

Name _____

Computer Organization
EE 3755
Midterm Examination
29 October 2001, 12:40-13:30 CST

Problem 1 _____ (16 pts)

Problem 2 _____ (16 pts)

Problem 3 _____ (20 pts)

Problem 4 _____ (16 pts)

Problem 5 _____ (16 pts)

Problem 6 _____ (16 pts)

Alias _____

Exam Total _____ (100 pts)

Good Luck!

Problem 1: Add Verilog code to the 32-bit carry-lookahead adder module below for the carry[5] wire. Use the generate and propagate signals.

(16 pts)

```
module cla_32(sum,a,b);
    input [31:0] a, b;
    output [32:0] sum;

    wire [31:0] g, p, carry;

    // Code for other carry signals omitted.
    //
    // Start answer here ↓

    assign      carry[5] =

        cla_slice s0(sum[0],g[0],p[0],a[0],b[0],carry[0]);
        cla_slice s1(sum[1],g[1],p[1],a[1],b[1],carry[1]);
        cla_slice s2(sum[2],g[2],p[2],a[2],b[2],carry[2]);
    // Code for other cla_slices omitted.
        cla_slice s5(sum[5],g[5],p[5],a[5],b[5],carry[5]);
    // Code for other cla_slices omitted.

endmodule
```

Problem 2: Complete the module below so that 32-bit output `sum` is the sum of its 32-bit inputs, `a` and `b`. If input `s` is 1 then `a` and `b` are signed integers, otherwise they are unsigned integers. Output `overflow` should be set to 1 if the sum overflows.

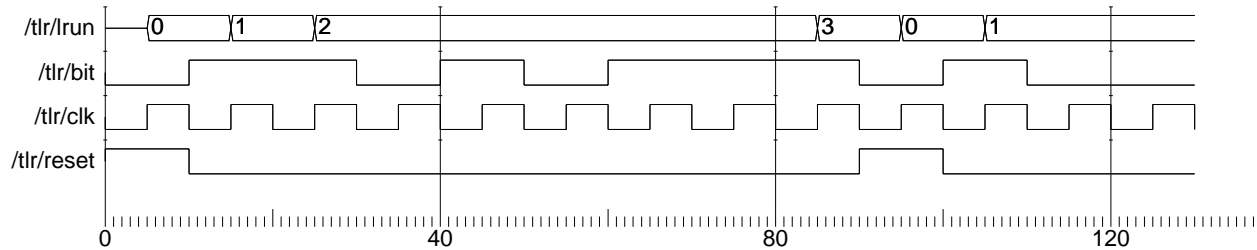
The module should synthesize to combinational logic. The solution can (and should) use the addition operator. (16 pts)

For partial credit, explain how to detect overflow in signed and unsigned addition.

```
module add(sum,overflow,a,b,s);
    input [31:0] a, b;
    input      s;
    output [31:0] sum;
    output      overflow;
```

```
endmodule
```

Problem 3: The module below examines a bit sequence one bit per clock cycle, as did the module in Homework 2 Problem 2. The input bit is valid on the positive edge of `clk`, that bit is the first of a new sequence if `reset` is high. The output, `lrun`, must be the longest number of consecutive 1's encountered since the last time `reset` was 1. For example, after 00111010 is received the output should be 3 since that is the length of the longest run of 1's. Also see the example timing below. As in Homework 2 the module must synthesize to sequential logic. (20 pts)



```

module longest_run(lrun,bit,reset,clk);
    output [31:0] lrun;
    input          bit, reset, clk;
  
```

endmodule

Problem 4: Show the hardware that the module below will synthesize in to. Indicate the types (edge- or level-triggered) of any registers synthesized.

(16 pts)

```
module syn(x,r,a,b,m,neg);
  input [31:0] a, b;
  input      m, neg;
  output [31:0] x, r;

  reg [31:0]    x, r, bn;

  always @( a or b or m or neg ) begin

    if( neg ) bn = -b; else bn = b;

    x = a + bn;

    if( m ) r = x + b;

  end

endmodule
```

Problem 5: Convert the following numbers: (16 pts)

Decimal 12 to 8-bit Binary:

Decimal -12 to 8-bit Binary:

Decimal $12\frac{5}{8}$ to Binary (as many bits as needed):

Decimal $12\frac{5}{8}$ to Normalized Binary Scientific Notation (write as $f \times 2^e$):

Decimal $12\frac{5}{8}$ to IEEE 754 Single Precision (Show in hexadecimal):

Problem 6: The high-radix Booth multiplier is faster than the radix-2 multipliers (for example, the streamlined signed multiplier). (16 pts)

(a) Compared to the streamlined signed multiplier, what additional hardware is needed for a high-radix Booth multiplier?

(b) Name two advantages that a high-radix Booth multiplier has over an ordinary high-radix multiplier.