7 Series FPGA AMS Targeted Reference Design

User Guide

Vivado Design Suite 2013.3

UG960 (v5.0) November 22, 2013

XILINX®

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
12/20/2012	1.0	Initial Xilinx release.
02/05/2013	2.0	Updated for Vivado® Design Suite 2012.4 and ISE® Design Suite 14.4. For this update, KC705, AC701, and VC707 designs support the Vivado Design Suite and the ZC702 design still supports the ISE Design Suite. Chapter 1, Introduction: Resource utilization tables were updated, and the AC701 Resource Utilization table was added (Table 1-2 to Table 1-5). Chapter 2, Getting Started: Added jumper settings for the AC701 board to Table 2-1. In Rebuilding the Hardware Design, page 25, *PlanAhead*™ was replaced with *Vivado* and the procedures were adjusted for the Vivado tool. Added the section Rebuilding the Hardware Design for ZC702, page 27. Chapter 3, Functional Description: Added section AC701 AMS Power Demo Design using XADC, page 58. Appendix E, Additional Resources: Updated resources and added resources for the Artix®-7 FPGA.

Date	Version	Revision
04/17/2013	3.0	Updated for Vivado Design Suite 2013.1 and ISE® Design Suite 14.5. Added a note after Table 1-5 about resource utilization number variation. In Hardware and Software Setup, page 14, Step 8: updated the reference to the `AMS101 Evaluation GUI V1.1.exe` file. Figure 2-10 and Figure 2-11 were updated to reflect the new revision 1.1. Figure 2-13 was replaced. Test Setup Requirements, page 24 was updated to refer to boards KC705 Revision 1.2, VC707 Revision 1.2, AC701 Revision 1.1, and ZC702 Revision 1.2. Rebuilding the Software Design, page 30 was updated for reference to SDK 14.5. In Equation 3-5, the denominator value was changed to *4096*. Table B-1, at row Connection Establishment: and column UART Reception, replaced old values of "[...] `0x1100`, `0x2100`, `0x3100`, and `0x4100` [...]" with new "[...] `0x1110`, `0x2120`, `0x3120`, and `0x4120` [...]." Table B-1, at row Decimation Value and column UART Reception, replaced old decimation value of "[...] 1, 2, 4, 8, 16, and 32 [...]" with new "[...] 1, 2, 4, 8, and 16 [...}." Appendix E, Additional Resources and cross references were updated. LabVIEW 64-bit Run-Time Engine 2011 was removed from References, page 81.
07/24/2013	4.0	Updated for Vivado Design Suite 2013.2. Updated step 7. d. under Quick Start, page 14. Replaced section Rebuilding the Hardware Design for ZC702, page 27. Updated links throughout the document.
11/22/2013	5.0	Updated for Vivado Design Suite 2013.3. Updated Rebuilding the Hardware Design, page 25. Updated Rebuilding the Hardware Design for ZC702, page 27. Updated the directory structure in Appendix C, Directory Structure and File Descriptions. Updated the link to the Declaration of Conformity, page 83.

Table of Contents

Appendix F: Regulatory and Compliance Information

Appendix G: Warranty

XILINX®

Chapter 1

Introduction

About this Guide

This Analog Mixed Signal (AMS) Targeted Reference Design (TRD) guide provides a reference design for evaluating various metrics and performance of the Xilinx Analog-to-Digital Converter (XADC) block. This guide provides an out-of-the box approach to evaluating the XADC block.

This document contains the following chapters:

- Chapter 1, Introduction introduces features of the reference design and terms used in this document.
- Chapter 2, Getting Started provides a step-by-step guide on setting up the design and preparing it to work.
- Chapter 3, Functional Description provides hardware and software design details.
- Chapter 4, Understanding ADC Metrics explains various ADC metrics, how to measure them, and analyzes the results obtained.
- Chapter 5, Applications describes the interactions of the LabVIEW GUI and the MicroBlaze™ processor.
- Appendix A, ADC Basics presents basic information about analog-to-digital conversion.
- Appendix B, Register Descriptions defines register-based communication between the LabVIEW GUI and the hardware design.
- Appendix C, Directory Structure and File Descriptions presents the structure of the design.
- Appendix D, Troubleshooting suggests some ideas if the design is not working as expected.
- Appendix E, Additional Resources lists Xilinx support sites and all references cited in the document.
- Appendix F, Regulatory and Compliance Information describes reference design compliance with various standards and directives.
- Appendix G, Warranty

List of Abbreviations

Table 1-1 lists abbreviations used in this document.

Table 1-1: **List of Abbreviations**

Abbreviation	Description
ADC	Analog-to-Digital Converter
ADD	Architecture Description Document
AMS	Analog Mixed Signal
DAC	Digital-to-Analog Converter
DNL	Differential Nonlinearity
EDK	Embedded Development Kit
EOC	End of Conversion
FFT	Fast Fourier Transform
FIR	Finite Impulse Response
FPGA	Field Programmable Gate Array
I2C	Inter-Integrated Circuit
INL	Integral Nonlinearity
MSPS	Mega-Samples Per Second
PMBus	Power Management Bus
RMS	Root Mean Square
RTE	Run Time Engine
SDK	Software Development Kit
SFDR	Spurious Free Dynamic Range
SINAD	Signal to Noise and Distortion ratio
SNR	Signal to Noise Ratio
SPI	Serial Peripheral Interface
THD	Total Harmonic Distortion
TRD	Targeted Reference Design
UART	Universal Asynchronous Receiver Transmitter
USB	Universal Serial Bus
XADC	Xilinx Analog-to-Digital Converter
XPS	Xilinx Platform Studio

Overview

The Analog Mixed Signal technology in Xilinx® 7 series devices offers a combination of a flexible analog block (the Xilinx® Analog-to-Digital Converter (XADC)) and programmable logic to scale and customize common analog interface requirements.

This design combines with the AMS101 evaluation card for easy evaluation of key performance metrics of the XADC. The AMS101 evaluation card provides a basic signal source in the form of a dual Digital-to-Analog Converter (DAC). External analog signals can also be applied to test points on the AMS101 card. The AMS evaluator LabVIEW GUI allows easy configuration of XADC operating modes, data collection, and analysis [Ref 1].

Figure 1-1 is a block representation of the XADC evaluation design. The design running on the 7 series FPGA is built using the Embedded Development Kit (EDK). All blocks represented in the FPGA design are available as IP cores from Xilinx.

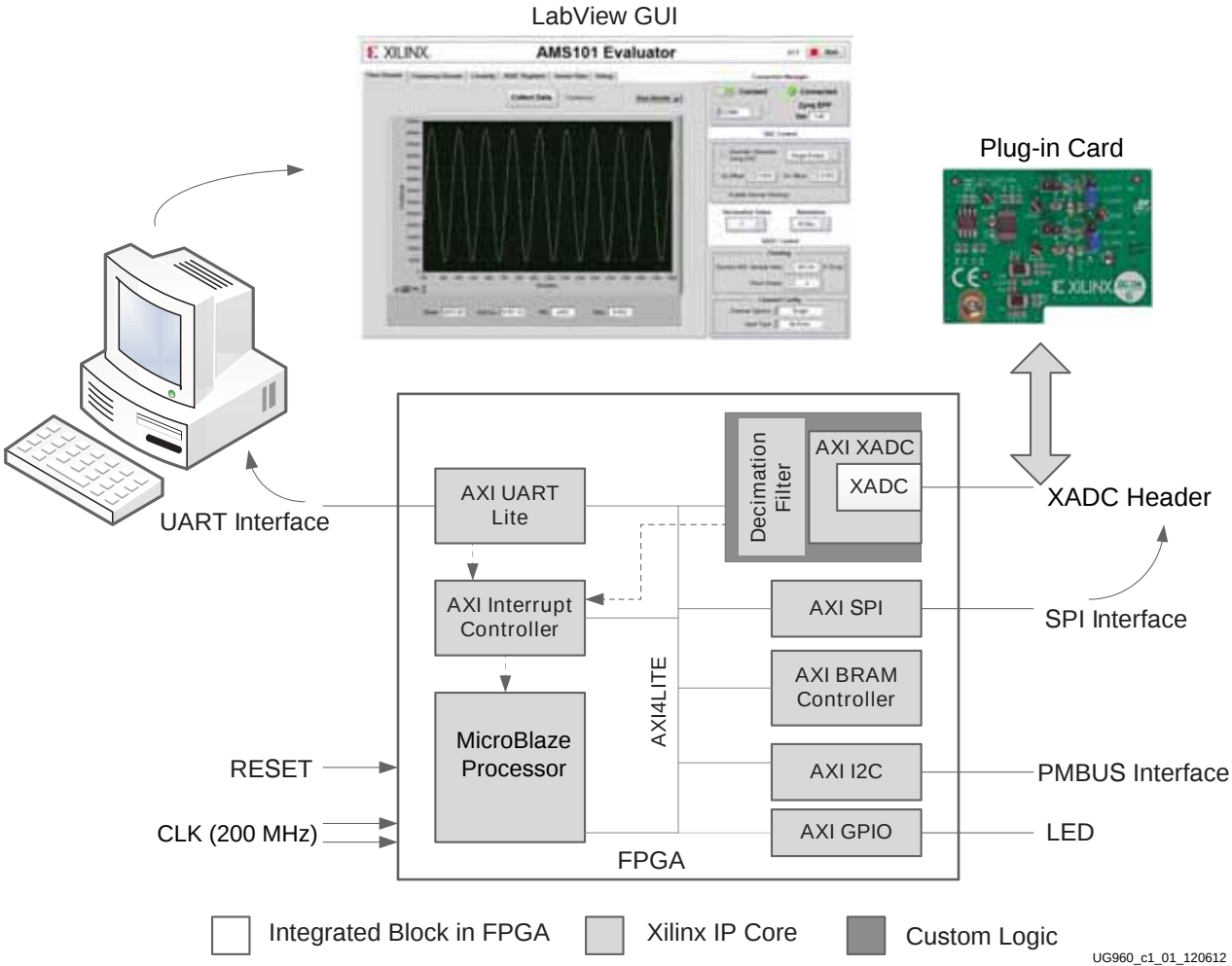


Figure 1-1: **XADC Evaluation Design Block Diagram**

The AMS evaluation platform consists of a Xilinx 7 series device base board (KC705, VC707, AC701, or ZC702), a plug-in mezzanine card (AMS101 evaluation card with DAC and signal conditioning), and a LabVIEW-based GUI (AMS Evaluator). XADC data is stored in the FPGA memory (block RAM) in raw 16-bit format and is transferred through the USB UART to the host PC, where it is processed and displayed in an easily

understandable format on the LabVIEW GUI. Additionally, the GUI provides options for configuring XADC operating modes.

A 200 MHz differential clock available on the KC705, VC707, AC701, and ZC702 boards is used and 100 MHz is derived from the differential clock for FPGA design operation.

Features

The following are features of the AMS reference design:

- XADC operating at a maximum supported rate of 1 mega-samples per second (MSPS)
  - Control of the sampling rate
  - Control of input signal type (bipolar or unipolar)
- Static characteristics
  - Integral non-linearity
  - Differential non-linearity
- Dynamic characteristics derived from Fast Fourier Transform
  - Signal-to-noise ratio
  - Total harmonic distortion
  - Signal-to-noise and distortion
  - Spurious free dynamic range
  - Effective number of bits
- Temperature sensor providing die temperature
- Power supply sensor monitoring V_{CCAUX}, V_{CCINT}, and V_{CCBRAM}
- Control of onboard power controllers for voltage variation

Resource Utilization

Resource utilization for KC705 is listed in the Table 1-2.

Table 1-2: **KC705 Resource Utilization**

Resource	Total Available	Usage
Slice registers	407,600	2,722 (1%)
Slice LUT	203,800	3,270 (1%)
RAMB36E1	445	64 (14%)
MMCME2_ADV	10	1 (10%)
DSP48E1 slice	840	63 (7%)
XADC	1	1 (100%)

Resource utilization for VC707 is listed in the Table 1-3.

Table 1-3: **VC707 Resource Utilization**

Resource	Total Available	Usage
Slice registers	607,200	2,753 (1%)
Slice LUT	303,600	3,353 (1%)
RAMB36E1	1,030	64 (6%)
MMCME2_ADV	14	1 (7%)
DSP48E1 slice	2,800	63 (2%)
XADC	1	1 (100%)

Resource utilization for AC701 is listed in the Table 1-4.

Table 1-4: **AC701 Resource Utilization**

Resource	Total Available	Usage
Slice registers	267,600	2,741 (1.02%)
Slice LUT	133,800	3,497 (2.61%)
RAMB36E1	365	64 (17.35%)
MMCME2_ADV	10	1 (10%)
DSP48E1 slice	740	63 (8.51%)
XADC	1	1 (100%)

Resource utilization for ZC702 is listed in the Table 1-5.

Table 1-5: **ZC702 Resource Utilization**

Resource	Total Available	Usage
Slice registers	106,400	1,463 (1%)
Slice LUT	53,200	1,550 (2%)
RAMB36E1	140	32 (22%)
MMCME2_ADV	4	0 (0%)
DSP48E1 slice	240	60 (27%)
XADC	1	1 (100%)

Note: The resource utilization numbers might vary by specific tool revisions.

Chapter 2

Getting Started

To facilitate easy evaluation of key performance metrics of the XADC and AMS technology, Xilinx developed the AMS evaluation platform for all 7 series FPGA and Zynq®-7000 AP SoC base boards. The AMS evaluation platform enables key ADC performance metrics to be observed and evaluated. The remainder of this document describes in detail the hardware and software that comprise the AMS evaluation platform.

Figure 2-1 shows the evaluation platform used for KC705 base board. A similar platform is used on the VC707 and AC701 boards. For the ZC702 board, a Cortex™-A9 processor is used instead of a MicroBlaze™ processor.

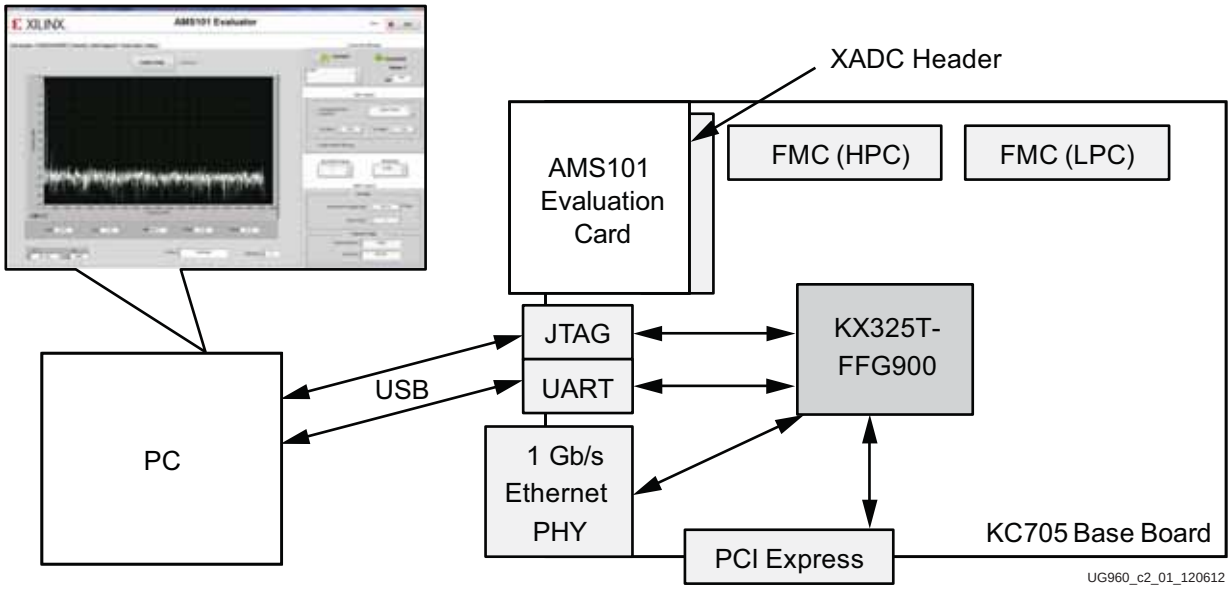


Figure 2-1: **AMS Evaluation Platform**

AMS Evaluation Platform Features

The AMS evaluation platform provides:

- A complete XADC and AMS evaluation solution
- An onboard signal source
- Configurable analog inputs
- An interactive GUI
- Interfaces for Xilinx FPGA base boards (the full list of supported base boards is listed in References, page 81)

Each base board kit contains:

- One AMS101 evaluation card
- USB-UART drivers
- A base board *Getting Started Guide*

Quick Start

Ten steps are needed to get the AMS evaluation platform up and running. This chapter covers how to perform these steps as well as how to run key ADC performance tests after set up.

Note: This document refers to the TRD version (v1_0) that was initially released. For subsequent releases, the design version will be upgraded but changes will not be reflected in this document. Though screenshots and figures will not show the upgraded versions of the TRD, this document will still apply to the updated design.

Hardware and Software Setup

1. Install the AMS Evaluator tool GUI.

   Download the AMS Evaluator installer files (7 Series FPGA and Zynq AMS Evaluator Installer for AMS Targeted Reference Design) at www.xilinx.com/support/documentation/ams101_evaluation_card.htm. Click the `setup.exe` file to install the National Instruments LabVIEW RunTime Engine needed to host the AMS Evaluator tool.

   The GUI itself has been built using National Instruments LabVIEW 2011 software. To enable use of the GUI without the need for a LabVIEW license, Xilinx has bundled the LabVIEW run-time engine with the GUI installer. During the installation process, the run-time engine is installed on the PC.

2. Connect the FPGA base board.

   Ensure that the FPGA base board power switch (e.g., SW15 on the KC705 base board) is in the OFF position. Figure 2-2 shows the position of the power switch on the board.

3. Connect the host PC to the UART port with the Standard-A plug to Mini-B plug USB cable. Also connect the Standard-A plug to Micro-B plug USB cable to the JTAG port. See the corresponding photo in the *Getting Started Guide* for each particular base board.

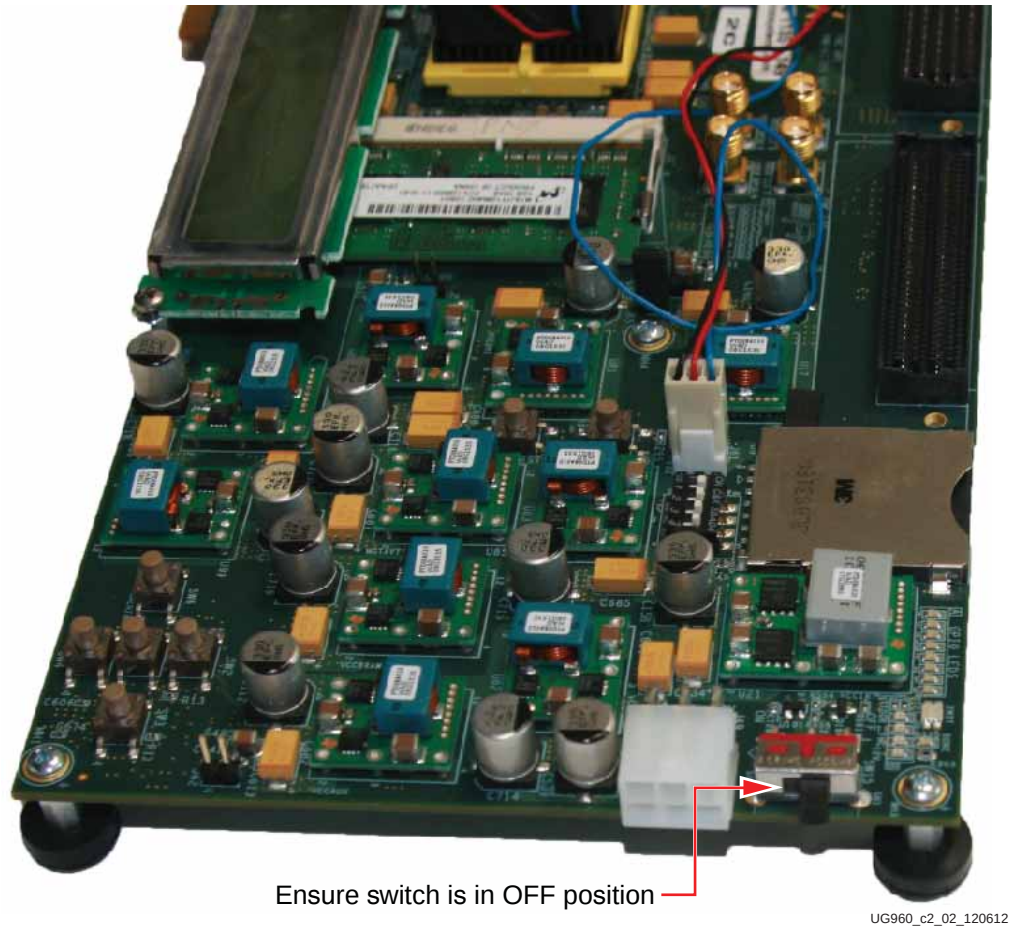


Figure 2-2: **Power Switch on the FPGA Base Board**

Figure 2-3 shows how to connect these on the KC705 base board.

Caution! Do not turn on the power switch until step 6, page 20.

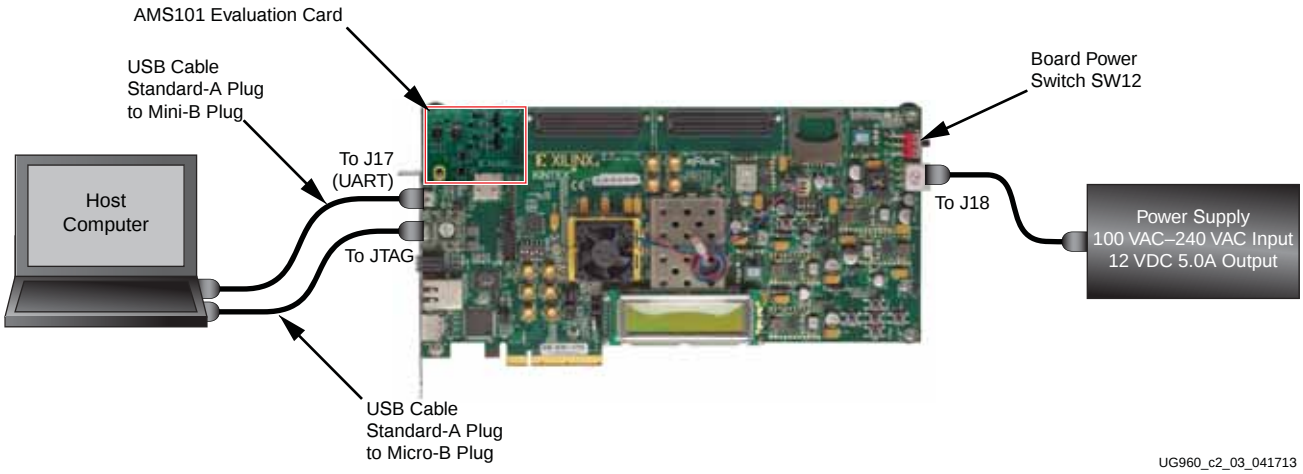


Figure 2-3: **FPGA Base Board Connectivity**

4. To enable AMS evaluation, configure the FPGA base board jumper settings as listed in Table 2-1.

   Note: The triangle indicates pin 1 for jumper settings on all Xilinx base boards.

Table 2-1: **Jumper Settings for Base Boards**

Jumper	Setting
Jumper Settings for the KC705 Board	
J43	In place
J68	In place
J48	In place between pins 2 and 3
J69	In place between pins 1 and 2
J47	In place between pins 1 and 2
J42	Not in place
Jumper Settings for the VC707 Board	
J10	In place
J53	In place
J43	In place between pins 2 and 3
J54	In place between pins 1 and 2
J42	In place between pins 1 and 2
J9	Not in place
Jumper Settings for the AC701 Board	
J43	In place between pins 2 and 3
J53	In place
J54	In place between pins 2 and 3
J42	In place between pins 1 and 2
J10	In place
J9	In place
J11	In place
Jumper Settings for the ZC702 Board	
J8	Not in place
J9	In place
J65	In place
J37	In place between pins 1 and 2

Table 2-1: **Jumper Settings for Base Boards *(Cont'd)***

Jumper	Setting
J38	In place between pins 2 and 3
J70	In place between pins 2 and 3

Notes:

1. See *KC705 Evaluation Board for the Kintex-7 FPGA User Guide* (UG810) [Ref 2] for more information.
2. See *VC707 Evaluation Board for the Virtex-7 FPGA User Guide* (UG885) [Ref 3] for more information.
3. See *AC701 Evaluation Board for the Artix-7 FPGA User Guide* (UG952) [Ref 4] for more information.
4. See *ZC702 Evaluation Board for the Zynq-7000 XC7Z020 All Programmable SoC User Guide* (UG850) [Ref 5] for more information.

5. Connect the AMS101 evaluation card to the XADC header on the base board.

   The AMS101 evaluation card connects to the FPGA base board by plugging the card into the XADC header on the base board. The AMS101 evaluation card connector and XADC header socket are keyed to align properly. Pin 1 on the XADC header needs to connect to pin 1 of the 20-pin connector on the AMS101 evaluation card. Figure 2-4 shows this connection.

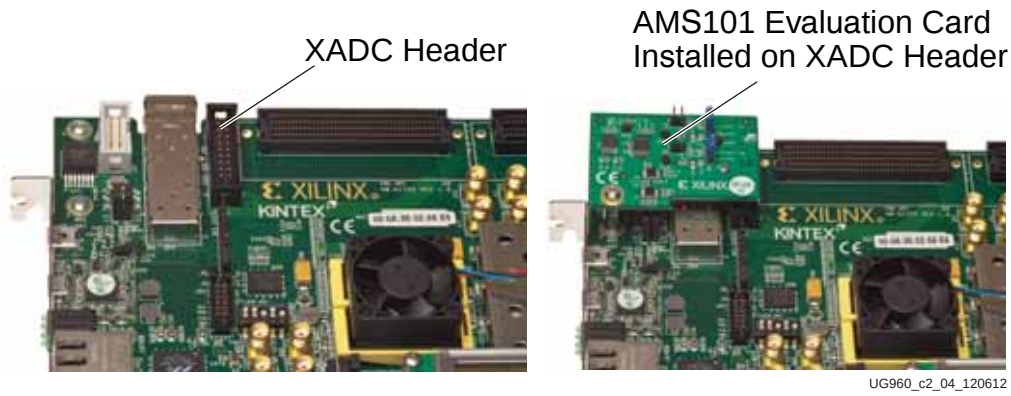


Figure 2-4: **AMS101 Evaluation Card Installed on the Base Board XADC Header**

Ensure that all the jumper settings are correct on the AMS101 evaluation card. Figure 2-5 shows an example of jumpers J3 and J5 (DACs enabled). Table 2-2 explains additional jumpers.

