VLSI SYSTEM DESIGN - FINAL PROJECT

A CNN accelerator for fruit recognition

男童俱樂部

黃昱澄 黃冠予 王昱承
俞杉麒 陳奕萍 賴致文

CONTENTS

1 System architecture

NN architecture

HW architecture

EPU architecture

2 Verification

EPU verification

Full sys verification

3 Overall results

Speed perf

Area perf

Power perf

1

System architecture

1. NN architecture
2. HW architecture
3. EPU architecture

NN architecture

	Imap	Kernel	Omap	Note	Time (s)
Conv 0	32*32*3	3*3*3*60	32*32*60	3x3 conv	0.014
Conv 1	32*32*60	1*1*60*33	32*32*33	1x1 conv	0.051
Conv 2	32*32*33	1*1*33*20	32*32*20	1x1 conv	0.019
Pool 0	32*32*20	32*32	1*1*20	32x32 Max	0.002

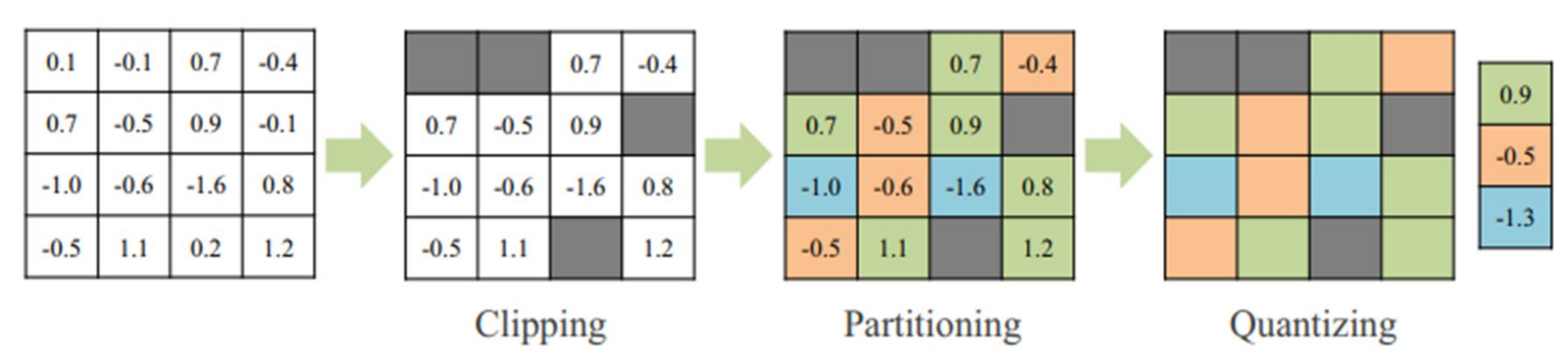
Dataset :

Source : **Fruit Classification** (kaggle)
Number of classes: **20** (fruits and vegetables)
Image size: **32*32**
Train : 200 img/per fruit, Total = **4000** images
Test : Average 50 img/per fruit, Total = **1000** images

