AceXtreme® Bridge Device

Software User's Manual

Model: BU-69094SX

The AceXtreme® Bridge Device Software Development Kit (SDK) provides the framework for efficient development of applications with DDC's series of MIL-STD-1553 Bridge Devices.

Custom Design Capability - DDC can customize designs for all cards, ranging from simple modifications of standard products to fully customized solutions for commercial, military, aerospace, and industrial applications.

For more information: www.ddc-web.com/BU-69094SX

DDC's Data Networking Solutions

MIL-STD-1553 | ARINC 429 | Fibre Channel | AFDX®/ARINC 664 | Ethernet

As the leading global supplier of data bus components, cards, and software solutions for the military, civil, and aerospace markets, DDC's data bus networking solutions encompass the full range of data interface protocols to support the real-time processing demands of field-critical data networking between systems and subsystems on military vehicles. These products, along with our traditional MIL-STD-1553 solutions, represent a wide and flexible array of performance and cost solutions, enabling DDC to support multi-generational programs.

Whether employed in increased bandwidth, high-speed serial communications, or traditional avionics and ground support applications, DDC's data solutions fulfill the expanse of military requirements including reliability, determinism, low CPU utilization, real-time performance, and ruggedness within harsh environments. Our use of in-house intellectual property ensures superior multi-generational support, independent of the life cycles of civil devices. Moreover, we maintain software compatibility between product generations to protect our customers' investments in software development, system testing, and end-product qualification.

MIL-STD-1553

DDC provides an assortment of quality MIL-STD-1553 rugged embedded and lab grade cards and components to meet your data conversion and data interface needs. Our 1553 data bus board solutions are integral elements of military, aerospace, and industrial applications. Our extensive line of military and space grade components provide MIL-STD-1553 interface solutions for microprocessors, PCI buses, and simple systems. Our 1553 data bus solutions are designed into a global array of aircraft, helicopter, unmanned vehicles, and missile programs.

ARINC 429

DDC also has a wide assortment of quality ARINC 429 embedded and lab grade cards and components, which will meet your data conversion and data interface needs. DDC's ARINC 429 components ensure the accurate and reliable transfer of flight-critical data. Our 429 interfaces support data bus development, validation, and the transfer of flight-critical data aboard civil aerospace platforms.

Fibre Channel

DDC has developed its line of high-speed Fibre Channel network access controllers and switches to support the real-time processing demands of field-critical data networking between sensors, computer nodes, data storage, displays, and weapons, for air, sea, and ground military vehicles. Fibre Channel's architecture is optimized to meet the performance,reliability, and demanding environmental requirements of embedded, real time, military applications, and designed to endure the multi-decade life cycle demands of military/aerospace programs.

AFDX®/ARINC 664

DDC provides powerful, field-proven AFDX®/ARINC 664 solutions for test, simulation, and system integration. These cards support both Airbus and Boeing AFDX protocols.

Ethernet

DDC offers a convienent solution to convert MIL-STD-1553, ARINC 429, and Ethernet protocol in any direction, in real-time, without a host computer.

Extensions to MIL-STD-1553

DDC offers a wide variety of solutions based on extensions of MIL-STD-1553 for emerging aerospace applications. Turbo 1553 increases the data rate of 1553 from 1 Mbps to 5 Mbps while maintaining the architectural features of MIL-STD-1553. Hyper 1553 provides high speed communication (50 to 100+ Mbps) over MIL-STD-1553 buses while operating concurrently with legacy 1 Mbps 1553 (similar to ADSL for telephone networks).

Form Factors, Software, & Drivers

DDC supplies MIL-STD-1553 and ARINC 429 board level products in a variety of form factors including USB, PCI-Express, PCMCIA, ExpressCard, AMC, PMC, XMC, PCI-104, PC/104-Plus, PC/104, PCI, cPCI, VME, and ISAbus boards. Our laboratory simulation and in-flight products include multi-function and single-function for system integration and production test environments. Our extensive line of military and space grade components provide MIL-STD-1553 interface solutions for microprocessors and simple systems. Our software is supplied in the form of menus, libraries, and drivers. We also offer additional software to expand our data networking range of options.

BU-69094SX

ACEXTREME® BRIDGE DEVICE SDK SOFTWARE USER'S MANUAL

MN-69094SX-001

105 Wilbur Place
Bohemia, New York 11716-2426
Tel: (631) 567-5600, Fax: (631) 567-7358
World Wide Web - http://www.ddc-web.com

For Technical Support - 1-800-DDC-5757 ext. 7771
United Kingdom - Tel: +44-(0)1635-811140, Fax: +44-(0)1635-32264
France - Tel: +33-(0)1-41-16-3424, Fax: +33-(0)1-41-16-3425
Germany - Tel: +49-(0)89-15 00 12-11, Fax: +49-(0)89-15 00 12-22
Japan - Tel: +81-(0)3-3814-7688, Fax: +81-(0)3-3814-7689
Asia - Tel: +65- 6489-4801

Revision	Date	Pages	Description
Pre Rev A	2/2013	All	Preliminary Release
Rev A	4/2013	All	Initial Release
Rev B	2/2015	30 - 113	Updated section 7 and section 8

1 PREFACE

This manual uses typographical conventions to assist the reader in understanding the content. This section will define the text formatting used in the rest of the manual.

1.1 Text Usage

- **BOLD** – text that is written in bold letters indicates important information and table, figure, and chapter references.
- `Courier New –` is used to indicate code examples.
- <…> - Indicates user entered text or commands.

1.2 Special Handling and Cautions

The BU-69094S is delivered on a Compact Disc. Proper care should be used to ensure that the discs are not damaged by heat.

1.3 Trademarks

All trademarks are the property of their respective owners.

1.4 Technical Support

In the event that problems arise beyond the scope of this manual, you can contact In the event that problems arise beyond the scope of this manual, you can contact DDC by the following:

US Toll Free Technical Support:
1-800-DDC-5757, ext. 7771

Outside of the US Technical Support:
1-631-567-5600, ext. 7771

Fax:
1-631-567-5758 to the attention of DATA BUS Applications

DDC Website:
www.ddc-web.com/ContactUs/TechSupport.aspx

Please note that the latest revisions of Software and Documentation are available for download at DDC's Web Site, www.ddc-web.com.

2 OVERVIEW

2.1 Description

The BU-67115W, BU-67116W and BU-67119W are different versions of DDC's AceXtreme Bridge Device (ABD). The first released version of the ABD is the BU-67119W non-ruggedized module for lab and production test applications. Future versions of the ABD will include the BU-67115W enclosed ruggedized module for embedded applications, and the BU-67116W ruggedized board, also for embedded applications.

The BU-67119W operates over an ambient air temperature range of 0 to 55° C, and includes an internal fan. The future BU-67115W and BU-67116W versions will operate over extended temperature ranges: 40 to 85° C case temperature, or -40 to 71° C ambient air temperature**.**

The ABD is available in a variety of channel configurations. All configurations include two 10/100/1000 Ethernet channels. Further, there are ordering options for one or two dual redundant MIL-STD-1553 channels, and for six ARINC 429 channels.

The ABD's MIL-STD-1553 channels can operate as BC, combined BC and Monitor, single RT, Multi-RT (up to 32) and combined single or Multi-RT and Monitor. All ARINC 429 channels may be independently programmed for either transmit or receive operation. The ABD also includes an IRIG-B time synchronization input, along with an option to configure the MIL-STD-1553 RT Address inputs to operate as up to 12 discrete digital I/Os.

The ABD offers a high degree of flexibility, and is therefore suitable for a wide range of embedded and lab applications. Initial versions of the ABD will provide its autonomous Protocol Conversion mode. Later versions of the ABD will include a remote access mode, in addition to the Protocol Conversion mode.

In its Protocol Conversion mode, the ABD is configured to provide autonomous communication bridging from any input channel(s) to any other channel(s). To minimize setup time and provide turnkey operation, the ABD includes a high-level Protocol Bridging layer API. Alternatively, users may develop their own bridging applications invoking DDC's AceXtreme MIL-STD-1553 and/or ARINC 429 APIs, along with the Linux socket interfaces for the two Ethernet channels.

In the future Remote Access Mode, users will be able to develop applications running on a remote computer communicating over Ethernet to the ABD's MIL-STD-1553 and/or ARINC 429 channels. In the remote access configuration, the user will be able to write applications running on a remote host invoking the AceXtreme MIL-STD-1553 and/or ARINC 429 APIs. As an alternative to developing application software, in

Remote Access Mode, the user will be able to operate the ABD using any of DDC's GUI software programs. These include:

- **BusTrACEr**, a simple menu program for generating and monitoring MIL-STD-1553 messages. In addition, BusTrACEr includes an option for the automatic generation of ANSI 'C' source code.
- **dataSIMS,** a software GUI tool for test and simulation applications. dataSIMS converts data to engineering units, allows the creation of graphical display formats, and may be used for either passive monitoring or simulation.
- **LabVIEW® and LabVIEW® Real-Time Support**. The BU-69093S0-XX0 software operates in conjunction with National Instruments' LabVIEW® or LabVIEW® Real-Time system design software to provide a simple interface and easy programming of the ABD's MIL-STD-1553 and/or ARINC 429 interfaces. Users can either create their own custom interfaces "from scratch" or may modify the samples that are provided.
- **Commercial Avionics Utilities Software Package.** The DD-42999S0-XX0 Data Bus Analyzer and Data Loader GUI software is for ARINC 429 data bus analysis and simulation. This GUI provides advanced filtering, message scheduling and triggering. In addition, it includes a graphical ARINC 615 data loader, providing a software interface to load data to and from airborne computers.

2.2 Features

General

- Overall Configuration:
  - Two 10/100/1000 BASE-T Ethernet Channels
  - Options for One or Two MIL-STD-1553 Channels.
    - In the future, the number of 1553 channels will be able to be increased to 3 or 4 by the installation of a DDC BU-67114Hx Mini-PCI Express card into the ABD's Mini-PCIe expansion slot
  - Option for Six ARINC 429 Transmit or Receive Channels
  - Intel Atom Processor (Dual-Core)
  - Embedded Linux Operating System
- Bridge Any Channel to Any Other Channel(s)
- Self-Contained Development Environment
- Lab and Rugged Versions
- Bridging Mode
  - Standalone "anything-to-anything" Bridge: Ethernet, 1553, ARINC 429
  - Bridging API for Turnkey Solutions

  - Ethernet Sockets, 1553 and ARINC 429 APIs for added flexibility
  - Built-in Linux development environment
- Remote Access Mode (Coming Soon!)
  - Enables Communication Over Ethernet to/from 1553 and ARINC 429 Channels
  - 1553 and ARINC 429 Software Drivers and/or GUIs Installed on Remote Host Computer

MIL-STD-1553

- One or Two Dual Redundant 1553 channels
  - Each 1553 channel can be independently programmed for BC, BC/MT, RT, Multi-RT, Monitor, RT/Monitor, or Multi-RT/Monitor operation.
  - Transformer-coupled 1553 bus connection (Consult factory for direct-coupled).
- 2 MB RAM per 1553 Channel
- Each Dual Redundant MIL-STD-1553 Channel:
  - BC or Multi-RT with Concurrent Bus Monitor
  - Support of MIL-STD-1553 A/B, STANAG-3838, and MIL-STD-1760
  - 2 Mb (64K x 36) Shared RAM
  - Transmit Inhibit Inputs for Monitor-only Applications
  - BC Disable Inputs for RT-only Applications
  - 48-bit/100ns Time Stamp
  - IRIG-106 Chapter 10 Monitor
- 1553 Bus Controller (BC)
  - Highly Autonomous Controller, with 32-Instruction Set
  - Streaming and Minor/Major Frame Scheduling of Messages
  - High and Low Priority Asynchronous Message Insertion
  - Modify Messages or Data While BC is running
- 1553 Remote Terminal (RT)
  - Emulate up to 31 RT Addresses Simultaneously
  - Multiple Buffering Techniques
  - Programmable Command Illegalization
  - Programmable Busy by Sub-address
  - RT Auto-Boot Option for MIL-STD-1760

- 1553 Bus Monitor (MT)
  - IRIG-106 Chapter 10 Compatibility
  - Filter Based on RT Address, T/R bit, Sub-Address
  - 48-bit, 100 ns/LSB Time Stamping
  - Advanced Bit Level Error Detection to Isolate Bus Failures
- IRIG-B Clock
  - IRIG-B Digital Time Code Input Enables 1 Hz Synchronization
  - 48-bit resolution, with options of 64, 32, …1 µS/LSB; or 500, 250, or 100 ns/LSB
  - 100 ns/LSB resolution for IRIG 106 Chapter 10 Monitor
- Capability for Fast DMA Transfers between 1553 Shared RAM and Atom Processor Local Memory.
- API Compatible with AceXtreme® MIL-STD-1553 SDK (BU-69092Sx)

ARINC 429

- Up to Six ARINC 429 channels programmable for Transmit or Receive operation
- High or Low Speed Orientation per ARINC 429 Channel
- Scheduled and FIFO ARINC 429 transmission
- FIFO and mailbox ARINC 429 reception
- API Compatible with DDC Commercial Avionics SDK (DD-42992Sx)

Software

- Linux Operating System and BSP
  - Ethernet Stacks, with UDP/IP and TCP/IP Sockets, Telnet, FTP, TFTP, SSH, and HCTP.
- DDC Protocol Bridging API, Providing Turnkey Bridging From Any Port to Any Other Port(s)
- DDC AceXtreme MIL-STD-1553 and ARINC 429 APIs, Including Sample Programs
- Built-in Editor, Allowing Editing and Saving Files Over Telnet
- Built-in 'C' Compiler
- Can Transfer Internal Files to a Host Computer, Edit Remotely, and Transfer Back to the ABD Before Compiling.

2.3 System Requirements

2.3.1 System Requirements for Protocol Conversion Mode

- Remote computer with Ethernet interface and Telnet.

2.3.2 System Requirements for Remote Access Mode

- Remote computer with Ethernet interface.
- Windows 2000/XP, Windows Vista 32/64-bit, Windows 7 32/64-bit, Linux, or VxWorks Operating System
    - Tornado software development environment for VxWorks platforms
- An appropriate compiler or development environment.
- Contact factory for additional operating systems

2.4 MIL-STD-1553 Capability

The AceXtreme Bridge Device implements provides up to two MIL-STD-1553 channels utilizing the DDC **AceXtreme** architecture. Each 1553 channel can be independently programmed for BC, RT, Monitor or true RT/Monitor and is configured with 2MB of onboard RAM per installed channel.

Each MIL-STD-1553 interface implements a transformer-coupled dual-redundant bus connection (consult factory for direct-coupled).

2.4.1 Bus Controller Mode

The **AceXtreme's** MIL-STD-1553 Bus Controller (BC) is based on the 32-bit architecture of DDC's **AceXtreme** 1553 Bus Controller.

AceXtreme's BC architecture retains much of the previous generation (Enhanced Mini-ACE, Mini-ACE Mark 3, Micro-ACE (TE), and Total-ACE) 1553 Bus Controller architecture. However, it expands upon it in specific areas to provide improved capabilities.

The **AceXtreme™** BC architecture is based on a built-in command interpreter with a set of 32 instructions. The command interpreter is a message sequence control engine that provides a high degree of flexibility for implementing 1553 Message lists, including major and minor frame scheduling. It separates 1553 message data from control/status data for the purposes of implementing different data block handling schemes, performing bulk data transfers, and implementing automatic message retries. It also includes the capability for automatic bus switchover for failed messages and reporting of various error and status conditions to the host processor by means of five user-defined interrupts and a general-purpose queue.

Two Asynchronous queues are also included, to improve the Bus Controller's efficiency and flexibility. The High Priority Queue (HPQ) enables the user to easily insert asynchronous messages into a running Message list, causing it to operate on the new message immediately. The Low Priority Queue (LPQ) enables the user to insert asynchronous messages which will only be processed when there's sufficient "dead-time" available on the bus at the end of a minor frame.

The **AceXtreme's** BC Engine implements all MIL-STD-1553B message formats. Message format is programmable on a message-by-message basis. Automatic retries and interrupt requests may be enabled or disabled for each individual messages. The BC performs all error checking required by MIL-STD-1553B. This includes validation of response time, sync type and sync encoding, Manchester II encoding, parity, bit count, word count, Status Word RT Address field, and various RT-to-RT transfer errors. The BC No-Response timeout value is also programmable to enable operation over long buses or through repeaters.

2.4.2 Remote Terminal Operation

The **AceXtreme** RT architecture builds upon the single-RT architecture of Enhanced Mini-ACE, Mini-ACE Mark 3, Micro-ACE(TE), and Total-ACE.

One of the major new features of **AceXtreme** is its Multi-RT capability. That is, the **AceXtreme** provides the capability to implement up to 31 independent Remote Terminals (up to 32 RTs if Broadcast is disabled).

The **AceXtreme** RT engine can also be configured to operate in a Single-RT legacy mode of operation. Single-RT operation supports hardware control of the RT address and automatic boot, allowing the **AceXtreme** to respond to commands with Status with its Busy bit set immediately following power turn-on without requiring configuration by the host.

For RT (and/or Monitor) applications, where the possibility of BC operation must be absolutely prohibited, the AceXtreme Bridge Device includes a DISABLE_BC input signal. In addition to single-RT and Multi-RT operation, the **AceXtreme** includes the following capabilities:

- Meets MIL-STD-1553A, MIL-STD-1553B, MIL-STD-1760, and STANAG 3838 standards.
- Multiple Data Handling Modes:
    - Single Buffer Mode

    - Double Buffer Mode
    - Circular Buffer Mode
    - Global Circular Buffer

- Command Illegalization by Subaddress/Word Count, and Mode Codes
- Programmable Busy by Subaddress
- Flexible Interrupt Conditions, Including 50% and 100% Rollover Interrupts for Circular Buffers
- Interrupt Status Queue with Programmable Filtering
- Time Tagging Options for Synchronize Mode Codes
- Option for RT Auto-Boot with Busy bit Set

2.4.3 Monitor Mode Operation

The **AceXtreme** Monitor engine provides the next generation DDC MIL-STD-1553 Monitor (MT) architecture. This new Monitor autonomously stores individual messages into a contiguous memory stack formatted as IRIG 106 Chapter 10 Data Packets.

The legacy Monitor modes of operation traditionally implemented in previous generations of DDC MIL-STD-1553 engines can be emulated easily through host software. All information and functionality supported on the legacy Monitor engines is supported on the **AceXtreme** Monitor engine, or may be extracted from the stored messages.

IRIG 106 Chapter 10 provides interoperability for such applications as test range telemetry, flight test instrumentation, mission recorders, video/data servers; surveillance and reconnaissance; health and usage monitoring; mission planning, debriefing, and training; and flight operations. IRIG 106 Chapter 10 defines a standardized file format, and within that specific representations for several types of flight data, including MIL-STD-1553 buses, PCM, analog, computer-generated data, images, discretes, UARTs, IEEE 1394, parallel, IRIG time, video, and voice. In addition, Chapter 106 provides standardization of time bases.

For MIL-STD-1553, IRIG 106 Chapter 10 defines packets that can encapsulate one of more 1553 messages. Within these packets, all messages are tagged with either a 48-bit relative or 64-bit absolute time stamp. For each message, there is also a block status word, which includes indications of bus channel and message validity, and identifies specific errors. The 1553 format also defines indications of response time, plus storage of all 1553 Command, Status, and Data Words, in the order received. For supporting IRIG 106 Chapter 10, the AceXtreme Bridge Device includes DMA capability, which enables high-speed transfers of monitored data from the 1553 monitor to the AceXtreme Bridge's Atom processor host space.

2.5 ARINC-429 Capability

The ABD offers an option to include six ARINC-429 channels. Each channel may be programmed to operate as a transmit or receive channel. All channels comply with the ARINC 429 electrical specification.

Each channel implements numerous software configurable interrupt, data handling, and error control options.

2.6 10/100/1000 Ethernet Capability

The ABD includes two Ethernet interfaces. Both Ethernet interfaces are capable of operating over 10 BASE-T, 100 BASE-T, or 1000 BASE-T physical layers, with auto-negotiation capability. The Linux stack running on the ABD's Atom processor supports TCP/IP and UDP/IP protocols.

2.7 Digital Discrete I/O Capability

The BU-67107F/M/I/T card includes six discrete digital signals that are individually programmable as inputs or outputs.

Discrete digital I/O channel 1 is the LSB and channel 6 is the MSB for any software accesses to these signals.

The discrete digital I/O signals are powered from an internal 3.3V power supply. The I/O block consists of a tri-stable output stage capable of 12 mA drive combined with a pulled-up, 3.3V tolerant, input stage. After power up/reset the output stage is tri-stated, presenting a pulled-up input to the Pn4 or front panel connector

The discrete outputs can be used for a variety of applications, including triggering events, indicating status, or general-purpose use.

3 MODES OF OPERATION

The ABD is planned to support two distinct modes of operation: (1) a protocol conversion mode; and (2) a remote access mode. The initial release of the ABD product will provide only the protocol bridging mode, while subsequent versions of the product will provide the remote access mode in addition to the protocol bridging mode.

3.1 Protocol Conversion Mode

Figure 1 shows an example of the AceXtreme Bridge Device operating in its Protocol Conversion Mode. In this mode, the ABD may be configured to provide autonomous communication bridging between any channel and any other channel(s).

To minimize setup time and provide turnkey operation, the ABD includes a high-level protocol bridging API. Alternatively, users may develop their own bridging applications invoking DDC's AceXtreme MIL-STD-1553 and/or ARINC 429 APIs, along with the Linux socket interfaces for the two Ethernet channels.

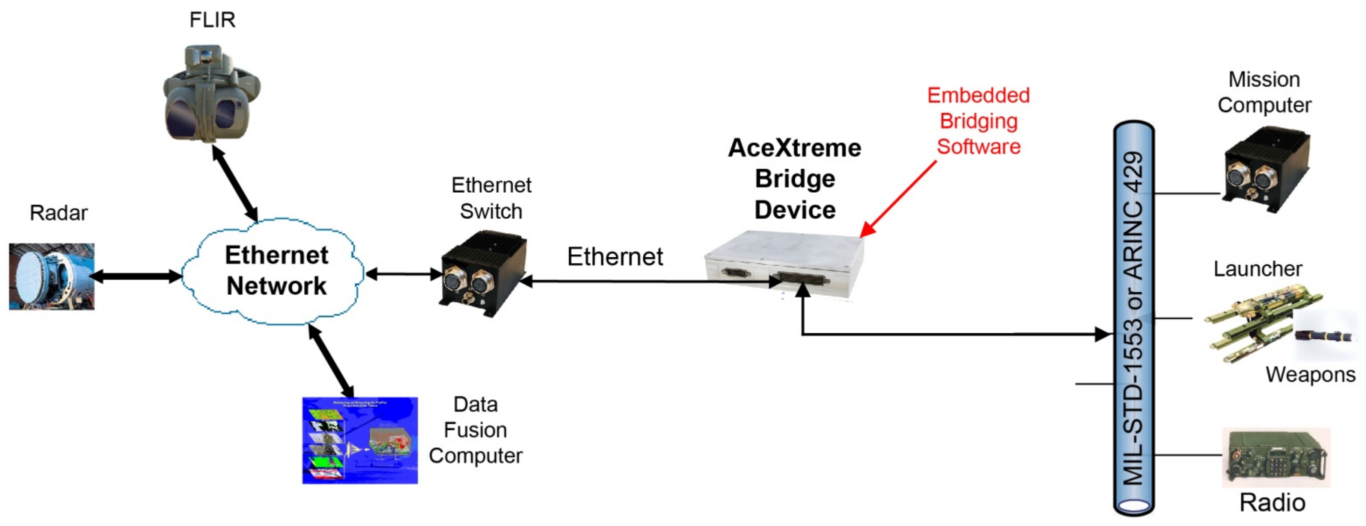


Figure 1. Example of the AceXtreme Bridge Device Used in Protocol Conversion Mode

3.2 Remote Access Mode

Figure 2 shows an example of the AceXtreme Bridge Device operating in its remote access mode. In remote access mode, users can develop applications running on a remote computer that communicates over Ethernet to the ABD's MIL-STD-1553 and/or ARINC 429 channels. In the remote access configuration, the user will be able to write applications running on a remote host that invoke the AceXtreme MIL-STD-1553 and/or ARINC 429 APIs. For use in remote access mode, DDC offers AceXtreme software drivers Remote Host Drivers for Windows 2000, XP, Vista, and 7; Linux kernel version 2.6.x; and WindRiver VxWorks versions 6.x.

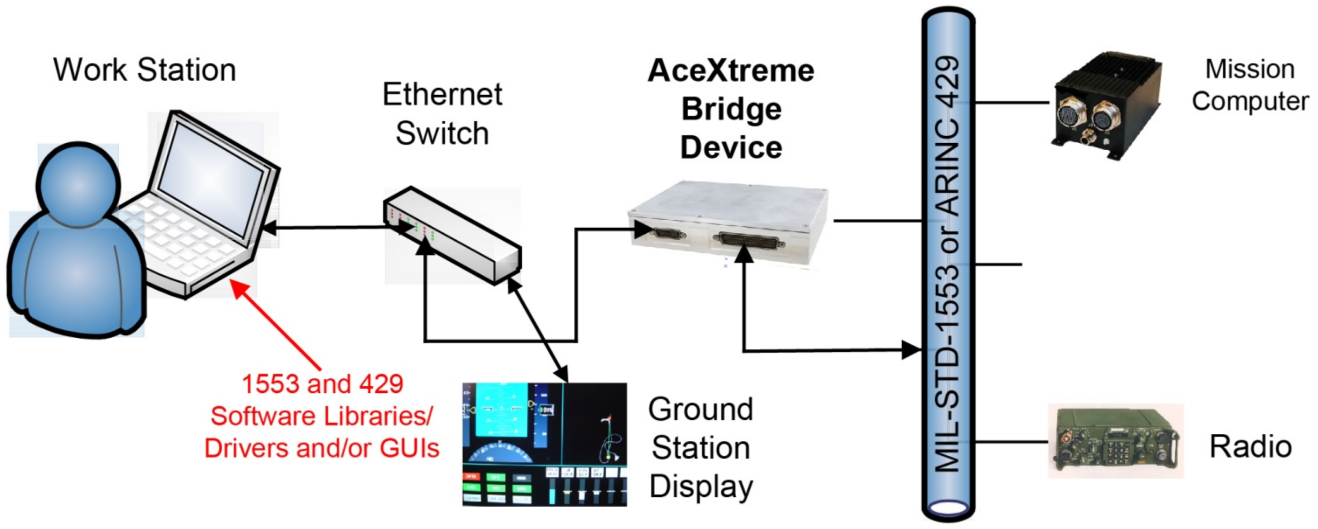


Figure 2. Example of the AceXtreme Bridge Device Used in Remote Access Mode

4 DEVICE STARTUP

The following section defines the steps to initially setup the AceXtreme Bridge Device and remotely communicate with it.

4.1 Minimal Cable Connections

The following is a list of equipment required to configure and communicate with the ABD. Please see the BU-6711[5/6/9] Quick Start Guide for more information.

- DDC AceXtreme Bridge Device (BU-6711[5/6/9])
- Standard RJ-45 Ethernet Cable.
- Wall Outlet.
- ABD Power Supply Adapter (DDC-78164-1)
- Desktop/Laptop with Telnet/SSH Client Installed.
- o Ability to assign static IP address/mask.
- BU-6711[5/6/9]W 'J1' Cable Assembly (DDC-78053-1)
- BU-6711[5/6/9]W 'J2' Cable Assembly (DDC-78055-1)

4.2 Powering up the device

- First make sure to unpack the AceXtreme Bridge Device and connect both '**J1**' and '**J2**' cable assemblies.
- Connect the Ethernet cable to the '**ETH-1'** connector on the '**J1**' Cable Assembly and your Desktop/Laptop's Ethernet Port.
- Connect the ABD Power Supply Adapter to the power receptacle on the '**J2**' Cable Assembly.
  - o See the AceXtreme Bridge Device Hardware Manual for more information.
- Connect the ABD Power Adapter to the Wall Outlet
- Place the device in "Configuration Mode" by attaching the supplied BNC Terminator on the 'J2' cable.
  - o See Section 9.1 for more information on Deployed Device States.
- For BU-67119W Models, Toggle the 'PWR' Switch to turn on the device.
- For All other models, plugging in the adapter will automatically apply power.

4.3 Internal Date and Time

The AceXtreme® Bridge device does not contain a battery backup to store system time. By default, the device will boot up with the following date and time.

January 01, 2020 08:00:00 AM

This date and time is statically set at the end of the ABD startup script (/etc/init.d/ddc_inet.sh).

If you would like to configure a different date and/or time, please edit the above script accordingly.

Alternatively, it is acceptable to query an NTP server or equivalent if it is available in your network configuration.

4.4 IP Address Configuration

DDC's AceXtreme Bridge Device will ship from the factory with a static IP configuration (See Section 4.3.1).

It is highly likely that you will need to modify the IP Settings before connecting the device to the desired final network location.

Once the ABD IP address settings have been modified for your network, it is acceptable to connect the device to any switch within your LAN.

In order to connect to the ABD for the first time, please set your host laptop/desktop to an IP address on the 192.168.1.x network.

(Reccomended: address: 192.168.1.90 netmask: 255.255.255.0)

4.4.1 Factory/Recovery IP Address Settings

Default configuration for the ABD IP Address settings is as follows:

Device:	**Ethernet Port 1 (eth0)**
Address:	**192.168.1.80**
Netmask:	**255.255.255.0**
Gateway:	**0.0.0.0**

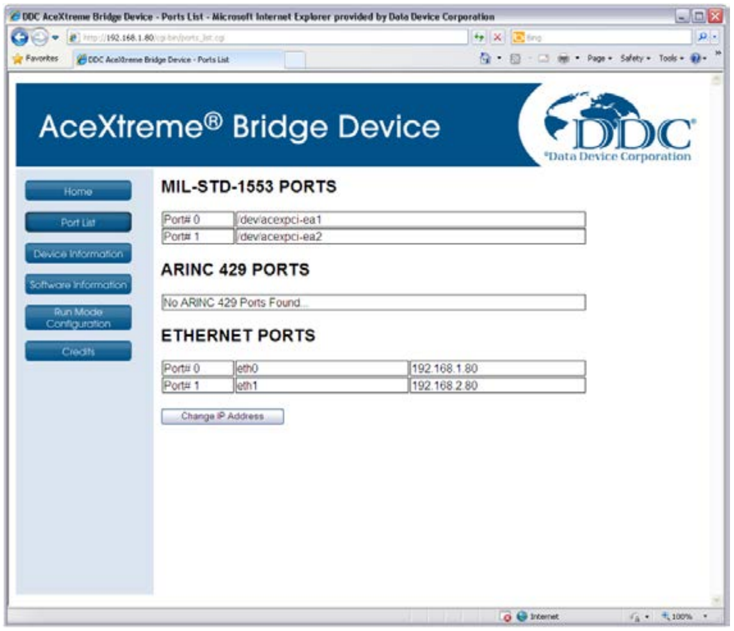
Device:	**Ethernet Port 2 (eth1)**
Address:	**192.168.2.80**
Netmask:	**255.255.255.0**
Gateway:	**0.0.0.0**

Note, if the IP Address settings are accidentally lost, you can boot the device in ‘Recovery Mode’, which will always set the IP settings to the above.

Please see Section 9.2 for more information on Recovery Mode.

4.4.2 Modifying ABD Static IP Configuration

The Ports List link from the AceXtreme Bridge Device Web Server displays the installed ports on the AceXtreme Bridge Device (ABD). The ports include the number of MIL-STD-1553 and ARINC-429 channels, along with the Ethernet Ports on the ABD.


DDC AceXtreme Bridge Device - Ports List - Microsoft Internet Explorer provided by Data Device Corporation
http://192.168.1.80/cgi-bin/ports_list.cgi
Favorites
DDC AceXtreme Bridge Device - Ports List
Page
Safety
Tools
AceXtreme® Bridge Device
DDC
Data Device Corporation
Home
Port List
Device Information
Software Information
Run Mode Configuration
Credits
Change IP Address
Internet
100%

MIL-STD-1553 PORTS

Port# 0	/dev/acexpci-ea1
Port# 1	/dev/acexpci-ea2

ARINC 429 PORTS

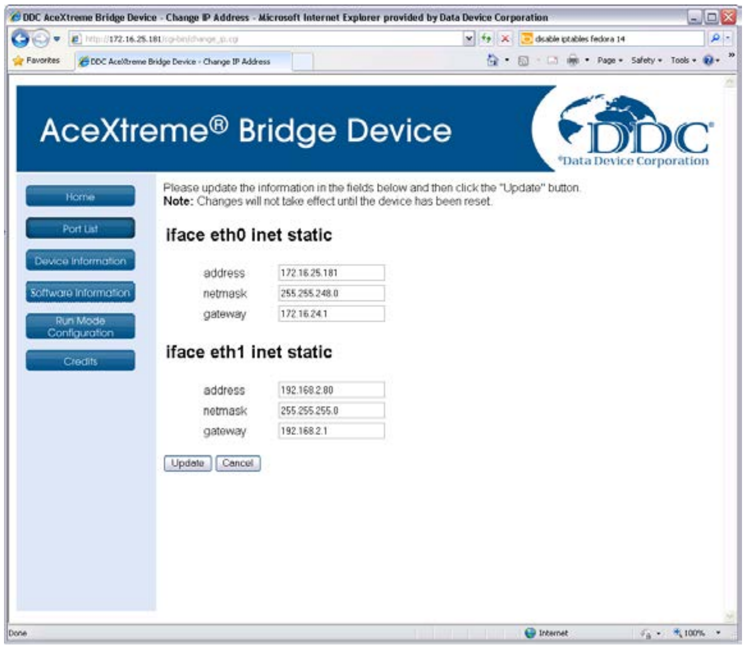
No ARINC 429 Ports Found...

