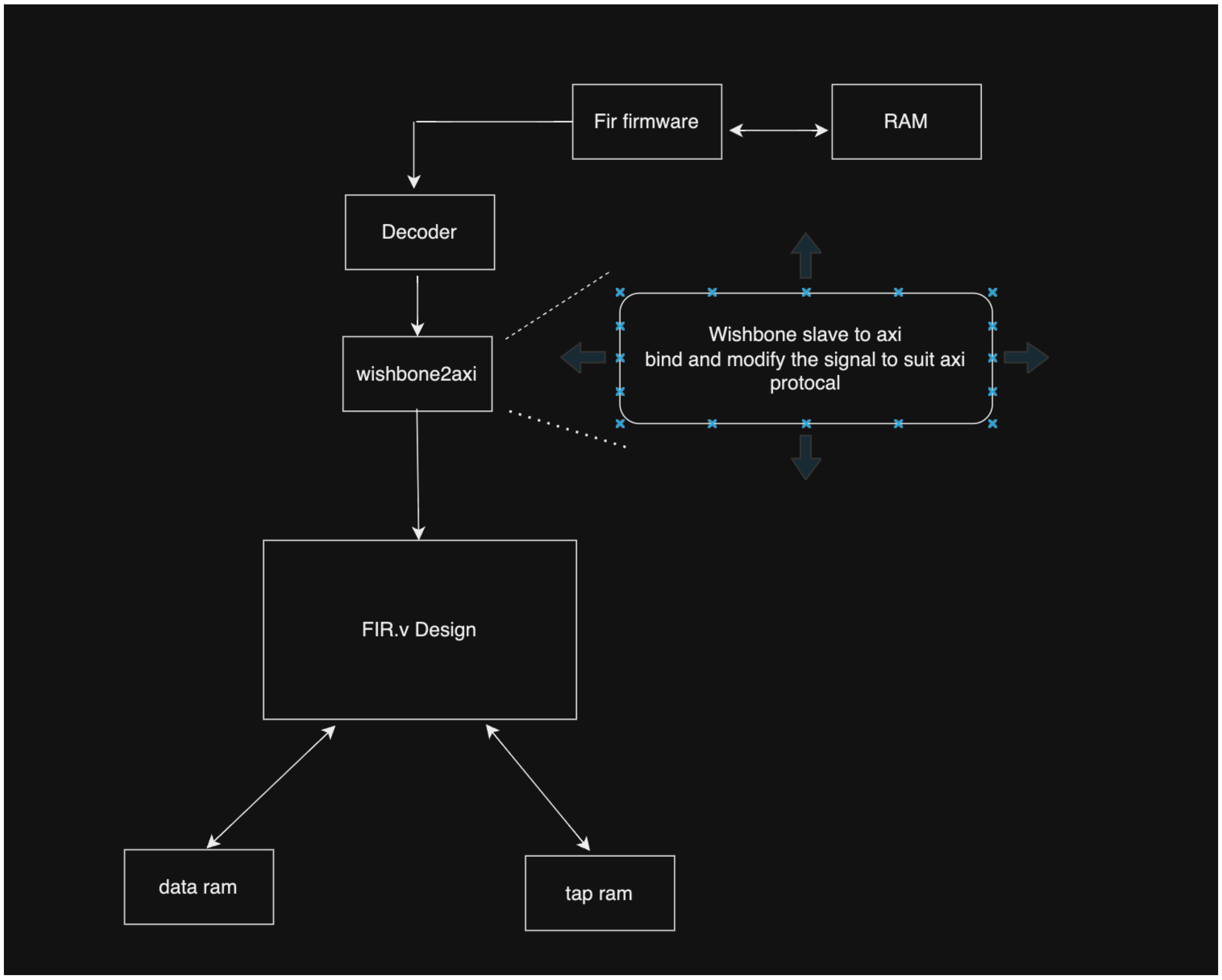
TEAM 8

SOC DESIGN LAB 4-2

TEAM 8

BLOCK DIAGRAM



How firmware handshaking to design

Outputs start with Mark (A5) on mprj[23:16] to notify Testbench to start latencytimer

