



Buzibachstrasse 31
6023 Rothenburg
Switzerland

Phone +41(0)41 541 50
40
info@zub.ch
www.zub.ch

ApossIDE

ApossC Commands

Version 7.00.102

Address zub machine control AG
Buzibachstrasse 31
CH-6023 Rothenburg
Switzerland
Phone +41-41-541 50 40
Fax +41-41-541 50 49
<http://www.zub.ch>
<http://www.aposs.ch>
info@zub.ch

Copyright © zub machine control AG

zub machine control AG reserves the right to make modifications to the software, documentation, and product as it sees fit without prior or subsequent notice to users.

No part of this publication may be reproduced or distributed in any form or by any means (e.g. photocopy, microfilm, electronic data exchange) or stored in a database or retrieval system, without the prior written permission of zub machine control AG.

While every precaution has been taken in the preparation of this manual, zub machine control AG assumes no responsibility for errors or omissions, or for damages resulting from the use of the information contained herein.

Trademark Many of the designations that are used by manufacturers and sellers to distinguish their products are claimed as trademarks. We point out that such software and hardware designations, brand names, and trademarks used in this publication are protected by law.

In particular, this includes the following:

APOSS is a registered trademark of zub machine control AG.

Microsoft, Windows 7, Windows 10 and Windows 11 are either registered trademarks or trademarks of the Microsoft Corporation in the USA and other countries.

None of the registered trademarks are marked in this publication. This, and the fact that the "R" symbol is missing, does not mean that the trademark is a free trademark.

Table of Contents

Imprint	1
Command Reference	3-4
Amplifier Commands	5
Axis and Information Commands	6-7
Communication CAN Basic Commands	8
Communication CANopen Commands	9
Communication EtherCAT Master Commands	10
Communication Serial Commands	11
Ethernet Commands	12-13
Data Recording Commands	14
Input/Output Commands	15
Interrupt Functions	16
Kinematics Commands	17-18
Memory Commands	19
State Machine Commands	20
Synchronization Commands	21
System Commands	22

Command Reference



In all of the reference pages that describe commands that refer to axes, the syntax descriptions will include one of the following:

(axis,...)

(axis, value,...)

"..." indicates that the command will accept multiple axes. If a "value" is required, then the "axis" and "value" parameters *must* be specified in *pairs* and each "value" will correspond to the preceding "axis". For these commands, it is also possible to specify the keyword **AXALL** for "axis" and specify a *single* "value" (if required). If a "value" is specified, then the same value is used for all axes. For example:

Syntax: AxisPosAbsStart(axis, position,...);

```
// Move axis 0 to 1000.
```

```
AxisPosAbsStart(0,1000);
```

```
// Move axis 2 to 0 and move axis 1 to 1000.
```

```
AxisPosAbsStart(2,0,1,1000);
```

```
// Move all axes to 1000.
```

```
AxisPosAbsStart(AXALL,1000);
```

(axis)

(axis, value)

If the command syntax does not include "...", then it will accept only a single axis. If any other parameters are required, then they will apply only to the specified axis. For example:

Syntax: res = Apos(axis);

```
pos = Apos(0);    // Position of axis 0.
```

All references to axes in ApossC, are made using **axis module numbers**. Axis modules numbers are **0-based**, so the first axis will be axis module 0, the second will be module 1, and so on.

All available commands are organized below into general functional groups:

Amplifier	Command concerning the internal or external power stage
Axis and Information	Commands to control the axis or get information of it
Communication (CAN Basic)	Commands to send and receive CAN messages
Communication (CANopen)	Commands to send and receive CANopen SDOs and PDOs
Communication (EtherCAT)	Commands to configure and control the EtherCAT masters.
Communication (Serial)	Commands to handle the RS485 and RS233 interfaces.
Communication (Ethernet)	Commands to handle the Ethernet interface.

