ACCELERATOR DESIGN WITH OPENCL

(ATHENS WEEK 19-24 MARCH, 2018)

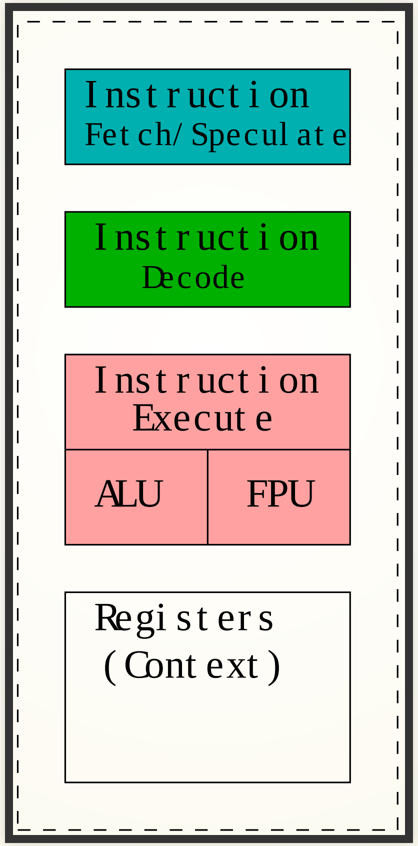
WHAT DO WE KNOW SO FAR ?

- There are three types of parallelism
  - Task Parallelism
  - Data Parallelism
  - Pipeline
- We saw the reasons for memory stalls and latency.
- The techniques to hide latency through Caching.
- The Virtual Memory.

WHAT DO WE KNOW SO FAR ?

- We saw the evolution of processors from
  - Uniprocessor to ...
  - Multicores with Simultaneous Multi-Threading.
- And we said hello to the world from our GPU (Mali-T628).

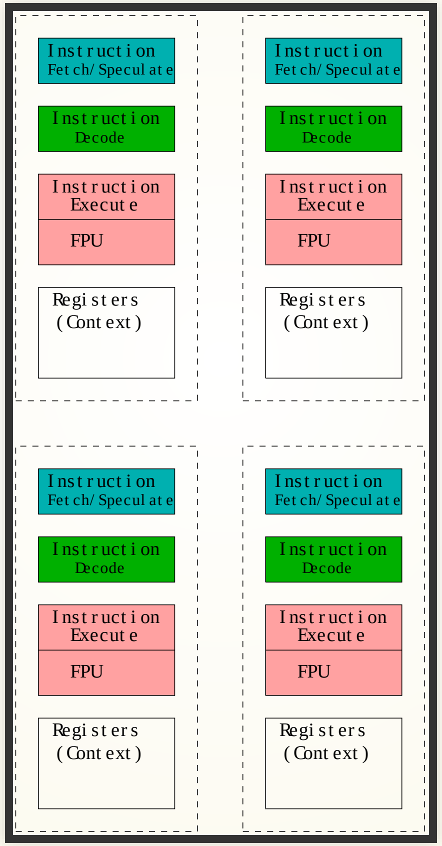
GPU ARCHITECTURE : UNIPROCESSOR



GPU ARCHITECTURE : EVOLUTION

- GPUs took a completely different path of evolution.
- Because they live in a embarrasingly data-parallel environment.
- The memory stalls/latency problems are still there.
- So are the solutions to hide them.