ETHERNET PORTS

Port# 0	eth0	192.168.1.80
Port# 1	eth1	192.168.2.80

Figure 3. Ports List from Web Server

The Change IP Address button will open a new page that allows the user to configure the static IP address of the AceXtreme Bridge Device. Make sure to click ‘Update” to save changes, then cycle device power to apply IP addresses.


DDC AceXtreme Bridge Device - Change IP Address - Microsoft Internet Explorer provided by Data Device Corporation
http://172.16.25.181/cgi-bin/change_ip.cgi
disable iptables fedora 14
Favorites
DDC AceXtreme Bridge Device - Change IP Address
Page
Safety
Tools
AceXtreme® Bridge Device
DDC
Data Device Corporation
Home
Port List
Device Information
Software Information
Run Mode Configuration
Credits
Update
Cancel
Done
Internet
100%

Please update the information in the fields below and then click the "Update" button.
Note: Changes will not take effect until the device has been reset.

iface eth0 inet static

address	172.16.25.181
netmask	255.255.248.0
gateway	172.16.24.1

iface eth1 inet static

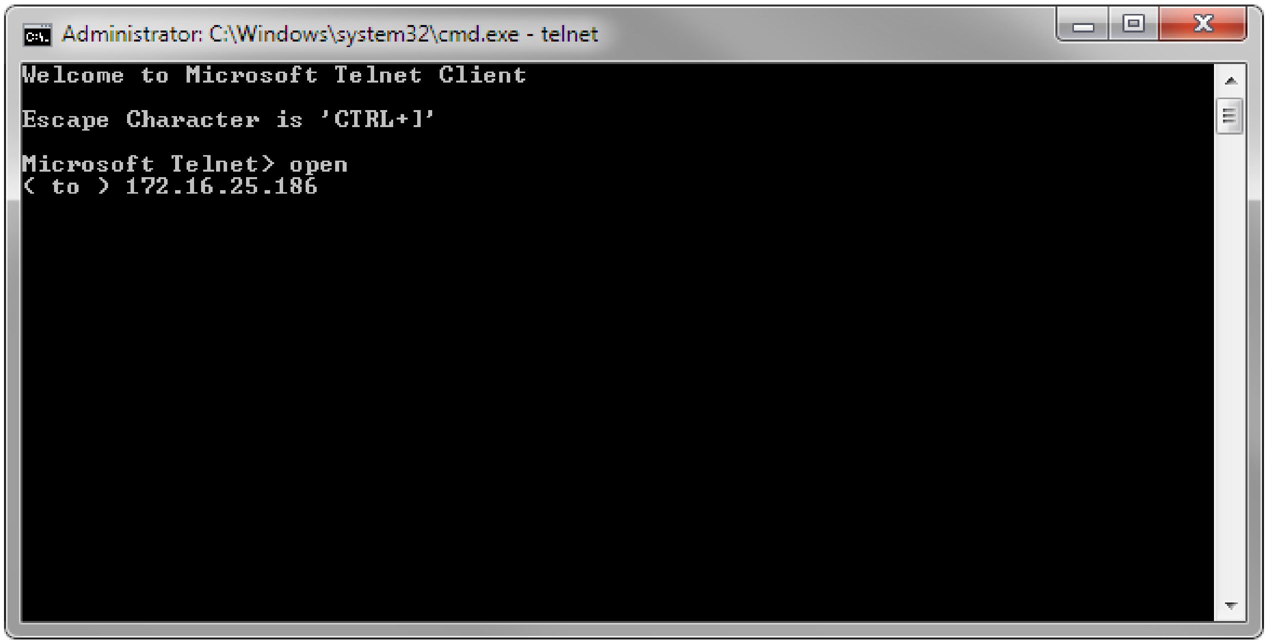
address	192.168.2.80
netmask	255.255.255.0
gateway	192.168.2.1

Figure 4. Changing IP Address through Web Server

There are two configurable ports on the ABD, the IP address, netmask and gateway are all configurable on the ABD. Make sure these settings are configured according to your network settings in order to access the device over your network. These settings can only be modified while the device is in configuration mode

4.5 Logging into the device

To connect to the AceXtreme Bridge Device through Telnet, open a Telnet shell on your host system and type “telnet” at the prompt. Once the telnet application has been started, type “open” and enter the IP address of your AceXtreme Bridge Device (see Section 4.4.2).

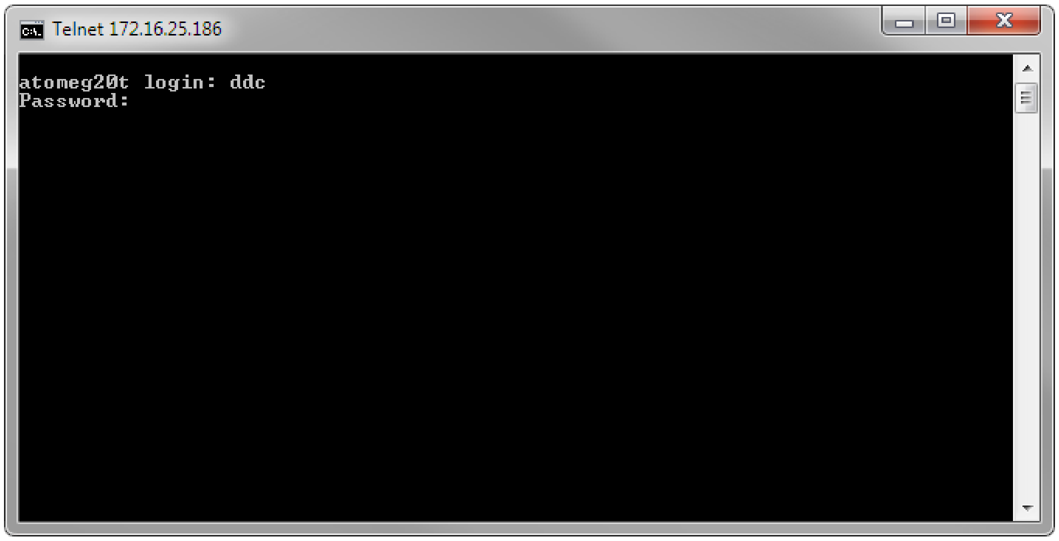


```
Welcome to Microsoft Telnet Client

Escape Character is 'CTRL+]'

Microsoft Telnet> open
( to ) 172.16.25.186
```

Figure 5. Connecting to ABD with Telnet

> ***Note:*** *The Telnet service is not installed by default on a Windows 7 PC. To enable the Telnet sevice see section 6.2.1.*

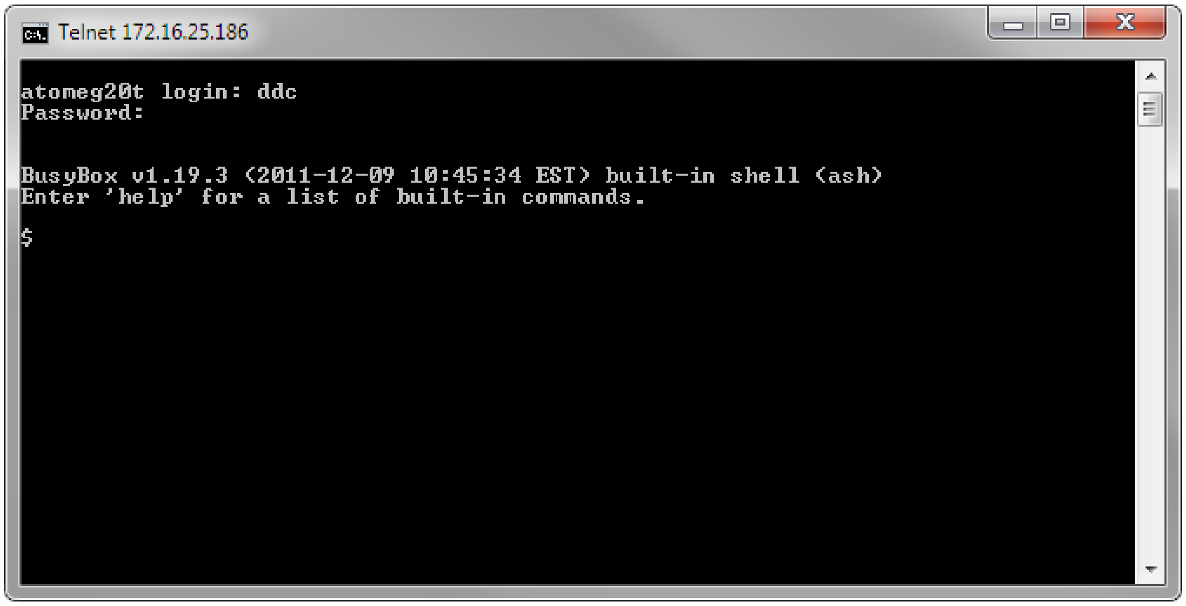
Once you have connected to the ADB with Telnet, you will be prompted to enter the username and password of your account on the ABD.



```
atomeg20t login: ddc
Password:
```

Figure 6. Login via Telnet

After entering the username and password you are now ready to start using the Linux Operating system installed on the AceXtreme Bridge Device.



```
atomeg20t login: ddc
Password:


BusyBox v1.19.3 (2011-12-09 10:45:34 EST) built-in shell (ash)
Enter 'help' for a list of built-in commands.

$
```

Figure 7. Logged into ABD via Telnet

5 USING THE REMOTE SHELL INTERFACE

DDC's AceXtreme Bridge Devices' (ABD) use the ASH built-in Shell interface, distributed by BusyBox. The shell gives the user access to the internal file system, compilers and sample applications.

The ABD will ship from the factory with all necessary Software to begin developing custom bridging applications.

Users familiar with UNIX/LINUX shell interfaces should be comfortable using the built-in shell.

For more information on using the ASH Shell, see www.busybox.net

5.1 Usernames and Passwords

All DDC AceXtreme Bridge Devices' ship with 2 user accounts, **root** and **ddc**. It is recommended that the '**root'** account only be used for extraordinary circumstances that require "Administrator" access. The '**ddc'** account can be used for normal configuration and programming of the device.

Username: **ddc**	Password: **ddc**	Description: **Normal User**
Username: **root**	Password: **ddc**	Description: **Adminsitrator**

Note, that if the device is booted in RECOVERY Mode, the above username/passwords will always be valid. See Section 9.2 for more information.

5.2 Changing Passwords

To change the password for either the **root** or the **ddc** account, first login (See Section 4.4) to the device as the user whose password is to be changed.

Type the following command:

```
passwd
```

You will be instructed to enter the users current (old) password, followed by the new desired password. You should observe a confirmation that the password was changed successfully.

```
# passwd
Changing password for ddc
Old password:  << TYPE OLD PASSWORD >>
New password:  << TYPE NEW PASSWORD >>
Password updated Successfully!
```

5.3 Directory Structure

The following is the directory structure within the AceXtreme Bridge Device. It consists of numerous samples, libraries and build files to show proper use of the DDC SDK Programming Interface.

All user files reside in the **"/home"** directory, which will be the default directory after logging in.

Table 1. AceXtreme Bridge Device SDK Directory Structure

Folder Name	Description of Contents
/home/ddc	Main User DDC Home Directory.
/home/ddc/run_mode_prog	Contains applications to execute on startup.
/home/ddc/ABD/429samples	Samples on using DDC's ARINC 429 SDK directly.
/home/ddc/ABD/1553samples	Samples on using DDC's MIL-STD-1553 SDK directly.
/home/ddc/ABD/ABDbridging	Protocol Bridging Layer (PBL) Home Directory.
/home/ddc/ABD/ABDbridging/library	Contains source code to rebuild the Protocol Bridging Layer.
/home/ddc /ABD/ABDbridging/samples	Contains samples for the Protocol Bridging Layer (PBL).

5.4 Editing Files

Once you have initiated a Telnet/SSH session (See Section 4.4), you can use the built-in editor to edit and save any text-based files (Such as Sample 'C' Source code).

The Built-in editor is *vi*, a popular text-based file editor.

Alternatively, you can transfer any file to a remote computer, edit them remotely, and transfer them back to the AceXtreme Bridge Device before compiling. See Section 5.5 for more information on transferring files.

For more information on using the *vi* Editor, please see www.vim.org

5.5 Transferring Files

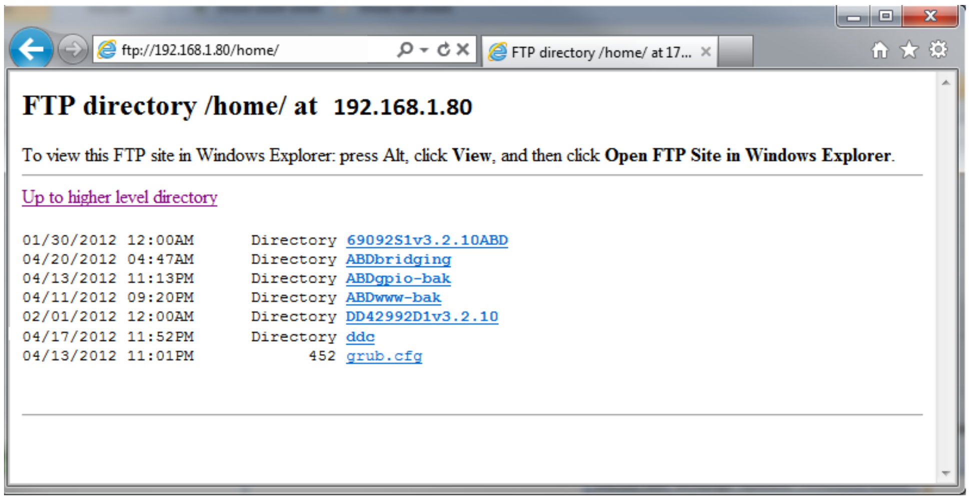
The AceXtreme Bridge Device contains an FTP (File Transfer Protocol) Server that allows external clients to transfer files remotely to and from the device. Note, the FTP Sever Port is 21. See Section 5.1 for login information.

FTP Clients are included with Windows, Linux, as well as numerous 3rd Party Tools.

Please the Below Links for more information using FTP.

Using Wiindows: http://windows.microsoft.com/en-us/windows-vista/File-Transfer-Protocol-FTP-frequently-asked-questions

Using Linux: http://tldp.org/HOWTO/FTP-3.html



FTP directory /home/ at 192.168.1.80

To view this FTP site in Windows Explorer: press Alt, click **View**, and then click **Open FTP Site in Windows Explorer**.

Up to higher level directory

```
01/30/2012 12:00AM      Directory 69092S1v3.2.10ABD
04/20/2012 04:47AM      Directory ABDbridging
04/13/2012 11:13PM      Directory ABDgpio-bak
04/11/2012 09:20PM      Directory ABDwww-bak
02/01/2012 12:00AM      Directory DD42992D1v3.2.10
04/17/2012 11:52PM      Directory ddc
04/13/2012 11:01PM            452 grub.cfg
```

Figure 8. Accessing ABD FTP Server using Windows Client (IE)

6 CONFIGURING NETWORK SERVICES

The AceXtreme Bridge Device is configured with a Web, Telnet, and FTP server. The Web, Telnet and FTP servers will give the required access to configure, and use the AceXtreme Bridge Device.

The ABD will ship from the factory with all supported services enabled. Users familiar with UNIX/LINUX shell interfaces should be comfortable using the built-in servers.

6.1 HTTP (Web) Server

The Web Server can be used to configure the AceXtreme Bridge Device while in configuration mode (see Section 9.1). The web server will provide information on the number of MIL-STD-1553 channels, ARINC-429 Receiver / Transmitter channels, and the Ethernet Ports on the ABD. The Web Server will also allow the user to modify settings on the ABD, such as the IP address of the Ethernet port, upgrade firmware, drivers or library of the MIL-STD-1553 and ARINC-429 channels.

6.1.1 Connecting to Web Server

To connect to the Web Server, open a web browser (such as Internet Explorer) and enter in the IP address of your device (see section 4.3) into the address bar of your web browser.

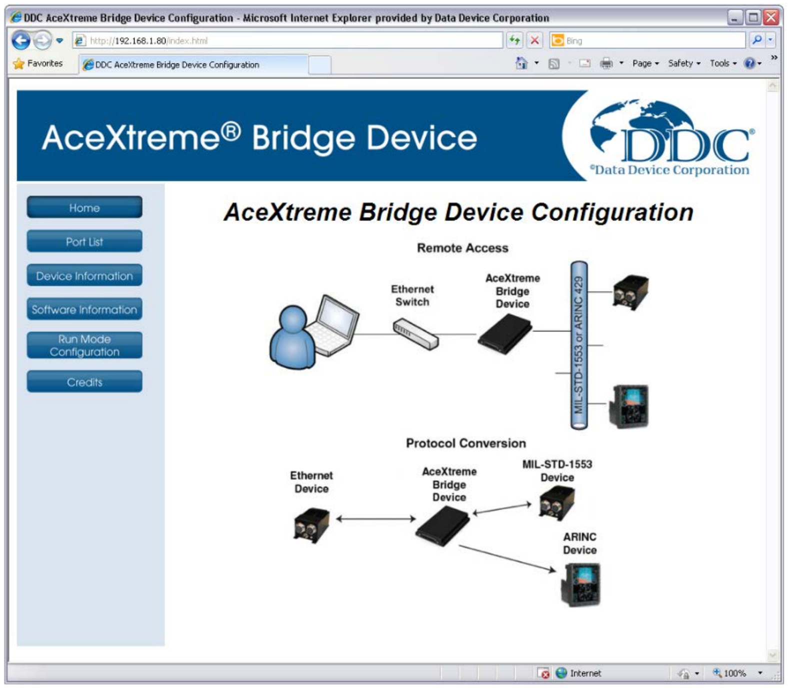
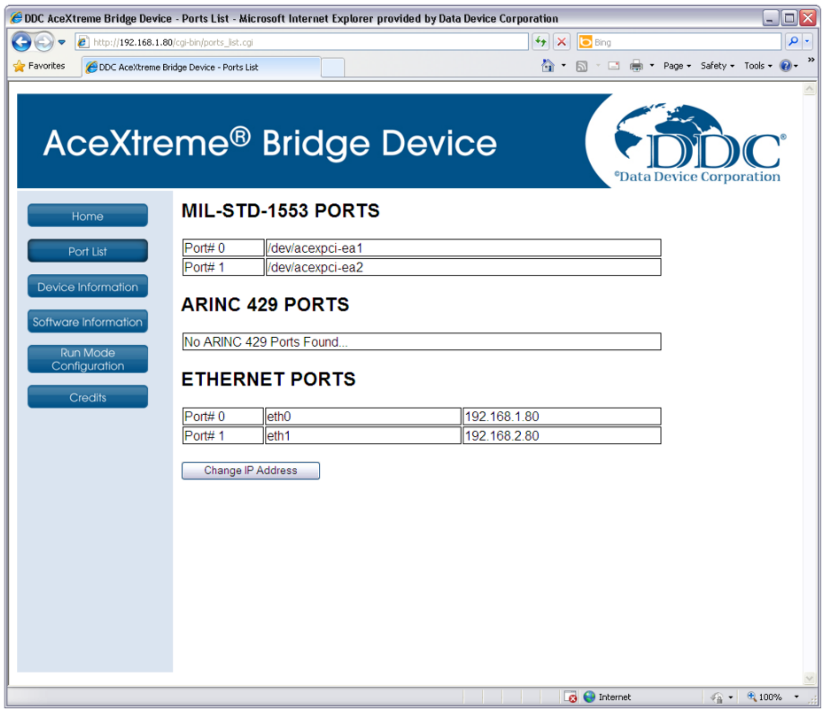


Figure 9. Accessing ABD Web Server using Windows Client (IE)

6.1.1 Ports List

The Ports List link from the AceXtreme Bridge Device Web Server displays the installed ports on the ABD. The ports include the number of MIL-STD-1553 and ARINC-429 channels, along with the Ethernet Ports on the ABD.


DDC AceXtreme Bridge Device - Ports List - Microsoft Internet Explorer provided by Data Device Corporation
http://192.168.1.80/cgi-bin/ports_list.cgi
Favorites
DDC AceXtreme Bridge Device - Ports List
Page
Safety
Tools
AceXtreme® Bridge Device
DDC
Data Device Corporation
Home
Port List
Device Information
Software Information
Run Mode Configuration
Credits
Change IP Address
Internet
100%

MIL-STD-1553 PORTS

Port# 0	/dev/acexpci-ea1
Port# 1	/dev/acexpci-ea2

ARINC 429 PORTS

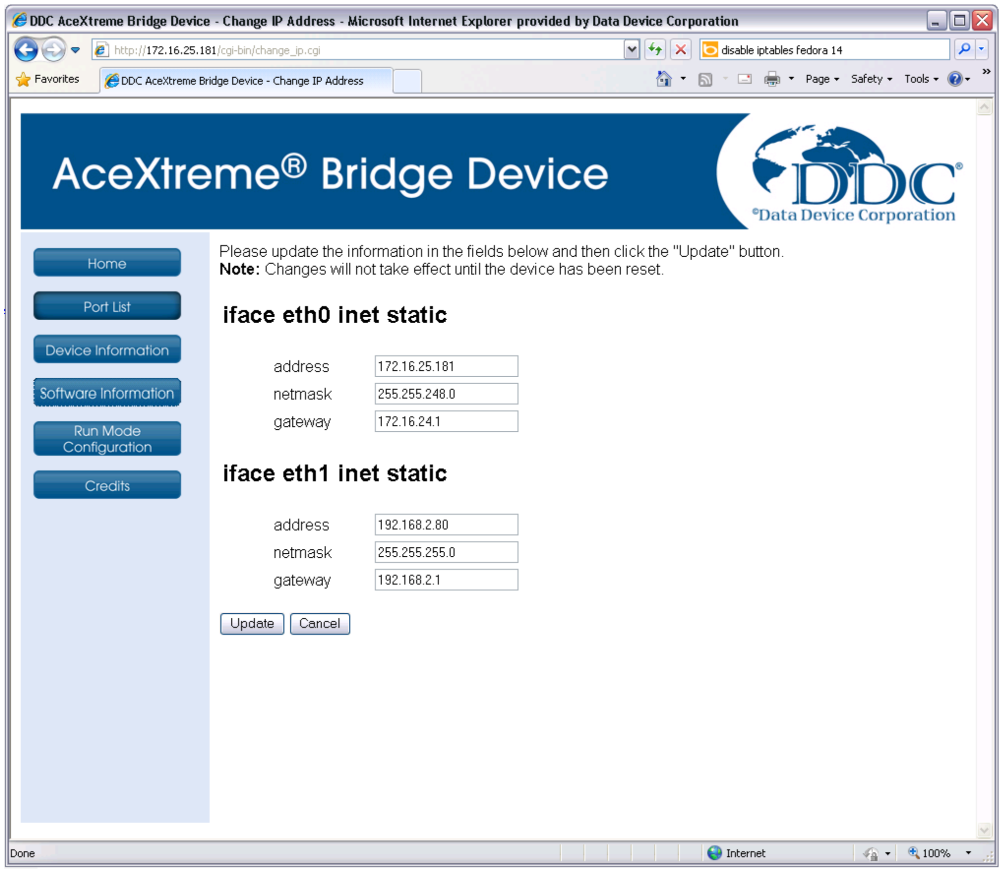
No ARINC 429 Ports Found...

ETHERNET PORTS

Port# 0	eth0	192.168.1.80
Port# 1	eth1	192.168.2.80

Figure 10. Ports List from Web Server

The Change IP Address button will open a new page that allows the user to configure the IP address of the AceXtreme Bridge Device.


DDC AceXtreme Bridge Device - Change IP Address - Microsoft Internet Explorer provided by Data Device Corporation
http://172.16.25.181/cgi-bin/change_ip.cgi
AceXtreme® Bridge Device
Home
Port List
Device Information
Software Information
Run Mode Configuration
Credits
Update
Cancel

Please update the information in the fields below and then click the "Update" button.
Note: Changes will not take effect until the device has been reset.

iface eth0 inet static

address	172.16.25.181
netmask	255.255.248.0
gateway	172.16.24.1

iface eth1 inet static

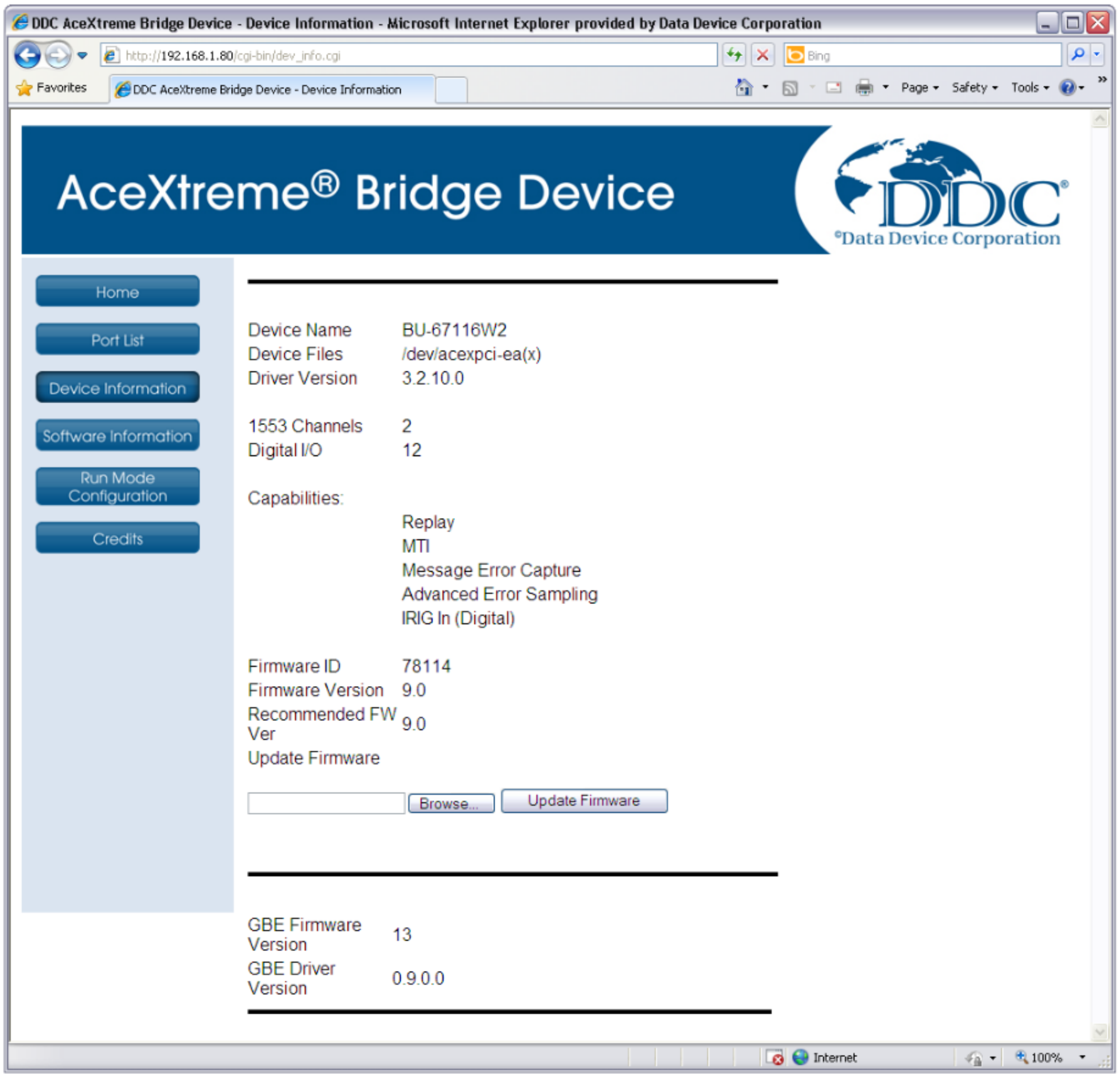
address	192.168.2.80
netmask	255.255.255.0
gateway	192.168.2.1

Figure 11. Changing IP Address through Web Server

There are two configurable ports on the ABD, the IP address, netmask and gateway are all configurable on the ABD. Make sure these settings are configured according to your network settings in order to access the device over your network. These settings can only be modified while the device is in configuration mode.

6.1.2 Device Information

The Device Information page on the AceXtreme Bridge Device displays the installed MIL-STD-1553 and ARINC-429 devices. The capabilities and driver information for these devices will also be listed on this page.

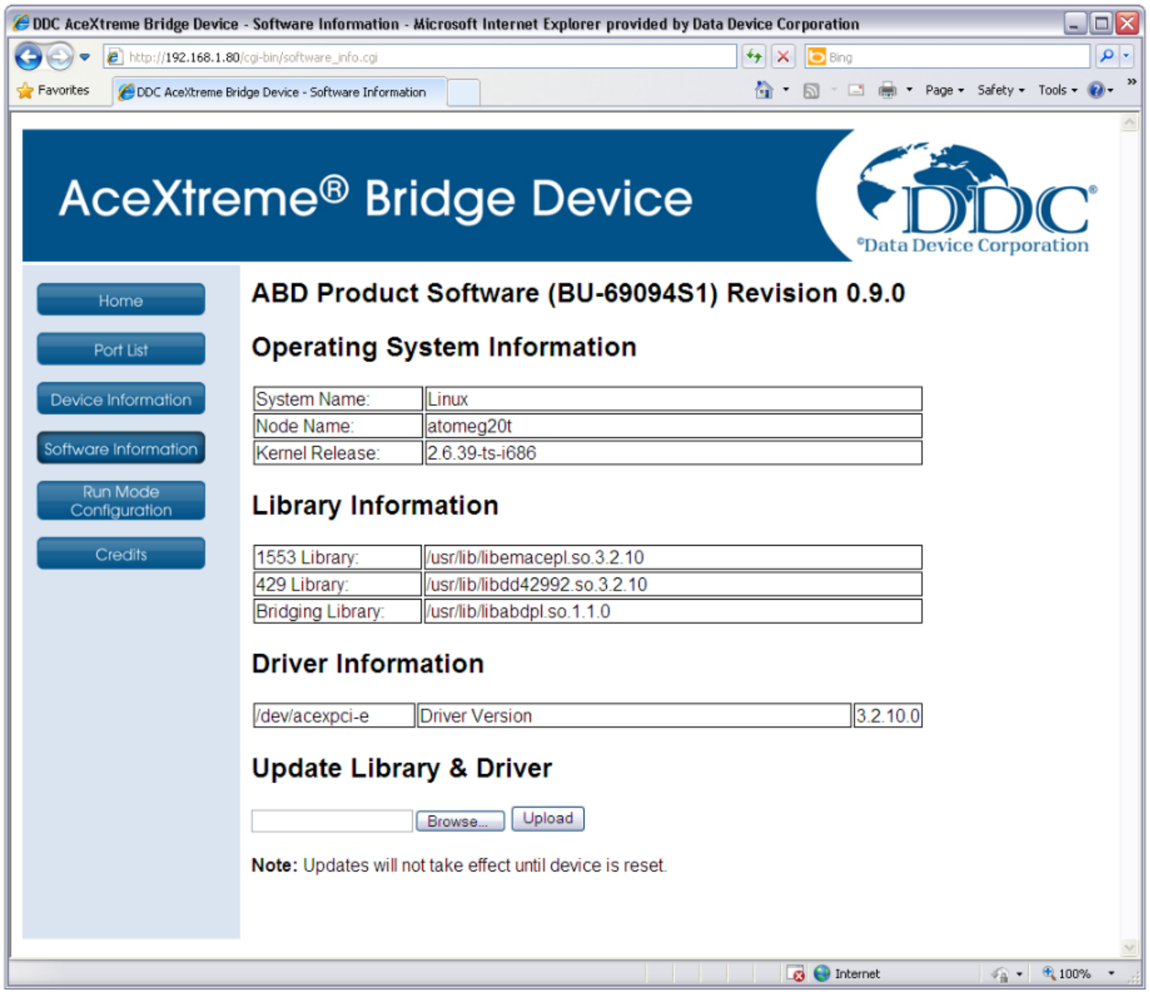

DDC AceXtreme Bridge Device - Device Information - Microsoft Internet Explorer provided by Data Device Corporation
http://192.168.1.80/cgi-bin/dev_info.cgi
AceXtreme® Bridge Device
DDC
Data Device Corporation
Home
Port List
Device Information
Software Information
Run Mode Configuration
Credits
Browse...
Update Firmware

Device Name	BU-67116W2
Device Files	/dev/acexpci-ea(x)
Driver Version	3.2.10.0
1553 Channels	2
Digital I/O	12
Capabilities:	
	Replay
	MTI
	Message Error Capture
	Advanced Error Sampling
	IRIG In (Digital)
Firmware ID	78114
Firmware Version	9.0
Recommended FW Ver	9.0
Update Firmware	
GBE Firmware Version	13
GBE Driver Version	0.9.0.0

Figure 12. ABD Device Information

6.1.3 Software Information

The Software Information page on the AceXtreme Bridge Device displays the installed MIL-STD-1553 and ARINC-429 SDK library versions. This page will also list the installed drivers for the DDC devices which utilize these libraries and their version. The Browse and Upload button can be used to upload a new library and / or driver provided by DDC.



