Projekt: GPIB		Titel: GPIB Command Set	
Ablage: Commands.doc	Dokument Nummer: #7	Revision: 4	Anzahl Seiten: 1
Autor: Dr. Hans-Joachim Fischer	Speicherdatum: 14.11.2003 18:03	Prüfer: Dr. Hans-Joachim Fischer	Prüfdatum: 14.11.2003
Grund der Überarbeitung: Photos inserted			
Bezogene Dokumente:			

SMDI
GPIB Command Set

This document describes the commands to control the GPIB-A1 via the GPIB bus. All commands and all values related to commands are transferred as ASCII text strings.

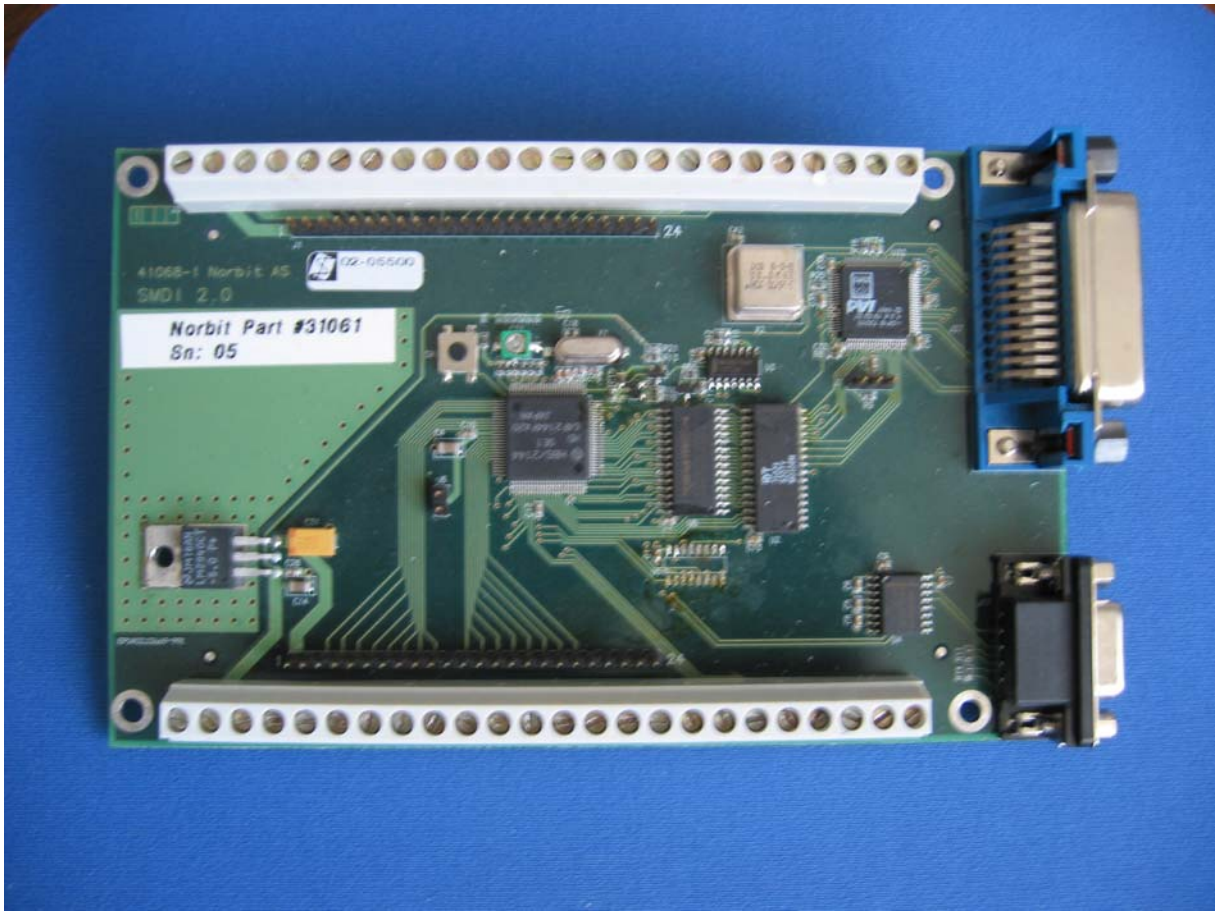
GPIB Interface

All commands from the GPIB-CIC[1] to the GPIB-A1 and all results from the GPIB-A1 to the GPIB-CIC are transferred via the GPIB interface in ASCII 7-bit format, i.e. the MSB always is set to '0'.

Reading from the GPIB-A1 requires first to issue a command to the GPIB-A1, otherwise the GPIB-A1 will not send anything. However this does not disturb the proper function of the GPIB-A1.

Every command string received from the GPIB and every answer string send to the GPIB has to be terminated with the character '\n' = 0xA (new line). This character is used in hpVEE to identify the end of a single read or write command.

The following photo shows the GPIB-A1 without extension module from top:



[1] CIC: "Controller in charge" is the active GPIB controller, e.g. hp VEE.

General Commands

Command	Description
*IDN?	**Queries for the HW identification.** Upon this command, the SMDI will write the HW identification to the GPIB output queue, e.g. "ESF GmbH,GPIB-A1,1,0"
*TST?	**Performs self test** actually does nothing but returns "0" to the GPIB output queue as required by standard
*RST	**Performs SW restart on the GPIB-A1**
:ERR?	**Queries the error message queue** Reports the latest error message and removes it from the queue. If the queue is empty, a special error message is returned.
:VERS?	**Queries for the SW version.** Upon this command, the SMDI will write the SW version to the GPIB output queue, e.g. "ESF GmbH SMDI SW Version 2.0"
:MDL?	**Queries the module type** Upon this command, the SMDI will write the actual setting of module type as integer value in ASCII format to the GPIB output queue.
:MDL *value*	**Sets the module type to the specified value** Valid values are 0 no module present 1 stepper motor drive present On successful recognition of command, error code 0 will be set.
:GPIB *value*	**Sets the GPIB device address of the module to the specified value.** The change of the address will be executed immediately. On successful recognition of command, error code 0 will be set.

