Operation and Installation Manual

Model 4284A Series
Digital RF Switch

api technologies corp.
› WEINSCHEL

Table of Contents

1. Safety Summary

1.1. Definitions

The following definitions apply to WARNINGS, CAUTIONS, and NOTICES may found throughout this manual.

WARNING

WARNING: An operating or maintenance procedure, practice, statement, condition, etc., which, if not strictly observed, could result in injury and/or death of personnel. Do not proceed beyond a WARNING symbol until all the indicated conditions have been fully understood and/or met.

CAUTION

CAUTION: An operating or maintenance procedure, practice, statement, condition, etc., which, if not strictly observed, could result in damage or destruction of the equipment or long-term health hazards to personnel. Do not proceed beyond a CAUTION symbol until all the indicated conditions have been fully understood and/or met.

NOTICE

NOTICE: An essential operating or maintenance procedure, condition, or statement that must be highlighted.

1.2. Detailed Precautions

The following WARNINGS, CAUTIONS and NOTES appear throughout the text of this manual and are repeated here for emphasis.

CAUTION

All procedures and/or steps identified as must be followed exactly as written and according to industry accepted ESDS device handling procedures. Failure to comply may result in ESD damage.

1.3. Electrostatic Discharge Sensitive (ESDS)

The equipment documented in this manual contains certain Electrostatic Discharge Sensitive (ESDS) components or parts. Therefore, certain procedures/steps are identified by the use of the symbol . This symbol is used in two ways:

- When the ESDS symbol is placed between a paragraph and title, that paragraph, including all subparagraphs, is considered ESDS device handling procedure.
- When the ESDS symbol is placed between a procedure/step number and the text, all of that procedure is considered an ESDS device handling procedure.

All procedures and/or steps identified as ESDS must be followed exactly as written and according to accepted ESDS device handling procedures. Failure to comply may result in ESDS damage

2. General Information

2.1. Purpose

This manual contains setup and operation information for the Weinschel Model 4284A Digital RF Switch. This manual is to be used in conjunction with the operation and installation of the Model 4284A. The manual also provides a description of the assembly and general maintenance procedures.

2.2. Equipment Overview

The 4284A is a SP4T Digital RF Switch operate over the 10 to 8000 MHz frequency range. It can be controlled either via USB or a variety of digital interfaces via the AUX mode connector, including parallel input (PIO), I2C, SPI, or a logic-level UART interface. AUX mode selection is done via USB command and can be changed via the user.

API Weinschel's LabView based USB Switch Control Software (SCS) can also be used in the operation of this series of digital Switches. The SCS will allow the user to setup, control, and perform test and measurements over a standard USB 2.0 communication interface. Refer to manual IM697 for additional information about the software.

3. Specifications

3.1. Electrical Specifications

Parameter	**Min**	**Typ**	**Max**	**Comments**
DC Power (AUX pin 9)				
VDC Supply Voltage	3.3V	5V	16V	Supply voltage 3.5V MIN for full spec compliance
IDC Supply Current (VDC=5V)		15mA	25mA	
AUX IO	Note: All AUX IO have weak pull-ups enabled by default			
VIH Input High Voltage VDC= 3.3V to 4.5V	2.0V		VDC+0.3	
VIH Input High Voltage VDC= 4.5V to 16V	2.0V		5.0V	
VIL Input Low Voltage VDC= 3.3V to 4.5V	-0.3V		0.15VDC	
VIL Input Low Voltage VDC= 4.5V to 16V	-0.3V		0.8V	
VOH Output High Voltage	2.6V			ILOAD = 3mA
VOL Output Low Voltage			0.6V	ILOAD = 3mA
IPU Pullup Current	25uA	130uA	300uA	User selectable
USB				
USB Supply Voltage (VBUS)	4.4V		5.25V	
D+/D- Input Voltage			3.6V	

3.2. Additional Specifications

Parameter	**Comments**
RF switching speed	+5 µs. (50% VCTL to 90% RF)
Control logic	PARALLEL, I2C, SPI, UART or USB
Operating Voltage	+3.3 to +16 VDC @ 25 mA
Supply current	+25 mA max
Temperature Range	-20° C to +85° C
CW power handling	+30dBm
RF connectors	SMA Female

Parameter	**Comments**
Control Connectors	The AUX control connector is an AMP-Latch 10-pin ribbon cable connector that mates with AMP P/N 746285-1 (supplied with each unit). The USB connector is a standard USB Mini-B.
Weight	83 g (2.92 oz.)

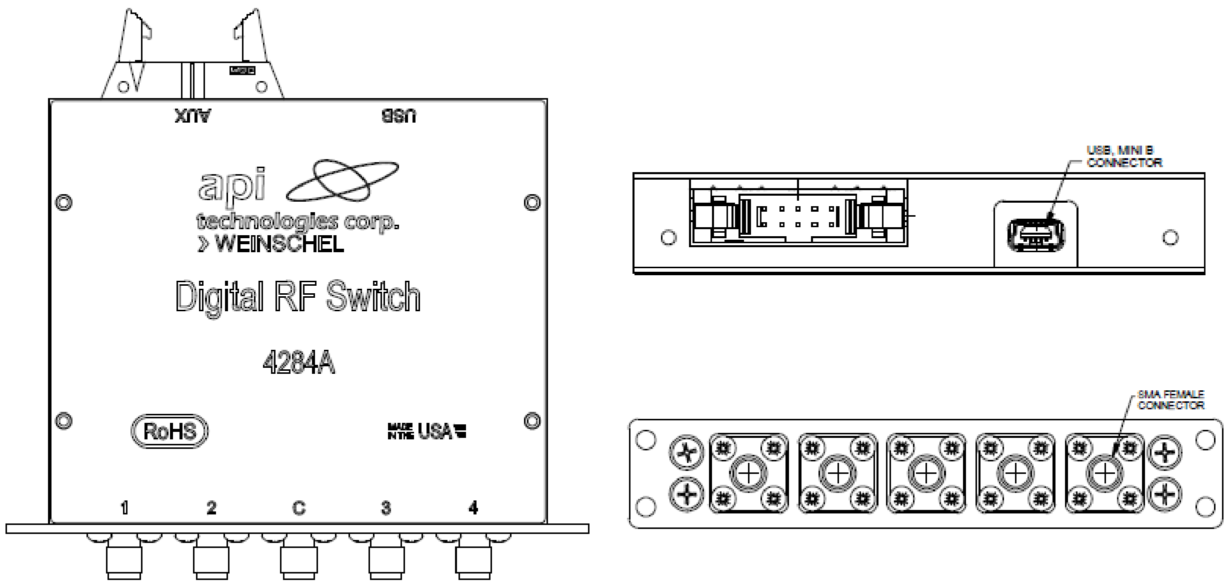


Figure 1: Front and side views of the model 4284A

3.3. DC Power Input

The 4284A can be powered from either the USB VBUS (5V) or the AUX VDC input. While USB operates at a nominal 4.75V-5.25V range, the AUX VDC supply input can accept a wider range of voltage, from 3.3V-16VDC. If both AUX power and USB VBUS are present then the 4284A will be powered from whichever provides the higher voltage. For AUX VDC voltages < 5V the input logic signals are limited to the VDC supply voltage. Otherwise, input logic signals are limited to a max voltage of 5V.

4. Installation and Operation

4.1. Mounting

Each Digital Switch is supplied with 16 mounting holes. The front and the back side of the switch contains a total of 8 mounting holes and each side contains four (2-56 UNC-2B x 4.1 [0.16 Deep]). Each of the rest of the sides contain 2 mounting holes (2-40 UNC-2B x 4.1 [0.16 Deep]). Refer to the appropriate Weinschel Specification/ICD drawing for the mounting hole locations

CAUTION

- When applying a signal to the RF connectors, DO NOT exceed the maximum allowable power level specifications of the unit.
- Do not over torque the SMA connectors more than 10 inch pounds. Damage may occur.

4.2. RF Connectors & Cable Installation

The Model 4284A contains five SMA female connectors labeled as 1, 2, C, 3, and 4 that mate nondestructively with SMA male connectors per MIL-STD-39012. Weinschel recommends a torque value of 7 to 8 inch pounds when connecting any cable to the Switch's RF connectors.

4.3. Control Connectors

4.3.1 AUX mode digital IO (10-pin 0.1" Header)

A variety of control interfaces can be used with the AUX Connector. Options include Parallel IO, I2C, SPI, UART, and USB. The SET AUX command allows the user to select the control interface for the AUX Connector. The table below describes the pinouts for the various control modes.

PIN	SIGNAL	PIO[1]				I2C	SPI	UART	USB
		Stage 1	Stage 2	Stage 3	Stage 4				
1	D0	1	0	0	0	A0	--	--	
2	D1	0	1	1	1	A1	--	--	
3	D2	0	0	1	0	A2	--	RXD	
4	D3	0	0	0	1	A3	--	TXD	
5	D4	--	--	--	--	TRIG	SSN	--	
6	D5	--	--	--	--	RESETN	SCLK	--	
7	D6	--	--	--	--	SCL	SDI	--	
8	D7	--	--	--	--	SDA	--	--	BOOTN
9	VDC	VDC	VDC	VDC	VDC	VDC	VDC	VDC	
10	GND	GND	GND	GND	GND	GND	GND	GND	GND

1. PIO Mode: Digital input low, turns OFF the desired path and digital input high, turns ON the desired path.

4.3.2 USB Mini-B

The table below lists the pinout of the USB connector.

