Motion Control Products

Application note
Controlling ACS drives over EtherCAT via CiA402 profile

AN00256

Rev A (EN)

ABB make a wide array of industrial drives that can use a FECA-01 option card to allow control over EtherCAT via the CiA402 drive profile

Introduction

This application note provides an insight into some of the techniques which can be used to easily implement control of one or more ABB industrial (ACS) drives over EtherCAT using the CiA402 drive profile.

The code that accompanies this application note is targeted at a motion capable AC500 PLC with a CM579-ETHCAT coupler and an ACS380 Machinery drive with a FECA-01 EtherCAT adapter fitted. The drive parameter backup as well as an optional CP600 HMI control program are embedded in the example project should you wish to expand the scope of the program operation.

Though this example discusses the ACS380, any drive that is compatible with the FECA-01 can use the methods and control techniques described. These drives include ACS380, ACS580, and ACS880. These drives are typically used in speed or torque control modes with asynchronous or permanent magnet motors which can be run either open loop (sensor less) or closed loop using optional (FEN-xx, MTAC-xx or BTAC-xx) feedback modules.

A lot of the general steps for configuring and adding objects to an Automation builder project are covered in the following application note; http://www.abbmotion.com/support/SupportMe/Downloads/DocsLib/AN00205-005.zip

Additional support and .xml files can be sourced from the FECA-01 home page; http://new.abb.com/drives/connectivity/fieldbus-connectivity/EtherCAT/EtherCAT-feca-01

The ACS drives support both the CiA402 drive profile and a manufacturer (ABB) specific drive profile and the AC500 PLC provides function blocks for both of these. However, it is more typical to use the ABB profile and the dedicated ACS function blocks when using AC500 as these are easier to use than the more generic CiA402 features (see application note AN00255 for further details and sample project for use of the ABB drives profile). If a third-party EtherCAT master device is used to control the ACS drives then this is a situation where the CiA402 drive profile would be selected instead. This is the situation we will detail in this application note, but we will use the AC500 PLC to illustrate this.

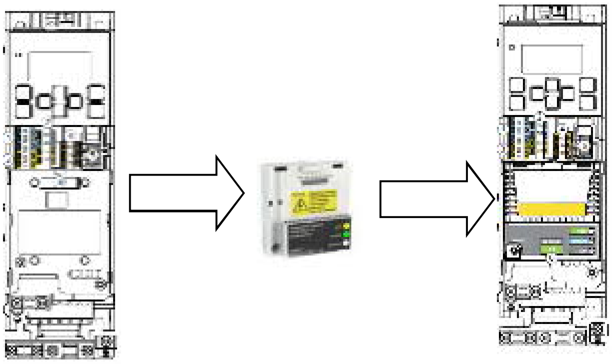
Pre-requisites

You will need to have the following to work through this application note:

- An ACS380 with firmware AMCK6 v1.73.8.0 or later (see parameter 07.05) and FECA-01 EtherCAT adaptor with firmware version 1.30 or later (or the included files are easily adapted to suit other ACS series drives).
- A PC or laptop running Automation Builder 1.2 with at least a standard license (DM100-TOOL).
- A copy of Drive Composer Pro 1.12 or later.
- An AC500 PM585 or PM59x-ETH PLC with CM579-ECAT communication module (CM579-ECAT module must be running firmware version 4.3.0.2 or later) or a PM595 PLC with integrated EtherCAT coupler
- A working knowledge of the basic operation of the AC500 PLC and ACS380 drive

Control hardware

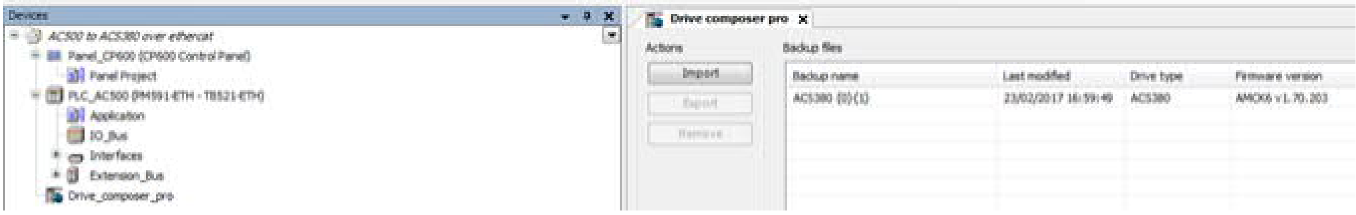
Not all ABB ACS drives support EtherCAT as standard and as such a FECA-01 needs to be added to the 'base variant'. This can be done either by adding a '+ code' at point of order (+K469 FECA-01-M Preconfigured EtherCAT protocol) or retro-fitting a FECA-01 after the drive is delivered. If you do the latter the FECA-01 must be hardware revision J or later.



