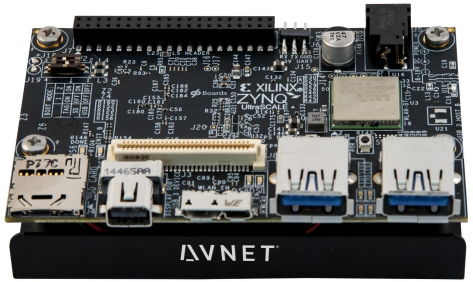
A Call to Action: Accelerating Python with FPGAs



AVNET
Reach Further™

96Boards Partner

PYNQ™

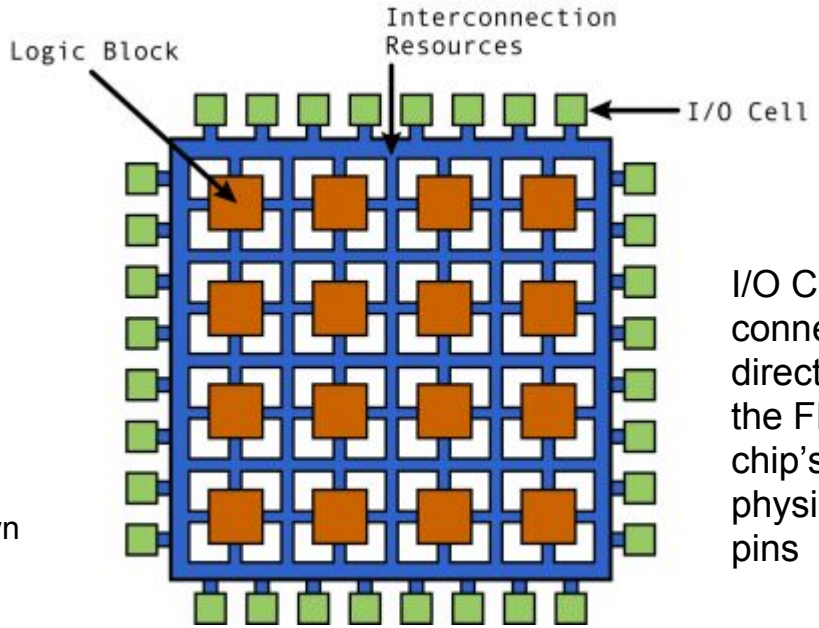
python™

ZYNQ UltraSCALE+

Intro to FPGAs

Field Programmable Gate Array (FPGA), under the hood:

The innards shown here are a simplification of what is inside the actual part

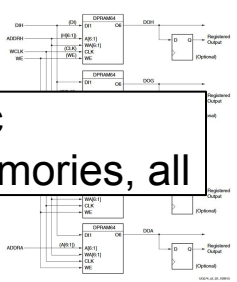


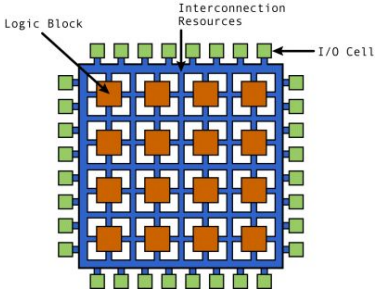
I/O Cells connect directly to the FPGA chip's physical pins

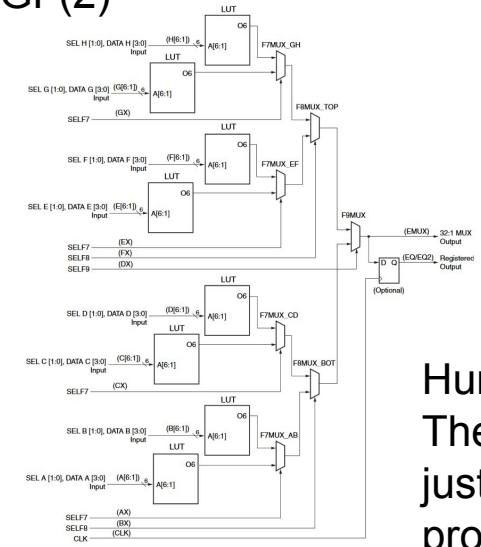
AVNET

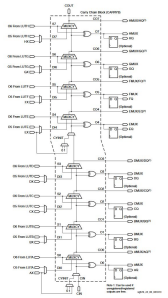
What's inside?

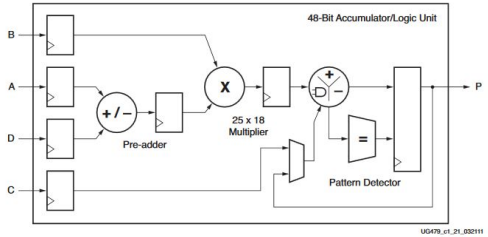
LUTs, Clocks, PLLs, Transceivers, Multiplexers, Logic Functions, Interconnects, Arithmetic operators and Memories, all GF(2)









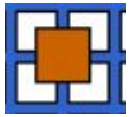


Hundreds of thousands of these programmable logic block widgets! They can be interconnected to create logic that can accomplish just about anything. A major part of the device programming involves place and route (the tools do this for you) which is how all the interconnects are setup to put it all together.

Note: logic in ZU3EG is not exactly as shown above but is similar

AVNET

Ultra96's Xilinx ZU3EG PL Internal Attributes:

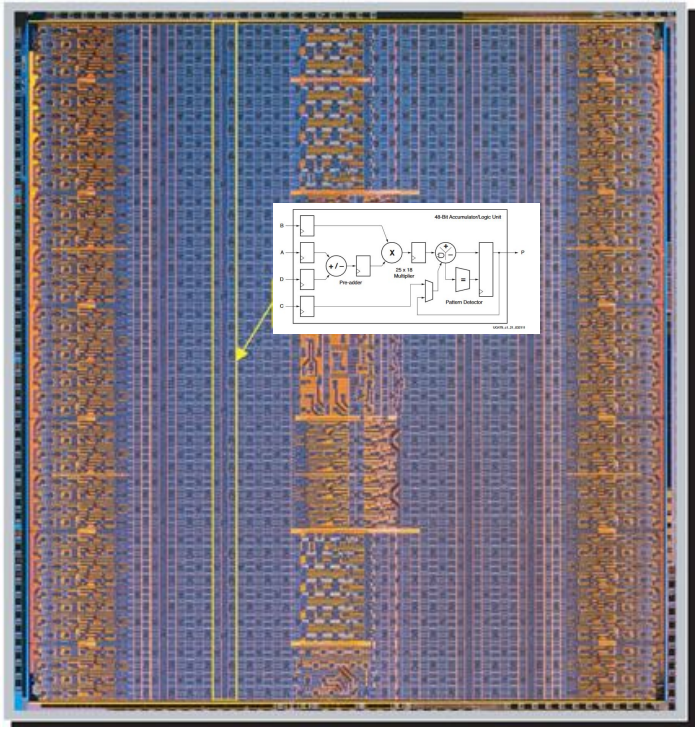


System Logic Cells	103,320	154,350
CLB Flip-Flops	94,464	141,120
CLB LUTs	47,232	70,560
Distributed RAM (Mb)	1.2	1.8
Block RAM Blocks	150	216
Block RAM (Mb)	5.3	7.6
UltraRAM Blocks	0	0
UltraRAM (Mb)	0	0
DSP Slices	240	360
CMTs	3	3
Max. HP I/O(1)	156	156
Max. HD I/O(2)	96	96
System Monitor	2	2

