Higbie Labs

Bus communication detail

RS485 bus serial settings are 500k baud, n,8,1.

Serial bus messages are human-readable, in the format:

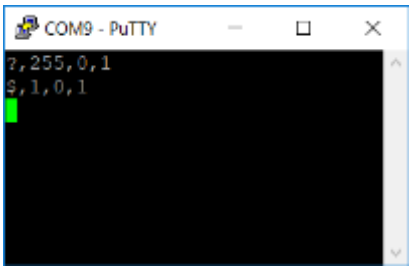
@	,	1	,	19	,	1	CR-LF	[CRC]
@, ?, $, !, S or # @ = Command ? = Query $ = Response ! = Scripty xfer S = Scripty ack # = Heartbeat	Comma delimiter	Bus address Unsigned integer	Comma delimiter	Register number Unsigned integer	Comma delimiter	Value 16- or 32-bit signed integer	Terminator Carriage return + line feed	Optional 16-bit CRC

The first character indicates what type of device issued the message, and also what type of message it is. A command (@) or query (?) or heartbeat (#) would be issued by a host, such as the desktop or tablet app, or the Hig-Hub joystick bus hub. A response to a query ($) would be issued by an actuator or a Hig-Hub bus hub upstream to a user-app. In the case of a command or query, the bus address is the recipient of the message. In the case of a response to a query, or a heartbeat, the bus address is the sender. The register number is the address within the actuator or hub for the function or parameter being accessed. Finally, the value is the number to be poked into the register with a command (@). In a query, the value is ignored. Full register maps for mover and hub at the end of this document. In the above example, register 19 is the motor on/off function. The value 1 is as Boolean TRUE, so this message is telling actuator 1 to turn on its motor coils.

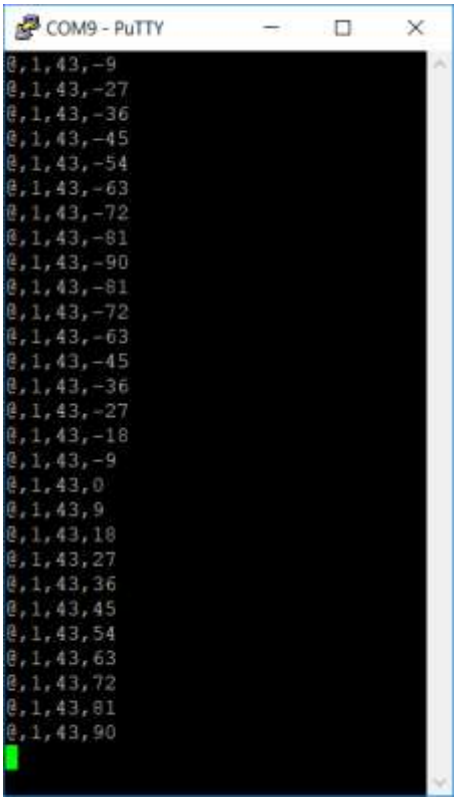
The user apps and Hig-Hub joystick bus hub are always address 0. When a Hig-Hub bus hub is present, it is the bus master, not the app. The app in that case is merely a front end for the bus hub. The bus hub has many of its own registers, which are different from the actuators' registers, and these can also be accessed by the user apps.

**IMPORTANT:
Because of bus arbitration issues, the desktop app should not be used on the actuator bus when a Hig-Hub bus hub is present. In this case, the desktop pc should be plugged into the "Host" port on the bus hub. When using a Hig-Hub hub, the hub issues the heartbeats, and heartbeats from the app are disabled.

By way of example, when a user clicks "Refresh actuator list," the app queries bus address 255, meaning all actuators, to send their bus addresses (register=0). In the case of a query, the value generally doesn't matter. But there must be a value, and here we use 1. In the screen grab below you can see actuator 1's response: actuator address 1, register=0, value=1. This example is unique because the address and value fields will always be the same number.



