



Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!



Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832

Email & Skype: info@chipsmall.com Web: www.chipsmall.com

Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China



Core429 Development Kit User's Guide

© 2009 Actel Corporation. All rights reserved.

Printed in the United States of America

Part Number: 50200185-0

Release: November 2009

No part of this document may be copied or reproduced in any form or by any means without prior written consent of Actel.

Actel makes no warranties with respect to this documentation and disclaims any implied warranties of merchantability or fitness for a particular purpose. Information in this document is subject to change without notice. Actel assumes no responsibility for any errors that may appear in this document.

This document contains confidential proprietary information that is not to be disclosed to any unauthorized person without prior written consent of Actel Corporation.

Trademarks

Actel, IGLOO, Actel Fusion, ProASIC, Libero, Pigeon Point and the associated logos are trademarks or registered trademarks of Actel Corporation. All other trademarks and service marks are the property of their respective owners.

Table of Contents

Introduction	5
Core429 Development Kit Contents	6
Core429 Development Kit Web Resources	6
Core429 Development Kit Design Files	6
User's Guide Contents	7
Required Items	7
1 Core429 Development Kit Hardware.....	9
Core429 IP Core	9
Core429 Development Kit	9
2 Core429 Demonstration Design.....	13
FPGA Design	13
Demonstration Design	17
A Programming the Flash Memory for Core8051.....	25
B Programming the Fusion FPGA on M1AFS-ADV-DEV-KIT	29
C Communication from HyperTerminal to M1AFS-ADV-DEV-KIT.....	31
D Product Support	33
Customer Service	33
Actel Customer Technical Support Center	33
Actel Technical Support	33
Website	33
Contacting the Customer Technical Support Center	33
Index	35

Introduction

The Core429 Development Kit (Figure 1) demonstration design enables you to evaluate the functionality of Actel's Core429 with a full development kit that includes ARINC 429 example software, Core429 programming files, ARINC 429 physical connections, and full user documentation. You can use the files design available from the "[Core429 Development Kit Web Resources](#)" to program an M1AFS1500-FG484 device to create an ARINC platform with four transmit channels and four receive channels. The targeted FPGA (M1AFS1500) is mounted on the M1AFS-ADV-DEV-KIT. The Core429-SA Daughter Card has ARINC 429 physical connections and is plugged directly onto an IP-DC-SA IP Daughter Card. The IP Daughter Card is plugged onto the M1AFS-ADV-DEV-KIT. With Core429 programmed into the FPGA and the Core429-SA Daughter Card connected to the M1AFS-ADV-DEV-KIT board via the IP Daughter Card, you have control of a complete ARINC 429 evaluation system.

Note: Core429 meets the ARINC 429-16 specification.

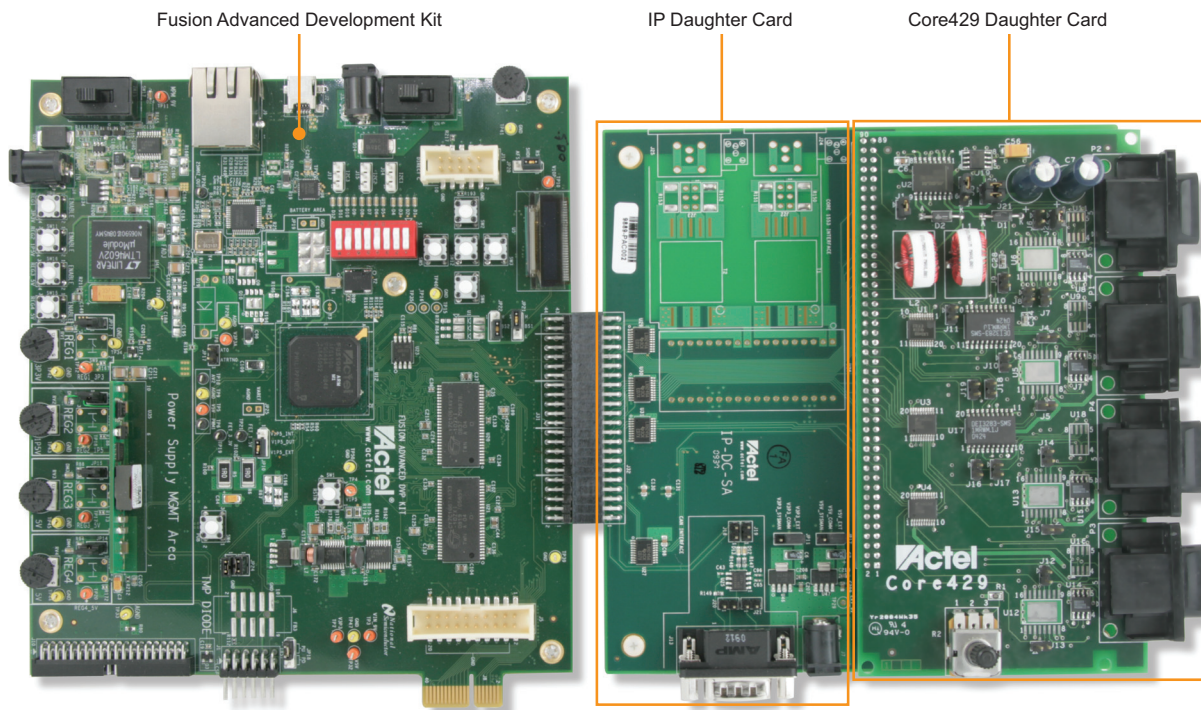


Figure 1 • Core429 Development Kit Version 2

Core429 Development Kit Contents

The Core429 Development Kit includes the items shown in Table 1:

Table 1 • Core429-DEV-KIT-2 Contents

Ordering Code	Description
Core429-SA	Core429-SA Daughter Card
M1AF5-ADV-DEV-KIT-PWR	Fusion Advanced Development Kit with power supplies
IP-DC-SA	IP Daughter Card
CORE429-RS232-CABLE	Core429 loopback cable

You can order the full development kit or if you already have a Fusion Advanced Development Kit, then order only the components needed.

Core429 Development Kit Web Resources

Core429-DEV-KIT Design Files

www.actel.com/download/rsc/?f=Core429_DEV_KIT_2_DF

Schematics

www.actel.com/products/hardware/devkits_boards/Core429_fadk.aspx

Core429 IP Handbook

www.actel.com/ipdocs/Core429_HB.pdf

Fusion Advanced Development Kit Documentation

www.actel.com/products/hardware/devkits_boards/fusion_adv.aspx#rsc

Core429 Development Kit Design Files

Associated files for this demonstration can be downloaded from the Actel website: www.actel.com/download/rsc/?f=CORE429_DEV_KIT_2_DF. These files include an Actel design example and programming files for the example design. The ZIP file contains the following folders:

- Designer_adb: ADB file for the demonstration design
- Driver: Contains the CP210x_Drivers.zip file, which has the driver for the USB-to-UART interface
- Programming_File: PDB programming files
- RTL: FPGA example design source files
- Script: Script file for testing the demo in Script mode
- Software_Hex_file: This folder has two subfolders:
 - Hexfile: Contains the hexadecimal file for Core8051
 - SoftwareProject: Contains the compiled Keil project, including source code

User's Guide Contents

This user's guide describes the contents, architecture, and guidelines for working with the Core429 Development Kit demonstration design and the Core429 Development Kit. This document provides the following:

- Detailed user information and description of the Core429 demonstration design
- Detailed reference material for implementing a new design using the Core429 Development Kit

This user's guide covers these topics:

- Core429 Development Kit overview
- Board description
- Core429 Development Kit demonstration design
- Running the demonstration
- Programming the flash memory for Core8051
- Programming the M1AFS1500 FPGA on the M1AFS-ADV-DEV-KIT
- Communication from HyperTerminal to the M1AFS-ADV-DEV-KIT

Core429 has both internal and external Loopback modes. This demonstration design supports both modes. While in internal Loopback mode, Tx data out of one channel loops back into the receiver on the same channel. To run the Core429 demonstration in internal Loopback mode, only the M1AFS-ADV-DEV-KIT board is needed. The Core429-SA Daughter Card and the IP-DC-SA IP Daughter Card are needed for running the demonstration in external Loopback mode.

