INTRODUCTION TO
AI IMPLEMENTATION IN FPGA VIA VITIA-AI

(Wed) 27.01.2021
Colloquium at University of Seoul, Dept. of Physics
Sunyoung Yoo

LITTLE ABOUT MYSELF

MSc in Physics at University of Seoul

Training Program for Robotics of KAR
(Korea Association of Robot Industry)

Research Assistant at SDU
MMMI, AI and Data Science Lab

TABLE OF CONTENTS

TABLE OF CONTENTS

01

FPGA

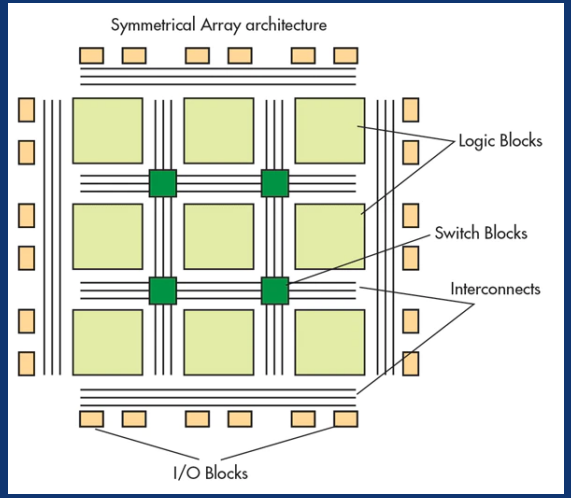
What is FPGA?
Why should we use it?

Field Programmable Gate Arrays

Field Programmable Gate Arrays

- Semiconductor devices based around a matrix of configurable logic blocks (CLBs) connected via programmable interconnects.
- Can be reprogrammed to desired application or functionality requirements after manufacturing.

— XILINX



Ref:https://www.electronicdesign.com/technologies/fpgas/article/21801527/the-principles-of-fpgas

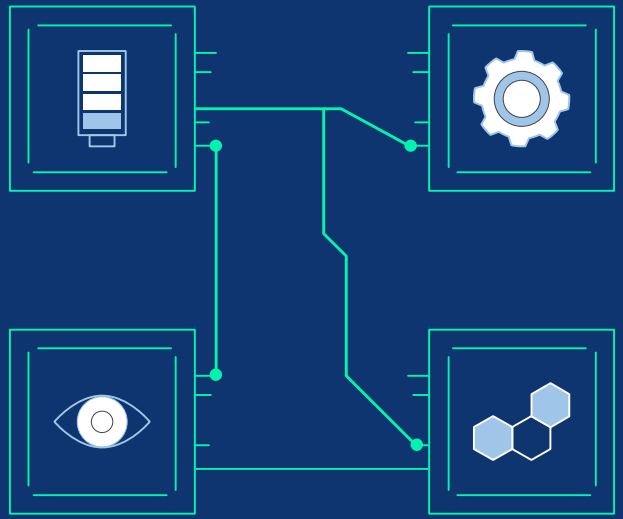
FPGA

ENERGY

Low energy consumption

PROTOTYPING

Fast prototyping



TIME

Low time consumption

VARIOUS OPTIONS

Easy to integrate with other devices

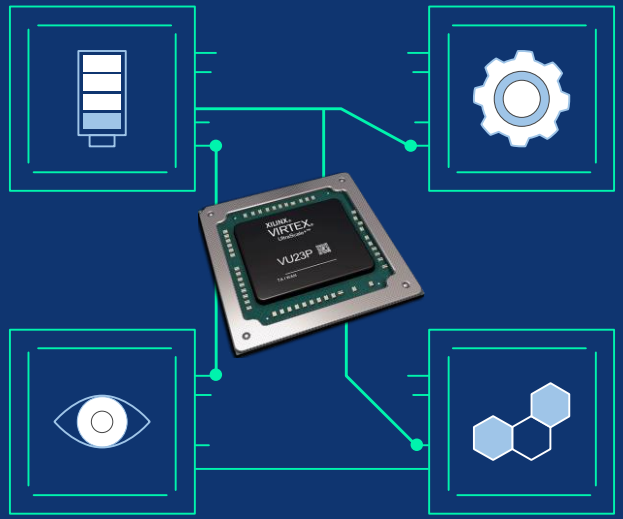
FPGA

ENERGY

Low energy consumption

PROTOTYPING

Fast prototyping



TIME

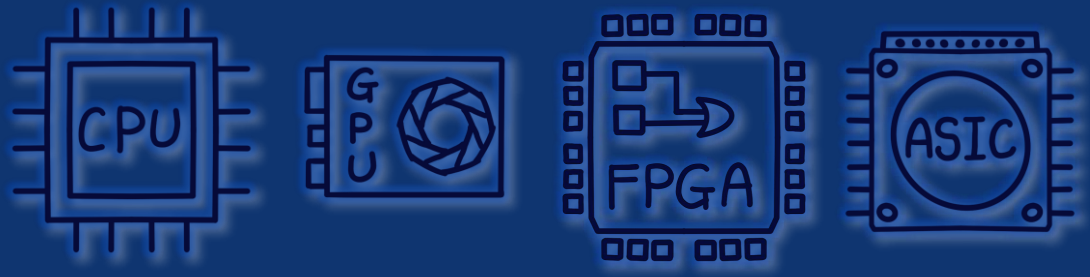
Low time consumption

VARIOUS OPTIONS

Easy to integrate with other devices

TIME & ENERGY EFFICIENCY

LOW → HIGH



HIGH → LOW

FLEXIBILITY

02

CAPSULE COLON ENDOSCOPY

About the project

CAPSULE COLON ENDOSCOPY

Capsule Endoscopy

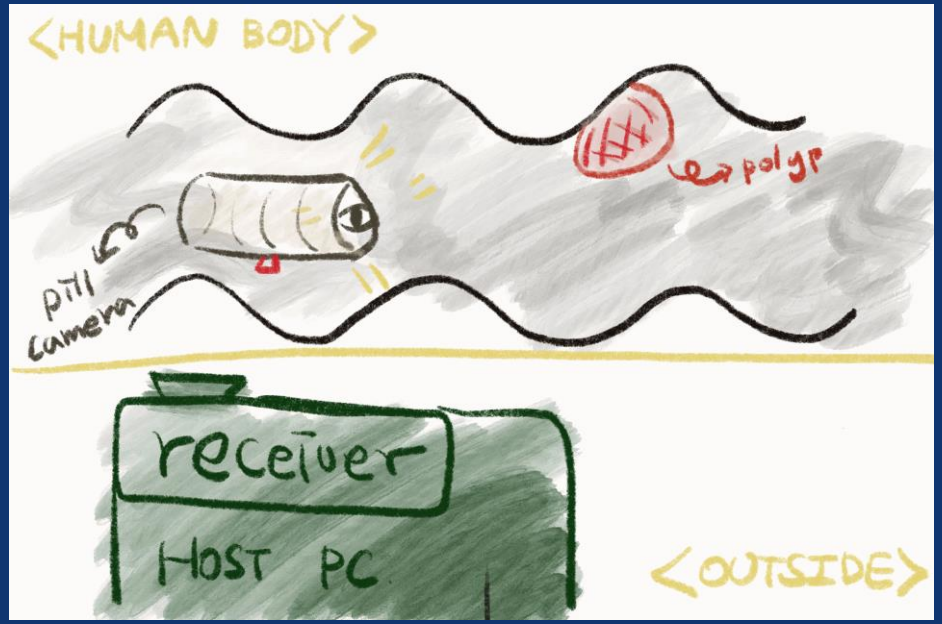
Devices used to perform endoscopy operations

