# PYNQ是你從未聽說過最酷的Python框架

林明憲(Alex Lin)

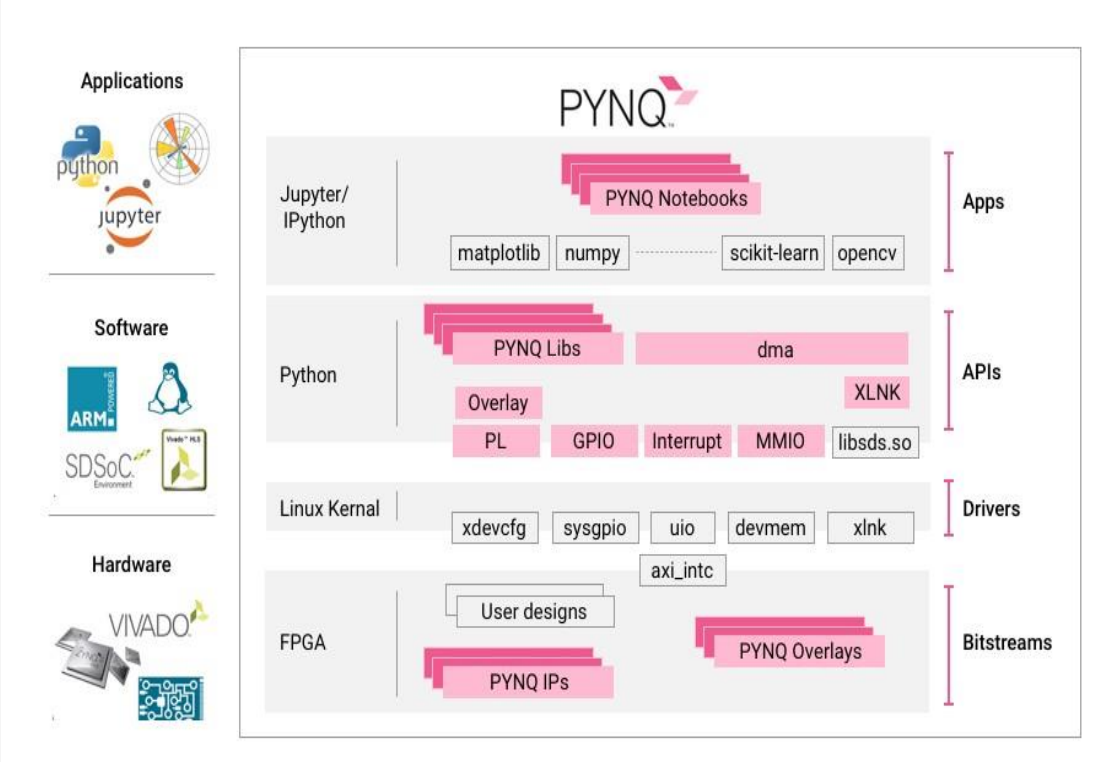
linminghsien11@gmail.com

# Agenda

- Introduction to The PYNQ Framework
  - What is PYNQ
  - Who is PYNQ for
  - What is the difference
- Exercise: Face & Emotion Recognition
  - Environments
  - Deployment
- Tutorials and Community Projects
- Summary

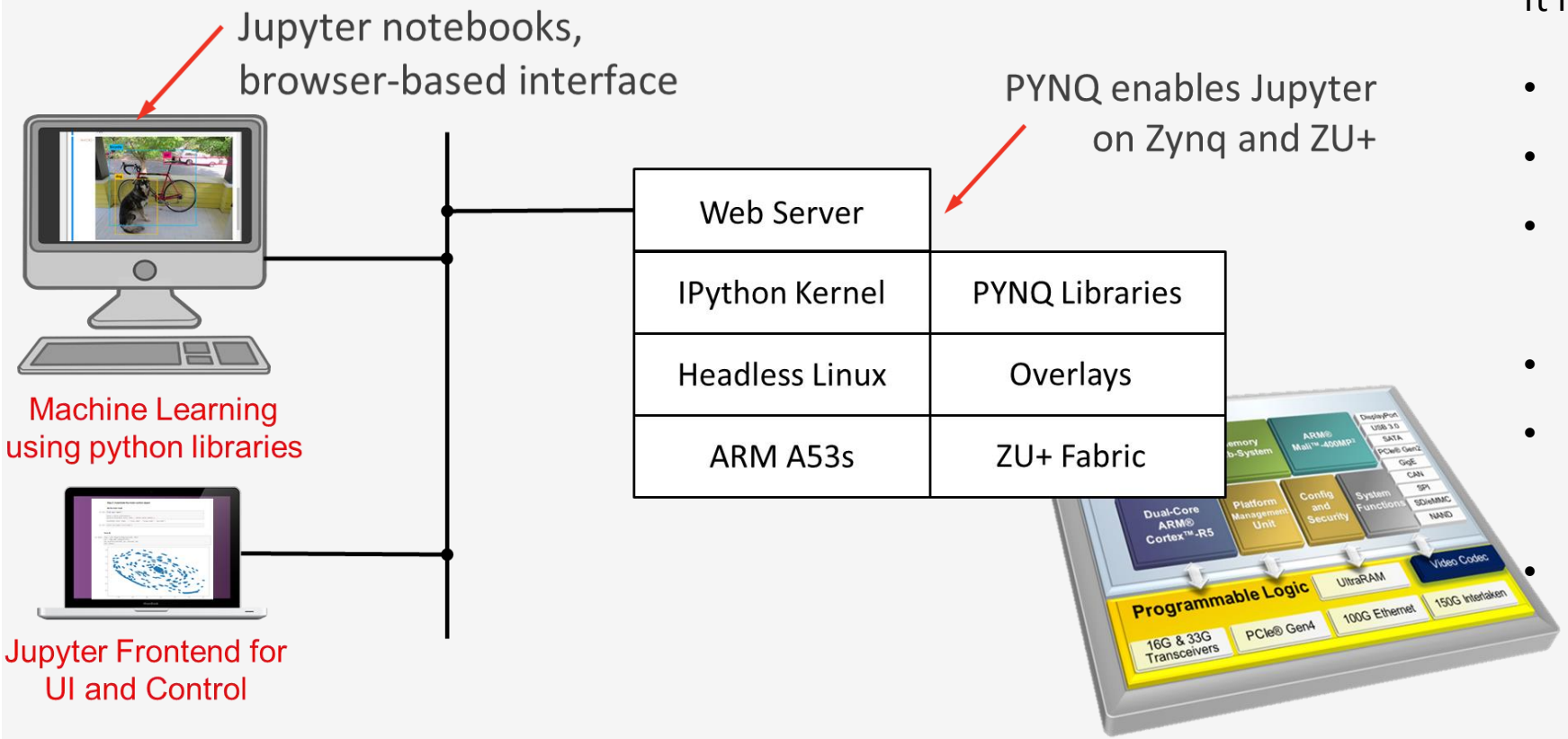
# What is PYNQ

- PYNQ is an open-source project from Xilinx® that makes it easier to use Xilinx platforms. Using the Python language and libraries, designers can exploit the benefits of programmable logic and microprocessors to build more capable and exciting electronic systems.
- PYNQ can be used with *Zynq*, *Zynq UltraScale+*, *Zynq RFSoC*, *Alveo* accelerator boards and AWS-F1 to create high performance applications with:
  - Parallel hardware execution
  - High frame-rate video processing
  - Hardware accelerated algorithms
  - Real-time signal processing
  - High bandwidth IO
  - Low latency control




