High-Voltage Supply Controller Software

(for PRESAMPLER and BPC)

User Manual and Reference

v1.1

Henk Boterenbrood
NIKHEF-H, Amsterdam

January, 1995

CONTENTS

1. Introduction

This document is a reference and a user manual for the software running on the 80C552 microcontroller which forms the heart of the high-voltage powersupply module which was developed to supply high-voltage to the photomultiplier tubes (PMs) of the **Presampler** (**PRES**) and the **Beam Pipe Calorimeter** (**BPC**) which form parts of the **ZEUS** detector at DESY, Hamburg. Hardware and software are entirely based on a design for the **Small-angle Rear Tracking Detector** (**SRTD**).

The 'normal' user should be aware that this document contains much more information than he needs to know or wants to know....

The microcontroller sets, monitors and controls the voltage and duty-cycle of 6 powersupplies, each of which will supply high-voltage to a number of PMs.
Additionally it sets and monitors an auxiliary powersupply which supplies the power to the high-voltage supplies.
In the following we will refer to the combination of controller, auxiliary supply and the high-voltage supplies as **HV-controller**.

2. Basic operation

The HV-controller is connected to the outside world by means of an RS232 serial connection operating at a baudrate of 9600.
Through this serial line the HV-controller can be controlled and the status of the powersupplies monitored. Also the initial calibration of each powersupply is performed under control of the user sitting at a terminal at the other end of the serial line.
The HV-controller behaves like a slave, which means it will start performing an action (e.g. switching-on a powersupply or reporting about the current status) only after receiving a command on the serial line.

All commands and messages to / from the HV-controller are in the form of ASCII-strings so that it is sufficient to connect a terminal to the RS232-line in order to operate the powersupplies, although for regular and continuous monitoring it will be more convenient to connect a hostcomputer to the RS232-line, to do the monitoring under software control.

It is possible to have up to 255 HV-controllers connected to one RS232 serial line; hex-switches on the controller-board can be set to uniquely identify an HV-controller; an HV-controller will only respond to commands addressed to it explicitly or to broadcasted commands (commands containing a wildcard address, see section 4); an arbitration scheme (hardware / software) prevents more than one controller at a time accessing the serial line for output.

The on-board 80C552 microcontroller controls the powersupply voltages by means of a set of Digital-to-Analog converters (DAC) with 6 bits resolution; a high-voltage is set combining two DAC outputs, a coarse and a fine setting, resulting in a resolution close to 12 bits. The operating voltage ranges from 800V to 1200V (BPC: 600V-1000V), with a resolution of about 0.15 V. The accuracy of the set voltages is within ca. ± 1 V with the control process running and within ca. ± 3 V without.

The auxiliary supply is also set using two 6 bit DACs, and is set at a nominal voltage of about 75 V.

Each supply has an enable / disable input by which the controller switches the supply on or off.

The voltage and current-load of all powersupplies (the auxiliary supply and the high-voltage supplies) are measured using the micro-controller's on-chip Analog-to-Digital Converter (ADC) with a resol-ution of 10 bits (about 0.4 V of high-voltage resolution and about 1 µA of current-load resolution).

Because of the digital-to-analog and analog-to-digital conversions involved each powersupply has been calibrated so that it can be used in a meaningful way; the calibration constants are stored in on-board EEPROM.

In normal active operation (after the user has started the control and monitoring process) the controller takes samples of the current-load and voltages of the active powersupplies, typically with a rate of 10 Hz; with a rate of typically 1 Hz averages are calculated and the voltage is adjusted if necessary and checks are made on the current settings and loads to see if everything still falls within preset safety and control margins; if this is not the case for a particular powersupply, the supply is switched off; this event is recorded in a statusword which the controller keeps for each powersupply.

The controller also records minimum and maximum values of duty-cycles and voltages measured which can be monitored by the user / host (not for the auxiliary powersupply).

Therefore the user / host should at regular intervals check the status of the powersupplies and take appropriate action whenever a supply is not functioning properly.

Settings and calibration parameters can be stored permanently in the on-board EEPROM, so that once stored they are used after every (power-up) reset.

The software allows for a new program to be downloaded if e.g. a new version of the software is available, but it might also be a totally different program, e.g. some testprogram. The program is stored in the onboard EEPROM. If certain programming rules are obeyed it is possible to switch from one program to the other and back.

3. Automatic trip recovery

An automatic trip recovery feature has been added to the controller software:

if the control process is running and a HV-supply is switched off by the software (a 'trip') because of a value running out of bounds, it will be switched on again after a short period of time; if a number of consecutive trips occur shortly after one another the HV-supply will remain switched off permanently, until e.g. the user switches it on again explicitly.

