g55_stack52 - 52 Slot 6-bit Stack

Group 55
Juliette Regimbal (260657238)
Qingzhou Yang (260687570)

March 20, 2017

1 Circuit Description

The *g55_stack52* circuit has 5 inputs and 4 outputs. There are two 6-bit inputs (`data` and `addr`), one 2-bit input (`mode`), and three 1-bit inputs (`enable, rst`, and `clk`). There are also two 6-bit outputs (`value` and `num`) and two 1-bit outputs (`empty` and `full`).

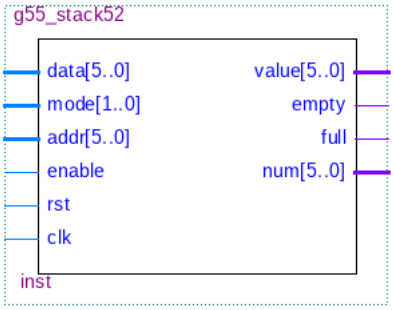

Figure 1: *g55_stack52* Pinout

The stack has four modes of operation: NOP (no operation), INIT, POP, and PUSH. These occur synchronously with the clock signal `clk`, which typically is set to a 50MHz signal. The mode of operation is set with `mode`, with NOP, INIT, POP, and PUSH corresponding to 00, 01, 10, and 11 respectively. The selected operation will only occur when `enable` is high. Otherwise nothing will happen. `data` is an unsigned integer that serves as the data input for

the POP operation. `addr` is an unsigned integer that selects the address of a certain slot in the stack. For example, an input of 000010 would correspond to slot 2. The value stored in the location specified by `addr` is available in `value`. `num` displays the number of entries in the stack up to a maximum of 52. `empty` and `full` are flags for when the stack has no entries and 52 entries respectively. `rst` is an asynchronous reset that sets all entries to 0 and empties the stack.

The NOP operation performs no operation. INIT uses all 52 possible slots in the stack, setting each to their index. For example, the zeroth entry has a value of 0, the first has a value of 1, and so on. POP clears the entry at the location specified by `addr` and shifts the entries with addresses greater than `addr` down by one. PUSH adds a value to the bottom of the stack (zeroth location) specified by the `data` input. All other entries are moved up one space. If the stack is full (52 entries), then the last is lost.

2 Circuit Components

As the *g55_stack52* circuit was done entirely in VHDL, there is no schematic diagram available. Each slot is its own component containing both a muxer (*BUSMUX*) used to handle shifting entries up and down, and a flip flop (*LPM_FF*) that stores the value. Each slot (*g55_stack_slot*) has a default value it initializes to. Within the *g55_stack52* circuit, a counter (*LPM_COUNTER*) is used to keep track of the number of slots used, a muxer (*LPM_MUX*) to route the value of the entry at `addr` to `value`. *g55_pop_enable* is used to handle which slots will shift on POP and PUSH operations.

3 Testing

The *g55_stack52* was tested using both simulations and a physical testbench. A timing simulation was done on the circuit to validate the various operations worked as expected. This test succeeded. After a testbench was created following the specifications below. It had a maximum propogation delay of 27.157 ns along the path from `addr[1]` to `segments_mod[2]`, which is the

second output from the *g55_7_segment_decoder* given the input of `Amod13` from *g55_mod13_v2*. This testbench was programmed onto the FPGA to actually test operation of the stack. Some minor issues occured. One was that sometimes the output states between the debouncer and stack sometimes made unexpected transitions. To solve this, one-hot states were used for the debouncer with any invalid state prompting a transition back to the start (button not pressed, no signal). All operations of the stack were shown to work to the TAs in the lab.

4 Testbench and FPGA

The testbench (*g55_lab3_v2*) contains a *g55_stack52*, two *g55_7_segment_decoder* circuits, a *g55_mod13_v2* circuit, and a *g55_debouncer* circuit. The enable is set to a pushbutton and goes through the debouncer before reaching the actual stack circuit. The *g55_mod13* takes `value` and sends the `Amod13` output to one of the *g55_7_segment_decoder* circuits and `floor13` to the other.

Altogether the testbench uses 989 logic elements as shown by the Flow Summary. With SignalTap II added to the circuit, the number of logic elements increases to 1569 logic elements. These values represent a use of 5% and 8% the total number of available logic elements respectively.