This is the description for the last two assignments. They are rather interrelated, and may need to be developed in unison, but the two assignments address two distinctly different parts of a processor. You are to design a multiple-cycle datapath (Assignment 8) and state machine control (Assignment 9) for the MIPS processor. At a minimum, your processor must be able to execute the following subset of MIPS instructions:

- **ADD**
- **ADDI**
- **LW**
- **BEQ**
- **SUB**
- **ANDI**
- **LB**
- **BNE**
- **AND**
- **ORI**
- **SW**
- **SLT**
- **OR**
- **MOVE**
- **SB**
- **SLTI**

Note that the SLT instruction is an R-type instructions, very similar to the SUB instruction, except that the result (placed in the destination register) is a 1 if the value in the first source register is less than the value in the second, and a 0 otherwise. Similarly, the SLTI is similar to a SUBI instruction (which is actually a pseudo-instruction and therefore doesn’t really exist in the MIPS instruction set).

Also, note that the MOVE instruction is actually a pseudo-instruction in the MIPS instruction set – however, we will implement it as if it were a “real” I-type instruction, where the IMM field in the instruction is not used. The op-code (bits 31-26) for the MOVE instruction we will use is 2410 (1816).

The test bench will include a memory module that will work much like the register file module you built for assignment 7 - it will do asynchronous reads (single ported) and synchronous writes.