Required Items

The following items are required in order to run the Core429 Development Kit demonstration:

- Core429_DEV_KIT_2_DF.zip file
- M1AFS-ADV-DEV-KIT-PWR (includes M1AFS-ADV-DEV-KIT board, two power supplies, and two USB cables)
- HyperTerminal or similar serial communication program
- PC system running Windows® XP operating system
- Core429-SA Daughter Card (not required for internal Loopback mode)
- IP-DC-SA IP Daughter Card (not required for internal Loopback mode)

Optional Items

- Low-cost programming stick (LCPS)
- FlashPro software, v8.5 or later
- Fusion Advanced Development Kit User's Guide
- Core429 IP license
- Libero® Integrated Design Environment (IDE) v8.5 SP2 or later

1 – Core429 Development Kit Hardware

Core429 IP Core

Actel Core429 provides a bus interface for as many as 16 ARINC 429 receivers and transmitters. Core429 is supported in all recent Actel antifuse, flash, and radiation-tolerant product families. A typical system implementation using the Core429 is shown in [Figure 1-1](#).

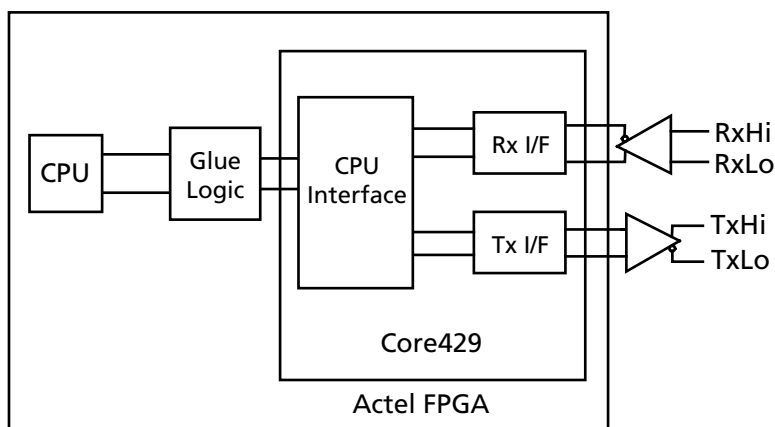


Figure 1-1 • Typical Core429 System with One Tx and One Rx

A typical ARINC 429 system requires a connection to a host CPU, used to set up the core and initialize the label definitions within the core. Core8051 is used as the host CPU in the Core429 demonstration design. Connection to an ARINC 429 data bus requires external line drivers and line receivers ([Figure 1-1](#)).

Core429 is available in the following versions:

- An evaluation version that allows core simulation with Actel Libero IDE or ModelSim®
- An obfuscated version that provides obfuscated RTL and precompiled testbenches
- An RTL version with full access to the source code

Refer to the [Core429 Handbook](#) for more information.

Core429 Development Kit

The Core429 Development Kit includes the Fusion Advanced Development Kit, the IP-DC-SA IP Daughter Card, and the Core429-SA Daughter Card. This section describes the board components briefly. The CD contains the board schematic and other required files.

Fusion Advanced Development Kit Board Description

The Fusion Advanced Development Kit ([Figure 1-2 on page 1-10](#)) provides a low-cost board for the system management platform, using Actel Fusion® FPGA devices. The evaluation board supports an ARM® Cortex™-M1 embedded processor on a Fusion device in the FGG484 package. The evaluation board includes the following:

- Ethernet and USB-to-UART interface for communication with the Fusion FPGA
- SRAM, parallel flash, and SPI flash
- I²C interface, organic light-emitting diode (OLED)

- Temperature diode, potentiometer, and pulse-width modulation (PWM) circuit
- Mixed-signal header for several daughter cards to be attached for extended mixed-signal applications
- Programming stick header so the low-cost programming stick (LCPS) can be attached to the board for programming

