INHERENTLY SECURE PROCESSING

Hive

The DOVER Edge

André DeHon andre@acm.org

Eli Boling, Rishiyur Nikhil, Darius Rad,

Julie Schwarz, Niraj Sharma, Joseph Stoy,

Greg Sullivan, Andrew Sutherland

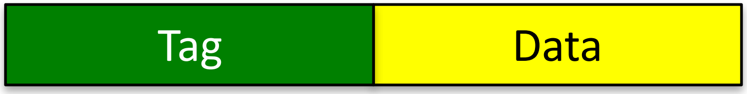
Draper

7/13/2016

Question

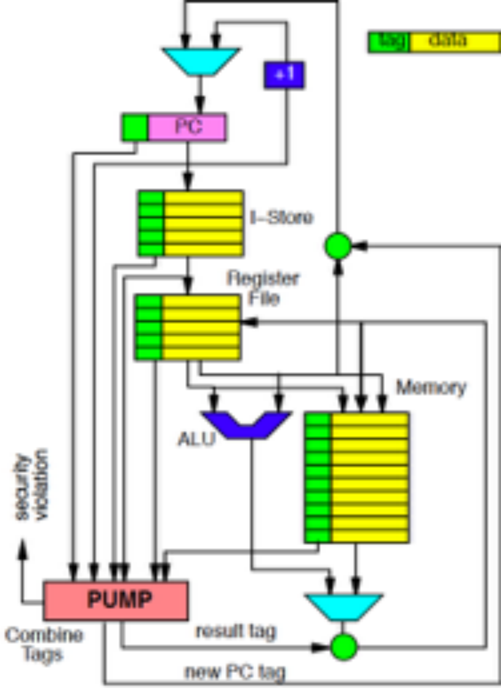
- How handle the “edge” of a tagged system, where necessary to deal with
  - I/O to the untagged world
  - Tag-oblivious, legacy devices and drivers
  - Tag-oblivious, Direct-Memory Access devices

DOVER=RISC-V + PUMP Software-Defined Metadata Processing (SDMP)

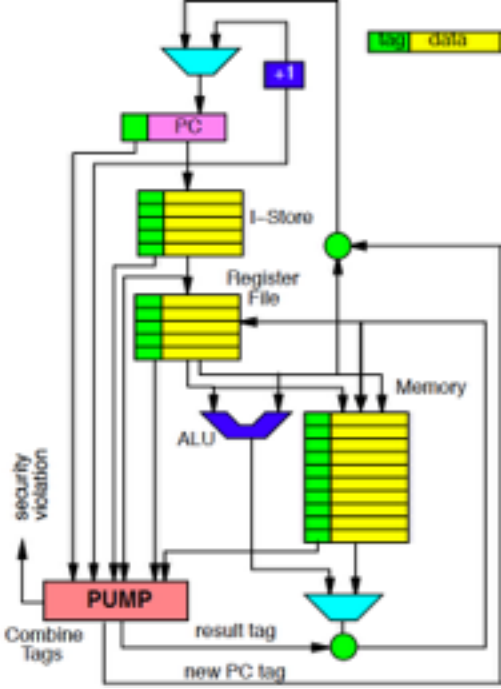


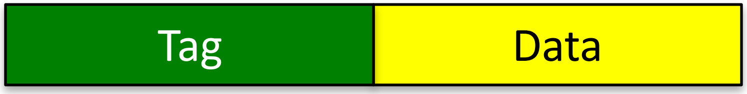
Tag

Data



DOVER=RISC-V + PUMP
Software-Defined Metadata Processing (SDMP)



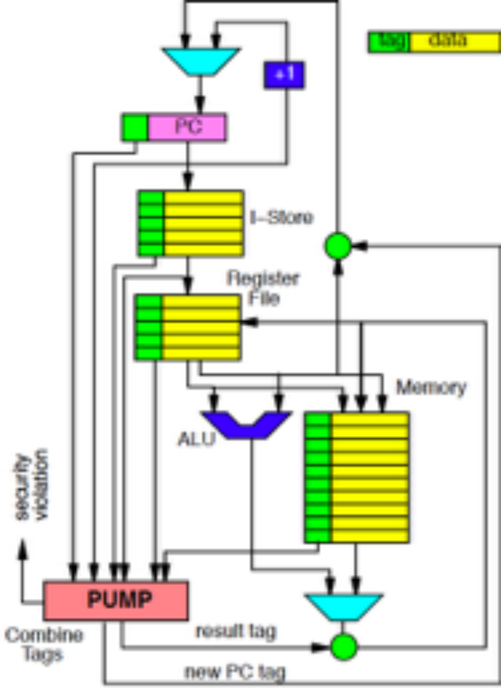


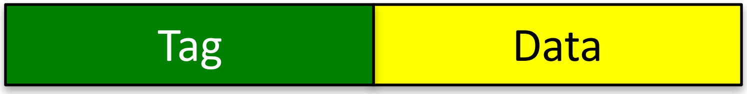
Tag

Data

- Add metadata tag to every word
  - Allows us to know information about data word
    - Type, where-came-from, buffer, secrecy

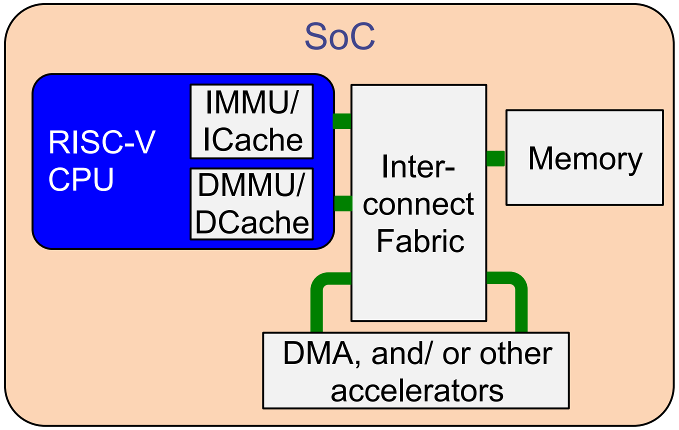
DOVER=RISC-V + PUMP Software-Defined Metadata Processing (SDMP)