PIN	SIGNAL	DESCRIPTION
1	VBUS	+5V
2	D-	Data-
3	D+	Data+
4	ID	unused
5	GND	Ground

4.4. USB/AUX Mode Interface Selection

The main operating mode of the 4284A is determined from the DC Power input. At power on the USB connector VBUS pin is examined, and if detected then the unit will operate in USB mode. Otherwise the 4284A will operate in one of the digital AUX modes powered via the AUX VDC power input. It is allowable to have both cables connected at the same time.

If an AUX mode is currently active the unit will detect a USB connect event and switch over to USB mode automatically.

Typically you would return to AUX mode by disconnecting the USB cable (or removing USB power). The USB command RUN AUX also allows switching from USB mode to an AUX mode via command, and does not require the AUX connector VDC power to be present.

4.5. AUX Interface Modes

There are four user-selectable digital interface AUX modes: PIO, I2C, SPI, and UART. The AUX mode selection is done via USB command (see SET AUX) and is stored in non-volatile memory (NVM) so that changes to the mode will be automatically applied at startup. The AUX digital interface pins vary in function depending on the selected mode. Each pin can have a software programmable weak pullup assigned, which is enabled by default for all pins (see SET WPU). The weak pull-up will provide a logic high to the pin if left unconnected.

4.5.1 PIO Mode

In PIO mode there are up to four parallel digital input signals, D0-D3. Each input represents a switch setting, with a logic low input = stage 1 setting and a logic high = stage 4 value for each control input as shown in the PIO column of the AUX mode table.

4.5.2 SPI Mode

SPI mode is a serial interface that operates as a 16-bit serial-in shift register and latch comprised of three signals: SSN low-active chip select, SCLK serial shift clock, and SDI serial data in. Data present on the SDI input is clocked into the shift register on the rising edge of SCLK. Data is comprised of 16-bits of programming data. The data should be left-justified in the 16-bit word so that the MSB is the first bit sent and any unused bits should be set to 0. Serial data is clocked in MSB first to LSB and must be in multiples of 8-bits. SSN must be asserted low before sending data to the Switch, allowing multiple Switches to be controlled via the same SCLK and SDI signals.

4.5.3 I2C Mode

I2C mode is a serial interface that uses two lines: SCL serial clock and SDA serial data, along with the optional controls RESETN, TRIG, and address bits A3-A0. Both the SCL and SDA connections are bidirectional open-drain lines, each requiring pull-up resistors to the logic supply voltage (5V max).

I2C messages consist of a device address byte, register select byte, and one or more data bytes depending on the register. The Register address will automatically increment after each byte transferred. Messages are framed using the standard I2C START, STOP, and ACK conditions. The I2C master should support clock stretching as the 4284A will hold the SCL clock low during the byte ACK phase until the data is accepted by the 4284A. The table below shows the I2C packet formatting.

START	DEV ADDR	REG ADDR	DATA	<DATA>	STOP

The 4284A is a slave I2C device that supports 7-bit slave addressing. The slave address can be set via hardware address pins A3-A0 on the AUX connector or via USB command (see SET I2CADDR). Using the hardware address pins allows for up to 16 Switches to share the same bus. In this mode the three upper bits of the address byte are fixed at 0b010. The I2C R/W bit is the LSB of the address byte, providing for device addresses 0b0100000x – 0b0101111x (0x40-0x5E). The table below shows the format of I2C 7-bit addressing.

I2C Device Address								
7	6	5	4	3	2	1	0	
0	1	0	A3	A2	A1	A0	R/W	010 : fixed bits A3-A0 : addr bits R/W bit : WR=0, RD=1

A device address can also be assigned using the USB SET I2CADDR command which allows the use of all 7 D7-D1 address bits, with the exception of the reserved address 0. A software assigned address overrides the hardware A3-A0 pins and connections to these pins are ignored. When specifying a software address always use the full 8-

bit byte value, with bit 0 set to 0 (it will be ignored as this is the I2C R/W bit). Setting the I2CADDR to 0 will remove any software assigned address and revert back to hardware addressing mode.

I2C mode provides two optional control inputs: RESETN and TRIG. RESETN is a low-active signal that will reset and reinitialize the Switch. The TRIG signal allows switching position changes to be performed on the TRIG input becoming asserted instead of changing immediately when the I2C command is sent, and can be programmed to be active-high or active-low (see I2CTRIG command). This can be used to synchronize multiple Switches.

4.5.4 UART Mode

UART mode is an asynchronous full-duplex serial interface consisting of two signals: RXD receive data in and TXD transmit data out. This provides a logic-level "COM port" style interface that can be used directly with most serial terminal emulators and control programs. The interface provides user-selectable standard baud rates from 9600 to 115200 (see SET BAUDRATE command) with a fixed data format of no parity, 8 data bits, 1 stop bit (N81).

This mode uses the same ASCII text-based messages and commands as the USB CDC interface.

4.6. AUX Application Modes

4.6.1 USB Mode AUX pin usage

When the 4284A operates in USB mode, AUX connector pin 8 (BOOTN) is used as a boot select pin. When power is first applied via USB VBUS, the state of the BOOTN pin is checked. If BOOTN is a logic-low level then the Switch powers up as a USB HID device (USB VID=25EA, PID=003C) into a special bootloader mode that can be used to download firmware updates. For normal USB operation leave the AUX pins unconnected. Consult with the factory for more information on performing program updates.

4.6.2 USB

In USB mode the Switch is controlled and powered via a standard USB 2.0 connection to a USB host. The 4284A operates as a USB CDC device (USB VID=25EA, PID=106D), so it may be controlled via any software that can communicate to a standard virtual COM port. Programming is done via simple ASCII text-based message strings to control the device (see the Command section later).

For ease of use, the 4284A has two modes of operation: console and raw mode. Console mode provides a simple command-line based interface that can be used in conjunction with any standard terminal emulator program. Console mode sends command prompts ('>'), echoes received characters, issues error messages, and supports the backspace key for simple editing, while raw mode is more suitable for programming. By default, the unit is shipped with Console mode enabled, but this operation can be change by the user (see the CONSOLE command for more details). A typical Console mode display is shown below:

```
>
API Weinschel 4284A USB RF switch V1.00
firmware: 194177301A
serialno: D88039DFD31D
alias: none

RF config: SP4T, 0, 4, 10MHz-8GHz

>help
```

```
       *CLS, *ESR?, *IDN?, *OPC?, *RST, *TST?, ERR?

       RFSW val
       RFSW?
       INCR
       DECR
       SEQ width interval count
       SEQ? width interval count

       ALIAS?
       DELAY msec
       REPEAT n
       RFCONFIG?

       CONSOLE [ENABLE|DISABLE|ON|OFF]
       CONSOLE?
       SET AUX [PIO|SPI|I2C|UART|SEQ]
       SET USB [CONNECT|PMT|RMT] val
       SET [ALIAS|BAUDRATE|RFSW|I2CADDR|I2CTRIG|PINOUT|WPU] val
       SET SEQ COUNT val
       SET SEQ [WIDTH|INTERVAL|TIME] val[ms|us]
       SHOW [SET|VERSION]
       FACTORY PRESET
       SYSTEST [EXT|PIO|PIO?|XSUM]
       REBOOT
       RUN [AUX|LOADER]
```

5. Command Operation

Commands are comprised of text-based ASCII strings. The command parser is case-insensitive, so either upper or lower case characters are acceptable. Command parameters may be separated with either an ASCII SPACE char (0x20) or an ASCII COMMA char (0x2E), but the separator character used must be the same within an individual command string. Additional SPACE characters are ignored. Input program messages may be terminated using either an ASCII CR character (0x0D) or an ASCII LF character (0x0A). Command message strings are limited to 128 characters total, including the terminator. Multiple commands can be included in one message by separating the individual commands with an ASCII SEMICOLON character ';' (0x3B), up to the 128 character message limit. Typically, Response messages sent from the device are terminated using both a CR (0x0D) and LF (0x0A) to terminate the message. The output terminator sequence may be changed using the RMT command. A list of supported commands can be seen by typing 'HELP' at the Console prompt.

The command structure/operation is similar to that used in IEEE 488.2, and includes some of the 488.2 Common Commands such as *IDN?, *RST, *CLS, and *OPC?, in addition to device specific commands. In 488.2, programming commands take one of two forms: a Program message or a Query message. Program messages are used to send commands to the device, while Query messages are used to elicit a response. Query commands are those that contain a '?' character. In general, the device does not generate any response to a program message unless the message contains a valid Query command. (Note that this does not apply when operating in Console mode, or when using some commands such as HELP which are designed to provide the user general information). You can use this feature to provide a method to synchronize command execution with the controller by appending a Query to the desired command, and waiting for the response. For example, sending "*CLS;*OPC?" will place a "1" in the output queue when the *CLS command has been executed. Query commands that return multiple values will have the values separated by an ASCII COMMA character (0x2E). If multiple Query commands are included in the same message, the individual query responses will be separated with an ASCII SEMICOLON character (0x3B).

Commands that loop or repeat (such as FADE and REPEAT) can be terminated by sending a BREAK condition, which is supported in both USB and UART modes. For USB, a BREAK is defined by the CDC class SEND_BREAK request code, while for UART mode a BREAK occurs when the sender's TX line is held at a logic 0 for longer than one frame time (11 bits).

An Error Queue is provided that logs the results of command/execution errors in a FIFO fashion. The queue entries can be read using the ERR? command, which returns both an error code and a descriptive text message, such as

```
101, "invalid command"
```

When the queue is empty, ERR? returns the message `0, "no error"`. The queue can be emptied by repeatedly sending ERR? until all entries are read from the queue, or via sending the *CLS message.