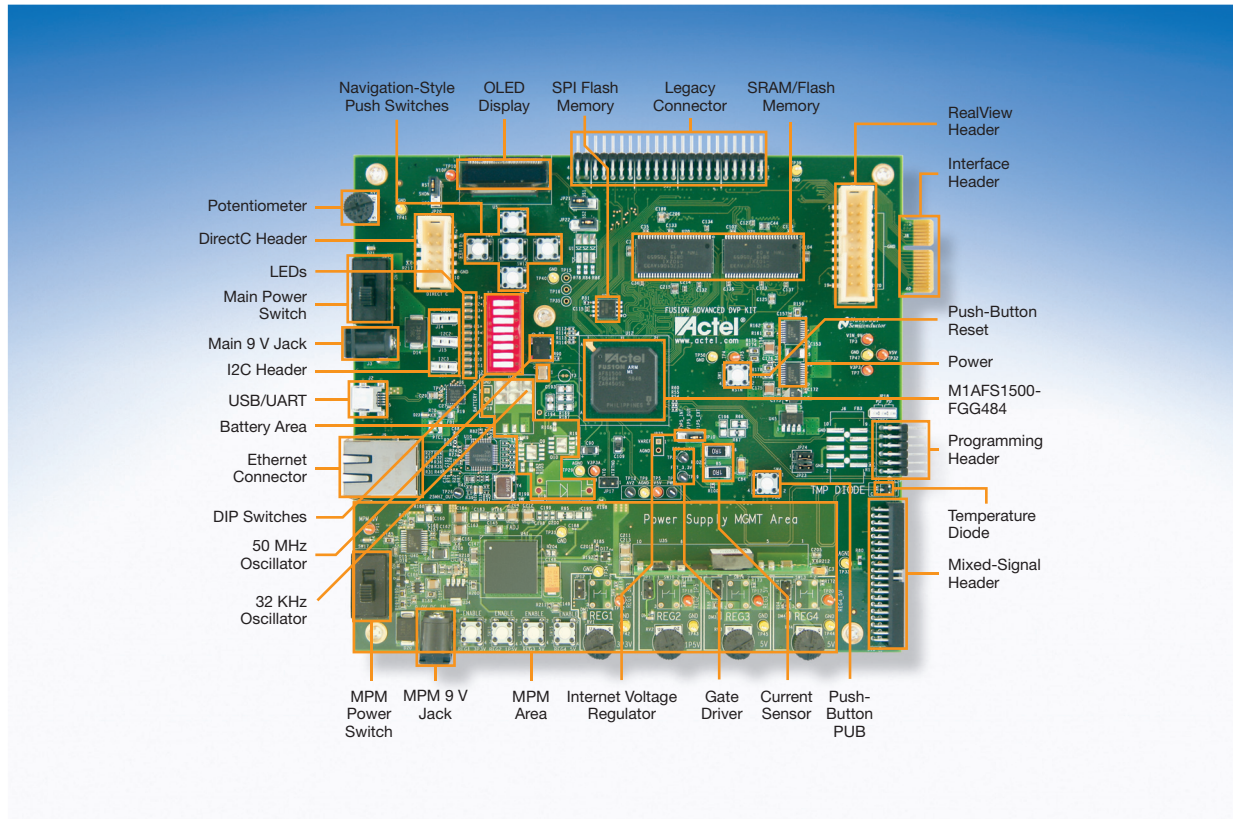


Figure 1-2 • Fusion Advanced Development Kit

Note: Make sure the jumpers are set according to the recommended default jumper settings defined in the *Fusion Advanced Development Kit User's Guide* (www.actel.com/documents/M1AFS_ADV_DEV_KIT_UG.pdf).

IP-DC-SA IP Daughter Card Description

The IP-DC-SA IP Daughter Card (Figure 1-3) acts as a buffer between the Fusion Advanced Development Kit board and the Core429-SA Daughter Card.

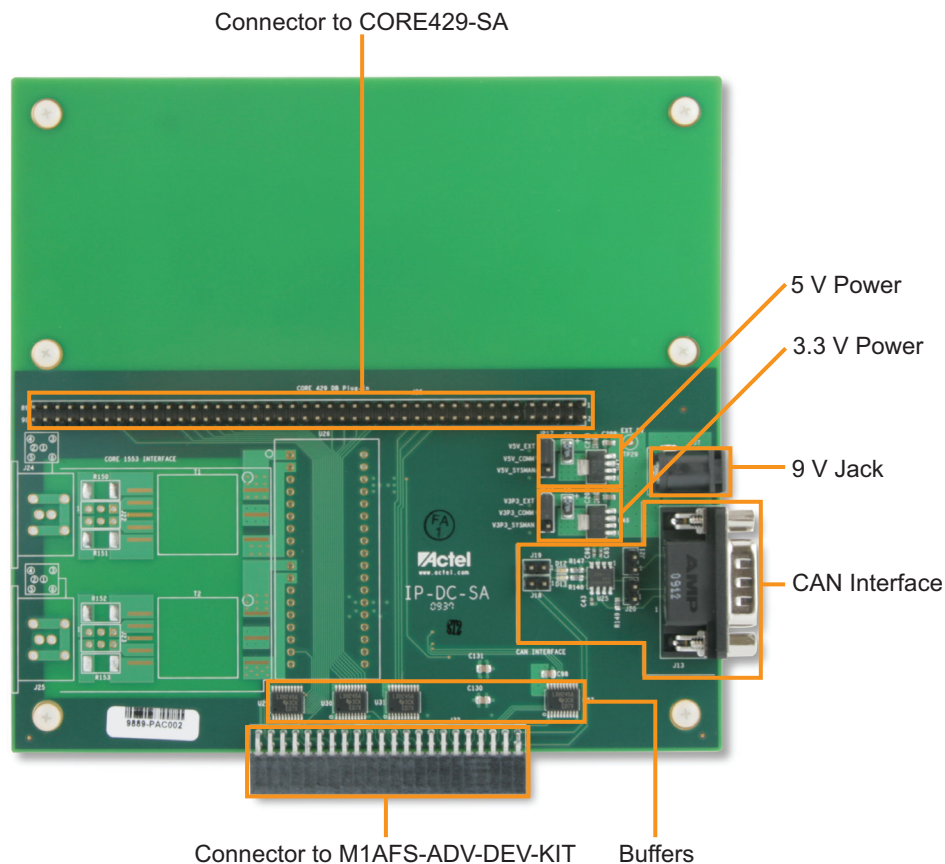


Figure 1-3 • IP-DC-SA IP Daughter Card

Jumpers J11 and J17

The daughter card must be connected to an external 9 V supply. When using the external power supply, make sure that JP11 and JP17 are properly installed, as described in Table 1-1.

Table 1-1 • JP11 and JP17 Settings

Jumper	Function	Default Setting
JP11	Jumper to select either external 3.3 V or 3.3 V provided through connector	Pin 1–2
JP17	Jumper to select either external 5 V or 5 V provided through legacy connector	Pin 1–2

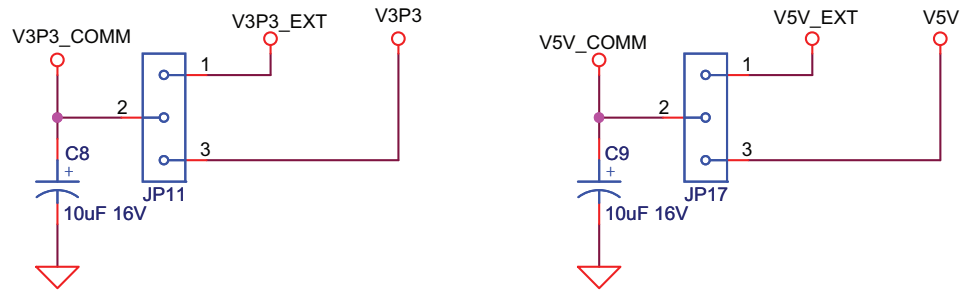


Figure 1-4 • Core429-SA Daughter Card Schematic

Core429-SA Daughter Card

The Core429-SA Daughter Card (Figure 1-5) contains the external line drivers and receivers needed to transmit and receive ARINC 429 data on an ARINC 429 data bus. The daughter card has four male DB9 connectors, one for each channel in the Core429 demonstration design. The Core429-SA Daughter Card connects to the IP-DC-SA IP Daughter Card board via the 90-pin connector, C429 Interface Connector.

