POSITIONING AND CONTOURING CONTROL SYSTEM
ADDIPOS APCI-8001
OPERATING MANUAL / OM

Board Revision: APCI-8001 Rev. B/C, Rev. 6/072007

1 Introduction

The APCI-8001 is a universal positioning and contouring control system for machine tools, robots, handling gear and special purpose machines. Movements and process sequences can be automated through the entry of CNC programmes and parameters.
The APCI-8001 is designed as a PCI expansion board for the IBM-AT computer range, IPC (Industrial Personal Computer) or compatibles, and is used to control 1 to 3 CNC axes with either servo or stepper motors.
During development work, particular attention was paid to ensuring suitability for use in an industrial environment, since it is in this area in particular that personal computers are becoming increasingly widespread.
This manual essentially describes the hardware and software components of the accessories included in the standard scope of delivery, and is divided into three parts:

- **OM**: Operating Manual
- **PM**: Programming and Reference Manual
- **CM**: Commissioning Manual

The *OM* describes the APCI-8001 hardware and software components available in the standard scope of delivery. The user is given an overview of the APCI-8001's functions, the operating modes and programming methods.

The *PM* describes the various types of programming involved. This section is rounded off by a command and reference list.

The *CM* describes how to install and commission all the system components contained in the scope of delivery.

For easier comprehension, we recommend that you work through the Operating Manual first.

2 System hardware

2.1 The APCI-8001 board

The APCI-8001 is a PCI plug-in board for AT or IPC (Industrial Personal Computer with AT interfaces) PCs or compatibles, and constitutes the control system's intelligence.
The minimum configuration, comprising an APCI-8001, supports three CNC axis channels with either stepper or servo drives (DC or AC/EC). The maximum configuration supports eight axis channels. For this, a daughter board (OPMF) for 5 further CNC axes is plugged into the APCI-8001 using sandwich technology.

The APCI-8001 is accommodated on a short PCI plug-in board for IBM AT or compatibles. Communication between the PC and APCI-8001 takes place via the PCI bus. This ensures a high data throughput for outputting commands and reading in status information. The APCI-8001 address is managed by Plug+Play Bios of the PC.

2.1.1 The APCI-8001 – Interface logic

A significant feature of the APCI-8001 board is the use of state-of-the-art PLD and FPGA technology. FPGA modules are logical modules with a high level of functionality in service that can be reprogrammed as often as required. These contain the logic for actual value acquisition by the measurement systems. This includes, *inter alia*, processing of many common incremental coders, processing of SSI absolute encoders, and the precise real-time enabled temporary storage of position values based on various conditions. Of course, customised adjustments can also be made to the application.
The complete I/O periphery is electrically isolated from the system electronics (PC logic). The interfacing of external components occurs according to current guidelines for industrial electronics.

2.1.2 Connection of the external system components

The external system components mentioned in [section 2.2], such as power amplifiers, incremental encoders, limit switches, inputs and outputs, are interfaced through a 50-pin SUB-D pin connector at the front of the APCI-8001 board.
Three CNC axes can be connected to this connector with either AC or DC motors.
The APCI-8001 provides three analogue channels with an output voltage range of +/-10V and 16bit resolution. These channels are electrically isolated from the APCI-8001's digital power supply (PC power supply) and are used to control commercial power amplifiers that are connected as speed or current regulators.
Three stepper and directional signals, each with an antivalent signal level and a control voltage of 5V (RS422) are provided to drive stepper motor amplifiers.
Three pulse acquisition channels, each with 32bit register width, are used for position acquisition using incremental encoders or linear scales. Pulses are acquired either for TTL sensors or for sensors with symmetrical outputs (RS422). The quadrature signals generated by these sensors are electronically quadrupled and are also electrically isolated from the system electronics. The maximum pulse input frequency is 2.00 MHz. The APCI-8001 can also evaluate a reset pulse. The actual position of the measurement system can be temporarily stored in registers using the reset pulse or a digital input. These measurement methods allow simple encoder verification, setting of reference marks or realtime-enabled position latch for measurement machines.
As an alternative to the above-mentioned incremental encoders, all standard SSI absolute encoders can also be evaluated.
The APCI-8001 supports the connection of 16 optically decoupled inputs and 8 optically decoupled outputs. The inputs and outputs are not grouped specific to the axes. However, through software planning of the inputs and outputs, axis-specific assignment can be assigned specific functions, such as a limit switch function, a reference switch function and power amplifier enable, etc. All inputs and outputs which are not assigned special functions are freely programmable.
A watchdog circuit ensures safe operating states, even in exceptional situations. A flash memory is used to record various operating parameters. This includes, for example, the output setpoint status after system power-up.