ABD Product Software (BU-69094S1) Revision 0.9.0

Operating System Information

System Name:	Linux
Node Name:	atomeg20t
Kernel Release:	2.6.39-ts-i686

Library Information

1553 Library:	/usr/lib/libemacepl.so.3.2.10
429 Library:	/usr/lib/libdd42992.so.3.2.10
Bridging Library:	/usr/lib/libabdpl.so.1.1.0

Driver Information

/dev/acexpci-e	Driver Version	3.2.10.0

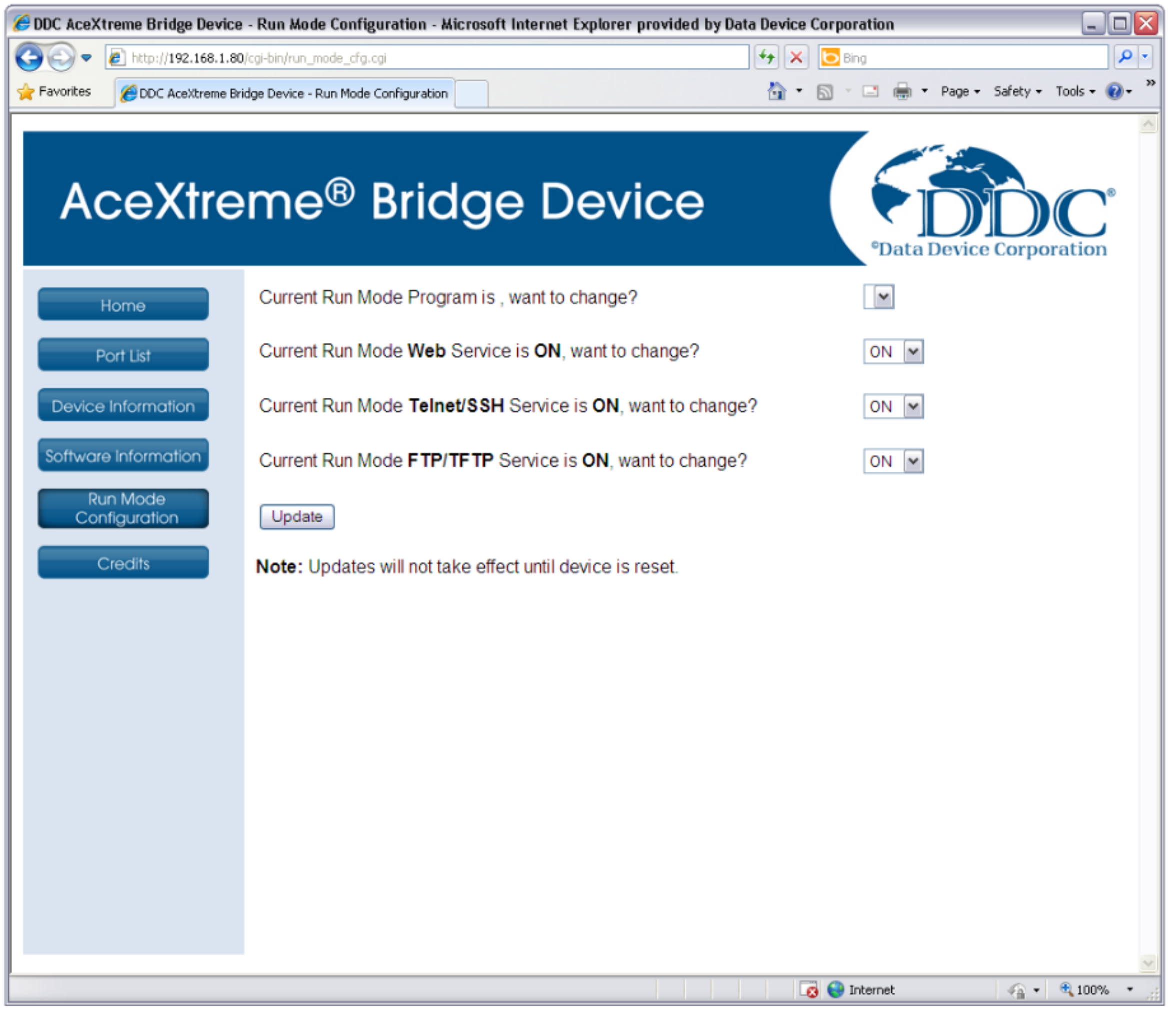
Update Library & Driver

Note: Updates will not take effect until device is reset.

Figure 13. ABD Software Information

6.1.4 Run Mode Configuration

The Run Mode Configuration page on the AceXtreme Bridge Device displays the current status of Run Mode Program and which services are on/off while the device is in 'Run Mode'.


DDC AceXtreme Bridge Device - Run Mode Configuration - Microsoft Internet Explorer provided by Data Device Corporation
http://192.168.1.80/cgi-bin/run_mode_cfg.cgi
Favorites
AceXtreme® Bridge Device
DDC
Data Device Corporation
Home
Port List
Device Information
Software Information
Run Mode Configuration
Credits
Update
Internet
100%

Current Run Mode Program is , want to change?

Current Run Mode **Web** Service is **ON**, want to change? ON

Current Run Mode **Telnet/SSH** Service is **ON**, want to change? ON

Current Run Mode **FTP/TFTP** Service is **ON**, want to change? ON

Note: Updates will not take effect until device is reset.

Figure 14. ABD Run Mode Configuration

The user can select which program to start when the device is powered on while in run mode. The Web Server, Telnet Server and FTP Server can also be turned On or Off.

6.2 Telnet/SSH Server

DDC's AceXtreme Bridge Device will ship from the factory with the Telnet/SSH Server enabled. The use of the Telnet server will allow the user to connect directly to the AceXtreme Bridge Device in order to interface with the Linux Operating System. In order to use the Telnet Server, the host system must have a Telnet client installed, and the service must be enabled on the host Operating System.

6.2.1 Enabling Telnet Windows 7

Windows 7 by default does not enable the Telnet service. To enable Telnet on a Windows 7 Operating System, the Telnet Windows Feature will have to be enabled. This is done by opening the Windows Control Panel and selecting Programs and Features.

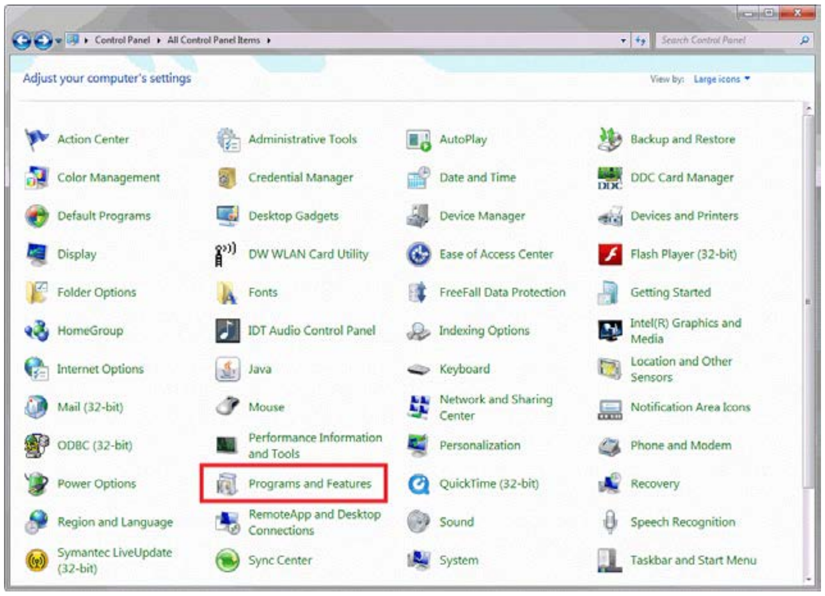
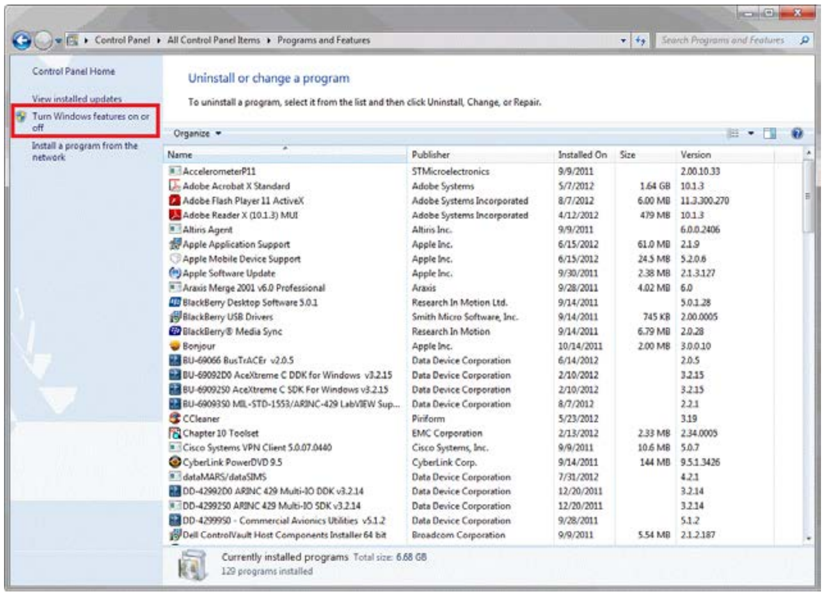


Figure 15. Programs and Features – Control Panel

Once the Programs and Features has been selected and opened, click on the link for Turn Windows Features on or off.


Control Panel ▸ All Control Panel Items ▸ Programs and Features
Search Programs and Features
Control Panel Home
View installed updates
Turn Windows features on or off
Install a program from the network
Organize

Uninstall or change a program

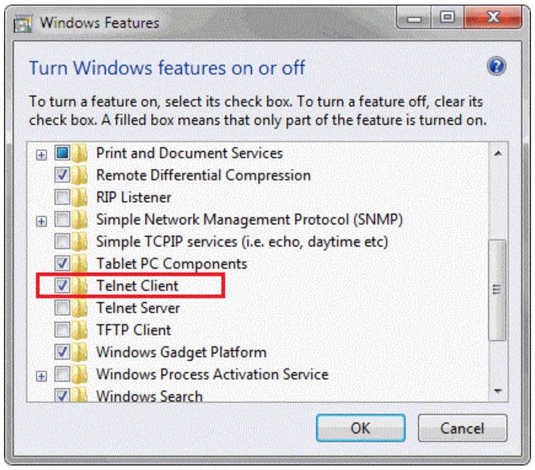
To uninstall a program, select it from the list and then click Uninstall, Change, or Repair.

Name	Publisher	Installed On	Size	Version
AccelerometerP11	STMicroelectronics	9/9/2011		2.00.10.33
Adobe Acrobat X Standard	Adobe Systems	5/7/2012	1.64 GB	10.1.3
Adobe Flash Player 11 ActiveX	Adobe Systems Incorporated	8/7/2012	6.00 MB	11.3.300.270
Adobe Reader X (10.1.3) MUI	Adobe Systems Incorporated	4/12/2012	479 MB	10.1.3
Altiris Agent	Altiris Inc.	9/9/2011		6.0.0.2406
Apple Application Support	Apple Inc.	6/15/2012	61.0 MB	2.1.9
Apple Mobile Device Support	Apple Inc.	6/15/2012	24.5 MB	5.2.0.6
Apple Software Update	Apple Inc.	9/30/2011	2.38 MB	2.1.3.127
Araxis Merge 2001 v6.0 Professional	Araxis	9/28/2011	4.02 MB	6.0
BlackBerry Desktop Software 5.0.1	Research In Motion Ltd.	9/14/2011		5.0.1.28
BlackBerry USB Drivers	Smith Micro Software, Inc.	9/14/2011	745 KB	2.00.0005
BlackBerry® Media Sync	Research In Motion	9/14/2011	6.79 MB	2.0.28
Bonjour	Apple Inc.	10/14/2011	2.00 MB	3.0.0.10
BU-69066 BusTrACEr v2.0.5	Data Device Corporation	6/14/2012		2.0.5
BU-69092D0 AceXtreme C DDK for Windows v3.2.15	Data Device Corporation	2/10/2012		3.2.15
BU-69092S0 AceXtreme C SDK For Windows v3.2.15	Data Device Corporation	2/10/2012		3.2.15
BU-69093S0 MIL-STD-1553/ARINC-429 LabVIEW Sup...	Data Device Corporation	8/7/2012		2.2.1
CCleaner	Piriform	5/23/2012		3.19
Chapter 10 Toolset	EMC Corporation	2/13/2012	2.33 MB	2.34.0005
Cisco Systems VPN Client 5.0.07.0440	Cisco Systems, Inc.	9/9/2011	10.6 MB	5.0.7
CyberLink PowerDVD 9.5	CyberLink Corp.	9/14/2011	144 MB	9.5.1.3426
dataMARS/dataSIMS	Data Device Corporation	7/31/2012		4.2.1
DD-42992D0 ARINC 429 Multi-IO DDK v3.2.14	Data Device Corporation	12/20/2011		3.2.14
DD-42992S0 ARINC 429 Multi-IO SDK v3.2.14	Data Device Corporation	12/20/2011		3.2.14
DD-42999S0 - Commercial Avionics Utilities v5.1.2	Data Device Corporation	9/28/2011		5.1.2
Dell ControlVault Host Components Installer 64 bit	Broadcom Corporation	9/9/2011	5.54 MB	2.1.2.187

Currently installed programs Total size: 6.68 GB
129 programs installed

Figure 16. Programs and Features – Control Panel

From the Windows Features window, scroll down the list of features while looking for Telnet Client.



Turn Windows features on or off

To turn a feature on, select its check box. To turn a feature off, clear its check box. A filled box means that only part of the feature is turned on.

- ■ Print and Document Services
- ☑ Remote Differential Compression
- ☐ RIP Listener
- ☐ Simple Network Management Protocol (SNMP)
- ☐ Simple TCPIP services (i.e. echo, daytime etc)
- ☑ Tablet PC Components
- ☑ Telnet Client
- ☐ Telnet Server
- ☐ TFTP Client
- ☑ Windows Gadget Platform
- ☐ Windows Process Activation Service
- ☑ Windows Search

Figure 17. Programs and Features – Control Panel

Once found, click the check box next to Telnet Client to enable the Windows feature, and click the OK button. Windows will then install the service, and the Windows Features Window will be closed. To confirm the feature has been installed open a command prompt and type telnet /?. The help feature for telnet will be displayed indicating it has been properly installed.

6.3 FTP/TFTP Server

DDC’s AceXtreme Bridge Device will ship from the factory with the FTP/TFTP Server enabled. The use of the FTP/TFTP server will allow the user to connect directly to the AceXtreme Bridge Device in order to transfer file on/off the ABD. In order to use the FTP/TFTP Server, the host system must have a FTP/TFTP client installed.

6.3.1 Connecting to ABD via FTP

To connect to the AceXtreme Bridge Device over FTP open your FTP client and enter in the IP address of the ABD. Once you have connected to the ADB with your FTP client, you will be prompted to enter the username and password of your account on the ABD. Please refer to section 5.1 for the default username and password on the AceXtreme Bridge Device.

7 USING THE PROTOCOL BRIDGING LAYER

The Protocol Bridge Layer (PBL) is a software middleware layer that allows the user to easily transfer data from one protocol to another.

It's supports user-defined bridging of MIL-STD-1553, ARINC-429 and Ethernet traffic in any combination and in any direction.

Bridging is achieved by developing an application using the PBL to define what specific data should be transferred. The user can define the source protocol/message and the destination protocol/message and the PBL will autonomously copy the data as defined.

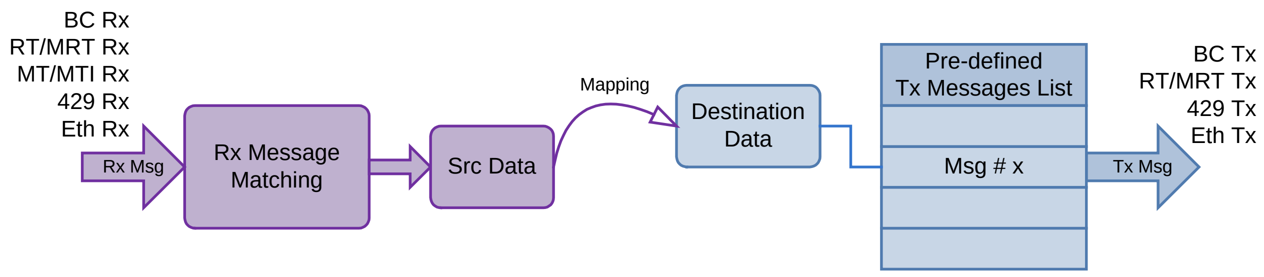


Figure 18. Protocol Bridging Layer Flow Diagram

The Protocol Bridging Layer is supplied as ANSI 'C' full source code and leverages DDCs MIL-STD-1553 and ARINC -429 Software Development Kits. This allows users too many any unique customizations to complete their desired Application.

7.1 Forwarding a port through Windows Firewall

To enable the use of the bridging samples you must first forward the 4DDC port number through your firewall.

First navigate to the **Control Pane**l and find **Windows Firewall**.

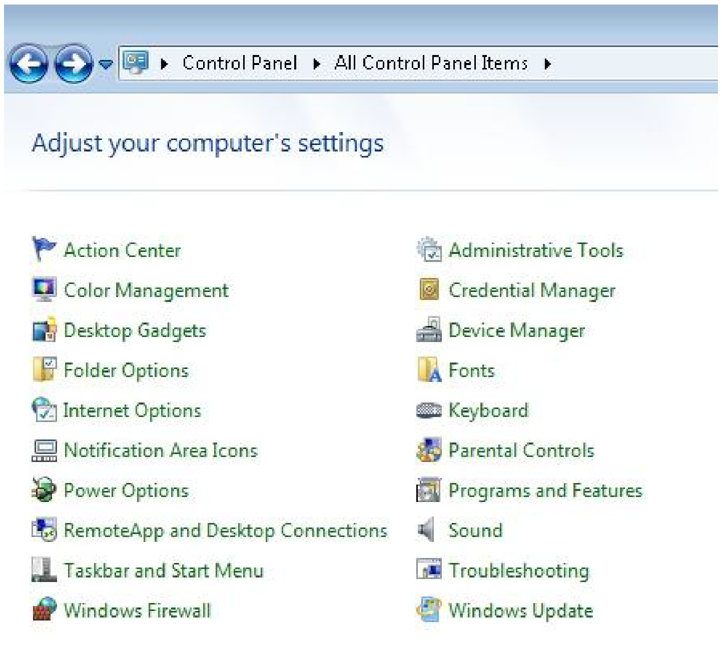
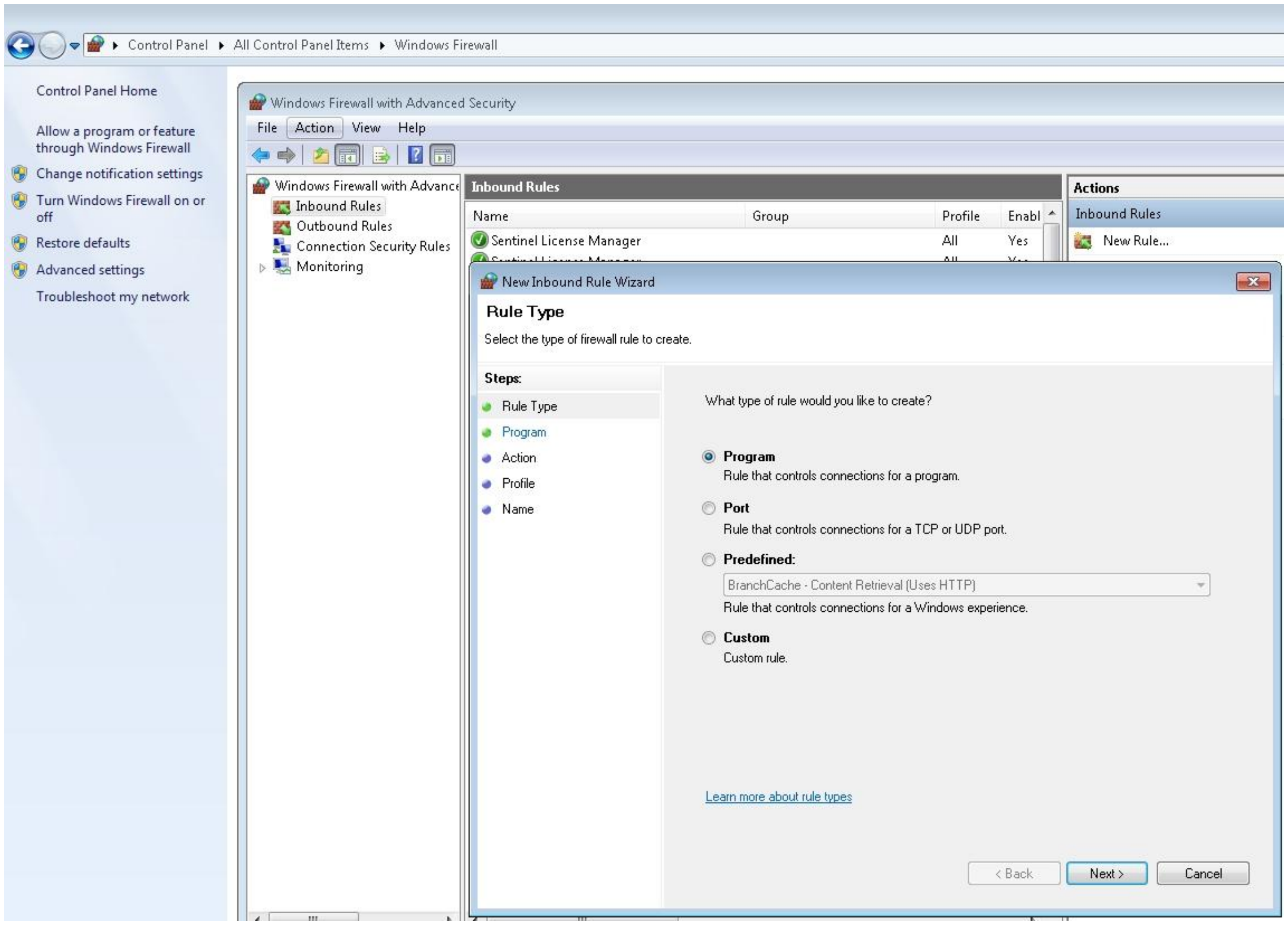


Figure 19. Windows Control Panel

1. Click **Advanced Settings** on the left hand side. Another window should pop up, **Windows Firewall with Advanced Security**.
2. Click **Inbound Rules** on the left side of the new window. Then Click **New Rule..** on the right hand side.
3. A **New Inbound Rule Wizard** should pop up.



Rule Type

Select the type of firewall rule to create.

What type of rule would you like to create?

[✓ Program] Rule that controls connections for a program.

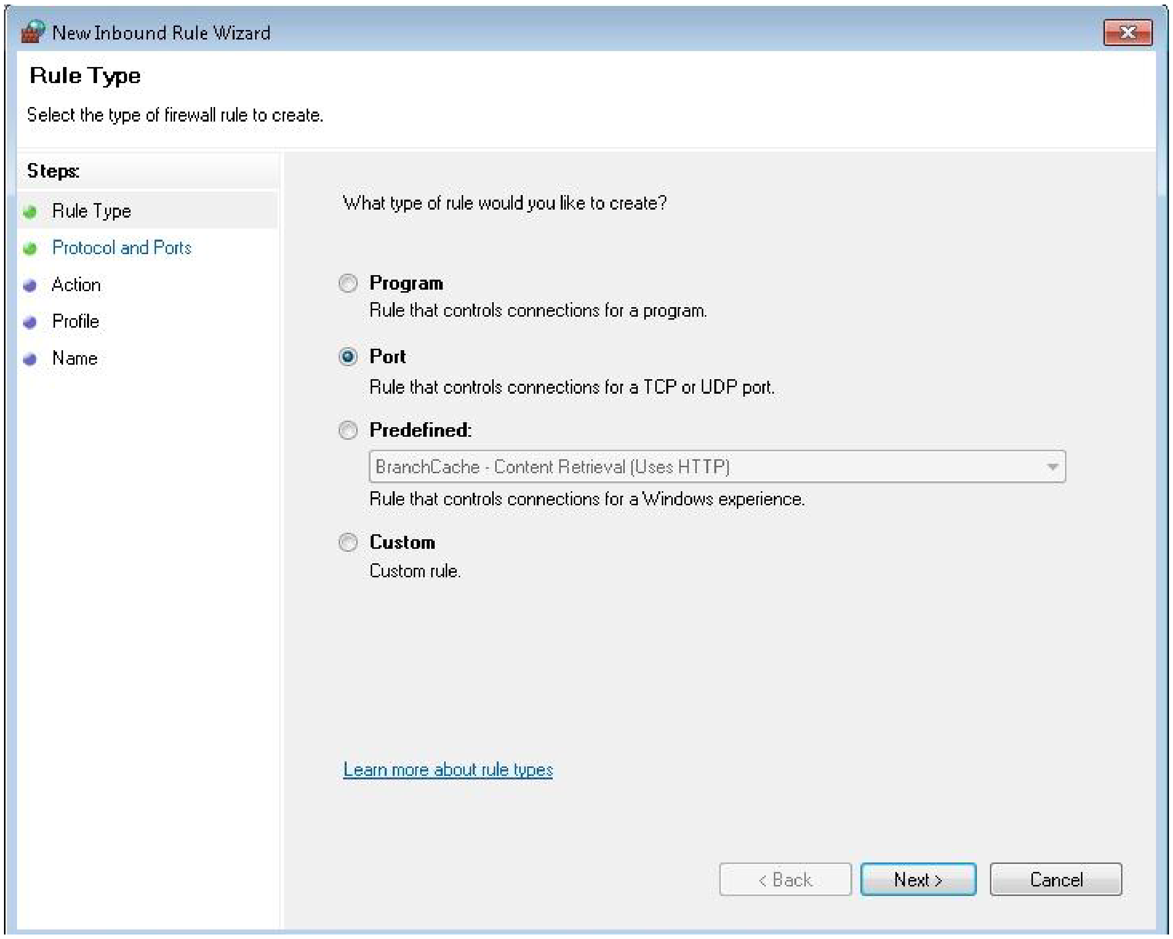
Port: Rule that controls connections for a TCP or UDP port.

Predefined: BranchCache - Content Retrieval (Uses HTTP). Rule that controls connections for a Windows experience.

Custom: Custom rule.

Figure 20. New Inbound Rule Window

4. In the Wizard, under the Rule Type Step select **Port**. Then Click **Next**.


New Inbound Rule Wizard

Rule Type

Select the type of firewall rule to create.

Steps:
- Rule Type
- Protocol and Ports
- Action
- Profile
- Name

What type of rule would you like to create?

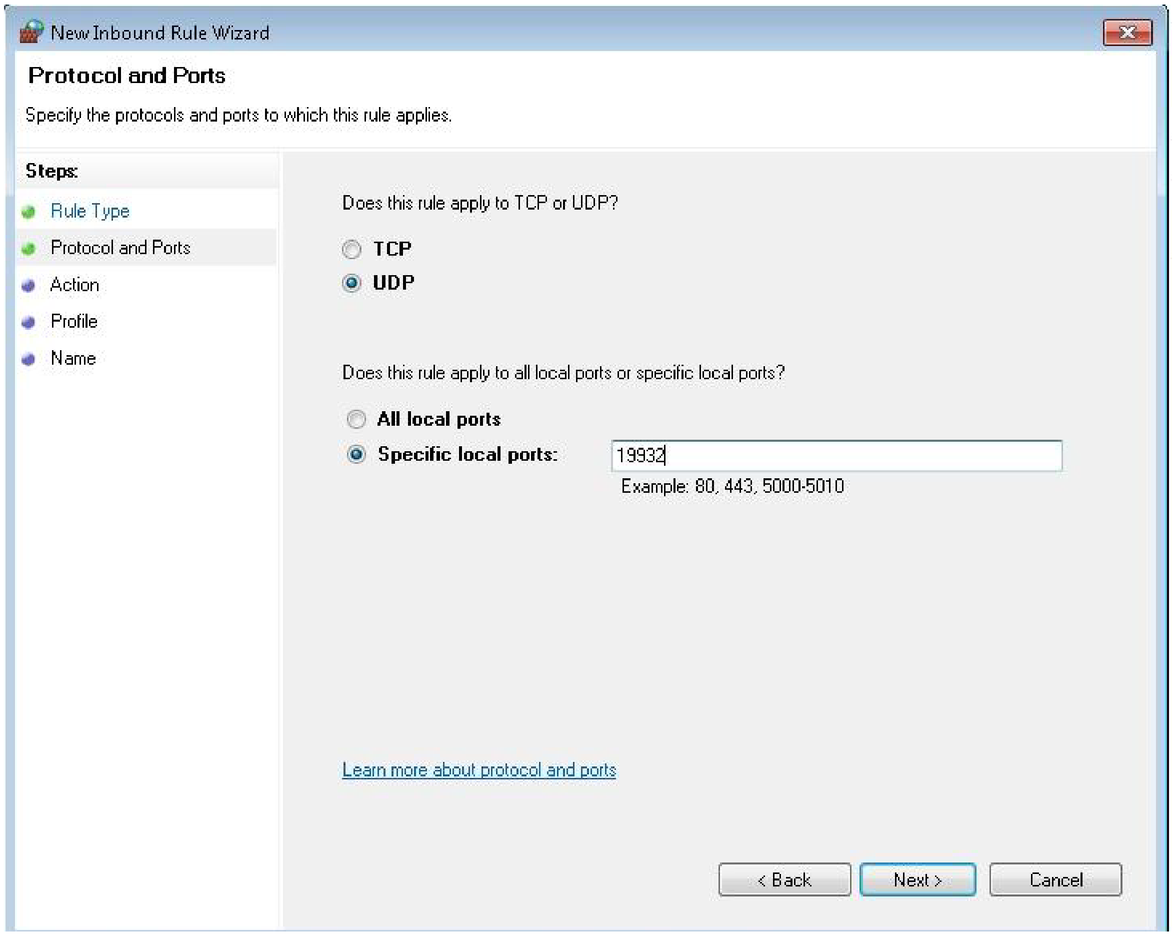
- ( ) **Program** Rule that controls connections for a program.
- (●) **Port** Rule that controls connections for a TCP or UDP port.
- ( ) **Predefined:** BranchCache - Content Retrieval (Uses HTTP) Rule that controls connections for a Windows experience.
- ( ) **Custom** Custom rule.

Learn more about rule types

< Back | Next > | Cancel

Figure 21. New Inbound Rule – Port

5. Under the Protocol and Ports step select **UDP.** Then under Specific local ports type in **19932**. This is the hex 4DDC port number converted to decimal. Click **Next**.


New Inbound Rule Wizard

Protocol and Ports

Specify the protocols and ports to which this rule applies.

Steps:
- Rule Type
- Protocol and Ports
- Action
- Profile
- Name

Does this rule apply to TCP or UDP?

TCP [✓ UDP]

Does this rule apply to all local ports or specific local ports?

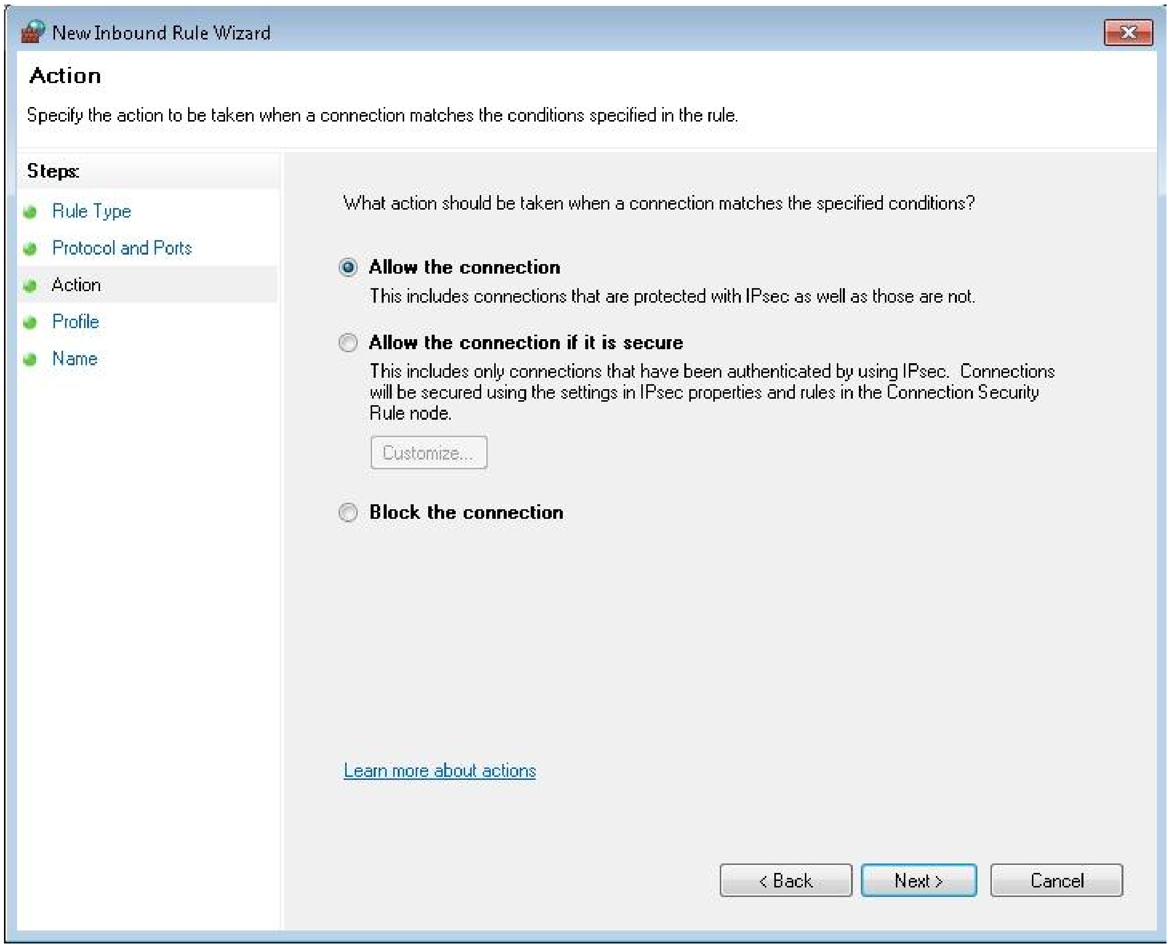
All local ports [✓ Specific local ports:] 19932

Example: 80, 443, 5000-5010

Learn more about protocol and ports

Figure 22. New Inbound Rule – Protocol and Ports

6. The next step is Action. The selection should be set to **Allow the connection**. Click **Next**.


New Inbound Rule Wizard

Action

Specify the action to be taken when a connection matches the conditions specified in the rule.

Steps:
- Rule Type
- Protocol and Ports
- Action
- Profile
- Name

What action should be taken when a connection matches the specified conditions?

