Core1553BBC

User Guide

For more information about Actel's products, call 888-99-ACTEL
or visit our Web site at http://www.actel.com

Actel Corporation • 955 East Arques Avenue • Sunnyvale, CA USA 94086
U.S. Toll Free Line: 888-99-ACTEL • Customer Service: 408-739-1010 • Customer Service FAX: 408-522-8044
Customer Applications Center: 800-262-1060 • Customer Applications FAX: 408-739-1540

Actel Europe Ltd. • Dunlop House, Riverside Way • Camberley, Surrey GU15 3YL • Tel: +44 (0)1276.401452 • Fax: +44 (0)1276.401490

Actel Japan • EXOS Ebisu Bldg. 4F • 1-24-14 Ebisu Shibuya-ku • Toyko 150 • Japan
Tel: +81 (0)334-457-671 Fax: +81 (0)334-457-668

502-00-003-0

Core1553BBC

User Guide

Actel Corporation, Sunnyvale, CA 94086

Printed in the United States of America

Part Number: 502-00-003-0

Release: May 2003

Trademarks

Table of Contents

Introduction

Core1553BBC provides a 1553B Bus Controller, compliant with MIL-STD-1553B that you can implement in various Actel FPGA families (including the RTSX-S radiation-tolerant devices). Three versions of the core are available: an evaluation version that allows core simulation with the Actel Libero toolset or ModelSim, a netlist version that provides netlists and pre-compiled test benches and finally, an RTL version with full access to the source code. The directory structure is shown in Figure 1.

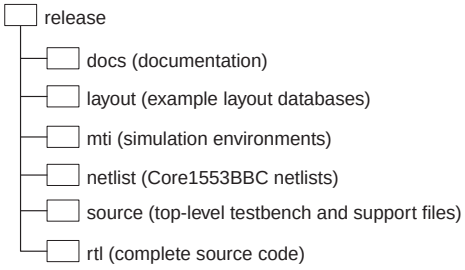

Figure 1. Core1553BBC Directory Structure

All the above directories are provided with the RTL version. The netlist version includes all directories apart from the RTL directories and the evaluation version only includes the *mti* and *docs* directories.

The netlist directory contains sixteen netlists, four for each supported family. This core release supports the SX-A, RTSX-S, Axcelerator, and Flash families. For each family netlists are provided with I/O and without I/O pads in both VHDL and Verilog formats. The SX-A netlists may also be used with the SX and RTSX families.

The “with I/O” netlists are used to create the layout databases provided in the *layout* directory; these databases may be used to program an FPGA device. If necessary, you can alter the pin locations within the Designer layout tool.

The “without I/O” netlists are intended for use when the core is instantiated in a user design. During the synthesis process the synthesis tool inserts the I/O pads.

The *source* directory contains the top level VHDL code for the testbenches and also some example support source files to ease design integration.

The *rtl* directory is provided to RTL licensees of the core. This directory contains VHDL and Verilog subdirectories containing the RTL source files.

Document Organization

The *Core1553BBC User Guide* contains the following chapters:

Chapter 1 - Using the Core1553BBC in the Libero IDE™

Chapter 2 - Standalone Simulation

Chapter 3 - Standalone Synthesis describes how to run two types of synthesis, netlist and RTL.

Chapter 4 - Standalone Layout

Chapter 5 - Verification Testbench describes the VHDL based verification testbench architecture.

Chapter 6 - User VHDL Testbench describes the customizable VHDL testbench architecture.

Chapter 7 - User Verilog Testbench describes your customizable Verilog testbench architecture.

Chapter 8 - Implementation Hints provides tips on how to integrate the Core1553BBC into your system.

Appendix A - Verfication Tests provides an overview of Actel's Core1553BBC testbench and a guide to procedures. It also describes the syntax for a variety of existing procedures and functions.

Appendix B - Product Support

1 Using the Core1553BBC in the Libero IDE™

This section describes how to use the Core1553BBC in the Libero Integrated Design Environment.

The following procedures explain how to load the User testbenches into the Libero environment. Once you load your user testbench into the Libero IDE you can modify it to suit your needs.

The files you load into Libero IDE depend on whether you have the evaluation, netlist or RTL versions of the core.

VHDL - Evaluation Cores

You must install the Core1553BBC in the Libero environment before you can modify the testbench.

To install the Core1553BBC in the Libero environment:

1. **Start Libero and create a new project, set the project name and select the family and HDL type.** Consult the Libero online help for more information on starting a project in Libero, selecting a family, etc.
2. **Import the QBCtbench.vhd file.** In the Libero File Manager import the QBCtbench.vhd file from the *$CORE1553BBC/source* directory to the Libero HDL directory.
3. **Copy the *Core1553BBC* directory to your Libero project.** In Windows explorer copy the *Core1553BBC* directory from *$CORE1553BBC/mti_Libero/user_vhdl* to the simulation directory in the Libero project.
4. **Copy the .do files to your Libero project.** In Windows explorer, copy the complibev.do and wavetb.do files from *$CORE1553BBC/mti_Libero/user_vhdl* to the simulation directory in the Libero project.

To run the simulation:

1. **In the Libero Tools -> Options -> Simulation window disable the "use Automatic Do file" option.**
2. **Run pre-synthesis simulation in Libero.** In the Libero Design Hierarchy pane select the QBCtbench.vhd and right-click and select “Run

Pre-Synthesis Simulation", and select "Run Modelsim without loading stimulus".

3. **Run "complibev.do" in ModelSim to compile the top-level testbench file and run the simulation.** Type "do complibev.do" in the ModelSim window.

   You can now edit the QBCtbench.vhd file and alter the testbench to perform different tests.

4. **(Optional) Run the Verification testbench.** Copy the command files "*.txt" and the runsim.do and wavetbv.do scripts from the *$CORE1553BBC/mti/verif_rtl* directory to the *simulation* directory in the Libero Project. Re-invoke modelsim as above and execute the runsim.do script, "do runsim.do". This testbench is fully described in "Verification Testbench" on page 27.

Synthesis and layout are not permitted with the evaluation core.