If the drive is ordered with the + K469 option then all the relevant settings should already be made within the module. If it is retro fitted after the drive has been ordered the parameters must be set up as required.

Drive parameters

In the Drive Composer Pro section of Automation Builder you can see there is a saved backup of ACS380 settings that will configure the drive to operate via EtherCAT using the CiA402 drive profile;



Devices

- AC500 to ACS380 over ethercat
  - Panel_CP600 (CP600 Control Panel)
    - Panel Project
  - PLC_AC500 (PM591-ETH - TB521-ETH)
    - Application
    - IO_Bus
    - Interfaces
    - Extension_Bus
  - Drive_composer_pro

Backup name	Last modified	Drive type	Firmware version
ACS380 (0)(1)	23/02/2017 16:59:49	ACS380	AMCK6 v1.70.203

If we wanted to set up a blank drive manually we would carry out the steps below;

- Start by adding the FECA-01 module
- Configure the drive to accept it by setting **50.01 FBA A enable** to **'Enable' (1)**
- This adapter in the ACS380 will always be called FBA A (though in ACS880 drives the FECA-01 could be installed in 'Slot A' or 'Slot B' and so would be FBA-A or FBA-B). The adapter type should then read the module identifier and **51.01 FBA A** type should change automatically to **EtherCAT (135)**
- Set ***50.04 FBA A ref1 type*** as required (Speed or Frequency are the only options for ACS380).This parameter selects the type and scaling of reference 1 received from fieldbus adapter A. In this instance this needs to be set to **Speed** (4) although a value of 0 could also be used as this sets the mode automatically depending on the drive's currently active operating mode. The scaling is defined by parameter ***46.01 Speed scaling*** which sets the maximum speed value (in our example we have left this set to the default of 1500rpm)
- Ensure the drive profile parameter **51.02** is set to **CiA402 (0)** - the alternative setting is **ABB Drives / 1** which is discussed in application note AN00255. Once set then **51.27** should be set to **Refresh** while online and the drive will henceforth operate with the CiA402 drive profile

Now the drive is ready to communicate with the PLC it must be configured to accept the signals to control it;

- Firstly the drive must be configured to ensure it is being controlled from the correct 'External Control Location'. There is a way to configure whether a drive can be controlled from single or multiple external locations (Ext 1 and Ext 2). By setting **19.11 Ext1/Ext2 Sel = EXT1 (0)** the drive will only use EXT1 as a control source (and in turn we will configure EXT1 to be a fieldbus)

- The Ext1 control command location is defined by parameter **20.01 Ext1 commands,** this can be set to accept commands from the control panel, digital inputs or fieldbus. To use the fieldbus adapter we must set this to **Fieldbus A (12)**
- When the start and stop is controlled via a fieldbus we must also set **20.02 Ext1 start trigger type** to **Level (1)**
- The profile being used by EXT1 can control the drive in either Speed or Torque mode. Setting **19.12 Ext1 control mode** to **Speed (2)** sets the drive in **velocity** control mode
- The source for the EXT1 demand/reference is defined by **22.11**. To use Ref 1 as sent by the fieldbus adapter A then we need to set **22.11 Ext1 speed ref 1** to **FBA Ref 1 (4)**

Once all of the relevant settings have been made the drive is configured and can now be used with the AC500 PLC.