Here's what sliding the Jog slider looks like. We start at 0 velocity, slide to max negative, then to max positive.:



Bus arbitration and heartbeats:

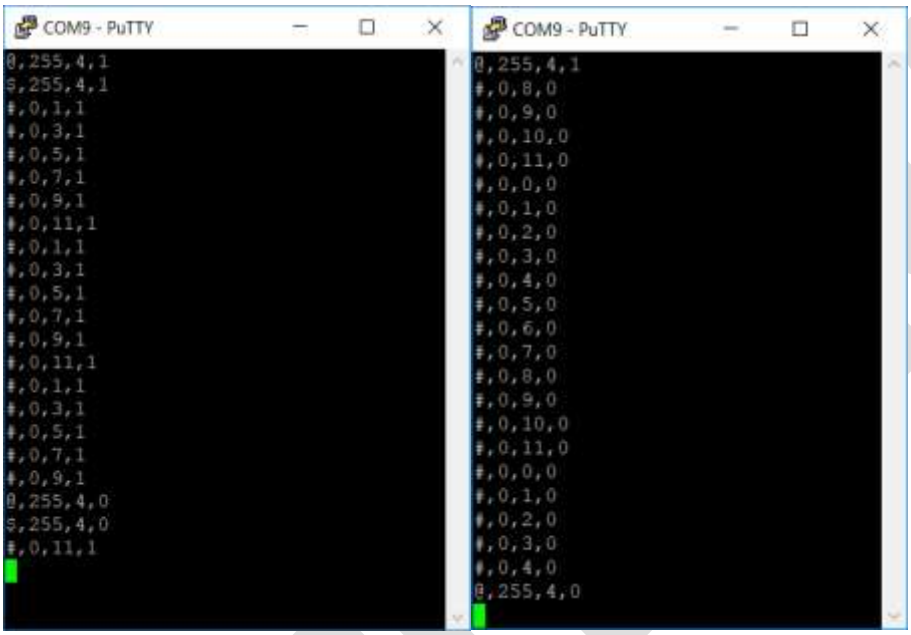
Because only one device can speak on the bus at a time, it's necessary to ensure that multiple devices' messages do not collide with each other. With the simple case of an app and a single actuator, this is easy to manage. But even here, problems can occur. Consider the case where the actuator is streaming its position and the app wants to send a message. Without some kind of synchronization, a collision could easily occur. We implement a simple scheme where a bus master (user app or Hig-Hub bus hub) issues periodic heartbeat messages. The mover bus intervals are 33ms; the desktop app's intervals are 99ms with a Hig-Hub as bus master, or 50ms when the Windows app is master. An actuator may only speak after receiving a heartbeat, with a delay time dependent on its bus address. So, for instance, actuator 2 speaks 1ms later than actuator 1. No actuator may speak during the first 8ms immediately after a heartbeat message. This is the master's time to send additional messages, such as user-initiated messages or joystick velocity messages from a Hig-Hub bus hub. The bus master is always address 0. Heartbeats or acknowledgements from the app will always start with "#", and will conform to the following:

#	,	0	,	5	,	1	CR-LF	[CRC]
Heartbeat	Comma delimiter	App or hub bus address	Comma delimiter	Heartbeat clock. Cycles 0 to 11.	Comma delimiter	1 = CRC 0 = no CRC	Terminator Carriage-return + line feed	Optional 16-bit CRC

