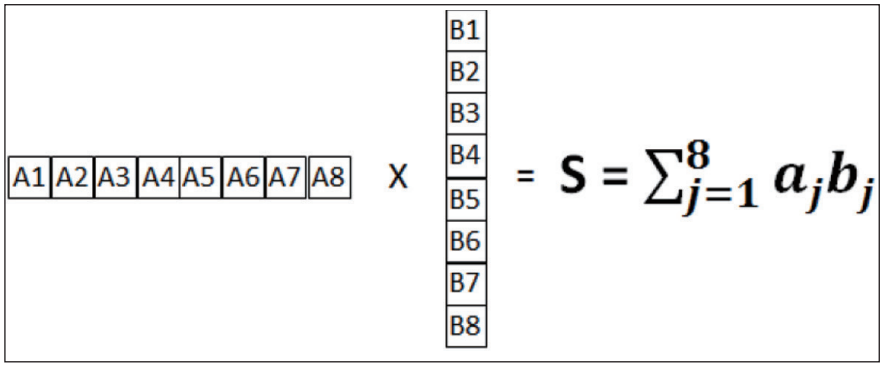
詳解 FPGA 如何實現 FP16 格式的點積運算實例

■作者：楊宇 /Achronix 資深現場應用工程師

通過使用 Achronix Speedster7t FPGA 中的機器學習加速器 MLP72，開發人員可以輕鬆選擇浮點 / 定點格式和多種位元寬，或快速應用塊浮點，並通過內部級聯 (cascad) 可以達到理想性能。

神經網路架構中的核心之一就是卷積層 (convolutional layer)，卷積的最基本操作就是點積。向量乘法的結果是向量的每個元素的總和相乘在一起，通常稱之為點積 (dot product)。此向量乘法如圖 1 所示：

圖 1：點積操作



該總和 S 由每個向量元素的總和相乘而成，因此 $S = a_1 b_1 + a_2 b_2 + a_3 b_3 + \cdots$

本文講述的是使用 FP16 格式的點積運算實例，展示了 MLP72 支援的數位類型和乘數的範圍。

此設計實現了同時處理 8 對 FP16 輸入的點積。該設計包含四個 MLP72，使用 MLP 內部的級聯路徑連接。每個 MLP72 將兩個並行乘法的結果相加 (即 $a_i b_i + a_{i+1} b_{i+1}$)，每個乘法都是 i_a 輸入乘以 i_b 輸入 (均為 FP16 格式) 的結果。來自每個 MLP72 的總和沿著 MLP72 的列級聯到上面的下一個 MLP72 塊。在最後一個 MLP72 中，在每個週期上，計算八個並行 FP16 乘法的總和。

最終結果是多個輸入週期內的累加總和，其中累加由 i_first 和 i_last 輸入控制。i_first 輸入信號指示累加和歸零的第一組輸入。i_last 信號指示要累加和加到累加的最後一組輸入。最終的 i_last 值可在之後的六個週期使用，並使用 i_last o_valid 進行限定。兩次運算之間可以無空拍。

配置說明

表 1：FP16 點積配置表

Input Format	Output Format	Parallel Multiplications	Number of MLP72
bf16	bf16	8	4

