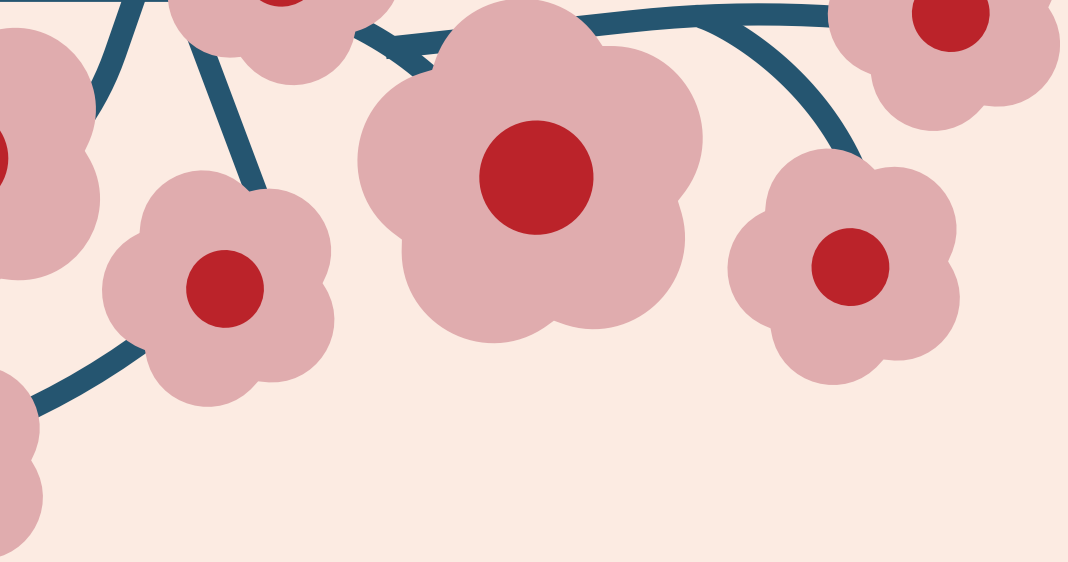
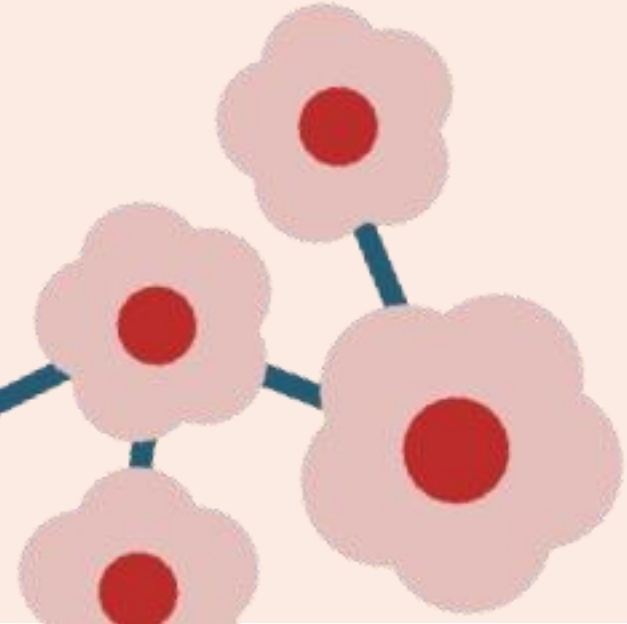
09 April 2025





CHERI

Efficient system-level support for CHERI Capabilities

The Capability Management Unit (CMU)

