SoCLab Final Project Report

FPGA-based Instant Image Recognition on Convolutional Neural Network

D06943020 鄧傑方

R06943086 張奕凡

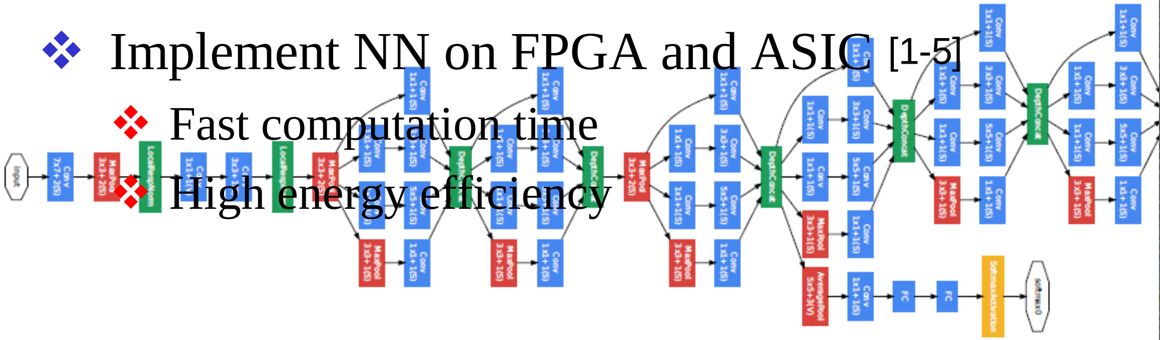
Advisor: Prof. An-Yeu Wu

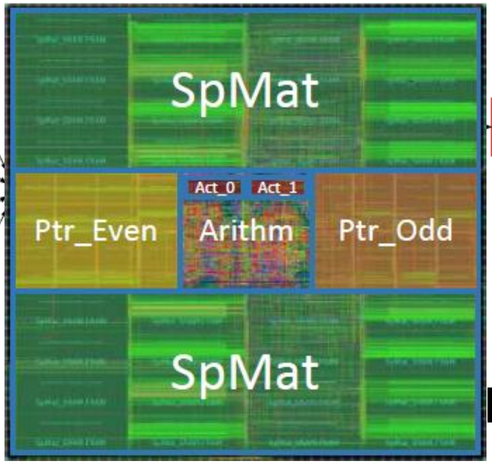
Date: 2018/1/15

Background

- Neural networks have many breakthroughs in recent years
  - Computer vision: image recognition, object detection
  - Speech domain: machine translation, chatbot
  - AlphaGo
- Bottleneck of deep neural network
  - Massive matrix multiplication
  - Long computation time
  - High energy consumption
- Implement NN on FPGA and ASIC [1-5]
  - Fast computation time
  - High energy efficiency

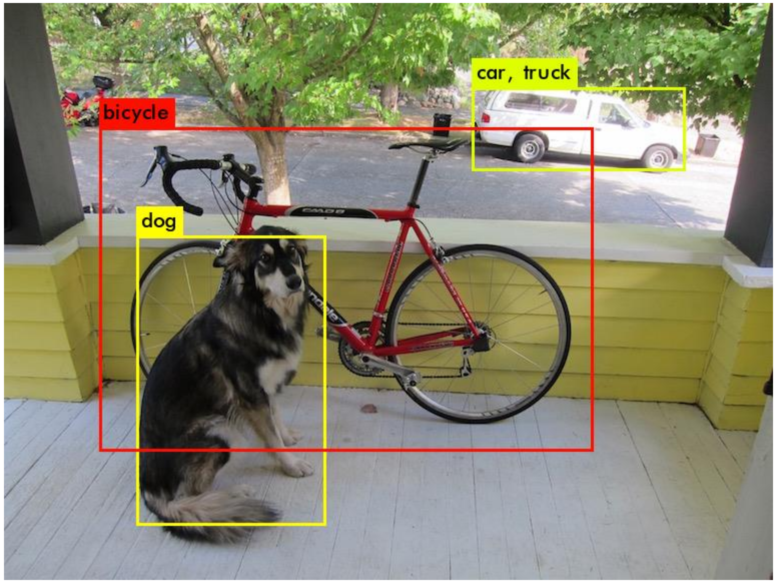
nature
THE INTERNATIONAL WEEKLY JOURNAL OF SCIENCE
At last – a computer program that can beat a champion Go player PAGE 484
ALL SYSTEMS GO