2.2 External system components

Due to the digital signal processing function and the standard setpoint and actual value interfaces that are available on the APCI-8001, the APCI-8001 can be used with different motor types and power amplifiers, irrespective of the output range.
The external components are selected to suit the application concerned, after taking into account the performance class, functionality, safety considerations and cost-efficiency constraints.
All commercial power amplifiers can be controlled using a +/- 10V setpoint value channel. The power amplifiers can be connected as speed controller or as current amplifiers.
Both stepper and servo motors can be selected. All motors with a position feedback feature are designated as servo motors. This category includes brushless (AC or EC) and brush-gear equipped DC motors, or hydraulic motors. A position acquisition function is required to position these motor types. In general, rotary transducers or linear scales are used for this purpose. To control stepper motor power output stages, the system provides pulse and directional signals. In contrast to a servo drive, the stepper motor drives can be run without a position feedback feature.
Additional external system components can be linked to the APCI-8001 via digital inputs or outputs. Software planning of these inputs and outputs makes it possible to implement limit switches, reference switches, emergency stop, amplifier enable and other functions.

2.3 Types of power amplifiers

Various types of power output stages can be used to provide the energy for the electrical drive. All the types listed below can be controlled directly by the APCI-8001.
One of the features determined by the different types of power amplifiers is the characteristic control response of the controlled systems, which has to be allowed for when setting the filter parameters [CM / section 6.2].
Note: Most power amplifiers, except stepper motor power amplifiers, are usually controlled through the analogue setpoint value channels of the APCI-8001 board. Pulse and directional signals, and their inverted pulse trains are available for the use of stepper motor power amplifiers. This interface, however, is sometimes also implemented on servo systems with digital power amplifiers. In this case, a positioning control must be dimensioned on the power amplifier. The APCI-8001 must then be planned as for a stepper motor.

2.3.1 Speed controllers

These are the usual commercially available units used for controlling the speed of a DC or brushless DC motor. The input variable here is usually a voltage of +/- 10V, corresponding to a speed of +/- maximum speed. The speed actual value is fed directly to the speed controller, usually as a tacho signal. When the amplifier is enabled, the speed controller will develop a holding torque even without a setpoint value signal, i.e. the motor axis will drift away more or less slowly, due to the input offset. The input offset can normally be set, but is temperature-dependent.

2.3.2 Current amplifiers

In contrast to the speed controller, the current amplifier does not need a tacho signal as an actual value feedback from the motor. The input setpoint value here has the significance of an armature current. Many manufacturers offer their power amplifiers as current amplifiers or speed controllers. The commercially available speed controllers can usually be quickly and easily converted into a current amplifier. Please follow the instructions of the manufacturer concerned. The amplification factor should be set so as to ensure that the setpoint maximum current corresponds to an input setpoint value of 10V.

2.3.3 Voltage amplifiers

In contrast to the speed controller, the voltage amplifier does not need a tacho signal as an actual value feedback from the motor. The input setpoint value here has the significance of an armature voltage. In practice, voltage amplifiers are not of major importance and are at most used for small ratings when the motor current is limited by the armature resistance. Note that the available acceleration decreases at higher speeds.

2.3.4 Stepper motor power amplifiers

Stepper motors usually have 2, 3 or 5 phases or windings to be controlled. In order to ensure a rotary motion, current has to flow in the individual motor windings in a defined cycle. All the usual stepper motor power output stages available on the market will generally satisfy these requirements, and are controlled with the pulse (one motor step per pulse) and directional (specifies the direction of rotation of the motor shaft) input signals. The above-mentioned current flow results from this input information.

3 Introduction to how to operate and programme the APCI-8001

The APCI-8001 can be operated and programmed in several different ways. All tools and drivers are customized for the 32bit Windows operating systems (Windows 95, 98, NT, 2000, XP).

3.1 Manual operation

The APCI-8001 is manually operated using the *mcfg.exe* utility programme, which is part of the TOOLSET software [TSW] [section 2.2].
This offers the user a multitude of operating types for controlling the APCI-8001, including manual operation of the complete axis system.

3.2 Programming with a PC application programme (PCAP)

The user uses a high-level programming language such as *C* or *Pascal* to create a user programme for use on the PC. With the help of function libraries, this application programme is used to execute mcug3.dll commands on the APCI-8001 via a 32bit DLL driver. This DLL in turn requires a hardware driver (mini-port driver), which enables accesses via the PCI bus. The PC user programme is responsible for the coordinated sequence of the individual axis systems in this operating type. The PC Application Programme shall be referred hereinafter as PCAP.
Commands are transferred using pre-defined commands. These in turn have been implemented as DLL functions for 32bit Windows. There is a source or header file and, if necessary, corresponding lib. files for the above-mentioned programming languages.
From the user's perspective, the DLL functions merely represent a functional extension of the respective programming language. The actual "intelligence" of the functions is in the DLL driver or in the control system.

