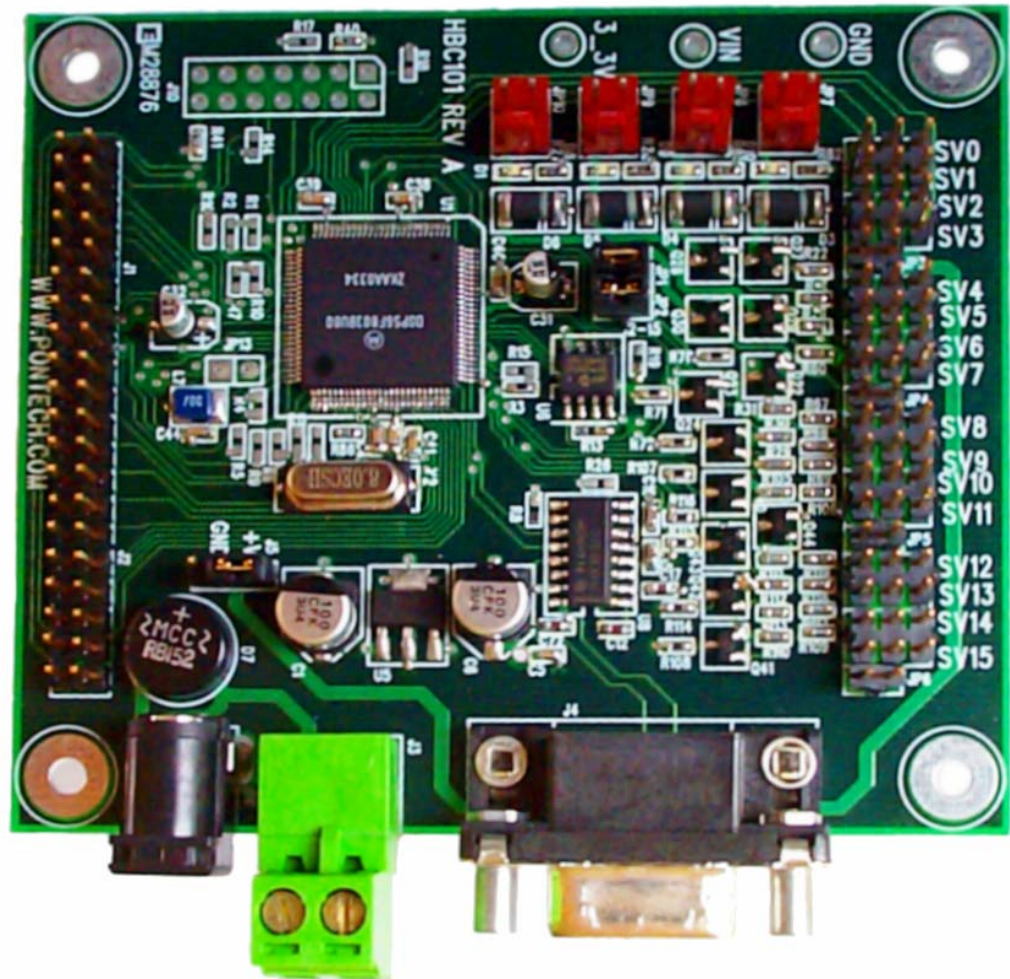


PONTTECH

HBC101

Hybrid Motor Controller



C3-V0.5 Firmware Manual

November 20, 2004

www.pontech.com

Preface

This documentation reflects the features and usage of the C3-V0.5 firmware release for the HBC101 Rev B. Be sure to check www.pontech.com for manual updates.

Revision History

- Jul 13 2004 C3-V0.5 Neural Robotics
 Added control of GPIO E2 & GPIO E3 Pins using PS, PC, and PT.
 Added input ability GPIO E2 & GPIO E3 Pins using RP.
- Jun 8 2004 Cx-V0.4 Neural Robotics
 Added ability to switch signal routing by a signal.
 Added sketchy IRQ A/B inputs.
 Changed SSR command to SSD, for signal routing default.
 Added SSR command to save map to EEPROM RAM Structure.
 Added SRS command to restore map from EEPROM RAM Structure.

