xantrex™

Smart choice for power

Modbus Serial Communications Protocol Specification

TITLE: Modbus Serial Communications Protocol		DOC NO. 503-0068-01-01	REV. E
PREPARED BY: Karl Isham	DATE: 12/13/2007	xantrex Smart choice for power	SHEET: Page 1 of 104

Revision History

Rev.	Date	Description of Change	Author
A	26 June 2006	First complete release	K. Isham
B	3 Oct 2006	• Corrected address range for direct slaves in section 5.1.1 • Added missing aggregates in Gateway specific map and added integrated HMI registers for wireless remote panel • Corrected XW specific map to make explicit battery input and output status registers • 5.2 Data Record Queues was changed to state that auto incrementing will only occur on a read of the index register • 3.4 and 5.5 now state that the same read request should be retried until it is completed • 5.6 now has additional restrictions on the number on the type of registers that may be aliased • 7.2.10 Energy History Map was changed to add months to the available log types, the ordering of elements in the queue was also made explicit • Added HMI Configuration map • Added appendix with state enumerations • Correct register address of Software Part Number • Added SCP device specific map • Added reprogramming to all Gateway proxied devices • First public release.	K. Isham
C	29 Mar 2007	• Update and add a few register definitions, such as system operation state, etc. • External Errata document describes all changes.	Y. Duan, J. Altstadt
D	12 Dec 2007	• General updates as per external errata document.	Y. Duan J. Fieldhouse R. Shuttleworth J. Altstadt
E	13 Dec 2007	• Section 7 Modbus Register Map – Added special note to implementers.	J. Altstadt

Xantrex Technology Inc.
8999 Nelson Way
Burnaby BC
Canada V5A 4B5

Table of Contents

1. Introduction

1.1 Purpose

This document explains the implementation of the Modbus communications protocol specific to Xantrex products. The Modbus protocol is an industry standard that allows a master station to interogate devices for its publically available data and set supported control and configuration paramaters.

This document assumes the reader is already familiar with the Modbus protocol and serial communication. The reader is directed to the documents listed in section 2 for general protocol specifications.

1.2 Scope

This protocol specification applies to selected renewable energy products offered by Xantrex. The applicability of individual registers verys by product. Please refer to the appendix for a list of registers and features supported by each product.

2. Related documents

Modicon PI-MBUS-300	Modbus Reference Guide, Rev.J
Modbus.org	Modbus over Serial Line Specification and Implementation Guide, V1.0
Modbus.org	Modbus Application Protocol Specification, V1.1a
Modbus.org	Modbus Messaging on TCP/IP Implementation Guide, V1.0a

3. Modbus Configuration

3.1 Supported Transmission Medium

3.1.1 RS-485

Products supporting Modbus communicate primarily via the RS-485 (TIA/EIA-485-A) communications standard. The RS-485 medium allows multiple devices to communicate on a single medium using a master/slave approach for arbitrating the bus.

All Xantrex products are 1/8 unit load (UL) devices. Depending on the mix of other devices from other vendors, biasing or polarization of devices on the network, this will allow for as few as 32 devices or as many as 255 devices on a single bus.

Please refer to individual product specifications for details on polarization and termination.

3.1.2 RS-232

Products may alternately support Modbus communications via the RS-232 (EIA-232-C) communications standard. The RS-232 medium is point-to-point and will only allow for one master and one slave. All other aspects of the Modbus serial protocol remain intact, including addressing.

3.1.3 TCP/IP

Products may alternately support the transmission of serial Modbus by encapsulation in TCP/IP packets (IETF RFC 793). All other aspects of the Modbus serial protocol remain intact, including addressing. Modbus slaves units operating in this capacity should all be in the same multicast group, such that queries from the master reach all "attached" slaves similar to a multi-drop bus configuration. Each slave then accepts/rejects the packet based on Modbus addressing, just as it would over a serial multi-drop medium.

The Xantrex WiPort/Gateway supports a limited subset of the Modbus over TCP/IP protocol as outlined in the following points:

- Transaction and Protocol ID field values are echoed back to the requesting IP address
- Only a single IP address (the Gateway/WiPort's address) is provided for all proxied Modbus devices. That is the Gateway and all devices accessible through it share the same IP address though they have unique unit IDs
- Only Modbus function-codes 0x03, Read Multiple Registers and 0x10, Write Multiple Registers are supported
- Only one Modbus packet is allowed per TCP/IP payload
- Each Modbus over TCP/IP payload must be 256 bytes or less
- Fragmenting of a Modbus message over multiple TCP/IP packets is not supported.
- Proxied devices may take up to five seconds to respond to a request

As a quick reference, the transporting TCP/IP payload appears as follows:

Transaction ID [2 bytes]	Protocol ID [2 bytes]	Length in bytes [2 bytes]	Modbus Unit ID [1 byte]	Modbus function code [1 byte]	Modbus payload [n bytes]

All fields are formatted in big-endian and the following notes apply:

- The transaction ID and protocol ID are treated as mentioned above
- The Length field is the length of the Modbus packet in bytes, which includes the unit ID, function code and payload length
- The Modbus CRC is not transmitted

3.2 Supported Modes of Transmission

The Modbus protocol supports two serial transmission modes, ASCII and RTU (Remote Terminal Unit). Xantrex only supports the RTU mode of operation.

3.3 Communications Parameters

Serial communications default to 8 data bits, no parity, and 1 stop bit. A default baud rate of 9600 BPS is used. These defaults were chosen to be compatible with other products common to renewable energy. Parameters are reconfigurable (see Common Configuration Map in section 7.4.1).

3.4 Exception Responses

Exceptions may be generated in response to commands from the Modbus master to signify reasons why a request packet cannot be honored. The table below describes the exception codes supported by Xantrex devices along with their possible causes.

Code	Name	Meaning
01	Illegal Function	An illegal function code is contained in the function field of the request packet. Xantrex devices only support functions 3 and 16.
02	Illegal Address	The address referenced in the data field of the request packet is invalid for the specified function.
03	Illegal Value	The value referenced in the data field of the request packet is not allowed for the referenced register.
06	Device Busy	The device is engaged in processing a long duration command. The master should retransmit the same request until it completes.

Table 1 - Exception Codes Supported by Xantrex Devices

3.5 Broadcasts

All Xantrex products support broadcast request packets from the master. As its name implies, broadcasts allow all devices to receive and process the same command from the Modbus master. Broadcasts are only valid with Function 16 (see section 4.3) and are triggered by setting the slave address to zero (0). All slaves will receive and execute the request, but will not respond.

4. Supported Modbus Functions

4.1 Function 3 (03h): Read Holding Registers

This function code is used to read the contents of one or more holding registers on the selected slave.

4.1.1 Query

The query message specifies the starting register and the number of registers to read.

Figure 1 - Function 03 Query Message Format

Slave Addr	Function Code **03**	Starting Register High	Starting Register Low	# of Register High	# of Register Low	CRC Low	CRC High

For example, read the clock maintained by the device, which has a unit ID of 5. The clock is located at address 8010_h and is in Uint32 format. Two registers will be requested because of the format:

Master	05 03 80 10 00 02 [CRC] [CRC]

4.1.2 Response

The response message contains the data read from registers on the slave. The registers occur in order from the first register requested through each sequential register that follows for the number of registers requested.

Figure 2 - Function 03 Response Message Format

Slave Addr	Function Code **03**	Byte Count	Register Data High	Register Data Low	. . .	CRC Low	CRC High

For example, if the UTC time of the device was 1130455700 (43616294_h - 27 Oct 2005 23:28:20 GMT), the response from the request made in section above would return as:

Slave	05 03 04 43 61 62 94 [CRC] [CRC]

4.2 Function 8 (08h): Diagnostics

This function code is used to test the communication link between the master and slave. It consists of a number of sub-functions that specify the type of test to be performed by the slave. Issuing a diagnostic command to a device will not effect its normal operation. Broadcasts are not supported with this command.

All Xantrex devices will, at a minimum, support sub-function code **00**, **Return Query Data**. Refer to product specific documentation for support of any other sub-functions.

4.2.1 Query

The query message is a request to loop back the provided data.

Figure 3 - Function 08 Query Message Format

Slave Addr	Function Code **08**	Function High **00**	Function Low **00**	Data High	Data Low	CRC Low	CRC High

For example, request that unit ID of 5 return back the data $A537_h$:

Master	05 08 00 00 A5 37 [CRC] [CRC]

4.2.2 Response

The response message loops back the same data as the request. The function code and the sub-function code are echoed, making query and response exactly the same.

Figure 4 - Function 08 Response Message Format

Slave Addr	Function Code **08**	Function High **00**	Function Low **00**	Data High	Data Low	CRC Low	CRC High

For example, given the request in the previous section, unit ID 5 would simply return the same data it was given:

Slave	05 08 00 00 A5 37 [CRC] [CRC]

4.3 Function 16 (10h): Preset Multiple Registers

This function gives a Modbus master control over the device. Control can be exercised either as direct manipulation over the device's functions or through the setting of configuration parameters.

4.3.1 Query

The query message specifies the register contents for a sequence of registers. This function may use the broadcast slave address (00) to preset the same values into all attached slaves.

Figure 5 - Function 16 (10h) Query Message Format

Slave Addr	Func Code **10**	Start Reg High	Start Reg Low	# of Reg High	# of Reg Low	Byte Count	Reg Data High	Reg Data Low	. . .	CRC Low	CRC High

For example, set the current UTC time of a device at unit ID 5. Assuming the current time was 27 Oct 2005 23:28:20 GMT, the UTC seconds would be 1130455700 (43616294_h). The clock is located at address 8010_h and is in Uint32 format. Two registers will be written:

Master	05 10 80 10 00 02 04 43 61 62 94 [CRC] [CRC]

4.3.2 Response

The response message returns the starting register and the number of registers that were set from data in the query.

Figure 6 - Function 16 (10h) Response Message Format

Slave Addr	Function Code **10**	Starting Register High	Starting Register Low	# of Register High	# of Register Low	CRC Low	CRC High

For example, the response from the request above to set the UTC register with the current time would include the register address 8010_h and an indication that 2 registers were written, since the format of this register is a Uint32. The complete response would be:

Master	05 10 80 10 00 02 [CRC] [CRC]

5. Special Functions

5.1 Slave Address Assignment

5.1.1 Direct Slaves

On first power-up the default slave address is set to 201. During manufacturing this default value is changed to the last two digits of the unit's serial number, for serial numbers ending in 01 through 100. If a serial number ends in 00, then its ID will be set to 100 to avoid the broadcast address. This default addressing scheme allows for the device to be placed in most configurations without modification. Should address conflicts occur, or should the installer wish to manually harmonize the IDs of all the units on the bus, the unit ID may be changed by writing the desired ID to the Modbus Unit ID register in the Common Configuration Map (Section 7.4.1).

Care should be exercised when changing any ID if a proxy is part of the system, since it automatically assigns addresses within a stated range (see section 5.1.2). The safest choice is to avoid explicitly assigning an address within this range.

5.1.2 Proxied Slaves

On devices that act as a proxy for a network of devices, the slave addresses are automatically harmonized and assigned by the proxy device. The constituents represented by the proxy and their assigned addresses may be discovered by reading the Device List from the unit (see section 7.2.2). Addresses assigned by the proxy will always be between 101 and 200 inclusive. The default address of the proxy itself will always be 201.

The addresses assigned to a device are granted by lease. If the device is removed from the network and later reinstalled, it will be assigned it's previously held unit ID. New devices will be given a previously unused address within the range noted above. If the supply of previously unused addresses becomes exhausted, then the oldest previously occupied address will be used.

The unit id of the proxy may be changed by writing the desired ID to the Modbus Unit ID register in the Common Configuration Map (Section 7.4.1). The unit id of any represented device may not be changed.

5.2 Data Record Queues

Collections of data sets are maintained in queues. Each data set, or record, in the collection is exposed one at a time for access via a group of registers that represent the members of the set.

An index register is used to access other records in the collection. The index may be explicitly set to access a specific record. Subsequent reads of the index will auto increment the index, so successive records can be retrieved simply by making additional read requests of the entire register set. Note that the index is incremented *after* the read takes place.

A register is also supplied with each record queue which indicates the total number of records currently in the queue. If read requests continue beyond the number of records in the queue, then the index will automatically wrap around.

A record queue can be likened to the programming concept of an array. The total number of records is equivalent to the range of the array, and the index register serves the same function as an array index. The registers comprising the members of the record are the structure that comprised the elements of the array. Figure 7 - Record Queue Array Metaphor, demonstrates this analogy.

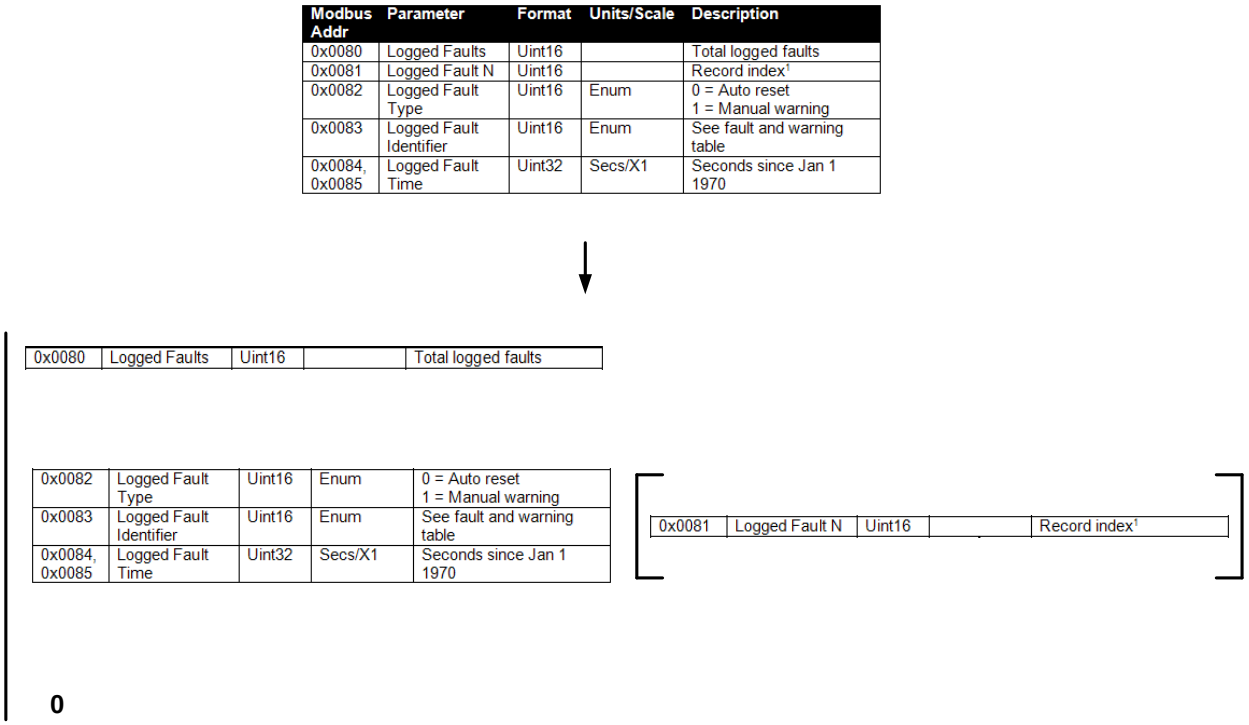


Figure 7 - Record Queue Array Metaphor

To use such a queue the master should follow this sequence:

1. Read the register that indicates the total number of records in the queue. This will establish the valid upper bound for the index and allow the master to calculate how many reads are required to retrieve all records.

2. Write to the index register to set the first desired record to read. Zero accesses the first record unless otherwise specified.
3. Issue a read request for the multiple registers comprising the record, starting at the index register in the record for the number of registers in the set.
4. Issue subsequent identically formed read requests to retrieve the next record in the queue until all records are records are retrieved.

Note that this sequence of operations must be followed each time a queue is accessed following any other Modbus register access. Due to the transitory nature of the data contained in queues, all queue actions must be pseudo-atomic to maintain data integrity. A queue cannot be set up, partially read, followed by an access to any register outside that queue, and then followed by a continuation of the queue read sequence. The read continuation will return an Illegal Value exception.

If random access to the queue interspersed with other register accesses is desired, then the queue must be reinitialized by reading the length of the queue and writing the desired index number. Note that the contents of the entire queue may have changed or possibly been re-ordered (e.g. new faults raised or cleared) between successive initializations, so this is not a recommended operating procedure.

It is recommended that all reads of queues include the index value as a cross check that the expected queue record was actually the one which was read.

This sequence is diagrammed below.

