Timing report

Mihailo Isakov, Boston University

April 22, 2017

Abstract

In this timing report, I test the performance of our backpropagation RTL design.

1 Introduction

In this report, I test out:

1. Functional level performance and latency
2. Synthesis results of the design

2 Functional level timing

For this experiment, I tested a small backpropagation module training a 1-layer, 4 neuron network, with synthetic inputs. The network does not learn anything meaningful, as I was only interested in the designs performance. The inputs themselves will not affect the behavior or performance of the network itself, as I only looked at the simulation results. From the experiment I wanted to get:

1. The average latency between a feed-forward result input to the backpropagator module, and the resulting weight update
2. The time each module spends in each state (IDLE, CALC, DONE)
3. The bottleneck of the network

Since almost all higher level modules do not have any logic in them, but serve only as wrappers, I broke down the design to the lowest level modules using the valid/ready protocol. Initially, I manually measured the latency of each module or a chain of modules. Due to the error-proneness of such an approach, I wrote a simple testbench that saved the signals I wanted to track (states, valid and ready signals) in a CSV file. From there on I analyzed the files in python, writing small useful scripts for tracking latency, percentage of time spent in a certain state, and a visualization script.

2.1 Results

The latency between the inputs arriving at the backpropagator module and the resulting weight update to be 30 clock cycles. After warming up and filling the pipeline, each module spends 15 cycles between to consecutive valid outputs. By analyzing the diagram, we can see that the bottleneck is the second vector_add module, located in the weight_controller submodule of the backpropagator. The module is the bottleneck since it adds the weight matrices and the weight updates, but uses the same tiling as the modules operating on the neuron signals, which are 4× shorter. I additionally include the table 2 showing the percentage of time each module spends in each state, along with the average number of cycles it spends between two valid signals.

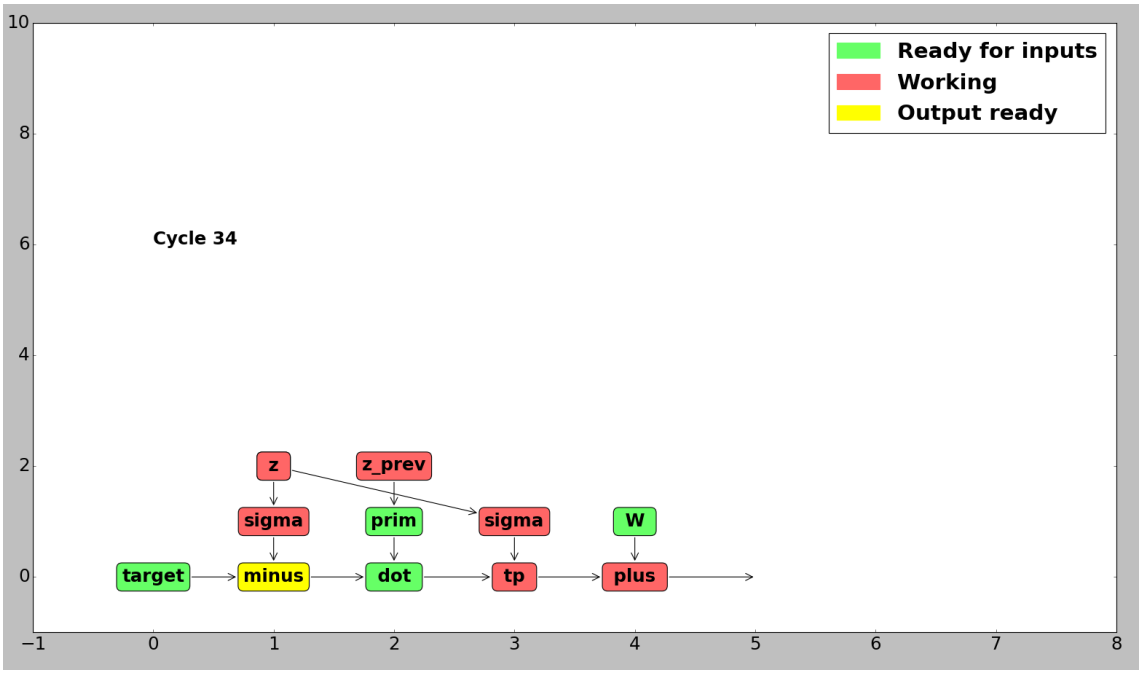

Figure 1: Screenshot of the script for visualizing the states of modules at a certain timestep, the average time between two consecutive outputs valid signals from each module, and average time spent in state. The data for the average time per cycle is updated by clicking of different modules.

Time spent in each state				
Name in diagram	IDLE state %	CALC state %	DONE state %	Cycles between valid
Sigma	13.68	20.29	66.03	14.78
Prim	7.04	20.22	72.74	14.84
Sigma	39.74	20.02	40.24	14.96
Minus	13.95	13.48	72.57	14.82
Dot	14.01	13.41	72.57	14.87
TP	66.43	26.56	7.01	14.98
Plus	40.54	52.85	6.61	15.00

Figure 2: Table showing the time spent in each state of the modules shown on the graph. Modules target, sigma, minus, prim, and dot are in the error_fetcher module of the backpropagator, while the others are in the weight_controller module. As we can see, the plus module spends the most time in the CALC state and is the one determining the maximum update rate of 15 cycles. The script better illustrates how the states change in time.