Note: The image in Figure 2-5 is for reference only and might not reflect the current revision of the board.

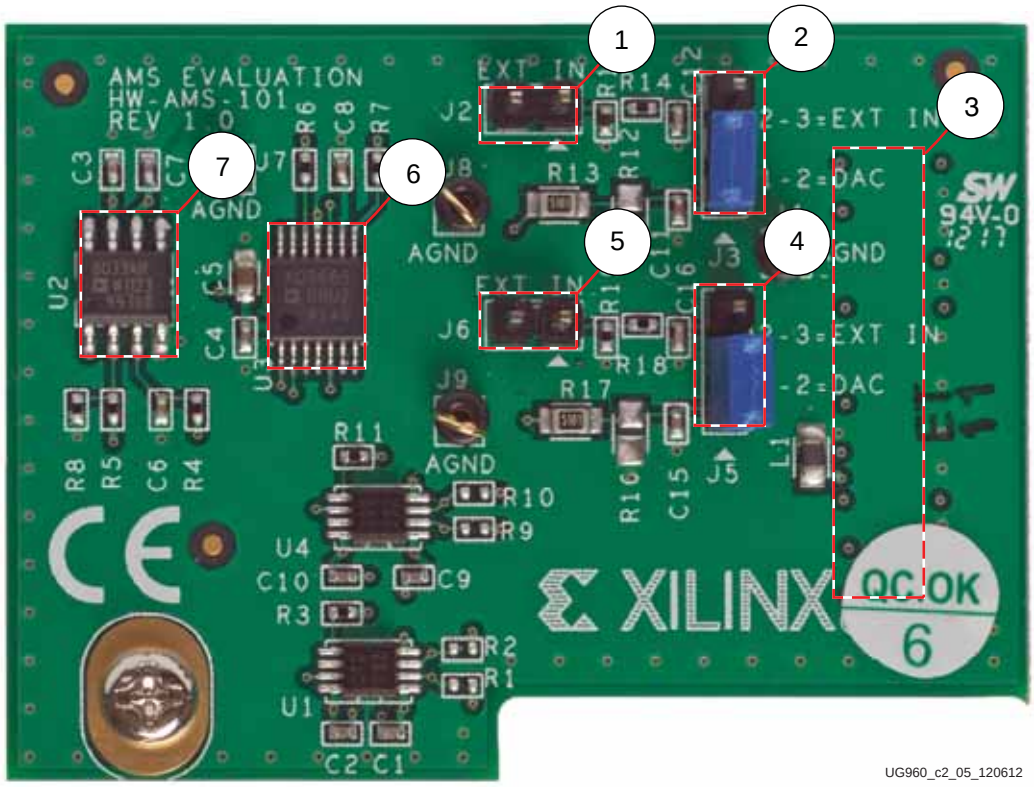
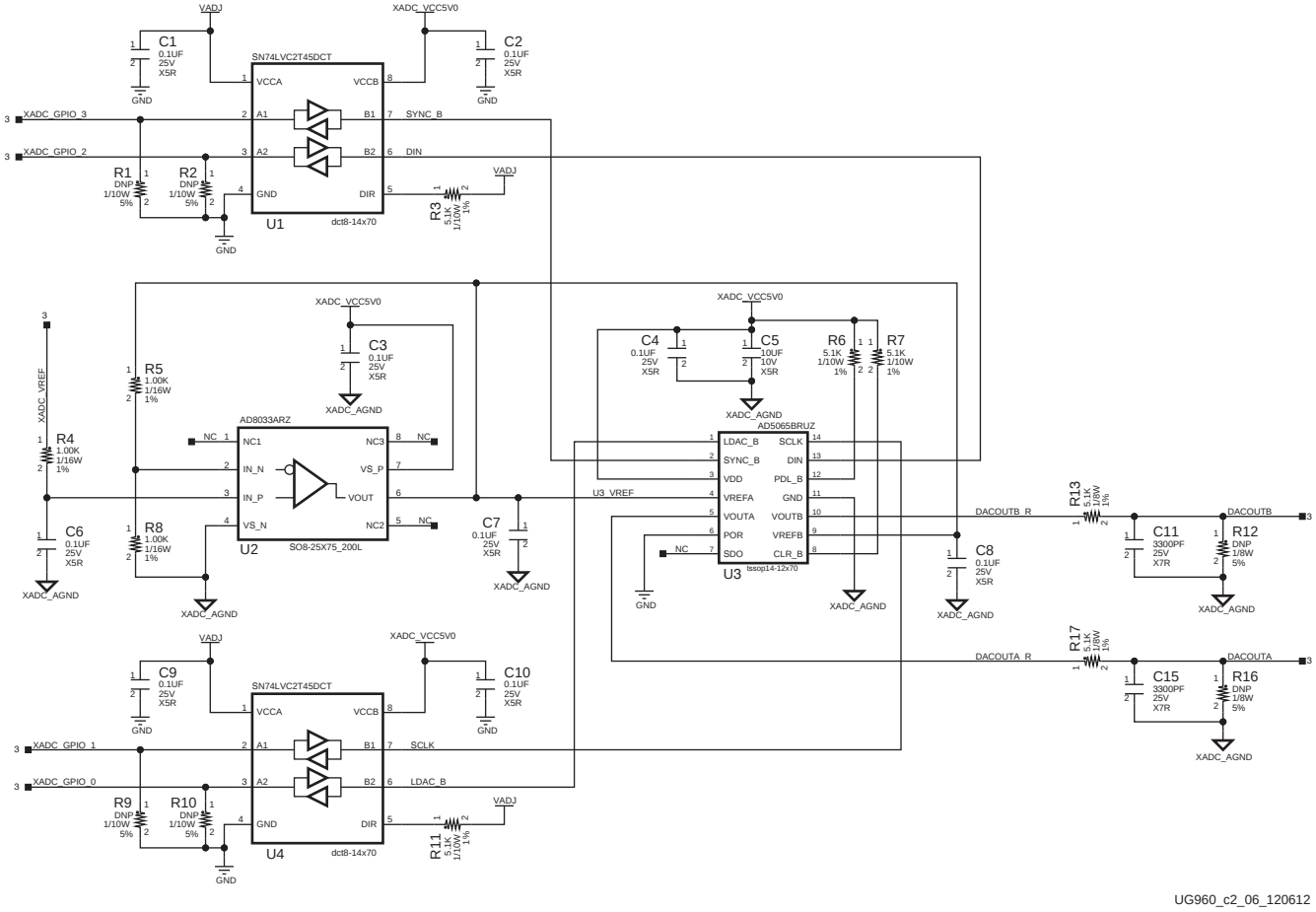


Figure 2-5: **AMS101 Evaluation Card Jumper Configuration**

Table 2-2: **AMS101 Evaluation Card Jumper Configuration Notes**

Callout	Reference Designator	Component Description	Notes	Schematics
1	J2	Jumper	External signal source to V_P positive analog input.	Figure 2-7
2	J3	Jumper	1–2 selects DAC signal source. 2–3 selects external input source on J2.	Figure 2-7
3		Connector	20-pin connector to XADC header on FPGA/Zynq-7000 AP SoC base board.	Figure 2-7
4	J5	Jumper	1–2 selects DAC signal source. 2–3 selects external input source on J6.	Figure 2-7
5	J6	Jumper	External signal source to V_N negative analog input.	Figure 2-7
6		DAC	16-bit DAC sets analog test voltage.	Figure 2-6
7		Amplifier	Reference buffer for DAC.	Figure 2-6

Schematics for the AMS101 evaluation card are shown in Figure 2-6 and Figure 2-7.



UG960_c2_06_120612

Figure 2-6: **AMS101 Evaluation Card Schematic (1 of 2)**

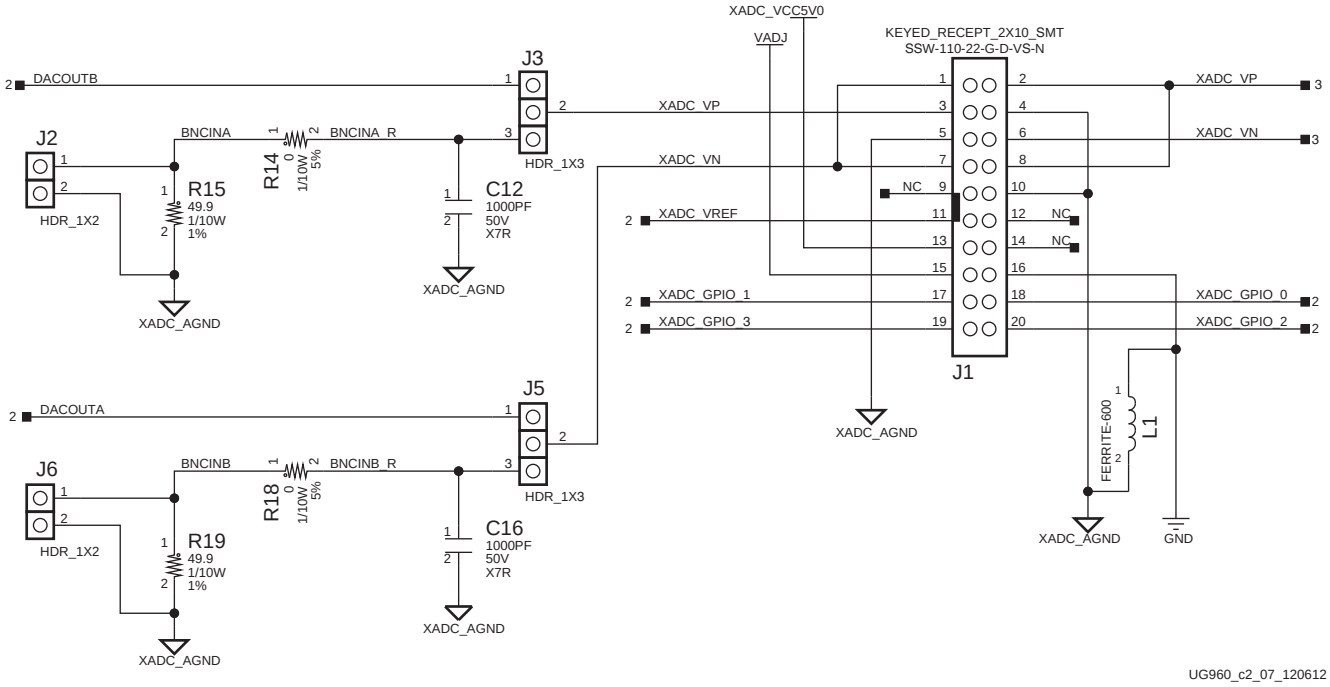


Figure 2-7: **AMS101 Evaluation Card Schematic (2 of 2)**

6. Power up the FPGA base board.

   The power switch can now be put in the ON position (switch toward the power plug). Figure 2-8 shows the location of the power switch. It also shows the LEDs illuminated on the FPGA base board. This should occur directly after the FPGA base board switch is flipped into the ON position. At this stage, hardware connection is complete.

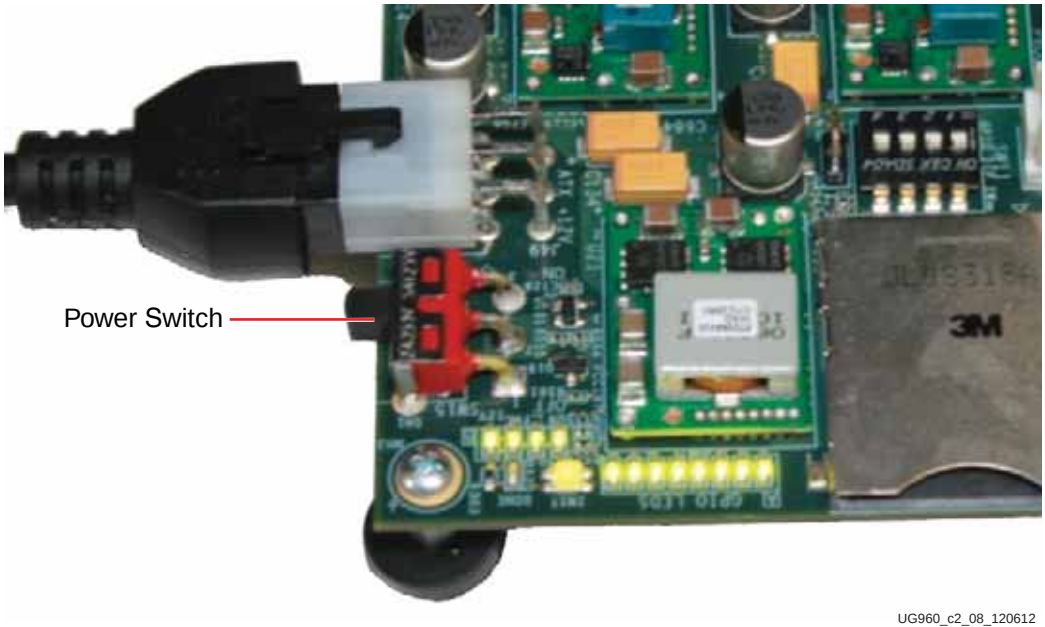


Figure 2-8: **Turning On the FPGA Base Board Power**

7. Download the design to the FPGA.

   For the AMS101 evaluation card to function, the FPGA needs to be programmed with the appropriate design. To do this, download the design to the FPGA.

   For KC705, VC707, and AC701 base boards use these steps:

   a. Open Vivado® Design Suite. Here is one example path for Vivado: `Start menu/Xilinx Design Tools/Vivado 2013.3/Vivado 2013.3`
   b. Create a Vivado Project.
   c. Open a Hardware Session.
   d. Open a new Hardware Target and run through the wizard.
   e. Open `xadc_eval_design.bit` from the `xadc_eval_design_XXABC/ready_to_test` directory.

   For the ZC702 base board, use the following steps:

   a. Copy `xadc_eval_design/ready_to_test/BOOT.bin` to the SD card.
   b. Set the third and fourth positions of SW16 to the ON state to boot from the SD card.
   c. Set the jumper positions as described in Table 2-1.
   d. Power up the board.

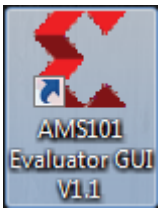
The LEDs on the FPGA base board should light up as the design is downloading. Figure 2-9 shows an example of the LEDs lit up after the KC705 board is programmed.

Figure 2-9: **LEDs after Programming the FPGA with the Design**

8. Run the AMS101 evaluator LabVIEW GUI executable file.

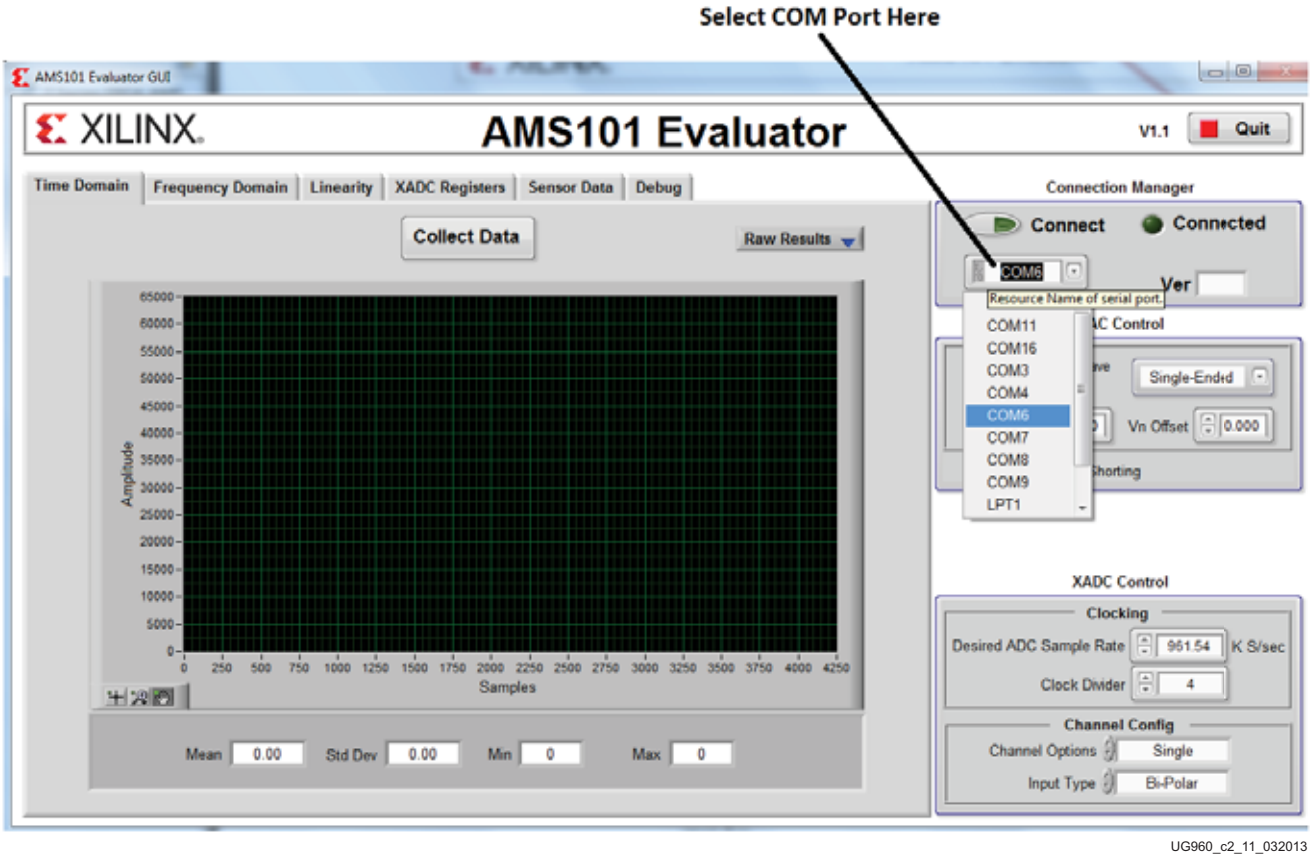
   If the AMS101 evaluator tool GUI was successfully installed, an icon should be displayed on the desktop and in the Windows start menu (see Figure 2-10). If the LabVIEW run-time engine has been downloaded from the National Instruments website, load the `AMS101 Evaluation GUI V1.1.exe` file. To open the AMS101 evaluator tool GUI, double-click the icon or run the `.exe` file. The GUI shown in Figure 2-11 should appear.

   Note: Do not press anything on the GUI until step 9 is performed.



UG960_c2_10_032013

Figure 2-10: **AMS Icon**



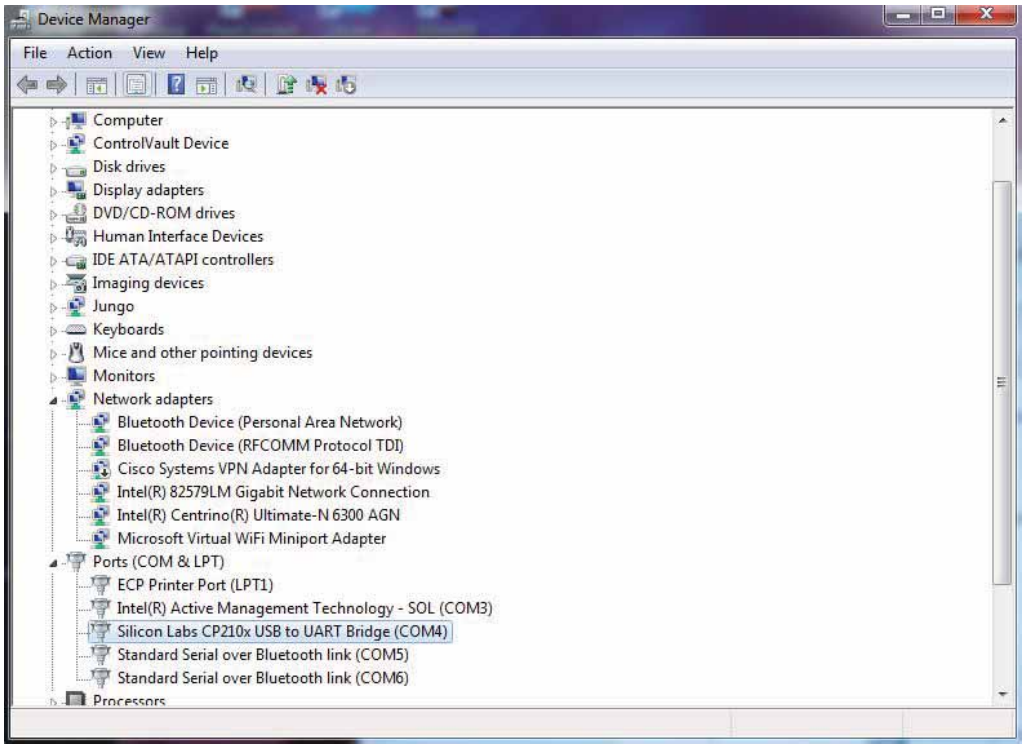
UG960_c2_11_032013

Figure 2-11: **AMS101 Evaluator Tool on Start-Up**

9. Connect to the UART port as detailed in the appropriate FPGA/processor base board guide:
   - *Kintex-7 FPGA KC705 Evaluation Kit Getting Started Guide (Vivado Design Suite)* (UG883) [Ref 6]
   - *Getting Started with the Virtex-7 FPGA VC707 Evaluation Kit (UG848)* [Ref 7]
   - *Zynq-7000 All Programmable SoC: ZC702 Evaluation Kit and Video and Imaging Kit Getting Started Guide (Vivado Design Suite) (UG926)* [Ref 8]
   - *Artix-7 FPGA AC701 Evaluation Kit Getting Started Guide (Vivado Design Suite) (UG967)* [Ref 9]

Set the USB-UART connection to a known port in the Device Manager as follows:

- Right-click **My Computer** and select **Properties**.
- Select **Hardware**. Click **Device Manager**.
- Find and right-click the Silicon Labs device in the list. Then select **Properties**.
- Click **Port Settings** and then **Advanced...**.
- Select an open COM port between COM1 and COM4. (See Figure 2-12.)



UG960_c2_12_120612

Figure 2-12: **UART-USB Port in Device Manager**

10. Select the appropriate COM port from the pull-down menu on the GUI as show in Figure 2-13. Then click **Connect**. The **Connected** circle to the right should turn green. If the AMS101 Evaluator tool is unable to connect, be sure the correct COM port is selected and click refresh.

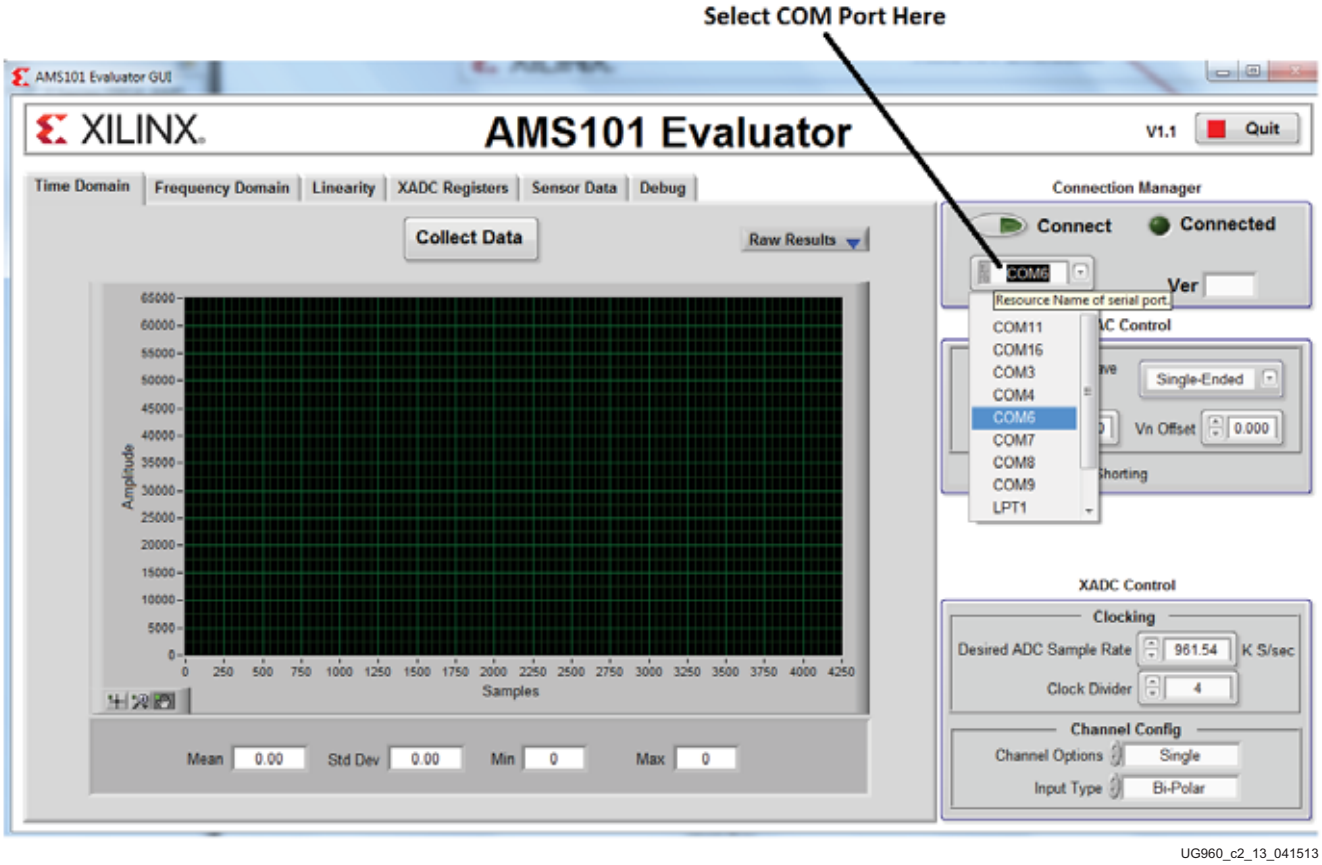


Figure 2-13: **AMS101 Evaluator Tool COM Port Selection**

Requirements

This section lists the prerequisites for testing this design.