Mark Hill
Distinguished Engineer & Lead CPU Architect – Codasip

System-level requirements for CHERI

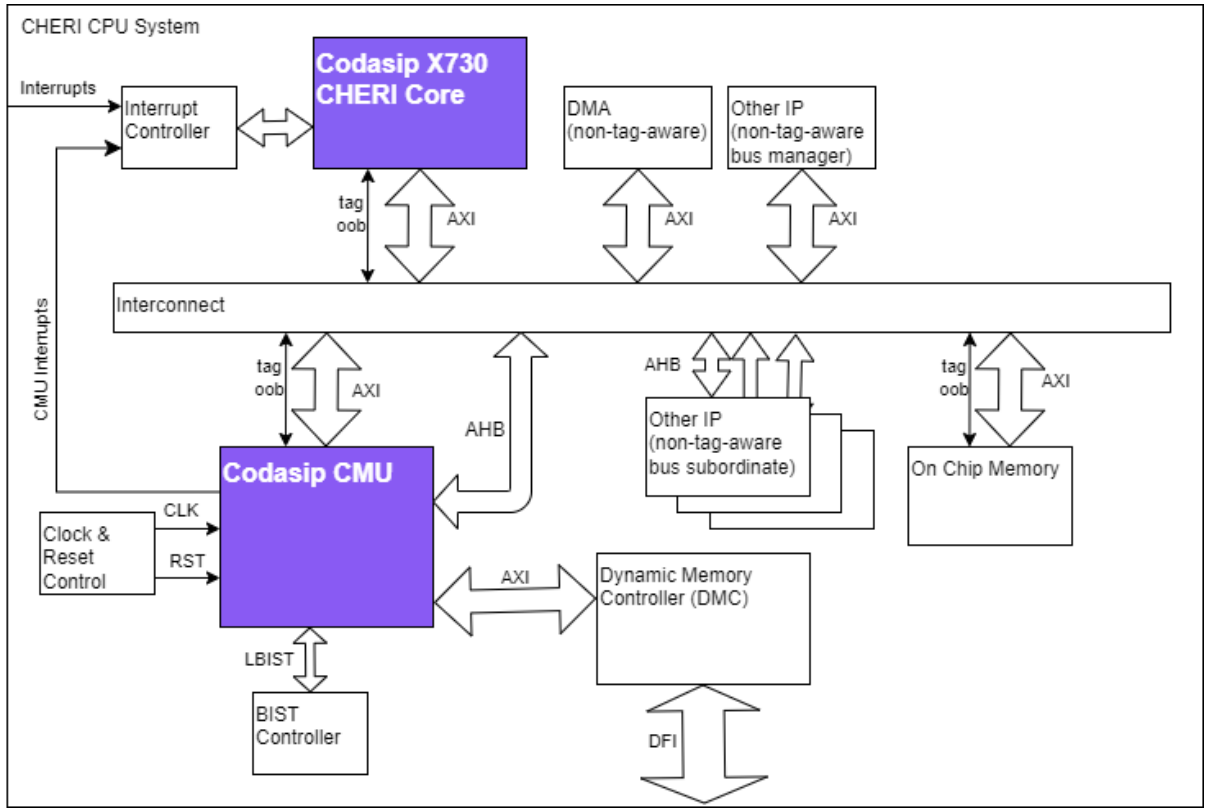
- A system must maintain a tag bit for every memory location that can hold a capability
- Tag and all bytes of the data in a capability must maintain single copy atomicity
  - Sometimes they must be split and stored in separate locations:
    - Integration of tag-aware subsystems (e.g. security islands) into non-tag aware systems
    - Integration with standard, non-tag aware, system IPs:
      - For example: Dynamic Memory Controllers and Last Level Caches.
- The CMU addresses these issues

