



Digi-Key和Imagination Technologies RISC-V指南

目錄

1	簡介	3
2	RISC-V背景	4
3	RISC-V基礎知識	5
4	動手實驗：使用Seeed Technologies Maix BiT電路板	7
5	動手實驗：在SparkFun RED-V電路板上使用SiFive SoC	21
6	動手實驗：使用Digilent Nexys A7實作軟體核心	30
7	軟體核心軟體安裝	36
8	建立SweRVolf核心並將其下載到FPGA	45
9	使用RISC-V進行產品開發	49
10	RISC-V的詳細資訊	52
11	結論	52
12	致謝	53
13	作者簡介	53
14	參考資料	54



1 簡介

RISC-V是一種相對較新的電腦技術，正在作為ARM的競爭技術積極推廣。

編寫本指南的目的是為希望瞭解RISC-V軟體和硬體知識的Digi-Key客戶和學員提供簡要介紹。

在2020年9月3日舉行的RISC-V全球論壇上，Imagination Technologies宣布推出面向RISC-V的「RVfpga：瞭解電腦架構的完整課程」全球教學專案，這是其計劃於2020年11月啟動的大學計劃的一部分。<https://university.imgtec.com>上提供了本課程的一些材料。

本文件假設讀者之前對RISC-V瞭解很少或根本不瞭解，而這正是作者在專案開始時所面臨的情況。本文件面向可能正在學習電腦架構的大學生，以及希望在教室或在家中擴充知識的電子/電腦工程師。需要Linux的一些基礎知識。

1.1 什麼是RISC-V？

RISC全名為「Reduced Instruction Set Computer」，即精簡指令集電腦。此處的V代表羅馬數字5。因此，RISC-V是指第5代電腦核心系列。它的發音是「Risk Five」。

RISC-V標誌是RISC-V International的註冊商標。



圖1：RISC-V標誌。圖片來源：www.RISC-V.org

有關規格等方面的詳細資訊，請造訪www.RISC-V.org

1.2 動手實驗

本指南不是純粹的理論指南，而是提供了RISC-V動手實驗練習，練習中採用了Digi-Key公司提供的三種不同電路板。列出的所有價格均為編寫本文時的價格，但可能隨著軟體的變化而改變：

1. 簡單易用、價格合理且充滿樂趣。[Seeed Technologies Co Ltd](#) Maix BiT 電路板可提供雙核RISC-V處理器、攝影頭、LCD螢幕和圖像識別軟體，價格僅為25美元。它採用MicroPython程式語言設計。[（本指南「Maix BiT」節的連結）](#)。
2. 中端晶片系統（System on a Chip，Soc）。用C語言或組合語言在兩塊[SparkFun](#)電路板上進行RISC-V SOC程式設計，價格分別為30美元和36美元。[（本指南「SparkFun」節的連結）](#)。
3. 嚴肅、知識層面要求較高，但價格昂貴。在[Digilent](#) Nexys A7電路板上採用的[Xilinx](#)現場可程式設計閘門陣列（Field Programmable Gate Array，FPGA）中使用Western Digital SweRV軟體核心，價格為265美元。但是，它們均可在軟體模擬器上執行。[（本指南「軟體核心」節的連結）](#)。

好消息是，使用的所有軟體都是免費的。



2 RISC-V背景

2.1 歷史

RISC-V由David Patterson、Krste Asanovic、Andrew Waterman和Yunusup Lee於2010年在加州大學伯克萊分校建立。當時已有Linux開放原始碼軟體，但是沒有等效的開放原始碼硬體。為了促進研究，他們建立了自己的指令集架構（Instruction Set Architecture，ISA）。他們在2011年製造了首個RISC-V晶片，並在2014年推出了首款商業產品。

這些年來，還有其他幾種RISC實作，包括Microchip PIC、ARM、Atmel AVR、MIPS、SuperH和SPARC。最初的RISC-1實作可追溯到1981年。

2.2 開放式架構和開放原始碼

要在諸如處理器或應用特定積體電路（Application Specific Integrated Circuit，ASIC）之類的積體電路中使用ARM®核心，必須先獲得授權。對於最新的核心，這可能需要數百萬美元，並且要花費一些時間。對於諸如NXP或Freescale之類的大型跨國公司，這筆費用只是1億美元總開發費用中的很小一部分。但是，當新的積體電路完成時，每售出一個就必須向ARM支付權利金。

RISC-V則有所不同。既沒有授權費，也沒有權利金。這意味著可以隨時啟動RISC-V實作，並且無需支付任何費用。這對小公司或印度和中國等新興市場來說是個絕佳的消息。RISC-V架構是固定的，可確保未來產品的回溯相容性。

此外，開放原始碼意味著設計可以修改。可以建立特殊指令來提高效能或給駭客的入侵製造麻煩。

[RISC-V開發軟體](#)（尤其是PlatformIO）是免費的。使用[Keil](#)編譯器的授權軟體（例如ARM MDK）的開銷可能很大，而且授權似乎總是在關鍵截止日期的前一天到期。

RISC-V核心通常是在Linux中而不是Windows中實作的，這使得核心和軟體都是開放原始碼的。

2.3 關鍵參與方

Andes Technology是中國台灣的一家半導體製造商。其產品之一是[N22](#)最小RISC-V核心，執行頻率為700 MHz，矽面積僅有0.013 mm²，專為可穿戴設備和物聯網（Internet of Things，IoT）應用設計。

Imagination Technologies是一家智慧財產權（Intellectual Property，IP）公司，專門研究用於手機、平板電腦、電腦和車輛視覺的圖形處理單元（Graphic Processing Unit，GPU）。其最著名的產品是[PowerVR](#)，這款第10代GPU「A系列」便採用了嵌入式RISC-V控制器。

PlatformIO提供了用於RISC-V開發的免費軟體工具。

SiFive由最初的三位RISC-V創造者Krste Asanović、Yunusup Lee和Andrew Waterman創立。它提供RISC-V核心、晶片系統（SoC）、IP和開發板。

Western Digital（SanDisk產品的製造商）致力於在未來產品中使用RISC-V，並且已經生產了自己的[認證RISC-V核心系列](#)（稱為SweRV）。

3 RISC-V基礎知識

儘管說RISC-V是電腦核心，更正確地說法應為，RISC-V是ISA。由使用者自行制定實作方案。

RISC-V規格由[RISC-V.org](https://riscv.org)以兩個文件形式提供：非特權ISA規格（用於基本操作）和特權ISA規格（用於作業系統）。

級別	編碼	名稱	縮寫
0	00	使用者/應用	U
1	01	監督器	S
2	10	保留	
3	11	機器	M

表1：RISC-V特權級別。圖片來源：RISC-V.org

大多數應用使用的都是使用者/應用級別，包括Zephyr作業系統。

3.1 核心命名慣例

RISC-V實作方案可透過16個或32個暫存器，以32位元、64位元或128位元實作，並使用規定的命名慣例。

名稱	功能
RV32I	整數指令集。32位元和32個暫存器
RV32E	嵌入式裝置的整數指令集。32位元和16個暫存器
RV64I	整數指令集。64位元和32個暫存器
RV128I	整數指令集。128位元和32個暫存器

表2：RISC-V基本ISA。圖片來源：RISC-V.org

另外，可選擇實作哪些指令。最常見的MAFD也統稱為G（通用）。

Letter	Functionality
M	Integer multiplication and division
A	Atomic Instructions
F	Single precision floating point
D	Double precision floating point
Q	Quad precision
L	Decimal floating point
C	Compressed instructions (16 bit instructions)
B	Bit manipulation
J	Dynamically translated languages
T	Transactional memory
P	Packed SIMD instructions
V	Vector operations
N	User level interrupts
H	Hypervisor

表3：RISC-V標準ISA擴展。圖片來源：RISC-V.org

因此對於使用最少資源的小型SoC，可能會指定RV32EMAB。這意味著嵌入式裝置的整數指令集、32位元和16個暫存器、整數乘法和除法、原子指令（以避免與中斷發生爭用）、沒有浮點數學運算，但具有位元操作。

另一方面，如果需要64位元RISC-V實作（包括所有四個通用指令MAFD，以及位元操作和使用者級別中斷），則稱為RV64GBN

3.2 可供下載的核心

目前已開發了許多RISC-V核心，大部分是由世界各地的大學完成的。這些核心用多種硬體描述語言（Hardware Description Language，HDL）編寫，例如VHDL、Verilog、System Verilog和不知名的Chisel。對於C程式語言設計師來說，最容易使用的是Verilog，因為它的語法類似於C語言。System Verilog是超集，也可用於驗證。請注意，Verilog是一種硬體描述語言，而不是程式設計語言。

有關可用的RISC-V核心清單，請參閱
<https://github.com/riscv/riscv-cores-list>

圖2提供了一些RISC-V核心的時間線和效能。

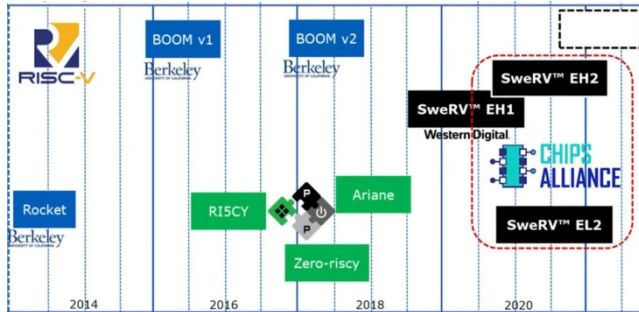


圖2：Boom、Rocket、Riscy和SweRV核心。圖片來源：Western Digital

Boom是用Chisel編寫的Berkeley無序RISC-V處理器。

lowRISC的Rocket核心是具有5級管線的RV64G變型。

RISCY是由蘇黎世聯邦理工學院和博洛尼亞大學開發的一個簡單核心。

SweRV是Western Digital提供的RISC-V處理器系列（共有3款），非常適合工業應用。

此外，Freedom是SiFive使用的核心。

3.3 與ARM和Intel相比的效能

Western Digital提供的圖3顯示了ARM、Intel和三款RISC-V核心 – Boom、Rocket和SweRV的相對效能。CoreMark是嵌入式微處理器基準聯盟（Embedded Microprocessor Benchmark Consortium，EEMBC）針對執行指定範圍常用運算（例如排序算法、循環冗餘校驗（Cyclic Redundancy Check，CRC）和狀態機等）時的實作效能所設計的基準。

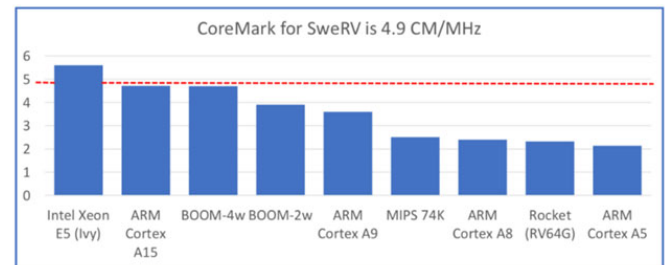


圖3：Western Digital CoreMark。圖片來源：Western Digital

圖中表明，各種RISC-V核心在效能方面可與ARM Cortex A5、ARM Cortex A8、ARM Cortex A9和ARM Cortex A15相媲美。Intel Xeon具有更高的效能。

4 動手實驗：使用Seeed Technologies Maix BiT電路板

在四個動手實驗板中，此電路板的成本最低且使用最簡單。

4.1 Maix BiT電路板說明

該電路板專為物聯網（IoT）和人工智慧（Artificial Intelligence，AI）設計，例如圖像識別。它以較低的價格提供了效能極高的處理器。

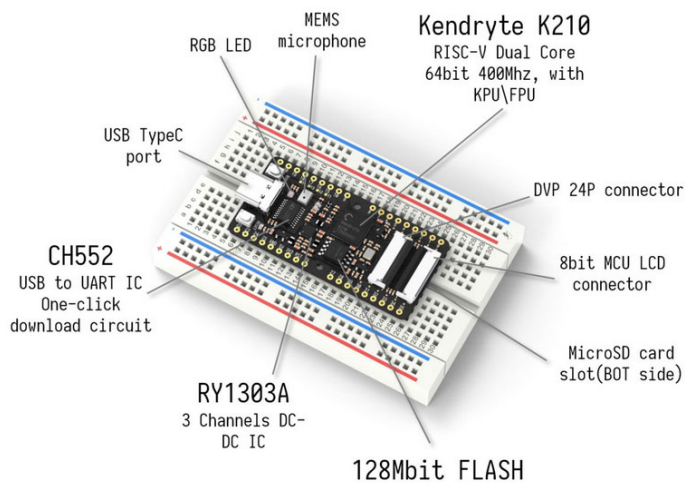


圖4：Maix BiT詳細資訊。圖片來源：Seeed Technologies Co Ltd

4.2 基本Maix BiT電路板

可使用基本Maix BiT電路板，即：

Seeed Technology Co. Ltd Sipeed Maix [BiT RISC-V AI+IoT](#) ,
Digi-Key 1597-1714-ND (14.65美元)

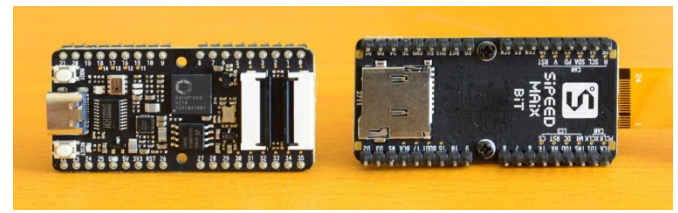


圖5：Seeed Technologies Maix BiT電路板

4.3 帶攝影頭和LCD的Maix BiT電路板

但是，對於動手實驗，建議使用帶攝影頭和LCD螢幕的下列套件：

Seeed Technology K210 [Sipeed Maix BiT Kit RISC-V AI+IoT](#)
Digi-Key 1597-1713-ND (24.21美元)

請注意：Maix BiT電路板未隨附USB-A轉USB-C纜線，因此需要額外購買：

Seeed Technology [USB-C纜線](#) Digi-Key 1597-106990248-ND (2.42美元)

4.3.1 隨附的Maix BiT元件

盒中包含一個Maix BiT電路板、一個攝影頭和一個LCD顯示器，還提供了一把螺絲起子和3顆螺絲 + 尼龍墊片，但實際上只需要2個螺絲 + 墊片。

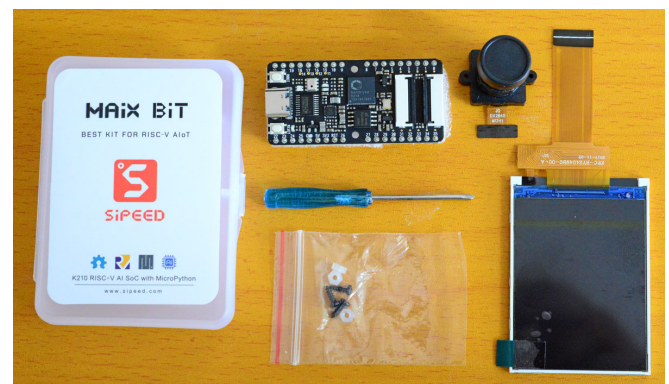


圖6：帶攝影頭、LCD和元件的MaixPy BiT盒和電路板。

4.3.2 組裝MaixPy BiT電路板

組裝時間：約5分鐘。這些元件很小，因此有點複雜。

需要的其他工具：用於在擰緊螺絲時固定尼龍墊片的鉗子。

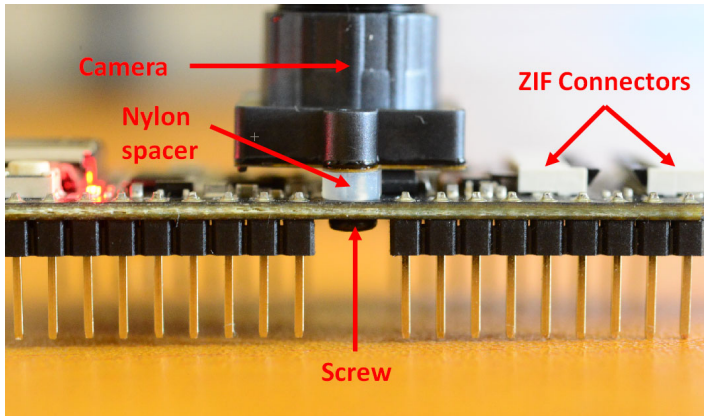


圖7：攝影頭安裝

4.3.3 Maix BiT組裝說明

1. 安裝兩個螺絲和尼龍墊片。
2. 向上推動黑色鎖定桿，將攝影頭纜線插入電路板中間附近的白色ZIF連接器中。
3. 向下推動黑色鎖定桿，將攝影頭纜線鎖定到位。
4. 需要彎曲攝影頭纜線以使攝影頭對準螺絲上方。
5. 用螺絲起子擰緊兩個螺絲，將攝影頭固定到位。
6. 將LCD纜線插入電路板邊緣附近的ZIF連接器中。
7. 向下推動黑色鎖定桿，將LCD鎖定到位。

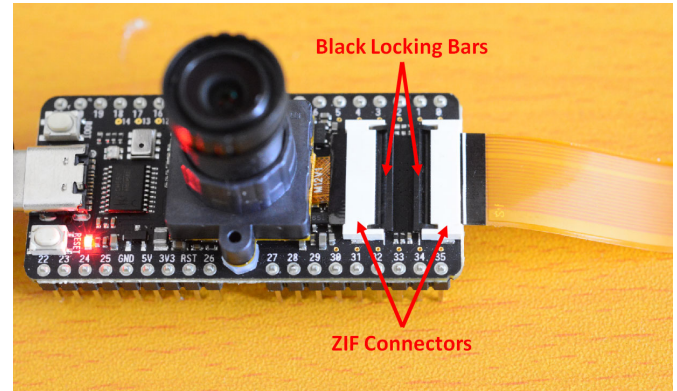


圖8：裝入ZIF插座的攝影頭和LCD

4.4 Maix BiT電路板軟體

該電路板已預先使用名為MaixPy的MicroPython解譯器進行程式設計。使用MaixPy IDE軟體或只需在序列終端機中輸入命令即可編寫程式。

Windows和Linux均可下載該軟體，但Linux需要一些額外的步驟。（[本指南的「Linux安裝」節連結](#)）。

4.5 在Windows上安裝MaixPy IDE

這是一個用於編寫程式碼和執行Maix BiT電路板的整合開發環境（Integrated Development Environment，IDE）。

安裝時間：約10分鐘。

MaixPy IDE軟體有點難以找到，因為它的位置看起來是空白的。

移至：<https://maixpy.sipeed.com/en>

此連結將重新導向到包含有用資訊的MaixPy文件頁面。

4.5.1 MaixPy文件網頁

按一下左側的「*Install MaixPyIDE(optional)*」（安裝MaixPyIDE（可選））。

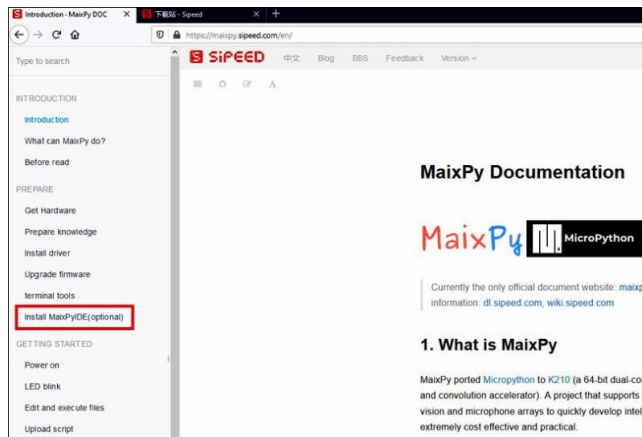


圖9：MaixPy文件頁面

4.5.2 移至MaixPy下載頁面

按一下 dl.sipeed.com



圖10：下載安裝套件

4.5.3 選擇目錄

按一下_

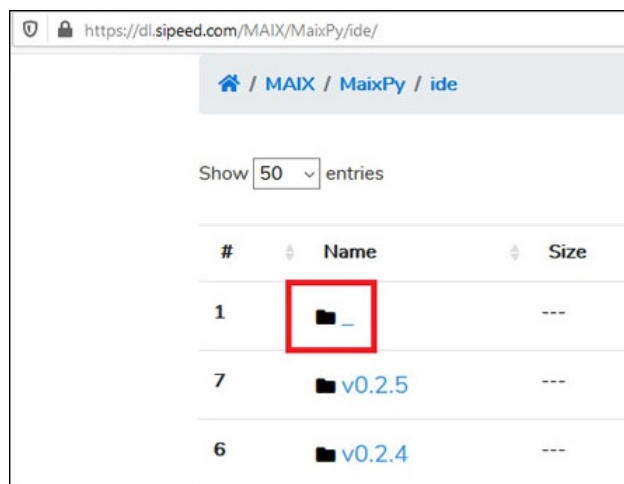


圖11：標有_的MaixPy IDE資料夾

4.5.4 選擇最新版本

按一下最新版本，此處為v0.2.5



圖12：選擇v0.2.5

4.5.5 選擇適用於Windows的MaixPY IDE

按一下maixpy-ide-windows-0.2.5.exe。檔案大小為85.5MB。

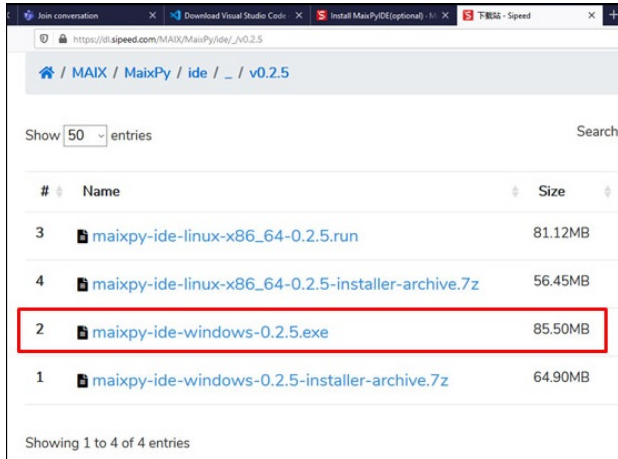


圖13：按一下適用於Windows的MaixPy IDE

4.5.6 下載MaixPy IDE (Windows)

