About

- MIAOW is the implementation of the compute unit, not any auxiliary logic that produces any graphics

Wiki Home

- Southern Islands ISA by AMD
- **TODO: What is GPGPU?**
- Available
    - RTL implementation of MIAOW compute unit
    - Testbench for verification
    - Benchmarks known to run
    - FPGA bootstrapping framework
- Future work
    - Synthesis scripts for area and power analysis
    - Verilog for hardware dispatcher for controlling multiple CUs (compute units)

Architecture

- Compute unit has a few logical components (like a CPU!)
- Lots of complexity in each component
- Steps
    - Fetch
        - Fetch retrieves instructions from the outside world
    - Wavepool
        - Wavepool tracks the progress of instruction execution (up to 40 wavefronts)
    - Decode
        - Similar to the job of the control unit in the CPU, it enables needed resources
    - Issue
        - Most complicated part. It tells when resources are available and execution can proceed. **Incredibly complicated and little documented!**
    - Exec
        - Generates an execution mask

        - Indicates which of the 64 threads in a wavefront are actually supposed to be executing
    - SALU - scalar ALU
        - Performs arithmetic operations such as branch instructions
    - SIMD - 16 wide integer vector ALU
        - There are 4 of these in a compute unit
        - From https://en.wikipedia.org/wiki/SIMD
            - Single Instruction, Multiple Data
            - Parallelism, but not concurrency
                - Parallel computations, but only a single process instruction
            - Examples: adjusting contrast on digital image, adjusting volume of digital audio
    - SIMF - 16 wide floating point vector ALU
        - Also, 4 of these
    - SGPR - Scalar general purpose register file
        - Also used for scalar arithmetic operations?
            - I think that, like a CPU, this is the register file providing value to the SALU
        - Also where constant data shared across multiple threads are stored
    - VGPR - Vector general purpose register file
        - To the vALU
    - RFA - Register file arbitrator
        - *Arbitrator is something that settles a dispute*
        - Mediate access to the single write port possessed by the VGPR
        - Prioritizes access to LSU b/c memory operations have high latency
        - If they are all filled, this permits progress on whichever wavefront was stalled on them
    - LSU
        - Deals with memory requests

Architecture Overview

- In project: instruction traces and memory values for benchmarks provided by AMD APP SDK

    - Tracing: instruction traces are logs for that instruction execution (breadcrumbs for debugging)
    - SDK: Software development tools for creation of applications
- Design was based off of AMD's Graphics Core Next (GCN) micro-architecture
    - https://www.youtube.com/watch?v=lLCCGS5vUx4
    - https://www.youtube.com/watch?v=v3dUhep0rBs
        - Pre-emption -> certain tasks are prioritized above other ones
        - Task switching can lower overall efficiency
        - **https://www.amd.com/Documents/GCN_Architecture_whitepaper.pdf**
- Compute unit - module that performs computation
- Dispatcher - module that schedules workloads on CUs. (Research group has several variants.)
    - Receives workloads from main processor
    - Commercial GPUs receive kernels and shaders in PCs
        - Kernel:
            - https://www.youtube.com/watch?v=mycVSMyShk8
                - Core of what is going on
                - Manages resources and processes
                - Micro-kernel, kernel talks to drivers
        - Shader:
            - Program that does shading / special effects
            - Most are coded for a GPU
            - Shading languages are used to program GPU rendering pipeline
            - Superseded fixed-function pipeline (hardcoded basic logic)?
            - Modern use introduced by Pixar (with RenderMan)
            - **Unified shader model** - both vertex and fragment processing
- Wavefront - (warp in Nvidia docs) - collection of threads (64) scheduled to run on compute unit.
    - Unit of execution dispatcher schedules on CUs
    - https://blogs.msdn.microsoft.com/nativeconcurrency/2012/03/26/warp-or-wavefront-of-gpu-threads/
        - Most basic unit of scheduling
        - Smallest executable unit of code

        - Processes a single instruction over all the threads in it at the same time
        - Minimum size of data processed in SIMD fashion
- Workitem - (thread in Nvidia docs)
- Workgroup - ('thread-block' in Nvidia) - collection of workitems that can work together, possibly across different wavefronts but mapped to a single CU
- Local data store - ('shared memory' in Nvidia) - communication and coordination between workitems in a workgroup
- Global data store - ('global memory') - sharing data between multiple workgroups
- Highly parallel compute accelerator from MIAOW instance
    - Includes memory controller as well
- What is IP?
- L1 caches - for scalar data and instructions
- 1 unified L2 cache
- Very similar to GCN's compute unit, difference is in memory organization (b/c of manufacturing processes)

Wavepool

- Keeps track of 40 concurrently executing wavefronts
- Keeps track state of each wavefront
- Each wavefront that is queued carries certain pieces of info
    - Several indicate what subset of CU storage has been allocated to wavefront
    - Internal state tracking mechanisms
        - Wavefront ID
        - LDS base address
        - SGPR base address
        - VGPR base address

LSU

- LSU is the **load store unit**
- Servicing memory requests issued by waveforms
- Modules - broken in 2
    - lsu_opcode_decoder
        - gets type of instruction

        - number of memory operations the request decompresses to
        - whether multiple operations to registers are necessary
        - some instructions need 1 word, others need 4
    - lsu_op_manager
        - info from lsu_opcode_decoder passed here
        - performs actual request
        - keeps track of progress
        - notifies issue unit when complete
        - read/write, address calc
        - state machine
        - currently, single slot LSU for simplicity of debugging
    - lsu_addr_calculator
        - mem addr calculation is convoluted
            - especially vector mem operations

Bringup and Verification

- Synopsys VCS 2012
- Tracemon - testing harness used to generate instruction traces

Udacity

- https://www.youtube.com/watch?v=usY0643pYs8
    - Why divide problem into blocks
    - When do blocks run
    - What order do they run
    - How do they cooperate? What are the limitations?
    - SM (compute unit)
        - Contains simple processors + memory
        - **GPU is responsible for organizing blocks to compute units**
        - **We give GPU a big pile of thread blocks**
        - **GPU will take care of assigning them to run**