Controller of High-Voltage Bipolar Power Supply / Amplifier with up to 5 Channels and an Optional Current Monitor

Firmware Version 1-00

User Manual

Document version A, created on Feb-18-2015

Contents

Figure List

Table List

Technical Data

Characteristics

- control and monitoring of up to 5 amplifier channels
- monitoring of amplifier supply voltage and heatsink temperature
- serving an interlock loop
- supply voltage supervising
- microcontroller: 16-bit RISC running at 16 MHz
- user-friendly graphical interface
- non-volatile memory (FRAM) to permanently store device setting
- USB and LAN data interface

Human Interface

- monochrome LCD display 128x64 pixel
  - pixel size: 0.5 mm
  - pixel color: yellow, background: blue
  - background illumination: white LED
- keypad: 5 keys: 4x direction + 1x "enter"
- rotary encoders:
  - 24 positions per revolution, integrated press button
  - one encoder located beneath the LCD for general control
  - up to 5 encoders for direct control of the output voltages
- optional external shutdown button via the interlock loop

Digital Interface

- interface parameters:
  - data rate: 230.4 kBaud
  - data bits: 8
  - stop bits: 1
  - parity: even
- USB interface according to USB 2.0 standard
  - connector: USB receptacle type B
  - data transfer rate: up to 12 MBit/s (*Full Speed*)
  - effective data transfer rate: >10 kBit/s

- LAN interface according to Ethernet 10/100BaseT standards
  connector: LAN receptacle RJ–45
  data transfer rate: 10/100 MBit/s (fast Ethernet)
  effective data transfer rate: >10 kBit/s

Brief Description

The amplifier controller is intended to manage the operation of up to 5 amplifier channels. Optionally, one channel can be replaced by a high-sensitivity current monitor. The amplifier controller is equipped with a USB and/or a LAN data interfaces. They allow to transfer measurement and settings data to or from the device and to control it remotely (see sections "Software Utilities" and "Software Interface").

The device contains a microcontroller that provides the human interface (see section "Human Interface"), controls the USB and LAN data interfaces, and executes several monitoring tasks. It also controls the digital-to-analog converters driving the amplifiers. Further, the controller contains monitoring circuits of the amplifiers and of the power supplies. They are based on analog-to-digital converters that measure the output and the supply voltages. Optionally, one analog-to-digital converter measures the output of the current sensor.

The device monitors an interlock loop that disables the outputs if it is opened. The interlock loop is typically connected to a vacuum controller, it should disable the device if the vacuum pressure exceeds a certain value above that discharges might occur.

The attached amplifiers and their power supplies are typically cooled by a heatsink located at the rear panel. The heatsink is passively cooled, the controller unit monitors the heatsink temperature. The attached amplifiers are disabled if the heatsink threatens to overheat or if the temperature sensor fails.

The actual settings of the output voltages and other operation parameters are stored in a non-volatile memory and are automatically restored at the next device startup.

The USB or LAN interface can be used to remotely control the device by simple Windows™ programs. These utilities can also obtain hardcopies, i.e. save the current LCD contents to a bitmap file at the host computer. Alternatively, you may write your own user software based on the supplied software interface (see section "Software Interface"). Upon request, software interface for LabVIEW or other development systems is available. The USB interface can also be used to upgrade the firmware. For a detailed description, see the section "Software Utilities".

Quick Setup Guide

To prepare the device for an experiment, you need just to connect all cables. You should, however, also check the operation parameters. To do so, follow the next steps:

- Ensure that the proper mains voltage is selected (consult the manual of the main device), connect the line cord and turn the device on. The LCD should light up.
- Press the middle key "enter" and start the system control by selecting the menu item "Control ► System". Check the displayed values and if desired change them (for details see section "System Preferences"). Finally, close the dialog box.
- Press the middle key "enter" and start the interface control by selecting the menu item "Control ► Interface". Check the displayed values and if desired change them. You should enable just the interface that you aim to use. Especially the LAN interface produces a significant amount of heat in the device, thus should be disabled if not used (for details see section "Interface Control"). Close the dialog box when you have finished the setup.
- Close the interlock loop. You may insert a BNC termination plug if the interlock loop should not be used. In a properly designed experiment, however, the interlock loop should be closed by a vacuum control unit when a sufficiently low pressure in the chamber with the electrodes driven by the device is reached.
- Press the middle key "enter" and start the power monitoring by selecting the menu item "Monitor ► Power". Check whether the device recognized that the interlock loop is closed (for details see section "Power Monitor"). Close the dialog box when you have finished checking the interlock loop.
- Turn the rotary encoder of a certain channel, the amplifier control will be started. Adjust the desired output voltage and press the encoder knob to turn the output on. The state of the channel should change from "off" to "on". Note that the amplifier control closes after 5 seconds of inactivity automatically. To prevent this, press any navigation key. Note further that during any larger voltage transition, the state may temporarily change to "fail" indicating that the output voltage does not match the set value exactly. If the device indicates the state "fail" permanently, the measured output voltage differs from the set value. The accepted tolerance is given by the settings

that can be displayed in the hardware information (for details see section "Hardware Information"). The output voltage is out of the tolerance if the output is overloaded or in case of any hardware failure. Please disconnect the output cable and if it solves the problem, check the connected hardware including cables. If the problem persists, check the output voltages by amplifier monitoring (for details see section "Amplifier Monitor") and contact the manufacturer of the device.

If the device is equipped with a current sensor, check also its operation:

- Press the middle key "enter" and start the current monitoring by selecting the menu item "Monitor ► Current". If no cable is connected to the sensor input, the displayed value should be around zero and the device should activate the most sensitive current range (consult the manual of the main device to find out these values).
- Inject current into the sensor input via an appropriate resistor. For instance a resistor of 1 MΩ will provide a current of 1 µA if it is attached to a voltage source of 1 V. Check the displayed value and compare it with the expected current (for details see section "Current Monitor"). In case of any discrepancy larger than several percent, contact the manufacturer of the device.

! **Warning:** Never overload the sensor input by applying a high current or by connecting a voltage source directly to the sensor input.

If you encounter problems, read carefully the corresponding section in this manual. If you cannot solve the problem, contact the manufacturer of the device. Before doing it, obtain hardcopies of the LCD panel showing all monitoring parameters and send them together with the problem description.

Human Interface

The microcontroller's firmware implements a user-friendly graphical interface. The device is controlled by the help of menus and dialog boxes, they provide an intuitive device control.

The device is equipped with a graphic liquid crystal display (LCD), a set of keys (keypad) and rotary encoders. The keys are arranged in a circle symbolizing the key function: There are four direction keys ("left", "right", "up", and "down") and a middle key for confirmation or selection ("enter").

The keys are used for menu navigation, for selecting dialog items, or for changing various values. The function of the keypad in every device state is symbolized on the LCD immediately above the keypad. A text describes the function of the middle key, alternatively just the symbol "↵" is displayed showing that the middle key can be used. Similarly, arrows show which of the keys has an influence on the operation in the current state. When a menu is active, the vertical direction keys are used to change the current selection. The right direction key as well the middle key opens a submenu, provided it is available. The left direction key closes the submenu or the main menu if there was no opened submenu. The middle key selects the menu item and launches the corresponding action.

The rotary encoder beneath the LCD is used to change numerical values. The function of the encoder in every device state is symbolized on the LCD immediately above the encoder knob. In device idle state, all rotary encoders can be used to control the output voltages. The encoder beneath the LCD shifts all voltages, whereas the remaining encoders control one particular channel. The encoder's push button is used to turn the corresponding channel on or off. If this feature is available, the symbol "⊡" is shown on the LCD above the encoder knob. For more details, see section "Amplifier Control"

The encoders use an enhanced speed control. This enables you to set precisely any desired value or rapidly make large changes, since the value steps are proportional to the rotational speed of the particular encoder. You can change the corresponding value in small steps when you rotate the encoder slowly or make large changes when you spin the encoder knob rapidly.