VHDL - Netlist Cores

Use the following steps to install, simulate, and run synthesis and layout on VHDL - Netlist cores. For more information on operations in Libero, please consult the Libero online help.

To install the Core1553BBC in the Libero Environment

1. **Start Libero and create a new project, set the project Name and select the family and HDL type.**
2. **Import the withoutio netlist file.** In the Libero File Manager pane import the required withoutio netlist file to the HDL directory, e.g. $CORE1553BBC/netlists/bc1553b_withoutio_sxa.vhd.
3. **Import the QBCtbench.vhd file into the Libero stimulus directory.** Import the QBCtbench.vhd file from the $CORE1553BBC/ source directory to the Libero stimulus directory.
4. **Copy the *Core1553BBC* directory to your Libero project.** In Windows explorer copy the *Core1553BBC* directory from *$CORE1553BBC/mti_Libero/user_vhdl* to the simulation directory in the Libero project.

5. **Copy the .do files to your Libero project.** In Windows explorer, copy the complibnl.do and wavetb.do files from *$CORE1553BBC/mti_Libero/user_vhdl* to the simulation directory in the Libero project.

To run the simulation

1. **In the Libero Tools -> Options -> Simulation window disable the "use Automatic Do file" option.**
2. **Set the netlist file as the design root.** Right-click the netlist file in the Libero Design Hierarchy pane and choose Set As Root.
3. **Run pre-synthesis simulation.** Right-click ModelSim Simulation and select “Run Pre-Synthesis Simulation”, then choose “Start Modelsim without loading stimulus”.
4. **Compile the top-level testbench file and run the simulation.** In the Modelsim window type “complibnl.do”.

   You can now edit the QBCtbench.vhd file and alter the testbench to perform different tests.
5. **(Optional) Run the Verification testbench.** Copy the command files "*.txt"and the runsim.do and wavetbv.do scripts from the *$CORE1553BBC/mti/verif_rtl* directory to the *simulation* directory in the Libero Project. Re-invoke modelsim as above and execute the runsim.do script, "do runsim.do". This testbench is fully described in “Verification Testbench” on page 27.

To run the synthesis and layout

1. **Set the netlist file as the design root.** Right-click the netlist file in the Libero Design Hierarchy pane and choose Set As Root.
2. **In the Libero Design Hierarchy pane right-click and select “Optimize and Insert pads”.**

   In the Synplicity window click RUN.
3. **Click layout when the Designer window appears.**

VHDL - RTL Cores

Use the procedure below to install, simulate, synthesize, and layout VHDL-RTL cores. For more information on operations in Libero, please consult the Libero online help.

To install the Core1553BBC in the Libero Environment:

1. **Start Libero and create a new project, set the project Name and select the family and HDL type.**
2. **Import the Core source files to the HDL directory.** In the File Manager pane, right-click and select Import. Browse to the files and click Open. The source files are all the files EXCEPT bccomps.vhd in the *$CORE1553BBC/rtl/vhdl/source* directory.
3. **Import the testbench files to the stimulus directory.** Use the File Manager pane in Libero to import the files.

   The testbench files are all the files in *$CORE1553BBC/rtl/vhdl/test*. Also import the bccomps.vhd file from the *$CORE1553BBC/rtl/vhdl/source* directory.
4. **Copy the .do files to your Libero project.** In Windows explorer, copy the complibrtl.do and wavetb.do files from *$CORE1553BBC/mti_Libero/user_vhdl* to the simulation directory in the Libero project.

To run the simulation:

1. **In the Libero Tools -> Options -> Simulation window disable the "use Automatic Do file" option.**
2. **Set the core as the design root.** Right-click the BC1553B.vhd file in the Libero Design Hierarchy pane and choose Set As Root.
3. **Run pre-synthesis simulation.** Right-click ModelSim Simulation and select “Run Pre-Synthesis Simulation”, then choose “Start Modelsim without loading stimulus”.
4. **Compile the top-level testbench file and run the simulation.** In the Modelsim window type “complibrtl.do”.

You can now edit the QBCtbench.vhd file and alter the testbench to perform different tests.

5. **(Optional) Run the Verification testbench.** Copy the command file "*.txt" and the runsim.do and wavetbv.do scripts from the *$CORE1553BBC/mti/verif_rtl* directory to the *simulation* directory in the Libero Project. Re-invoke modelsim as above and execute the runsim.do script, "do runsim.do". This testbench is fully described in "Verification Testbench" on page 27.

To run the synthesis and layout:

1. **Set the BC1553B.vhd file as the design root.** Right-click the BC1553B.vhd file in the Libero Design Hierarchy pane and choose Set As Root.
2. **Synthesize the design.** Right-click the BC1553B.vhd file in the Design Hierarchy and select "Synthesize."

   Click RUN in the Synplicity window. When the Designer window appears click Layout.

Verilog - Evaluation Cores

You must install the Core1553BBC in the Libero environment before you can modify the testbench.

To install the Core1553BBC in the Libero environment:

1. **Start Libero and create a new project, set the project name and select the family and HDL type.** Consult the Libero online help for more information on starting a project in Libero, selecting a family, etc.
2. **Import the QBCtbench.v file.** In the Libero File Manager import the QBCtbench.v file from the *$CORE1553BBC/source* directory to the Libero HDL directory.
3. **Copy the *Core1553BBC* directory to your Libero project.** In Windows explorer copy the *Core1553BBC* directory from *$CORE1553BBC/mti_Libero/user_vlog* to the simulation directory in the Libero project.

4. **Move the .do files to your Libero project.** In Windows explorer, copy the complibev.do and wavetb.do files from *$CORE1553BBC/mti_Libero/user_vlog* to the simulation directory in the Libero project.

To run the simulation:

1. **In the Libero Tools -> Options -> Simulation window disable the "use Automatic Do file" option.**
2. **Run pre-synthesis simulation in Libero.** In the Libero Design Hierarchy pane select the QBCtbench.v and right-click to select “Run Pre-Synthesis Simulation”, and select “Run Modelsim without loading stimulus”.
3. **Run “complibev.do” in ModelSim to compile the top-level testbench file and run the simulation.** Type “complibev.do” in the ModelSim window.