按一下「Save File」（儲存檔案），然後按兩下maixpy-ide-windows-0.2.5.exe進行安裝。

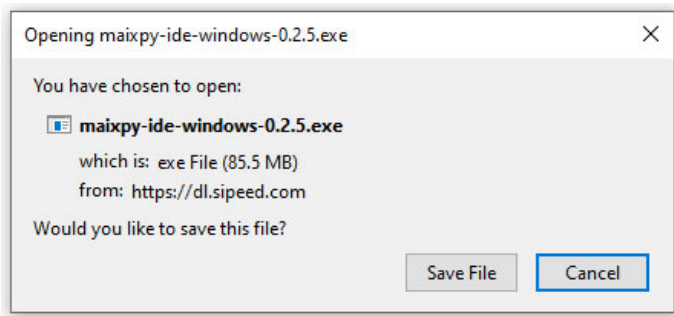


圖14：下載maixpy-ide-windows

程式將安裝到：

C:\Program Files
(x86)\MaixPyIDE\bin\maixpyide.exe

4.5.7 MaixPy (Windows) 的捷徑

預設情況下，MaixPy不會在桌面上放置捷徑圖示。要建立MaixPy的捷徑，請用滑鼠右鍵按一下maixpyide.exe，然後選擇「Create shortcut」（建立捷徑），捷徑隨後將儲存在桌面上。

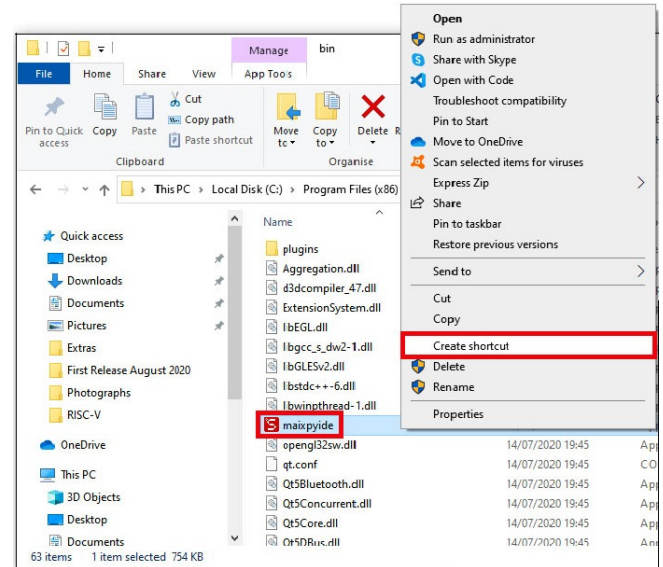


圖15：MaixPy IDE的捷徑

(在MaixPy IDE中執行程式的連結)

4.6 使用Linux安裝MaixPy IDE

安裝時間：約10分鐘。下載的檔案大小為287 MB。

還有一個命令行版本可供使用，需要40分鐘來下載和安裝。

MaixPy IDE軟體有點難以找到，因為它的位置看起來是空白的。

移至<https://maixpy.sipeed.com/en>

此連結將移至包含有用資訊的MaixPy文件頁面。

4.6.1 MaixPy文件網頁

按一下「*Install MaixPyIDE (optional)*」(安裝MaixPyIDE (可選))

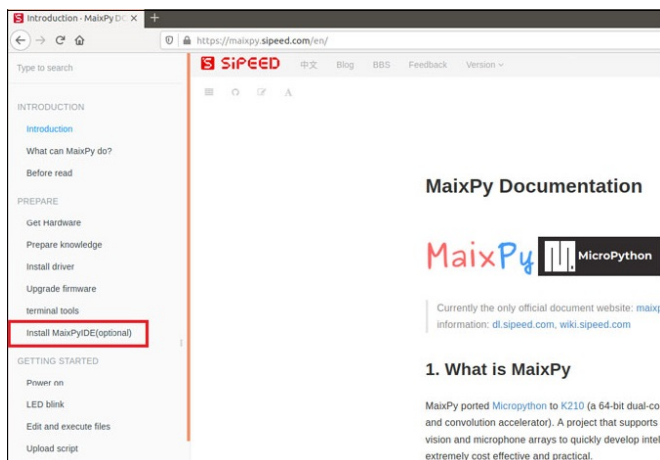


圖16：MaixPy文件頁面

4.6.2 Seeed下載頁面

按一下`dl.seeed.com`，移至下載頁面。



圖17：適用於Linux的MaixPy下載安裝套件

4.6.3 選擇資料夾

按一下標有_的資料夾

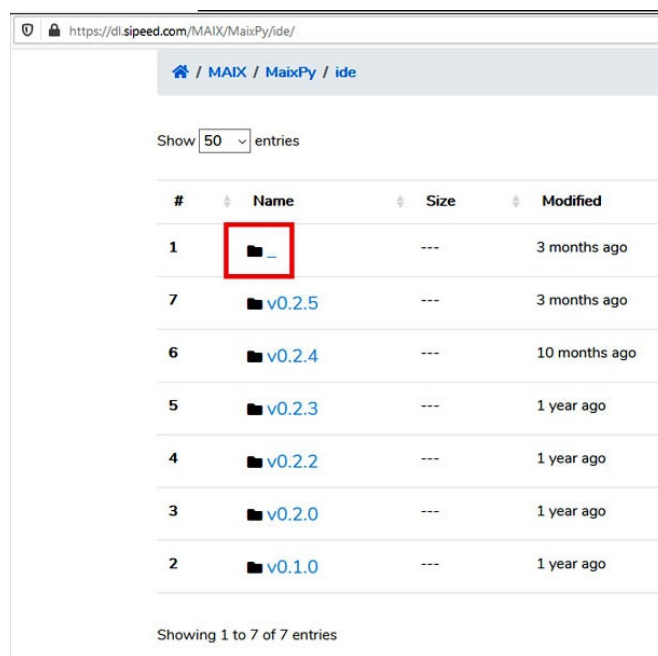


圖18：選擇標有_的資料夾

4.6.4 選擇最新版本

按一下v0.2.5

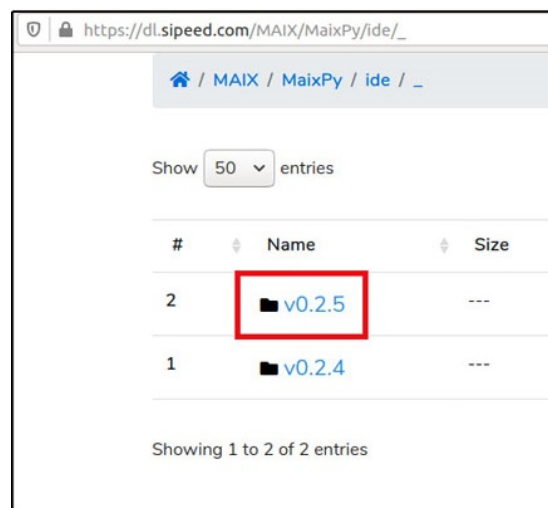


圖19：選擇v0.2.5

4.6.5 選擇檔案

按一下：

maixpy-ide-linux-x86_64-0.2.5.run

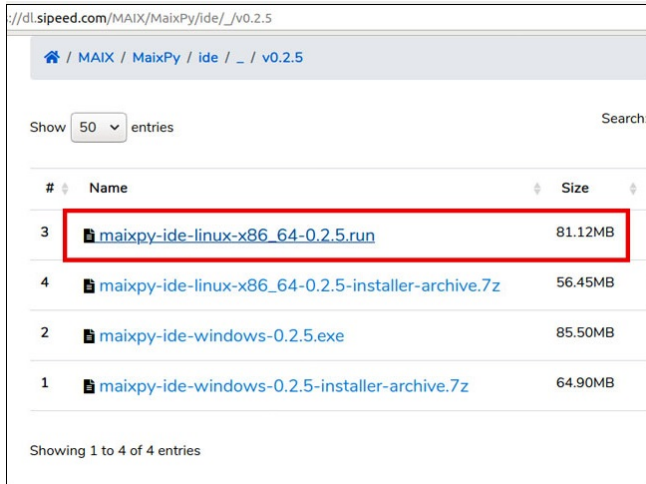


圖20：選擇Linux執行檔

4.6.6 儲存Linux安裝檔

按一下「OK」（確定）以儲存檔案。

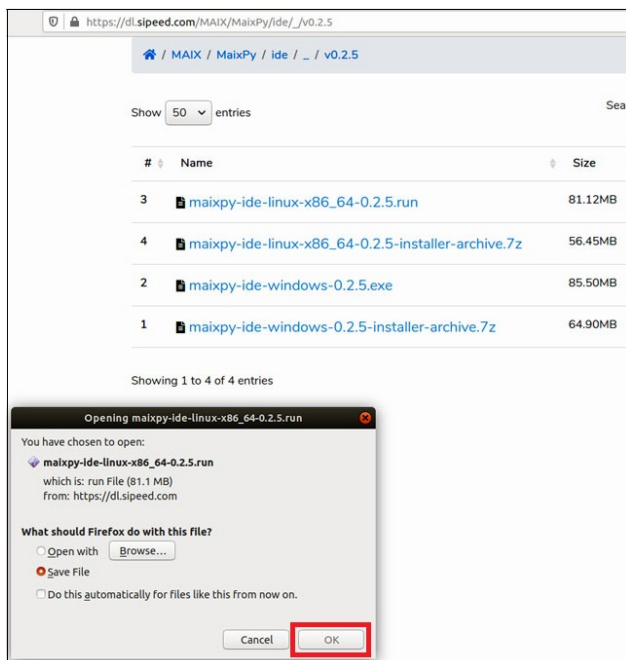


圖21：儲存Linux安裝檔

4.6.7 USB連接埠存取

預設情況下，USB連接埠通常不可用，這會阻止MaixPy程式執行。



開啟Ubuntu終端機並輸入：

```
sudo adduser myname dialout
```

其中myname用電腦使用者名稱替換，與/home/myname/中相同

重新啟動電腦，讓變更生效。

如果不執行該步驟，稍後將出現提示。

4.6.8 在Linux中執行下載的檔案



開啟Ubuntu終端機。

為MaixPy提供的指令適用於0.2.2版，而非針對0.2.5版所編寫，因此已過時。

```
chmod +x maixpy-ide-linux-x86_64-0.2.2.run
./maixpy-ide-linux-x86_64-0.2.2.run
```

圖22：MaixPy指令

改為輸入後來的檔案名稱：

```
chmod +x maixpy-ide-linux-x86_64-0.2.5.run
./maixpy-ide-linux-x86_64-0.2.5.run
```

隨即應出現MaixPy IDE安裝精靈。

4.6.9 MaixPy IDE安裝精靈



圖23：MaixPy安裝精靈

按一下「Next>」（下一步>），然後使用預設值以執行精靈的步驟。完成後，應顯示開始畫面。

4.6.10 MaixPy開始畫面

如果開始畫面未顯示來自攝影頭的圖片（畫面格緩衝區）和紅綠藍（RGB）訊號，請按一下畫面右側的



圖示。

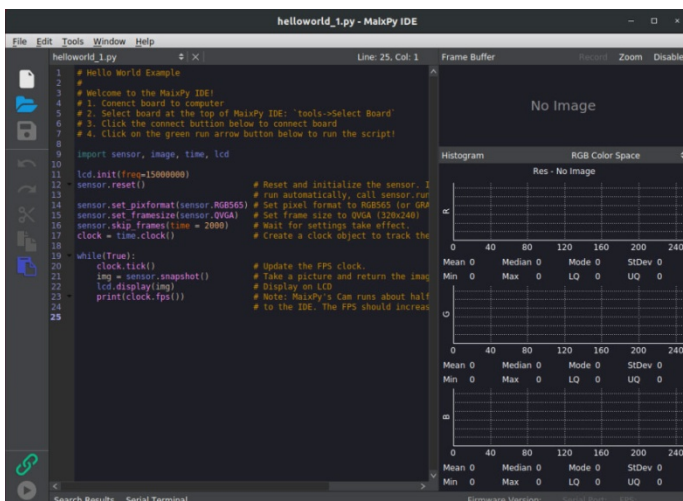


圖24：MaixPy Linux IDE畫面

4.7 在MaixPy IDE中執行程式

Linux和Windows的程序類似。

4.7.1 插入電路板

將USB纜線插入電腦，LCD上將顯示歡迎訊息。

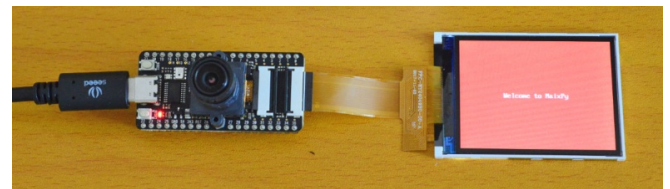


圖25：Maix Bit攝影頭和LCD操作

4.7.2 啟動MaixPy

如果MaixPy IDE尚未執行，則移至Ubuntu搜尋工具並輸入MaixPy。按一下MaixPy IDE圖示。

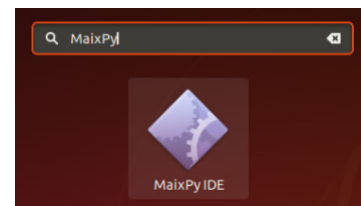


圖26：MaixPy IDE Linux圖示

4.7.3 選擇程式

首次通電時預設會載入helloworld_1.py程式。

4.7.4 選擇電路板

在工具列上，依序選擇「Tools」（工具）、「Select Board」（選擇電路板）和Sipeed Maix Bit (with Mic)（Sipeed Maix Bit（帶麥克風））。

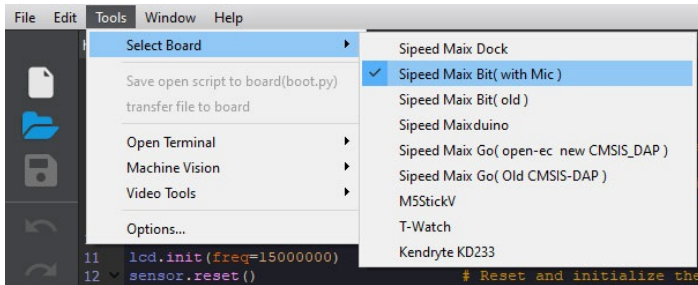


圖27：MaixPy IDE選擇電路板

4.7.5 選擇序列埠

按一下畫面左下方的綠色圖示以顯示「Connect – MaixPy IDE」（連接 – MaixPy IDE）功能表。將有多個序列埠可供選擇。

選擇正確的序列埠後，它會在幾秒鐘內連接，且左下方的綠色圖示將變為紅色。

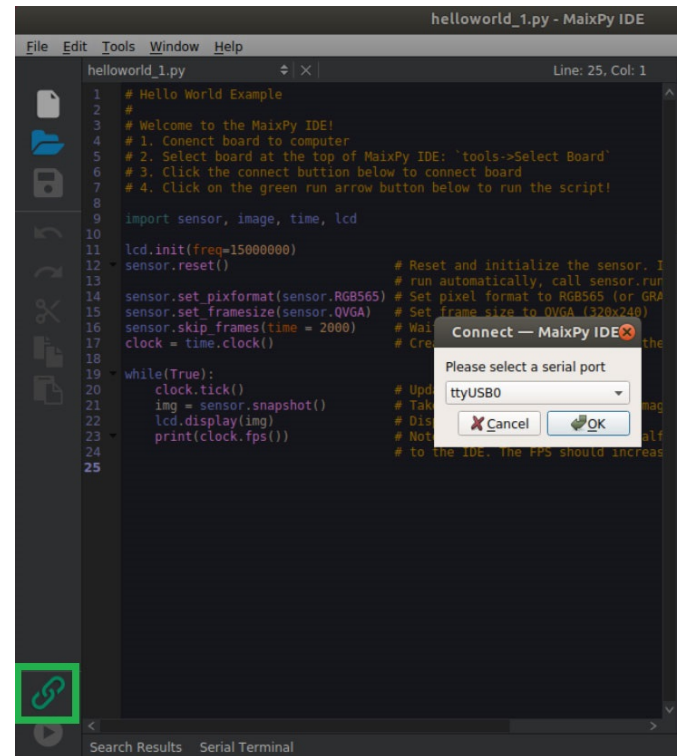


圖28：MaixPy序列連接

4.7.6 首次執行時的Linux快顯視窗

如果MaixPy IDE尚無法使用Linux USB連接埠，則會彈出類似於下面的提醒，但實際的電腦使用者名稱不是「richard」。請按照說明操作。

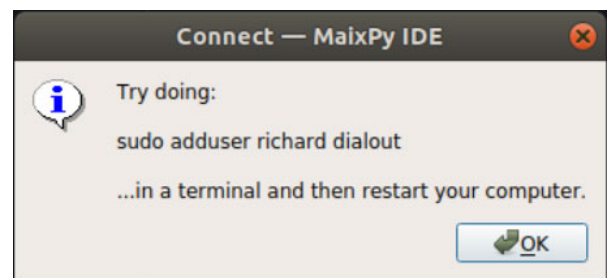



圖29：對話框組

4.7.7 執行程式

按一下畫面左下方的綠色箭頭，執行程式。

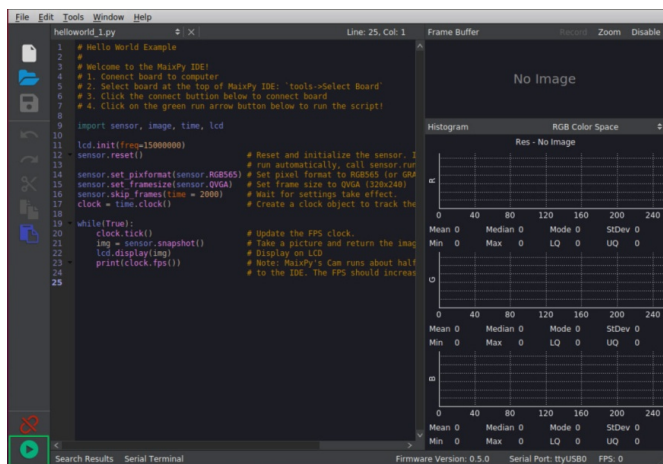


圖30：設定程式以準備好執行

4.7.8 Hello World程式執行

helloworld_1.py程式現在正在執行。電腦會顯示攝影頭的圖片以及RGB光譜。LCD螢幕也會顯示該圖片。

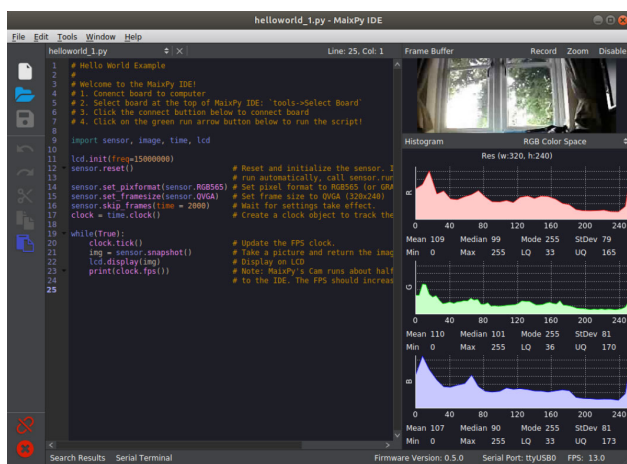


圖31：在Linux中執行Hello World應用程式

要停止程式，請按一下「Stop」（停止）圖示



4.8 其他MaixPy專案

此外還提供了其他專案，但是需要下載。在MaixPy IDE中，依序按一下「File」（檔案）、「More examples」（更多範例）和「Examples on github repo」（github存放庫上的範例），移至Github。這適用於Linux和Windows版本。

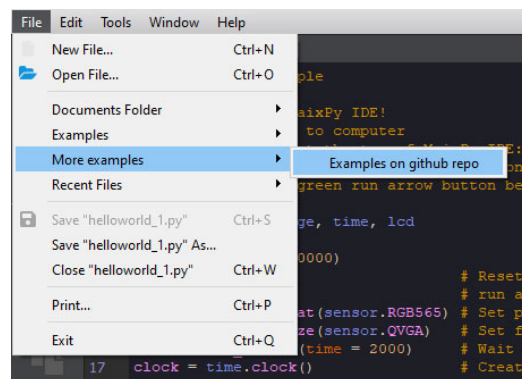


圖32：更多MaixPy範例

4.8.1 下載MaixPy範例

下載MaixPy_scripts-master.zip並儲存，然後解壓縮檔案。

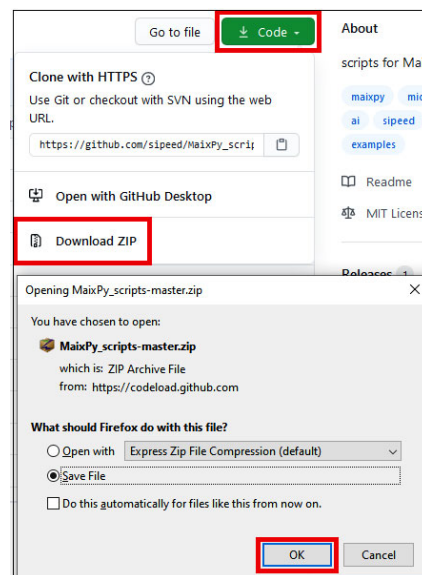


圖33：MaixPy範例

4.9 人臉識別專案

所需時間：約15分鐘。這涉及到下載和安裝另外兩個程式。

MaixPy的其他範例在machine_vision目錄中提供了人臉識別程式demo_find_face.py。將其載入到MaixPy IDE中。

4.9.1 其他必要檔案

該專案需要在執行之前將名為face_model_at_0x300000.kfpkg的附加模型載入到MaixPy BIT電路板上。為此，需要kflash_gui工具。

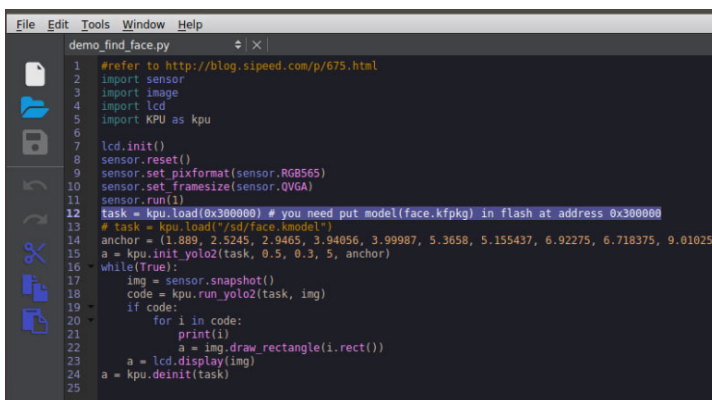


圖34：人臉識別程式附加檔案

請注意：在早期版本的kflash_gui上，預設下載位置為0x000000。這意味著可能將face_model_at_0x300000.kfpkg下載到位址0x000000，而不是正確的位址0x300000。這會改寫MicroPython解譯器，而電路板根本不會執行。因此需要重新設計電路板的程式。

4.10 更新Maix BiT快閃記憶體

所需時間：約10分鐘。

kflash_gui工具可在Windows和Linux上執行。使用最新軟體可以迅速而輕鬆地燒寫電路板。

4.10.1 下載kflash_gui

可從以下位置下載kflash_gui：

https://github.com/sipeed/kflash_gui/releases.