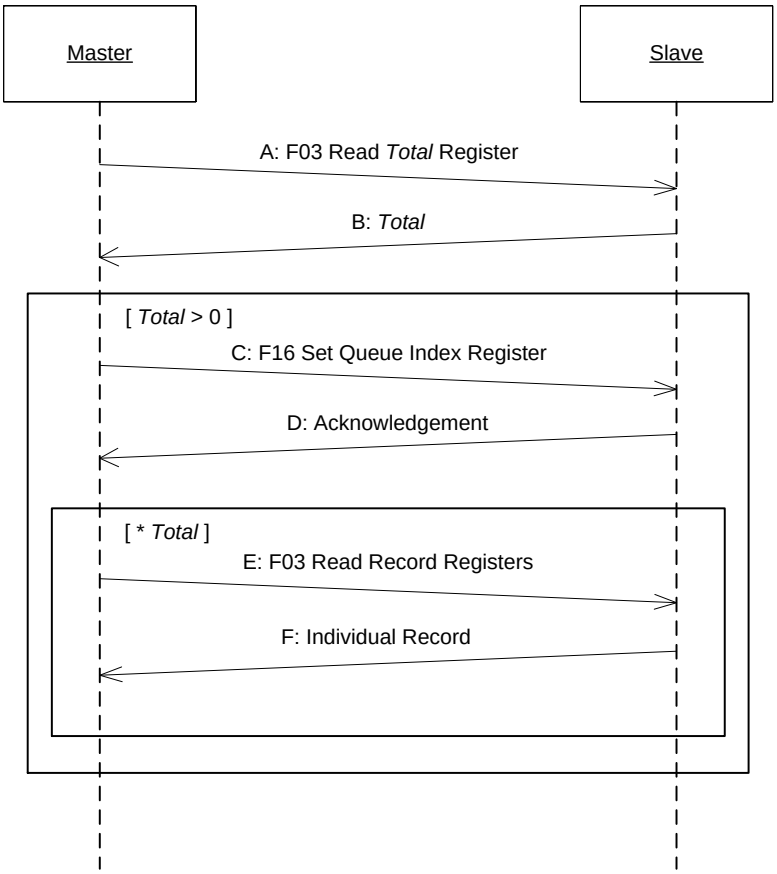


Figure 8 - Reading Contents of a Record Queue

For example consider the historical fault log (see Common Read-only Status Map in section 7.2.1. Actual register numbers may vary.). The register map is shown below:

Modbus Addr	Parameter	Format	Units/Scale	Description
0x0080	Logged Faults	Uint16		Total logged faults
0x0081	Logged Fault N	Uint16		Record index[1]
0x0082	Logged Fault Type	Uint16	Enum	0 = Auto reset 1 = Manual warning
0x0083	Logged Fault Identifier	Uint16	Enum	See fault and warning table
0x0084, 0x0085	Logged Fault Time	Uint32	Secs/X1	Seconds since Jan 1 1970

Assume unit ID 5 has logged three faults. To read the entire fault log:

(1) Read the **Logged Faults** register.

Master	05 03 **00 80** 00 01 [CRC] [CRC]
Slave	05 03 02 ***00 03*** [CRC] [CRC]

Note that ***3*** is returned as the total number of records available in the queue.

(2) Set the **Logged Faults N** register to the first index, ***0***.

Master	05 10 **00 81** 00 01 02 **00 00** [CRC] [CRC]
Slave	05 10 00 81 00 01 [CRC] [CRC]

(3) Read the first fault record. The record starts at **0x0081$_h$** and spans **5** registers.

Master	05 03 **00 81 00 05** [CRC] [CRC]
Slave	05 03 0A 00 00 [TYP_h TYP_l] [ID_h ID_l] [TM_h TM TM TM_l] [CRC] [CRC]

(4) Read the second fault record.

Master	05 03 **00 81 00 05** [CRC] [CRC]
Slave	05 03 0A 00 01 [TYP_h TYP_l] [ID_h ID_l] [TM_h TM TM TM_l] [CRC] [CRC]

(5) Read the third fault record.

Master	05 03 **00 81 00 05** [CRC] [CRC]
Slave	05 03 0A 00 02 [TYP_h TYP_l] [ID_h ID_l] [TM_h TM TM TM_l] [CRC] [CRC]

5.3 Device Calibration

Specific devices may support calibration using Modbus. Calibration is calculated internal to the device. Modbus is only used to sequence the unit and lay in the values measured by external equipment. The device compares the given measured values with those it measured internally to calculate gain and offset. Gain and offset corrections are stored internally in non-volatile memory.

Calibrating generally follows this sequence:

1. Set the Calibration Selection register to the device specific enumeration representing the value to calibrate
2. In conjunction with an external bench setup, issue other Modbus commands, as required, to place the unit in an operating state necessary to create the low setpoint.

3. Set the Measured Value Low register with the measured value obtained though the test setup. The units and scale of the register are device and value specific.
4. In conjunction with an external bench setup, issue other Modbus commands, as required, to place the unit in an operating state necessary to create the high setpoint.
5. Set the Measured Value High register with the measured value obtained though the test setup. The units and scale of the register are device and value specific.
6. Set the Calibration Selection register to 0 to signal and end of the calibration sequence, calculate the gain and offset, and store it in non-volatile memory.

This sequence is diagrammed below.

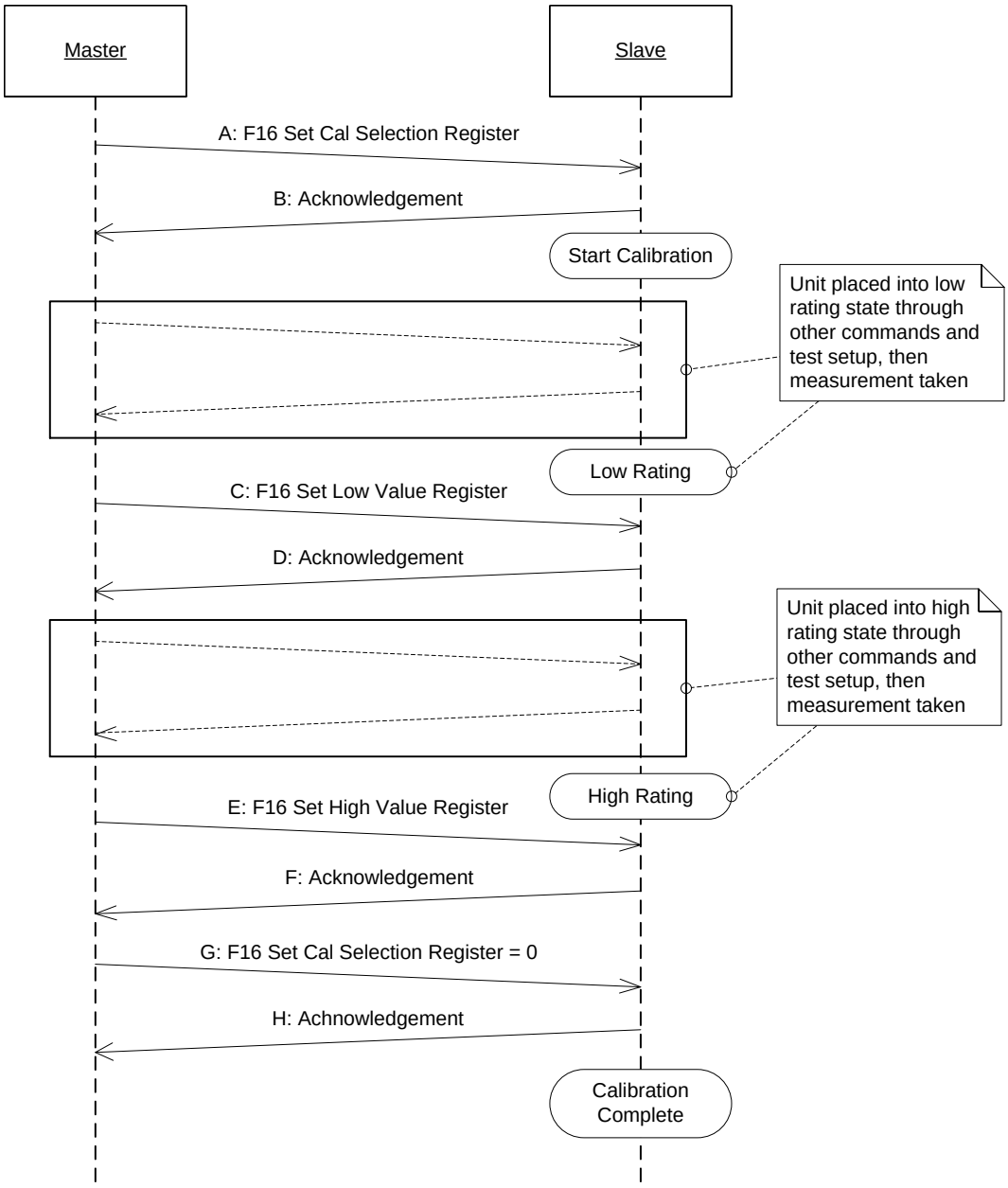


Figure 9 Device Calibration

Refer to the Calibration Control Map in section 7.3.2 for the details concerning the calibration registers. Calibration selection values, exact set-up procedures for each selection and setpoint, and data formats for measured values for each selection are device specific. Refer to device documentation for specific details.

5.4 Device Reprogramming

Specific devices may support field reprogramming using Modbus. Reprogramming is requires resetting the unit to hand control over to a boot loader, which implements a limited register map Modbus for the reprogramming sequence. The new program is downloaded to the device in blocks. Once reprogramming is complete, the device is reset again to hand off control to the new program. This is diagrammed below.

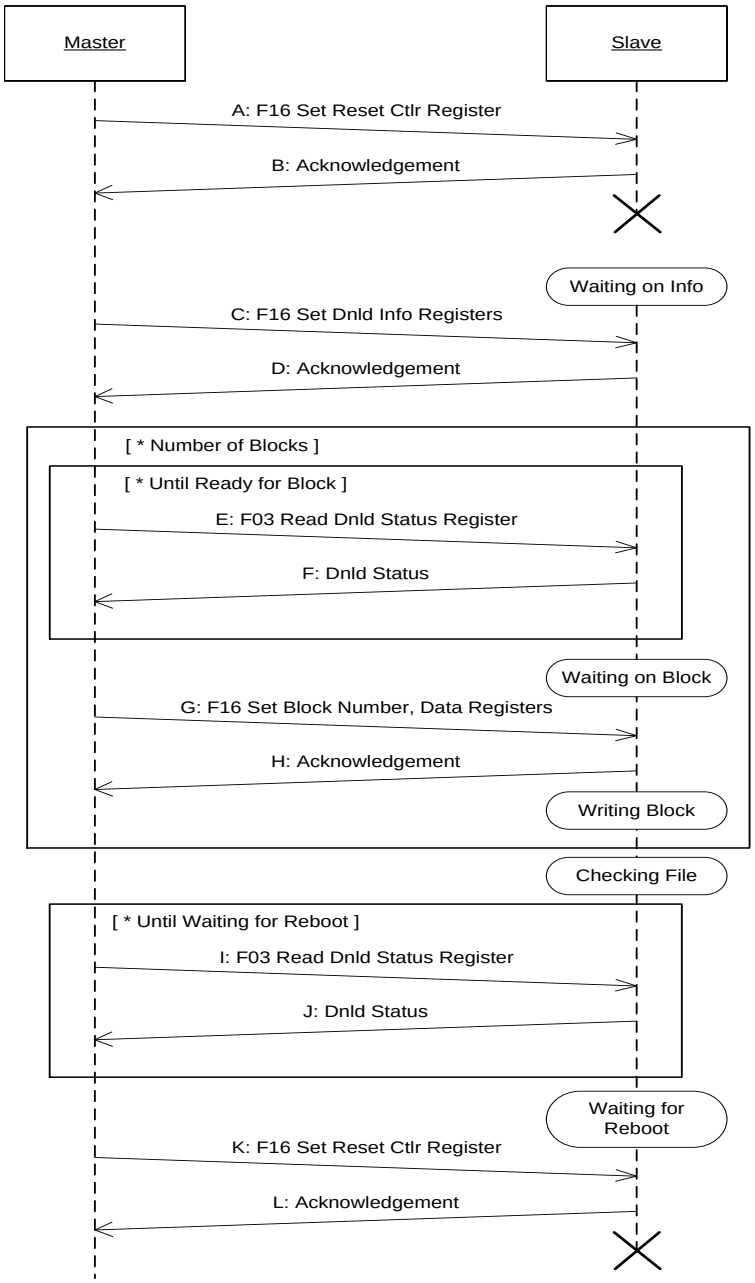


Figure 10 Device Reprogramming

5.5 Network Proxy

Certain devices may act proxies for other devices interconnected on another networking medium. These proxies are designed to represent the devices on the other network as if they were simply additional Modbus slaves. The proxy does this by responding to multiple slave addresses. Virtual slave addresses are assigned by the proxy. The list of devices and their assigned addresses may be discovered by querying specific registers on the proxy (see 7.2.2 for details on the Device List register map).

Responses from a virtual Modbus device may be delayed due to the time it takes to propagate requests to the actual device and refresh locally cached registers on the proxy. When such is the case the proxy device will return an exception code **06** in response to a request addressed to the virtual device (see section 3.4). If this exception is encountered, the master should continue to retry the same request until it completes. The master shouldn't go on to another request while the Gateway is trying to retrieve the data for the previous request. A complete transaction is considered to be all the traffic between the initial request, any number of request retries with 06 responses, and the final response with the requested registers.

The register data for the proxied device will be invalidated in the event the device is removed from its network.

5.6 Register Aliasing

Devices may support register aliasing, where a register can be duplicated at another address. Once linked the register may be accessed at either address. This may be useful for optimizing reads or writes by picking scattered parameters of interest and grouping them together in contiguous registers.

The mechanism for establishing the duplicates is a straightforward association of the fixed with the aliased address. The Register Alias Configuration Map in section 7.4.3 provides the means. It is a record queue (see section 5.2), except the total number of records can be added simply by writing past the end of the queue.

For example, let's say we want to access the following two registers with the same read:

Modbus Addr	Parameter	Format	Units/Scale	Description
0x0201, 0x0202	DC Input Voltage	Uint32	VDC/X100	
0x0701,	AC Output	Uint32	Vrms/X100	

0x0702	Voltage			

In their current locations they are not adjacent. Two read requests will be necessary to read the registers. What we can do is use the aliasing feature to place both registers adjacent to one and other in another part of the map, such as a device specific region:

Modbus Addr	Parameter	Format	Units/Scale	Description
Reserved for device specific registers				
0x4000-0x7FFF	Device Specific	--	--	--

To do this the master can use the following configuration registers (actual register numbers may vary, refer to the Register Alias Configuration Map in section 7.4.3 for actual addresses):

Modbus Addr	Parameter	Format	Units/Scale	Description
0x80B0	Register Aliases	Uint16		Total number of aliases
0x80B1	Alias N	Uint16		Record index
0x80B2	Fixed Register Address	Uint16		Register address from existing map
0x80B3	Alias Register Address	Uint16		Desired secondary address

1. Read the register that indicates the total number of records in the queue.
2. Write this total number, plus one, to the index register (write one beyond what it currently holds).
3. Issue a write request for two registers starting at the **Fixed Address Register**. The first register is written with the present address of the register. The second register is written with the desired secondary address of the register
4. Issue subsequent write requests for all register/alias pairs you wish to map.

To complete this example (and assuming the unit ID is 5), the programming sequence would proceed as follows:

(1) Read the **Register Aliases** register.

Master	05 03 **80 B0** 00 01 [CRC] [CRC]
Slave	05 03 02 ***00 03*** [CRC] [CRC]

Note that ***3*** is returned as the total number of records available in the queue (assumed for example only).

(2) Set the **Alias N** register to this value, ***3***. This will add a 4th element to the record queue by starting to write at index 3.

Master	05 10 **80 B1** 00 01 02 **00 03** [CRC] [CRC]
Slave	05 10 80 B1 00 01 [CRC] [CRC]

(3) Set the first alias, 0x0201 to 0x4000 by writing this register pair to 0x80B2

Master	05 10 **80 B2** 00 02 04 **02 01 40 00** [CRC] [CRC]
Slave	05 10 80 B2 00 02 [CRC] [CRC]

(4) Set the second alias, 0x0202 to 0x4001 by writing this register pair to 0x80B2

Master	05 10 **80 B2** 00 02 04 **02 02 40 01** [CRC] [CRC]
Slave	05 10 80 B2 00 02 [CRC] [CRC]

(5) Set the third alias, 0x0701 to 0x4002 by writing this register pair to 0x80B2

Master	05 10 **80 B2** 00 02 04 **07 01 40 02** [CRC] [CRC]
Slave	05 10 80 B2 00 02 [CRC] [CRC]

(6) Set the fourth alias, 0x0702 to 0x4003 by writing this register pair to 0x80B2

Master	05 10 **80 B2** 00 02 04 **07 02 40 03** [CRC] [CRC]
Slave	05 10 80 B2 00 02 [CRC] [CRC]

Now DC Input Voltage will be accessible at 0x0201 and 0x4000, and the AC Output Voltage will be accessible at 0x0701 and 0x4003. These register could now be read together, since they are adjacent in their aliased locations.

A current alias can be changed by setting the index register to a specific alias and overwriting. There is no way to actually delete a single alias from the queue. Pseudo deletes can be accomplished changing an alias to an unused address. Otherwise, the master should use the Clear Log register (section 7.3.1) to erase the entire alias list and reprogram the desired alias again.

