Lab 4-1 (exmem-FIR) Report

Name: 陳揚哲/李柏辰/楊正宇

Overview

In this lab, we implement the Caravel SoC system. This system is roughly composed of three parts, which are management SoC wrapper, user project wrapper and housekeeping. This time, we only need to focus on the interconnect between management SoC and user project as well as the fir firmware code. We will learn how the module in the system operates with user project wrapper and how to control the mprj io signal.

Firmware code

When CPU wants to run a firmware code, it would read the assembly code, which is transfer from firmware code, from spiflash by wishbone bus. But in this lab, in order to increase the speed of accessing data, we accress the code to store at the RAM in user project wrapper first. In this way, CPU can get the code from user project through wishbone bus directly.

In assembly code, it use Booth's Algorithm to implement the multiple. Register a0 and a2 are took to store the partial sum and result of multiple. The multiplicand is stored at register a1.

The execution starts at 38000008, it checks the lowest bit is whether 0 or 1 and stores the result at a3. If it's 1, the value at a2 would be added to a0. Otherwise, the order jumps to 3800014 and perform an unsighned right shift on a1 by 1 bit. Then, a2 perform a left shift by 1 bit. At 3800001c, a1 would be check whether it's 0 or not. If it's 0, the calculation finishes. Otherwise, jumping back to 38000008 and continuing the iteration.

```
Disassembly of section .mprjram:

38000000 <__mulsi3>:
38000000: 00050613          mv  a2,a0
38000004: 00000513          li  a0,0
38000008: 0015f693          andi  a3,a1,1
3800000c: 00068463          beqz  a3,38000014 <__mulsi3+0x14>
38000010: 00c50533          add a0,a0,a2
38000014: 0015d593          srli  a1,a1,0x1
38000018: 00161613          slli  a2,a2,0x1
3800001c: fe0596e3          bnez  a1,38000008 <__mulsi3+0x8>
38000020: 00008067          ret
```

As shown below, we can see that the address allocate for user project starts at 0x38000000. In the word, the place where the firmware code stored at starts at here.

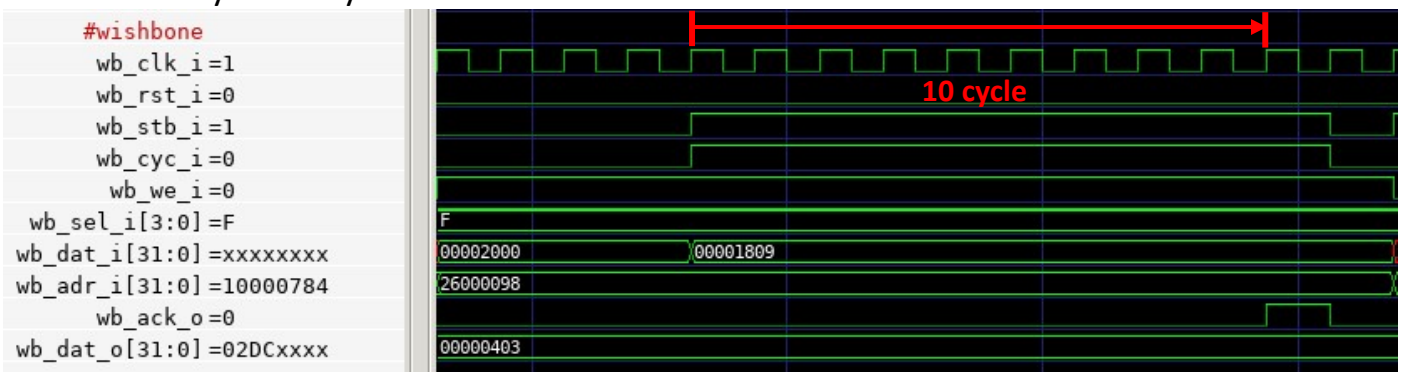
```
MEMORY {
    vexriscv_debug : ORIGIN = 0xf00f0000, LENGTH = 0x00000100
    dff : ORIGIN = 0x00000000, LENGTH = 0x00000400
    dff2 : ORIGIN = 0x00000400, LENGTH = 0x00000200
    flash : ORIGIN = 0x10000000, LENGTH = 0x01000000
    mprj : ORIGIN = 0x30000000, LENGTH = 0x00100000
    mprjram : ORIGIN = 0x38000000, LENGTH = 0x00400000
    hk : ORIGIN = 0x26000000, LENGTH = 0x00100000
    csr : ORIGIN = 0xf0000000, LENGTH = 0x00010000
}
```

At the end of our assembly code, we can find that the last instruction executes at 380001f8. Therefore, the address allocated to the fir firmware code from 38000000 to 380001f8 in the bram of user project. It occupies 508 byte to storing, it's equivalent to 0.498KB (4064 bit). Those code we stored can just run one time calculation. In the fir.c, it would execute the calculation for 11 time.