Final Accuracy = 80.6%

NN architecture

For Example : **Conv0**



In8.hex

1	20
2	20
3	20
…	
3072	20

×

W2.hex
0EFF8
10_11_11_11_11_10_00

W8.hex
05_FA_ED_00

+

Bias32.hex

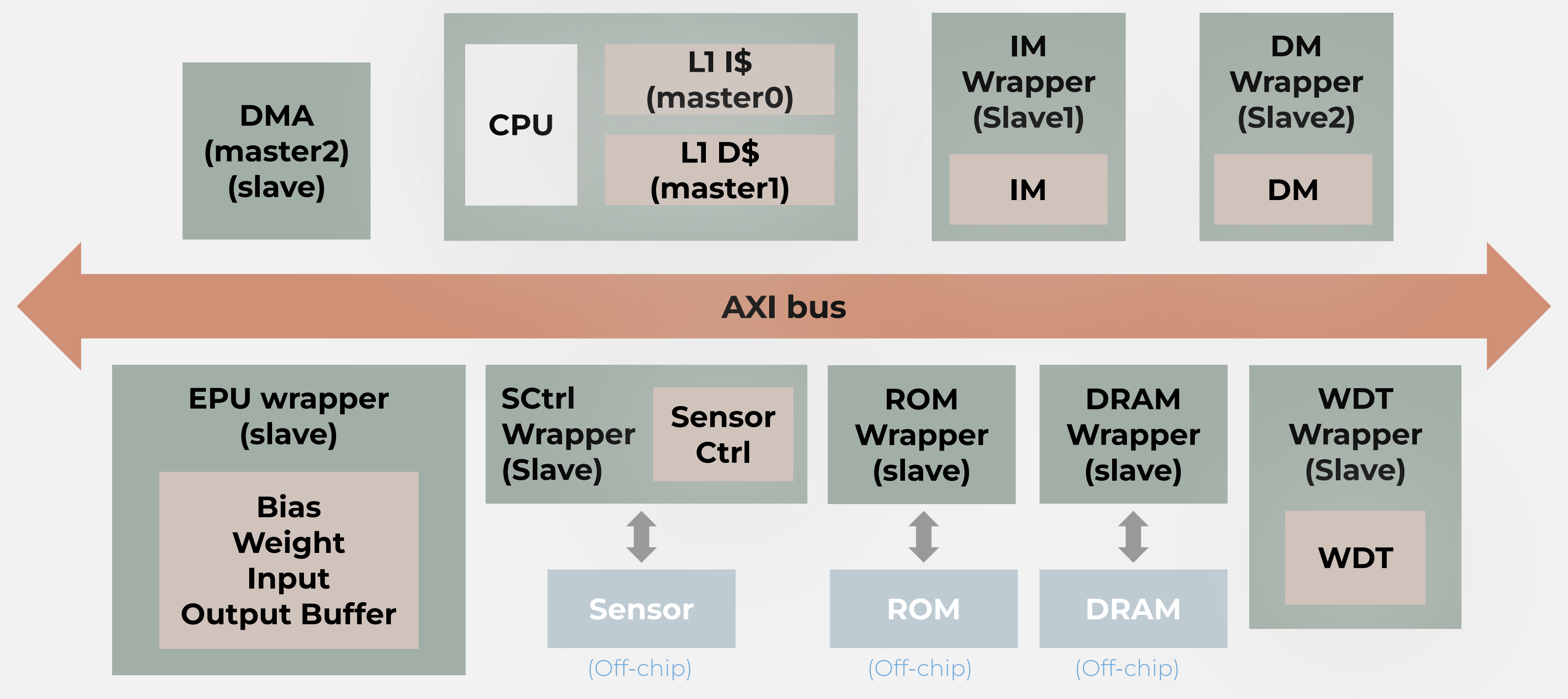
1	00000111
2	FFFFFDEC
3	00000080
…	
60	FFFFFFF6

=

Out8.hex

1	00
2	00
3	00
…	
61440	00

HW architecture



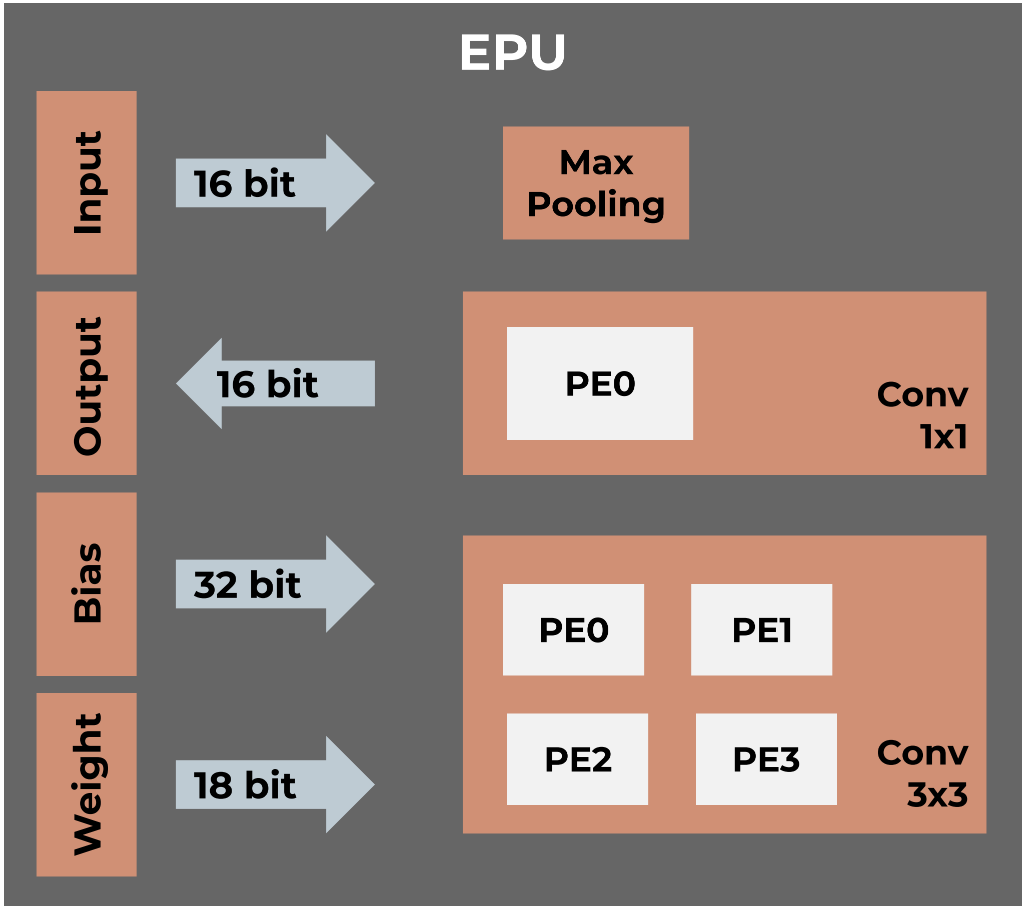
EPU architecture

EPU structure

- Weight buffer(180KB)
- Bias buffer(2KB)
- Input buffer(384KB)
- Output buffer(384KB)