Note also that for parameters that span multiple registers, each register address of the parameter must aliased, as in the example above. Failure to do so will result in unpredictable results. Record queues should not be aliased as they will not have the ability to auto increment (see section 5.2). A maximum of 200 registers may be aliased. Only read-status registers and read-write command registers may be aliased.

6. Modbus Registers

6.1 Register Format

6.1.1 Reserved Numeric Values

The extreme positive values for each numeric format are reserved. These are useful as padding during multiple register writes, where a register is embedded in a group of registers and needs to be left untouched. On reads, these values take on special meaning as defined below.

6.1.1.1 Data Not Available

Data not available shall be represented by the highest possible value of the data format. For example, for a 16-bit unsigned integer the value is 65,535 ($FFFF_h$). For a 16-bit signed integer, the value is 32,767 ($7FFF_h$). This value read from a register indicates that the device does not have the particular value. Writing this value to a register will have no effect on its contents.

6.1.1.2 Out of Range

Out of range shall be represented by the second highest possible value of the data format. For example, for a 16-bit unsigned integer the value is 65,534 ($FFFE_h$). For a 16-bit signed integer, the value is 32,766 ($7FFE_h$). This value, read from a register, indicates an error condition such as an out of range value. Writing this value to a register will have no effect on its contents.

6.1.1.3 Reserved

The third highest value of a given data format shall be reserved. For example, for a 16-bit unsigned integer the value is 65,533 ($FFFD_h$). For a 16-bit signed integer, the value is 32,765 ($7FFD_h$).This value shall not be used. However, writing this value to a register will have no effect on its contents.

6.1.2 Packed Boolean Format - Bool

Boolean, or two state values, are packed 16 per register. Individual bits are referenced in the address map according to their bit number within the register, B0 through B15.

Figure 11 - Packed Boolean Format

Register N
B15 B14 B13 B12 B11 B10 B9 B8 B7 B6 B5 B4 B3 B2 B1 B0

6.1.3 Packed 8-bit Integer Format – Uint8

8-bit values are packed two to a register, separated into upper and lower byte fields. This format is used to represent octet strings, or to pack two unsigned integer values together that need to be atomically set together. For example, a reset command where the processor to reset is specified in one byte and the reset type is specified in the other.

For paired unsigned integer values, each value is represented in the register map as occupying the high or low byte position of the register.

Figure 12 - Packed 8-bit Integer Format

Register N	
High Byte	Low Byte
B7 . . . . . B0	B7 . . . . . B0

If the format is used for string data, then a fixed range of contiguous registers are specified along with an indication of how many bytes are contained in the string. Data is arranged such that the start of the string begins in the high byte of register N and the last byte of the string occupies the low byte of register N + [(x – 1) / 2], where x is the number of bytes in the string. Strings are nul terminated, and are nul padded to fill the remaining byte(s) in the register(s). String sizes exceeding the total payload length of a Modbus PDU are not permitted.

All read operations on strings must read the entire defined string length, starting from the first register of the string, or an error is returned.

All write operations on strings must start from the first register of the string or an error is returned. The string must be nul terminated and nul padded if the terminating nul is at the start of a new register. It is not required to write the defined string length.

Figure 13 - String Format

Register N		Register N+1		Register …		Register N+[(x-1)/2]	
Char	Char	Char	Char	Char	Char	Char	Char
C_1	C_2	C_3	C_4	C…	C…	C_{x-1}	C_x

6.1.4 16-bit Integer Format – Uint16 and Sint16

16-bit integer values are contained in a single register. If the value is unsigned then the range of valid values is 0 to 65532. If the value is signed then the range is -32768 to +32764 in two's complement. Note that in each case the maximum value is limited by required reserved values (see section 6.1.1).

Figure 14 - 16-bit Integer Format

Register N
B15 . . . . . B8 B7 . . . . . B0

The signed and unsigned versions of this format are referred to in the register map as Sint16 and Uint16, respectively.

Byte order in the register is Big Endian.

6.1.5 32-bit Integer Format – Uint32 and Sint32

To accommodate values larger than that reached with a 16-bit number, a 32-bit format is provided that spans two registers. In signed and unsigned 32 bit integer formats the 32-bit value is split between two consecutive 16-bit registers. The first register (at address *N*) is the high-order word, and the second register (at address *N + 1*) is the low-order word:

$$\text{Value} = (\text{register}_N * 65535) + \text{register}_{N+1}$$

All read and write operations on 32-bit integers must access both defined registers in an atomic action, or an error is returned.

Figure 15 - 32-bit Integer Format

Register N	Register N+1
B31 . . . B23 B22 . . . B16	B15 . . . B8 B7 . . . B0

If the value is unsigned then the range of valid values is 0 to 4,294,967,292. If the value is signed then the range is -2,147,483,648 to +2,147,483,644 in two's compliment. Note that in each case the maximum value is limited by reserved values (see section 6.1.1).

The signed and unsigned versions of this format are referred to in the register map as Sint32 and Uint32, respectively.

The overall byte order for both registers is Big Endian.

6.1.6 Enumerated Value Format

A list of options is represented by a numeric relationship. For example battery type is enumerated as:

0 = Flooded
1 = Gel
2 = AGM
3 = Custom
4 = Deep Cycle
5 = Optima

Enumerations can be contained within unsigned packed 8-bit or unsigned 16-bit formats. The meaning of each enumerated value is captured in the register map or in individual device documentation.

6.2 Numeric Units and Scale

6.2.1 Units

Scalar values are generally useless without some indication of the units of measure. The units for each register are noted in the register map. Where the units are left blank, the associated value is unit-less, such as a count or enumeration.

6.2.2 Scale

Numeric values are scaled to represent real numbers in an integer format. This fixed-point representation fixed for each register. For example, a battery voltage of 12.4 VDC would be represented with an integer value of 1240 using X100 scaling. The scaling of each register is noted in the register map. Where the scale is left blank, X1 scaling is assumed or the value is scale-less, such as a count or enumeration.

6.3 Register Types

As a simplification, Xantrex only employs the Modbus Holding Registers. These registers have been subdivided into categories descriptive of the kind of operation that is being demanded of the device. By definition all Holding Registers are read-write. Depending on the operation, the device may restrict access to certain registers to read-only.

6.3.1 Read-only Status Registers

Status operations are intended to solicit data from the device that reports on dynamic measurements, accumulated and calculated data, or operational state. Attempts to write to these registers will return a **02** exception, illegal address. Exceptions are made in the case of registers that control what is displayed in other registers. An example of this is the Index register used to select an individual record in a queue (see section 5.2).

6.3.2 Read-write Control Registers

Control registers allow dynamic control over the operation of the device, such as enabling and disabling run-time mode, causing changes in state, or commanding the device to perform an operation.

6.3.3 Read-write Configuration Registers

Configuration registers allow the features of the device to be customized for the individual installation.

6.3.4 Invalid Registers

In the register map, there are gaps between some registers. Moreover, not all registers defined in the register map may be used on device, depending on its function.

Invalid registers store no information. Since multiple register functions may contain a mix of valid and invalid registers, a read or write request containing an invalid register will ***not*** be rejected with an exception. When an invalid register is read it will return the Data Not Available value (see section 6.1.1.1). Writes to an invalid registers are ignored.

7. Modbus Register Map

The following details the standard registers support on Xantrex devices. Common registers should be accessible on all devices. Function specific registers are only accessible on devices supporting the indicated functionality. A chart showing the applicability of all registers by product is shown in the appendices.

An individual product may support other device specific registers for configuration and diagnostics. Product specific documentation will detail these registers.

Special note to implementers: The register definitions in this section should be considered as the equivalent of C language `typedef struct` definitions, while the

definitions in the Appendixes are the equivalent of the corresponding variable declarations.

7.1 Loader Specific Registers

The following status registers are common to all Xantrex devices that support field reprogramming through Modbus and are only available when the device's boot loader is active. Refer to section 5.4 for the details on device reprogramming.

7.1.1 Loader Reset Control Map

The following registers are collocated at the same address as the non-loader control registers of the same function.

Modbus Addr	Parameter	Format	Units/Scale	Description
Reset Command				
0xF000 L	Reset Controller	Uint8		Controller instance 0 to 15
0xF000 H	Reset Type	Uint8	Enum	0 = Reboot (no others allowed in loader mode)

7.1.2 Loader Read-only Status Map

Modbus Addr	Parameter	Format	Units/Scale	Description
0xFF70	Loader State	Uint16	Enum	0 = Waiting on info 1= Waiting on block 2 = Writing block 3 = Checking file 4 = Waiting for reboot
0xFF71	Download Result	Uint16	Enum	0 = No error 1 = Invalid info 2 = invalid block 3 = block write error 4 = file CRC error

7.1.3 Loader Read-write Control Map

Modbus Addr	Parameter	Format	Units/Scale	Description
Download Information Record				
0xFF72	Dnld Controller	Uint16		Controller instance 0 to 15
0xFF73, 0xFF74	File Size	Uint32		Number of bytes in downloaded file
0xFF75	CRC	Uint16		CCITT 16-bit CRC
0xFF76	Total Blocks	Uint16		Total number of blocks to expect
File Download Block				
0xFF77	Block Number	Uint16		The block number of the data in the Block Data registers
0xFF78	Block Size	Uint16		The number of bytes in the block (maximum of 242)
0xFF79 – 0xFFF1	Block Data	Uint8 X Block Size		Content of the block
0xFFF2 – 0xFFFF	Reserved for expansion			

7.2 Read-Only Status Registers

7.2.1 Common Status Map

The following status registers are common to all Xantrex devices.

Modbus Addr	Parameter	Format	Units/Scale	Description
Product Info				
0x0000 – 0x0009	Product Model Designation	Uint8 x 20		"C" style null terminated ASCII string
0x000A – 0x0013	Finished Goods Assembly (FGA) Number	Uint8 x 20		"C" style null terminated ASCII string
0x0014 – 0x001D	Serial Number	Uint8 x 20		"C" style null terminated ASCII string

Modbus Addr	Parameter	Format	Units/Scale	Description
0x001E – 0x0027	Software Part Number	Uint8 X 20		"C" style null terminated ASCII string. Software revision is always the last two dash separated tuples in the string
0x0028 – 0x007E	Reserved for repeat of registers above for additional included products represented by a single Modbus slave.			
Active Fault and Warning Change				
0x007F	Change Counter	Uint16		XB system Information, only apply to Gateway. This counter will increase 1 when there is any xb devices have active fault or warning change. Loop back to 0 after reaches 0xFFFE. 0xFFFF reserved for not valid.
Active Faults Record Queue (see section 5.2)				
0x0080	Active Faults	Uint16		Current active faults
0x0081	Active Fault N	Uint16		Record index[1]
0x0082	Active Fault Type	Uint16	Enum	0 = Auto reset escalating 1 = Auto reset 2 = Manual fault
0x0083	Active Fault Identifier	Uint16	Enum	See fault and warning table
0x0084, 0x0085	Active Fault Time	Uint32	Secs/X1	Seconds since Jan 1 1970
0x0086 – 0x0099	Active Fault String	Uint8 X 40		"C" style null terminated ASCII string
0x009A - 0x009F	*Reserved*			
Active Warning Record Queue (see section 5.2)				
0x00A0	Active Warnings	Uint16		Current active warnings
0x00A1	Active Warn N	Uint16		Record index[1]
0x00A2	Active Warn Type	Uint16	Enum	0 = Auto reset 1 = Manual warning
0x00A3	Active Warn Identifier	Uint16	Enum	See fault and warning table
0x00A4, 0x00A5	Active Warn Time	Uint32	Secs/X1	Seconds since Jan 1 1970
0x00A6 – 0x00B9	Active Warn String	Uint8 X 40		"C" style null terminated ASCII string
0x00BA - 0x00CE	*Reserved*			

Modbus Addr	Parameter	Format	Units/Scale	Description
Present Device State				
0x00CF	Device State	Unit16	enum	See state definition table in appendix A
0x00D0	System State	Uint16	Bit Field 0: energy flow off 1: energy flow on	0x0001 = Grid to AC load 0x0002 = Gen to AC load 0x0004 = Battery to Gen 0x0008 = Battery to Grid 0x0010 = Grid to Battery 0x0020 = Gen to Battery 0x0040 = PV to Battery 0x0080 = PV to Grid 0x0100 = Battery to AC load
0x00E0 – 0x00FF	Reserved for expansion or repeat of registers above			

Notes:
[1]The contents of this register specifies an index into a set of records. Writing to this register will set the record number to retrieve for a read on the registers that follow. On subsequent reads of any register in the set, the index will auto increment to the next available value.

7.2.2 Device List Status Map

The following status registers are present on devices that act as a proxy for a network of devices.

Modbus Addr	Parameter	Format	Units/Scale	Description
Device List Record Queue (see section 5.2)				
0x0100	Proxied Devices	Uint16		Total number of proxied devices
0x0101	Proxied Device N	Uint16		Record index[1]
0x0102	Device Type ID	Uint16	enum	0 = Do not Care 1 = Gateway reserved 2 = XW 3 = GT 4 = MPPT 5 = AGS 6 = SCP1(Wired Remote) 7 = SCP2

Modbus Addr	Parameter	Format	Units/Scale	Description
0x0103	Modbus Virtual Unit ID[2]	Uint16		Slave address 100 – 199 assigned for virtual device access.
0x0104 – 0x010D	Device Name	Uint8 X 20		"C" style null terminated ASCII string
0x010E – 0x010F	Reserved for expansion			

Notes:
[1]The contents of this register specifies an index into a set of records. Writing to this register will set the record number to retrieve for a read on the registers that follow. On subsequent reads of any register in the set, the index will auto increment to the next available value.

[2]The unit ID assigned by the proxy cannot be changed with the virtual devices' Modbus Unit ID register (section 7.4.1). Any attempt to write to that register will be ignored.

7.2.3 Connection Map

The following status registers are present on devices that act as a proxy for a network of devices. The map represents an identification of individual relationships that connect or link entities together in the system. For an example, a DC output may be associated with a battery. If devices are nodes then connections can be considered vertices. Such a view can only be formulated by a proxy device, due to its global view of the supported network.

Modbus Addr	Parameter	Format	Units/Scale	Description
Connection Record Queue (see section 5.2)				
0x0110	Connections	Uint16		Total number of connections
0x0111	Connection N	Uint16		Record index[1]
0x0112	Connection ID	Uint16	enum	Unique net number that links two or more devices 0x1XXX : AC Connections 0x2XXX : DC Connections Refer Appendix B
0x0113 – 0x011C	Connection Name	Uint8 X 20		"C" style nul terminated ASCII string

Modbus Addr	Parameter	Format	Units/Scale	Description
0x011D – 0x011F	Reserved for expansion			

Notes:
[1]The contents of this register specifies an index into a set of records. Writing to this register will set the record number to retrieve for a read on the registers that follow. On subsequent reads of any register in the set, the index will auto increment to the next available value

7.2.4 History Device List Map

The following registers are present on devices that have energy history recording for retreval.

Modbus Addr	Parameter	Format	Units/Scale	Description
Device List Record Queue (see section 5.2)				
0x0180	Total Devices	Uint16		Total number of devices
0x0181	Device N	Uint16		Record index[1]
0x0182	Device Type ID	Uint16	enum	1 = Gateway, system 2 = XW 3 = GT 4 = MPPT
0x0183	Device Serial ID	Uint16		Part of Product Serial Number
0x0184 – 0x018F	Reserved for expansion			

Notes:
[1]The contents of this register specifies an index into a set of records. Writing to this register will set the record number to retrieve for a read on the registers that follow. On subsequent reads of any register in the set, the index will auto increment to the next available value.

7.2.5 DC Input Status Map

The following status registers are present on devices which have DC inputs.

Modbus Addr	Parameter	Format	Units/Scale	Description
DC Input *Note: each input mapped to separate set of registers*				
0x0200	Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)
0x0201, 0x0202	DC Input Voltage	Uint32	VDC/X100	
0x0203, 0x0204	DC Input Current	Uint32	ADC/X100	
0x0205, 0x0206	DC Input Power	Uint32	W/X1	
0x0207 – 0x020F	*Reserved for expansion*			
0x0210 – 0x02FF	*Reserved for repeat of registers above for additional inputs*			

7.2.6 DC Source Status Map

The following status registers are present on all devices that monitor a replenishable DC energy source.

Modbus Addr	Parameter	Format	Units/Scale	Description
DC Source *Note: each monitored source mapped to separate set of registers*				
0x0300	Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)
0x0301	DC Source Temperature	Sint16	C/X10	If temperature is monitored
0x0302	DC Source State of Charge	Uint16	%/X1	If source is rechargeable
0x0303	DC Source State of Health	Uint16	%/X1	If source is a battery
0x0304	DC Source Time Remaining	Uint16	Min/X1	If source has a lifetime

Modbus Addr	Parameter	Format	Units/Scale	Description
0x0305	DC Source Capacity Remaining	Uint16	AHr/X1	If source is a battery
0x0306 – 0x030F	*Reserved for expansion*			
0x0310 – 0x03FF	*Reserved for repeat of registers above for additional sources*			

7.2.7 DC Output Status Map

The following status registers are present on devices which have DC outputs.