Test Setup Requirements

Hardware

- A hardware board with AMS card, any of:
  - KC705 Revision 1.2
  - VC707 Revision 1.2
  - AC701 Revision 1.1
  - ZC702 Revision 1.2
- USB-UART cable
- Download cable for FPGA programming
- External signal generator (optional)

Software

Install the LabVIEW Run Time Engine (Run Time Engine Version 2011) [Ref 1].

Rebuilding the Hardware Design

You can rebuild the hardware design using the Vivado design tool or XPS flow. This section lists steps required to rebuild the hardware design using the Vivado flow.

1. Browse to `hardware/vivado/scripts` and run **`vivado -source xadc_eval_design_gui`**
2. Click **`Generate Bitstream`** to implement the design as shown in Figure 2-14.

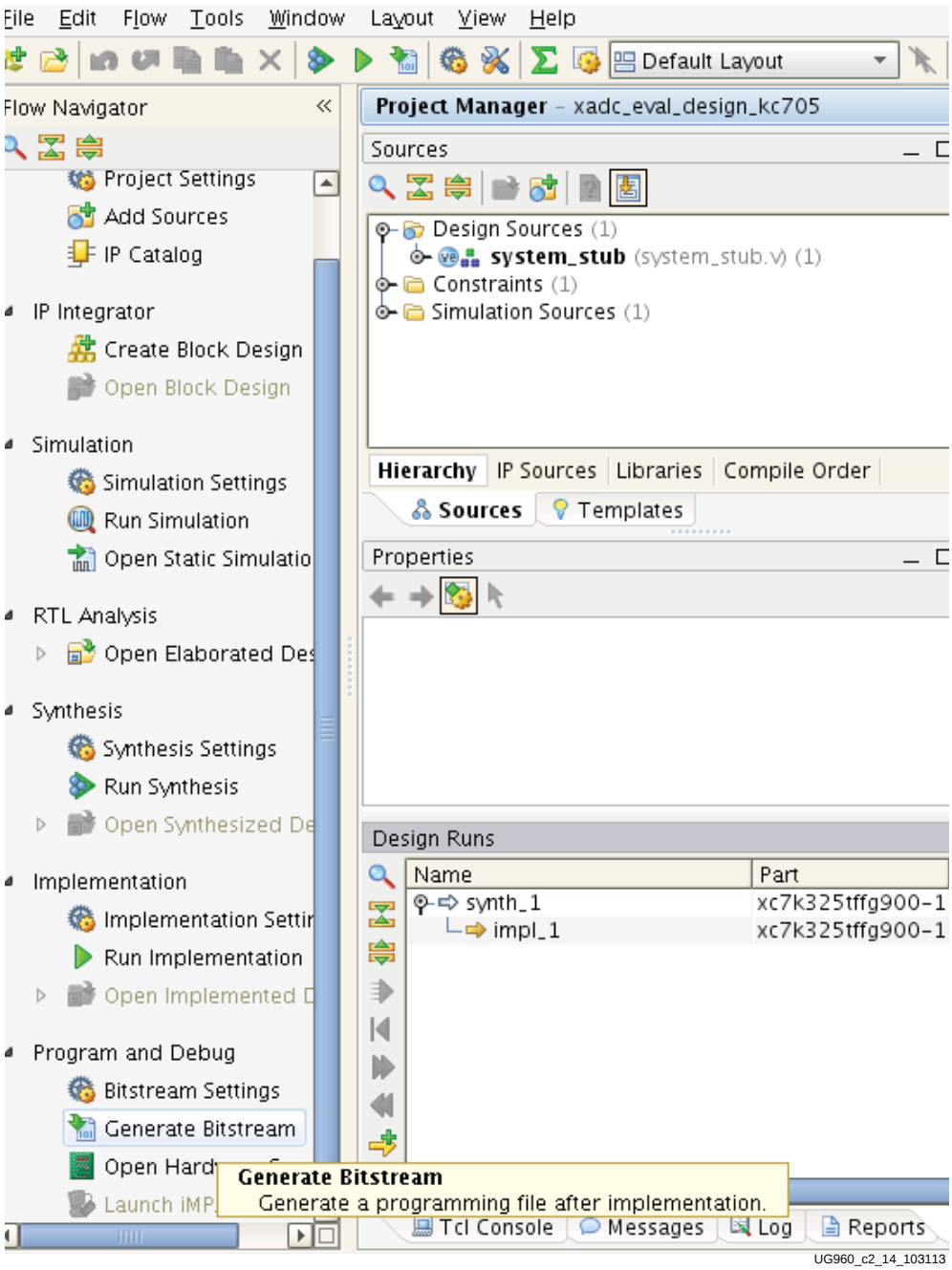


Figure 2-14: **Selection of Export Design**

3. Click **`Open Implemented Design`**. Go to **`File`** and select **`Export Design`** as shown in Figure 2-15.

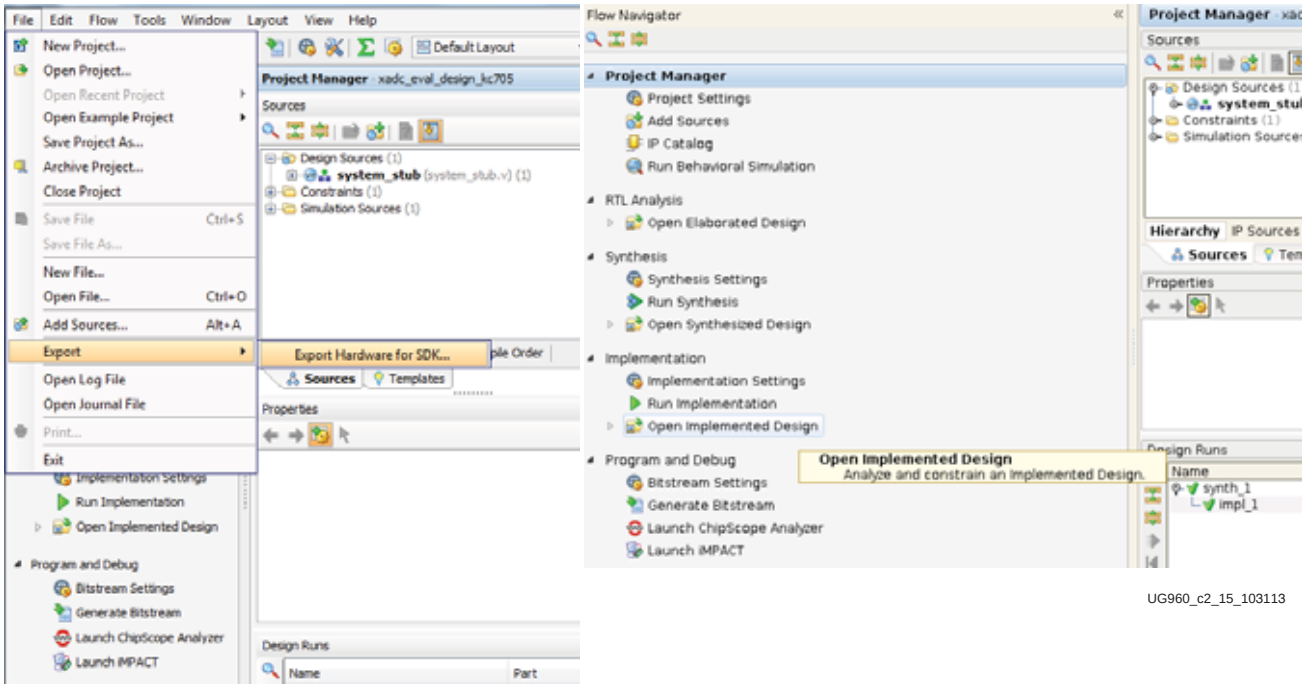


Figure 2-15: **Export Design**

4. Click **`OK`** as shown in Figure 2-16.

Figure 2-16: **Exporting Hardware Platform to SDK**

This step re-implements the design and copies the `bitstream`, `BMM`, and `system.xmp` file in the `SDK/SDK_Export` folder.

Rebuilding the Hardware Design for ZC702

You can rebuild the hardware design for ZC702 using the Vivado tool or XPS flow. This section lists steps required to rebuild the hardware design using the Vivado tool flow.

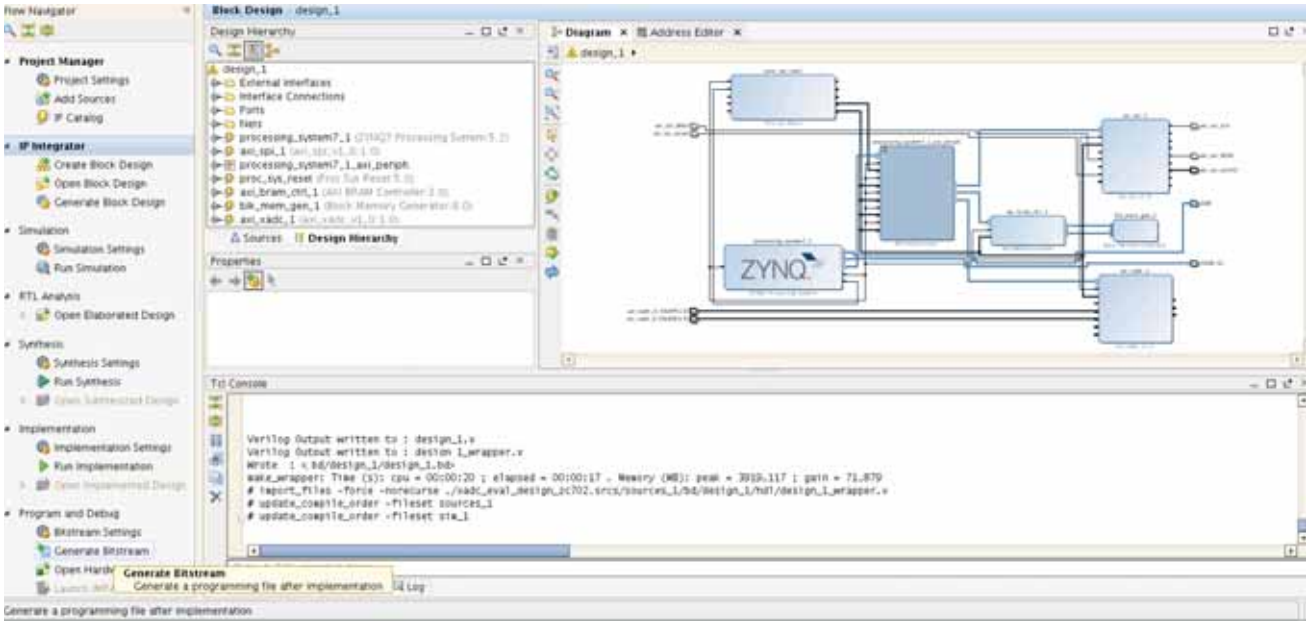
To rebuild the hardware design:

1. Start the Vivado tool in GUI mode by typing
   `vivado -source xadc_eval_design_gui`
   in the Linux command prompt. Figure 2-17 appears.

UG960_c2_17_103113

Figure 2-17: **IP Integrator in Vivado GUI Mode**

2. Click **`Generate Bitstream`** to implement the design as shown in Figure 2-18.



UG960_c2_18_103113

Figure 2-18: **Generating Bitstream**

3. A dialog box appears. Click **Yes** to launch synthesis and implementation as shown in Figure 2-19.

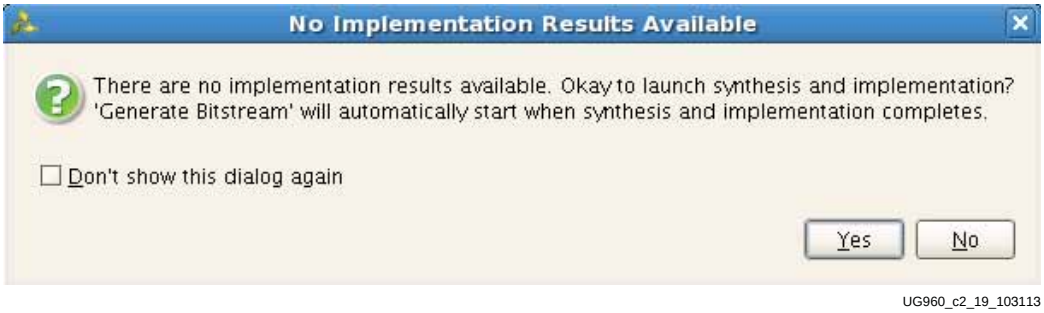


Figure 2-19: **Launch Synthesis and Implementation**

4. Click **Open Implemented Design** as shown in Figure 2-20.

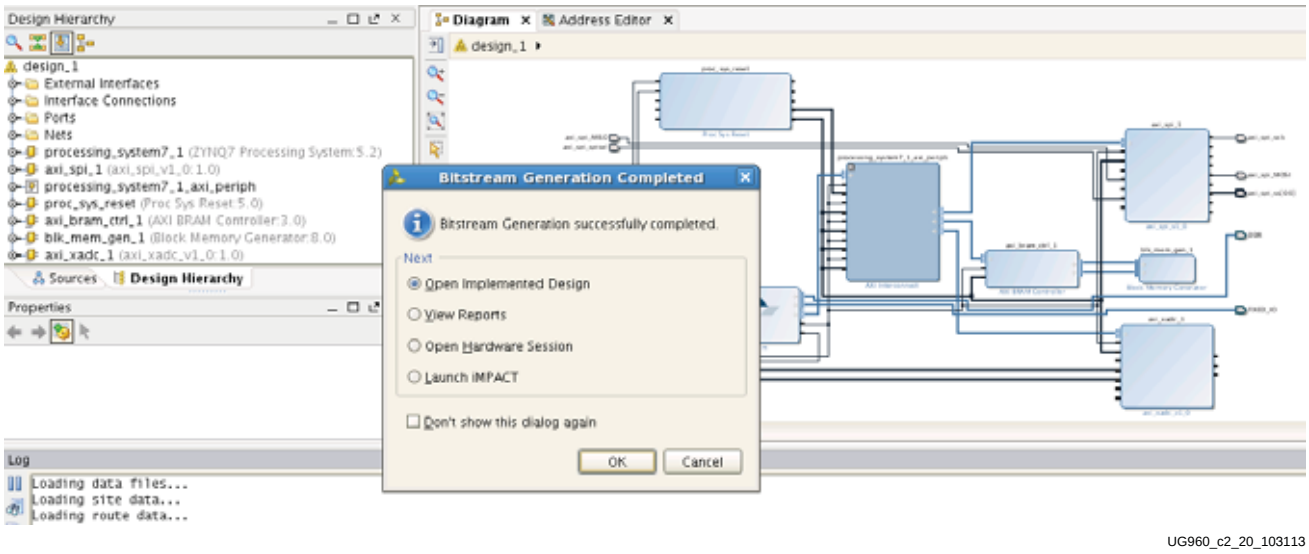


Figure 2-20: **Open Implemented Design**

5. Go to **File** and select **Export Design** as shown in Figure 2-21.

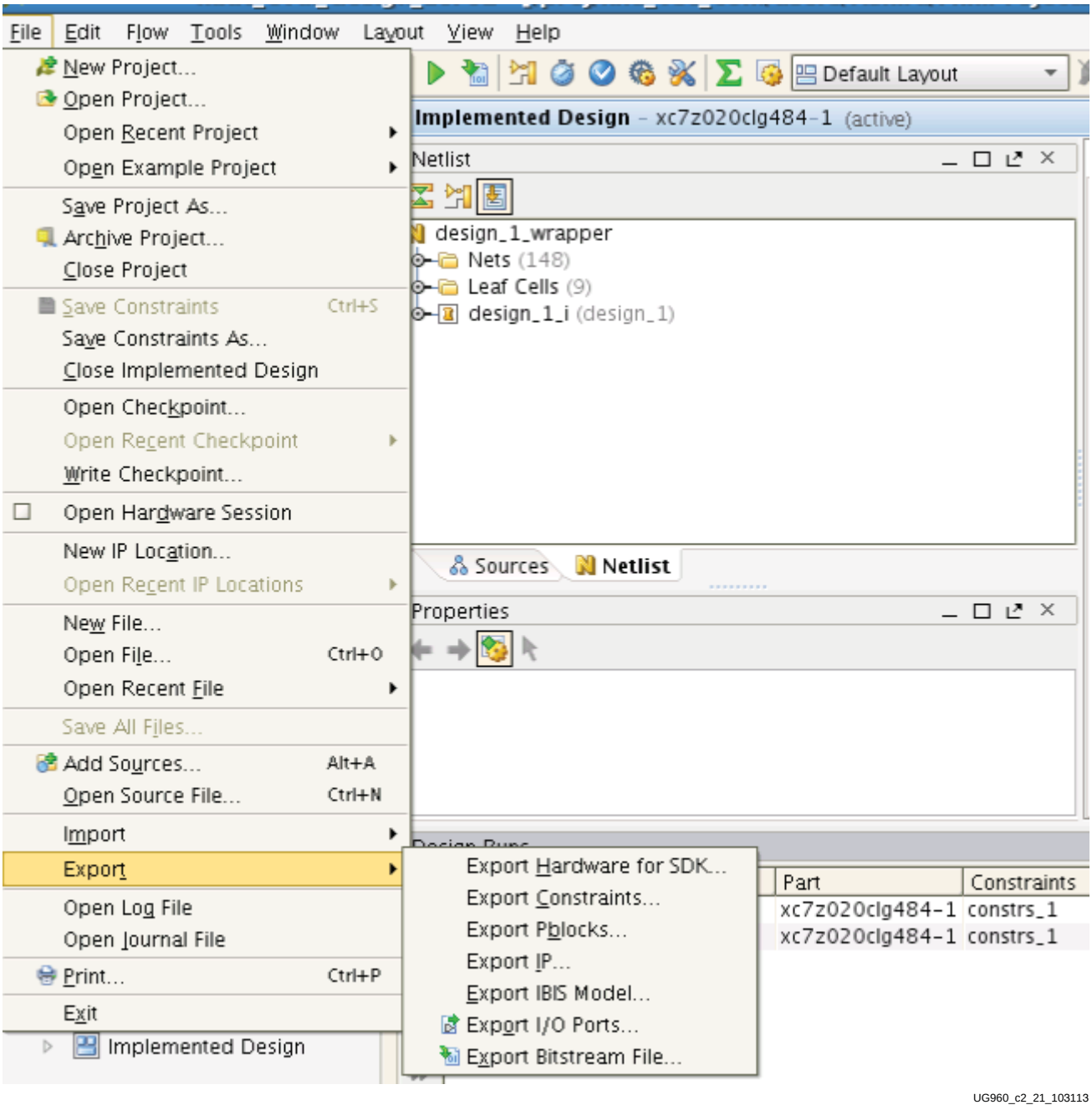


Figure 2-21: **Export Hardware for SDK**

Figure 2-22: **Exporting Hardware Platform to SDK**

6. Click **OK** as shown in Figure 2-22.

This step re-implements the design and copies the `bitstream`, `BMM`, and `system.xmp` file in the `SDK/SDK_Export` folder.

Rebuilding the Software Design

You can rebuild the software design using the Xilinx SDK, which ships with the Vivado Design Suite build. Use these steps to build the design using the SDK:

1. Copy the AMS reference design `.zip` file in the host PC. Unzip the design file.
2. Copy `SDK_Export` from the `xadc_eval_design_kc705.sdk`, `xadc_eval_design_vc707.sdk`, `xadc_eval_design_ac701.sdk`, and `xadc_eval_design_zc702.sdk` area in the `vivado_proj` folder to the folder where the SDK project needs to be created.
3. Open the Xilinx SDK from the Windows environment by clicking **Start > All Programs > Xilinx Design Tools > SDK 14.5 > Xilinx Software Development Kit**. Select work space as the location where the SDK_Export design is copied (Figure 2-23).

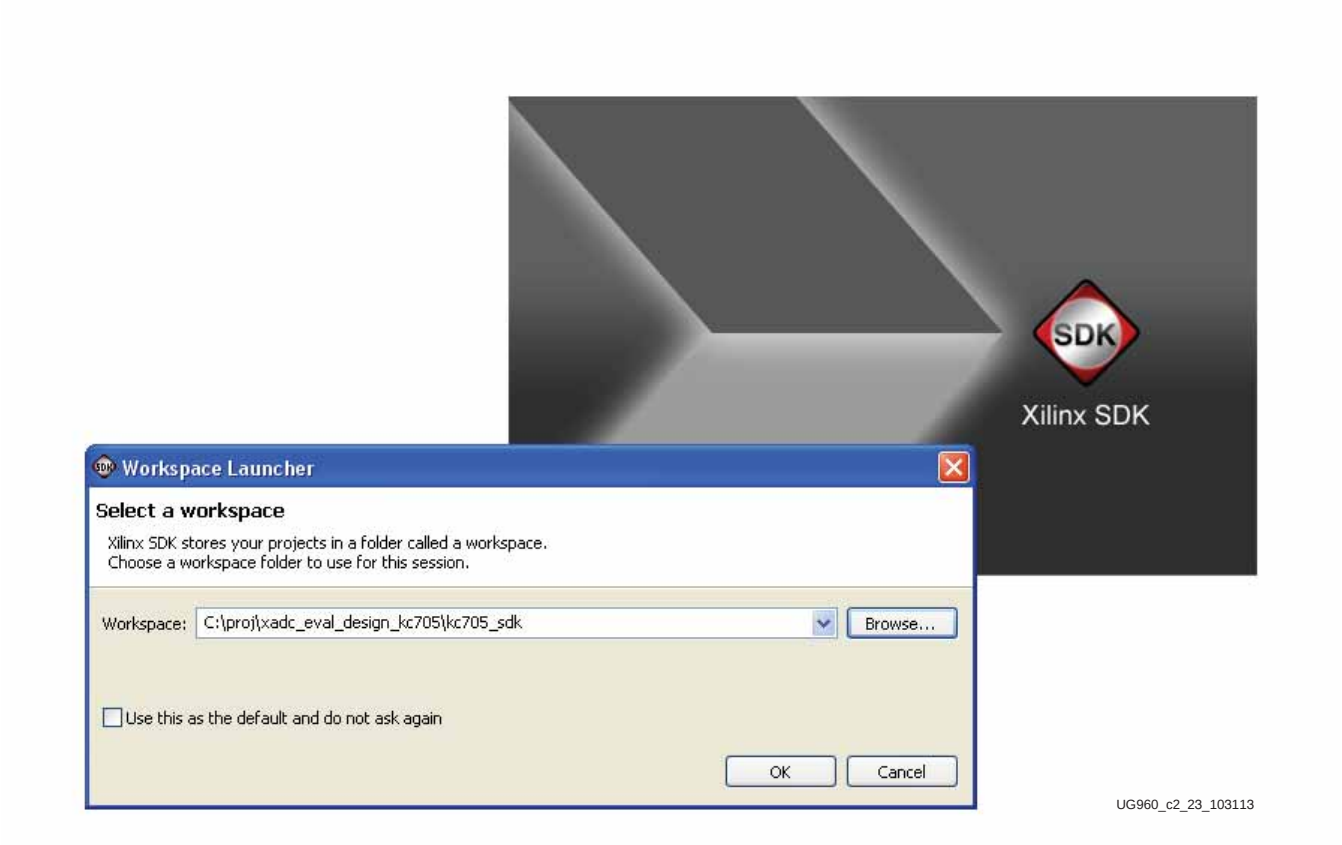
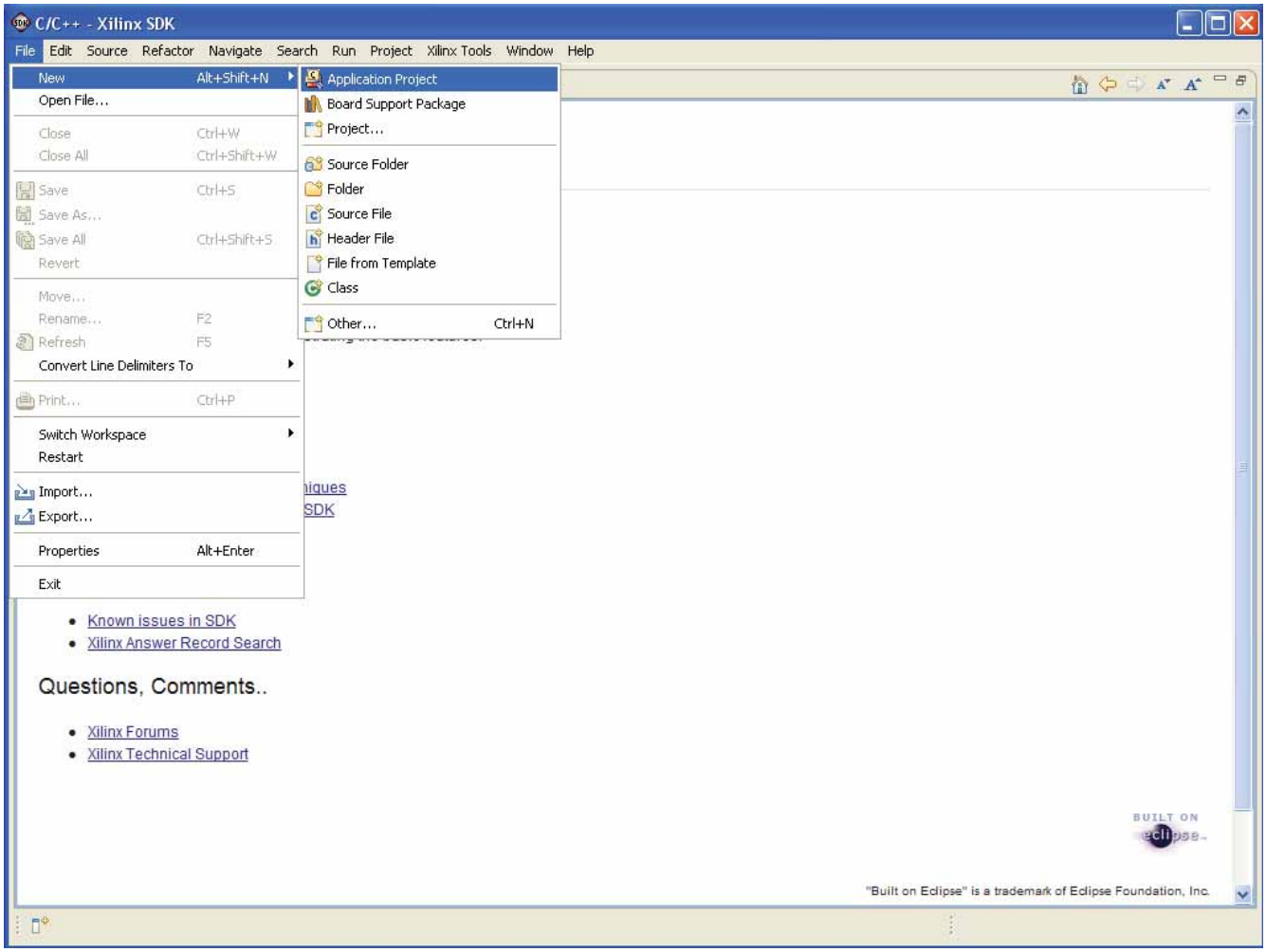