				
Capsule	PillCam® SB 3 Given Imaging	EndoCapsule® Olympus America	MiroCam® IntroMedic Company	OMOM® Jinshan Science and Technology
Size	Length: 26.2 mm Diameter: 11.4 mm	Length: 26 mm Diameter: 1 1mm	Length: 24.5 mm Diameter: 10.8 mm	Length: 27.9 mm Diameter: 13 mm
Weight	3.00g	3.50g	3.25-4.70g	6.00g
Battery life	8 hours or longer	8 hours or longer	11 hours or longer	6-8 hours or longer
Resolution	340x340	512x512	320x320	640x480
Frames per second	2 fps or 2-6 fps	2 fps	3 fps	2 fps
Field of view	156°	145 °	170°	140°
Communication	Radio frequency communication	Radio frequency communication	Human body communication	Radio frequency communication
FDA approval	Yes	Yes	Yes	No
Price per capsule	$500	$500	$500	$250

Key Features

- low energy consumption
- long battery life
- stable communication
- small size
- harmless inside human body

CAPSULE COLON ENDOSCOPY



Key Features

- low energy consumption
- long battery life
- stable communication
- small size
- harmless inside human body

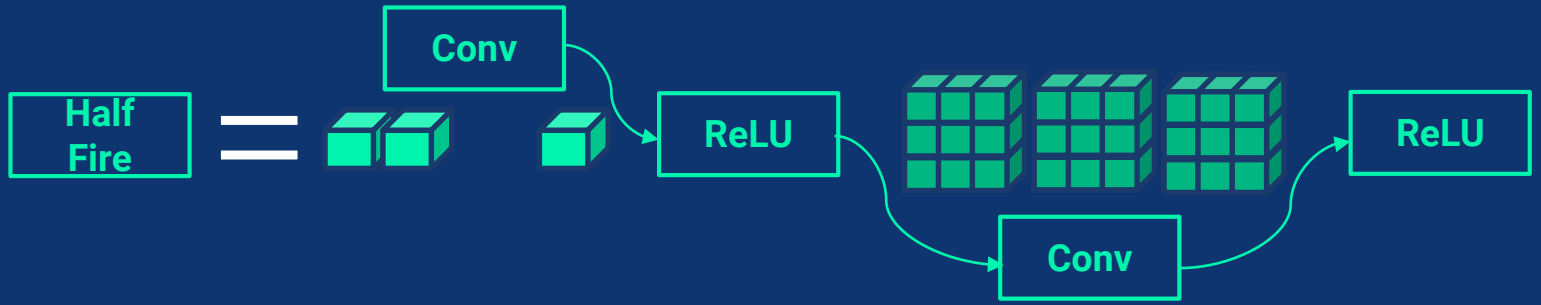
=> processing images before sending to the host can be more efficient.

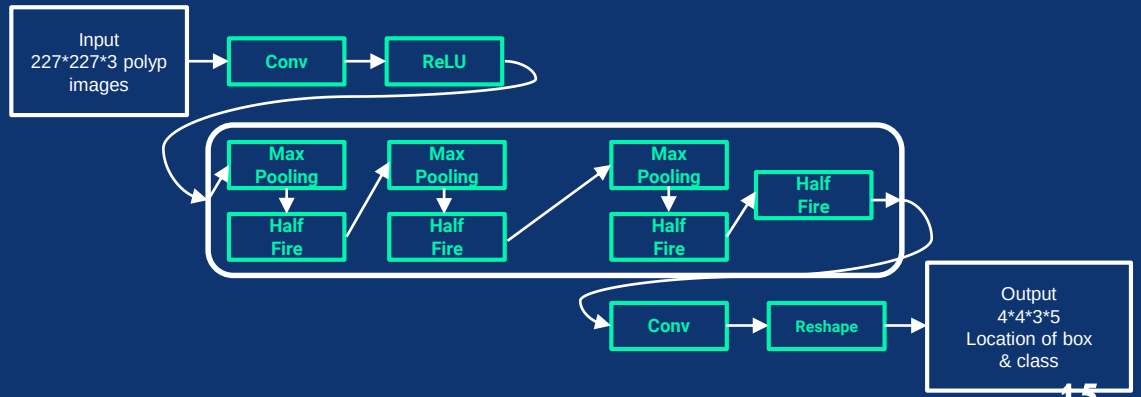
03

AI TRAINING

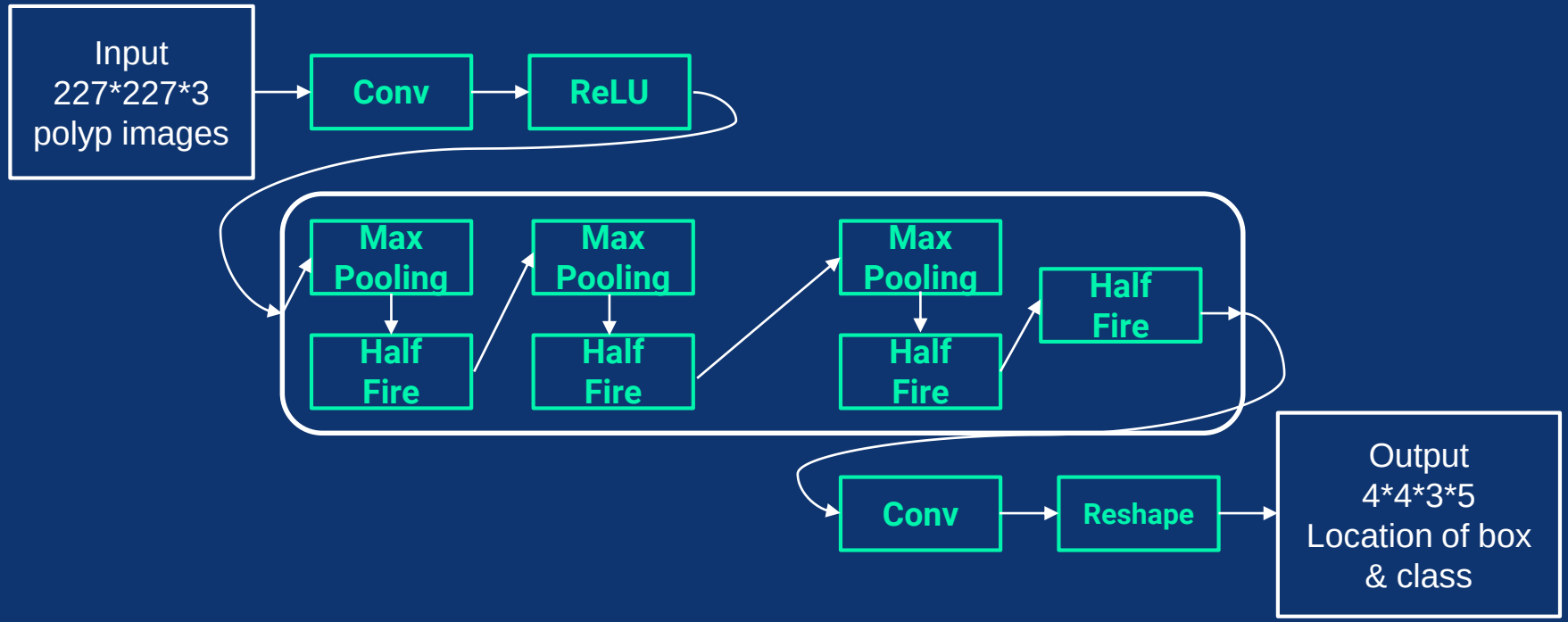
AI Model Design
Model Training

MODEL DESIGN

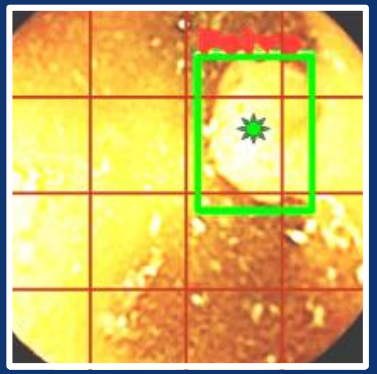




MODEL DESIGN



OUTPUT PROCESSING



- 4 X 4 cells on each Image
- 3 Candidates for each cell
- Remove Overlapped Candidates

