VERIFICATION PLAN DOCUMENT for WISHBONE2UART

# EVOLUTION OF DOCUMENT

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Modifications** |
| 0.1 | 14.08.  2019 | Aleksandar Komazec | Initial release |
| 0.2 | 17.08 | Aleksandar Komazec | Checkers are updated |
| 0.3 | 26.08 | Aleksandar Komazec | Wishbone checkers are updated |
| 0.4 | 28.08 | Aleksandar Komazec | Chapters 1,2,3 and 4 have been updated |
| 0.5 | 17.09 | Aleksandar Komazec | Chapters 2,3 and 4 have been updated |
| 0.5 | 11.10 | Aleksandar Komazec | Chapter 6 has been updated |

# REFERENCES

**DUT documents:**

DUT’s specification documents:

1. UART\_spec.pdf
2. UART\_Protocol.pdf

DUT’s other documents:

* 16550.pdf

Protocol specification documents:

* WISHBONE\_B4\_Protocol.pdf

Protocol other documents:

* 8251-08-serial.pdf
* serial-communication-uart.pdf

# TABLE OF CONTENTS

[EVOLUTION OF DOCUMENT 2](#__RefHeading___Toc5683_1062397625)

[REFERENCES 3](#__RefHeading___Toc5685_1062397625)

[TABLE OF CONTENTS 4](#__RefHeading___Toc5687_1062397625)

[1. INTERFACES 6](#__RefHeading___Toc5689_1062397625)

[1.1 UART Interface 6](#__RefHeading___Toc5691_1062397625)

[1.1.1 Checker list 6](#__RefHeading___Toc5693_1062397625)

[1.1.2 Coverage list 6](#__RefHeading___Toc5695_1062397625)

[1.2 WISHBONE Interface 7](#__RefHeading___Toc5732_1062397625)

[1.2.1 Checker list 7](#__RefHeading___Toc5693_10623976251)

[1.2.2 Coverage list 9](#__RefHeading___Toc5695_10623976251)

[2. FUNCTIONAL FEATURES 10](#__RefHeading___Toc5697_1062397625)

[2.1 Master operation 10](#__RefHeading___Toc5699_1062397625)

[2.1.1 Checker list 10](#__RefHeading___Toc5701_1062397625)

[2.1.2 Coverage list 12](#__RefHeading___Toc5703_1062397625)

[3. REGISTER ACCESS 13](#__RefHeading___Toc5705_1062397625)

[3.1 Receiver Buffer Register (RBR) 13](#__RefHeading___Toc5707_1062397625)

[3.1.1 Checker list 13](#__RefHeading___Toc5709_1062397625)

[13](#__RefHeading___Toc7457_2099151742)

[3.1.2 Coverage list 14](#__RefHeading___Toc5711_1062397625)

[3.2 Transmitter Holding Register (THR) 14](#__RefHeading___Toc5832_1062397625)

[3.2.1 Checker list 14](#__RefHeading___Toc5709_10623976251)

[3.2.2 Coverage list 14](#__RefHeading___Toc5711_10623976251)

[3.3 Interrupt Enable Register (IER) 15](#__RefHeading___Toc5834_1062397625)

[3.3.1 Checker list 15](#__RefHeading___Toc5709_10623976252)

[3.3.2 Coverage list 15](#__RefHeading___Toc5711_10623976252)

[3.4 Interrupt Identification Register (IIR) 15](#__RefHeading___Toc5836_1062397625)

[3.4.1 Checker list 15](#__RefHeading___Toc5709_10623976253)

[3.4.2 Coverage list 15](#__RefHeading___Toc5711_10623976253)

[3.5 FIFO Control Register (FCR) 16](#__RefHeading___Toc5838_1062397625)

[3.5.1 Checker list 16](#__RefHeading___Toc5709_10623976254)

[3.5.2 Coverage list 16](#__RefHeading___Toc5711_10623976254)

[3.6 Line Control Register (LCR) 17](#__RefHeading___Toc5840_1062397625)