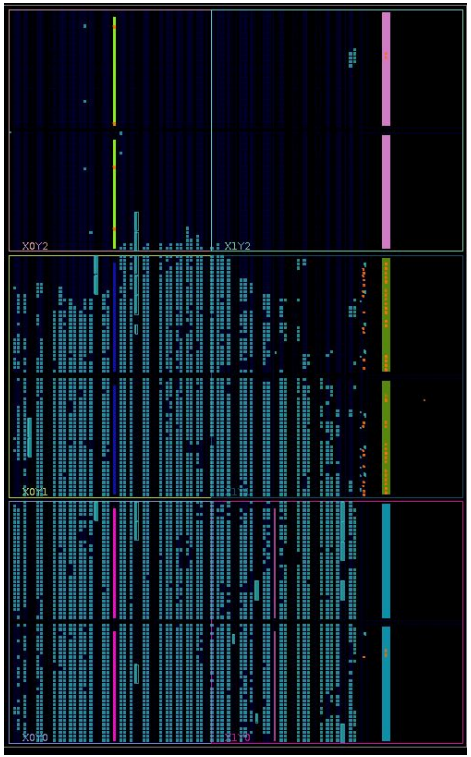
AVNET

FPGA Innards:

Raw chip die:
(not actual ZU3EG)



Compiler (synthesis)
place and route output:



AVNET

What can FPGAs (PL) do for me?

The number ONE of many reasons to use FPGA:

XILINX.

AVNET

Reasons to use an FPGA for your Python:

1. For many solutions the PL will be orders of magnitude faster

2. Precise timing capabilities (picoseconds jitter accuracy) for control of hardware

3. Determinism of algorithmic execution (no cache, preemption, task switching, threads or interrupts if you design it that way)

4. You can do things beyond what the PS can do with Python, even design your own CPU/GPU! See Xilinx's MicroBlaze™ for PYNQ: https://pynq.readthedocs.io/en/v2.0/pynq_libraries/pynq_microblaze_subsystem.html

5. The art of designing hardware with software can be rewarding and enjoyable!

Other areas where FPGAs excel:

Reprogrammability allows for in the field upgrades of hardware

Crytography:
AES, Blowfish, Twofish, RSA, Triple DES

Cellular 4/5G

Build your own CPU or GPU:
MicroBlaze™, RISC, Custom

Prototyping for ASICs

IoT: read multiple sensors simultaneously

MIMO

GNURadio
THE FREE & OPEN SOFTWARE RADIO ECOSYSTEM

Digital Motor Control:
Servo, Stepper, PWM, PDM

Scientific Computing

Parallel Processing

Cryptocurrencies:
BitCoin, Etherum

Kalman Filtering

Signal Processing:
FIR, IIR, OFDM, FFT, Correlators, CORDIC, Interp, Decimation, NCO, Mixers, Polyphase Filtering, Wavelets

Forward Error Correction Systems:
Turbo, LDPC, POLAR, Viterbi, BCH, RS

OpenCV

Direct control of other hardware

Machine Learning

Can be faster than GPUs for some things

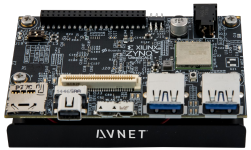
Re-routing hardware signals on fixed PCB for flexibility

Bridge between CPUs or CPUs and other hardware

AVNET

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



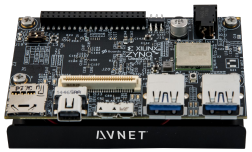
Full 1080p Images per Second in HW: 66.3
Full 1080p Images per Second in *SW: .01

HW Acceleration Factor: *6171

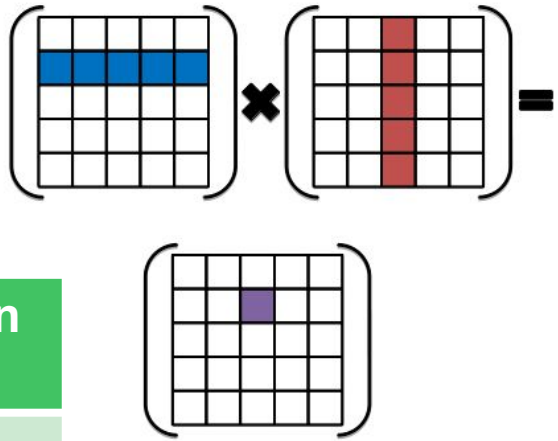
* The SW used for benchmark was running on the Ultra96 ARM Cortex™ A53 cores with same OS as the HW tests @ ~1.3GHz. Other platforms that have somewhat faster ARM cores could do a little better with just SW. Other platforms with their own hardware accelerators will also run faster than pure SW.

AVNET

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*

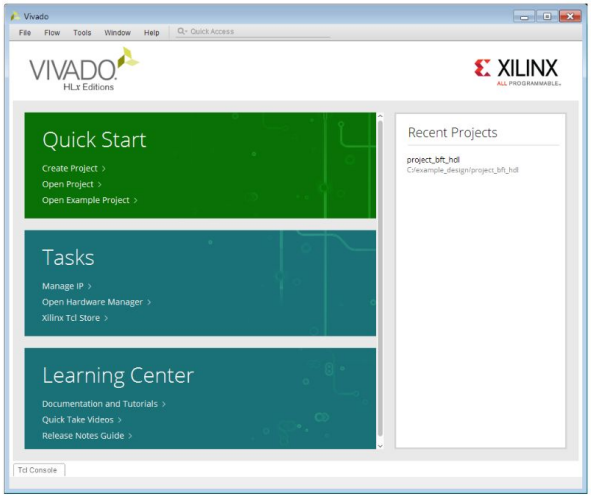
AVNET

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**

Ultra96 tools intro

Vivado HLx IDE for Ultra96 PL:



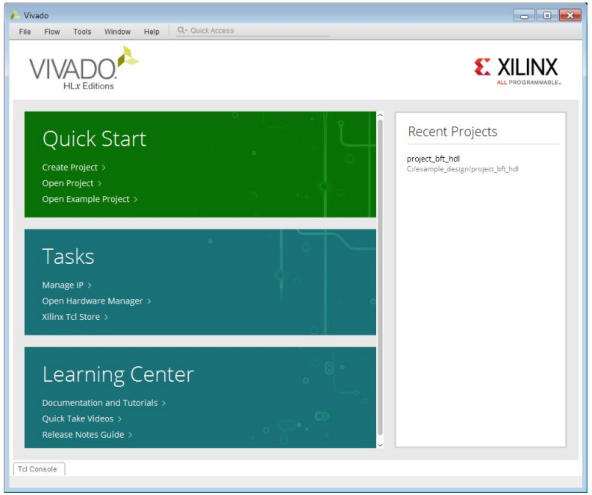
- Vivado is the main Xilinx tool that converts RTL source code files into FPGA hardware
- It is a GUI project manager
- Performs hardware verification and debugging
- Tallies and manages the internal FPGA resources
- Allows for schematic block based hardware design
- Exports and imports to many other Xilinx tools
- Performs power consumption analysis for the ZYNQ MPSoC
- Generates the bitstream/overlay files used for programming the PL
- Accepts C/C++ and converts it into hardware: High Level Synthesis (HLS)