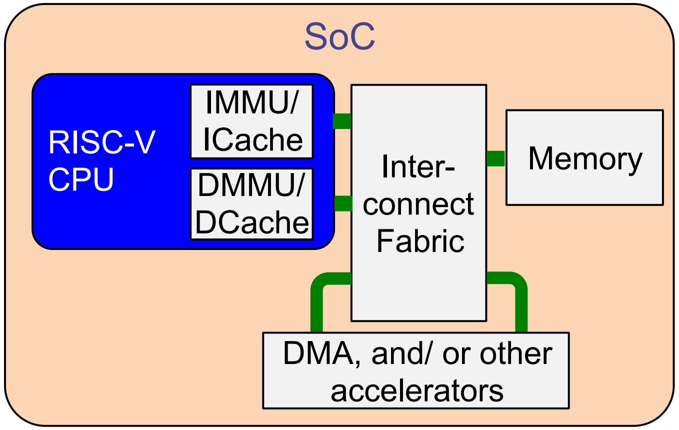
- Add metadata tag to every word
  - Allows us to know information about data word
    - Type, where-came-from, buffer, secrecy
- Mediate every instruction by checking tags
  - Allows enforcement of safety/security policies
    - Memory safety, CFI, integrity, information flow

Challenge


SoC
RISC-V CPU
IMMU/ ICache
DMMU/ DCache
Inter-connect Fabric
Memory
DMA, and/ or other accelerators

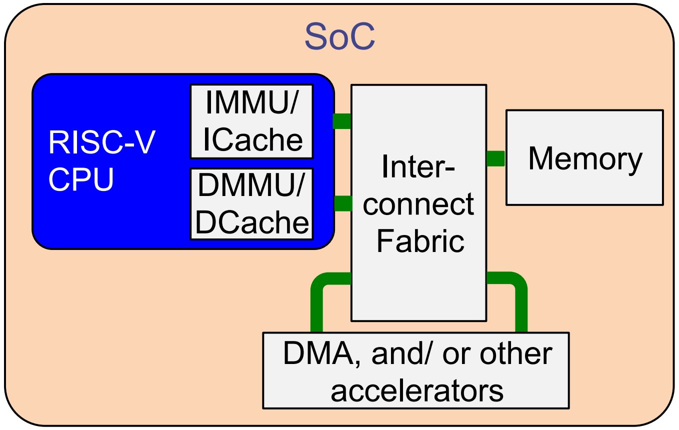
Challenge

- Processor not only entity can write to memory.



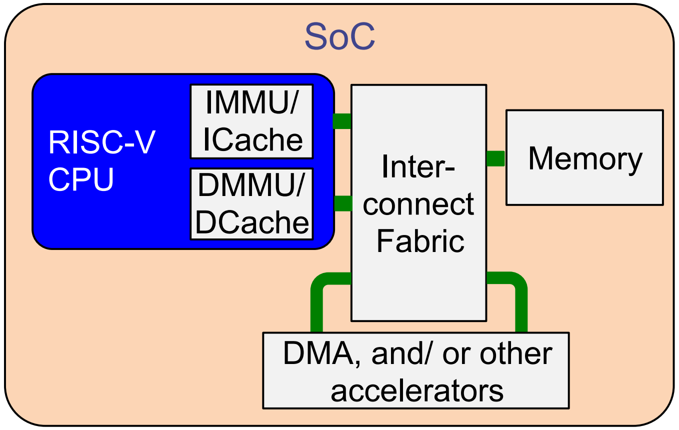
Challenge

- Processor not only entity can write to memory.
- DMA engines for performance
  - E.g., networking, graphics

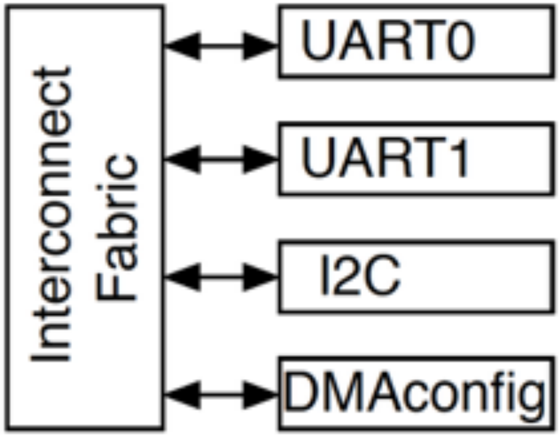


Challenge

- Processor not only entity can write to memory.
- DMA engines for performance
  - E.g., networking, graphics
- Danger: undermine security policies
  - Write over OS code
  - Read secret data

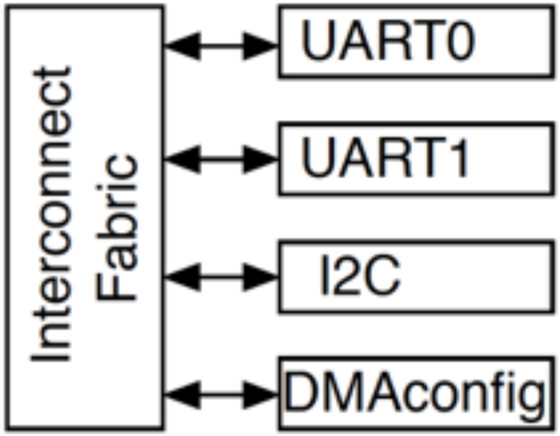


Fine-Grained Slave Device Control



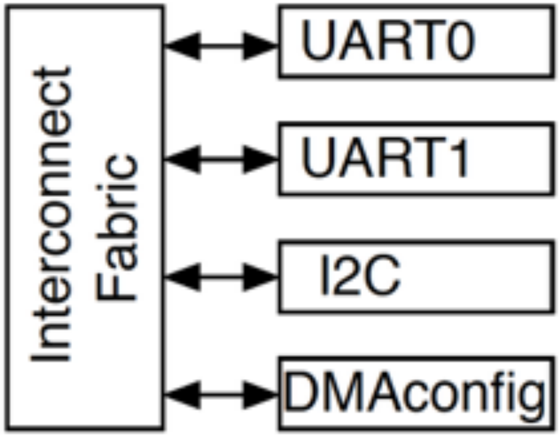
Fine-Grained Slave Device Control

- Tag memory-mapped devices
  - Use to write rules to control access



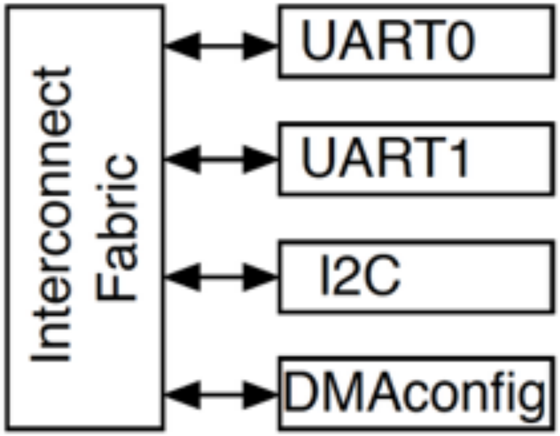
Fine-Grained Slave Device Control

- Tag memory-mapped devices
  - Use to write rules to control access
- Control “who” can write to which devices
  - Which code, which task
  - not just kernel/user



