Jetic Gū

1. Handwritten submissions and proprietary formats (e.g. Pages or MS Word) **will not be graded**.
2. Mathematical expressions must be written entirely using LaTeX, otherwise **50%-100%** of marks will be deducted.
3. Circuits must be **tested** using switches/probs against a truth table. Untested circuits will receive 0.

Submission File structure:

```
submission.zip
    — mycache.py
    — circuit1.cct
    — circuit2.cct
    — csci250.clf
```

The cct files are 2.5pt each, `mycache.py` is worth 5pt.

Lab 2

1. Cache simulator (5pt). Here are the instructions:
    1. Download jetic.org/dl/mycache.py and jetic.org/dl/mycache_float.py
    2. `MyCache` class takes L1-L3 cache latency and block sizes as parameters, as well as main memory access latency.
    3. You should modify the `access` method in `mycache.py`, this method takes a `list` of `int` as parameters, for which you'll need to simulate cache operations.
    4. You can implement the levels of cache using whatever data structures you want. If a target address block is a hit in the current level, access is provided and you log the latency. If it's a miss, you'll need to copy the entry from a lower-level cache, and log both the latency at the current level and lower-levels.
    5. If a new block needed to be added to the current cache but the cache is full, the oldest entry is replaced with the new one.
    6. You can test your implementation by using the provided `mycache_test.py`. You should add more test cases, and do not change the interface of the `MyCache` class.
    7. Submit `mycache.py` only for this question.
2. Implement a 16bit RAM component using the `PROM/RAM wizard` (2.5pt).
    1. Use `Model Wizard` to create a component called `RAM16bit`;
    2. Show your component working in `circuit1.cct` using HEX keyboards and switch.
3. Implement a 2bit address 4bit SRAM unit in LogicWorks (2.5pt).
    1. Use SR Latch and gates to implement an SRAM cell, save it in your library as `1bit SRAM Cell` in your library, input `B`, `nB`, `Select`, Output `C`, `nC`.

2. Put `Data_In, Data_Out` as 4bit IO, input `A` as 2bit, `Bit_Select` as 1bit, `R_nW` as 1bit.
3. Show your component working in `circuit2.cct` using HEX keyboards and switchs (2.5pt)