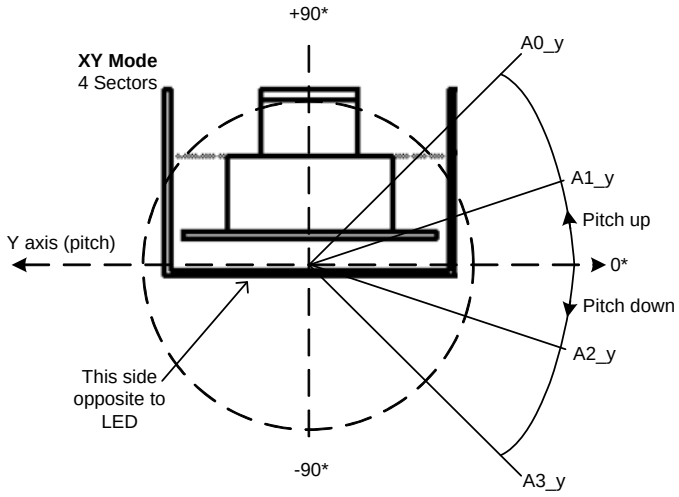
For example If,
A0_x driver_value = 0 (OFF)
A1_x driver_value = 1 (ON)
Then the XY mode 1 operation is as follow:



Functionality:
When the angle surpasses A0_x trigger_angle the output changes state to A0_x driver_value = OFF until the next sector A1_x trigger_angle is reached.
Once A1_x trigger_angle is surpassed output will change state to the new A1_x driver_value = ON, until the angle reaches -70* degrees.
When going back to zero, any angle that is within a sector will drive the respective output according to the driver_value controlling that sector.

Y axis (pitch)

The same explanation for the X axis, applies to the Y-axis. The only difference is the orientation of the controller.



7 Boot Delay

The boot delay is the time that must expire before the output reflects the status of the measured angle. In other words boot delay is like the boot up delay that a specific system might inheritably have. The boot delay on the controller can then be changed to accommodate for the system's boot up delay.

There are two variables that control the boot delay, Output Boot Delay and Output Boot Delay Value.

Output Boot Delay

Holds the numerical value representing the actual Boot Delay time in milliseconds. (First convert the hex value to decimal, and then write it to the index).

Output Boot Delay Value

Status required of the output while the system is booting up. Writing a 1 to the index sets the output ON and writing a 0 (zero) sets the output OFF.

For example let's imagine a system takes 2 seconds to boot up, but it requires the controller to disable any movements in the system by turning ON a valve. The controller can then be programmed to turn the output ON while booting up.

Each output (A and B) has their independent control. The object variables are:

index$_h$	*sub*	*obj.*	*variable*	*Notes*
6002		VAR	co_output_A_boot_delay	Protected variable (in 10 milliseconds, 0,10,20 ... 5000ms) 500 = 5000ms
6003		VAR	co_output_A_boot_delay_value	Protected variable (active while booting ==1 or deactive == 0)
6004		VAR	co_output_B_boot_delay	Protected variable (in 10 milliseconds, 0,10,20 ... 5000ms) 500 = 5000ms
6005		VAR	co_output_B_boot_delay_value	Protected variable (active while booting ==1 or deactive == 0)

Note: 100 decimal = 1000 milliseconds
1 decimal = 10 milliseconds

Example using ESD

a. Set the Index to 6002 (co_output_A_boot_delay) and click on Write
b. In Sub:0, enter the value 0x00000064 to set the boot delay to 1 second. Click on Write to send the new value. Click on Close to finish
c. Set the Index to 6003 (co_output_A_boot_delay_value) and click on Write
d. In Sub:0, enter the value 0x00000001 to set the boot delay value to turn output A ON while booting up delay time expires. Click on Write to send the new value. Click on Close to finish

8 Inverse Mode

The Inverse Mode is controlled by the objects 6006 and 6007. See table bellow

index$_h$	*sub*	*obj.*	*variable*	*Notes*
6006		VAR	co_output_A_inverse_mode_on	Protected variable (0 == not active, 0x23 == output is inverted)
6007		VAR	co_output_B_inverse_mode_on	Protected variable (0 == not active, 0x23 == output is inverted)

In order to understand better the functionality of the Inverse Mode, let's imagine a scenario in which this mode could be used.