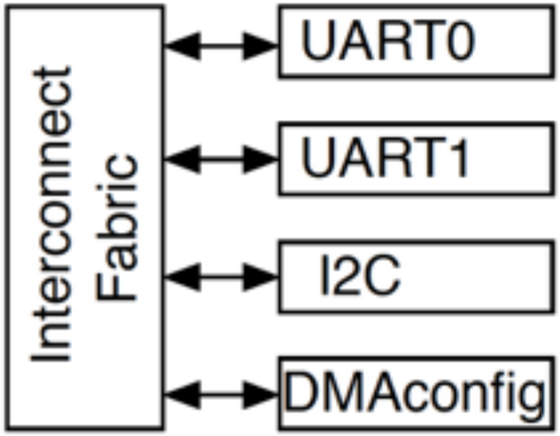
Fine-Grained Slave Device Control

- Tag memory-mapped devices
  - Use to write rules to control access
- Control “who” can write to which devices
  - Which code, which task
  - not just kernel/user
- Give configuration control to particular drivers, without giving the driver control to all devices or other privileges

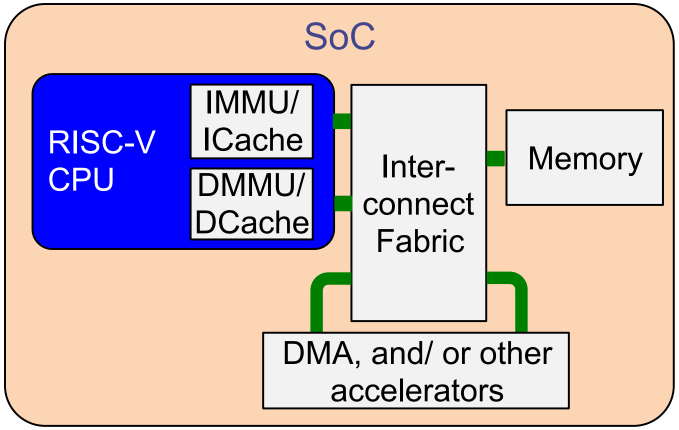


Fine-Grained Slave Device Control

- Tag memory-mapped devices
  - Use to write rules to control access
- Control "who" can write to which devices
  - Which code, which task
  - not just kernel/user
- Give configuration control to particular drivers, without giving the driver control to all devices or other privileges
- Mark memory-mapped slave data as untrusted
  - and by source

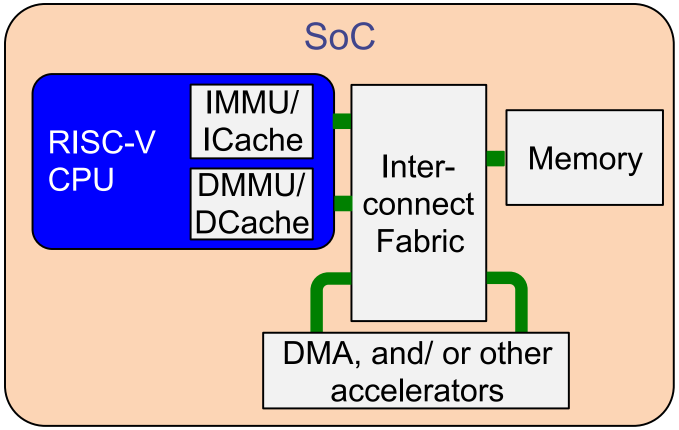


DMA IO Policies



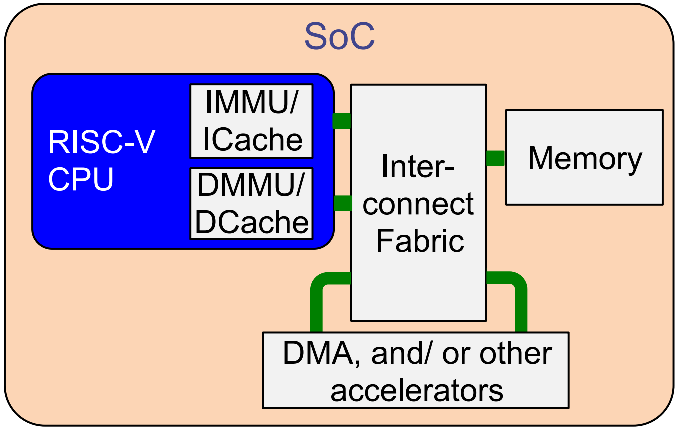
DMA IO Policies

- Containment -- who's allowed to read/write buffer



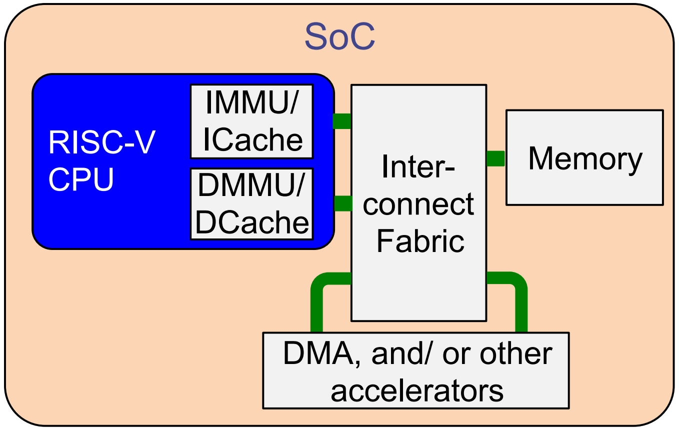
DMA IO Policies

- Containment -- who's allowed to read/write buffer
- Integrity – mark incoming data as untrusted
  - At least until validated



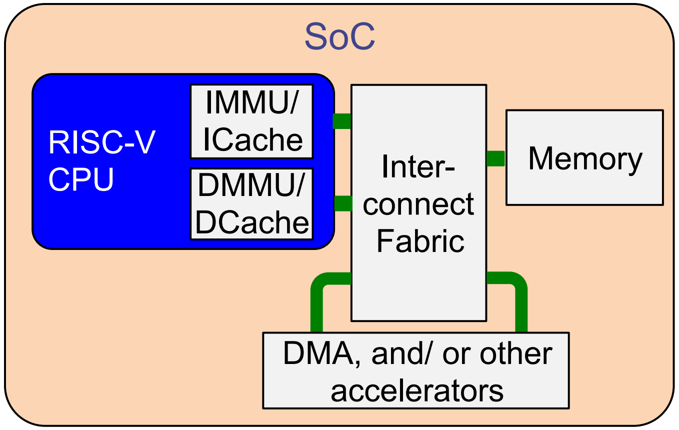
DMA IO Policies

- Containment -- who's allowed to read/write buffer
- Integrity – mark incoming data as untrusted
  - At least until validated
- Secrecy -- not allow classified data to exit system
  - Without proper encryption

