Systolic Array based CNN Accelerator

10-class Sign-Language Gesture Inference (64×64 INT8)

hl3999 Harvey Lu

lw3227 Linxiao Wu

cc5397 Cheng-wen Chu

cx2355 Chengcheng Xu

mz3143 Mingyuan Zheng

COLUMBIA ENGINEERING
The Fu Foundation School
of Engineering and Applied Science

Motivation

Real CNN inference on embedded hardware — not just RTL simulation

Goal.

Run a full INT8 CNN end-to-end on DE1-SoC, driven from a browser, with no accuracy loss vs. FP32.

Task.

American Sign Language digits 0–9, 64×64 grayscale, 10-class inference.

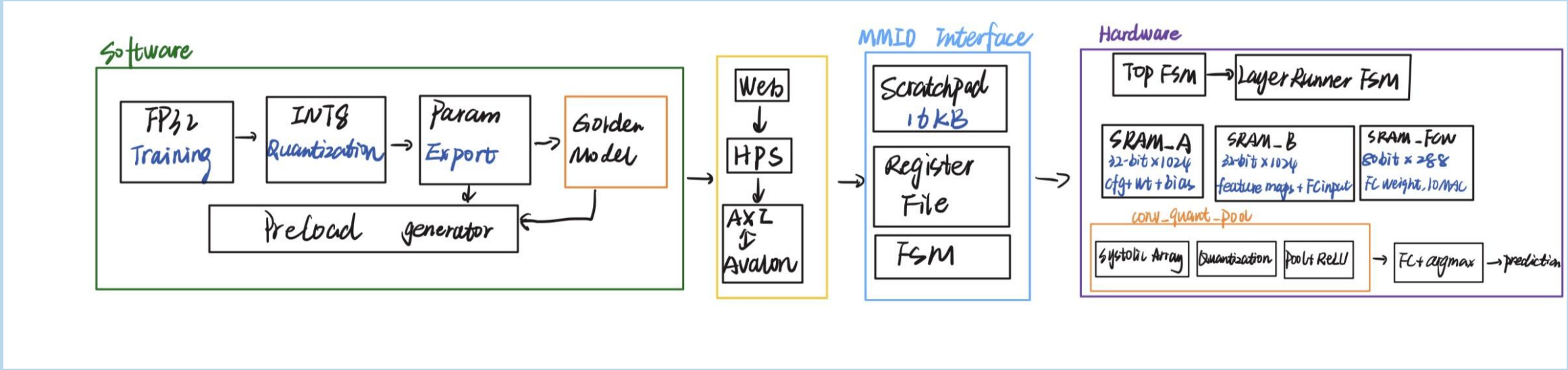
Result.

96.77% accuracy, 4.65× faster than the HPS CPU baseline.

Ten target classes (Kaggle ASL Digits, 2,062 images)

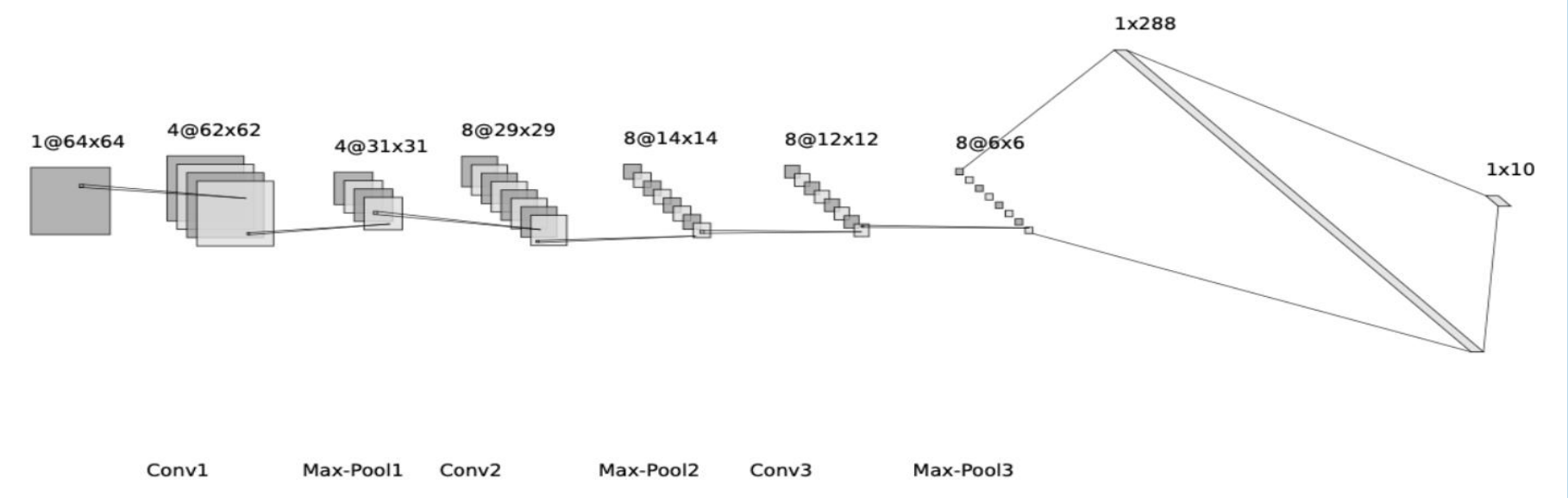
System Overview

Offline software toolchain feeds an on-chip RTL pipeline through a single 32-bit streaming load port.



CNN Model & Dataset

3 conv stages + FC head, deliberately small so every layer maps cleanly onto the accelerator.



Trainable parameters:	Dataset images:	Accuracy (in testing dataset):	FP32 / INT8 agreement
3,810	**2,062**	**96.77%**	**100%**

Convolution Datapath: conv → quant → pool

Single streaming pass per layer. No external buffering between stages.