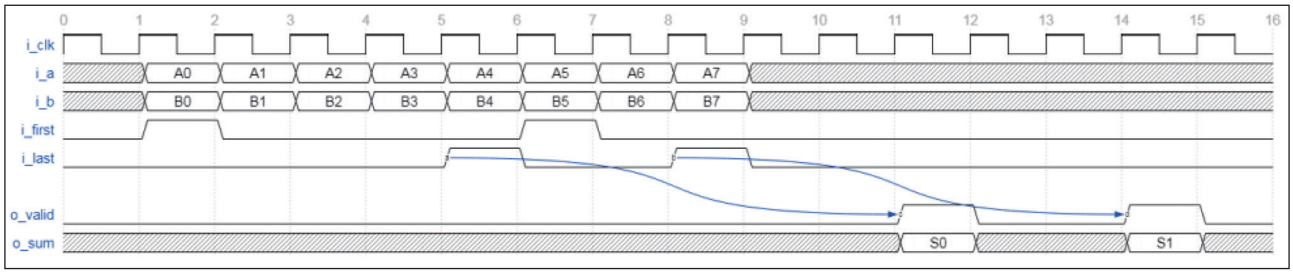
埠說明

表 2：FP16 點積埠說明表

Port	Direction	Description
i_a	Input	"a" input to multiplication. Array of 4 x 2 x fp16 (128 bits).
i_b	Input	"b" input to multiplication. Array of 4 x 2 x fp16 (128 bits).
i_first	Input	Indicates first group of inputs to sum and start accumulation. Sets internal accumulator to 0.
i_last	Input	Indicates last group of inputs to sum and accumulate.
o_sum	Output	fp16 accumulated output.
o_valid	Output	Validates o_sum output.

