



# UART RS-232 Maximum Baud Rate Design Example Tutorial

Date: 1 December 2016

Revision:1.0

©2015 Altera Corporation. All rights reserved. ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS and STRATIX words and logos are trademarks of Altera Corporation and registered in the U.S. Patent and Trademark Office and in other countries. All other words and logos identified as trademarks or service marks are the property of their respective holders as described at [www.altera.com/common/legal.html](http://www.altera.com/common/legal.html). Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

## Table of Contents

1	Introduction	3
2	Prerequisite	3
3	Requirements	3
4	Design Example Files	3
5	Block Diagram	4
6	Experimental Setup	6
	6.1 Generate programming files	6
	6.2 Program Hardware Image into FPGA	6
	6.3 Run the design	7
7	Revision History	9

## 1. Introduction

This example is a test functionality for UART RS-232 Serial Port IP which contains a NIOS® II processor and Dual UART RS-232 IP. The design example implements a basic UART RS-232 functionality of Variable Baud Rate On real-time basis. This means the developer can set the required Baud Rate of data transfer from NIOS II Application. Furthermore, this design demonstrates a standard method of developing a UART Application with NIOS II.

## 2. Prerequisite

You are required to have knowledge in instantiating and developing a system with a Nios II processor. Altera recommends that you go through the online tutorials and training materials provided on Altera's website before using this design example.

Related information:

- [Nios II Gen2 Software Developer's Handbook](#)
- [Embedded Peripherals IP User Guide](#)

## 3. Requirements

The hardware and software requirements for the design example are:

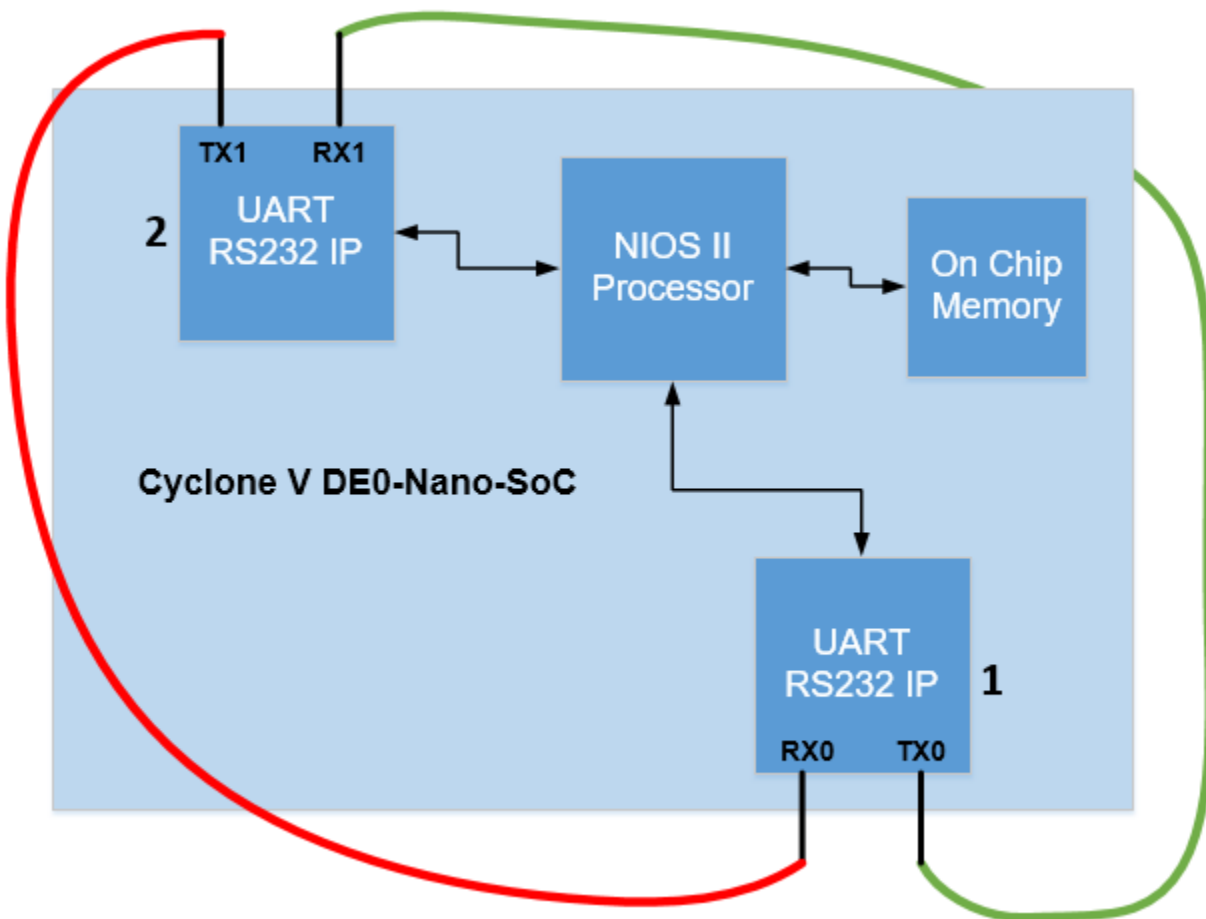
- Cyclone V DE0-Nano SoC board.
- Altera Quartus Prime version 16.1 with Nios II EDS.
- USB cable for programming the device.
- Two Jumper wire (female).

