Board Support Package for Columbus Project

CDL Test Plan

Version 0.1

Brite Semiconductor

July 13, 2016

Table of Contents

[1. Introduction 5](#_Toc456256904)

[2. Testing System Overview 6](#_Toc456256905)

[3. Testing Case Design 7](#_Toc456256906)

[3.1 Timer 7](#_Toc456256907)

[3.1.1 Test Case cb-timer-1: timer start/stop test 7](#_Toc456256908)

[3.1.2 Test Case cb-timer-2: timer reload test 7](#_Toc456256909)

[3.1.3 Test Case cb-timer-3: timer count mode test 8](#_Toc456256910)

[3.1.4 Test Case cb-timer-4: timer run mode test 8](#_Toc456256911)

[3.1.5 Test Case cb-timer-5: timer free run mode test 9](#_Toc456256912)

[3.1.6 Test Case cb-timer-6: timer frequency division test 9](#_Toc456256913)

[3.1.7 Test Case cb-timer-7: timer interrupt test 10](#_Toc456256914)

[3.1.8 Test Case cb-timer-8: timer capture mode test 10](#_Toc456256915)

[3.1.9 Test Case cb-timer-9: timer compare mode test 11](#_Toc456256916)

[3.1.10 Test Case cb-timer-10: timer pwm mode test 11](#_Toc456256917)

[3.2 Watch Dog Timer 12](#_Toc456256918)

[3.2.1 Test Case cb-wdt-1: wdt start/stop test 12](#_Toc456256919)

[3.2.2 Test Case cb-wdt-2: wdt load value test 12](#_Toc456256920)

[3.2.3 Test Case cb-wdt-3: wdt restart test 12](#_Toc456256921)

[3.2.4 Test Case cb-wdt-4: wdt generate reset test 13](#_Toc456256922)

[3.2.5 Test Case cb-wdt-5: wdt frequency division test 14](#_Toc456256923)

[3.2.6 Test Case cb-wdt-6: wdt interrupt generate test 14](#_Toc456256924)

[3.2.7 Test Case cb-wdt-7: wdt interrupt handle test 15](#_Toc456256925)

[3.3 RTC/BBU 15](#_Toc456256926)

[3.3.1 Test Case cb-rtc-1: 15](#_Toc456256927)

[3.4 UART 16](#_Toc456256928)

[3.4.1 Test Case cb-uart-1: loopback transmit/receive test 16](#_Toc456256929)

[3.4.2 Test Case cb-uart-2: loopback modem control test 16](#_Toc456256930)

[3.4.3 Test Case cb-uart-3: basic send test 17](#_Toc456256931)

[3.4.4 Test Case cb-uart-4: basic receive test 17](#_Toc456256932)

[3.4.5 Test Case cb-uart-5: basic rs232 transmit / receive test 18](#_Toc456256933)

[3.4.6 Test Case cb-uart-6: baudrate setting test 19](#_Toc456256934)

[3.4.7 Test Case cb-uart-7: stop bits setting test 19](#_Toc456256935)

[3.4.8 Test Case cb-uart-8: parity bits setting test 20](#_Toc456256936)

[3.4.9 Test Case cb-uart-9: data bits setting test 21](#_Toc456256937)

[3.4.10 Test Case cb-uart-10: hardware flowctrol test 22](#_Toc456256938)

[3.4.11 Test Case cb-uart-11: dma mode rs232 transmit / receive test 22](#_Toc456256939)

[3.4.12 Test Case cb-uart-12: high baudrate setting test 23](#_Toc456256940)

[3.4.13 Test Case cb-uart-13: modem control test 23](#_Toc456256941)

[3.4.14 Test Case cb-uart-14: line break test 24](#_Toc456256942)

[3.4.15 Test Case cb-uart-15: interrupt test 25](#_Toc456256943)

[3.5 GIC/INTERRUPT 25](#_Toc456256944)

[3.5.1 Test Case cb-gic-1: cpu internal interrupt(PPI) test 25](#_Toc456256945)

[3.5.2 Test Case cb-gic-2: external interrupt(SPI) generate test 26](#_Toc456256946)

[3.6 DMAC 26](#_Toc456256947)

[3.6.1 Test Case cb-dma-1: mem to mem transfer 26](#_Toc456256948)

[3.6.2 Test Case cb-dma-2: mem to mem with diffrent configuration 27](#_Toc456256949)

[3.6.3 Test Case cb-dma-3: dma channel test 29](#_Toc456256950)

[3.6.4 Test Case cb-dma-4: multi-block transfer test 30](#_Toc456256951)

[3.6.5 Test Case cb-dma-5: memory to peripheral test 31](#_Toc456256952)

[3.6.6 Test Case cb-dma-6: peripheral to memory test 31](#_Toc456256953)

[3.6.7 Test Case cb-dma-7: dma interrupt test 32](#_Toc456256954)

[3.6.8 Test Case cb-dma-8: multi-channel transfer test 32](#_Toc456256955)

[3.7 GPIO 33](#_Toc456256956)

[3.7.1 Test Case cb-gpio-1: gpio output test 33](#_Toc456256957)

[3.7.2 Test Case cb-gpio-2: gpio input test 33](#_Toc456256958)

[3.7.3 Test Case cb-gpio-3: input interrupt generate test 34](#_Toc456256959)

[3.7.4 Test Case cb-gpio-4: interrupt type test 35](#_Toc456256960)

[3.8 I2C 36](#_Toc456256961)

[3.8.1 Test Case cb-i2c-1: basic master/slave mode write read test 36](#_Toc456256962)

[3.8.2 Test Case cb-i2c-2: speed mode test 37](#_Toc456256963)

[3.8.3 Test Case cb-i2c-3: device address mode test 38](#_Toc456256964)

[3.8.4 Test Case cb-i2c-4: external dma support test 38](#_Toc456256965)

[3.8.5 Test Case cb-i2c-5: interrupt test 39](#_Toc456256966)

[3.8.6 Test Case cb-i2c-6: master mode write read using real peripheral 40](#_Toc456256967)

[3.8.7 Test Case cb-i2c-7: slave address test 40](#_Toc456256968)

[3.8.8 Test Case cb-i2c-8: robust test 41](#_Toc456256969)

[3.9 QSPI 42](#_Toc456256970)

[3.9.1 Test Case cb-qspi-1: qspi flash probe test 42](#_Toc456256971)

[3.9.2 Test Case cb-qspi-2: qspi flash erase test 42](#_Toc456256972)

[3.9.3 Test Case cb-qspi-3: qspi flash write/read test 43](#_Toc456256973)

[3.9.4 Test Case cb-qspi-4: dma mode test 43](#_Toc456256974)

[3.9.5 Test Case cb-qspi-5: baudrate setting test 44](#_Toc456256975)

[3.9.6 Test Case cb-qspi-6: interrupt test 44](#_Toc456256976)

[3.9.7 Test Case cb-qspi-7: qspi direct map mode test 45](#_Toc456256977)

[3.9.8 Test Case cb-qspi-8: qspi legacy mode flash access test 45](#_Toc456256978)

[3.9.9 Test Case cb-qspi-9: qspi direct/indirect mix mode test 46](#_Toc456256979)

[3.9.10 Test Case cb-qspi-10: qspi bit mode test 46](#_Toc456256980)

[3.9.11 Test Case cb-qspi-11: standard spi operation test 47](#_Toc456256981)

[3.10 SPI 48](#_Toc456256982)

[3.10.1 Test Case cb-spi-1: spi master loopback test 48](#_Toc456256983)

[3.10.2 Test Case cb-spi-2: spi master/slave mode basic tx/rx test 48](#_Toc456256984)

[3.10.3 Test Case cb-spi-3: spi master/slave tx/rx both transfer test 49](#_Toc456256985)

[3.10.4 Test Case cb-spi-4: baudrate setting test 50](#_Toc456256986)

[3.10.5 Test Case cb-spi-5: transfer data width test 50](#_Toc456256987)

[3.10.6 Test Case cb-spi-6: Motorola and TI mode test 51](#_Toc456256988)

[3.10.7 Test Case cb-spi-7: clock phase test 52](#_Toc456256989)

[3.10.8 Test Case cb-spi-8: dma mode test 52](#_Toc456256990)

[3.10.9 Test Case cb-spi-9: interrupt test 53](#_Toc456256991)

[3.10.10 Test Case cb-spi-10: robust transfer test 54](#_Toc456256992)

[3.10.11 Test Case cb-spi-11: access spi data flash test 54](#_Toc456256993)

[3.11 USB 55](#_Toc456256994)

[3.11.1 Test Case cb-usb-1: device mode: mass storage class test 55](#_Toc456256995)

[3.11.2 Test Case cb-usb-2: master mode: access mass storage class udisk 55](#_Toc456256996)

[3.11.3 Test Case cb-usb-3: master mode: access mass storage class udisk using fix data 56](#_Toc456256997)

[3.11.4 Test Case cb-usb-4: device mode: mass storage class test using dma mode 56](#_Toc456256998)

[3.11.5 Test Case cb-usb-5: master mode: access mass storage class udisk using dma mode 57](#_Toc456256999)

[3.11.6 Test Case cb-usb-6: otg feature test 57](#_Toc456257000)

[3.11.7 Test Case cb-usb-7: interrupt test 58](#_Toc456257001)

[3.12 NANDFLASH 58](#_Toc456257002)

[3.12.1 Test Case cb-nfc-1: nand probe test 58](#_Toc456257003)

[3.12.2 Test Case cb-nfc-2: nand block erase test 59](#_Toc456257004)

[3.12.3 Test Case cb-nfc-3: basic nand read/write test 59](#_Toc456257005)

[3.12.4 Test Case cb-nfc-4: nand fifo mode test 60](#_Toc456257006)

[3.12.5 Test Case cb-nfc-5: ecc support test 60](#_Toc456257007)

[3.12.6 Test Case cb-nfc-6: interrupt test 61](#_Toc456257008)

[3.13 MMC/SDIO 61](#_Toc456257009)

[3.13.1 Test Case cb-mmc-1: sd memory card probe test 61](#_Toc456257010)

[3.13.2 Test Case cb-mmc-2: sd memory card single block read/write test 62](#_Toc456257011)

[3.13.3 Test Case cb-mmc-3: sd memory card multi-block read/write test 62](#_Toc456257012)

[3.13.4 Test Case cb-mmc-4: sd memory card data bus width test 63](#_Toc456257013)

[3.13.5 Test Case cb-mmc-5: sd memory card transfer date rate test 63](#_Toc456257014)

[3.13.6 Test Case cb-mmc-6: sd memory card dma mode test 64](#_Toc456257015)

[3.13.7 Test Case cb-mmc-7: sd memory card write protect test 64](#_Toc456257016)

[3.13.8 Test Case cb-mmc-8: sd memory card interrupt test 65](#_Toc456257017)

[3.13.9 Test Case cb-mmc-9: sdio card register access test 65](#_Toc456257018)

[3.13.10 Test Case cb-mmc-10: sdio card memory access test 66](#_Toc456257019)

[3.13.11 Test Case cb-mmc-11: sdio card detect test 66](#_Toc456257020)

[3.14 IPC 67](#_Toc456257021)

[3.14.1 Test Case cb-ipc-1: 67](#_Toc456257022)

[3.15 ADC 67](#_Toc456257023)

[3.15.1 Test Case cb-adc-1: adc single mode single channel test 67](#_Toc456257024)

[3.15.2 Test Case cb-adc-2: adc single mode multi-channel test 68](#_Toc456257025)

[3.15.3 Test Case cb-adc-3: adc continuous mode single channel test 68](#_Toc456257026)

[3.15.4 Test Case cb-adc-4: adc continuous mode multi-channel test 69](#_Toc456257027)

[3.15.5 Test Case cb-adc-5: adc interrupt test 69](#_Toc456257028)

[3.15.6 Test Case cb-adc-6: dma mode test 69](#_Toc456257029)

[3.16 ECC/RSA 70](#_Toc456257030)

[3.16.1 Test Case cb-ecc-1: 70](#_Toc456257031)

[3.17 AES 70](#_Toc456257032)

[3.17.1 Test Case cb-aes-1: 70](#_Toc456257033)

[3.18 SHA 71](#_Toc456257034)

[3.18.1 Test Case cb-sha-1: 71](#_Toc456257035)

[3.19 OTP 71](#_Toc456257036)

[3.19.1 Test Case cb-otp-1: otp non-secure program test 71](#_Toc456257037)

[3.19.2 Test Case cb-otp-2: otp secure program test 72](#_Toc456257038)

[3.19.3 Test Case cb-otp-3: key disable test 72](#_Toc456257039)

[3.19.4 Test Case cb-otp-4: otp interrupt handler test 73](#_Toc456257040)

[3.20 CRC 73](#_Toc456257041)

[3.20.1 Test Case cb-crc-1: 73](#_Toc456257042)

[3.21 RNG 74](#_Toc456257043)

[3.21.1 Test Case cb-rng-1: 74](#_Toc456257044)

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Volans Guo | 2016-7-13 | Initial | 0.1 |
|  |  |  |  |

## Introduction

This document is the CDL Test Plan of Board Supporting Packages(BSP) for the Brite FPGA Platform and Target IC built by the Columbus Project. The document mainly include the test case design.