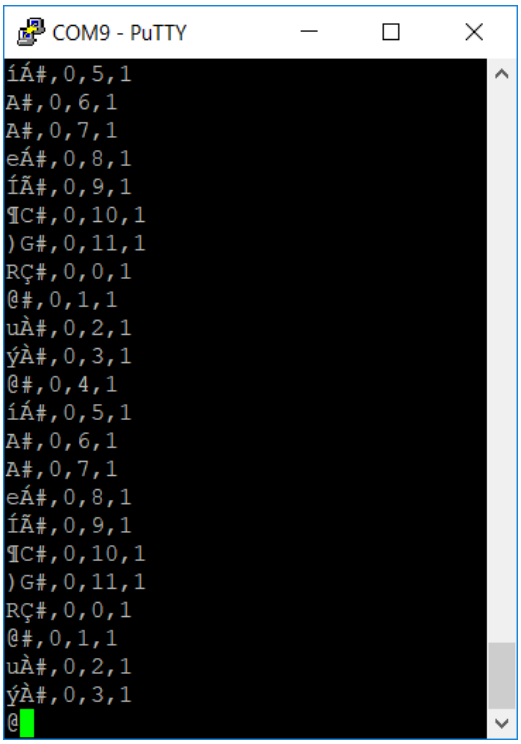
For ease of debugging, the heartbeat can be disabled from the apps, by issuing message “@,255,4,0”. This stops the app from issuing the heartbeats, and instructs the actuators not to wait for them, but to speak at will.

The user apps and Hig-Hub joystick bus master are always address 0. When a Hig-Hub joystick bus master is present, it is the bus master, not the app. The app in that case is merely a front end for the bus master. Therefore, it will be the bus master issuing the heartbeats, not the app.

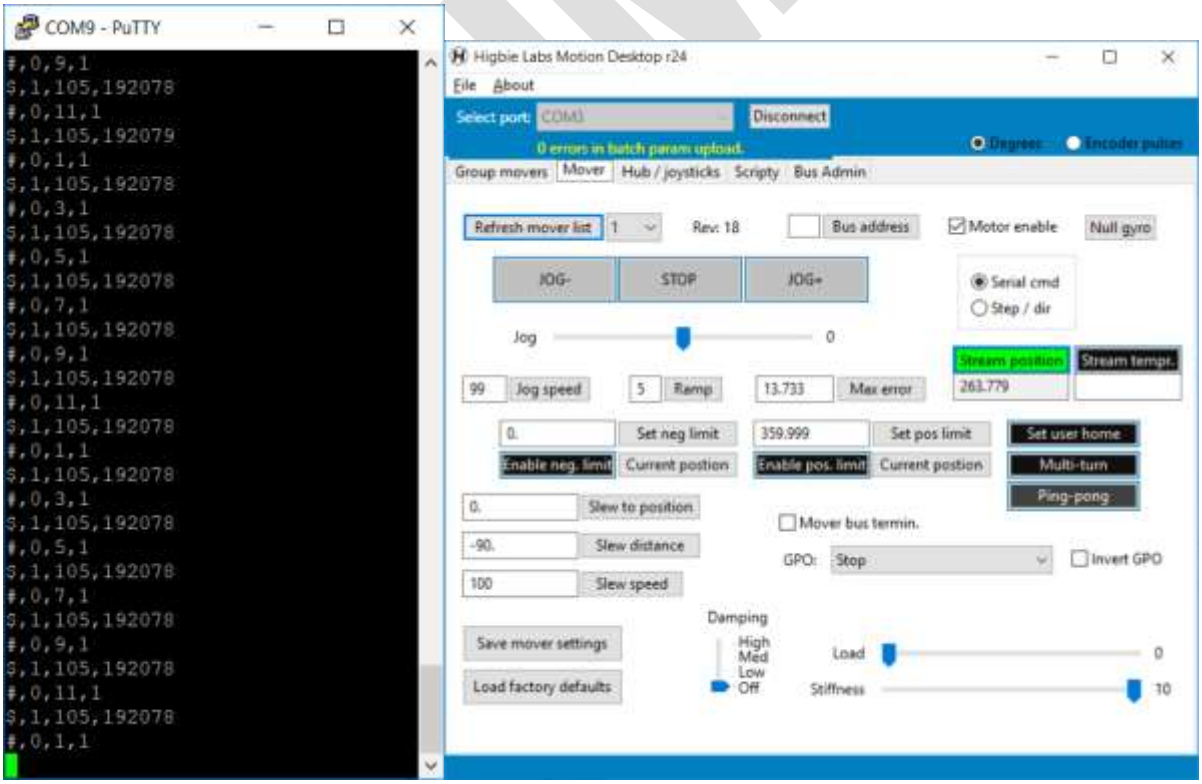
Below are examples showing started and stopped, for the pc app bus on left, and the mover bus on the right. You can see that the pc bus clock only shows every third count. This is to save pc serial port headroom. The mover bus shows every count. The purpose of the clock counter is that streaming data can easily be arbitrated at a lower rate to conserve bus bandwidth at the pc or Bluetooth. Also, with the pc app internal clock so close to the heartbeat frequency, we can avoid collisions by having movers and hub speak on even heartbeats while the app speaks on odd heartbeats.



