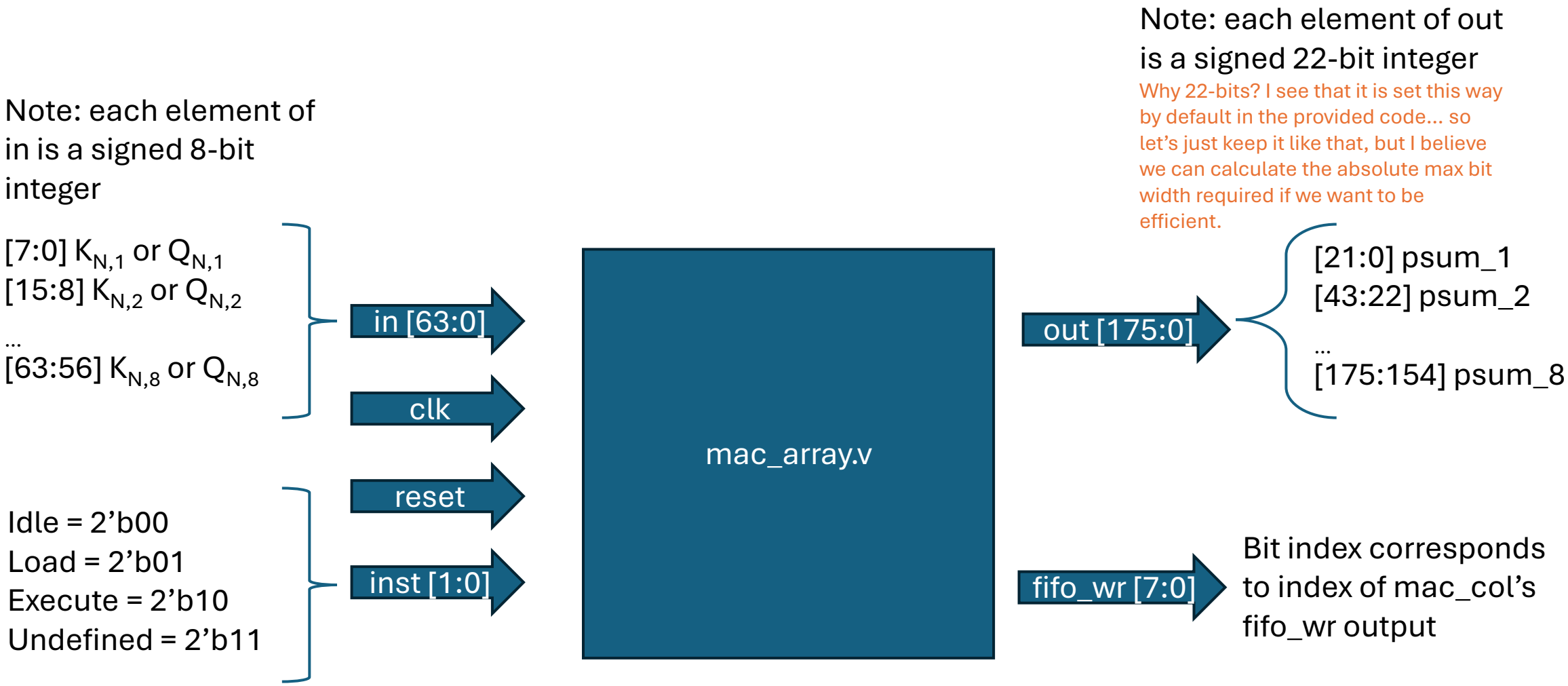

Note: each element of in is a signed 8-bit integer
[7:0] $K_{N,1}$ or $Q_{N,1}$
[15:8] $K_{N,2}$ or $Q_{N,2}$
...
[63:56] $K_{N,8}$ or $Q_{N,8}$
in [63:0]
clk
reset
inst [1:0]
Idle = 2'b00
Load = 2'b01
Execute = 2'b10
Undefined = 2'b11
mac_array.v
out [175:0]
fifo_wr [7:0]
Note: each element of out is a signed 22-bit integer
Why 22-bits? I see that it is set this way by default in the provided code... so let's just keep it like that, but I believe we can calculate the absolute max bit width required if we want to be efficient.
[21:0] psum_1
[43:22] psum_2
...
[175:154] psum_8
Bit index corresponds to index of mac_col's fifo_wr output

