On-chip Memory Based Binarized Convolutional Deep Neural Network Applying Batch Normalization Free Technique on an FPGA

2018-10-27

Overview

- 2017 IEEE International Parallel and Distributed Processing Symposium Workshops
- 因为想到BN不好在FPGA上执行找了下相关文章。这篇文章描述了在推断过程中对于部署在FPGA上的全二值化网络如何进行一种Batch Normalization Free的硬件友好操作来处理BN，还是有借鉴意义的
- Code： https://github.com/itayhubara/BinaryNet （Shift base ）

Introduction

- In this paper, we propose a batch normalization free binarized CNN which is mathematically equivalent to one using batch normalization. The proposed CNN treats the binarized inputs and weights with the integer bias.
- BN Free是一种与BN操作数学等价的操作：需要二值化的Input和Weight和一个bias

BN Free

- BN操作造成的影响：In that case, the additional multiplication and addition require more hardware, while the memory access for its parameters reduces system performance and increases power consumption.

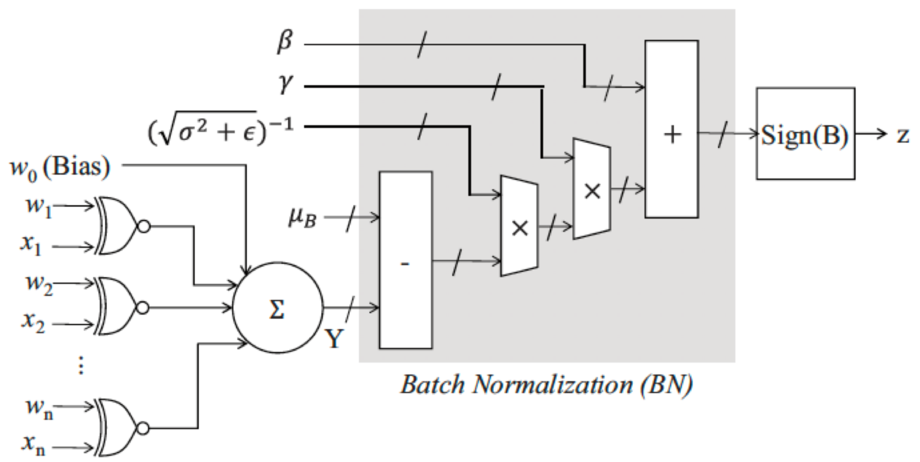

Fig. 4. Binarized AN with batch normalization (BN).

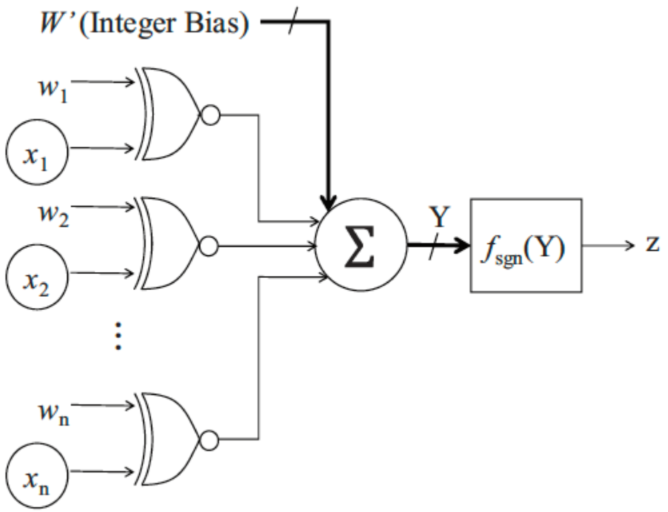

Fig. 7. BN free binarized AN.

BN Free

问题有两个：1.这个Bias是怎么算完存到On-Chip Memory里面的？（shfit base BN？ Code：https://github.com/itayhubara/BinaryNet ）