Digital I/O commands

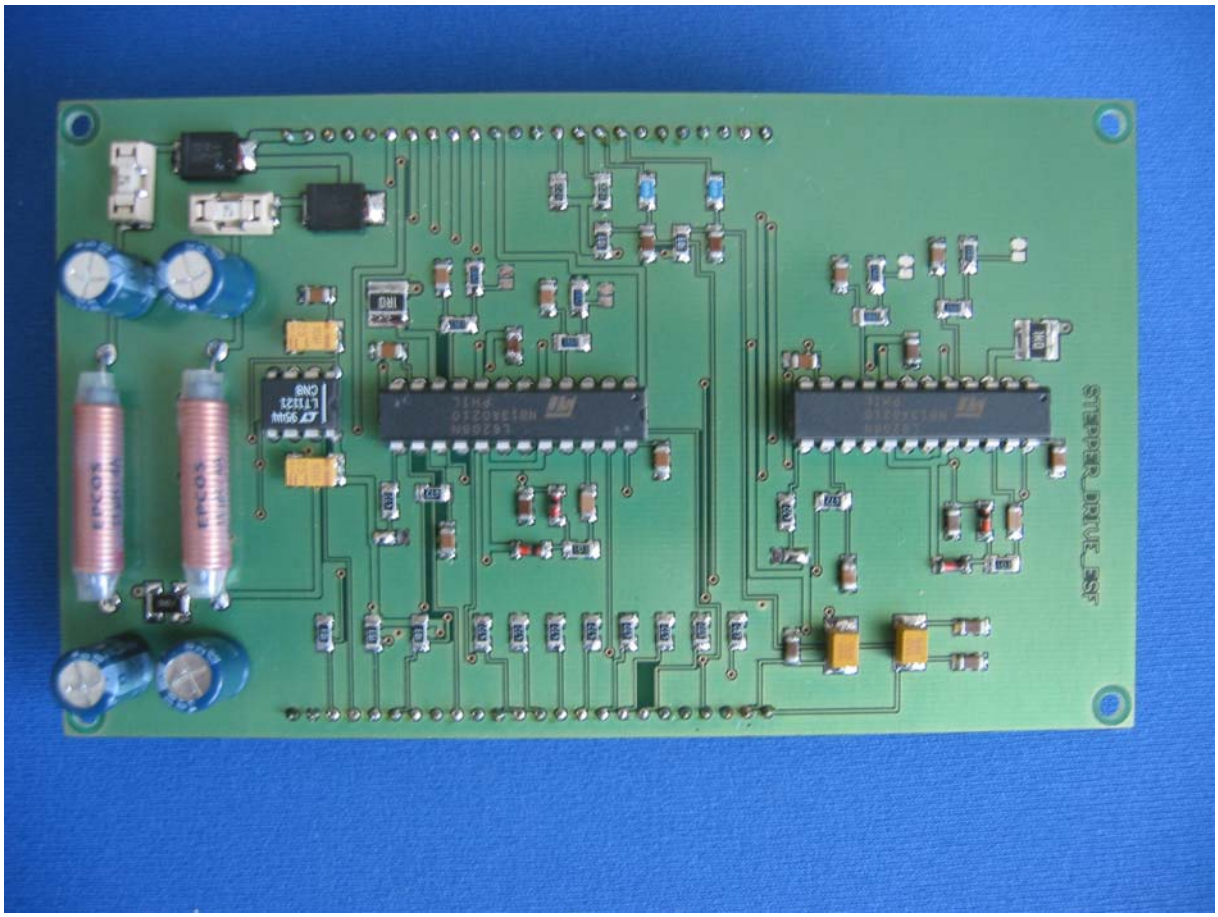
Command	Description
:DIN *bit*	**Read one or all bits from the digital input DIN (Port 2).** *bit*: Indicates the bit position in hexadecimal format. Allowed values are ['0x01', '0x02', '0x04', '0x08', '0x10', '0x20', '0x40', '0x80', '0xFF']. The value '0xFF' indicates to read all eight bits. On error, a value of 256 will be written to the GPIB output queue. This command is maintained only for compatibility purposes. New implementations may use the new command :D2I after proper setting of the port direction. :DIN is a valid command only in case the direction of port 2 is set properly.
:D1I *bit*	**Read one or all bits from the digital input D1I (Port 1).** *bit*: Indicates the bit position in hexadecimal format. Allowed values are ['0x01', '0x02', '0x04', '0x08', '0x10', '0x20', '0x40', '0x80', '0xFF']. The value '0xFF' indicates to read all eight bits. On error, a value of 256 will be written to the GPIB output queue. :D1I is a valid command only in case the direction of port 1 is set properly.
:D2I *bit*	**Read one or all bits from the digital input D2I (Port 2).** *bit*: Indicates the bit position in hexadecimal format. Allowed values are ['0x01', '0x02', '0x04', '0x08', '0x10', '0x20', '0x40', '0x80', '0xFF']. The value '0xFF' indicates to read all eight bits. On error, a value of 256 will be written to the GPIB output queue. :D2I is a valid command only in case the direction of port 2 is set properly.
:DOUT *bit, value*	**Set one or all bits to the digital output DOUT (Port 1).** *bit*: Indicates the bit position in hexadecimal format. Allowed values are ['0x01', '0x02', '0x04', '0x08', '0x10', '0x20', '0x40', '0x80', '0xFF']. The value '0xFF' indicates to set the whole byte. *value*: '0x0', '0x1' for single bits, and ['0x0' … '0xFF'] for whole byte. On successful recognition of command, error code 0 will be set. This command is maintained only for compatibility purposes. New implementations may use the new command :D1O after proper setting of the port direction. :DOUT is a valid command only in case the direction of port 1 is set properly.
:D1O *bit, value*	**Set one or all bits to the digital output D1O (Port 1).** *bit*: Indicates the bit position in hexadecimal format. Allowed values are ['0x01', '0x02', '0x04', '0x08', '0x10', '0x20', '0x40', '0x80', '0xFF']. The value '0xFF' indicates to set the whole byte. *value*: '0x0', '0x1' for single bits, and ['0x0' … '0xFF'] for whole byte. On successful recognition of command, error code 0 will be set. :D1O is a valid command only in case the direction of port 1 is set properly.