Modbus Addr	Parameter	Format	Units/Scale	Description
DC Output *Note: each output mapped to separate set of registers*				
0x0400	Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)
0x0401, 0x0402	DC Output Voltage	Uint32	VDC/X100	
0x0403, 0x0404	DC Output Current	Uint32	ADC/X100	
0x0405, 0x0406	DC Output Power	Uint32	W/X1	
0x0407	DC Output % of Maximum	Uint16	%/X1	
0x0408 - 0x040F	*Reserved for expansion*			
0x0410 – 0x04FF	*Reserved for repeat of registers above for additional outputs*			

7.2.8 AC Input Status Map

The following status registers are present on all devices which have an AC input.

Modbus Addr	Parameter	Format	Units/Scale	Description
AC Input *Note: each AC input mapped to separate set of registers*				
0x0500	Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)
0x0501, 0x0502	AC Input Voltage	Uint32	Vrms/X100	
0x0503, 0x0504	AC Input Current	Uint32	Arms/X100	
0x0505	AC Input Frequency	Uint16	Hz/X10	
0x0506, 0x0507	AC Input Real Power	Uint32	Watts/X1	
0x0508, 0x0509	AC Input Reactive Power	Uint32	Vars/X1	
0x050A, 0x050B	AC Input Apparent Power	Uint32	VAs/X1	
0x050C	AC Input Power Factor	Sint16	X100	
0x050D - 0x050F	*Reserved for expansion*			
0x0510 - 0x05FF	*Repeat of registers above for additional lines or sources*			

7.2.9 AC Source Status Map

The following status registers are present on all devices that monitor an AC energy source.

Modbus Addr	Parameter	Format	Units/Scale	Description
AC Source *Note: each AC source mapped to separate set of registers*				
0x0600	Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)

Modbus Addr	Parameter	Format	Units/Scale	Description
0x0601	AC Level Qualification	Uint16	Enum	0 = Not Qualifying 1 = Qualifying 2 = Missing 3 = Too Low 4 = Too High 5 = Good
0x0602	AC Freq Qualification	Uint16	Enum	0 = Not Qualifying 1 = Qualifying 2 = Missing 3 = Too Low 4 = Too High 5 = Good
0x0603, 0x0604	Elapsed AC Qualified Time	Uint32	Sec/X1	Total number of seconds that source has been qualified
0x0605 – 0x060F	*Reserved for expansion*			
0x0610 – 0x06FF	*Reserved for repeat of registers above for additional lines or sources*			

7.2.10 AC Output Status Map

The following status registers are present on all devices which supply an AC output.

Modbus Addr	Parameter	Format	Units/Scale	Description
AC Output *Note: each AC output mapped to separate set of registers*				
0x0700	Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)
0x0701, 0x0702	AC Output Voltage	Uint32	Vrms/X100	
0x0703, 0x0704	AC Output Current	Uint32	Arms/X100	
0x0705	AC Output Frequency	Uint16	Hz/X10	
0x0706, 0x0707	AC Output Real Power	Uint32	Watts/X1	

Modbus Addr	Parameter	Format	Units/Scale	Description
0x0708, 0x0709	AC Output Reactive Power	Uint32	Vars/X1	
0x070A, 0x070B	AC Output Apparent Power	Uint32	VAs/X1	
0x070C	AC Output Power Factor	Sint16	X100	
0x070E - 0x070F	*Reserved for expansion*			
0x0710 - 0x07FE	*Reserved for repeat of registers above for outputs*			

7.2.11 Energy History Status Map

The following status registers are present on all devices which supply a history of energy production.

Note that there are additional configurable settings which affect the energy logs. The Start Day of Week and Start Day of Month registers described in section 7.4.1 are used to align the history to a user specified day in each case. This has the effect of chunking data on on-standard boundaries to comply with local customs or billing periods as appropriate to the needs of the end user.

The history logs use 00:00 Local Time (midnight) to determine the start of day to create logs of days or longer periods; i.e. any power generated after 00:00 Local Time will be allocated to the following day.

Modbus Addr	Parameter	Format	Units/Scale	Description
Energy Log Record Queue (see section 5.2) *Note: Each AC line, source, or output mapped to separate set of registers*				
0x07FE	Device ID	Uint16		Only gateway accept this register to provide removed device history data Has to use together with 0x07FF~0x0802
0x07FF	Device TYPE	Uint16		Only gateway accept this register to provide removed device history data Has to use together with 0x7FE, 0x0800~0x0802

Modbus Addr	Parameter	Format	Units/Scale	Description
0x0800 – 0x0802	Energy Log Type[2]	Uint16	enum	0 = previous hour 1 = previous day 2 = previous week 3 = previous month 4 = today 5 = lifetime 6 = previous 15 min (system only) 7 = previous 5 min
	Timebase[5]	Uint32	Secs/X1	The first record in the queue contains the energy generated at this time. Seconds since Jan 1 1970 UTC. If 0, the current time is used.
0x0803	Previous Energy Log N	Uint16		Record index [1] Total number of records set by Log Type [2]: 0: 24 [3,4] 1: 31 [4] 2: 5 [4] 3: 12 [4] 4: 1 5: 1 6: 16 7: 12
0x0804 – 0x0805	Energy	Sint32	KWhr/X10	
0x0806 – 0x0807	Peak Power	Sint32	W/X1	
0x0808 – 0x0809	Harvest Time	Uint32	Secs/X1	Time unit has been harvesting energy, if connected to a PV array
0x080A-0x080D	*Reserved*			
0x080E – 0x08FD	*Repeat of registers above for additional lines, sources, or outputs*			

Notes:
[1]The contents of this register specifies an index into a set of records. Writing to this register will set the record number to retrieve for a read on the registers that

follow. On subsequent reads of any register in the set, the index will auto increment to the next available value.
[2]The total number of records in the log is a fixed value determined by the Log Type.
[3]In the event of a device reset, the device will persistently maintain the total cumulative energy for the day so as not to cause a discontinuity in external energy monitoring software.
[4] The queue is ordered from the most recent (index 0) to the least recent (index *n*).
[5] The timebase has to be provided together with the type.

7.2.12 Internal Sensor Status Map

The following status registers are present on devices where internal sensors are monitored.

Modbus Addr	Parameter	Format	Units/Scale	Description
Internal Temperature Status *Note: Each sensor mapped to separate set of registers*				
0x0900	Temperature	Sint16	C/X10	
0x0901 – 0x090F	*Reserved for expansion*			
0x0910 – 0x09FF	*Reserved for repeat of registers above for additional sensors*			

7.2.13 Auxiliary Output Triggers Status Map

The following status registers are present on all devices which support auxiliary output triggers.

Modbus Addr	Parameter	Format	Units/Scale	Description
Aux Output Triggers *Note: Each auxiliary output trigger mapped to separate set of registers*				
0x0A00	Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)

Modbus Addr	Parameter	Format	Units/Scale	Description
0x0A01	Trigger State	Uint16	enum	1 = Auto On 2 = Auto Off 3 = Manual On 4 = Manual Off
0x0A02 L	Trigger On Reason	Uint8	enum	0 = Not On 1 = Manual On 2 = Battery Volts Low 3 = Battery Volts High 4 = Array Volts High 5 = Battery Temp Low 6 = Battery Temp High 7 = Heat Sink Temp High 8 = Fault
0x0A02 H	Trigger Off Reason	Uint8	enum	0 = Not Off 1 = Manual Off 2 = No Active Trigger 3 = Trigger Override 4 = Fault
0x0A03, 0x0A04	Trigger Output Voltage	Uint32	VDC/X100	
0x0A05, 0x0A06	Trigger Output Current	Uint32	ADC/X100	
0x0E07 - 0x0E0F	*Reserved for expansion*			
0x0A10 – 0x0AFF	*Repeat of registers above for additional aux trigger outputs*			

7.2.14 Automatic Generator Status Map

The following status registers are present on all devices which support an automatic generator start function.

Modbus Addr	Parameter	Format	Units/Scale	Description
Automatic Generator Start *Note: Each genset mapped to separate set of registers*				

Modbus Addr	Parameter	Format	Units/Scale	Description
0x0B00	Generator Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)
0x0B01	DC Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)
0x0B02	Generator Operation State	Uint16	Enum	0 = Quiet Time 1 = Auto On 2 = Auto Off 3 = Manual On 4 = Manual Off 5 = Gen Shutdown 6 = Ext Shutdown 7 = AGS Fault 8 = Suspend 9 = Not Operating
0x0B03	Generator Status	Uint16	Enum	0 = Preheating 1 = Start Delay 2 = Cranking 3 = Starter Cooling 4 = Warming Up 5 = Cooling Down 6 = Spinning Down 7 = Shutdown Bypass 8 = Stopping 9 = Running 10 = Stopped 11 = Crank Delay
0x0B04 L	Generator On Reason	Uint8	enum	0 = Not On 1 = DC Voltage Low 2 = Battery SOC Low 3 = AC Current High 4 = Contact Closed 5 = Manual On 6 = Exercise 7 = Non Quiet Time 8 = Ext On via AGS 9 = Ext On via Gen 10 = Unable To Stop 11 = AC Power High 12 = DC Current High

Modbus Addr	Parameter	Format	Units/Scale	Description
0x0B04 H	Generator Off Reason	Uint8	enum	0 = Not Off 1 = DC Voltage High 2 = Battery SOC High 3 = AC Current Low 4 = Contact Opened 5 = Reached Absorp 6 = Reached Float 7 = Manual Off 8 = Max Run Time 9 = Max Auto Cycle 10 = Exercise Done 11 = Quiet Time 12 = Ext Off via AGS 13 = Safe Mode 14 = Ext Off via Gen 15 = Ext Shutdown 16 = Auto Off 17 = Fault 18 = Unable To Start 19 = Power Low 20 = DC Current Low 21 = AC Good
0x0B05	Generator Runtime	Uint16	Hrs/X10	Total run time since the AGS was commissioned and since the last Clear Command
0x0B06 - 0x0B0F	*Reserved*			
0x0B10 – 0x0BFF	*Repeat of registers above for additional gensets*			

7.2.15 Nominal Ratings Status Map

This section defines the registers which describe the nominal ratings of all devices. These values are taken from the device specifications.

Modbus Addr	Parameter	Format	Units/Scale	Description
Device Power Rate				

Modbus Addr	Parameter	Format	Units/Scale	Description
0x0C00, 0x0C01	Power Rating	Uint32	Watts	
Temperature Range				
0x0C02	Temperature1 Min	Sint16	C/X10	
0x0C03	Temperature1 Max	Sint16	C/X10	
0x0C04 – 0x0CFF	Reserved for expansion or repeat of registers above			
DC Voltage and current Range				
0x0D00, 0x0D01	DC voltage Min	Uint32	VDC/X100	
0x0D02, 0x0D03	DC voltage Max	Uint32	VDC/X100	
0x0D04, 0x0D05	DC current Min	Uint32	IDC/X100	
0x0D06, 0x0D07	DC current Max	Uint32	IDC/X100	
0x0D08 – 0x0DFF	Reserved for expansion or repeat of registers above			
AC Voltage and current Range				
0x0E00, 0x0E01	AC voltage Min	Uint32	VAC/X100	
0x0E02, 0x0E03	AC voltage Max	Uint32	VAC/X100	
0x0E04, 0x0E05	AC current Min	Uint32	IAC/X100	
0x0E06, 0x0E07	AC current Max	Uint32	IAC/X100	
0x0E08	AC Freq Min	Uint16	Hz/X10	
0x0E09	AC Freq Max	Uint16	Hz/X10	
0x0E0A – 0x0EFF	Reserved for expansion or repeat of registers above			

7.2.16 Built In Self Test Result Status Map

This section defines the registers reserved for record of built in self test (BIST) result.

Modbus Addr	Parameter	Format	Units/Scale	Description
Self test result				
0x0F00	Test Result	Uint16	bitmap	0x0000 = OK 0x0001 = GPIO err 0x0002 = Extern RAM err 0x0004 = Extern Flash err 0x0008 = NVMEM err 0x0010 = RTC err
0x0F01-0x0FFF	Reserved for standard data set expansion.			

7.2.17 Fault/Warning/Event Logging Status Map

The following status registers are common to all Xantrex devices for fault, warning and event history logging.

Modbus Addr	Parameter	Format	Units/Scale	Description
Fault Log Record Queue (see section 5.2)				
0x1000	Logged Faults	Uint16		Total logged faults
0x1001	Logged Fault N	Uint16		Record index[1]
0x1002	Logged Fault Type	Uint16	Enum	0 = Auto reset 1 = Manual warning
0x1003	Logged Fault Identifier	Uint16	Enum	See fault and warning table
0x1004, 0x1005	Logged Fault Time	Uint32	Secs/X1	Seconds since Jan 1 1970
0x1006 – 0x1019	Logged Fault String	Uint8 X 40		“C” style null terminated ASCII string
0x101A – 0x101F	Reserved for repeat of registers above for additional included products represented by a single Modbus slave.			
Warnimg Log Record Queue (see section 5.2)				
0x1020	Logged Warnings	Uint16		Total logged warnings
0x1021	Logged Warn N	Uint16		Record index[1]
0x1022	Logged Warn Type	Uint16	Enum	0 = Auto reset 1 = Manual warning
0x1023	Logged Warn Identifier	Uint16	Enum	See fault and warning table

Modbus Addr	Parameter	Format	Units/Scale	Description
0x1024, 0x1025	Logged Warn Time	Uint32	Secs/X1	Seconds since Jan 1 1970
0x1026 – 0x1039	Logged Warn String	Uint8 X 40		“C” style null terminated ASCII string
0x103A – 0x103F	Reserved for repeat of registers above for additional included products represented by a single Modbus slave.			
State Log Record Queue (see section 5.2)				
0x1040	Logged States	Uint16		Total logged state
0x1041	Logged State N	Uint16		Record index[1]
0x1042	Logged Operating State	Uint16	Enum	See state definition table
0x1043, 0x1044	Logged State Time	Uint32	Secs/X1	Seconds since Jan 1 1970
0x1045 – 0x1058	Logged State String	Uint8 X 40		“C” style null terminated ASCII string
0x1059 – 0x10FF	Reserved for repeat of registers above for additional included products represented by a single Modbus slave.			

7.2.18 Software Version Status Map

The following status registers are common to all Xantrex devices for software version logging.

Modbus Addr	Parameter	Format	Units/Scale	Description
Software Versions				
0x1100 ~ 0x1109	Loader version String	Uint8 X20		“C” style null terminated ASCII string
0x110A – 0x1113	Application software version	Uint8 X 20		“C” style null terminated ASCII string
0x1114 – 0x11FF	Reserved for repeat of registers above for additional software version string			
0x1200 – 0x17FF	Reserved for repeat of registers above for additional included products represented by a single Modbus slave.			

7.2.19 Reserved Status Map Register Blocks

This section defines the registers reserved for the addition of standard status data sets.

Modbus Addr	Parameter	Format	Units/Scale	Description
Reserved for expansion				
0x1800-0x3FFF	Reserved for standard data set expansion.			

7.2.20 Device Specific Status Map

This section defines the registers reserved for device specific mapping of status data registers. Refer to the register map in the appendix or device documentation for details.

Modbus Addr	Parameter	Format	Units/Scale	Description
Device Specific				
0x4000 - 0x7FFF	Device specific registers			

7.3 Read-write Control Registers

7.3.1 Common Control Map

The following control registers are common to all Xantrex devices.

Modbus Addr	Parameter	Format	Units/Scale	Description
Reset Command				
0xF000L	Reset Controller	Uint8		Controller instance 0 to 15
0xF000H	Reset Type	Uint8	Enum	0 = Reboot 1 = Download 2 = Factory 3 = OEM 4 = Clear NV Memory 5 = Energy totals
System Control Command				
0xF001	System Control Command	Uint16	Enum	0 = Hibernate 1 = Power Save 2 = Safe 3 = Operating 4 = Diagnostic 252 = Last Mode

Modbus Addr	Parameter	Format	Units/Scale	Description
Clear Command				
0xF002	Clear Log	Uint16	Bit Field	0x01 = Fault Log 0x02 = Active Faults 0x04 = Warning Log 0x08 = Active Warnings 0x10 = State/Event Log 0x20 = Comm Stat 0x40 = Statistics 0x80 = Register Aliases 0xFF = All of the above
0xF003	Clear Specific Fault ID	Uint16	Enum	See fault and warning table
0xF004	Clear Specific Warning ID	Uint16	Enum	See fault and warning table
0xF005	Button pushed	Uint16	Enum	0x0000 = No action 0x0001 = Button Pushed
0xF006 – 0xF00F	Reserved for expansion			

7.3.2 Calibration Control Map

The following control registers are defined on devices that support their calibration through Modbus.

Modbus Addr	Parameter	Format	Units/Scale	Description
Calibration Command – see section 5.3				
0xF010	Calibration Selection	Uint16	Enum	0 = End calibration; perform the calibration and store calculated values N = Device specific selection of value to calibrate. Writing to this register starts calibration
0xF011, 0xF012	Measured Low Value	-int32		Device and value specific format, scale and units
0xF013, 0xF014	Measured High Value	-int32		Device and value specific format, scale and units

Modbus Addr	Parameter	Format	Units/Scale	Description
0xF015 – 0xF01F	Reserved for expansion			

7.3.3 Charger Control Map

The following configuration registers are to be used on Xantrex devices which feature a charger.