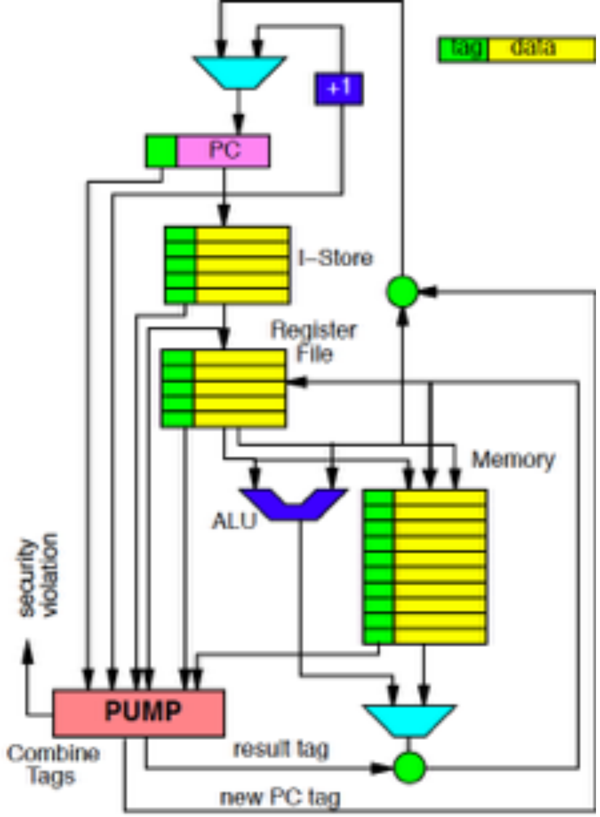


DMA IO Policies

- Containment -- who's allowed to read/write buffer
- Integrity – mark incoming data as untrusted
  - At least until validated
- Secrecy -- not allow classified data to exit system
  - Without proper encryption
- Data presence/synchronization
  - only grab valid data
  - only overwrite empty buffer
    - and mark as non-empty when write

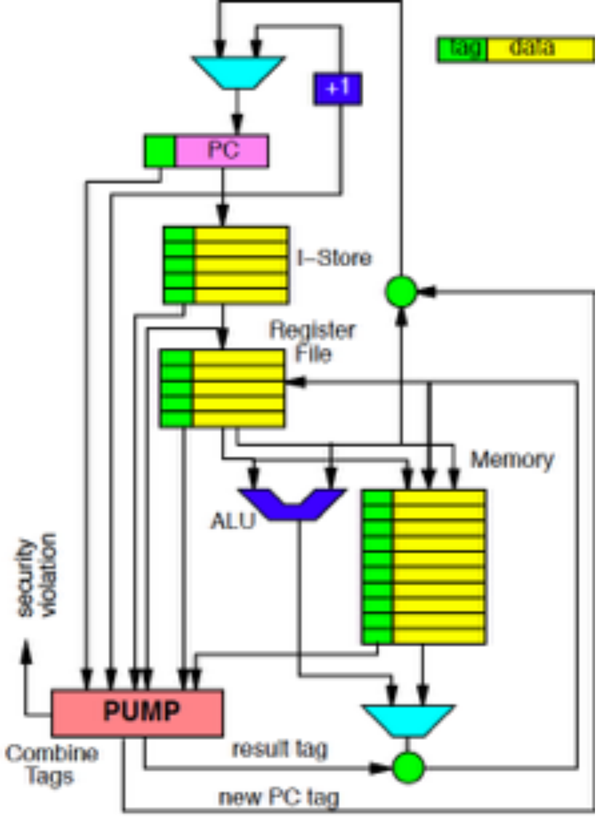


SDMP Policy Function

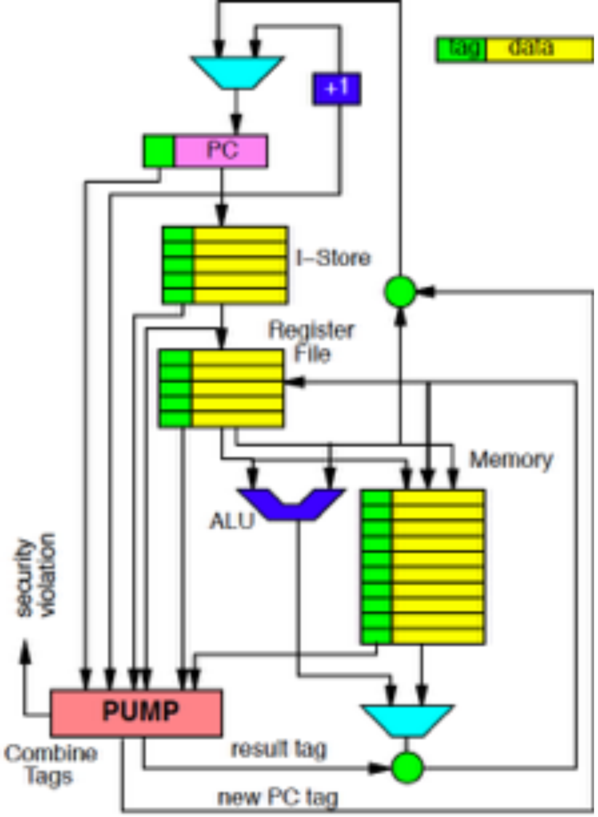


SDMP Policy Function

- Every word has metadata

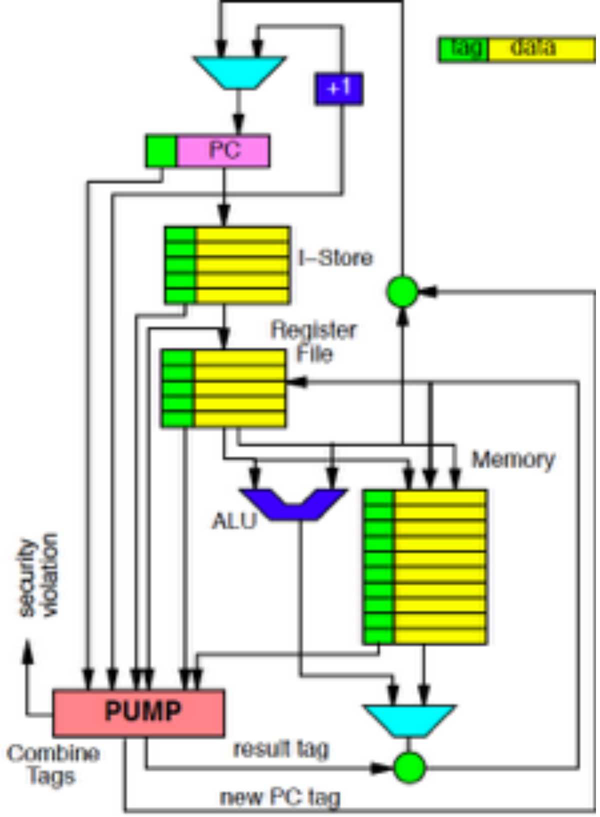


SDMP Policy Function



- Every word has metadata
- PUMP is a function from:
  - Opcode, PC_{tag}, $Instr_{tag}$, $RS1_{tag}$, $RS2_{tag}$, MR_{tag}