:D2O *bit, value*	**Set one or all bits to the digital output D2O (Port 2).** *bit*: Indicates the bit position in hexadecimal format. Allowed values are ['0x01', '0x02', '0x04', '0x08', '0x10', '0x20', '0x40', '0x80', '0xFF']. The value '0xFF' indicates to set the whole byte. *value*: '0x0', '0x1' for single bits, and ['0x0' … '0xFF'] for whole byte. On successful recognition of command, error code 0 will be set. :D2O is a valid command only in case the direction of port 2 is set properly.
:D1D? :D2D?	**Queries for the actual direction of port 1 (P40 .. P47) or 2 (PB0 .. PB7)** Return value will be either "in": reading from the port or "out": writing to the port.
:D1D *direction* :D2D *direction*	**Sets communication direction of port 1 (P40 .. P47) or port 2 (PB0 .. PB7) on the module.** *direction*: "in" (or any other string starting with 'in') indicates reading from the port to the GPIB. "out" or any other string starting with 'out') indicates writing from the GPIB to the port. On successful recognition of command, error code 0 will be set.
:D1S? :D2S?	**Queries for the default setting of port 1 in output mode** Return value will be "0x<pattern>, e.g. "0x3A".
:D1S *value* :D2S *value*	**Defines the default setting for port 1 or port 2, used after power up / reset if port is switch to output** On successful recognition of command, error code 0 will be set.

Stepper Commands

The stepper commands apply only in case the stepper extension module is inserted.

The stepper extension module is shown in the following photo:



Command	Description
:SMOT *motor*	**Pre-selects one out of two motors.** *motor*: Motor to be controlled afterwards. Allowed values are '0' and '1'. On successful recognition of command, error code 0 will be set.
:STIC *tics*	**Pre-selects the number of tics.** This number is used to set the speed. *tics*: Number of tics. Allowed values are '1' … '15'. The time for MAXPOSITION steps is *tics* * 4s. On successful recognition of command, error code 0 will be set.

:SSP *switch*	**Set the pre-selected motor whether to use service request with serial polling** when using the :NULL command. *switch*: 0 = false. Do not use service request 1 = true. Use service request The serial poll response will be: 2: Success for motor 0 3: Success for motor 1 4: Failure for motor 0 5: Failure for motor 1. Note: The response as transmitted via the GPIB is 64 plus the value specified before. The value of 64 indicates that the device requested service from the GPIB CIC. On successful recognition of command, error code 0 will be set.
:SP?	**Queries the pre-selected motor whether service request is set.** The response is 0 or 1; see :SSP
:NULL *search*	**Requests the pre-selected motor with low speed (*tics* = 15) to search the zero-position.** *search*: Single sided search range in multiple steps around actual motor position. First a search clockwise is done. If zero-position is not found, then a search counter clockwise with twice the steps is done. Motor stops automatically at the zero-position. Optionally the GPIB-A1 can request a SRQ from the GPIB controller once the zero-position is detected in order to announce the success of the operation; see command :SSP. On successful recognition of command, error code 0 will be set. Note: As long as GPIB-A1 searches for the zero position, it will not be able to act as a GPIB Talker. Will be changed in a later version of the firmware.
:NULL?	**Queries the pre-selected motor whether it knows its true position,** i.e. whether it previously has detected actively the zero-position. The response is either "Yes: Motor x knows position" or "No: Motor x position unknown". This command is maintained only for compatibility purposes. It may be replaced by the new and more flexible command :POS?.
:POS *position*	**Requests the pre-selected motor with the pre-selected speed to reach the indicated position.** *position*: New position in multiples of steps. Allowed values are '0' … MAXPOSITION-1[2]. This command will be performed only, if the GPIB-A1 knows the actual position of the motor. On successful recognition of command and successful validation, error code 0 will be set.

