APPLICATION NOTES -

PIGNONE P6008 RTU CUSTOM PROTOCOL

Bristol Babcock
Revision: January, 1999

TABLE OF CONTENTS

1. INTRODUCTION

These application notes describe the Pignone Custom protocol functionality in the 33XX RTUs. At present only the 386-Protected Mode RTUs will support this protocol. It will be combined with the Modbus protocols and will not be included in the standard protocol set.

2. REFERENCES

Refer to the following documents for additional information:

- PIGNONE P6008 RTU PROTOCOL ANALYSIS, September 15, 1998
- *D4044 - ACCOL II Reference Manual*
- *D4051 - ACCOL Workbench for Windows*
- *D4066 - ACCOL II Custom Protocols Manual*

3. DEFINITIONS

- 386EX Protected Mode RTU - A DPC 3330, DPC 3335, or RTU 3310 remote process controller, equipped with the 386EX Protected Mode CPU.

4. FUNCTIONAL REQUIREMENTS

The requirements for this custom protocol comes from the references document “Pignone P6008 Protocol Analysis”, dated September 15, 1998.

5. FUNCTION DESCRIPTION

This custom Interface allows 33XX to function as Pignone Master. At present only the 3330 Protected Mode RTUs will support this protocol. The Pignone Master mode (Master) is used when a 33XX has to collect data from or write data to the Pignone P6008 RTUs (Slaves). The Pignone Master performs appropriate conversion to/from ACCOL signals.

A 33XX can be set up to act as a Master to a number of Slave devices on one or more communications ports. All ports in the Protected mode can be configured as custom ports. When the 33XX is set up to act as a Master the programmer will usually assign one

ACCOL Custom module per slave, with each module specifying the same Communication port but a different Slave address. One Custom module can be used but the user must then change the Slave address before executing the module. When there are many slaves, more than one Custom port can be assigned to balance the activities.

A custom protocol is enabled in an ACCOL load when the load is configured with a proper Custom Port(s) and includes appropriate Custom module(s). The ACCOL Custom module is a generic module to support any one of the various custom protocols available but takes on a “Specific Identity” with the assigned custom mode. What follows is a description of the configuration that assigns this “Pignone Custom Protocol Identity” to the Custom module.

5.1 Setting Up the Custom Port

The communications port used for Pignone messages must be assigned as a Custom Port in the *COMMUNICATIONS section of the ACCOL source file. The port's mode of operation is selected as follows:

MODE - Set this to 165 (same as in the MODE terminal of the custom module described above.

BAUD - Set the port baud rate. Valid baud rates are from 300 to 38,400.

DATA BITS - The number of bits per character is fixed at 8.

STOP BITS - The number of stop bits are fixed at 1.

PARITY - The parity is fixed at Odd.

P1 - Retry count. Set this to Communications Attempt Count. This value specifies the number of attempts which will be made to communicate, i.e. the initial attempt, plus this count minus 1, following a failure. The default is 3 attempts total (the initial attempt, plus two retries.) Valid values for P1 are:

0 - will cause default of 3 to be used
1 - no retries (single attempt)
2 - 1 retry
3 - 2 retries (the default)

P2 - Not used.

5.2 Configuring the Custom Module

The ACCOL Custom module has specific terminals. These terminals when configured as described below changes the generic Custom module to act as the Pignone Protocol Master module to the connected Pignone Slave RTUs.

MODE
Default: None, entry required
Format: Analog signal or constant
Input/Output: Input must be set to 165 for Pignone Master mode.

LIST
Default: None, entry required
Format: Analog signal or constant
Input/Output: Input is the number of the signal list containing the signals needed to accomplish Pignone communications. This list is called the Pignone Master List.

STATUS
Default: None
Format: Analog signal
Input/Output: Output contains a status code value to indicate module status, communication line errors and other error conditions. Following codes apply:

2 = communication transaction in progress
1 = communication has been requested
0 = transaction completed successfully
-1 = This error is reported by the Custom module when it detects that the custom mode for Pignone is not supported by the current custom firmware.
-2 = invalid Custom (Pignone) List specified
-3 = invalid Port specified
-4 = invalid Slave Address specified
-5 = invalid Function Type code specified
-6 = invalid Response Timeout value specified
-7 = invalid I/O List specified
-8 = Execution Mode value is incorrect
-9 = Execution Time value is incorrect
-10 = Response message length or content invalid
-25 = input character overrun error detected
-26 = input character parity error detected
-27 = input character framing error detected
-28 = input security check (CRC) error detected
-29 = timed out waiting for response
-30 = invalid msg framing characters received
-31 = transmit timed out waiting for CTS

-32 = unexpected I/O failure

5.2.1 Pignone Master Signal List

The signal list assigned to the LIST terminal of the Custom Module must contain signals as follows.

Signal 1 (Port Number) -

An analog signal whose value identifies the port to be used for communication with Modbus slaves. The port must be a Custom port configured for mode X(tbd)

Value	Port
1.0	A
2.0	B
3.0	C
4.0	D
5.0	BIP1
6.0	BIP2
7.0	G
8.0	H
9.0	I
10.0	J

Signal 2 (Slave Address) -

An analog signal whose value specifies the address of the Pignone Slave RTU. Slave addresses may range from 1.0 to 63.0.