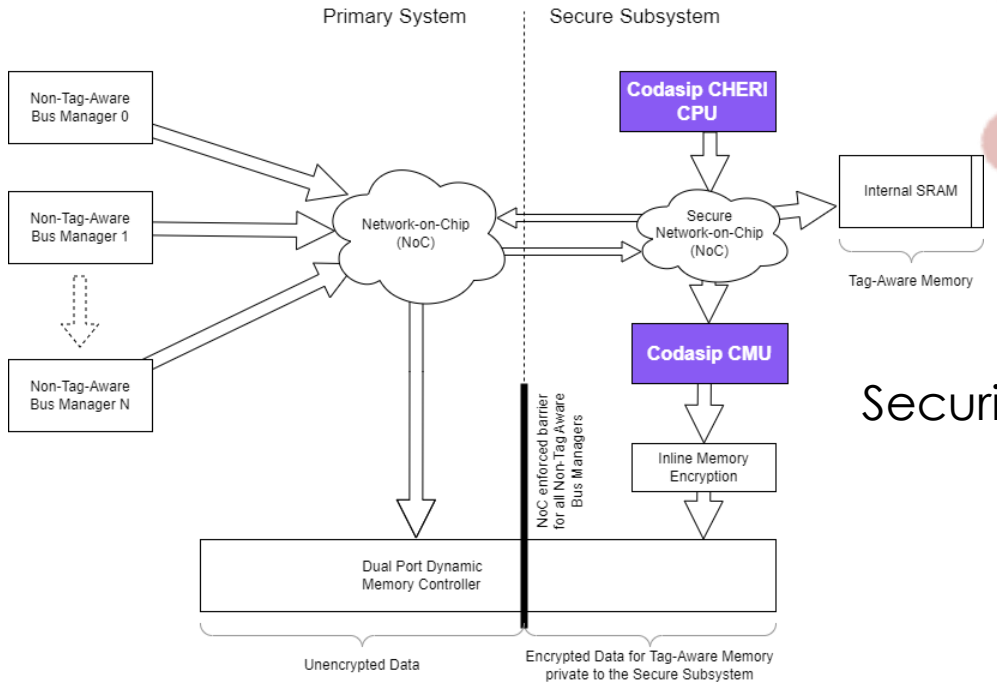
CMU Challenges

- Preserving atomicity of tag and data stored in different locations
  - In a system with multiple bus managers and subordinates
  - Downstream bus fabric and IPs are aggressively reordering transactions (to optimize throughput)
- Minimizing overhead
  - Both in terms of bandwidth and latency
  - To a level comparable to non-CHERI implementations
- Maximizing design re-use
  - by supporting a wide-range of use-cases

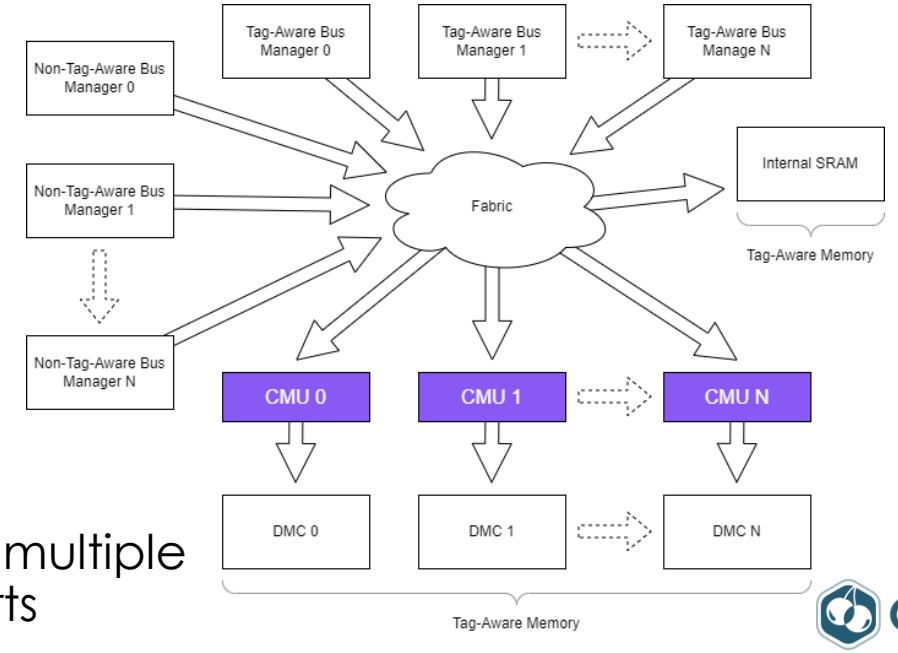
Maximising Design Re-Use
Example deployments

