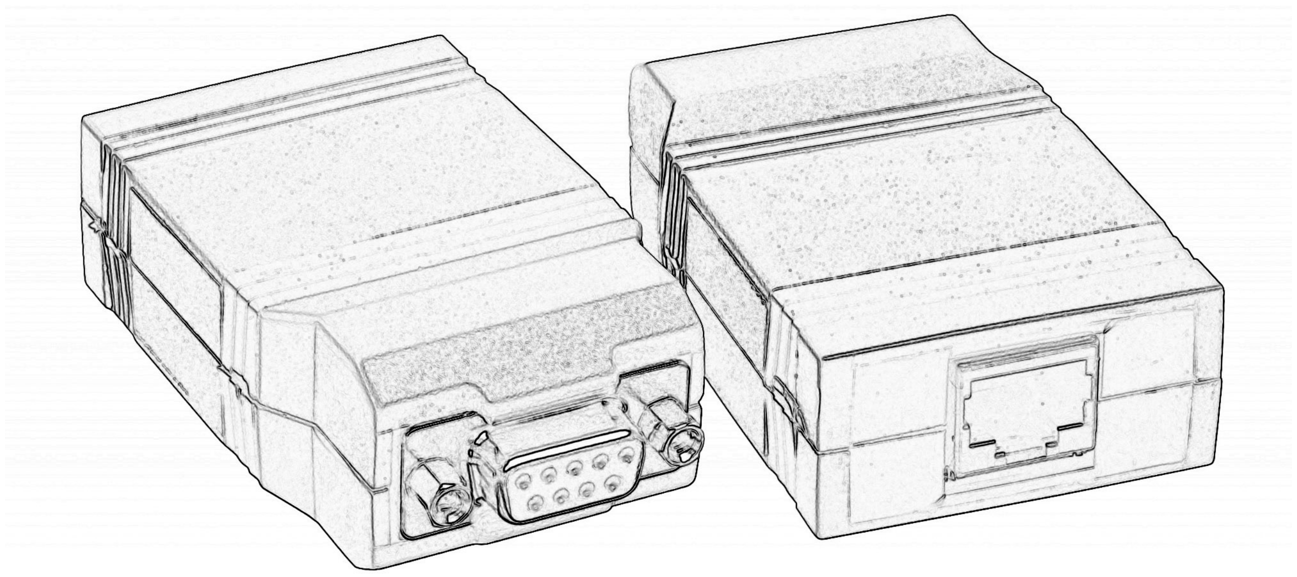
International Bar Code



716/718 Converter User's Guide V 1.96

27 MAY 08

This manual provides information on how to setup, program, and interface the Model 716 and 718 converters.

Revision pages of this manual are marked in the lower center of each page, noting the current revision level, and revision date.

Any errors or omissions from this manual should be noted and sent to the Technical Services dept. of IBC for correction.

If you need any additional information concerning ibc products, contact IBC's Technical Support department from 08:00 to 17:00 Eastern Standard Time, at 860-659-9660, or e-mail us (support@interbar.com). Technical information and update information is also available on the internet at our home page http://interbar.com.

IBC Document ID:	UG716
Version:	1.96
Revision:	0.0
Date:	27 MAY 08

Contents

Notice for 718 Converters

Please note that all references in this manual to the 716 converter also refer to the 718 converter. The only difference between the 716 and 718 converters is that the 716 converter contains an rs232 interface, while the 718 converter contains a usb interface.

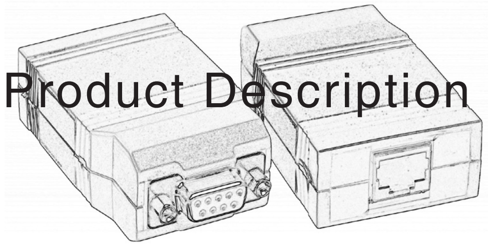
All references to rs232 inputs, outputs, and conversions also apply to usb inputs, outputs, and conversions for the model 718.

The 718 converter is normally set up to take it's power from the reader side, not the usb port. An internal jumper can be used to power the converter from the usb side. Note that you cannot power the converter from both sides. Either power the converter from the reader side, or the usb side, but not both. If the jumper is ON, then do not power from the reader side. If the jumper is OFF, then power from the reader side.

In order to operate IBC usb equipment, a special usb driver is required. This driver can be found on the IBC web page under the software section. This driver sets up a virtual comm port on the pc so that you can communicate with the 718 converter serially through a standard pc comm port.

Users familiar with the 718R product will know that the 718R converter, like the 716R converter, can utilize wiegand/aba inputs only, and no wiegand/aba outputs, since there are not enough pins on the RJ45 jack for wiegand/aba inputs, outputs, and the 3-wire relay form C connections. 718R converters, however, manufactured after June 2008 have an additional set of jumpers, for configuring the 718R to utilize wiegand/aba inputs, or outputs. This allows the 718R to be used in oem configurations where the primary use is a usb (serial) to wiegand conversion, but it is also convenient to have a relay on board as well.

Product Description



The 716 converter is a universal data converter which can be used for converting between different data formats which are customarily used in the access control industry. Supported formats include:

- Wiegand
- Magstripe (including f2f)
- Wand Emulation
- Rs232

There are two input ports and two output ports. Each port can be individually programmed for the specific type of conversion that you need.

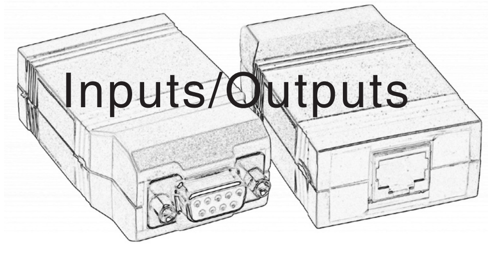
Programming is acheived by sending serial commands to the converter. IBC also provides a utility software program for programming the converter.

Multiple conversions are allowed which means that the converter is not limited to one-direction traffic only. You can, for example, use the converter "in-line" to convert wiegand to rs232, and then convert rs232 back to wiegand. Not only are multiple conversions allowed, but they can operate asynchronously (independant of each other).

The 716 is flash downloadable through the serial port. This allows for easy updates of program code in the field.

Power requirements for the 716 are 12VDC. The 716-5 runs on 5VDC, and the 716C runs on variable voltage 5VDC through 12VDC.

Inputs/Outputs



There are 2 input ports and 2 output ports. These ports are:

- Input 1 (wiegand and mag)
- Input 2 (rs232, or usb for 718)
- Output 1(wiegand, mag, wand, f2f)
- Output 2 (rs232, or usb for 718)

Additionally, there is an output line which can be used either as an led control, or for magstripe f2f output.

Input 1 will accept wiegand data up to 250 bits and magstripe track 1 or track 2 data. Input 2 is an rs232 (or usb) serial port which accepts incoming serial data, including data to be converted, and programming commands.

Output 1 is a driven open-collector port which can output standard wiegand (up to 250 bits), alphanumeric wiegand, aba wiegand, magstripe (track 1 or 2) emulation, and wand emulation data.

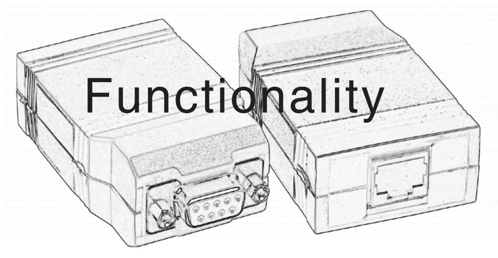
Output 2 is an rs232 (or usb) serial port used for outputting converted data serially.

The f2f output line is a non-driven open collector output, which requires a pull-up in the receiving equipment. The pull-up can be to 5V or 12V.

The Rs232 serial (716) inputs and outputs are located on a db9 connector for easy connection to a pc. The usb serial (718) input and output Is located on a usb type B connector. The wiegand, magstripe, and wand connections are located on an RJ45 connector, allowing for easy disconnect of the converter when required.

Power and ground can be supplied on the RJ45 connection, or on the serial side.

There is also an led control line on the RJ45 connector which can be controlled using serial commands.



This is a complete list of the types of **data conversions** which are available in the 716:

- Wiegand to wiegand
- Wiegand to magstripe & f2f
- Wiegand to wand emulation
- Wiegand to rs232
- Magstripe to wiegand
- Magstripe to magstripe & f2f
- Magstripe to wand emulation
- Magstripe to rs232
- Rs232 to wiegand
- Rs232 to magstripe&f2f
- Rs232 to wand emulation
- Rs232 to RS232 with masking

The 716 also supports alphanumeric wiegand and wieaba wiegand formats for inputs and outputs. Magnetic stripe data (input and output) can be either 5 bit characters (track 2,3 aba) or 7 bit characters (track 1 character set). Additionally, f2f output is supported with both the track 1 character set and the track 2 character set.

Each input can be programmed to convert data and then send that data out one of the outputs.

To convert data, you must program the converter to tell it what it is you are reading (the input), how you want to convert it, and then where to send it out. The converter is shipped standard with certain defaults pre-programmed. These defaults are listed in appendix A.

Each input in the converter operates asynchronously. This means that the converter can accept an input from multiple inputs at one time. The converter will, however, process each input conversion on a first-come, first-serve basis, and therefore output only one conversion at a time. Any other inputs which are currently active will still be processed, but they will not be outputted until the first transmission is complete.

For example, you might connect a proximity reader with a wiegand output to input # 1, and send serial data to the converter on the serial port, at the same time. Both of them can be done at the same time, and the converter will process and convert the data from input # 1 first, and then from the serial port.

Once the converter receives data from one of it's inputs, it will not process any more data from that input until the data conversion is complete.

Each input is programmed independently of the other, and can be programmed for one specific conversion only. As a general rule, you program the converter to take one specific input and send it to one specific output.

