在 Embedded Workbench 開發工具中

如何實現堆疊保護來提高代碼的安全性

■作者：IAR Systems

隨著越來越多的嵌入式產品連接到外部網路，嵌入式產品的資訊安全性 (Security) 越來越多地被人們關注。其中既包括直接連接到外部網路，比如通過 Wi-Fi 連接；也包括間接連接到外部網路，比如汽車中的 ECU 通過 CAN 匯流排與 T-box 相連，而 T-box 通過移動網路可以連接到外部網路。特別是對於一些高功能安全性 (Safety) 要求的產品，如工業，汽車，醫療產品等，資訊安全成為了功能安全的前提。

在 C/C++ 中，堆疊緩存溢出 (Stack Buffer Overflow) 是一種常見的錯誤：當程式往堆疊緩存 (Stack Buffer) 寫資料時，由於堆疊緩存通常採用固定長度，如果需要寫的資料長度超過堆疊緩存的長度時，就會造成堆疊緩存溢出。堆疊緩存溢出會覆蓋堆疊緩存臨近的堆疊資料，其中可能包含函數的返回位址，就會造成函數返回時異常。如果堆疊緩存溢出是攻擊者利用代碼的漏洞蓄意造成的，它就稱為堆疊粉碎 (Stack Smashing)。堆疊粉碎是常用的一種攻擊手段。

堆疊金絲雀 (Stack Canaries)，因其類似於在煤礦中使用金絲雀來感測瓦斯等氣體而得名，它可以用於在函數返回之前檢測堆疊緩存溢出來實現堆疊保護 (Stack Protection)，從而提高代碼的安全性。

相對於很多更加關注發揮器件性能的原廠開發工具，一些在行業中被廣泛使用的商用開發工具更加關注性能和安全性的平衡性和完整性。本文以過去數十年來在行業中被廣泛採用的商用工具鏈 IAR Embedded Workbench 為例，介紹如何在工具中實現堆疊保護，從而提高代碼的安全性。

堆疊粉碎

在 C/C++ 中，堆疊 (Stack) 用於保存程式正常運行 (比如函式呼叫或者中斷搶佔) 的臨時資料，可能包含如下資料：

■沒有儲存在寄存器中的函數參數和區域變數

■沒有儲存在寄存器中的函數返回值和函數返回位址

■CPU 和寄存器狀態

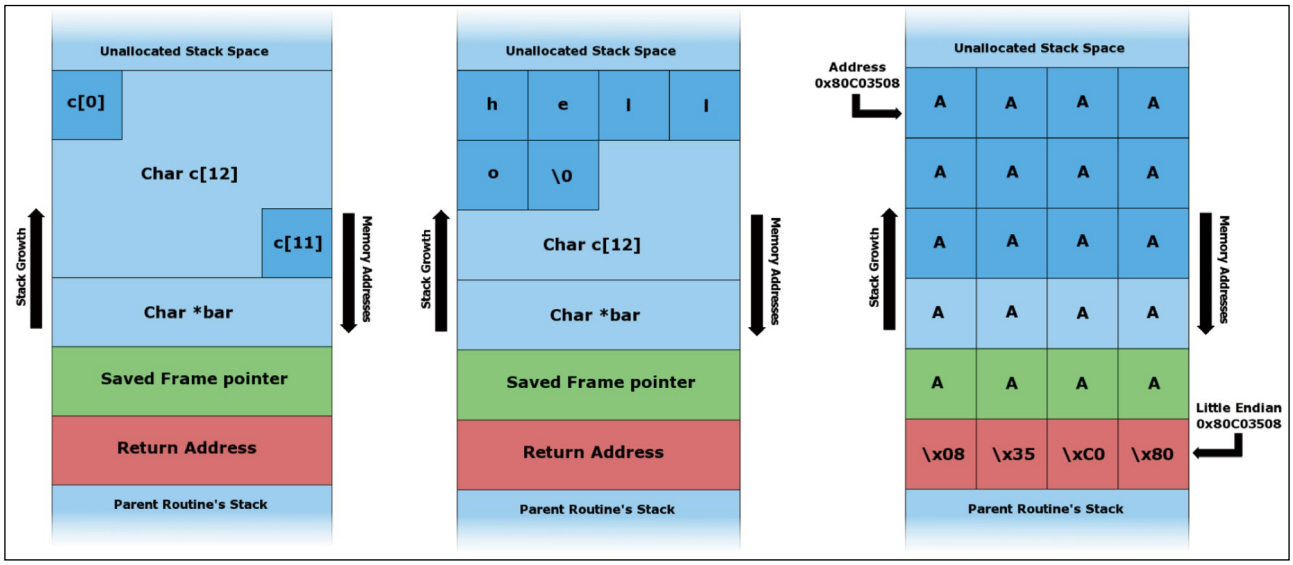
由於堆疊保存的是保證程式正常運行的臨時資料，堆疊緩存溢出會覆蓋堆疊緩存臨近的堆疊資料，這些資料可能包含函數的返回位址，如果發生時一般會造成程式運行異常。攻擊者經常利用這一點來進行堆疊粉碎攻擊。

