DAS-429UNET/RTx

Avionics Communication Device

User's Manual

Avionics Communication Device

EXCALIBUR SYSTEMS

One Step Ahead

311 Meacham Ave ♦ Elmont ♦ NY 11003 ♦ tel. (516) 327-0000 ♦ fax (516) 327-4645
e-mail: excalibur@mil-1553.com website: www.mil-1553.com

Table of Contents

Figures

Tables

1 Introduction

Chapter 1 provides an overview of the *UNET*.

Caution: Make sure there is no I/O communication while disconnecting any of the cables. Connecting or disconnecting the cables during communication can seriously damage the *DAS-429UNET/RTx*.

1.1 Overview

The *DAS-429UNET/RTx* is an intelligent, ARINC 429 interface device. Its small size and ability to interface through USB or Ethernet interfaces make it a complete solution for developing, testing and performing system simulation of the ARINC 429 bus, both in the lab and in the field.

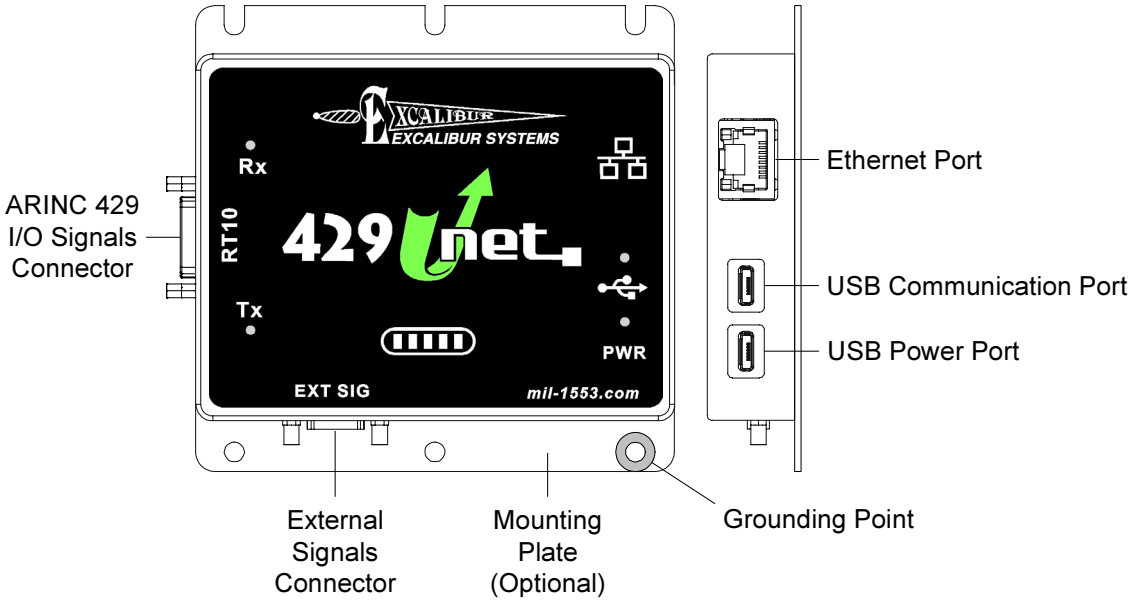


Figure 1-1 ***DAS-429UNET/RTx***

Note:

- For an explanation of the LED indicators, see **4.2 LED Indicators** on page 4-4.
- For ordering information see **Chapter 8: Ordering Information.**

The *DAS-429UNET/RTx* supports up to ten ARINC 429 channels in any combination of transmitters and receivers. Each of these channels feature error injection and detection capabilities.

The receive channels allow for the storage of all selected Labels with status and Time Tag information appended to each word. The receivers allow for filtering and multi-storage modes of Data Words.

The transmit channels operate via a transmit 'instruction stack' which allows scheduling of data transmissions and reduces the need for host computer intervention.

In addition, this device provides IRIG B input and 8 Discrete I/O signals. The Discrete channel can record changes in the input Discrete with an associated Time Tag via a built-in FIFO. Output Discretes are open collector, capable of handling up to 32V with a maximum sink current of 100 mA each.

You can set for each Discrete:

- Whether it is input or output
- Input voltage level: TTL (0 – 5V) or Avionics (0 – 32V)
- Whether to enable or disable debounce on inputs

Multiple *UNET*s can operate via USB ports on the same computer. In addition, multiple units can operate on the same network, by programming each one with a unique IP address, and can be accessed from any computer on the network.

The *DAS-429UNET/RTx* shares its API with the entire RTx family so that applications currently running on our PCIe, PCI, ExpressCard, or PCMCIA cards, will run without change on this device.

1.1.1 Battery Option

The *DAS-429UNET/RTx* is available with an internal rechargeable battery. This allows the user to operate the *DAS-429UNET/RTx* without the need for any external power source. The internal battery can also be used as an auxiliary power source that can provide the additional power required for high data transmission conditions where external power sources are limited, for example, when the external power is provided by a host laptop computer. In this case, the internal battery can be used to minimize the amount of power drawn from the laptop's battery, which could be a critical requirement in the field where an external power supply is not readily available. For more information, see **4.4.1 Battery Information (Battery Option Only)** on page 4-13.

1.2 Block Diagram

Figure 1-2 shows a block diagram of the *DAS-429UNET/RTx*. For information on connecting the cables, see **2.2 Connecting the Cables** on page 2-2.

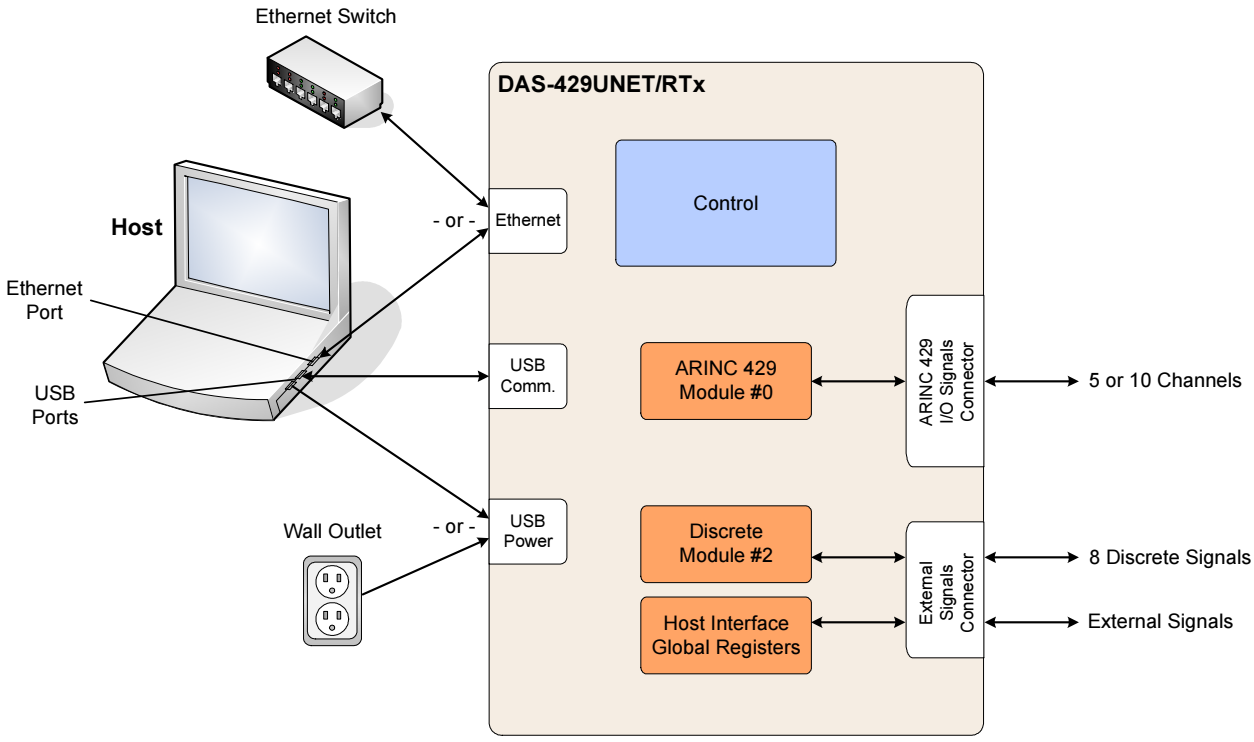


Figure 1-2 *DAS-429UNET/RTx* Block Diagram

1.3 Product Features

General Features
- 5 or 10 ARINC 429 channels, each programmable as receive or transmit
- Memory mapped, 64K x 8 dual-port RAM
- Programmable Hardware Trigger Output
- Interrupt and Polling modes of operation
- Programmable (per channel):
- Buffer size
- Bit rates (Hi, Lo, Programmable)
- Parity (Odd, Even, On, Off)
- 8 Discrete I/O signals
- Smart power management
- Power source: Computer USB ports, USB power supply
- Mechanical configuration options:
- Mounting plate
- Hard-wired ARINC 429 I/O flat cable or panel mounted connector

Transmit Channel Features
- **Transmit Modes**
- Interblock Time
- Data Rate
- FIFO Data
- **Transmit Features**
- Transmit Sync Time (between Words)
- One-shot, N-times or Loop transmission
- **Transmit Error Injection per Block**
- Bit Count Hi/Lo
- Sync Time
- Stretch Bit
- Bit Rate (frequency)
- Parity

Receive Channel Features
- **Receive Modes**
- Look-up Table
- Sequential per channel
- Sequential Merge Mode (stores data from all Receive channels in one buffer)
- **Receive Features**
- Receive Features
- Word Status Tagging
- Word Time Tagging (32-bit) or IRIG B Time Tagging (64-bit)
- IRIG B input (standard IRIG B120 serial time code)
- Label/Data Filtering
- Start Triggers
- Receive Error Count per channel
- Receive Count Interval Trigger
- **Receive Error Detection per Word**
- Bit Count
- Sync Time
- Parity
- Bit Coding Error

Physical Characteristics
- Dimensions: 98.5mm x 76mm x 18mm (not including connectors)
- Weight (/RT10 basic configuration, w/o battery): 180g

Software Support
- *Software Tools*: Advanced API with C source code. The *Software Tools* are available for several operating systems. See the Downloads section of our website.
- *Exalt Plus*: Excalibur Analysis Laboratory Tools for Windows (optional)

Operating Environment
- Operating Temp: -40° to + 75°C
- Humidity: 5% - 90% noncondensing
- MTBF (/RT10): 157,500 hours at 25°C, G_F, S217F

Host Interface
- Selectable USB 2.0 or 100Mbps Ethernet
- Power for /RT5 (max.): +5V @ 750 mA
- Power for /RT10 (max.): +5V @ 1000 mA

1.4 Supporting Software

The *UNET* has the following supporting software:

- *Excalibur Software Tools* – Advanced API functions written in C language that enable you to write application and diagnostic programs. For more information, see **Chapter 3: Developing Applications**.
- *Mystic* – A Windows test and simulation program for ARINC 429 bus communications data. *Mystic* can be from our website. Go to www.mil-1553.com, click **Downloads**, and then click **Applications**.
- *Discrete Generator* – A Windows program that enables you to configure the Discrete channels on your Discrete module, run the module and view the status of incoming Discretes. You can also configure and monitor interrupts based on status of each Discrete channel. For more information, see **2.8 Running the Discrete Generator** on page 2-29.
- *Exalt Plus* (***Ex****calibur* ***A****nalysis* ***L****aboratory* ***T****ools*) – An andvanced Windows program that enables you to monitor and analyze bus activity in real-time; to record bus activity that you can replay later (even when no modules are present and the computer is not connected to the bus); and to simulate bus activity by transmitting data over the bus. This program is an optional add-on to your purchase. For more information contact an Excalibur sales representative. See **1.6 Technical Support** on page 1-7.

1.5 Discrete Channel Information

The following sections describe the Discrete voltage/current levels and configuration.

Discrete Channel Configuration

The input voltage level and debounce setting for each input Discrete are set in the Discretes Configuration Registers, see **Discretes Configuration Registers** on page 7-12, or via the *M4KDiscrete Software Tools*, see the *M4KDiscrete Software Tools Programmer's Reference*.

The following sections provide more information about input and output Discretes.

Input Discrete Voltage Levels

Input Discretes can operate at either TTL or Avionics voltage levels, as defined in the following table.

Type	Voltage		Value
TTL	0V – 0.8V	=	Logic 0
	2V – 5V	=	Logic 1
Avionics	0V – 3V	=	Logic 0
	7.5V – 32V	=	Logic 1

Table 1-1 Input Discrete Voltages

In the standard *UNET*, the voltage level must be set externally by pulling up the Discrete I/O line to the desired voltage level, 5V for TTL or 7.5V – 32V for Avionics. The recommended value for the pull up resistors is 10K Ohms. When the line is not grounded, the recorded Discrete value will be 1 and when it is grounded the value will be 0.

A custom *UNET* can be ordered with the voltage pulled up internally to the desired voltage level. For more information, contact our Sales Department. See **1.6 Technical Support** on page 1-7.

Note: The voltage level of each Discrete must be set in the Discretes Configuration Registers, see **Discretes Configuration Registers** on page 7-12, or via the *M4KDiscrete Software Tools*, see the *M4KDiscrete Software Tools Programmer's Reference*. If a Discrete is set to Avionics voltage level and the voltage is (incorrectly) pulled up to 5V, the Discrete will always record a value of 0.

Input Discrete Debounce Setting

Each input Discrete can be configured with or without debounce. When debounce is activated, any change on the input Discrete will only be registered after 60 msec.

Output Discretes

All output Discretes are open collector. Each output Discrete must be pulled up to the required voltage (maximum of 32V) externally to the *UNET*. A maximum

of 100 mA can be sunk on any individual output Discrete. The external pull up resistor should be chosen carefully to ensure that the maximum sink current is not exceeded. The recommended value for the pull up resistors is 10K Ohms.

When an output Discrete is set to 0, the output Discrete will be shorted to ground. When an output Discrete is set to 1, the output Discrete will be left floating and thus pulled up to the voltage used for that output Discrete via the external pull up resistor.

1.6 Technical Support

Excalibur Systems is ready to assist you with any technical questions you may have. For technical support, click the **Technical Support** link on the **Support** page of our website: www.mil-1553.com. You can also contact us by phone. To find the location nearest you, refer to the **Contact Us** page of our website. For corrections to this manual, email our Technical Writing Department directly at: tw@mil-1553.com. Please include the name of the manual, the manual revision number (located on the last page of the manual), and the location within the manual.

2 Installation and Setup

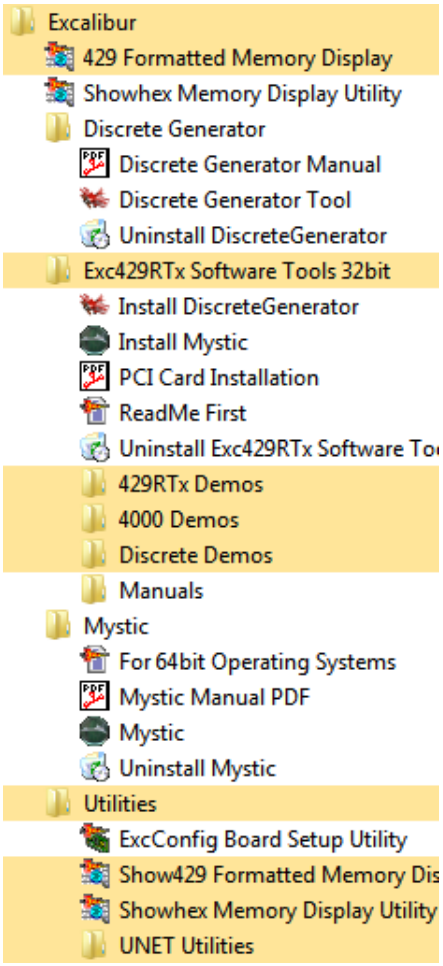
Chapter 2 provides installation and setup information for the *UNET*.

2.1 Installing the Software on the Host Computer

This section describes how to install the *UNET* software on the host computer that will be used to control the *UNET* via USB or Ethernet.

To install the *UNET* software:

1. Insert the *Excalibur Installation CD* into your computer's CD drive.
   The InstallShield Wizard is displayed.
2. Follow the on-screen instructions.
   The software is installed, and the following items are added to the Start menu:



Note: In Windows 10, all shortcuts are under the Excalibur entry in the Start menu.

2.2 Connecting the Cables

This section describes how to connect the *UNET*'s cables. The *DAS-429UNET/RTx* can communicate with host via USB or Ethernet. For cabling instructions for both configurations, see **2.2.1 Connecting the DAS-429UNET/RTx Cables** on page 2-3.

Note: When setting up the *UNET*, it is recommended to check the *UNET*'s status via its LED states. See **4.2 LED Indicators** on page 4-4.

2.2.1 Connecting the *DAS-429UNET/RTx* Cables

Cabling for USB Communication with Host

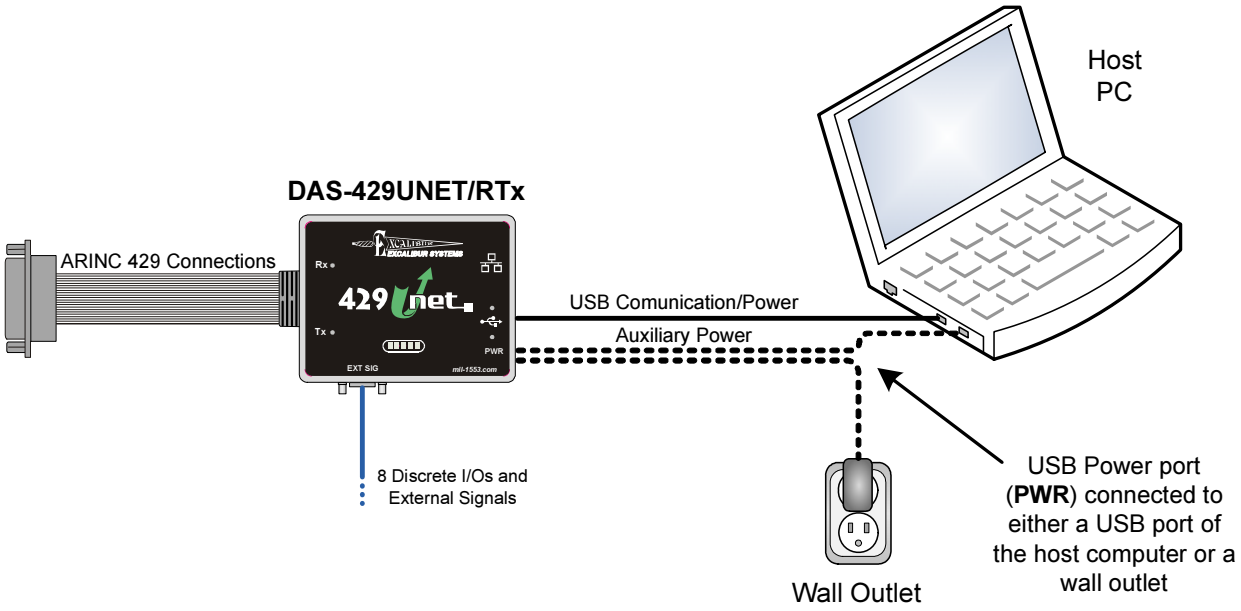


Figure 2-1 USB Communication with Host

To connect the cables:

1. Connect the ARINC 429 I/O connector of the *DAS-429UNET/RTx* to your ARINC 429 device.
2. Connect the USB Communication port ⇻ of the *UNET* to the USB port of the host computer using the Micro-B USB to Standard-A USB adapter cable provided by Excalibur. In most cases (depending on the load on the ARINC 429 bus and capabilities of the host computer), the power supported through the USB Communication port ⇻ will be enough to operate the *UNET*. If the power from the USB Communication port ⇻ is insufficient, connect the USB Power port (**PWR**) of the *UNET* to another USB port of the host computer, or to a wall outlet using the 5V USB Power Supply provided by Excalibur Systems. For power information when using an internal battery, see **4.4.1 Battery Information (Battery Option Only)** on page 4-13.

Caution: Use only the USB Power Supply provided by Excalibur Systems. Excalibur Systems is not responsible for any damage or lack of functionality when a different power supply is used.

Cabling for Ethernet Communication with Host

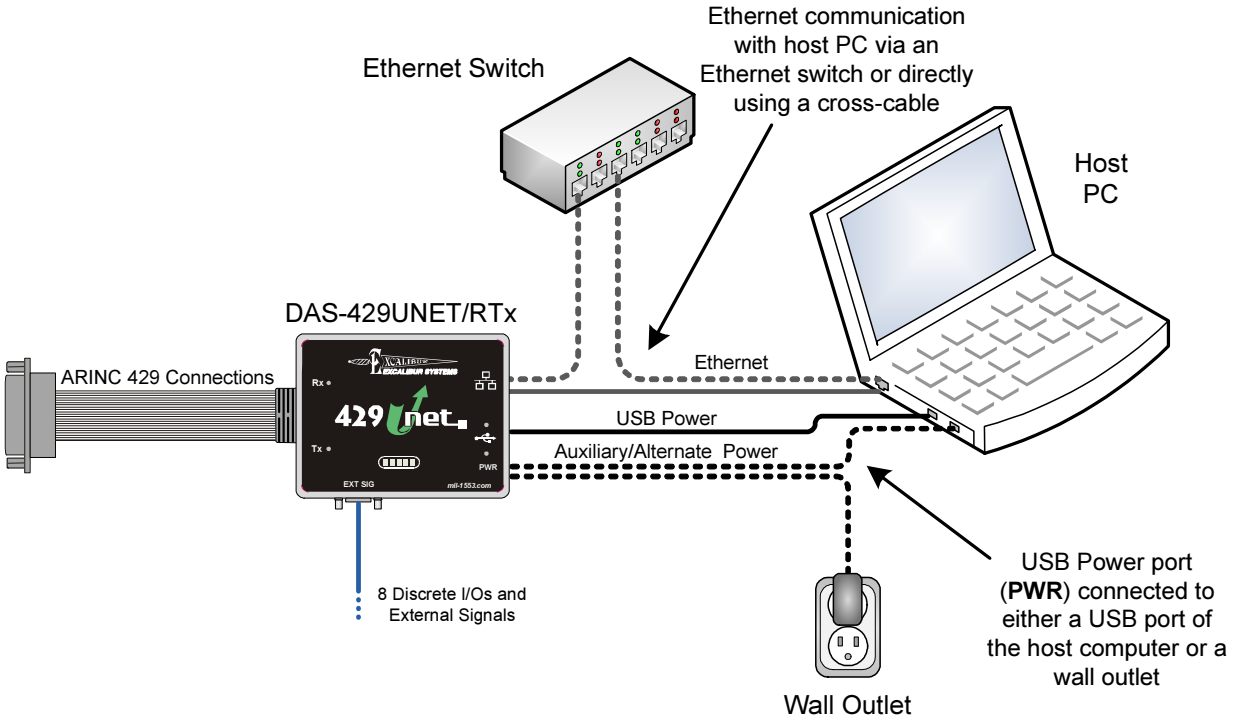


Figure 2-2 Ethernet Communication with Host

To connect the cables:

1. Connect the RJ45 port of the *DAS-429UNET/RTx* to an Ethernet switch or to the host computer via an Ethernet cable (not included).

Note: When connecting the *DAS-429UNET/RTx* directly to the host computer you will need a cross-cable, unless the computer's network card can automatically switch the Ethernet signals.

2. Connect the ARINC 429 I/O connector of the *DAS-429UNET/RTx* to your ARINC 429 device.
3. To supply power to *DAS-429UNET/RTx*, connect the USB Communication port ⇻ to the USB port of any computer. In most cases (depending on the load on the ARINC 429 bus and capabilities of the host computer), the power supported through the USB Communication port ⇻ will be enough to operate the *UNET*. If the power from the USB Communication port ⇻ is insufficient, connect the USB Power port (**PWR**) of the *UNET* to another USB port of the host computer, or to a wall outlet using the 5V USB Power Supply provided by Excalibur Systems. For power information when using an internal battery, see **4.4.1 Battery Information (Battery Option Only)** on page 4-13.

Caution: Use only the USB Power Supply provided by Excalibur Systems. Excalibur Systems is not responsible for any damage or lack of functionality when a different power supply is used.

2.3 Installing the USB Driver on the Host Computer

Note: USB driver installation is only required when using the USB interface for communication. When using the USB port for power only, USB driver installation is not required.

When you connect the *UNET* to a computer connected to the Internet, Windows automatically finds the USB driver. In the event that the computer is not connected to the Internet, install the device driver using the Device Manager.

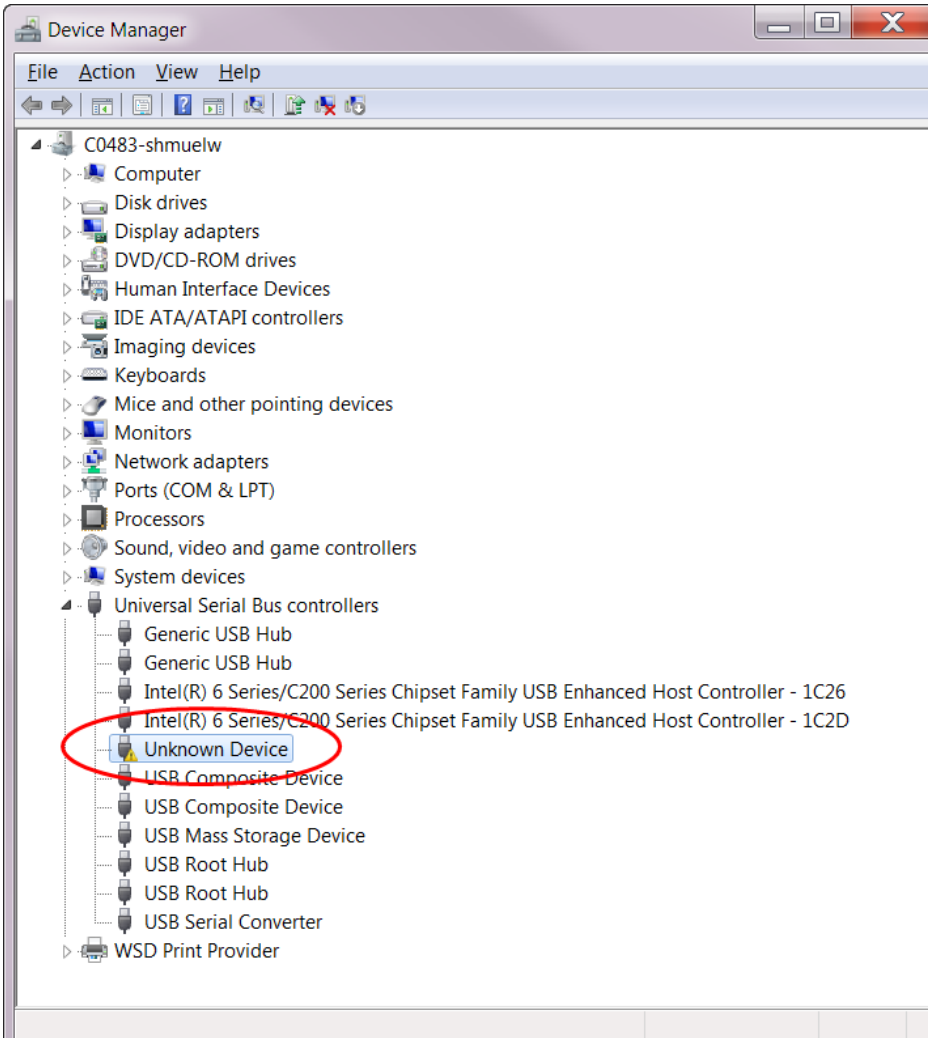
To install the USB driver manually:

1. Connect one of the USB cables to the host computer and to the USB Communication port of the *DAS-429UNET/RTx*.
2. Right-click on **My Computer**, and select **Properties** from the context menu.
3. Do one of the following:

   For Windows 7: Click **Device Manager**.

   For Windows XP: Click the **Hardware** tab, then click **Device Manager**.

   The Device Manager is displayed.
4. Expand the Universal Serial Bus controllers group of devices.



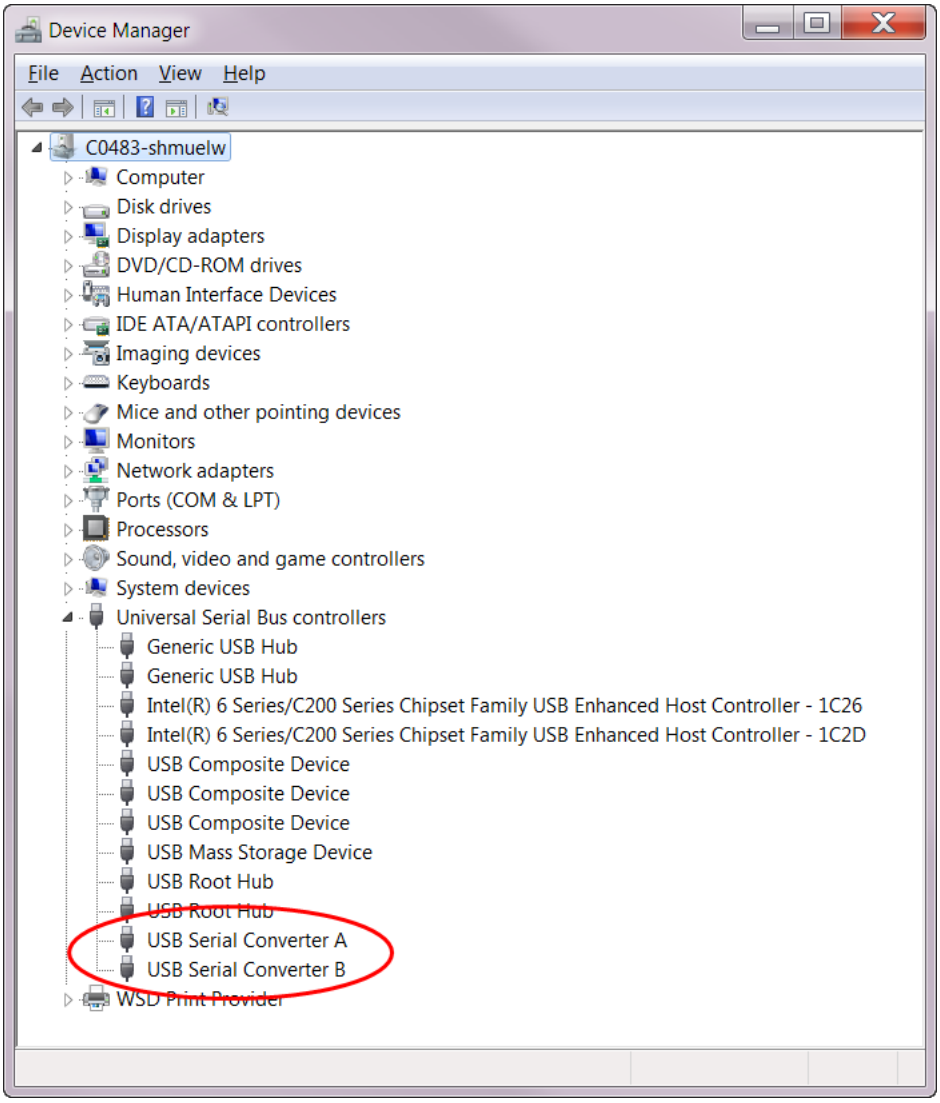
5. Right-click the unknown USB device, and select **Update Driver Software** from the context menu.
6. Choose the option to browse your computer for the driver.
7. Select the folder: **C:\Excalibur\429UNET\USB Driver**. (This path assumes the default software installation location.)
8. Click **Next**.

   The driver is installed.

Note: In the event that the installation is not successful, try connecting the USB cables to the front USB ports or to other USB ports, and reinstalling the driver. Also, do not use an extension cable during the driver installation.

9. Disconnect the USB cable(s), then reconnect the cables.
10. Open the Device Manager and verify that the USB driver was installed correctly.

    If the driver is installed correctly, USB Serial Converter A and USB Serial Converter B appear in the Universal Serial Bus controllers group.



2.4 Assigning a Device Number on the Host Computer

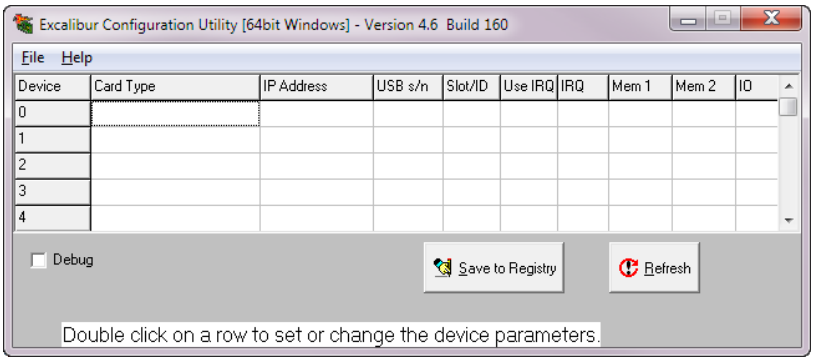
This section describes how to use the ExcConfig utility to assign a device number to the *UNET* and save the device number in the Windows System Registry. The device number is required as a parameter of several of the *Software Tools* functions.

Note: When assigning a device number to the *UNET*, the instructions differ for communicating with the *UNET* via USB or Ethernet.

To assign a device number to the *UNET*:

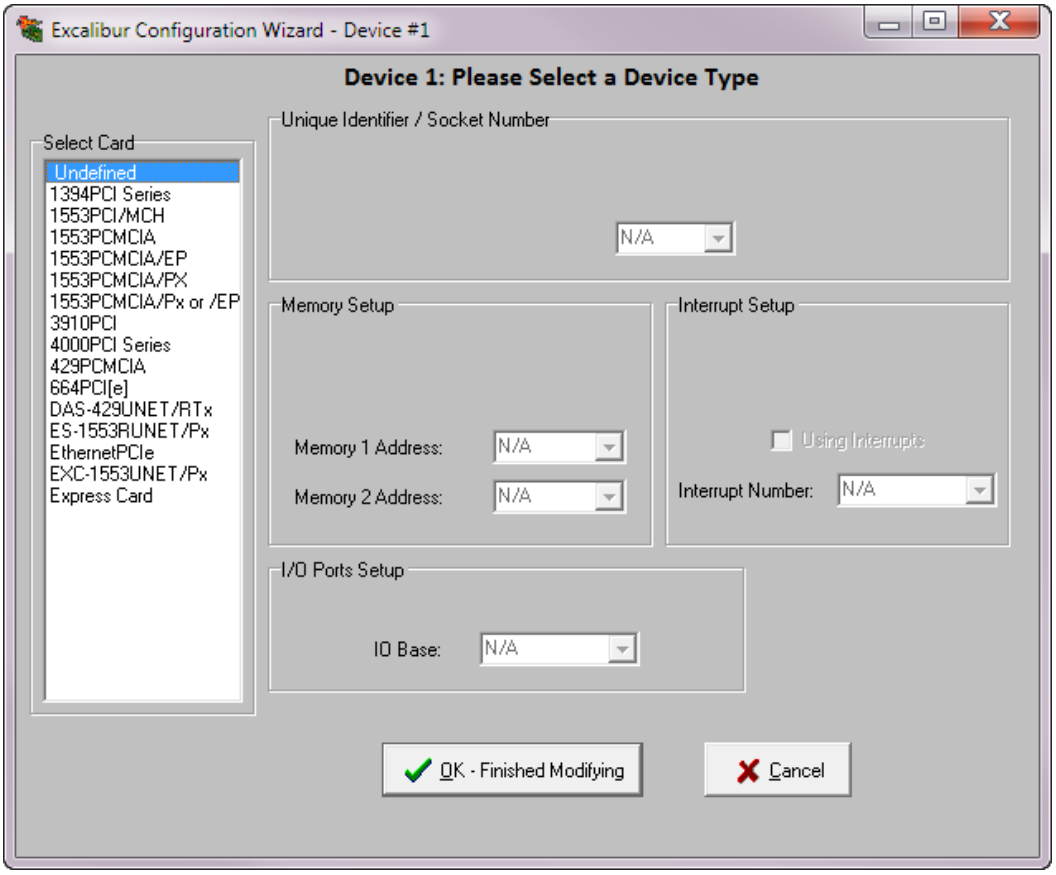
1. Run the ExcConfig: Click **Start | All Programs | Excalibur**, right-click **ExcConfig Board Setup Utility**, then select **Run as administrator**. (When using Windows XP, running as administrator is not required.)

