USB2-F-7x01 Full-Speed USB to 1-Port CANBus Adapter

Datasheet

Document Reference No.: CP_000050
Version 1.4
Issue Date: 2019-03-26

The USB2-F-7001 and USB2-F-7101 provide a simple method of adapting CANbus devices to USB. The USB2-F-7101adds optical isolation for the CAN interface.

Flexible mounting options allow the USB2-F-7001 and USB2-F-7101 to be used in a variety of applications. Indicator LEDs provide functional status.

Connective Peripherals Pte Ltd
178 Paya Lebar Road, #07-03 Singapore 409030
Tel.: +65 67430980   Fax: +65 68416071

E-Mail (Support): support@connectiveperipherals.com Web: www.connectiveperipherals.com/products

Neither the whole nor any part of the information contained in, or the product described in this manual, may be adapted or reproduced in any material or electronic form without the prior written consent of the copyright holder. This product and its documentation are supplied on an as-is basis and no warranty as to their suitability for any particular purpose is either made or implied. Connective Peripherals Pte Ltd will not accept any claim for damages howsoever arising as a result of use or failure of this product. Your statutory rights are not affected. This product or any variant of it is not intended for use in any medical appliance, device or system in which the failure of the product might reasonably be expected to result in personal injury. This document provides preliminary information that may be subject to change without notice. No freedom to use patents or other intellectual property rights is implied by the publication of this document. Connective Peripherals Pte Ltd, 178 Paya Lebar Road, #07-03 Singapore 409030. Registered Number: 201617872E

Table of Contents

1 Introduction

1.1 Functional Description

The USB2-F-7001 and USB2-F-7101 are USB to CANbus adapters which incorporate the FTDI FT245R USB to FIFO interface IC device. The USB2-F-7101adds optical isolation for the CAN interface. These adapters provide a fast, simple way to CANbus devices to a host PC with a USB port. Throughout this manual, both adapters will be identified as USB2-F-7x01 where common features are identified.

Each USB2-F-7x01 contains a small internal electronic circuit board mounted inside of a plastic case capable of withstanding industrial temperature ranges. The FT245R datasheet, DS_FT245R, is available at http://www.ftdichip.com. The integrated electronics include a Microchip PIC with the ECAN controller. LEDs give a visual indication of the CANbus status.

The USB2-F-7x01 does not require a power supply. It is powered from the USB connection. The USB2-F-7001 requires 150mA (450mA for the USB2-F-7101) and must be connected directly to a USB Host port, or a self-powered USB hub (i.e. with an external power supply). Bus-powered USB hubs are not supported since they cannot provide the required current.

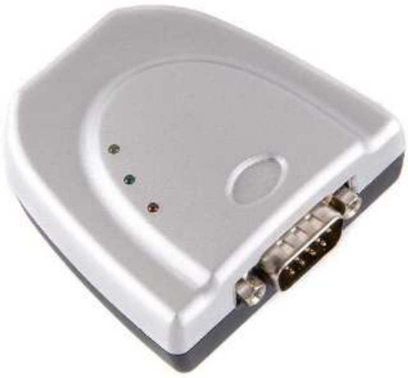


Figure 1.1 –USB2-F-7x01

Flexible mounting allows the USB2-F-7x01 to be used in a variety of applications, from a portable adapter to accompany a laptop to permanent installations in industrial and commercial locations.

The enclosure incorporates a standard USB-B device connector for connection to an upstream host or hub port. CANbus signals are available on an industry-standard DE-9P connector conforming to the CAN-in-Automation (CiA) DS102-2 pin-out. The maximum CANbus data rate is 1Mbps.

The USB2-F-7x01 adapter requires USB device drivers, available free from https://www.connectiveperipherals.com, which are used to make the USB2-F-7x01 appear as a Virtual COM port (VCP). Communications in VCP mode are accomplished with ASCII commands to the CAN controller. ASCII commands are listed in Section 4.1.

Another USB device driver included with the download, the D2XX driver, can also be used with application software to directly access the FT245R, and used in conjunction with a CAN command Application Programming Interface (API) through the use of DLLs. The high-level API is listed in the USB2-F-7x01 API Guide (https://www.connectiveperipherals.com/products/canbus-solutions/usb2-f-7101.html).

1.2 LED Description

The USB2-F-7x01 uses three LEDs to indicate a valid link as well as CANbus status. The table below uses the following LED definitions.

- ON = LED constantly lit
- OFF = LED is constantly not lit
- Blinking = 2.5Hz, alternating ON and OFF
- Flicker = 10Hz, alternating ON and OFF
- Single Flash = Short flash ON (200msec), followed by long off phase (1000msec)
- Double Flash = Two short flashes on (200msec) separated by short off phases (200msec), followed by long off phase (1000msec)
- Triple Flash = Three short flashes on (200msec) separated by short off phases (200msec), followed by long off phase (1000msec)

LED Colour	Function	Description		
Yellow	USB Enumerated	***Active LED***	***State***	***Description***
		Off	Inactive	The USB2-F-7x01 has not been enumerated by the host system.
		On	Active	The USB2-F-7x01 has been enumerated and is ready
Green	RUN	***CAN Run LED***	***State***	***Description***
		Blinking	PREOPERATIONAL	The device initialized
		Single flash	STOPPED	The device is in state STOPPED (Channel is Closed)
		On	OPERATIONAL	The device is in state OPERATIONAL (Channel is Open)
		Flicker	LISTEN	The device is in state OPERATIONAL (Channel is Open) No frames will be transmitted Note: 3 device minimum requirement
Red	ERR	***ERR LED State***	***Description***	***Category***
		Off	No error	The device is in working condition
		Blinking	Invalid Configuration	General configuration error
		Single flash	Warning limit reached	At least one of the error counters of the CAN controller has reached or exceeded the warning level (too many error frames)
		On	Bus off	The CAN controller is bus off, and not involved in CANbus activities

Table 1.1 – LED Description

Upon initial power up, system reboot or executing the (R)eset command, the LEDs will be in the following state: Yellow LED = ON, Green LED = Single Flash, Red LED = Off.