In these examples you can also see from the last number that crc’s were disabled for capturing the mover bus, for the sake of readability. Below is what it looks like with crc’s enabled:



The next views show streaming position behavior with heartbeat enabled. To reduce chances of communication link saturation, the position messages are only sent every other heartbeat. You can also see that the conversion between encoder pulses and degrees is being done in the app. Raw position 192078 on the bus equals 263.779 degrees in the app.



**IMPORTANT:
Because of bus arbitration issues, the desktop app should not be used on the actuator bus when a Hig-Hub bus hub is present. In this case, the desktop pc should be plugged into the "Host" port on the bus hub.

Handshaking

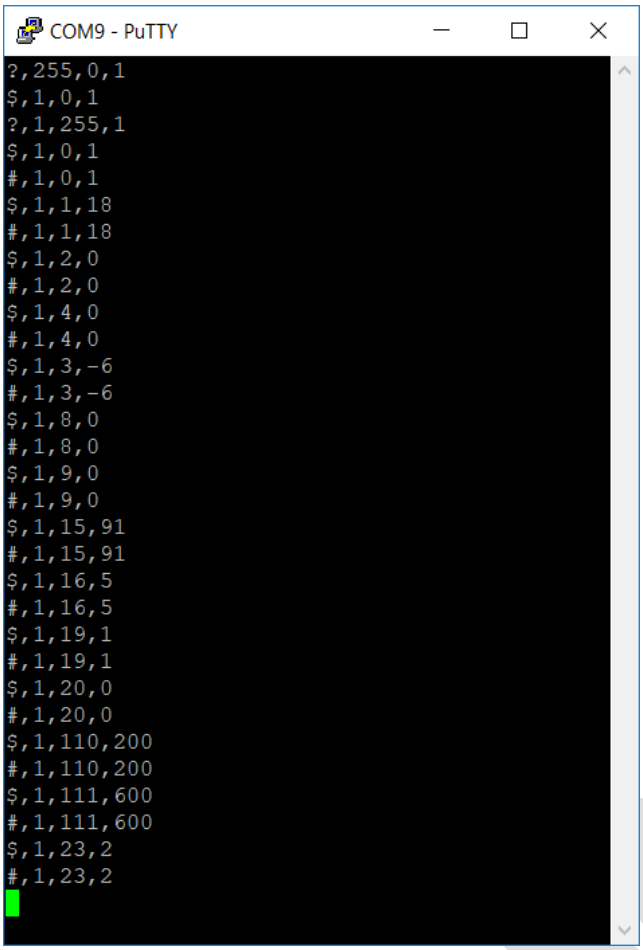
The Hig-Bus system provides limited handshaking and flow control. The recipient of a message is expected to issue an acknowledgement within a certain amount of time. If an acknowledgement is not received, the sender re-sends the message once. This applies to all messages except for global commands (bus address = 255), real-time motion commands (joysticks) and streaming position or temperature. For a query, the response to the query serves as the acknowledgement. A mover's or hub's acknowledgement to a command from the app is character "$," the target's bus address, and the commanded register and value.