Each input can be masked and reformatted prior to conversion and transmission. In order to do this, all inputs which are not character in nature, such as wiegand, are decimalized prior to masking and conversion. This means that for wiegand inputs, you will need to know the structure of the wiegand input in order to perform a conversion. The converter cannot convert wiegand data unless it knows what the structure is. The same applies to magnetic stripe data. The converter will need to know which magstripe track you are reading, because the character sets are different.

It is also possible to connect two physical readers to the same input port, as long as they are open-collector output. Please note that you can define only one output conversion type and one mask sequence for both wiegand and magstripe inputs.

There are certain limitations on the field sizes for both inputs and outputs. It is necessary to understand these limitations prior to programming the converter for a specific conversion.

These limitations are:

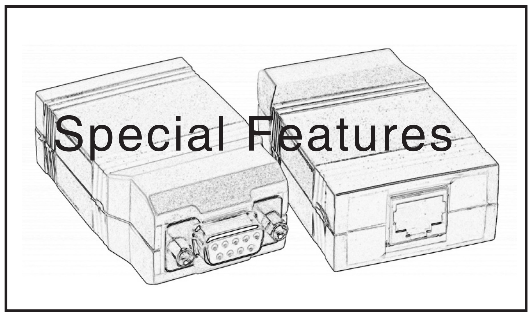
Standard wiegand Input	250 bits max, 64 bits per field
Standard wiegand Output	250 bits max, 64 bits per field
Serial input/output	45 characters
Magstripe input/output	80 characters
Alphanumeric wiegand	80 characters
Aba wiegand	80 characters

Versions 1.4 and later allow for the programming of 2 different wiegand input formats, with separate masking.

Version 1.94 and later allow for the programming of 3 different wiegand input formats.

Versions 1.7 and later allows for serial-to-serial conversion, with masking.

Versions 1.8 and later allow you to select certain pre-processing functions to be performed on the data prior to conversion. Pre-processing functions include hexadecimal conversions, Amtech toll tag conversions, and a serial alpha substitution mode.



There are a number of special features which the converter has which provide huge flexibility in converting data. A few of these features are shown below.

Programmable input terminator

The input terminator for the serial connection is programmable. You can change it to any ascii character, allowing you to convert serial streams that do not end in a carriage return which is the standard terminator.

Character masking

All inputs can be masked with either character deletion or insertion.

Individual wiegand field support

The wiegand format supported by the converter includes not only the site code and id code but also issue codes. Each of these fields can be individually masked, transposed, and converted to form a new number for output transmission.

Wiegand msb first and lsb first support

Some manufacturer's controllers require wiegand bit formats with lsb first as opposed to the standard msb first. The 716 supports both of these formats.

Programmable wiegand parity

The converter can insert standard parity or a calculated parity based on a user mask.

Line feed character deletion

A line feed character can be optionally ignored on the serial input.

Wiegand constant field support

Some wiegand formats require a certain number of bits to be static in the format. The 716 supports the programming of these bits.

Wiegand field override support

Any of the wiegand fields can be overridden with a constant value.

Magstripe C start support

The magnetic stripe conversion allows for both the standard B start on track 2 as well as a C start, on both the input side and output (converted) side.

Input stream dumping

The converter optionally can "dump" any received data as a serial stream of characters with each character representing an input bit. Input streams can easily be analyzed this way.

The "dump" feature of the 716 is useful for debugging and identifying card formats. The wiegand/magstripe input can be programmed to dump received data through the rs232 connection. Using this feature, you can actually look at the bit structure of wiegand, proximity, and magstripe cards. This is useful when attempting to identify the format of cards for conversion.

Version 1.8 and later allow the input streams to be dumped using different formats, including bit-by-bit transmission, and bit compression into either 8-bit ascii characters, or hexadecimal chatacters.

Programmable pulse times

Output pulses for wiegand and magnetic stripe are programmable. Both the pulse width and timing between pulses are programmable. Pulse timing for f2f signals are fixed at 1ms per 0 bit transition.

Magstripe alpha deletion

Alphabetic characters can optionally be deleted from Track 1 magstripe input streams.

Legacy wiegand card support

In instances where the 716 is used to read legacy wiegand cards, the 716 can be programmed to perform reverse read detection. Some older wiegand readers will actually transmit wiegand data if the card is read backwards, causing misreads in many systems which do not perform parity calculations. The 716 can be programmed to detect this condition, and automatically reverse the field for proper decoding. This applies to 26 bit and 28 bit cards only.

Parity generation

The 716 also contains powerful features for parity generation. Parity can be constructed in the traditional way (consecutive bits for both left and right parity), and it also can be constructed using parity *masks*, where you specify exactly which bits will be used for parity and what the algorithm is. Traditional parity bits can be stripped from input streams if you wish, and then replaced with your own algorithm if you need to do this.

Multiple field support

Wiegand and Proximity cards normally have different data items encoded in them. These items, or fields, usualy contain what is known as a site code (also know as facility code), and an ID number (also known as badge number). In some cases, there may be additional data such as an issue code, some constant data, and parity data. The 716 can be programmed to treat each of these fields differently, and *override or replace* any field. This can be done both at the bit level and also at the decimalized level.

Serial-to-serial masking

Serial RS232 input data can be masked, reformatted, and then retransmitted out the serial port.

Preprocessing

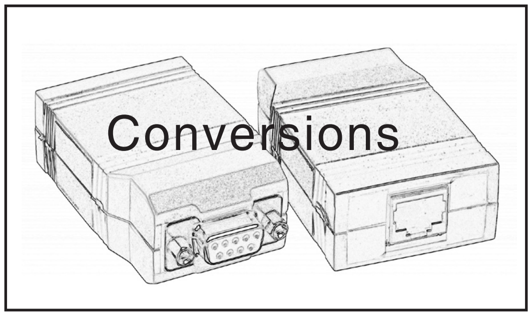
Version 1.8 and later allows you to use one of the predefined preprocessing options on input data. The preprocessing options can be selectively turned on or off. When turned on, these options preprocess the input data and then replace the input data with the result, prior to performing the conversion.

Presently, there are 3 preprocessing functions which can be used in the converter. More preprocessing functions will be added in the future. The 3 current preprocessors are:

- Hexadecimal input conversion
- Amtech 26 bit encoded format conversion
- Serial alpha substitution conversion

The hexadecimal preprocessor interprets serial input data as hexadecimal characters, and then reformats that data into a serial data stream representative of the value of the hexadecimal characters. The Amtech 26 bit encoded preprocessors autodiscriminates Amtech 26 bit encoded data transmissions, and replaces the data with an unencoded Amtech 26 bit transmission. The Serial alpha substitution preprocessor substitutes ascii chatacter values for alphanumeric data, allowing alpha data to be converted into wiegand.

More information on the preprocessors can be found on page 22.



Here we will take a look at the input and output conversion options. It is suggested that you read through this section from beginning to end, as many constructs are not repeated for each conversion description.

Standard Wiegand Input

Wiegand data is received on input # 1, specifically on the Data 1 and Data 0 input lines. The wiegand data can be any bit size up to 250 bits, which is the maximum allowed by the converter.

Wiegand input data can be converted to serial data (rs232), magstripe data (clock&data or f2f, track 1 or track 2 character set), wand emulation (in code39), or wiegand data (standard wiegand, alphanumeric wiegand, or wieaba).

To process wiegand input data, you need to know the following information concerning the wiegand input you will be converting:

- site bit start position and length
- id bit start position and length
- issue code start position and length (if used)
- site character length
- id character length
- issue chracter length (if used)
- total input bits

The reason that you need to know these items is because prior to conversion the converter will attempt to *decimalize* the input bit stream. It will take the bit information that you have programmed in, and create 3 unique numbers - one number for the site code, 1 number for the badge number, and 1 number for the issue code (if it is used).

If you are not certain about the bit format, you can use the *serial dump mode* (explained later) to look at the bits and figure out what the format is, or get the information from the manufacturer of the card.

To decimalize the wiegand input data, you also need to program the converter to tell it how many decimal characters to use for each field. For example, with an 8 bit site code the maximum value the site code can be is 255, which is 3 characters. The 716 needs to be programmed with the field sizes for the site code, id field, and issue code (if you are using the issue code).

Wiegand input data (after it is decimalized) can then be masked and reordered prior to transmission.

Note that for wiegand inputs, the converter will process data in the order of site, badge, and issue. The resulting decimalized data will also be created in this order. For example, if you set the site field to 3 characters, the id field to 5 characters, and the issue field is ignored - if you read a card with a site code of 1 and an id code of 3333 then the resulting decimalized string is 00103333. This is the string you will see if you do a wiegand to rs232 (serial) conversion, with no masking.

Magstripe Input

Magstripe input is received on Input #1 (the same input port as is used for wiegand input). The converter can read magstripe track1 data, track2 data, or track3 data.

You do not need to know the format of the magstripe data unless you wish to mask the input data, but you do need to tell the converter which magstripe character set is being used (track 1 or track2) because these character sets are different. Track 1 magnetic stripes normally use the track1 character set, although there sometimes are exceptions to this rule. Track 2 magnetic stripes almost always use the track2 character set, while track 3 magnetic stripe data normally uses the track 2 character set.

For track 2 magnetic stripe data, you can program the converter to use the "C" start character rather than the traditional B start character. Some manufacturers use the C start on their cards rather than the standard B start. The C start is programmable for both magstripe inputs and outputs.

The magstripe data is decoded inside the converter, masked if necessary, and then output in the programmed format. Magstripe data can be converted to wiegand, serial, or magstripe.

Note that you can program the converter to convert both wiegand data and magstripe data on Input #1. In this mode, the converter will autodiscriminate between the two different input formats, and convert appropriately using the programmed parameters.

Magstripe input data can be converted to wiegand (all forms), magstripe (clock&data or f2f), or serial data.

RS232 Serial Input

Serial data can be converted to wiegand, magstripe, serial, or Code39 barcode (wand emulation). The serial data is received on Input port #2 (the rs232 db9 connector).

Serial data can be masked the same way as decimalized wiegand data or magstripe data. In other words, you can delete or insert characters into the stream, and extract only the information you need to construct the output stream.