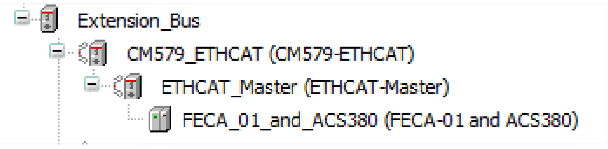
Configuring Automation builder hardware

In case you don't have the target files as part of your Automation builder installation, or if you want to ensure you have the most up to date versions you can download the zip file from the link below (or from the FECA-01 home page);

https://search.abb.com/library/Download.aspx?DocumentID=9AKK105152A4454&LanguageCode=en&DocumentPartId=&Action=Launch

Then extract the zip file and in Automation Builder (via Tools > Device repository > Install) add the downloaded file**; FECA-01_1.30_ACS380_v2.00.0.4.xml** to the repository.

Add a motion capable AC500 CPU to the project, then add a CM579_ETHCAT to Slot 1 of your configuration and add to the ETHCAT Master section a 'FECA-01 and ACS380'. Your hardware tree will look something like this;


Extension_Bus
CM579_ETHCAT (CM579-ETHCAT)
ETHCAT_Master (ETHCAT-Master)
FECA_01_and_ACS380 (FECA-01 and ACS380)

Note that only the ACSM1 and the motion variant of ACS880 (ACS880-M04) drives support synchronisation over EtherCAT (i.e. operation in either SM [Synchronous Mode] or DC [Distributed Clock] modes), so for the other drives in the ACS series which operate in Asynchronous / Free run mode (such as the ACS380 we are using in this example) there is no need to configure any further network settings relating to the FECA module other than adding Process Data Object mappings for the data to be exchanged with the drive (the EtherCAT master cycle time would be set to suit any other slave devices that do require synchronisation, such as a MicroFlex e190 servo drive for example).

Understanding mapping objects and Process Data Objects

To communicate with any EtherCAT node on a cyclic basis we can configure data to be exchanged in both directions. This data is known as Process Data Objects (or PDOs). Within the FECA-01 .xml file there are default sets of PDO's available called data objects maps. These can be found in the **Automation Builder > Extension Bus > FECA-01 configuration > Process Data** tab. There are data mapping objects for both PLC outputs (drive RxPDO) and PLC inputs (drive TxPDO) which contain certain default PDO data elements. These can also be edited by enabling 'Expert settings' and adding or removing other PDO's in the **Expert Process Data** tab if needed.

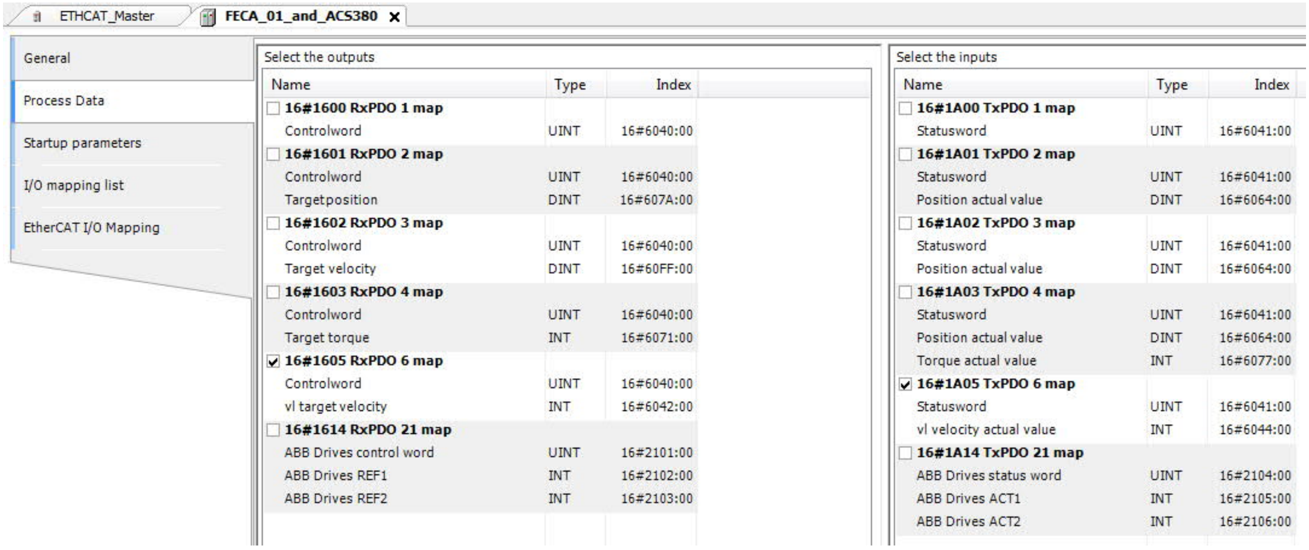
These PDO sets have been created to simplify configuration of the drive to suit pre-defined operation modes and to suit whether the drive is being used in conjunction with the ABB drive profile or the CiA402 drive profile.