Parameter dump on connect

When the app first connects to a bus hub or a mover, the hub or mover sends all its parameters in a batch to the app so it can populate its gui. In this case, flow control is provided to help prevent losses in the large bulk of data. After the app receives a parameter, it acknowledges in lieu of heartbeat with "#," the sender's bus address, the parameter register and the value that it received. If the sender does not receive this acknowledgment, it will re-send the message once. At the end of the batch, the hub or mover will send a total count of the number of re-sends, with register 150: $,[address],150,{re-sends]. The number of re-sends will be displayed at the top of the app.

Below, you can see what happens when the user selects the mover from the pulldown. The app issues a query for all registers in the selected mover (register=255, message= "?,1,255,1") in line 3. Also, internally, the app assigned bus address 1 for all future messages, until a different actuator is selected. This screen grab also illustrates the acknowledgements from the app, using the "#" start character.

To speed up the batch transfer, the Windows app precedes the batch query with the command to turn off heartbeats. (255, 4, 0). At the end of the transfer, it turns heartbeats back on if they had been on before the batch transfer (255, 4, 1).



Actuator bus schedule:

0ms to 9ms		10ms	11ms	12ms	13ms	14ms	15ms	16ms	17ms
#Hub/app Heartbeat	@Joysticks and/or commands	$ Mover 1	$ Mover 2	$ Mover 3	$ Mover 4	$ Mover 5	$ Mover 6	$ Mover 7	$ Mover 8

Optional CRC error-checking

When crc is enabled, all message characters, including terminating CR-LF, are run through a 16-bit CCITT crc using polynomial 0x8005. The crc is transmitted low byte first after the end of each message. The crc is only used on the actuator bus.

Transfer of “Scripty” programs between app and Hub

For loading “Scripty” programs into the hub from the Windows app, the app first sends the SCRIPTY LOAD FROM PC command (register 109). The Hub resets its non-volatile memory to the beginning and waits for program instructions. The app sends the program instructions with leading character “S.” This distinguishes the program instructons from normal, immediate instructions. The Hub’s acknowledgements use leading character “y.”

“Hig-Mover” actuator register map

Hig-Mover actuator register map **UNUSED ADDRESSES ARE RESERVED. DO NOT USE!!**

REGISTER NAME	REGISTER NUMBER	TYPE	USE AS BOOLEAN	NOTES
BUS_ID	0	SIGNED 16-BIT		
FIRMWARE_REVISION	1	SIGNED 16-BIT		
TERM_485_STATE	2	SIGNED 16-BIT	X	
RESERVED	3	SIGNED 16-BIT		
USE_HEARTBEATS	4	SIGNED 16-BIT	X	
PING_PONG	5	SIGNED 16-BIT	X	
ENA_TEMP_STREAM	6	SIGNED 16-BIT	X	
COILS_TEMP	7	SIGNED 16-BIT		
RESERVED	8	SIGNED 16-BIT	X	
RESERVED	9	SIGNED 16-BIT	X	
LP_FILTER	10	SIGNED 16-BIT		"Damping"
RESERVED	11	SIGNED 16-BIT	X	
LOG_JOYSTICK_CURVE	12	SIGNED 16-BIT	X	
USER_HOME	13	SIGNED 16-BIT	X	
MULTI_TURN	14	SIGNED 16-BIT	X	
SLEW_MAX_SPEED	15	SIGNED 16-BIT		
SLEW_RAMP_INCR	16	SIGNED 16-BIT		
USER_HOME_SET	17	SIGNED 16-BIT	X	
STOP_CMD	18	SIGNED 16-BIT	X	
COILS	19	SIGNED 16-BIT	X	
BOOST	20	SIGNED 16-BIT		
PRESET_FROM_CURRENT	21	SIGNED 16-BIT	X	
GO_PRESET	22	SIGNED 16-BIT	X	
WAVE_FREQUENCY	23	SIGNED 16-BIT		sq=0, tri=1, sin=2
SLEWING_IN_PROGRESS	24	SIGNED 16-BIT	X	
RESERVED	25	SIGNED 16-BIT		
RESERVED	26	SIGNED 16-BIT		
RESERVED	27	SIGNED 16-BIT		
PP_GAIN	28	SIGNED 16-BIT		
PI_GAIN	29	SIGNED 16-BIT		
PD_GAIN	30	SIGNED 16-BIT		
RP_GAIN	31	SIGNED 16-BIT		
RI_GAIN	32	SIGNED 16-BIT		
RD_GAIN	33	SIGNED 16-BIT		
RESERVED	34	SIGNED 16-BIT	X	
RESERVED	35	SIGNED 16-BIT	X	
RESERVED	36	SIGNED 16-BIT	X	
RESERVED	37	SIGNED 16-BIT	X	
RESERVED	38	SIGNED 16-BIT	X	
POS_LIMIT_ENABLE	39	SIGNED 16-BIT	X	