Main Program Menu

When the device starts, the initialization takes place and the manufacturer logo is shown for couple of seconds. Then, the device enters the idle state (see Fig. 1). The lower right corner shows the uptime, i.e. the time elapsed from the last device start.

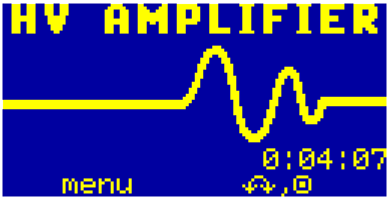


Fig. 1. Program display in device idle state.

By pressing the middle key "enter", the main menu can be accessed (see Fig. 2). You can control the device (submenu "Control", see also section "Device Control"), monitor the device function (submenu "Monitor", see also section "Device Monitoring"), obtain information about the device (submenu "Information", see also section "Device Information"), or manage the settings (submenu "Adjustment").

Note that the submenu "Monitor" contains the menu item "Current" only if the current sensor is installed (see also sections "Hardware In-

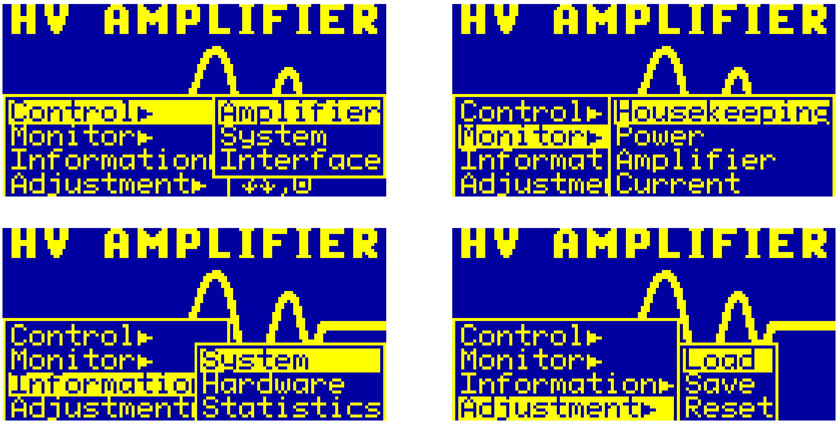


Fig. 2. Main program menu.
The figure shows the different submenus of the main program menu.

formation" and "Current Monitor").

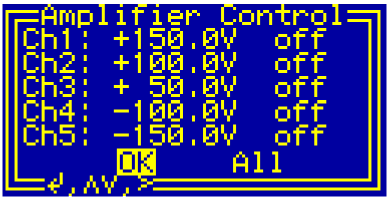
The system settings are the data that can be modified in the submenu "Control". You can save the current settings (menu item "Adjustment ► Save"), load the settings back (menu item "Adjustment ► Load"), or reset them to default values (menu item "Adjustment ► Reset"). The recent system settings are saved automatically to a non-volatile memory when the device powers down and are loaded back when the device starts (see also section "Housekeeping Data"). You may save the settings using the main menu prior to the next device shutdown if you have made changes that you wish to keep permanently. When you choose to load the settings back, the settings will be reloaded and the last saved state will be restored, i.e. the settings will be set to the state immediately before the last shutdown or after the last startup, or to the state when you have saved the settings. Further, you may reset the settings to their default values. This is advisable if you wish to restore the original device state.

Device Control

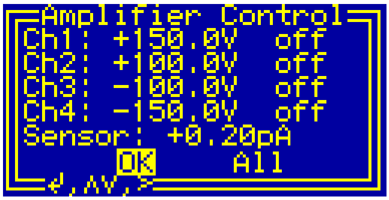
Several dialog boxes are available to control the device and to setup the device behavior.

Amplifier Control

Using the dialog box "Amplifier Control", you can set the output voltages of the device (see Fig. 3). Note that this dialog box opens automatically if you turn or press any encoder knob in the device idle state. In this case, the dialog box remains opened for 5 seconds after the last knob motion. You can prevent the automatic closing if you press any navigation key. Note that the pressed key will be evaluated, thus may execute a certain function.



```
Amplifier Control
Ch1: +150.0V  off
Ch2: +100.0V  off
Ch3: + 50.0V  off
Ch4: -100.0V  off
Ch5: -150.0V  off
     OK      All
↲,ΛV,>
```



```
Amplifier Control
Ch1: +150.0V  off
Ch2: +100.0V  off
Ch3: -100.0V  off
Ch4: -150.0V  off
Sensor: +0.20pA
     OK      All
↲,ΛV,>
```

Fig. 3. Dialog box "Amplifier Control".
The figures show the dialog box for two different devices: the left one with 5 and the right with 4 installed channels incl. current sensor.

The number of the shown items depends on the number of installed channels. Additionally, if a current sensor is installed (see section "Current Monitor"), the measured value is displayed to provide an immediate feedback between the adjusted values and the measured current (see right part of Fig. 3).

To set a particular output voltage, use the vertical direction keys and rotate the encoder knob beneath the LCD. Alternatively, you may rotate the encoder knob corresponding to the particular channel. The device will select the channel and set the output voltage directly. Note that if the particular channel is a part of a differential group, the encoder knob affects the voltages of both channels in the group (see also section "System Preferences").

When the device is turned on or when the interlock loop (see also section "Power Monitor") is open, the output voltages are turned off. This is indicated by the state "off" behind the particular output voltage. You may turn the output voltage on and off again by pressing the encoder knob. You may either select the desired channel by the vertical direction keys and press the encoder knob beneath the LCD or press the encoder knob corresponding to the desired channel.

If a particular output voltage is turned on, the state changes to "on". This indicates that the output voltage has reached the desired value with a tolerance given by the item "Precision" in the dialog box "Hardware Information" (see also section "Hardware Information"). If the output voltage differs by a larger value than given by the tolerance, the state changes to "failure". If this happens, check the connected load. Turn the outputs off, disconnect the output cable and turn the outputs on again. If this solved the problem, the connected load has a lower impedance than the device can drive or even shows a short circuit. If the problem persists, check the device settings and operation parameters. Contact the manufacturer if you cannot solve the problem by yourself.

Note that if the output voltage changes by a larger value, the device indicates the state "failure" for a short time. This is a normal behavior during voltage transitions, since the device needs a certain time to adjust the output voltages and to obtain correct measurement values of them.

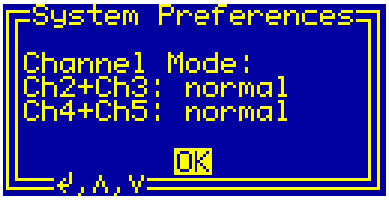
You may change all voltages by a certain offset if you select the item "All" by vertical and/or horizontal direction keys and rotate the encoder knob beneath the LCD. You can reach this function easily, if you rotate the encoder knob beneath the LCD in the device idle state. This also opens the dialog box "Amplifier Control" and selects the item "All".

Similarly, you may turn all output voltages on and off again by pressing the encoder knob beneath the LCD if the item "All" is selected. You may also press the encoder knob beneath the LCD in the device idle state, this opens the dialog box "Amplifier Control" and toggles the output voltages directly.

If you aim to manipulate the output cables, the connectors, or the connected electrodes, use the dialog box "Amplifier Control" to turn off and check the output voltages. Be sure that you open the connectors or touch the contacts only if the output voltages are turned off and the indicated state is "off". This ensures, that the output voltages have decreased to values that are not lethal.

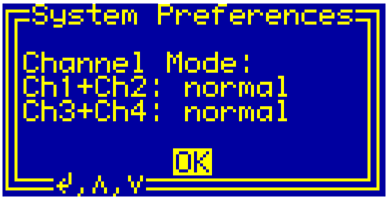
System Preferences

The dialog box "System Preferences" (see Fig. 4) shows the current system settings and enables you to change them. The number of the displayed items and their names differ according to the number of installed channels (see also section "Hardware Information").


System Preferences

Channel Mode:
Ch2+Ch3: normal
Ch4+Ch5: normal

OK


System Preferences

Channel Mode:
Ch1+Ch2: normal
Ch3+Ch4: normal

OK

Fig. 4. Dialog box "System Preferences".
The figures show the dialog box for two different devices: the left one with 5 and the right with 4 installed channels.

