FIRMWARE MANUAL

UMC motion control firmware

Description

UMC firmware is the generic firmware of the whole range of UMC UniSwarm motions controllers. Functionalities can vary following your hardware board reference and revision. Please check the specific hardware datasheet.

Features

- Compatibility with CANOpen protocol with CiA DS402 profile
- Type of motors can be changed without hardware modification

Interfaces

- CAN Fd bus compatible with CANOpen
- RS485 / RS422 interface for protocols like Modbus, Profibus or DMX512...

Motor

- DC brushed, brush-less and stepper
- Torque, speed and position limits set by software

Sensors

- Configurable sensors inputs
- Position, velocity, torque, limit switches

Mode

- TQ: Torque profile mode
- VL: Velocity mode
- PP: Profile position mode
- IP: Interpolated position mode

Contents

Chapter 1

Operating modes

1.1 Drives and motion control device profile

The CiA 402 specifies the CANopen interface for management of drive and motion controllers. This profile allow to control the controllers via several operating modes:

- Torque modes:
  - Profile torque mode (TQ)
- Velocity modes:
  - Velocity Mode (VL)
- Position modes:
  - Profile Position Mode (PP)
  - Interpolated position mode (IP)

1.1.1 Object

In this chapter, most of the objects described are indexed in the Standardized profile area. (see Object dictionary) with the index range 0x6000 to 0x9FFF.

1.1.2 Multi axes

In the index range of the standardized profile area (0x6000 to 0x6FFF), the device can manage up to 2 axes. In multi-axis devices, objects can be accessed by shifting as follows:

Index range	Description
0x6000 to 0x67FF	Axis 1
0x6800 to 0x6FFF	Axis 2

Table 1.1: Index range of standardized profile area

1.1.3 The power drive system (PDS)

The PDS FSA can be seen as a black box that control the application behavior when a control device is using the PDS.
The PDS assure control even if the communication doesn't work properly.
The PDS FSA depends on CanOpen status.
The controlword received via the network and local signals are used to manage the PDS.
The statusword is produced by the drive device and used to send the state of the PDS.
Error detection signals are used to drive the PDS FSA.
The status and the possible control sequence of the PDS are defined by the FSA of the PDS.
Each special internal and external behavior is represented by a unique state.
The commands accepted depend on the state of the PDS FSA.

State machine

The state machine of the power drive system controls the power electronics according to user parameters and events that occurred.

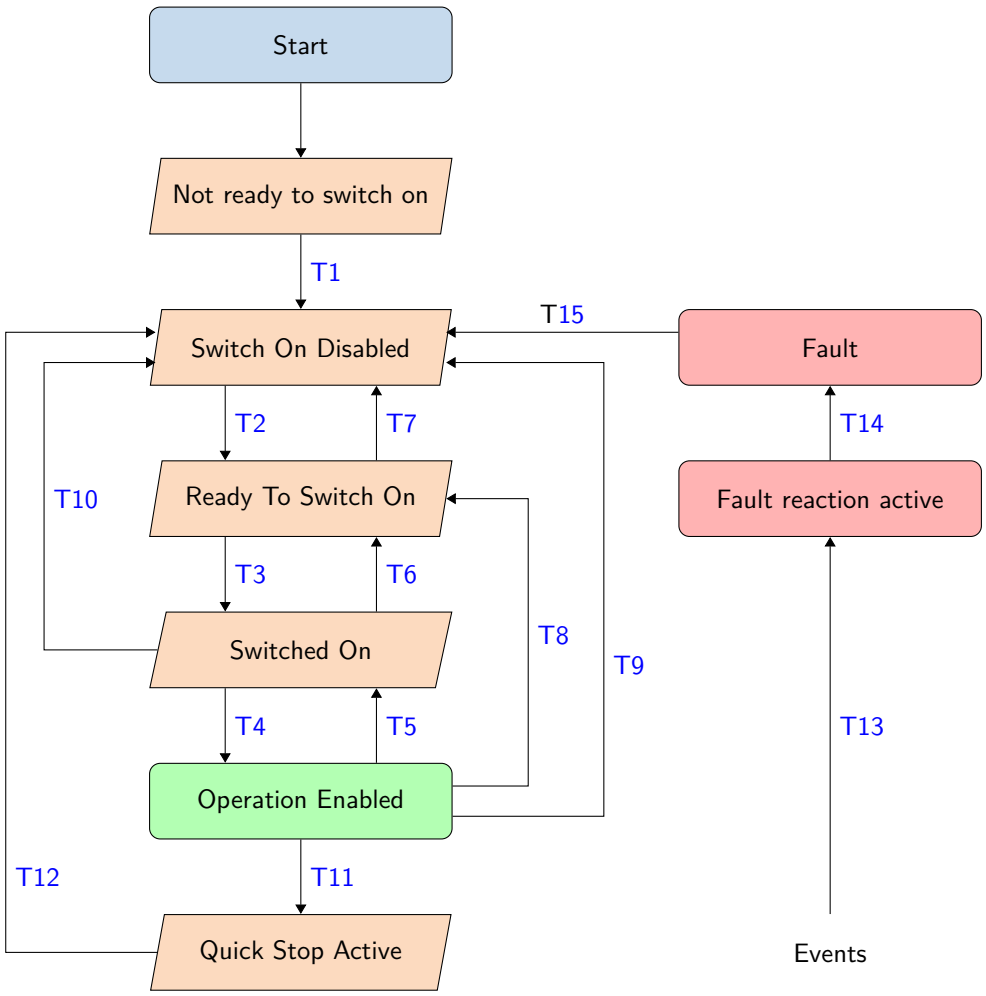


Figure 1.1: State machine of 402

When the system is started, it enters the start state and automatically switches to the Not ready to switch onstate if NMT is in the Start state.

Supported functions

Function	States							
	Not ready to switch on	Switch On Disabled	Ready To Switch On	Switched On	Operation Enabled	Quick Stop Active	Fault reaction active	Fault
Brake applied,if present	Yes	Yes	Yes	Yes	Yes/No	Yes/No	Yes/No	Yes
Low-level power applied	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
High-level power applied	Yes/No	Yes/No	Yes/No	Yes	Yes	Yes	Yes	Yes/No
Drive function Enabled	No	No	No	No	Yes	Yes	Yes	No
Config-uration Allowedd	Yes	Yes	Yes	Yes	Yes/No	Yes/No	Yes/No	Yes

Table 1.2: Supported functions

Transition events and actions

Transition	Event(s)	Action(s)
T0	Automatic transition after power-on or reset application	Drive device self-test and/or self initialization shall be performed.
T1	Automatic transition	Communication shall be activated.
T2	Shutdown command from control device or local signal	None
T3	Switch on command received from control device or local signal	The high-level power shall be switched on, if possible.
T4	Enable operation command received from control or local signal	The drive function shall be enabled and all internal set-points cleared.
T5	Disable operation command received from control device or local signal	The drive function shall be disabled.
T6	Shutdown command received from control device or local signal	The high-level power shall be switched off, if possible.
T7	Quick stop or disable voltage command from control device or local signal	None
T8	Shutdown command from control device or local signal	The drive function shall be disabled, and the high-level power shall be switched off, if possible.
T9	Disable voltage command from control device or local signal	The drive function shall be disabled, and the high-level power shall be switched off, if possible.
T10	Disable voltage or quick stop command from control device or local signal	The high-level power shall be switched off, if possible.
T11	Quick stop command from control device or local signal	The quick stop function shall be started.
T12	Automatic transition when the quick stop function is completed and quick stop option code is 1, 2, 3 or 4, or disable voltage command received from control device (depends on the quick stop option code)	The drive function shall be disabled, and the high-level power shall be switched off, if possible.
T13	Fault signal	The configured fault reaction function shall be executed.
T14	Automatic transition	The drive function shall be disabled; the high-level power shall be switched off, if possible.
T15	Fault reset command from control device or local signal	A reset of the fault condition is carried out, if no fault exists currently on the drive device; after leaving the fault state, the fault reset bit in the controlword shall be cleared by the control Device.

Table 1.3: Transition options

When the motor is not connected to the high level power, it must rotate freely if it is not braking.

When the drive function is deactivated the motor must not receive power. All the control values such as torque, velocity or position should not be treated.

If the drive function is activated, energy can be provided to the motor and control values should be treated.

If there is a default detected by the drive device, it should reach the active fault state. When this state is reached a special fault reaction is executed.

After this reaction, the state of the drive device changes to the fault state.

When the default is no longer active, the fault state can be left after a reset command.

After a fatal error, it is necessary to switch-off the drive device instantly, as it can no longer control the motor.

Transition options The behavior of transitions can be modified with the following objects:

Transition	Object
Events	0x6007 a1_Abort_connection_option
T11	0x605A a1_Quick_stop_option
T8	0x605B a1_Shutdown_option
T5	0x605C a1_Disable_operation_option
T13	0x605E a1_Fault_reaction_option

Table 1.4: Transition options

1.1.4 Modes of operation

Several modes of operation are implemented and user selectable. The modes can not operate in parallel. Inconsistencies and incorrect behaviour should be avoided by the control device.

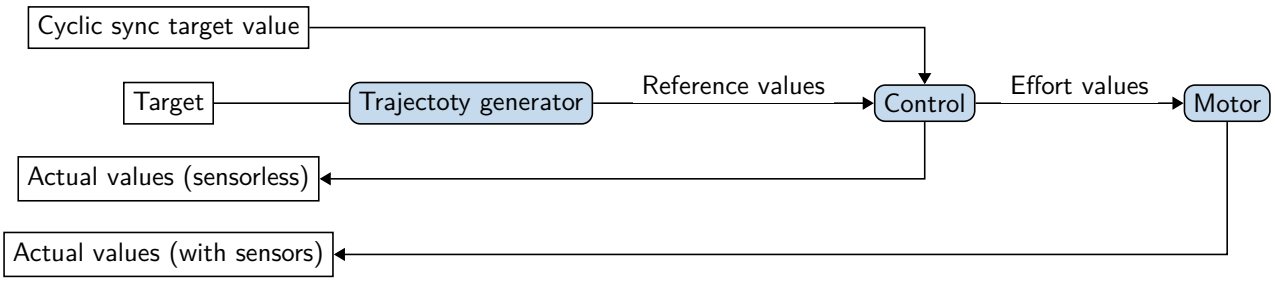


Table 1.5: Relation between different value parameters

The 0x6060 a1_Modes_of_operation object enables speed mode by setting the value to 2.

1.1.5 Definition of parameters

All the objects described below correspond to axis 1, see Multi axes for ohter axes.

Input objects

Name	Description
0x6040 a1_Controlword	control the power drive system state machine
0x6060 a1_Modes_of_operation	modify the operation mode
0x6007 a1_Abort_connection_option	
0x605A a1_Quick_stop_option	
0x605B a1_Shutdown_option	
0x605C a1_Disable_operation_option	
0x605D a1_Halt_option	
0x605E a1_Fault_reaction_option	

Table 1.6: Configuration objects

Output objects

Name	Description
0x6041 a1_Statusword	status of the power drive system finite state
0x6061 a1_Modes_of_operation_display	
0x6044 a1_vl_Velocity_actual_value	

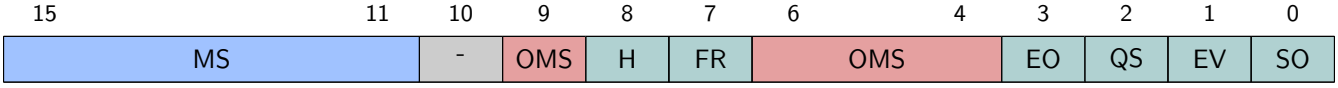
Table 1.7: Status objects

0x6040 a1_Controlword

This object allow to control the power drive system state machine. The start of any movement is specific to the operating mode.

Index	SubIndex	Name		
0x6040	0	a1_Controlword		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT16	RW,RPDO	0	-	[;]

Figure 1.2: Object description 0x6040.0 *a1_Controlword*



- MS: Manufacturer-specific
- OMS: Operation mode specific, mode depending
- H: Halt, behavior depending on the specific operating mode :
  Activate or deactivate the controlled movement, the behavior of Halt depends on the specific operating mode and is adjustable by 0x605D a1_Halt_option.
- FR: Fault reset
- EO: Enable operation
- QS: Quick stop
- EV: Enable voltage
- SO: Switch on

Figure 1.3: Frame of Controlword

Command	Control Word					Transitions
	Bit 7	Bit 3	Bit 2	Bit 1	Bit 0	
Shutdown	0	X	1	1	0	T2, T6, T8
Switch on	0	0	1	1	1	T3
Switch on + enable operation	0	1	1	1	1	T3 + T4
Disable voltage	0	X	X	0	X	T7,9,T10,T12
Quick stop	0	X	0	1	X	T7,T10,T11
Disable operation	0	0	1	1	1	T5
Enable operation	0	1	1	1	1	T4,T16
Fault reset	1	X	X	X	X	T15

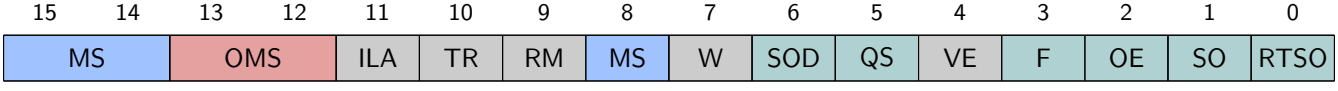
Table 1.8: Controlword: Command coding

0x6041 a1_Statusword

This object provide the status of the power drive system finite state automaton. The state machine state is provided by the bit combination of 0x6041 *a1_Statusword*.

Index	SubIndex	Name		
0x6041	0	a1_Statusword		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT16	RO,TPDO	-	-	[;]

Figure 1.4: Object description 0x6041.0 *a1_Statusword*



- b0 **RTSO**: Ready to switch on
- b1 **SO**: Switched on

- b2 **OE**: Operation enabled
- b3 **F**: Fault
- b4 **VE**: Voltage enabled
- b5 **QS**: Quick stop
- b6 **SOD**: Switch on disabled
- b7 **W**: Warning
- b8 **MS**: Manufacturer-specific
- b9 **RM**: remote
- b10 **TR**: Target reached
- b11 **ILA**: Internal limit active, mode depending
- b12-13 **OMS**: Operation mode specific, mode depending
- b14-15 **MS**: Manufacturer-specific

Figure 1.5: Frame of Statusword

Bit 6,5,3,2,1,0	State
xxxx xxxx x0xx 0000	Not ready to switch on
xxxx xxxx x1xx 0000	Switch on disabled
xxxx xxxx x01x 0001	Ready to switch on
xxxx xxxx x01x 0011	Switched on
xxxx xxxx x01x 0111	Operation enabled
xxxx xxxx x00x 0111	Quick stop active
xxxx xxxx x0xx 1111	Fault reaction active
xxxx xxxx x0xx 1000	Fault

Table 1.9: Statusword: State coding

0x6060 a1_Modes_of_operation

This object allow to modify the operation mode.

Index	SubIndex	Name		
0x6060	0	a1_Modes_of_operation		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT8	RW,RPDO	4	-	[-128;10]

Figure 1.6: Object description 0x6060.0 *a1_Modes_of_operation*

Value	Definition
0	No mode change/no mode assigned
1	Profile Position Mode
2	Velocity Mode
3	Profile velocity mode
4	Profile torque mode
5	Reserved
6	Homing mode
7	Interpolated position mode
8	Cyclic sync position mode
9	Cyclic sync velocity mode
10	Cyclic sync torque mode
11	Cyclic sync torque mode with commutation angle

Figure 1.7: Option of Modes of operation

0x6061 a1_Modes_of_operation_display

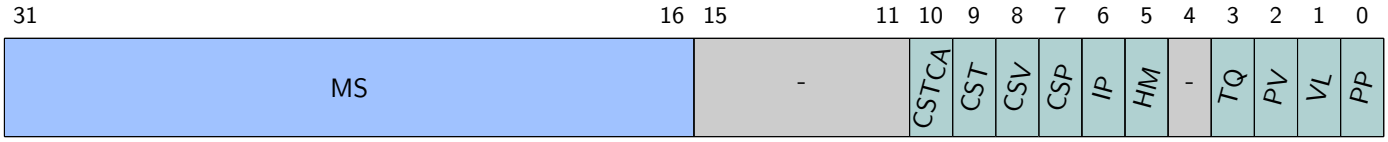
This object provide the actual operation mode.

Index	SubIndex	Name		
0x6061	0	a1_Modes_of_operation_display		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT8	RO,TPDO	-	-	[-128;10]

Figure 1.8: Object description 0x6061.0 *a1_Modes_of_operation_display*

0x6502 a1_Supported_drive_modes

This object provide the supported drive modes.



- MS: Manufacturer-specific
- CSTCA: Cyclic sync torque mode with commutation angle
- CST: Cyclic sync torque mode
- CSV: Cyclic sync velocity mode
- CSP: Cyclic sync position mode
- IP: Interpolated position mode
- HM: Homing mode
- TQ: Profile torque mode
- PV: Profile velocity mode
- VL: Velocity Mode
- PP: Profile Position Mode

Figure 1.9: Frame of Supported drive modes

Bit	Value
0	Not supported
1	Supported

Table 1.10: Supported drive modes: coding

Index	SubIndex	Name		
0x6502	0	a1_Supported_drive_modes		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RO	75	-	[0;0xFFFFFFFF]

Figure 1.10: Object description 0x6502.0 *a1_Supported_drive_modes*

0x6007 a1_Abort_connection_option

This object is used to configure the command to be executed when an event occurs:

- Bus-off: problem on the network
- Heartbeat, Node guarding
- Network management services (NMT) events: change of status (NMT Start; Stop, PreOp)

- Network management services (NMT) reset: reset application and reset communication

Index	SubIndex	Name		
0x6007	0	a1_Abort_connection_option		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW	1	-	[0;3]

Figure 1.11: Object description 0x6007.0 *a1_Abort_connection_option*

Value	Definition
0	No action
1	Fault signal
2	Disable voltage command
3	Quick stop command

Figure 1.12: Option of Abort connection

Note: Each option can be configured with a corresponding object (0x605E a1_Fault_reaction_option, 0x605C a1_Disable_operation_option and 0x605A a1_Quick_stop_option

0x605A a1_Quick_stop_option

This object is used to configure the action to be executed when a quick stop function occurs.

Index	SubIndex	Name		
0x605A	0	a1_Quick_stop_option		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW	2	-	[0;6]

Figure 1.13: Object description 0x605A.0 *a1_Quick_stop_option*

Value	Definition
0	Disable drive function
1	Slow down ramp (mode depending) and stay in Switch On Disabled
2	Quick stop ramp (mode depending) and stay in Switch On Disabled
5	Slow down ramp (mode depending) and stay in Quick Stop Active
6	Quick stop ramp (mode depending) and stay in Quick Stop Active

Figure 1.14: Option of Quick stop

0x605B a1_Shutdown_option

This object is used to configure the action to be executed when there is a transition from Operation Enabled state to Ready To Switch On state.

Index	SubIndex	Name		
0x605B	0	a1_Shutdown_option		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW	1	-	[0;1]

Figure 1.15: Object description 0x605B.0 *a1_Shutdown_option*

Value	Definition
0	Disable the drive function (switch-off the drive power stage)
1	Slow down ramp (mode depending) and deactivation of the drive function

Figure 1.16: Option of Shutdown option

0x605C a1_Disable_operation_option

This object is used to configure the action to be executed when there is a transition from Operation Enabled state to Switched On state.

Index	SubIndex	Name		
0x605C	0	a1_Disable_operation_option		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW	1	-	[0;1]

Figure 1.17: Object description 0x605C.0 *a1_Disable_operation_option*

Value	Definition
0	Disable the drive function (switch-off the drive power stage)
1	Slow down ramp (mode depending) and deactivation of the drive function

Figure 1.18: Option of Disable operation

0x605D a1_Halt_option

This object is used to configure the action to be executed when a halt function occurs.

Index	SubIndex	Name		
0x605D	0	a1_Halt_option		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW	2	-	[1;2]

Figure 1.19: Object description 0x605D.0 *a1_Halt_option*

Value	Definition
1	Slow down ramp, mode depending
2	Quick stop ramp, mode depending

Figure 1.20: Option of Halt

Note: Whatever the option, the state of the power drive system stays at Operation Enabled

0x605E a1_Fault_reaction_option

This object is used to configure the action to be executed when a fault is detected.

Index	SubIndex	Name		
0x605E	0	a1_Fault_reaction_option		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW	2	-	[0;2]

Figure 1.21: Object description 0x605E.0 *a1_Fault_reaction_option*

Value	Definition
0	Disable drive function, motor is free to rotate
1	Slow down ramp, mode depending
2	Quick stop ramp, mode depending

Figure 1.22: Option of Fault reaction

1.1.6 Error codes

Error codes	Meaning
2110	Short circuit/earth leakage (input)
2230	Short circuit/earth leakage (device internal)
2320	Short circuit/earth leakage (motor-side)
2350	Load level fault (I 2 t, thermal state)
2351	Load level warning (I 2 t, thermal state)
3110	Mains over-voltage
3120	Mains under-voltage
3130	Phase failure
3140	Mains frequency
3210	DC link over-voltage
3220	DC link under-voltage
3230	Load error
3310	Output over-voltage
3320	Armature circuit
3330	Field circuit
4300	Temperature drive
4400	Temperature supply
5100	Supply device hardware
5110	Supply low voltage
5120	Supply intermediate circuit
5200	Control device hardware
5300	Operating unit
5400	Power section
5410	Output stages
5420	Chopper
5430	Input stages
5440	Contacts
5450	Fuses
5500	Hardware memory
6010	Software reset (watchdog)
6310	Loss of parameters
6320	Parameter error
7100	Power additional modules
7110	Brake chopper
7120	Motor
7200	Measurement circuit
7300	Sensor
7400	Computation circuit
7500	Communication
7600	Data storage (external)
8400	Velocity speed controller
8500	Position controller
8600	Positioning controller
8700	Sync controller
8800	Winding controller
8900	Process data monitoring
8A00	Control monitoring

Table 1.11: Error codes

1.2 Profile torque mode

1.2.1 Introduction

This mode allows operation at regulated torque via a ramp generator that can be configured by various settings, input torque adjustment, slope ramp adjustment and other optional parameters.

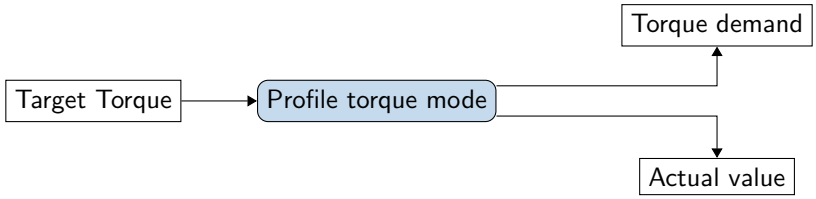


Figure 1.23: Profile torque mode

1.2.2 Functional description

Profile torque mode

Here is the functional diagram with all the parameters.

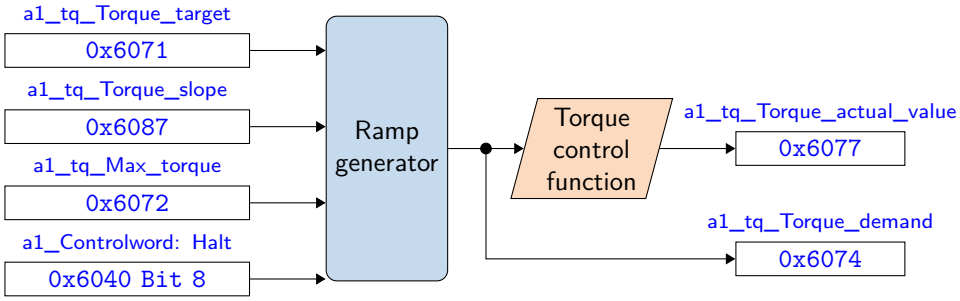


Figure 1.24: functional diagram of Velocity mode with all objects

Ramp generator

The Profile torque mode is used the 0x6087 *a1_tq_Torque_slope*for generate ramp.