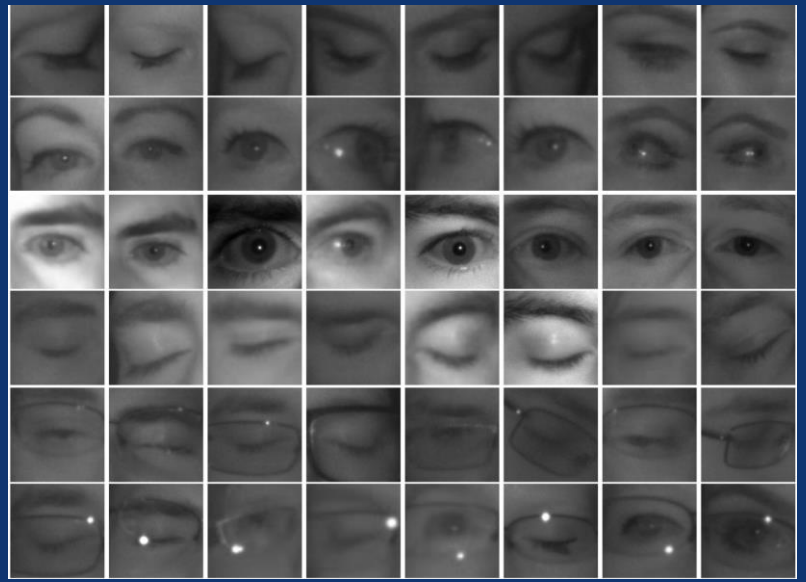
=> Objectness (Class confidence)

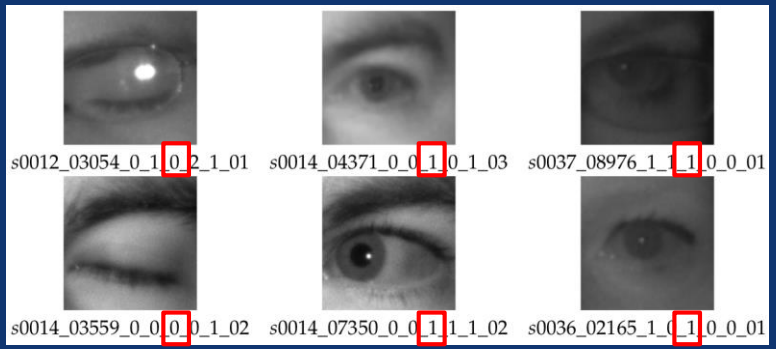
+ Bounding Box

CAN’T SHOW YOU THESE IMAGES, SORRY!

BINARY CLASSIFICATION

http://mrl.cs.vsb.cz/eyedataset





0: closed
1: open

BINARY CLASSIFICATION

http://mrl.cs.vsb.cz/eyedataset

https://github.com/YooSunYoung/binary_classification_vitis_ai_tutorial

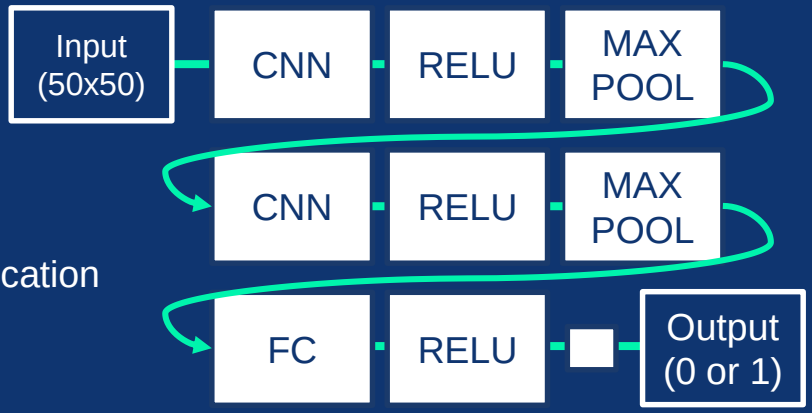
- Image Preprocess
  - randomly select images
  - rescale images into 50X50 gray images
  - split images into training and test set
  - save images into npy files
- Model
  - small model for single class binary classification
- Train
- Test
- Freeze
  - convert model into protobuf (.pb) format

BINARY CLASSIFICATION

http://mrl.cs.vsb.cz/eyedataset

https://github.com/YooSunYoung/binary_classification_vitis_ai_tutorial

- Image Preprocess
  - randomly select images
  - rescale images into 50X50 gray images
  - split images into training and test set
  - save images into npy files
- Model
  - small model for single class binary classification
- Train
- Test
- Freeze
  - convert model into protobuf (.pb) format