NEG_LIMIT_ENABLE	40	SIGNED 16-BIT	X	
PERCENT_CURRENT	41	SIGNED 16-BIT		
RESERVED	42	SIGNED 16-BIT	X	
JOG_SLIDER	43	SIGNED 16-BIT		
JOG_CW_CCW	44	SIGNED 16-BIT		
ENA_POSN_STREAM	47	SIGNED 16-BIT	X	
STEP_DIR	48	SIGNED 16-BIT	X	
NEG_LIMIT_FROM_CURRENT	49	SIGNED 16-BIT	X	
POS_LIMIT_FROM_CURRENT	50	SIGNED 16-BIT	X	
JOG_MAX_SPEED	54	SIGNED 16-BIT		
DEADBAND	55	SIGNED 16-BIT		
CRC_ENABLE	56	SIGNED 16-BIT	X	
RESERVED	60	SIGNED 16-BIT		
RESERVED	61	SIGNED 16-BIT		
RESERVED	62	SIGNED 16-BIT		
FACTORY_RESTORE	63	SIGNED 16-BIT	X	
WAVE_RUN	64	SIGNED 16-BIT	X	
WAVE_SHAPE	65	SIGNED 16-BIT		
WAVE_CAPTURE	66	SIGNED 16-BIT	X	
WAVE_CAPTURE_TWO_TRACES	67	SIGNED 16-BIT	X	0=2 traces, 1= one trace
WAVE_DUMP_DATA	68	SIGNED 16-BIT	X	
WAVE_NUM_SAMPS	69	SIGNED 16-BIT		
ENABLE_SIMPLE_TUNING	70	SIGNED 16-BIT	X	
TUNING LOAD	71	SIGNED 16-BIT		
TUNING BANDWIDTH	72	SIGNED 16-BIT		
QUERY_TUNING	73	SIGNED 16-BIT	X	
GPIO_FUNC	74	SIGNED 16-BIT		
GPIO_INVERT	75	SIGNED 16-BIT	X	
INTEGRAL_CLAMP	76	SIGNED 16-BIT	X	
FOLLOWING_ERROR_FAULT	79	SIGNED 16-BIT	X	
RESERVED	77	SIGNED 16-BIT		
RESERVED	78	SIGNED 16-BIT		
OVERTEMP_FAULT	80	SIGNED 16-BIT	X	
QUERY STATUS	81	SIGNED 16-BIT		
CMD COMM STATUS	82	SIGNED 16-BIT		
JOG_CW	83	SIGNED 16-BIT	X	
JOG_CCW	84	SIGNED 16-BIT	X	
TRIG_SLEW_DISTANCE	85	SIGNED 16-BIT	X	
TRIG_SLEW_TARGET	86	SIGNED 16-BIT	X	
NEG_LIMIT	101	SIGNED 32-BIT		
POS_LIMIT	102	SIGNED 32-BIT		
SLEW_DISTANCE	103	SIGNED 32-BIT		
SLEW_TARGET	104	SIGNED 32-BIT		
CURR_POSN	105	SIGNED 32-BIT		