The number of consecutive trips that causes the HV-supply to be switched off permanently can be set by the user; if this number is set to zero or to 1 the controller behaves as if the automatic trip recovery is not enabled.

The total number of trips during an 'enabled'-period for every HV-supply is stored by the controller and these 'trip-counters' can be inspected by the user.

After a trip the HV-supply is switched on again after 5 sample-periods; after the supply automatically has been switched on the usual control-delay period applies.

4. Command/reply protocol

A command to the HV-controller has the following syntax:

<**s**><**n**>[**.**<**m**>]<**cmd**>[<**par**>]<CR>

with

s	= command tag ('**P**' for PRES, '**B**' for BPC).
n	= address of the HV-controller (0-255, or *; * meaning all HV-controllers connected to this RS232 line) this command applies to.
m	= number of the powersupply on this HV-controller the controller the command applies to (0, 1, 2, ... or *; * meaning all high-voltage powersupplies; number 0 is the auxiliary supply and often has to be addressed separately); if the preceding period and **m** are not present it is assumed to be *.
cmd	= three-letter command mnemonic.
par	= numeric parameter (in the form of an ASCII string), but only if the command requires a parameter; if the command requires a parameter and the parameter is missing it is assumed to be zero.

A command will always result in a reply, unless the HV-controller address is not correct or a controller with the specified address is not connected to the serial line; every HV-controller reads all command-lines, but will ignore all lines that do not contain its address.

Lines are interpreted by the HV-controller on a line-by-line basis: only after a carriage-return <CR> has been received the input is interpreted and processed and an acknowledge or error message is returned.

A reply from the HV-controller has the following syntax:

<**r**><**n**>**.**<**m**><**cmd**>**.**[<**par1**>**.**<**par2**>...]<LF><CR>

with

r	= reply tag ('**p**' for PRES, '**b**' for BPC).
n	= address of the HV-controller (0-255); if all HV-controllers were addressed by means of a *-wildcard every individual HV-controller will send a reply.
m	= number of the powersupply on this HV-controller the command applied to (0, 1, 2, ... or *; * meaning all high-voltage powersupplies; number 0 is the

auxiliary supply); if the preceding period and **m** were not present in the command in the reply **m** will be a *-wildcard character.

cmd = the same three-letter command mnemonic if the command was processed successfully, "**ERR**" otherwise.

par1,
par2 = return value(s) (in the form of ASCII strings, separated by periods); the number of return values depends on the command given.

4.1 Controller operation examples

Assuming that all settings are stored in EEPROM, issuing the following commands to the (PRES) HV-controller(s) are sufficient to switch on all powersupplies (the auxilary supply (no. 0) has to be switched on separately) and start the measure-and-adjust process:

```
P*.0ENA
P*ENA
P*CTR1
```

Assuming that all voltage settings stored in EEPROM should be overruled by a new setting of 980V (for every HV-supply, but not for the auxiliary supply of course):

```
P*.0ENA
P*SVO980
P*ENA
P*CTR1
```

Subsequently each HV-controller at regular intervals should at least be questioned about the status of its powersupplies:

```
P*RSS
```

or each controller separately (assuming we have e.g. 3 controllers):

```
P1RSS
P2RSS
P3RSS
```

A full status of one particular powersupply can be requested by (for example):

```
P2.1RSA
```

5. Calibration of the powersupplies

When we calibrate a powersupply we determine the parameters that define the conversion functions between DAC-values and Volts, between Volts and ADC-values and between current-loads and ADC-value. Each of these transfer functions is assumed to be linear, which in practice proves to be a valid assumption.

The voltage **V** as a function of the coarse-setting DAC value $\mathbf{D_{coarse}}$ is:

$$\mathbf{V = aD_{coarse} + b}$$

The additional voltage **? V** added by the fine-setting DAC-value $\mathbf{D_{fine}}$ is:

$$\mathbf{?\,V = a'D_{fine}}$$

The ADC-value $\mathbf{A_V}$ as a function of voltage **V** is:

$$\mathbf{A_V = cV + d}$$

and finally ADC-value $\mathbf{A_I}$ as a function of current-load value **I** (in units of 0.1 microAmperes) and Voltage **V** is:

$$\mathbf{A_I = eI + fV + g}$$

In order to be able to calculate parameters **a**, **b**, **a'**, **c**, **d**, **e**, **f** and **g,** it is sufficient to take 2 samples of each transfer function.

The accuracy of the set high voltage is dependent mainly on the accuracy and stability of the parameters describing the measured ADC-value-to-voltage conversion, i.e. parameters **c** and **d**.