此處使用的是kflash_gui_v1.5.3_windows.7z。更高版本會導致Avast Virus檢查程式出現問題。

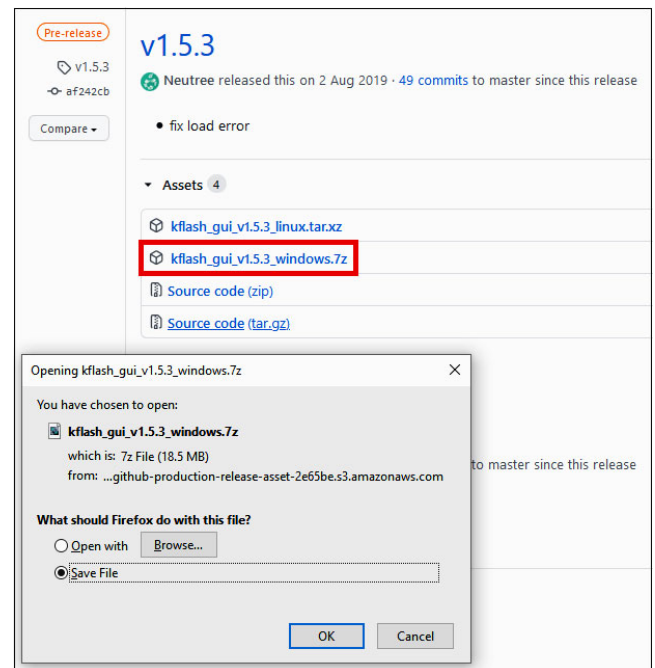



圖35：下載kflash GUI

解壓縮.7z檔案。如果電腦不支援.7z檔案，可從以下網址從線上獲取免費工具，例如Express Zip：

<https://www.nchsoftware.com/software/utilities.html>.

要執行程式，請移至解壓縮的檔案，然後按兩下

 **kflash_gui** 圖示。

請注意：kflash_gui安裝不會在桌面上放置捷徑。如果需要捷徑，則需要手動建立。

4.10.2 下載人臉模型

從<https://github.com/sipeed/MaixPy/releases>下載 face_model_at_0x300000.kfpkg 並儲存檔案。

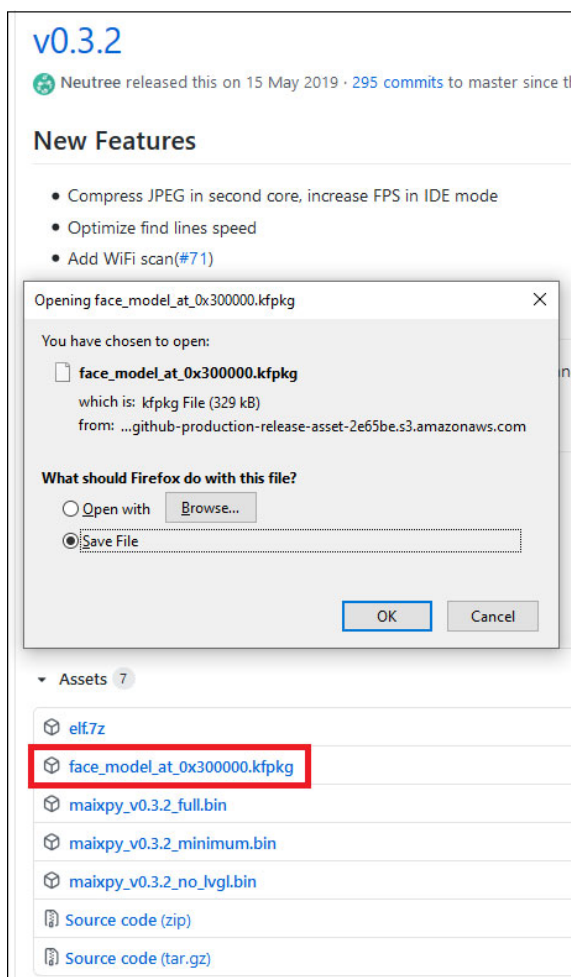


圖36：下載人臉模型

4.10.3 使用kflash GUI設計人臉模型程式

按一下  **kflash_gui** 圖示。按一下「Open File」（開啟檔案），然後選擇 face_model_at_0x300000.kfpkg。依序選擇 *Sipeed Maix Bit (with Mic)*（*Sipeed Maix Bit*（帶麥克風）電路板和序列埠，然後按一下「Download」（下載）。

下載需要大約20秒。

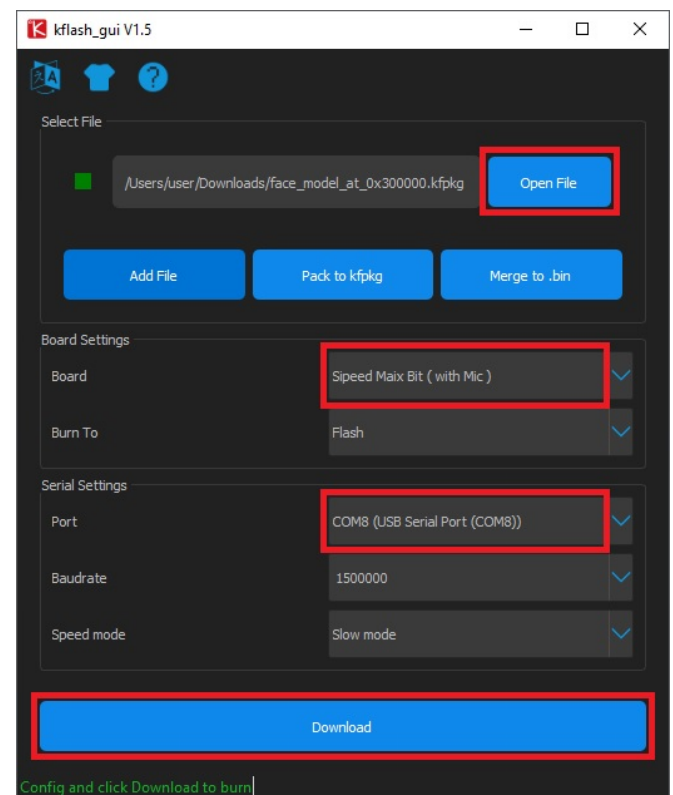


圖37：kflash GUI

4.11 執行人臉識別程式

返回MaixPy IDE，然後執行machine_vision目錄中的demo_find_face.py。

識別到人臉後，將在LCD和電腦螢幕上繪製一個矩形。

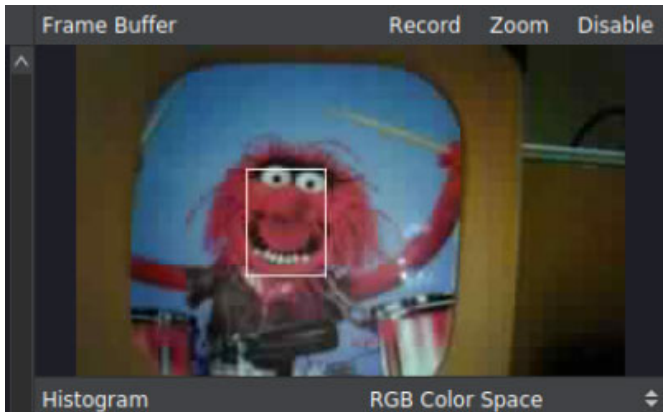


圖38：人臉識別範例

4.12 重新燒錄Maix BiT電路板主軟體

kflash_gui還可用於在Maix BiT電路板上重新設計MicroPython解譯器的程式。韌體更新十分頻繁，但並非全部都與MaixPy IDE相容。

4.12.1 下載最新軟體

移至：

<https://dl.sipeed.com/MAIX/MaixPy/release/master>。

其中包含MaixPy版本的清單。按一下最新版本。

#	Name	Size	Modified
72	maixpy_v0.5.0_120_g384ea9a	---	3 days ago
71	maixpy_v0.5.0_110_g7caf245	---	1 week ago
70	maixpy_v0.5.0_106_g67c538f	---	2 weeks ago
69	maixpy_v0.5.0_104_gbbd4c98	---	3 weeks ago

圖39：MaixPy版本主索引

4.12.2 選擇最新MaixPy

按一下以_minimum_with_ide_support.bin結尾的檔案並儲存。

檔案具有IDE支援至關重要。

#	Name	Size	Modified
5	maixpy_v0.5.0_120_g384ea9a.bin	1.97MB	3 days ago
7	readme.txt	1.71KB	3 days ago
6	maixpy_v0.5.0_120_g384ea9a_minimum_with_ide_support.bin	732.50KB	3 days ago
1	maixpy_v0.5.0_120_g384ea9a_with_lvgl.bin	2.20MB	3 days ago

圖40：Maix BiT主軟體下載

4.12.3 設計Maix BiT電路板程式

開啟kflash_gui。使用「Open File」（開啟檔案）進行選擇：

```
maixpy_v0.5.0_120_g3943a9a_minimum_with_ide_support.bin
```

依序選擇電路板和序列埠，然後按一下「Download」（下載）。

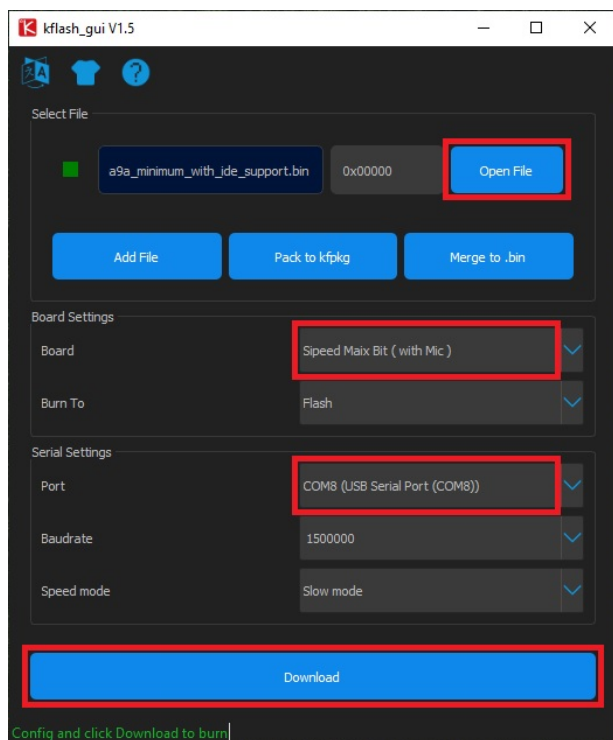


圖41：重新設計Maix BiT快閃記憶體程式

4.13 在序列終端機上執行MaixPy

所需時間：約5分鐘。

除了使用MaixPy IDE外，也可以在序列終端機上執行程式。這種方法快速輕鬆，對於檢查Maix BiT電路板配置和軟體版本十分有用。它也可以用作學習MicroPython的工具。

4.13.1 設定序列終端機

如果尚未安裝，請下載並安裝任何免費的序列終端機程式，例如Putty或TeraTerm。

將序列埠速度設定為115200傳輸速率。在另一台電腦上，連接埠COM8可能有所不同。

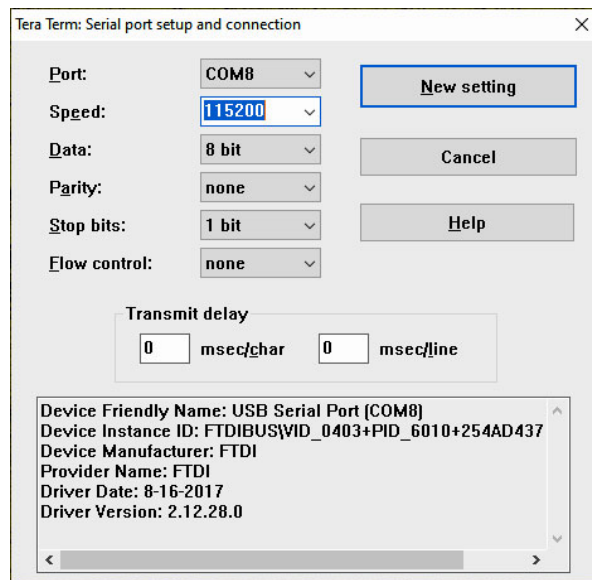


圖42：序列終端機配置

4.13.2 按下並釋放重設按鈕

按下並釋放MaixPy BiT電路板上的重設按鈕。

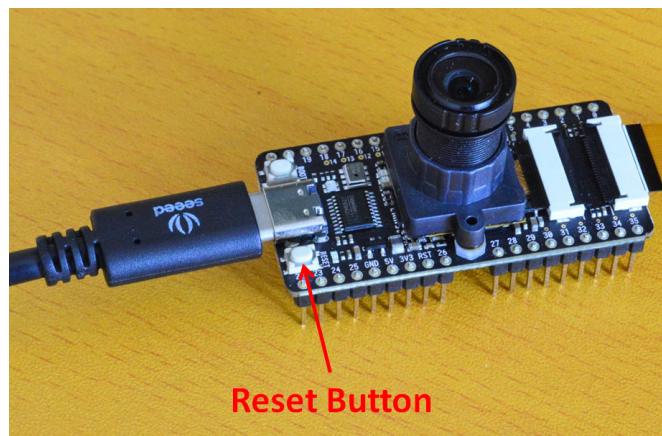
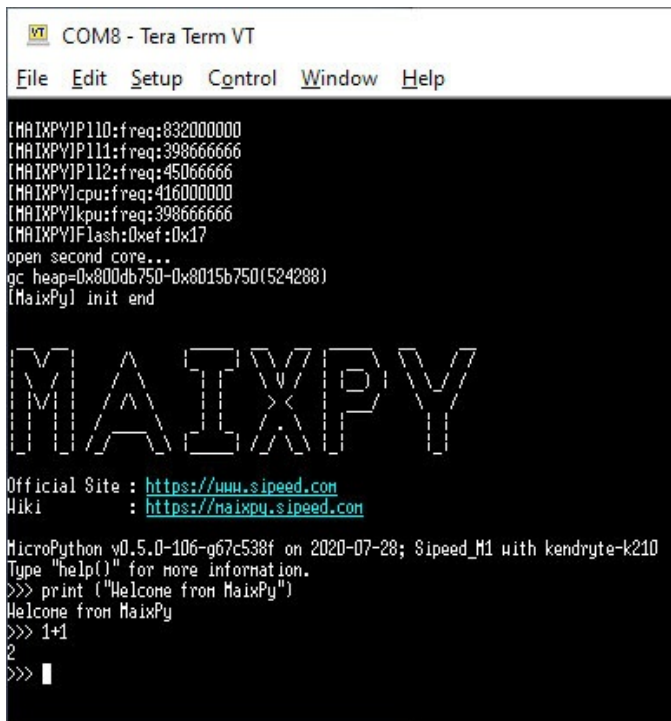


圖43：Maix BiT電路板重設按鈕

4.13.3 在序列終端機上執行MaixPy



```
COM8 - Tera Term VT
File Edit Setup Control Window Help

[MAIXPY]P110:freq:832000000
[MAIXPY]P111:freq:398666666
[MAIXPY]P112:freq:450666666
[MAIXPY]cpu:freq:416000000
[MAIXPY]kpu:freq:398666666
[MAIXPY]Flash:0xef:0x17
open second core...
gc heap=0x800db750-0x8015b750(524288)
[MaixPy] init end

MAIXPY

Official Site : https://www.sipeed.com
Wiki : https://maixpy.sipeed.com

MicroPython v0.5.0-106-g67c538f on 2020-07-28; Sipeed_M1 with kendryte-k210
Type "help()" for more information.
>>> print ("Welcome from MaixPy")
Welcome from MaixPy
>>> 1+1
2
>>> █
```

圖44：序列終端機上的MaixPy

輸入`print ("Welcome from MaixPy")`，將回應「Welcome from MaixPy」。

輸入`1+1`，回應為2。

5 動手實驗：在SparkFun RED-V電路板上使用SiFive SoC

這是中檔選項，其中兩塊電路板設計在FreedomStudio IDE中使用C語言進行程式設計。

5.1 SparkFun影片

Digi-Key網站上提供了[Shawn Hymel](#)製作的一個精彩短片，展示了如何使用這些電路板。在購買電路板、安裝軟體之前，或稍後等待軟體下載時，這部短片非常值得一看。

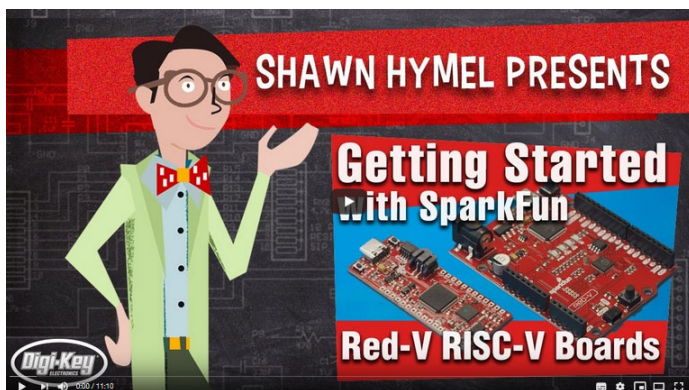


圖45：RED-V入門。來源：Digi-Key網站。

Digi-Key網站的Red Board和RED-V Thing Plus頁面上也提供了影片的連結。

5.2 硬體

基於SiFive Freedom Everywhere RISC-V SoC的SparkFun Electronics提供了兩塊電路板。兩塊電路板均與SiFive1 Rev B電路板和軟體相容：

小型SparkFun [RED-V SIFIVE RISC-V THING PLUS](#) - 1568-DEV-15799-ND (29.95美元)

稍大的SparkFun Electronics FE310 [Red Board](#) - 1568-DEV-15594-ND 39.95 (39.95美元)

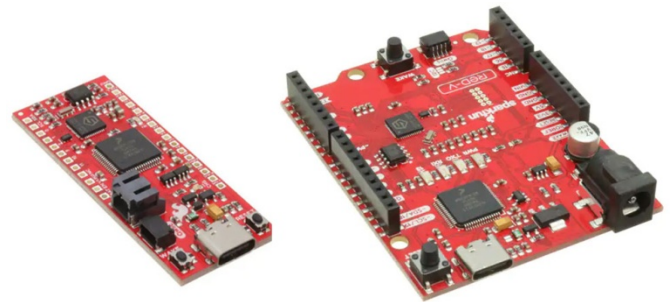


圖46：SparkFun RED-V Thing Plus（左側）和Red Board（右側）。
圖片來源：SparkFun Electronics

備註：Thing Plus和Redboard都沒有USB-C轉USB-A纜線，因此需要自行購買，例如：

Seeed Technology USB-C纜線Digi-Key 1597-106990248-ND (2.42美元)

