

Jini Intelligent Computing Workbook of Lab. #3

Preamble

Lab. #3 在 AlveoU50 的 Platform 專案目錄下有四個 Applications 專案目錄：

- vts_Opt1Baseline
- vts_Opt2KernelParallel
- vts_Opt3DataBurst
- vts_Opt4ArrayPartition

以上目錄中皆包含該專案的原始碼檔。

1. Introduction

本實驗為 Vitis OpenCL/XRT 實作，以 Xilinx Alveo U50 PCIe 加速卡為基礎。Xilinx Alveo U50 為 PCIe 介面之 FPGA 加速卡，以 Linux server 為平台透過建置 Xilinx XRT runtime 架構，再以 OpenCL 語言開發 host program，將 bitstream (.xclbin) 檔案下載至 Xilinx Alveo U50 加速卡，並運行 host program 的流程控制。

此外本實驗就算沒有 Alveo U50 加速卡，仍然可以在使用者 PC 做到 Software/Hardware Emulation，Hardware Emulation 的模擬結果與真實在 FPGA 運行相近。

Note：

因 Windows 版本的 Vitis 不支援 Alveo U50 等 PCIe 介面之 FPGA 加速卡，本次實驗將全部在 Linux 系統上實作，若無 Linux PC 亦可在 Windows 上以 Oracle VM VirtualBox 等 virtual machine 運行。採用 VM 請特別注意分配給 VM 的記憶體不要太少，建議分配 8 GB 以上，CPU 也可以多分配一些以加快模擬速度。

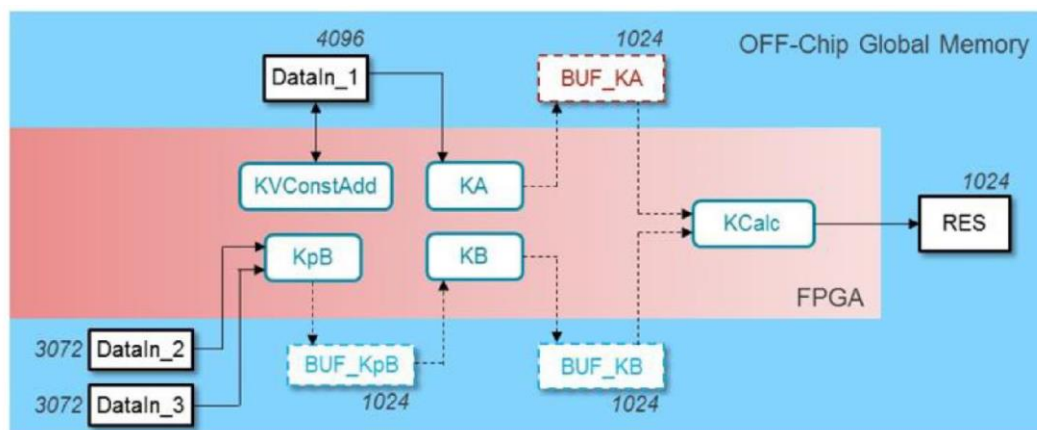
2. Vitis Application Acceleration

【施作環境為在使用者 PC/laptop/notebook (Linux Base) 。】

本實驗共有四個專案，分別對應 Baseline、Kernel Parallel、Data Burst 及 Array Partition 四種不同的組態。


四個組態的實驗步驟皆相同，請仔細比較各組態 **source code** 及產生的 **Application Timeline** 與 **Profile Summary** 的差異。下方步驟以 Baseline 作為範例。

下圖為本次實驗的架構，由五個 kernel function 及七個在 global memory 中的 data buffer 組成。



2.1. Create and Setup Project


登入先前註冊好的帳號，在 Terminal 輸入 vitis 開啟 Vitis 程式，並設定好運行的 Launch directory。

 Vitis IDE Launcher @HLS03 ×

Select a directory as workspace

Vitis IDE uses the workspace directory to store its preferences and development artifacts.

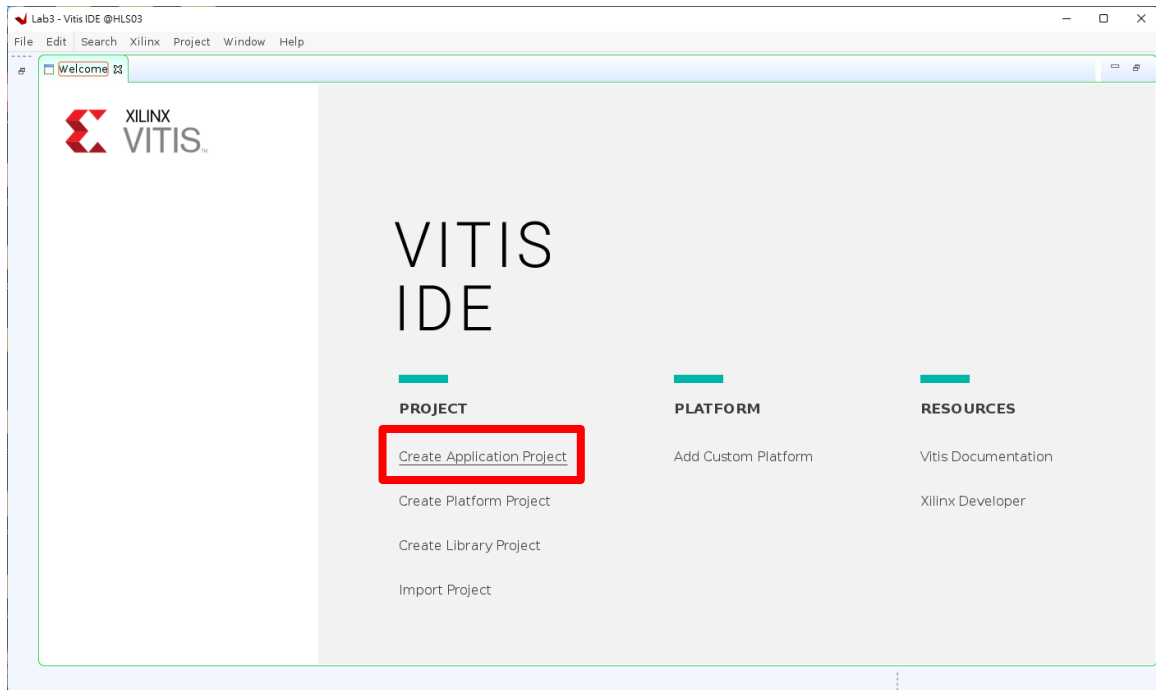
Workspace:



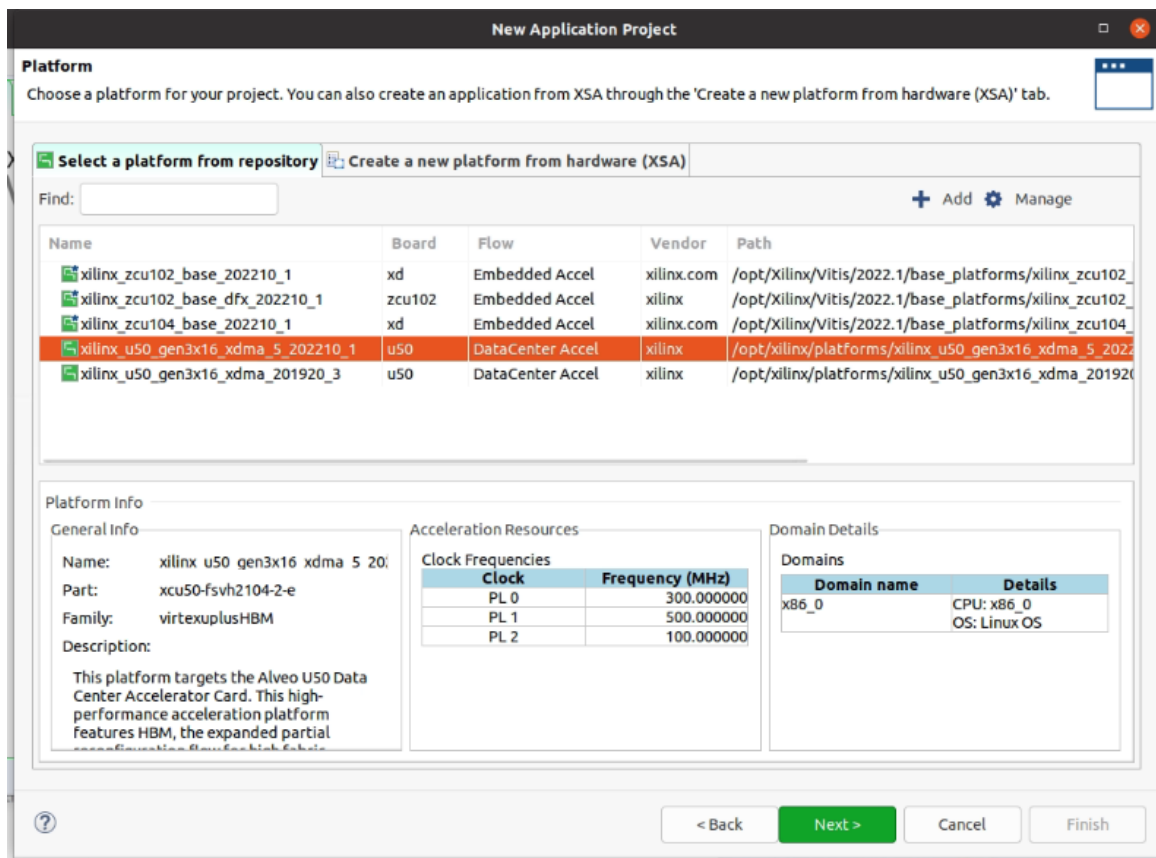
☐ Use this as the default and do not ask again

▶ Restore other Workspace

開啟主畫面後，點選 Create Application Project。



Platform 選擇 xilinx_u50_gen3x16_xdma_5_202210_1 (注意是選 2022 的喔)



為專案命名，下方 System project name 會自動填上毋須修改。

New Application Project @HLS03

Application Project Details

Specify the application project name and its system project properties

Application project name:

System Project

Create a new system project for the application or select an existing one from the workspace

Select a system project

+ Create new...

System project details

System project name:

Target processor

Select target processor for the Application project.

| Processor | Associated applications |
|-----------|-------------------------|
| x86 SMP | vts_Opt1Baseline |

Show all processors in the hardware specification ☐

? < Back Next > Cancel Finish

選擇 Empty Application，點選 Finish 建立專案。

New Application Project @HLS03

Templates

Select a template to create your project.

Available Templates:

Find:

Acceleration templates with PL and AIE accelerators

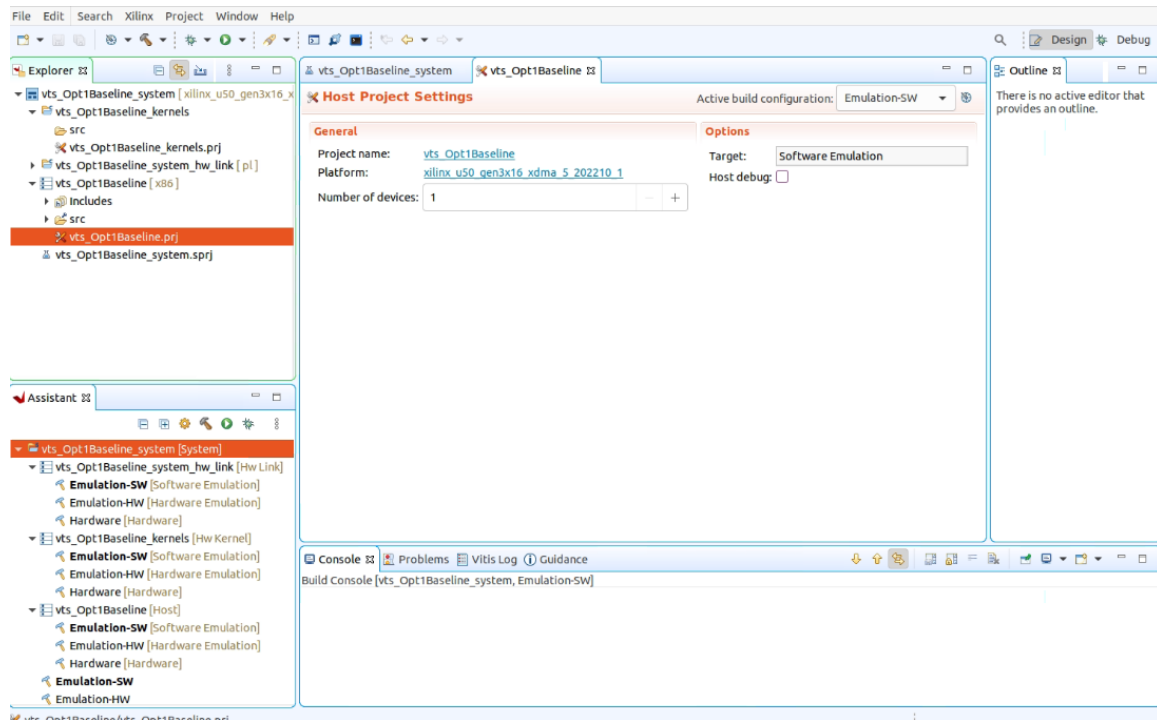
- Empty Application
- Empty Application (XRT Native API's)
- Simple Vector Addition

Empty Application

Creates a new Empty application

Vitis IDE Examples... Vitis IDE Libraries...

? < Back Next > Cancel Finish



左上方 Explorer 內可看到在 vts_Opt1Baseline_system 專案底下有三個專案：

1. vts_Opt1Baseline_kernels 專案負責 compile kernel function。
2. vts_Opt1Baseline_system_hw_link 專案負責將 kernel link 起來產生 bitstream file (.xclbin)。
3. vts_Opt1Baseline 專案負責 host program 的部分。

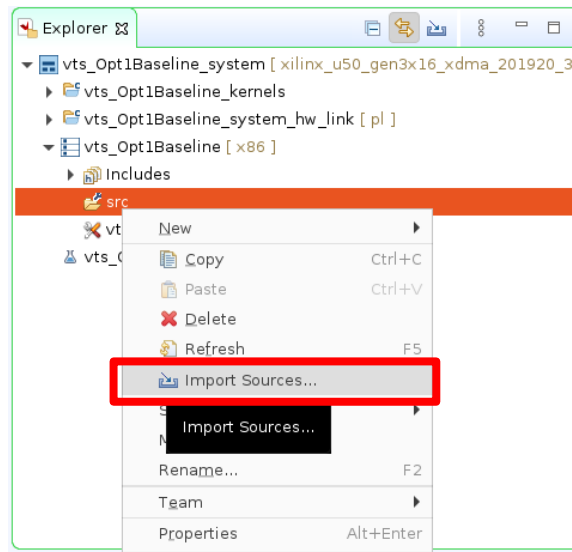
左下方 Assistant 內顯示了各個專案的建置和模擬狀態，以及各項工作產生的 report。

中間的 Project Editor 顯示專案部分屬性，且可以直接對各專案進行設定。

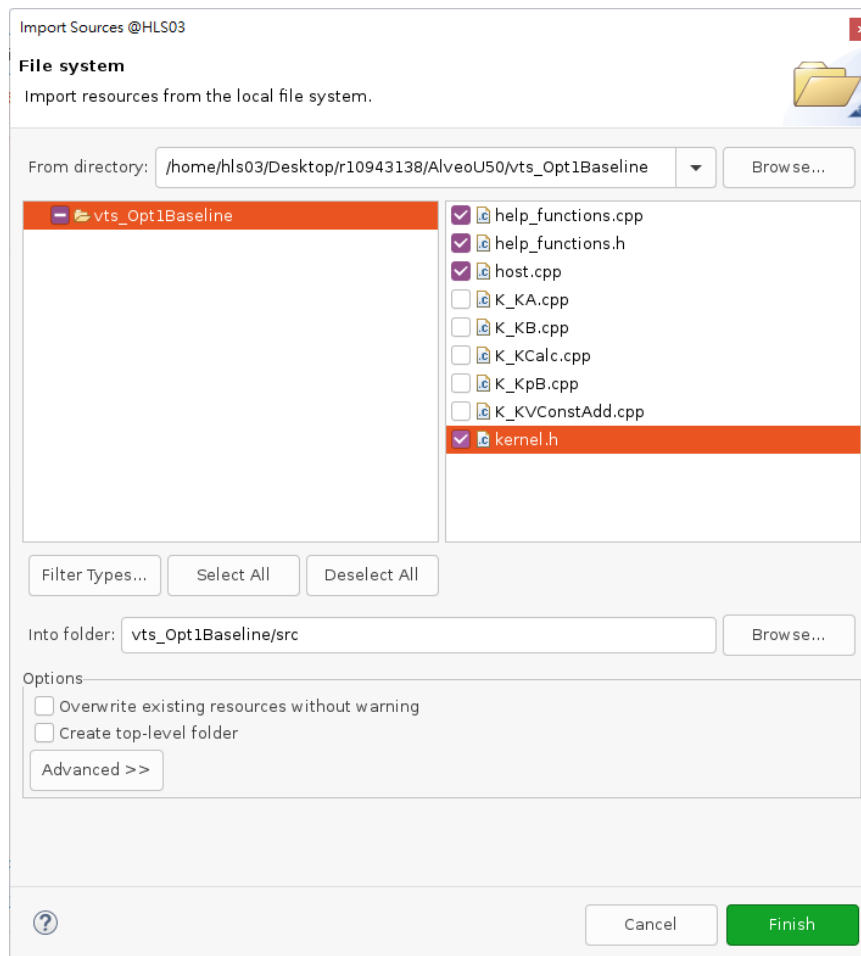
下方有 Console 顯示工作狀態，且可以在各專案不同組態的 console 間切換。