   The ExcConfig main screen is displayed, showing all Excalibur devices saved in the Windows System Registry of this computer. Each row on the ExcConfig main screen is associated with a device number (0–15).



2. Double-click one of the cells in an empty row, for example, Device #1.

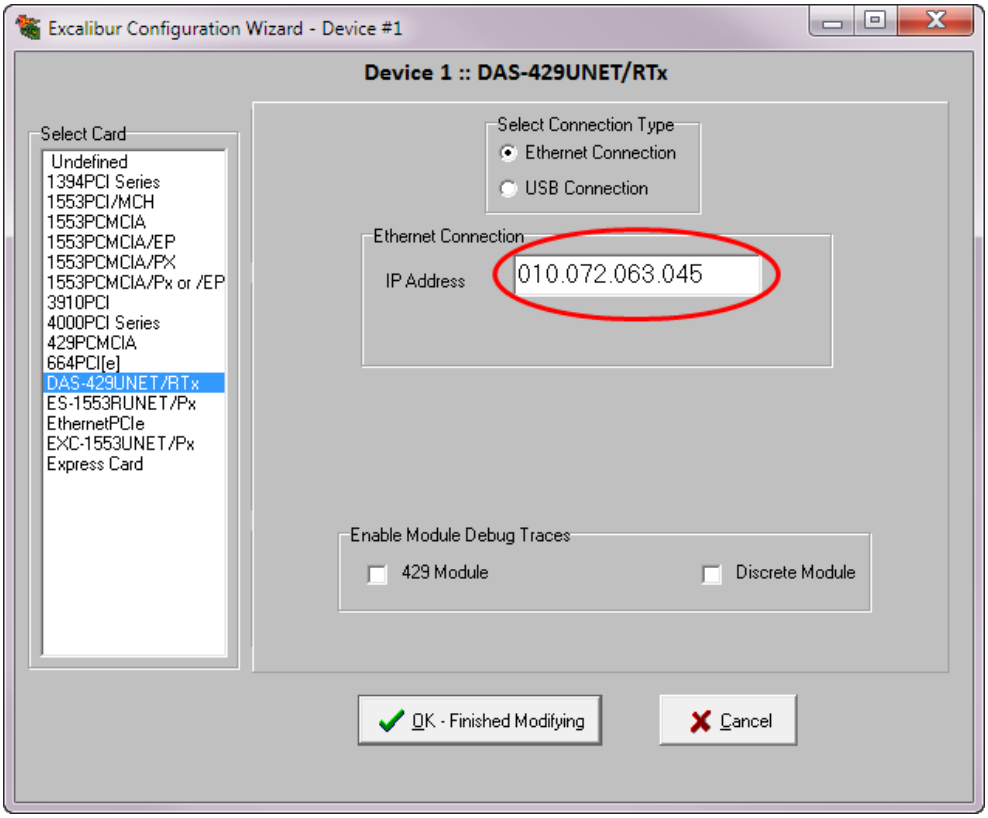
The Excalibur Configuration Wizard is displayed.



3. In the Select Card section, select **DAS-429UNET/RTx** or **ES-429RUNET/RTx**.
4. Do one of the following:
   - When using an *DAS-429UNET/RTx* and communicating with the host via Ethernet, continue with Step 5.
   - When using an *DAS-429UNET/RTx* and communicating with the host via USB, continue with Step 10.

5. In the Select Connection Type section, click the **Ethernet Connection** option button.

   The IP Address field is displayed with the default IP address of the *DAS-429UNET/RTx*.



Note: The following step is only required when there will be two or more *UNET*'s on the same network. In this case, each *UNET* must have a unique IP address. Otherwise, you can leave the default IP address.

6. When using two or more *UNET*'s on the same network, do the following: In the Ethernet Connection section, in the IP Address field, type the IP address that you want to use for the *UNET*.

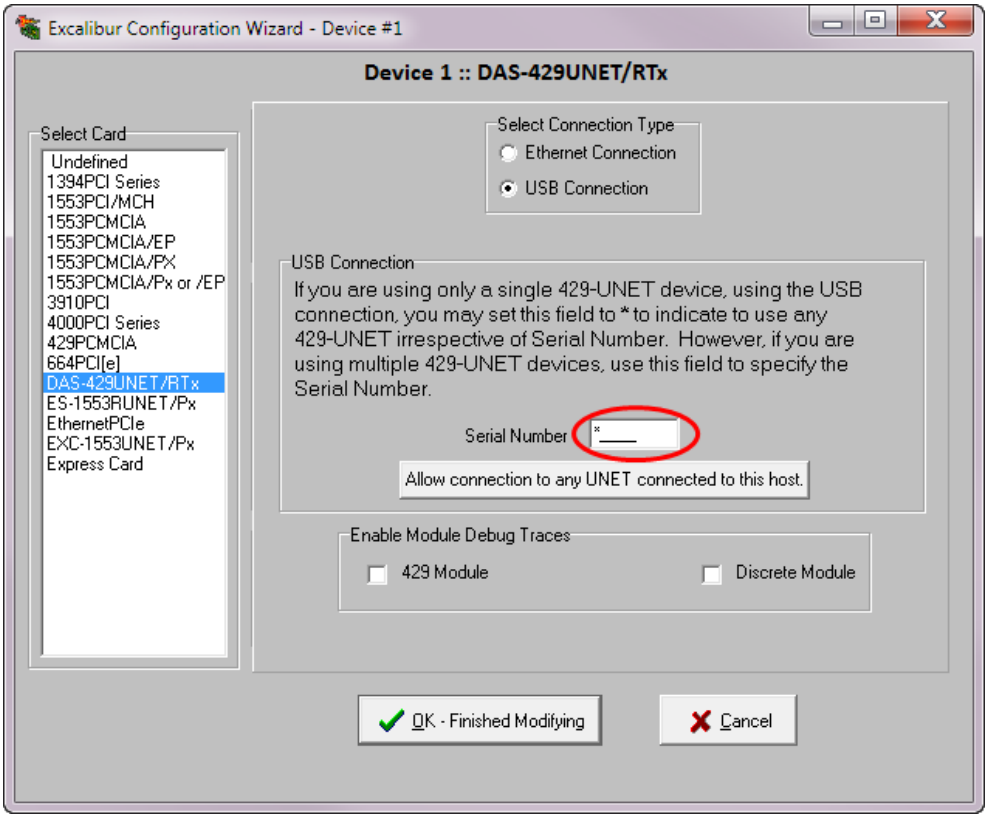
Note: ExcConfig saves these values in the Windows registry, but does not download the new IP address to the *UNET*. Downloading the IP address to the *UNET* will be done later. (See **2.5 Setting the UNET's Ethernet Settings** on page 2-11.)

7. Only when instructed by Excalibur Support, in the Enable Module Debug Traces section, select one more of the modules: **429 Module 0, Discrete Module**. This creates a log file that traces the communication between the host and the channels (modules) on the *UNET*. Note that when this debug functionality is not required, it is recommended to clear these checkboxes.
8. Click **OK - Finished Modifying**.

   The configuration details of the device are displayed on the ExcConfig main screen.
9. Continue with Step 15 on page 2-11.

10. (This step is a continuation from Step 4 when using an *DAS-429UNET/RTx* and communicating with the host via USB.) In the Select Connection Type section, click the **USB Connection** option button.

    The Serial Number field is displayed.



Note: The following step is only required when there will be two or more *UNET*'s of the same type (1553 or 429) connecting to the computer via USB. Otherwise, you can skip the next step and leave the asterisk (*). This will allow this device number to communicate with any *UNET* connected to the computer via USB.

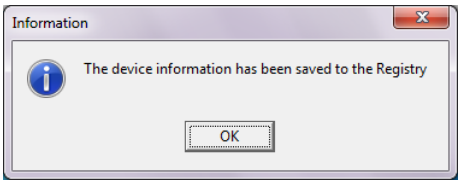
11. Type the serial number located at the back of the *UNET* in the Serial Number field. This will assign the selected device number to this *UNET* only.

Note: You can click the **Allow connection to any UNET connected to this host** button to remove the serial number in the Serial Number field and replace it with an asterisk.

12. Only when instructed by Excalibur Support, in the Enable Module Debug Traces section, select one more of the modules: **429 Module 0, Discrete Module**. This creates a log file that traces the communication between the host and channels (modules) on the *UNET*. Note that when this debug functionality is not required, it is recommended to clear these checkboxes.
13. Click **OK - Finished Modifying**.

    The configuration details of the device are displayed on the ExcConfig main screen.
14. Continue with Step 15 on page 2-11.

15. To save the configuration, click **Save to Registry** on the ExcConfig main screen. The following confirmation dialog box is displayed.



16. Click **OK**.
17. Click the **X** at the top-right of the ExcConfig main screen to exit the ExcConfig utility.

2.5 Setting the *UNET*'s Ethernet Settings

Note: This section is only relevant when using Ethernet for communication between the host and the *UNET*.

The following sections describe how to set the user-defined IP address and MAC address of the *UNET*. When there are two or more *UNET*'s on the same network, it is important to change the *UNET*'s IP address and MAC address, and not to continue using the default IP address and MAC address.

2.5.1 Overview of Setting the *UNET*'s Ethernet Settings

On the *DAS-429UNET/RTx*, there are two ways to set the user-defined IP address and MAC address: via the *UNET*'s USB Communication port or via Ethernet. When the USB option is available, it is recommended to set the user-defined IP Address and MAC address via USB.

The following sections provide detailed instructions for setting the user-defined IP address and MAC address via USB and via Ethernet.

2.5.2 Setting the Ethernet Settings via USB

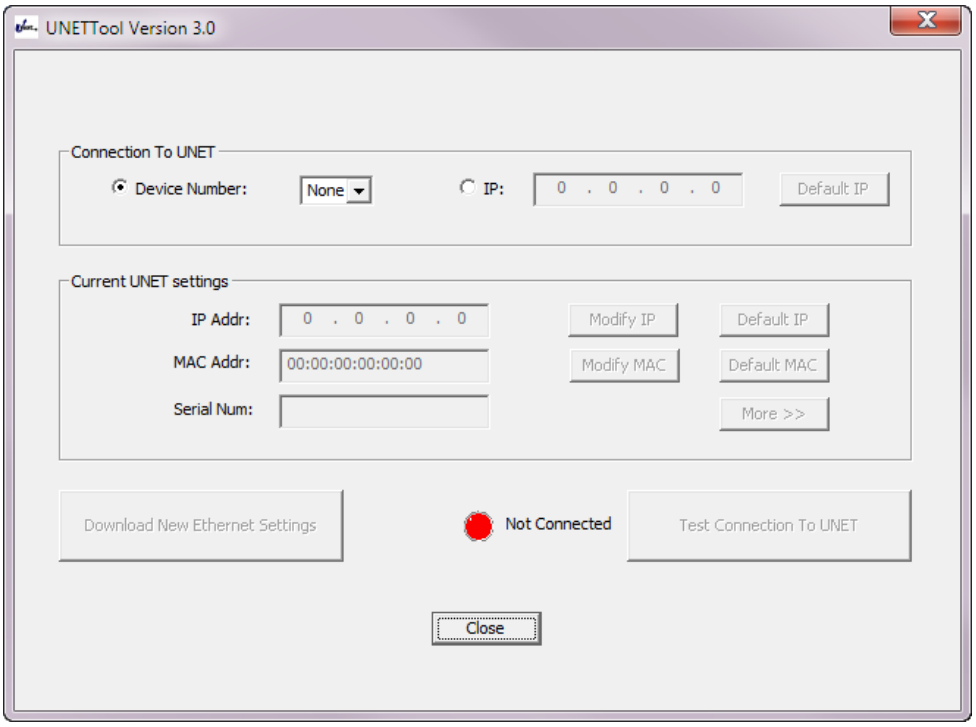
To set the Ethernet Settings via USB:

1. Connect one side of one of the supplied USB cables to one of the computer's available USB ports and the other side to the USB Communication port of the *DAS-429UNET/RTx*, and make sure the LED is lit.
2. Install the USB driver on the host computer (if it was not already installed). See **2.3 Installing the USB Driver on the Host Computer** on page 2-5.
3. Assign a device number selecting USB as the connection type and leave the *UNET*'s default serial number (*). See **2.4 Assigning a Device Number on the Host Computer** on page 2-7.

Note: You can have two device numbers assigned to the same *UNET*, one for communicating via USB and one for Ethernet.

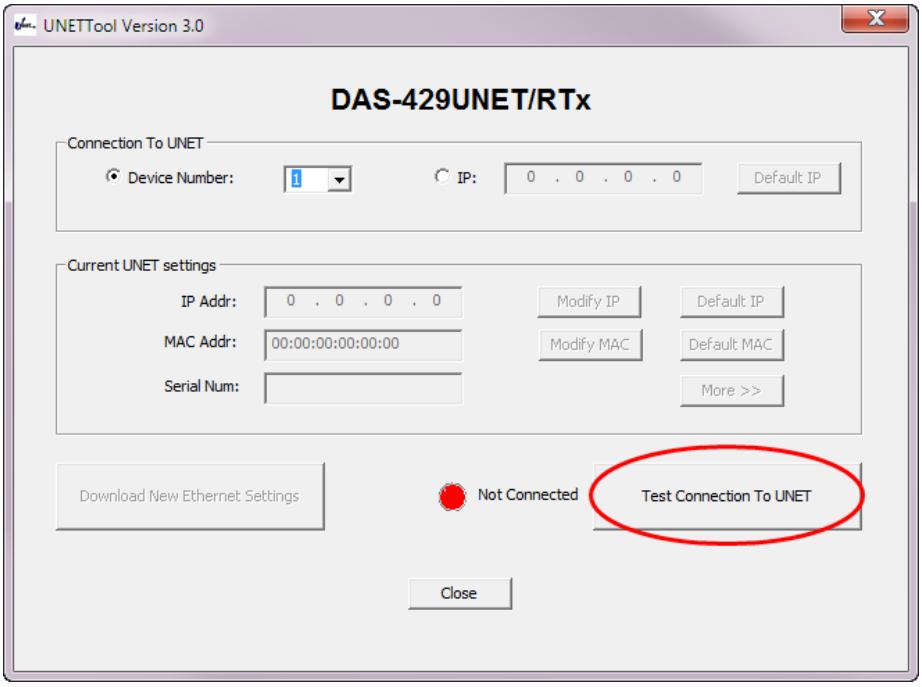
4. Run the UNETTool: Click **Start | All Programs | Excalibur | 429UNET | UNET Utilities | UNET GUI Tool**.

   The UNETTool main screen is displayed.



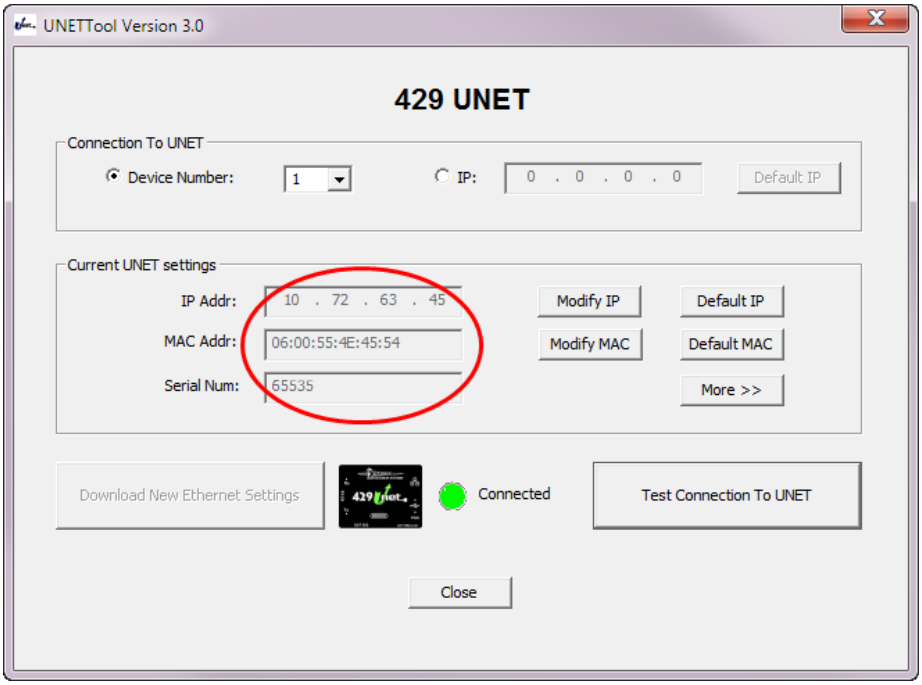
5. In the Connection to UNET section of the screen, select the device number that was assigned to the *DAS-429UNET/RTx* in Step 3, for example, Device #1.

The Test Connection To UNET button becomes active.

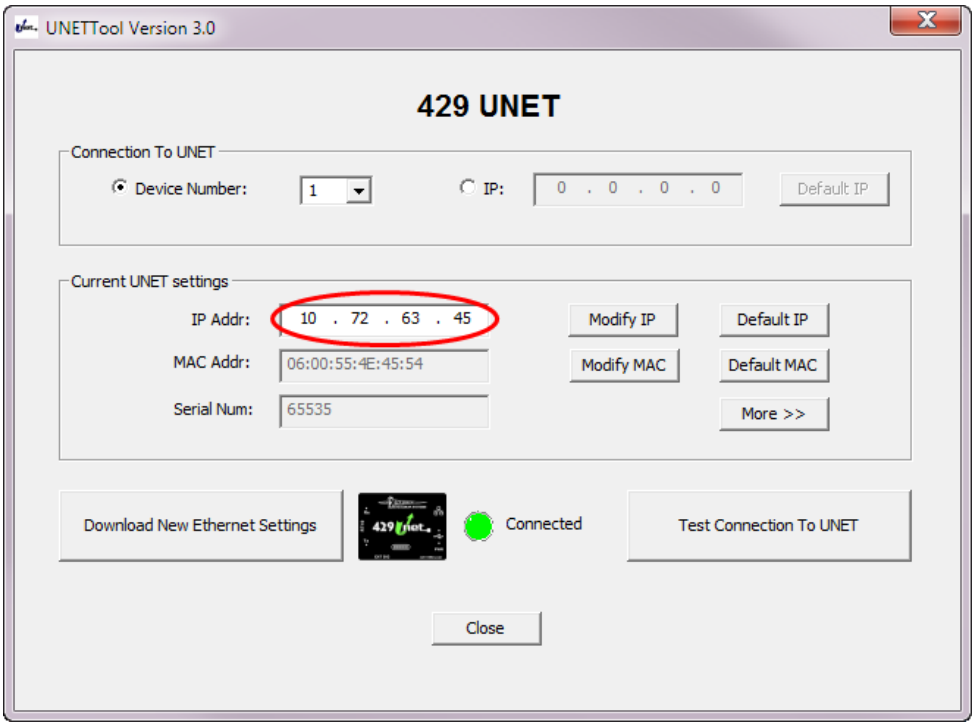


6. Click **Test Connection To UNET**.

   The UNETTool attempts to connect to the *UNET* using the specified device number. When the UNETTool succeeds in connecting to the *UNET*, the LED on the UNETTool turns green, and the *UNET*'s IP address, MAC address and serial number are displayed in the Current UNET settings section of the UNETTool main screen.

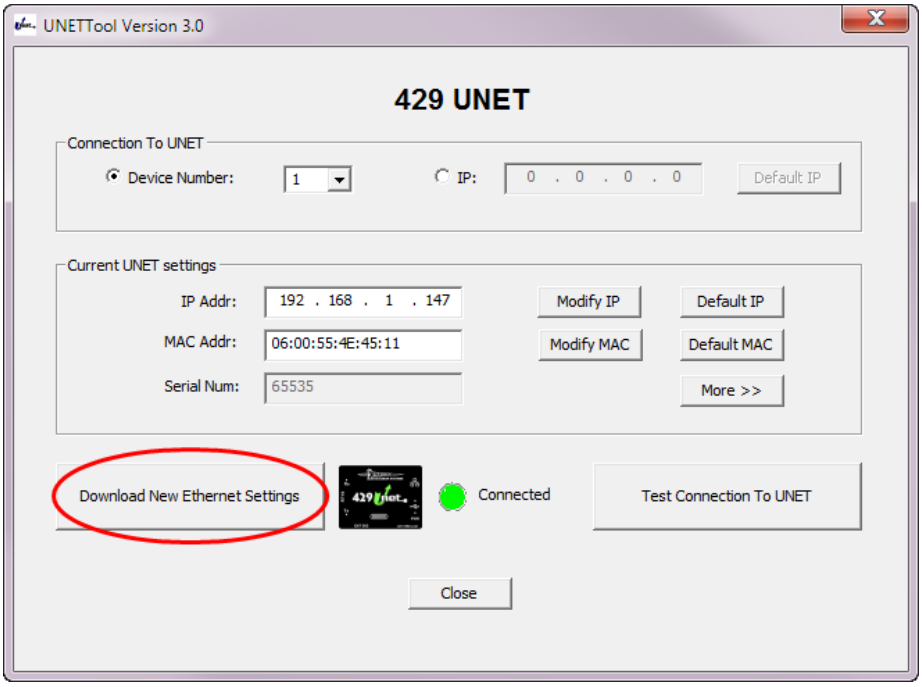


7. Click **Modify IP**.
   The IP Addr field becomes read/write.



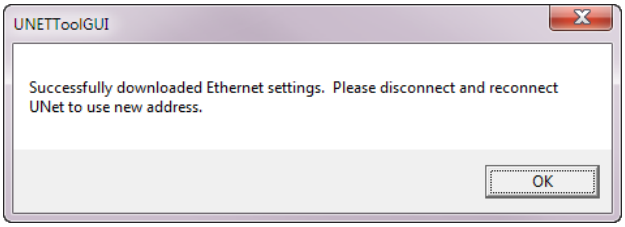
8. In the IP Addr field, type the desired user-defined IP address, for example, 192.168.1.147. If you are not sure of what IP address to use, ask your system administrator for a valid IP address on your network.
9. (Optional) To revert to the default IP address, click **Default IP**.
10. Click **Modify MAC**.
    The MAC Addr field becomes read/write.
11. In the MAC Addr field, type the desired user-defined MAC address, for example, 06:00:55:4E:45:11. If you are not sure of what MAC address to use, ask your system administrator for a valid MAC address.
12. (Optional) To revert to the default MAC address, click **Default MAC**.

The Download New Ethernet Settings button is enabled.



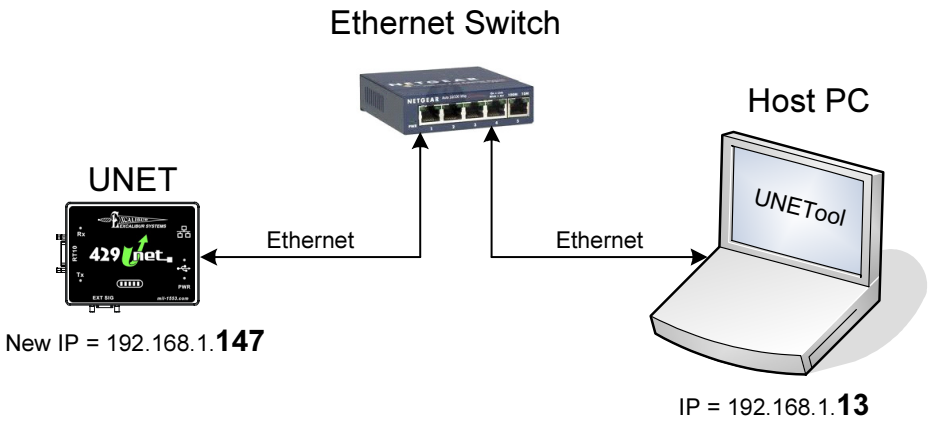
13. Click **Download New Ethernet Settings**.

    The following dialog box is displayed.



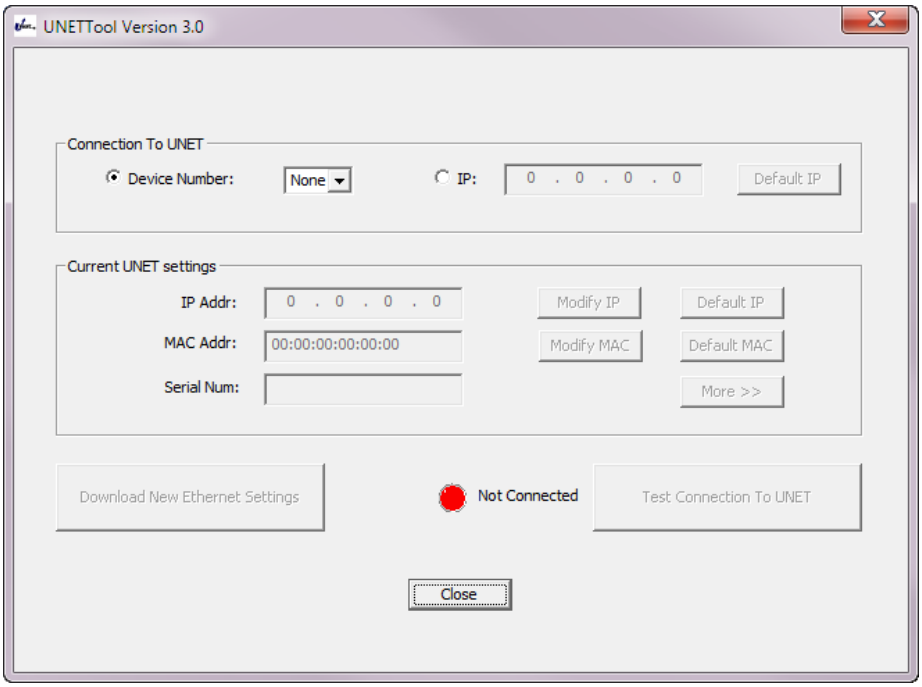
14. Click **OK**.
15. In order for the changes to take effect, restart the *UNET* as follows: Disconnect all USB cables from the *UNET* for several seconds, then reconnect them.
16. Click **Close** to exit the UNETTool.

17. Connect the *UNET* to the host PC via Ethernet. The following figure is an example.



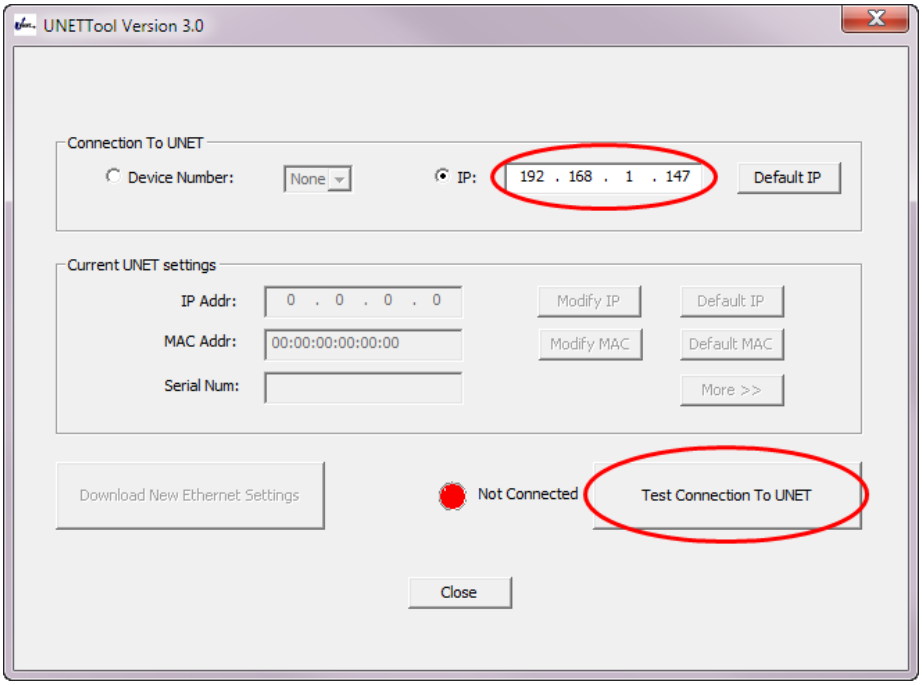
18. Turn off your computer's firewall. When using Windows 7 or later, make sure the firewall of both public and private networks is turned off.
19. Reopen the UNETTool: Click **Start | All Programs | Excalibur | 429UNET | UNET Utilities | UNET GUI Tool**.

    The UNETTool main screen is displayed.



20. In the Connection to UNET section, click the **IP** option button, then type the new IP address of the *UNET*.

    The Test Connection to UNET button is enabled.



21. Click **Test Connection To UNET**.

    The UNETTool attempts to connect to the *UNET* via Ethernet using the new IP address. When the UNETTool succeeds in connecting to the *UNET*, the LED on the UNETTool turns green, and the new IP address and MAC address are displayed in the Current UNET settings area of the UNETTool. This confirms that user-defined IP address and MAC address was modified successfully.

22. Turn the firewall back on.
23. Add the *UNET*'s user-defined IP address as an exception to the firewall.
24. Test the connection to the *UNET* again (steps 20 and 21).
25. Click **Close**.

2.5.3 Setting the Ethernet Settings via Ethernet

When connecting via Ethernet to the *UNET* for the first time, the **default IP** (10.72.63.45) must be used since the *UNET* does not yet have a user-defined IP address set. When connecting to the *UNET* using the default IP, make sure there is only one *UNET* connected to the network, since any *UNET* on the network will respond to commands sent to the default IP address.

In order to connect to the *UNET* with the default IP address you must temporarily change the Local Area Connection settings on the host computer to configure the host computer to be on the same network as the *UNET*, then set the *UNET*'s user-defined IP address and MAC address.

Note: When there are two or more *UNET*'s on the same network, it is important to change the *UNET*'s IP address and MAC address, and not to continue using the default IP address and MAC address.

After changing the *UNET*'s IP address, change the Local Area Connection settings on the host computer back to its original setting.

To change the Local Area Connection settings on the host computer:

1. On the Windows 7 Taskbar, click the **Start**, type **ncpa.cpl**, then press **Enter**. (On Windows XP, click **Start | Run**, type **ncpa.cpl**, then click **OK**.)
2. Right-click **Local Area Connection**, then select **Properties**.

   The Local Area Connection Properties dialog box is displayed.

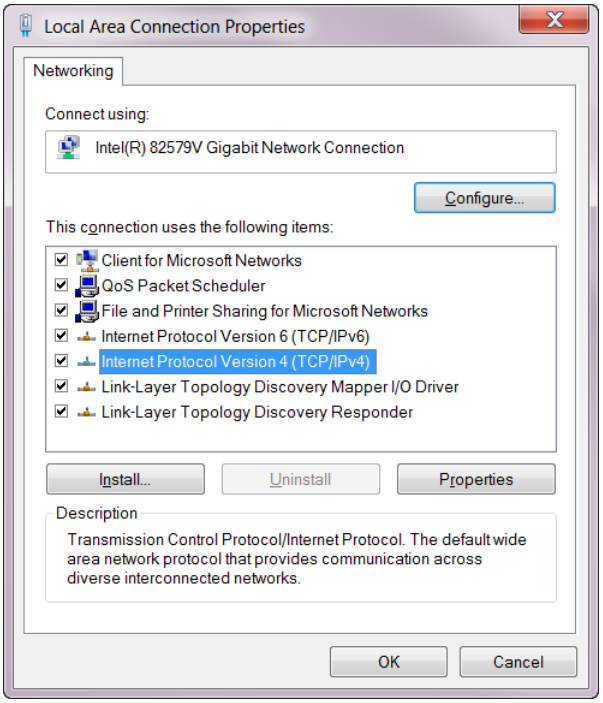


Figure 2-3 Local Area Connection Properties Dialog Box

3. Double-click **Internet Protocol Version 4 (TCP/IPv4)**.

The Internet Protocol Version 4 (TCP/IPv4) Properties dialog box is displayed.

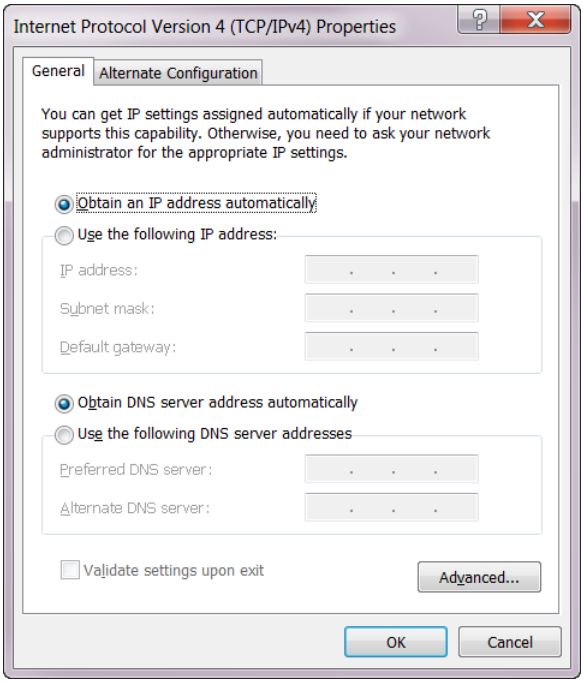


Figure 2-4 Local Area Connection Properties Dialog Box

Note: Make a note of the current settings in the Internet Protocol Version 4 (TCP/IPv4) Properties dialog box, so that you can return to these settings after connecting to the *UNET* and configuring its user-defined IP address.

4. Select **Use the following IP address**.
5. In the IP address field, type **10.72.63.x**, where **x** is any number from 1 to 255 except for 45, which is the IP address of the *UNET*.
6. In the Subnet mask field, type **255.255.255.0**.
7. Click **OK** in the Internet Protocol Version 4 (TCP/IPv4) Properties dialog box.
8. Click **OK** in the Local Area Connection Properties dialog box.

Note: There may be a short delay after changing the Local Area Connection's IP address, before you can connect with the *UNET*.

To set the *UNET*'s Ethernet settings:

1. Use Ethernet cables to connect the *UNET* to the host computer via an Ethernet switch as shown in **Figure 2-5**.

   In **Figure 2-5** the host computer is configured to be on the same network as the *UNET* (10.72.63.*), as described in previous procedure. (See page 2-18.) The last segment of the IP address of the host computer can be any value from 1 to 255 except for 45, which is the IP address of the *UNET*.