AVNET

Vivado HLx IDE for Ultra96 PL:

VIVADO HLx Editions

Hardware describing source code

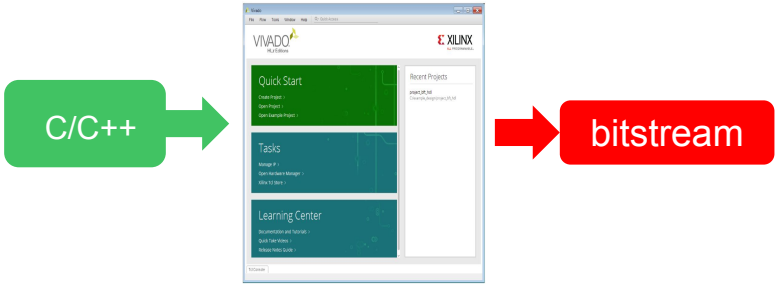


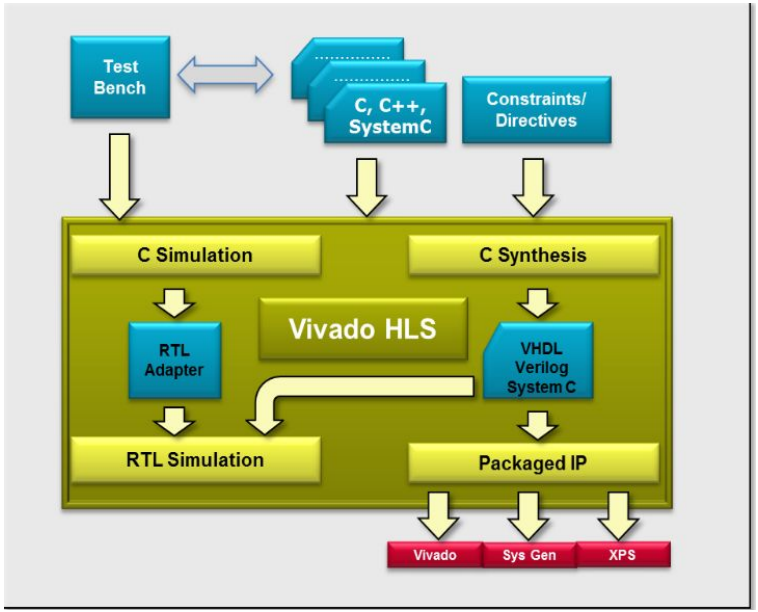
FPGA bitstream (like a binary)

AVNET

Vivado HLS for PL design:

- HLS allows one to create hardware with C/C++ but…
- It does not handle moving the data between the PS and PL for you, this may be what you want anyways especially if you are using a stand-alone FPGA or PYNQ moves the data for you, it depends





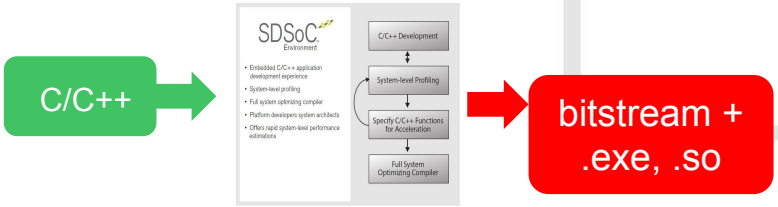
AVNET

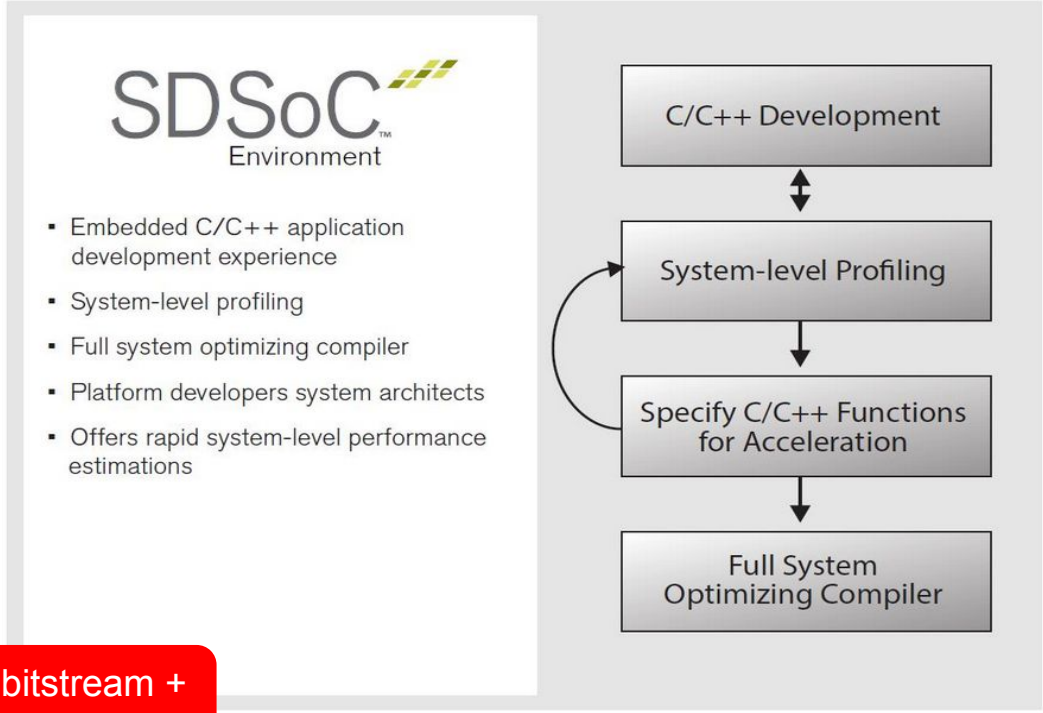
SDSoC for Ultra96 PS and PL:

- SDSoC is a separate GUI from Vivado

- Coverts your C/C++ into a hybrid system using both the PS and PL

- It moves the data between the two for you and creates PL!