The difference between the measured voltage and the requested voltage is used to adjust the current set voltage, independent of what the set voltage actually is according to the DAC-value-to-voltage settings on the two DACs; that explains why the accuracy of the set high voltage depends mainly on the accuracy and stability of parameters **c** and **d**.

Next follows an example of a complete calibration procedure for a (PRES) HV-controller; the command lines to be typed by the user are in bold text type; see the chapter on calibration commands for a full explanation of the commands used; voltage values and calibration parameters shown are just examples; they will be different for each HV-controller calibrated.

Switch to calibration-mode:

P1CAL
p1.*.CAL.1

Calibration of the auxiliary supply voltage[1] should be done first; it goes in the same way as the calibration of a high-voltage supply.

Calibration of HV-supply 1:

P1.1CAV
p1.1CAV.10.0
P1GVO7658
p1.1GVO.7658.90.0
P1GVO11553
p1.1GVO.11553.50.0
P1GVO9576
p1.1GVO.9576.50.100
P1GVO9662

Now calibration constants **a**, **b**, **a'**, **c** and **d** are calculated:
p1.1GVO.9662.8098.715.146.1771.-1204

The same voltage calibration procedure is to be followed for all other HV-supplies.

Calibration of a HV-supply's current-load, the first step without any external load (the set voltage dependency and offset of the current-load are determined: factor **fV + g** in the above formula):

P1.1CAC
p1.1CAC.75.10.0.0.90.0.0.50.0

Now the load on the supply should be increased, e.g. by connecting a resistor to the first[2] of the dynode high-voltage outputs, and the current through this resistor should be measured (if any of the other outputs is used a multiplication factor should be applied to the measured current before it is entered) in units of 0.1 μA:

P1GCU2395

Now calibration constants **e**, **f** and **g** are calculated:
p1.1GCU.2395.120.342.4370

The same current-load calibration procedure is to be followed for all other high-voltage supplies.

[1] To be measured at pin 2 of the cascade-to-microboard connector.
[2] = lowest voltage branch (on the box: on the left hand side of the HV-connector, having the 'COMM' connector on the right hand side.

Now save all parameters in EEPROM:

```
P1.0SVP
p1.0SVP.8098.715.146.1771.-1204.100.300.0
P1.1SVP
p1.0SVP.8030.721.147.1756.-1201.132.317.3345
P1.2SVP
p1.0SVP.8180.725.146.1758.-1209.121.376.5460
etc. etc.
```

6. Controller commands

Below two lists of command mnemonics and their symbolic name are shown; the first table contains the commands used during normal operation, the second table contains the commands used when the powersupplies are being calibrated; in principle commands from this second group are only used once; calibration parameters are subsequently stored in the onboard EEPROM and are not expected to change (much) in time.

Mnem	Name
CTR	CONTROL_MODE
DEE	DELETE_EEPROM_PROGRAM
DIS	DISABLE_SUPPLY
ENA	ENABLE_SUPPLY
HLP	HELP
LHX	LOAD_INTELHEX
RCA	READ_CURRENT_ADC
RCU	READ_CURRENT
RDC	READ_DARKCURRENT
RPS	READ_PROCESSOR_STATUS
RSA	READ_SUPPLY_ALL
RSE	READ_SETTINGS
RSS	READ_SUPPLY_STATUS
RST	RESET
RVA	READ_VOLTAGE_ADC
RVO	READ_VOLTAGE
SCD	SET_CONTROL_DELAY
SCF	SET_CONTROL_FREQUENCY
SMC	SET_MAXIMUM_CURRENT
SMT	SET_MAX_TRIPS
SPY	SPY_MODE
SSF	SET_SAMPLE_FREQUENCY
SVO	SET_VOLTAGE
SVS	SAVE_SETTINGS
SWI	SWITCH_PROGRAM

Mnem	Name
CAC	CALIBRATE_CURRENT
CAL	CALIBRATION_MODE
CAV	CALIBRATE_VOLTAGE
DEP	DELETE_PARAMETERS
GCU	GET_CURRENT
GVO	GET_VOLTAGE
RPA	READ_PARAMETERS
SDC	SET_DAC_COARSE
SDF	SET_DAC_FINE
SVP	SAVE_PARAMETERS

A description and explanation of each command, with examples using PRES HV-controller address 1 and powersupply address 2 follow.

It is always possible to use a wildcard for the HV-controller address (except for the **HLP** command); for some commands it is not allowed to use a wildcard as a power-supply address and for several commands the use of a powersupply address is meaningless altogether and is then left out.

The cases where a command returns an error are not considered here; see section 7 for a description of possible error conditions.

6.1 Commands for normal operation

6.1.1 CTR

command: **P1CTR1** or **P1CTR0**
reply: **p1.*CTR.1** or **p1.*CTR.0**

Switches the control mode on or off; if the parameter in the command is 1 the control process is started, if it is 0 this process is stopped. Supplying a powersupply address is meaningless.

