DSE CEO

Memo

To: Design Engineering Team

From: Chris Vessey, CEO, Digital Systems Enterprises

CC: Taymag Corporation Implementation Team Lead

Date: 3/21/2013

Re: Specifications for new base model washing machine

Folks,

We have just received the specifications from the TayMag Corporation for the new washing machine they are proposing to release in the third quarter of next year.

This model is a very basic system, configured for multivoltage and non-stable electrical environments as are found in developing countries.

As a result, the control circuit must be stable and fault-tolerant. Please ensure that the circuit cannot become stalled, as some countries do not use plugs for equipment, but hard-wire it directly to the electrical supply, making it difficult for the consumer to reset. We in North America would consider this an unacceptably dangerous practice, of course.

The chassis manufacturer will supply the power transformer which will drive your circuitry, as well as the two-mode motor assembly, sensors, valve actuator assemblies and signalling hardware. You will supply the interconnects for these devices as pin assignments in your FPGA or CPLD designs, clearly designated with **useful signal names** so that they will know how to lay out the socket connections for the chip.

Please note: the lid safety sensor is currently optional, as our target markets are largely non-North American. As of this date, it is not required in any target country, but *could* be included as a switch-configurable option (that is, a switch would enable the lid sensor processing, and the actual lid sensor switch would then drive a safety cutoff function.)

The manufacturer needs this chip ***in two weeks***, so you will have until then to complete your design. There is no amount of work too great for you most dedicated workers, and I know that you will go without food, sleep or contact with your families to ensure this work is done! I expect that you will make DSE proud with your excellent and efficient design and your attention to detail and deadlines.

Of course, failure to meet deadlines or produce a quality product will mean **revocation of lunch and washroom privileges for the remainder of the year, and will result in your sleep time allocation being reduced by half.**

Be happy in your work.

-Vessey

Specifications for the TayMag® Washer Mk I

Three wash water temperature settings: cold, lukewarm, boiling
One rinse water temperature setting: cold
Options: extra rinse cycle
Two motor control lines: one for agitator drive, one for tank spin drive (the motor module contains an automatic transmission shifter which handles the motor control.)

Operation

User selects water temperature and whether or not there is an extra rinse cycle. User presses START.

Tank is filled with COLD (cold water inlet only), LUKEWARM (cold and hot water inlet) or BOILING (hot water inlet only) water until the FILL sensor indicates that the washing tank is full. The valves to control this are, of course, electrically actuated. All valves are then returned to the off state.

Tank AGITATOR is then turned on. A 10-minute WASH_ACTIVE timer is activated, which signals WASH_DONE when complete. AGITATOR is deactivated.

Tank DRAIN is then activated to cause the drain pump to drain the tank, until the EMPTY sensor indicates that the tank is now empty of water (with the exception of that in the clothes).

Tank is then filled with COLD water until the FILL sensor indicates that the tank is full. Tank AGITATOR is then turned on. A 5-minute RINSE_ACTIVE timer is activated, which signals RINSE_DONE when complete. AGITATOR is turned off.

Tank DRAIN is then activated to drain the tank, until the EMPTY sensor indicates that the tank is now empty of water (with the exception of that in the clothes).

Tank SPIN is then turned on. A 2-minute SPIN_ACTIVE timer is activated, which signals SPIN_DONE when complete. Tank SPIN is then turned off. Pumping is not required from spinning, since the quantity and rate of water flow is small.

If the EXTRA rinse cycle control was activated, then the tank is re-filled with cold water and a second rinse cycle takes place, spinning when complete.

At the end of the wash, an ALERT music box is activated to signal to the user that the wash is done. It plays Beethoven's "Ode to Joy"

Optional Safety Features (not required in some countries)

If, at any time when the AGITATOR or SPIN motors are active, the LID is opened, the AGITATOR or SPIN motor is temporarily shut off, and any active timer is put into PAUSE.

Auxiliary Components

Certain auxiliary circuit components are being manufactured in Taiwan. As such, you will not need to create these.