Serial input data is normally terminated by a hexadecimal 0D (carriage return). When the converter sees this character, it knows that the data transmission is complete and it can start the conversion.

The 0D is the default terminator character in the converter, but it can be changed to any value that you wish. If you need to convert data that is coming from a device that sends a carriage return-linefeed sequence, for example, you can trigger on the linefeed as the terminator, and then mask out the carriage return using the masking commands. If you have version 1.8 firmware (or later), you can also automatically delete the linefeed character as another option. The converter has a command which will tell it to ignore all linefeed characters coming in on the serial line.

Wiegand Output

Wiegand output streams can be in either standard wiegand, alphanumeric wiegand, or wieaba. Alphanumeric wiegand is a mode where each character is transmitted using 8 bits (msb first). Alphanumeric data can be sent in this way electrically over a wiegand interface.

Wieaba is a mode where each character is electrically transmitted as wiegand but the character set used is the aba (magstripe track 2 or 1) character set.

To output wiegand data, the converter assumes that you have a decimalized field ready for output. This decimalized field can be either serial data which was received from the serial port, magstripe data which was received from input #1, or wiegand data which was received from input #1, and then decimalized internally by the converter.

To output in standard wiegand, you need to program the converter for the proper bit structure that you want to output. You need to tell the converter the following information:

- # site bits
- # id field bits
- # issue bits
- # constant bits
- # site characters
- # id characters
- # issue characters
- parity information

You need to program in the number of site, id, and issue characters because the source of data for the wiegand output is a decimalized string of data. Even if the data to convert was originally a wiegand input stream, it was decimalized internally in the converter so you still need to program in this information.

For example, if you want to convert serial to 26 bit wiegand, you might want to allow for the maximum field sizes allowable in the 26 bit format. This would be 3 digits for the site field, and 5 digits for the id field. You would program the # of site characters to 3, and the number of id characters to 5. To get a wiegand output with a site of 23 and and id of 4000, you would then send the converter 02304000 through the serial input to make the conversion.

You will notice we used 8 characters here, because we specified that 3 digits were to be used as a site code, and 5 would be used for the id field. Remember that the converter always processes data in the order of site, id, and then issue. Following these rules, if we sent the converter "234000" it would not work, since we specified 8 characters were needed. In this case, if our intent was still to use 23 as a site code, and 4000 as the id code, we would set the site characters parameter to 2, and the id characters parameter to 4.

Likewise, the internal strings created from magstripe inputs, or wiegand inputs, after masking, must of course satisfy these requirements. The internal decimalized string that you instruct the converter to make from an input, must be large enough to satisfy the requirements for the output mode that you selected.

Let's say that you want to convert 33 bit wiegand to 26 bit wiegand. Knowing that the maximum id number in a 26 bit wiegand is 65535, as long as all of our cards in the 33 bit format are below 65535 we can do it easily. You program the converter for the wiegand input format, and specify that it is to build an internal string of 3 digits for the site, and 5 for the id. Then, you program the converter to use 3 digits for the site code, and 5 for the id, when making the wiegand output (the same as we did above for a wiegand to serial conversion), and also program in the 26 bit structure. The converter will then convert the 33 bit format to a 26 bit format using your parameters.

Besides programming the bit structures for the wiegand output, you also need to tell the converter how it is to compute parity, if you will have parity bits in the wiegand output. Detailed information on the parity output can be found in the Advanced Programming section.

Magstripe Output

Magstripe output can be in the track 1 or track 2 character set. Track 2 emulation can start with either the standard B start character, or the optional C start character.

Data which is transmitted as magstripe data must meet the requirements necessary for magstripe conversion. For example, alphanumeric data can be transmitted using the track 1 character set, but it cannot be transmitted using the track 2 character set, because track 2 is numeric only.

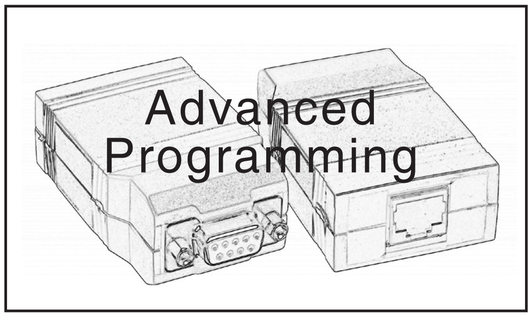
Magstripe can be output electrically in the clock & data mode (default) or optionally in the f2f mode. Note that the physical output line used for f2f output is different than the line used for clock&data output.

Serial Output

Serial output data is transmitted using the programmed baud rate and parity, and is followed by the programmed termination characters (if any). The internally generated string created from the input conversion, after masking) is the string that is transmitted out the serial output.

Wand Emulation Output

Wand emulation data is transmitted using the code39 charcter set. Wiegand input data, magstripe input data, and serial input data can be converted into code39 output. Note that the converter supports only the standard code39 character set, not the extended character set which supports lower case characters.



There are a number of advanced programming features in the 716. These features are discussed below.

Serial Dump Mode

Wiegand and Magstripe input data can optionally be dumped to the serial output port. When you program the converter to output in "standard" dump mode, the converter will output either a "1" character or a "0" character for each 1 bit or 0 bit in the data. This is a useful feature for examining the data from a card in bit form, or for determining how many bits may be on a card.

When in dump mode, the converter processes all bits irregardless of the input programming parameters. Even if you specify, for example, that a wiegand input is supposed to have 26 bits - if you are in the dump mode, you can read and dump any length up to the maximum 250 bits. There is no restriction on the number of bits that can be dumped from a magstripe card.

Dump mode data can be sent to the serial port only, and is terminated by the programmed serial terminator character(s).

Version 1.8 introduces the additional functionality of selecting up to 6 different "dump mode" styles. This is because in some situations using a serial dump of the input data may be preferable to actually peforming a conversion. The 6 programmable dump mode options are:

● Mode 0 (default)	This mode transmits 1 character, a "1" or a "0", for each bit received.
● Mode 1	This mode transmits 1 character for each 8 bits of data received. Bit 7 of the first character represents the first bit received, bit 6 the second bit received, and so on. The last character will be padded with zero bits if the received input data is not an even multiple of 8 bits.
● Mode 2	This mode is the same as mode 1, except 1 additional byte is transmitted at the beginning of the sequence. This byte contains the actual bit count of the data (in binary).
● Mode 3	This mode transmits 1 hexadecimal character for each 4 bits received. For example, if '1111' is received, then the character sent is an F to represent the 4 one bits. In this mode, leading zero bits in the data are ignored, and the last character bit sequence will be padded with zero bits if the input data is not an exact multiple of 4 bits, excluding leading zero bits.

● Mode 4	This mode is the same as mode 3, except leading zero bits are not ignored. The last character sequence may have padded zeroes to ensure the bit length is an even multiple of 4.
● Mode 5	Same as mode 2, transmitting one character for each 8 bits received, with 1 byte containing the bit count at the beginning. Mode 5 transmits 8 characters (64 bits) always, with an additional byte at the beginning representing the actual bit count.

Masking

Masking allows you to change the input data prior to output. Weigand input, Magstripe input, and serial input can all be masked.

There are two things that you can do with the masking;

1) Select characters from the input to be included in the output stream.
2) Insert characters at any position.

For example, let's say that you are converting an aba track 2 magstripe to serial. The magstripe contains 10 characters but you want to convert only the first five. You will set the making parameters to take only 5 characters starting at the first character position. This will leave you with 5 characters only that you will output. If you need to prefix the five characters with something, you can also use the mask commands to do that as well.

You can program up to 5 mask parameters for each type of input - serial, magstripe, and wiegand. The mask parameters are executed in order from the first to the fifth mask.

Both serial and magstripe data are character by nature, but wiegand data is not. Wiegand data is in a binary form and must be converted to a decimalized form prior to any masking. The way in which wiegand input is decimalized is controlled by the wiegand input parameters. When you perform maksing on wiegand data, you perform the masking after the wiegand data has been internally decimalized.

Note that for wiegand inputs, when decimalization occurs, the converter converts the site code first, then the id code, and then the issue code. The final product of the decimalization is a string of characters in the order of site, id, and then issue.

You need to remember that the decimalization is in this order so that if you perform masking you will perform them on the proper fields. As an example, if we are reading a 26 bit wiegand input and we set the site characters to 3, the id characters to 5, and turn off the issue code, then, with a wiegand input of 012 for a site and 30001 for an id, the resulting decimalized string is 01230001. When you perform the masking you will need to specify character positions for extraction. It is therefore necessary to remember the internal format of the decimalized number when performing any masks.

Wiegand Input Programming

Wiegand data is decimalized in the converter prior to masking or output. Even if you are converting wiegand to another form of wiegand, the converter must still decimalize the wiegand data before the conversion takes place.

In order to decimalize the data, the converter needs to know the complete bit structure of the wiegand input. The wiegand format typically consists of a site code (some people call it a facility code), and id number, and possibly an issue code and parity. The converter allows you to individually program each of these fields; however not all fields are required by the converter. If the input wiegand data contains all of these fields and your intent is to maintain field integrity on output, then you will need to specify each of these fields. If you do not need to maintain field integrity, then you do not need to specify each field.

For example, if you need to convert a standard 26 bit wiegand to a 30 bit wiegand structure, and the only difference between the two is that the 30 bit structure has 14 bits for the site code rather than 8, then you could use only one field. You can set the id field to 24 bits (the 26 bits minus the parity) and convert the full 24 bits to 28 bits for the output (28 bits being 30 minus 2 parity bits).

Integrity will be maintained because after the decimalization and conversion back to binary the result will be the same.

Note that although the converter can generate parity for wiegand outputs, it does not check the parity for wiegand inputs. Parity is ignored by the converter for inputs, except in the special case where you have specified 26 bit legacy support. In this mode, the converter assumes that you are reading a standard 26 bit wiegand card (not a prox card with a wiegand output). Some wiegand card readers allow the card to be read in both directions. In this mode, the converter will validate the parity in order to sense the direction of the card swipe. In all other cases, parity is ignored on the input.