[2] MAXPOSITION is equal 599. In a later version of the firmware, any value for MAXPOSITION may be set.

:POS?	**Queries the actual stop position of the pre-selected motor.** The response is *flag, position* where *flag* is set to 1 in case the position is known 0 in case the position is unknown and *position* is set to a value in the valid range of 0 … MAXPOSITION-1, or 0 in case, the position is not known.
:STEP *steps*	**Requests the pre-selected motor with the pre-selected speed to do the indicated number of steps.** *steps*: Number of steps. Positive value is clockwise rotation. Negative value is counter clockwise rotation. The value of zero is not allowed. Allowed values are such, that the new target position is within the range of '0' to MAXPOSITION-1, including the boarders. This command will be performed only, if the GPIB-A1 knows the actual position of the motor. On successful recognition of command and successful validation, error code 0 will be set.
:STOP?	**Queries the pre-selected motor whether it rotates or stops actually.** The response is either "Yes: Motor x stops" or "No: Motor x rotates".
:ABORT	**Stops the pre-selected motor immediately.** On successful recognition of command, error code 0 will be set. The stop position can be retrieved using the command :POS?
:HOLD?	**Queries the actual setting for the HOLD parameter of the pre-selected motor.** The response is 0 or 1.
:HOLD *value*	**Sets the HOLD parameter for the pre-selected motor.** *value*: 0 – motor current will be switched off after the target position is reached 1 – motor current keeps always on On successful recognition of command and successful validation, error code 0 will be set.

Note: All stepper commands will be available only if the module type (see ":MDL") is set to 1.

Auxiliary Commands

:rs232debug	**Toggles the state of the debug interface at UART 1** The response is either "debug" if the debugging function is switched on or "normal" otherwise. Default is "debug".

Parameters stored in non-volatile memory

The following parameters are stored in the non-volatile memory of the module.

Parameter	Default value	Description
Version id	1	If the structure of the parameter record will be changed a new version-id will be used. If the version doesn't match, default values will be used.
GPIB address	25	The device address of the module. Can be changed using command ":GPIB"
MAXPOSITION	600	This parameter limits the *steps / position*. The motor can be moved between 0 and MAXPOSITION-1. This parameter can be changed only via the debugging interface.
TCORA	29	time base for motor clock – no user access implemented.
ClockFactor	4	If the motor drive works in half-step mode it needs 4 clocks for one motor step, but if the motor drive works in full-step mode it needs only 2 clocks for one motor step. Using this parameter an adjustment can be done. Accessible only via the debugging interface.
DebugFlag	0 = debug	A flag which can be toggled by the ":rs232debug" command. If this flag is not zero, some debug information will be transfer over the UART1 and some auxiliary command may be invoked over UART1.
ModuleType	0 = no module	Specifies the module type. The commands ":MDL?" and ":MDL" allow access to this parameter. For details see command description.
Port1Dir	1	Direction of port 1 (P40 .. P47)
Port2Dir	0	Direction of port 2 (PB0 .. PB7)
Port1Default	0	default output pattern port 1, used after reset when Port1Dir is set to output
Port2Default	0	default output pattern port 2, used after reset when Port2Dir is set to output
rfu	0	reserved area for future use
Checksum	calculated	Used for consistence checking of the parameter record. If the checksum is corrupted, default values will be used.