1.3 Block Diagram

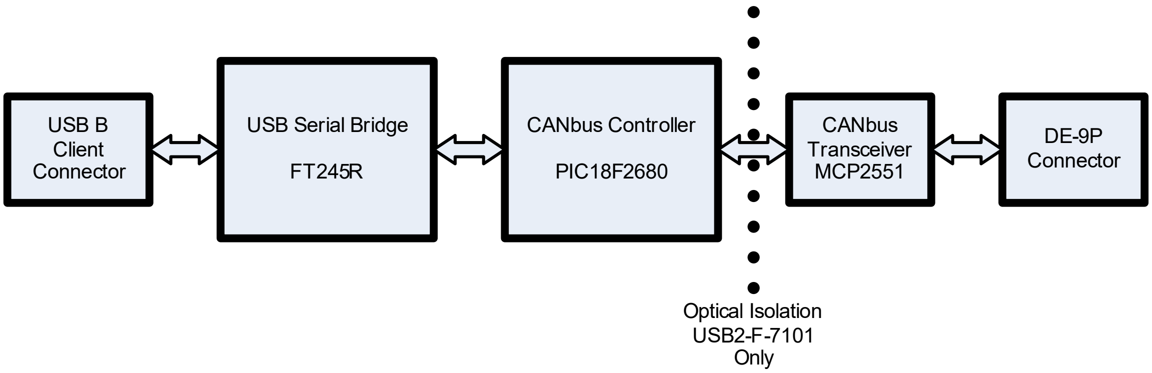


Figure 1.2 – USB2-F-7x01 Block Diagram

1.3.1 Block description

USB B Client Connector

This connector provides the interface for connection to a USB Host or Hub port. A standard "A to B" cable is provided, though one of a different length may also be used. The maximum cable length is 5 meters, according to the USB 2.0 specification.

FTDI FT245R

The FTDI FT245R provides a USB-to-FIFO interface to the CANbus Controller. Operating system device drivers are required in order to work with the FT245R to provide the Direct or Virtual COM Port functionality.

CANbus Controller

The Microchip PIC18F2680 runs firmware that converts the communication from the FT245R to CANbus protocol.

Optical Isolation (USB2-F-7101 only)

The USB2-F-7101 provides 2000V optical isolation between the CAN transceiver and the PIC18F2680.

CANbus Transceiver

The Microchip MCP2551 converts logic level signals from the CANbus controller to CANbus physical signals. These signals consist of a differential pair, CAN_H and CAN_L.

DE-9P Connector (Male)

The DE-9P connector is configured in an industry standard (CiA DS102-2) pin-out to provide connection to CANbus peripherals through standard cables. See section 3.1.2.

1.4 Features

- Adds one CANbus port by connecting to USB
- Easy plug & play installation and CANbus device connection
- Works with USB 1.1 & 2.0 Host and Hub ports
- Industry Standard FTDI chip set & drivers for maximum compatibility
- Installs as standard Windows COM port or USB Serial Device
- COM Port Number can be changed to COM1 to COM4 or any other COM port number required
- Supports Windows Server 2008, 2003, Vista, XP 2000, Linux, Mac OS X
- FIFO: 128 byte transmit buffer, 384 byte receive buffer
- Powered by USB port. No external power adapter required.
- CANbus speed up to 1Mbps
- One DE-9P male connector
- LEDs indicate Initialization and CANbus status for monitoring port status & easy diagnostics
- High-temperature plastic enclosure
- Operating temperature of -40°C to +85°C
- USB cable of 0.9 meter included

1.5 Performance Figures

Parameter	Performance
USB Interface	12Mbps USB 2.0 Full-Speed
CANbus Interface	10Kbps to 1Mbps

Table 1.2 – Performance Figures

1.6 Ordering Information

Part Number	Description
USB2-F-7001	CAN-Plus Full-Speed USB to 1-Port CANbus Adapter
USB2-F-7101	CAN-Plus Full-Speed USB to 1-Port CANbus Adapter with optical isolation

Table 1.3 – Ordering Information

2 Installation

2.1 Hardware Installation

There are no switches or jumpers to configure on the USB2-F-7x01.

2.1.1 Mounting

The USB2-F-7x01 is provided with two mounting options: UniClip™ Wall/DIN rail mount or rubber feet. The UniClip Wall/DIN rail mount allows the USB2-F-7x01 to be permanently mounted to a wall or attached to a DIN rail. The rubber feet can be used when mobility or desktop use is desired.

NOTE: The UniClip provides a permanent mounting style. The USB2-F-7x01 case may be damaged if the UniClip is removed.

The following figures indicate the mounting styles.

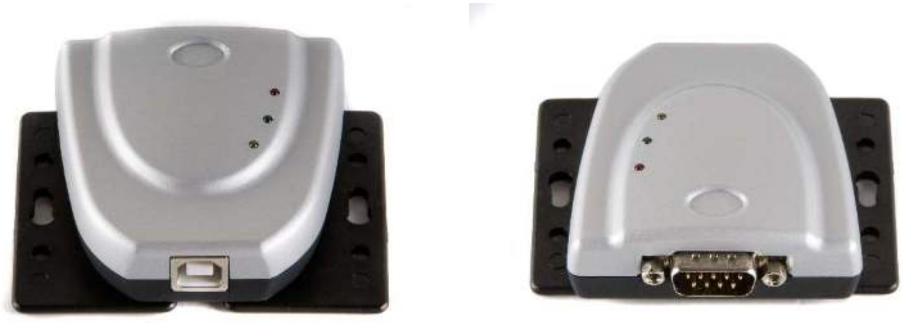


Figure 2.1 – USB2-F-7x01 with UniClip Brackets

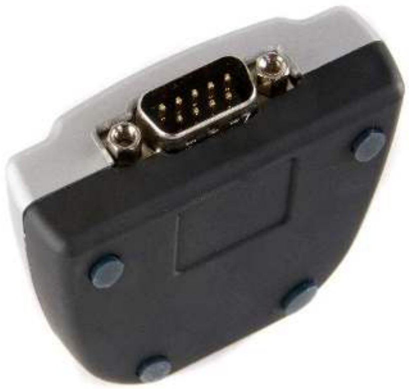


Figure 2.2 – USB2-F-7x01 with Rubber Feet

2.1.2 Wiring

A standard 0.9m USB "A" to "B" cable is provided.

Insert the A-plug into an available USB Host or Hub port. Insert the B-plug into the B-receptacle on the USB2-F-7x01.

The CANbus interface is provided through a DE-9P connector as defined by the CiA DS102-2 pin-out. Bus termination is *not* provided. A CANbus network requires 120Ω termination resistors at each end as noted in Figure 2.3. Wiring to CANbus nodes through the middle of the network must ensure a short tap length.