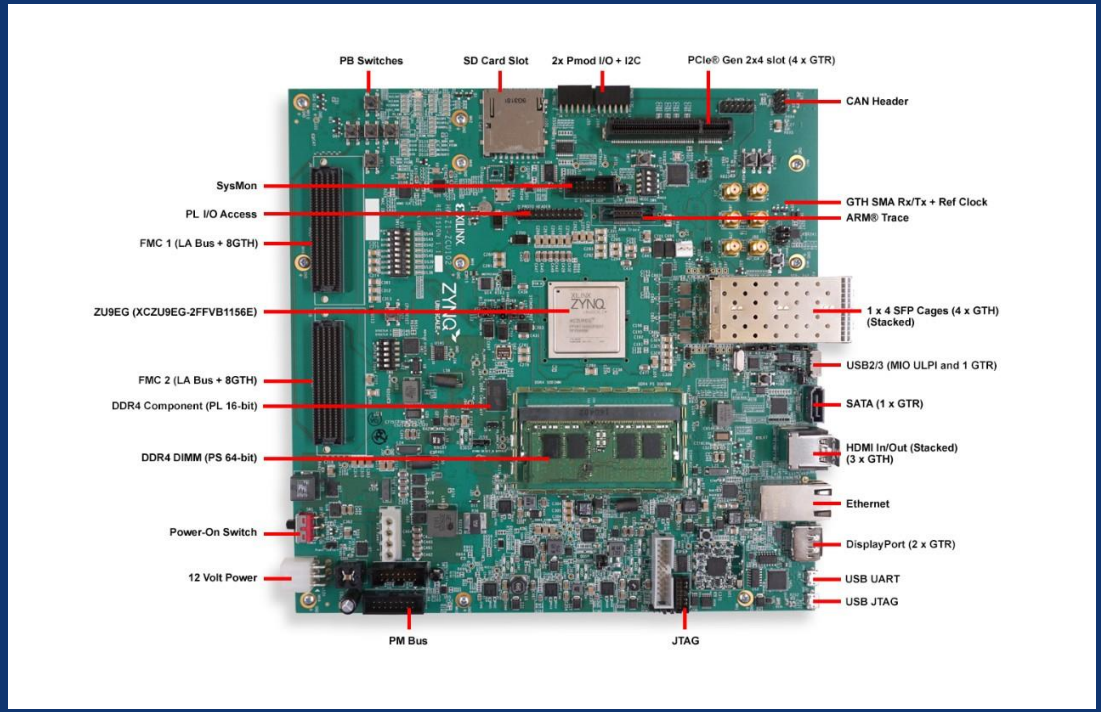
Loss: sigmoid_cross_entroy

04

ZYNQ EVALUATION BOARD ZCU-102

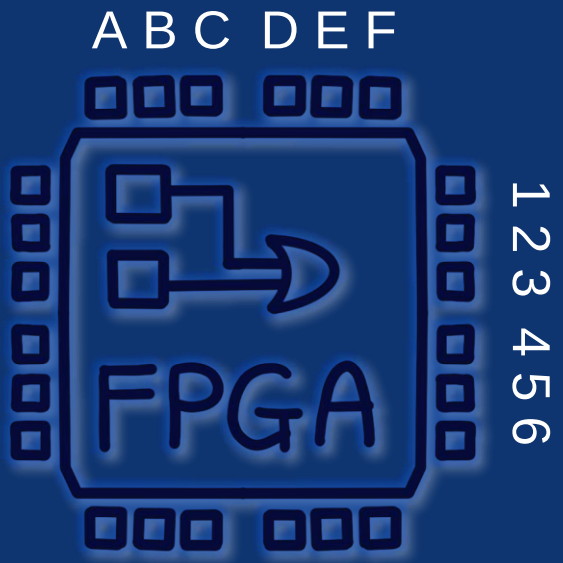
Hardware Information

ZCU-102



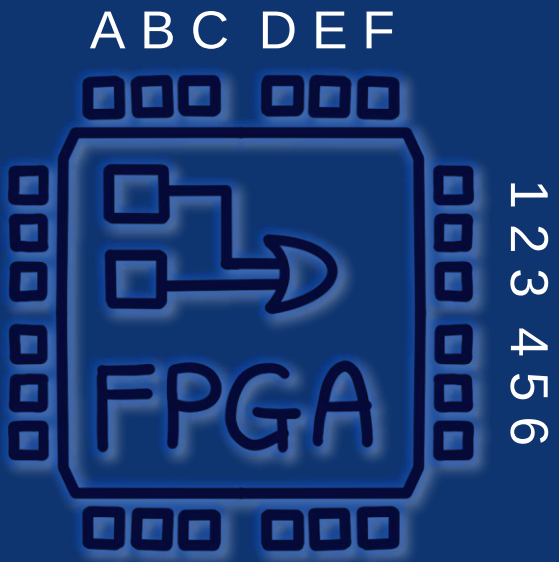
- Evaluation board for FPGA with various peripheral options

PERIPHERALS



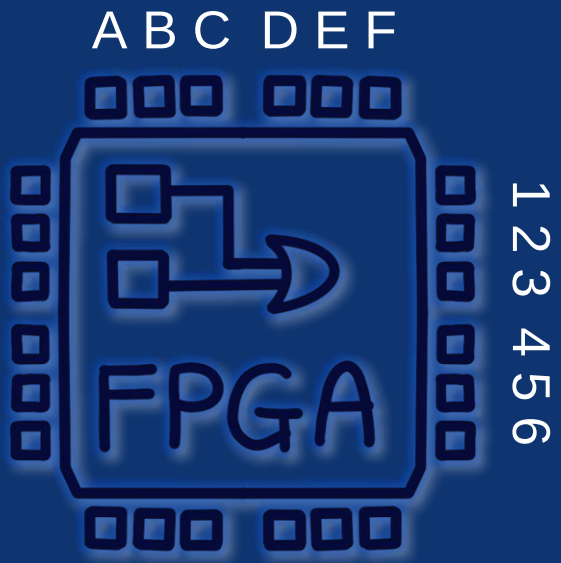
- Evaluation board for FPGA with various peripheral options

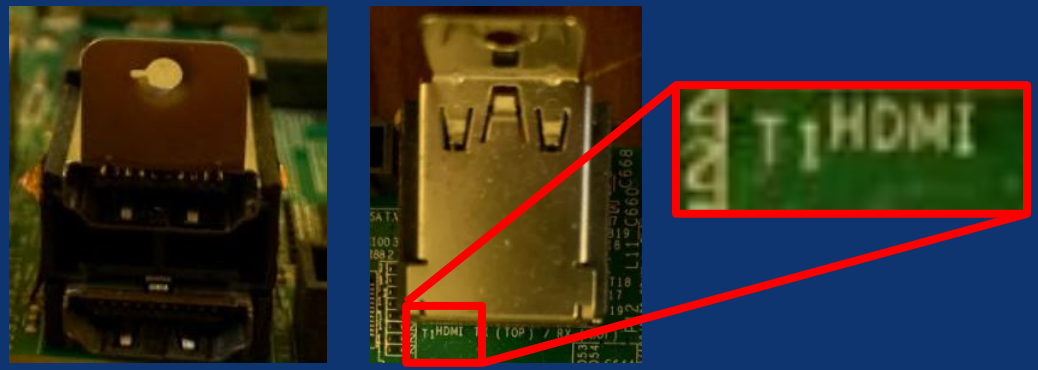
PERIPHERALS



- Evaluation board for FPGA with various peripheral options

PERIPHERALS





- Evaluation board for FPGA with various peripheral options

DETAIL INFORMATION

All pheripheral options can be found in the user guide linked below.

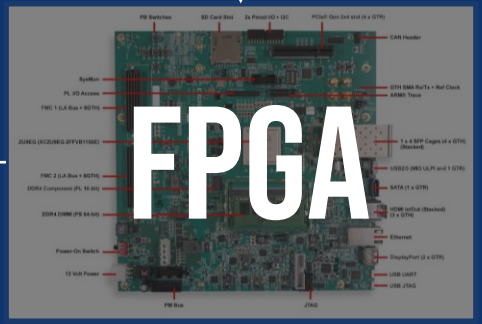
https://www.xilinx.com/support/documentation/boards_and_kits/zcu102/ug1182-zcu102-eval-bd.pdf

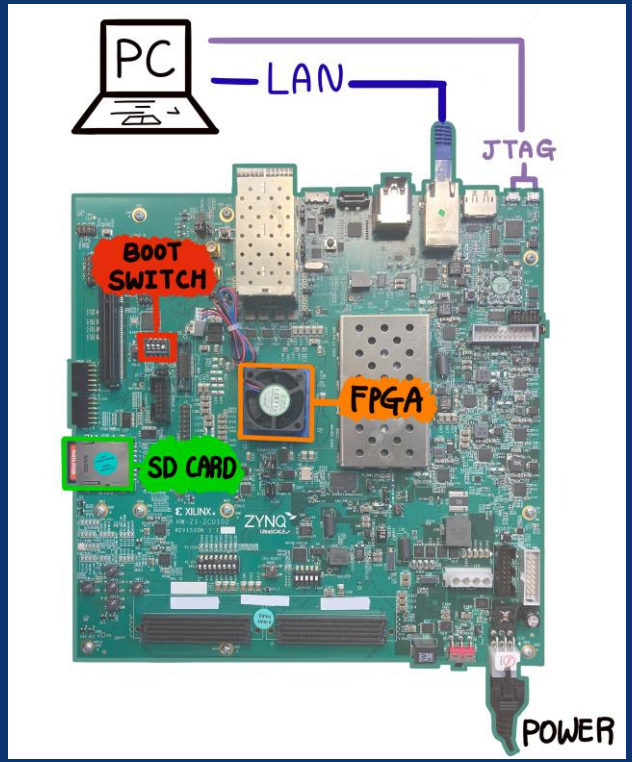
- Evaluation board for FPGA with various peripheral options