Figure 2-23: **Selection of Work Space in SDK**

4. Click **`File > New > Application Project`** as shown in Figure 2-24.



UG960_c2_24_103113

Figure 2-24: **Creating a New Application Project**

5. Select **`Create new`** under `Hardware Platform` as shown in Figure 2-25.

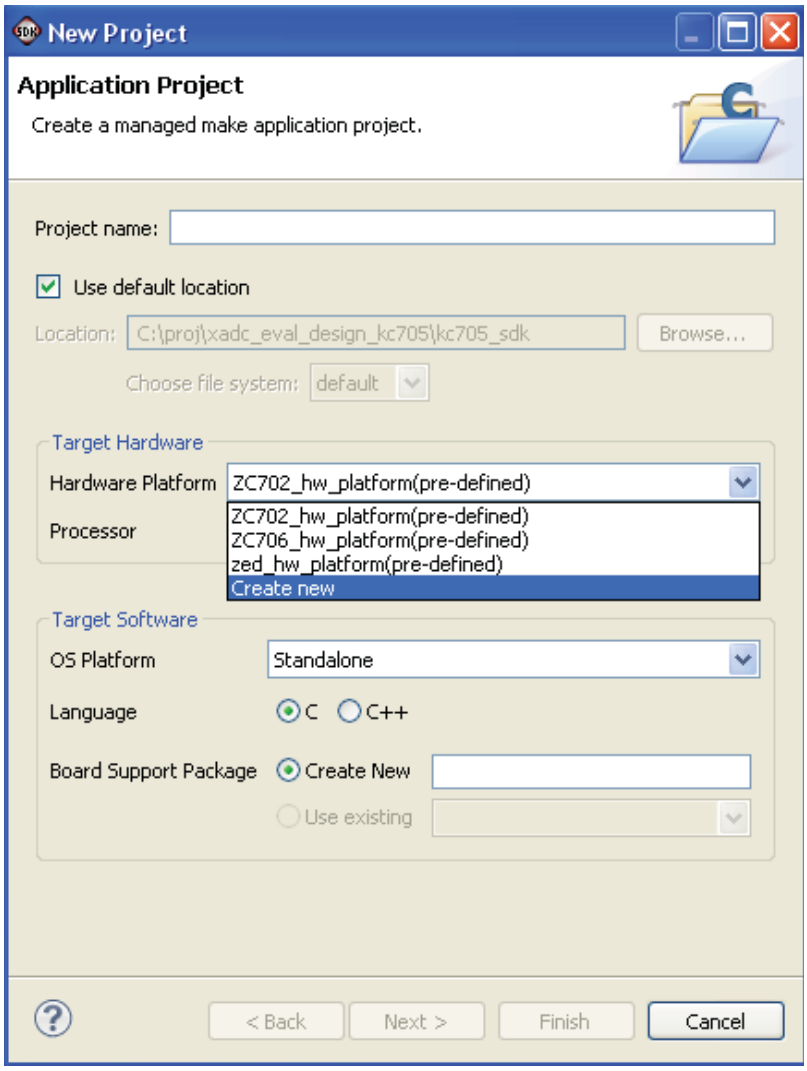


Figure 2-25: **Selecting Create New Hardware Platform**

6. Create **`New Hardware Project`**. Browse to the target hardware specification (see Figure 2-26).

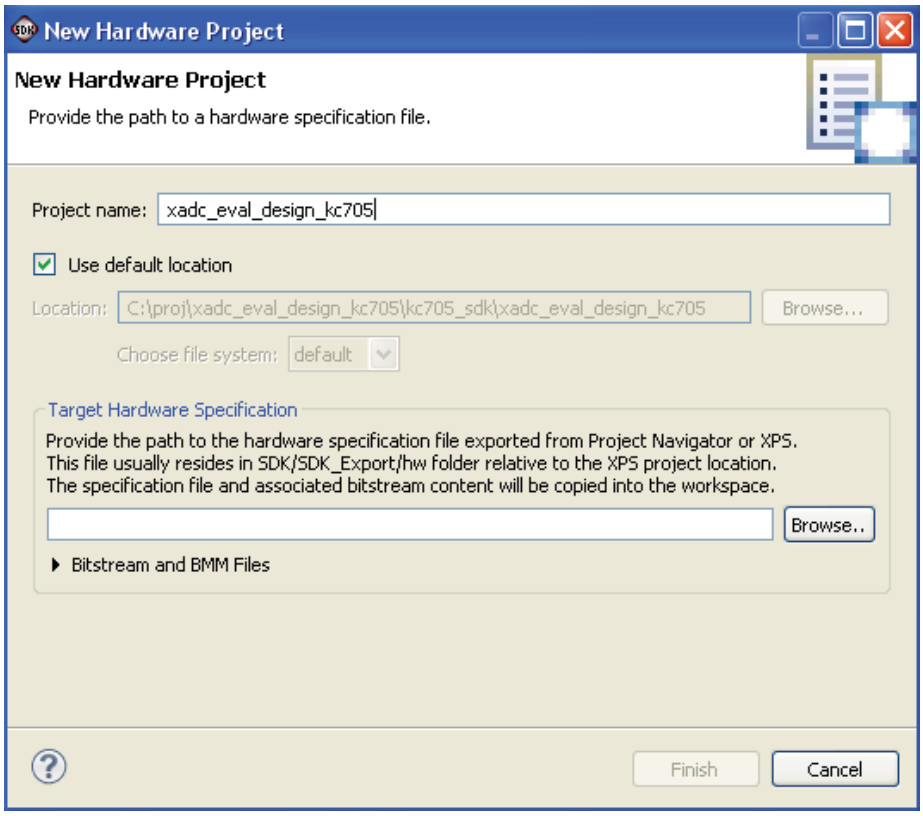
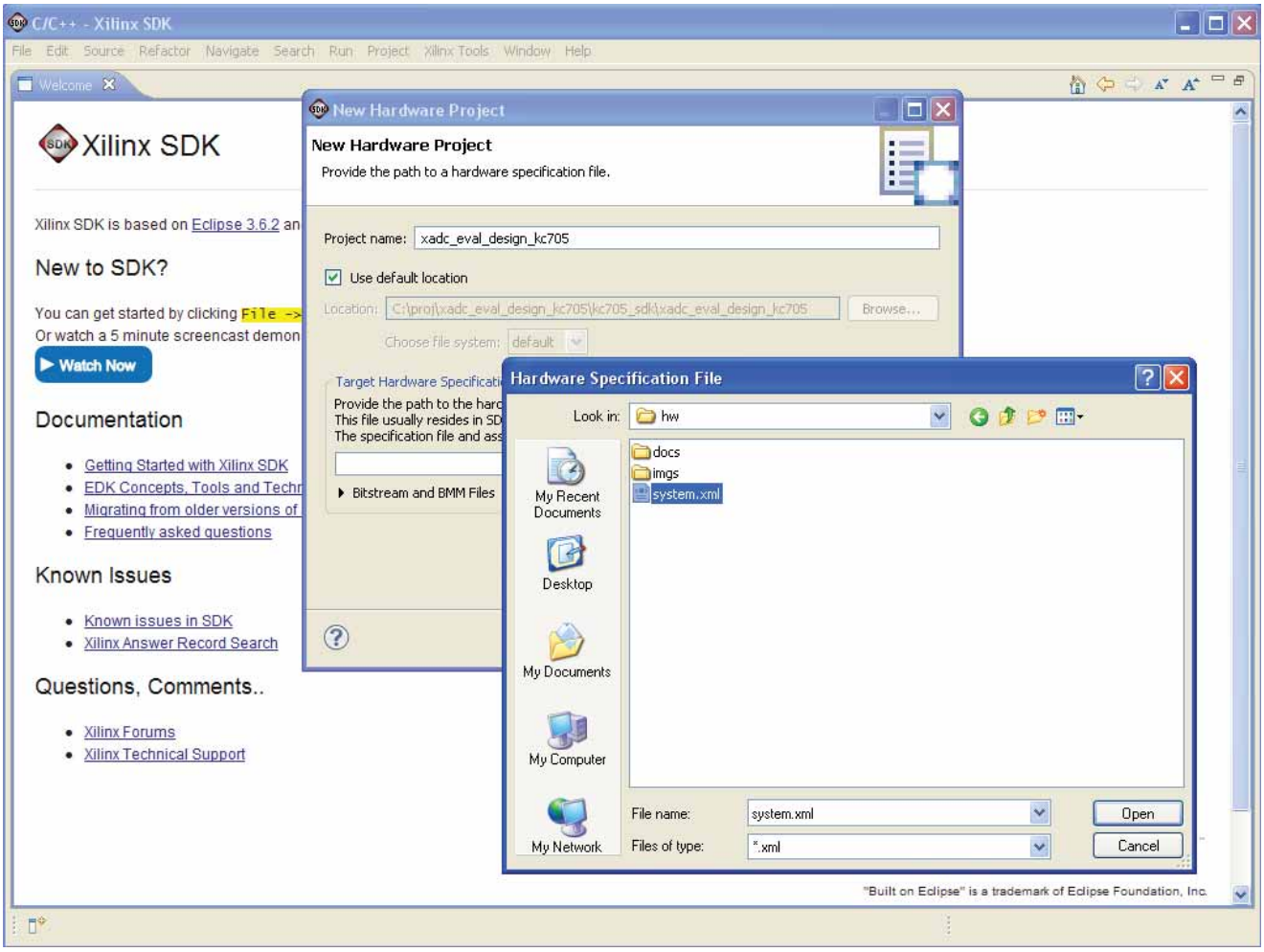


Figure 2-26: **Selecting Hardware Platform**

7. Select **`system.xml`** from the SDK_Export/hw area as shown in Figure 2-27.



UG960_c2_27_103113

Figure 2-27: **Selecting system.xml File**

8. Click **`Finish`**. (See Figure 2-28.)

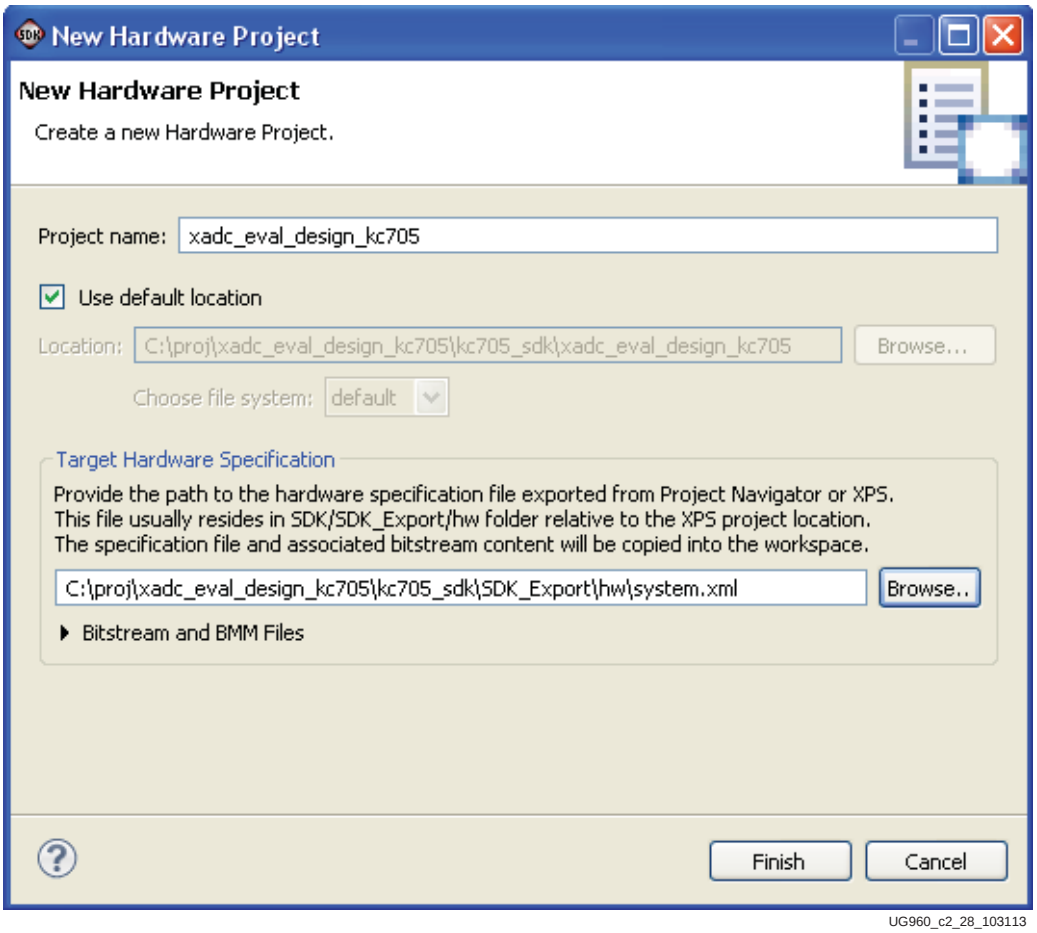


Figure 2-28: **Creating a New Hardware Platform**

9. Create an Application Project. Click **Next**. See Figure 2-29.

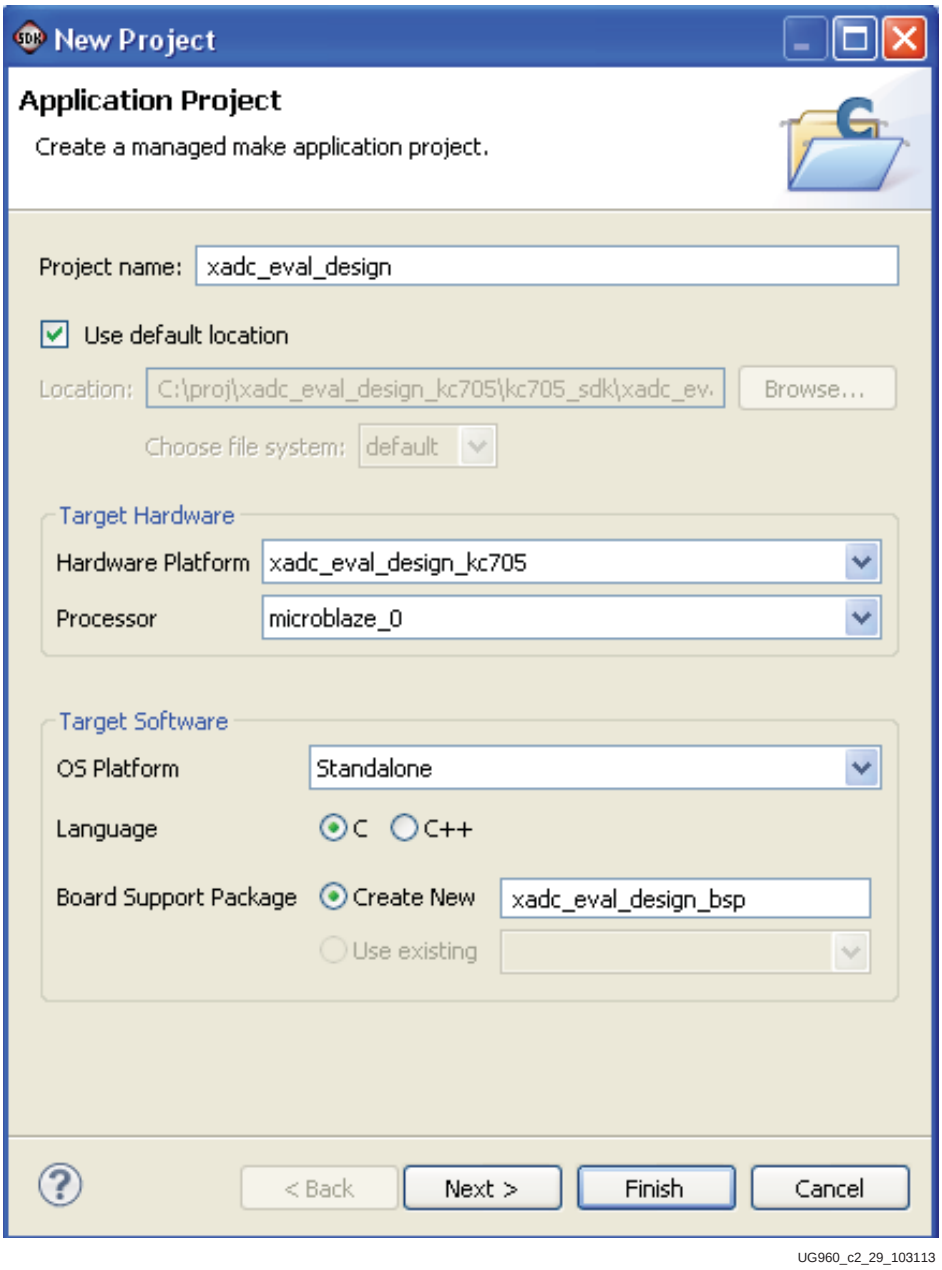


Figure 2-29: **Creating a New Application Project**

10. Select **`Empty Application`**. Click **`Finish`**. See Figure 2-30.

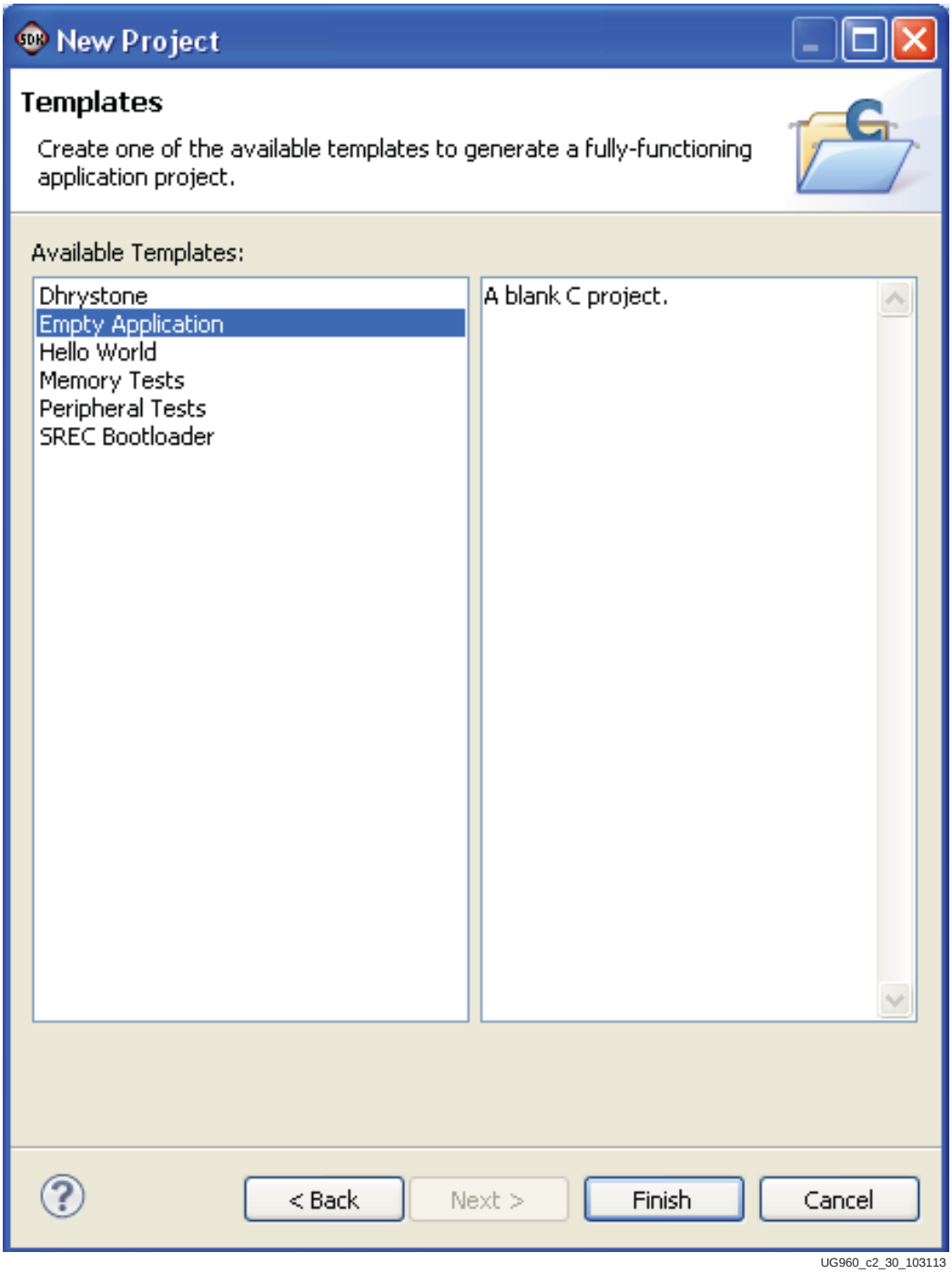


Figure 2-30: **Creating Empty Application**

11. Copy the `xadc_eval_design/src` folder from the unzipped reference design folder and replace the newly created `xadc_eval_design/src` folder. Xilinx SDK automatically compiles the files.
12. Right-click **`xadc_eval_design`**. Select **`C/C++ Build Settings`** as shown in Figure 2-31.

