Major Feature	Minor Feature
HMAC384	
	single and multi-block messages
	Reset test
	Re-init between hmac operations
	Interface
	Registers I/O
	Other registers that get decoded to HMAC
	Performance test
	Nightly regression with mbedlts checker enabled
	Error conditions triggers and status
ECC	
	Key Gen
	Key Sign
	Key Verify
	Reset Test
	Interface
	Registers I/O
	Other registers that get decoded to ECC
	Performance test
	Nightly regression with mbedlts checker enabled
	Error conditions triggers and status
HMAC DRBG	
	Reset Test
	Interface
	Registers I/O
	Other registers that get decoded to ECC
	Performance test
	Nightly regression with mbedlts checker enabled
	Error conditions triggers and status
SHA384/512/256	
	Reset Test
	Interface
	Registers I/O
	Other registers that get decoded to SHA384
	Performance test
	Nightly regression with SHA checker enabled
	Error conditions triggers and status
KeyVault Flows	

Major Feature	Minor Feature
	UDS Flow (Cold boot UDS deobf runs, but on warm boot, even if the register bits are written, the flow itself should not be executed)
	FE Flow (Cold boot FE deobf runs, but on warm boot, even if the register bits are written, the flow itself should not be executed)
	HMAC
	SHA
	ECC Read/Write from KV
	Lock, clear dest mask operation
	Reset Test
	Interface
	Registers I/O
	Error conditions triggers and status
PCR Vault Flows	
	PCR Extend
	PCR signing
	PCR register properties & reset
Data Vault	
	Register access val (RW & lock aspects)
	Register reset val (stickiness)
SoC Interface	
	Register access
	SoC to uC basic protocol
	uC to SoC basic protocol
	Arbitration Conflicts with both requests arising at the same time
	PAUSER based filtering
	Mailbox being used by a different entity than the 'locking' entity
	Mailbox force-unlock in the middle of multiple commands
	Unaligned register writes
	Mailbox interrupt to UC flows

Major Feature	Minor Feature
	Mailbox flows crossed with warm reset
	Mailbox flows crossed with cold reset
	Mailbox unperturbed under FW update reset
	Mailbox DLEN violation - DLEN larger than Mailbox
	Mailbox DLEN violation - Sender writes more data than DLEN
	Mailbox DLEN violation - Sender writes more data than Mailbox size
	Mailbox DLEN violation - Receiver reads more data than DLEN
	Mailbox DLEN violation - Receiver reads more data than Mailbox size
	Mailbox DLEN violation - Receiver writes more response data than DLEN
	Mailbox delay handling - Sender/Receiver injects random delays throughout mailbox procedure
	Mailbox - sender tries to read dataout
	Mailbox - receiver tries to write datain
	Mailbox - Check data received by uC matches data sent by SoC
	Mailbox - Check response data received by SoC matches data sent by uC
	Mailbox - Check data received by SoC matches request data sent by uC
	Fuse register writes
	Fuse register unauthorized updates being dropped to WO fuses
	Read of fuse registers must be blocked for unauthorized fuses (eg. UDS, FE etc.); Follow the arch spec
	FW Update flow with associated reset
	FW Update flow and Key vault behavior
	FW Update flow and PCR vault behavior
	FW Update flow loads image to ICCM/DCCM without corruption

Major Feature	Minor Feature
	Security access controls change, key/asset flushing
	Warm & Cold Reset and Key vault behavior
	Warm & Cold Reset and PCR vault behavior
	Security access controls change, key/asset flushing across warm & Cold resets
	idle clock gating entry & exit
	Boot flow
	Warm reset flow
	Crypto logic flush on security state transition
	TOP Interface input wires to interrupt trigger
	Arch & Fuse registers should be accessible even when the mailbox is in progress
	Test SHA acceleration HW API - cross it with mailbox flows from internal facing & SOC facing locks as well as access other registers during this process to ensure they progress appropriately with the right data/flow while SHA acceleration results match as expected
	FW Update reset should ensure that SHA block is atomic to internal FW - or wait for the SHA operation to complete and give the round robin to FW if it has requested or about to request as a part of FW update reset
	SOC write to locked fuses get silently dropped
	Fuse rewrite on ready_for_fuse being written by the FW and bootFSM although transitions to BOOT_DONE again, none of the uController or others signals should be effected - No fuses should be updateable
	Mailbox data available read happening at the same time as a SOC register access
	TRNG REQ API Verification
	Caliptra TOP level JTAG Validation
	SOC-IFC interrupt register validation
RiscV Core (uC)	
	Exceptions