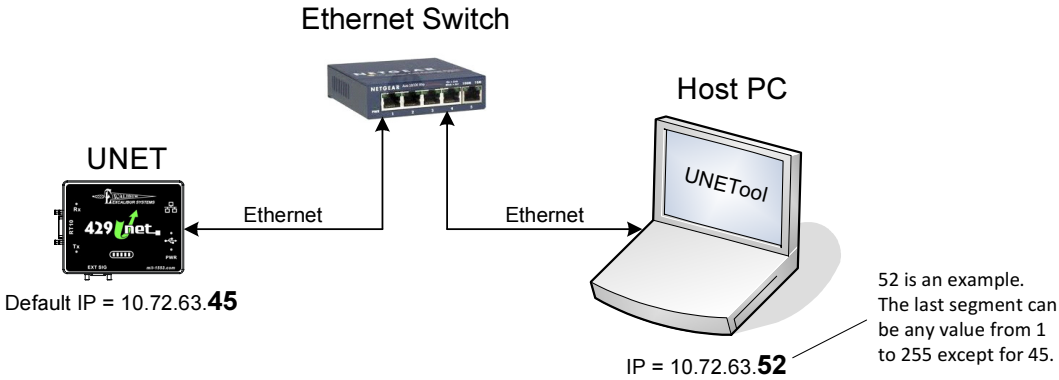
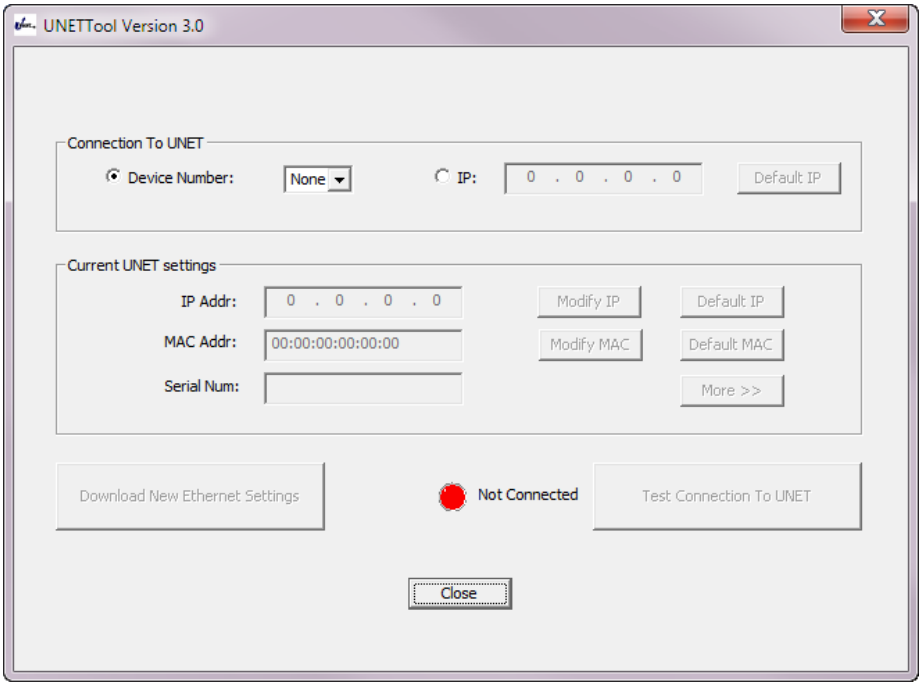


Figure 2-5 Network Setup for Connecting with the *UNET*'s Default IP

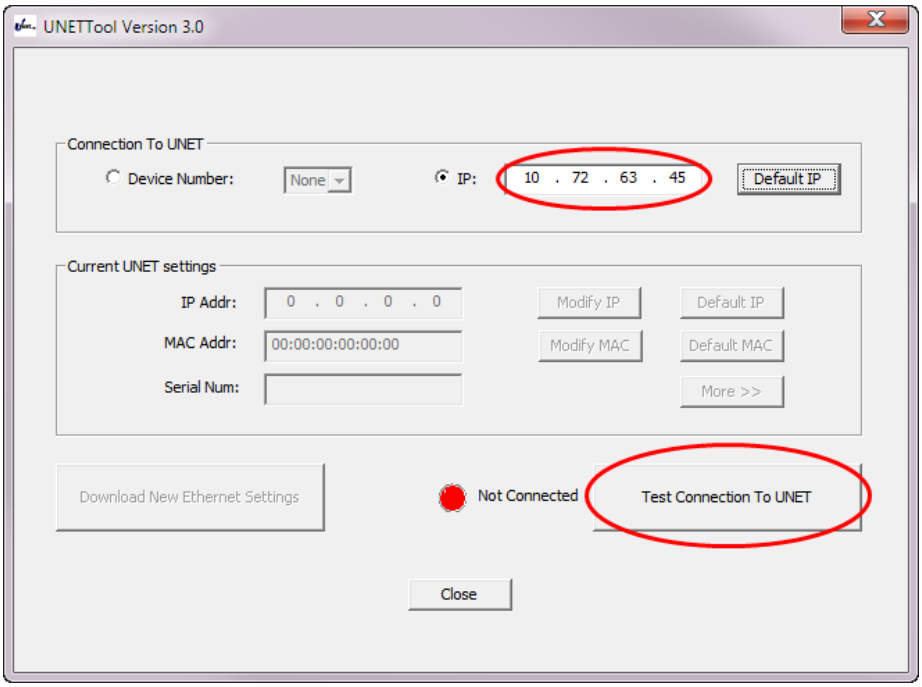
2. For power, do one of the following:
   - Connect the supplied power adapter to a wall outlet and its USB connector to the USB Power port (**PWR**) of the *UNET*.
   - Connect one side of the supplied USB cable to one of the computer's available USB ports and the other side to either the USB Power port (**PWR**) or the USB Communication port ⇽ of the *UNET*.
3. Turn off your computer's firewall. When using Windows 7 or later, make sure the firewall of both public and private networks is turned off.
4. Run the UNETTool: Click **Start | All Programs | Excalibur | 429UNET | UNET Utilities | UNET GUI Tool**.

The UNETTool main screen is displayed.



5. In the Connection to UNET section of the screen, click the **IP** option button.
6. Click **Default IP**.

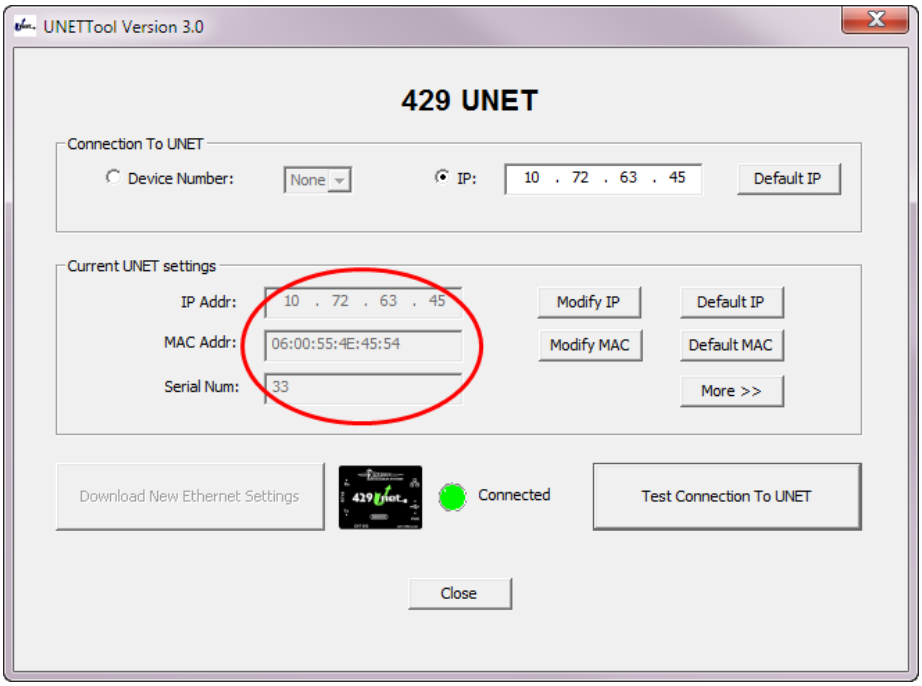
   The default IP of the *UNET* is displayed and the Test Connection to UNET button is enabled.



7. Click **Test Connection To UNET**.

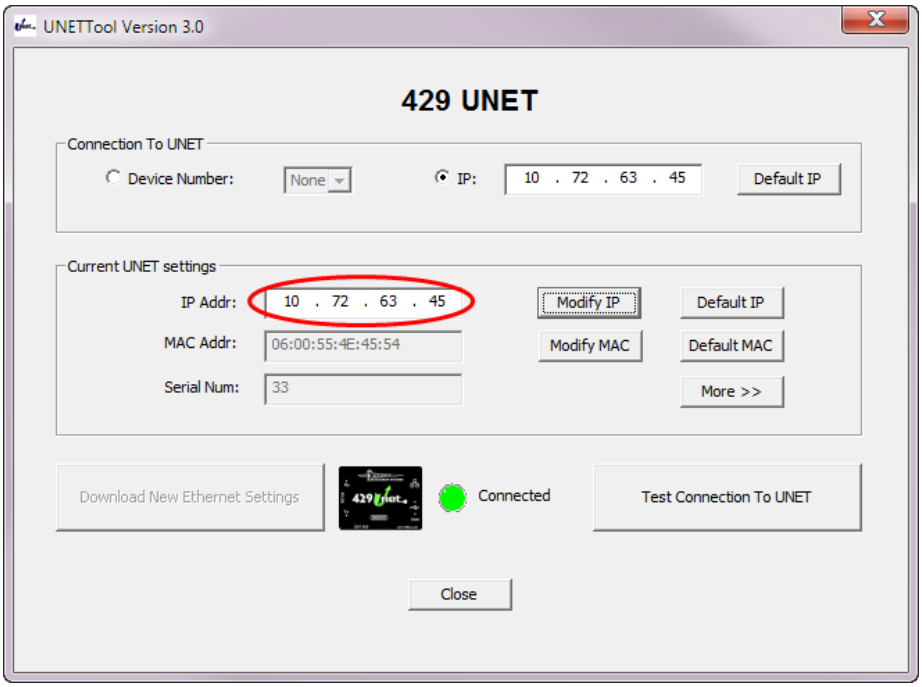
   The UNETTool attempts to connect to the *UNET* via Ethernet using the default IP address. When the UNETTool succeeds in connecting to the

UNET, the LED on the UNETTool turns green, and the *UNET*'s IP address, MAC address and serial number are displayed in the Current UNET settings section of the UNETTool main screen.



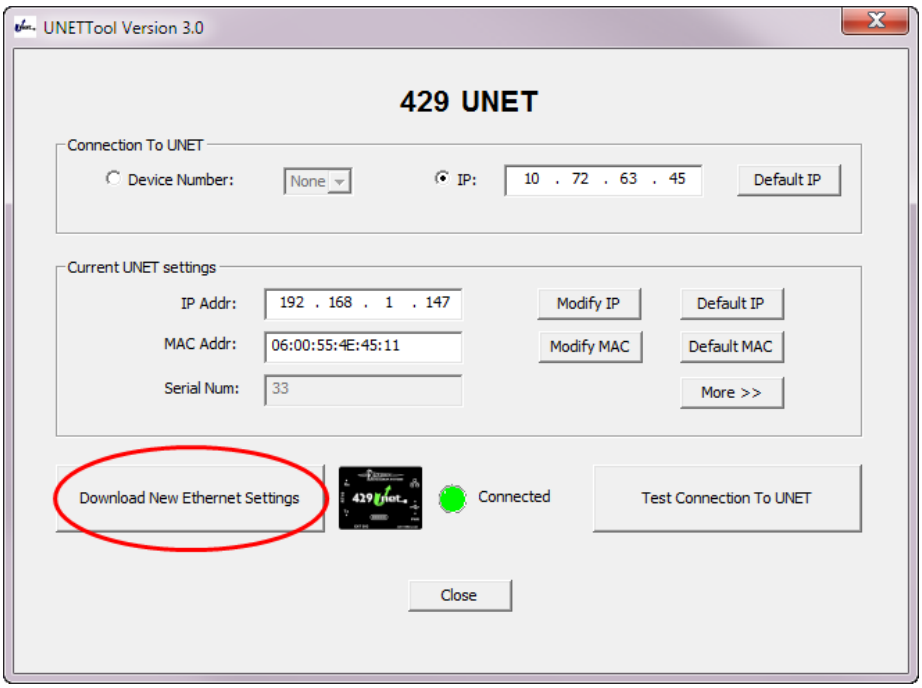
8. Click **Modify IP**.

   The IP Addr field becomes read/write.

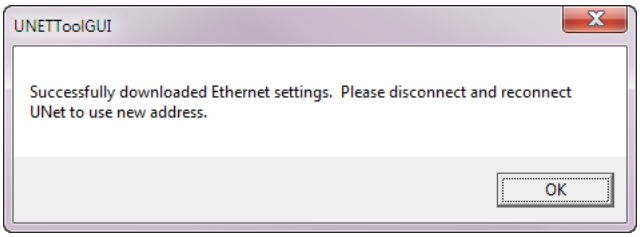


9. In the IP Addr field, type the desired user-defined IP address, for example, 192.168.1.147.
10. (Optional) To revert to the default IP address, click **Default IP**.

11. Click **Modify MAC**.
    The MAC Addr field becomes read/write.
12. In the MAC Addr field, type the desired user-defined MAC address.
    The Download New Ethernet Settings button is enabled.



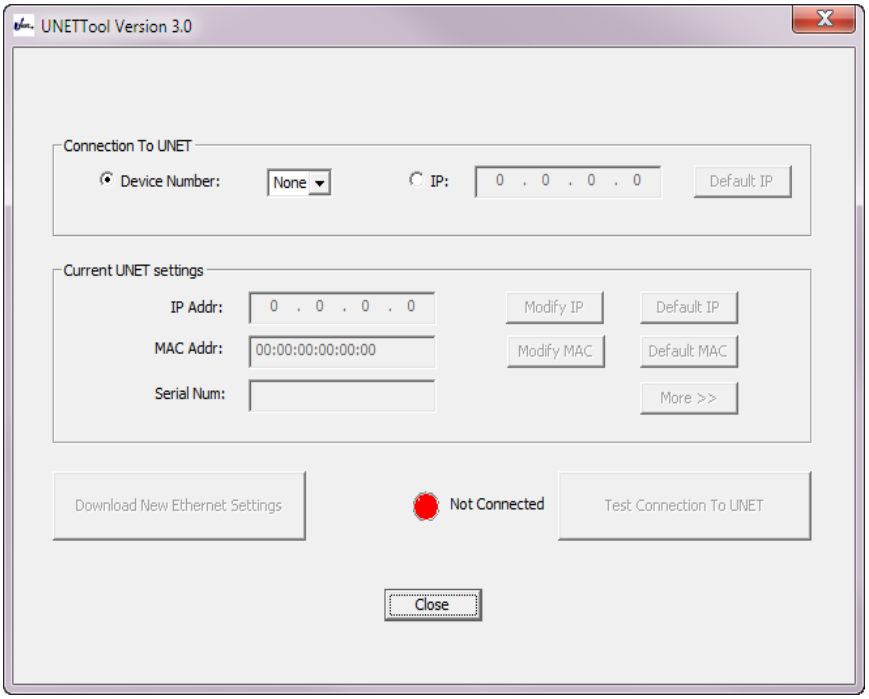
13. Click **Download New Ethernet Settings**.
    The following dialog box is displayed.



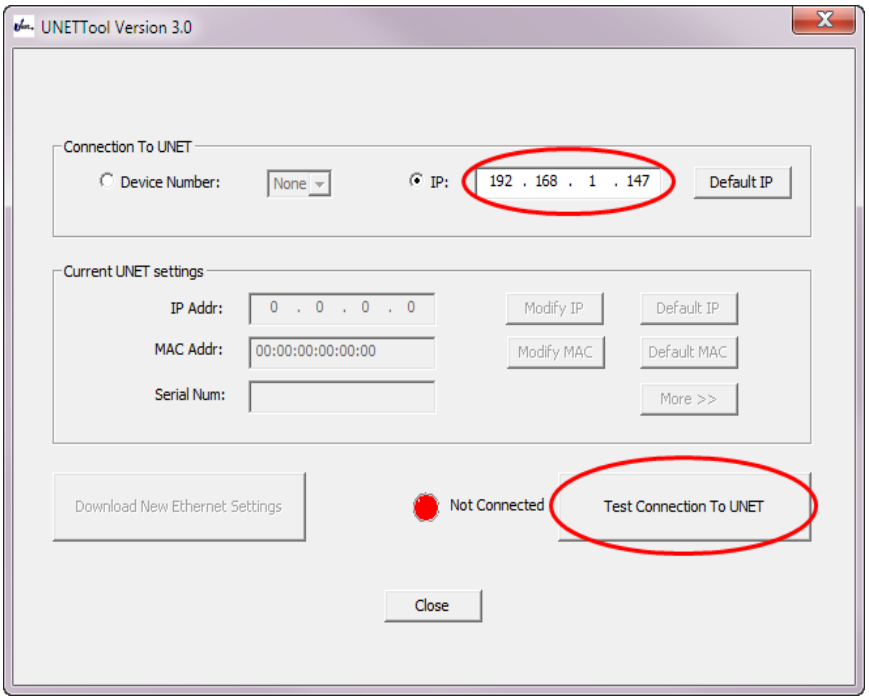
14. Click **OK**.
15. In order for the changes to take effect, restart the *UNET* as follows: Disconnect all USB cables from the *UNET* for several seconds, then reconnect them.
16. Click **Close** to exit the UNETTool.
17. Configure the host computer to have the IP address of the local network by undoing the steps performed in steps 4 and 6 of the previous procedure. See page 2-19.

18. Reopen the UNETTool: Click **Start | All Programs | Excalibur | 429UNET | UNET Utilities | UNET GUI Tool**.

    The UNETTool main screen is displayed.



19. In the Connection to UNET section, click the **IP** option button, then type the new IP address of the *UNET*.



20. Click **Test Connection To UNET**.

    The UNETTool attempts to connect to the *UNET* using the new IP address. When the UNETTool succeeds in connecting to the *UNET*, the LED on the

UNETTool turns green, and the new IP address and MAC address are displayed in the Current UNET settings area of the UNETTool. This confirms that user-defined IP address was modified successfully.

21. Turn the firewall back on.
22. Add the *UNET*'s user-defined IP address as an exception to the firewall.
23. Test the connection to the *UNET* again (steps 19 and 20).
24. Click **Close**.

2.6 Testing the Connection Using the Device Number

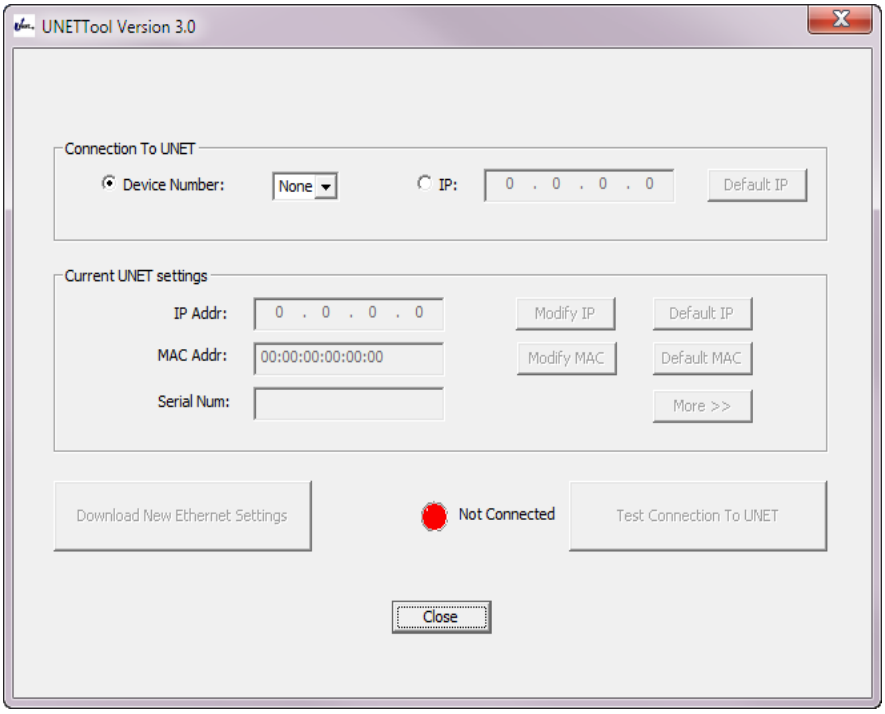
This section describes how to test the connection between the host and the *UNET* using the assigned device number that was assigned using the ExcConfig utility. (See **2.4 Assigning a Device Number on the Host Computer** on page 2-7.)

When testing the connection using the device number, the communication method used to connect to the *UNET* matches the communication method specified when defining the device number. If Device #1 was defined as using USB communication, and Device #2 was defined as using Ethernet communication, then when you select Device #1 the UNETTool will perform the test using USB communication, and when you select Device #2 the UNETTool will perform the test using Ethernet communication.

To test the connection to the *UNET* using the device number:

1. Make sure that the cables are connected as described in **2.2 Connecting the Cables** on page 2-2 and that a device number was assigned as described in **2.4 Assigning a Device Number on the Host Computer** on page 2-7.
2. Run the UNETTool: Click **Start | All Programs | Excalibur | 429UNET | UNET Utilities | UNET GUI Tool**.

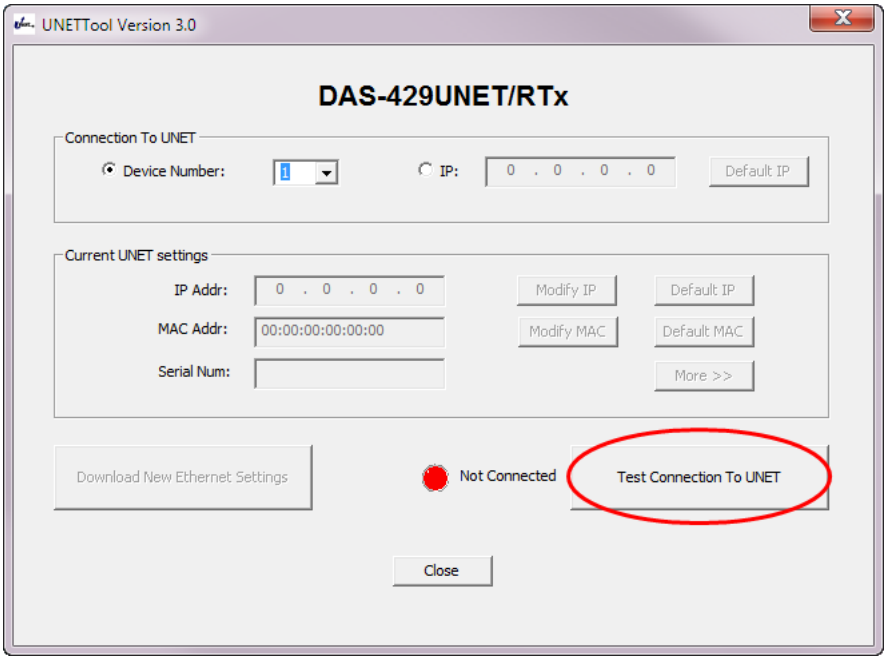
   The UNETTool main screen is displayed.



3. In Connection to UNET section, click the **Device Number** option button, then select the device number of the *UNET*, for example, 1.

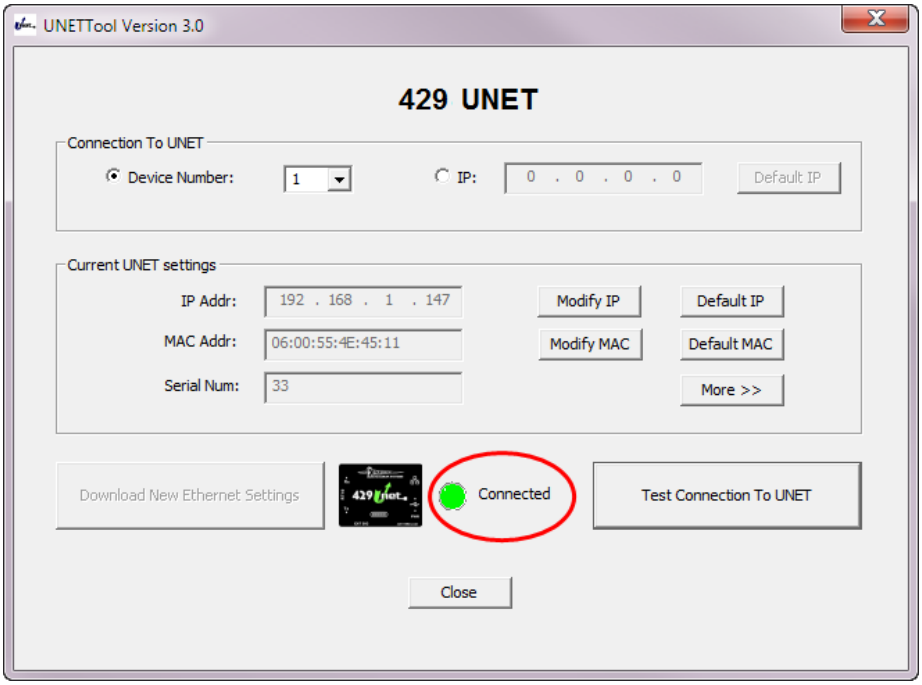
   The Test Connection To UNET button becomes available.

Note: The device number must match the device number that was assigned using the ExcConfig utility. See **2.4 Assigning a Device Number on the Host Computer** on page 2-7.



4. Click **Test Connection To UNET**.

   The UNETTool attempts to connect to the *UNET* using the selected device number. When the UNETTool succeeds in connecting to the *UNET*, the LED on the UNETTool turns green and the status changes to **Connected**.



5. Click **Close**.

2.7 Running a Test Program

Excalibur Software Tools include demo programs to verify that the *UNET* is operating properly.

Excalibur Software Tools are advanced API functions written in C language that enable you to write application and diagnostic programs. The functions are divided into two parts: module-level functions and board-level functions. The demo programs are also divided into module level and carrier board level demo programs.

The source code for the demo programs is also provided as a guide to develop your own applications using the *Software Tools*. The source code is located at:
C:\Excalibur\429UNET\demos_429
C:\Excalibur\429UNET\demos_4000
C:\Excalibur\429UNET\demos_discrete
(The above paths assume the default software installation location.)

Some of the demo programs are available on the Start menu at:
Start | All Programs | Excalibur | 429UNET | Demos (Visual Studio 2008)

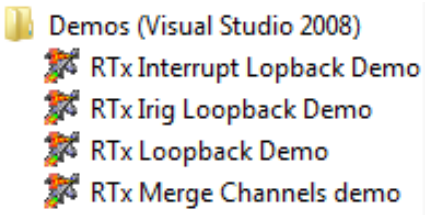


Figure 2-6 Demo Programs on the Start Menu

For more information on the functions used in the demo programs, see:

- *429RTx & Discrete Software Tools Programmer's Reference*
- *4000 Family Carrier Board Software Tools Programmer's Reference*

These manuals can be found at: **Start | All Programs | Excalibur | 429UNET | Manuals**.

Note: These demo programs are console applications. To test the Discrete I/Os using a GUI application, use the Discrete Generator. See **2.8 Running the Discrete Generator** on page 2-29.

To run a test program:

1. Make sure that the cables are connected as described in **2.2 Connecting the Cables** on page 2-2.
2. Connect a loopback cable to one transmit channel and one receive channel.
3. Double-click (for example) **demo_rtx_int.exe** located in: **C:\Excalibur\429UNET\demos_429\bin**. (This path assumes the default software installation location.)

   – or –

   Click **Start | All Programs | Excalibur | 429UNET | Demos (Visual Studio 2008) | RTx Interrupt Loopback Demo**.
4. When prompted for a device number, use the device number defined using ExcConfig. See **2.4 Assigning a Device Number on the Host Computer** on page 2-7.

This device number should also be used when developing your own applications.

5. When prompted for a module number, use 0.
6. When prompted for a transmit channel, use the number of the transmit channel connected to the loopback cable.
7. When prompted for a receive channel, use the number of the receive channel connected to the loopback cable.

   When the system is set up properly, **demo_rtx_int.exe** shows the number of interrupts received by the host computer.

Note: When prompted for a module number when running a demo program in **demos_4000**, use 4. When prompted for a module number when running a demo program in **demos_discrete**, use 2.

2.8 Running the Discrete Generator

The *Discrete Generator* is a Windows GUI program that enables you to configure the Discrete channels on your Discrete module, run the module and view the status of incoming Discretes. You can also configure and monitor interrupts based on status of each Discrete channel.

To install the *Discrete Generator*:

1. Click **Start | All Programs | Excalibur | 429UNET | Install DiscreteGenerator**.

   The InstallShield Wizard is displayed.
2. Follow the on-screen instructions.

To run the *Discrete Generator*:

- Click **Start | All Programs | Excalibur | Discrete Generator | Discrete Generator Tool**.

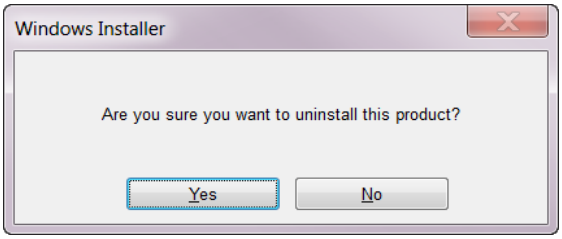
  The *Discrete Generator* main screen is displayed.

  For information on using the *Discrete Generator*, see the *Discrete Generator User's Manual* located at:
  Start | All Programs | Excalibur | Discrete Generator | Discrete Generator Manual.

2.9 Uninstalling the Software

This section describes how to uninstall the *UNET* software. This is required when upgrading to a newer software revision.

1. Click **Start | All Programs | Excalibur | 429UNET | Uninstall DAS-429UNET - CD Rev 1**.
   The following confirmation dialog box is displayed.



2. Click **Yes**.
   The *UNET* software is uninstalled.

3 Developing Applications

Chapter 3 describes how to port an application developed for other Excalibur ARINC 429 and Discrete modules/channels to use with the *UNET*, and brief instructions for developing new applications for the *UNET* using *Excalibur Software Tools*.

Excalibur Software Tools are advanced API functions written in C language that enable you to write application and diagnostic programs.

These functions are divided into two parts: module-level functions for the ARINC 429 and Discrete modules; and board-level functions. The *Software Tools* are fully described in the following manuals:

- *429RTx & Discrete Software Tools Programmer's Reference*
- *M4KDiscrete Software Tools Programmer's Reference*
- *4000 Family Carrier Board Software Tools Programmer's Reference*

These manuals can be found at: **Start | All Programs | Excalibur | 429UNET | Manuals**.

3.1 Porting an Existing Application to Work with the *UNET*

This section describes how to port an existing application to the *UNET* that was developed for an ARINC 429 and Discrete module/channels on a different Excalibur board.

To port your application to the *UNET*:

1. In your original application environment, locate the currently used Excalibur DLLs of the *M4K429RTx* module/channel and the 4000 carrier board. If you are porting applications for Discrete, locate the DLL currently used for the Discrete module.
2. Rename the DLLS by adding a suffix such as **_ORIGINAL.dll**.
3. Run your application.

   You should receive an error that the Excalibur DLLs are missing.
4. If you receive an error that the Excalibur DLLs are missing, continue with Step 7.

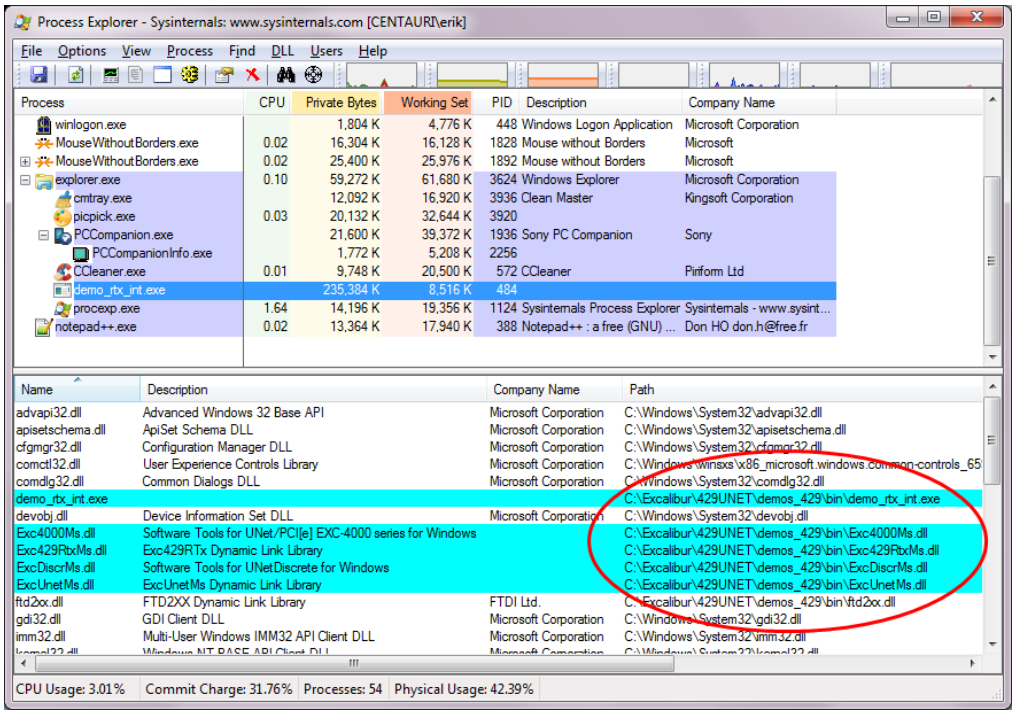
   If you do not receive an error that the Excalibur DLLs are missing, and your application is running, than you have renamed DLLs (in Step 2) that are not being used by your application. In this case, close your application, look for the DLLs in **C:\Windows\System32**, rename them as in Step 2, and run your application.
5. If you receive an error that the Excalibur DLLs are missing, continue with Step 7.

   If you do not receive an error that the Excalibur DLLs are missing, and your application is running, use the Process Explorer utility to locate the DLLs used by your application, as follows:

   a. Double-click **procexp.exe** located in **C:\Excalibur\429UNET\UNET utilities**.

      The Process Explorer License Agreement screen appears.

b. Click **Agree**.
   The Process Explorer main screen is displayed.
c. Make sure your application is still running.
d. Select your application in the list of processes.
e. Click the View DLLs icon on the toolbar.
   The DLLs used by your application are displayed in the lower half of the window.



Note: If the View DLLs icon does not appear on the toolbar, and the DLLs are not displayed in the lower half of the screen, click the View Handles icon, then click the View DLLs icon.

6. Close your application, rename the DLLs as in Step 2, and run your application.
   You will receive an error that the Excalibur DLLs are missing.
7. Copy the relevant DLLs (for your modules and/or carrier board) from the appropriate subfolder in **C:\Excalibur\429UNET\lib**, to the folder where your original DLL was located.

Note: For older applications, make sure the DLL names match the ones originally used by your application. If not, rename them to match the DLL names that were used in your application.

8. In addition to the DLLs for the modules and/or carrier board, some additional DLLs are required:
   - For Microsoft Visual Studio, copy **ExcUnetMs.dll** and **ftd2xx.dll** to the folder where your application is located.
   - For Borland, copy all the additional DLLs to the folder where your application is located.
9. Run your application.

   There should not be a missing DLL error, but there should be an error regarding initializing the card, such as **Init Failed**.
10. Run ExcConfig and set the device number of the *UNET* to match the device number used in your application. See **2.4 Assigning a Device Number on the Host Computer** on page 2-7.
11. Make sure you perform all other relevant installation steps in **Chapter 2: Installation and Setup** before running your application.
12. Run your application.

    It should run without any errors with the *UNET*.
13. If the application still does not run properly, copy the LIB files from the appropriate subfolder in **C:\Excalibur\429UNET\lib** and recompile your application.

3.2 Developing New Applications for the *UNET*

To develop new applications for the *UNET*:

1. Add the **include** folder from **C:\Excalibur\429UNET\include** to your project settings in your development environment.
2. Add the relevant **lib** folder for Microsoft Visual Studio (Debug or Release) or Borland, from **C:\Excalibur\429UNET\lib** to your project settings in your development environment.
3. Add the relevant LIB file names (for your modules and/or carrier board) to your project.
4. Create your application.
5. Make sure you perform all relevant installation steps in **Chapter 2: Installation and Setup** before running your application.
6. Run your application.

Note: If you are using Microsoft Visual Studio, you can use the source code of a demo as programming examples. The demos are located in:
C:\Excalibur\1553UNET\demos_429
C:\Excalibur\1553UNET\demos_discrete
C:\Excalibur\1553UNET\demos_4000

4 Mechanical and Electrical Specifications

Chapter 4 describes the mechanical and electrical specifications of the *UNET*.