EXAMPLE APPLICATION

IMGAES

INFERENCE
RESULT



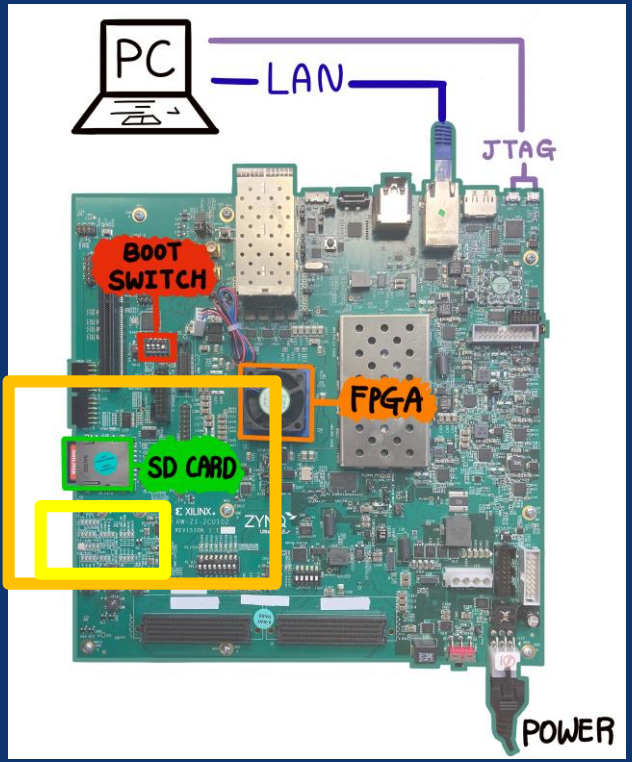


ZCU-102

- SD boot and LAN connection set-up

Boot switch should be set as ↑↓↓↓ before power on

ZCU-102

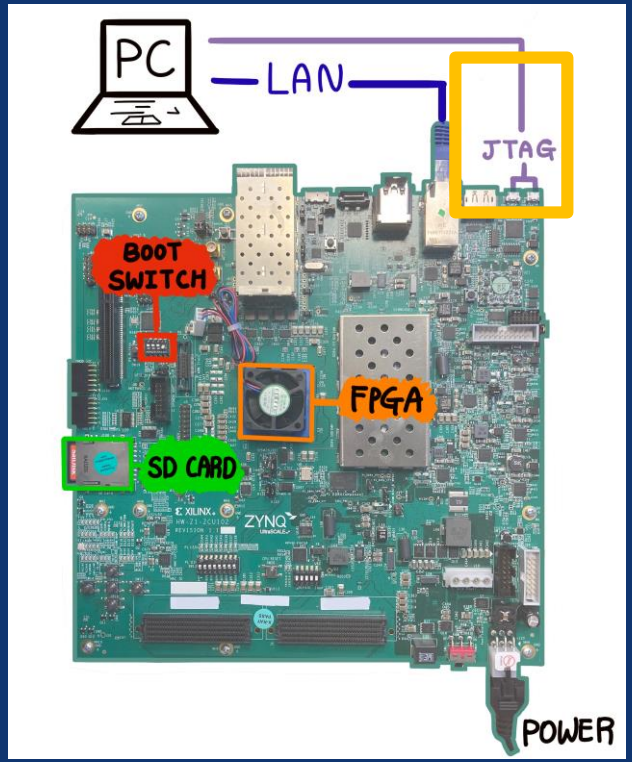


- Install Petalinux (pre-built) on SD card

Pre-built Peta Linux images:
https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18842316/Linux+Prebuilt+Images

How to install linux on SD card:
- Windows: https://www.etcher.net/ *recommend
- Linux: use ‘dd’ command

Ref: https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18841655/Prepare+Boot+Medium



ZCU-102

- Install Petalinux (pre-built) on SD card

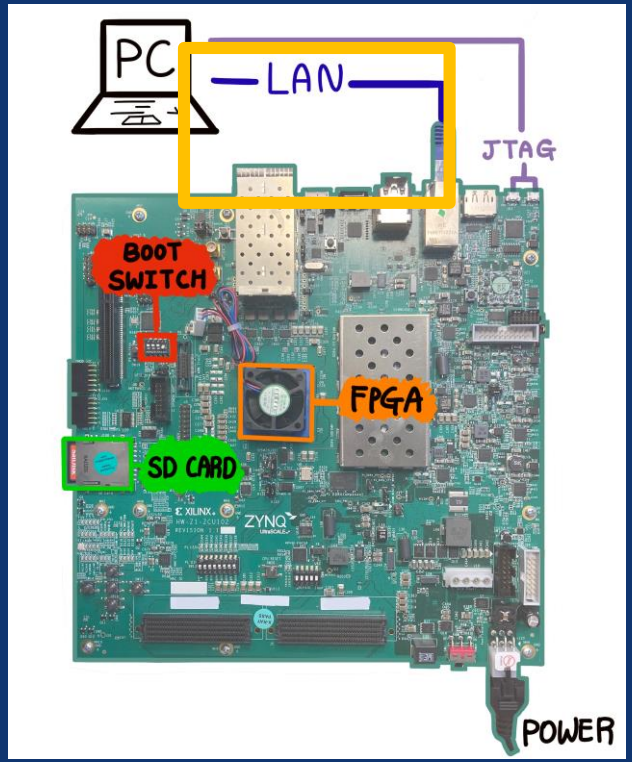
Pre-built Peta Linux images:
https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18842316/Linux+Prebuilt+Images

How to install linux on SD card:
- Windows: https://www.etcher.net/ *recommend
- Linux: use ‘dd’ command

Ref: https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18841655/Prepare+Boot+Medium

- UART Connection

Ref: https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18842446/Setup+a+Serial+Console



ZCU-102

- Install Petalinux (pre-built) on SD card

Pre-built Peta Linux images:
https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18842316/Linux+Prebuilt+Images

How to install linux on SD card:
- Windows: https://www.etcher.net/ *recommend
- Linux: use ‘dd’ command

Ref: https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18841655/Prepare+Boot+Medium

- UART Connection

Ref: https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18842446/Setup+a+Serial+Console

- LAN Connection (Static IP Configuration)

STATIC IP CONFIGURATION

ZCU-102 board

```
root@xilinx-zcu102-2020_1:~# sudo vi /etc/network/interfaces
```

PC (Ubuntu)

```
(base) syo@SYO:~$ sudo vi /etc/network/interfaces
```

STATIC IP CONFIGURATION

ZCU-102 board

```
root@xilinx-zcu102-2020_1:~# sudo vi /etc/network/interfaces
```

```
# Wired or wireless interfaces
auto eth0
iface eth0 inet static
        address 192.168.8.125
        netmask 255.255.255.0
        network 192.168.8.0
        broadcast 192.168.8.255
        gateway 192.168.8.1
```

PC (Ubuntu)

*DNS might need to be configured

```
(base) syo@SYO:~$ sudo vi /etc/network/interfaces
```

```
# auto enp4s0
# iface enp4s0 inet dhcp
auto enp4s0
iface enp4s0 inet static
        address 192.168.8.124
        netmask 255.255.255.0
        network 192.168.8.0
        broadcast 192.168.8.255
        gateway 192.168.8.1
```

STATIC IP CONFIGURATION

ZCU-102 board

```
root@xilinx-zcu102-2020_1:~# sudo vi /etc/network/interfaces
```

```
# Wired or wireless interfaces
auto eth0
iface eth0 inet static
        address 192.168.8.125
        netmask 255.255.255.0
        network 192.168.8.0
        broadcast 192.168.8.255
        gateway 192.168.8.1
```

PC (Ubuntu)

*dns might need to be configured

```
(base) syo@SYO:~$ sudo vi /etc/network/interfaces
```

```
# auto enp4s0
# iface enp4s0 inet dhcp
auto enp4s0
iface enp4s0 inet static
        address 192.168.8.124
        netmask 255.255.255.0
        network 192.168.8.0
        broadcast 192.168.8.255
        gateway 192.168.8.1
```

STATIC IP CONFIGURATION

ZCU-102 board

```
root@xilinx-zcu102-2020_1:~# sudo vi /etc/network/interfaces
root@xilinx-zcu102-2020_1:~# sudo /etc/init.d/networking restart
```

PC (Ubuntu)

```
(base) syo@SYO:~$ sudo vi /etc/network/interfaces
(base) syo@SYO:~$ sudo /etc/init.d/networking restart
[ ok ] Restarting networking (via systemctl): networking.service.
(base) syo@SYO:~$
```

STATIC IP CONFIGURATION

ZCU-102 board

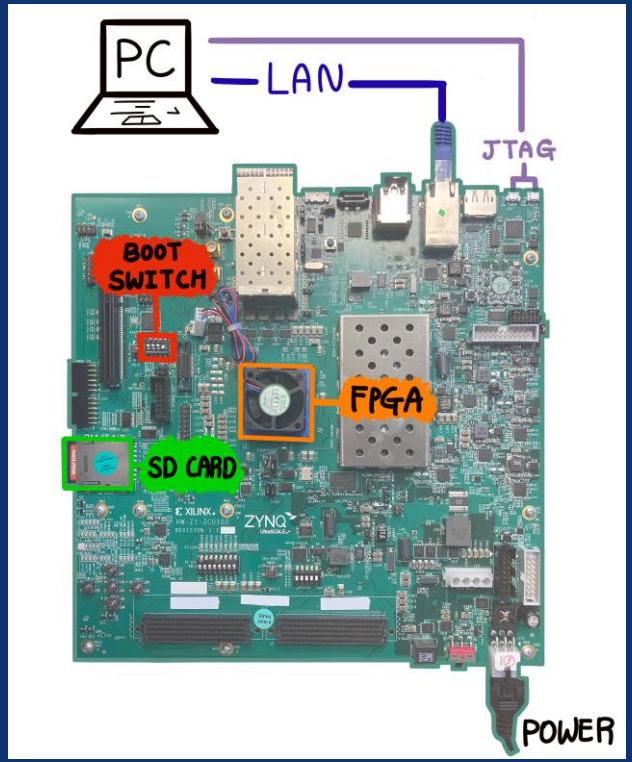
```
root@xilinx-zcu102-2020_1:~# sudo vi /etc/network/interfaces
root@xilinx-zcu102-2020_1:~# sudo /etc/init.d/networking restart
```