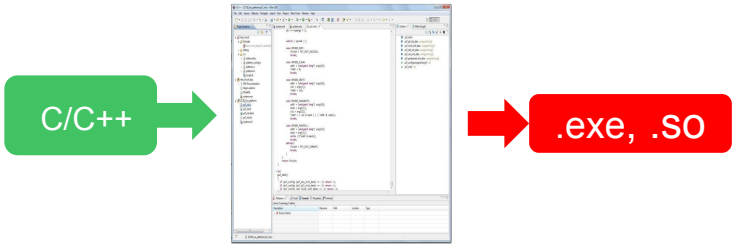


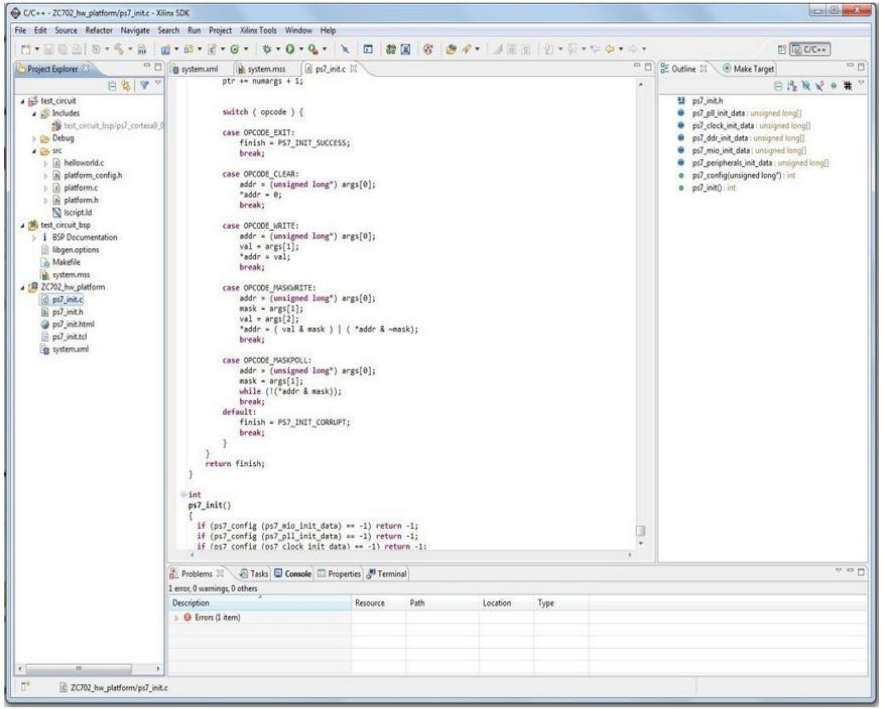


AVNET

XSDK tool for Ultra96 PS:

- XSDK is an eclipse based front end for software development on the PS
- XSDK can edit compile and debug C/C++
- XSDK can also program ZYNQ parts

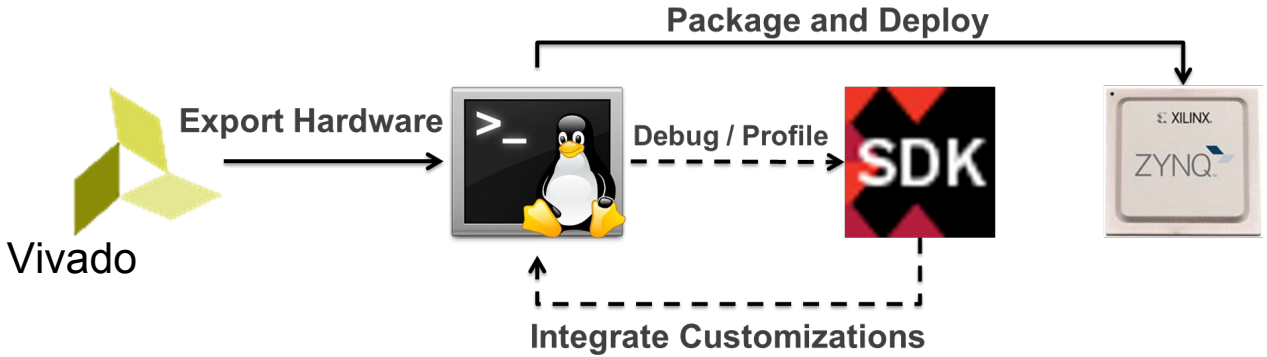




AVNET

PetaLinux for Ultra96

- aarch64 Linux Kernel
- Embedded Linux with Xilinx enhancements made to run on the ZU3EG
- Simple and time saving Xilinx ‘petalinux-’ cmds to drive Yocto
- Yocto based configuration and development tools
- Multiple choices for root FS, including ram based, PetaLinux, Debian and Ubuntu
- Integrates with PL hardware designs



AVNET

FPGA Hardware Design Languages (HDL):

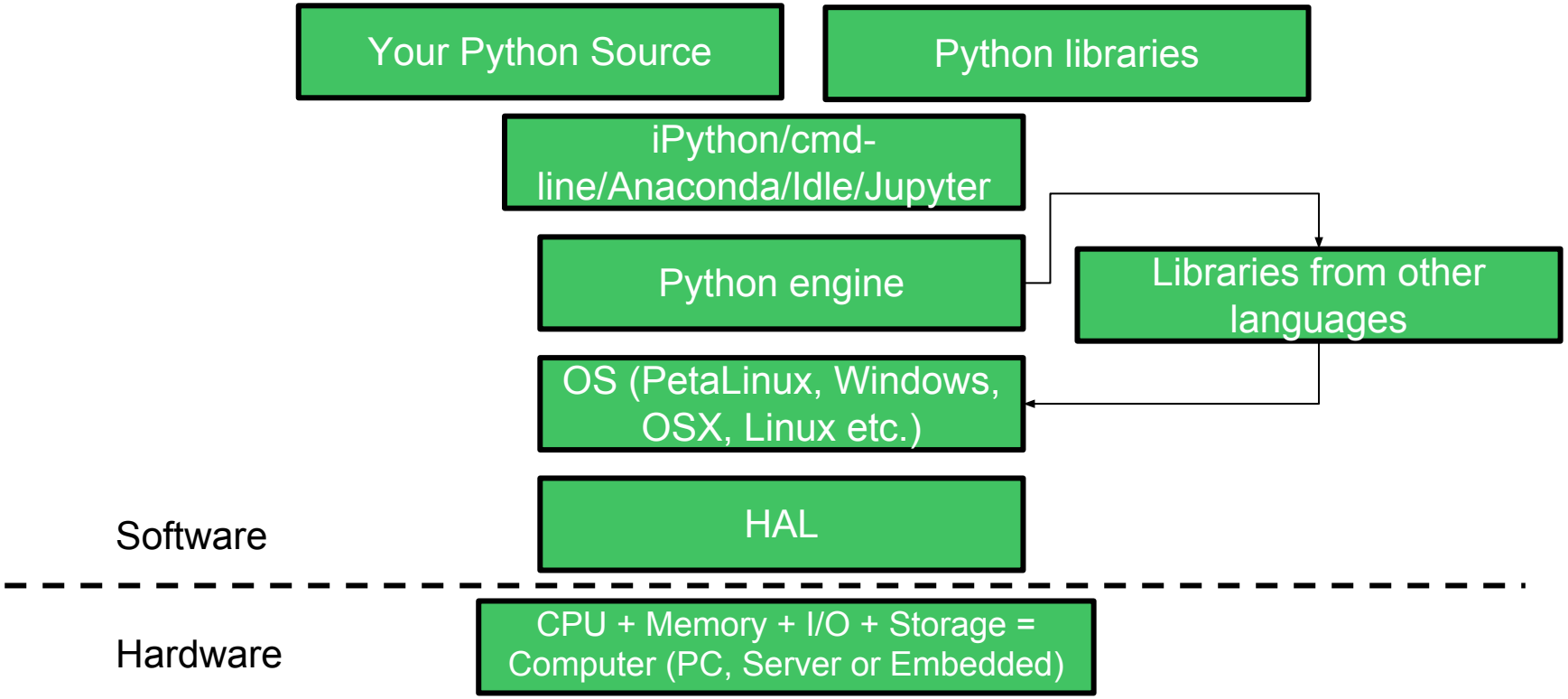
- Not that long ago hardware designers had fewer choices for PL programming languages, one of which was VHDL.
- Designers involved with ASIC design often use a language called Verilog, FPGAs can also be programmed using it.
- These languages work very well and are still supported. They are elegant and very powerful but less people are familiar with them compared to Python or C/C++.
- Xilinx reached out to the software community and created tools to allow them to design hardware using C/C++: SDSoC and HLS.
- There is also a 3rd party project that allows hardware design using Python itself. See http://www.myhdl.org
- Xilinx also allows hardware design using drag and drop blocks, this is referred to as IP Integrator.