Data Recording	Commands to set speed and velocity, to start, stop, and to position.
Input/Output	Commands to set and read inputs and outputs.
Interrupt Functions	Commands to manage interrupts.
Kinematics	Commands for multi-axis control using selected robot kinematics.
Memory	Commands for data transfer and for saving of permanent data
State Machine	Commands related to state machines.
Synchronization	Commands to synchronize a slave with a master.
System	Various system commands
Debug	Various debug commands

Amplifier Commands

Commands for the internal amplifier

AmpErrorClear	Clears an error in the amplifier
HallDetectStart	Starts the hall detection procedure
HallDetectStatus	Polls the hall detection procedure status
MotorAlignStart	Starts the motor alignment procedure
MotorAlignStatus	Polls the motor alignment procedure status

Axis and Information Commands

Commands to initialize the axes and the control unit start up and define the zero point(s):

Acc	Set the acceleration for motion commands.
Apos	Query the current actual position of an axis.
AposDiff	Overflow handling of incremental encoders in applications.
Avel	Query the actual velocity of axis.
AxisControl	Select the axis control (ON, OFF ...).
AxisCvelStart	Start constant velocity mode.
AxisCvelStop	Stops the drive in constant velocity mode.
AxisHomeStart	Starts the integrated homing function.
AxisHomeStop	Stops (aborts) the integrated homing function
AxisHomeStatus	Returns the state of the integrated homing function.
AxisLinAbsStart	Simultaneous absolute positioning of several axes.
AxisLinRelStart	Simultaneous relative positioning of several axes.
AxisMoving	Test whether or not an axis is moving.
AxisPosAbsStart	Positioning in relation to actual zero point.
AxisPosAbsStartTvel	Position to the specified target point, ending with a specified target velocity.
AxisPosRelStart	Positioning in relation to actual position.
AxisStatus	Query axis and control status.
AxisStatusClear	Clear the slave and master index flags.
AxisStop	Stop any axis movement with Dec ramp.
AxisTargetReached	Test whether or not a controlled movement is active.
AxisWaitReached	Wait until the target position is reached.
Cpos	Query the current command position of an axis.
CposDiff	Overflow handling of incremental encoders in applications.
CurvePos	Retrieve slave curve position that corresponds to the current master position of the curve.
Cvel	Set the velocity for constant velocity controlled motor movements.
Dec	Set the deceleration for motion commands.
DefCposOrigin	Set the command position as zero point.

DefMasterCpos	Define initial position of the master.
DefMasterOrigin	Set the current master position as the zero point for the master.
DefOrigin	Set the current position as zero point.
Ipos	Query the last index or marker position of the slave.
IposDiff	Overflow handling of incremental encoders in applications.
JerkFinVel	Calculate the final velocity for a jerk-limited stop with maximum acceleration/deceleration.
JerkStopDist	Calculate the necessary distance for a jerk-limited stop with maximum deceleration.
Mapos	Query the current actual position of the master.
MaposDiff	Overflow handling of incremental encoders in applications.
Mavel	Query the actual velocity of the master.
Mipos	Query last index or marker position of the master.
MiposDiff	Overflow handling of incremental encoders in applications.
ResetOrigin	Erase temporary zero point.
SetMasterOrigin	Set any position as the zero point for the master.
SetOrigin	Set absolute position as temporary zero point.
TrackError	Query the the actual tracking error of an axis.
Tvel	Set the target velocity for the AxisPosAbsStartTvel command.

Communication CAN Basic Commands

Commands for executing basic CAN functions:

CanDelete	Erases all or single CAN objects.
CanIn	Reads an object.
CanInit	Initializes the necessary objects (PDOs) for data exchange of CANopen nodes or enables extended CANINIT, CANIN function.
CanOpenRestart	Initializes the CANopen functionality.
CanOut	Sends message (active).
DefCanIn	Defines a receive object.
DefCanOut	Defines a transmit object in the CAN controller.
CANINPUT	Calls an interrupt handler routine when a CAN telegram arrives.

Communication CANopen Commands

All global and axis parameters with a parameter code can be set and read with these commands:

LinkPdo	Link the system variable with PDO and copy in the internal parameters.
LinkSdo	Copying an internal object in the PDO.
SdoRead	Reads SDO of a connected CANopen device.
SdoReadSeg	Segmented read of SDOs (unpacked).
SdoReadSegp	Segmented read of SDOs (packed).
SdoStatus	Checks the result of an active communication.
SdoWrite	Sets an SDO of a connected CANopen device and uses the data exchange without defined number of valid data bytes (= "Data set size is not indicated").
SdoWriten	Sets an SDO and specifies the number of valid data bytes (= "Data set size is indicated").
SdoWriteSeg	Segmented write of SDOs (only for EtherCAT master).

Communication EtherCAT Master Commands

Commands for handling an EtherCAT master:

ECatMasterCommand	Controls the EtherCAT master.
ECatMasterConfig	Configures the EtherCAT master.
ECatMasterInfo	Reads information from the EtherCAT master.