Major Feature	Minor Feature
	Interrupts
	Timers
	ICCM lock flow
	ROM basic execution flow
	ICCM instruction execution with data population to DCCM
	Mailbox to ICCM Copy and Execute
	Mailbox to ICCM copy in two separate chunks (like FMC, RT)
	Mailbox to DCCM copy of data blob
	Mailbox to ICCM and DCCM copy; Execute ICCM; Lock ICCM; DMA Access ICCM and Cause NMI
	Mailbox to ICCM and DCCM copy; Execute ICCM and write to DCCM; Lock ICCM; Execute ICCM and write to DCCM
	Writing to error registers when ICCM/DCCM copy has a failure (exceptions?)
	ICCM lock should be write once until a FW update reset happens
	uC access to ROM on LSU/datapath
Internal Global	
	Internal register validation & coverage
	per periphiral interrupt validation (interrupt enables/disables, status clearing, triggers)
	reset domain validation
	Output register/wire control validation
	FW faulty interference on cryptos while key vault triggered flows are happening
	NMI validation
	Configuration validation w/ Negative testing
	Fuse download
	Subtractive decoding path testing
Error validation	
	SRAM ECC
	uController has parity?
	Crypto errors?
	APB protocol errors?
	AHB-lite errors?
	Any FW triggered writes to signal fatal errors

Major Feature	Minor Feature
	WD (that needs petting) triggered error
AHB Decoder	
	Address decoding - global
	Address decoding - peripherals internal
	Non accessible decoding
UART (instrumentation)	
Verilator	

Description
HMAC receives 384-bit key and input message of variable length (single and multi-block messages) to generate 384-bit tag. The HMAC_random_test also exercises reset case
Assert reset while doing HMAC operation, and make sure that everything is cleared.
Verify HMAC of a message completed successfully before re-init for the next message
AHB-lite interface access validation
Register rd/wr from crypto side and ahb-lite side
As Col.B says
As Col.B says
Continual regression
Validate crypto defined error conditions if any
ECC receives scalar input and generates two outputs of pubkey.
ECC receives the hashed message, privkey, and nonce, and generates the signature R and S.
ECC receives the hashed message, pubkey, and signature R and S, and validate the signature.
AHB-lite interface access validation
Register rd/wr from crypto side and ahb-lite side
As Col.B says
As Col.B says
Continual regression
Validate crypto defined error conditions if any
As Col.B says
AHB-lite interface access validation
Register rd/wr from crypto side and ahb-lite side
As Col.B says
As Col.B says
Continual regression
Validate crypto defined error conditions if any
As Col.B says
AHB-lite interface access validation
Register rd/wr from crypto side and ahb-lite side
As Col.B says
As Col.B says
Continual regression
Validate crypto defined error conditions if any

Description
Also verifies obfuscation key behavior on latching only on powergood
Same as above
Crypto's usage of KV as a client and server
Crypto's usage of KV as a client and server
Crypto's usage of KV as a client and server
KV function testing
As Col.B says
AHB-lite interface access validation
Register rd/wr from crypto side and ahb-lite side
Validate KV defined error conditions if any
PCR extension functionality per PCR entry
PCR signing flow with KV7 key
Ensure lock attributes and reset attribute per PCR entry work correctly
Read/write tests to registers to ensure accesses aligns with read/write permissions defined in the spec. Validate CPTRA_RESET_REASON to ensure the WARM and FW_UPD_RESET bits are being appropriately set on the warm & fw_upd reset conditions
Verifies SoC to uC communication via mailbox
Verifies uC to SoC communication via mailbox
uC and SOC access to mailbox happening concurrently, while one is using it. Mailbox is atomic
User A has a lock but user B targetting the registers User A has the lock and populating the data; User B is interleaving the data write requests; User A has the lock & poulated the data; User B tries to access the output User A does all the flow; User B tries to set execute User A does all the flow; User B tries to unlock If uC force unlock is implemented, test that out while SOC User has locked the mailbox and the flow is at various stages of the mailbox protocol
One of the cases above
Needs more thought - this will almost certainly hang the test
Register access are aligned to 32-bit boundary - unaligned access get force aligned by HW and return the data
Mailbox lock req indication when uC has it locked Mailbox exec indication thru interrupt

Description
Mailbox is at various stages of execution (lock, ready, data-in, exec, data-out, wait for unlock) and warm reset happens
Same as above with cold reset
Same as above but mailbox & all registers should remain the same
--- need to include other variants of violating response data length requirement --- How is the response data length qualified?
Fuses are being written correctly, locked (example: UDS, FE) correctly post write for certain fuses; Warm reset can rewrite the fuses but no change should happen to fuses are locked. FUSE_WR_DONE should block the write of the fuses once that bit is set to '1. To reupdate the fuses, Caliptra FW should reset that bit to zero and then SOC can update Any update to In-field programmable (IFP) fuses inside caliptra fuse registers will require cold boot. Meaning all fuses except this are sticky. Note: At SOC & system level IFP fuses inside the physical fuse bank can be updated without a cold reset (that is outside of Caliptra RTL's implementation context). All fuses MUST be “LOCKED” on FUSE_WR_DONE and even IFPs after fuses are updated CANNOT be downloaded/written back into RTL. Implying they are write ONCE. FUSE_WR_DONE can be reset only on cold reset. All Caliptra fuses must be RO for Caliptra FW. uController should not be able to access certain fuse registers as defined in the register rdl and others are non-accessible
See above line
 As decribed in col.B
Middle of FW update flow happening and a surpise warm reset happens; **Expectation is we re-request the FW**
Locked keys must remain intact. No keys must be reset by HW. Must validate clear functionality as a part of KV val itself
 Same as PCRs in terms of registers behavior
Check to confirm that contents loaded to ICCM/DCCM match input FW image