下面通過一個簡單的例子來說明堆疊粉碎攻擊：

```
void foo(char *bar)
{
char c[12];
strcpy(c, bar);  // no bounds checking
}
```

foo( ) 函數將函數參數輸入複製到本地堆疊變數 c。如下圖 B 所示：當函數參數輸入小於 12 個字元時，foo( ) 函數會正常工作。如下圖 C 所示：當函數參數輸入大於 11 個字元時，foo( ) 函數會覆蓋本地

圖：堆疊粉碎示例



A 資料複製前　　B "hello" 作為函數參數輸入　　C "AAAAAAAAAAAAAAAAAAAA\x08\x35\xC0\x80" 作為函數參數輸入

堆疊的資料，將函數返回位址覆蓋為 0x80C03508，當 foo( ) 函數返回時，會執行位址 0x80C03508 對應的代碼 A，代碼 A 有可能包含攻擊者提供的 shell 代碼，從而使攻擊者獲得操作許可權。

堆疊保護

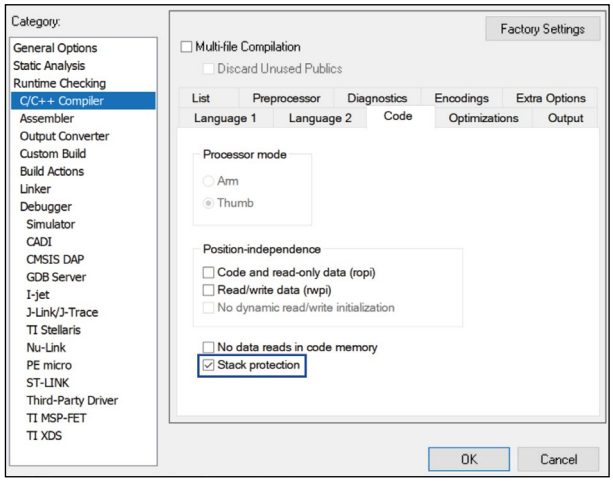
因其功能類似於在煤礦中用來發現瓦斯的金絲雀而得名的堆疊金絲雀 (Stack Canaries)，可以用於在函數返回執行惡意程式碼之前檢測堆疊緩存溢出。其檢測原理是：當調用函數時，將需要保存的臨時資料保存到堆疊，然後放置一個堆疊金絲雀，當函數返回時，檢查堆疊金絲雀的值是否發生改變；如果發生改變，說明堆疊已被篡改，否則說明堆疊沒有被篡改。

下面介紹如何在 IAR Embedded Workbench 這種廣受歡迎的商用工具鏈中實現堆疊保護，從而提高代碼的安全性：

在 IAR Embedded Workbench 中，會使用啓發模式 (Heuristic) 來決定函數是否需要堆疊保護：如果函數區域變數包含陣列類型或者結構體成員包含陣列類型，或者區域變數的位址在該函數外被使用，該函數需要堆疊保護。