Configuring the FPGA (NOT designing the HW):

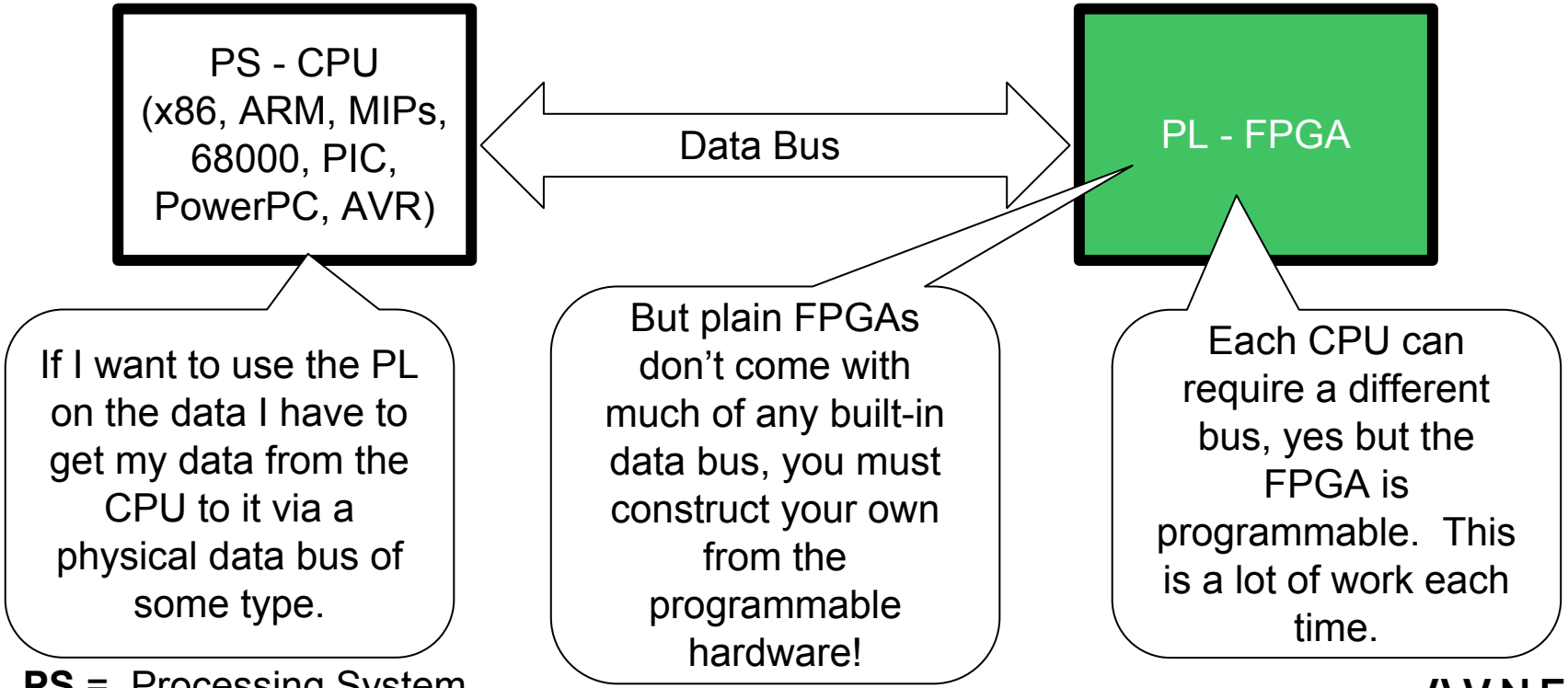
1. Write your code to define the hardware using your preferred method
2. Use the FPGA compiler (HLS, SDSoC, VHDL, Verilog, GUI Block Design) running on a PC to map your design for you into the internal logic blocks and interconnects.
3. The final output from the FPGA compiler will be a file that contains the information to configure the device. This is called a **bitstream** file in the PYNQ context it will be called an **overlay**.
4. The bitstream file will need to be transferred from the PC to an external CPU or ZYNQ device and then to the PL portion of the device using a precise protocol. Xilinx details this protocol and provides tools to configure the ZYNQ devices. This is called configuring the FPGA and is analogous to a software OS loading and running an .exe.

Python and FPGAs

Traditional context for full Python systems:



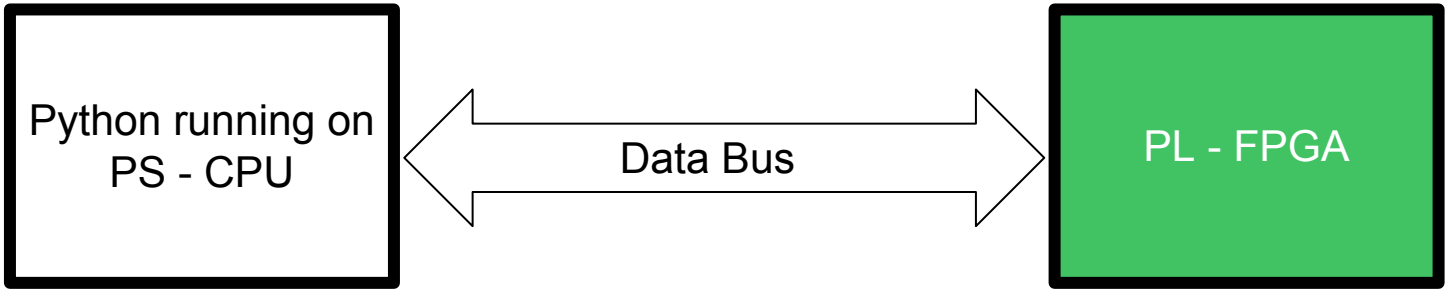
In the beginning (and still an option - 2 separate devices):



PS = Processing System

AVNET

Python data to bits and back again:



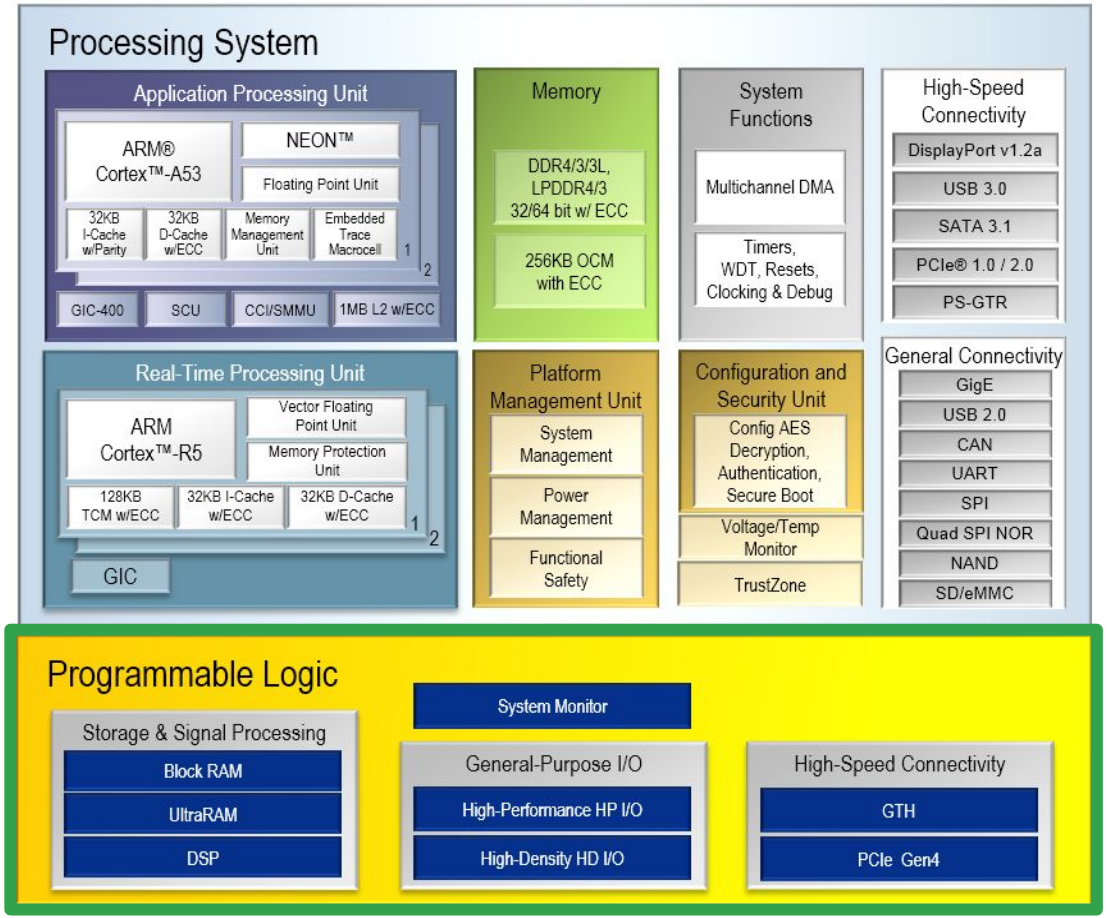
Python handles data representing information great but not as good at bits!

FPGAs great at bits but not as good at directly handling high level Information!

Various Python to C conversion techniques exist: CFFI, c-types, Cython, spam etc. Use your favorite.

From C/C++ you can then easily do bits and talk to hardware like the FPGA

Ultra96 PS + PL all in one ZYNQ device:



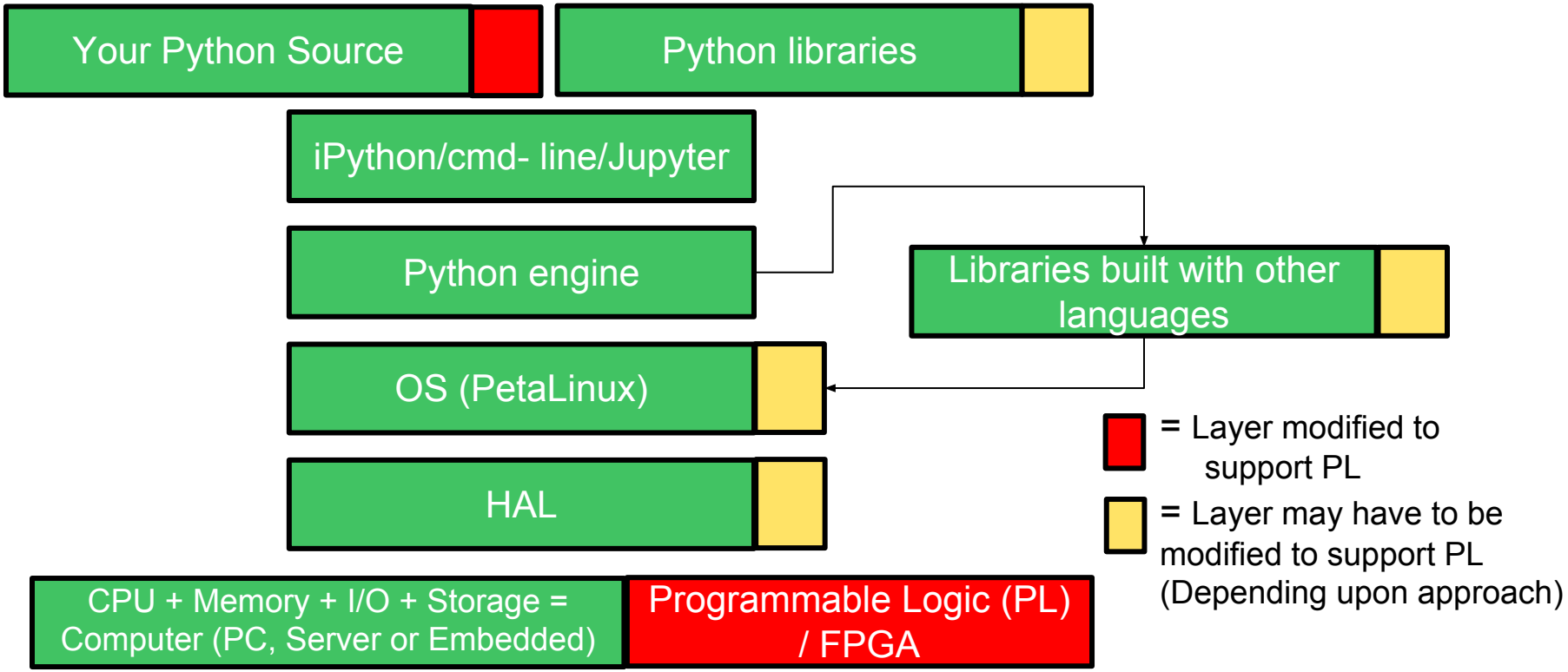
Enter Ultra96's ZU3EG ZYNQ UltraScale+ MPSoC

The bus between the CPU and FPGA are in the same chip and Xilinx has designed the data bus between them for you.

PS = Processing System
PL = Programmable Logic (FPGA)

AVNET

Python on Ultra96 PetaLinux Platform:



AVNET

PYNQ

An easier path - Xilinx PYNQ™ for Ultra96:

▪**What is PYNQ?**

An open source software **framework** designed to make Ultra96 more Python friendly and easier for Python to interact with the PL on embedded system platforms. It is comprised of:

- PetaLinux (aarch64 kernel)
- Ubuntu Bionic root file-system
- Full Python (as opposed to Micro Python)
- Jupyter Notebooks
- Python libraries for using the Xilinx PS and PL

▪**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!