ZYNQ Zynq-7000 ZYNQ UltraSCALE+ ZYNQ RFSoC XILINX ALVEO

# PYNQ Basics

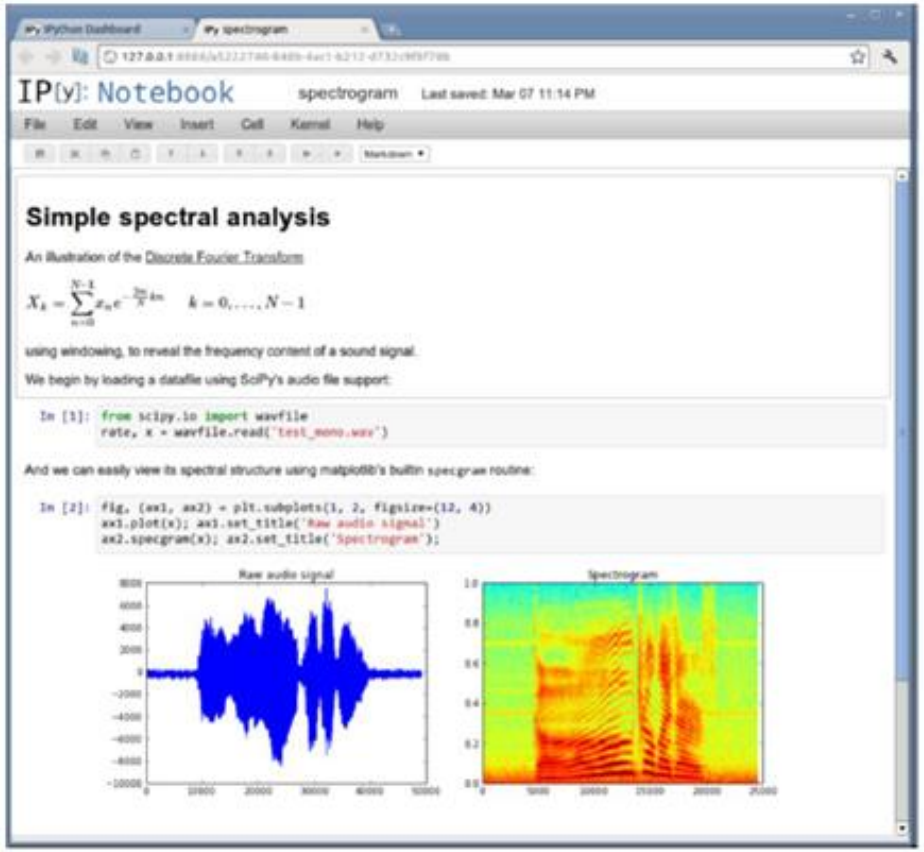



It is comprised of:

- Petalinux(Linux Kernel)
- Ubuntu Bionic root file-system
- Full Python(as opposed to Micro Python)
- Jupyter Notebooks
- Python libraries for using the Xilinx PS and PL
- Overlays(aka Hardware Libraries)

**Why would I want to use it?** Has the ability to make some of your slow Python programs run FAST, really really FAST and allows Python to control hardware that other platforms could only dream about. It can also dramatically reduce design time and effort!

# Jupyter Notebooks: browser-based development … with rich, multi-media support

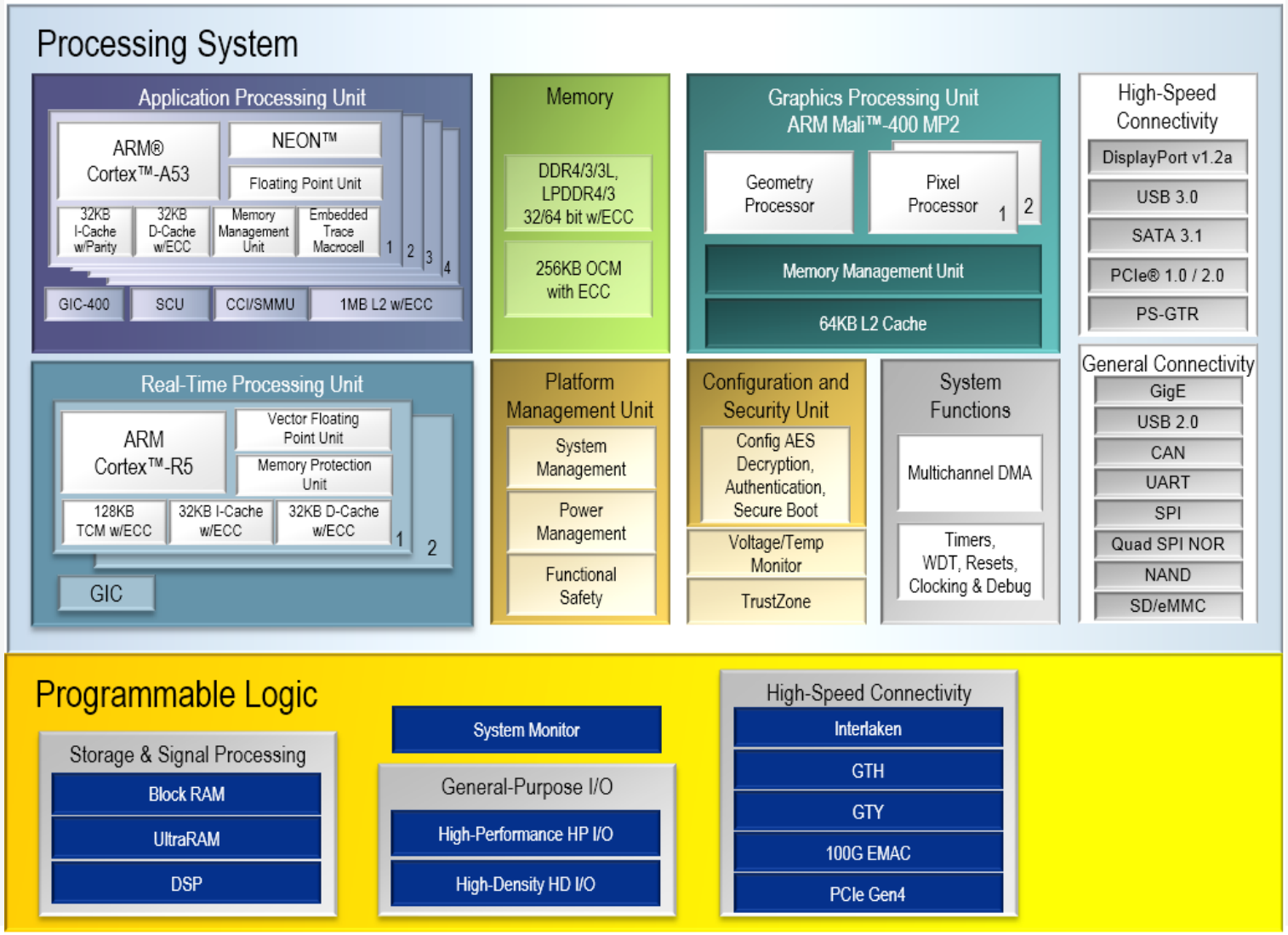



- Designed for
  - Interactive, exploratory computing
  - Reproducible results
- Ideal for
  - Teaching and learning
  - Projects and research
- Rapid adoption
  - Across multiple disciplines

github.com/ipython/ipython/wiki/A-gallery-of-interesting-IPython-Notebooks

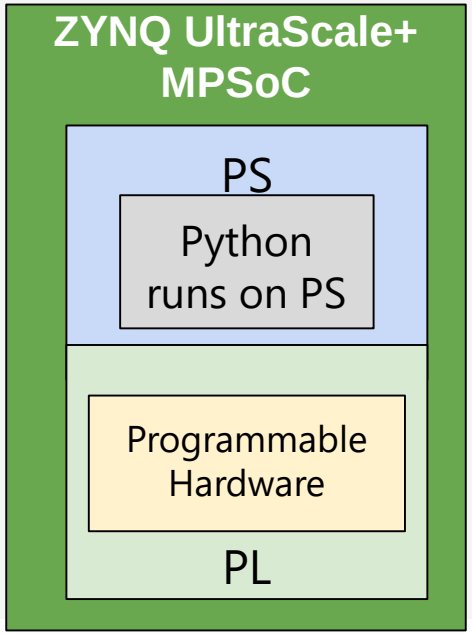
**Part of a broader trend towards new, web-based IDEs**

# PS + PL = APSoC(All programmable SoC)




**PS** = Processing System

**PL** = Programmable Logic(FPGA)



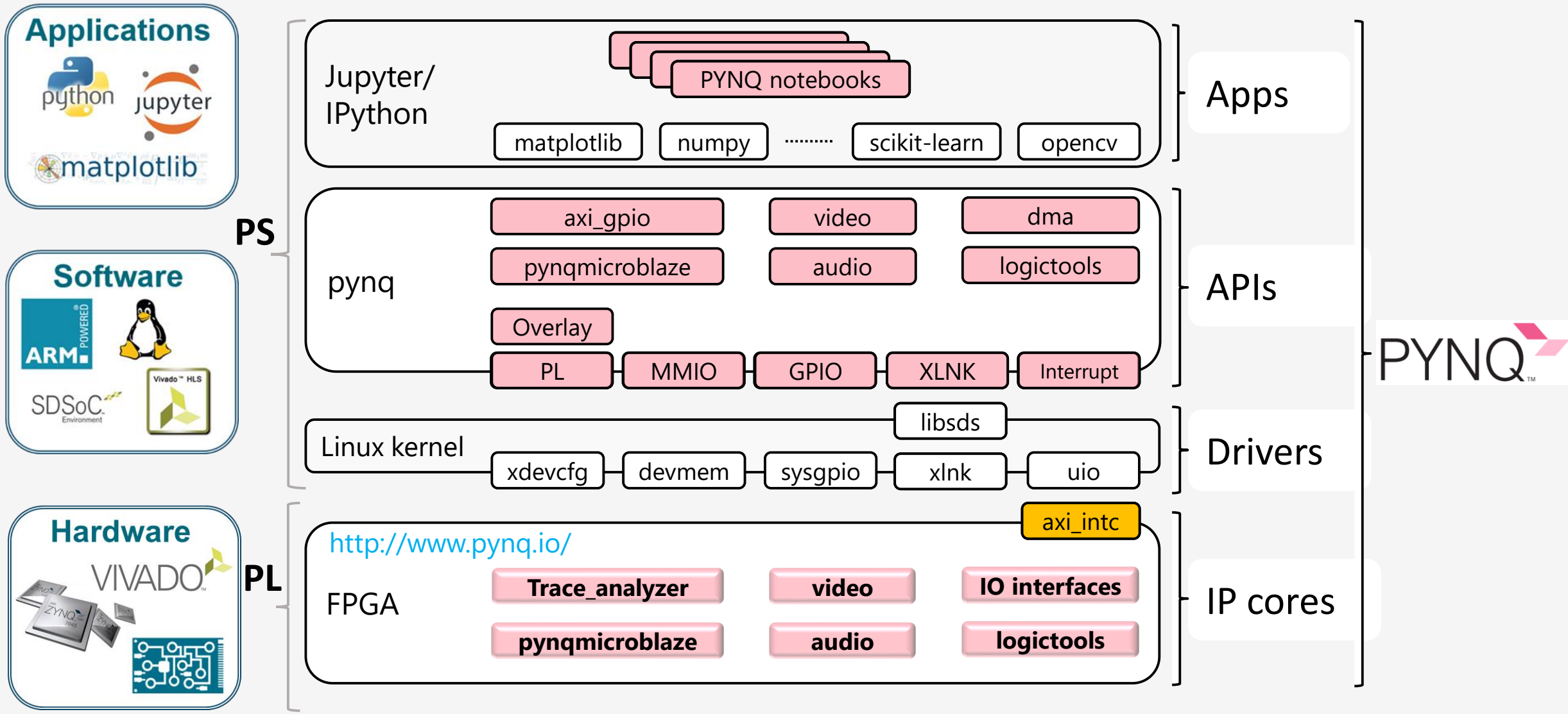

Overlays

# Introduction to Overlays (key differentiation for YOU)

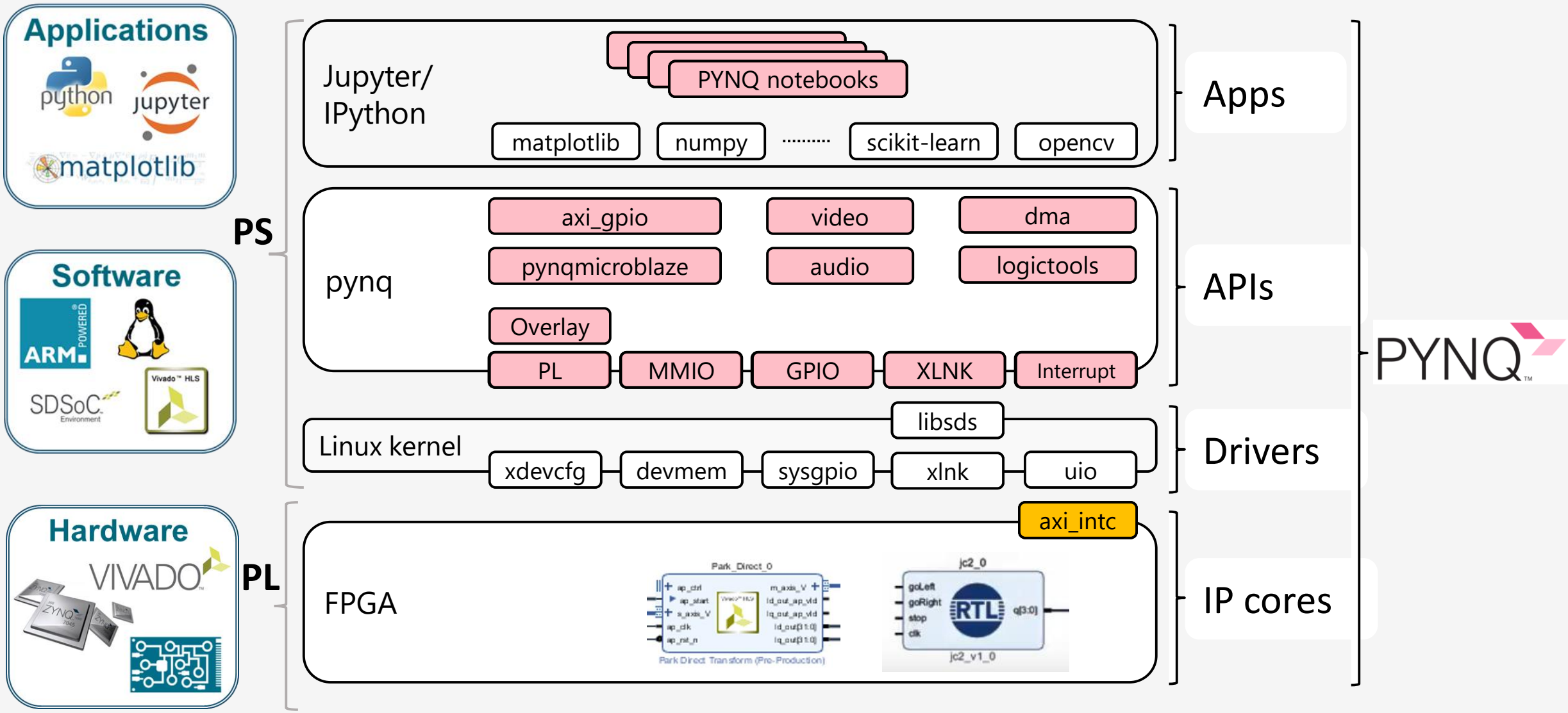
**What is an Overlay?**
Overlays are FPGA designs, using the design pattern concept, that extend user application from processing system to programmable logic, efficiently pre-processing the data prior to interaction by Arm Processor