IAR Embedded Workbench 安裝目錄下面 \src\lib\runtime 包含 stack_protection.c，裡面包含了 __stack_chk_guard 變數和 __stack_chk_fail 函數，可以作為範本使用：其中 __stack_chk_guard 變數就是堆疊金絲雀的值，在函數返回時，如果檢測到堆疊金絲雀的值被篡改，就會調用 __stack_chk_fail 函數。

1. 將 IAR Embedded Workbench 安裝目錄下面 \src\lib\runtime 資料夾的 stack_protection.c 拷貝並添加到工程。

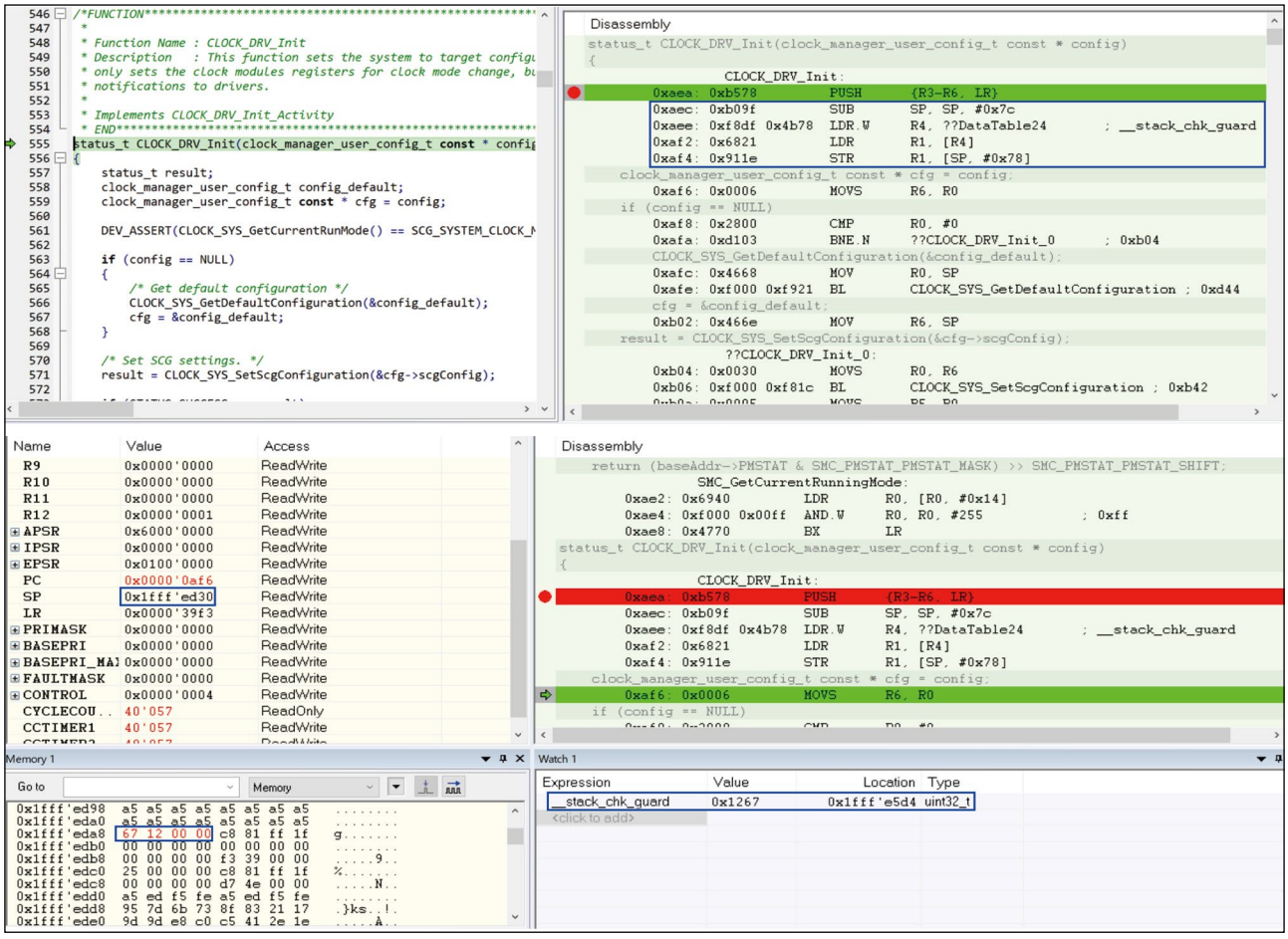


2. 在 IAR Embedded Workbench 中啓用堆疊保護。

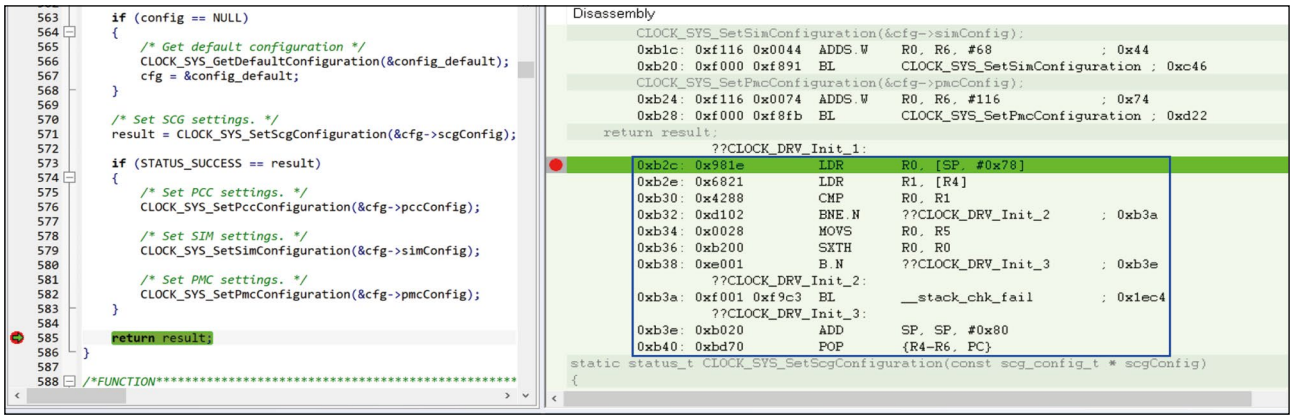
3. 在代碼中聲明堆疊保護相關的 __stack_chk_guard 變數和 __stack_chk_fail 函數。

```
extern uint32_t __stack_chk_guard;
__interwork __nounwind __noreturn void __
stack_chk_fail(void);
```