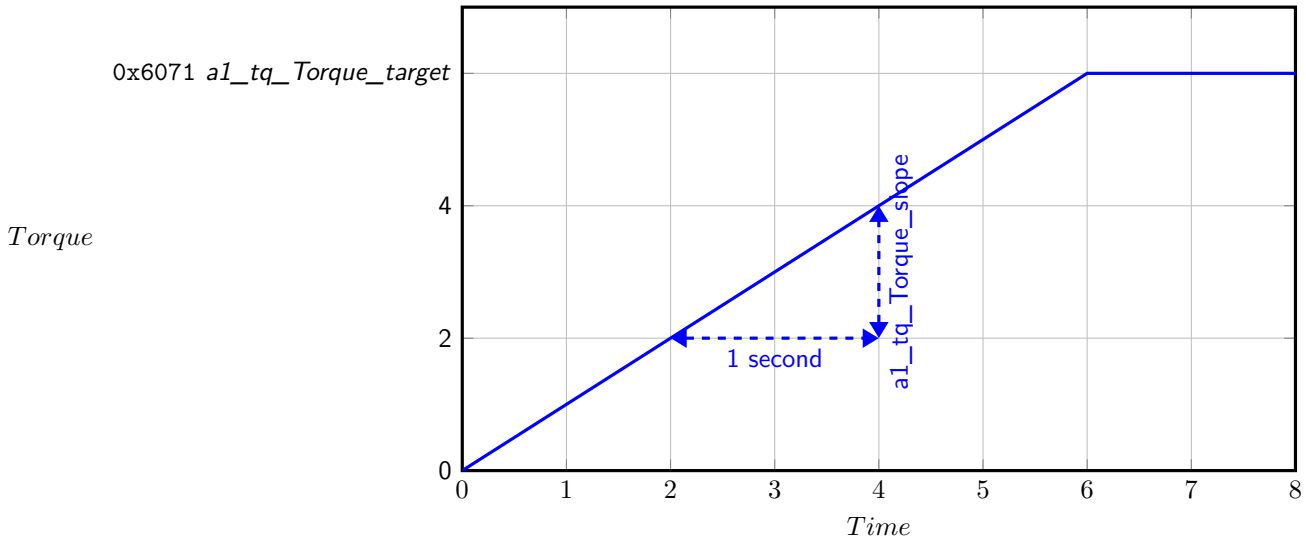


Figure 1.25: Feature of torque ramp

Mode activation

The 0x6060 a1_Modes_of_operation object enables speed mode by setting the value to 4.

1.2.3 Definition of parameters

All the objects described below correspond to axis 1, see Multi axes for ohter axes.

Input objects

Name	Description
0x6060 a1_Modes_of_operation	Modes of operation
0x6071 a1_tq_Torque_target	Requested target torque
0x6087 a1_tq_Torque_slope	Slope of ramp
0x6072 a1_tq_Max_torque	Max of torque

Table 1.12: Configuration objects

Output objects

Name	Description
0x6041 a1_Statusword	Status
0x6074 a1_tq_Torque_demand	Torque generated by the generator trajectory
0x6077 a1_tq_Torque_actual_value	Actual Torque value

Table 1.13: Status objects

0x6040 a1_Controlword

The bit 8 of 0x6040 a1_Controlword are used in Profile torque mode.

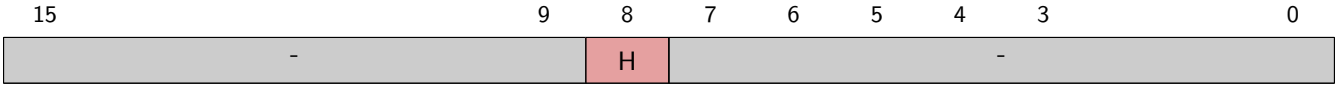


Table 1.14: Frame of Controlword for Profile torque mode

Bit	Name	Value	Definition
8	**H**: Halt	0	No command
		1	Motor stops according to the halt option code (see 0x605D a1_Halt_option)

Table 1.15: Definition of Controlword bits (bit 8)

0x6041 a1_Statusword

The bit 11 (Internal limit active) of 0x6041 a1_Statusword is used in Profile torque mode.

Table 1.16: Frame of Statusword for Profile torque mode

Bit	Name	Value	Definition
10	**TR**: Target reached	0	Halt = 0: Target torque not reached,
			Halt = 1: Axis decelerates
		1	Halt = 0: Target torque reached,
			Halt = 1: Velocity of axis is 0

Table 1.17: Definition of Statusword bits (bit 10)

0x6071 a1_tq_Torque_target

This object is torque set-point. The value shall be given per thousand of rated torque.

Index	SubIndex	Name		
0x6071	0	a1_tq_Torque_target		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW,RPDO	0	-	[-0x7FFF;0x7FFF]

Figure 1.26: Object description 0x6071.0 *a1_tq_Torque_target*

0x6072 a1_tq_Max_torque

The configured maximum permissible torque in the motor. The value shall be given per thousand of rated torque.

Index	SubIndex	Name		
0x6072	0	a1_tq_Max_torque		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT16	RW,RPDO	30000	-	[0;0xFFFF]

Figure 1.27: Object description 0x6072.0 *a1_tq_Max_torque*

0x6074 a1_tq_Torque_demand

The output value of the trajectory generator. The value shall be given per thousand of rated torque.

Index	SubIndex	Name		
0x6074	0	a1_tq_Torque_demand		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	0	-	[-0x7FFF;0x7FFF]

Figure 1.28: Object description 0x6074.0 *a1_tq_Torque_demand*

0x6077 a1_tq_Torque_actual_value

The actual value of the torque. It shall correspond to the instantaneous torque in the motor. The value shall be given per thousand of rated torque.

Index	SubIndex	Name		
0x6077	0	a1_tq_Torque_actual_value		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	0	-	[-0x7FFF;0x7FFF]

Figure 1.29: Object description 0x6077.0 *a1_tq_Torque_actual_value*

0x6078 a1_tq_Current_actual_value

The actual value of the current. It shall correspond to the current in the motor. The value shall be given per thousand of rated current.

Index	SubIndex	Name		
0x6078	0	a1_tq_Current_actual_value		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO	0	-	[-0x7FFF;0x7FFF]

Figure 1.30: Object description 0x6078.0 *a1_tq_Current_actual_value*

0x6087 a1_tq_Torque_slope

The configured rate of change of torque. The value shall be given in units of per thousand of rated torque per second.

Index	SubIndex	Name		
0x6087	0	a1_tq_Torque_slope		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW,RPDO	1	-	[0;0xFFFFFFFF]

Figure 1.31: Object description 0x6087.0 *a1_tq_Torque_slope*

0x6088 **a1_tq_Torque_profile_type**

The configured type of profile used to perform a torque change.

Index	SubIndex	Name		
0x6088	0	a1_tq_Torque_profile_type		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW	0	-	[0;1]

Figure 1.32: Object description 0x6088.0 *a1_tq_Torque_profile_type*

Value	Definition
0	Linear ramp

Figure 1.33: Option of 0x6088 *a1_tq_Torque_profile_type*

1.3 Velocity Mode

1.3.1 Introduction

This mode allows operation at regulated speed via a ramp generator that can be configured by various settings, input speed adjustment, acceleration and deceleration ramp adjustment and other optional parameters.

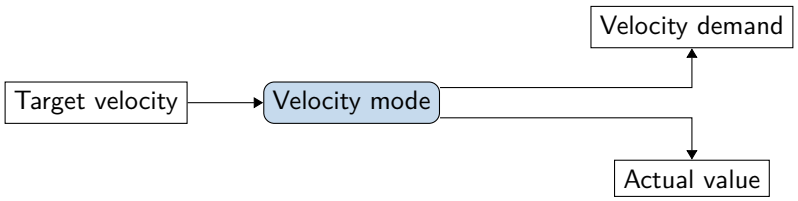
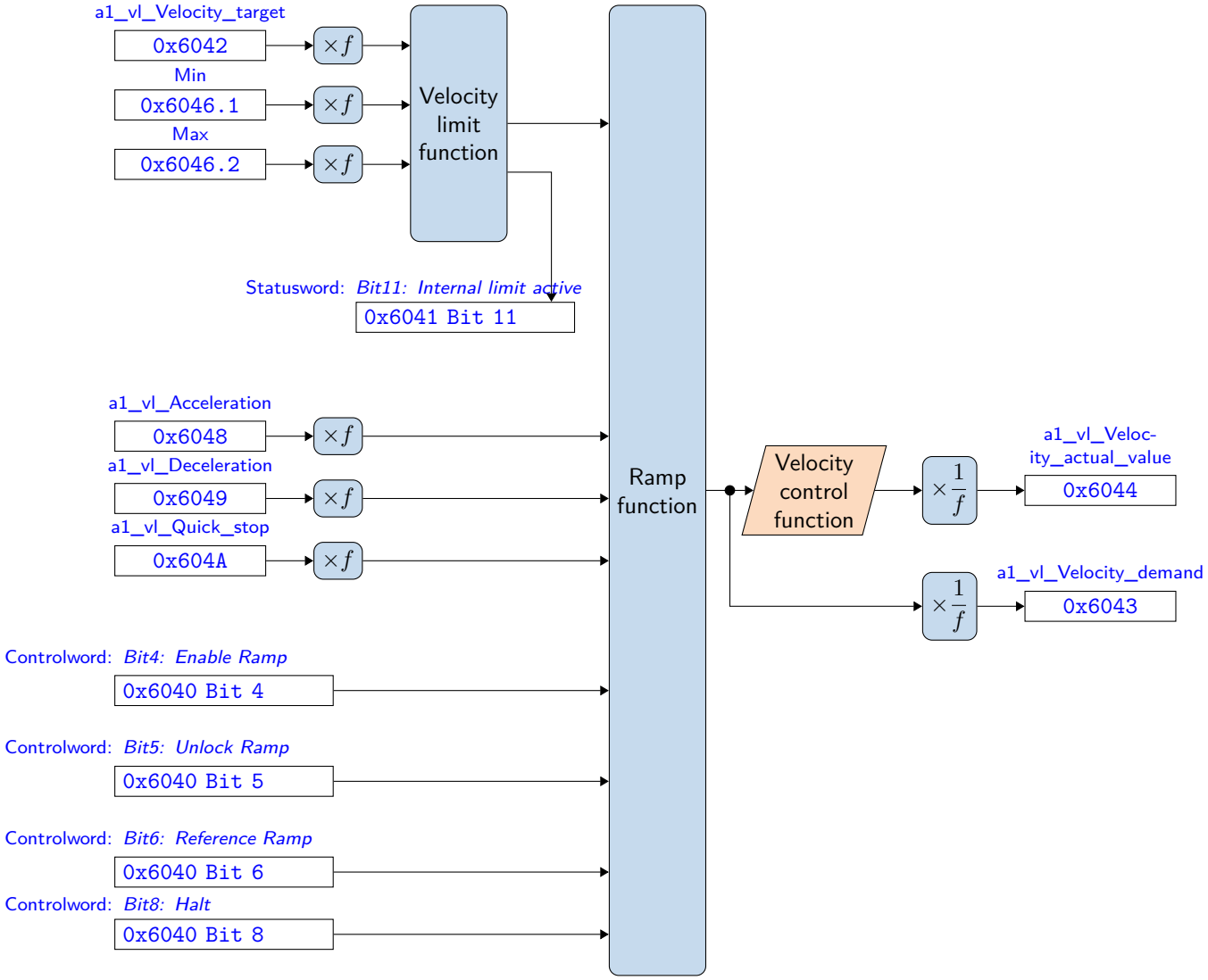


Figure 1.34: Velocity mode

1.3.2 Functional description

Velocity mode

Here is the functional diagram with all the parameters.



$$f = \frac{\texttt{0x604B.1}}{\texttt{0x604B.2}} * \frac{\texttt{0x604C.1}}{\texttt{0x604C.2}}$$

Figure 1.35: functional diagram of Velocity mode with all objects

Ramp generator

The Velocity profile is used to limit the increase or decrease the speed:

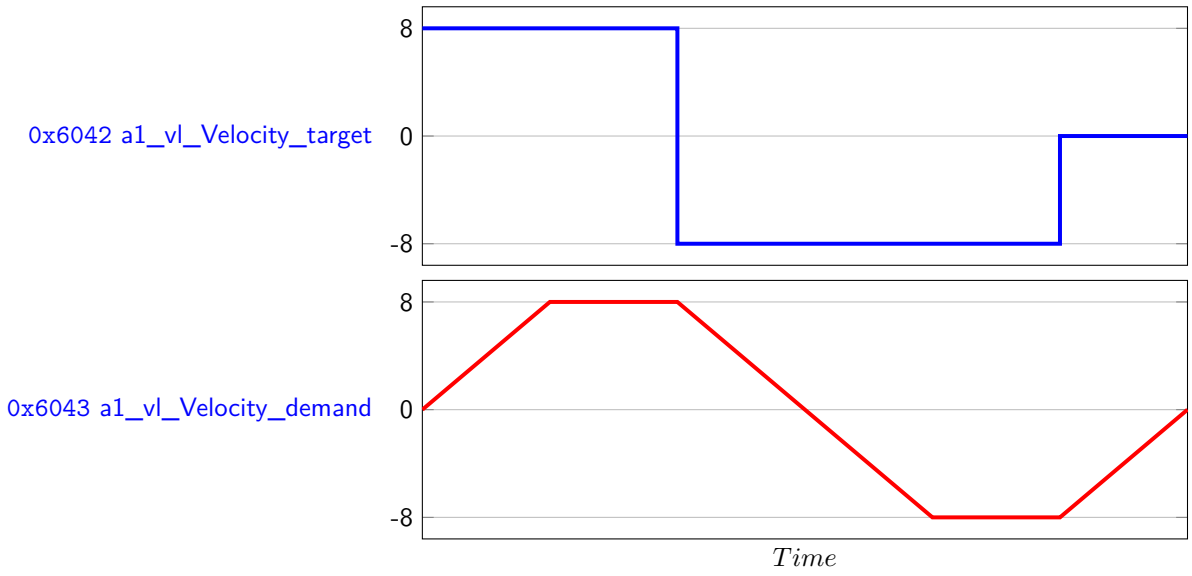


Figure 1.36: Velocity profile

Factor function

This function transforms a user units to internal units.

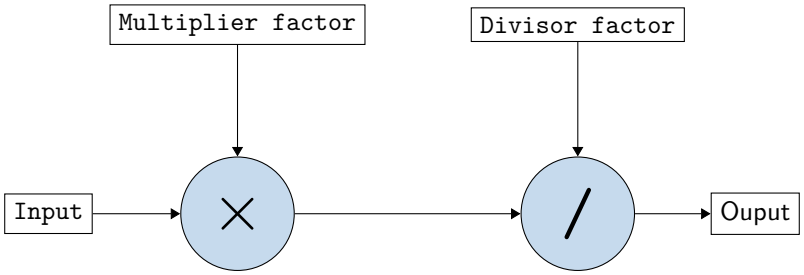


Figure 1.37: Factor function

This function transforms a internal units to user units.

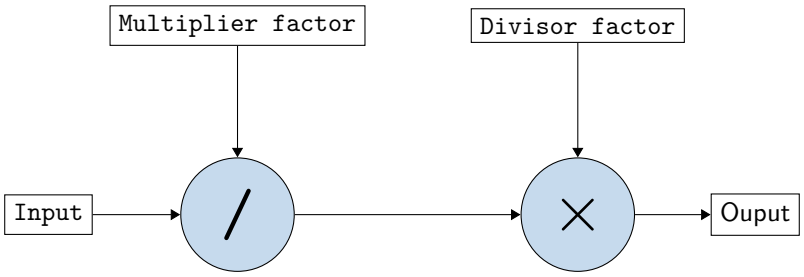


Figure 1.38: Reverse factor function

Mode activation

The 0x6060 a1_Modes_of_operation object enables speed mode by setting the value to 2.

1.3.3 Definition of parameters

All the objects described below correspond to axis 1, see Multi axes for ohter axes.

Input objects

Name	Description
0x6060 a1_Modes_of_operation	Modes of operation
0x6040 a1_Controlword	Control
0x6042 a1_vl_Velocity_target	Requested velocity
0x6046 a1_vl_Min_Max	Maximal and minimal velocity
0x6048 a1_vl_Acceleration	Maximal cruising acceleration
0x6049 a1_vl_Deceleration	Maximal cruising deceleration
0x604A a1_vl_Quick_stop	Maximal cruising deceleration of Quick Stop

Table 1.18: Configuration objects

Output objects

Name	Description
0x6041 a1_Statusword	Status
0x6043 a1_vl_Velocity_demand	Velocity generated by the generator trajectory
0x6044 a1_vl_Velocity_actual_value	Actual velocity value

Table 1.19: Status objects

0x6040 a1_Controlword

The bit 4, bit 5, bit 6, and bit 8 of 0x6040 a1_Controlword are used in Velocity mode.

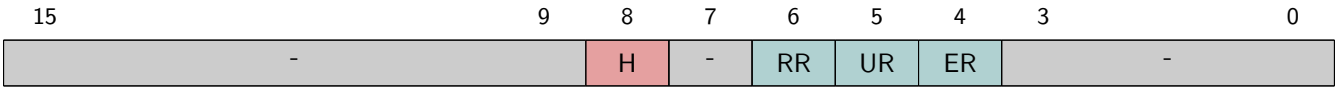


Figure 1.39: Frame of Controlword for velocity mode

Bit	Name	Value	Definition
4	**ER**: Enable ramp	0	Disable ramp
		1	Enable ramp
5	**UR**: Unlock ramp	0	Ramp output value is locked
		1	Ramp output value = Ramp input value
6	**RR**: Reference ramp	0	Ramp input set to zero
		1	Ramp input used ramp reference
8	**H**: Halt	0	No command
		1	Axis stopped accordingly to halt option code (see 0x605D a1_Halt_option

Figure 1.40: Definition of Controlword bits (bit 4, bit 5, bit 6, and bit 8)

Halt	Enable ramp	Unlock ramp	Reference ramp	Priority
0	0	X	X	Disable ramp
0	1	0	X	lock ramp
0	1	1	0	ramp set to zero with normal ramp
0	1	1	1	Normal ramp Reference
1	X	X	X	ramp set to zero with: see Halt option 0x605D

Figure 1.41: Priority of Controlword bits (Halt/Enable/Unlock/Reference)

0x6041 a1_Statusword

The bit 11 of 0x6041 a1_Statusword is used in velocity mode.

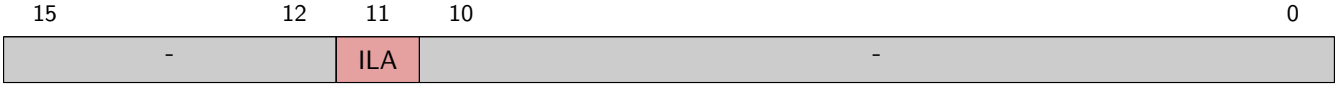


Figure 1.42: Frame of Statusword for velocity mode

Bit	Name	Value	Definition
11	**ILA**: Internal limit active	0	No internal limit
		1	Internal limit is active

Figure 1.43: Definition of Statusword bits (bit 11)

0x6042 a1_vl_Velocity_target

This object defines the target speed of the system. Positive values mean forward direction and negative values mean reverse direction.

The object 0x604B a1_vl_Set_point_factor and the 0x604C a1_vl_Dimension_factor allow to modify the units or the resolution.

Index	SubIndex	Name		
0x6042	0	a1_vl_Velocity_target		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW,RPDO	0	-	[-0x7FFF;0x7FFF]

Figure 1.44: Object description 0x6042.0 *a1_vl_Velocity_target*

0x6043 a1_vl_Velocity_demand

This object provides the instantaneous velocity generated by the ramp function. Positive values mean forward direction and negative values mean reverse direction.

Index	SubIndex	Name		
0x6043	0	a1_vl_Velocity_demand		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	-	-	[-0x7FFF;0x7FFF]

Figure 1.45: Object description 0x6043.0 *a1_vl_Velocity_demand*

0x6044 a1_vl_Velocity_actual_value

This object provide the velocity at the motor spindle. Positive values mean forward direction and negative values mean reverse direction.

Index	SubIndex	Name		
0x6044	0	a1_vl_Velocity_actual_value		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	-	-	[-0x7FFF;0x7FFF]

Figure 1.46: Object description 0x6044.0 *a1_vl_Velocity_actual_value*

0x6046 a1_vl_Min_Max

This object defines the minimum and maximum velocity. The sub-object 0x6046.1 *Min* indicates a negative or positive minimum velocity. The sub-object 0x6046.2 *Max* indicates a negative or positive maximum velocity.

The object 0x604B a1_vl_Set_point_factor and the 0x604C a1_vl_Dimension_factor allow to modify the units or the resolution.

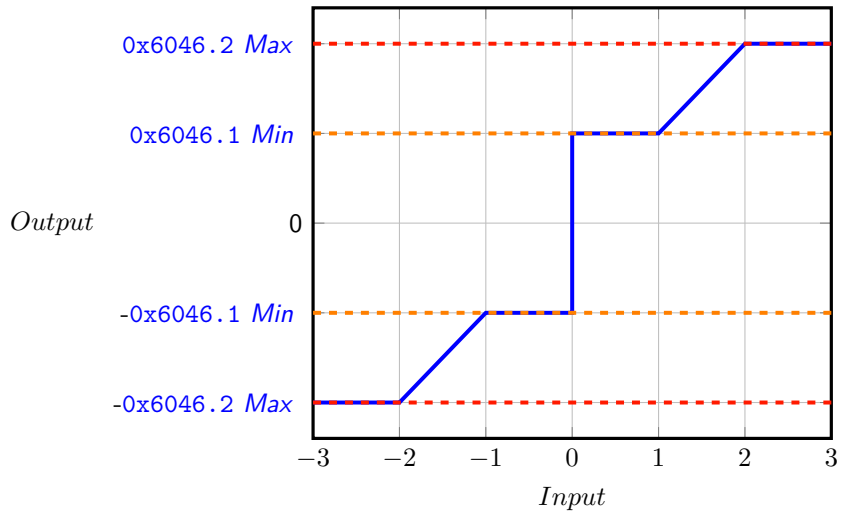


Figure 1.47: Characteristic of minimum and maximum of velocity

Note: Special case when going from "Switched On" state to "Operation Enabled" state: if the speed is equal to 0 when passing, then the demand remains at zero.