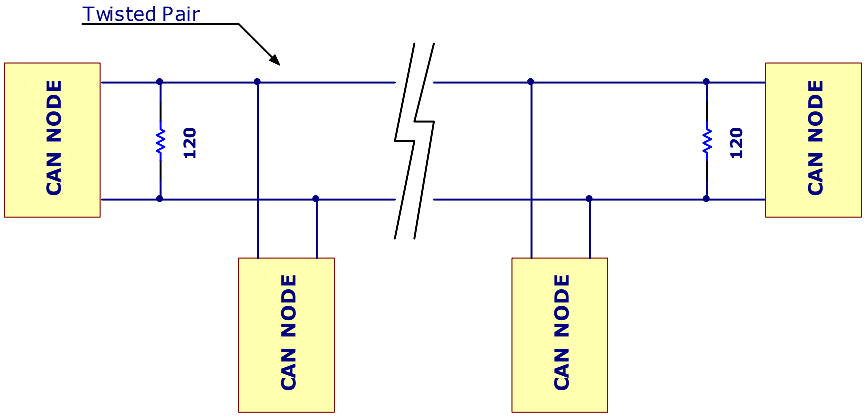


Figure 2.3 – CANbus network

2.2 Device Driver Installation

The USB2-F-7x01 adapter requires USB device drivers, available free from https://www.connectiveperipherals.com, which are used to make the USB2-F-7x01 appear as a Virtual COM port (VCP). Communications in VCP mode are accomplished with ASCII commands to the CAN controller.

Another USB device driver included with the download, the D2XX driver, can also be used with application software to directly access the FT245R, and used in conjunction with a CAN command Application Programming Interface (API) through the use of DLLs. The API is defined in the USB2-F-7x01 API Guide.

The USB2-F-7x01 uses the USB Vendor ID of 0x0403 and Product ID of 0xFAC6.

2.2.1 Microsoft Windows

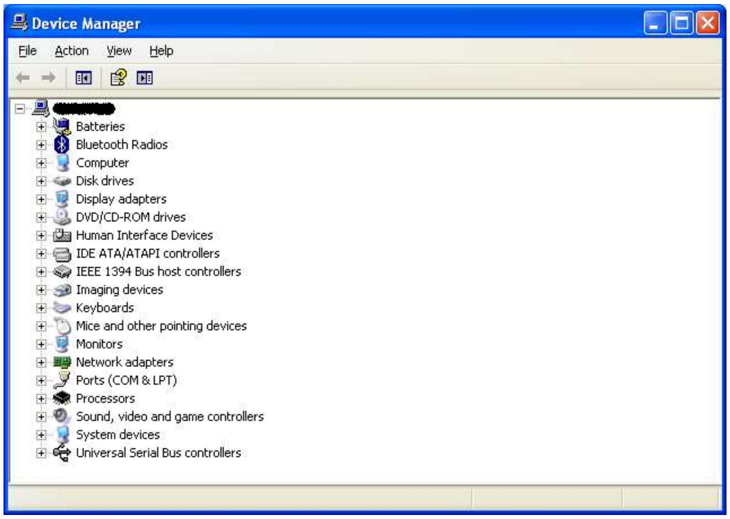
Installing Drivers

1) Login to your system as Administrator, or a user with Administrator rights.
2) The device drivers are WHQL-certified and available through the Windows Update service.
3) If you are unable to use the Windows Update service, download the Connective Peripherals drivers from https://www.connectiveperipherals.com. Extract the files to a folder on your local hard drive.
4) Plug in the USB2-F-7x01. When the Hardware Wizard appears, select to browse your computer and point the Wizard to the extracted folder.
5) Follow the on-screen instructions to complete the driver install.

Windows application software can access the USB2-F-7x01 through a COM port or D2XX direct function calls. For software which uses a COM port, the assignment of the COM port number determined as noted below.

COM Port Assignment

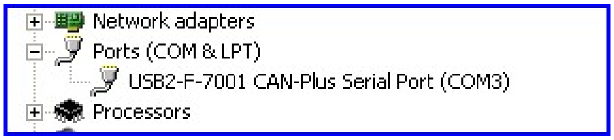
To determine which COM port has been assigned, open the Windows Device Manager from the System Control Panel.



[redacted]
- Batteries
- Bluetooth Radios
- Computer
- Disk drives
- Display adapters
- DVD/CD-ROM drives
- Human Interface Devices
- IDE ATA/ATAPI controllers
- IEEE 1394 Bus host controllers
- Imaging devices
- Keyboards
- Mice and other pointing devices
- Monitors
- Network adapters
- Ports (COM & LPT)
- Processors
- Sound, video and game controllers
- System devices
- Universal Serial Bus controllers

Figure 2.4 – Device Manager

Click on the Plus "+" sign next to the Ports tree to list the available COM port. You will see "Connective Peripherals USB COM Port", followed by a COMn assignment. In the figure below, the USB2-F-7x01 is assigned to COM3.



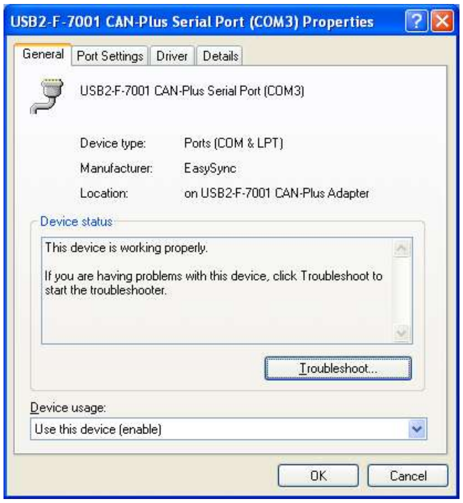
- Network adapters
- Ports (COM & LPT)
  - USB2-F-7001 CAN-Plus Serial Port (COM3)
- Processors

Figure 2.5 – COM Port Assignment

Use this COM port number in order to access the USB2-F-7x01 with application software that uses a COM port.

If an application requires use of a different COM port number, the assignment may be changed through the Advanced Driver Options settings.

From the Device Manager listing above, right-click on the USB Serial Port and select Properties.



USB2-F-7001 CAN-Plus Serial Port (COM3) Properties

USB2-F-7001 CAN-Plus Serial Port (COM3)

Device type: Ports (COM & LPT)

Manufacturer: EasySync

Location: on USB2-F-7001 CAN-Plus Adapter

Device status

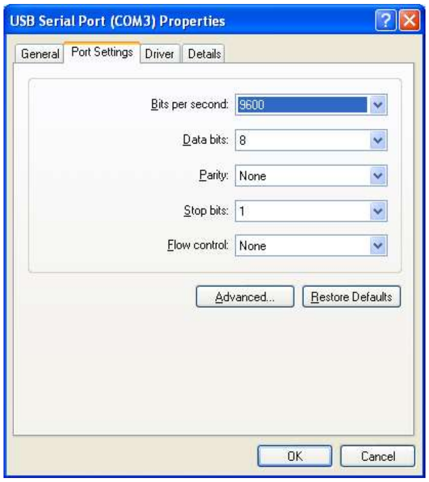
This device is working properly.

If you are having problems with this device, click Troubleshoot to start the troubleshooter.