Unless otherwise specified, commands revert to their default setting at system reset/poweron, with the exception of the system setup and configuration commands which store their setting in non-volatile memory (NVM).

5.1. Command Reference

In the command descriptions that follow, argument types are described using the following additional conventions to indicate the relative size of the parameter:

Argument Type	**Relative Size**
Byte	Used to indicate an 8-bit unsigned integer
Word	Used to indicate a 16-bit unsigned integer
Int8	8-bit integer
Int16	16-bit integer
Int32	32-bit integer
String	Character data, including the max number of characters allowable. (ie string8 has a max of 8 chars)

Numeric arguments default to decimal (base 10) notation, but may optionally be provided in hex if appropriate by using a "0x" prefix (ie 0x0A = 10 dec)

Required command keywords are shown in CAPITAL letters, and arguments are shown in *italics*. Square brackets '[]' may be used to indicate a selection or optional parameter, for example [*select*]. Optional parameters, if not supplied by the user, assume the default setting specified in the text.

5.2. Application Specific Commands

RFSW

Function: set Switch
Syntax: RFSW *value*
Argument(s): *value* Switch setting, in integer *value* =1-4 positions
Remarks: This command sets the RF Switch to it's four different stages. If *value* is 1, then the switch will be switched to stage 1.
Return Value: none
Example(s):

```
RFSW 1       // Switch switches to stage 1
RFSW 4       // Switch switches to stage 4
```

RFSW?

Function: read Switch setting
Syntax: RFSW?
Argument(s): none
Remarks: This command returns the current setting of the Switch
Return Value: Switch setting
Example(s):

```
RFSW 1       // Switch switches to stage 1
RFSW?        // read switch setting
1            // returns switch setting (stage 1)
```

INCR

Function: increment Switch setting
Syntax: INCR
Argument(s): none
Remarks: This command increments the current setting of the Switch by the STEPSIZE setting.
Return Value: none
Example(s):

```
RFSW 1       // sets switch to position 2.
INCR         // increments RFSW value by 1 and sets switch to stage 2.
```

DECR

Function: decrement Switch setting
Syntax: DECR
Argument(s): none
Remarks: This command decrements the current setting of the Switch.
Return Value: none
Example(s):

```
RFSW 2       // sets switch to position 2.
DECR         // decrements RFSW value by one and sets switch to stage 1.
```

5.3. 488.2 Common Commands

***CLS**

Function: clears the error status
Syntax: *CLS
Argument(s): none
Remarks: This function clears the Error Queue
Return Value: none
Example(s):

```
*CLS
```

***IDN?**

Function: Reads the system identification information
Syntax: *IDN?
Argument(s): none
Remarks: This function is used to read the system identification info, which is a string consisting of the following data: manufacturer, model, serial number, and firmware version.
Return Value: *idstr* string id info
Example(s):

```
*IDN?
API Weinschel, 4284A, 0004A3DB3013, V1.40
```

***OPC?**

Function: Operation complete query
Syntax: *OPC?
Argument(s): none
Remarks: This function loads a '1' into the output queue when the Program Message Unit is executed. Its primary use is to provide an indication of command completion by including the command as the last one in a series of commands. It can be useful to synchronize operation and to prevent input buffer overflow.
Return Value: 1 integer constant command completed
Example(s):

```
RFSW 1; RFSW 2; *OPC?
1     // sends a '1' response when the three commands have been executed
```

***ESR?**

Function: Event Status Register query
Syntax: *ESR?
Argument(s): none
Remarks: This function reads the 488.2 Event Status Register. Reading the register also clears it.
Return Value: *int8* integer status register
Example(s):

```
*ESR?
128          // indicates a Command Error
```

***RST**

Function: Performs a device application level reset.
Syntax: *RST
Argument(s): none
Remarks: This function is used to reset the device application settings.
Return Value: none
Example(s): `*RST`

***TST?**

Function: Self-test query
Syntax: *TST?
Argument(s): none
Remarks: This function performs an internal self-test. Upon completion, the results of the test are loaded into the output queue.
Return Value: *testresults* integer '0' indicates test passed. Non-zero indicates test failed.
Example(s):

```
*TST?
0            // returns a '0' when the test completes successfully.
```

ERR?

Function: Read the Error Queue
Syntax: ERR?
Argument(s): none
Remarks: This function returns the last entry in the error status queue, and a string description of the error code. Repeating the command will return the next entry, until the error queue is empty and returns a zero. The error queue may be cleared via the *CLS command. Note that when using the command-line interface the Error Queue contents are automatically displayed after each command prior to issuing the CLI prompt.
Return Value: error number, "error description"
Example(s):

```
ERR?
101, "invalid command"
ERR?
0, "no error"
```

5.4. Setup and Configuration Commands

NOTE: The SET commands are used to update settings which are stored in non-volatile memory (NVM), and do not typically take effect until the next poweron or restart event (see REBOOT) unless otherwise noted. The current SET parameter values can be viewed using SHOW SET.

SET AUX

Function: Sets the AUX mode function

Syntax: SET AUX *mode*

Argument(s): *mode* PIO, SPI, I2C, UART, SEQ

Remarks: This command sets the AUX mode interface and/or AUX application function. The default mode is PIO.

Return Value: none

Example(s):

```
SET AUX I2C  // set AUX mode to I2C
```

SET USB

Function: Sets USB connect and Message Terminator characters

Syntax:

SET USB CONNECT *msecs*	Connection time delay to console signon message output
SET USB PMT *val*	Program Message Terminator (input)
SET USB RMT *val*	Response Message Terminator (output)

Argument(s): *val* word, eos characters

Remarks: These commands set the output Response Message Terminator (RMT) and Program Message Terminator (PMT) sequences. The *val* parameter specifies the character sequence used, and can specify up to two characters, typically as a hex word high byte-low byte pair. Common definitions for the terminators include the ASCII CR (0x0D) and LF (0x0A) characters. A single character may be specified either by using 0 for the high byte, such as 0x000D, or by only specifying a single character (ie 0x0D). On output the characters are sent low byte then high byte, unless it is specified as 0. Note that the CONSOLE mode will always use a fixed CRLF (0x0A0D) sequence.
The CONNECT function sets the delay for the signon message display shown in CONSOLE mode when a connection is detected.

Return Value: none

Example(s):

```
SET USB RMT 0x0A0D        // set output sequence as CR-LF
SET USB RMT 0x0D          // set output sequence as a single CR character
```

SET ALIAS

Function: Sets user-defined ALIAS string

Syntax: SET ALIAS *name*

Argument(s): *name* character string (max length of 8)

Remarks: This command sets a user-defined string value that is returned by the ALIAS? command. It can be used to help identify Switchs when used in multiple setups. Alpha-numeric values are allowed, up to a max of 8 characters.
To remove an existing alias send SET ALIAS with no parameter.

Return Value: none

Example(s):

```
SET ALIAS 1234
ALIAS?
```

```
1234
```

SET BAUDRATE

Function: AUX UART serial port baud rate setting
Syntax: SET BAUDRATE *rate*
Argument(s): *rate* 9600, 19200, 38400, 57600, and 115200 (default)
Remarks: This function sets the baud rate for the AUX mode UART serial port. This command takes effect immediately.
Return Value: none
Example(s):

```
SET BAUDRATE 115200
```

SET RFSW

Function: sets default power on RFSW setting
Syntax: SET RFSW *settings*
Argument(s): *settings* any valid switch setting (1 – 4 for SP4T)
Remarks: This command sets the default poweron RFSW setting.
Return Value: none
Example(s):

```
SET RFSW 1
```

SET I2CADDR

Function: sets the AUX mode I2C slave address
Syntax: SET I2CADDR *addr*
Argument(s): *addr* I2C slave address byte
Remarks: This command sets the AUX mode I2C address. The addr parameter can be any even number value from 0-254 (bit 0 must be 0 as this is the I2C R/W bit). Setting I2CADDR = 0 enables the AUX A3-A0 hardware address pins.
Return Value: none
Example(s):

```
SET I2CADDR 0x6E
```

SET I2CTRIG

Function: sets the AUX mode I2C external trigger function
Syntax: SET I2CTRIG *mode*
Argument(s): *mode* bit 0 = trigger enable/disable: 0=disabled, 1=enabled
bit 1 = trigger edge: 0=falling/negative edge, 1=rising/positive edge
Remarks: This command sets the AUX mode I2C external trigger function.
Return Value: none
Example(s):

```
SET I2CTRIG 0x03          // trig enabled, rising edge
```

SET WPU

Function: sets the AUX weak pullups
Syntax: SET WPU *pin_mask*
Argument(s): *pin_mask* byte, 0-255 (default)
Remarks: This command controls the setting of the AUX connector weak pullup function. Setting a bit=0 disables the pullup, and bit=1 enables the pullup for that pin. Bit 0 is the setting for pin 1 D0, and bit 7 for pin 8 D7. Weak pull-ups on all pins are enabled by default.
Return Value: none
Example(s):

```
SET WPU 0x0F // enable pull-ups for pins 1-4 (I2C A0-A3)
```

RUN AUX

Function: Runs the AUX mode selection
Syntax: RUN AUX
Remarks: This command switches the unit from USB operation into the AUX mode selected via SET AUX. It will shut down the USB connection and reboot the unit into AUX mode, allowing AUX mode to run via USB power.
Example(s):

```
>RUN AUX                  // reboot into AUX mode, starting PULSE mode
disconnecting USB. rebooting in AUX mode...
```

SHOW SET

Function: display all SET parameters
Syntax: SHOW SET
Argument(s): none
Remarks: This command displays all non-volatile SET parameters.
Example(s):

```
>SHOW SET
console: 1
aux: 0, PIO
baudrate: 115200
rfsw: 0
i2caddr: 0x00, using D3-D0
i2ctrig: 0, trig disable
wpu: 0xFF
pinout: 0
pmt: 0x0A0D
rmt: 0x0A0D
connect: 500
alias: none
seq_width: 10us
seq_interval: 0
seq_count: 20
seq_time: 1000
SSPADD: 0x00
```

SHOW VERSION

Function: displays firmware version
Syntax: SHOW VERSION
Argument(s): none
Remarks: This command displays the firmware version and serial number information
Example(s):

```
>show version

API Weinschel 4284A USB RF switch V1.00
firmware: 194177301A
serialno: D88039DFD31D
alias: none
```

FACTORY PRESET

Function: initializes non-volatile memory
Syntax: FACTORY PRESET
Argument(s): none
Remarks: This command erases all user-modifiable non-volatile memory, which sets the memory to all 1's (0xFF). On the next reset/reboot, the memory will be initialized with factory default settings. This can be used to clean the device in secure environments.