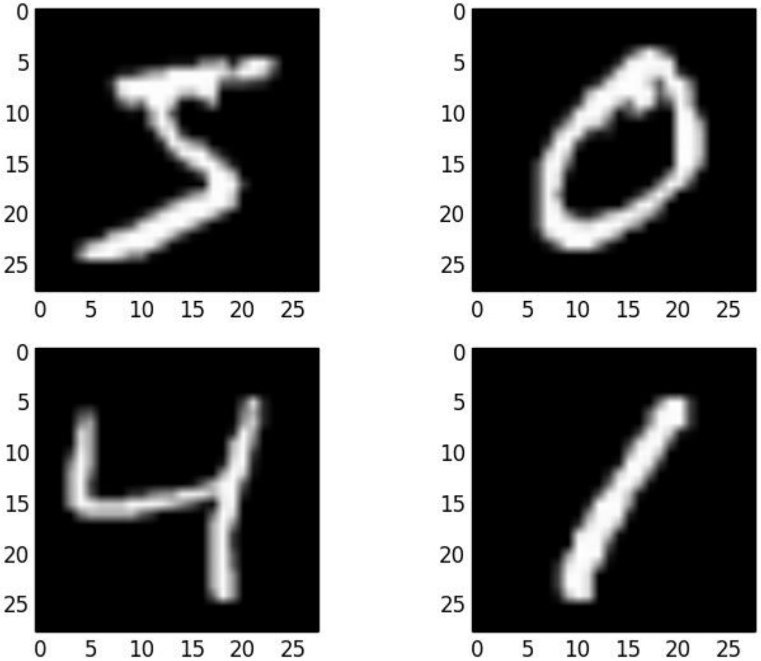




leNet

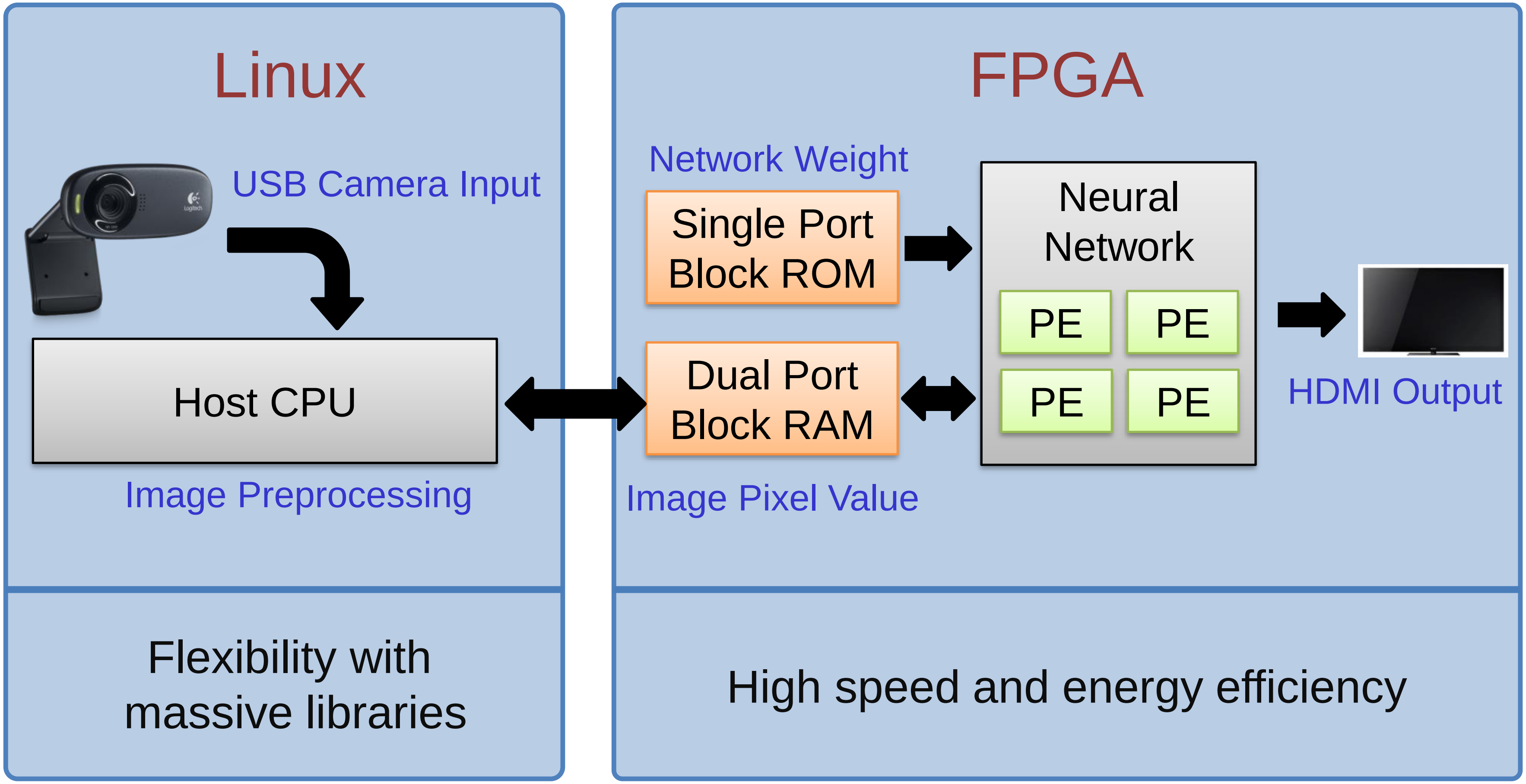
Our Target





- Real-time object classification
  - One of the main techniques for self-driving cars
- Handwritten digits recognition
  - Easier to implement
  - Can be extended to more complex applications

Software/Hardware Co-design Overview [5]