AVNET

What PYNQ does for you:

Provides a Python “pynq” library with the following helper functionality:

- HW Interrupts
- Manipulate hardware pins
- Map physical memory into Python for PL/PS xfer
- Overlay – program the FPGA bitstream from Python
- Read various PS and PL attributes
- Utilize DMA to move data between PL and PS
- Primitives to help accelerate parts of numpy
- pynq.lib contains objects to manipulate some of the board’s external hardware and operate custom MicroBlaze™ CPUs in the PL.

Read all about it:
https://pynq.readthedocs.io/en/v2.2.1/pynq_package.html

pynq.interrupt Module

pynq.gpio Module

pynq.lib Package

pynq.mmio Module

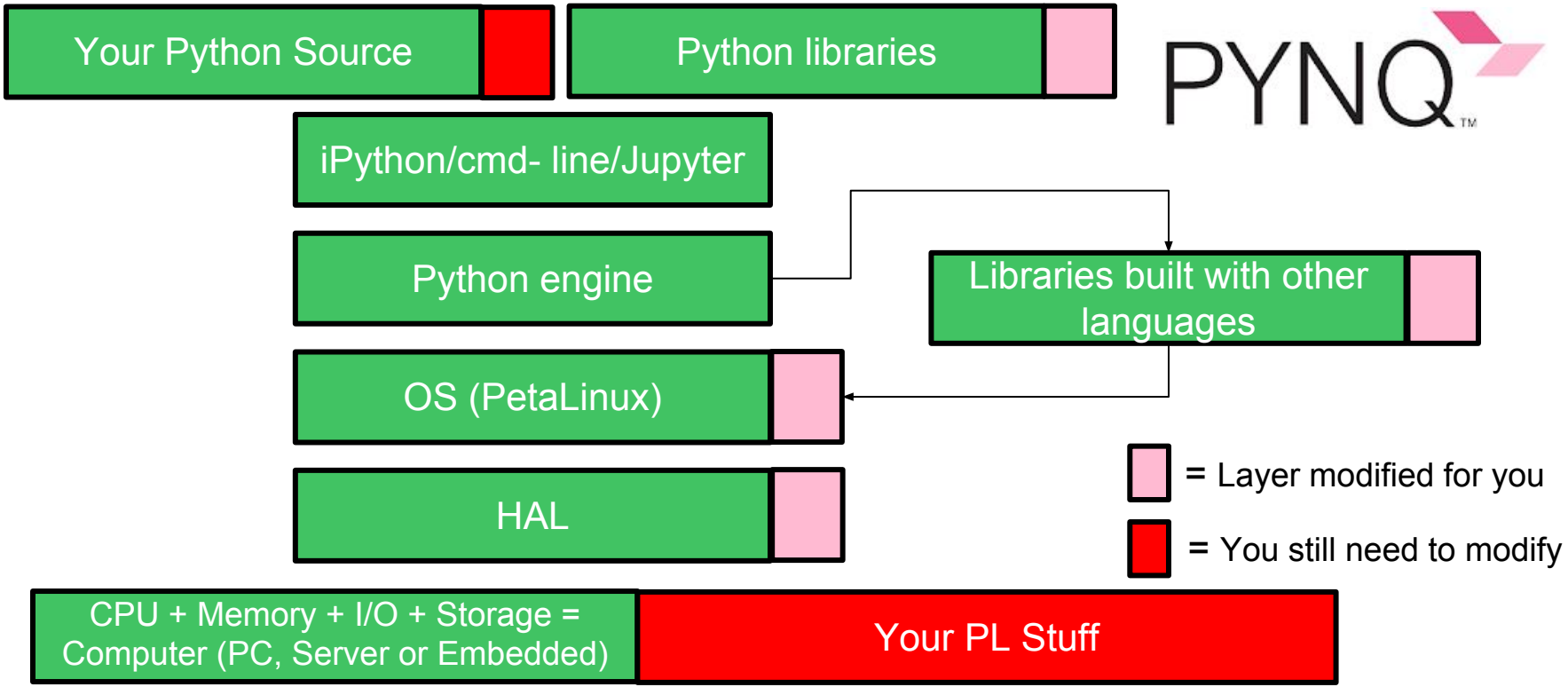
pynq.overlay Module

pynq.ps Module

pynq.pl Module

pynq.xlnk Module

Python on the Ultra96 PYNQ Platform:

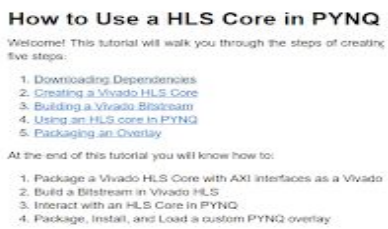


AVNET

Great place to start: www.pynq.io/community.html

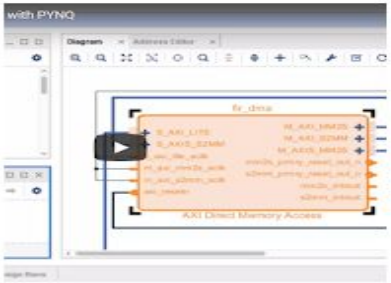
Tutorials and other resources

Tutorial: HLS filter example

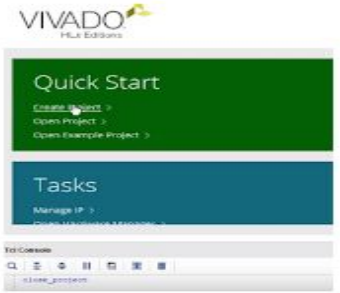


Video: Custom HLS adder IP

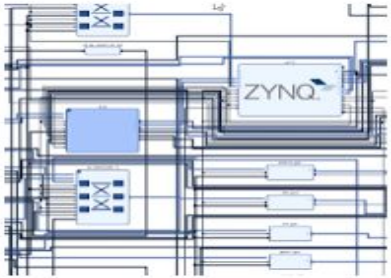
Video: Accelerate FIR software function



Video: Add existing IP to a PYNQ overlay



Video: Control custom IP using GPIO

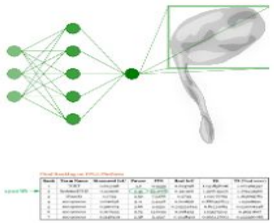


Great place to start: www.pynq.io/community.html

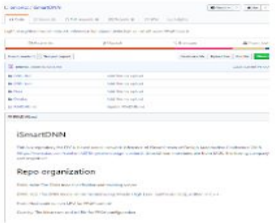
Community Projects

A selection of projects from the PYNQ community is shown below. Note that some examples are built on different versions of the PYNQ image.

spoonNN
ETH Zurich
FPGA-based neural network inference project



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

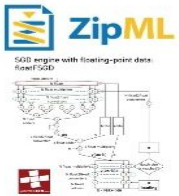


TGIIF
1st place in the DAC 2018 design contest for neural network object detection

cv2PYNQ
FAU
Accelerated OpenCV image filtering library.