Table Of Contents

Preface	2
Revision History	2
Table Of Contents	3
Board Operation	5
Notation Conventions	5
Notes for SV203 and STP100 users	6
Commands Descriptions	6
Board Control	7
BDn Select Board	7
BRn Set serial bit rate.	7
WRn m Write RAM	7
RRn Read RAM	7
WSS Write System Settings to EEPROM	7
RSS Read System Settings from EEPROM.....	7
V? Firmware Version	8
? Help Query.....	8
RC Servo Control	8
MSn Select Operating Mode	8
MS? Read Current Operating Mode.....	8
SVn Select Servo	8
Mn Move to absolute position.....	9
M?n Read absolute position.....	9
In Incremental move relative to current position	9
SSn m Set Signal Routing	10
Default direct routing.....	10
Routing an output signal from an output command.....	10
Routing an output signal from an input signal.....	11
Routing an input command from an input signal.....	11
Routing an input command from an output command	11
SSD Default the Current Signal Routing Table	12
SSR Copy Signal routing table	12
SRS Copy RAM to Servo routing table.....	12
CSR Copy Servo Output Positions to RAM	12
CRS Copy RAM to Servo Output Positions	12
SPE Set PPM Value.....	12
SPE? Read PPM Value.....	13
Digital I/O.....	13
PSn Pin Set	13
PCn Pin Clear	13
PTn Pin Toggle.....	13
RP?n Read Pin.....	13
Analog to Digital Conversion.....	13

AD?n Read a voltage on the A/D port	13
Notes	15
Warranty and Copyrights.....	16
Warranty	16
Copyright and Trademarks	16
Disclaimer of Liability	16

Board Operation

Connect the servos to the board's RC Servo Outputs and any RC Receiver Outputs to the RC Servo inputs. Connect a straight through DB9 cable from the RS-232 Port of the HBC101 and the other end to the COMM port of a PC. Power the board and the servos should return to the neutral position.

The HBC101 processes information one ASCII string command at a time. Each command string follows the format:

L n L n ... <enter> (maximum of 20 characters/line)

Where *L* is the command letter(s), *n* is a decimal integer number(s), and <enter> is ASCII 13. Please refer to Command Description of the manual for a complete listing.

For example, the commands to select a board, select a servo and move to a position are BD, SV, and M, respectively. If your want to move servo 3 of a board with an ID number 1 to position 85, you would send the flowing command string.

BD1SV3M18500 <enter>

Spaces or commas for ease of reading can also separate the commands.

BD1 SV3 M18500 <enter> -or-
BD1,SV3,M18500 <enter> -or-
BD 1 SV 3 M 18500 <enter>

A terminal program may be used to test the functions of the board. The default setting on the board is 115,200 bps, No Parity, 8-Bits, 1 Stop Bit.

Once a board or servo is selected, it will stay selected until power is removed or another select command is received. For example, if the following commands were sent:

BD1 SV2 M10000 <enter>

The next command will still move servo 2:

M15000 <enter>

Any parameter value for the command not in the range of the command will be ignored.

The board will start processing the command string when it receives the <enter> or ASCII 13 character.

Notation Conventions

Hexadecimal numbers are noted in using the Motorola assembly language convention of pre-pending the numerical value with the '\$' character. For example the decimal value 65535 would be noted as \$FFFF.

Notes for SV203 and STP100 users

The HBC101 represents a jump in parsing technology that is available on the SV203 or STP100 products. The two biggest differences that users of these products will notice are that the commands are now case independent and that numerical values can be entered in hexadecimal by proceeding the value with a '\$' symbol.

Commands Descriptions

The HBC101 command set is divided up into functional sub systems. The list of sub systems and a brief summary is given below:

Functional:

- BD - Board Control
- SV - RC Servo Motor Control
- DIO - Digital Input and Output
- ADC - Analog to Digital Conversion
- STP - Stepper Motor Control
- HB - H-Bridge/DC Brush Motor Control
- PWM - Pulse Width Modulation Control

Board Control

BDn ***Select Board***

Before the board will accept any commands, it must first be enabled. To enable the board, you must send the **BD** command followed by the board ID number <n>. The default ID number of the board is 1. So simply send the following to enable the board:

```
BD1 <enter>
```

The board ID number can be user-redefined by using the **WE** command (see Commands Descriptions Page – p.20). This allows multiple boards of different ID number to be connected to the same serial port.