Linux

USB Camera Input

Host CPU

Image Preprocessing

Flexibility with massive libraries

FPGA

Network Weight

Single Port Block ROM

Dual Port Block RAM

Image Pixel Value

Neural Network

PE PE

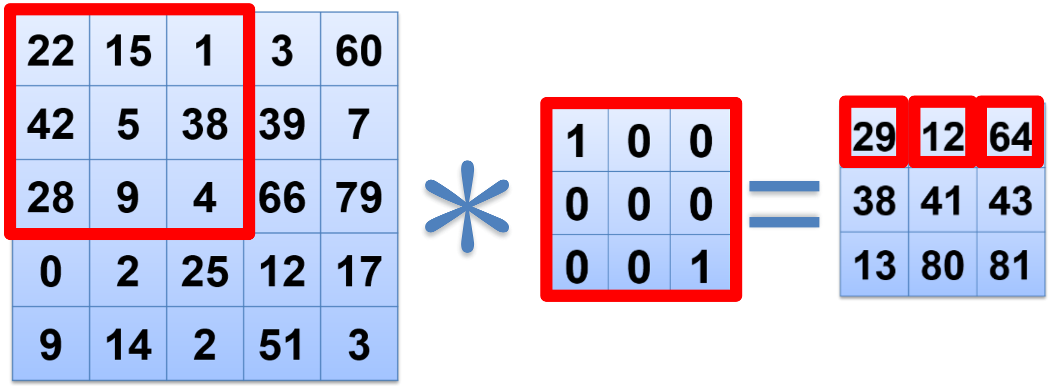
PE PE

HDMI Output

High speed and energy efficiency

HW: Neural Network

- Parameterized module
- Convolutional layer



22	15	1	3	60
42	5	38	39	7
28	9	4	66	79
0	2	25	12	17
9	14	2	51	3

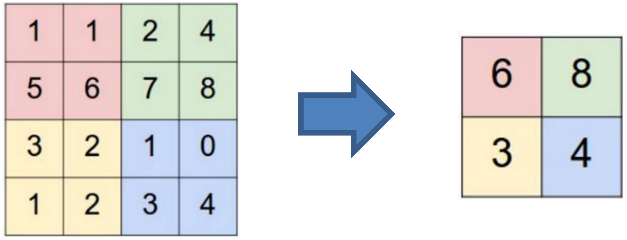
*

1	0	0
0	0	0
0	0	1

=

29	12	64
38	41	43
13	80	81

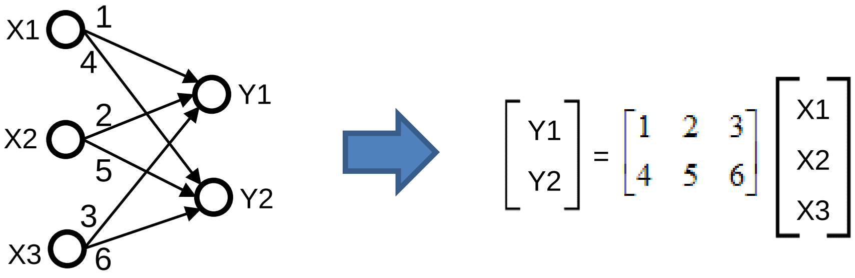
- Maxpooling layer



1	1	2	4
5	6	7	8
3	2	1	0
1	2	3	4

6	8
3	4

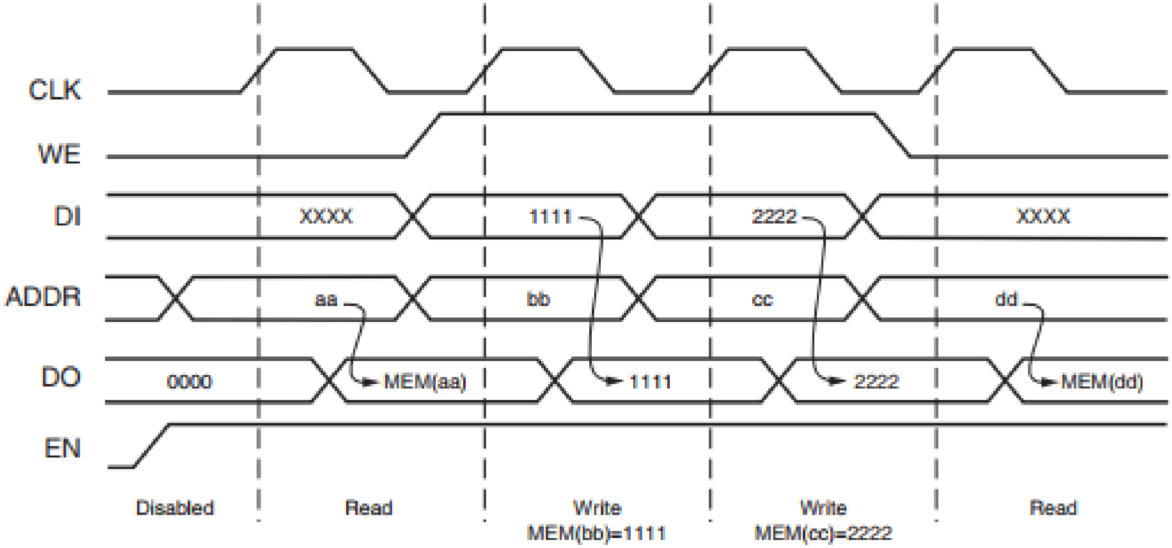
- Dense layer
- Relu layer
  - $f(x) = \max(0, x)$
- Softmax layer
  - Find biggest value



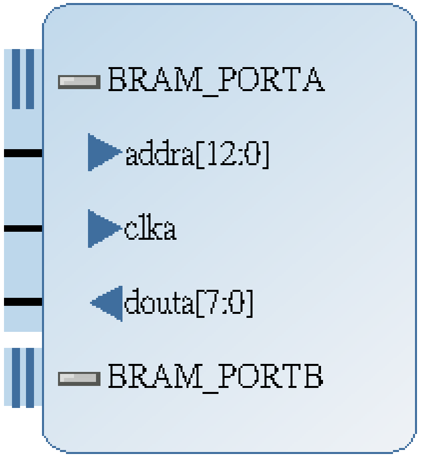
$$\begin{bmatrix} Y1 \\ Y2 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \begin{bmatrix} X1 \\ X2 \\ X3 \end{bmatrix}$$

