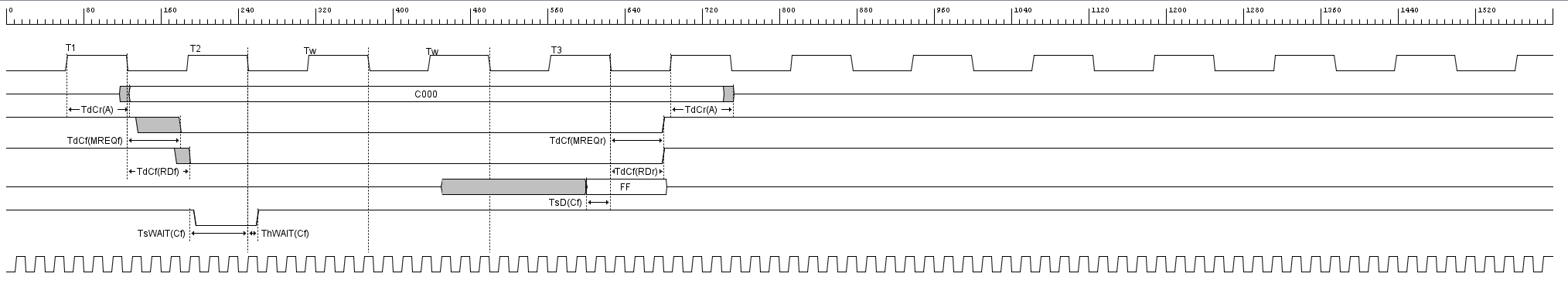
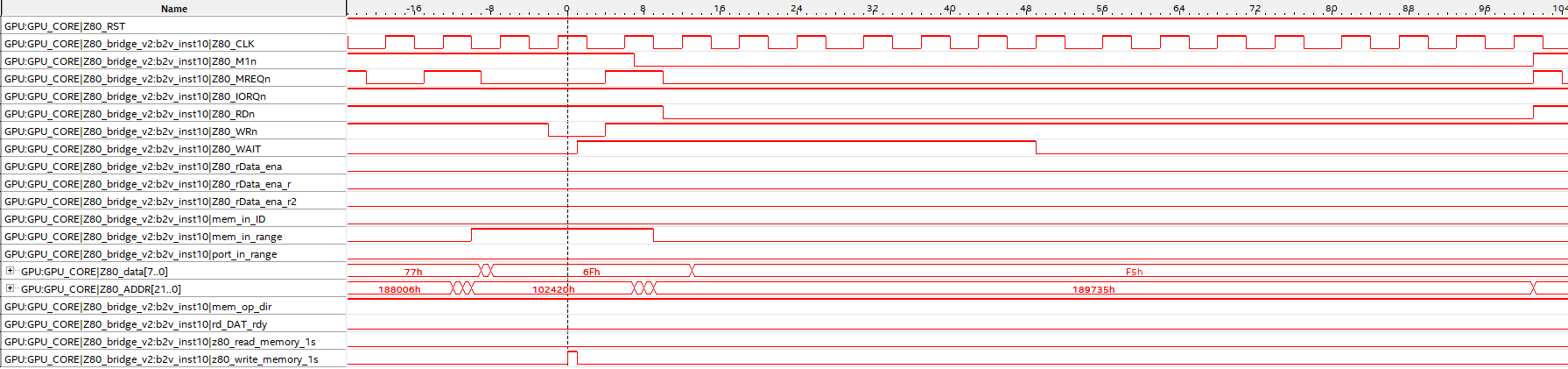
uCOM/DECA WAIT STATE INSERTION

# Memory Read or Write

The chart below shows the timing of memory read or write cycles other than an op code fetch cycle. These cycles are generally three clock periods long unless wait states are requested by memory through the WAIT signal. The MREQ signal and the RD signal are used the same way as in a fetch cycle. In a memory write cycle, the MREQ also becomes active when the address bus is stable so that it can be used directly as a chip enable for dynamic memories. The WR line is active when the data on the data bus is stable so that it can be used directly as a R/W pulse to virtually any type of semiconductor memory. Furthermore, the WR signal goes inactive one-half T state before the address and data bus contents are changed so that the overlap requirements for almost any type of semiconductor memory type is met.