Device usage: Use this device (enable)

Figure 2.6 – Access COM Port Properties

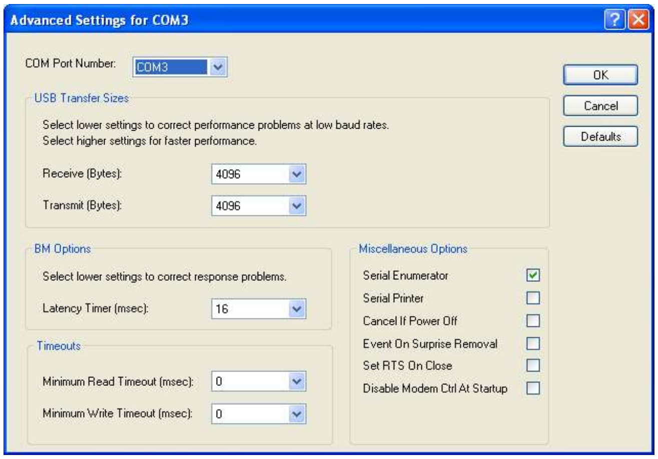
Next, click on the “Port Settings” tab.



Bits per second: 9600
Data bits: 8
Parity: None
Stop bits: 1
Flow control: None

Figure 2.7 – Settings Tab

Then click on the “Advanced…” button.



COM Port Number: COM3

USB Transfer Sizes

Select lower settings to correct performance problems at low baud rates.
Select higher settings for faster performance.

Receive (Bytes): 4096
Transmit (Bytes): 4096

BM Options

Select lower settings to correct response problems.

Latency Timer (msec): 16

Timeouts

Minimum Read Timeout (msec): 0
Minimum Write Timeout (msec): 0

Miscellaneous Options

Serial Enumerator ☑
Serial Printer □
Cancel If Power Off □
Event On Surprise Removal □
Set RTS On Close □
Disable Modem Ctrl At Startup □

Figure 2.8 – Advanced Options

This will display the various advanced settings. Note the COM port assignment in the upper left. Clicking on the drop-down list will display the available port numbers. Select one that is not in use and click OK on each dialog box to activate the selection. A unique USB serial number is programmed into the USB2-F-7x01. Windows will use this USB serial number to remember the assigned COM port number across power cycles and if the USB2-F-7x01 is removed and inserted to any USB port on the same host computer. This USB serial number is different from the CAN Firmware serial number described later in this datasheet.

2.2.2 Mac OS X, Linux, Windows CE

Device drivers and Installation guides for Mac OS X, Linux and Windows CE are available for download on the Connective Peripherals web site.

2.3 Firmware Update

The USB2-F-7x01 firmware can be updated for bug fixes and enhanced features. In order to update the firmware, obtain the updated firmware from https://www.connectiveperipherals.com and follow the instructions contained in the readme file of the download.

3 Connections

3.1 External Connectors

3.1.1 USB

The USB-F-7x01 is a downstream USB 2.0 Device. A standard USB Series "B" receptacle is mounted inside the USB-F-7x01 to facilitate connection to an upstream USB Host or Hub.

Pin Number	Pin Type	Description
1	Power	VBUS – USB Power provided from upstream USB Host or Hub
2	Bidirectional	D– = USB data signal, negative polarity
3	Bidirectional	D+ = USB data signal, positive polarity
4	Ground	GND = USB signal ground
Shield	Case Ground	Drain = typically connected to the host PC case

Table 3.1 – USB "B" Receptacle Pin-Out

3.1.2 CANbus

The CANbus port conforms to the CiA DS102-2 pin-out through a 9-contact D-Sub Pin connector. Pin assignments are noted below:

Pin Number	Pin Type	Description
1	n/a	Not Connected
2	Bidirectional	CAN_L – CANbus low-side signal of differential pair
3	Ground	Signal Ground
4	n/a	Not Connected
5	n/a	Not Connected
6	Ground	Signal Ground
7	Bidirectional	CAN_H – CANbus high-side signal of differential pair
8	n/a	Not Connected
9	n/a	Not Connected
Shield	Case Ground	Drain = typically connected to the host PC case

Table 3.2 – DE-9P DS102-2 CANbus Pin-Out

4 Application Programming Interfaces

4.1 ASCII commands

The following ASCII commands provide a means of utilizing an interactive terminal program, or communication through a standard COM or TTY port. Unrecognized commands return the ASCII BELL character (0x07). Some useful settings for a terminal program to assist in the human readability of command responses are:

- Echo typed characters locally – the USB2-F-7x01 does not echo characters.
- Append line feeds to incoming line ends.
- With the use of the FTDI FT245 USB-FIFO interface, serial port bit rate settings are not typically necessary; however, some terminal programs may yield unpredictable results with slow rates. Set the COM port to the highest available bit rate. The USB2-F-7x01 will maintain the most efficient data rate available.

The CANbus messages are received at all times by default and sent to the host through the FTDI FT245 USB-FIFO interface immediately. Firmware version 2.2 supports sustained read of CAN messages sent 1ms apart as well as a bursts of up to 100 CAN messages sent 50us apart.

Help (H, h or ?)

Summary
Lists available commands.

Definition
H<CR>, h<CR> or ?<CR>

Parameters
None

Remarks
Example:
?<CR>
List available commands.
Return Codes
List of Supported Commands

O – Open CAN Channel
C – Close CAN Channel
t – Transmit Standard Frame
T – Transmit Extended Frame
F – Read Status Flag
V – USB2-F-7001 Version
N – Serial Number of USB2-F-7001
Z – TimeStamp Option On/Off
S – Set CAN Bit Rate
s – Set BRGCON Registers
m – Set Acceptance Mask
M – Set Acceptance Filter
B – Enter Bootload Mode
L – Set Listen Mode
E – Clear Buffers and Echo Char
R – Reset USB2-F-7001
H, ? or h – Help on USB2-F-7001 Commands<CR> = OK
<BELL> = ERROR

4.1.1 Set CAN Channel Timing – simple (S)

Summary

Configure CAN interface with a pre-configured channel transmission rate.

NOTE: The CAN channel will revert to its prior state after execution. For example if the channel is Open when this command is executed, the channel will update the setting and return to the Open state.

Definition

Srate_selector<CR>

Parameters

rate_selector = 0 = 10Kbps
1 = 20Kbps
2 = 50Kbps
3 = 100Kbps
4 = 125Kbps
5 = 250Kbps
6 = 500Kbps
7 = 800Kbps
8 = 1Mbps

Remarks

Example:
S5<CR>
Set CAN transmission rate to 250Kbps

Return Codes

<CR> = OK
<BELL> = ERROR

4.1.2 Set CAN Channel Timing – advanced (s)

Summary

Configure CAN interface with a custom channel transmission rate.