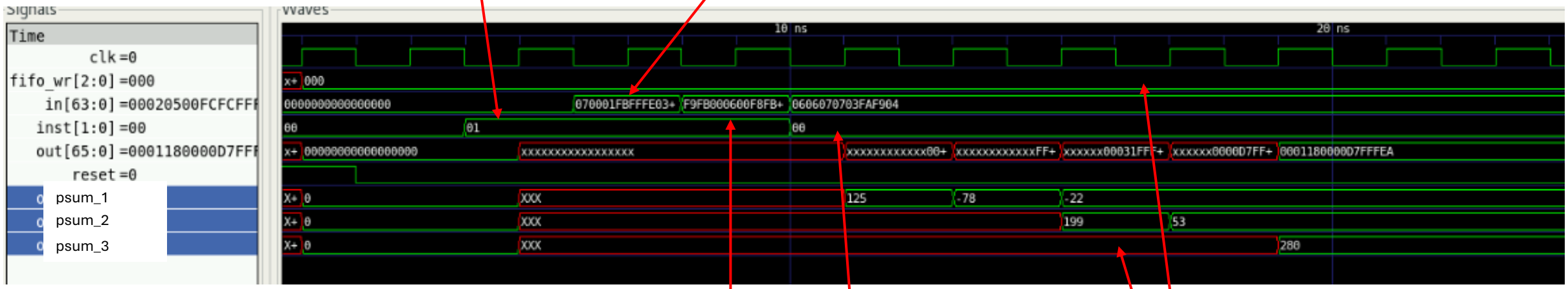
Loading

This example will be for a 3-column mac_array (3 K vectors) and 3 Q vectors

The project requires 8-column mac_array (8 K vectors) and 8 Q vectors

1. Assert load inst

2. Wait one cycle, then load first K vector (this will be stored in the first mac_col's registers



3. Load the next K vector

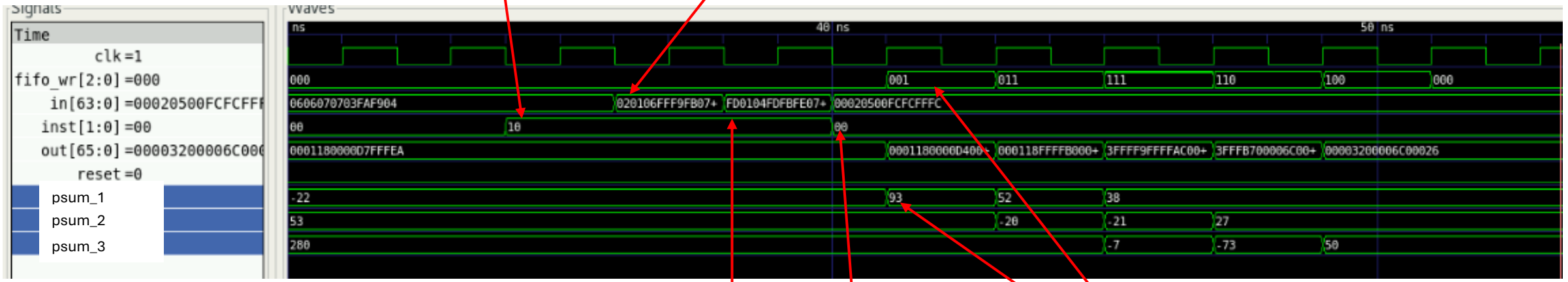
4. Load the last K vector, and on the same cycle de-assert load inst to idle

5. Some values will show up on psums, but since the fifo_wr is not asserted, it doesn't matter

Executing

1. Assert execute inst

2. Wait one cycle, then load first Q vector



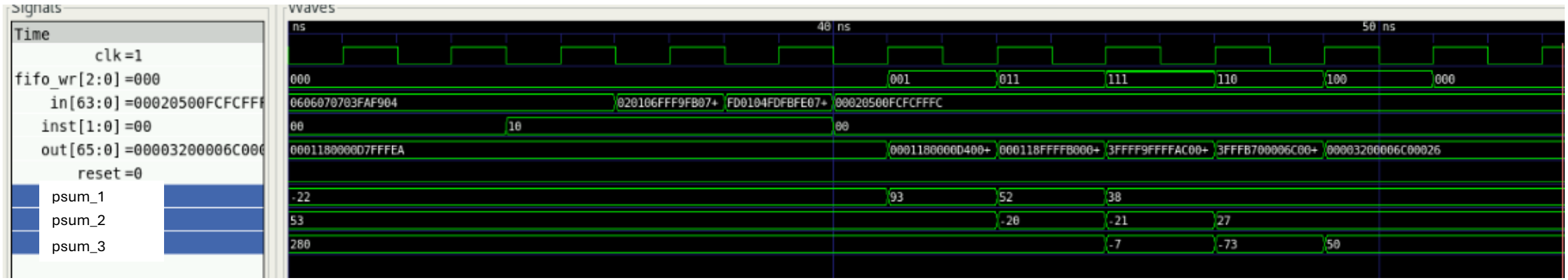
3. Load the next Q vector

4. Load the last Q vector, and on the same cycle de-assert load inst to idle

5. The first mac_col's psum should output 2 cycles after the first Q has been loaded, along with the corresponding fifo_wr being asserted. Why 2 cycles? I added a pipeline to the multiplication + summation already since I anticipate this will be a critical path, but this is arbitrary.

6. Observe all of the psums that are output as the Q vectors move through the columns; note the fifo_wr vector changing

Verification



Once doing irun on the compiled testbench, the log should print the loaded Q and K values from the .txt files. It should also print the predicted psums.

```
##### K data txt reading #####
K#:            0
Q#:            0
Predicted psum:           93
Q#:            1
Predicted psum:           52
Q#:            2
Predicted psum:           38
K#:            1
Q#:            0
Predicted psum:          -20
Q#:            1
Predicted psum:          -21
Q#:            2
Predicted psum:           27
K#:            2
Q#:            0
Predicted psum:           -7
Q#:            1
Predicted psum:          -73
Q#:            2
Predicted psum:           50
Finished.
[h3le@ieng6-ece-01]:mac_array:1027$ wave mac_array_tb.vcd

GTKWave Analyzer v3.3.118 (w)1999-2023 BSI
```

mac_col 1

mac_col 2

mac_col 3

Outputs look good!