SCB-MBP Stress Test
User Guide

Version: v0.2

User guide for Minibackplane-SCB stress test.

Dissemination Level:	**Open with LGPL license**
Project:	**Open hardware**
Distribution level:	**PP (project public)**
Developer:	**Miguel Méndez, Seven Solutions** mmendez@sevensols.com

Project website address: http://www.ohwr.org/projects/wr-switch-testing

DOCUMENT INFORMATION

No. and name:	MBP- SCB Stress Test by Seven Solutions				
Status	Final	**Due**		**Date:**	06/08/2012
Revision:	0.2				
Author:	Miguel Méndez Macías				

DIFFUSION LIST

Name	Affiliation	Name	Affiliation
All			

APPROVALS

	Name	Affiliation	Date	Visa
Authors and Project Leader	Miguel Méndez	Seven Solutions	03/08/2012	√

DOCUMENT HISTORY

Rev #	Date	Author name / Affiliation	Description
0.0	25/06/2012	Miguel Mendez	First draft for review
0.1	15/07/2012	Gabriel Ramirez	Minor changes
0.2	06/08/2012	Miguel Mendez	Project modifications and document review

Dissemination level		
PU	Public with a LGPL license	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

TABLE OF CONTENTS

Table of Diagrams and Figures

Table of Tables

No table of figures entries found.

1. ACRONYMS LIST

OHWR	Open Hardware Repository
WB	Wishbone
7S	Seven Solutions S.L.
SCB	Switch Core Board
MBP	MiniBackPlane Board
PCB	Printed Circuit Board
FPGA	Field-Programmable Gate Array
SRAM	Static Random-Access Memory
QDRII	Quad Data Rate II
HDL	Hardware Description Language
VHDL	IEEE standard HDL: “VHSIC HDL”
Verilog	IEEE standard HDL
ISE	Xilinx tool for FPGA design solutions
GPIO	General Purpose Input/Output
OS	Operating System
MMU	Memory Management Unit
MPU	Memory Protection Unit

2. ABOUT THE GUIDE

The MBP-SCB Stress test user guide provides information about the implementation, customization and simulation of the stress test project developed to evaluate power consume, temperature range as well as peripherals test.

Since not all the HDL can be uploaded in the repository due to copyrights and exclusive licenses, some textual steps and information are added in order to fulfil the implementation process and achieve a better understanding of the project developed.

This document and the associate project are developed by Seven Solutions in the frame of Open Hardware with LGPL license.

3. INTRODUCTION

The project “StressTestMBP_SCB” is provided to test and stress the SCB and Minibackplane board in order to evaluate the power consume and temperature range that the WR switch can suffer. At the same time, it evaluates and tests the main SCB peripherals (PLL, QDRII memory, serial ports,…) and some Minibackplane peripherals (LEDS,I2C, FPGA-USB,…).

This document provides a brief description of the project implemented: from the HW introduced and how it is used until SW code implemented inside a softcore called Microblaze from Xilinx. This project is developed with the Xilinx tool packet (EDK, SDK and ISE) and it uses specific IP-cores developed to configure and control the peripherals and some commercial IP-core provided by Xilinx that control standard protocols like UART, I2C, System Monitor,…

This test is completely isolate from the ARM so all the information is shown through the serial USB provided by the Minibackplane.

A summary of folders and files is also detailed in the next section in order to make easier to find and understand the project.

Warning: This implementation is designed to stress peripherals and increase board temperature, therefore it can be harmful if it is not used with a FAN or is running long time without supervision.

4. FILES AND FOLDER STRUCTURE OVERVIEW

You can checkout files with the following command:
svn co http://www.ohwr.org/projects/wr-switch-testing

The following points describe how the folder and files are organized and what they include:

- ***StressTestMBP_SCB*** (Project folder). It includes a compressed file with the EDK project, all configuration files and specific IP cores implemented for this design, so it can be used and modified for the user whenever it is necessary. This project needs the Xilinx tool packet to be opened: EDK tool for HW modifications and SDK tool for software modifications (it is possible to not modify HW (“system.bit” file) and work just with SDK to change the software part).

- ***Test file*** (Test folder). It includes the .BIT file ready to be dumped into the FPGA. All HW and SW are introduced inside this file so the project will start sending data through the USB-UART (Minibackplane FPGA-USB connection) as soon as the FPGA is programmed.

