National Tsing Hua University
Department of Electrical Engineering
EE429200 IC Design Laboratory, Fall 2021

Lab 08: Coding for Synthesis

Assigned on Nov 4, 2021

Due day in Nov 11, 2021

Objective

In this lab, you will learn:

1. Implementation and analysis different hardware architecture of digital filter.

Demo checklist:

- ☐ Show simulation results and performance of the one multiplier and one adder design.
- ☐ Show simulation results and performance of the two parallel architecture.
- ☐ Answer the following questions
    1. Record different hardware architectures (direct, one multiplier and one adder, pipeline, parallel).
    2. Analyze the advantage or disadvantage of each hardware architecture.
    3. Check the ***resources report*** in synthesis report, analyze the adder/multiplier which are used in both of your designs. Are there any resource sharing in your design?

Environment Setup

Copy lab file packages from ee4292. Decompress the package and enter it. You can check the file list in Appendix.

```
$ cp ~ee4292/iclab2021/lab08.zip .
$ unzip lab08.zip
$ cd lab08/
```

Note: please run simulation in the ***sim*** directory to maintain a fine data management.

Description

In lab6, we have implemented the filter for ECG application. And we are now going to further explore the design space of the filter architecture. As shown in the optional action item in lab8, there are many possible architectures to improve the performance of the filter. You are going to implement a **one multiplier and one adder architecture** and a **two-parallel architecture** of the filter in this lab. The corresponding architecture diagram are shown in Fig 1 and 3. You can start from the fir1.v which is the finished

version of lab6 or your own fir1.v. Since the output behavior of parallel architecture are different from last lab, you may also need to modify the state machine in this lab.

Action Items

I. One multiplier and one adder architecture.

1. Please use only one multiplier and one adder architecture as shown in figure 1. You can refer to figure 2 to check your function correctness.

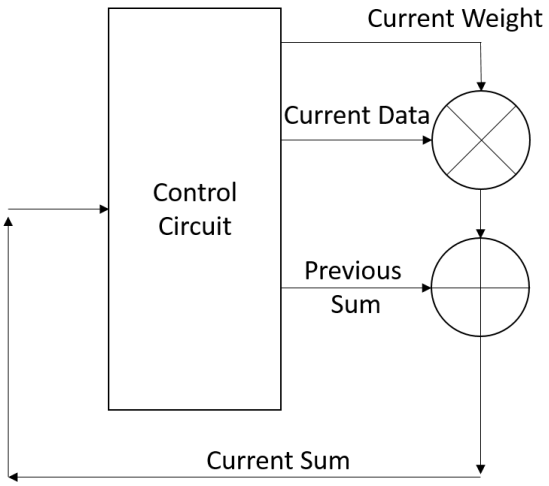

Figure 1. one multiplier and one adder architecture.

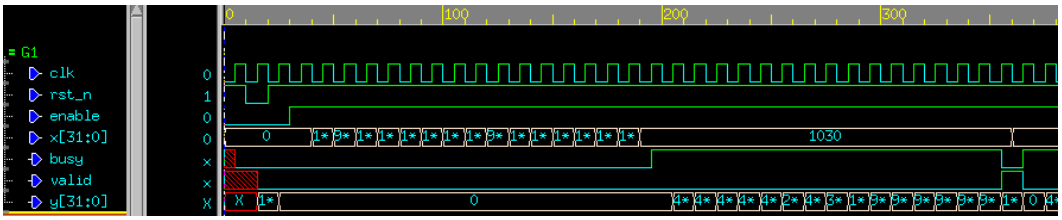

Figure 2. Waveform of one multiplier and one adder architecture.

