BRM1553RT-FE

MIL-STD-1553 REMOTE TERMINAL FRONT END

&

BRM1553RT-ERL

EXTENDED RELIABILITY LOGIC

MIL-STD-1553 REMOTE TERMINAL FRONT END FPGA CORE

USER'S MANUAL

Based on BRM1553D-RT IP core
Version 8.4
DO254-DAL-A Verified

TABLE OF CONTENTS

TABLE OF FIGURES

1. INTRODUCTION

1.1 ABOUT THIS MANUAL

This document is the user's manual for BRM1553FE-RT and the BRM1553ERL-RT Intellectual Property (IP) cores. These IP cores implement the exact same interface to the back-end logic, but are different in the implementation of the internal state machines and logic.

Unless otherwise specified in this document, all reference made to the BRM1553FE-RT, is equal to the BRM1553ERL-RT. Whenever a specific reference for the BRM1553ERL-RT is required, then it will be mentioned in the document.

Note:
Please note that it is assumed that the user of this manual is knowledgeable about the Mil-Std-1553 protocol, FPGA design and the VHDL hardware design language.

1.2 ABOUT BRM1553FE-RT AND BRM1553ERL-RT

The BRM1553FE-RT IP core is intended for aircraft and ground avionics equipment, whereas the BRM1553ERL-RT IP core is intended to be used in space applications, as its implementation takes into account problems that are specific to outer space.

Note:
The main difference between BRM1553ERL version and BRM1553FE is that the 'ERL' state machines are controlled by a type of watchdog that constantly monitors the 1553 bus. This watchdog is not a simple time-out counter, but rather a 1553 driven time-out detector. By this, the state machines, even if for any reason get stuck, are reset at the right point of time to get ready for a new message on the bus. There is no functional difference between the two versions.

The BRM1553FE-RT IP core provides a simple to use link for MIL-STD-1553 Remote Terminals (RT) that need to interface the MIL-STD-1553B Notice 2 bus.

This IP core is typically used in RTs that are directly controlled by the messages sent by the MIL-STD-1553B bus controller (BC). In most cases such RTs will not need to incorporate neither a Central Processing Unit (CPU) nor its software to correlate between the terminal and the MIL-STD-1553 bus.

The BRM1553FE-RT on its own was validated to meet the MIL-STD-1553B Notice 2 Remote Terminal Validation test plan, thus reliving the user from mastering the standard.

The following diagram shows the inputs and outputs of the IP core:

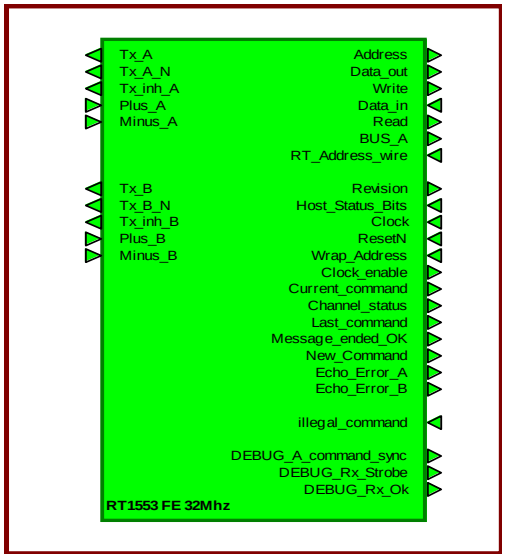

Figure 1:BRM1553FE-RT Inputs/Outputs

1.3 BRM1553FE-RT KEY FEATURES

- Very low FPGA area usage (less than 900 4-LUT space).
- Synthesis-able into any FPGA family(*).
- Uses any even whole number clock frequency from 12 Mhz.
- CPU host interface typically not needed.
- Connects to any transceiver-transformer pair.
- Passed full validation testing.
- Simple to use “Message Bus” interface to terminal.
- State Machine implementation, no micro code.
- Extended Reliability Logic option adds logic to make the core more reliable for harsh environments typically seen in space.

1.4 TERMS USED IN THIS DOCUMENT

- **Terminal** – The whole box that connects to the MIL-STD-1553 bus.
- **FPGA** – Field Programmable Gate Array, a programmable device that contains the IP core and user logic.
- **User Logic** – Logic Circuit that resides in the FPGA that is not the IP core.
- **Core** – The supplied logic circuit that interfaces to MIL-STD-1553 bus.
- **Message Bus** – a set of signals and well-defined timing that interfaces the core with the user logic inside the FPGA, within the Remote Terminal.
- **Command word bit order** – The command word contains 16 bits, bits 15..11 are the terminal address, bit 10 is the transmit or receive bit, bit 9..5 are the sub address, and bits 4..0 is the word count.

1.5 IP CORE DELIVERABLES

The IP core is supplied with the following components:

- EDIF net list for desired FPGA family and clock frequency.
- This user's manual.
- Sample VHDL code that incorporates the core.
- Synthesis script for sample code.

2. BASIC FUNCTIONALITY

2.1 BLOCK DIAGRAM

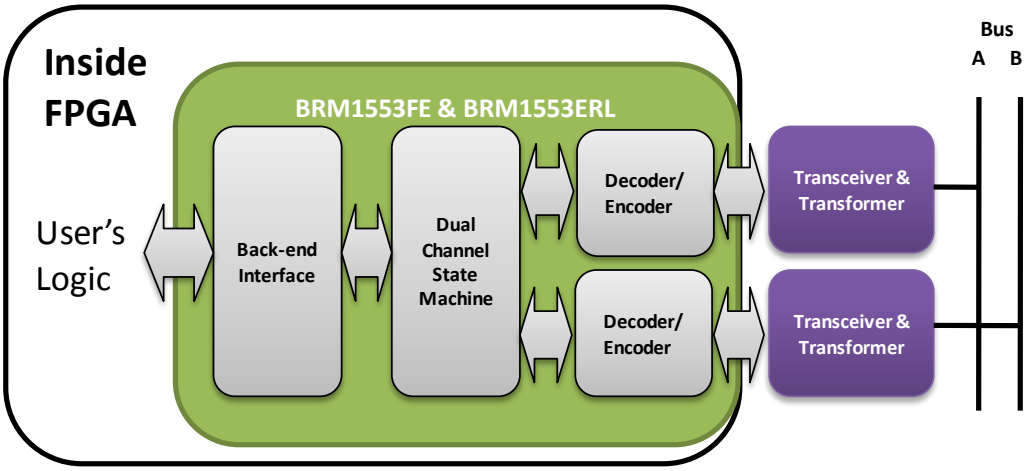

Figure 2:BRM1553FE-RT & BRM1553ERL-RT Block Diagram

The BRM1553FE-RT core incorporates two decoders that translate the serial bus messages from the MIL-STD-1553 Manchester coding and format, into a 16 bit parallel data accompanied by status indications for each word.

A set of these bus words are identified by the dual channel state machine as a valid message, being transmit, received, mode or broadcast message.

When the State Machine determines it has to either store a word to the terminal or fetch a word from the terminal it creates a write or read cycle to the user's logic to store or fetch that word.

When a word is fetched and needs to be transmitted, the state machine gets it from the backend and loads it into the encoder, which in turn encodes the 16 bits into the Manchester coding and off to the transceivers to the 1553 bus.

The backend also incorporates a 32 by 16 bit memory block that is needed for data loop-back test which is part of the MIL-STD-1553B Notice 2 requirements.

