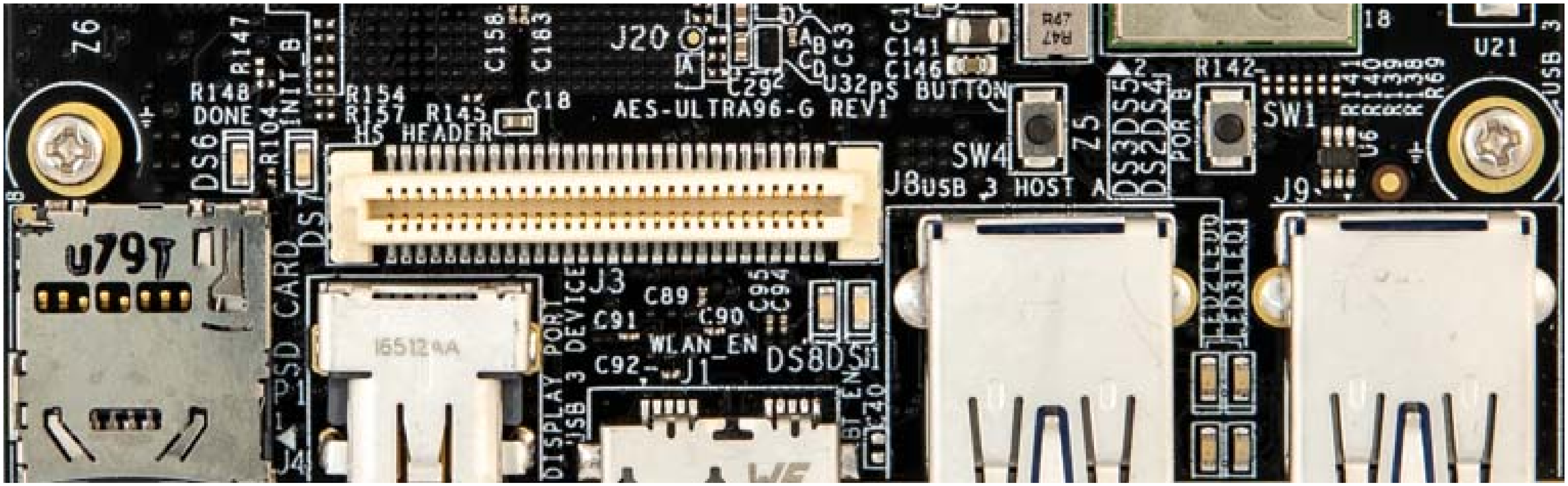
Mastering Programmable Logic with Xilinx SDSoC

San Jose - 01 OCT 2018

XILINX DEVELOPER FORUM

Reach Further™



Title: Mastering Programmable Logic with Xilinx SDSoC

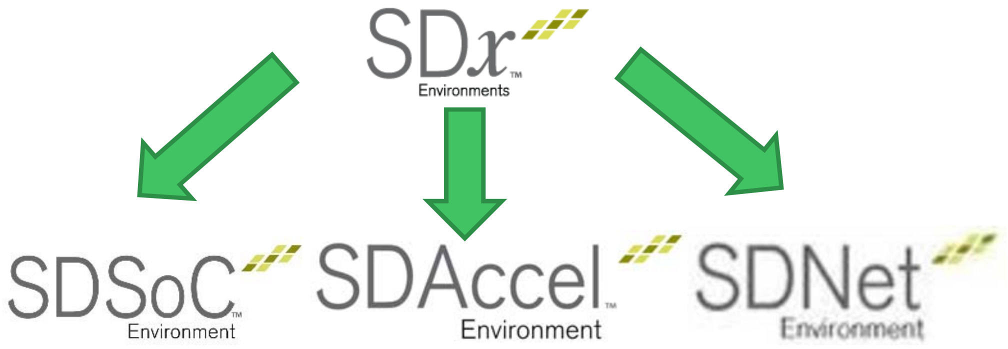
- Abstract: Using proven flows for SDSoC, the student will learn how to navigate SDSoC. Through hands-on labs, we will create a design for a provided platform and then also create a platform for the Avnet Ultra96. You will see how to accelerate an algorithm in the course lab. This experience should give you the background to assist you in developing custom platforms with custom algorithms, accelerated by SDSoC.

Agenda

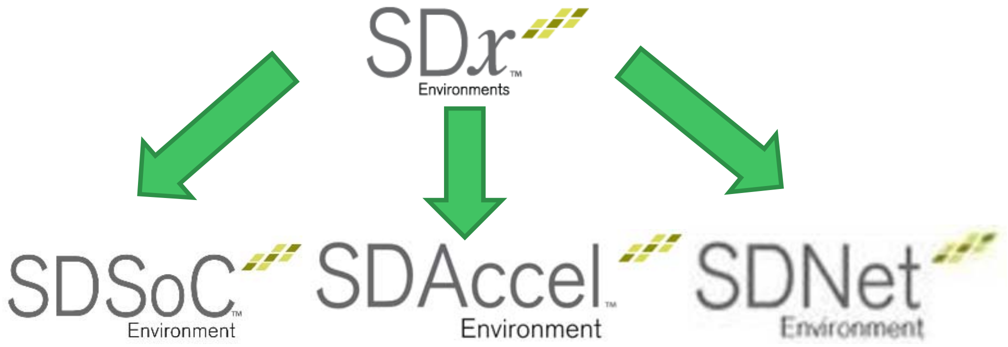
- SDSoC -- What, Why, Where?
- Why is this important to me?
- Overview of Design Flow
- Platforms…what are they?
- Hardware Platform
- Software Platform
- Final Platform Details
- Lab 1
  - Use the pre-built platform to demonstrate the end goal for this training

What is SDSoC?

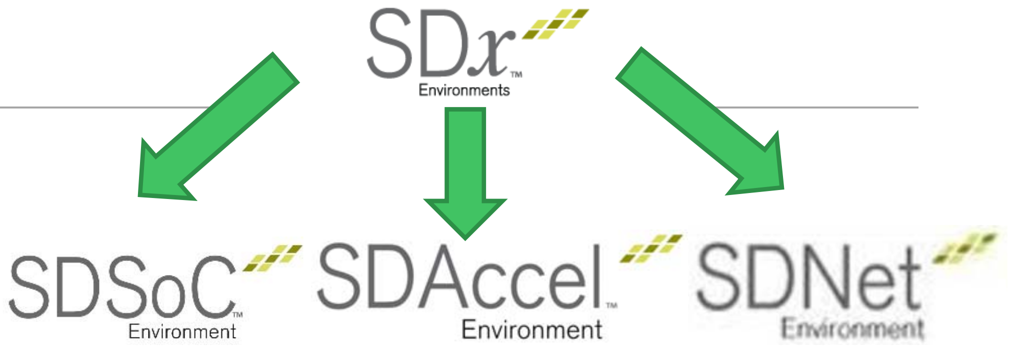
What is SDSoC ?



What is SDSoC ?



What is SDSoC ?



- SDSoC is part of the SDx umbrella
- That umbrella includes:
  - SDSoC
    - Software Defined accelerators for a System on Chip
  - SDAccel
    - Software Defined Accelerators (targeting more of a PURE Programmable Logic)
  - SDNet
    - Software Defined Specification Environment for Networking

What is SDSoC?

- SDSoC delivers
  - System level profiling
  - Automated software acceleration in programmable logic
  - Automated system connectivity generation
  - Libraries to speed programming

What is SDSoC?

- In other words
  - It is an advanced tool suite
  - Eases the burden of managing the resources
  - Eases the burden of managing the interconnect of more complex designs
  - Deals with the complexities of tool suites for you!
  - All the while providing a better embedded design experience.

What is SDSoC?

- SDSoC is more a resource manager providing an easier path to using the suite of included tools

- What does all this mean?
  - Your job with SDSoC is to create a platform
  - You are to identify functions for acceleration (profiling or manually)
  - You define what resources are available to the platform
  - SDSoC will look at the C/C++/OpenCL and USE the available resources to most optimally hit your power/performance targets!

Why is this important to me?