To convert the wiegand input data, you must tell the converter **how** to decimalize the input data. To do this you need to know the starting bit position and bit length for each field that you will extract in the input stream.

Each of the input fields that you define are decimalized into a certain number of characters so that masking (if any) can occur prior to data output. You should specify the maximum number of characters needed to ensure that the input bit stream can be converted correctly. A table of the number of bits vs number of characters is located in Appendix B.

If we take a look at the standard 26 bit wiegand input - we will see that the maximum character size for the site code would be 3 (255 is the mamimum in 8 bits). For the ID field, it would be 5 (65535 is the maximum in 16 bits). You program the converter to allow 3 characters for the site field, and 5 characters for the ID field. These parameters will be large enought for the 26 bit format.

If you are reading an input which has larger bit fields than those discussed earlier, then you will have to set the field character sizes to a larger value. You can use the table in Appendix B to calculate those field sizes.

Note that the converter cannot process any individual field larger than 20 characters or 64 bits.

After wiegand data is received and decimalized, any masking that you have programmed for the wiegand input will be performed. The result will be a decimalized number that is then converted to the proper programmed output format.

Wiegand Output Programming

Serial data, magstripe data, and wiegand data can be converted to wiegand. There are three types of wiegand output formats supported by the converter. These are

- Standard Wiegand
- Alphanumeric Weigand
- Wieaba

Standard wiegand refers to the standard wiegand containing site code and id codes in binary.

Alphanumeric wiegand is a special form of wiegand supported by IBC where alphanumeric characters are transmitted electrically as wiegand. 8 bits (msb first) are transmitted for each character. There is no terminator character when transmitting alphanumeric wiegand.

Wieaba is a format used by Northern Computers and other manufacturers where the data is actually aba (magstripe track2 characters electrically transmitted as wiegand, 5 bits per character.

For alphanumeric wiegand and wieaba, there are no programmable parameters in the 716. These are straightforward transmissions which require no special programming.

Standard wiegand outputs must be programmed because the converter needs to know the format (bit structure) to use for the wiegand output.

To output in standard wiegand you need to tell the converter whether you want a site code, issue code, and id code, Also, if you want parity in the output stream you need to program the converter for the type of parity, and how to compute it (even, odd, or masked). Lastly, if there are any bits which need to remain constant in the output, they need to be programmed as well. These bits are programmed into the *constant* field.

Note that for wiegand inputs, when decimalization occurs, the converter converts the site code first, then the id code, and then the issue code. The final product of the decimalization is a string of characters in the order of site, id, and then issue. It is necessary to remember this when constructnig your output format.

Magstripe Input

The converter will accept magstripe data on Input 1. This is the same input line that wiegand data is also received on; however the 716 can process and convert both wiegand and magstripe data on the same input. The converter will autodiscriminate between wiegand and magstripe signals.

Magstripe data encoded in the track1 character set and magstripe data encoded in the track 2 character set can be read. Note that track3 magstripe data is normally encoded using the track 2 character set.

For the track 2 character set, both B start sentinels and C start sentinels can be read.

For conversions to serial data, the start sentinel, stop sentinel, and lrc values can be transmitted along with the magstripe data. Conversions from magstripe to wiegand or magstripe to magstripe cannot transmit these values because they are non-numeric values. A direction indicator (f for forward and r for reverse) can also prefix magstripe data for serial conversions only. This prefix will tell you which direction the magstripe was scanned in.

The seperator characters, if any, can optionally be converted to any value before transmission.

A length can also be set for the magstripe which will limit magstripe reads to a specific length. This length does not include the start or stop sentinels, or the lrc.

For track 1 magstripe inputs, alphas can optionally be deleted.

Wiegand Overrides

Each processed wiegand output field can be overridden by a preprogrammmed constant value. The override value(s) can be up to 16 bits in length. You can override a field which is larger than 16 bits; however the converter will override only the lowest 16 bits. Overrides are assumed to be left-justified, not right-justified.

Wiegand Constant Field

The wiegand constant field can be inserted into the wiegand output stream at any position. This field can be up to 16 bits in length, and contain any combination of 1's and 0's which need to be placed at a specific position in the output stream. Note that the constant field is inserted into the bit stream only if you have programmed this option ON. You must not only program the constant, but also turn the option on in order fo rthe constant to be inserted.

Wiegand Parity

Wiegand parity is not checked for wiegand inputs (except for legacy mode); however for wiegand outputs there are a number of options for parity. You can elect to output the wiegand data with no parity, standard parity, or a masked parity.

With no parity, the wiegand data is transmitted including only the fields that you have selected.

If you include standard parity in the output stream, then there is one parity bit at the beginning of the stream (left parity) and 1 parity bit and the end of the stream (right parity). The left side parity is calculated using sequential bits starting at bit #2 and continuing for the programmed parity bit length. The right side parity is calculated in the same way but starts at the last bit and works backwards for the specified number of bits. Both the left parity and right parity can be calculated as either an even or odd parity.

Parity can also be masked. In this mode, you must supply a mask of 64 bits to the converter. Each 1 bit in the mask signifies a specific bit location to be used in calculating the parity. Bit positions which have a 0 are ignored and not included in the parity calculation. The calculated parities can be either even or odd parity, and are inserted at the beginning of the stream and at the end.

Serial Terminators

Because the 716 may be connected to serial equipment which does not trigger on a carriage return, or transmit a carriage return, both the input terminators for serial transmissions and the output terminators are programmable.

Serial input transmissions can be terminated by any hexadecimal character except 00 and 7F. 00 and 7F are reserved characters for the 716 input. Serial output transmissions may have up to 2 terminator characters. The default is 1 terminator only, which is a carriage return. You can program these 2 terminator characters to any hex value.

Wiegand Field Order

Wiegand output transmissions normally consist of a site code, an id code, and optionally an issue code. The 716 also allows for a constant field as well. The order of transmission of these fields are programmable in the 716. In other words, you do not need to transmit a standard wiegand structure as left parity, site code, id code, and then right parity. The 716 allows you to transmit, for example, left parity + id + issue + site + right parity. You can order the output fields any way you like.

Lsb Support

Most wiegand formats contain fields, such as a site code, that are represented with the most significant bit first (MSB). The 716 allows you to specify that the order should be reversed (LSB) so that the least significant bit is first. This can be done on any field.

Speaker Control

There is an internal beeper inside the 716 which beeps each time a sucessful conversion is made. This feature can optionally be turned off.

Upon startup when power is first applied, the 716 should beep 3 times at different frequencies. This is an indication that the 716 has passed it's self test and is ready to operate. This "startup" beep can also be optionally turned off.

The duration of the beep for a good conversion indication can be programmed, as well as the speaker frequency.

Code39 Output

The 716 supports the conversion of data to wand emulation. The data is output as Code39 data. Note that the full ascii set is not supported, only the standard 43 character set. Lower case characters, therefore, cannot be transmitted using the wand emulation output setting.

Serial Parity

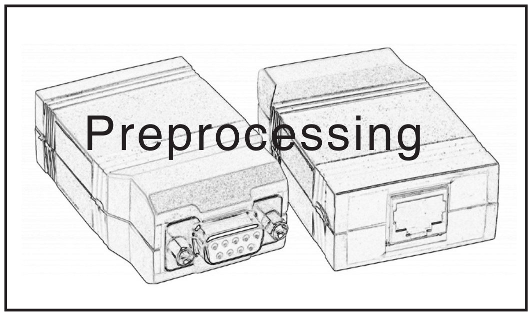
The 716 supports the generation of parity for serial transmissions. Parity is allowed for serial inputs, but it is not checked or validated on the input side. Parity can be even, odd, or none.

Ignore Line Feeds

Some serial devices normally transmit a carriage-return/linefeed (CR-LF) sequence at the end of transmissions. The 716 normally triggers on the carrige return as the indicator that the transmission is complete. If a device is sending a CR-LF sequence to the converter, the linefeed remains in the converter buffer and may impede subsequent conversions. Version 1.8 and later of the 716 allow you to program the converter to ignore all linefeeds that it sees on the serial input. This is an easier way than to attempt to use the masking prameters to eliminate the linefeed.

Suppress Leading Zeroes

Leading zeroes can be suppressed for serial outputs when converting wiegand to serial or magstripe to serial.



Preprocessing is new to version 1.8. Simply put, preprocessing allows you to "identify" certain input data sequences or styles and reformat or replace that data prior to the converter performing a conversion.

The preprocessing options are programmed into the converter by IBC and are not re-programmable; however additional preprocessing conversions will be available in the future.

There are 2 preprocessing options available in version 1.8. These options are:

1) Amtech 26 bit encoded format preprocessing

2) Hexadecimal character input preprocessing

There is one additional preprocessing option available with version 1.93:

1) Serial Alpha Substitution

The Amtech 26 bit encoded format processing autodiscriminates this format on the serial (rs232) input line. The Amtech 26 bit encoded format is of the form:

#xxxxx &hh:mm:ss:hhsmm/dd/yy, where xxxxx=encoded data

The standard Amtech 26 bit format uses 10 characters after the # to represent a site code and id code. The encoded format uses a special encoding format to encode those numbers into 5 characters, followed by 5 spaces.

The Amtech 26 bit encoded preprocessor in the 716 detects this format by looking for a sequence of 32 characters on the serial input that follow the format described above. If the converter sees this format, it automatically decodes the encoded data and replaces the 10 character portion of the input stream to look like a standard Amtech 26 bit format. Only these 10 characters are affected. The remainder of the serial input stream is left intact.

Note that the preprocessor does not ignore or replace any other data on the serial input. Only data that follows this format is affected by the preprocessor when this option is turned on.

The hexadecimal character input preprocessor converts hexadecimal characters received on the serial input to a bit stream based on your parameters. The bit stream is then decimalized into a numeric string, and the serial input data is replaced with the resultant string, so that data conversion can take place.