Example Scenario:

Normal Operation
A Tilt sensor has been set as follow:

Output A comes ON (where Output A ON = valve ON = machine enabled) when both X and Y angles are within -10° and +10°. So the output stays in its normal status (ON) and the machine is functional as long as both angles don't go past -10° or +10°.

Problem encountered
The problem is that a new customer requires the output to be OFF when both angles are within -10° and +10° because the hydraulic system has a reverse logic (in other words Output A ON = Valve ON = Machine Disable).

Solution
The variable co_output_A_inverse_mode_on is set to 0x23 (hex value) to activate output inverted.
In this mode the output stays in its normal status (now OFF) and the machine is functional as long as both angles don't go past -10° or +10°.

→The Inverse Mode functionality changes the logic (ON or OFF) of the outputs.

***Note:**
If this mode is set correctly there is NO need to utilize the Control Logic Mode

9 Control Logic Mode

Manipulating the variables co_output_A_control_logic and co_output_B_control_logic activates the Control Logic.

index$_h$	*sub*	*obj.*	*variable*	*Notes*
6008		VAR	co_output_A_control_logic	Protected variable, Used only in XY mode, 0x0F -> output_A is on if both inclinometers agree ==LOGIC_AND, 0x0A -> output_A is on if one inclinometer states that == LOGIC_OR
6009		VAR	co_output_B_control_logic	Protected variable, Used only in XY mode, 0x0F -> output_B is on if both inclinometers agree ==LOGIC_AND, 0x0A -> output_B is on if one inclinometer states that == LOGIC_OR

LOGIC AND = Output is active only and only if both axis agree with the required parameters.
LOGIC OR = Output is active if one or the other or both axis agree with the required parameters.

Example Scenario:

LOGIC AND
Output A comes ON (where Output A ON = valve ON = machine enabled) when both X and Y angles are within -10° and +10°. So the output stays in its normal status (ON) and the machine is functional as long as both angles don't go past -10° or +10°.

LOGIC OR
Output A comes ON (where Output A ON = valve ON = machine enabled) when either X or Y (or both) angle is within -10° and +10°. So the output stays in its normal status (ON) and the machine is functional as long as at least one angle does not go past -10° or +10°.

This logic is helpful if for example, a system requires the Output to come ON or OFF (depending on the logic) to show an OUT OF LEVEL signal when either X or Y axis is beyond a preset angle.

*****Note:**
This mode could be achieved by setting the Inverse Mode correctly. If possible, it is recommended to utilize the Inverse Mode instead of setting the Control Logic for the outputs.

10 Zeroing The Tilt Sensor

1.1 Mode 1 360°

To zero the unit in the 360° Mode 1 the hex value 0x36 must be written to the object index 5060 *co_set_mounting_offsets*. See table bellow for all details.

1.2 Mode 2 X-Y

To zero the unit in the X-Y Mode 2 the hex value 0x12 must be written to the object index 5060 *co_set_mounting_offsets*. See table bellow for all details.

