GENERAC®
POWER SYSTEMS, INC.

Modbus for the DG Series Generators

Models 004311-0
004313-0

INTRODUCTION

The DG series of generators are controlled by "The F Panel." This module has both an RS232 and an RS485 communication port that can be used to share data. Both ports provide the same function, such that either port is interchangeable in terms of the data and protocol, only the physical signals are different. The RS232 port has facilities to allow the connection of a modem for remote monitoring. Depending upon the particular installation, one or both of these ports may be available for communication with the generator. Some installations may use both ports, for example, a multi—generator site with peak shaving and each generator fitted with a remote annunciator. If you have a linked multi—generator site, then data about each generator can be obtained from the separate master controllers in the same protocol. Examples of connections are described later in this manual.

The method used to communicate with all DG series controllers is the industry standard Modbus protocol. Details about this protocol may be obtained from the following website:

http://www.modicon.com/techpubs/intr7.html

Generac produces a software application known as "GENLINK", which is a PC based program designed to allow communication with individual generators and linked DG series generators via the master controller. The physical connection between GENLINK and the generators may be a direct serial link or it can be via a modem on the RS232 port.

GENLINK uses the Modbus protocol to continuously monitor and display the state of each generator and master controller. Data is displayed in the form of mimic diagrams showing the generator's key data (such as voltage, current, power, etc.). For multiple set installations, an overall mimic diagram is displayed. This overall mimic will show the power layout indicating which sections are live and the positions of the transfer switches. GENLINK can also be used to change the operating parameters of the generators and the master controller.

It is not necessary to use GENLINK as the basis for communication with the DG series. Since Modbus has been chosen as the standard protocol, any piece of equipment capable of being programmed as a Modbus master can be used for data exchange.

MODBUS PROTOCOL

Modbus controllers can be set up to communicate in one of two modes — ASCII or RTU. The DG series controllers will only work in RTU modes (where communications are done in binary — not ASCII). The speed of communication (baud rate) is programmable, but limited to be either 9600 or 4800 baud.

The DG controllers act as slaves in a master/slave configuration and depending on the installation, many slaves can be linked together to one master. In order to differentiate units from each other, each unit is assigned a unique address (even if there is only one unit). These addresses are programmed into the slave units from their front panels.

Data is stored and accessed from uniquely defined registers within the F panel. Each register contains specific data such as RPM, HZ, and VOLTS, etc. and can be accessed or changed as defined by the Modbus protocol.

GENERATOR (SLAVE) ADDRESSING

In a single generator installation, it is not necessary to program an address into the controller. This is because each controller will always respond to a "universal address", thus allowing a GENLINK user to link to any site. Security is performed not by address, but by a password. The universal address is decimal 250. The unique address will default to 1.

In a multiple generator installation, each generator must have a unique address. This can be entered from the front panel under the engine parameters menu.

The master controller must also have an address but it will default to 1 which is sufficient for most purposes. If the master controller is linked to a building network, you may need to alter it's address to avoid conflicts with other equipment connected to the same network. The master controller also needs to be programmed with both the number of generator sets connected to it and their addresses.

SETTING UP COMMUNICATION PARAMETERS

The speed of communication (baud rate), number of stop bits and parity are all programmable from the front panel of the controllers. These should be set up to match each other and the Modbus master (e.g. GENLINK).

We suggest: 4800 baud, 8 data bits, NO parity, 2 stop bits.

This is the default setting for both the "F" panel and the master controller RS485 port. The RS232 port is set as: 9600 baud, 8 data bits, NO parity, 1 stop bit.

CONNECTIONS TO THE "F" PANEL

In order to make physical connections to the ports, it is necessary to open the generator's enclosure.

⚠ DANGER ⚠

OBSERVE ALL SAFETY PRECAUTIONS AS DESCRIBED IN THE GENERATOR MANUAL. ENSURE THAT THE UTILITY VOLTAGE HAS BEEN DISCONNECTED FROM THE UNIT AND **THAT THE BATTERY AND THE UTILITY BATTERY CHARGER HAVE BEEN DISCONNECTED.** THE EMERGENCY STOP SWITCH SHOULD BE PRESSED AND THE FRONT PANEL KEY SWITCH SHOULD BE IN THE "OFF" POSITION.

Locate the transfer switch enclosure below or opposite the voltmeter and ammeter at the back of the generator. Remove the cover plate. The communications terminal strip is located on the left hand wall.

The RS485 connections are to the following wires:

Wire# 392 — RS485+

Wire# 393 — RS485 –

The RS232 connections are to the following wires:

Wire# 387 — RS232 RX

Wire# 388 — RS232 TX

Wire# 389 — RS232 COMMON

Connection should be via twisted pair shielded cable.