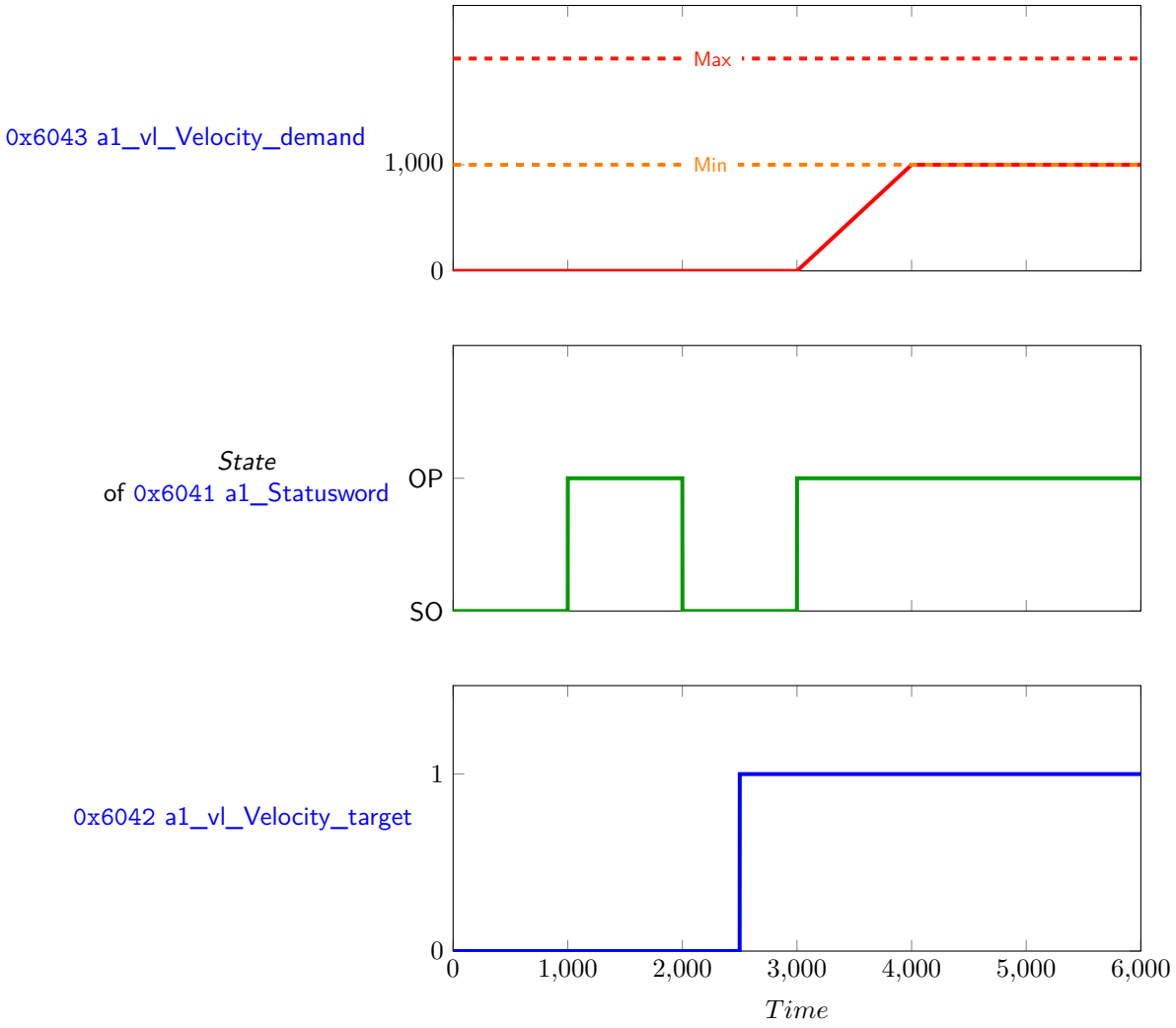


Figure 1.48: Special case when going from "Switched On(SO)" state to "Operation Enabled(OP)" state

`0x6046.1` ***Min***

Index	SubIndex	Name		
0x6046	1	Min		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW,RPDO	0	-	[0;0xFFFFFFFF]

Figure 1.49: Object description 0x6046.1 *Min*

0x6046.2 *Max*

Index	SubIndex	Name		
0x6046	2	Max		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW,RPDO	100	-	[0;0xFFFFFFFF]

Figure 1.50: Object description 0x6046.2 *Max*

0x6048 a1_vl_Acceleration

This object defines the acceleration ramp, configurable with delta speed and delta time, as shown by the following equation and graphic:

$$Acceleration = \frac{\text{0x6048.1}Delta\_speed}{\text{0x6048.2}Delta\_time} \tag{1.1}$$

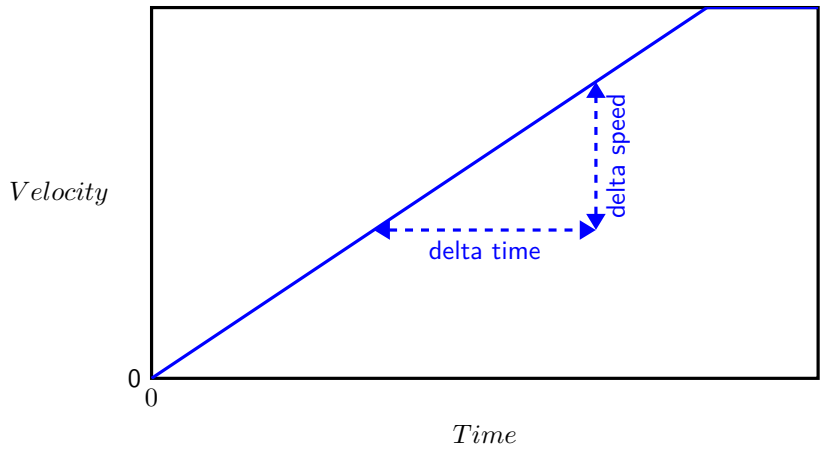


Figure 1.51: Feature of acceleration ramp

The object 0x604B a1_vl_Set_point_factor and the 0x604C a1_vl_Dimension_factor allow to modify the units or the resolution.
The value of delta speed cannot be equal to zero. The value of delta time shall be given in s and cannot be equal to zero.

0x6048.1 *Delta_speed*

Index	SubIndex	Name		
0x6048	1	Delta_speed		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW,RPDO	1	-	[0x1;0xFFFFFFFF]

Figure 1.52: Object description 0x6048.1 *Delta_speed*

0x6048.2 *Delta_time*

Index	SubIndex	Name		
0x6048	2	Delta_time		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT16	RW,RPDO	1	-	[0x1;0xFFFF]

Figure 1.53: Object description 0x6048.2 *Delta_time*

0x6049 a1_vl_Deceleration

This object defines the deceleration ramp, configurable with delta speed and delta time, as shown by the following equation and graphic:

$$Deceleration = \frac{\texttt{0x6049.1}Delta\_speed}{\texttt{0x6049.2}Delta\_time} \tag{1.2}$$

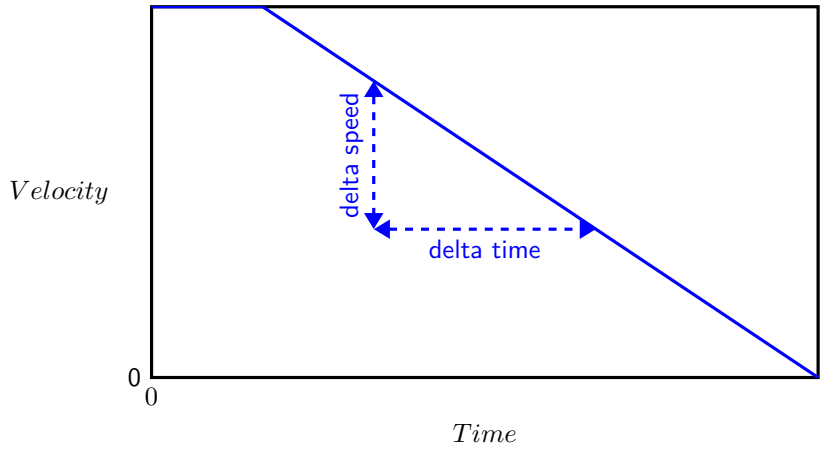


Figure 1.54: Feature of deceleration ramp

The object 0x604B a1_vl_Set_point_factor and the 0x604C a1_vl_Dimension_factor allow to modify the units or the resolution.
The value of delta speed cannot be equal to zero. The value of delta time shall be given in s and cannot be equal to zero.
Note: If delta speed is equal to 0 or if the object is not implemented, the values of 0x6048 a1_vl_Acceleration will be used

0x6049.1 *Delta_speed*

Index	SubIndex	Name		
0x6049	1	Delta_speed		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW,RPDO	1	-	[0x1;0xFFFFFFFF]

Figure 1.55: Object description 0x6049.1 *Delta_speed*

0x6049.2 *Delta_time*

Index	SubIndex	Name		
0x6049	2	Delta_time		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT16	RW,RPDO	1	-	[0x1;0xFFFF]

Figure 1.56: Object description 0x6049.2 *Delta_time*

0x604A a1_vl_Quick_stop

This object defines the deceleration ramp for a quick stop, configurable with delta speed and delta time, as shown by the following equation and graphic:

$$Deceleration = \frac{\texttt{0x604A.1}Delta\_speed}{\texttt{0x604A.2}Delta\_time} \tag{1.3}$$

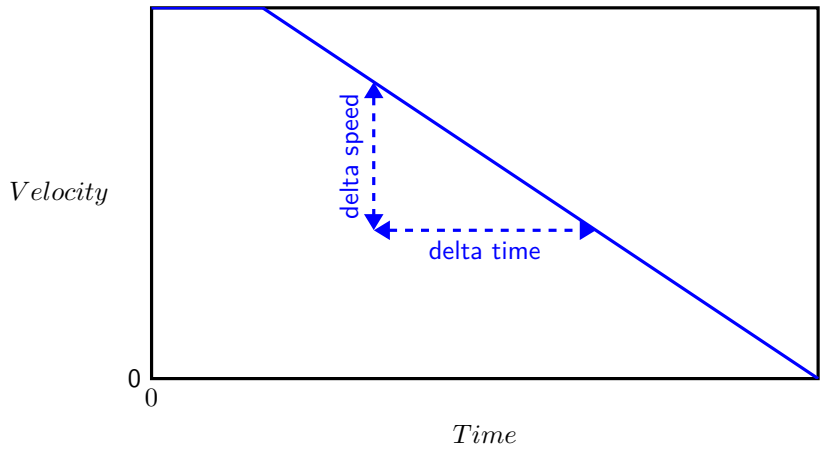


Figure 1.57: Feature of deceleration ramp for quick stop

The object 0x604B a1_vl_Set_point_factor and the 0x604C a1_vl_Dimension_factor allow to modify the units or the resolution.
The value of delta speed cannot be equal to zero. The value of delta time shall be given in s and cannot be equal to zero.
Note: If delta speed is equal to 0 or if the object is not implemented, the values of 0x6048 a1_vl_Acceleration will be used

0x604A.1 *Delta_speed*

Index	SubIndex	Name		
0x604A	1	Delta_speed		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW,RPDO	1	-	[0x1;0xFFFFFFFF]

Figure 1.58: Object description 0x604A.1 *Delta_speed*

0x604A.2 *Delta_time*

Index	SubIndex	Name		
0x604A	2	Delta_time		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT16	RW,RPDO	1	-	[0x1;0xFFFF]

Figure 1.59: Object description 0x604A.2 *Delta_time*

0x604B a1_vl_Set_point_factor

The resolution or directing range of the velocity setpoint can be modified with numerator and denominator of this object. This factor has no influence in the limit function and the ramp function.

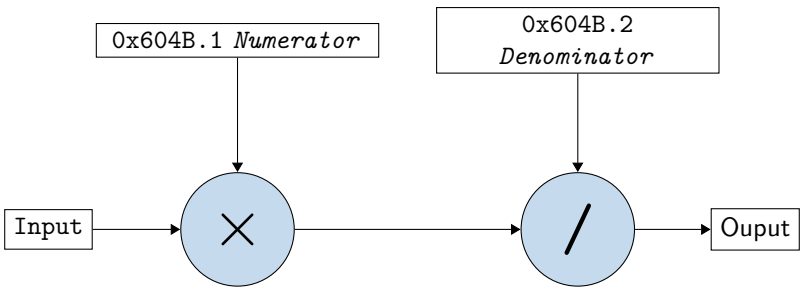


Figure 1.60: Factor function

0x604B.1 *Numerator*

Index	SubIndex	Name		
0x604B	1	Numerator		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW,RPDO	1	-	[0x1;0x7FFF]

Figure 1.61: Object description 0x604B.1 *Numerator*

0x604B.2 *Denominator*

Index	SubIndex	Name		
0x604B	2	Denominator		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW,RPDO	1	-	[0x1;0x7FFF]

Figure 1.62: Object description 0x604B.2 *Denominator*

0x604C a1_vl_Dimension_factor

This object serves to change unit or to scale specific units of the user. This object convert the specific velocity units to the LSB per millisecond or LSB per millisecond to specific velocity units.

$$Velocity_{internal}\,[LSB/ms] = Velocity\,[user\ units] \times \texttt{0x604C}\textit{a1\_vl\_Dimension\_factor}\,[LSB/ms/user\ units] \quad (1.4)$$

0x604C.1 *Numerator*

Index	SubIndex	Name		
0x604C	1	Numerator		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT32	RW,RPDO	1	-	[0x1;0x7FFF]

Figure 1.63: Object description 0x604C.1 *Numerator*

0x604C.2 *Denominator*

Index	SubIndex	Name		
0x604C	2	Denominator		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT32	RW,RPDO	1	-	[0x1;0x7FFF]

Figure 1.64: Object description 0x604C.2 *Denominator*

1.4 Profile Position Mode

1.4.1 Introduction

Profile allows movement at position, applying a target position to the generator trajectory. It generates a request position for the position control loop.

The generator of trajectory take into account limit values of speed, acceleration, deceleration during movement.

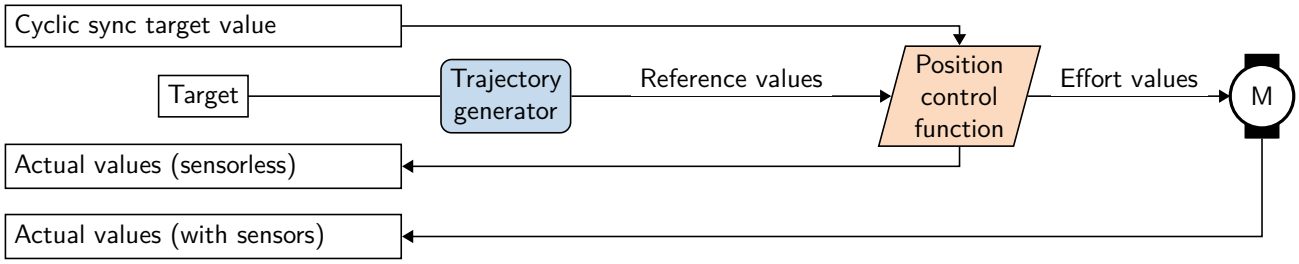


Figure 1.65: Profile Position Mode

1.4.2 Functional description

The setting of set-points is controlled by 0x6040 a1_Controlword: *New set-point, Change set-immediately, Absolute or relative* value as well as status of 0x6041 a1_Statusword : *Set-point acknowledge* and *Target reached*

This profile position mode works with a single set point so a single set-point is expected by the device.

Profile position mode

The general structure of this mode is shown in the following diagram with all objects involved for trajectory generator.

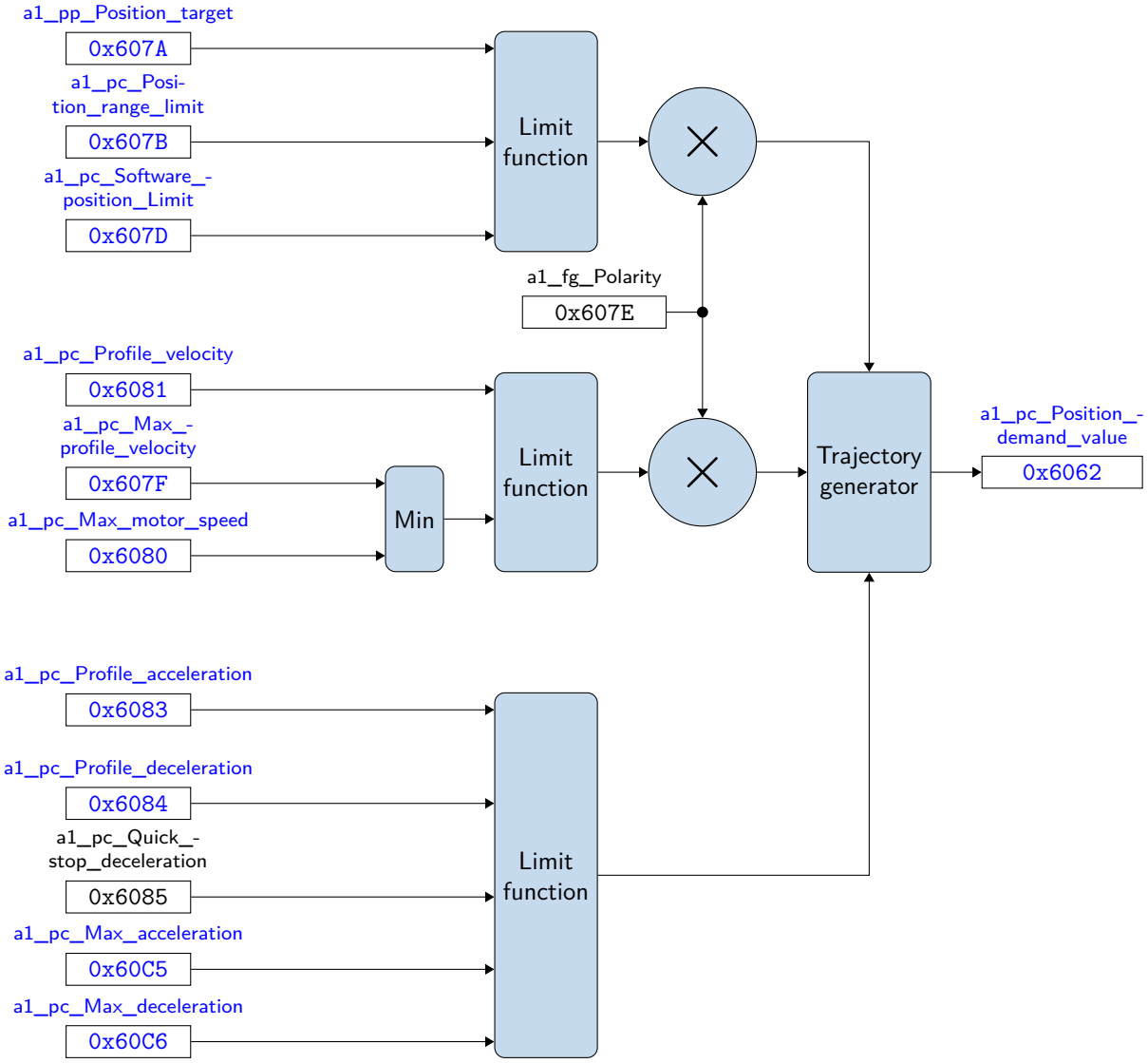


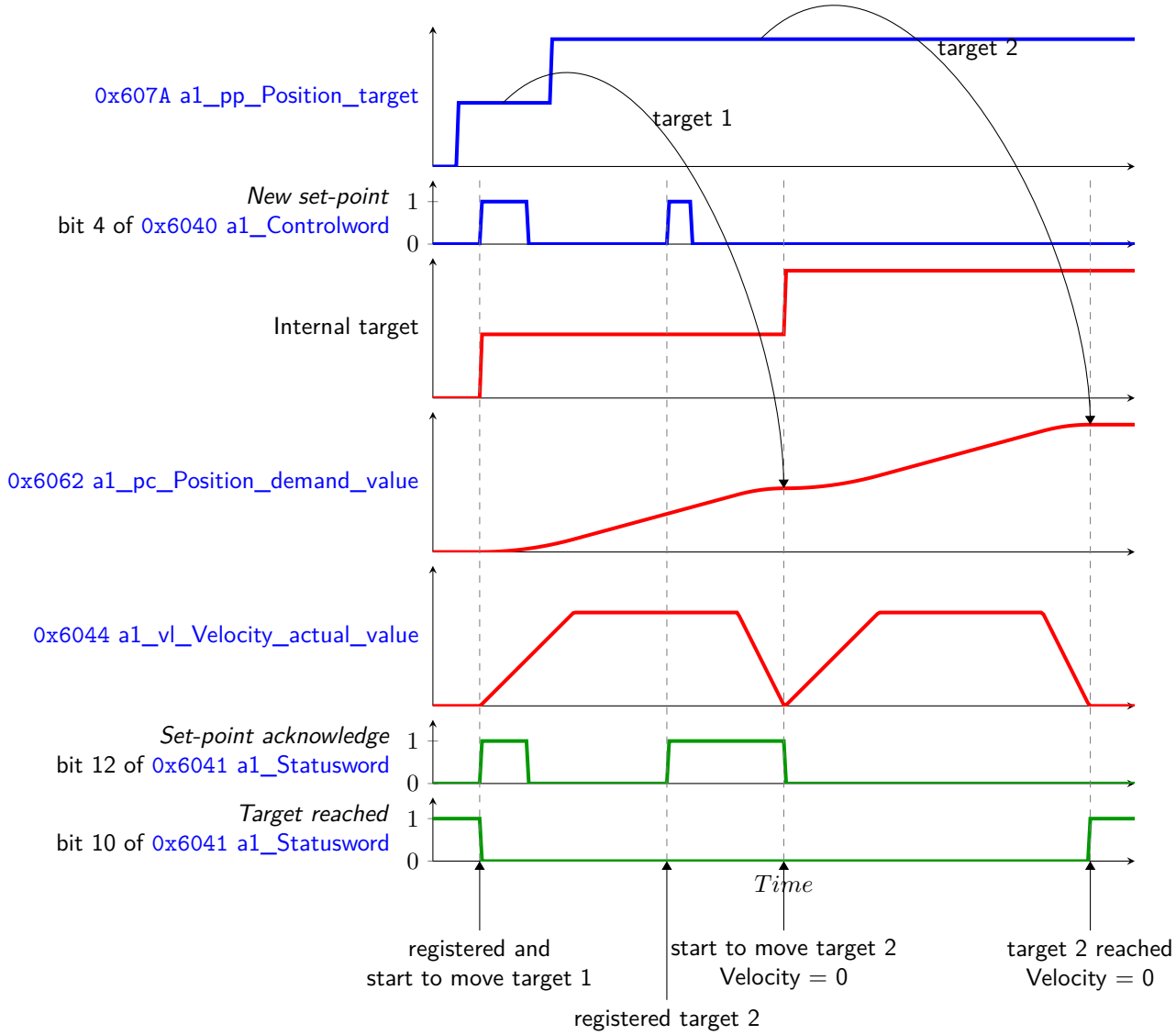
Figure 1.66: functional diagram of Profile Position Modewith all objects

Processing of a new setpoint

A new set-point receipted by device will apply on a rising edge of the *New set-point bit* of the 0x6040 a1_Controlword. *Change set-immediately and Absolute or relative* modify processing of a new set-point.

A new 0x6081 a1_pc_Profile_velocity, 0x6083 a1_pc_Profile_acceleration and 0x6084 a1_pc_Profile_deceleration as well as control their respective limits are taken into account when executing the new set-point in accordance with the two methods described below.

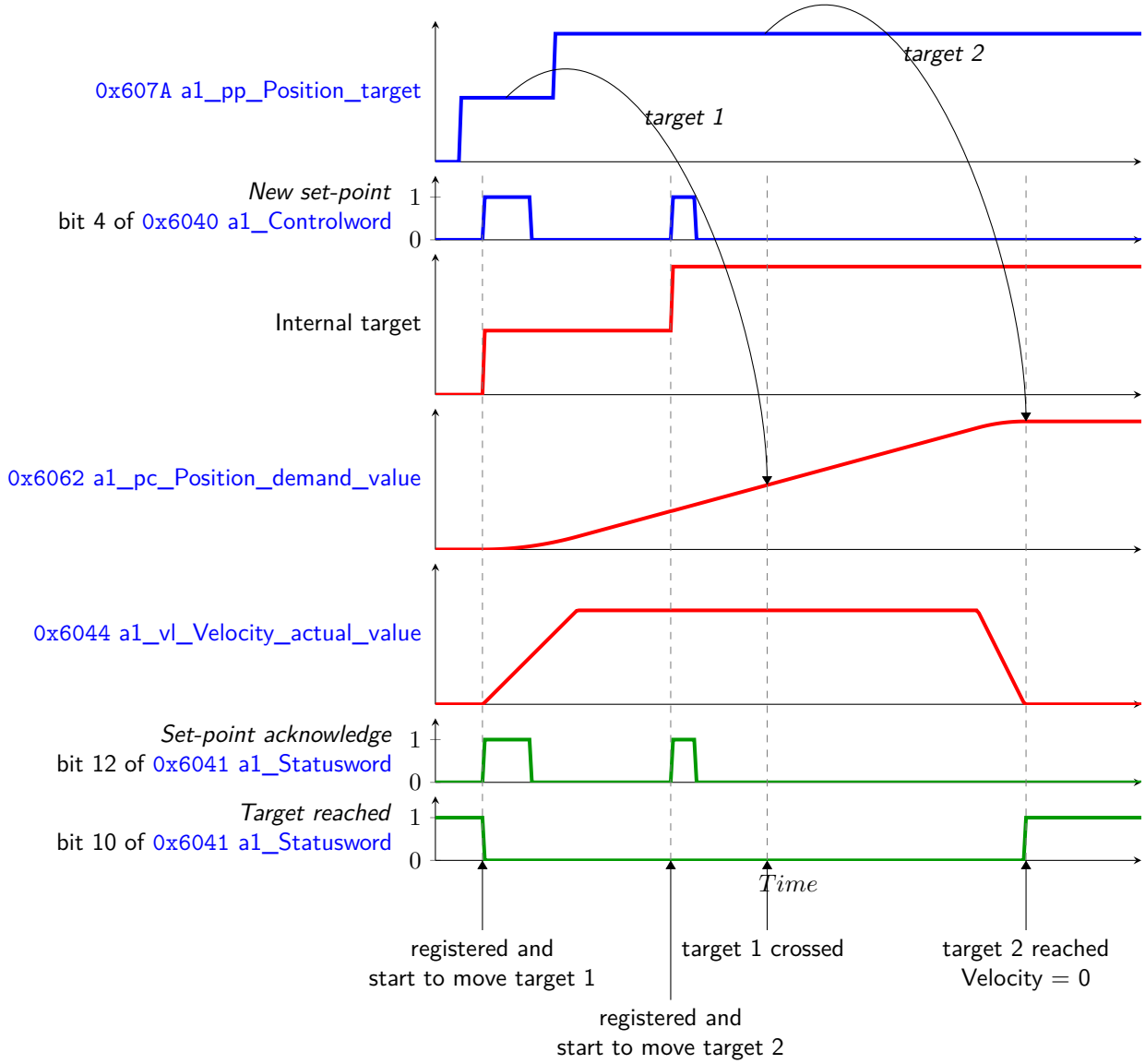
- Two methods are available Set of set-points and Single set-point according to *Change set immediately bit*, when a set-point is being executed and the *New set-point bit* is activated
  - **Set of set-points**: *Change set immediately bit* = 0.
    the new set-point will only be processed after the previous has been reached and set the *Set-point acknowledgment bit* in the 0x6041 a1_Statusword to 1 until the set-point previous has been reached.