The control process measures and adjusts the HV-supplies that are enabled (see **ENA**) according to the set voltage, measure and control frequencies defined (see **SVO**, **SSF**, **SCF**).

If the powersupply still has to be set to the desired voltage when the control process is started the control process will wait a certain period of time as defined by the set control delay (see **SCD**) in order to let the voltage settle, before starting to regulate it.

6.1.2 DEE

command: **P1DEE**
reply: **p1DEE.**

Makes the program stored in EEPROM invalid; subsequent **SWI** commands do not work anymore, until a new program is downloaded.

6.1.3 DIS

command: **P1.2DIS**
reply: **p1.2DIS.**

Disables one powersupply or all HV-supplies if the *-wildcard is used; the DACs for this supply are set to zero.

6.1.4 ENA

command: **P1.2ENA**
reply: **p1.2ENA.**

Enables one powersupply or all HV-supplies if the *-wildcard is used. If the control process is active it will take care of setting the voltage to the required value. If the control process is not active the voltage will go to a value corresponding to a DAC value of zero or to a value previously set with the **SVO** command.

The auxiliary powersupply is not controlled by the control process and thus gets immediately set to its requested voltage when it is enabled or by the **SVO** command, independent of the control process.

6.1.5 HLP

command: **P1HLP**
reply: <version_number>
<list_of_mnemonics>
p1.*HLP.

The version number and the list of mnemonics are only produced when the controller is in 'spy'-mode (see **SPY**). Supplying a powersupply address is meaningless. A controller address wildcard is not allowed.

6.1.6 LHX

command: **P1LHX**
<file_in_IntelHex_format>
reply: **p1.*LHX.9872**

After the command is given the controller will keep on reading lines and interpreting them as 'Intel-Hex' formatted lines until a socalled end-of-data line is read or if the first chararacter of a line is 'Q' (to enable the user to escape from the command when e.g. a wrong file without an end-of-data line was downloaded).

The lines are parsed and written to RAM in binary form (the original file is in ASCII format). When the end-of-data line has been received the EEPROM is programmed, which might take several seconds depending on the size of the program. Then a reply as shown above is returned in which the number represents the number of bytes of RAM used to temporarily store the code file.

Supplying a powersupply address is meaningless.

If an error occurred somewhere in the process the reply will be:

P1.*ERR.<line_no>**.**<error_no>

in which <line_no> is the number of the line in the 'Intel-Hex' formatted file which led to the error and <error_no> is a number stating the type of error that occurred (see section 7).

6.1.7 RCA

command: **P1.2RCA**
reply: **p1.2RCA.485**

Reads the current value of the current-load of a powersupply in ADC-units (a number between 0 and 1023); if the *-wildcard is used the reply contains the duty-cycle ADC-value of all HV powersupplies, e.g.:

p1.*RCA.876.653.485.322.426.376

6.1.8 RCU

command: **P1.2RCU**
reply: **p1.2RCU.256**

Reads the current value of the current-load of a powersupply and displays it in units of 0.1 µA; if the *-wildcard is used the reply will contain the current-loads of all HV-powersupplies, e.g.:

p1.*RCU.870.1363.250.345.563.723

6.1.9 RDC

command: **P1RDC**
reply: **p1RDC.423.369.401.345.389.432**

Reads the socalled dark-current values of all HV-powersupplies in ADC-units; this value is the zero-load offset of the current in ADC-units for the currently requested high-voltage value of each supply. This information is of no interest to the common user.

6.1.10 RPS

command: **P1RPS**
reply: **p1.*RPS.11.0**

Returns the software version number and whether the code is running from EEPROM (1) or not (0). In the example above the version is **1.1** and the code is not running from EEPROM (but EPROM). Supplying a powersupply address is meaningless.

6.1.11 RSA

command: **P1.2RSA**
reply: **p1.2RSA.0.1000.1000.1020.970.1070.560.108.775.345.0.0**

Returns current status (see **RSS**), set and measured values of one particular powersupply. A single powersupply address has to be provided; a wildcard is not allowed. The values returned are respectively:

- the supply status,
- last measured average voltage (V),
- requested voltage (V),
- last set voltage (V),
- minimum measured voltage (V),
- maximum measured voltage (V),
- last measured average current-load (µA*10),
- minimum measured current-load (µA*10),
- maximum measured current-load (µA*10),
- the currently valid dark-current (ADC-units),
- the trip counter of this supply
- and the last error status that occurred.