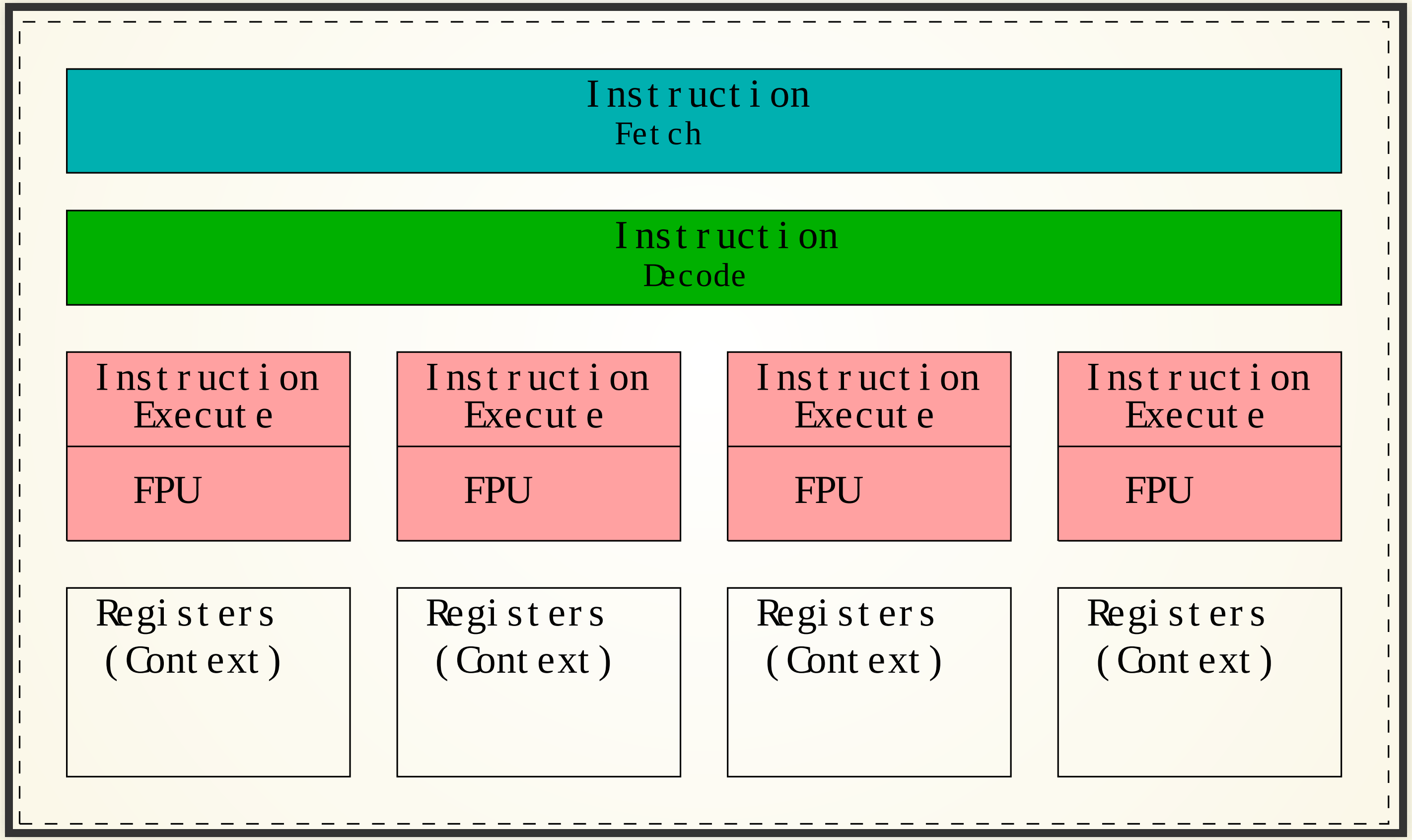
GPU ARCHITECTURE : MIMD



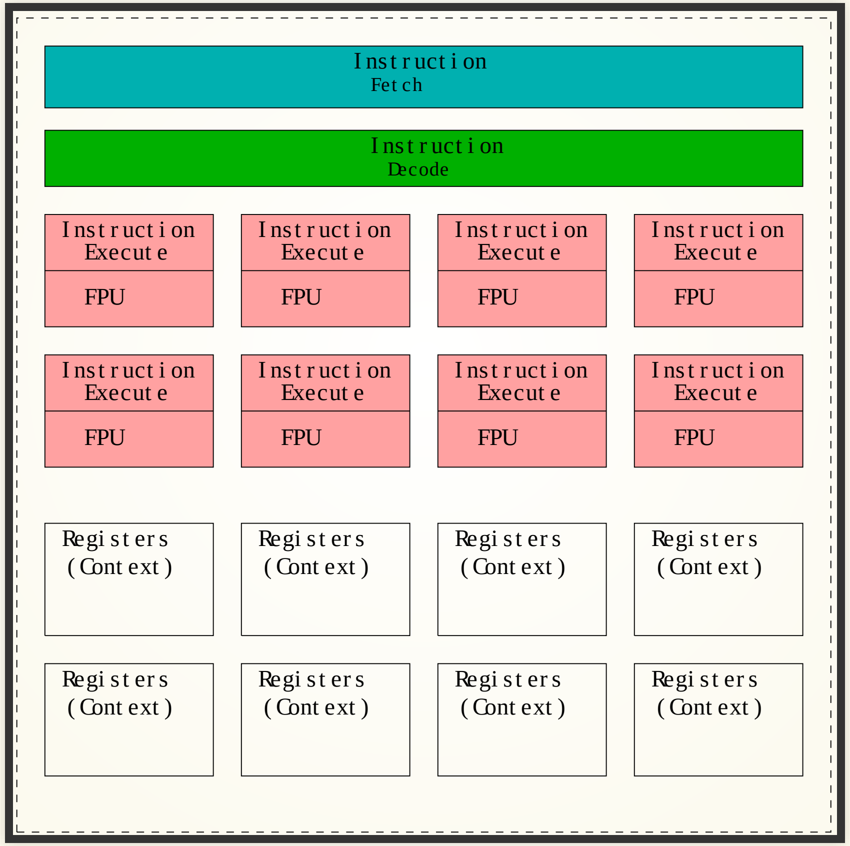
GPU Architecture : Evolution

- MIMD, but wait, we don't need the mutliple-instruction streams.
- let' get rid of them.