A simple example would be a proximity reader that outputs data serially, as hexadecimal characters, rather than output numeric characters. The hexadecimal characters represent a bit stream (similar to the 716's dump mode option 3). For example, let's take a 26 bit wiegand format with a facility code of 8 and an id of 8. The 26 bits would look like this:

10000100000000000000010000

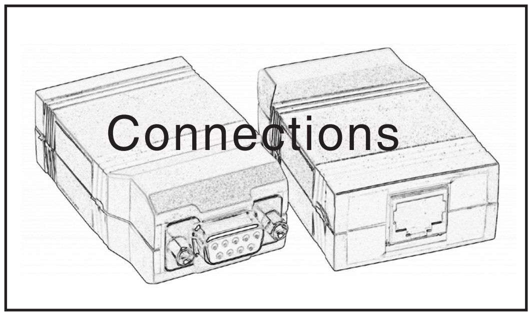
Represented as hexadecimal characters, this bit stream would be 8400040 or 1080010 depending on whether the bit data is transmitted as hex left justified and padded with zero bits, or right justified and prefixed with zero bits.

The 716 hexadecimal preprocessor can handle both formats. The 716 will decode the hexadecimal input into a bit stream internally. The appropriate bits are then extracted and decimalized into a numeric string inernally based on parameters that you program into the 716, and the serial input data is replaced with the resultant string.

Looking at our example above, taking the 8400040 which we know is a left-justified hexadecimal string, you would program the converter to treat the converted bit stream as a 26 bit wiegand format, in which case bits 2 through 9 would be the site code, and 10 through 25 would be the id code. Similar to other wiegand programming in the 716, you will also tell the converter to use x characters for the extracted site information, and x characters for the extracted id information. For the purpose of this example, we will use 3 for the site and 5 for the id. The converter will preprocess the 8400040 into a bit stream, extract the bits as you have programmed, and then create the decimalized string of 00800008, and replace the serial input stream with this data. You can then implement any of the standard serial input 716 conversions on this result.

The Serial Alpha Substitution preprocessor allows you to convert alpha characters and numbers to their decimal ascii equivalents. This is used if you want to convert a serial stream to wiegand, but the serial data contains alpha characters. Each character in the stream (up to 9 maximum) is converted to a 2-digit number, using the ascii value of the character. For example, "12" becomes "4950", "A12" becomes "654950". The resulting string replaces the original string for eventual conversion.

Note that because the 716 is firmware upgradable, new preprocessing functions can be added easily when new firmware is released. If you have a need for a specific type of preprocessing that you would like added to the 716, contact IBC.



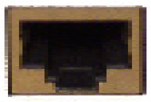
There are 2 sets of connectors on the converter.

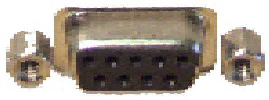
On one side is the DB9 connector for rs232 communictionsor a Type B connector forr usb connections. The rs232 db9 is a female db9 connector, and is wired as a direct connect to a pc. You do not need a null modem cable when connecting to a pc.

On the other side is the RJ45 connector which contains the wiegand/magstripe inputs and the wiegand/magstripe/wand emulation outputs, as well as the f2f and led control line.

For model 716, the input voltage must be 12VDC.
For model 716-5, the input voltage must be 5VDC.

Model 716-C can take any voltage from 5VDC to 12VDC.


RJ45 connector

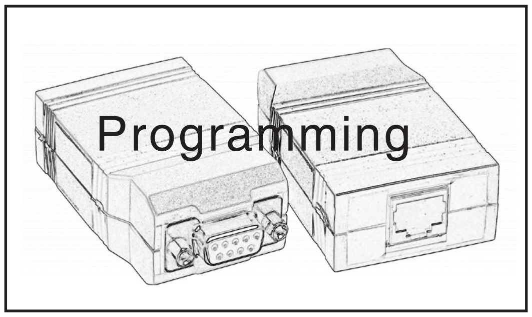

DB9 connector

From left to right, the RJ45 pins are

- Data 1/Mag Data IN
- Data 1/Mag Data/Wand OUT
- Ground
- +12VDC or +5VDC depending on model
- Data 0/Mag Clock IN
- Data 0/Mag Clock OUT
- Mag Media (card present) OUT
- Led Control OUT / F2F OUT

The supplied RJ45 cable which ships with the 716 is pinned out as follows:

Red	+12VDC
Black	Ground
Brown	Led control line / F2F
Blue	Mag media
Grey	Data 1 / Mag Data IN
Green	Data 0 / Mag Clock IN
Orange	Data 1 / Mag Data OUT
Yellow	Data 0 / Mag Clock OUT



Programming commands must start with a **null** (hex 00) character and end with either the programmed serial input termination character (normally hex 0D, carriage-return), or a **delete** (hex 7F) character.

For each command, the uppercase characters must be typed in exactly as shown. The lowercase characters represent parameters which are required. Each parameter is positional and must be entered.

Do **not** include spaces in the command as they are shown here in the commands. The spaces are only shown for clarity.

Please note that for some inputs and outputs, multiple commands must be issued in order to fully program the input or output.

If the programming command is accepted by the converter, the converter will respond with OK followed by the terminator character. If the command is not accepted, the converter will respond with NOK followed by the terminator character.

You can also program the converter directly using the 176 Converter Utility located on the IBC web site.

Note that to program version 1.8 and above, you will need version 1.8 of the utility software. If you use a version of the utility software prior to 1.8, it will not support all of the version 1.8 commands, and you will have to enter those commands by hand using the "terminal mode".

Note that to program version 1.94 and above, you need version 1.94 of the utility program. To program version 1.96 and above, you need version 1.96 of the utility program.

Important

Please note that 128 bit support was added in version 1.81 and 250 bit support was added in version 1.95. All commands which use a 3-digit number for bit positions or lengths, are supported in version 1.81 and higher. Pervious versions required these values to be 2 digits long, not 3.

Any earlier version of the firmware can be updated to any later version of the firmware using the utility. Note that when you update firmware, it is a good idea to "reset" the 716 using the reset command so that all parameters used in the newer firmware are set properly. This will cause the 716 to lose any settings you had previously set, so you must reprogram the unit as well after performing the reset.

Also note that some serial commands, i.e. baud rate, I/f ignore, etc.. may be programmed but will not take effect unless the unit is restarted. This is done so that you can continue to program the unit using the same serial settings, until you are complete with the programming.

Serial Port General Setup Command

S1G b p xx yy zz o m b

b =	baud rate	1	=	1200
		2	=	2400
		3	=	4800
		4	=	9600
		5	=	19200
		6	=	38400
		7	=	57600
		8	=	115200
p =	parity	O	=	7 bits odd parity
		E	=	7 bits even prity
		N	=	8 bits no parity
xx =	input terminator	xx	=	hexadecimal value of input terminator
yy =	output terminator 1	xx	=	hexadecimal value of output terminator 1, 00 for none
zz =	output terminator 2	xx	=	hexadecimal value of output terminator 2, 00 for none
o =	output conversion	0	=	no conversion
		1	=	convert to magstripe
		2	=	convert to wand emulation
		3	=	convert to standard wiegand
		4	=	convert to alphanumeric wiegand
		5	=	convert to aba wiegand
		6	=	convert to serial (version 1.8 and above)
m =	masking	1	=	masking turned on, 0=masking off
b =	beep on	1	=	beep on after data convert, 0=no beep

Notes

- The p parameter refers only to rs232 transmissions and not receipts. Parity is not checked for rs232 received data.
- If m is set to a 1, then masking will occur on the serial input before final conversion and transmission.
- Output conversion mode "6", serial-to-serial, is not supported in firmware versions below 1.8.
- This command will reset the "ignore linefeed" option. If you wish to ignore line-feed characters on the serial input, the ignore linefeed command X1 must be sent after this command.

Serial Masking Commands (version 1.7+)

```
S1K p1 l1 p2 l2 p3 l3 p4 l4 p5 l5          S1Q2 ll
S2K p1 l1 p2 l2 p3 l3 p4 l4 p5 l5          S1Q3 ll
S3K p1 l1 p2 l2 p3 l3 p4 l4 p5 l5
```

px =	position to start extraction	00=constant indicator
lx =	length of extraction	00=all characters
ll =	serial input length	00=ignore

Notes

- There are 3 possible serial masks that can be entered in. Mask # 1 is the default mask. Masks # 2 and # 3 are executed only if the input data matches the length programmed for that mask using the S1Q commands. In other words, if you want to perform masking only on serial inputs that have a length of 10, then use the S1Q2 command to set the length of mask 2 to 10. All serial inputs with a length of 10 will automatically use that mask.
- To disable mask 2 or 3, set their length parameters to 00.
- If masking is turned on, then mask 1 is always used by default if the serial input length does not match the lengths set for mask 2 or mask 3. If masks 2 and 3 are turned off by setting their lengths to 00, mask 1 is still used by default if the masking is turned on. In this case, to allow a complete pass-through of the serial data, set mask 1 to a start position of 01, and a length of 00.
- Px and Lx parameters must be entered in as hexadecimal, i.e. 0A represents 10.
- There are 5 possible masking actions which can occur for each mask. Action # 1 occurs first, followed by 2,3,4, and 5. The converter will execute them in this order.
- To extract a **substring** from the serial data received, set the px parameter to the starting character position to extract, and set lx to the number of characters to extract. Setting lx to a 00 tells the converter to select all characters starting at the px position.

  Example: take the first five characters from the input stream only:

  S1K01050000000000000000

  Example: take all characters starting at the fifth character:

  S1K05000000000000000000
- To insert a **constant** into the data, enter a 00 for the px parameter and enter the constant into the lx parameter. The constant must be entered in as a hexadecimal value, i.e. 48="0", 49="1".

  Example: insert a "5" at the beginning ao the stream, and then extract the remaining characters:

  S1K00530100000000000000
- A mask entry with a P value of 00 and a L value of 00 is ignored.