2.2 "MESSAGE BUS" INTERFACE

The BRM1553FE-RT interfaces to the user logic through the Message Bus. This bus emulates a standard CPU bus type, which includes a 16-bit address bus, 16-bit data out bus, 16-bit data in bus, a write pulse and a read pulse.

When a relevant word is received from the Bus Controller (BC) on the 1553 bus, a WRITE cycle is sequenced on the message bus. When a word needs to be transmitted to the BC, a READ cycle is sequenced.

The READ and WRITE cycles are generated only for valid words in valid messages. If a 1553 message fails, the IP core will stop its writing or reading over the Message bus till its end. A new 1553 message will re-start the READ and WRITE cycles.

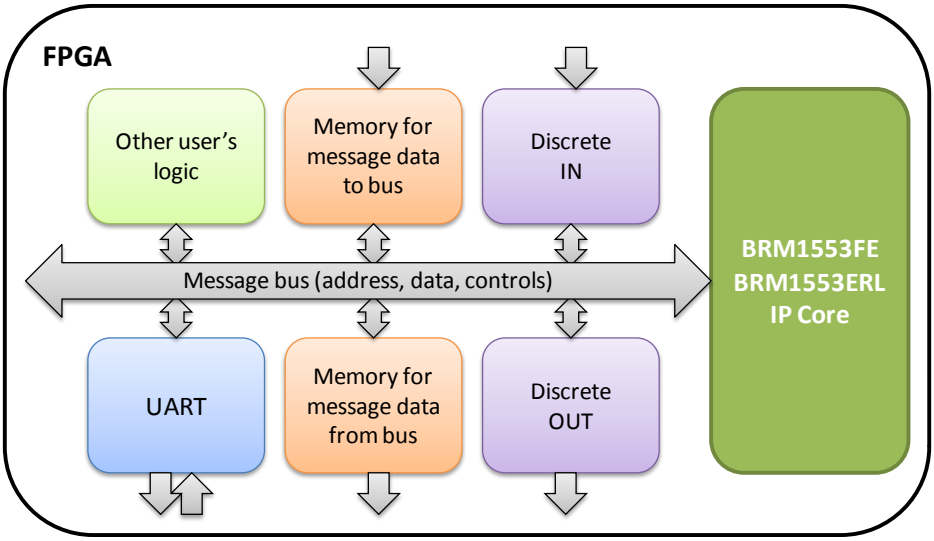

Figure 3:Message Bus interface examples

In addition to READ and WRITE cycles, there are additional control signals, which allow the terminal to better understand what is going on the 1553 bus, and respond accordingly. These signals include 'Channel_Status', 'Last_Command', 'Message_Ended_OK', 'New_Command', 'Echo_Error_A' and 'Echo_Error_B'.
These signals are discussed later on in this manual.

2.3 RESPONSE TO MODE CODES

The RT has some additional hardware allocated for response to certain mode code commands.

The following list summarizes responses per mode code. All commands are echoed to backend for user.

Mode	T/R	Code	BCST	Action Taken by the BRM1553FE-RT
Dynamic bus control	1	00000	Yes	No special action. Command sent to backend. Clear Status Response. DBC bit in status word is kept 0.
Synchronize	1	00001	Yes	No special action. Command sent to backend. Clear status response.
Transmit Status	1	00010	No	Send status response. Do not clear status bits.
Initiate Self-Test	1	00011	Yes	No special action. Command sent to backend. Clear status response.
Transmitter Shutdown	1	00100	Yes	Mute transmission from other bus. Clear status response.
Override Transmitter Shutdown	1	00101	Yes	Unmute transmission on other bus. Clear status response.
Inhibit Terminal Flag Bit	1	00110	Yes	Inhibit bit 0 of status response. Send 0 even if it is requested to be 1.
Override Inhibit Terminal Flag Bit	1	00111	Yes	Override Inhibit bit 0 of status response. Send the requested Host status bit 0 in status response.
Reset RT	1	01000	Yes	Sends clean status. 4 us later it resets all FFs in RT. “Current command” is not reset.
Transmit Vector Word	1	10000	No	Sends clean status, followed by transmit data that is present on “Data_in” lines during the backend read cycle.
Synchronize with data	0	10001	Yes	Sends clean status, standard write data cycle on message bus with data received.
Transmit Last Command	1	10010	No	Sends clean status, send the latest valid command (excluding this command).
Transmit BIT Word	1	10011	No	Sends clean status, followed by transmit data that is present on “Data_in” lines during the backend read cycle.
Selected Transmitter Shutdown	1	10100	Yes	Mute transmission from other bus. Clear status response. Data word is written to backend (no use for that word).
Override Selected Transmitter Shutdown	1	10101	Yes	Unmute transmission on other bus. Clear status response. Data word is written to backend (no use for that word).

3. HARDWARE INTERFACE

The following sections explain the Message Bus. Proper understanding of the message bus is the key for a successful usage of the core.

3.1 WRITE CYCLE

A 'receive' message on the 1553 bus instructs the terminal to receive and potently use the data contents of the message for its operation.

A receive command with several words will result with the same number of write cycles on the message bus. One Write cycle for each word that is received from the bus.

The following figure shows a single Write cycle:

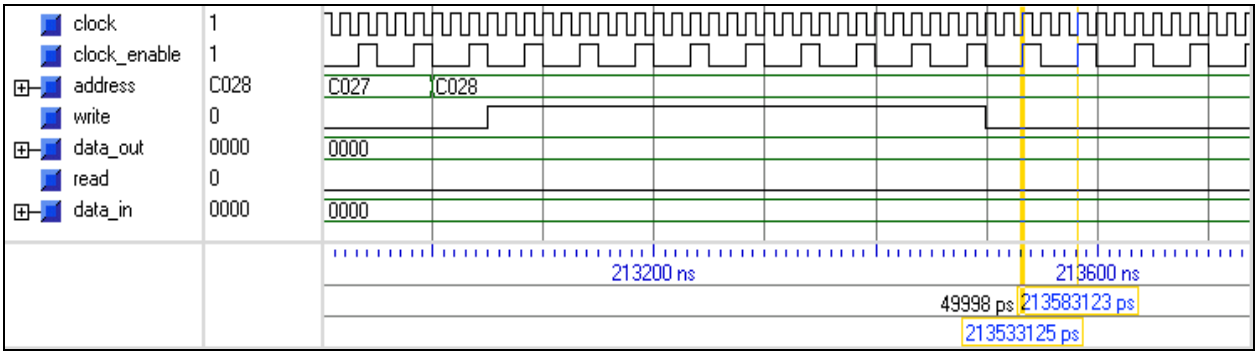

Figure 4:Write Cycle

A write cycle is fully synchronous to the clock cycle. The user code should work with the same clock that is supplied to the IP core, and **sample data on the rising edge of that clock**. The address and data will not change during cycle.

The write cycle would last for about 1 microsecond, depending on the clock frequency. If the clock cycle is 16 Mhz it will last for 16 clock cycles, if the clock is 32Mhz it will last for 32 clock cycles.

This long period is provided for applications that require indirect memory access based on the address value.

The address bus is of 16 bits. The address is derived from the 1553 command.

Bits 15..11 are the received Terminal address of the command. They would be identical to the terminal address bits input to the core unless the message is a broadcast command, then the 5 bits will all be logic '1'.

