Using the New Low-Cost S32V SOM Development Platform for Vision/ADAS Processing

Kushal Shah

Applications Engineer

AMP

October 2018 | AMF-AUT-T2897

NXP | CONNECTS

Agenda

- Introduction: S32V2xx Processor
- Introduction: S32V2xx EVB & Eco-system
  - Lab - Out-of-box Experience - EVB, VSDK and Tools
- S32 Design Studio for Vision and Graph tools
- Application Building Using
  - Lab - ISP – Image Signal Processor
  - Lab - APEX2 – Image Cognition Processor
- Performance Comparison of APEX2 Accelerator With ARM NEON & A53 Core

This Session

- The NXP development environment consists of evaluation boards, development tools, SDK, drivers and operating systems
- This session uses low cost development board (SBC-S32V234) developed by our partner MicroSys
- Get familiar with different components of a complete ecosystem
  - Linux BSP
  - Vision SDK
  - IDE and Graph tools for ISP and APEX2 engines
  - Performance comparison

Introduction to S32V2xx Processor

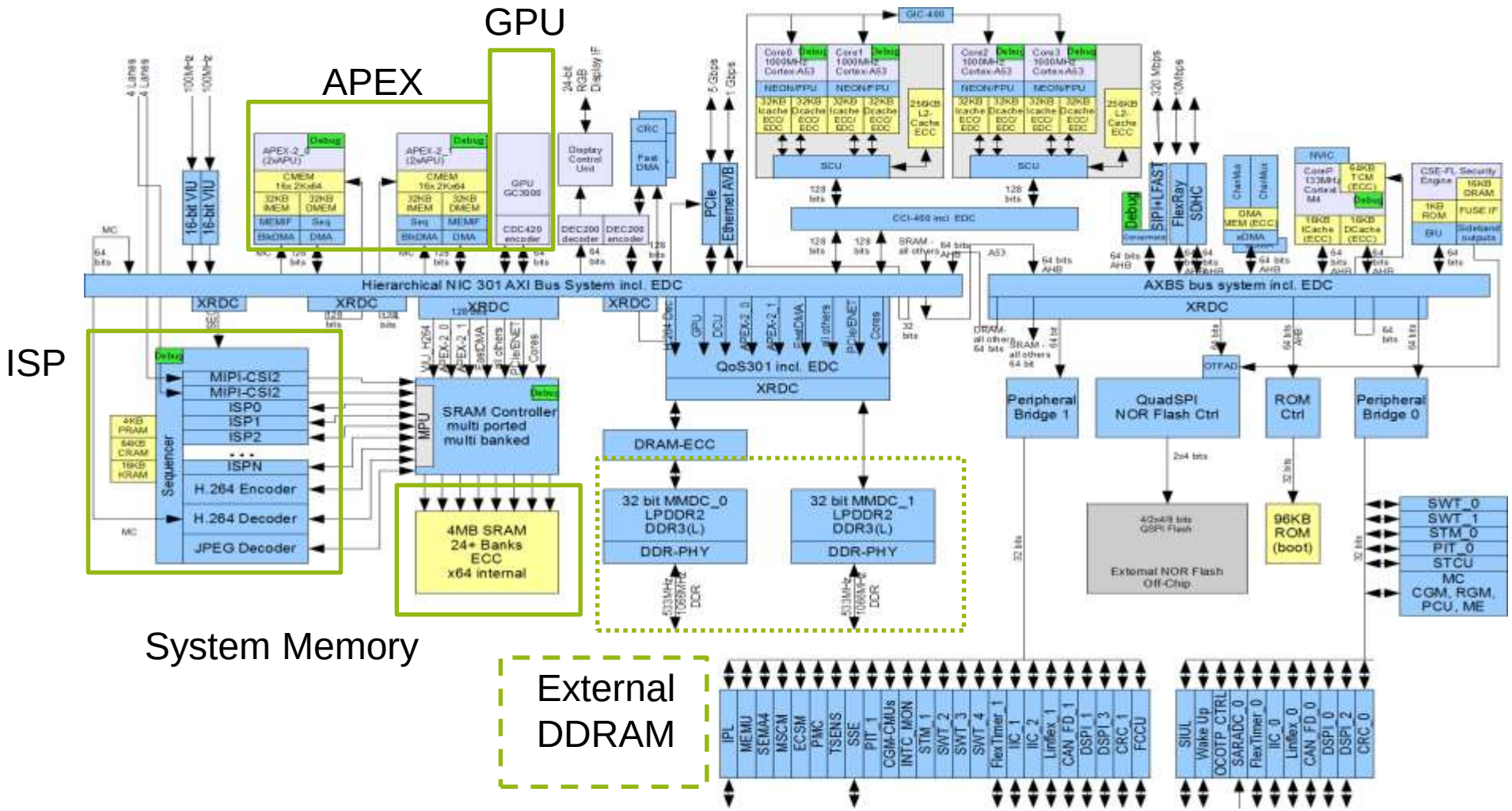
S32V2xx Processor and Use Cases

- Vision Processor
  - Up to Quad ARM® Cortex®-A53 CPUs @1GHz and One Cortex-M4 CPU @133MHz
  - Dedicated accelerators for vision processing
    - APEX2 – Image Cognition Processor
    - ISP – Image Signal Processor
    - GPU – Graphical Processing Unit
  - Developed according to ISO 26262 with an integrated safety concept

- Automotive Applications: Front view, Surround view and Data Fusion systems

- Industrial Applications: Driver monitoring, Surveillance, Drone applications, Industrial vision for Safety etc.

S32V2xx System Architecture



S32V2xx Processing Pipeline

Image Pre-Processing
- ISP
- Multi-streams Connectivity

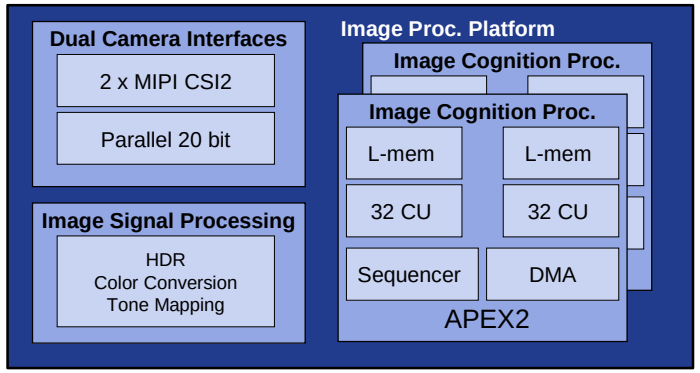
Features Extraction
- Corners,
- Edges,
- Intensity gradients
- shapes

Features Classification & Prediction
- SVM
- Adaboost
- K-nearest Neighbor

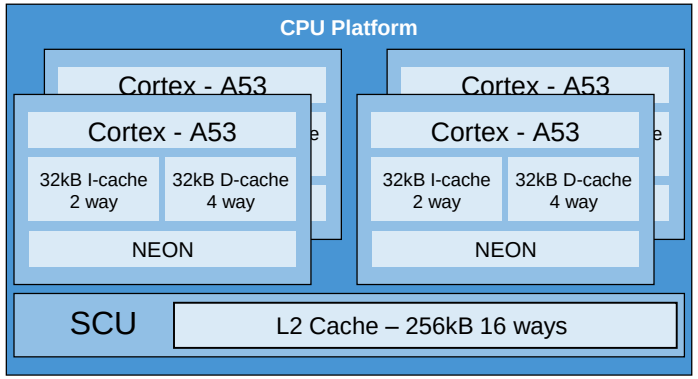
Multi Image Processing
- Tracking, Motion Estimation
- Optical Flow & Disparity
- Stitching

Object Recognition & Fusion
- Object Recognition/Pedestrian
- Augmented Reality
- Face Detection

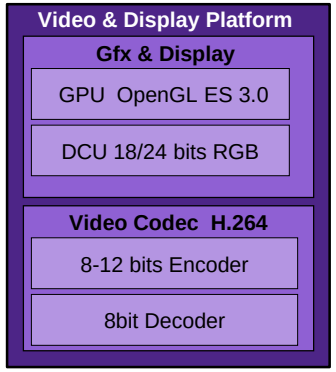
Gfx Overlay & Video Distribution
- Safe Fusion
- Graphic Overlay & Display
- 2D vs. 3D Projection



High Bandwidth Operations
Scalable MIMD Local Memory
Soft ISP



Scalable RISC – Data Fusion
SIMD Co-Processor - Neon
Memory Hierarchy and coherency

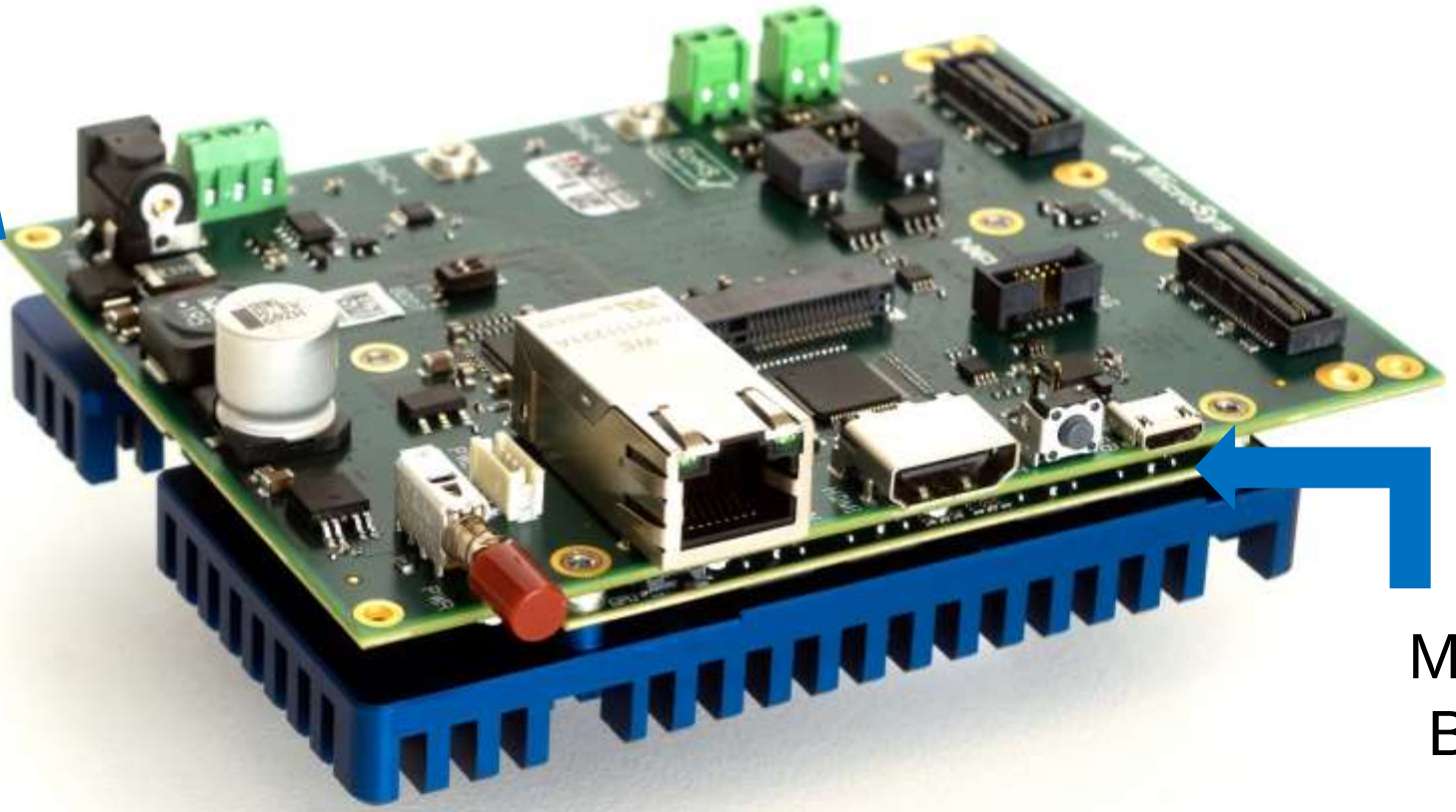


Graphic
Video Codec
Smart Display

Introduction to S32V2xx EVB & Eco-system

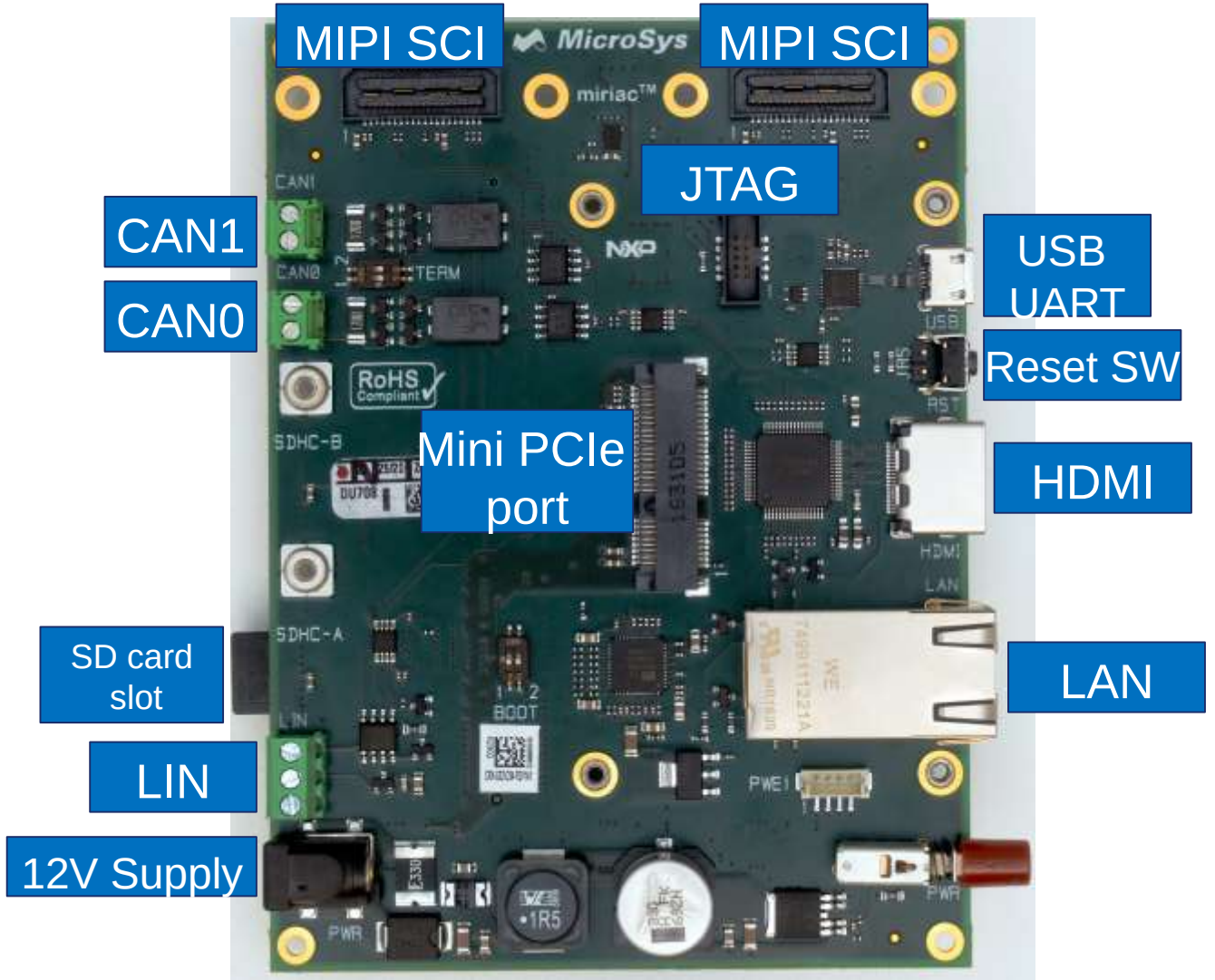
System on Module(SOM) Board: SBC-S32V234

Carrier board

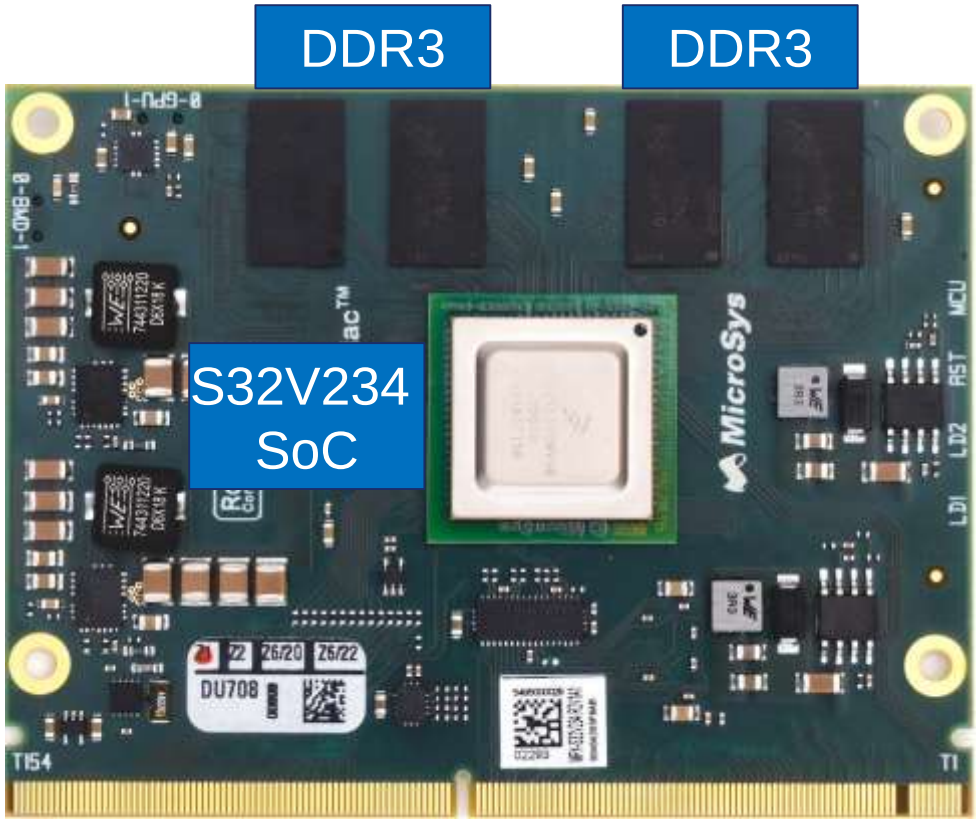


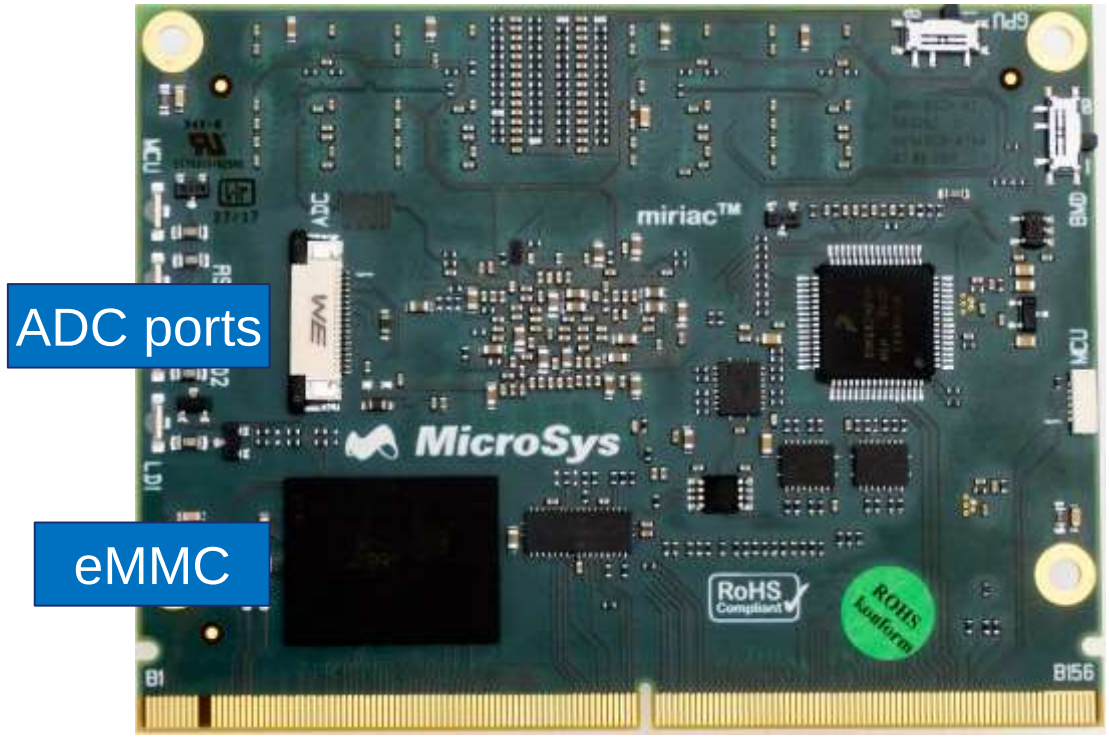
Module Board

SBC-S32V234: Carrier Board

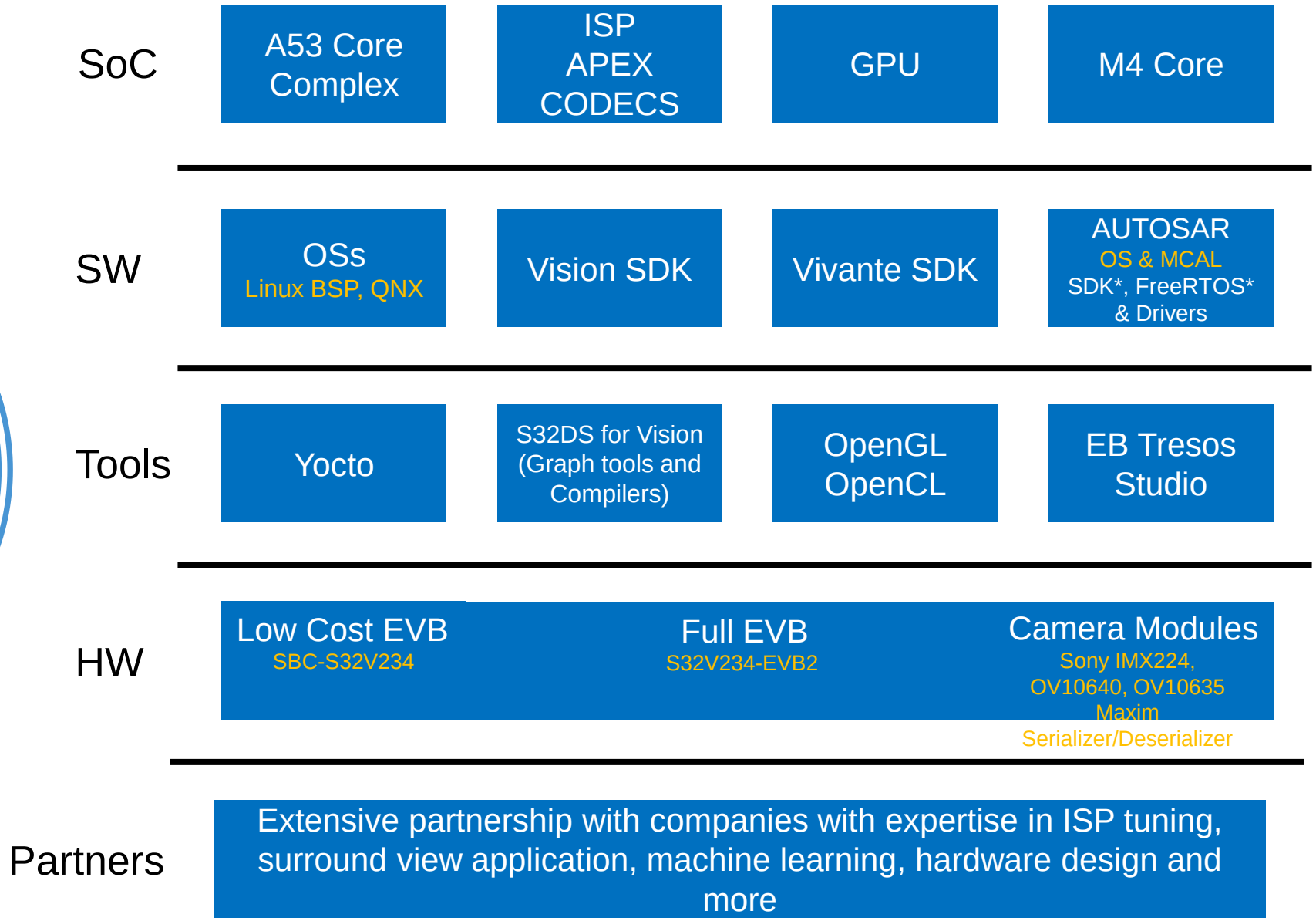


SBC-S32V234: Module Board





Ecosystem: A high-level picture



* EAR Q4 2018
SBC-S32V234: Linux BSP support in end of March in BSP v16.1

Documents, Software and Tools

- Documents
  - Device:
    - Reference Manual(RM), Data sheet, Security RM
  - EVB:
    - User guide, Schematics, Quick Start Guide
  - Application notes
- Camera modules
  - Sony - IMX224, OmniVision - 10635&10640
  - Maxim Deserializer - MAX9286
  - Schematics
- Tools:
  - S32 Design Studio for Vision IDE
  - Vision Toolbox for MATLAB™
  - Partner Tools
- Software
  - NXP Linux BSP
  - Vision SDK for APEX, ISP & Codecs
  - Vivante SDK for GPU
  - FreeRTOS & SDK (M4 only, EAR Q4 2018)
  - AUTOSAR OS(M4 only) & MCAL(both A53 and M4 cores)
  - Drivers:
    - Inter Platform Communication Driver
    - AVB video listener
    - A53 and M4 structural core self test software
    - Memory built in self test software
    - Safe boot

Refer Quick Start Guide for detail information
nxp.com/s32v
nxp.com/sbc-s32v234

S32V Partner Support Network

(Select logo for URL)

Hardware

S32V Training

Premium Toolchain Software

Distribution & E-tailer Support

Vision Software Partners

Camera Image Partners

Vision SDK

- APEX libraries
  - APEX-CV
  - APEX-native
  - APU programming and ACF support
  - APEX emulation support
- ISP libraries
  - Basic ISP kernels
  - MIPI-CSI2, VIU support
- Codec Libraries
  - H264 encoder/decoder and JPEG decoder support
- IO Drivers
  - Display control unit, camera, I2C, UART, QSPI
- 3rd party SW Support
  - OpenCV, pthread, ffmpeg, boost, eigen etc
- Latest Linux BSP Image
- Demos
- Documents

S32 Design Studio for Vision IDE

- Eclipse Neon 4.6 Framework Based IDE
- Built-in Tool Chains
  - GNU Tools for ARM® Embedded Processors (launchpad) build (6.3.0 20180215 and 4.9.3 20170510)
  - NXP ARM GNU Compilers 6.3 for 32-bit and 64-bit bareboards and Linux
  - NXP APU toolchain
  - NXP ISP assembler
- Built-in Libraries
  - Vision SDK
  - OpenCV
- Debug Support
  - NXP S32 Debugger support
  - P&E Multilink/Cyclone/OpenSDA (with P&E GDB Server)
  - Lauterbach premium
  - Linux Remote debug
- Integrated DDR Configuration & Validation Tool

S32 Design Studio for Vision IDE

- Graph Tools
  - APEX & ISP graph tool
  - Use case:
    - Vision pipeline development for that accelerator
    - Uses Vision SDK libraries to generate code
    - Generated code can be used in main application

NXP Linux BSP

- Linux kernel version 4.14.34
- Customizable using Yocto distribution
- All binaries are vanilla without VSDK support
- Binaries with VSDK support can be found in VSDK >> OS folder
- Support to more packages can be added by Yocto project

Lab 1: Out-of-box Experience – EVB, VSDK and Tools

- Task
  - See printed instructions
- Learn
  - Get started with SBC-S32V234
  - Get started with S32 Design Studio for Vision IDE
    - Establish SSH connection with PC
  - Get to know S32V234 Linux and vision SDK
- DIY
  - Run different built-in demos

S32 Design Studio for Vision

Live Demo

ISP – Image Signal Processing

ISP Applications

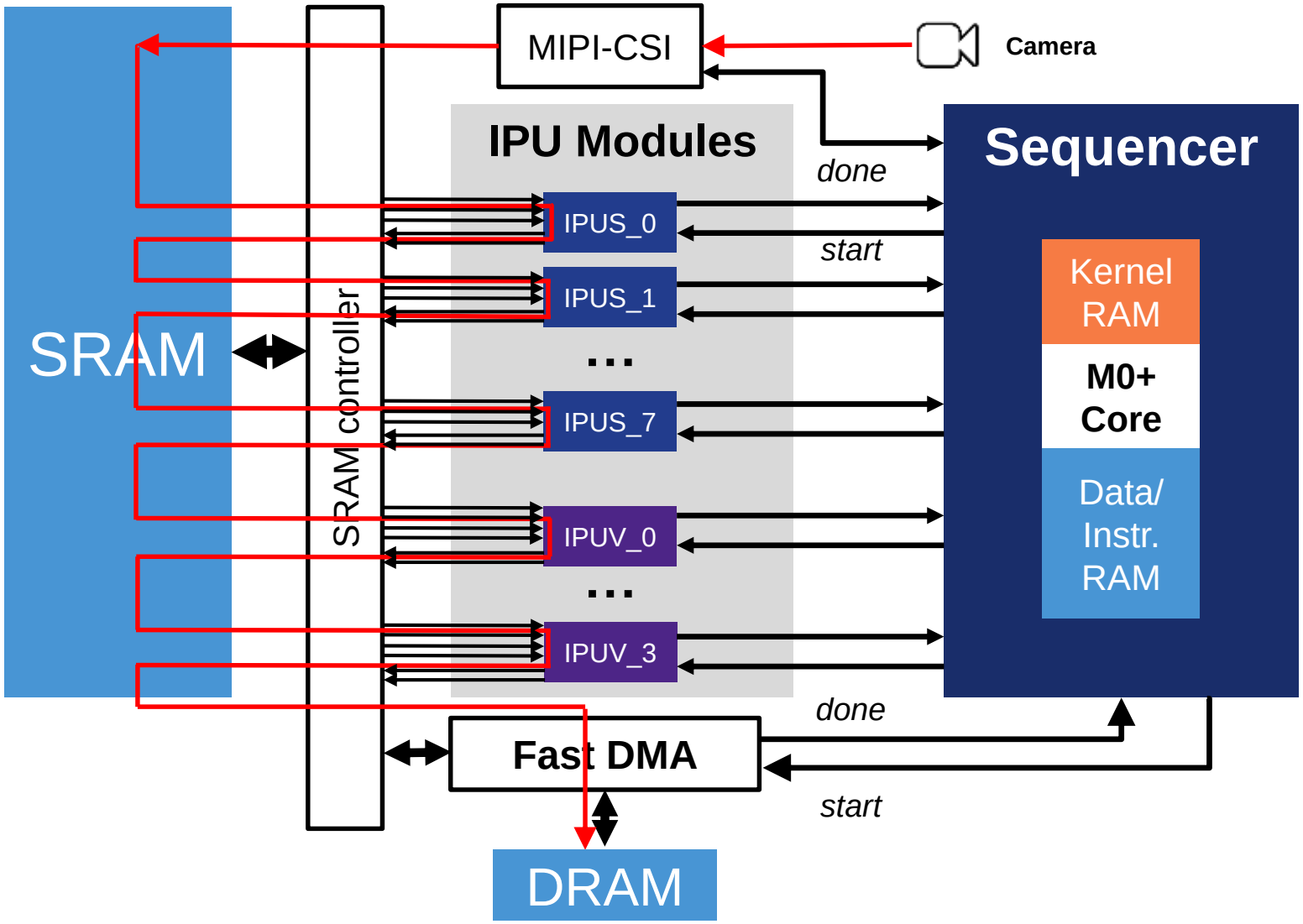
- De Bayering
- Dead pixel
- Black level correction
- Exposure
- White balancing
- HDR
- Vignetting
- RGB2YUV
- Noise Filter
- Format Conversion