6.1.12 RSE

command: **P1RSE**
reply: **p1.*RSE.1.100.10.2.1000.75.1020.1020.1020.1020.1020.1020.2**

Returns the current settings, being:

- controlprocess on/off (0=off, 1=on),
- the sample frequency (Hz*10),
- the frequency of regulating (Hz*10),
- the control delay (s),
- the maximum current-load (µA*10),
- the required voltages for all supplies (the HV-supplies and the auxiliary supply) in Volts
- and the number of consecutive trips after which a HV-supply will be switched off permanently (no automatic trip recovery anymore).

Supplying a powersupply address is meaningless.

Note that the current frequencies are multiplied by a factor of 10. In the example above the frequencies are 10 Hz and 1 Hz respectively and the control delay is 2 seconds; the control process will wait 2 seconds in order to let the voltage settle before starting to regulate the supply voltage (if the control-process is active); the maximum current-load is 100 µA.

6.1.13 RSS

command: **P1RSS**
reply: **p1.*RSS.0.3.1.0.0.0.0.0.2.0.0.0.0.0**

Returns the current on/off-status of all supplies (the first 7 numbers in this reply; the auxiliary supply and 6 HV-supplies respectively) and 7 trip counters (one for each supply; the counter of the auxiliary supply --the first number-- will always be zero, because it is never automat-ically switched off). Supplying a powersupply address is meaningless.

Per supply a byte describes whether the supply is switched on or off and if switched off, whether there was an abnormal situation that caused it to be switched off, where set bits have the following meaning:

0x01	disabled
0x02	disabled due to measured current-load above allowed maximum
0x04	disabled due to measured voltage out of allowed range
0x08	disabled due to set voltage out of allowed range
0x10	disabled due to set voltage out of absolute supply range
0x20	disabled due to on-board powerfailure
0x40	error occurred in setting a DAC belonging to this supply (I^2C-bus)
0x80	not defined
0x100	error occurred in enabling / disabling a supply (I^2C-bus)

In the example at the beginning of this paragraph the reply shows that all HV-supplies and the auxiliary supply are switched on except supply 1 and 2; supply 1 is switched off because its current-load went above its allowed maximum (and it tripped twice), and supply 2 is just switched off (no abnormal cause).

6.1.14 RST

command: **P1RST**
reply: none

Causes the controller to be reset by its watchdog timer; parameters, constants etc. are restored to their default or EEPROM-stored values. (If running from EEPROM there will be a switch back to the program in EPROM.)

6.1.15 RVA

command: **P1.2RVA**
reply: **p1.2RVA.485**

Reads the current value of the voltage of a powersupply in ADC-units (a number between 0 and 1023); if the *-wildcard is used the reply will

contain the voltage ADC-value of all HV-powersupplies, e.g.:
p1.*RDA.876.53.485.567.876.532

6.1.16 RVO

command:	**P1.2RVO**
reply:	**p1.2RVO.1025**

Reads the current value of the voltage of a powersupply; if the *-wildcard is used the reply will contain the voltages of all HV-powersupplies, e.g.: **p1.*RDU.1023.1024.1021.1023.1024.1020**

6.1.17 SCD

command:	**P1SCD5**
reply:	**p1.*SCD.5**

Sets the control delay for all HV powersupplies to the requested value. The control delay is the delay in seconds taken into account by the control process when the supply is switched on or when the set voltage is changed; this in order to let the voltage settle first before starting to regulate it. Supplying a powersupply address is meaningless.

6.1.18 SCF

command:	**P1SCF20**
reply:	**p1.*SCF.20**

Sets the frequency with which averages are calculated for the voltage and duty-cycle of active powersupplies and the frequency with which these quantities are checked and/or regulated. The required frequency should be multiplied by a factor of 10. In the example above the control frequency is set to 2 Hz.

Supplying a powersupply address is meaningless; the control frequency applies to all HV-supplies controlled by this HV-controller.

6.1.19 SMC

command:	**P1SMC1200**
reply:	**p1.*SMC.1200**

Sets the maximum allowed current-load of each HV-supply in units of 0.1 µA; in the above example the value is set at 120 µA.

6.1.20 SMT

command: **P1SMT2**
reply: **p1.*SMT.2**

Sets the maximum number of consecutive trips; if a HV-supply trips again within a certain period of time, the supply is disabled and is not switched on again automatically.

Two trips are called 'consecutive' if at the first control-loop after the previous trip the supply trips again. Setting the number to 0 or 1 will cause the supply NOT to be switched on automatically after a trip at all.

6.1.21 SPY

command: **P1SPY1** or **P1SPY0**
reply: **p1.*SPY.1** or **p1.*SPY.0**

Switches the socalled 'spy' mode on or off; if the parameter in the command is 1 the spy-mode is switched on, if it is 0 the spy-mode is switched off. In 'spy'-mode the output produced by the HV-controller will be a bit more informative. This is useful when a user is directly connected to the HV-controller with his terminal. Default spy-mode is off.