建立好專案後第一步要在專案裡加入 source code。

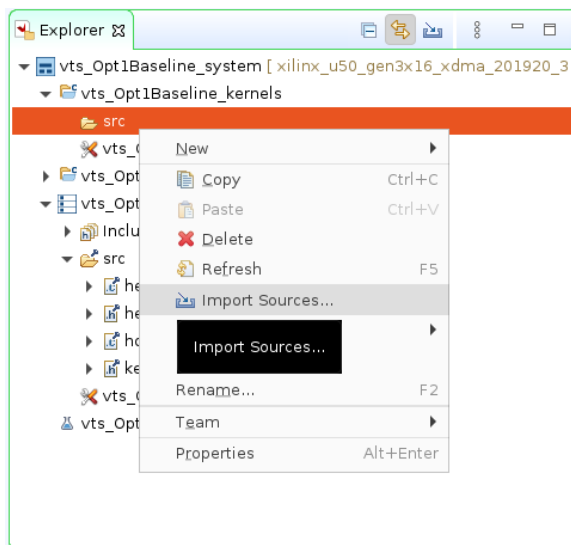
首先在 Explorer 中右鍵點選 vts_Opt1Baseline 專案底下的 src 資料夾，點選 Import Sources 加入 host program 的 source code。



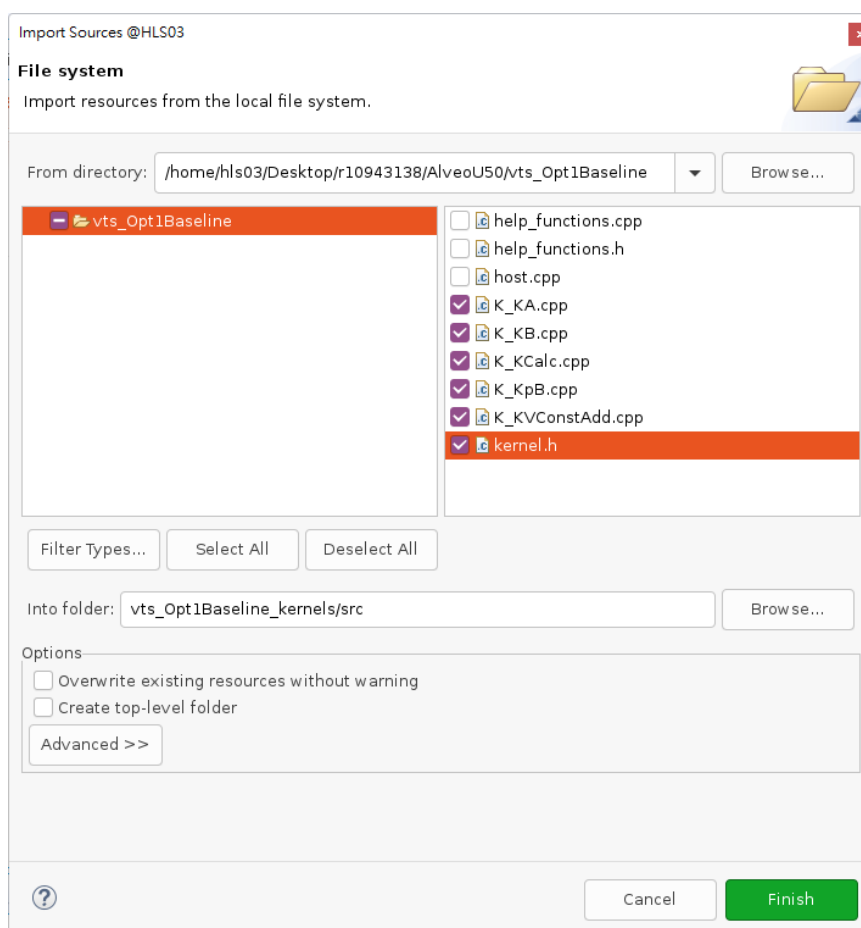
選擇提供的 source code 資料夾，勾選 **help_functions.cpp**、**help_functions.h**、**host.cpp** 以及 **kernel.h**。