GPU ARCHITECTURE : SIMD



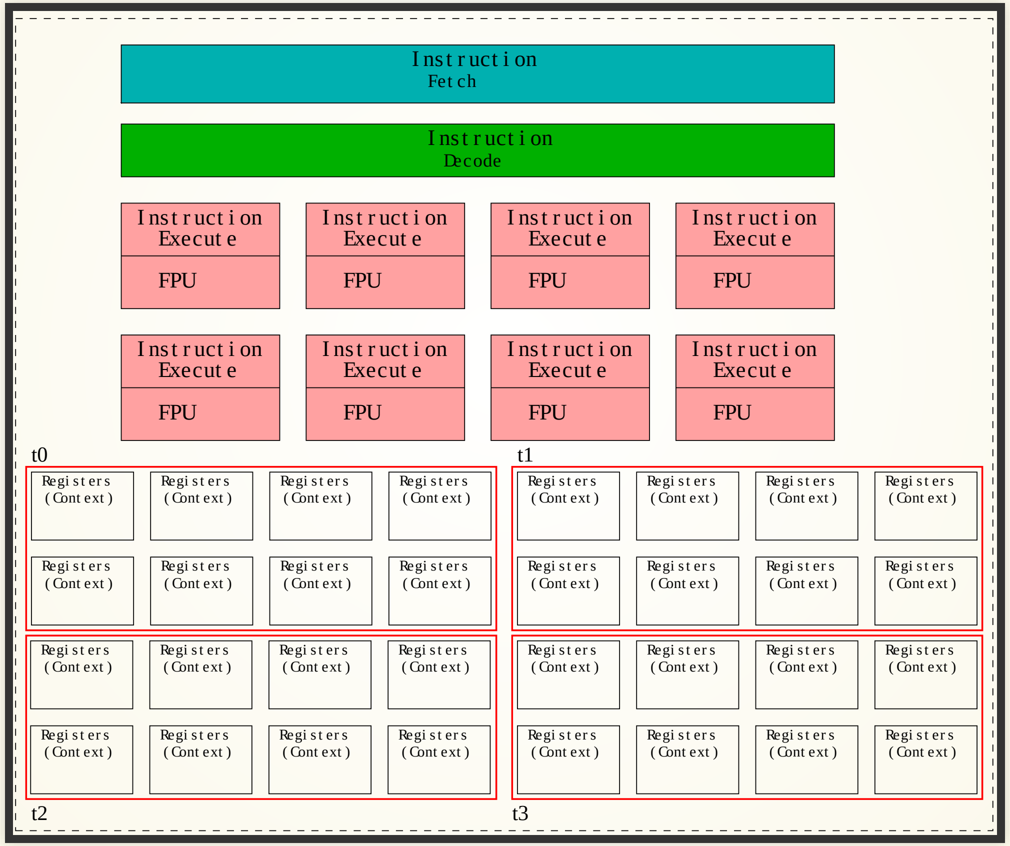
GPU ARCHITECTURE : MORE SIMD



GPU ARCHITECTURE : MORE SIMD

- Let's not forget our old friend Multi-Threading.
- Which helped us manage latency.