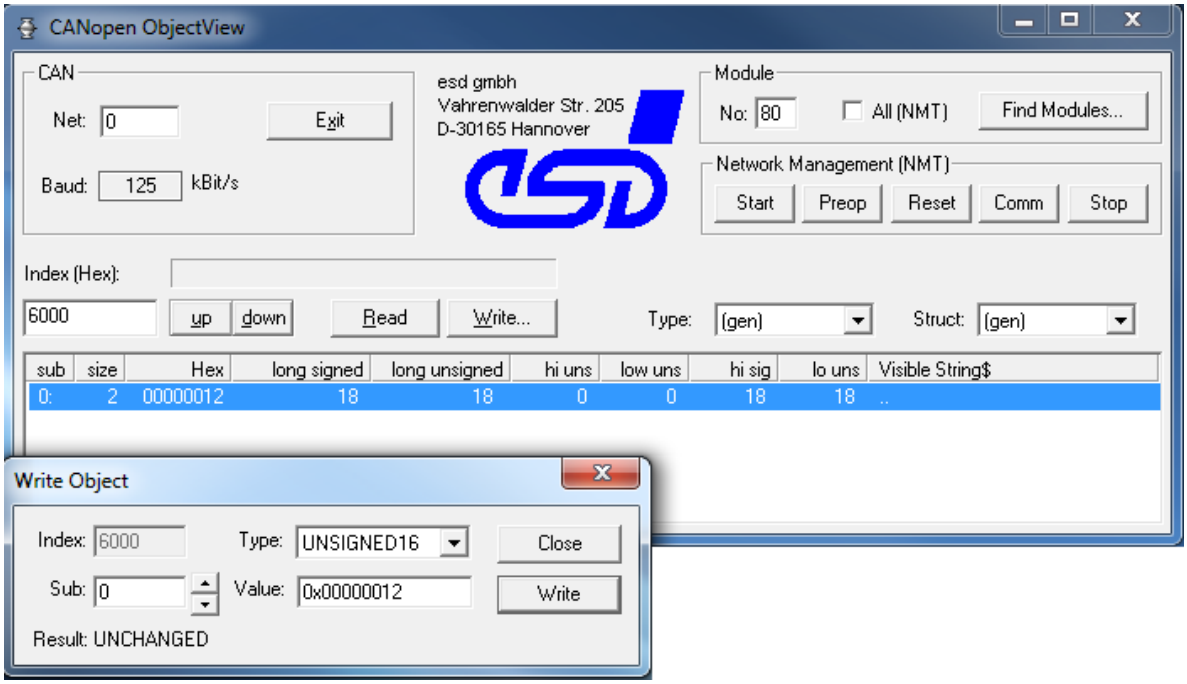
index$_h$	*sub*	*obj.*	*variable*	*Notes*

5060		VAR	co_set_mounting_offsets	Writing 0x12 sets the real angles of X and Y axis (objects 2006 and 2007) as offsets to the objects 5050 and 5051. Writing 0x36 sets the real angle (object 2008) as 360 degrees mode offset to the object 5052.

Example using ESD

a. Set the Index to 5060 (co_set_mounting_offsets) and click on Read

b. In Sub:0, enter the value 0x00000012 to set the mounting offsets for X-Y mode 2 to zero. Click on Write to send the new value. Click on Close to finish



11 Saving Parameters (CAN)

To save the application specific changes to the parameters, write the hex value 0x65766173 in sub 0 to the index 1010 *co_store_params*. See table bellow for all details.

index$_h$	*sub*	*obj.*	*variable*	*type*	*bytes*	*access*	*default value (profile)*	*Static value (flash)*	*Notes*
1010	0	ARRAY	co_store_params	U32	4	RO	1-127	6	
	1			U32	4	RW		1	All settings (0x 65 76 61 73 saves the parameters)
	2			U32	4	RW		1	Communication 1000-1FFF
	3			U32	4	RW		1	Application 6000-9FFF
	4			U32	4	RW		1	Manufacturer 2000-5FFF