## Testing System Overview

## Testing Case Design

### Timer

#### Test Case cb-timer-1: timer start/stop test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm timer can start and stop  count mode: increase  when start, timer counter increase; when stop, timer counter keep fixed | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "timer" from device list |  |
| 2 | input "timer\_start\_stop 1" | timer\_start\_stop: test pass |
| 3 | [repeat]input " timer\_start\_stop id"  to test other timer (1-8) | timer\_start\_stop: test pass |

#### Test Case cb-timer-2: timer reload test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| test when trigger reload, count restart from the load value | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "timer" from device list |  |
| 2 | input "timer\_reload 1" | timer\_reload: test pass |
| 3 | [repeat] input " timer\_reload id" to test other timer (id: 1-8) | timer\_reload: test pass |

#### Test Case cb-timer-3: timer count mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| timer counter increase/decrease according count mode  check with ‘free run mode’/’user defined mode’  check with ‘one shot mode’/’auto reload mode’ | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "timer" from device list |  |
| 2 | input " timer\_count\_mode 1" | timer\_count\_mode: test pass |
| 3 | [repeat]input "timer\_count\_mode id"  to test other timer (1-8) | timer\_count\_mode: test pass |

#### Test Case cb-timer-4: timer overflow test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| When timer overflow, diffrent mode with diffrent action  Free run mode : counter always wrap from ‘0’ or ‘0xffffffff’ and continue run  User define mode :  one shot mode, when counter overflow,timer stop  auto reload mode, when counter overflow, reload value from load val and continue run | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "timer" from device list |  |
| 2 | input "timer\_overflow 1" | timer\_overflow: test pass |
| 3 | [repeat]input " timer\_overflow id"  to test other timer (1-8) | timer\_overflow: test pass |

#### Test Case cb-timer-5: timer soft reset test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Config timer run using default configuration(UP)  After do sw reset, timer should stop and current counter should be ‘0’ | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "timer" from device list |  |
| 2 | input "timer\_reset 1" | timer\_ reset: test pass |
| 3 | [repeat]input "timer\_ reset id"  to test other timer (1-8) | timer\_ reset: test pass |

#### Test Case cb-timer-6: timer frequency division test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| test diffrent division works and correct  division : 0-8 (1—1/256) | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "timer" from device list |  |
| 2 | no div  input "timer\_div 1 0" | Print ‘1 second elapsed’ about 1 second ;  You can view waveform using oscilloscope ; period should be 1 ms |
| 3 | 1/2 div  input "timer\_div 1 1" | Print ‘1 second elapsed’ about 1 second ;  You can view waveform using oscilloscope ; period should be 2 ms |
| 4 | 1/4 div  input "timer\_div 1 2" | Print ‘1 second elapsed’ about 1 second ;  You can view waveform using oscilloscope ; period should be 4 ms |
| 5 | 1/8 div  input "timer\_div 1 3" | Print ‘1 second elapsed’ about 1 second ;  You can view waveform using oscilloscope ; period should be 8 ms |
| 6 | 1/16 div  input "timer\_div 1 4" | Print ‘1 second elapsed’ about 1 second ;  You can view waveform using oscilloscope ; period should be 16 ms |
| 7 | 1/32 div  input "timer\_div 1 5" | Print ‘1 second elapsed’ about 1 second ;  You can view waveform using oscilloscope ; period should be 32 ms |
| 8 | 1/64 div  input "timer\_div 1 6" | Print ‘1 second elapsed’ about 1 second ;  You can view waveform using oscilloscope ; period should be 64 ms |
| 9 | 1/128 div  input "timer\_div 1 7" | Print ‘1 second elapsed’ about 1 second ;  You can view waveform using oscilloscope ; period should be 128 ms |
| 10 | 1/256 div  input "timer\_div 1 8" | Print ‘1 second elapsed’ about 1 second ;  You can view waveform using oscilloscope ; period should be 256 ms |
| 11 | [repeat]  input "timer\_div id div" for other timers  div: 0-8 |  |

#### Test Case cb-timer-7: timer interrupt test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm interrupt to gic can be handled correctly | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "timer" from device list |  |
| 2 | input "timer\_int 1" | timer\_int: test pass |
| 3 | [repeat] other timer test  input "timer\_int id"  id: 1-8 | timer\_int: test pass |

#### Test Case cb-timer-8: timer capture mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| test capture mode : set diffrent capture mode(first/second event) and capture event ;  then trigger the event | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "timer" from device list |  |
| 2 | input "timer\_capture id mode event"  id: 1-8  mode: 0,1 (first/second valid event)  event: 1-3 (hight to low ; low to high ; both) |  |
| 3 | Trigger diffrent event at timer capture pin | check the console output, compare with the capture trigger action to judge pass or fail. Console output like this :  timer 1: capture second for event high to low  timer 1 started  now trigger the event and check console output  capture: 1732fbb6 173a17fe  current timer count: 173b14b0 |

#### Test Case cb-timer-9: timer compare mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| test compare mode | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "timer" from device list |  |
| 2 | input "timer\_compare id compare\_second"  id: 1-8  compare\_val : 1-85 second | check the console output and compare\_val to judge test pass or fail |

#### Test Case cb-timer-10: timer pwm mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| test pwm mode setting | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "timer" from device list |  |
| 2 | Test using diffrent parameters  input "timer\_pwm id mode out\_mode pwm\_pol"  id: 1-8  mode: 1-3 (overflow ;match & overflow ; match)  event: 0,1 (pulse ; toggle)  pwm\_pol : 0,1 (Positive, Negative) only effect for event: 0(pulse)  testcase setting : timer clk : 50000000  overflow counter : 30000  match counter : 10000 | check oscilloscope waveform and parameters to judge pass or fail |

### Watch Dog Timer

#### Test Case cb-wdt-1: wdt start/stop test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm wdt can start and stop  when start, wdt counter decrease; when stop, wdt counter keep fixed | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "watch dog timer" from device list |  |
| 2 | input "wdt\_start\_stop" | console out: wdt\_start\_stop: test pass |

#### Test Case cb-wdt-2: wdt load value test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm when start, wdt counter start from the configured load value  note : don’t set first parameters too small ; because wdt will reset in short time if it’s too small | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "watch dog timer" from device list |  |
| 2 | input "wdt\_reload start 0x10000000" | console output: wdt\_reload: test pass |
| 3 | input "wdt\_reload start 0x20000000" | console output: wdt\_reload: test pass |

#### Test Case cb-wdt-3: wdt restart test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm we can feed wdt successfully  (when do this, watch dog timer reload from load value)  note : don’t set first parameters too small ; because wdt will reset in short time if it’s too small | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "watch dog timer" from device list |  |
| 2 | input " wdt\_reload restart 0x10000000" | console output: wdt\_reload: test pass |
| 3 | input " wdt\_reload restart 0x20000000" | console output: wdt\_reload: test pass |

#### Test Case cb-wdt-4: wdt generate reset test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm we wdt counter reach, reset pulse generated  also check the reset pulse length meet the configuration  reset pulse time calculate :  (reset\_pulse\_lengh\_config/50000000) second  Eg : 0x10000🡺65536 65536/50000000 = 0.0013 second | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "watch dog timer" from device list |  |
| 2 | input "wdt\_reset 0x10000 3"  [using oscilloscope  to check the rst pin's](http://192.168.2.4/testlink/lib/testcases/tcEdit.php?doAction=editStep&testcase_id=243&tcversion_id=244&goback_url=http%3A%2F%2F192.168.2.4%2Ftestlink%2Flib%2Ftestcases%2FarchiveData.php%3Ftcase_id%3D243%26show_mode%3Dshow&step_id=251) waveform | Cdl test restart after 3 second  can view the rst pin waveform:  match the length '0x10000' cycle  pclk :50000000  reset pulse  about : 1.3 ms |
| 3 | input "wdt\_reset 0x100 3"  [using oscilloscope  to check the rst pin's](http://192.168.2.4/testlink/lib/testcases/tcEdit.php?doAction=editStep&testcase_id=243&tcversion_id=244&goback_url=http%3A%2F%2F192.168.2.4%2Ftestlink%2Flib%2Ftestcases%2FarchiveData.php%3Ftcase_id%3D243%26show_mode%3Dshow&step_id=251) waveform | [Cdl test restart after 3 second](http://192.168.2.4/testlink/lib/testcases/tcEdit.php?doAction=editStep&testcase_id=139&tcversion_id=140&goback_url=http%3A%2F%2F192.168.2.4%2Ftestlink%2Flib%2Ftestcases%2FarchiveData.php%3Ftcase_id%3D139%26show_mode%3Dshow&step_id=457)  [can view the rst pin waveform](http://192.168.2.4/testlink/lib/testcases/tcEdit.php?doAction=editStep&testcase_id=139&tcversion_id=140&goback_url=http%3A%2F%2F192.168.2.4%2Ftestlink%2Flib%2Ftestcases%2FarchiveData.php%3Ftcase_id%3D139%26show_mode%3Dshow&step_id=457): match the length '0x100' cycle  reset pulse about : 5 us |

#### Test Case cb-wdt-5: wdt frequency division test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| test diffrent division works and correct | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "watch dog timer" from device list |  |
| 2 | input "wdt\_div 0 0"  1/2 freq | Cdl test restart after several second ; progress displayed  You can see the counter decrease  also can [using oscilloscope  to check the rst pin's](http://192.168.2.4/testlink/lib/testcases/tcEdit.php?doAction=editStep&testcase_id=243&tcversion_id=244&goback_url=http%3A%2F%2F192.168.2.4%2Ftestlink%2Flib%2Ftestcases%2FarchiveData.php%3Ftcase_id%3D243%26show_mode%3Dshow&step_id=251) waveform |
| 3 | input "wdt\_tc4 0 1"  1/4 freq | Cdl test restart after several second ; progress displayed  You can see the counter decrease  also can [using oscilloscope  to check the rst pin's](http://192.168.2.4/testlink/lib/testcases/tcEdit.php?doAction=editStep&testcase_id=243&tcversion_id=244&goback_url=http%3A%2F%2F192.168.2.4%2Ftestlink%2Flib%2Ftestcases%2FarchiveData.php%3Ftcase_id%3D243%26show_mode%3Dshow&step_id=251) waveform |
| 4 | Repeat other parameters group  Div1 : 0-13  Div2 : 0-14 | Cdl test restart after several second ; progress displayed  You can see the counter decrease  also can [using oscilloscope  to check the rst pin's](http://192.168.2.4/testlink/lib/testcases/tcEdit.php?doAction=editStep&testcase_id=243&tcversion_id=244&goback_url=http%3A%2F%2F192.168.2.4%2Ftestlink%2Flib%2Ftestcases%2FarchiveData.php%3Ftcase_id%3D243%26show_mode%3Dshow&step_id=251) waveform |

#### Test Case cb-wdt-6: wdt interrupt timeout test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm wdt counter reach, interrupt generated (in 1 second)  also check the interrupt timeout feature, after timeout, reset generated  wdt using div : 0x96(1/(64\*1024)) | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "watch dog timer" from device list |  |
| 2 | input " wdt\_interrupt\_timeout 0" | console output: interrupt generate at xx ms :  near 1000ms   about 83 second,reset will restart cdl |
| 3 | input " wdt\_interrupt\_timeout 1" | console output: interrupt generate at xx ms :  near 1000ms   about 41 second,reset will restart cdl |
| 4 | input " wdt\_interrupt\_timeout 2" | console output: interrupt generate at xx ms :  near 1000ms   about 20 second,reset will restart cdl |
| 5 | input " wdt\_interrupt\_timeout 3" | console output: interrupt generate at xx ms :  near 1000ms   about 10 second,reset will restart cdl |
| 6 | input " wdt\_interrupt\_timeout 4" | console output: interrupt generate at xx ms :  near 1000ms   about 5 second,reset will restart cdl |
| 7 | input " wdt\_interrupt\_timeout 5" | console output: interrupt generate at xx ms :  near 1000ms   about 2 second,reset will restart cdl |
| 8 | input " wdt\_interrupt\_timeout 6" | console output: interrupt generate at xx ms :  near 1000ms   about 1 second,reset will restart cdl |

#### Test Case cb-wdt-7: wdt interrupt handle test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm interrupt to gic can be handled correctly | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "watch dog timer" from device list |  |
| 2 | input "wdt\_int" | wdt\_ int: test pass |

