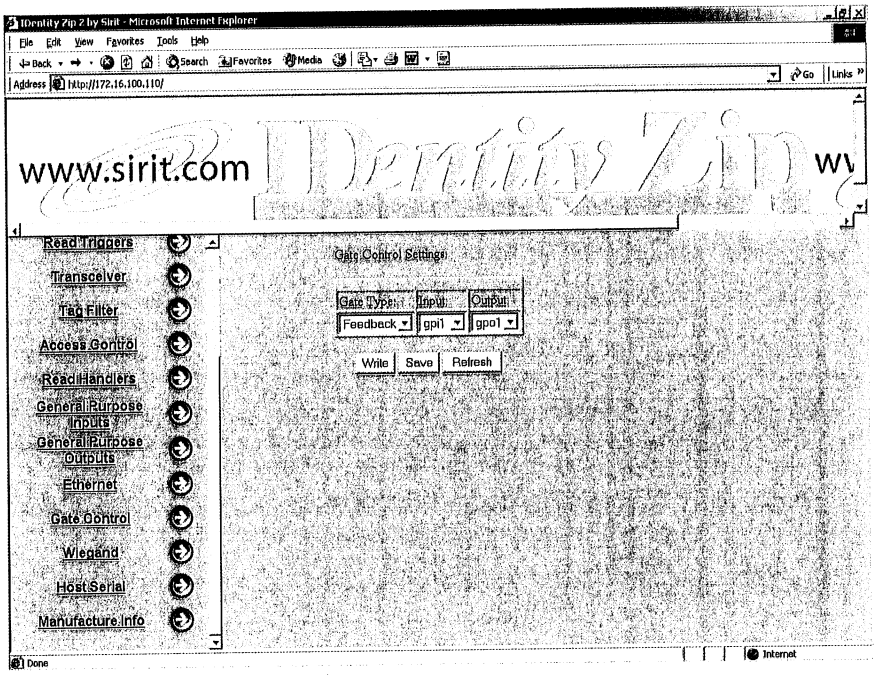
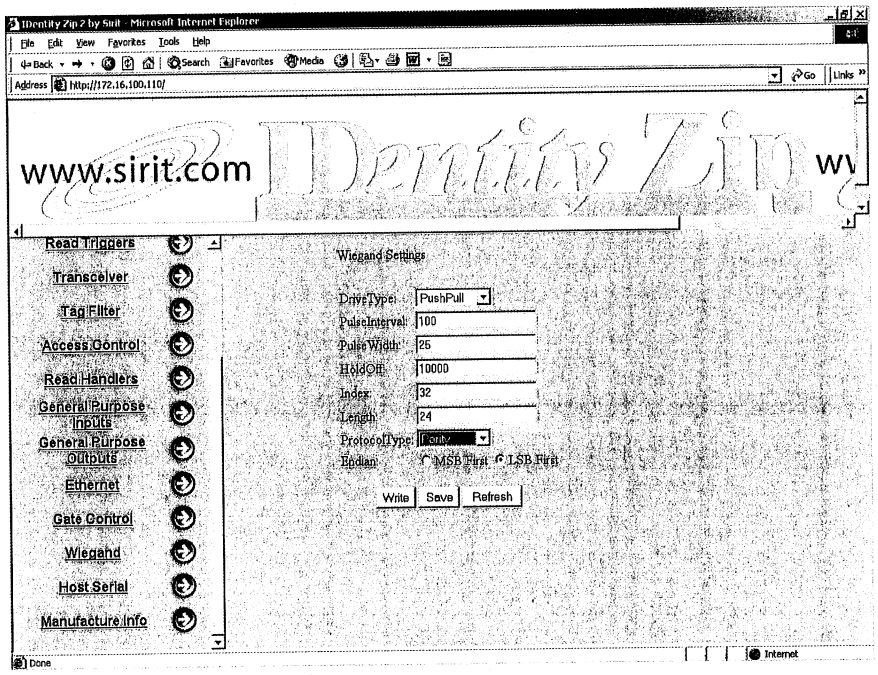


5.13 Wiegand Port Configuration

The Wiegand port is a simplex type communication compliant to the SIA 26-bit standard as well as extended formats.