SiFive提供適用於Windows和Linux的軟體。遺憾的是，作者無法讓Linux版本執行，因此這裡只考慮Windows版本。

5.3 SiFive Freedom Studio軟體安裝 (Windows)

安裝需要大約20分鐘，下載檔的大小為1.3 GB。

移至www.SiFive.com/software，將頁面向下捲動到Freedom Studio部分。

按一下Windows，然後儲存.zip檔案。

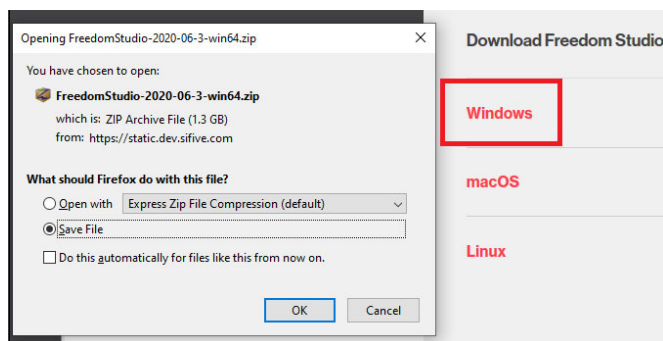



圖47：SiFive Freedom Studio (Windows)

5.3.1 Freedom Studio檔案位置

解壓縮檔案。導覽到安裝目錄，例如：

C:\Users\user\FreedomStudio-2020-06-3-win64\

按兩下  FreedomStudio 以執行它。

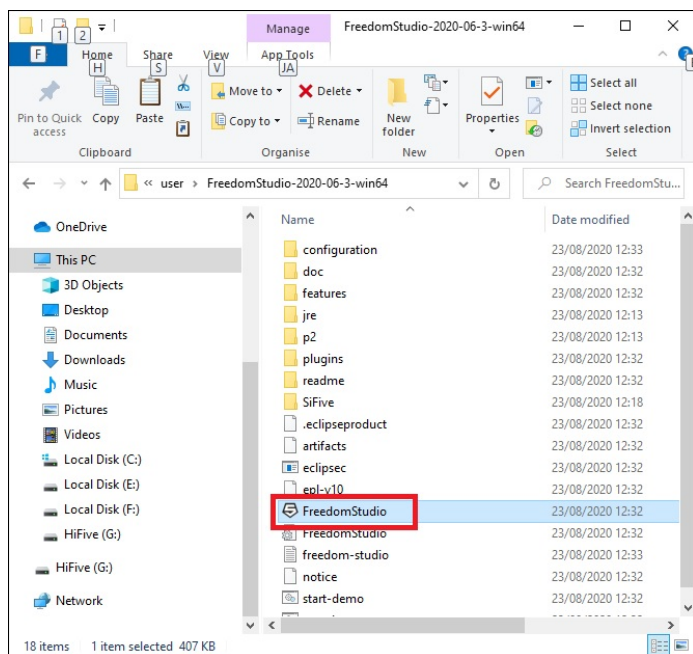


圖48：FreedomStudio檔案位置

5.3.2 Freedom Studio啟動畫面

應出現Freedom Studio圖示。如果需要，可將捷徑新增至桌面。



圖49：Freedom Studio啟動畫面

5.4 建立新的軟體專案

如果尚未執行，請啟動Freedom Studio。

5.4.1 插入電路板

建議在建立程式之前插入電路板，因為它會自動設定偵錯配置。使用USB-C轉USB-A纜線。可使用任一SparkFun電路板。

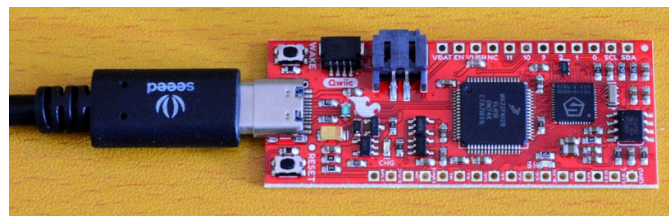


圖50：用USB-C纜線連接Thing Plus Board

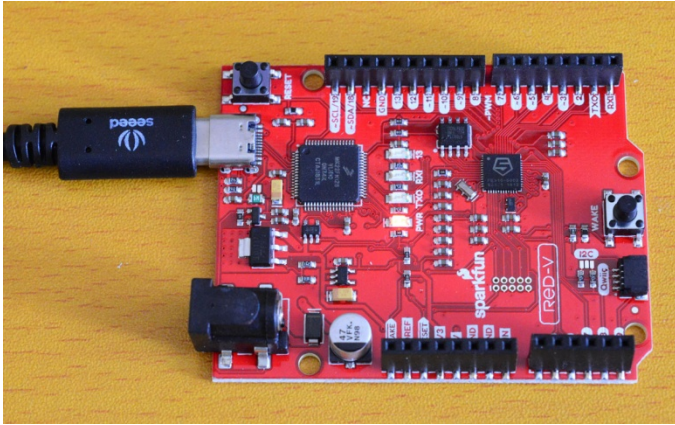


圖51：用USB-C纜線連接Red Board

5.4.2 建立新的Freedom E SDK軟體專案

移至畫面頂部的「SiFive Tools」（SiFive工具）索引標籤，按一下「Create a new Freedom E SDK Software Project」（建立一個新的Freedom E SDK軟體專案）。

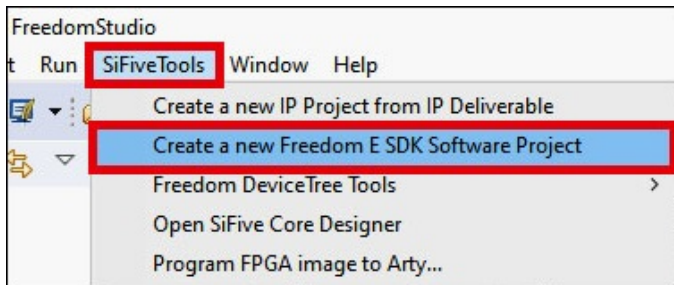


圖52：建立新的Freedom E SDK軟體專案

5.4.3 電路板選擇

支援多種電路板。從下拉式功能表中選擇 *siFive-hifive1-revb*。它同時相容Red Board和Thing Plus Board。

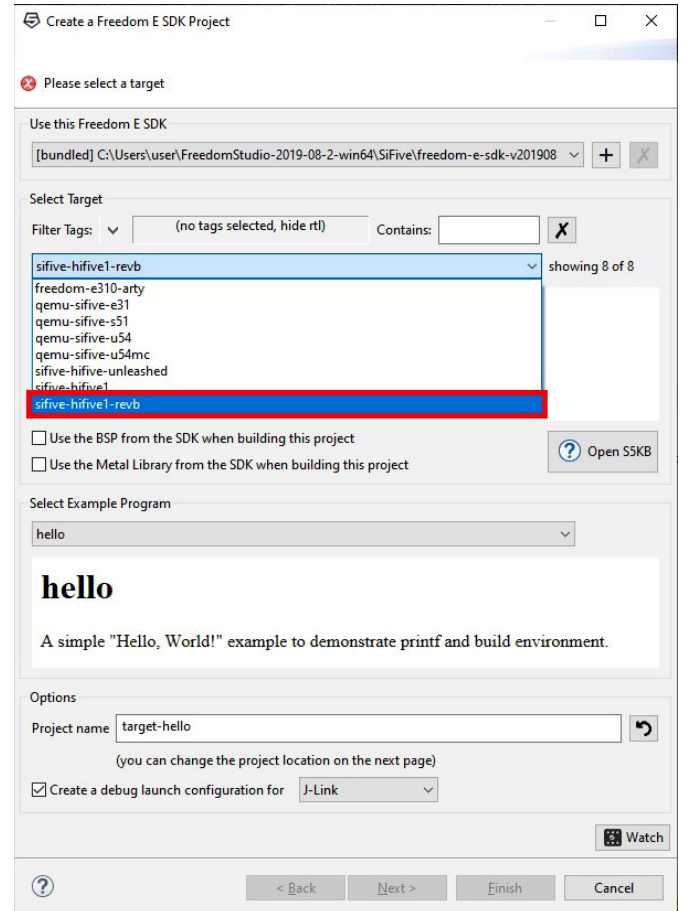


圖53：SiFive電路板選擇

5.4.4 範例程式選擇

從「*Select Example Program*」（選擇範例程式）下拉式功能表中，可選擇一系列範例專案。

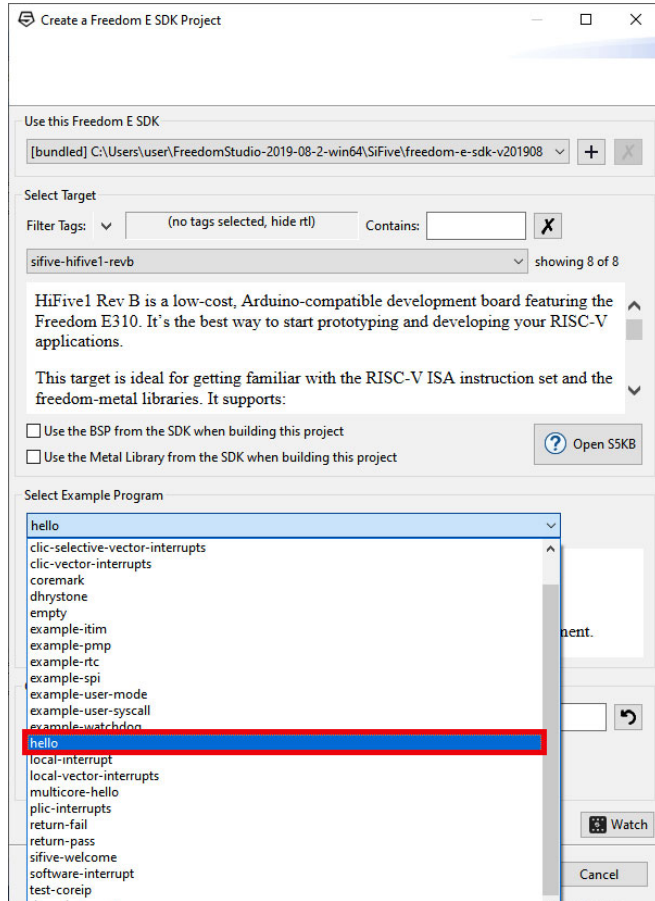


圖54：SiFive範例專案

這些範例專案是其他專案的絕佳起點。目前，請繼續使用*hello*。

5.4.5 偵錯啟動配置

在視窗底部，已自動設定「*Create a debug launch configuration for J-Link*」（建立J-Link的偵錯啟動配置）。原因是電路板已插入。按一下「*Finish*」（完成）。隨即將自動建立專案。

