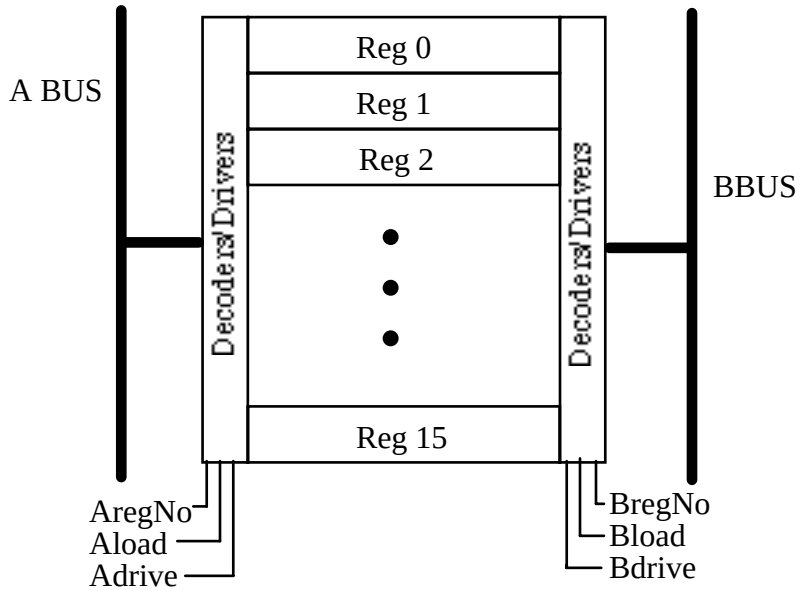
Project Assignment #7 **DUE: Wednesday, May 7th**

1) Copy the file **pr_step7.vhld** from ~degroat/ee762_assign
2) The task is to write a behavioral description of a set of **16** 16-bit dual-ported registers for the datapath that we discussed. Behavioral means that you will use a process statement. The signals used to interface to the two busses are the register number to be accessed, a load signal which tells the register when to latch the data, and a drive signal that tells the register when to place the data on the bus. Note that the register set knows nothing about time, only the value of the control signals. Also remember that the registers are dual ported and can be loaded from and/or drive the busses simultaneously, i.e., for example you can be loading register 0 from the ABUS and at the same time be driving the contents of register 4 onto the BBUS, or driving the contents of register 3 onto the ABUS and the contents of register 11 onto the BBUS. Timing for the load and drive cycles are also shown. You can assume that the controller (not the registers) will insure no conflicts take place, i.e., loading and driving the same register at the same time.

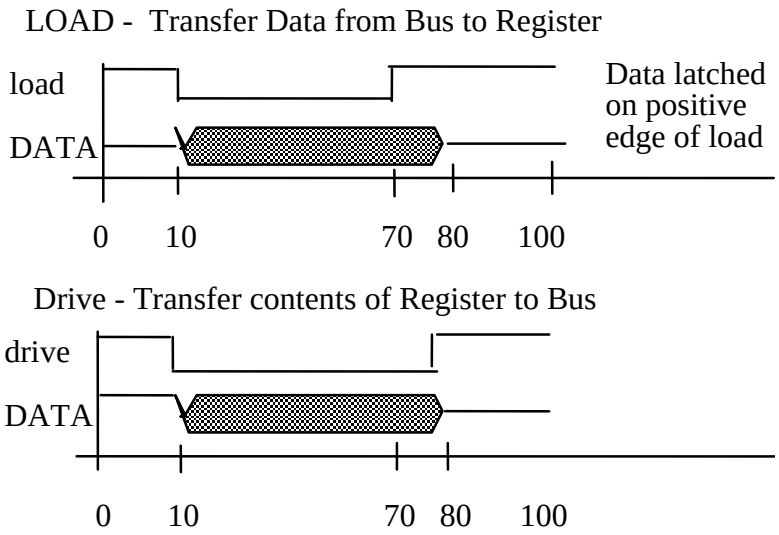
Operation is as follows:

• Loading a value into the register: Bus must be driven by another device which places data on the bus at 10 ns and keeps data on the bus until 80 ns. The value on the bus is latched into the named register on the rising edge of that bus's load signal at 70ns. The processor's controller would insure the timing of the load signal. At this point the timing of the load signal is controlled by the test bench.

• Driving a value on the bus: When the drive signal goes low, place contents of named register on the bus and continue driving the value on the bus until the drive signal goes high.



Protocol for the Bus Control Signals



3) The file has a self checking testbench for testing your entity/architecture of the register set. You are to write a behavioral architecture for the register set using **only** the PROCESS statement and <u>using a variable array to store the contents of the register set.</u> Thus you will use a single process to model the dual ported registers.

Remember that the following BOOLEAN conditions check for:
Level Sensitive:

Signal Low	IF (the_sig = '0') THEN -- valid for BIT or STD_LOGIC
Signal High	IF (the_sig = '1') THEN
Signal Low	IF (the_sig = '0' OR the_sig = 'L') -- for STD_LOGIC
Signal High	IF (the_sig = '1' OR the_sig = 'H')

Edge Sensitive Tests (for TYPE BIT or STD_LOGIC)

High to Low	IF (the_sig = '0' AND the_sig'event) THEN
Low to High	IF (the_sig = '1' AND the_sig'event) THEN

Edge Sensitive Test for STD_LOGIC that is complete

```
High to Low
IF ((the_sig'LAST_VALUE = '1' OR the_sig'LAST_VALUE = 'H') AND
            (the_sig = '0' OR the_sig = 'l') AND the_sig'EVENT) THEN
Low to High
IF ((the_sig'LAST_VALUE = '0' OR the_sig'LAST_VALUE = 'L') AND
            (the_sig = '1' OR the_sig = 'H') AND the_sig'EVENT) THEN
```

4) Turn in:
• a copy of your VHDL code
• results of simulation using zoom->full_size. <u>YOU NEED TO RUN FOR 32 us</u>.
• a list of simulation results
There are .do files for this step, PS7_list.do and PS7_wave.do.