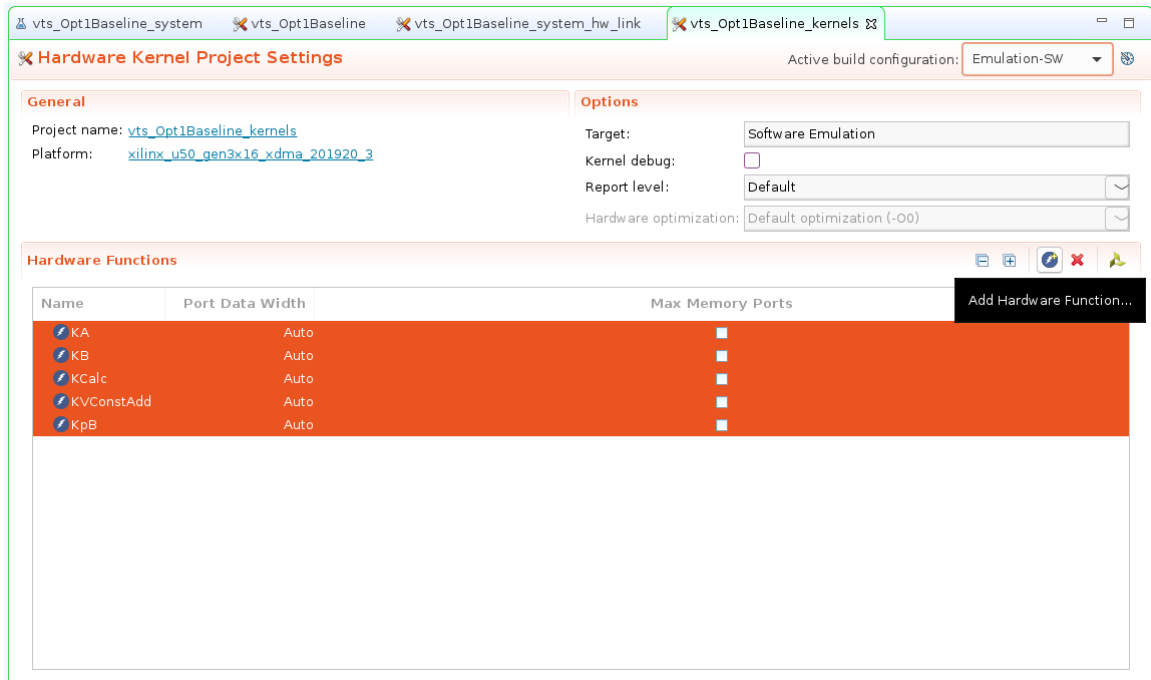
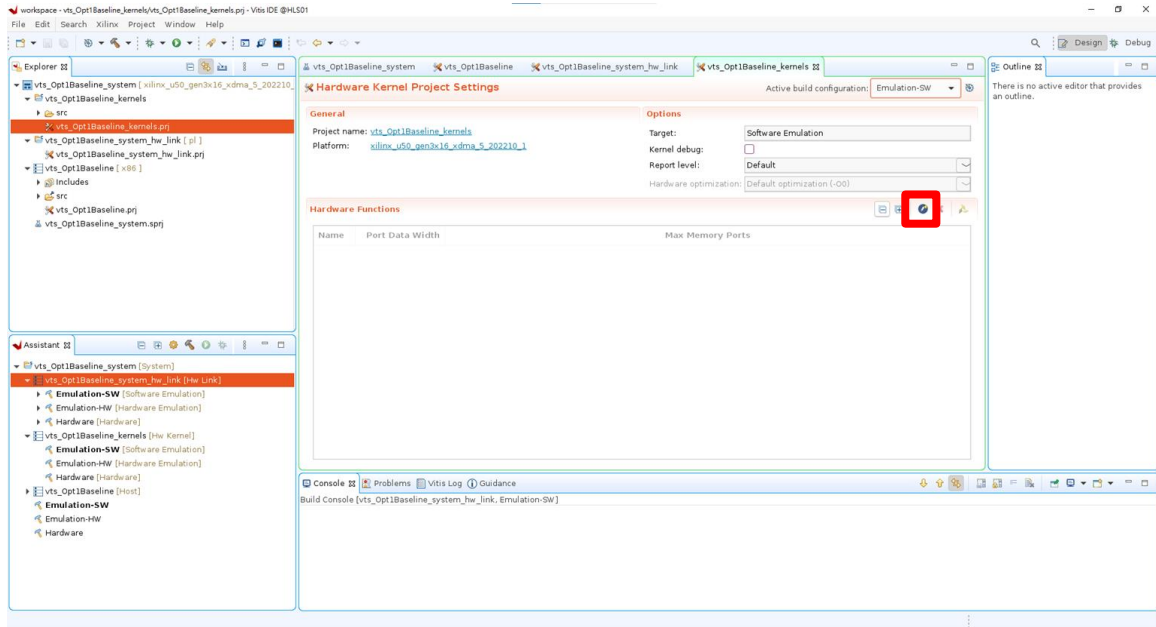
接著在 vts_Opt1Baseline_kernels 專案底下加入 kernel function 的 source code 。



選擇提供的 source code 資料夾，勾選 **K_KA.cpp**、**K_KB.cpp**、**K_Kcalc.cpp**、**K_KpB.cpp**、**K_KVConstAdd.cpp** 以及 **kernel.h** 。



最後加入 hardware functions，打開 vts_Opt1Baseline_kernels 專案 (vts_Opt1Baseline_kernels.prj)，在 Project Editor 點選 Add Hardware Functions，並在彈出的視窗中選擇 KA、KB、KCalc、KVConstAdd 以及 KpB 加入。完成後底下會列出所有 kernel functions。



2.2. Software Emulation

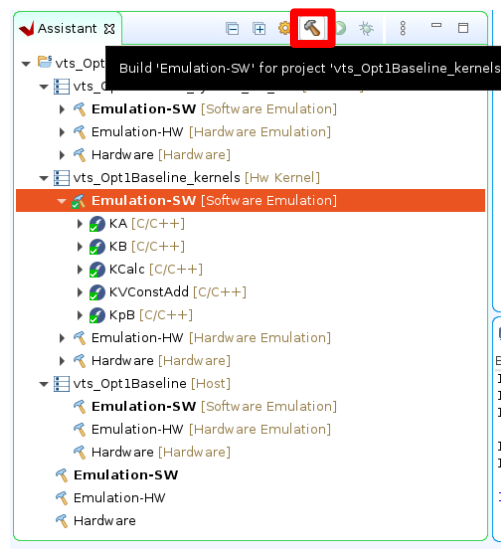
Software emulation 是以軟體函式形式直接傳遞引數來模擬結果，類似於 Lab. #1 中的 C simulation。

2.2.1. Build Project

要執行 emulation 前要先建置專案，產生模擬需要的執行檔及 bitstream file。

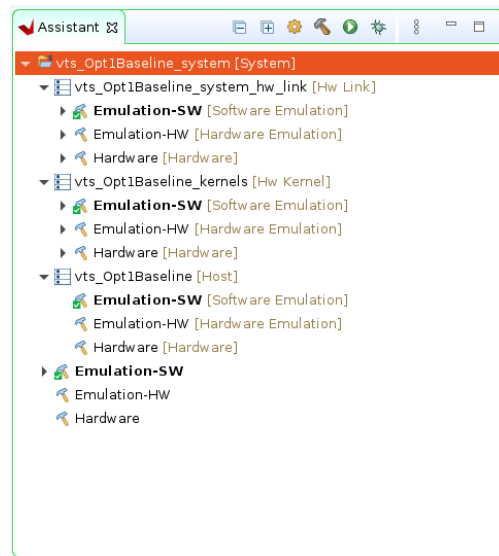
必須依照 kernel → hw_link → host → system 的順序來建置專案！