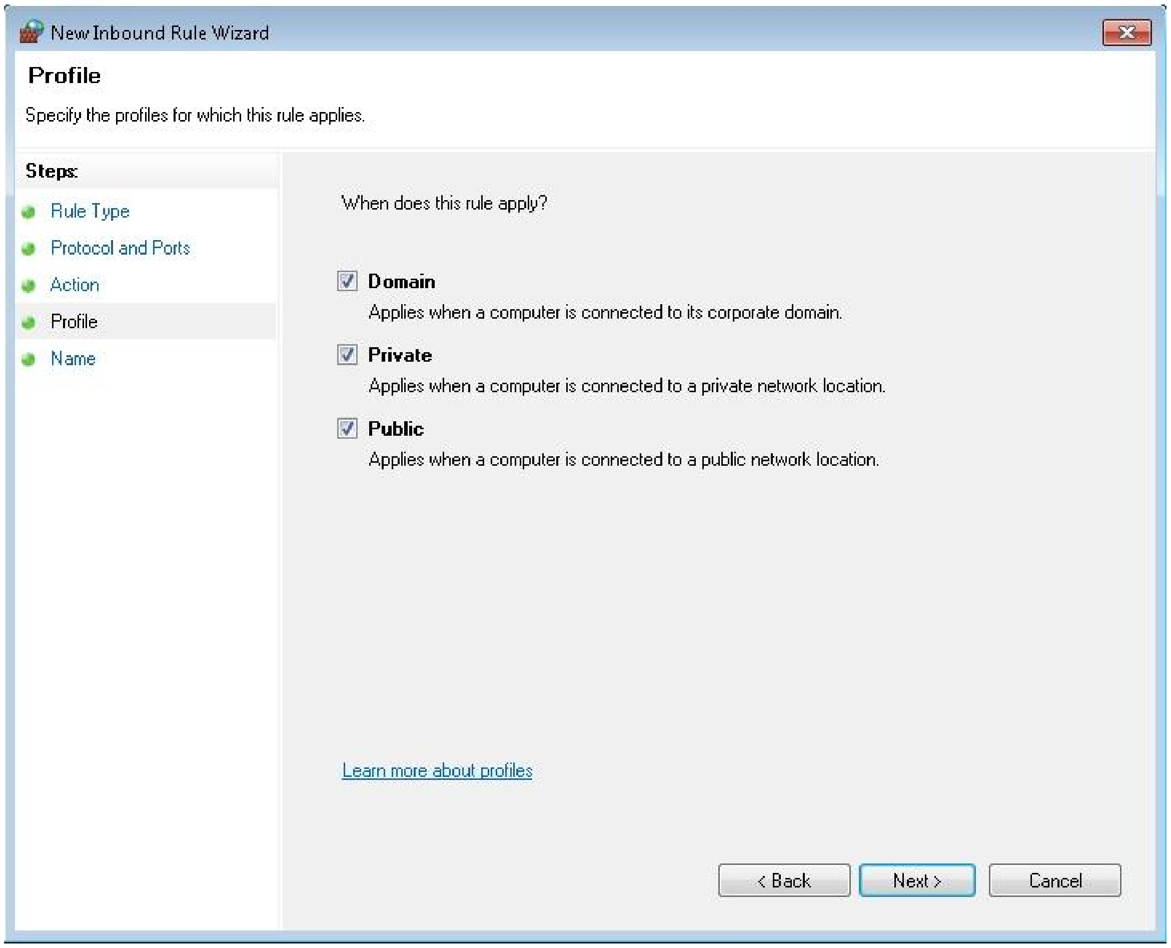
[✓ Allow the connection]
This includes connections that are protected with IPsec as well as those are not.

Allow the connection if it is secure
This includes only connections that have been authenticated by using IPsec. Connections will be secured using the settings in IPsec properties and rules in the Connection Security Rule node.

Block the connection

Figure 23. New Inbound Rule – Action

7. Under the profile step, all of the options should be checked off. Click **Next**.



Profile

Specify the profiles for which this rule applies.

When does this rule apply?

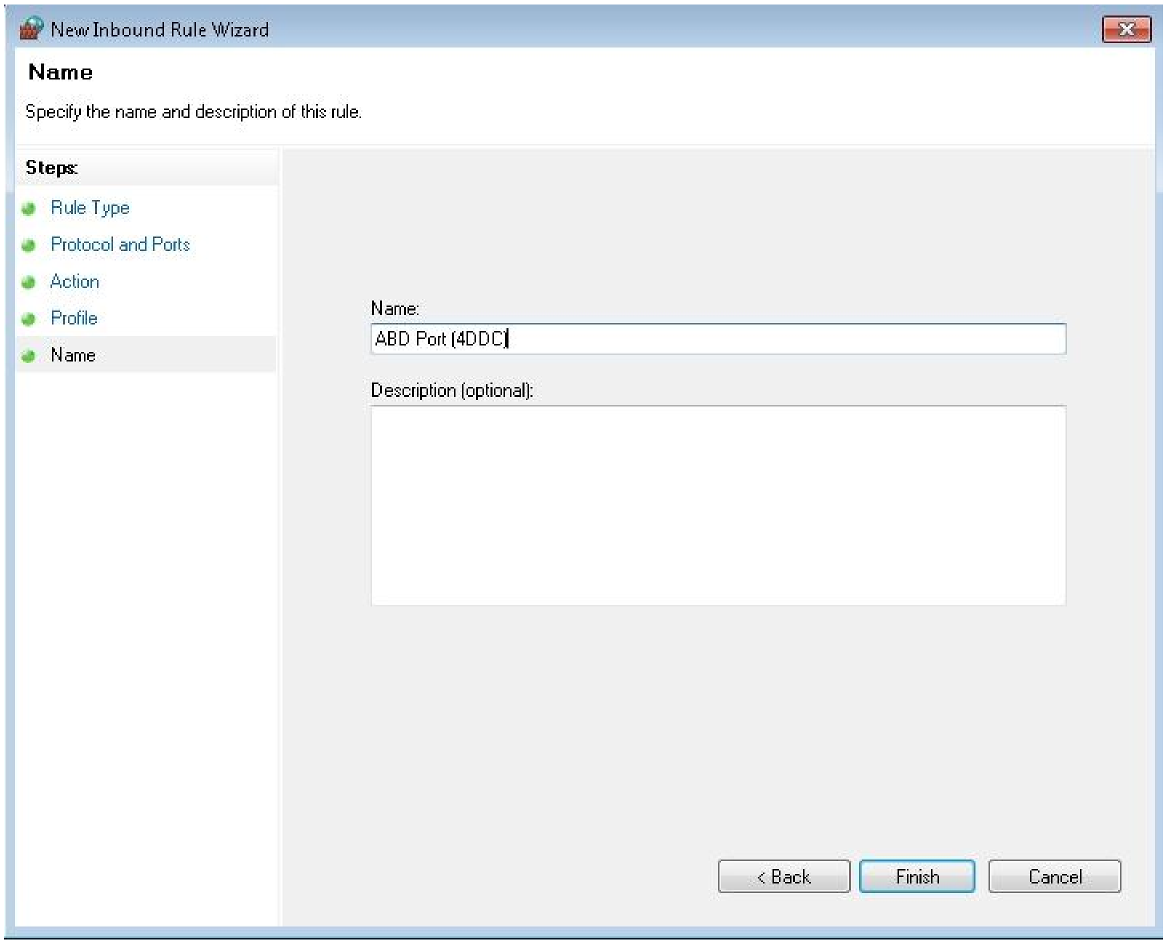
☑ **Domain**
Applies when a computer is connected to its corporate domain.

☑ **Private**
Applies when a computer is connected to a private network location.

☑ **Public**
Applies when a computer is connected to a public network location.

Figure 24. New Inbound Rule – Profile

8. In the **Name** field you can name it anyway you see fit. **ABD port (4DDC)** was chosen for this example. Description is optional. Click **Finish** after you are done.



Name

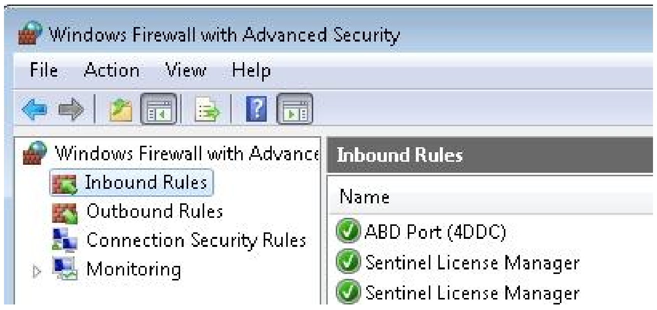
Specify the name and description of this rule.

Name: ABD Port (4DDC)

Description (optional):

Figure 25. New Inbound Rule – Port Name

After you click **Finish**, you should see your new Inbound Rule with a green check mark beside it. This confirms it was created and enabled successfully.


Windows Firewall with Advanced Security
File
Action
View
Help
Windows Firewall with Advance
Inbound Rules
Outbound Rules
Connection Security Rules
Monitoring

Inbound Rules

Name
✓ ABD Port (4DDC)
✓ Sentinel License Manager
✓ Sentinel License Manager

Figure 26. New Inbound Rule – Confirm Rule

7.2 Compiling Applications

All DDC AceXtreme Bridge Devices contain a complete target complier to compile and build any supplied samples or to compile any custom user applications.

Compiler Version: gcc version 4.5.3, (Timesys 20111107).

To build any sample, first navigate to the PBL Samples directory

cd /home/ddc/ABD/ABDbridging/samples

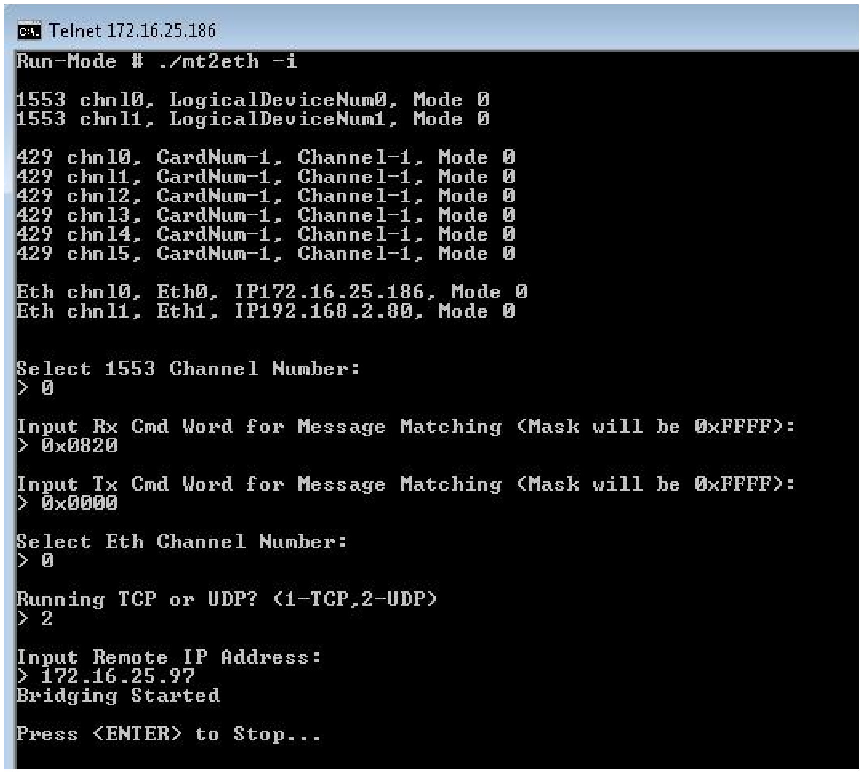
Then type 'make' followed by the sample to be rebuilt. See Example below.

make bc2eth

7.2.1 Using the Bridging Samples

In order to use the sample applications with the settings of your network, make sure the sample is run with the **-i** option.

This option allows a user to input the necessary information from the way their system is setup. This avoids having to manually edit the sample and recompile.



```
Run-Mode # ./mt2eth -i

1553 chnl0, LogicalDeviceNum0, Mode 0
1553 chnl1, LogicalDeviceNum1, Mode 0

429 chnl0, CardNum-1, Channel-1, Mode 0
429 chnl1, CardNum-1, Channel-1, Mode 0
429 chnl2, CardNum-1, Channel-1, Mode 0
429 chnl3, CardNum-1, Channel-1, Mode 0
429 chnl4, CardNum-1, Channel-1, Mode 0
429 chnl5, CardNum-1, Channel-1, Mode 0

Eth chnl0, Eth0, IP172.16.25.186, Mode 0
Eth chnl1, Eth1, IP192.168.2.80, Mode 0


Select 1553 Channel Number:
> 0

Input Rx Cmd Word for Message Matching (Mask will be 0xFFFF):
> 0x0820

Input Tx Cmd Word for Message Matching (Mask will be 0xFFFF):
> 0x0000

Select Eth Channel Number:
> 0

Running TCP or UDP? (1-TCP,2-UDP)
> 2

Input Remote IP Address:
> 172.16.25.97
Bridging Started

Press <ENTER> to Stop...
```

Figure 27. Bridge Sample

Select 1553 Channel Number – 1553 channel a user decides to use.

Input Rx Cmd Word for Message Matching – This is for the specific Receive command a user would want to forward. This example uses a BC -> RT1/SA1/WC32 command.

Input Tx Cmd Word for Message Matching – Specific Transmit commands will be forwarded. This sample does not forward any RT -> BC messages

Select Eth Channel Number – Ethernet channel the ABD is connected to on your network.

Running TCP or UDP? – Select which protocol a user will be using. This sample uses UDP.

Input Remote IP Address - This is the IP address of your computer.

7.3 Procotol Bridging API Dictionary

The following API functions and structures encompass the Protocol Bridging Layer. Fully-functional User Samples are also supplied to show expected usage (See Section 7.3).

Table 2. Protocol Bridging Layer (PBL) API Functions

Function Name	Category	Page
abd1553AddMappingRule	MIL-STD-1553	75
abd1553AddMatchingEntry	MIL-STD-1553	73
abd1553BcAddMsgToList	MIL-STD-1553	67
abd1553BcCreateMsg	MIL-STD-1553	65
abd1553EnableRT	MIL-STD-1553	61
abd1553EnableSA	MIL-STD-1553	63
abd1553Free	MIL-STD-1553	44
abd1553Initialize	MIL-STD-1553	43
abd1553Setup	MIL-STD-1553	49
abd1553Start	MIL-STD-1553	55
abd1553Stop	MIL-STD-1553	56
abd429AddMappingRule	ARINC-429	80
abd429AddMatchingEntry	ARINC-429	78
abd429AddTxMsg	ARINC-429	69
abd429Free	ARINC-429	46

Table 2. Protocol Bridging Layer (PBL) API Functions

Function Name	Category	Page
abd429Initialize	ARINC-429	45
abd429Setup	ARINC-429	51
abd429Start	ARINC-429	57
abd429Stop	ARINC-429	58
abdEthAddMappingRule	Ethernet	85
abdEthAddMatchingEntry	Ethernet	83
abdEthAddTxMsg	Ethernet	71
abdEthFree	Ethernet	48
abdEthInitialize	Ethernet	47
abdEthSetup	Ethernet	53
abdEthStart	Ethernet	59
abdEthStop	Ethernet	60
abdFree	All Modes	42
abdInitialize	All Modes	40

abdInitialize

This function will initialize all Protocols and Channels for use with Protocol Bridging.

PROTOTYPE

```
#include “abdDev.h”
S16BIT abdInitialize(void);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

RESET

MODE

Any

PARAMETERS

None

DESCRIPTION

This function will initialize all populated protocol (1553, 429, Ethernet) channels and place them in the SETUP state.

RETURN VALUE

Negative Value = Error
0 = Success

EXAMPLE

```
S16BIT wResult = 0;

/* Initialize All Channels */
if((wResult = abdInitialize())
{
    printf(“abdInitialize Failed.  Error Code: %d,wResult);
}
```

SEE ALSO

abd1553Initialize()	**abd429Initialize()**
abdEthInitialize()	

abdFree

This function will free (reset) all Protocols and Channels to an uninitialized state.

PROTOTYPE

```
#include “abdDev.h”
S16BIT abdFree(void);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

RESET, SETUP, RUN

MODE

Any

PARAMETERS

None

DESCRIPTION

This function will stop and reset all populated protocol (1553, 429, Ethernet) channels and return them in the RESET state.

RETURN VALUE

Negative Value = Error
0 = Success

EXAMPLE

```
S16BIT wResult = 0;

/* Free All Channels */
if((wResult = abdFree())
{
    printf(“abdFree Failed.  Error Code: %d,wResult);
}
```

SEE ALSO

abd1553Free() **abd429Free()**
abdEthFree()

abd1553Initialize

This function will initialize a MIL-STD-1553 channel to an uninitialized state..

PROTOTYPE

```
#include "setup1553.h"
S16BIT abd1553Free(U16BIT u16Abd1553Chnl);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

RESET, SETUP, RUN

MODE

Any

PARAMETERS

u16Abd1553Chnl	(input parameter) MIL-STD-1553 Channel Number Valid Values: 0, 1

DESCRIPTION

This function will initialize a specific MIL-STD-1553 channel for protocol bridging and place it in the SETUP state.

RETURN VALUE

Negative Value = Error
0 = Success

EXAMPLE

```
S16BIT wResult = 0;
U16BIT w1553Chnl = 0;

/* Initialize 1553 Channel */
if((wResult = abd1553Initialize(w1553Chnl))
{
    printf("abd1553Initialize      %d      Failed.      Error      Code:
%d,w1553Chnl,wResult);
}
```

SEE ALSO

abdInitialize() **abd429Initialize()**
abdEthInitialize()

abd1553Free

This function will reset a MIL-STD-1553 channel to an uninitialized state.

PROTOTYPE

```
#include "setup1553.h"
S16BIT abd1553Free(U16BIT u16Abd1553Chnl);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

RESET

MODE

Any

PARAMETERS

u16Abd1553Chnl	(input parameter) MIL-STD-1553 Channel Number Valid Values: 0, 1

DESCRIPTION

This function will stop and reset a specific MIL-STD-1553 channel and return it to the RESET state.

RETURN VALUE

Negative Value = Error
0 = Success

EXAMPLE

```
S16BIT wResult = 0;
U16BIT w1553Chnl = 0;

/* Free 1553 Channel */
if((wResult = abd1553Free(w1553Chnl))
{
    printf("abd1553Free %d Failed. Error Code: %d,w1553Chnl,wResult);
}
```

SEE ALSO

abdFree() **abd429Free()**
abdEthFree()

abd429Initialize

This function will initialize a ARINC 429 channel for use with Protocol Bridging.

PROTOTYPE

#include “setup429.h”
S16BIT abd429Initialize(U16BIT u16Abd429Chnl);

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

RESET

MODE

Any

PARAMETERS

u16Abd429Chnl	(input parameter) ARINC 429 Channel Number Valid Values: 0,1,2,3,4,5

DESCRIPTION

This function will initialize a specific ARINC 429 channel for protocol bridging and place it in the SETUP state.

RETURN VALUE

Negative Value = Error
0 = Success

EXAMPLE

```
S16BIT wResult = 0;
U16BIT w429Chnl = 0;

/* Initialize 429 Channel */
if((wResult = abd429Initialize(w429Chnl))
{
    printf(“abd429Initialize      %d      Failed.      Error      Code:
%d,w429Chnl,wResult);
}
```

SEE ALSO

abdInitialize()	**abd1553Initialize()**
abdEthInitialize()	

abd429Free

This function will reset an ARINC 429 channel to an uninitialized state.

PROTOTYPE

#include “setup429.h”
S16BIT abd429Free(U16BIT u16Abd429Chnl);

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

RESET, SETUP, RUN

MODE

Any

PARAMETERS

u16Abd429Chnl	(input parameter) ARINC 429 Channel Number Valid Values: 0,1,2,3,4,5

DESCRIPTION

This function will stop and reset a specific ARINC-429 channel and return it to the RESET state.

RETURN VALUE

Negative Value = Error
0 = Success

EXAMPLE

```
S16BIT wResult = 0;
U16BIT w429Chnl = 0;

/* Free 429 Channel */
if((wResult = abd429Free(w429Chnl))
{
    printf(“abd429Free %d Failed. Error Code: %d,w429Chnl,wResult);
}
```

SEE ALSO

abdFree() **abd1553Free()**
abdEthFree()

abdEthInitialize

This function will initialize a Ethernet channel for use with Protocol Bridging.

PROTOTYPE

```
#include “setupEth.h”
S16BIT abdEthInitialize(U16BIT u16Abd429Chnl);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

RESET

MODE

Any

PARAMETERS

u16AbdEthChnl	(input parameter) Ethernet Channel Number Valid Values: 0,1

DESCRIPTION

This function will initialize a specific Ethernet channel for protocol bridging and place it in the SETUP state.

RETURN VALUE

Negative Value = Error
0 = Success

EXAMPLE

```
S16BIT wResult = 0;
U16BIT wEthChnl = 0;

/* Initialize Ethernet Channel */
if((wResult = abdEthInitialize(wEthChnl))
{
     printf(“abdEthInitialize %d Failed. Error Code:
     %d,wEthChnl,wResult);
}
```

SEE ALSO

abdInitialize() **abd1553Initialize()**
abd429Initialize()

abdEthFree

This function will reset an Ethernet channel to an uninitialized state.

PROTOTYPE

#include “setupEth.h”
S16BIT abdEthFree(U16BIT u16Abd429Chnl);

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

RESET, SETUP, RUN

MODE

Any

PARAMETERS

u16AbdEthChnl	(input parameter) Ethernet Channel Number Valid Values: 0,1

DESCRIPTION

This function will stop and reset a specific Ethernet channel and return it to the RESET state.

RETURN VALUE

Negative Value = Error
0 = Success

EXAMPLE

```
S16BIT wResult = 0;
U16BIT wEthChnl = 0;

/* Free Ethernet Channel */
if((wResult = abdEthFree(wEthChnl))
{
    printf(“abdEthFree %d Failed. Error Code: %d,wEthChnl,wResult);
}
```

SEE ALSO

abdFree() **abd1553Free()**
abd429Free()

abd1553Setup

This function will setup a specific 1553 channel in a specific mode of operation.

PROTOTYPE

```
#include "setup1553.h"
S16BIT abd1553Setup(U16BIT u16Abd1553Chnl, U16BIT abd1553Mode);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

SETUP

MODE

Any

PARAMETERS

u16Abd1553Chnl	(input parameter) 1553 Channel Number Valid Values: 0,1
u16Abd1553Mode	(input parameter) 1553 Mode of Operation Valid Values:

ABD_1553_BC	Bus Controller Mode
ABD_1553_MRT	Multi-Remote Terminal Mode
ABD_1553_MT	Bus Monitor Mode

DESCRIPTION

This function will setup an initialized 1553 channel into one of three modes (Bus Controller, Multi-Remote Terminal, or Bus Monitor)

RETURN VALUE

Negative Value = Error
0 = Success

abd1553Setup (continued)

EXAMPLE

```
S16BIT wResult   = 0;
U16BIT w1553Chnl = 0;

/* Initialize 1553 channel in Bus Monitor Mode */
if((wResult = abd1553Setup(w1553Chnl, ABD_1553_MT))
{
    printf(“abd1553Setup %d Failed. Error Code: %d,w1553Chnl,wResult);
}
```

SEE ALSO

abd429Setup() **abdEthSetup()**

abd429Setup

This function will setup a specific ARINC 429 channel in a specific mode of operation.

PROTOTYPE

```
#include "setup429.h"
S16BIT abd429Setup(U16BIT u16Abd429Chnl, U16BIT abd429Mode,
                              S16BIT s16Abd429Speed, S16BIT s16Abd429Parity);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

SETUP

MODE

Any

PARAMETERS

u16Abd429Chnl	(input parameter) 1553 Channel Number Valid Values: 0,1,2,3,4,5	
u16Abd429Mode	(input parameter) 429 Data Direction Valid Values:	
	ABD_429_RX	ARINC 429 Receiver
	ABD_429_TX	ARINC 429 Transmitter
s16Abd429Speed	(input parameter) 429 Data Speed Valid Values:	
	DD429_LOW_SPEED	Low-Speed 429
	DD429_HIGH_SPEED	High-Speed 429
u16Abd429Parity	(input parameter) 429 Parity Selection Valid Values:	
	DD429_NO_PARITY	No Parity
	DD429_ODD_PARITY	Odd Parity
	DD429_EVEN_PARITY	EvenParity

DESCRIPTION

This function will setup the ARINC 429 direction, speed and parity for an initialized 429 channel.

RETURN VALUE

Negative Value = Error
0 = Success

abd429Setup (continued)

EXAMPLE

```
S16BIT wResult   = 0;
U16BIT w429Chnl = 0;

/* Initialize 429 channel as a receiver */
if((wResult = abd429Setup(w429Chnl, ABD_429_RX,
    DD429_HIGH_SPEED, DD429_ODD_PARITY))
{
    printf("abd429Setup %d Failed. Error Code: %d,w429Chnl,wResult);
}
```

SEE ALSO

abd1553Setup () **abdEthSetup()**

abdEthSetup

This function will setup a specific Ethernet channel in a specific mode of operation.

PROTOTYPE

```
#include “setupEth.h”
S16BIT abdEthSetup(U16BIT u16AbdEthChnl, U16BIT abdEthMode, U16BIT u16PortNum);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

SETUP

MODE

Any

PARAMETERS

u16Abd429Chnl	(input parameter) 1553 Channel Number Valid Values: 0,1	
u16AbdEthMode	(input parameter) IP Protocol Selection Valid Values:	
	ABD_ETH_TCP	Use TCP/IP Protocol
	ABD_ETH_UDP	Use UDP/IP Protocol
u16PortNum	(input parameter) TCP/UDP Port (Socket) Number Valid Values: 1-65535	

DESCRIPTION

This function will setup the Ethernet Upper IP Protocol and the Socket Port Number. Note, please make sure to select a port number that does not conflict with existing network services.

RETURN VALUE

Negative Value = Error
0 = Success

abdEthSetup (continued)

EXAMPLE

```
S16BIT wResult   = 0;
U16BIT wEthChnl = 0;
U16BIT wEthPort = 5000;


/* Initialize Ethernet channel for UDP/IP on Port 5000 */
if((wResult = abdEthSetup(w429Chnl, ABD_ETH_UDP, wEthPort));
{
    printf("abdEthSetup %d Failed. Error Code: %d,wEthChnl,wResult);
}
```

SEE ALSO

abd1553Setup () **abd429Setup()**

abd1553Start

This function will start operation of a setup MIL-STD-1553 channel.

PROTOTYPE

```
#include "setup1553.h"
S16BIT abd155Start(U16BIT u16Abd1553Chnl);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

SETUP

MODE

Any

PARAMETERS

u16Abd1553Chnl	(input parameter) MIL-STD-1553 Channel Number Valid Values: 0, 1

DESCRIPTION

This function will start a specific MIL-STD-1553 channel and make it active (transmit and receive) on the 1553 bus.

RETURN VALUE

Negative Value = Error
0 = Success

EXAMPLE

```
S16BIT wResult = 0;
U16BIT w1553Chnl = 0;

/* Start 1553 Channel */
if((wResult = abd1553Start(w1553Chnl))
{
    printf("abd1553Start %d Failed. Error Code: %d,w1553Chnl,wResult);
}
```

SEE ALSO

abd1553Stop() **abd1553Setup ()**
abd1553Initialize()

abd1553Stop

This function will stop operation of a running MIL-STD-1553 channel.

PROTOTYPE

```
#include “setup1553.h”
S16BIT abd155Stop(U16BIT u16Abd1553Chnl);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

RUN

MODE

Any

PARAMETERS

u16Abd1553Chnl	(input parameter) MIL-STD-1553 Channel Number Valid Values: 0, 1

DESCRIPTION

This function will stop a specific MIL-STD-1553 channel from being active (transmit and receive) on the 1553 bus.

RETURN VALUE

Negative Value = Error
0 = Success

EXAMPLE

```
S16BIT wResult = 0;
U16BIT w1553Chnl = 0;

/* Stop 1553 Channel */
if((wResult = abd1553Stop(w1553Chnl))
{
    printf(“abd1553Stop %d Failed. Error Code: %d,w1553Chnl,wResult);
}
```

SEE ALSO

abd1553Start() **abd1553Setup ()**
abd1553Initialize()

abd429Start

This function will start a setup ARINC 429 channel.

PROTOTYPE

```
#include “setup429.h”
S16BIT abd429Start(U16BIT u16Abd429Chnl);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

SETUP

MODE

Any

PARAMETERS

u16Abd429Chnl	(input parameter) ARINC 429 Channel Number Valid Values: 0,1,2,3,4,5

DESCRIPTION

This function will start a specific ARINC 429 channel and allow it to transmit or receive.

RETURN VALUE

Negative Value = Error
0 = Success

EXAMPLE

```
S16BIT wResult = 0;
U16BIT w429Chnl = 0;

/* Start 429 Channel */
if((wResult = abd429Start(w429Chnl))
{
    printf(“abd429Start %d Failed. Error Code: %d,w429Chnl,wResult);
}
```

SEE ALSO

abd429Initialize() **abd429Stop()**
abd429Setup()

abd429Stop

This function will stop a running ARINC 429 channel.

PROTOTYPE

#include “setup429.h”
S16BIT abd429Stopt(U16BIT u16Abd429Chnl);

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

RUN

MODE

Any

PARAMETERS

u16Abd429Chnl	(input parameter) ARINC 429 Channel Number Valid Values: 0,1,2,3,4,5

DESCRIPTION

This function will stop a specific ARINC 429 channel from running (transmit and received).

RETURN VALUE

Negative Value = Error
0 = Success

EXAMPLE

```
S16BIT wResult = 0;
U16BIT w429Chnl = 0;

/* Stop 429 Channel */
if((wResult = abd429Stop(w429Chnl))
{
    printf(“abd429Stop %d Failed. Error Code: %d,w429Chnl,wResult);
}
```

SEE ALSO

abd429Initialize() **abd429Start()**
abd429Setup()

abdEthStart

This function will start a setup Ethernet channel.

PROTOTYPE

#include “setupEth.h”
S16BIT abdEthStart(U16BIT u16AbdEthChnl);

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

SETUP

MODE

Any

PARAMETERS

u16AbdEthChnl	(input parameter) Ethernet Channel Number Valid Values: 0,1

DESCRIPTION

This function will start a specific Ethernet channel and allow it to transmit or receive.

RETURN VALUE

Negative Value = Error
0 = Success

EXAMPLE

```
S16BIT wResult = 0;
U16BIT wEthChnl = 0;

/* Start Ethernet Channel */
if((wResult = abdEthStart(wEthChnl))
{
    printf(“abdEthStart %d Failed. Error Code: %d,wEthChnl,wResult);
}
```

SEE ALSO

abdEthInitialize() **abdEthStop()**
abdEthSetup()

abdEthStop

This function will stop a running Ethernet channel.

PROTOTYPE

#include “setupEth.h”
S16BIT abdEthStopt(U16BIT u16AbdEthChnl);

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

RUN

MODE

Any

PARAMETERS

u16AbdEthChnl	(input parameter) Ethernet Channel Number Valid Values: 0,1,

DESCRIPTION

This function will stop a specific Ethernet channel from running (transmit and received).

RETURN VALUE

Negative Value = Error
0 = Success

EXAMPLE

```
S16BIT wResult = 0;
U16BIT wEthChnl = 0;

/* Stop Ethernet Channel */
if((wResult = abdEthStop(wEthChnl))
{
    printf(“abdEthStop %d Failed. Error Code: %d,wEthChnl,wResult);
}
```

SEE ALSO

abdEthInitialize()	**abdEthStart()**
abdEthSetup()	

abd1553EnableRT

This function will enable an specific RT Address to transmit and/or receive.

PROTOTYPE

```
#include “setup1553.h”
S16BIT abd1553EnableRT(U16BIT u16Abd1553Chnl, S8BIT s8RtAddr);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

SETUP

MODE

1553 RT

PARAMETERS

u16Abd1553Chnl	(input parameter) MIL-STD-1553 Channel Number Valid Values: 0, 1
s8RtAddr	(input parameter) RT Address to Enable Valid Values:0-31

DESCRIPTION

This function will enable a specific MIL-STD-1553 RT Address to be active on the specified 1553 channel. Note, that this function does not enable RT Subaddresses, only the RT Address.

RETURN VALUE

Negative Value = Error
0 = Success

EXAMPLE

```
S16BIT wResult = 0;
U16BIT w1553Chnl = 0;

/* Enable RT #01 */
if((wResult = abd1553EnableRT(w1553Chnl, 0x01))
{
    printf(“abd1553EnableRT      %d      Failed.      Error      Code:
            %d,w1553Chnl,wResult);
}
```

abd1553EnableRT (continued)

SEE ALSO

abd1553EnableSA()	**abd1553Start()**
abdEthInitialize()	

abd1553EnableSA

This function will enable an RT Subaddress within a setup and enabled RT Address.