## 4. Design Example Files

This design example consists of hardware image and Nios II software. The Application hardware and software images include simple design to run the Dual UART application.

File Name	Description
Output_files/uart.sof	Main Hardware image of the design
Software/software.zip	The software files of NIOS II application used in this example
Top.v	This is the top-level file of the design
Uart.qsys	Qsys file of the NIOS II system

## 5. Block Diagram



**Figure 1: UART RS232 Design Diagram**

Figure 1 shows the various IP blocks within the design example. Details of the connection between each IP blocks can be found in the included Qsys file for this design example. From Figure 1, three main steps can be summarized:

- 1- The UART 1 will send some random characters through TX0.
- 2- The UART 2 will receive the sent characters from UART 1 through RX1.
- 3- NIOS II processor will print out some data on console.

- 4- The colored lines in Figure 1 represent the jumper wires (female) that connect GPIO 0 Pin 1 with GPIO 1 Pin 1. Kindly refer to [Cyclone V DE0-Nano-SOC user guide](#) (Page 26). Figure 3-18 from this document can be seen in Figure 2.

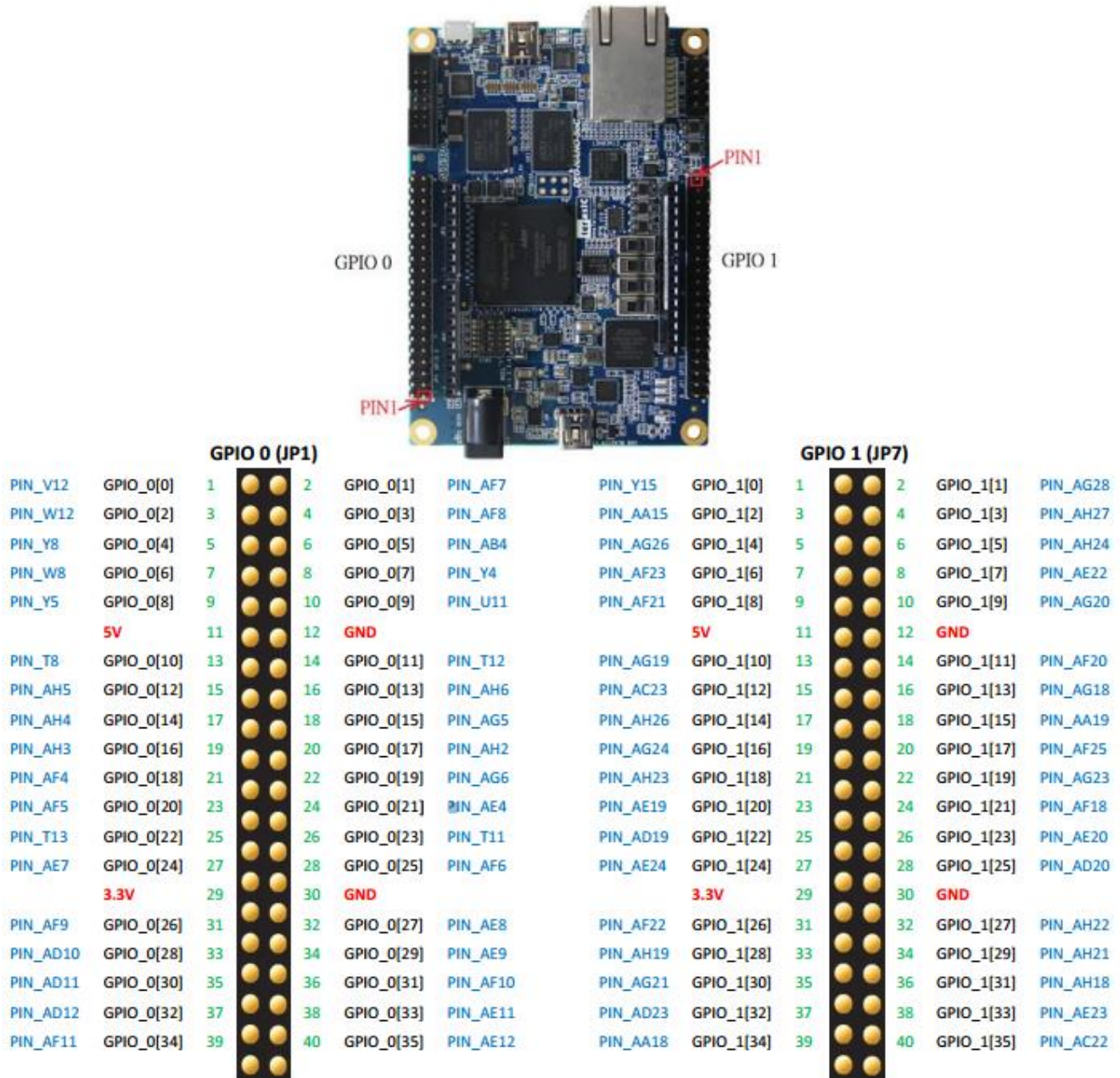


Figure 2: CV DE0-NANO-SOC GPIO Pin Arrangement (originally from [here](#))