Each of the output PDO sets contains a Controlword mapping and each of the input PDO sets contains a Statusword mapping. The other PDOs included with each PDO set depend on which drive profile is used and how the drive is to operate (e.g. RxPDO 3 contains a Target Velocity PDO mapping which might be used if an ACSM1 motion variant drive was operating with the CiA402 drive profile in cyclic synchronous velocity mode)…

In this application note we are considering operation with the CiA402 drive profile (please refer to AN00255 for further information about the available PDO sets and how these are used in conjunction with the ABB drive profile). When using the CiA402 drive profile the ACS drives access data from specific objects defined by the CiA402 profile for the control word, status word and references/targets (objects 0x6040 and 0x6041 for the control word and status word respectively for example). The following table details the possible operating modes for ACS drives when controlled using the CiA402 profile…

Operation mode	**ACS355**	**ACS380**	**ACS580**	**ACS850**	**ACS880 speed**	**ACS880 motion**	**ACSM1 speed**	**ACSM1 motion**
Velocity	✓	✓	✓	✓	✓	✓	✓	✓
Profile torque	✓	✓	✓	✓	✓	✓	✓	✓
Profile velocity	✓	✓	✓	✓	✓	✓	✓	✓
Profile position	✗	✗	✗	✗	✗	✓	✗	✓
Homing	✗	✗	✗	✗	✗	✓	✗	✓
Cyclic synchronous torque	✓	✓	✓	✓	✓	✓	✓	✓
Cyclic synchronous velocity	✗	✗	✗	✓	✓	✓	✓	✓
Cyclic synchronous position	✗	✗	✗	✗	✗	✓	✗	✓

The PDO data set that you should select depends on the operating mode the drive is to be used in. In this example we will operate the drive in "Velocity mode" (CiA402 defines this as operating mode 2) so the most appropriate PDO set for this is 6. The screenshot below shows how we select this PDO data set from the Process Data tab in Automation Builder…



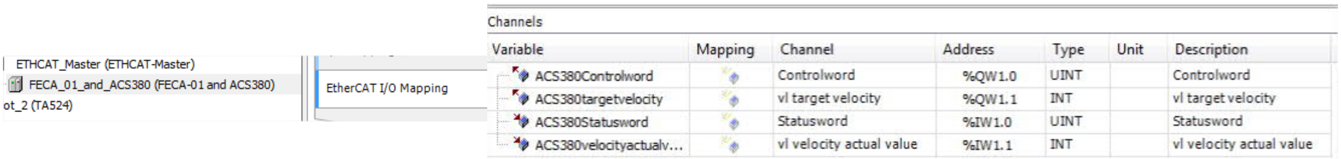
Select the outputs

Name	Type	Index
□ 16#1600 RxPDO 1 map		
Controlword	UINT	16#6040:00
□ 16#1601 RxPDO 2 map		
Controlword	UINT	16#6040:00
Targetposition	DINT	16#607A:00
□ 16#1602 RxPDO 3 map		
Controlword	UINT	16#6040:00
Target velocity	DINT	16#60FF:00
□ 16#1603 RxPDO 4 map		
Controlword	UINT	16#6040:00
Target torque	INT	16#6071:00
☑ 16#1605 RxPDO 6 map		
Controlword	UINT	16#6040:00
vl target velocity	INT	16#6042:00
□ 16#1614 RxPDO 21 map		
ABB Drives control word	UINT	16#2101:00
ABB Drives REF1	INT	16#2102:00
ABB Drives REF2	INT	16#2103:00

