**DAM Validation Plan**

1. Introduction

The DAM IP is located before the MC in SoC.

The DAM is use for data manipulating.

It manipulates and implement algorithms on data directed to it. The algorithms are encrypting (and decrypting), ECC, parity calculations, and more.

It passes transactions of different types and protocols from different sources towards the memory.

There is an option to implement multi-instance of it.

1. Tests
   1. *DAM ECC*
      1. **Direct – full write and read to non-protected addr**

Issue Full-Write access to a non-protected address followed by read access

Self-Check - compare read data to written data fail test if no traffic (idle traffic)

* + 1. **Direct – partial write and read to non-protected addr**

Issue Partial-Write access to a non-protected address followed by read access

Self-Check - compare read data to written data fail test if no traffic (idle traffic)

* + 1. **Direct – full write and read to protected addr**

Issue Full-Write access to a Protected address followed by read access

Self-Check - compare read data to written data fail test if no traffic (idle traffic)

* + 1. **Direct – partial write and read to protected addr**

Issue Partial-Write access to a Protected address followed by read access

Self-Check - compare read data to written data fail test if no traffic (idle traffic)

* + 1. **Direct – full write and read to protected addr with single error**

Configure single-error injection to the relevant address, issue Full-Write access of 32B request to a Protected address followed by read access

Self-Check - Compare read data to written data, verify receiving single error detection & correction indication, fail test if no traffic (idle traffic)

* + 1. **Direct – partial write and read to protected addr with single error**

Configure single-error injection to the relevant address, issue Partial-Write access to a Protected address followed by read access

Self-Check - Compare read data to written data, verify receiving single error detection & correction indication, fail test if no traffic (idle traffic)

* + 1. **Direct – full write and read to protected addr with double error**

Configure double-error injection to the relevant address, issue Full-Write access to a protected address followed by read access

Self-Check - Compare read data to written data, verify receiving double error detection fail test if no traffic (idle traffic)

* + 1. **Direct – partial write and read to protected addr with double error**

Configure double -error injection to the relevant address, issue Partial-Write access to a Protected address followed by read access

Self-Check - Compare read data to written data, verify receiving double error detection

Fail test if no traffic (idle traffic)

* + 1. **Direct – two errors**

Creating two consecutive errors :

1. second error before first was cleared
2. second error AFTER first was cleared
   * 1. **Scenario Details –**

* Write to several different addresses
* inject errors on those addresses
* Back-to-Back read from those addresses to get back-to-back errors.
* check both correctable and non-correctable errors
  + 1. **Run PM**

Basic Power Mode scenario while ECC is enabled, and traffic is on.

* + 1. **Stress test**

Load the system with many read/write/write partial transactions in parallel, check that the system is working properly.

* 1. *DAM Encryption*

DAM encryption uses 3 types of ranges addresses:

1. Fully protected Range (Addr X 🡪 Addr Y)

* protected by 3 types of protection
  + PROTECTION\_TYPE\_1
  + PROTECTION\_TYPE\_2
  + PROTECTION\_TYPE\_3
* issues ERROR\_TYPE\_1

1. Partially Protected Range (Addr A 🡪 Addr G)

* 1 type of protection
  + PROTECTION\_TYPE\_1
* issues ERROR\_TYPE\_2

1. Excluded Range – Protected only in a specific use case

* 1 type of protection
  + PROTECTION\_TYPE\_1
    1. ***Direct - Full Write and Read to Partially Protected addr***

Issue Full-Write access to a partial-protected address followed by read access

Self-Check - compare read data to written data fail test if no traffic (idle traffic)

* + 1. ***Direct - Full Write and Read to Fully Protected addr***

Issue Full-Write access to a partial-protected address followed by read access Self-Check - compare read data to written data fail test if no traffic (idle traffic)

* + 1. ***Direct - Partial Write and Read to Partially Protected addr***
    2. ***Direct - Partial Write and Read to Fully Protected address***
    3. ***Direct – ERROR\_TYPE\_1 test***

Issue Write followed by read and inject ERROR\_TYPE\_1.

Use all 3 types of Ranges

* + 1. ***Direct – ERROR\_TYPE\_2 test***

Issue Write to the Fully Protected Range followed by read and inject ERROR\_TYPE\_2.

* + 1. ***Direct – ERROR\_TYPE\_3 test***

Issue Write to the Fully Protected Range followed by read and inject ERROR\_TYPE\_2.

* + 1. ***Direct – Multiple instances test with ERROR\_TYPE\_1***

Issue Write followed by read and inject ERROR\_TYPE\_1 for Fully Protected address.

Run the test when both DAMs are enabled

* + 1. ***Direct - Excluded Range test***

Send traffic to both excluded and non-excluded areas, make sure that the excluded are not being encrypted, while the regular ones are.