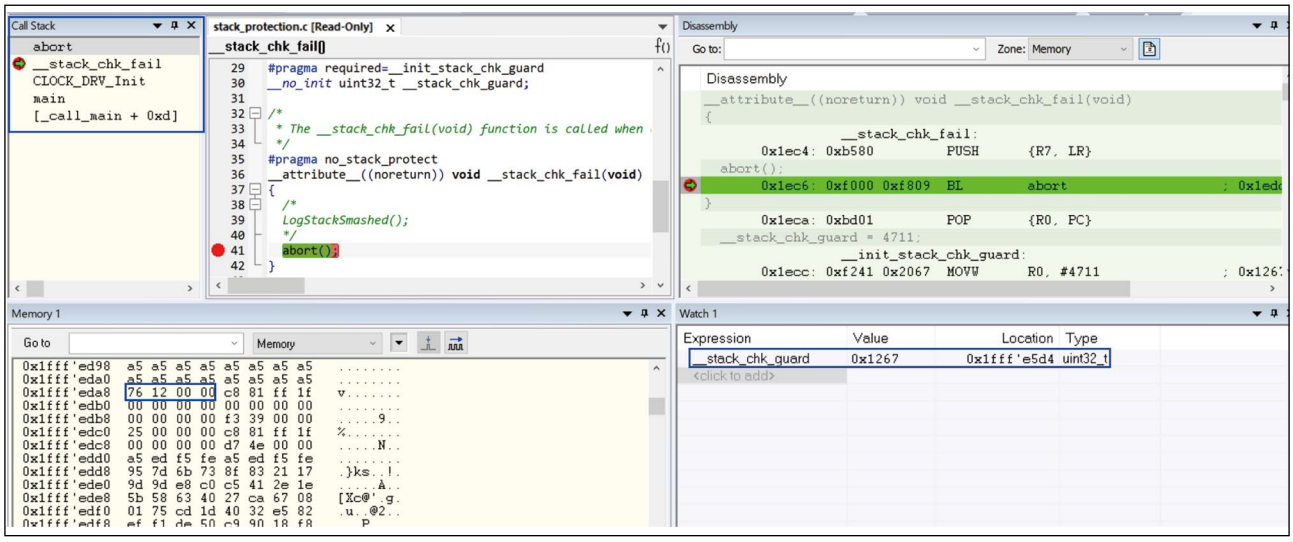
4. 編譯工程。編譯器會在需要堆疊保護的函數中添加如下操作：在函數入口處先入棧 (Push)，然後再額外保存堆疊金絲雀，具體的值用戶可以在 stack_protection.c 中更改 __stack_chk_guard；在函數出口，會檢測堆疊金絲雀的值是否還是 __



在函數出口處打斷點，然後運行程式，在函數返回時，會先檢測堆疊金絲雀的值是否還是 __stack_chk_guard，如果不是，說明堆疊被篡改，會調用 __stack_chk_fail 函數。



改變堆疊金絲雀的值使之與 __stack_chk_guard 不一致，然後運行程式，函數返回時將會調用 __stack_chk_fail 函數：



stack_chk_guard，如果不是，說明堆疊被篡改，會調用 __stack_chk_fail 函數。

調試

將中斷點打到需要堆疊保護的函數反彙編 (Disassembly) 入口，暫停後發現編譯器在函數入口處入棧操作之後額外將堆疊金絲雀保存：

總結

本文主要介紹了堆疊粉碎攻擊如何利用堆疊緩存溢出來影響代碼的安全性。通過在 IAR Embedded Workbench 中實現堆疊保護可以檢測堆疊的完整性，從而提高代碼的安全性。

參考文獻：

1. https://en.wikipedia.org/wiki/Stack_buffer_overflow
2. https://cwe.mitre.org/data/definitions/121.html
3. https://en.wikipedia.org/wiki/Buffer_overflow_protection
4. https://www.iar.com/knowledge/learn/programming/stack-protection-in-iar-embedded-workbench/
5. IAR C/C++ Development Guide(Stack protection)

CTA

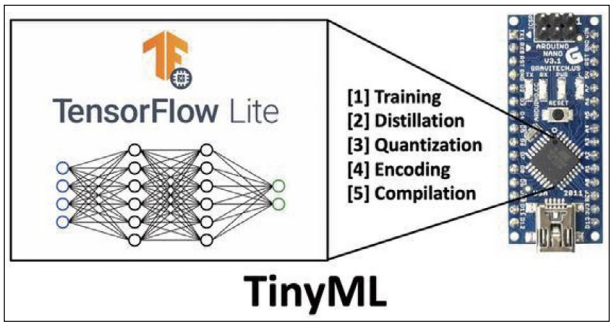
嵌入式人工智慧 / 機器學習 (AI/ML) 以

“生態 + 整合 + 定制”差異化發展

■作者：陳嬌 劉朝暉

隨著嵌入式處理器的能力不斷提升，超小型化的硬體加速器不斷被引入，以及原廠及商業的開發環境和工具不斷出現，嵌入式人工智慧 / 機器學習 (AI/ML) 技術在近幾年得到了快速的發展。同時因為這些技術與千姿百態的各種應用需求十分貼近，因此正在進入差異化發展的新空間，未來其增長速度將可以比肩甚至超過需要強大資源體系的、立足良好通信條件的和基於雲的人工智慧應用。