Select the inputs

Name	Type	Index
□ 16#1A00 TxPDO 1 map		
Statusword	UINT	16#6041:00
□ 16#1A01 TxPDO 2 map		
Statusword	UINT	16#6041:00
Position actual value	DINT	16#6064:00
□ 16#1A02 TxPDO 3 map		
Statusword	UINT	16#6041:00
Position actual value	DINT	16#6064:00
□ 16#1A03 TxPDO 4 map		
Statusword	UINT	16#6041:00
Position actual value	DINT	16#6064:00
Torque actual value	INT	16#6077:00
☑ 16#1A05 TxPDO 6 map		
Statusword	UINT	16#6041:00
vl velocity actual value	INT	16#6044:00
□ 16#1A14 TxPDO 21 map		
ABB Drives status word	UINT	16#2104:00
ABB Drives ACT1	INT	16#2105:00
ABB Drives ACT2	INT	16#2106:00

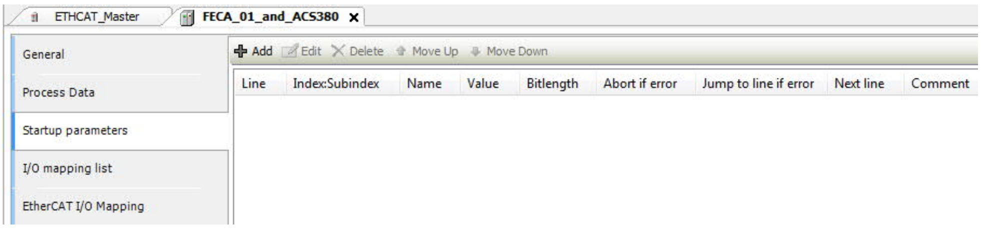
After these selections have been made we can see that the variables appear in the EtherCAT I/O mapping Tab. In here we can enter variable names for all the mapped variables that will appear in the IEC61131-3 programming environment;


ETHCAT_Master (ETHCAT-Master)
FECA_01_and_ACS380 (FECA-01 and ACS380)
ot_2 (TA524)
EtherCAT I/O Mapping

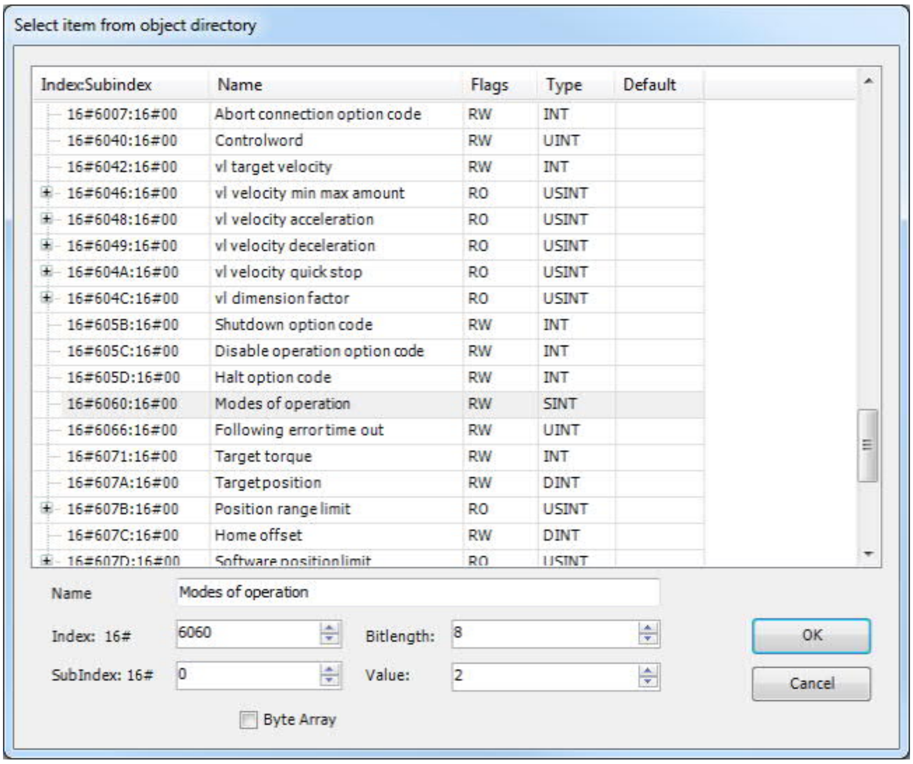
Channels

Variable	Mapping	Channel	Address	Type	Unit	Description
ACS380Controlword		Controlword	%QW1.0	UINT		Controlword
ACS380targetvelocity		vl target velocity	%QW1.1	INT		vl target velocity
ACS380Statusword		Statusword	%IW1.0	UINT		Statusword
ACS380velocityactualv…		vl velocity actual value	%IW1.1	INT		vl velocity actual value