6.1.22 SSF

command: **P1SSF100**
reply: **p1.*SSF.100**

Sets the frequency with which the voltage and the duty-cycle of the active supplies are sampled. The required frequency should be multiplied by a factor of 10. In the example above the sample frequency is set to 10 Hz.

Supplying a powersupply address is meaningless; the sample frequency applies to all supplies controlled by this HV-controller, including the auxiliary supply.

6.1.23 SVO

command: **P1.2SVO1120**
reply: **p1.2SVO.1120**

Sets the required voltage for the stated powersupply to the stated value (in Volts). The DACs for this supply are set to the proper values unless the control process is active and the supply is a HV-supply, then the control process will take care of setting the voltage. A *-wildcard for the powersupply address is allowed; then the voltage setting applies to all HV-supplies.

6.1.24 SVS

command:	**P1SVS**
reply:	**p1SVS.100.10.2.800.70.1000.1000.1000.1000.1000.1000.1**

Saves the current general settings in EEPROM for permanent storage, so that from now on they will be the current values after every subsequent (power-up) reset.

The command returns the saved current settings, being the sample frequency (Hz*10), the frequency of regulating (Hz*10), the control delay (s), the maximum current-load (µA*10), the required voltages for all supplies (the auxiliary supply and the HV-supplies) in Volts, and finally the maximum number of trips in a row before a supply remains switched off permanently. Supplying a powersupply address is meaningless.

Note that the current and the frequencies are multiplied by a factor of 10. In the above example the current is 80 µA and the frequencies are 10 Hz and 1 Hz respectively.

6.1.25 SWI

command:	**P1SWI**
reply:	none

If a valid program is present in EEPROM or when the present program is running from EEPROM this command will switch to the start of the program in EEPROM or EPROM respectively. See also section 8.

6.2 Commands for calibration

6.2.1 CAC

command: **P1.2CAC**
reply: **p1.2CAC.75.10.0.0.90.0.0.50.0**

Notifies the HV-controller that a current-load calibration of a particular supply is to take place. The controller will determine the ADC-measurement dependency on the set voltage (will take a few seconds; the reply contains the settings it has used for this procedure) and then wait for one **GCU** current-value user-input after which it can calculate the ADC-to-current-load conversion parameters.

A single powersupply address has to be provided; a wildcard is not allowed.

6.2.2 CAL

command: **P1CAL1** or **P1CAL0**
reply: **p1.*CAL.1** or **p1.*CAL.0**

Switches the calibration mode on or off; if the parameter in the reply is 1 calibration mode is switched on, if it is 0 calibration mode is switched off.

The HV-controller has to be in calibration mode to be able to perform the commands described in the other sections (i.e. **CAC**, **CAD**, **DEP**, **GCU**, **GVO**, **RPA**, **SDC**, **SDF**, **SVP**)

6.2.3 CAV

command: **P1.2CAV**
reply: **p1.2CAV.25.0**

Notifies the HV-controller that a voltage calibration of a particular supply is to take place. The powersupply is enabled and the supply is set to its first calibration setting; the two numbers in the reply state what this setting is for the DACs of this powersupply: in the example above the coarse-setting DAC is set to 25% of its full scale and the fine-setting DAC to 0% of its full scale.

The controller will now expect a total of 4 **GVO** user-inputs with voltages in Volts so that it can calculate the DAC-to-Volts and ADC-to-Volts parameters after the last **GVO** input.

A single powersupply address has to be provided; a wildcard is not allowed.

6.2.4 DEP

command:	**P1.2DEP**
reply:	**p1.2DEP.**

Deletes the calibration parameters stored in EEPROM for the stated supply; after the next reset the default parameters will be used for this supply.

6.2.5 GCU

command:	**P1GCU2240**
reply:	**p1.2GCU.2240.120.300.3500**

Tells the HV-controller the present current-load in units of 0.1 µA. The controller will measure the corresponding ADC-value and calculate the ADC-to-current-load conversion parameters **e, f** and **g**; these parameters are returned in the reply as shown above; they are multiplied by 1000 and displayed as an integer, although inside the program they are reals; so the approximate values are:

e = 0.120
f = 0.300
g = 3.500

There is no need to provide a powersupply address; the address provided previously by the **CAC**-command is used.