Example(s):

```
>factory preset

>reboot
API Weinschel 4284A USB Attn V1.40
firmware: 1012532301C
serialno: 0004A3DB3013
alias: none

RF config: 4284A-95.5, 95.75, 0.25, 300KHz-6GHz
error 31: nvm format
error 32: nvm defaults
```

5.5. Misc. Commands

ALIAS?

Function: read user-assigned alias string
Syntax: ALIAS?
Argument(s): none
Remarks: This command returns the current alias name string (see SET ALIAS). If no alias has been assigned then the command returns 'none'.
Example(s):

```
ALIAS?
none
SET ALIAS "AT101B"; ALIAS?
AT101B
```

CONSOLE

Function: Console mode enable
Syntax: CONSOLE *mode*
Argument(s): *mode* byte 0, 1, 2, 3 or OFF, ON, ENABLE, DISABLE
Remarks: This function enables/disables the console mode command-line interface and optionally updates the nvm setting. Setting *mode*=0 turns console off, *mode*=1 turns console on, *mode*=2 enables the console, and *mode*=3 disables the console. Modes 0 and 1 (OFF and ON) update the nvm setting, while modes 2 and 3 (ENABLE and DISABLE) do not.
Return Value: none
Example(s):

```
CONSOLE ON                 // turns on the console and updates nvm setting
CONSOLE 0                  // turns off the console and updates nvm setting
CONSOLE ENABLE             // turns on console for this session only
CONSOLE DISABLE            // turns off console for this session only
```

CONSOLE?

Function: Console mode query
Syntax: CONSOLE?
Argument(s): none
Remarks: This function returns the console mode nvm setting
Return Value: *nvm* integer
Example(s):

```
CONSOLE?
1
```

DELAY

Function: Delays execution (pause)
Syntax: DELAY *msecs*
Argument(s): *msecs* word, 0-65535 in msecs
Remarks: This command pauses execution for the specified time in msecs.
Return Value: none
Example(s):

```
RFSW 1; DELAY 100; RFSW 2       // waits 100 msecs between RFSW commands
```

REBOOT

Function: system reset
Syntax: REBOOT
Argument(s): none
Remarks: This command performs a system reboot, similar to a poweron reset.
Return Value: none
Example(s):

```
>reboot
API Weinschel 4284A USB RF switch V1.00
firmware: 194177301A
serialno: D88039DFD31D
alias: none

RF config: SP4T, 0, 4, 10MHz-8GHz
```

REPEAT

Function: Enables command repetition/looping
Syntax: REPEAT [*count]*
Argument(s): *count* word, 1-65535
Remarks: This function causes the remainder of the current program message to be repeated *count* number of times. Omitting the *count* parameter or specifying REPEAT 0 will result in the maximum number of iterations. Any commands in the program message prior to REPEAT are executed only once. The operation can be terminated via a BREAK condition.
Return Value: none
Example(s):

```
rfsw 1; REPEAT 50; DELAY 100          // set rfsw to 1, repeats INCR and DELAY
                                      50 times
```

RFCONFIG?

Function: read current RF configuration
Syntax: RFCONFIG?
Argument(s): none
Remarks: This command displays the current Switch configuration, including the model, max attn, default stepsize, and frequency range
Example(s):

```
RFCONFIG?
SP4T, 0, 4, 10MHz-8GHz
```

RUN LOADER

Function: runs the USB HID bootloader function
Syntax: RUN LOADER
Remarks: This command forces a reboot into the USB HID bootloader for downloading program updates. The HID bootloader requires an external program for downloading the update .hex file. Consult the factory for more information.
Example(s):

```
>RUN LOADER  // invokes the USB HID bootloader for update
```

```
disconnecting USB. rebooting in HID mode...
```

SYSTEST

Function:	system test functions
Syntax:	see below examples
	SYSTEST displays various voltage and device states
	SYSTEST EXT performs a loopback test on the external AUX connector (requires ext connections)
	SYSTEST PIO sets the AUX PIO pin states
	SYSTEST PIO? reads the AUX PIO pin states
	SYSTEST XSUM performs a checksum on the internal program flash memory
Remarks:	This command performs various selftest functions.

NOTE: After using SYSTEST it is recommended that the unit be reset for normal operation.

Example usage:

SYSTEST

Displays USB VBUS voltage, AUX VDC voltage, serial number device check, and the AUX pin states.

```
>systest
vbus: 5160mV
aux vdc: 16mV
unio device: detected
aux pio: 0b11111111
```

SYSTEST EXT

Performs a loopback test on the external AUX connector (requires ext connections).
A return value of 0 indicates the loopback test passed, and any other value indicates a failure.
NOTE: this test drives the AUX PIO pins as outputs. Do NOT connect external signals to the AUX connector while running this test.

```
>systest ext
0
```

SYSTEST PIO *byte*

Sets the state of the AUX D7-D0 pins to the *byte* value specified.
NOTE: This command drives the AUX PIO pins as outputs.

```
>systest pio 0x55
>systest pio 0xAA
```

SYSTEST PIO?

Reads the state of the AUX D7-D0 pins.

```
>systest pio?
aux pio: 0b11111111
```

SYSTEST XSUM

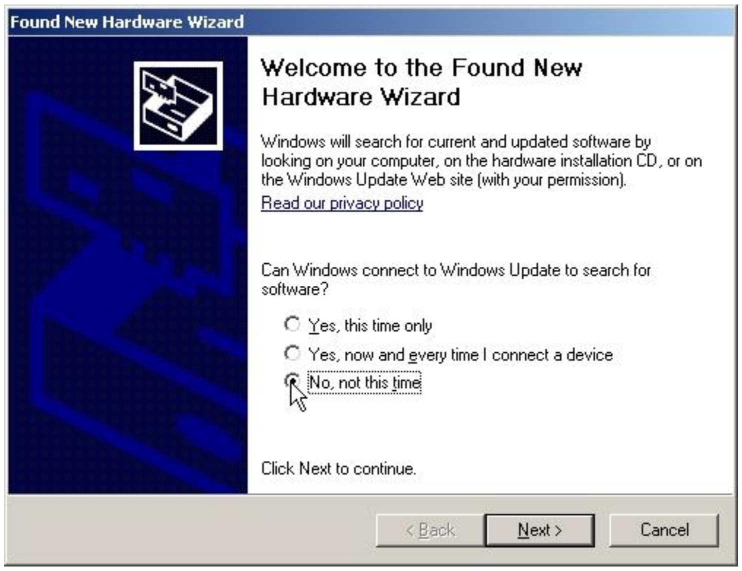
Performs a checksum on the internal program flash memory.

```
>systest xsum
xsum: 0xB108
```

6. USB Driver Installation

When you connect a 4284A to a computers USB port for the first time, you should be presented with the New Hardware Wizard. Follow the steps shown below to install the USB CDC inf file.

NOTE: A copy of the INF information is included in paragraph 3-6 of this document. If you do not have an electronic copy, you can create one using Notepad. Copy and paste the information into Notepad and save it as a plain text file with the name awusbcdc.inf



Found New Hardware Wizard

Welcome to the Found New Hardware Wizard

Windows will search for current and updated software by looking on your computer, on the hardware installation CD, or on the Windows Update Web site (with your permission).

Read our privacy policy

Can Windows connect to Windows Update to search for software?

○ Yes, this time only
○ Yes, now and every time I connect a device
[✓ No, not this time]

Click Next to continue.

Found New Hardware Wizard

Welcome to the Found New Hardware Wizard

Windows will search for current and updated software by looking on your computer, on the hardware installation CD, or on the Windows Update Web site (with your permission).

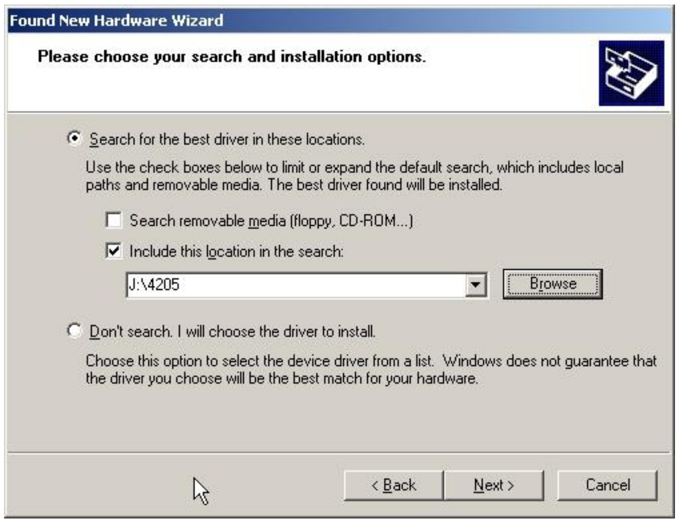
Read our privacy policy

Can Windows connect to Windows Update to search for software?