Modbus Addr	Parameter	Format	Units/Scale	Description
Charger Control Commands Note: Each charger mapped to a separate set of registers				
0xF100	Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)
0xF101	Charger Enable	Uint16	Enum	0 = Disabled 1 = Enabled
0xF102	Equalize Activate	Uint16	Enum	0 = Disabled 1 = Activate
0xF103	Force to Charge State	Uint16	Enum	1 = bulk 2 = float 3 = no float
0xF104 – 0xF10F	*Reserved for expansion*			
0xF110 – 0xF1FF	*Reserved for repeat of registers above for additional charger outputs*			

7.3.4 Inverter Control Map

The following configuration registers are to be used on Xantrex devices which feature an inverter.

Modbus Addr	Parameter	Format	Units/Scale	Description
Inverter Control Commands Note: Each inverter mapped to a separate set of registers				

Modbus Addr	Parameter	Format	Units/Scale	Description
0xF200	Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)
0xF201	Inverter Enable	Uint16	Enum	0 = Disabled 1 = Enabled
0xF202	Search Mode Enable	Uint16	Enum	0 = Disabled 1 = Enabled
0xF203	Grid-Tie Enable	Uint16	Enum	0 = Disabled 1 = Enabled
0xF204	Sell Enable	Uint16	Enum	0 = Disabled 1 = Enabled
0xF205	Force Sell Enable	Uint16	Enum	0 = Disabled 1 = Enabled
0xF206 – 0xF20F	*Reserved for expansion*			
0xF210 – 0xF2FF	*Reserved for repeat of registers above for additional charger outputs*			

7.3.5 Automatic Generator Start Control Map

The following configuration registers are to be used on Xantrex devices with generator control.

Modbus Addr	Parameter	Format	Units/Scale	Description
Auto Gen Start Control Commands Note: Each genset mapped to a separate set of registers				
0xF300	Generator Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)
0xF301	DC Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)
0xF302	Generator Mode	Uint16	Enum	0 = Manual Off 1 = Manual On 2 = Automatic

Modbus Addr	Parameter	Format	Units/Scale	Description
0xF303	Clear Runtime	Uint16		
0xF304 – 0xF30F	*Reserved for expansion*			
0xF310 – 0xF3FF	*Repeat of registers above for additional generator outputs*			

7.3.6 Maximum Power Point Tracking Control Map

Modbus Addr	Parameter	Format	Units/Scale	Description
Maximum Power Point Tracking Control Commands Note: Each MPP tracker is mapped to a separate set of registers				
0xF400	Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)
0xF401	MPPT Enable	Uint16	Enum	0 = Disabled 1 = Enabled
0xF402 - 0xF40F	*Reserved for expansion*			
0xF410 – 0xF43F	*Repeat of registers above for additional MPP trackers*			

7.3.7 Reserved Standard Control Map Register Blocks

This section defines the registers reserved for the addition of standard control data sets.

Modbus Addr	Parameter	Format	Units/Scale	Description
Reserved for expansion				
0xF440 - 0xF7FF	Reserved for standard data set expansion.			

7.3.8 Device Specific Control Map

This section defines the registers reserved for device specific mapping of control registers. Refer to the register map in the appendix or device documentation for details.

Modbus Addr	Parameter	Format	Units/Scale	Description
Device Specific				
0xF800-0xFF6F	Device Specific Registers			

7.4 Read-write Configuration Registers

7.4.1 Common Configuration Map

The following configuration registers are common to all Xantrex devices.

Modbus Addr	Parameter	Format	Units/Scale	Description
Protocol Configuration				
0x8000	Protocol[1]	Uint16	Enum	0 = Reserved 1 = Modbus 2 = Factory CLI
0x8001, 0x8002	Modbus Turn Around Delay[1]	Uint32	mS/X1	0 – 1000mS, if exceeded, an 06 (Busy) exception will be issued
0x8003	Modbus Unit ID[1]	Uint16		Slave address 1 - 247
0x8004	Baud Rate[1]	Uint16	Enum	0 = 1200 1 = 2400 2 = 4800 3 = 9600 4 = 14400 5 = 19200 6 = 28800 7 = 38400 8 = 50000 9 = 57600 10 = 76800 11 = 100000 12 = 115200 13 = 230400 14 = 460800 15 = 921600

Modbus Addr	Parameter	Format	Units/Scale	Description
0x8005	Parity[1]	Uint16	Enum	0 = None 1 = Even 2 = Odd
0x8006	Data Bits[1]	Uint16		# of data bits, 7 or 8
0x8007	Stop Bits[1]	Uint16		# of stop bits, 1 or 2
0x8008	Bus Polarization[1]	Uint16		0 = Disabled 1 = Enabled
0x8009	Bus Termination[1]	Uint16		0 = Disabled 1 = Enabled
0x800A – 0x800F	Reserved for expansion			
Time and Date				
0x8010, 0x8011	UTC Seconds	Uint32	Secs/X1	Seconds since Jan 1 1970
0x8012	Local Offset	Sint16	Mins/X1	Local time = UTC + Offset, if device displays local time
0x8013	Start Day of Month	Uint16		1~31 Note, if there is no such day in a month , the last day of that month will be used
0x8014	Start Day of Week	Uint16	enum	1 = Sunday 2 = Monday 3 = Tuesday 4 = Wednesday 5 = Thursday 6 = Friday 7 = Saturday
0x8015 – 0x801F	Reserved for expansion			

Notes:
[1]Cannot be set on a proxied device. Proxy is only point of access on Modbus for all constituent devices. Slave ID of proxied device is automatically set, and can be discovered via the Device List (see section 7.2.2).

7.4.2 Personalization Configuration Map

The following configuration registers are present on Xantrex devices that support the storage of personalization settings. These have no effect on the operation of the unit.

They only act as a persistent electronic label for the convenience and function of the master.

Modbus Addr	Parameter	Format	Units/Scale	Description
Personalization				
0x8020	Localization	Uint16	Enum	0 = English 1 = German 2 = French 3 = Spanish 4 = Italian 5 = Korean
0x8021 – 0x802A	Device Alias[1]	Uint8 x 20		“C” style null terminated ASCII string
0x802B – 0x803E	Device Memo 1	Uint8 x 40		“C” style null terminated ASCII string
0x803F – 0x8052	Device Memo 2	Uint8 x 40		“C” style null terminated ASCII string
0x8053 – 0x80AF	Reserved for expansion			

[1]On proxied devices, this name may be set by the standards present on the proxied network. Changing the alias may or may not persist on these devices. Refer to the specific register map or product documentation for the device.

7.4.3 Register Alias Configuration Map

The following configuration registers are present on Xantrex devices that support register aliasing (see section 5.6)

Modbus Addr	Parameter	Format	Units/Scale	Description
Register Alias Record Queue (see section 5.2)				
0x80B0	Register Aliases	Uint16		Total number of aliases
0x80B1	Alias N	Uint16		Record index[1]
0x80B2	Fixed Register Address	Uint16		Register address from existing map

Modbus Addr	Parameter	Format	Units/Scale	Description
0x80B3	Alias Register Address	Uint16		Desired secondary address Only 0x4000~0x7FFFcan be used. One address can only be used once.
0x80B4 – 0x80FF	Reserved for expansion			

Notes:
[1]The contents of this register specifies an index into a set of records. Writing to this register will set the record number to expose for a read or right on the registers that follow. On subsequent reads/writes of any register in the set, the index will auto increment to the next available value.

Unlike record queues on for read only registers, this queue will not wrap when the index exceeds the number of records given by Register Aliases. Instead, a new record will be added.

7.4.4 AC Input Configuration Map

The following status registers are present on all devices which support AC transfer.

Modbus Addr	Parameter	Format	Units/Scale	Description
AC Input *Note: Each AC Input mapped to separate set of registers*				
0x8100	Config Type	Uint16	Enum	Set, this persistently selects which configuration is active on subsequent read or writes to the registers that follow. 00 = User 01 = OEM Defaults 10 = Factory Defaults 11 = Reserved
0x8101	Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)

Modbus Addr	Parameter	Format	Units/Scale	Description
0x8102	Low AC Volts Transfer Limit	Uint16	Vrms/X10	
0x8103	Low AC Frequency Transfer Limit	Uint16	Hz/X10	
0x8104	High AC Volts Transfer Limit	Uint16	Vrms/X10	
0x8105	High AC Frequency Transfer Limit	Uint16	Hz/X10	
0x8106 - 0x810F	*Reserved for expansion*			
0x8110 - 0x81FF	*Repeat of registers above for additional AC lines*			

7.4.5 Battery Configuration Map

The following status registers are present on all devices which support battery charging.

Modbus Addr	Parameter	Format	Units/Scale	Description
Battery Configuration *Note: Each battery bank mapped to separate set of registers*				
0x8200	Config Type	Uint16	Enum	Set, this persistently selects which configuration is active on subsequent read or writes to the registers that follow. 00 = User 01 = OEM Defaults 10 = Factory Defaults 11 = Reserved
0x8201	Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)

Modbus Addr	Parameter	Format	Units/Scale	Description
0x8202	Battery Type	Uint16	Enum	0 = Flooded 1 = Gel 2 = AGM 3 = Custom 4 = Deep Cycle 5 = Optima
0x8203	Battery Nominal Voltage	Uint16	Enum - Representing nominal voltage	12 =12V 24 = 24V 36 = 36V 48 = 48V 60 = 60V
0x8204	Battery Bank Capacity	Uint16	Ahr/X1	
0x8205	Battery Temp Coefficient	Sint16	mV/DegC X10	
0x8206	Battery Peukert Exponent	Uint16	x/X10	1.0 ≤ x ≥ 1.5
0x8207	Charge Efficiency Factor	Sint16	%/X10	-124.0% to 124.0%
0x8208	Battery Temp Without Sensor	Uint16	Enum	0 = Cold 1 = Warm 2 = Hot
0x8209, 0x820A	Battery Bulk Voltage Setpoint	Uint32	VDC/X100	
0x820B, 0x820C	Battery Float Voltage Setpoint	Uint32	VDC/X100	
0x820D, 0x820E	Battery Equalize Voltage Setpoint	Uint32	VDC/X100	
0x820F	Battery Equalize Time	Uint16	Min/X1	
0x8210	Battery Absorption CV Time	Uint16	Min/X1	
0x8210 - 0x821F	*Reserved for expansion*			
0x8220 - 0x82FF	*Repeat of registers above for additional battery banks*			

7.4.6 Charger Configuration Map

The following configuration registers are present on Xantrex devices that feature a charger.

Modbus Addr	Parameter	Format	Units/Scale	Description
Charger Configuration *Note: Each Charger mapped to separate set of registers*				
0x8300	Config Type	Uint16	Enum	Set, this persistently selects which configuration is active on subsequent read or writes to the registers that follow. 00 = User 01 = OEM Defaults 10 = Factory Defaults 11 = Reserved
0x8301	Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)
0x8302	Maximum Charge Rate	Unit16	A/X1	
0x8303	Charger Algorithm	Uint16	Enum	0x00 = Invalid 0x01 = 3 Stage 0x02 = 2 Stage 0x03 = CVCC 0x04 = Trickle
0x8304	Charge Time Begin	Uint16	Min/X1	Start of charge time in minutes since midnight < 1440
0x8305	Charge Time End	Uint16	Min/X1	End of charge time in minutes since midnight < 1440
0x8306 - 0x830F	Reserved for expansion			
0x8310 - 0x83FF	*Reserved for repeat of registers above for additional chargers*			

7.4.7 Inverter Configuration Map

The following configuration registers are present on Xantrex devices that feature an inverter.

Modbus Addr	Parameter	Format	Units/Scale	Description
Inverter Configuration *Note: Each Inverter mapped to separate set of registers*				
0x8400	Config Type	Uint16	Enum	Set, this persistently selects which configuration is active on subsequent read or writes to the registers that follow. 00 = User 01 = OEM Defaults 10 = Factory Defaults 11 = Reserved
0x8401	Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)
0x8402 – 0x8403	Low Battery Cut-out	Uint32	VDC/X100	
0x8404	Low Battery Cut-Out Delay	Uint16	Sec/X1	
0x8405 – 0x8406	Low Battery Cut-in	Uint32	VDC/X100	
0x8407 – 0x8408	Inverter High Battery Cut-out	Uint32	VDC/X100	
0x8409	Search Watts	Uint16	W/X1	
0x840A	Search Spacing	Uint16	Sec/X1	
0x840B – 0x840C	Grid-tie High Battery Cut-out	Uint32	VDC/X100	
0x840D – 0x840E	Battery Sell Volts	Uint32	VDC/X100	

Modbus Addr	Parameter	Format	Units/Scale	Description
0x840F – 0x8410	Grid-tie Low Battery Transfer	Uint32	VDC/X100	
0x8411	Max Sell Amp	Uint16	ADC/X10	
0x8412	Grid Amps AC	Uint16	ADC/X10	
0x8413	Sell Duration	Uint16	Minutes/X1	
0x8414 - 0x841F	*Reserved for expansion*			
0x8420 - 0x84FF	*Reserved for repeat of registers above for additional inverters*			

7.4.8 Auxiliary Output Triggers Configuration Map

The following status registers are present on all devices which support auxiliary output triggers.

Modbus Addr	Parameter	Format	Units/Scale	Description
Aux Output Triggers Config *Note: Each auxiliary output trigger mapped to separate set of registers*				
0x8500	Config Type	Uint16	Enum	Set, this persistently selects which configuration is active on subsequent read or writes to the registers that follow. 00 = User 01 = OEM Defaults 10 = Factory Defaults 11 = Reserved
0x8501	Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)
0x8502, 0x8503	Aux Out Voltage	Uint32	VDC/X100	
0x8504, 0x8505	Aux Out Current	Uint32	ADC/X100	
0x8506	Trigger Active Level	Uint16	enum	0 = Active Low 1 = Active High

Modbus Addr	Parameter	Format	Units/Scale	Description
0x8507	Trigger Source	Uint16	enum	0 = Manual Off 1 = Manual On 2 = Auto Off 3 = Auto On 4 = Batt V Low Off 5 = Batt V Low On 6 = Batt V High Off 7 = Batt V High On 8 = Array V High Off 9 = Array V High On 10 = Batt Temp Low Off 11 = Batt Temp Low On 12 = Batt Temp High Off 13 = Batt Temp High On 14 = Heat Sink Temp High Off 15 = Heat Sink Temp High On 16 = Fault Off 17 = Fault On
0x8508	Trigger Enable	Uint16	enum	0 = Disable 1 = Enable 2 = Auto
0x8509, 0x850A	Trigger Level	Uint32		Units and scale depend on Trigger Source: 0 = n/a 1 = n/a 2 = n/a 3 = n/a 4 = VDC/X100 5 = VDC/X100 6 = VDC/X100 7 = VDC/X100 8 = VDC/X100 9 = VDC/X100 10 = C/X10 11 = C/X10 12 = C/X10 13 = C/X10 14 = C/X10 15 = C/X10 16 = n/a 17 = n/a
0x850B	Trigger Delay	Uint16	Sec/X1	

Modbus Addr	Parameter	Format	Units/Scale	Description
0x850C - 0x850F	*Reserved for expansion*			
0x8510 - 0x85FF	*Reserves for repeat of registers above for additional aux trigger outputs*			

7.4.9 Automatic Generator Configuration Map

The following status registers are present on all devices which support an automatic generator.