[3.6.1 Checker list 17](#__RefHeading___Toc5709_10623976255)

[3.6.2 Coverage list 17](#__RefHeading___Toc5711_10623976255)

[3.7 Modem Control Register (MCR) 17](#__RefHeading___Toc5946_1062397625)

[3.7.1 Checker list 17](#__RefHeading___Toc5709_10623976256)

[Coverage list 18](#__RefHeading___Toc5711_10623976256)

[3.8 Line Status Register (LSR) 18](#__RefHeading___Toc5844_1062397625)

[3.8.1 Checker list 18](#__RefHeading___Toc5709_10623976257)

[3.8.2 Coverage list 18](#__RefHeading___Toc5711_10623976257)

[18](#__RefHeading___Toc5846_1062397625)

[3.9 Divisor Latch Register (DLR byte 1) 19](#__RefHeading___Toc5958_1062397625)

[3.9.1 Checker list 19](#__RefHeading___Toc5709_10623976258)

[3.9.2 Coverage list 19](#__RefHeading___Toc5711_10623976258)

[3.10 Divisor Latch Register (DLR byte 2) 19](#__RefHeading___Toc5848_1062397625)

[3.10.1 Checker list 19](#__RefHeading___Toc5709_10623976259)

[3.10.2 Coverage list 20](#__RefHeading___Toc5711_10623976259)

[3.11 Debug Register 1 (DR1) 20](#__RefHeading___Toc5850_1062397625)

[3.11.1 Checker list 20](#__RefHeading___Toc5709_106239762510)

[3.11.2 Coverage list 20](#__RefHeading___Toc5711_106239762510)

[3.12 Debug Register 2 (DR2) 20](#__RefHeading___Toc5852_1062397625)

[3.12.1 Checker list 20](#__RefHeading___Toc5709_106239762511)

[3.12.2 Coverage list 21](#__RefHeading___Toc5711_106239762511)

[3.13 Modem Status Register 21](#__RefHeading___Toc5854_1062397625)

[3.13.1 Checker list 21](#__RefHeading___Toc5709_106239762512)

[3.13.2 Coverage list 21](#__RefHeading___Toc5711_106239762512)

[4. RESET AND CLOCK 22](#__RefHeading___Toc5713_1062397625)

[4.1 Reset 22](#__RefHeading___Toc5948_1062397625)

[4.1.1 Checker list 22](#__RefHeading___Toc5709_1062397625121)

[4.1.2 Coverage list 22](#__RefHeading___Toc5711_1062397625121)

[4.2 Clock 23](#__RefHeading___Toc5950_1062397625)

[4.2.1 Checker list 23](#__RefHeading___Toc5709_1062397625122)

[4.2.2 Coverage list 23](#__RefHeading___Toc5711_1062397625122)

[5. DEBUG AND TEST 24](#__RefHeading___Toc5715_1062397625)

[6. TEST LIST 24](#__RefHeading___Toc5717_1062397625)

[6.1 Register\_access\_test\_tc 24](#__RefHeading___Toc7459_2099151742)

[6.2 Interrupt\_triger\_test\_tc 24](#__RefHeading___Toc7461_2099151742)

[6.3 Indicator\_line\_trigger\_test\_tc 25](#__RefHeading___Toc7463_2099151742)

[6.4 Fifo\_wr\_rd\_test\_tc 25](#__RefHeading___Toc7465_2099151742)

## INTERFACES

### UART Interface

#### Checker list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Checker name | Description | Implementation Details |
| 1 | knwn\_stx\_srx\_chk | Value on STX\_PAD\_O and SRX\_PAD\_O must be known for the time spent on stop bit, data bits,parity bits and stop bits | Protocol checker in interface |
| 2 | dsr\_changes\_dtr\_follows\_chk | When DSR\_PAD\_I is changed  DTR\_PAD\_O must follow and be changed too. | Protocol checker in interface |
| 3 | cts\_changes\_rts\_follows\_chk | When CTS\_PAD\_I is changed RTS\_PAD\_O must follow and be changed too | Protocol checker in interface |