PC (Ubuntu)

```
(base) syo@SYO:~$ sudo vi /etc/network/interfaces
(base) syo@SYO:~$ sudo /etc/init.d/networking restart
[ ok ] Restarting networking (via systemctl): networking.service.
(base) syo@SYO:~$
```

```
(base) syo@SYO:~$ ssh root@192.168.8.125
```

ZCU-102



- Install Petalinux (pre-built) on SD card

Pre-built Peta Linux images:
https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18842316/Linux+Prebuilt+Images

How to install linux on SD card:
- Windows: https://www.etcher.net/ *recommend
- Linux: use ‘dd’ command

Ref: https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18841655/Prepare+Boot+Medium

- UART Connection

Ref: https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18842446/Setup+a+Serial+Console

- LAN Connection (Static IP Configuration)

- Evaluation board for FPGA with various peripheral options

05

VITIS-AI, AI IMPLEMENTATION

How to implement trained model
in FPGA via Vitis-AI

HARDWARE PROGRAMMING

HARDWARE PROGRAMMING

VHDL

HARDWARE PROGRAMMING

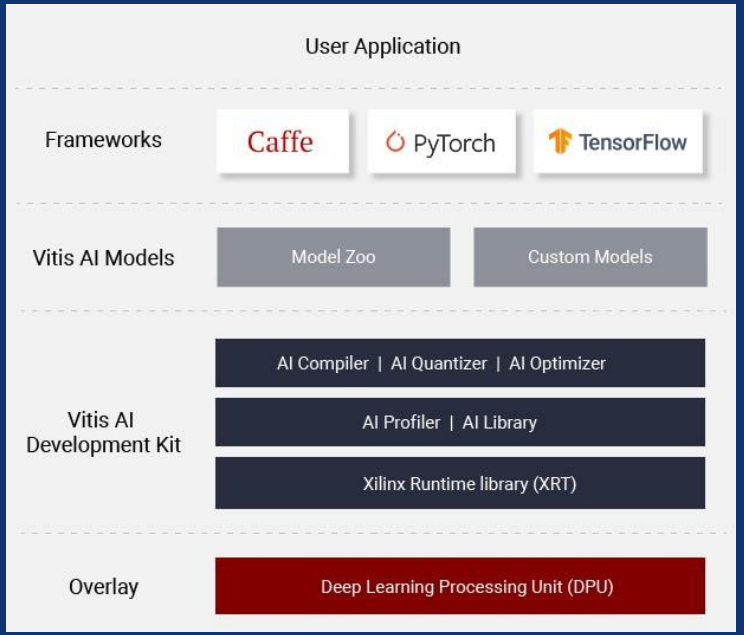
VHDL

VERILOG

HARDWARE PROGRAMMING

HARDWARE PROGRAMMING

HARDWARE PROGRAMMING



VITIS-AI ENVIRONMENT SET-UP

VITIS-AI ENVIRONMENT SET-UP

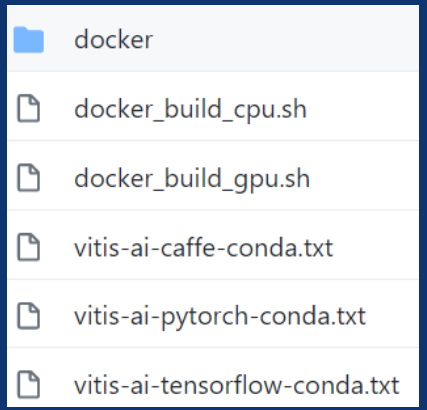
VITIS-AI ENVIRONMENT SET-UP

https://hub.docker.com/r/xilinx/vitis-ai

VITIS-AI ENVIRONMENT SET-UP

https://hub.docker.com/r/xilinx/vitis-ai

https://github.com/Xilinx/Vitis-AI/tree/master/setup/docker

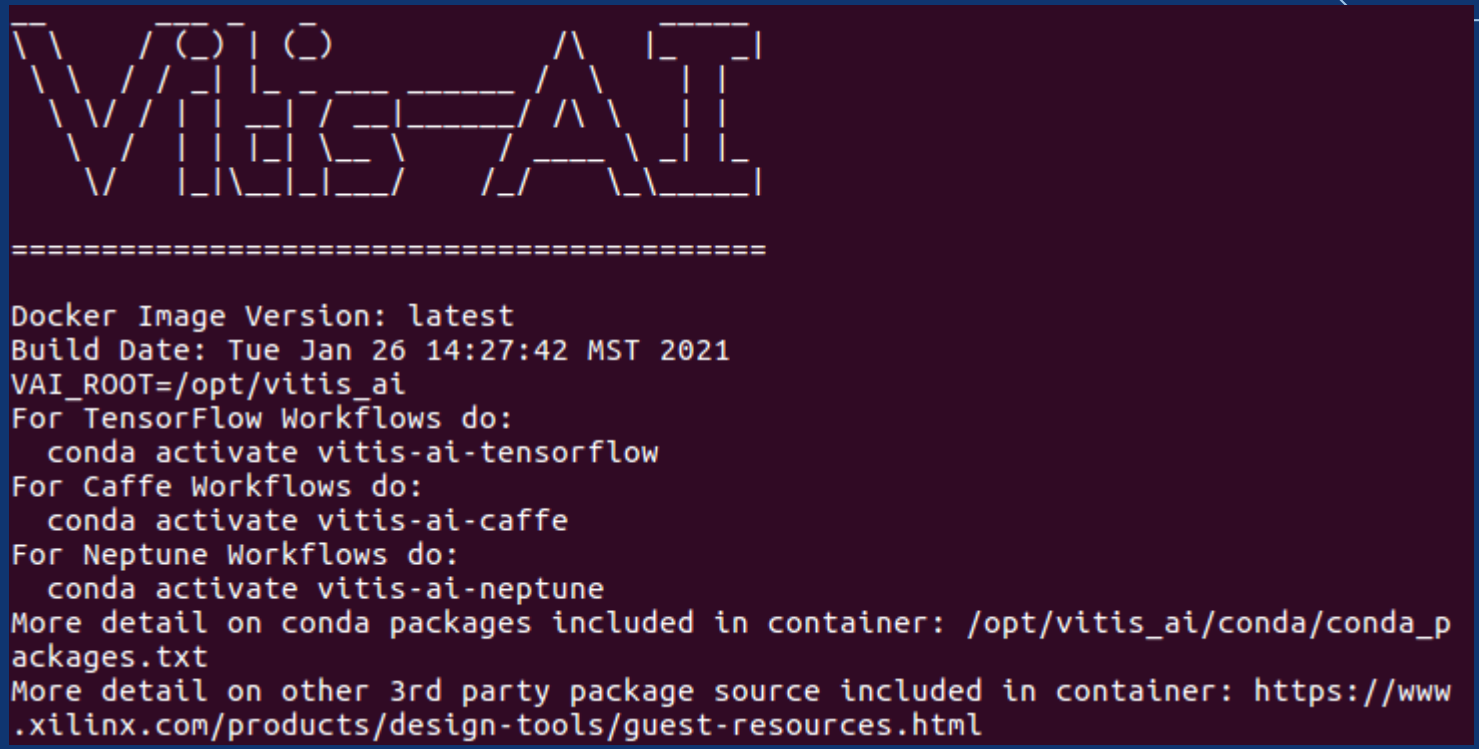


VITIS-AI ENVIRONMENT SET-UP

https://github.com/Xilinx/Vitis-AI/tree/master/setup/docker

```
(base) syo@SYO:~/Vitis-AI$ ./docker_run.sh xilinx/vitis-ai:latest
```

-v {pwd} /workspace
By default


__ ___ _ _ _____
\ \ / (_) | (_) /\ |_ _|
\ \ / / _| |_ _ ___ _____ / \ | |
\ \/ / | | __| / __|_____/ /\ \ | |
\ / | | |_| __ \ / ____ \ _| |_
\/ |_|__|_|___/ /_/ ______|
==========
Docker Image Version: latest
Build Date: Tue Jan 26 14:27:42 MST 2021
VAI_ROOT=/opt/vitis_ai
For TensorFlow Workflows do:
conda activate vitis-ai-tensorflow
For Caffe Workflows do:
conda activate vitis-ai-caffe
For Neptune Workflows do:
conda activate vitis-ai-neptune
More detail on conda packages included in container: /opt/vitis_ai/conda/conda_p
ackages.txt
More detail on other 3rd party package source included in container: https://www
.xilinx.com/products/design-tools/guest-resources.html

VITIS-AI ENVIRONMENT SET-UP

```
(base) syo@SYO:~/Vitis-AI$ ./docker_run.sh xilinx/vitis-ai:latest
```

-v {pwd} /workspace
By default

In Docker Container

```
syo@SYO:/workspace$ conda activate vitis-ai-tensorflow
```

vitis-ai-neptune
vitis-ai-caffe

VITIS-AI ENVIRONMENT SET-UP

```
(base) syo@SYO:~/Vitis-AI$ ./docker_run.sh xilinx/vitis-ai:latest
```

-v {pwd} /workspace
By default

In Docker Container

```
syo@SYO:/workspace$ conda activate vitis-ai-tensorflow
(vitis-ai-tensorflow) syo@SYO:/workspace$ which python
/opt/vitis_ai/conda/envs/vitis-ai-tensorflow/bin/python  -> PYTHON PATH
(vitis-ai-tensorflow) syo@SYO:/workspace$
```

VITIS-AI AI IMPLEMENTATION

Ref: https://github.com/Xilinx/Vitis-AI-Tutorials/tree/CIFAR10-Classification-with-TensorFlow
Ref: https://beetlebox.org/category/tutorials/

https://github.com/YooSunYoung/binary_classification_vitis_ai_tutorial

Docker Container

VITIS-AI AI IMPLEMENTATION

Ref: https://github.com/Xilinx/Vitis-AI-Tutorials/tree/CIFAR10-Classification-with-TensorFlow
Ref: https://beetlebox.org/category/tutorials/

- Environment Setting
- Training
- Freeze Model
- Evaluate Frozen Graph
- Quantize
- Evaluate Quantized Graph
- Compile

Docker Container

VITIS-AI AI IMPLEMENTATION

Ref: https://github.com/Xilinx/Vitis-AI-Tutorials/tree/CIFAR10-Classification-with-TensorFlow
Ref: https://beetlebox.org/category/tutorials/

- Environment Setting
- Training
- Freeze Model
- Evaluate Frozen Graph
- Quantize
- Evaluate Quantized Graph
- Compile

Docker Container

VITIS-AI AI IMPLEMENTATION

Ref: https://github.com/Xilinx/Vitis-AI-Tutorials/tree/CIFAR10-Classification-with-TensorFlow
Ref: https://beetlebox.org/category/tutorials/

- Environment Setting
- Training
- Freeze Model
- Evaluate Frozen Graph
- Quantize
- Evaluate Quantized Graph
- Compile

Docker Container

VITIS-AI AI IMPLEMENTATION

Ref: https://github.com/Xilinx/Vitis-AI-Tutorials/tree/CIFAR10-Classification-with-TensorFlow
Ref: https://beetlebox.org/category/tutorials/

```
(vitis-ai-tensorflow) syo@SYO:/workspace$
```

/workspace/object_detection/

```
|-- img
|-- log
|-- output
|-- quantize_results
|-- target
|-- calib_list.txt
|-- compile.sh
|-- input_fn.py
|-- quantize_recipe.sh
`-- model.pb
```

Docker Container

VITIS-AI AI IMPLEMENTATION

Ref: https://github.com/Xilinx/Vitis-AI-Tutorials/tree/CIFAR10-Classification-with-TensorFlow
Ref: https://beetlebox.org/category/tutorials/

```
(vitis-ai-tensorflow) syo@SYO:/workspace$
```

/workspace/object_detection/

|-- **img** : **png images for calibration**
|-- **log** : **log files**
|-- **output** : **output of compiler, DPU executable files**
|-- **quantize_results** : **output of quantizer**
|-- **target** : **application and DPU executable files**
|-- calib_list.txt : **list of images for calibration** *ls img > calib_list.txt
|-- compile.sh : **bash script for compiling**
|-- input_fn.py : **calibration function for quantization**
|-- quantize_recipe.sh : **bash script for quantization**
`-- **model.pb** : **frozen model**

VITIS-AI AI IMPLEMENTATION

Ref: https://github.com/Xilinx/Vitis-AI-Tutorials/tree/CIFAR10-Classification-with-TensorFlow
Ref: https://beetlebox.org/category/tutorials/

- Quantize

quantize_recipe.sh

```
vai_q_tensorflow quantize \                    : vitis-ai command
    --input_frozen_graph model.pb \            : frozen model
    --input_nodes normalized_gray_image \
    --input_shapes ?,50,50,1 \
    --output_nodes  final_output \
    --input_fn input_fn.calib_input \          : input_fn.py
    --method 0 \
    --gpu 0 \
    --calib_iter 30 \
    --output_dir ./quantize_results \          : output directory
    --weight_bit 8 \                           : from 32-bit to 8-bit
    --activation_bit 8 \
```

Docker Container

VITIS-AI AI IMPLEMENTATION

Ref: https://github.com/Xilinx/Vitis-AI-Tutorials/tree/CIFAR10-Classification-with-TensorFlow
Ref: https://beetlebox.org/category/tutorials/

- Quantize

quantize_recipe.sh

```
vai_q_tensorflow quantize \
    --input_frozen_graph model.pb \
    --input_nodes normalized_gray_image \
    --input_shapes ?,50,50,1 \
    --output_nodes  final_output \
    --input_fn input_fn.calib_input \
    --method 0 \
    --gpu 0 \
    --calib_iter 30 \
    --output_dir ./quantize_results \
    --weight_bit 8 \
    --activation_bit 8 \
```

Already determined in the frozen model

Docker Container

VITIS-AI AI IMPLEMENTATION

Ref: https://github.com/Xilinx/Vitis-AI-Tutorials/tree/CIFAR10-Classification-with-TensorFlow
Ref: https://beetlebox.org/category/tutorials/

- Quantize

quantize_recipe.sh

```
(vitis-ai-tensorflow) syo@SYO:/workspace/object_detection$ sh quantize_recipe.sh
INFO: Checking Float Graph...
INFO: Float Graph Check Done.
INFO: Calibrating for 30 iterations...
100% (30 of 30) |############################| Elapsed Time: 0:00:01 Time:  0:00:01
INFO: Calibration Done.
INFO: Generating Deploy Model...
INFO: Deploy Model Generated.
******************** Quantization Summary ********************
INFO: Output:
  quantize_eval_model: ./quantize_results/quantize_eval_model.pb
  deploy_model: ./quantize_results/deploy_model.pb
```

VITIS-AI AI IMPLEMENTATION

Ref: https://github.com/Xilinx/Vitis-AI-Tutorials/tree/CIFAR10-Classification-with-TensorFlow
Ref: https://beetlebox.org/category/tutorials/