Description
Security state changes from non-debug to debug mode and security assets should be cleared; should be able REINIT the cryptos (using FW) before JTAG open is set; JTAG open by uController itself falls into JTAG functionality validation
Warm reset: Locked keys must remain intact on warm reset. No keys must be reset by HW. Must validate clear functionality as a part of KV val itself Cold reset: Everything is cleared
Warm reset: Locked PCRs must remain intact on warm reset. No PCRs must be reset by HW (?). Unlocked PCRs clear behavior part of PCR val itself or thru ucontroller tests Cold reset: Everything is cleared
Warm Reset: See above but should try to change the security state while under warm reset and then deassert the reset and then check behavior
Generic wires causing clock ungating; Global clock disable; Enable only on the SOC register write; uController doing idle entry and exit autonomously; Entry is ALWAYS uC controlled if the SOC allows clk gating
Boot flow as documetated in the arch spec
Basic assertion & deassertion while various flows are in progress
REINIT capability of the cryptos from HW POV (FW is expected to do this when security state transitioning happens)
GENERIC input wires being able to trigger interrupt on toggling
Arch, Fuse register path thru APB should be independent of mailbox (Same with TRNG)
see Col.B - cross with FW update reset too
See col.B
See col.B
See col.B
See col.B
TRNG REQ API should still be the same on FW update reset
JTAG path for uController debug (able to inject any uController instruction; ability to read any of the AHB-lite registers that uC FW has permission to access) Unlock of JTAG from secured to debug should be ONLY on a FW write once uC is brought out of reset JTAG access to mailbox protocol and mailbox registers BootFSM break functionality to stop the BootFSM before uC is brought up on a warm or cold reset; In this condition, TAP debug, manuf as well as prod register should be writeable....FW will honor the register based on the security state. TAP GO command allows progress to bring uC out of reset after BootFSM break.
Esp. for AHB error return status

Description
Being able to set and unset the timers; WD timer functionality to trigger FATAL
Lock ICCM and unlock on a FW update reset, Warm reset, Cold reset Access ICCM just after ICCM lock reset but before reset assertion (cannot happen due to how code is, but still stress check)
uC out of reset -> exec ROM -> mailbox to ICCM and DCCM -> Execute -> again copy to ICCM and again execute
uC out of reset -> exec ROM -> mailbox to ICCM and DCCM -> Execute -> again copy to ICCM and again execute
uC out of reset -> exec ROM -> mailbox to ICCM and DCCM -> Execute -> again copy to ICCM and again execute
uC out of reset -> exec ROM -> mailbox to ICCM and DCCM -> Execute -> again copy to ICCM and again execute
uC out of reset -> exec ROM -> mailbox to ICCM and DCCM -> Execute -> again copy to ICCM and again execute
As Col B says
As Col B says
As Col B says
As Col B says
Basic val to access for ROM for data in addition to IFU path Conflicts b/w IFU B2B and LSU B2B access and stalling each other; LSU B2B access along with peripheral access
NOTE: Some error interrupts unimplemented
Accesses outside of peripheral space; accesses to ICCM/DCCM outside of the allocated space; access to mailbox SRAM outside of allocation space; accessing cryptos with wrong configuration/protocol

Description
As col.B says
As col.B says

Tests & Checkers	Functional Coverage %
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	
100%	

Tests & Checkers	Functional Coverage %
1oo%	
1oo%	
1oo%	
1oo%	
1oo%	
1oo%	
1oo%	
1oo%	
1oo%	
1oo%	
6o%	
0p8	
1oo%	
0p8	
0p8	
50%	
1oo%	
1oo%	
1oo%	
9o%	
5o%	

Tests & Checkers	Functional Coverage %
1oo%	
1oo%	
1oo%	
0p8	
0p8	
0p8	
0p8	
0p8	
1oo%	
95%	
1oo%	
1oo%	
1oo%	
Need to cross check and mark it	
1oo%	

Tests & Checkers	Functional Coverage %
1oo%	
1oo%	
1oo%	
1oo%	
5o%	
1oo%	
1oo%	
1oo%	
50%	

Tests & Checkers	Functional Coverage %
1oo%	
50%	
1oo%	
1oo%	
1oo%	
1oo%	
1oo%	
1oo%	
1oo%	
9o%	
80%	
80%	
1oo%	
1oo%	
1oo%	

Tests & Checkers	Functional Coverage %
100%	
100%	
100%	
100%	

Comments
To be included in top level

Comments

Comments

Comments

Comments
We need to check if there is any stress val to be additionally done here post 0p8

Comments
Same as subtractive decoding path
Not production, primarily for debug