3.3 Programming as a stand-alone system (SAP)

Another option is to create programmes for the so-called stand-alone operating mode. This operating mode enables an operating programme that has already been loaded onto the APCI-8001 to be executed automatically, that is, without support from a PCAP. This means that the PC can handle other tasks. This Stand-alone Application Programme shall be referred to hereinafter as SAP.
The SAP is created by the user with the aid of an editor or the CNC-Edit editor, which is integrated in the *mcfg.exe* utility programme. The syntax for this user programme is similar to that for *Pascal* and permits simple and flexible programme creation. Once the user programme has been created, an Autocode file is generated using the *NCC* compiler, which is available both as a command line compiler [section 4.3] and integrated in the mcfg.exe utility programme. This autocode file can be transferred to the APCI-8001 and executed there automatically without support from the PC.
If synchronisation with a PCAP running in parallel is needed, this can be implemented using commonly pre-defined variables, which are accessible in read and write modes by the PC and the APCI-8001 alike.

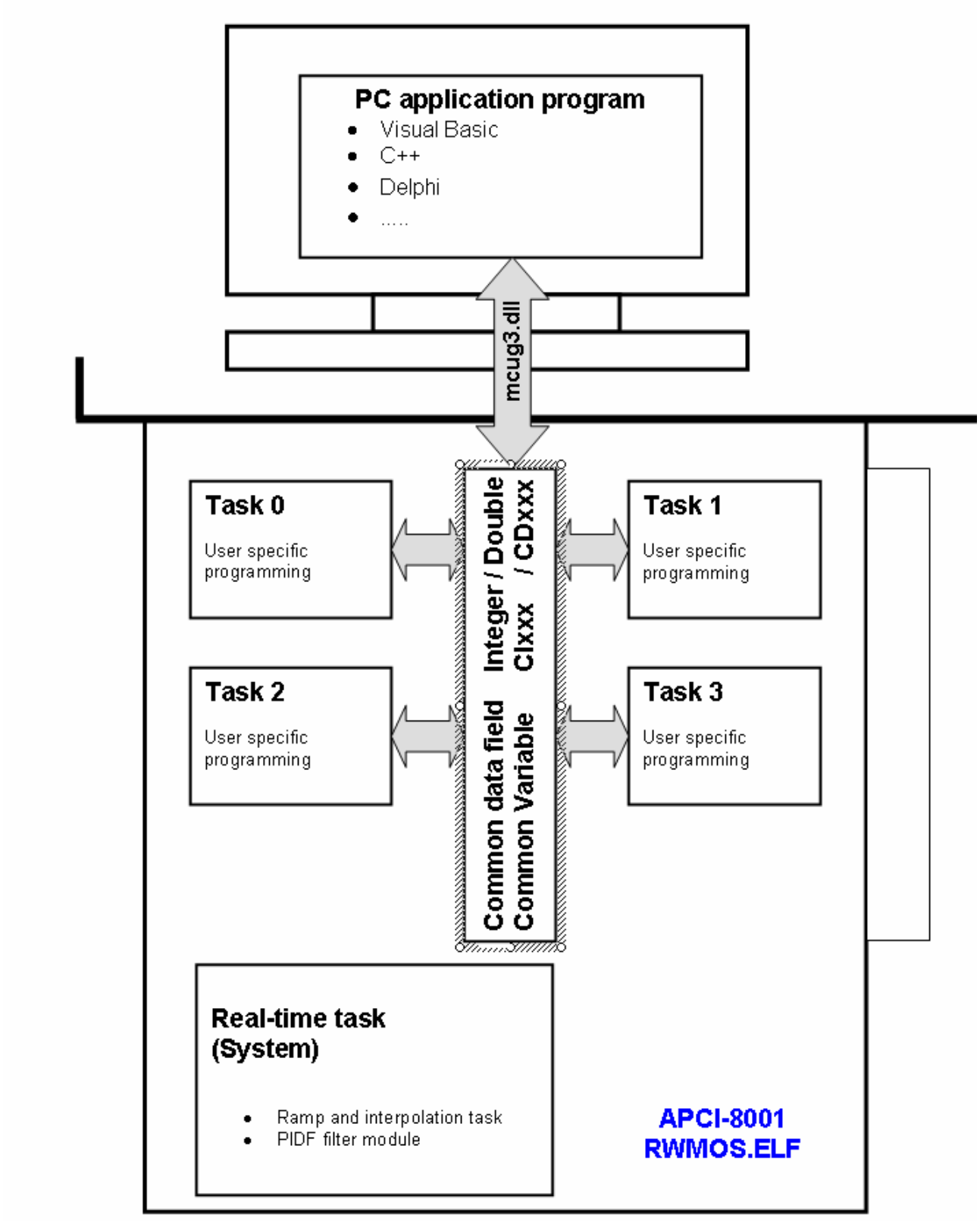
4 System software

This section describes the utility programmes for the TOOLSET software [TSW] contained in the standard scope of delivery. This information is important for correct operation of the control system and for the creation of user programmes.

4.1 The rw_MOS operating system software (rwmos.elf)

The *rw_MOS* operating system software is located in the rwmos.elf file and contains the executable machine code for the APCI-8001 CPU system. *rw_MOS* contains various software modules, which are required for the functionality of the control system. For a more detailed understanding of this, the most important software modules are explained in the following sections for interested users.

APCI-8001: Operating system



4.1.1 PIDF filter module

This module controls the motors to bring them to a desired setpoint position or speed using a digital PIDF filter. This filter is modelled on the analogue filter by processing at short, identical intervals. The control cycle time is set to 1.28 ms for all axis channels, but can be varied within certain constraints (100µs – 5ms). All actual value data that is required for the filter computation are also read in and processed in this scan cycle. Setpoint values are then output on the corresponding setpoint value channels.
Note: The PIDF filter is used both for servo and stepper motors. That is, the respective control loop must also be closed before using the axes in stepper motors. Here, the position is controlled at the actual output step pulse. An external position feedback feature is not required.

4.1.2 Ramp and interpolation module

The motors are brought to a desired target position using ramp functions with defined acceleration and pre-defined maximum velocity. This task is handled by the ramp and interpolation module. This module is synchronised with the PIDF filter module and is also executed once per scan cycle (1.28 ms) for all axes.
Another function handled by this module is to synchronise several axes in order to enable a path or space curve to be processed in interpolated mode. There are various interpolation procedures: linear, circular, helical and spline.