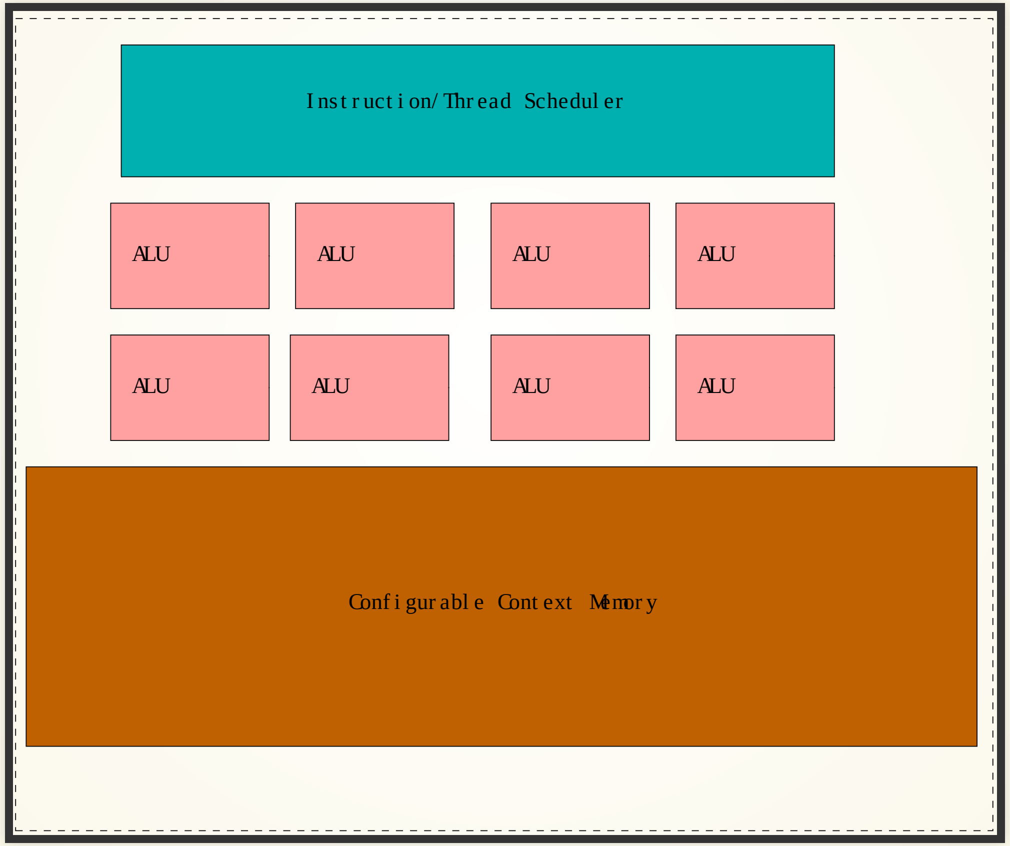
GPU ARCHITECTURE : SIMD WITH MULTI-THREADING.