Although we have selected a PDO data set suitable for the drive to operate in "Velocity mode" we also need to ensure the drive starts in this mode when the EtherCAT network becomes operational. This is done by adding a start-up command to the Automation Builder configuration that sets the drive's "Operating Mode" object to the required value. Select the FECA module in the device tree and in the right hand pane select the "Startup parameters" tab…



Click on the “Add” button to show a dialog that lists all of the available objects on the FECA module. Scroll through this dialog and find object 16#6060 (the CiA402 Modes of Operation object). Select this object and enter a value of 2 for this (this is the value that corresponds to “Velocity mode”)…


Select item from object directory

Index:Subindex	Name	Flags	Type	Default
16#6007:16#00	Abort connection option code	RW	INT	
16#6040:16#00	Controlword	RW	UINT	
16#6042:16#00	vl target velocity	RW	INT	
16#6046:16#00	vl velocity min max amount	RO	USINT	
16#6048:16#00	vl velocity acceleration	RO	USINT	
16#6049:16#00	vl velocity deceleration	RO	USINT	
16#604A:16#00	vl velocity quick stop	RO	USINT	
16#604C:16#00	vl dimension factor	RO	USINT	
16#605B:16#00	Shutdown option code	RW	INT	
16#605C:16#00	Disable operation option code	RW	INT	
16#605D:16#00	Halt option code	RW	INT	
16#6060:16#00	Modes of operation	RW	SINT	
16#6066:16#00	Following error time out	RW	UINT	
16#6071:16#00	Target torque	RW	INT	
16#607A:16#00	Target position	RW	DINT	
16#607B:16#00	Position range limit	RO	USINT	
16#607C:16#00	Home offset	RW	DINT	
16#607D:16#00	Software position limit	RO	USINT	

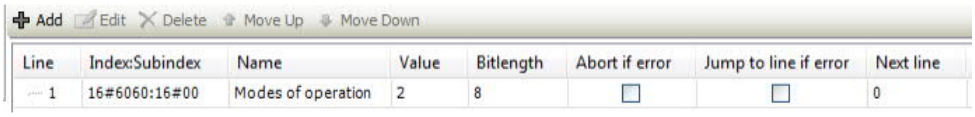
Name: Modes of operation

Index: 16# 6060 Bitlength: 8

SubIndex: 16# 0 Value: 2

□ Byte Array

Click ‘OK’ and this start-up command will be added to the Startup parameters tab.



Line	Index:Subindex	Name	Value	Bitlength	Abort if error	Jump to line if error	Next line
1	16#6060:16#00	Modes of operation	2	8	□	□	0

Adding this command to the Startup parameters is the easiest way to configure the drive to operate in Velocity mode, but it is also possible to use a SDO command (i.e. asynchronous write to the same 16#6060 object) in the PLC program (this might be necessary if the application needs to switch between different operating modes for example). Please refer to application note AN00242 for further information about the use of SDO commands via EtherCAT.

The table below lists all of the possible CiA402 Modes of Operation and their associated values. Remember to check whether the drive you are using supports a specific mode (as shown in our earlier table on page 4 of this application note)…

Value	Mode of Operation
-128 to -1	Manufacturer specific operation mode (not used)
0	No mode assigned
1	Profile position
2	Velocity
3	Profile velocity
4	Profile torque
5	Reserved
6	Homing
7	Interpolated position

8	Cyclic synchronous position
9	Cyclic synchronous velocity
10	Cyclic synchronous torque
11 to 127	Reserved

It should also be noted that parameter 99.04 on our ACS380 also affects the operation mode that can be used:

99.04 = VECTOR: TORQ: Operation modes can be; Profile Torque, Cyclic synchronous torque.
99.04 = VECTOR: SPEED: Operation modes can be; Velocity, Profile velocity
99.04 = SCALAR: FREQ: Operation mode can be; Velocity

Controlling the drive using the CiA402 profile

Now we have mapped the PDO data necessary for control and monitoring of our drive and set the required operation mode we are ready to start creating PLC logic to manipulate this data.