Bit 10 would be '0' since this is a receive message.

Bit 9..5 are the sub address of the command.

Bits 4..0 indicate which word is being stored. The first word to be stored would be “00000”, the second “00001”… and if there are 32 words, the last one would be “11111”.

Note: if the command was for 3 words then the word count field would be 00011. The Address’ 5 LSBs would then be 00000 for the first word, 00001 for the second, and 00010 for the third.

RECEIVE MODE COMMANDS

If the message is a receive mode command with data, bits 9..5 would be all zero or all ones as was sent on the bus.

Bits 4..0 would be the actual 5 LSBs of the mode command message indicating which mode command were received.

If the message is a receive mode command without data, no write cycles will be generated.

A ‘New_Command’ signal would be generated (as explained in later paragraphs) which allows the user code to track these commands if required.

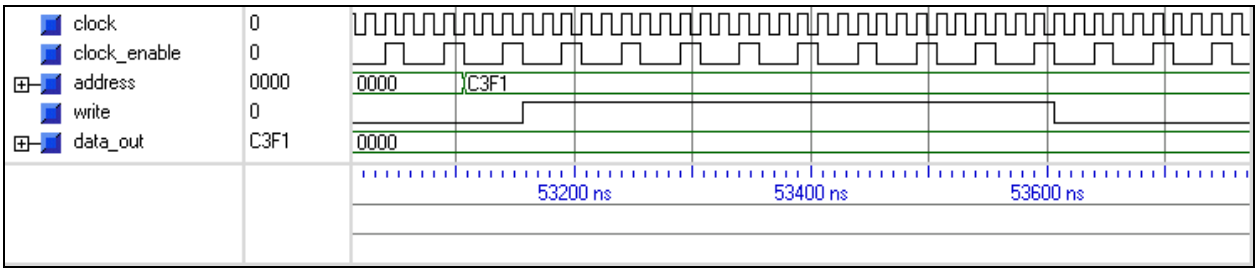

Figure 5:Receive Mode Command Cycle

3.2 READ CYCLE

A transmit message on the 1553 bus instructs the terminal to transmit terminal data.
A transmit message with 13 words will result with 13 read cycles on the message bus.

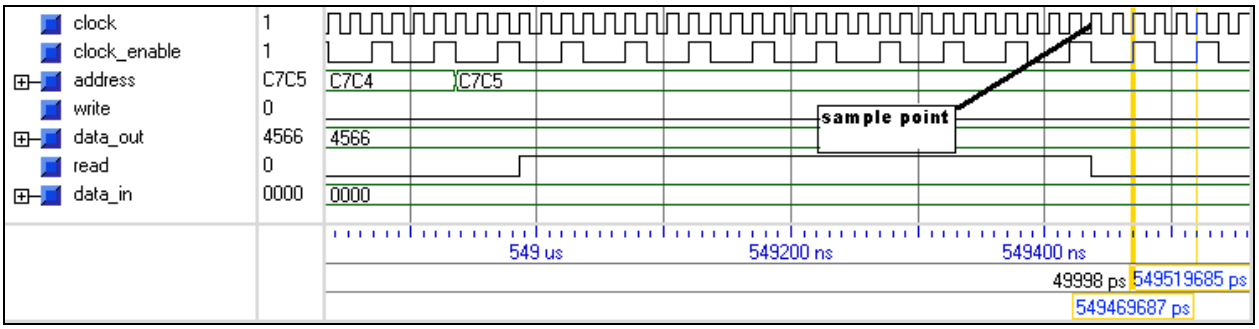

Figure 6:Read Cycle

A read cycle is fully synchronous to the clock cycle. The user code should work with the same clock that is supplied to the RT1553FE, and supply the data from a register or memory that is clocked by the **rising edge of that clock.**

The data is sampled internally in the core on the last rising edge of the clock when read is still asserted.

The read cycle would last for around 1 microsecond.

This long period is provided for applications that require indirect memory access based on the address value.

The first read cycle will occur about 6 us after the 'new_command' signal (see below) pulse that announced the arrival of a valid transmit command. During this time, the user logic may fetch relevant data for transmission.

The address is 16 bits. The address is derived from the 1553 command.

Bits 15..11 are the received Terminal address of the command. They would be identical to the terminal address bits input to the core unless the message is a broadcast command, then the 5 bits will all be logic '1'.

Bit 10 is always '1' since this is a transmit message.

Bits 9..5 are the sub address of the command.

Bits 4..0 indicate which word is being read. The first word to be read with "00000", the second with "00001"… and if there are 32 words, the last one would be "11111".

TRANSMIT MODE COMMANDS

If the message is a transmit mode command with data, bits 9..5 would be all zero or all ones as was sent on the bus.

Bits 4..0 would be the actual 5 LSBs of the mode command message indicating which mode is it.

If the message is a receive mode command without data, no read cycles will be generated. A 'new_command' signal would be generated (as explained in later paragraphs) which allows the user code to track these commands if required.

See Figure 7: below for details.

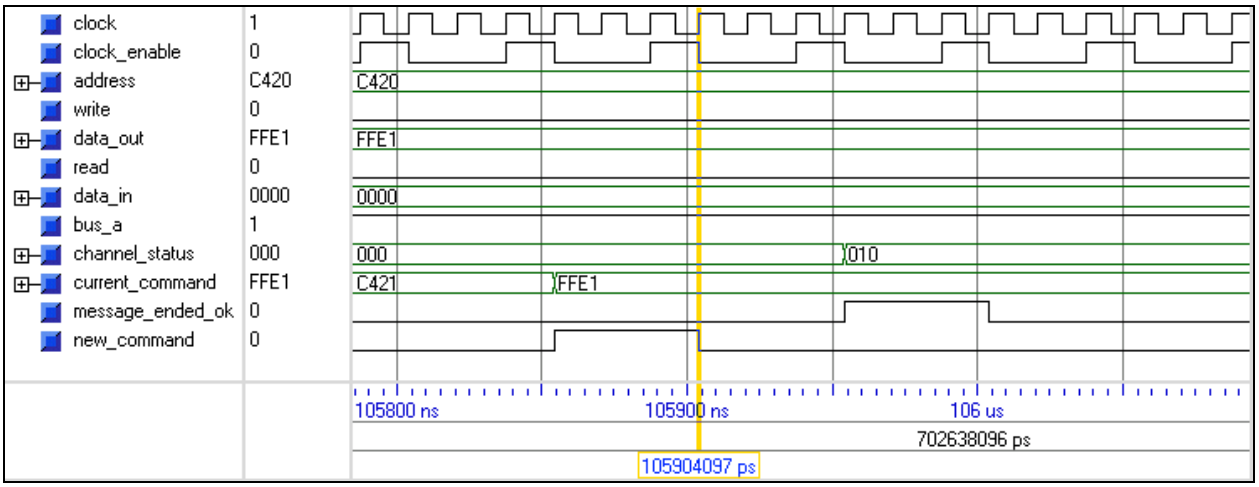

Figure 7:Transmit Mode Command

3.3 MESSAGE BUS SIGNALS

See Figure 8: below for details.

Address 16 bit address for write or read. The address 11 MSBs are identical to the current command being processed. The 5 LSBs indicate which word is being written or read. For the first word of the sub-address these 5 bits will be 00000, for the second 00001, and so on until 11111 for the 32nd word is such exist in the message.