4.1 Mechanical Outline

The following are sample configurations. For all possible configurations, see **Chapter 8: Ordering Information.**

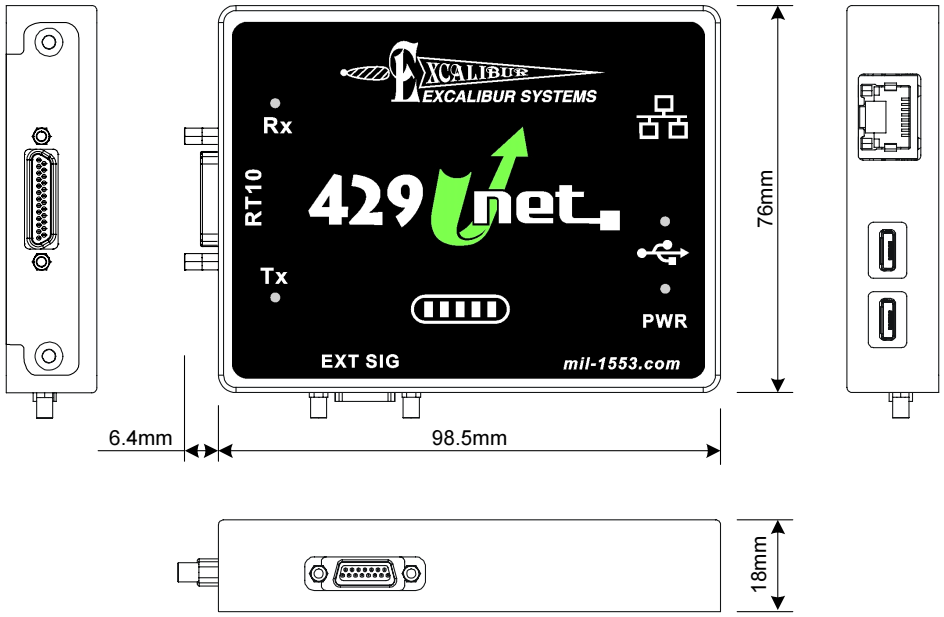


Figure 4-1 ***DAS-429UNET/RTx-M***

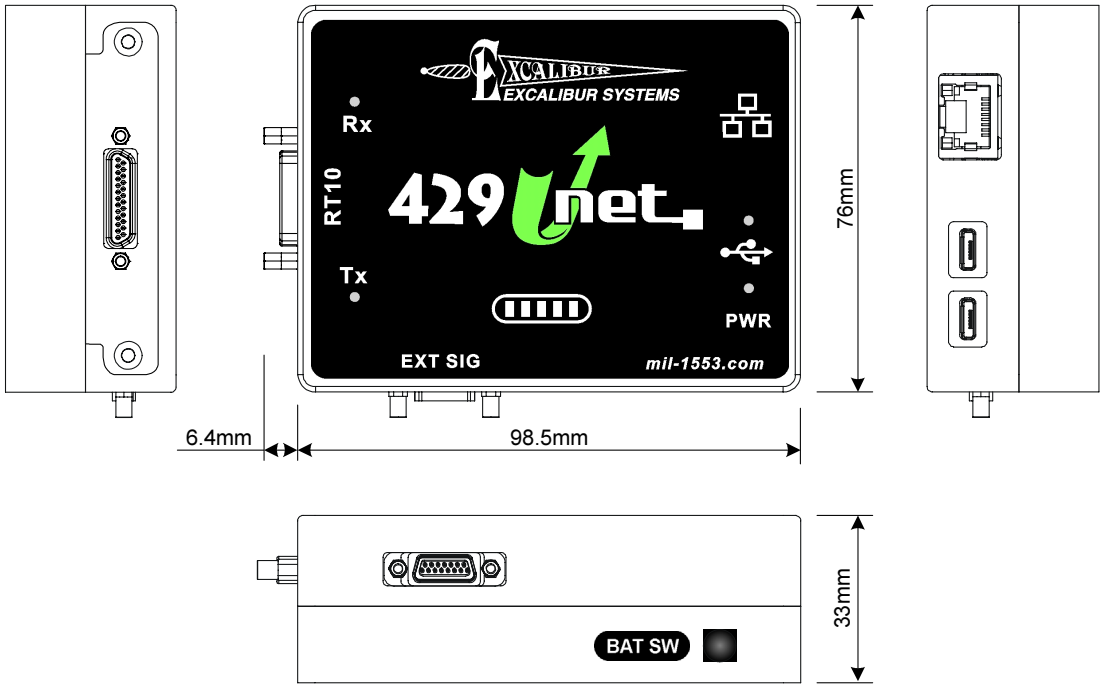


Figure 4-2 ***DAS-429UNET/RTx-M-B*** **with Internal Battery**

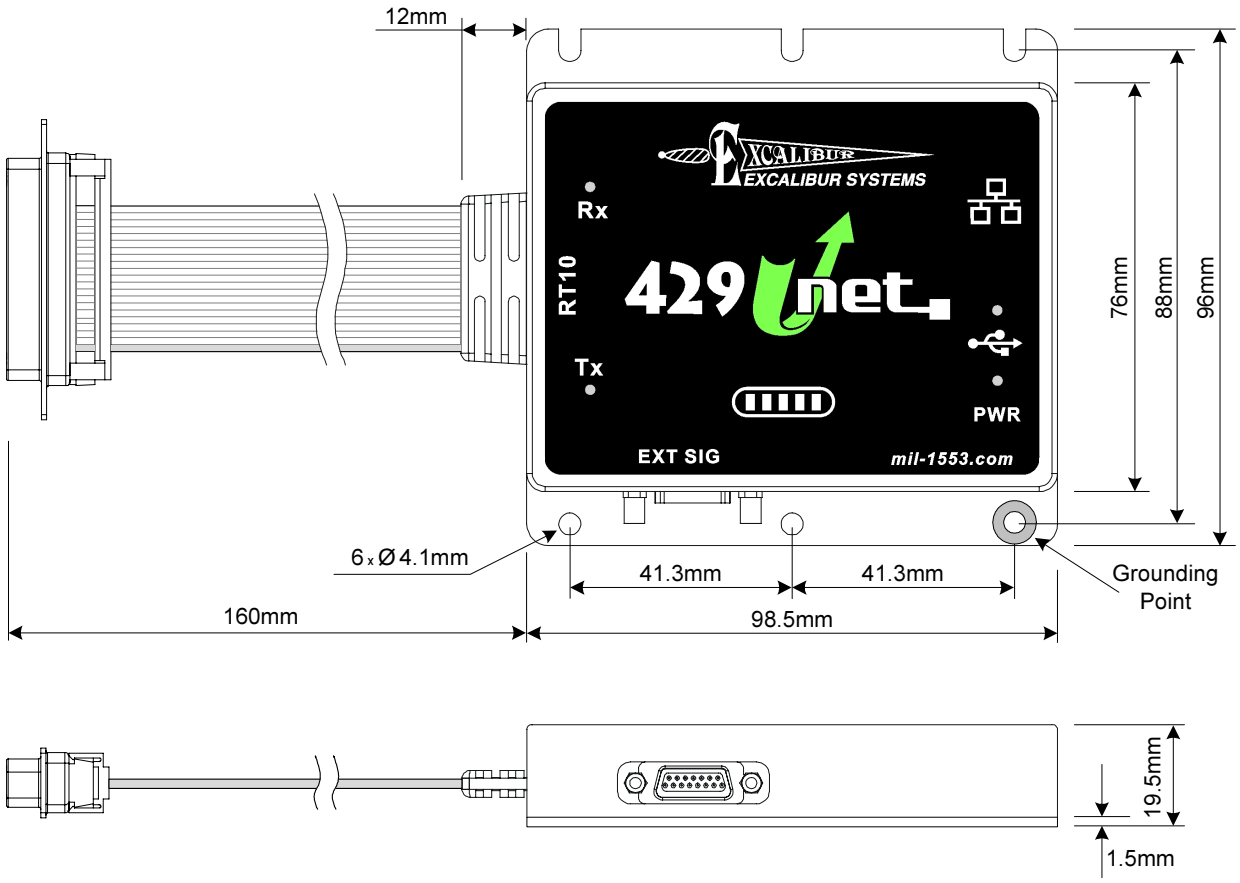


Figure 4-3 *DAS-429UNET/RTx*-C-P with Hard-wired Cable and Mounting Plate

4.2 LED Indicators

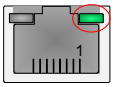
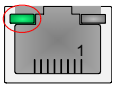
LED	Color	State	Indication
Rx	Blue	Not lit	Power has not been supplied - or - The available ARINC 429 channels failed to initialize - or - *UNET* is either in Fast Charge Mode or Sleep Mode
		Solid	The available ARINC 429 channels are powered up and ready
		Flashing	Data is being received by the available ARINC 429 channels
Tx	Blue	Not lit	Power has not been supplied - or - The available ARINC 429 channels failed to initialize - or - *UNET* is either in Fast Charge Mode or Sleep Mode
		Solid	The available ARINC 429 channels are powered up and ready
		Flashing	Data is being transmitted by the available ARINC 429 channels
	Green	Not lit	No USB communication cable is connected - or - The USB communication cable is connected, but the host has not yet establish a connection with the *DAS-429UNET/RTx* (USB enumeration failed)
		Solid	USB communication cable is connected, the *DAS-429UNET/RTx* has established a connection with the host, and the bus is active
PWR	Green	Not lit	No USB power cable is connected
		Solid	USB power cable is connected and receiving power
	Green	Not lit	No Ethernet cable connected
		Solid	Ethernet cable connected, but no communication on bus
		Flashing	Activity on Ethernet bus
	Green	Not lit	No Ethernet cable connected
		Solid	Ethernet cable connected and Ethernet bus is set to 100 Mbps

Table 4-1 LED Indicators

Table 4-2 describes the functionality of the Battery LEDs. These LEDs only exist when the *DAS-429UNET/RTx* has an internal battery (-B configuration).

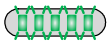
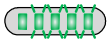
LED	Color	State	Indication
	Battery LED States when Charging in Normal Mode (see page 4-14)		
	Green	None lit	0% charge in the internal battery
	Green	All battery LEDs flashing in sequence	Charging, 0% – 19% charge
	Green	1 LED solid, the rest of the LEDs flashing in sequence	Charging, 20% – 39% charge
	Green	2 LEDs solid, the rest of the LEDs flashing in sequence	Charging, 40% – 59% charge
	Green	3 LEDs solid, the rest of the LEDs flashing in sequence	Charging, 60% – 79% charge
	Green	4 LEDs solid, one flashing	Charging, 80% – 99% charge
	Green	All LEDs solid	100% charge in Normal Mode (or Fast Charge Mode)
	Battery LED States when Discharging in Normal Mode (see page 4-14)		
	Green	All LEDs solid	Discharging, 100% – 91% charge in the internal battery
	Green	4 LEDs solid, 1 LED flashing	Discharging, 90% – 81% charge
	Green	4 LEDs solid	Discharging, 80% – 71% charge
	Green	3 LEDs solid, 1 LED flashing	Discharging, 70% – 61% charge
	Green	3 LEDs solid	Discharging, 60% – 51% charge
	Green	2 LEDs solid, 1 LED flashing	Discharging, 50% – 41% charge
	Green	2 LEDs solid	Discharging, 40% – 31% charge
	Green	1 LED solid, 1 LED flashing	Discharging, 30% – 21% charge
	Green	1 LED solid	Discharging, 20% – 11% charge
	Green	1 LED flashing	Discharging, 10% – 5% charge
	Green	3 LEDs flashing in an alternating pattern	Discharging, 4% – 0% charge
	Green	No LEDs lit	0% charge

Table 4-2 Battery LED Indicators

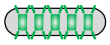
LED	Color	State	Indication
	Battery LED States in Fast Charge and Sleep Modes (see page 4-14)		
	Green	All battery LEDs flashing in sequence	Charging in Fast Charge Mode, 0% – 99% charge
	Green	All LEDs solid	100% charge in Fast Charge Mode (or Normal Mode)
	Green	Middle LED flashing	Sleep Mode

Table 4-2 Battery LED Indicators (Continued)

4.3 Connectors

The following sections describe the connectors of the *DAS-429UNET/RTx*. For information on connecting the cables, see **2.2 Connecting the Cables** on page 2-2.

Caution: **Make sure there is no I/O communication while disconnecting any of the cables. Connecting or disconnecting the cables during communication can seriously damage the DAS-429UNET/RTx.**

4.3.1 I/O Connector

The -C configuration (*DAS-429UNET/RTx-C*) has a hard-wired flat cable with a standard DB 25-pin female connector for all ARINC 429 signals.

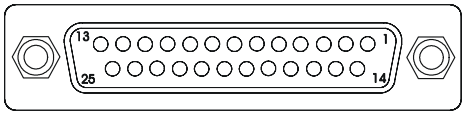


Figure 4-4 Standard DB 25-pin Connector – Front View

The -M configuration (*DAS-429UNET/RTx-M*) has a micro miniature DB 25-pin female connector mounted on the front panel for all ARINC 429 signals.

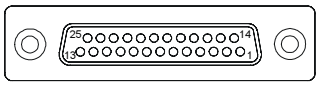


Figure 4-5 Micro Miniature DB 25-pin connector on *DAS-429UNET/RTx-M* – Front View

For the -M configuration, Excalibur supplies a 30 cm (11.8 in.) adapter cable with a standard DB 25-pin female connector. This connector is the same as the connector on the -C configuration. See Figure 4-4.

All these connectors have the same pinouts, which are listed in Table 4-3 on page 4-7.

Pin #	Signal Name	Description
1	CH0L	Channel 0 ARINC low line connection
2	CH1L	Channel 1 ARINC low line connection
3	CH2L	Channel 2 ARINC low line connection
4	CH3L	Channel 3 ARINC low line connection
5	CH4L	Channel 4 ARINC low line connection
6	GND	Ground, provided for cable shield connection
7	CH5L	Channel 5 ARINC low line connection
8	CH6L	Channel 6 ARINC low line connection
9	CH7L	Channel 7 ARINC low line connection
10	CH8L	Channel 8 ARINC low line connection
11	CH9L	Channel 9 ARINC low line connection
12-13	N/C	Not connected
14	CH0H	Channel 0 ARINC high line connection
15	CH1H	Channel 1 ARINC high line connection
16	CH2H	Channel 2 ARINC high line connection
17	CH3H	Channel 3 ARINC high line connection
18	CH4H	Channel 4 ARINC high line connection
19	CH5H	Channel 5 ARINC high line connection
20	CH6H	Channel 6 ARINC high line connection
21	CH7H	Channel 7 ARINC high line connection
22	CH8H	Channel 8 ARINC high line connection
23	CH9H	Channel 9 ARINC high line connection
24-25	N/C	Not connected

Table 4-3 I/O Connector Pinouts

4.3.2 USB Communication Connector

The USB Communication Connector is a standard 5-pin, Micro-B USB connector. A 1-meter Micro-B to Standard-A cables is supplied.

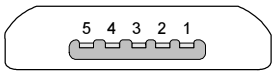


Figure 4-6 USB Communication Connector – Front View

Pin #	Signal Name	Description
1	VBUS	+5V power supply from USB bus
2	DATA-	USB DATA- signal
3	DATA+	USB DATA+ signal
4	N\C	Not connected
5	GND	Ground

Table 4-4 USB Communication Connector Pinouts

4.3.3 USB Power Connector

The USB Power Connector is a standard 5-pin, Micro-B USB connector. A 1-meter Micro-B to Standard-A cables is supplied.

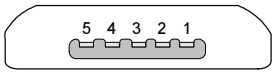


Figure 4-7 USB Power Connector – Front View

Pin #	Signal Name	Description
1	VBUS	+5V power supply from USB bus
2	PwCon-	USB power configuration signal
3	PwCon+	USB power configuration signal
4	N\C	Not connected
5	GND	Ground

Table 4-5 USB Communication Connector Pinouts

4.3.4 Ethernet Connector

The Ethernet Connector is a standard RJ45, 8-pin, Ethernet connector.

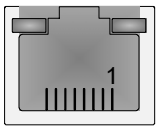


Figure 4-8 Ethernet Connector – Front View

Pin #	Signal Name	Description
1	TX+	Ethernet Transmit Data+
2	TX-	Ethernet Transmit Data-
3	RX+	Ethernet Receive Data+
4	N/C	Not Connected
5	N/C	Not Connected
6	RX-	Ethernet Receive Data-
7	N/C	Not Connected
8	N/C	Not Connected

Table 4-6 Ethernet Connector Pinouts

4.3.5 External Signals Connector Pinouts

The External Signals Connector is a 1.27mm pitch micro D, 15-pin connector, P/N: Molex 83612-9020.

The mating connector, P/N: Molex 83422-9014, and a socket crimp terminal, P/N: Molex 83000-0083, are supplied by Excalibur.

A custom adapter cable for the External Signals Connector can be ordered from Excalibur. Contact Excalibur Sales. See **1.6 Technical Support** on page 1-7.

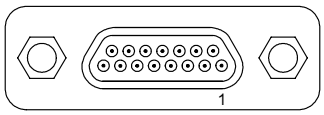


Figure 4-9 External Signals Connector – Front View

Pin #	Signal Name	Description
1	DIO0	Discrete Channel 0 connection[1]
2	DIO2	Discrete Channel 2 connection[1]
3	GND	Ground connection for Discrete channels
4	DIO5	Discrete Channel 5 connection[1]
5	DIO7	Discrete Channel 7 connection[1]
6	EXTTCLKI	External Time Tag Clock Input (nominal value: 1MHz). This signal supplies an external global clock for the Time Tags of all the channels. Use the signal to synchronize the Time Tags that are implemented on the channels[2] to other boards or systems.[3] See **Time Tag Clock Select Register** on page 5-4.
7	EXTTRSTn	External Time Tag Reset TTL Input. Use this low active pulsed signal (minimum 100 nsec.wide) to simultaneously reset the Time Tags of all the channels from an external source. Use the signal to synchronize these Time Tags to other boards or systems.[3]
8	GND	Provides ground reference for the digital signal connections.
9	DIO1	Discrete Channel 1 connection[1]
10	DIO3	Discrete Channel 3 connection[1]
11	DIO4	Discrete Channel 4 connection[1]
12	DIO6	Discrete Channel 6 connection[1]
13	OUTRIGn	Interrupt/Trigger Output. This low active output provides trigger pulses of approximately 400 nsec. width and is activated under software control upon the same conditions as interrupts. See **Receive Merge Interrupt/ Trigger Condition Register** on page 6-9 and **Channel x Interrupt/ Trigger Condition Register** on page 6-52. This output is an open-collector with 330-ohm pull-up resistor.
14	IRIG B	IRIG B120 Input. This should be a 1KHz sine wave, amplitude modulated, IRIG B signal with a 3:1 modulation ratio at 3V typical.
15	EXTTRSOn	Global Time Tag Reset TTL Output. This low active signal is activated each time a Global Time Tag Reset is applied. Use the signal to synchronize other boards or systems to the Time Tags that are implemented on the channels.[2] This signal is activated by either an internal Global Time Tag Reset (see **Software Reset Register** on page 5-3) or by an External Time Tag Reset signal (EXTTRSTn).[3]

Table 4-7 External Signals Connector Pinouts

1. See **1.5 Discrete Channel Information,** on page 1-6.
2. See **Time Tag Counter** on page 6-17 and **Time Tag Hi & Lo** on page 7-14 for a description of how the Time Tag clock is implemented for each channel.
3. See **4.3.5.1 Synchronizing with an External Source** on page 4-11 and **4.3.5.2 Synchronizing Between DAS-429UNET/RTx Devices** on page 4-12.

4.3.5.1 Synchronizing with an External Source

To synchronize a single *DAS-429UNET/RTx* to an external system, the external clock source and the external reset must be connected to the EXTTCLKI and the EXTTRSTn signals respectively.

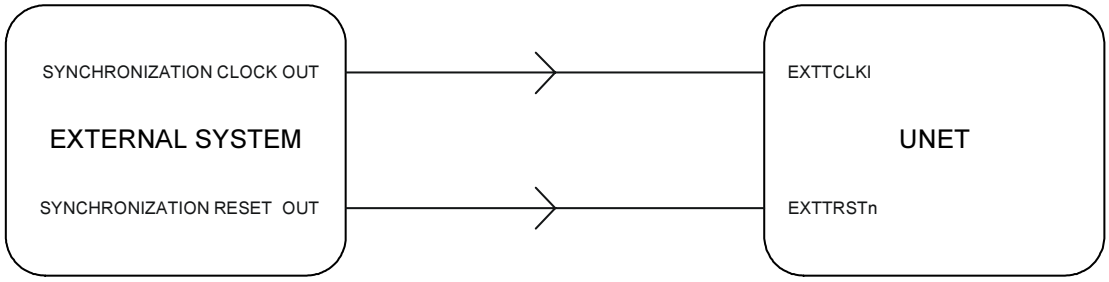


Figure 4-10 Synchronization of a Single *DAS-429UNET/RTx* Board to an External System

To synchronize an external system to a single *DAS-429UNET/RTx*, the EXTTRSOn signal must be connected to the external reset of the external system.

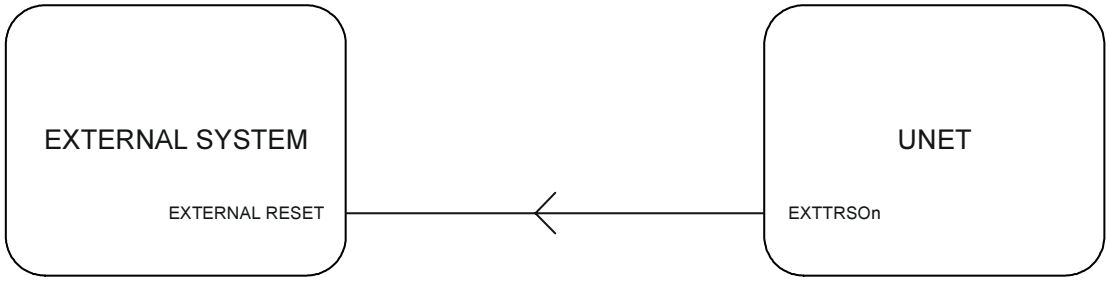


Figure 4-11 Synchronization of an External System to a Single *DAS-429UNET/RTx* Board

Note: The synchronization clock and reset signals may be connected to multiple targets to achieve system wide synchronization.

4.3.5.2 Synchronizing Between *DAS-429UNET/RTx* Devices

To synchronize multiple *DAS-429UNET/RTx* devices the **EXTTRSOn** signal of one board must be connected to all the **EXTTRSTn** signals of the remaining devices.

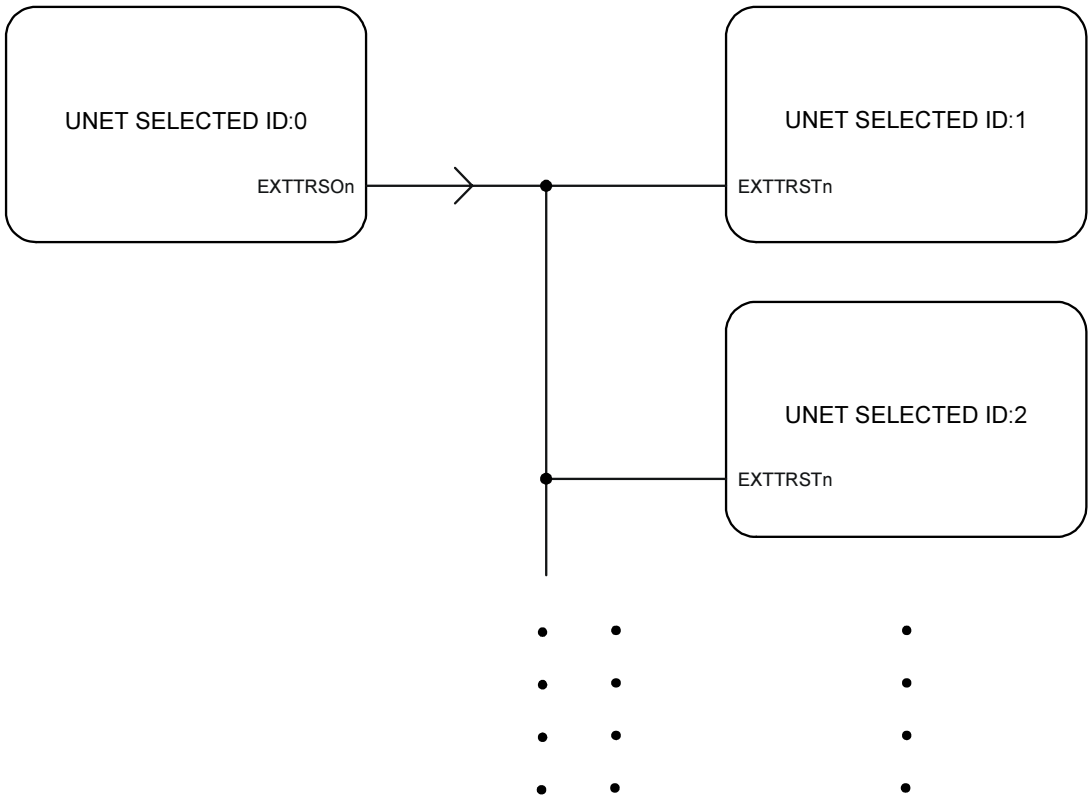


Figure 4-12 Synchronization Between *DAS-429UNET/RTx* Devices

4.4 Power Requirements for *UNET*

The maximum power requirements of the *UNET* @5V are:

RT5	*RT10*	Conditions
+5V @ 500 mA	+5V @ 500 mA	All channels set to receive (default on power-up)
+5V @ 50 mA	+5V @ 50 mA	Add this amount for each channel set to transmit at high speed, maximum load and 100% duty cycle
+5V @ 750 mA	+5V @ 1000 mA	Total amount for all channels set to transmit at high speed, maximum load and 100% duty cycle

Table 4-8 *DAS-429UNET/RTx* Power Requirements

Note:

- Use only the USB Power Supply provided by Excalibur Systems. Excalibur Systems is not responsible for any damage or lack of functionality when a different power supply is used.
- In most cases (depending on the load on the ARINC 429 bus and capabilities of the host computer), the power supported through the USB Communication port will be enough to operate the *UNET*. If the power from the USB Communication port is insufficient, connect the USB Power port (**PWR**) of the *UNET* to another USB port of the host computer, or to a wall outlet using the 5V USB Power Supply provided by Excalibur Systems. Each of these three power sources (USB Communication port to PC USB, USB Power port (**PWR**) to PC USB or USB Power port (**PWR**) to wall outlet) can be used separately as the prime power source.

4.4.1 Battery Information (Battery Option Only)

The *UNET*'s internal rechargeable battery can be recharged via one (or both) of the *UNET*'s USB ports. The time to fully charge the internal battery is dependent on the amount of power required to operate the *UNET* and the operating mode. (See **4.4.1.1 UNET Operating Modes (Battery Option Only)** on page 4-14.) The charging time lengthens with the increase of data being transmitted.

When operating solely from the battery, the *UNET* can:

- Transmit on 5 channels with a 100% duty cycle, full load, for approximately 6 hours.
- Transmit on 10 channels with a 100% duty cycle, full load, for approximately 3 hours.

The amount of power and transmission time that the *UNET* can provide when running solely from the internal battery is influenced by the environmental temperature, the age of the battery and the number of charging cycles. Like all rechargeable batteries, the amount of power it can provide is reduced over time, and in lower temperature environments.

4.4.1.1 *UNET* Operating Modes (Battery Option Only)

The *UNET* has three operating modes that are controlled by a power button on the side of the *UNET*:

- **Normal Mode** – This mode is for normal usage. Use this mode for transmitting and receiving. In this mode, the *UNET* uses as much power as needed to transmit/receive. When connected to an external source, the *UNET* will charge the battery when it is receiving more power than necessary to transmit/receive, and will draw power from the battery when it is not receiving enough power to transmit/receive.
- **Fast Charge Mode** – This mode is for charging the *UNET*'s battery at the fastest possible rate. The *UNET* can only enter this mode when it is connected to external power (wall socket or computer). You cannot transmit or receive in this mode.
- **Sleep Mode** – This is a low power usage mode that can be used to save battery power when the *UNET* is **not** connected to external power (wall socket or computer). You cannot transmit or receive in this mode.

When you turn the *UNET* on, it starts up in Normal Mode. To change to Fast Charge Mode or Sleep Mode, press and hold the power button for 2 seconds. When the *UNET* is connected to any external power, the *UNET* goes into Fast Charge Mode. In this mode, all the LEDs will flash in sequence. (See Table 4-2 on page 4-6.) When the *UNET* is not connected to external power, the *UNET* goes into Sleep Mode. In this mode, the middle battery LED flashes, and all of the other LEDs are off. (See Table 4-2 on page 4-6.)

To return to Normal Mode (from either Fast Charge Mode or Sleep Mode), press the power button for half a second.

4.4.1.2 Tuning OFF the *UNET* (Battery Option Only)

You can turn the *UNET* completely off by pressing the power button for 4 seconds. This can only be done when there is no external power connected. Therefore, the *UNET* can only be turned off in Normal mode when no external power is connected or in Sleep Mode.

Turning off the *UNET* allows it to be stored for long periods without draining the battery. When the *UNET* is turned off, the battery LEDs will turn on and then off in a "train-like" fashion. In the off state, a short press on the power button, or connecting the *UNET* to external power, will power up the *UNET* into Normal Mode.

5 Host Interface Global Registers (Advanced)

Chapter 5 describes the *UNET*'s Host Interface Global registers. This chapter is one of several chapters that describe how to operate the *UNET* via the memory registers, which enables you to develop your own software tools. When using Excalibur's software tools, this chapter is not required.

The following topics are covered:

5.1 Host Interface Global Registers Map

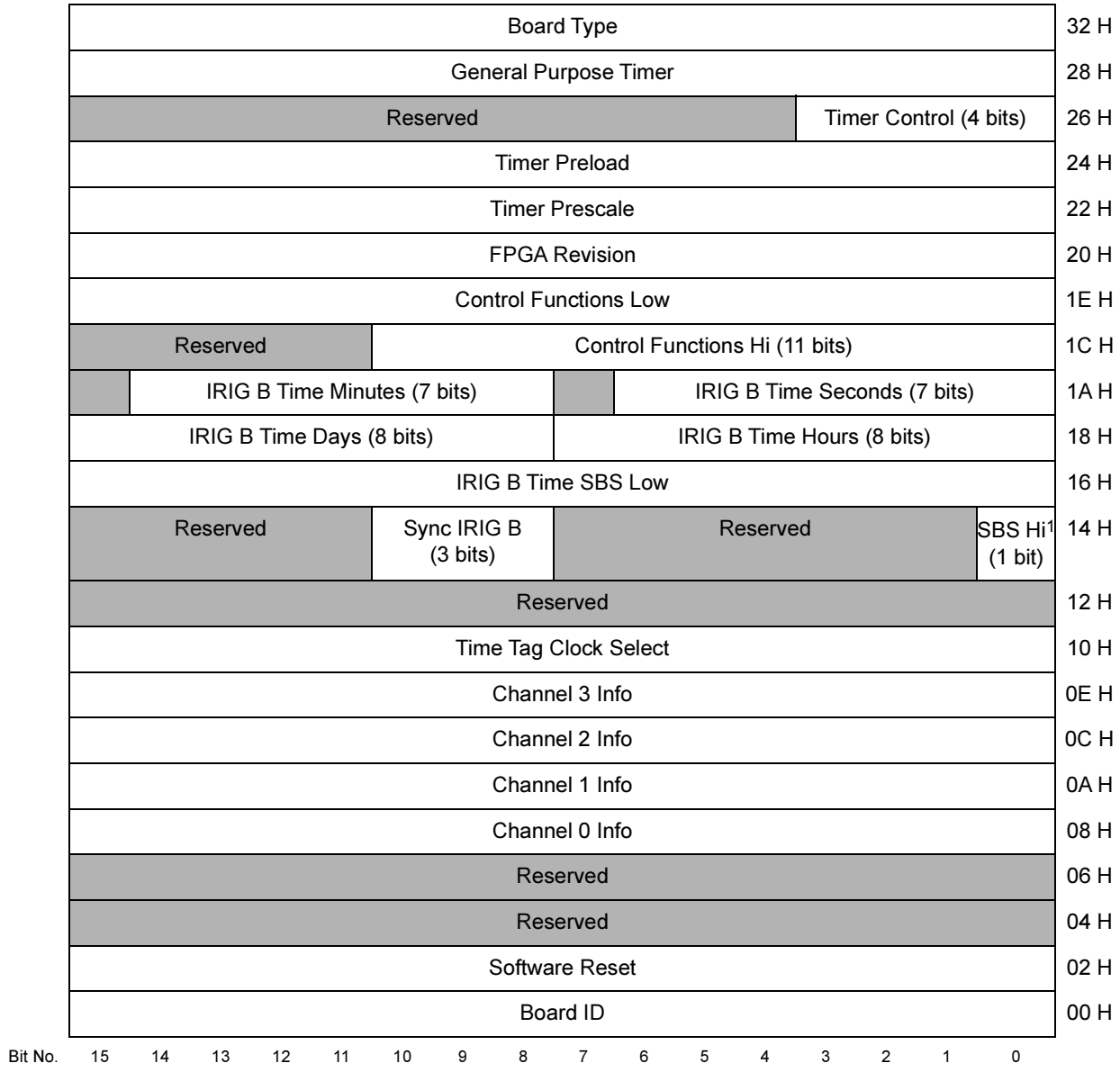


Figure 5-1 Host Interface Global and IRIG B Registers Map

1. IRIG B Time SBS Hi Register

5.1.1 Board Identification Register

Address: 00 (H)
Length 16 bits

Read only The Board Identification register comprises the following identification items.

Bit	Description
00-15	Hard coded to the value 4000 H

Table 5-1 Board Identification Register

5.1.2 Software Reset Register

Address: 02 (H)
Length 16 bits

Read/Write The Software Reset register performs reset operations of the channels. Individual channels may be reset.

Bit 04, the Global Time Tag reset bit, resets all the channel's Time Tag counters.

Bit	Description	
05-15	Reserved – set to 0	
04	Global time tag reset	1 = reset all time tag counters 0 = no effect
03	Reserved – set to 0	
02	Channel 2 reset	1 = reset channel 0 = no effect
01	Channel 1 reset	1 = reset channel 0 = no effect
00	Channel 0 reset	1 = reset channel 0 = no effect

Table 5-2 Software Reset Register

5.1.3 Channel Info Registers

Address: 08, 0A, 0C, 0E (H)
Length 16 bits each

Read only The Channel Info Registers provide identification information for each of the channels. On the *UNET*, Channel 0 is always *M4K429RTx* and Channel 3 is always *M4KDiscrete*.

Bit	Description	
12-15	Channel ID	00 H = Channel 0 Info register 01 H = Channel 1 Info register 02 H = Channel 2 Info register 03 H = Channel 3 Info register
05-11	Reserved – set to 0	
00-04	Channel type	04 H = M4K429RTx module 0D H = M4KDiscrete channel 1F H = no channel installed

Table 5-3 Channel Info Registers

5.1.4 Time Tag Clock Select Register

Address: 10 (H)
Length 16 bits

Read/Write The Time Tag Clock Select Register is used to set either an internal (1 MHz) or external source for the board's Global Time Tag Clock. See section **4.3.5 External Signals Connector Pinouts** on page 4-9, for details of the External Time Tag Clock.

Bit	Description	
01-15	Reserved – set to 0	
00	Time Tag Clock Select	1 = External Source 0 = Internal Source [**Default**]

Table 5-4 Time Tag Clock Select Register

5.1.5 FPGA Revision Register

Address: 20 (H)
Length 16 bits

Read only The FPGA Revision register contains the FPGA revision of the board.

5.2 IRIG B Host Interface Global Registers

The *UNET* is able to receive and decode standard serial IRIG B120 time code format signals (1 KHz carrier wave, sine wave – amplitude modulated, 100 peaks per second). The *DAS-429UNET/RTx* receives IRIG B signals via its External Signals Connector. See **4.3.5 External Signals Connector Pinouts** on page 4-9.

The IRIG B signal, which contains 3 types of words within each Time Code Frame, can be used to synchronize the Time Tags of the channels on the *UNET*.

1st Word	Time-of-year in binary coded decimal (BCD) notation in hours, minutes and seconds.
2nd Word	Set of bits reserved for decoding various control, identification and other special purpose functions.
3rd Word	Seconds-of-day weighted in straight binary seconds (SBS) notation

These three words can be stored and displayed in the IRIG B Host Interface Global registers 14 - 1E (H).

See **Figure 5-1 Host Interface Global and IRIG B Registers Map** on page 5-2 for the location of the registers on the memory map.

Note: The synchronization of IRIG B time can take up to two seconds. IRIG B functions are meant to be used on an occasional basis, not on a constant basis.

5.2.1 Sync IRIG B Register

Address: 14 (H)
Bits 08 – 10

Read/Write The 3-bit Sync IRIG B register controls the synchronization of a channel's Time Tags relative to the IRIG B input signal and the display of the IRIG B time within the IRIG B time registers.