Signal 3 (Function Code) -

An analog signal whose value specifies the Function Code.

Signal Value	Function Code
1.0	INIT (initialize the status bytes in the slave RTU)
2.0	GSTA (Get Status from the slave RTU)
3.0	GEN (General Acquisition)
4.0	SA (Specified Analog): Read one or more specified Analog Inputs.
5.0	AB (Analog Measurement Blocks): Read from 1 to 4 blocks of analog points. Data Precision, Single or Double, depends on the selection (described later in this list)
6.0	SB (Signal Block): Read from 1 to 4 blocks of discrete points.
7.0	SS (Specified Signals): Read 1 or more bytes of discrete points.
8.0	MO (Momentary Output): Activate selected output point.
9.0	PO (Permanent Output): Drive mosaic light on the alarm panel.

Signal 4 (Input/Output List Number) -

An analog signal containing the number of the signal list used to control input or output of data. Signals in this list can be I/O signals capable of receiving and sending data, or the signals in the list can specify appropriate parameter/values for the selected function code. Configuration of this list is dependent on the function code and described later.

Signal 5 (Precision) -

A logical signal that makes selection between Doubl or Single precision analog values:

ON - Double precision

Double Precision value are derived from 2 successive bytes in the response message. The first byte and the high order nibble of the second byte represent to value and the low nibble of the second byte includes four flags - reported as an analog value. Thus the maximum data value can range from 0x000 to 0xFFF. For example: 23 8C results in the data value of 0x238 or 568 decimal and flag value of 12 decimal.

OFF- Single precision (Default)

Single Precision value are derived from a single byte in the response message. By the protocol definition this byte is equal to

the most significant 2 nibbles of the double precision value and the third nibble as defaulted to 0. Thus the Single Precision values may be imprecise, compare to the double precision values, by 0-15 only. There are no flags associated with the data values. The maximum data value can range from 0x00x to 0xFFx. For example: value 23 in the message is equated to 0x230 hex or 560 decimal.

Signal 6 (Count) -

An analog signal containing the number of inputs to read (outputs to write). Default is 1. Value depends on the Function Code. Also length of some of the responses depends on this value, see individual command description for details.

Signal 7 (Index) -

An analog signal containing the first input to read (output to write). Value can range from 0-n . Default is 0 (the first of several consecutive).

Signal 8 (Done) -

An analog or logical signal used to indicate completion of a communication request, i.e., a message was sent and a reply received or a timeout occurred. Logical signals are set Off when the communication request is initiated and On when the communication request is complete. Analog signals are incremented by 1 when the request is complete.

Signal 9 (Response Timeout) -

An analog signal whose value specifies the amount of time in seconds to wait for a response message from the slave. The value can range from 0.0 to 65.534 seconds. System time resolution is 4 msec. When this signal is set to 0 the response is expected immediately, i.e. depending on the next clock tick which can be anywhere from 0 to 4 msec. If a response is not received immediately after the message is transmitted then a -29 error is reported. The response time out is distinct from the RTS/CTS Delay (defined later) and begins after the request message has been transmitted.

Signal 10 (RTS/CTS Delay) -

This is an optional analog signal which specifies a time delay. The time delay can either be used to monitor for CTS being raised, or to delay transmitting a message; the choice of how it is used is specified using “RTS/CTS Delay Mode”. The delay value can range from 0.001 to 65.534 seconds. If Signal “RTS/CTS Delay Mode” is not defined as an analog signal, or if its value is out of the specified range, then a default delay of 2.5 seconds will be used.

Signal 11 - (RTS/CTS Delay Mode) -

This is an optional signal which specifies how the RTS/CTS delay will be used. There are two choices for the mode.

- Monitor For CTS Mode: After RTS is raised, the time delay will be used as the maximum time to wait within which CTS must be received. If CTS is received at any time before this delay expires, the message transmission begins. If CTS is NOT received prior to the expiration of the delay, no response will be sent, and a -31 error will be reported. If this is an analog signal, this mode is activated by setting its value to a positive number. If this is a logical signal, this mode is activated by turning the signal ON. If this is a string signal, this mode will automatically be chosen. If "RTS/CTS Delay" signal is wired, but this signal is omitted, this mode will automatically be chosen.

- Message Transmit Delay Mode: After RTS is raised, a delay timer will be started. (The length of the delay is determined by the value of the "RTS/CTS Delay Mode" signal). No message will be sent until after this delay has expired. The value of CTS does not affect the operation of this mode. NOTE: In order for this mode to work, RTS-CTS must be jumpered or the CTS must be received before the specified delay expires. If this is an analog signal, this mode is activated by setting it to 0. If this signal is a logical signal, this mode is activated by turning it OFF.

Signal 12 - (Communication Monitor List Number) -
An analog signal containing the number of the signal list used to monitor communication activity. This list is described later.

5.2.2 Communication Monitor List

Signal 1 - (Received Message String):
A string signal. It must be 64 byte long. When this signal is set to CE (Control Enabled), it is used to store the received messages. When it is set to CI (Control Inhibited) it is not updated. In normal operation this signal should be set to CI to avoid unnecessary overhead on the CPU. Set it to CE mode only while in debugging mode.