時序圖

圖 2：FP16 點積時序圖



其中，

- A0 = Array of N × a inputs, i.e., $\{a_1, a_2, a_3 .. a_n\}[0]$
- B0 = Array of N × b inputs, i.e., $\{b_1, b_2, b_3 .. b_n\}[0]$
- S0 = Sum of array of multiplications, = $\{a_1*b_1 + a_2*b_2 + a_3*b_3 + ... a_n*b_n\}[0] + \{a_1*b_1 + a_2*b_2 + a_3*b_3 + ... a_n*b_n\}[1] + \{a_1*b_1 + a_2*b_2 + a_3*b_3 + ... a_n*b_n\}[2] + ...$

圖 3：MLP 進位鏈

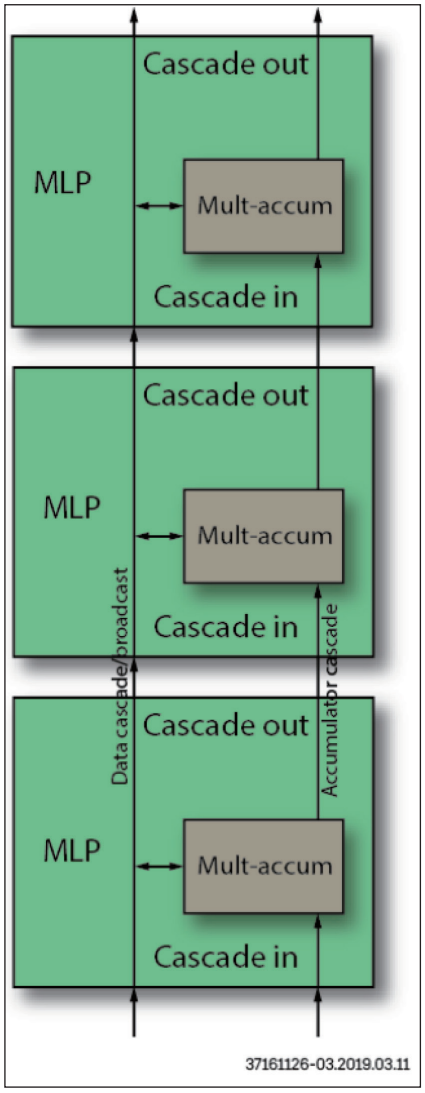
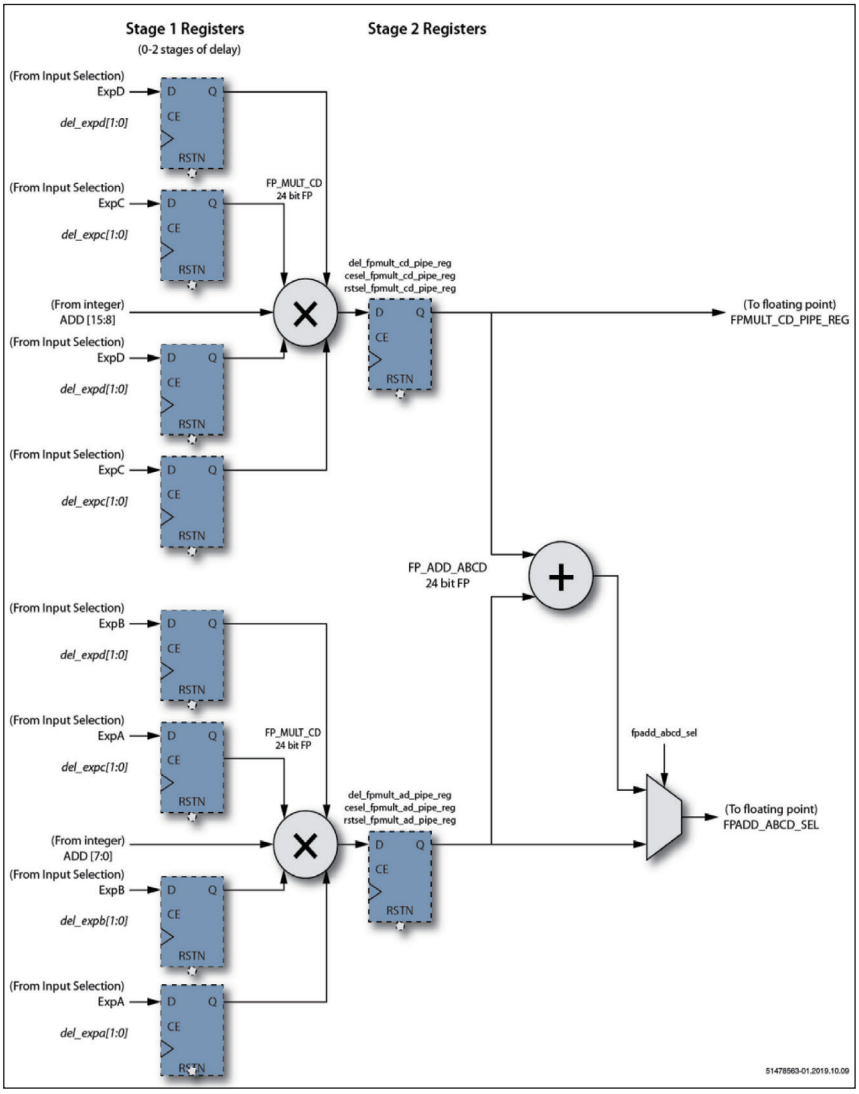