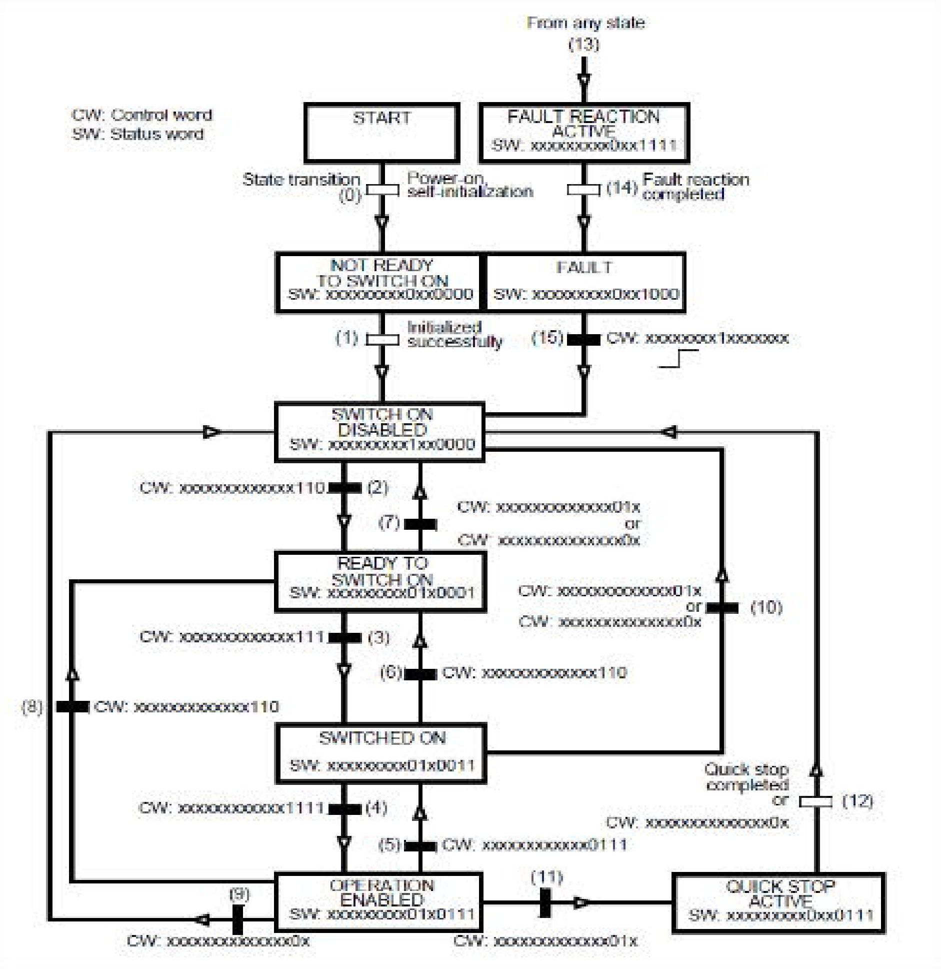
CiA402 profile Control word and Status word
The PS553-DRIVES library does not provide any 'ready-made' function blocks for use with the CiA402 profile (if using an ABB PLC you are likely to use the ABB drive profile and the function blocks provided by the PS553-DRIVES library instead – please refer to application note AN00255 for further details). Instead the user must create their own functions to manipulate the CiA402 control word as necessary. The example AC500 project included with this application note contains a function block to illustrate how this may be achieved.

The CiA402 profile Control and Status word structures when the drive is operating in Velocity mode (as per our example) are shown in the tables below…

CiA402 Control word		
Bit	Description	Comment
0	Switch on	
1	Enable voltage	
2	Quick stop	
3	Enable operation	
=== 4…6 Operation mode specific ===		
4	Ramp function generator enable	When in Velocity Mode
5	Ramp function generator unlock	When in Velocity Mode
6	Ramp function generator use ref.	When in Velocity Mode
7	Fault reset	
8	Halt	