Signal 2 - (Transmitted Message String):
A string signal. It must be 64 byte long. When this signal is set to CE (Control Enabled), it is used to store the transmitted messages. When it is set to CI (Control Inhibited) it is not updated. In normal operation this signal should be set to CI to avoid unnecessary overhead on the CPU. Set it to CE mode only while in debugging mode.

5.2.3 I/O List

Note: This document does not describe the command format as it is available in the referenced requirement document: Pignone P6008 RTU Protocol Analysis.

The I/O List used by the Master to format input/output requests and as a result may require unique configuration for each function code. It is the ACCOL programmer responsibility to configure the I/O list correctly. In general:

- If a mismatch is found between the signal types and input/ measurement/ output/status type then on the input functions signals are not updated and a warning is reported and processing is aborted at that point. On the output functions an error is reported and command is not executed.
- If the list does not contain enough number of signals then on the input functions the processing is stopped after the last signal has been updated and an overflow warning is reported. On the output functions an error is reported and command is not executed.
- The COUNT and INDEX signals are optional signals in that there are defaults associated with these signals depending on the function type which may or may not be sufficiently valid for the given function code type.

5.2.3.1 INIT (initialize status bytes) - Function Code: 01

NA - List is not required.

This message does not generate a response from the slave RTU.

5.2.3.2 GSTA (Get Status) - Function Code: 02

This function allows the user to inquire if any changes have occured in the RTU.

When there are no changes outstanding the response to this message is 2 bytes long that includes the address and checksum bytes. When one or more changes are outstanding the response to this message is 4 bytes long that includes the address and checksum bytes.

The I/O list must include following 16 signals. If the list is not configured accordingly an error is reported and command is not executed. When the response indicates there are no outstanding changes all of the following signals are set to OFF/FALSE. If the response indicates one or more changes then the signals are updated based on the corresponding change status in the response.

Note that bits from the response bytes are mapped to signals in random order.

Signal 1 - A logical signal to report if changes were detected to signals of Signal Block 0 - Message Byte 1 - Bit 1.

Signal 2- A logical signal to report if changes were detected to signals of Signal Block 1 - Message Byte 1 - Bit 2.
Signal 3- A logical signal to report if changes were detected to signals of Signal Block 2 - Message Byte 2 - Bit 1.
Signal 4- A logical signal to report if changes were detected to signals of Signal Block 3 - Message Byte 2 - Bit 2.
Signal 5- A logical signal to report if changes were detected to Measurements in Block 0 - Message Byte 1 - Bit 3.
Signal 6- A logical signal to report if changes were detected to Measurements in Block 1 - Message Byte 1 - Bit 4.
Signal 7- A logical signal to report if changes were detected to Measurements in Block 2 - Message Byte 2 - Bit 3.
Signal 8- A logical signal to report if changes were detected to Measurements in Block 3 - Message Byte 2 - Bit 4.
Signal 9- A logical signal to report the RTU's Restart status - Message Byte 1 - Bit 5.
Signal 10- A logical signal to report Event Detected state - Message Byte 1 - Bit 6.
Signal 11- A logical signal to report Incomplete Sequence status - Message Byte 1 - Bit 7.
Signal 12- A logical signal to report Sequence Breakout status - Message Byte 1 - Bit 8.
Signal 13- A logical signal to report an Analog to Digital Conversion Failure - Message Byte 2 - Bit 5.
Signal 14- A logical signal to report Events List overflow - Message Byte 2 - Bit 6.
Signal 15- A logical signal to report Sequence Running status - Message Byte 2 - Bit 7.
Signal 16- A logical signal to report Peripheral input request - Message Byte 2 - Bit 8.

5.2.3.3 GEN (General Acquisition) - Function Code: 03

This general acquisition function allows the users to acquire all 8 Analog input registers and all 16 Discrete Input registers. Received values are reported in the 16 signals provided in the I/O list.

The response to this message is 20 bytes long that includes the address and the checksum bytes.

If it is desired to have the received analog status flags associated with the analog values then the corresponding analog signals must be alarm signals. This will allow to report the received status bits to the corresponding flags of the analog signals as follows:

Status	Destination Flag
H	HIHI Alarm
L	LOLO Alarm
O	HI Alarm
V	Questionable

If the list is not configured as follows then a warning is reported and update takes place based on the signal type from the I/O list imposed on the received response message (i.e. each analog signal is mapped to 2 bytes of data and each logical signal is mapped to 1 byte of data).

		Example Value* Flag
Signal 1-	An analog signal to store Analog Input Register 1 - Message Bytes 1-2	800.0
Signal 2-	An analog signal to store Analog Input Register 2 - Message Bytes 3-4	3999.0
Signal 3-	An analog signal to store Analog Input Register 3 - Message Bytes 5-6	2131.0
Signal 4-	An analog signal to store Analog Input Register 4 - Message Bytes 7-8	1924.0
Signal 5-	An analog signal to store Analog Input Register 5 - Message Bytes 9-10	1522.0
Signal 6-	An analog signal to store Analog Input Register 6 - Message Bytes 11-12	1765.0
Signal 7-	An analog signal to store Analog Input Register 7 - Message Bytes 13-14	3.0
Signal 8-	An analog signal to store Analog Input Register 8 - Message Bytes 15-16	1.0
Signal 9-	A logical signal to store Discrete Input 1 - Message Byte 17 - Bit 1	1
Signal 10-	A logical signal to store Discrete Input 2 - Message Byte 17 - Bit 2	0
Signal 11-	A logical signal to store Discrete Input 3 - Message Byte 17 - Bit 3	1
Signal 12-	A logical signal to store Discrete Input 4 - Message Byte 17 - Bit 4	0
Signal 13-	A logical signal to store Discrete Input 5 - Message Byte 17 - Bit 5	1
Signal 14-	A logical signal to store Discrete Input 6 - Message Byte 17 - Bit 6	1
Signal 15-	A logical signal to store Discrete Input 7 - Message Byte 17 - Bit 7	1
Signal 16-	A logical signal to store Discrete Input 8	1

- Message Byte 17 - Bit 8

Signal 17- A logical signal to store Discrete Input 9 1
- Message Byte 18 - Bit 1

Signal 18- A logical signal to store Discrete Input 10 0
- Message Byte 18 - Bit 2

Signal 19- A logical signal to store Discrete Input 11 1
- Message Byte 18 - Bit 3

Signal 20- A logical signal to store Discrete Input 12 1
- Message Byte 18 - Bit 4

Signal 21- A logical signal to store Discrete Input 13 1
- Message Byte 18 - Bit 5

Signal 22- A logical signal to store Discrete Input 14 0
- Message Byte 18 - Bit 6

Signal 23- A logical signal to store Discrete Input 15 0
- Message Byte 18 - Bit 7

Signal 24- A logical signal to store Discrete Input 16 0
- Message Byte 18 - Bit 8

Signal 25- An analog signal to pack above 16 logicals 7669.0

Signal 26- An analog signal to store 4 bit alarm value 1
for the Analog Input Register 1 - Message Byte 2 - low nibble

Signal 27- An analog signal to store 4 bit alarm value 1
for the Analog Input Register 2
- Message Byte 4 - low nibble

Signal 28- An analog signal to store 4 bit alarm value 1
for the Analog Input Register 3
- Message Byte 6 - low nibble

Signal 29- An analog signal to store 4 bit alarm value 1
for the Analog Input Register 4
- Message Byte 8 - low nibble

Signal 30- An analog signal to store 4 bit alarm value 1
for the Analog Input Register 5
- Message Byte 10 - low nibble

Signal 31- An analog signal to store 4 bit alarm value 1
for the Analog Input Register 6
- Message Byte 12 - low nibble

Signal 32- An analog signal to store 4 bit alarm value 5
for the Analog Input Register 7
- Message Byte 14 - low nibble

Signal 33- An analog signal to store 4 bit alarm value 5
for the Analog Input Register 8
- Message Byte 16 - low nibble

* Received Message:

32 01 F9 F1 85 31 78 41 5F 21 6E 51 00 35 00 15 F5 1D

Input#	Value	Status HLOV	Count (Hex/Decimal)
AI-1	32 01	0001	320/800
AI-2	F9 F1	0001	F9F/3999
AI-3	85 31	0001	853/2131
AI-4	78 41	0001	784/1924
AI-5	5F 21	0001	5F2/1522
AI-6	6E 51	0001	6E5/1765
AI-7	00 35	0101	003/003
AI-8	00 15	0101	001/001
DI-1/16	F5 1D		

5.2.3.4 SA (Specified Analog): - Function Code: 04

This function allows for the reading of 1 or more contiguous analog points on the AI card.

Precision - may be set to Single precision or Double precision; default of Single precision is assumed. If the request was for the Double Precision data then it will include the status flags (H, L, O, and V) which are reported as described under the GEN Function Code.

Count - set this to “m” to indicate number of inputs to read; default reads 1 input. Response length is affected by this value.

Index - set to indicate the first input to read (0-n); default of 0 (the first input) is assumed.

The response to this message is, 2 + m * 2 bytes long (when in Double Precision) or 2 + m bytes long (when in Single Precision). In both cases the message length includes the address and checksum bytes.

When the response is in double precision the list should include “m * 2” number of analog signals. The inputs are stored in signals 1-m in this list and the associated flags are stored in signals m + 1 to m + m. When the response is in single precision the list should include “m” number of analog signals. The inputs are stored in signals 1-m in this list. There are no associated flags. If this list is shorter then after updating all the signals in the list, an overflow error is reported at the status terminal.

Signal 1- An analog signal to store Analog Input Register 1
Signal 2- An analog signal to store Analog Input Register 2

: :
Signal m- An analog signal to store Analog Input Register m
Signal m+1- An analog signal to store flags for Analog Input Register 1 (DP)
Signal m+2- An analog signal to store flags for Analog Input Register 2 (DP)
: :
Signal m+m- An analog signal to store flags for Analog Input Register m (DP)

5.2.3.5 AB (Analog Measurement Blocks) - Function Code: 05

This Function Code allows for the reading of 1 to 4 blocks of the analog points configured in the RTU. It is assumed that each block contains exactly 8 measurements.