You can enable the board in two other ways: You can pre-enable the board at power-up by changing the default settings. (See **WSS** command); or you can enable the board by sending an ID number 0, such as:

```
BD0 <enter>
```

This will override the ID number checking and any boards connected to the network will be enabled regardless of the ID number of the board. No board will respond with a prompt to a Carriage Return <ASCII 13> if a **BD0** command has been sent. This prevents collision on the RS-485 line.

BRn ***Set serial bit rate.***

This command allows you to select the baud rate of your choice. Issuing the **WSS** command afterwards is necessary for the baud rate to be restored the next time the board is powered. If you change the baud rate and have trouble re-establishing communication before issuing **WSS**, simply recycle the power on the HBC101. If you are still unable to communicate with the HBC101 please contact tech.support@pontech.com.

NOTE: The **BR** command will clear any commands that follow the same line in the serial buffer. This is done so that a **WSS** command must be issued at the newly requested baud rate as a safety measure.

WRn m ***Write RAM***

Write 16-bit value *m* into the “System Settings” memory structure at location *n*.

RRn ***Read RAM***

Read the 16-bit value stored in the “System Settings” memory structure stored at location *n*.

WSS ***Write System Settings to EEPROM***

Write the “System Settings” memory structure to EEPROM

RSS ***Read System Settings from EEPROM***

Restore the “System Settings” memory structure from EEPROM

V? *Firmware Version*

Query firmware version.

? *Help Query*

Visit Pontech.com for assistance.

RC Servo Control

MSn *Select Operating Mode*

The operating mode affects the way the board addresses servos as well as the servos resolution. Your HBC101 has two modes, 0 and 1. By default your HBC101 should be in mode 1, otherwise known as 100ns mode. Mode 0 is also known as SV203 mode, where in this mode the board can be used in the same fashion as a SV203. Although this mode was added for some backwards compatibility, we at Pontech feel it is best to avoid using this mode unless absolutely necessary due to the loss of servo resolution.

MS? *Read Current Operating Mode*

This command is used to read the current operating mode.

SVn *Select Servo*

On power-up, pin SV0 is pre-selected. To select another servo or to make sure the servo is selected, send **SV** followed by the servo number. The servo number must be between 0-15 if in Mode 1(100ns) or 1-16 if in Mode 0(SV203). Below is a table illustrates respective servo numbers while in different modes.

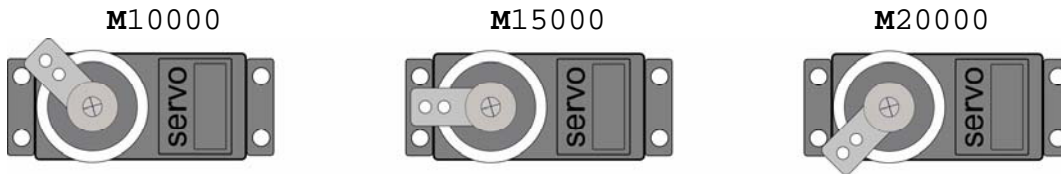
Pin name as shown on Board	Servo # in Mode 1 (100ns)	Servo # in Mode 0 (SV203)
SV0	SV0	SV1
SV1	SV1	SV2
SV2	SV2	SV3
SV14	SV14	SV15
SV15	SV15	SV16

SV1 <enter>

Servo 1 will be selected. Any move commands that follow will operate on servo 1.

Mn Move to absolute position

This command will move the selected servo to an absolute position. The range of the position is between 500 and 65,535 in Mode 1(100ns) or 1 and 255 in Mode 0 (SV203). On a RC-type servo, the maximum mechanical movement is about 180 degrees. The 10000 to 20000 position ranges gives a precision of a about 0.009 degrees. This precision is much greater than most servos can respond to. The figure below shows servo position at respective value in 100ns Mode.

***M?n Read absolute position***

This command will return the last position received by the board. If *n* is an output servo (0 – 7) this value will be the value set by the last **M** command issued. If *n* is an input servo (8 – 15) this value will be the timing value of the last PPM pulse received.

In Incremental move relative to current position

This command will move the servo relative to its current position by adding or subtracting the value entered to the current position.

```
M10000 <enter>
I1000 <enter>
I-2000 <enter>
```

The selected servo will first move to position 10000, then to position 11000 (10000+1000), and then finally to position 9000 ((10000+1000) - 2000).