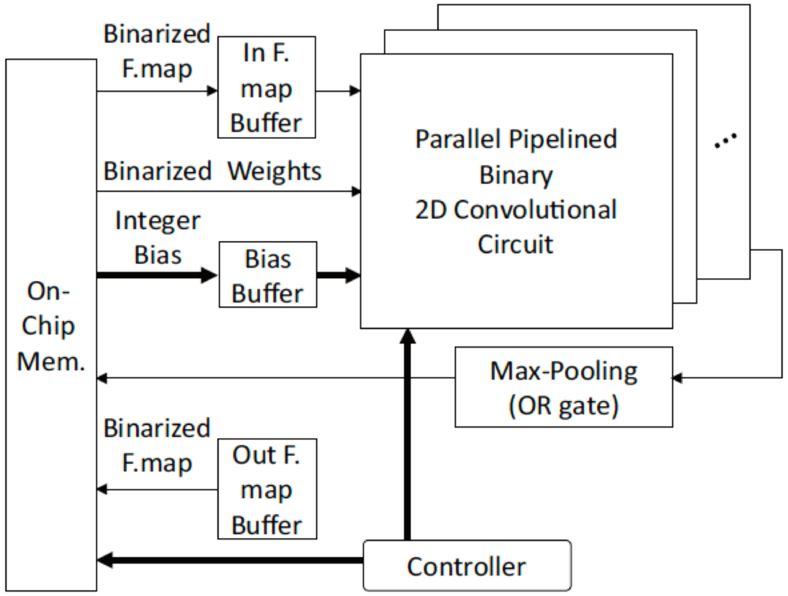

Fig. 10. Overall Architecture.

As shown in Algorithm 3.1, the BN normalizes the internal variables Y. Let Y' be the output of the BN operation. Then, we have

$$\begin{aligned} Y' &= \gamma \frac{Y - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta \\ &= \frac{\gamma}{\sqrt{\sigma_B^2 + \epsilon}} \left(Y - \left(\mu_B - \frac{\sqrt{\sigma_B^2 + \epsilon}}{\gamma} \beta \right) \right). \end{aligned}$$

From above expression, the signed activation function becomes

$$f'_{sgn}(Y) = \begin{cases} 1 & \left(if\ Y < -\mu_B + \frac{\sqrt{\sigma_B^2+\epsilon}}{\gamma} \beta \right) \\ -1 & (otherwise) \end{cases}$$

That is, the value of the active function is determined by the value of the above equation. In this case, since $x_0 = 1$, Y can be equivalent to the following expression:

$$\begin{aligned} Y &= \sum_{i=0}^{n} w_i x_i - \mu_B + \frac{\sqrt{\sigma_B^2 + \epsilon}}{\gamma} \beta \\ &= \sum_{i=1}^{n} w_i x_i + \left(w_0 - \mu_B + \frac{\sqrt{\sigma_B^2 + \epsilon}}{\gamma} \beta \right) \\ &= \sum_{i=1}^{n} w_i x_i + W'. \end{aligned} \tag{4}$$

BN Free

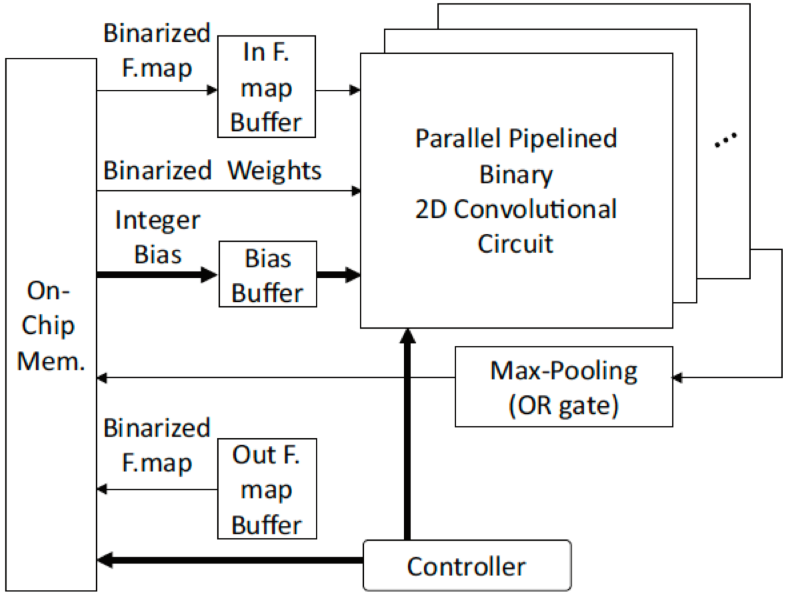

Fig. 10. Overall Architecture.