6.2.6 GVO

command:	**P1GVO10402**
reply:	**p1.2GVO.10402.75.25**
	or
	p1.2GVO.10402.12543.700.103.1277.-708

Tells the HV-controller the current voltage in tenths(!) of a Volts (in the example above 1040.2 V). The controller will measure the corresp-onding ADC-value and when this was the last step in the calibration it will calculate the ADC-to-Volts conversion parameters **a**, **b** and **a'** and the Volts-to-ADC conversion parameters **c** and **d**; after calculating the conversion parameters they are returned in the reply as shown above in the second reply, displayed as integers although inside the program they are represented by reals; parameters **a** and **a'** are multiplied by 1000 before they are put in the reply. The values in the example above are thus (approximately):

a	= 12.543
b	= 700.0
a'	= 0.103
c	= 1.277
d	= -708.0

If the GVO-command is not the final calibration step, the reply will contain --apart from the voltage entered by the user-- the next settings of the coarse- and fine-setting DAC, in percentages of the full DAC-ranges of the powersupply being calibrated, in the example above 75% and 25% respectively.

There is no need to provide a powersupply address; the address provided previously by the **CAV**-command is used.

6.2.7 RPA

```
command:  P1.2RPA
reply:    p1.2RPA.12543.700.543.1277.-708.120.300.3200
```

Reads the currently used values of the calibration parameters that are used in the linear conversions of DAC and ADC units to Volts and Amperes. A single powersupply address has to be provided; a wildcard is not allowed.

The numbers contained in the reply are parameters **a**, **b**, **a'**, **c**, **d**, **e, f** and **g** respectively, where **a**, **a'**, **c**, **e, f** and **g** are multiplied by a factor of 1000.0. All these parameters inside the controller program are real numbers, but in the reply to **RPA** they are truncated to an integer.
In the example above the parameters are thus (approximately):

a	= 12.543
b	= 700.0
a'	= 0.543
c	= 1.277
d	= -708.0
e	= 0.120
f	= 0.300
g	= 3.200

6.2.8 SDC / SDc

```
command:  P1.2SDC60
reply:    p1.2SDC.60
```

Sets the DAC which does the coarse setting for the stated powersupply to the stated value, which is a percentage of the full DAC range. A *-wildcard for the powersupply address is allowed. (Command **SDc** takes as parameter the DAC-value to be set (0-63)).

6.2.9 SDF / SDf

command: **P1.2SDF60**
reply: **p1.2SDF.60**

Sets the DAC which does the fine setting for the stated powersupply to the stated value, which is a percentage of the full DAC range. A *-wildcard for the powersupply address is allowed. (Command **SDc** takes as parameter the DAC-value to be set (0-63)).

6.2.10 SVP

command: **P1.2SVP**
reply: **p1.2SVP.**

Stores the calibration parameters for the stated powersupply in EEPROM for permanent storage, so that they will be used from now on after every (power-up) reset.

The reply returns the stored values in the same way as described above in the section on command **RPA**.

6.3 Specials

6.3.1 COM

command: **P1COM1** or **P1COM0**
reply: **p1.*COM.1** or **p1.*COM.0**

Enables (1) or disables (0) the use of the RTS and CTS handshake lines during RS232 serial output, which is necessary in case a socalled COM-box is installed to multiplex the serial lines of several HV-controllers onto one RS232 line. Default the software assumes a COM-box is present; if this is not the case (e.g. when testing an individual HV-controller through a terminal connected directly to the HV-controller) the first command after a reset should be the **COM** command, otherwise the software will hang up in the serial output.

7. Error numbers

If the reply of the HV-controller to a command contains the same command mnemonic it means that the command has been processed correctly. If however the reply contains the mnemonic **ERR** something went wrong. The number in the reply following the **ERR**-mnemonic identifies the nature of the error.

Sometimes the **ERR**-reply contains more numbers; the error identifier number is always the last number of the reply.

Below a list of the error identifiers and their symbolic name is shown.

Error	Name
200	EEPROM_TIMEOUT
220	CALIB_VOLT_ZERO
221	CALIB_DAC_ZERO
222	CALIB_ADC_ZERO
223	CALIB_CURRENT_ZERO
230	HEX_CHECKSUM_ERROR
231	HEX_NO_CODE
232	HEX_INCOMPLETE
233	HEX_TOO_BIG
240	I2C_TIMEOUT
241	I2C_STATUS
248	LINE_TOO_LONG
249	LINE_INCOMPLETE
250	ERROR_IN_ADDRESS
251	ERROR_IN_PARAMETER
252	PAR_OUT_OF_RANGE
253	ILLEGAL_CMD
254	UNKNOWN_CMD

A description of each error number follows:

- **EEPROM_TIMEOUT**

A timeout occurred on programming data --like settings, parameters or code bytes of a downloaded program-- in the EEPROM.

- **CALIB_VOLT_ZERO**

Sensible voltage calibration parameters could not be calculated because there is a zero voltage difference between different calibration steps.