首先在 Assistant 中選擇 kernel 專案底下的 Emulation-SW，接著按下上方 Build project 按鈕進行建置。



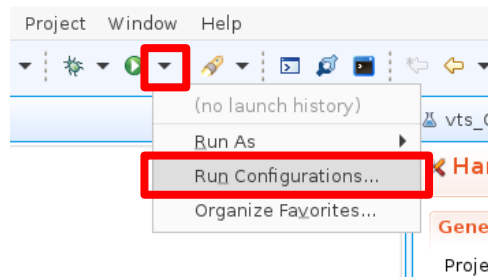
kernel 建置完成後再依前述順序點選其他專案的 Emulation-SW 並進行建置，system 的部分請直接點選 vts_Opt1Baseline_system[System]來建置。

完成建置後 Assistant View 顯示如下，建置成功會有綠色打勾標示。

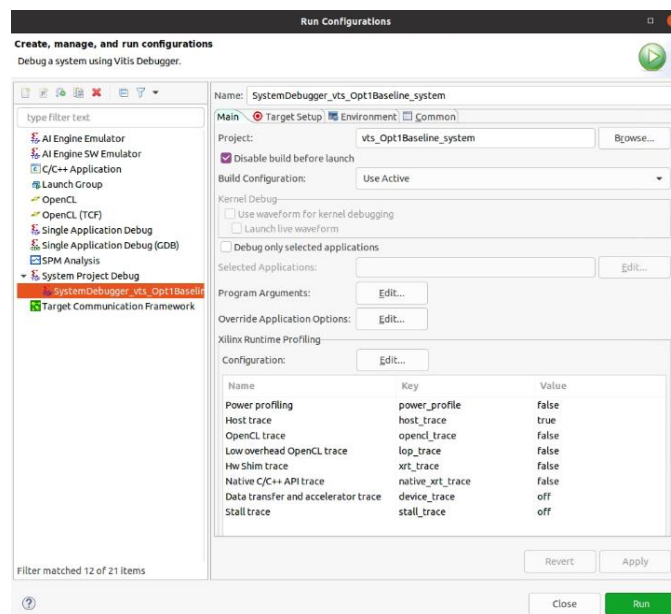
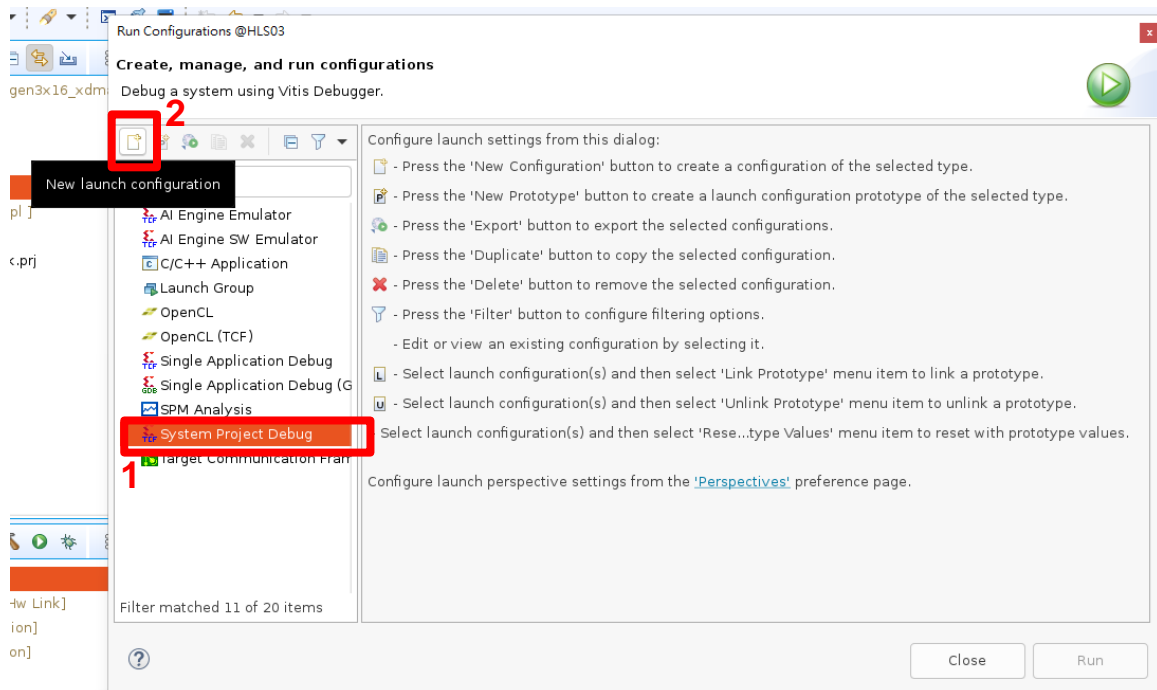