Bit		Description
10	1	Set by board to indicate that the current IRIG B time has been stored in the IRIG B registers
	0	No IRIG B time has been stored in the IRIG B registers. This bit must be reset by the user after the board has written a '1'.
09	1	Stores and displays the IRIG B time and control functions into the 6 IRIG B registers (14-1E [H]) corresponding to the previous valid IRIG B message. If bit 08 is set, then the IRIG B time will be stored at the same time that the Time tags are reset. To calculate the realtime to which the Time tags are synchronized the user will need to **add** '1' to the value of the IRIG B time stored into these registers.
	0	The previous valid IRIG B message should not be stored in the IRIG B registers. This bit will be automatically reset by the board after the storage of the IRIG B time.
08	1	Resets and synchronizes Time Tags of all the channels to the next rising edge of the on-time Reference Point **Pr** of the IRIG B signal. Also sets Bit 09 to a value of '1' in order to store and display the IRIG B time and control functions into the 6 IRIG B registers.
	0	No reset/synchronization of Time tags relative to the **Pr** of the IRIG B signal. This bit will be automatically reset by board after reset of time tags

Table 5-5 Sync IRIGB Register

Note: All bits are read and write.

5.2.2 IRIG B Time SBS High Register

Address: 14 (H)
Bit 0

Read only The IRIG B Time SBS High register contains the MSB of the 17 bit straight binary representation of the seconds-of-day code word within the IRIG B message.

5.2.3 IRIG B Time SBS Low Register

Address: 16 (H)
Bits 15 – 0

Read only The IRIG B Time SBS Low register contains the lower 16 bits of the 17 bit straight binary representation of the seconds-of-day code word within the IRIG B message.

5.2.4 IRIG B Time Days Register

Address: 18 (H)
Bits 15 – 6

Read only The IRIG B Time Days register contains the days value of the BCD time-of-year subword within the IRIG B coded message.

5.2.5 IRIG B Time Hours Register

Address: 18 (H)
Bits 5 – 0

Read only The IRIG B Time Hours register contains the hours value of the BCD time-of-year subword within the IRIG B coded message.

5.2.6 IRIG B Time Minutes Register

Address: 1A (H)
Bits 14 – 8

Read only The IRIG B Time Minutes register contains the minutes value of the BCD time-of-year subword within the IRIG B coded message.

5.2.7 IRIG B Time Seconds Register

Address: 1A (H)
Bits 6 – 0

Read only The IRIG B Time Seconds register contains the seconds value of the BCD time-of-year subword within the IRIG B coded message.

5.2.8 Control Functions Registers

Hi Register Address: 1C (H) / Bits 10 – 0
Low Register Address: 1E (H) / Bits 15 – 0

Read only The IRIG B time code formats reserve 27 bits known as Control Functions. The Control Functions are for user-defined encoding of various control, identification or other special purpose functions. No standard coding system exists. The control bits may be programmed in any predetermined coding system.

5.2.9 FPGA Revision Register

Address: 20 (H)
Bits 15 – 0

Read only The FPGA Revision register contains the FPGA revision of the board.

5.3 Global Timer Registers

See **Figure 5-1** on page 5-1 for location of the registers on the memory map.

5.3.1 Timer Prescale Register

Address: 22 (H)
Bits 15 – 0

Read/Write The Timer Prescale Register defines the resolution of the General Purpose Timer. It is based on the Global Time Tag Clock (nominally 1 MHz) and thus will give the General Purpose Timer resolution as follows:

Timer Prescale Register Value (DEC)	General Purpose Time Resolution (μsec)
0 or 1	1 (default)
2	2
3	3
•	•
•	•
•	•
10	10
•	•
•	•
•	•
65535	65535

Table 5-6 Timer Prescale/General Purpose Timer Resolution

Note: The Timer Prescale register can only be changed when the timer has been stopped.

5.3.2 Timer Preload Register

Address: 24 (H)
Bits 15 – 0

Read/Write The value stored in the Timer Preload Register sets the starting count value for the General Purpose Timer from which it will start to count down. The Timer Preload Register can only be changed while the timer is stopped and has a maximum count value of 65535.

Note: The General Purpose Timer will not start counting if a value of zero is stored into the Timer Preload Register.

Default value: 00 00

5.3.3 Timer Control Register

Address: 26 (H)
Bits 3 – 0

Read/Write The Timer Control Register is used to control the General Purpose Timer register. The value stored in bits 01 to 03 take effect when the General Purpose timer reaches a value of zero. Bit 00 is used to start and stop the General Purpose Timer. The values of bits 01 – 03 can only be changed when the General Purpose Timer register is stopped.

Default value: 00 00

Bit	Description		
04-15	Reserved - set to 0		
03	Global reset on count completed	1	Causes global reset of all installed channels
		0	No effect
02	Reserved		
01	Reload Mode	1	Reload Mode
		0	Non-reload/One-shot Mode
00	Start/Stop	1	Start
		0	Stop

Table 5-7 Timer Control Register

5.3.4 General Purpose Timer Register

Address: 28 (H)
Bits 15 – 0

Read Only The General Purpose Timer Register stores the current count value of the General Purpose Timer. The General Purpose Timer is controlled by the Timer Control Register. When the General Purpose Timer is started it will count down to zero, at which point either an interrupt can be generated and or all installed channels can be reset.

If the General Purpose Timer is in Reload Mode then the current value in Timer Preload Register will be stored into the General Purpose Timer and the timer will start to count down from this value.

If the General Purpose Timer is in non-reload/One Shot Mode, when it reaches zero it will stop and a value of zero will be displayed in the General Purpose Timer Register. In this case bit 00 (Start/Stop bit) of the Timer Control Register will automatically be set to zero in this case. If the General purpose Timer Register is then started it will start to count from the current Timer Preload Register value automatically (without the need to do a write to the Timer Preload Register).

At any point in time, the General Purpose Timer can be stopped at the current count value. When a start is then issued, the General purpose Timer will start to count down from this current count value. If the user wishes to stop the counter and start from the original preload value or from a new preload value, this value will need to be rewritten into the Timer Preload register prior to the restarting of the General Purpose Timer register.

Note: The maximum clock period of the General Purpose Timer is 4295 seconds (1 hour, 11min & 35 Seconds).

5.3.5 Board Type Register

Address: 32 (H)
Length 16 bits

Read only The Board Type register comprises the following items.

Bit	Description
00-15	Hard coded to the value 5502 H

Board Type Register

6 ARINC 429 Module Control Registers (Advanced)

Chapter 6 describes how to run the ARINC 429 module via the memory registers, which enables you to develop your own software tools. When using Excalibur's software tools, this chapter is not required.

The following topics are covered:

6.1 Getting Started

The module operation makes extensive use of pointers for setting up the size and location on both the receiver and data blocks, transmitter instruction stacks and receiver Look-up Tables. Each channel has its own pointer registers so that the unique memory areas may be allocated for each channel. It is also possible to share memory areas. For example, more than one receiver channel may point to and use the same Label Look-up Table that controls which labels will be stored by the channel.

The flowchart Figure 6-1 **Module Operation Flowchart** on page 6-4, describes the necessary steps to set up transmit and receive channels. For details how to operate each specific protocol, see the sections: **6.5 Transmit Channel Operation** on page 6-20 and **6.6 Receive Channel Operation** on page 6-29.

After power-up, perform the following steps:

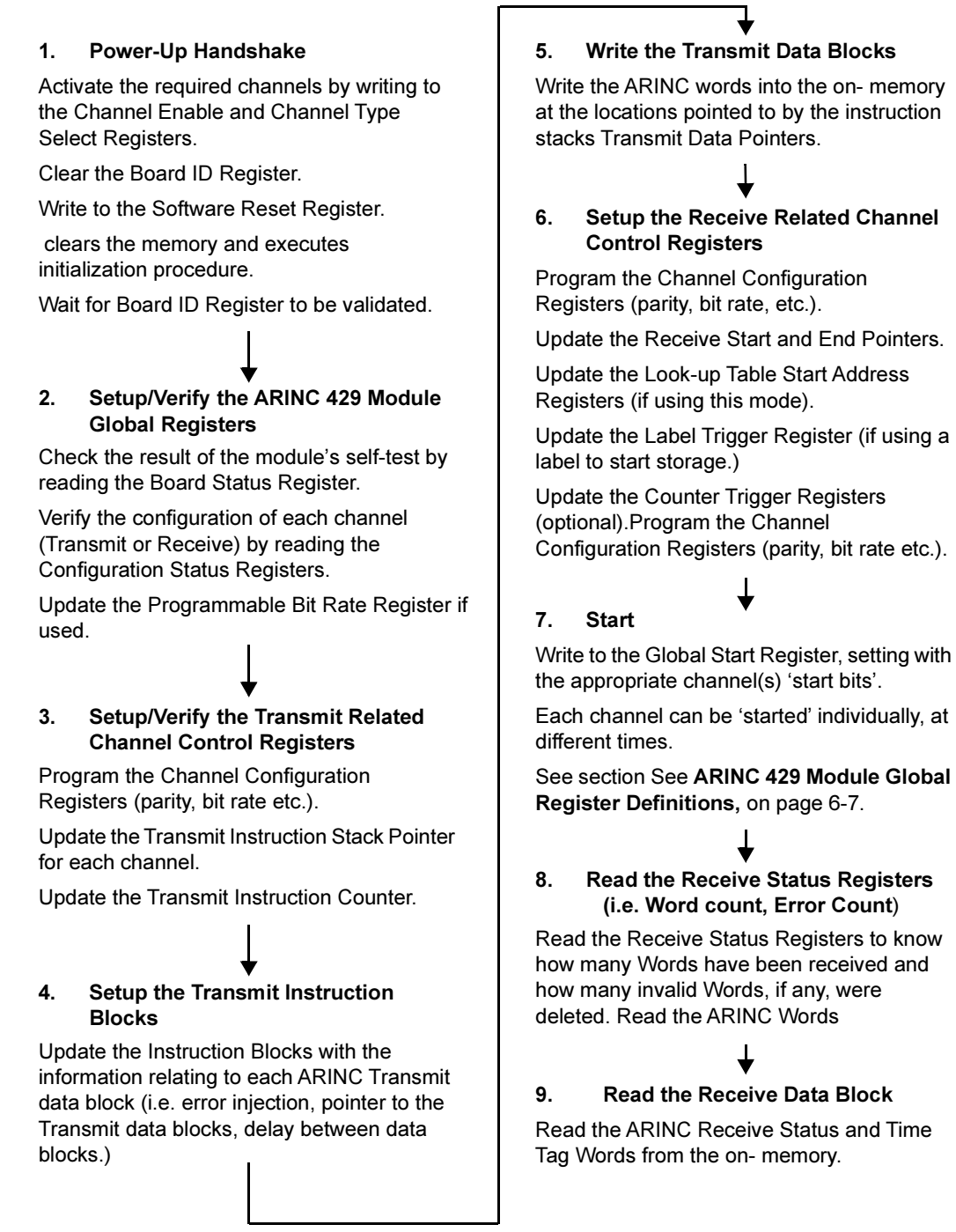


Figure 6-1 Module Operation Flowchart

6.2 ARINC 429 Module General Memory Map

The ARINC 429 Module General Memory Map is a summary of the major memory areas in the ARINC 429 module. Detailed memory maps are provided later in this chapter.

The ARINC 429 module occupies 64K x 8 of the module's 128K memory space. The 64K are divided into Control registers that reside at the upper end and the main memory block, which stores all the Stacks, Receive and Transmit Data Blocks and Look-up Tables.

Transmit Instruction Stack Transmit Data Blocks Receive Data Blocks Receive Look-Up Tables	00000 – 0F9FF H
Reserved	0FA00 – 0FD1F H
Channel Control Register Block 0	0FD20 – 0FD4E H
Channel Control Register Block 1	0FD50 – 0FD0FE H
Channel Control Register Block 2	0FD80 – 0FDAE H
Channel Control Register Block 3	0FDB0 – 0FDDE H
Channel Control Register Block 4	0FDE0 – 0FE0E H
Channel Control Register Block 5	0FE10 – 0FE3E H
Channel Control Register Block 6	0FE40 – 0FE6E H
Channel Control Register Block 7	0FE70 – 0FE9E H
Channel Control Register Block 8	0FEA0 – 0FECE H
Channel Control Register Block 9	0FED0 – 0FEFE H
ARINC 429 Module Global Control Registers	0FF00 – 0FFFF H
Reserved Memory Space	10000 – 1FFFF H

Figure 6-2 ARINC 429 Module General Memory Map

6.3 ARINC 429 Module Global Registers Memory Map

The ARINC 429 Module Global Registers are registers that affect all ARINC 429 channels.

Reserved	0FF00–0FF37 H		
Module Options	0FF38	Channel 5 Configuration Status	0FF64 H
Reserved	0FF3A	Channel 6 Configuration Status	0FF66 H
Interrupt Status Busy	0FF3C H	Channel 7 Configuration Status	0FF68 H
Receive Merge Status	0FF3E H	Channel 8 Configuration Status	0FF6A H
Receive Merge Interrupt/Trigger Condition	0FF40 H	Channel 9 Configuration Status	0FF6C H
Receive Merge Configuration	0FF42 H	Programmable Bit Rate	0FF6E H
Receive Merge Label Trigger	0FF44 H	Interrupt Status	0FF70 H
Receive Merge Error Count	0FF46 H	Firmware Revision	0FF72 H
Receive Merge Interval Count Trigger	0FF48 H	Board Status	0FF74 H
Receive Merge Buffer Wraparound	0FF4A H	Board ID	0FF76 H
Receive Merge Word Count Trigger	0FF4C H	Reserved	0FF78 H
Receive Merge Word Count	0FF4E H	Start / Stop	0FF7A H
Receive Merge Filter Table Start Address	0FF50 H	Time Tag Reset	0FF7C H
Receive Merge Current Pointer	0FF52 H	Reserved	0FF7E – 0FF82 H
Receive Merge End Pointer	0FF54 H	Software Reset	0FF84 H
Receive Merge Start Pointer	0FF56 H	Reserved	0FF86 – 0FF8E H
Receive Data Storage Mode	0FF58 H	Time Tag Counter	0FF90 – 0FF92 H
Channel 0 Configuration Status	0FF5A H	Hardware Revision Register	0FF94 H
Channel 1 Configuration Status	0FF5C H	Reserved	0FF96 H
Channel 2 Configuration Status	0FF5E H	Channel Enable	0FF98 H
Channel 3 Configuration Status	0FF60 H	Channel Type Select	0FF9A H
Channel 4 Configuration Status	0FF62 H	Reserved	0FF9C–0FFFF H

Figure 6-3ARINC 429 Module Global Registers Memory Map

6.4 ARINC 429 Module Global Register Definitions

6.4.1 Module Options Register

Address: 0FF38 (H)

The Module Options Register shows whether the module is Nios-based. There are several features that are only available on Nios-based modules, use this register to determine if those features are available.

Bit	Bit Name	Description
03-15	Reserved – set to 0	
02	Nios II Processor	1 = This is a newer Nios-based module. 0 = This is an older Motorolla-based module.
00-01	Reserved – set to 0	

Module Options Register

6.4.2 Interrupt Status Busy Register

Address: 0FF3C (H)

The Interrupt Status Busy Register indicates if the Channel *x* Status Register, the Receive Merge Status Register and the Interrupts Status Register may be accessed by the user.

Bit	Bit Name	Description
11-15		Reserved
10	**MRGBSY**	1 = Receive Merge Status Register is busy – do not access the register 0 = The contents of the register is valid and may be accessed
09	**CH9BSY**	1 = Channel 9 Status Register and the Global Interrupt Status Register are busy – do not access 0 = Channel 9 Status Register may be accessed
08	**CH8BSY**	1 = Channel 8 Status Register and the Global Interrupt Status Register are busy – do not access 0 = Channel 8 Status Register may be accessed
07	**CH7BSY**	1 = Channel 7 Status Register and the Global Interrupt Status Register are busy – do not access 0 = Channel 7 Status Register may be accessed
06	**CH6BSY**	1 = Channel 6 Status Register and the Global Interrupt Status Register are busy – do not access 0 = Channel 6 Status Register may be accessed
05	**CH5BSY**	1 = Channel 5 Status Register and the Global Interrupt Status Register are busy – do not access 0 = Channel 5 Status Register may be accessed
04	**CH4BSY**	1 = Channel 4 Status Register and the Global Interrupt Status Register are busy – do not access 0 = Channel 4 Status Register may be accessed
03	**CH3BSY**	1 = Channel 3 Status Register and the Global Interrupt Status Register are busy – do not access 0 = Channel 3 Status Register may be accessed
02	**CH2BSY**	1 = Channel 2 Status Register and the Global Interrupt Status Register are busy – do not access 0 = Channel 2 Status Register may be accessed

Interrupt Status Busy Register

Bit	Bit Name	Description
01	CH1BSY	1 = Channel 1 Status Register and the Global Interrupt Status Register are busy – do not access 0 = Channel 1 Status Register may be accessed
00	CH0BSY	1 = Channel 0 Status Register and the Global Interrupt Status Register are busy – do not access 0 = Channel 0 Status Register may be accessed

Interrupt Status Busy Register (Continued)

Note: Before accessing the Global Interrupt Register, the user should wait until the Interrupt Status Busy Register = 0. The user then has at least 15 msec. to safely access the status registers.

6.4.3 Receive Merge Status Register

Address: 0FF3E (H)

The Receive Merge Status Register indicates the operational status of the Merge Mode receive buffer. This register can be used to poll the status of the channel or it can be used with interrupts. When used in conjunction with interrupts the register indicates the condition or conditions, which caused the interrupt.

A logic 1 indicates an active bit. Writing a 0 to this register resets the status bits.

Bit	Bit Name — Interrupt Causes
07-15	Reserved
06	Stopped on Buffer Full
05	Error Word Received
04	Word Count Trigger
03	Interval Count Trigger
02	Label Received
00-01	Reserved

Receive Merge Status Register

Note: The Label Received status bit is set upon receipt of any label for which an interrupt has been requested via the Filter Table.

6.4.4 Receive Merge Interrupt/Trigger Condition Register

Address: 0FF40 (H)

The Receive Merge Interrupt/Trigger Condition Register sets the Interrupt and Trigger condition or conditions for receive channels in Merge Mode. The Trigger Conditions set a pulse on the External Trigger Output.

Trigger Condition Bits 08 – 15		Interrupt Condition Bits 00 – 07	
Bit	**Bit Name**	**Bit**	**Bit Name**
15	Reserved	**07**	Reserved
14	Stopped on buffer full	**06**	Stopped on buffer full
13	Error Word received	**05**	Error Word received
12	Data Word Count Trigger	**04**	Data Word Count Trigger
11	Interval Count Trigger	**03**	Interval Count Trigger
10	Label Received	**02**	Label Received
08–09	Reserved	**00–01**	Reserved

Receive Merge Interval/Trigger Condition Register

Note:
1. The Label Received interrupt or trigger only occurs upon reception of a label that has been marked for interrupt in the Filter Table.
2. To activate the Interval Count Trigger interrupt or trigger, the Receive Merge Interval Count Trigger Register must also be set. (see **Receive Merge Interval Count Trigger Register** on page 6-10).
3. To activate Data Word Count Trigger interrupt or trigger, the Receive Merge Word Count Trigger Register must also be set. (see **Receive Merge Word Count Trigger Register** on page 6-11).

6.4.5 Receive Merge Configuration Register

Address: 0FF42 (H)

The Receive Merge Configuration Register sets up run parameters for Merge Mode.

Bit	Bit Name	Description
10-15	**Reserved**	Set to 0
09	**Enable Receive Filter Table**	1 = Enable Filter table. Stores labels per table 0 = Disables table. Stores all labels.
08	**Reserved**	Set to 0
07	**Receive Label Trigger**	1 = Start data storage upon receipt of label *xx*. (see **Receive Merge Label Trigger Register**) 0 = Receive stores data without Start Label Trigger.
06	**Receive Wrap Around**	1 = Data storage is halted when the buffer is full. 0 = Receive wraps around the data in the block.
00-05	**Reserved**	Set to 0

Receive Merge Configuration Register

6.4.6 Receive Merge Label Trigger Register

Address: 0FF44 (H)

The Receive Merge Label Trigger Register is used in conjunction with the Receive Label Trigger bit in the Receive Merge Configuration Register to begin the reception and storage of data upon receipt of a unique ARINC label.

The module will not store any ARINC words received prior to the first instance of this label.

Bit	Description
08–15	Set to 0
00–07	Trigger Label

Receive Merge Label Trigger Register

6.4.7 Receive Merge Error Count Register

Address: 0FF46 (H)

The Receive Merge Error Count Register is a 16-bit counter. The register indicates the number of error words received on the channel. This counter Register wraps around and is reset only by the user.

6.4.8 Receive Merge Interval Count Trigger Register

Address: 0FF48 (H)

The Receive Merge Interval Count Trigger Register (a 16-bit value) allows the user to generate an interrupt (or pollable bit) every *N* number of words, where *N* is the value written to this register. For example, to request an interrupt after every five ARINC words, write 05 to this register.

To generate an interrupt or trigger, the appropriate bit must also be set in the Receive Merge Interrupt Condition Register (see **Receive Merge Interrupt/Trigger Condition Register** on page 6-9).

6.4.9 Receive Merge Buffer Wraparound Register

Address: 0FF4A (H)

The Receive Merge Buffer Wraparound Register contains 2 bits for synchronization with the host.

Bit	Description
15	Multiple Wraparound - data lost
14	Single Wraparound - the receive buffer has wrapped around once since the last data read
00–13	Reserved

Receive Merge Buffer Wraparound Register

6.4.10 Receive Merge Word Count Trigger Register Address: 0FF4C (H)

The Receive Merge Word Count Trigger Register sets a trigger (used for polling or interrupts) and a flag that indicates when a specific number of words have been received (1 - 65535).

To generate a trigger or interrupt, the appropriate bit in the Receive Merge Interrupt/ Trigger Condition Register must also be set. (see **Receive Merge Interrupt/ Trigger Condition Register** on page 6-9).

Note: This trigger is set when the value in the Receive Merge Word Counter matches the value set in this register.

6.4.11 Receive Merge Word Counter Address: 0FF4E (H)

The Receive Merge Word Counter indicates the number of ARINC words received (0 - 65535). This register wraps around to 0 after it reaches 65535. The user can reset the register only when the channel is stopped.

6.4.12 Receive Merge Filter Table Start Address Address: 0FF50 (H)

The Receive Merge Filter Table Start Address sets the start address of the (256×8) Label Filter Table as described in the Sequential Storage Mode (see section **Receive Buffer Storage Sequence** on page 6-30.) The address must be on a word boundary.

6.4.13 Receive Merge Current Pointer Address: 0FF52 (H)

The Receive Merge Current Pointer indicates the current address where the next ARINC receive word is to be placed in the Receive buffer. This pointer value is incremented after the entire receive block (ARINC word, Time Tag, and Status) is written into memory.

6.4.14 Receive Merge End Pointer Address: 0FF54 (H)

The Receive Merge End Pointer sets the End Address of the Receive Data buffer. The data will wrap around or stop when the buffer is full, (when the End Address is reached), depending upon the contents of the Receive Merge Configuration Register Wraparound bit. (See **Receive Merge Configuration Register** on page 6-9).

6.4.15 Receive Merge Start Pointer Address: 0FF56 (H)

The Receive Merge Start Pointer sets the Start Address of the Receive Data buffer. The address must be on a word boundary within the Receive Data Blocks area.

Example: To cause the Merge buffer to begin at byte offset 001A0 (H), write a 01A0 (H) to this register.

6.4.16 Receive Data Storage Mode Register

Address: 0FF58 (H)

The Receive Data Storage Mode register is used to select the Receive Data Storage Mode and the Merge Mode option. ARINC Data Words can be stored with, or without, Time Tag and Status words appended to the data block. The Merge Mode Control Registers are used only when the Merge Mode option is selected.

Bit	Bit Name	Description
05-15	Reserved – set to 0	
04	IRIG B Time Tag	0 = Use a 32-bit, 10 microsecond precision, Time Tag. 1 = Use a 64-bit IRIG B Time Tag. In order to use an IRIG B Time Tag, the module must be mounted on a carrier board that has IRIG B wired to the module, and the carrier board must be receiving an IRIG B signal. The IRIG B input must be using the standard IRIG B120 serial time code.
02-03	Reserved – set to 0	
01	Merge Mode option	0 = Sequential Mode – stores data sequentially per channel, utilizing different receive buffer areas for each channel. 1 = Merge Mode – utilizes a single receive buffer for all channels. Each Receive Status Word is tagged with Channel Code information.
00	Receive Data Storage Mode option	0 = Standard Mode – appends both Time Tag and Status Words to each ARINC word stored in memory 1 = Store Only Data

Receive Data Storage Mode Register

Note: 1. If Data Only Storage Mode is selected (bit 00 set to 1), storage will be per independent channel regardless of the state of bit 01.

2. Data Only Storage Mode is not available in Look-up Table Mode.

3. The Receive Data Storage Mode Register can only be changed when all the channels are turned off for at least 1 msec. (Stop/Start register = 0).

6.4.17 Channel *x* Configuration Status Register

Address: 0FF5A – 0FF6C (H)

These Registers indicate to the host the type of channel configured in each channel socket on the module.

Bit	Description				
04-15	Reserved – set to 0				
00-03	Configuration Status Code				
	Bit 3	Bit 2	Bit 1	Bit 0	Configuration Status Code
	0	0	0	0	Undefined Channel
	0	0	0	1	ARINC-429 Receive Channel
	0	0	1	0	ARINC-429 Transmit Channel

Channel *x* Configuration Status Register

6.4.18 Programmable Bit Rate Register

Address: 0FF6E (H)

The Programmable Bit Rate Register selects the programmable bit rate value for the ARINC channels. This register is only read by the firmware when the Start/ Stop Register contains a value of 0, i.e. all channels are inactive for at least 1 msec.

Bit	Description
15	Set to 1
11-14	Reserved – set to 0
00-10	Bit Rate value

Programmable Bit Rate Register

To calculate the Bit Rate Value (BRV)

$$BRV = \frac{5000}{\text{freq (KHz)}} - 1$$

Example: Desired programmable bit rate 12.5 KHz:

$$BRV = \frac{5000}{12.5\text{KHz}} - 1 = 399\text{Dec (018F H)}$$

Write the value 818F (H) to this register.

6.4.19 Interrupt Status Register

Address: 0FF70 (H)

The Interrupt Status Register indicates which channel issued the interrupt: 1 = Active. The status bit or bits are only reset by the user.

Bit	Bit Name
10-15	Reserved – set to 0
09	Channel 9 Interrupt Status Bit
08	Channel 8 Interrupt Status Bit
07	Channel 7 Interrupt Status Bit
06	Channel 6 Interrupt Status Bit
05	Channel 5 Interrupt Status Bit
04	Channel 4 Interrupt Status Bit
03	Channel 3 Interrupt Status Bit
02	Channel 2 Interrupt Status Bit
01	Channel 1 Interrupt Status Bit
00	Channel 0 Interrupt Status Bit

Interrupt Status Register

6.4.20 Firmware Revision Register

Address: 0FF72 (H)

The Firmware Revision Register indicates the revision level of the firmware. For example: 0114 (H) = Rev 1.14.

6.4.21 Board Status Register

Address: 0FF74 (H)

The Board Status Register indicates the result of the module's self-test. The self-test is performed on module power-up and after a software reset. However, on power-up the Channel Status bits (bits 00 – 09) are all set to 0. To test the channels, you must enable the desired channels, select whether the channels should be transmit or receive, then perform a software reset. (See **Channel Enable Register** on page 6-18, **Channel Type Select Register** on page 6-19 and **Software Reset Register** on page 6-17.) After a software reset, the Channel Status bit is set to 1 for each channel that tested OK.

Bit	Bit Name	
14–15	Reserved	Set to 0
13	Extended Time Support	1 = Extended Interblock Time/Data Rate Mode supported 0 = Not supported (See **Interblock Time/Data Rate Word** on page 6-23.)
12	Baud Rate Generator	1 = 40 MHz
11	Host Ready Timeout	
10	Memory Status Bit	1 = Memory OK 0 = Memory failed
09	Channel 9 Status bit	1 = Channel present and self-test OK 0 = Channel not present or self-test failed
08	Channel 8 Status bit	1 = Channel present and self-test OK 0 = Channel not present or self-test failed
07	Channel 7 Status bit	1 = Channel present and self-test OK 0 = Channel not present or self-test failed
06	Channel 6 Status bit	1 = Channel present and self-test OK 0 = Channel not present or self-test failed
05	Channel 5 Status bit	1 = Channel present and self-test OK 0 = Channel not present or self-test failed
04	Channel 4 Status bit	1 = Channel present and self-test OK 0 = Channel not present or self-test failed
03	Channel 3 Status bit	1 = Channel present and self-test OK 0 = Channel not present or self-test failed
02	Channel 2 Status bit	1 = Channel present and self-test OK 0 = Channel not present or self-test failed
01	Channel 1 Status bit	1 = Channel present and self-test OK 0 = Channel not present or self-test failed
00	Channel 0 Status bit	1 = Channel present and self-test OK 0 = Channel not present or self-test failed

Board Status Register

Note:

- The module continues to operate on condition of channel self-test failure, but will not operate on condition of memory failure.
- To check whether a channel is configured as receive or transmit, read the Channel *x* Configuration Register. See **Channel x Configuration Register** on page 6-56.

6.4.22 Board ID Register

Address: 0FF76 (H)

The Board ID register indicates that the module has completed its initialization sequence and that the module is ready to be accessed by the host. The module writes the value A429 (H) into this register when ready.

6.4.23 Start/Stop Register

Address: 0FF7A (H)

The user can start one or more channels at the same time — wait a minimum of 500 μsec. between writes to this register.

Bit	Description	
10-15	Reserved – set to 0	
09	Channel 9 Start Bit	1 = Start 0 = Stop
08	Channel 8 Start Bit	1 = Start 0 = Stop
07	Channel 7 Start Bit	1 = Start 0 = Stop
06	Channel 6 Start Bit	1 = Start 0 = Stop
05	Channel 5 Start Bit	1 = Start 0 = Stop
04	Channel 4 Start Bit	1 = Start 0 = Stop
03	Channel 3 Start Bit	1 = Start 0 = Stop
02	Channel 2 Start Bit	1 = Start 0 = Stop
01	Channel 1 Start Bit	1 = Start 0 = Stop
00	Channel 0 Start Bit	1 = Start 0 = Stop

Start/Stop Register

Note: Only after the Start/Stop Register contains a 0 for at least 1 msec. do any changes in a channel's Configuration, Programmable Bit Rate, or Receive Data Storage Mode Registers take effect on the module.

6.4.24 Time Tag Reset Register

Address: 0FF7C (H)

Write Writing any value to the Time Tag Reset Register causes the Time Tag to be reset to 0.

6.4.25 Software Reset Register Address: 0FF84 (H)

Write Writing a 0 to the Software Reset Register resets the module. Following a reset, the module executes a self-test, both memory and channels, clears all the dual-port RAM, resets the module's interrupts and updates the Board Status Register. The module then indicates that it is ready by writing a value of A429 (H) to the Board ID Register.

6.4.26 Time Tag Counter Address: 0FF90–0FF92 (H)

The two Time Tag Counter Words represent the current value of the free-running 32-bit Time Tag counter. The counter can be read at any time. The following sequence must be kept:

First: 0FF90 H (Lo word – LSB)
Last: 0FF92 H (Hi word – MSB)

The resolution of the counter is 10 μsec. The counter is reset to 0 upon power-up or software reset or through the Time Tag Reset register. After the reset operation, the counter starts counting. When the counter reaches the value FFFF FFFF (H), it wraps around to 0 and continues counting.

6.4.27 Hardware Revision Register Address: 0FF94 (H)

Read The Hardware Revision register indicates the module type and the revision level of the hardware.

Bit	Description	
12-15	Module type	5 = RT5 A = RT10
00-11	Hardware revision	021 H = Rev 2.1 022 H = Rev 2.2 023 H = Rev 2.3 ... ...

Hardware Revision Register

6.4.28 Channel Enable Register

Address: 0FF98 (H)

Write

The Channel Enable register is used to enable/activate the required channels. This register, in conjunction with the Channel Type Select register, must be set first, before any other operations are carried out on the module. This register is reset to 0000 after module reset.

Note: In order for the on-module micro controller to recognize the channel's definitions, a software reset has to be applied. (See **Software Reset Register** on page 6-17.)

Bit	Description	
10-15	Reserved – set to 0	
09	Channel 9 enable bit	1 = enabled 0 = disabled
08	Channel 8 enable bit	1 = enabled 0 = disabled
07	Channel 7 enable bit	1 = enabled 0 = disabled
06	Channel 6 enable bit	1 = enabled 0 = disabled
05	Channel 5 enable bit	1 = enabled 0 = disabled
04	Channel 4 enable bit	1 = enabled 0 = disabled
03	Channel 3 enable bit	1 = enabled 0 = disabled
02	Channel 2 enable bit	1 = enabled 0 = disabled
01	Channel 1 enable bit	1 = enabled 0 = disabled
00	Channel 0 enable bit	1 = enabled 0 = disabled

Channel Enable Register

6.4.29 Channel Type Select Register

Address: 0FF9A (H)

Write

The Channel Type Select Register sets each channel's type, either transmit or receive. This register, in conjunction with the Channel Enable Register, must be set prior to any other operations are carried out on the module. This register is reset to 0000 after module reset. See **Module Operation Flowchart,** on page 6-4.

Note: In order for the on-module micro-controller to recognize the channel's definitions, a software reset has to be applied. (See **Software Reset Register** on page 6-17.)

Bit	Description	
10-15	Reserved – set to 0	
09	Channel 9 type select bit	1 = transmit 0 = receive
08	Channel 8 type select bit	1 = transmit 0 = receive
07	Channel 7 type select bit	1 = transmit 0 = receive
06	Channel 6 type select bit	1 = transmit 0 = receive
05	Channel 5 type select bit	1 = transmit 0 = receive
04	Channel 4 type select bit	1 = transmit 0 = receive
03	Channel 3 type select bit	1 = transmit 0 = receive
02	Channel 2 type select bit	1 = transmit 0 = receive
01	Channel 1 type select bit	1 = transmit 0 = receive
00	Channel 0 type select bit	1 = transmit 0 = receive

Channel Type Select Register

6.5 Transmit Channel Operation

Each transmit channel has three basic modes of operation: Interblock Time Mode, Data Rate Mode and FIFO Mode.

Interblock Time Mode and Data Rate Mode are described in the following sections. FIFO Mode is described in **6.5.2 FIFO Mode** on page 6-25.

Note: In addition to the above modes, there is also a Translation Mode, which uses one channel to receive and one channel to transmit. For more information, see **6.7 Translation (Dual Channel) Operation** on page 6-38.

6.5.1 Interblock Time Mode and Data Rate Mode

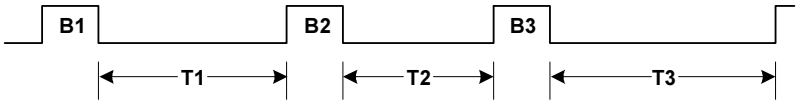
The transmit mode is selected via the Channel *x* Configuration register. See **Channel x Configuration Register** on page 6-56.

In Interblock Time and Data Rate modes, Words are sent out in blocks (groups of Words). Each block is sent with dedicated parameters including word count, interblock time (or data rate) and error injection. The blocks reside sequentially in stack structure, each pointing to its Data Buffer. In Interblock Time Mode, the blocks can be sent either continuously or *N* times. In Data Rate Mode, blocks are always sent continuously.

Interblock Time Mode and Data Rate Mode are recommended for use when transmitting the same data repeatedly or when the data to transmit needs to be updated in real-time. For transmitting a large amount of known data, such as a file, it recommended to use FIFO Mode. (See **6.5.2 FIFO Mode** on page 6-25.)

Note: When using Data Rate Mode, the value of the Channel *x* Transmit Loop Counter register is ignored and the data is transmitted continuously. See **Channel x Transmit Loop Counter** on page 6-52.