人工智慧並不是一個近幾年才提出的名詞，但是在近幾年隨著谷歌 AlphaGo 戰勝人類圍棋世界冠軍等事件的推動，才使諸如卷積神經網路、深度學習和機器學習等技術走進了大眾的視野，同時也使“人工智慧 = 資料 + 演算法 + 算力”的模型得到廣泛的認同。



其結果是，在很多人的印象中，人工智慧和機器學習就是在英特爾最新伺服器處理器或者英偉達的 GPU 加速模組的基礎上的全新的、巨大的演算法應用，特別是人工智慧的訓練更是一場資源消耗戰，成為了進入門檻很高的新興領域，傳統上設計 MCU 或者 SoC 的晶片企業基本上與高貴的 AI/ML 無緣。

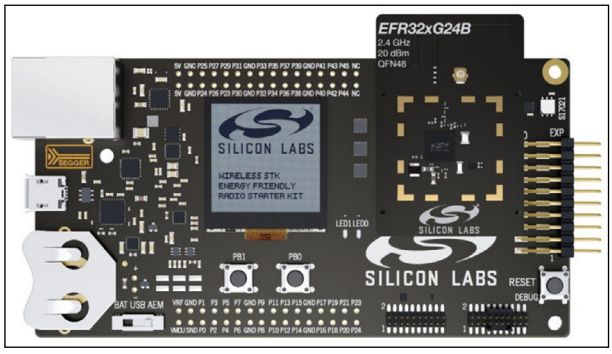
但是人們很快發現在非常領先的人工智慧企業所提供的解決方案中，不僅包括諸如自動駕駛路況分析、自然語言處理、快速醫學影像識別和高頻金融交易等複雜功能，也包括更大量車牌識別、智慧音箱喚醒詞識別、便攜智慧健康監測設備、人臉識別開機和智慧家居安防等 Lite 級別的應用。

在市場強烈需求的拉動下，隨著谷歌的開源 TensorFlow Lite 嵌入式機器學習架構和類似產品的推出，以及諸如 Imagination 公司的 PowerVR 神經網路加速器 (NNA) 等硬體加速器在移動設備或嵌入式設備上商用，各種功耗和成本更低的，以及更加小巧的嵌入式 AI/ML 功能解決方案不斷湧現。

通過分析，我們認為：嵌入式 AI/ML 的廣泛興起，帶來了與傳統 AI 技術以“人工智慧 = 資料 + 演算法 + 算力”為中心的發展範式不同的新模式，針對特定或者一些應用和功能的嵌入式 AI/ML 的重點已轉向“生態 + 整合 + 定制”。下面我們從融入物聯網生態、硬體和商用開發工具整合、以及基於 RISC-V 開發定制處理器這三個方面來進行分析：

爲嵌入式 AI/ML 帶來最新 Matter 協定和物聯網大生態

物聯網晶片、軟體和解決方案供應商 Silicon Labs (亦稱“芯科科技”)，該公司在業界以支持最全面的物聯網通信協定和提供優異的產品性能而著名，其客戶包括智慧家居、智慧城市、工業與商業、



智慧醫療和能源等領域內的領導廠商。

今年初，該公司宣佈推出其 BG24 和 MG24 系列 2.4 GHz 無線 SoC，它們不僅都支持最新的 Matter 物聯網通信協定，還分別支援藍牙和多協定操作，同時還為電池供電的邊緣設備和應用提供了人工智慧 / 機器學習功能，並帶來了高性能無線功能和物聯網大生態。