- The .bit and .tcl files are generated by using Xilinx Tools
- Files are stored in the PYNQ overlay library
- PYNQ parses the files & recognizes the custom overlay
- Write custom Jupyter notebooks or scripts to interact

# Inside PYNQ – *the "base overlay"*



# Inside PYNQ – *Your overlay*

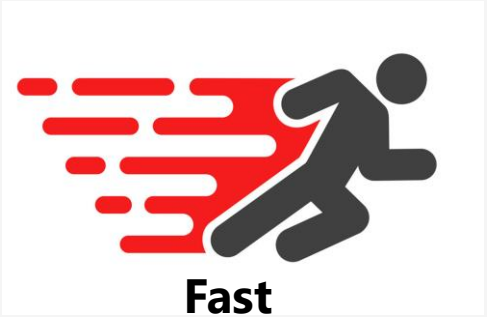


# PYNQ – Who is it for?

1. Interested in edge computing, IoT, embedded systems, custom designed hardware or signal processing?
2. Familiar with Python and want to interact with Xilinx Ultrascale+ Programmable Logic?
3. Need to accelerate embedded Python for machine learning inference or any other edge or data science app?
4. Looking for a way to communicate technical information through computational notebooks?
5. Need an embedded Linux rapid prototyping platform for Python and beyond?
6. Already familiar with Raspberry Pi™?

# What is the difference

- Embedded system with Full Python
- Scalable Hardware Platform
  - Cortex-A9, Cortex-A53, RFSoC (Cortex-A53+RF), Alveo (x86 + FPGA)
- Hardware and Software Codesign Platform
- Software Defined x Platform
  - x: Networking, Radio, Hardware, storage … etc
- Full stack development Platform from App to IC design
- Fast



# Use case – Machine Learning computer vision on Ultra96

FINN Binary Neural Network (BNN) Demo on Ultra96

- http://www.wiki.xilinx.com/Zynq+UltraScale%EF%BC%8B+MPSoC+Accelerated+Image+Classification+via+Binary+Neural+Network+TechTip

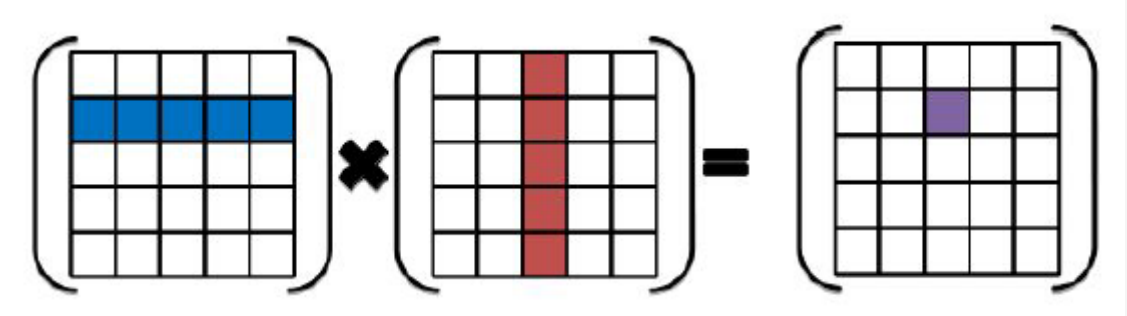

Full 1080p Images per Second in HW: 66.3
Full 1080p Images per Second in *SW: .01

**HW Acceleration Factor: *6171**

* The SW used for benchmark was running on the Ultra96 ARM Cortex™ A53 cores with same OS as the HW tests @ ~1.3GHz. Other platforms that have somewhat faster ARM cores could do a little better with just SW. Other platforms with their own hardware accelerators will also run faster than pure SW.

# Use case - Matrix Multiply Algorithm Acceleration

- 32x32 Matrix Multiply of floats
- Algorithm developed in C
- Then accelerated in programmable logic
- Python can make calls into the C code for this

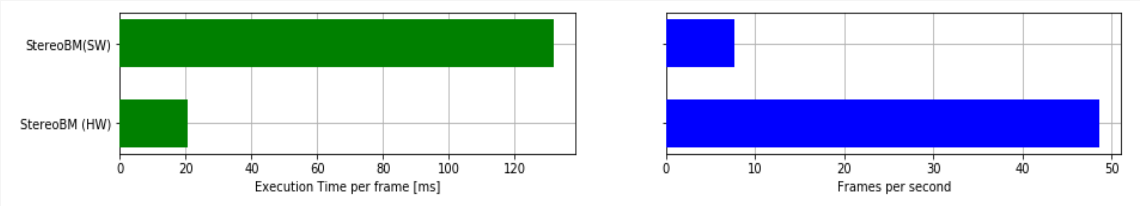


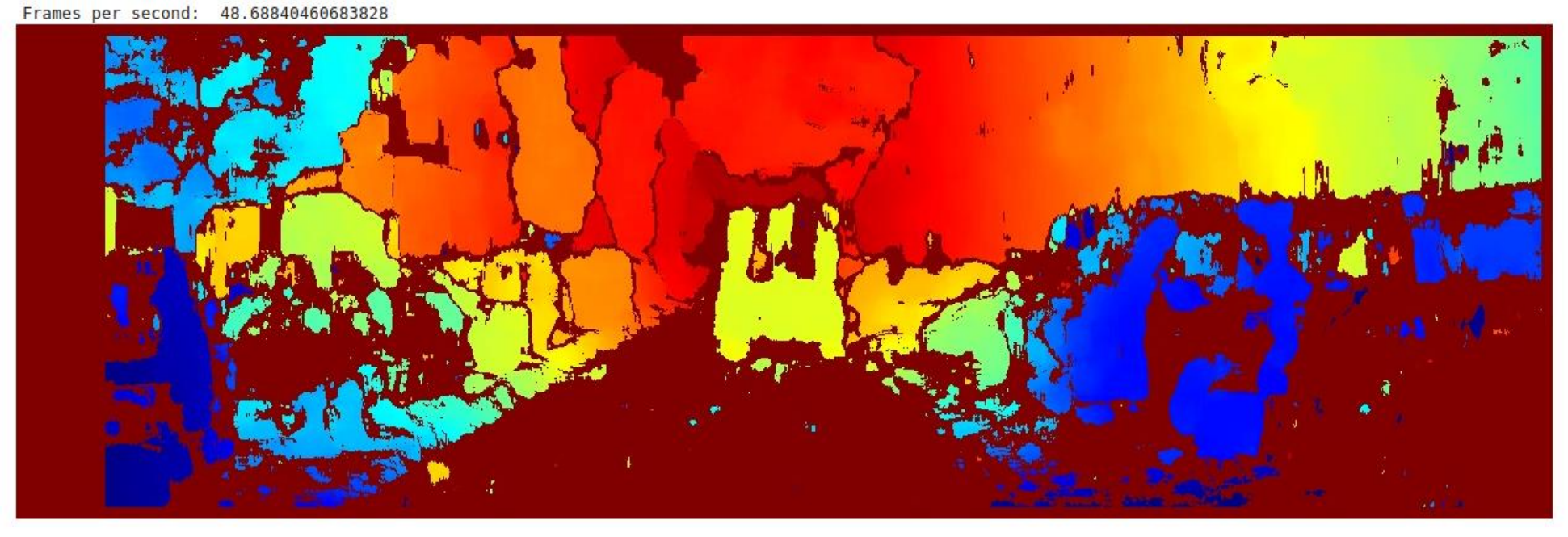
| Processor-only Cycles | Accelerated Cycles | Acceleration |
|---|---|---|
| 1578615 | 65725 | 24x |

**Daniel Rozwood** -- *Ultra96 SDSoC Platform for v2018.2*

# Use case - OpenCV Algorithm Acceleration with stereo BM

- OpenCV library : Stereo Block Matching
- https://github.com/Xilinx/PYNQ-ComputerVision.git
- HW 49 vs SW 8 images per second!