INT8 activation

Systolic Array

3×3 sliding window
INT8 × INT8 MACs
parallel partial sums

23-bit

Quantization

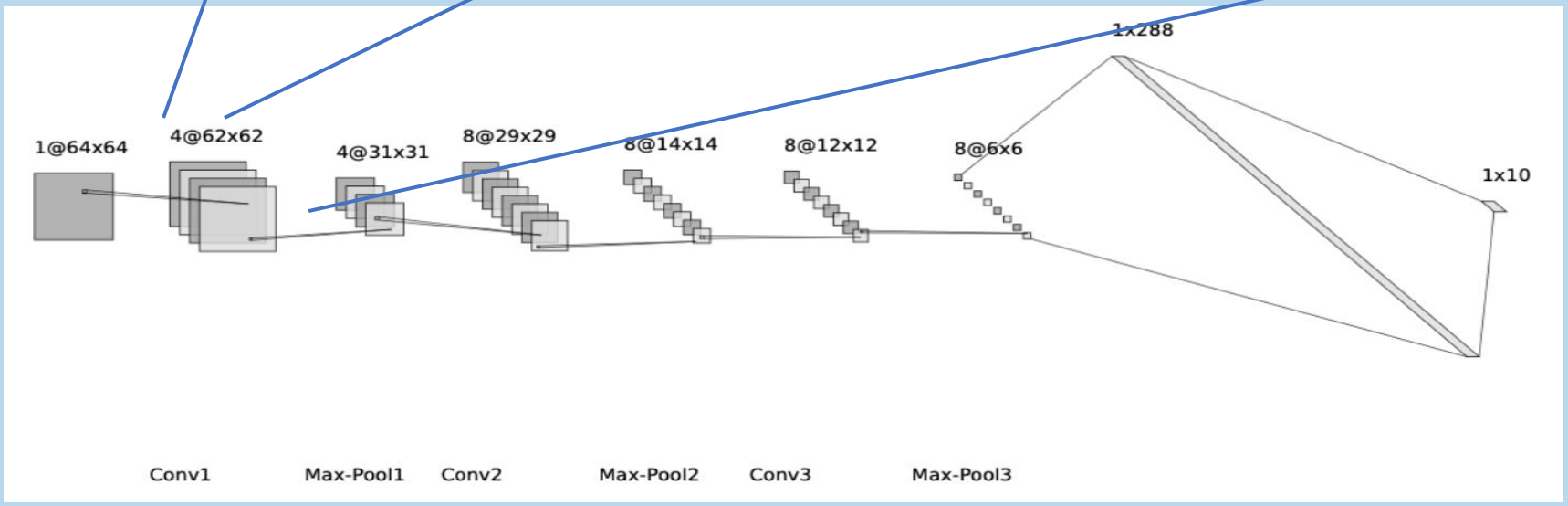
scale · shift · bias
round → clip(-128,127)
per-channel multiplier

INT8

Pool + ReLU

2×2 stride-2 max-pool
fused with ReLU
streaming output

INT8 → SRAM_A/B

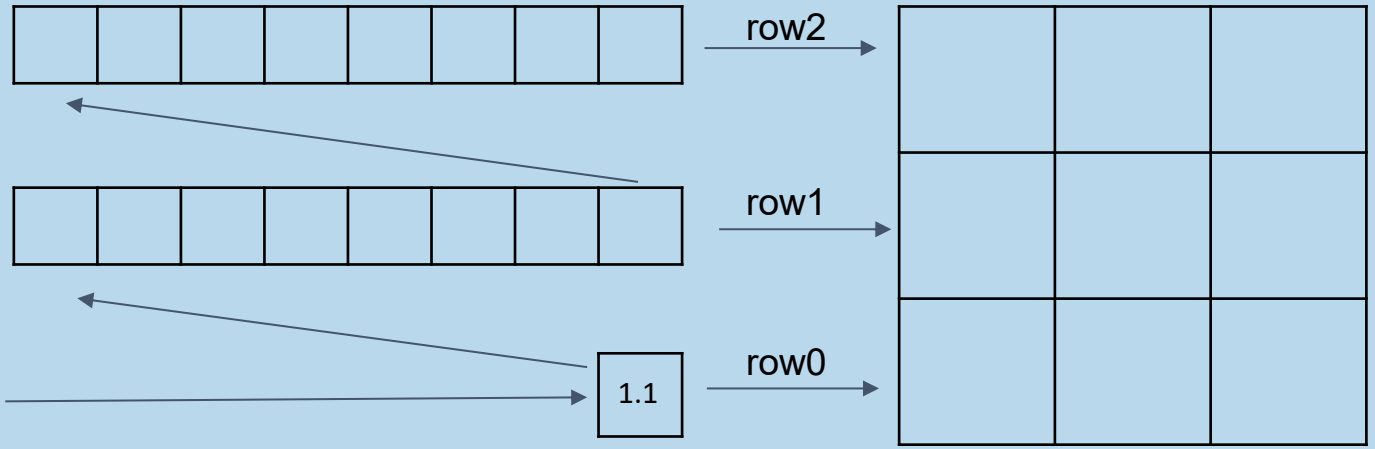


32-bit SRAM stream → 3×3 sliding window

32-bit SRAM stream → 3×3 sliding window

input pictures (ex. 8x8, 1 channel)

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

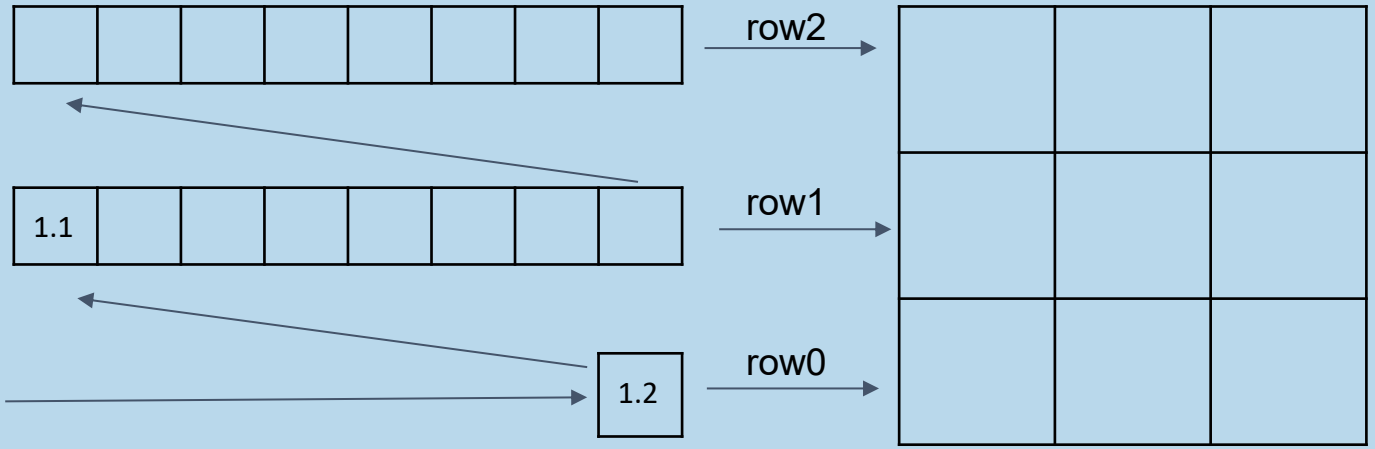


32-bit SRAM stream → 3×3 sliding window

32-bit SRAM stream → 3×3 sliding window

input pictures (ex. 8x8, 1 channel)