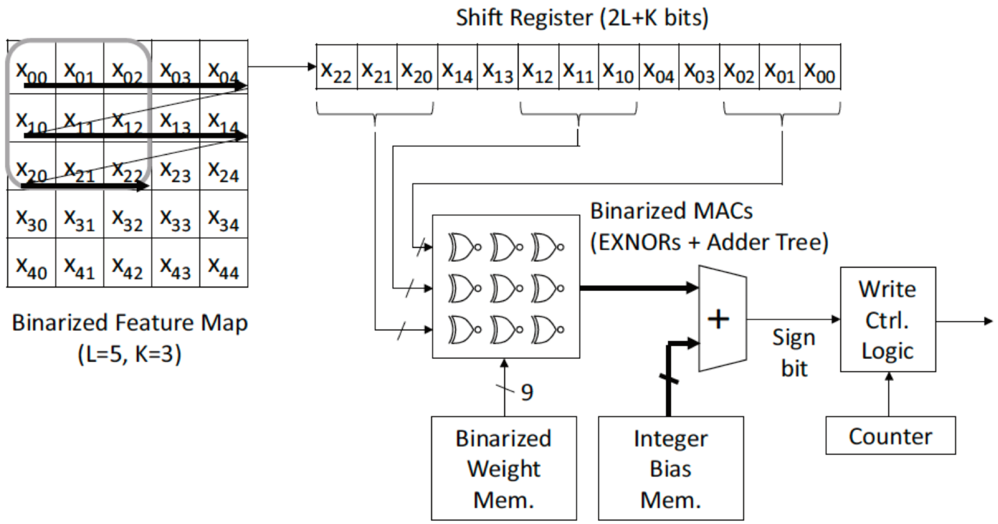

Fig. 8. Pipelined Binary 2D Convolutional Circuit.

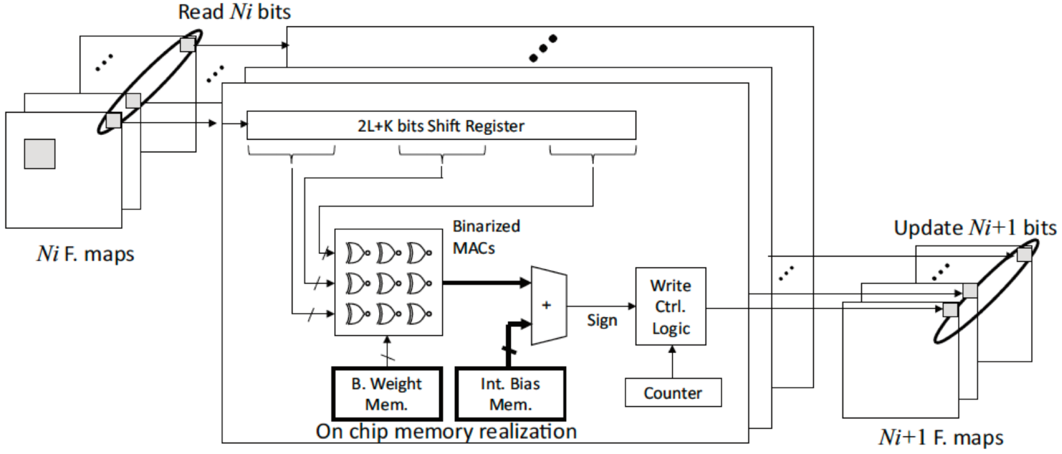

Fig. 9. Parallel Pipelined Binary 2D Convolutional Circuit.

BN Free

TABLE III. COMPARISON WITH OTHER FPGA REALIZATIONS.

Year	2010 [2]	2014 [15]	2015 [38]	2016 [35]	Proposed
FPGA	Virtex5 SX240T	Zynq XC7Z045	Virtex7 VX485T	Zynq XC7Z045	Zynq UltraScale+ MPSoC ZU9EG
Clock (MHz)	120	150	100	150	150
Memory Bandwidth (GB/s)	—	4.2	12.8	4.2	**139.6**
Quantization Strategy	48bit fixed	16bit fixed	32bit float	16bit fixed	1bit (Binary)
Power (W)	14	8	18.61	9.63	22
Performance (GOPS)	16	23.18	61.62	187.80	**460.80**
Area Efficiency $\times 10^{-4}$) (GOPS/Slice)	4.30	—	8.12	35.8	**96.1**
Power Efficiency (GOPS/W)	1.14	2.90	3.31	19.50	**20.94**

TABLE IV. COMPARISON WITH EMBEDDED PLATFORMS WITH RESPECT TO THE VGG16 FORWARDING (BATCH SIZE IS 1).

Platform	Embedded CPU	Embedded GPU	FPGA
Device	Quad-core ARM Cortex-A57	256-core Maxwell GPU	Zynq UltraScale+ MPSoC
Clock Freq.	1.9 GHz	998 MHz	150 MHz
Memory	16GB eMMC Flash	4GB LPDDR4	32.1 Mb BRAM
Time [msec] (FPS)	4210.0 (0.23)	156.1 (6.40)	**31.8** **(31.48)**
Power [W]	7	17	22
Efficiency [fps/W]	0.032	0.376	**1.431**