○ Yes, this time only
○ Yes, now and every time I connect a device
[✓ No, not this time]

Click Next to continue.

Navigate to the drive/folder containing the awusbcdc.inf file, and select 'Next'


Found New Hardware Wizard

Please choose your search and installation options.

(●) Search for the best driver in these locations.

Use the check boxes below to limit or expand the default search, which includes local paths and removable media. The best driver found will be installed.

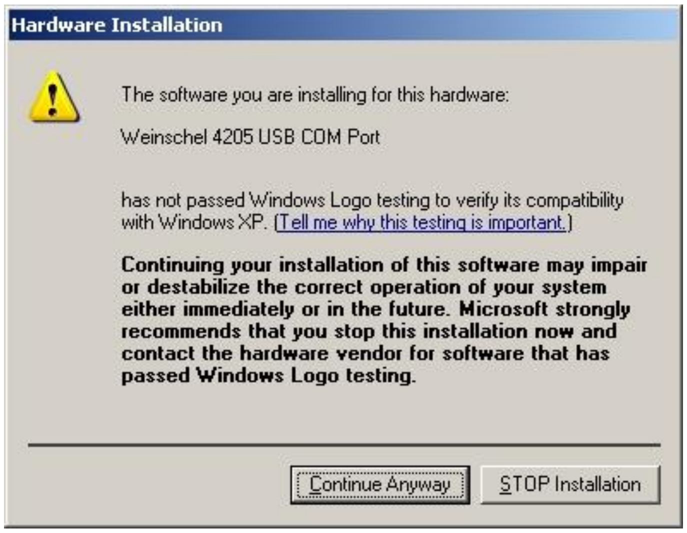
☐ Search removable media (floppy, CD-ROM...)

☑ Include this location in the search:

J:\4205

( ) Don't search. I will choose the driver to install.

Choose this option to select the device driver from a list. Windows does not guarantee that the driver you choose will be the best match for your hardware.

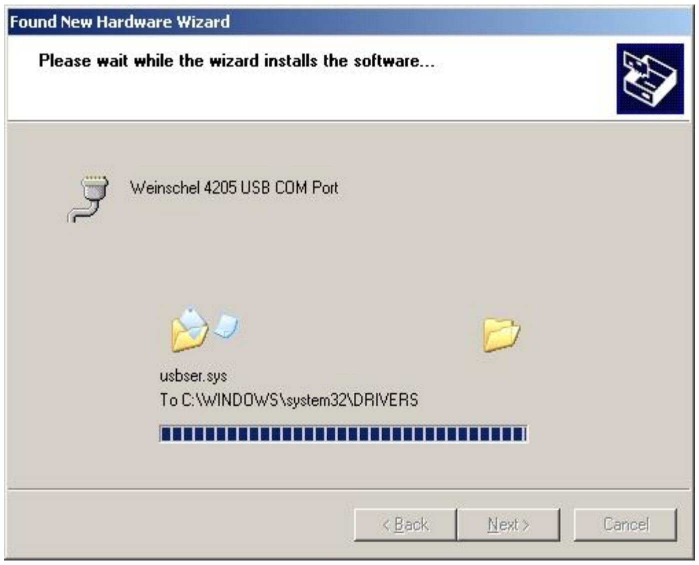

Hardware Installation

The software you are installing for this hardware:

Weinschel 4205 USB COM Port

has not passed Windows Logo testing to verify its compatibility with Windows XP. (Tell me why this testing is important.)

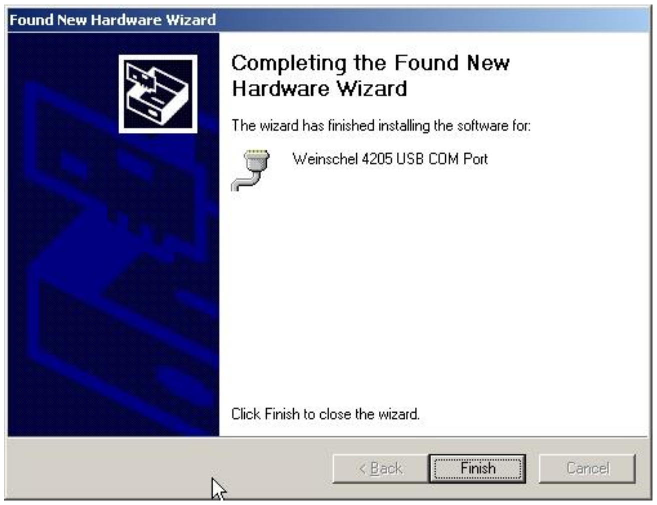
Continuing your installation of this software may impair or destabilize the correct operation of your system either immediately or in the future. Microsoft strongly recommends that you stop this installation now and contact the hardware vendor for software that has passed Windows Logo testing.


Found New Hardware Wizard

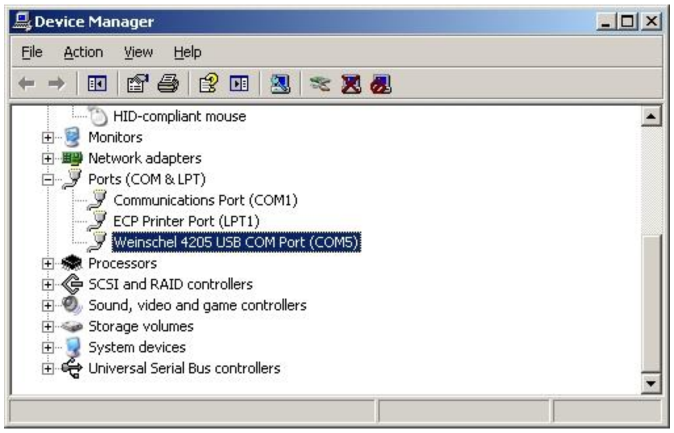
Please wait while the wizard installs the software...

Weinschel 4205 USB COM Port

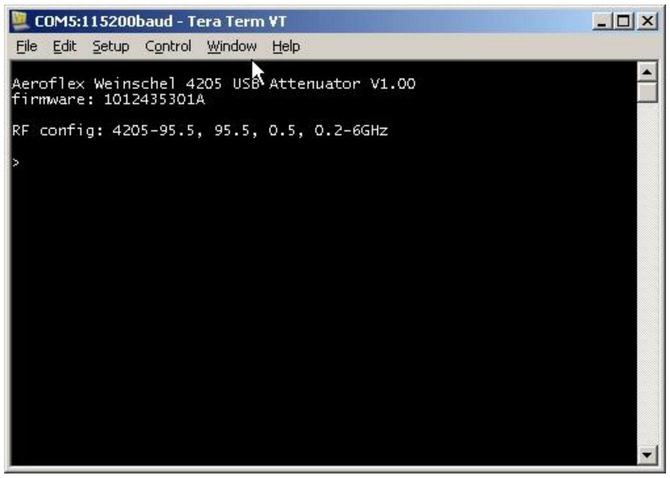
usbser.sys
To C:\WINDOWS\system32\DRIVERS


Found New Hardware Wizard
Completing the Found New Hardware Wizard
The wizard has finished installing the software for:
Weinschel 4205 USB COM Port
Click Finish to close the wizard.
< Back
Finish
Cancel

To verify that the driver has installed properly, view the Ports section in Device Manager. You should see the 4284A listed as a USB COM port. Note the assigned COM port number.



Using a terminal emulator, open a connection to the COM port shown above. The default COM port settings should be acceptable, as these are unused by the 4284A. If the 4284A is in Console mode (the default), you should see the sign on message.



```
Aeroflex Weinschel 4205 USB Attenuator V1.00
firmware: 1012435301A

RF config: 4205-95.5, 95.5, 0.5, 0.2-6GHz

>
```

The COM port numbers are assigned by Windows based on the device USB VID, PID, and Serial Number. The 4284A uses VID 0x25EA and PID 0x106D. The Switchs are shipped with the USB serial number automatically assigned by the microcontroller. This serial number is a different number than that of the unit as a whole.