RISC-V sends X[n] to FIR

RISC-V receives Y[n] from FIR

When transfer finish, write final Y and EndMark to record the latency

Testbench will finally check correctness by checking mprj[31:24], and print out the latency

Interface protocol between different Design

The interface between firmware and user project is through wishbone. The Wishbone protocol also defines a number of control signals, which are used to coordinate the data transfer between the bus master and the bus slave. signals include : Address ,Write , Data, Clock.
And the functionality of testbench is combined them together to a simulation unit so that we can utilize caravel platform to transfer data and control signals.

What is the FIR engine theoretical throughput, i.e. data rate? Actually measured throughput?

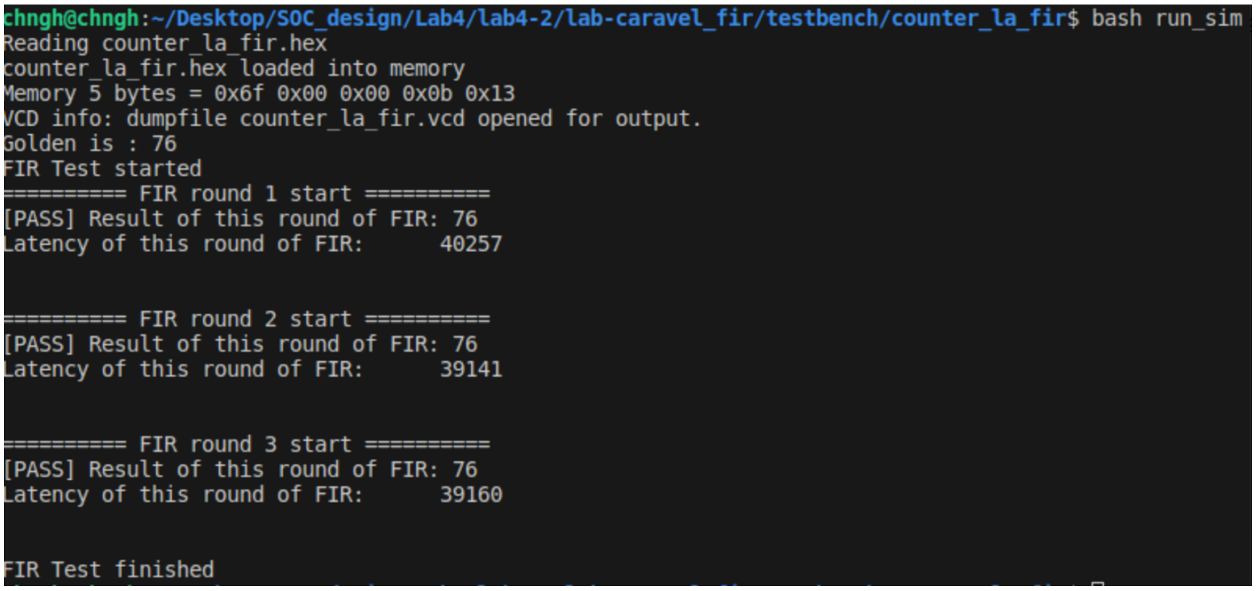
Data rate :
In the design the output is 3467us - 3441us = 26us per output which means that in every output of fir design 26us generate an output.

Throughput:
The definition of throughput is total workload in every time unit
Therefore, from perspective of output 692us-324us=368us having 3 outputs. 3/368us is equal to 0.00833*10^9 = 8.33*10^6 is the throughput.

Can you suggest other method to improve the performance?