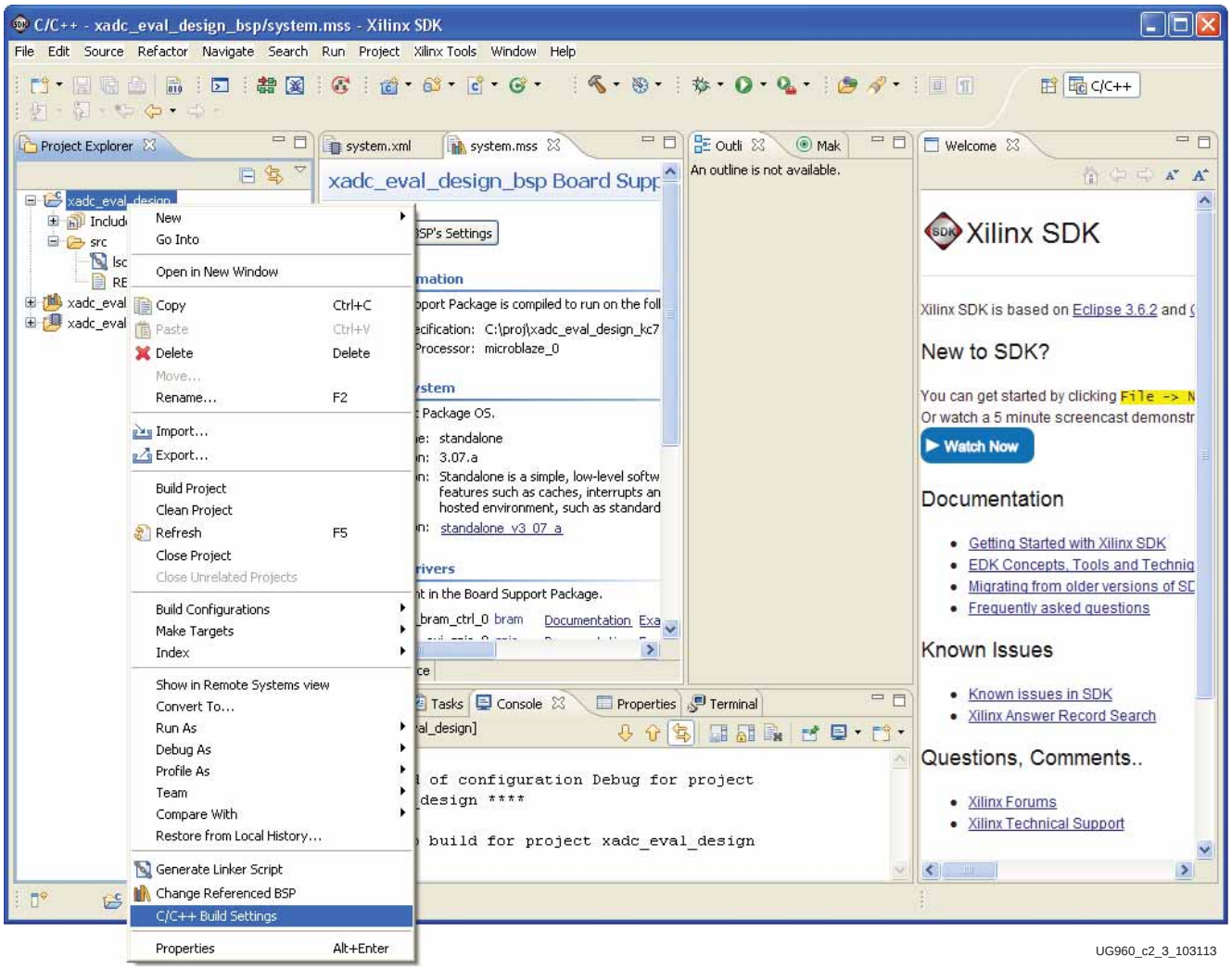
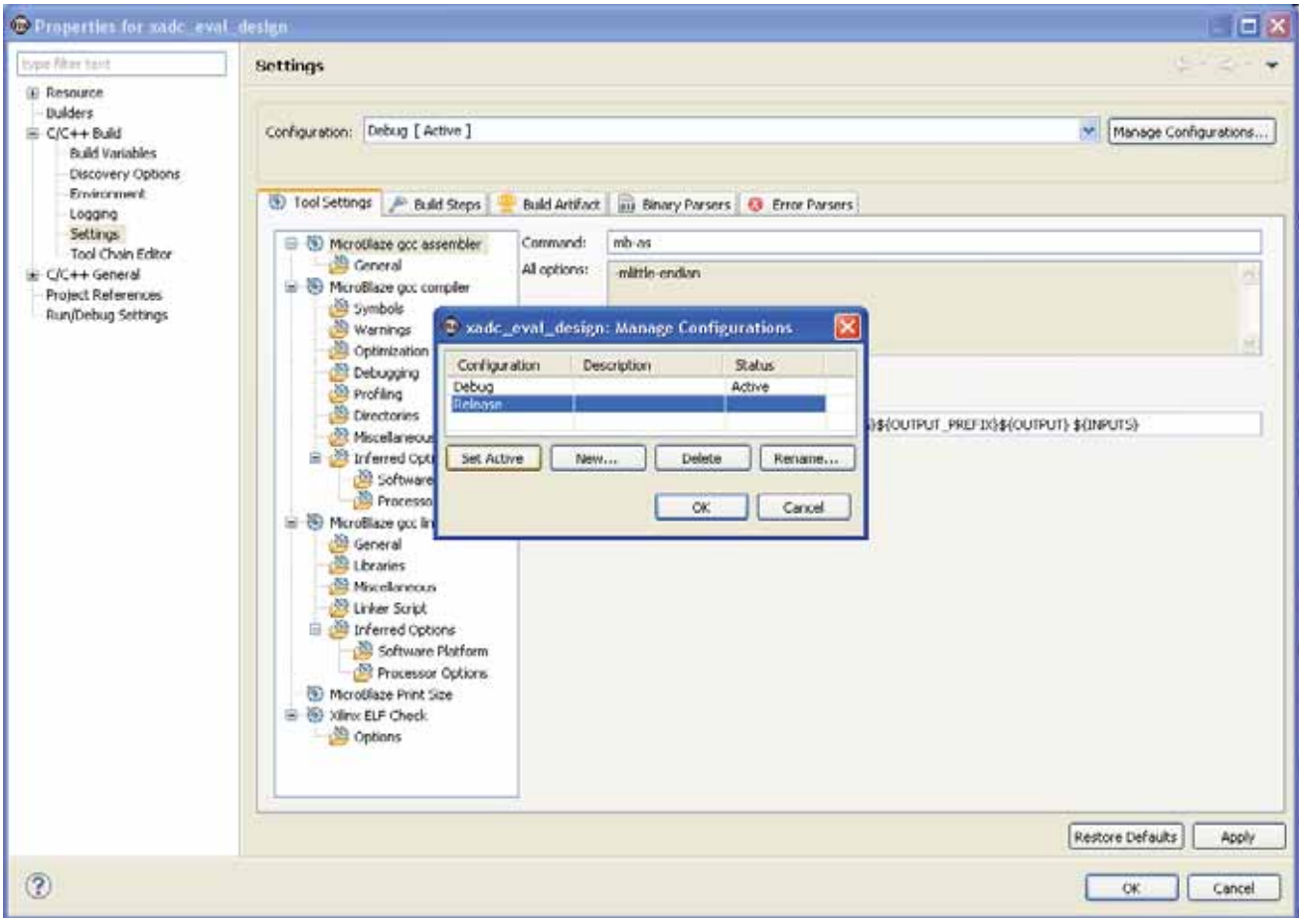


Figure 2-31: **Selecting C/C++ Build Settings**

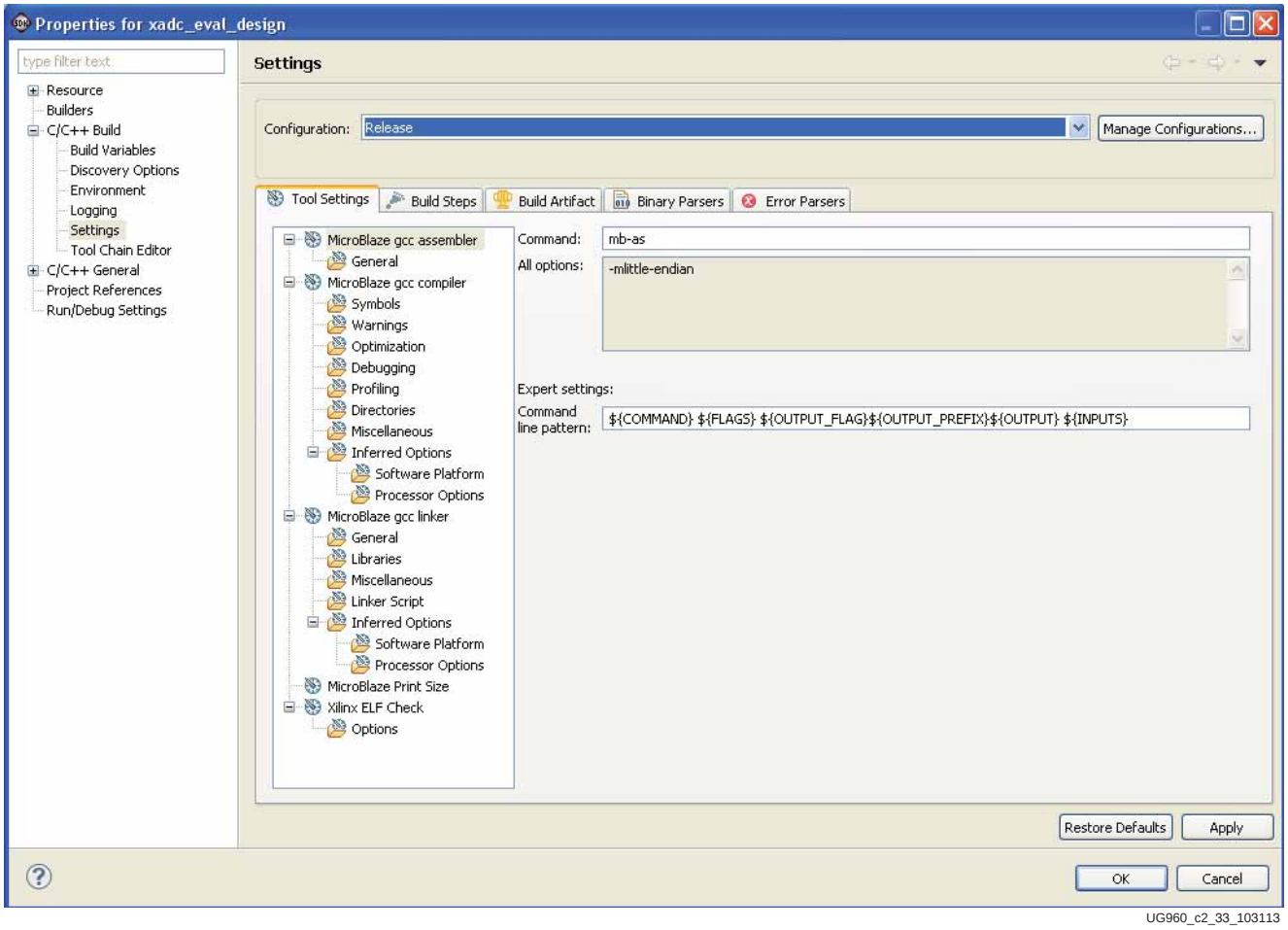
13. Click **`Manage Configurations`** and select **`Release`**. Click **`Set Active`**. See Figure 2-32.



UG960_c2_32_103113

Figure 2-32: **Selecting Release Build**

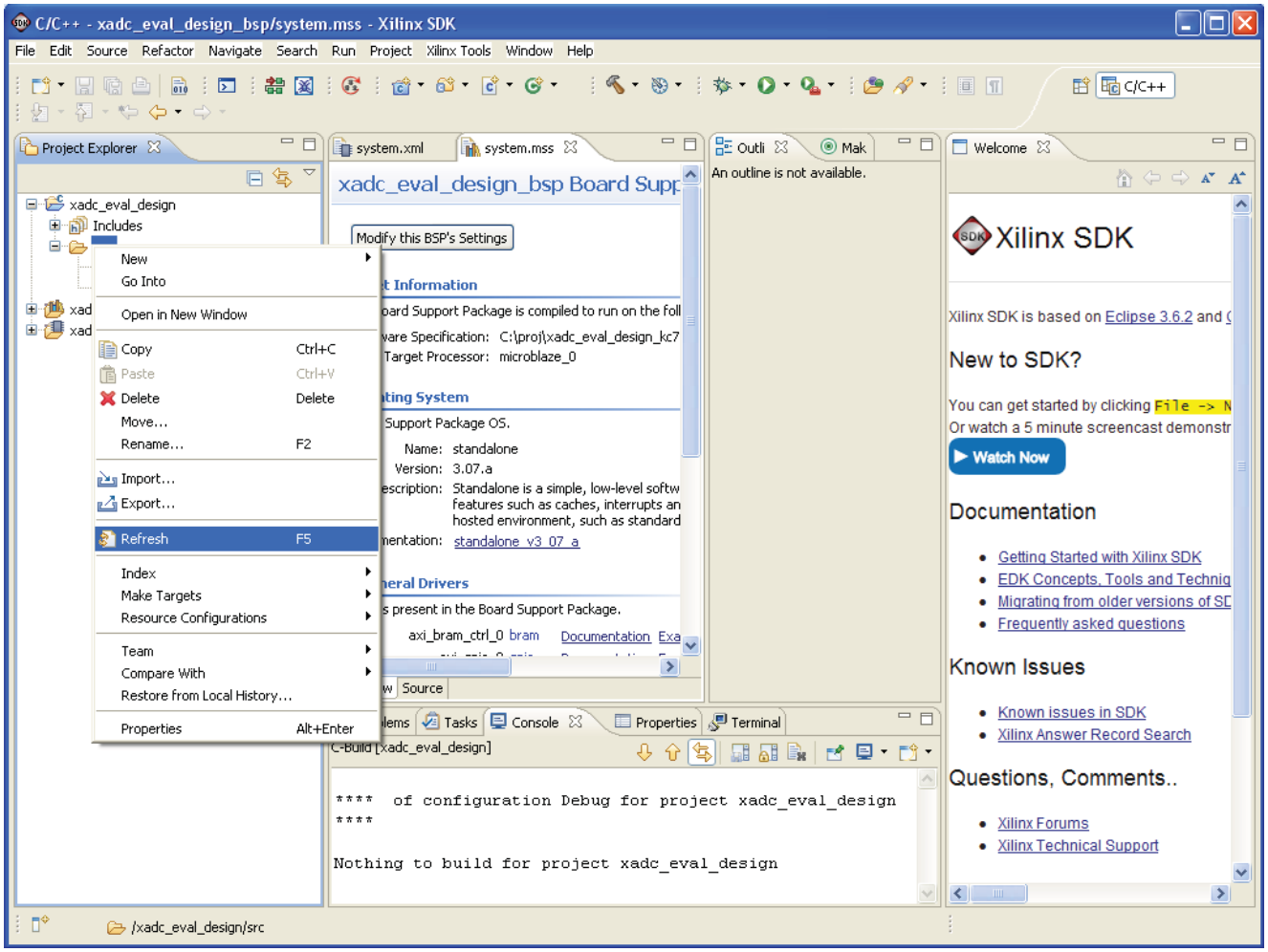
14. Select **`Release as Configuration`** as shown in Figure 2-33.



UG960_c2_33_103113

Figure 2-33: **Selecting Release as Configuration**

15. Select **xadc_eval_design > src** and right-click to select **Refresh** as shown in Figure 2-34.



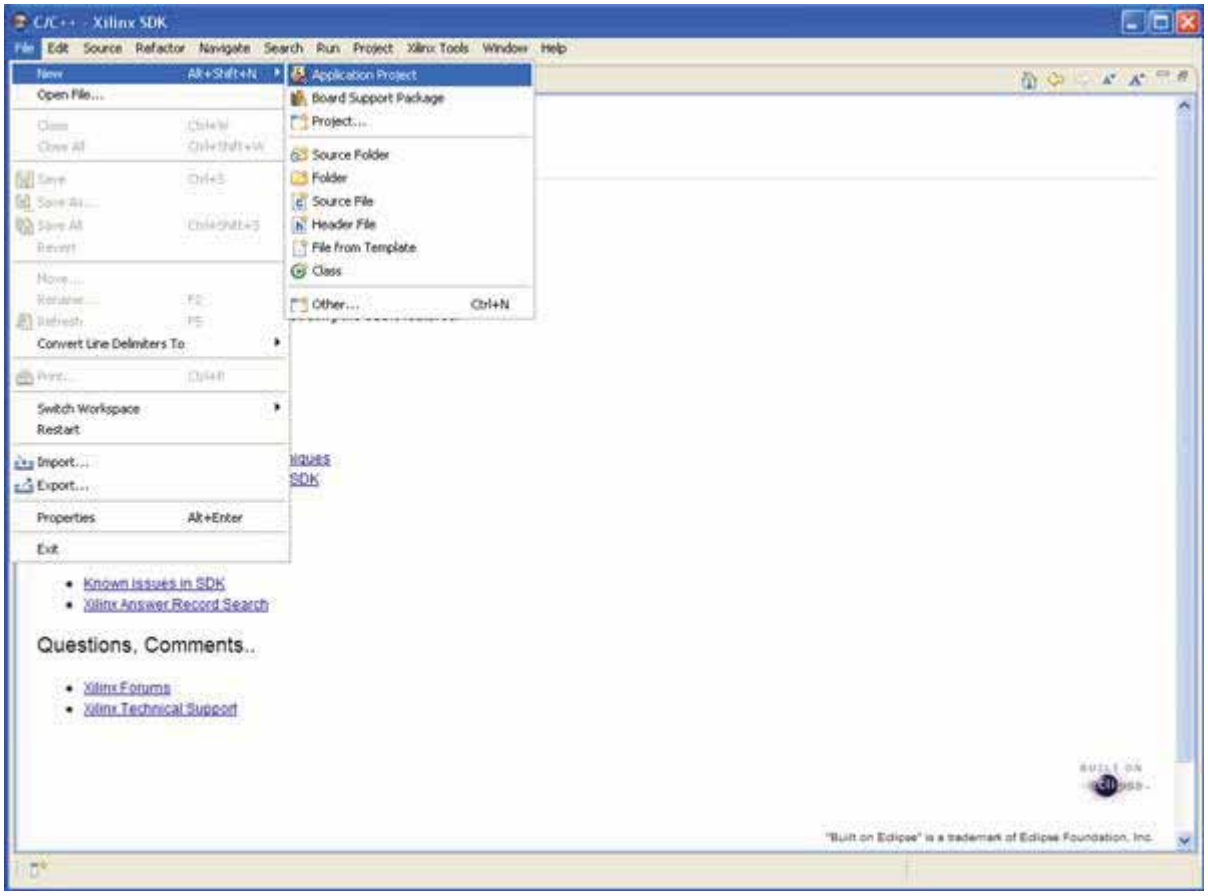
UG960_c2_34_103113

Figure 2-34: **Refresh Source Files**

Rebuilding the Zynq-7000 FSBL for the ZC702 Board

To rebuild the Zynq-7000 All Programmable SoC (AP SoC) FSBL for the ZC702 board:

1. Click **`File > New > Application Project`** as shown in Figure 2-35.



UG960_c2_35_103113

Figure 2-35: **Selecting New Application Project**

2. Set `Project name` and select **`Use Existing`** board support package. Click **`Next`**. See Figure 2-36.

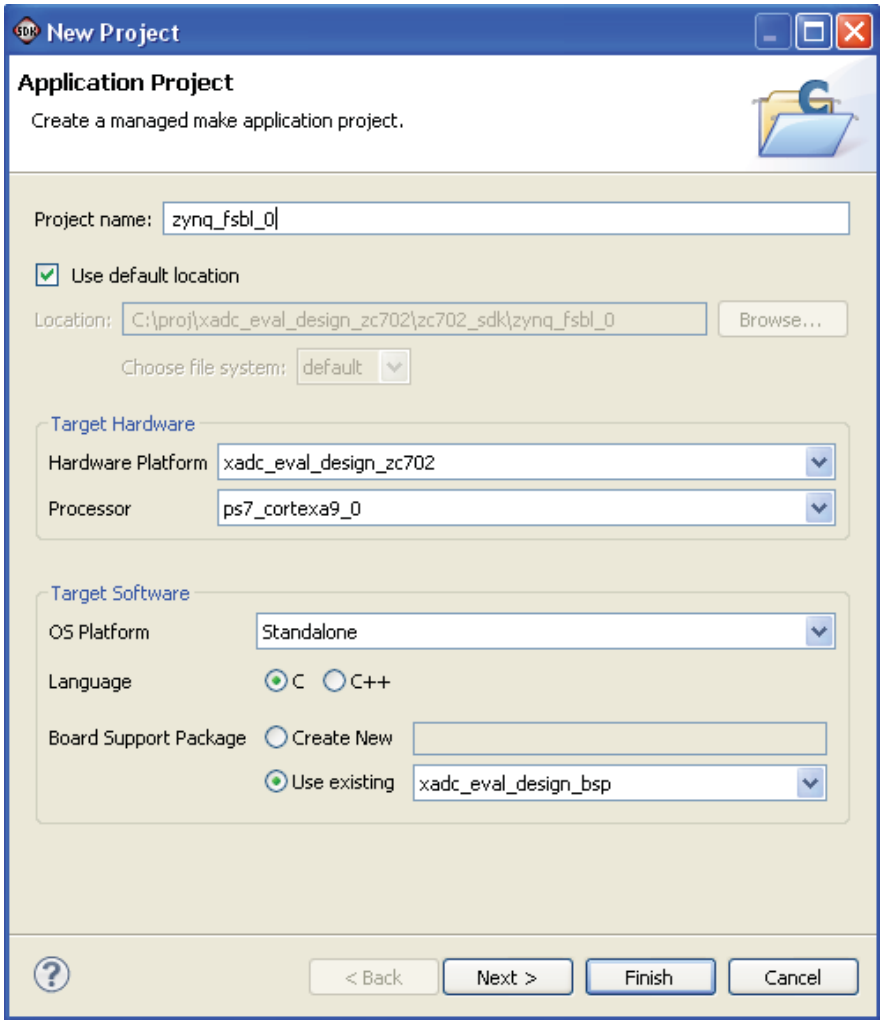
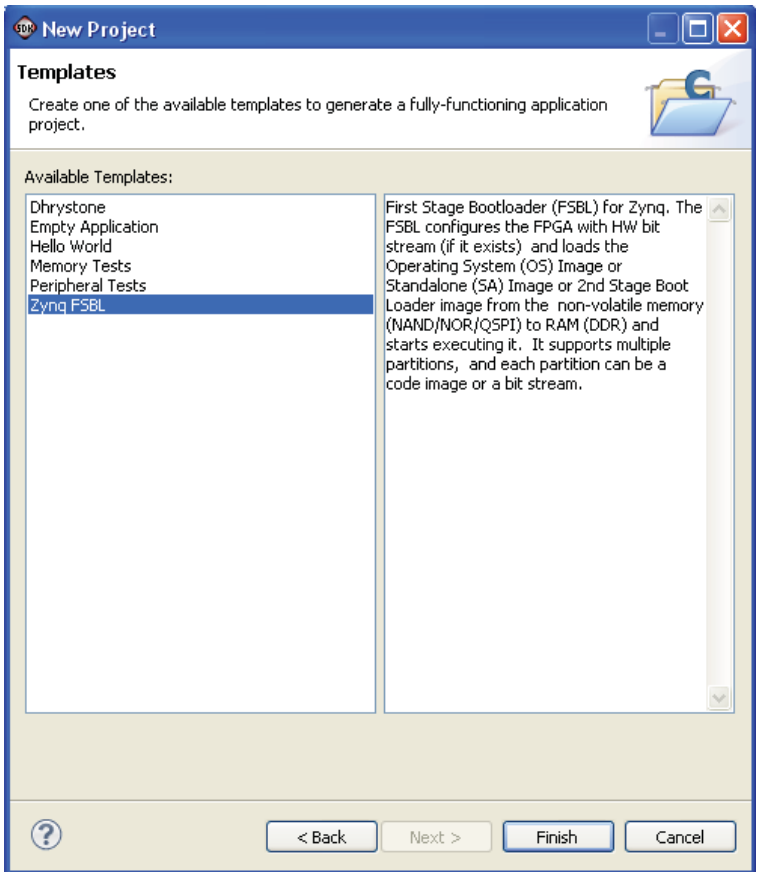


Figure 2-36: **Creating a New FSBL Application Project**

3. Select **`Zynq FSBL`**. Click **`Finish.`** See Figure 2-37.



UG960_c2_37_103113

Figure 2-37: **Selection of Zynq FSBL**

Functional Description

This design provides an evaluation platform for the Xilinx Analog-to-Digital Converter (XADC). The hardware design gathers the XADC output code samples and the software LabVIEW GUI processes these samples to derive various ADC metrics from them.

Hardware Architecture

The various blocks in the KC705, VC707, and AC701 hardware design are shown in Figure 3-1.

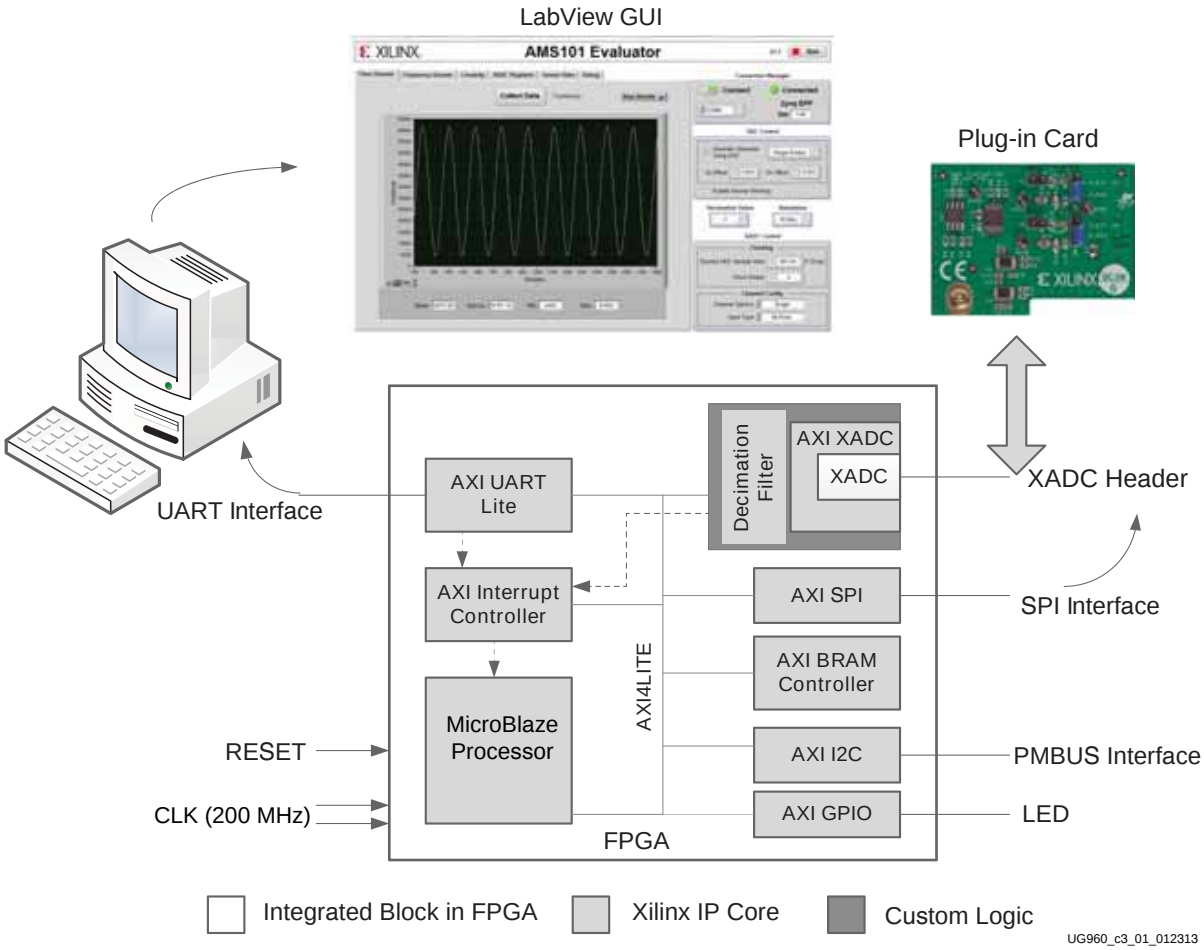


Figure 3-1: **KC705, VC707, and AC701 Hardware Block Diagram**

This is an AXI4-Lite system with a MicroBlaze™ soft processor as the master.

The system has the following peripherals:

- **AXI UART Lite**—This is the asynchronous serial communication link (operating at 115200 baud per second) between the hardware design and the PC. For details on this core refer to *LogiCORE IP AXI UART Lite Product Guide for Vivado Design Suite* (PG142) [Ref 10].
- **AXI SPI**—This interface is used to program the Analog Devices' Digital-to-Analog Converter (DAC) AD5065 on the AMS101 evaluation card. For details on this IP, refer to *LogiCORE IP AXI Serial Peripheral Interface (AXI SPI)* (DS742) [Ref 11]. Information on DAC programming is provided in DAC Programming Using SPI, page 55.
- **AXI Block RAM Controller**—This is the storage medium for XADC samples, histogram data, and so on. For further details on this IP, refer to *LogiCORE IP AXI Block RAM (BRAM) Controller Product Guide* (PG078) [Ref 12].
- **AXI GPIO**—This provides an interface to general purpose input/output. For details on this IP, refer to *LogiCORE IP AXI GPIO Product Guide for Vivado Design Suite* (PG144) [Ref 13].
- **AXI IIC**—This provides a low speed two-wire interface to external devices. For details on this IP, refer to *LogiCORE IP AXI IIC Bus Interface Product Guide* (PG090) [Ref 14]. In this design, this IP is used to drive the power management bus (PMBus) protocol which is explained in PMBus Over I2C, page 58.
- **AXI INTC**—This is the interrupt controller that concentrates multiple interrupt inputs from various peripherals to a single interrupt output to the processor. For details on this IP, refer to *LogiCORE IP AXI Interrupt Controller (INTC) Product Guide* (PG099) [Ref 15]. In this design, AXI UART Lite and AXI XADC interrupts are connected to this core. However, use of the interrupt mode is optional because it impacts the maximum sampling rate at which the XADC can operate.