NOTE: The CAN channel will revert to its prior state after execution. For example if the channel is Open when this command is executed, the channel will update the setting and return to the Open state.

Definition

Saabbcc<CR>

Parameters

aa = contents of PIC 18F2680 BRGCON1 register (in hexadecimal)
bb = contents of PIC 18F2680 BRGCON2 register (in hexadecimal)
cc = contents of PIC 18F2680 BRGCON3 register (in hexadecimal)

Remarks

The USB2-F-7x01 utilizes a 24MHz clock for the PIC 18F2680. Use this value when referring to the PIC datasheet if a custom transmission rate different from those provided with the "S" command above is required.

Example:

```
s01BE07<CR>
```

Set CAN transmission rate to 250Kbps.
NOTE: With this command, multiple combinations of BRGCON1, BRGCON2 and BRGCON3 can yield the same bit rate.

Return Codes

<CR> = OK
<BELL> = ERROR

4.1.3 Set Acceptance Mask (m)

Summary

The Acceptance Mask, in conjunction with the Acceptance Filter (M), defines which received messages (i.e. of a specific ID or range of CAN IDs) will be passed to the USB interface. The Acceptance Mask value corresponds to bits within a range of valid CAN IDs for either standard or extended CAN messages.

Set Acceptance Mask (m) command should be executed *prior* to Set Acceptance Filter (M).

NOTE: The CAN channel will revert to its prior state after execution. For example if the channel is Open when this command is executed, the channel will update the setting and return to the Open state.

Definition

miii<CR> for standard 11-bit CAN messages
miiiiiiii<CR> for extended 29-bit CAN messages

Parameters

iii = standard 11-bit CAN mask (0x000 through 0x7FF)
iiiiiiii = extended 29-bit CAN mask (0x00000000 through 0x1FFFFFFF)
A value of "0" in a bit location indicates that the bit location ID value is to be *ignored* when filtering messages.
Default is to pass all frames (Acceptance Mask = 0x000 for standard messages and 0x00000000 for extended messages)

Remarks

Example

```
m700<CR>
```

Set Acceptance Mask to check bits 10, 9 and 8 against the filter. Bits 7 thorugh 0 are ignored as "don't care". Use the Acceptance Mask in conjunction with the Acceptance Filter, defined next.

Return Codes

<CR> = OK

<BELL> = ERROR

4.1.4 Set Acceptance Filter (M)

Summary

The Acceptance Filter, in conjunction with the Acceptance Mask (m), defines which received messages (i.e. of a specific ID or range of CAN IDs) will be passed to the USB interface. The Acceptance Filter value corresponds to a valid CAN ID for either standard or extended CAN messages.

The Set Acceptance Mask (m) command should be executed *prior* to the Set Acceptance Filter (M) command.

NOTE: The CAN channel will revert to its prior state after execution. For example if the channel is Open when this command is executed, the channel will update the setting and return to the Open state.

Definition

Miii<CR> for standard 11-bit CAN messages
Miiiiiiii<CR> for extended 29-bit CAN messages

Parameters

iii = standard 11-bit CAN ID (0x000 through 0x7FF)
iiiiiiii = extended 29-bit CAN ID (0x00000000 through 0x1FFFFFFF)
Default is to pass all frames (Acceptance Filter = 0x7FF for standard messages and 0x1FFFFFFF for extended messages).

Remarks

Example

```
M1FF<CR>
```

Set Acceptance Filter to receive standard messages with the CAN ID of 0x1FF. If used in conjunction with the Acceptance Mask example above, frames of the range 0x100 through 0x1FF will be passed and all other CAN IDs blocked.

Return Codes

<CR> = OK
<BELL> = ERROR

4.1.5 Enable Time Stamp (Z)

Summary

Sets or clears time stamp on received frames. This value is persistent in EEPROM across reset or restart. Four (4) bytes are added to the end of a received frame. This value is a rolling 16-bit counter that increments once every millisecond and rolls over at 60,000mS (1 minute). Valid hex values are 0x0000 through 0xEA5F.

NOTE: The CAN channel will revert to its prior state after execution. For example if the channel is Open when this command is executed, the channel will update the setting and return to the Open state.

Definition

Zn<CR>

Parameters

n = 0 = disable time stamp feature

1 = enable time stamp feature

Remarks

Default is OFF (disable time stamp)

Only use when required to change functionality.

Example:
Z1<CR>

Enable time stamp.

Return Codes

<CR> = OK
<BELL> = ERROR

4.1.6 Open CAN Channel (O)

Summary

Opens CAN channel for read/write operations. CAN channel must be initiated with selected speed (S or s) prior to use of Open.

Definition

O<CR>

Parameters

None

Remarks

Example:
O<CR>

Open CAN channel in normal communication mode.

Once the CAN channel is open, it is necessary to execute the Echo/Synchronize (E or e) command to flush the data buffers. Once the CAN channel is synchronized, received data is automatically sent from the CAN network to the controlling application.

Return Codes

<CR> = OK
<BELL> = ERROR

4.1.7 Open CAN Channel for Listen Only (L)

Summary

Opens the CAN channel in Listen Only mode. This is essentially the same as Open, although attempts to use either Transmit command (T or t) will result in an error. Listen Only mode is useful for monitoring the CAN channel without interaction from the CAN bus. CAN channel must be initiated with selected speed (S or s) prior to use of Listen.

When in Listen mode, the channel must first be Closed (C) and then opened with Open (O) prior to attempting to transmit frames.

NOTE: A minimum of three nodes are required on the CANbus network for Listen mode to function. To monitor CAN traffic with only two devices, use the Open (O) command.

Definition

L<CR>

Parameters

None

Remarks

Example:
L<CR>

Open CAN channel in Listen Only mode
Received data is automatically sent from the CAN network to the controlling application.
The RUN (green) LED will flicker when this mode is active.

Return Codes

<CR> = OK
<BELL> = ERROR

4.1.8 Close CAN Channel (C)

Summary

Closes the CAN channel. This command provides backward compatibility with some existing CANbus adapters.

Definition

C<CR>

Parameters

None

Remarks

Example:
C<CR>

Close CAN channel.

Return Codes

<CR> = OK
<BELL> = ERROR

4.1.9 Echo / Synchronize (E)

Summary

Clear transmit data buffers. Data buffers should be cleared immediately after opening the CAN channel to prevent erroneous as a result of sending old data in the buffer.

Definition

E<CR>

Parameters

None

Remarks

Example:
    E<CR>

Clear data buffers.

Return Codes

E<CR> = OK
<BELL> = ERROR

4.1.10 Transmit Standard CAN Frame – 11-bit ID (t)

Summary

Transmits a standard CAN frame with an 11-bit ID. The CAN channel must be Open (O) prior to transmitting any messages.