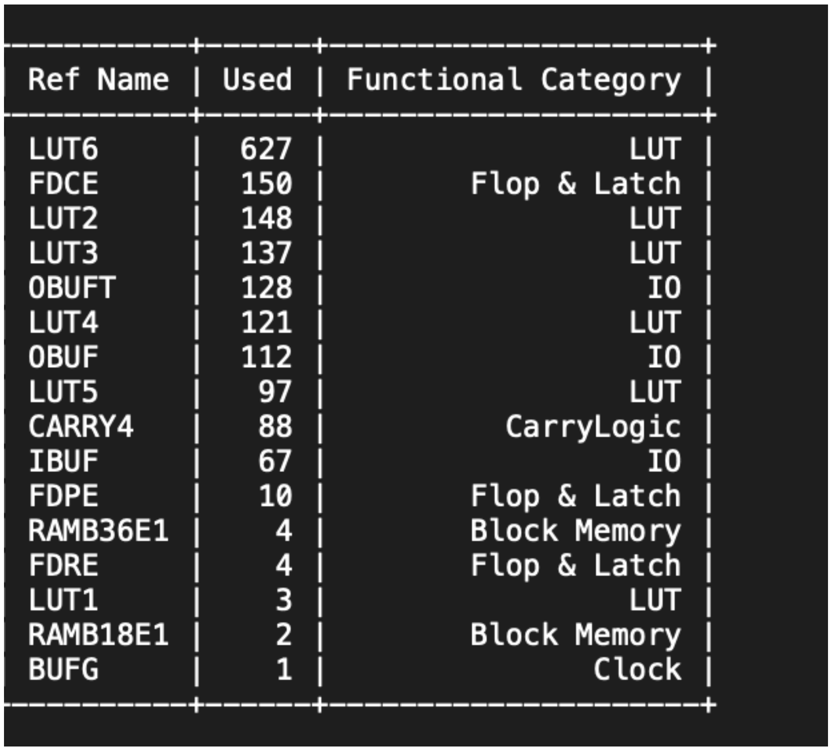
In this project, especially in firmware, have significant effect on caravel soc.
Instead of define function outside the main function writing the function ex. input output
Into fir function can greatly reduce the latency.

Simulation log:

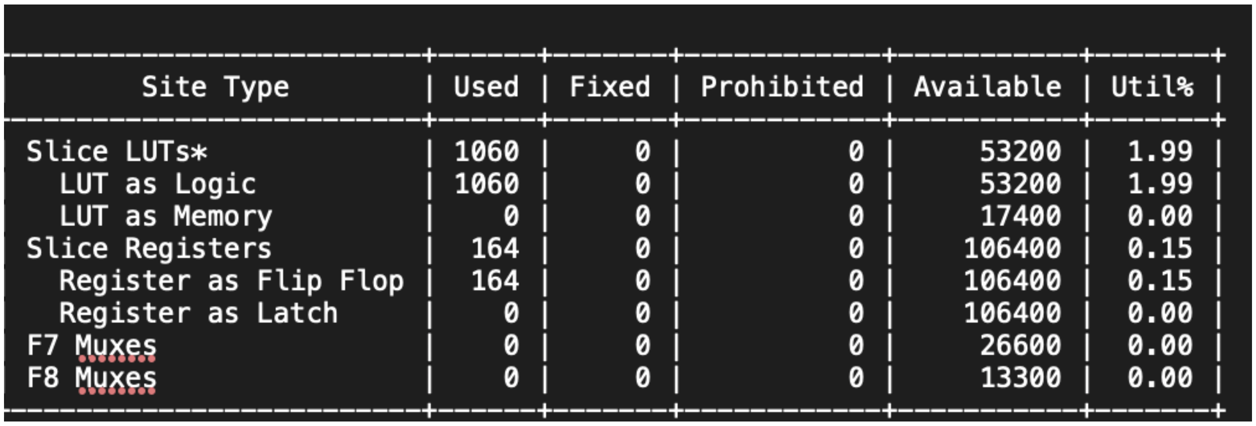


```
chngh@chngh:~/Desktop/SOC_design/Lab4/lab4-2/lab-caravel_fir/testbench/counter_la_fir$ bash run_sim
Reading counter_la_fir.hex
counter_la_fir.hex loaded into memory
Memory 5 bytes = 0x6f 0x00 0x00 0x0b 0x13
VCD info: dumpfile counter_la_fir.vcd opened for output.
Golden is : 76
FIR Test started
========== FIR round 1 start ==========
[PASS] Result of this round of FIR: 76
Latency of this round of FIR:      40257


========== FIR round 2 start ==========
[PASS] Result of this round of FIR: 76
Latency of this round of FIR:      39141


========== FIR round 3 start ==========
[PASS] Result of this round of FIR: 76
Latency of this round of FIR:      39160


FIR Test finished
```