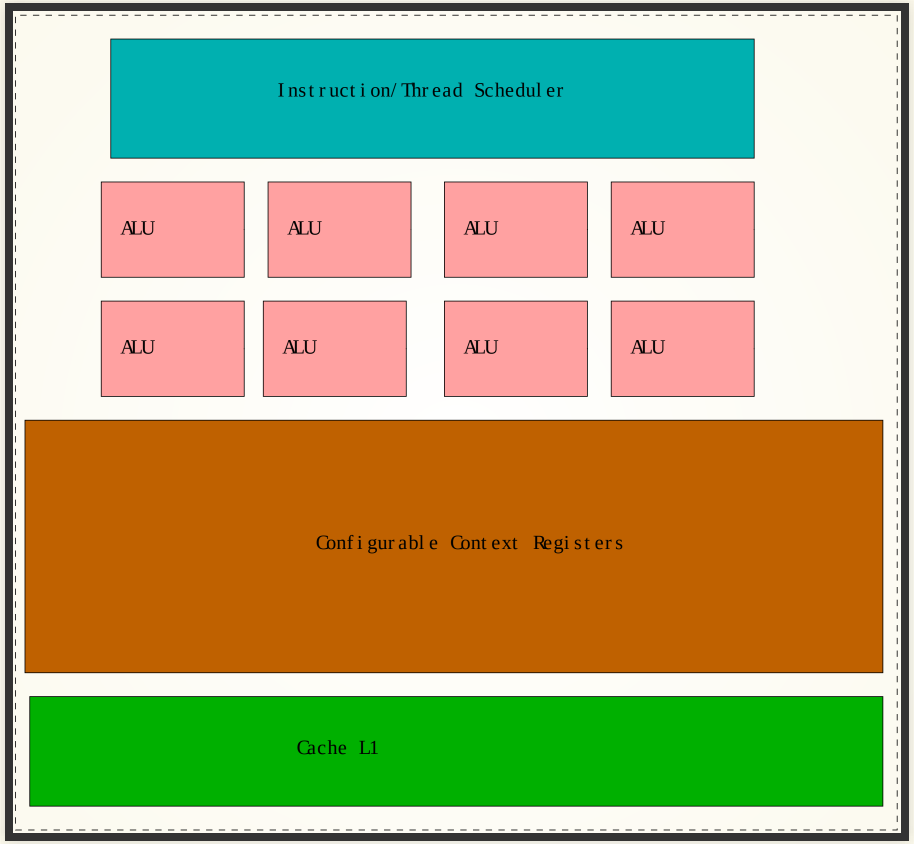
QUIZ

- What is the peak performance of this core in Gflops ?