Typical use case, with DDR memory





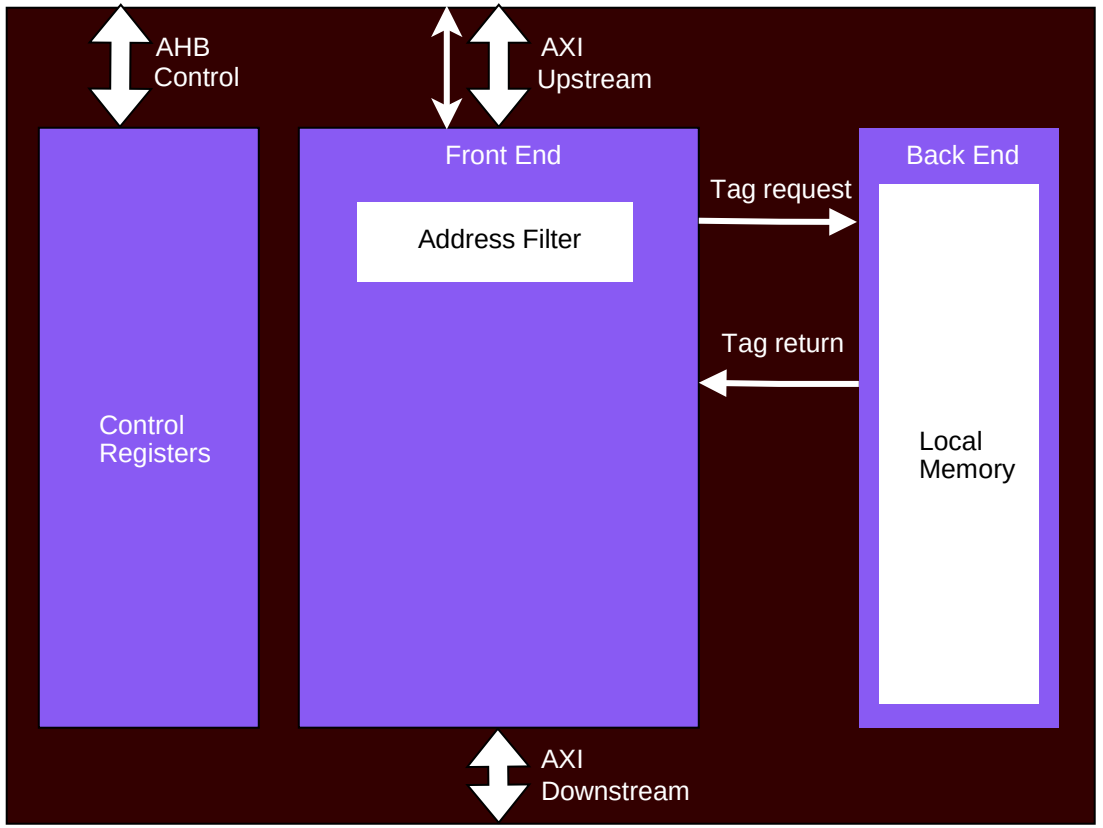
Security Island



HPC/Server with multiple DRAM ports

Tag Stash Mode

- All tag bits held in local memory within the CMU
  - \+ No additional external memory requests ever generated
  - \- Requires 1-bit of local memory for every location in external memory which can hold a capability.
  - \- On-chip memory demand gets too high for larger external memories
    - Realistic limit is about 128Mbytes
- Example use case:
  - Tag bits for a security island requiring some off-chip capability storage.



