

Softmax\_cnn

Hardware Documentation



June 9, 2020

by: Basant Loay

Softmax Activation Function

1. **Block diagram**

A screenshot of a cell phone

Description automatically generatedSoftmax Layer for 1D input:

1. **Each unit Verilog code**

* **exponential.v**

**This module to calculate exponential of floating point input by using Taylor expansion method. It calls module term.v that multiply input number by its powered then multiply by constant by using mul.v modules and calls addition to sum the calculated 6 terms.**

* **division.v**

**The division module based on Newton Raphson method to calculate reciprocal of divisor in reciporical.v.**

* **reciporical.v**

**By using algorithm of Newton Raphson method to calculate reciprocal of Divisor**

**The used algorithm TAKEN FROM:**

1. <https://stackoverflow.com/questions/9220498/newtons-method-for-finding-the-reciprocal-of-a-floating-point-number-for-divisi>
2. <https://en.wikipedia.org/wiki/Division_algorithm#Newton.E2.80.93Raphson_division>

**It calls module Xncalculation.v (THIS MODULE FOR CALCULATING Xn value as X = X \* (2-D\*X) ) then multiply reciprocal (Xn) by Dividend.**

* **maxValue.v**

**This module compares between 2 consequative numbers' exponent in array to get larger, but if two exponents equal, it compares two mantissas. (all values of input array are positive as it is output of exponential module no need to check on sign bit)**

* **Sofmax\_Layer.v**

This module take 1D input array and applies **exponential.v,** then apply **Accumlator1.v** on the exponential array to get summation, then applies **division.v** to calculate probability of each element then applies **maxValue.v** to get largest output.

1. **Each unit Verilog testbench code**

* exponential\_tb.v 🡺 exponential.v
* reciporical\_tb.v 🡺 reciporical.v
* division\_tb.v 🡺 division.v
* Softmax\_Layer\_tb 🡺 Softmax\_Layer.v

**( all values converted through** <https://www.h-schmidt.net/FloatConverter/IEEE754.html> **)**

1. **exponential\_tb.v Results**

* input value ‘b10111111100000000000000000000000 🡺 -1

output value ‘b00111110101111000111000111001000 🡺0.368

A screenshot of a computer

Description automatically generated

* input value ‘b00000000000000000000000000000000 🡺 0

output value ‘b00111111100000000000000000000000 🡺1

A screenshot of a computer

Description automatically generated

1. **reciporical\_tb.v Results**

* input value ‘b01000000000000000000000000000000 🡺 2

output value ‘b 00111111000000000000000000000001 🡺 0.500000059

A screenshot of a computer

Description automatically generated

* input value ‘b01000001001000000000000000000000 🡺 10

output value ‘b 00111101110011001100110011001110 🡺 0.1000000089

A screenshot of a computer

Description automatically generated

1. **division\_tb.v Results**

input value ‘b01000000100000000000000000000000 🡺 4

input value ‘b 01000000110000000000000000000000 🡺6

output value ‘b 00111111001010101010101010101100 🡺 0.66666674614

**A screenshot of a computer

Description automatically generated**

1. **Softmax\_Layer\_tb.v Results**

input 1D array 🡺‘h000000003f8000000000000000000000000000000000000000000000000000000000000000000000

output 1D array

🡺3daec5e63e6d85783daec5e63daec5e63daec5e63daec5e63daec5e63daec5e63daec5e63daec5e6

Max output probability

A screenshot of a computer

Description automatically generated🡺 3e6d8578

input 1D array

🡺‘bf80000000000000bf0000003f3333333f07ae14be6b851fbf5d70a43eb4d6a13f1ef9db3dfced91

output 1D array

🡺3d0530eb3db4f0573d5b7d8b3e362e6c3e19b37f3d8fc3123d1864803e00cb2c3e2857fb3dccb923

Max output probability

🡺 3e362e6c

1. **For utilization report**

The synthesis board was Artix-7 AC701 (largest board downloaded on pc) and utilization percentage is 99.75% because available LUTs are 134600. If board was Kertix ACU107, the utilization percentage decrease to 60%.

[**Softmax\_Layer\_utilization\_synth7.rpt**](Softmax_Layer_utilization_synth7.rpt)

[**Softmax\_Layer\_utilization\_synth5.pdf**](Softmax_Layer_utilization_synth5.pdf)

1. **For schematic**

<schematic_Softmax5.pdf>