Precision - may be set to Single precision or Double precision; default of Single precision is assumed. If the request was for the Double Precision data then it will include the status flags (H, L, O, and V) which are reported as described under the GEN Function Code.

Count - set this to "m" indicating the number of blocks to read (1-4); a default of 1 (Block 0) is assumed. Response length is affected by this value.

Index - set to indicate the first Block to read (0-3) or default of 0 (the first measurement) is assumed.

The response to this message is 2 + (m * 8 * 2) bytes long (when in Double Precision) or 2 + (m * 8) bytes long (when in Single Precision). In both cases the message length includes the address and checksum bytes.

When the response is in double precision the list should include m * 8 * 2 analog signals. The measurements are stored in signals 1 to (m* 8) in this list and the associated flags are stored in signals (m * 8)+1 - ((m + m) * 8). When the response is in single precision the list should include 1 to (m* 8) analog signals for the expected measurements. If this list is shorter then after updating all the signals in the list, an overflow error is reported at the status terminal.

Signal 1- An analog signal to store Analog Input Register 1
Signal 2- An analog signal to store Analog Input Register 2
: :
Signal m* 8- An analog signal to store Analog Input Register m
Signal m* 8+1-An analog signal to store flags for Analog Input Register 1 (DP)
Signal m* 8+2-An analog signal to store flags for Analog Input Register 2 (DP)
: :
Signal (m+m)* 8-An analog signal to store flags for Analog Input Register m (DP)

5.2.3.6 SB (Signal Block) - Function Code: 06

This function allows for the reading of 1 to 4 blocks (B0, B1, B2, and B3) of Discrete points (DI). It is assumed that each block represents 16 DIs, i.e. 2 bytes.

Count - set this to "m" to the number of blocks to read (1-4); a default of 1 (Block 0) is assumed. Response length is affected by this value.

Index - set this to "n" to indicate the first Block to read (0-3) or default of 0 (Block 0) is assumed.

The response to this message is 2 + m * 2 bytes long that includes the address and checksum bytes.

List must include m * 2 * 8 logical signals. If this list is shorter then after updating all the signals in the list, an overflow error is reported at the status terminal. The DIs in the response message are mapped to the signals in the list as follows:

		Example Value*
Signal 1-	SB n - Discrete Input 1 - Message Byte 0 - Bit 1	1
Signal 2-	SB n - Discrete Input 2 - Message Byte 0 - Bit 2	0
Signal 3-	SB n - Discrete Input 3 - Message Byte 0 - Bit 3	1
Signal 4-	SB n - Discrete Input 4 - Message Byte 0 - Bit 4	1
Signal 5-	SB n - Discrete Input 5 - Message Byte 0 - Bit 5	1
Signal 6-	SB n - Discrete Input 6 - Message Byte 0 - Bit 6	1
Signal 7-	SB n - Discrete Input 7 - Message Byte 0 - Bit 7	1
Signal 8-	SB n - Discrete Input 8 - Message Byte 0 - Bit 8	1
Signal 9-	SB n - Discrete Input 9 - Message Byte 1 - Bit 1	1
Signal 10-	SB n - Discrete Input 10- Message Byte 1 - Bit 2	0
Signal 11-	SB n - Discrete Input 11- Message Byte 1 - Bit 3	1
Signal 12-	SB n - Discrete Input 12- Message Byte 1 - Bit 4	1
Signal 13-	SB n - Discrete Input 13- Message Byte 1 - Bit 5	1
Signal 14-	SB n - Discrete Input 14- Message Byte 1 - Bit 6	0
Signal 15-	SB n - Discrete Input 15- Message Byte 1 - Bit 7	0
Signal 16-	SB n - Discrete Input 16- Message Byte 1 - Bit 8	0

Note: As above add 16 logical signals for each additional signal block.

* Example: COUNT = 1, INDEX = 0 and the received message included data bytes "F5 1D" then the first 16 signals are updated as above.

5.2.3.7 SS (Specified Signals) - Function Code: 07

This function allows for the reading of 1 or more contiguous bytes of Discrete points (DI). Each "byte" of response includes 8 DIs.

The response to this message depends on the Count field. Generally this can be 1 or 2 bytes of DIs leading to 3 or 4 bytes long response that includes the address and the checksum bytes.

Count - set this to "m" to indicate number of "bytes" to read; a default of 1 byte (8 DIs) is assumed.

Index - set to indicate the first byte to read (0-n); a default of 0 (the first byte) is assumed.

List must include m * 8 logical signals. Each successive data byte in the response maps to eight successive signals in this list. If this list is shorter then after updating all the signals in the list, an overflow warning is reported at the status terminal. The DIs in the response are mapped to the signals in the list as described for the Signal Block function.

5.2.3.8 MO (Momentary Output) - Function Code: 08

This function allows for the activation of the selected output channel/point on the selected output card.

The response to this message is 4 bytes long and includes the checksum. It must be identical to the request message.

Count - set to indicate the Output Card Number; a default of 1 (the first Output Card).