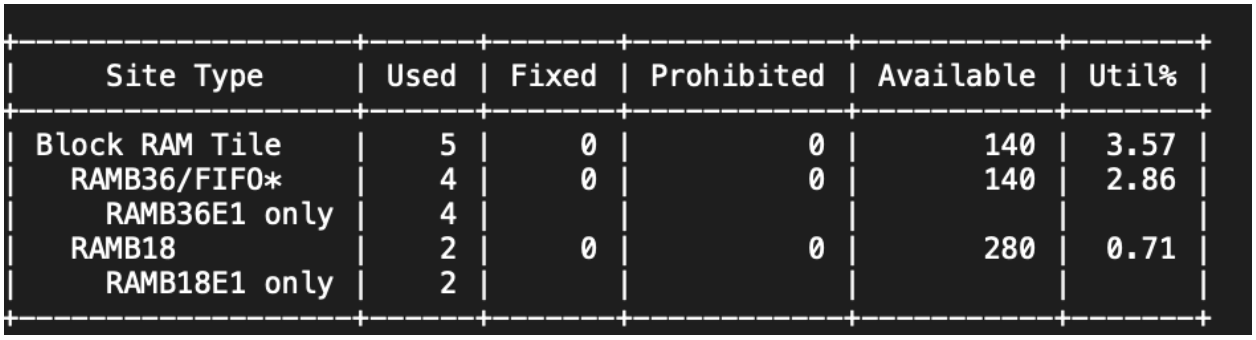
Utilization result:



```
+----------+------+---------------------+
| Ref Name | Used | Functional Category |
+----------+------+---------------------+
| LUT6     |  627 |                 LUT |
| FDCE     |  150 |        Flop & Latch |
| LUT2     |  148 |                 LUT |
| LUT3     |  137 |                 LUT |
| OBUFT    |  128 |                  IO |
| LUT4     |  121 |                 LUT |
| OBUF     |  112 |                  IO |
| LUT5     |   97 |                 LUT |
| CARRY4   |   88 |          CarryLogic |
| IBUF     |   67 |                  IO |
| FDPE     |   10 |        Flop & Latch |
| RAMB36E1 |    4 |        Block Memory |
| FDRE     |    4 |        Flop & Latch |
| LUT1     |    3 |                 LUT |
| RAMB18E1 |    2 |        Block Memory |
| BUFG     |    1 |               Clock |
+----------+------+---------------------+
```