EPU mode

- Conv3x3
  - 4PE,each with 9 MACs
  - zero padding
  - Relu
- Conv1x1
  - 1PE with 9 MACs
  - Relu
- Max pooling



2

Verification

1. EPU Verify
2. FULL Sys Verify

EPU verification

A. Stand-alone testbench for EPU

B. TB loads input/weight/bias data into RTL-simulated SRAM buffers.

C. TB pulls start signal to high

D. EPU starts computation and writes results to output buffer.

E. EPU pulls finish signal to high

F. TB verify the content of output buffer.

	Imap	Kernel	Omap	Note	Time (s)
Conv 0	32*32*3	3*3*3*60	32*32*60	3x3 conv	0.014
Conv 1	32*32*60	1*1*60*33	32*32*33	1x1 conv	0.051
Conv 2	32*32*33	1*1*33*20	32*32*20	1*1 conv	0.019
Pool 0	32*32*20	32*32	1*1*20	32*32 Max	0.002

Full sys verification

Program flow

A. Assume ALL input/weight/bias data in DRAM.
B. CPU runs booting program with DMA.
C. Use DMA to move data from DRAM to EPU's buffer.
D. CPU writes to EPU ctrl registers.
E. 8-bit weight shared by that layer
F. "start" signal
G. EPU writes to output buffer as CPU stuck at WFI.
H. EPU finishes and send interrupt. CPU continues with ISR.
I. CPU writes ctrl signals for next layer.
J. Trigger "In-Output buffer swap"
K. Output of this layer is the input of next layer
L. If done, DMA move data from EPU to DRAM.
M. TB verify the content of DRAM.

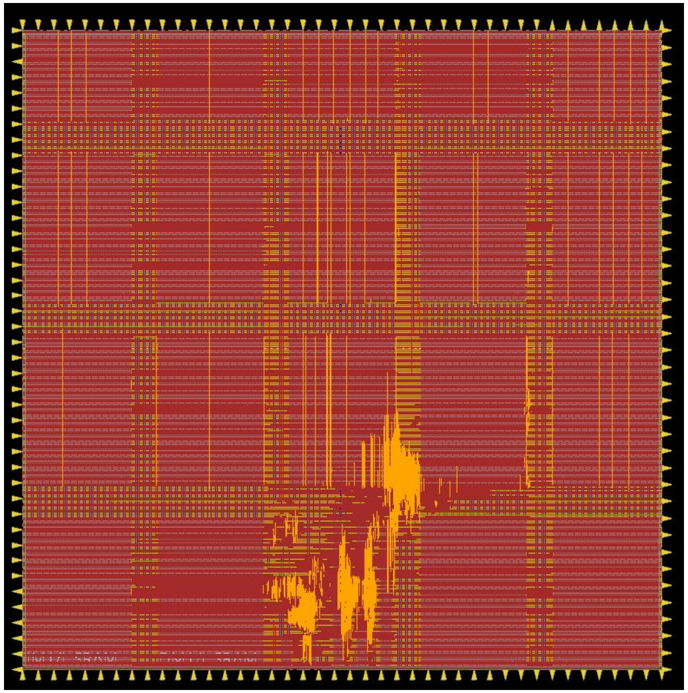
3

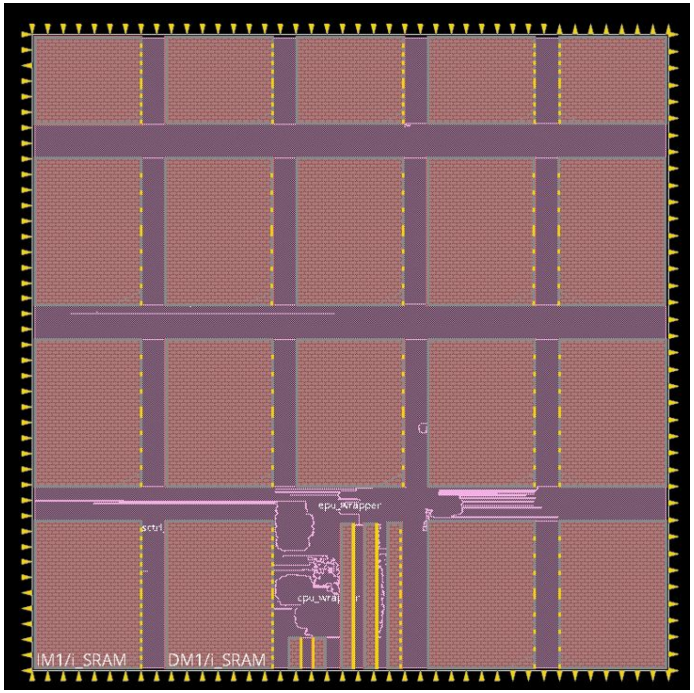
Overall results

1. Speed perf
2. Area perf
3. Power perf

Overall Results

Clock period	12.5ns
Area	**69447383.21 um^2**
Power	**304.26555650mW**





Q&A