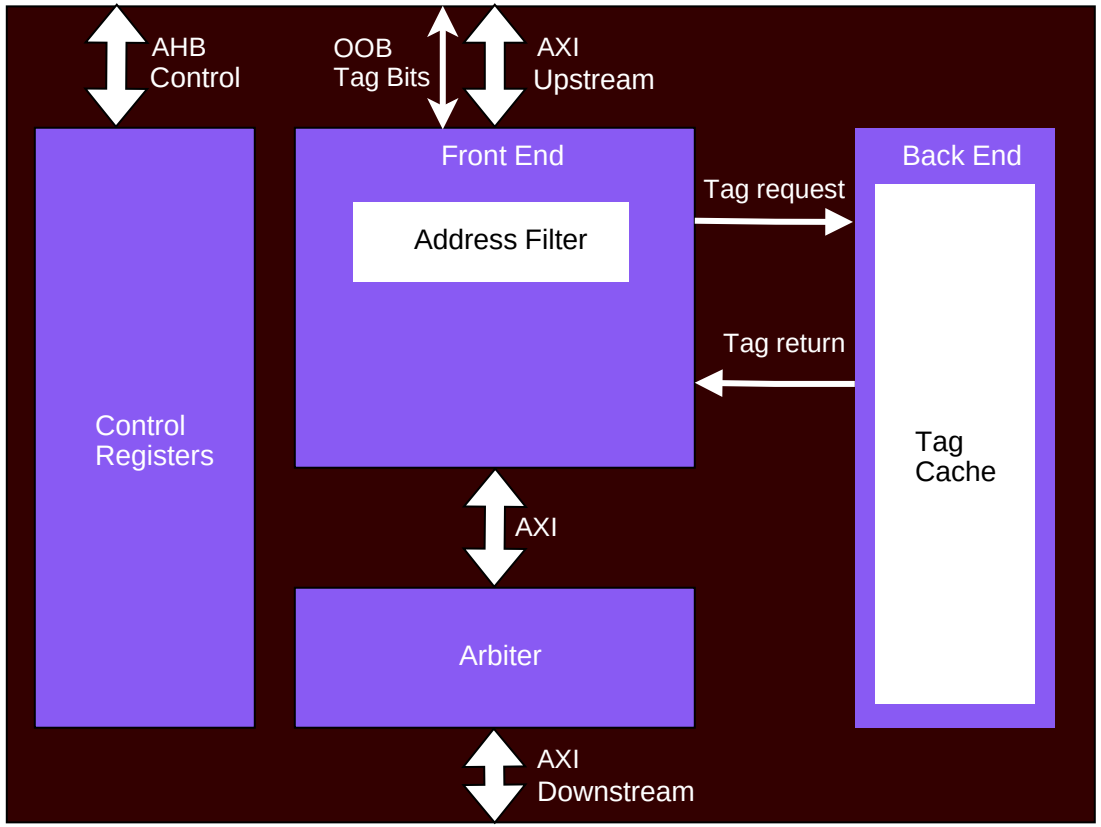
Tag Cache Mode

- Tags bits for recently accessed memory stored in a tag cache
- Each Tag Cache Line caches the tags for nominally a 4K memory block
  - When the line size is 256-bit and CLEN is 128-bit

+ No limit on size of memory on which cache operates

- General issues associated with caching e.g. non-determinism compared to Tag Stash Mode

- When valid tags are sparse, cache can fill with lots of "empty" lines containing no valid tags