#### Coverage list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Coverage name | Description | Implementation Details |
| 1 | stx\_value\_cov | Values 0 and 1 done the coverage of STX\_PAD\_O | Protocol coverage in monitor |
| 2 | srx\_value\_cov | Values 0 and 1 done the coverage of SRX\_PAD\_I | Protocol coverage in monitor |
| 4 | dsr\_modem\_cov | Values 0 and 1 done the coverage of DSR\_PAD\_I | Protocol coverage in monitor |
| 5 | dtr\_modem\_cov | Values 0 and 1 done the coverage of DTR\_PAD\_O | Protocol coverage in monitor |
| 6 | dtr\_dsr\_modem\_p1\_cross\_cov | It will cover:  dsr\_modem\_cov = 1 and dtr\_modem\_cov =1 | Protocol coverage in monitor |
| 7 | dtr\_dsr\_modem\_p0\_cross\_cov | It will cover:  dsr\_modem\_cov = 0 and dtr\_modem\_cov =0 | Protocol coverage in monitor |
| 7 | cts\_modem\_cov | Values 0 and 1 done the coverage of CTS\_PAD\_I | Protocol coverage in monitor |
| 8 | rts\_modem\_cov | Values 0 and 1 done the coverage of RTS\_PAD\_O | Protocol coverage in monitor |
| 9 | cts\_rts\_modem\_p1\_cross\_cov | It will cover:  cts\_modem\_cov = 1 and rts\_modem\_cov = 1 | Protocol coverage in monitor |
| 10 | cts\_rts\_modem\_p0\_cross\_cov | It will cover:  cts\_modem\_cov = 0 and rts\_modem\_cov = 0 | Protocol coverage in monitor |

### WISHBONE Interface

#### Checker list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Checker name | Description | Implementation Details |
| 1 | sig\_valid\_until\_stb\_chk | WB\_ADDR\_I, WB\_WE\_I , WB\_SEL\_I, WB\_DAT\_I, TGA\_I, TGC\_I must be valid (stable and 0 or 1) during WB\_STB is active. | Protocol checker in interface |
| 2 | cyc\_held\_until\_end\_chk | In standard mode, WB\_CYC\_I must be held until transfer end (WB\_STB\_I negedge) | Protocol checker in interface |
| 3 | cyc\_held\_until\_end\_pip\_mode\_chk | In pipelined mode, WB\_CYC\_I must be held until transfer end (WB\_ACK\_O negedge) | Protocol checker in interface |
| 4 | ack\_low\_stb\_cyc\_raise\_chk | WB\_ACK\_O must be low when WB\_STB\_I and WB\_CYC\_I are raised. | Protocol checker in interface |
| 5 | stb\_held\_until\_ack\_chk | In standard mode, WB\_STB\_I must be held until WB\_ACK\_O. | Protocol checker in interface |
| 6 | data\_valid\_while\_ack\_chk | WB\_DATA\_O must be valid (stable and known (0 or 1)) when WB\_ACK\_O is high. | Protocol checker in interface |
| 7 | stb\_stable\_stall\_high\_chk | In pipelined mode, WB\_STB\_I must be stable when WB\_STALL\_O is high | Protocol checker in interface |
| 8 | lock\_held\_until\_end\_chk | In standard mode, WB\_LOCK\_I must be held until transfer end (WB\_STB\_I negedge) | Protocol checker in interface |
| 9 | lock\_held\_until\_end\_pip\_mode\_chk | In pipelined mode, WB\_LOCK\_I must be held until transfer end (WB\_ACK\_O negedge) | Protocol checker in interface |