   You can now edit the QBCtbench.v file and alter the testbench to perform different tests.
4. **(Optional) Run the Verification testbench.** Copy the *Core1553B* directory, the command files "*.txt", runsim.do and the wavetbv.do scripts from the *$CORE1553BBC/verif_rtl* directory to the simulation directory in the Libero Project. Reinvoke ModelSim and execute the runsim.do script, "do runsim.do". This testbench is fully described in “Verification Testbench” on page 27.

Synthesis and layout are not permitted with the evaluation core.

Verilog - Netlist Cores

Use the following steps to install, simulate, and run synthesis and layout on Verilog - Netlist cores. For more information on operations in Libero, please consult the Libero online help.

To install the Core1553BBC in the Libero Environment

1. **Start Libero and create a new project, set the project Name and select the family and HDL type.**

2. **Import the withoutio netlist file.** In the Libero File Manager import the required withoutio netlist file to the HDL directory, e.g. $CORE1553BBC/netlists/bc1553b_withoutio_sxa.v.
3. **Import the QBCtbench.v file into the Libero stimulus directory.** Import the QBCtbench.v file from the *$CORE1553BBC/source* directory to the Libero stimulus directory.
4. **Copy the *Core1553BBC* directory to your Libero project.** In Windows explorer copy the *Core1553BBC* directory from *$CORE1553BBC/mti_Libero/user_vlog* to the simulation directory in the Libero project.
5. **Move the .do files to your Libero project.** In Windows explorer, copy the complibnl.do and wavetb.do files from *$CORE1553BBC/mti_Libero/user_vlog* to the simulation directory in the Libero project.

To run the simulation

1. **In the Libero Tools -> Options -> Simulation window disable the "use Automatic Do file" option.**
2. **Set the netlist file as the design root.** Right-click the netlist file in the Libero Design Hierarchy pane and choose Set As Root.
3. **Run pre-synthesis simulation.** Right-click ModelSim Simulation and select “Run Pre-Synthesis Simulation”, then choose “Start Modelsim without loading stimulus”.
4. **Compile the top-level testbench file and run the simulation.** In the Modelsim window type “complibnl.do”.

   You can now edit the QBCtbench.v file and alter the testbench to perform different tests.
5. **(Optional) Run the Verification testbench.** Copy the *Core1553B* directory, the command files "*.txt", runsim.do and the wavetbv.do scripts from the *$CORE1553BBC/verif_rtl* directory to the simulation directory in the Libero Project. Reinvoke ModelSim and execute the runsim.do script, "do runsim.do". This testbench is fully described in “Verification Testbench” on page 27.

To run the synthesis and layout

1. **Set the netlist file as the design root.** Right-click the netlist file in the Libero Design Hierarchy pane and choose Set As Root.

2. **In the Libero Design Hierarchy pane right-click and select “Optimize and Insert pads”.**

   In the Synplicity window click OK.

3. **Click layout when the Designer window appears.**

Verilog - RTL Cores

Use the procedure below to install, simulate, synthesize, and layout Verilog - RTL cores. For more information on operations in Libero, please consult the Libero online help.

To install the Core1553BBC in the Libero Environment:

1. **Start Libero and create a new project, set the project Name and select the family and HDL type.**
2. **Import the Core source files to the HDL directory.** In the File Manager pane, right-click and select Import. Browse to the files and click Open. The source files are all the files EXCEPT *bccomps.v in $CORE1553BBC/rtl/vlog/source.*
3. **Import the testbench files to the stimulus directory.** Use the File Manager pane in Libero to import the files.

   The testbench files are all the files in *$CORE1553BBC/rtl/vlog/test*.
4. **Copy the .do files to your Libero project.** In Windows Explorer, copy the complibrtl.do and wave.do files from *$CORE1553BBC/mti_Libero/user_vlog* to the simulation directory in the Libero project.

To run the simulation:

1. **In the Libero Tools -> Options -> Simulation window disable the "use Automatic Do file" option.**
2. **Set the BC1555B.v file as the design root.** Right-click the BC1555B.v file in the Libero Design Hierarchy pane and choose Set As Root.

3. **Run pre-synthesis simulation.** Right-click ModelSim Simulation and select “Run Pre-Synthesis Simulation”, then choose “Start Modelsim without loading stimulus”.

4. **Compile the top-level testbench file and run the simulation.** In the Modelsim window type “complibrtl.do”.

   You can now edit the QBCtbench.v file and alter the testbench to perform different tests.

5. **(Optional) Run the Verification testbench.** Copy the *Core1553B* directory, the command files "*.txt", runsim.do and the wavetbv.do scripts from the *$CORE1553BBC/verif_rtl* directory to the simulation directory in the Libero Project. Reinvoke ModelSim and execute the runsim.do script, "do runsim.do". This testbench is fully described in “Verification Testbench” on page 27.

To run the synthesis and layout:

1. **Set the BC1553B.v file as the design root.** Right-click the BC1553B.v netlist file in the Libero Design Hierarchy pane and choose Set As Root.

2. **Synthesize the design.** Right-click the BC1553B.v netlist file in the Design Hierarchy and select “Synthesize.”

   Click OK in the Synplicity window. When the Designer window appears click Layout.

2
Standalone Simulation

Core1553BBC has three separate simulation testbenches, a full verification environment for the 1553B Bus Controller core, and two user testbenches (one in Verilog and the other in VHDL). You can modify the user testbenches for Core1553BBC integration in your system. Details of these testbenches are provided in “Verification Testbench” on page 27, “User VHDL Testbench” on page 33, and “User Verilog Testbench” on page 35.

Verification Environment

Seven separate directories provided within the *mti* directory enable you to rerun core verification (Figure 2-1).

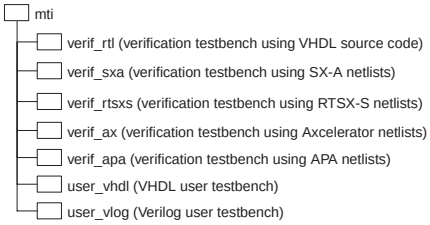

Figure 2-1. mti *Library Directories*

The Core1553BBC verification testbench is provided as pre-compiled ModelSim library files.