Use the vertical direction keys to change the currently selected item. The selected item is highlighted by inverting the displayed characters. By pressing the middle key "enter", the settings can by modified or the dialog box can be closed (see the selection shown in Fig. 4).

The shown value "normal" is the default settings, it means that the corresponding channels behave independently on each other. If you change the value to "diff.", the device couples the control of the corresponding channels to simplify a generation of differential voltages. The control of the first channel (i.e. channel 2 for the first item in Fig. 4) controls the mean voltage of the both outputs, while the voltage difference between them is kept constant. The control of the second channel (i.e. channel 3 for the first item in Fig. 4) controls the voltage difference between the both outputs, while the mean voltage is kept constant. This procedure helps you, for instance, to control deflection electrodes, since the both control values have a different physical meaning. Whereas the mean voltage of the electrode pair determines the focusing properties within an electrode system, the voltage difference influences the deflection properties.

Interface Control

The dialog box "Interface Control" (see Fig. 5) shows the current interface settings and enables you to change them.

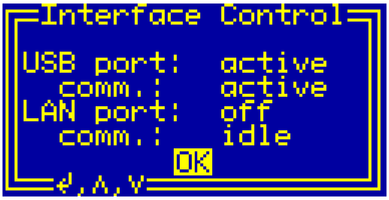

```
Interface Control
USB port:  active
  comm.:   active
LAN port:  off
  comm.:   idle
         OK
↵,∧,∨
```

Fig. 5. Dialog box "Interface Control".

Use the vertical direction keys to change the currently selected item. The selected item is highlighted by inverting the displayed characters. By pressing the middle key "enter", the settings can by modified or the dialog box can be closed (see the selection shown in Fig. 5).

The setting of each particular interface can be changed from "active" or "idle" to "off". If the interface state is "off", the interface is disabled and no communication is possible. If the LAN interface is disabled, also its power supply is deactivated to lower heat production. The remaining interface states indicate that the interface is enabled. If the displayed state is "active", the interface is activated and a communication can be launched. If the displayed state is "idle", the interface is activated but cannot communicate. For the LAN interface, the "idle" state is displayed for a short startup time when the interface is being activated and changes to "active" after several 100 ms. For the USB interface, the "idle" state is displayed if there is no connection to a remote host. If a cable connects the device with a functioning remote host, the state changes to "active".

The communication state shown as the items "comm." indicates the state of the handshake lines of the particular interface. The communication state changes between "active" and "idle" depending on the communication activity on the particular interface. The communication state is "idle" if you do not access the device and it changes to "active" if you start a communication software on the remote host.

Device Monitoring

To monitor the function of the device, several dialog boxes are available. You can use them to check the function of the device if you encounter any problem.

Housekeeping Data

The dialog box "Housekeeping Data" (see Fig. 6) shows the supply voltages of the device and the temperature and load of the microcontroller.

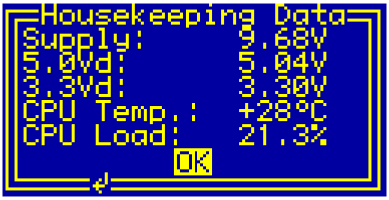

```
Housekeeping Data
Supply:       9.68V
5.0Vd:        5.04V
3.3Vd:        3.30V
CPU Temp.:    +28°C
CPU Load:     21.3%
         OK
```

Fig. 6. Dialog box "Housekeeping Data".

The value "Supply" is the voltage supplying the voltage regulators. The voltage value is under normal conditions around 9 V providing enough headroom to produce the subsequent 5 V supply. When the value drops down below 6.5 V, the controller starts to prepare the power shutdown by saving the settings to a non-volatile memory. When the value approaches 6 V, for instance during a mains dropout, the controller is reset and halted until the supply voltage recovers.

The values "5.0Vd" and "3.3Vd" are the output voltages of the regulators supplying the logical circuits. The nominal values are 5.0 V and 3.3 V, respectively.

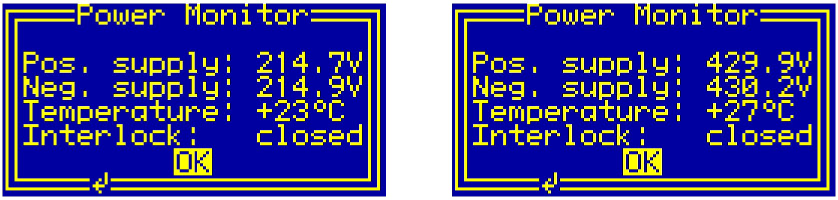
The value "CPU Temp." shows the temperature of the microcontroller. The temperature is measured by an on-chip sensor, the accuracy is around 1°C.

The value "CPU Load" is the obtained load of the microcontroller. The value gives the approximate percentage of the time in that the microcontroller is active. The load increases especially if the software utility for remote controlling is active (see section "Utility RemoteControl").

The values are updated about 2-times per second, the CPU load once per second.

Power Monitor

The dialog box "Power Monitor" (see Fig. 7) shows the voltages supplying the amplifiers, the heatsink temperature, and the state of the interlock input.



```
Power Monitor
Pos. supply: 214.7V
Neg. supply: 214.9V
Temperature: +23°C
Interlock:   closed
       OK
```

```
Power Monitor
Pos. supply: 429.9V
Neg. supply: 430.2V
Temperature: +27°C
Interlock:   closed
       OK
```

Fig. 7. Dialog box "Power Monitor".
The figures show the dialog box for two different devices with a different nominal output voltage: 200 V (left) and 400 V (right).

The values "Pos. supply" and "Neg. supply" are, respectively, the positive and negative voltages supplying the output amplifiers of the device. The nominal values of the voltages are 15-20 V higher than the maximum output voltages of the device; Fig. 7 shows the typical values for a device with maximum voltages of ±200 V.

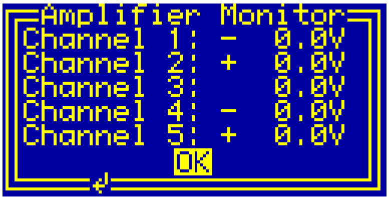
The value "Temperature" is the temperature of the heatsink at the rear plate. The measurement uses a semiconductor sensor and has an accuracy of about 2°C. If the sensor is disconnected, "N.C." is displayed instead of the temperature value.

The value "Interlock" shows the interlock loop status. The possible values are "closed" or "open". If the interlock state is "open", the output voltages cannot be activated.

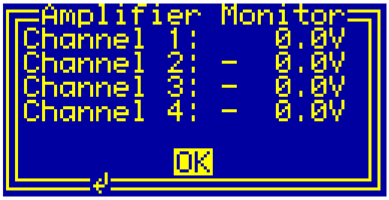
The values are updated about 2-times per second, the interlock state once per millisecond. The microcontroller checks the values continuously and turns the output voltages off if the temperature rises above the allowed value (see also section "Hardware Information") or if the interlock loop is opened. The microcontroller is also able to discover failures such as a disconnected temperature sensor; such failure causes an immediate deactivation of the output voltages.

Amplifier Monitor

The dialog box "Amplifier Monitor" (see Fig. 8) shows the measured output voltages.



```
Amplifier Monitor
Channel 1: -   0.0V
Channel 2: +   0.0V
Channel 3:     0.0V
Channel 4: -   0.0V
Channel 5: +   0.0V
         OK
```



```
Amplifier Monitor
Channel 1:     0.0V
Channel 2: -   0.0V
Channel 3: -   0.0V
Channel 4: -   0.0V

         OK
```

Fig. 8. Dialog box "Amplifier Monitor".
The figures show the dialog box for two different devices: the left one with 5 and the right with 4 installed channels.

The displayed values are obtained by a special circuit that measures the operation parameters of the output amplifiers. The circuit is calibrated to provide precise values if the output amplifiers work properly, i.e. if the output voltages approach the set values. If the output voltages differ from the set values, the precision of the circuit may decrease significantly. Thus, the measured values can be used to verify the output voltages only, do not rely on the displayed values if you observe significant differences to the expected output voltages.