HOME_OFFSET	106	SIGNED 32-BIT		X
MAX_ERROR	107	SIGNED 32-BIT		
USER_HOME_OFFSET	108	SIGNED 32-BIT		
WAVE_MIN_POSITION	110	SIGNED 32-BIT		
WAVE_MAX_POSITION	111	SIGNED 32-BIT		
PRESET_1	112	SIGNED 32-BIT		
PRESET_2	113	SIGNED 32-BIT		
PRESET_3	114	SIGNED 32-BIT		
PRESET_4	115	SIGNED 32-BIT		
PRESET_5	116	SIGNED 32-BIT		
PAGE_QUERY	255	"1"		
EE_SAVE	255	"1"		

“Hig-Hub” bus master register map

Hig-Hub actuator register map UNUSED ADDRESSES ARE RESERVED. DO NOT USE!!

REGISTER NAME	REGISTER NUMBER	TYPE	USE AS BOOLEAN	NOTES
The following registers are accessed using bus address "0"				
JOYSTICK 0-7 TARGET ASSIGN	0-7	16-BIT SIGNED INTEGER		Select which mover / gain
JOYSTICK 0-7 ENABLES	8-15	16-BIT SIGNED INTEGER	X	
JOYSTICK 0-7 INVERT	16-23	16-BIT SIGNED INTEGER	X	
JOYSTICK 0-7 JOG-SWITCH	24-31	16-BIT SIGNED INTEGER	X	
JOYSTICK 0-7 SENSITIVITY	32-39	16-BIT SIGNED INTEGER		
JOYSTICK 0-7 DEADBAND	40-47	16-BIT SIGNED INTEGER		
JOYSTICK 0-7 SMOOTHING	48-55	16-BIT SIGNED INTEGER		
JOYSTICK 0-7 NULL	56-63	16-BIT SIGNED INTEGER	X	
JOYSTICK 0-7 ALT. FUNCTIONS	64-71	16-BIT SIGNED INTEGER		Not supported
RESERVED	72-79	16-BIT SIGNED INTEGER		
JOYSTICK 0-7 LOG CURVE	80-87	16-BIT SIGNED INTEGER	X	
LOAD FACTORY SETTINGS	89	16-BIT SIGNED INTEGER	X	
SCRIPTY RUN	106	16-BIT SIGNED INTEGER	X	
SCRIPTY STOP	107	16-BIT SIGNED INTEGER	X	
SCRIPTY SEND TO PC	108	16-BIT SIGNED INTEGER	X	
SCRIPTY LOAD FROM PC	109	16-BIT SIGNED INTEGER	X	
GPI 1-4 FUNCTIONS	110-113	16-BIT SIGNED INTEGER		
GPI 1-4 INVERT	114-117	16-BIT SIGNED INTEGER	X	
RESERVED	118-121	16-BIT SIGNED INTEGER		
RESERVED	124	16-BIT SIGNED INTEGER		
HUB-APP SERIAL TERMINATOR	125	16-BIT SIGNED INTEGER	X	
RESERVED	126	16-BIT SIGNED INTEGER		
HUB-MOVER SERIAL TERMINATOR	127	16-BIT SIGNED INTEGER	X	
HUB PRESENT	128	16-BIT SIGNED INTEGER	X	Dummy ack’er
HUB PRESENT 2	129	16-BIT SIGNED INTEGER	X	Dummy ack’er
GPO 1-4 FUNCTIONS	130-133	16-BIT SIGNED INTEGER		
GPO 1-4 INVERT	134-137	16-BIT SIGNED INTEGER	X	
PAGE_QUERY	255	"1"	X	
EE_SAVE	255	"1"	X	