Serial to Weigand Conversion Command 1 (of 6)

S1W1 ss sl bs bl is il sb bb ib cb

ss	=	site character start position	00 to ignore site field
sl	=	site character length	00 to ignore site field
bs	=	id character start position	00 to ignore id field
bl	=	id character length	00 to ignore id field
is	=	issue character start position	00 to ignore id field
il	=	issue character length	00 to ignore id field
sb	=	# bits to use for site	00 to ignore site field
bb	=	# bits to use for id	00 to ignore id field
ib	=	# bits to use for issue code	00 to ignore issue field
cb	=	# bits to use for constant field	00 to ignore const field

Notes

- The **start positions** refer to the character position in the input serial stream that the field starts in.
- The **lengths** refer to the character lengths in the input serial stream to be used in the calculation.

  Example:

  The serial stream is an 8 digit number to be encoded into a site code of 3 digits and an id field of 5 digits : ss should be 01, sl should be 03 bs should be 04, bl should be 05, is should be 00, il should be 00. This example assumes that the 8 digits serial data to convert is in the format sssiiiii where sss is the site code, iiiii is the id field.

- The **# bits** parameters refer to the actual number of bits this field will use in the wiegand output that you will be creating.

  Example:

  You want a 26 bit wiegand output which consists of 8 bits for the site code and 16 bits for the id field. There is no issue code and you will not be inserting any constant data into the wiegand output bit stream: sb=09, bb=16, ib=00, cb=00.

- You can specify up to 64 bits for any of the fields (site, id, issue, constant); how ever the total length of the final wiegand structure cannot be greater than 250.

Serial to Weigand Conversion Command 2 (of 6)

S1W2 p l r s b i c lll rrr x y z

p	=	parity type	0=none, 1=standard, 2=use the mask
l	=	left parity style	1=odd, any other value=even
r	=	right parity style	1=even, any other value=odd
s	=	site order	1=lsb first, any other value=msb
b	=	id (badge) order	1=lsb first, any other value=msb
i	=	issue order	1=lsb first, any other value=msb
c	=	constant field order	1=lsb first, any other value=msb
lll	=	left parity count	standard parity only
rrr	=	right parity count	standard parity only
x	=	site override	1=site override on, any other value=off
y	=	id override	1=id override on, any other value=off
x	=	issue override	1=issue override on, any other value=off

Notes

- **Standard Parity** refers to parity which is calculated as a sequential # of adjacent bits, the length being determined by the **parity count** parameter(s). Most wiegand streams calculate parity in this way. Left parity is calculated starting at the second bit (the first bit is the parity placeholder) for a length of parity count. Right parity starts at the last bit (before the right parity) and works backwards consecutive bits as defined by the right parity count parameter.
- **Masked Parity** refers to parity which is calculated using a mask of bits determining which bit positions to use.
- **Parity Style** refers to whether the calculated parity should be even or odd parity.
- **Order** refers to the direction of the bits. Most systems use MSB first however you can instruct the 716 to do LSB first instead.
- **lll** and **rrr** are 2 digits for versions prior to 1.81.
- Turning on any **Override** means that you will be specifying an override value that will replace the existing value in the field prior to transmission.

Example:

We will be converting to a standard wiegand 26 bit format which has even parity for the left side parity and odd parity for the right side, 12 bits (exclusive) each. All output fields are msb first, and we will be overriding the site code with a value.

The command is S1W2 10000001212100

Serial to Weigand Conversion Command 3 (of 6)

S1W3 a b c d

a =	first field	0=none, 1=site, 2=id, 3=issue, 4=constant
b =	second field	0=none, 1=site, 2=id, 3=issue, 4=constant
c =	third field	0=none, 1=site, 2=id, 3=issue, 4=constant
d =	fourth field	0=none, 1=site, 2=id, 3=issue, 4=constant

Notes

● This command specifies the **order** that the wiegand fields are to be transmitted in. Standard wiegand normally has the site code followed by the id code (badge number) followed by the issue code. You specify the ordering of the fields with this command. Fields are transmitted in the order of a,b,c,d above.

Example: Transmit only site code followed by id number field:

S1W31200

Serial to Weigand Conversion Command 4 (of 6)

S1W4 ssss iiii bbbb cccc

ssss = site override	16 bit override (4 hex characters)
iiii = issue override	16 bit override (4 hex characters)
bbbb = badge (id) override	16 bit override (4 hex characters)
cccc = constant value	16 bit constant value (4 hex characters)

Notes

● This command is used to specify **override** values for a specific wiegand field. The override must also be turned on for the override to take effect.

● The **constant value** can be up to 16 bits which may be used as a constant field in the wiegand output, placed in any position in the wiegand output bit stream.

● Data for any of the 4 parameters must be entered in in hexadecimal. For example, if you want to override the site code with a 23, then the ssss parameter must be 0017, which is the hexadecimal repreentation for 23.

Serial to Weigand Conversion Commands 5&6 (of 6)

S1W5 llllllllllllllll
S1W6 rrrrrrrrrrrrrrrr

llllllllllllllll	= left parity mask
rrrrrrrrrrrrrrrr	= right parity mask

Notes

- These commands are used for setting the parity masks. If you elect to use a masked parity, then you must enter in a 64 bit value to use for the mask. The commands take a 16 character hexadecimal value, which represents 64 bits total. Each 1 bit in the entered mask represents a bit to be used from the wiegand stream for calculating parity. Note that the bit mask values should be right justified. Note that although you can output up to 250 bits, the parity masking can be done only on outputs up to 128 bits.

Serial to Magstripe Conversion Command

S1M t n c f

t	=	magstripe character set	1= use track 1 set, any other value=tk2
n	=	neuron mode	1= use neuron mode, and other value=no
c	=	c start	1= use c start, any other value=no
f	=	f2f	1= magstripe output is transmitted in F2F

Notes

- Set the t parameter to the magstripe track emulation that you want to convert to.
- Neuron mode refers to Neuron emulation, which is a mode where both data and clock signals need to rise together at the same time. Use this only if your controller requires it. Note this option is meaningless when using F2F.
- Set the C start option to a 1 if you want to send a C start character as opposed to a B start character. This pertains to track 2 emulation only.
- Set the F2F option on if you want to output the data in F2F. Both the track 1 and track 2 character sets, as well as the C start option, can be specified when using F2F.

Input Port (wiegand and magstripe) General Setup Command

I1IG i o m a b p

i	=	input type	1	=	wiegand
			2	=	magstripe
			3	=	alphanumeric wiegand
			4	=	wieaba
			5	=	wiegand or magstripe
o	=	output conversion	1	=	convert to magstripe
			2	=	convert to wand emulation
			3	=	convert to serial (rs232)
			4	=	dump to serial port (rs232)
			5	=	convert to standard wiegand
			7	=	convert to wieaba
			8	=	convert to alphanumeric wiegand
m	=	masking	1	=	masking turned on, 0=off
a	=	unused			put a 0 in this parameter always
b	=	beep	1	=	beep for a good conversion, 0=off
p	=	parity check	1	=	parity check on for legacy wiegand, 0=off

Notes

- Setting the **input type** parameter to 5 allows the converter to process both wiegand data and magstripe data on the input port. Each of these have their own conversion parameters in the 716 which are separate from each other. It is possible to therefore convert both wiegand inputs and magstripe inputs at the same time, with different conversions based on the type of input.

- Setting the **output conversion** parameter to 4 is a special mode where the converter will dump (bit by bit) each bit out the serial rs232 port. The serial termination character(s) are appended to the end of the output stream, but no other masking or data checking occurs.

- The **parity check** option is set only for checking parity on a 26 bit legacy (wiegand wire) card. Set this parameter on if you want parity checking. This option will also auto check to see if the card was read backwards.
  In Version 1.4 and later, this option also applies to 28 bit cards.

- If using output conversion 4, **dump to serial mode**, the format of the output data is determined by the dump parameter (defined later).

Magstripe Input Setup Command

I1IM t c s I d a xx yy (required for magstripe inputs only)

t	=	track	1	=	track 1, otherwise track 2
c	=	c start	1	=	look for a "c" start on track2
s	=	start/stop	1	=	include start/stop characters
l	=	lrc	1	=	include lrc character
d	=	direction	1	=	include direction indicator
a	=	alpha delete	1	=	delete alphas from input
xx	=	length			length of input, 00 for any
yy	=	seperator char			translate sep char to this hex value

Notes

- If c start is set, then for track 2 magstripe input only, the converter will allow only magstripes that have a "c" start, as opposed to the traditional "b" start.
- If start/stop is set, for serial outputs only, the converter will include the start and stop character from the magstripe in the output stream.
- If lrc is set, for serial outputs only, the converter will include the lrc from the magstripe as part of the output stream.
- If direction is set, for serial outputs only, the output stream will be prefixed by either an "f" for forward, or a "r" for reverse, indicating the direction of the original swipe.
- If alpha delete is set, then all alpha characters in the input are deleted prior to conversion. This option affects magstripe data using the track 1 character set only.
- If a length is set, then the magstripe which is read must have the same length as the length entered. This length does not include the start/stop characters as well as the lrc character, if those items have been turned on. To allow any length magstripe, use a length of 00.
- If you want to translate the seperator character to anything other than standard, then you can set xx to the hexadecimal value of the character that the seperator character is to be replaced with. Use 00 for no conversion.