Cable size: 24 AWG 0-250 feet

18 AWG 0-1200 feet

CONNECTIONS TO THE MASTER CONTROLLER

In a multiple generator application, there is a master controller. These are either a peak shave module or an island mode module. There is also the option of having a building management control module. By incorporating a building management control module into the system, it now serves as the master controller in the modbus network, in line with the multiple generators. A RS232 or multi-drop RS485 communicates between the two controllers. The building management control module is a slave in this connection, but is still the master controller over the multiple generator connection. This connection allows interaction between both the master and any of the multiple generators on an independent basis.

Connection to the Power Manager Controller is done by connecting the RS 485 wires to TB2-RS485 #2 positive (+) and TB2-RS485 #2 negative (–).

STARTING AND STOPPING USING MODBUS COMMANDS

The F panel can be started and stopped using the Modbus protocol. In order to do this, the key switch must be in the "Auto" position and there must be a good utility supply (otherwise the generator will run in standby mode).

To run the generator without paralleling, the mb_start variable (Modbus register address 0026h) should be set to 0001h. This will start the generator but will not connect the generator to the utility. Setting this variable back to zero will stop the generator again.

To run the generator in parallel using the power and power factor settings on the generator front panel, first make sure that the screen_en variable (Modbus register address 0132h high byte) is set to 0001h. This can also be set from the front panel. Start the generator by setting the mb_start variable to 1 as above and then set the parallel_en variable to 1 (this is the high byte of Modbus register 002a — in other words set this register to 0100h). This will close the transfer switch once the generator has synchronized with the utility, and use the power setting from the front panel display. Note that this power setting (and the power factor setting) can also be changed using Modbus (power setting is register 012bh and power factor is 012ah). These settings are written into EEPROM, meaning that they are retained even after the generator stops. However, it is not advisable to change these values too frequently as the EEPROM has a limited number of write cycles. To disconnect, set the mb_start variable back to zero which will cause the generator to disconnect from the utility and run for the preset cooldown period before stopping.

If the screen_en variable is set to zero, the F panel will use the analog power input instead of the front panel setting. The power factor is still set from the front panel.

The generator can also be run in peak-shave mode by setting the mb_start variable to 0002h instead of 0001h. This will start the generator and automatically parallel. There is no need to set parallel_en in this mode. However, instead of using the front panel settings for power and power factor, it will use the values in ps_kw (Modbus register address 0027h) and ps_pf (Modbus register address 0028h). These values are stored in RAM which means that they are not retained if the F panel is powered down, but they can be written to more frequently than the EEPROM values.

REGISTER ADDRESSES AND THEIR DATA

Although data is accessed as 16 bit registers, some data is stored as a byte and therefore one register access will bring back 2 "lots" of data. The following table shows the locations of the available data and is correct as of 8-16-00. The locations should not change but may be added to in the future. The table is in the form of a "C" program header file so you can see which variables

are in what format (signed, unsigned, word or byte). Some variables are accessible for both read and write while others are read only variables. Generac accepts no responsibility for operation as a result of changing these variables.

Editor's Notes:

Variables with a _l extension are usually the low level flags for a warning. (Non zero flags signify a warning)

Variables with a _h extension are usually the high level flags for a warning. (Non zero flags signify a warning)

Variables with a _s extension are the shutdown flags. (Non zero flags signify a warning)

