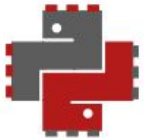
PyMTL3

CSE 293
Final Project
Farzaneh Rabiei



PyMTL3

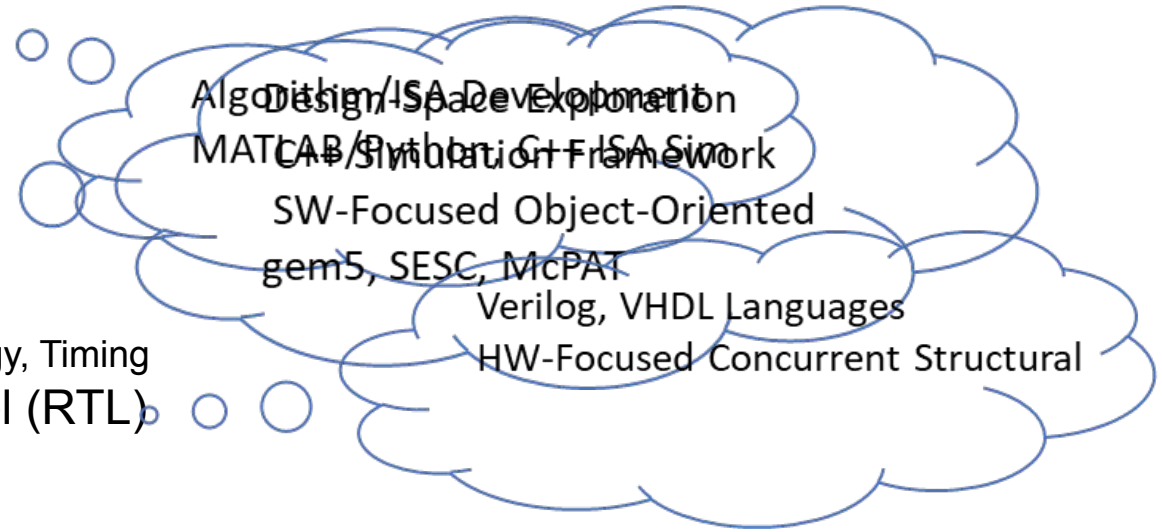
PyMTL3 is a Python-based hardware modeling framework digital systems.

Motivation:

- Productivity and Readability
- Modularity and Compositionality
- Simulation and Verification
- Integration with Python Ecosystem
- Education and Research

PyMTL3: Multi-Level Modeling

1. Functional-level (FL)
   a. Behavior
   b. Functionality
   c. No Timing
2. Cycle-level (CL)
   a. Behavior
   b. Cycle-Approximate
   c. Analytical Area, Energy, Timing
3. Register-transfer-level (RTL)
   a. Cycle-Accurate
   b. Resource-Accurate
   c. Bit-Accurate
   d. Gate-Level Area, Energy and Timing



Some of the PyMTL3 features:

- InPort and OutPort classes
- Wire class
- Update_ff → on every rising clock edge
- Update → one or more times within a clock cycle
- Structural composition to connect child components
- Parameterized components
- Line tracing and Waveforms
- Verification with Unit Testing

Lab1: PyMTL3 Low-Level Modeling

Part1: Adder

- FullAdder
- 4-bit Ripple Carry Adder using FullAdder

- Learning Outcome:
  - Syntax of new language
  - Low-level modeling
  - Combinational logic

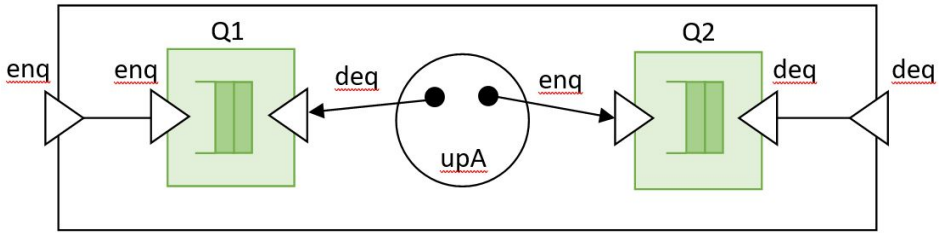
Lab1: PyMTL3 High-Level Modeling and RTL-Level

Part 2: Finding a MinMax of 3 inputs

- Write a PyMTL3 code to find the min and Max between 3 inputs in high-level
- Create an RTL version of the code
- Create the systemVerilog code
  - Use VerilogTranslationPass
- Visualize their designs
  - textWave
- Learning Outcome:
  - Structural RTL model
  - High-level modeling

Lab1: PyMTL3 High-Level Modeling

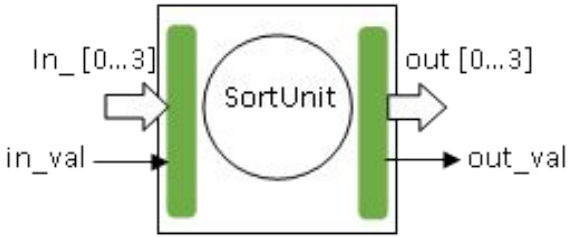
- Part 3: Double Queue Function-Level
  - QueueFL
  - DoubleQueueFL
  - ram_1r1w_sync
  - FIFO_ram



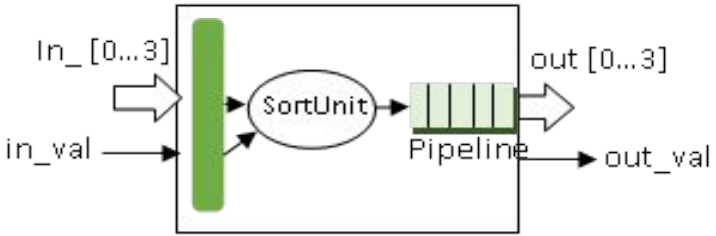
- Learning Outcome:
  - Features of High-level modeling
  - Clearly separate the sequential child components from the combinational child components

Lab2: Function-Level / Cycle-level Modeling

Part1: Sort



- SortFL
  - Change the code to accept any arbitrary array size
  - Change the code to use merge sort instead of quick sort
  - Add a random test case for a sort unit
- SortCL
  - Change the code to accept any arbitrary array size
  - Change the code to 4-stage pipeline currently it has 3-stage pipeline
- Learning Outcome:
  - FL\CL



Lab2: Big Data in PyMTL3

Part2: Matrix

- Multiplication
- A huge data streaming input and finding the most frequent value

- Learning Outcome:
  - Optimize the design
  - Fit the design into target hardware

Lab2:

Part 3:

- Counter for seven segment
  - Implement a Counter that count one in every 4 cycles
  - Show the result on the seven Segment

- Learning Outcome:
  - Change the clock cycle
  - Sequential logic

References

[1] S. Jiang, P. Pan, Y. Ou, and C. Batten, “PyMTL3: A Python Framework for Open-Source Hardware Modeling, Generation, Simulation, and Verification,” *IEEE Micro*, vol. 40, no. 4, pp. 58–66, Jul. 2020, doi: 10.1109/MM.2020.2997638.

[2] “ece5745-tut3-pymtl.pdf.” Accessed: Jun. 04, 2023. [Online]. Available: https://www.csl.cornell.edu/courses/ece5745/2022sp/handouts/ece5745-tut3-pymtl.pdf

[3] “pymtl-tutorial-overview-isca2019.pdf.” Accessed: Jun. 04, 2023. [Online]. Available: https://www.csl.cornell.edu/pymtl2019/pymtl-tutorial-overview-isca2019.pdf

questions?