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

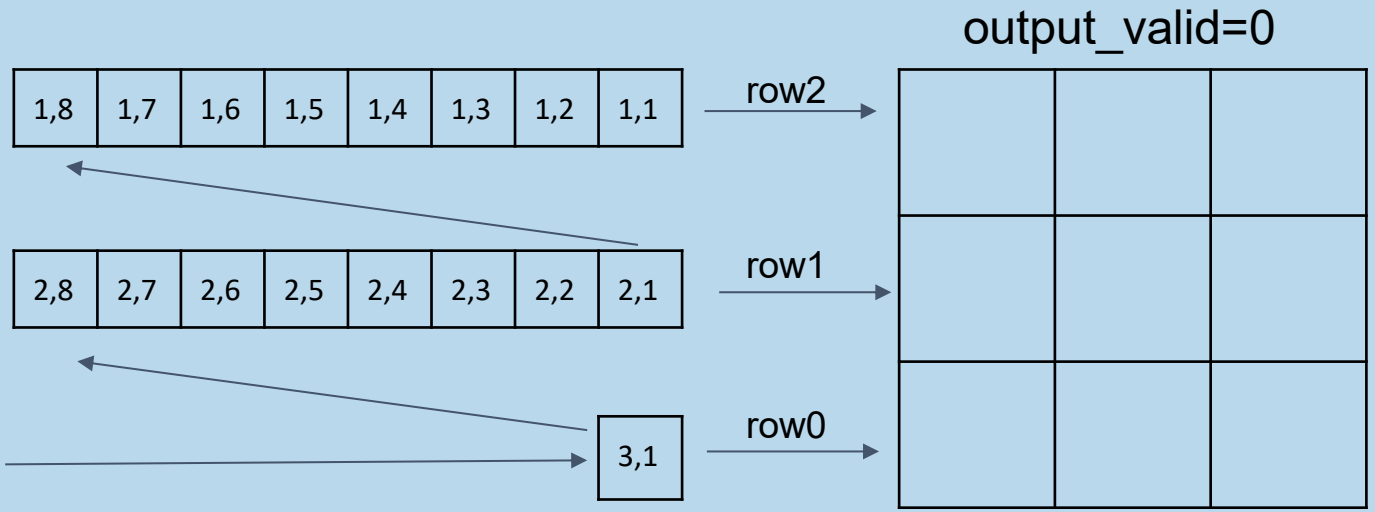


32-bit SRAM stream → 3×3 sliding window

32-bit SRAM stream → 3×3 sliding window

input pictures (ex. 8x8, 1 channel)

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

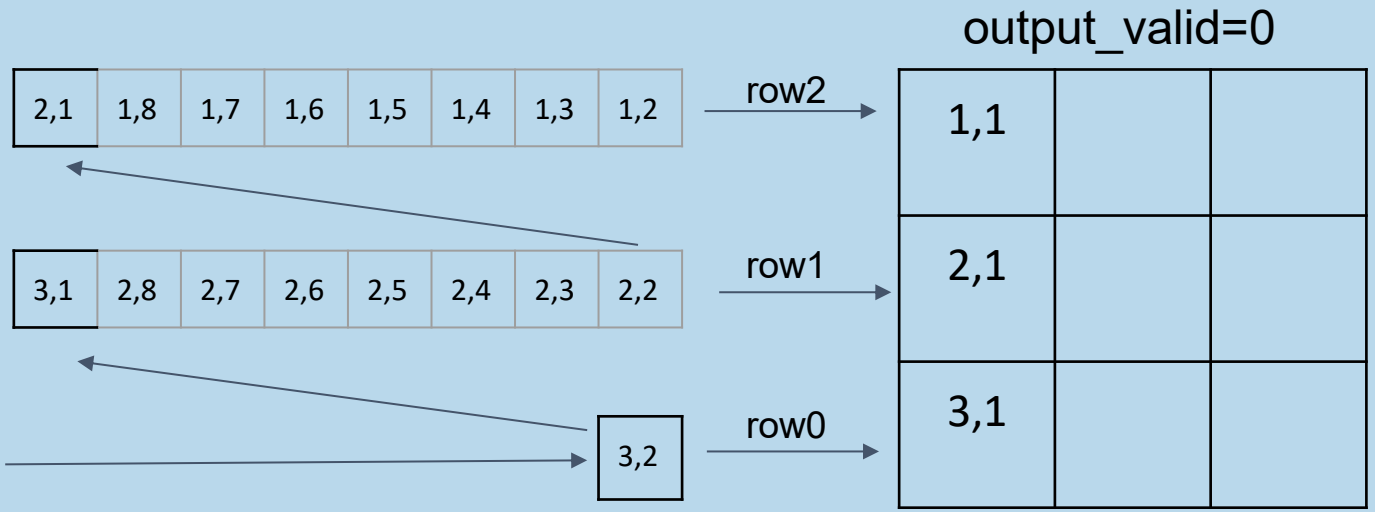


32-bit SRAM stream → 3×3 sliding window

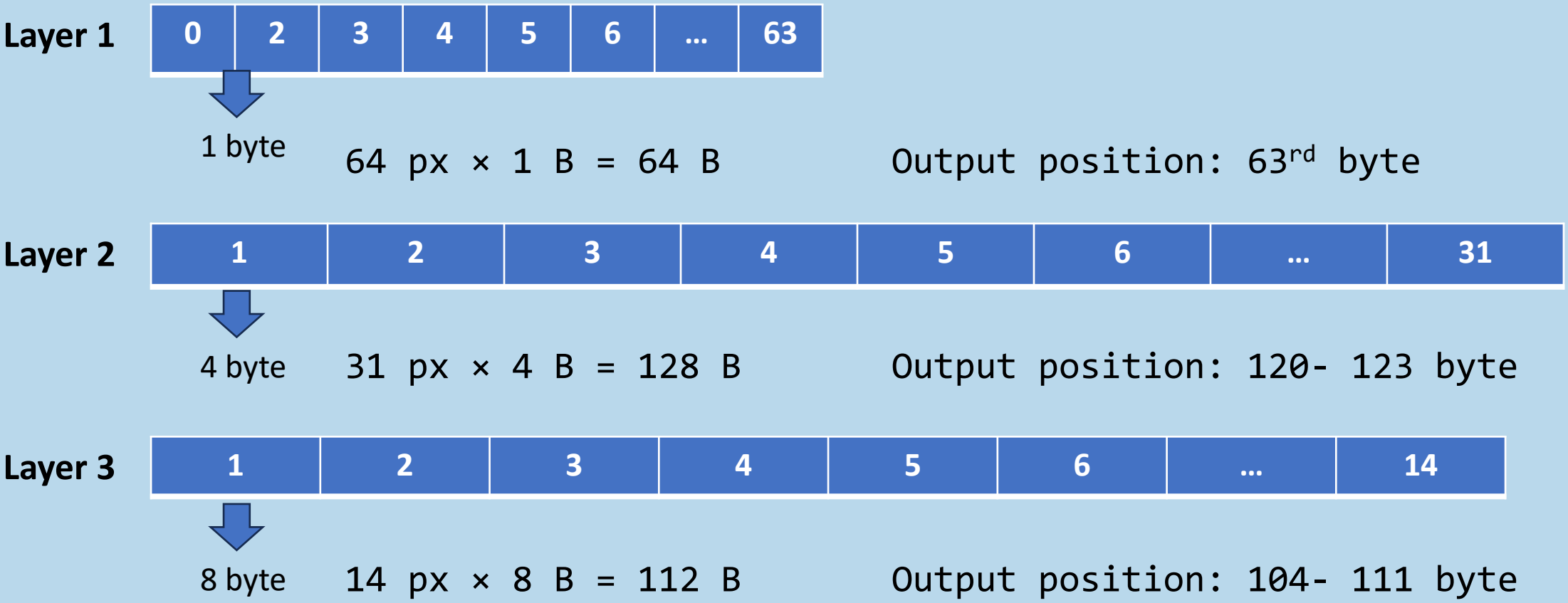
32-bit SRAM stream → 3×3 sliding window

input pictures (ex. 8x8, 1 channel)