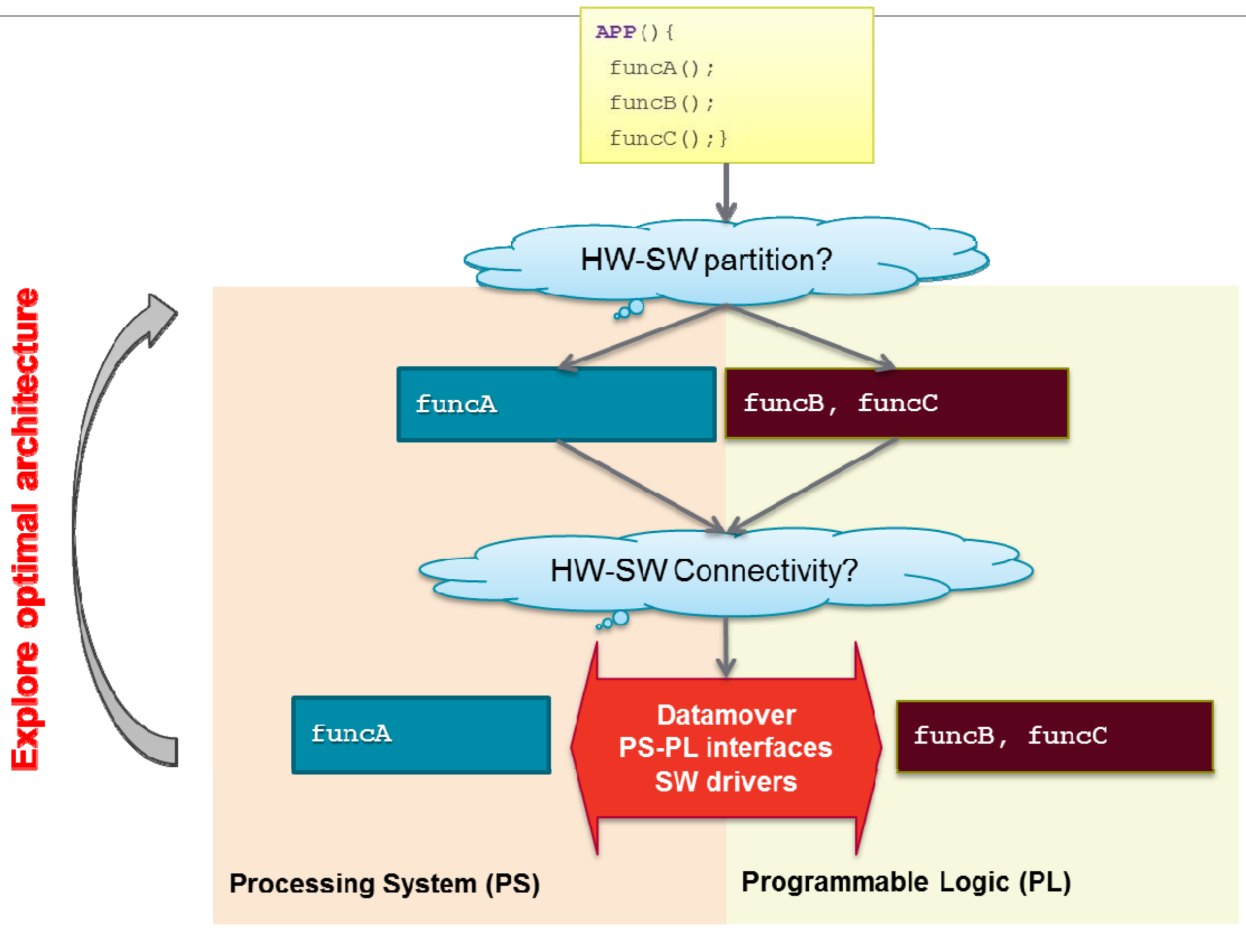
…Why is this important to me?

- You do not have to be a FPGA expert!
  - SDSoC is another step closer to more easily empowering Engineers who have little to no FPGA/HDL experience

- Problems are more complex
  - SDSoC allows a designer to take a **system's approach**
  - SDSoC allows you to import your existing Zynq design and start developing new applications in C/C++/OpenCL
  - Designers can concentrate more effort on the solution
    - Instead of determining gate level logic

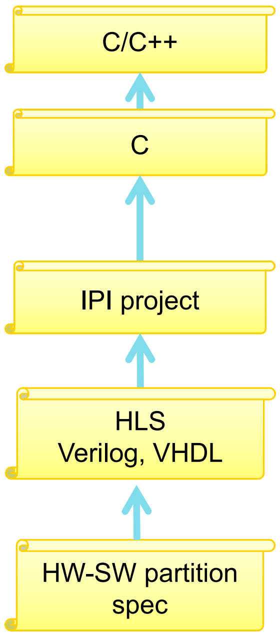
Overview of Design Flow

Typical Zynq Development Flow

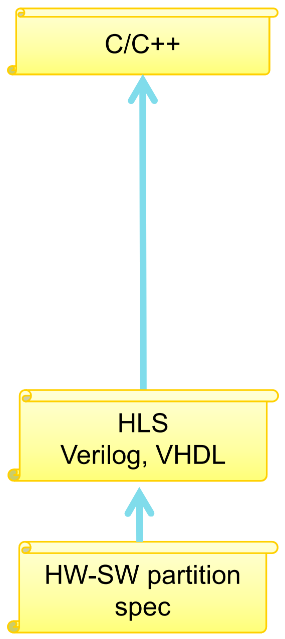


AVNET

After SDSoC:

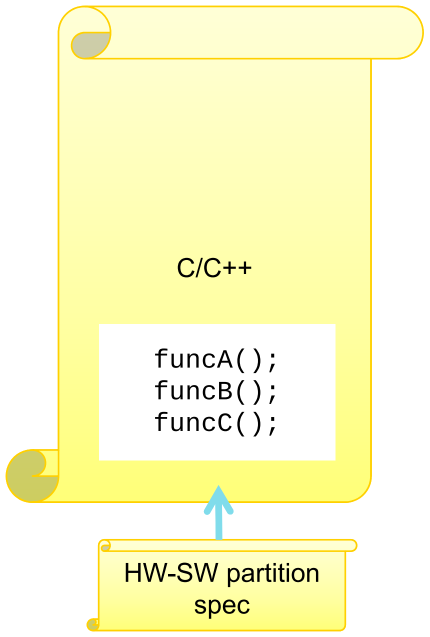


After SDSoC:



- Remove the manual design of SW drivers and HW connectivity

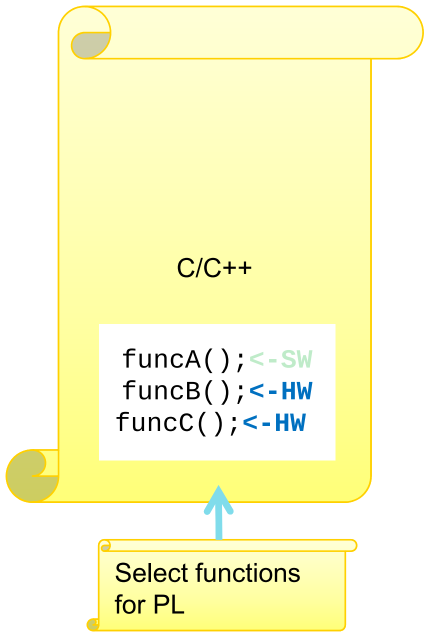
After SDSoC:



```
funcA();
funcB();
funcC();
```

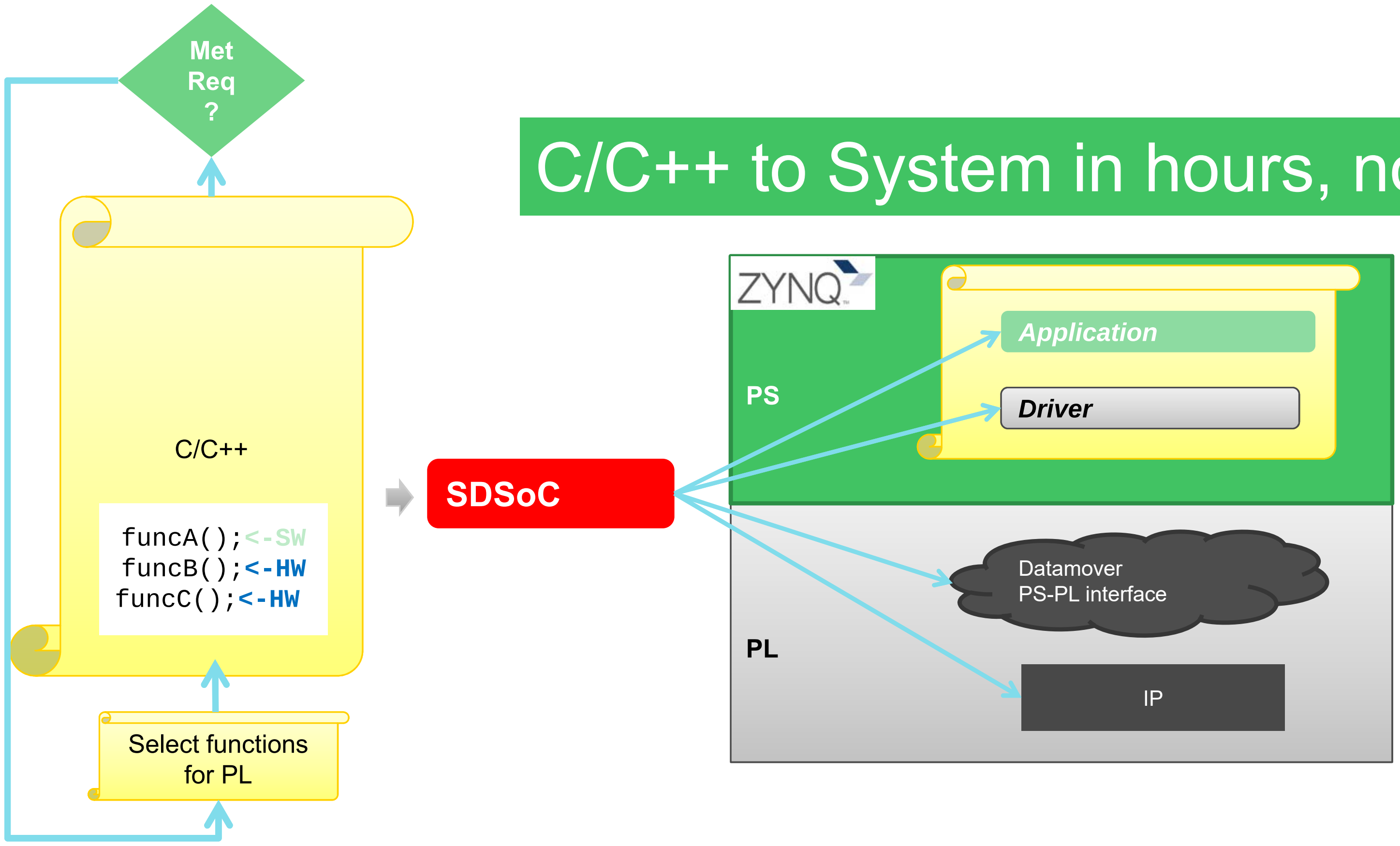
- Remove the manual design of SW drivers and HW connectivity
- Use the C/C++ end application as the input calling the user algorithm IPs as function calls

After SDSoC:



- Remove the manual design of SW drivers and HW connectivity
- Use the C/C++ end application as the input calling the user algorithm IPs as function calls
- Partition set of functions to Programmable Logic by a single click

After SDSoC: Automatic System Generation



C/C++ to System in hours, not days

Platforms… what are they?