#### Coverage list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Coverage name | Description | Implementation Details |
| 1 | wb\_stb\_cov | It covers all values (0,1) of WB\_STB\_O | Functional coverage in monitor |
| 2 | wb\_cyc\_cov | It covers all values (0,1) of WB\_CYC | Functional coverage in monitor |
| 3 | stb\_cyc\_cross\_cov | It mixes them:  cov\_WB\_STB  cov\_WB\_CYC | Functional coverage in monitor |
| 4 | data\_input\_8\_cov | All values different from reset value will cover this in 8 bit data mode | Functional coverage in monitor |
| 5 | data\_input\_32\_cov | All values different from reset value will cover this in 32 bit data mode | Functional coverage in monitor |
| 6 | we\_cov | It covers all states of WE | Functional coverage in monitor |
| 7 | data\_i\_8\_we\_cross\_cov | It mixes cov\_WE = and data\_input\_8\_cov | Functional coverage in monitor |
| 8 | data\_i\_32\_we\_cross\_cov | It mixes cov\_WE = and data\_input\_32\_cov | Functional coverage in monitor |
| 9 | data\_output\_8\_cov | All values different from reset value will cover this in 8 bit data mode | Functional coverage in monitor |
| 10 | data\_output\_32\_cov | All values different from reset value will cover this in 32 bit data mode | Functional coverage in monitor |

## FUNCTIONAL FEATURES

### Master operation

#### Checker list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Checker name | Description | Implementation Details |
| 1 | transmit\_fifo\_chk | Data loaded to transmiter fifo must be the same like data from STX\_PAD\_O | Implemented in scoreboard |
| 2 | transmit\_fifo\_data\_known\_chk | Data transmited from the transmiter fifo over UART must be 0 or 1 | Implemented in scoreboard |
| 3 | transmit\_fifo\_THRE\_flag\_chk | THRE(Transmitter Holding Register Empty) interrupt flag will be set if the fifo transmitter is empty. | Implemented in scoreboard |
| 4 | receive\_fifo\_chk | Data loaded to receive fifo will be transmitted over UART | Implemented in scoreboard |
| 5 | receive\_fifo\_data\_known\_chk | Data transmited from the receiver fifo over wishbone must be 0 or 1 | Implemented in scoreboard |
| 6 | receive\_fifo\_RDA\_flag\_chk | RDA (Receiver data available) flag will be set if fifo trigger level is reached. | Implemented in scoreboard |
| 7 | receive\_fifo\_TI\_flag\_chk | TI (Timeout indication) flag will be set if there is at least 1 character in receiver fifo but no character has been input to the receiver fifo or read from it for the last 4 character times. | Implemented in scoreboard |
| 8 | receive\_fifo\_RLS\_flag\_chk | RLS (Receiver Line Status) flag will be set if parity, overrun or farming error or Break interrupt is happened | Implemented in scoreboard |
| 9 | receive\_fifo\_DR\_flag\_chk | DR (Data Ready) flag will be set if at least one character has been received to the receiver fifo. | Implemented in scoreboard |
| 10 | receive\_fifo\_OE\_flag\_chk | OE (Overrun Error) flag will be set if the receiver fifo is full. | Implemented in scoreboard |
| 11 | receive\_fifo\_PE\_flag\_chk | PE (Parity Error) flag will be set if parity error occurs in the receiver fifo. | Implemented in scoreboard |
| 12 | receive\_fifo\_FE\_flag\_chk | FE (Farming Error) flag will be set if stop bit that is received is not valid. | Implemented in scoreboard |
| 13 | receive\_fifo\_BI\_flag\_chk | BI (Break Interrupt) flag will be set when SRX\_PAD\_I line is held in logic 0 for a time of one chaacter | Implemented in scoreboard |
| 14 | transmit\_empty\_indicator\_chk | Transmitter Empty Indicator will be set when transmitter fifo and transmitter shift register are empy. | Implemented in scoreboard |
| 15 | receive\_data\_error\_bit\_chk | This bit will be set as soon as any data character in the UART 16550 IP design has parity or framing error or the break indication active. | Implemented in scoreboard |
| 16 | receiver\_fifo\_chk | Data loaded to SRX\_PAD\_O receiver fifo must be the same like data from WB\_DAT\_O | Implemented in scoreboard |
| 17 | receiver\_fifo\_data\_known\_chk | Data sent to SRX\_PAD\_O must be 0 or 1 | Implemented in scoreboard |