#### Test Case cb-wdt-8: wdt on-the-fly read/write test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Confirm when watch dog running, we can write/read watch dog register successfully  Will display current counter ; and change reload value in the test | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "watch dog timer" from device list |  |
| 2 | input "wdt\_fly" | Counter decrease ;  After reload counter decrease from reload value ; |

### RTC/BBU

#### Test Case cb-rtc-1: rtc timer set/get test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Setting one rtc time value ; then read back to verify  Command format : rtc\_set\_time mode year month day hour min sec  Mode : 0 12 hour mode ; 1 24 hour mode  Year/month/day/hour/min/sec : 10-base number ; hour must be according 24 hour mode  Year : 1900-2099 | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "rtc" from device list |  |
| 2 | Input “rtc\_set\_time 0 2016 7 12 19 21 23” |  |
| 3 | Input “rtc\_get\_time 0” | Display time should match the setting value(not same,because timer increaseing . |
| 4 | Repeat: step 2-3 to set other value to check  Input “rtc\_set\_time mode year month day hour min sec “  Year: 1900-2099  Month:1-12  Day:1-31  Hour: 0-23  Minute: 0-59  Second: 0-59 |  |

#### Test Case cb-rtc-2: rtc timer 12/24 hour mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Setting one rtc time value ; then read back to verify  Check 12/24 hour mode set/get correctly | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "rtc" from device list |  |
| 2 | Set time using 12 hour mode : AM time  Input “rtc\_set\_time 0 2016 7 12 7 21 23”  Display using 12 hour mode  Input “rtc\_get\_time 0“  Display using 24 hour mode  Input “rtc\_get\_time 1“ | Display time should match the setting value(not same,because timer increaseing |
| 3 | Set time using 12 hour mode : PM time  Input “rtc\_set\_time 0 2016 7 12 17 21 23”  Display using 12 hour mode  Input “rtc\_get\_time 0“  Display using 24 hour mode  Input “rtc\_get\_time 1“ | Display time should match the setting value(not same,because timer increaseing |
| 4 | Set time using 24 hour mode : AM time  Input “rtc\_set\_time 1 2016 7 12 7 21 23”  Display using 12 hour mode  Input “rtc\_get\_time 0“  Display using 24 hour mode  Input “rtc\_get\_time 1“ | Display time should match the setting value(not same,because timer increaseing |
| 5 | Set time using 24 hour mode : PM time  Input “rtc\_set\_time 1 2016 7 12 17 21 23”  Display using 12 hour mode  Input “rtc\_get\_time 0“  Display using 24 hour mode  Input “rtc\_get\_time 1“ | Display time should match the setting value(not same,because timer increaseing |

#### Test Case cb-rtc-3: rtc timer cover test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Setting one rtc time value ; then read back to verify  Cover conner case of year/month/day/hour/minute/second | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "rtc" from device list |  |
| 2 | Cover year  Input “rtc\_set\_time 0 1970 7 12 19 21 23” | Display time should match the setting value(not same,because timer increaseing . |
| 3 | Input “rtc\_set\_time 0 2099 7 12 19 21 23” | Display time should match the setting value(not same,because timer increaseing . |
| 4 | Cover month  Input “rtc\_set\_time 0 2016 1 12 19 21 23” | Display time should match the setting value(not same,because timer increaseing . |
| 5 | Input “rtc\_set\_time 0 2016 12 12 19 21 23” | Display time should match the setting value(not same,because timer increaseing . |
| 6 | Conver day  rtc\_set\_time 0 2016 7 1 19 21 23 | Display time should match the setting value(not same,because timer increaseing . |
| 7 | Input “rtc\_set\_time 0 2016 7 31 19 21 23” | Display time should match the setting value(not same,because timer increaseing . |
| 8 | Conver hour  Input “rtc\_set\_time 0 2016 7 12 0 21 23” | Display time should match the setting value(not same,because timer increaseing . |
| 9 | Input “rtc\_set\_time 0 2016 7 12 23 21 23” | Display time should match the setting value(not same,because timer increaseing . |
| 10 | Conver minute  Input “rtc\_set\_time 0 2016 7 12 19 0 23” | Display time should match the setting value(not same,because timer increaseing . |
| 11 | Input “rtc\_set\_time 0 2016 7 12 19 59 23” | Display time should match the setting value(not same,because timer increaseing . |
| 12 | Conver second  Input “rtc\_set\_time 0 2016 7 12 19 21 0” | Display time should match the setting value(not same,because timer increaseing . |
| 13 | Input “rtc\_set\_time 0 2016 7 12 19 21 59” | Display time should match the setting value(not same,because timer increaseing . |

#### Test Case cb-rtc-4: rtc timer increase test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Setting one rtc time value ; then wait one segment increase,then read back to verify  Cover conner case of year/month/day/hour/minute/second | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "rtc" from device list |  |
| 2 | Cover year  Input “rtc\_set\_time 0 2016 12 31 23 59 57” | Display time should match the setting value(not same,because timer increaseing . |
| 3 | Wait 3 second  Rtc\_get\_time | Time should be :  Year 2017 |
| 4 | You can repeat step 2-3 to test more conner case |  |

#### Test Case cb-rtc-5: rtc timer alarm test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Setting one rtc time value ; then wait one segment increase,then read back to verify  Cover conner case of year/month/day/hour/minute/second | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "rtc" from device list |  |
| 2 | Second alarm  Input “rtc\_alarm 0 2016 12 31 23 59 10 1”  Wait 3 second | Should output : current alarm1 time: xxxx |
| 3 | minute alarm  Input “rtc\_alarm 0 2016 12 31 23 50 58 2”  Wait 3 second | Should output : current alarm1 time: xxxx |
| 4 | hour alarm  Input “rtc\_alarm 0 2016 12 31 20 59 58 3”  Wait 3 second | Should output : current alarm1 time: xxxx |
| 5 | day alarm  Input “rtc\_alarm 0 2016 12 20 23 59 58 5”  Wait 3 second | Should output : current alarm1 time: xxxx |
| 6 | month alarm  Input “rtc\_alarm 0 2016 10 31 23 59 58 6”  Wait 3 second | Should output : current alarm1 time: xxxx |
| 7 | year alarm  Input “rtc\_alarm 0 2016 12 31 23 59 58 7”  Wait 3 second | Should output : current alarm1 time: xxxx |

#### Test Case cb-rtc-6: rtc interrupt test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm interrupt to gic can be handled correctly | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "rtc" from device list |  |
| 2 | Input “rtc\_int 0 2016 12 31 23 59 10 1” | rtc\_int: test pass |

### UART

#### Test Case cb-uart-1: loopback transmit/receive test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| set to loopback mode  test for all uart port(uart0/uart1/uart2/uart3)  confirm received data same with the transmited | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "uart0/1/2/3" from device list |  |
| 2 | input "uart\_loopback 1024" | console out: "uart\_loopback: test pass" |
| 3 | input "uart\_loopback 2043" | console out: uart\_loopback: test pass |

#### Test Case cb-uart-2: loopback modem control test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| uart0/1/2/3: test rts/cts control signal  uart3: test all other modem control signal | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "uart0/1/2/3" from device list |  |
| 2 | input "uart\_modem\_loopback" | console output: uart\_modem\_loopback: test pass |

#### Test Case cb-uart-3: basic send test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| send one string out  for check and debug basic functions | | |
| Preconditions | | |
| connect serial port to PC  PC open the serial using general serial tools | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "uart0/1/2/3" from device list |  |
| 2 | send using polling mode,send one time  send string "abcdefghijklmnopqrstuvwxyz0123456789"  input "uart\_send 0" | pc received the string |
| 3 | send using dma mode,send one time  send string "abcdefghijklmnopqrstuvwxyz0123456789"  input "uart\_send 1" | pc received the string |

#### Test Case cb-uart-4: basic receive test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| receive and print out  for check and debug basic functions | | |
| Preconditions | | |
| connect serial port to PC  PC open the serial using general serial tools | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "uart0/1/2/3" from device list |  |
| 2 | test uart receive using polling mode  input "uart\_receive 0" | when pc tools send string;  board side received and display |
| 3 | test uart receive using dma mode  input "uart\_receive 1" | when pc tools send string;  board side received and display |

#### Test Case cb-uart-5: basic rs232 transmit / receive test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| set 115200,n,8,1 (baudrate,parity,data bits, stop bits) no flowctrl  device: send random data; then readback to verify  pc: serial echo server; just received and send back data | | |
| Preconditions | | |
| connect uart port to PC serial port;pc using sser.exe tools | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "uart0/1/2/3" from device list |  |
| 2 | pc run: sser port 115200 n 8 1 0"  input "uart\_transfer 100" | uart\_transfer: test pass |
| 3 | input "uart\_transfer 1000" | uart\_transfer: test pass |

#### Test Case cb-uart-6: baudrate setting test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| x,n,8,1 : set diffrent baudrate ,confirm diffrent baudrate works fine  300< baudrate < 460800  note: PC may can't reach 3.6884Mbps,  even can't reach 1M, high baudrate will test using cb-34  device: send random data; then readback to verify  pc: serial echo server; just received and send back data | | |
| Preconditions | | |
| connect uart port to PC serial port;pc using sser.exe tools  pc will set same baudrate when test | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "uart0/1/2/3" from device list |  |
| 2 | pc run: sser port 1200 n 8 1 0"  input "uart\_config 1200 n 8 1 0" | uart\_config: test pass |
| 3 | pc run: sser port 9600 n 8 1 0"  input "uart\_config 9600 n 8 1 0" | uart\_config: test pass |
| 4 | pc run: sser port 38400 n 8 1 0"  input "uart\_config 38400 n 8 1 0" | uart\_config: test pass |
| 5 | pc run: sser port 460800 n 8 1 0"  input "uart\_config 460800 n 8 1 0" | uart\_config: test pass |
| 6 | pc run: sser port 300 n 8 1 0"  input "uart\_config 300 n 8 1 0" | uart\_config: test pass |

#### Test Case cb-uart-7: stop bits setting test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| set 115200,n,8,x (baudrate,parity,data bits, stop bits) x=1,2  set 115200,n,5,2 (baudrate,parity,data bits, stop bits)  confirm diffrent stop bits transfer OK  device: send random data; then readback to verify  pc: serial echo server; just received and send back data | | |
| Preconditions | | |
| connect uart port to PC serial port;pc using sser.exe tools  pc will set same configuration when test | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "uart0/1/2/3" from device list |  |
| 2 | pc run: sser port 115200 n 8 1 0"  input "uart\_config 115200 n 8 1 0" | uart\_config: test pass |
| 3 | pc run: sser port 115200 n 8 2 0"  input "uart\_config 115200 n 8 2 0" | uart\_config: test pass |
| 4 | pc run: sser port 115200 n 5 2 0"  input "uart\_config 115200 n 5 2" | uart\_config: test pass |

#### Test Case cb-uart-8: parity bits setting test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| set 115200,x,8,1 (baudrate,parity,data bits, stop bits) x=n,e,o  confirm diffrent parity mode works  device: send random data; then readback to verify  pc: serial echo server; just received and send back data | | |
| Preconditions | | |
| connect uart port to PC serial port;pc using sser.exe tools  pc will set same configuration when test | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "uart0/1/2/3" from device list |  |
| 2 | pc run: sser port 115200 n 8 1 0"  input "uart\_config 115200 n 8 1 0" | uart\_config: test pass |
| 3 | pc run: sser port 115200 e 8 1 0"  input "uart\_config 115200 e 8 1 0" | uart\_config: test pass |
| 4 | pc run: sser port 115200 o 8 1 0"  input "uart\_config 115200 o 8 1 0" | uart\_config: test pass |

#### Test Case cb-uart-9: data bits setting test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| set 115200,n,x,1 (baudrate,parity,data bits, stop bits) x=5,6,7,8  confirm diffrent data bits transfer OK  device: send random data; then readback to verify  pc: serial echo server; just received and send back data | | |
| Preconditions | | |
| connect uart port to PC serial port;pc using sser.exe tools  pc will set same configuration when test | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "uart0/1/2/3" from device list |  |
| 2 | pc run: sser port 115200 n 8 1 0"  input "uart\_config 115200 n 8 1 0" | uart\_config: test pass |
| 3 | pc run: sser port 115200 n 7 1 0"  input "uart\_config 115200 n 7 1 0" | uart\_config: test pass |
| 4 | pc run: sser port 115200 n 6 1 0"  input "uart\_config 115200 n 6 1 0" | uart\_config: test pass |
| 5 | pc run: sser port 115200 n 5 1 0"  input "uart\_config 115200 n 5 1 0" | uart\_config: test pass |