All graphics are generated from real device.

Figure 1.67: Set of set-points method

– **Single set-point**: *Change set immediately bit* = 1:
The new set-point will be processed immediately and set the *Set-point acknowledgment bit* in the 0x6041 a1_Statusword to 1 until the *New set-point bit* is set to 0.



All graphics are generated from real device.

Figure 1.68: Single set-point method

- *Absolute or relative* bit of 0x6040 a1_Controlword define if the *New set-point bit* is:
  - absolute value, position reached = 0x607A a1_pp_Position_target
  - relative value, position reached = 0x607A a1_pp_Position_target + 0x6062 a1_pc_Position_demand_value.

Note: arg1

Trajectory generator

The trajectory generator uses the motion profile: Linear ramp (trapezoidal profile). Below, motion profiles of acceleration, velocity and position:

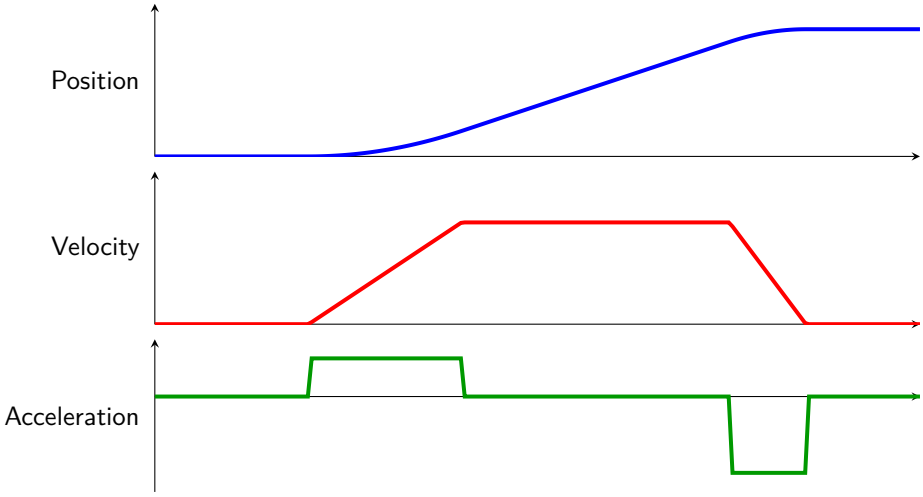


Figure 1.69: Linear ramp (trapezoidal profile)

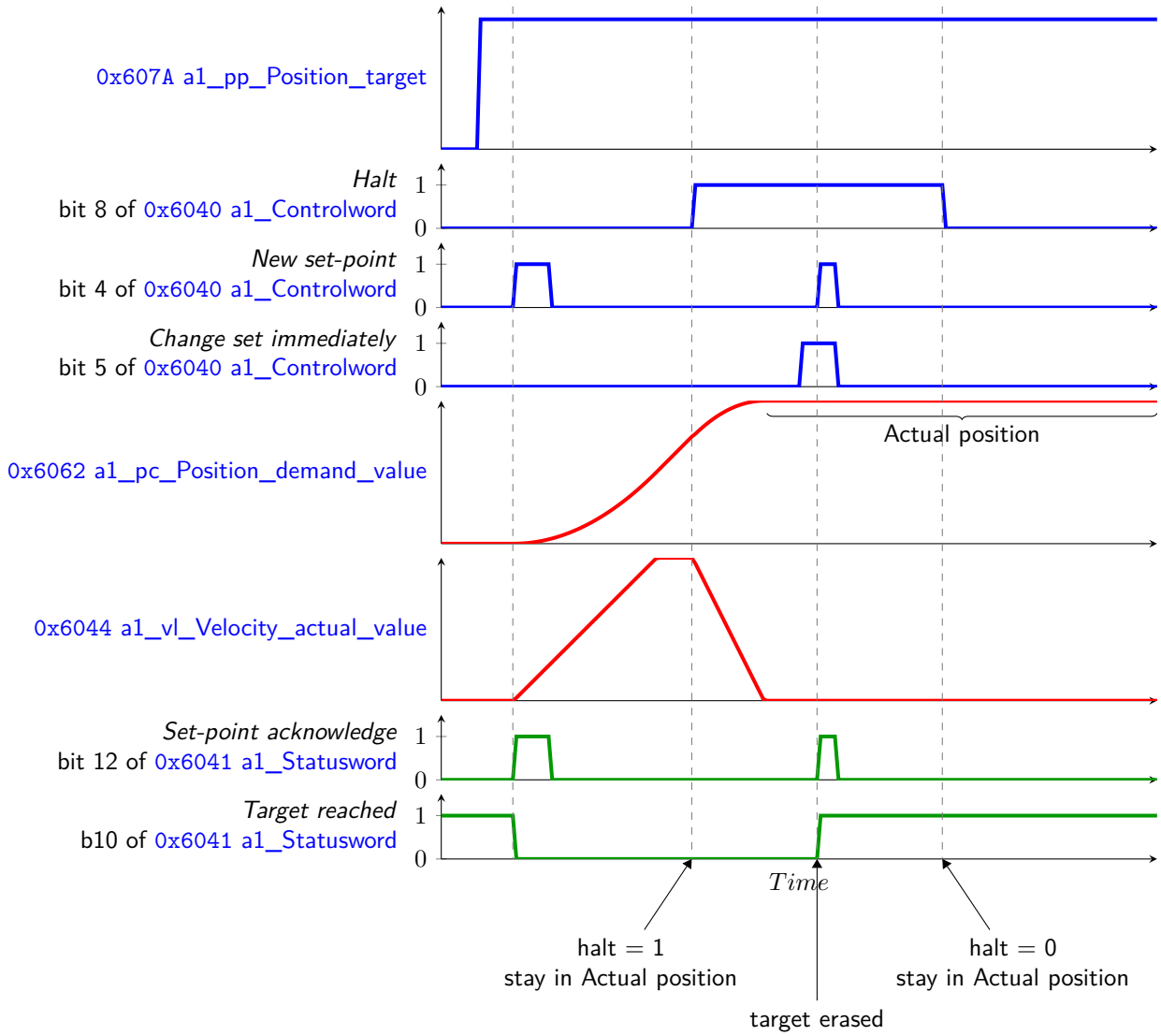
Mode activation

The 0x6060 a1_Modes_of_operation object enables speed mode by setting the value to 1.

Specific usage

Halt bit of Controlword The stop bit is used to stop movement by decelerating the axis, so that the speed approaches zero. The behavior of halt is in agreement to 0x605D a1_Halt_option. Two specific uses are described below :

- **Halt bit in conjunction with new set-point bit:**
  A new set point can be defined by activating *new set-point bit* and setting new set-point during halt actived. After releasing the stop bit, actual set-point processing continues and continues with the new set-point set during shutdown.
- **Erase processed set-point during Halt:**
  This procedure allows to erase processed set-point and set the actual position as a new set-point. After releasing the *halt bit* the position stay to actual position.



All graphics are generated from real device.

Figure 1.70: Erase processed set-point

Inter state transition The behavior then transition of state Operation Enabledto another state is to stopping the movement by decelerating the axis, so that the speed approaches zero. In agreement to Transition options.

Inter mode transition These transition are possible from the Operation Enabledstate. The behavior is to stop the movement by decelerating the axis, so that the speed approaches zero. The behavior is in agreement to 0x605D a1_Halt_option. Once the velocity is equal to zero, the changement of mode is realized.

1.4.3 Definition of parameters

All the objects described below correspond to axis 1, see Multi axes for ohter axes.

Input objects

Name	Description
0x6060 a1_Modes_of_operation	Modes of operation
0x6040 a1_Controlword	Control
0x607A a1_pp_Position_target	Requested target position
0x607B a1_pc_Position_range_limit	Maximal and minimal position range limits
0x607D a1_pc_Software_position_Limit	Maximal and minimal software position limits
0x607F a1_pc_Max_profile_velocity	Maximal speed limit
0x6080 a1_pc_Max_motor_speed	Maximal speed limit for the motor
0x6081 a1_pc_Profile_velocity	Maximal cruising speed
0x6083 a1_pc_Profile_acceleration	Maximal cruising acceleration
0x6084 a1_pc_Profile_deceleration	Maximal cruising deceleration
0x60C5 a1_pc_Max_acceleration	Maximal acceleration
0x60C6 a1_pc_Max_deceleration	Maximal deceleration

Table 1.20: Configuration objects

Output objects

Name	Description
0x6041 a1_Statusword	Status
0x6062 a1_pc_Position_demand_value	Position generated by the generator trajectory
0x6064 a1_pc_Position_actual_value	Actual position value

Table 1.21: Status objects

0x6040 a1_Controlword

The bit 4, bit 5, bit 6, and bit 8 of 0x6040 a1_Controlword are used in Profile Position Mode.

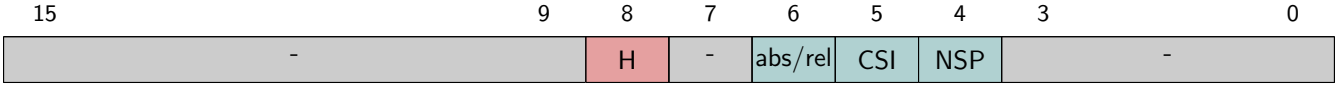


Table 1.22: Frame of Controlword for Profile Position Mode

Bit	Name	Value	Definition
4	**NSP**: New set-point	0	Target not applied
		0 → 1	Target applied
5	**CSI**: Change set immediately	0	Wait until the actual target position is complete before applying the new position.
		1	New target position is applied to immediately.
6	**abs/rel**: absolute/relative	0	Target position is in absolute value
		1	Target position is in relative value
8	**H**: Halt	0	No command
		1	Axis stopped accordingly to halt option code (see 0x605D a1_Halt_option)

Figure 1.71: Definition of Controlword bits (bit 4, bit 5, bit 6, and bit 8)

0x6041 a1_Statusword

The bit 10, 12, 13 of 0x6041 a1_Statusword is used in Profile Position Mode.

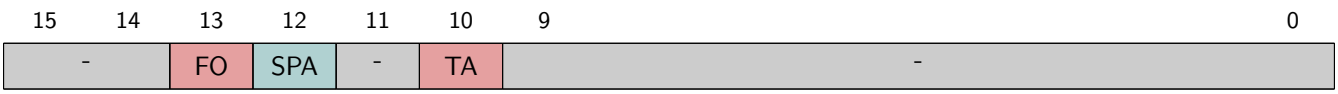


Figure 1.72: Frame of Statusword for velocity mode

Bit	Name	Value	Definition
10	**TA**: Target reached	0	Halt = 0 : Target position not reached Halt = 1 : Axis decelerates
		1	Halt = 0 : Target position reached Halt = 1 : Velocity of axis is 0
12	**SPA**: Set-point acknowledge	0	Previous target already processed, waiting for a new target
		1	Target in process
13	**FO**: Following error	0	No following error
		1	Following error

Figure 1.73: Definition of Statusword bits (bit 11)

0x607A a1_pp_Position_target

Target position requested from the position generator to which it must go using the current settings of motion control parameters. This value is given in user-defined position units.

Index	SubIndex	Name		
0x607A	0	a1_pp_Position_target		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT32	RW,RPDO	0	-	[;]

Figure 1.74: Object description 0x607A.0 *a1_pp_Position_target*

0x6062 a1_pc_Position_demand_value

Position generated by the generator trajectory and used at the position control input. This value is given in user-defined position units.

Index	SubIndex	Name		
0x6062	0	a1_pc_Position_demand_value		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT32	RO,TPDO	-	-	[;]

Figure 1.75: Object description 0x6062.0 *a1_pc_Position_demand_value*

0x6064 a1_pc_Position_actual_value

Actual position value of measurement device. This value is given in user-defined position units.

Index	SubIndex	Name		
0x6064	0	a1_pc_Position_actual_value		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT32	RO,TPDO	-	-	[;]

Figure 1.76: Object description 0x6064.0 *a1_pc_Position_actual_value*

0x607B a1_pc_Position_range_limit

Defines maximal and minimal position range limits, allow limit the numerical range of the target value. The value wrap automatically to the other end of the range when target value reaching or exceeding these limits. For prevented overflow of the input value, set software position limits.To disable the position range limits, the min and max be set to 0. These values are given in user-defined position units.

0x607B.1 *Min*

Index	SubIndex	Name		
0x607B	1	Min		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT32	RW,RPDO	-262143	-	[;]

Figure 1.77: Object description 0x607B.1 *Min*

`0x607B.2` ***Max***

Index	SubIndex	Name		
0x607B	2	Max		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT32	RW,RPDO	262144	-	[;]

Figure 1.78: Object description `0x607B.2` *Max*

`0x607D` **a1_pc_Software_position_Limit**

Defines maximal and minimal software position limits, defines the absolute position limits for the position demand value and the position actual value. To disable the software position limits, the min and max be set to 0. These values are given in user-defined position units.

`0x607D.1` ***Min***

Index	SubIndex	Name		
0x607D	1	Min		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT32	RW,RPDO	-262143	-	[;]

Figure 1.79: Object description `0x607D.1` *Min*

`0x607D.2` ***Max***

Index	SubIndex	Name		
0x607D	2	Max		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT32	RW,RPDO	262144	-	[;]

Figure 1.80: Object description `0x607D.2` *Max*

`0x607F` **a1_pc_Max_profile_velocity**

Defines the maximum speed limit. This value is given in user-defined units.

Index	SubIndex	Name		
0x607F	0	a1_pc_Max_profile_velocity		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW,RPDO	1	-	[;]

Figure 1.81: Object description `0x607F.0` *a1_pc_Max_profile_velocity*

`0x6080` **a1_pc_Max_motor_speed**

Defines the maximum speed limit for the motor. This value is given in user-defined units.

Index	SubIndex	Name		
0x6080	0	a1_pc_Max_motor_speed		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW,RPDO	1	-	[;]

Figure 1.82: Object description 0x6080.0 *a1_pc_Max_motor_speed*

`0x6081` **a1_pc_Profile_velocity**

Defines the maximum cruising speed. This value is given in user-defined units.

Index	SubIndex	Name		
0x6081	0	a1_pc_Profile_velocity		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW,RPDO	1	-	[;]

Figure 1.83: Object description 0x6081.0 *a1_pc_Profile_velocity*

0x6083 a1_pc_Profile_acceleration

Defines the maximum cruising acceleration. This value is given in user-defined units.

Index	SubIndex	Name		
0x6083	0	a1_pc_Profile_acceleration		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW,RPDO	1	-	[0;a1_pc_Profile_deceleration]

Figure 1.84: Object description 0x6083.0 *a1_pc_Profile_acceleration*

0x6084 a1_pc_Profile_deceleration

Defines the maximum cruising deceleration. This value is given in user-defined units.

Index	SubIndex	Name		
0x6084	0	a1_pc_Profile_deceleration		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW,RPDO	1	-	[;]

Figure 1.85: Object description 0x6084.0 *a1_pc_Profile_deceleration*

0x6086 a1_pp_Motion_profile_type

Defines the type of ramp of the motion profile used.

Index	SubIndex	Name		
0x6086	0	a1_pp_Motion_profile_type		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW,RPDO	0	-	[;]

Figure 1.86: Object description 0x6086.0 *a1_pp_Motion_profile_type*

Value	Description
0	Trapeziodal profile

Figure 1.87: Option of 0x6086 *a1_pp_Motion_profile_type*

0x60C5 a1_pc_Max_acceleration

Defines the maximum acceleration. This value is given in user-defined units.

Index	SubIndex	Name		
0x60C5	0	a1_pc_Max_acceleration		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW,RPDO	1	-	[;]

Figure 1.88: Object description 0x60C5.0 *a1_pc_Max_acceleration*

0x60C6 a1_pc_Max_deceleration

Defines the maximum deceleration. This value is given in user-defined units.

Index	SubIndex	Name		
0x60C6	0	a1_pc_Max_deceleration		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW,RPDO	1	-	[;]

Figure 1.89: Object description 0x60C6.0 *a1_pc_Max_deceleration*

1.5 Interpolated position mode

1.5.1 Introduction

This mode allows the axis to be moved according to several set-point positions over a defined period. For each period, the axis moves to the next memorized position by calculating intermediate points at an appropriate speed.

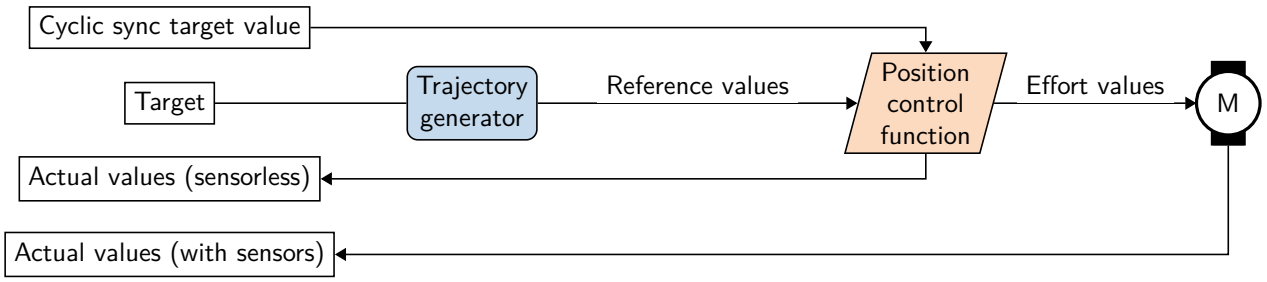


Figure 1.90: Interpolated position mode

1.5.2 Functional description

Here is the functional diagram with all the parameters.

Interpolated position mode

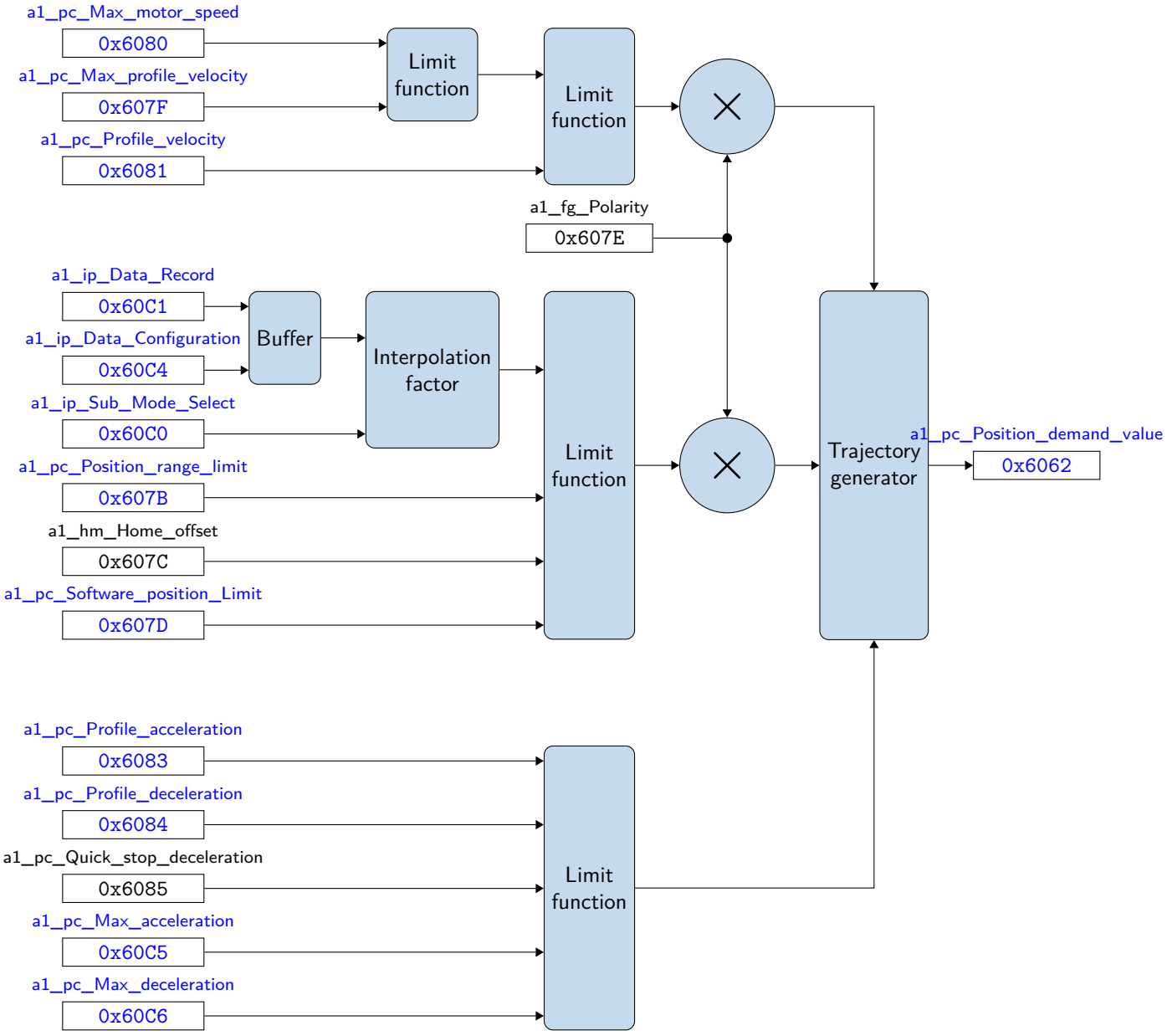


Figure 1.91: functional diagram of Interpolated position mode with all objects

State machine of Interpolated position mode

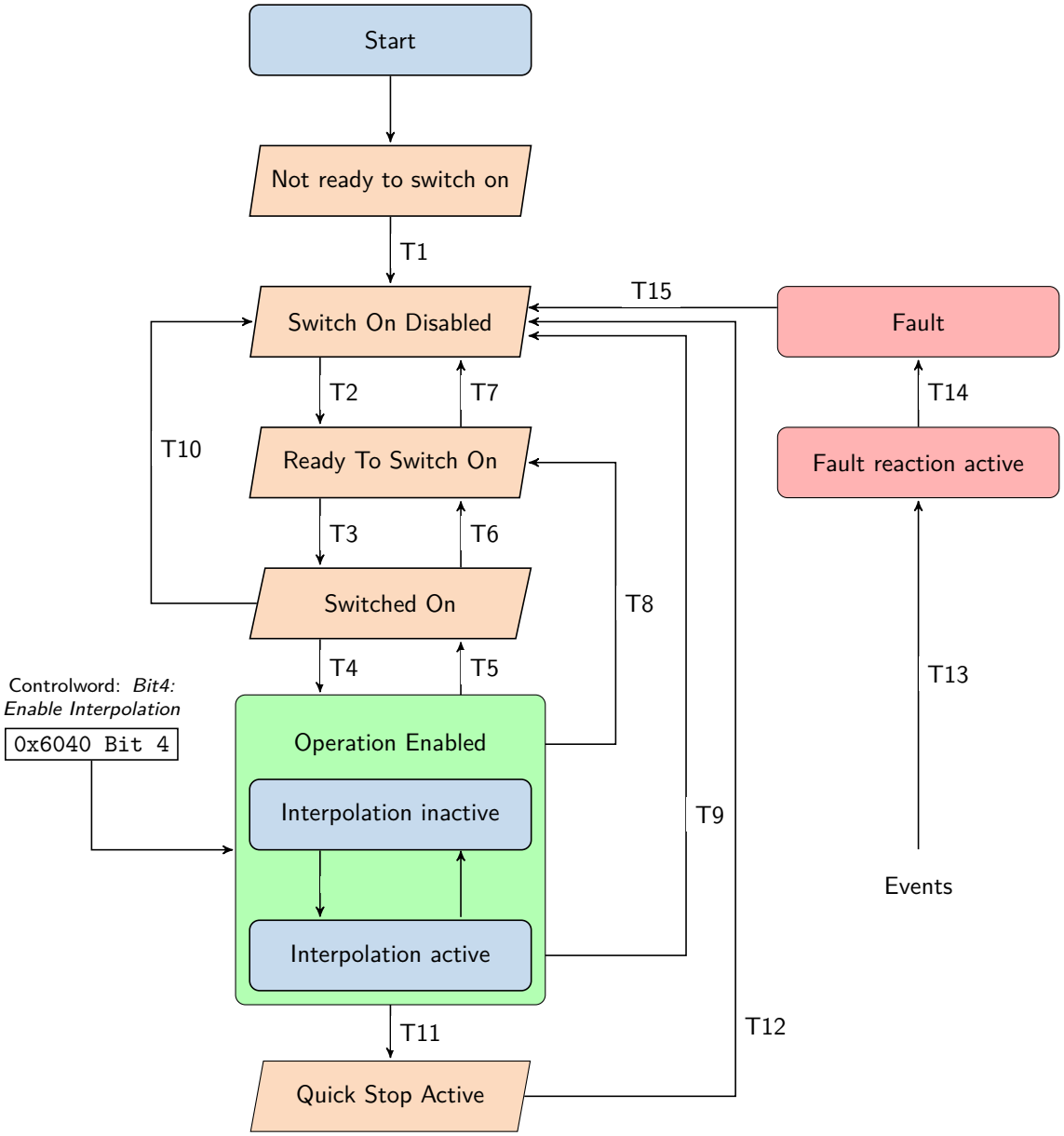


Figure 1.92: State machine of Interpolated position mode

State and transition of Interpolated position mode

Each state supports different functions described below:

State	Function
Interpolation inactive	The axis don't move. Input data is accepted. Buffer cleared on transition.
Interpolation active	The axis move. Input data is accepted.