To run the verification testbench:

1. **Start ModelSim.**
2. **Change the directory to the *mti/verif_xxx* directory.** For example:

   ```
   mti/verif_rtl
   ```

3. **Refresh the simulation library with the “do refresh.do” command.**
4. **Run the simulation with the “do runsim.do” command.** The simulation starts and displays a menu of commands that you can enter.

   To run the complete verification suite enter the “doall” command, to run a short demonstration simulation enter the “demo” command.

If you use any of the netlist versions then you need to set up the Vital Libraries within the Actel Libero/Designer system (See the *Libero User's Guide*). The *verif_rtsxs* directory uses the SX-A Vital library. Netlist simulation runs are much longer than the RTL simulation run.

RTL licensees can use the compvhdl.do script instead of the "refresh.do" script to recompile the core and testbenches from the provided VHDL source code. The *verif_rtl* directory also has a "compvlog.do" script that simulates the Verilog source code rather than the VHDL source code within the VHDL test harness. If you wish to use the "compvlog" script, you must have a dual-language Modelsim license.

User Testbenches

Two user modifiable testbenches are provided, one in VHDL and the other in Verilog. These testbenches are intended to aid you as you integrate the core into the actual system.

The testbenches are provided as a pre-compiled ModelSim library (the top-level source files are provided in the source directory). The pre-compiled files are in the *mti/user_vhdl* or *mti/user_vlog* directory.

To run the VHDL user testbenches:

1. **Start ModelSim.**
2. **Change the directory to the *mti/user_vhdl* directory.**
3. **Refresh the simulation library with the "do refresh.do" command.**
4. **Run the simulation "do runsim.do".** The simulation starts and runs several 1553B messages.

RTL licensees can use the compvhdl.do script instead of the refresh.do script to recompile the testbenches from the provided VHDL source code.

To run the Verilog user testbenches:

1. **Start ModelSim.**
2. **Change the directory to the *mti/user_vlog* directory.**
3. **Refresh the simulation library with the "do refresh.do" command.**

4. **Run the simulation with the “do runsim.do” command.** The simulation starts and runs several 1553B messages.

RTL licensees can use the “compvlog.do” script instead of the refresh.do script to recompile the testbenches from the provided Verilog source code.

3

Standalone Synthesis

This chapter describes how to run synthesis for both netlists and RTL.

Synthesis for Netlists

The top-level entity/module name for the core is BC1553B. To simplify instantiation the component declarations are provided in the source directory (*bccomps.vhd* and *bccomps.v*).

Use either the Verilog or VHDL netlist without I/O cells. For example, the VHDL netlist for SX-A devices is bc1553b_withoutio_sxa.vhd, which is found in the *netlists* directory.

Tie the configuration inputs (ASYNCIF and CPUMEMEN) high or low as desired. During synthesis the synthesis tool optimizes the netlist by removing unnecessary logic. You can also drive these inputs with logic if required, but the core implementation is slightly larger.

Use a clock network to drive the CLK and RSTINn inputs to the core.

When using Synplicity with a Verilog flow you need to include the (Synplicity supplied) component macro library in the source file list. These libraries are located in the following directories:

```
synplify_install_dir/lib/actel
synplify_install_dir/lib/proasic
```

Use the macro library file that corresponds to your target architecture.

Synthesis for RTL

Instantiate the Core1553BBC top-level entity/module in your design. The top-level entity/module names for the core is BC1553B. To ease instantiation the component declarations are provided in the source directory (bccomps.vhd and bccomps.v)

The VHDL example below shows the included files for the Synplicity project.

```
add_file -vhdl -lib work "rtl/vhdl/core/bcbackend.vhd"
add_file -vhdl -lib work "rtl/vhdl/core/bcprotocol.vhd"
add_file -vhdl -lib work "rtl/vhdl/core/bcdecoder.vhd"
add_file -vhdl -lib work "rtl/vhdl/core/bcencoder.vhd"
```

```
add_file -vhdl -lib work "rtl/vhdl/core/BC1553B.vhd"
```

The Verilog synplicity project would be:

```
add_file -verilog -lib work "rtl/verilog/core/bcbackend.v"
add_file -verilog -lib work "rtl/verilog/core/bcprotocol.v"
add_file -verilog -lib work "rtl/verilog/core/bcdecoder.v"
add_file -verilog -lib work "rtl/verilog/core/bcencoder.v"
add_file -verilog -lib work "rtl/verilog/core/BC1553B.v"
```

Tie the configuration inputs (ASYNCIF and CPUMEMEN) high or low as desired. During Synthesis the synthesis tool optimizes the netlist, removing the unused logic. You can also drive these inputs with logic if required, but the core implementation is slightly larger.

Use a clock network to drive the CLK and RSTINn inputs to the core.

4 Standalone Layout

Once synthesized, run the core through the Designer layout tool to produce the programming files for the FPGA. The Core1553BBC meets timing in all supported families, so you can use standard layout flows. Actel recommends that you set the Core1553BBC clock input to match your desired clock speed (12, 16, 20 or 24 MHz) and enable timing driven layout.

The release contains a *layout* directory with a fully placed-and-routed core for each of the supported families. The *netlists* directory contains the matching netlists (for example, bc1553b_withio_sxa.v). These netlists have IO pads inserted around the core; do not use them when the core is integrated in another design.

5
Verification Testbench

Actel has developed a 1553B verification testbench that you can use to verify the core performance per the 1553B specification. The testbench is coded in VHDL and includes a Bus Controller and Remote Terminals connected to backend interfaces. A procedural testbench controls the various blocks and implements the tests. The source code is not made available with netlist licenses of the core (Figure 5-1).

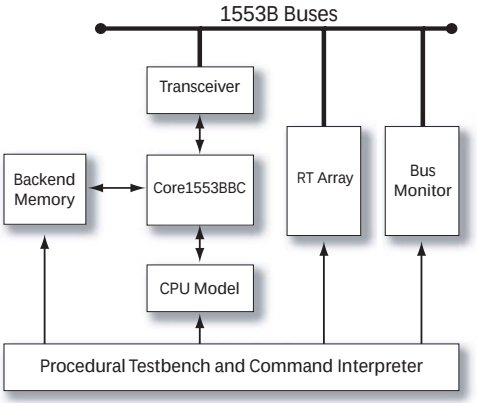

Figure 5-1. Verification Testbench

The testbench contains the blocks listed in Table 5-1.