12 Setting Sensor for constant angle transmission over CAN

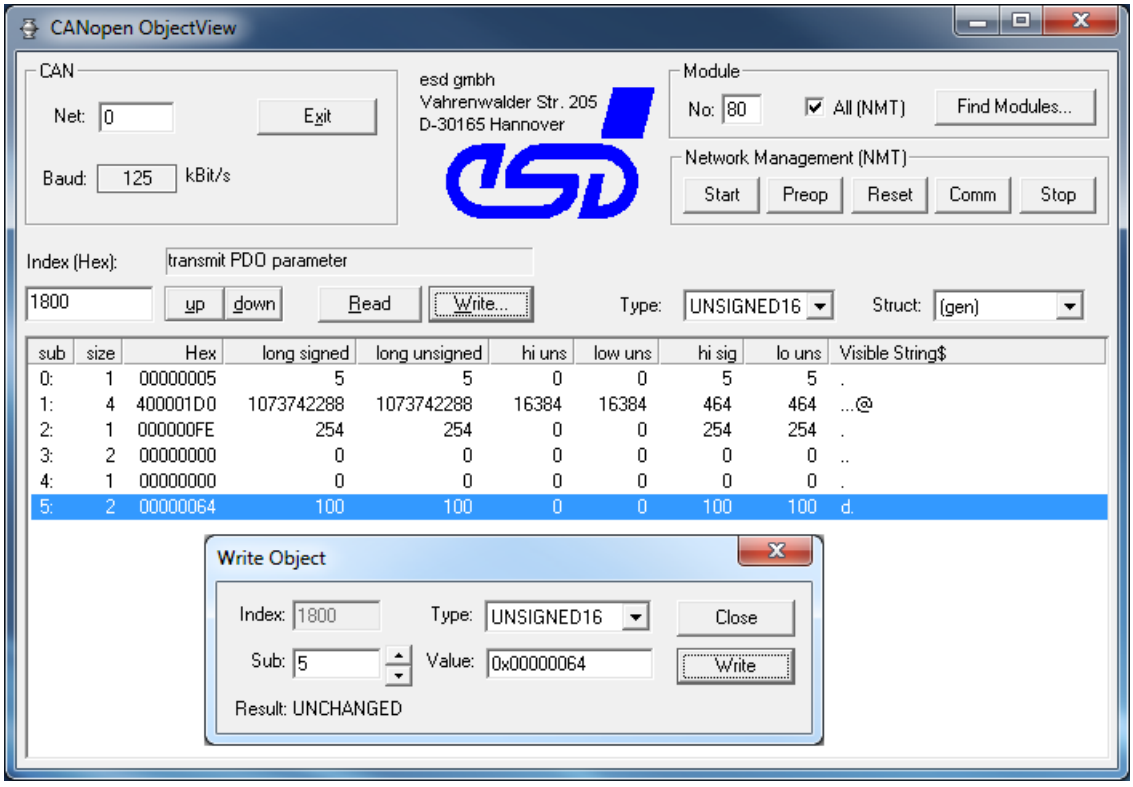
In order to have the tilt sensor unit transmitting the angles over CANopen the user MUST activate the Transmit PDO event timer. This even is located in object 1800 h. See page 6 of the User's Manual. *This could be done while the unit is operational*. Follow the steps below:

1. Access object 1800 h
2. Go to sub-index 5 (in object 1800 h) which is the Event timer for the transmit PDO (1800 h). This sub-index has Read and Write access.
3. Write to this sub-index 5, a hexadecimal value that represents the time interval in milliseconds.
   For example:
   If you want the unit to transmit the angles every 100 mS
   Go to object 1800 h sub-index 5 and write a 0x64 (hex) = 100 decimal value

   See table below for some other typical values

Hex Value	Decimal Value	PDO Transmission Interval
64	100	100 mS
1F4	500	500 mS
3E8	1000	1000 mS = 1 second

1800	0	RECORD	co_tpdo_comm_par[0]	U8	1	RO	2-5	5	
	1			U32	4	RW	0x180 + nodeID	0x80000000	PDO:n cob-id
	2			U8	1	RW		0xFE	Transmit type
	3			U16	2	RW		0	Inhibit time
	4			U8	1	RW		0	Reserved
	5			U16	2	RW		0	Event timer
1A00	0	RECORD	co_tpdo_map_par[0]	U8	1	RO		3	
	1			U32	4	RO		0x20010010	
	2			U32	4	RO		0x20020010	
	3			U32	4	RO		0x20030010	
2000		VAR	co_temperature	I8	1	RO		25	
2001		VAR	co_angle_X	I16	2	RO		0	Includes the mounting offset object 5050. value range -900 to 900 (-90.0 – 90.0) degrees
2002		VAR	co_angle_Y	I16	2	RO		0	Includes the mounting offset object 5051.value range -900 to 900 (-90.0 – 90.0) degrees
2003		VAR	co_angle_360	I16	2	RO		0	Includes the mounting offset object 5052. value range 0 to 3599 (0 – 359.9) degrees



4. After the desired Event Timer value has been written, save the changes by going to object 1010 h, sub-index 1 and write the following value 0x65766173.