HW: Single Port Block ROM [6-7]

- Store pre-trained MNIST model weight
  - Quantize to eight bit
  - Multiply 4: maintain accuracy and avoid overflow
  - Save to .coe and load into BRAM

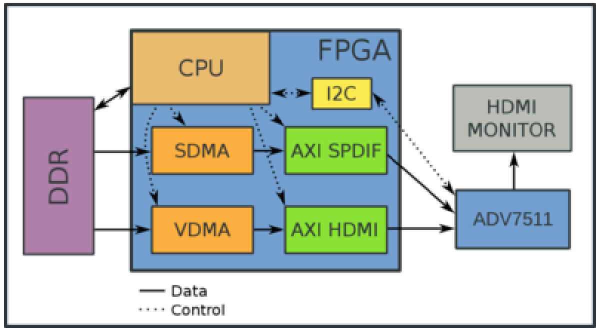


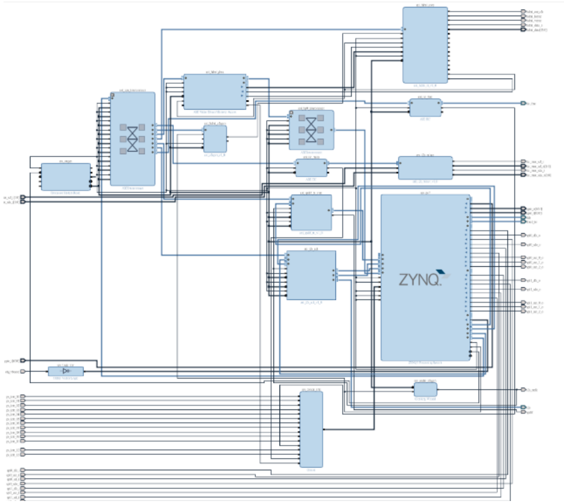
WRITE_FIRST Mode



HW: HDMI Output [8-10]

- FPGA
  - Insufficient time to make out how HDMI hardware works
- Embedded linux
  - Need HDMI driver
  - Hard to make HDMI hardware controllable
- Applicable tutorials and examples are few
  - Most are implemented in VGA
  - Many examples are no longer maintained
    - Bugs exist or version mismatch





- Replaced with seven-segment display and LED

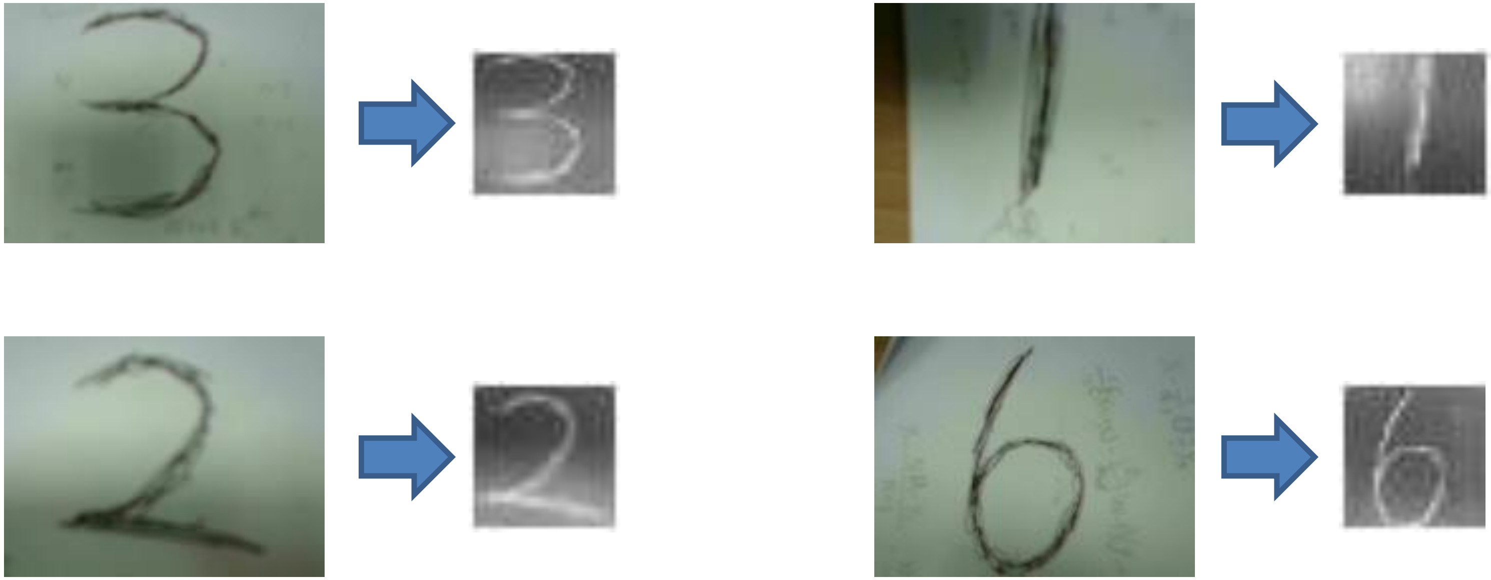
SW: USB Camera Input [11-12]

- USB camera uses H.264 as video compression standard
- Enable USB camera drivers @ kernel configuration for Linaro
- Video4Linux (V4L) is a collection of device drivers and an API for supporting real-time video capture on Linux systems
  - Suitable for USB webcam