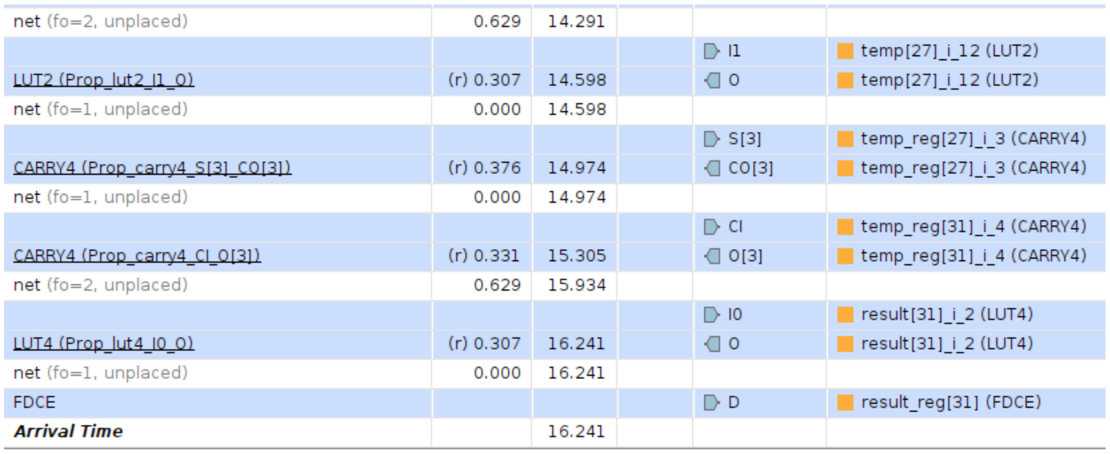
```
+-------------------------+------+-------+------------+-----------+-------+
|        Site Type        | Used | Fixed | Prohibited | Available | Util% |
+-------------------------+------+-------+------------+-----------+-------+
| Slice LUTs*             | 1060 |     0 |          0 |     53200 |  1.99 |
|   LUT as Logic          | 1060 |     0 |          0 |     53200 |  1.99 |
|   LUT as Memory         |    0 |     0 |          0 |     17400 |  0.00 |
| Slice Registers         |  164 |     0 |          0 |    106400 |  0.15 |
|   Register as Flip Flop |  164 |     0 |          0 |    106400 |  0.15 |
|   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 |
+-------------------------+------+-------+------------+-----------+-------+
```



```
+--------------------+------+-------+------------+-----------+-------+
|     Site Type      | Used | Fixed | Prohibited | Available | Util% |
+--------------------+------+-------+------------+-----------+-------+
| Block RAM Tile     |    5 |     0 |          0 |       140 |  3.57 |
|   RAMB36/FIFO*     |    4 |     0 |          0 |       140 |  2.86 |
|     RAMB36E1 only  |    4 |       |            |           |       |
|   RAMB18           |    2 |     0 |          0 |       280 |  0.71 |
|     RAMB18E1 only  |    2 |       |            |           |       |
+--------------------+------+-------+------------+-----------+-------+
```

Timing report:

The maximum delay path is 16.241us.

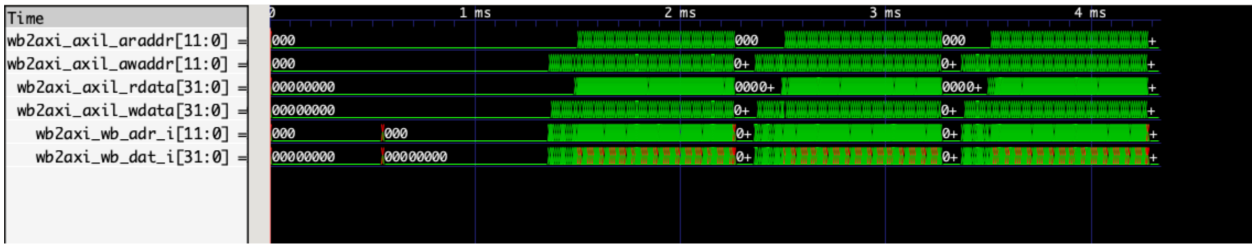


net (fo=2, unplaced)	0.629	14.291			
				I1	temp[27]_i_12 (LUT2)
LUT2 (Prop_lut2_I1_O)	(r) 0.307	14.598		O	temp[27]_i_12 (LUT2)
net (fo=1, unplaced)	0.000	14.598			
				S[3]	temp_reg[27]_i_3 (CARRY4)
CARRY4 (Prop_carry4_S[3]_CO[3])	(r) 0.376	14.974		CO[3]	temp_reg[27]_i_3 (CARRY4)
net (fo=1, unplaced)	0.000	14.974			
				CI	temp_reg[31]_i_4 (CARRY4)
CARRY4 (Prop_carry4_CI_O[3])	(r) 0.331	15.305		O[3]	temp_reg[31]_i_4 (CARRY4)
net (fo=2, unplaced)	0.629	15.934			
				I0	result[31]_i_2 (LUT4)
LUT4 (Prop_lut4_I0_O)	(r) 0.307	16.241		O	result[31]_i_2 (LUT4)
net (fo=1, unplaced)	0.000	16.241			
FDCE				D	result_reg[31] (FDCE)
Arrival Time		16.241			