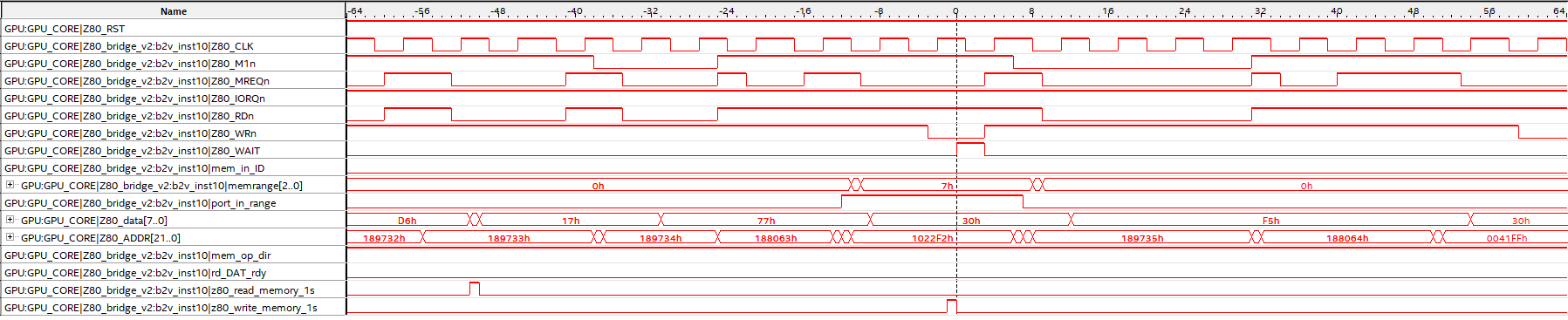
Below is a SignalTap output for a GPU RAM write operation:

This particular memory op was performed with a WAIT register value of 0x07.

One issue raised by the use of SignalTap is single-clock assertions of mem\_in\_range – these seem to be due to random values on the address bus between address value changes, which may be causing spurious RD’s and WR’s to be picked up by the WAIT state engine in Z80\_bridge.

FIX: mem\_in\_range replaced with memrange[2:0] to ignore single-clock pulses and only assert if the address is stable for at least 3 GPU clock cycles.

SignalTap is an extremely powerful piece of software to debug issues in the HDL. From watching live capture of data from the FPGA, I’ve made the following observations:

1. memrange[2:0] goes HIGH when port\_in\_range is also HIGH on occasion. The example below was captured whilst writing the character ‘0’ (ASCII 0x30) to the screen memory:  
   
2. During a WR op, WAIT isn’t asserted in time to pull Z80\_WAIT low in time for T2. This results in the WAIT applying to the next Z80 machine cycle, causing *undocumented behaviourTM*.
3. The Z80\_WAIT line stays low a lot longer than the WAIT signal to the 2N3904 transistor on the uCOM/DECA interface card does. Alternative, faster transistors are required.

So, after some HDL refactoring and examination of the SignalTap traces, it was apparent that bad HDL and the inability to pull WAIT low in time for a WR op was the issue. I’ve refactored the HDL so WAIT is pulled low the instant a memory op is detect (whether RD or WR, it doesn’t wait to find out), but only if the WAIT\_VAL is non-zero. In effect, the user can turn on/off WAIT state insertion in software.

The following table shows the relationship between WAIT\_VAL and the number of inserted WAIT states:

|  |  |
| --- | --- |
| WAIT\_VAL | # WAIT states |
| 0 | 0 |
| 1 | 2 |
| 3 | 3 |
| 5 | 5 |
| 16 | 6 |
| 22 | 7 |
| 30 | 8 |
| 40 | 9 |
| 51 | 10 |
| 63 | 11 |