The angles are transmitted in ID = 0x180 +Node ID.

For example, the default ID for this sensor is 0x50. Therefore, the angles will be transmitted in:

0x180 + 0x50 = 0x1D0

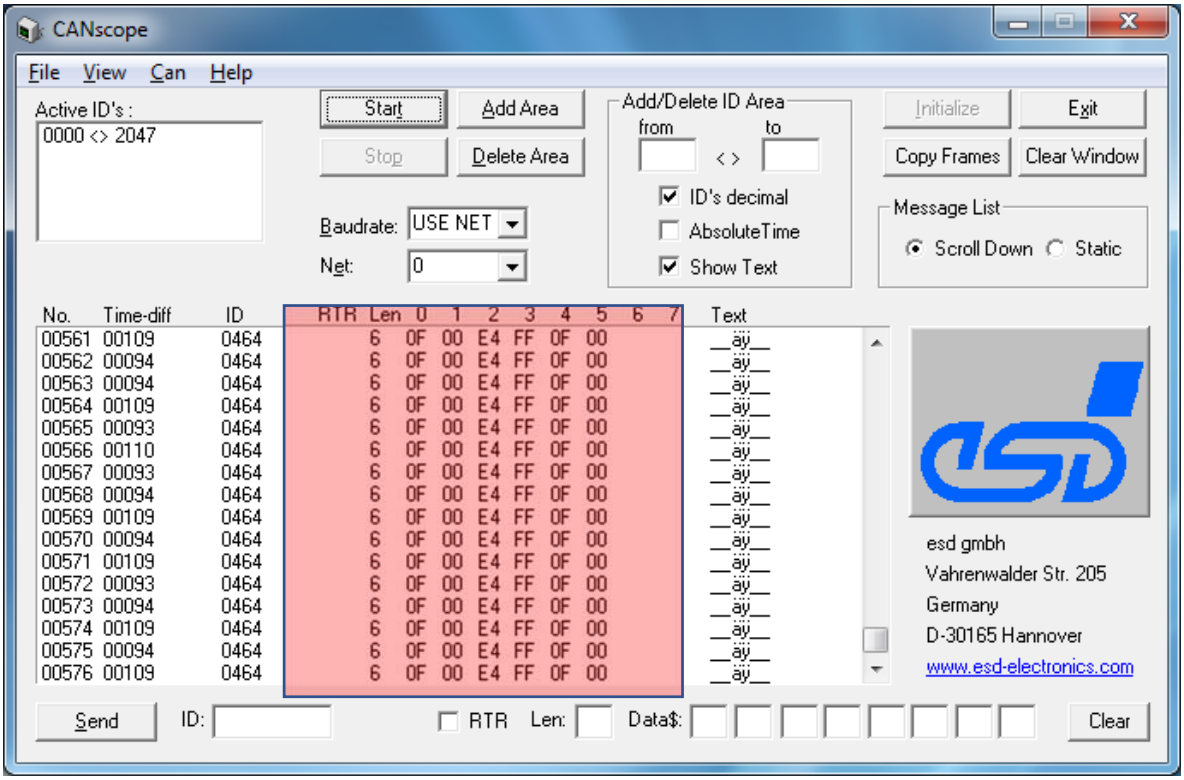
DATA INDEX

The data transmitted over CANopen is transmitted over 6 bytes and it is arranged in the following way:

ID	D1	D2	D3	D4	D5	D6
0x180 + Node ID	MS byte	LS byte	MS byte	LS byte	MS byte	LS byte
	Angle X		Angle Y		Angle 360◦	

Where,
MS byte = Most Significant Byte
LS byte = Least Significant Byte

See figure below for an example,



13 Version History

Version	Date	Author	Modifications
1	December 18th, 2018	Erik Sasse	Initial Draft