Error Codes

This table shows error codes for all options of the SMDI and the attachable modules. Some of the commands and the related error messages may not be implemented in a specific device. The maximum number of characters in an error message is 31.

Source	Number	Message
Several	00	"00: No error"
Several	01	"01: Command?"
Command SHS	02	"02 SHS: Port"
Command DOUT	03	"03 DOUT: Parameters"
Command DOUT	04	"04 DOUT: Address hex code"
Command DOUT	05	"05 DOUT: Value hex code"
Command DOUT	06	"06 DOUT: Address"
Command DOUT	07	"07 DOUT: Bit value"
Command DOUT	08	"08 DOUT: Byte value"
Command SMIO	09	"09 SMIO: Direction"
Command MIOO	10	"10 MIOO: Direction"
Command MIOO	11	"11 MIOO: Parameters"
Command MIOO	12	"12 MIOO: Address hex code"
Command MIOO	13	"13 MIOO: Value hex code"
Command MIOO	14	"14 MIOO: Address"
Command MIOO	15	"15 MIOO: Bit value"
Command MIOO	16	"16 MIOO: Byte value"
Command SMOT	17	"17 SMOT: Value"
Command STIC	18	"18 STIC: Value"
Command SSP	19	"19 SSP: Value"
Command NULL	20	"20 NULL: Value"
Command POS	21	"21 POS: Value"
Command STEP	22	"22 STEP: Value"
Command MOUT	23	"23 MOUT: Parameters"
Command MOUT	24	"24 MOUT: Address hex code"
Command MOUT	25	"25 MOUT: Value hex code"
Command MOUT	26	"26 MOUT: Address"
Command MOUT	27	"27 MOUT: Bit value"
Command MOUT	28	"28 MOUT: Byte value"
Commands STEP and POS	29	"29 Motor: No rotation"
Command HOLD	30	30 HOLD: Value
Command GPIB	31	31 GPIB: Value

Command MDL	32	32 MDL: Value
Command D1O	51	51 D1O: parameters
Command D1O	52	52 D1O: address hex code
Command D1O	53	53 D1O: value
Command D1O	54	54 D1O: address
Command D1O	55	55 D1O: bit value
Command D1O	56	56 D1O: byte value
Command D1D	57	57 D1D: parameters
Command D1S	58	58 D1S: value
Command D2O	61	61 D2O: parameters
Command D2O	62	62 D2O: address hex code
Command D2O	63	63 D2O: value
Command D2O	64	64 D2O: address
Command D2O	65	65 D2O: bit value
Command D2O	66	66 D2O: byte value
Command D2D	67	67 D2D: parameters
Command D2S	68	68 D2S: value
Command ERR	99	"99 ERR: No error message available"

Connectors

J3

J3 is specific to the extension module used.

Pin	Name	Dir.	Description
1-5	GND	-	Ground
6-21	IOports	-	General purpose I/O lines. Dependent on module.
22	GND	-	Ground
23	M-	Power In	External power supply for module M- may be connected to GND, dependent on module.
24	M+	Power In	

The following drawing shows J3 as used for the stepper module.

GND	n.c.	C1	C0	LED	GND	M1: D - A	M0: A - D	GND	M-	M+

J5

J5 is the standard connector unique for all extension modules.

Pin	Name	Dir.	Description
1	V+	Power In	Main power input Allowed range: 7,5 V … 12 V.
2	V-	Power In	
3	GND	-	Ground
4-11	DOUT (Port 1)	In/**Out**	Digital input/output port. TTL voltage. Default is output.
12-19	DIN (Port 2)	**In**/Out	Digital input/output port. TTL voltage. Default is input.
20	GND	-	Ground
21	HSi	In	Handshake interrupt input. TTL voltage.
22	HSo	Out	Handshake output. TTL voltage.
23-24	GND	-	Ground

The following drawing shows J5.

V+	V-	GND	DOUT: 0 - 7	DIN: 0 - 7	GND	HSi	HSo	GND