1.) (10 points) Create a high-level state machine that initializes a 16x32 register file’s contents to all 0s, beginning the initialization when an input $\text{rst}$ is 1.

2.) (20 points) Create a high-level state machine for a digital bath-water controller. The system has a 3-bit input $\text{ratio}$ indicating the desired ratio of cold water to hot water, and a bit input $\text{on}$ indicating that the water should flow. The system has two 4-bit outputs $\text{hflow}$ and $\text{cflow}$, controlling the hot water flow rate and the cold water flow rate. The sum of these two rates should always equal 16. Your high-level state machine should determine the output values for $\text{hflow}$ and $\text{cflow}$ such that the ratio of hot water to cold water is as close as possible to the desired ratio, while the total flow is always 16. **Hint:** as there are only 8 possible ratios, a reasonable solution may use one state for each ratio.

3.) (25 points) Using the RTL design method, create an RTL design of a reaction timer circuit that measures the time elapsed between the illumination a light and the pressing of a button by a user. The reaction timer has three inputs, a clock input $\text{clk}$, a reset input $\text{rst}$, and a button input $\text{B}$, and three outputs, a light enable output $\text{len}$, a 11-bit reaction time output $\text{rtime}$, and a slow output indicating the user was not fast enough. The reaction timer works as follows. On reset, the reaction timer waits for 10 seconds before illuminating the light by setting $\text{len}$ to 1. The reaction timer then measures the length of time in milliseconds before the user presses the button $\text{B}$, outputting the time as a 11-bit binary number on $\text{rtime}$. If the user did not press the button within 2 seconds (2000 milliseconds), the reaction timer will set the output slow to 1 and output 2000 on $\text{rtime}$. Assume your clock input has a frequency of 1 kHz. Design the datapath to structure, but design the controller to an FSM only. **Hint:** This is a control dominated RTL design problem.

4.) (25 points) Convert the following C-like code, which calculates the greatest common divisor (GCD) of the two 8-bit numbers $a$ and $b$, into a high-level state machine.

```c
Inputs: byte a, byte b, bit go
Outputs: byte gcd, bit done
GCD:
    while(1) {
        while(!go);
        done = 0;
        while ( a != b ) {
            if( a > b ) {
                a = a - b;
            } else {
                b = b - a;
            }
        }
        gcd = a;
        done = 1;
    }
```

5.) (5 points) Calculate the approximate number of DRAM bit storage cells that will fit on an IC with a capacity of 10 million transistors.

6.) (5 points) Calculate the approximate number of SRAM bit storage cells that will fit on an IC with a capacity of 10 million transistors.

7.) (5 points) Summarize the main differences between DRAM and SRAM memories.

8.) (5 points) Summarize the main differences between EEPROM and flash memories.
Extra Credit: (25 points)
Starting with the soda machine dispenser design described in lecture, create a block diagram and high-level state machine for a soda machine dispenser that has a choice of two soda types, and that also provides change to the consumer. A coin detector provides the circuit with a 1-bit input \( c \) that becomes 1 for one clock cycle when a coin is detected, and an 8-bit input \( a \) indicating the coin’s value in cents. Two 8-bit inputs \( s_1 \) and \( s_2 \) indicate the cost of the two soda choices. The user’s soda selection is controlled by two buttons \( b_1 \) and \( b_2 \) that when pushed will output 1 for one clock cycle. If the user has inserted enough change for their selection, the circuit should set either output bit \( d_1 \) or \( d_2 \) to 1 for one clock cycle, causing the selected soda to be dispensed. The soda dispenser circuit should also set an output bit \( c_r \) to 1 for one clock cycle if change is required, and should output the amount of change required using on an 8-bit output \( c_a \). Use the RTL design method to convert the high-level state machine to a controller and a datapath. Design the datapath to structure, but design the controller to the point of an FSM only.