Index - set to indicate the Channel/Point Number on the Output Card. Value can range from 0-7 to correspond to 1^{st}-8^{th} Channel/Point respectively. A default of 1, the second channel, is assumed.

List must include following signals otherwise an error is reported and command is not executed.

Signal 1 - Analog signal (required) - this signal specifies the Execution Mode:
- 01 = Immediate Execution
- 04 = Selection in Check-back type
- 05 = Execution in Check-back type

Signal 2 - Analog signal (required) - this signal specifies the Execution Time:

00 = Execution time is short
02 = Execution time is long.

When the execution mode is “Immediate Execution” the custom module need to be run only once. If it is desired to exercise the Check-back style of interface then first execute the custom module with the Execution Mode set to “Selection in Check-back type” wait for the response from the RTU and if there are no errors then execute the custom module again with the Execution Mode set to “Execution in Check-back type”.

5.2.3.9 PO (Permanent Output) - Function Code: 09

This function allows to drive the mosaic lights on the alarm panel.

The response to this message is 6 bytes long and includes the checksum. It must be identical to the request message.

Count - is not used.

Index - set to the output Word Number (0-127); a default of 0 (the first word) is assumed.

List must include following signals otherwise an error is reported and command is not executed.

Signal 1 - Analog signal (required) - this signal specifies the Execution Mode:
01 = Immediate Execution
04 = Selection in Check-back type
05 = Execution in Check-back type

		Example Value*
Signal 2-	Output Byte 0 - Bit 1	1
Signal 3-	Output Byte 0 - Bit 2	0
Signal 4-	Output Byte 0 - Bit 3	1
Signal 5-	Output Byte 0 - Bit 4	1
Signal 6-	Output Byte 0 - Bit 5	1
Signal 7-	Output Byte 0 - Bit 6	1
Signal 8-	Output Byte 0 - Bit 7	1
Signal 9-	Output Byte 0 - Bit 8	1
Signal 10-	Output Byte 1 - Bit 1	1
Signal 11-	Output Byte 1 - Bit 2	0
Signal 12-	Output Byte 1 - Bit 3	1
Signal 13-	Output Byte 1 - Bit 4	1
Signal 14-	Output Byte 1 - Bit 5	1
Signal 15-	Output Byte 1 - Bit 6	0

Signal 16- Output Byte 1 - Bit 7 0
Signal 17- Output Byte 1 - Bit 8 0

* Transmitted data bytes: F5 1D

When the execution mode is "Immediate Execution" the custom module need to be run only once. If it is desired to exercise the Check-back style of interface then first execute the custom module with the Execution Mode set to "Selection in Check-back type" wait for the response from the RTU and if there are no errors then execute the custom module again with the Execution Mode set to "Execution in Check-back type".

5.2.3.10 ACK (Acknowledgment) - Function Code: 10

NA - List is not required.

This message generates an echo back response from the RTU.

The command is two bytes long: 4X, LCC

5.3 ACCOL Programming Considerations

The ACCOL task and module concept allows very flexible scanning and data collection schedule. Timing is strictly limited by the capabilities of the slave RTUs. Generally on any given port the Master performs message transactions with a single slave one at a time. Multiple ports allow to have multiple transactions in progress concurrently - albeit with one slave on each line/port. Programming concept is for Master to send a request and wait for the successful/error completion of the request and update of the DONE signal and then issue the next request, either to the same Slave RTU or to another Slave RTU on the same line/port. The request is further controlled by setting the Retry count at the parameter P1 of the Port Configuration and by setting the time out values in the Pignone Master List.

To simplify the ACCOL programming consider the following implementation: Create a separate custom module for each salve RTU. Configure these custom modules with the appropriate Master Signal List, and appropriate I/O list for the selected command. The separate custom module also allows to monitor for the status of the selected slave RTU. As mentioned earlier the RTU custom firmware is designed to handle one request-response transaction at a time. Any other requests are queued and served first in first out order. Also the multiple execution of the same custom module while a previous transaction is in progress is ignored until the transaction in progress is completed. When a transaction is completed it's status is reported at the STATUS terminal of the Custom Module and the DONE signal of the Master Signal List is updated. These two signals must be used to coordinate the processing of the completed transaction whether it is

successful or it has failed. The different scan rates can be programmed using either separate ACCOL tasks running at separate task rate or by controlling with the logic in the ACCOL task using the variables. Grouping of the custom modules whether by the scan rate or by port/line they interface on is strictly application dependent. In the extreme scenario one can thin see having separate task with a single Custom Module dedicated to a given slave RTU. This allows easy control over the scan rate handle successful / error completion states in simplified manner.

6. PIGNONE PROTOCOL MESSAGE INTERFACE

This section decribes all command messages from the Master to Slave and response messages from slave to the Master. This information is copied from the "Pignone P6008 RTU Protocol Analysis" document listed in the reference section.

6.1 GET STATUS (GSTA)

The GSTA function allows the user to inquire if any changes have occurred in the RTU. These changes may be related to various aspects of the RTU.