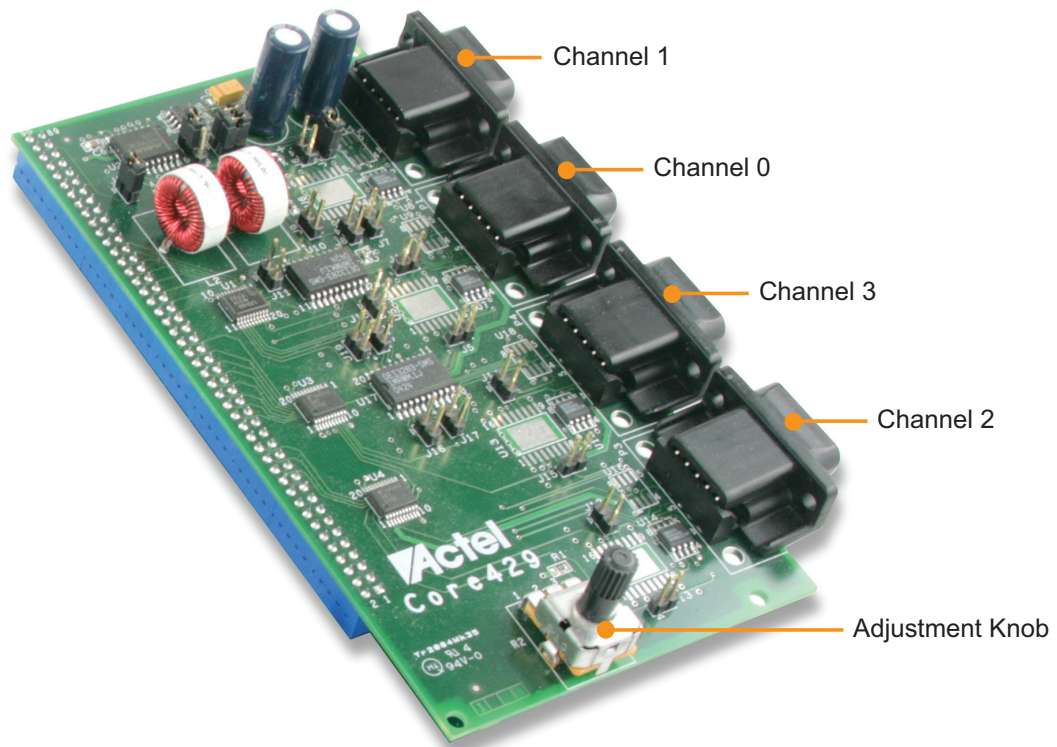


Figure 1-5 • Core429-SA Daughter Card

2 – Core429 Demonstration Design

The Core429 demonstration design (Figure 2-1) allows you to evaluate the ARINC 429 bus interface by implementing four ARINC 429 transmitters and four ARINC 429 receivers in a single M1AFS1500-FG484 FPGA.

FPGA Design

The demonstration design consists of the following blocks:

- Complete Core429 core with 4 transmitters and 4 receivers
- Core429 CPU interface
- 8051 CPU: Core8051 and its memory blocks
- Core8051 special function register (SFR) interface
- Core8051 serial interface

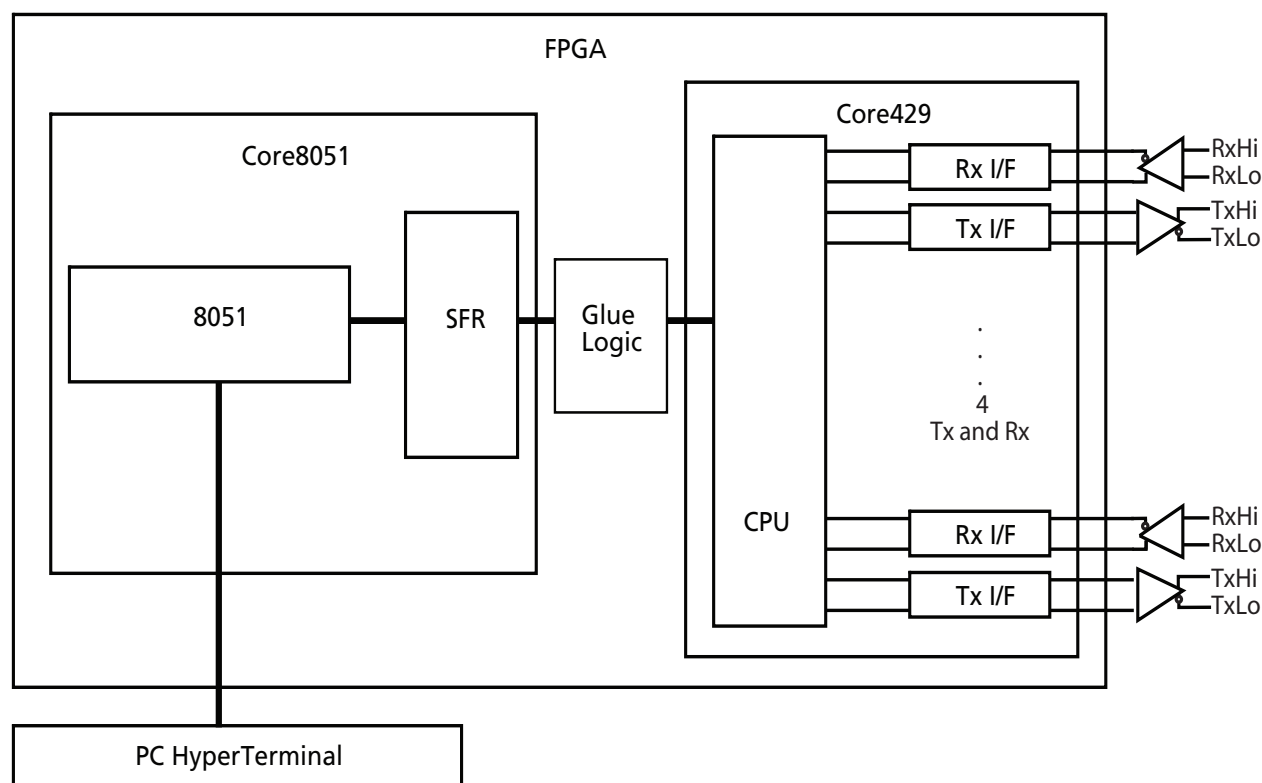


Figure 2-1 • Core429 Demo Design Architecture

Core429

Core429 is configured with four transmitter (Tx) and four receiver (Rx) blocks. Each Rx block is responsible for assembling the incoming serial data into 32-bit ARINC 429 words. The CPU can configure the Rx block to perform label comparisons and gap/parity checks on the incoming data. The default demonstration setup contains four Rx channels configured to run at a data rate of

100 Kbps. Each Tx block is responsible for transmitting ARINC data out of the internal FIFO when a complete ARINC word is written to the FIFO by the CPU interface. This block can be configured to operate in Loopback mode so that transmitted ARINC data does not leave the FPGA and is received by the corresponding Rx block. The default demonstration configuration contains four Tx channels that run at a data rate of 100 Kbps and have loopback disabled (messages are sent through the connectors, not internally looped back). You can change the setting from the control register. Refer to the *Core429 Handbook* for more information.

Core429 CPU Interface

The CPU interface allows Core8051 to access the Rx and Tx blocks. Access is necessary to configure control registers of both the Rx and Tx blocks, monitor the status registers, write to Rx label memory, and read/write ARINC data.

Core8051

Core8051 is an 8-bit microprocessor used to control the ARINC system for the demonstration design and interfaces with the serial interface via the special function register bus. Refer to the *Core8051 Datasheet* for more information.

Core8051 Special Function Register (SFR) Interface

The Core8051 SFR interface provides access to the Core8051 memory-mapped special function registers that reside either internally or externally. The demonstration design implements external special function registers to communicate between the system CPU (Core8051) and Core429. The SFR interface is controlled by Core8051 and communicates with the Core429 CPU interface to address the external SFRs and provide the data necessary for the Core429 system in the demonstration design to operate.

Core8051 Serial Interface

The Core8051 serial interface allows Core8051 to communicate with the UART port, which is connected to the off-chip USB-to-UART interface. The USB-to-UART interface allows HyperTerminal to communicate with the Fusion FPGA via a USB port, instead of a physical RS-232 serial communication port. HyperTerminal is a serial communications application program that can be installed in the Windows operating system. With a USB driver properly installed, and the correct COM port and communication settings selected, you can use HyperTerminal to communicate with a design running on the Fusion FPGA device. Core8051 serial communication utilizes Core8051's internal serial channel and interrupt services.