Current Monitor

The dialog box "Current Monitor" (see Fig. 9) shows the measured current of the optional sensor.

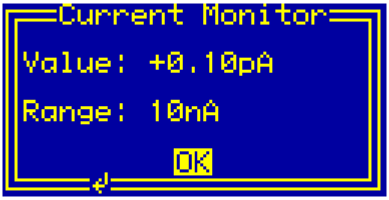

Current Monitor
Value: +0.10pA
Range: 10nA

Fig. 9. Dialog box "Current Monitor".

The current sensor is a sensitive transimpedance preamplifier that converts its input current into a voltage, which is measured by an analog-to-digital converter. The preamplifier has three current ranges, thus can measure currents in a range extending over several orders of magnitude. The ranges are switched automatically, the controller increases or decreases the sensitivity if the magnitude of the measured current is too high or low for the particular range.

The transimpedance preamplifier keeps its input at virtual ground, i.e. holds the input voltage at a potential close to zero. This implies that the device sets also the potential of a connected collector electrode to zero. Note that if the current sensor is overloaded, i.e. if the input current exceeds the preamplifier capability, the circuit is not more able to set the input voltage to zero. This may also happen temporarily if the device switches the measurement range.

The input of the current sensor has a typical capacitance of several nanofarad. This suppresses coupling of any high-frequency noise into the input and also reduces the abovementioned effect. To change the input potential, the input capacitance has to be charged; dependent on the input current, this may take a time that is much longer than the transition due to range switching.

The current sensor is an optional circuit that can be installed instead of the 5th device channel, i.e. the sensor is available on devices with less than 5 channels only.

Device Information

To obtain information about various parts of the device, several dialog boxes are available. You can use them to check the firmware version and other parameters that may be required when you encounter any problem and like to contact the manufacturer.

System Information

The dialog box "System Information" (see Fig. 10) summarizes general information about the device. The text "ID" shows the product identification, "Product No." is the unique product serial number. The entry "FW version" designates the firmware version, "FW date" is the compilation date of the firmware. In case of any trouble with the device, please use these values when contacting the manufacturer.

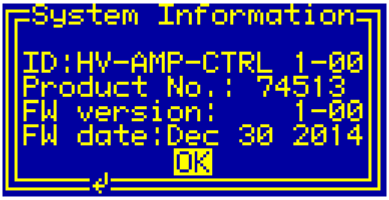
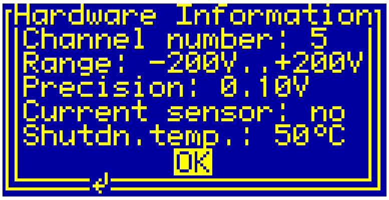

```
System Information
ID:HV-AMP-CTRL 1-00
Product No.: 74513
FW version:    1-00
FW date:Dec 30 2014
```

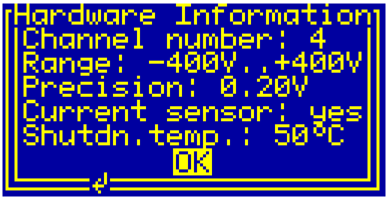
Fig. 10. Dialog box "System Information".

Hardware Information

The dialog box "Hardware Information" (see Fig. 11) summarizes general information about the device hardware.



```
Hardware Information
Channel number: 5
Range: -200V..+200V
Precision: 0.10V
Current sensor: no
Shutdn.temp.: 50°C
OK
```



```
Hardware Information
Channel number: 4
Range: -400V..+400V
Precision: 0.20V
Current sensor: yes
Shutdn.temp.: 50°C
OK
```

Fig. 11. Dialog box "Hardware Information".
The figures show the dialog box for two different devices: the left one with 5 installed channels for up to 200 V and the right with 4 installed channels for up to 400 V incl. current sensor.

The item "Channel number" shows the number of installed channels, the item "Range" displays the range of their output voltages.

The item "Precision" shows the allowable tolerance of the output voltage. If the difference between the set and measured output voltage is larger than this value, the state of the corresponding output changes to "failure" (see also section "Amplifier Control").

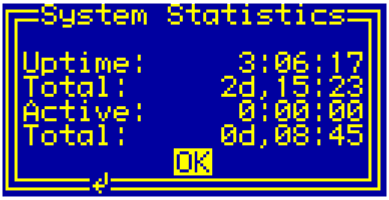
The item "Current sensor" indicates if the optional current sensor is installed. Note that the sensor is available on devices with four or less channels only.

The item "Shutdn.temp." shows the heatsink temperature at that the device shuts down, i.e. turns off the output voltages.

The values from the dialog box "Hardware Information" can be remotely accessed by software (see the function **`COM_AMP5_GetHardwareCapability`**), thus can be used to modify the software function according to the installed hardware features.

Statistics

The dialog box "Statistics" (see Fig. 12) shows information about the operation time of the device. The "Uptime" shows the time elapsed since powering on the device or since the last firmware restart. The next line "Total" shows the total operation time of the device.



```
System Statistics
Uptime:      3:06:17
Total:     2d,15:23
Active:      0:00:00
Total:     0d,08:45
```

Fig. 12. Dialog box "Statistics".

The time "Active" shows the active time of the device outputs elapsed since powering on the device or since the last firmware restart. The next line "Total" shows the total operation time of the amplifier.

Software Utilities

The software utilities can be found in the directory "Program" of the enclosed software package. Before using them, the virtual USB and/or LAN port drivers must be installed (see section "Driver Installation"). The utilities do not require any additional installation, you need only to copy them to a suitable directory on your computer.

Utility AmpTester

The AmpTester is a simple Windows™ program that runs in text mode and is supposed to be launched from a command line. It enables you to control the device remotely and check the operation parameters. Launching the utility `AmpTester.exe` without any parameters displays a simple help:

```
AmpTester
```

To start the program, you need to specify at least the number of the virtual COM port to which the device is attached:

```
AmpTester COMNumber
```

If the device is, for instance, attached to the port COM3, start the utility by the following command:

```
AmpTester 3
```

The program should start without any error message. In case of any problem, check whether the port number matches the system settings (see sections "Driver Installation").

To check the communication, press the key 'p' to obtain the product identification text. The device should respond as follows:

```
Product identification: HV-AMP-CTRL 1-00
```

If an error occurs, please consult the section "Functionality of the Software Interface". The tables 2 and 3 explain the possible error messages; they should help you to localize the reason for the software failure.

If the device responds properly, you may try other program commands. Press '?' to obtain the help listing of all available commands. Finally, press 'Esc' to exit the program.

In practice, you may prefer to use the command line mode instead of the interactive mode. The command line mode allows you, for instance, to save the complete commands in batch files for repeated usage. For instance, to view the set values of the output voltages of a device attached to COM3, enter the following command:

```
AmpTester 3 -o -t
```

Similarly, the command:

```
AmpTester 3 -e -t
```

displays the states of the outputs, the command:

```
AmpTester 3 -m -t
```

shows the measured values of the output voltages.

To set the output voltage of the channel 1 to 72.5 V, please enter

```
AmpTester 3 -O1 72.5 -t
```

to enable the channel 1, enter the command:

```
AmpTester 3 -E1 Y -t
```

Table 1 summarizes all allowable command line parameters of the program AmpTester. The parameters are processed from left to right. If an error in the command line is encountered, the program stops with an error text showing the allowed values of the parameters.

If the parameter `-t` is found, the program stops without processing any following parameter. If you do not specify the parameter `-t` at all, the program does not stop and enters the interactive mode after having processed the complete command line.

The quiet program mode reduces the program text output, contrary to that, the debug mode provides a detailed output for error analysis.

Tips

You can combine several command line parameters. For instance, the following command:

```
AmpTester 3 -O1 72.5 -E1 Y -t
```

set the output voltage of the channel 1 to 72.5 V and enables it.

Tab. 1. Command line parameters of the program AmpTester.