Request:

0	0	Address

Response for no changes outstanding:

1	1	Address

Response for some changes outstanding:

0	0	Address

SB	IS	E	R	M 1	M 0	S 1	S 0

PR	SR	EO	CF	M 3	M 2	S 3	S 2

Where: Sx = Changes detected in signals in blocks x
Mx = Changes detected in Measurements in blocks x
R = Restart of RTU.
E = Event detected.
EO = Events list overflow.
SR = Sequence running.
IS = Incomplete sequence.
SB = Sequence breakout.

PR = Peripheral input request.
CF = Analog to digital converter failure.

The following is an example of two GSTA (01) requests immediately after an INIT (C1) request:

Note that the last byte of each message is the checksum.

INIT Request: C1 40 7E
INIT Response: <none>

GSTA Request: 01 FE
GSTA Response: 01 15 00 EB

GSTA Request: 01 FE
GSTA Response: 01 04 00 FA

Note that the INIT request initializes the two status bytes of the RTU, and that this request does not have a reply.

The RTU restarts and sets the status bits R, M0, S0 indicating to the user that the RTU is in the process of restarting and that all measurements and signal blocks should be read.

6.2 GENERAL ACQUISTION (GEN)

The following request/response messages show the actual bytes of a GEN message exchange.

GEN Request: 84 7B
GEN Response: 84 32 01 F9 F1 85 31 78 41 5F 21 6E 51 00 35 00 15 F5 1D 44

Note that the last byte of each message is the checksum. The first 2 bits of the first byte contains the function code (8=GEN) and the last 6 bits contain the destination RTU address 4 found on that communication line.

For this request, all of the 8 analog input registers are returned first, followed by all of the 16 discrete inputs.

AI#	**COUNT (Hex)**	**DESCRIPTION**
AI-1 =	32 01	Reference input 1
AI-2 =	F9 F1	Reference input 2
AI-3 =	85 31	Pression tete puits
AI-4 =	78 41	Pression Collecteur
AI-5 =	5F 21	Debit Gaz
AI-6 =	6E 51	Pression annulaire

AI-7 = 00 35 Reserve 1
AI-8 = 00 15 Reserve 2
DI1-16 = F5 1D 16 Discrete inputs

As shown, 2 bytes are returned for each analog input. The raw count is contained in the first 3 nibbles (4bits/nibble) and the status bits (H,L,O,V) are contained in the last nibble. For example the 3rd analog input register contained a value of Hex 85 31.

8	5			
3	H	L	O	V

The raw count value for this analog input would be Hex 853 or 2131 counts (in decimal).

The status bits H and L are set when the respective High and Low alarm limits are violated. The overrange status bit O is set when the measurement exceeds the range of the ADC (Analog –to- Digital Converter).

The validity status bit V is normally set to 1 and it reflects the status of the ADC. This bit will be set to zero if the ADC on the card fails.

The following provides the discrete input mapping relative to the last two bytes of the GEN message response before the checksum.

TB0	**STATE**		**HEX**	**Note**
1	1			
2	0			
3	1			
4	0	5		
5	1			
6	1			
7	1			
8	1	F		
TB1	STATE			
9	1		Validity control bit =1 at all times.	
10	0			
11	1			
12	1	D		
13	1			
14	0			
15	0			
16	0	1		

Note that the validity control bit is simply a jumpered input on the digital input card, and it serves as a verification that all of the 16 contact inputs should never be zero. This

ensures that after an initialization of the data registers, the next GSTA request will report a change in the signal block.

6.3 SPECIFIED ANALOG (SA)

The SA function allows for the reading of 1 or more contiguous analog points on the AI card. The data points can be requested as double precision (2 bytes) or single precision (1 byte).

The following SA request is targeted for RTU at address 1 (byte1=C1 and High Nibble of byte2=7). The request asks for 4 (Low Nibble of byte2=3) contiguous analog points, starting after the 2nd analog input (Low Nibble of Byte3=2). The data is to be returned in double precision (High Nibble of Byte3=0).

Request:	C1	73	02	4F				
Response:	C1	88	51	8A	71	FF	FB	31 41	68
AI-n:		3		4		5		6	

The following SA request is targeted for RTU at address 1 (byte1=C1 and High Nibble of byte2=7). The request asks for 4 (Low Nibble of byte2=3) contiguous analog points, starting after the 2nd analog input (Low Nibble of Byte3=2). The data is to be returned in single precision (High Nibble of Byte3=8).

Request:	C1	73	82	CF		
Response:	C1	88	8A	FF	31	F2
AI-n:		3	4	5	6	

Note that the data is returned in single precision (the 2 most significant nibbles only) and the byte containing the least significant nibble and the status nibble is not returned. When using single precision option, the raw count accuracy is reduced by 15 counts in the worst case.

6.4 ANALOG MEASUREMENT BLOCKS, DOUBLE PRECISION (ABD)

The ABD function allows for the reading of 1 to 4 blocks of analog points configured in the RTU. The RTU has only the first block configured (B0). The following ABD (High Nibble of byte1=C and High Nibble of byte2=2) request asks for the reading of all of the analog points in RTU at address 1 (Low Nibble of byte1=1) that are configured in the first block (Low Nibble of byte2=1).