SSn m Set Signal Routing

Set signal output *n* routing from signal input *m*.

The **SS** command is a matrix that allows selection of output control from and input.

Currently, only servos signal can be routed to other servo signals.

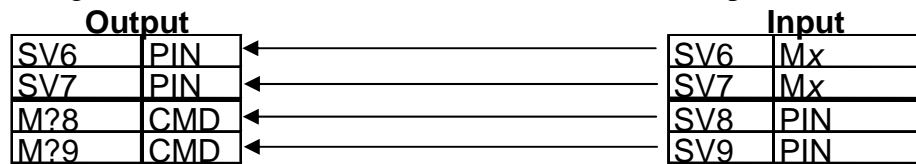
SS n Position	Related Board Item	Note
0..7	SV0..SV7	Set output control for SV <i>n</i> from <i>m</i>
8..15	SV8..SV15	Set query control for SV <i>n</i> input <i>m</i>

Default direct routing

By default each *n* element of the **SS** array is set to its equivalent board pin *m*.

Example:

The default setting of **SS 7 7** ties the commands **SV7 Mx** to the SV7 pin on the board.



Routing an output signal from an output command

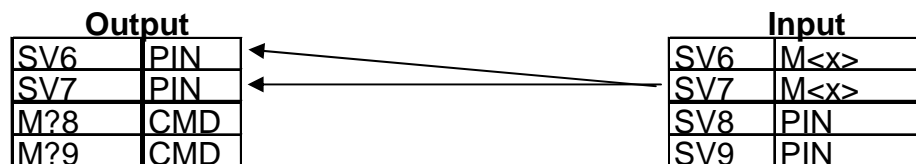
The signal on the pin **SVn** is command by the command **SVm**.

Example:

SS6 7 <enter>

This command will set the signal routing such that pin SV6 on the board will be controlled by commands issued to **SV7 Mx**.

If **SS 7** has the default setting of 7 (**SS 7 7**) then the above **SV** command will move servos connected to pin SV6 and SV7 at the same time to the same position.



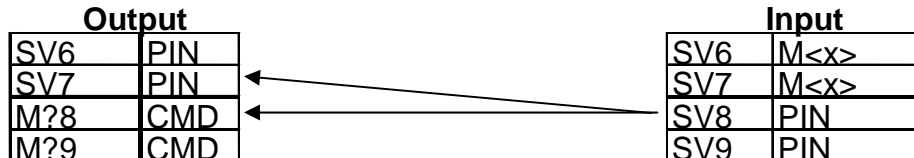
Routing an output signal from an input signal

The signal on the pin **SV n** is commanded by the signal on the pin **SV m** .

Example:

`SS7 8 <enter>`

This command will set the signal routing such that pin **SV7** on the board will be controlled by commands by the signal on the pin **SV8** on the board.



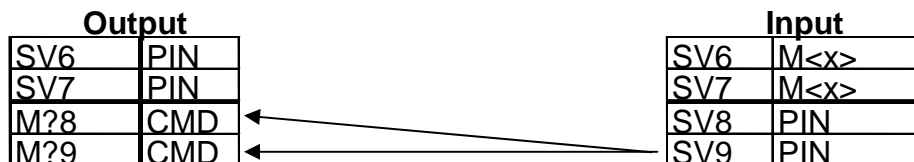
Routing an input command from an input signal

The command **M? n** returns the timing information detected on pin **SV m** .

Example:

`SS8 9 <enter>`

This command will set the signal routing such that the command **M?8** will be return the timing pulse detected on the pin **SV9** on the board.



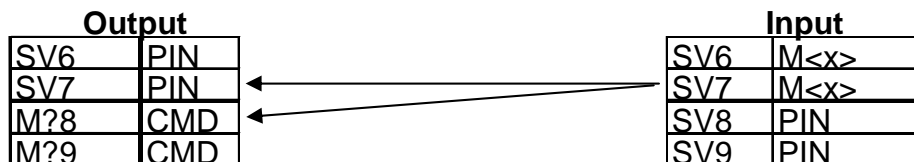
Routing an input command from an output command

The command **M? n** returns the timing information detected on pin **SV m** .