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

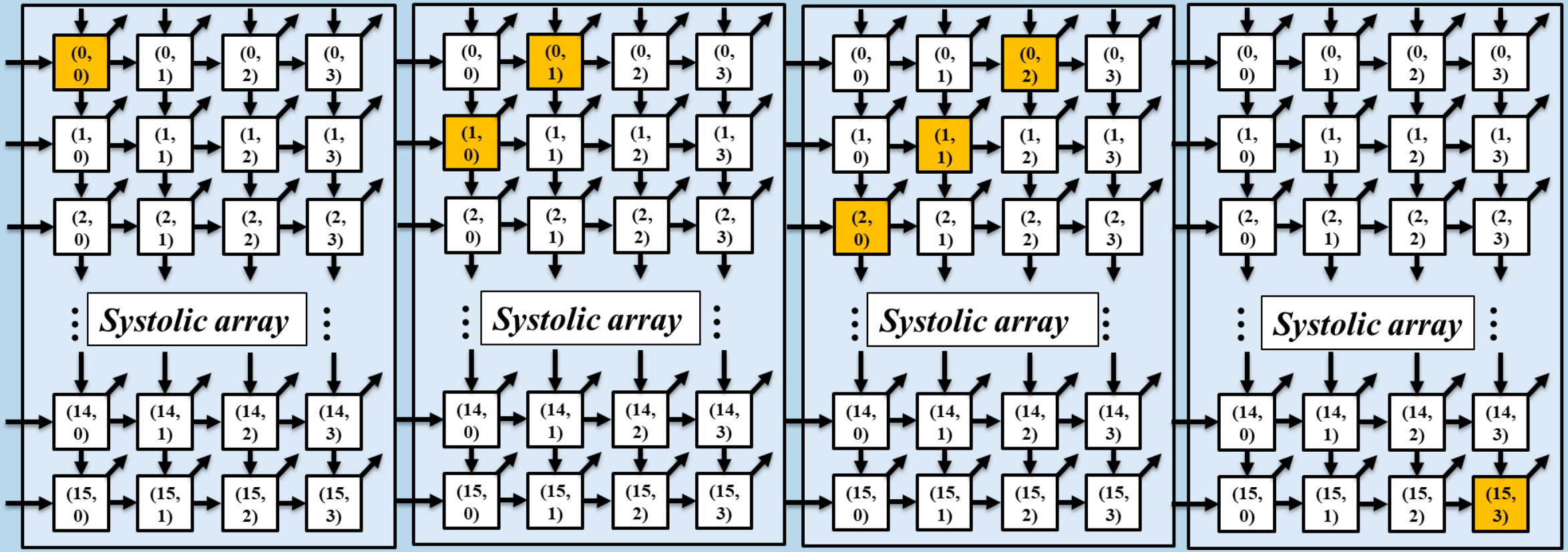


Line Buffer configuration under 3 Layer



Systolic Array — 16×4 PE Grid, reused across all layers

Same fabric runs L1/L2/L3 conv and FC; only the dot-product depth k changes per mode.

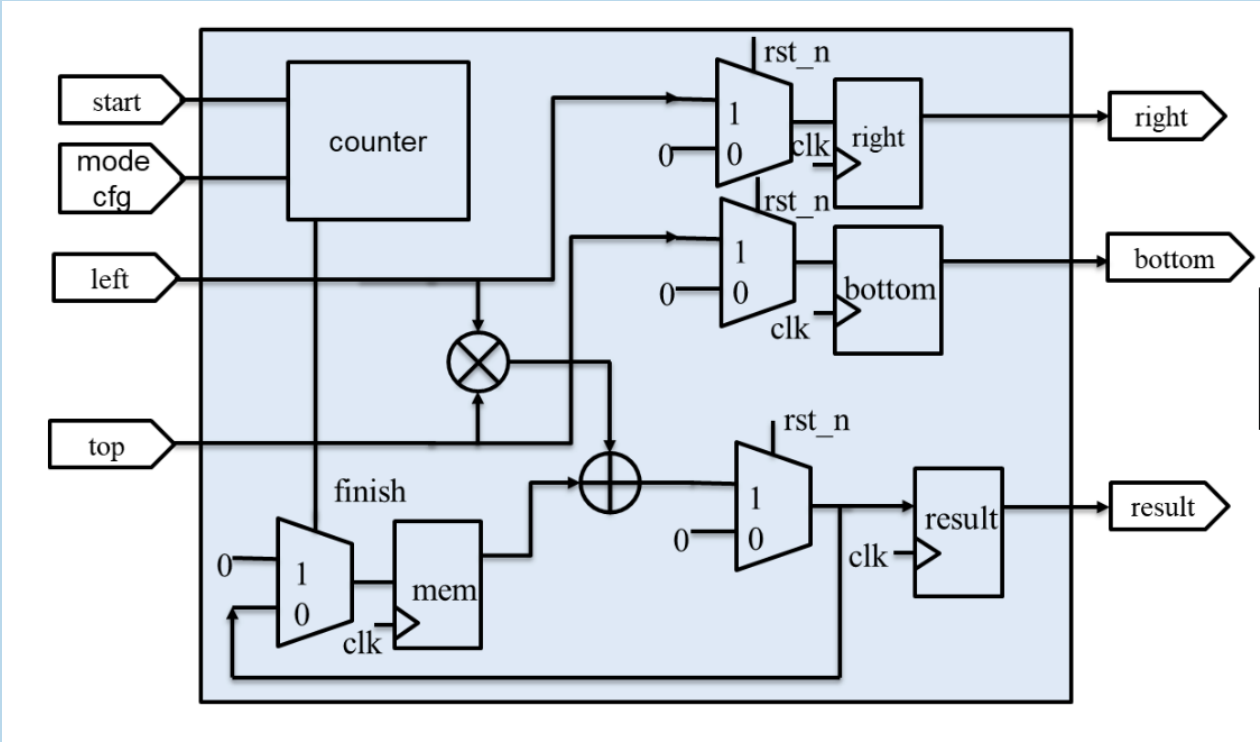


Single PE

Offline software toolchain feeds an on-chip RTL pipeline through a single 32-bit streaming load port.

