Assignment No. 10

http://www.eng.tau.ac.il/~guy/Computer_Architecture/arch_home.html

Scoreboard Scheduling:

1. Consider a single general purpose register R and a single clock cycle. How many concurrent reads and writes can occur in the Scoreboard Algorithm to register R?

   What does this imply about the required functionality of the GPR (i.e. number of read and write ports)?

2. Draw a table which has a row for every "register" used by the data-structure of the Scoreboard algorithm, and a column for every phase. Fill each entry in the table with information about whether the register is read or written in that phase. If it is read, what value is expected $(0, 1$, input value etc.)? If it is written, what value is written?

   Add two columns to the tables that specify the maximum number of concurrent reads and the maximum number of concurrent writes.

3. Each line of the Scoreboard Algorithm (as described in class) corresponds to one clock cycle in which multiple read and write accesses are made to the Scoreboard data-structures.

   Consider an implementation of the Scoreboard Algorithm in which multiple accesses are not implemented in a single cycle but spread in time. Assume that the *order* of the accesses is preserved. (That means that when parallel accesses are issued, the requested order is the order of the accesses in the line of the algorithm).

   In which lines in the Scoreboard Algorithm does it matter that multiple accesses are not done in the same cycle? Suggest and prove an order of accesses for these lines.

4. The Scoreboard Algorithm that was described in class was designed for a "straight line" programs (namely, no jumps, no branches, no exceptions).

   Extend the algorithm to deal with branches and jumps. Try to minimize the modifications. Outline the correctness proof.

5. Consider a scheduling algorithm which:

   (a) Does not run into structural conflicts.

   (b) Does not run into deadlock.

   (c) Avoids data dependency conflicts.

   (d) Issues instructions in order.

   Prove that the scheduling algorithm executes straight line programs correctly.