Table 5-1. Verification Testbench RT Configuration

Abbreviation	Definition
CORE1553BBC	The bus controller
TRANSCEIVER	Models the 1553B transceiver
CPU	Models the CPU interface to the core
BACKEND	Provides the backend end memory connected to the bus controller. Operates in both Asynchronous and Synchronous modes with Programmable access times.

Table 5-1. Verification Testbench RT Configuration (Continued)

Abbreviation	Definition
RT ARRAY	Three RTs (RT 0, 1 & 2) based on the Core1553BRT. Each RT automatically maps receive sub-addresses to transmit sub-addresses and has programmable response times, as well as other error injection functions.
BUS MONITOR	Bus monitor that monitors 1553B activity detects error conditions.
INTERPRETER	Processes user input or command files and runs the simulations

The Core1553BBC verification testbench uses a command interpreter to apply high-level stimulus to the bus controller. This enables you to set the bus controller memory and registers. When started the simulation initializes and then wait for user input; a simple command file is shown below.

```
# Example Script

MONITOR 0 1 0 0    ! Turn on the Bus Monitor

# Set up message list
IL  #0000 $DOMSG  #0100
IL  #0002 $DOMSG  #0200
IL  #0004 $HALT   #0055

# BC to RT Message
MEM  #0100 #00F0 1.0.8.12  #0000 #0180 #0000 #0000 #0000
FILL #0180 #1111 32 1

# RT to BC message
MEM #0200 #00F1 1.1.8.12  #0000 #0280 #0000 #0000 #0000

REG $SETUP #5700! Enable clocks and set to 16MHz
GO #0000 #0055! Write to control registers and wait for interrupt

STATS
```

This command file sets up a message list that processes two messages, a BC to RT and RT to BC message. The command file writes to the bus controller SETUP register and the control register after programming the memory. The testbench then waits for the HALT instruction to generate the interrupt.

Supported Commands

The command interrupter supports the commands shown in Table 5-2; more detailed information can be found by using the Help command when the simulation is running.

Table 5-2. Supported Commands

DEMO	Runs demo.txt
DOALL	Runs doall.txt
AUTO	Runs doall.txt and quits
MONITOR CPU BUS BC RT	Turn Monitors on/off
PAUSE	Pause Simulation
GO [LISTPTR] [UVECT]	Start the BC, Wait for BC Interrupt
ABORT	Abort the BC
STOP	Stop the BC
ASYNC LISTPTR [UVECT]	Start the BC Async msg, Wait for BC INT
INT [UVECT]	Wait for BC INT and verify vector
REG [ADDR DATA]	Display or set BC registers
REGCMP ADDR DATA [MASK]	Compare BC register
REGBYTE UL ADDR DATA	Register Byte Write
DISPLAY ADDR [N]	Display BC memory
DISPLAY RT ADDR [N]	Display RT memory
MEM ADDR DATA [DATA]...	Set BC memory
MEMSET ADDR DATA [DATA]...	Set BC memory
MEMCMP ADDR DATA [MASK]	Verify BC memory
MEMBYTE UL ADDR DATA	Memory Byte Write

Table 5-2. Supported Commands (Continued)

FILL ADDR DATA [N][INC]	Fill BC memory
FILLMEM ADDR DATA [N][INC]	FIll BC memory
FILLCMP ADDR DATA [N][INC]	Verify BC memory
IL ADDR INST [FLAG] PARA	Set BC memory using OPCODES, etc.
BC CMD PARA	Set up the BC Backend
RT RTn CMD PARA	Set up the RT
MEMTEST ADDR SIZE LOOP [FAIL]	BC memory test
CALL filename	Run Commands from File
DO filename	Run Commands from File
INCLUDE filename	Run Commands from File
WAIT [CLOCKS]	Run Simulation for X Clocks, or 20 µs
STATS	Display Simulation Statistics
JUMP LABEL	Ignore Commands until Label matches
LABEL LABEL	Label for JUMP instruction
HELP	Display Help information
ECHO	Echo's text to simulation log
QUIT	Quit
.	Repeat the last command

Data for the commands can be entered in several forms, as shown in Table 5-3.

Table 5-3. Data Forms for Commands

1234	Decimal
#1234	Hexadecimal
A123	Automatically switches to hexadecimal
1.0.23.12	1553B Command Word RT=1 TX=0 SA=23 WC=12
#1F.#1.#1F.#01	1553B Command Word with hexadecimal values
$OPCODE	Enables you to enter OPCODES, as per the 1553BBC datasheet
$FLAG	Enables you to enter FLAGs, as per the 1553BBC datasheet
$REGISTER	Enables you to enter REGISTER, as per the 1553BBC datasheet

Command Files

Actel supplies a set of command files that you can use to verify the core. These command files provide 100% code coverage for the Core1553BBC RTL source code. The command files are listed in Table 5-4; a detailed list of the tests each of these command files performs is provided in Appendix A on page 41.

Table 5-4. Command Files

Command	Description
Async	Test the asynchronous message operation
Basic	Runs the ten basic message types
Demo	Simple simulation for demonstration use
Memsetup	Sets up a message list used for testing
Memcheck	Verifies all memory locations are as expected after messages in the MEMSETUP command file have been processed

Table 5-4. Command Files (Continued)

Command	Description
Memory	Tests the backend interface.
Opcodes	Tests the BC instructions
Flags	Tests the flag operations
Timers	Tests the BC timers and clocks
Retries	Tests the retry system
Rterrors	Tests with RT error conditions
Doall	Runs all of the above command files

Alternatively, you can create your own command files and invoke them using the INCLUDE command.

6 User VHDL Testbench

Actel provides an example testbench that you can use as the starting point for design verification of the core in your design. A block diagram of the testbench is shown in Figure 6-1.