Platforms…what are they?

- A *platform* is a base system designed for reuse
  - Built using Vivado, SDK (now optional), SDx and OS tools
  - Your Hardware is Defined: Processing system, I/O subsystems, memory interfaces, …, with a well-defined port interface (AXI, AXI-S, clock, reset, interrupt)
  - Software Definition: OS, device drivers, boot loaders, file system, libraries,…

Platforms…what are they?

- SDSoC extends the platform with application-specific hardware and software
  - Complete solution generated from C/C++ source
  - Solution can include RTL packaged

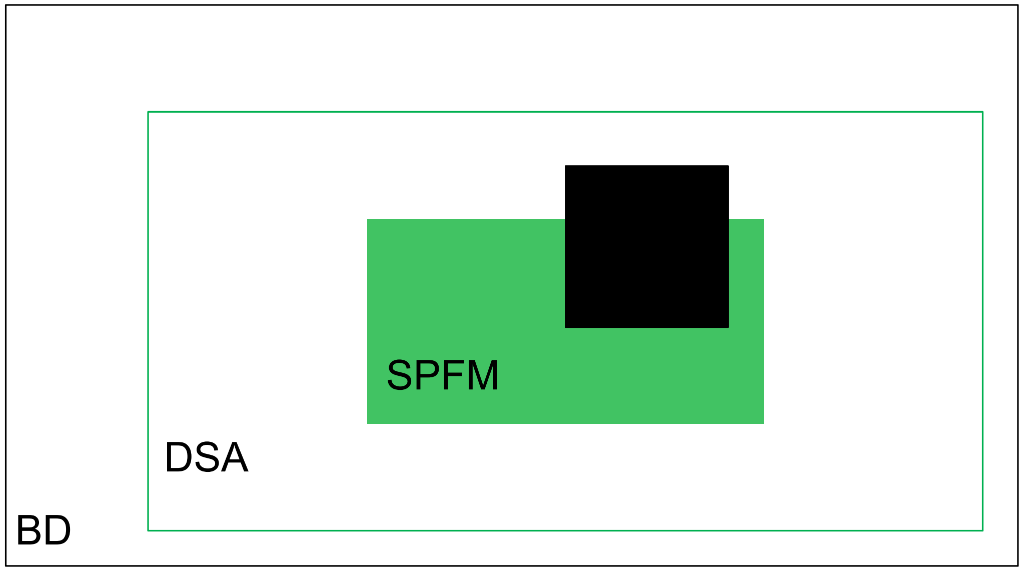
- SDSoC treats platforms as independent solution spaces
  - Generate IPs for your solution based on the resources available
  - Each solution is tailored for each platform
  - A solution can have the target platform switched through little more than a few GUI clicks!

- Platforms must include
  - Processing IP Block
  - At least one general purpose AXI master port
  - Interrupt Concatenation, Clock Reset Blocks

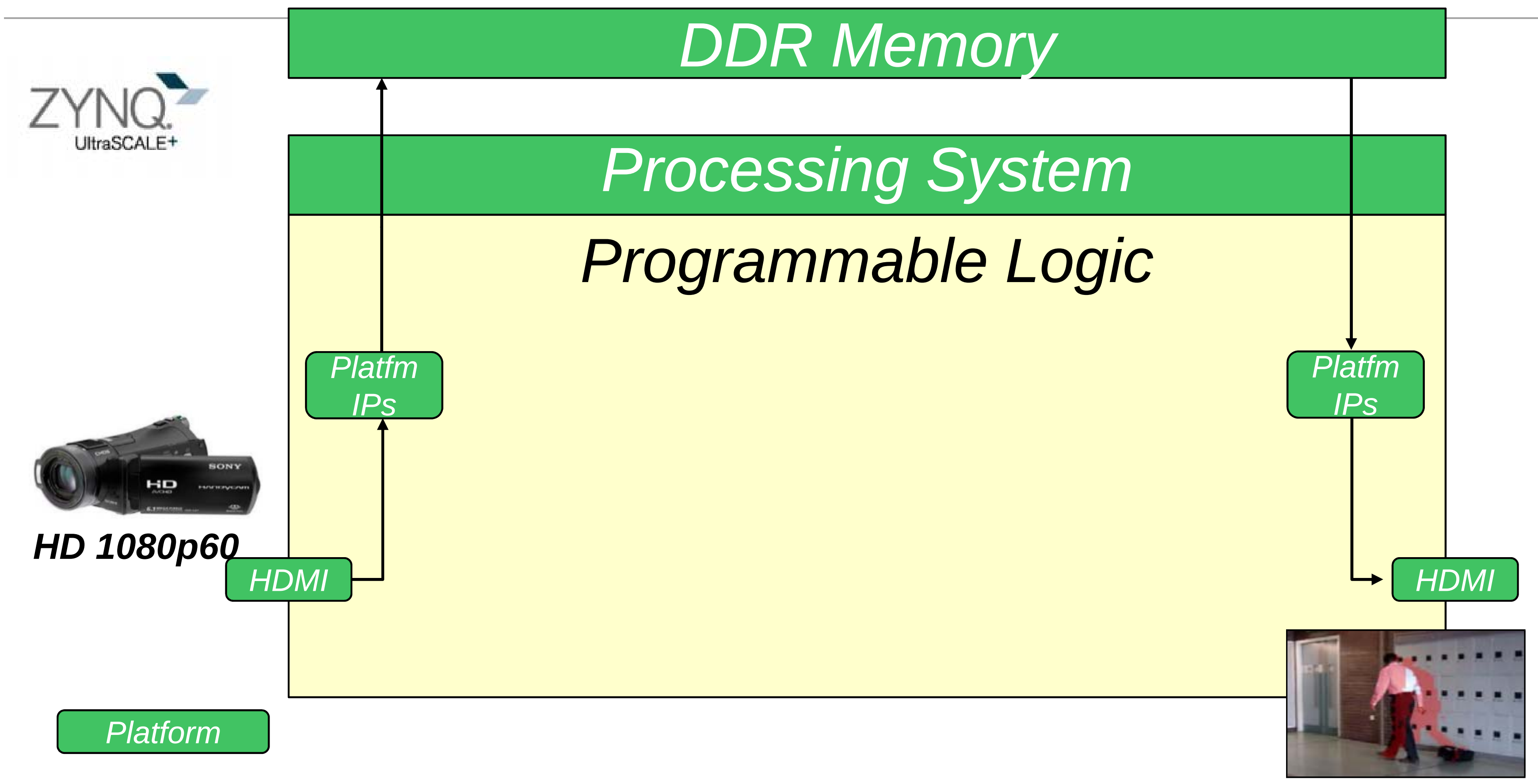
XPFM: A Pictoral

- XPFM
  - Platform Reference to the location of the DSA and SPFM
- BD
  - Block Design
- DSA
  - Device Support Archive
- SPFM
  - Software Platform Descriptor XML