There are no restrictions regarding the selection of the axes involved in an interpolation procedure in the standard version (up to 3 axes) or in the multi-axis version (more than 3 axes).
The velocity trajectory is normally trapezoidal and is parameterised according to the acceleration, maximum velocity and target velocity. When S-profiles are selected, an s-form speed course can be set trapezoidal acceleration course.
Another important feature of the ramp and interpolation module is the second-order dynamic response module. The ramp generation feature implemented in many conventional positioning control systems demands a movement stop in the desired target position. This restriction has been eliminated by implementing this dynamic response module. The user can programme any target velocity for the desired target position. The axis system is moved to the desired target position, taking into account the maximum velocity and maximum acceleration, and reaches the programmed target velocity at precisely this point. This method enables movement profiles of any desired complexity to be generated.

4.1.3 PC interface module

This module is the basis for the PC application programming (PCAP programming) and executes the functions and commands called by the user. A wide range of commands is available. These commands are called from the PC Application Programme, which is created in a high-level programming language.

4.1.4 Stand-alone CNC module

This module is the basis for stand-alone application programming (SAP programming), and is one of the most powerful characteristics of the APCI-8001. Up to four stand-alone user programmes (SAP programmes) can be created using the *NCC* compiler provided in the scope of delivery. These are processed automatically in multi-task mode by the four CNC tasks. Since the drive monitoring function can also be handled completely by an SAP programme, the PC is available for other jobs.

4.1.5 Booting rw_MOS

The APCI-8001 *rw_MOS* operating system software must be transferred to the APCI-8001 each time the system is powered up, and triggers a system initialisation routing (hardware reset). This load operation is required at least once per system start and is concluded within a few seconds. The advantage of booting compared with a ROM-resident operating system is that customer-specific changes to the APCI-8001

operating programme can be easily incorporated. As soon as the boot operation has been completed, the APCI-8001 is ready for operation.

4.2 The *mcfg.exe* utility programme

The *mcfg.exe* utility programme is executed as a 32bit MDI application for Windows and offers a user-friendly environment for developing SAP programmes, as well as a powerful commissioning, diagnosis and configuration interface.
A detailed description is given in the respective manuals or sections. However, the configuration of the system data is described in brief below, as this is required for the first time the control system is commissioned.
The *mcfg.exe* utility programme requires the *system.dat* and *rwmos.elf* files. The *system.dat* file contains the user-specific settings for the respective control system. During installation, a default file from the installation CD is used or generated by the *sysgen.exe* utility programme.
When you start the *mcfg.exe* file, all the axis channels for which the *cef* error flat is set are displayed. This error results from data inconsistency between the system file (*system.dat*) and APCI-8001 board(s). The error can be cleared by saving in the active [System Data] window.