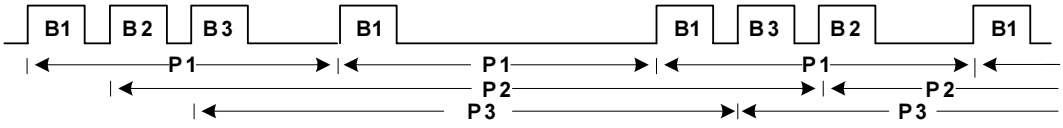
Interblock Time Mode sends out blocks of Words sequentially; each block has its own Interblock delay time.



B - Block
T - Interblock time

Figure 6-4 Interblock Time Mode

Data Rate Mode sends out blocks of Words periodically; each block is sent according to it's own data rate.



B - Block
T - Interblock time
P - Block period time

Figure 6-5 Data Rate Mode

To use Interblock Time Mode or Data Rate Mode, set bits 11 and 12 of the Channel x Configuration Register to 0, and select either Interblock Time Mode or Data Rate Mode by setting Bit 08 of this register to 0 or 1.

In these modes, the user:

- Sets up the transmitting channel's parameters by writing to the various Channel *x* Control Registers.
- Creates an Instruction Stack for the transmitting channel.
- Writes the data into the dual-port RAM.
- Starts transmission by writing to the Start/Stop Register found within the ARINC 429 Module Global Control Register area.

The sequence of writes to memory is not important, except for the write to Start Register operation, which is performed last.

Transmit Instruction Block

The Transmit Instruction Stack is divided into Instruction Blocks, each containing four Words. Each Instruction Block relates to a Data Buffer. A Data Buffer contains one or more ARINC Words which the user desires to transmit with the same amount of delay time between each Word. The stack is sequential, so that the first Instruction Block follows the second Data Block, and so on.

The Instruction Block consists of:

1st Word: The Control Word, which contains error injection parameters.

2nd Word: Contains 2 bytes:

- An 8-bit Word Count which instructs the module as to the number of ARINC Words to transmit within a particular block, and
- An 8-bit, inter-word delay value which programs the time between Words within the same buffer.

3rd Word: Contains a 16-bit, user-supplied data pointer. This is a 16-bit address (must be a Word boundary) that points to the beginning of the Data Words within the memory.

4th Word: Is the Interblock Time/Data Rate value and is used to program the time between Data Block transfers or the transmission period for the specific data block.

Figure 6-6 illustrates the Transmit Instruction Stack structure.

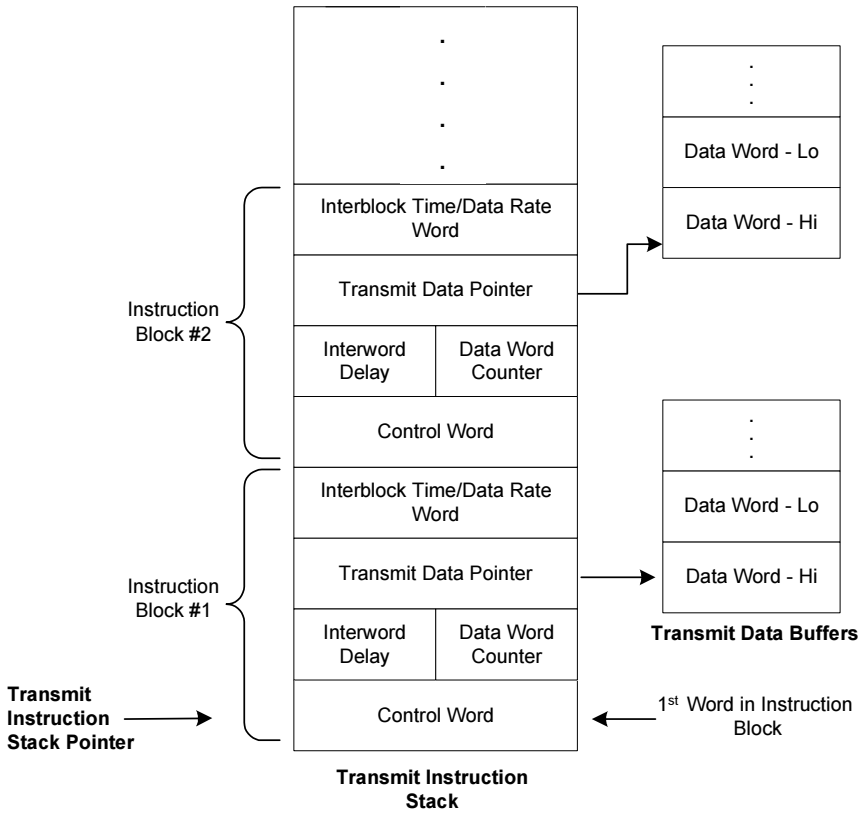


Figure 6-6 Transmit Instruction Stack Structure

Control Word Definition

This is the 1st Word in the ARINC Transmit Instruction Block.

Bit	Bit Name	Description
06-15	**Reserved**	Set to 0
05	**Suppress Parity [1]**	1 = Forces a no-parity condition within the Word even though 'parity-on' has been selected in the Channel *x* Configuration Register 0 = Regular parity as set up in the Channel *x* Configuration Register
04	**Bit Count Lo Error**	1 = Thirty-one ARINC bits are transmitted within each Word within the block 0 = No error
03	**Bit Count Hi Error**	1 = Thirty-three ARINC bits are transmitted within each Word within the block 0 = No error
02	**Stretch Bit Error**	1 = The 2nd ARINC bit within each Word within the block is 'stretched' causing a Manchester coding error. 0 = No error
01	**Null Bit Error**	1 = A Null Bit Error is inserted within the second bit of each Word within the block (ARINC Bit 02). 0 = No error
00	**Parity Error**	1 = A Parity Error is inserted within all ARINC Words within the block. 0 = No error

Control Word Definition

1. The **Suppress Parity** is useful when most of the Data Buffers contain standard data with parity and a few buffers contain data type without parity. In such a case, the parity can be set ON for all Words within the Channel x Configuration Register while exceptional cases, can be forced to NO PARITY, using this bit.

Interword Delay/Data Word Counter Word

This Word is divided into two bytes, the Interword Delay (high byte) and the Data Word Counter (low byte).

The Interword Delay byte specifies the time between Words within this data block. The resolution is in the form of 'bit times' according to the transmission bit rate.

Bit Rate	Resolution
Lo-Speed	80μsec/bit
Hi-Speed	10μsec/bit
Programmable	$\frac{1}{\text{Programmable Bit Rate (MHz)}}$

Interword Delay Resolution

Note: The ARINC specification defines the minimum Interword time as 4 bit times, so values less that 4 may be interpreted on the receiving side as Sync error.

The Word Counter byte specifies the number of data Words within this Data block (1-255).

Transmit Data Pointer

The Transmit Data Pointer is used to set the start address of the block's Transmit Data Buffer. The address must be a Word boundary. The size of the buffer is determined by the Data Word Count value.

Interblock Time/Data Rate Word

The Interblock Time/Data Rate Word has two functions:

- In **Interblock Time Mode,** the Instruction Blocks are accessed sequentially and their associated Data Words transmitted according to this sequential order. The Interblock Time allows the user to specify the time between Data Blocks. It is inserted after the block transmission.
- In **Data Rate Mode**, the user can specify the transmission period of the particular Data Block. If the number *N* is written to this location, then the message is transmitted every *N* bit times.

The resolution of this 16-bit Word is according to the bit rate selected in the Channel *x* Configuration register, which is the same as in the Interword Delay byte. (For a detailed explanation of the resolution, see **Interword Delay/Data Word Counter Word** on page 6-23).

The maximum value for the Interblock Time/Data Rate is 65,535 bit times, which is 655 msec. in Hi-speed and 5,243 msec. in Lo-speed. When a longer Interblock Time/Data Rate is required, the user can enable Extended Time Mode. In Extended Time Mode, the actual Interblock Time/Data Rate implemented on the

bus will be twice the specified value. (See **Channel x Configuration Register** on page 6-56.)

Note: In Data Rate Mode, a scratch buffer for the firmware must be allocated via the **Channel x Scratch Buffer Start Register** and **Channel x Scratch Buffer End Register,** see page 6-51. This buffer must be at least eight bytes long for each block defined.

Transmit Data Block Format

Figure 6-7 illustrates the format of the Transmit Data Words within the memory.

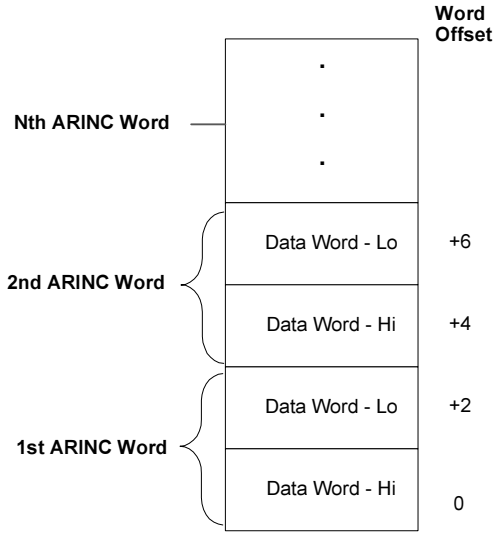


Figure 6-7 Transmit Data Block

Figure 6-8 defines the locations and bit definitions of the data bytes within the memory. The numbers shown in the two Words represent the ARINC Word bit numbers.

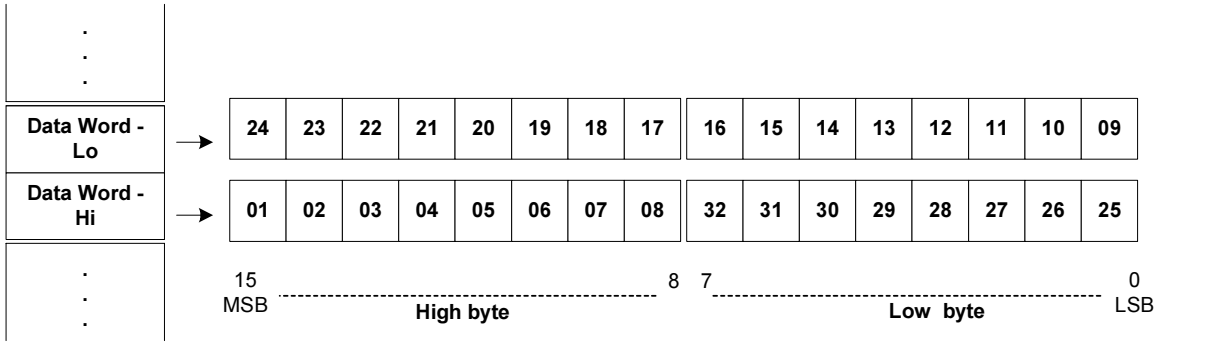


Figure 6-8 32-Bit Transmit Data Words Format

1. The numbers contained within the Words in Figure 6-8 represent the ARINC bit locations within the 32-bit Word.
2. The ARINC Word bits are transmitted in the following order:

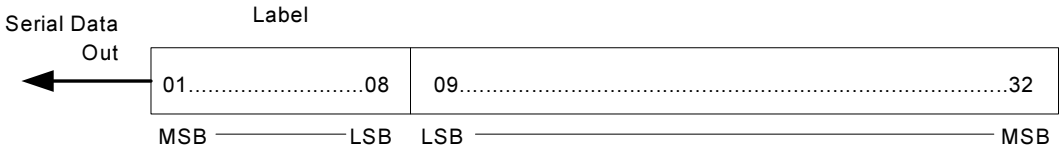


Figure 6-9 32-Bit Word Transmit Order Bit

3. Bits 09 through 32 are ordered from LSB to MSB (as opposed to the Label field that is organized from MSB to LSB). For this reason, in the Data Block, the Hi-Word is first, followed by the Lo-Word, with the label and the ARINC field 32 through 25 in the Hi-Word and bits 24 through 09, in the Lo-Word.

6.5.2 FIFO Mode

FIFO Mode is selected via the Channel *x* Configuration register. See **Channel x Configuration Register** on page 6-56.

In this mode, the user may define multiple buffers for each channel. Each buffer is used to transmit labels at a selected frequency. If a buffer is defined to transmit at a frequency of 50 milliseconds, the labels in that buffer will be transmitted at a rate of one 32-bit word every 50 milliseconds. If the length of the buffer is set to ten words, the first word will be sent at time 0,the second 50 milliseconds later and so on, until the final word is sent 450 milliseconds after the first. While these words are being transmitted, the user may refill the buffer, thereby creating a situation in which data is being transmitted at a constant rate. The user only needs to update the buffer every 500 milliseconds.

FIFO Mode is recommended for use when transmitting a large amount of known data, such as a file. When transmitting the same data repeatedly or when the transmitted data changes in real-time, it is recommended to use Interblock Time Mode or Data Rate Mode. (See **6.5.1 Interblock Time Mode and Data Rate Mode** on page 6-20.)

Note: When using FIFO Mode, the value of the Channel *x* Transmit Loop Counter register is ignored and the data is transmitted continuously. See **Channel x Transmit Loop Counter** on page 6-52.

In this mode, the user:

- Sets up the transmitting channel's parameters by writing to the various Channel *x* Control Registers.
- Allocates the number of buffers desired for each channel.
- Creates the buffers.
- Writes the data into the buffers.
- Starts transmission by writing to the Start/Stop Register found within the ARINC 429 Module Global Control Register area.

The sequence of writes to memory is not important, except for the write to Start Register operation, which is performed last.

Allocating Space for Buffer Pointers

Set the Transmit Instruction Stack Pointer to point to an array of 16-bit words, which will be used as pointers to the buffers. This array is called the array of buffer pointers.

Set the Transmit Instruction Counter to the number of buffers that will be created.

Allocating Buffers

Set an entry within the array of buffers to point to the buffer to be allocated. Allocate space for a 10 word header plus 3 words for each message in the buffer, with each word containing 16 bits. For example, for a 15 message buffer, allocate 55 16-bit words.

Figure 6-10 illustrates the structure of one buffer.

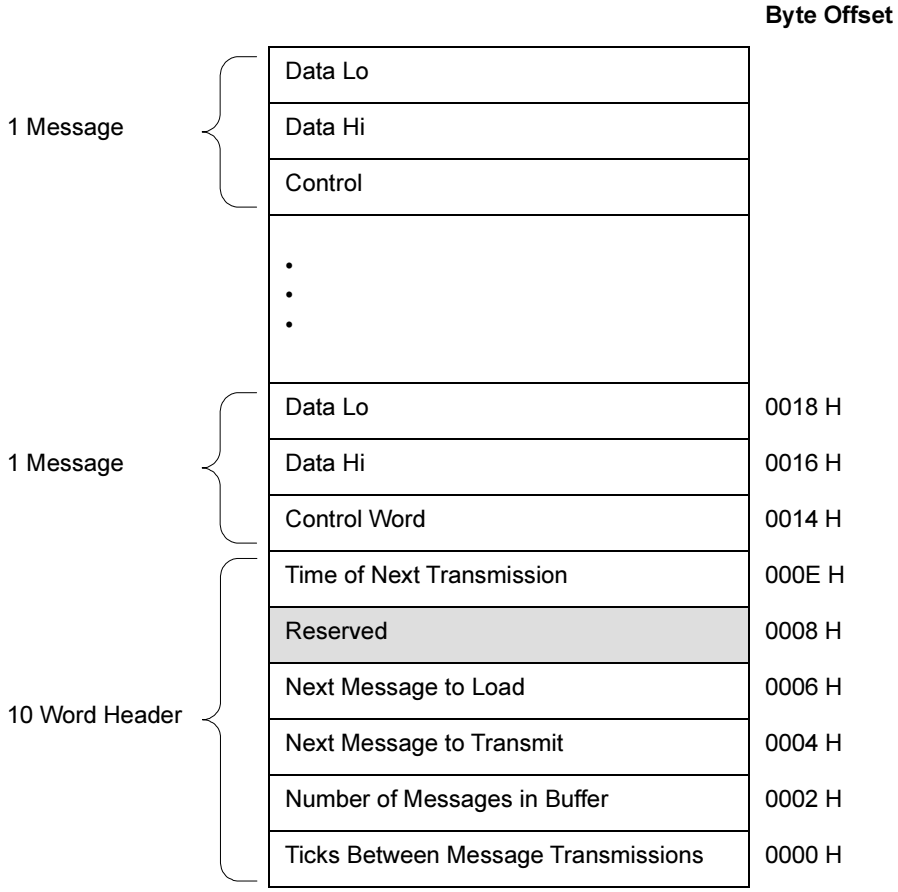


Figure 6-10 Buffer Structure

Ticks Between Message Transmissions **Byte Offset: 0000 (H)**

The number of ticks between the transmission of each message in the buffer. Each tick is 100 microseconds. This allows up to six seconds between message transmission.

Number of Messages in Buffer **Byte Offset: 0002 (H)**

How many messages are in the buffer. When this number is reached, the module wraps around to the start of the message area.

Next Message to Transmit **Byte Offset: 0004 (H)**

The index of next word to transmit. This must be initialized to 0. It is updated by the firmware.

Next Message to Load **Byte Offset: 0006 (H)**

Index of the next message that **Load_Transmit_FIFO_RTx** will load. This is not used by the firmware but is here for the use by the host software.

Time of Next Transmission **Byte Offset: 000E (H)**

Time Tag of the next transmission. This must be initialized to 0. It is updated by the firmware.

Control Word **Byte Offset: 0014 (H)**

Bit	Bit Name	Description
06-15	**Reserved**	Set to 0
05	**Suppress Parity** [1]	1 = Forces a no-parity condition within the Word even though 'parity-on' has been selected in the Channel *x* Configuration Register 0 = Regular parity as set up in the Channel *x* Configuration Register
04	**Bit Count Lo Error**	1 = Thirty-one ARINC bits are transmitted within each Word within the block 0 = No error
03	**Bit Count Hi Error**	1 = Thirty-three ARINC bits are transmitted within each Word within the block 0 = No error
02	**Stretch Bit Error**	1 = The 2nd ARINC bit within each Word within the block is 'stretched' causing a Manchester coding error. 0 = No error
01	**Null Bit Error**	1 = A Null Bit Error is inserted within the second bit of each Word within the block (ARINC Bit 02). 0 = No error
00	**Parity Error**	1 = A Parity Error is inserted within all ARINC Words within the block. 0 = No error

Control Word Definition

1. The **Suppress Parity** is useful when most of the Data Buffers contain standard data with parity and a few buffers contain data type without parity. In such a case, the parity can be set ON for all Words within the Channel x Configuration Register while exceptional cases, can be forced to NO PARITY, using this bit.

Data Hi **Byte Offset: 0016 (H)**

The high 16 bits of the transmit data.

Data Lo **Byte Offset: 0018 (H)**

The low 16 bits of the transmit data.

6.6 Receive Channel Operation

The user sets up each receive channel's mode of operation by writing to various Control Registers (global and channel specific). Each receive channel has three basic modes of operation:

Sequential Mode:	Stores data sequentially per channel, utilizing different receive buffer areas for each channel
Merge Mode:	Stores all receive channel data sequentially into one receive buffer area
Look-Up Table Mode:	Allows the user to store Words in specific locations of memory according to the label

In all these modes, the Data Words are stored with a 16-bit Receive Status Word and a Time Tag value. In Data Only option the Status and Time Tag are not stored. In Merge Mode, the channel ID information (indicates on which channel the data was received) is contained within the Receive Status Word.

The format of the Time Tag depends on the value of the IRIG Time Tag bit in the Receive Data Storage Mode register. See **Receive Data Storage Mode Register** on page 6-12. The two Time Tag formats are described in **Time Tag Word Format** on page 6-32.

In Receive Channel Operation the module is initialized in a wait loop, looking for a START command from the computer. This command is issued by writing to the Global Start Register. (See the **Global Start/Stop Register** on page 6-16). The command instructs the module to begin operation on the selected channels.

Sequential and Merge modes are described in detail in the following sections. Look-Up Table Mode is described in **6.6.2 Look-up Table Mode Operation** on page 6-36.

Note: In addition to the above modes, there is also a Translation Mode, which uses one channel to receive and one channel to transmit. For more information, see **6.7 Translation (Dual Channel) Operation** on page 6-38.

6.6.1 Sequential and Merge Modes

Sequential Mode

The Sequential Mode has a software-selectable feature that filters the storage of the specific, user-defined labels or stores all labels within one buffer. (See Figure 6-11 **Receive Sequential Mode Buffer Structure with Standard Time Tags** on page 6-31.) The data buffer's size and location within the memory is programmed via a Start and End pointer. (See **Channel x Receive Data Start Pointer** and **Channel x Receive Data End Pointer**, on page 6-55). Each received ARINC Data Word is tagged with a status Word, indicating the status of the receive Word and a 32-bit Time Tag value.

These five, 16-bit Words make up a single receive data block. Alternatively, the Sequential Mode offers the user the capability of storing only the ARINC data without the Time Tag and Status Words. This global selection affects all receive channels. (See **Receive Data Storage Mode Register** on page 6-12.) A 16-bit register indicates the number of invalid Words received. Interrupts and pollable status registers allow for numerous event recognition and are described in the Channel

Control Register section of this manual. (See **Channel x Configuration Register** on page 6-56).

To use Sequential Mode, set Bit 01 of the Receive Data Storage Mode Register to 0 and set Bit 05 of the Channel x Configuration Register to 1.

Merge Mode

The Merge Mode operates in the same manner as the Sequential Mode except that all the receive channels are merged into one Data Buffer area. The Control registers for the Merge Mode are located on the **ARINC 429 Module Global Registers Map** on page 6-6 and defined in section **6.4 ARINC 429 Module Global Register Definitions**.

In this mode the Receive Data Blocks are stored in sequential order and each receive Status Word is tagged with a channel ID, indicating the channel on which the data was received. Each data block contains a Time Tag word as in the standard Sequential Mode of operation.

To use Merge Mode, set Bit 01 of the Receive Data Storage Mode Register to 1. When this bit is set to 1, Bit 05 of the Channel x Configuration Register is ignored.

Receive Buffer Storage Sequence

Figures 6-11 and 6-12 illustrate how Receive Data Blocks are stored within the dual-port RAM, while in the Sequential Mode of operation.

The Channel *x* Receive Error Count Register (see page 6-53) is updated with every invalid Word which is stored. The Start and End pointers set up the buffer size. The Receive Data Storage will stop when the end pointer is reached or will wrap around to the beginning of the buffer, depending upon the condition of the Receive Wrap Around bit in the Channel *x* Configuration Register (see page 6-56). The format of the Time Tag depends on the value of the IRIG Time Tag bit in the Receive Data Storage Mode register. (See **Receive Data Storage Mode Register** on page 6-12.)

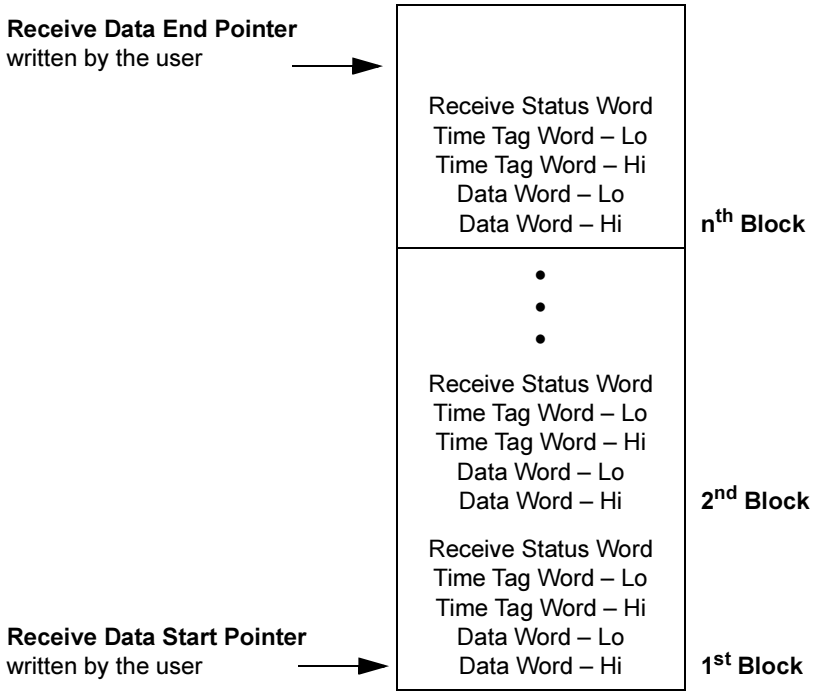


Figure 6-11 Receive Sequential Mode Buffer Structure with Standard Time Tags

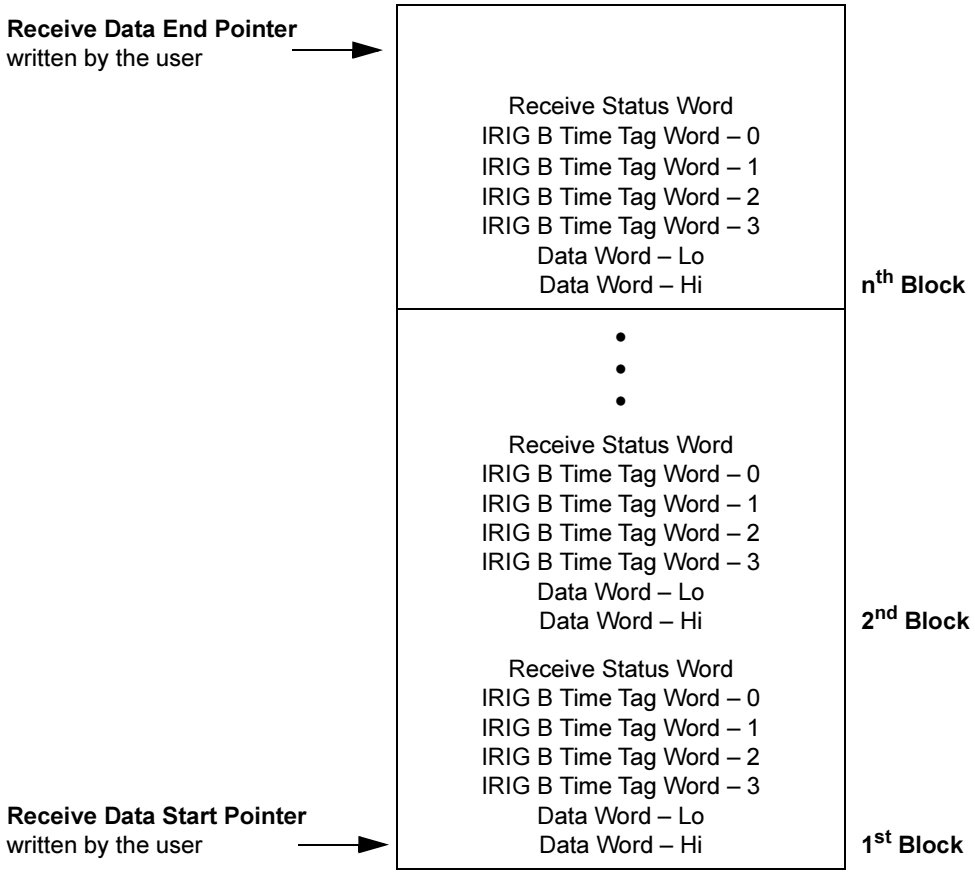


Figure 6-12 Receive Sequential Mode Buffer Structure with IRIG B Time Tags

Receive Data Word Format

The received ARINC Word is stored as two 16-bit Words within the memory (Hi-word followed by a Lo-word).

The numbers shown within the two Words represent the ARINC bit numbers.

32-Bit ARINC Word Receive Format

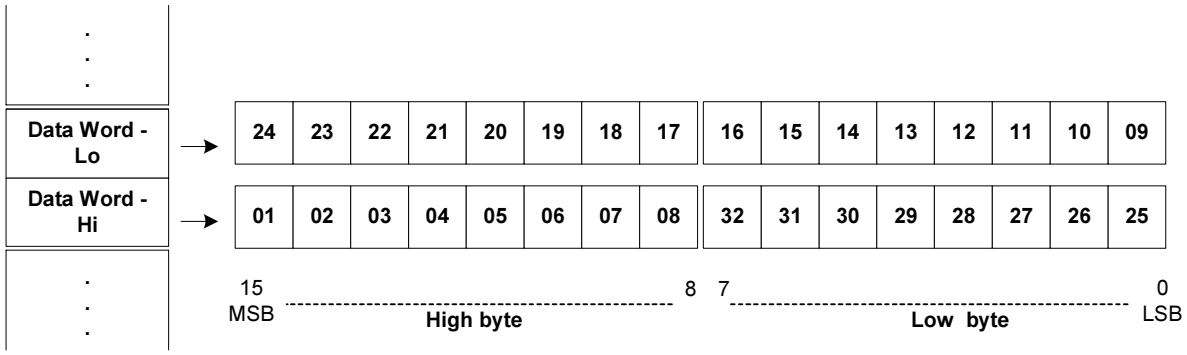


Figure 6-13 32-Bit Receive Data Word Format

Note: 1. The ARINC Word bits are received in the following order:

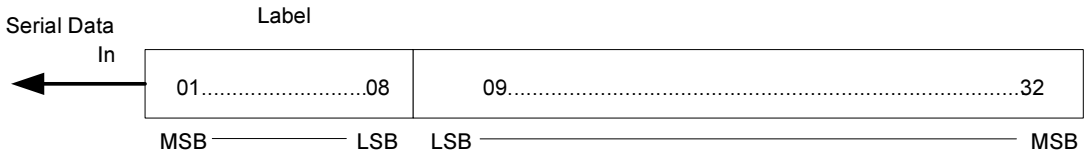


Figure 6-14 32-Bit ARINC Word Order Bit

2. Bits 09 through 32 are ordered from LSB to MSB (as opposed to the Label field which is organized from MSB to LSB). For this reason, in the Data Block the Hi-word is first, followed by the Lo-word, with the Label and the ARINC field 32 through 25 in the Hi-word, and bits 24 through 09, in the Lo-word.

Time Tag Word Format

The format of the Time Tag depends on the value of the IRIG Time Tag bit in the Receive Data Storage Mode register. (See **Receive Data Storage Mode Register** on page 6-12.)

Standard (Non-IRIG B) Time Tag Format

When using the standard (non-IRIG B) Time Tag, the Time Tag is a 32-bit Word made up of two 16-bit Words: Time Tag-Hi and Time Tag-Lo. The Time Tag resolution is 10 microseconds.

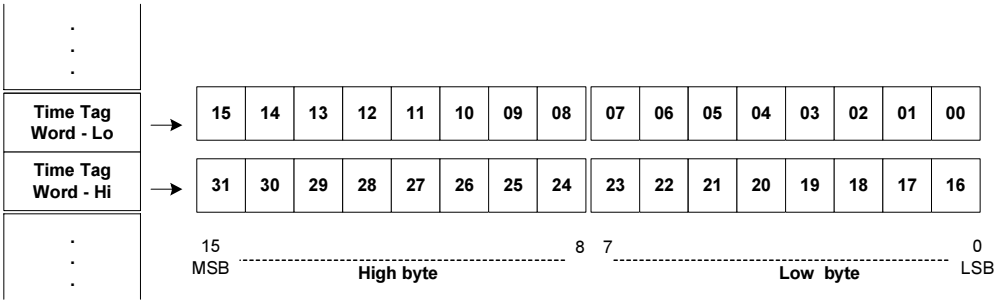


Figure 6-15 Time Tag Word Format in Standard (Non-IRIG B) Mode

IRIG B Time Tag Format

When using the IRIG B Time Tag, the Time Tag is made up of four 16-bit Words.

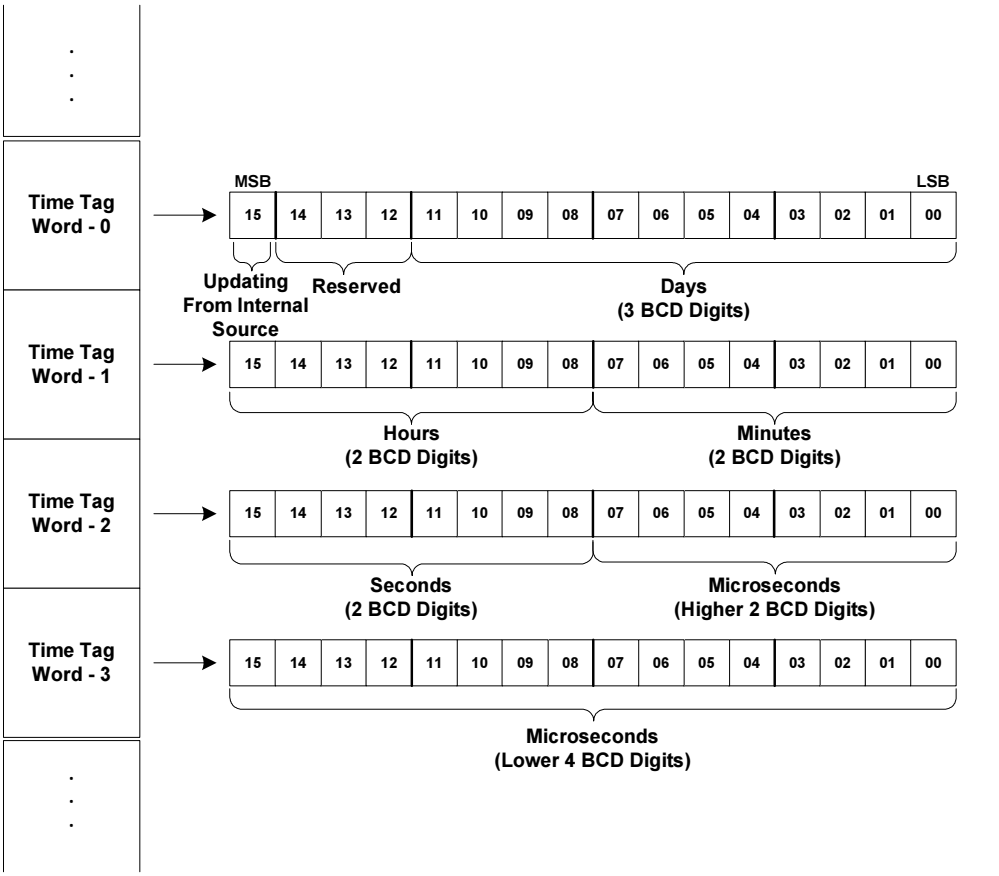


Figure 6-16 Time Tag Word Format in IRIG B Mode

The Updating from Internal Source bit indicates whether a valid IRIG B signal is being received by the module. When this bit is set to 1, the module is not receiving an IRIG B signal, and the module is updating the Time Tag based on its internal clock.