SDMP Policy Function



- Every word has metadata
- PUMP is a function from:
  - Opcode, PC_{tag}, $Instr_{tag}$, $RS1_{tag}$, $RS2_{tag}$, MR_{tag}
- To:
  - Allowed?
  - PC_{tag}
  - $Result_{tag}$ (RD, memory result)

Extend for IO

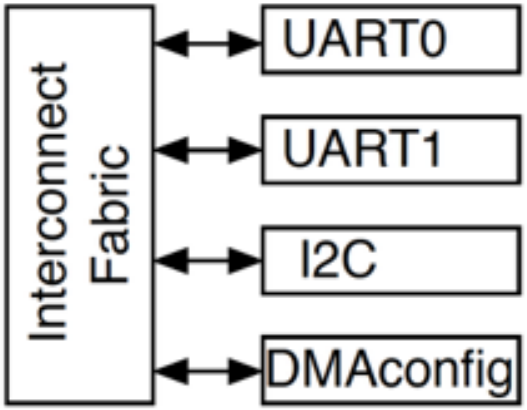
Extend for IO

- Same function:

  Opcode, PC_{tag}, $Instr_{tag}$, $RS1_{tag}$, $RS2_{tag}$, MR_{tag}

  → Allowed?, PC_{tag}, $Result_{tag}$

Extend for IO

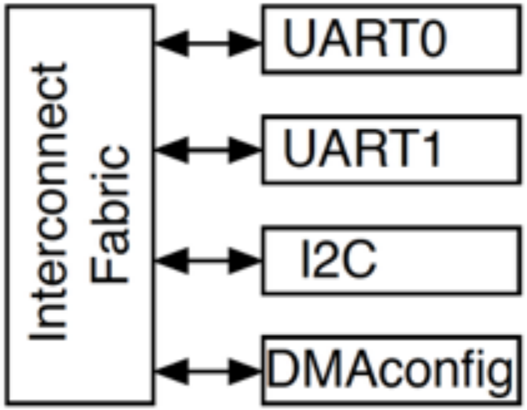


- Same function:

  Opcode, PC_{tag}, $Instr_{tag}$, $RS1_{tag}$, $RS2_{tag}$, MR_{tag}

  → Allowed?, PC_{tag}, $Result_{tag}$
- Slave devices present MR_{tag}

Extend for IO

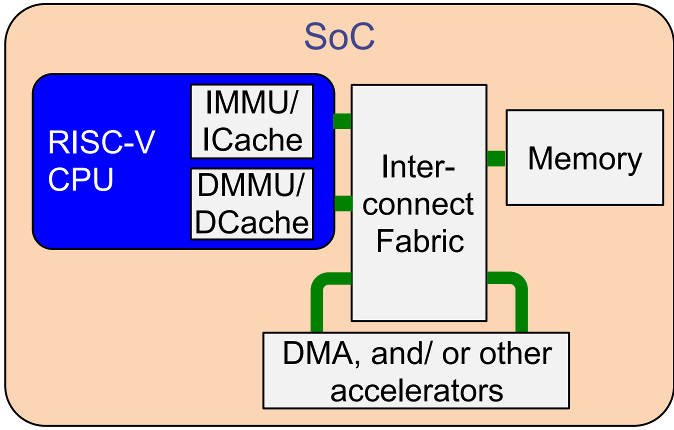


- Same function:

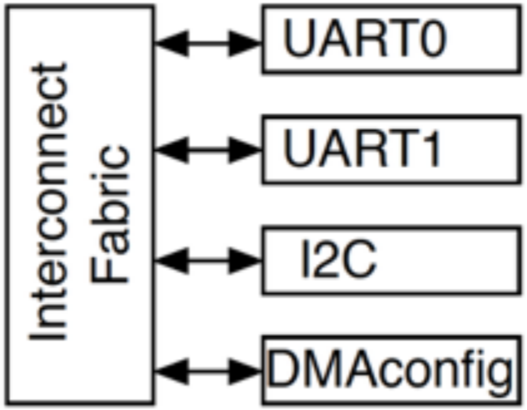
  Opcode, PC_{tag}, $Instr_{tag}$, $RS1_{tag}$, $RS2_{tag}$, MR_{tag}

  → Allowed?, PC_{tag}, $Result_{tag}$

- Slave devices present MR_{tag}
- Opcode: {DMA-load,
  DMA-store}



Extend for IO

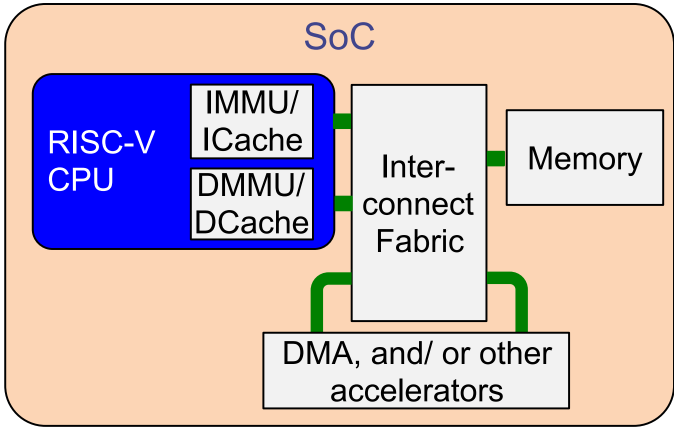


- Same function:

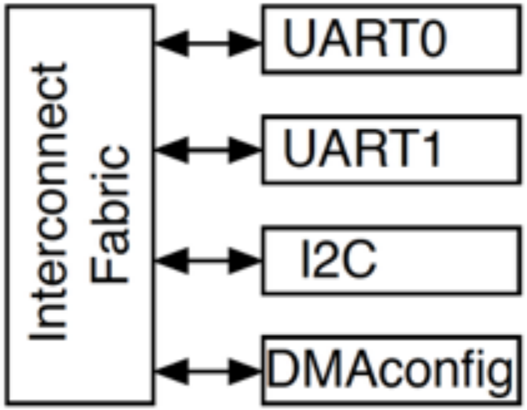
  Opcode, PC_{tag}, $Instr_{tag}$, $RS1_{tag}$, $RS2_{tag}$, MR_{tag}