4.2.1 Editing the system data in the [System Data] window

This window is used to edit the axis-specific motor and system parameters. The parameters involved are system-specific default values for the individual axis channels. Most parameters can also be requested and set during the runtime using special read and write commands. The information that is recorded here is stored in the SYSTEM.DAT file.
The correct axis number must be selected for all entries. The individual data is edited in groups under several menus.

Note: The system data must be transferred to the APCI-8001 at least once after a control system boot procedure (e.g. in mcfg.exe), so that the operating programme (*rwmos.elf*) can be used on the APCI-8001 board. The data is loaded either from the PCAP user programme or from the *mcfg.exe* utility programme. In *mcfg.exe* and in the various example programmes, the system file (*system.dat*) is only transferred once.

4.2.1.1 Symbolic axis name

Each axis channel can be assigned a symbolic *Axis Name* {sn} with up to 10 characters. This axis name is defined automatically in the *rw_SymPas* programming language.

4.2.1.2 Motor type

The *Motor Type* {mt} parameter can be used to choose between stepper and servo drives. Motor type-specific parameters are specified in the [Motor-specific Parameters] menu.

4.2.1.3 Axis type

The *Axis Type* parameter {at} cannot be selected. It is obtained automatically by selecting the counter unit of the gear factor {gf}. If the unit concerned is a distance unit (mm, m, inch...), the axis type is specified as translatory. If angular units (rad, deg...) are selected, the axis type is defined as rotational. For rotational axes, the position values are restricted to one rotation, i.e. they are reset after the start angle has been passed.
Note: If an interpolated procedure for more than one axis is required, the axes involved must be of the same type.

4.2.1.4 Unit for displaying the position registers

In the *Position register display* field, you can select a display unit used internally for *mcfg.exe*. If position setpoint values or position actual values are displayed in the various status windows, the display will allow for this unit.

4.2.1.5 Display accuracy of the position register

The *Display precision* field can be used to specify the accuracy of the above-mentioned position registers. The value entered specifies the number of decimal places.

4.2.1.6 Encoder slits or number of steps for stepper motor axes

The number of *Encoder Slits* {slsp} is entered along with a unit of measurement. You can parameterise both rotational (angular encoders or rotary transducers) and translatory (linear scales) pulse measuring systems. This value is quadrupled internally as the APCI-8001's evaluation electronics also performs a quadrupling operation.
With stepper motor systems, this # *Step Pulses* value is not quadrupled. The actual value corresponds to the number of output step pulses.

4.2.1.7 Gear factor

The *Gear Factor* {gf} parameter specifies a step-up or step-down ratio between the actual value pulse acquisition and feed travel or angle of rotation. The gear factor is completed by a denominator and counter unit. The denominator unit selected is used to define two system variables. The first of these is the axis type, which is defined as either translatory (linear axes) for distance units or as rotational (round, rotary axes) for angular units. The selected unit is also the basic unit for all axis-specific movement commands (*jog* commands) and their profile parameters (velocity and acceleration, described below).

4.2.1.8 Jog (rapid traverse) parameters

The Jog (rapid traverse) parameters specify the axis-specific default values for rapid traverse positioning mode. These are acceleration *Maximum jog acceleration* {jac}, velocity *Maximum jog velocity* {jvl} and target velocity *Jog target velocity* {jtvl}. The target velocity is usually set to 0.

4.2.1.9 Home (reference travel) parameters

The Home (reference travel) parameters {hac} and {hvl}, like the rapid traverse parameters, specify the default values for the reference point search run. They are used automatically with all *home* commands as parameters for profile generation. Usually, the home velocity *Maximum home velocity* is only a fraction of the jog velocity, particularly when the reference switch is located near a limit switch.

4.2.1.10 Stop deceleration

This value {sdec} specifies the stop deceleration with which a standstill is reached for a limit switch event (only for limit switches that are declared as SMD limit switches). Deceleration for the job stop commands is also according to this deceleration value.

4.2.1.11 Maximum position error

The *Maximum position error* {mpe} parameter is used to specify the maximum permitted deviation between the setpoint and actual position of the motor axis. If this value is exceeded, this error will not affect profile generation and position control, but it will be displayed in the *axst* axis status register. Reaction to this status register can be either event-controlled or by query.
Note: The position error monitoring function will only be executed if the position control loop is closed and a value greater than zero is specified for {mpe}.