Definition

tiiildd..dd<CR>

Parameters

iii = 3-digit identifier in hex (000 through 7FF)
l = Length (number of bytes) of data message
dd = Data bytes in hex (00 through FF).
The number of bytes must match message length

Remarks

Example:
    t34580123456789ABCDEF

Transmit a 11-bit ID frame with
ID = 0x345
Length = 8 bytes
Data = 0x01, 0x23, 0x45, 0x67, 0x89, 0xAB, 0xCD, 0xEF

Example:
    t4560

Transmit an 11-bit ID frame with
ID = 0x456

Length = 0 bytes
Data = zero bytes (no data)

Return Codes

z<CR> = OK
<BELL> = ERROR

4.1.11 Transmit Extended CAN Frame – 29-bit ID (T)

Summary

Transmits an extended CAN frame with a 29-bit ID. The CAN channel must be Open (O) prior to transmitting any messages.

Definition

Tiiiiiiiildd..dd<CR>

Parameters

iiiiiii = 8-digit identifier in hex (00000000 through 1FFFFFFF)
l = Length (number of bytes) of data message
dd = Data bytes in hex (00 through FF).
The number of bytes must match message length

Remarks

Example:
T1234567880123456789ABCDEF

Transmit a 29-bit ID frame with
ID = 0x12345678
Length = 8 bytes
Data = 0x01, 0x23, 0x45, 0x67, 0x89, 0xAB, 0xCD, 0xEF

Example:
T123456780

Transmit an 11-bit ID frame with
ID = 0x12345678

Length = 0 bytesData = zero bytes (no data)

Return Codes

Z<CR> = OK
<BELL> = ERROR

4.1.12 Get Status Flags (F)

Summary

Get CANbus and controller status. A two-byte BCD number is returned to correspond to the 8-bits of the COMSTAT register of the PIC18F2680.

Definition

F<CR>

Parameters

None

Remarks

Example
F<CR>
Get CANbus status

Return Codes

xx<CR> = OK
xx = CANbus status (A bit set to "1" indicates a true condition):

```
bit 0 = Flag bit 1 –or– Flag bit 2 is set
bit 1 = Receive Warning: 127 >= Receive Error Counter > 95
bit 2 = Transmit Warning:  Transmit Error Counter > 95
bit 3 = Receive Bus Passive:  Receive Error Counter > 127
        Will cause RED LED to Single Flash
bit 4 = Transmit Bus Passive:  Transmit Error Counter > 127
        Will cause RED LED to Single Flash
bit 5 = Transmit Bus-OFF:  Transmit Error Counter > 255
        Will cause RED LED to remain ON
bit 6 = Receive Buffer 1 Overflow
bit 7 = Receive Buffer 0 Overflow
```

<BELL> = ERROR

4.1.13 Get Hardware and Firmware Versions (V)

Summary

Get hardware and firmware version numbers of USB2-F-7x01. Each value consists of a two-digit, binary coded decimal (BCD) number.

Definition

V<CR>

Parameters

None

Remarks

Example
V<CR>
Get serial number

Return Codes

Vxxyy<CR> = OK
  xx = hardware version
  yy = firmware version
<BELL> = ERROR

4.1.14 Get Serial Number (N)

Summary

Get serial number of USB2-F-7x01.

Definition

N<CR>

Parameters

None

Remarks

Example
  N<CR>
Get serial number

Return Codes

Nxxxx<CR> = OK
  xxxx = serial number of the USB2-F-7x01. It is possible to have alphanumeric values.
<BELL> = ERROR

4.1.15 Reset Microcontroller (R)

Summary

Resets PIC18F2680 MCU. Configurations are preserved in EEPROM. This command is useful if the USB2-F-7x01 becomes unresponsive. LEDs will be in the state mentioned in Section 1.2.

Definition

R<CR>

Parameters

None

Remarks

Example
  R<CR>
Reset PIC18F2680

Return Codes

<CR> = OK.
<BELL> = ERROR

4.1.16 Prepare Bootloader (B)

Summary

Resets PIC18F2680 MCU into Bootloader mode. Only use this command immediately prior to loading new firmware onto the USB2-F-7x01.

Definition

B<CR>

Parameters

None

Remarks

Example
    B<CR>
Prepare to load new firmware

NOTE: The firmware programming utility must be run in order to recover from the Bootloader Mode.

See https://www.connectiveperipherals.com for the latest version of the utility.

Return Codes

Entering Bootloader Mode…
Boot:>

<BELL> = ERROR

5 Electrical details

5.1 USB

Parameter	Description	Minimum	Typical	Maximum	Units	Conditions
USB_VCC	Input Power Voltage	4.25	5.0	5.25	V	Present when USB cable is attached and USB Host or Hub powered.
I_{cc}	USB2-F-7001 USB current USB2-F-7101 USB Current			150 450	mA mA	USB Host or Self-Powered Hub ports only. Bus-powered hubs are not supported.

Table 5.1 – USB Electrical Details

5.2 CANbus

Parameter	Description	Minimum	Typical	Maximum	Units	Conditions
V_{trans}	Transmitter output voltage swing	0.5		4.5	V	See Microchip MCP2551 datasheet for complete details
V_{rec}	Receiver input voltage range	-0.3		5.3	V	See Microchip MCP2551 datasheet for complete details
	ESD HBM		± 6		KV	See Microchip MCP2551 datasheet for complete details

Table 5.2 – CANbus Electrical Details

5.3 Optical Isolation (USB2-F-7101 only)

Parameter	Description	Minimum	Typical	Maximum	Units	Conditions
V_{iso}	Dielectric Insulation Voltage			2,000	V	1 Minute Duration

Table 5.3 – Optical Isolation Electrical Details (USB2-F-7101 only)