- Compile

compile.sh

```
compile() {
vai_c_tensorflow \
    --frozen_pb ./quantize_results/deploy_model.pb \
    --arch /opt/vitis_ai/compiler/arch/DPUCZDX8G/ZCU102/arch.json \
    --output_dir ./output/ \
    --net_name simple_net \
    --options "{'mode':'normal'}"
}

compile | tee ./log/compile_log_zcu102
```

Docker Container

VITIS-AI AI IMPLEMENTATION

Ref: https://github.com/Xilinx/Vitis-AI-Tutoria
Ref: https://beetlebox.org/category/tutorials/

- Compile

compile.sh

```
(vitis-ai-tensorflow) syo@SYO:/workspace/object_detection$ sh compile.sh

Kernel topology "simple_net_kernel_graph.jpg" for network "simple_net"
kernel list info for network "simple_net"
                               Kernel ID : Name
                                       0 : simple_net

                             Kernel Name : simple_net
--------------------------------------------------------------------------------
                             Kernel Type : DPUKernel
                               Code Size : 0.01MB
                              Param Size : 3.91MB
                            Workload MACs : 109.72MOPS
                          IO Memory Space : 0.03MB
                               Mean Value : 0, 0, 0,
                        Total Tensor Count : 5
                  Boundary Input Tensor(s)   (H*W*C)
               normalized_gray_image:0(0) : 50*50*1

                 Boundary Output Tensor(s)   (H*W*C)
                         final_output:0(0) : 1*1*1

                          Total Node Count : 4
                             Input Node(s)   (H*W*C)
                                 Conv2D(0) : 50*50*1

                            Output Node(s)   (H*W*C)
                               MatMul_1(0) : 1*1*1



**********************************************
* VITIS_AI Compilation - Xilinx Inc.
**********************************************
```

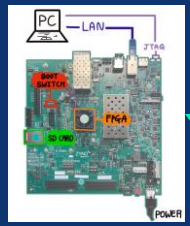
Docker Container

VITIS-AI AI IMPLEMENTATION

Ref: https://github.com/Xilinx/Vitis-AI-Tutorials/tree/CIFAR10-Classification-with-TensorFlow
Ref: https://beetlebox.org/category/tutorials/

- File Transfer

```
/object_detection/target/
|-- dpuv2_rundir
|-- images
|-- single_image
|-- app.py
|-- compile_so.sh
|-- dpu_simple_net.elf
|-- runner.py
`-- simple_net_kernel_graph.gv
```



target.tar.gz

```
scp target.tar.gz root@192.168.8.125
```

Docker Container

VITIS-AI AI IMPLEMENTATION

- Image Feeding

https://github.com/YooSunYoung/binary_classification_vitis_ai_tutorial/blob/master/image_feeder.py

Feed images Via Socket Communication

PC

VITIS-AI AI IMPLEMENTATION

Ref: https://github.com/Xilinx/Vitis-AI-Tutorials/tree/CIFAR10-Classification-with-TensorFlow
Ref: https://beetlebox.org/category/tutorials/

```
root@xilinx-zcu102-2020_1:~/target# tar -xzvf target.tar.gz
```

- Compile

```
root@xilinx-zcu102-2020_1:~/target# sh compile_so.sh
root@xilinx-zcu102-2020_1:~/target# ls dpuv2_rundir/
libdpumodelsimple_net.so  meta.json
```

VITIS-AI AI IMPLEMENTATION

Ref: https://github.com/Xilinx/Vitis-AI-Tutorials/tree/CIFAR10-Classification-with-TensorFlow
Ref: https://beetlebox.org/category/tutorials/

- Runner.py

```
class Runner:
  # tensor format enum
  TensorFormat = type('', (), {})()
  TensorFormat.NCHW = 0
  TensorFormat.NHWC = 1

  def __init__(self, path):
    metaFile = os.path.join(path, "meta.json")
    if not os.path.isfile(metaFile):
      raise AssertionError("meta.json file %s not found" % metaFile)

    # select .so file based on path/meta.json
    with open(metaFile) as f:
      meta = json.load(f)
      libFile = self._parse_path(meta['lib'])

    if not libFile or not os.path.isfile(libFile):
      raise AssertionError("C++ library .so file %s not found" % libFile)

    self._libFile = os.path.abspath(libFile)
    self._lib = cdll.LoadLibrary(self._libFile)

    self._lib.DpuPyRunnerCreate.argtypes = [c_char_p]
    self._lib.DpuPyRunnerCreate.restype = c_void_p
    self._lib.DpuPyRunnerGetInputTensors.argtypes = [c_void_p,
      POINTER(c_void_p), POINTER(c_int)]
    self._lib.DpuPyRunnerGetOutputTensors.argtypes = [c_void_p,
      POINTER(c_void_p), POINTER(c_int)]
    self._lib.DpuPyRunnerGetTensorFormat.argtypes = [c_void_p]
    self._lib.DpuPyRunnerGetTensorFormat.restype = c_int
    self._lib.DpuPyRunnerExecuteAsync.argtypes = [c_void_p,
      POINTER(np.ctypeslib.ndpointer(c_float, flags="C_CONTIGUOUS")),
      POINTER(np.ctypeslib.ndpointer(c_float, flags="C_CONTIGUOUS")),
      c_int, POINTER(c_int)]
    self._lib.DpuPyRunnerExecuteAsync.restype = c_int
    self._lib.DpuPyRunnerWait.argtypes = [c_void_p, c_int]
    self._lib.DpuPyRunnerWait.restype = c_int
    self._lib.DpuPyRunnerDestroy.argtypes = [c_void_p]

    self._runner = self._lib.DpuPyRunnerCreate(path.encode('utf-8'))
```

DPU library wrapper

FPGA Board

VITIS-AI AI IMPLEMENTATION

Ref: https://github.com/Xilinx/Vitis-AI-Tutorials/tree/CIFAR10-Classification-with-TensorFlow
Ref: https://beetlebox.org/category/tutorials/

- Runner.py

```
def get_input_tensors(self):
  ptr = c_void_p()
  n = c_int(0)
  self._lib.DpuPyRunnerGetInputTensors(self._runner, byref(ptr), byref(n))
  tensors = []
  for i in range(n.value):
    tensors.append(Tensor.from_address(ptr.value + (i*sizeof(Tensor))))
  return tensors

def get_output_tensors(self):
  ptr = c_void_p()
  n = c_int(0)
  self._lib.DpuPyRunnerGetOutputTensors(self._runner, byref(ptr), byref(n)
  tensors = []
  for i in range(n.value):
    tensors.append(Tensor.from_address(ptr.value + (i*sizeof(Tensor))))
  return tensors
```

VITIS-AI AI IMPLEMENTATION

Ref: https://github.com/Xilinx/Vitis-AI-Tutorials/tree/CIFAR10-Classification-with-TensorFlow
Ref: https://beetlebox.org/category/tutorials/

- Runner.py

```
def execute_async(self, inputs, outputs):
  """
    Args:
      inputs: list of numpy arrays
      outputs: list of numpy arrays

      order of numpy arrays in inputs/outputs must match
        the order in get_input_tensors() and get_output_tensors()
  """
  status = c_int(0)
  ret = self._lib.DpuPyRunnerExecuteAsync(self._runner,
                    self._numpy_list_2_cptr_list(inputs),
    self._numpy_list_2_cptr_list(outputs),
    inputs[0].shape[0], byref(status))

  if status.value != 0:
    raise RuntimeError("Runner.execute_async could not enqueue new DPU job")

  return ret
```

FPGA Board

VITIS-AI AI IMPLEMENTATION

Ref: https://github.com/Xilinx/Vitis-AI-Tutorials/tree/CIFAR10-Classification-with-TensorFlow
Ref: https://beetlebox.org/category/tutorials/

- App.py

  https://github.com/YooSunYoung/binary_classification_vitis_ai_tutorial/blob/master/object_detection/target/app.py

06

DEMONSTRATION

THANKS!

Do you have any questions?

syo@mmmi.sdu.dk