#### Test Case cb-uart-10: hardware flowctrol test

|  |  |  |  |
| --- | --- | --- | --- |
| Summary | | | |
| set 115200,n,8,1 (baudrate,parity,data bits, stop bits) hardware flowctrl  device: send random data; then readback to verify  pc: serial echo server; just received and send back data; with hardware flowctrl enabled  then set 115200,n,8,1 (baudrate,parity,data bits, stop bits)  no flowctrl  this time should lock and test can't proceed | | | |
| Preconditions | | |
| connect uart port to PC serial port;pc using sser.exe tools | | |
| Step | | | |
| # | Step actions | Expected Results | |
| 1 | run cdl test suit and select "uart0/1/2/3" from device list |  | |
| 2 | pc run: sser port 115200 n 8 1 1"  input "uart\_flowctrl 10" | first 10 loop will success with 0 fail  next 10 will lock or fail | |

#### Test Case cb-uart-11: dma mode rs232 transmit / receive test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| set 115200,n,8,1 (baudrate,parity,data bits, stop bits)  using dma mode for transmit/receive,confirm it works fine  device: send random data; then readback to verify  pc: serial echo server; just received and send back data | | |
| Preconditions | | |
| connect uart port to PC serial port;pc using sser.exe tools | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "uart0/1/2/3" from device list |  |
| 2 | pc run: sser port 115200 n 8 1 0"  input "uart\_dma 100" | uart\_dma: test pass |
| 3 | input "uart\_dma 10000" | uart\_dma: test pass |

#### Test Case cb-uart-12: high baudrate setting test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| x,n,8,1 : set diffrent baudrate ,confirm diffrent baudrate works fine  460800 < baudrate < 3.6884Mbps  device: send random data; then readback to verify  server side: serial echo server; just received and send back data | | |
| Preconditions | | |
| connect two columbus chip board(FPGA/EVB) UART port | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "uart0/1/2/3" from device list |  |
| 2 | server side: input "uart\_echo 921600 n 8 1 0"  slave side:input "uart\_config 921600 n 8 1 0" | uart\_config: test pass |
| 3 | server side: input "uart\_echo 1843200 n 8 1 0"  slave side:input "uart\_config 1843200 n 8 1 0" | uart\_config: test pass |
| 4 | server side: input "uart\_echo 3110400 n 8 1 0"  slave side:input "uart\_config 3110400 n 8 1 0" | uart\_config: test pass |
| 5 | server side: input "uart\_echo 3686400 n 8 1 0"  slave side:input "uart\_config 3686400 n 8 1 0" | uart\_config: test pass |

#### Test Case cb-uart-13: modem control test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm modem control correctly  master side: set modem control output  slave side: check coresponding signal correct | | |
| Preconditions | | |
| connect two columbus board(FPGA/EVB) | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "uart0/1/2/3" from device list |  |
| 2 | master side: input "uart\_modem\_master all 0"  slave side: input "uart\_modem" |  |
| 3 | master side: input "uart\_modem\_master dtr 1" | slave side console output:  dsr change:0 to 1 |
| 4 | master side: input "uart\_modem\_master rts 1" | slave side console output:  cts change:0 to 1 |
| 5 | master side: input "uart\_modem\_master out1 1" | slave side console output:  ri change:0 to 1 |
| 6 | master side: input "uart\_modem\_master out2 1" | slave side console output:  dcd change:0 to 1 |
| 7 | master side: input "uart\_modem\_master dtr 0" | slave side console output:  dsr change:1 to 0 |
| 8 | master side: input "uart\_modem\_master rts 0" | slave side console output:  cts change:1 to 0 |
| 9 | master side: input "uart\_modem\_master out1 0" | slave side console output:  ri change:1 to 0 |
| 10 | master side: input "uart\_modem\_master out2 0" | slave side console output:  dcd change:1 to 0 |

#### Test Case cb-uart-14: line break test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| test line break generate and detect  master side: set modem control output  slave side: check break condition occur | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "uart0/1/2/3" from device list |  |
| 2 | slave side: input "uart\_line" |  |
| 3 | master side: input "uart\_line\_master" | slave side: detect break sequence |

#### Test Case cb-uart-15: interrupt test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| trigger some interrupt state, and check interrupt handle | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "uart0/1/2/3" from device list |  |
| 2 | input "uart\_int" | uart\_int\_handler,status=%x  uart\_int: test pass |

### GIC/INTERRUPT

#### Test Case cb-gic-1: cpu internal interrupt(PPI) test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| use generic timer (a9 global timer) to trigger the interrupt  to verify gic works fine | | |
| Preconditions | | |
| ZYNQ platform only | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "gic" from device list |  |
| 2 | input "gic\_int" | gic\_int: test pass |

#### Test Case cb-gic-2: external interrupt(SPI) generate test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm external peripheral interrupt can be processed correctly(handler is called) | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | refer the test case of peripheral  refer: cb-dma-7: dma interrupt test result |  |

### DMAC

#### Test Case cb-dma-1: mem to mem transfer

|  |  |  |
| --- | --- | --- |
| Summary | | |
| sram<-->ddr ; ddr<-->ddr ;sram<-->sram  single block transfer mode  [cmd format: dma\_m2m ch src\_addr dst\_addr length] | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "dma0/1/2" from device list |  |
| 2 | test  sram to ddr transfer  input "dma\_m2m 0 0x10000000 0x42000000 2048" | console output: dma\_m2m: test pass |
| 3 | test ddr to ddr transfer  input "dma\_m2m 0 0x412000000 0x42100000 2048" | console output: dma\_m2m: test pass |
| 4 | test ddr to sram transfer  input "dma\_m2m 0 0x42000000 0x10000000 2048" | console output: dma\_m2m: test pass |
| 5 | test sram to sram transfer  input "dma\_m2m 0 0x10000000 0x10010000 2048" | console output: dma\_m2m: test pass |
| 6 | [option]test using other memory location and size  input "dma\_m2m ch src\_addr dst\_addr length"  notice:  DDR: 0x42000000 -- 0x44000000               SRAM: 0x10000000-- 0x10020000 | console output: dma\_m2m: test pass |

#### Test Case cb-dma-2: mem to mem with diffrent configuration

|  |  |  |
| --- | --- | --- |
| Summary | | |
| using ddr<--->ddr : src:0x42000000 dst:0x42100000  bus width, burst size, block transfer length; addr update mode  check each param configuration  [cmd format: dma\_cfg ch trans\_len src\_dst bus\_width burst\_size addr\_update] | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "dma0/1/2" from device list |  |
| 2 | input "dma\_cfg 0 2048 0 0 0 0"  ch:0  transfer len: 2048; actual 2048 bytes  src transfer width: byte  src burst size: 1  src addr : increase | console output: dma\_cfg: test pass |
| 3 | input "dma\_cfg 0 2048 0 1 1 0"  ch:0  transfer len: 2048; actual 2048\*2 bytes  src transfer width: 2 bytes  src burst size: 4  src addr : increase | console output: dma\_cfg: test pass |
| 4 | input "dma\_cfg 0 2048 0 2 2 0"  ch:0  transfer len: 2048; actual 2048\*4 bytes  src transfer width: 4 bytes  src burst size: 8  src addr : increase | console output: dma\_cfg: test pass |
| 5 | input "dma\_cfg 0 2048 0 2 6 0"  ch:0  transfer len: 2048; actual 2048\*4 bytes  src transfer width: 4 bytes  src burst size: 128  src addr : increase | console output: dma\_cfg: test pass |
| 6 | input "dma\_cfg 0 2048 0 2 7 0"  ch:0  transfer len: 2048; actual 2048\*4 bytes  src transfer width: 4 bytes  src burst size: 256  src addr : increase | console output: dma\_cfg: test pass |
| 7 | input "dma\_cfg 0 2048 0 2 1 1"  ch:0  transfer len: 2048; actual 2048\*4 bytes  src transfer width: 4 bytes  src burst size: 4  src addr : decrease | console output: dma\_cfg: test pass |
| 8 | input "dma\_cfg 0 2048 0 2 1 2"  ch:0  transfer len: 2048; actual 2048\*4 bytes  src transfer width: 4 bytes  src burst size: 4  src addr : fixed | console output: dma\_cfg: test pass |
| 9 | configure dst parameters  input "dma\_cfg 0 2048 1 0 0 0"  input "dma\_cfg 0 2048 1 1 1 0"  input "dma\_cfg 0 2048 1 2 2 0"  input "dma\_cfg 0 2048 1 2 6 0"  input "dma\_cfg 0 2048 1 2 7 0"  input "dma\_cfg 0 2048 1 2 1 1"  input "dma\_cfg 0 2048 1 2 1 2" | console output: dma\_cfg: test pass |

#### Test Case cb-dma-3: dma channel test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm each channel(total 8) works  using mem to mem single block transfer | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "dma0/1/2" from device list |  |
| 2 | input "dma\_m2m 0 0x42000000 0x42010000 2048" | console output: dma\_m2m: test pass |
| 3 | input "dma\_m2m 1 0x42000000 0x42010000 2048" | console output: dma\_m2m: test pass |
| 4 | input "dma\_m2m 2 0x42000000 0x42010000 2048" | console output: dma\_m2m: test pass |
| 5 | input "dma\_m2m 3 0x42000000 0x42010000 2048" | console output: dma\_m2m: test pass |
| 6 | input "dma\_m2m 4 0x42000000 0x42010000 2048" | console output: dma\_m2m: test pass |
| 7 | input "dma\_m2m 5 0x42000000 0x42010000 2048" | console output: dma\_m2m: test pass |
| 8 | input "dma\_m2m 6 0x42000000 0x42010000 2048" | console output: dma\_m2m: test pass |
| 9 | input "dma\_m2m 7 0x42000000 0x42010000 2048" | console output: dma\_m2m: test pass |

#### Test Case cb-dma-4: multi-block transfer test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| test multi-block transfer(using LLI list), mem to mem mode | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "dma0/1/2" from device list |  |
| 2 | input "dma\_mb 0 0x42000000 0x42010000 2048" | console output: dma\_mb: test pass |
| 3 | input "dma\_mb 0 0x42000000 0x42010000 10240" | console output: dma\_mb: test pass |
| 4 | input "dma\_mb 5 0x42000000 0x42010000 102400" | console output: dma\_mb: test pass |
| 5 | input "dma\_mb 5 0x42000000 0x42010000 10240" | console output: dma\_mb: test pass |

#### Test Case cb-dma-5: memory to peripheral test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| memory to peripheral test  this may contained in coreponding peripheral test case  eg: uart,spi,qspi,i2c | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | please refer the result of peripheral  qspi : cb-qspi-4 : qspi dma mode test for dma0  i2c: cb-i2c-4 : i2c1 dma test for dmac1  uart: cb-uart-11 : uart0 dma test for dmac2 |  |

#### Test Case cb-dma-6: peripheral to memory test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| peripheral to memory test  this may contained in coreponding peripheral test case  eg: uart,spi,qspi,i2c | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | please refer the result of peripheral  i2c: cb-i2c-4 i2c0 dma test for dmac0  i2c1 dma test for dmac1  uart: cb-uart-11 : uart0 dma test for dmac2 |  |

#### Test Case cb-dma-7: dma interrupt test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm dma interrupt to ARM generated | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "dma0/1/2" from device list |  |
| 2 | input "dma\_int 0 " | console output:dma\_int\_handler,status=xx   ch 0 tfr int triggered  console output: dma\_int: test pass |
| 3 | input "dma\_int 5" | console output:dma\_int\_handler,status=xx   ch 5 tfr int triggered  console output: dma\_int: test pass |
| 4 | You can input "dma\_int ch " to test other channel |  |

#### Test Case cb-dma-8: multi-channel transfer test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm at least two channel's transfer at the same time(m2m) | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "dma" from device list |  |
| 2 | input "dma\_mc 0 1" | console output: dma\_mc: test pass |
| 3 | input "dma\_mc 3 5" | console output: dma\_mc: test pass |

