6.004 Fall 2018
L12 – Non-pipelined Multicycle Processors
Take-Home Problem

Due at the beginning of recitation R12 on Friday October 19.

Discuss the viability and pros/cons of modifying the doExecute rule on L21 Slide 17 as follows:

```module mkProcMulticycle(Empty);
  ...
  rule doExecute if (state == Execute);
    let inst <- mem.resp;
    Code to 1. execute all instructions except
    Memory and multicycle instructions; go to Fetch
    Initiate instruction fetch
    Or 2. initiate memory access; go to LoadWait
    Or 3. initiate multicycle instruction; go to MCWait
    ...
  endmodule
```

Viability: Yes! (See below)

Pros: Non-memory instructions are executed in one cycle! → reducing the average cycles

Cons: Longer propagation delay for the doExecute path for non-memory instructions → The system clock needs to slow down (Larger clock period)

Updating PC with the calculated nextPC for `doFetch` (Original) vs. the calculated nextPC propagates through a MUX and then a request interface of the Memory (Modified)

Subtlety: We are assuming that the memory’s `req` and `resp` ports are independent, execute whenever they are ready and can be used at the same time. However, depending on a memory technology, using `req` and `resp` port at the same time may cause a deadlock situation. Note that the new `doExecute` rule tries to use both ports at the same time.