  → Allowed?, PC_{tag}, $Result_{tag}$

- Slave devices present MR_{tag}
- Opcode: {DMA-load, DMA-store}
- PC_{tag} – state of DMA



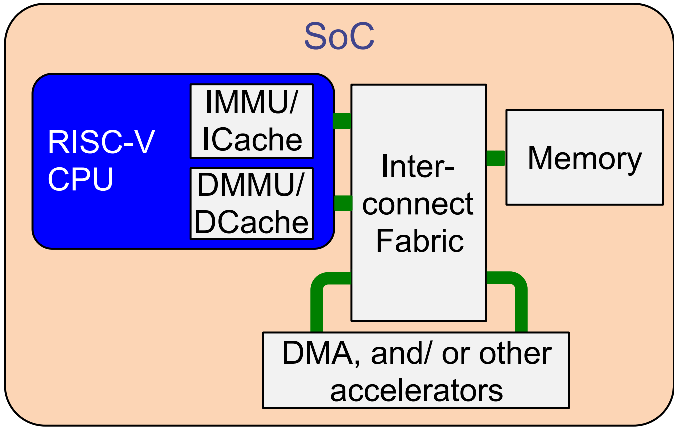
Extend for IO



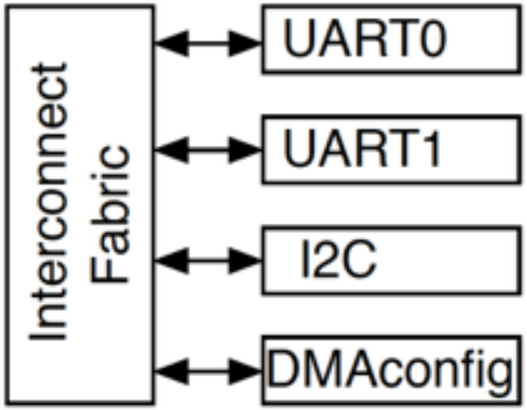
- Same function:

  Opcode, PC_{tag}, $Instr_{tag}$, $RS1_{tag}$, $RS2_{tag}$, MR_{tag}

  → Allowed?, PC_{tag}, $Result_{tag}$
- Slave devices present MR_{tag}
- Opcode: {DMA-load, DMA-store}
- PC_{tag} – state of DMA
- $Instr_{tag}$ – identify DMA source



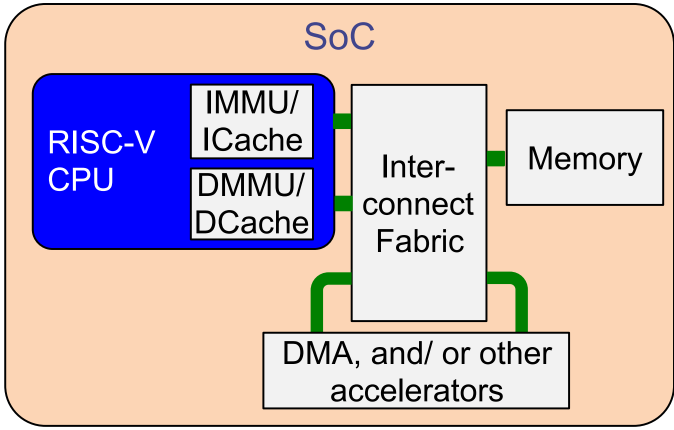
Extend for IO



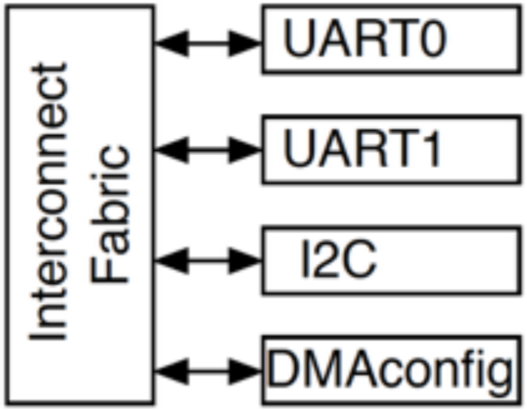
- Same function:

  Opcode, PC_{tag}, $Instr_{tag}$, $RS1_{tag}$, $RS2_{tag}$, MR_{tag}

  → Allowed?, PC_{tag}, $Result_{tag}$
- Slave devices present MR_{tag}
- Opcode: {DMA-load, DMA-store}
- PC_{tag} – state of DMA
- $Instr_{tag}$ – identify DMA source
- $RS1_{tag}$, $RS2_{tag}$ – public, untrusted



Extend for IO

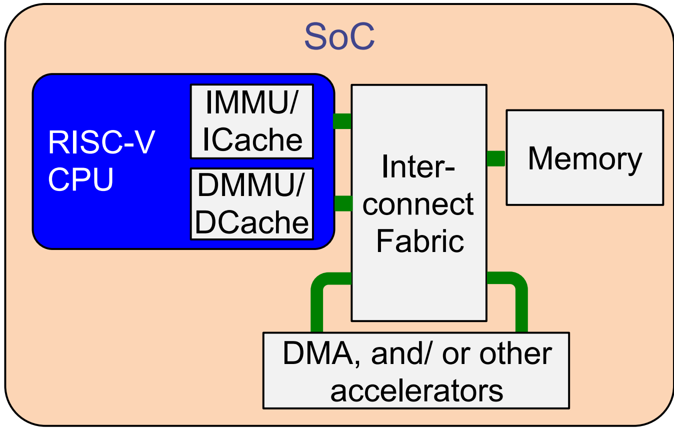


- Same function:

  Opcode, PC_{tag}, $Instr_{tag}$, $RS1_{tag}$, $RS2_{tag}$, MR_{tag}

  → Allowed?, PC_{tag}, $Result_{tag}$

- Slave devices present MR_{tag}
- Opcode: {DMA-load,
  DMA-store}
- PC_{tag} – state of DMA
- $Instr_{tag}$ – identify DMA source
- $RS1_{tag}$, $RS2_{tag}$ – public, untrusted
- MR_{tag}, $Result_{tag}$ – on memory read/written



Misbehavior Models

Misbehavior Models

- Execution on processor → halt offending process

Misbehavior Models

- Execution on processor → halt offending process
- DMA not necessarily associated with a process

Misbehavior Models

- Execution on processor → halt offending process
- DMA not necessarily associated with a process