Full processing for
Up to one 2Mpix camera @ 30fps!
Partial processing for
Up to 4 x 1Mpix cameras @ 30fps

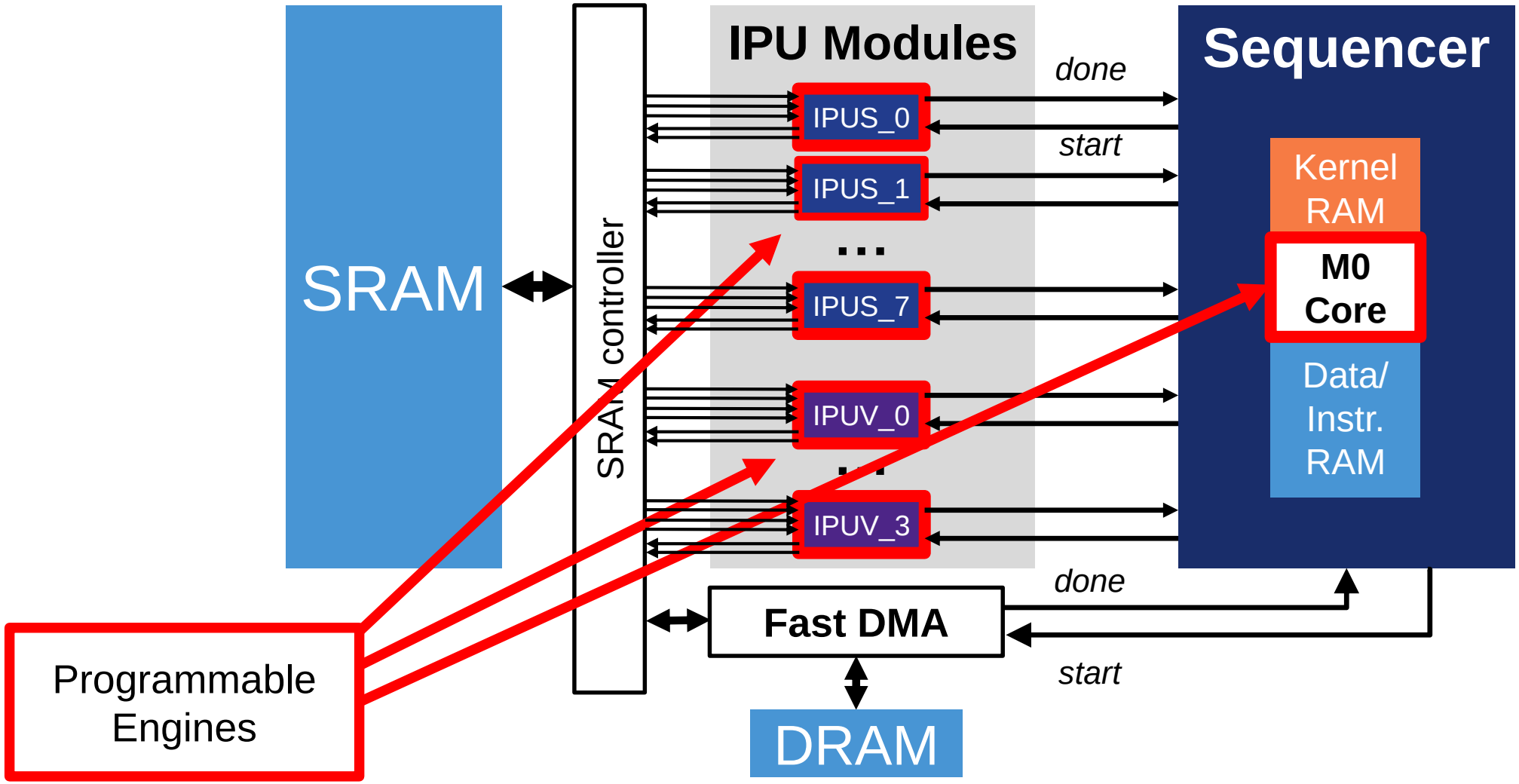
ISP: The Engines (IPU's: Image Processing Units)

- Multiple engines in MIMD pipeline working on a sliding input window
- 8x Scalar Engine
- 4x 4-way Vector Engine
- 500MHz per Engine: 6000MHz total
- User configurable and programmable
- StreamDMA engines:
- The StreamDMA supports several features: scaling, inverse scan, padding at frame boundaries, different data types (8bit/16bit)
- Pixels are 16bit fixed point data in signed or unsigned mode

ISP Sub-system



ISP Sub-system



Lab 2: ISP Processing Using Visual Graph Tool

- Task
  - See printed instructions
- Learn
  - Get started with S32 Design Studio for Vision
    - Make project from built-in examples
    - Make, compile and debug new application
  - How to use visual graph tool to configure ISP pipeline
  - Modify ISP kernel
- DIY
  - Change MIPI port in graph and rebuild the project