- Black Box
  - SDSoC created IP

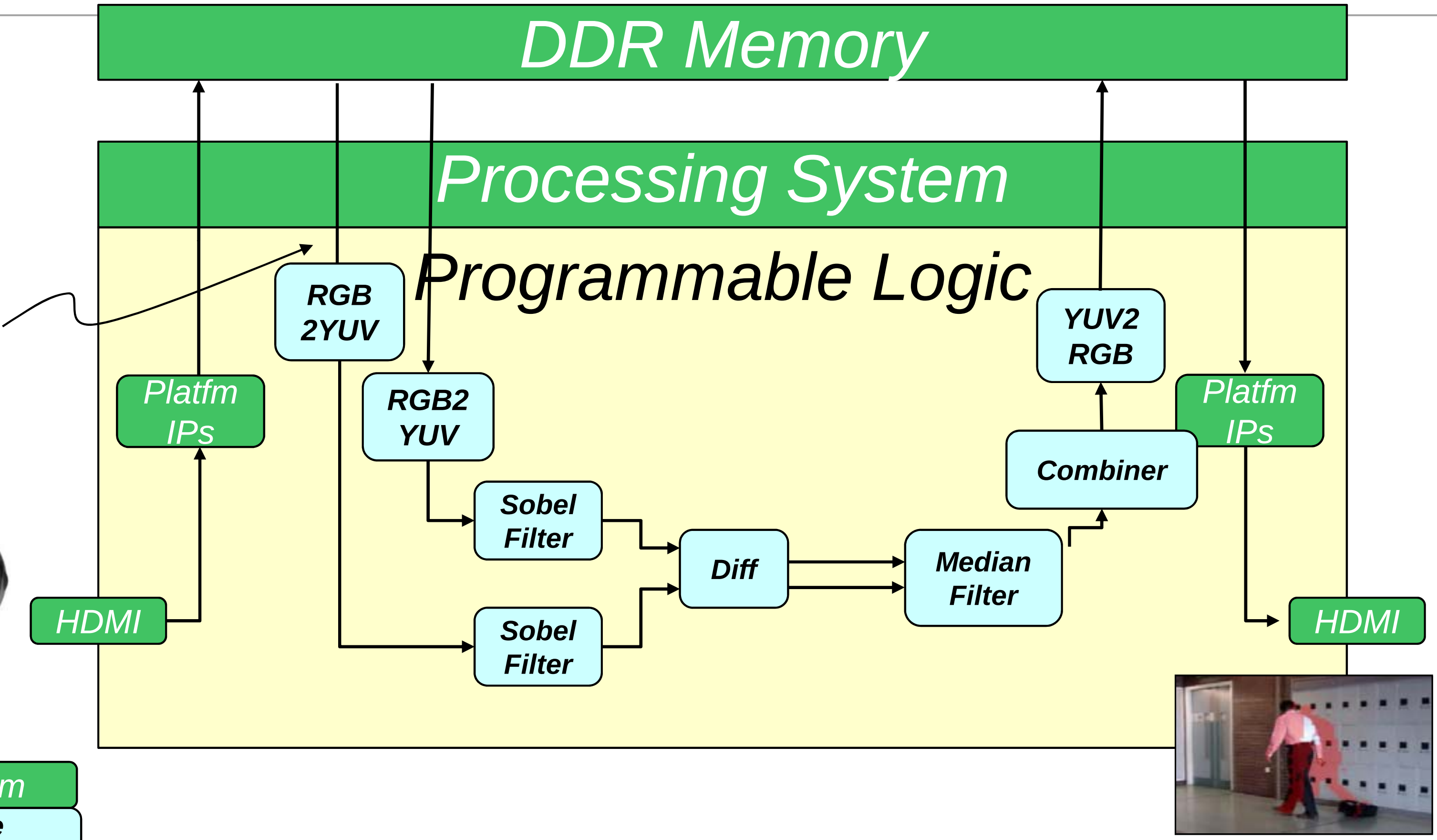


Example: Motion Detection Application

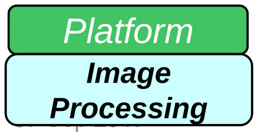


Example: Motion Detection Application

Data Movers added and managed

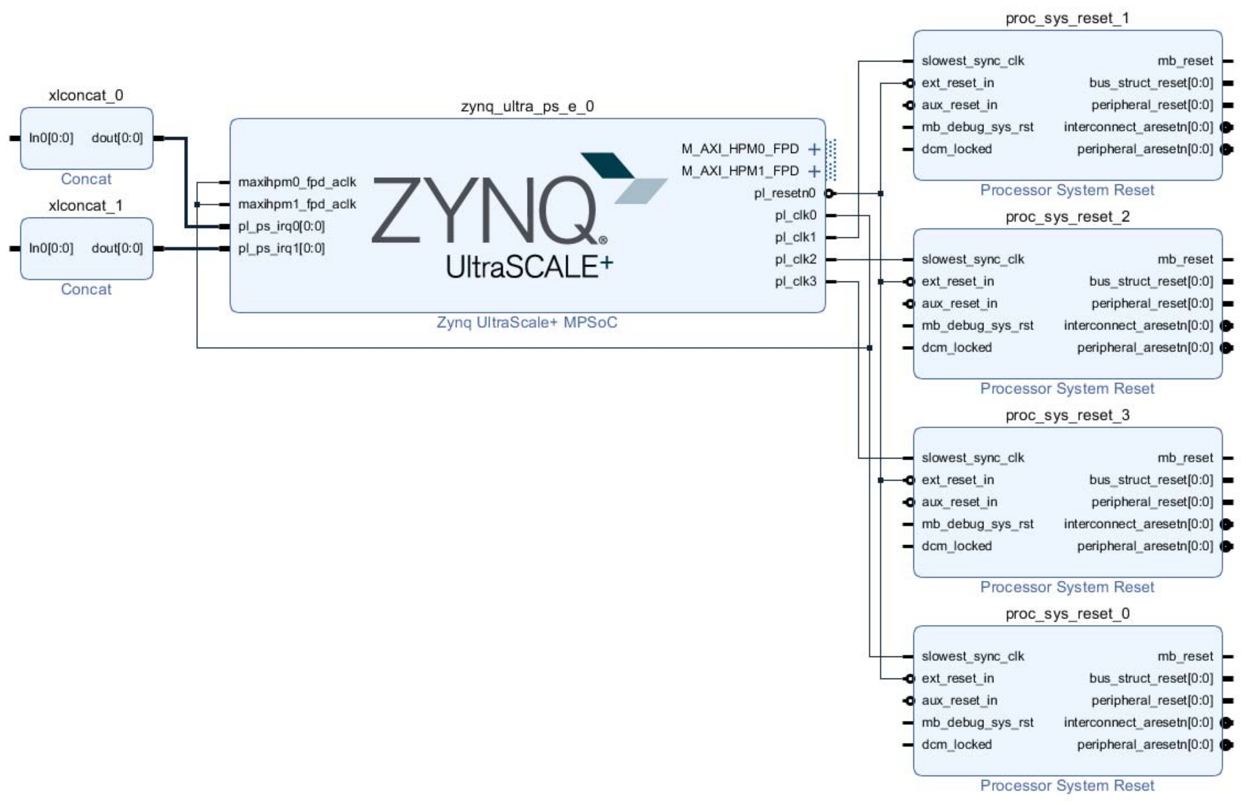


HD 1080p60

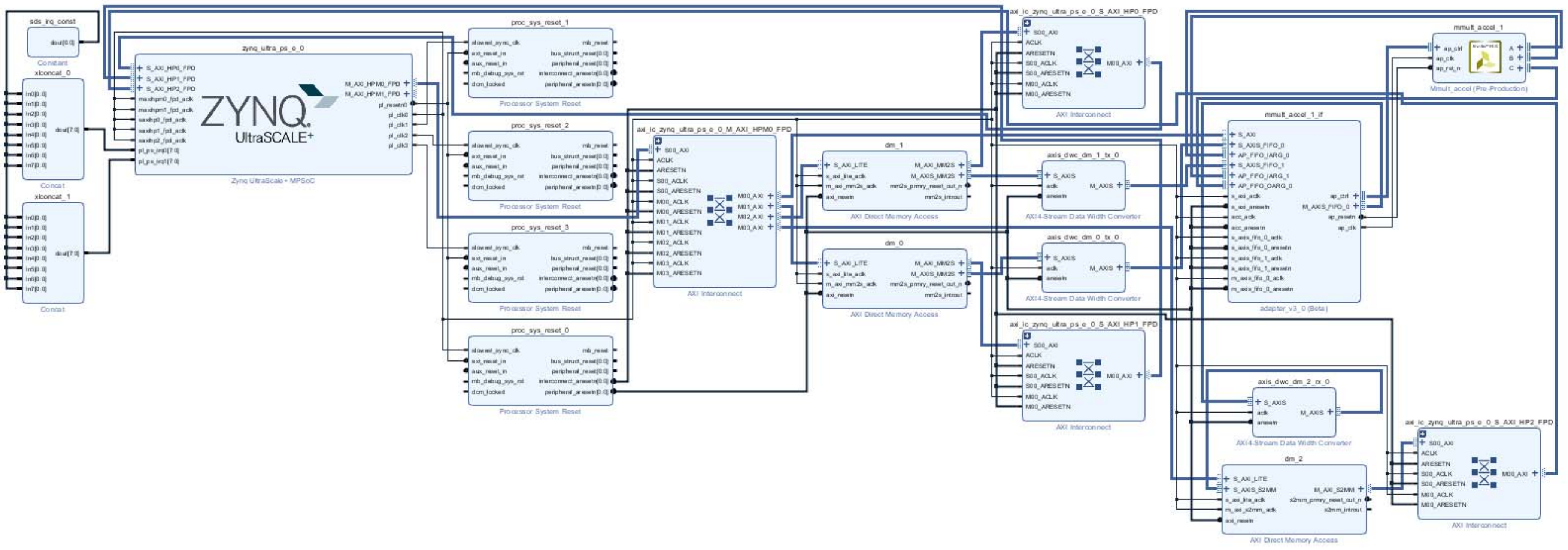


AVNET

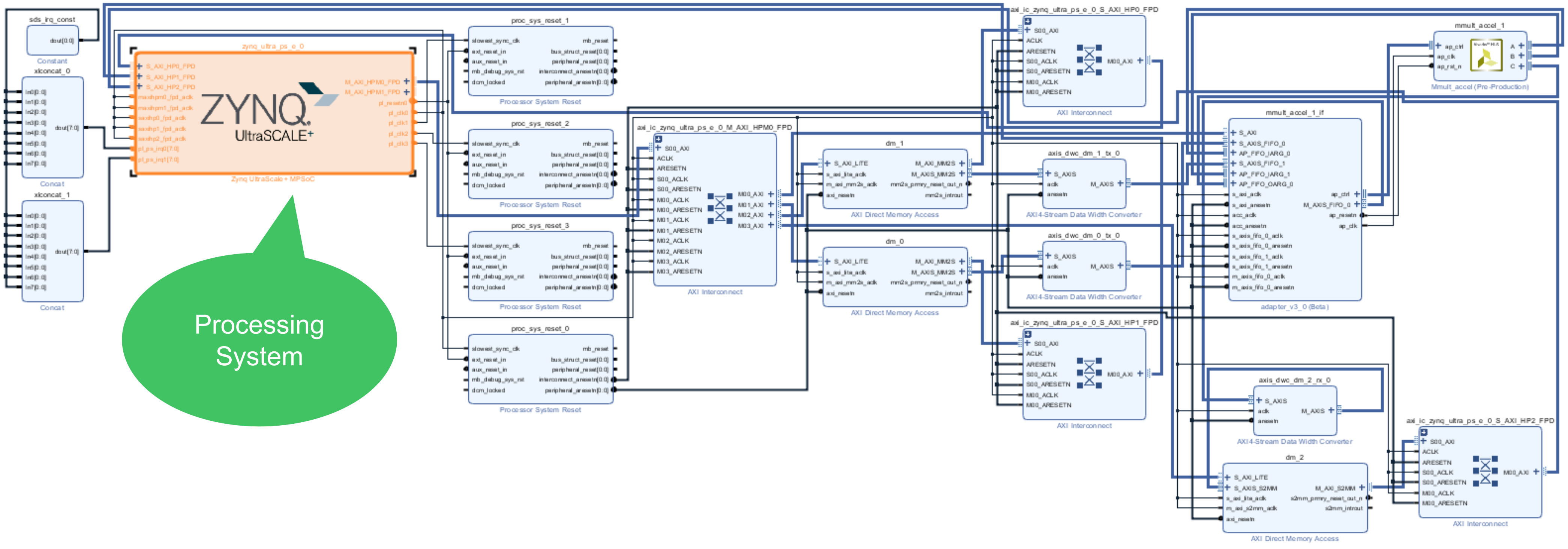
Example: Ultra96 Platform



Example: Ultra96 Platform with Matrix Multiply

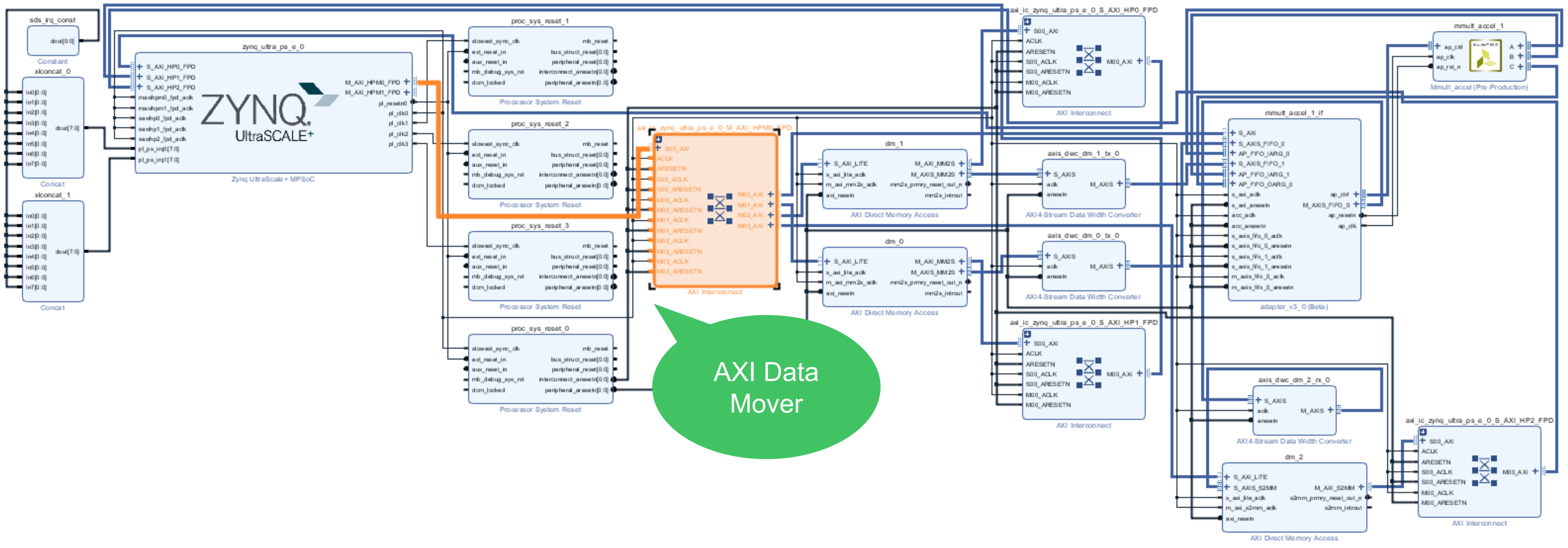


Example: Ultra96 Platform with mmult

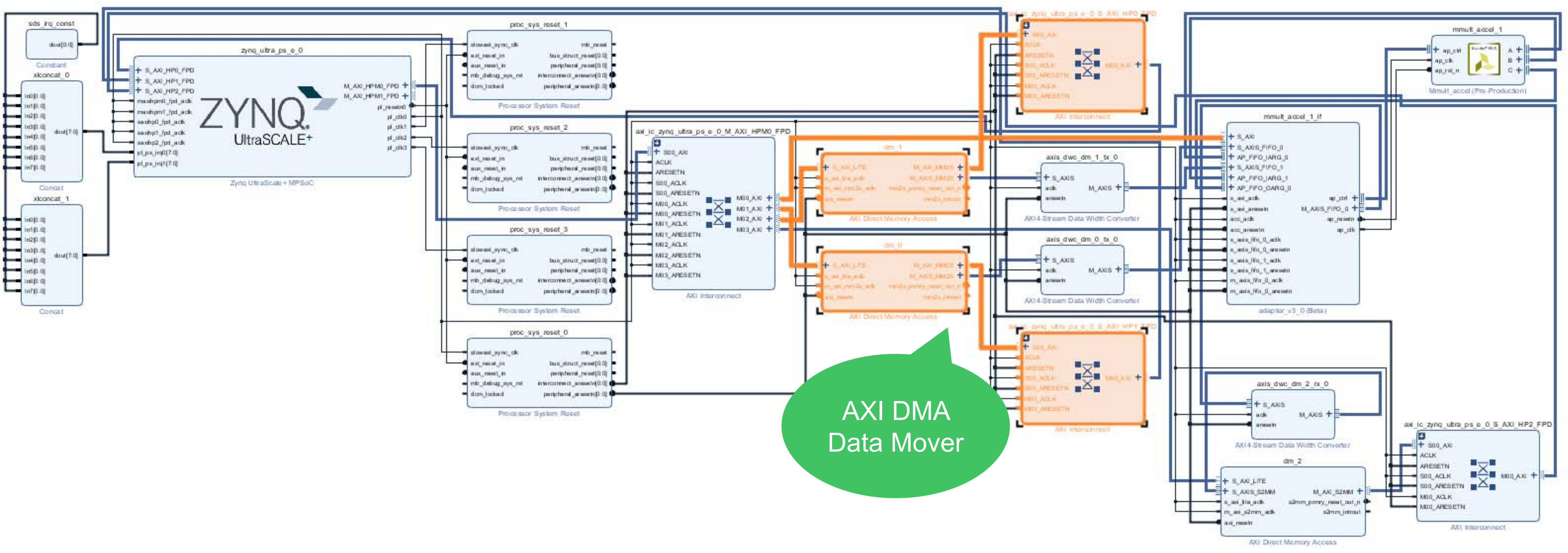


AVNET

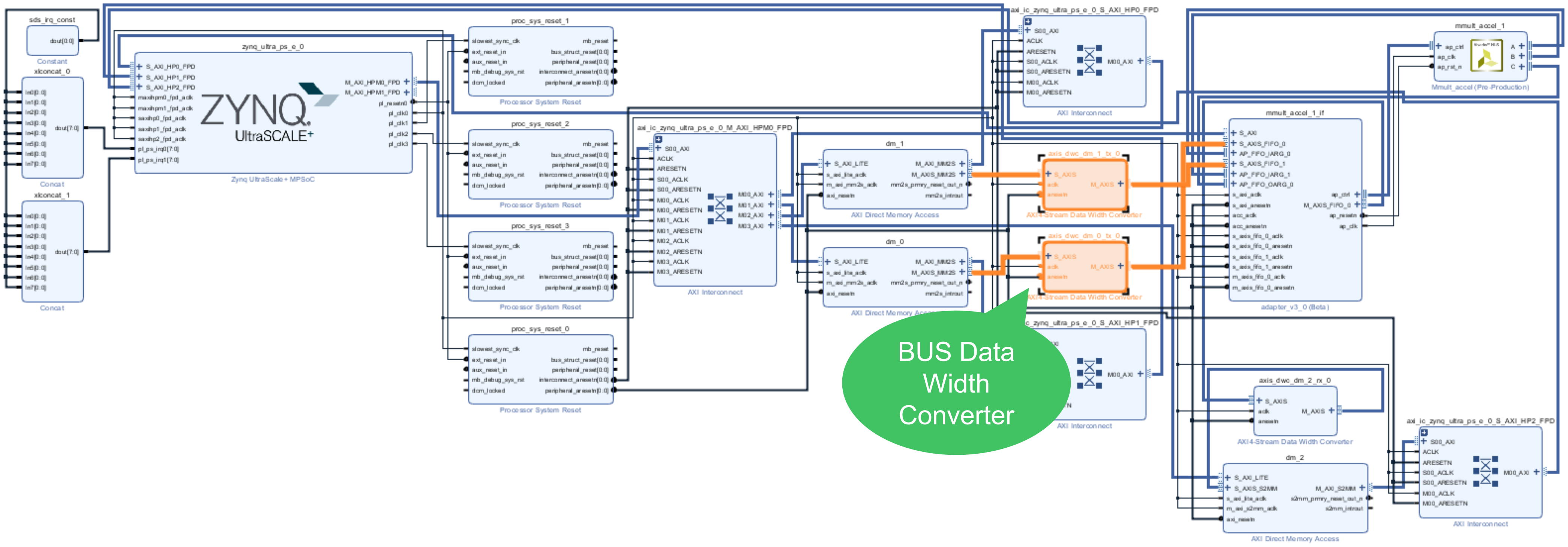
Example: Ultra96 Platform with mmult



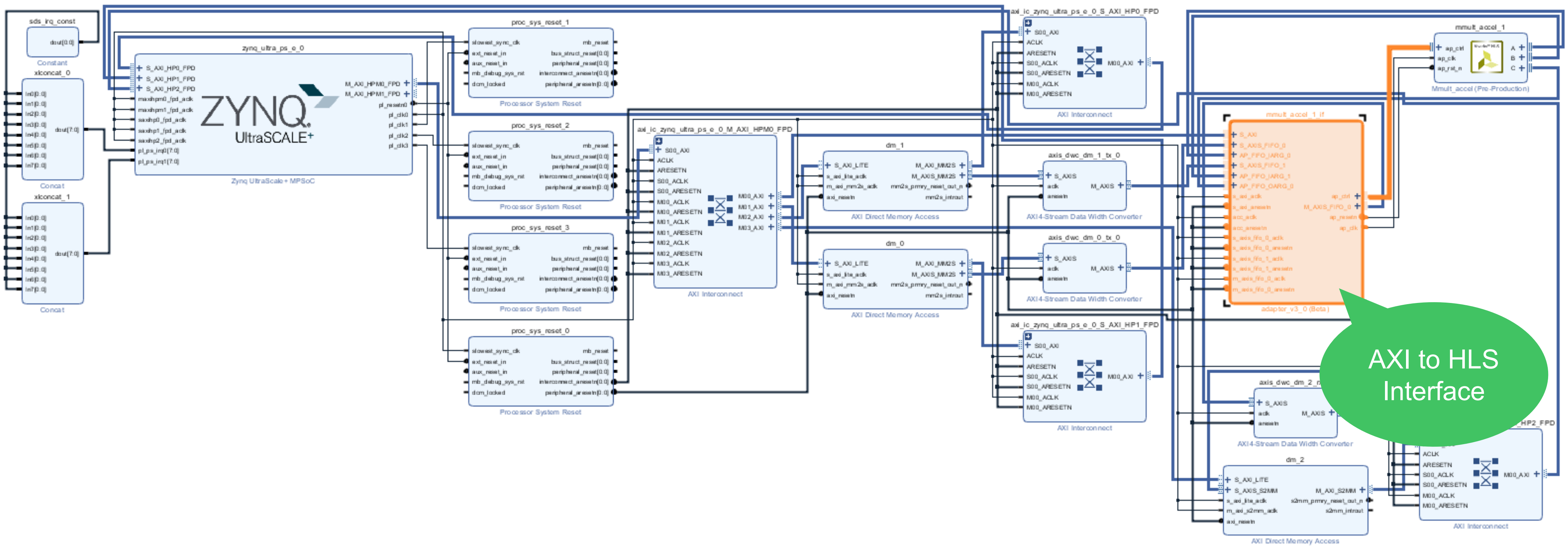
Example: Ultra96 Platform with mmult



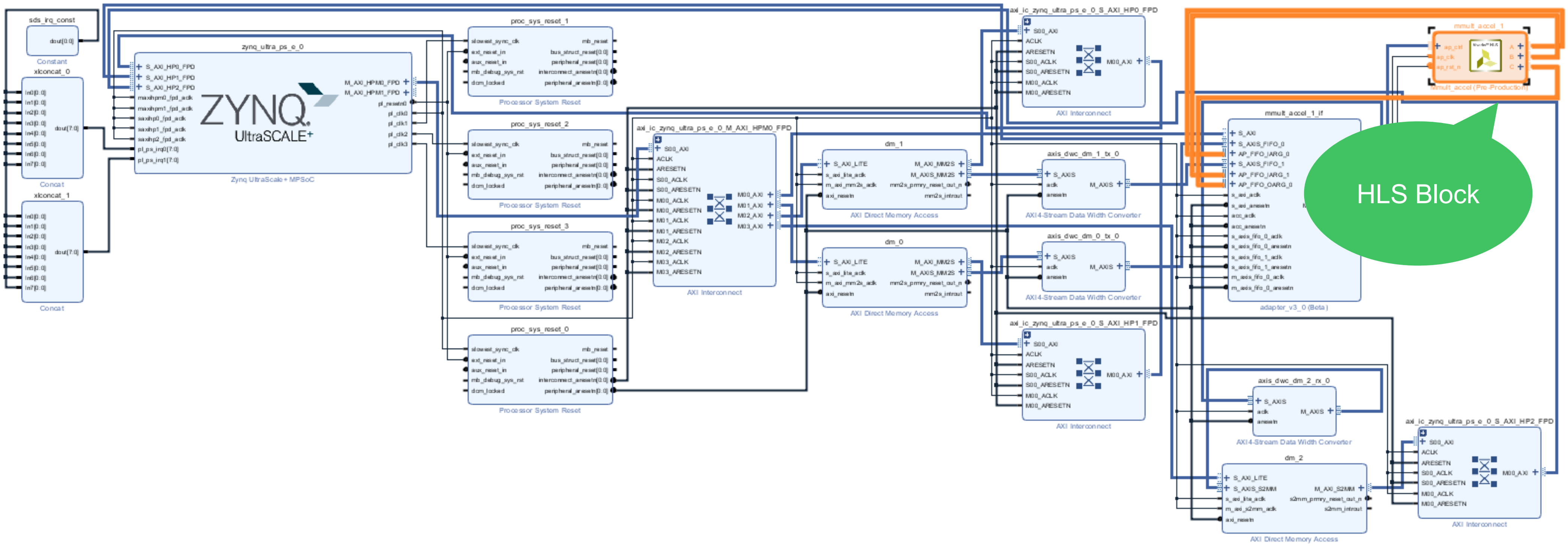
Example: Ultra96 Platform with mmult



Example: Ultra96 Platform with mmult

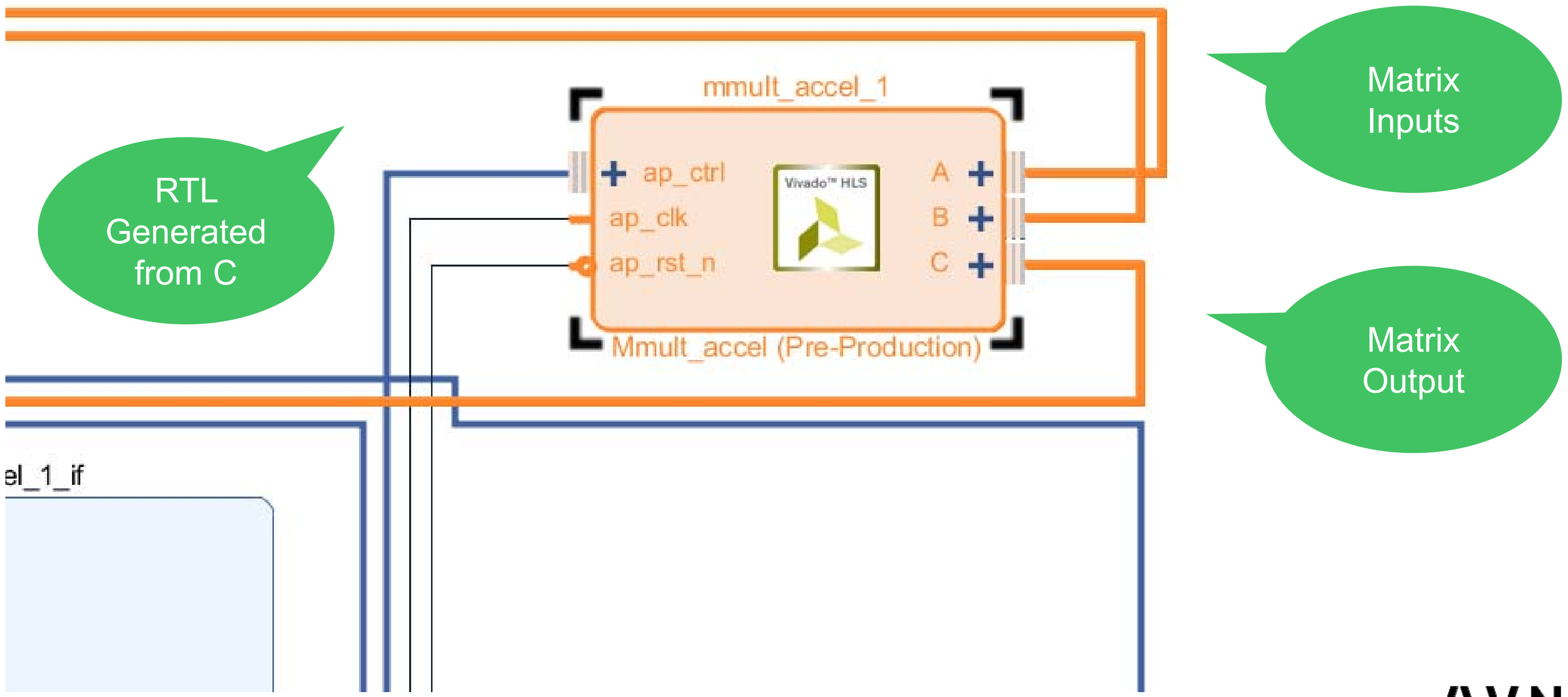


Example: Ultra96 Platform with mmult

