Register File

This document provides the specifications of the register file (**regfile**) module.

The **regfile** module takes several inputs and has 5 output signals (see Figure in **processorComponents)**. Here is a brief description of some of the input signals.

- **source_reg1, source_reg2**: the register number of the two registers holding the source operands. The value currently stored in these registers must be returned in the output ports **reg1_data** and **reg2_data**.
-
- **destination_reg**: the register number of the register to be written with **destination_result_data** on the positive edge of the clock signal **clk**, provided the write is enabled (**clk_en** and **wr_destination_reg** are both asserted).

- **movil, movih**: 1-bit commands indicating which bye of the **destination_reg** must be updated with **immediate** data.

- **newcarry, newborrow**: 1-bit values of the carry and borrow coming out of the ALU. These values should update the *carry* and *borrow* flags in the **regfile** module on the positive edge of **clk** provided **clk_en** is asserted.

- **reset**: 1-bit signal to reset the register file module by initializing the registers and the carry/borrow flags (specific values provided in the stub).

- **clk**: 1-bit periodic clock signal used to synchronize updates across all sequential components of the processor.

- **clk_en**: used to obtain the logical effect of a slower clock. The physical clock may run at a much higher frequency but we want updates in our circuit to occur at a lower frequency. A common way to do this is to derive a **clk_en** signal that has a lower frequency, and to only perform the flip flop updates when the **clk_en** is also asserted at the edge of the fast running clock.
- Since we are running simulations with a logical clock (rather than a physical crystal oscillator generating the signal) this is largely ceremonial here. Look in the testbench to see what is the **clk_en** signal provided to **regfile**.