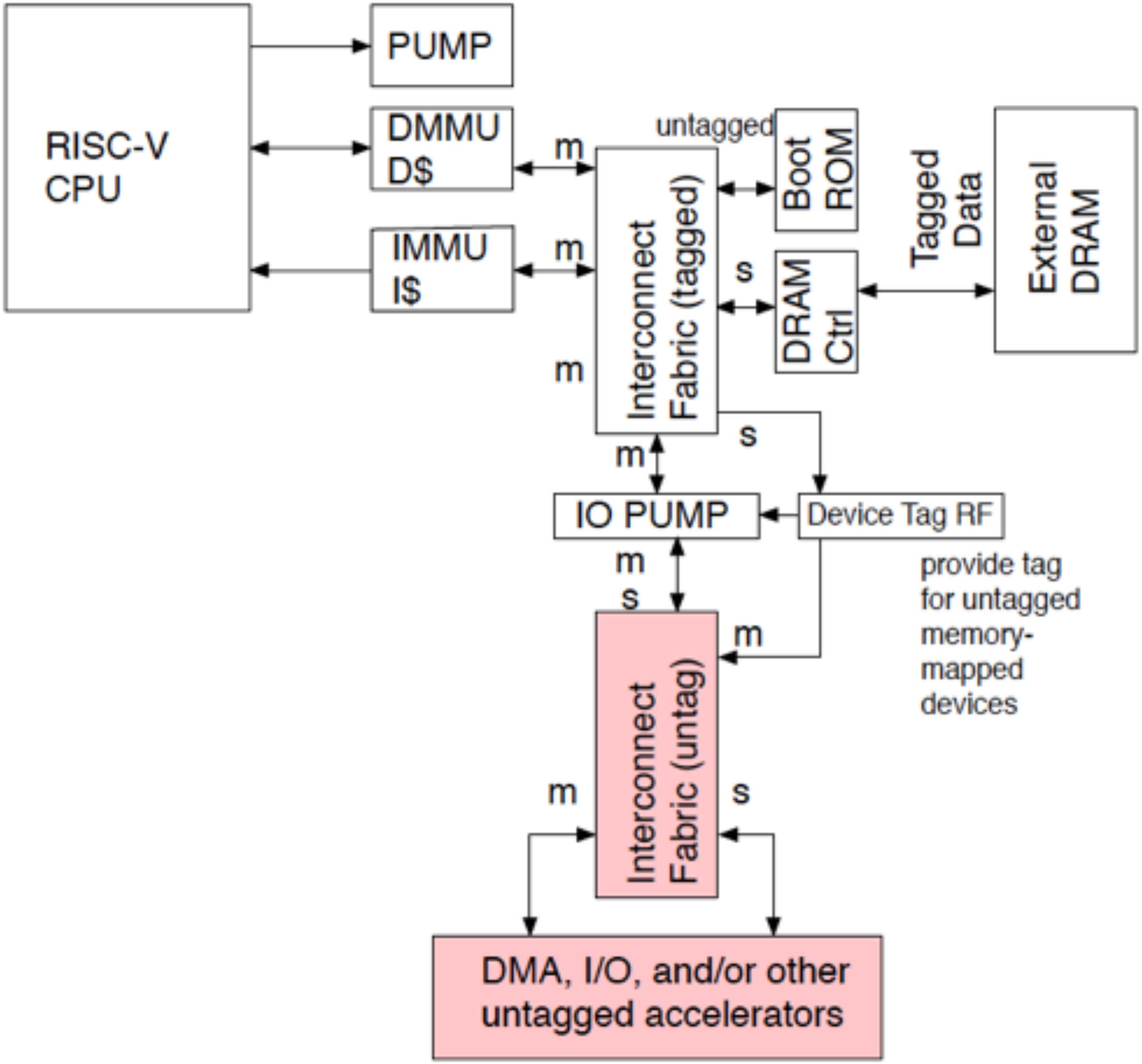
1. Disable offending device
   - Set PC_{tag} for device to DISABLED
   - Discard all operation from DISABLED devices
   - Requires privileged device reset to re-enable

Misbehavior Models

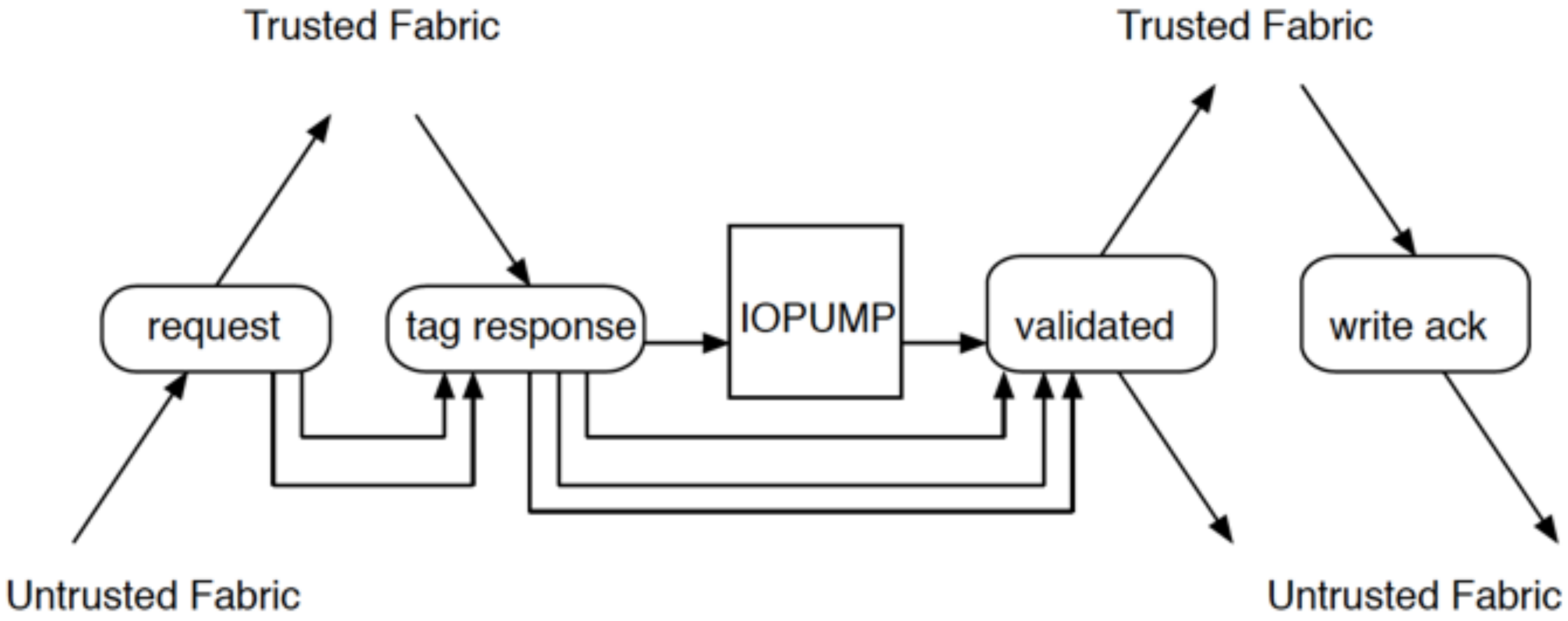
- Execution on processor → halt offending process
- DMA not necessarily associated with a process