### GPIO

#### Test Case cb-gpio-1: gpio output test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm each pins can configure as output,and output high/low level | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "gpio" from device list |  |
| 2 | [input "gpio\_output 4 1"](http://192.168.2.4/testlink/lib/testcases/tcEdit.php?doAction=editStep&testcase_id=243&tcversion_id=244&goback_url=http%3A%2F%2F192.168.2.4%2Ftestlink%2Flib%2Ftestcases%2FarchiveData.php%3Ftcase_id%3D243%26show_mode%3Dshow&step_id=251)  [​using oscilloscope or multimeter to check the pin's](http://192.168.2.4/testlink/lib/testcases/tcEdit.php?doAction=editStep&testcase_id=243&tcversion_id=244&goback_url=http%3A%2F%2F192.168.2.4%2Ftestlink%2Flib%2Ftestcases%2FarchiveData.php%3Ftcase_id%3D243%26show_mode%3Dshow&step_id=251) [electrical level](http://www.baidu.com/link?url=ne1UuOQ_LBwhCJqZJlnqUW92wN7dwbOUWsirrER9mk2WA7spNJbIFq8p0sr7XkiYKZQGCi1szCPk9nJ7asKQDPSKu0vOehrDl6oEVEF1d1CRbIDQQQdMKzirVYeXNFai) | gpio 4 pins: high |
| 3 | [input "gpio\_output 4 0"](http://192.168.2.4/testlink/lib/testcases/tcEdit.php?doAction=editStep&testcase_id=243&tcversion_id=244&goback_url=http%3A%2F%2F192.168.2.4%2Ftestlink%2Flib%2Ftestcases%2FarchiveData.php%3Ftcase_id%3D243%26show_mode%3Dshow&step_id=251)  [using oscilloscope or multimeter to check the pin's](http://192.168.2.4/testlink/lib/testcases/tcEdit.php?doAction=editStep&testcase_id=243&tcversion_id=244&goback_url=http%3A%2F%2F192.168.2.4%2Ftestlink%2Flib%2Ftestcases%2FarchiveData.php%3Ftcase_id%3D243%26show_mode%3Dshow&step_id=251) [electrical level](http://www.baidu.com/link?url=ne1UuOQ_LBwhCJqZJlnqUW92wN7dwbOUWsirrER9mk2WA7spNJbIFq8p0sr7XkiYKZQGCi1szCPk9nJ7asKQDPSKu0vOehrDl6oEVEF1d1CRbIDQQQdMKzirVYeXNFai) | gpio 4 pins: low |
| 4 | repeat step 2-3 to test other pins  gpio\_output pin val  pin : gpio number (4,5,6,7)  val : 0,1 |  |

#### Test Case cb-gpio-2: gpio input test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| each pins can configure as input, we can obtain input level(high/low) | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "gpio" from device list |  |
| 2 | configure external pin 4 to low level  (connect pin on board with groud)  input "gpio\_input 4 0" | gpio\_input: test pass |
| 3 | configure external pin 4 to high level  (connect pin on board with vcc)  input "gpio\_input 4 1" | gpio\_input: test pass |
| 4 | repeat step 2-3 to test other pins  gpio\_input pin val  pin : gpio number (4,5,6,7)  val : 0,1 |  |

#### Test Case cb-gpio-3: input interrupt generate test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| input pins can generate interrupt; set low level trigger | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "gpio" from device list |  |
| 2 | configure external pin 4 to high level  (connect pin on board with vcc)  input "gpio\_int 4" |  |
| 3 | configure external pin 4 to low level | console output:  gpio\_int\_handler,status=1 count=xxx |
| 4 | configure external pin 4 to high level  then configure to low level to re-trigger the interrupt  input 'q' to quit the testcase | console output:  gpio\_int\_handler,status=1 count=xxx |
| 5 | repeat 2-4 to test other pins |  |

#### Test Case cb-gpio-4: interrupt type test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| configure input pins as level trigger(low); edge trigger (rising/falling/both) | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "gpio" from device list |  |
| 2 | configure external pin 4 to low level  input "gpio\_int\_type 4 edge rise"  configure external pin 4 to high level | console output:  gpio\_int\_handler,status=1 count=xxx |
| 3 | input " gpio\_int\_type 4 edge fall"  configure external pin 4 to low level | console output:  gpio\_int\_handler,status=1 count=xxx |
| 4 | input " gpio\_int\_type 4 edge both"  configure external pin 4 to high level  configure external pin 4 to low level | console output:  gpio\_int\_handler,status=1 count=xxx  gpio\_int\_handler,status=1 count=xxx |
| 5 | Test level trigger : high active  input " gpio\_int\_type 4 level high"  configure external pin 4 to high level | console output:  gpio\_int\_handler,status=1 count=xxx |
| 6 | Test level trigger : low active  input " gpio\_int\_type 4 level low"  configure external pin 4 to low level | console output:  gpio\_int\_handler,status=1 count=xxx |

#### Test Case cb-gpio-5: gpio debounce test test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Test enable/disable debounce ; or set diffrent debounce clk | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "gpio" from device list |  |
| 2 | Test debounce disabled ;  configure external pin 4 to high level  (connect pin on board with vcc)  input "gpio\_debounce 4 0 25000000" |  |
| 3 | configure external pin 4 to low level | console output:  gpio\_int\_handler,status=1 count=xxx |
| 4 | Test debounce enabled ;  configure external pin 4 to high level  (connect pin on board with vcc)  input "gpio\_debounce 4 1 25000000" |  |
| 5 | configure external pin 4 to low level | console output:  gpio\_int\_handler,status=1 count=xxx |
| 6 | Test debounce enabled with diffrent debounce clk setting.  configure external pin 4 to high level  (connect pin on board with vcc)  input "gpio\_debounce 4 1 10000" |  |
| 7 | configure external pin 4 to low level | console output:  gpio\_int\_handler,status=1 count=xxx |

### I2C

#### Test Case cb-i2c-1: basic master/slave mode write read test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| master/slave set as standard speed mode; 7 bit address, no dma  master: write data, then read back to verify  slave: received data and store when master write; then send back when master request read | | |
| Preconditions | | |
| connect two two i2c port of columbus board(FPGA/EVB) | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "i2c0/1/2" from device list  (both master/slave side) |  |
| 2 | slave side:  input "i2c\_slave 100000 0x45 7b 1" |  |
| 3 | master side:  input "i2c\_master 0 100000 0x45 7b 0 1" | i2c\_master: test pass |
| 4 | master side:  input "i2c\_master 0 100000 0x45 7b 10 1024" | i2c\_master: test pass |
| 5 | repeat step 4 to test other regs  cmd:i2c\_master dma\_mode speed dev addr\_mode reg len  dma\_mode :0 not using dma  speed : 100000  dev : 0x45  reg: 0--2047  len: 2048—1  reg+len should not exceed 2048 | i2c\_master: test pass |
| 6 | slave side: input 'q' to quit the slave mode |  |

#### Test Case cb-i2c-2: speed mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| master/slave set as diffrent speed  same test procedure with basic master/slave mode write read test  master/slave side should set same speed | | |
| Preconditions | | |
| connect two two i2c port of columbus board(FPGA/EVB) | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "i2c0/1/2" from device list  (both master/slave side) |  |
| 2 | slave side:  input "i2c\_slave 400000 0x45 7b 1" |  |
| 3 | master side:  input "i2c\_master 0 400000 0x45 7b 0 512" | i2c\_master: test pass |
| 4 | repeat step 3 to test other regs  cmd:i2c\_master dma\_mode speed dev addr\_mode reg len  reg: 0--2047 | i2c\_master: test pass |
| 5 | slave side: input 'q' to quit the slave mode |  |
| 6 | repeat step 2-6, to set other i2c speed  speed for standard: < 400 k  speed for fast:  400k <= speed < 3400k |  |

#### Test Case cb-i2c-3: device address mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| master/slave set 7bit and 10 bit address mode  same test procedure with basic master/slave mode write read test | | |
| Preconditions | | |
| connect two two i2c port of columbus board(FPGA/EVB) | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "i2c0/1/2" from device list |  |
| 2 | slave side:  input "i2c\_slave 100000 0x235 10b 1" |  |
| 3 | master side:  input "i2c\_master 0 100000 0x235 10b 0 1024" | i2c\_master: test pass |
| 4 | master side:  input "i2c\_master 0 100000 0x235 7b 0 1024" | i2c\_master: test fail |
| 5 | slave side: input 'q' to quit the slave mode |  |

#### Test Case cb-i2c-4: external dma support test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| master set using dma mode  same test procedure with basic master/slave mode write read test | | |
| Preconditions | | |
| connect two two i2c port of columbus board(FPGA/EVB) | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "i2c0/1/2" from device list  (both master/slave side) |  |
| 2 | slave side:  input "i2c\_slave 100000 0x45 7b 1" |  |
| 3 | master side:  input "i2c\_master 1 100000 0x45 7b 0 512" | i2c\_master: test pass |
| 4 | repeat step 4 to test other regs  cmd:i2c\_master dma\_mode speed dev addr\_mode reg len  dma\_mode :1 using dma mode  speed : 100000  dev : 0x45  addr\_mode : 7b  reg: 0--2047  len: 2048—1  reg+len should not exceed 2048 for test i2c slave device only have 2048 data reg | i2c\_master: test pass |
| 5 | slave side: input 'q' to quit the slave mode |  |

#### Test Case cb-i2c-5: interrupt test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm interrupt handle correctly when trigger some interrupt events | | |
| Preconditions | | |
| connect two two i2c port of columbus board(FPGA/EVB) | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "i2c0/1/2" from device list  (both master/slave side) |  |
| 2 | input "i2c\_int" | i2c\_int: test pass |
| 3 | input 'q' to quit test |  |

#### Test Case cb-i2c-6: slave address test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| test configurate diffrent slave address works  also check if master using a diffrent slave dev addr, test may fail | | |
| Preconditions | | |
| connect two two i2c port of columbus board(FPGA/EVB) | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "i2c0/1/2" from device list  (both master/slave side) |  |
| 2 | slave side:  input "i2c\_slave 100000 0x45 7b 1" |  |
| 3 | master side:  input " i2c\_master 0 100000 0x45 7b 0 512" | i2c\_master: test pass |
| 4 | testing use a wrong slave dev addr  master side:  input "i2c\_master 0 100000 0x55 7b 0 512" | i2c\_master: test fail |
| 5 | slave side: input 'q' to quit the slave mode |  |

#### Test Case cb-i2c-7: robust test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| master generate random reg off and length, also the data  do write/read test  until user input 'q' to quit | | |
| Preconditions | | |
| connect two two i2c port of columbus board(FPGA/EVB) | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "i2c0/1/2" from device list  (both master/slave side) |  |
| 2 | slave side:  input "i2c\_slave 100000 0x45 7b 1" |  |
| 3 | master side:  input "i2c\_robust 0 100000 0x45" | i2c\_robust:  total loop count and fail count is printed |
| 4 | robust testing using dma mode  master side: input 'q' to quit  then: input "i2c\_robust 1 100000 0x45" | i2c\_robust:  total loop count and fail count is printed |
| 5 | slave side: input 'q' to quit the slave mode  master side: input 'q' to quit the robust test |  |

### QSPI

#### Test Case cb-qspi-1: qspi flash probe test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| read the data flash id, and compare with spec  cmd format : qspi\_probe dma\_mode speed phase cs | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "qspi（spi0）" from device list |  |
| 2 | input "qspi\_probe 0 10000000 0 0" | detect spi flash: GD25Q16B  qspi\_probe: test pass |

#### Test Case cb-qspi-2: qspi flash erase test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| erase the flash block; and read back to verify，erased sector should contain all 'ff'  cmd format : qspi\_erase dma\_mode speed phase cs off len | | |
| Preconditions | | |
| cb-qspi-1: qspi flash probe test pass | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "qspi（spi0）" from device list |  |
| 2 | input "qspi\_erase 0 10000000 0 0 0 8192" | qspi\_erase: test pass |
| 3 | [options]repeat step 2 using other offset and len;  "qspi\_erase dma\_mode speed phase cs offset len" | qspi\_erase: test pass |

#### Test Case cb-qspi-3: qspi flash write/read test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| write some random data to qspi data flash; then read back to verify  note: must first probe to get flash parameters; and erase the part of write; or write may fail  cmd format : qspi\_write dma\_mode speed phase cs off len | | |
| Preconditions | | |
| cb-qspi-1: qspi flash probe test pass  cb-qspi-2:qspi flash erase test pass | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "qspi（spi0）" from device list |  |
| 2 | input "qspi\_write 0 10000000 0 0 0 8192" | qspi\_write: test pass |
| 3 | [options]repeat step 2 using other offset and len;  "qspi\_write 0 10000000 0 0 offset len" | qspi\_write: test pass |

#### Test Case cb-qspi-4: dma mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| configure using dma mode  write some random data to qspi data flash; then read back to verify  cmd format :qspi\_write dma\_mode speed phase cs off len | | |
| Preconditions | | |
| cb-qspi-1: qspi flash probe test pass  cb-qspi-2:qspi flash erase test pass | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "qspi（spi0）" from device list |  |
| 2 | input "qspi\_write 1 10000000 0 0 0 4096" | qspi\_write: test pass |
| 3 | [options]repeat step 2 using other offset and len and speed  "qspi\_write 1 speed 0 0 offset len" | qspi\_write: test pass |