PROTOTYPE

```
#include “setup1553.h”
S16BIT abd1553EnableSA(U16BIT u16Abd1553Chnl, S8BIT s8RtAddr, U16BIT u16SA,
                       U16BIT* pTxDataBuf, U16BIT u16TxDataBufLen);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

SETUP

MODE

1553 RT

PARAMETERS

u16Abd1553Chnl	(input parameter) MIL-STD-1553 Channel Number Valid Values: 0, 1
s8RtAddr	(input parameter) RT Address Reference Valid Values:0-31
u16SA	(input parameter) RT Subaddress to Enable Valid Values:0-31
pTxDataBuf	(input parameter) Pointer to initial Data for RT Transmit Commands
u16TxDataBufLen	(input parameter) Size (in Words) of pTxDataBuf Valid Values:0-32

DESCRIPTION

This function will enable a specific MIL-STD-1553 RT Subaddress (within the specified RT Address) to be active on the assigned 1553 channel. Note that this the RT Address must first be enabled by calling **abd1553EnableRT()**.

RETURN VALUE

Negative Value = Error
0 = Success

abd1553EnableSA (continued)

EXAMPLE

```
S16BIT wResult = 0;
U16BIT w1553Chnl = 0;
U16BIT u16Data[64] =
{
    0x1111, 0x2222, 0x3333, 0x4444, 0x1111, 0x2222, 0x3333, 0x4444,
    0x1111, 0x2222, 0x3333, 0x4444, 0x1111, 0x2222, 0x3333, 0x4444,
    0x1111, 0x2222, 0x3333, 0x4444, 0x1111, 0x2222, 0x3333, 0x4444,
    0x1111, 0x2222, 0x3333, 0x4444, 0x1111, 0x2222, 0x3333, 0x4444
};

/* Enable RT #01 SA #02*/
if((wResult = abd1553EnableSA(w1553Chnl, 0x01, 0x02, u16Data, 32))
{
    printf(“abd1553EnableSA      %d      Failed.      Error      Code:
            %d,w1553Chnl,wResult);
}
```

SEE ALSO

abd1553EnableRT() **abd1553Start()**
abd1553Stop()

abd1553BcCreateMsg

This function will create a 1553 Message to be sent by a Bus Controller.

PROTOTYPE

```
#include "setup1553.h"
S16BIT abd1553BcCreateMsg(U16BIT u16Abd1553Chnl, S16BIT s16MsgID,
                          ABD_1553_BC_MSG_CREATE_INFO sMsgCreateInfo,
                          U16BIT *pDataBuf, U16BIT u16DataBufSize);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

SETUP

MODE

1553 BC

PARAMETERS

u16Abd1553Chnl	(input parameter) MIL-STD-1553 Channel Number Valid Values: 0, 1
s16MsgID	(input parameter) User-Assigned Message ID 1-65535
sMsgCreateInfo	(input parameter) 1553 Message Information
pDataBuf	(input parameter) Pointer to initial Data for BC-RT Commands
u16TxDataBufLen	(input parameter) Size (in Words) of pTxDataBuf Valid Values:0-32

DESCRIPTION

This function will create a 1553 Bus Controller Message to be transmitted by the Bus Controller. The user has the option of addling this message to the synchronous 1553 BC Bus List or asynchronously sending it whenever new bridged data is received.

RETURN VALUE

Negative Value = Error
0 = Success

abd1553BcCreateMsg (continued)

EXAMPLE

```
S16BIT wResult = 0;
U16BIT w1553Chnl = 0;
ABD_1553_BC_MSG_CREATE_INFO sMsgCreateInfo;
U16BIT u16Data[32] =
{
    0x1111, 0x2222, 0x3333, 0x4444, 0x1111, 0x2222, 0x3333, 0x4444,
    0x1111, 0x2222, 0x3333, 0x4444, 0x1111, 0x2222, 0x3333, 0x4444,
};

/* BC->RT1/SA1 msg with 16 data words */
sMsgCreateInfo.msgType = BC_TO_RT; /* BC->RT Message */
sMsgCreateInfo.bSched = TRUE;      /* Will be added to the Bus List. */
sMsgCreateInfo.u16RTRx = 1;        /* RT 01 */
sMsgCreateInfo.u16SARx = 1;        /* SA 01 */
sMsgCreateInfo.u16WC   = 16;       /* 16 Data Words */
sMsgCreateInfo.u16MsgGapTime = 0;  /* No additional Gap Time */
sMsgCreateInfo.u32MsgOptions  =  ACE_BCCTRL_CHL_A;  /*  Send  on  Channel  A
*/

/* Create New 1553 BC Message */
if((wResult = abd1553BcCreateMsg(w1553Chnl,sMsgCreateInfo,u16Data,16)))
{
    printf(“abd1553BcCreateMsg     %d     Failed.     Error     Code:
            %d,w1553Chnl,wResult);
}
```

SEE ALSO

abd1553BcAddMsgToList() **abd1553Start()**
ABD_1553_BC_MSG_CREATE_INFO

abd1553BcAddMsgToList

This function will add an existing 1553 Message to the synchronous BC Bus List.

PROTOTYPE

#include “setup1553.h”
S16BIT abd1553BcAddMsgToList (U16BIT u16Abd1553Chnl, S16BIT s16MsgID);

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

SETUP

MODE

1553 BC

PARAMETERS

u16Abd1553Chnl	(input parameter) MIL-STD-1553 Channel Number Valid Values: 0, 1
s16MsgID	(input parameter) User-Assigned Message ID 1-65535

DESCRIPTION

This function will create a 1553 Bus Controller Message to be transmitted by the Bus Controller. The user has the option of addling this message to the synchronous 1553 BC Bus List or asynchronously sending it whenever new bridged data is received.

RETURN VALUE

Negative Value = Error
0 = Success

EXAMPLE

```
S16BIT wResult = 0;
U16BIT w1553Chnl = 0;

/* Add Message ID 01 to the BC Bust List */
if((wResult = abd1553BcAddMsgToList(w1553Chnl,0x01))
{
    printf(“abd1553BcAddMsgToList            %d            Failed.Error
Code:%d,w1553Chnl,wResult);
}
```

abd1553BcAddMsgToList (continued)

SEE ALSO

abd1553BcCreateMsg() **abd1553Start()**
abd1553Stop()

abd429AddTxMsg

This function will add a 429 Message to the Schedule or FIFO Pending Queue

PROTOTYPE

```
#include "setup429.h"
S16BIT abd429AddTxMsg(U16BIT u16Abd429Chnl, U16BIT u16MsgId, U32BIT u32InitMsgData,
                                BOOLEAN bSched, S16BIT s16Frequency, S16BIT s16Offset);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

SETUP, RUN

MODE

ARINC 429

PARAMETERS

u16Abd429Chnl	(input parameter) ARINC 429 Channel Number Valid Values: 0,1,2,3,4,5
s16MsgID	(input parameter) User-Assigned Message ID 1-65535
u32InitMsgData	(input parameter) Initial ARINC 429 Message (32-bit value format)
bSched	(input parameter) TRUE = Schedule the Message at a defined periodic Rate FALSE = Send the message when new data is available (FIFO)
s16Frequency	(input parameter) Only applies to Periodic Messages. (bSched = TRUE) Defines the frequency of the transmission in milliseconds Valid Values:1-32767
s16Offset	(input parameter) Only applies to Periodic Messages. (bSched = TRUE) The offset in milliseconds relative to the other scheduled words. Must be less than the “s16Frequency’ input parameter. Valid Values:1-32767

abd429AddTxMsg (continued)

DESCRIPTION

This function schedules a transmission of an ARINC-429 Message If ‘bSched” is set to TRUE, the Message will transmit at the specified periodic rate. If ‘bSched” is FALSE, the message will only be sent when new bridge data linked to this message is received (FIFO). Note, the 429 Channel Must be started (using **abd429Start()**) for transmissions to begin.

RETURN VALUE

Negative Value = Error
0 = Success

EXAMPLE

```
S16BIT wResult = 0;
U16BIT w429Chnl = 0;
U32BIT u32Data = 0x12345678; /* ARINC SDI/Label/Data Message */

/* Setup Tx Msg.(ID=0x01) Scheduled, freq 1000ms, Offset 0ms */
if((wResult = abd429AddTxMsg(w429Chnl, 0x01, u32Data, TRUE, 1000, 0))
{
    printf(“abd429AddTxMsg %d Failed. Error Code: %d,w429Chnl,wResult);
}
```

SEE ALSO

abd429Setup() **abd429Start()**
abd429Stop()

abd4EthAddTxMsg

This function will add a Ethernet Message to the FIFO Pending Queue.

PROTOTYPE

```
#include “setupEth.h”
S16BIT abdEthAddTxMsg(U16BIT u16AbdEthChnl, U16BIT u16MsgId, char * remoteIP,
                      S16BIT s16DestPort, U32BIT u32MsgLength);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

SETUP, RUN

MODE

ETHERNET

PARAMETERS

u16AbdEtheChnl	(input parameter) Ethernet Channel Number Valid Values: 0,1
s16MsgID	(input parameter) User-Assigned Message ID 1-65535
remoteIP	(input parameter) Remote IP Address to send to (String Format) Example: “192.168.1.1”
s16DestPort	(input parameter) Only applies to UDP/IP configured Channels. Enter -1 for TCP/IP configured Channels. Remote Port to send to. Valid Values: 1-65535
u32MsgLength	(input parameter) Size of the message (in bytes) Valid Values:1-65535

abd4EthAddTxMsg (continued)

DESCRIPTION

This function schedules transmission of an Ethernet Message The Message will only be sent with new bridge data linked to this message becomes available. Note, the Ethernet Channel must be started (using **abdEthStart()**) for transmissions to begin.

RETURN VALUE

Negative Value = Error
0 = Success

EXAMPLE

```
S16BIT wResult = 0;
U16BIT wEthChnl = 0;

/* Setup UDP Msg. (ID=0x01) to “192.168.1.1”, Port 5000 */
if((wResult = abdEthAddTxMsg(wEthChnl,0x01,”192.168.1.1”,5000,1024))
{
    printf(“abdEthAddTxMsg %d Failed. Error Code: %d,w429Chnl,wResult);
}
```

SEE ALSO

abdEthSetup()	**abdEthStart()**
abdEthStop()	

abd1553AddMatchingEntry

This function will create a Matching Entry for received/monitored MIL-STD-1553 traffic.

PROTOTYPE

```
#include “setup1553.h”
S16BIT abd1553AddMatchingEntry(U16BIT u16Abd1553Chnl, U16BIT u16MatchingId,
                               U16BIT u16RxCmdWrd, U16BIT u16RxCmdWrdMask,
                               U16BIT u16TxCmdWrd, U16BIT u16TxCmdWrdMask);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

SETUP

MODE

1553 ANY

PARAMETERS

u16Abd1553Chnl	(input parameter) MIL-STD-1553 Channel Number Valid Values: 0, 1
u16MatchingId	(input parameter) User Assigned Matching ID. Valid Values:1-65535
u16RxCmdWrd	(input parameter) RT Receive Command Word to Match. Valid Values:0x0000-0xFFFF
u16RxCmdWrdMask	(input parameter) RT Receive Command Word Mask Value. Valid Values:0x0000-0xFFFF
u16TxCmdWrd	(input parameter) RT Transmit Command Word to Match. Valid Values:0x0000-0xFFFF
u16TxCmdWrdMask	(input parameter) RT Transmit Command Word Mask Value. Valid Values:0x0000-0xFFFF

abd1553AddMatchingEntry (continued)

DESCRIPTION

This function will create a Matching Entry for received/monitored 1553 traffic. If any monitored 1553 message matches the Logical AND of the either the Receive Command Word/Mask or the Transmit Command Word/Mask, then the data will be made available to bridge to a destination channel using the Protocol “Mapping Rule” functions.

Note, 1553 data cannot be bridged to any other channel before defining a Matching Entry.

RETURN VALUE

Negative Value = Error
0 = Success

EXAMPLE

```
S16BIT wResult = 0;
U16BIT w1553Chnl = 0;

/* Create a Matching Entry for BC->RT01/SA01/WC32 (CmdWord=0x0820) */
if((wResult                                                          =
abd1553AddMatchingEntry(w1553Chnl,0x01,0x0820,0xFFFF,0,0xFFFF))
{
    printf(“abd1553AddMatchingEntry Failed.  Error Code: wResult);
}
```

SEE ALSO

abd1553EnableRT() **abd1553Start()**
abd429AddMappingRule()

abd1553AddMappingRule

This function will create a bridge from a 1553 Matching Entry to the desired destination location.

PROTOTYPE

```
#include "setup1553.h"
S16BIT abd1553AddMappingRule(U16BIT u16Abd1553Chnl, U16BIT u16MatchingId,
                             U8BIT u8SrcStartingWrdIndex, U8BIT u8SrcStartingBit,
                             U16BIT u16SrcLength,
                             PABD_MAPPING_DEST_INFO pMappingDestInfo);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

SETUP

MODE

1553 ANY

PARAMETERS

u16Abd1553Chnl	(input parameter) MIL-STD-1553 Channel Number Valid Values: 0, 1
u16MatchingId	(input parameter) Matching ID of the desired Matching Entry Valid Values:1-65535
u8SrcStartingWrdIndex	(input parameter) Starting 1553 Word Location to bridge (from Matching Entry) Valid Values:0-31
u8SrcStartingBit	(input parameter) Starting Bit location (from Starting Word) to bridge Valid Values:0-15
u16SrcLength	(input parameter) Length of data (in bits) to bridge (Starting from Word/Bit above) Valid Values:1-704
pMappingDestInfo	(input parameter) Desired Destination Message to bridge data into.

abd1553AddMappingRule (continued)

DESCRIPTION

This function will bridge (copy) data from a defined 1553 Matching Entry to a desired destination message. If the referenced 1553 matching entry message has been updated, the protocol bridging layer will autonomously copy it into the destination message.

Note, this function does not affect the scheduling of the target destination message. It is up to the user to either schedule the destination message in a bus list or to assign it to be sent asynchronously.

The MIL-STD-1553 Message format is as follows. Users shall reference this format when assigning which words/bits to bridge 1553 Source or Destination data.

Table 3. ABD_1553_UNIFIED MSG_STRUCTURE

Struct Member	Word Location	Description
u16IntraPktTimeStamp	0-3	64-bit Timestamp of last reception.
u16BlkStats	4	1553 Block Status Word
u16GapTime	5	1553 Message Gap Time
u16Length	6	Length of Message (in bytes)
u16RxCmdWrd	7	1553 RT Receive Command Word
u16RxStatusWrd	8	1553 RT Receive Status Word
u16TxCmdWrd	9	1553 RT Transmit Command Word
u16TxStatusWrd	10	1553 RT Transmit Status Word
u16WrdCnt	11	1553 Word Count (Data Payload)
u16TRforModeCode	12-44	0-31 Data Words (Payload)

RETURN VALUE

Negative Value = Error
0 = Success

abd1553AddMappingRule (continued)

EXAMPLE

```
S16BIT wResult = 0;
U16BIT w1553Chnl = 0;
ABD_MAPPING_DEST_INFO sDestInfo;

/* Map 1553 Data to Ethernet */
sDestInfo.u8MappingDestType = 2;                          /* Destination =
Ethernet */
sDestInfo.u.sEthMappingDest.u16ChnlNum = 0;       /* Ethernet Channel 0
*/
sDestInfo.u.sEthMappingDest.u16MsgId = 1;         /* Ethernet Message ID
1 */
sDestInfo.u.sEthMappingDest.u16StartingByte = 0; /* Ethernet Start Byte
0 */
sDestInfo.u.sEthMappingDest.u16StartingBit = 0;  /* Ethernet Start Bit
0 */

/* Bridge 32 (16 bits*32) words from 1553 Msg ID #1 to an Ethernet
Message */
if((wResult=abd1553AddMappingRule(w1553Chnl,1,12,0,16*32,&sDestInfo))
{
    printf(“abd1553AddMappingRule Failed.  Error Code: wResult);
}
```

SEE ALSO

abd429AddMappingRule()	**ABD_MAPPING_DEST_INFO**
abd1553Start()	

abd429AddMatchingEntry

This function will create a Matching Entry for received ARINC 429 traffic.

PROTOTYPE

```
#include "setup429.h"
S16BIT abd429AddMatchingEntry(U16BIT u16Abd429Chnl, U16BIT u16MatchingId,
                              U16BIT u16LabelSDI, U16BIT u16LabelSDIMatchingMask)
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

SETUP

MODE

ARINC 429 RX

PARAMETERS

u16Abd429Chnl	(input parameter) ARINC-429 Channel Number Valid Values: 0,1,2,3,4,5
u16MatchingId	(input parameter) User Assigned Matching ID. Valid Values:1-65535
u16LabelSDI	(input parameter) ARINC 429 Label/SDI Word to Match (A429 Word Bits 0-9) Valid Values:0x0000-0xFFFF
u16LabelSDIMatchingMask	(input parameter) ARINC 429 Label/SDI Word Mask Value. Valid Values:0x0000-0x03FF

DESCRIPTION

This function will create a Matching Entry for received ARINC 429 traffic. If any received ARINC 429 message matches the Logical AND of the Label/SDI Word and Mask, then the data will be made available to bridge to a destination channel using the Protocol "Mapping Rule" functions.

Note, ARINC 429 data cannot be bridged to any other channel before defining a Matching Entry.

RETURN VALUE

Negative Value = Error
0 = Success

abd429AddMatchingEntry (continued)

EXAMPLE

```
S16BIT wResult = 0;
U16BIT w429Chnl = 0;
U16BIT wLabelSDI = 0x0102; /* SDI = 0x01B, Label = 0x02 */

/* Create a Matching Entry (ID=0x01) for SDI=0x01B, Label=0x02 */
if((wResult = abd429AddMatchingEntry(w429Chnl,0x01,wLabelSDI,0x03FF))
{
    printf(“abd429AddMatchingEntry Failed.  Error Code: wResult);
}
```

SEE ALSO

abd429Setup() **abd429Start()**
abd429AddMappingRule()

abd429AddMappingRule

This function will create a bridge from a ARINC 429Matching Entry to the desired destination.

PROTOTYPE

```
#include “setup429.h”
S16BIT abd429AddMappingRule(U16BIT u16Abd429Chnl, U16BIT u16MatchingId,
                            U8BIT u8SrcStartingBit, U8BIT u8StartingDWord
                            U16BIT u16SrcLength,
                            PABD_MAPPING_DEST_INFO pMappingDestInfo);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

SETUP

MODE

ARINC 429 RX

PARAMETERS

u16Abd429Chnl	(input parameter) ARINC 429 Channel Number Valid Values: 0,1,2,3,4,5
u16MatchingId	(input parameter) Matching ID of the desired Matching Entry Valid Values:1-65535
u8SrcStartingDWord	(input parameter) Starting DWord (32-bits) within ARINC Message to bridge Valid Values:0-2
u8SrcStartingBit	(input parameter) Starting Bit location (from above DWord) to bridge Valid Values:0-31
u16SrcLength	(input parameter) Length of data (in bits) to bridge (Starting from Bit above) Valid Values:1-96
pMappingDestInfo	(input parameter) Desired Destination Message to bridge data into.

abd429AddMappingRule (continued)

DESCRIPTION

This function will bridge (copy) data from a defined ARINC 429 Matching Entry to a desired destination message. If the referenced ARINC 429 message has been updated, the protocol bridging layer will autonomously copy it into the destination message.

Note, this function does not affect the scheduling of the target destination message. It is up to the user to either schedule the destination message in a bus list or to assign it to be sent asynchronously.

The ARINC 429 Message format is as follows. Users shall reference this format when assigning which DWords/bits to bridge ARIINC 429 Source or Destination data.

Table 4. ARINC_429_MSG_STRUCTURE

Variable Name	DWord Location	Bit Location	Description
Label	0	0-7	Message Label
SDI	0	8-9	Source/Destination Identifier (SDI)
Data	0	10-28	Data Payload
SSM	0	29-30	Sign/Status Matrix
P	0	31	Parity Bit
Time Stamp High	1	0-31	Upper 32-bits of 64-bit TimeStamp
Time Stamp Low	2	0-31	Lower 32-bits of 64-bit TimeStamp

RETURN VALUE

Negative Value = Error
0 = Success

abd429AddMappingRule (continued)

EXAMPLE

```
S16BIT wResult = 0;
U16BIT w429Chnl = 0;
ABD_MAPPING_DEST_INFO sDestInfo;

/* Map 1553 Data to Ethernet */
sDestInfo.u8MappingDestType = 2;                          /* Destination =
Ethernet */
sDestInfo.u.sEthMappingDest.u16ChnlNum = 0;       /* Ethernet Channel 0
*/
sDestInfo.u.sEthMappingDest.u16MsgId = 1;         /* Ethernet Message ID
#1 */
sDestInfo.u.sEthMappingDest.u16StartingByte = 0; /* Ethernet Start Byte
0 */
sDestInfo.u.sEthMappingDest.u16StartingBit = 0;   /* Ethernet Start Bit
0 */

/* Bridge 32 bits (all) from 429 Msg ID #1 to an Ethernet Message */
if((wResult=abd429AddMappingRule(w429Chnl,1,0,0,32,&sDestInfo))
{
    printf("abd429AddMappingRule Failed.  Error Code: wResult);
}
```

SEE ALSO

abd429AddMappingRule()	**ABD_MAPPING_DEST_INFO**
abd429Start()	

abdEthAddMatchingEntry

This function will create a Matching Entry for received Ethernet traffic.

PROTOTYPE

```
#include "setupEth.h"
S16BIT abdEthAddMatchingEntry(U16BIT u16AbdEthChnl, U16BIT u16MatchingId,
                              char * remoteIpaddr, char * addrMask);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

SETUP

MODE

ETHERNET

PARAMETERS

u16AbdEthChnl	(input parameter) Ethernet Channel Number Valid Values: 0,1
u16MatchingId	(input parameter) User Assigned Matching ID. Valid Values:1-65535
RemoteIpaddr	(input parameter) IPv4 IP Address to Match (String Format)
addrMask	(input parameter) IPv4 IP Address Word Mask Value.

DESCRIPTION

This function will create a Matching Entry for received Ethernet traffic. If any received Ethernet message matches the Logical AND of the Remote IP Address Value and Mask, then the data will be made available to bridge to a destination channel using the Protocol "Mapping Rule" functions.

Note, Ethernet data cannot be bridged to any other channel before defining a Matching Entry.

RETURN VALUE

Negative Value = Error
0 = Success

abdEthAddMatchingEntry (continued)

EXAMPLE

```
S16BIT wResult = 0;
U16BIT wEthChnl = 0;

/* Create a Matching Entry for IP 192.168.1.2 */
if((wResult =

abdEthAddMatchingEntry(wEthChnl,0x01,”192.168.1.2”,”255.255.255.255”))
{
    printf(“abdEthAddMatchingEntry Failed.  Error Code: wResult);
}
```

SEE ALSO

abdEthSetup()	**abdEthStart()**
abdEthAddMatchingEntry()	

abdEthAddMappingRule

This function will create a bridge from an Ethernet Matching Entry to the desired destination.

PROTOTYPE

```
#include "setupEth.h"
S16BIT abdEthAddMappingRule(U16BIT u16AbdEthChnl, U16BIT u16MatchingId,
                            U32BIT u32SrcStartingByte, U32BIT u32SrcStartingBit,
                            U32BIT u32SrcLength,
                            PABD_MAPPING_DEST_INFO pMappingDestInfo);
```

HARDWARE

BU-67115, BU-67116, BU-67119

STATE

SETUP

MODE

ETHERNET

PARAMETERS

u16AbdEthChnl	(input parameter) Ethernet Channel Number Valid Values: 0,1
u16MatchingId	(input parameter) Matching ID of the desired Matching Entry Valid Values:1-65535
u32SrcStartingByte	(input parameter) Starting Byte location to bridge Valid Values:0-65535
u32SrcStartingBit	(input parameter) Starting Bit location (from above Byte) to bridge Valid Values:0-7
u32SrcLength	(input parameter) Length of data (in bits) to bridge (Starting from Byte/Bit above) Valid Values:1-524288
pMappingDestInfo	(input parameter) Desired Destination Message to bridge data into.

abdEthAddMappingRule (continued)

DESCRIPTION

This function will bridge (copy) data from a defined Ethernet Matching Entry to a desired destination message. If the referenced Ethernet packet has been updated, the protocol bridging layer will autonomously copy it into the destination message.

Note, this function does not affect the scheduling of the target destination message. It is up to the user to either schedule the destination message in a bus list or to assign it to be sent asynchronously.

The Ethernet Message format is as follows. Users shall reference this format when assigning which bits to bridge Ethernet Source or Destination data.

Table 5. ETHERNET_MSG_STRUCTURE

Variable Name	Byte Location	Description
Length	0-3	Length of TCP Message (in bytes)
Data	4-65535	User-Defined Data Payload

RETURN VALUE

Negative Value = Error
0 = Success

EXAMPLE

```
S16BIT wResult = 0;
U16BIT wEthChnl = 0;
ABD_MAPPING_DEST_INFO sDestInfo;

/* Map Ethernet Data to 1553 Remote Terminal(RT 01, SA01 Word/Bit 0/0*/
sDestInfo.u8MappingDestType = 0;                /* Destination = 1553 */
sDestInfo.u.s1553MappingDest.u16ChnlNum = 0;   /* 1553 Channel 0 */
sDestInfo.u.s1553MappingDest.s16MsgId = -1;    /* N/A for 1553 RT */
sDestInfo.u.s1553MappingDest.s8RtAddr = 1;     /* 1553 RT Address 01 */
sDestInfo.u.s1553MappingDest.s8SA     = 1;     /* 1553 SA 01 */
sDestInfo.u.s1553MappingDest.u16StartingDataWord = 0;/* 1553 DataWord 0
*/
sDestInfo.u.s1553MappingDest.u16StartingBit = 0;  /* 1553 Bit 0 */

/* Bridge 16 Words from Ethernet to 1553 RT 01 SA 01, Words 0-15 */
if((wResult=abdEthAddMappingRule(wEthChnl,1,4,0,16*16,&sDestInfo))
{
    printf("abdEthAddMappingRule Failed.  Error Code: wResult);
}
```

SEE ALSO

abdEthAddMatchingEntry() **ABD_MAPPING_DEST_INFO**
abdEthStart()

7.4 Structures

ABD_1553_BC_MSG_CREATE_INFO

Structure used in creating MIL-STD-1553 Bus Controller Messages..

PROTOTYPE

```
#include "setup1553.h"
typedef struct _ABD_1553_BC_MSG_CREATE_INFO
{
        ABD_1553_BC_MSG_TYPE msgType;
        BOOLEAN bSched;
        U16BIT u16RTRx;
        U16BIT u16SARx;
        U16BIT u16RTTx;
        U16BIT u16SATx;
        U16BIT u16WC;
        U16BIT u16MsgGapTime;
        U32BIT u32MsgOptions;
        U16BIT u16TRforModeCode;
        U16BIT u16ModeCmd;
        U16BIT u16MajorFrmTime;
} ABD_1553_BC_MSG_CREATE_INFO, *PABD_1553_BC_MSG_CREATE_INFO;
```

Table 6. ABD_1553_BC_MSG_CREATE_INFO Structure

Struct Member	Description
msgType	Type of Message. Valid Choice are: BC_TO_RT, RT_TO_BC, RT_TO_RT, MODE, BROADCAST, BROADCAST_RT_TO_RT, BROADCAST_MODE, END_OF_MINOR, END_OF_MAJOR.
bSched	TRUE = Message will be added to the sched. Bus List via abd1553BcAddMsgToList FALSE = Message will be send asynchronously when new data is available.
u16RTRx	RT Address for Receive RT.
u16SARx	RT Subaddress for Receive RT.
u16RTTx	RT Address for Transmit RT.
u16SATx	RT Subaddress for Transmit RT.
u16WC	Word Count or Mode Code
u16MsgGapTime	Gap Time (in us) to next message (Only applied to sched. Messages).
u32MsgOptions	1553 Message Options.
u16TRforModeCode	Transmit (1) / Receive (0) for Mode Code commands
u16ModeCmd	1553 Mode Command.
u16MajorFrmTime	Major Frame Time (in 100us increments). Only applies to END_OF_MAJOR messages.

ABD_1553_BC_MSG_CREATE_INFO (continued)

SEE ALSO

abd1553BcCreateMsg()	**abd1553Start()**
abd1553BcAddMsgToList()	

ABD_MAPPING_DEST_INFO (*PABD_MAPPING_DEST_INFO)

Structure (union) to define protocol bridging destination location.

PROTOTYPE

```
#include “abdDev.h”
typedef struct _ABD_MAPPING_DEST_INFO
{
         U8BIT u8MappingDestType;

         union ABD_MAPPING_DEST_INFO_UNION
         {
             ABD_1553_MAPPING_DEST_INFO s1553MappingDest;
             ABD_429_MAPPING_DEST_INFO  s429MappingDest;
             ABD_ETH_MAPPING_DEST_INFO  sEthMappingDest;
         } u;
}ABD_MAPPING_DEST_INFO, *PABD_MAPPING_DEST_INFO;
```

Table 7. ABD_MAPPING_DEST_INFO Structure

Struct Member	Description
U8MappingDestType	Destination Protocol: 1 = MIL-STD-1553 2 = ARINC 429 3 = ETHERNET
u.s1553MappingDest	MIL-STD-1553 Bridging Destination Information
u.s429MappingDest	ARINC-429 Bridging Destination Information
u.sEthMappingDest	Ethernet Bridging Destination Information

SEE ALSO

abd1553AddMappingRule() **abd429AddMappingRule()**
abdEthAddMappingRule()

ABD_1553_MAPPING_DEST_INFO

Structure to define MIL-STD-1553 protocol bridging destination location.

PROTOTYPE

```
#include “abdDev.h”
typedef struct _ABD_1553_MAPPING_DEST_INFO
{
        U16BIT u16ChnlNum;
        S16BIT s16MsgId;
        S8BIT  s8RtAddr;
        S8BIT  s8SA;
        U16BIT u16StartingDataWord;
        U16BIT u16StartingBit;
}ABD_1553_MAPPING_DEST_INFO, *PABD_1553_MAPPING_DEST_INFO;
```

Table 8. ABD_1553_MAPPING_DEST_INFO Structure

Struct Member	Description
u16ChnlNum	Channel Number of the destination MIL-STD-1553 channel.
s16MsgId	Message ID of destination BC Message. Note, Only applicable if 1553 Target is configured as a Bus Controller. Set to ‘-1’ for 1553 channels configured as Remote Terminals. Valid Values: 1-65535
s8RtAddr	RT Address of destination. Note, Only applicable if 1553 Target is configured as Remote Terminal(s). Set to ‘-1’ for 1553 channels configured as Bus Controller.
s8SA	RT Subaddress of destination. Note, Only applicable if 1553 Target is configured as Remote Terminal(s). Set to ‘-1’ for 1553 channels configured as Bus Controller.
u16StartingDataWord	Starting 1553 Word within destination to bridge (copy) data to. Valid Values : 0-31
u16StartingBit	Starting Bit (from above 1553 Word) within destination to bridge (copy) data to. Valid Values : 0-15

SEE ALSO

abd1553AddMatchingEntry() **abd1553AddMappingRule()**
ABD_MAPPING_DEST_INFO

ABD_429_MAPPING_DEST_INFO

Structure to define ARINC-429 protocol bridging destination location.

PROTOTYPE

```
#include “abdDev.h”
typedef struct _ABD_429_MAPPING_DEST_INFO
{
        U16BIT u16ChnlNum;
        S16BIT s16MsgId;
        U16BIT u16StartingBit;
}ABD_429_MAPPING_DEST_INFO, *PABD_429_MAPPING_DEST_INFO;
```

Table 9. ABD_429_MAPPING_DEST_INFO Structure

Struct Member	Description
u16ChnlNum	Channel Number of the destination ARINC-429 channel.
s16MsgId	Message ID of destination ARINC-429 Message. Note, Only applicable if destination 429 channel is configured as a transmitter.
u16StartingBit	Starting Bit within ‘Data’ portion (bits 10-30) of destination ARINC 429 32-bit Message. Valid Values : 10-30

SEE ALSO

abd429AddMatchingEntry() **abd429AddMappingRule()**
ABD_MAPPING_DEST_INFO

ABD_ETH_MAPPING_DEST_INFO

Structure to define Ethernet protocol bridging destination location.

PROTOTYPE