- Open /dev/video0
- VIDIOC_QUERYCAP
- VIDIOC_REQBUFS
- V4L2_MEMORY_MMAP
- VIDIOC_STREAMON
- VIDIOC_DQBUF
- VIDIOC_STREAMOFF

SW: Image Preprocessing

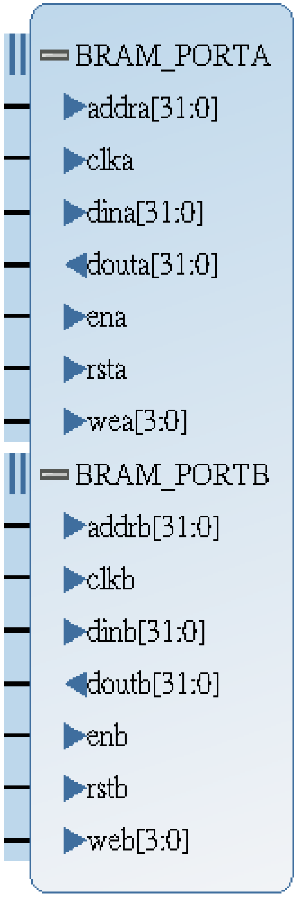
- The image for MNIST is 28x28 pixel and value from 0 to 1
- The image from webcam is 120x160 pixel and color
  - Cut picture to 120x120
  - Down-sample to 28x28
  - Color map to 0~1



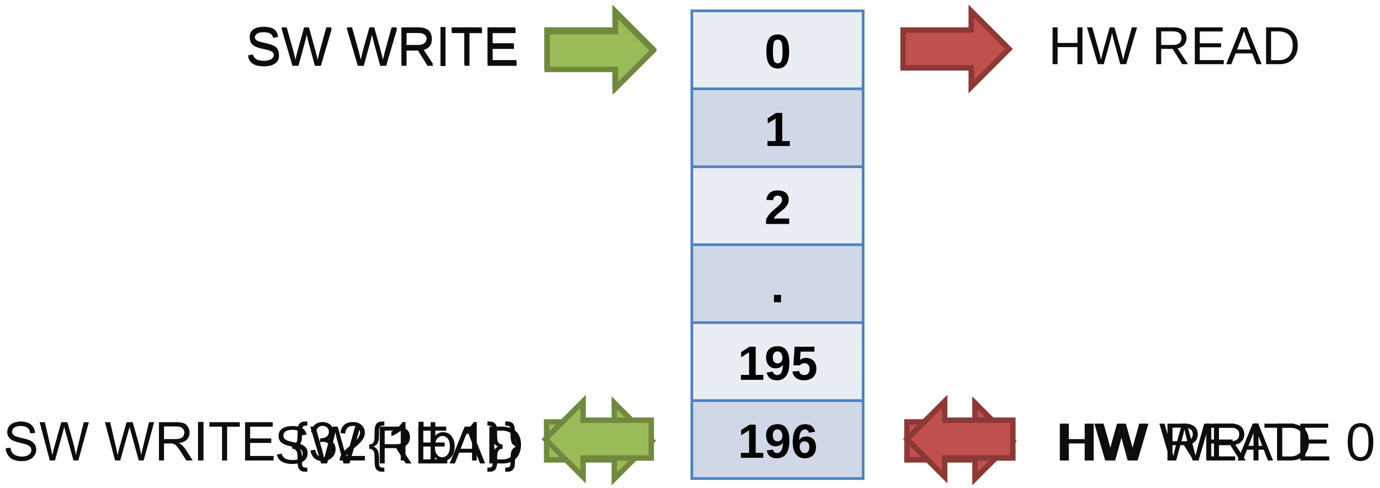
HW: Dual Port Block RAM (1/3) [13-15]

- The bridge for HW and SW
  - SW write images into BRAM
  - HW read images from BRAM
- Image size: 28x28 pixels, 8 bits for each pixel