#### Coverage list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Coverage name | Description | Implementation Details |
| 1 | uart\_data\_transmit\_stx\_pad\_o\_cov | All values in Fifo Transmitter that are different from reset value will cover it | Functional coverage in monitor |
| 1 | uart\_data\_receive\_srx\_pad\_o\_cov | All values in Fifo Receiver that are different from reset value will cover it | Functional coverage in monitor |
| 3 | uart\_num\_of\_bits\_cov | m\_number\_of\_bits LCR [1:0] value (5,6,7,8) covers it | Functional coverage in monitor |
| 4 | uart\_number\_of\_stop\_bits\_cov | m\_number\_of\_stop\_bits value LCR [2] (1, 1.5 , 2) covers it | Functional coverage in monitor |
| 5 | uart\_parity\_enable\_cov | m\_parity\_enable value LCR [3] (0,1) covers it | Functional coverage in monitor |
| 6 | uart\_parity\_type\_cov | m\_parity\_type value LCR [4] (0,1) covers it | Functional coverage in monitor |
| 7 | uart\_stick\_parity\_cov | m\_stick\_parity value LCR [5](0,1) covers it | Functional coverage in monitor |
| 8 | uart\_break\_control\_cov | m\_break\_control value LCR [6](0,1) covers it | Functional coverage in monitor |

## REGISTER ACCESS

### Receiver Buffer Register (RBR)

#### Checker list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Checker name | Description | Implementation Details |
| 1 | RBR\_rst\_val\_chk | Reset value of RBR should be 00h | Implemented in monitor |
| 2 | RBR\_read\_only\_chk | RBR register is read only register. Any write access will not change value. 00h should be read always. | Implemented in monitor |
| 3 | RBR\_read\_knwn\_value\_chk | Value read from RBR must be known (0 or 1) | Implemented in monitor |

#### 

#### Coverage list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Coverage name | Description | Implementation Details |
| 1 | RBR \_read\_cov | 0 will cover WB\_WE\_I. | Implemented in monitor |
| 2 | RBR\_value\_cov | All values that are received to RBR different from 0x00 will cover it. | Implemented in  monitor |

### Transmitter Holding Register (THR)

#### Checker list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Checker name | Description | Implementation Details |
| 1 | THR\_rst\_val\_chk | Reset value of THR should be 00h | Implemented in monitor |
| 2 | THR\_write\_only\_chk | THR register is write only register. Any write access will change value. | Implemented in monitor |

#### Coverage list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Coverage name | Description | Implementation Details |
| 3 | THR\_value\_cov | Cover that write access to Transmitter holding register is done with data different than 0xFF. | Implemented in  monitor |

### Interrupt Enable Register (IER)

#### Checker list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Checker name | Description | Implementation Details |
| 1 | IER\_rst\_val\_chk | Reset value of IER should be 00h | Implemented in monitor |
| 2 | IER\_same\_value\_chk | Value read from IER must be the same as the written value. | Implemented in monitor |
| 3 | IER\_read\_knwn\_value\_chk | Value read from IER must be known (0 or 1) | Implemented in monitor |

#### Coverage list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Coverage name | Description | Implementation Details |
| 1 | IER\_value\_cov | The address data range can be 0x01 up to 0x0F. | Implemented in monitor |

### Interrupt Identification Register (IIR)

#### Checker list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Checker name | Description | Implementation Details |
| 1 | IIR\_rst\_val\_chk | Reset value of THR should be C1h | Implemented in scoreboard |
| 2 | IIR\_read\_only\_chk | IIR register is read only register. Any write access will not change value. C1h should be read always. | Implemented in scoreboard |
| 3 | IIR\_read\_knwn\_value\_chk | Value read from IIR must be known (0 or 1) | Implemented in monitor |
| 3 | IIR\_range\_value\_chk | Read value from IIR should be from  C0h to CDh | Implemented in scoreboard |