* + 1. ***Stress traffic***

Load the system with many read/write/write partial transactions in parallel. Randomize data, address, DAM Encryption configurations.

* + 1. ***Keys basic write read test***

A simple sanity test checking that all keys are configured correctly and can handle all transaction types with encryption and decryption

* + 1. ***B2B (Back-to-Back) transactions-***

Send multiple B2B transactions, use only Core source, randomize aspects of address ranges, keys, etc.

* + 1. ***Ordering test – WAR, RAW***

Run write after read, and read after write, to same and different addresses.

To all address ranges

* + 1. ***Non-Dependent RD & WR at same cycle***

Issue read and write at the same cycle to different address

* 1. *DAM Encryption and ECC*

Run with encryption **and** ECC enabled.

* + 1. **Write and read to ECC protected address**

Issue Write access to a Protected address followed by read access

* + 1. **Write and read to Encryption protected address**

Issue Write access to a Protected address followed by read access

To all Encryption ranges.

* + 1. **Write and read to non-protected address**

Issue Write access to a non-protected address followed by read access

To ECC non-protected range and to Encryption excluded range.

* 1. *Security*

Validate DAM security

* 1. *Random Tests*

Random tests are random in both Configuration and Traffic.

* + 1. **Random Configuration**
* Single/Multi instances
  + Run with only 1 DAM enabled
  + Run with both DAMs enabled
* DAM ECC and Encryption enablement

1. Only ECC Enabled
2. Only Encryption Enabled
3. Both ECC and Encryption Enabled

* Protected Range and Size

Configure Protected Ranges configuration registers to the following scenarios:

1. All memory is protected
2. Random number of address spaces ranges are protected,
3. Different sizes of ranges

* ECC Caching Enabled/Disabled – ECC internal caching
* Encryption types (3 types)
  + 1. **Random Traffic**

Generate random number of accesses, randomize the following:

* Addresses
  1. Addresses for the random tests will be randomized with some probability to use the same address again (Protected addresses and non-protected addresses)
  2. Randomize from different ranges
     1. ECC Protected/Non-Protected
     2. Encryption Ranges
* Access type (opcode)
* Read/Write/Write partial
* Access Source (Core, ACC1,2,3…)
* Randomly inject Errors – ECC error and/ or encrypting errors
  1. *Global*
     1. **Registers Test**
* Read all of DAM registers, validate reset value.
* Write to all DAM writable registers and read

1. Checkers
   1. *Traffic Checker*

Implement traffic checker in PP (post processing). Checker should validate that when DAM is enabled traffic is directed through DAM (ECC and/or Encryption according to which is enabled)

* 1. *ECC checker*
* Check ECC indication for protected address – by monitoring an internal DAM signal.
* Check correctable error report –

1. By reading DAM status register
2. By PP – validating that the error was reported in SoC

* Check non-correctable error report

1. By reading DAM status register
2. By PP – validating that the error was reported in SoC
   1. *Encryption*
      1. **Encrypted/Decrypted Checker**

By monitoring DAM Encryption internal signal, validate that data was encrypted and decrypted (for relevant addresses)

* + 1. **Post ERROR\_TYPE\_1 Checker – stop traffic**

After ERROR\_TYPE\_1 error, READ from Fully Protected range should be stuck (only in the specific DAM instance) ALL OTHER ACCESSES to all other ranges should continue.

* + 1. **Encryption Error Checker -reported**

Validate that Encryption Error is reported in Registers

and sent to SOC.

* 1. Existing checkers usage

Integrate and activate, at the relevant tests, the following SoC checkers:

* + 1. **E2E Checker**

End to End checker, checks that read receives the correct data

* + 1. **Memory interface protocol checker**
    2. **Main fabric Data protocol checker**
    3. **Control Registers protocol checker**

1. Coverage
   1. Addresses Coverage

Cover groups will contain several address ranges to make sure that address distribution in the simulations is wide enough.

* ECC Non protected addresses
* ECC Protected addresses – from all protected ranges
* Encryption Fully Protected Range
* Encryption Partially Protected Range
* Encryption Excluded Range
  1. Access Type

Cover groups in PP, to makes sure that DAM random tests are randomizing all access types supported by DAM

* 1. *Access Direction*

Cover Read/Write/Write Partial

* 1. *Encryption coverage*
     1. **Keys and key status**

16 keys, 3 encryption status.

* + 1. **Metadata Distribution**

MAC is metadata used in PRMRR to manage caching

* + 1. **Cache – hit/miss, fill/evict**
  1. *ECC coverage*
     1. **ECC Error type – Correctable/Non correctable**

Cover both correctable and Non-Correctable Error types, cross with protected ranges to assure that we get both types of Error to each range of protected address.

* + 1. **Cache Utilization**

The internal cache utilization is a measure of the variety of addresses randomized – For example if we get cache “hits” it means that addresses were repeated at a specific time interval.