6 Mechanical Details

6.1 Module Mechanical Dimensions

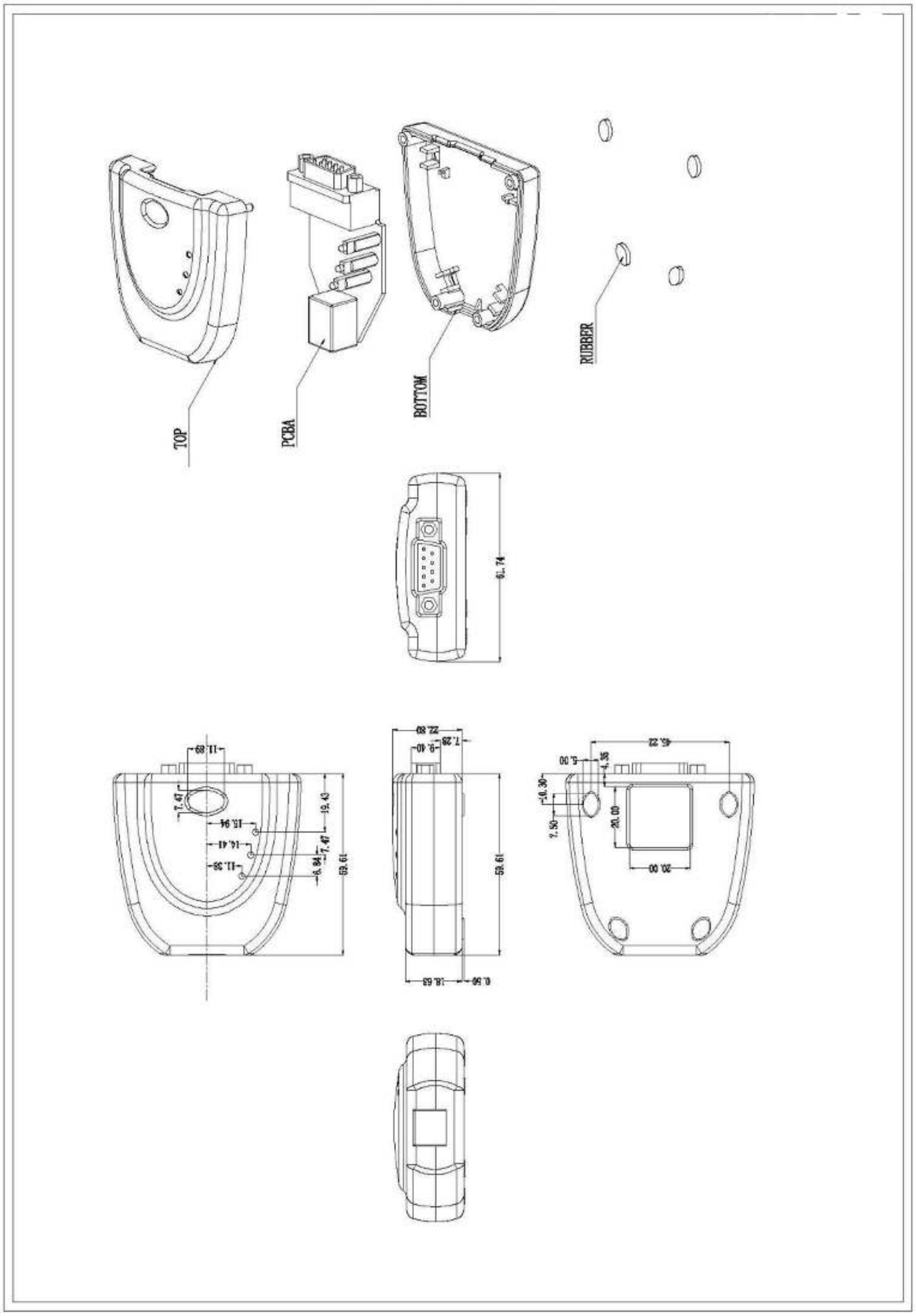


Figure 6.1 – USB2-F-7x01 Case Dimensions

6.2 UniClip™ Mechanical Dimensions

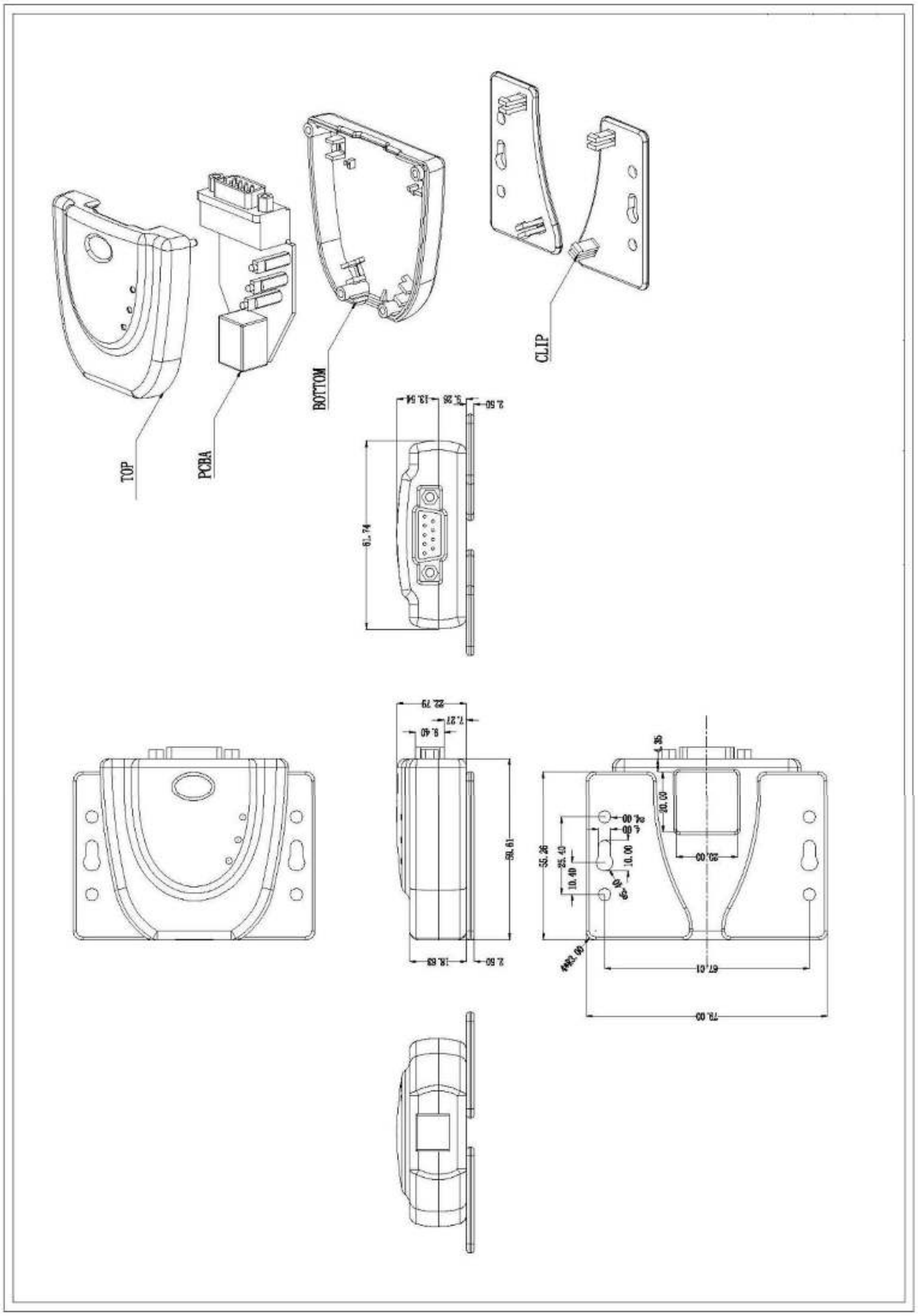


Figure 6.2 – USB2-F-7x01 Case Dimensions with UniClip

7 Physical Environment Details

7.1 Storage Temperature

Parameter	Description	Minimum	Typical	Maximum	Units	Conditions
T	Storage Temperature Range	TBD		TBD	ºC	