- ***Doc*** (Document folder). It includes the documents related to this design.

These folders are located in: ***/hdl/StressTest/***

5. PROYECT DESIGN AND DESCRIPTION

This design is divided in two different parts: the HW part where the peripheral controllers, stress tests and architecture (bus, softcore) are implemented; and the SW part which controls and enables the peripherals depending upon the sensors data received. This SW is responsible of sending this information though the UART in order to be shown by the user. The following sections describe each part of the design.

5.1 Hardware design

The next Figure shown the HW design and the controllers developed for this project. The following paragraphs will provide a brief description of each one:

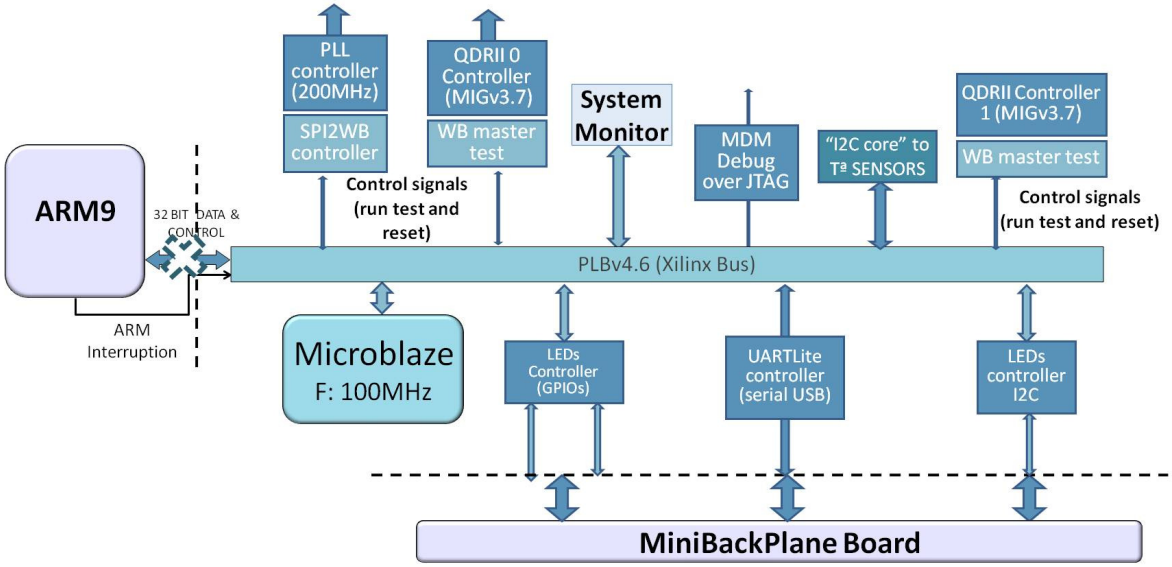

Figure 1: HW block diagram (StressTest design)

First it is important to mention that this design is completely isolate from the ARM therefore all test information is shown through the serial USB provided by the Minibackplane. Nevertheless, this design is implemented in the way that the ARM can program the FPGA through the FPGA serial configuration.

- Microblaze processor, PLB bus and MDM IPcore: The architecture is developed with the Xilinx tools chain so the structure is based on Xilinx IP-cores. The PLB bus is a 32bits Xilinx proprietary bus used for this design, no special bandwidth is needed for this design. Microblaze is the soft-processor provided by Xilinx and, in this design, it is implemented with a few features: 100MHz frequency clock, 32KB internal memory, NO cache, NO floating point,… This processor is used to control and reset the external peripherals in case that it is needed and to provide the sensors information to the user. (see SW section 5.2). The MDM controller is used for debugging purposes.