⬡ Tag Group-Bit Cache Mode

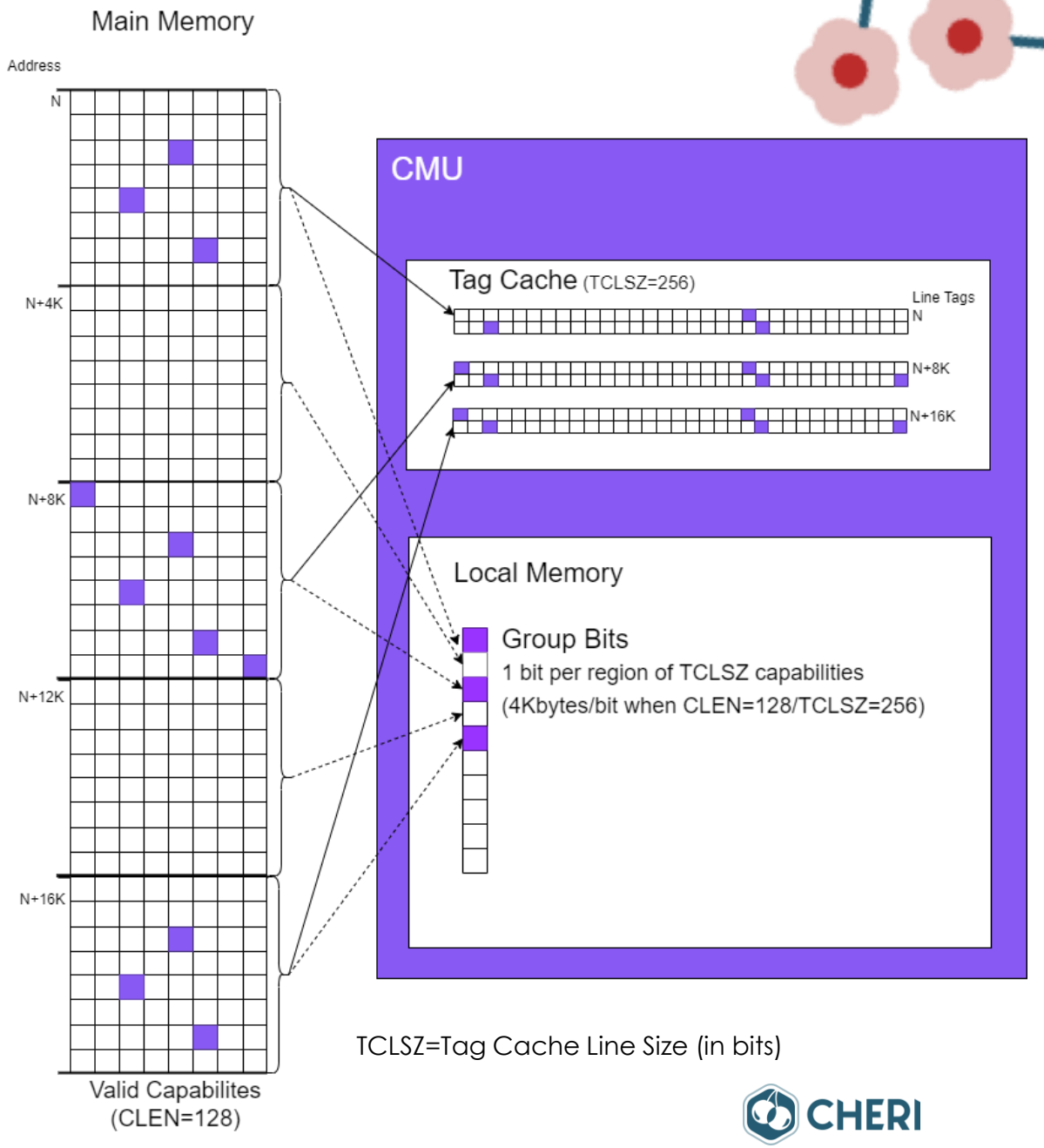
An enhancement to *Tag Cache Mode*.