CiA402 Status word		
Bit	Description	Comment
0	Ready to switch on	
1	Switched on	
2	Operation enabled	
3	Fault	
4	Voltage enabled	
5	Quick stop	
6		
7	Warning	
8	Drive-specific	
9	Remote	
10	Target reached	
11	Internal limit active	
=== 12 and 13 Operation mode dependent ===		
12	No Function	When in Velocity mode
13	No Function	When in Velocity mode

To control the drive using the CiA402 profile requires that the Control word is transitioned through various states according to the requirements of the CiA402 profile state machine. This is illustrated below…



CW = Control word, SW = Status word, 0 = bit must be 0, 1 = bit must be 1, x = bit can be 0 or 1

The table below details the possible drive states:

State	Description	Status word
Not ready to switch on	Drive is initialising and therefore cannot be enabled. EtherCAT communication is active	Bits 0-3,6 = 0
Switch on disabled	Initialisation complete. Drive is not enabled at this point	Bits 0-3 = 0, bit 6 = 1
Ready to switch on	Mains voltage may be applied to the drive but at this point it still is not enabled (there is no requirement for Mains voltage at this point)	Bits 1-3,6 = 0, bits 0,5 = 1
Switched on	Mains voltage is applied, power amplifier is ready, drive is not yet enabled	Bits 2,3,6 = 0, bits 0,1,5 = 1
Operation enabled	Power applied to motor, no faults, parameters may be changed	Bits 3,6 = 0, bits 0,1,2,5 = 1
Fault reaction active	Non-fatal fault has occurred in the drive. The Quick Stop function is being executed and during this time the drive function is enabled and power is applied to the motor	Bit 6 = 0, bits 0-3 = 1
Fault	A fault has occurred in the drive and the drive is not (and cannot be) enabled	Bits 0-2,6 = 0, bit 3 = 1
Quick stop active	The Quick Stop function is being executed. Drive function is enabled and power is applied to the motor	Bits 3,6 = 0, bits 0-2 = 1

CiA402 profile references

The CiA402 profile variable ‘**vl target velocity**’, object 16#6042:00 (which is copied to REF1) is a 16-bit word containing a sign bit and a 15-bit integer. As we configured our drive for velocity mode this is the value that will be sent to the drive to command its velocity. The reference values are scaled directly in rpm.

Actual values

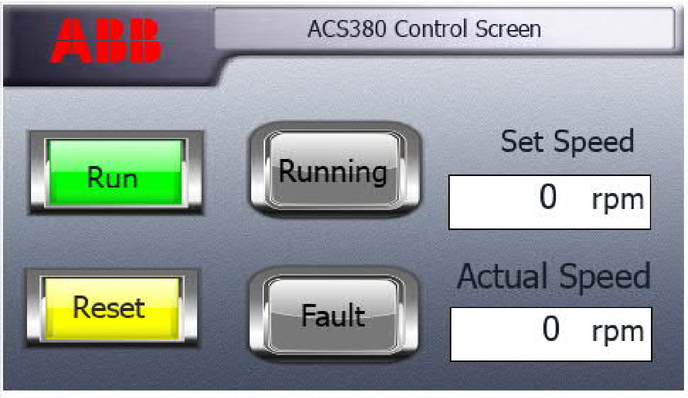
The CiA402 profile variable '**vl velocity actual value**', object 16#6044:00 (which is copied from ACT1) is a 16-bit word containing a sign bit and a 15-bit integer. The actual values are scaled directly in rpm.

In the example project included with this application note there is no scaling of these values required as reference values are written to, and read from, the PDO variables directly with no special scaling needed.

Example AC500 control program

If you want to test the above functionality or look at the logic to control the drive using the CiA 402 profile then please consider the AC500 code attached as a guideline. This contains a function block which will cycle through the state machine as described previously when given the commands to Run, Reset or issue a new command velocity the drive.

If you want to use the CP600 program to interact with the example AC500 project you can use it to run and reset the drive as well as setting a speed reference by clicking on the set speed box and entering a new value. The HMI will also report back on the drive state via the Running and Fault lights and display the actual speed.


ACS380 Control Screen

Set Speed 0 rpm

Actual Speed 0 rpm

Contact us

For more information please contact your
Local ABB representative or one of the following:

new.abb.com/motion
new.abb.com/drives
new.abb.com/drivespartners
new.abb.com/PLC