2. Record your synthesis result.

Area (um^2)	Timing (ns)	Cycles (#)	Performance

II. Parallel architecture.

1. Parallel structure is a MIMO system, multiple inputs can be processed in one cycle, as a result, the overall throughput is improved. Figure 3 is parallel architecture, it can be expressed as below:

$$y[2n] = \sum_{i=0}^{15} x[2n-i] * a_i$$

$$y[2n+1] = \sum_{i=0}^{15} x[2n-i+1] * a_i$$

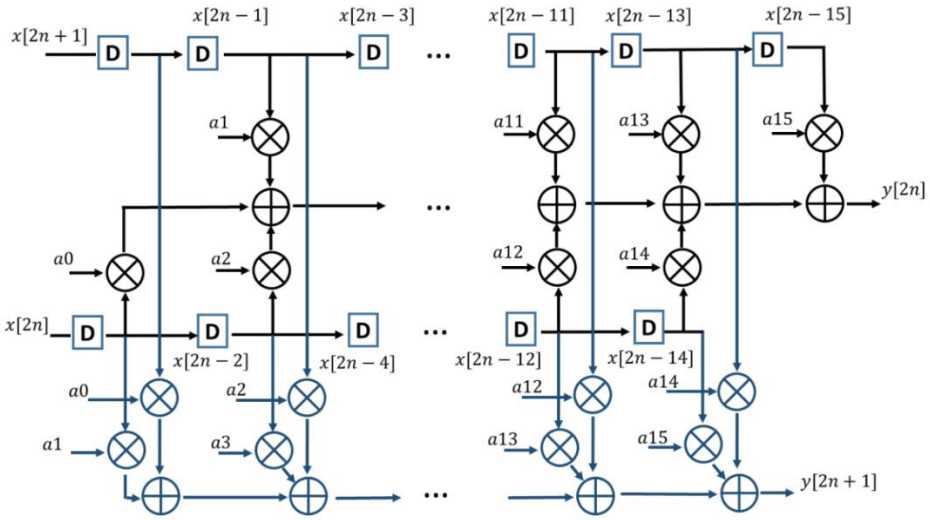

Figure 3. Parallel structure.

2. The circuit I/O should be modified to feed and receive multiple data in one cycle.

I/O name	Number of bits	Description
clk	**1**	**Clock input**
rst_n	**1**	**Active low reset**
enable	**1**	**Module enable**
x0	**N**	**ECG data input**
x1	**N**	**ECG data input**
busy	**1**	**Module busy**
valid	**1**	**Output data valid**
y0	**N**	**Processed data output**
y1	**N**	**Processed data output**

3. Figure 4 is the waveform of parallel architecture, you can refer to it to check your function correctness.

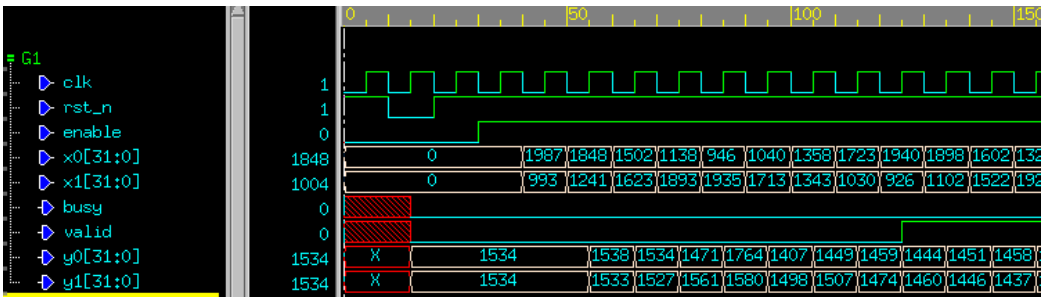

Figure 4. Waveform of parallel architecture.

4. The parallel architecture is obvious a trade-off between power and area, so except for comparing the performance index, try to synthesis the parallel architecture with double clock cycle, and see how much power is saved (with the same throughput as the original architecture). Record your synthesis result.

Area (um^2)	Timing (ns)	Cycles (#)	Performance

Appendix

Directory	**Filename**	**Description**
hdl	fir1.v	Your design here
hdl	fir1_parallel.v	Your design here
sim	fir1_parallel_test.v	Testbench
sim	fir1_test.v	Testbench
sim	ecg.csv	Test pattern
sim	data_truth.csv	Golden response
syn	0_readfile.tcl	Synthesis scripts
syn	1_setting.tcl	Synthesis scripts
syn	2_compile.tcl	Synthesis scripts
syn	3_report.tcl	Synthesis scripts
syn	synthesis.tcl	Synthesis scripts
syn	run_dc.bat	Synthesis command
syn	.synopsys_dc.setup	Synthesis setup file