4.2.1.12 Software limits

For each motor axis, you can define a left-hand (*Software limit left side* {sll}) and right-hand *(Software limit right side* {slr}) software limit. However, the software limits are only monitored if the respective axis has been referenced by calling the shp function. If the software limit is exceeded for referenced systems, you can use a parameter to specify how this error state is to be handled. You have the following options:

NOFUNC (No Function) The software limit will be ignored.

TOM (Turn Off Motor) No value will be output on the setpoint value channel in the case of servo drives, since the axis would move deeper into the limit switch area. In the case of speed controllers, this means a speed setpoint value of 0 with a corresponding holding torque. But in the case of current amplifiers this means a current setpoint value of 0 and no holding torque. If the position setpoint value falls below the actual position, the axis will be moved in uncontrolled mode. If the position setpoint value falls below the limit position, the limit switch status will be cancelled again.

SMA (Stop Motor Abruptly) The axis will be held at the specified limit position in position control. Movement beyond the software limit is prevented. If the position setpoint value falls below the limit position, the limit switch status will be cancelled again.

SMD (Stop Motor Deceleration) When responding to this software limit, the axis will be automatically decelerated with the axis-specific delay {*sdec}* to velocity 0 and then held in position control. Movement beyond the limit is prevented. If the position setpoint value falls below the limit position, the limit switch status will be cancelled again. This function type is particularly recommended for cascaded speed control loops and stepper motor drives.

4.2.1.13 In-position window

The *In position window* {ipw} parameter is used to specify when the *ip* flag is set in the *axst* register. This flag is set after the end of the profile has been reached (*pe* flag in the *axst*) and if the position differential between the setpoint and actual position of the motor axis as specified in {ipw} is not met. All APCI-8001 digital outputs planned with the IP function are set or reset in the same manner as the *ip* flag.
Note: In position window monitoring takes place only if a value greater than zero has been specified for {ipw}.

4.2.1.14 Filter parameters

For these parameters, a distinction is made between stepper and servo drives. For servo axes, a PIDF filter parameter set can be set with coefficients {kp} for proportional amplification, {ki} for integral-action coefficient, {kd} for derivative-action coefficient and {kpl} for an additional phase lead. The factors {kfca} and {kfcv} can be used to specify compensation parameters for current amplifiers or speed controllers. These enable axis positioning to be made almost free of position error, even at high acceleration values. The filter is set according to the information given in the [CM / section 6.2]. In the case of stepper motor systems, there is only a {kp} filter parameter which is set by the system.

4.2.1.15 Manipulated variable limitation

In the case of servo axes, the {maxmcp} and {minmcp} variables can be used to limit the manipulated variable output to the maximum and minimum values you require. It is usual to set the maximum value to 10 V and the minimum value to –10V. If you reduce the maximum value, this means the manipulated variable for the power amplifier is also reduced. In the case of speed controllers, this means a reduction in the speed manipulating range, and in the case of current amplifiers a reduction in the torque range.

4.2.1.16 Manipulated variable compensation

In the case of servo axes with hydraulic motors, the {mcpcp} and [mcpcm} variables can be used to set a compensation voltage for the manipulated variable output. In closed loop control mode, these are the minimum output voltages on the analogue setpoint value channel. The system-entailed switch-on hysteresis at values customarily used for controlling the hydraulic motors can thus be suppressed.
No compensation voltage is normally required with other types of servo motors.

4.2.1.17 Inverting the manipulated variables

The *Invert motor command port* parameter can be used to invert the sign of the manipulated variable (motor command port) in the case of servo axes. This is particularly helpful when there is a phase angle rotation in the controlled system. The phase angle between the manipulated variable and the angel actual value is determined by the polarity of the motor lines and encoder signals or mechanical components, such as the gear unit.

4.2.1.18 Changing the count direction

The *Change encoder count direction* parameter is used to invert the count direction for the pulse acquisition channels. This is particularly helpful when there is a phase angle rotation in the controlled system.

4.2.1.19 Polarity of the index signal

The *Polarity of the index signal* parameter is used to specify the polarity of the zero-track signal (index pulse) of a pulse encoder.
Note: The current status of the zero-track signal is also displayed in the [Dialog Functions Menu][Show Inputs / Status ..] menu.

4.2.1.20 Start-Stop frequency

This parameter is reserved for future extensions.

4.2.1.21 Pulse acquisition with stepper motor systems

Usually, stepper motor axes are controlled in open loop mode. This means that the motor system follows a setpoint frequency and the actual position always has to coincide with the computed setpoint position.
As an option, this parameter can be used to involve in the drive system a pulse encoder building onto the stepper motor system. This will also enable the stepper motor axes to be operated in closed loop mode, which is helpful in certain circumstances (this parameter is not currently supported).