## 6. Experimental Setup

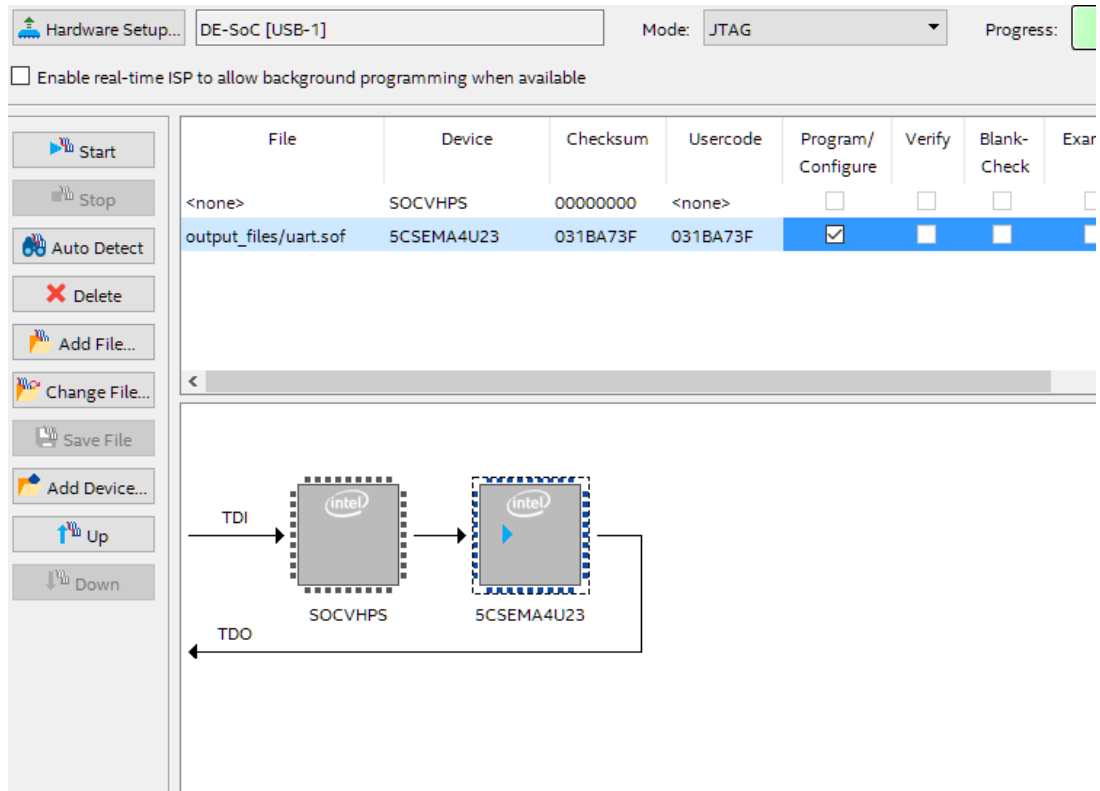
### 6.1 Generate Programming Files

First, follow the steps on Design Store webpage to prepare the design template in Quartus Prime software. The master files are provided together with the design example. You may proceed to program the FPGA run the design without any recompilation or programming file generation. The default programming file “output\_files\uart.sof” can be used to configure the FPGA with the hardware image of this design. If you intend to modify the design, you need to generate qsys and compile the design based on your changes.

### 6.2 Program Hardware Image Into FPGA

To program the hardware image and Nios II firmware into the board, perform the following steps:

1. Connect the USB Blaster cable from PC to the on-board USB Blaster II connector and power up the board.
2. Open the “Programmer” in Quartus Prime software, then click “Hardware Setup...” and select “USB-BlasterII”.
3. Click “Auto Detect” and select “5CSEMA4”.
4. Right-click on the selected device “5CSEMA4” and select “Change File”.
5. Open the programming file “output\_files \uart.sof”.
6. Check the “Program/Configure” checkbox for “5CSEMA4”.
7. Click “Start” to start programming.
8. Make sure the download is successful.
9. Power cycle the board.



**Figure 3: Programmer window**

### 6.3 Run The Design

To setup the environment for Nios II software modification and recompilation, perform the following steps:

1. Extract archive files “software\software.zip” to folder “software\UART\_RXTX” and “software\ UART\_RXTX\_bsp”.
2. Open the “Nios II Software Build Tools for Eclipse” from “Tools” menu in Quartus Prime software.
3. Go to “File > Import...”, then select “Import Nios II Software Build Tools Project” in “Nios II Software Build Tools Project” folder and click Next.
4. Browse to “software\ UART\_RXTX” folder in “Project location” field.
5. Enter “UART\_RXTX” in “Project name” field, then click “Finish”.
6. Repeat step 3 to step 5, but change the “Project location” to “software\ UART\_RXTX\_bsp” folder and change the “Project name” to “UART\_RXTX\_bsp”.
7. Repeat steps 2 to 6 for Application\_1 and Application\_2 folders.
8. You should see the “UART\_RXTX” and “UART\_RXTX\_bsp” directory in “Project Explorer” as shown in Figure 4. The environment is now ready for modification.