#### Test Case cb-qspi-5: baudrate setting test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| set diffrent baudrate  do flash write/read test  cmd format :qspi\_write dma\_mode speed phase cs off len | | |
| Preconditions | | |
| connect two two i2c port of columbus board(FPGA/EVB) | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "qspi（spi0）" from device list |  |
| 2 | input "qspi\_write 0 1000000 0 0 0 4096" | qspi\_write: test pass |
| 3 | input "qspi\_write 0 5000000 0 0 4096 4096" | qspi\_write: test pass |
| 4 | input "qspi\_write 0 50000000 0 0 8192 8192" | qspi\_write: test pass |
| 5 | [options] you can repeat last using diffrent datarate(speed)  "qspi\_write 0 speed 0 0 offset len" | qspi\_write: test pass |

#### Test Case cb-qspi-6: interrupt test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm interupt handle correctly when trigger some interupt event  cmd format :qspi\_int dma\_mode speed phase cs | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "qspi（spi0）" from device list |  |
| 2 | input "qspi\_int 0 10000000 0 0" | qspi\_int: test pass |
| 3 | input 'q' to quit test |  |

#### Test Case cb-qspi-7: qspi direct map mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| test using direct map mode to access flash  first erase whole chip  write/read whole chip  cmd format :qspi\_direct dma\_mode speed phase cs | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "qspi（spi0）" from device list |  |
| 2 | input "qspi\_direct 0 10000000 0 0" | qspi\_direct: test pass |

#### Test Case cb-qspi-8: qspi legacy mode flash access test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| test flash access using legacy mode | | |
| Preconditions | | |
| connect two two i2c port of columbus board(FPGA/EVB) | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "qspi（spi0）" from device list |  |
| 2 | not using standard qspi driver; register direct access fifo  input "qspi\_legacy\_flash" | qspi\_legacy: test pass |

#### Test Case cb-qspi-9: qspi direct/indirect mix mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| using direct mode write some page; then using indirect mode to read and verify  cmd format :qspi\_mix dma\_mode speed phase cs wmode rmode | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "qspi（spi0）" from device list |  |
| 2 | direct write/direct read  input "qspi\_mix 0 10000000 0 0 2 2" | qspi\_mix: test pass |
| 3 | direct write/indirect read  input " qspi\_mix 0 10000000 0 0 2 0" | qspi\_mix: test pass |
| 4 | indirect write /direct read  input " qspi\_mix 0 10000000 0 0 0 2" | qspi\_mix: test pass |
| 5 | indirect write/indirect read  input " qspi\_mix 0 10000000 0 0 0 0" | qspi\_mix: test pass |

#### Test Case cb-qspi-10: qspi bit mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| test write/read using 4 bit mode/2 bit mode/1 bit mode  cmd format :qspi\_mode dma\_mode speed phase cs wmode rmode off len | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "qspi（spi0）" from device list |  |
| 2 | using 4 bit mode  input "qspi\_mode 0 10000000 0 0 2 2 0 8192" | qspi\_mode: test pass |
| 3 | using 2 bit mode read(1 bit write)  input "qspi\_mode 0 10000000 0 0 1 1 0 8192" | qspi\_mode: test pass |
| 4 | using 1 bit mode  input "qspi\_mode 0 10000000 0 0 0 0 0 8192" | qspi\_mode: test pass |

### SPI

#### Test Case cb-spi-1: spi master loopback test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| spi send some data and received at the same time, compare them should be same | | |
| Preconditions | | |
| connect spi TX/RX pin | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "spi1/2" from device list |  |
| 2 | Test loopback using fifo mode  input "spi\_loopback 0 1000000 0 0 0 8 1000" | console output:  spi\_loopback: test pass |
| 3 | Test loopback using dma mode  input "spi\_loopback 0 1000000 0 0 0 8 1000" | console output:  spi\_loopback: test pass |
| 4 | Repeat : spi\_loopback dma\_mode speed type phase cs bus\_width len  Using other parameters | spi\_loopback: test pass |

#### Test Case cb-spi-2: spi master/slave mode basic tx/rx test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| master : send some data; then read back and verify  slave: send/received according master command  use half duplex mode for spi transfer  conver fifo and dma mode | | |
| Preconditions | | |
| connect each pin of two spi port of same columbus board(FPGA/EVB) | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "spi1/2" from device list |  |
| 2 | slave side:  input "spi\_slave 1 1000000 0 0 0 8" |  |
| 3 | ‘q’ to quit spi slave dev menu  Then select "spi1/2" from device list for master dev |  |
| 4 | master side: using fifo mode  input "spi\_master 0 1000000 0 0 0 8 1000" | spi\_master: test pass |
| 5 | master side: using dma mode  input "spi\_master 1 1000000 0 0 0 8 1000" | spi\_master: test pass |
| 6 | Repeat : spi\_master dma\_mode speed type phase cs bus\_width len  Using other [len] parameter to test diffrent length | spi\_master: test pass |
| 7 | Repeat : 2-6 using other parameters  Note : master/slave should have matched parameters | spi\_master: test pass |

#### Test Case cb-spi-3: spi master/slave mode full duplex transfer test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| master : send some data; then read back and verify  slave: send/received according master command  use full duplex mode for spi transfer | | |
| Preconditions | | |
| connect each pin of two spi port of same columbus board(FPGA/EVB) | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "spi1/2" from device list |  |
| 2 | slave side:  input "spi\_slave 1 1000000 0 0 0 8" |  |
| 3 | ‘q’ to quit spi slave dev menu  Then select "spi1/2" from device list for master dev |  |
| 4 | master side: using fifo mode  input "spi\_full\_duplex 0 1000000 0 0 0 8 1000" | spi\_full\_duplex: test pass |
| 5 | master side: using dma mode  input "spi\_full\_duplex 1 1000000 0 0 0 8 1000" | spi\_full\_duplex: test pass |
| 6 | Repeat : spi\_full\_duplex dma\_mode speed type phase cs bus\_width len  Using other parameters | spi\_full\_duplex: test pass |

#### Test Case cb-spi-4: baudrate setting test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| set diffrent baudrate  use same procedure of  basic tx/rx test | | |
| Preconditions | | |
| connect each pin of two spi port of same columbus board(FPGA/EVB) | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "spi1/2" from device list |  |
| 2 | slave side:  input "spi\_slave 1 1000000 0 0 0 8" |  |
| 3 | ‘q’ to quit spi slave dev menu  Then select "spi1/2" from device list for master dev |  |
| 4 | master side: using dma mode  input "spi\_master 1 1000000 0 0 0 8 1000" | spi\_master: test pass |
| 5 | Repeat 1-4 using other baud rate setting |  |

#### Test Case cb-spi-5: transfer data width test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| configure using diffrent data width (4-32)  run same procedure of basic tx/rx test | | |
| Preconditions | | |
| connect each pin of two spi port of same columbus board(FPGA/EVB) | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "spi1/2" from device list |  |
| 2 | slave side:  input "spi\_slave 1 1000000 0 0 0 4" |  |
| 3 | ‘q’ to quit spi slave dev menu  Then select "spi1/2" from device list for master dev |  |
| 4 | master side: using dma mode  input "spi\_master 1 1000000 0 0 0 4 1000" | spi\_master: test pass |
| 5 | Repeat 1-4 using other data width setting  Data width : 4-32 |  |

#### Test Case cb-spi-7: clock phase test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| configure diffrent clock phase  run same procedure of basic tx/rx test write/read whole chip | | |
| Preconditions | | |
| connect spi port of two columbus board(FPGA/EVB) | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "spi1/2" from device list |  |
| 2 | slave side:  input "spi\_slave 1 1000000 0 0 0 8" |  |
| 3 | ‘q’ to quit spi slave dev menu  Then select "spi1/2" from device list for master dev |  |
| 4 | master side: using dma mode  input "spi\_master 1 1000000 0 0 0 8 1000" | spi\_master: test pass |
| 5 | Repeat 1-4 using other spi phase and pority set  Phase  : 0-3 |  |

#### Test Case cb-spi-8: dma mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| configure using dma mode  run same procedure with basic tx/rx test | | |
| Preconditions | | |
| connect spi port of two columbus board(FPGA/EVB) | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "spi1/2" from device list |  |
| 2 | slave side:  input "spi\_slave 1 1000000 0 0 0 8" |  |
| 3 | ‘q’ to quit spi slave dev menu  Then select "spi1/2" from device list for master dev |  |
| 4 | master side: using dma mode  input "spi\_master 1 1000000 0 0 0 8 1000" | spi\_master: test pass |
| 5 | Repeat : spi\_master dma\_mode speed type phase cs bus\_width len  Using other parameters | spi\_master: test pass |

#### Test Case cb-spi-9: interrupt test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm interrupt handle correctly when trigger some interrupt event | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "spi1/2" from device list |  |
| 2 | input "spi\_int" | spi\_int: test pass |
| 3 | input 'q' to quit test |  |

#### Test Case cb-spi-10: robust transfer test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| repeat basic tx/rx test,until user cancel it  master generate random data; send to slave, then read back to verify | | |
| Preconditions | | |
| connect spi port of two columbus board(FPGA/EVB) | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "spi1/2" from device list |  |
| 2 | slave side:  input "spi\_slave 1 1000000 0 0 0 8" |  |
| 3 | ‘q’ to quit spi slave dev menu  Then select "spi1/2" from device list for master dev |  |
| 4 | master side: using fifo mode  input "spi\_robust 0 1000000 0 0 0 8"  master side: input 'q' to quit | Console will output test procedure and display total error number |
| 5 | master side: using dma mode  input "spi\_robust 1 1000000 0 0 0 8"  master side: input 'q' to quit | Console will output test procedure and display total error number |
| 6 | Repeat 2-5 : to test other parameters setting |  |

### USB

#### Test Case cb-usb-1: device mode: mass storage class test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| configure usb as device mode; mass storage class device  connect PC and operate it;(using DDR or SD card or NANDFLASH) | | |
| Preconditions | | |
| SD CARD or NANDFLASH debug OK | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "usb0/1" from device list |  |
| 2 | input "usb\_dev\_mass 0 mmc1" | pc can access the mass storage class device successfully |
| 3 | input "usb\_dev\_mass 0 ddr" | pc can access the mass storage class device successfully |
| 4 | input 'ctrl+c' to quit the testcase |  |

#### Test Case cb-usb-2: master mode: access mass storage class udisk

|  |  |  |  |
| --- | --- | --- | --- |
| Summary | | | |
| configure usb as master mode;   support mass storage class udisk functions  connect the port using a udisk and test it  usb write random data to specified block(sector) then read back to verify | | | |
| Preconditions | | |
|  | | |
| Step | | | |
| # | Step actions | Expected Results | |
| 1 | run cdl test suit and select "usb0/1" from device list |  | |
| 2 | connect udisk to board's usb port using OTG cable  input "usb\_host\_mass 0 1000 10 5" | usb\_host\_mass: test pass | |

#### Test Case cb-usb-3: master mode: access mass storage class udisk using fix data

|  |  |  |
| --- | --- | --- |
| Summary | | |
| configure usb as master mode;   support mass storage class udisk functions  connect the port using a udisk and test it  usb write fixed data to specified block(sector) then read back to verify; then we can check udisk at PC side | | |
| Preconditions | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "usb0/1" from device list |  |
| 2 | connect udisk to board's usb port using OTG cable  input "usb\_host\_mass\_fix 0 1000 10 " | usb\_host\_mass\_fix: test pass |
| 3 | Check the udisk at PC side,block: 1000—1010 | Should contain data:0-ff sequence |

#### Test Case cb-usb-4: device mode: mass storage class test using dma mode

|  |  |  |
| --- | --- | --- |
| Summary | | |
| configure usb as device mode; mass storage class device  configure using dma mode  connect PC and operate it;(using DDR or SD card or NANDFLASH) | | |
| Preconditions | | |
| SD CARD or NANDFLASH debug OK | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "usb0/1" from device list |  |
| 2 | input "usb\_dev\_mass 1 mmc1"  according actual mmc interface "mmc0" or "mmc1" | pc can access the mass storage class device successfully |
| 3 | input 'ctrl+c' to quit the testcase |  |

#### Test Case cb-usb-5: master mode: access mass storage class udisk using dma mode

|  |  |  |
| --- | --- | --- |
| Summary | | |
| configure usb as master mode;   support mass storage class udisk functions  configure using dma mode  connect the port using a udisk and test it  usb write random data to specified block(sector) then read back to verify | | |
| Preconditions | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "usb0/1" from device list |  |
| 2 | connect udisk to board's usb port using OTG cable  input "usb\_host\_mass 1 1000 10 5" | usb\_host\_mass: test pass |

#### Test Case cb-usb-6: otg feature test

|  |  |  |
| --- | --- | --- |
| Summary | | |
|  | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "usb0/1" from device list |  |
| 2 |  |  |
|  |  |  |

#### Test Case cb-usb-7: interrupt test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm interrupt handle correctly when trigger interrupt event | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "usb0/1" from device list | 1 |
| 2 | input "usb\_int" | 2 |
| 3 | input 'q' to quit test | 3 |