CAN over EtherCAT CoE is also supported. Therefore, any kind of SDO communication works also for the EtherCAT master.

Please check the list: [Communication CANopen Commands](#)

Communication Serial Commands

Commands for handling serial communication:

SerialSetup	Initializes the serial interface handler.
SerialReadComm	Checks for a completely received telegram on the serial interface.
SerialWriteComm	Sends a complete telegram over the serial interface.
SerialReadChar	Reads a single character from the serial interface.
print / printcontrol	The print command can also be used to print to the serial interface (see sample ISR_Keypressed.mc)

Ethernet Commands

Ethernet commands:

EthernetGetConnectionStatus

Get the current ethernet connection status.

EthernetOpenClient

Opens a socket connection to a socket server..

EthernetOpenServer

Opens a socket server, which waits for a connection from a client.

EthernetReceiveTelegram

Receives a telegram if there is one in the buffer.

EthernetSendTelegram

Sends a telegram.

EthernetClose

Close a socket connection.

General return values

ETHSOCK_RETVAL_OK = 0, // No error, everything OK

LWIP_ERR_MEM = -1, // Out of memory error.

LWIP_ERR_BUF = -2, // Buffer error.

LWIP_ERR_TIMEOUT = -3, // Timeout.

LWIP_ERR_RTE = -4, // Routing problem.

LWIP_ERR_INPROGRESS = -5, // Operation in progress

LWIP_ERR_VAL = -6, // Illegal value.

LWIP_ERR_WOULDBLOCK = -7, // Operation would block.

LWIP_ERR_USE = -8, // Address in use.

LWIP_ERR_ISCONN = -9, // Already connected.

LWIP_ERR_ABRT = -10, // Connection aborted.

LWIP_ERR_RST = -11, // Connection reset.

LWIP_ERR_CLSD = -12, // Connection closed.

LWIP_ERR_CONN = -13, // Not connected.

LWIP_ERR_ARG = -14, // Illegal argument.

LWIP_ERR_IF = -15, // Low-level netif error

ETHSOCK_RETVAL_INVALIDHANDLE = -21, // Invalid handle has been passed

ETHSOCK_RETVAL_NOFREESOCKETS = -22, // Limit of socket connection exceeded

ETHSOCK_RETVAL_NOTCONNECTED = -23, // The socket is not connected

ETHSOCK_RETVAL_TGTOOLONG = -24, // Too much data has been attempted to send

ETHSOCK_RETVAL_TXOVERFLOW = -25, // Previous data not yet sent

ETHSOCK_RETVAL_RXOVERFLOW = -26, // Previously received data has not been read and new data has arrived

ETHSOCK_RETVAL_RXTOOLONG = -27, // Received telegram is too long for the internal buffer

ETHSOCK_RETVAL_NOTSUPPORT = -28, // This controller does not support the socket feature

ETHSOCK_RETVAL_PROTNOTSUPPORT = -29 // The requested protocol type is not supported

Data Recording Commands

Commands for configuring and using arrays for recording test results:

RecordDest	Defines the memory section for saving recorded data.
RecordIndex	Specifies system variables for data recording.
RecordTime	Defines the sample interval (i.e. the amount of cycles between samples).
RecordTimeBase	(Optional) Defines the time base (i.e. the cycle time).
RecordType	Defines "one time" or "cyclic" recording.
RecordStart	Starts the data recording.
RecordStop	Stops the data recording.

Input/Output Commands

Commands to set and read inputs and outputs:

AnalogInput	Reads analog input and process data objects (PDO) of CAN objects.
AnalogOutput	Sets analog output (freely available) and an output, which is initialized as a drive.
DigInput	Reads input bit-by-bit (individually).
DigInputByte	Reads input by bytes (units of 8).
DigOutput	Sets digital outputs bit-by-bit (single).
DigOutputByte	Sets digital outputs by bytes (units of 8).

Interrupt Functions

Commands for managing interrupts:

- InterruptSetup** Sets up interrupt handler functions.
- InterruptDelete** Deletes interrupt handler functions.
- InterruptEnable** Enables interrupt handling.
- InterruptDisable** Disables interrupt handling.