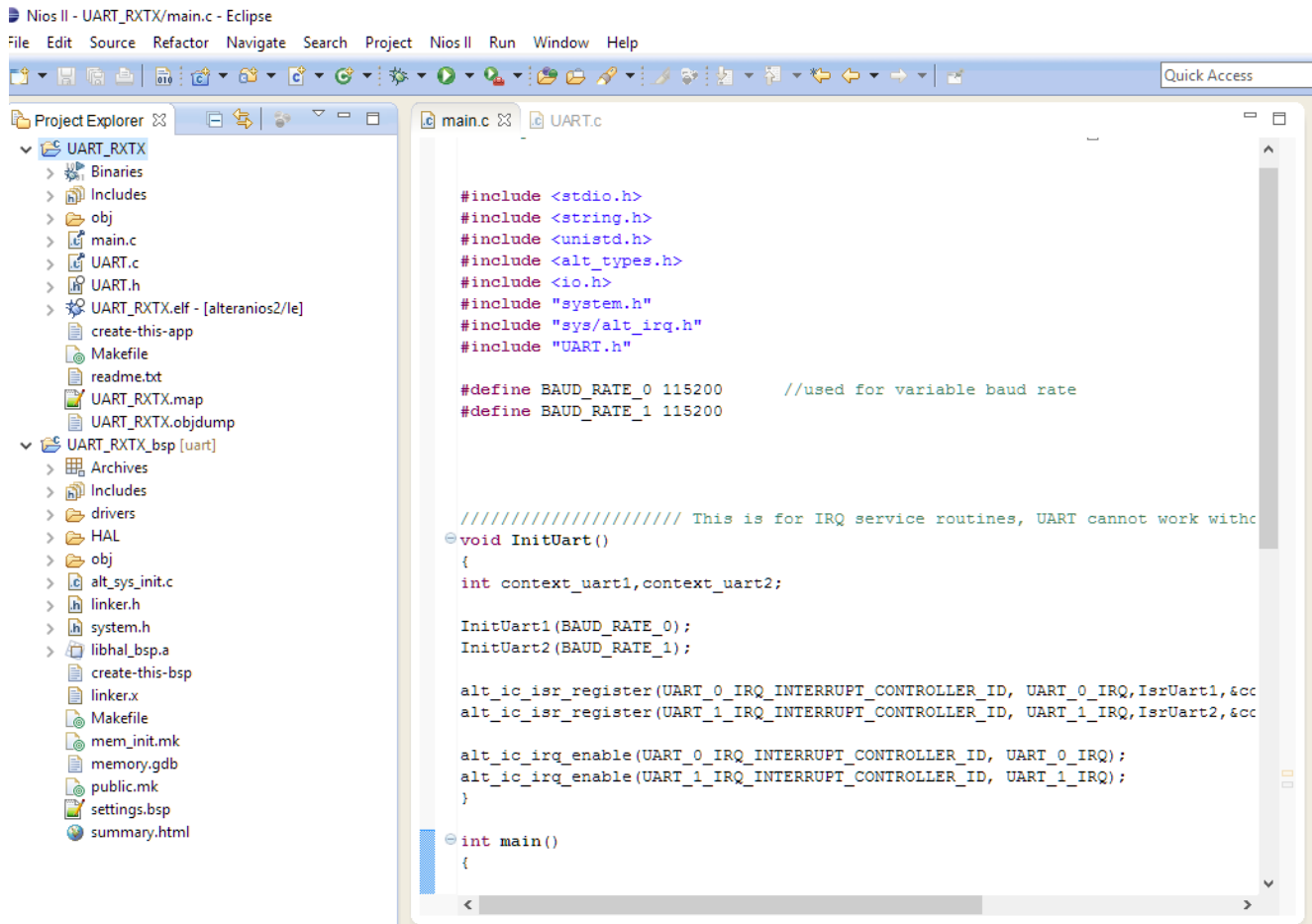


Figure 4: Project Explorer window

The software application of this design contains two main files:

1- **Main.c**

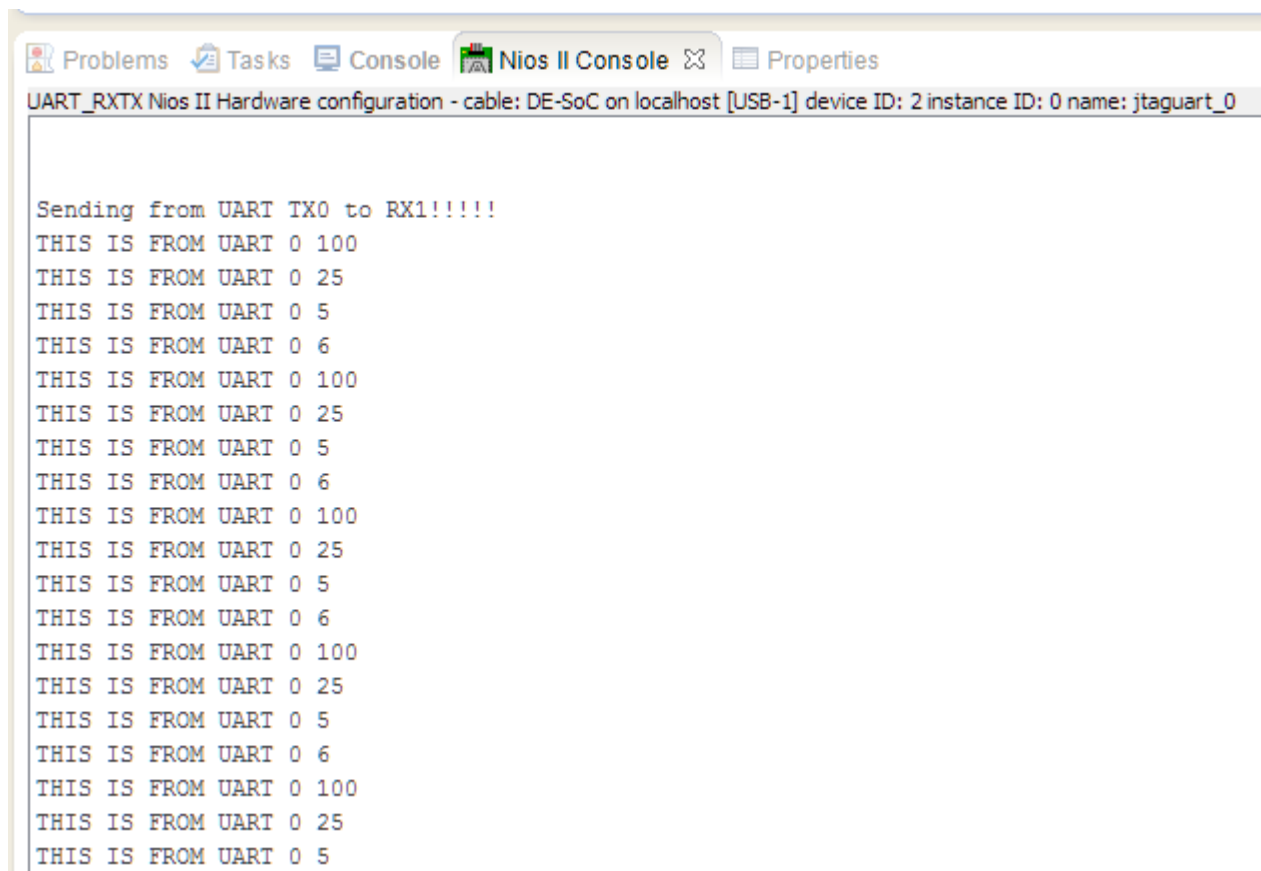
This is the main C file of the application. It contains the random data generation, send, and print out.

2- **UART.c**

This is the main UART functions for read/write operations. This file contains RX and TX buffers, head and tail for data transfer, driver initialization, Interrupt Service Routine (ISR) for UART, data transmission and collection, and registers check procedures.



After you build the UART\_RXTX application, you can download the .elf file. Please make sure your FPGA is already configured with the uart.sof file, check Figure 2. Once the application runs, you will notice few messages come out from NIOS II eclipse console. These messages to show the correct reception of characters from UART 0 TX0 to UART 1 RX1, as seen in Figure 5.



The screenshot shows the Eclipse IDE interface with the 'Nios II Console' tab active. The console title bar reads 'UART\_RXTX Nios II Hardware configuration - cable: DE-SoC on localhost [USB-1] device ID: 2 instance ID: 0 name: jtaguart\_0'. The console output is as follows:

```
Sending from UART TX0 to RX1!!!!  
THIS IS FROM UART 0 100  
THIS IS FROM UART 0 25  
THIS IS FROM UART 0 5  
THIS IS FROM UART 0 6  
THIS IS FROM UART 0 100  
THIS IS FROM UART 0 25  
THIS IS FROM UART 0 5  
THIS IS FROM UART 0 6  
THIS IS FROM UART 0 100  
THIS IS FROM UART 0 25  
THIS IS FROM UART 0 5  
THIS IS FROM UART 0 6  
THIS IS FROM UART 0 100  
THIS IS FROM UART 0 25  
THIS IS FROM UART 0 5  
THIS IS FROM UART 0 6  
THIS IS FROM UART 0 100  
THIS IS FROM UART 0 25  
THIS IS FROM UART 0 5
```

**Figure 5: NIOS II eclipse console view**

## 7. Revision History

Date	Revision	Description
10 December 2016	1.0	Initial Release