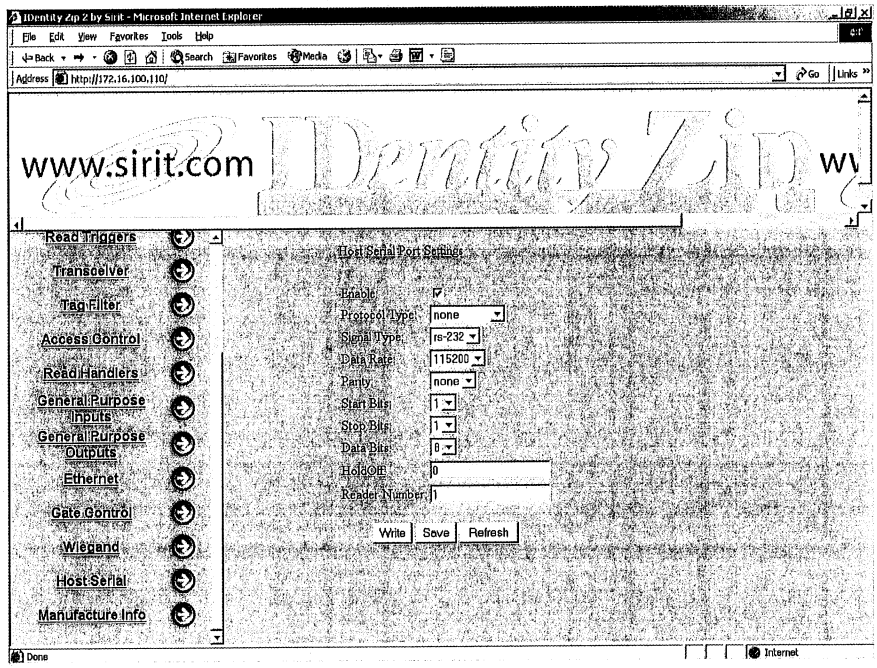


Transmission on the Wiegand interface can only occur when a corresponding Host Upload Event has been configured and satisfied. As

before, the pulse width is the time that a data pulse is active for valid data and the pulse interval is the time between those pulses. Index is the offset (in bits) within the translated tag data where the serialization information will begin. Length specifies the length of the data packet, not including any protocol overhead. Both Parity and Checksum protocols are supported in addition to raw data format. Endian type denotes whether most significant (Big) or least significant (Little) will be transmitted first.

5.14 Host Serial

The serial port supports a variety of protocols and signaling.



To enable the host port, check the enable box. This serial port supports WPS and ACOM protocols, as well as RS-232, RS-422, and RS-485 signaling. To change any of the parameters, simply choose the value from the pull down menus. Hold off time refers to the time that is inserted between output data.

6.0 XML Command Set

XML Remote Procedure Calls (RPC) are used to remotely manipulate parameters on the ID*entity* MaX reader. These calls may be executed using the HTTP POST command issued to the reader device web-server. The XML command set is detailed in this section.

6.1 Get Parameter

When executed, this command will return the requested parameter value. One or more parameter names may be requested. The number of parameters is limited by overall document size limitation.

Command:

```
<methodName>GetParam</methodName>
<params>
        <param>
                <name><string> System.Time</string></name>
                <value><dateTime.iso8601>20050717T14:08:55</dateTime.iso8601></vlaue>
        </param>
</params>
```

Response:

```
<methodResponse>
        <params>
                <param>
                        <name><string>System.Time</string></name>
                        <value><>
                </param>
                <param>
                        <name><string>ReadTrigger.Enable</string></name>
                        <value>
                                <struct>
                                        <member>
                                                <name><string>Index</string></name>
                                                <value><i4>0</i4></value>
                                        </member>
                                </struct>
                        </value>
                </param>
        </params>
</methodResponse>
```

6.2 Set Parameter

When executed, this command will set the requested parameter.

Command:

```
<methodName>SetParam</methodName>
<params>
        <param>
                <name><string>System.Time</string></name>
                <value><dateTime.iso8601>20050717T14:08:55</dateTime.iso8601></vlaue>
        </param>
</params>
```

Response:

```
<methodResponse>
        <params>
                <param>
                        <name><string>System.Time</string></name>
                        <value><dateTime.iso8601>20050717T14:08:55</dateTime.iso8601></value>
```

```
                    </param>
            </params>
</methodResponse>
```

6.3 Parameter Summary

Each of the following parameters may be configured using the Get and/or Set Parameter commands.

Field Name	Attrib.	Type	Description	Notes
Firmware.Name	R	string	Name of firmware. Ex) "Integrated Zip Reader"	
Firmware.MajorVer	R	i4	Major release version.	
Firmware.MinorVer	R	i4	Minor release version. I.e. updates	
Firmware.BuildVer	R	i4	Internal development version.	
System.Time	RW	Iso8601	Current reader date/time.	
ReadTrigger.Enable	RW	boolean	1 = Read trigger is enabled, 0 = Disabled	2,3
ReadTrigger.Input	RW	string	GPI# = A General Purpose Input is selected.	2
ReadTrigger.Output	RW	string	GPO# = The General Purpose Output to be used.; Blank = no discrete output.	2
ReadTrigger.Duration	RW	i4	Minimum amount of time the transceiver will attempt to interrogate the RF field.	2
ReadTrigger.Interval	RW	i4	Amount of time between internal timer input assertions.	
ReadTrigger.Retrigger	RW	boolean	1 = Read Duration restarts each time an active input is detected, 0 = Duration doesn't re-start.	2,3
Transceiver.TagType	RW	array[string]	"EPC Class 0", "EPC Class 1", "EPC Class 1 Gen 2", "Identity Flex", "Identity Zip", "Title 21" and/or "iso18000-6".	
ReadFilter.ValidationCount	RW	i4	Number of times a Tag ID must be received before being processed. 1 = First Id processed.	
ReadFilter.ValidationAttempts	RW	i4	Number of times a Tag ID verification may be attempted in attaining the required ValidationCount.	
ReadFilter.DuplicateFilterEnable	RW	boolean	1 = Duplicate Tag ID's are discarded, 0 = Duplicate ID's are not discarded.	3
ReadFilter.DuplicateTimeout	RW	i4	Minimum amount of time, in seconds, before allowing a duplicate Tag ID to be processed.	
ReadFilter.PassbackEnable	RW	boolean	1 = Tag ID timeout is re-started if a duplicate Tag ID is found within the original timeout period; 0 = Timeout is not changed/updated.	3
AccessControl.Enable	RW	boolean	1 = Tag ID is used to reference table entries to determine action.; 0 = All tag ID's are discarded.	3
AccessControl.TranslationDefault	RW	string	The translation ID to be used when a table lookup fails.	
AccessControl.HandlerIDDefault	RW	i4	The Handler ID to be used when a table lookup fails. A value of '-1' disables this feature.	
AccessControl.DealerHandlerID	RW	I4	Indicates the handler to be used when access is granted based upon Dealer Code. A value of '-1' disables this feature.	
AccessControl.DealerCode	RW	I4	If set, this code must match the tags dealer code value. A value of '-1' disables this feature.	
AccessControl.FacilityCode	RW	I4	If set, this code must match the tags facility code value. A Dealer Code must also be present for this value to be used. A value of '-1' disables this feature.	