Note: in mode commands (bits 5 to 9 are 00000 or 11111) these 5 LSBs are equal to the current command. Some mode commands have a single data associated with them.

Write Write pulse. When high, user logic should take the data from Data_out lines for its usage. This pulse is 1 microsecond long. Use should use the write pulse as enable, and latch data with the rising edge of the clock.

Data_out 16 bit data used in write cycles. These bits hold the data that was received from the 1553 bus and need to be stored to the subsystem.

Read Read pulse. When high, user logic should provide the data to Data_in lines for the core to transmit to the bus controller over the 1553 bus. This pulse is 1 micro second long.

Data_in 16 bit data used in read cycles. These bits hold the data that is going to be transmitted to the 1553 bus.

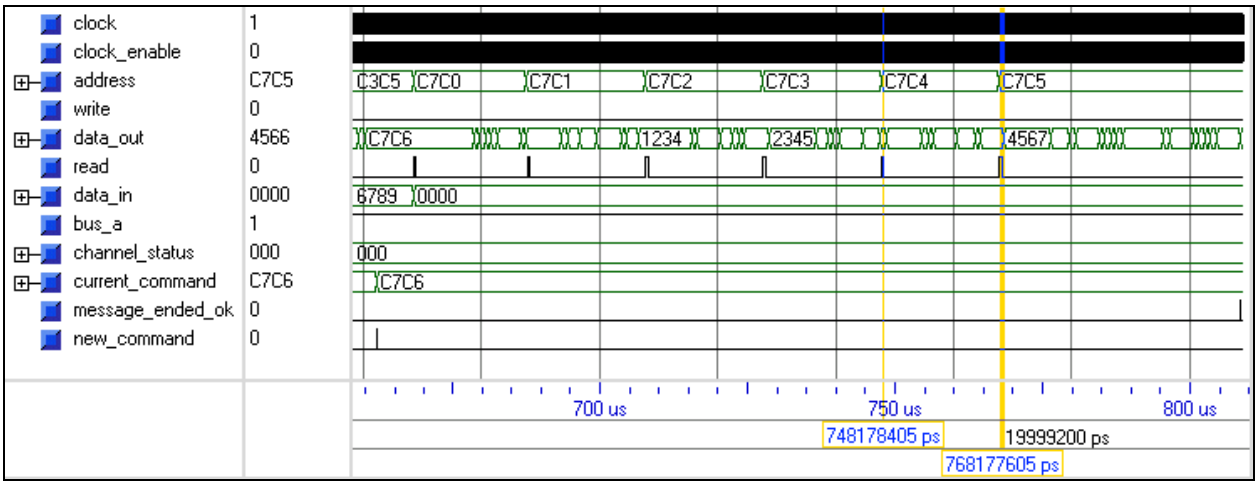

Figure 8:Message Bus Signals

3.4 CONTROL SIGNALS

INPUT SIGNALS

The BRM1553FE-RT requires several controls to properly configure the core for work. The controls are:

RT_Address_wire — 6 bits that define the address of the Remote Terminal on the bus. The 5 LSBs are the address. These bits should not change during the operation of the core. If, for example, the terminal address is 20 decimal, then that value of these bits should be “10100” (4..0).

The MSBit (5) is the odd parity bit of the address. If parity is wrong, the core has no address and will not respond to any command other than broadcast commands which are considered valid even with wrong parity.

Clock — The core is synthesized to work with a specific clock frequency. This pin should be connected to that appropriate clock. The clock should be connected to one of the global clock buffers of the FPGA.

CPU_ResetN — An active low signal that asynchronously resets all of the FFs in the core. This signal is internally sampled to the clock. This signal does not reset the 32 word memory inside the core that holds the wrap around data. This signal can be used to inhibit the core from usage.

Wrap_address — (4..0) 5 bits that define the 1553 sub address for which the data received by the core should transmitted back as defined by the standard. Normally this sub address should be 30 (11110).

Host_status_bits (10..0) An 11 bits vector defined by the Terminal (FPGA logic) that change the value of the status word replied by the core for each valid non broadcast command. Bits 0 (terminal flag), 1(Dynamic Bus Control), 2 (subsystem flag), 3 (busy), 4(Broadcast), 5..7 (Reserved), 8 (service request), 9 (instrumentation), 10 (Message Error) effect the status return. If bit 3 is set high, busy takes effect only when the bus is idle or at the end of a valid command, with the rise of the valid command pulse. The core will become busy, which will result in no data response for transmit commands, and no back-end read cycles, the core will transmit only the status word with busy bit set, as defined by the 1553 standard. Receive messages will be serviced by the core, thus it is up to the terminal logic which asserted the busy bit, to decide if to use the data or not. The other non-discarded bits' effect would be to set the appropriate bit in the status respond. Bit 4 and Bit 10 are provided to user effect, but are also managed by the IP. It is strongly recommended to set them to '0' at all times.

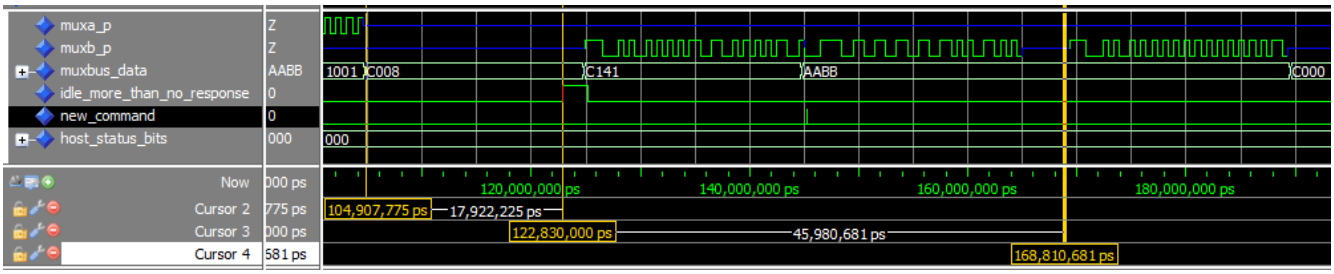

Figure 9:Host Status Bits sampling point

The Host Status bits are sampled into the RT just before transmitting the status word (at point 168,810,681 ps above) with the exception of the following signals:

Bit 0 – Terminal flag may be masked off by a mode command from the Bus Controller.

Bit 3 – Busy. This bit is sampled on rising edge of "new command" and effects this new command. It is also sampled when the bus is idle as seen above. In the above timing it is 18 us after completion of last word of message.

Bit 4 – Broadcast bit. Can also be set by host, but discouraged. IP sets it in accordance with 1553 standard.

Bit 10 – Message Error. Can also be set by host, but discouraged. IP sets it in accordance with 1553 standard. Can be used in MIL-STD-1553A.