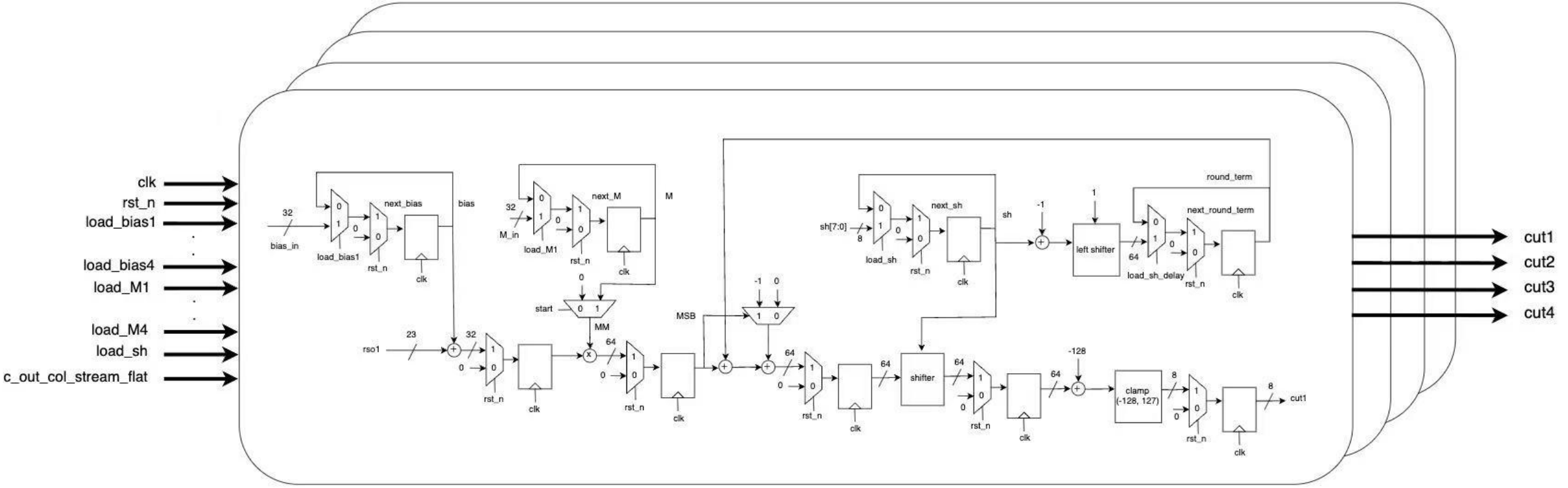
Mode reuse: same 16×4 fabric, four k values

Mode	Kernel	Acc times	Output shape
Conv1	L1: 3×3×1 → 4ch	9	4row -> 4ch
Conv2	L2: 3×3×4 → 8ch	36 (9*4ch)	4row * 2 passes -> 4ch*2
Conv3	L3: 3×3×8 → 8ch	72 (9*8ch)	4row * 2 passes -> 4ch*2



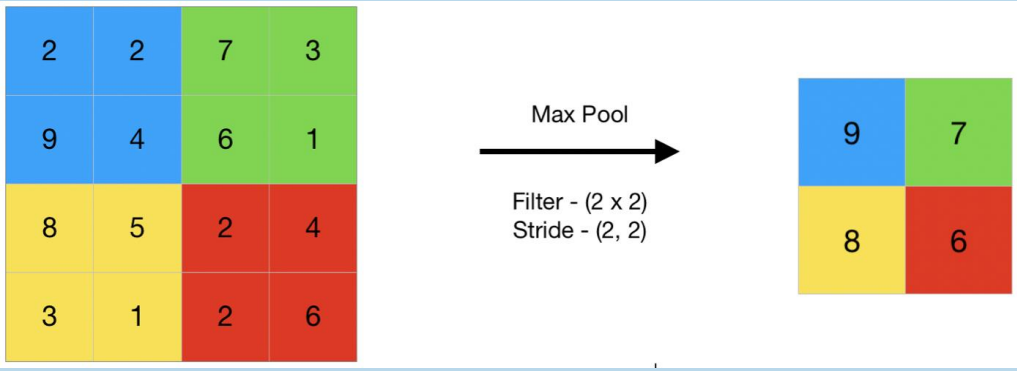
Quantization · 32-bit accumulator → 8-bit activation

4 PEs run in parallel — one INT8 result per systolic-array column per cycle.

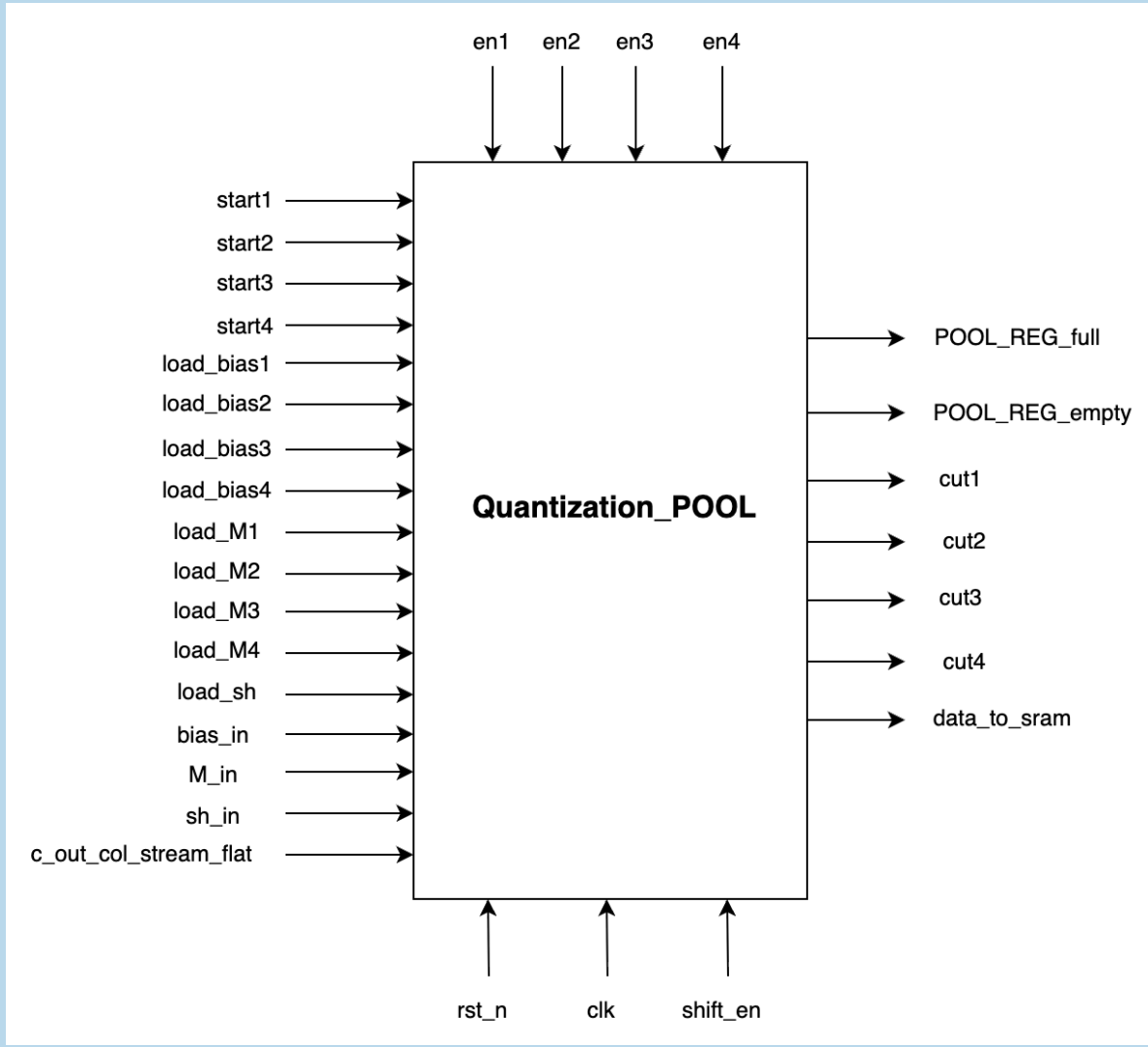


$$q_y = \text{clip}\left(\left(\underbrace{(\text{acc} + b)}_{\text{stage 1}} \cdot \underbrace{M}_{\text{stage 2}} + \underbrace{r - \text{sgn}}_{\text{stage 3}} \right) \underbrace{\gg \text{sh}}_{\text{stage 4}} - \underbrace{z}_{\text{stage 5}}, \; -128, \; 127 \right)$$