2.2.2. Run Emulation

執行 emulation 前要先設定其組態。點選 Run Configurations



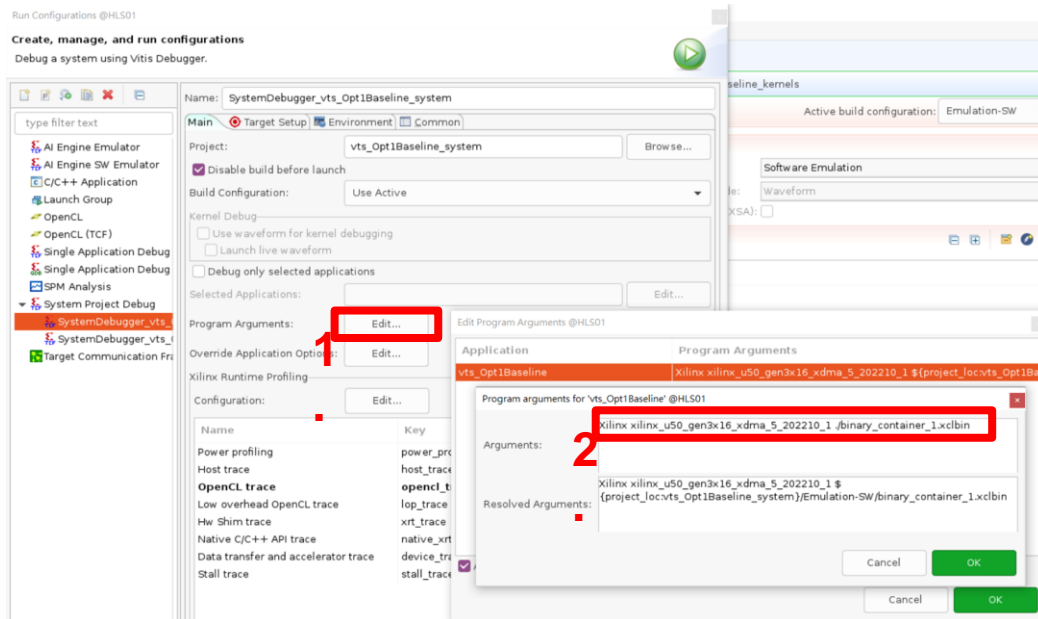
新增一個 System Project Debug 組態。



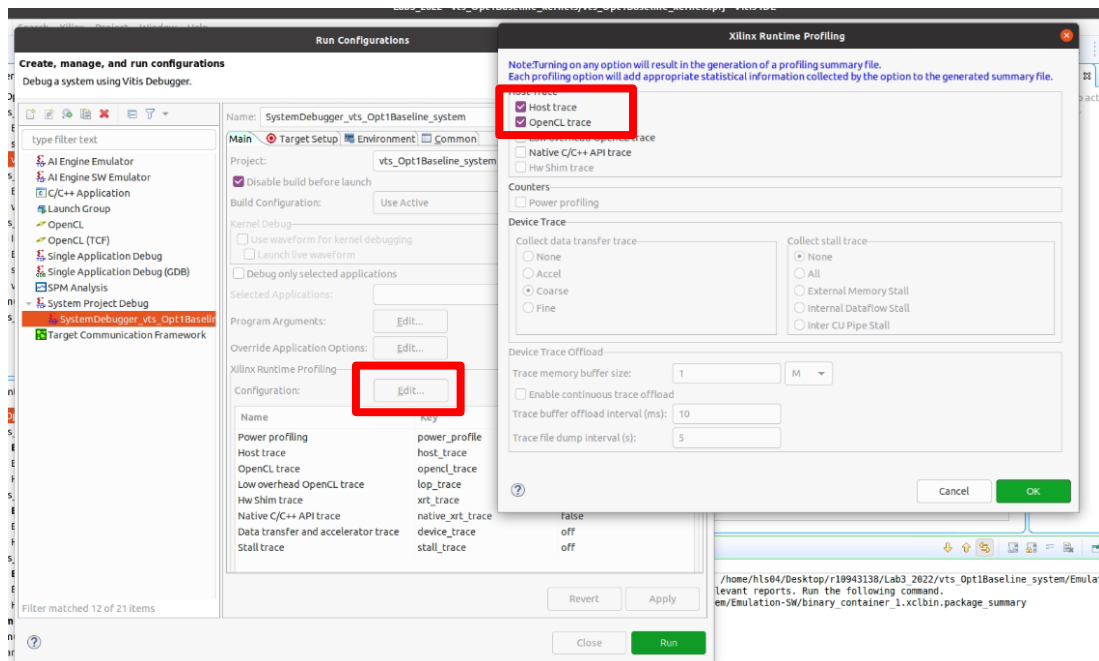
編輯 Program Arguments，提供 host program 所需的三個 arguments：

Xilinx xilinx_u50_gen3x16_xdma_5_202210_1 ./binary_container_1.xclbin

(用空白隔開這三項)

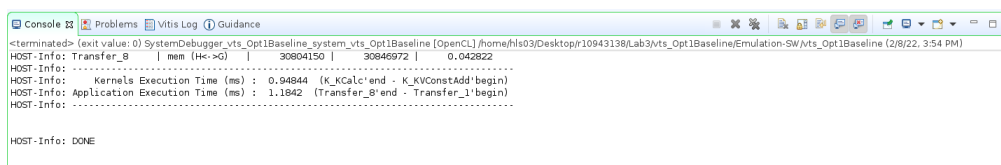


編輯 Xilinx Runtime Profiling 的 Augments，勾選 Host trace 跟 OpenCL trace。



點選 OK，再點選 Run 開始執行 software emulation。

Emulation 完成後，會在 Console 看到 DONE 訊息。



2.2.3. Analysis

2.3. Hardware Emulation

Hardware emulation 是以軟體模擬 XRT runtime 到 kernel FPGA 的行為，類似於 Lab. #1 中的 Co-simulation。