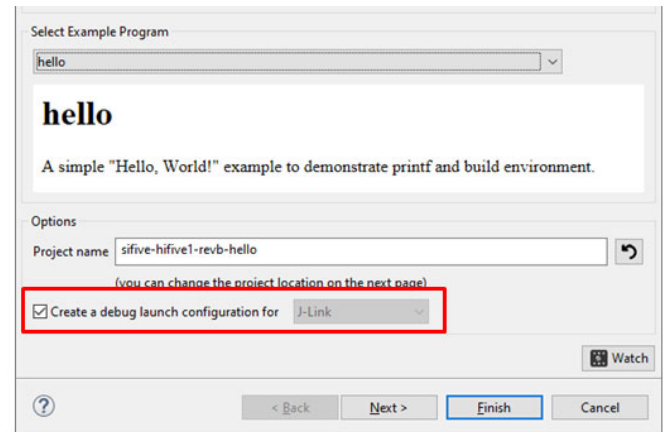


圖55：選擇範例程式和偵錯啟動配置

5.4.6 檢查.elf檔案

「*Edit Configuration*」（編輯配置）視窗將自動彈出。檢查是否已建立可執行檔和可連結檔*hello.elf*。這是設計電路板程式的必要步驟。

可按一下「*Debug*」（偵錯）直接偵錯，但目前需要先按一下「*Close*」（關閉），因為還有一些步驟需要設定。

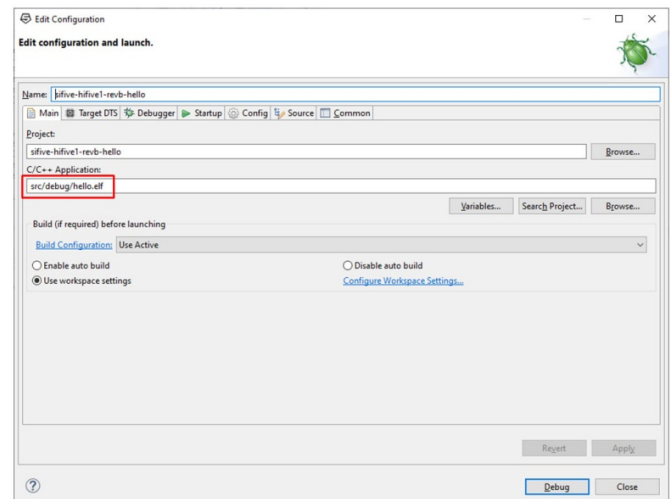


圖56：編輯配置功能表

5.4.7 控制台和原始程式碼視圖

「*Console*」（控制台）視窗提供了相關的建立資訊 – 所使用的編譯器、程式碼大小和建立時間。

第一次建立需要一段時間，因為它需要編譯專案中的所有檔案。後續建立只會重新編譯實際變更的檔案，因此速度要快得多。

hello.c視窗提供C程式碼。

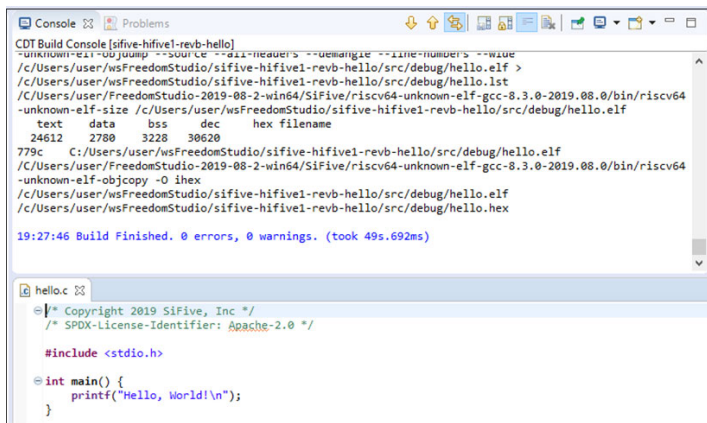


圖57：控制台和來源視窗

5.4.8 修改hello.c原始程式碼

如果按原狀偵錯hello.c程式，則不會看到很多內容，因為它只會執行終止操作。因此，將圖58中所示的hello.c修改為顯示Hello, World! 10次。

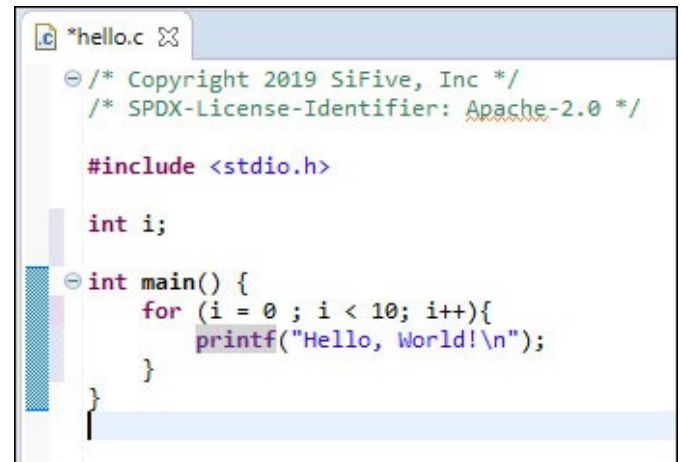




圖58：修改後的hello.c檔案

可按一下工具列的「*Build*」（編譯）圖示來建立程式以檢查語法，但當執行「*Debug*」（偵錯）時，該操作會自動完成。

5.4.9 偵錯程式

要開始偵錯程式，請按一下工具列上的「*Debug*」

（偵錯）圖示或按下鍵盤的F11鍵，也可先按一下工具列上的「*Run*」（執行）再按一下「*Debug*」（偵錯）。

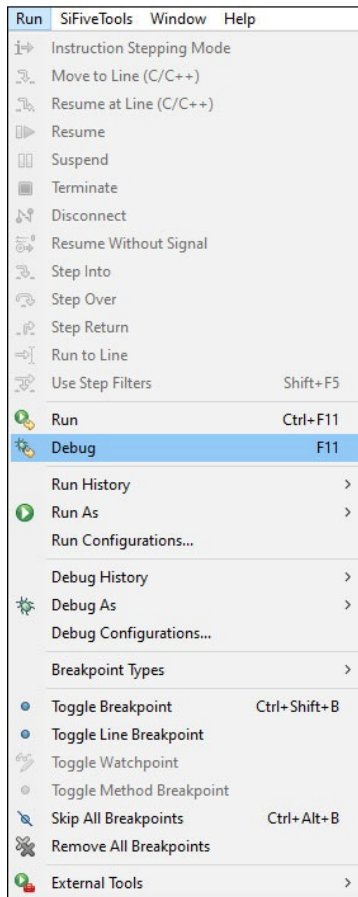


圖59：開始偵錯

5.4.10 偵錯工具在main()處停止

偵錯工具將在main()開始處停止。

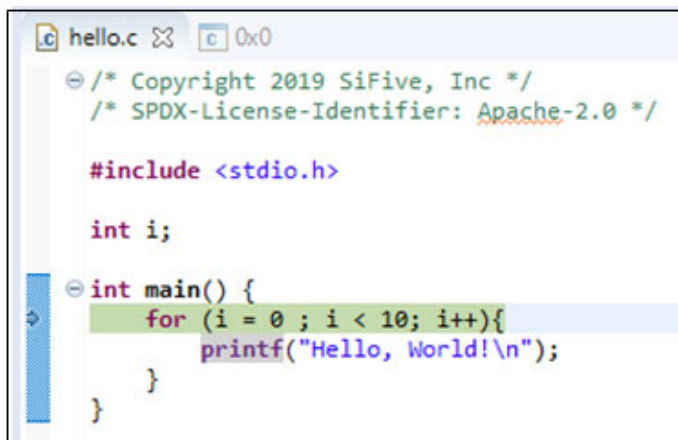




圖60：偵錯工具在main()處停止

5.4.11 設定用於printf的序列終端機

按一下「Terminal」（終端機） 索引標籤，然後按一下畫面右側的「Open a Terminal」（開啟終端機） 圖示。

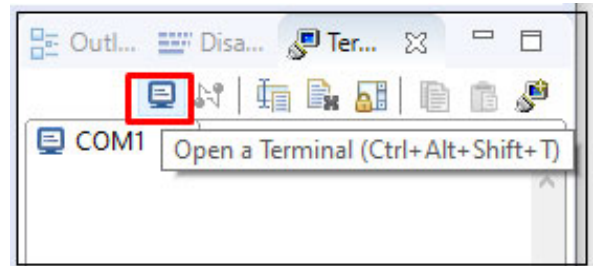


圖61：「Open a Terminal」（開啟終端機）圖示

5.4.12 啟動終端機

將序列終端機設定為115200傳輸速率。有多個序列埠，因此可能需要進行多次嘗試。

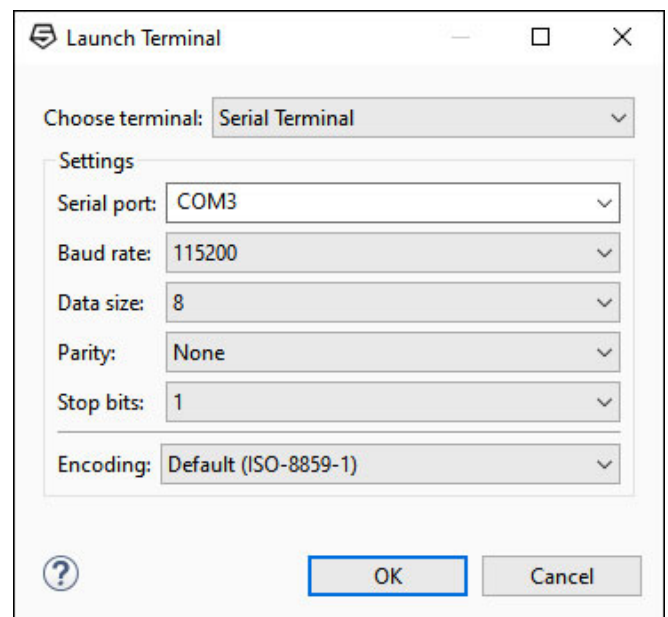



圖62：序列終端機設定

5.4.13 偵錯輸出

要單步執行程式碼，請按一下工具列上的「*Step Over*」（單步跳過）圖示，或多次按下鍵盤的F6功能鍵，以便使Hello, World!出現十次。

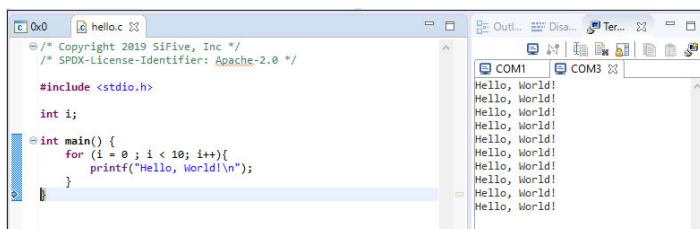


圖63：COM3上的偵錯輸出

printf訊息在同一程式的視窗中顯示，這對於偵錯和程式碼覆蓋非常有用。這比必須連接到單獨的終端機容易得多。

5.4.14 反組合視窗

圖64顯示了稍作修改的hello.c程式版本，用於與ARM®進行比較。

要檢視組成C程式碼的組合語言指令，請按一下「*Disassembly*」（反組合）索引標籤

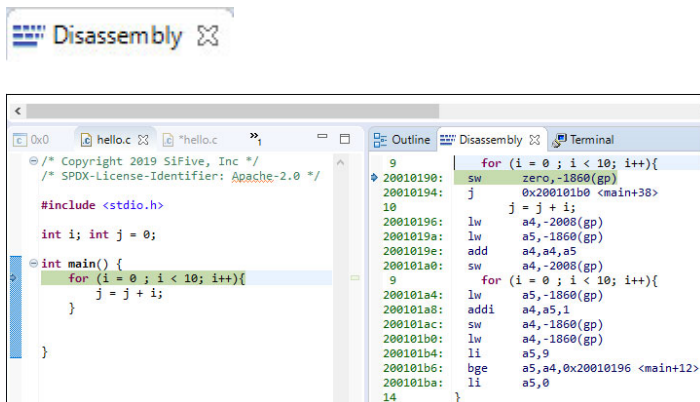


圖64：反組合視窗

5.4.15 與ARM Cortex M0的比較

圖65顯示了等效的ARM® Cortex® M0+反組合語言程式碼。

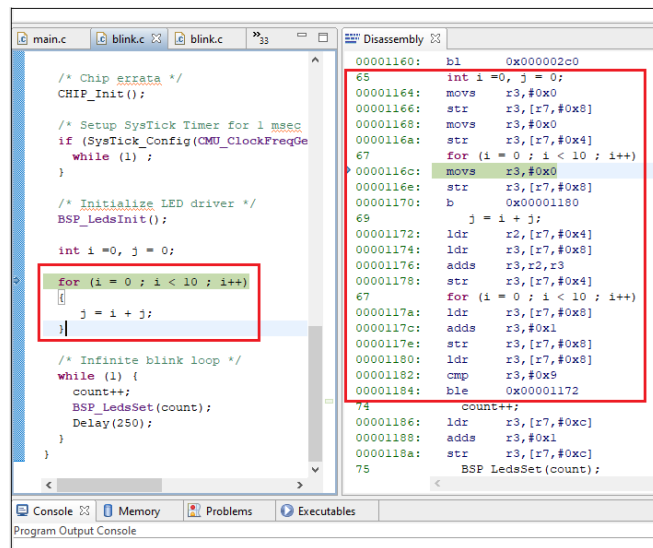




圖65：與ARM Cortex M0的比較

它表示在此情況下，RISC-V所需的指令少於ARM。

5.4.16 停止程式

要停止程式並從頭開始再次執行程式，請按一下工具列上或「*Console*」（控制台）視窗中的「*Terminate*」（終止）圖示。

要再次啟動偵錯程式，請按一下工具列上的「*Debug*」（偵錯）圖示, 或按下鍵盤的F11鍵。

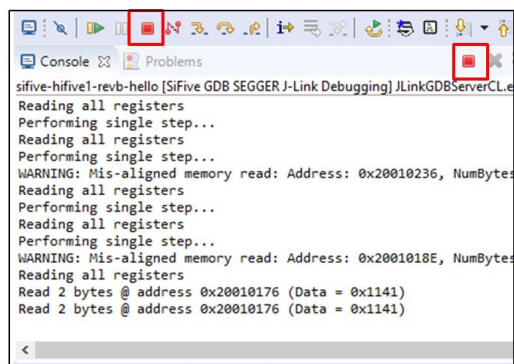


圖66：終止程式

5.4.17 關閉並重新開啟現有專案

當現有的Freedom Studio專案關閉並再次開啟時，會有一個小麻煩。關閉Freedom Studio，然後再次將其開啟。按一下專案，然後按一下「*Debug*」（偵錯）



圖示。程式無法偵錯，並顯示錯誤訊息：

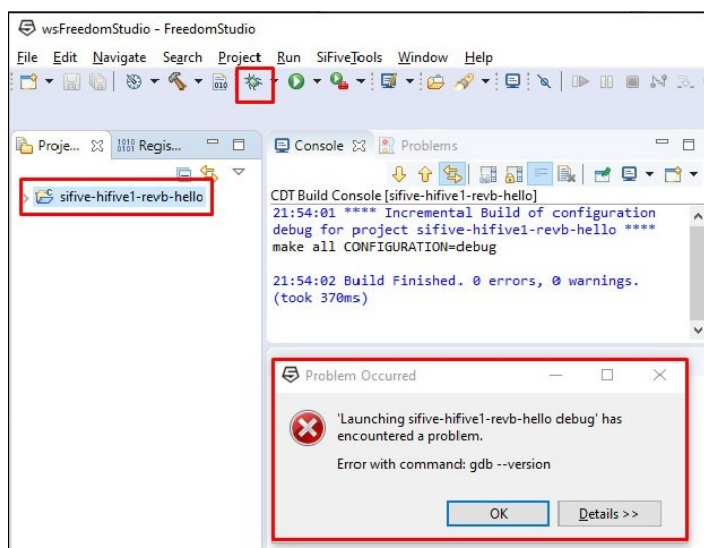


圖67：重新開啟專案時出現問題

5.4.18 選擇偵錯配置

從畫面頂部的工具列中，選擇「*Run*」（執行），然後選擇「*Debug Configurations*」（偵錯配置）。

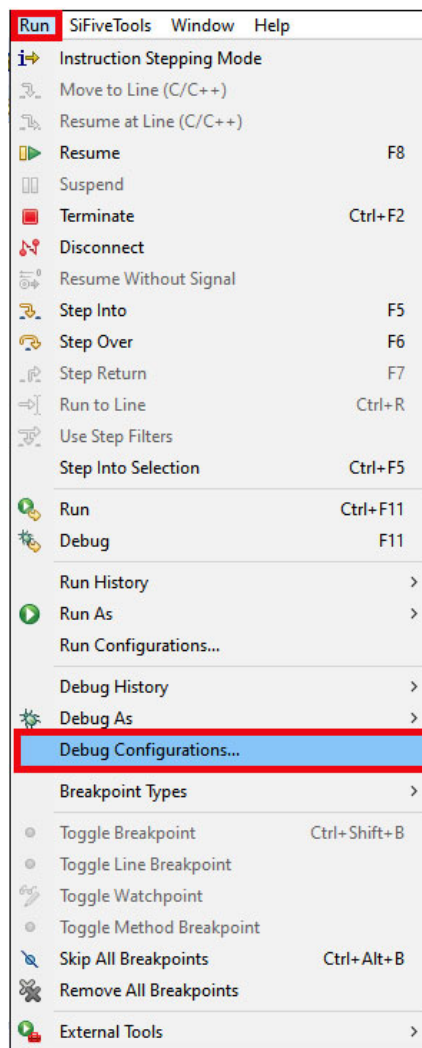


圖68：選擇偵錯配置

5.4.19 建立、管理和執行配置

展開「*Sifive GDB SEGGER J-Link Debugging*」（Sifive GDB SEGGER J-Link偵錯）索引標籤並按一下要偵錯的專案，此處為sifive-hifive-1-revb-hello。按一下「*Debug*」（偵錯）。現在應該可以偵錯程式了。

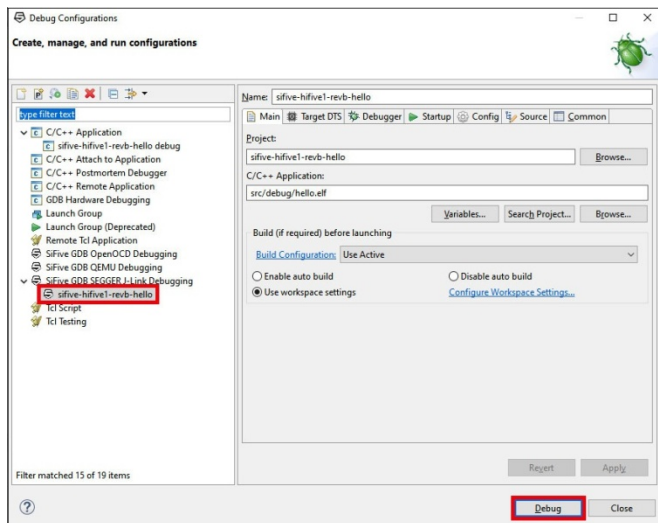


圖69：選擇要偵錯的專案

6 動手實驗：使用Digilent Nexys A7實作軟體核心

這是指南中對技術要求最高的部分，涉及將RISC-V核心嵌入到FPGA中，非常適合詳細研究電腦架構，但對於某些讀者而言可能過於複雜。

6.1 Imagination Technologies大學計劃提供的支援



在FPGA上實作軟體核心是一項艱巨的任務。幸運的是，Imagination Technologies大學計劃正在全球範圍內發佈如何在Xilinx Artix-7 FPGA上實作RISC-V核心的教材，也就是眾所周知的「RVfpga：瞭解電腦架構的完整課程」。這包括一整套的實驗（20個）、文件、範例和帶解決方案的練習。所有必要軟體都可以免費下載。這為使用者免去了大量的工作和棘手問題。這裡提供的材料版本佔整個課程的10%。

要獲得RVfpga材料，首先需要在以下網站上註冊：

<https://university.imgtec.com/forums/?wpforo=signup>

填寫註冊表單的過程可能有些長，因為這是專為大學所設計的。但是，RVfpga材料可供大學以外的人員使用，因此，在要求填寫與大學相關的欄位時，可以輸入來自商業或其他組織的等效內容。註冊所需的時間少於10分鐘，因為並非所有欄位都必須填寫。電話號碼不允許有空格。

註冊後，登入並請求從「Teaching Resources」（教學資源）頁面下載材料：

<https://university.imgtec.com/teaching-download/>

下載材料相當於大學三個學期的本科學習課程，涵蓋了20個實驗，而且全部免費。此外，還有相應的論壇可供探討問題和回答問題。

2021年第一季度將開設碩士級「建立SoC」課程。

6.2 軟體核心選項

假設專案不會耗用太大成本，那麼相較於使用標準處理器，在FPGA中使用軟體核心可以為設計人員提供更大的自由度。可以從www.opencores.org下載多種現成的週邊設備。快速傅立葉變換（Fast Fourier Transform, FFT）、數位濾波器和複雜加密算法等演算法可以在硬體而非軟體中實作，這有助於提高處理速度。由於幾乎可以將週邊設備連接到任何引腳，因此PCB佈線要容易得多。

6.3 執行程式所需的電路板

如果僅用於軟體模擬和軟體偵錯，則不需要電路板。

使用的電路板包括：

[Digilent Nexys A7 Nexys A7 ECE FPGA Trainer Board](#)

Digi-Key 1286-1081-ND（265.00美元）

也可以使用它的前一代產品（現在已經過時的Nexys4 DDR電路板），因為它們非常相似。



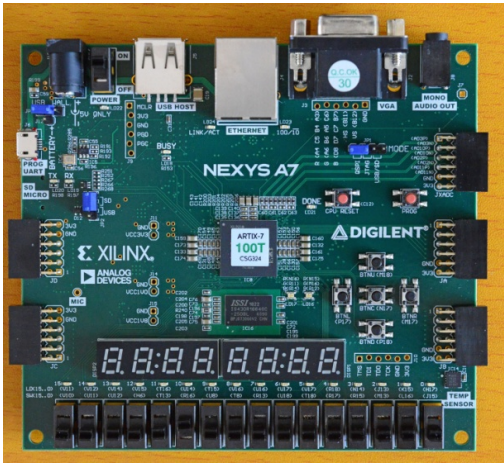


圖70：Digilent Nexys A7電路板

6.4 必要軟體

需要以下程式：

程式	用途
Xilinx Vivado 2019.2	建立構成RISC-V核心的Verilog .v和System Verilog .sv檔案。還會將產生的.bit檔案下載到硬體
Visual Studio Code (VSCode)	用於C語言和組合語言軟體開發的IDE
PlatformIO	VSCode的附加元件，可為RISC-V處理器提供支援
Open OCD	開放原始碼晶片上偵錯工具
RISC-V Toolchain	建立 RISC-V 專案所需的編譯器和其他檔案
Verilator	用於Verilog .v和System Verilog .sv檔案的硬體模擬器和處理器
Whisper	Western Digital的指令集模擬器（Instruction Set Simulator，ISS）。允許在電腦上執行和偵錯程式碼，無需硬體
Digilent Board Files	Digilent Nexys A7電路板的特定配置檔
GTKWave	波形檢視器，用於檢視底層系統時序
RVfpga	軟體核心、程式碼範例和Verilog/System Verilog專案
RVfpga Labs	20個實驗

表4：必要程式

所有程式目前僅在Linux上執行，但Windows和Mac支援正在開發中。完整的下載和安裝時間約為4小時。

6.5 執行RISC-V實作

要在Xilinx FPGA中的RISC-V上執行程式碼，需要執行兩項任務：

- 1. 使用.bit檔案配置FPGA硬體。
- 2. 編譯程式碼並將.elf檔案下載到FPGA中。

6.5.1 FPGA硬體和軟體元件之間的關係

圖71說明了如何使用Xilinx Vivado處理Verilog .v和System Verilog .sv檔案，並將其轉換為可編程到FPGA中的.bit檔案。

C語言或組合語言程式碼是使用PlatformIO進行編譯，並可將.elf檔案下載到支援其執行和偵錯的FPGA。

通過單一共享USB連接埠與Digilent Nexys A7電路板通訊。不能同時使用Vivado和Visual Studio Code與電路板通訊。



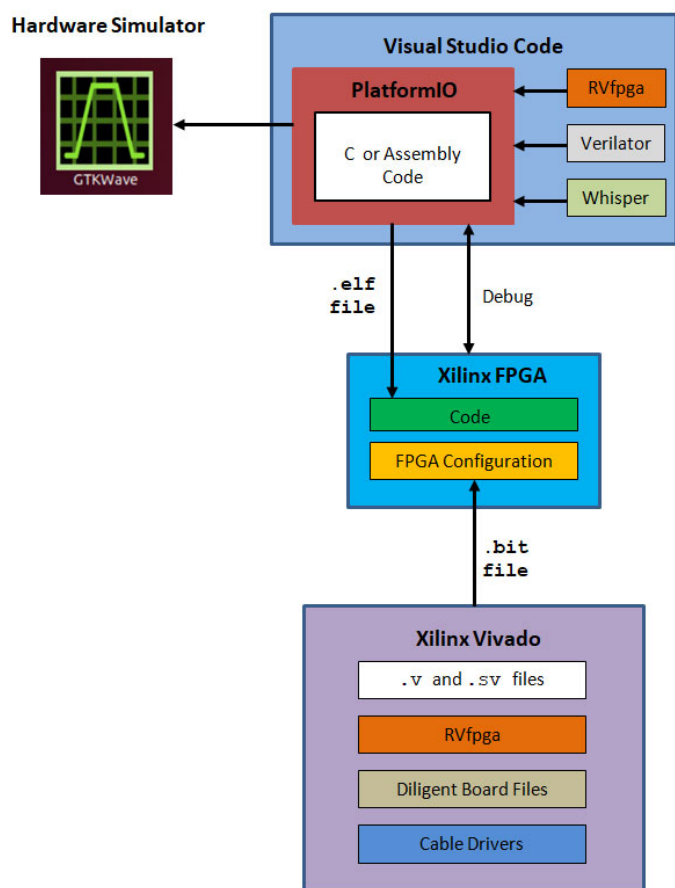


圖71：Visual Studio、FPGA和Xilinx Vivado的關係

6.6 Western Digital SweRV核心

儘管有各種各樣的RISC-V核心可用，但其中很多是學術研究的一部分，並未經過完全認證。並不是說這些核心不能正常工作，但是對於工業產品，RISC-V核心必須經過充分驗證和認證，以避免潛在的高昂產品責任成本。

Western Digital（2019年收入165.7億美元）開發了一個系列共三款SweRV核心（分別為EH1、EH2和EL2），經認證可用於商業產品。

6.6.1 SweRV核心清單

核心名稱	管線階段	執行緒	尺寸 (mm) @TSMC	CoreMark/ MHz
EH1	9	單	0.110@28 nm	4.9
EH2	9	雙	0.067@16 nm	6.3
EL2	4	單	0.023@16 nm	3.6

表5：Western Digital SweRV核心。圖片來源：Western Digital

此處的尺寸是指從台灣積體電路製造股份有限公司（Taiwan Semiconductor Manufacturing Company，TSMC）獲得的數據。

這三個核心均為RV32IMC，這意味著整數指令集、32位元和32個暫存器、整數乘法和除法以及壓縮指令（16位元）。

6.6.2 SweRV核心之間的區別

EH1和EH2核心均具有完整的管線功能，區別是EH1支援單執行緒，而EH2支援雙執行緒。EH2的規格更高（6.3 CoreMark/MHz），但架構也更複雜。

如果尺寸和成本十分重要，則EL2核心是一個不錯的選擇，但效能會下降到3.6 CoreMark/MHz。它使用具有4級而不是9級的簡化管線。

EH1核心已經在Diligent Nexys A7電路板上實作並經過了測試，因此已被選擇用於本指南的動手實驗部分。未來計劃新增效能更高的EH2核心和資源較少的EL2核心。

6.6.3 SweRV EH1核心詳細資訊

SweRV EH1核心的方塊圖如圖72所示：

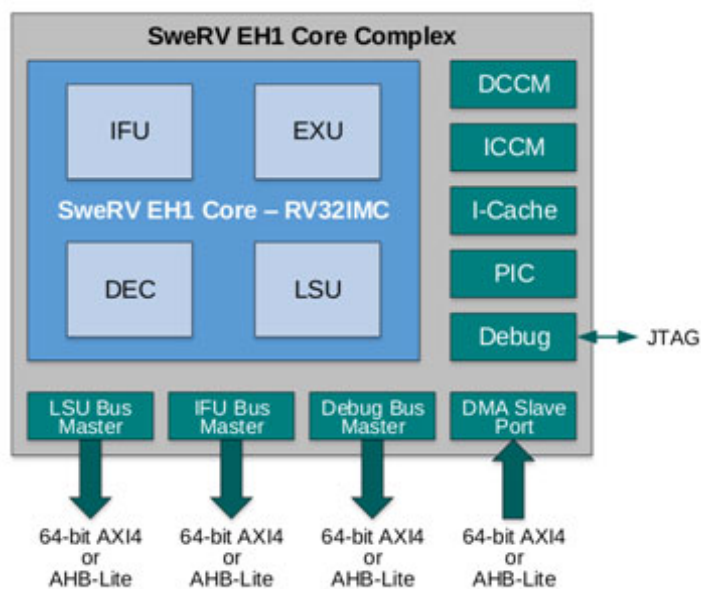


圖72：SweRV EH1核心組合。圖片來源：Western Digital

核心分為4個單元，分別為指令擷取（IFU）、執行（EXU）、解碼（DEC）和載入/儲存（LSU）。

6.6.4 SweRV EH1的各個元件

縮寫	說明
IFU	指令擷取單元
EXU	執行單元
DEC	解碼單元
LSU	載入儲存單元
DCCM	緊密耦合的資料記憶體
ICCM	緊密耦合的指令記憶體
I-Cache	指令快取
PIC	可程式化中斷控制器
AXI	高級可擴展介面
AHB	高級高效能匯流排

表6：SweRV EH1元件

AXI4和AHB-Lite是兩種行業標準互連匯流排。JTAG用於程式碼下載和偵錯。

6.7 SweRVolf核心

就其本身而言，SweRV EH1只是一個基本的電腦核心，它需要其他元件才能具有實際用途。Olof Kindgren為SweRV編寫了一組包裝函式，稱為SweRVolf。RVfpga的作者使用一些有用的附加週邊設備擴展了SweRVolf，並重新組織了HDL來源。

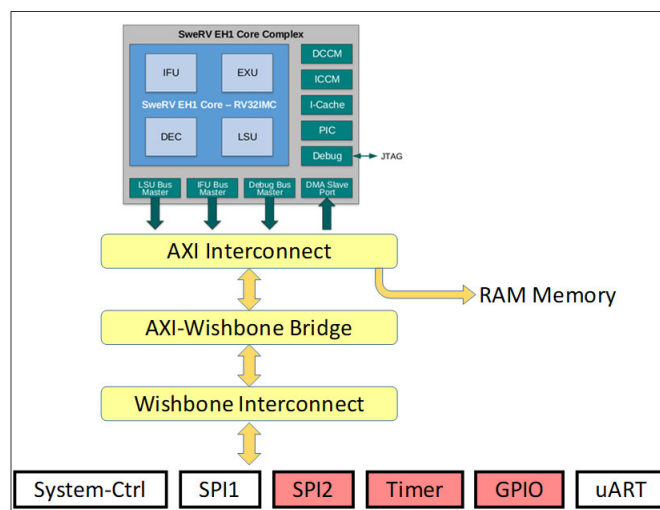


圖73：SweRVolf核心。圖片來源：Imagination Technologies UP

6.7.1 擴展SweRVolf元件

擴展SweRVolf實作分為以下元件：

- Wishbone 互連：Opencores 互連，比 AXI 更簡單。
- axi2wb：AXI轉Wishbone匯流排轉換器。
- GPIO：來自Opencores的通用輸入輸出，具有64個輸入/輸出連接埠。
- 定時器：來自Opencores
- SPI：開放原始碼序列週邊設備介面控制器（從https://opencores.org/projects/simple_spi獲取）。
- UART：開放原始碼UART控制器（從<https://opencores.org/projects/uart16550>獲取）。
- 開機ROM：開機ROM包含第一階段的開機載入器。重設系統後，SweRV將開始從該區域獲取初始指令。
- RAM：SweRVolf核心不包含記憶體控制器，但保留了其記憶體映射的前128 MB並可存取AXI匯流排，讓使用者得以加入RAM記憶體。
- SPI快閃記憶體：也可以使用上一節介紹的SPI控制器來包含SPI快閃記憶體（或四路SPI）。