- PLL controller: This controller is a custom IP-core developed by Seven Solutions and based on the WB2SPI core provided by OHWR repository. The PLL peripheral is control physically by a serial bus SPI but a specific register configuration have to be done, therefore we implemented a IP-core with a master wishbone which writes the specific data register and, after configuration, the peripheral registers value is read and checked the correct configuration. As we said, the wishbone master implemented is connected directly to the wishbone slave provided by the WB2SPI core mention before. This controller is enabled or disabled by the processor.
- QDRII controller: This controller is a custom IP-core developed by Seven Solutions and based on the QDRII2WB core provided by OHWR repository. The QDRII peripheral is controlled by two wishbone slaves but a specific design is implemented to test and stress both channels and QDRIIs. Therefore we implemented an IP-core with two wishbone masters in each QDRII which write the data into all memory words and, afterward, read these memory words value to check the correct written. As we said, these masters implemented are connected directly to the QDRII2WB core mention before. This controller is enabled or disabled by the processor and, when it is enabled, an infinite loop of writing and reading is done until it is disabled. Each QDRII has its own independent controller, so this core is replicated into the design.
- System Monitor: IPcore provides by Xilinx which gives information about the FPGA on-chip temperature and FPGA voltage levels (internal voltage and auxiliary voltage).
- I2C controller: This IPcore is provided by Xilinx and it is used to control 2 different peripherals in this design:
    - First instantiation of this core is done to control the temperature sensors provided by the SCB board. Depending on the SCB version we can have three or four temperature sensors around the board: PLL and LDO sensor; Power Supply sensor (in the last version this part of the board has two sensors: left and right); and sensor below FPGA.
    - The second instantiation of the I2C controller is used to control the MiniBackplane LEDs of each SFP. Different modes and colours can be chosen.

- UART controller: this IPcore is used to control the serial USB provided by the Minibackplane. All information about the test, temperature and process is shown through this peripheral. This is the correct configuration of your hyperterminal to see this information:
        - 115200 baud speed.
        - 8 data bits.
        - No parity.
        - No control.

Warning: This implementation is designed to stress peripherals and increase board temperature, therefore it can be harmful if it is not used with a FAN or is running long time without supervision.

5.2 Software design

The hardware described above is implemented to be controlled and enabled by this software but this SW do not send or receive from/to the QDRII neither configure the PLL registers since that is done in HW. This software is used just to enable the peripheral tests and data flow (enabled the HW IP-core) implemented in hardware. Nevertheless the interface with the user is controlled by the software to provide information through UART or switch on/off MBP LEDs or peripheral tests. These are the main test done in software:

- First it is tested the correct configuration of the USB-UART peripheral printing a message which can be used to check the correctness of the controller and peripheral as well as the HyperTerminal user configuration.
- Second test is the MBP LEDs controlled by I2C serial bus. During 2/3 seconds it switches on and off all LEDs in their different colours and modes. A small function in SW to control the I2C core is developed.
- Afterward, the PLL and LDO controller is activated and tested, which means that it provides the clocks needed for the QDRII memories (187.5MHz) and the Virtex-6 GTX (125MHz).
- When the PLL provides the clocks needed for both QDRII is time to start the QDRII test. In software it is possible to enable or reset the QDRII test in case that it is needed. Information about the number of iterations are provided (each iteration corresponds to write and read all memory space) as well as errors found in those iteration.
- Finally an infinite loop is created. It is executed each 3/4 seconds. All the measures are shown though the USB-UART from the Minibackplane board. These are the measurement done:
    - It shows the QDRII test, number of iterations and errors.
    - FPGA on-chip actual temperature and maximum temperature reached.
    - FPGA internal and auxiliary voltage.
    - Board temperature from the different sensors (Below FPGA sensor, PLL & LDO sensor and Power supply sensor)

  When 75 degrees are reached inside the FPGA a warning message is shown and the test should be finished.
  This infinite loop also switches on and off the GPIO LEDs from the Minibackplane each time an iteration is completed. To do a visual check, the three last LEDs are switched on when the PLL and each QDRII run properly (a reset state of these cores will switch off the LEDs).

6. HOW TO USE THIS TEST

These are the steps user should follow to implement and run this test correctly:

1. In order to show the test information, a connection between the Minibackplane and the PC through the MBP USB should be done. Specific drivers for this serial USB should be downloaded.
2. Open your HyperTerminal with the features describe in the hardware section 5.2. (115200,8bits,No parity,1bit, No control)
3. Program the FPGA with the .BIT file provided.
4. Information about the test peripherals, board temperature and voltage is shown in our HyperTerminal. Once the temperature goes up too much a warning is shown and the test shown be finished switching off the board.

Warning: This implementation is designed to stress peripherals and increase board temperature, therefore it can be harmful if it is not used with a FAN or is running long time without supervision.