Parameter	Explanation
`-On f`	set the output voltage of the channel n to the value f
`-o`	get the output voltages
`-En b`	set the state of the channel n to the value b (=Y/N)
`-e`	get the state of all outputs
`-m, -M`	get the measured output voltages
`-i, -I`	get the interlock state
`-r, -R`	get the sensor current
`-h, -H`	get the housekeeping data
`-s, -S`	get the HV supply data of the amplifiers
`-c, -C`	get the hardware capabilities
`-d, -D`	get the device firmware date
`-v, -V`	get the device firmware version
`-p, -P`	get the product identification text
`-n, -N`	get the product number
`-u, -U`	get the device uptime
`-k, -K`	simulate the keypad, this starts an inter-active procedure that must be finished by pressing 'Esc'
`-l`	get display data in 8 steps
`-L`	get whole display data at once
`wn f g p, Wn f g p`	cycle the output voltage of the channel n between the voltages f and g, the cycle period is p milliseconds, the cycling must be finished interactively by pressing SPACE

Parameter	Explanation
`-b, -B`	get the device buffer status
`-z, -Z`	purge the communication
`-t, -T`	terminate the program
`-q, -Q`	quiet mode

`-g`
`-G`
`-?`
	debug mode
	debug mode with output into `Debug.txt`
	show the online help and terminate

To save the current device configuration, you may enter the command:

```
AmpTester 3 -q -o -t > AmpOutputs.txt
```

It outputs the set values of the output voltages into the file `AmpOutputs.txt`. Using this file, you can protocol the measurement or, in case of any error, restore the device parameters.

To setup all voltages, you may create a batch file:

```
AmpTester 3 -q -O1 10 -O2 20 -O3 30 -O4 40 -O5 50
   -o -t
```

By executing this file, the output voltages are set, then the set and measured values are displayed.

Utility FlashLoader

The FlashLoader is a simple 32-bit Windows™ program running in text mode and is supposed to be launched from a command line. It enables you to upgrade the firmware of the device. You should perform the upgrade if you have received or downloaded a new firmware file from the device manufacturer. Launching the utility `FlashLoader.exe` without any parameters displays a simple help text with the expected syntax of the command line.

Before upgrading the firmware, you should first test the device and the communication by verifying the current firmware version. To do so, connect the device via USB to the host computer and start the following command:

```
FlashLoader COMNumber Firmware.txt -v
```

where `COMNumber` is the number of the virtual COM port to which the device is attached and `Firmware.txt` is the file containing the current firmware. If the device is, for instance, attached to the port COM3, start the utility by the following command:

```
FlashLoader 3 Firmware.txt -v
```

If the proper parameters are specified, the verification starts and produces the following output:

```
Code file Firmware.txt from 01/02/2015, 12:00:00
Flash Loader 1.12
Verifying code file Firmware.txt
Verifying finished at Fri, 01/02/2015, 13:00:00
50258 (C452h) bytes processed, 49920 (C300h) bytes
verified
Resetting the target
Program finished ok
```

During the verify procedure, a message box is displayed at the device LCD informing the user that the flash loader has been activated. When the verify finishes, the device is restarted.

! **Attention:** Never start the utility if the device is active. The utility stops all processes running at the microcontroller, thus the monitoring is not available and the outputs would not be deactivated if a critical situation arises. To be completely sure that the device cannot be activated before the FlashLoader is started, open the interlock loop.

! Note that the firmware upgrade can be done using the USB interface only, the utility fails if you try to start it using the LAN interface.

If any error occurs, do not proceed with the firmware upgrade. To locate the reason for the error, follow the instructions described in the section "Utility AmpTester".

If the verify has succeeded, you may start the firmware upgrade by entering the command:

```
FlashLoader COMNumber Firmware.txt
```

where `COMNumber` is the number of the virtual COM port and `Firmware.txt` is the file with the new firmware. The program should produce the following output:

```
Code file Firmware.txt from 01/02/2015, 12:00:00
Flash Loader 1.12
Programming code file Firmware.txt
Programming finished at Fri, 01/02/2015, 13:00:00
94634 (171AAh) bytes processed, 94976 (17300h)
bytes programmed
Resetting the target
Program finished ok
```

During the programming procedure, a message box is displayed at the device display. When the programming finishes, the device is restarted with the new firmware.

If an error occurs, the flash loader utility at the microcontroller may remain active. This is the case if the message box at the device display is still present and the device did not restart. In this case, you may retry the action with the command line parameter '-i':

```
FlashLoader COMNumber Firmware.txt -i
```

This will prevent the utility at the host computer from initializing the flash loader utility at the microcontroller again and it will just try to reprogram the file `Firmware.txt`. If the error persists, contact the manufacturer.

! **Attention:** You must not power down the device if the firmware upgrade did not succeed. Otherwise, the device will not operate properly or might even not restart at all. Should this happen, it would be necessary to reprogram the device in the factory.

If the current firmware is damaged so that the device is inoperable, you may try to start the flash loader utility at the microcontroller manually. Press, simultaneously, the horizontal direction keys and hold them while powering on the device. If this small part of the firmware is still working, the flash loader will start. Then, try to launch the FlashLoader with the command line parameter '-s':

```
FlashLoader COMNumber Firmware.txt -s
```

This does not start the flash loader utility at the microcontroller but will only try to reprogram the file `Firmware.txt`. If an error occurs that you cannot solve, contact the manufacturer.

Utility RemoteControl

The RemoteControl is a 32-bit Windows™ GUI program that enables you to control the device remotely and obtain a hardcopy of the current LCD content.

To launch the utility, start the following command:

```
RemoteControl COMNumber
```

The number `COMNumber` is the number of the virtual COM port to which the device is attached. When started with a proper parameter, the program opens a dialog box that shows the current content of the LCD at the device (see Fig. 13). The direction keys and the encoder can be simulated by clicking the corresponding buttons or using the keyboard shortcuts. For the detailed description of them, see the online help.

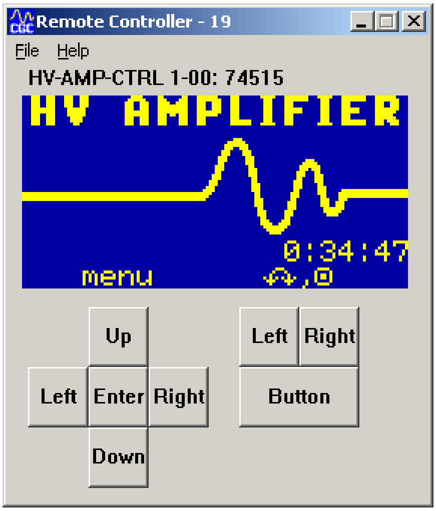


Fig. 13. Utility RemoteControl.

You may obtain a hardcopy of the LCD at any time by selecting the menu item File ► Save Hardcopy or by pressing Ctrl+S. The program opens a dialog box to select the destination bitmap file. The hardcopy is performed immediately without waiting for any confirmation.

Note that the utility RemoteControl continuously transfers display data over the USB or LAN interface. Compared to the idle state, this leads to a higher load of the microcontroller controlling the device. Further, the utility prevents any other program from accessing the device via the interface.

Driver Installation

Installation of the Virtual Port for the USB Interface

The virtual port driver is required for the operation of the device with a USB interface. The installation files are located in the directory "USB" of the enclosed software package. If you use the operating system MS Windows™, please note the following:

- As an alternative to the drivers from the enclosed software package, you can use the update function of the operating system at the host computer or download the most recent driver from the homepage of the manufacturer of the USB adapter. The drivers are located at the following address: http://www.ftdichip.com/Drivers/VCP.htm. Please choose the correct driver version according to your operating system.
- To install the driver, administrative rights are required.
- The installation is described in detail in one of the pdf documents located in the abovementioned directory. Please read this description carefully before starting the installation.
- After the installation, the number of the virtual port can be set. You can change the settings in the device manager by opening the settings of the device *USB Serial Port*. To modify the settings, administrative rights are required. The settings are applied immediately, you do not need to reboot the PC to activate them.