Request:	C1	21	1F								
Response:	C1	32	01	FA	01	88	51	8A	61	FF	FB	31	41	16	61	1531	E3
AI-n:		1		2		3		4		5		6		7		8	

6.5 ANALOG MEASUREMENT BLOCKS, SINGLE PRECISION (ABS)

The ABS function allows for the reading of 1 to 4 blocks of analog points configured in the RTU. The RTU has only the first block configured (B0). The following ABS (High Nibble of byte1=C and High Nibble byte2=1) request asks for the reading of all of the analog points in RTU at address 1 (Low Nibble of byte1=1) that are configured in the first block (Bit 1 of byte2=1). Note that the data is returned in single precision (the 2 most significant nibbles only) and the byte containing the least significant nibble and the status nibble is not returned.

Request:	C1	11	2F							
Response:	C1	32	FA	88	8A	FF	31	18	17	35
AI-n:		1	2	3	4	5	6	7	8	

Note that when using single precision option, the raw count accuracy is reduced by 15 counts in the worst case.

6.6 SIGNAL BLOCK (SB)

The Signal Block (SB) function allows for the reading of 1 to 4 blocks (B0,B1,B2,B3) of discrete points configured in the RTU. As witnessed by the INIT and GSTA request, only the first block (B0) is configured in the RTU. The following SB request to RTU at address 1 (byte1=C1) asks for the reading of this block (bit 1 of byte2=01) :

Request: C1 01 xx
Response: C1 F5 1D yy

The response to the above request returns the status of all the discrete inputs on the digital input card in byte2 and byte3, F5 and 1D respectively. Refer to the GENERAL REQUEST (GEN) description for the bit mapping relative to the input card.

6.7 SPECIFIED SIGNALS (SS)

The Specified Signal (SS) function allows for the reading of 1 or more contiguous bytes of discrete points. For example, the following SS request to RTU at address 1 (byte1=C1, High Nibble of Byte2=6) asks for the reading of 1 byte (Low Nibble of Byte=0) of discrete data starting at the second byte (Low Nibble of Byte=1) in the block:

Request: C1 60 01 xx
Response: C1 1D yy

The response to the above request returns the status of the lower half of the DI card (byte2=1D). Refer to the GENERAL REQUEST (GEN) description for the bit mapping relative to the input card.

The following SS request to RTU at address 1 (byte1=C1, High Nibble of Byte2=6) asks for the reading of 1 byte of discrete data starting at the first byte in the block:

Request: C1 60 00 xx
Response: C1 F5 yy

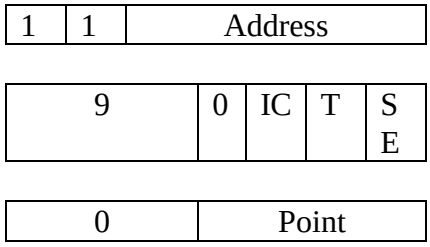
The response to the above request returns the status of the upper half of the DI card (byte2=F5). Refer to the GENERAL REQUEST (GEN) description for the bit mapping relative to the input card.

6.8 MOMENTARY OUTPUT (MO)

Each of the RTUs has one Contact output point on the second channel of an 8-contact output card.

The Operator interface was used to activate the output of RTU#1 on line 4 (01RN2). Using the time stamping feature of the serial test analyzer, the following messages were observed on the serial test analyzer at the time of the activation of the output from the operator console.

MO Request : C1 96 01 A9
MO Response:C1 96 01 A9
MO Request : C1 97 01 A8
RO Response: C1 97 01 A8



Where: Address = RTU address
IC = Immediate execution (0) or Check-back (1)
T = Execution time short (0) or long (1).
SE = Selection (0) or Execution (1).
Point = Output point on the Output card (0=1st, 1=2nd, 2=3rd… 7=8th)

Note that the first MTU request is for a selection for a check-back type with long execution time (byte2=96), immediately followed by a second MTU request for the execution for a check-back type with long execution time (97). In both requests, the RTU responds with the same message.

6.9 PERMANENT OUTPUT (PO)

The permanent output (PO) function is used when driving the mosaic light on the alarm panel.

Request:

1	1	RTU Address					
A			x	IC	x	S E	
0	Output Word Number						
Output Data4			Output Data3				
Output Data2			Output Data1				

Response: Same as the request message.

IC	SE	
0	1	= Immediate execution
1	0	=Selection in a check-back type
1	1	=Execution in a check-back type

Data1 =Least significant nibble
Data4 =Most significant nibble

6.10 CHECKSUM ALGORITHM

The last character of a message is the LCC (Longitudinal Check Character). This character is the exclusive-OR 1's complement of all of the bytes in the message. For an SB (Signal Block) message to RTU#13:

REQUEST: CD 01

The message appears below and shows the checksum character 33 calculated as follows:

	CD	1100	1101
	01	0000	0001
LCC	33	0011	0011

RESPONSE: CD 18 01

The message appears below and shows the checksum character 2B calculated as follows:

	CD	1100	1101
	18	0001	1000
	01	0000	0001
LCC	2B	0010	1011

/* End of the Pignone P6008 RTU Custom Protocol Application Notes. */

Return to Application Notes Menu

Return to the List of Manuals