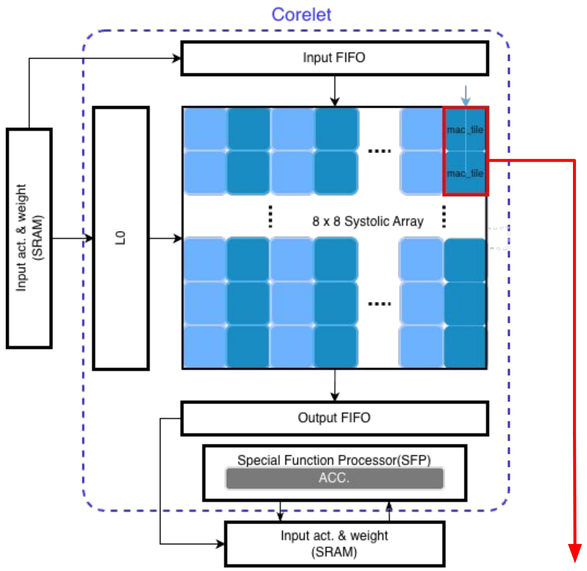
Energy-Efficient Reconfigurable Systolic Array with Huffman Compression and Clock Gating

VLSI Scream

Chi-Han Chiu, Gary(Kai-Jui) Weng, Yun-Chen Tsai, Bing-Cheng Chiang, Yufan Wang

Foundation

- **Combined Reconfigurability (Alpha 1)**



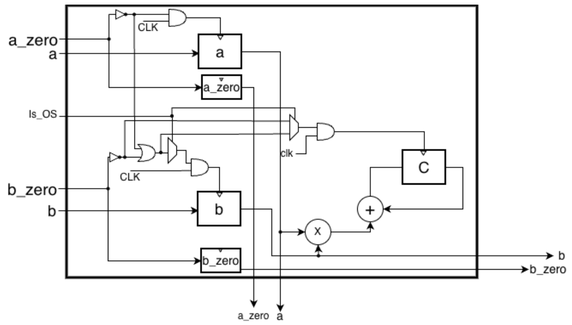
	ws_2bit	ws_4bit	os_2bit	os_4bit
in_n	psum	psum	{8'b0, w1, w0}	{12'b0, w}
in_w	act, w0, w1	act, w	act	act

Implemented on Cyclone 10 GX

	Vanilla	Mixed
Frequency	125 MHz	125 MHz
Total Register	12,155	12,667
Total DSP	60	64
LUT	8,424	9,062
Data Delay	7.571 ns	7.205 ns
Power	1119.85 mW	1256.2 mW

Alpha 2 - Clock Gating

- **Less power consumption than no clock gating design.**



	Activation	Weight
Sparsity	86 %	16 %

Table 1. Sparsity of Activation and Weight

$$Sparsity_{new} = Sparsity_{Act.} + Sparsity_{Wgt.} - Sparsity_{Act.} \times Sparsity_{Wgt.}$$
$$= 86\,\% + 16\,\% - 86\,\% \times 16\,\%$$
$$= 88.24\,\%$$

- **Our design can reach maximum of 88.24% power reduction. (may be various to different FPGA.)**
- **To maximize the performance of this technique, the next target will be to enable fully shut down of the transistors.**

Alpha 3 - Huffman Compression

Value	0000	0001	0010	0011	0100	0101	0110	0111
Count	248	11	10	5	4	4	4	2
Save (bits)	-744	-22	-10	0	+4	+8	+12	+10

$$4 \times 288 - \sum Save = 410$$

- **With Huffman Compression, we can get a 36.11% bit reduction for activation.**

Alpha 4 - Whole Conv. Layer

$$y = \frac{x - \mathrm{E}[x]}{\sqrt{\mathrm{Var}[x] + \epsilon}} * \gamma + \beta$$

- **Implementing Batch Normalization, ReLU, and Max Pooling Layer into our proposed 2-D systolic array.**
- **With these layers, it is able to fully go through the whole VGG16 CNN.**

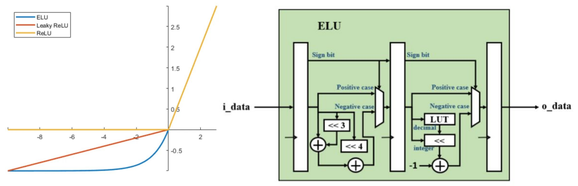
Alpha 5 - Model Fusion

$$\begin{cases} x_{conv} = w_{conv} * x_{in} + b \\ x_{bn} = \dfrac{(x_{conv} - \mu)}{\sqrt{\sigma^2 + \epsilon}}\gamma + \beta \end{cases} \qquad \begin{cases} w' = \dfrac{\gamma}{\sqrt{\sigma^2 + \epsilon}} w_{conv} \\ b' = \dfrac{b - \mu}{\sqrt{\sigma^2 + \epsilon}}\gamma + \beta \end{cases}$$

$$x_{bn} = w' * x_{in} + b'$$

- **BN parameters are fixed at inference and fused into adjacent Convolution layers, eliminating runtime normalization.**
- **This reduces computation by 4.3% and hardware cost by 25%, while simplifying datapath design and improving efficiency.**

Alpha 6 - Flex Act. Func.

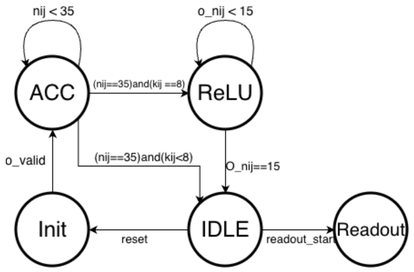


- **Includes a flexible interface that supports all three of the most widely adopted activation functions—ReLU, Leaky ReLU, and ELU.**

$$\exp(x) = 2^{\log_2 \exp(x)} = 2^{\frac{x}{\ln 2}}$$

- **The right module is our implemented ELU Function. Since e^x is not hardware-friendly and its direct implementation incurs excessive area overhead, it is transformed into the above equivalent form.**

Alpha 7 - FSM in SFU



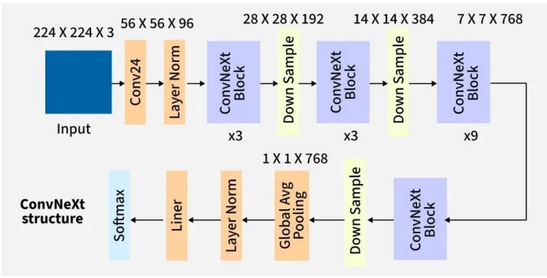
- **TB has no control to PSUM memory.**
- **Inst_w is pipelined appropriately in core.v.**

```
// high level instructions from TB
input   [1:0]   inst_w,

// L0 mem ctrls from TB
input           CEN_xmem,
input           WEN_xmem,
input   [10:0]  A_xmem,
input   [bw*col-1:0] D_xmem, //32b

// from/to TB
input [3:0]     kij,  //for SFU
input           readout_start, //trigger for output stage
output  [psum_bw*col-1:0] readout //16*8b
```

Alpha 8 - ConvNext Application



- **We aim to implement and evaluate ConvNeXt, a modern architecture known for achieving the same accuracy (~90%, 4-bit) with significantly fewer parameters.**

	VGGNet	ConvNext
model size	68.7 MB	33.0 MB