6.7.2 SweRVolf記憶體映射

所有元件的記憶體映射如下：

核心映射	記憶體位址範圍
開機ROM	0x80000000 - 0x80000FFF
系統控制器	0x80001000 - 0x8000103F
SPI1	0x80001040 - 0x8000107F
SPI2	0x80001100 - 0x8000113F
定時器	0x80001200 - 0x8000123F
GPIO	0x80001400 - 0x8000143F
UART	0x80002000 - 0x80002FFF

表7：SweRVolf記憶體映射

6.8 SweRVolf Verilog和System Verilog 檔案

圖74的檔案階層以Verilog .v和System Verilog .sv格式顯示。這些均由RVfpga套件提供。

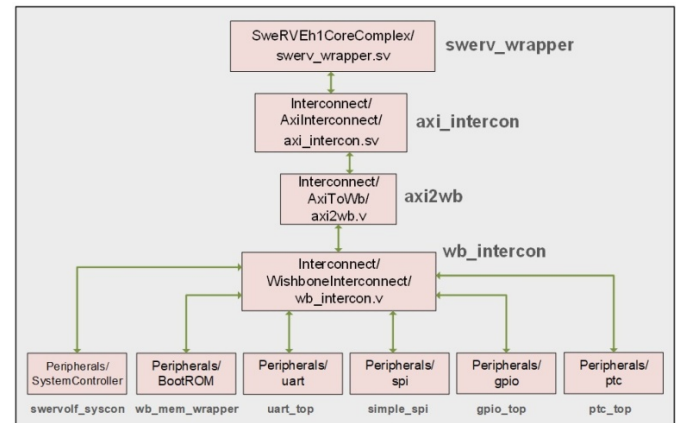


圖74：SweRVolf檔案結構。圖片來源：Imagination Technologies UP

6.9 Nexys A7的SweRVolf階層

RVfpga套件提供了SweRVolf專案的所有元件，供使用者下載。

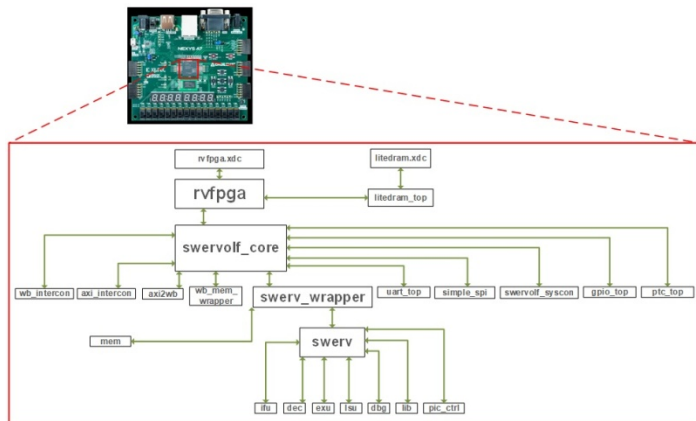


圖75：Nexys A7中的SweRVolf專案階層。圖片來源：Imagination Technologies UP

Communication controller 6	
Project	Files
★ Ethernet 10GE MAC	●
★ Ethernet MAC 10/100 Mbps	●
★ I2C controller core	●
★ sd card controller	●
★ Small 1-wire (onewire) master, with Altera tools integration	●
★ UART to Bus	●
Crypto core 3	
Project	Files
★ AES	●
★ Avalon AES ECB-Core (128, 192, 256 Bit)	●
★ SHA3 (KECCAK)	●
ECC core 2	
Project	Files
★ Reed Solomon Decoder (204,188)	●
★ Viterbi Decoder (AXI4-Stream compliant)	●

圖76：一些經過認證的開放原始碼核心。來源：www.opencores.org

6.10 增強SweRVolf實作

可新增其他週邊設備來擴展原始SweRVolf實作。

6.10.1 Opencores提供的週邊設備

SweRVolf是一種開放規格，所有原始程式碼都可以從Github下載。這意味著可以對其進行修改，而且可以新增週邊設備。www.opencores.org/projects提供了各種各樣的專案。圖76顯示了一些經過認證的開放原始碼核心的範例：

6.10.2 實驗室中新增的週邊設備

在實驗室中，為Nexys A7電路板新增了以下週邊設備：

- 5個按鈕（通過新GPIO實作）
- 三色LED（使用PWM）
- 溫度感應器（使用I²C）
- 7段顯示器驅動程式
- SPI加速計驅動程式。

7 軟體核心軟體安裝

7.1 下載RVfpga

在安裝用於軟體核心開發的軟體之前，需要下載RVfpga套件，因為它包含若干必要檔案。有關詳細資訊，請參閱第6.1節。

下載所有檔案。RVfpga套件的理想安裝位置是主目錄，例如 /home/myusername/RVfpga，其中「myusername」是實際的電腦名稱。當一個範例如下時：

```
[RVfpgapath] /RVfpga/
```

只需輸入：

```
~/RVfpga/
```

請確保所有檔案都具有讀/寫權限。

以下範例使用了RVfpga的最新版本，可能會有所變化。建議參考Imagination Technologies「RVfpga：入門指南」來瞭解最新資訊。

7.2 關於Xilinx Vivado

這是到目前為止最耗時的動作，因為在Ubuntu 18.04 Linux中需要設定多個程式。Windows和Mac支援目前正在開發。Xilinx下載和安裝需要約2.5小時。

選擇Xilinx有以下幾個原因：

1. Xilinx處於FPGA市場的高端，具有極為豐富的高規格產品。

2. 從職業角度和市場技能來看，FPGA技術對Xilinx的需求最為廣泛。只需快速搜尋求職網站即可確認這一點。
3. SweRVolf是使用Xilinx編寫的。
4. SiFive也為其核心使用Xilinx。

7.3 有關在Linux中安裝Xilinx Vivado的重要資訊

在執行Xilinx下載之前（如果尚未完成），需要在電腦上安裝Ubuntu 18.04。Ubuntu磁碟分割至少需要100 GB。不要使用Ubuntu 20.04。

Xilinx指定了哪些版本的Ubuntu與其軟體相容，而Ubuntu 20.04目前不在相容版本之列。

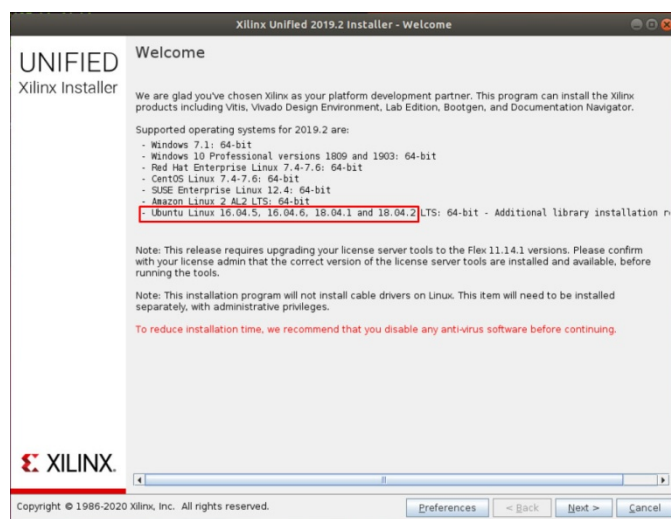


圖77：Xilinx統一安裝程式相容性

不要像作者一樣下載Ubuntu 20.04，結果在2個半小時後才發現它不相容且無法執行。該過程將需要從頭開始，刪除所有內容，然後安裝正確的版本Ubuntu 18.04。

7.3.1 Xilinx Vivado系統要求

Xilinx Vivado的大小為16 GB（因此下載時間較長），並且需要56 GB的安裝空間（因此磁碟分割大小為100 GB）。如果磁盤空間不足，將產生錯誤。

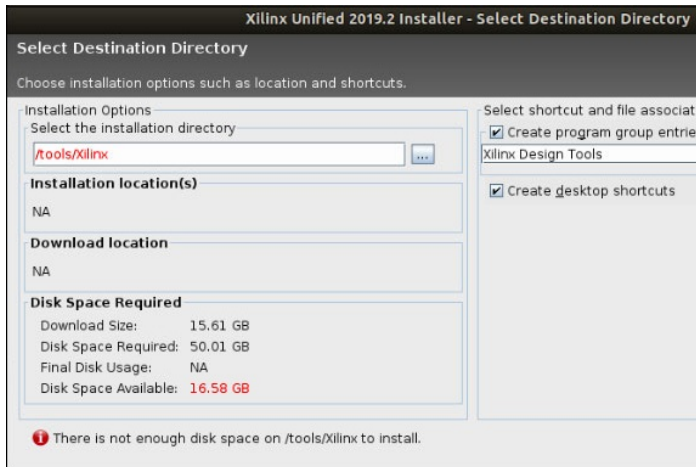


圖78：Xilinx Vivado系統要求

7.4 在Linux中安裝Xilinx Vivado

該程式用於為軟體核心修改和建立Verilog .v和System Verilog .sv檔案，並將其下載到Digilent Nexys A7電路板。

如果沒有Digilent Nexys A7電路板，或者該電路板僅用於執行電腦模擬，則可以稍後再安裝Vivado。

7.4.1 下載Xilinx軟體

要下載Xilinx軟體，需要建立一個Xilinx帳戶（如果尚不存在）。

移至<https://www.xilinx.com/support/download.html>，選擇2019.2版本進行下載

按一下「Xilinx Unified Installer 2019.2: Linux Self Extracting Web Installer」（Xilinx統一安裝程式2019.2：Linux自我解壓縮Web安裝程式）進行下載。

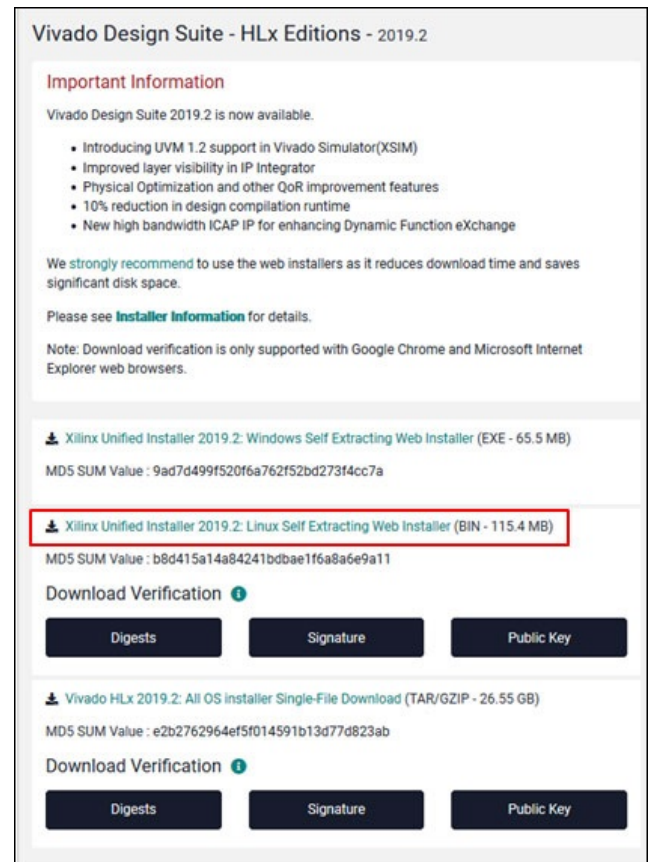




圖79：Xilinx統一安裝程式2019.2

7.4.2 變更二進位檔案的權限


當Xilinx_Unified_2019.2_1106_2127_Lin64.bin檔案下載到/Downloads目錄後，它是無法執行的。

使用「Ubuntu Files」（Ubuntu檔案）工具，用滑鼠右鍵按一下檔案名稱，然後選擇「Properties-Permissions」（屬性-權限），將其變為可執行檔。按一下「Allow executing file as program」（允許將檔案作為程式執行）。

7.4.3 執行二進位檔案

開啟Ubuntu終端機 ，輸入以下內容將其設定為根：

```
sudo su
```

返回到「Ubuntu Files」（Ubuntu檔案）  工具，將以下檔案拖放到「Terminal」（終端機）視窗：

```
Xilinx_Unified_2019.2_1106_2127_Lin64.bin
```

在Ubuntu終端機中按下歸位字元，執行二進位檔案。

7.4.4 選擇要安裝的Vivado產品

在「Select Product to Install」（選擇要安裝的產品）視窗中，選擇Vivado而不是Vitis。

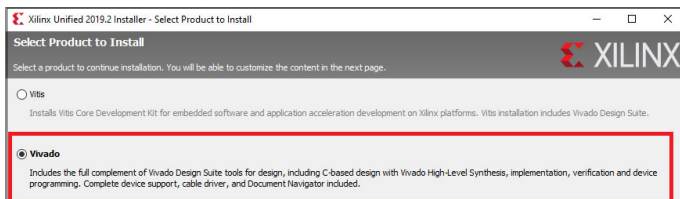


圖80：選擇Vivado

7.4.5 選擇WebPACK

在「Select Edition to Install」（選擇要安裝的版本）視窗中，選擇免費版本 Vivado HL WebPACK。

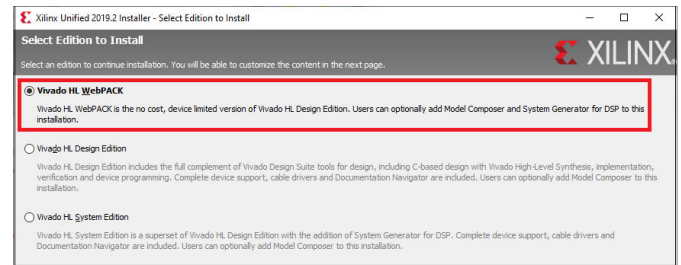


圖81：選擇要安裝的版本

按照剩餘步驟完成安裝，並使用預設值。

7.4.6 安裝Digilent Nexys A7電路板的纜線驅動程式

開啟Ubuntu終端機  並導覽至：

```
/tools/Xilinx/Vivado/2019.2/data/xicom/cable_drivers/lin64/install_script/install_drivers/
```

然後輸入：

```
sudo ./install_drivers
```

7.4.7 安裝電路板檔案

移至<https://github.com/Digilent/vivado-boards>。

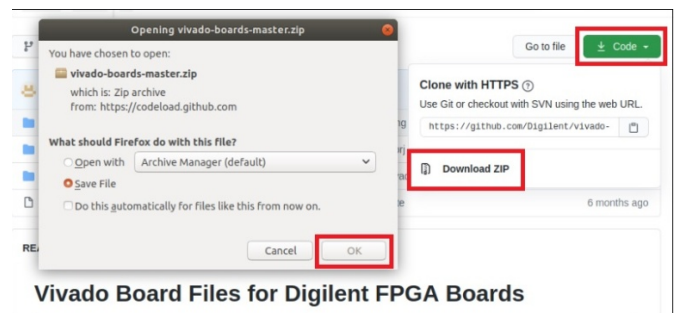



圖82：下載Vivado Boards Master

下載 .zip 檔案並使用「*Ubuntu Files*」(Ubuntu 檔案)  工具將其解壓縮。

在 *Ubuntu* 終端機  中導覽到：

```
/Downloads/vivado-boards-master/new/board_files
```

需要將此處的檔案複製到 Vivado board_files 目錄。輸入：

```
sudo cp -r *  
  
/tools/Xilinx/Vivado/2019.2/data/boards/board_files
```

重要的電路板檔案是 nexys-A7-100t。

7.4.8 在Linux下執行Vivado

開啟 *Ubuntu* 終端機 。

導覽到目錄：

```
/tools/Xilinx/Vivado/2019.2，然後輸入：  
  
source settings64.sh  
  
vivado &
```

7.5 安裝Visual Studio Code和PlatformIO

所需時間：約20分鐘。

Visual Studio Code (VSCode) 是用於軟體開發和偵錯的IDE。

PlatformIO 是 VSCode 的附加元件，可為 RISC-V 和許多其他處理器提供支援。

7.5.1 下載Visual Studio Code

移至 <https://code.visualstudio.com/Download>。按一下 .deb 下載檔案。檔案大小目前為 60.5 MB

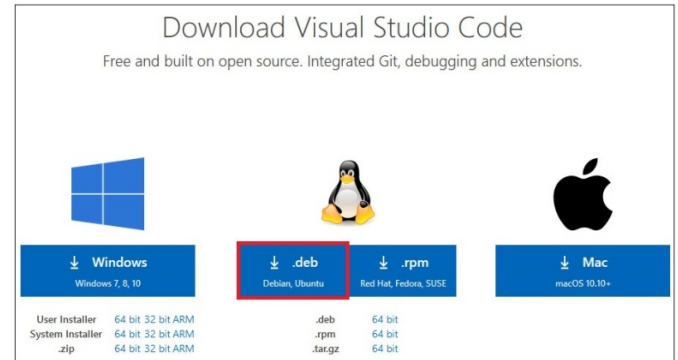


圖83：下載Visual Studio Code

7.5.2 解壓縮Visual Studio Code

開啟 *Ubuntu* 終端機  並輸入：

```
cd ~/Downloads  
  
sudo dpkg -i code*.deb  
  
要執行VSCode，輸入：  
  
code
```

7.5.3 安裝PlatformIO

PlatformIO 為 RISC-V 提供特定支援。

在 *Ubuntu* 終端機  中輸入下列指令，以載入一些必要的公用程式：

```
sudo apt install -y python3-distutils  
python3-venv
```

如果尚未執行VSCode，則在搜尋功能表中輸入Visual Studio Code，然後選擇它將其定位。或者，按一下 *Visual Studio Code* 圖示。

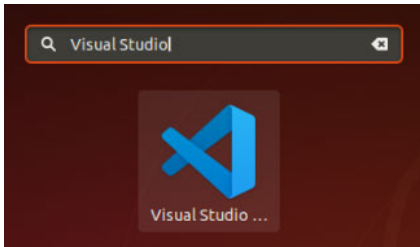


圖84：Visual Studio Code圖示

7.5.4 VS Code中的PlatformIO擴展

在VSCode中，按一下「*Extensions*」（擴展）圖示

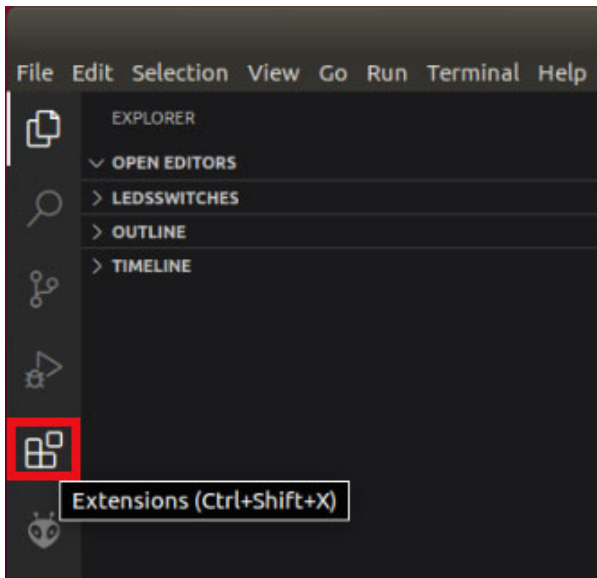


圖85：VSCode擴展軸

7.5.5 安裝PlatformIO擴展功能

在搜尋方塊中輸入platformio。移至 *PlatformIO IDE*，然後按一下「*Install*」（安裝）按鈕。

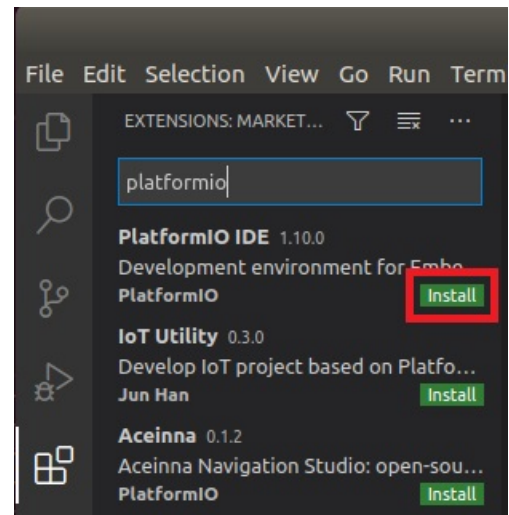


圖86：PlatformIO IDE擴展

7.5.6 重新載入PlatformIO

底部的「*OUTPUT*」（輸出）視窗提供有關安裝進度的資訊。

完成後，按一下「*Reload Now*」（立即重新載入），這樣PlatformIO核心便會安裝到VSCode中。

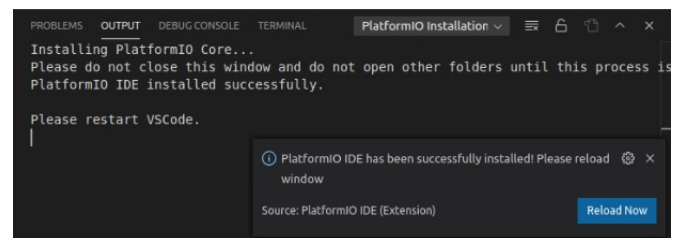



圖87：安裝PlatformIO後立即重新載入

7.5.7 安裝Chips Alliance套件

下一步是選擇處理器目標。本例中為SweRVolf，它是已下載RVfpga套件的一部分。

按一下 **PlatformIO**  按鈕（位於左側軸上）以檢視「Quick Access」（快速存取）功能表，然後按一下「PlatformIO Core CLI」（PlatformIO 核心 CLI）選項。