The software can also be used at computers running other operating systems than MS Windows™, for instance, Linux. You can launch the software utilities using the Windows™ API translator wine (see http://www.winehq.org/).

Starting with Linux Kernel 3.0.0-19, all FTDI devices are already supported without the necessity of compiling additional kernel modules. For more details, see the directory "USB" of the enclosed software package or consult the homepage of the manufacturer of the USB adapter: http://www.ftdichip.com/Drivers/VCP.htm.

The system has to be configured in the following way:

- Use, for instance, the program 'dmesg' to find out to which USB port the device is attached: Look for a line similar to "FTDI USB Serial Device converter now attached to ttyUSB0"

- Link the Linux device to the virtual COM port of wine:
  `ln -s /dev/ttyUSB0 ~ /.wine/dosdevices/com3`
  This assumes that the device is attached to ttyUSB0 and will be linked to COM3

Installation of the Virtual Port for the LAN Interface

The virtual port driver is required for the operation of the device with a LAN interface. The installation files are located in the directory "LAN" of the enclosed software package. If you use the operating system MS Windows™, please note the following:

- Locate the installation program in the enclosed software package or download an actual version of the software from the following address: http://tibbo.com/downloads/soi/tdst.html. Please choose the correct driver version according to your operating system.
- To install the driver, administrative rights are required.
- Start the installation program and choose the custom installation. You need to select just the core files and the start-menu shortcuts, other features are not required for the communication.
- Open the installed utility "Tibbo Connection Wizard" and proceed with installing the virtual port. Choose the desired COM number for the "Virtual Serial Port" and select the "Device Server" from the list. Let other settings unchanged and finish the installation. Note that the device that you wish to connect to must be accessible during installing the virtual port. The port driver is not digitally signed, you have to allow its installation despite any warnings.
- The installation is complete, you can connect the device via the selected COM port.
- Note that the COM number of a previously installed virtual port cannot be changed. You can, however, select the attached device in the settings of the virtual port. To do so, open the Windows™ device manager, locate the virtual port and open its properties. Select the tab "VSP Properties" and click on the button "Browse for DS...". Note that you need administrative rights for modifying the port properties. The settings are applied immediately, you do not need to reboot the PC to activate them.

Note that you may need to install several virtual ports if you like to connect several devices simultaneously. If you install just one virtual port but wish to communicate with more devices, you will need to select the proper device server before starting the communication, thus will need administrative rights.

Software Interface

The software interface for the device consists of a 32-bit dynamic link library `COM-AMP5.dll`. It is located in the directory "Program" of the enclosed software package. The software interface is a stand-alone software package, it does not require any additional library or driver, except the virtual port drivers (see section "Driver Installation").

The user functions in the dynamic link library `COM-AMP5.dll` can be called from any popular programming language. For the details, please consult the user manual of your compiler. The definition of the interface functions is located in the C-header file `COM-AMP5.h`. If your compiler cannot create an import library from the dynamic link library `COM-AMP5.dll`, please link the library `COM-AMP5.lib` instead of the dynamic link library to your project.

To simplify the integration in Pascal software packages, all functions of the software interface use the Pascal calling convention.

The sample program at the end of this section is written for the Borland C++ compiler. However, you should also be able to compile it with other C++ compilers without needing to modify it.

Functionality of the Software Interface

The software interface controls up to eight communication channels for the data transfer from/to different devices. Each communication channel must be opened before starting the communication. The open procedure assigns the selected communication channel to a virtual serial port. Next, the virtual serial port is configured and the port buffers are cleared. The used communication channel should be closed at the program end. If this fails to happen, the software interface does it automatically for you when the dynamic link library `COM-AMP5.dll` is unloaded from the system memory.

Most of the interface functions require as a first parameter the variable `PortNumber` that is the number of the communication channel, which should be used for the particular operation. This number is an unsigned integer that must be lower than `COM_AMP5_MAX_PORT`, i.e. must be in the range from 0 to 7.

The return value of most interface functions is a signed 32-bit number (`int`) describing the success of the particular operation. The last return value can be reloaded by the function **`COM_AMP5_State`**. Table 2 summarizes the possible return values together with the error messages, which can also be obtained by the function **`COM_AMP5_ErrorMessage`**. The error codes in Tab. 2 correspond to the codes `COM_AMP5_ERR_xxx` in the C-header file `COM-AMP5.h`. If a failure in data transfer has occurred, you can discover the reason by calling the functions **`COM_AMP5_IO_State`** and **`COM_AMP5_IO_ErrorMessage`**. The first one returns the last I/O error, the second one the corresponding error message (see Tab. 3.).

If you have troubles understanding the errors or establishing the communication, please contact the manufacturer of the device.

Tab. 2. Return values of the interface functions

Return value	Error message	Description
0	No error	The data transfer finished success-fully.
-2	Error opening the port	The port could not be opened. For the possible reasons, see Tab. 3.
-3	Error closing the port	The port could not be closed. For the possible reasons, see Tab. 3.
-4	Error purging the port	The port buffers could not be cleared.
-5	Error setting the port control lines	The port control lines could not be set.
-6	Error reading the port status lines	The port status lines could not be read.
-7	Error sending command	The data transfer to the device failed. For the possible reasons, see Tab. 3.
-8	Error sending data	
-9	Error sending termination character	
-10	Error receiving command	The data transfer from the device failed. For the possible reasons, see Tab. 3.
-11	Error receiving data	
-12	Error receiving termination character	
-13	Wrong command received	The device sent an unexpected re-sponse.
-14	Wrong argument received	
-15	Wrong argument passed to the function	One of the arguments passed to the function was out of the allow-able range.

Return value	Error message	Description
-50	Channel not implemented	The channel with the number passed to the function is not installed.
-51	Current sensor not implemented	The current sensor is not installed.
-100	Device not connected	The port status lines indicate that the device is not connected.
-101	Device not ready	The port status lines indicate that the device is not ready. This may be caused by a running background process. Repeat the operation to resolve the problem.
-102	Device state could not be set to not ready	The device did not react properly. Try to reset the communication or restart the device by powering it off and on.
-200	PortNumber out of range	The `PortNumber` must be lower than `COM_AMP5_MAX_PORT`.
-300	Memory for the port data could not be allocated	The port initialization failed and the requested amount of memory could not be allocated.
-400	Error opening the file for debugging output	The text file for debugging output could not be opened.
-401	Error closing the file for debugging output	The file for debugging output could not be closed.

Tab. 3. I/O errors

Return value	Error message	Description
0	No error	The data transfer finished successfully.
1	Port has not been opened yet	You attempted to use the communication channel before having opened it.
2	Cannot open the port	The specified port could not be opened. Either the port does not exist or it is being currently used by another program.
3	Cannot get the state of the port	The system could not get the state of the port.
4	Cannot set the state of the port	The system could not set the state of the port.
5	Cannot set the timeouts for the port	The system could not set the timeouts for the port.
6	Cannot clear the port	The system could not clear the port buffers.
7	Error reading data from the port	The system could not read data from the port. Most probably, no data is available because the device is either disconnected or does not respond.
8	Error writing data to the port	The system could not write data to the port.
9	Wrong data amount written to the port	The system could not write the proper data amount to the port.
10	Error setting the control lines of the port	The system could not set the state of the port control lines.
11	Error reading the status lines of the port	The system could not get the state of the port status lines.
12	Device is busy	The device is not ready to communicate.

Communication Control

Function COM_AMP5_Open

```
int pascal COM_AMP5_Open (WORD PortNumber,
  BYTE COMNumber);
```

Opens the specified communication channel and returns an error code according to Tab. 2.

The function opens the channel with the number `PortNumber` and attaches it to the serial port with the number `COMNumber`. The variable `PortNumber` must be lower than the number of implemented channels `COM_AMP5_MAX_PORT`. The serial port number `COMNumber` must point to a valid serial port to that the controller is attached. Note that this number is the number of the COM port, i.e. `COMNumber = 1` points to the port COM1. The function accepts all numbers that are supported by the operating system, thus any port between COM1 and COM255 can be used for the communication.