Example:

`SS8 7 <enter>`

This command will set the signal routing such that the command **M?11** will be return the timing pulse set by the command **SV7**.



SSD *Default the Current Signal Routing Table*

Defaults the Signal Map, basically doing to work of:

```
SS0 0
SS1 1
...
SS14 14
SS15 15
```

SSR *Copy Signal routing table to RAM*

Saves the current Signal Map to the RAM structure.

SRS *Copy RAM to Servo routing table*

Restores the saved Signal Map from the RAM structure as the current Signal Map.

CSR *Copy Servo Output Positions to RAM*

Copy the current servo position data to the RAM structure positions that hold the Servo Startup Position.

CRS *Copy RAM to Servo Output Positions*

Copy the RAM structure positions that hold the Servo Startup Position to the current servo position.

SPE *Set PPM Value*

The **SPE** command allows the servo PPM functionality of the servo input and output pins to be enabled or disabled. This command accepts a sixteen bit unsigned value, where each bit corresponds to a single servo pin on the board shown in the table below. By default this value is set to 65535 (\$FFFF) there by enabling all PPM servo outputs and inputs on the board.

Pin name as shown on Board	SPE value (decimal)	SPE value (hex)
SV0	1	0001
SV1	2	0002
SV2	4	0004
SV3	8	0008
SV4	16	0010
SV5	32	0020
SV6	64	0040
SV7	128	0080
SV8	256	0100
SV9	512	0200
SV10	1024	0400
SV11	2048	0800
SV12	4096	1000
SV13	8192	2000
SV14	16384	4000
SV15	32768	8000

NOTE: The SV outputs (SV0-7) are affected by the last two digits of the SPE value in hex (\$00XX). While the SV inputs (SV8-15) are affected by the first two digits (\$XX00).

SPE? Read PPM Value

The **SPE?** command queries the value set by the last **SPE** command.

Digital I/O

PSn Pin Set

PCn Pin Clear

PTn Pin Toggle

These commands allow you to use the servo port as a digital output by setting, clearing or toggling individual bits of the servo port. In order to use the port as a digital output, the servo PWM must first be turned off by sending a **SPE** command with each servo pin bit, set to zero, that you want to use as digital output.

```
SPE$FF7F <enter>
```

```
PS7 <enter>
```

Pin SV7 of the servo port will be set high (5 Volts).

```
SPE$FFBF <enter>
```

```
PT6 <enter>
```

Pin SV6 of the servo port will be toggled/flipped
(Set high if pin was low, or cleared if pin was high).

RP?n Read Pin

This command allows reading of unused servo input pins 8-15. The returned value is either 0 or 1.

Analog to Digital Conversion**AD?n Read a voltage on the A/D port**

The HBC101 has an 8-Channel, 12-bit ADC (analog to digital converter). Each channel *n* can be specified by sending a value between 0 and 7 to the board. When the board receives this command, it will read the specified voltage on the pin and return a value between 0 to 4095, inclusive which represents a voltage between 0 to 3.3 Volts.

```
AD?1 <enter>
```

To calculate the voltage from the returned value use the following formula:

Voltage = value / 4096 * 3.3 Volts

If wires were connected as the figure below and the pot was in the middle position, the board will return a value close to 128 followed by <ASCII 13> and <ASCII 10>, which is about 2.5 Volts.

Notes

Warranty and Copyrights

Warranty

ProLinear/PONTECH, Inc. warrants its products against defects in materials and workmanship for a period of 90 days.

If you discover a defect, ProLinear/PONTECH, Inc. will, at its option, repair, replace, or refund the purchase price. Simply return the product with a description of the problem and a copy of your invoice (if you do not have your invoice, please include your name and telephone number).

The warranty shall become void if the product has been damaged by accident, abuse, or misuse.

Copyright and Trademarks

Copyright © 2004 by ProLinear/PONTECH, Inc. All rights reserved.

HBC101 is a trademark of ProLinear/PONTECH, Inc.

All other trademarks and registered trademarks belongs to their respective owner.

Disclaimer of Liability

ProLinear/PONTECH, Inc. is not responsible for special, incidental, or consequential damages resulting from any breach of warranty, or under any legal theory, including lost profits, downtime, goodwill, damage to or replacement of property, and any costs or recovering, reprogramming, or reproducing and data stored in or used with ProLinear/PONTECH, Inc.