Variable declaration		Modbus address		
Alarm status variables set if alarm is present: READ ONLY, Stored in RAM				
unsigned char	oil_press_1	0000	Hi	low oil pressure warning
unsigned char	oil_press_s	0000	Lo	low oil pressure shutdown
unsigned char	coolant_temp_h	0001	Hi	high coolant temperature warning
unsigned char	coolant_temp_s	0001	Lo	high coolant temperature shutdown
unsigned char	coolant_temp_l	0002	Hi	low coolant temperature warning
unsigned char	oil_temp_h	0002	Lo	high oil temperature warning
unsigned char	oil_temp_s	0003	Hi	high oil temperature shutdown
unsigned char	batt_volts_l	0003	Lo	low battery voltage warning
unsigned char	batt_volts_h	0004	Hi	high battery voltage warning
unsigned char	rpm_s	0004	Lo	overspeed (high rpm) shutdown
unsigned char	rpm_l	0005	Hi	underspeed (low rpm) warning
unsigned char	gen_volts_h	0005	Lo	overvoltage shutdown
unsigned char	gen_volts_l	0006	Hi	undervoltage shutdown
unsigned char	freq_h	0006	Lo	over frequency shutdown
unsigned char	freq_l	0007	Hi	under frequency shutdown
unsigned char	fuel_level_h	0007	Lo	high fuel level warning
unsigned char	fuel_level_l	0008	Hi	low fuel level warning
unsigned char	fuel_level_s	0008	Lo	low fuel level shutdown
unsigned char	inter_comm_f	0009	Hi	internal error shutdown
unsigned char	fail_start_s	0009	Lo	over-crank shutdown
unsigned char	water_level_s	000a	Hi	coolant level sensor failed
unsigned char	rpm_sens_s	000a	Lo	rpm sensor failed
unsigned char	start_inhibit_n	000b	Hi	start inhibited due to oil press.
unsigned char	emstop_s	000b	Lo	emergency stop shutdown
unsigned char	oil_press_f	000c	Hi	oil pressure sensor failed
unsigned char	oil_temp_f	000c	Lo	oil temperature sensor failed
unsigned char	coolant_temp_f	000d	Hi	coolant temperature sensor failed
unsigned char	*engine_run_status	000d	Lo	
unsigned char	*engine_running	000e	Hi	shows if engine running
unsigned char	system_status	000e	Lo	

*See notes on page 5.

Variable declaration			Modbus address		
"Analog Values"		**READ ONLY, Stored in RAM**			
signed	int	oil_press	000f		
unsigned	int	rpm	0010		
unsigned	int	freq	0011		
signed	int	oil_temp	0012		
unsigned	int	fuel_level	0013		
signed	int	coolant_temp	0014		
unsigned	int	batt_volts	0015		
unsigned	int	gen_voltsA	0016		
unsigned	int	gen_voltsB	0017		
unsigned	int	gen_voltsC	0018		
unsigned	char	power	0019	Hi	
unsigned	char	power_factor	0019	Lo	
unsigned	int	load_amps	001a		
unsigned	int	util_volts	001b		
unsigned	int	util_freq	001c		
unsigned	int	gen_amps	001d		
struct Dispctrl_bit Dispctrl			001e	Hi	
char Transfer_error_no			001e	Lo	
struct RS232_bit RS232_bitstr			001f	Hi	
struct Output_bit Output			001f	Lo	
union PORTA_store_bit PORTA_store			0020	Hi	
union PORTB_store_bit PORTB_store			0020	Lo	
union PORTP_store_bit PORTP_store			0021	Hi	
union PORTT_store_bit PORTT_store			0021	Lo	
union Alarm_bitaddr Alarm			0022		
union AVR_23_bit AVR_23			0023	Hi	
union AVR_01_bit AVR_01			0023	Lo	
unsigned long hours_run			0024 + 0025		
struct mbstart_bit mb_start			0026		READ AND WRITE ACCESS
unsigned	int	ps_kw	0027		READ AND WRITE ACCESS
unsigned	int	ps_pf	0028		READ AND WRITE ACCESS
unsigned	char	reset_genlink	0029	Hi	READ AND WRITE ACCESS
unsigned	char	parallel_en	002a	Hi	READ AND WRITE ACCESS
unsigned	int	modem_disconnect	002b		READ AND WRITE ACCESS

Notes for Page 4

Note 1:

Engine_run_status

0 = Keyswitch in off position
1 = Started from manual keyswitch start
2 = Started from remote start
3 = Started from serial link (Genlink)
4 = Started from exerciser (no transfer)
5 = Started from parallel switch
6 = Started from exerciser (with transfer)
7 = Started due to utility loss
8 = Stopped with keyswitch in auto position
9 = Started from peak shave module

Note 2:

Engine_running

Bit 0 = Engine running
Bit 1 = Utility breaker status
Bit 2 = Generator breaker status

Note 3:

This variable can have the following values:

Reset = 0	Running Up = 6	Cool Down = 11
Stopped = 1	Warming Up = 7	Stopping = 12
Preheating = 2	Load Accept = 8	Alarm Stopping = 13
Starting = 3	Warming Active = 9	Alarm Stopped = 14
Starting PAUSE (between cranks) = 5		Alarms Active = 10

Editors notes:

The following locations are setpoints in EEprom and are the stored parameters which are not lost when power is removed. (Settings such as cooldown time etc) The parameters have both **READ AND WRITE ACCESS**.