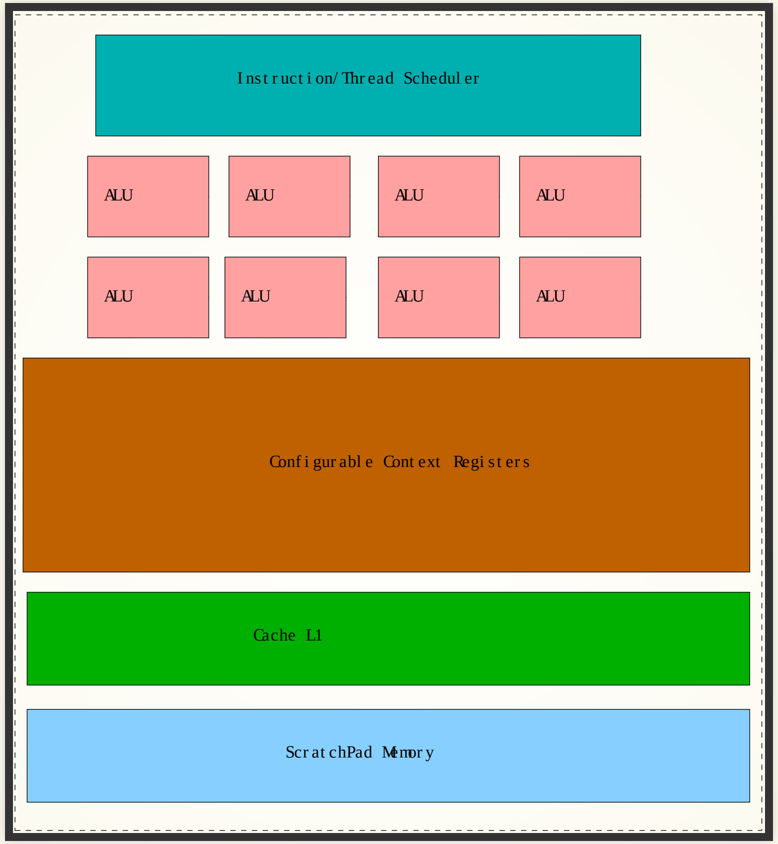
GPU ARCHITECTURE : REFINEMENTS



GPU ARCHITECTURE : REFINEMENTS

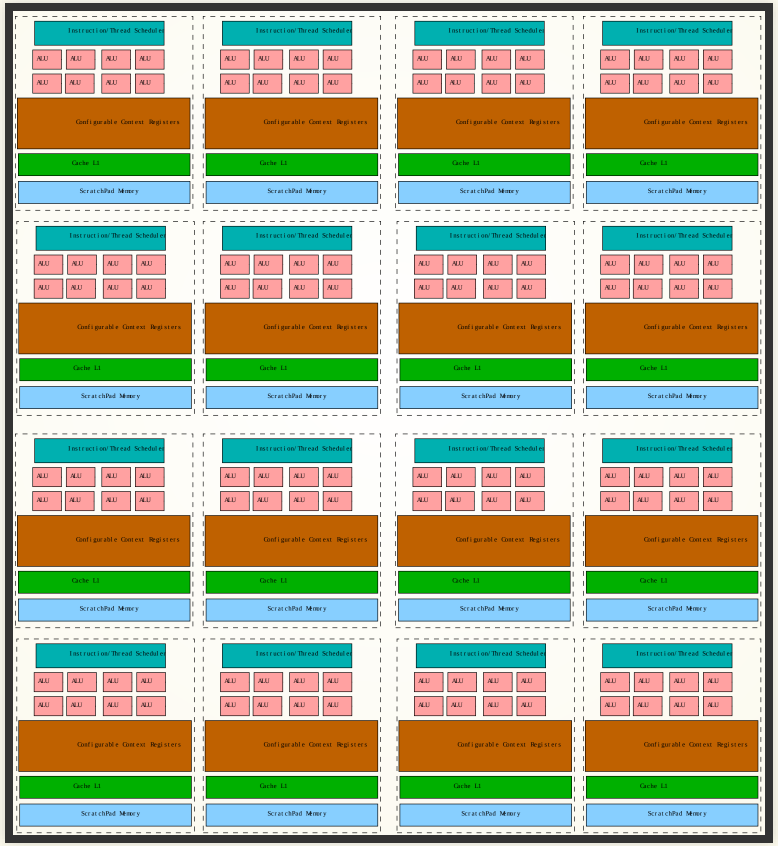


GPU ARCHITECTURE : REFINEMENTS



- Adding Scratchpad memory, so that threads can communicate locally.