#### Coverage list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Coverage name | Description | Implementation Details |
| 1 | IR\_value\_cov | All values that are different from reset value (C1h ) are included in this coverage. | Implemented in monitor |

### FIFO Control Register (FCR)

#### Checker list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Checker name | Description | Implementation Details |
| 1 | FCR\_rst\_val\_chk | Reset value of THR should be C0h‬ | Implemented in scoreboard |
| 2 | FCR \_wr\_only\_chk | FCR register is write only register. Any write access will change value. | Implemented in scoreboard |

#### Coverage list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Coverage name | Description | Implementation Details |
| 1 | FCR\_value\_cov | All values different from the reset value (11000000b) will included in coverage. | Implemented in  monitor |

### Line Control Register (LCR)

#### Checker list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Checker name | Description | Implementation Details |
| 1 | LCR\_rst\_val\_chk | Reset value of LCR should be 03h | Implemented in monitor |
| 2 | LCR\_same\_value\_chk | Value read from LCR must be the same as the written value. | Implemented in monitor |
| 3 | LCR\_read\_knwn\_value\_chk | Value read from LCR must be known (0 or 1) | Implemented in monitor |

#### Coverage list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Coverage name | Description | Implementation Details |
| 1 | LCR\_value\_cov | Value of Line control register that is different from 00000011b will be included in coverage. | Implemented in monitor |

### Modem Control Register (MCR)

#### Checker list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Checker name | Description | Implementation Details |
| 1 | MCR\_rst\_val\_chk | Reset value of MCR should be 00h | Implemented in scoreboard |
| 2 | MCR \_wr\_only\_chk | MCR register is write only register. Any write access will change value. | Implemented in scoreboard |
| 3 | MCR\_same\_value\_chk | Value read from MCR must be the same as the written value. | Implemented in monitor |
| 4 | MCR\_read\_knwn\_value\_chk | Value read from MCR must be known (0 or 1) | Implemented in monitor |

#### Coverage list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Coverage name | Description | Implementation Details |
| 1 | MCR\_value\_cov | All values of different from 0x00 will cover it. | Implemented in  monitor |

### Line Status Register (LSR)

#### Checker list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Checker name | Description | Implementation Details |
| 1 | LSR\_rst\_val\_chk | Reset value of LSR should be 60h | Implemented in scoreboard |
| 2 | LSR\_read\_only\_chk | LSR register is read only register. Any write access will not change value. 60h should be read always. | Implemented in scoreboard |
| 3 | LSR\_read\_knwn\_value\_chk | Value read from LSR must be known (0 or 1) | Implemented in monitor |

#### Coverage list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Coverage name | Description | Implementation Details |
| 1 | LSR\_value\_cov | All values different from 01100000 will be included into coverage. | Implemented in  monitor |

### 

### Divisor Latch Register (DLR byte 1)

#### Checker list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Checker name | Description | Implementation Details |
| 1 | DLR\_byte\_one\_rst\_val\_chk | Reset value of DLR byte 1 should be 00h | Implemented in Scoreboard |
| 2 | DLR\_byte\_one same\_value\_chk | Value read from DLR\_byte\_one must be the same as the written value. | Implemented in monitor |
| 3 | DLR\_byte\_one\_value\_chk | Value read from DLR\_byte\_one must be known (0 or 1) | Implemented in monitor |

#### Coverage list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Coverage name | Description | Implementation Details |
| 1 | DLC1\_vvalue\_cov | Value of Divisor Latch Register (byte 1) that is different from reset value will be included into coverage. | Implemented in monitor |

### Divisor Latch Register (DLR byte 2)

#### Checker list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Checker name | Description | Implementation Details |
| 1 | DLR\_byte\_two\_rst\_val\_chk | Reset value of DLR byte 2 should be 00h | Implemented in Scoreboard |
| 2 | DLR\_byte\_two same\_value\_chk | Value read from DLR\_byte\_two must be the same as the written value. | Implemented in monitor |
| 3 | DLR\_byte\_two\_value\_chk | Value read from DLR\_byte\_two must be known (0 or 1) | Implemented in monitor |