# Programmable Logic Acceleration vs. CPU/GPU

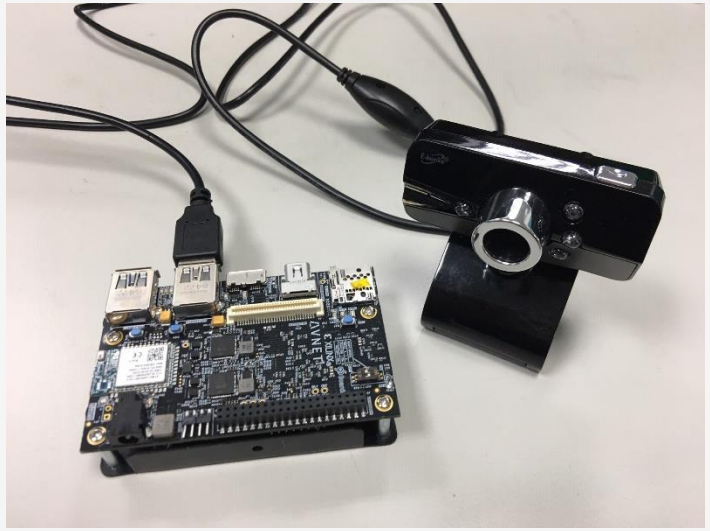
| Domain / Topic | Title /Author /DOI | Improvement vs CPU+GPU | Improvement vs CPU-Only |
|---|---|---|---|
| **Digital Signal Processing** Sliding Windows | A Performance and Energy Comparison of FPGAs, GPUs, and Multicores for Sliding Window Applications, Fowers, http://dx.doi.org/10.1145/2145694.2145704 | **11x** | **57x** |
| **Graph Processing** Tree-reweighted Message Passing (TRW-S) | GraphGen for CoRAM: Graph Computation on FPGAs, Weisz, http://dx.doi.org/10.1109/FCCM.2014.15 | **10.3x** | **14.5x** |
| **Monte Carlo Simulation** Random Number Generation | A Comparison of CPUs, GPUs, FPGAs, and Massively Parallel Processor Arrays for Random Number Generation, Thomas, http://dx.doi.org/10.1145/1508128.1508139 | **3x** | **30x** |
| **Machine Vision** Moving Average with Local Difference (MALD) | CPU, GPU and FPGA Implementations of MALD: Ceramic Tile Surface Defects Detection Algorithm, Hocenski, http://dx.doi.org/10.7305/automatika.2014.01.317 | **14x** | **35x** |
| **Bioinformatics** *De Novo* Genome Assembly | Hardware Accelerated Novel Optical *De Novo* Assembly for Large-Scale Genomes, Kastner, http://dx.doi.org/10.1109/FPL.2014.6927499 | **8.5x** | **11.9x** |
| **Atmospheric Modelling** Solvers for Global Atmospheric Equations | Accelerating Solvers for Global Atmospheric Equations through Mixed-Precision Data Flow Engine, Gan, http://dx.doi.org/10.1109/FPL.2013.6645508 | **4x** | **100x** |

# Exercise: Face & Emotion Recognition

Refer to : https://www.hackster.io/dnhkng/caffein-ai-tor-711e4e

Hardware Components

- Avnet Ultra96
- Webcam



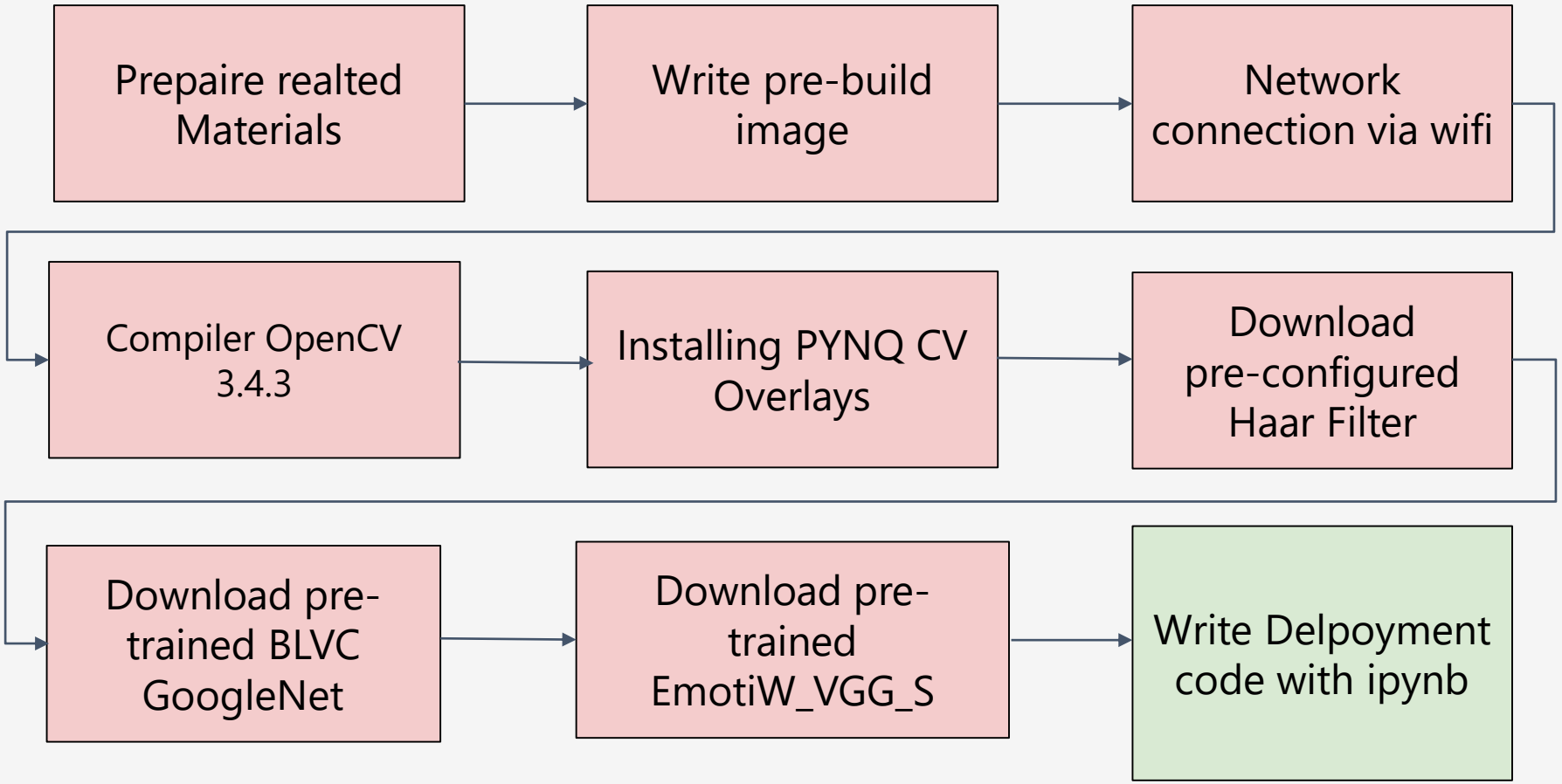
Software Apps

- Xilinx PYNQ
- SciKit-Learn
- OpenCV

Adopted technology

- OpenCV : Laplacian Kernel(Blur Detection), Haar Cascade(Face Detetion)
- Deep Learning: Facenet(Face Recognition), Emonet(Emotion Detection)
- Decision tree : Random Forests