ReadHandler.Description	RW	string	User-configured string describing the action. Ex) "Grant Access - Green Light" or "Deny Access - Red Light"	2
ReadHandler.Output	RW	array[string]	Array of up to 8 items referencing the desired output module; GPO#, WIEGAND, etc.,.	2
GateControl.GateType	RW	string	"Normal" or "Feedback"	
GateControl.Input	RW	string	GPI# = The General Purpose Input to monitor for gate feedback.	
GateControl.Output	RW	string	GPO# = The General Purpose Output to signal for gate control.	
HostSerial.Enable	RW	boolean	1 = Serial module is enabled, 0 = Serial module is disabled.	3
HostSerial.ProtocolType	RW	string	WPS, ASCOM	
HostSerial.SignalType	RW	string	RS-232, RS-422, RS-485	
HostSerial.DataRate	RW	i4	Numeric data rate. Ex) 115200, 57600	
HostSerial.Parity	RW	string	Parity bit type preceeding each byte transmitted/received; Ex) None, Odd, Even	
HostSerial.StartBits	RW	i4	Number of data '1' start bits to preceed each data byte transmitted.	
HostSerial.StopBits	RW	i4	Number of data '0' stop bits to follow each data byte transmitted.	
HostSerial.DataBits	RW	i4	Number of data bits per byte transmitted.	
HostSerial.HoldOff	RW	i4	Amount of time between any two packets, in milli-seconds.	
HostSerial.ReaderNum	RW	I4	Reader number used for calculating lane number in some host protocols. Valid values are 1-64.	
Wiegand.DriveType	RW	string	"none", "OpenDrain" or "PushPull"	
Wiegand.PulseInterval	RW	i4	Amount of time, in milliseconds, between output pulses.	
Wiegand.PulseWidth	RW	i4	Amount of time, in milliseconds, per pulse.	
Wiegand.HoldOff	RW	i4	Amount of time between any two packets, in milli-seconds.	
Wiegand.Index	RW	i4	Indicates the number of least significant translated data bits to disregard.	
Wiegand.Length	RW	i4	Indicates the number of data bits to transmit regardless of index.	
Wiegand.ProtocolType	RW	i4	"none", "parity" or "checksum"	
Wiegand.Endian	RW	boolean	1 = LSB first, 0 = MSB first.	3
GPI.ActiveState	RW	string	"ActiveLow", "ActiveHigh", "RisingEdge", "FallingEdge"	2
GPI.PulseWidth	RW	I4	Input de-glitch timeout in milliseconds. Maximum value of 1 second.	
GPI.Value	R	boolean	1 = Input high, 0 = Input Low	
GPO.DriveType	RW	string	"none", "OpenDrain" or "PushPull"	2
GPO.ActiveState	RW	boolean	1 = Active High, 0 = Active Low	2,3
GPO.PulseWidth	RW	i4	Active state pulse width, in milliseconds.	2
GPO.PulsePeriod	RW	i4	Minimum inactive state period between pulses, in milliseconds.	2
GPO.PulseRepeat	RW	i4	Number of pulse iterations per outputs.	2
Ethernet.Enable	RW	boolean	1 = Ethernet I/O is enabled, 1 = Ethernet is disabled	3
Ethernet.DefaultIP	RW	string	Default IP address in octet format. Ex)192.168.0.4	
Ethernet.DefaultMask	RW	string	Default subnet mask in octet format. Ex)255.255.255.0	
Ethernet.DefaultGateway	RW	string	Default subnet mask in octet format. Ex)192.168.0.1	
Ethernet.IP	RW	string	Default IP address in octet format. Ex)192.168.0.4	5
Ethernet.Mask	RW	string	Default subnet mask in octet format. Ex)255.255.255.0	5
Ethernet.Gateway	RW	string	Default subnet mask in octet format. Ex)192.168.0.1	5
Ethernet.DHCPEnable	RW	boolean	1 = DHCP is enabled, 0 = DHCP is disabled.	3
Ethernet.POSTEnable	RW	boolean	1 = HTTP POST command is enabled, 0 = POST is disabled.	3

Translation.IndexStored	R	boolean	True = Index backup is up-to-date, False = Index has not been stored.	3
Esn.Nomenclature	RW	string	Manufacturing Nomenclature. ex)"IntegratedZip"	4
Esn.BoardVer	RW	String	Manufacturing Board Version. ex)"2.2.3"	4
Esn.Serial	RW	String	Manufacturing Serial Number. ex)"0123456789"	4
Esn.TestDate	RW	String	Manufacturing Test Date. Format: MM/DD/YYYY	4
Esn.ManufactureCode	RW	String	Manufacturing Code. Ex)"0123"	4
Esn.MacAddress	RW	String	Manufacturing assigned unique Ethernet HW address. Ex)"001122334455"	4

Note	Description
1	Field instance must be specified with Tag ID (string type) as an index.
2	Field instance must be specified with a numeric index value appended to the field name. Ex) "GPI.ActiveState0". The default index assumed is zero if none is specified. Ex) "GPI.ActiveState"
3	For 'boolean' fields, '0' = False and '1' = True
4	Fields are read-write to RAM address space. Storage of values to non-volatile memory require usage of the 'StoreEsn' command.
5	Run-time parameter only, variable is not stored in ROM at any point

6.4 Save Configuration

This method will commit configuration parameters to non-volatile memory.