BG24 和 MG24 無線 SoC 代表業界前沿的生態、功能和技術組合，其中包括支援無線多協定、長電池壽命 (低功耗)、機器學習、以及面向物聯網邊緣應用的安全性。Silicon Labs 為它們提供的全新軟體工具包支援開發人員通過一些常用的工具套件 (如 TensorFlow)，來快速構建及部署 AI/ML 演算法。

為了實現 AI/ML 算力，BG24 和 MG24 系列率先整合了專用的 AI/ML 加速器，幫助開發人員部署人工智慧或機器學習功能並解決功耗難題。這種專用硬體旨在快速高效地處理複雜計算，內部測試顯示其性能提升最高達 4 倍，能效提升最多達 6 倍。由於機器學習計算是在本地設備上而不是在雲端進行，因此消除了網路延遲，加快了決策和行動。

此外，BG24 和 MG24 系列還具有 Silicon Labs 產品組合中最大的快閃記憶體和隨機存取記憶體 (RAM) 容量，使其可支援多協議、Matter 以及用大型資料集訓練 ML 演算法。這些晶片載有獲得了 PSA 3 級認證的 Secure Vault™ 物聯網安全技術，可為門鎖、醫療設備和其他需小心部署的產品提供所需的高安全性。

高集成度嵌入式 AI/ML 配合領先商用開發工具

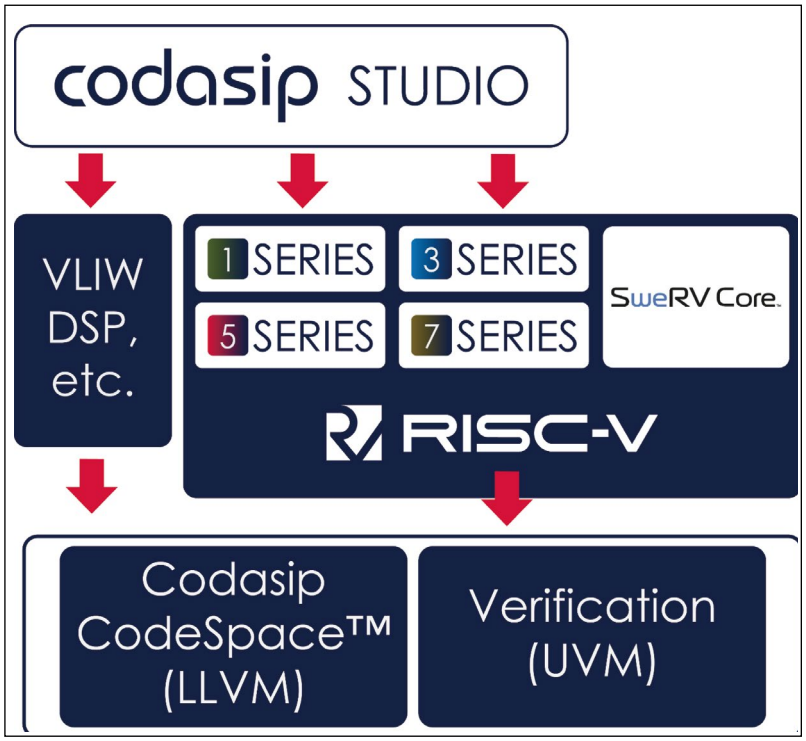
IAR Systems 是嵌入式開發軟體和服務的廠商，其 IAR Embedded Workbench 工具鏈已在全球獲得廣泛採用。IAR Systems 的開發工具為 Alif Semiconductor 高整合度的 Ensemble 和 Crescendo 系列晶片提供支援，打造了基於人工智慧的、高效的微控制器 (MCU) 和融合處理器，賦能下一代嵌入式互聯應用。

對更多功能的整合代表了嵌入式 AI/ML 的一個發展方向，Alif Semiconductor 的這些高能效產品系列提供多達 4 個處理內核，以及人工智慧 / 機器學習 (AI/ML) 加速、多層安全、整合的 LTE Cat-M1 和 NB-IoT 連接、全球導航衛星系統 (GNSS) 定位等功能，從而使其應用範圍得到了大幅擴展。

