

How to Prepare DPU Development Environment

버전 0 변경 6

2022 년 1 월 25 일 (2021 년 4 월 18 일)

요약

이문서는 DPU (Deep-learning Processing Unit)을 설계하고 검증하는데 필요한 환경과 프로그램 그리고 EDA (Electronics Design Automation)툴 설치에 관해 정리한다.

차례

요약	1
차례	1
1 개요	3
2 VirtualBox 설치 (Windows 사용자의 경우)	4
3 VirtualBox에 Ubuntu 설치 (Windows 사용자의 경우)	5
3.1 Ubuntu VDI 내려받기	6
3.2 Ubuntu 설치	6
3.3 Ubuntu 사용	7
3.4 확장 이미지 추가	8
4 Ubuntu에 필요한 프로그램 설치	8
4.1 Vim packages	8
4.2 Compiler packages	8
5 PyTorch 설치	9
5.1 Python3 설치	9
5.2 Anaconda3 설치	9
5.3 PyTorch 설치	9
5.4 필요한 패키지 설치	10
5.5 설치 확인	10
6 Xilinx Vivado WebPack 설치	11
6.1 Xilinx 계정 만들기	11
6.2 Vivado WebPack 설치	12
6.2.1 Windows의 경우	15
6.2.2 Ubuntu의 경우	15
6.3 설치 확인	16
7 GTKWave 설치	17

7.1 설치 확인	17
8 Cosim_bfm_library 설치	18
9 DLR: Deep-Learning Routines 설치	18
10 DLB: Deep-Learning Blocks 설치	18
11 변경이력	18

1 개요

DPU (Deep-learning Processing Unit)을 설계하는 환경은 다음 [그림 1]과 같다. 만약 Linux (e.g., Ubuntu)를 사용한다면 VirtualBox 와 Windows 는 무시할 수 있다.

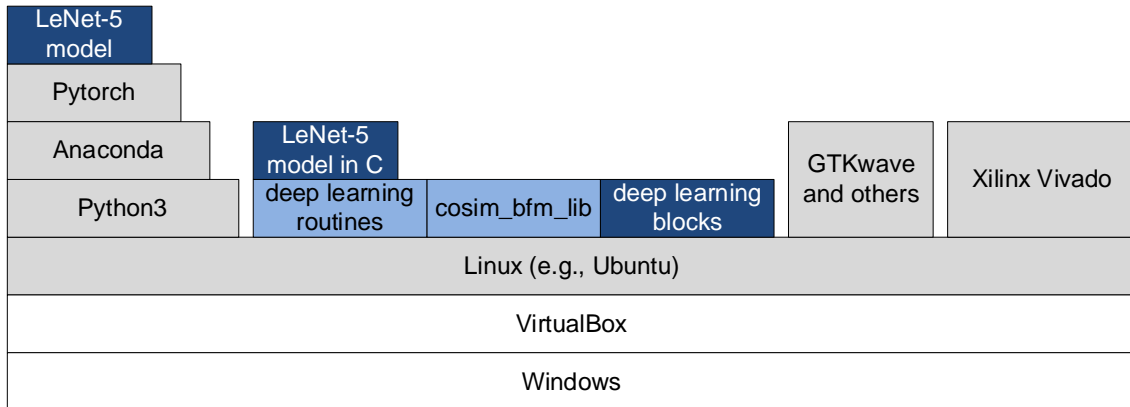


그림 1: 전체 개발 환경

각각은 다음을 목적으로 사용한다.

- Pytorch: Deep-learning model 을 개발하여 학습하기 위함
- Anaconda: Python 의 버전 의존성을 관리하기 위함
- Python3: Pytorch 를 위한 기본 환경
- DLR (Deep-Learning Routines): HLS(High-Level Synthesis) 가능한 Deep-learning primitive 들의 C/C++ 라이브러리
- CBL (Co-simulation BFM Library): HW-SW transaction-level co-simulation
- DLB (Deep-Learning Blocks): 합성가능한 Deep-learning primitive 들의 RTL
- GTKwave: VCD (Value Change Dump) 형식의 시뮬레이션 결과 신호 파형을 보기 위함.
- Xilinx Vivado: FPGA 용 HDL(Hardware Description Language, e.g., Verilog or SystemVerilog) 설계를 시뮬레이션하고 합성하기 위함

Linux (Ubuntu)를 사용한다면 '4 Ubuntu 에 필요한 프로그램 설치'부터 참고한다.

만약 Windows 환경을 사용할 경우, 다음을 추가로 설치하여 환경을 구성한다.

- VirtualBox: 가상머신
- Linux: 사용할 운영체제

[그림 2]는 Windows OS 환경에서 Ubuntu 를 사용하기 위한 환경을 개념적으로 표현한 것이며, 가상머신(Virtual Machine)을 통해 필요한 환경을 구성한다.

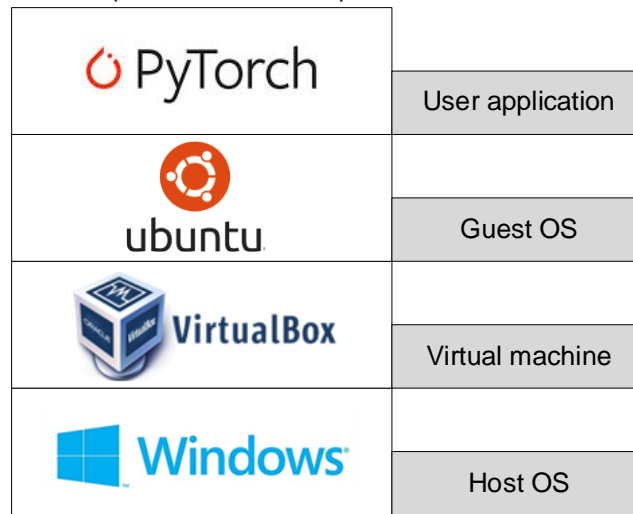


그림 2: 가상머신(virtual machine) 환경

전반적인 순서는 다음과 같다.

1. Windows 에 VirtualBox 를 설치 (Installing VirtualBox on Windows host machine)
2. VirtualBox 에 Ubuntu 를 설치 (Installing Ubuntu guest Operating System on VirtualBox)
3. Ubuntu 에 필요한 프로그램을 설치 (Installing user programs and libraries on Ubuntu)
 - GNU GCC
 - Additional if required for your applications
 - ✧ OpenCV, Python3, PyTorch, ...

2 VirtualBox 설치 (Windows 사용자의 경우)

[그림 3]에서와 같이 다음 웹 사이트에서 'Winodws hosts'용 VirtualBox 와 Extension Pack 을 내려 받는다.

- <https://www.virtualbox.org/wiki/Downloads>

내려 받은 프로그램을 설치한다. (단, 버전은 적절한 것으로 선택한다.)

- VirtualBox-6.1.30.exe

VirtualBox 6.1.30 platform packages

- Windows hosts
- OS X hosts
- Linux distributions
- Solaris hosts
- Solaris 11 IPS hosts

The binaries are released under the terms of the GPL version 2.

See the [changelog](#) for what has changed.

You might want to compare the checksums to verify the integrity of download *be favored as the MD5 algorithm must be treated as insecure!*

- SHA256 checksums, MD5 checksums

Note: After upgrading VirtualBox it is recommended to upgrade the guest

VirtualBox 6.1.30 Oracle VM VirtualBox Extension Pack

- All supported platforms

Support for USB 2.0 and USB 3.0 devices, VirtualBox RDP, disk encryption [chapter from the User Manual](#) for an introduction to this Extension Pack. the [VirtualBox Personal Use and Evaluation License \(PUEL\)](#). Please install version of VirtualBox.

그림 3: VirtualBox 내려받기

대략적인 순서는 다음 그림과 같이 진행되며, 설치가 잘 되었다면 [그림 4]의 제일 마지막 icon 이 desk-top window 에 생성될 것이다.

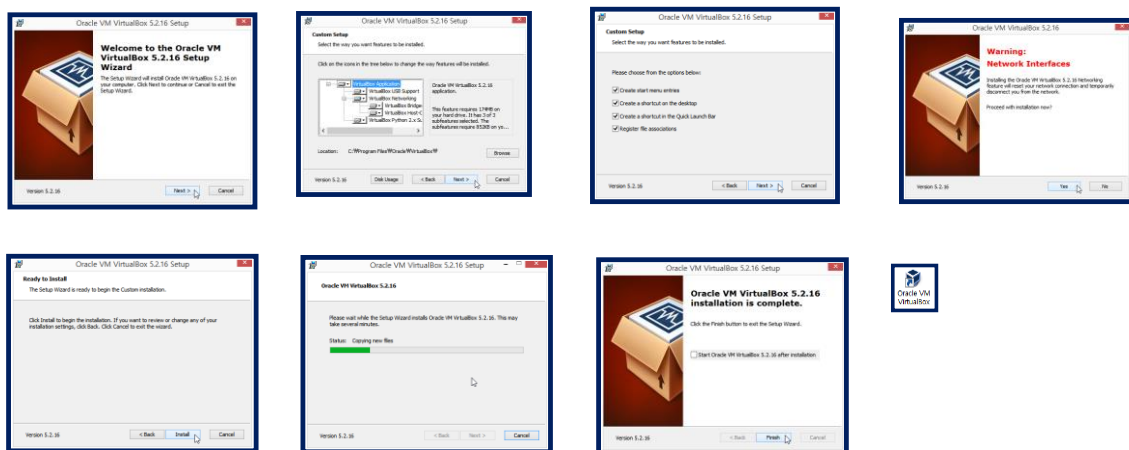


그림 4: VirtualBox 설치 순서

3 VirtualBox 에 Ubuntu 설치 (Windows 사용자의 경우)

가상머신에 Ubuntu 를 설치하는 방법은 다음 두 가지가 있으나, 여기서는 VDI 를 이용한다.

- VDI (Virtual Desktop Infrastructure)

- ISO (Optical disk image from the ISO 9660 file system)

3.1 Ubuntu VDI 내려받기

다음 사이트의 'VirtualBox Images' 페이지에서 'Ubuntu' 메뉴에서 VDI 를 내려 받는다.

- <https://www.osboxes.org/virtualbox-images/>

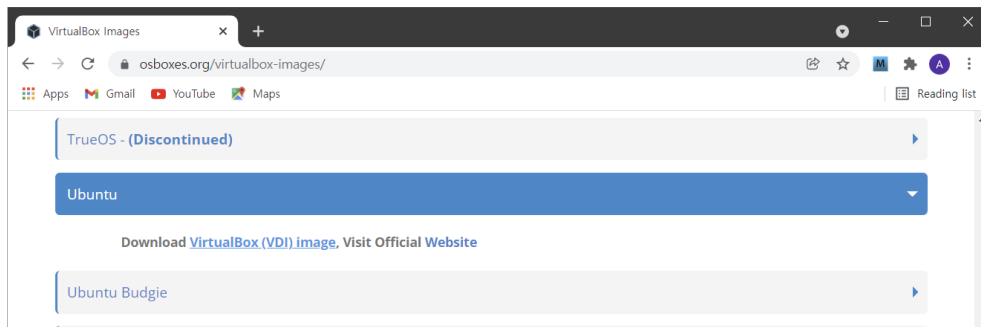


그림 5: VirtualBox image site

위 과정은 다음 사이트로 연결되고, 해당 사이트에서 'Ubuntu 18.04.6'을 내려 받는다. (내려 받은 후, 파일이름은 64bit.7z 일 수 있음)

- <https://www.osboxes.org/ubuntu/>

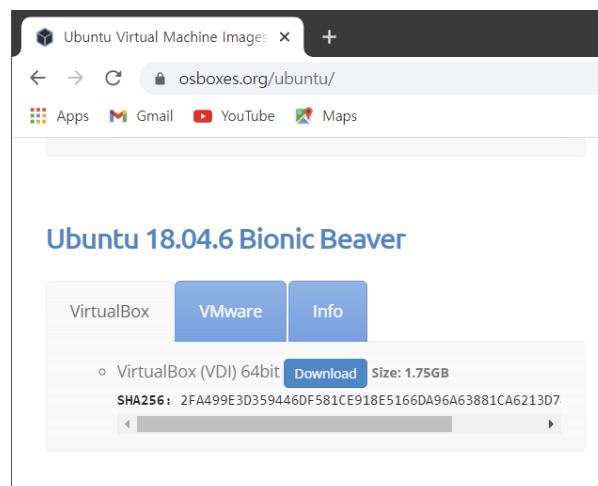


그림 6: Ubuntu VDI 내려 받기

3.2 Ubuntu 설치

미리 내려 받은 VDI 파일의 압축을 푼 후, 미리 설치된 VirtualBox 를 실행하고, VDI 를 로딩한다.

VDI: Virtual Desktop Infrastructure

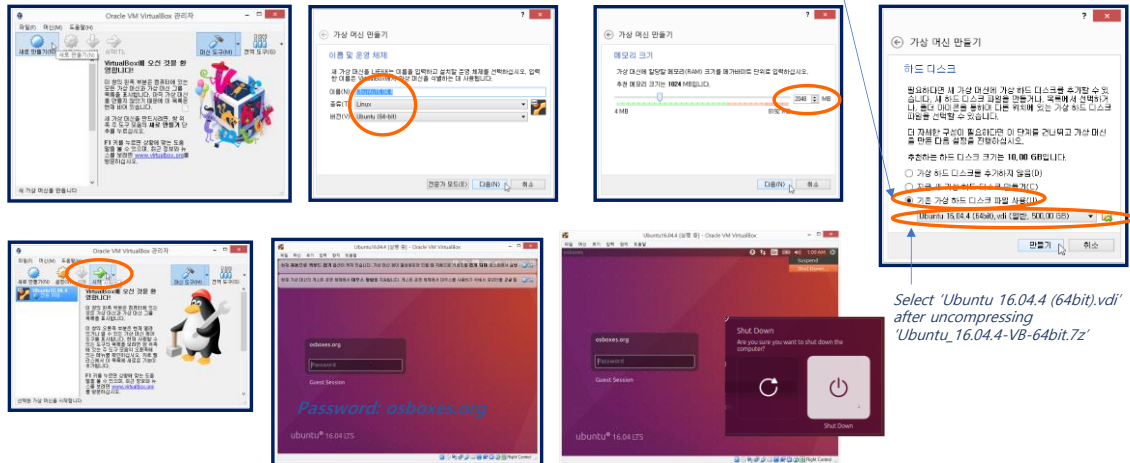


그림 7: VDI 로딩

3.3 Ubuntu 사용

모든 것이 잘 설치되어 VirtualBox 를 시작하면 [그림 8]과 같은 창이 보일 것이고, 여기서 '시작'을 클릭하여 Ubuntu 를 시작한다.

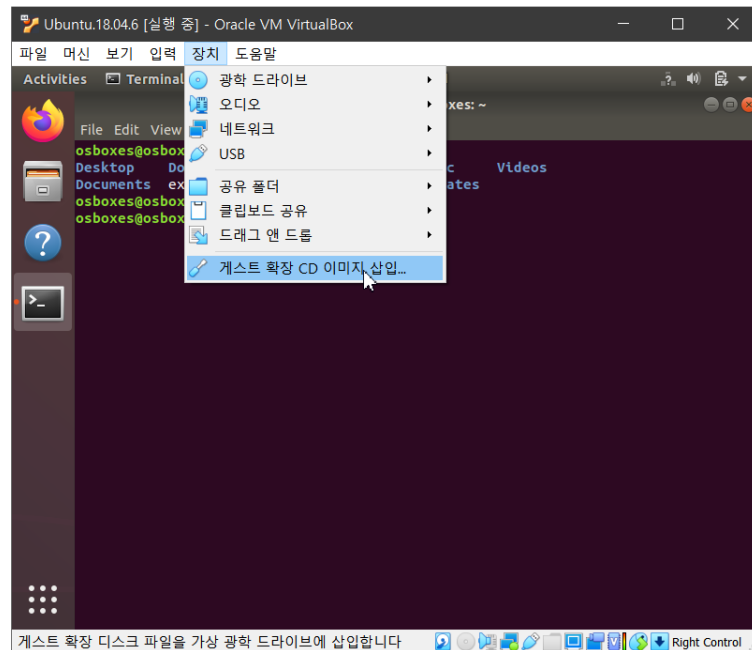


그림 8: Ubuntu 시작

VDI 로 설치한 Ubuntu 의 경우 다음과 같은 사용자 이름과 암호를 사용한다.

- Ubuntu 사용자 이름: osboxes.org
- Ubuntu 암호: osboxes.org

3.4 확장 이미지 추가



4 Ubuntu 에 필요한 프로그램 설치

다음은 Ubuntu 에서 실행하는 것이다.

4.1 Vim packages

```
$ sudo apt update  
$ sudo apt install -y vim
```

4.2 Compiler packages

프로그램을 컴파일하는 등 소프트웨어 개발에 필요한 기본 패키지들을 추가한다.

```
$ sudo apt update  
$ sudo apt install -y build-essential cmake unzip pkg-config
```

- gcc – This tool is the GNU compiler for the C Programming language.
- g++ – This package is the GNU compiler for the C++ programming language.
- libc6-dev – This is the GNU C library. This package contains the development libraries and header files used to compile simple C and C++ scripts.
- make – This is a useful utility that is used for directing the compilation of programs. The make tool interprets a file called a “makefile” that directs the compiler how to work.

- dpkg-dev – We can use this package to unpack, build and upload Debian source packages. This utility is useful if you want to package your software for Debian based system.
- cmake
- unzip
- pkg-config

5 PyTorch 설치

다음은 Ubuntu 에서 실행하는 것이다.

5.1 Python3 설치

```
$ sudo apt update
$ sudo apt install -y python3.7
$ sudo apt install -y python3-pip
```

5.2 Anaconda3 설치

```
$ cd /tmp
$ curl -O https://repo.anaconda.com/archive/Anaconda3-2021.05-Linux-x86\_64.sh
$ bash Anaconda3-2021.05-Linux-x86_64.sh
```

Conda 환경을 시작하기 위한 명령과 conda 환경을 종료하는 명령어는 다음과 같다.

```
$ conda activate
(base) $

..... conda 환경

(base) $ conda deactivate
```

5.3 PyTorch 설치

다음은 Conda 환경에서 실행하는 것이다.

다음 사이트를 방문하여 [그림 9]와 같이 선택한 후, 설치 명령어를 사용한다.

- <https://pytorch.org/get-started/locally/>

PyTorch Build	Stable (1.10)	Preview (Nightly)	LTS (1.8.2)
Your OS	Linux	Mac	Windows
Package	Conda	Pip	LibTorch
Language	Python	C++ / Java	Source
Compute Platform	CUDA 10.2	CUDA 11.3	ROCm 4.2 (beta)
Run this Command:	<pre>conda install pytorch torchvision torchaudio cpuonly -c pytorch</pre>		

그림 9: PyTorch

```
(base) $ conda install pytorch torchvision torchaudio cpuonly -c pytorch
```

설치가 완료되었다면, 다음과 같이 PyTorch의 버전을 확인할 수 있다.

```
(base) [osboxes@osboxes] python --version
Python 3.8.8
(base) [osboxes@osboxes] python
Python 3.8.8 (default, Apr 13 2021, 19:58:26)
[GCC 7.3.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import torch
>>> print(torch.__version__)
1.10.0
>>> quit()
(base) [osboxes@osboxes]
```

5.4 필요한 패키지 설치

다음은 Conda 환경에서 실행하는 것이다.
다음과 같은 Python 패키지들을 설치한다.

```
(base) $ pip3 install matplotlib
(base) $ pip3 install tensorboard
(base) $ pip3 install tensorboardX
```

5.5 설치 확인

첨부된 프로젝트(LeNet-5.pytorch.check.zip)의 압축을 풀어서 적절한 디렉토리(\$HOME/work)에 복사한 후, 해당 디렉토리에서 LeNet-5 학습을 실행한다.

```
$ conda activate
```

```
(base) $ cd LeNet-5.pytorch.check
(base) $ make run.train
.....
(base) $ conda deactivate
$
```

Conda 와 Python3 그리고 PyTorch 가 잘 설치되었다면 아래와 같은 결과를 확인할 수 있다. 단, 학습의 정확도는 매번 변화가 있으므로, 출력되는 내용은 다를 수 있다. 학습 과정에서 인터넷을 통해 MNIST 데이터 셋을 내려 받아야 하므로 네트워크가 동작해야 된다.

```
osboxes@osboxes: ~/work/SharedWithWin/LeNet-5.pytorch.check
File Edit View Search Terminal Help
t.test/MNIST/raw/t10k-labels-idx1-ubyte.gz
5120it [00:00, 18417527.00it/s]
Extracting dataset.test/MNIST/raw/t10k-labels-idx1-ubyte.gz to dataset.test/MNIST/raw
/home/osboxes/anaconda3/lib/python3.8/site-packages/torch/utils/data/dataloader.py:478: UserWarning: This DataLoader will create 8 worker processes in total. Our suggested max number of worker in current system is 1, which is smaller than what this DataLoader is going to create. Please be aware that excessive worker creation might get DataLoader running slow or even freeze, lower the worker number to avoid potential slowness/freeze if necessary.
  warnings.warn(_create_warning_msg(
idx: 0, error: 2.313102960586548
idx: 100, error: 2.182865858078003
idx: 200, error: 0.9749828577041626
idx: 300, error: 0.46717387437820435
idx: 400, error: 0.0992812067270279
idx: 500, error: 0.08698524534702301
epoch: 0, accuracy: 0.9152, elapsed: 0:00:16.173702 0:00:16.173713
-----
Look checkpoints/mnist_model_final.pth
Look checkpoints/mnist_params_final.pth
Look checkpoints/mnist_model_final.onnx
(base) [osboxes@osboxes]
```

그림 10: LeNet-5 학습 시험

6 Xilinx Vivado WebPack 설치

다음은 Ubuntu 에서 수행하는 것이며, 아래 내용은 Windows 에 Vivado 를 설치하는 과정이지만, Ubuntu 에서 설치하는 것도 유사할 것이다.

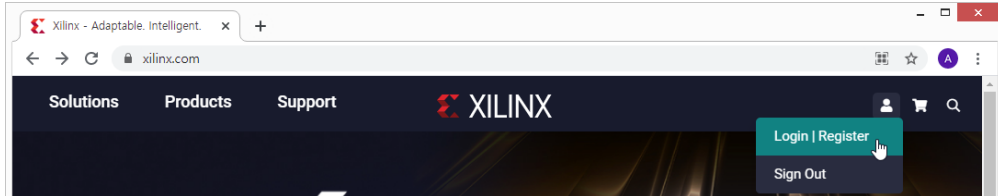
HDL simulator 로 다음을 사용한다.

- Xilinx Vivado WebPack 의 XSIM
 - ✧ WebPack 의 경우, License 없이 사용이 가능하며, 시뮬레이터와 합성기 등이 포함되지만, Xilinx FPGA 중 소규모 소자만을 지원하는 한계가 있다.

6.1 Xilinx 계정 만들기

Xilinx 웹 페이지의 중요 자원(패키지)에 접근하려면 Xilinx account 가 필요하므로, 미리 만든다. Xilinx 계정이 있다면 새로 만들 필요는 없다.

1. <https://www.xilinx.com/>
2. 위 1 번 웹 페이지의 오른쪽 위 계정 아이콘 선택한 후, 'Login|Register'을 선택한다.



3. 'Create Account' 선택 후, 필요한 정보를 입력한다.

A screenshot of the Xilinx 'Sign-In' form. It features two input fields: 'E-mail Address' and 'Password'. Below the 'E-mail Address' field, there is a red error message: 'Please enter an e-mail address'. A red 'Sign In' button is positioned below the password field. At the bottom of the form, there is a 'Create Account' button, which is highlighted with a hand cursor. Below this button, there are links for 'Forgot / Reset Password?' and 'Resend account activation e-mail?'. At the very bottom, a small note states: 'We're consistently improving our site security. Account login now uses e-mail addresses. Learn More'.

4. 필요한 정보를 입력한 후, 본인 이메일 계정으로 확인 이메일을 받아 후속 처리를 한다.

6.2 Vivado WebPack 설치

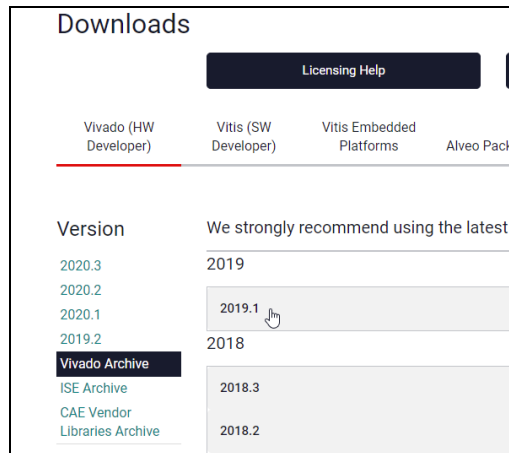
'How to install Xilinx Vivado WebPack'으로 인터넷 검색을 하여 적절한 자료를 참고하여 설치한다. 또는 다음 사이트를 참고할 수 있을 것이다.

- <https://blog.digilentinc.com/how-to-download-xilinxs-free-vivado-webpack-edition/>

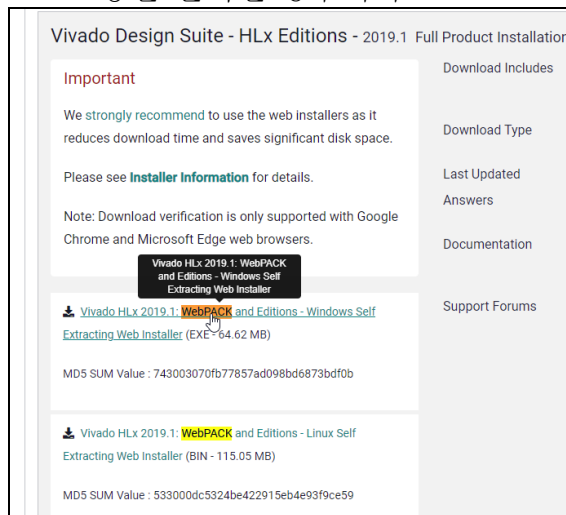
아래 설명한 순서로 설치를 할 수도 있다. 다음 웹 페이지를 통해 WebPack 2019.1¹용 Vivado WebPack 을 내려받아 설치한다.

1. <https://www.xilinx.com/support/download.html>
2. 위 1 번 웹 페이지에서 'Vivado Archive'를 선택하고, 이어서 오른쪽의 '2019.1'을 선택한다.

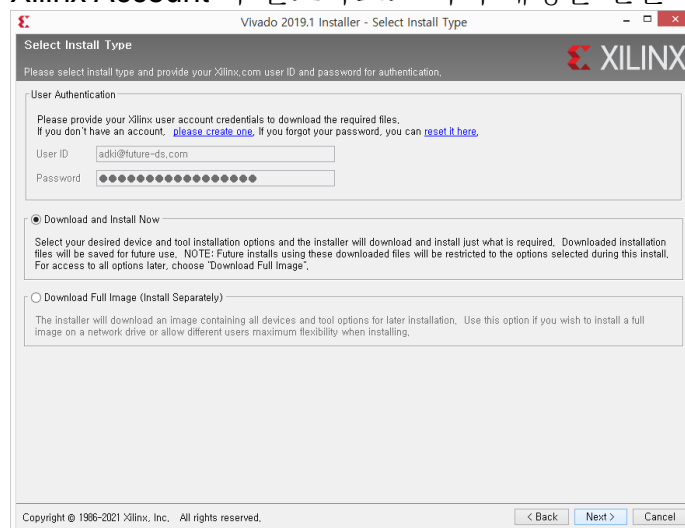
¹ Vivado.2017 은 SystemVerilog 지원에 한계가 있다.



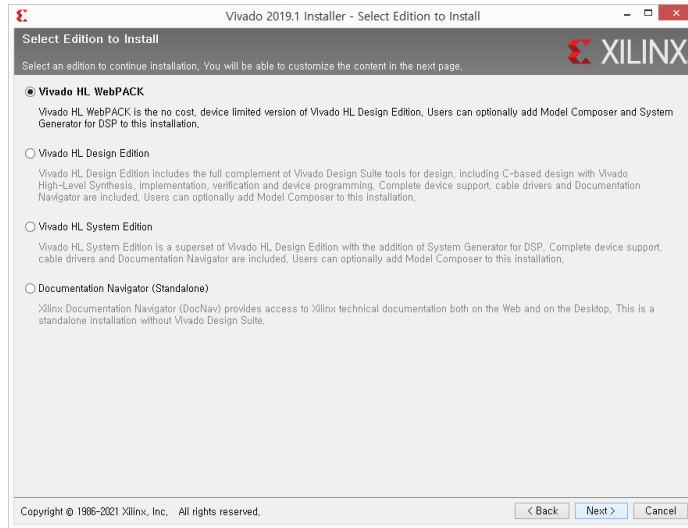
3. 위 2 번의 ‘2019.1’ 페이지에서 사용할 운영체제에 맞추어 내려 받는다.
다음 그림은 ‘Windows’용을 선택한 경우이다.



4. 내려 받은 파일을 설치한다.
5. 설치 과정에 “A New Version is Available”을 확인하고 나오면,
“Continue”를 선택하여 최신 버전이 아닌 2019.1 버전을 사용한다.
6. 설치 과정에 Xilinx Account 가 필요하므로 미리 계정을 만들어야 한다.



7. 'Select Install Type'에서 'Download and Install Now'를 선택한다.
8. 'Select Edition to Install'에서 'Vivado HL WebPACK'을 선택한다.

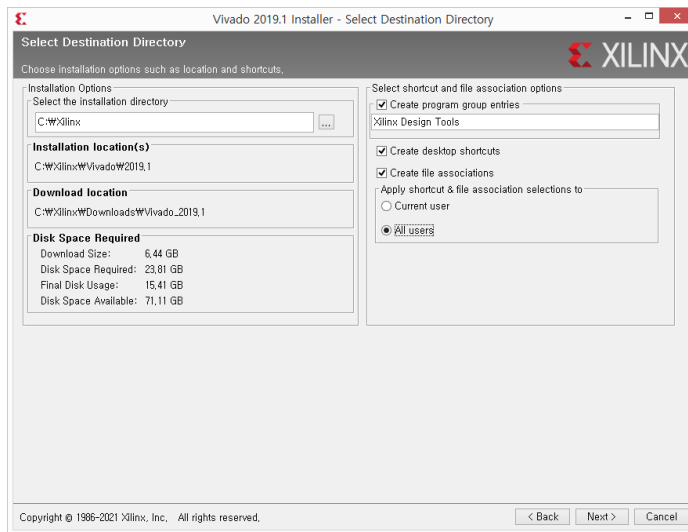


9. 'Vivado HL WebPACK'에서 기본을 그대로 유지하거나, 아니면 일부 사용하지 않는 소자나 패키지를 제외시켜서 설치에 소요되는 디스크 공간을 줄인다. (약 10Gbyte 가 필요하다)
 - 'bootgen' program²을 사용하려면 'Software Development Kit (SDK)' 을 선택해야 함.



10. 'Select Destination Directory'에서 대부분 기본값을 유지하고, 필요에 따라 'Current user' 대신 'All users'를 선택한다.

² 'bootgen' : Xilinx tool stitches binary files for PS and PL together and generate device boot images.

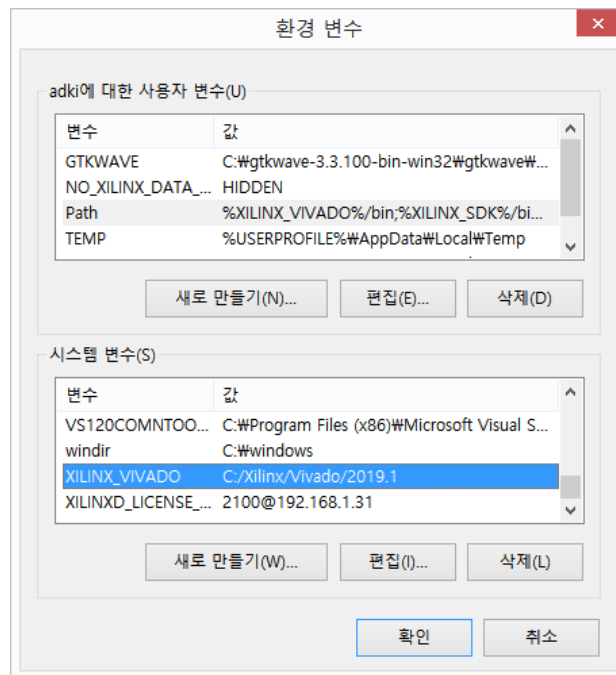


11. 이후, on-screen instruction 을 잘 읽고 적절하게 대응하여 설치한다.

6.2.1 Windows 의 경우

설치가 완료되면 다음을 확인한다.

Control Panel → System → Advanced system settings → Environment Variables



- System variables → "Path" 환경 변수에 다음이 포함되어야 함
 - ✧ C:\Xilinx\Vivado\2019.1\bin
- System variables → "XILINX_VIVADO" 환경 변수에 다음이 설정되어야 함
 - ✧ C:\Xilinx\Vivado\2019.1

6.2.2 Ubuntu 의 경우

Vivado 버전에 따라 설치되는 기본 디렉토리는 다를 수 있으며, 다음을 확인한다. 아래에서 '2019.1'은 설치된 Vivado 버전이므로 실제로 설치된 버전을 사용한다.

```
$ ls /opt/Xilinx/Vivado/2019.1/setting64.sh
```

또는

```
$ ls /tools/Xilinx/Vivado/2019.1/setting64.sh
```

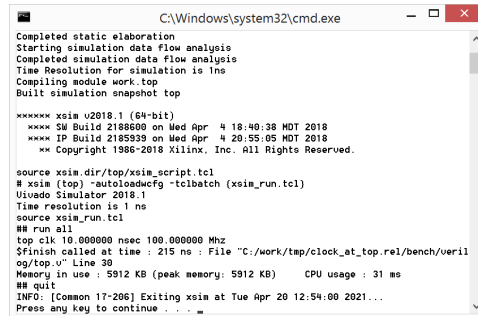
6.3 설치 확인

첨부된 프로젝트(clock_at_top.rel.zip)의 압축을 풀어서 적절한 디렉토리(\$HOME/work)에 복사한다.

Ubuntu 의 경우, 다음을 실행한다. ('settings64.sh'이 설치된 디렉토리는 Vivado 의 버전과 사용자 설정에 따라 다를 수 있다)

```
$ source /opt/Xilinx/Vivado/2019.1/settings64.sh
```

'clock_at_top.rel/sim/xsim' 디렉토리에서 'RunSimulation.sh'을 실행한다. 잘 실행되었다면 아래 그림과 같이 elaboration 과 simulation 이 수행된다.

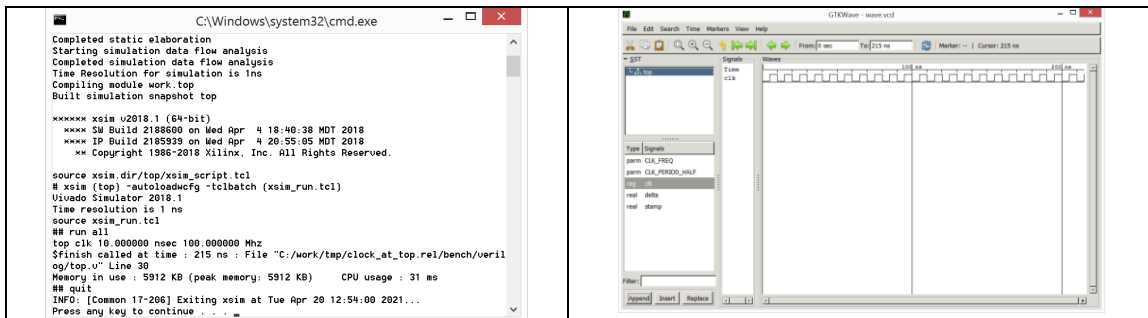


```
C:\Windows\system32\cmd.exe
Completed static elaboration
Starting simulation data flow analysis
Completed simulation data flow analysis
Time Resolution for simulation is 1ns
Compiling module work_top
Built simulation snapshot top

***** xsim v2018.1 (64-bit)
***** SW Build 2188600 on Wed Apr  4 18:40:38 MDT 2018
***** IP Build 2185939 on Wed Apr  4 20:55:05 MDT 2018
***** Copyright 1986-2018 Xilinx, Inc. All Rights Reserved.

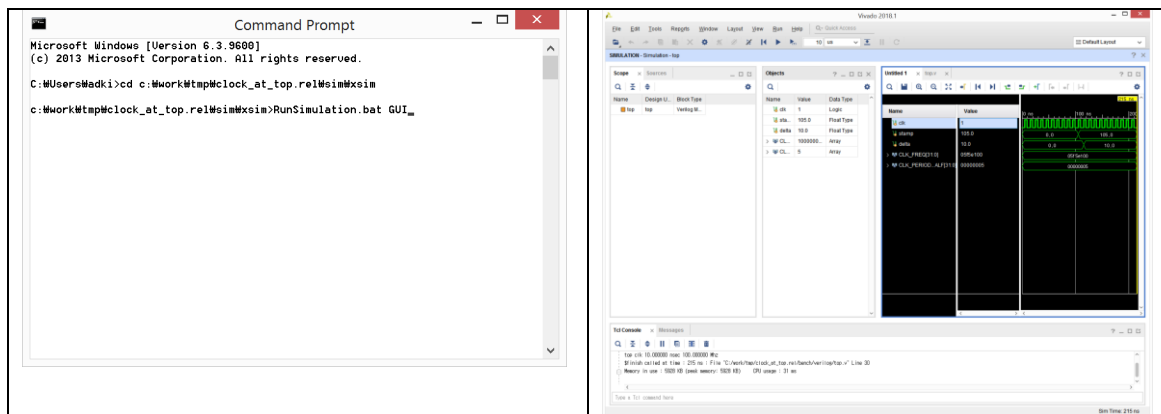
source xsim_dir/top/xsim_script.tcl
# xsim (top) -autoloadcfg -tclbatch (xsim_run.tcl)
Vivado Simulator 2018.1
Time resolution is 1 ns
source xsim_run.tcl
## run all
top clk 10.000000 nsec 100.000000 Mhz
$finish called at time : 215 ns : File "C:/work/tmp/clock_at_top.rel/bench/verilog/top.v" Line 30
Memory in use : 5912 KB (peak memory: 5912 KB) CPU usage : 31 ms
## quit
INFO: [Common 17-206] Exiting xsim at Tue Apr 20 12:54:00 2021...
Press any key to continue . . .
```

GTKwave 가 설치되어 있다면, 'RunGtwave.sh'를 실행하고, 잘 실행되었다면 아래 그림의 오른쪽과 같이 신호파형을 볼 수 있게 된다. (단, 아래 그림은 'top.clk' 신호를 선택한 경우)



GTKWave 대신 XSIM 에서 지원하는 내장 파형 뷰어를 사용할 수도 있는데, 이 경우는 다음과 같이 실행한다.

1. clock_at_top.rel/sim/xsim 디렉토리로 이동
2. 'RunSimulation.sh GUI' 명령 실행
3. Vivado Simulator 창에서
 - A. Window → Waveform 메뉴선택
 - B. 제일아래 명령어 입력 창에 "run all"



XSIM 과 Vivado 사용법은 Xilinx 관련 문서를 참고한다.

7 GTKWave 설치

'How to install GTKWave'로 인터넷 검색을 하여 적절한 자료를 참고하여 설치한다. 또는 다음 웹 페이지를 참고하고,

☆ <http://gtkwave.sourceforge.net/>

Ubuntu 에서는 다음 명령어로 설치한다.

```
$ sudo apt update
$ sudo apt install gtkwave
```

7.1 설치 확인

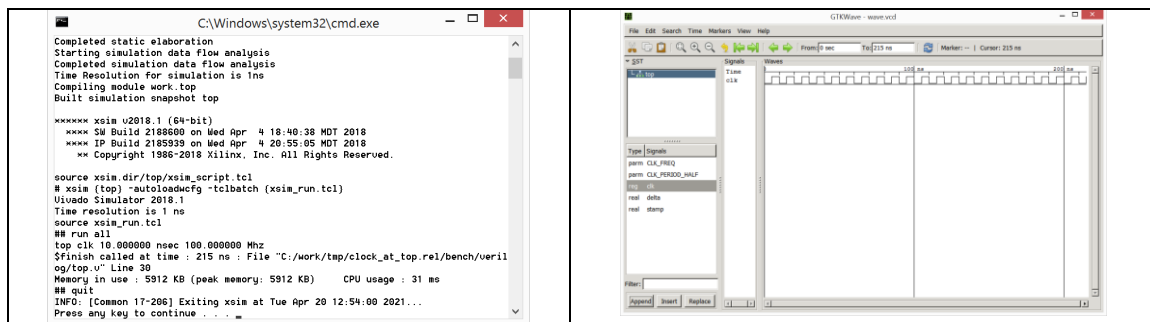
첨부된 프로젝트(clock_at_top.rel.zip)의 압축을 풀어서 적절한 디렉토리(C:/work)에 복사한다.

Ubuntu 의 경우, 다음을 실행한다. ('settings64.sh'이 설치된 디렉토리는 Vivado 의 버전과 사용자 설정에 따라 다를 수 있다)

```
$ source /opt/Xilinx/Vivado/2019.1/settings64.sh
```

‘sim/xsim’ 디렉토리에서 ‘RunSimulation.bat’을 실행한다. (Desktop 에서 double click 하거나 DOS CMD 창에서 ‘RunSimulation.bat’를 실행) 잘 실행되었다면 아래 그림의 왼쪽과 같이 elaboration 과 simulation 이 수행된다.

이어 ‘RunGtkwave.bat’를 실행하고, 잘 실행되었다면 아래 그림의 오른쪽과 같이 신호파형을 볼 수 있게 된다. (단, 아래 그림은 ‘top.clk’ 신호를 선택한 경우)



8 Cosim_bfm_library 설치

다음 두 개 사이트의 내용을 사용한다.

- https://github.com/adki/DPI_Tutorial
- https://github.com/adki/cosim_bfm_library
- https://github.com/adki/AMBA_AXI_AHB_APB

첫 번째 것은 DPI (Direct Programming Interface)에 대한 소개와 예제에 대한 것이며, 두 번째 것은 DPI 를 이용하여 AMBA AXI BFM 을 이용한 HW-SW 통합시뮬레이션에 대한 것이다. AMBA AXI 에 대한 것은 세 번째 것을 참고한다.

9 DLR: Deep-Learning Routines 설치

다음은 C/C++로 된 딥러닝 레이어 코드이며, 딥러닝 네트워크를 이해하는데 도움이 될 것이고, 설계한 딥러닝 블록의 기능 검증에 사용할 수 있다.

- https://github.com/github-fds/Deep_Learning_Routines

10 DLB: Deep-Learning Blocks 설치

DLB 는 이번 강좌에서 다룰 내용이며 강좌 중에 코드를 공유하게 된다.

11 변경이력

- 2022.01.25: ‘bootgen’ 내용 추가 (기안도)

- 2022.01.20: 일부 수정 (기안도)
- 2021.11.28: 내용 보강 (기안도)
- 2021.04.18: 작성 시작 (기안도, adki@future-ds.com)

– End of document –