You must call this function prior to any other communication function. If the function returns an error, no data communication is possible.

Function COM_AMP5_Close

```
int pascal COM_AMP5_Close (WORD PortNumber);
```

Closes the specified communication channel and returns an error code according to Tab. 2.

You can use the function to free the port for another application.

If an application that has exclusively used the software interface `COM-AMP5.dll` finishes, the opened communication channel closes automatically. Thus, the programmer does not need to call the function `COM_AMP5_Close` explicitly.

Function COM_AMP5_Purge

```
int pascal COM_AMP5_Purge (WORD PortNumber);
```

Clears the port data buffers and returns an error code according to Tab. 2.

The function can be used to repair a disturbed communication. In case of a crash of a user program, this function should be called to erase data incorrectly received from the device.

The function `COM_AMP5_Purge` is automatically called by the function **`COM_AMP5_Open`**.

Function COM_AMP5_Buffer_State

```
int pascal COM_AMP5_Buffer_State
  (WORD PortNumber, BOOL & Empty);
```

Gets the state of the device's input data buffer in the variable `Empty`. The return value is an error code according to Tab. 2.

When a large data amount should be transferred to the device, this function can be used to ensure that the input data buffer contains enough free space. If the return value of the variable `Empty` is false, the input buffer is not empty and there is no guarantee that the device will be able to receive the data. This situation can occur if the device has just received a large amount of data and its processing has not yet finished. In such a case, the call to the function `COM_AMP5_Buffer_State` should be repeated after several milliseconds until the return value becomes `true`.

Function COM_AMP5_Device_Purge

```
int pascal COM_AMP5_Device_Purge
  (WORD PortNumber, BOOL & Empty);
```

Clears the device's output data buffer and gets the state of the device's input data buffer in the variable `Empty` like the function **`COM_AMP5_Buffer_State`**. The return value is an error code according to Tab. 2.

The function can be used to repair a disturbed communication. If the device does not respond properly, the function `COM_AMP5_Device_Purge` should be called repeatedly until it returns the value `true` in the variable `Empty`.

Amplifier control

Function COM_AMP5_Get_OutputState

```
int pascal COM_AMP5_Get_OutputState
  (WORD PortNumber, WORD Channel,
  BOOL & Enabled);
```

Gets the state of the specified channel in the variable `Enabled` and returns an error code according to Tab. 2.

The variable `Channel` specifies the device channel. It must be lower than `COM_AMP5_MAX_CHANNEL`, i.e. in the range between 0 and 4, otherwise the error `COM_AMP5_ERR_ARGUMENT` is reported. If the variable `Channel` specifies a channel that is not installed, the error `COM_AMP5_ERR_CHANNEL_RANGE` is reported (see Tab. 2). To prevent the error, obtain the number of the installed device channels first by calling the function **`COM_AMP5_GetHardwareCapability`**.

Function COM_AMP5_Set_OutputState

```
int pascal COM_AMP5_Set_OutputState
  (WORD PortNumber, WORD Channel,
  BOOL & Enabled);
```

Sets the state of a specified channel according to the variable `Enabled`, gets the set value back in the variable `Enabled`, and returns an error code according to Tab. 2.

For the variable `Channel`, see the description of the function **`COM_AMP5_Get_OutputState`**.

After calling the function `COM_AMP5_Set_OutputState`, you should check the return value in the variable `Enabled` to obtain the real state of the device. If you, for instance, try to enable a channel of a device with opened interlock loop (see the function **`COM_AMP5_Get_InterlockState`**), the enable procedure fails and the variable `Enabled` returns `FALSE`.

Function COM_AMP5_Get_OutputVoltage

```
int pascal COM_AMP5_Get_OutputVoltage
  (WORD PortNumber, WORD Channel,
  double & OutputVoltage);
```

Gets the output voltage of a specified channel in the variable `OutputVoltage`, and returns an error code according to Tab. 2.

For the variable `Channel`, see the description of the function **`COM_AMP5_Get_OutputState`**.

The return value in the variable `OutputVoltage` is the set output voltage in Volts.

Function COM_AMP5_Set_OutputVoltage

```
int pascal COM_AMP5_Set_OutputVoltage
  (WORD PortNumber, WORD Channel,
  double & OutputVoltage);
```

Sets the output voltage of a specified channel according to the variable `OutputVoltage`, gets the set value back in the variable `OutputVoltage`, and returns an error code according to Tab. 2.

For the variable `Channel`, see the description of the function **`COM_AMP5_Get_OutputState`**.

The return value in the variable `OutputVoltage` is the set output voltage in Volts. Note that the voltage is applied on the output only if the channel is enabled.

After calling the function `COM_AMP5_Set_OutputVoltage`, you should check the return value in the variable `OutputVoltage` to obtain the real value set by the device. If you, for instance, try to set an output voltage outside the specified range, the device truncates the value and the variable `OutputVoltage` returns the modified value.

Monitoring

Function COM_AMP5_Get_OutputMonitor

```
int pascal COM_AMP5_Get_OutputMonitor
    (WORD PortNumber, WORD Channel,
    double & OutputVoltage);
```

Gets the measured output voltage of a specified channel in the variable `OutputVoltage` and returns an error code according to Tab. 2.

For the variable `Channel`, see the description of the function **`COM_AMP5_Get_OutputState`**.

The return value in the variable `OutputVoltage` is the set output voltage in Volts. Note that the precision of the measurement decreases if the output voltage is - for instance due to overloading - significantly different from the expected value (see also section "Amplifier Monitor").

Function COM_AMP5_Get_HousekeepingData

```
int pascal COM_AMP5_Get_HousekeepingData
    (WORD PortNumber, double & VoltageSupply,
    double & Voltage5V, double & Voltage3V,
    double & Temperature);
```

Gets the housekeeping data and returns an error code according to Tab. 2.

The return values in the variables `VoltageSupply`, `Voltage5V`, and `Voltage3V` are supply voltages in Volts. The return value in the variable `Temperature` is the microcontroller temperature in °C. For more details, see section "Housekeeping Data".

Function COM_AMP5_Get_HVSupplyData

```
int pascal COM_AMP5_Get_HVSupplyData
  (WORD PortNumber, double & VoltagePos,
  double & VoltageNeg,
  double & Temperature);
```

Gets the high-voltage supply data and returns an error code according to Tab. 2.

The return values in the variables `VoltagePos` and `VoltageNeg` are the supply voltages of the output amplifiers in Volts. The return value in the variable `Temperature` is the heatsink temperature in °C. For more details, see section "Power Monitor".

Function COM_AMP5_Get_InterlockState

```
int pascal COM_AMP5_Get_InterlockState
  (WORD PortNumber, BOOL & Closed);
```

Gets the state of the interlock loop in the variable `Closed` and returns an error code according to Tab. 2.

Note that the channels of a device can be enabled only if the interlock loop is closed, i.e. if the variable `Closed` returns `TRUE` (see also the function **`COM_AMP5_Set_OutputState`**).

Function COM_AMP5_Get_CurrentMonitor

```
int pascal COM_AMP5_Get_CurrentMonitor
  (WORD PortNumber, double & SensorCurrent);
```

Gets the measured sensor current in the variable `SensorCurrent` and returns an error code according to Tab. 2.

The return value in the variable `SensorCurrent` is the measured sensor current in Amperes (for details see section "Current Monitor")

Note that if you call this function on a device without an installed current sensor, the error `COM_AMP5_ERR_SENSOR` is reported (see Tab. 2). To prevent the error, obtain the presence of the current sensor first by calling the function **`COM_AMP5_GetHardwareCapability`**.

Error Handling

Function COM_AMP5_State

```
int pascal COM_AMP5_State (WORD PortNumber);
```

Returns the state of the software interface according to Tab. 2.

This function can be used to obtain the last return value of an interface function.

The function does not have any influence on the communication and can be called at any time.

Function COM_AMP5_ErrorMessage

```
const char * pascal COM_AMP5_ErrorMessage
    (WORD PortNumber);
```

Returns the error message corresponding to the state of the software interface. The return value is a pointer to a null-terminated character string according to Tab. 2.