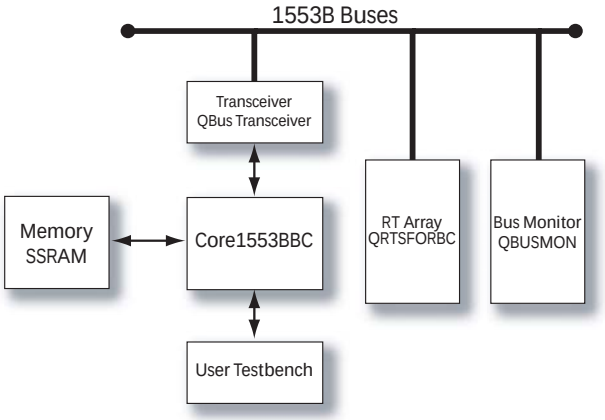

Figure 6-1. User VHDL Testbench

Table 6-1 lists the blocks and their descriptions.

Table 6-1. User VHDL Block Descriptions

Block	Description
QRTSFORBC	This block implements up to 30 1553B RT's connected to the 1553B buses. The number of RT's is set by the NRTS generic, when NRTS=4 RT's 0 to 3 are available. The RT's provide 32 transmit and receive sub-addresses, received data is copied to the transmit memory. When requested to transmit a vector word the RT will use the data word received from the "synchronize with DATA" mode code. The RT is based on the Core1553BRT IP core.
QBUSTRANSCEIVER	This block implements a single channel 1553B transceiver. It connects directly to the transceiver interface on the 1553B BC core and the 1553B bus.

Table 6-1. User VHDL Block Descriptions

Block	Description
QBUSMON	This block monitors the 1553B bus and displays the bus traffic. It connects directly the 1553B buses.
SSRAM	This block is synchronous SSRAM that can be connected directly to the core1553BBC backend interface. It implements 64K*16 of memory.

The main process in the testbench simply writes to the Core1553BBC CPU interface, and can program the Core1553BBC registers as well as the memory. To simplify the testbench the following procedure calls are provided:

```
procedure cpu_write_reg(address: integer range 0 to 7; data : integer) ;
procedure cpu_write_mem(address, data : integer) ;
procedure cpu_read_reg(address: integer range 0 to 7; data : out integer) ;
procedure cpu_read_mem(address : integer; data : out integer) ;
procedure cpu_write_mblk(address, data0, data1, data2, data3 : integer) ;
```

The first four procedures provide simple read and write functions to Core1553BBC registers or the memory. The fifth procedure allows the MSGBLK to be programmed with a single call, the four data values set the first 4 words in the MSGBLK (MSGTYPE, CW1, CW2 & DATAPTR), and the remaining four locations in the MSGBLK are cleared.

Actel recommends that you study the QBCTBENCH.vhd file provided in the source directory to fully understand how this testbench operates.

7

User Verilog Testbench

Actel provides an example testbench that you can use as the starting point for design verification of the core in your design. A block diagram of the testbench is shown in Figure 7-1.

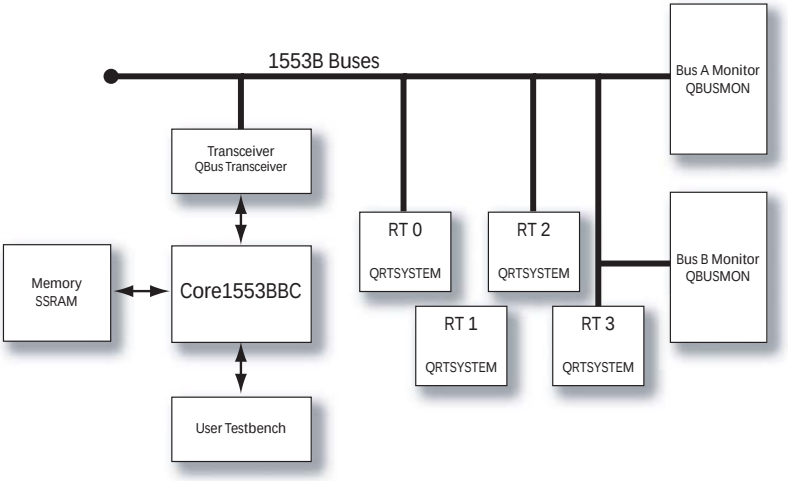

Figure 7-1. User Verilog Testbench

Table 7-1 lists the Blocks and Descriptions for the User Verilog Testbench.

Table 7-1. User Verilog Testbench Block Descriptions

Block	Description
QRTSYSTEM	This block implements a single 1553B RT connected to the 1553B buses. The RT provides 32 transmit and receive sub-addresses, received data is copied to the transmit memory. When requested to transmit a vector word the RT uses the data word received for the "synchronize with DATA" mode code. The RT is based on the Core1553BRT IP core.
QBUSTRANSCEIVER	This block implements a 1553B transceiver. It connects directly to the transceiver interface on the 1553B BC core and the 1553B bus.

Table 7-1. User Verilog Testbench Block Descriptions

Block	Description
QBUSMON	This block monitors the 1553B bus and displays the bus traffic. It connects directly the 1553B buses.
SSRAM	This block is a synchronous SSRAM that can be connected directly to the core1553BBC backend interface. It provides 64*16 of memory

The main process in the testbench simply writes to the Core1553BBC CPU interface, and can program the Core1553BBC registers as well as the memory. To simplify the testbench Actel provides the following tasks:

```
task cpu_write_reg;
    input [15:0] address;
    input [15:0] data;

task cpu_write_mem;
    input [15:0] address;
    input [15:0] data;

task cpu_read_reg;
    input [15:0] address;
    output [15:0] data;

task cpu_read_mem;
    input  [15:0] address;
    output [15:0] data;

task cpu_write_mblk;
    input [15:0] address;
    input [15:0] data0;
    input [15:0] data1;
    input [15:0] data2;
    input [15:0] data3;
```

The first four tasks above provide simple read and write functions to Core1553BBC registers or the memory. The fifth task allows the MSGBLK to be programmed with a single task, the four data values set the first four words in the MSGBLK (MSGTYPE, CW1, CW2 & DATAPTR), and the remaining four locations in the MSGBLK are cleared.