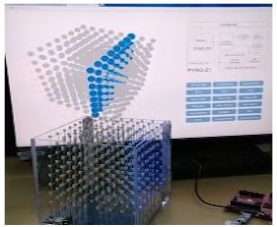
Video processing
KU Leuven
Hardware accelerated videoprocessing

ZipML–PYNQ
ETH Zurich
Hardware accelerated compression



PYNQ bot
IT Tallaght
Control of robotic car from PYNQ

PYNQ LED cube
Fudan University, Xilinx China
Controlling an LED cube from PYNQ



AVNET

When will Xilinx PYNQ™ be available for Ultra96?

PYNQ™

- PYNQ for Ultra96 is coming soon, expected 1st week of Oct. 2018!!!

http://www.pynq.io & https://github.com/Xilinx/PYNQ
http://zedboard.org/product/ultra96

ZYNQ UltraSCALE+

Ultra96 PYNQ platform will be hosted on Avnet's github:
http://github.com/Avnet

jupyter

Linux

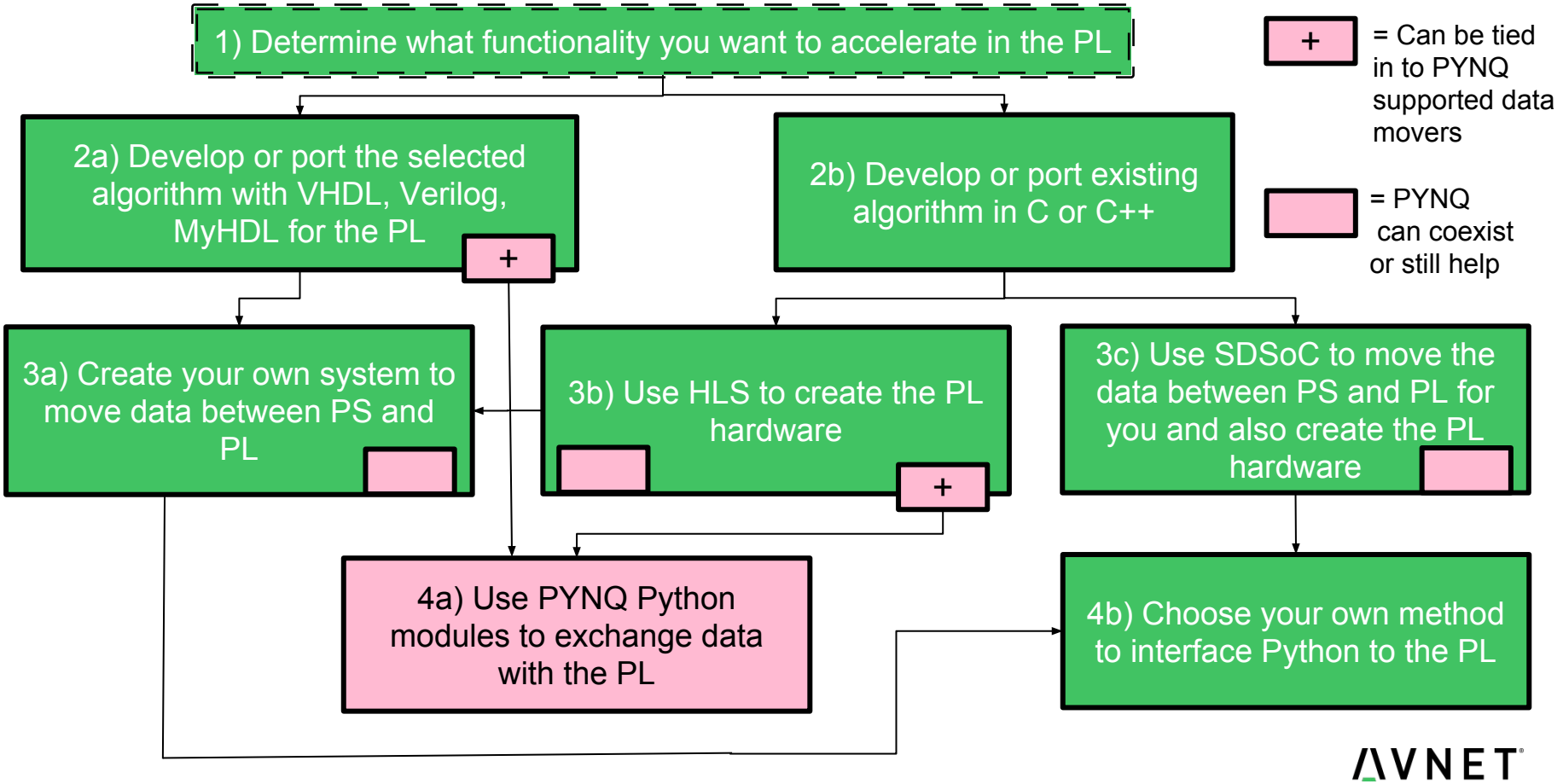
python™

XILINX

96Boards Partner

AVNET

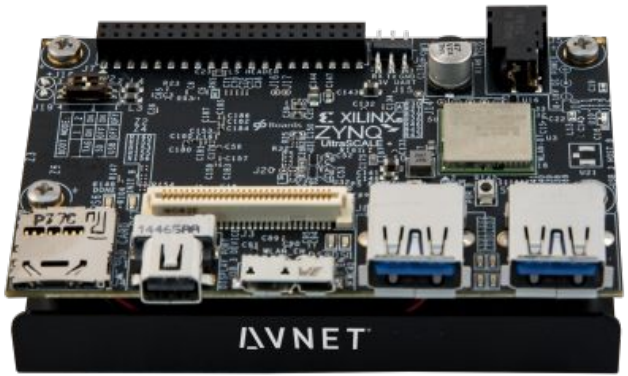
Summary of Accelerating Python development workflow:



AVNET

Acquire your own Ultra96 board for $249:

http://zedboard.org/product/ultra96

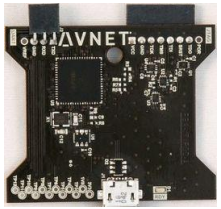


Includes:

- Ultra96 development board
- 16 GB pre-loaded MicroSD card + adapter
- Voucher for SDSoC license from Xilinx
- Quick-start instruction card

Does not include (but necessary):

- External 12V 2A 96boards adapter



Optional Accessories:

- Seed Studios Grove Starter Kit for 96boards
- Other compatible accessories
- JTAG to USB adapter board

96Boards Partner

AVNET

Acknowledgements:

THANK YOU for attending!

Much gratitude to my friend **Aron Khan** who weathered v1.0 of my presentation and offered very useful advice to improve it

A special salute to all the folks in the trenches who carry the PYNQ flag, especially:

Dr. Yun “Rock” Qu PhD, Dr. Graham Schelle PhD, Cathal McCabe, Peter Ogden, Anurag Dubey

And thank you to the following people who also helped provide for the opportunity to share this with you:

Avnet: Kevin Keryk, Bryan Fletcher

AVNET

Get more help on PYNQ™ and Python for Ultra96:

PYNQ Workshop: https://github.com/Xilinx/PYNQ_Workshop

See other's examples (Ultra96 examples will be added soon):
http://www.pynq.io/community

See Avnet's Ultra96 tutorials (more on the way):
http://zedboard.org/support/design/24166/156

Join Xilinx's Developer Zone to access free tools:
https://www.xilinx.com/products/design-tools/software-zone.html

AVNET