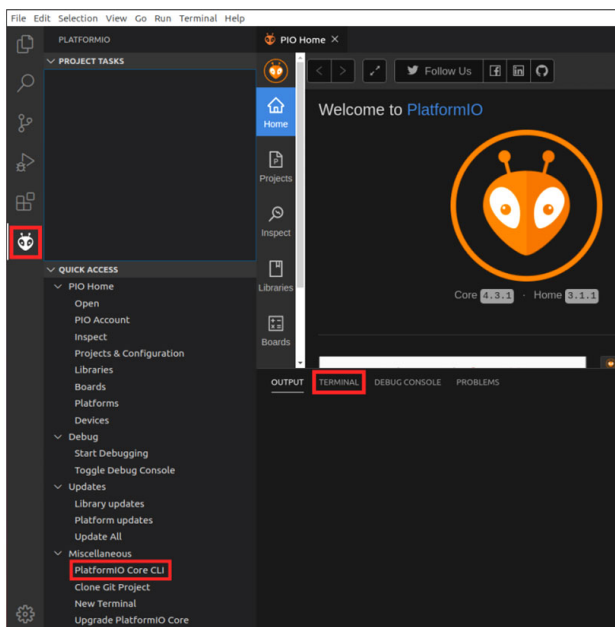


圖88：選擇PlatformIO核心目標

在底部視窗中，從「OUTPUT」（輸出）視窗變更為「TERMINAL」（終端機）視窗。這允許在VSCode中輸入命令。

7.5.8 安裝Chips Alliance平台

通過在「TERMINAL」（終端機）上執行以下命令來安裝Chips Alliance平台：

```
~/platformio/penv/bin/pio platform
install [RVfpgaPath]/RVfpga/platform-
chipsalliance --with-all-packages
```

其中 [RVfpgaPath] 是 RVfpga 檔案的儲存位置。例如，它可能位於 /home/myname/，其中 myname 是實際的電腦名稱。

該套件包含預先建立的RISC-V工具鏈、用於RISC-V的OpenOCD偵錯工具、Verilator、Whisper、JavaScript和Python指令碼。此外，還提供了RVfpga位檔案和範例程式碼。

7.5.9 （選用）檢查是否已載入SweRVolf

要確認swervolf_nexys是否已載入，在「TERMINAL」（終端機）視窗中輸入：

```
~/platformio/penv/bin/pio boards | more，
然後向下捲動到swervolf_nexys
```

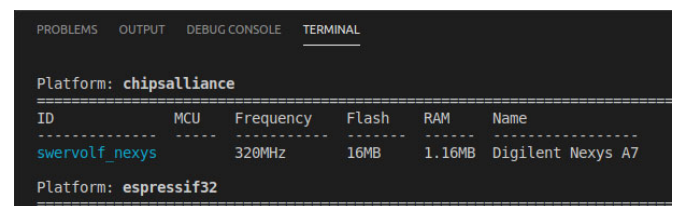


圖89：檢查swervof_nexys是否已載入

7.5.10 關閉並重新開啟VSCode

啟動PlatformIO後，請關閉並重新開啟VSCode。

備註：一般情況下，.platformio檔案為隱藏狀態。

要在Ubuntu終端機  中搜尋.platformio，輸入：

```
ls -a
```

這會使隱藏的檔案可見。

或者，使用「*Ubuntu Files*」（Ubuntu檔案）工具



顯示隱藏檔案。

7.6 安裝Verilator和GTKWave

所需時間：約20分鐘。

Verilator是適用於PlatformIO使用的Verilog和System Verilog檔案的硬體模擬器。

要安裝將在Linux中本機執行的Verilator，必須開啟



Ubuntu終端機。

首先安裝預設情況下未安裝的必要公用程式：

```
sudo apt-get install git make autoconf
g++ flex bison libfl2 libfl-dev
```

安裝Verilator程式

```
git clone
https://git.veripool.org/git/verilator
cd verilator
git pull
git checkout v4.020
autoconf
./configure
make
sudo make install
```

最後安裝波形檢視器

```
sudo apt-get install -y gtkwave
```

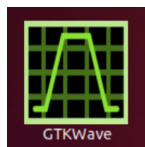


圖90：GTK Wave圖示

7.6.1 典型Verilator波形

Verilator用作硬體模擬器時十分複雜，超出了本文的範圍。「RVfpga：入門指南」提供了完整的詳細資訊。但是，時鐘和取指單元（Instruction Fetch Unit，IFU）的典型波形為：

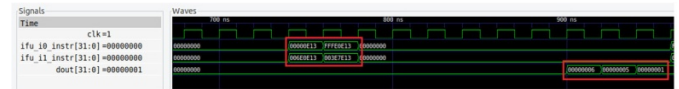


圖91：典型Verilator波形。

7.7 使用Whisper在PlatformIO中執行C程式

Western Digital有一個名為Whisper的指令集模擬器（Instruction Set Simulator，ISS），是針對在沒有任何硬體的情況下，對SweRV RISC-V核心進行軟體驗證所開發的。它是RVfpga下載包的一部分。

7.7.1 開啟資料夾

開啟VSCode並進入PlatformIO。

在PlatformIO的頂端功能表軸上，按一下「File」（檔案），然後按一下「Open Folder」（開啟資料夾）。無需實際開啟檔案。

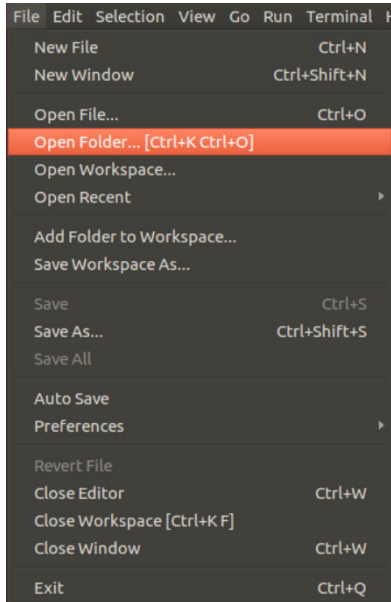


圖92：開啟模擬檔案的資料夾

7.7.2 Vector Sorting Whisper範例程式碼

選擇目錄VectorSorting，然後按一下「OK」（確定）。

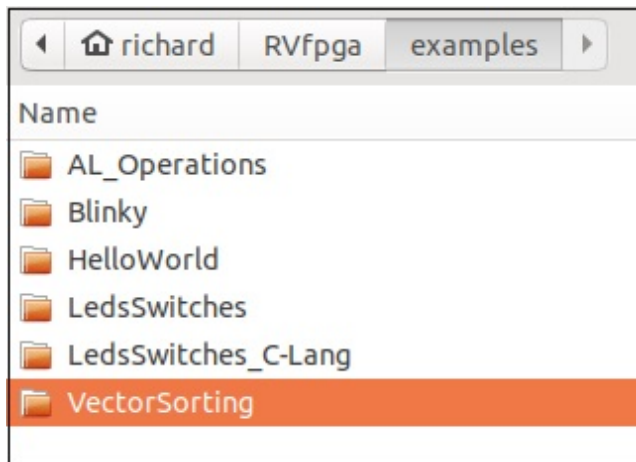


圖93：VectorSorting資料夾

7.7.3 修改platformio.ini

要使用Whisper，必須修改platformio.ini檔案。開啟該檔案。

取消註釋行：

```
debug_tool = whisper
```

然後儲存它。

重要資訊：確保debug_tool與上面的字元對齊，且前面沒有空格。圖94顯示了存在意外空格時產生的錯誤。

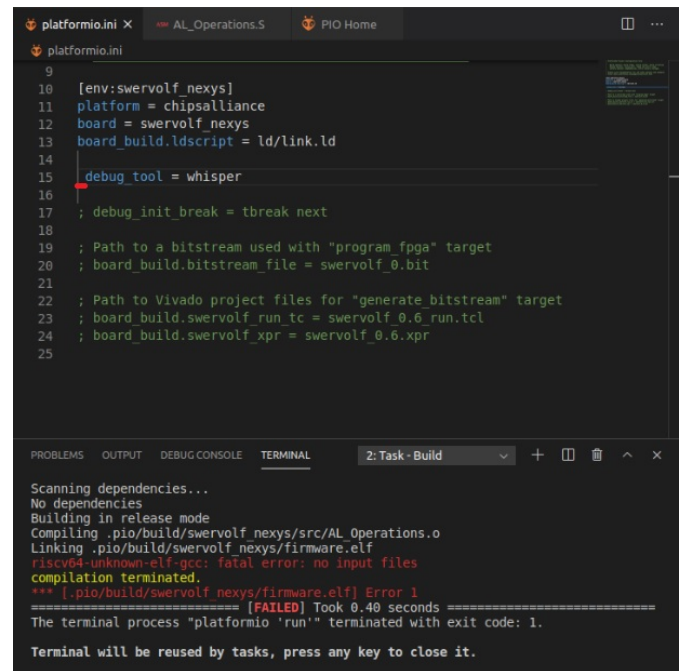


圖94：錯誤的platformio.ini設定

7.7.4 使用Whisper執行模擬

通過按一下第10行左側放置一個中斷點。

通過依序按一下  和  來啟動偵錯工具。

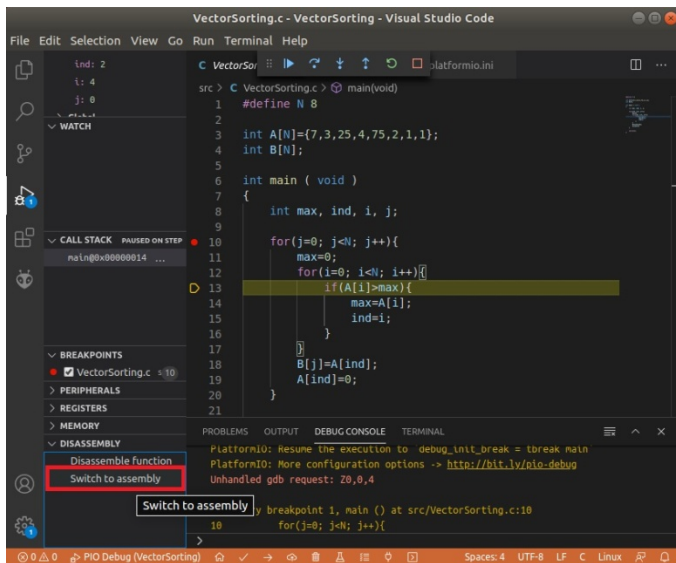


圖95：顯示第10行的程式RISC-V指令和中斷點

現在可以使用鍵盤功能鍵F10單步執行程式碼。

按一下畫面左下角的「Disassembly」（反組合）可顯示RISC-V指令。

7.7.5 檢視RISC-V組合語言

C編譯器產生的組合語言如圖96所示。

要返回到C原始程式碼，按一下「Switch to code」（切換到程式碼）。

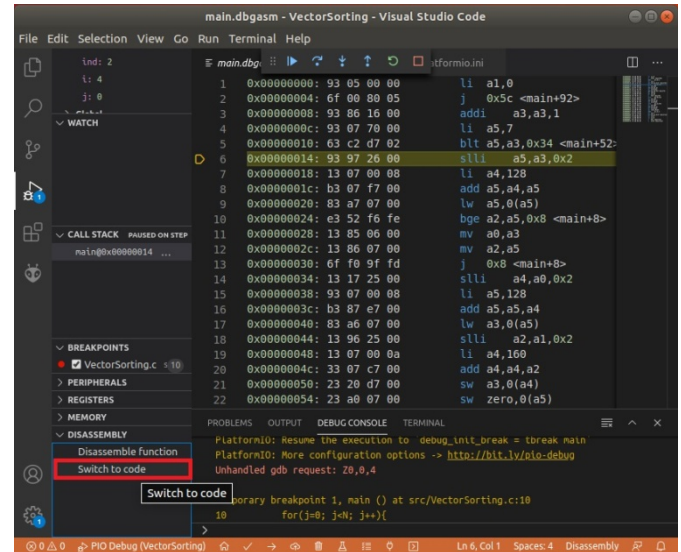


圖96：RISC-V組合語言視圖

有關偵錯的詳細資訊，請參閱「RVfpga：入門指南」。

8 建立SweRVolf核心並將其下載到FPGA

所需時間：約5分鐘

8.1 建立SweRVolf核心

好消息是不需要執行此部分，這要歸功於馬德里康普頓斯大學的RVfpga團隊。作為RVfpga套件的一部分，Xilinx Vivado專案提供了RVfpga.bit檔，該檔案可直接下載到Digilent Nexys A7硬體中。

