ELEC 379 : MICROCOMPUTER SYSTEM DESIGN
1997/98 WINTER SESSION TERM 2

Assignment 3 - RTL Design

due February 11, 1998

Question 1

In this assignment you will design, simulate and synthesize a 16 by 16 bit unsigned multiplier.

Use the following entity declaration for your design:

```
entity mult is
   port (
   a, b : in unsigned (15 downto 0) ;
   c : out unsigned (31 downto 0) ;
   clk, reset : in std_logic ) ;
end  mult ;
```

Your circuit should multiply `a` by `b` and output the product on `c`. You may not use the multiplication operator in your design. Instead, use the following algorithm:

1 set `c` to zero.

2 if the least-significant bit of `a` is '1', add `b` to `c`.

3 shift `a` right by 1 bit and `b` left by 1 bit (this can be done at the same time as the addition in step (2)).

4 if `a` is non-zero go to step 2, else stop.

The computation should start on the rising edge of the clock signal (`clk`) if the `reset` signal is asserted. `a` and `b` will only be valid when `reset` is asserted. The computation must terminate (output a valid result to `c`) in 35 clock cycles or less. `reset` will only be asserted for one clock cycle per computation. Your design must output intermediate results (sums) to the `c` output while the product is being computed.

Give a list of the registers required to implement the above algorithm including their widths. You may use internal registers that are wider than the corresponding inputs.

List a set of operations (register transfers with combinational functions) that are sufficient to implement the above algorithm (e.g. "set `c` to zero"). You may include more than one register transfer in each operation.

List a set of datapath control signals that are sufficient to carry out all of the required operations.

List a set of status signals going to the controller that are sufficient for the controller to operate properly.

List a set of controller states that are sufficient to carry out the operation of the device. For each state describe the operation (register transfer) to be performed during (and latched at the end of) that state. List each possible combination of controller state and datapath status and give the corresponding next controller state (i.e. the state transition table).

Split up your design into a datapath entity and a controller entity (you may use different names is you wish). You may place all of your code in the same file if you wish.

Write a synthesizeable architecture to implement the multiplier.

Include comments in your code as described the previous assignments.

Use the test bench in the file `~elec379/asg3tb.vhd` to test your design. Your design may take up to 35 clock cycles to complete the computation (fewer clock cycles is OK too). You may also want to make your own copy of the testbench and run it with different input values when testing.

When your design simulates without errors, read the instructions on the course Web page on compiling a hierarchical design, synthesize your design using Synopsys Design Compiler and plot schematics for each of the three entities.

Hand in the following: (1) lists described above: register names and widths, operations and register transfers, datapath control and status signals, controller states, and controller state transition table; (2) your VHDL code; (3) the log of your simulation; (4) the schematic of your synthesized circuit.