Figure 1.93: Supported functions of state

The change of state is carried out by bit 4 of the object 0x6040 *a1_Controlword*.

Linear interpolated

The interpolated position mode allows to calculate the intermediate position points between two position set-points.

At each period defined by object 0x60C2 a1_ip_Time_Period, the mode go to next set-point. The mode calculates the velocity appropriate for reach the next position.

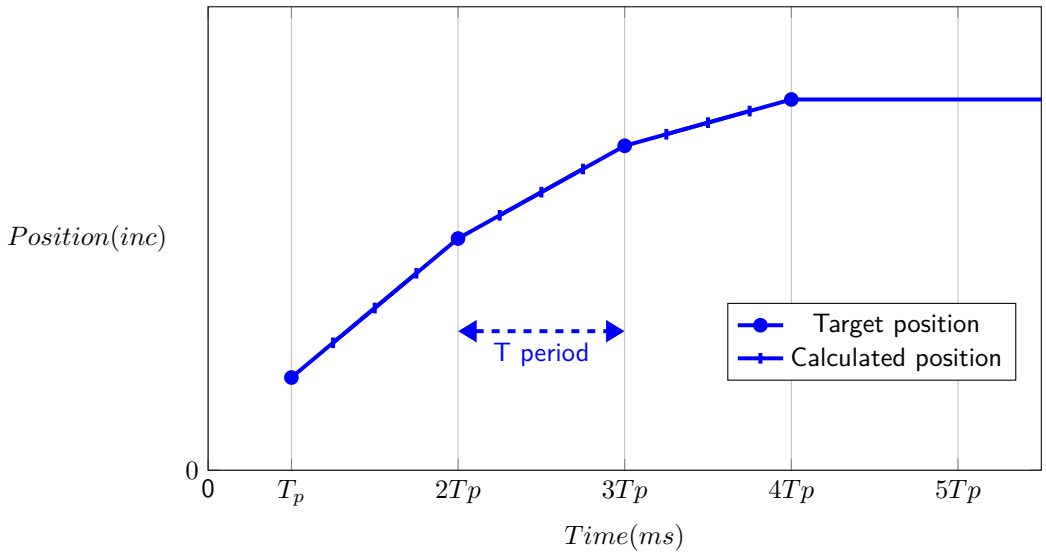


Figure 1.94: Linear interpolation

The axis continues its movement as long as there are position set-points in the buffer.

Buffer

The buffer is organized as FIFO (First In, First Out). When the device receive a new position, it place at the end of the queue and at each period, the device take the set-point from the top of the queue. The size of buffer is 32 items. When the buffer is empty, the device keep the latest set-point. The object 0x60C4 a1_ip_Data_Configuration allow to have informations on the buffer. When the buffer is full, the new positions are rejected.

Mode activation

The 0x6060 a1_Modes_of_operation object enables speed mode by setting the value to 7.

Specific usage

Halt command of Controlword The Halt command of Controlword allows the axis to be stopped at any time when bit 8 is set to 1. When bit 8 is to 0, the normal cycle is resumed by continuing the execution of the ramp.

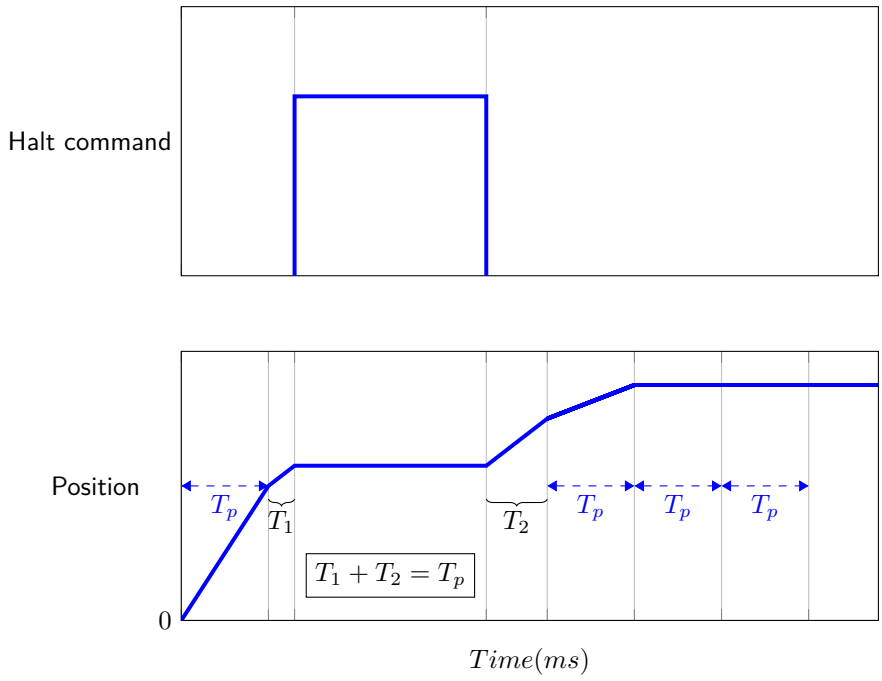


Figure 1.95: Halt Command Operation Graph

1.5.3 Definition of parameters

All the objects described below correspond to axis 1, see Multi axes for ohter axes.

Input objects

Name	Description
0x6060 a1_Modes_of_operation	Modes of operation
0x6040 a1_Controlword	Control
0x60C1 a1_ip_Data_Record	Requested target position
0x60C2 a1_ip_Time_Period	Configuration interpolation cycle time
0x60C4 a1_ip_Data_Configuration	Configuration buffer
0x60C0 a1_ip_Sub_Mode_Select	Type of interpolation
0x607B a1_pc_Position_range_limit	Maximal and minimal position range limits
0x607D a1_pc_Software_position_Limit	Maximal and minimal software position limits
0x607F a1_pc_Max_profile_velocity	Maximal speed limit
0x6080 a1_pc_Max_motor_speed	Maximal speed limit for the motor
0x6081 a1_pc_Profile_velocity	Maximal cruising speed
0x6083 a1_pc_Profile_acceleration	Maximal cruising acceleration
0x6084 a1_pc_Profile_deceleration	Maximal cruising deceleration
0x60C5 a1_pc_Max_acceleration	Maximal acceleration
0x60C6 a1_pc_Max_deceleration	Maximal deceleration

Table 1.23: Configuration objects

Output objects

Name	Description
0x6041 a1_Statusword	Status
0x6062 a1_pc_Position_demand_value	Position generated by the generator trajectory

Table 1.24: Status objects

0x6040 a1_Controlword

The bit 4 and bit 8 of 0x6040 a1_Controlword are used for Interpolated position mode.

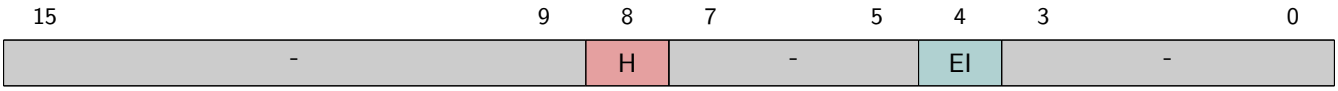


Figure 1.96: Frame of Controlword for Interpolated position mode

Bit	Name	Value	Definition
4	**EI**: Enable interpolation	0	Disable interpolation
		1	Enable interpolation
8	**H**: Halt	0	Execute instruction of bit 4
		1	Axis stopped accordingly to halt option code (see 0x605D a1_Halt_option)

Figure 1.97: Definition of Controlword bits (bit 4 and bit 8)

0x6041 a1_Statusword

The bit 11, bit 10, bit 12 and bit 13 of 0x6041 a1_Statusword are used for Interpolated position mode.

Figure 1.98: Frame of Statusword for velocity mode

Bit	Name	Value	Definition
10	**TR**: Target reached	0	Target position not (yet) reached
		1	Target position reached
11	**ILA**: Internal limit active	0	No internal limit
		1	Internal limit is active
12	**IMA**: Ip mode active	0	Interpolation inactive
		1	Interpolation active
13	**FE**: Following error	0	No following error
		1	Following error

Figure 1.99: Definition of Statusword bits (bit 10, bit 11, bit 12, bit 13)

0x60C0 a1_ip_Sub_Mode_Select

This object define the interpolation mode. The device only provides a linear interpolation algorithm.

Index	SubIndex	Name		
0x60C0	0	a1_ip_Sub_Mode_Select		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW,RPDO	0	-	[-32768;0]

Figure 1.100: Object description 0x60C0.0 *a1_ip_Sub_Mode_Select*

Value	Definition
0	Linear interpolation

Figure 1.101: Option of Interpolation sub mode

0x60C1 a1_ip_Data_Record

This object add a new set-point position to the interpolation algorithm. Only one interpolation data record can be sent to the device at a time. On reception, he interpolation data record is placed in a FIFO buffer (section 1.5.2), defining a new position setpoint.

As a reminder: The buffer can contain a maximum of 32 interpolation data record.

Warning: The position delta must not exceed 0x7FFFFFFF.

0x60C1.1 *Set-point*

Index	SubIndex	Name		
0x60C1	1	Set-point		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT32	RW,RPDO	0	-	[-0x7FFFFFFF;0x7FFFFFFF]

Figure 1.102: Object description 0x60C1.1 *Set-point*

0x60C2 a1_ip_Time_Period

This object defines interpolation cycle time or period in scientific notation in seconds. The 0x60C2.1 *Time_units* indicates the coefficient and the 0x60C2.2 *Time_index* indicates the exponent.
So the cycle time is given by:

$$Interpolation\,cycle\,time = unit * 10^{index} \tag{1.5}$$

0x60C2.1 *Time_units*

Index	SubIndex	Name		
0x60C2	1	Time_units		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT8	RW,RPDO	4	-	[0x00;0xFF]

Figure 1.103: Object description 0x60C2.1 *Time_units*

0x60C2.2 *Time_index*

Index	SubIndex	Name		
0x60C2	2	Time_index		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT8	RW,RPDO	-3	-	[-3;1]

Figure 1.104: Object description 0x60C2.2 *Time_index*

Note: The change of duration or cycle period will be taken into account on the next transition from Interpolation inactive state to Interpolation active state.

0x60C4 a1_ip_Data_Configuration

This object provide the maximum buffer size, the actual buffer length and a clear buffer command.

0x60C4.1 *Maximum_buffer_size* Give the maximum number of interpolated positions the buffer can store:

Index	SubIndex	Name		
0x60C4	1	Maximum_buffer_size		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RO	32	-	[0x00000000;0xFFFFFFFF]

Figure 1.105: Object description 0x60C4.1 *Maximum_buffer_size*

0x60C4.2 *Actual_buffer_size* Give the current number of interpolated positions in the buffer:

Index	SubIndex	Name		
0x60C4	2	Actual_buffer_size		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW,RPDO	0	-	[0x00000000;0xFFFFFFFF]

Figure 1.106: Object description 0x60C4.2 *Actual_buffer_size*

0x60C4.6 *Buffer_clear* Allow to clear the buffer and delete all recorded positions. This object reacts to a write event.

Index	SubIndex	Name		
0x60C4	6	Buffer_clear		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT8	RW,TPDO	0	-	[0x00;0x01]

Figure 1.107: Object description 0x60C4.6 *Buffer_clear*

Value	Definition
0	Clear buffer
!= 0	No clear

Figure 1.108: Option of Buffer clear

Chapter 2

Board configuration

2.1 UMC configuration and status

This chapter describes the specific status objects of the UMC

2.1.1 Definition of parameters

This area indicates different parameter statuses.

Input objects

Name	Description
0x2810 Driver_temperature_config	

Table 2.1: Configuration objects

Output objects

Name	Description
0x2801 Driver_temperature	
0x2802 Back_EMF	
0x2803 Current	

Table 2.2: Status objects

0x2801 Driver_temperature

Indicates temperature of driver 1 and 2.

Index	SubIndex	Name		
0x2801	1	Driver_temperature_1		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	-	-	[;]

Figure 2.1: Object description 0x2801.1 *Driver_temperature_1*

0x2802 Back_EMF

0x2802.1 *Back_EMF_A*

Index	SubIndex	Name		
0x2802	1	Back_EMF_A		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	-	-	[;]

Figure 2.2: Object description 0x2802.1 *Back_EMF_A*

0x2802.2 *Back_EMF_B*

Index	SubIndex	Name		
0x2802	2	Back_EMF_B		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	-	-	[;]

Figure 2.3: Object description 0x2802.2 *Back_EMF_B*

0x2802.3 *Back_EMF_C*

Index	SubIndex	Name		
0x2802	3	Back_EMF_C		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	-	-	[;]

Figure 2.4: Object description 0x2802.3 *Back_EMF_C*

0x2802.4 *Back_EMF_D*

Index	SubIndex	Name		
0x2802	4	Back_EMF_D		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	-	-	[;]

Figure 2.5: Object description 0x2802.4 *Back_EMF_D*

0x2803 Current

0x2803.1 *Current_A*

Index	SubIndex	Name		
0x2803	1	Current_A		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	-	-	[;]

Figure 2.6: Object description 0x2803.1 *Current_A*

0x2803.2 *Current_B*

Index	SubIndex	Name		
0x2803	2	Current_B		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	-	-	[;]

Figure 2.7: Object description 0x2803.2 *Current_B*

0x2803.3 *Current_C*

Index	SubIndex	Name		
0x2803	3	Current_C		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	-	-	[;]

Figure 2.8: Object description 0x2803.3 *Current_C*

0x2803.4 *Current_D*

Index	SubIndex	Name		
0x2803	4	Current_D		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	-	-	[;]

Figure 2.9: Object description 0x2803.4 *Current_D*

0x2804 PWM

0x2804.1 *PWM_A*

Index	SubIndex	Name		
0x2804	1	PWM_A		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	-	-	[;]

Figure 2.10: Object description 0x2804.1 *PWM_A*

0x2804.2 *PWM_B*

Index	SubIndex	Name		
0x2804	2	PWM_B		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	-	-	[;]

Figure 2.11: Object description 0x2804.2 *PWM_B*

0x2804.3 *PWM_C*

Index	SubIndex	Name		
0x2804	3	PWM_C		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	-	-	[;]

Figure 2.12: Object description 0x2804.3 *PWM_C*

0x2804.4 *PWM_D*

Index	SubIndex	Name		
0x2804	4	PWM_D		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	-	-	[;]

Figure 2.13: Object description 0x2804.4 *PWM_D*

0x2810 Driver_temperature_config

Objects involved :0x2810 *Driver_temperature_config*

0x2810.1 *Protection_schmitt_triggers_low*

Index	SubIndex	Name		
0x2810	1	Protection_schmitt_triggers_low		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW	1	-	[;]

Figure 2.14: Object description 0x2810.1 *Protection_schmitt_triggers_low*

0x2810.2 *Protection_schmitt_triggers_high*

Index	SubIndex	Name		
0x2810	1	Protection_schmitt_triggers_low		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW	1	-	[;]

Figure 2.15: Object description 0x2810.1 *Protection_schmitt_triggers_low*

2.2 Axis synchronization mode

2.2.1 Introduction

This mode permit to synchronized two axis. This synchronization is authorized on the UMC card with the control of 2 axes (axes 1 and 2).

The synchronization mode is configurable with object 0x2A01 S12_Synchro_config.

2.2.2 Mode activation

The activation of the axis synchronization mode is conditioned by several parameters:

- both axes are in same state : Operation Enabled
- both axes are in same operating mode (see 0x6060 a1_Modes_of_operation)
- the gap between two axes must be < 0x2A01.2 *Max_diff*

if these conditions are not met and 0x2A01.1 *Mode_Synchro* != 0, the target sended is 0.

2.2.3 Definition of parameters

This area indicates different parameter statuses.

Input objects

Name	Description
0x2A01 S12_Synchro_config	

Table 2.3: Configuration objects

Output objects

Name	Description
0x2A00 S12_Synchro_status	

Table 2.4: Status objects

0x2A00 S12_Synchro_status

This area indicates different parameter statuses.

0x2A00.1 ***Flag*** Indicates the status of synchronization mode.

Index	SubIndex	Name		
0x2A00	1	Flag		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	-	-	[1;5]

Figure 2.16: Object description 0x2A00.1 *Flag*

Flag	Description
1	SYNC OFF
2	SYNC INSIDE WINDOW
3	SYNC ACTIVE CORRECTOR
4	SYNC ERROR MODE
5	SYNC ERROR MAX

Table 2.5: Flag

`0x2A00.2` ***Error*** Indicates error between axis.

Index	SubIndex	Name		
0x2A00	2	Error		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RO,TPDO	-	-	[0x80000000;0x7FFFFFFF]

Figure 2.17: Object description `0x2A00.2` *Error*

`0x2A00.3` ***Corrector*** Indicates applied corrector.

Index	SubIndex	Name		
0x2A00	3	Corrector		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RO,TPDO	-	-	[0x80000000;0x7FFFFFFF]

Figure 2.18: Object description `0x2A00.3` *Corrector*

`0x2A01` S12_Synchro_config

The object controls the synchronization of axis 2 on axis 1.

`0x2A01.1` ***Mode_Synchro*** Define the mode of synchronization.

Index	SubIndex	Name		
0x2A01	1	Mode_Synchro		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT8	RW,RPDO	0	-	[0;1]

Figure 2.19: Object description `0x2A01.1` *Mode_Synchro*

Mode_Synchro	Description
0	Off
1	On

Table 2.6: Mode_Synchro

`0x2A01.2` ***Max_diff*** Configure the max acceptable difference.

Index	SubIndex	Name		
0x2A01	2	Max_diff		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW,RPDO	5242880	-	[0x80000000;0x7FFFFFFF]

Figure 2.20: Object description `0x2A01.2` *Max_diff*

`0x2A01.3` ***Coeff*** Configure the coefficient.

Index	SubIndex	Name		
0x2A01	3	Coeff		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW,RPDO	8388608	-	[0x80000000;0x7FFFFFFF]

Figure 2.21: Object description `0x2A01.3` *Coeff*

`0x2A01.4` ***Window*** Configure the synchronization acceptance window.

Index	SubIndex	Name		
0x2A01	4	Window		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW,RPDO	131072	-	[0x80000000;0x7FFFFFFF]

Figure 2.22: Object description `0x2A01.4` *Window*

`0x2A01.5` ***Offset*** Configure the offset.

Index	SubIndex	Name		
0x2A01	5	Offset		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW,RPDO	0	-	[0x80000000;0x7FFFFFFF]

Figure 2.23: Object description `0x2A01.5` *Offset*

2.3 General object of Axis

This chapter describes the specific status objects of the UMC with the index range `0x4X00` to `0x4X1F`.

2.3.1 Multi axes

In the index range of the manufacturer-specific profile area (`0x4000` to `0x4FFF`), the device can manage up to 2 axes. In multi-axis devices, objects can be accessed by shifting as follows:

Index range	Description
0x4000 to 0x41FF	Axis 1
0x4200 to 0x43FF	Axis 2

Table 2.7: Index range of Specific Board Object

Each index range is split as follows:

Index range	Description
`0x4n00` to `0x4n1F`	Status Object

Table 2.8: Index range for each Axis

Note: "n" : Index range of Axis and "@" : number of axis

2.3.2 Definition of parameters

This area indicates different parameter statuses.

Input objects

Name	Description
a@_Motor_status	
a@_Motor_temperature_config	
a@_Brake_config	

Table 2.9: Configuration objects

Output objects

Name	Description
a@_Motion_status	
a@_Motor_status	

Table 2.10: Status objects

`0x4n00` **a@_Motion_status**

`0x4n00.1` ***Error***

Index	SubIndex	Name		
0x4n00	1	Error		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT16	RO,TPDO	-	-	[;]

Figure 2.24: Object description `0x4n00.1` *Error*

0x4n06 a@_Motor_status

0x4n06.1 *Hall_raw_value*

Index	SubIndex	Name		
0x4n06	1	Hall_raw_value		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT8	RO,TPDO	-	-	[;]

Figure 2.25: Object description 0x4n06.1 *Hall_raw_value*

0x4n06.2 *Hall_phase*

Index	SubIndex	Name		
0x4n06	2	Hall_phase		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT8	RO,TPDO	-	-	[;]

Figure 2.26: Object description 0x4n06.2 *Hall_phase*

0x4n06.3 *Powered_phase*

Index	SubIndex	Name		
0x4n06	3	Powered_phase		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT8	RO,TPDO	-	-	[;]

Figure 2.27: Object description 0x4n06.3 *Powered_phase*

0x4n06.4 *Command*

Index	SubIndex	Name		
0x4n06	4	Command		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	-	-	[;]

Figure 2.28: Object description 0x4n06.4 *Command*

0x4n06.5 *Error*

Index	SubIndex	Name		
0x4n06	5	Error		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT16	RO,TPDO	-	-	[;]

Figure 2.29: Object description 0x4n06.5 *Error*

0x4n06.6 *Temperature*

Index	SubIndex	Name		
0x4n06	6	Temperature		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	-	-	[;]

Figure 2.30: Object description 0x4n06.6 *Temperature*

0x4n07 a@_Motor_config

0x4n07.1 *Type*

Index	SubIndex	Name		
0x4n07	1	Type		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT16	RW	0	-	[;]

Figure 2.31: Object description 0x4n07.1 *Type*

0x4n07.2 *Peak_current*

Index	SubIndex	Name		
0x4n07	2	Peak_current		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	0	-	[0x0000000;0xFFFFFFFF]

Figure 2.32: Object description 0x4n07.2 *Peak_current*

0x4n07.3 *Pole_pair*

Index	SubIndex	Name		
0x4n07	3	Pole_pair		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	1	-	[0x0000000;0xFFFFFFFF]

Figure 2.33: Object description 0x4n07.3 *Pole_pair*

0x4n07.4 *Max_velocity*

Index	SubIndex	Name		
0x4n07	4	Max_velocity		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	0	-	[0x0000000;0xFFFFFFFF]

Figure 2.34: Object description 0x4n07.4 *Max_velocity*

0x4n07.5 *Velocity_constant*

Index	SubIndex	Name		
0x4n07	5	Velocity_constant		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	0	-	[0x0000000;0xFFFFFFFF]

Figure 2.35: Object description 0x4n07.5 *Velocity_constant*

0x4n07.6 *Current_constant*

Index	SubIndex	Name		
0x4n07	6	Current_constant		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	0	-	[0x0000000;0xFFFFFFFF]

Figure 2.36: Object description 0x4n07.6 *Current_constant*

0x4n07.7 *Flags*

Index	SubIndex	Name		
0x4n07	7	Flags		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT16	RW	0	-	[0;1]

Figure 2.37: Object description 0x4n07.7 *Flags*

0x4n81 a@_Motor_temperature_config

0x4n81.1 *Sensor_type*

Index	SubIndex	Name		
0x4n81	1	Sensor_type		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT32	RW	0	-	[;]

Figure 2.38: Object description 0x4n81.1 *Sensor_type*

0x4n81.2 *Sensor_constant*

Index	SubIndex	Name		
0x4n81	2	Sensor_constant		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT32	RW	1	-	[;]

Figure 2.39: Object description 0x4n81.2 *Sensor_constant*

0x4n81.3 *Protection_schmitt_triggers_low*

Index	SubIndex	Name		
0x4n81	3	Protection_schmitt_triggers_low		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW	1	-	[;]

Figure 2.40: Object description 0x4n81.3 *Protection_schmitt_triggers_low*

0x4n81.4 *Protection_schmitt_triggers_high*

Index	SubIndex	Name		
0x4n81	4	Protection_schmitt_triggers_high		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW	1	-	[;]

Figure 2.41: Object description 0x4n81.4 *Protection_schmitt_triggers_high*

0x4n82 a@_Brake_config

0x4n82.1 *Brake_maintain*

Index	SubIndex	Name		
0x4n82	1	Brake_maintain		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT32	RW	0	-	[;]

Figure 2.42: Object description 0x4n82.1 *Brake_maintain*

0x4n82.2 *Brake_peak*

Index	SubIndex	Name		
0x4n82	2	Brake_peak		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT32	RW	1	-	[;]

Figure 2.43: Object description 0x4n82.2 *Brake_peak*

Value	Flags
0x0001	REVERSE_POLARITY

Table 2.11: Flags motor

0x4n07 a@_Motor_config

Objects involved : 0x4007 *a1_Motor_config*.

