Yinhao Qian (YIQ25)

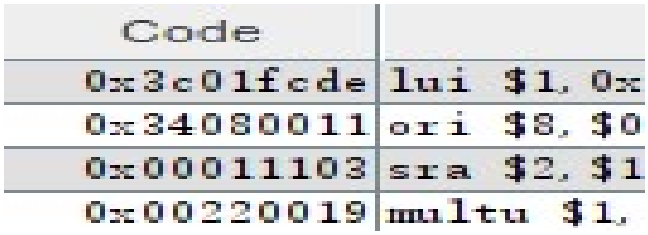
ECE 1195 – Advanced Digital Design

Lab 4 – Multiplier Test Report

To test the multiplier module from our MIPS CPU, we would like to first wrote an simple assembly code that will set 2 registers as multiplier and multiplicand. To keep all instructions under 8 lines, the following GPRs are used:

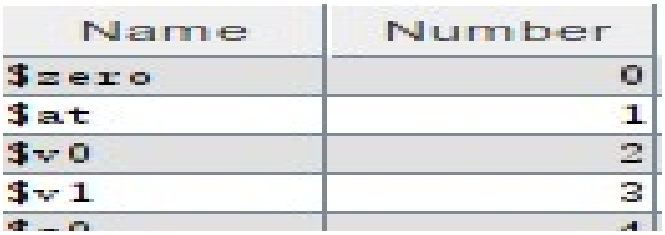
GPR1: Multiplier = 0xFCDE0000 GPR2: Multiplicant = 0xFFCDE000 GPR8: Address Saver Base = 0x00000011

The first three lines initializes the above 3 variables, as we can tell from Figure 1.1 and Figure 1.2



Code	
0x3c01fcde	lui $1, 0x
0x34080011	ori $8, $0
0x00011103	sra $2, $1
0x00220019	multu $1,

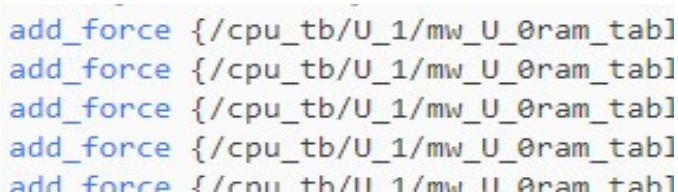
Figure 1.1 – Assembly Code from MARS Simulator



Name	Number
$zero	0
$at	1
$v0	2
$v1	3

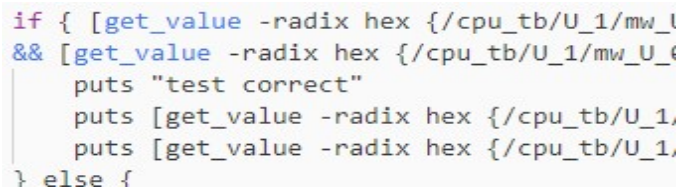
Figure 1.2 – Register Values before Multiplication

Correspondingly, we will need to force the memory table by the instuction code as in Figure 2.1. We will first precalculate the value of (0xFCDE0000 * 0xFFCDE000), which results in 0xACE3001B40000000. By loading the HI and LO into GPR1 and GPR2 and compare the loaded values with our pre-calculated values, the program will report correct if it matches with our expection, demostrated by Figure 2.2. If not, it will report incorrect. The values of HI and LO will also be shown to TCL console as well.



```
add_force {/cpu_tb/U_1/mw_U_0ram_tabl
add_force {/cpu_tb/U_1/mw_U_0ram_tabl
add_force {/cpu_tb/U_1/mw_U_0ram_tabl
add_force {/cpu_tb/U_1/mw_U_0ram_tabl
add_force {/cpu_tb/U_1/mw_U_0ram_tabl
```

Figure 2.1 – Forcing Memory Values in TCL



```
if { [get_value -radix hex {/cpu_tb/U_1/mw_U
&& [get_value -radix hex {/cpu_tb/U_1/mw_U_0
    puts "test correct"
    puts [get_value -radix hex {/cpu_tb/U_1/
    puts [get_value -radix hex {/cpu_tb/U_1/
} else {
```

Figure 2.2– Checking Saved Values in Memory in TCL

After running the simulation , we could also inspect the waveforms from Vivado simulator. Fortunately, the HI and LO values (loaded to memory[8] and memory[9]) matches our pre-calculated product. We can safety draw the conclusion that our multiplier module is working as intended.

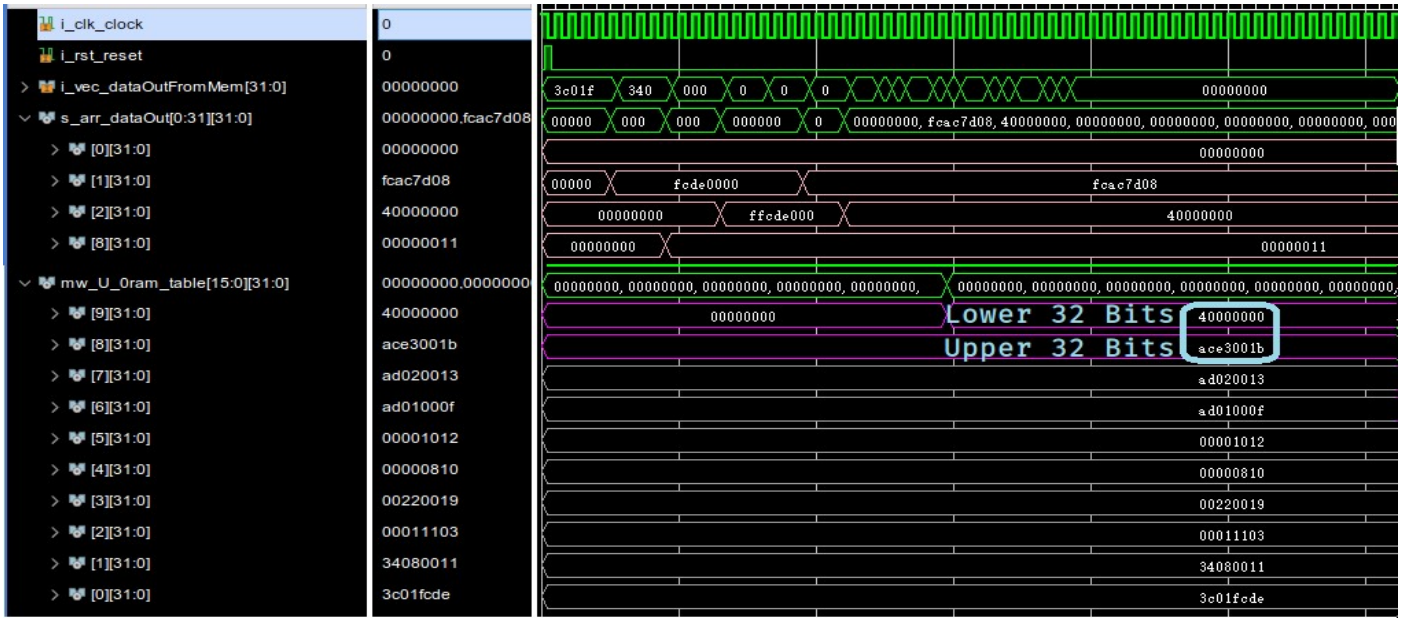


Figure 3 – Waveforms from Vivado Simulator