DIP Switch

Configuration of the demonstration design is handled by the 8-position DIP switch located at position S1. The DIP switch is also used to copy the demonstration design from on-board SRAM to on-board flash memory. [Figure 2-2 on page 2-15](#) shows the DIP switch bank. These switches are

wired to inputs of the FPGA so that open corresponds to logic 1 (4.7 K pull-up to 3.3 V) and closed corresponds to logic 0 (GND).

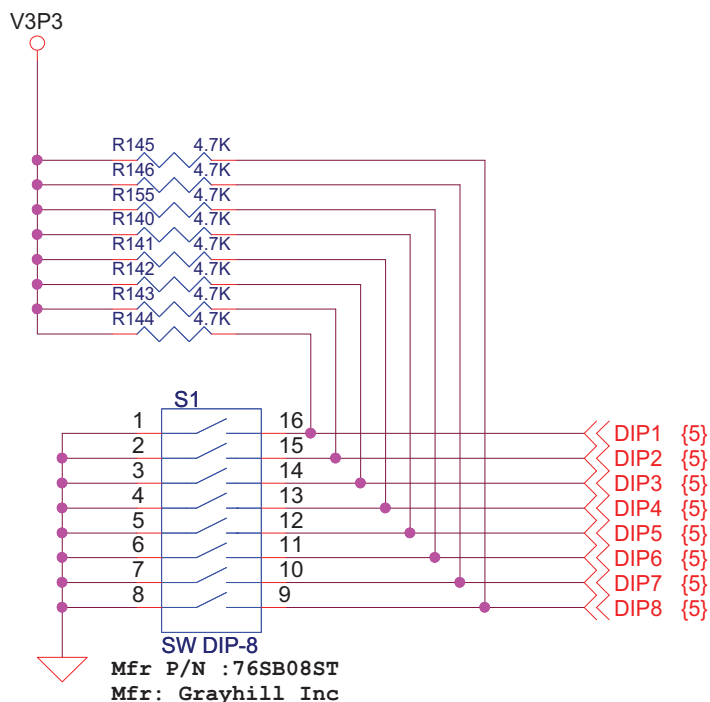


Figure 2-2 • DIP Switch Schematic

HyperTerminal Commands

Table 2-1 lists the available commands for use in controlling the demonstration design via HyperTerminal.

Table 2-1 • Core429 Demonstration HyperTerminal Commands

Commands	Function	Format
Global Commands		
Ch	Set current channel	ch <X> (0-3)
Esc	Enter command mode, exit auto Tx/Rx mode	–
Version	Displays version of terminal interface	–
Channel Commands		
Adlbl	Add a label value to Rx label memory	adlbl <01 [02] [03] ... [FF]> (maximum 80 characters per line)
Config	Current channel's Rx and Tx settings	–
Conrx	Read current channel's Rx control register	conrx
	Write to Rx control register	conrx <XX>
Contx	Read current channel's Tx control register	contx
	Write to Tx control register	contx <XX>

Table 2-1 • Core429 Demonstration HyperTerminal Commands (continued)

Commands	Function	Format
Label	Reset label memory and set new labels. Predefined labels shown in Table 2-2.	label <01 [02] [03] ... [FF]> (maximum 80 characters per line)
Loop	Enable/disable internal Loopback mode for current channel	loop <X> (0 – disable, 1 – enable)
Reg	Write data to a register on current channel	reg <N> <data> (N is 9-bit CPU address in hex)
	Read data from a register	reg <N> (N is 9-bit CPU address in hex)
Rx	Empty current channel's Rx FIFO to screen	–
Statrx	Read current channel's Rx status register	–
Stattx	Read current channel's Tx status register	–
Tx	Transmit data on current channel	tx <XXXXXXXX> (8 hex characters maximum)

Note: HyperTerminal specifies all ARINC data and labels as hexadecimal values.

HyperTerminal Command Examples

Table 2-2 lists HyperTerminal command examples.

Table 2-2 • Core429 HyperTerminal Command Examples

Keystrokes	Description
>ch 0 [return]	Select channel 0.
>reg 000 0 [return]	Read Rx register 0 on channel 0. <i>Note:</i> Refer to the CPU address bit positions of the Core429 handbook for more information.
>reg 064 0 [return]	Write 0 to Rx register 1 on channel 3. <i>Note:</i> Refer to the CPU address bit positions of the Core429 handbook for more information.
>tx ABCDEF12 [return]	Transmits ABCDEF12 from Core429 transmitter on the current channel. Same effect as the following sequence (assume ch0): >reg 10 ABCDEF12 [return]
>rx [return]	Reads Rx FIFO on the current channel until the FIFO is empty. Same effect as repeatedly typing (while mode = 0): >reg 0 [return]
>label 12 45 4 9 [return]	Resets the current channel's Rx label memory and writes hexadecimal values 12, 45, 4, and 9. Same effect as the following: >reg 64 40 [return] (to tell the Rx block to reset label memory) >reg 6C 12 [return] >reg 6C 45 [return] >reg 6C 4 [return] >reg 6C 9 [return]

Table 2-2 • Core429 HyperTerminal Command Examples (continued)

Keystrokes	Description
>adlbl 5 8 7 1 3 [return]	Adds 5, 8, 7, 1, and 3 to existing values in the Rx label memory without resetting the label memory
>contx 02 [return]	Writes 2 (00000010) to the current channel's Tx control register (Loopback mode enabled)

Demonstration Design

Demonstration Configuration

On power-up, the default configuration information is written to the control register of each Rx and Tx block.

The default Rx block configuration is as follows:

- Data rate is 100 Kbps
- Label recognition is enabled
- Parity is disabled
- Decoder is disabled
- Bit matching is disabled
- The label memory is not initialized.

The default Tx block configuration is as follows:

- Data rate is 100 Kbits/s
- Loopback is disabled
- Parity is disabled

The Rx and Tx configuration can be changed by using HyperTerminal to write the appropriate values to the Rx and Tx control registers. Refer to the [Core429 Handbook](#) for more information about the appropriate register values for each available configuration.

Setting Up the Demonstration Design

The Fusion device in the Core429 Development Kit boards is preprogrammed with demonstration designs. If you want to reprogram the FPGA, or if the FPGA is reprogrammed with a different design, follow the steps in the "[Programming the Flash Memory for Core8051](#)" section on [page A-25](#) to program the FPGA. In addition, this demonstration design uses Core8051, which can be run from flash or SRAM. In this demo we will run from SRAM. If you want to run from flash, follow the steps in the "[Programming the Flash Memory for Core8051](#)" section on [page A-25](#).

Once the FPGA is programmed with the correct design, set up the board according to the steps below:

1. Connect one end of the USB mini cable to USB port J2 on the M1AFS-ADV-DEV-KIT board (labeled USB2 in [Figure 2-3 on page 2-18](#)) and connect the other end to the USB port on your PC.

2. Connect one end of a 9 V power supply to power input J3 on the M1AFS-ADV-DEV-KIT board (Figure 2-3) and plug the supply into an electrical outlet.