6.1. awusbcdc.inf Installation File

```
;------------------------------------------------------------------------------
;Note: When the driver package is signed, any modifications to this .inf file will
;break the signature, and the driver package will need to be re-signed.
;------------------------------------------------------------------------------
; Modified Windows USB CDC Abstract Control Model Serial Driver Setup File
; Copyright (C) 2014 Aeroflex Weinschel
; Copyright (C) 2012 Microchip Technology Inc.

[Version]
Signature="$Windows NT$"
Class=Ports
ClassGuid={4D36E978-E325-11CE-BFC1-08002BE10318}
Provider=%MFGNAME%
CatalogFile=%MFGFILENAME%.cat
DriverVer=01/04/2013,5.2.2800.1

[Manufacturer]
%MFGNAME%=DeviceList,NTamd64

;------------------------------------------------------------------------------
;  Vendor and Product ID Definitions
;------------------------------------------------------------------------------
[DeviceList]
%DESCRIPTION%=DriverInstall, USB\VID_25EA&PID_106D         ; 4205-95.5
%DESCRIPTION%=DriverInstall, USB\VID_25EA&PID_106E         ; 4205-63.5
%DESCRIPTION%=DriverInstall, USB\VID_25EA&PID_106F         ; 4205-31.5
%DESCRIPTION%=DriverInstall, USB\VID_25EA&PID_206C         ; 83xx
%DESCRIPTION%=DriverInstall, USB\VID_25EA&PID_4157         ; generic serial CDC

[DeviceList.NTamd64]
%DESCRIPTION%=DriverInstall, USB\VID_25EA&PID_106D         ; 4205-95.5
%DESCRIPTION%=DriverInstall, USB\VID_25EA&PID_106E         ; 4205-63.5
%DESCRIPTION%=DriverInstall, USB\VID_25EA&PID_106F         ; 4205-31.5
%DESCRIPTION%=DriverInstall, USB\VID_25EA&PID_206C         ; 83xx
%DESCRIPTION%=DriverInstall, USB\VID_25EA&PID_4157         ; generic serial CDC

;------------------------------------------------------------------------------
;  Windows 32bit OSes Section
;------------------------------------------------------------------------------
[DriverInstall.nt]
include=mdmcpq.inf
CopyFiles=FakeModemCopyFileSection
AddReg=DriverInstall.nt.AddReg

[DriverInstall.nt.AddReg]
HKR,,DevLoader,,*ntkern
HKR,,NTMPDriver,,%DRIVERFILENAME%.sys
HKR,,EnumPropPages32,,"MsPorts.dll,SerialPortPropPageProvider"
```

```
[DriverInstall.NT.Services]
include=mdmcpq.inf
AddService=usbser, 0x00000002, LowerFilter_Service_Inst

;------------------------------------------------------------------------------
;  Windows 64bit OSes Section
;------------------------------------------------------------------------------
[DriverInstall.NTamd64]
include=mdmcpq.inf
CopyFiles=FakeModemCopyFileSection
AddReg=DriverInstall.NTamd64.AddReg

[DriverInstall.NTamd64.AddReg]
HKR,,DevLoader,,*ntkern
HKR,,NTMPDriver,,%DRIVERFILENAME%.sys
HKR,,EnumPropPages32,,"MsPorts.dll,SerialPortPropPageProvider"

[DriverInstall.NTamd64.Services]
include=mdmcpq.inf
AddService=usbser, 0x00000002, LowerFilter_Service_Inst

;------------------------------------------------------------------------------
;  Common Sections
;------------------------------------------------------------------------------
[DestinationDirs]
DefaultDestDir=12

[SourceDisksNames]
[SourceDisksFiles]
[FakeModemCopyFileSection]

[LowerFilter_Service_Inst]
DisplayName= %SERVICE%
ServiceType= 1
StartType  = 3
ErrorControl = 0
ServiceBinary = %12%\usbser.sys

;------------------------------------------------------------------------------
;  String Definitions
;------------------------------------------------------------------------------
; These strings can be modified to customize your device
;------------------------------------------------------------------------------
[Strings]
MFGFILENAME="awusbcdc"
DRIVERFILENAME ="usbser"
MFGNAME="AeroflexWeinschel"
DESCRIPTION="Weinschel USB COM Port"
SERVICE="USB Serial Emulation Driver"
```

6.2. Updating the 4284A Firmware using USB HID Bootloader

The USB HID Bootloader is a PC application that communicates with the onboard HID Bootloader of the 4284A and allows updating the application program firmware of the A.

In order to use this program, you will need to have the .NET framework version 4 installed on your computer. If you do not have.NET framework 4.0 installed, a non-descript error message will occur when trying to launch the executable, and the program will not open. You may also need to install the Microsoft Visual C++ 2010 Redistributable Package.

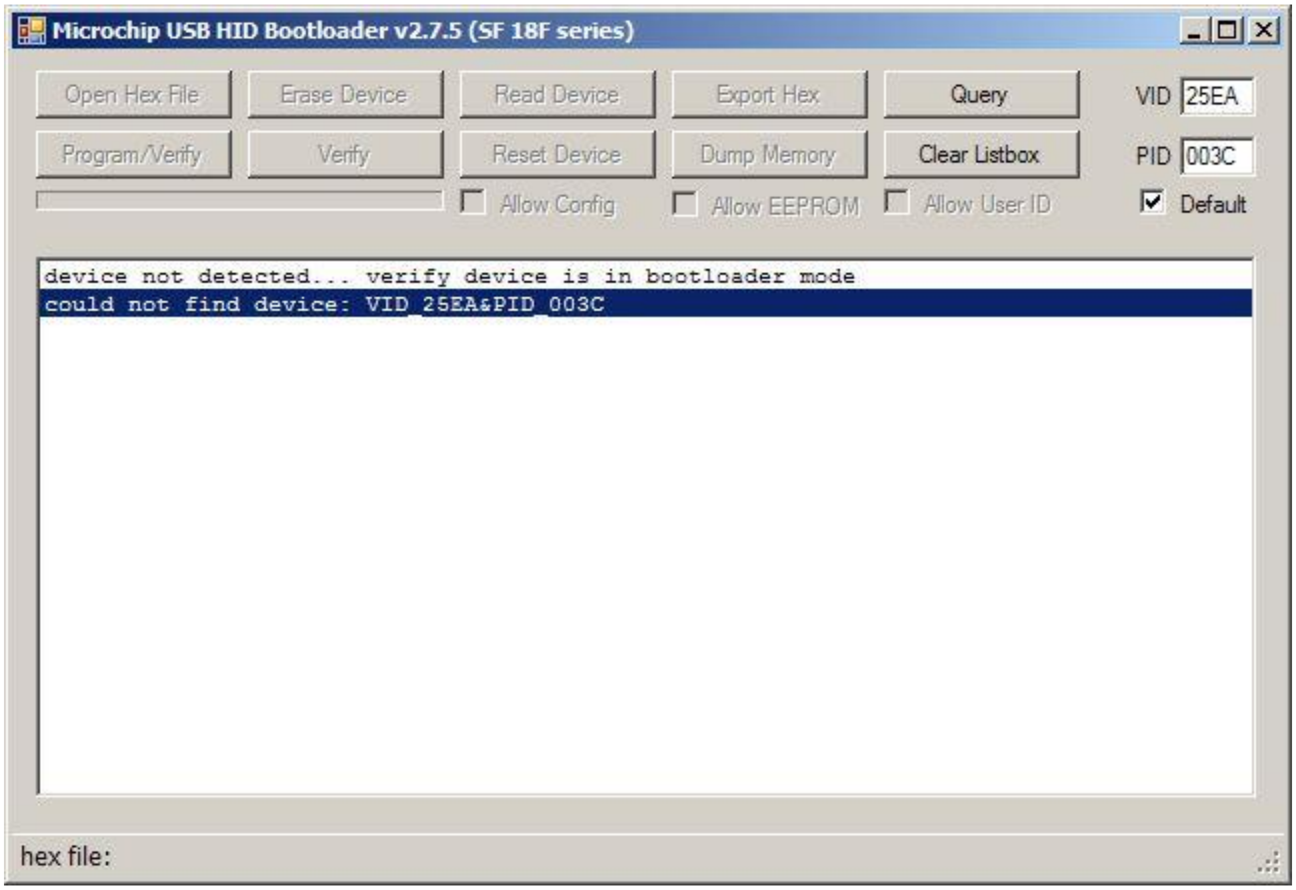
If you do not yet have them installed, they can be freely downloaded from the Microsoft website.

Microsoft Visual C++ 2010 Redistributable Package (x86)
http://www.microsoft.com/en-us/download/details.aspx?id=5555

Microsoft Visual C++ 2010 Redistributable Package (x64)
http://www.microsoft.com/en-us/download/details.aspx?id=14632

.NET V4 framework
http://www.microsoft.com/en-us/download/details.aspx?id=24872

Once installed, run the HIDBootLoader.exe program (or HIDBootLoaderx64.exe for 64-bit environments).
You should see the following window:



VID 25EA
PID 003C
[✓] Default

```
device not detected... verify device is in bootloader mode
could not find device: VID_25EA&PID_003C
```

hex file:

You can use one of two methods to set the 4284A into HID bootloader mode:

