1: Write a module in verilog for a tristate buffer. It should have two inputs, enable and in, and one output which goes high-z if enable is low, and reflects the input otherwise.

```verilog
module triBuff(input En, input D, output Out);
    assign Out = (En)?D:'bz;
endmodule
```

trinary operator: $\text{cond}\ ?\ \text{it1} : \text{it0}$

2: The following test bench is meant to test a two-bit adder. Write the module it is testing in behavioral verilog.

```verilog
module bench;
    reg A[0:1];
    reg B[0:1];
    wire C[0:2];

    adder a(A, B, C);

    initial begin
        $dumpvars(0,bench);
        A=0;
        B=0;
        #10;
        A=1;
        B=2;
        #10;
        A=3;
        B=3;
        #10;
        A=2;
        B=2;
        #10;
        $finish;
    end
endmodule
```
3: Benefits and Costs of Behavioral Verilog: In the instances below, select whether Behavioral or Structural Verilog would be the better option:

- Constructing a working model of a microprocessor: B
- Predicting the delay of an adder: S
- Evaluating timing hazards in an HDMI system: S
- Simulating the behavior of a large system of USB peripherals: B
- Checking for decoding spikes on the output of a combinational system: S
- Simulating the internal processes in an SRAM: S
- Checking for setup and hold time violations in a state machine: S

X: Supposing you have a GPIO module with two one bit registers, one each for direction and data. Write verilog showing it being used as an input and as an output. (Two separate circuits, but you don’t need to write a complete module, just show the relevant lines of structural code.) The GPIO module is instantiated by:

```verilog
gpio #1(datain, dataout, directionin, clock, gpio_inout);
```

As Input:

```verilog
assign val = dataout;
```

As Output:

```verilog
assign datain = val;
```

5: Suppose you have the following CPU:

```verilog
cpu #1(inout [0:7] Data, out [0:15] Address, clk, reset);
```

A: Draw a diagram connecting it to a bank of 8 LEDs memory mapped to address 0x8000:
B: Write verilog code for these connections: (MSI Components like MUXes & RAM allowed)

\[
\begin{align*}
\text{RAM} & \text{-chip}: r1(\text{Address}[14:0], \text{RAM-data bus}) \\
\text{bidir-mux}: m1(\text{Data}, \text{Address}[15], \text{RAM-data bus, LEDs})
\end{align*}
\]

6: Write C code to count in binary on the LEDs.

```c
int main()
{
    int8* val;
    val = 0x8000;
    while (1)
    {
        *val++;
    }
}
```
7: Suppose you have a GPIO with 1pF of internal capacitance to ground and an output resistance of 10k ohms. What is the maximum achievable frequency of this IO?

\[ \text{Decay time: } \frac{1}{3RC} \]

\[ \text{Bit rate: } \frac{1}{3RC} = \frac{1}{30\text{ns}} = 33.3 \text{ MBPS} \]

8: Battery life is typically measured in Amps*Hours. A single rechargeable AA battery is rated for 2450mA-hours and has a terminal voltage of 1.5 volts. Assuming a very efficient converter to 3 volts exists and a microcontroller needs 3.6mA, how long can this battery run it?

\[ E = P \cdot t \quad 2450 \text{mA-hours} @ 1.5V \rightarrow 1225 \text{mA-hours} @ 3V \quad (I \cdot V \text{ ratio const}) \]

\[ \frac{1225 \text{mA-hours}}{3.6 \text{mA}} = 340.28 \text{ hours} \]

9: Assuming light travels at $1.5 \times 10^8$ m/s in circuit boards and a RAM is 10cm away from the CPU, what is the fastest clock that can access the RAM in a single clock cycle?

\[ \frac{0.1m}{1.5 \times 10^8 \text{m/s}} = 0.67\text{ns} \quad \text{round trip: } 1.33\text{ns} \quad \text{clock: } \frac{1}{1.33\text{ns}} = 750\text{MHz} \]

10: A processor can be safely overclocked to 1.3x its operating frequency, but runs hotter. How much more heat energy must be removed from the processor to maintain the same temperature?

\[ \text{Charge per clock cycle} \rightarrow \text{constant.} \rightarrow 1.3 \text{x clock speed} \]

Heat = Energy so more energy directly sets heat removal.

Heat removed is 1.3x.