The various blocks in the ZC702 hardware design are shown in Figure 3-2.

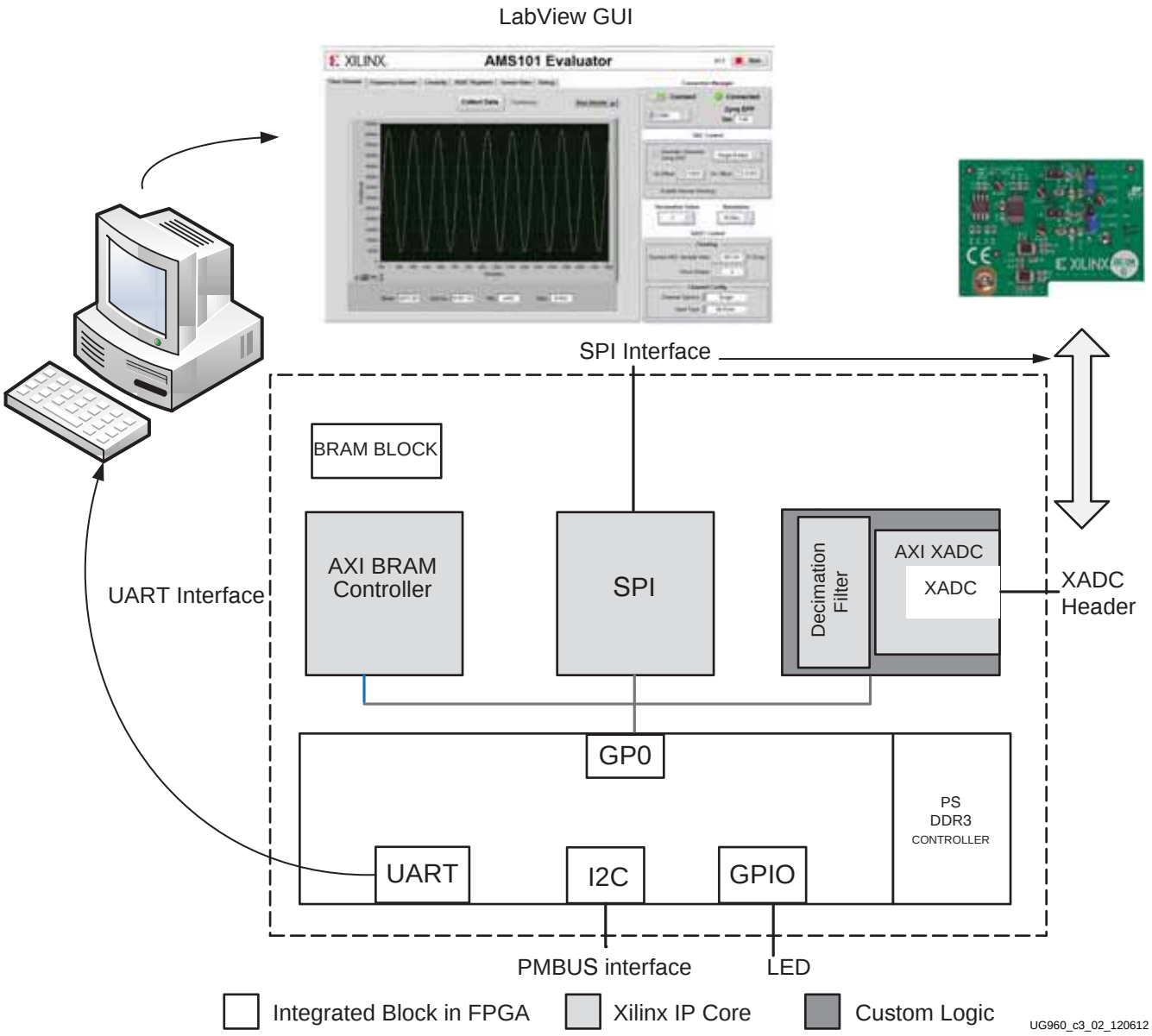


Figure 3-2: **ZC702 Hardware Block Diagram**

The difference between the above two designs is that in ZC702 design, the SPI, inter-integrated circuit (I2C), UART, and GPIO peripherals are present as hard blocks on the processor subsystem, unlike the KC705, VC707, and AC701 designs where they are implemented as softcore IPs.

Xilinx Analog-to-Digital Converter

Xilinx Analog-to-Digital Converter (XADC) is the analog subsystem in 7 series FPGAs. It includes dual, independent, 1 MSPS, 12-bit ADCs and a 17-channel analog multiplexer front end. The block also contains on-chip voltage and temperature sensors.

The specifications of the block are listed in Table 3-1.

Table 3-1: **XADC Specifications**

Parameter	Values	Comments
Resolution	12 bits	
Integral nonlinearity	±2 LSB (max.)	
Differential nonlinearity	±1 LSB (max.)	No missing codes
Sample Rate	0.1 MS/s (min.)	
	1 MS/s (max.)	
Signal to noise ratio	60 dB (min.)	F_{SAMPLE} = 500 KS/s, F_{IN} = 20 KHz
RMS code noise	2 LSB (external 1.25V reference)	
	3 LSB (on-chip reference)	
Total harmonic distortion	75 dB (min.)	F_{SAMPLE} = 500 KS/s, F_{IN} = 20 KHz

XADC Sampling Rate

For an ADC, throughput is measured as shown in Equation 3-1.

$$\text{ADC Throughput} = \frac{1}{\text{conversion time} + \text{acquisition time}} \qquad \textit{Equation 3-1}$$

Twenty-six ADCCLK cycles are required to acquire an analog signal and perform conversion. The acquisition time is the time taken by the sampling capacitor to charge up to the voltage on the analog input. See Equation 3-2,

$$t_{ACQ} = 10 \times R_{MUX} \times C_{SAMPLE} \qquad \textit{Equation 3-2}$$

where R_{MUX} is the resistance of the analog multiplexer circuit and C_{SAMPLE} is the capacitance of the sampling capacitor.

For a dedicated channel (V_P/V_N), the minimum acquisition time required is ~3 ns. For auxiliary channels, which have a much larger value of resistance, it is ~300 ns. Refer to *7 Series FPGAs and Zynq-7000 All Programmable SoC XADC Dual 12-Bit 1 MSPS Analog-to-Digital Converter User Guide* (UG480) [Ref 16] for details on these calculations.

The sampling rate of the XADC can be varied by changing the clock division ratio, which in turn varies the ADCCLK (as ADCCLK = DCLK/2). This is done by writing to bits [15:8] of Configuration register 2. These bits select the division ratio between the DRP clock (DCLK) and the lower frequency ADC clock (ADCCLK) used for the ADC. See Table 3-2.

Table 3-2: **DCLK Division Selection**

CD7	CD6	CD5	CD4	CD3	CD2	CD1	CD0	Division
0	0	0	0	0	0	0	0	2
0	0	0	0	0	0	0	1	2
0	0	0	0	0	0	1	0	2
0	0	0	0	0	0	1	1	3
0	0	0	0	0	1	0	0	4
-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-
1	1	1	1	1	1	1	0	254
1	1	1	1	1	1	1	1	255

Notes:

1. Minimum division ratio is 2, for example, ADCCLK = DCLK/2.

Input Mode

The XADC performance can be evaluated using both a unipolar and a bipolar signal.

Unipolar Mode

The nominal analog input range to the ADC is 0V to 1V in this mode. The ADC produces a zero code (0x000) when 0V is present at the input and a full scale code of all ones (0xFFF) when 1V is present at the input. The output coding is straight binary. The LSB size in volts is equal to $1V/2^{12}$ which is 244 µV. See Figure 3-3

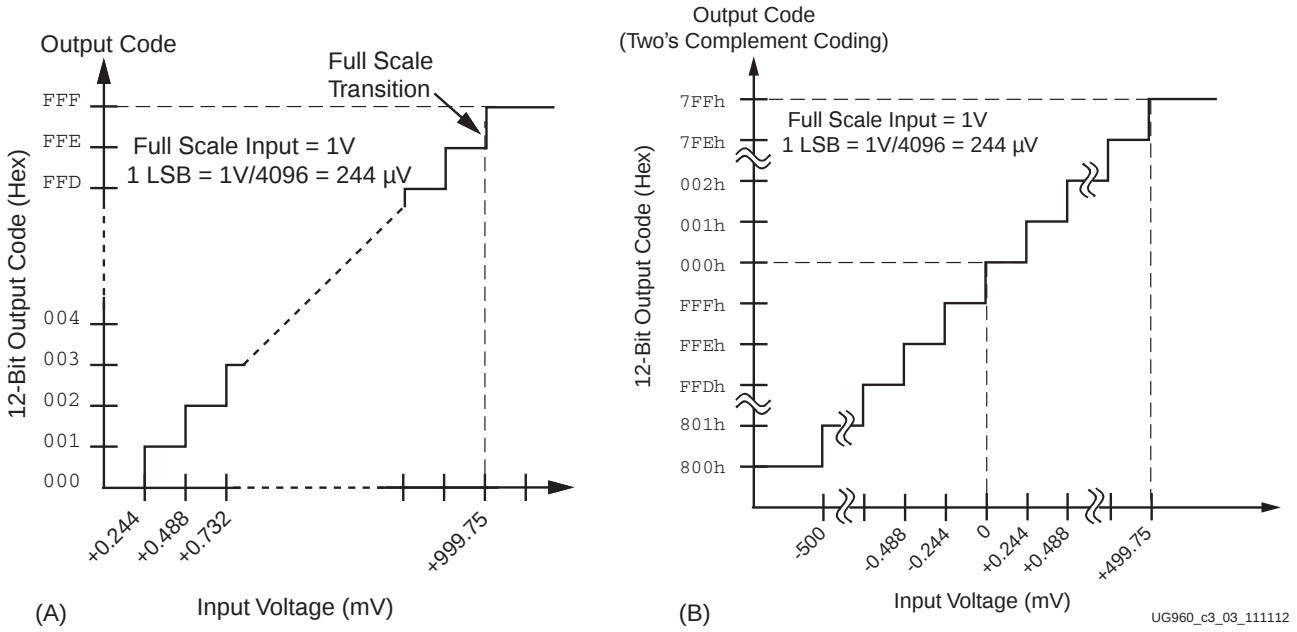


Figure 3-3: **XADC Unipolar (A) and Bipolar (B) Transfer Functions**

Bipolar Mode

XADC can accommodate true differential bipolar analog signal types in this mode. Since both magnitude and sign are needed for differential signal types, the output coding is two's complement here. This is intended to indicate the sign of the input signal on V_P relative to V_N. Output code of `0x800` represents –500 mV analog input and output code of `0x7FF` represents 500 mV analog input.

Sensor

The XADC has temperature and power supply sensors whose output voltage is digitized by the ADC and provided in its status registers.

This design also showcases the temperature and power supply voltage monitoring capability of the XADC block.

Temperature Sensor

The XADC contains a temperature sensor that produces a voltage output proportional to the die temperature. The output voltage of the sensor is shown in Equation 3-3,

$$Voltage = 10 \times \frac{kT}{q} \times In(10) \qquad \text{Equation 3-3}$$

where,

k is Boltzmann's Constant = 1.38 x 10^{-23} J/K

T is the temperature in Kelvin

q is the charge on an electron = 1.6 x 10^{-19} C

The output voltage of this sensor is digitized by the ADC to produce a 12-bit digital output code.

The temperature sensor (Figure 3-4) has a transfer function given by Equation 3-4.

$$Temp(°C) = \left(\frac{\text{ADC Code} \times 503.975}{4096} - 273.15\right) \qquad \text{Equation 3-4}$$

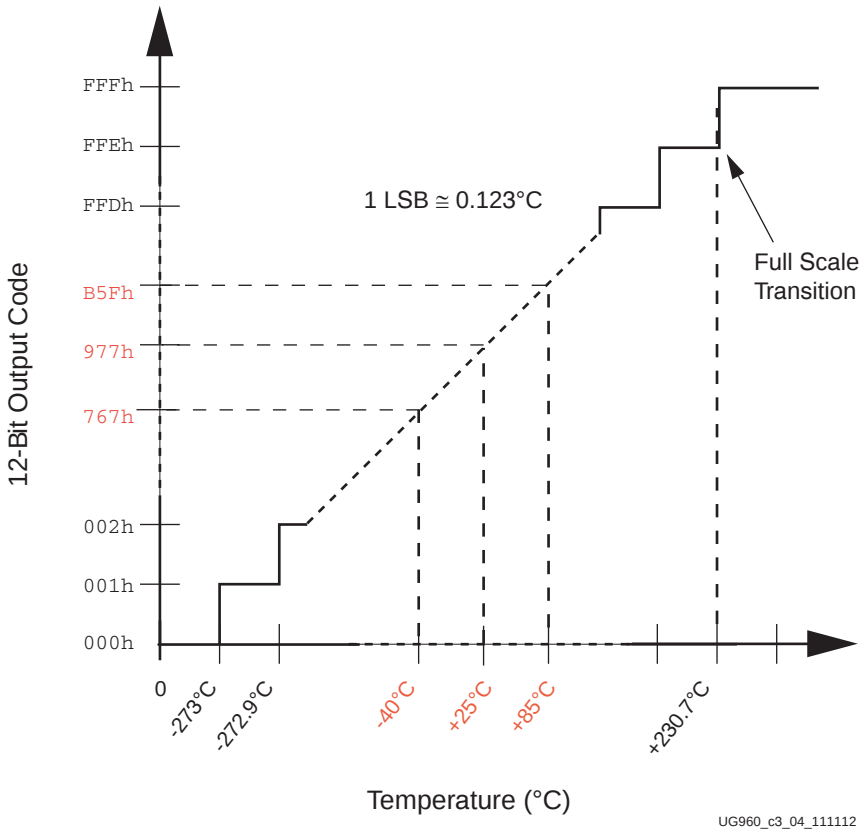


Figure 3-4: **Temperature Sensor Transfer Function**

This sensor has a maximum measurement error of +/- 4% over a range of –40°C to 125°C. The temperature measurement result is found in the Status register at `0x00`.

Power Supply Sensor

The XADC includes on-chip sensors that allow monitoring of FPGA power supply voltages using the ADC. The sensors sample and attenuate (by a factor of 3) the power supply voltages V_{CCINT}, V_{CCAUX} and V_{CCBRAM} on the package power supply balls.

The transfer function of the sensor is shown by Equation 3-5.

$$Voltage = \frac{\text{ADC Code}}{4096} \times 3V \qquad \textit{Equation 3-5}$$

This can be used to measure voltages in the range 0V to V_{CCAUX} + 5% with a resolution of approximately 0.73 mV.

The XADC power supply sensors have transfer function that generates a full scale ADC output code of `0xFFF` with a 3V input voltage (see Figure 3-5). This voltage is outside the allowed supply range, but the FPGA supply measurements map into this range. Thus V_{CCINT} = 1V generates an output code of 1/3 x 4096 = 0x555. The XADC monitors V_{CCINT}, V_{CCAUX}, and V_{CCBRAM}. The measurement results are stored in Status registers at offsets 0x01, 0x02, and 0x06, respectively.

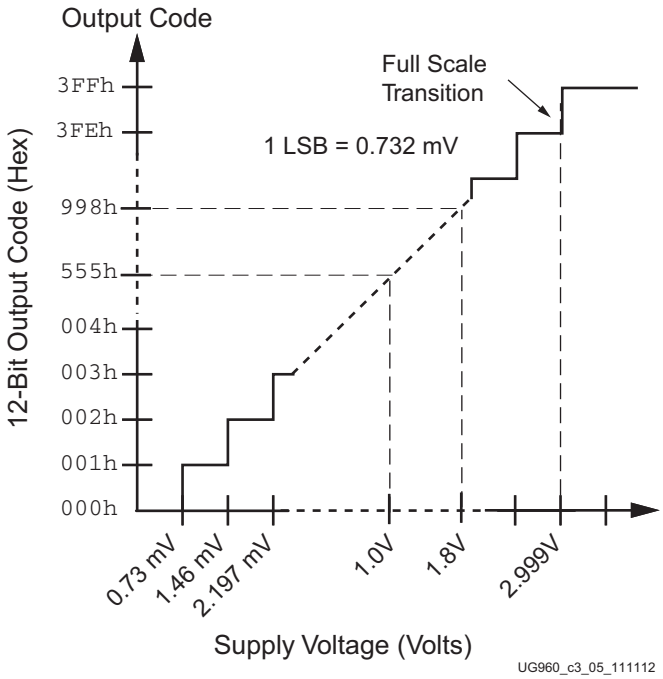


Figure 3-5: **Ideal Power Supply Transfer Function**

Decimation Filter

The AMS reference design enables decimation of the XADC data by a certain factor, effectively trading off input bandwidth for higher SNR performance. The available factors are 1, 2, 4, 8, or 16. A decimation of 1 indicates that the XADC data is passed directly to the application program without any filtering or decimation. The decimation function is carried out in the FPGA using very little resources. The core building block is the *decimate by 2* block. It first passes the XADC data through a half-band filter and then decimates by a factor of 2, as shown in Figure 3-6. Decimating by 2 cuts the input bandwidth in half. To achieve a decimate by 4, the FPGA passes the XADC data through the decimate by 2 block and feeds back its output to the block's input so that it can be band-limited and decimated by 2 again, giving an overall decimation rate of 4. For a decimation rate of 8, the data is looped back through the decimate by 2 block a second time.

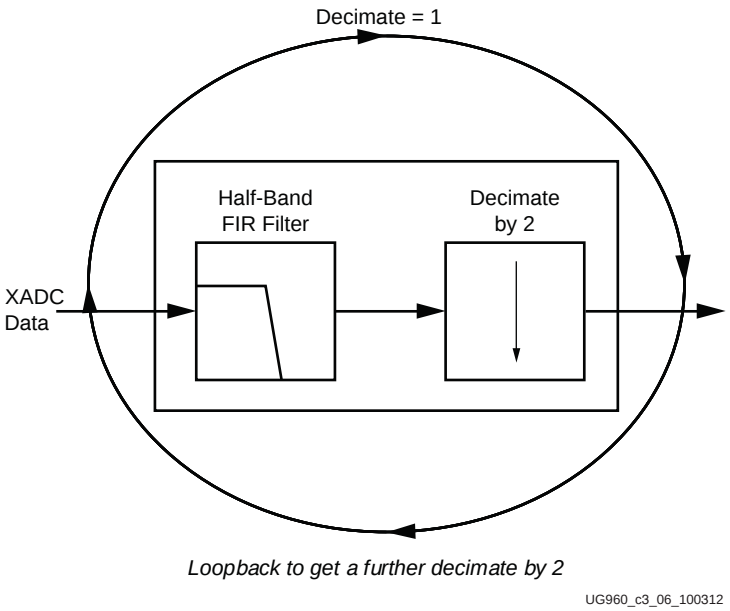


Figure 3-6: **Decimate by Two**

DAC Programming Using SPI

The AMS101 evaluation card contains an onboard signal source in the form of the AD5065 Digital-to-Analog Converter. AD5065 is R-2R type of dual DAC with 16-bit resolution.

The input coding of the DAC is binary; the ideal output voltage is given by Equation 3-6,

$$V_{OUT} = V_{REF} \times D / 2^N \qquad \text{Equation 3-6}$$

where,

V_{REF} is the Reference voltage

D is the decimal equivalent of the binary code loaded to the DAC register

N is the DAC resolution

Communication with the FPGA is done with the Serial Peripheral Interface (SPI) through the XADC header pins.

Configuring DAC

The AMS101 evaluation card embeds Analog Device's AD5065 DAC programmable through the SPI interface. The DAC provides an input configuration register which is updated when data is received in the SPI interface attached to the DAC. In the targeted design platform (TDP), the MicroBlaze processor is configured as an SPI master and DAC as a slave. the master initiates a write operation at the falling edge of the SPI clock and asserts the SPI Slave Select signal. Each write operation consists of a 32-bit serial write and Slave Select is asserted for 32 SPI clock cycles, after which it is de-asserted.

Every subsequent write into the DAC can happen at an interval of 2 µs. MicroBlaze processor decides appropriate command and address values to be written to the DAC

register based on the DAC configuration. In the TDP, polling mode of operation is performed to write different digital codes to DAC input register.

Figure 3-7, Table 3-3, and Table 3-4 provide details for register programming.

The data bits in the Input register (Figure 3-7) comprise the input code.

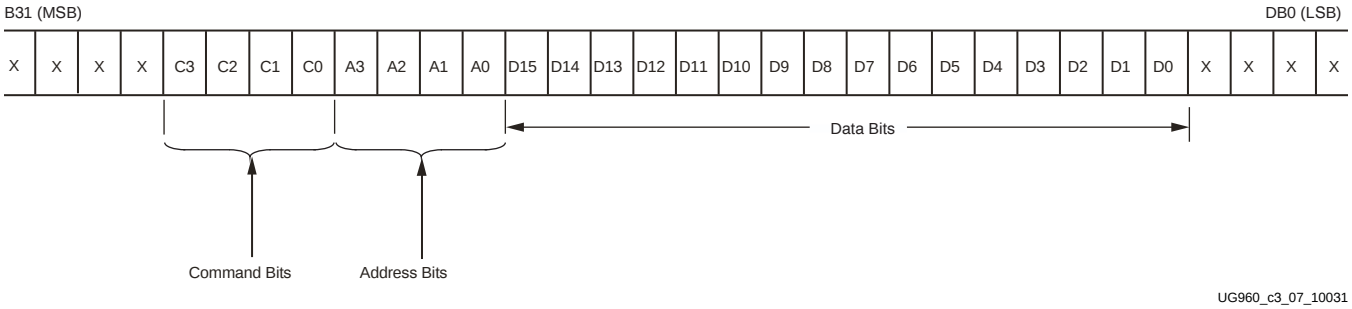


Figure 3-7: **AD5065 Input Register**

Table 3-3: **AD5065 Address Commands**

Address (n)				Selected DAC Channel
A3	A2	A1	A0	
0	0	0	0	DAC A
0	0	1	1	DAC B
0	0	0	1	Reserved
0	0	1	0	Reserved
1	1	1	1	Both DACs

Table 3-4: **AD5065 Command Description**

Command				Description
C3	C2	C1	C0	
0	0	0	0	Write to Input register n[1]
0	0	0	1	Update DAC register n[1]
0	0	1	0	Write to Input register n, update all (software $\overline{\text{LDAC}}$).
0	0	1	1	Write to and update DAC channel n[1]
0	1	0	0	Power down/power up DAC
0	1	0	1	Load clear code register
0	1	1	0	Load $\overline{\text{LDAC}}$ register
0	1	1	1	Reset (power-on reset)

Table 3-4: **AD5065 Command Description *(Cont'd)***

Command				Description
C3	C2	C1	C0	
1	0	0	0	Set up DCEN register (daisy chain enable)
1	0	0	1	Reserved
1	1	1	1	Reserved

Notes:

1. Refer to Table 3-3 for the defined register.
2. The input address and command combinations not defined above are to be considered reserved.

The LabVIEW GUI provides an option for writing to both DAC A and DAC B individually.

Ramp Signal Generation for Linearity Testing

The AD5065 DAC can be programmed to generate ramp type output, which can in turn be applied to XADC V_P/V_N input. XADC samples this analog input and produces digital code. The generated digital codes can be used to characterize the XADC for INL and DNL errors.

To generate the ramp signal, the SPI interface is initialized first with the polling mode of operation. The MicroBlaze processor writes a 32-bit burst into DAC, every time incrementing the DAC code by 1 till the DAC code reaches `7FFF`. For each DAC-produced analog output, XADC samples the voltage, produces the digitized output, and flags the conversion by End of Conversion status. The XADC's output is stored in block RAM for subsequent linearity analysis.

For sampling the ramp produced by DAC, XADC needs to be operated at a slower rate to cope up with the slow running DAC. Because the DAC can be updated at a 2 µs interval, XADC needs to be run at 500 kHz.

To generate a ramp source for linearity test, the input code to DAC should be incremented every 1, 2, or 4 XADC samples. DAC A starts at `0x0000` and be incremented till `0x7FFF`. DAC B is hard-coded to `0x0100`.

Sinusoidal Signal Generation

The DAC can be programmed to generate a sinusoidal output of a specific frequency. Sine wave samples are stored in block RAM as part of the peripheral initialization procedure. The MicroBlaze processor reads the block RAM samples, adds the appropriate address and command value for the DAC, and writes to the DAC's input register. After all samples for a sine wave cycle are written, the MicroBlaze processor loops over the same block RAM samples, writes the values to the DAC, and the procedure repeats.

The ratio of sampling frequency to the input signal frequency is kept to be a prime integer for coherent sampling. In the Xilinx 7 series AMS reference design, this prime number is chosen to be 23. For the computation of FFT, 4096 sine wave samples spread across 23 sine wave cycles are acquired. Thus each sine wave cycle has 4096/23 = 178 samples. The XADC requires 1 microsecond (µs) to process one sample, thus 178 samples require 178 µs. This causes the frequency of the sine wave to be 1/178 µs = 5.6 kHz.

For normal data collection for analysis, the DAC should be programmed to generate a sinusoidal source for the XADC. A fully differential signal can be created using DACA as P and DAC B as N inputs. A digital sine wave can be written to block RAM. Incrementing

through each of these block RAM addresses and writing input codes to DAC would produce a sine wave. This operation can be looped over to provide several periods of the wave. Frequency of the sinusoid can be varied by reading from the block RAM either slowly or quickly.

PMBus Over I2C

The KC705, VC707, and ZC702 hardware platforms have TI power regulators (UCD9248). The power management bus (PMBus) is the open standard protocol for communication with power controller devices. The PMBus is a low speed interface and is an extension of I2C. PMBus is a fully shared bus in which each device is allocated a unique address. The I2C interface is used to drive the PMBus in this design.