Wash Timer [10 minute]	Inputs: start_wash, pause_wash	Outputs: wash_active, wash_done
Rinse Timer [5 minute]	Inputs: start_rinse, pause_rinse	Outputs: rinse_active, rinse_done
Spin Timer [2 minute]	Inputs: start_spin, pause_spin	Outputs: spin_active, spin_done
Musical Alert	Inputs: alert	Outputs: *none (only sound!)*

Recommendations

The timer modules can be assumed (that is, you'll use switches on the Altera simulator board to simulate their outputs, which are inputs to your circuit, and a pulser for the clock), or you can create them by using cascaded frequency dividers (see my counter example.) You could have the timers count down on the LED display, if you wish. You can MUX the outputs to provide the timer on the same digits, if you wish. Obviously, you won't run them for minutes; seconds would be acceptable for test purposes.

How much detail you want to put into this project depends on you and your group (four people per group, **maximum –** but you may work in groups of 2 or 3, if you prefer.)

I would expect to see:

1. An ASM chart representing the circuit.
2. Logic diagrams for any combinational circuitry required to drive the states and outputs.
3. Your completed design in Quartus II (using either block diagram or Verilog, your choice).

I would be pleased to see you happily implement this on a DE-1 board. **Remember, always ask yourself: "Is this good for the *company*?"**

Some meta-thoughts on the TayMag Mk I Washer Design . . .

Ok, we can dispense with the formal description stuff and consider some real points here.

I know you'll probably sit and ponder this lab project, and think, *oh krap*.

I also know that, in all probability, you will make this a much harder problem than it is.

It *is* a hard problem, but **not intractable**.

Here are some pointers that you'll find useful.

1. It's a lot easier to modularize if you express things in Verilog instead of BDF files. It's POSSIBLE to make BDF files into modules, but that's a whole other layer of difficulty.

2. You can find out how Quartus expresses things like DFFs in Verilog by simply creating a BDF from one, then using the Create option to create a Verilog file from it. (Hint: it's behavioural.)

3. Not every control line **must** be directly expressed as an output variable of the sequential circuitry. As was stated in lecture, a counter coupled to a decoder allows each counter state (i.e. *stage* in our project) to be represented by an activating line.

4. The central counter which controls wash stages does not have to be connected to a clock crystal – there can be *other signals* which might indicate it's time to move to the next stage.

5. Some components, like the unit that reads the water temperature selector buttons and opens and closes valves, aren't even sequential. They can be modelled as their own little non-sequential units.

6. Would it not be possible to use a MUX to read values from the equipment associated with various stages, and route that value to the master stage counter? Just sayin'.

7. You don't HAVE to simulate the timers. You can use switch or button inputs to simulate their functionality. However, if you WANT to simulate them, you would need to use a really big counter (which becomes a "frequency divider" and take the input off of an appropriate clock pin and divide that puppy up so as to yield a count pulse time of about 1sec. For simulation, 1sec = 1min of real time. Of course, changing it to work on minutes is just a different division factor, right?

I will supply you with a timer prototype. You can then use this as a template for any timers you need. If you need different timer durations, copy the code and modify it. The next page will show you the sample timer.

```
module CountdownTimer10(CLOCK, reset, start, value, pause, active, done);

input CLOCK, reset, start, pause;
output active, done;
output [3:0]value;             // provides the value (max 4 bits) of the
     // current time on the timer – you’ll pass this on to a BCD decoder,
     // and then to a 7-segment display if you want to see the timer count
reg active, done;
reg [3:0]value;
reg enabled;
integer clockedValue;
parameter defaultValue = 10;   // number of time units (seconds) this timer
                               // will run
parameter clockHZ = 50000000;  // we assume you’ll use the 50MHz clock feed
                               // if not, you’ll adjust this accordingly

initial
     begin
            value = defaultValue;
            enabled = 0;
            clockedValue = 0;
            done = 0;
            active = 0;
     end

always @(negedge CLOCK)
     begin
            active = enabled & ~pause;
            if (reset)
            begin
                  value = defaultValue;
                  enabled = 0;
                  clockedValue = 0;
                  done = 0;
                  active = 0;
            end
            if (enabled & ~pause)
                  clockedValue = clockedValue + 1;
            if (clockedValue > clockHZ)
            begin
                  value = value - 1;
                  clockedValue = 0;
            end
            if (value == 0)
            begin
                  enabled = 0;
                  done = 1;
            end
        if (start & ~enabled & ~done)
            begin
                  done = 0;
                  enabled = 1;
                  value = defaultValue;
            end
     end
endmodule
```