The following naming conventions are used.
Variables with a _lsp extension are low setpoints
Variables with a _hsp extension are high setpoints
Variables with a _ssp extension are shutdown setpoints

You should not write to EEPROM MEMORY locations on a continual basis as EEPROM MEMORY has a limited number of write cycles before it "wears out".

eeprom unsigned long hours_run_ep[6]	**NOT ACCESSIBLE FROM MODBUS**	
eeprom unsigned int start_attempts	0100	
eeprom unsigned int pause_time	0101	
eeprom unsigned int cool_time	0102	
eeprom unsigned int warm_time	0103	
eeprom unsigned int hold_off_time	0104	
eeprom unsigned int mbus_spare1	0105	
eeprom unsigned int mbus_spare2	0106	
eeprom unsigned int start_time	0107	
eeprom unsigned int rpm_started	0108	
eeprom unsigned int preheat_time	0109	
eeprom unsigned int fuel_level_lsp	010a	
eeprom unsigned int fuel_level_hsp	010b	
eeprom unsigned int gen_volts_lsp	010c	
eeprom unsigned int gen_volts_hsp	010d	
eeprom unsigned int rpm_lsp	010e	
eeprom unsigned int rpm_ssp	010f	
eeprom unsigned int batt_volts_hsp	0110	
eeprom unsigned int batt_volts_lsp	0111	
eeprom signed int oil_temp_ssp	0112	
eeprom signed int oil_temp_hsp	0113	
eeprom signed int coolant_temp_ssp	0114	
eeprom signed int coolant_temp_hsp	0115	
eeprom signed int coolant_temp_lsp	0116	
eeprom signed int oil_press_ssp	0117	
eeprom signed int oil_press_lsp	0118	
eeprom unsigned int flywheel_teeth	0119	
eeprom unsigned char userpwd[8]	011a	
eeprom unsigned char panel_id[8]	011e	
eeprom unsigned char avr_droop_gain	0122	Hi
eeprom unsigned char avr_droop_select	0122	Lo
eeprom unsigned char avr_stability	0123	Hi
eeprom unsigned char avr_gain	0123	Lo
eeprom unsigned char avr_sensing	0124	Hi
eeprom unsigned char avr_freq	0124	Lo
eeprom int gov_gain	0125	
eeprom int gov_diff	0126	
eeprom int avr_uf_slope	0127	
eeprom int avr_uf_corner	0128	
eeprom int avr_vsetpt	0129	
eeprom int avr_pf	012a	
eeprom int avr_kw	012b	
eeprom unsigned char modem_sel	012c	Hi
eeprom unsigned char gov_stability	012c	Lo
eeprom unsigned char preheat_en	012d	Hi
eeprom unsigned char transfer_en	012d	Lo
#ifdef CT_RATIO_CODE_REQUIRED		
@eeprom char ct_ratio	012e	Hi
#else		
@eeprom char spare_param	012e	Hi
#endif		

eeprom char avr_spare4	012e Lo	
eeprom unsigned int freq_hsp	012f	
eeprom unsigned int freq_lsp	0130	
eeprom unsigned int fuel_level_ssp	0131	
eeprom unsigned char screen_en	0132 Hi	
eeprom unsigned char fuel_level_en	0132 Lo	
eeprom int gvscale	0133	
eeprom int avr_vscale	0134	
eeprom unsigned int log_start	0135 - alarm log start	
eeprom unsigned int log_end	015d - alarm log end (alarm log is read only)	
eeprom char modbus_id	slave address	NOT ACCESSIBLE
eeprom char RS232_baud	RS232 baud rate	NOT ACCESSIBLE
eeprom char RS232_mode	RS232 comms mode	NOT ACCESSIBLE
eeprom char RS485_baud	RS485 baud rate	NOT ACCESSIBLE
eeprom char RS485_mode	RS485 comms mode	NOT ACCESSIBLE
eeprom unsigned int eeprom_checksum		NOT ACCESSIBLE

EXAMPLES

To get registers 0x000 to 0x000e the request string (in decimal) would be:

50	address=50, say
3	function (fetch registers)
0	Start address high byte
0	Start address low byte
0	Number of registers to get high byte
15	Number of registers to get low byte
crc	low
crc	high

The response will be:

50	address
3	function
30	byte count
?	data
?	"
?	"
…	"
crc	low
crc	high

To write to registers the string would be:

50	address
16	function (load registers)
0	start address, high byte
0	start address, low byte
0	number of registers to write to, high byte
15	number of registers to write to, low byte
30	byte count
xx	data
xx	data
..	data
crc	low
crc	high

GENERAC® POWER SYSTEMS, INC.
P.O. BOX 8 • WAUKESHA, WI 53187
PH: (414) 544-4811 • FAX: (414) 544-4851

Part No. D2901 **Revision 0 (02/06/01)** **Printed in U.S.A.**