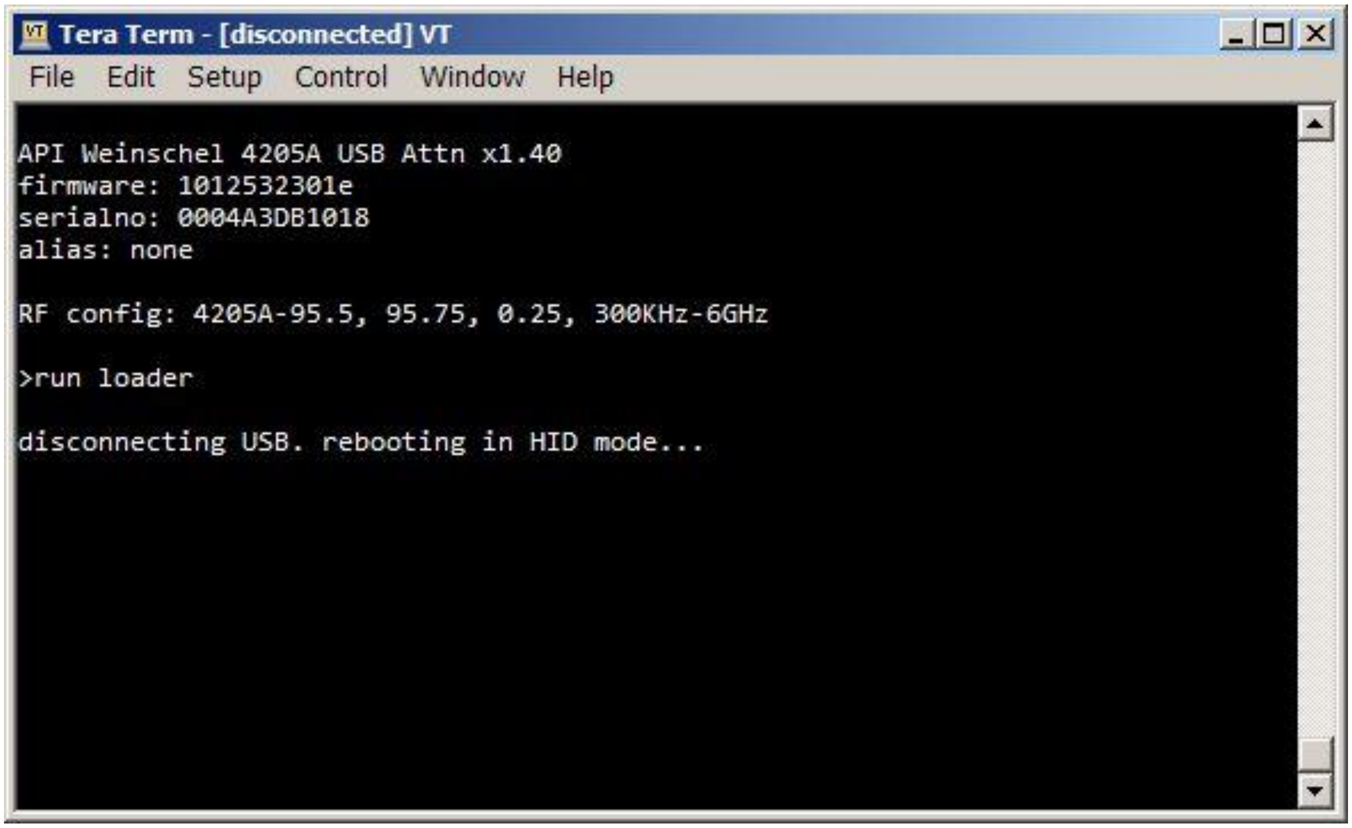
6.2.1 Method 1

Connect a shorting jumper between the 10-pin TTL header connector pins 8 and 10. When you connect the 4284A to the PC USB port it will power up in bootloader mode and the program should detect the device and read the chip configuration setup.

NOTE: be sure to remove the jumper after programming.

6.2.2 Method 2

From CDC serial mode, send the command RUN LOADER. The 4284A will exit CDC mode, disconnect itself and then reboot into HID bootloader mode.


Tera Term - [disconnected] VT
File Edit Setup Control Window Help

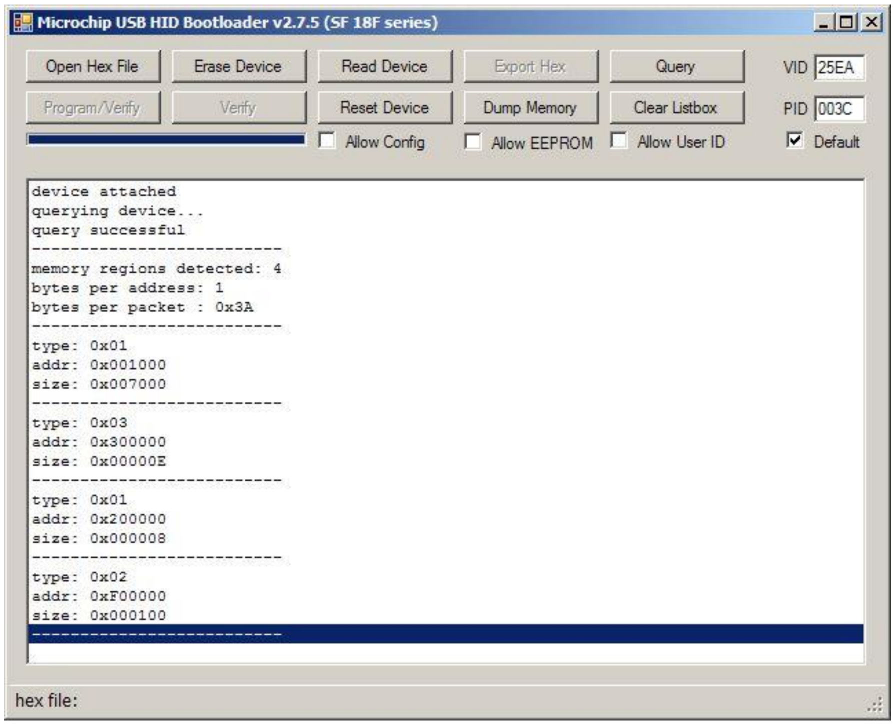
```
API Weinschel 4205A USB Attn x1.40
firmware: 1012532301e
serialno: 0004A3DB1018
alias: none

RF config: 4205A-95.5, 95.75, 0.25, 300KHz-6GHz

>run loader

disconnecting USB. rebooting in HID mode...
```

When the 4284A boots into HID loader mode the bootloader app should detect it, retrieve configuration info from the device, and display a screen similar to:

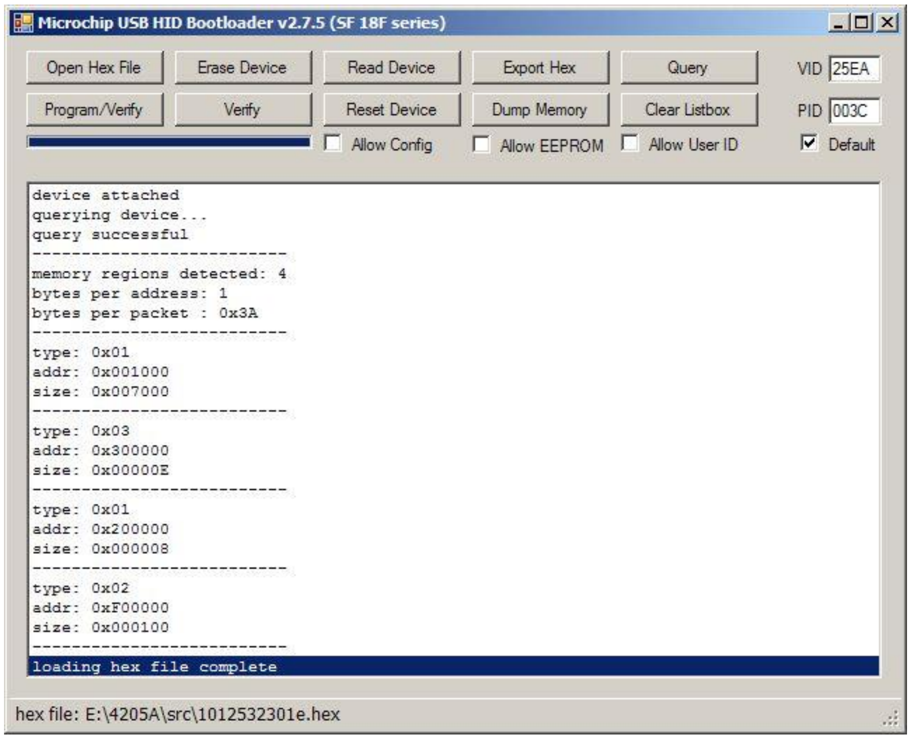


VID 25EA
PID 003C
☐ Allow Config ☐ Allow EEPROM ☐ Allow User ID ☑ Default

```
device attached
querying device...
query successful
--------------------------
memory regions detected: 4
bytes per address: 1
bytes per packet : 0x3A
--------------------------
type: 0x01
addr: 0x001000
size: 0x007000
--------------------------
type: 0x03
addr: 0x300000
size: 0x00000E
--------------------------
type: 0x01
addr: 0x200000
size: 0x000008
--------------------------
type: 0x02
addr: 0xF00000
size: 0x000100
--------------------------
```

hex file:

To download a new program, click 'Open Hex File' and navigate to the location with the HEX update file. Select the .HEX file and click 'Open'. This should load the HEX file and enable the 'Program/Verify' button.