The function does not have any influence on the communication and can be called at any time.

Function COM_AMP5_IO_State

```
int pascal COM_AMP5_IO_State
    (WORD PortNumber);
```

Returns the interface state of the serial port according to Tab. 3.

The function can be used to obtain the result of the last I/O operation at the port.

The function does not have any influence on the communication and can be called at any time.

Function COM_AMP5_IO_ErrorMessage

```
const char * pascal COM_AMP5_IO_ErrorMessage
  (WORD PortNumber);
```

Returns the error message corresponding to the interface state of the serial port. The return value is a pointer to a null-terminated character string according to Tab. 3.

The function does not have any influence on the communication and can be called at any time.

Function COM_AMP5_OpenDebugFile

```
int pascal COM_AMP5_OpenDebugFile
  (WORD PortNumber, const char * FileName);
```

Opens a text file for debugging output and returns an error code according to Tab. 2.

You can use the debugging output in case of any communication problems. If a communication error occurs, a detailed error analysis is saved in the specified text file.

The function does not have any direct influence on the communication and can be called at any time.

Function COM_AMP5_CloseDebugFile

```
int pascal COM_AMP5_CloseDebugFile
  (WORD PortNumber);
```

Closes the file for debugging output and returns an error code according to Tab. 2.

The function does not have any direct influence on the communication and can be called at any time.

Various Functions

Function COM_AMP5_Version

```
WORD pascal COM_AMP5_Version();
```

Returns the version of the software interface (the dynamic link library `COM-AMP5.dll`).

The function should be used to check whether a software interface with the correct version is being used. The function should be called prior to any other function of the software interface.

The return value is an unsigned 16-bit integer (WORD). The higher byte contains the main version number, the lower byte the subversion, i.e. the version order within the main version. The identical main version number of two different libraries `COM-AMP5.dll` means that they implement the same functions. The difference in the subversion denotes that various corrections have been made to the interface functions. If the main version number is different, there is no guarantee that the library can be used. Mostly, the user software has to be recompiled or modified to match the new definition of the software interface.

Function COM_AMP5_GetHardwareCapability

```
int pascal COM_AMP5_GetHardwareCapability
  (WORD PortNumber, BYTE & ChannelNumber,
  double & MinVoltage, double & MaxVoltage,
  double & Precision, BOOL & SensorPresent,
  double & ShutdownTemperature);
```

Gets the device hardware capabilities and returns an error code according to Tab. 2. The return values are the parameters that can be viewed by the dialog box "Hardware Information" (see section "Hardware Information").

The return value in the variable `ChannelNumber` is the number of installed amplifier channels. If you try to access data of a not installed channel, the functions **`COM_AMP5_Get_OutputState`**, **`COM_AMP5_Set_OutputState`**, **`COM_AMP5_Get_OutputVoltage`**, **`COM_AMP5_Set_OutputVoltage`**, and

`COM_AMP5_Get_OutputMonitor` will return the value `COM_AMP5_ERR_CHANNEL_RANGE` (see Tab. 2).

The return values in the variables `MinVoltage` and `MaxVoltage` are the output voltage limits of the installed amplifier channels in Volts. If you try to set the output voltage to a value outside these limits, the controller will adjust it.

The return value in the variable `Precision` is the allowable tolerance of the output voltage in Volts. If the difference between the set and measured output voltage is larger than this value, the state of the corresponding output changes to "failure" (see also section "Amplifier Control").

The return value in the variable `SensorPresent` indicates if the optional current sensor is installed. If you try to access data of a not installed sensor, the function **`COM_AMP5_Get_CurrentMonitor`** will return the value `COM_AMP5_ERR_SENSOR` (see Tab. 2).

The return value in the variable `ShutdownTemperature` is the heatsink temperature in °C at that the device shuts down, i.e. turns off the output voltages.

Function COM_AMP5_GetUptime

```
int pascal COM_AMP5_GetUptime
  (WORD PortNumber, DWORD & Seconds,
  WORD & Milliseconds);
```

Gets the device uptime in the variables `Seconds` and `Milliseconds`. The function return value is an error code according to Tab. 2.

The device uptime is the time elapsed from the last (re)start of the device. The function can be used, for instance, to see whether the device restarted unexpectedly.

Function COM_AMP5_FW_Ver

```
int pascal COM_AMP5_FW_Ver (WORD PortNumber,
  WORD & Version);
```

Gets the device's firmware version in the variable `Version`. The function return value is an error code according to Tab. 2.

The return value in the variable `Version` should be used to check whether the software interface is in an appropriate version.

The return value is similar to the return value of the function **`COM_AMP5_Version`**. It is an unsigned 16-bit integer (WORD) containing the main version and the subversion numbers. The software interface is compatible to the firmware if the main version numbers are identical.

Function COM_AMP5_FW_Date

```
int pascal COM_AMP5_FW_Date (WORD PortNumber,
  char * DateString);
```

Gets the device's firmware date in the variable `DateString`. The function return value is an error code according to Tab. 2.

The return value in the variable `DateString` is a null-terminated character string with the firmware compilation date. The buffer passed to the function must be created before the function call, it must be at least 16 bytes long.

Function COM_AMP5_Prod_No

```
int pascal COM_AMP5_Prod_No (WORD PortNumber,
  DWORD & Number);
```

Gets the device's product number in the variable `Number`. The function return value is an error code according to Tab. 2.

Function COM_AMP5_Prod_ID

```
int pascal COM_AMP5_Prod_ID (WORD PortNumber,
  char * Identification);
```

Gets the device's product identification in the variable `Identification`. The function return value is an error code according to Tab. 2.

The return value in the variable `Identification` is a null-terminated character string with the product identification. The buffer passed to the function must be created before the function call; it should be 81 bytes long.

Sample Communication Program

```
#include <stdio.h>
#include <windows.h>
#include "COM-AMP5.h"

// handle a possible error:
static int Error (int code)
   {
   printf ("%s\n", COM_AMP5_ErrorMessage (0));
   return code;
   }

int main (int argc, char * argv[])
   {
   if (argc != 2)
      {
      printf ("The number of the COM port must be specified!\n");
      return -1;
      }
   // check the DLL version:
   WORD DLL_Version = COM_AMP5_Version();
   if ((DLL_Version & 0xFF00) != 0x0100)
      {
      printf ("Wrong DLL Version: expected 1.x, found %d.%02d\n",
         DLL_Version / 0x100, DLL_Version & 0xFF);
      return 1;
      }

   // open the port 0:
   WORD COMNumber;
   sscanf (argv[1], "%hu", &COMNumber);
   if (COM_AMP5_Open (0, COMNumber))
      return Error (2); // handle a possible error

   // Get device hardware capabilities:
   BYTE ChannelNumber; BOOL SensorPresent;
   double MinVoltage, MaxVoltage, Precision;
   double ShutdownTemperature;
   if (COM_AMP5_GetHardwareCapability (0,
      ChannelNumber, MinVoltage, MaxVoltage,
      Precision, SensorPresent, ShutdownTemperature))
      return Error (3); // handle a possible error
   // Get measured output voltages
   printf ("Measured output voltages:\n");
   for (WORD Channel = 0; Channel < ChannelNumber; Channel++)
      {
      double OutputVoltage;
      if (COM_AMP5_Get_OutputMonitor (0, Channel, OutputVoltage))
         return Error (4); // handle a possible error
      printf ("\tCh%hd: %+.3lfV\n", Channel, OutputVoltage);
      }

   // Get interlock state
   BOOL Closed;
   if (COM_AMP5_Get_InterlockState (0, Closed))
      return Error (5); // handle a possible error
   printf ("Interlock: %s\n", Closed ? "closed" : "open");

   // Get measured sensor current
   if (SensorPresent)
      {
      double SensorCurrent;
      if (COM_AMP5_Get_CurrentMonitor (0, SensorCurrent))
         return Error (6); // handle a possible error
      printf ("Current sensor: %+.3leA\n", SensorCurrent);
      }

   //finish the program:
   return 0;
   }
```