Receive Sequential Mode Status Word

Bit	Bit Name	Description
12-15	**Reserved**	
08-11	**Merge Channel ID Code** ***(Merge Mode only)***	0000 – Data received over Channel 0 0001 – Data received over Channel 1 0010 – Data received over Channel 2 0011 – Data received over Channel 3 0100 – Data received over Channel 4 0101 – Data received over Channel 5 0110 – Data received over Channel 6 0111 – Data received over Channel 7 1000 – Data received over Channel 8 1001 – Data received over Channel 9 **Note**: Channel ID Code: bit 8 = LSB Channel ID Code: bit 11 = MSB
07	**Valid Word**	1 = The Received ARINC word was valid in all respects (Global bit). 0 = Not a valid Word
06	**Reserved**	
05	**Gap [Sync] Time Error**	1 = Gap [Sync] Time Error occurred between Words (less than a 4 bit times between Words. 0 = No error
04	**Invalid Coding Error**	1 = Bit level coding error was detected in the ARINC Word 0 = No error
03	**Parity Error**	1 = A parity error was detected in the ARINC Word. 0 = No error
02	**Lo Bit Count / Invalid Word Error**	1 = A Lo Bit Count or a Null bit Error was detected in the ARINC Word. 0 = No error
01	**Hi Bit Count Error**	1 = A Hi Bit Count or a Null bit Error was detected in the ARINC Word. 0 = No error
00	**Word Received**	1 = Data is in memory. This bit is cleared while data is in the process of being updated. 0 = No received Word

Receive Sequential Mode Status Word

Receive Sequential Mode Filter Table Diagram

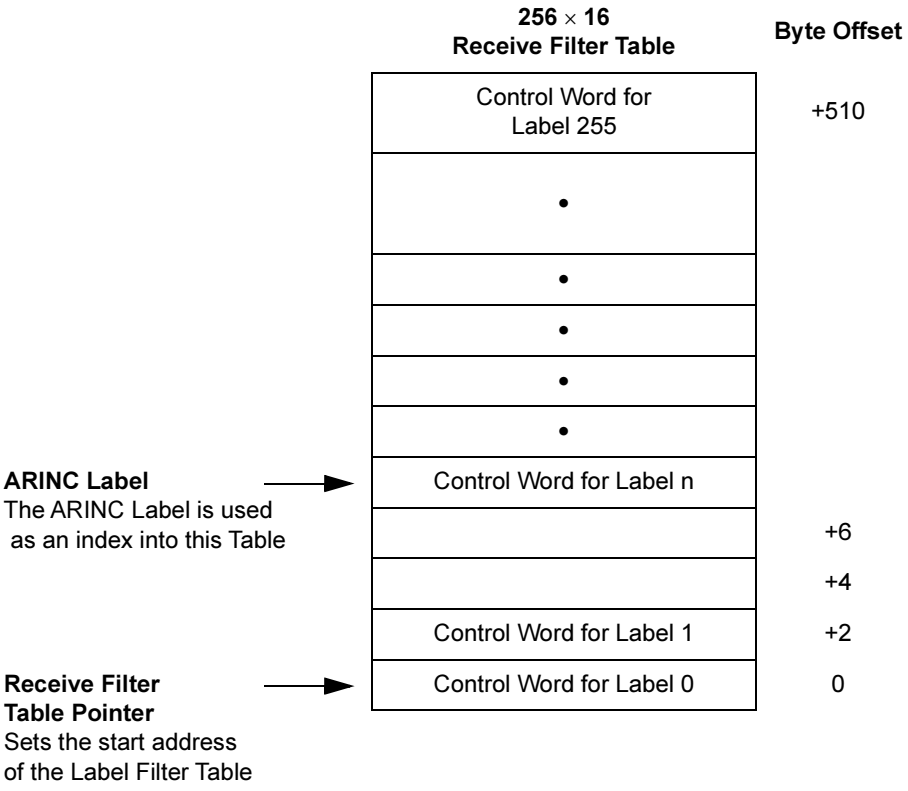


Figure 6-17 Receive Sequential Mode Filter Table Diagram

Bit	Description
02-15	Reserved
01	1 = Interrupt 0 = Don't Interrupt
00	1 = Store Word 0 = Don't Store

Label Control Word Structure (Write)

6.6.2 Look-up Table Mode Operation

In Look-up Table Mode, the Word's label is used by the module as an offset to a 256-word Look-up Table. The Table is programmed by the user with address pointers as to where to write the Receive Data Block. Each block contains a:

- 32-bit ARINC Word,
- 32-bit Time Tag
- Error Count Word
- Status Word

The 256-word Table can be placed anywhere within the memory via a user-programmable Receive Look-up Table Start Address Register.

The user has the ability to monitor the operational status of each channel and to be interrupted on various events. In addition, there exist pollable registers that can be used with or instead of interrupt processing.

To use Look-up Table Mode, set Bit 05 of the Channel x Configuration Register to 0 and set Bit 01 of the Receive Data Storage Mode Register to 0.

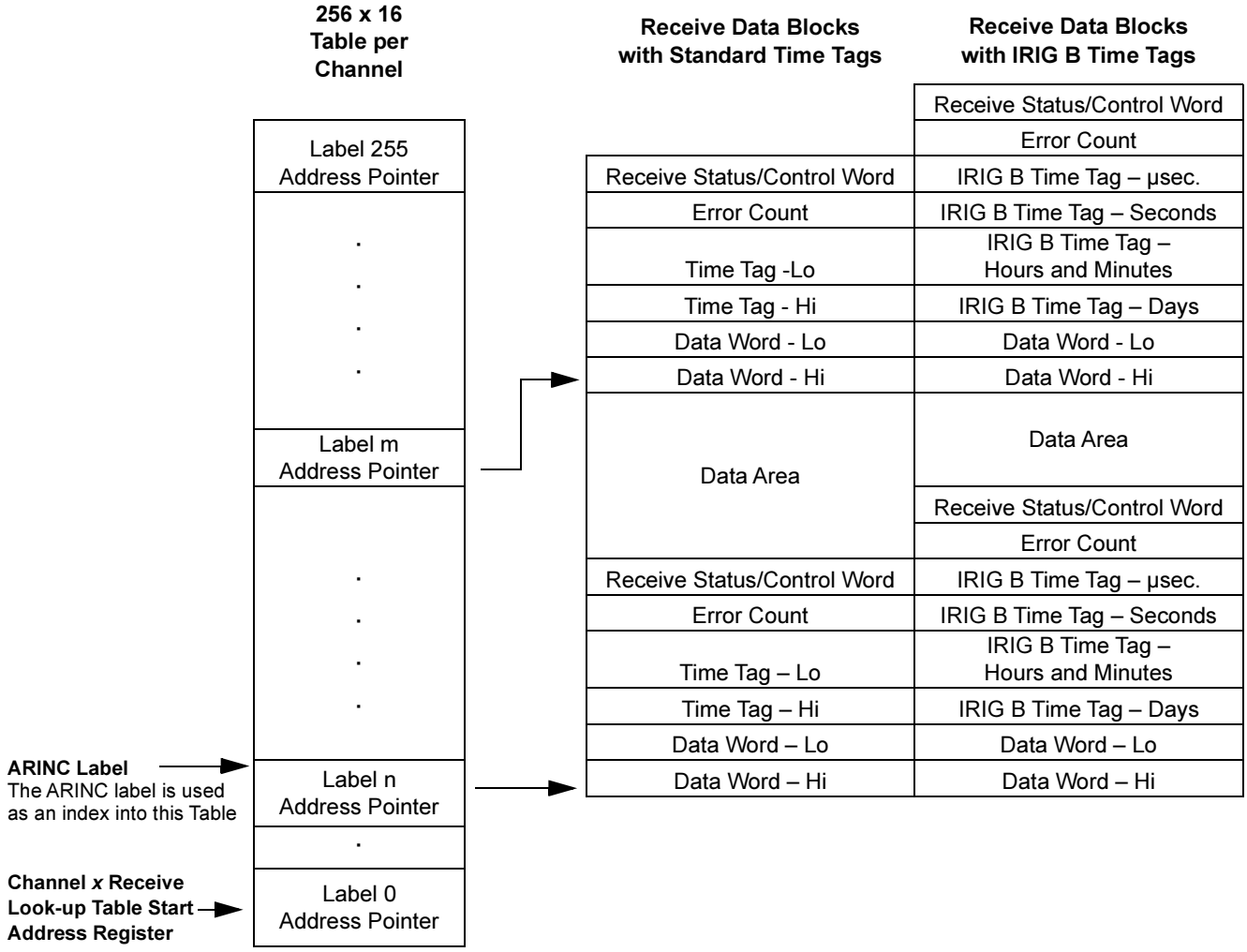


Figure 6-18 Receive Look-up Table Structure

Receive Data Block Description

The Receive Data Block components are:

Data Word — The ARINC Data Word is stored as two 16-bit Words with the same format as Sequential Mode. See **Receive Data Word Format** on page 6-32.

Time Tag Word — The standard (non-IRIG B) Time Tag Word is stored in two 16-bit Words with the same format as Sequential Mode.

The IRIG Time Tag is stored in four 16-bit Words with the same format as Sequential Mode.

See **Time Tag Word Format** on page 6-32.

Error Count Word — The Error Count Word indicates the number of error Words received on a particular label.

Receive Status/ Control Word — See **Table 6-1 Look-up Table: Received Status/Control Word.**

	Bit	Bit Name	Description
Control Byte	**15**	Enable Label Interrupt	Enables the 'interrupt on label received' capability. This bit is used in conjunctions with the Channel *x* Interrupt/ Trigger Condition Register
	08–14	Reserved	Set to 0
Status Byte	**07**	Valid Word	1 = The Received ARINC Word was valid in all respects (Global bit) 0 = Not a valid Word
	06	Reserved	Set to 0
	05	Gap[Sync] Time Error	1 = Gap[Sync] Time Error occurred between Words (less than 4 bit times between Words.) 0 = No error
	04	Invalid Coding Error	1 = Bit level coding error was detected in the ARINC Word 0 = No error
	03	Parity Error	1 = A parity error was detected in the ARINC Word 0 = No error
	02	Lo Bit Count/ In Valid Word Error	1 = A Lo Bit Count or a Null bit error was detected in the ARINC Word 0 = No error
	01	Hi Bit Count Error	1 = A Lo Bit Count or a Null bit error was detected in the ARINC Word 0 = No error
	00	Word Received	1 = Data is in memory. This bit is cleared while data is in the process of being updated 0 = No Word received

Table 6-1 Look-up Table: Received Status/Control Word

6.7 Translation (Dual Channel) Operation

Translation Mode is selected by setting the Translation Mode bit in the Channel *x* Configuration Register of two contiguous channels. The first of the two channels must be a receive channel, the second must be a transmit channel. In this mode, labels received by the receive channel are retransmitted over the transmit channel after undergoing a translation function. A different translation function may be assigned for each label to be received.

To use Translation Mode, set Bit 11 of the Channel x Configuration Register in the receive channel to 1 and set Bit 12 to 0.

In Data Translation operation the user:

- Sets up the receive and transmit channel's parameters by writing to the various Channel *x* Control Registers. In Translation Mode, only bits 00 – 04 of the transmit channel should be set.
- Allocates a translation table for the receive channel.
- Assigns translation functions.
- Writes the data into the FIFOs.
- Starts operation by writing to the Start/Stop Register found within the ARINC 429 Module Global Control Register area. Only the bits associated with the receive channel must be set.

6.7.1 Allocating a Translation Table in a Channel

Set the Filter Table Start Address to point to an array of 256 16-bit words which will be used as pointers to the translation entries. This array is called the array of translation entries.

6.7.2 Allocating a Translation Table Entry

Set an entry within the array of translation entries to point to the Translation Table entry to be allocated. Space must be allocated for a 12 16-bit words per entry. The index into the array of translation entries corresponds to the label of the word to be translated.

6.7.3 Using Translation Functions

To use translation functions, specify which translation functions you want to use in the highest two words of the Translation Table entry, by setting specific bits to 1, and specify the data to be used for the translation functions in the remaining words of the Translation Table entry. You can use up to five translation functions per Translation Table entry. See Figure 6-19 and the following sections.

The structure of a Translation Table entry is as follows:

	Byte Offset
Translation Functions	0000 H
Data for Function 1	0004 H
Data for Function 2	0008 H
Data for Function 3	000C H
Data for Function 4	0010 H
Data for Function 5	0014 H

Figure 6-19 Translation Table Entry

Data for Function 5 **Byte Offset: 0000 H**

The data to be used for the fifth function. The fifth function is the function represented by the highest bit that is set to 1 within the highest two words of the Translation Table entry (byte offsets 0014 H – 0017 H). See **Translation Functions** on page 6-40.

Data for Function 4 **Byte Offset: 0004 H**

The data to be used for the fourth function selected.

Data for Function 3 **Byte Offset: 0008 H**

The data to be used for the third function selected.

Data for Function 2 **Byte Offset: 000C H**

The data to be used for the second function selected.

Data for Function 1 **Byte Offset: 0010 H**

The data in this location will be used for the first function selected. For example, if Bit 0 of byte offset 0014 H is set to 1, this data will be AND with the 24 bits of data of the incoming label. If Bit 0 of byte offset 0014 H is set to 0 and Bit 1 is set to 1, this data will be ORed with the 24 bits of data of the incoming label, and so fourth.

Translation Functions **Byte Offset: 0014 H**

The bits at byte offset 0014 H specify which translation functions to apply. You can use up to five functions for each Translation Table entry. For example, you can AND the value in the incoming label with a specified value, then OR the result with another value.

Bit	Bit Name	Description
09	**SKIP Function**	1 = Do not retransmit the incoming message 0 = Retransmit the incoming message
08	**REPLACE LABEL Function**	1 = Replace the incoming 8-bit label value with the supplied value; the data will be retransmitted as is 0 = Do not use this function
07	**REPLACE ALL Function**	1 = Replace the incoming label and data with the supplied value; all 32 bits will be replaced 0 = Do not use this function
06	**SUBTRACT Function**	1 = Subtract the supplied value from the incoming message data; this will not affect the label but may alter the SDI bits 0 = Do not use this function
05	**ADD Function**	1 = Add the supplied value to the incoming message data; this will not affect the label but may alter the SSM bits 0 = Do not use this function
04	**NOR Function**	1 = NOR the supplied value with the incoming message data; 1 NOR 1 = 0, 1 NOR 0 = 0, 0 NOR 1 = 0, 0 NOR 0 = 1 0 = Do not use this function
03	**XOR Function**	1 = XOR the supplied value with the incoming message data; 1 XOR 1 = 0, 1 XOR 0 = 1, 0 XOR 1 = 1, 0 XOR 0 = 0 0 = Do not use this function
02	**NAND Function**	1 = NAND the supplied value with the incoming message data; 1 NAND 1 = 0, 1 NAND 0 = 1, 0 NAND 1 = 1, 0 NAND 0 = 1 0 = Do not use this function
01	**OR Function**	1 = OR the supplied value with the incoming message data; 1 OR 1 = 1, 1 OR 0 = 1, 0 OR 1 = 1, 0 OR 0 = 0 0 = Do not use this function
00	**AND Function**	1 = AND the supplied value with the incoming message data; 1 AND 1 = 1, 1 AND 0 = 0, 0 AND 1 = 0, 0 AND 0 = 0 0 = Do not use this function

Receive Sequential Mode Status Word

6.8 Channel Control Registers Maps

6.8.1 Channel 0 Control Register Block Memory Map

The registers are marked read only (R) or write only (W) — only the Receive Error Count Register is both read and write.

Reserved	0FD20-0FD22 H
Channel 0 Scratch Buffer End (W)	0FD24 H
Channel 0 Scratch Buffer Start (W)	0FD26 H
Channel 0 Status Register (R)	0FD28 H
Channel 0 Interrupt / Trigger Conditions (W)	0FD2A H
Reserved	0FD2C H
Channel 0 Transmit Loop Counter (W)	0FD2E H
Channel 0 Transmit Instruction Counter (W)	0FD30 H
Channel 0 Transmit Instruction Stack Pointer (W)	0FD32 H
Channel 0 Receive Label Trigger (W)	0FD34 H
Channel 0 Receive Error Count (R/W)	0FD36 H
Channel 0 Receive Interval Counter Trigger (W)	0FD38 H
Reserved	0FD3A H
Channel 0 Receive Data Word Counter Trigger (W)	0FD3C H
Channel 0 Receive Buffer Wraparound (W)	0FD3E H
Channel 0 Receive/Transmit Data Word Count (R)	0FD40 H
Channel 0 Receive Filter Table Start Address (W)	0FD42 H
Channel 0 Receive Look-up Table Start Address (W)	0FD44 H
Channel 0 Receive Data Current Pointer (R)	0FD46 H
Channel 0 Receive Data End Pointer (W)	0FD48 H
Channel 0 Receive Data Start Pointer (W)	0FD4A H
Reserved	0FD4C H
Channel 0 Configuration (W)	0FD4E H

Figure 6-20 Channel 0 Control Register Block Map

6.8.2 Channel 1 Control Register Block Memory Map

The registers are marked read only (R) or write only (W) — only the Receive Error Count Register is both read and write.

Reserved	0FD50-0FD52 H
Channel 1 Scratch Buffer End (W)	0FD54 H
Channel 1 Scratch Buffer Start (W)	0FD56 H
Channel 1 Status Register (R)	0FD58 H
Channel 1 Interrupt / Trigger Conditions (W)	0FD5A H
Reserved	0FD5C H
Channel 1 Transmit Loop Counter (W)	0FD5E H
Channel 1 Transmit Instruction Counter (W)	0FD60 H
Channel 1 Transmit Instruction Stack Pointer (W)	0FD62 H
Channel 1 Receive Label Trigger (W)	0FD64 H
Channel 1 Receive Error Count (R/W)	0FD66 H
Channel 1 Receive Interval Counter Trigger (W)	0FD68 H
Reserved	0FD6A H
Channel 1 Receive Data Word Counter Trigger (W)	0FD6C H
Channel 1 Receive Buffer Wraparound (W)	0FD6E H
Channel 1 Receive/Transmit Data Word Count (R)	0FD70 H
Channel 1 Receive Filter Table Start Address (W)	0FD72 H
Channel 1 Receive Look-up Table Start Address (W)	0FD74 H
Channel 1 Receive Data Current Pointer (R)	0FD76 H
Channel 1 Receive Data End Pointer (W)	0FD78 H
Channel 1 Receive Data Start Pointer (W)	0FD7A H
Reserved	0FD7C H
Channel 1 Configuration (W)	0FD7E H

Figure 6-21 Channel 1 Control Register Block Map

6.8.3 Channel 2 Control Register Block Memory Map

The registers are marked read only (R) or write only (W) — only the Receive Error Count Register is both read and write.

Reserved	0FD80-0FD82 H
Channel 2 Scratch Buffer End (W)	0FD84 H
Channel 2 Scratch Buffer Start (W)	0FD86 H
Channel 2 Status Register (R)	0FD88 H
Channel 2 Interrupt / Trigger Conditions (W)	0FD8A H
Reserved	0FD8C H
Channel 2 Transmit Loop Counter (W)	0FD8E H
Channel 2 Transmit Instruction Counter (W)	0FD90 H
Channel 2 Transmit Instruction Stack Pointer (W)	0FD92 H
Channel 2 Receive Label Trigger (W)	0FD94 H
Channel 2 Receive Error Count (R/W)	0FD96 H
Channel 2 Receive Interval Counter Trigger (W)	0FD98 H
Reserved	0FD9A H
Channel 2 Receive Data Word Counter Trigger (W)	0FD9C H
Channel 2 Receive Buffer Wraparound (W)	0FD9E H
Channel 2 Receive/Transmit Data Word Count (R)	0FDA0 H
Channel 2 Receive Filter Table Start Address (W)	0FDA2 H
Channel 2 Receive Look-up Table Start Address (W)	0FDA4 H
Channel 2 Receive Data Current Pointer (R)	0FDA6 H
Channel 2 Receive Data End Pointer (W)	0FDA8 H
Channel 2 Receive Data Start Pointer (W)	0FDAA H
Reserved	0FDAC H
Channel 2 Configuration (W)	0FDAE H

Figure 6-22 Channel 2 Control Register Block Map

6.8.4 Channel 3 Control Register Block Memory Map

The registers are marked read only (R) or write only (W) — only the Receive Error Count Register is both read and write.

Reserved	0FDB0-0FDB2 H
Channel 3 Scratch Buffer End (W)	0FDB4 H
Channel 3 Scratch Buffer Start (W)	0FDB6 H
Channel 3 Status Register (R)	0FDB8 H
Channel 3 Interrupt / Trigger Conditions (W)	0FDBA H
Reserved	0FDBC H
Channel 3 Transmit Loop Counter (W)	0FDBE H
Channel 3 Transmit Instruction Counter (W)	0FDC0 H
Channel 3 Transmit Instruction Stack Pointer (W)	0FDC2 H
Channel 3 Receive Label Trigger (W)	0FDC4 H
Channel 3 Receive Error Count (R/W)	0FDC6 H
Channel 3 Receive Interval Counter Trigger (W)	0FDC8 H
Reserved	0FDCA H
Channel 3 Receive Data Word Counter Trigger (W)	0FDCC H
Channel 3 Receive Buffer Wraparound (W)	0FDCE H
Channel 3 Receive/Transmit Data Word Count (R)	0FDD0 H
Channel 3 Receive Filter Table Start Address (W)	0FDD2 H
Channel 3 Receive Look-up Table Start Address (W)	0FDD4 H
Channel 3 Receive Data Current Pointer (R)	0FDD6 H
Channel 3 Receive Data End Pointer (W)	0FDD8 H
Channel 3 Receive Data Start Pointer (W)	0FDDA H
Reserved	0FDDC H
Channel 3 Configuration (W)	0FDDE H

Figure 6-23 Channel 3 Control Register Block Map

6.8.5 Channel 4 Control Register Block Memory Map

The registers are marked read only (R) or write only (W) — only the Receive Error Count Register is both read and write.

Reserved	0FDE0-0FDE2 H
Channel 4 Scratch Buffer End (W)	0FDE4 H
Channel 4 Scratch Buffer Start (W)	0FDE6 H
Channel 4 Status Register (R)	0FDE8 H
Channel 4 Interrupt / Trigger Conditions (W)	0FDEA H
Reserved	0FDEC H
Channel 4 Transmit Loop Counter (W)	0FDEE H
Channel 4 Transmit Instruction Counter (W)	0FDF0 H
Channel 4 Transmit Instruction Stack Pointer (W)	0FDF2 H
Channel 4 Receive Label Trigger (W)	0FDF4 H
Channel 4 Receive Error Count (R/W)	0FDF6 H
Channel 4 Receive Interval Counter Trigger (W)	0FDF8 H
Reserved	0FDFA H
Channel 4 Receive Data Word Counter Trigger (W)	0FDFC H
Channel 4 Receive Buffer Wraparound (W)	0FDFE H
Channel 4 Receive/Transmit Data Word Count (R)	0FE00 H
Channel 4 Receive Filter Table Start Address (W)	0FE02 H
Channel 4 Receive Look-up Table Start Address (W)	0FE04 H
Channel 4 Receive Data Current Pointer (R)	0FE06 H
Channel 4 Receive Data End Pointer (W)	0FE08 H
Channel 4 Receive Data Start Pointer (W)	0FE0A H
Reserved	0FE0C H
Channel 4 Configuration (W)	0FE0E H

Figure 6-24 Channel 4 Control Register Block Map

6.8.6 Channel 5 Control Register Block Memory Map

The registers are marked read only (R) or write only (W) — only the Receive Error Count Register is both read and write.

Reserved	0FE10-0FE12 H
Channel 5 Scratch Buffer End (W)	0FE14 H
Channel 5 Scratch Buffer Start (W)	0FE16 H
Channel 5 Status Register (R)	0FE18 H
Channel 5 Interrupt / Trigger Conditions (W)	0FE1A H
Reserved	0FE1C H
Channel 5 Transmit Loop Counter (W)	0FE1E H
Channel 5 Transmit Instruction Counter (W)	0FE20 H
Channel 5 Transmit Instruction Stack Pointer (W)	0FE22 H
Channel 5 Receive Label Trigger (W)	0FE24 H
Channel 5 Receive Error Count (R/W)	0FE26 H
Channel 5 Receive Interval Counter Trigger (W)	0FE28 H
Reserved	0FE2A H
Channel 5 Receive Data Word Counter Trigger (W)	0FE2C H
Channel 5 Receive Buffer Wraparound (W)	0FE2E H
Channel 5 Receive/Transmit Data Word Count (R)	0FE30 H
Channel 5 Receive Filter Table Start Address (W)	0FE32 H
Channel 5 Receive Look-up Table Start Address (W)	0FE34 H
Channel 5 Receive Data Current Pointer (R)	0FE36 H
Channel 5 Receive Data End Pointer (W)	0FE38 H
Channel 5 Receive Data Start Pointer (W)	0FE3A H
Reserved	0FE3C H
Channel 5 Configuration (W)	0FE3E H

Figure 6-25 Channel 5 Control Register Block Map

6.8.7 Channel 6 Control Register Block Memory Map

The registers are marked read only (R) or write only (W) — only the Receive Error Count Register is both read and write.

Reserved	0FE40-0FE42 H
Channel 6 Scratch Buffer End (W)	0FE44 H
Channel 6 Scratch Buffer Start (W)	0FE46 H
Channel 6 Status Register (R)	0FE48 H
Channel 6 Interrupt / Trigger Conditions (W)	0FE4A H
Reserved	0FE4C H
Channel 6 Transmit Loop Counter (W)	0FE4E H
Channel 6 Transmit Instruction Counter (W)	0FE50 H
Channel 6 Transmit Instruction Stack Pointer (W)	0FE52 H
Channel 6 Receive Label Trigger (W)	0FE54 H
Channel 6 Receive Error Count (R/W)	0FE56 H
Channel 6 Receive Interval Counter Trigger (W)	0FE58 H
Reserved	0FE5A H
Channel 6 Receive Data Word Counter Trigger (W)	0FE5C H
Channel 6 Receive Buffer Wraparound (W)	0FE5E H
Channel 6 Receive/Transmit Data Word Count (R)	0FE60 H
Channel 6 Receive Filter Table Start Address (W)	0FE62 H
Channel 6 Receive Look-up Table Start Address (W)	0FE64 H
Channel 6 Receive Data Current Pointer (R)	0FE66 H
Channel 6 Receive Data End Pointer (W)	0FE68 H
Channel 6 Receive Data Start Pointer (W)	0FE6A H
Reserved	0FE6C H
Channel 6 Configuration (W)	0FE6E H

Figure 6-26 Channel 6 Control Register Block Map

6.8.8 Channel 7 Control Register Block Memory Map

The registers are marked read only (R) or write only (W) — only the Receive Error Count Register is both read and write.

Reserved	0FE70-0FE72 H
Channel 7 Scratch Buffer End (W)	0FE74 H
Channel 7 Scratch Buffer Start (W)	0FE76 H
Channel 7 Status Register (R)	0FE78 H
Channel 7 Interrupt / Trigger Conditions (W)	0FE7A H
Reserved	0FE7C H
Channel 7 Transmit Loop Counter (W)	0FE7E H
Channel 7 Transmit Instruction Counter (W)	0FE80 H
Channel 7 Transmit Instruction Stack Pointer (W)	0FE82 H
Channel 7 Receive Label Trigger (W)	0FE84 H
Channel 7 Receive Error Count (R/W)	0FE86 H
Channel 7 Receive Interval Counter Trigger (W)	0FE88 H
Reserved	0FE8A H
Channel 7 Receive Data Word Counter Trigger (W)	0FE8C H
Channel 7 Receive Buffer Wraparound (W)	0FE8E H
Channel 7 Receive/Transmit Data Word Count (R)	0FE90 H
Channel 7 Receive Filter Table Start Address (W)	0FE92 H
Channel 7 Receive Look-up Table Start Address (W)	0FE94 H
Channel 7 Receive Data Current Pointer (R)	0FE96 H
Channel 7 Receive Data End Pointer (W)	0FE98 H
Channel 7 Receive Data Start Pointer (W)	0FE9A H
Reserved	0FE9C H
Channel 7 Configuration (W)	0FE9E H

Figure 6-27 Channel 7 Control Register Block Map

6.8.9 Channel 8 Control Register Block Memory Map

The registers are marked read only (R) or write only (W) — only the Receive Error Count Register is both read and write.

Reserved	0FEA0-0FEA2 H
Channel 8 Scratch Buffer End (W)	0FEA4 H
Channel 8 Scratch Buffer Start (W)	0FEA6 H
Channel 8 Status Register (R)	0FEA8 H
Channel 8 Interrupt / Trigger Conditions (W)	0FEAA H
Reserved	0FEAC H
Channel 8 Transmit Loop Counter (W)	0FEAE H
Channel 8 Transmit Instruction Counter (W)	0FEE0 H
Channel 8 Transmit Instruction Stack Pointer (W)	0FEB2 H
Channel 8 Receive Label Trigger (W)	0FEB4 H
Channel 8 Receive Error Count (R/W)	0FEB6 H
Channel 8 Receive Interval Counter Trigger (W)	0FEB8 H
Reserved	0FEBA H
Channel 8 Receive Data Word Counter Trigger (W)	0FEBC H
Channel 8 Receive Buffer Wraparound (W)	0FEBE H
Channel 8 Receive/Transmit Data Word Count (R)	0FEC0 H
Channel 8 Receive Filter Table Start Address (W)	0FEC2 H
Channel 8 Receive Look-up Table Start Address (W)	0FEC4 H
Channel 8 Receive Data Current Pointer (R)	0FEC6 H
Channel 8 Receive Data End Pointer (W)	0FEC8 H
Channel 8 Receive Data Start Pointer (W)	0FECA H
Reserved	0FECC H
Channel 8 Configuration (W)	0FECE H

Figure 6-28 Channel 8 Control Register Block Map

6.8.10 Channel 9 Control Register Block Memory Map

The registers are marked read only (R) or write only (W) — only the Receive Error Count Register is both read and write.

Reserved	0FED0-0FED2 H
Channel 9 Scratch Buffer End (W)	0FED4 H
Channel 9 Scratch Buffer Start (W)	0FED6 H
Channel 9 Status Register (R)	0FED8 H
Channel 9 Interrupt / Trigger Conditions (W)	0FEDA H
Reserved	0FEDC H
Channel 9 Transmit Loop Counter (W)	0FEDE H
Channel 9 Transmit Instruction Counter (W)	0FEE0 H
Channel 9 Transmit Instruction Stack Pointer (W)	0FEE2 H
Channel 9 Receive Label Trigger (W)	0FEE4 H
Channel 9 Receive Error Count (R/W)	0FEE6 H
Channel 9 Receive Interval Counter Trigger (W)	0FEE8 H
Reserved	0FEEA H
Channel 9 Receive Data Word Counter Trigger (W)	0FEEC H
Channel 9 Receive Buffer Wraparound (W)	0FEEE H
Channel 9 Receive/Transmit Data Word Count (R)	0FEF0 H
Channel 9 Receive Filter Table Start Address (W)	0FEF2 H
Channel 9 Receive Look-up Table Start Address (W)	0FEF4 H
Channel 9 Receive Data Current Pointer (R)	0FEF6 H
Channel 9 Receive Data End Pointer (W)	0FEF8 H
Channel 9 Receive Data Start Pointer (W)	0FEFA H
Reserved	0FEFC H
Channel 9 Configuration (W)	0FEFE H

Figure 6-29 Channel 9 Control Register Block Map

6.9 Channel Control Register Definitions

6.9.1 Channel *x* Scratch Buffer End Register — Write

Data Rate Mode

For transmission in Data Rate Mode, the Channel *x* Scratch Buffer End Register contains the end address of the scratch buffer that must be assigned by the user. See **Channel x Scratch Buffer Start Register** on page 6-51.

6.9.2 Channel *x* Scratch Buffer Start Register — Write

Data Rate Mode

For transmission in Data Rate Mode, the user must allocate a scratch buffer for use by the firmware. Its length must be at least (Transmit Instruction Counter × 10 + 4) bytes. If the buffer is not long enough, the channel will turn itself off without transmitting.

The Channel *x* Scratch Buffer Start Register contains the start address of this buffer. The address must be a word boundary.

6.9.3 Channel *x* Status Register — Read

The Channel *x* Status Register indicates the operational status of the channel. This register can be used to poll the status of the channel or it can be used with interrupts. When used in conjunction with interrupts, the register indicates the condition or conditions, which caused the interrupt.

A logic 1 indicates an active bit. The user must reset status bits, by writing a 0 to this register.

Bit	Bit Name – Interrupt Cause
07-15	Reserved
06	Receive - Stopped on Buffer Full
05	Receive - Error Word Received
04	Receive - Data Word Count Trigger
03	Receive - Interval Count Trigger
02	Receive - Label Received
01	Transmit - End of Frame
00	Transmit - End of Block

Channel *x* Status Register

Look-up Mode

In Look-Up Mode, the Label Received status bit is set upon receipt of any label for which an interrupt has been requested via the label's Control byte.

Sequential Mode

In Sequential Mode, the Label Received status bit is set upon receipt of any label for which an interrupt has been requested via the Filter Table.

6.9.4 Channel *x* Interrupt/Trigger Condition Register — Write

The Channel *x* Interrupt/Trigger Condition Register sets the Interrupt and Trigger condition or conditions of the channel. The Trigger Conditions set a pulse on the External Trigger Output.

Trigger Condition Bits 08 – 15		Interrupt Condition Bits 00 – 07	
Bit	**Bit Name**	**Bit**	**Bit Name**
15	Reserved	**07**	Reserved
14	Receive – Stopped on buffer full	**06**	Receive – Stopped on buffer full
13	Receive – Error Word received	**05**	Receive – Error Word received
12	Receive – Data Word Count Trigger [1]	**04**	Receive – Data Word Count Trigger [1]
11	Receive – Interval Count Trigger [2]	**03**	Receive – Interval Count Trigger [2]
10	Receive – Label Received [3]	**02**	Receive – Label Received [3]
09	Transmit – End of Frame	**01**	Transmit – End of Frame
08	Transmit – End of Block	**00**	Transmit – End of Block

Receive Merge Interval/Trigger Condition Register

1. To activate the Receive – Data Word Count Trigger interrupt or trigger, the Channel x Receive Data Word Counter Trigger register must also be set.
2. To activate the Receive – Interval Count Trigger interrupt or trigger, the Channel x Receive Interval Counter Trigger register must also be set.
3. The Receive – Label Received interrupt or trigger only occurs upon reception of a label, which has been marked for interrupt in a Filter Table (in Sequential Mode) or in a Control Byte (in Look-up Table Mode.

6.9.5 Channel x Transmit Loop Counter — Write

The Channel *x* Transmit Loop counter sets the number of times to execute the transmit instruction blocks (the frame): *N* Times or Continuous Loop. If the continuous value is selected, setting the related channel bit in the Global Start/ Stop Register to a 0 can terminate the channel's operation.

Bit	Value	
00–15	0000	= Continuous
	0001	= One time
	0002	= Two times
		•
		•
		•
	FFFF	= 65535 times

Channel *x* Transmit Loop Counter

Note: When using Data Rate Mode or FIFO Mode, the value of this register is ignored and the data is transmitted continuously. See **Channel x**

Configuration Register on page 6-56. For an explanation of all transmit modes, see **6.5 Transmit Channel Operation** on page 6-20.

6.9.6 Channel *x* Transmit Instruction Counter — Write

The Channel *x* Transmit Instruction Counter sets the number of Transmit Instruction blocks to process. These instruction blocks taken together comprise a frame.

6.9.7 Channel *x* Transmit Instruction Stack Pointer — Write

The Channel *x* Transmit Instruction Stack Pointer sets the starting address of the Transmit Instruction Stack**.** The address must be a word boundary within the Transmit Instruction Stack area. For example, to place the Transmit Instruction stack starting at location 0300 (H), write 0300 (H) to this register.

6.9.8 Channel *x* Receive Label Trigger Register — Write

Sequential Mode

The Channel *x* Receive Label Trigger Register is used in conjunction with the Receive Label Trigger bit in the Channel *x* Configuration Register (see **Channel x Configuration Register** on page 6-56). This register enables the reception and storage of data upon receipt of a unique ARINC label. The module will not store any ARINC words received prior to the first instance of this label.

Bit	Description
08–15	Set to 0
00–07	Trigger Label

Channel *x* Receive Label Trigger Register

6.9.9 Channel x Receive Error Count Register — Read/Write

Sequential Mode

The 16-bit Channel *x* Receive Error Count Register indicates the number of errors received on a particular channel. To reset the register, write 0000 to it.

6.9.10 Channel x Receive Interval Counter Trigger Register — Write

Sequential Mode

The Channel *x* Receive Interval Counter Trigger Register allows the user to generate an interrupt and set a flag upon reception of every *N* number of words, where *N* is the value written to this register. For example, to request an interrupt after every five ARINC words, write 0005 (H) to this register.

To generate an interrupt or a trigger, the appropriate bit must also be set in the Channel *x* Interrupt / Trigger Condition Register (see **Channel x Interrupt/Trigger Condition Register** on page 6-52).

6.9.11 Channel x Receive Data Word Counter Trigger Registers — Write

The Channel *x* Receive Data Word Counter Trigger Register lets the user generate an interrupt and set a flag, which indicates when a specific number of words have been received (1-65535). To generate an interrupt, the appropriate bit must also be set in the Channel *x* Interrupt/Trigger Condition Register (see **Channel x Interrupt/Trigger Condition Register** on page 6-52).

Note: This trigger is set when the value in the Receive Data Word Counter matches the value set in this register.

6.9.12 Channel *x* Receive Buffer Wraparound Register — Write

The Channel *x* Receive Buffer Wraparound Register contains 2 bits for synchronization with the host.

Bit	Description
15	1 = Multiple Wraparound - Data Lost
14	1 = Single Wraparound since last data read
00-13	0

Channel *x* Receive Buffer Wraparound Register

Note: Excalibur software drivers handle these bits. If these drivers, are used, they do not need to be modified.