```
#include “abdDev.h”
typedef struct _ABD_ETH_MAPPING_DEST_INFO
{
        U16BIT u16ChnlNum;
        U16BIT u16MsgId;
        U16BIT u16StartingByte;
        U16BIT u16StartingBit;
}ABD_ETH_MAPPING_DEST_INFO, *PABD_ETH_MAPPING_DEST_INFO;
```

Table 10. ABD_ETH_MAPPING_DEST_INFO Structure

Struct Member	Description
u16ChnlNum	Channel Number of the destination Ethernet channel.
s16MsgId	Message ID of destination Ethernet packet
u16StartingByte	Starting Byte within destination Ethernet Packet. Valid Values : 0-65535
u16StartingBit	Starting Bit (from above starting byte) within destination Ethernet Packet. Valid Values : 0-7

SEE ALSO

abdEthAddMatchingEntry() **abdEthAddMappingRule()**
ABD_MAPPING_DEST_INFO

7.5 Available Samples

This Protocol Bridging Layer API is supplied with many examples of the proper use of the SDK and the capabilities of the hardware. The examples provided demonstrate numerous bridging techniques between the 3 available protocols (MIL-STD-1553, ARINC-429, and Ethernet).

In all cases, the examples have been provided as source codes with an appropriate “Makefiles” that may be used to build the executable.

The samples reside in the following directory on the ABD Target:

/home/ddc/ABD/ABDbridging/samples

7.5.1 Received ARINC-429 to MIL-STD-1553 Bus Controller (arinc2bc)

This sample application shows how the user can bridge a received ARINC Message to a MIL-STD-1553 Receive (BC→RT) Message.

7.5.2 Received ARINC-429 to Transmit Ethernet (arinc2eth)

This sample application shows how the user can bridge a received ARINC Message to an Ethernet transmit Packet.

7.5.3 MIL-STD-1553 Bus Controller to Transmit Ethernet (bc2eth)

This sample application shows how the user can bridge data from a MIL-STD-1553 Transmit (RT→BC) to an Ethernet transmit Packet.

7.5.4 Received Ethernet to ARINC-429 Transmit (eth2arinc)

This sample application shows how the user can bridge a received Ethernet Packet to an ARINC Transmit Message.

7.5.5 Received Ethernet to MIL-STD-1553 Bus Controller (eth2bc)

This sample application shows how the user can bridge a received Ethernet Packet to a MIL-STD-1553 Receive (BC→RT) Message.

7.5.6 Received Ethernet to MIL-STD-1553 Remote Terminal (eth2rt)

This sample application shows how the user can bridge a received Ethernet Packet to a MIL-STD-1553 Remote Terminal.

7.5.7 MIL-STD-1553 Remote Terminal to Transmit Ethernet (mrt2eth)

This sample application shows how the user can bridge data received on a MIL-STD-1553 Remote Terminal to an Ethernet transmit Packet.

7.5.8 MIL-STD-1553 Bus Monitor to ARINC-429 Transmit (mt2arinc)

This sample application shows how the user can bridge monitored MIL-STD-1553 Data to an ARINC Transmit Message.

7.5.9 MIL-STD-1553 Bus Monitor to Transmit Ethernet (mt2eth)

This sample application shows how the user can bridge monitored MIL-STD-1553 Data to an Ethernet Transmit Packet.

8 USING REMOTE ACCESS MODE

When the AceXtreme Bridge Device is put into Remote Access Mode, a virtual backplane is created between the applications running on a host computer and the MIL-STD-1553 / ARINC 429 interfaces on the ABD.   Remote Access mode allows for the ABD to be plugged into the network while the user uses the 1553 and/or 429 channels on the ABD from a remote location. Remote Access mode is supported under Windows, Linux, and VxWorks.

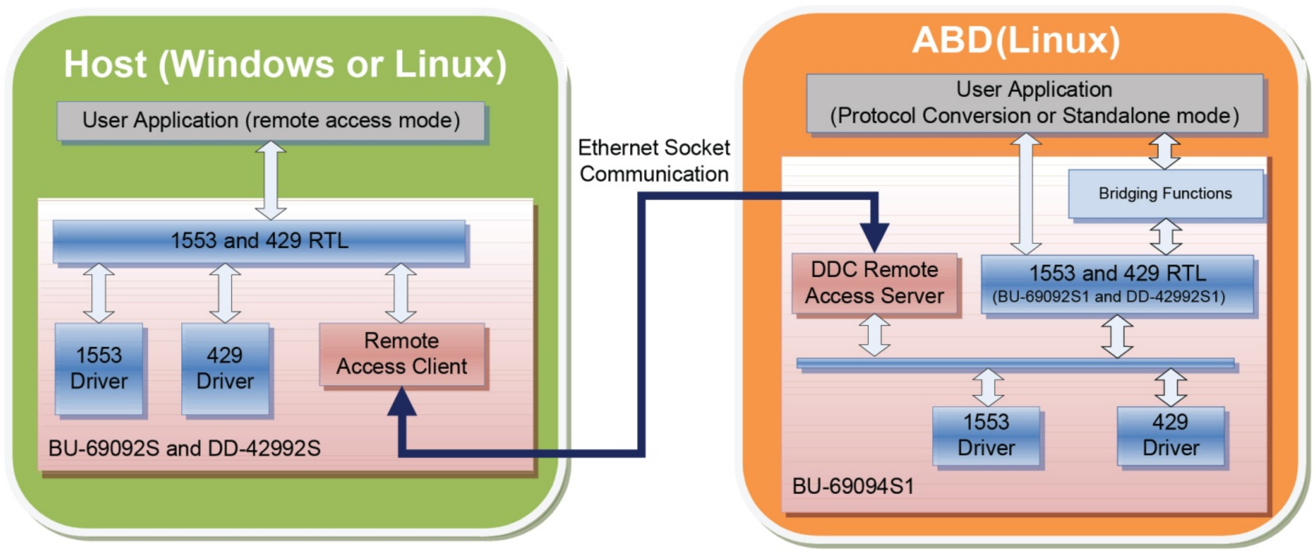


Figure 28. ABD Remote Access interaction

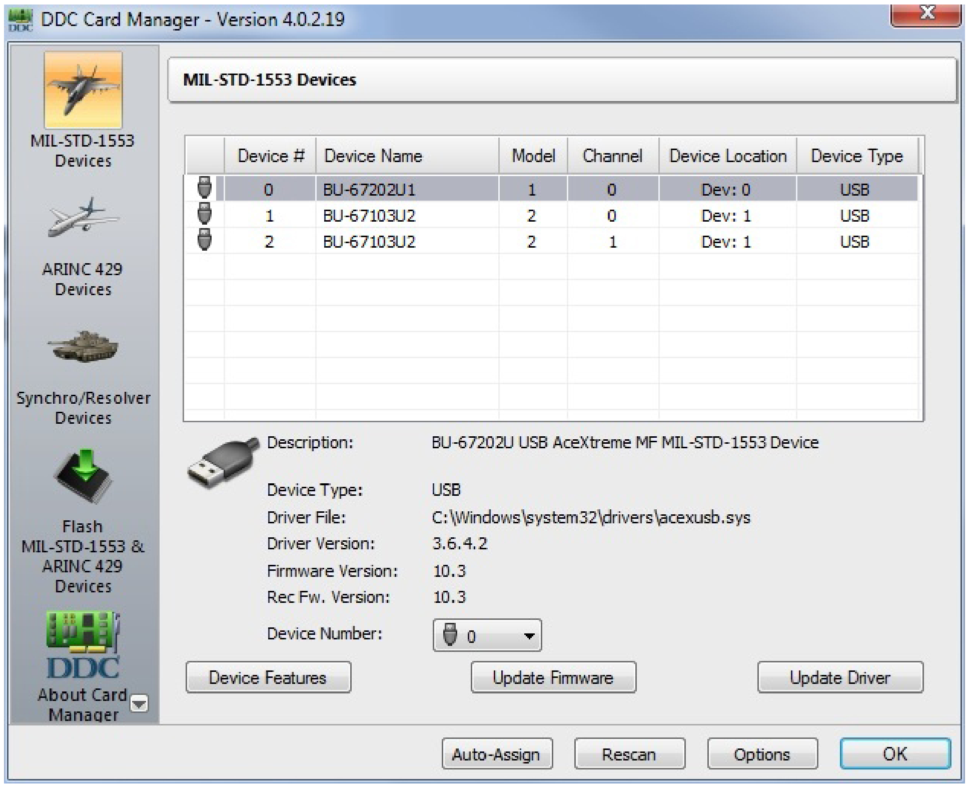
When using a Windows Operating system with the AceXtreme Bridge Device in Remote Access mode, all of DDC's software tools can be used. These tools include the AceXtreme® SDK, ARINC 429 SDK, BusTrACEr®, *data*SIMS®, Commercial Avionics Utilities, and DDC's LabVIEW® Support Package.

8.1 Remote Access Mode With Windows

To use the ABD with a Windows host Operating system, the DDC Card Manager must be used to find the ABD and assign Logical Device Numbers (LDNs) to the 1553 and 429 channels on the ABD. The DDC Card Manager is bundled with the DDC SDK software in the BU-69092S0 AceXtreme SDK, or the DD-42992S0 ARINC 429 SDK. These SDKs are needed to use the ABD under Windows and should be installed per the SDK installation instructions. Once the SDKs have been installed and the LDNs have been assigned, the ABD is ready for use with DDC's Windows software support packages.

8.1.1 Assigning Logical Devices Numbers

Assigning a LDN to the 1553 and 429 channels on the ABD is done by first opening the DDC Card Manager, located in the Windows Control Panel or as a shortcut in Start Menu. Once the DDC Card Manager is opened, there will be several buttons on the left side of the card manager. These buttons allow you to select which interface type you will be assigning an LDN for your device. The button options include **MIL-STD-1553, ARINC 429** and **Synchro/ Resolver Devices** (the ABD does not contain any Synchro / Resolver channels).



MIL-STD-1553 Devices

Device #	Device Name	Model	Channel	Device Location	Device Type
0	BU-67202U1	1	0	Dev: 0	USB
1	BU-67103U2	2	0	Dev: 1	USB
2	BU-67103U2	2	1	Dev: 1	USB

Description:	BU-67202U USB AceXtreme MF MIL-STD-1553 Device
Device Type:	USB
Driver File:	C:\Windows\system32\drivers\acexusb.sys
Driver Version:	3.6.4.2
Firmware Version:	10.3
Rec Fw. Version:	10.3
Device Number:	0

Figure 29. DDC Card Manager (Windows)

The DDC Card Manager will need to know the IP address of your ABD. To enter in the IP Address click **Options** on the lower right side of the DDC Card Manager.

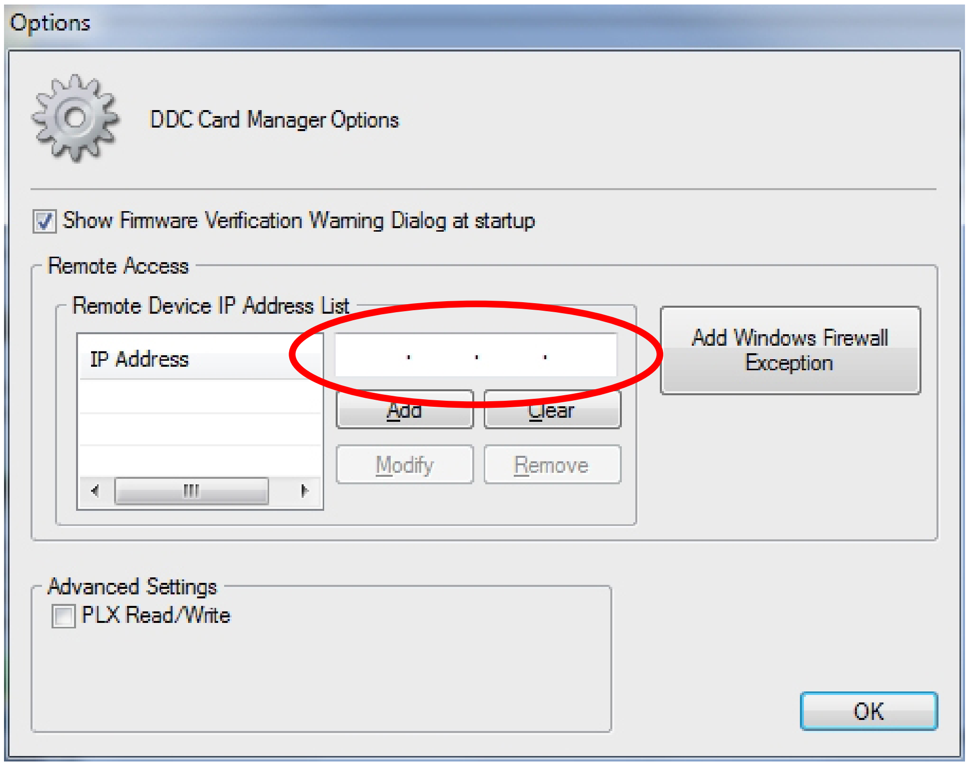
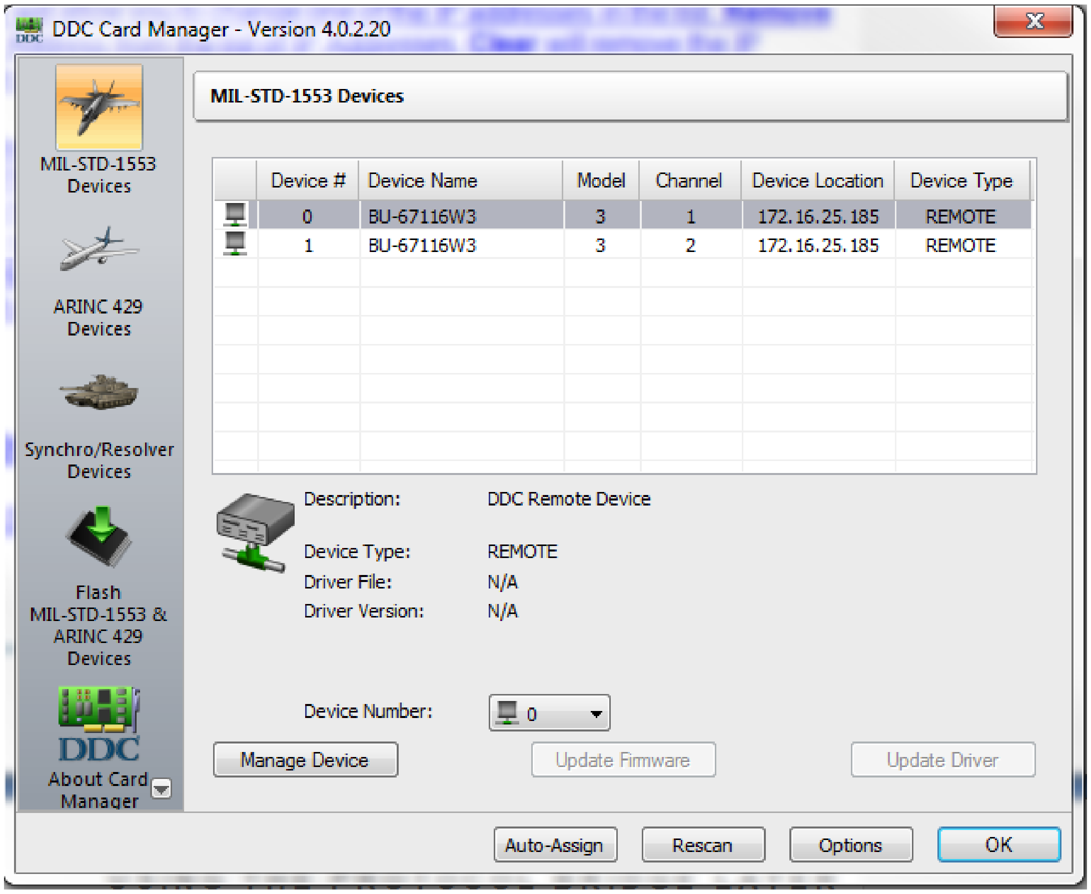

Options
DDC Card Manager Options
Show Firmware Verification Warning Dialog at startup
Remote Access
Remote Device IP Address List
IP Address
Add
Clear
Modify
Remove
Add Windows Firewall Exception
Advanced Settings
PLX Read/Write
OK

Figure 30. DDC Card Manager Options Dialog

The options dialog window will be displayed, allowing the user to enter the IP Address of the ABD in the circled area above. Clicking **Add** will insert the IP Address in the list of addresses the DDC Card Manager will search for upon loading, or when rescan is selected. **Modify** will allow you to change one of the IP addresses in the list. **Remove** will delete the IP Address from the list of IP Addresses. **Clear** will remove the IP Address from the circled box.

Once the IP Address of the ABD has been added, click **OK** to return to the main dialog window of the DDC Card Manager. Clicking **Rescan** will cause the DDC Card Manager to search for any ABDs based on the provided IP address and add the ABD’s channels to the appropriate sections. At this point, a LDN may be assigned to the 1553 and or 429 channels, and the LDNs may be used with the Windows Host operation system and DDC software.



	Device #	Device Name	Model	Channel	Device Location	Device Type
	0	BU-67116W3	3	1	172.16.25.185	REMOTE
	1	BU-67116W3	3	2	172.16.25.185	REMOTE

Description: DDC Remote Device

Device Type: REMOTE

Driver File: N/A

Driver Version: N/A

Device Number: 0

Figure 31. DDC Card Manager with Remote MIL-STD-1553 Devices.

Note: The 429 LDNs are assigned by clicking **ARINC 429 Devices**.

8.1.2 Removing ABD from Card Manager

When removing the ABD from the Windows host, the LDNs and IP address must be removed. It is recommended to first set all LDNs to **NONE** for any channels on the ABD. Then click **Options** to open the DDC Card Manager Options dialog. The ABD's IP address will be in the IP Address list. Click on it to highlight the IP address and then click the remove button to remove the IP Address from the list.

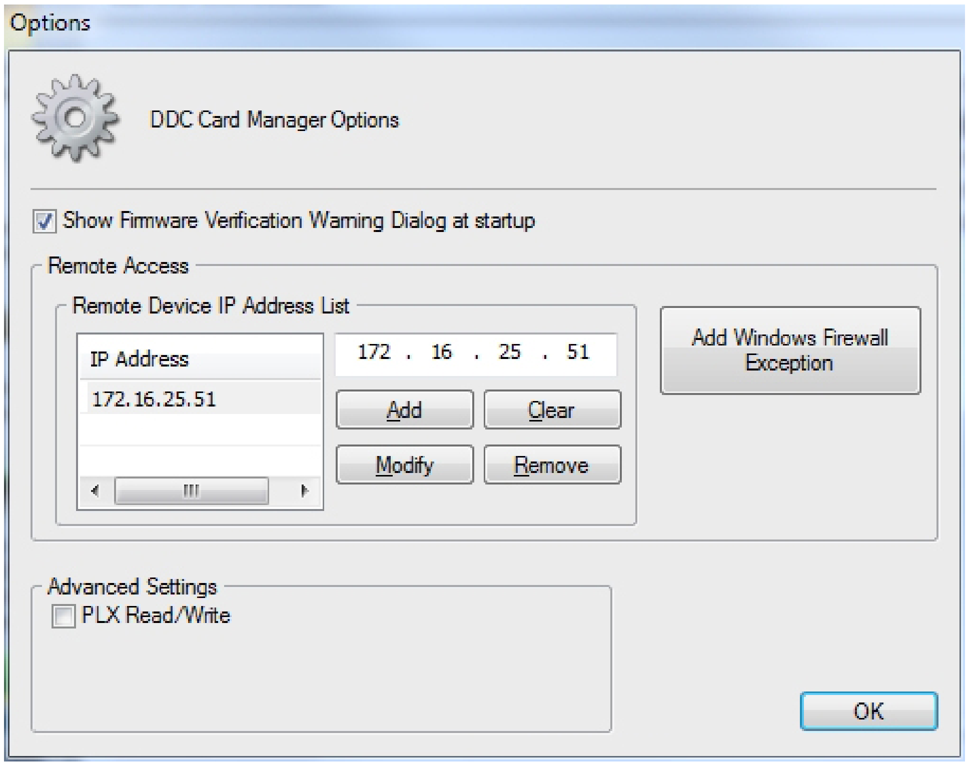


Figure 32. DDC Card Manager – Remove IP

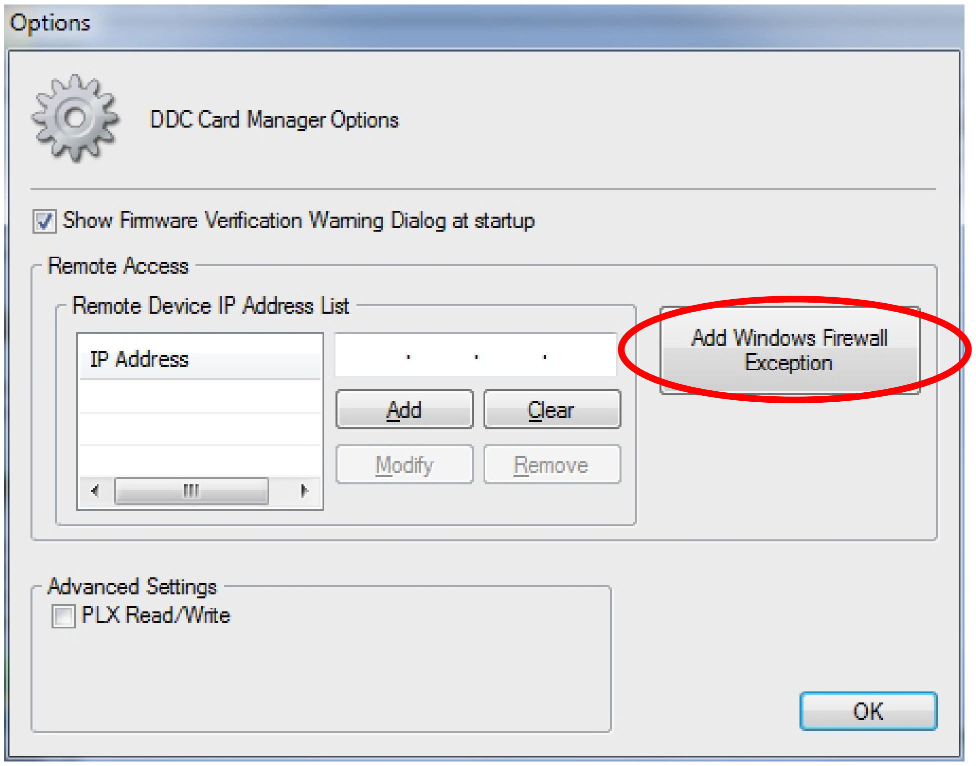
Once the IP Address has been removed, the DDC Card Manager will no longer see the 1553/429 channels on the ABD.

8.1.3 Forwarding Port through Windows Firewall for Remote Access

Windows 7 and Windows 8 may require Ports to be opened within the Windows Firewall for Remote Access Mode. These ports may be opened through the DDC Card Manager (version 4.0.2.19 or later).

The DDC Card Manager (version 4.0.2.19 or later) has the option to open the ports needed by the ABD when using Remote Access Mode. In order to open these ports the user must first open the DDC Card Manager and click **Options** (see Figure 29).

Once the options dialog has been opened, click **Add Windows Firewall Exception** to add the required ports to your firewall.

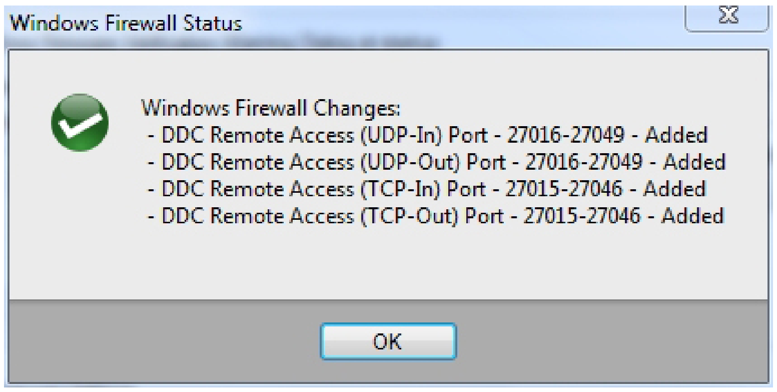


☑ Show Firmware Verification Warning Dialog at startup

☐ PLX Read/Write

Figure 33. DDC Card Manager Options Dialog

Clicking **Add Windows Firewall Exception** will add the necessary ports starting at port 27016. The number of ports opened will depend on the channel count of your device.



```
Windows Firewall Changes:
 - DDC Remote Access (UDP-In) Port - 27016-27049 - Added
 - DDC Remote Access (UDP-Out) Port - 27016-27049 - Added
 - DDC Remote Access (TCP-In) Port - 27015-27046 - Added
 - DDC Remote Access (TCP-Out) Port - 27015-27046 - Added
```

Figure 34. DDC Card Manager Firewall Changes.

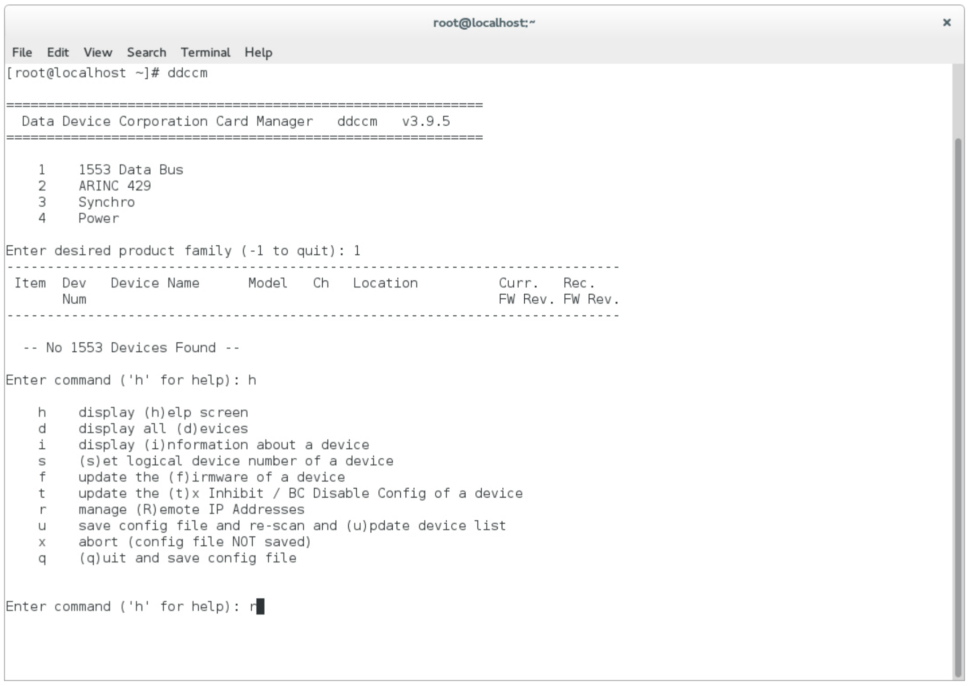
8.2 Remote Access Mode With Linux

To use the ABD with a Linux host Operating system, the DDC Card Manager must be used to find the ABD and assign Logical Device Numbers (LDNs) to the 1553 and 429 channels on the ABD. The DDC Card Manager is bundled with the DDC SDK software in the BU-69092S1 AceXtreme SDK, or the DD-42992S1 ARINC 429 SDK. These SDKs are needed to use the ABD under Linux and should be installed per the SDK installation instructions. Once the SDKs have been installed and the LDNs have been assigned, the ABD is ready for use with DDC's Linux software support packages.

8.2.1 Assigning Logical Devices Numbers

Assigning a LDN to the 1553 and 429 channels on the ABD is done by first opening the DDC Card Manager. This is done from the prompt, by typing **ddccm**. To run **ddccm**, the user must have root privileges; otherwise the Card Manager will report an error.

After starting **ddccm**, enter '**1'** for **1553 Data Bus channels** or '**2'** for **ARINC 429**. Figure 35 shows the MIL-STD-1553 channels that are available when selecting 1 from the prompt. The lack of channels is due to the IP Address of the ABD not being added to the IP Address list. At the prompt enter in '**h**' to have **ddccm** display the help context menu which displays a list of commands.

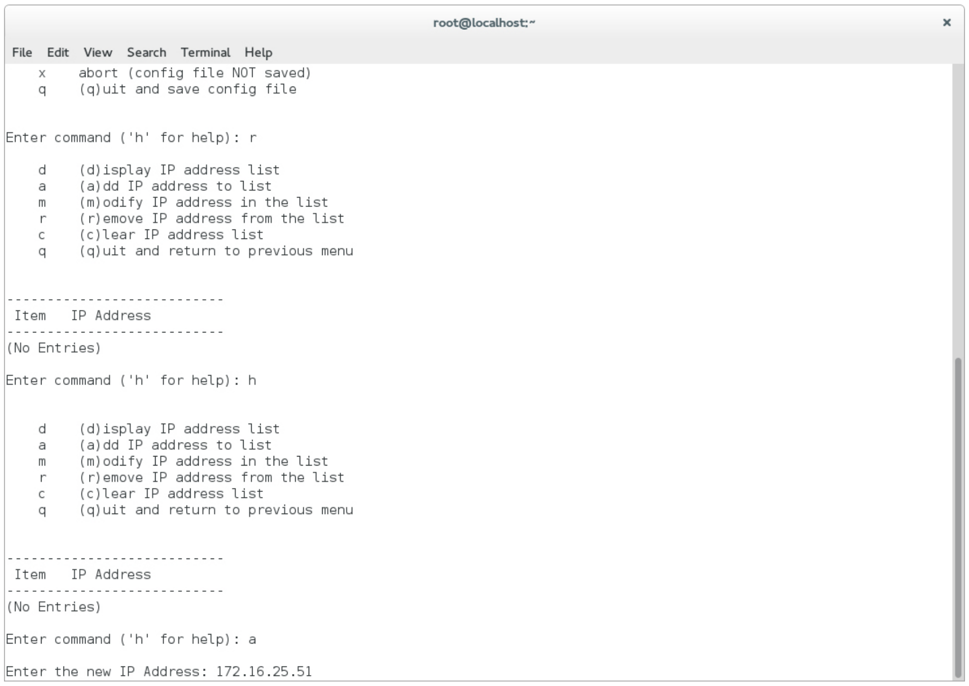


```
[root@localhost ~]# ddccm

==========================================================
  Data Device Corporation Card Manager   ddccm   v3.9.5
==========================================================

    1    1553 Data Bus
    2    ARINC 429
    3    Synchro
    4    Power

Enter desired product family (-1 to quit): 1
----------------------------------------------------------------------------
 Item  Dev   Device Name      Model   Ch   Location          Curr.   Rec.
       Num                                                   FW Rev. FW Rev.
----------------------------------------------------------------------------

  -- No 1553 Devices Found --

Enter command ('h' for help): h

    h    display (h)elp screen
    d    display all (d)evices
    i    display (i)nformation about a device
    s    (s)et logical device number of a device
    f    update the (f)irmware of a device
    t    update the (t)x Inhibit / BC Disable Config of a device
    r    manage (R)emote IP Addresses
    u    save config file and re-scan and (u)pdate device list
    x    abort (config file NOT saved)
    q    (q)uit and save config file


Enter command ('h' for help): r
```

Figure 35. Running ddccm (Linux)

From the prompt, enter '**a**' to **(a)dd an IP address to the list** of searchable IP addresses for the card manager. The card manager will now prompt the user for the

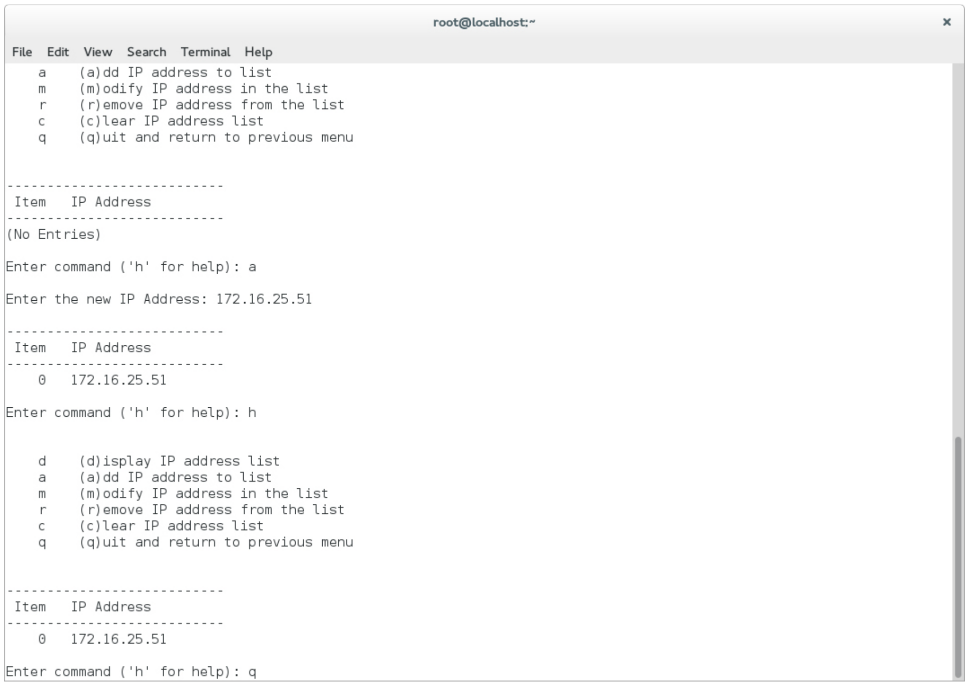
IP address to add to the list of IP addresses. Enter in the IP address of the ABD and press enter.



```
    x    abort (config file NOT saved)
    q    (q)uit and save config file


Enter command ('h' for help): r

    d    (d)isplay IP address list
    a    (a)dd IP address to list
    m    (m)odify IP address in the list
    r    (r)emove IP address from the list
    c    (c)lear IP address list
    q    (q)uit and return to previous menu


---------------------------
 Item   IP Address
---------------------------
(No Entries)

Enter command ('h' for help): h


    d    (d)isplay IP address list
    a    (a)dd IP address to list
    m    (m)odify IP address in the list
    r    (r)emove IP address from the list
    c    (c)lear IP address list
    q    (q)uit and return to previous menu


---------------------------
 Item   IP Address
---------------------------
(No Entries)

Enter command ('h' for help): a

Enter the new IP Address: 172.16.25.51
```

Figure 36. Selecting Add/Entering IP Address of ABD

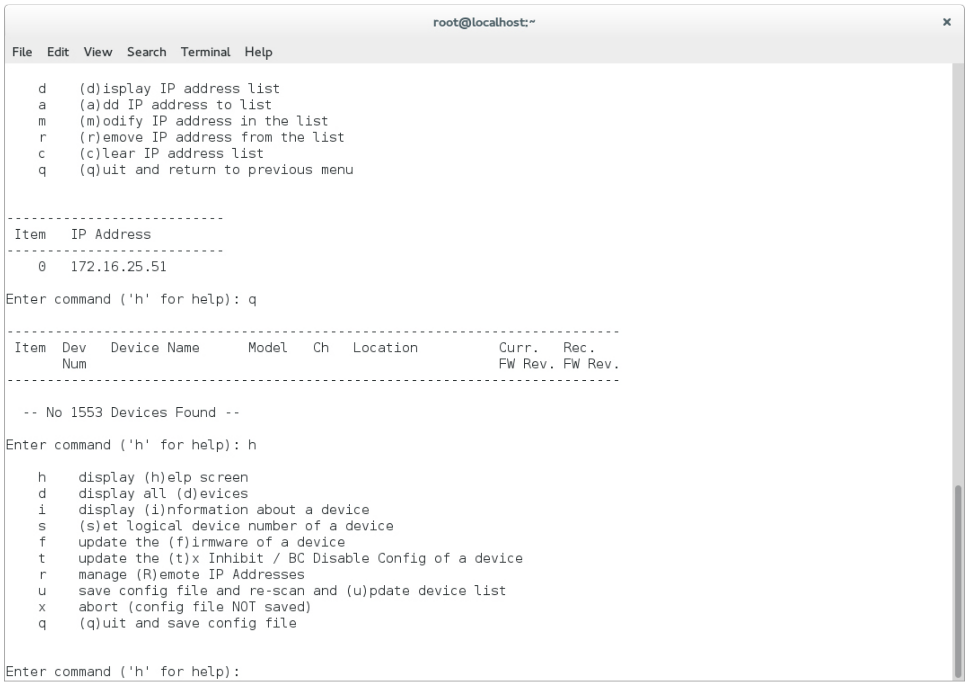
After entering in the IP address, enter in ‘**h**’ at the prompt again to display a new list of selectable commands to perform within **ddccm**. Enter in ‘**q**’ to exit the current level and return back to the Device listing portion of the **ddccm**.