Command:

```
<methodName>SaveConfigurationParams</methodName>
```

Response:

```
<methodResponse>
        <params>
                <param>
                        <name><string>StatusCode</string></name>
                        <value><i4>1</i4></value>
                </param>
        </param>
</methodResponse>
```

6.5 Load Configuration

This method will restore configuration parameters from non-volatile memory.

Command:

```
<methodName><string>LoadConfigurationParams</string></methodName>
```

Response:

```
<methodResponse>
        <params>
                <param>
                        <name><string>StatusCode</string></name>
                        <value><i4>1</i4></value>
                </param>
        </param>
</methodResponse>
```

6.6 Read Translation Table Entries

This method will read one or more records from the translation table. The specified Tag ID parameter will indicate the first record to be read. The Count parameter indicates the requested number of Tag ID records to read in ascending order. The actual number of records returned is limited by the number of ascending entries available and maximum file length. If the specified Tag ID is "00", the lowest numeric value ID will be the first record returned. If the specified Tag ID cannot be found in the translation table, the next available Tag ID will be returned first based on ascending numeric Tag ID value. A method response without any 'struct' entries indicates no records were available to be read.

Each entry within the translation table is unique based upon Tag ID. Therefore, to sequentially read a table that exceeds the maximum individual file length, this command should be executed multiple times with the specified Tag ID equal to the last ID read plus 1.

Command:

```
<methodName>ReadTranslationRecords</methodName>
<params>
        <param>
                <name><string>TagID</string></name>
                <value><string>1</string></value>
        </param>
        <param>
                <name><string>Count</string></name>
                <value><i4>1<i4></value>
        </param>
</params>
```

Response:

```
<methodResponse>
        <structs>
                <struct>
                        <member>
                                <name><string>TagID</string></name>
                                <value><string>01234567890ABCDEF</string></value>
                        </member>
```

```
                    <member>
                        <name><string>TranslationID</string></name>
                        <value><string>0011002200330044</string></value>
                    </member>
                    <member>
                        <name><string>HandlerID</string></name>
                        <value><i4>2</i4></value>
                    </member>
                </struct>
                .............
                <struct>
                    <member>
                        <name><string>TagID</string></name>
                        <value><string>1100220033004400</string></value>
                    </member>
                    <member>
                        <name><string>TranslationID</string></name>
                        <value><string>0011002200330044</string></value>
                    </member>
                    <member>
                        <name><string>HandlerID</string></name>
                        <value><i4>2</i4></value>
                    </member>
                </struct>
        </structs>
</methodResponse>
```

6.7 Write Translation Table Entries

This method will write translation table entries to the internal reader translation table. For optimum indexing speed, entries should be pre-sorted in ascending order according to Tag ID. If a large number of existing Tag IDs are to be uploaded, the existing table should be deleted and new sorted table uploaded for optimal upload performance. Once translation tables and/or entries are written, the Store Index command should be executed to store the indexing updates and facilitate quicker system reboot time.