Illegal_command This signal informs the core that the message is illegal as defined by the MIL-STD-1553B standard. If illegalization is used, this signal is sampled after one microsecond from new_command. It is strongly suggested to place decode logic on the "Current_command" bus to decode those messages that are illegal and set the illegal_command signal within a clock or two. If set high, the core will respond to transmit commands with a status word with Message Error set and no data words. The core will perform no READ operations. For receive commands, no WRITE cycles would be cycled by the core. Note that new_command and message_ended_ok signals will be functioning normally as in legal command. If illegalization is not used, please tie this signal to '0'. *For illegal single word messages, such as broadcast no data, the illegal_command signal is sampled AFTER arrival of message_ended_ok.*

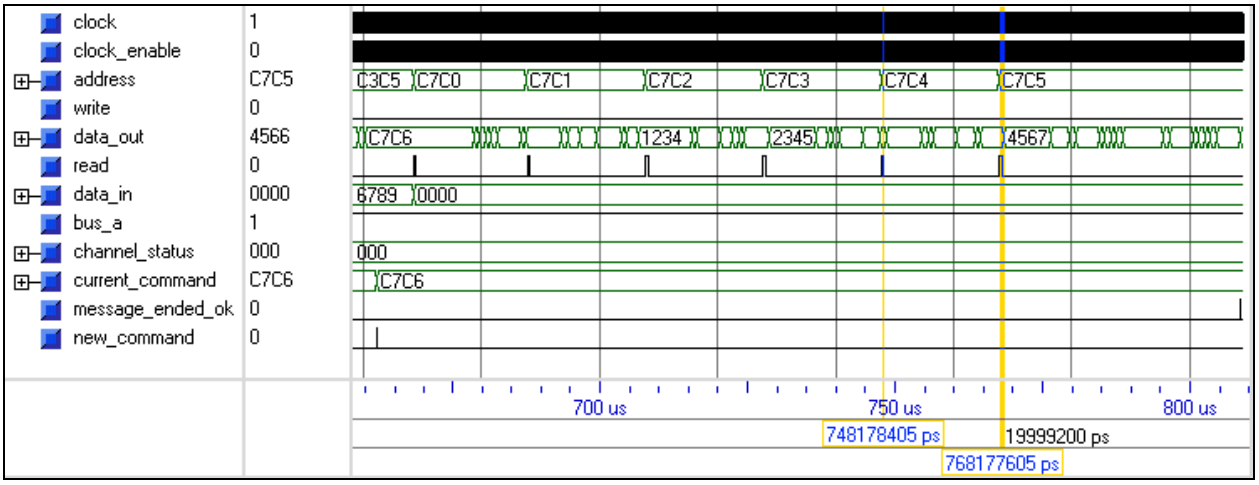

Figure 10: Input Control Signals

OUTPUT SIGNALS

The RT1553FE provides the following control signals. All control signals are synchronous to the rising edge of the clock. In order to use these control signals the user code should sample them on the rising_edge of the clock that drives the core.

Clock_Enable The core internally generates a clock_enable signal. All of the FFs inside the core and the interface controls change their value with the rising edge of the clock when the clock_enable is active high. The frequency of clock_enable depends on the clock frequency for which the core is synthesized, and is internally generated. Normally this frequency would be between 14 and 30 Mhz.

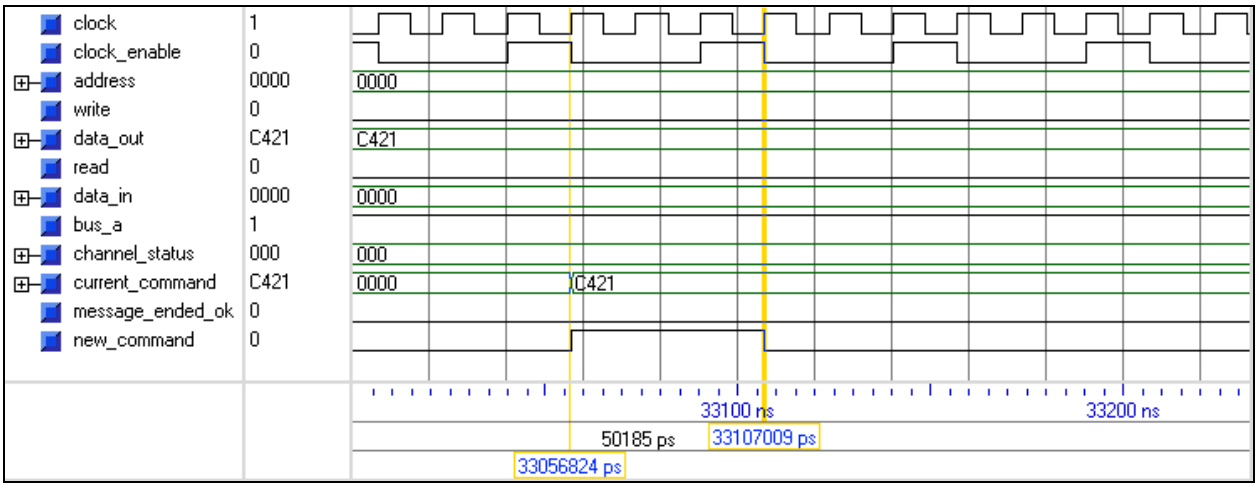

Figure 11: Clock_Enable; New_Command Signals

New_Command Active high signal that indicates that a new command has arrived and it is a valid command intended for this Remote Terminal. The value of the newly received command is available on the current_command bus.

Note: the MIL-STD-1553 standard supports the option to stop the process of a valid message before its complete as a result of bus switching or command superseding. In such event, the core responds as defined by the standard, and will switch to the new valid command and discontinue the interrupted message.

Current_command defines the active command that is being processed by the core. The value of this bus is updated with the new_command signal and will stay stable until a new valid command is received.

Message_ended_OK An active high signal indicating that the current message has ended OK without any errors. The terminal may choose not use the data written by the core until the message ends OK. Only valid words before a non-valid word are written to the back-end, so some terminals may choose to use these words regardless of this signal. Message_ended_OK is set after the last bit of the message is transferred on the MIL-STD-1553 bus.

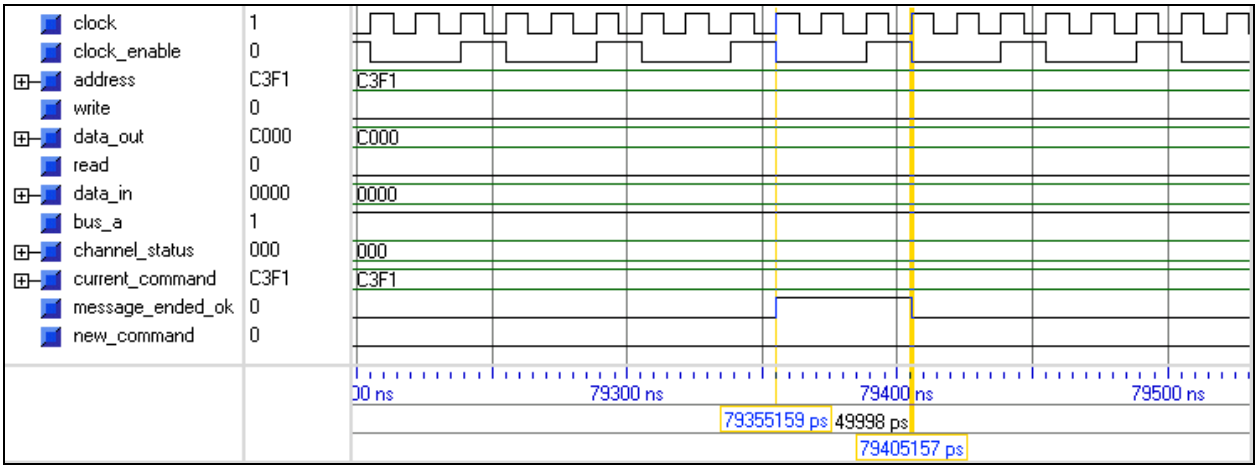

Figure 12: Message_Ended_OK Signal

Channel_status (10..0) The actual 11 LSBs of the status word replied by the core to the Bus Controller. Bit 10, the MSB, is the Message Error bit indicating that an error was found in the last processed message. It is set exactly when the error was found, and reset when a new valid command is received in accordance with the standard.