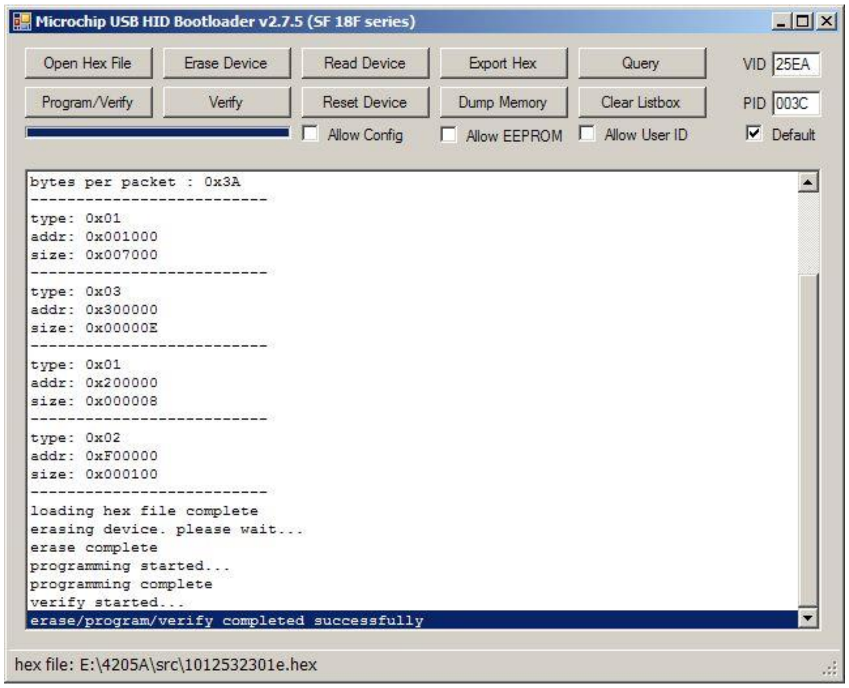


VID 25EA
PID 003C
☐ Allow Config ☐ Allow EEPROM ☐ Allow User ID ☑ Default

```
device attached
querying device...
query successful
--------------------------
memory regions detected: 4
bytes per address: 1
bytes per packet : 0x3A
--------------------------
type: 0x01
addr: 0x001000
size: 0x007000
--------------------------
type: 0x03
addr: 0x300000
size: 0x00000E
--------------------------
type: 0x01
addr: 0x200000
size: 0x000008
--------------------------
type: 0x02
addr: 0xF00000
size: 0x000100
--------------------------
loading hex file complete
```

hex file: E:\4205A\src\1012532301e.hex

Click the 'Program/Verify' button and programming should begin

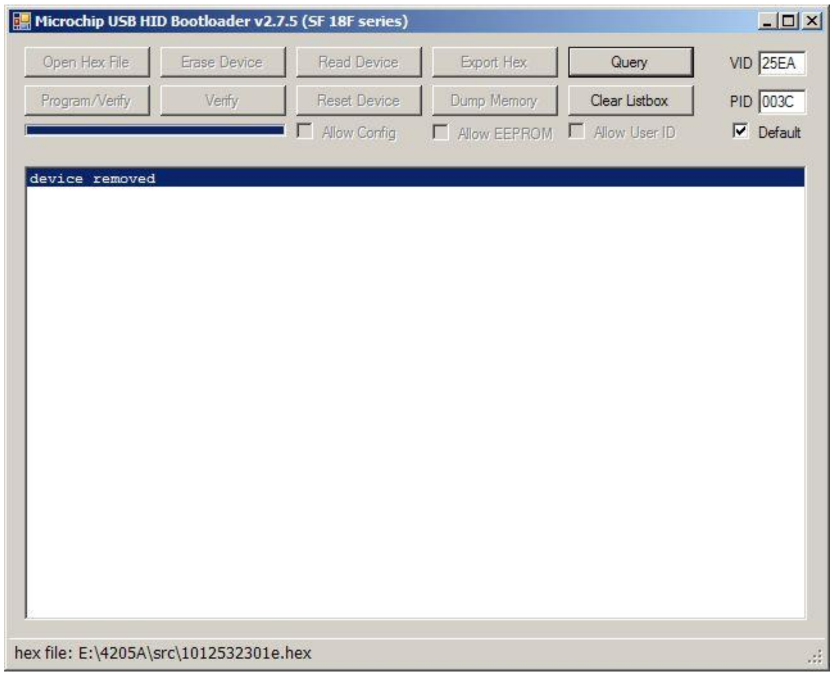


VID 25EA
PID 003C
Default [✓]

```
bytes per packet : 0x3A
---------------------------
type: 0x01
addr: 0x001000
size: 0x007000
---------------------------
type: 0x03
addr: 0x300000
size: 0x00000E
---------------------------
type: 0x01
addr: 0x200000
size: 0x000008
---------------------------
type: 0x02
addr: 0xF00000
size: 0x000100
---------------------------
loading hex file complete
erasing device. please wait...
erase complete
programming started...
programming complete
verify started...
erase/program/verify completed successfully
```

hex file: E:\4205A\src\1012532301e.hex

Once programming has completed you can select 'Reset Device' and the 4284A will exit HID mode, disconnect, and reboot into CDC mode.

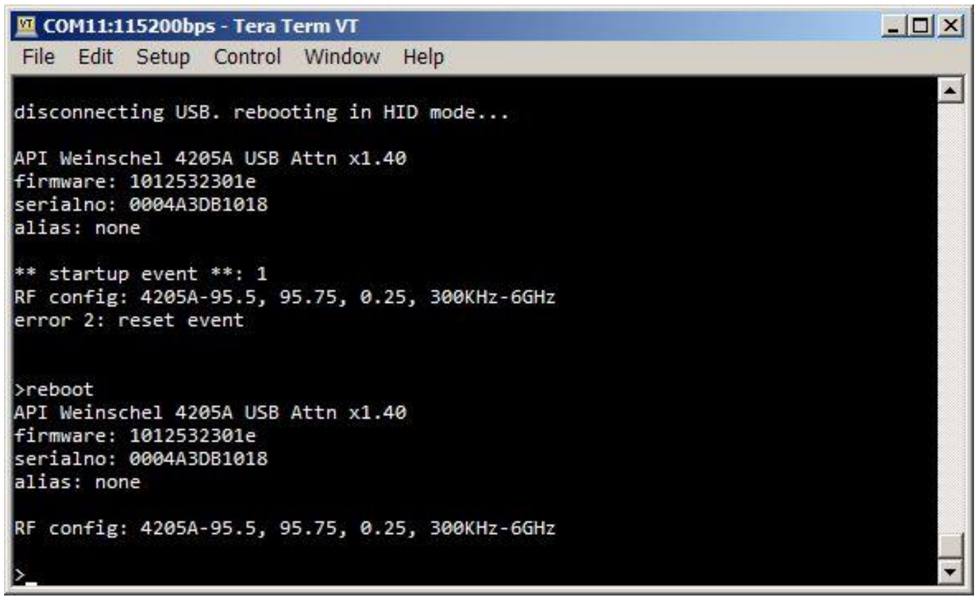


VID 25EA
PID 003C
Default [✓]

```
device removed
```

hex file: E:\4205A\src\1012532301e.hex

When you initially return to CDC mode you may see warning messages about the reset startup and nvm warning messages, depending upon the version update. After rebooting the unit (REBOOT or plug/unplug) these messages should disappear.


COM11:115200bps - Tera Term VT
File Edit Setup Control Window Help

```
disconnecting USB. rebooting in HID mode...

API Weinschel 4205A USB Attn x1.40
firmware: 1012532301e
serialno: 0004A3DB1018
alias: none

** startup event **: 1
RF config: 4205A-95.5, 95.75, 0.25, 300KHz-6GHz
error 2: reset event


>reboot
API Weinschel 4205A USB Attn x1.40
firmware: 1012532301e
serialno: 0004A3DB1018
alias: none

RF config: 4205A-95.5, 95.75, 0.25, 300KHz-6GHz

>_
```

7. Factory Service and Repairs

NOTICE

DO NOT return any instrument or component to Weinschel without receiving prior factory authorization.

Please contact the Weinschel Customer Service Department to discuss your product and resolve any issues that may be corrected without returning the product to the factory. If the issue cannot be corrected, you may be issued an RMA number and instructed to return the product. Additionally, you may be requested to submit additional information regarding the product failure to help verify your complaint.

When contacting customer service, please provide the following information:

1. Product Model Number
2. Product Serial Number
3. Date of Original Purchase
4. Company Name
5. Name
6. Phone Number

If a product has been approved to be returned to the factory, follow these instructions to ensure timely service.

1. If possible, use the original packing container and cushioning material. If the original materials are not available, use a strong shipping container and protect the product with shock absorbing material.
2. Shock absorbing material should be 3/4 inch thickness or greater and should protect all sides of the unit, as well as prevent movement.
3. Attach a tag to the product with the following information:
    - Model and serial numbers of all returned products
    - Service being requested
    - Description of malfunction
    - Return address
    - Authorization to conduct repairs
    - Return authorization number (RMA #)
4. Seal the packaging and mark it as FRAGILE.
5. Ship the product to the listed addressor or to an authorized sales representative. This information will be supplied by Weinschel.

8. Contacting Weinschel

Please use the general information below to contact Weinschel for any inquires.

Mail	Weinschel 5305 Spectrum Drive Frederick, MD 21703-7362 U.S.A.
Fax	1-301-846-9116
Phone	Toll Free: 1-800-638-2048 Toll call: 1-301-846-9222
Website	http://weinschel.apitech.com/
E-mail	weinschel-sales@apitech.com

8.1. Manufacturer Warranty

PRODUCTS - Weinschel, a part of API Technologies Corp., warrants each product it manufactures to be free from defects in material and workmanship under normal use and service anywhere in the world. Weinschel's only obligation under this Warranty is to repair or replace, at its plant, any product or part thereof that is returned with transportation charges prepaid to Weinschel by the original purchaser within TWO YEARS from the date of shipment.

The foregoing Warranty does not apply Weinschel's sole opinion to products that have been subject to improper or inadequate maintenance, unauthorized modifications, misuse, or operation outside the environmental specifications for the product.

SOFTWARE PRODUCTS - Weinschel software products are supplied without representation or Warranty of any kind. Weinschel, therefore, assume no responsibility and will not accept liability (consequential or otherwise) arising from the use of program materials, disk, or tape.

The Warranty period is controlled by the Warranty document furnished with each product and begins on the date of shipment. All Warranty returns must be authorized by Weinschel prior to their return.
Weinschel's Quality System Certified to:

INTERTEK ISO 9001:2008

ANAB ACCREDITED

Certificate No.: 94-289K

9. Revision History

Revision	Date	Description of Changes
X2	7/11/19	ERN xx-xxx: Initial Release