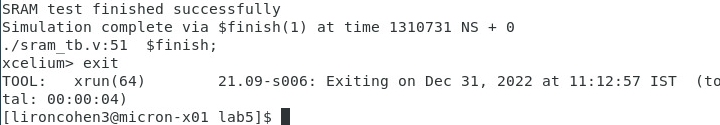
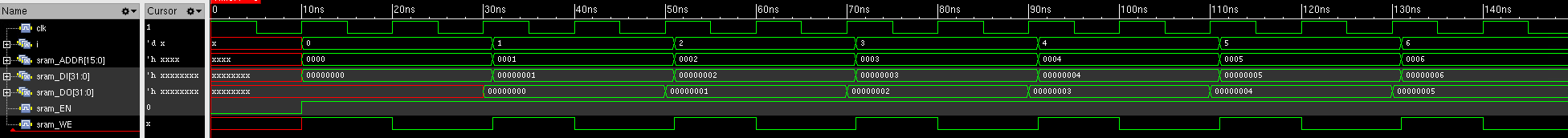
**Lab5 Report**Liron Cohen 207481268  
Yuval Mor 209011543

**Question 1 - SRAM verification**

We filled the missing code in the sram.v module and the sram\_tb.v module.

We ran the xrun command the simulation finished successfully:



The waveform is:

As we can see for example, on the clock rises, and the integer 1 is written to address 0001 (shows in ) and on the clock rises and the output data that is read from address 0001 is the value 1 (shows in ), so the read after write is working and the simulation succeeded.

**Question 2 - processor implementation (no DMA)**

We filled the missing code in the alu.v module, sp.v module and ctl.v module.

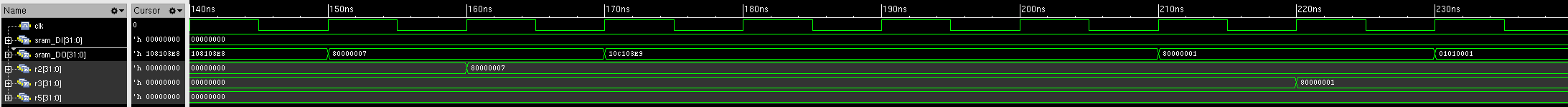
The required files are attached.

**Question 3 - verification #1: example.bin**

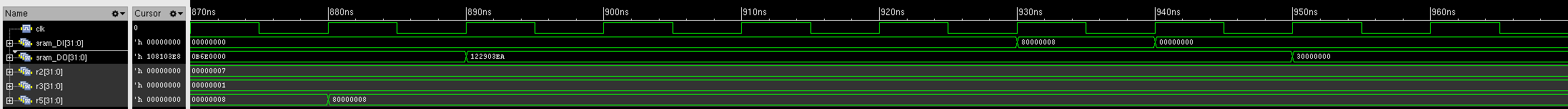
We ran the simulation of the provided example.bin and made sure the traces are matched.

**Question 4 - verification #2: add**

We ran the simulation of our add.bin program and made sure it's trace matched the lab 2 cycle trace.

The relevant waveforms:

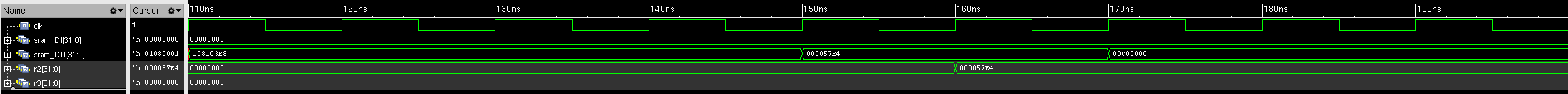
We can see that in , the value of is the first number (80000007) and it is read into in .

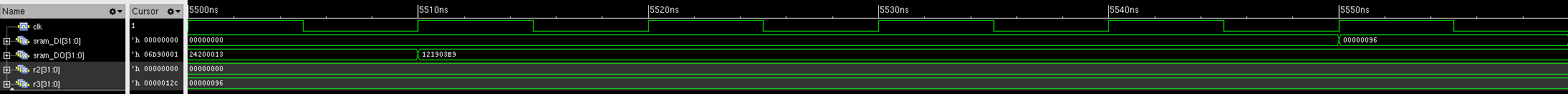
We can also see that in , the value of is the second number (80000001) and it is read into in .

We can see that in , the value of is the result (80000008) and it is written to in .

**Question 5 - verification #3: sqrtq**

We ran the simulation of our sqrtq.bin program and made sure it's trace matched the lab 2 cycle trace.

The relevant waveforms:

We can see that in , the value of is the input to the program () and it is read into in .  
We can see that in , the value of is the result () and it is written to in .

**Question 6 - DMA implementation**

We added the DMA state machine that works in parallel to the main program while using the following added define statements:

`define DMA\_STATE\_IDLE 0

`define DMA\_STATE\_FETCH0 1

`define DMA\_STATE\_FETCH1 2

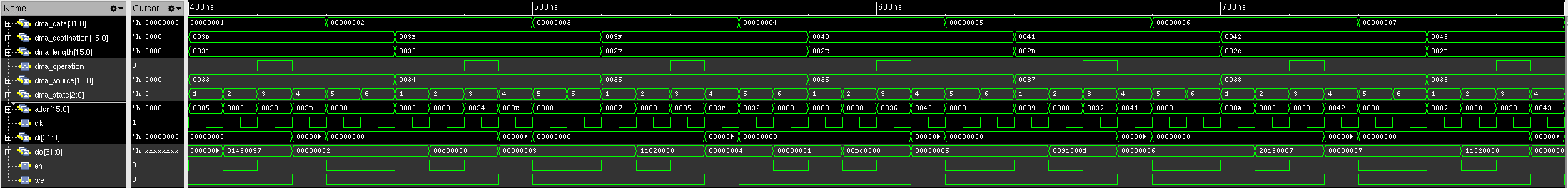
`define DMA\_STATE\_DEC0 3

`define DMA\_STATE\_DEC1 4

`define DMA\_STATE\_EXEC0 5

`define DMA\_STATE\_EXEC1 6

**Question 7 - DMA verification**

We verified our DMA design by running the DMA test program from the previous lab that contains memory access of both the assembly program and the DMA machine. We got the following waveforms: