

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. Installation

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

2.1. Vitis/Vivado/Vitis HLS Software Installation

首先至 Xilinx 官方下載頁面下載 Linux 版本安裝檔: Xilinx Unified Installer 2021.2: Linux Self Extracting Web Installer。

連結網址: <u>https://www.xilinx.com/support/download.html</u>

Xilinx Unified Installer 2021.2: Linux Self Extracting Web Installer (BIN - 272.8 MB)

MD5 SUM Value : d4fe2978f735e4353f6ccff3405b488b

Download Verification 🕦



下載完成後,請於檔案資料夾輸入以下指令([filename]為剛下載好的 安裝檔)來將安裝檔更改權限為可執行:

\$> chmod +x [filename]

接著執行安裝檔。

\$> sudo ./[filename]

安裝過程可參考 Tools Installation Guide 内Windows 版本安裝說明,另外

一樣須注意安裝過程需要約 240 GB 的硬碟空間。



Select Destination Directory	
Choose installation options such as location and shortcuts.	🕹 ÄILIINA:
Installation Options Select the installation directory	Select shortcut and file association options
/opt/Xilinx	Xilinx Design Tools
Installation location(s)	Create desktop shortcuts

安裝完成後輸入以下指令開啟 Vivado License Manager(/opt/Xilinx/

為安裝路徑):

\$> cd /opt/Xilinx/Vitis/2021.2/bin

\$> sudo ./vlm

接著同Tools Installation Guide 中說明匯入下載的 license,若點選 Connect Now

沒有反應請直接上 Xilinx 官網下載 license。

連結網址:

https://www.xilinx.com/support/licensing_solution_center.html

進入後點選 Xillinx Product Licensing Site 並登入下載 license。

匯入成功會看到以下通知。





2.2. U50 Platform/XRT Installation

要在Vitis 中使用U50, 須安裝U50 platform, 請至以下連結下載 package: <u>https://www.xilinx.com/products/boards-and-</u> kits/alveo/u50.html#gettingStarted首先選擇使用的 tools 及OS 版本。

Select your options to obtain the matching download files					
Alveo U50 Accelerator Card					
Tools Version	2021.2 2021.1 2020.2.1 Archive				
Platform Type	Gen3X16-XDMA				
Architecture	x86_64				
Operating System	RHEL/CentOS Ubuntu				
OS Version	18.04 20.04				

下載下方顯示的三個檔案,解壓縮後一一執行安裝,可用以下指令進行 安裝:

\$> sudo apt install [filename]

1. Download the Xilinx Runtime

The Xilinx runtime (XRT) is a low level communication layer (APIs and drivers) between the host and the card.

xrt_202120.2.12	2.427_18.04-amd64-xrt.(deb (13.05 MB)	*
DIGEST 🕹	SIGNATURE 🕹	PUBLIC KEY	公

2. Download the Deployment Target Platform

The deployment target platform is the communication layer physically implemented and flashed into the card.

xilinx-u50_202	1.2_2021_1021_1001-all	.deb.tar.gz (18.48 MB) 🛛 📩
DIGEST 🖧	SIGNATURE 🕹	PUBLIC KEY 🕹

Develop Your Own Accelerated Applications

In addition to steps 1 and 2, follow steps 3 and 4 for development using the Vitis design flow.

3.	Download the Development Target Platform
	The development target platform is required if you are building your own

The development target platform is required if you are building your own applications. Available Alveo Platforms





2.3. Bash Shell Setting

在Vitis 運行的Linux server 上需要設定環境 bash shell, 在個人帳戶

下.bashrc 檔加入環境設定 (nano ~/.bashrc)

#source Xilinx Vitis/XRT
source /opt/Xilinx/Vitis/2021.2/settings64.sh
source /opt/xilinx/xrt/setup.sh

設定完成後重新開啟 Terminal, 確定有順利 source 到。



2.4. Install Additional Packages

建置專案時需要有 opencl-headers 以及 gcc-multilib 套件, 輸入以下指令安裝:

\$> sudo apt install opencl-headers

\$> sudo apt install gcc-multilib



3. 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。



3.1. 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 Laur	ncher @HLS03	×
Select a dir Vitis IDE user	ectory as workspace s the workspace directory to store its preferences and development artifacts.	
<u>W</u> orkspace:	/home/hls03/Desktop/r10943138/Lab3	
Use this a	as the default and do not ask again t her Workspace	
	Cancel	



v 1	ab3 - Vitis IDE @H	ILS03			-		×
File	Edit Search	Xilinx Project Window Help					
8	🗖 Welcome 🛿	3				-	8
	3	XILINX VITIS					
			VITIS				
			IDE				
					_		
			PROJECT	PLATFORM	RESOURCES		
			Create Application Project	Add Custom Platform	Vitis Documentation		
			Create Platform Project		Xilinx Developer		
			Create Library Project				
			Import Project				

開啟主畫面後, 點選 Create Application Project。

選擇先前安裝好的 U50 platform。

A)' tab.				-		•	
Select a platf	form from repository	Cre	eate a new platform	from hardwa	re (XS	<i>4</i>)	
nd:						+ /	Add 🍄 Manage
Name		Board	Flow	Vendor	Path		
🔄 xilinx_u50_g	gen3x16_xdma_20192 ເ	J50	DataCenter Accel	xilinx	/opt/xi	linx/platforms/xilinx_	_u50_gen3x16_xdma_2
letferm lefe							
latform Info General Info Name: xi	ilinx_u50_gen3x16_xdn	na_2	cceleration Resources- Clock Frequencies			oomain Details Domains	
latform Info General Info Name: xi Part: xi	ilinx_u50_gen3x16_xdn cu50-fsvh2104-2-e	ha_2	cceleration Resources- Clock Frequencies Clock F	Frequency (MH	iz)	oomain Details Domains Domain name	Details
latform Info General Info Name: xi Part: xi Family: vi	ilinx_u50_gen3x16_xdn cu50-fsvh2104-2-e irtexuplusHBM	na_2	cceleration Resources- Clock Frequencies Clock F PL 0 PL 1	-requency (MF 300.000 500.000	iz) 000 000	Domain Details Domains Domain name x86_0	Details CPU: x86_0 OS: Linux OS
latform Info General Info Name: xi Part: xi Family: vi Description:	ilinx_u50_gen3x16_xdn cu50-fsvh2104-2-e irtexuplusHBM	na_ź	cceleration Resources Clock Frequencies PL 0 PL 1 PL 2	Frequency (MF 300.000 500.000 50.000	Iz) 000 000	oomain Details Domains <mark>Domain name</mark> x86_0	Details CPU: x86_0 OS: Linux OS



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

r Application Project @HLS03 Dlication Project Details acify the application project name and its	s system project properties
pplication project name: vts_Opt1Basel	ine
Create a new system project for the ap	olication or select an existing one from the workspace 🛛 👔
Select a system project	System project details
Create new	System project name: vts_Opt1Baseline_system
	Target processor
	Select target processor for the Application project.
	Processor Associated applications
	x86 SMP vts_Opt1Baseline
	Show all processors in the hardware specification

選擇 Empty Application, 點選 Finish 建立專案。

New Application Project @HLS03					x
Templates					•••
Select a template to create your project.					
Available Templates:					
Find:		Empty Applicatio	on		
	s	Creates a new Emp	ty application		
Empty Application					
Empty Application (XRT Native API's) Simple Vector Addition					
Vitis IDE Examples					
?		< Back	Next >	Cancel	Finish



🗂 • 🗟 🕲 • 🗞 • 🕸 • O • 🛷 • 🖾 🖉 🔳	6 6 • 0 •				٩	Design 🏘 D
Explorer 🕱 🔤 🗄 🕴 🗖 🗖	🛎 vts_Opt1Baseline_system	St vts_Opt1Baseline ₽		- 0	BE Outline 1	-
<pre>bolers II</pre>	2 vtopt1Baseline.system X Host Project Settinn General Project mark: vts_Opt Plaform: sline Number of devices: 1	St vit. Optilaseline II 18seline 50.genixid_sima_201920_1 - +	Options Target: Host debug:	Active build configuration: Emulation:SV Software Emulation	Be Outline to	tive editor that provid
Austrant X B R & O P I P P P Contrasting system (% O P) Contrasting system (%	Console II Problems B Build Console (vts_opt18asein]Vitis Log () Guidance _gyttem_thw_link.Emulation-SW)		\$ \$	J 51 = 9k	₫ ₩ • ⊡ • =

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。
- 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。

🗲 Explorer 🕱			è	000		
	Baseline_system [xilinx pt1Baseline_kernels pt1Baseline_system_hw_	_u50_gen3x: link [pl]	16_x	dma_:	2019	20_3
Vis_0 Incl	udes					
💕 src						
% ∨t	New	•				
≚ vts_0	🛅 Сору	Ctrl+C				
	💼 Paste	Ctrl+V				
	💢 Delete					
L _	🔄 Re <u>f</u> resh	F5				
	🚵 Import Sources					
	S Import Sources N	ľ	T			
	Rena <u>m</u> e	F2				
	T <u>e</u> am	Þ				
l	Properties	Alt+Enter				

選擇提供的 source code 資料夾, 勾選 help_functions.cpp、help_functions.h、

host.cpp 以及 kernel.h。

Import Sources @H	S03			x
File system	from the local file system			
From directory:	/home/hls03/Desktop/r10	943138/AlveoU50/vts_Opt1	Baseline 🔻	Browse
Filter Types	LBaseline Select All Desel	<pre></pre>	ions.cpp ions.h pp tAdd.cpp	
Into folder: vts	_Opt1Baseline/src			Browse
Options Overwrite ex Create top-lo Advanced >>	isting resources without w evel folder	arning		
?			Cancel	Finish



强 Explorer 🕱		🖻 😫 🖻	8
▼ ■ vts_Opt1Ba ▼ ● vts_Opt	aseline_system [xilinx_u5 1Baseline_kernels	0_gen3x16_x	dma_201920_3 [
🚌 src			
🔀 vts_(New	•	
🕨 📑 vts_Opt	📄 Сору	Ctrl+C	
🕶 📘 vts_Opt	💼 Paste	Ctrl+V	
🕨 🚮 Inclu	💢 Delete		
🔻 💕 src	街 Refresh	F5	
) 💽 he	눮 Import Sources		
⊧ ⊡inte ⊧⊡inte ⊧⊒iike	Import Sources	+	
🔀 vts_(Rena <u>m</u> e	F2	
≚ vts_Opt	T <u>e</u> am	+	
	Properties	Alt+Enter	
L			

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。

Import Sources @H	LS03			×
File system				
Import resources	s from the local file system.			
From directory:	/home/hls03/Desktop/r10943138/A	lveoU50/vts_Opt1Baseline	•	Browse
📃 🖶 vts_Opt	1Baseline	leinetions.cpp		
		🗌 🖻 help_functions.h		
		📃 🖻 host.cpp		
		🔽 🖻 K_KA.cpp		
		🔽 🖻 К_КВ.срр		
		🔽 🖻 K_KCalc.cpp		
		🔽 🖻 К_КрВ.срр		
		K_KVConstAdd.cpp		
		🗹 🖸 kernel.h		
Filter Types	Select All Deselect All			
Into folder: vts	_Opt1Baseline_kernels/src			Browse
Options				
Overwrite ex	kisting resources without warning			
Create top-l	evel folder			
Advanced >>				
?		Ca	ncel	Finish



下一步要加入 binary container, 打開 vts_Opt1Baseline_system_hw_link 專案(雙 擊 hw_link.prj), 在 Project Editor 點選 Add Binary Container 後, 底下會出現 binary_container_1。

🛎 vts_Opt1Baseline_s	ystem 💥 vts_Opt1Baseline 🛛 💥 vts_Opt1Ba	seline_system_hw_link 🕱		- 8
🔀 Hardware Lini	k Project Settings		Active build configuration:	Emulation-SW 🔻 🛞
General		Options		
Project name: <u>vts_0</u>	Opt1Baseline_system_hw_link	Target:	Software Emulation	
Platform: <u>xilinx</u>	< <u>u50_gen3x16_xdma_201920_3</u>	Kernel debug mode:	Waveform	~
		Export hardware (XS	5A):	
Hardware Functio	ns			E 🗄 🖻 🖉 🗙 🗌
Name	Compute Units			Add Binary Container
🖬 binary contai	ner 1			

最後要加入 hardware functions, 打開 vts_Opt1Baseline_kernels 專案 (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。

K Hardware Kernel Project Settings Active build configuration: Emulation-SW General Options Project name: vts_Opt1Baseline_kernels Target: Software Emulation Platform: xilinx_u50_gen3x16_xdma_201920_3 Kernel debug: Report level: Default ~ Hardware Functions Image: Software Emulation (-00) Name Port Data Width Max Memory Ports Add Hardware Function. KB Auto KB Auto Kcalc Auto Kx0 Auto Kx0 Auto Kx8 Auto	KHardware Kernel Project Settings Active build configuration: Emulation-SW General Options Project name: vts_Opt1Baseline_kernels Target: Software Emulation Platform: xilinx_u50_gen3x16_xdma_201920_3 Kernel debug: Report level: Default Report level: Default Hardware optimization: Default optimization (-00) KA Auto KA Auto KA Auto KA Auto KCalc Auto KCalc Auto KCalc Auto KorstAdd Auto KorstAdd Auto Kanto KorstAdd Kuto KorstAdd KorstAdd Kuto KorstAdd KorstAdd Kuto KorstAdd	🛎 vts_Opt1Baseline_	system 🛛 🛠 vts_Opt1Baseline	🛠 vts_Opt1Baseline_system_hw_link	🔀 vts_Opt	:1Baseline_kernels 🕱			
General Options Project name: vts_OptIBaseline_kernels Target: Software Emulation Platform: xlinx_US0_gen3x16_xdma_201920_3 Kernel debug: Report level: Report level: Default Hardware Functions Image: Software Emulation (-00) Name Port Data Width Max Memory Ports Add Hardware Function. KB Auto KB Auto Kcalc Auto KvConstAdd Auto KpB Auto	General Options Project name: vts_Opt1Baseline_kernels Target: Software Emulation Platform: xlinx_u50_gen3x16_xdma_201920_3 Kernel debug:	🔀 Hardware Ke	rnel Project Settings		-	Active build configuratio	on: Emulatio	n-SW ·	• 🛞
Project name: vts_Opt1Baseline_kernels Platform: xilinx_u50_gen3x16_xdma_201920_3 Report level: Default Hardware optimization: Default optimization (-00) Hardware Functions Name Port Data Width Max Memory Ports Add Hardware Function KB Auto KCalc Auto KCalc Auto KKB Auto KKB Auto KKB Auto KKB Auto KKB Auto KKB Auto	Project name: <u>vts_Opt1Baseline_kernels</u> Platform: <u>xilinx_u50_gen3x16_xdma_201920_3</u> Kernel debug: Report level: <u>Default</u> Hardware optimization: Default optimization (-00) Hardware Functions Name Port Data Width Max Memory Ports Add Hardware Function KKA Auto KKB Auto KKCalc Auto KKVConstAdd Auto KKpB Auto	General		Options					
Platform: xilinx_u50_gen3x16_xdma_201920_3 Kernel debug: Report level: Default Hardware optimization: Default optimization (-00) Hardware Functions Name Port Data Width Max Memory Ports Add Hardware Function KA Auto KB Auto KCalc Auto KKB Auto KKB Auto KKPB Auto	Platform: xilinx_u50_gen3x16_xdma_201920_3 Kernel debug: Report level: Default Hardware optimization: Default optimization (-00) Hardware Functions Name Port Data Width Max Memory Ports Add Hardware Function KA Auto KA Auto KKA Auto KKalc Auto KKVConstAdd Auto KKPB Auto	Project name: <u>vts</u>	Opt1Baseline_kernels	Target:		Software Emulation			
Report level: Default Hardware optimization: Default optimization (-00) Hardware Functions Name Port Data Width Max Memory Ports Add Hardware Function. 《 KA Auto 《 KB Auto 《 KCalc Auto 《 KVConstAdd Auto 《 KpB Auto	Report level: Default Hardware optimization: Default optimization (-00) Hardware Function: Name Port Data Width Max Memory Ports Add Hardware Function KA Auto KB Auto KKalc Auto KVconstAdd Auto	Platform: <u>xilir</u>	<u>1x_u50_gen3x16_xdma_201920_3</u>	Kernel debu	g:				
Hardware optimization : Default optimization (-00)	Hardware optimization: Default optimization (-00) Hardware Functions Name Port Data Width KA Auto KA Auto KB Auto KKalc Auto KVconstAdd Auto			Report leve	1:	Default			
Hardware Functions Name Port Data Width Max Memory Ports Add Hardware Functions	Hardware Functions Name Port Data Width KA Auto KB Auto KCalc Auto KCOnstAdd Auto KpB Auto			Hardware o	ptimization:	Default optimization (-00)			
Name Port Data Width Max Memory Ports Add Hardware Function. KA Auto Add Hardware Function. KB Auto Auto KB Kalac Auto	Name Port Data Width Max Memory Ports Add Hardware Function KA Auto KB Auto KCalc Auto KVConstAdd Auto KpB KpB	Hardware Functi	ons				Ē₿	Ø ×	A
KA Auto KB Auto KCalc Auto KVConstAdd Auto KpB Auto	KA Auto KB Auto KCalc Auto KVConstAdd Auto KpB Auto	Name	Port Data Width	Μ	ax Memor	y Ports	Add Hard	ware Func	tion
KB Auto KCalc Auto KVConstAdd Auto KpB Auto	KB Auto KCalc Auto KVConstAdd Auto KpB Auto	🖉 KA	Auto						
KCalc Auto KVConstAdd Auto KpB Auto	 ✓ KCalc Auto ✓ KVConstAdd Auto ✓ KpB Auto 	🖉 КВ			•				
KVConstAdd Auto KpB Auto	✓ KVConstAdd Auto ✓ KVpB Auto	🖉 KCalc							
KpB Auto	✓ KpB Auto	💋 KVConstAdd			-				
		💋 КрВ	Auto						



Add Hardware Functions @HLS03		×
Select an item to open (? = any character, $* =$ any string):		00
		•
More Options		
Matching items:		
◎ KA(int *, int *) - K_KA.cpp		
◎ KB(int *, int *) - K_KB.cpp		
○ KCalc(int *, int *, int *) - K_KCalc.cpp		
KVConstAdd(unsigned int, int *) - K_KVConstAdd.cpp		
Крв(Int *, Int *, Int *) - К_Крв.срр		
	Cancel Of	K

3.2. 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。

3.2.1. 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。

IMPORTANT:

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。





3.2.2. 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_201920_3 ./binary_container_1.xclbin

(用空白隔	開這三項)					
Run Configurations @HLS03						X I Bas
Create, manage, and run configura	tions					문 Outline 없
Debug a system using Vitis Debugger.					ration: Emulation-SW 🔻 🐯	There is no active editor an outline.
🖹 🖻 🔕 🗎 🗶 🖻 🍸 🖛	Name: SystemDebugger_vts	_Opt1Baseline_system				
type filter text	Main 📀 Target Setup 🖉 Er	nvironment 🔲 Commo	on			
Al Engine Emulator	Project:	vts_Opt1Baseline_s	ystem	Browse		
AI Engine SW Emulator	🔽 Disable build before laund	sh			<u> </u>	
C/C++ Application	Build Configuration:	Use Active		-		
Scaunch Group	Kernel Debug					
<pre>// OpenCL (TCF)</pre>	Use waveform for kernel	debugging	Edit Program Arguments @HLS03			×
👫 Single Application Debug	Launch live waveform		Application	2 Program	n Arguments	
🗧 Single Application Debug (GDB)	Debug only selected appli	ications	vts_Opt1Baseline	Xilinx xili	inx u50 gen3x16 xdma 201920 3 /b	inary container 1.xclbin
SPM Analysis	Selected Applications:					
System Project Debug System Debugger vts Opt1Bas	Program Arguments	Edit		-		
Target Communication Framewor	Program Argumenta.	Earch				
	Override Application Options:	Edit				
	Xilinx Runtime Profiling		-			
	Configuration:	Edit				
	Name	Key				
	OpenCL summary	opencl_su	n			
	Power profiling	power_pro	ofi 🗹 Automatically add binary co	ntainer(s) to argumer	nts	
Filter matched 12 of 21 items					Cano	сel ОК

編輯 Xilinx Runtime Profiling 的 Configuration, 設定為 OpenCL summary and Open

CL trace $_{\circ}$

Run Configurations @HLS03			Xilinx Runtime Profiling @HLS03	×
Create, manage, and run configura Debug a system using Vitis Debugger.	tions		Counters OpenCL summary Pow er profiling	
Image: Second	Name: SystemDebugger_vts_Opt1Baselini Main Orarget Setup Environment Debug only selected applications Selected Applications: Edit Override Application Options: Edit Configuration: Ledit Name OpenCL summary OpenCL summary OpenCL summary OpenCL summary OpenCL trace Low overhead OpenCL trace In Hw Shim trace Stall tr	System 2. Common Turns o Turns o Turns o Key Spencl_summar popencl_trace op_trace trt_trace ata_transfer_tr tall_trace	Hist Trace	Collect stall trace O None All External Memory Stall Inter cu Pipe Stall M Cancel OK
0			Close Run	<u></u>

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

Emulation 完成後, 會在 Console 看到 DONE 訊息。





3.2.3. Analysis

Emulation 執行完畢後,在 Assistant View 的 vts_Opt1Baseline 專案底下的 Emulation-SW 中會產生一個 Run Summary,可以雙擊打開 Vitis Analyzer 查看各項 report 進行分析。



在 Vitis Analyzer 中,點選 Application Timeline 可查看 host program 以及 kernel 運行的時序。





3.3. 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。

3.3.1. 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 \rightarrow hw_link \rightarrow host \rightarrow system)

※在 build hw_link 時,在 setting 加上 V++ 參數

	┥ Assistant 🔀			
		E E 🥝 🔦 O 🎋	8	
	✓ ➡ vts_Opt1B	aseline_syste		
	▼ 📘 vts_Opt	:1Baseline_sy Hw Link]		
	🕨 🂰 Emu	lation-SW [Software Emulation]		
	🕨 💰 Emu	lation-HW [Hardware Emulation]		
	🕨 🍕 Hard	ware [Hardware]		
	▼ 🛃 vts_Opt	Baseline_kernels [Hw Kernel]		
	🕨 🚮 Emu	lation-SW [Software Emulation]		
		ware [Hardware]		
	► Ko_opt	lation-SW [Software Emulation]		
	🕨 💰 Emu	lation-HW [Hardware Emulation]		
			, ,	
Hardware Function Settings @ic21				×
type filter text 🔳 🖪	Ø KA			(- ▼ <) ▼ 8
✓ [™] vts_Opt1Baseline_syster	Name: KA			
 Vts_Opt1Baseline Vts Opt1Baseline keri 	Stall profiling:			
▶ ≪ Emulation-SW	Max memory ports:			
✓ <i>✓ Emulation-HW</i>	Port data width: Au	uto 💌		
СКВ	Extra source files:			
KCalc	V++ configuration settings:			Apply
KVConstAdd		· · · · · · · · · · · · ·		
Hardware	V++ command line options:r	his.pre_tcl/his_config.tcl	a	
5 個 kernel 都要		Software Emulation ビー	opfig tel 败须	
田同一個 bls.con	fig tel 即可	上息 IIIS_C	.omg.tcl 哈徑	
	- A			
	Refresh			
	V++ Compiler Command Line —			
	\${XILINX_VITIS}/bin/v++			
	compile			
	hls.pre_tcl/hls_co	onfig.tcl		
	-o"build/KA.xo" "/sr	c/K_KA.cpp"		
			Revert	Apply
			Canad	
			Cancel	Apply and Close



3.3.2. 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。

🛎 vts_Opt1Baseline_system 🕱 🔀 vts_Opt1Baseli	ne 🛛 💥 vts_Opt1Baseline_system_hv	v_link 🛛 🛠 vts_Opt18	Baseline_kernels	
👗 System Project Settings			Active build configuratio	: Emulation-HW 🔻 👂
General		Options		Emulation-SW Emulation-HW
Project name: <u>vts_Opt1Baseline_system</u>		Target:	Hardware Emulation	Hardware
Runtime: OpenCL		Packaging options:		
Application Projects				+ - ×
Domain Name	Application Project Name		Build Configuratio	n
pl pl ×86	vts_Opt1Baseline_system_hw_link vts_Opt1Baseline vts_Opt1Baseline		Emulation-HW Emulation-HW Emulation-HW	

3.3.3. Analysis

步驟同 Software Emulation,請參考前述步驟,打開在 Emulation-HW 中的 Run Summary 查看各項 report 進行分析。



											Layout
≭ ≑ 4 _	xrt (Hardware Emulation) ×										
xrt (Hardware Emulation)											
 Summary 	Summary × Timeline Trace	Simulation Time	») ×								8 11
System Diagram	0.40.0.25 * *	¥ +[]	IN STREET	Fe of -F of	Compare						
Platform Diagram								418.802 s			
Run Guidance											
Profile Summary	Name	Value	4	410.0000000	0 \$	415.0000	00000 \$	420.0000	0000 \$	425.00000000 s	
Timeline Trace	Parallel Write 2										
	Copy	00									
	 Kernel Enqueues 				_						
	 xiink_uso_gntainer_1:ka Kemel Engueue 1 										
	v vilov u50 g. ptaiper 1:KB										
	Kernel Enqueue 1									8	
	vilinx u50 geiner 1:KCal										1
	Kernel Engueue 1										KCalc
	v xilinx_u50:KVConstAdd			1							
	Kernel Enqueue 1			KVConst Add							
	v xlinx_u50_gainer_1:KpB	1					1				
	Kernel Enqueue 1	КрВ					КрВ				
	vilinx_u50_genxdma_201920_3										
	v binary_container_1.xclbin										
	Compute Unit KA_1										
	Executions Executions					1					_
	Parallel Executions					HUNNING					
	Read Channel					_					
	Write Channel										
	Compute Unit KB 1										
	Executions									1	
	Parallel Executions									Running	
	v m_axi_gmem[0] (AIR)										
	Read Channel								-		
	Write Channel										_
	Compute Unit KCalc_1										
	Executions										1
	Parallel Executions										Running
	v m_axi_gmem] (A B R)										
	Read Channel										