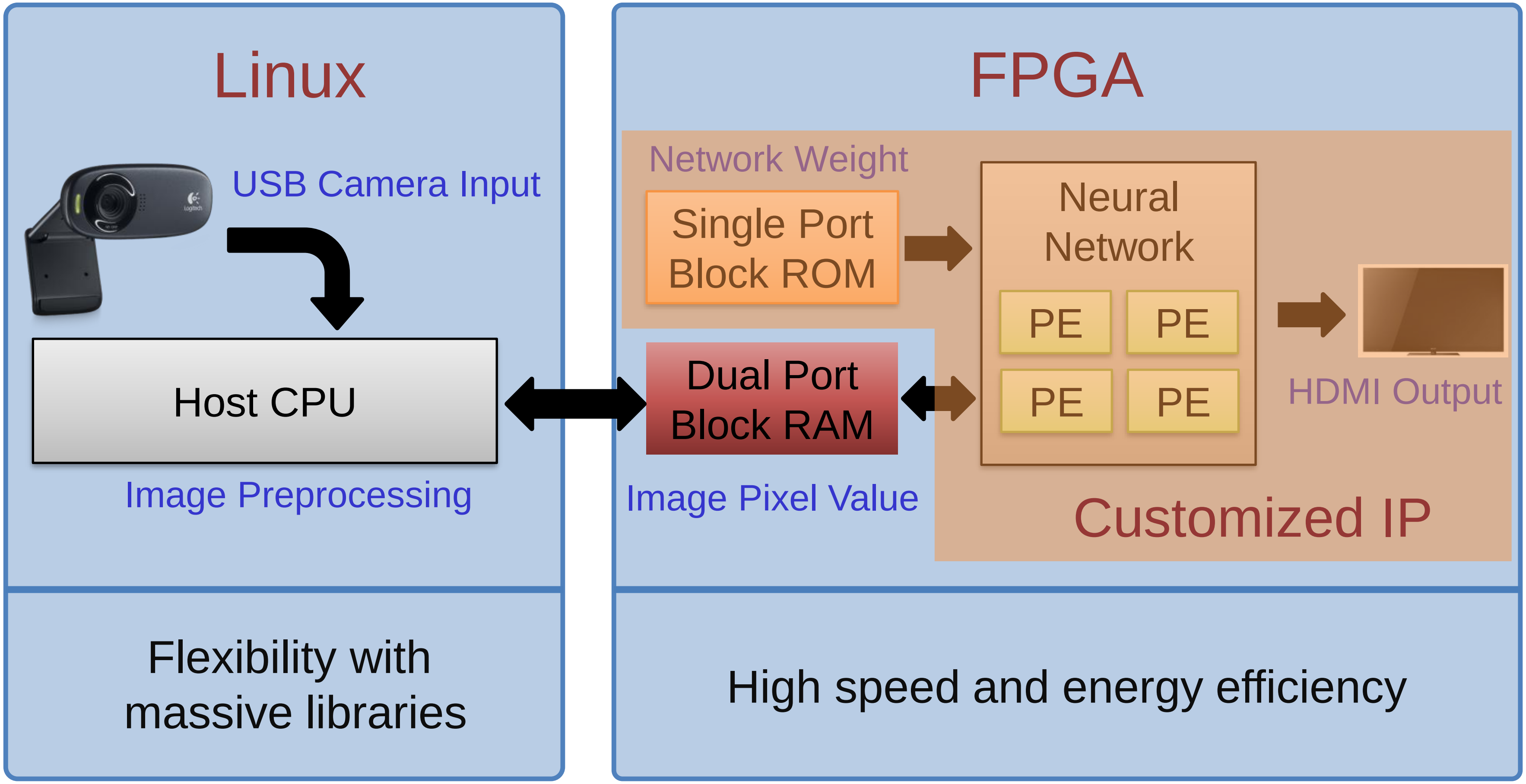
➔ Need 32 bits (Width) x 196 (Depth) dual port BRAM



- Protocol between HW and SW



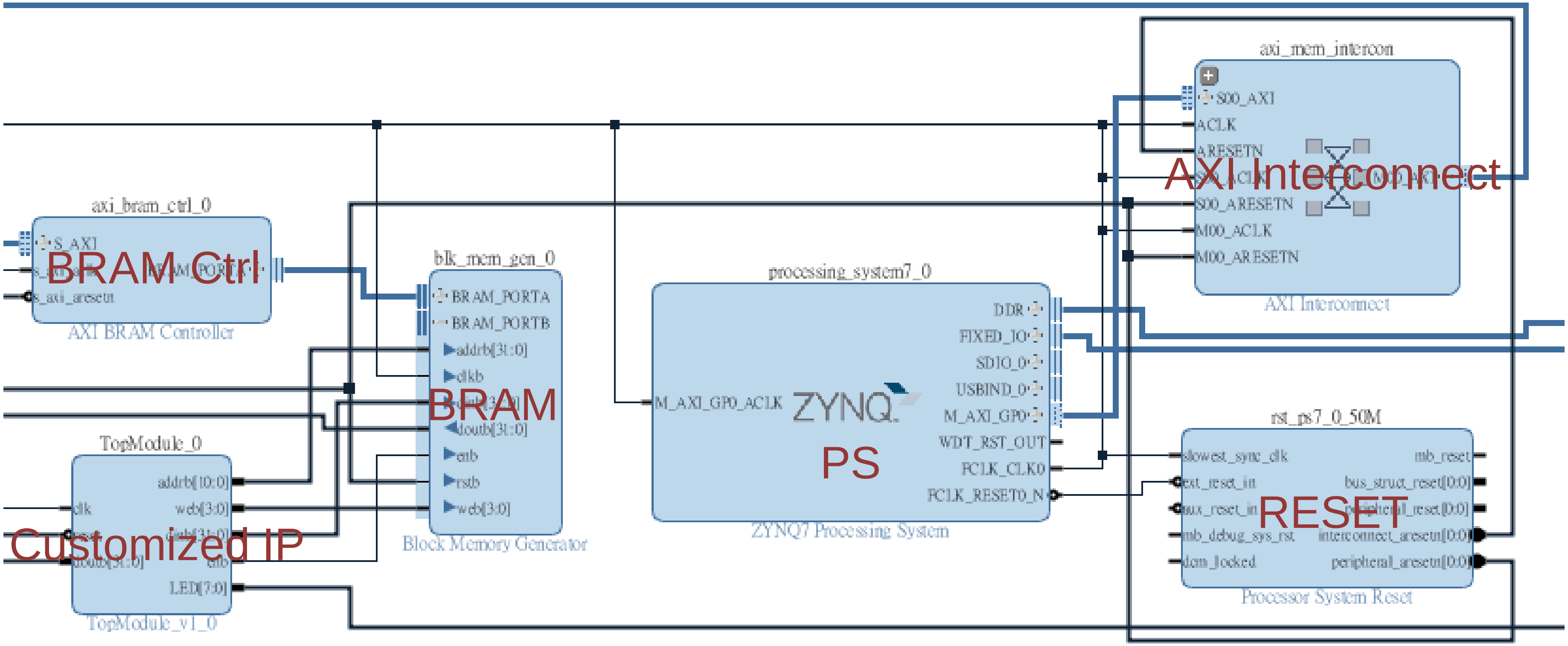
HW: Dual Port Block RAM (2/3) [13-15]