Bus_A Logic high when the current command is on bus A, and low for B.

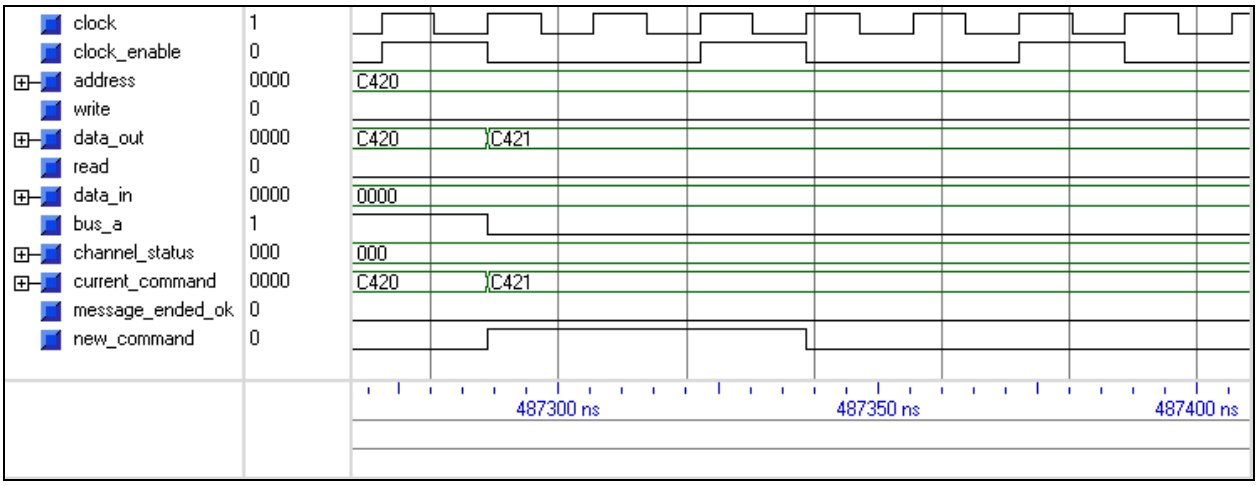

Figure 13: Bus A Signal

Revision 16 bits defining the revision of the core.

Echo_Error_A Loop back error indication. Each word sent to the bus is echoed back into the receiver. The echo is checked to be identical to the data sent. If the echo data is not same as sent, or does not exist, this line will be set to logic high for a single clock cycle with CE high.

Echo_Error_B same as Echo_Error_A but for bus B.

Last_command The last command as defined in “Transmit Last command mode word”. Typically not to be used.

3.5 FRONT-END SIGNALS

The front-end signals are 10 lines connected to the 1553 transceiver on to its digital side. These signals should be placed on normal pads of the FPGA used.

These signals support all popular 1553 transceivers. Some transceivers’ idle state is indicated by both receive signals high, and some low. The core supports both types automatically.

Tx_A In phase Transmit bit. Zero when no transmission.

Tx_A_N Inverted Transmit bit. It is inverted from Tx_A only during transmission, and is zero otherwise.

Tx_inh_A Transmission inhibit signal connected to transceiver. This signal is normally high, and is asserted low by the core during transmission or if endless transmission error occurs.

Plus_A	In phase received signal. After the bus is quite, this signal should be the first to rise to high, since command or status sync starts with 1.5 us high. This signal should be either constant high or low when bus is not active.
Minus_A	Inverted receive signal during reception. When bus is not active, this signal should be at the same level of Plus_A, either high or low.

Same signals exist for Channel B.

4. ERROR CONDITIONS

1. In case the command word has an error, no activity would occur on the Message Bus. No Status would be sent, and typically the Bus Controller would re transmit that command.

2. In case of a valid 'receive command' word is received, the core would write all of the valid data words associated with that message on the message bus. However, a data word that contained a fault and all the following words in that message would not be written on the message bus.

3. Receive messages that contained a fault would prevent a status reply to the bus controller. Message_ended_OK will not be set in that case.

4. A valid 'transmit command' will always request the data word from the message bus, and if error occur, then it is up to the Bus Controller to request that message again.

5. Some designs would wait for Message_ended_OK to be set in order to accept the data words that were received by means of buffering. Other designs can use the words that were received properly and not wait for Message_ended_OK. As of the first invalid word, the back-end message bus will not read or write any further words, and the message handling is dropped until the next message.

6. In any case in which 'Message_ended_OK' is not set at the end of the message, it is likely that the Bus Controller would retransmit that same 'receive command'.

7. A typical usage of the Echo_Error_A/B signals is to latch them into a Flip Flop and report the value of this Flip Flop to the 1553 Bus Controller through the Terminal Flag (bit 0 of the status word). The TF bit is generated from Host_Status_Bits, bit 0.

5. BRM1553FE-RT SAMPLE DESIGN

The BRM1553FE-RT sample design includes the following parts:

- RT Sample – which is a VHDL example of a design that uses the core.
- A Transceiver – VHDL model that connects the core with 2 buses.
- A bus tester – VHDL model that generates 1553 messages and checks the return replies all based on user commands.
- A top Test bench that instantiates all of these components to a working example.
- A synthesis script for the RT sample.
- A simulation script for compiling the core and running it.