Command:

```
<methodName>WriteTranslationRecords</methodName>
        <structs>
                <struct>
                    <member>
                        <name><string>TagID</string></name>
                        <value><string>01234567890ABCDEF</string></value>
                    </member>
                    <member>
                        <name><string>TranslationID</string></name>
                        <value><string>0011002200330044</string></value>
                    </member>
                    <member>
                        <name><string>HandlerID</string></name>
                        <value><i4>2</i4></value>
                    </member>
                </struct>
                .............
                <struct>
                    <member>
                        <name><string>TagID</string></name>
```

```
                        <value><string>1100220033004400</string></value>
                    </member>
                    <member>
                        <name><string>TranslationID</string></name>
                        <value><string>0011002200330044</string></value>
                    </member>
                    <member>
                        <name><string>HandlerID</string></name>
                        <value><i4>2</i4></value>
                    </member>
                </struct>
        </structs>
</methodName >
```

Response:

```
<methodResponse>
        <params>
                <param>
                        <name><string>Count</string></name>
                        <value><i4>0</i4></value>
                </param>
        </params>
</methodResponse>
```

6.8 Delete Translation Table Entries

This method will delete one or more translation table entries starting with the specified Tag ID and continuing in ascending Tag ID order until the Count value is reached or the translation table is empty. The number of entries deleted is returned.

If the ExactMatch parameter is set to '1', the specified TagId must exist within the Translation Table. Otherwise, TagId's greater than the specified Id will be deleted. To delete all records within the Translation Table regardless of Id, specify a TagId of "", Count of of '-1' and ExactMatch of '0'.

Command:

```
<methodName>DeleteTranslationRecords</methodName>
<params>
        <param>
                <name><string>TagID</string></name>
                <value><string>1</string></value>
        </param>
        <param>
                <name><string>Count</string></name>
                <value><i4>1<i4></value>
        </param>
        <param>
                <name><string>ExactMatch</string></name>
                <value><boolean>1<boolean></value>
        </param>
</params>
```

Response:

```
<methodResponse>
        <params>
                <param>
                        <name><string>Count</string></name>
                        <value><i4>1<i4></value>
                </param>
        </params>
</methodResponse>
```

6.9 Store Translation Table Index

This method will commit the RAM based translation table indexes to FLASH memory. If an up-to-date index table has already been transferred to FLASH memory, it will not be re-written.

Command:

```
<methodName>StoreTranslationIndex</methodName>
```

Response:

```
<methodResponse>
        <params>
                <param>
                        <name><string>StatusCode</string></name>
                        <value><i4>0</i4></valuelean>
                </param>
        </params>
</methodResponse>
```

6.10 Read Transaction Log

This method will read one or more records from the transaction log. The first execution of this command should specify the 'ResetStart' as '1' (True) to mark the extraction start point. Subsequent execution(s) of this command will sequentially return older remaining entries provided the 'ResetStart' parameter is specified as '0'. This results in the most recent transaction log entries being extracted first.

Command:

```
<methodName>ReadTransactionRecords</methodName>
<params>
        <param>
                <name><string>ResetStart</string></name>
```

```
                <value><boolean>1</boolean></value>
        </param>
        <param>
                <name><string>MaxCount</string></name>
                <value><i4>2</i4></value>
        </param>
</params>
```

Response:

```
<methodResponse>
        <structs>
                <struct>
                        <member>
                                <name><string>TagID</string></name>
                                <value><string>0011002200330044</string></value>
                        </member>
                        <member>
                                <name><string>Time</string></name>
                                <value><dateTime.iso8601>20050717T14:08:55</dateTime.iso8601>
                                </value>
                        </member>
                        <member>
                                <name><string>HandlerRec</string></name>
                                <value><i4>2</i4></value>
                        </member>
                </struct>
                .....
                <struct>
                        <member>
                                <name><string>TagID</string></name>
                                <value><string>0011002200330044</string></value>
                        </member>
                        <member>
                                <name><string>Time</string></name>
                                <value><dateTime.iso8601>20050717T14:08:56</dateTime.iso8601>
                                </value>
                        </member>
                        <member>
                                <name><string>HandlerRec</string></name>
                                <value><i4>2</i4></value>
                        </member>
                </struct>
        </structs>
</methodResponse>
```

6.11 Upload Firmware

This method is used to buffer firmware on the device for upgrade. Multiple methods may be executed to load an entire firmware image. A status code response is generated indicating whether or not parameters are formatted correctly and expected values were found.