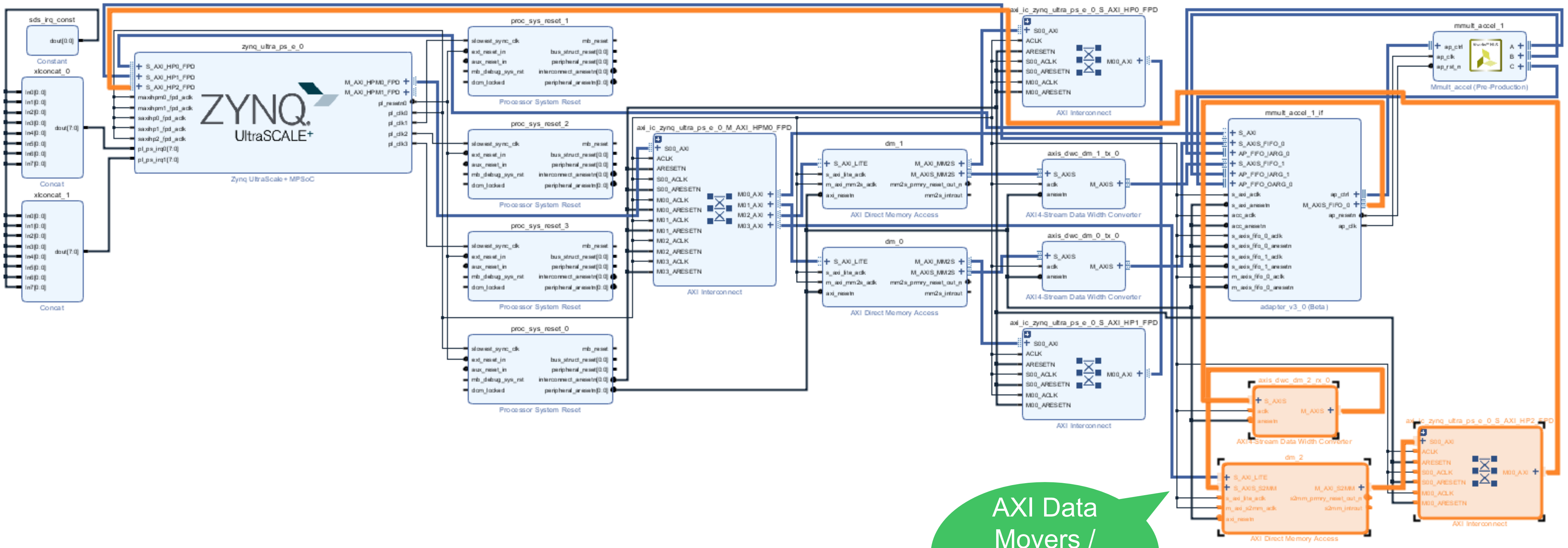


AVNET

Example: Ultra96 Platform with mmult



Example: Ultra96 Platform with mmult



AXI Data Movers / DMA back into PS

AVNET

Hardware Platform

Device Support Archive

- Device Support Archive (DSA)
  - Design in Vivado
  - Debug in Vivado
  - Ensure meeting timing with proper constraints
    - Most basic systems work with inherited constraints
  - Packages are signed and in the FUTURE bad signatures will cause BUILD STOPS

- You need to define what hardware is available
  - Xilinx has created a specific set of TCL commands for this
  - **Or use a manufacturer provided support package**

Device Support Archive

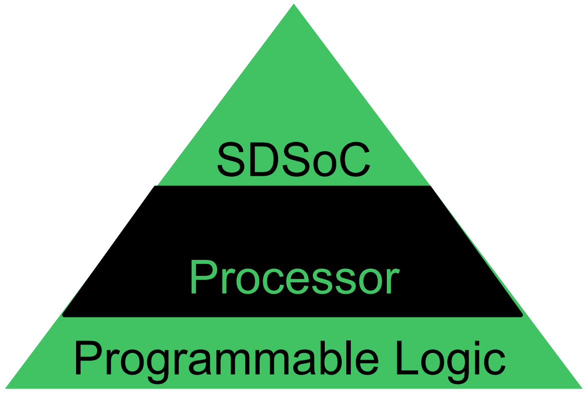
- In other words
  - DSA is our sandbox of resources
  - Provides the framework for the container of the solution space / hardware that is used
  - Provides the structure of the available hardware for solution generation