5.1 TEST BENCH TOP

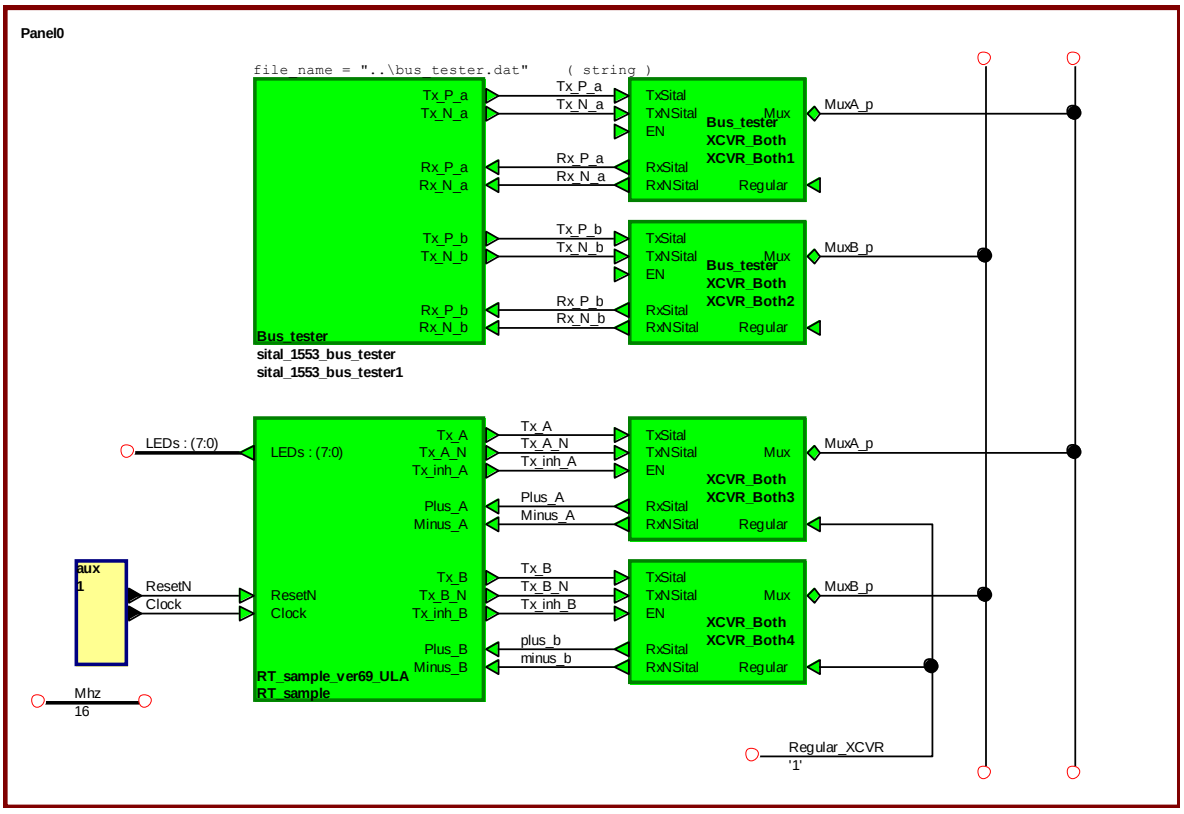

Figure 14: Test Bench Top

The RT_Sample block is driven by the clock32Mhz and resetN signals on one side, and is connected to two transceiver models on the opposite side.

The Bus Tester is connected to the same buses A and B via two transceivers.

5.2 RT SAMPLE

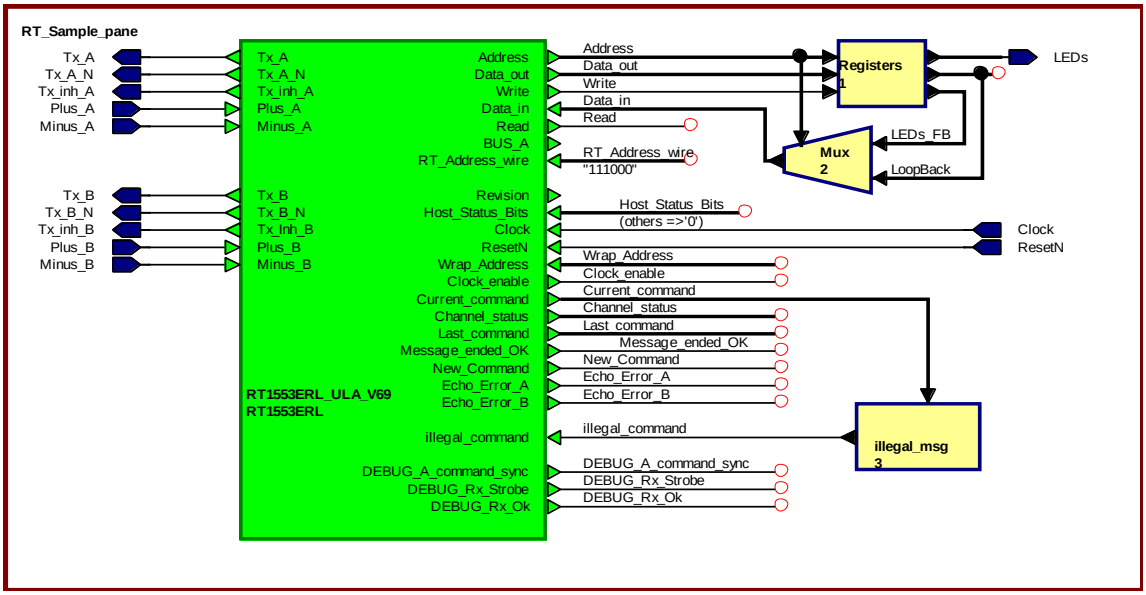

Figure 15: RT Sample Block Diagram

On the left side are the signals that connect with the transceiver, both transmit lines and receive lines.

The RT Address wire is programmed to 11000, thus this RT will respond to commands sent to address 18Hex (24).

The Wrap Address is set to 1E Hex (30), thus words sent to this sub-address will be stored inside the core, and sent back upon request of data words for sub address 30.

The core was precompiled to run with a 32 Mhz clock, and thus uses the Clock32Mhz.

The reset should come from a Q output of a FF sampled by this Clock32Mhz to assure a clean start.

The CPU style bus is connected to two registers, the LEDs and The LoopBack.

The LEDs register will latch the 8 lower bits of a 1553 word sent to the IP core using the command x"C021" which is Terminal Address 18Hex, Receive, Sub address 01 word count 01.

Note: The same command with word count bigger than 1 will also load the LEDs register from the first data word.

The loopback register is a 16 bit register that strobes the address x"F063", which means it is the forth word received to Terminal address 1E, Receive, Sub Address 03, Word count 4 or higher.

If the commands are Transmit commands rather than receive commands, for the same sub addresses, the loopback and LEDs registers values are echoed back to the core and onwards to the 1553 bus.

Note: for detailed description of the RTL design please refer to the VHDL description supplied with this manual.

5.3 BUS TESTER

The bus tester is a complex VHDL design, which is able to parse and read commands from a regular text file.

The set of commands allows the composer to simulate valid and invalid MIL-STD-1553 traffic.

The Bus Tester can be instructed to transmit words onto the bus, or receive words from the bus and verify their value.

The lexical elements of the language are as follows:

#	A commented line starts with the # character.
<*word*>	is a 4 nibble element such as AB01.
<1553 *word*>	can be described in one of the following options: C <TA T/R SA WC> Command sync with command data built from Terminal Address Transmit or Receive, Sub Address, and Word Count all in Hex. S <word> Status Sync (same as command sync) with a 4 nibble word. D <word> Data sync with a 4 nibble word.
<*message*>	Could be one or more <1553 word> elements. Message example: C 18 R 01 03 D 2222 D 3333 D 1234 S C000
<*status*>	Could be of the form： CS Clear Status NR No Response

Language supported commands:

TxA <*message*>	Transmit *message* on bus A
TxB <*message*>	Transmit *message* on bus B
ReceiveA <*message*>	receives a message on bus A.
ReceiveB <*message*>	receives a message on bus B.
checkA <*status*>	Checks Either Clear Status or No Response
CheckB <*status*>	Checks Either Clear Status or No Response

WAIT_UNTIL	RxA A word is received fully on bus A RxB A word is received fully on bus B TxA_Rdy The A transmitter double buffer is ready for another transmit TxB_Rdy The B transmitter double buffer is ready for another transmit example : WAIT_UNTIL RxA will wait until a word was received from bus A
WAIT_FOR *<time>*	Wait for time delay. Time delay could be 1 ns or 23 us or 3456 ns no decimal point in time unit
echo	40 characters echoed to screen
break	stops the program.

The commands should be located in a file called **bus_teser.dat** located in the hds2004 sub directory.

A sample bus_tester.dat file is supplied with the core, please read through and edit to your desired commands.