The MicroBlaze processor is the master and all the UCD9248 devices connected are slaves.

At the start of a transaction, the master transmits a 7-bit address (followed by zero indicating that master is going to transmit the next byte) of the intended slave followed by the command. The PMBus specification describes the commands in detail and *UCD92xx Digital PWM System Controller PMBus Command Reference* [Ref 19] provides a good summary of the commands.

For write transactions, the command is followed by write data from the master and for reads, the slave address (7-bit address followed by one to indicate reads) is sent with repeat start to get the read data from the slave.

AC701 AMS Power Demo Design using XADC

The AC701 evaluation board contains onboard multiplexers that multiplex voltage and current for the rails listed in Table 3-5.

Table 3-5: **Summary of Voltage Rails Monitored through External Multiplexer**

Rail Name	Voltage	Current
V_{CCINT}	No	Yes
V_{CCAUX}	No	Yes
V_{CCBRAM}	No	Yes
1.5V Supply	Yes	Yes
V_{CCO_ADJ}	Yes	Yes
1.8V Supply	Yes	Yes
3.3V Supply	Yes	Yes
MGTAVCC	Yes	Yes
MGTAVTT	Yes	Yes

The V_{CCINT}, V_{CCAUX} and V_{CCBRAM} voltage levels are measured by the XADC on-board sensors.

The current sense values of V_{CCINT}, V_{CCAUX}, V_{CCBRAM}, 1.5V supply and V_{CCO_ADJ} along with voltage levels of 1.5V supply, V_{CCO_ADJ} and 1.8V supply are available on the AC701 onboard MUX positioned at U13. The differential output of the MUX is connected to auxiliary pin 1 (VAUXP/N 1) of XADC and the channel is sampled periodically by the MicroBlaze program.

The current sense values of 1.8V Supply, 3.3V Supply, MGTAVCC and MGTAVTT along with voltage levels of 3.3V Supply, MGTAVCC and MGTAVTT are available on the AC701 onboard MUX positioned at U14. The differential output of the MUX is connected to auxiliary pin 9 (VAUXP/N 9) of XADC and the channel is sampled periodically by the MicroBlaze™ processor program.

You can read the voltage, current, and power number of each rail by browsing to the Power Monitor tab in the LabVIEW GUI.

Clocking Scheme

The design runs at a uniform clock of 100 MHz. The AXI4-Lite interface connects all the peripherals with the MicroBlaze processor operating at 100 MHz. The internal clocks required by peripherals are generated from the 100 MHz AXI4-Lite clock. The SPI IP generates the SPI clock by dividing the AXI4-Lite clock by 4. XADC IP generates the ADCCLK by dividing the 100 MHz clock with the user-set divider value.

A 200 MHz differential clock is applied to the design. The clock is divided by two and applied to the MicroBlaze processor.

Table 3-6 summarizes various clocking schemes used in the design.

Table 3-6: **Clocking Schemes**

Interface	Clocking Scheme
AXI4-Lite	100 MHz MicroBlaze processor clock
SPI master clock	100 MHz CPU clock divided by 4
DCLK of XADC	100 MHz CPU clock
ADCCLK of XADC	Generated from DCLK
I2C clock	100 MHz MicroBlaze processor clock
UART clock	100 MHz MicroBlaze processor clock

Software Architecture

The software on the MicroBlaze processor is responsible for the following:

1. Initialization and configuration of all peripherals in the design
2. Getting commands from UART, interpreting them, and performing one of the following functions:
   a. Programming DAC over SPI
   b. Programming XADC registers
   c. Reading appropriate XADC data, storing it in block RAM, and sending it back over UART to the PC
   d. Programming the UCD9248 device over PMBus for voltage variation

The software layers are as shown in Figure 3-8. The drivers for each of the peripherals are provided by the EDK. The application managing various MicroBlaze processor routines is written for this design.

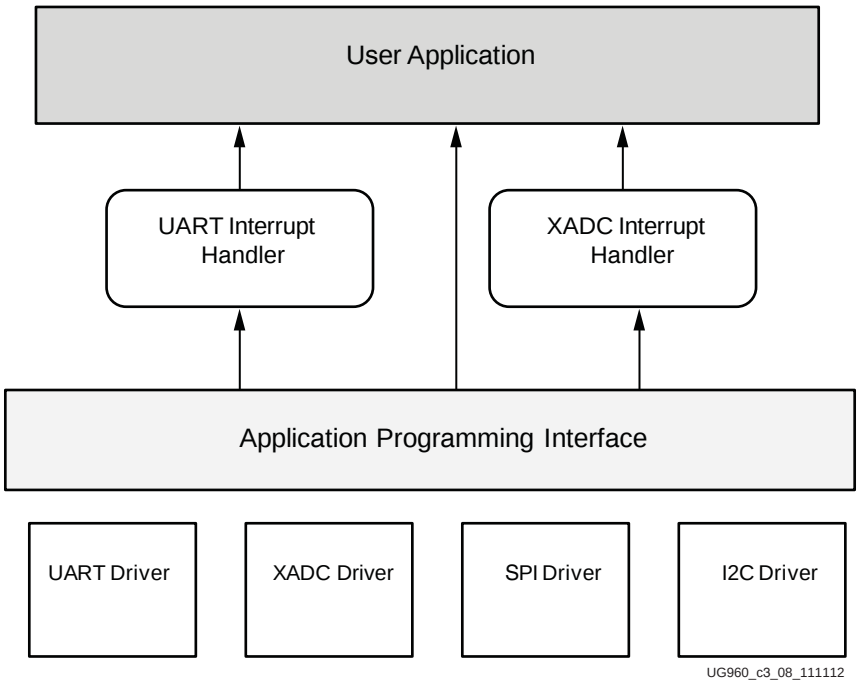


Figure 3-8: **Software Layers**

The XADC produces an End-of-Conversion (EOC) output at the end of each conversion cycle. On the occurrence of every EOC (this can be either polled or used as an interrupt—use of interrupts prevents the design from operating at speed hence polling is desirable), the status register of XADC containing the output digital code is read and the value read is written into block RAM.

1. **Raw Data Collection—**4096 continuous samples from XADC are collected in hardware, buffered, and sent over UART for FFT calculation. Various frequency domain metrics are derived from it.
   a. **V_P/V_N—**For this channel, the input signal source could either be an external sinusoid signal generator or the DAC on the AMS card programmed periodically with sine wave samples so as to generate a sine wave of a fixed frequency.
2. **Histogram—**The histogram is plotted in the time domain tab of AMS101 Evaluator GUI for two use cases:
   a. **DC signal analysis**—To analyze the noise and the mean of the noise probability distribution, which is a Gaussian distribution.
   b. **Linearity analysis**—Every XADC code has a code hit and the data can be used to analyze the linearity of the XADC.
3. **Sensor data—**The GUI gets 8 bytes of raw samples each of temperature, V_{CCAUX}, V_{CCINT}, and V_{CCBRAM} from hardware to be plotted.
   a. **Voltage supply variation—**UCD9248 power controllers available on KC705, VC707, and UCD90120 available on ZC702 boards can be programmed to vary voltage values. This programming is done through the PMBus based on user selection from the GUI.

   Caution! This design showcases a way to vary voltage through the MicroBlaze processor and PMBus. Be extremely careful when varying voltages, to ensure the allowed range is not exceeded, which can damage the board or parts.

Figure 3-9 summarizes the MicroBlaze soft processor tasks.

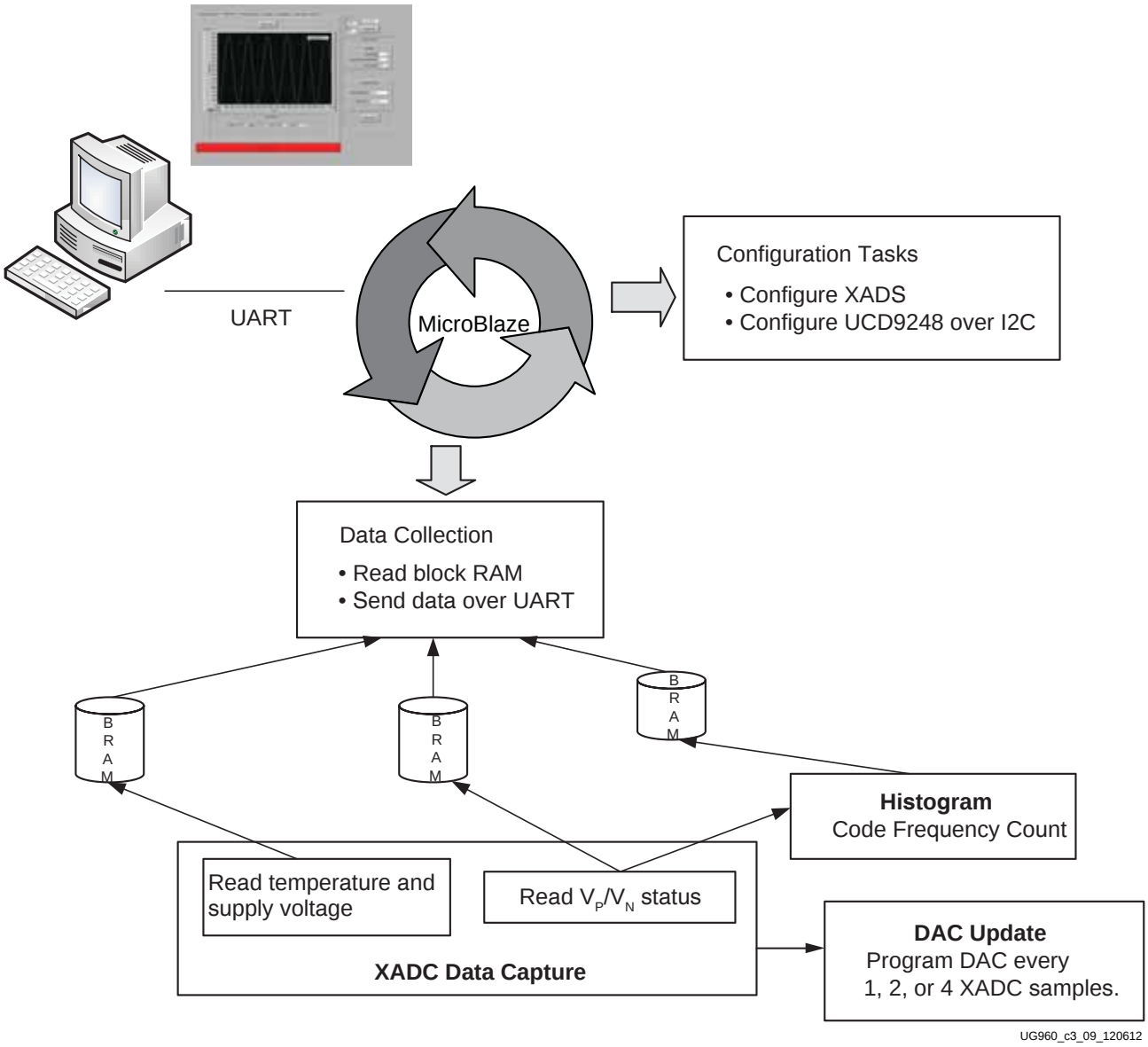


Figure 3-9: **Summary of MicroBlaze Soft Processor Tasks**

XILINX

Chapter 4

Understanding ADC Metrics

Appendix A, ADC Basics provides a quick overview on basics of analog-to-digital converters which can be used as a refresher before getting further into this chapter.

This chapter discusses various ADC metrics, their measurements, and result interpretation in detail.

Linearity

An ideal ADC exhibits a linear transfer function. All ADCs suffer from non-linearity errors caused by their physical imperfections, causing their output to deviate from a linear function (or some other function, in the case of a deliberately non-linear ADC) of their input. These errors can sometimes be mitigated by calibration, or prevented by testing.

Linearity specifications are important in image processing applications employing ADCs.

Differential and *integral* nonlinearity are the two important metrics defining ADC linearity.

Differential Nonlinearity (DNL)

Differential non-linearity is defined as the variation in analog step sizes away from 1 Least Significant Bit (LSB). Assuming an ideal Analog-to-Digital Converter (ADC) with finite digital codes exactly 1 LSB apart (DNL error = 0) or an ideal Digital-to-Analog Converter (DAC) with analog output values exactly one code apart (DNL error = 0), the DNL error is defined as the difference between the ideal and the measured code transitions for successive codes for an ADC or the difference between the ideal and the measured output value between successive DAC codes. Variation in code size is determined by the matching accuracy of converter components.

If a code is too wide, it is said to have a positive DNL error. If it is too narrow, it has a negative DNL error.

DNL—a static specification, relates to signal-to-noise ratio (SNR)—a dynamic specification. SNR tends to become worse as DNL departs from zero.

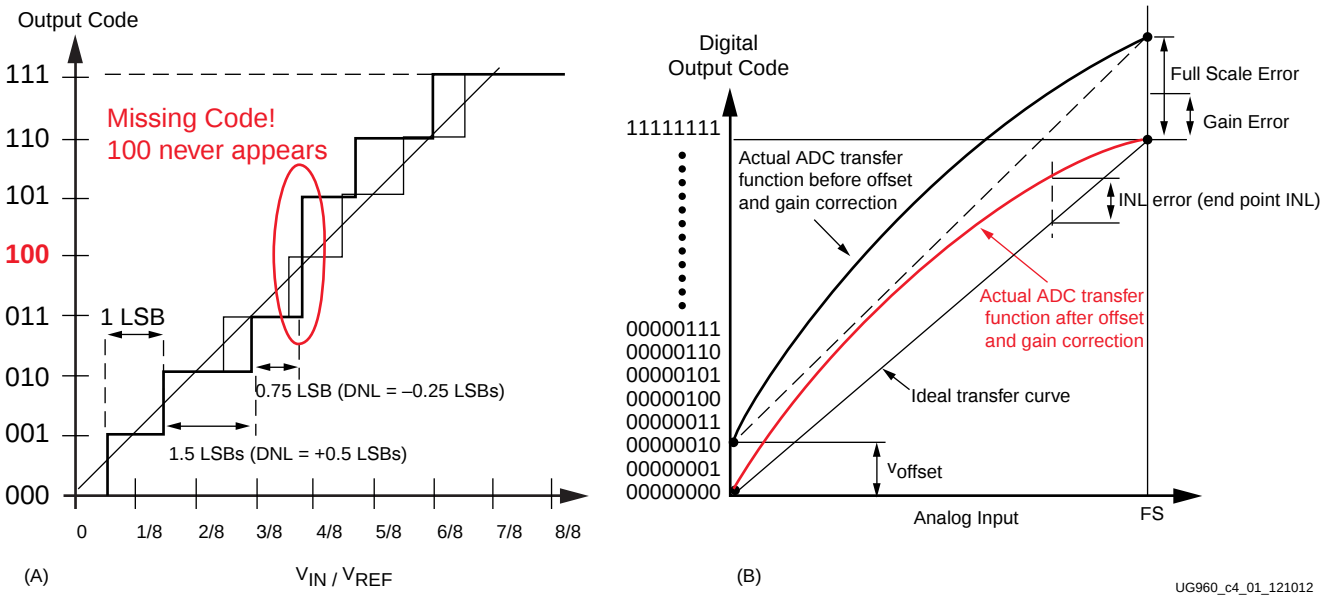


Figure 4-1: **Plot depicting DNL (a) and INL (b)**

Integral Nonlinearity (INL)

The integral non-linearity is defined as deviation from a straight line after offset and gain errors are removed. INL is the amount of deviation of the measured transfer functions of an Analog-to-Digital Converter (ADC) or a Digital-to-Analog Converter (DAC) from the ideal transfer function (defined as a straight line drawn from zero to full scale). Sometimes a "best-fit" straight line is used, where the ideal transfer function is represented by a straight line drawn between the end points of the actual transfer function. See Figure 4-1.

INL—a static specification, relates to total harmonic distortion (THD)—a dynamic specification. THD tends to become worse as INL departs from zero.

Test Methodology

INL/DNL data is gathered from a histogram obtained while the input signal applied to XADC is ramping. In this design, the MicroBlaze™ processor builds the histogram data from the XADC samples obtained while DAC ramps from 0V to 1V.

The histogram essentially counts the frequency of occurrences of an output code. The output code frequencies are stored in FPGA block RAM while a ramp is being applied to ADC's input. Each block RAM location corresponds to an output code from the XADC. If a specific code is received from XADC, the data in the address corresponding to that code is incremented. For example, if the data coming from the XADC for a particular sample is `0x100` then the content in memory location `0x100` increments by 1.

As an example, assume a 4-bit ADC has output codes from that ADC as listed in Table 4-1.

Table 4-1: **Output Codes from ADC**

BRAM Address	Code
ADC[0]	0x1
ADC[1]	0x3
ADC[2]	0x4
ADC[3]	0x3
ADC[4]	0x1

Corresponding to 16 possible results, 0x0, 0x1,…, 0xF, 16 block RAM locations are considered and the content of each is initialized to 0x0. Every time an ADC data is read, the data in that memory location is incremented.

Thus after the five ADC samples listed above pass through the histogram function, the data in memory is as follows:

Address	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	0x8	0x9	0xA	0xB	0xC	0xD	0xE	0xF
Data	0	2	0	2	1	0	0	0	0	0	0	0	0	0	0	0

The LabVIEW GUI reads this block RAM content for INL/DNL plots. This approach of histogram test is also referred to as the *code density test*.

Dynamic

Dynamic specifications of ADC are important in high speed applications such as digital communications, ultrasound imaging, and instrumentation. The key dynamic specifications include signal-to-noise ratio (SNR), total harmonic distortion (THD) and spurious free dynamic range (SFDR). These dynamic specifications are expressed in the frequency domain, using the Fast Fourier Transform (FFT).

Measurement of dynamic specifications helps in understanding how the conversion function alters the spectrum of a signal transmitted through a system.

Signal-to-Noise Ratio (SNR)

Signal-to-noise ratio is a ratio of the output signal amplitude to the output noise level, not including the harmonics or DC. For an Analog-to-Digital Converter with resolution N, the SNR for an input sinusoid is given by Equation 4-1. *dB* represents unit in decibels.

Equation 4-1

$$SNR = 6.02 \times N + 1.76 \text{ dB}$$

Total Harmonic Distortion (THD)

Total harmonic distortion gives an indication of a circuit's linearity in terms of its effect on the harmonic content of a signal. THD is ratio of the RMS total of the first several harmonic components to the RMS value of the output signal and relates the RMS sum of the amplitudes of harmonics to the amplitude of the fundamental.

To calculate THD, sum the power in each of the harmonics and divide by the total power of the fundamental. Thus, the equation for THD is shown in Equation 4-2.

$$THD = \sqrt{\frac{P1 + P2 + \ldots}{P_{Fund}}} \qquad \textit{Equation 4-2}$$

Signal-to-Noise and Distortion (SINAD)

Signal to noise and distortion, also known as *signal to noise plus distortion* is a combination of the SNR and THD specifications. It is calculated from SNR and THD per Equation 4-3.

$$SINAD = 20 \times \log\sqrt{10^{-SNR/10} + 10^{THD/10}} \qquad \textit{Equation 4-3}$$

SINAD compares all undesired frequency components with the input frequency thus providing an overall measure of an ADC's dynamic performance. The higher the SINAD figure, the better the general performance.

Spurious Free Dynamic Range (SFDR)

Spurious free dynamic range is the difference between the RMS value of the desired output signal and the highest amplitude output frequency that is not present in the input, expressed in dB. Neither THD nor SINAD can ever be better than SFDR. Mathematically, SFDR can be expressed as Equation 4-4.

Equation 4-4

$$SFDR = 20 \times \log(\text{Amplitude of Fundamental (RMS)} / \text{Amplitude of Largst Spur } (RMS))dB$$

Test Methodology

Arrays of raw data (4096 samples) gathered from XADC are buffered in block RAM and transmitted to PC over UART. This data is processed in the LabVIEW GUI—first windowed, followed by FFT operation. Parameters SNR, THD, and SFDR are calculated based on the result of the FFT operation.

Fast Fourier Transform

The FFT process starts at the ADC, which converts a continuous analog signal to a discrete digital representation (sampling). The ADC outputs these digital words at a rate set by the sample frequency. The FFT algorithm assumes that the input signal is periodic over the number of points in FFT—this requirement is not practical in most applications. Hence, a technique called *windowing* is used. Windows are used to reduce the spectral leakage that results from using the FFT on non-periodic or dynamic signals. Window selections for dynamic measurements aim to keep the fundamental energy in the main lobe and have negligible leakage to the side-lobes.

Thus, time domain signals are transformed to frequency domain. The magnitude portions of the frequency elements are used to determine the signal power distribution with respect to the frequency, by constructing a power spectrum plot.

FFT Magnitude Plot

Quantitative measurements of system performance can be made from the results presented in the power spectrum as shown in Figure 4-2. In an ideal system, all the energy is concentrated in bins corresponding to the frequency of the input sine wave. The noise

floor of the spectrum is flat and at a level corresponding to the number of bits of resolution used by the ADC. Various ratios quantify how a signal is corrupted with noise and distortion.

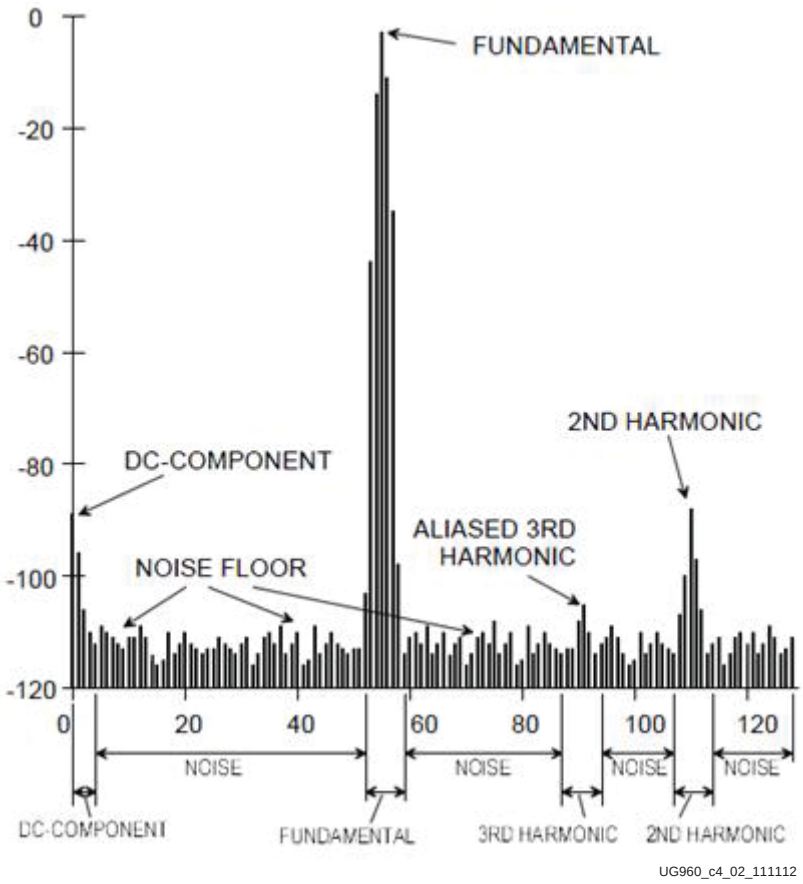


Figure 4-2: **Components of FFT Magnitude Plot**

In Figure 4-2, the Y-axis represents the power concentrated in the frequency range represented by each bin. Power is plotted in terms of decibels (dB) and the ratio is taken with respect to the signal power of a full scale sinusoid.

The highest bin locates the fundamental signal.

When using an external signal generator to supply an AC signal (rather than the AD5065 DAC), synchronization of the external generator and the Convert Start signal of the XADC is required for coherent sampling. Windowing can be avoided by using coherent sampling—this places certain restrictions on input and sampling frequencies. The idea is to sample the input waveform at different phases each cycle so that more information is added to FFT each cycle.

Coherent sampling can be achieved in one of the two ways-

- Clock output from the FPGA design can be used as a sync signal for the external generator.
- An external clock signal can be used to drive the sync signal of the generator and the clock input of the FPGA design.

DC

Code Transition Noise

As the analog input voltage is increased, an ideal ADC maintains a constant output code until the transition region is reached, at which point the output code instantly jumps to the next value and remains there until the next transition region is reached. A theoretically perfect ADC has zero code transition noise and a transition region width equal to zero (Figure 4-3). A practical ADC has a certain amount of code transition noise and therefore a transition region width that depends on the amount of input referred noise present.

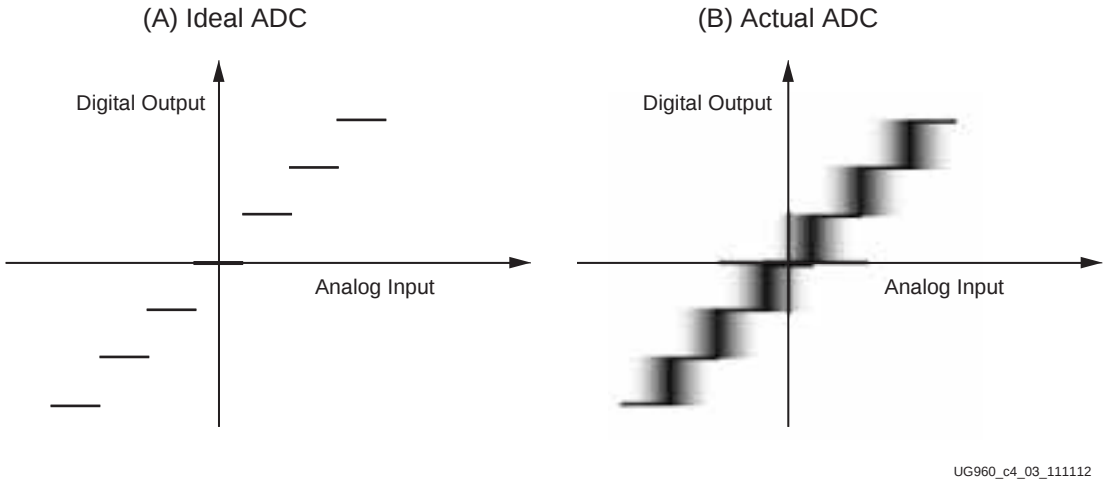


Figure 4-3: **Code Transition Noise and ADC Transfer Function**

This noise is present even for DC input signals. This is most often characterized by examining the histogram of a number of output samples when the input to ADC is a DC value.