Command:

```
<methodName>UploadFirmware</methodName>
<params>
        <param>
                <name>Format</name>
                <value>extendedIntelHex</value>
```

```
        </param>
        <param>
                <name>RecordIndex</name>
                <value><i4>0</i4></value>
        </param>
        <param>
                <name>TotalRecords</name>
                <value><i4>3</i4></value>
        </param>
        <param>
                <name>TargetCode</name>
                <value><i4>1234</i4></value>
        </param>
        <param>
                <name>DataRecords</name>
                <value><array><data>
                        <value>:020000040002F8</value>
                        <value>:1000000000000FE11F00C0E3120080E300F021E1D7</value>
                        ...
                        <value>:00000001FF</value>
                </data></array></value>
        </param>
</params>
```

Response:

```
<methodResponse>
        <params>
                <param>
                        <name><string>StatusCode</string></name>
                        <value><i4>0</i4></value>
                </param>
        </param>
</methodResponse>
```

6.12 Execute Firmware

This method initiates internal transfer/upgrade of firmware from after buffers have been successfully loaded using the Upload Firmware command. Following execution of this command, two seconds will be given before the device goes off-line & begins the firmware transfer.

Command:

```
<methodName><string>ExecuteFirmwareLoad</string></methodName>
```

Response:

```
<methodResponse>
        <params>
                <param>
                        <name><string>StatusCode</string></name>
                        <value><i4>0</i4></value>
                </param>
        </param>
</methodResponse>
```

6.14 Example XML-RPC transaction

```
POST /RPC2 HTTP/1.0
User-Agent: ZipReader
Host: ZipReader.sirit.com
Content-Type: text/xml
Content-length: 181

<?xml version="1.0"?>
<methodCall>
        <methodName>GetParam</methodName>
        <params>
                <param>
                        <name><string>Firmware.Name</string></name>
                </param>
                <param>
                        <name><string>Firmware.MajorVer</string></name>
                </param>
                <param>
                        <name><string>Firmware.MinorVer</string></name>
                </param>
        </params>
</methodCall>
```

An example XML-RPC response is as follows:

```
HTTP/1.1 200 OK
Connection: close
Content-Length: 158
Content-Type: text/xml
Date: Fri, 17 Jul 2005 19:55:08 GMT
Server: ZipReader

<?xml version="1.0"?>
<methodResponse>
        <params>
                <param>
                        <name><string>Firmware.Name</string></name>
                        <value><i4>1</i4></value>
                </param>
                <param>
                        <name><string> Firmware.MajorVer</string></name>
                        <value><i4>1</i4></value>
                </param>
                <param>
                        <name><string>Firmware.MinorVer</string></name>
                        <value><i4>2</i4></value>
                </param>
        </params>
</methodResponse>
```

Appendix A

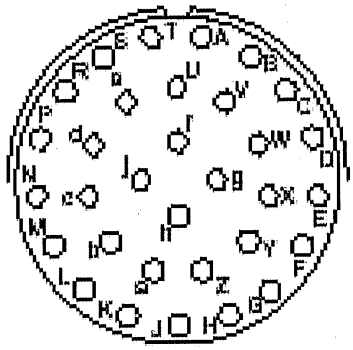
Default Values for Configuration Parameters

Parameter	Default Value	Range
Date and Time	US Central (GMT-6)	
GPI		
Active state	Low	
Pulse width	100 ms	
GPO		
Active state	Low	
Drive type	Push pull	
Pulse width	25 ms	1 – 999 ms
Pulse period	50 ms	1 – 999 ms
Pulse repeat	0	
Serial Port		
Port enable	Enabled	
Protocol	None	
Signal type	RS-232	
Baud rate	115200	9600 - 115200
Parity	None	
Start bit	1	
Stop bit	1	1 - 2
Data bits	8	5 - 8
Hold off	20 ms	0 – 9999 ms
Wiegand Port		
Hold off	10 ms	0.1 - 1000 ms
Drive type	Push pull	
Pulse width	25 us	20 – 100 us
Pulse Interval	100 us	100 – 20000 us
Index	24	
Length	32	24 - 128
Protocol	Parity	
Endian type	Little	
Access Control		