Modbus Addr	Parameter	Format	Units/Scale	Description
Automatic Generator Start Configuration *Note: one set per supported genset mapped to separate set of registers*				
0x8600	Config Type	Uint16	Enum	Set, this persistently selects which configuration is active on subsequent read or writes to the registers that follow. 00 = User 01 = OEM Defaults 10 = Factory Defaults 11 = Reserved
0x8601	Generator Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)
0x8602	DC Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)
Generator Configuration				
0x8603	Generator Interface Type	Uint16	enum	1 = Type 1 to 30 = Type 30

Modbus Addr	Parameter	Format	Units/Scale	Description
0x8604	Relay 3 Usage	Uint16	enum	0 = Not Used 1 = Run 2 = Glow and Stop 3 = Glow and Stop with Shutdown bypass 4 = Crank 5 = Preheat 6 = Warm Up and Cool Down 7 = Start and Stop 8 = Preheat with Shutdown Bypass 9 = Momentary Run 10 = Pulse Stop
0x8605	Preheat Time	Uint16	Sec/X1	
0x8606	Generator Warm up Time	Uint16	Sec/X1	
0x8607	Maximum Run Time	Uint16	Min/X1	
0x8608	Generator Cool Down Time	Uint16	Sec/X1	
0x8609	Run Hold Time	Uint16	Sec/X1	
0x860A	Crank Time	Uint16	Sec/X1	
0x860B	Crank Retry Time	Uint16	Sec/X1	
0x860C	Starter Cool Down Time	Uint16	Sec/X1	
0x860D	Start Retries	Uint8		
0x860E-0x860F	*Reserved*			
Generator Exercise Time				
0x8610	Exercise Period	Uint8	Days/X1	
0x8611	Exercise Duration	Uint8	Mins/X1	
0x8612	Exercise Time of Day	Uint16	Mins/X1	Start of exercise time in minutes since midnight < 1440
Quiet Time				
0x8613	Quiet Time Enable	Uint16	enum	0 = disable 1 = enable

Modbus Addr	Parameter	Format	Units/Scale	Description
0x8614	Quiet Time Begin	Uint16	Min/X1	Start of quiet time in minutes since midnight < 1440
0x8615	Quiet Time End	Uint16	Min/X1	End of quiet time in minutes since midnight < 1440
SOC Trigger				
0x8616	Start SOC Enable	Uint8	enum	0 = disable 1 = enable
0x8617	Start SOC	Uint8	%/X10	
0x8618	Stop SOC Enable	Uint8	enum	0 = disable 1 = enable
0x8619	Stop SOC	Uint8	%/X10	
0x861A-0x861F	*Reserved*			
DC Source Voltage Trigger				
0x8620	Start DC Voltage Enable	Unit16	enum	0 = disable 1 = enable
0x8621, 0x8622	Start DC Voltage 30sec	Unit32	VDC/X100	
0x8623, 0x8624,	Start DC Voltage 15min	Unit32	VDC/X100	
0x8625, 0x8626	Start DC Voltage 2Hr	Unit32	VDC/X100	
0x8627, 0x8628	Start DC Voltage 24Hr	Unit32	VDC/X100	
0x8629	Stop DC Voltage Enable	Unit16	enum	0 = disable 1 = enable
0x862A, 0x862B	Stop DC Voltage	Unit32	VDC/X100	
AC Load Current Trigger				
0x862C	AC Load Start Enable	Unit16	enum	0 = disable 1 = enable
0x862D, 0x862E	Start AC Current Level	Uint32	Arms/X1000	
0x862F	Current Level Start Delay	Uint16	Min/X10	
0x8630	AC Load Stop Enable	Unit16	enum	0 = disable 1 = enable
0x8631, 0x8632	Stop AC Current Level	Uint32	Arms/X1000	
0x8633	Current Level Stop Delay	Uint16	Min/X10	

Modbus Addr	Parameter	Format	Units/Scale	Description
Charger State Trigger				
0x8634	Stop on Absorption	Uint8	enum	0 = disable 1 = enable
0x8635	Stop on Float	Uint8	enum	0 = disable 1 = enable
External Thermostat Trigger				
0x8636	Thermostat 1 Start Enable	Uint8	enum	0 = disable 1 = enable
0x8637	Thermostat 2 Start Enable	Uint8	enum	0 = disable 1 = enable
0x8638 - 0x863F	*Reserved for expansion*			
0x8635 - 0x86FF	*Reserved for repeat of registers above for additional automatic gensets*			

7.4.10 HMI Configuration Map

The following configuration registers are present on Xantrex human machine interface (HMI) devices.

Modbus Addr	Parameter	Format	Units/Scale	Description
HMI Configuration *Note: Each included HMI mapped to separate set of registers*				
0x8700	Config Type	Uint16	Enum	Set, this persistently selects which configuration is active on subsequent read or writes to the registers that follow. 00 = User 01 = OEM Defaults 10 = Factory Defaults 11 = Reserved
0x8701	Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)
0x8702	Data Update Interval	Uint16	Sec/X1	The rate at which data is refreshed (wireless panels only)

Modbus Addr	Parameter	Format	Units/Scale	Description
0x8703	Optimum Intensity Period	Uint16	Sec/X1	The time period the display is at maximum brightness
0x8704	Fade Intensity Period	Uint16	Sec/X1	The time period the display is at reduced brightness
0x8705	Fade Intensity	Uint16	%/X1	Brightness of the LCD for Fade intensity
0x8706	History Display Format	Uint16	Enum	1 = rightmost data is the most recent completed hour 2 = rightmost data is always close of day at midnight
0x8707, 0x8708	Tariff Rate	Uint32	currency/ X100,000	The cost for 1kWh of energy in the local currency (e.g. dollars, yen).
0x8709, 0x870A	NOx Avoided	Uint32	g/kWh/ X100,000	The grams of NOx emitted when generating 1kWh of energy.
0x870B, 0x870C	CO2 Avoided	Uint32	g/kWh/ X10,000	The grams of CO2 emitted when generating 1kWh of energy.
0x870D	Measurement Units	Uint16	Enum	Display values in metric or imperial units. 0 = imperial 1 = metric
0x870E, 0x870F	Fault Text Colour	Uint32	00RRGGBB	The RGB value of text displayed for fault conditions.
0x8710, 0x8711	Positive Value Colour	Uint32	00RRGGBB	The RGB value of positive number values.
0x8712, 0x8713	Negative Value Colour	Uint32	00RRGGBB	The RGB value of negative number values.
0x8714	12/24 Hour Clock	Uint16	Enum	Display time on the X axis using a 12 hour clock (using AM/PM) or 24 hour clock. 0 = 12 hour clock 1 = 24 hour clock

Modbus Addr	Parameter	Format	Units/Scale	Description
0x8715	Month Names	Uint16	Enum	Display the months on the X axis using the month names (JAN, FEB, … DEC) or numbers (1, 2, … 12). 0 = month names 1 = month numbers
0x8716 - 0x873F	*Reserved for expansion*			
0x8740 - 0x87FF	*Reserves for repeat of registers above for additional HMIs on the device*			

7.4.11 Instance Configuration Map

The following registers are valid for all Xanbus network connected devices.

Modbus Addr	Parameter	Format	Units/Scale	Description
AC Input *Note: Each AC Input mapped to separate set of registers*				
0x8800	Config Type	Uint16	Enum	Set, this persistently selects which configuration is active on subsequent read or writes to the registers that follow. 00 = User 01 = OEM Defaults 10 = Factory Defaults 11 = Reserved
0x8801	Instance	Uint16		0~256
0x8802 - 0x88FF	*Reserved for expansion*			

7.4.12 Maximum Power Point Tracking Configuration Map

Modbus Addr	Parameter	Format	Units/Scale	Description

Modbus Addr	Parameter	Format	Units/Scale	Description
Maximum Power Point Tracker Configuration Note: Each MPP tracker is mapped to a separate set of registers				
0x8900	Config Type	Uint16	Enum	Set, this persistently selects which configuration is active on subsequent read or writes to the registers that follow. 00 = User 01 = OEM Defaults 10 = Factory Defaults 11 = Reserved
0x8901	Connection ID	Uint16		If device is proxied, numbered relationship between system entities (see 7.2.3)
0x8902	MPPT Power Max Percentage	Uint16	%/x1	Max power output as a percent of nominal rating 0 – 110%
0x8903 – 0x8904	MPPT Voltage Reference	Uint32	VDC/x100	Reference voltage to track to
0x8905 – 0x8906	MPPT Voltage Reference Minimum	Uint32	VDC/x100	Minimum reference voltage to track to
0x8907 – 0x8908	MPPT Voltage Reference Maximum	Uint32	VDC/x100	Maximum reference voltage to track to
0x8909	MPP Tracker Rate	Uint16	Secs/x10	Power tracker rate
0x890A – 0x890B	MPP Tracker Step Size	Uint32	VDC/x100	Power tracker step size (perturb size)
0x890C - 0x890F	*Reserved for expansion*			
0x8910 – 0x89FF	*Repeat of registers above for additional MPP trackers*			

7.4.13 Feature Enable/Disable Configuration Map

The following registers are valid for all Xanbus network connected devices.

Modbus Addr	Parameter	Format	Units/Scale	Description
AC Input *Note: Each AC Input mapped to separate set of registers*				
0x8A00	Config Type	Uint16	Enum	Set, this persistently selects which configuration is active on subsequent read or writes to the registers that follow. 00 = User 01 = OEM Defaults 10 = Factory Defaults 11 = Reserved
0x8A01	Feature ID	Uint16	Enum	1: Remote power off 2: No load Derating 3. Restore Inverter mode on power up
0x8A02	Feature En/Disable	Uint16	Enum	0: Disabled 1: Enabled
0x8A03 - 0x8AFF	*Reserved for expansion*			

7.4.14 Reserved Configuration Map Register Blocks

This section defines the registers reserved for the addition of standard configuration data sets.

Modbus Addr	Parameter	Format	Units/Scale	Description
Reserved for expansion				
0x8B00-0x8FFF	Reserved for standard data set expansion.			

7.4.15 Device Specific Configuration Map

This section defines the registers reserved for device specific mapping of configuration registers. Refer to the register map in the appendix or device documentation for details.

Modbus Addr	Parameter	Format	Units/Scale	Description
Device Specific				

Modbus Addr	Parameter	Format	Units/Scale	Description
0x9000 - 0xEFFF	Device Specific			

Appendix A: State Enumerations

State	Enum Value
HIBERNATE	0
POWER_SAVE	1
SAFE	2
OPERATING	3
DIAGNOSTIC	4
LOADER	5
QUIET_TIME	265
AUTO_ON	266
AUTO_OFF	267
MANUAL_ON	268
MANUAL_OFF	269
GENERATOR_SHUTDOWN	270
EXTERNAL_SHUTDOWN	271
AGS_FAULT	272
SUSPEND	273
NOT_OPERATING	274
BULK	769
ABSORPTION	770
OVERCHARGE	771
EQUALIZE	772
FLOAT	773
CONSTANT_VI	775
CHARGE	785
ABSORPTION_EXIT_PENDING	786
GROUND_FAULT	787
INV	1024
AC_PASSTHRU	1025
LOAD_SENSE_ACTIVE	1027
LOAD_SENSE_READY	1029
GRID_TIED	1033
GRID_SUPPORT	1034
GEN_SUPPORT	1035
SELL_TO_GRID	1036
LOAD_SHAVING	1037
SCREEN_SAVER	1280
SCREEN_ACTIVE	1281

Appendix B: Connection ID Enumerations

Connection ID	Enum Value
AC-ANONYMOUS	0x1002
AC-SHORE1~16	0x1003~0X1012
AC-GEN1~16	0x1013~0x1022
AC-AC1~16	0x1023~0x1032
AC-LOAD1~16	0x1033~0x1042
AC-GRID1~16	0x1043~0x1052
DC-ANONYMOUS	0x2002
DC-HOUSE-BATT-BANK1~6	0x2003~0x2008
DC-START-BATT-BANK1~6	0x2009~0x200E
DC-SOLAR-ARRAY1~16	0x2015~0x2024

Appendix C: Gateway Modbus Map

Supported Special Functions	Reference	Notes/Exceptions
Default address assignment	5.1.2	Default address always 99
Data Record Queues	5.2	
Device Reprogramming	5.4	
Network Proxy	5.5	
Register Aliasing	5.6	The configured aliases apply to all the proxied devices globally. Setting aliases on the Gateway is equivalent to configuring all the proxied devices at the same time.

Modbus Addr Range	Supported Data Set	Detailed Reference	Notes/Exceptions
0x0000 – 0x0027	Product Info	7.2.1	
0x0028 – 0x004F	Remote Panel Product Info	7.2.1	
007F	Active Flt/Wrn Change	7.2.1	
0x00CF	Gateway State	7.2.1	
0x00D1	Remote Panel State	7.2.1	0 = never seen 1 = active 2 = inactive 3 = Loader
0x00D0	System State	7.2.1	
0x0100 – 0x010D	Device List	7.2.2	
0x0110 – 0x011C	Connection Map	7.2.3	
0x0180 – 0x0183	Device List	7.2.4	

0x0201 – 0x0206	System PV Array DC Input	7.2.5	Aggregated PV DC Input status of all connected devices Only real power supported, Other reply with 0xFFFF
0x0211 – 0x0216	System Inverter DC Input	7.2.5	Aggregated DC Input status of all connected inverters Only real power supported, Other reply with 0xFFFF
0x0302	System Battery SOC	7.2.6	Aggregated battery bank State of Charge
0x0312	Remote Panel Battery SOC	7.2.6	
0x0401 – 0x0406	System Charger DC Output	7.2.7	Aggregated Battery DC Output status of all connected chargers Only real power supported, Other reply with 0xFFFF
0x0411 – 0x0416	System Inverter DC Output	7.2.7	Aggregated DC Output status of all connected inverters Only real power supported, Other reply with 0xFFFF
0x0501 – 0x0507	System AC Input	7.2.8	Aggregated AC Input status of all connected devices Only real power supported, Other reply with 0xFFFF
0x0701 – 0x0707	System Inverter AC Output	7.2.9	Aggregated AC Output status of all connected inverters Only real power supported, Other reply with 0xFFFF
0x0800 – 0x0809	System Energy History	7.2.11	Aggregated history of system
0x07FE – 0x0809	Removed XB DEVICE Energy History	7.2.11	Removed xb device history data 1
0x080E – 0x0819	Removed XB DEVICE Energy History	7.2.11	Removed xb device history data 2
0x081E – 0x0829	Removed XB DEVICE Energy History	7.2.11	Removed xb device history data 3
0x082E – 0x0839	Removed XB DEVICE Energy History	7.2.11	Removed xb device history data 4
0x0900	Remote Panel Ambient Temperature	7.2.12	

0x0C00, 0x0C01	Power Rate	7.2.15	System total
0x0C02, 0x0C03	Temperature range	7.2.15	Remote Panel Temp sensor
0x0D00 - 0x0D07	PV range DC	7.2.15	System total
0x0D10 - 0x0D17	Battery range	7.2.15	System total
0x0E00 - 0x0E09	AC range	7.2.15	System total
0x0F00	Self Test Result	7.2.16	
0x1100 - 0x1109	Gateway Loader version	7.2.18	
0x110A - 0x1113	Gateway Application version	7.2.18	
0x1200 - 0x1209	Remote Loader version	7.2.18	
0x120A - 0x1213	Remote Application version	7.2.18	
0xF000	Reset Command	7.3.1	controller 0 = Gateway controller 1 = Remote Panel
0xF002	Clear Log	7.3.1	
0xF005	Button pushed	7.3.1	For remote panel communication initialization
0xFF70 – 0xFFF2	Remote Panel Reprogramming	7.1.2 7.1.3	
0x8000 – 0x8007	Protocol Configuration	7.4.1	Bus biasing and termination not supported Only one stop bit supported
0x8010 – 0x8014	Time and Date Configuration	7.4.1	
0x8020 – 0x803E	Personalization	7.4.2	
0x80B0 –	Register Alias Configuration	7.4.3	

0x80B3			
0x9000 – 0x9007	Remote Panel Protocol Configuration	7.4.1	
0x9020 – 0x903E	Remote Panel Personalization	7.4.2	
0x8700 – 0x8715	HMI Configuration	7.4.10	

Appendix D: GT Series Grid-tie Inverter Modbus Map

Supported Special Functions	Reference	Notes/Exceptions
Default address assignment	5.1.2	By proxy through Gateway
Data Record Queues	5.2	
Device Reprogramming	5.4	By proxy through Gateway
Register Aliasing	5.6	Globally through Gateway

Modbus Addr Range	Supported Data Set	Detailed Set Reference	Notes/Exceptions
0x0000 – 0x0027	Product Info	7.2.1	
0x0080 – 0x0099	Active Faults	7.2.1	
0x00A0	Active Warning	7.2.1	Value always 0xFFFF
0x1000 – 0x1019	Fault Log	7.2.16	
0x1020	Warning Log	7.2.16	Value always 0xFFFF
0x00CF	Device State	7.2.1	
0x0200 – 0x0206	PV Array DC Input Status	7.2.5	
0x0700 – 0x0707	AC Output Status	7.2.10	Only real power supported
0x0800 – 0x0809	AC Output Energy History	7.2.11	Logged by Gateway, supplied by proxy
0x0900	Heat Sink Temp	7.2.11	
0x0C00, 0x0C01	Power Rate	7.2.15	Device
0x0C02, 0x0C03	Temp range	7.2.15	For heat sink display
0x0D00 -	PV range	7.2.15	Device

0x0D07			
0x0E00 - 0x0E09	AC range	7.2.15	Device
0x1100 - 0x1109	Loader version	7.2.18	
0x110A - 0x1113	Application version	7.2.18	
0xF000	Reset Command	7.3.1	
0xF001	System Control Command	7.3.1	
0xF002	Clear Log	7.3.1	
0xF003	Clear Specific Fault	7.3.1	
0xF201	Inverter on/off	7.3.4	
0x8021 – 0x803E	Personalization	7.4.2	
0x8800, 0x8801	Instance	7.4.11	Xanbus Related

Note: Fault IDs of GT defined in Appendix K

Appendix E: XW Series Charge Controller Modbus Map

Supported Special Functions	Reference	Notes/Exceptions
Default address assignment	5.1.2	By proxy through Gateway
Data Record Queues	5.2	
Device Reprogramming	5.4	By proxy through Gateway
Register Aliasing	5.6	Globally through Gateway

Modbus Addr Range	Supported Data Set	Detailed Set Reference	Notes/Exceptions
0x0000 – 0x0027	Product Info	7.2.1	
0x0080 – 0x0099	Active Faults	7.2.1	
0x00A0 – 0x00B9	Active Warnings	7.2.1	
0x1000 – 0x1019	Fault Log	7.2.17	
0x1020 – 0x1039	Warning Log	7.2.17	
0x00CF	Device State	7.2.1	
0x0200 – 0x0206	PV Array DC Input Status	7.2.5	
0x0301	Battery Temperature	7.2.6	
0x0401 – 0x0407	Battery DC Output Status	7.2.7	
0x0800 – 0x0809	PV Energy History	7.2.111	Logged by Gateway, supplied by proxy
0x0900	Heatsink	7.2.12	