### NANDFLASH

#### Test Case cb-nfc-1: nand probe test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| read the flash id, and compare with spec | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "nandflash" from device list |  |
| 2 | input "nand\_probe 1 0" | nand\_probe: test pass |

#### Test Case cb-nfc-2: nand block erase test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| erase the flash block; and read back to verify，erased sector should contain all 'ff' | | |
| Preconditions | | |
| cb-nfc-1: nand probe test pass | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "nandflash" from device list |  |
| 2 | input "nand\_erase 1 0 0" | nand\_erase: test pass |
| 3 | [options]repeat step 2 using other position  "nand\_erase dma\_mode cs block\_off"  Dma\_mode : 0 using fifo mode 1 dma mode  Cs : 0 | nand\_erase: test pass |

#### Test Case cb-nfc-3: basic nand read/write test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| write some pages,then read back to verify  using dma mode | | |
| Preconditions | | |
| cb-nfc-1: nand probe test pass  cb-nfc-2: nand erase test pass | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "nandflash" from device list |  |
| 2 | write pages in block 0 using raw mode(no ecc)  input "nand\_write 1 0 1 0" | nand\_write: test pass |
| 3 | write pages in block 0 using ecc mode  input "nand\_write 1 0 0 0" | nand\_write: test pass |
| 4 | [options]repeat step 2 using other position  "nand\_write dma\_mode cs rw\_mode block\_off"  dma\_mode: 0 using fifo ; 1 using dma  rw\_mode: 0 ecc 1 raw | nand\_write: test pass |

#### Test Case cb-nfc-4: nand fifo mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| testing fifo mode(default dma mode) | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "nandflash" from device list |  |
| 2 | write pages in block 0 using raw mode(no ecc)  input "nand\_write 0 0 1 0" | nand\_write: test pass |
| 3 | write pages in block 0 using ecc mode  input "nand\_write 0 0 0 0"  dma\_mode: 0 using fifo ; 1 using dma  rw\_mode: 0 ecc 1 raw | nand\_write: test pass |

#### Test Case cb-nfc-5: ecc support test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| configure use diffrent ecc setting  do basic read/write test | | |
| Preconditions | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "nandflash" from device list |  |
| 2 | ecc: 2 bit  input "nand\_ecc 1 0 2 0" | nand\_ecc: test pass |
| 3 | ecc: 4 bit  input "nand\_ecc 1 0 4 0" | nand\_ecc: test pass |
| 4 | ecc: 8 bit  input "nand\_ecc 1 0 8 0" | nand\_ecc: test pass |
| 5 | ecc: 16 bit  input "nand\_ecc 1 0 16 0" | nand\_ecc: test pass |

#### Test Case cb-nfc-6: interrupt test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm interrupt handle correctly when trigger some interrupt event | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "nandflash" from device list |  |
| 2 | input "nand\_int 0 0" | nand\_int: test pass |

### MMC/SDIO

#### Test Case cb-mmc-1: sd memory card probe test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| sd card init and identify the card,also obtain the card info(capacity,block size,etc) | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "mmc0/1" from device list |  |
| 2 | input "mmc\_probe 0 25000000 4" | mmc card information...  mmc\_probe: test pass |

#### Test Case cb-mmc-2: sd memory card single block read/write test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| single block mode  write some blocks, then read back to verify | | |
| Preconditions | | |
| cb-mmc-1: sd memory card probe test pass | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "mmc0/1" from device list |  |
| 2 | input "mmc\_single 0 25000000 4 0 10" | mmc\_single: test pass |
| 3 | input "mmc\_single 0 25000000 4 100 100" | mmc\_single: test pass |
| 4 | [options]repeat step3 using other block start/cnt  "mmc\_single 0 25000000 4 lba cnt" | mmc\_single: test pass |

#### Test Case cb-mmc-3: sd memory card multi-block read/write test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| using multi-block mode  write some blocks then read back to verify | | |
| Preconditions | | |
| cb-mmc-1: sd memory card probe test pass | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "mmc0/1" from device list |  |
| 2 | input "mmc\_multi 0 25000000 4 0 100" | mmc\_multi: test pass |
| 3 | input "mmc\_multi 0 25000000 4 100 200" | mmc\_multi: test pass |
| 4 | [options]repeat step3 using other block start/cnt  "mmc\_multi 0 25000000 4 lba cnt" | mmc\_multi: test pass |

#### Test Case cb-mmc-4: sd memory card data bus width test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| configure as 1-bit 2-bit 4-bit  do multi-block read/write test | | |
| Preconditions | | |
| cb-mmc-1: sd memory card probe test pass | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "mmc0/1" from device list |  |
| 2 | 4 bit mode  input "mmc\_multi 0 25000000 4 100 200" | mmc\_multi: test pass |
| 3 | 1 bit mode  input "mmc\_multi 0 25000000 1 100 200" | mmc\_multi: test pass |
| 4 | 2 bit mode  input "mmc\_multi 0 25000000 2 100 200" | mmc\_multi: test pass |

#### Test Case cb-mmc-5: sd memory card transfer date rate test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| configure diffrent data rate  do multi-block read/write test | | |
| Preconditions | | |
| cb-mmc-1: sd memory card probe test pass | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "mmc0/1" from device list |  |
| 2 | input "mmc\_multi 0 25000000 4 100 200" | mmc\_multi: test pass |
| 3 | input "mmc\_multi 0 5000000 4 100 200" | mmc\_multi: test pass |
| 4 | input "mmc\_multi 0 52000000 4 100 200" | mmc\_multi: test pass |
| 5 | [options]repeat step4 using other data rate  "mmc\_multi 0 date-rate 4 lba cnt" | mmc\_multi: test pass |

#### Test Case cb-mmc-6: sd memory card dma mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| configure using dma mode  do multi-block read/write test | | |
| Preconditions | | |
| cb-mmc-1: sd memory card probe test pass | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "mmc0/1" from device list |  |
| 2 | input "mmc\_multi 1 25000000 4 100 200" | mmc\_multi: test pass |
| 3 | [options]repeat step2 using other data rate,block start,block cnt  "mmc\_multi 1 speed 4 lba cnt" | mmc\_multi: test pass |

#### Test Case cb-mmc-7: sd memory card interrupt test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm interrupt handle correctly when trigger some interrupt event | | |
| Preconditions | | |
| cb-mmc-1: sd memory card probe test pass | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "mmc0/1" from device list |  |
| 2 | input "mmc\_int" | mmc\_int: test pass |
| 3 | input 'q' to quit test |  |

#### Test Case cb-mmc-8: sdio card register access test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| we can access wifi adapter register(read/write) | | |
| Preconditions | | |
| connect with sdio peripheral: sdio wifi adapter | | |
| Step | | |
| # | Step actions | Expected Results |
|  |  |  |
|  |  |  |
|  |  |  |

#### Test Case cb-mmc-9: sdio card memory access test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| we can access wifi adapter memory(read/write) | | |
| Preconditions | | |
| connect with sdio peripheral: sdio wifi adapter | | |
| Step | | |
| # | Step actions | Expected Results |
|  |  |  |
|  |  |  |
|  |  |  |

#### Test Case cb-mmc-10: sdio card detect test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| we can detect sdio card,and read the vendor information | | |
| Preconditions | | |
| cb-mmc-1: sd memory card probe test pass | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "mmc0/1" from device list |  |
| 2 | input "mmc\_int" | mmc\_int: test pass |
| 3 | input 'q' to quit test |  |

### IPC

#### Test Case cb-ipc-1:

|  |  |  |
| --- | --- | --- |
| Summary | | |
|  | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
|  |  |  |
|  |  |  |

### ADC

#### Test Case cb-adc-1: adc single mode single channel test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| test single scan mode with only one channel enable | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "adc" from device list |  |
| 2 | input "adc\_ss 0" | according console print ch 0 data value to judge if it's pass or fail |
| 3 | [repeat] you can repeat "adc\_ss ch" to do ADC operation and check the result |  |

#### Test Case cb-adc-2: adc single mode multi-channel test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| single mode,multiple channel enabled | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "adc" from device list |  |
| 2 | input "adc\_sm" | according console print ch 1/3/7 data value to judge if it's pass or fail |

#### Test Case cb-adc-3: adc continuous mode single channel test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| continuous mode with only sone channel select | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "adc" from device list |  |
| 2 | input "adc\_cs 0" | according console print ch 0 data value to judge if it's pass or fail |
| 3 | [repeat] you can repeat "adc\_cs ch" to do ADC operation and check the result |  |

#### Test Case cb-adc-4: adc continuous mode multi-channel test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| continuous mode, multiple channel select | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "adc" from device list |  |
| 2 | input "adc\_cm" | according console print ch 1/3/7 data value to judge if it's pass or fail |

#### Test Case cb-adc-5: adc interrupt test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm interrupt handle correctly | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "adc" from device list |  |
| 2 | input "adc\_int" | adc\_int\_handler,status=%x  adc\_int: test pass |

#### Test Case cb-adc-6: dma mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| test dma mode works using continuous mode single channel test | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "adc" from device list |  |
| 2 | input "adc\_dma" | according console print ch 0 data value to judge if it's pass or fail |

### ECC

#### Test Case cb-ecc-1: ecc basic test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| test if domain parameter curve of elliptic curve are correct, G=(order+1)G  Include ecc : 192 bit, 224 bit, 256 bit, 384 bit | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "ecc" from device list |  |
| 2 | Input "ecc\_test\_basic" | ecc\_test\_basic: test passed! |

#### Test Case cb-ecc-2: ecc sign/verify test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| test ecc sign hash and ecc verify signature  Include ecc : 192 bit, 224 bit, 256 bit, 384 bit | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "ecc" from device list |  |
| 2 | Input "ecc\_test\_sign\_verify" | ecc\_test\_sign\_verify: test passed! |

#### Test Case cb-ecc-3: ecc shamir test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| test ECC  Shamir's Trick, which use R = kP + lQ for calculate acceleration in ecc verify signature | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "ecc" from device list |  |
| 2 | Input "ecc\_test\_shamir" | ecc\_test\_shamir: test passed! |

### AES

#### Test Case cb-aes-1: aes ecb mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Test aes ecb mode encrypt/decrypt, and compare with sw algorithm  Usng random key(128/256 bits), random plain data | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "aes" from device list |  |
| 2 | Using ‘128’ bits key  Input "aes\_ecb 0 128" | Aes\_ecb: test pass |
| 3 | Using ‘256’ bits key  Input "aes\_ecb 0 256" | Aes\_ecb: test pass |

#### Test Case cb-aes-2: aes cbc mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Test aes cbc mode encrypt/decrypt, and compare with sw algorithm  Usng random key(128/256 bits), random iv,random plain data | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "aes" from device list |  |
| 2 | Using ‘128’ bits key  Input "aes\_cbc 0 128" | Aes\_cbc: test pass |
| 3 | Using ‘256’ bits key  Input "aes\_cbc 0 256" | Aes\_cbc: test pass |

#### Test Case cb-aes-3: aes ctr mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Test aes ctr mode encrypt/decrypt, and compare with sw algorithm  Usng random key(128/256 bits), random iv,random plain data | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "aes" from device list |  |
| 2 | Using ‘128’ bits key  Input "aes\_ctr 0 128 0 1 " | Aes\_ctr: test pass |
| 3 | Using ‘256’ bits key  Input "aes\_ctr 0 256 0 1" | Aes\_ctr: test pass |

#### Test Case cb-aes-4: aes ccm mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Test aes ccm mode encrypt/decrypt, and compare with sw algorithm  Usng random key(128/256 bits), random nonce data,random aad data,random plain data | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "aes" from device list |  |
| 2 | Using ‘128’ bits key  Input "aes\_ccm 0 128 " | Aes\_ccm: test pass |
| 3 | Using ‘256’ bits key  Input "aes\_ccm 0 256 " | Aes\_ccm: test pass |
| 4 | Test diffrent nonce len,aad len, mac len  Input "aes\_ccm 0 256 0 noncelen aadlen maclen"  Noncelen : 1-14  Aadlen : 0-64  Maclen :1-15 | Aes\_ccm: test pass |

#### Test Case cb-aes-5: aes gcm mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Test aes gcm mode encrypt/decrypt, and compare with sw algorithm  Usng random key(128/256 bits), random nonce data,random aad data,random plain data | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "aes" from device list |  |
| 2 | Using ‘128’ bits key  Input "aes\_gcm 0 128 " | Aes\_gcm: test pass |
| 3 | Using ‘256’ bits key  Input "aes\_gcm 0 256 " | Aes\_gcm: test pass |
| 4 | Test diffrent nonce len,aad len, mac len  Input "aes\_gcm 0 256 0 ivlen aadlen maclen"  ivlen : 1-14  Aadlen : 0-64  Maclen :1-15 | Aes\_gcm: test pass |