Wiegand Input Setup Commands

```
I1IW bbb sss sl x bbs bl y iis il z sc bc ic         (1st input format)
I1IX bbb sss sl x bbs bl y iis il z sc bc ic         (2nd input format)
I1IV bbb sss sl x bbs bl y iis il z sc bc ic         (3rd input format)
```

bbb	=	total bits	number of bits being read in
sss	=	site start	starting bit number for the site code, 00 if no site
sl	=	site length	number of bits for the site code, 00 if no site
x	=	site direction	1-backwards (lsb first), 0=normal (msb first)
bbs	=	badge (id) start	starting bit number for the id code, 00 if no id
bl	=	badge (id) length	number of bits for the id code, 00 if no id
y	=	id direction	1-backwards (lsb first), 0=normal (msb first)
iis	=	issue code start	starting bit number for the issue code, 00 if no issue
il	=	issue code length	number of bits for the issue code, 00 if no issue
z	=	issue code direction	1-backwards (lsb first), 0=normal (msb first)
sc	=	site characters	# of characters to convert the site code into
bc	=	badge (id) characters	# of characters to convert the id code into
ic	=	issue code characters	# of characters to convert the issue code into

Notes

- the **total bits** parameter specifies the number of bits that the converter is expect ing to read. Only bit streams with this number of bits are processed. All others are ignored, unless the converter is programmed to dump the bits in dump mode.
- The first bit in a stream is considered bit 1.
- Set any **direction** parameter to 1 if the input field is backwards with lsb first.
- The **characters** parameter specifies the number of characters to use when decimalizing the input. Remember that the wiegand input is first decimalized into a site field, id field, and issue field prior to final conversion, These charac-ter sizes must be large enough to contain the decimalized value for any of the fields input. The maximum allowable field size is 20, which is enough for a full 64 bit input.
- Two different input formats can be defined and processed. To turn off the 2nd input format, make the total bit count 0**.**
- **bbb,sss,bbs,iis** are 2 digits for versions prior to 1.81.

Magstripe and Wiegand Input Masking Commands

```
I1IK p1 l1 p2 l2 p3 l3 p4 l4 p5 l5         (1st format)
I1IY p1 l1 p2 l2 p3 l3 p4 l4 p5 l5         (2nd format)
I1IZ p1 l1 p2 l2 p3 l3 p4 l4 p5 l5         (3rd format)
```

px	=	position to start extraction	00=constant indicator
lx	=	length of extraction	00=all characters

Notes

- This command is the same as the serial input mask command. Please refer to the documentation on that command for further notes.
- Both wiegand and magstripe inputs use the same mask. There is not a separate mask for weigand and a separate mask for magstripe.
- The 2nd mask format will be used for wiegand inputs that match the 2nd wiegand format described above.

Magstripe/Wiegand to Weigand Conversion Command 1 (of 6)

I1OW1 ss sl bs bl is il sb bb ib cb

ss =	site character start position	00 to ignore site field
sl =	site character length	00 to ignore site field
bs =	id character start position	00 to ignore id field
bl =	id character length	00 to ignore id field
is =	issue character start position	00 to ignore id field
il =	issue character length	00 to ignore id field
sb =	# bits to use for site	00 to ignore site field
bb =	# bits to use for id	00 to ignore id field
ib =	# bits to use for issue code	00 to ignore issue field
cb =	# bits to use for constant field	00 to ignore const field

notes

- The **start positions** refer to the character position in the masked input stream that the field starts in. The masked input stream is the wiegand data after it has been decimalized and masked, or the magstripe stream after it has been masked.
- The **lengths** refer to the number of characters to use for generating the wiegand output number.

  Example:

  The wiegand input stream is 26 bits consisting of a site code (max 255) and a badge number (max 65535) which are internally converted to an 8 digit number (3 digits for the site, 5 digits for the badge) based on the parameters programmed in for the wiegand input conversion. To follow this same convention for outputting the wiegand data, the site start position should be set to 1, and the length set to 3. The badge (id) start position should be set to 4, and the length to 5.

- The **# bits** parameters refer to the actual number of bits this field will use in the wiegand output that you will be creating.

  Example:

  You want a 26 bit wiegand output which consists of 8 bits for the site code and 16 bits for the id field. There is no issue code and you will not be inserting any constant data into the wiegand output bit stream: sb=09, bb=16, ib=00, cb=00.

Magstripe/Wiegand to Weigand Conversion Command 2 (of 6)

i1OW2 p l r s b i c lll rrr x y z

p	=	parity type	0=none, 1=standard, 2=use the mask
l	=	left parity style	1=odd, any other value=even
r	=	right parity style	1=even, any other value=odd
s	=	site order	1=lsb first, any other value=msb
b	=	id (badge) order	1=lsb first, any other value=msb
i	=	issue order	1=lsb first, any other value=msb
c	=	constant field order	1=lsb first, any other value=msb
lll	=	left parity count	standard parity only
rrr	=	right parity count	standard parity only
x	=	site override	1=site override on, any other value=off
y	=	id override	1=id override on, any other value=off
x	=	issue override	1=issue override on, any other value=off

Notes

- **Standard Parity** refers to parity which is calculated as a sequential # of adjacent bits, the length being determined by the **parity count** parameter(s). Most wiegand streams calculate parity in this way. Left parity is calculated starting at the second bit (the first bit is the parity placeholder) for a length of parity count. Right parity starts at the last bit (before the right parity) and works backwards consecutive bits as defined by the right parity count parameter.
- **Masked Parity** refers to parity which is calculated using a mask of bits determining which bit positions to use.
- **Parity Style** refers to whether the calculated parity should be even or odd parity.
- **lll** and **rrr** are 2 digits for versions prior to 1.81.
- **Order** refers to the direction of the bits. Most systems use MSB first however you can instruct the 716 to do LSB first instead.
- Turning on any **Override** means that you will be specifying an override value that will replace the existing value in the field prior to transmission.

  Example:

  We will be converting to a standard wiegand 26 bit format which has even parity for the left side parity and odd parity for the right side, 12 bits (exclusive) each. All output fields are msb first, and we will be overriding the site code with a value.

  The command is I1OW210000001212100

Magstripe/Wiegand to Weigand Conversion Command 3 (of 6)

I1OW3 a b c d

a =	first field	0=none, 1=site, 2=id, 3=issue, 4=constant
b =	second field	0=none, 1=site, 2=id, 3=issue, 4=constant
c =	third field	0=none, 1=site, 2=id, 3=issue, 4=constant
d =	fourth field	0=none, 1=site, 2=id, 3=issue, 4=constant

Notes

- This command specifies the **order** that the wiegand fields are to be transmitted in. Standard wiegand normally has the site code followed by the id code (badge number) followed by the issue code. You specify the ordering of the fields with this command. Fields are transmitted in the order of a,b,c,d above.

Example: Transmit only site code followed by id number field:

I1OW31200

Magstripe/Wiegand to Weigand Conversion Command 4 (of 6)

I1OW4 ssss iiii bbbb cccc

ssss = site override	16 bit override (4 hex characters)
iiii = issue override	16 bit override (4 hex characters)
bbbb = badge (id) override	16 bit override (4 hex characters)
cccc = constant value	16 bit constant value (4 hex characters)

Notes

- This command is used to specify **override** values for a specific wiegand field. The override must also be turned on for the override to take effect.
- The **constant value** can be up to 16 bits which may be used as a constant field in the wiegand output, placed in any position in the wiegand output bit stream.
- Data for any of the 4 parameters must be entered in in hexadecimal. For example, if you want to override the site code with a 23, then the ssss parameter must be 0017, which is the hexadecimal representation for 23.

Magstripe/Wiegand to Weigand Conversion Commands 5&6 (of 6)

I1OW5 llllllllllllllll
I1OW6 rrrrrrrrrrrrrrrr

llllllllllllllll	= left parity mask
rrrrrrrrrrrrrrrr	= right parity mask

Notes

- These commands are used for setting the parity masks. If you elect to use a masked parity, then you must enter in a 64 bit value to use for the mask. The commands take a 16 character hexadecimal value, which represents 64 bits total. Each 1 bit in the entered mask represents a bit to be used from the wiegand stream for calculating parity. Note that the bit mask values should be right justified.

Magstripe/Wiegand to Magstripe Conversion Command

I1OM t n c f

t	=	magstripe character set	1= use track 1 set, any other value=tk2
n	=	neuron mode	1= use neuron mode, and other value=no
c	=	c start	1= use c start, any other value=no
f	=	f2f output	1= use F2F output rather than clock&data

Notes

- Set the t parameter to the magstripe track emulation that you want to convert to.
- Neuron mode refers to Neuron emulation, which is a mode where both data and clock signals need to rise together at the same time. Use this only if your controller of receiver requires it.
- Set the C start option to a 1 if you want to send a C start character as opposed to a B start character. This pertains to track 2 emulation only.
- Set the F2F option on if you want to output the data in F2F. Both the track 1 and track 2 character sets, as well as the C start option, can be specified when using F2F.

Magstripe and Wiegand Timing Command

T a b c d

a =	mag time between pulses	in milliseconds, 1 thru 5
b =	wiegand time between pulses	in milliseconds, 1 thru 5
c =	wiegand pulse duration	1=50us, 2=100us, 3=150us
d =	mag pulse duration	1=50us, 2=100us, 3=150us

Notes

- Standard timing for magnetic stripe is a 100us pulse and 1ms between pulses.
- Standard timing for wiegand is a 50us pulse and 1ms between pulses.
- The timing for F2f signals is fixed, at 1ms for a 0 phase transition.

Speaker frequency and duration Command

P fff 111 222 333 444

fff =	frequency	010 thru 255 (010 is the higher frequency)
111=	wie/mag duration	001 thru 255, 100ms increments
222=	unused	set this to 000
333=	serial duration	001 thru 255, 100ms increments
444=	unused	set this to 000

Notes

- The speaker has a resolution of 100ms. Setting the duration to 1 may cause the speaker to turn on only momentarily. It is suggested to always add one to your duration to ensure proper operation.

Hexadecimal Preprocessor (serial inputs) version 1.8+

X2 bb ss sl is il sc ic j

bb	=	total # of bits to process
ss	=	starting bit for the site code
sl	=	site code length in bits
is	=	starting bit for the id
il	=	id code length in bits
sc	=	number of characters to convert the site code into
ic	=	number of characters to convert the id code into
j	=	0 = do not trim leading zero bits, 1 = trim leading zero bits

Notes

- This command sets up the parameters for the serial input hexadecimal preprocessor. Sending this command turns on the preprocessor. To turn off the preprocessor, send the command with a bb parameter of 00.
- Turning on this preprocessor turns off any of the other preprocessors which effect the serial input data. Only one preprocessor may run at a time on the serial input.
- If this preprocessor is turned on, serial input data in converted based on the parameters entered in this command. Serial input data which does not meet this format is ignored and not processed.
- For further information, see pages 22 and 23.