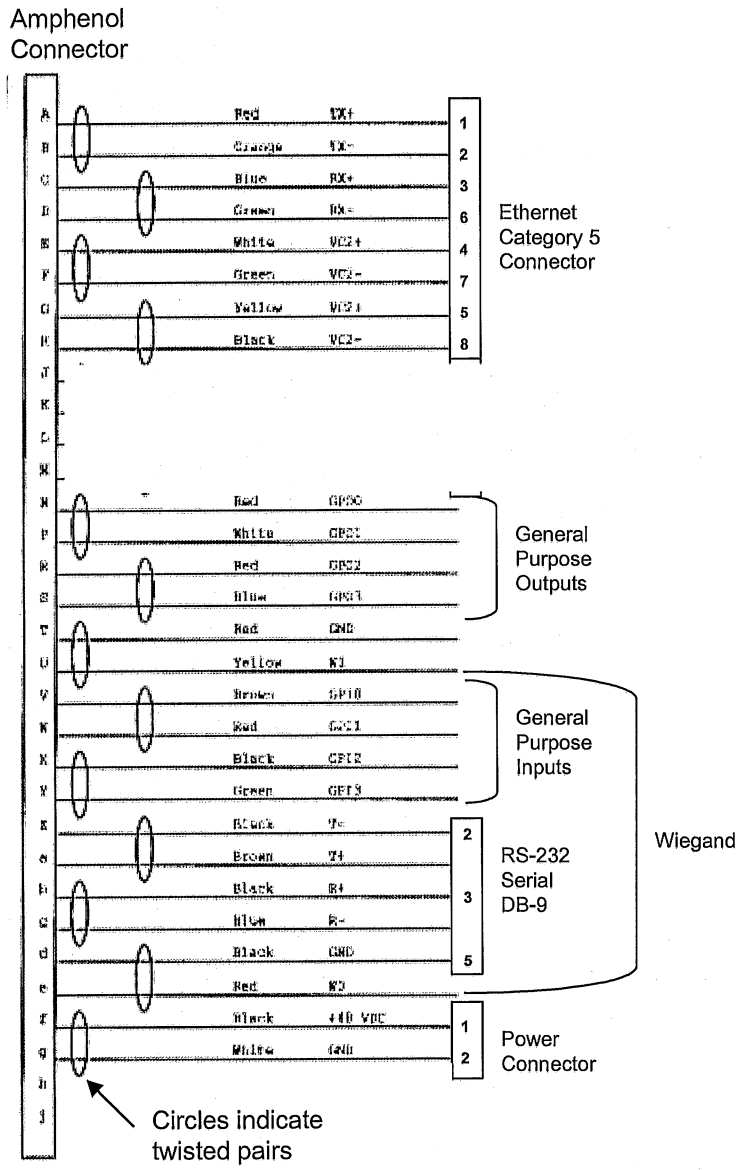
Control enable	Disabled	
Default Translation ID	FFFFFFFFFFFFFFFF	
Handler ID	0	0 - 8
Dealer handler ID	0	0 - 8
Dealer code	Not used	
Facility code	Not used	
Gate Control		
Gate type	Normal	
Input	None	GPI0 – GPI3
Output	None	GPO0 – GPO3
Trigger Settings		
Trigger enable	Enabled	
Trigger input	T0	GPI0 – GPI3
Trigger output	None	GPO0 – GPO3
Trigger duration	100 ms	
Retrigger	Enabled	
Filter		
Duplicate Filter Enable	Enabled	
Filter timeout	2 sec	1 – 255 sec
Validation count	1	1 - 255
Validation attempts	5	1 - 255
Passback	Disabled	
Ethernet		
Enabled	Enabled	
DHCP Enabled	Enabled	
POST Enabled	Enabled	
Current ip	172.18.27.231	
Currnet subnet mask	255.255.255.0	
Current gateway	172.18.27.254	
Default ip	172.18.27.231	
Default subnet mask	255.255.255.0	
Default gateway	172.18.27.254	

Appendix B

Interface Cable Pinout



Amphenol connector pinout
(rear face of socket insert shown)



Interface cable and connector
pin assignments