```
    a    (a)dd IP address to list
    m    (m)odify IP address in the list
    r    (r)emove IP address from the list
    c    (c)lear IP address list
    q    (q)uit and return to previous menu


---------------------------
 Item   IP Address
---------------------------
(No Entries)

Enter command ('h' for help): a

Enter the new IP Address: 172.16.25.51

---------------------------
 Item   IP Address
---------------------------
    0   172.16.25.51

Enter command ('h' for help): h


    d    (d)isplay IP address list
    a    (a)dd IP address to list
    m    (m)odify IP address in the list
    r    (r)emove IP address from the list
    c    (c)lear IP address list
    q    (q)uit and return to previous menu


---------------------------
 Item   IP Address
---------------------------
    0   172.16.25.51

Enter command ('h' for help): q
```

Figure 37. Entering IP Address

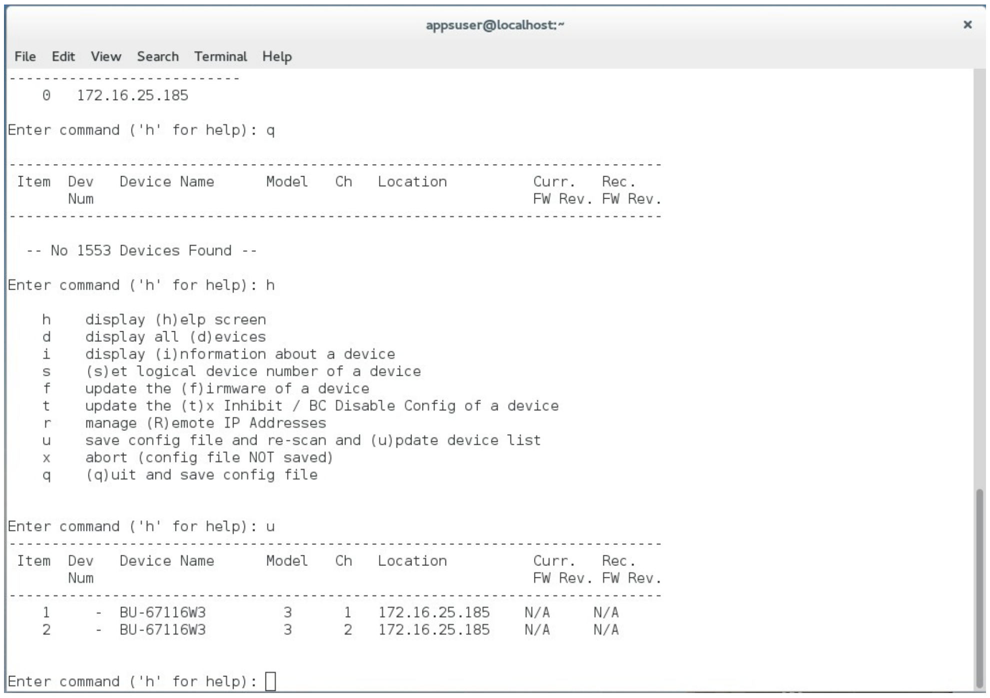
From the prompt, select '**u**' to **save the configuration file and to perform a re-scan and (u)pdate device list** for MIL-STD-1553 and ARINC 429 devices. Any found devices on the ABD will be displayed as shown by Figure 38. To see the ARINC 429 devices, ddccm must be exited and run again.



```
    d    (d)isplay IP address list
    a    (a)dd IP address to list
    m    (m)odify IP address in the list
    r    (r)emove IP address from the list
    c    (c)lear IP address list
    q    (q)uit and return to previous menu


---------------------------
 Item   IP Address
---------------------------
    0   172.16.25.51

Enter command ('h' for help): q

----------------------------------------------------------------------------
 Item  Dev   Device Name      Model   Ch   Location          Curr.   Rec.
       Num                                                   FW Rev. FW Rev.
----------------------------------------------------------------------------

  -- No 1553 Devices Found --

Enter command ('h' for help): h

    h    display (h)elp screen
    d    display all (d)evices
    i    display (i)nformation about a device
    s    (s)et logical device number of a device
    f    update the (f)irmware of a device
    t    update the (t)x Inhibit / BC Disable Config of a device
    r    manage (R)emote IP Addresses
    u    save config file and re-scan and (u)pdate device list
    x    abort (config file NOT saved)
    q    (q)uit and save config file


Enter command ('h' for help):
```

Figure 38. Saving Configuration file

The final step is to assign an LDN to each channel on the ABD. This is done by entering ‘**s**’ from the prompt for **(s)et logical device number of a device**. This causes the card manager to ask for which item number, and what LDN number to use for the channel. To switch between ARINC 429 and MIL-STD-1553 devices, the user must close and then reopen **ddccm**.

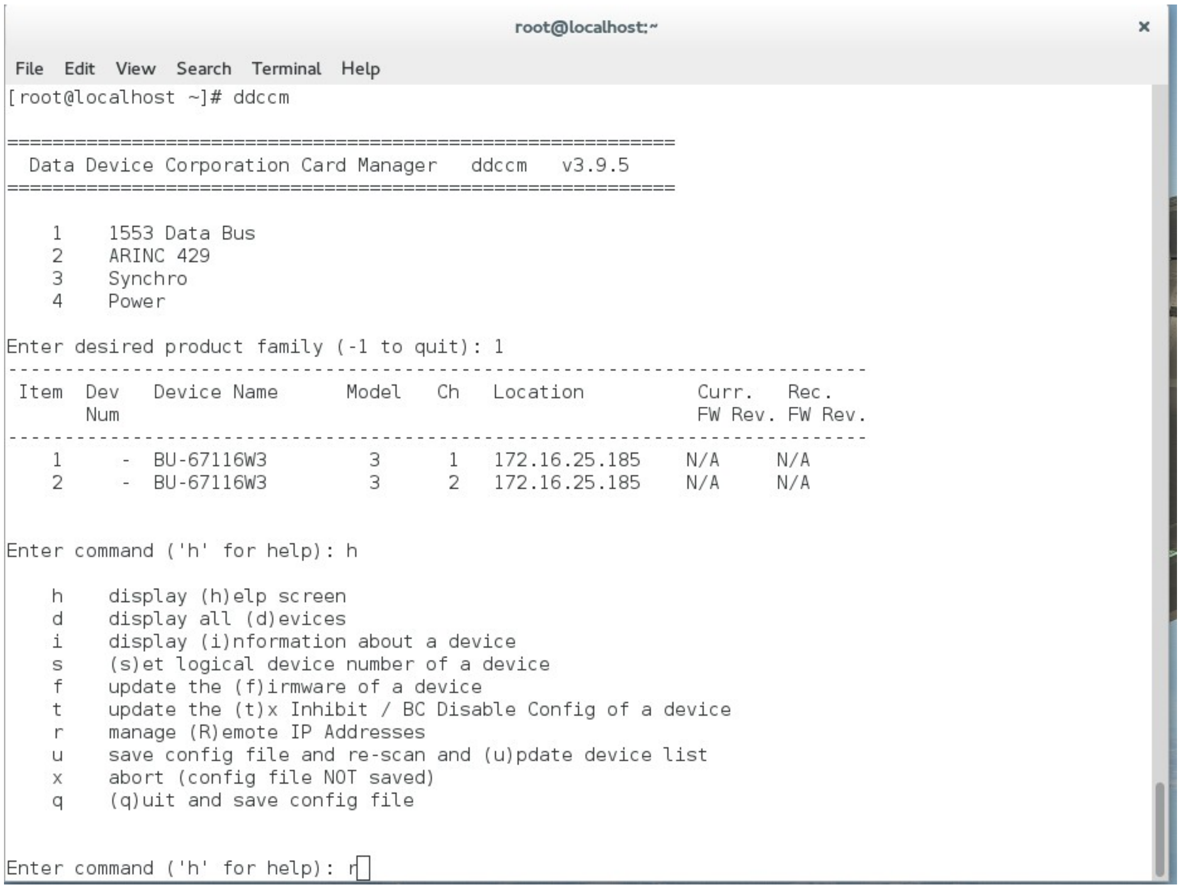


```
---------------------------
    0   172.16.25.185

Enter command ('h' for help): q

----------------------------------------------------------------------------
 Item  Dev   Device Name      Model   Ch   Location          Curr.   Rec.
       Num                                                   FW Rev. FW Rev.
----------------------------------------------------------------------------

  -- No 1553 Devices Found --

Enter command ('h' for help): h

    h    display (h)elp screen
    d    display all (d)evices
    i    display (i)nformation about a device
    s    (s)et logical device number of a device
    f    update the (f)irmware of a device
    t    update the (t)x Inhibit / BC Disable Config of a device
    r    manage (R)emote IP Addresses
    u    save config file and re-scan and (u)pdate device list
    x    abort (config file NOT saved)
    q    (q)uit and save config file


Enter command ('h' for help): u
----------------------------------------------------------------------------
 Item  Dev   Device Name      Model   Ch   Location          Curr.   Rec.
       Num                                                   FW Rev. FW Rev.
----------------------------------------------------------------------------
    1     -  BU-67116W3         3      1   172.16.25.185    N/A     N/A
    2     -  BU-67116W3         3      2   172.16.25.185    N/A     N/A


Enter command ('h' for help):
```

Figure 39. Listing Found Devices

8.2.2 Removing ABD from Card Manager

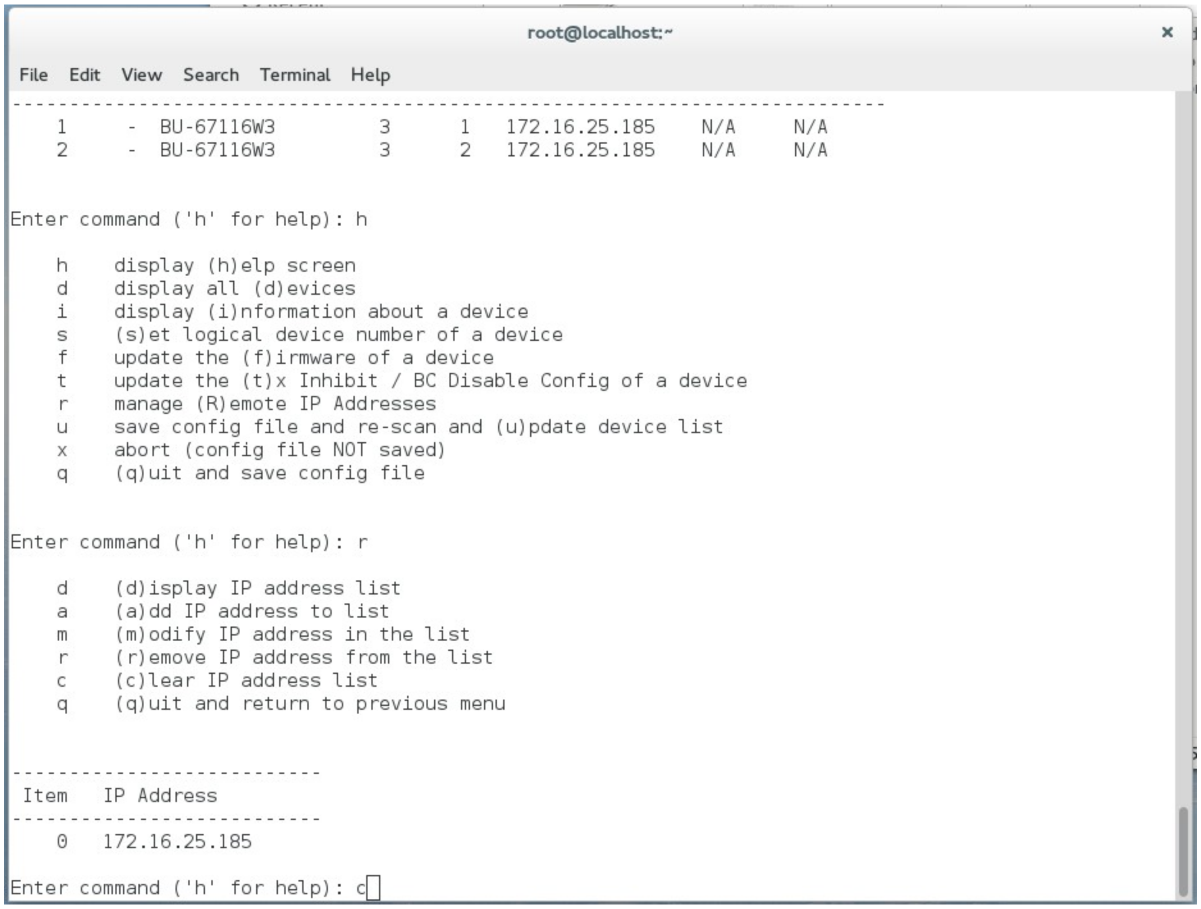
The DDC Card Manager (**ddccm**) can be used to remove the ABD's IP Address from the searchable IP Address list as well. To remove the IP Address, open the DDC Card Manager by entering in **ddccm** at the prompt with superuser privileges. Select either MIL-STD-1553 or ARINC 429 from the first menu selection. Enter '**r**' to select **(r)emove IP addres from the list**.



```
[root@localhost ~]# ddccm

============================================================
  Data Device Corporation Card Manager   ddccm   v3.9.5
============================================================

    1    1553 Data Bus
    2    ARINC 429
    3    Synchro
    4    Power

Enter desired product family (-1 to quit): 1
-----------------------------------------------------------------------------
 Item  Dev   Device Name      Model   Ch   Location          Curr.    Rec.
       Num                                                   FW Rev.  FW Rev.
-----------------------------------------------------------------------------
    1      -  BU-67116W3         3      1   172.16.25.185     N/A      N/A
    2      -  BU-67116W3         3      2   172.16.25.185     N/A      N/A


Enter command ('h' for help): h

    h    display (h)elp screen
    d    display all (d)evices
    i    display (i)nformation about a device
    s    (s)et logical device number of a device
    f    update the (f)irmware of a device
    t    update the (t)x Inhibit / BC Disable Config of a device
    r    manage (R)emote IP Addresses
    u    save config file and re-scan and (u)pdate device list
    x    abort (config file NOT saved)
    q    (q)uit and save config file


Enter command ('h' for help): r
```

Figure 40. Removing IP Address

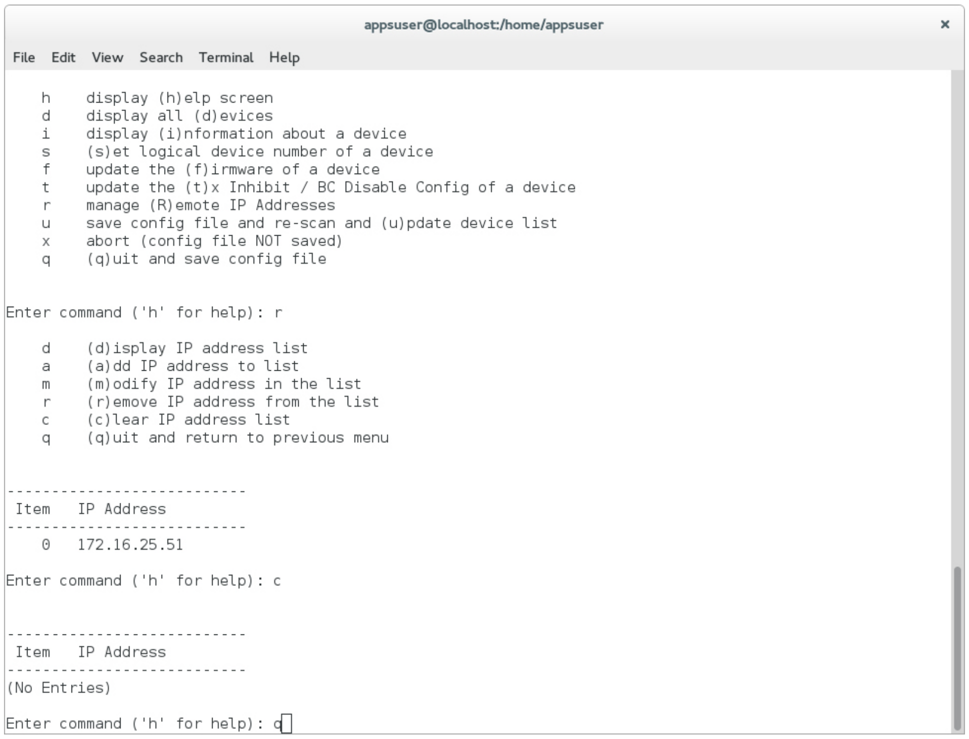
After selecting ‘**r**’ to enter the mange **(R)emote IP Addresses** section of the card manager, select ‘**c**’ to **(c)lear IP address list**. This will remove all IP Addresses from the DDC Card Managers list. If the user wishes to remove a particular IP Address the ‘**r**’ selection can be used.



```
----------------------------------------------------------------------------
    1     -  BU-67116W3        3      1   172.16.25.185    N/A      N/A
    2     -  BU-67116W3        3      2   172.16.25.185    N/A      N/A


Enter command ('h' for help): h

    h    display (h)elp screen
    d    display all (d)evices
    i    display (i)nformation about a device
    s    (s)et logical device number of a device
    f    update the (f)irmware of a device
    t    update the (t)x Inhibit / BC Disable Config of a device
    r    manage (R)emote IP Addresses
    u    save config file and re-scan and (u)pdate device list
    x    abort (config file NOT saved)
    q    (q)uit and save config file


Enter command ('h' for help): r

    d    (d)isplay IP address list
    a    (a)dd IP address to list
    m    (m)odify IP address in the list
    r    (r)emove IP address from the list
    c    (c)lear IP address list
    q    (q)uit and return to previous menu


---------------------------
 Item   IP Address
---------------------------
    0   172.16.25.185

Enter command ('h' for help): c
```

Figure 41. Clearing IP Address.

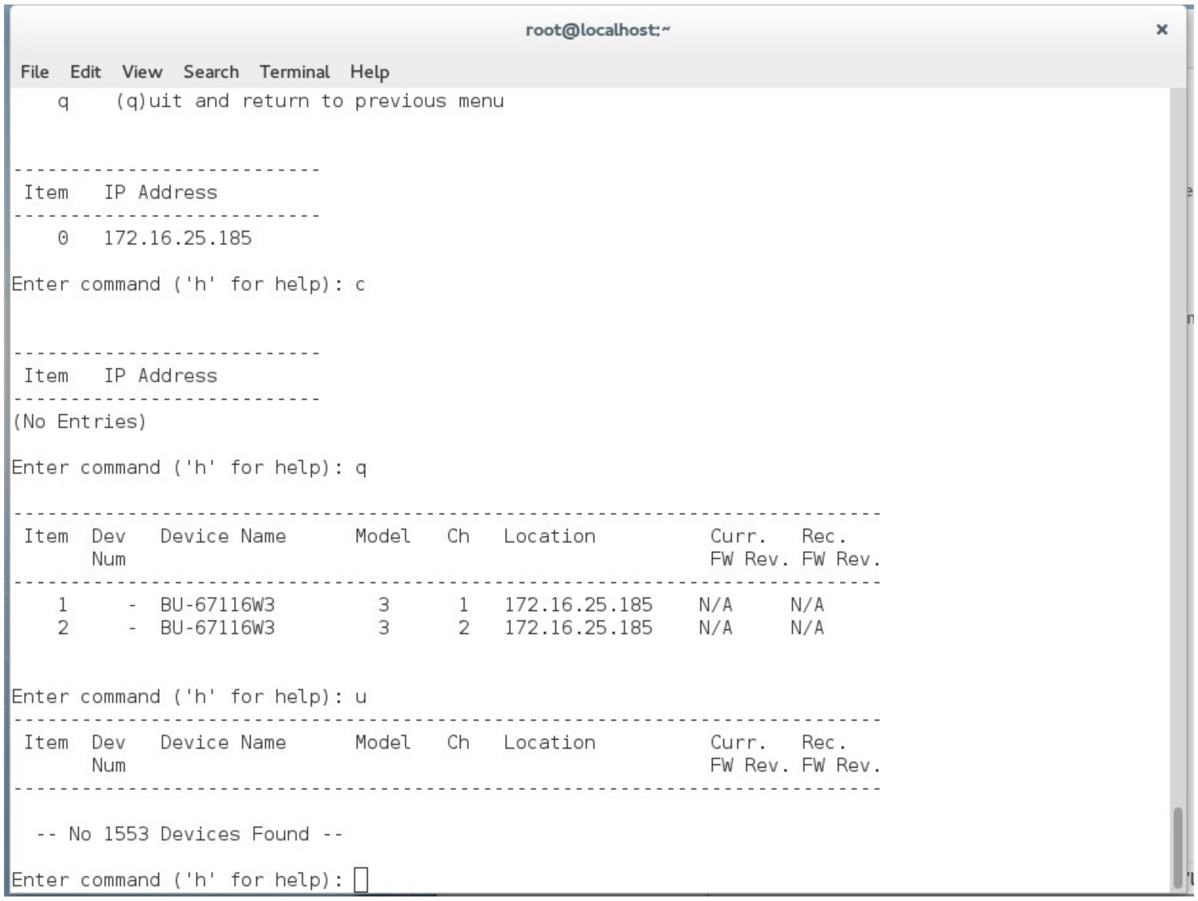
After removing the IP Address from the list, select '**q**' to exit back to Device List portion of the DDC Card Manager.


appsuser@localhost:/home/appsuser
File Edit View Search Terminal Help

```
    h    display (h)elp screen
    d    display all (d)evices
    i    display (i)nformation about a device
    s    (s)et logical device number of a device
    f    update the (f)irmware of a device
    t    update the (t)x Inhibit / BC Disable Config of a device
    r    manage (R)emote IP Addresses
    u    save config file and re-scan and (u)pdate device list
    x    abort (config file NOT saved)
    q    (q)uit and save config file


Enter command ('h' for help): r

    d    (d)isplay IP address list
    a    (a)dd IP address to list
    m    (m)odify IP address in the list
    r    (r)emove IP address from the list
    c    (c)lear IP address list
    q    (q)uit and return to previous menu


---------------------------
 Item   IP Address
---------------------------
    0   172.16.25.51

Enter command ('h' for help): c


---------------------------
 Item   IP Address
---------------------------
(No Entries)

Enter command ('h' for help): q
```

Figure 42. Returning to previous menu.

To update the device list, enter ‘**u**’ at the prompt, and the DDC Card Manager will no longer see any ABD devices.



```
    q    (q)uit and return to previous menu


---------------------------
 Item   IP Address
---------------------------
    0   172.16.25.185

Enter command ('h' for help): c


---------------------------
 Item   IP Address
---------------------------
(No Entries)

Enter command ('h' for help): q

-----------------------------------------------------------------------------
 Item  Dev    Device Name      Model    Ch    Location          Curr.    Rec.
       Num                                                      FW Rev.  FW Rev.
-----------------------------------------------------------------------------
    1      -  BU-67116W3         3       1    172.16.25.185    N/A      N/A
    2      -  BU-67116W3         3       2    172.16.25.185    N/A      N/A


Enter command ('h' for help): u
-----------------------------------------------------------------------------
 Item  Dev    Device Name      Model    Ch    Location          Curr.    Rec.
       Num                                                      FW Rev.  FW Rev.
-----------------------------------------------------------------------------

  -- No 1553 Devices Found --

Enter command ('h' for help):
```

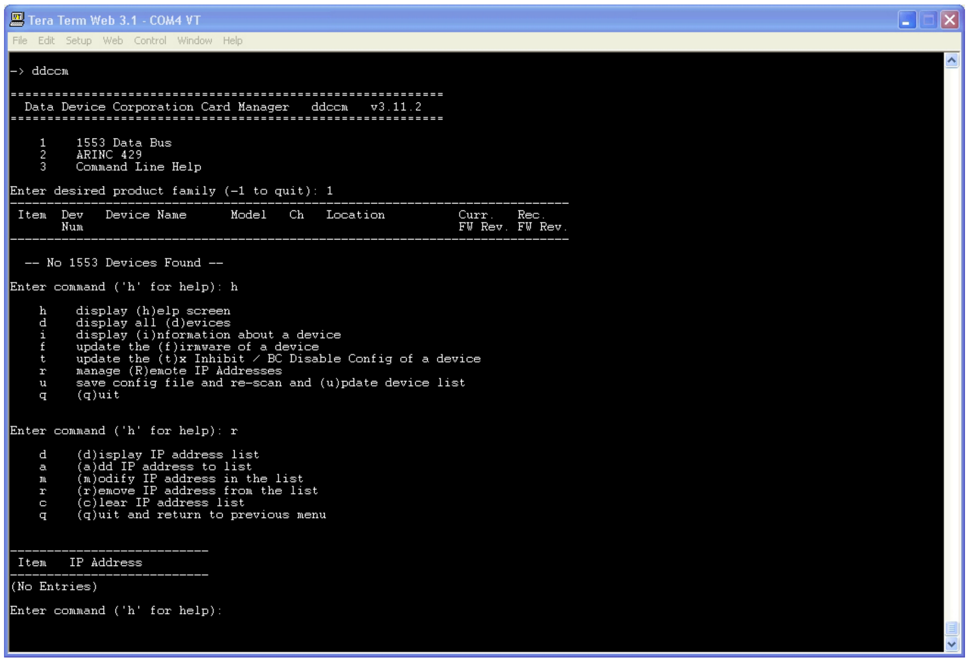
Figure 43. Updating Device List.

8.3 Remote Access Mode With VxWorks

To use the ABD with a VxWorks host Operating system, the DDC Card Manager must be used to find the ABD and assign Logical Device Numbers (LDNs) to the 1553 and 429 channels on the ABD. The DDC Card Manager is bundled with the DDC SDK software in the BU-69092S2 AceXtreme SDK, or the DD-42992S2 ARINC 429 SDK. These SDKs are needed to use the ABD under VxWorks and should be installed per the SDK installation instructions. Once the SDKs have been installed and the LDNs have been assigned, the ABD is ready for use with DDC's VxWorks software support packages.

8.3.1 Assigning Logical Devices Numbers

Assigning a LDN to the 1553 and 429 channels on the ABD is done by first opening the DDC Card Manager. This is done from the prompt, by typing **ddccm**. After starting **ddccm**, enter '**1**' for **1553 Data Bus channels** or '**2**' for **ARINC 429**. Figure 44 shows the MIL-STD-1553 channels that are available when selecting 1 from the prompt. The lack of channels is due to the IP Address of the ABD not being added to the IP Address list.   At the prompt enter in ‘**h**’ to have **ddccm** display the help context menu which displays a list of commands.

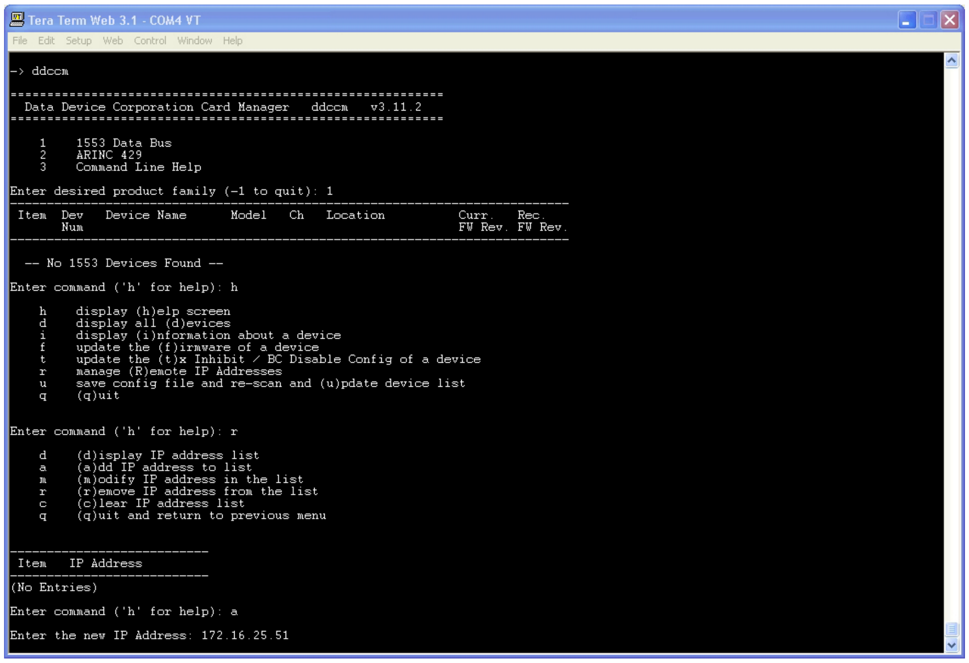


Tera Term Web 3.1 - COM4 VT
File Edit Setup Web Control Window Help

```
-> ddccm

===========================================================
  Data Device Corporation Card Manager   ddccm   v3.11.2
===========================================================

    1    1553 Data Bus
    2    ARINC 429
    3    Command Line Help

Enter desired product family (-1 to quit): 1
----------------------------------------------------------------------------
 Item  Dev   Device Name      Model   Ch   Location           Curr.   Rec.
       Num                                                    FW Rev. FW Rev.
----------------------------------------------------------------------------

  -- No 1553 Devices Found --

Enter command ('h' for help): h

    h    display (h)elp screen
    d    display all (d)evices
    i    display (i)nformation about a device
    f    update the (f)irmware of a device
    t    update the (t)x Inhibit / BC Disable Config of a device
    r    manage (R)emote IP Addresses
    u    save config file and re-scan and (u)pdate device list
    q    (q)uit


Enter command ('h' for help): r

    d    (d)isplay IP address list
    a    (a)dd IP address to list
    m    (m)odify IP address in the list
    r    (r)emove IP address from the list
    c    (c)lear IP address list
    q    (q)uit and return to previous menu


---------------------------
 Item   IP Address
---------------------------
(No Entries)

Enter command ('h' for help):
```

Figure 44. Running ddccm (VxWorks)

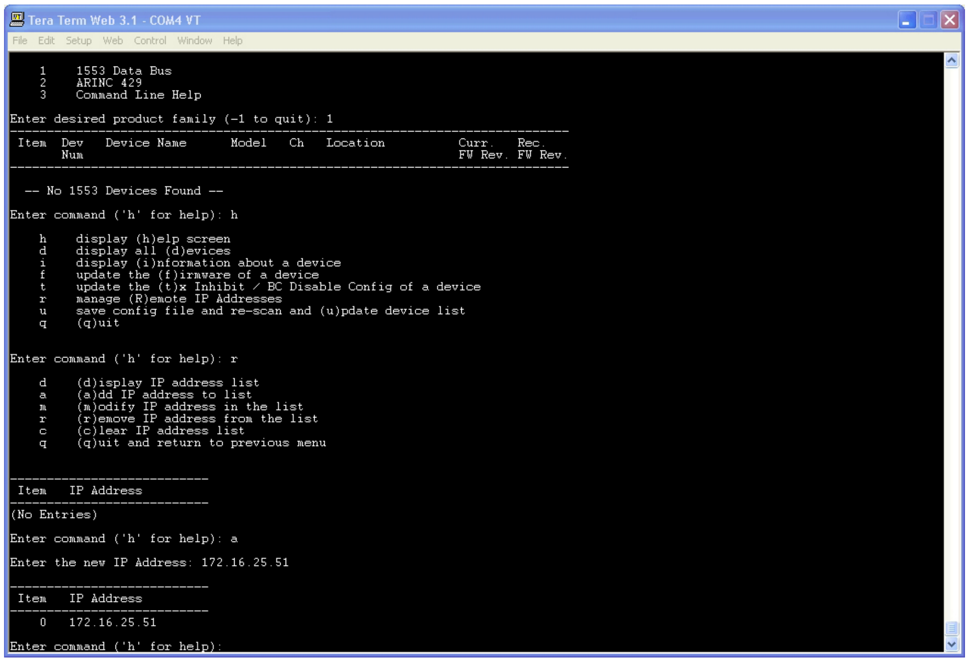
From the prompt, type ‘**a**’ to **(a)dd an IP address to the list** of searchable IP addresses for the card manager. The card manager will now prompt the user for the IP address to add to the list of IP addresses. Enter in the IP address of the ABD and press **enter**.


Tera Term Web 3.1 - COM4 VT
File Edit Setup Web Control Window Help

```
-> ddccm

==========================================================
  Data Device Corporation Card Manager   ddccm   v3.11.2
==========================================================

    1    1553 Data Bus
    2    ARINC 429
    3    Command Line Help

Enter desired product family (-1 to quit): 1
---------------------------------------------------------------------------
 Item  Dev   Device Name      Model   Ch   Location          Curr.   Rec.
       Num                                                   FW Rev. FW Rev.
---------------------------------------------------------------------------

  -- No 1553 Devices Found --

Enter command ('h' for help): h

    h    display (h)elp screen
    d    display all (d)evices
    i    display (i)nformation about a device
    f    update the (f)irmware of a device
    t    update the (t)x Inhibit / BC Disable Config of a device
    r    manage (R)emote IP Addresses
    u    save config file and re-scan and (u)pdate device list
    q    (q)uit

Enter command ('h' for help): r

    d    (d)isplay IP address list
    a    (a)dd IP address to list
    m    (m)odify IP address in the list
    r    (r)emove IP address from the list
    c    (c)lear IP address list
    q    (q)uit and return to previous menu


---------------------------
 Item   IP Address
---------------------------
(No Entries)