A simple program would look like this:

```
WAIT_FOR 2 us

ECHO transmit a receive command with two data words to the UUT.
TxA  C 18 R 01 02  D 0000  D 0001

 # wait until the data word is transmitted + 2 us.
WAIT_FOR 22 us

ECHO  wait for a reply
WAIT_UNTIL RxA

# check that the 10 LSBs of the received status is Clean Status.
CHECKA CS

# allow for an intermessage gap to the next command.
WAIT_FOR 6 us
```

6. RT_32_2 SAMPLE EDIF

The RT core is supplied both in gate level VHDL format as synthesized to an FPGA target device to be used for simulation, and an EDIF file to be used for place and route with the Vendor Place and route tool.

In order to run the VHDL gate level model, the user should create a gate level library and compile the relevant vendor gates description files to that library. A TCL script is supplied for that.

7. BRM1553D HARDWARE KEY

This chapter is not relevant for the ERL version.

Sital IP Key is required for operating Sital's IP Cores. The key uses a 6 pin small package. One pin is connected to GND (pin 1) and one pin (pin 2) should be connected to the "security pin" at the FPGA. The security key is part of the IP core and the user must assign a bi-directional pin at the FPGA to support the key.

This pin should also be connected to a pull-up resistor of 1K to 2.2K. All other pins (3, 4, 5 and 6) are not connected

7.1 OPERATING CIRCUIT

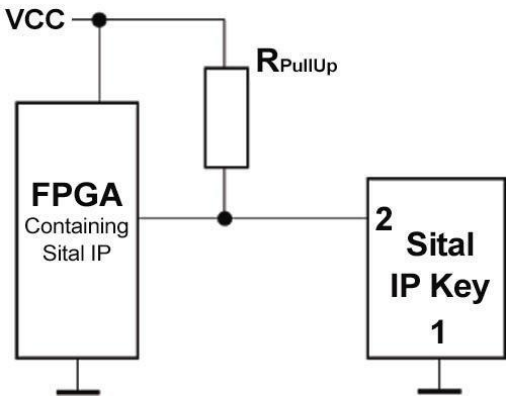

Figure 16: Hardware Key Operating Circuit

VCC supply in the range of 3V to 5.2V.

7.2 PIN CONFIGURAION

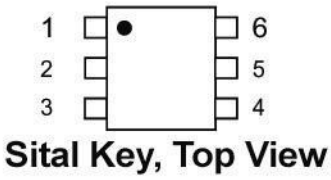

Figure 17: Hardware Key top view

Pinout:
Pin 1 ---------- GND
Pin 2 ---------- IO
All other pins --- Not Connected

7.3 HARDWARE KEY SYNTHESIS PROCEDURE

The IP core EDIF net list embeds a single PAD for the signal “to_key” that connects to the external key.

If you are synthesizing your design with a synthesizer, and you are making the 1553 IP core a black box, your synthesizer will not know that a PAD is inside the black box and embed a buffer for that pin.

You must then issue a “NOPAD” command for that pin in your VHDL code.

For every synthesizer this is unique, please refer to its manual for instructions. In Precision Synthesis, for example, use “NOPAD” attribute.

If your synthesizer reads your HDL code AND the EDIF provided for the core, this operation is not required.

7.4 HARDWARE KEY ELECTRICAL CHARACTERISTICS

(T_A = -40°C to +85°C; see Note 1.)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Pullup Voltage	V_{PUP}		2.8		5.25	V
Pullup Resistance	R_{PUP}		1		2.2	kΩ
Input Capacitance	C_{IO}	(Notes 2, 3)			1000	pF
Input Load Current	I_L	I/O pin at V_{PUP}	0.05		6.7	µA
High-to-Low Switching Threshold	V_{TL}	(Notes 3, 4, 5)	0.5		V_{PUP} – 1.8	V
Input Low Voltage	V_{IL}	(Notes 6)			0.3	V
Low-to-High Switching Threshold	V_{TH}	(Notes 3, 4, 7)	1.0		V_{PUP} –1.0	V
Switching Hysteresis	V_{HY}	(Notes 3, 4, 8)	0.21		1.70	V
Output Low Voltage	V_{OL}	At 4mA (Note 9)			0.4	V
Data Retention (Notes 10, 11)	t_{DR}	At 85°C (worst case)	40			years

Note 1: Specifications at T_A = -40°C are guaranteed by design only and not production-tested.
Note 2: Maximum value represents the internal parasite capacitance when V_{PUP} is first applied. If a 2.2kΩ resistor is used to pull up the data line, 2.5µs after V_{PUP} has been applied the parasite capacitance will not affect normal communications.
Note 3: Guaranteed by design, characterization and/or simulation only. Not production tested.
Note 4: V_{TL}, V_{TH}, and V_{HY} are a function of the internal supply voltage, which is itself a function of V_{PUP}, R_{PUP}, and capacitive loading on IO. Lower V_{PUP}, higher R_{PUP}, and heavier capacitive loading all lead to lower values of V_{TL}, V_{TH}, and V_{HY}.
Note 5: Voltage below which, during a falling edge on IO, a logic 0 is detected.
Note 6: The voltage on IO needs to be less or equal to $V_{IL(MAX)}$ at all times the FPGA is driving IO to a logic-0 level.
Note 7: Voltage above which, during a rising edge on IO, a logic 1 is detected.
Note 8: After V_{TH} is crossed during a rising edge on IO, the voltage on IO has to drop by at least V_{HY} to be detected as logic '0'.
Note 9: The I-V characteristic is linear for voltages less than 1V.
Note 10: Data retention is degraded as T_A increases.
Note 11: Guaranteed by 100% production test at elevated temperature for a shorter time; equivalence of this production test to data sheet limit at operating temperature range is established by reliability testing.

8. ORDERING INFORMATION

Part Number (*) Optional parameters (*)

BRM1553FE-RT-S-KP-AL-66

Part Number	Clock Frequency (Optional)
12 200	Number represents clock frequency in MHz. Any even number between 12 and 200 is available.

Part Number	FPGA Selection (Optional)
AL	Altera
XL	Xilinx
LT	Lattice
AC	Actel
QL	QuickLogic

Part Number	Licensing Option
KP	Hardware Key, Lead-Free
KC	Hardware Key, No Lead-Free
ST	Sticker (for ERL), can be left blank

Part Number	Transceiver Option
S	Sital Discrete Transceiver
T	Any Other Transceiver
	<blank> Any Other Transceiver

Part Number	IP Core Type
BC	Bus Controller
RT	Remote Terminal
MT	Monitor Terminal
RTMT	Remote Terminal & Monitor Terminal
BCRTMT	Bus Controller, Remote Terminal & Monitor

Part Number	IP Core
FE	Front End
ERL	Extended Reliability

(*) Part Number must include IP Core Type, Transceiver Option and Licensing Option.
FPGA Selection, Memory Configuration and Clock Frequency are optional parameters.

APPENDIX A: CHANGES TRACKING

List of changes in this document

From Version 2.3:

Added Parity bit check for Terminal address lines.

Corrected a couple of syntax errors.

From Version 2.5:

Added illegalization support.

Added BIT word import.

Update some signal's names.

17-Apr-2013:

Added a note regarding ERL version.

17 Atir Yeda St., Kfar-Saba, ISRAEL 44643

Email: info@sitaltech.com

Website: http://www.sitaltech.com

The information provided in this User's Guide is believed to be accurate; however, no responsibility is assumed by Sital Technology for its use, and no license or rights are granted by implication or otherwise in connection therewith. Specifications are subject to change without notice.

Please visit our Web site at www.sitaltech.com for the latest information.