8.2 連接Nexys A7電路板

使用隨附的USB纜線插入Nexys4 A7電路板。將電源開關滑動到開啟位置，因為出廠預設值為關閉。

8.3 將核心下載到Digilent Nexys A7

這需要先下載RVfpga套件，因為其中包含RVfpga.bit檔。請參閱第6.1節。

8.3.1 在Vivado中開啟硬體管理員

按第7.4.8節所示開啟Vivado。

按一下「Open Hardware Manager >」（開啟硬體管理員）



圖97：開啟Vivado硬體管理員

8.3.2 開啟硬體目標

硬體管理員將自動偵測未開啟的硬體。按一下「Open Target」（開啟目標），然後按一下「Auto-detect」（自動偵測）。

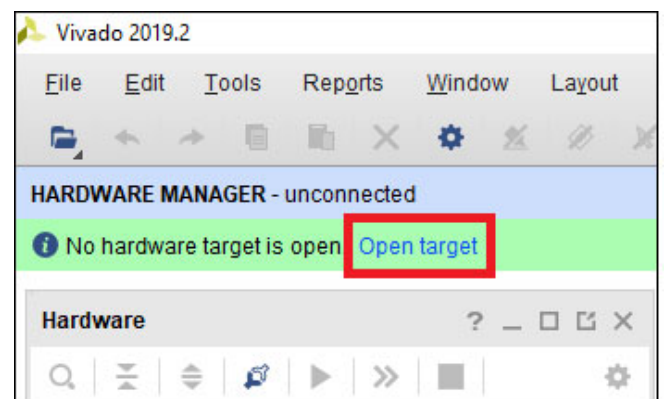


圖98：開啟硬體目標

8.3.3 設計FPGA配置的程式

按一下「*Program device*」（設計裝置程式）。

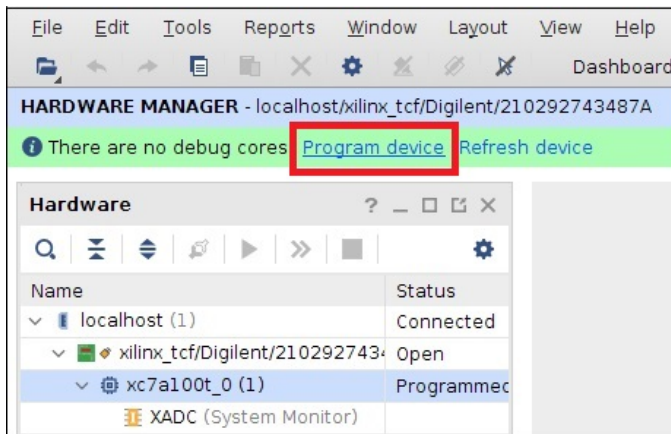


圖99：設計FPGA配置的程式

8.3.4 選擇位元串流檔

對於位元串流檔，請選擇：

[RvfpgaPath]/RVfpga/src/RVfpga.bit

「*Debug probes file*」（偵錯探頭檔）可以留空。按一下「*Program*」（程式設計）。電路板應在大約10秒鐘內進行程式設計。

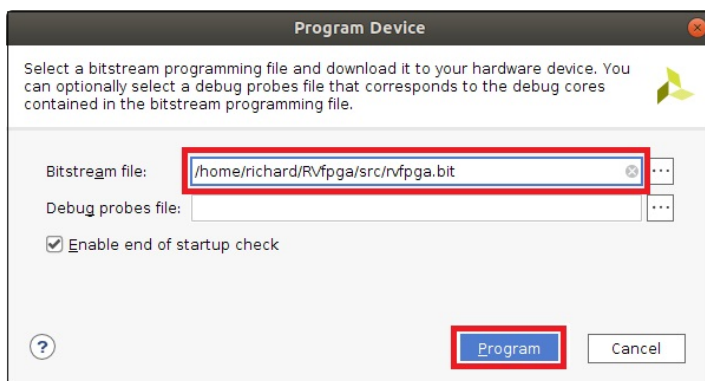



圖100：下載位元串流檔

8.3.5 關閉Vivado

這是重要的一步。退出Vivado以釋放USB連接埠，否則無法使用VSCode下載和執行程式。

按一下右側的「*Close Hardware Manager*」（關閉硬體管理員）圖示 。

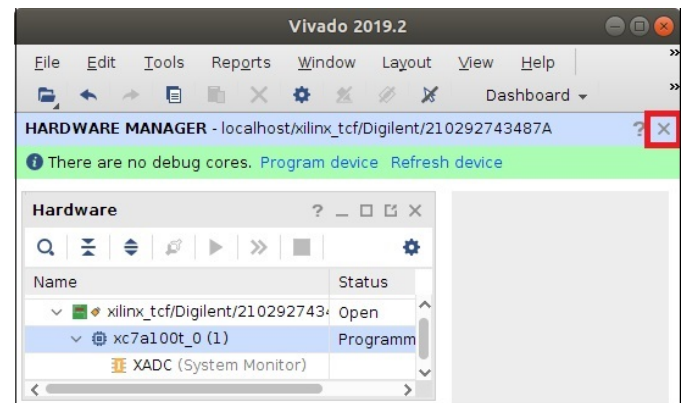


圖101：退出Vivado

8.4 在NexysA7電路板上執行程式

所需時間：約5分鐘。

8.4.1 開啟要執行的程式

在搜尋軸中輸入Visual Studio Code，再按一下

VSCode  圖示以開啟VSCode。

在頂端功能表軸上，按一下「*File*」（檔案），然後按一下「*Open Folder*」（開啟資料夾）。

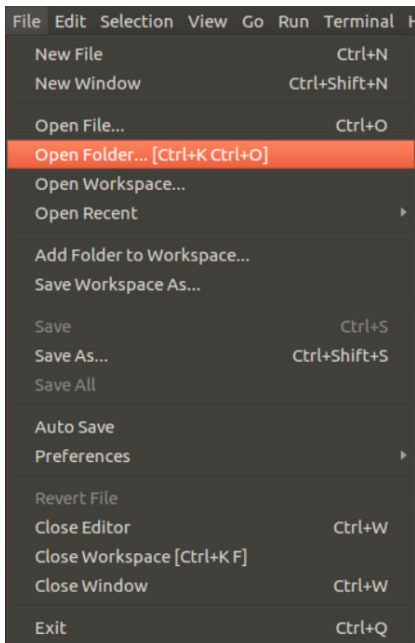


圖102：開啟資料夾

無需開啟檔案。

8.4.2 選擇程式LedsSwitches

導覽到RVfpga/examples，然後按一下LedsSwitches。其他範例可以稍後再試。

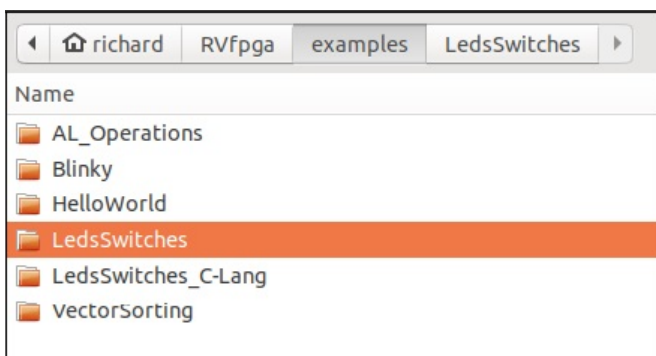


圖103：LED和開關程式

8.4.3 開啟LED和開關程式

這是一個最小的組合語言程式，它會持續讀取16個開關，然後將值輸出到16個LED。RVfpga範例中還有一個C程式碼版本。

展開>src，然後按一下LedsSwitches.s以顯示組合語言程式碼。

按一下畫面左下方的「PlatformIO Build」（PlatformIO 建立）圖示。

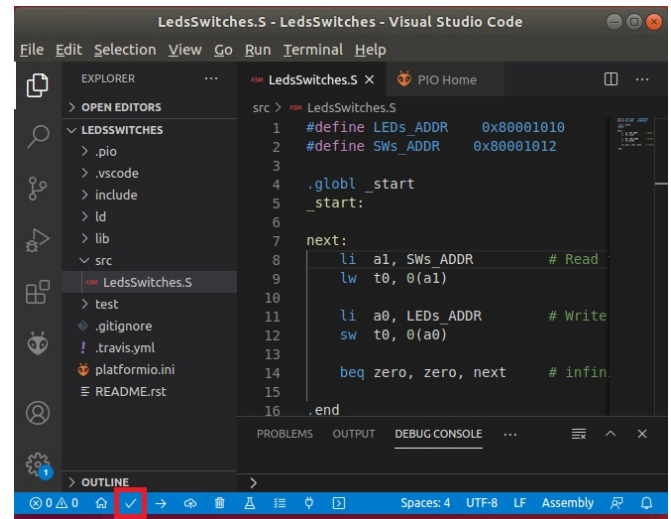


圖104：建立LedsSwitches程式

8.4.4 執行LedsSwitches程式

從工具列中選擇「Run」（執行），然後選擇「Start Debugging」（開始偵錯），或者按下鍵盤功能鍵F5。程式將持續執行。

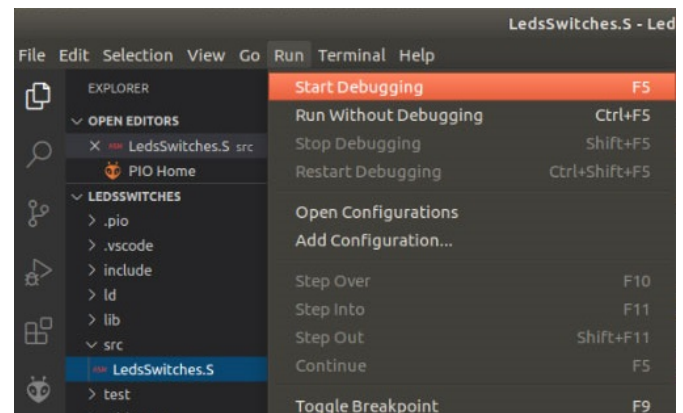


圖105：開始偵錯LED和開關程式

8.4.5 執行RISC-V核心的Nexys A7電路板 - 由開關控制的LED

操作電路板底部的滑動開關。綠色LED可以單獨點亮和熄滅。

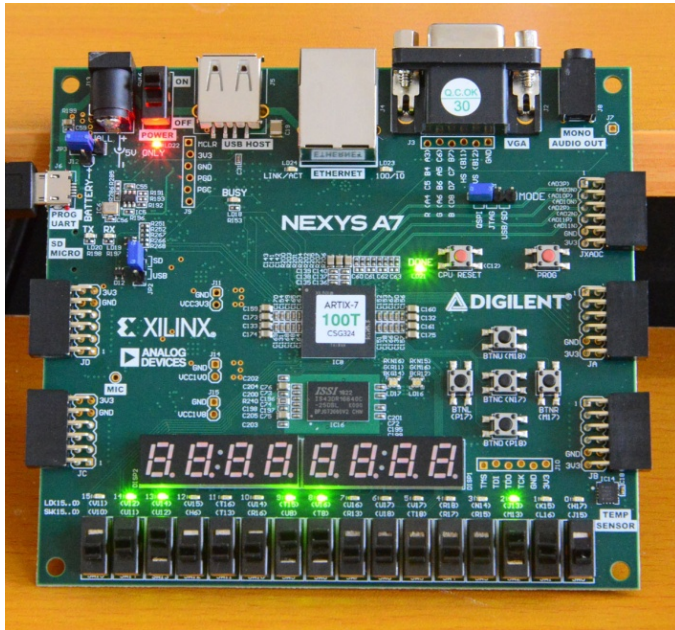


圖106：執行RISC-V核心的Nexys A7電路板

備註：在最新版本的RVfpga.bit上，7段顯示器顯示0000 0000。為了清楚起見，這裡顯示了一個簡單的版本。

9 使用RISC-V進行產品開發

在本指南中，我們使用Kendryte K210和SiFive FE310-G002這兩款SoC RISC-V處理器完成動手實驗。本章介紹這兩款裝置在安裝到使用者電路板上時所能提供的功能，這些功能可用於進一步開發或者批量生產的產品。

9.1 Kendryte K210

這款處理器在Seeed Technologies Co Ltd Maix BiT電路板上使用。它是一款功能強大的處理器，價格適中。

9.1.1 Kendryte K210處理器核心

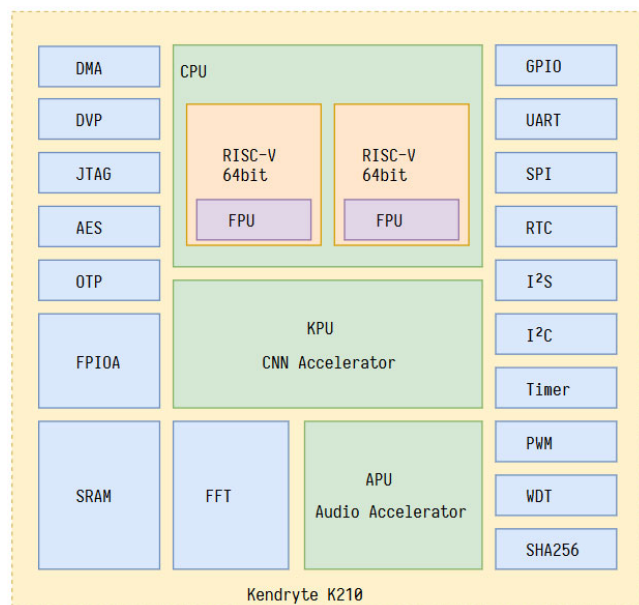


圖107：Kendryte K210架構概述。圖片來源：Kendryte K210資料工作表。

它包含兩個（而非一個）以400 MHz執行的RISC-V 64位元浮點處理器。此外，還有一系列有用的內部元件和介面。

縮寫	說明
CNN	神經網路
DMA	直接記憶體位址
DVP	數位影片連接埠
JTAG	程式設計和偵錯介面
AES	高級加密標準
OTP	一次性可程式化快閃記憶體
FPIOA	現場可程式化輸入和輸出陣列（可將GPIO映射到任何引腳）
SRAM	靜態隨機存取記憶體
FFT	快速傅立葉變換
GPIO	通用輸入/輸出3V3和1V8
UART	序列埠
SPI	序列週邊設備介面
RTC	即時時鐘
I2S	相互整合式音訊。音訊編解碼器的標準介面
I2C	用於低速週邊設備的積體電路間匯流排
PWM	脈衝寬度模組
WDT	看門狗定時器
SHA256	用於驗證資料完整性的安全雜湊演算法

表8：Kendryte K210縮寫

使GPIO成為雙路3V3/1V8非常有用。某些無線電模組（例如LTE/GSM模組）僅支援1V8操作。如果沒有雙電壓GPIO，將需要額外的電平轉換器，但這需要額外的成本。

現場可程式化輸入/輸出陣列（Field Programmable Input/Output Array，FPIOA）使印刷電路板（Printed Circuit Board，PCB）的佈局更加簡單，因為可以對引腳進行分組來簡化佈線。

Kendryte K210處理器僅提供球柵陣列（Ball Grid Array，BGA）封裝，這使其只適用於機器佈局。

9.1.2 Seeed Technologies MaixPy BiT實作

此外，對於實驗或小批量生產，基本的MaixPy BiT可以輕鬆安裝到麵包板上，或者使用安裝引腳焊接到另一個PCB上。

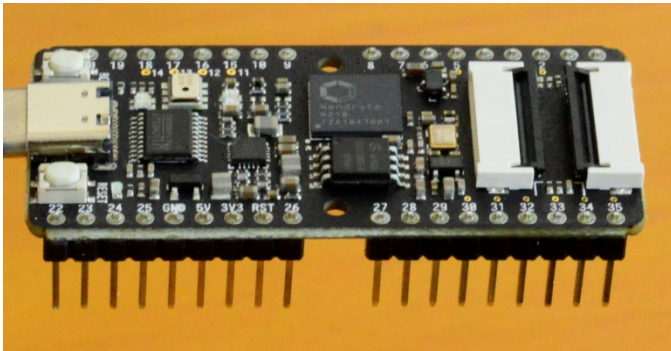


圖108：帶安裝引腳的Maix BiT電路板。Seeed Technologies Maix-I模組

9.1.3 Maix-I

在PCB上實作任意處理器時的一個問題是，要使處理器正常工作，需要很多附加元件 – 去耦電容、晶振、直流-直流轉換器、電感和重設電路等。

為了減輕硬體工程師的工作負擔，提供了兩個Maix-I模組：

[MAIX-I](#) Digi-Key 1597-1717-ND（8.91美元）和

[Maix-I with Wi-Fi](#) Digi-Key 1597-1715-ND（10.82美元）。

對於小批量生產或早期開發板，這兩個模組極具吸引力。

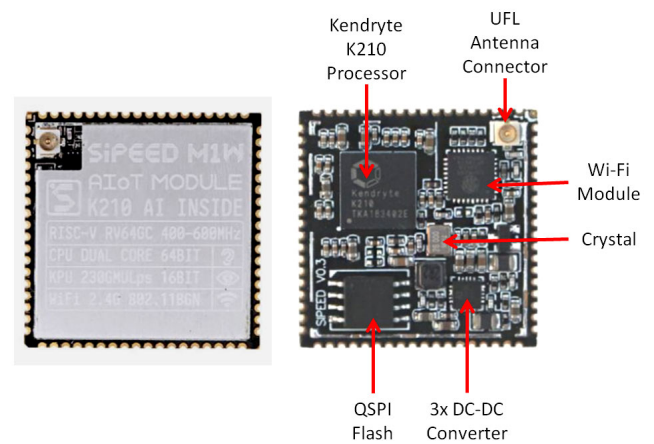


圖109：帶Wi-Fi的Maix-I模組。圖片來源：Seeed Technologies Co Ltd（已修改）。

兩個Maix-I模組不僅包含Kendryte K210處理器，還包含四路SPI快閃記憶體、晶振、直流-直流轉換器以及選購的Wi-Fi模組（價格稍高）。

可能的應用是由主處理器通過一個簡單的序列埠驅動的專用圖形處理器。由於可以使用MicroPython開發專案，因此軟體開發時間會很短。

9.1.4 對MaixPy BiT的Visual Studio Code支援

MaixPy BiT使用MicroPython進行程式設計，但也可以選擇C或C++。

此外，Xilinx FPGA軟體核心使用的PlatformIO還支援MaixPy BiT電路板和Maix-I模組上使用的Kendryte K210。但是，作者還未嘗試過。

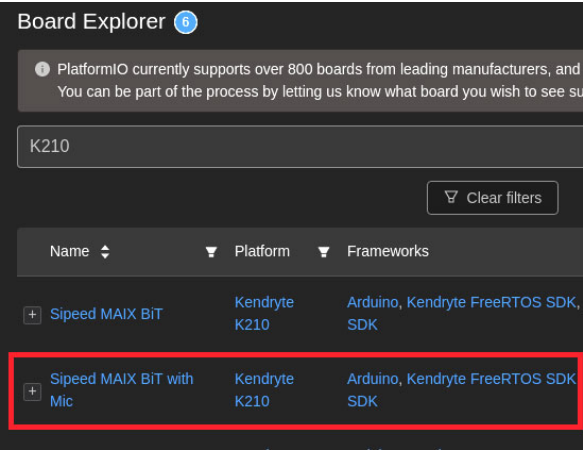


圖110：對MaixPy BiT的VSC PlatformIO支援

9.2 SiFive

RED-V Redboard 上使用的處理器是 SiFive Freedom Everywhere FE310-G002。

9.2.1 Freedom Everywhere FE310-G002 RISC-V 引腳排列

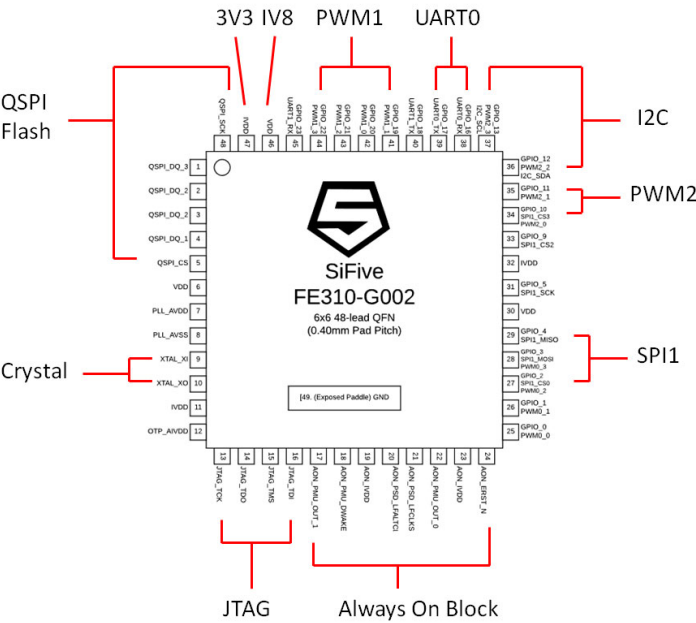


圖111：SiFive FE310-G002引腳排列。圖片來源：SiFive FE310-G002資料工作表（已修改）。

FE310-G002採用48引腳QFN封裝，執行頻率最高為320 MHz。

9.2.2 FE310-G002 GPIO引腳多工功能

Name	Pin	GPIO	PWM	SPI	UART	I2C
GPIO_0	25	0 I/O	PWM0_0 O			
GPIO_1	26	1 I/O	PWM0_1 O			
GPIO_2	27	2 I/O	PWM0_2 O	SPI1_SS0		
GPIO_3	28	3 I/O	PWM0_3 O	SPI1_MOSI		
GPIO_4	29	4 I/O		SPI1_MISO		
GPIO_5	31	5 I/O		SPI1_SCK		
GPIO_9	33	9 I/O		SPI1_SS2		
GPIO_10	34	10 I/O	PWM2_0 O	SPI1_SS3		
GPIO_11	35	11 I/O	PWM2_1 O			
GPIO_12	36	12 I/O	PWM2_2 O			I2C0_SDA
GPIO_13	37	13 I/O	PWM2_3 O			I2C0_SCL
GPIO_16	38	16 I/O			UART0_RX I	
GPIO_17	39	17 I/O			UART0_TX O	
GPIO_18	40	18 I/O			UART1_TX O	
GPIO_19	41	19 I/O	PWM1_1 O			
GPIO_20	42	20 I/O	PWM1_0 O			
GPIO_21	43	21 I/O	PWM1_2 O			
GPIO_22	44	22 I/O	PWM1_3 O			
GPIO_23	45	23 I/O			UART1_RX I	

表9：SiFive連接埠引腳輔助功能。圖片來源：SiFive FE310-G002資料工作表。

GPIO引腳是多工的，這意味著最多可以有3個PWM、1個SPI、1個I2C和2個UART。請注意，沒有類比至數位轉換器（Analog to Digital Converter，ADC）。由於它採用QFN48封裝，難以手工焊接（或脫焊），因此最適合機器佈局。

9.2.3 SparkFun Electronics RED-V SiFive Thing Plus實作

對於實驗或小批量生產，也可以使用RED-V SIFIVE RISC-V THING PLUS。1568-DEV-15799-ND 的成本為29.95美元。需要增加連接引腳。SparkFun網站上提供了原理圖。



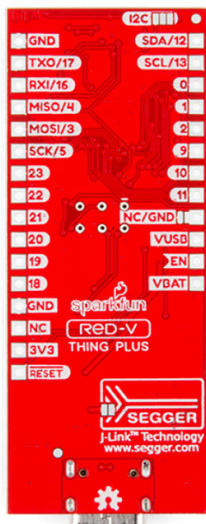


圖112：SparkFun Electronics SiFive Thing Plus仰視圖。圖片來源：SparkFun Electronics

10 RISC-V的詳細資訊

有關RISC-V的詳細資訊，請參閱YouTube和RISC-V.org網站上的RISC-V峰會議程。RISC-V的發明者David Patterson和Krste Asanovic都是幽默的演講者。YouTube上還提供了Maix BiT、SparkFun和Digilent电路板的影片。

對於那些希望在本文介紹的FPGA中執行SweRV核心後鞏固並獲得更多知識的讀者，請關注將於2020年11月發佈的Imagination Technologies「RVfpga：瞭解電腦架構的完整課程」。它由20個實驗組成，分為4個部分，本指南涵蓋了第1部分的一些內容：

第1部分：Vivado專案和程式設計 – Vivado和Verilator、C程式設計、Whisper、RISC-V應用程式二進位介面（Application Binary Interface，ABI）、程序呼叫慣例、合併C和組合語言程式碼。

第2部分：I/O系統 – 驅動Digilent Nexys A7電路板上的7段顯示器、中斷、定時器和序列匯流排（SPI、I2C和UART）。

後續課程將於2021年第3季度發佈：

第3部分：RISC-V核心 – 瞭解核心結構、管線、風險和實作新指令

第4部分：RISC-V記憶體系統 – 瞭解快取控制器、快取命中和未命中、修改快取，瞭解ICCM（指令緊密耦合記憶體）和DCCM（資料緊密耦合記憶體）。

有關RISC-V的兩本相關參考書的詳細資訊，請另行參閱「參考資料」一節。

11 結論

對比RISC-V的不同電路板後，MaixPy BiT被證明最容易使用且充滿最多樂趣。有意思的是，成本最低的電路板（帶攝影頭和LCD）能夠以雙核RISC-V處理器的形式提供最強大的計算能力，而價格卻非常適中。它是學習機器視覺和面部識別的理想工具。由於它使用MicroPython程式設計，因此即使是沒有程式設計背景的人也可輕鬆上手。

SparkFun Electronics RED-V Thing Plus和Red Board非常適合業余愛好者，或者希望學習如何使用C語言或組合語言進行RISC-V程式設計，以便自行開發產品的電子/軟體工程師。SiFive的軟體工具易於使用，其中有許多有用的範例，經過修改便可用於其他應用。SiFive的文件也很易於閱讀。適合ARM或TI處理器的Eclipse工具的使用者對此應該不會感到困難。

在本RISC-V指南中，只能簡要介紹如何在Xilinx FPGA中實作軟體核心。本指南面向的是學習電腦架構的學生和從事大型專案的工程師。這是一項複雜的業務，建議不要在沒有詳細說明的情況下進行。即使以前從未使用過軟體核心，但只要認真遵循Imagination Technologies「RVfpga：入門指南」並使用RVfpga套件，便可於一天內在Nexys A7電路板上成功執行RISC-V核心。在四款電路板中，這是迄今為止技術性最強的一種。對於想要發揮聰明才智的讀者，可以參加後續活動Imagination Technologies「RVfpga：瞭解電腦架構的完整課程」。

我們在RISC-V上投入了大量精力，唯有時間才能證明它是ARM的有力競爭對手。開放原始碼ISA且沒有權利金無疑為RISC-V帶來了競爭優勢，因為它能夠提供一些非常強大的低成本處理器。

12 致謝

非常感謝馬德里大學的Daniel chaver Martinez教授允許作者使用他的RISC-V教學資料，還要感謝教授的學生Daniel Gonzalez允許作者使用他的碩士論文。沒有上述授權，將無法編寫本指南。

最後，特別感謝Imagination Technologies的Robert Owen構思了本指南並最終完稿出版。

13 作者簡介

Richard Sikora擔任硬體和軟體工程師已逾30年，從事過30多種不同處理器和數位訊號處理器（Digital Signal Processors，DSP）的相關工作。多年來，他設計了稱重機、製程控制儀器、飛機點火裝置、自動販賣機、智慧卡讀卡機，以及煙霧和一氧化碳監視器等設備。此外，他還為Texas Instruments DSP和Matlab編製過多個教學工具套件。



14 參考資料

嵌入式微處理器基準聯盟

<https://www.eembc.org/coremark/>

RISC-V標誌<https://riscv.org/RISC-V-branding-guidelines-and-materials/>

RISC-V規格<https://riscv.org/technical/specifications/>

其他核心<https://chipsalliance.org>

Kendryte K210資料工作表https://s3.cn-north-1.amazonaws.com.cn/dl.kendryte.com/documents/kendryte_datasheet_20181011163248_en.pdf

Wishbone匯流排

<http://indico.ictp.it/event/a11204/session/35/contribution/22/material/0/0.pdf>

<https://venturebeat.com/2019/12/11/risc-v-grows-globally-as-an-alternative-to-arm-and-its-license-fees/>

SweRV核心下載位址：

<https://github.com/chipsalliance/Cores-SweRV>

SweRV程式設計師參考手冊

https://github.com/chipsalliance/Cores-SweRV/blob/master/docs/RISC-V_SweRV_EH1_PRM.pdf

SweRVolf：

<https://github.com/chipsalliance/Cores-SweRVolf>

PlatformIO：

<https://github.com/platformio/platformio-core>

Verilator：

<https://github.com/verilator/verilator>

Whisper：

<https://github.com/westerndigitalcorporation/swerv-ISS>

DANIEL LEÓN GONZÁLEZ，《適合工業物聯網的專用RISC-V晶片系統的FPGA實作》，碩士學位論文，馬德里康普頓斯大學，2020年7月。

數位設計和電腦架構，Sarah Harris和David Money Harris，第二版（RISC-V版本將於2021年發佈）
https://www.amazon.com/Digital-Design-Computer-Architecture-Harris/dp/0123944244/ref=sr_1_1?crid=1232QZSYQ20GW&dchild=1&keywords=digital+design+and+computer+architecture+2nd+edition&qid=1597397347&srefix=digital+design+and+%2Caps%2C219&sr=8-1

RISC-V Reader，David Patterson和Andrew Waterman
https://www.amazon.com/RISC-V-Reader-Open-Architecture-Atlas/dp/0999249118/ref=sr_1_3?dchild=1&keywords=Patterson+RISC-V&qid=1597397389&sr=8-3