- Nominally 1 bit of local memory allocated for each 4K block of memory where capabilities can be stored
  - This is when the line size is 256-bit and CLEN is 128-bit
- This Group Bit is set if the block contains one or more valid capabilities

+ Avoids 'wasting' cache space on empty lines

+ When the bit is clear, no tag store access needed:
  + For any load
  + For any non- or invalid capability write

+ Local memory requirement a fraction (1/256) of that for *Tag Stash Mode*

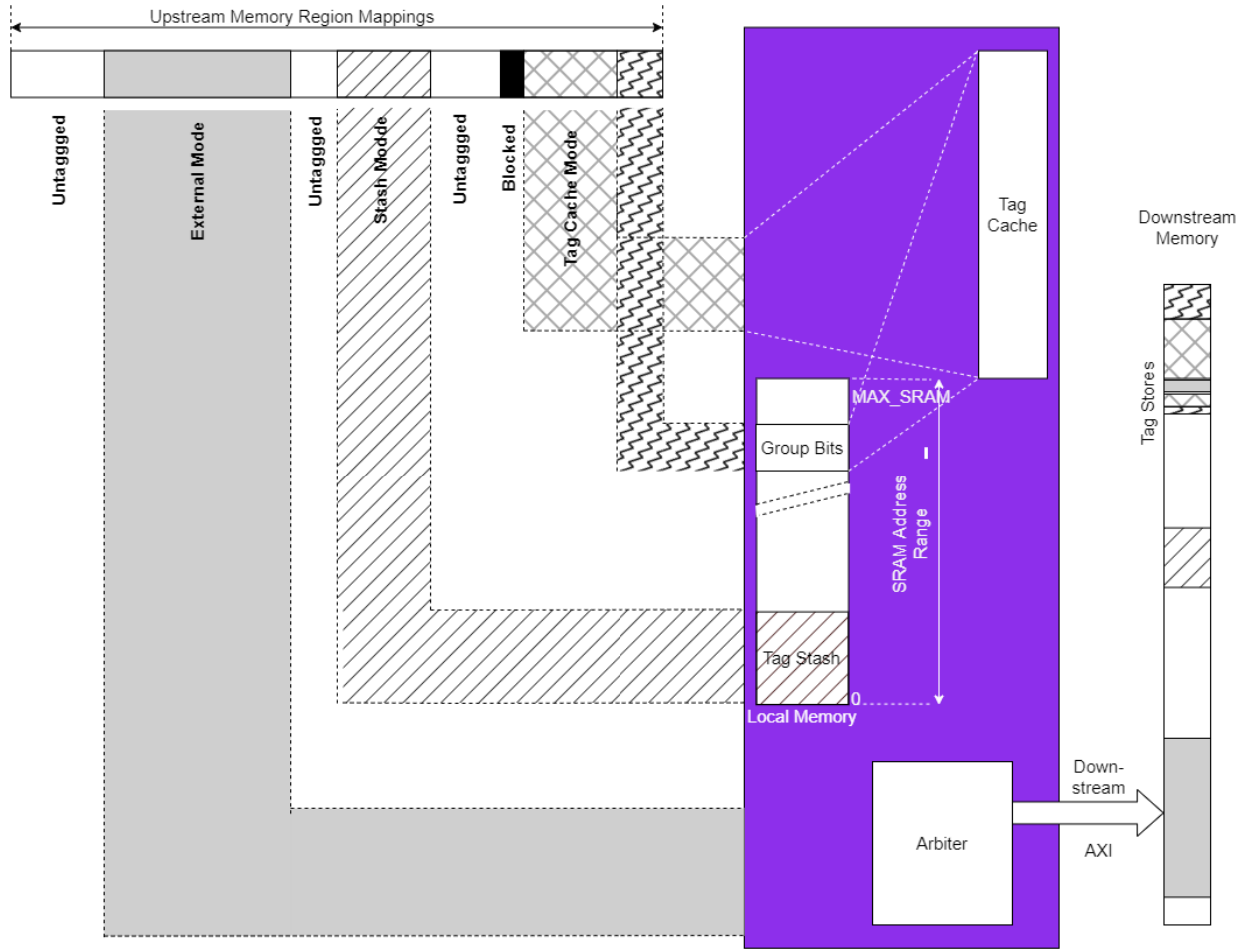


TCLSZ=Tag Cache Line Size (in bits)