Quant + Max_Pooling



Layer	IN_Shape	Output shape
Layer1	62x62	31x31
Layer2	29x29	14x14
Layer3	12x12	6x6



Fully-Connected

Step 1: Bias Preload
- load_bias loads bias into bias_reg
- acc and counter are reset

→

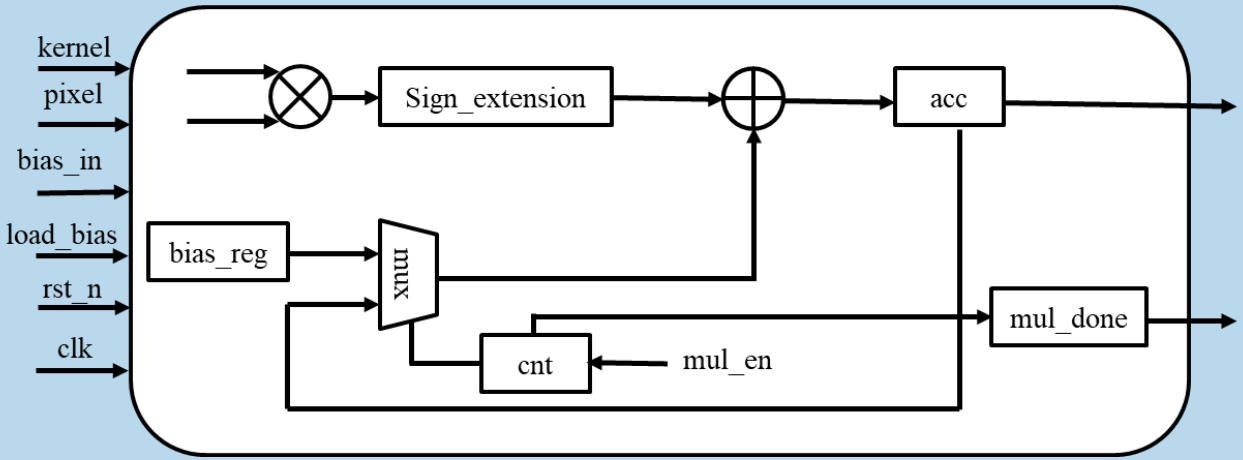
Step 2: MAC Operation (mul_en = 1)
- Each cycle: acc += pixel × kernel

→

Step 3: First Cycle Optimization
- acc = bias + product

→

Step 4: Completion
- After K = 288 cycles, mul_done is asserted



Cycle 0: load_bias

Cycle 1: acc = bias + product

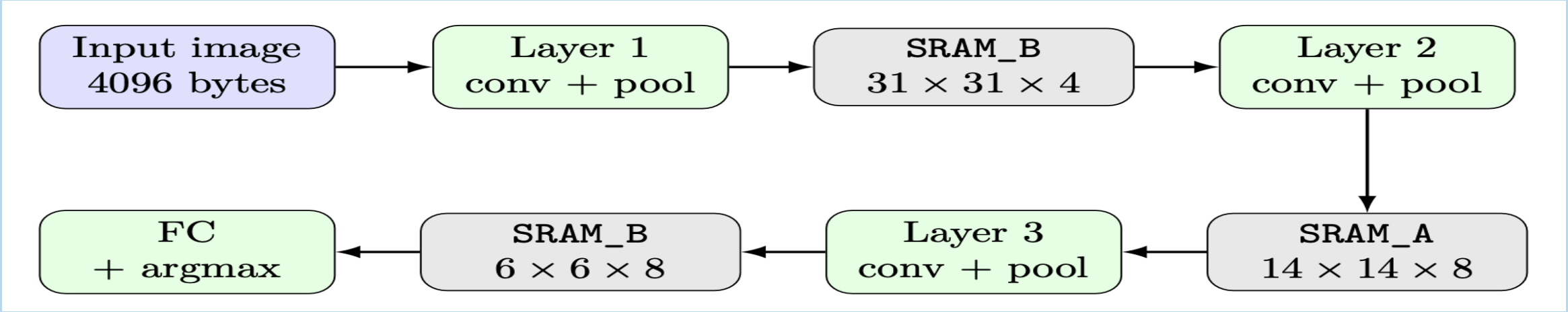
Cycle 2~287: acc += product

Cycle 288: mul_done = 1

Signal	Description
pixel	Input feature value
kernel	Weight value
bias_in	Bias for output channel
load_bias	Load bias into MAC
mul_en	Enable MAC operation

On-chip Memory & Reuse

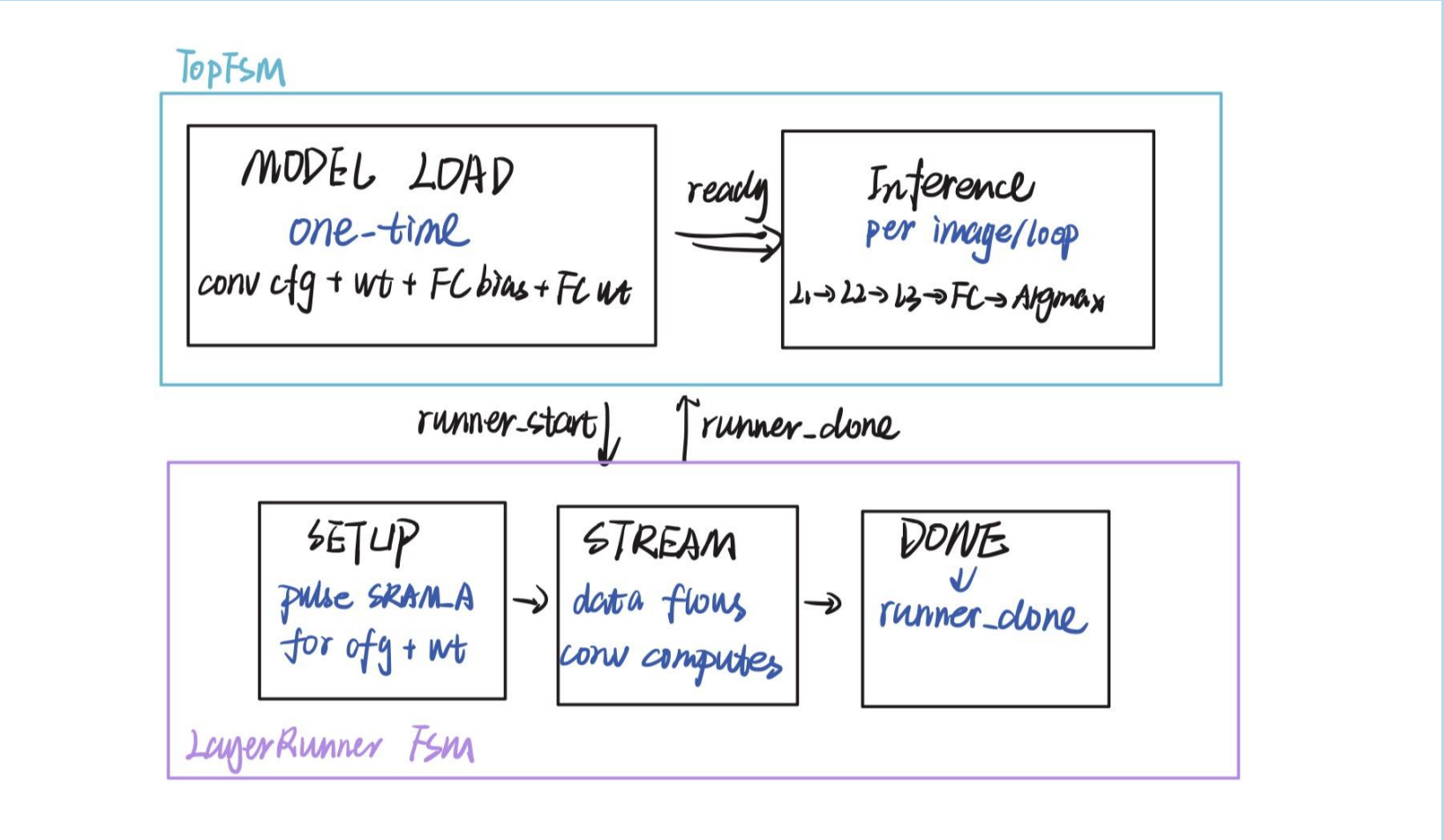
Three logical memories; SRAM_B is reused across L1, L3 outputs and as the FC input buffer.