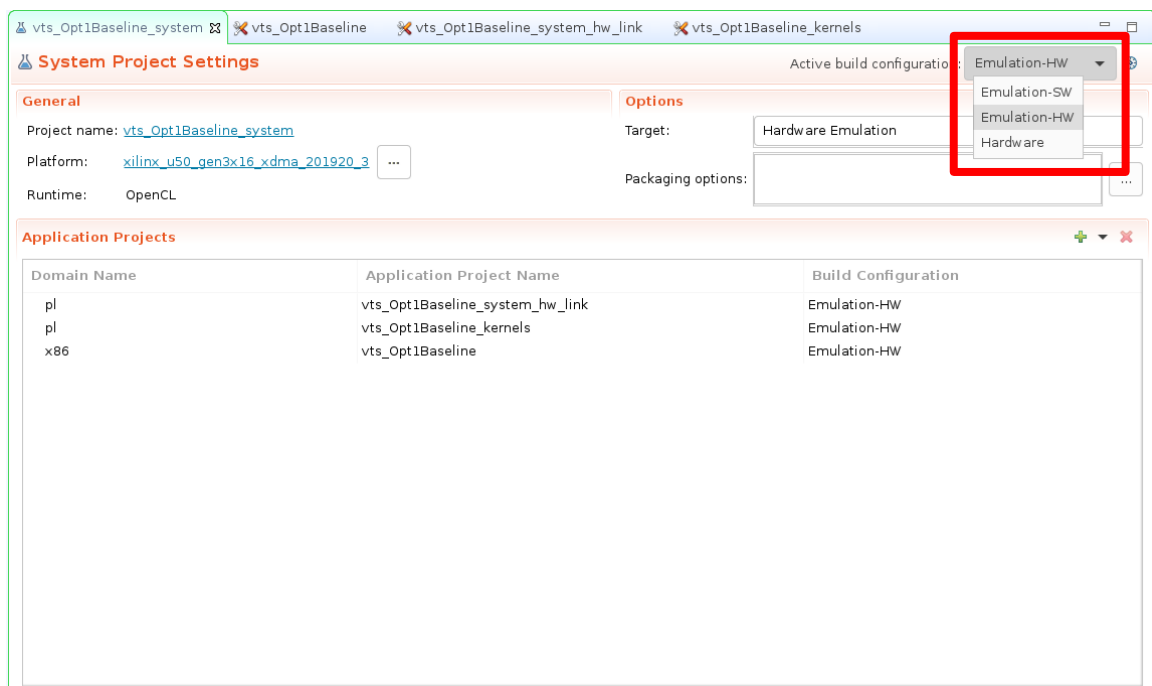
2.3.1. Build Project

步驟同 Software Emulation，請參考前述步驟，改為點選 Emulation-HW。過程視電腦配備可能需要數十分鐘。

(kernel → hw_link → host → system)

2.3.2. Run Emulation

請將 Active build configuration 設定為 Emulation-HW，即可直接使用相同的 Run Configuration 毋須修改，其餘步驟與 Software Emulation 相同，請參考前述步驟。過程視電腦配備可能需要數十分鐘。





2.4. Hardware

組態 Hardware 產生可在 Alveo 加速卡運行的程式 (host program) 及 FPGA 的 bitstream file (.xclbin)。

本實作 Linux server 上已建置 Xilinx Alveo U50 加速卡，即可將 Vitis 建置好的 host program 及 bitstream 直接在伺服器中運行。

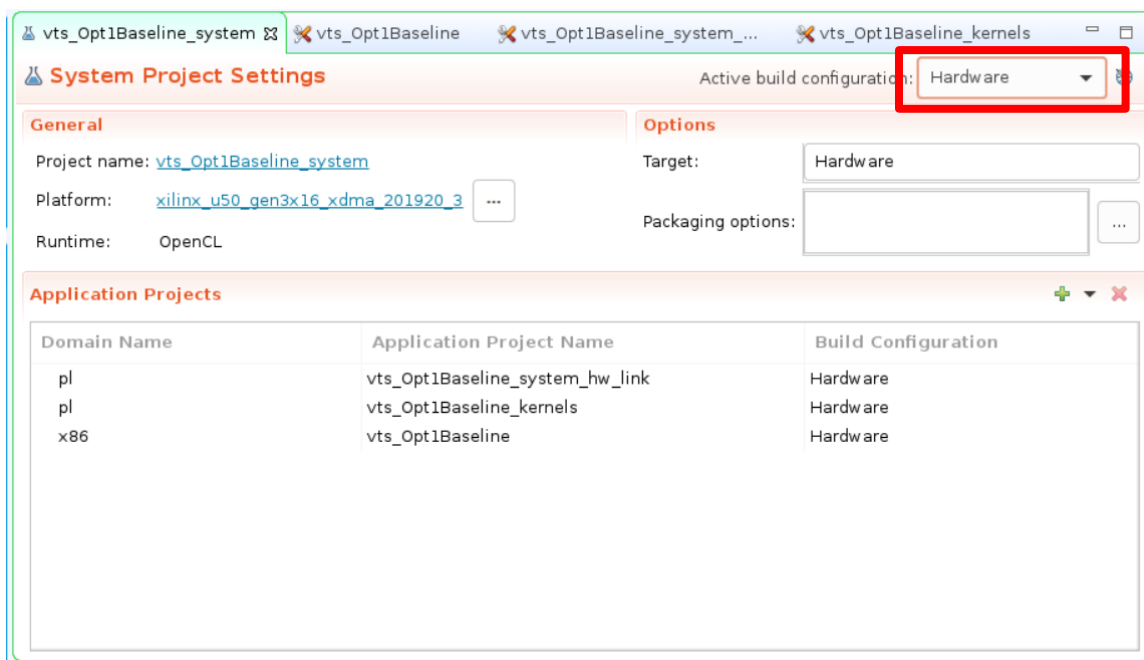
2.4.1. Build Project

步驟同 Software Emulation，請參考前述步驟，改為點選 Hardware。過程視電腦配備可能需要數十分鐘。(又比 HW emulation 更久)

(kernel → hw_link → host → system)

2.4.2. Run Hardware

請將 Active build configuration 設定為 Hardware，即可直接使用相同的 Run Configuration 毋須修改，其餘步驟與 Software Emulation 相同，請參考前述步驟。



2.4.3. Analysis

步驟同 Software Emulation，請參考前述步驟，並更改 Run Configurations。

編輯 Program Arguments，提供 host program 所需的三個 arguments：

Xilinx xilinx_u50_gen3x16_xdma_base_5 ./binary_container_1.xclbin

(用空白隔開這三項)

打開在 Hardware 中的 Run Summary 查看各項 report 進行分析。