Actel recommends that the QBCTBENCH.v file provided in the source directory is studied to fully understand how this testbench operates.

8

Implementation Hints

Core1553BBC requires to be connected to a memory block to function. Core1553BBC allows the memory to be connected to the core in two ways, shared memory and its own memory.

Shared Memory

In this mode the core shares the CPU memory. Core1553BC will assert its MEMREQ output and when granted by the bus arbiter assumes control of the memory and completes its memory access cycle.

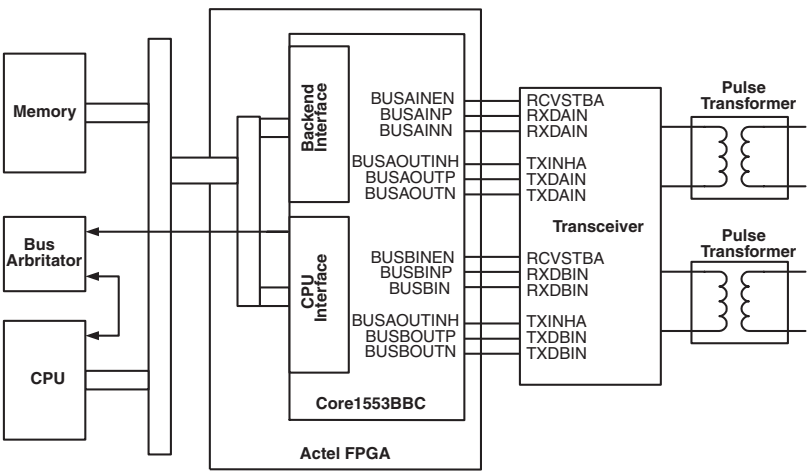

Figure 8-1. Core1553BBC with Shared Memory

Shared memory implementations can reduce overall cost as no special memory block needs to be implemented for the Bus controller, but the core requires direct access to the CPU memory bus.

In shared memory systems the CPUMEMEN input must be tied low.

Own Memory

In this mode the core has its own memory block. The CPU accesses the memory though the CPU interface of the core. The core provides the arbitration function allowing both the 1553B logic and the CPU interface to access the memory block.

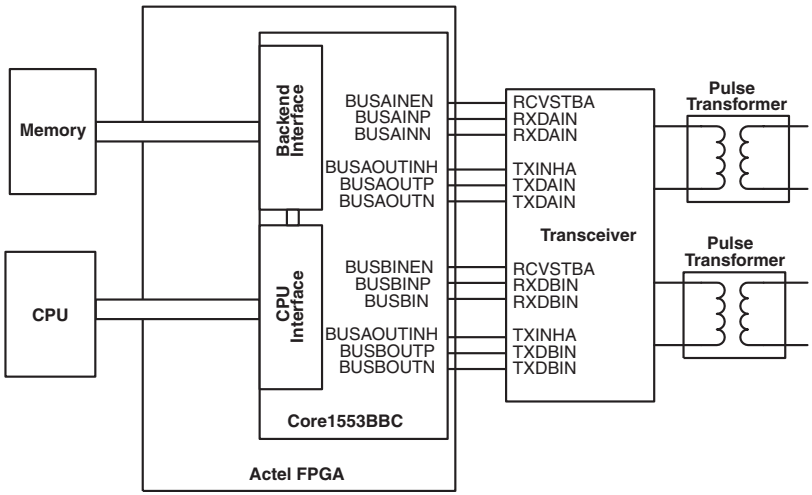

Figure 8-2. Core1553BBC with Own Memory

Actel recommends this implementation for FPGA devices that have on-chip RAM (APA & AX). The core backend interface can be directly connected to the FPGA synchronous memory block.

When the Core1553BBC has its own memory, CPUMEMEN input must be tied high.

Transceivers

Core1553BBC needs a 1553B transceiver to drive the 1553B bus. Core1553BBC is designed to directly interface to common MIL-STD-1553 transceivers, such as the DDC BU-63147, and the Aeroflex ACT4402. When using ProASIC^{PLUS} or Axcelerator, level translators are required to connect the

5V output levels of the 1553B transceivers to the 3.3V input levels of the FPGA.

In addition to the transceiver, a pulse transformer is required for interfacing to the 1553B bus. Figure 8-1 and Figure 8-2 show the connections required from the Core1553BBC to the transceivers and then to the bus via the pulse transformers.

A

Verfication Tests

The command files (provided) perform the tests shown in Table A-1.

Table A-1. VHDL Function Calls

File	Tests Performed
Async	Tests the asynchronous message operation Normal ASYNC message ASYNC message when a normal frame is active ASYNC message when already active ASYNC message when the stack is full RETAS instruction in normal message list RETAS instruction when the stack is empty ASYNC flag bit
Basic	Runs the ten basic message types and then checks the MSGBLK locations, including the 1553B status word and the BC transfer status word
Memory	Tests the backend interface. Tests access through the CPU interface to the backend memory Repeats the memory test whilst 1553B messages are being transmitted Checks the memory access failure timer at all operating frequencies Checks both Asynchronous and Synchronous memory modes Checks the MEMREQ & MEMGNT signals Checks the MEMWAIT input Checks byte operation
Opcodes	Tests the BC instructions Checks the clock and timer instructions Checks stack operation including overflow and underflow Checks illegal instruction detection
Flags	Tests the flag operations Checks all flag settings for both true and false conditions Checks illegal bits in the 1553B status word etc., sets the flags

Table A-1. VHDL Function Calls (Continued)

File	Tests Performed
Timers	Tests the BC timers and clocks Checks timers works with different input clock rates Checks the clock scalars work Check 1553B encoder and decoders at different input clock rates Check the inter message gap, set by the setup register Check the inter message gap, set by the message block Checks the RT response timer Checks clock value can be transmitted as mode code data
Retries	Tests the retry system Checks correct number of retries are carried out Checks both retry modes, same bus or alternate bus
Rterrors	Tests with RT error conditions Parity Error in SW Parity Error in DW Manchester Error in SW Manchester Error in DW Inverted SYNC on SW Inverted SYNC on DW Word Counts None, +1, -1 33 Mode Code, extra data Mode Code, no data No Response SW Incorrect RT Field RTRT no response TX RT RTRT no response RX RT RTRT SW's wrong Message Error SettingsMessage Error Settings RTRT Transmitter loopback test, data error Transmitter loopback test, sync error (The 1553B encoders and decoders are similar to those in the Core1553BRT core, the verification testbench supplied with the Core1553BRT also verifies that the decoder can cope with transmission rate errors and noisy received data)

The doall script invokes all the tests listed above, several tests use the memsetup and memcheck command files to initiate 1553B messages and verify the transfers.