Flexibility with massive libraries

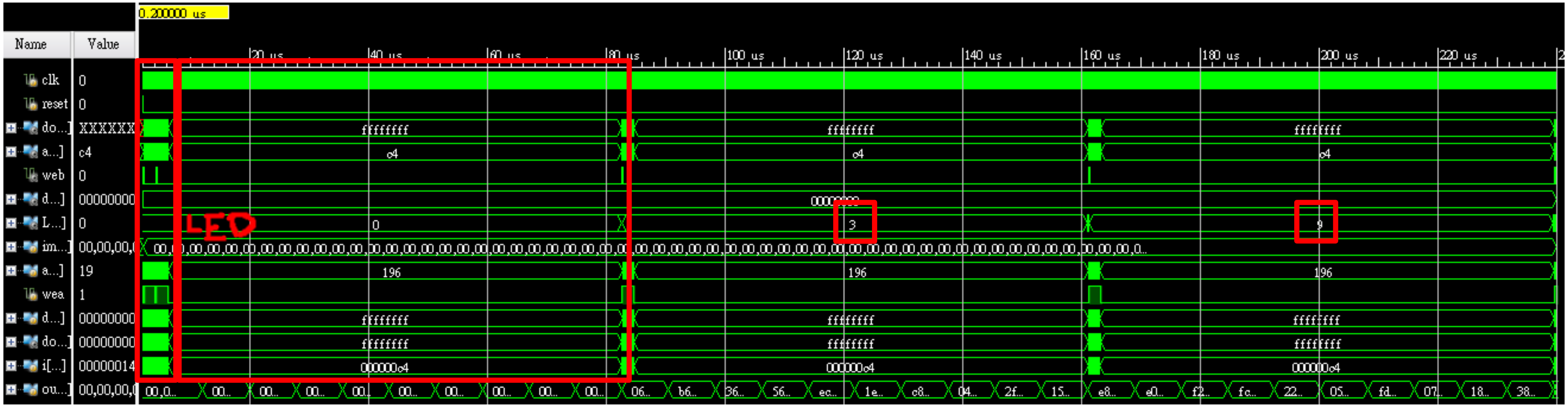
High speed and energy efficiency

HW: Dual Port Block RAM (3/3) [13-15]



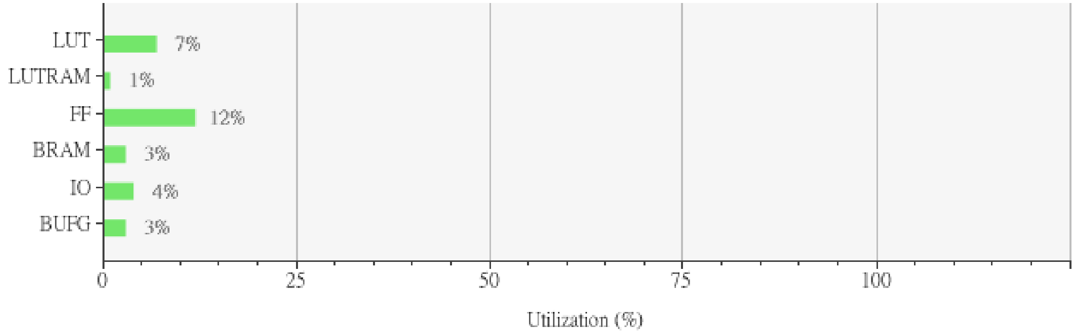
- Test on standalone
  - Can’t read the value written into BRAM, always get 0
  - Use LED[7:0] as debug tool…