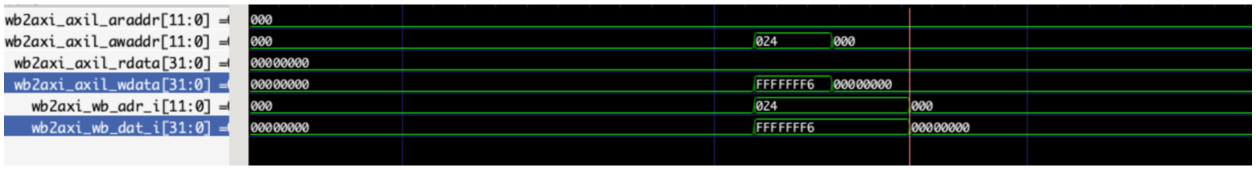
Waveform :

In this design we have 3 sets of data transfer to the fir design and every set have 64 data added into design

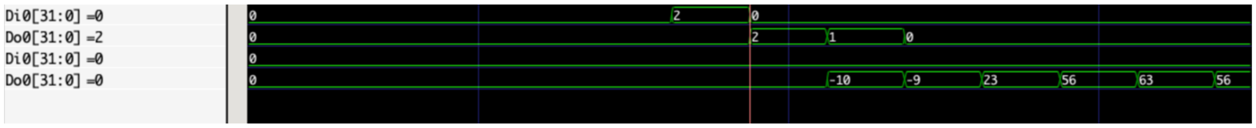
As following we can see 3 set of data



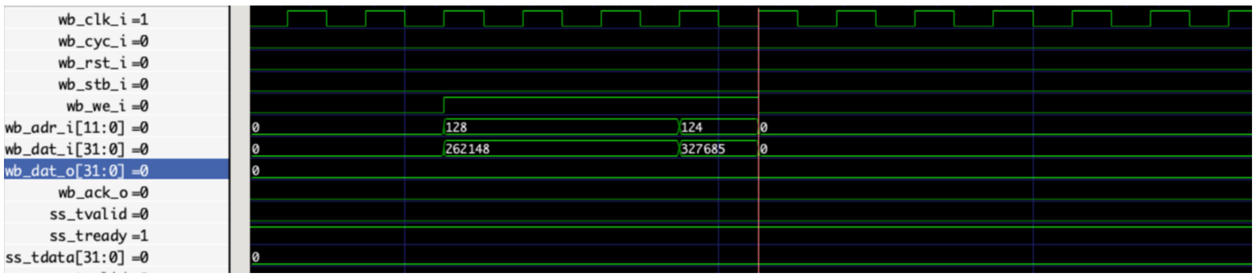
Here is one zoom out of the picture above.



Here is the tap_ram and data_ram waveform. We can see the data is accurately write into the ram that we defined



Wishbone signal from wishbone master to slave and convert to AXI



Review

In this lab, we have learned how to use caravel soc to deploy a platform and its detail.
In lab, writing a robust firmware code is essential because the firmware code can drastically affect the performance. It is also a good practice to have experience like how to use wishbone as protocol to transfer data to user project and use mprj pin to deliver result which is from fir back to RISC CPU.
Although this time we have not yet used logic analyzer that connect to CPU to transfer. I will try to use it next time as a challenge.

Github link:

https://github.com/Charlee0207/SOC-Design/tree/main/Lab4-2