	Temperature		
0x0A00 – 0x0A06	Aux Output Triggers Status	7.2.13	
0x0C00, 0x0C01	Power Rate	7.2.15	Device
0x0C02, 0x0C03	Temp range	7.2.15	Device battery senor
0x0C04, 0x0C05	Temp range	7.2.15	Device heat sink
0x0D00 - 0x0D07	PV range	7.2.15	Device
0x0D10 - 0x0D07	Battery range	7.2.15	Device
0x1100 - 0x1109	Loader version	7.2.18	
0x110A - 0x1113	Application version	7.2.18	
0xF000	Reset Command	7.3.1	
0xF001	System Control Command	7.3.1	
0xF002	Clear Log	7.3.1	
0xF100 - 0xF103	Charger Control	7.3.3	
0xF400 - 0xF401	Charger Control	7.3.6	
0x8021 – 0x803E	Personalization	7.4.2	
0x8200 – 0x8210	Battery Configuration	7.4.5	Battery Peukert Exponent, Charge Efficiency Factor not supported
0x8300 – 0x8303	Charger Configuration	7.4.6	Charge time not supported
0x8500 – 0x850B	Aux Output Configuration	7.4.8	

0x8800, 0x8801	Instance	7.4.11	Xanbus Related
0x8900 _ 0x8908	MPPT Configuration	7.4.12	

Note: Fault IDs and Warning IDs of Charger defined in Appendix K

Appendix F: XW Series Inverter/Charger Modbus Map

Supported Special Functions	Reference	Notes/Exceptions
Default address assignment	5.1.2	By proxy through Gateway
Data Record Queues	5.2	
Device Reprogramming	5.4	By proxy through Gateway
Register Aliasing	5.6	Globally through Gateway

Modbus Addr Range	Supported Data Set	Detailed Set Reference	Notes/Exceptions
0x0000 – 0x0027	Product Info	7.2.1	
0x0080 – 0x0099	Active Faults	7.2.1	
0x00A0 – 0x00B9	Active Warnings	7.2.1	
0x1000 – 0x1019	Fault Log	7.2.17	
0x1020 – 0x1039	Warning Log	7.2.17	
0x1040 – 0x1058	State Log	7.2.17	
0x00CF	Device State	7.2.1	
0x0200 – 0x0206	Battery DC Input Status	7.2.5	
0x0301	Battery Temperature	7.2.6	
0x0302	Battery SOC	7.2.6	
0x0400 – 0x0407	Battery DC Output Status	7.2.7	

0x0500 – 0x0507	AC1 AC Input Status	7.2.8	Only real power supported
0x0510 – 0x0517	AC2 AC Input Status	7.2.8	Only real power supported
0x0600 – 0x0604	AC1 Source Status	7.2.9	
0x0610 – 0x0614	AC2 Source Status	7.2.9	
0x0700 – 0x0707	AC1 AC Output Status	7.2.10	Only real power supported
0x0710 – 0x0717	Critical Loads AC Output Status	7.2.10	Only real power supported
0x0800 – 0x0809	Grid-tie Energy History	7.2.11	AC1 Output Logged by Gateway, supplied by proxy
0x0810 – 0x0819	Generator Energy History	7.2.11	AC2 Input Logged by Gateway, supplied by proxy
0x0820 – 0x0829	Critical Loads Energy History	7.2.11	Logged by Gateway, supplied by proxy
0x0830 – 0x0839	Grid Input Energy History	7.2.11	Logged by Gateway, supplied by proxy
0x0840 – 0x0849	XW Energy History	7.2.11	Logged by Gateway, supplied by proxy Max power and harvest are not valid
0x0900	Transformer Temperature	7.2.12	
0x0910	FET 1 Temperature	7.2.12	
0x0920	FET 2 Temperature	7.2.12	
0x0930	Capacitor Temperature	7.2.12	
0x0A00 – 0x0A06	Aux Output Triggers Status	7.2.13	
0x0C00,	Power Rate	7.2.15	Device

0x0C01			
0x0C02, 0x0C03	Temp range	7.2.15	Battery sensor
0x0C04, 0x0C05	Temp range	7.2.15	Device Transformer, FET and battery temperature
0x0D00 - 0x0D07	Battery range	7.2.15	Device
0x0E00 - 0x0E09	AC range	7.2.15	Device
0x1100 - 0x1109	Loader version	7.2.18	
0x110A - 0x1113	Application version	7.2.18	
0xF000	Reset Command	7.3.1	
0xF001	System Control Command	7.3.1	
0xF002	Clear Log	7.3.1	
0xF100 – 0xF103	Charger Control	7.3.3	
0xF200 – 0xF205	Inverter Control	7.3.4	
0x8021 – 0x803E	Personalization	7.4.2	
0x8100 – 0x8105	AC1 Input Configuration	7.4.4	
0x8110 – 0x8115	AC2 Input Configuration	7.4.4	
0x8200 – 0x8210	Battery Configuration	7.4.5	Battery Peukert Exponent, Charge Efficiency Factor not supported
0x8300 – 0x8305	Charger Configuration	7.4.6	
0x8400 –	Inverter Configuration	7.4.7	

0x8413			
0x8500 – 0x850B	Aux Output Configuration	7.4.8	
0x8800, 0x8801	Instance	7.4.11	Xanbus Related
0x8A00 – 0x8A02	Feature En/Disable Configuration	7.4.13	

Note: Fault IDs of XW defined here.

Fault Description	Fault Id
AC_OUT_UV_SD	1
AC_OUT_OV_SD	2
AC1_IN_L1_UV_SD	3
AC1_IN_L1_OV_SD	4
AC1_IN_L1_UF_SD	5
AC1_IN_L1_OF_SD	6
AC1_IN_L2_UV_SD	7
AC1_IN_L2_OV_SD	8
AC1_IN_L2_UF_SD	9
AC1_IN_L2_OF_SD	10
AC2_IN_L1_OV_SD	11
AC2_IN_L1_UV_SD	12
AC2_IN_L2_OV_SD	13
AC2_IN_L2_UV_SD	14
AC2_IN_L1_OF_SD	15
AC2_IN_L1_UF_SD	16
ACBF_AC1_L1_SD	17
ACBF_AC1_L2_SD	18
ACBF_AC2_L1_SD	19
ACBF_AC2_L2_SD	20
ACBF_ACX_L1L2_SD	21
ACBF_ACX_L1_SD	22
AI_QUAL_OF_SD	23
AI_QUAL_UF_SD	24
AI_OF_SD	25
AI_UF_SD	26
AI_OV_L1_FAST_SD	27
AI_OV_L2_FAST_SD	28
AI_QUAL_OV_L1L2_SD	29
AI_OV_L1L2_FAST_SD	30
AI_OV_L1_SLOW_SD	31

Fault Description	Fault Id
AI_OV_L2_SLOW_SD	32
AI_OV_L1L2_SLOW_SD	33
AI_UV_L1_SLOW_SD	34
AI_UV_L2_SLOW_SD	35
AI_UV_L1L2_SLOW_SD	36
AI_UV_L1_FAST_SD	37
AI_UV_L2_FAST_SD	38
AI_QUAL_L1L2_UV_SD	39
AI_UV_L1L2_FAST_SD	40
APS_UV_SD	41
APS_OV_SD	42
BATT_UT_SD	43
BATT_OT_SD	44
CAP_OT_SD	45
CONTROLLER_ERR_SD	46
DC_UV_IMM_SD	47
DC_UV_SD	48
DC_OV_SD	49
DEAD_BATT_CHG	50
EE_SD	51
EE_CAL_FAIL	52
EE_CONFIG_FAIL	53
EE_DEFAULT_FAIL	54
EE_LOG_FAIL	55
EE_STRINGS_FAIL	56
FET1_OT_SD	57
FET2_OT_SD	58
GO_CONFIG_YOURSELF_ERR	59
INVALID_FLT_CODE	60
INVALID_WRN_CODE	61
INVALID_INTERRUPT	62
OL_PRIM_SD	63
OL_SEC_ONE_SD	64
OL_SEC_TWO_SD	65
SYS_CFG_SD	66
WD_RST_SD	67
XFMR_OT_SD	68
XSIG_BAD_SD	69

Note: Warning IDs of XW defined here.

Warning Description	Warning Id

Warning Description	Warning Id
AC_IN_L1_UV_WRN	3
AC_IN_L1_OV_WRN	4
AC_IN_L2_UV_WRN	7
AC_IN_L2_OV_WRN	8
AC2_IN_L1_OV_WRN	11
AC2_IN_L1_UV_WRN	12
AC2_IN_L2_OV_WRN	13
AC2_IN_L2_UV_WRN	14
BATT_TEMP_OT_WRN	44
CAP_OT_WRN	45
INV_DC_UV_WRN	48
DC_OV_WRN	49
EE_WRN	51
FET1_OT_WRN	57
FET2_OT_WRN	58
OL_PRIM_WRN	63
OL_SEC_ONE_WRN	64
OL_SEC_TWO_WRN	65
XFMR_OT_WRN	68
EQUALIZE_ABORT_WRN	95
CANNOT_EQUALIZE_WRN	96
BATT_TEMP_SENSOR_SHORT_WRN	97
BATT_TEMP_SENSOR_REMOVED_WRN	98
AUTORSTR_EN_WRN	99

Appendix G: XW Series AGS Modbus Map

Supported Special Functions	Reference	Notes/Exceptions
Default address assignment	5.1.2	By proxy through Gateway
Data Record Queues	5.2	
Device Reprogramming	5.4	By proxy through Gateway
Register Aliasing	5.6	Globally through Gateway

Modbus Addr Range	Supported Data Set	Detailed Set Reference	Notes/Exceptions
0x0000 – 0x0027	Product Info	7.2.1	
0x0080 – 0x0099	Active Faults	7.2.1	
0x00A0 – 0x00B9	Active Warnings	7.2.1	
0x1000 – 0x1019	Fault Log	7.2.17	
0x1020 – 0x1039	Warning Log	7.2.17	
0x1040 – 0x1044	State Log	7.2.17	Strings not supported
0x00CF	Device State	7.2.1	
0x0B00 – 0x0B05	Auto Gen Start Status	7.2.14	
0x1100 - 0x1109	Loader version	7.2.18	
0x110A - 0x1113	Application version	7.2.18	

0xF000	Reset Command	7.3.1	
0xF001	System Control Command	7.3.1	
0xF002	Clear Log	7.3.1	
0xF300 – 0xF303	Auto Gen Start Control	7.3.5	
0x8021 – 0x803E	Personalization	7.4.2	
0x8600 – 0x8637	Auto Gen Start Configuration	7.4.9	
0x8800, 0x8801	Instance	7.4.11	

Note: Fault IDs and Warning IDs of AGS defined in Appendix K.

Appendix H: SCP Modbus Map

Supported Special Functions	Reference	Notes/Exceptions
Default address assignment	5.1.2	By proxy through Gateway
Data Record Queues	5.2	
Device Reprogramming	5.4	By proxy through Gateway
Register Aliasing	5.6	Globally through Gateway

Modbus Addr Range	Supported Data Set	Detailed Set Reference	Notes/Exceptions
0x0000 – 0x0027	Product Info	7.2.1	
0x0080 – 0x0099	Active Faults	7.2.1	
0x00A0 – 0x00B9	Active Warnings	7.2.1	
0x1000 – 0x1019	Fault Log	7.2.17	
0x1020 – 0x1039	Warning Log	7.2.17	
0x1040 – 0x1044	State Log	7.2.17	Strings not supported
0x00CF	Device State	7.2.1	
0x1100 - 0x1109	Loader version	7.2.18	
0x110A - 0x1113	Application version	7.2.18	
0xF000	Reset Command	7.3.1	
0xF001	System Control	7.3.1	

	Command		
0xF002	Clear Log	7.3.1	
0x8010 – 0x8012	Time and Date Configuration	7.4.1	
0x8021 – 0x803E	Personalization	7.4.2	
0x8800, 0x8801	Instance	7.4.11	

Note: Fault IDs and Warning IDs of SCP defined in Appendix K.

Appendix I: Device Discovery

Assuming the modbus network baudrate is the same, Modbus devices can typically be discovered in one of two ways: either the master is manually configured with all the hard coded addresses, or the master must scan through the range of possible addresses using a Modbus function 8 query message (similar to a ping in the internet network domain). Each method has its own set of tradeoffs.

Manual configuration of the master has the least run-time impact. If the master is told the specific Modbus addresses to use, there is no need for the master to send request messages to devices which do not exist. This works very well for static networks, where all the devices are known at the time of commisioning and the devices can be easily individually configured for a specific address. It doesn't work as well for installations where devices may be added at any time, or where devices determine their own addresses without human intervention.

Modbus function 8 (diagnostic) sub-function 0 (query) is a request that the slave device send the received packet back to the master. This function can be used to determine if a device exists at a given address. The master can send this message to every possible Modbus slave address (1-247). If it receives a response, then it keeps track of the newly discovered device. If it doesn't receive a response, then it simply probes the next address. The major drawback with this scenario is that not receiving a response takes extra time and ties up the master. The response time-out is application specific, but the Modbus over Serial Line specification states that the time out is typically a minimum of 1 second at 9600baud. This means that it could take a minimum of 247 seconds to scan for every possible slave address on the bus. Section 5.1.1 states that there can only be 100 non-proxied Modbus slaves on the bus, so the scan time would effectively be reduced to a minumum of 100 seconds.

There are two choices for the Master to scan the bus.

1. The master could scan the complete address range continuously.
2. The master could scan the complete address range only when requested by the user.

If the master is scanning continuously, then every other desired transaction on the bus would potentially be delayed waiting for the query request to time out. A reasonable solution could be to have a long delay between successive query. This reduces the likelyhood that any particular transaction would be impacted, but would add an element of randomness to the communications.

If the master scans the bus only when requested by the user, then the only time there would be an impact was when the user knew that devices had been added to or removed from the bus. Regular transactions would not be impacted. The typical minimum scan time of 98 seconds would only be incurred once during system maintenance functions.

Appendix J: Wireless Remote Panel Reprogramming Procedure Through Gateway485

1. Reprogramming Initiation

Reprogramming can only be started when the wireless remote is plugged into the charging unit. The remote does not switch to a low-power mode of operation when the charger is plugged in, so the opportunity exists for the remote to frequently poll the gateway to determine if a software load is available.

When the user requests a software upgrade, reprogramming tools will send a command to set the remote panel to the loader state.

The reprogramming process will start after Remote Panel is in Loader state.

2. Modbus Registers involved in reprogramming and their handlings

Reprogramming Wireless Remote Panel through Gateway485 actually involves the interaction of two Modbus network. The Remote Panel is the Modbus master of the network comprised of the Gateway485 and the Remote Panel. The reprogramming tool is the Modbus master of the other network which includes the Gateway485 as slave.

The Remote Panel updates its status stored in Gateway485 for the reprogramming tool to read. On the other side, the Remote Panel gets the configurations and commands from the Gateway485 which were set by the reprogramming tool.

There cannot be a great delay in the communication between the Remote Panel and tools, since the actions are typically driven by an end user with a standard web browser.

The tool will implement a read/write sequence to make sure that the values read from the gateway registers are the latest and that the peer device is active. The read sequence must write 0xFFFF to the desired register, and then poll that register for a changed value, which should be the latest value updated by the Remote Panel. Similarly, the write operation must be followed by a read to see the actual Remote Panel value.