  - You can think of the DSA as a definition of not only the size of the sandbox, but what toys are provided and which are available to be used by the solution

One way to think about this…

- SDSoC works in layers
  - UG1146’s work flow follows this same methodology!

- This can be thought of as a pyramid
  - The Device Support Archive is the base



Software Platform

Generating the Software Platform

- Software Platform (SPFM)
  - Design in SDK / Ubuntu Virtual Machine
  - Debug using SDK / Ubuntu Virtual Machine
  - Ensure all necessary drivers are defined and functional
    - Test your platform

- You need to define what software is being used
- This includes
  - Tested, working Linker Script
  - Build Flags / Commands needed for your OS
  - OS definitions (if needed)

- Note: SDx GUI can now self create basic Software Platforms

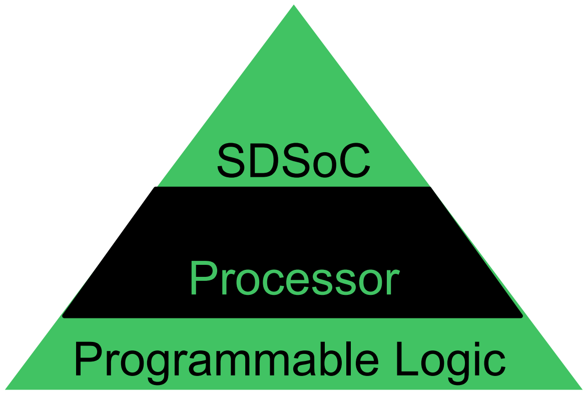
Generating the Software Platform

- In other words