4.2.2 Editing the hardware parameters [Dig. Inputs] + [Dig. Outputs]

This menu is used for axis-specific planning of the hardware characteristics for the APCI-8001's digital inputs and outputs.

4.2.2.1 APCI-8001 digital inputs [Dig. Inputs]

All digital inputs can be assigned to a special function in the *Special Function* field. The special function becomes operative whenever the correspondingly planned digital input is activated. The significance and operation of all possible special functions is explained below:

NOFUNC (No Function) The input has no special function. It serves merely as a freely programmable digital input.

REF (Reference Switch) The inputs planned with this function act as reference or stop switches. If a digital input is activated by a reference switch (cam), for example, during the execution of special reference travel commands (e.g. *jhl()* command), the selected axis channel will be decelerated to a velocity of 0 with the axis-specific reference travel deceleration.

LSL_TOM (Limit Switch Left Turn Off Motor) The inputs planned with this function act in the same way as left-hand hardware limit switches. When this input responds, no value will be output on the setpoint value channel in the case of servo drives, since the axis would move deeper into the limit switch area. In the case of speed controllers, this means a speed desired value of 0 with a corresponding holding torque. But in the case of current amplifiers this means a current desired value of 0 and no holding torque. The limit switch is normally tripped by movement in a negative direction and exceeding the corresponding limit. If the position setpoint value falls below the actual position, the axis will be moved in uncontrolled mode. If the position falls below the setpoint value at which the limit switch was identified, the limit switch status will be cancelled again.

LSL_SMA (Limit Switch Left Stop Motor Abruptly). This input also functions in the same manner as left-hand hardware limit switch, but when activated it causes the axis to be held at its current position in the position control operating mode. Movement beyond the limit is prevented. If the position setpoint value falls below the limit position, the limit switch status will be cancelled again.

LSL_SMD (Limit Switch Left Stop Motor Decelerated). This input also functions in the same manner as a left-hand hardware limit switch, but when activated causes the axis to decelerate from the current position to velocity 0, with the deceleration that is specified in *Stop deceleration {sdec}.* Movement beyond the limit is prevented. If the position setpoint value falls below the limit position, the limit switch status will be cancelled again. This should be the preferred limit switch option.

LSR_TOM (Limit Switch Right Turn Off Motor) This mode of operation here is identical to that for LSL_TOM, except that this limit switch is planned for the right-hand limit.

LSR_SMA (Limit Switch Right Stop Motor Abruptly). The mode of operation here is identical to that for LSL_SMA, except that this limit switch is planned for the right-hand limit.

LSR_SMD (Limit Switch Right Stop Motor Decelerated). This input also functions in the same manner as a right-hand hardware limit switch, but when activated causes the axis to decelerate from the current position to velocity 0, with the deceleration that is specified in *Stop deceleration {sdec}.* Movement beyond the limit is prevented. If the position setpoint value falls below the limit position, the limit switch status will be cancelled again. This should be the preferred limit switch option.

EO (Emergency Out) This input signals via a Bit in the axst register that an emergency stop button incorporated in the drive system has been pressed.
Note: This input has no effect on the drive system. It is at the user's discretion how he/she reacts to this event.
This signal can be evaluated by interrogation [PM / section 4.4.48 – rdaxst()] or by means of an EVENT handler [PM / section 6.4.2].

DR (Drive Ready) This input signals whether the power amplifier connected to this channel is showing operational readiness. This signal can be evaluated by interrogation [PM / section 4.4.48 – rdaxst()] or by means of an EVENT handler [PM / section 6.4.3].

UI (User Input) This input has no effect on the drive system. Since an EVENT handler is also available for this input type, the alternative cyclical interrogation (polling) of inputs can be dispensed with. This signal can be evaluated by interrogation of the axst register [PM / section 4.4.48 – rdaxst()] or by means of an EVENT handler [PM / section 6.4.7].

LSL_SMD (Limit Switch Left Turn Off Motor (with) Deceleration) The inputs planned with this function act in the same way as left-hand hardware limit switches. When this input responds, the axis is automatically decelerated with the axis-specific delay {*sdec}* to velocity 0 and then held in position control. Movement beyond the limit is prevented. The limit switch is normally tripped by movement in a negative direction and exceeding the corresponding limit. If the position setpoint value falls below the position, the limit switch status will be cancelled again. This function type is particularly recommended for cascaded speed control loops and stepper motor drives.
Note: The *sdec* delay must be set so as to ensure that the drive is stopped safely, without the motor axis running into a mechanical limit and thus causing damage. To protect the drive, additional hardware limit switches should be set, which only enable the power amplifiers in the permitted direction of travel.