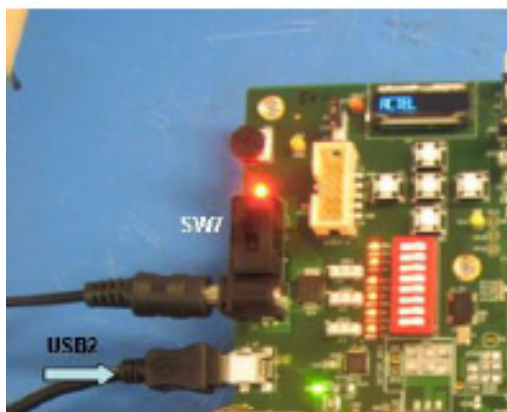


Figure 2-3 • Connecting the USB and 9 V Power Supplies on the M1AFS-ADV-DEV-KIT Board

3. Set the DIP switch at S1 to the default configuration of [8:1] = 11000000.

Note: These switches are wired to inputs of the FPGA so that open corresponds to logic 1 (4.7 K pull-up to 3.3 V) and closed corresponds to logic 0 GND.

4. Determine the COM port assigned to the USB interface. Open the Windows Control Panel and double-click the **System** icon. Click the **Hardware** tab and click the **Device Manager** button. Expand the **Ports (Com & LPT)** item in the Device Manager. Look for the CP2102 USB to UART Bridge Controller in the list of ports. The COM port in parentheses identifies the COM port assigned to this device (Figure 2-4). If you do not see CP2102, you need to install the driver from the CP210x_Drivers.zip archive. Refer to "Installing the M1AFS-ADV-DEV-KIT Board USB Serial Driver" section on page C-31 for more information.

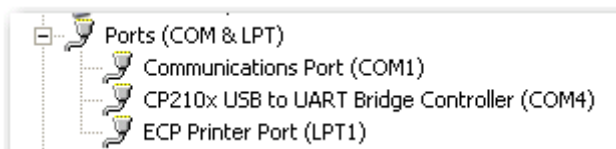


Figure 2-4 • Determining the USB Serial Port's COM Port

5. Open HyperTerminal and set the communication parameters as shown below. Set the COM port to match the USB serial port of the board, as shown in Step 4.
 - 2,400 bits per second
 - 8 data bits
 - Parity set to none
 - 1 stop bit
 - Flow control set to none

- From the **File** menu, choose **Properties**, and click the **Settings** tab to open the HyperTerminal Properties Settings window (Figure 2-5).

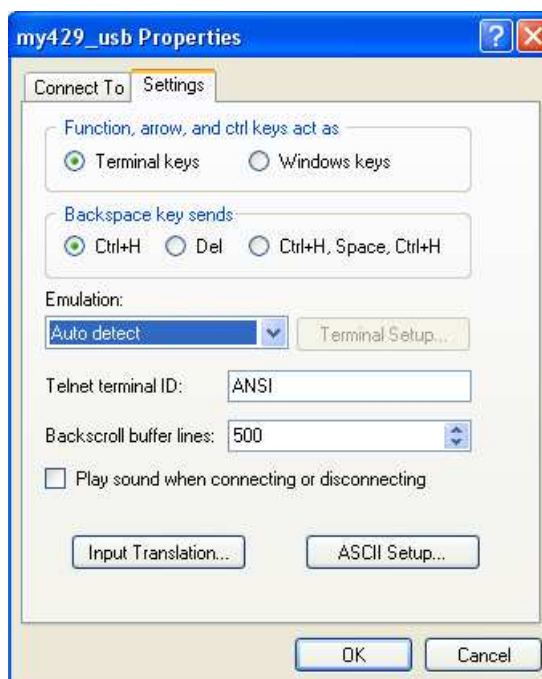


Figure 2-5 • HyperTerminal Properties Settings Window

- Click the **ASCII Setup** button to open ASCII Setup window and change Line delay to 500 milliseconds (Figure 2-6).

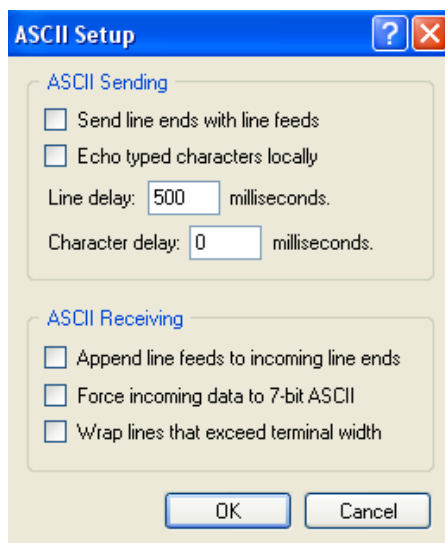


Figure 2-6 • Setting ASCII Setup in HyperTerminal

Running the Demonstration Design

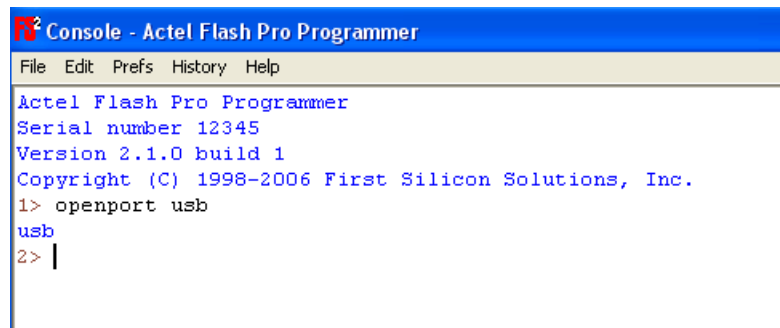
The next step is to load the 8051 hex file in SRAM using FS2 debugger software. If you want to run the demo from flash, you must program the flash first. Refer to the "Programming the Flash Memory for Core8051" section on page A-25.

To load the hexadecimal files via the ISA-Actel51 debugger:

1. Ensure that you have the ISA-Actel51 debugger software installed. You can download it from the Actel website (www.actel.com/download/reg/default.aspx?f=ISA_Actel51).
2. Turn the power switch on the M1AFS-ADV-DEV-KIT board, SW7, to the ON position.
3. Launch the ISA-Actel51 debugger.

Note: If this is your first time launching the debugger, you will be prompted to select the port to use for default communication. Select USB and accept the default TCK frequency (4 MHz) and the default TV_{CC} value (2500 mV).

4. From the ISA-Actel51 debugger **Window** menu, select **Console** to open the FS2 ISA-Actel51 Console, and type the **openport usb** command (Figure 2-7).



```
Console - Actel Flash Pro Programmer
File Edit Prefs History Help
Actel Flash Pro Programmer
Serial number 12345
Version 2.1.0 build 1
Copyright (C) 1998-2006 First Silicon Solutions, Inc.
1> openport usb
usb
2> |
```

Figure 2-7 • Setting USB in ISA-Actel51 Console

5. Close the ISA-Actel51 Console.
6. From the **Tools** menu of the ISA-Actel51 debugger, select **Load Hex**.
7. Set the address to **0x0000x** and the filename to **Demo.hex**. The Demo.hex file is included in the Core429_DEV_KIT_DS.zip file, in the *Software_Hex_file/HexFile* folder.

- Click **OK** and wait for the hex file to be loaded (Figure 2-8).