  - This generally leverages custom DSA / hardware builds
  - It can also leverage manufacturer provided files
  - Works hand-in-hand with provided BSP, DSA, etc.
  - Newer versions of SDx can now self-generate most of these files

- Final Platform creation is an extension of this platform

One way to think about this…

- Remember, this can be thought of as a pyramid
  - The Software Definition is the central support
- SDSoC leverages this support structure to place itself at the top



Final Platform Details

Generating the Final Platform

- Platforms are a superset of all the provided details of the configuration / structure of your hardware / software / OS

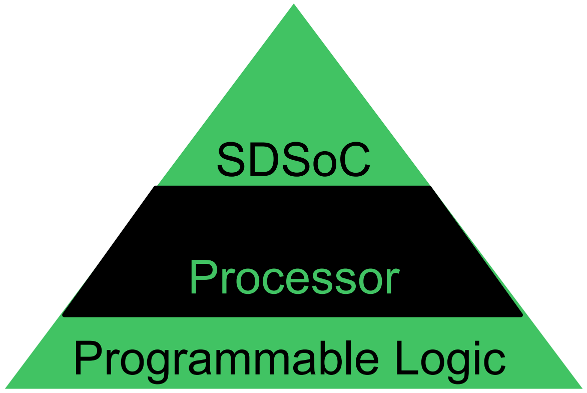
- Platforms can be imported into the tools
  - For easy leveraging of Demo Boards
  - Project retargeting
  - Reduction of time to market

- Avnet and other manufacturers provide SDSoC compatible platforms for download and import into your tools!