0x4n07.1 *Type*

Index	SubIndex	Name		
0x4n07	1	Type		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT16	RW	0	-	[;]

Figure 2.44: Object description 0x4n07.1 *Type*

Value	Motor type
0x0	No motor
0x0101	DC motor
0x0201	BLDC trapezoidal
0x0202	BLDC sinusoidal with hall
0x0203	BLDC sinusoidal with incremental encoder

Table 2.12: Motor type

0x4n07.2 *Peak_current*

Index	SubIndex	Name		
0x4n07	2	Peak_current		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	0	-	[0x0000000;0xFFFFFFFF]

Figure 2.45: Object description 0x4n07.2 *Peak_current*

0x4n07.3 *Pole_pair*

Index	SubIndex	Name		
0x4n07	3	Pole_pair		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	1	-	[0x0000000;0xFFFFFFFF]

Figure 2.46: Object description 0x4n07.3 *Pole_pair*

0x4n07.4 *Max_velocity*

Index	SubIndex	Name		
0x4n07	4	Max_velocity		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	0	-	[0x0000000;0xFFFFFFFF]

Figure 2.47: Object description 0x4n07.4 *Max_velocity*

0x4n07.5 *Velocity_constant*

Index	SubIndex	Name		
0x4n07	5	Velocity_constant		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	0	-	[0x0000000;0xFFFFFFFF]

Figure 2.48: Object description 0x4n07.5 *Velocity_constant*

`0x4n07.6` ***Current_constant***

Index	SubIndex	Name		
0x4n07	6	Current_constant		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	0	-	[0x0000000;0xFFFFFFFF]

Figure 2.49: Object description `0x4n07.6` *Current_constant*

`0x4n07.7` ***Flags***

Index	SubIndex	Name		
0x4n07	7	Flags		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT16	RW	0	-	[0;1]

Figure 2.50: Object description `0x4n07.7` *Flags*

2.4 Control Loop of Axis

2.4.1 Multi axes

In the index range of the manufacturer-specific profile area (`0x4000` to `0x43FF`), the device can manage up to 2 axes. In multi-axis devices, objects can be accessed by shifting as follows:

Index range	Description
0x4000 to 0x41FF	Axis 1
0x4200 to 0x43FF	Axis 2

Table 2.13: Index range of Specific Board Object

Each index range of Control Loop is split as follows:

Index range	Description
0x4n20 to 0x4n3F	Control Loop Torque
0x4n40 to 0x4n5F	Control Loop Velocity
0x4n60 to 0x4n7F	Control Loop Position

Table 2.14: Index range for each Axis

Note: 'n' : common instance of index; '@' : number of axis

2.4.2 Definition of parameters

Objects in this area configure the control loop (PID) of Torque, Velocity and Position and also configure sensor. Some object indicates differents status.

This index range of Control Loop is split as follows:

Index range	Description	Index Object	Description
0x4n20 to 0x4n3F	Control Loop Torque	0x4n20	a@_Torque_PID_status
		0x4n21	a@_Torque_PID_config
		0x4n22	a@_Torque_sensor_status
		0x4n23	a@_Torque_sensor_config
		0x4n24	a@_Torque_sensor_filter
		0x4n25	a@_Torque_sensor_conditioning
0x4n40 to 0x4n5F	Control Loop Velocity	0x4n40	a@_Velocity_PID_status
		0x4n41	a@_Velocity_PID_config
		0x4n42	a@_Velocity_sensor_status
		0x4n43	a@_Velocity_sensor_config
		0x4n44	a@_Velocity_sensor_filter
		0x4n45	a@_Velocity_sensor_conditioning
0x4n60 to 0x4n7F	Control Loop Position	0x4n60	a@_Position_PID_status
		0x4n61	a@_Position_PID_config
		0x4n62	a@_Position_sensor_status
		0x4n63	a@_Position_sensor_config
		0x4n64	a@_Position_sensor_filter
		0x4n65	a@_Position_sensor_conditioning

Table 2.15: Index range of Control Loop and corresponding object

Note: 'n' : common instance of index; '@' : number of axis

0x4n20 ***a@_Torque_PID_status***
0x4n40 ***a@_Velocity_PID_status***
0x4n60 ***a@_Position_PID_status***

Objects involved : 0x4020 *a1_Torque_PID_status*, 0x4040 *a1_Velocity_PID_status*, 0x4060 *a1_Position_PID_status*.

0x4n20.1 ***Input*** Indicates input of control loop PID.

Index	SubIndex	Name		
0x4n20	1	Input		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RO,TPDO	-	-	[0x80000000;0x7FFFFFFF]

Figure 2.51: Object description 0x4n20.1 *Input*

0x4n20.2 ***Error*** Indicates error of control loop PID

Index	SubIndex	Name		
0x4n20	2	Error		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RO,TPDO	-	-	[0x80000000;0x7FFFFFFF]

Figure 2.52: Object description 0x4n20.2 *Error*

0x4n20.3 ***Integrator*** Indicates integrator of control loop PID

Index	SubIndex	Name		
0x4n20	3	Integrator		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RO,TPDO	-	-	[0x80000000;0x7FFFFFFF]

Figure 2.53: Object description 0x4n20.3 *Integrator*

`0x4n20.4` ***Output*** Indicates output of control loop PID

Index	SubIndex	Name		
0x4n20	4	Output		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RO,TPDO	-	-	[0x80000000;0x7FFFFFFF]

Figure 2.54: Object description `0x4n20.4` *Output*

`0x4n21` ***a@_Torque_PID_config***
`0x4n41` ***a@_Velocity_PID_config***
`0x4n61` ***a@_Position_PID_config***

Objects involved : `0x4021` *a1_Torque_PID_config*, `0x4041` *a1_Velocity_PID_config*, `0x4061` *a1_Position_PID_config*.

`0x4n21.1` ***P*** Configure the proportional of control loop PID

Index	SubIndex	Name		
0x4n21	1	P		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW	52428800	-	[0x80000000;0x7FFFFFFF]

Figure 2.55: Object description `0x4n21.1` *P*

`0x4n21.2` ***I*** Configure the integral of control loop PID

Index	SubIndex	Name		
0x4n21	2	I		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW	1310720	-	[0x80000000;0x7FFFFFFF]

Figure 2.56: Object description `0x4n21.2` *I*

`0x4n21.3` ***D*** Configure the derivative of control loop PID

Index	SubIndex	Name		
0x4n21	3	D		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW	-	-	[0x80000000;0x7FFFFFFF]

Figure 2.57: Object description `0x4n21.3` *D*

`0x4n21.4` ***Min*** Configure the minimun of control loop PID

Index	SubIndex	Name		
0x4n21	4	Min		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW	-	-	[0x80000000;0x7FFFFFFF]

Figure 2.58: Object description `0x4n21.4` *Min*

`0x4n21.5` ***Max*** Configure the maximun of control loop PID

Index	SubIndex	Name		
0x4n21	5	Max		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW	-	-	[0x80000000;0x7FFFFFFF]

Figure 2.59: Object description `0x4n21.5` *Max*

`0x4n21.6` ***Threshold*** Configure the threshold of control loop PID

Index	SubIndex	Name		
0x4n21	6	Threshold		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW	0	-	[0x80000000;0x7FFFFFFF]

Figure 2.60: Object description 0x4n21.6 *Threshold*

`0x4n21.7` ***Frequency_divider*** Configure the threshold of control loop PID

Index	SubIndex	Name		
0x4n21	7	Frequency_divider		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW	1	-	[-0x7FFF;0x7FFF]

Figure 2.61: Object description 0x4n21.7 *Frequency_divider*

`0x4n21.8` ***PID_flags*** Flags for PID

Index	SubIndex	Name		
0x4n21	8	PID_flags		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT16	RW	0	-	[0;1]

Figure 2.62: Object description 0x4n21.8 *PID_flags*

Value	Flags
0x0001	ANTI_REVERSE

Table 2.16: PID flags

`0x4n22` ***a@_Torque_sensor_status***
`0x4n42` ***a@_Velocity_sensor_status***
`0x4n62` ***a@_Position_sensor_status***

Objects involved : 0x4022 *a1_Torque_sensor_status*, 0x4042 *a1_Velocity_sensor_status*, 0x4062 *a1_Position_sensor_status*.

`0x4n22.1` ***Raw_Data*** Indicates raw data output from the sensor

Index	SubIndex	Name		
0x4n22	1	Raw_Data		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RO,TPDO	-	-	[0x80000000;0x7FFFFFFF]

Figure 2.63: Object description 0x4n22.1 *Raw_Data*

`0x4n22.2` ***Flags***

Index	SubIndex	Name		
0x4n22	2	Flags		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RO,TPDO	-	-	[-0x7FFF;0x7FFF]

Figure 2.64: Object description 0x4n22.2 *Flags*

`0x4n22.3` ***Value*** Indicates value output after filter

Index	SubIndex	Name		
0x4n22	3	Value		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RO,TPDO	-	-	[0x80000000;0x7FFFFFFF]

Figure 2.65: Object description 0x4n22.3 *Value*

`0x4n23` ***a@_Torque_sensor_config***
`0x4n43` ***a@_Velocity_sensor_config***
`0x4n63` ***a@_Position_sensor_config***

Objects involved : 0x4023 *a1_Torque_sensor_config*, 0x4043 *a1_Velocity_sensor_config*, 0x4063 *a1_Position_sensor_config*.

`0x4n23.1` ***Sensor_select*** Selected sensor

Index	SubIndex	Name		
0x4n23	1	Sensor_select		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT16	RW	0	-	[0x00;0xFFFF]

Figure 2.66: Object description 0x4n23.1 *Sensor_select*

Value	Sensor type
0x0000	OFF
0x1100	TORQUE_FROM_MOTOR
0x1200	VELOCITY_FROM_MOTOR
0x1300	POSITION_FROM_MOTOR
0x2200	POSITION_FROM_VELOCITY
0x2300	VELOCITY_FROM_POSITION
0x3101	QEI_CH1
0x3102	QEI_CH2
0x3201	SSI_CH1
0x3202	SSI_CH2
0x4001	ANALOG_CH1
0x4002	ANALOG_CH2

Table 2.17: Sensor type

`0x4n23.2` ***Frequency_divider*** Configure the frequency divider of sensor

Index	SubIndex	Name		
0x4n23	2	Frequency_divider		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW	1	-	[-0x7FFF;0x7FFF]

Figure 2.67: Object description 0x4n23.2 *Frequency_divider*

`0x4n23.3` ***Sensor_flags*** Flags

Index	SubIndex	Name		
0x4n23	3	Sensor_flags		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT16	RW	0	-	[0;0]

Figure 2.68: Object description 0x4n23.3 *Sensor_flags*

`0x4n23.4` ***Param_0*** Configure paramater 0 depends on the sensor

Index	SubIndex	Name		
0x4n23	4	Param_0		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW	0	-	[-0x7FFF;0x7FFF]

Figure 2.69: Object description 0x4n23.4 *Param_0*

`0x4n23.5` ***Param_1*** Configure paramater 1 depends on the sensor

Index	SubIndex	Name		
0x4n23	5	Param_1		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW	0	-	[-0x7FFF;0x7FFF]

Figure 2.70: Object description 0x4n23.5 *Param_1*

`0x4n23.6` ***Param_2*** Configure paramater 2 depends on the sensor

Index	SubIndex	Name		
0x4n23	6	Param_2		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW	0	-	[-0x7FFF;0x7FFF]

Figure 2.71: Object description 0x4n23.6 *Param_2*

`0x4n23.7` ***Param_3*** Configure paramater 3 depends on the sensor

Index	SubIndex	Name		
0x4n23	7	Param_3		
Data Type	**Acces**	**Default**	**Unit**	**Range**
INT16	RW	0	-	[-0x7FFF;0x7FFF]

Figure 2.72: Object description 0x4n23.7 *Param_3*

`0x4n24` ***a@_Torque_sensor_filter***
`0x4n44` ***a@_Velocity_sensor_filter***
`0x4n64` ***a@_Position_sensor_filter***

Objects involved : 0x4024 *a1_Torque_sensor_filter*, 0x4044 *a1_Velocity_sensor_filter*, 0x4064 *a1_Position_sensor_filter*.

`0x4n24.1` ***Filter_select*** Selected filter, currently no choice

Index	SubIndex	Name		
0x4n24	1	Filter_select		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT16	RW	0	-	[0x00;0xFF]

Figure 2.73: Object description 0x4n24.1 *Filter_select*

`0x4n24.2` ***Param_0*** Configure paramater 0 depends on the filter

Index	SubIndex	Name		
0x4n24	2	Param_0		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW	0	-	[0x80000000;0x7FFFFFFF]

Figure 2.74: Object description 0x4n24.2 *Param_0*

`0x4n24.3` ***Param_1*** Configure paramater 1 depends on the filter

Index	SubIndex	Name		
0x4n24	3	Param_1		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW	0	-	[0x80000000;0x7FFFFFFF]

Figure 2.75: Object description 0x4n24.3 *Param_1*

`0x4n24.4` ***Param_2*** Configure paramater 2 depends on the filter

Index	SubIndex	Name		
0x4n24	4	Param_2		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW	0	-	[0x80000000;0x7FFFFFFF]

Figure 2.76: Object description 0x4n24.4 *Param_2*

`0x4n24.5` ***Param_3*** Configure paramater 3 depends on the filter

Index	SubIndex	Name		
0x4n24	5	Param_3		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW	0	-	[0x80000000;0x7FFFFFFF]

Figure 2.77: Object description 0x4n24.5 *Param_3*

`0x4n25` ***a@_Torque_sensor_conditioning***
`0x4n45` ***a@_Velocity_sensor_conditioning***
`0x4n65` ***a@_Position_sensor_conditioning***

`0x4n25.1` ***Pre_offset*** Configure the pre-offset of sensor

Index	SubIndex	Name		
0x4n25	1	Pre_offset		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW	0	-	[0x80000000;0x7FFFFFFF]

Figure 2.78: Object description 0x4n25.1 *Pre_offset*

`0x4n25.2` ***Scale*** Configure the scale of sensor

Index	SubIndex	Name		
0x4n25	2	Scale		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW	65536	-	[0x80000000;0x7FFFFFFF]

Figure 2.79: Object description 0x4n25.2 *Scale*

`0x4n25.3` ***Post_offset*** Configure the post-offset of sensor

Index	SubIndex	Name		
0x4n25	3	Post_offset		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW	0	-	[0x80000000;0x7FFFFFFF]

Figure 2.80: Object description 0x4n25.3 *Post_offset*

`0x4n25.4` ***Error_min*** Configure the error min of sensor

Index	SubIndex	Name		
0x4n25	4	Error_min		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW	-2147483648	-	[0x80000000;0x7FFFFFFF]

Figure 2.81: Object description 0x4n25.4 *Error_min*

`0x4n25.5` ***Error_max*** Configure the error max of sensor

Index	SubIndex	Name		
0x4n25	5	Error_max		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW	-2147483647	-	[0x80000000;0x7FFFFFFF]

Figure 2.82: Object description 0x4n25.5 *Error_max*

`0x4n25.6` ***Threshold_min*** Configure the threshold min of sensor

Index	SubIndex	Name		
0x4n25	6	Threshold_min		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW	-2147483648	-	[0x80000000;0x7FFFFFFF]

Figure 2.83: Object description 0x4n25.6 *Threshold_min*

`0x4n25.7` ***Threshold_max*** Configure the threshold max of sensor

Index	SubIndex	Name		
0x4n25	7	Threshold_max		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW	2147483647	-	[0x80000000;0x7FFFFFFF]

Figure 2.84: Object description 0x4n25.7 *Threshold_max*

`0x4n25.8` ***Threshold_mode*** Configure the threshold mode of sensor

Index	SubIndex	Name		
0x4n25	8	Threshold_mode		
Data Type	**Acces**	**Default**	**Unit**	**Range**
Q15.16	RW	0	-	[0x80000000;0x7FFFFFFF]

Figure 2.85: Object description 0x4n25.8 *Threshold_mode*

Value	Threshold type
0x0000	No threshold
0x0001	Min-max
0x0002	Modulo
0x0003	Old value

Table 2.18: Threshold type

Chapter 3

CANOpen

3.1 CANOpen

CANOpen is based on bus CAN (Controller Area Network) which is ISO-11898 standardized. CANOpen protocol is standardized by the CAN In Automation (CIA) under the name of CiA301.

3.1.1 CAN - CANOpen

CANOpen uses standard CAN frames to communicate on the bus. The frames are identified with an Id : CAN-ID. It is coded on 11 bits. The CAN-ID with the lowest value has priority if multiple frames are sent at the same time from different devices.

The simplified CANOpen frame:

11 bit	0	1	2	3	4	5	6	7
CAN-ID	Data							

Table 3.1: Frame CANOpen

Note: A CAN message can contain a maximum of 8 bytes of usefull data.
Note: The data is always sent on the bus in **little-endian format**.

3.1.2 CANOpen Node

Each device, also called Node, is identified by a unique identifier in the network called: Node-Id. The Node-Id can take a value in the range [1-127].

All descriptions and functionalities of a Node are described in the EDS (Electronic Data Sheet) file.

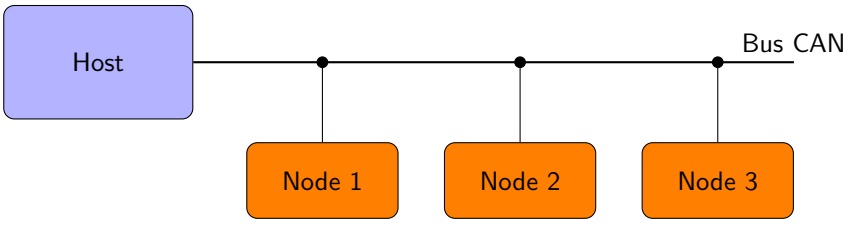


Table 3.2: Bus CAN

3.1.3 Communication

The CanOpen defines three communication protocol sequences :

- Master/Slave protocol
- Client/Server protocol
- Producer/Consumer protocol

Master/Slave protocol

This protocol works on a network where only one CanOpen master is present for a specific functionality. And therefore the other CanOpen devices are slaves. The master sends a request and the slave responds.

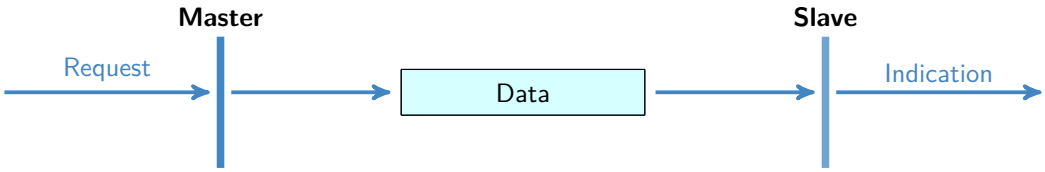


Figure 3.1: Unconfirmed Master/Slave protocol

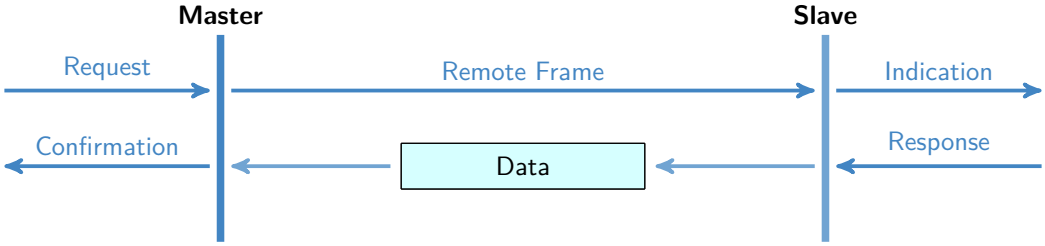


Figure 3.2: Confirmed Master/Slave protocol

Client/Server protocol

This protocol is used between a client and a server. When the client makes a request (download / upload), the server triggers the processing of the request. The server responds to the request when the task is completed.

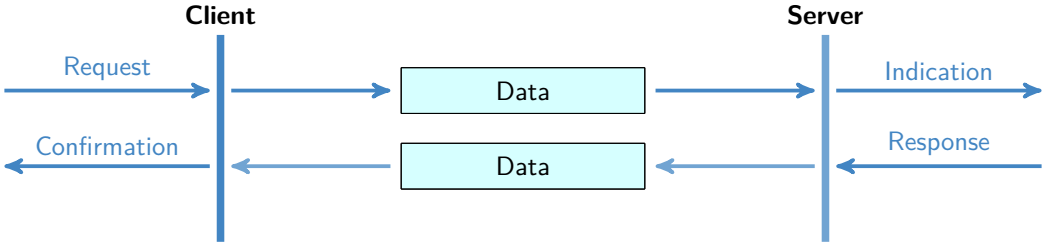


Figure 3.3: Client/Server protocol

Producer/Consumer protocol

This protocol works with a producer that sends a message that can be received by one or more devices on the network. The producer does not receive confirmation.

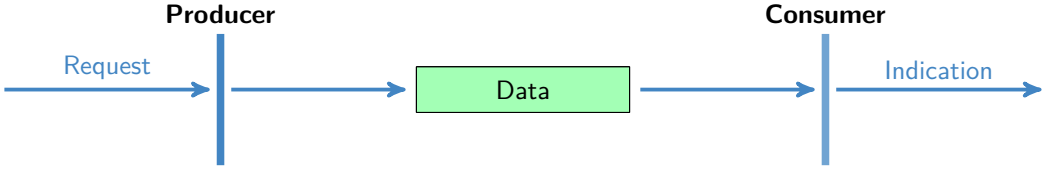


Figure 3.4: Push Producer/Consumer protocol

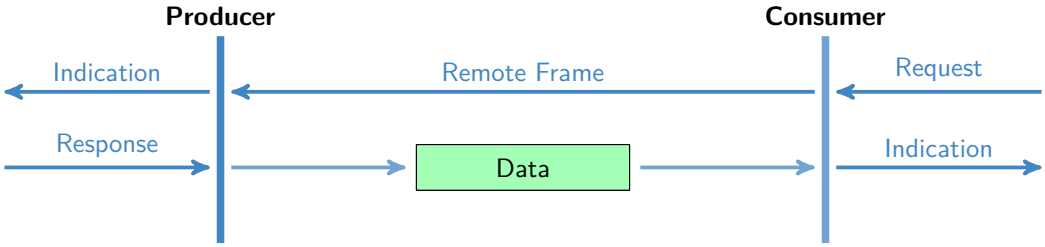


Figure 3.5: Pull Producer/Consumer protocol

3.2 CANOpen Services

The services provided by the CANOpen stack allow a standard communication between the devices on the network. These services allow communication object exchanges. There are 5 types of services:

- Network management services
  - Network management services (NMT)
  - Node guarding, network equipment monitoring
  - Heartbeat, network equipment monitoring
  - Boot-up
- Service data object (SDO), providing read and write access to the dictionary of objects
- Process data object (PDO), allowing to transmit data in real time process:
  - TPDO Transmit-PDO for use in data transmission
  - RPDO Receive-PDO for use in data reception
- SYNC, synchronization object used by PDO.
- EMCY, emergency

Each type of message is defined by the allocation of the following CAN-IDs:

Services	Default CAN-ID	CAN-ID configurable
NMT	0x000	
SYNC	0x080	yes
EMCY	0x080 + Node-Id	yes
TPDO1	0x180 + Node-Id	yes
RPDO1	0x200 + Node-Id	yes
TPDO2	0x280 + Node-Id	yes
RPDO2	0x300 + Node-Id	yes
TPDO3	0x380 + Node-Id	yes
RPDO3	0x400 + Node-Id	yes
TPDO4	0x480 + Node-Id	yes
RPDO4	0x500 + Node-Id	yes
TSDO	0x580 + Node-Id	
RSDO	0x600 + Node-Id	
Boot-Up	0x700 + Node-Id	
Nodeguarding and Heartbeat	0x700 + Node-Id	

Table 3.3: Index of CAN-ID Services

Some services can have a CAN-ID configurable by a communication object:

Services	Object
SYNC	0x1005
EMCY	0x1014
TPDOX	0x180X
RPDOX	0x140X

Table 3.4: Index of CAN-ID Services

But the changing CAN-ID should not interfere with the following reserved CAN-IDs:

CAN-ID	used by COB
0x000	NMT
0x00 0x07F	reserved
0x101 0x180	reserved
0x581 0x5FF	default SDO (tx)
0x601 0x67F	default SDO (rx)
0x6E0 0x6FF	reserved
0x701 0x77F	NMT Error Control
0x780 0x7FF	reserved

Table 3.5: Restricted CAN-ID

3.3 Object dictionary

The object dictionary is a collection of all the data items which have an influence on the behavior of the application objects, the communication objects and the state machine used on the device. Each device on the network has its own object dictionary.