B

Product Support

Actel backs its products with various support services including Customer Service, a Customer Technical Support Center, a web site, an FTP site, electronic mail, and worldwide sales offices. This appendix contains information about contacting Actel and using these support services.

Actel U.S. Toll-Free Line

Use the Actel toll-free line to contact Actel for sales information, technical support, requests for literature, Customer Service, investor information, and using the Action Facts service.

The Actel toll-free line is (888) 99-ACTEL.

Customer Service

Contact Customer Service for non-technical product support, such as product pricing, product upgrades, update information, order status, and authorization.

From Northeast and North Central U.S.A., call (408) 522-4480.
From Southeast and Southwest U.S.A., call (408) 522-4480.
From South Central U.S.A., call (408) 522-4434.
From Northwest U.S.A., call (408) 522-4434.
From Canada, call (408) 522-4480.
From Europe, call (408) 522-4252 or +44 (0) 1276 401500.
From Japan, call (408) 522-4743.
From the rest of the world, call (408) 522-4743.
Fax, from anywhere in the world (408) 522-8044.

Actel Customer Technical Support Center

Actel staffs its Customer Technical Support Center with highly skilled engineers who can help answer your hardware, software, and design questions. The Customer Technical Support Center spends a great deal of time creating application notes and answers to FAQs. So, before you contact us, please visit our online resources. It is very likely we have already answered your questions.

Guru Automated Technical Support

Guru is a web-based automated technical support system accessible through the Actel home page (http://www.actel.com/guru/). Guru provides answers to technical questions about Actel products. Many answers include diagrams, illustrations, and links to other resources on the Actel web site.

Web Site

Actel has a World Wide Web home page where you can browse a variety of technical and non-technical information. The URL is http://www.actel.com.

Visit the Actel IP website at http://www.actel.com/ip.

Contacting the Customer Technical Support Center

Highly skilled engineers staff the Technical Support Center from 7:00 A.M. to 6:00 P.M., Pacific Time, Monday through Friday. Several ways of contacting the Center follow:

Electronic Mail

You can communicate your technical questions to our e-mail address and receive answers back by e-mail, fax, or phone. Also, if you have design problems, you can e-mail your design files to receive assistance. We constantly monitor the e-mail account throughout the day. When sending your request to us, please be sure to include your full name, company name, and your contact information for efficient processing of your request.

The technical support e-mail address is **tech@actel.com**.

Telephone

Our Technical Support Center answers all calls. The center retrieves information, such as your name, company name, phone number and your question, and then issues a case number. The Center then forwards the information to a queue where the first available application engineer receives the data and returns your call. The phone hours are from 7:00 A.M. to 6:00 P.M., Pacific Time, Monday through Friday. The Technical Support numbers are:

(408) 522-4460
(800) 262-1060

Customers needing assistance outside the US time zones can either contact technical support via email (tech@actel.com) or contact a local sales office. Please see our list of Worldwide Sales Offices.

Worldwide Sales Offices

Headquarters

Actel Corporation
955 East Arques Avenue
Sunnyvale, California 94086
Toll Free: 888.99.ACTEL

Tel: 408.739.1010
Fax: 408.739.1540

US Sales Offices

California

Bay Area
Tel: 408.328.2200
Fax: 408.328.2358

Irvine
Tel: 949.727.0470
Fax: 949.727.0476

Newbury Park
Tel: 805.375.5769
Fax: 805.375.5749

Colorado

Tel: 303.420.4335
Fax: 303.420.4336

Florida

Tel: 407.977.6846
Fax: 407.977.6847

Georgia

Tel: 770.277.4980
Fax: 770.277.5896

Illinois

Tel: 847.259.1501
Fax: 847.259.1575

Massachusetts

Tel: 978.244.3800
Fax: 978.244.3820

Minnesota

Tel: 651.917.9116
Fax: 651.917.9114

New Jersey

Tel: 609.517.0304

North Carolina

Tel: 919.654.4529
Fax: 919.674.0055

Pennsylvania

Tel: 215.830.1458
Fax: 215.706.0680

Texas

Tel: 972.235.8944
Fax: 972.235.965

International Sales Offices

Canada

235 Stafford Rd. West,
Suite 106
Nepean, Ontario K2H 9C1

Tel: 613.726.7575
Fax: 613.726.8666

France

Actel Europe S.A.R.L.
361 Avenue General de Gaulle
92147 Clamart Cedex

Tel: +33 (0)1.40.83.11.00
Fax: +33 (0)1.40.94.11.04

Germany

Lohweg 27
85375 Neufahrn

Tel: +49 (0)8165.9584.0
Fax: +49 (0)8165.9584.1

Italy

Via de Garibaldini, No. 5
20019 Settimo Milanese,
Milano, Italy

Hong Kong

39th Floor
One Pacific Place
88 Queensway
Admiralty, Hong Kong

Tel: 852-22735712

Japan

EXOS Ebisu Building 4F
1-24-14 Ebisu Shibuya-ku
Tokyo 150

Tel: +81 (0)3.3445.7671
Fax: +81 (0)3.3445.7668

Korea

30th Floor, ASEM Tower,
159-1 Samsung-dong, Kangam-ku,
Seoul, Korea

Tel: +82.2.6001.3382
Fax: +82.2.6001.3030

United Kingdom

Dunlop House,
Riverside Way
Camberley,
Surrey GU15 3YL

Tel: +44 (0)1276.401452
Fax: +44 (0)1276.401490