- **CALIB_DAC_ZERO**

Sensible voltage calibration parameters could not be calculated because there is a zero DAC-setting difference between different calibration steps.

- **CALIB_ADC_ZERO**

Sensible voltage calibration parameters could not be calculated because there is a zero ADC-reading difference between different calibration steps.

- **CALIB_CURRENT_ZERO**

Sensible calibration parameters could not be calculated because there is a zero current-load difference between different calibration steps.

- **HEX_CHECKSUM_ERROR**

A checksum error was detected in a line of the downloaded Intel-hex format file.

- **HEX_NO_CODE**

The user issued the command to switch to the program in EEPROM, but no valid program is present in EEPROM.

- **HEX_INCOMPLETE**

The downloaded Intel-Hex format file did not have a proper end-of-data line.

- **I2C_TIMEOUT**

A timeout occurred on receiving an acknowledge on the on-board I^{2}C-bus.

- **I2C_STATUS**

An incorrect status was detected during an on-board I^{2}C-bus operation.

- **LINE_TOO_LONG**

The (command)line received by the controller is longer than the controller can store (the line length is limited to 50 characters including <CR>).

- **LINE_INCOMPLETE**

Somehow the <CR> of the (command)line is lost.

- **ERROR_IN_ADDRESS**

The address of the addressed supply/supplies in the commandline is incorrect (for the given command).

- **ERROR_IN_PARAMETER**

A parameter in the command line is incorrect.

- **PAR_OUT_OF_RANGE**

A parameter in the commandline is outside its valid range.

- **ILLEGAL_CMD**

The command given in the commandline is valid but illegal in this mode/version of the program.

- **UNKNOWN_CMD**

The command given does not occur in the list of available commands.

8. Running code from EPROM or EEPROM

If a version of the HV-supply controller software is produced it is important to know whether the code will be stored in EPROM (32 or 64 KByte) or EEPROM (32 KByte), in other words: whether the code is to be the default running code when the controller is switched on or whether the code is downloaded by the user, and then run by issuing the command **SWI.**

Note that downloading needs to be done only once, but to run this code after switching on the controller the **SWI**-command has to be given first to switch from the default program in EPROM to the version that was previously downloaded to EEPROM.

In ***global.h*** when running from EPROM the following definition has to be present (to protect against switching to an 'empty' EEPROM, or downloading code to EEPROM when the running program itself is already stored in EEPROM, overwriting itself):

```
#define RUNNING_EEPROM 0
```

and when running from EEPROM:

```
#define RUNNING_EEPROM 1
```

In order to be able to make a distinction between versions, it is recommended to change the **version** number in ***hv_main.c*** to a unique number. In the example below the version is set to **a.b.**
char version = 0xab;

The current version number can then be obtained using command **RPS.**

Module ***switch.s03*** **s**hould contain the following assembler code to be able to switch from EPROM to EEPROM (Note: in case of a 32 KByte EPROM or EEPROM the code bytes have to be put in the HEX-file by hand at address 07FFEH !!):

```
        PUBLIC switch_program
        ASEG
        ORG 0FFFEH    ; 64K EPROM or 32K mirrored in 08000H-0FFFFH
PIO DEFINE 090H    ; 80C552  port P1
switch_program:
        CPL  PIO.3
        END
```

and the following code to be able to switch from EEPROM to EPROM:

```
        PUBLIC switch_program
        ASEG
        ORG 0FFFEH    ; 32K EEPROM mirrored in 08000H-0FFFFH
PIO DEFINE 090H    ; 80C552  port P1
switch_program:
        CPL  PIO.3
        END
```

If the above directions are followed the user can (safely) switch from the program (version) in EPROM to the one in EEPROM and back.

9. Default Settings

When no settings are ever saved the default settings are:

sample-frequency:	10.0	Hz	
control-frequency:	1.0	Hz	
control delay:	3	s	
requested high-voltage:	1000	V	
auxiliary supply voltage:	75	V	
maximum allowed current-load	100.0	µA	

and:

maximum number of trips:	1		

and -'hardwired' in the code-:

minimum high-voltage to set:	800	V	(BPC: 600 V)
maximum high-voltage to set:	1200	V	(BPC: 1000 V)
minimum sample-frequency:	1.0	Hz	
maximum sample-frequency:	20.0	Hz	
minimum control-frequency:	0.1	Hz	
maximum control-frequency:	10.0	Hz	
range of no adjustment to voltage:	measured voltage = requested voltage ± 0.3 V		
range of allowed measured voltage:	measured voltage = required ADC-count ± 50 (ca. 20V)		
range of allowed set voltage:	set voltage = requested voltage ± 50 V		