Memory	Width * Depth	Stores	Reused as
SRAM_A	32-bit × 1024 (4 KB)	conv config + conv weights + FC bias	Layer-2 pooled output buffer
SRAM_B	32-bit × 1024 (4 KB)	Activation feature maps	L1 + L3 pooled outputs, FC input buffer
SRAM_FCW	80-bit × 288 used (~2.88 KB)	FC weights (10 INT8 per slot)	Drives 10 parallel FC MACs per cycle

Control FSMs · TopFSM + LayerRunnerFSM

Two-level state machines: TopFSM sequences the whole network, LayerRunnerFSM drives one layer's SRAM transaction.



Hardware/Software Interface — MMIO

32-bit Avalon-MM slave at HPS physical 0xff200000 (lightweight H2F bridge).

address[12] selects the space

`address[12] = 0`

Scratchpad BRAM
4094× 32-bit words (16 KiB)
host writes model + image

`address[12] = 1`

Register file
14 16-bit registers (control · status · base/len)
see next slide for register map

Signal	Dir	Width	Meaning
address	in	13	word addr; bit 12 = space selector
writedata	in	32	data written by HPS
byteenable	in	4	byte write enables
write	in	1	write strobe
read	in	1	read strobe
chipselect	in	1	transaction qualifier
readdata	out	32	data returned to HPS

Register Map

32 32-bit registers in the address[12] = 1 space — low 16 bits keep the original control/status ABI.

Core registers · idx 0–13

Reg	idx	Access	Bit fields / description
CONTROL	0	WO	[0] start_model_load [1] start_inference [2] clear status
STATUS	1	RO	[1] busy [2] model_loaded [3] inf_done [7:4] predicted_class
CONV_CFG_BASE	2	RW	scratchpad base — 32-bit word index, conv config segmen
CONV_CFG_LEN	3	RW	length in 32-bit words — conv config
CONV_WT_BASE	4	RW	scratchpad base — 32-bit word index, conv weight segment
CONV_WT_LEN	5	RW	length in 32-bit words — conv weights
FC_BIAS_BASE	6	RW	scratchpad base — 32-bit word index, FC bias segment
FC_BIAS_LEN	7	RW	length in 32-bit words — FC bias
FC_BASE	8	RW	scratchpad base — 32-bit word index, FC weight segment
FC_LEN	9	RW	length in 32-bit words — FC weights
IMAGE_BASE	10	RW	scratchpad base — 32-bit word index, input image segment
IMAGE_LEN	11	RW	length in 32-bit words — input image
PREDICT	12	RO	[3:0] latched predicted class (0–9)
IF_ERROR	13	RO	[0] load while busy [1] inf while busy [2] inf before model [3] addr OOR

Core registers · idx 14–31

Stage	LO idx	HI IDX	Holds
Profile_L1	14	15	conv L1 cycles
Profile_L2_P0	16	17	conv L2 pass-0 cycles
Profile_L2_P1	18	19	conv L2 pass-1 cycles
Profile_L3_P0	20	21	conv L3 pass-0 cycles
Profile_L3_P1	22	23	conv L3 pass-1 cycles
Profile_FC	24	25	FC cycles
Profile_Argmax	26	27	Argmax cycles
Profile_Total	28	29	End to end cycles

Diagnstics

LAST_WRITE	30	RO	Byte enable[3:0], last_writedata[27:0]} · echo of most recent host write
MAGIC	31	RO	0x434E4E32 ("CNN2") · sanity check / driver handshake

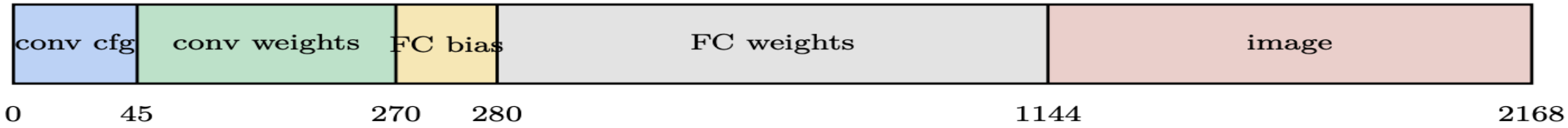
RO = read-only **WO** = write-only **RW** = read/write
CONTROL[0] kicks model preload · CONTROL[1] kicks inference · CONTROL[2] clears status & error

Scratchpad Layout & Replay

Host writes payloads, sets CONTROL bit, wrapper walks regions and streams into system_top.

Table 9: Default scratchpad layout for the active 10-class model.

Region	Base word	Words	Bytes	Content
`conv_cfg`	0	45	180	layer/pass configuration words
`conv_wt`	45	225	900	packed convolution weights
`fc_bias`	270	10	40	10 FC bias words
`fcw`	280	864	3456	packed fully connected weights
`image`	1144	1024	4096	packed 64×64 INT8 image



Replay path *(triggered by CONTROL[0] for model, CONTROL[1] for inference)*

`scratchpad BRAM` → replay engine walks the configured base/len regions in order → `load_data, load_valid, load_last` → `system_top`

Performance & Resources

Cyclone V (5CSEMA5F31C6), 50 MHz fabric clock — single inference profile.

Board CPU Time	FPGA Wait Time
5.18 ms	**1.11 ms**
FPGA RTL Time	**Speedup**
0.325 ms	**4.65x**

RTL 325.4 us from on-chip counters. Board wait 1.11 ms. Remote request 283.92 ms. Board total 5.08 ms, load 3.38 ms, program 0.55 ms.