#### Test Case cb-aes-5: aes cmac mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Test aes cmac mode encrypt, and compare with sw algorithm  Usng random key(128/256 bits), random plain data | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "aes" from device list |  |
| 2 | Using ‘128’ bits key  Input "aes\_cmac 0 128 " | Aes\_cmac: test pass |
| 3 | Using ‘256’ bits key  Input "aes\_cmac 0 256 " | Aes\_cmac: test pass |

#### Test Case cb-aes-6: aes test using otp key

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Test aes encrypt/decrypt using otp key, and compare with sw algorithm | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | Run cb-otp-6: otp secure full test first; so otp contain valid key |  |
| 2 | run cdl test suit and select "aes" from device list |  |
| 3 | Test ecb mode  Input "aes\_ecb 0 256 1" | Aes\_ecb: test pass |
| 4 | Test cbc mode  Input "aes\_cbc 0 256 1" | Aes\_cbc: test pass |
| 5 | Test ctr mode  Input "aes\_ctr 0 256 1 1 | Aes\_ctr: test pass |
| 6 | Test ccm mode  Input "aes\_ccm 0 256 1" | Aes\_ccm: test pass |
| 7 | Test gcm mode  Input "aes\_gcm 0 256 1" | Aes\_gcm: test pass |
| 8 | Test cmac mode  Input "aes\_cmac 0 256 1" | Aes\_cmac: test pass |

### SHA

#### Test Case cb-sha-1: sha256 checksum test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Generate a random data buffer, calculate sha256 checksum using hw ;  Then calculate using sw code to compare verify hw works correctly.  Test diffrent length : at lease conver 0,1,2 segments | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "sha" from device list |  |
| 2 | Buffer len : 4 byte  Input "sha256 4" | Sha256: test pass |
| 3 | Buffer len : 64 byte  Input "sha256 64" | Sha256: test pass |
| 4 | Buffer len : 100 byte  Input "sha256 100" | Sha256: test pass |
| 5 | Buffer len : 128 byte  Input "sha256 128" | Sha256: test pass |
| 6 | Buffer len : 200 byte  Input "sha256 200" | Sha256: test pass |
| 7 | Repeat using other test buffer length  Sha256 len | Sha256: test pass |

#### Test Case cb-sha-2: sha384 checksum test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Generate a random data buffer, calculate sha384 checksum using hw ;  Then calculate using sw code to compare verify hw works correctly.  Test diffrent length : at lease conver 0,1,2 segments | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "sha" from device list |  |
| 2 | Buffer len : 4 byte  Input "sha384 4" | sha384: test pass |
| 3 | Buffer len : 64 byte  Input " sha384 64" | sha384: test pass |
| 4 | Buffer len : 100 byte  Input " sha384 100" | sha384: test pass |
| 5 | Buffer len : 128 byte  Input " sha384 128" | sha384: test pass |
| 6 | Buffer len : 200 byte  Input " sha384 200" | sha384: test pass |
| 7 | Repeat using other test buffer length  sha384 len | sha384: test pass |

#### Test Case cb-sha-3: interrupt test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| confirm interrupt handle correctly when trigger some interrupt event | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "sha" from device list |  |
| 2 | Input " sha\_int" | sha\_int: test pass |

#### Test Case cb-sha-4: robust test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Generate random length random data ; do sha384/sha256 calculate and compare the result with sw calculated | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "sha" from device list |  |
| 2 | Input " sha\_robust" | Total failed should be ‘0’ |

### OTP

#### Test Case cb-otp-1: otp non-secure program test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| using otp test mode  program one otp bit of non-secure area; then read back to verify | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | Power down the board then power on to remove the effect of previous test  run cdl test suit and select "otp" from device list |  |
| 2 | input "otp\_set\_onebit non-secure 4 1" | otp\_non\_secure\_set\_onebit: test pass |
| 3 | Do soft reset, and reload the cdl |  |
| 4 | otp\_get\_onebit non-secure 4 1 | Val should be ‘1’ |
| 5 | [repeat]input "otp\_set\_onebit non-secure row col"  test other position  note : don’t set ‘row <4’ ; it may disable hw interface,may have side effect for other test | otp\_non\_secure\_set\_onebit: test pass |

#### Test Case cb-otp-2: otp secure program test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| using otp test mode  program one otp bit of secure area; then read back to verify | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | Power down the board then power on to remove the effect of previous test  run cdl test suit and select "otp" from device list |  |
| 2 | input " otp\_set\_onebit secure 0 1" |  |
| 3 | Do soft reset, and reload the cdl ; or secure otp can’t be read correctly |  |
| 4 | otp\_get\_onebit secure 0 1 | Val should be ‘1’ |
| 5 | [repeat] step 2-4 to test other location  Note : don’t test the bit only hw can access |  |

#### Test Case cb-otp-3: key disable test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| using otp test mode  disable key; then verify we can't read correct setting of the key | | |
| Preconditions | | |
| some specified key should program before this test | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | Power down the board then power on to remove the effect of previous test  run cdl test suit and select "otp" from device list |  |
| 2 | input " otp\_test secure" |  |
| 3 | Do soft reset, and reload the cdl ; or secure otp can’t be read correctly |  |
| 4 | otp\_get\_sec\_data | secure: test pass |
| 5 | Check before disable,correct value can be read  otp\_key\_disable 0  Display all secure key can be access by sw  otp\_secure\_show | otp\_key\_disable: test pass |
| 6 | Check after disable,can’t read correct value, read always ‘0’  otp\_key\_disable 1  Display all secure key can be access by sw  otp\_secure\_show | otp\_key\_disable: test pass |

#### Test Case cb-otp-4: otp interrupt handler test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| test interrupt handle correctly | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "otp" from device list |  |
| 2 | input "otp\_int" | otp\_int: test pass |

#### Test Case cb-otp-5: otp non-secure full test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| using otp test mode  program random data of all bit of non-secure area; then read back to verify | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | Power down the board then power on to remove the effect of previous test  run cdl test suit and select "otp" from device list |  |
| 2 | input " otp\_test non-secure" | otp\_non\_secure: test pass |
| 3 | input " otp\_non\_secure\_show " to view all non-secure data |  |

#### Test Case cb-otp-6: otp secure full test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| using otp test mode  program specfied fixed data of all bit of secure area; then read back to verify  note : the secure area only access by hw readback should be zero | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | Power down the board then power on to remove the effect of previous test  run cdl test suit and select "otp" from device list |  |
| 2 | input " otp\_test secure" |  |
| 3 | Do soft reset, and reload the cdl ; or secure otp can’t be read correctly |  |
| 4 | otp\_get\_sec\_data | secure: test pass |
| 5 | input " otp\_ secure\_show " to view all secure data can be accessed by sw |  |

#### Test Case cb-otp-7: otp non-secure byte program test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| using otp test mode  program one otp byte of non-secure area; then read back to verify | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | Power down the board then power on to remove the effect of previous test  run cdl test suit and select "otp" from device list |  |
| 2 | input "otp\_set\_onebyte non-secure 4 0x2f" | otp\_non\_secure,set\_byte: test pass |
| 3 | Do soft reset, and reload the cdl |  |
| 4 | otp\_get\_onebyte non-secure 4 | Val should be ‘0x2f’ |
| 5 | [repeat]input "otp\_set\_onebyte non-secure row val"  test other position  note : don’t set ‘row <4’ ; it may disable hw interface,may have side effect for other test | otp\_non\_secure,set\_byte: test pass |

#### Test Case cb-otp-8: otp secure byte program test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| using otp test mode  program one otp byte of secure area; then read back to verify | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | Power down the board then power on to remove the effect of previous test  run cdl test suit and select "otp" from device list |  |
| 2 | input " otp\_set\_onebyte secure 4 0x2f" |  |
| 3 | Do soft reset, and reload the cdl ; or secure otp can’t be read correctly |  |
| 4 | otp\_get\_onebit secure secure 4 | Val should be ‘0x2f’ |
| 5 | [repeat] step 2-4 to test other location  Note : don’t test the bit only hw can access |  |

#### Test Case cb-otp-9: otp interface disable test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| using otp test mode  program bit of first 4 byte of non-secure area; then read back to verify  then check coresponding hw interface functions | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | Power down the board then power on to remove the effect of previous test  run cdl test suit and select "otp" from device list |  |
| 2 | Disble uart 1 interface  input " otp\_set\_onebit non-secure 1 1" |  |
| 3 | Do soft reset, and reload the cdl |  |
| 4 | otp\_get\_onebit non-secure 1 1  you can list config using " otp\_config\_show " | Val should be ‘0x1’ |
| 5 | Run uart1 test case | Uart 1 test case should fail |
| 6 | [repeat] step 2-5 to test other interface(change parametes according your test target)  Please refer the OTP document for which otp bit corepoding which function/interface |  |

### CRC

#### Test Case cb-crc-1: calculate crc16 xmodem mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Give specified length, generate random input data, and calculate crc16 xmodem;  Then compare with software crc16 | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "crc" from device list |  |
| 2 | input "crc\_xmodem 100" | crc\_xmodem : test pass. |
| 3 | input " crc\_xmodem 10240" | crc\_xmodem : test pass. |
| 4 | Calculate when length is not word align  input " crc\_xmodem 101" | Reject for invalid parameter:  length should be mutiple of 4 |

#### Test Case cb-crc-2: calculate crc16 ccitt mode test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Give specified length, generate random input data, and calculate crc16 ccitt;  Then compare with software crc16 | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "crc" from device list |  |
| 2 | input "crc\_ccitt 100" | crc\_ ccitt: test pass. |
| 3 | input " crc\_ ccitt 10240" | crc\_ ccitt: test pass. |
| 4 | Calculate when length is not word align  input " crc\_ ccitt 101" | Reject for invalid parameter:  length should be mutiple of 4 |

#### Test Case cb-crc-3: calculate crc16 xmodem using multiple buffer

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Generate five 2k buffer with random data, and calculate crc16 xmodem one by one,  Simulate scatter buffers crc16 calculation.  Compare with sw algorithm to verify | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "crc" from device list |  |
| 2 | input "crc\_xmodem 100" | crc\_xmodem : test pass. |
| 3 | input " crc\_xmodem 10240" | crc\_xmodem : test pass. |
| 4 | Calculate when length is not word align  input " crc\_xmodem 101" | Reject for invalid parameter:  length should be mutiple of 4 |

#### Test Case cb-crc-2: crc16 robust test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| generate random length random input data, and calculate crc16 ;  Then compare with software crc16 | | |
| Preconditions | | |
| Cb-crc-1 passed | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "crc" from device list |  |
| 2 | input "crc\_robust" | 8 loop len=5944 pass,total failed 0 |
| 3 | input "q" to quit | Report total failed should be ‘0’ |

### RNG

#### Test Case cb-rng-1: random gen manually check

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Print out random number generated by ‘rng’ hw and check | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "rng" from device list |  |
| 2 | Input " rng\_manual" |  |
| 3 | Press any key to display 4 random int number |  |
| 4 | Press ‘q’ to quit the test |  |

#### Test Case cb-rng-2: random gen quality check

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Generate a group of random number and evaulate the quality of random  Result is the diffrent of result pi with the constant ‘PI’ ; little is good | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "rng" from device list |  |
| 2 | SW random generator  Input "rng\_quality 1" |  |
| 3 | SW random generator, each time we change the random seed using timer  Input "rng\_quality 2" |  |
| 4 | RNG hw generator  Input "rng\_quality 0" | Diff is at the level 10^(-3) |

### EPF

#### Test Case cb-epf-1: epf trigger test

|  |  |  |
| --- | --- | --- |
| Summary | | |
| Trigger epf by switch power off | | |
| Preconditions | | |
|  | | |
| Step | | |
| # | Step actions | Expected Results |
| 1 | run cdl test suit and select "epf" from device list |  |
| 2 | Input "epf\_test" |  |
| 3 | connect gpio\_pmu\_sel(TP23(J62H\_31)) to high; so gpio pins function as PMU output  GPIO0[0] with arm\_sleep\_mode; 🡺 1  GPIO0[1] with arm\_pdn\_en; 🡺1  GPIO0[2] with plc\_sleep\_mode;  GPIO0[3] with plc\_pdn\_en;  GPIO0[4] with rf\_sleep\_mode;  GPIO0[5] with rf\_pdn\_en;  GPIO0[6] with fs\_timer\_cnt\_en;  GPIO0[7] with tcxo\_en. |  |
| 4 | Pull epf input to low (TP22,connect to GND) | epf\_int\_handler,status=1  epf high to low triggered  epf housekeeping done  you can view GPIO for PMU output state |
| 5 | Disconnect EPF INPUT from GND(back to default 1.8 V) | Cdl test restart |