為了讓這些功能得到更好的發揮，就需要利用諸如 IAR Systems 的 Arm 開發工具這些在行業中已被驗證過的、領先的編譯器技術，對代碼大小和速度都進行優化，另外還提供高性能的調試功能，從而為企業提供了一個很好的平臺。

2021 年 11 月，IAR Systems 宣佈其最新版本的 IAR Embedded Workbench for Arm 增加了對 Arm Cortex-M55 處理器的支援。該處理器是一款支援 AI 技術的 Cortex-M 系列處理器，帶來了節能的數位信號處理 (DSP) 和機器學習功能。

雙方此次合作可以支援 Ensemble 或 Crescendo 元件的應用開發商利用 IAR Embedded Workbench for Arm 開發工具鏈，以實現高性能的



且強大的代碼優化功能，充分發揮器件的 AI/ML 潛能，同時又盡可能地保持能源效率。

RISC-V 使嵌入式 AI/ML 可針對邊緣應用實現定制

多樣化的需求是嵌入式應用的特徵之一，MCU 供應商長期以來是通過不同的處理器內核與外設搭配來滿足用戶的個性化需求。而 RISC-V 的興起，帶來了定制處理器這一新的潮流，這一潮流將繼續延伸到嵌入式 AI/ML 領域，並得到業內領先廠商的支持。

Codasip 就是一家提供領先的 RISC-V 處理器 IP 和高級處理器設計工具的供應商，為 IC 設計者提供 RISC-V 開放 ISA 的所有優勢，以及定制處理器 IP 的獨特能力。Codasip 在今年 2 月推出了兩款專為定制處理器而優化的最新低功耗嵌入式 RISC-V 處理器內核 L31 和 L11。

基於這些新內核，客戶可以很方便地使用 Codasip Studio 工具去定制處理器設計，以支援諸如神經網路、AI/ML 等具有挑戰性的應用，包括例如物聯網邊緣計算等極小型化的、功率受限的應用。Codasip 的內核可定制功能是其成功的基石，目前全球已有超過 20 億顆處理器使用了 Codasip 的 IP。

Codasip L31/L11 嵌入式內核運行在谷歌的 TensorFlow Lite for Microcontrollers(TFLite Micro) 上，並利用 Codasip Studio 工具定制一類全新的嵌入式 AI 內核，可為 AI/ML 計算密集型和內部資源有限的嵌入式系統等應用提供足夠的性能。不同應用對元件的需求也有巨大的差異，而且現有的處理器並不能很好地載入 AI/ML 應用。

Codasip 可提供 “ 創造差異化設計 ” 模式，意味著使用其 Studio 工具的客戶，可以根據其特定系統、軟體及應用程式的要求來定制處理器。通過將 TFLite Micro、RISC-V 定制指令以及 Codasip 處理器設計工具三者相結合，就可以為嵌入式的、高效率的邊緣神經網路處理功能帶來低延遲、高安全性、快速通信和低功耗等優勢。

展望未來：新的應用與新的技術都將不斷湧現

隨著產業的發展，嵌入式 AI/ML 技術和應用都將得到進一步的發展，基於我們提出的 “ 生態 + 整合 + 定制 ” 新範式，以及不斷推陳出新的邊緣應用，我們可以看到在未來一些新的技術值得高度關注，比如新的、適合邊緣應用的硬體加速器和安全技術。

以硬體加速器為例，近年來廣泛興起並得到高度關注的 xPU 將會從雲端走向嵌入式應用；在一些應用場景中，還需要針對演算法和標準的演進和升級對硬體進行再程式設計，那麼諸如 Achronix 公司的 Speedcore 嵌入式 FPGA (eFPGA) 這樣的 IP 產品也會從伺服器和資料中心市場走入嵌入式 AI/ML 應用，推動採用不同硬體加速器的異構計算模式向前發展。 CTA