; Slow 1100mV 85C Model Fmax Summary

Fmax	Restricted Fmax	Clock Name
72.98 MHz	72.98 MHz	clock_50
1184.83 MHz	717.36 MHz	soc_system:u_soc_system|soc_system_hps_0:hps_0|soc_

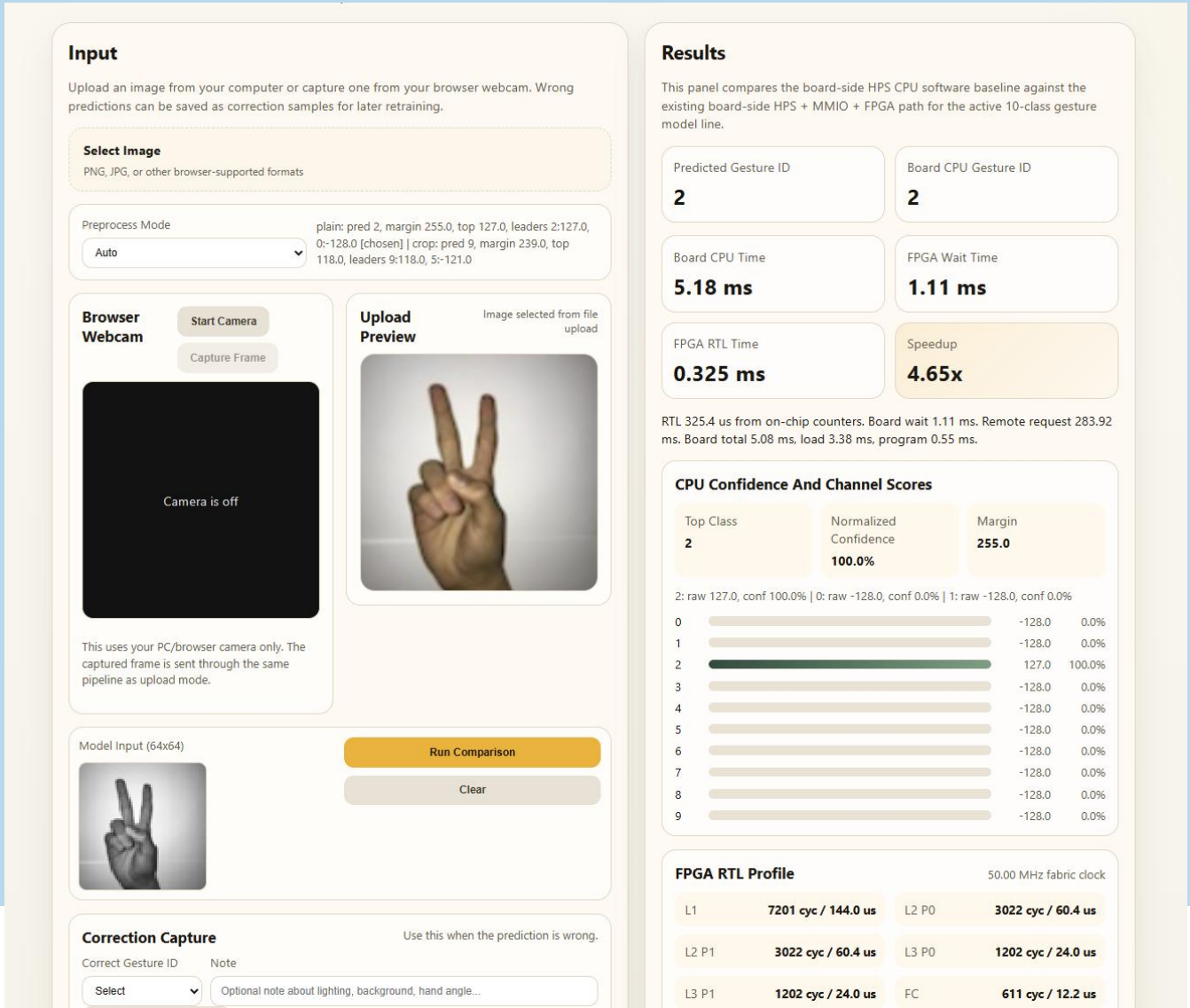
This panel reports FMAX for every clock in the design, regardless of the user-specifi

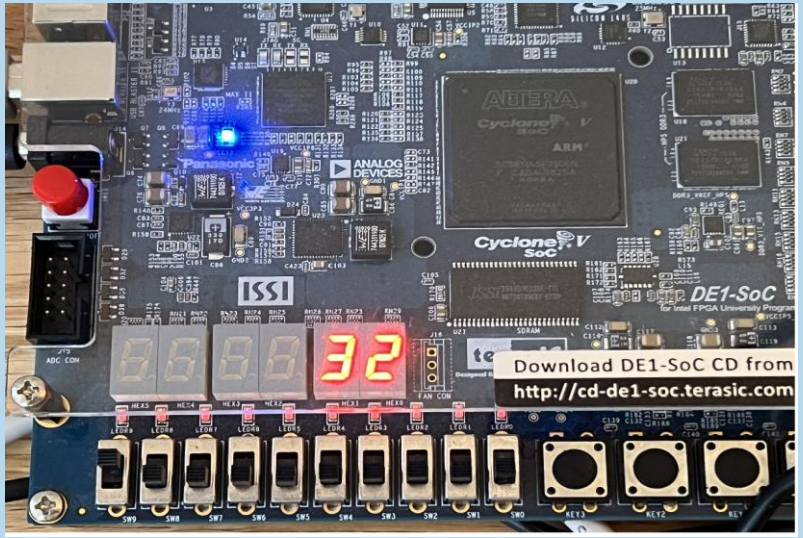
; Fitter Summary

Fitter Status	Successful - Mon May 11 18:26:35 2026
Quartus Prime Version	21.1.0 Build 842 10/21/2021 SJ Lite Edition
Revision Name	soc_system
Top-level Entity Name	soc_system_top
Family	Cyclone V
Device	5CSEMA5F31C6
Timing Models	Final
Logic utilization (in ALMs)	18,357 / 32,070 (57 %)
Total registers	26916
Total pins	164 / 457 (36 %)
Total virtual pins	0
Total block memory bits	241,664 / 4,065,280 (6 %)
Total RAM Blocks	29 / 397 (7 %)
Total DSP Blocks	86 / 87 (99 %)
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI PMA TX Serializers	0
Total PLLs	0 / 6 (0 %)
Total DLLs	1 / 4 (25 %)

Validation

Per-layer MATLAB golden reference matches INT8 TFLite tensors; CPU and FPGA agree on every test digit.





HEX1 = 3, HEX0 = 2 → model loaded + predict_done, predicted class = 2

HEX	Source	Meaning
HEX0	`predict_class[3:0]`	predicted gesture (after predict_done)
HEX1	`{model_loaded, predict_done}`	0=idle 2=loaded 3=loaded+done
HEX2	`interface_error[3:0]`	low nibble of error code
HEX5	`constant 'E'`	marker — only when IF_ERROR ≠ 0