1. Disable offending device
   - Set PC_{tag} for device to DISABLED
   - Discard all operation from DISABLED devices
   - Requires privileged device reset to re-enable
2. Discard offending operation
   - Set $Result_{tag}$ to DISCARD
     - Redact data, discard write
   - Allow device to continue

DOVER SoC Architecture



DMA Validation Architecture



RISC-V Integration

RISC-V Integration

- IOPUMP generates interrupt on Rule Miss

RISC-V Integration

- IOPUMP generates interrupt on Rule Miss
- Memory mapped devices
  - IOPUMP
  - Device Register File

RISC-V Integration

- IOPUMP generates interrupt on Rule Miss
- Memory mapped devices
  - IOPUMP
  - Device Register File
- Device tags control access to IOPUMP, DeviceTagRF
  - Setup during boot, before engage PUMP

RISC-V Integration

- IOPUMP generates interrupt on Rule Miss
- Memory mapped devices
  - IOPUMP
  - Device Register File
- Device tags control access to IOPUMP, DeviceTagRF
  - Setup during boot, before engage PUMP
- IOPUMP CSRs similar to PUMP CSRs
  - Rule inputs and outputs

Miss Handler

Miss Handler

- Uses same rule function as Processor / PUMP miss handler

Miss Handler

- Uses same rule function as Processor / PUMP miss handler
- Dispatched from interrupt

Miss Handler

- Uses same rule function as Processor / PUMP miss handler
- Dispatched from interrupt
- Insertion into
  - Memory-mapped IOPUMP CSRs
  - Instead of RISC-V PUMP CSRs

Conclusions

Conclusions

- Complete protection, must manage, the Edge of the tagged metadata system

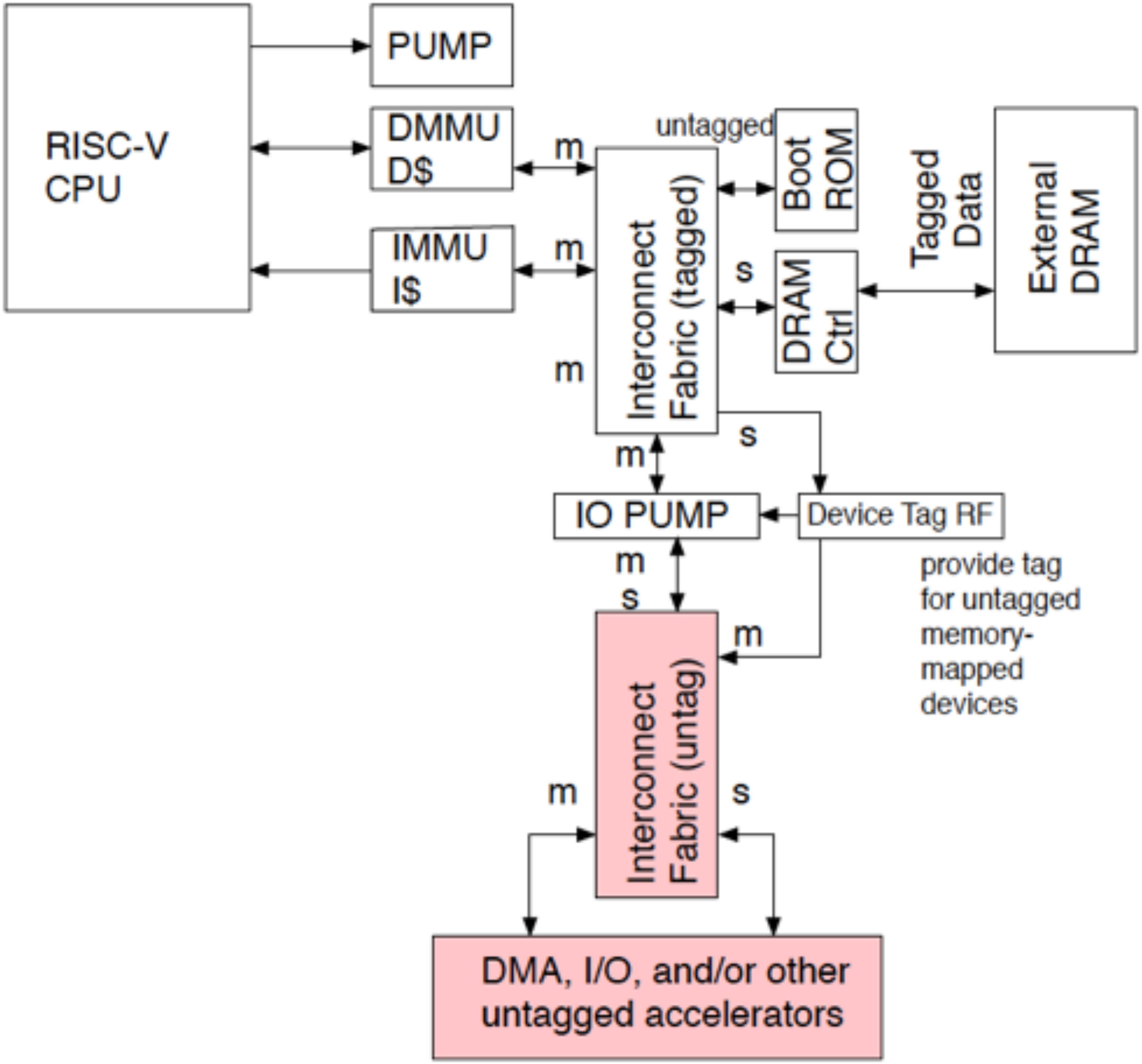
Conclusions

- Complete protection, must manage, the Edge of the tagged metadata system
- Complete, Fine-Grained Mediation
  - For all actors, including DMA IO

Conclusions

- Complete protection, must manage, the Edge of the tagged metadata system
- Complete, Fine-Grained Mediation
  - For all actors, including DMA IO
- Maintain uniform SDMP model to the Edge
  - Slave device tags
  - Mediated DMA

DOVER SoC Architecture



provide tag for untagged memory-mapped devices