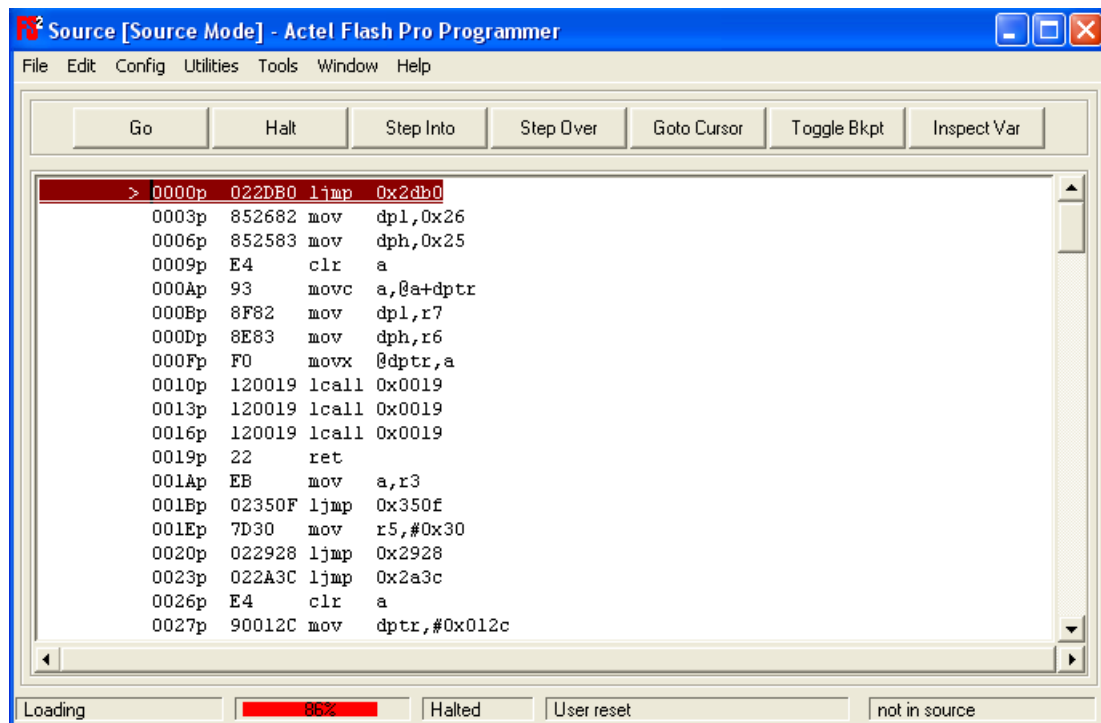
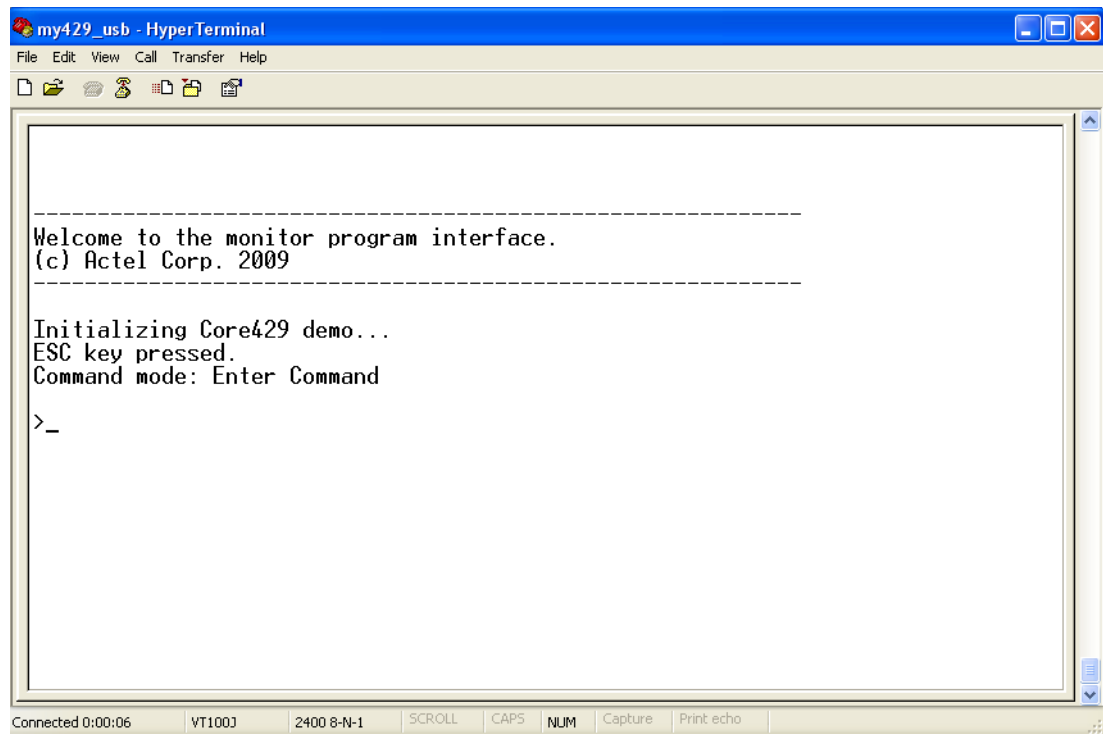


Figure 2-8 • Loading the Hexadecimal File with the ISA-Actel51 Debugger

- When loading is completed, press the **Go** button.

10. Core8051 will start running and the demo design will display a message. Press **Esc** to enter the command prompt on HyperTerminal (Figure 2-9).



```
-----
Welcome to the monitor program interface.
(c) Actel Corp. 2009
-----
Initializing Core429 demo...
ESC key pressed.
Command mode: Enter Command
>_
```