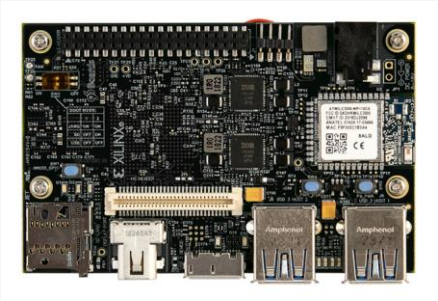
# How to Start


Prepaire realted Materials
Write pre-build image
Network connection via wifi
Compiler OpenCV 3.4.3
Installing PYNQ CV Overlays
Download pre-configured Haar Filter
Download pre-trained BLVC GoogleNet
Download pre-trained EmotiW_VGG_S
Write Delpoyment code with ipynb

# Required materials

Micro SD card firmware image
(urls provided on subsequent slides)

Ultra96 v2 (or v1)



Micro B USB cable: 2.0 (or 3.0)

Micro SD card **>= 16GB** (the faster the better but no min speed) Delkin, Sandisk, others? (Samsung EVO won't work)


balenaEtcher

(or Win32DiskImager)

Host PC, Laptop:
Windows 10, Ubuntu Linux 16/18 LTS, Mac, Chromebook

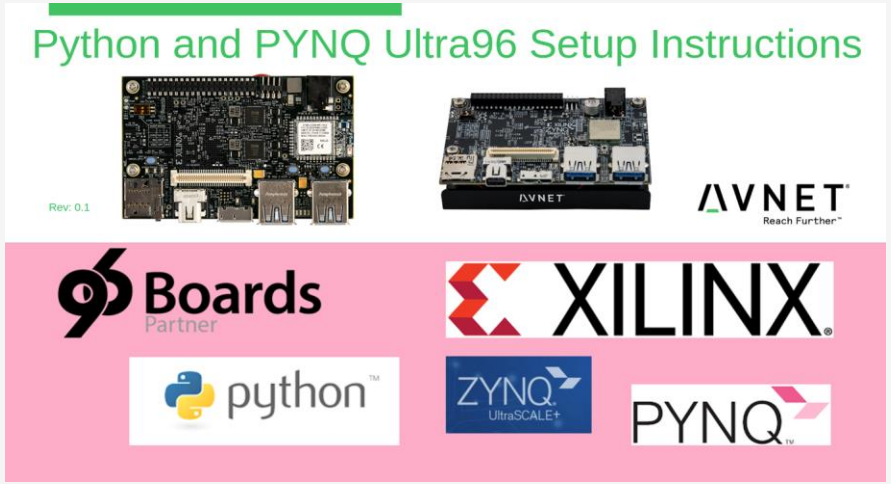
Microsoft ubuntu

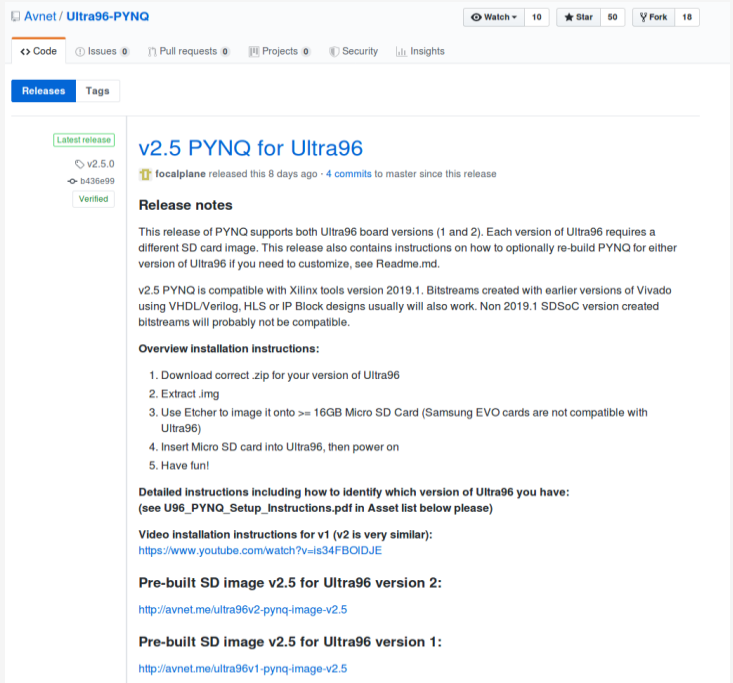
With compatible browser: Firefox, Chrome or Safari

96boards **12V 4A** international supply!

# PYNQ Install and Step

- To proceed PYNQ v2.5 must already be installed and running on the Ultra96
- If not, obtain the v2.5  release from: https://github.com/Avnet/Ultra96-PYNQ/releases
- Follow the setup instructions here (or use the latest under Github releases): https://github.com/Avnet/Ultra96-PYNQ/releases/download/v2.5.0/U96_PYNQ_Setup_Instructions.pdf



# Next.....

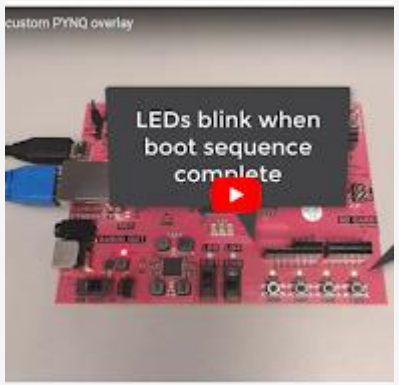
- Network connection via: http://192.168.2.1:9090/notebooks/common/wifi.ipynb
- OpenCV 3.4.2 : https://github.com/opencv/opencv/archive/3.4.3.zip
- PYNQ CV Overlays : https://github.com/Xilinx/PYNQ-ComputerVision.git
- pre-configured Haar Filter: https://github.com/opencv/opencv/blob/master/data/haarcascades/haarcascade_frontalface_default.xml
- pre-trained BLVC GoogleNet model : http://dl.caffe.berkeleyvision.org/bvlc_googlenet.caffemodel
- pre-trained EmotiW_VGG_S model: https://drive.google.com/open?id=0BydFau0VP3XSNVYtWnNPMU1TOGM

# Results

# Tutorials and other resources

PYNQ Tutorials workshop

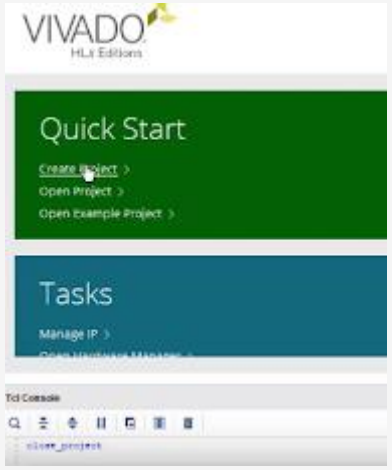

HLS filter example (FPGA Developer)



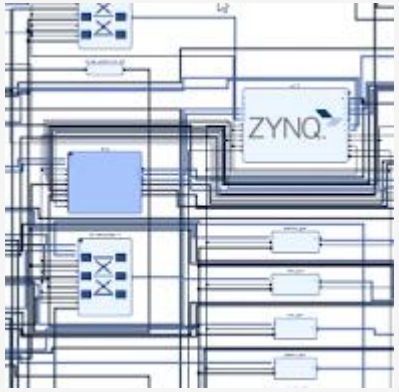

PYNQ HLS tutorial (Dustin Richmond)


Accelerate FIR function

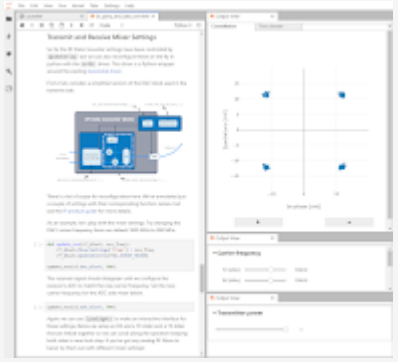

Video: Control custom IP using Vivado




Video: Add existing IP to a PYNQ overlay




PYNQ RFSoC tutorial workshop



PYNQ bootcamp (High school level)

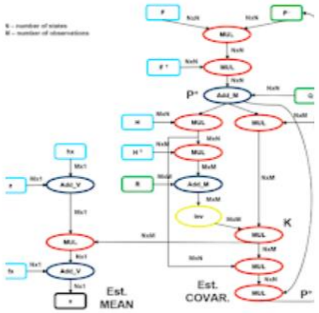
# PYNQ Community: more Application Notebooks

## PYNQ community projects

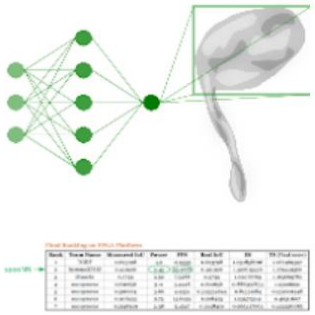
**PYNQ Hello World**
Hardware accelerated image resizer example

**RISC-V on PYNQ**
**UCSD**


**Extended Kalman filter**
**University Sydney**

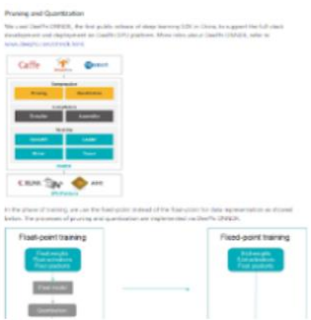



**spoonNN**
**ETH Zurich**
FPGA-based neural network inference project



**iSmart DNN**
FPGA-based neural network inference for DAC 2018 contest

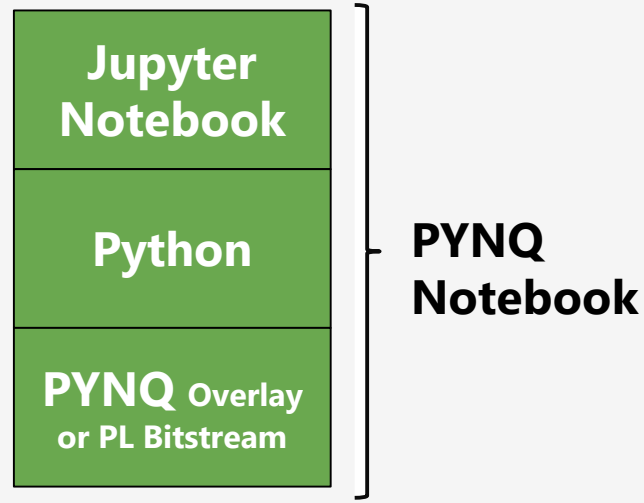

**TGIIF**
FPGA-based neural network inference for DAC 2018 contest



**cv2PYNQ**
**FAU**
Accelerated OpenCV image filtering library.

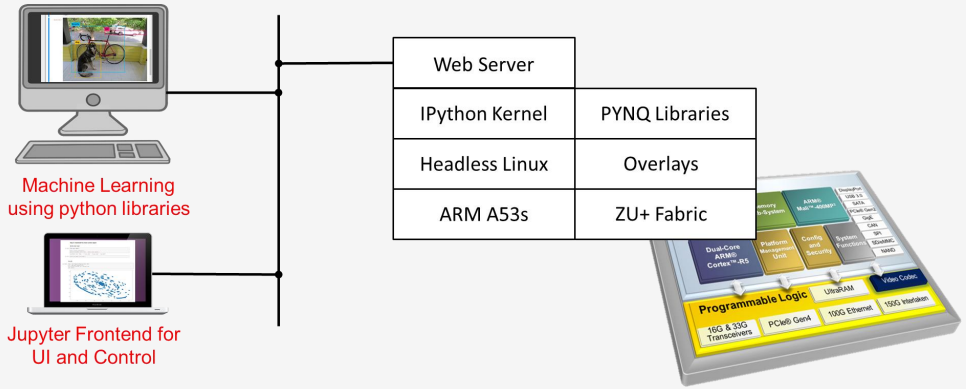

**Video processing**
**KU Leuven**
Hardware accelerated videoprocessing

http://www.pynq.io/community.html



# Summary

- PYNQ is a Productivity Framework
- PYNQ includes software, libraries & overlays
- Overlays are FPGA designs that extends user application from processing system to programmable logic
- Jupyter notebooks & python programming for quick exploration
- Developers can create custom overlays & package the project making available an hardware library to a wider audience



# Learning more: ZYNQ MPSoC and PYNQ

**Free e-version** or for purchase hard copy available from Amazon:

Exploring Zynq MPSoC with PYNQ and Machine Learning Applications

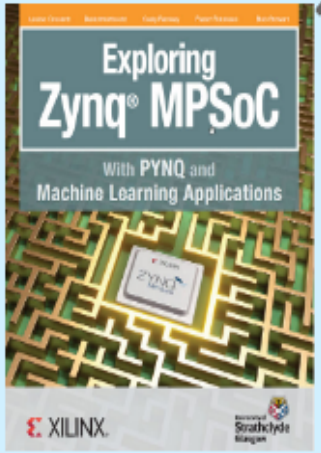
https://www.zynq-mpsoc-book.com

**Free eBook: Exploring Zynq MPSoC with PYNQ and Machine learning applications**

The book introduces **Zynq MPSoC** (Multi-Processor System-on-Chip) from Xilinx that combines an **Arm Cortex-A53** based processing systems, **Arm Cortex-R5** real-time processors, and **FPGA** programmable logic.

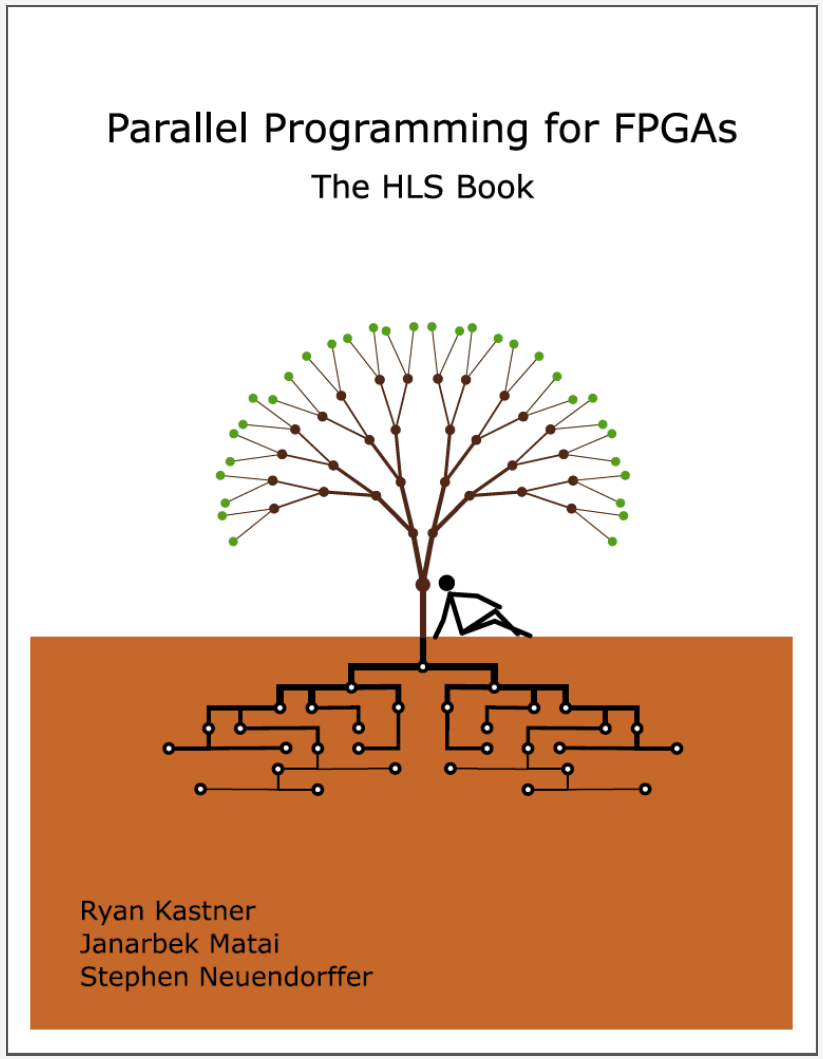
Topics include Zynq MPSoC **architecture**, **design tools**, hardware/software **co-design**, and **software-defined** methodologies. It features sepcial sections on **PYNQ** (Python-based framework for Zynq) and **machine learning**.

Download **free** from www.zynq-mpsoc-book.com or get a hard-copy on Amazon and other booksellers.




Close

# Learning more: Open source HLS book




Parallel Programming for FPGAs is an open-source book aimed at teaching hardware and software developers how to efficiently program FPGAs using high-level synthesis

http://kastner.ucsd.edu/hlsbook/

# Learning more: PYNQ

PYNQ main website:

https://pynq.io

Documentation:

https://pynq.readthedocs.io/

PYNQ support forums:

https://discuss.pynq.io

# Learning more: Ultra96 PYNQ

Github for Ultra96 PYNQ for customers who want to customize the OS:

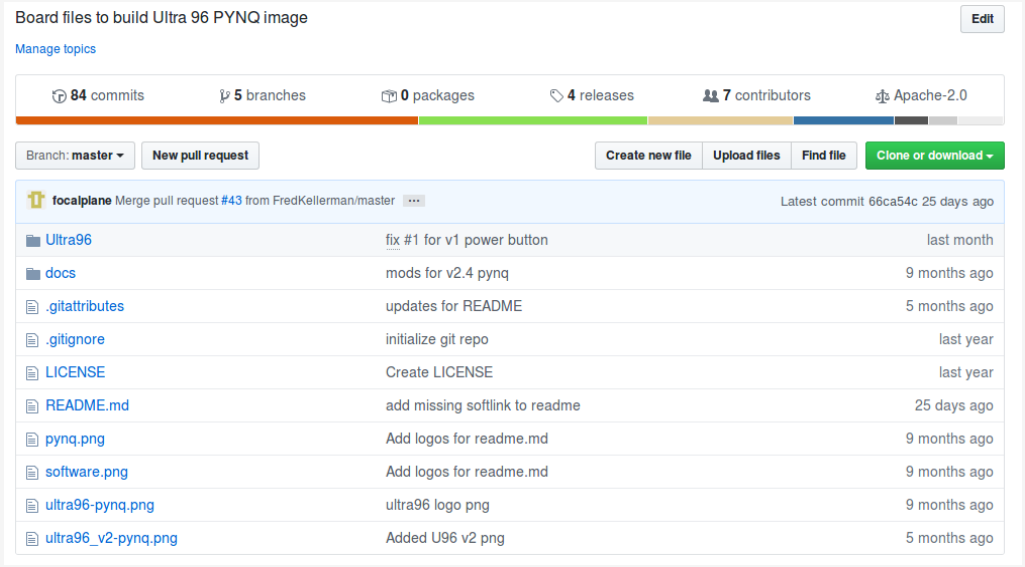
https://github.com/Avnet/Ultra96-PYNQ

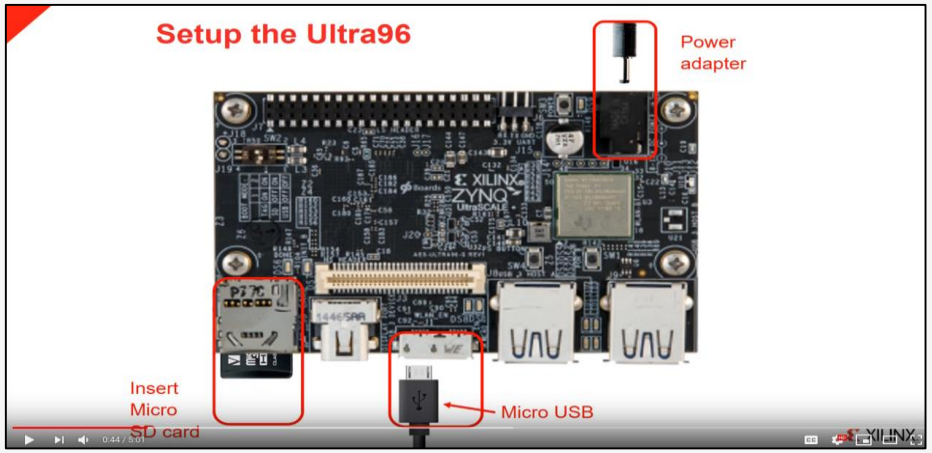
Documentation:

https://ultra96-pynq.readthedocs.io/

Ultra96 PYNQ setup installation video:

https://www.youtube.com/watch?time_continue=3&v=is34FBOlDJE



# Learning more: Xilinx hardware design

**Xilinx XUP on-line:**

- PYNQ Z1/Z2 (also mostly applicable to Ultra96) High Level Synthesis (create hardware designs with C/C++)
- Click prior year workshops for additional free materials

https://www.xilinx.com/support/university/vivado/vivado-workshops/Vivado-high-level-synthesis-flow-zynq.html

https://github.com/xupgit/High-Level-Synthesis-Flow-on-Zynq-using-Vivado-HLS

Vivado IP Block Design: https://github.com/xupgit/FPGA-Design-Flow-using-Vivado


README.md

Vivado FPGA Design Flow on Zynq

This workshop provides participants the necessary skills to develop digital design in Xilinx FPGA fabric and become familiar with synthesis, implementation, I/O planning, simulation, static timing analysis and debug features of Vivado.

The labs have been developed on a PC running Microsoft Windows 10 professional edition and using **Vivado 2018.2** version tools.These labs can also be run using WebPack edition.