The object dictionary is divided into several areas:

Index range	Description
0x0000	Reserved
0x0001 to 0x025F	Data types
0x0260 to 0x0FFF	Reserved
0x1000 to 0x1FFF	Communication profile area
0x2000 to 0x5FFF	Manufacturer-specific profile area
0x6000 to 0x9FFF	Standardized profile area

Table 3.6: Object dictionary area

The **Communication profile area** contain the communication specific parameters. These objects are common to all CANopen devices.

The **Standardized profile area** contain all data objects common to a profiles of CANopen devices that may be read or written via the network. The objects from 6000 h to 9FFF h describe parameters and functionality.

The **Manufacturer-specific** profile area contains the objects for specific UniSwarm features.

3.3.1 Description of the object dictionary

The objects of the dictionary are described by several parameters. This description is materialized by an EDS file: Electronic Data Sheet. ASCII format respecting a strict syntax that can be used by the bus configuration software.

Index and sub-index

These form the unique identifier of an object in the objects dictionary in hexadecimal notation.

Object code

The object code denotes what kind of object is at a particular index within the objects dictionary.
They can be one of the following:

Object name	Description	Codage
NULL	An object with no data fields	0x00
DOMAIN	A large variable amount of data	0x02
DEFTYPE	A type definition for simple data type such as a Boolean, Unsigned16	0x05
DEFSTRUCT	Defines a new record type	0x06
VAR	A single value	0x07
ARRAY	A data area in which each entry is of the same data type.	0x08
RECORD	A data area that contains entries that are a combination of simple data types.	0x09

Table 3.7: Object code

Data type

The data type information indicates the data type of the object.

Index	Name	Size in byte
0x0001	Boolean	1
0x0002	Integer8	1
0x0003	Integer16	2
0x0004	Integer32	4
0x0005	Unsigned8	1
0x0006	Unsigned16	2
0x0007	Unsigned32	4
0x0008	Real32	4
0x0009	VISIBLE STRING	...
0x000A	OCTET STRING	...
0x000B	UNICODE STRING	...
0x000C	TIME OF DAY	..
0x000D	TIME DIFFERENCE	...
0x000F	Domain	...
0x0010	Integer24	3
0x0011	Real64	8
0x0012	Integer40	5
0x0013	Integer48	6
0x0014	Integer56	7
0x0015	Integer64	8
0x0016	Unsigned24	3
0x0018	Unsigned40	5
0x0019	Unsigned48	6
0x001A	Unsigned56	7
0x001B	Unsigned64	8
0x0020	PDO COMMUNICATION PARAMETER	...
0x0021	PDO MAPPING	...
0x0022	SDO PARAMETER	...
0x0023	IDENTITY	...

Table 3.8: Data type

Note: Data type with indices from 0x0001 to 0x0007, 0x0010, from 0x0012 to 0x0016 , and from 0x0018 to 0x001B may be mapped in order to define the appropriate space in the RPDO.

Note: Data type with indices from 0x0008 to 0x000F, 0x0011, from 0x0020 to 0x0023 shall not be mapped into RPDOs

Access usage

Acces object:

- rw: read and write access
- wo: write only access
- ro: read only access
- const: read only access, value is constant

In addition, there are the access attributes for the PDOs:

- rww: read and write access and can be mapped on RPDO
- rwr: read and write access and can be mapped on TPDO

3.4 Network management services (NMT)

Network management (NMT) follows a master-slave structure. All devices are NMT slaves but the network must have a device master (device master, computer or other).

The service provides a tool to initiate, start, monitor, reset or stop the devices. Monitoring is made with the Node guarding and Heartbeat functionalities.

3.4.1 NMT Network management

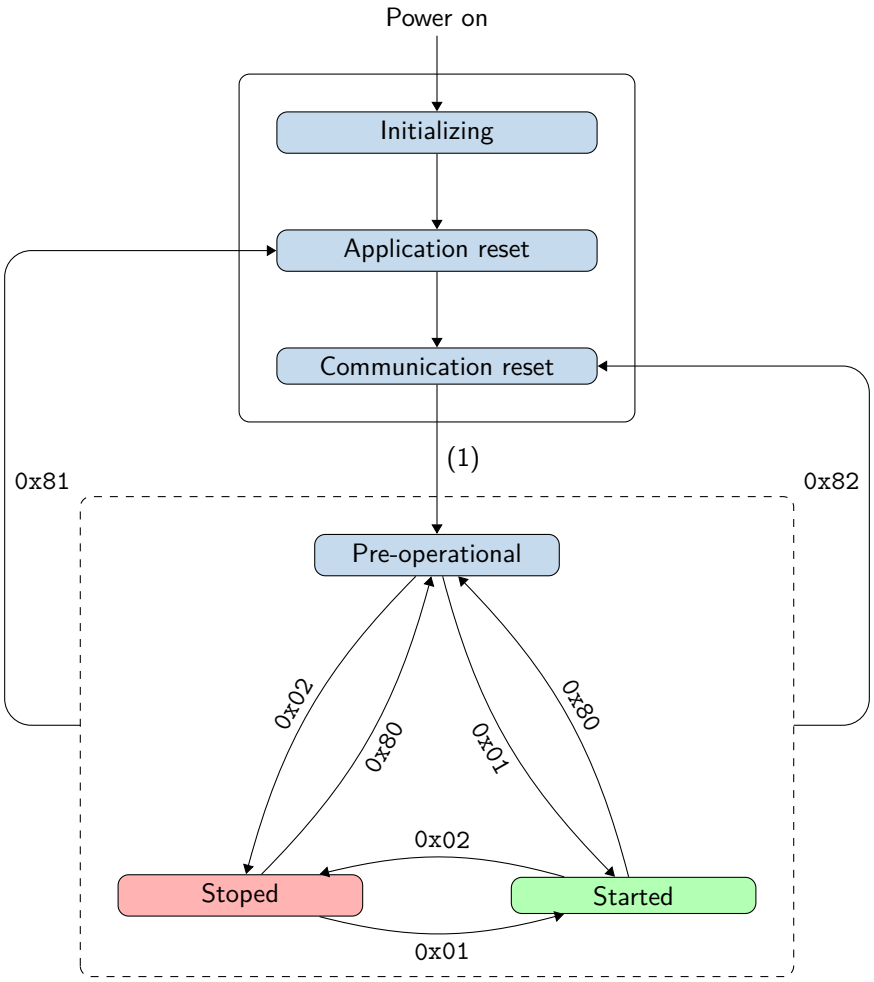
The NMT master controls the state of each NMT slave. The state can be chosen among the following ones: Stopped, Pre-operational, Started, Initialization.

NMT state machine

The NMT state machine determines the behavior of the communication function unit.

- state **Initialization**
  - **Initializing**: the device enters this state after a power-on or an hardware/software reset.
  - **Application reset**: The object dictionnary in Manufacturer-specific profile area, index range 0x2000 to 0xFFFF is reset.
  - **Communication reset**: The object dictionnary in Communication profile area, index range 0x1000 to 0x1FFF is reset.
- state **Pre-operational**
- state **Started**
- state **Stopped**

The following figure show the state diagram:



(1) Automatic switch to Pre-operational and send Boot-up message

Figure 3.6: Different status of a CAN Open node

NMT frame

The NMT frame allow to change the state of device. An NMT frame have the CAN-ID 0x000 and a payload of two bytes. The first one is the mode and the second one the node id.

CAN-ID	Byte[0]	Byte[1]
0x000	Mode	Node-Id

Figure 3.7: NMT frame

Mode value	Description
0x01	Start node
0x02	Stop node
0x80	Enter in Pre-operational mode
0x81	Application reset
0x82	Communications reset

Figure 3.8: MODE values in NMT frame

NMT States and Services

Authorized services according to the NMT state:

	Pre-operational	Started	Stopped
PDO		X	
SDO	X	X	
SYNC	X	X	
EMCY	X	X	
Node guarding	X	X	X
Heartbeat	X	X	X

Figure 3.9: NMT States and authorized Services

3.4.2 Node Guarding - Heartbeat

Two services are available to detect an error on the CAN network: the **Node guarding** service and the **Heartbeat** service.

- **Node guarding**: the master sends a message periodically and each slave has to respond within a time limit.
- **Heartbeat**: each slave sends a message with his state without prior request from the master.

Note: The **Heartbeat** service has priority over the **Node guarding** service. Activation of the **Heartbeat** service results in the deactivation of the **Node Guarding** service.

Node guarding

This service monitors the status of devices on the bus and makes it possible to detect remote errors on the network.
The master and the slave monitor each other: the master cyclically requests the NMT status of the slave. In each response from the slave, the Toggle-bit (bit 7) is toggled.

Two monitoring functions are implemented:

- **Node guarding**: The master can react accordingly if these responses are not sent or if the slave always responds with the same bit Toggle.
- **Life guarding**: The slave monitors the reception of messages from the master, if the message is not sent within the allotted time, the Life Time, the slave triggers an EMCY message (with ode 0x8130) see table 3.11.
  The Life time : *Life Time = 0x100C Guard Time x 0x100D Life Time Factor*.

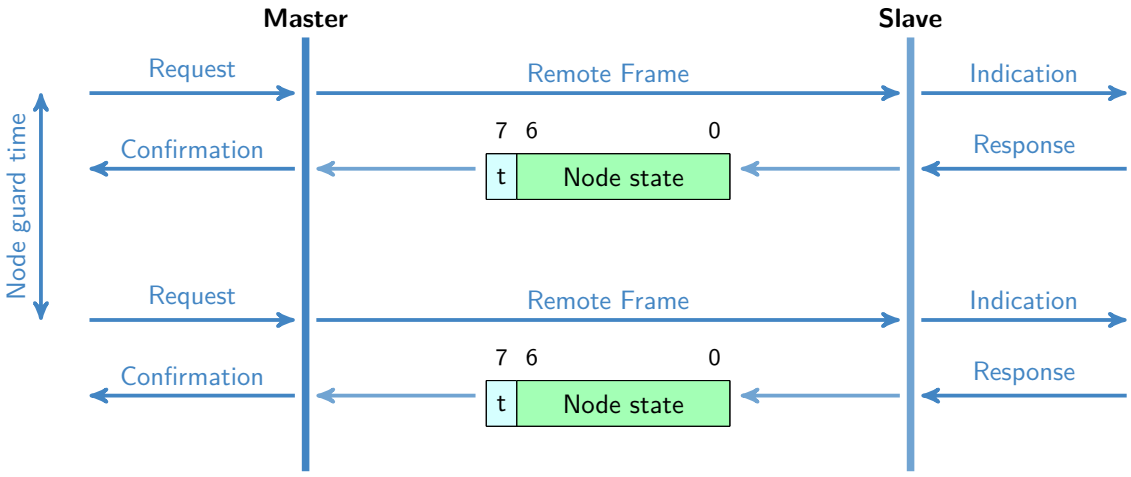
Note: This service is activated by setting value in object 0x100C Guard Time and 0x100D Life Time Factor other than zero.

Node guarding frame The node guarding service follows a master / slave architecture: the NMT master sends an RTR (Remote Transmission Request) message with CAN-ID 700 + node Id to a slave and the slave responds with an 8-bit message.

The response message is built with a Toggle-bit (bit 7) and the current NMT state of the slave in bits 6 to 0. For the first response and after a NMT reset, the toggle bit should be 0.

CAN-ID	Byte[0]
0x700 + Node-Id	0x00

Figure 3.10: Node guarding frame



- t: Toggle Bit
- Node state:
  - 4: Stopped
  - 5: Started
  - 27: Pre-operational

Figure 3.11: Node guarding protocol

Heartbeat

The Heartbeat works in Producer/Consumer mode with one producer and 0 minimum consumer.

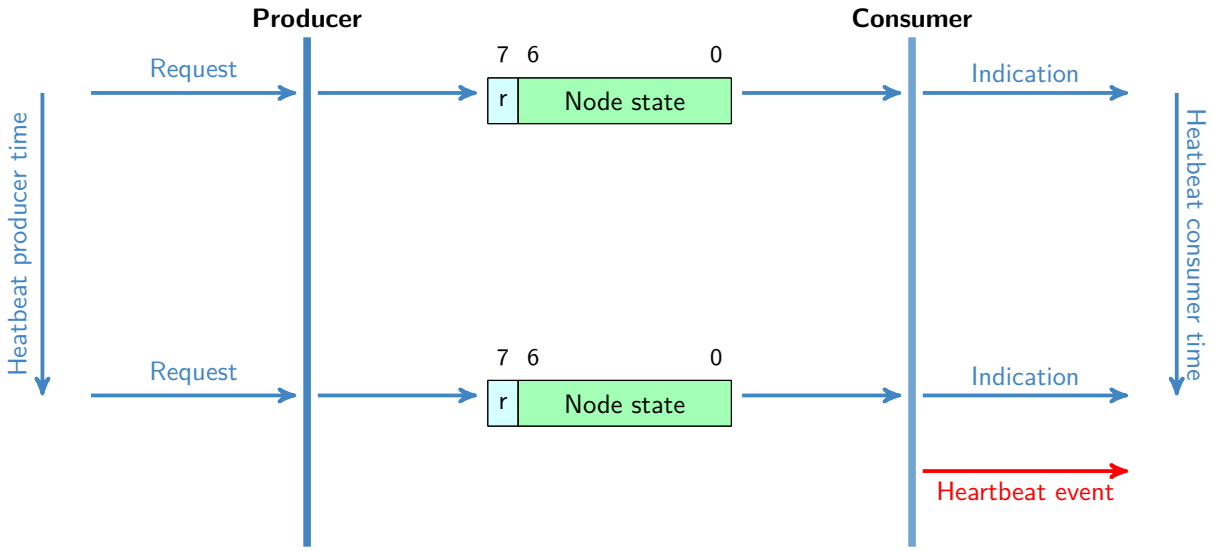
The producer heartbeat send a heartbeat message periodically, with the time between two messages defined by the "Producer heartbeat time" object.

The consumer check if it received a message in the time defined by the object "Consumer heartbeat time".

Note: This service is activated by setting the producer heartbeat time object in object 0x1017 Producer Heartbeat Time to a value other than zero.

CAN-ID	Byte[0]
0x700 + Node-Id	0x00

Figure 3.12: Heartbeat frame



- r: reserved (always 0)
- Node state:
  - 0: Boot-Up
  - 4: Stopped
  - 5: Operational
  - 127: Pre-operational

Figure 3.13: Heartbeat protocol

3.4.3 Boot-up

After power-up, the slave sends a Boot-up message to indicate that the Initializing phase is complete.

CAN-ID	Byte[0]	
0x700 + Node-Id	0x00	

Figure 3.14: Boot-up frame

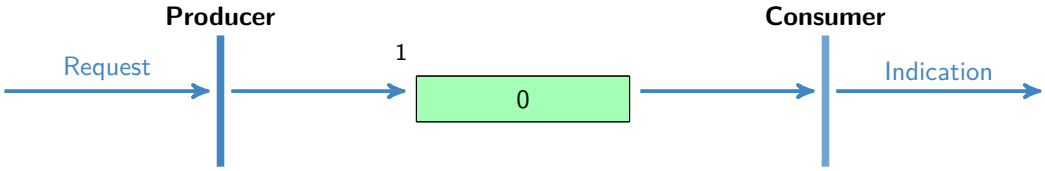


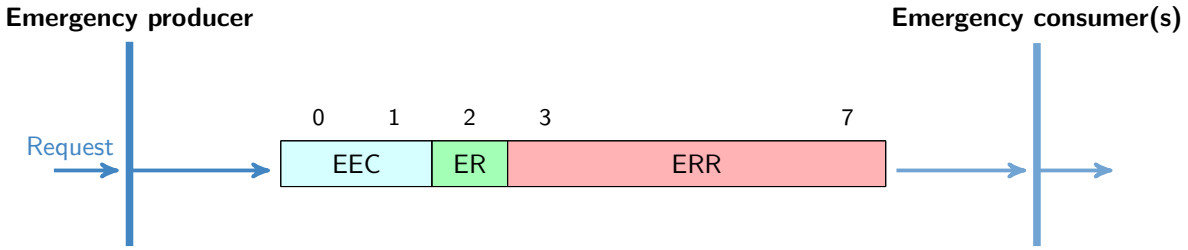
Figure 3.15: Boot-up protocol

3.5 EMCY

The EMCY service is used to transmit application faults associated with each station. When a fault is detected the device send a EMCY message with Emergency error code, Error register and error code (optional).

CAN-ID	Byte[0]	
0x080 + Node-Id	...	

Figure 3.16: EMCY frame



- EEC: Emergency error code
- ER: Error register (see object 0x1001 *Error Register*)
- ERR: Manufacturer-specific error code (Optional)

Figure 3.17: Emergency protocol with Byte[0]

EEC	Description
0x8130	Life guard error or heartbeat error

Table 3.9: Emergency error codes

3.5.1 Definition of parameters

0x1001 Error Register

Index	SubIndex	Name		
0x1001	0	Error Register		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT8	RO,TPDO	-	-	[;]

Figure 3.18: Object description 0x1001.0 *Error Register*

Bit	Description
0	Generic error
1	Current
2	Voltage
3	Temperature
4	Communication error (overrun, error state)
5	Device profile specific
6	reserved (always 0 b)
7	manufacturer-specific

Table 3.10: Error code register of object 0x1001

0x1029 Error Behaviour

Sets the behavior in the event of a serious device failure in the Operational NMT state. By default, the device automatically enter the Pre-operational NMT state.

Failures include the following communication errors:

- CAN interface bus stop conditions
- NodeGuarding "time out"
- Heartbeat "time out"
- Serious errors can also be caused by internal device failures.

`0x1029.1` ***Communication Error***

Index	SubIndex	Name		
0x1029	1	Communication Error		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT8	RW	0	-	[0;2]

Figure 3.19: Object description 0x1029.1 *Communication Error*

Bit	Description
0	Change to NMT state Pre-operational (default)
1	No change of the NMT state
2	Change to NMT state Stopped

Table 3.11: Error class values of 0x0x1029.1

3.6 Service data object (SDO)

This service provides access to the device object dictionary by an Index and Sub-index without time constraints in writing or reading.

Each network device is an SDO server, the one that holds the OD. The client refers to the node requesting to read or write an object value in the server's object dictionary

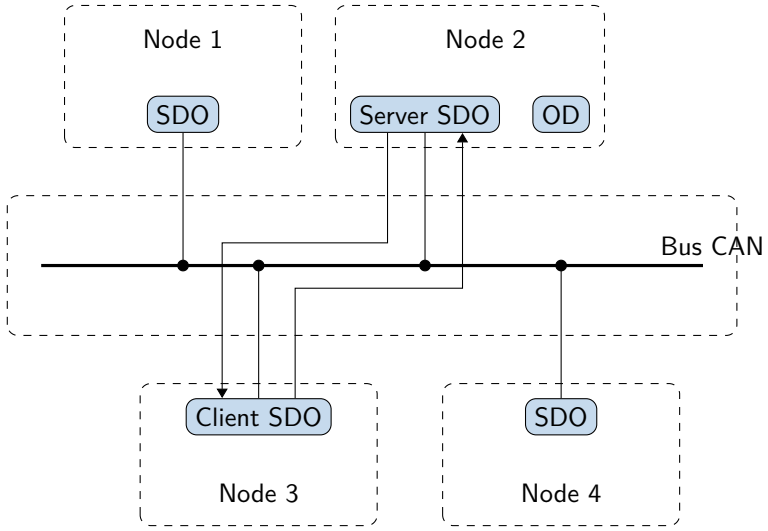


Table 3.12: Server/Client SDO

3.6.1 SDO message

CAN-ID	Byte[0-7]
CAN-ID+ Node-Id	...

Figure 3.20: SDO frame

- Cmd: Command
- Index: Index of object
- Sub-Index:

Figure 3.21: Expedited SDO Upload detail of frame

3.6.2 Expedited Transfer

This mode of communication is used to write or read data in object. The size of the object must be inferior or equal to 4 bytes. An answer is expected after each request, either with data, with a confirmation or with an error message.