#### Coverage list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Coverage name | Description | Implementation Details |
| 1 | DLR2\_vvalue\_cov | Value of Divisor Latch Register (byte 2) that is different from reset value will be included into coverage. | Implemented in monitor |

### Debug Register 1 (DR1)

#### Checker list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Checker name | Description | Implementation Details |
|  |  |  |  |

#### Coverage list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Coverage name | Description | Implementation Details |
| 1 | DR1\_value\_cov | All 32 bit values will be included in DR1 coverage | Implemented in monitor |

### Debug Register 2 (DR2)

#### Checker list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Checker name | Description | Implementation Details |
|  |  |  |  |

#### Coverage list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Coverage name | Description | Implementation Details |
|  |  |  |  |

### Modem Status Register

#### Checker list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Checker name | Description | Implementation Details |
| 1 | MSR\_rst\_val\_chk | Reset value of DLR byte 2 should be x0h | Implemented in scoreboard |
| 2 | MSR\_read\_only\_chk | MSR register is read only register. Any write access will not change value. x0h should be read always. | Implemented in scoreboard |
| 3 | MSR\_value\_chk | Value read from MSR must be known (0 or 1) | Implemented in monitor |

#### Coverage list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Coverage name | Description | Implementation Details |
| 1 | MSR\_value\_cov | All values of modem control register different from xxxx0000 will cover it. | Implemented in  monitor |

## RESET AND CLOCK

### Reset

#### Checker list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Checker name | Description | Implementation Details |
| 1 | reset\_data\_stable\_chk | While WB\_RST\_I is active, WB\_STB , WB\_CYC must be negated. | Protocol checker in interface |
| 2 | reset\_at\_least\_chk | WB\_RST\_I must be asserted for at least one complete clock cycle. | Protocol checker in interface |
| 3 | pow\_condition\_rst\_clk | WB\_RST\_I must be high at the beginning of the simulation | Protocol checker in interface |

#### Coverage list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Coverage name | Description | Implementation Details |
| 1 | rst\_cov | It covers all values of reset | Coverage in monitor |

### Clock

#### Checker list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Checker name | Description | Implementation Details |
| 1 | r\_edge\_signal\_clk\_chk | All Wishbone signals have to be sensitive at rising edge | Protocol checker in interface |
| 2 | clk\_isnt\_working\_chk | It checks that the interface isn’t reacting to WB\_RST when the clock is stopped. | Protocol checker in interface |

#### Coverage list

|  |  |  |  |
| --- | --- | --- | --- |
| Num | Coverage name | Description | Implementation Details |
| 1 | clk\_cov | It covers all values of clock | Coverage in monitor |

## DEBUG AND TEST

*This chapter has to contain list of checkers and coverage items related to debug and test features.*

## TEST LIST

6.1 Register\_access\_test\_tc

6.2 Interrupt\_trigger\_test\_tc

6.3 Indicator\_line\_trigger\_test\_tc

6.4 Fifo\_wr\_rd\_test\_tc

### Register\_access\_test\_tc

* Read the value from all the registers in the UART ip core after reset happens.
* Write value different from reset value to all the registers.
* Then, read value from read only registers has to be reset value (not a new written value), and read value from write only registers has to be different from the current value

### Interrupt\_triger\_test\_tc

* Enable all interrupts in Interrupt enable register.
* Make a situation where all of interrupts will be triggered one by one.
* Then, make a situation where all of interrupts will be shut off one by one respectively.

### Indicator\_line\_trigger\_test\_tc

* Make a situation where all of error indicators (LSR bits) will be triggered.
* Then, make a situation where all of error indicators will be shut off.

### Fifo\_wr\_rd\_test\_tc

* Configure divisor latches in order to set baud rate
* Write randomized 16 data to fifo transmitter in order to assure that UART IP provides the same data on wishbone interface.
* Write randomized 16 (maximum depth) data to fifo receiver in order to assure that data from UART protocol and wishbone protocol are the same