Enter command ('h' for help): a

Enter the new IP Address: 172.16.25.51
```

Figure 45. Selecting Add / Entering IP Address of ABD

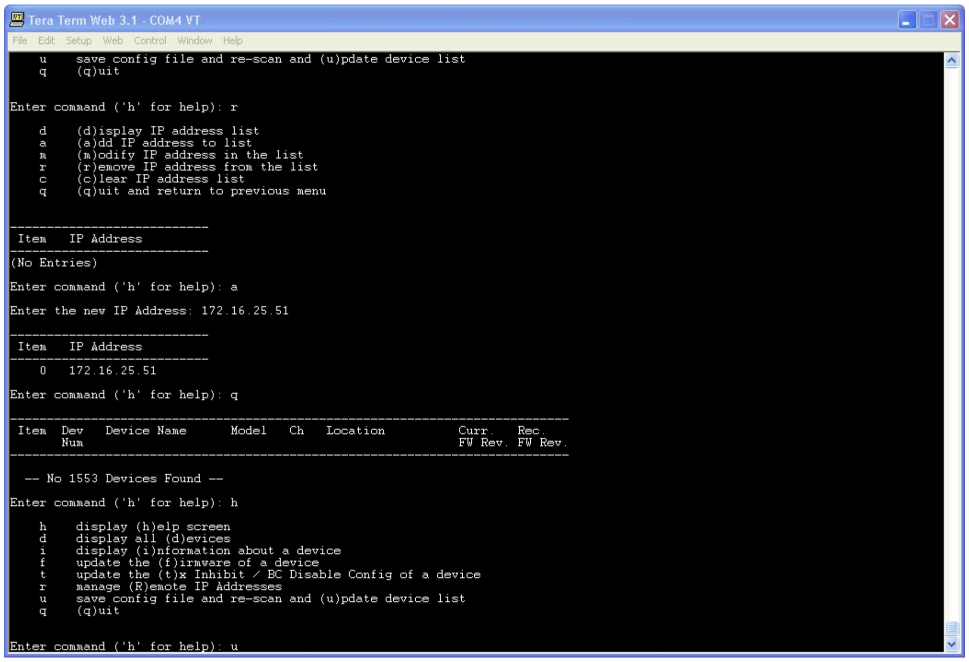
After entering in the IP address, type ‘**h**’ at the prompt again to display a new list of selectable commands to perform within **ddccm**. Type ‘**q**’ to exit the current level and return back to the Device listing portion of the **ddccm**.


Tera Term Web 3.1 - COM4 VT
File Edit Setup Web Control Window Help

```
    1    1553 Data Bus
    2    ARINC 429
    3    Command Line Help

Enter desired product family (-1 to quit): 1
---------------------------------------------------------------------------
 Item  Dev   Device Name      Model   Ch   Location          Curr.   Rec.
       Num                                                   FW Rev. FW Rev.
---------------------------------------------------------------------------

  -- No 1553 Devices Found --

Enter command ('h' for help): h

    h    display (h)elp screen
    d    display all (d)evices
    i    display (i)nformation about a device
    f    update the (f)irmware of a device
    t    update the (t)x Inhibit / BC Disable Config of a device
    r    manage (R)emote IP Addresses
    u    save config file and re-scan and (u)pdate device list
    q    (q)uit

Enter command ('h' for help): r

    d    (d)isplay IP address list
    a    (a)dd IP address to list
    m    (m)odify IP address in the list
    r    (r)emove IP address from the list
    c    (c)lear IP address list
    q    (q)uit and return to previous menu


---------------------------
 Item   IP Address
---------------------------
(No Entries)

Enter command ('h' for help): a

Enter the new IP Address: 172.16.25.51

---------------------------
 Item   IP Address
---------------------------
    0   172.16.25.51

Enter command ('h' for help):
```

Figure 46. Entering IP Address

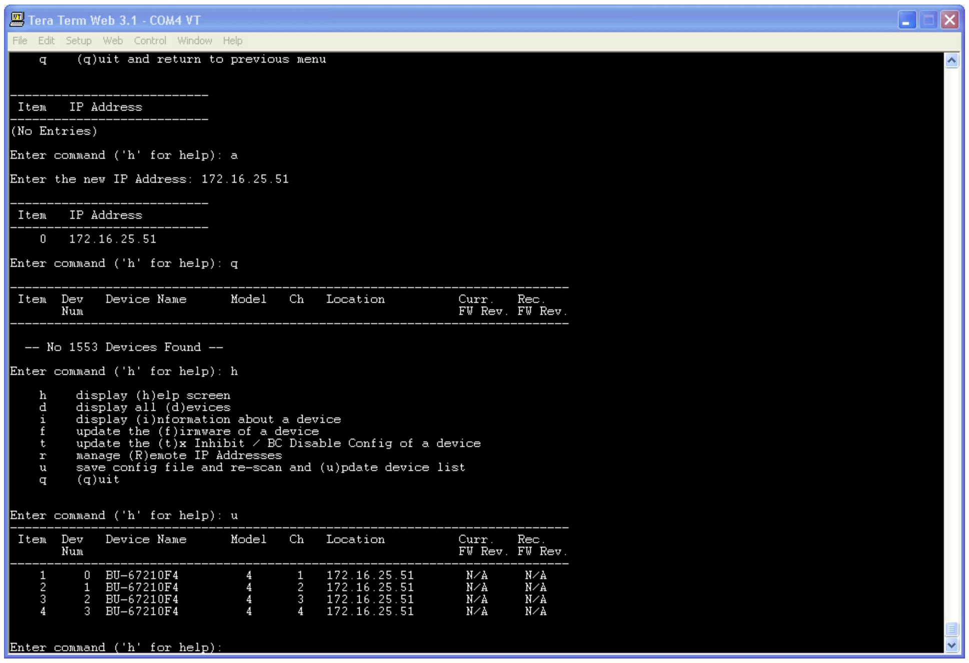
From the prompt, select ‘**u**’ to **save the configuration file and to perform a re-scan and (u)pdate device list** for MIL-STD-1553 and ARINC 429 devices. Any found devices on the ABD will be displayed as shown by Figure 47. To see the ARINC 429 devices, **ddccm** must be exited and run again.


Tera Term Web 3.1 - COM4 VT
File Edit Setup Web Control Window Help

```
    u    save config file and re-scan and (u)pdate device list
    q    (q)uit


Enter command ('h' for help): r

    d    (d)isplay IP address list
    a    (a)dd IP address to list
    m    (m)odify IP address in the list
    r    (r)emove IP address from the list
    c    (c)lear IP address list
    q    (q)uit and return to previous menu


---------------------------
 Item   IP Address
---------------------------
(No Entries)

Enter command ('h' for help): a

Enter the new IP Address: 172.16.25.51

---------------------------
 Item   IP Address
---------------------------
    0   172.16.25.51

Enter command ('h' for help): q

----------------------------------------------------------------------------
 Item  Dev   Device Name      Model   Ch   Location          Curr.   Rec.
       Num                                                   FW Rev. FW Rev.
----------------------------------------------------------------------------

  -- No 1553 Devices Found --

Enter command ('h' for help): h

    h    display (h)elp screen
    d    display all (d)evices
    i    display (i)nformation about a device
    f    update the (f)irmware of a device
    t    update the (t)x Inhibit / BC Disable Config of a device
    r    manage (R)emote IP Addresses
    u    save config file and re-scan and (u)pdate device list
    q    (q)uit


Enter command ('h' for help): u
```

Figure 47. Save Configuration file

The final step is to assign an LDN to each channel on the ABD. This is done by entering ‘**s**’ from the prompt for **(s)et logical device number of a device**. This causes the card manager to ask for which item number, and what is the LDN number to use for the channel. To switch between ARINC 429 and MIL-STD-1553 devices, the user must close and then reopen **ddccm**.

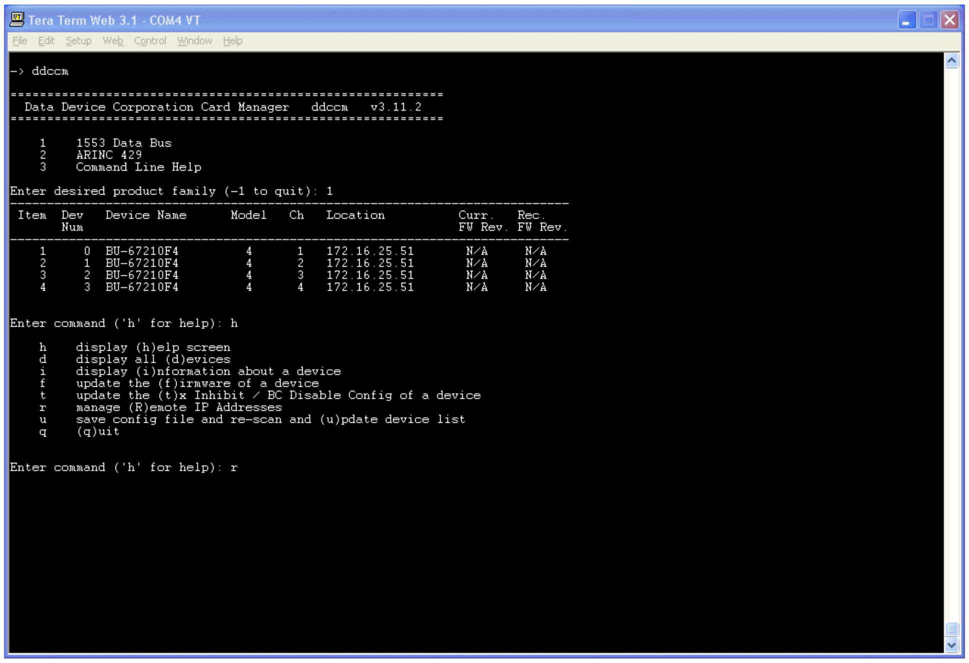

Tera Term Web 3.1 - COM4 VT
File Edit Setup Web Control Window Help

```
    q    (q)uit and return to previous menu


---------------------------
 Item   IP Address
---------------------------
(No Entries)

Enter command ('h' for help): a

Enter the new IP Address: 172.16.25.51

---------------------------
 Item   IP Address
---------------------------
    0   172.16.25.51

Enter command ('h' for help): q

-----------------------------------------------------------------------------
 Item  Dev   Device Name      Model   Ch   Location          Curr.   Rec.
       Num                                                   FW Rev. FW Rev.
-----------------------------------------------------------------------------

  -- No 1553 Devices Found --

Enter command ('h' for help): h

    h    display (h)elp screen
    d    display all (d)evices
    i    display (i)nformation about a device
    f    update the (f)irmware of a device
    t    update the (t)x Inhibit / BC Disable Config of a device
    r    manage (R)emote IP Addresses
    u    save config file and re-scan and (u)pdate device list
    q    (q)uit


Enter command ('h' for help): u
-----------------------------------------------------------------------------
 Item  Dev   Device Name      Model   Ch   Location          Curr.   Rec.
       Num                                                   FW Rev. FW Rev.
-----------------------------------------------------------------------------
    1     0  BU-67210F4          4      1  172.16.25.51        N/A     N/A
    2     1  BU-67210F4          4      2  172.16.25.51        N/A     N/A
    3     2  BU-67210F4          4      3  172.16.25.51        N/A     N/A
    4     3  BU-67210F4          4      4  172.16.25.51        N/A     N/A


Enter command ('h' for help):
```

Figure 48. Listing Found Devices

8.3.2 Removing ABD from Card Manager

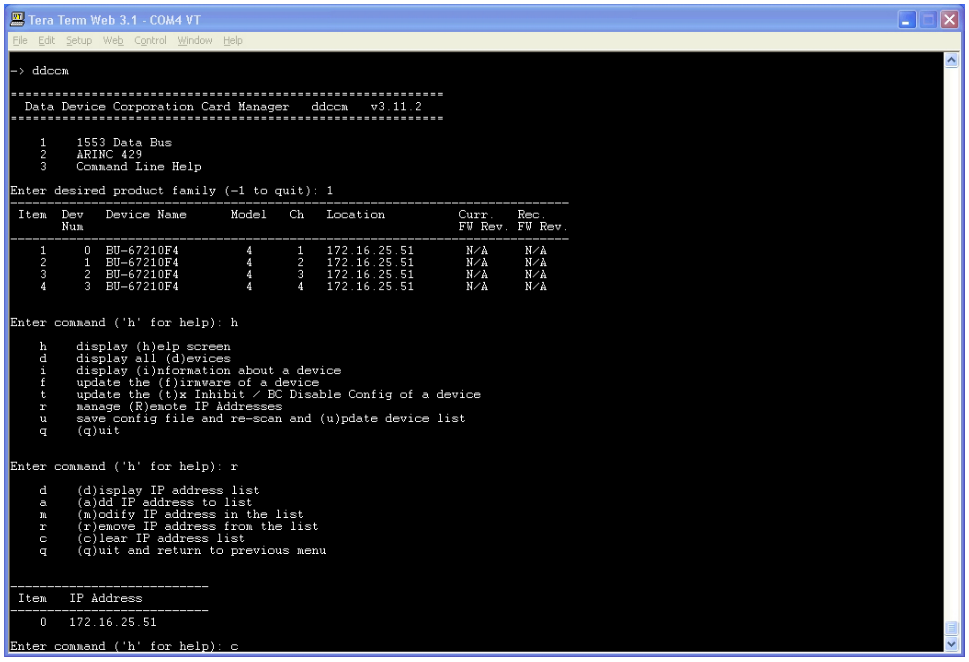
The DDC Card Manager (**ddccm**) can be used to remove the ABD's IP Address from the searchable IP Address list as well. To remove the IP Address, open the DDC Card Manager by typing **ddccm** at the prompt with superuser privileges. Select either MIL-STD-1553 or ARINC 429 from the first menu selection. Enter '**r**' to select **(r)emove IP addres from the list**.


Tera Term Web 3.1 - COM4 VT
File Edit Setup Web Control Window Help

```
-> ddccm

==========================================================
  Data Device Corporation Card Manager   ddccm   v3.11.2
==========================================================

    1    1553 Data Bus
    2    ARINC 429
    3    Command Line Help

Enter desired product family (-1 to quit): 1
----------------------------------------------------------------------------
 Item  Dev   Device Name      Model   Ch   Location          Curr.    Rec.
       Num                                                   FW Rev.  FW Rev.
----------------------------------------------------------------------------
    1     0  BU-67210F4         4      1   172.16.25.51       N/A      N/A
    2     1  BU-67210F4         4      2   172.16.25.51       N/A      N/A
    3     2  BU-67210F4         4      3   172.16.25.51       N/A      N/A
    4     3  BU-67210F4         4      4   172.16.25.51       N/A      N/A


Enter command ('h' for help): h

    h    display (h)elp screen
    d    display all (d)evices
    i    display (i)nformation about a device
    f    update the (f)irmware of a device
    t    update the (t)x Inhibit / BC Disable Config of a device
    r    manage (R)emote IP Addresses
    u    save config file and re-scan and (u)pdate device list
    q    (q)uit


Enter command ('h' for help): r
```

Figure 49. Removing IP Address

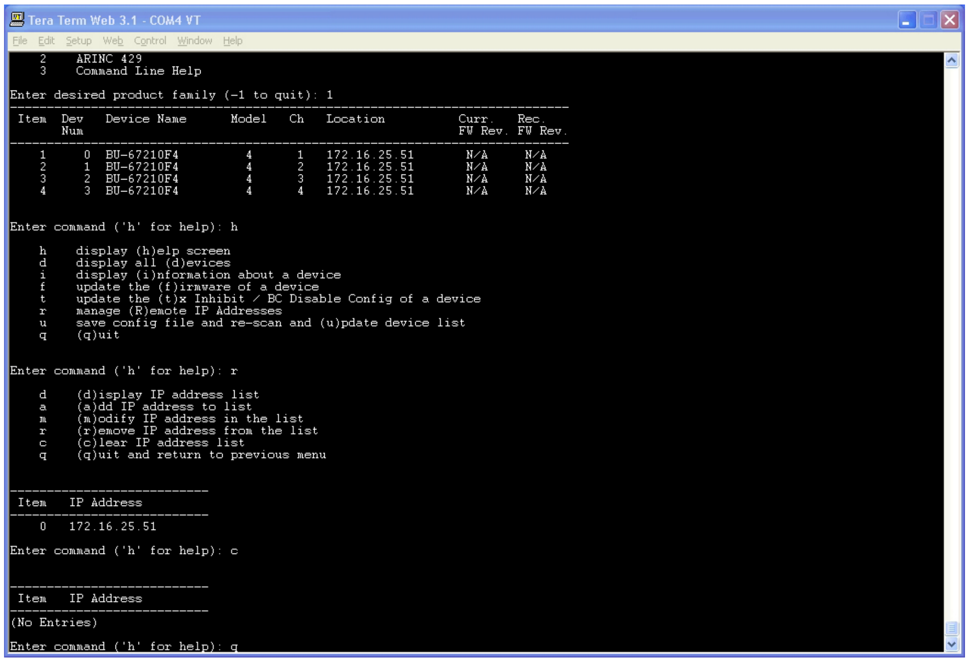
After selecting ‘**r**’ to enter the mange **(R)emote IP Addresses** section of the card manager, select ‘**c**’ to **(c)lear IP address list**. This will remove all IP Addresses from the DDC Card Managers list. If the user wishes to remove a particular IP Address the ‘**r**’ selection can be used.


Tera Term Web 3.1 - COM4 VT
File Edit Setup Web Control Window Help

```
-> ddccm

==========================================================
  Data Device Corporation Card Manager   ddccm   v3.11.2
==========================================================

    1    1553 Data Bus
    2    ARINC 429
    3    Command Line Help

Enter desired product family (-1 to quit): 1
----------------------------------------------------------------------------
 Item  Dev   Device Name      Model   Ch   Location          Curr.    Rec.
       Num                                                   FW Rev.  FW Rev.
----------------------------------------------------------------------------
    1     0  BU-67210F4         4      1   172.16.25.51       N/A      N/A
    2     1  BU-67210F4         4      2   172.16.25.51       N/A      N/A
    3     2  BU-67210F4         4      3   172.16.25.51       N/A      N/A
    4     3  BU-67210F4         4      4   172.16.25.51       N/A      N/A


Enter command ('h' for help): h

    h    display (h)elp screen
    d    display all (d)evices
    i    display (i)nformation about a device
    f    update the (f)irmware of a device
    t    update the (t)x Inhibit / BC Disable Config of a device
    r    manage (R)emote IP Addresses
    u    save config file and re-scan and (u)pdate device list
    q    (q)uit


Enter command ('h' for help): r

    d    (d)isplay IP address list
    a    (a)dd IP address to list
    m    (m)odify IP address in the list
    r    (r)emove IP address from the list
    c    (c)lear IP address list
    q    (q)uit and return to previous menu


---------------------------
 Item   IP Address
---------------------------
    0   172.16.25.51

Enter command ('h' for help): c
```

Figure 50. Clearing IP Address

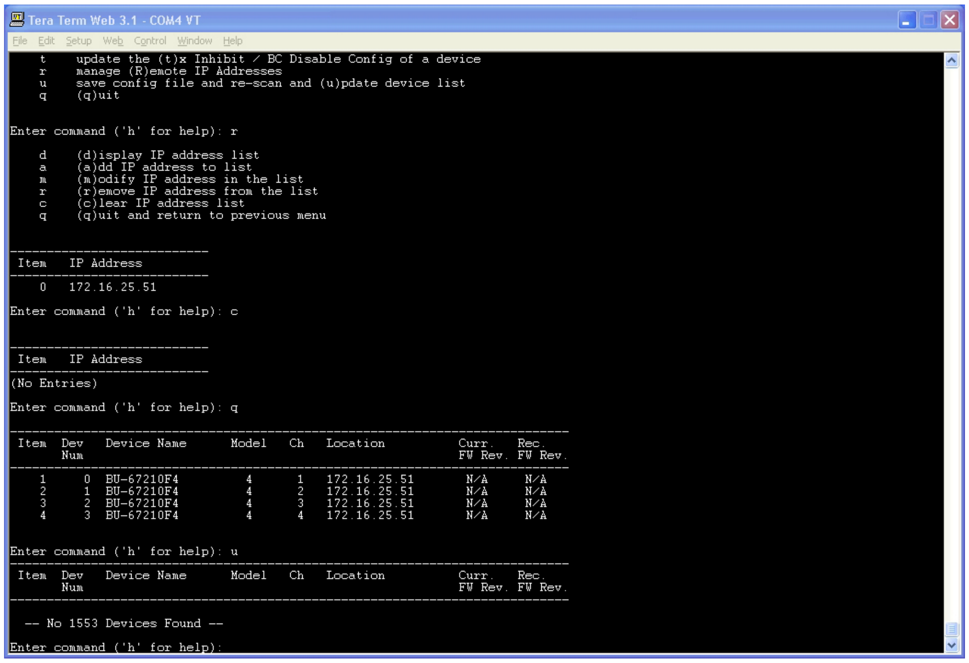
After removing the IP Address from the list, select ‘**q**’ to exit back to Device List portion of the DDC Card Manager.


Tera Term Web 3.1 - COM4 VT
File Edit Setup Web Control Window Help

```
    2    ARINC 429
    3    Command Line Help

Enter desired product family (-1 to quit): 1
-----------------------------------------------------------------------------
 Item  Dev   Device Name      Model   Ch   Location          Curr.   Rec.
       Num                                                   FW Rev. FW Rev.
-----------------------------------------------------------------------------
    1     0  BU-67210F4         4      1   172.16.25.51       N/A     N/A
    2     1  BU-67210F4         4      2   172.16.25.51       N/A     N/A
    3     2  BU-67210F4         4      3   172.16.25.51       N/A     N/A
    4     3  BU-67210F4         4      4   172.16.25.51       N/A     N/A


Enter command ('h' for help): h

    h    display (h)elp screen
    d    display all (d)evices
    i    display (i)nformation about a device
    f    update the (f)irmware of a device
    t    update the (t)x Inhibit / BC Disable Config of a device
    r    manage (R)emote IP Addresses
    u    save config file and re-scan and (u)pdate device list
    q    (q)uit


Enter command ('h' for help): r

    d    (d)isplay IP address list
    a    (a)dd IP address to list
    m    (m)odify IP address in the list
    r    (r)emove IP address from the list
    c    (c)lear IP address list
    q    (q)uit and return to previous menu


---------------------------
 Item   IP Address
---------------------------
    0   172.16.25.51

Enter command ('h' for help): c


---------------------------
 Item   IP Address
---------------------------
(No Entries)

Enter command ('h' for help): q
```

Figure 51. Returning to Previous Menu

To update the device list, enter ‘**u**’ at the prompt, and the DDC Card Manager will no longer see any ABD devices.


Tera Term Web 3.1 - COM4 VT
File Edit Setup Web Control Window Help

```
    t    update the (t)x Inhibit / BC Disable Config of a device
    r    manage (R)emote IP Addresses
    u    save config file and re-scan and (u)pdate device list
    q    (q)uit


Enter command ('h' for help): r

    d    (d)isplay IP address list
    a    (a)dd IP address to list
    m    (m)odify IP address in the list
    r    (r)emove IP address from the list
    c    (c)lear IP address list
    q    (q)uit and return to previous menu


---------------------------
 Item   IP Address
---------------------------
    0   172.16.25.51

Enter command ('h' for help): c


---------------------------
 Item   IP Address
---------------------------
(No Entries)

Enter command ('h' for help): q

-----------------------------------------------------------------------------
 Item  Dev   Device Name      Model   Ch   Location           Curr.    Rec.
       Num                                                    FW Rev.  FW Rev.
-----------------------------------------------------------------------------
    1     0  BU-67210F4         4      1   172.16.25.51        N/A      N/A
    2     1  BU-67210F4         4      2   172.16.25.51        N/A      N/A
    3     2  BU-67210F4         4      3   172.16.25.51        N/A      N/A
    4     3  BU-67210F4         4      4   172.16.25.51        N/A      N/A


Enter command ('h' for help): u
-----------------------------------------------------------------------------
 Item  Dev   Device Name      Model   Ch   Location           Curr.    Rec.
       Num                                                    FW Rev.  FW Rev.
-----------------------------------------------------------------------------

  -- No 1553 Devices Found --

Enter command ('h' for help):
```

Figure 52. Updating Device List

AceXtreme Bridge Device

9 UPGRADING AND RECOVERING THE DEVICE

The AceXtreme Bridge Device is capable of In-Field upgrading, and recovery. In-Field upgrading allows the ABD to be updated to DDC latest Drivers and Library without having to ship the unit back to DDC. The recovery mode can be used to flash a backup image of the original configuration onto the AceXtreme Bridge Device . Care should be taken when using the recovery mode as all settings, and data will be erased after putting the device into Recovery mode.

9.1 Setting device signal states

The AceXtreme Bridge Device has three modes of operation, Configuration Mode, Run Mode and Recovery mode. To select a mode of operation the DDC-78053 cable is required. This cable allows the user to select between Run Mode and Configuration Mode. If the cable is not connected, the device will start in Run Mode.

9.1.1 Run Mode Signal State

The AceXtreme Bridge Device will be in Run Mode when the DDC-78053 Cable is not connected to the ABD. When the DDC-78053 Cable is connected to the ABD, the AceXtreme Bridge Device will be in Run Mode when the supplied BNC short connector is not attached to J2 of the DDC-78053 cable.

9.1.2 Configuration Mode Signal State

To put the AceXtreme Bridge Device into Configuration Mode, connect the BNC short connector to J2. Critical ABD settings (such as IP address and Enabling/Disabling Network Services) may be modified on the AceXtreme Bridge Device .

9.1.3 Recovery Mode Signal State

Recovery Mode on the AceXtreme Bridge Device may be used to flash a backup image onto the primary partition of the ABD. All data and settings will be lost when using Recovery mode. To put the AceXtreme Bridge Device into Recovery Mode, Pin 47 must be grounded before applying power. Available ground pins on J1 are pins 1, 2, 3, 4, 11, 18, 26,33,46 and 49.

See the BU-6711[5/6/8]W Hardware Manual for more information.

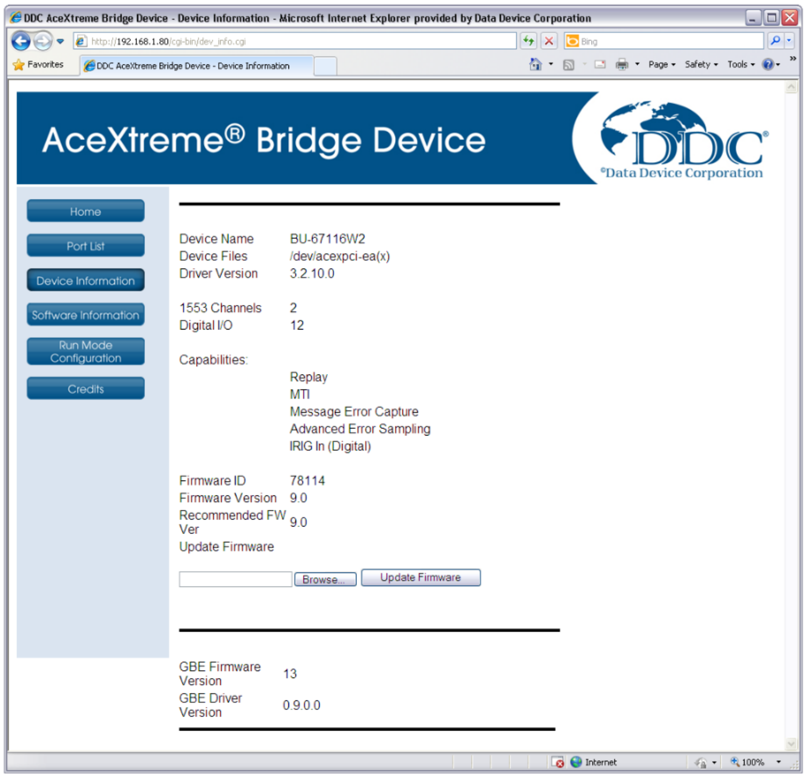
9.2 Recovery Mode

When necessary, Recovery Mode on the AceXtreme Bridge Device may be used to put the ABD back into a know default state (factory settings). This is done by grounding Pin 47 on J1 and booting the ABD. The AceXtreme Bridge Device will then automatically boot into the backup flash and start the recovery process. Once completed, power off the ABD and disconnect the J1 pin from ground (a complete power cycle is required for the update to complete). Restart the ABD, and it will now boot up in its default configuration from the factory.

9.3 HTTP (Web) Update Method

To update the DDC SDKs and drivers the Web Server may be used.   The Software Information page on the AceXtreme Bridge Device displays the installed MIL-STD-1553 and ARINC-429 SDK library versions. The Browse and Upload button can be used to upload a new library and / or driver provided by DDC.

Device Firmware can also be updated in a similar fashion by the Update Firmware section on the Device Information Tab.


DDC AceXtreme Bridge Device - Device Information - Microsoft Internet Explorer provided by Data Device Corporation
http://192.168.1.80/cgi-bin/dev_info.cgi
Bing
Favorites
DDC AceXtreme Bridge Device - Device Information
Page
Safety
Tools
AceXtreme® Bridge Device
DDC
Data Device Corporation
Home
Port List
Device Information
Software Information
Run Mode Configuration
Credits
Browse...
Update Firmware
Internet
100%

Device Name	BU-67116W2
Device Files	/dev/acexpci-ea(x)
Driver Version	3.2.10.0
1553 Channels	2
Digital I/O	12
Capabilities:	
	Replay
	MTI
	Message Error Capture
	Advanced Error Sampling
	IRIG In (Digital)
Firmware ID	78114
Firmware Version	9.0
Recommended FW Ver	9.0
Update Firmware	
GBE Firmware Version	13
GBE Driver Version	0.9.0.0

Figure 53. AceXtreme Bridge Device Internal Directory Structure

Data Device Corporation

Leadership Built on Over 50 Years of Innovation

Military | Commercial Aerospace | Space | Industrial

Data Device Corporation (DDC) is the world leader in the design and manufacture of high-reliability data bus products, motion control, and solid-state power controllers for aerospace, defense, and industrial automation applications. For more than 50 years, DDC has continuously advanced the state of high-reliability data communications and control technology for MIL-STD-1553, ARINC 429, AFDX®, Synchro/Resolver interface, and Solid-State Power Controllers with innovations that have minimized component size and weight while increasing performance. DDC offers a broad product line consisting of advanced data bus technology for Fibre Channel networks; MIL-STD-1553 and ARINC 429 Data Networking cards, components, and software; Synchro/Resolver interface components; and Solid-State Power Controllers and Motor Drives.

Product Families

Data Bus | Synchro/Resolver Digital Conversion| Power Controllers | Motor Controllers

DDC is a leader in the development, design, and manufacture of highly reliable and innovative military data bus solutions. DDC's Data Networking Solutions include MIL-STD-1553, ARINC 429, AFDX®, Ethernet and Fibre Channel. Each Interface is supported by a complete line of quality MIL-STD-1553 and ARINC 429 commercial, military, and COTS grade cards and components, as well as software that maintain compatibility between product generations. The Data Bus product line has been field proven for the military, commercial and aerospace markets.

DDC is also a global leader in Synchro/Resolver Solutions. We offer a broad line of Synchro/Resolver instrument-grade cards, including angle position indicators and simulators. Our Synchro/Resolver-to-Digital and Digital-to-Synchro/Resolver microelectronic components are the smallest, most accurate converters, and also serve as the building block for our card-level products. All of our Synchro/Resolver line is supported by software, designed to meet today's COTS/MOTS needs. The Synchro/Resolver line has been field proven for military and industrial applications, including radar, IR, and navigation systems, fire control, flight instrumentation/simulators, motor/motion feedback controls and drivers, and robotic systems.

As the world's largest supplier of Solid-State Power Controllers (SSPCs) and Remote Power Controllers (RPCs), DDC was the first to offer commercial and fully-qualified MIL-PRF-38534 and Class K Space-level screening for these products. DDC's complete line of SSPC and RPC boards and components support real-time digital status reporting and computer control, and are equipped with instant trip, and true I^2T wire protection. The SSPC and RPC product line has been field proven for military markets, and are used in the Bradley fighting vehicles and M1A2 tank.

DDC is the premier manufacturer of hybrid motor drives and controllers for brush, 3-phase brushless, and induction motors operating from 28 Vdc to 270 Vdc requiring up to 18 kilowatts of power. Applications range from aircraft actuators for primary and secondary flight controls, jet or rocket engine thrust vector control, missile flight controls, to pumps, fans, solar arrays and momentum wheel control for space and satellite systems.

Certifications

Data Device Corporation is ISO 9001: 2008 and AS 9100, Rev. C certified.

DDC has also been granted certification by the Defense Supply Center Columbus (DSCC) for manufacturing Class D, G, H, and K hybrid products in accordance with MIL-PRF-38534, as well as ESA and NASA approved.

Industry documents used to support DDC's certifications and Quality system are: AS9001 OEM Certification, MIL-STD-883, ANSI/NCSL Z540-1, IPC-A-610, MIL-STD-202, JESD-22, and J-STD-020.

The first choice for more than 50 years—DDC
DDC is the world leader in the design and manufacture of high reliability data interface products, motion control, and solid-state power controllers for aerospace, defense, and industrial automation.

Inside the U.S. - Call Toll-Free 1-800-DDC-5757

Headquarters and Main Plant
105 Wilbur Place, Bohemia, NY 11716-2426
Tel: (631) 567-5600 Fax: (631) 567-7358
Toll-Free, Customer Service: 1-800-DDC-5757

Web site: www.ddc-web.com

Data Device Corporation

Outside the U.S. - Call 1-631-567-5700

United Kingdom: DDC U.K., LTD
James House, 27-35 London Road, Newbury,
Berkshire RG14 1JL, England
Tel: +44 1635 811140 Fax: +44 1635 32264

France: DDC Electronique
10 Rue Carle-Hebert
92400 Courbevoie France
Tel: +33-1-41-16-3424 Fax: +33-1-41-16-3425

Germany: DDC Elektronik GmbH
Triebstrasse 3, D-80993 München, Germany
Tel: +49 (0) 89-15 00 12-11
Fax: +49 (0) 89-15 00 12-22

Japan: DDC Electronics K.K.
Dai-ichi Magami Bldg, 8F, 1-5, Koraku 1-chome,
Bunkyo-ku, Tokyo 112-0004, Japan
Tel: 81-3-3814-7688 Fax: 81-3-3814-7689
Web site: www.ddcjapan.co.jp

Asia: Data Device Corporation - RO Registered in Singapore
Blk-327 Hougang Ave 5 #05-164
Singapore 530327
Tel: +65 6489 4801