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	3470	35200	9%
Number of Slice LUTs	2817	17600	16%
Number of fully used LUT-FF pairs	1647	4640	35%
Number of bonded IOBs	360	100	360%
Number of Block RAM/FIFO	5	60	8%
Number of BUFG/BUFGCTRLs	1	32	3%
Number of DSP48E1s	12	80	15%

Figure 3: Synthesis report from Xilinx ISE

3 Design synthesis

I have tried to synthesize the design, and after removing a couple of errors, the synthesis report looks as in figure 3. The large number of IOB wires and an issue with the license server prevents me from moving further with mapping. I extracted the important parts of the synthesis report, and show them here. As we can see from the report, the maximum frequency is 117.860 MHz. This is likely due to the design using blocking statements instead of adder trees for a small part of the matrix-vector multiplication, which I will fix soon. We can also see that the design synthesizes 4 $32x32$ multipliers, which are likely used for indexing. Some refactoring will get rid of the large multipliers. As we can see in figure 3, the design uses only 9% of the registers, and 16% of the LUTs. The design uses 360 IOBs out of the 100 available, preventing the toolchain from continuing.

```
Advanced HDL Synthesis Report

Macro Statistics
# RAMs                                              : 10
 1024x36-bit dual-port block RAM                    : 1
 1024x9-bit dual-port block RAM                     : 8
 2x256-bit dual-port distributed RAM                : 1
# MACs                                               : 12
 4x4-to-8-bit Mult with pre-adder                   : 12
# Multipliers                                        : 34
 10x9-bit multiplier                                : 8
 32x32-bit multiplier                               : 4
 32x4-bit multiplier                                : 8
 4x3-bit multiplier                                 : 12
 4x4-bit multiplier                                 : 2
# Adders/Subtractors                                 : 39
 10-bit subtractor                                  : 2
 16-bit adder                                       : 1
 17-bit adder                                       : 1
 3-bit adder                                        : 7
 32-bit adder                                       : 4
 4-bit adder                                        : 12
 4-bit subtractor                                   : 8
 5-bit adder                                        : 4
# Registers                                          : 3677
 Flip-Flops                                         : 3677
# Comparators                                        : 29
 1-bit comparator equal                             : 1
 3-bit comparator greater                           : 5
 3-bit comparator lessequal                         : 7
 4-bit comparator greater                           : 7
 4-bit comparator lessequal                         : 2
```

```
 5-bit comparator greater                                : 2
 5-bit comparator lessequal                              : 5
# Multiplexers                                           : 7637
 1-bit 2-to-1 multiplexer                                : 6819
 10-bit 2-to-1 multiplexer                               : 32
 11-bit 2-to-1 multiplexer                               : 1
 12-bit 2-to-1 multiplexer                               : 1
 13-bit 2-to-1 multiplexer                               : 1
 14-bit 2-to-1 multiplexer                               : 1
 15-bit 2-to-1 multiplexer                               : 1
 16-bit 2-to-1 multiplexer                               : 237
 19-bit 2-to-1 multiplexer                               : 168
 2-bit 2-to-1 multiplexer                                : 32
 2-bit 4-to-1 multiplexer                                : 1
 3-bit 2-to-1 multiplexer                                : 29
 32-bit 2-to-1 multiplexer                               : 143
 36-bit 2-to-1 multiplexer                               : 5
 4-bit 2-to-1 multiplexer                                : 11
 40-bit 2-to-1 multiplexer                               : 6
 5-bit 2-to-1 multiplexer                                : 13
 6-bit 2-to-1 multiplexer                                : 11
 7-bit 2-to-1 multiplexer                                : 11
 8-bit 2-to-1 multiplexer                                : 11
 9-bit 2-to-1 multiplexer                                : 103
# Logic shifters                                         : 36
 496-bit shifter logical right                           : 8
 54-bit shifter logical right                            : 1
 60-bit shifter logical right                            : 1
 63-bit shifter logical right                            : 8
 70-bit shifter logical right                            : 18
=========================================================================

Final Register Report

Macro Statistics
# Registers                                              : 3470
 Flip-Flops                                              : 3470


=========================================================================
*                            Design Summary
*
=========================================================================
WARNING:Xst:1336 -  (*) More than 100% of Device resources are used

Clock Information:
------------------
+-----------------------------------+-----------------------------+-------+
| Clock Signal                      | Clock buffer(FF name)       | Load  |
+-----------------------------------+-----------------------------+-------+
| clk                               | BUFGP                       | 3525  |
+-----------------------------------+-----------------------------+-------+

Asynchronous Control Signals Information:
-----------------------------------------
+-----------------------------------+-----------------------------------------------------------------------+-
| Control Signal                    | Buffer(FF name)
| Load  |
+-----------------------------------+-----------------------------------------------------------------------+-
```

```
error_propagator/dot/Sh172(XST_GND:G)| NONE(weight_controller/sigma/LUT/Mram_ram1)| 16
|
-----------------------------------------+-----------------------------------------+--
```

Timing Summary:

Speed Grade: -3

Minimum period: 8.485ns (Maximum Frequency: 117.860MHz)
Minimum input arrival time before clock: 1.356ns
Maximum output required time after clock: 0.987ns
Maximum combinational path delay: No path found