LSR_SMD (Limit Switch Right Stop Motor (with) Deceleration) This mode of operation here is identical to that for LSL_SMD, except that this limit switch is planned for the right-hand limit.

LP (Latch Position) When activating this input, the actual position {rp} of the corresponding motor axis is stored temporarily. If a latch procedure has been triggered, the *lspf* flag of the *axst* register is set. Then the temporarily stored position can be read in using the PCAP command rdlp () [PM / section 4.4.80] or the SAP axis qualifier *lp*. The *lpsf* flag is automatically deleted by the read in procedure. The maximum delay of the latch procedure is 2 scanning intervals (2.56 ms). This is a (delayed) software latch, which can be used for all digital inputs. There is **one** delay-free hardware input for each axis, which is always active. Only the polarity can be changed in this case.

4.2.2.2 Inverting the APCI-8001 digital inputs

All APCI-8001 digital inputs can be inverted axis-specifically by means of the software. NC or NO contacts can thus be operated at the respective inputs without additional hardware. The information on whether an input is to be inverted or not is stored on the APCI-8001 flash memory.

Note: In the factory, all inputs of the APCI-8001 system electronics have been planned without inversion. The inputs are activated when a voltage of +24V is applied.

4.2.2.3 APCI-8001 digital outputs [Dig. Outputs]

All digital outputs can be assigned to a special function in the *Special Function* field. The significance and operation of these special functions is explained below:

NOFUNC (No Function) The output has no special function. It serves merely as a freely programmable digital output.

PAE (Power Amplifier Enable) This output is set whenever the corresponding axis channel is switched into position control. It is used to enable the external power amplifier module. This is enabled by the *cl()* command. The output is reset as soon as the position control is switched off. This is the case, for example, with the *ra(), rs()* or *ol()* commands.
The power amplifiers must be enabled in the following situations, for example: In interference situations or due to the system-entailed offset drift in the case of speed regulator devices.

Note: All outputs are reset in terms of the hardware whenever the hardware is reset following a power-on, power-fail or reset procedure. The commercially available power amplifiers are enables with a potential-free relay contact. These could be controlled by a PAE planned output.
A relay point (NO) is provided at the 10-pin FB connector X5, for each axis channel. In the factory, this relay point is planned with PAE function (see Commissioning Manual Chapter 5.2.7).

IP (In position) This output is set whenever the axis channel involved has reached the end of its profile, and in addition to the actual position and the setpoint position are within the position window specified in {ipw} [section 4.2.1.13].

MPE (Maximum position error) This output is set whenever the position error specified in {mpe} is exceeded [section 4.2.1.11].

SIGN_SPEC (Sign / Special Function) This output has application specific functionality that may vary according to the used RWMOS version. When MotorType „STEPPER_NDX“ **Fehler! Verweisquelle konnte nicht gefunden werden.** is selected, at these outputs the direction signal is output.

4.2.2.4 Initial state of the APCI-8001 digital outputs

All digital outputs can be assigned a default value in the *Set/Reset Output* field. This value is output after power-up, especially after booting or a hardware reset. During the reset operation, 0 V is output at all digital outputs.
The initial state of the digital outputs can be set axis-specifically. However, the value “1” or “set” is dominant, i.e. if “set” is activated for any axis, the significant of other axis plans is no longer of any importance.

4.3 The *ncc.exe* utility programme

Using the *ncc.exe* command line compiler, Windows 32 systems can generate stand-alone application programmes (SAP programmes) directly from the DOS level (input request). The compiler is completely identical to the NCC compiler incorporated in *mcfg.exe*, and is called as follows:

NCC filename [Option]

The text file with the name *filename* must contain an SAP programme. The ending *".src*" is automatically assumed as the file extension name. The various options [*Option*] are explained below and must be separated from each other by blanks.

If error-free compilation has proved possible, then an autocode file with the file name *filename.cnc* is created in the current directory. If an error occurs, compilation will be aborted and the faulty line number will be displayed on the screen, together with an error text.

NCC command line compiler options [*Option*]

Option	Function
FS	(Full System) The system does not check how many axes are actually available in the system, i.e. up to 18 different axes designators can be referenced. The default names of these axes designators are A1 .. A18.
SC	(Syntax Check) The system executes only a compiler run, without generating a CNC files. This is primarily done for syntax checking the SAP source text programme.
TSK x	(Task Selection, x = 0..3) The autocode file is generated for Task x. But if the SAP programme contains a {$TASK} compiler command, the TSK x parameter is ignored.