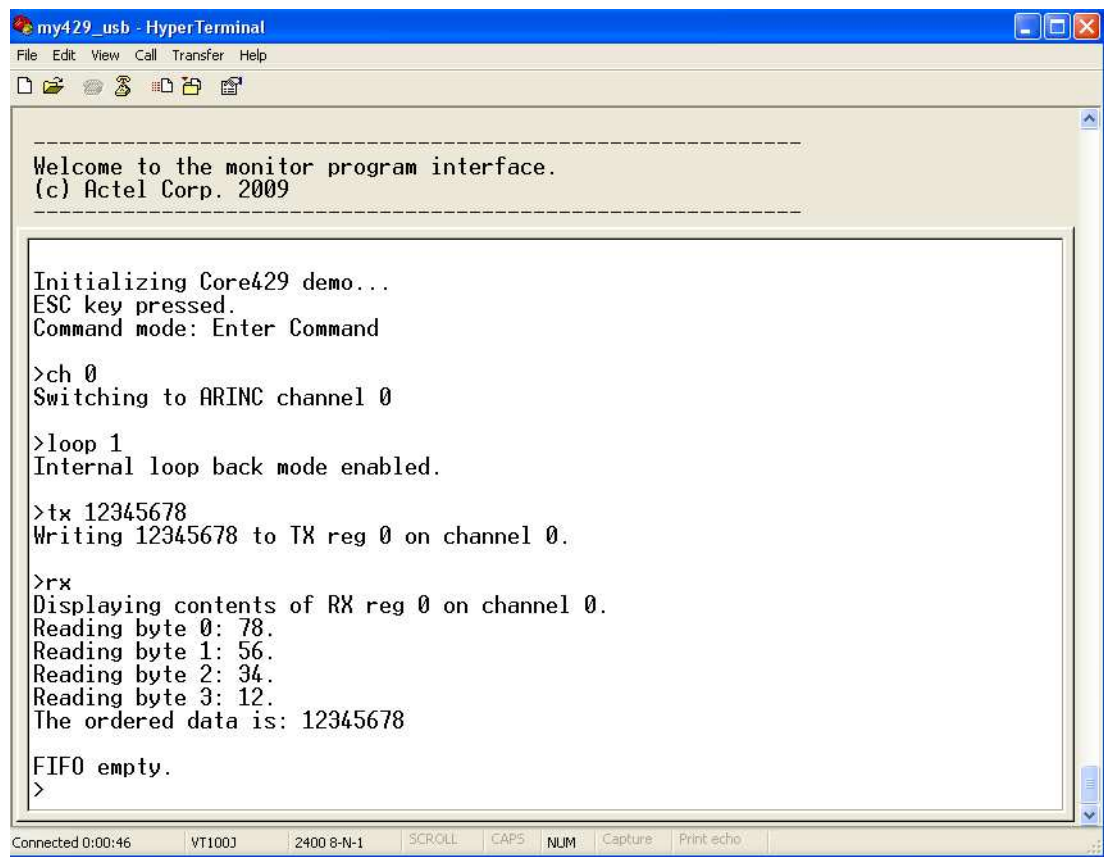
Figure 2-9 • HyperTerminal Window

Running the Demonstration in Internal Loopback Mode

In this section, the demonstration design will run while Core429 is operating in internal Loopback mode. The Core429-SA Daughter Card is not required for this mode of operation, since the transmitted data does not leave the FPGA, but is looped to the Rx block on the same channel as the transmitter.

1. Select a channel using the **ch** command; for example: **ch 0**.
2. Enable loopback on that channel by typing **loop 1 [return]**.
3. Transmit 32-bit ARINC data by typing **tx <data> [return]**, where <data> represents eight hexadecimal values to comprise a 32-bit ARINC data word.

4. Empty the Rx FIFO by typing `rx` [return].
Compare the received data with the transmitted data (Figure 2-10).



```
my429_usb - HyperTerminal
File Edit View Call Transfer Help

-----
Welcome to the monitor program interface.
(c) Actel Corp. 2009
-----

Initializing Core429 demo...
ESC key pressed.
Command mode: Enter Command

>ch 0
Switching to ARINC channel 0

>loop 1
Internal loop back mode enabled.

>tx 12345678
Writing 12345678 to TX reg 0 on channel 0.

>rx
Displaying contents of RX reg 0 on channel 0.
Reading byte 0: 78.
Reading byte 1: 56.
Reading byte 2: 34.
Reading byte 3: 12.
The ordered data is: 12345678

FIFO empty.
>
```

Figure 2-10 • HyperTerminal Window Showing Internal Loopback

Running the Demonstration in External Loopback Mode

In this section, the demonstration design will run while Core429 is operating in external Loopback mode with the Core429-SA Daughter Card. By connecting two of the channels using the supplied DB9 cables, the transmitted data will be received by the ARINC 429 Rx blocks.

1. Switch off the M1AFS-ADV-DEV-KIT board power using switch SW7 if the power is on.
2. Connect the Core429-SA Daughter Card to the M1AFS-ADV-DEV-KIT board.
3. Connect the second 9 V power supply to power input J7 of the Core429-SA Daughter Card on the IP-DC-SA IP Daughter Card and plug the supply into an electrical outlet.
4. Connect channels 0 and 1 using the supplied DB9 cables. Channel 0 is the left-most channel and channel 1 is on its immediate right.
5. Turn the power switch on the M1AFS-ADV-DEV-KIT board, SW7, to the ON position. After a brief initialization period, HyperTerminal will display a status message:

```
-----  
Welcome to the monitor program interface.  
(c) Actel Corp. 2009  
-----
```

```
Initializing Core429 demo...  
ESC key pressed.  
Command mode: Enter Command
```

>

6. From the **Transfer** menu, choose **Send Text File**. Browse to select the script file `channel0_to_channel1.txt` and click **Open** to start the download. The script programs Tx and Rx of channel 0 and channel 1, sends data from channel 0 to channel 1, and reads data from channel 1.

The script folder contains several text files that can be used to test additional modes. You can also create your own text file to configure the Core429 modes of operation. Refer to the [Core429 Handbook](#) for more information about implementing the various configurations of Core429 and HyperTerminal.

A – Programming the Flash Memory for Core8051

1. If you do not have the ISA-Actel51 debugger software, download the software from the Actel website (www.actel.com/download/reg/default.aspx?f=ISA_Actel51) and install it.
2. Connect the USB cable between the J1 pins on the Actel M1AFS-ADV-DEV-KIT board and the PC through the Actel low-cost programming stick.
3. Connect the USB cable between the J2 pins on the Actel M1AFS-ADV-DEV-KIT board and the PC for communication via HyperTerminal.
4. Connect one end of a 9 V power supply to the power input J3 on the M1AFS-ADV-DEV-KIT board. Plug the supply into an electrical outlet.
5. Set the DIP switch S1 to the following configuration: Switch[8:1] = 11000000.

Note: These switches are wired to inputs of the FPGA so that open will correspond to logic 1 (4.7 K pull-up to 3.3 V) and closed will correspond to logic 0 GND.

6. Turn the power switch on the M1AFS-ADV-DEV-KIT board, SW7, to the ON position.
7. Determine the COM port assigned to the USB interface. Open the Windows Control Panel and double-click the **System** icon. Click the **Hardware** tab and click the **Device Manager** button. Expand the **Ports (Com & LPT)** item in the Device Manager. Look for the CP2102 USB to UART Bridge Controller in the list of ports. The COM port in parentheses identifies the COM port assigned to this device ([Figure A-1](#)). If you do not see CP2102, you need to install the CP210x_Drivers driver. Refer to the "[Installing the M1AFS-ADV-DEV-KIT Board USB Serial Driver](#)" section on [page C-31](#) for more information.

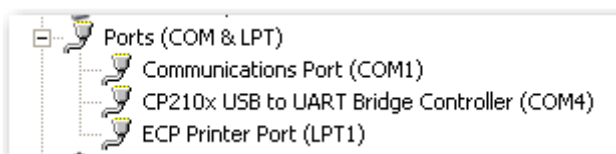


Figure A-1 • Determining the USB Serial Port's COM Port

8. Open HyperTerminal and set the communications parameters as shown below. Set the COM port to match the USB serial port of the board, as shown in [Step 7](#).
 - 2,400 bits per second
 - 8 data bits
 - Parity set to none
 - 1 stop bit
 - Flow control set to none

If you want to use HyperTerminal to send data to the demonstration design, you need to change Line delay to 500 milliseconds.

9. Launch the ISA-Actel51 debugger.

Note: If this is your first time launching the debugger, you will be prompted to select the port to use for default communication. Select USB and accept the default TCK frequency (4 MHz) and the default TV_{CC} value (2500 mV).