The following registers of the Gateway are used for reprogramming the Remote Panel:

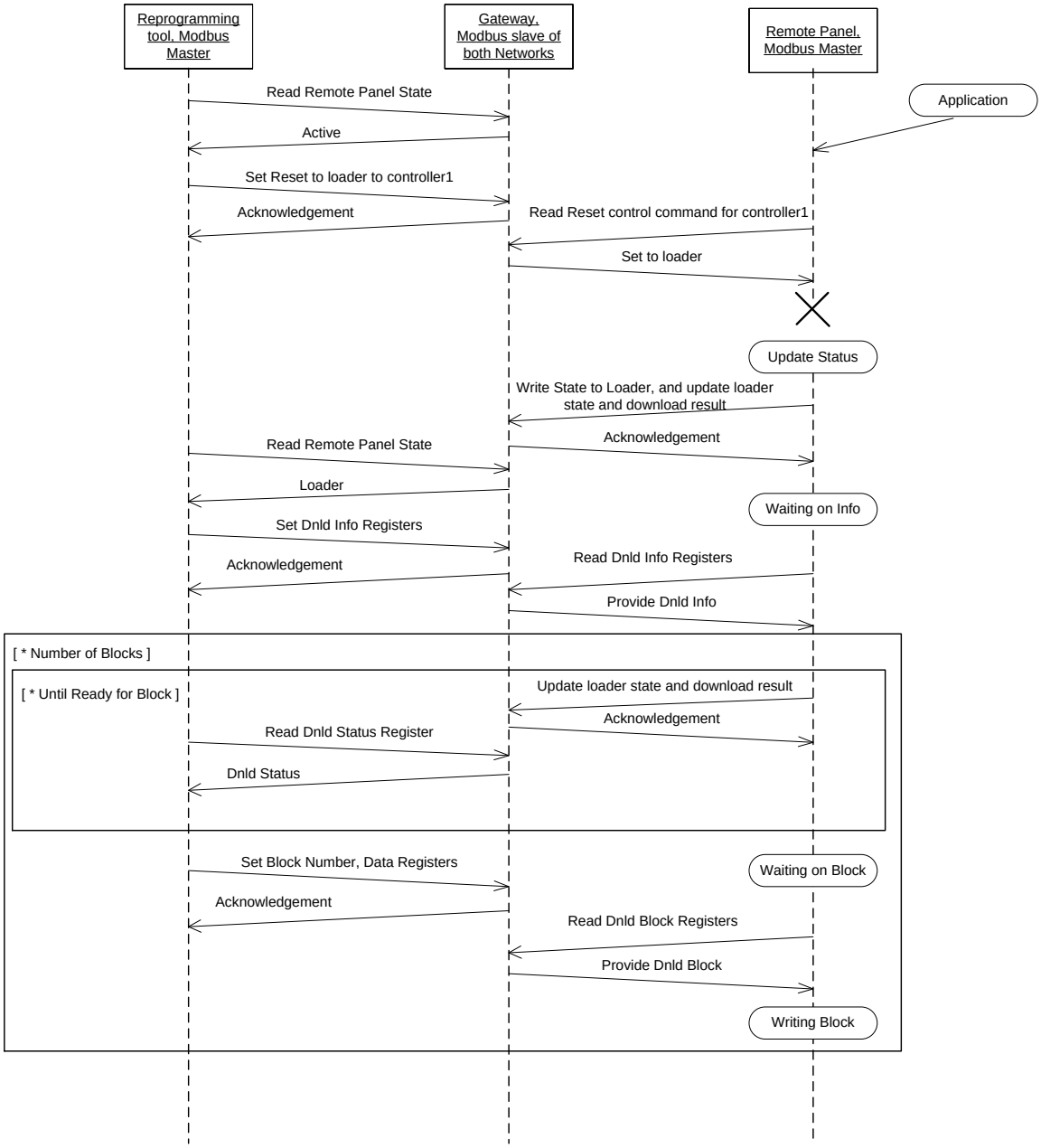
0x00D1: Remote Panel operation status shows whether it is in active or not, whether it is running the loader, or running the normal application.

0xF000: Reset command has to have the “controller” half of the register set to 1 to control the Remote Panel.

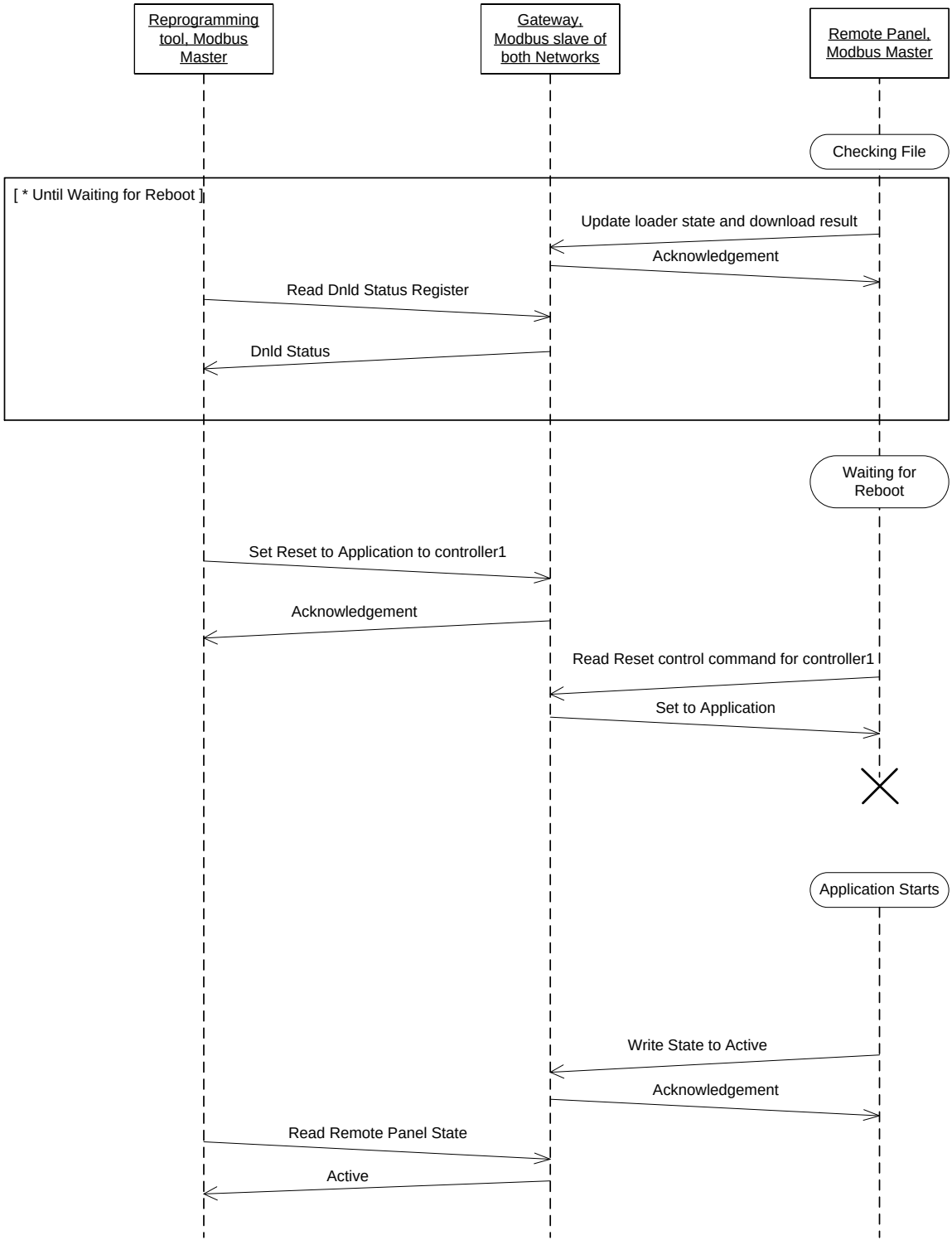
0xFF70, 0xFF71: Remote Panel loader state and download result.

0xFF72~0xFFF2: Download Information and Download block control command from reprogramming tools to Remote Panel.

3. Reprogramming sequence diagram


Reprogramming tool, Modbus Master
Gateway, Modbus slave of both Networks
Remote Panel, Modbus Master
Application
Read Remote Panel State
Active
Set Reset to loader to controller1
Acknowledgement
Read Reset control command for controller1
Set to loader
Update Status
Write State to Loader, and update loader state and download result
Acknowledgement
Read Remote Panel State
Loader
Waiting on Info
Set Dnld Info Registers
Acknowledgement
Read Dnld Info Registers
Provide Dnld Info
[* Number of Blocks]
[* Until Ready for Block]
Update loader state and download result
Acknowledgement
Read Dnld Status Register
Dnld Status
Set Block Number, Data Registers
Waiting on Block
Acknowledgement
Read Dnld Block Registers
Provide Dnld Block
Writing Block

Continued



4. Reprogramming file format

Reprogramming file is in ELF format.

The loader stores the file size, CRC, and total Blocks when getting download Information; and erases the program area to get ready for reprogramming.

Block Data is a multiple of 8 bytes with the maximum size determined by the ZigBee protocol packet size.

Loader updates the file size received, CRC, and block received while receiving a new block of data, and programming the code.

After receiving all blocks of data, check for file size and CRC. If everything is fine at this point, the loader automatically reboots.

Appendix K: XanBus Device Fault and Warning ID

Fault Description	Fault Id
TRANSISTOR_OVER_TEMPERATURE_SHUTDOWN1	0
TRANSISTOR_OVER_TEMPERATURE_SHUTDOWN2	1
CAPACITOR_OVER_TEMPERATURE_SHUTDOWN1	2
TRANSFORMER_OVER_TEMPERATURE_SHUTDOWN1	3
BATTERY_OVER_TEMPERATURE_SHUTDOWN1	4
AMBIENT_OVER_TEMPERATURE_SHUTDOWN1	5
AC_OUTPUT_OVERLOAD_SHUTDOWN1	6
AC_OUTPUT_OVERLOAD_PEAK_CURRENT_SHUTDOWN1	7
NEUTRAL_LOSS_SHUTDOWN	8
DC_OVER_VOLTAGE_SHUTDOWN1	9
DC_UNDER_VOLTAGE_IMMEDIATE_SHUTDOWN1	10
DC_UNDER_VOLTAGE_SHUTDOWN1	11
AC_INPUT_LINE1OVER_VOLTAGE_SHUTDOWN	12
AC_INPUT_LINE1UNDER_VOLTAGE_SHUTDOWN	13
AC_INPUT_LINE2OVER_VOLTAGE_SHUTDOWN	14
AC_INPUT_LINE2UNDER_VOLTAGE_SHUTDOWN	15
AC_INPUT_LINE1OVER_FREQUENCY_SHUTDOWN	16
AC_INPUT_LINE1UNDER_FREQUENCY_SHUTDOWN	17
AC_INPUT_LINE2OVER_FREQUENCY_SHUTDOWN	18
AC_INPUT_LINE2UNDER_FREQUENCY_SHUTDOWN	19
ADC_CONVERSION_TIMEOUT	22
AC_OUTPUT_OVER_VOLTAGE_SHUTDOWN1	23
AC_OUTPUT_UNDER_VOLTAGE_SHUTDOWN1	24
APS_OVER_VOLTAGE	25
APS_UNDER_VOLTAGE	26
OPS_OVER_VOLTAGE	27
OPS_UNDER_VOLTAGE	28
AC_BACKFEED_SHUTDOWN	29
BATTERY_UNDER_TEMPERATURE	30
ECHO_CHARGER_SHORT_CIRCUIT	31
ECHO_CHARGER_REVERSE_POLARITY	32
ECHO_CHARGER_INPUT_OVER_VOLTAGE	33
ECHO_CHARGER_INPUT_UNDER_VOLTAGE	34

Fault Description	Fault Id
ECHO_CHARGER_OVER_TEMPERATURE	35
ECHO_CHARGER_UNDER_TEMPERATURE	36
ECHO_CHARGER_NOT_COMMUNICATING	37
LOST_INTERPROCESSOR_COMMUNICATIONS	40
DEAD_BATTERY_CHARGE_TIMEOUT	41
AC_OUTPUT_OVER_VOLTAGE_SHUTDOWN2	42
AC_OUTPUT_UNDER_VOLTAGE_SHUTDOWN2	43
GENERATOR_LINE1INPUT_OVER_VOLTAGE_SHUTDOWN	44
GENERATOR_LINE1INPUT_UNDER_VOLTAGE_SHUTDOWN	45
GENERATOR_LINE2INPUT_OVER_VOLTAGE_SHUTDOWN	46
GENERATOR_LINE2INPUT_UNDER_VOLTAGE_SHUTDOWN	47
GENERATOR_LINE1INPUT_OVER_FREQUENCY_SHUTDOWN	48
GENERATOR_LINE1INPUT_UNDER_FREQUENCY_SHUTDOWN	49
GENERATOR_LINE2INPUT_OVER_FREQUENCY_SHUTDOWN	50
GENERATOR_LINE2INPUT_UNDER_FREQUENCY_SHUTDOWN	51
AC_INPUT_L1L2120OVER_FREQUENCY_SHUTDOWN	52
AC_INPUT_L1L2120UNDER_FREQUENCY_SHUTDOWN	53
APS1OFF	54
HEATSINK1OVER_TEMPERATURE_SHUTDOWN	55
GROUND_FAULT	56
EXTERNAL_SHUTDOWN	57
AC_OUTPUT1VOLTAGE_SHUTDOWN	58
AC_OUTPUT1FREQUENCY_SHUTDOWN	59
AC_OUTPUT1IMPEDANCE_SHUTDOWN	60
RECONNECTING1	61
DCDC1SATURATED	62
USER_SHUTDOWN	63
NEUTRAL_OVER_CURRENT1	64
OVER_VOLTAGE_ANTIISLANDING	65
UNDER_VOLTAGE_ANTIISLANDING	66
OVER_FREQUENCY_ANTIISLANDING	67
UNDER_FREQUENCY_ANTIISLANDING	68
BATTERY_CONFLICTING_SETUP	69
DC_OVER_VOLTAGE_SHUTDOWN2	70
DC_OVER_CURRENT_SHUTDOWN	71
SPS_OVERLOAD	72

Fault Description	Fault Id
MAX_START_RETRIES	200
UNABLETO_STOP_GENERATOR	201
GEN_STOPBY_EXT_SHUTDOWN	203
CLOCK_NOT_SET	250
DEVICE_LOST	251
NEW_DEVICE	252
SILICON_SERIAL_ID_FAILURE	500
NON_VOLATILE_ERROR	501
WATCHDOG_RESET	502
REALTIME_CLOCK_FAILURE	503
LOST_NETWORK_CONNECTION	504
CONTROLLER_FAILURE	505
WRONG_FAULT_IDENTIFIER	506
WRONG_WARNING_IDENTIFIER	507
INVALID_INTERRUPT	508
ASSOCIATIONS_INVALID	509

Warning Description	Warning Id
TRANSISTOR_OVER_TEMPERATURE_WARNING1	0
TRANSISTOR_OVER_TEMPERATURE_WARNING2	1
TRANSISTOR_OVER_TEMPERATURE_WARNING2	2
TRANSFORMER_OVER_TEMPERATURE_WARNING1	3
BATTERY_OVER_TEMPERATURE_WARNING1	4
AMBIENT_OVER_TEMPERATURE_WARNING1	5
AC_OUTPUT_OVERLOAD_WARNING	6
AUTO_RESTART_AFTER_FAULT_ENABLED_WARNING	7
BATTERY_SENSOR_NOT_PRESENT_WARNING	9
BATTERY_SENSOR_SHORT_WARNING	10
DC_OVER_VOLTAGE_WARNING1	11
DC_UNDER_VOLTAGE_WARNING1	12
AC_LINE1INPUT_OVER_VOLTAGE_WARNING	13
AC_LINE1INPUT_UNDER_VOLTAGE_WARNING	14
AC_LINE2INPUT_OVER_VOLTAGE_WARNING	15
AC_LINE2INPUT_UNDER_VOLTAGE_WARNING	16
CANNOT_EQUALIZE	18
EQUALIZATION_TERMINATED_ABNORMALLY	19

Warning Description	Warning Id
AC_OUTPUT_OVER_VOLTAGE_WARNING1	20
AC_OUTPUT_UNDER_VOLTAGE_WARNING1	21
AC_OUTPUT_OVER_VOLTAGE_WARNING2	22
AC_OUTPUT_UNDER_VOLTAGE_WARNING2	23
GENERATOR_LINE1INPUT_OVER_VOLTAGE_WARNING	24
GENERATOR_LINE1INPUT_UNDER_VOLTAGE_WARNING	25
GENERATOR_LINE2INPUT_OVER_VOLTAGE_WARNING	26
GENERATOR_LINE2INPUT_UNDER_VOLTAGE_WARNING	27
GENERATOR_LINE1INPUT_OVER_FREQUENCY_WARNING	28
GENERATOR_LINE1INPUT_UNDER_FREQUENCY_WARNING	29
GENERATOR_LINE2INPUT_OVER_FREQUENCY_WARNING	30
GENERATOR_LINE2INPUT_UNDER_FREQUENCY_WARNING	31
AC_INPUT_L1L2120OVER_FREQUENCY_WARNING	32
AC_INPUT_L1L2120UNDER_FREQUENCY_WARNING	33
EXTERNAL_GENSET_SHUTDOWN_WARNING	200
EXTERNAL_GENSET_STARTUP_WARNING	201
UNABLE_TO_START_GENSET_WARNING	202
REACHED_MAX_RUN_TIME_WARNING	203
MAX_GENSET_CYCLE_WITH_NO_INTERVENTION_WARNING	204
GEN_ON_NOT_AGS	205
ORPHANED_AUTO_START_TRIGGER	206
ORPHANED_AUTO_STOP_TRIGGER	207
NO_AUTO_TRIGGERS_DEFINED	208
GEN_OFF_NOT_AGS	209
FAILED_TO_SET_VALUE_WARNING	250
CONFIRM_EQUALIZATION_WARNING	251
CONFIRM_RESTORE_DEFAULTS_WARNING	252
ABORTED_EQUALIZATION	253
FAILED_TO_SET_VALUE_DUE_TO_SYSTEM_MODE	254
CLOCK_NOT_SET	255
DEVICE_LOST	256
NEW_DEVICE	257
NODE_INSTANCE_DUPLICATED	258
LOST_NETWORK_CONNECTION	500
DEFAULTS_RESTORED_ON_POWERUP_WARNING	501
MISSING_EXPECTED_STATUS	502

Warning Description	Warning Id
ASSOCIATIONS_INVALID	599