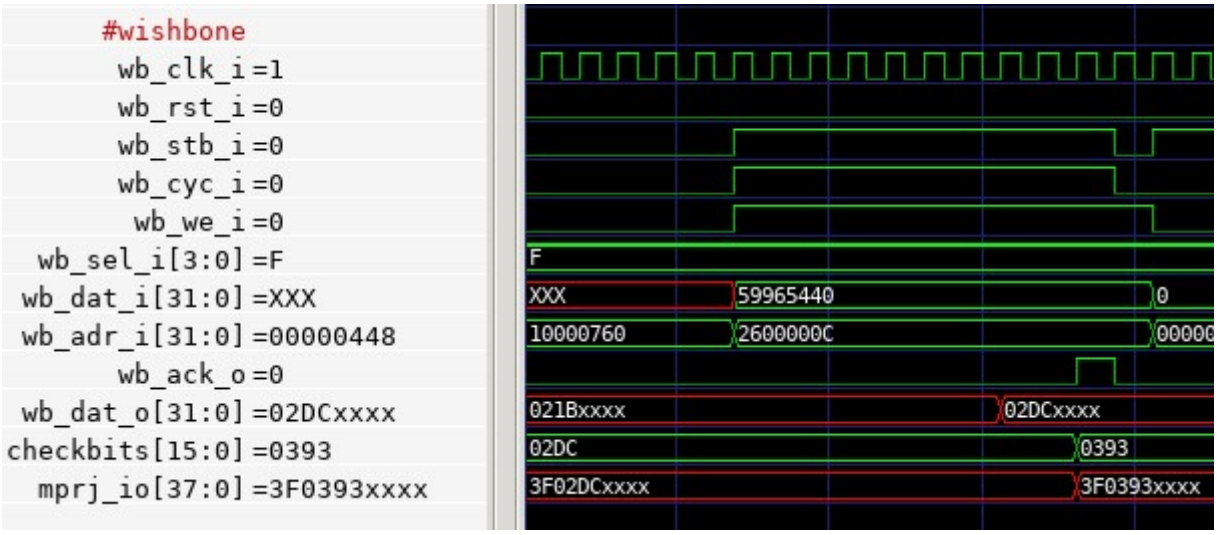
```
380001d4: fef42623            sw  a5,-20(s0)
380001d8: fec42703            lw  a4,-20(s0)
380001dc: 00a00793            li  a5,10
380001e0: fae7dce3            bge a5,a4,38000198 <fir+0x1c>
380001e4: 08800793            li  a5,136
380001e8: 00078513            mv  a0,a5
380001ec: 01c12083            lw  ra,28(sp)
380001f0: 01812403            lw  s0,24(sp)
380001f4: 02010113            addi  sp,sp,32
380001f8: 00008067            ret

Disassembly of section .riscv.attributes:
```

Interface

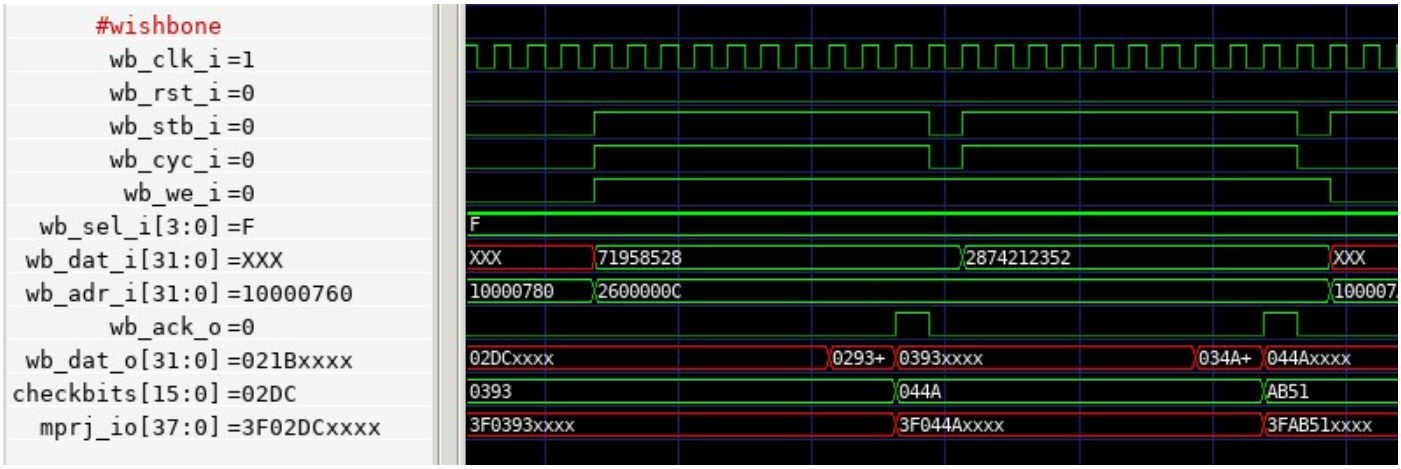
As shown below, when CYC and STB signal assert, ACK signal would be asserted after 10 cycle delay.



According to the figure below, after CPU running the firmware code we stored in bram of user project. The output data would be transferred from CPU through wishbone bus.



After finishing the whole transfer of output data, checkbits would become 0xAB510000. If testbench receive the value from checkbits, it would know CPU finish the whole fir calculation



Synthesis

LUT & FF

Site Type	Used	Fixed	Prohibited	Available	Util%
Slice LUTs*	26	0	0	53200	0.05
LUT as Logic	26	0	0	53200	0.05
LUT as Memory	0	0	0	17400	0.00
Slice Registers	17	0	0	106400	0.02
Register as Flip Flop	17	0	0	106400	0.02
Register as Latch	0	0	0	106400	0.00
F7 Muxes	0	0	0	26600	0.00
F8 Muxes	0	0	0	13300	0.00

BRAM

2. Memory

Site Type	Used	Fixed	Prohibited	Available	Util%
Block RAM Tile	2	0	0	140	1.43
RAMB36/FIFO*	2	0	0	140	1.43
RAMB36E1 only	2				
RAMB18	0	0	0	280	0.00