圖 4：MLP 乘法功能階段框圖



那麼，以上運算功能如何對應到 MLP 內部呢？其後的細節已分為 MLP72 中的多個功能階段進行說明。

進位鏈

首先請看圖 3，MLP 之間的進位元鏈結構，這是 MLP 內部的專用走線，可以保證級聯的高效執行。

乘法階段

圖 4 是 4MLP 乘法功能階段框圖，圖 5 是 MLP 中浮點乘法功能階段，其中寄存器代表一級可選延遲。

MLP72 浮點乘法級包括兩個 24 位全浮點乘法器和一個 24 位全浮點加法器。兩個乘法器執行 A×B 和 C×D 的平行計算。加法器將兩個結果相加得到 A×B + C×D。

乘法階段有兩個輸出。下半部分輸出可以在 A×B 或 (A×B + C×D) 之間選擇。上半部分輸出始終為 C×D。

圖 6：FP16 點積邏輯框圖

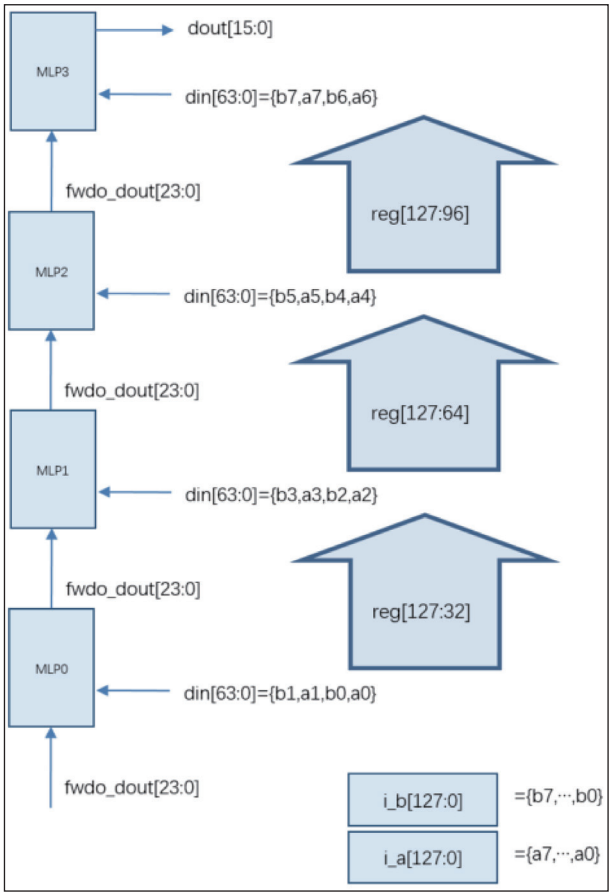
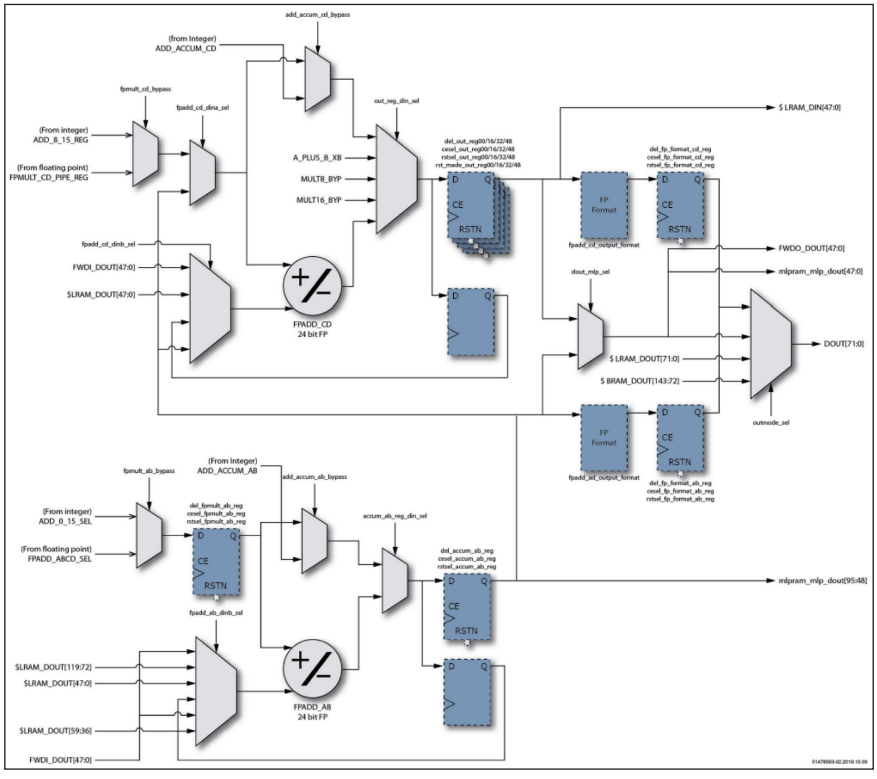


圖 5：MLP 浮點輸出階段框圖



乘法器和加法器使用的數位格式由位元組選擇參數以及和參數設置的格式確定。

浮點輸出具有與整數輸出級相同的路徑和結構。MLP72 可以配置為在特定階段選擇整數或等效浮點輸入。輸出支持兩個 24 位全浮點加法器，可以對其進行加法或累加配置。進一步可以載入加法器 (開始累加)，可以將其設置為減法，並支援可選的舍入模式。