GPU: MULTIPLE SHADER CORES



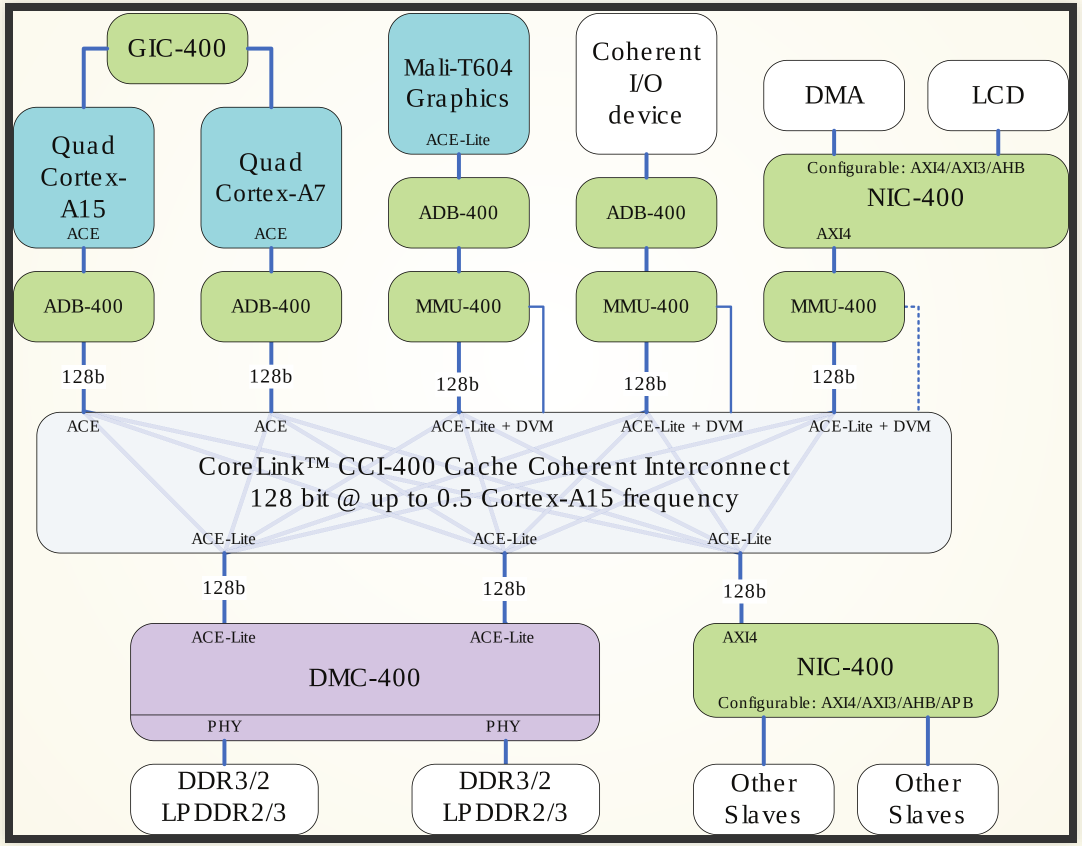
OUR GPU : MALI T628

- ARM MidGard family.
- Can be configures for 4-16 cores.
- configurable SIMD
  - 2x FP64, 4x FP32, 8x FP16, 2x int64, 4x int32, 8x int16, 16x int8
- Two L1 Caches/ Shader core 16KB
- L2 Cache can be configured for upto 64KB.
- Each core Rated at 17 Flops/cycle. (FP32)
- 64 byte Cache lines

SOURCE: MALI T628

- https://community.arm.com/graphics/b/blog/posts/the-mali-gpu-an-abstract-machine-part-3---the-midgard-shader-core
- https://community.arm.com/graphics/f/discussions/6557/mali-t628-gpu-activity-in-streamline

EXAMPLE HETEROGENEOUS SOCS



EXPRESSING PARALLEISM

- NDRangeKernel
- global_work_size() defines that total no. of elements.
- if each element is independent it is also the number of work_items.
- each work item can be associated with one thread.

EXPRESSING PARALLEISM

- the global work can be separated into groups.
- get_group_id() gives the id of the group.
- get_local_id() gives the id of the local work item within the group.

WORK ITEM RELATED FUNCTIONS:

- get_work_dim()
- get_global_size()
- get_global_id()
- get_local_size()
- get_local_id()
- get_num_groups()
- get_group_id()
- get_global_offset()

SYNCHRONIZATION FUNCTIONS: MEM FENCE

- mem_fence: all memory accesses preceding mem_fence must end before starting memory accesses following mem_fence.
- read_mem_fence : only for loads.
- write_mem_fence: only for stores.
  - arguments: CLK_LOCAL_MEM_FENCE: only load/stores to local memory.
  - arguments: CLK_GLOBAL_MEM_FENCE: only load/stores to global memory.

SYNCHRONIZATION FUNCTIONS: BARRIER

- All work-items in a work-group must execute this function before the work group can proceed.
- Barrier also issues a mem_fence either to CLK_LOCAL_MEM_FENCE or CLK_GLOBAL_MEM_FENCE.
- There is no way to synchronize work items in different work groups.

LAB WORK 1

- Vector addition with size N
- Calculate speedup with varying N.
- Measure Flops/s.
- Calculate the average of a vector.
- Calculate the average of a vector using workgroups.
- Measure speedup.

LAB WORK 2

- Write a Matrix multiplication routing with two matrices of size M x K, K x N.
- where M=K=N
- measure speed up
- use streamline to see various statistics about Cache/TLB miss.
- Measure Flops/S.

DEBUGGER: MGD

- in a405-xx.enst.fr (desktop) clone the git depot.
- source init.sh > /dev/null
- module load mali/4.4
- mgd
  - in odroid
  - source init_odroid.sh
  - mgddaemon
  - make debug

PERFORMANCE MONITOR: STREAMLINE

- run start_gator.sh in tpt39/
  - cd tpt39; ./start_gator.sh&
- in a405-XX.enst.fr
  - $ source init.sh
  - $ module load mali/4.4
  - $ streamline