Kinematics Commands

Commands for multi-axis control using selected kinematics:

KinematicsCpoint	Query the current kinematics position.
KinematicsWorkToMc	Set a work-to-machine transformation that will be used with the selected kinematics.
KinematicsSetup	Select the multi-axis kinematics model to be used with a robot.
PathAcc	Set the acceleration to be used when following a path.
PathContinue	Continue following a path that has been paused.
PathDec	Set the deceleration to be used when following a path.
PathJerk	Set the limited-jerk values to be used when following a path.
PathMcToUu	Convert machine coordinates to axis coordinates.
PathMove	Move to a specified point, using the selected kinematics.
PathNext	Set the next multi-axis path curve to be used when the current path ends.
PathStart	Start following a multi-axis path curve, based on the selected kinematics.
PathStop	Stop a running path movement.
PathUuToMc	Convert axis coordinates to machine coordinates.
PathVel	Set the velocity to be used when following a path.
TransIdent	Set a transformation to the identity transformation (i.e. no transformation).
TransInverse	Transform a dpoint variable by the inverse of a transformation.
TransPoint	Transform a dpoint variable by a transformation.
TransReflect	Reflect a transformation across the YZ plane (i.e. $X = 0$).
TransRotateXY	Rotate a transformation in the XY plane (i.e. about the Z axis).
TransRotateYZ	Rotate a transformation in the YZ plane (i.e. about the X axis).
TransRotateZX	Rotate a transformation in the ZX plane (i.e. about the Y axis).
TransScale	Scale a transformation.
TransStretch	Stretch a transformation.

TransTranslate

Translate a transformation.

Commands to access elements within a Path array:

PathArrayIndex

Determine the array index of various data blocks within a Path curve array.

Memory Commands

Command for managing controller memory:

Delete	Delete all arrays in RAM defined by the DIM command.
MemoryDump	Save SDO data on the Micro-SD card.
MemoryDumpStatus	This command can be used to poll the state/result of a MemoryDump command.
Save	Save DIM arrays, axis parameters, global parameters, or user parameters in permanent memory.
DimArray	Get the reference of the DIM array from its number.
DimArrayNum	Get the array number from the array name.

State Machine Commands

State machine commands:

SmClear	Clear the event queue.
SmConfig	Define the state machine configuration parameters.
SmEvent	Define an event and its parameters.
SmInput	Request an event when a digital input changes.
SmMachine	Define a state machine variable.
SmParam	Request an event when a parameter value changes.
SmPeriod	Start a periodic timer.
SmPosition	Request an event when the motor passes a specified position.
SmPost	Create an event and post it to the end of the event queue.
SmPublish	Create an event and publish for all state machines.
SmRun	Start state machines running.
SmState	Define a state machine state, including its event handlers and substates.
SmStop	Stop running state machines.
SmSubscribe	Subscribe to receive published events.
SmTimer	Start a "once only" timer.
SmUnsubscribe	Request to stop receiving published events.
SmUrgent	Create an event and post it to the head of the event queue.

Synchronization Commands

Commands to synchronize a slave with a master:

SyncCam	Enter CAM mode synchronization and wait in an "idle" state.
SyncCamMarkerMaster	Synchronization in CAM mode with master marker correction.
SyncCamMarkerSlave	Synchronization in CAM mode with slave marker correction.
SyncCamStart	Start slave for synchronization in CAM mode.
SyncCamStop	Stop slave after the CAM synchronization.
SyncDefOrigin	Defines master-slave relation for the next SyncPos or SyncMarker command.
SyncError	Queries actual synchronization error of the slave.
SyncSetCurve	Set a CAM curve.
SyncShiftOrigin	Relative shifting of the origin of synchronization.
SyncMarker	Synchronization of angle/position with marker correction.
SyncMarkerRestart	Resets a marker or resets marker handling.
SyncPos	Synchronization of angle/position.
SyncStatus	Queries flag for synchronization status.
SyncStatusClear	Resetting of the flags MERR and MHIT.
SyncVel	Synchronization of velocity.

Commands to access elements within a CAM array:

CamArrayIndex	Determine the array index of various data blocks within a CAM curve array.
CamArrayLen	Calculate the required length of a CAM array.

System Commands

Various system commands:

Delay	Time delay.
Delayus	Time delay in microseconds.
DimArray	Pass a DIM array to functions by referencing the array number of the DIM array rather than the array name.
ErrorAxis	Determine the axis causing the last error.
ErrorClear	Clear the last error.
ErrorInfo	Query additional information about the current error.
ErrorNo	Query the actual error code.
Time	Query the system time.
TimeDiffus	Calculate an elapsed time.
TimeHw	Query the system timer.
UserStatus	Set the user status.
WdActivate	Activate the watchdog.
WdReset	Reset the watchdog.