最終輸出階段支持將浮點輸出格式化為 MLP72 支持的三種浮點格式中的任何一種。此功能使 MLP72 可以外部支援大小一致的浮點輸入和輸出 (例如 fp16 或

圖 7：FP16 點積在 MLP 內部資料流程圖

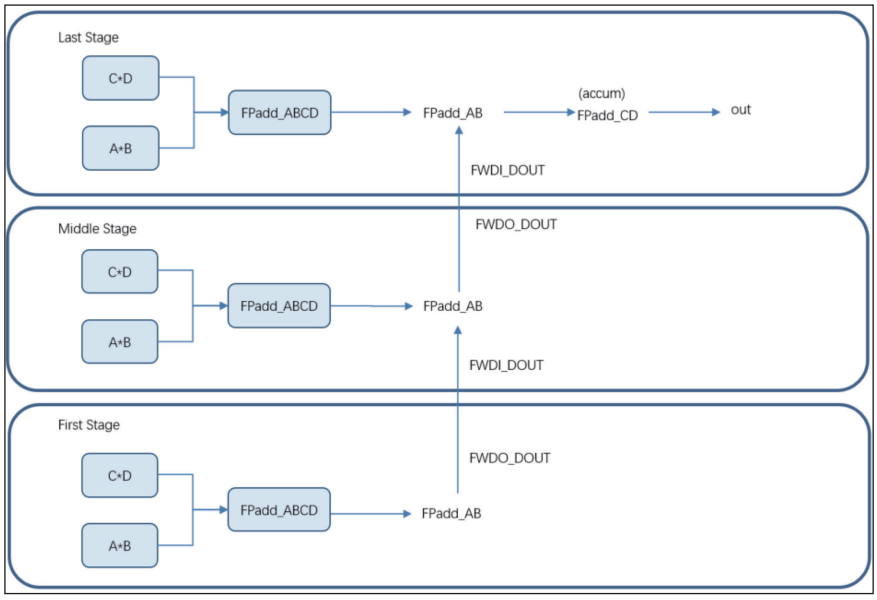


表 1

Collective Summary of All Corners					
	Slack (ns)		Frequency (MHz)		
Clock / Group	setup	hold	Target	Upper Limit	Comment
clk	0.217	0.035	750.0	895.6	---

bfloat16)，而在內部以 fp24 執行所有計算。

需要強調的是本設計輸入和輸出都是 FP16 格式，中間計算過程，即進位鏈上的 fwdo_out 和 fwdi_dout 都是 FP24 格式。具體邏輯框圖如圖 6 所示：

MLP 內部資料流程示意圖：

圖 7 為 FP16 點積在 MLP 內部資料流程圖，最終 ACE 的時序結果如表 1：

如需瞭解更多產品細節，請發送郵件到 Dawson.Guo@Achronix.com。CTA

趨勢科技與米蘭理工大學共同發表 OT 程式開發基本安全原則

趨勢科技日前發表一份新的研究報告指出老舊程式語言的設計缺陷，同時也提出一些程式設計安全原則來協助工業 4.0 開發人員大幅減少軟體的受攻擊面，希望藉此降低營運技術 (OT) 環境中斷營運的情況。

這份趨勢科技與米蘭理工大學 (Politecnico di Milano) 合作的研究顯示一些老舊程式語言的設計缺陷如何造成自動化程式的漏洞。這些資安缺陷可讓駭客挾持工業機器人與自動化設備，進而造成生產線中斷或智慧財產遭竊。根據這項研究指出，工業自動化領域目前還不知該如何偵測及防範這類漏洞攻擊，因此，業界有必要開始建立和實施一些網路資安與程式設計安全實務原則。針對這點，這份研究也提出了一些新的原則。

一些老舊專屬系統的程式設計語言，如：RAPID、KRL、AS、PDL2 及 PacScript 在當初設計時都未設想到駭客可能的攻擊方式。這些幾十年前的產物，現在卻成了今日工廠自動化的重要關鍵。問題是它們無法輕易修補漏洞，這些使用專屬語言撰寫的自動化程式不僅可能存在漏洞的問題，研究人員還證明，其中的某個語言可能被用來製作一種新型的自我複製惡意程式。

Trend Micro Research 與米蘭理工大學還共同開發了一套專利申請中的工具來偵測工作程式中的漏洞或惡意程式碼，希望藉此防範執行時期問題。

如需更進一步的資訊，請參閱：https://www.trendmicro.com/vinfo/us/security/news/internet-of-things/unveiling-the-hidden-risks-of-industrial-automation-programming