Test Methodology

Arrays of raw data gathered from XADC are buffered in block RAM and transmitted to PC over UART periodically.

An analog input voltage is applied, a histogram of counts per code is plotted, and the mean output code and its standard deviation is calculated.

An FFT is also performed on the received data and the FFT plot is available in the frequency domain tab of AMS101 Evaluator GUI.

XILINX

Chapter 5

Applications

A LabVIEW based graphical user interface tool configures the UART interface that connects the application to the MicroBlaze™ processor. The GUI performs the following operations:

1. It monitors the raw XADC samples, and calculates the mean and standard deviation of received data.
2. It computes FFT of received data, and calculates SNR, THD, and ENOB. It provides options to configure the decimation filter.
3. It plots the XADC's INL and DNL errors.
4. It reads XADC configuration registers.
5. It configures the onboard TI power regulator and plots the XADC acquired sensor data.

The GUI conveys the user-configured information to the MicroBlaze processor using the UART-to-USB bridge interface. As a response to the GUI command, the MicroBlaze processor sends 4096 XADC samples to the LabVIEW GUI for further processing the data and plotting the data on screen.

The GUI provides flexibility to plot data in Raw format or in Histogram format, where each ADC code is represented as bin hit in the Time Domain tab. In each of the Time Domain, Frequency Domain, Linearity, and Sensor Data tabs, you initiate the data collection by triggering the **`Collect Data`** button. Figure 5-1 shows the LabVIEW GUI that is used to evaluate XADC. You can browse through various tabs that display various performance parameters of XADC.

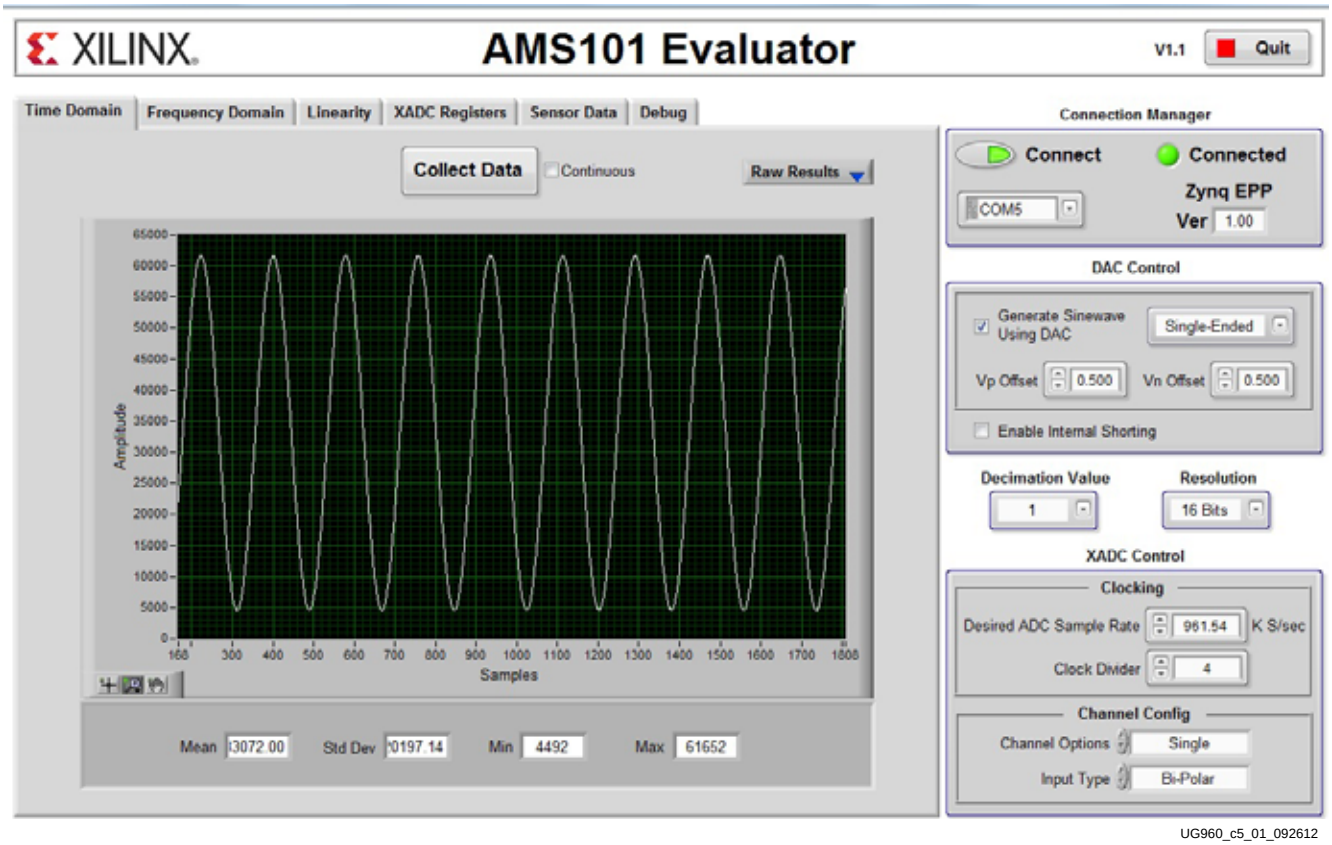


Figure 5-1: **Time Domain Tab of LabVIEW GUI**

XILINX

Appendix A

ADC Basics

An Analog-to-Digital Converter (ADC) converts an analog signal to a binary number (fraction V_{IN}/V_{REF} where V_{IN} is the input voltage and V_{REF} is the reference voltage of ADC). The output specifies what fraction the analog input is of the analog reference.

The precision or accuracy of this conversion is defined by the resolution (bits) of an ADC. For example, an ADC with 12-bit wide output word can represent a number between 0 – 4095 ($2^{12} - 1$). Thus, the smallest fraction that can be represented by the 12-bit output is 1/4096 and the only other fractional outputs that can be represented by the ADC are integral multiples of this.

Figure A-1 depicts the transfer function of a 3-bit ADC. The Least Significant Bit (LSB or the code width) is defined in Equation A-1.

$$1LSB = V_{REF}/2^N \quad \text{Equation A-1}$$

where,

V_{REF} is the reference voltage of ADC

N is ADC resolution

The output digital codes have an inherent quantization error of +/- ½ LSB, i.e., the quantized digital code represents an analog voltage that can be anywhere within +/- ½ LSB from the mid-point between adjacent digital codes.

The transfer function of ADC would then be defined by Equation A-2.

$$\text{ADC Code} = 2^N \times (V_{IN}/V_{REF}) \quad \text{Equation A-2}$$

The transfer function is similar to a staircase where each tread represents a particular digital output code and each riser represents a transition between two adjacent codes.

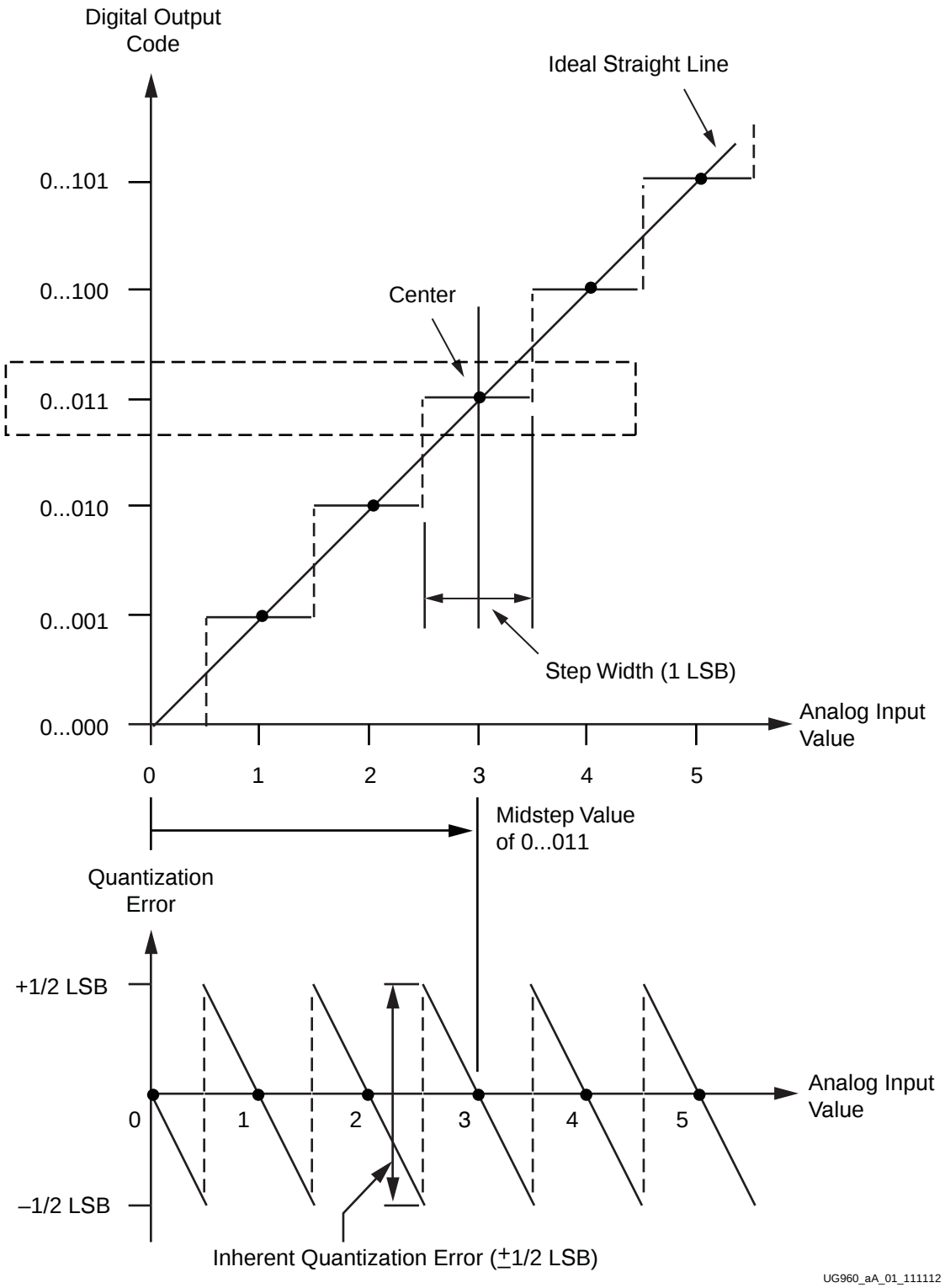


Figure A-1: **ADC Transfer Function and Quantization Error**

Register Descriptions

This section defines the register-based communication between the LabVIEW GUI and the hardware design.

The address limit would be 16 bits and the data value would be 16 bits.

For Writes (16 bits of address; 16 bits of write data), OK acknowledgment is provided:

```
W AAAA DDDD      //- Sent data
OK               //- Hardware response
```

For Reads, R AAAA is followed by read data.

See Table B-1 for register map information.

Table B-1: **Register Map**

GUI Commands/Clicks	UART Transmission		UART Reception
	Address	Data	
Connection Establishment: Design version register (read only)	0x0000		Read-only design version register. Returns 16-bits of data. Design version (D11-8).(D7-0) D15-12 indicates the device family, because the same GUI can be used for designs on different families. 0001- AC701 0010 - KC705 0011 - VC707 0100 - ZC702 For the Artix®-7 FPGA, Kintex®-7 FPGA, Virtex®-7 FPGA, and Zynq®-7000 AP SoC reference designs, it returns 0x1110, 0x2120, 0x3120, and 0x4120, respectively. UART Command: R 0000
Time Domain/Frequency Domain Tab: Collect data (read only)	0x0001	—	FPGA sends back 4096 samples (8K bytes) of raw data for further processing by LabVIEW GUI. UART Command: R 0001
Linearity Tab: Collect histogram data (read only)	0x0002	—	FPGA sends back 4096 bytes of raw data to be used for INL/DNL plots. UART Command: R 0002

Table B-1: **Register Map *(Cont'd)***

GUI Commands/Clicks	UART Transmission		UART Reception
	Address	**Data**	
Time Domain/Frequency Domain Tab: DAC voltage (user-programmed)	0x0003 (for DAC A) 0x0013 (for DAC B)	Value	Provide user-entered voltage value: DDDD = 0x1999 when user programs 100 mV = 0x3332 when user programs 200 mV = 0x4CCB when user programs 300 mV = 0x6664 when user programs 400 mV = 0x7FFD when user programs 500 mV = 0x9996 when user programs 600 mV = 0xB32F when user programs 700 mV = 0xCCC8 when user programs 800 mV = 0xE661 when user programs 900 mV = 0xFFFF when user programs 1V UART Command: W 0003 DDDD
Time Domain/Frequency Domain Tab: Sinusoid generation (write only)	0x0004	0x0000	When DAC is selected as sinusoidal source on either of the tabs. UART Command: W 0004 DDDD
Sensor Tab: Collect sensor data (read only)	0x0005	—	FPGA sends back 8B (4 raw samples) each for temperature, V_{CCAUX}, V_{CCINT}, V_{CCBRAM}. UART Command: R 0005
Time/Frequency Domain Tab: Collect V_{REFN}/short data	0x0006	—	FPGA sends back 4096 samples (8KB) of raw data. UART Command: R 0006
Decimation Value	0x000A	Value	Provide the decimation value selected by user. Only 1, 2, 4, 8, and 16 are allowed. UART Command: W 000A DDDD
Update XADC			
Clocking selection (write only)	0x0007	Value	This is the clock divider value (allowed 2 - 255). UART Command: W 0007 DDDD
Channel option (write only)	0x0008	Value	0x0001 - Single channel 0x0002 - Simultaneous sampling mode UART Command: W 0008 DDDD
Input type (write only)	0x0009	Value	0x0001 - Unipolar input 0x0002 - Bipolar input UART Command: W 0009 DDDD

Table B-1: **Register Map *(Cont'd)***

GUI Commands/Clicks	UART Transmission		UART Reception
	Address	Data	
Read XADC Registers			
Read any valid XADC register	0x0100 + Reg offset	0x00	Registers to be read to be user- selectable based on check box or some kind of indicator. UART Command: R 01xx 16-bit value of the register as read is returned.
Debug register	0x0200	0x0001	This enables printing of debug messages at various points which are otherwise suppressed. UART Command: W 0200 0001
Update UCD9248			
- V_{CCAUX}	0x3410	Value	UART Command: W 3410 DDDD
- V_{CCINT}	0x3400	Value	UART Command: W 3400 DDDD
- V_{CCBRAM}	0x3610	Value	UART Command: W 3610 DDDD

The address scheme for PMBus is laid out as follows.

A[31:16] indicates the device address; A[15:8] indicates page and A[7:0] indicates Voltage or Current access (0 – voltage; 1 – current; 2 – V_{OUT} MAX (reserved for future use)).

It is required to make sure that programmed voltage values are within these allowed ranges:

$$0.9 \leq Vccint \leq 1.0$$

$$1.7 \leq Vccaux \leq 1.8$$

$$0.9 \leq Vccbram \leq 1.0$$

Appendix C

Directory Structure and File Descriptions

The directory structure of the design is depicted in Figure C-1.

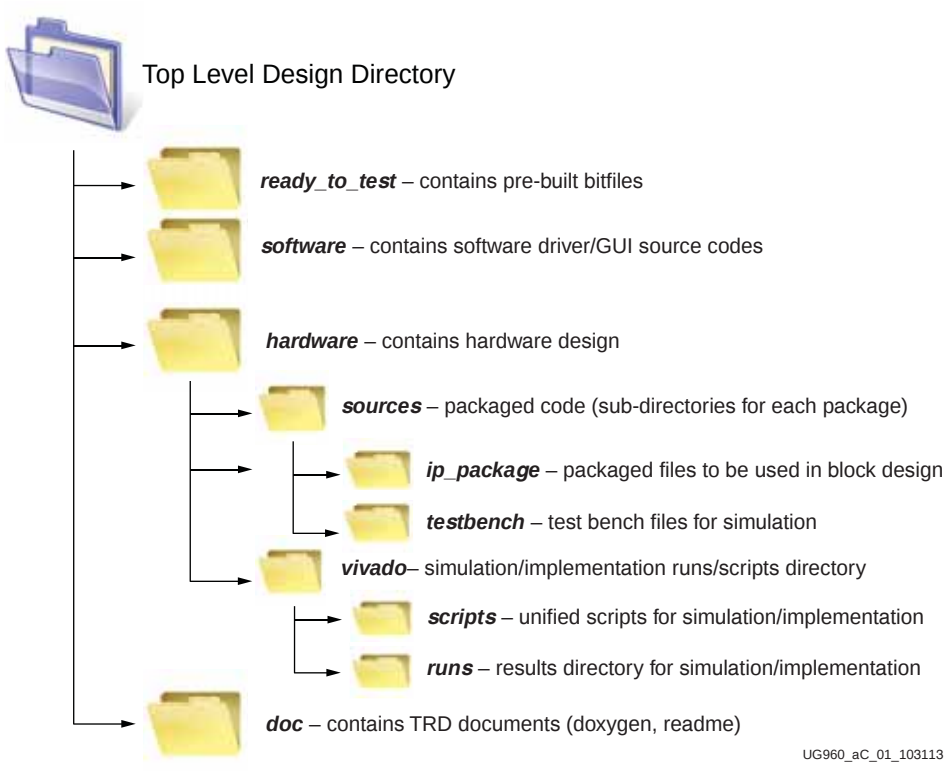


Figure C-1: **Directory Structure**

- `hardware/sources`—This is the hardware design directory consisting of the XPS hardware specification file, XDC, and the custom pcores used in the design. The hardware can be rebuilt using these files.
- `vivado`—This directory contains the Vivado tool scripts under the `scripts` directory. The `runs` directory contains the results
- `software`—This is the software design directory consisting of the *SDK Export* area from XPS project, software source, board support package, and associated hardware platform specification. The SDK workspace can be rebuilt by importing these on SDK invocation.

- `ready_to_test`—This directory provides a design that can be tested out of the box. This directory contains the design bitfile, ELF for software, and a Tcl script for programming the FPGA, downloading the ELF, and starting the MicroBlaze™ processor.
- `readme.txt`—This is the `readme` file for the design, providing information on versions, requirements, known issues, and so on.

XILINX®

Appendix D

Troubleshooting

Table D-1 provides information on troubleshooting the design in case it does not work as expected.

Table D-1: **Suggested Corrective Actions**

Issue	Suggested Resolution
LabVIEW GUI does not open up.	Check if the recommended version of the LabView Run-Time engine is installed.
LabVIEW GUI does not seem to be able to connect to hardware.	Check in the Device Manager under COM port connections if the **`Silicon Labs CP210x USB to UART Bridge`** shows up. Check if the COM port selected in the LabVIEW GUI is the same as shown in Device Manager.

XILINX

Appendix E

Additional Resources

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see the Xilinx Support website.

For continual updates, add the Answer Record to your myAlerts.

For definitions and terms, see the Xilinx Glossary.

Solution Centers

See the Xilinx Solution Centers for support on devices, software tools, and intellectual property at all stages of the design cycle. Topics include design assistance, advisories, and troubleshooting tips.

References

The most up to date information related to the 7 series FPGA AMS Targeted Reference Design and its documentation is available on these websites.

AMS101 Evaluation Card

AMS101 Evaluation Card documentation

AMS101 Evaluation Card Master Answer Record (AR 52165)

Analog Mixed Signal

AMS101 Instructor-led Training and Online Training

Artix-7 FPGA AC701 Evaluation Kit

Artix-7 FPGA AC701 Evaluation Kit documentation

Artix-7 FPGA AC701 Evaluation Kit Master Answer Record (AR 51900)

Kintex-7 FPGA KC705 Evaluation Kit

Kintex-7 FPGA KC705 Evaluation Kit documentation

Kintex-7 FPGA KC705 Evaluation Kit Master Answer Record (AR 45934)

Virtex-7 FPGA VC707 Evaluation Kit

Virtex-7 FPGA VC707 Evaluation Kit documentation

Virtex-7 FPGA VC707 Evaluation Kit Master Answer Record (AR 45382)

Zynq-7000 All Programmable SoC ZC702 Evaluation Kit

Zynq-7000 All Programmable SoC ZC702 Evaluation Kit documentation

Zynq-7000 All Programmable SoC ZC702 Evaluation Kit Master Answer Record (AR 47864)

These Xilinx documents provide supplemental material useful with this guide:

1. National Instruments
   - National Instruments: LabVIEW Run-Time Engine 2011 (32-bit Standard RTE)
   - National Instruments: LabVIEW Resources to Help You Get Started
2. *KC705 Evaluation Board for the Kintex-7 FPGA User Guide* (UG810)
3. *VC707 Evaluation Board for the Virtex-7 FPGA User Guide* (UG885)
4. *AC701 Evaluation Board for the Artix-7 FPGA User Guide* (UG952)
5. *ZC702 Evaluation Board for the Zynq-7000 XC7Z020 All Programmable SoC User Guide* (UG850)
6. *Kintex-7 FPGA KC705 Evaluation Kit Getting Started Guide (Vivado Design Suite)* (UG883)
7. *Getting Started with the Virtex-7 FPGA VC707 Evaluation Kit* (UG848)
8. *Zynq-7000 All Programmable SoC: ZC702 Evaluation Kit and Video and Imaging Kit Getting Started Guide (Vivado Design Suite)* (UG926)
9. *Artix-7 FPGA AC701 Evaluation Kit Getting Started Guide (Vivado Design Suite)* (UG967)
10. *LogiCORE IP AXI UART Lite Product Guide for Vivado Design Suite* (PG142)
11. *LogiCORE IP AXI Serial Peripheral Interface (AXI SPI)* (DS742)
12. *LogiCORE IP AXI Block RAM (BRAM) Controller Product Guide* (PG078)
13. *LogiCORE IP AXI GPIO Product Guide for Vivado Design Suite* (PG144)
14. *LogiCORE IP AXI IIC Bus Interface Product Guide* (PG090)
15. *LogiCORE IP AXI Interrupt Controller (INTC) Product Guide* (PG099)
16. *7 Series FPGAs and Zynq-7000 All Programmable SoC XADC Dual 12-Bit 1 MSPS Analog-to-Digital Converter User Guide* (UG480)
17. *AC701 Base Targeted Reference Design User Guide (Vivado Design Suite)* (UG964)
18. *AMS101 Evaluation Card User Guide* (UG886)
19. Texas Instruments: *UCD92xx Digital PWM System Controller PMBus Command Reference*

XILINX

Appendix F

Regulatory and Compliance Information

This product is designed and tested to conform to the European Union directives and standards described in this section.

Declaration of Conformity

See the AMS101 Evaluation Card CE Declaration of Conformity.

Directives

2006/95/EC, *Low Voltage Directive (LVD)*

2004/108/EC, *Electromagnetic Compatibility (EMC) Directive*

Standards

EN standards are maintained by the European Committee for Electrotechnical Standardization (CENELEC). IEC standards are maintained by the International Electrotechnical Commission (IEC).

Electromagnetic Compatibility

EN 55022:2010, *Information Technology Equipment Radio Disturbance Characteristics – Limits and Methods of Measurement*

EN 55024:2010, *Information Technology Equipment Immunity Characteristics – Limits and Methods of Measurement*

This is a Class A product. In a domestic environment, this product can cause radio interference, in which case the user might be required to take adequate measures.

Safety

IEC 60950-1:2005, *Information technology equipment – Safety, Part 1: General requirements*

EN 60950-1:2006, *Information technology equipment – Safety, Part 1: General requirements*

Markings

This product complies with Directive 2002/96/EC on waste electrical and electronic equipment (WEEE). The affixed product label indicates that the user must not discard this electrical or electronic product in domestic household waste.

This product complies with Directive 2002/95/EC on the restriction of hazardous substances (RoHS) in electrical and electronic equipment.

This product complies with CE Directives 2006/95/EC, *Low Voltage Directive (LVD)* and 2004/108/EC, *Electromagnetic Compatibility (EMC) Directive.*

XILINX®

Appendix G

Warranty

THIS LIMITED WARRANTY applies solely to standard hardware development boards and standard hardware programming cables manufactured by or on behalf of Xilinx ("Development Systems"). Subject to the limitations herein, Xilinx warrants that Development Systems, when delivered by Xilinx or its authorized distributor, for ninety (90) days following the delivery date, will be free from defects in material and workmanship and will substantially conform to Xilinx publicly available specifications for such products in effect at the time of delivery. This limited warranty excludes:
(i) engineering samples or beta versions of Development Systems (which are provided "AS IS" without warranty); (ii) design defects or errors known as "errata";
(iii) Development Systems procured through unauthorized third parties; and
(iv) Development Systems that have been subject to misuse, mishandling, accident, alteration, neglect, unauthorized repair or installation. Furthermore, this limited warranty shall not apply to the use of covered products in an application or environment that is not within Xilinx specifications or in the event of any act, error, neglect or default of Customer. For any breach by Xilinx of this limited warranty, the exclusive remedy of Customer and the sole liability of Xilinx shall be, at the option of Xilinx, to replace or repair the affected products, or to refund to Customer the price of the affected products. The availability of replacement products is subject to product discontinuation policies at Xilinx. Customer may not return product without first obtaining a customer return material authorization (RMA) number from Xilinx.

THE WARRANTIES SET FORTH HEREIN ARE EXCLUSIVE. XILINX DISCLAIMS ALL OTHER WARRANTIES, WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT, AND ANY WARRANTY THAT MAY ARISE FROM COURSE OF DEALING, COURSE OF PERFORMANCE, OR USAGE OF TRADE. (2008.10)

Do not throw Xilinx products marked with the "crossed out wheeled bin" in the trash. Directive 2002/96/EC on waste electrical and electronic equipment (WEEE) requires the separate collection of WEEE. Your cooperation is essential in ensuring the proper management of WEEE and the protection of the environment and human health from potential effects arising from the presence of hazardous substances in WEEE. Return the marked products to Xilinx for proper disposal. Further information and instructions for free-of-charge return available at: www.xilinx.com\ehs\weee.htm.