The user should clear bit 14 each time the first word of the buffer is read. When the buffer wraps around bit 14 is checked. If bit 14 is '0', the module sets the bit to indicate a Single Wraparound ('1'). If bit 14 is '1', the module sets bit 15 to indicate a Multiple Wraparound ('1') has occurred.

6.9.13 Channel *x* Receive/Transmit Data Word Count Register — Read

Sequential Mode

The Channel *x* Receive/Transmit Data Word Count Register indicates the number of ARINC words received (0-65535). This register wraps around to 0 after it reaches 65535. The user may reset the register to 0, only when the channel is stopped.

6.9.14 Channel *x* Receive Filter Table Start Address — Write

Sequential Mode

The Channel *x* Receive Filter Table Start Address sets the Start Address of the 256 × 16 Label Filter Table as described in section **6.6.1 Sequential and Merge Modes** on page 6-29. The address must be a word boundary. It is valid for several channels to use the same Filter Table.

This table is valid only if the Channel *x* Configuration Register Enable Receive Filter Table bit is set. (See bit 09 in the **Channel x Configuration Register** on page 6-56.)

6.9.15 Channel *x* Receive Look-up Table Start Address — Write

Look-up Table Mode

The Channel *x* Receive Look-Up Table Start Address sets the start address of the 256 × 16 Receive Look-Up Table (see bit 05 in the **Channel x Configuration Register** on page 6-56). The address must be a word boundary.

This address points to the *first* location of the Look-Up Table. The module stores one ARINC data block for each label received. The data block contains: 32-bit ARINC word, 32-bit Time Tag, and the 16-bit Receive Status Word. The subsequent reception and storage of another ARINC word will overwrite the Data Block with the same ARINC label.

6.9.16 Channel *x* Receive Data Current Pointer — Read

Sequential Mode

In Sequential Mode the Channel *x* Receive Data Current Pointer indicates the address where the next ARINC receive word is to be placed in the buffer. This pointer value is incremented after the entire receive block (ARINC word, Time Tag, and Status word) is written into memory.

Look-up Table Mode

In Look-Up Table Mode this register contains the address of the last ARINC receive word written to the receive area.

6.9.17 Channel *x* Receive Data End Pointer — Write

Sequential Mode

The Channel *x* Receive Data End Pointer sets the end address of the Receive Data buffer. The address must be a word boundary. The data will wrap around or stop when the buffer is full, (when the End Address is reached), depending upon the contents of the Receive Wrap Around control bit in the Channel *x* Configuration Register.

6.9.18 Channel *x* Receive Data Start Pointer — Write

Sequential Mode

The Channel *x* Receive Data Start Pointer Register sets the start address of the Receive Data buffer. The address must be a word boundary within the Receive Data Blocks area.

Example: To cause the Channel *x* Receive Data Buffer to begin at address 001A0 (H), write 01A0 (H) to this register.

6.9.19 Channel *x* Configuration Register — Write

The Channel *x* Configuration Register sets up various channel parameters. Bits that are not used (for example, receive-related bits while operating as a transmitter) are ignored by the module.

Bit	Bit Name	Description
13-15	Reserved	Set to 0
12	Transmit Mode Select 0	This bit works in conjunction with Bit 08. See the table in Bit 08. 1 = FIFO Mode selected as the transmit mode. (See **6.5.2 FIFO Mode** on page 6-25.) 0 = Interblock Time Mode selected as the transmit mode. (See **6.5.1 Interblock Time Mode and Data Rate Mode** on page 6-20.) **Note:** When using Data Rate Mode or FIFO Mode, the value of the Channel *x* Transmit Loop Counter register is ignored and the data is transmitted continuously. See **Channel x Transmit Loop Counter on page 6-52**.
11	Translation Mode	1 = Translation Mode selected. Translation Mode works in pairs of channels. Labels received by the first channel are retransmitted over the next channel after a translation function is performed. See **6.7 Translation (Dual Channel) Operation** on page 6-38. 0 = Translation Mode not selected.
10	Extended Time Mode	1 = Extended Time Mode selected 0 = Standard Time Mode selected When Extended Time Mode is selected, the Interblock Time/Data Rate is doubled on the bus. See **Interblock Time/Data Rate Word** on page 6-23. This bit is only relevant if Extended Time Mode is supported by the firmware. See **Board Status Register** on page 6-15.
09	Enable Receive Filter Table	1 = Enable Filter Table (Stores labels per table) 0 = Disables table. Stores all labels.
08	Transmit Mode Select 1	This bit works in conjunction with Bit 12. See the table below. 1 = Data Rate Mode (per data block) – Data blocks sent on a scheduled data rate basis. For example: Data Block 1 every 50 msec. Data Block 2 every 25 msec. In this mode a scratch buffer must be allocated via the Channel *x* Scratch Buffer Start/End registers. For more information, see **6.5.1 Interblock Time Mode and Data Rate Mode** on page 6-20. 0 = Interblock Time Mode or FIFO Mode (depending on the value of Bit 12) **Note:** When using Data Rate Mode or FIFO Mode, the value of the Channel *x* Transmit Loop Counter register is ignored and the data is transmitted continuously. See **Channel x Transmit Loop Counter** on page 6-52.
07	Receive Label Trigger	1 = Start data storage upon receipt of label xx. See **Channel x Receive Label Trigger Register** on page 6-53. 0 = Receive stores data without Start Label Trigger

Bit 08	Bit 12	Transmit Mode
0	0	Interblock Time Mode selected
0	1	FIFO Mode selected
1	0	Data Rate Mode selected
1	1	

Channel *x* Configuration Register

Bit	Bit Name	Description
06	**Receive Wrap Around**	1 = Data storage is halted when the buffer is full. 0 = Receive wraps around the data within the block. (This bit is used in Sequential Storage Mode only.)
05	**Receive Storage Mode**	1 = Sequential Storage Mode 0 = Look-up Table Mode (see **Channel x Receive Look-up Table Start Address** on page 6-55). This bit is ignored if Bit 01 of the Receive Data Storage Mode Register is set to 1 (Merge Mode).
04	**Parity: Even/Odd**	1 = Even 0 = Odd - Standard ARINC Mode
03	**Parity: On/Off**	1 = Off 0 = On
02	**Transmit Rise/ Fall Time**	1 = Lo Speed (10 +/- 5μsec.) 0 = Hi Speed (1.5 +/- 0.5μsec.)
00-01	**Bit Rate**	If the Programmable Bit Rate is selected then the bit rate is defined by the global **Programmable Bit Rate Register** (see page 6-13).

Bit 01	Bit 00	Channel Bit rate
0	0	12.5 KHz – Lo Speed
0	1	100 KHz – Hi Speed
1	0	Programmable
1	1	Undefined

Channel *x* Configuration Register (Continued)

The Channel *x* Configuration Register can only be written to when *all* the channels are turned off (via the Start/Stop Register).

The module should be started (via the Start/Stop Register) only after a minimum of 1msec. from the time that the contents of the register have been modified.

All active Channel *x* Configuration Registers should be set up immediately following the Board ID validation, before programming other parameters.

7 Discrete Module Control Registers (Advanced)

Chapter 7 describes how to run the Discrete I/Os via the memory registers, which enables you to develop your own software tools. When using Excalibur's software tools, this chapter is not required.

The following topics are covered:

7.1 Discretes Module Memory Map

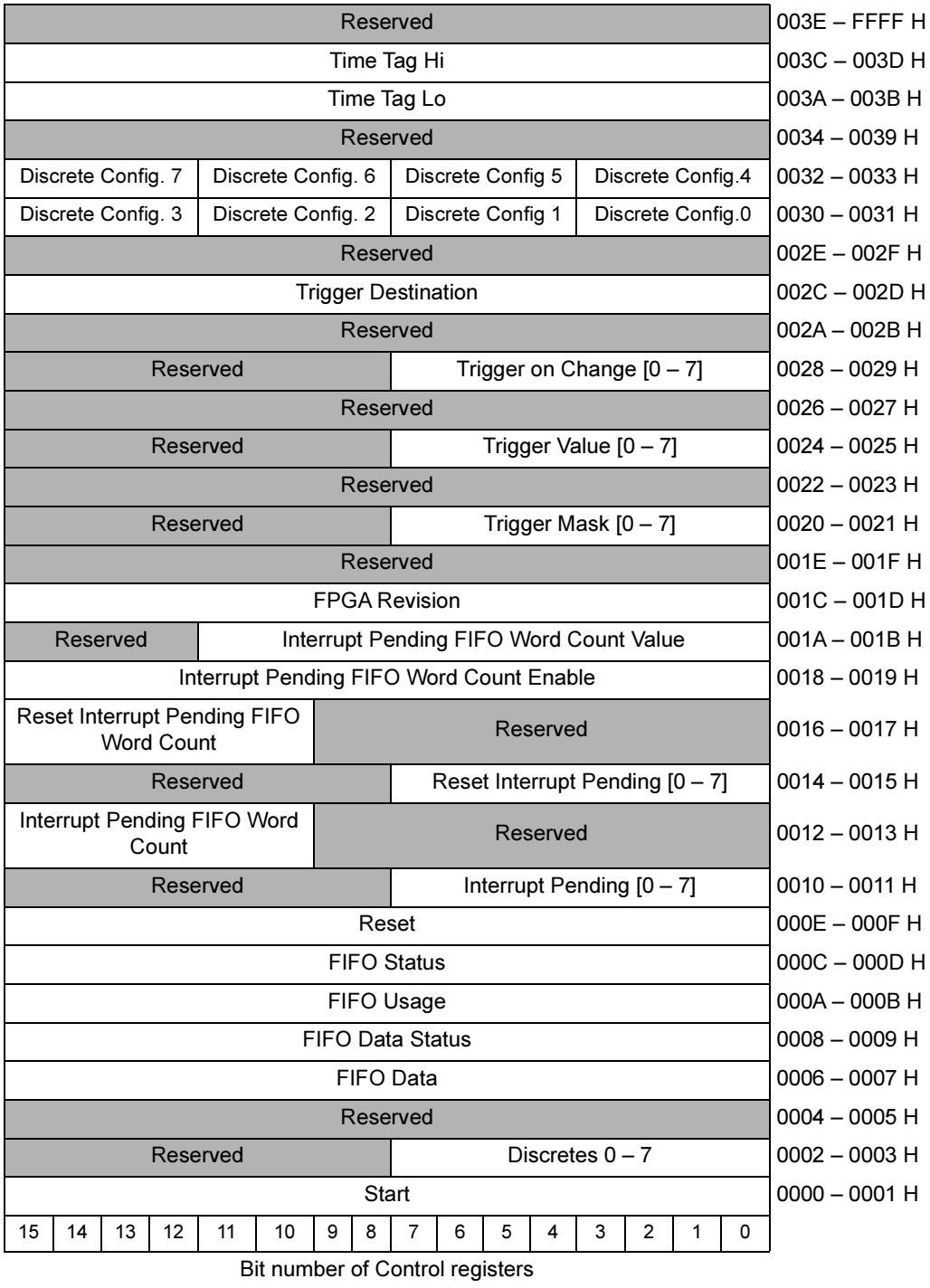


Figure 7-1 Discretes Control Registers Map

7.2 Discrete Control Register Definitions

7.2.1 Start Register

Address: 00000 – 00001 (H)

Read/Write The Start Register controls the start/stop operation of the module.

Bit	Description
01-15	Reserved
00	Start/Stop operation 0 = Stop module - **Default** All bits in the Discrete Input Register remain constant and are not effected by changes in the voltage levels on the external Discrete lines. The bits in the Discrete Output registers can not be altered and the value on the output external Discrete lines will be dependent on the last value stored in the Output register before the module was stopped. Configuration of the Discretes is allowed 1 = Start module All bits in the Input and Output registers can be updated and/or changed. Configuration of the Discretes is *not* allowed

Start Register

7.2.2 Discrete Registers

Address: 00002 – 00003 (H) Discretes 0 – 7

When the module is stopped, each Discrete can be programmed by the user either as an Input Discrete or as an Output Discrete. See **Discretes Configuration Registers,** on page 7-12. The bits in the Discrete register control and represent the values associated with these Discretes.

Read only **Discretes Configured as Input [Default]**

When a Discrete has been configured as an input, the associated bit of this register represents the value stored in the input Discrete. Either an Avionics or TTL Discrete logic-level will be stored, depending on the voltage threshold configured for the Discrete. (See **Discretes Configuration Registers,** on page 7-12)

When the module is *stopped,* all bits in this register remain constant and are not affected by changes in the logic-level voltages on the Input Discretes. When the module is *started,* all the bits in this register will be updated/changed according to the current logic-level voltages on the input Discretes. (See **Start Register,** page 7-3.)

Note When an input Discrete has been configured with External Debounce *enabled* (see **Discretes Configuration Registers,** on page 7-12) a change in the input Discrete will only be stored in the associated bit of this register if the input Discrete is stable for a minimum of 60ms.

Bit	Signal	Description
08-15	Reserved	
07	IN7	Logic-level voltage on input Discrete 7
06	IN6	Logic-level voltage on input Discrete 6
05	IN5	Logic-level voltage on input Discrete 5
04	IN4	Logic-level voltage on input Discrete 4
03	IN3	Logic-level voltage on input Discrete 3
02	IN2	Logic-level voltage on input Discrete 2
01	IN1	Logic-level voltage on input Discrete 1
00	IN0	Logic-level voltage on input Discrete 0

Read only: Input Discrete Register - Discretes 0 – 9: Address: 00002 – 00003 (H)

Note All Input Discretes are sampled on either power-on, software reset or a start of the module. Thus for any of these actions, the voltage level on the external Discrete lines will be stored to the bits of this register.

Write only **Discretes Configured as Output**

When an I/O Discrete has been configured as an output, the associated bit of this register controls the value set on the output Discrete. The threshold voltage level of the output Discrete will depend on what voltage the line is pulled up to (since the output Discretes are open collector). See **Discrete Channel Information,** on page 1-6.

When the module is *stopped,* all bits in this register cannot be changed and the output Discretes are kept at the voltage level that was set before the board was stopped. When the module is *started,* all the bits in this register can be changed and will update the voltage levels set to the associated Discrete output accordingly. (See **Start Register,** page 7-3)

All bits of the register have a default value of 1, which will tri-state the external Discrete lines.

Bit	Signal	Description
08-15	Reserved	
07	OUT7	Logic-level voltage on output Discrete 7
06	OUT6	Logic-level voltage on output Discrete 6
05	OUT5	Logic-level voltage on output Discrete 5
04	OUT4	Logic-level voltage on output Discrete 4
03	OUT3	Logic-level voltage on output Discrete 3
02	OUT2	Logic-level voltage on output Discrete 2
01	OUT1	Logic-level voltage on output Discrete 1
00	OUT0	Logic-level voltage on output Discrete 0

Read/Write: Output Discrete Register - Discretes 0 – 9: Address: 00002 – 00003 (H)

7.2.3 FIFO Data Register Address: 0006 – 0007(H)

Read only

The *DAS-429UNET/RTx* contains a FIFO which stores changes that occur on the input Discretes. The Fifo data register displays the value of the oldest data stored in the FIFO. When any of the Discretes are triggered on a Rising Edge, Falling Edge or Change, the new value stored in the Discrete registers will be stored in the FIFO along with an associated Time tag.

See **Discretes 0 – 7 Trigger Mask Register,** page 7-9,
Discretes 0 – 7 Trigger Value Register, page 7-10 and
Discretes 0 – 7 Trigger on Change Register, page 7-11.

After every trigger, four 16-bit words will be stored in the FIFO. The words are stored in the following order:

1. Discrete Inputs 0 – 7
2. 16 zeros (reserved)
3. The Low word of the TTAG associated with the stored Discretes
4. The High word of the TTAG associated with the stored Discretes

Note The unused upper bits of the Discrete Input words mentioned above (1& 2) are set to zero.

When any word is stored in the FIFO, the value in the FIFO Usage register will be incremented by one. When any word is read from the FIFO, the value in the FIFO Usage register will be decremented by one and that word will be deleted from the memory of the FIFO.

The Time tag that is stored in the FIFO with the Discrete data has a size of 32-bits.

Time Tag is reset to 0 upon a power up, a channel reset, or a global software reset, and starts counting. (See **Reset Register**, on page 7-7 and **Software Reset Register**, on page 5-3.) When it reaches the value FFFF FFFF (H), the counter wraps around to 0 and continues counting. To re-initialize to 0 without doing a software reset, write to the Time Tag Reset register.

The Time Tag can also be reset via the External Time Tag Reset signal from the External Signals Connector, allowing synchronization to other systems. See the **External Signals Connector Pinouts**, on page 4-9.

The Time Tag resolution is 4 μs. So the Time Tag will wrap around after approximately 4.8 hours. The 4 μs resolution is based on the *UNET*'s 1 μs global clock for Time Tags. You can change the Discrete channel's Time Tag resolution by using an external Time Tag clock source via the EXTTCLKI pin of the External Signals Connector. (See the **External Signals Connector Pinouts**, on page 4-9.) When using an external clock, the Discrete channel will multiply the clock resolution by 4. For example, if the clock is taken from an external source that has a 10 μs resolution, the Discrete channel Time Tag will have a 40 μs resolution.

The reset value of all the bits in the FIFO is 0, i.e., all data in the FIFO is cleared on a reset.

7.2.4 FIFO Data Status Register

Address: 0008 – 0009 (H)

Read only

The FIFO Data Status Register displays the *type* of data that is displayed in the FIFO Data Register.

Bit	Description	Value	
02-15	Reserved - Set to 0		
00-01	FIFO Data Register type	00	DIscrete inputs 0 – 9 [**Default**]
		01	Discrete inputs 10 – 19
		10	Time tag - Low word
		11	Time tag - High word

FIFO Data Status register

7.2.5 FIFO Usage Register

Address: 000A – 000B (H)

Read only

The FIFO Usage Register defines how much of the FIFO's memory has been used.

Note

1. The FIFO has a maximum capacity of 4092, 16 bit words.
2. For each trigger, 4 words will be automatically stored in the FIFO

Bit	Description
12-15	Reserved - Set to 0
00-11	Defines the number of words stored in the FIFO **Default: 0000**

FIFO Usage register

7.2.6 FIFO Status Register

Address: 000C – 000D (H)

Read only

The FIFO Status Register defines the status of the FIFO memory

Bit	Description
04-15	Reserved - Set to 0
03	0 FIFO memory is not full **Default** 1 FIFO memory is full [4092 words]
02	0 FIFO has sufficient memory left - there are less than 4072 words stored in FIFO **Default** 1 FIFO memory is nearly full: there are 4072 or more words stored in FIFO
01	0 FIFO memory is not empty 1 FIFO memory is empty **Default**
00	0 FIFO memory is nowhere near empty: there are 20 or more words stored in FIFO 1 FIFO memory is nearly empty - there are less than 20 words stored in FIFO **Default**

FIFO Status register

7.2.7 Reset Register Address: 000E – 000F (H)

Write only Writing any value to the Reset register will cause a software reset to the module. A reset will cause all registers to restore to the default values all clear all data from the FIFO.

7.2.8 Discrete Inputs [0 – 7] Interrupt Pending Register Address: 0010 – 0011 (H)

Use the Discrete Inputs [0 – 7] Interrupt Pending Register to identify which Discrete or Discretes (0 – 7) produced a trigger and may have requested an interrupt/external trigger.

The default value for all these bits is 0.

Bit	Description
08-15	Reserved - set to 0
07	0 = No trigger generated by Discrete 7 1 = Trigger generated by Discrete 7
06	0 = No trigger generated by Discrete 6 1 = Trigger generated by Discrete 6
05	0 = No trigger generated by Discrete 5 1 = Trigger generated by Discrete 5
04	0 = No trigger generated by Discrete 4 1 = Trigger generated by Discrete 4
03	0 = No trigger generated by Discrete 3 1 = Trigger generated by Discrete 3
02	0 = No trigger generated by Discrete 2 1 = Trigger generated by Discrete 2
01	0 = No trigger generated by Discrete 1 1 = Trigger generated by Discrete 1
00	0 = No trigger generated by Discrete 0 1 = Trigger generated by Discrete 0

Discrete Inputs [0 – 7] Interrupt Pending Register

7.2.9 Reset Discrete Inputs [0 – 7] Interrupt Pending Register Address: 0014 – 0015 (H)

Write only Writing to the Reset Discrete Inputs [0 – 7] Interrupt Pending Register will reset the corresponding bits of the Discrete Inputs [0 – 7] Interrupt Pending Register.

Bit	Description
10-15	Reserved - set to 0
00-09	Reset Discrete Inputs [0 – 7] Interrupt Pending 0 = No effect 1 = Reset the corresponding bit in the Discrete Inputs [0 – 7] Interrupt Pending Register

Reset Discrete Inputs [0 – 7] Interrupt Pending Register

Note The Discrete Inputs [0 – 7] Interrupt Pending Register bits are not cleared automatically, they must be reset with this register.

7.2.10 Interrupt on FIFO Word Count Enable Register Address: 0018 – 0019 (H)

Read/Write When the bit of this register is set, then the Interrupt on FIFO Word Count register will be enabled.

Bit	Description
01-15	Reserved set to 0
00	0 Interrupt on FIFO Word Count will be disabled - **Default** 1 Interrupt on FIFO Word Count will be enabled

Interrupt on FIFO Word Count Enable Register

7.2.11 Interrupt on FIFO Word Count Value Register Address: 001A – 001B (H)

Read/Write When the number of words stored in the FIFO is equal to or greater than user defined value stored in this register, then an interrupt will be generated (if this feature has been enabled see **Interrupt on FIFO Word Count Enable Register,** page 7-8).

Bit	Description
12-15	Reserved set to 0
00-11	Interrupt on FIFO Word Count Value: **Default 0FFF**

7.2.12 FPGA Revision Register Address: 001C – 001D (H)

Read only The FPGA Revision register contains the FPGA revision of the module.

7.2.13 Discretes 0 – 7 Trigger Mask Register

Address: 0020 – 0021 (H)

Read/Write

The Discrete 0 – 7 Trigger Mask Register defines which input Discretes are to be monitored for activity. When any of the masked inputs change, a trigger will be produced depending on the value of the Trigger Value Register. (See **Discretes 0 – 7 Trigger Value Register**, on page 7-10 or **Discretes 0 – 7 Trigger on Change Register**, on page 7-11) The output of the trigger is dependent on the **Trigger Destination Register**, on page 7-11. The Discrete that produced the trigger is indicated in the **Discrete Inputs [0 – 7] Interrupt Pending Register**, page 7-7.

The default value for all the bits of the register is 0.

Bit	Description
08-15	Reserved - set to 0
07	0 = Do not monitor Discrete 7 1 = Monitor Discrete 7
06	0 = Do not monitor Discrete 6 1 = Monitor Discrete 6
05	0 = Do not monitor Discrete 5 1 = Monitor Discrete 5
04	0 = Do not monitor Discrete 4 1 = Monitor channel 4
03	0 = Do not monitor Discrete 3 1 = Monitor Discrete 3
02	0 = Do not monitor Discrete 2 1 = Monitor Discrete 2
01	0 = Do not monitor Discrete 1 1 = Monitor Discrete 1
00	0 = Do not monitor Discrete 0 1 = Monitor Discrete 0

Discrete 0 – 7 Trigger Mask Register

7.2.14 Discretes 0 – 7 Trigger Value Register

Address: 0024 – 0025 (H)

Read/Write The Discrete 0 – 7 Trigger Value Register defines if a trigger will be generated by a rising edge or a falling edge on the specific input Discretes 0 – 7.

The default value for all the bits of the register is 0.

Bit	Description
08-15	Reserved
07	0 = Rising edge on input Discrete 7 1 = Falling edge on input Discrete 7
06	0 = Rising edge on input Discrete 6 1 = Falling edge on input Discrete 6
05	0 = Rising edge on input Discrete 5 1 = Falling edge on input Discrete 5
04	0 = Rising edge on input Discrete 4 1 = Falling edge on input Discrete 4
03	0 = Rising edge on input Discrete 3 1 = Falling edge on input Discrete 3
02	0 = Rising edge on input Discrete 2 1 = Falling edge on input Discrete 2
01	0 = Rising edge on input Discrete 1 1 = Falling edge on input Discrete 1
00	0 = Rising edge on input Discrete 0 1 = Falling edge on input Discrete 0

Discretes 0 – 7 Trigger Value Register

7.2.15 Discretes 0 – 7 Trigger on Change Register

Address: 0028 – 0029 (H)

Read/Write The Discretes 0 – 7 Trigger On Change Register defines if a trigger will be generated by a change (rising edge or falling edge) on the specific input Discretes 0 – 7.

The default value for all the bits of this register is 0.

Bit	Description
08-15	Reserved – Set to 0
07	0 = Do not Trigger on change of input Discrete 7 1 = Trigger on change of input Discrete 7
06	0 = Do not Trigger on change of input Discrete 6 1 = Trigger on change of input Discrete 6
05	0 = Do not Trigger on change of input Discrete 5 1 = Trigger on change of input Discrete 5
04	0 = Do not Trigger on change of input Discrete 4 1 = Trigger on change of input Discrete 4
03	0 = Do not Trigger on change of input Discrete 3 1 = Trigger on change of input Discrete 3
02	0 = Do not Trigger on change of input Discrete 2 1 = Trigger on change of input Discrete 2
01	0 = Do not Trigger on change of input Discrete 1 1 = Trigger on change of input Discrete 1
00	0 = Do not Trigger on change of input Discrete 0 1 = Trigger on change of input Discrete 0

Discretes 0 – 7 Trigger on Change Register

7.2.16 Trigger Destination Register

Address: 002C – 002D (H)

Read/Write Whenever a trigger occurs, the value of the Discrete registers will be added to the Discrete FIFO along with a Time Tag. The trigger may also be used to generate an interrupt or external signal.

Use the Trigger Destination Register to control where the trigger on the input or inputs is directed.

Bit	Description		
02-15	Reserved - set to 0		
00-01	Trigger Output		
	Bit 01	**Bit 00**	
	0	0	Not Connected
	0	1	Routed to interrupt line
	1	0	Reserved
	1	1	Reserved

Trigger Destination Register

For a description of the External Signal and its pinout, see the **External Signals**

Connector Pinouts, page 4-9.

7.2.17 Discretes Configuration Registers

Address: 0030 – 0031 (H) Discretes 0 – 3
0032 – 0033 (H) Discretes 4 – 7

Read/Write Each of these registers controls the configuration of 4 separate Discretes. There are 4 bits associated with each Discrete, each with the functionality outlined below. Note these registers can only be changed while the board has been stopped.

Note

1. For information concerning Discrete set as Inputs or Outputs, see **Discrete Registers,** page 7-3.
2. For information about Input Threshold Designation see **Discrete Channel Information,** page 1-6.
3. If External Debounce has been enabled, then a change on any of the input Discretes will only be stored once the input is stable for a minimum of 60 ms.

7.2.18 Discretes 0 – 3 Configuration Registers

Address: 0030 – 0031 (H)

The default value for all the bits of the register is 0.

Bit	Discrete #	Name	Description
15		Reserved	
14	**3**	Input Debounce Designation	0 = External Debounce disabled 1 = External Debounce enabled
13	**3**	Input Threshold Designation	0 = TTL Voltage Threshold 1 = Avionics Voltage Threshold
12	**3**	I/O Designation	0 = Set as Input 1 = Set as Output
11		Reserved	
10	**2**	Input Debounce Designation	0 = External Debounce disabled 1 = External Debounce enabled
09	**2**	Input Threshold Designation	0 = TTL Voltage Threshold 1 = Avionics Voltage Threshold
08	**2**	I/O Designation	0 = Set as Input 1 = Set as Output
07		Reserved	
06	**1**	Input Debounce Designation	0 = External Debounce disabled 1 = External Debounce enabled
05	**1**	Input Threshold Designation	0 = TTL Voltage Threshold 1 = Avionics Voltage Threshold
04	**1**	I/O Designation	0 = Set as Input 1 = Set as Output
03		Reserved	
02	**0**	Input Debounce Designation	0 = External Debounce disabled 1 = External Debounce enabled
01	**0**	Input Threshold Designation	0 = TTL Voltage Threshold 1 = Avionics Voltage Threshold
00	**0**	I/O Designation	0 = Set as Input 1 = Set as Output

Discretes 0 – 3 Configuration Register

7.2.19 Discretes 4 – 7 Configuration Registers

Address: 0032 – 0033 (H)

The default value for all the bits of the register is 0.

Bit	Discrete #	Name	Description
15		Reserved	
14	**7**	Input Debounce Designation	0 = External Debounce disabled 1 = External Debounce enabled
13	**7**	Input Threshold Designation	0 = TTL Voltage Threshold 1 = Avionics Voltage Threshold
12	**7**	I/O Designation	0 = Set as Input 1 = Set as Output
11		Reserved	
10	**6**	Input Debounce Designation	0 = External Debounce disabled 1 = External Debounce enabled
09	**6**	Input Threshold Designation	0 = TTL Voltage Threshold 1 = Avionics Voltage Threshold
08	**6**	I/O Designation	0 = Set as Input 1 = Set as Output
07		Reserved	
06	**5**	Input Debounce Designation	0 = External Debounce disabled 1 = External Debounce enabled
05	**5**	Input Threshold Designation	0 = TTL Voltage Threshold 1 = Avionics Voltage Threshold
04	**5**	I/O Designation	0 = Set as Input 1 = Set as Output
03		Reserved	
02	**4**	Input Debounce Designation	0 = External Debounce disabled 1 = External Debounce enabled
01	**4**	Input Threshold Designation	0 = TTL Voltage Threshold 1 = Avionics Voltage Threshold
00	**4**	I/O Designation	0 = Set as Input 1 = Set as Output

Discretes 4 – 7 Configuration Register

7.2.20 Time Tag Hi & Lo

Address: **003A – 003B (H)**
003C – 003D (H)

Read only

This Time Tag is a free-running 32-bit counter on the Discrete channel. Time Tag is reset to 0 upon a power up, a channel reset, or a global software reset, and starts counting. (See **Reset Register**, on page 7-7 and **Software Reset Register**, on page 5-3.) When it reaches the value FFFF FFFF (H), the counter wraps around to 0 and continues counting. To re-initialize to 0 without doing a software reset, write to the Time Tag Reset register.

The Time Tag can also be reset via the External Time Tag Reset signal from the External Signals Connector, allowing synchronization to other systems. See the **External Signals Connector Pinouts**, on page 4-9.

The Time Tag resolution is 4 μs. So the Time Tag will wrap around after approximately 4.8 hours. The 4 μs resolution is based on the *UNET*'s 1 μs global clock for Time Tags. You can change the Discrete channel's Time Tag resolution by using an external Time Tag clock source via the EXTTCLKI pin of the External Signals Connector. (See the **External Signals Connector Pinouts**, on page 4-9.) When using an external clock, the Discrete channel will multiply the clock resolution by 4. For example, if the clock is taken from an external source that has a 10 μs resolution, the Discrete channel Time Tag will have a 40 μs resolution.

The user may read the Time Tag counter at any time. Read the two 16-bit words of the Time Tag counter value sequentially, first Lo word, then Hi word.

The counter must be read in the following sequence:

1. Read 003A H – Lo word (16 bit, read only)
2. Read 003C H – Hi word (16 bit, read only)

8 Ordering Information

Chapter 8 describes the part numbers to indicate when ordering a *UNET*. Replace '***x***' in the part numbers with the required number of ARINC 429 channels (5 or 10).

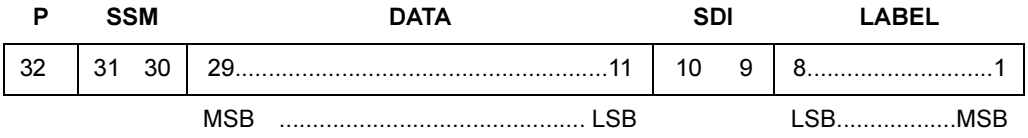
Note: The *DAS-429UNET/RTx* is supplied with two Micro-B to Standard-A USB cables, and a USB power supply.

Part Number	Option	Description
DAS-429UNET/RT*x*-M (Replace '*x*' with the number of channels desired, 5 or 10)		ARINC 429 USB/Ethernet device with up to 10 channels with a micro DB 25-pin connector mounted on the front panel. In addition, there are 8 Discrete channels. For this configuration, Excalibur supplies a 30 cm (11.8 in.) adapter cable with a DB 25 female connector. A 15-pin micro DB15 mating connector for the External Signals Connector is also provided.
DAS-429UNET/RT*x*-C (Replace '*x*' with the number of channels desired, 5 or 10)		ARINC 429 USB/Ethernet device with up to 10 channels with a hard-wired flat cable with a DB 25-pin connector. In addition, there are 8 Discrete channels. A 15-pin micro DB15 mating connector for the External Signals Connector is provided.
	-P	Add this suffix for a mounting plate.
	-B	Add this suffix for an internal rechargeable battery.

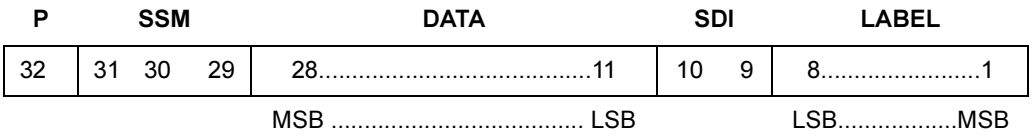
Appendix A ARINC 429 Basic Word Formats

All data sent over the ARINC bus is composed of 32-bit words. A number of different formats can be used, as the following diagrams show:

Format 1



Format 2



Format 3

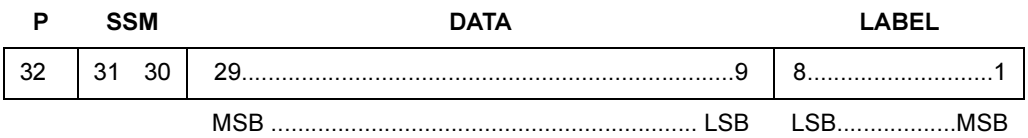


Figure A-1 ARINC 429 Basic Word Formats

Key:

Field	Description
LABEL	Information identifier
SDI	Source/Destination Identifier bits
DATA	Information data: may be presented in one of the following formats
	BNR (Binary)
	BCD (Binary Coded Decimal)
	Discrete
	Maintenance and Acknowledgment
	Alpha Numeric (ISO Alphabet no.5)
SSM	Sign/Status Matrix bits
P	Parity bit (Odd parity)
LSB	Least Significant bit
MSB	Most Significant bit

Bits are transmitted starting with bit 1, the final bit transmitted is the parity bit, bit 32. The label is transmitted with the most significant bit first while the data is transmitted least significant bit first.

The **LABEL** is an octal value from 1 to 377 representing a particular type of data. Most labels are defined in the specification though some are reserved for future needs. Many labels are multiply defined in the specification based on the type of equipment being used.

The **SDI** field is used when a transmitter is connected to multiple receivers but not all data is meant to be used by all the receivers. In this case each receiver will be assigned an SDI value and will look only at labels that match its SDI value. While the specification calls for SDI 00 to be universally accepted, a good deal of equipment appears to disregard this requirement.

The **DATA** field contains the actual data to be sent. A number of data formats are defined in the specification. Binary Coded Decimal (BCD) format uses each four bits to contain a single decimal digit. BNR data is a binary coding. For both data types the specification calls out the units, the resolution, the range, the number of bits used and how frequently the label should be sent. A discrete type has multiple single bit fields defined within a single label. A number of other formats are described in the specification.

SSM, which is sometime 3 bits long, is used for information which helps interpret the numeric value in the data field. Examples of SSM values might be Plus, North, East, Right, To or Above.

P is the parity bit. ARINC 429 calls for odd parity. The parity bit is the last bit sent over the bus.

Appendix B ARINC 429 Connection Precautions

Appendix B describes connection precautions for ARINC 429 cards and modules:

1. Verify the ARINC-429 line is not overloaded beyond the spec:

   Rload > 400 ohm

   Cload < 30,000 pF
2. Use shielded twisted pair wires with typical impedance of 60 to 80 Ohms.
3. Ensure that there is common ground between the connected systems in order to avoid potential differences.
4. Connect/Disconnect cables while the card is powered OFF or *not* transmitting at least.
5. Special care needs to be taken while applying probes of measuring instruments to avoid shorting out signals.

May 2017, Rev A-4