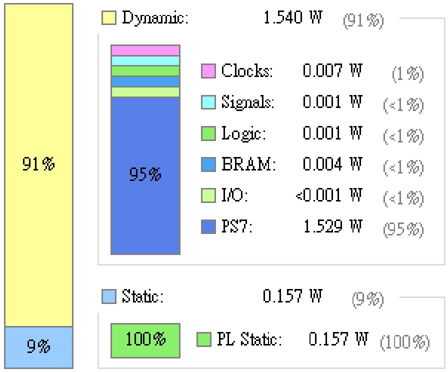
Simulation Result



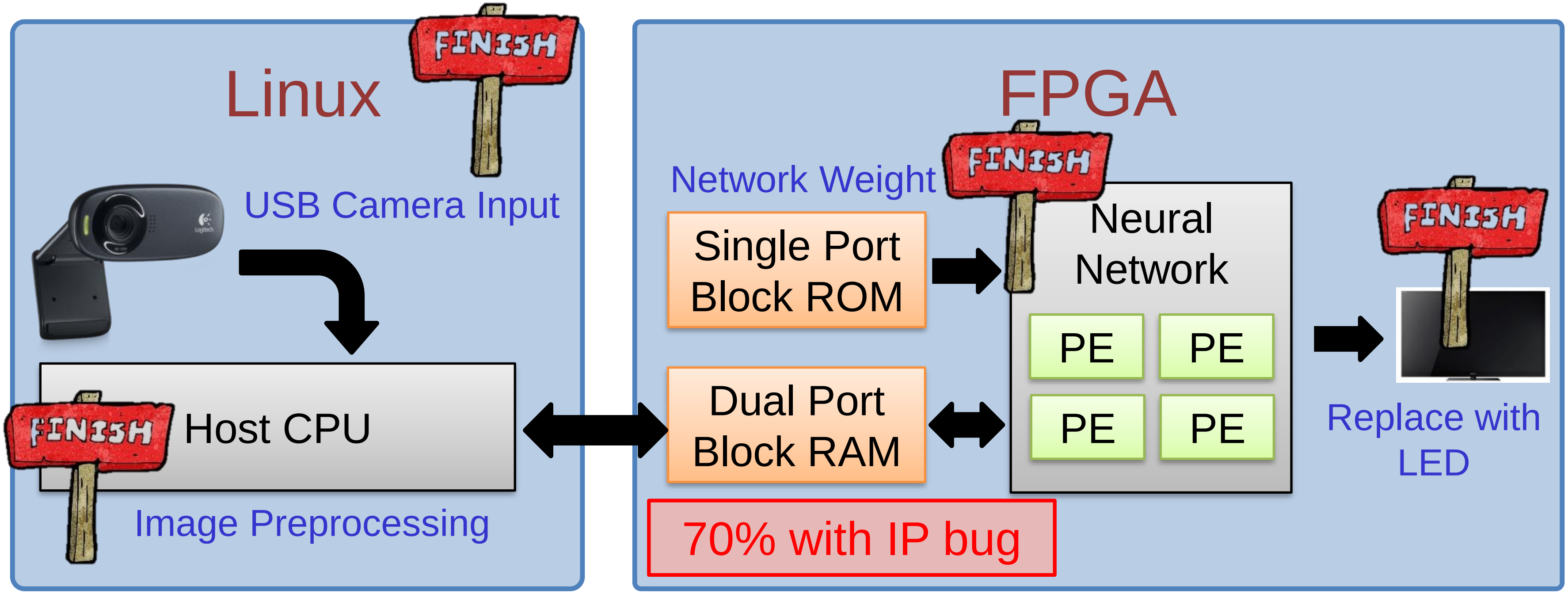
SW Write HW Calc Predict Result3 Next Result 9

- Testbench performs ideally
  - Without pipeline, predict one image in about 8000 cycles





Conclusion



- Hardware bug is every where
  - Error messages are difficult to understand, FPGA bug, tool bug, version bug…
  - Experience-based
- Many thanks to TA奕達 & 俊棋學長

Reference (1/2)

[1] Han, Song, et al. “EIE: efficient inference engine on compressed deep neural network.” *Proceedings of the 43rd International Symposium on Computer Architecture*. IEEE Press, 2016.

[2] Convolution Neural Network CNN Implementation on Altera FPGA using OpenCL: https://www.youtube.com/watch?v=78Qd5t-Mn0s

[3] Farabet, Clément, *et al*. “Hardware accelerated convolutional neural networks for synthetic vision systems,” *Proceedings of 2010 IEEE International Symposium on Circuits and Systems (ISCAS),* 2010.

[4] Zhao, Wenlai, *et al*, “F-CNN: An FPGA-based framework for training Convolutional Neural Networks,” *2016 IEEE 27th International Conference on Application-specific Systems, Architectures and Processors (ASAP),* 2016.

[5] Guo, Kaiyuan, *et al*, “Software-Hardware Codesign for Efficient Neural Network Acceleration,” *IEEE Micro* 37.2 (2017): 18-25.

[6] XILINX ROM 使用教程 http://cocdig.com/docs/show-post-43205.html

[7] 7 Series FPGAs Memory Resources User Guide https://www.xilinx.com/support/documentation/user_guides/ug473_7Series_Memory_Resources.pdf

[8] Linux with HDMI video output on the ZED, ZC702 and ZC706 boards https://wiki.analog.com/resources/tools-software/linux-drivers/platforms/zynq

[9] ZYNQ平台的HDMI驱动测试 http://blog.csdn.net/rzjmpb/article/details/50212875

Reference (2/2)

[10] FMC-HDMI-CAM + PYTHON-1300-C Getting Started Design, Vivado 2014.4 http://picozed.org/content/fmc-hdmi-cam-python-1300-c-getting-started-design-vivado-20144

[11] Interfacing a USB WebCam and Enable USB Tethering on ZYNQ-7000 AP SoC Running Linux https://medium.com/@chathura.abeyrathne.lk/interfacing-a-usb-webcam-and-enable-usb-tethering-on-zynq-7000-ap-soc-running-linux-1ba6d836749d

[12] (原创)基于ZedBoard的Webcam设计(一)：USB摄像头(V4L2接口)的图片采集 http://www.cnblogs.com/surpassal/archive/2012/12/19/zed_webcam_lab1.html

[13] 双口BRAM的使用 http://blog.chinaaet.com/kevinc/p/5100051535

[14] AXI Block RAM (BRAM) Controller v4.0 LogiCORE IP Product Guide https://www.xilinx.com/support/documentation/ip_documentation/axi_bram_ctrl/v4_0/pg078-axi-bram-ctrl.pdf

[15] Block Memory Generator v8.3 LogiCORE IP Product Guide https://www.xilinx.com/support/documentation/ip_documentation/blk_mem_gen/v8_3/pg058-blk-mem-gen.pdf

Job Assignment

- 奕凡
  - Survey HDMI and try to fix bug
  - Image preprocessing to fit MNIST model
  - Slide (30%)
- 傑方
  - Neural network hardware
  - Train MNIST model and load weight to BRAM
  - Seven-segment display
  - USB camera driver on Linaro and software implement
  - Dual port block ram
  - Slide (70%)