SDO Reading

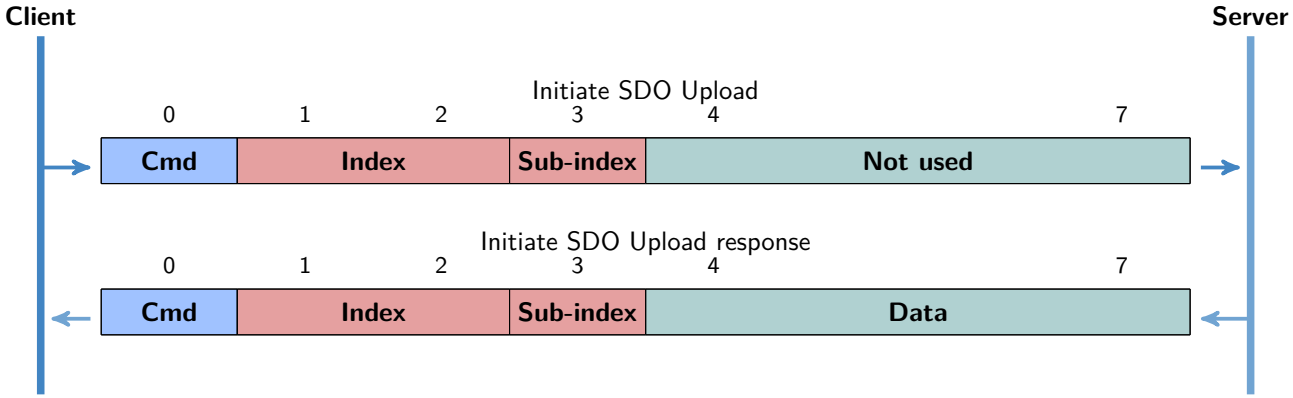


Figure 3.22: Expedited SDO Upload protocol

Cmd	Description
0x40	Upload/Reading request
0x4F	Upload/Reading response for data of size of 1 byte
0x4B	Upload/Reading response for data of size of 2 byte
0x47	Upload/Reading response for data of size of 3 byte
0x43	Upload/Reading response for data of size of 4 byte
0x2F	Download/Writing for data of size of 1 byte
0x2B	Download/Writing for data of size of 2 byte
0x27	Download/Writing for data of size of 3 byte
0x23	Download/Writing for data of size of 4 byte
0x60	Download/Writing response

Figure 3.23: List of Cmd

Initiate SDO Upload/Reading request

0	1	2	3	4			7
Cmd	Index		Sub-index	Data			
0x40	lsb	msb	Sub-index	0	0	0	0

Initiate SDO Upload/Reading response

Read response for a data size of 1 byte:

0	1	2	3	4			7
Cmd	Index		Sub-index	Data			
0x4F	lsb	msb	Sub-index	data	0	0	0

Read response for a data size of 2 byte:

0	1	2	3	4			7
Cmd	Index		Sub-index	Data			
0x4B	lsb	msb	Sub-index	lsb	msb	0	0

Read response for a data size of 3 byte:

0	1	2	3	4			7
Cmd	**Index**		**Sub-index**	**Data**			
0x47	lsb	msb	Sub-index	lsb	···	msb	0

Read response for a data size of 4 byte:

0	1	2	3	4			7
Cmd	**Index**		**Sub-index**	**Data**			
0x43	lsb	msb	Sub-index	lsb	···	···	msb

SDO Download/Writing

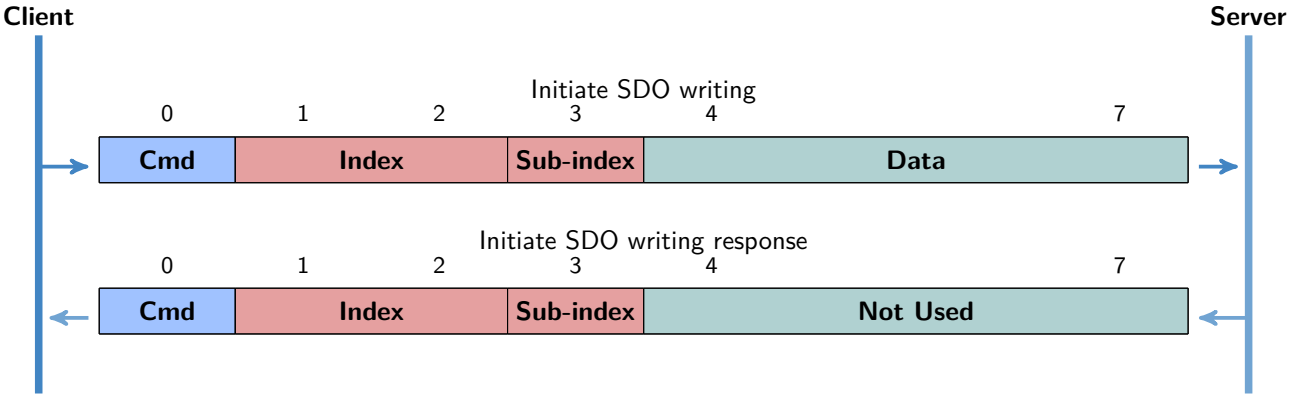


Figure 3.24: Expedited SDO Download protocol

Initiate SDO Download/Writing

Write request for data of size of 1 byte:

0	1	2	3	4			7
Cmd	**Index**		**Sub-index**	**Data**			
0x2F	lsb	msb	Sub-index	data	0	0	0

Write request for data of size of 2 byte:

0	1	2	3	4			7
Cmd	**Index**		**Sub-index**	**Data**			
0x2B	lsb	msb	Sub-index	lsb	msb	0	0

Write request for data of size of 3 byte:

0	1	2	3	4			7
Cmd	**Index**		**Sub-index**	**Data**			
0x27	lsb	msb	Sub-index	lsb	···	msb	0

Write request for data of size of 4 byte:

0	1	2	3	4			7
Cmd	**Index**		**Sub-index**	**Data**			
0x23	lsb	msb	Sub-index	lsb	···	···	msb

Initiate SDO Download/Writing response

0	1	2	3	4			7
Cmd	**Index**		**Sub-index**	**Data**			
0x60	lsb	msb	Sub-index	0	0	0	0

3.6.3 SDO abort transfer

Error response:

0	1	2	3	4			7
Cmd	**Index**		**Sub-index**	**SDO abort codes**			
0x80	lsb	msb	Sub-index	lsb	···	···	msb

3.6.4 SDO abort codes

Error codes	Description
0x05030000	Toggle bit not alternated
0x05040000	SDO protocol timed out
0x05040001	Client/server command specifier not valid or unknown
0x05040002	Invalid block size (block mode only)
0x05040003	Invalid sequence number (block mode only)
0x05040004	CRC error (block mode only)
0x05040005	Out of memory
0x06010000	Unsupported access to an object
0x06010001	Attempt to read a write only object
0x06010002	Attempt to write a read only object
0x06020000	Object does not exist in the object dictionary
0x06040041	Object cannot be mapped to the PDO
0x06040042	The number and length of the objects to be mapped would exceed PDO length
0x06040043	General parameter incompatibility reason
0x06040047	General internal incompatibility in the device
0x06060000	Access failed due to an hardware error
0x06070010	Data type does not match, length of service parameter does not match
0x06070012	Data type does not match, length of service parameter too high
0x06070013	Data type does not match, length of service parameter too low
0x06090011	Sub-index does not exist
0x06090030	Invalid value for parameter (download only)
0x06090031	Value of parameter written too high (download only)
0x06090032	Value of parameter written too low (download only)
0x06090036	Maximum value is less than minimum value
0x060A0023	Resource not available: SDO connection
0x08000000	General error
0x08000020	Data cannot be transferred or stored to the application
0x08000021	Data cannot be transferred or stored to the application because of local control
0x08000022	Data cannot be transferred or stored to the application because of the present device state
0x08000023	Object dictionary dynamic generation fails or no object dictionary is present
0x08000024	No data available

Table 3.13: SDO error codes

3.7 Process data object (PDO)

The purpose of the "Process Data Objects (PDO)" is to provide a transfer of data in real time during the operation of the controller. This service is performed without protocol overload or confirmation. The size of a PDO frame is variable and depends on the size of the object.

The PDO provides an interface to the application objects in the object dictionary. Data type and mapping of objects is determined by the corresponding PDO mapping structure in the object dictionary.

The PDO configuration process (PDO Mapping) allows to configure the number of objects in a PDO. This process uses the SDO service.
There are two types of PDO, the Transmit-PDO (TPDO) for use in data transmission and the Receive-PDO (RPDO) for use in data reception. The data transmission via the PDO service operates according to a producer / consumer relationship: RPDOs are frames received from the master or others nodes. TPDOs are frames transmitted to others.

PDOs are described by the PDO communication parameter and the PDO mapping parameter.

Note: A node can have a maximum of four TPDOs and four RPDOs.

There is an index couple for each PDO: the columns "Index RPDO communication" and "Index RPDO mapping" provide the indexes of specials objects used to read or modify the parameters of communication objects via an SDO object:

RPDO	CAN-ID	Object Index	
		Index RPDO communication	Index RPDO mapping
RPDO1	0x200 + Node-Id	0x1400	0x1600
RPDO2	0x300 + Node-Id	0x1401	0x1601
RPDO3	0x400 + Node-Id	0x1402	0x1602
RPDO4	0x500 + Node-Id	0x1403	0x1603

Table 3.14: Index used of RPDOs

TPDO	CAN-ID	Object Index	
		Index TPDO communication	Index TPDO mapping
TPDO1	0x180 + Node-Id	0x1800	0x1A00
TPDO2	0x280 + Node-Id	0x1801	0x1A01
TPDO3	0x380 + Node-Id	0x1802	0x1A02
TPDO4	0x480 + Node-Id	0x1803	0x1A03

Table 3.15: Index used of TPDOs

3.7.1 PDO message

There is two ways to transmit a PDO message:

- Synchronous transmission: the object are synchronized on SYNC.
- Event-driven transmission

This PDO is only activated if the status of CanOpen is "Started". It is necessary to activate the PDOs, for that the "valid" bit of CAN-ID must be to set to 0x0.

Example:
to activate or deactivate the TPDO1, the object with the index 0x1800 and sub-index 0x1 is used:

- Deactivate TPDO1: set sub-index 1 to 0x80000181
- Activate TPDO1: set sub-index 1 to 0x00000181

The value 0x181 is the id of TPDO1.

After each SYNC, two things happen in this order:

- for TPDOs: the slaves samples and copy the data into TPDOs which are then sent on the bus.
- for RPDOs: the previous received RPDO data from master is copied in the objects database and made available to the application.

3.7.2 SYNC

The master in a CAN Open network sends a unique sync frame for all the nodes. At reception, the nodes transmit theirs TPDOs and apply their RPDOs. A SYNC frame has the ID 0x080 and does not contain a payload.

CAN-ID
0x080

Figure 3.25: SYNC frame

The frequency of SYNC frame can be variable, but is always sent by the master.

3.7.3 PDO dynamic mapping

General procedure:
The following procedure shall be used for re-mapping, which may take place during the NMT state Pre-operational and during the NMT state Operational, if supported:

1. Deactivate PDO: setting the valid bit to 0x1 of sub-index 0x01 of the PDO communication parameter.
2. Disable mapping: setting sub-index 0x00 to 0.
3. Modify mapping: changing the values of the corresponding sub-indexes.
4. Enable mapping: setting sub-index 0x00 to the number of mapped objects.
5. Activate PDO: setting the valid bit to 0 of sub-index 0x01 of the PDO communication parameter.

3.7.4 PDO parameter objects

0x140n RPDO Parameter X

Note: 'n' : common instance of index (0,1,2,3) and 'X' : number of RPDO (1,2,3,4)
Objects involved : 0x1400 *RPDO Parameter 1*, 0x1401.0, 0x1402 *RPDO Parameter 3*, 0x1403 *RPDO Parameter 4*

The PDO parameter describes the communication abilities of the PDO.

Index	Sub-index	Name	Data type
RPDO: 0x1400 to 0x1403	0x00	Highest sub-index supported	Unsigned8
	0x01	COB ID	Unsigned32
	0x02	Transmission type	Unsigned8
	0x03	Inhibit time	Unsigned16
	0x04	reserved	Unsigned8
	0x05	Event timer	Unsigned16

Table 3.16: PDO config objects

0x140n.1 *COB ID*

Index	SubIndex	Name		
0x140n	1	COB ID		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	512	-	[0x00000080;0xFFFFFFFF]

Figure 3.26: Object description 0x140n.1 *COB ID*

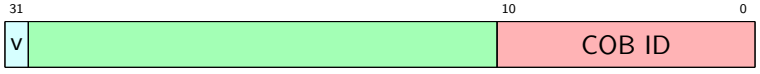


Table 3.17: COB ID

- **v**: valid

Value	Description
0x00	PDO exists / is valid
0x01	PDO does not exist / is not valid

Table 3.18: Description of bit 31: v

- **COB ID** of PDO

`0x140n.2` ***Transmission Type***

Index	SubIndex	Name		
0x140n	2	Transmission Type		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT8	RW	0	-	[;]

Figure 3.27: Object description 0x140n.2 *Transmission Type*

The transmission of the PDO depends on the configuration of the Transmission Types parameters which can be:

Value	Description
0x00 to 0xF0	Synchronous
0xFE	Event

Table 3.19: Description of RPDO transmission type

- **Synchronous**: the synchronous transmission type means that the data is transmitted immediately, but it is applied when SYNC is received. The SYNC service provides a data synchronization signal over the network.
- **Event**: the Event-driven transmission type means that the PDO may be received at any time and that the data is applied immediately after reception.

`0x160n` **RPDO Mapping X**

Note: 'n' : common instance of index (0,1,2,3) and 'X' : number of RPDO (1,2,3,4)
Objects involved : 0x1600 *RPDO Mapping 1*, 0x1601 *RPDO Mapping 2*, 0x1602 *RPDO Mapping 3*, 0x1603 *RPDO Mapping 4*

The PDO mapping parameter contains informations about the content of the PDO.

	Sub-index	Name	Data type
0x1600 to 0x1603	00	Number of mapped objects in PDO	Unsigned8
	01	1st object to be mapped	Unsigned32
	02	2 nd object to be mapped	Unsigned32
	...	...	Unsigned32
	64	64 th object to be mapped	Unsigned32

Figure 3.28: PDO mapping parameter

- **Sub-index 0x00**: the number of valid object entries within the mapping record. If it is equal to 0, the Mapping is disabled.
- **Sub-index 0x01 to 0x40**: the information of the mapped application objects. The object describes the content of the PDO by the index, sub-index and length of the mapped object.

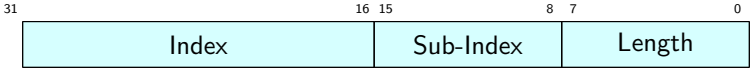


Figure 3.29: Structure of PDO mapping

Note: The MSB is first.
Note: The length is the length of object in bits.

0x160n *RPDO Mapping X*

Index	SubIndex	Name		
0x160n	0	RPDO Mapping X		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT8	RW	2	-	[0;8]

Figure 3.30: Object description 0x160n.0 *RPDO Mapping X*

0x180n TPDO Parameter X

Note: 'n' : common instance of index (0,1,2,3) and 'X' : number of RPDO (1,2,3,4)
Objects involved : 0x1800 *TPDO Parameter 1*, 0x1801 *TPDO Parameter 2*, 0x1802 *TPDO Parameter 3*, 0x1803 *TPDO Parameter 4*

The PDO parameter describes the communication abilities of the PDO.

Index	Sub-index	Name	Data type
TPDO: 0x1800 to 0x1803	0x00	Highest sub-index supported	Unsigned8
	0x01	COB ID	Unsigned32
	0x02	Transmission type	Unsigned8
	0x03	Inhibit time	Unsigned16
	0x04	reserved	Unsigned8
	0x05	Event timer	Unsigned16

Table 3.20: PDO config objects

Index	SubIndex	Name		
0x180n	1	COB ID		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	384	-	[0x00000080;0xFFFFFFFF]

Figure 3.31: Object description 0x180n.1 *COB ID*

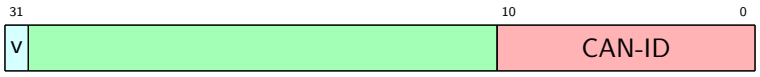


Table 3.21: COB ID

- **v**: valid

Value	Description
0x00	PDO exists / is valid
0x01	PDO does not exist / is not valid

Table 3.22: Description of bit 31: v

- **COB ID** of PDO

0x180n.2 *Transmission Type*

Index	SubIndex	Name		
0x180n	2	Transmission Type		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT8	RW	1	-	[;]

Figure 3.32: Object description 0x180n.2 *Transmission Type*

The transmission of the PDO depends on the configuration of the Transmission Types parameters which can be:

Value	Description
0x00	Synchronous (acyclic)
0x01 to 0xF0	Synchronous cyclic every N SYNC
0xFC	RTR-only (synchronous)
0xFD	RTR-only (event-driven)
0xFE	Event-driven

Table 3.23: Description of TPDO transmission type

- **Synchronous acyclic**: On internal event, the sampling will start and will transmit after the next SYNC.
- **Synchronous**: the TPDO is transmitted after each SYNC received. The sampling of the data will start and will transmit on reception of each the SYNC received.
- **RTR-only (synchronous)**: On RTR (Remote Transmission Request) received, the sampling will start and will transmit after the next SYNC.
- **RTR-only (event-driven)**: On RTR (Remote Transmission Request) received, the sampling start and transmit immediately.
- **Event-driven**: sampling may be transmitted at any time when the internal event happen.

`0x180n.3` ***Inhibit Time*** It's a minimum interval time for PDO transmission. This parameter is available only for the transmission type `0xFE` and `0xFF`. The value is defined as multiple of 100 μs. The value of 0 shall disable the inhibit time. The value shall not be changed while the PDO exists.

Index	SubIndex	Name		
0x180n	3	Inhibit Time		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT16	RW	0	-	[;]

Figure 3.33: Object description `0x180n.3` *Inhibit Time*

`0x180n.6` ***SYNC start value*** The SYNC message of which the counter value equals the SYNC Start value is be regarded as the first received SYNC message. The value of 0 shall disable the SYNC Start value.

Index	SubIndex	Name		
0x180n	6	SYNC start value		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT8	RW	0	-	[;]

Figure 3.34: Object description `0x180n.6` *SYNC start value*

`0x1A0n` TPDO Mapping X

Note: 'n' : common instance of index (0,1,2,3) and 'X' : number of RPDO (1,2,3,4)
Objects involved : `0x1A00` *TPDO Mapping 1*, `0x1A01` *TPDO Mapping 2*, `0x1A02` *TPDO Mapping 3*, `0x1A03` *TPDO Mapping 4*

The PDO mapping parameter contains information about the content of the PDO.

	Sub-index	Name	Data type
0x1A00 to 0x1A03	00	Number of mapped objects in PDO	Unsigned8
	01	1st object to be mapped	Unsigned32
	02	2 nd object to be mapped	Unsigned32
	...	...	Unsigned32
	64	64 th object to be mapped	Unsigned32

Figure 3.35: PDO mapping parameter

- **Sub-index** `0x00`: the number of valid object entries within the mapping record. If it is equal to 0, the Mapping is disabled.

- **Sub-index** 0x01 to 0x40: the information of the mapped application objects. The object describes the content of the PDO by the index, sub-index and length of the mapped object.

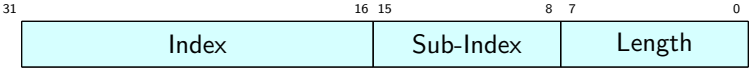


Figure 3.36: Structure of PDO mapping

Note: The MSB is first.
Note: The length is the length of object in bits.

Index	SubIndex	Name		
0x1A0n	0	TPDO Mapping X		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT8	RW	2	-	[0;8]

Figure 3.37: Object description 0x1A0n.0 *TPDO Mapping X*

3.8 Object description

3.8.1 Communication Profile Area object

0x1000 Device Type

This object provide informations about the device type.

Index	SubIndex	Name		
0x1000	0	Device Type		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RO	402	-	[;]

Figure 3.38: Object description 0x1000.0 *Device Type*

It is composed of two field, the device profile and additional information, both on 16 bits.

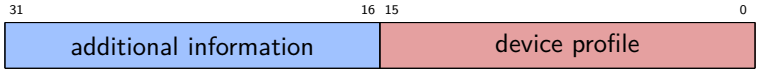


Table 3.24: Frame of Device Type

Device profile		Additional information	
Value	Description	Value	Description
0x191	CiA 401 standard is supported		
0x192	CiA 402 standard is supported		

Table 3.25: Description of device type

0x1008 Manufacturer Device Name

Index	SubIndex	Name		
0x1008	0	Manufacturer Device Name		
Data Type	**Acces**	**Default**	**Unit**	**Range**
VSTRING	RO	UMC1BDS32 motion controller	-	[;]

Figure 3.39: Object description 0x1008.0 *Manufacturer Device Name*

0x1009 Manufacturer Hardware Version

Index	SubIndex	Name		
0x1009	0	Manufacturer Hardware Version		
Data Type	**Acces**	**Default**	**Unit**	**Range**
VSTRING	RO	v1.1.0	-	[;]

Figure 3.40: Object description 0x1009.0 *Manufacturer Hardware Version*

0x1010 Store Parameters

This object stores the value of the parameters according to the Communication profile , Manufacturer profile and Standardized profile or all profiles. The profile areas are defined to Object dictionary.

This functionnality can only work in State PreOp if this is not the case, the device respond with the SDO abort transfer service (SDO abort code: 0x08000021).

Signature To start backup of parameters, a specific signature is required to avoiding wrong manipulation. Specific signature is written in appropriate subindex.

The signature is "save":

3	2	1	0
e	v	a	s
0x65	0x76	0x61	0x73

Figure 3.41: Signature of store

Upon receipt of the correct signature in the appropriate subindex, the device restore the default settings and then confirm the SDO transmission (SDO download initiation response).

If an erroneous signature is written, the device refuse to store the defaults and respond with the SDO abort transfer service (SDO abort code: 0x08000020).

Different possibility to store:

Index	SubIndex	Name		
0x1010	1	Save all Parameters		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	0	-	[;]

Figure 3.42: Object description 0x1010.1 *Save all Parameters*

Index	SubIndex	Name		
0x1010	2	Save Communication Parameters		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	0	-	[;]

Figure 3.43: Object description 0x1010.2 *Save Communication Parameters*

Index	SubIndex	Name		
0x1010	3	Save Standardized Parameters		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	0	-	[;]

Figure 3.44: Object description 0x1010.3 *Save Standardized Parameters*

Index	SubIndex	Name		
0x1010	4	Save Manufacturer Parameters		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	0	-	[;]

Figure 3.45: Object description 0x1010.4 *Save Manufacturer Parameters*

0x1011 Restore Default Parameters

This object restores the factory or saved values of the parameters according to the Communication profile , Manufacturer profile and Standardized profile or all profiles. The profile areas are defined to Object dictionary.

This functionnality can only work in State PreOp if this is not the case, the device respond with the SDO abort transfer service (SDO abort code: 0x08000021).

Signature To start the restoration of parameters, a specific signature is required to avoiding wrong manipulation. Specific signature is written in appropriate subindex.

The signature is "load":

3	2	1	0
d	a	o	l
0x64	0x61	0x6F	0x6C

Figure 3.46: Signature of restore

Upon receipt of the correct signature in the appropriate subindex, the device restore the default settings and then confirm the SDO transmission (SDO download initiation response).

If an erroneous signature is written, the device refuse to restore the defaults and respond with the SDO abort transfer service (SDO abort code: 0x08000020).

Automatic restore This feature determines an automatic restore after a NMT service reset node, NMT service reset communication or power cycled. This functionality is configured with a command (table below) written in the appropriate subindex.

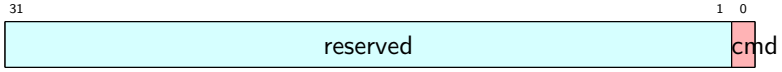


Table 3.26: Automatic restore

Value	Description
0x00	Device don't restore settings automatically
0x01	Device restore settings automatically

Table 3.27: Description of bit 0: cmd

Different possibility to restore:

Index	SubIndex	Name		
0x1011	1	Restore all Factory Parameters		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	0	-	[0;1]

Figure 3.47: Object description 0x1011.1 *Restore all Factory Parameters*

Index	SubIndex	Name		
0x1011	2	Restore Factory Communication Parameters		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	0	-	[0;1]

Figure 3.48: Object description 0x1011.2 *Restore Factory Communication Parameters*

Index	SubIndex	Name		
0x1011	3	Restore Factory Standardized Parameters		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	0	-	[0;1]

Figure 3.49: Object description 0x1011.3 *Restore Factory Standardized Parameters*

Index	SubIndex	Name		
0x1011	4	Restore Factory Manufacturer Parameters		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	0	-	[0;1]

Figure 3.50: Object description 0x1011.4 *Restore Factory Manufacturer Parameters*

Index	SubIndex	Name		
0x1011	5	Restore all saved Parameters		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	1	-	[0;1]

Figure 3.51: Object description 0x1011.5 *Restore all saved Parameters*

Index	SubIndex	Name		
0x1011	6	Restore saved Communication Parameters		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	1	-	[0;1]

Figure 3.52: Object description 0x1011.6 *Restore saved Communication Parameters*

Index	SubIndex	Name		
0x1011	7	Restore saved Standardized Parameters		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	1	-	[0;1]

Figure 3.53: Object description 0x1011.7 *Restore saved Standardized Parameters*

Index	SubIndex	Name		
0x1011	8	Restore saved Manufacturer Parameters		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	1	-	[0;1]

Figure 3.54: Object description 0x1011.8 *Restore saved Manufacturer Parameters*

0x100A Manufacturer Software Version

Index	SubIndex	Name		
0x100A	0	Manufacturer Software Version		
Data Type	**Acces**	**Default**	**Unit**	**Range**
VSTRING	RO	v1.0.2	-	[;]

Figure 3.55: Object description 0x100A.0 *Manufacturer Software Version*

0x100C Guard Time

Index	SubIndex	Name		
0x100C	0	Guard Time		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT16	RW	0	ms	[0;65535]

Figure 3.56: Object description 0x100C.0 *Guard Time*

0x100D Life Time Factor

Index	SubIndex	Name		
0x100D	0	Life Time Factor		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT8	RW	0	-	[0;255]

Figure 3.57: Object description 0x100D.0 *Life Time Factor*

0x1016 Consumer Heartbeat Time

Index	SubIndex	Name		
0x1016	1	Consumer Heartbeat Time		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RW	0	ms	[0x0;0x007FFFFF]

Figure 3.58: Object description 0x1016.1 *Consumer Heartbeat Time*

0x1017 Producer Heartbeat Time

Index	SubIndex	Name		
0x1017	0	Producer Heartbeat Time		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT16	RW	0	ms	[0;65535]

Figure 3.59: Object description 0x1017.0 *Producer Heartbeat Time*

0x1018 Identity Object

Index	SubIndex	Name		
0x1018	1	Vendor Id		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RO	1186	-	[;]

Figure 3.60: Object description 0x1018.1 *Vendor Id*

Index	SubIndex	Name		
0x1018	2	Product Code		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RO	4097	-	[;]

Figure 3.61: Object description 0x1018.2 *Product Code*

Index	SubIndex	Name		
0x1018	3	Revision number		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RO	1	-	[;]

Figure 3.62: Object description 0x1018.3 *Revision number*

Index	SubIndex	Name		
0x1018	4	Serial number		
Data Type	**Acces**	**Default**	**Unit**	**Range**
UINT32	RO	0	-	[;]

Figure 3.63: Object description 0x1018.4 *Serial number*

Appendix A

Firmware version history

Version	Date	Change
1.0.1	2021/10/23	Initial public version

Appendix B

Datasheet revision history

Revision	Date	Change
A	2020-03-20	Initial public revision
B	2021-11-08	Reviewed structure of document