Table 7.1 – Storage Temperature

7.2 Operating Temperature

Parameter	Description	Minimum	Typical	Maximum	Units	Conditions
T	Operating Temperature Range	–40		+85	ºC	5% to 95% RH, non condensing

Table 7.2 – Operating Temperature

8 Environmental Approvals & Declarations

8.1 EMI Compatibility

FCC and CE

The USB2-F-7x01 has been tested to be compliant with both FCC Part 15 Subpart B and European EMC Directive.

NOTE: This is a Class B product. In a domestic environment, this product may cause radio interference, in which case the user may be required to take adequate measures.

NOTE: This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

8.2 Safety

The USB2-F-7x01 is defined as Limited Power Supply (LPS) device, with operating voltages under 60VDC.

8.3 Environmental

The USB2-F-7x01 is a lead-free device that complies with the following environmental directives: RoHS, WEEE, REACH, PFOS and DecaBDE.

8.4 Reliability

The USB2-F-7x01 is designed as a robust USB-Serial adapter for use in many environments. There are no user-serviceable parts. Any failure will require a replacement of the unit.

8.4.1 Mean Time To Failure (MTTF)

The MTTF is TBD.

8.5 Import / Export Information

Import / Export Information for USB2-F-7001 and USB2-F-7101	
Country of Origin	China
Harmonized Code	8471.80.1000
Product Description	USB to CANbus Computer Adapter, Single Port
USA ECCN	EAR99 – No License Required

Table 8.1 – Import / Export Information

9 Troubleshooting

9.1 Hardware

If the Yellow LED is not lit when plugged into the host our hub, check the following:

- USB cable is properly inserted at both ends
- Computer power is ON
- Computer is not in Sleep or Standby
- If a USB Hub is used, be sure it is set for "Self-Powered" operation
- If a USB Hub is used, be sure all cables are properly inserted.

If communication appears to be functioning to the USB2-F-7x01, but CAN communication is not functioning, the CANbus cables may need checked:

- Termination: The CANbus network requires 120ohm termination at the furthest points on the network. If additional termination resistors are present through the CANbus network or if one of the termination resistors is missing at one of the ends, the CANbus network will not communicate.

- Twisted Pair: The CANbus network consists of a differential pair of signals which greatly reduce noise on the signals. Failure to use twisted pair may cause erroneous communications, or the CANbus network to completely fail communications.

- Polarity: The CANbus network consists of a CAN_H and CAN_L signal. Care must be taken to follow the polarity. Always connect CAN_H to CAN_H, and CAN_L to CAN_L.

Ensure the CANbus network is only connected to the USB2-F-7x01 *after* connection to a USB Host or HUB port. See Section 1.2 for default LED power-up patterns. The CANbus network can remain connected through system reboots.

9.2 Device Driver

Ensure the latest device driver is in use. See https://www.connectiveperipherals.com. If other devices with FTDI chips are installed in the system, check with all manufacturers of these devices for the latest device drivers. See the Installation guides for additional details: http://ftdichip.com/Documents/InstallGuides.htm

Common Windows Device Driver Troubles:

- DEVICE TIMES OUT: The default settings of the device driver assume typical data transfers of hundreds to thousands or more bytes at a given time. Some applications, such as a GPS device, only send data in short packets, often only a few bytes. If this is the case, it may be necessary to adjust the driver buffer size and/or latency timer to smaller values. These values can be adjusted through the advanced driver options as noted in Figure 2.13. The buffer size can be reduced to 64 bytes. The latency timer can be set as low as 2ms. A setting of 1ms will cause unnecessary USB traffic and could adversely affect data transmission.

- ERRATIC MOUSE POINTER: The device driver defaults to query an attached device to find out whether it is a mouse or modem, consistent with native COM port operation. Some serial peripherals constantly send short packets of data, causing the host system to "think" a mouse or modem has been attached. These short packets will interfere with normal mouse operation causing the pointer to jump around the screen. If this happens, disconnect the CANbus network cable and *uncheck the Serial Enumerator* option, also found on the advanced driver options screen in Figure 2.13.

- COM PORT IN USE: Windows keeps track of all COM port assignments. If multiple Connective Peripheral products have been connected to a single system, the COM port number will increase, even if the other devices are not attached. If the higher COM port assignments are not acceptable for the application, known unused COM port numbers should be uninstalled according to the Installation guide: http://ftdichip.com/Documents/InstallGuides.htm.

10 Technical Support

Technical support may be obtained from your nearest Connective Peripherals office. See details below.

E-Mail (Support): support@connectiveperipherals.com

Web: http://www.connectiveperipherals.com/products

11 Contact Information

Global Headquarters – Singapore

Connective Peripherals Pte Ltd
178 Paya Lebar Road
#07-03
Singapore 409030

Tel: +65 67430980
Fax: +65 68416071

E-Mail (Sales)	sales@connectiveperipherals.com
E-Mail (Support)	support@connectiveperipherals.com
Web Site URL	http://www.connectiveperipherals.com
Web Shop URL	http://www.connectiveperipherals.com

Appendix B – List of Figures and Tables

List of Figures

List of Tables

Appendix B – Revision History

Revision	Changes	Date
1.0	Initial release	2009-04-01
1.01	Added reference to USB2-F-7001 API Guide	2009-05-01
1.02	Corrected Get Flags "F" command return value	2009-05-20
1.1	Changed "Acceptance Code" to "Acceptance Filter" throughout the document. Corrected Acceptance Mask and Acceptance Filter definitions. Added notices that USB Bus-Powered Hubs are not supported	2009-10-15
1.2	Added references to USB2-F-7101 and USB2-F-7x01 to cover both adapters Changed colours to black/silver	2010-02-10
1.3	Updated web references to single EasySYNC reference. Added Tech notes and apps notes references. Changed EasySync to EasySYNC to match logo	2010-05-12
1.31	Corrected initial LED power-on state	2010-10-01
1.32	Added LED Flicker state for Listen mode	2011-04-10
1.4	Re-branding to reflect the migration of the product from EasySync to Connective Peripherals name – logo change, copyright changed, contact information Changed, all internal hyperlinks changed.	2019-03-26