Amtech 26 bit encoded format Preprocessor (serial inputs) version 1.8+

X3 f

f = 1 - processor on 0 - processor off

Notes

- See page 22 for a detailed explanation.
- Turning on this preprocessor turns off any of the other preprocessors which effect the serial input data. Only one preprocessor may run at a time on the serial input.

Dump Mode format Command (version 1.8+)

X4 f

f = style of dump mode output

Notes

- See pages 14 and 15 for a detailed explanation of the different dump mode output styles.

Alpha Substitution Preprocessor (serial inputs) (version 1.93+)

X5 f

f = 1 - processor on 0 - processor off

Notes

- See page 23 for a detailed explanation.
- Turning on this preprocessor turns off any of the other preprocessors which effect the serial input data. Only one preprocessor may run at a time on the serial input.

Zero Suppression for serial outputs (version 1.94+)

X6 f

f = 1 - suppress leading zeroes 0 - don't suppress leading zeroes

Notes

- Suppresses leading zeroes for serial outputs when converting from wiegand to serial, or magstripe to serial.

Speaker startup tone Command

A x

x	=	startup tone on/off	0=off, 1=on

Notes

- When the 716 turns on, it will beep three times. You can turn this feature off by sending this command with a parameter of 0.

Linefeed ignore Command

X1 x

x	=	ignore linefeeds on the serial input	0=off, 1=on

Notes

- This command must be issued **after** the S1G command, because the S1G command will reset this parameter to 0. You **must** restart the converter in order for this command to take effect.

Reset Command

U

Notes

- This command resets the 716 converter to the factory defaults. Any programming that you have done to the converter will be lost if you issue this command.

Restart Command (v1.96+)

Y

Notes

- This command restarts the 716 converter with the latst programmed parameters.

Serial Control Commands

The following are commands which you can send to 716 converter to control the led, turn on the speaker, and receive the converter's version information in real-time.

Note that although these commands are not programming commands, they still need to be proceeded by the NULL character.

=xx	Turn Led ON for xx seconds then turn OFF
-xx	Turn Led OFF for xx seconds then turn ON
)xx	Blink Led ON/OFF for xx seconds then turn OFF
(xx	Blink Led ON/OFF for xx seconds then turn ON
&xx	Turn Speaker on for xx time, xx=100ms increments
!xx	Turn on relay (716R, 718R) for xx seconds
V	Return version identification string. Note that all versions of the 716 converter series will return "716" as part of the version identification. 716R, 718, and 718R converters also return "716". There is no differentiation amongst these converters with the version identification string.

Note that if F2f output is being used, you should not issue any of the Led commands because F2F uses the same line as the led control..

Appendix A - Converter Defaults

This table shows the defaults which are normally stored in the converter when it leaves the factory.

Wiegand/Magstripe Input	Allow both, with magstripe set to track 2 and wiegand set to a standard 26 bit card. Convert to serial.
Serial Input	No preprocessors turned on, convert serial to magstripe track 2
Masking	No masks are loaded or turned on.
Dump Mode	0 (dump bit by bit)
Ignore Linefeed	off

Appendix B - Wiegand Bit Size Table

This table shows the number of bits required to encode a specific number into wiegand. This table can also be used to see how many charcters are required to decimalize wiegand input fields.

Bits	Decimal Number	Bits	Decimal Number
1	1	33	8 589 934 591
2	3	34	17 179 869 183
3	7	35	34 359 738 367
4	15	36	68 719 476 735
5	31	37	137 438 953 471
6	63	38	274 877 906 943
7	127	39	549 755 813 887
8	255	40	1 099 511 627 775
9	511	41	2 199 023 255 551
10	1 023	42	4 398 046 511 103
11	2 047	43	8 796 093 022 207
12	4 095	44	17 592 186 044 415
13	8 191	45	35 184 372 088 831
14	16 383	46	70 368 744 177 663
15	32 767	47	140 737 488 355 327
16	65 535	48	281 474 976 710 655
17	131 071	49	562 949 953 421 311
18	262 143	50	1 125 899 906 842 623
19	524 287	51	2 251 799 813 685 247
20	1 048 575	52	4 503 599 627 370 495
21	2 097 151	53	9 007 199 254 740 991
22	4 194 303	54	18 014 398 509 481 983
23	8 388 607	55	36 028 797 018 963 967
24	16 777 215	56	72 057 594 037 927 935
25	33 554 431	57	144 115 188 075 855 871
26	67 108 863	58	288 230 376 151 711 743
27	134 217 727	59	576 460 752 303 423 487
28	268 435 455	60	1 152 921 504 606 846 975
29	536 870 911	61	2 305 843 009 213 693 951
30	1 073 741 823	62	4 611 686 018 427 387 903
31	2 147 484 647	63	9 223 372 036 854 775 807
32	4 294 967 295	64	18 446 744 073 709 551 615

Appendix C - ASCII Table

This table shows the ascii character set, with decimal and hex equivalents. This table is useful when entering in hex or decimal constants when programming.

Decimal	Hex	Character	
000	00	NULL	
001	01	SOH	CTRL-A
002	02	STX	CTRL-B
003	03	EXT	CTRL-C
004	04	EOT	CTRL-D
005	05	ENQ	CTRL-E
006	06	ACK	CTRL-F
007	07	BEL	CTRL-G
008	08	BS	CTRL-H
009	09	HT	CTRL-I
010	0A	LF	CTRL-J
011	0B	VT	CTRL-K
012	0C	FF	CTRL-L
013	0D	CR	CTRL-M
014	0E	SO	CTRL-N
015	0F	SI	CTRL-O
016	10	DLE	CTRL-P
017	11	XON	CTRL-Q
018	12	DC2	CTRL-R
019	13	XOF	CTRL-S
020	14	DC4	CTRL-T
021	15	NAK	CTRL-U
022	16	SYN	CTRL-V
023	17	ETB	CTRL-W
024	18	CAN	CTRL-X
025	19	EM	CTRL-Y
026	1A	SUB	CTRL-Z
027	1B	ESC	
028	1C	FS	
029	1D	GS	
030	1E	RS	
031	1F	US	
032	20	SPACE	
033	21	!	
034	22	"	
035	23	#	
036	24	$	
037	25	%	
038	26	&	
039	27	'	
040	28	(	
041	29	)	
042	2A	*	
043	2B	+	
044	2C	,	
045	2D	-	
046	2E	.	
047	2F	/	
048	30	0	
049	31	1	
050	32	2	
051	33	3	
052	34	4	
053	35	5	
053	36	6	
054	37	7	
055	38	8	
056	39	9	
058	3A	:	
059	3B	;	
060	3C	<	
061	3D	=	
062	3E	>	
063	3F	?	
064	40	@	
065	41	A	
066	42	B	
067	43	C	
068	44	D	
069	45	E	
070	46	F	
071	47	G	
072	48	H	
073	49	I	
074	4A	J	
075	4B	K	
076	4C	L	
077	4D	M	
078	4E	N	
079	4F	O	
080	50	P	
081	51	Q	
082	52	R	
083	53	S	
084	54	T	
085	55	U	
086	56	V	
087	57	W	

Decimal	Hex	Character
088	58	X
089	59	Y
090	5A	Z
091	5B	[
092	5C	\
093	5D	]
094	5E	^
095	5F	_
096	60	`
097	61	a
098	62	b
099	63	c
100	64	d
101	65	e
102	66	f
103	67	g
104	68	h
105	69	i
106	6A	j
107	6B	k
108	6C	l
109	6D	m
110	6E	n
111	6F	o
112	70	p
113	71	q
114	72	r
115	73	s
116	74	t
117	75	u
118	76	v
119	77	w
120	78	x
121	79	y
122	7A	z
123	7B	{
124	7C	|
125	7D	}
126	7E	~
127	7F	DEL

Appendix D - Version 1.96 changes

Version 1.96 added a restart command, allowing you to restart the converter without disconnecting power.

Version 1.6 also supports cloning converters. Any converter with version 1.96 or later can be cloned using the utility program.

Appendix E - 716R and 718R Addendum

The 716R and 718R versions of the converters contain an internal relay which can be controlled via software.

The main purpose of the "R" versions is to provide not only a serial-operated relay but also to provide a reader interface. The 716R can be used as a control device, converting data from a prox reader into rs232 for host control, while also providing a relay which can be activated upon host control. In this configuration, simply adding a 716R to a prox reader will give you a controller-like box allowing you to read the converted prox data, and also control a door or device.

The 718R operates in the same way, allowing you to control the reader and relay using a usb connection.

Note that you cannot power a strike or lock through the 716R or 718R, or from the usb interface on a pc. The relay on these units is completely isolated and has formC style contacts. Power for the door/strike/lock must be supplied by a separate supply.

In order to operate IBC usb equipment, a special usb driver is required. This driver can be found on the IBC web page under the software section. This driver sets up a virtual comm port on the pc so that you can communicate with the 718 converter serially through a standard pc comm port.

Users familiar with the 718R product will know that the 718R converter, like the 716R converter, can utilize wiegand/aba inputs only, and no wiegand/aba outputs, since there are not enough pins on the RJ45 jack for wiegand/aba inputs, outputs, and the 3-wire relay form C connections. 718R converters, however, manufactured after June 2008 have an additional set of jumpers, for configuring the 718R to utilize wiegand/aba inputs, or outputs. This allows the 718R to be used in oem configurations where the primary use is a usb (serial) to wiegand conversion, but it is also convenient to have a relay on board as well.

718R converters are shipped configured with wiegand/aba inputs as the defaults.

718R converters are shipped with the usb power jumper OFF. By default, these converters must be powered on the reader side.