One way to think about this…

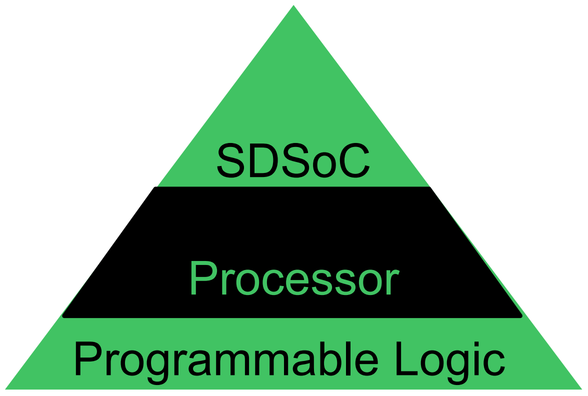
- Remember, this can be thought of as a pyramid
  - The platform itself is the pyramid as a whole

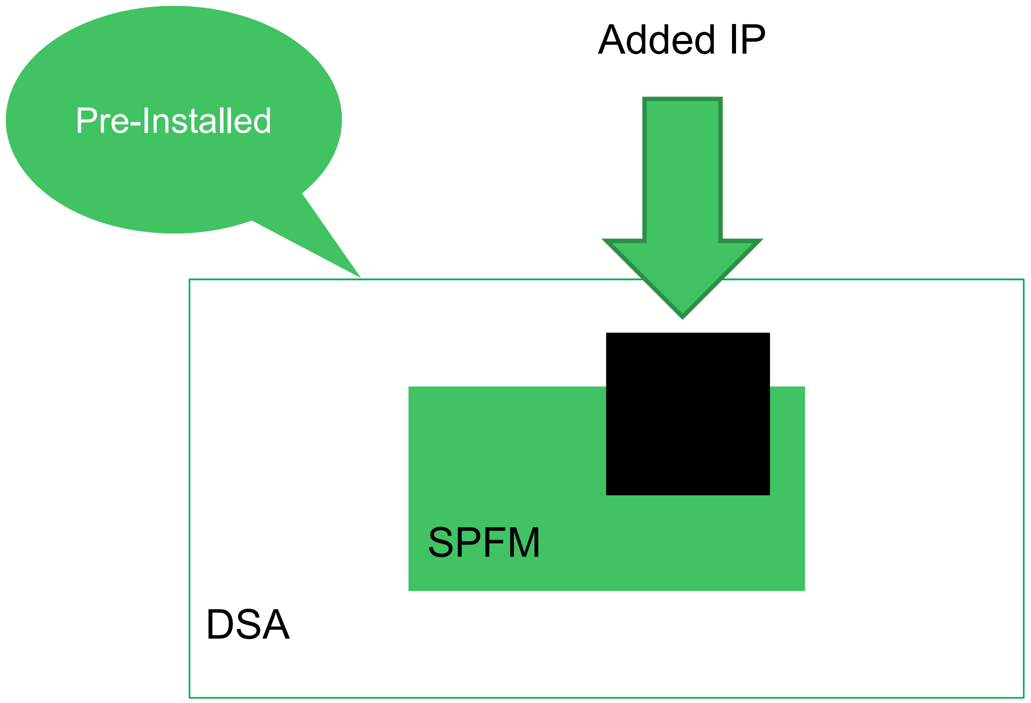
- SDSoC leverages this support structure to create customized solutions which span both software and customized hardware accelerators



Lab 1

LAB 1 – Matrix Multiply on Ultra96



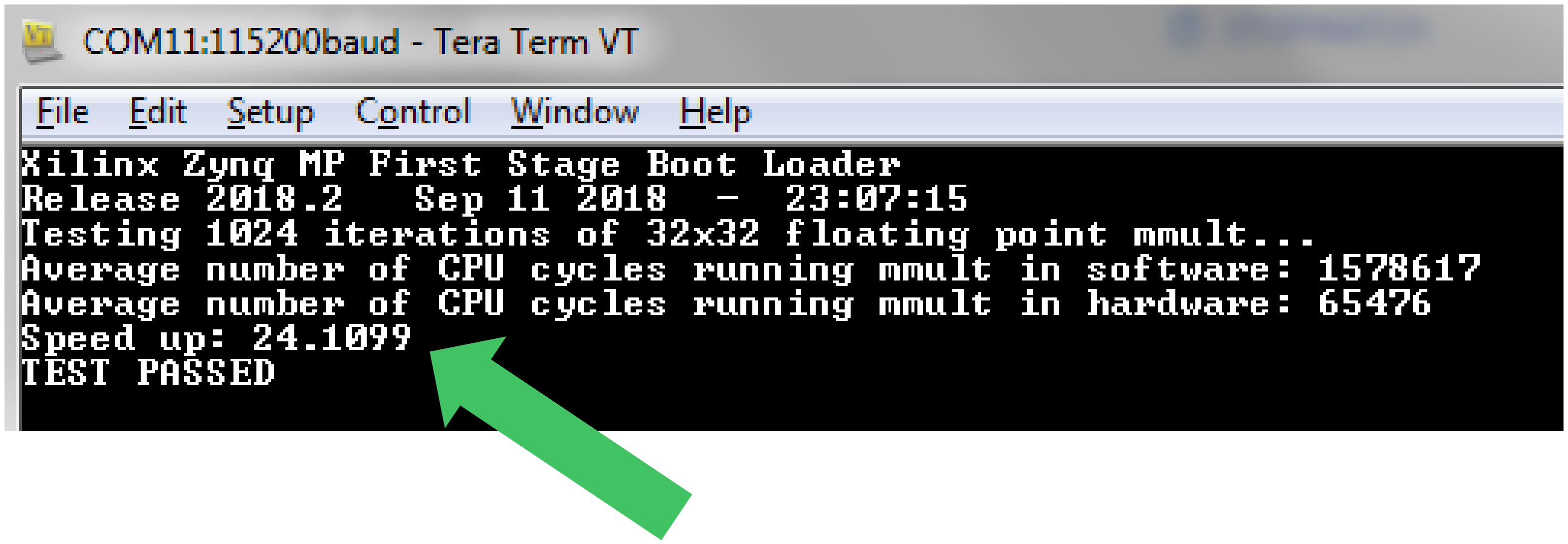


LAB 1 – Matrix Multiply on Ultra96

- Why Matrix Multiply?
  - For our purposes, this is the Hello World of SDSoC!
    - It could be anything
  - The C portion of the project is just “black” box to us
  - Our interest is the **end result!**

LAB 1 – Matrix Multiply on Ultra96

- Goals
  - Using a provided platform, provided example USE the tools
  - Get a big picture about what SDSoC does for you


COM11:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
Xilinx Zynq MP First Stage Boot Loader
Release 2018.2   Sep 11 2018  -  23:07:15
Testing 1024 iterations of 32x32 floating point mmult...
Average number of CPU cycles running mmult in software: 1578617
Average number of CPU cycles running mmult in hardware: 65476
Speed up: 24.1099
TEST PASSED
```

Thank you!

Additional Resources

Go Further With SpeedWays!

- Continue your SDSoC Education with an Avnet SpeedWay
  - Developing Zynq Software
    - In the Developing Zynq Software Speedway, you will be introduced to Xilinx SDK and shown how it offers everything necessary to make Zynq software design easy. This class will cover these capabilities, including BSP creation, built-in drivers, example C code, interrupts, debugging, flash programming, I2C interface between a TE connectivity Pmod, and where to get more help.
  - Developing Zynq Hardware
    - In the Developing Zynq Hardware Speedway, you will be introduced to the single ARM Cortex –A9 Processor core as you explore its robust AXI peripheral set. Doing so you will utilize the Xilinx embedded systems tool set to design a Zynq AP SoC system, add Xilinx IP as well as custom IP, run software applications to test the IP, and finally Debug your embedded system.

Go Further With SpeedWays!

- Continue your SDSoC Education with an Avnet SpeedWay
  - Integrating Sensors on MiniZed with PetaLinux
    - From within an Ubuntu OS running within a virtual machine, learn how to install PetaLinux 2017.1 and build embedded Linux targeting MiniZed. In the hands-on labs learn about Yocto and PetaLinux tools to import your own FPGA hardware design, integrate user space applications, and configure/customize PetaLinux. Use Xilinx SDK tools to develop/debug user applications and use example application code to capture data from a TE Connectivity HTU21D I2C sensor. Where service is available, an Internet of Things (IoT) application will be used to publish the sensor data periodically to the cloud using the MQTT protocol. Source code for the user applications are included with the hands-on labs for you to leverage as a launch point in your custom design.
  - A Practical Guide to Getting Started with Xilinx SDSoC
    - Using proven flows for SDSoC, the student will learn how to navigate SDSoC. Through hands-on labs, we will create a design for a provided platform and then also create a platform for the Avnet MiniZed. You will see how to accelerate an algorithm in the course lab. This experience should give you the background to assist you in developing custom platforms with custom algorithms, accelerated by SDSoC.

AVNET

Other places to start with SDSoC?

- Get an idea about what the tool can and does do!
  - SDSoC Environment User Guide (UG1027)
  - SDSoC Environment Tutorial: Introduction (UG1028)
  - SDSoC Environment Platform Development Guide (UG1146)
  - Example designs online
    - Xilinx.com, Search “SDSoC Platform Examples”
  - Tool Provided Examples
    - See Lab 1
  - Adam Taylor’s MicroZed Chronicles
    - Start at Part 85
    - https://forums.xilinx.com/t5/Xcell-Daily-Blog/Adam-Taylor-s-MicroZed-Chronicles-Part-85-SDSoC-the-first/ba-p/633707
    - UG1028 for v2018.2 follows a very similar flow

Always a catch?

- NOTE:
  - Ensure you are using proper document versions!
    - Use Xilinx Doc Nav
    - Search DIRECTLY on xilinx.com
    - Add version strings when using Google!
      - “UG1146 v2018.2”

For more Information on HLS or Pragmas

SDx Developer Topics

- https://www.xilinx.com/html_docs/xilinx2017_2/sdaccel_doc/topics/pragmas/ref-pragma_HLS_interface.html

Getting Started with Vivado High-Level Synthesis

- https://www.xilinx.com/video/hardware/getting-started-vivado-high-level-synthesis.html

- Avnet's Blog for Advanced Implementation of C-Callable SDSoC IP
- https://www.element14.com/community/groups/fpga-group/blog/2018/07/16/implementing-fft-accelerators-with-sdsoctm-using-open-source-software-and-c-callable-ip

AVNET