CHERI

Combining Modes: Address Filtering

- CMU has a configurable number of address filters
- Each Address Filter Table Entry (AFTE) defines a region
  - Uses a NAPOT encoding scheme similar to that used in the PMP
- Unmapped regions are handled as untagged
- Tag handling strategies can be assigned on a per region basis at run-time
- Memory used as tag storage can, by default, only accessed by the CMU



Other Industrial Quality Level Features

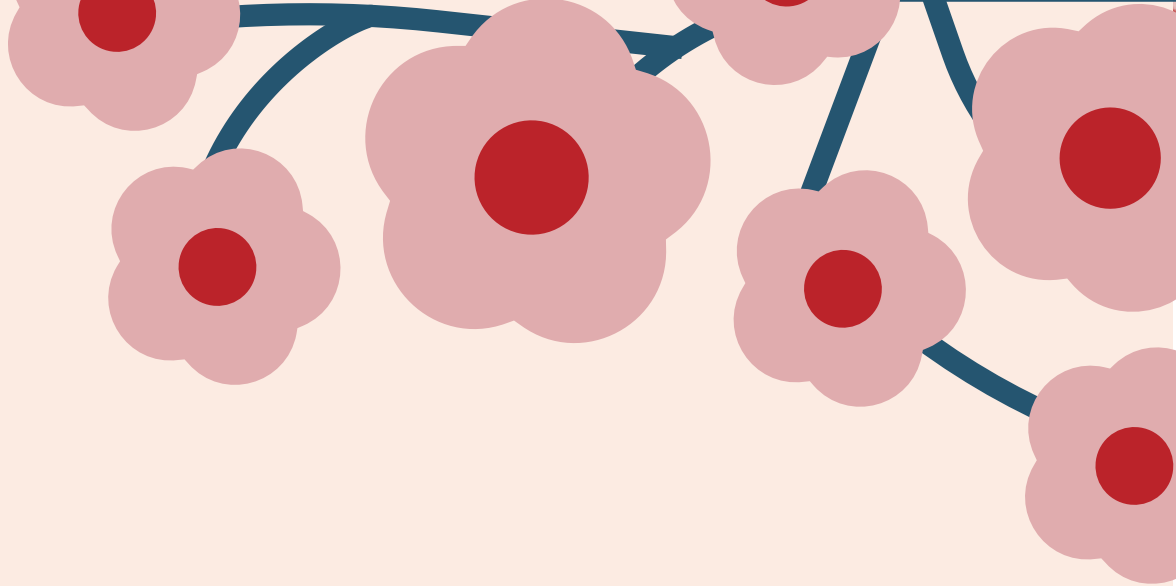
- Implemented using Codasip's Process Development Framework (CPDF)
  - This process has been certified to ISO 26262:2018 (ASIL D) and ISO/SAE 21434:2021
  - Suitable for use in functional safety applications
- Reliability, Availability and Serviceability (RAS) features (all optional):
  - ECC protection of all RAMs local to the CMU
  - Parity support on AXI ports
  - RV-I RERI (RAS Error Record Register Interface) Specification compliant error reporting
- Performance Monitoring
  - A configurable number of performance counters for optimising/evaluating the efficiency of the tag handling
- Efficient Tag Revocation
  - Can be programmed with a physical address region in which to invalidate all tags

Status and Acknowledgements

- Release now available for partner evaluation
- Evaluation Platform
  - Integrates CMU with Codasip X730 core on an FPGA board
  - Booting Linux in Tag Group Cache Mode and running Doom!
  - Platform release for partners available imminent
  - Next step integrating performance counters into Linux profiler we can perform detailed performance study
- This work acknowledges the research done at Cambridge University in this area(Efficient Tagged Memory by Alexandre Joannou et al.)

Codasip

UNIVERSITY OF CAMBRIDGE



CHERI

THANK YOU

Contact mark.hill@codasip.com

Web www.codasip.com

Codasip