APEX2 – Image Cognition Processor

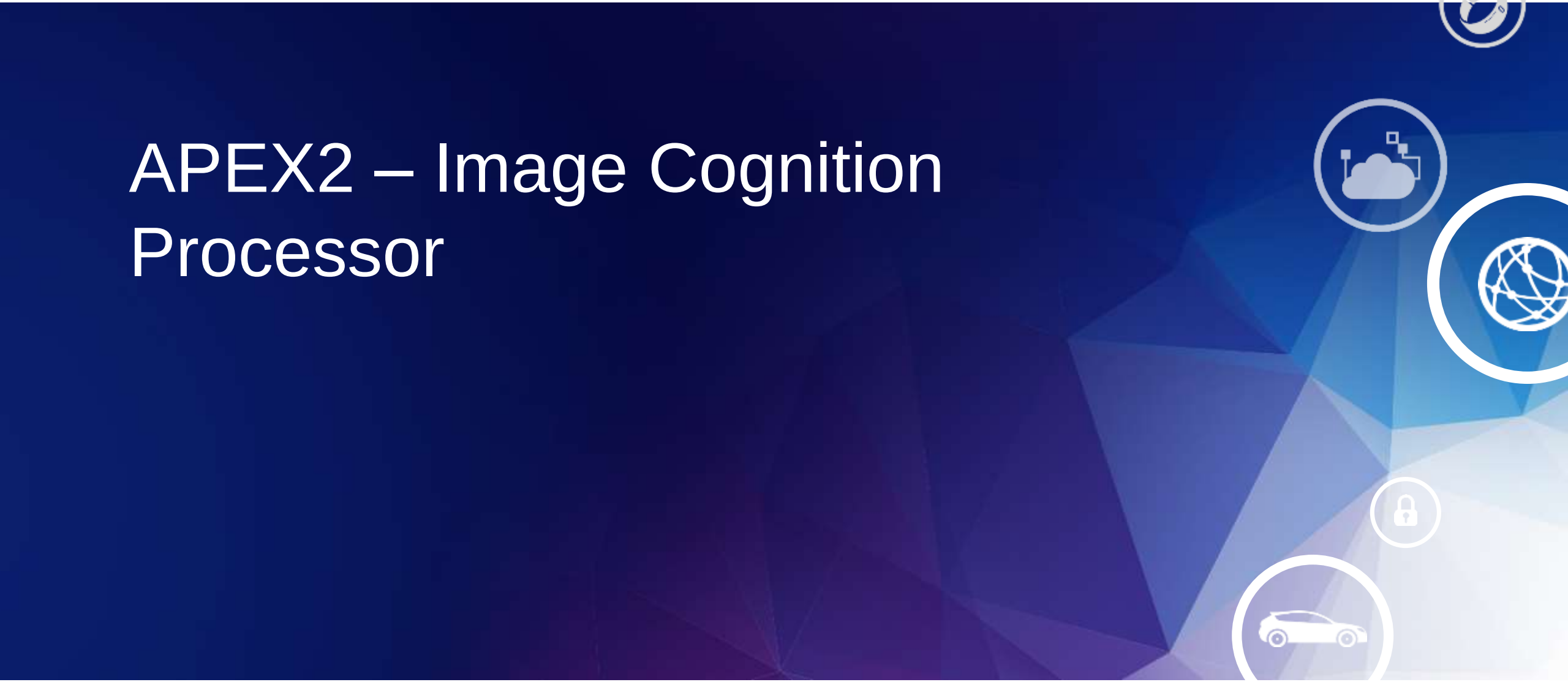
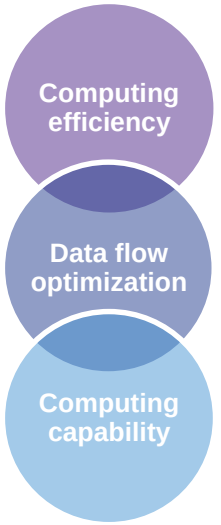
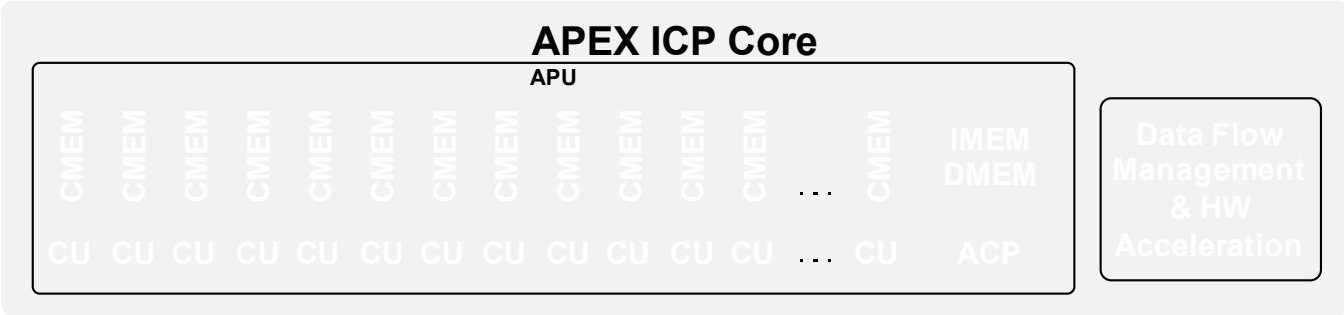


Image Cognition Processor (ICP) Architecture

- APU: Processor inside the APEX engine
- APU is a massively parallel processor
  - Single Instruction Multiple Data (SIMD) Machine
  - Computational Units (CU) + local dedicated memory (CMEM)
  - Array Control Processor (ACP)
    - ACP manages execution and data movement
    - Dispatches instructions to the different CUs
    - Performs address decoding
    - Execute scalar operations
- Data Flow Management
  - Data transfers via DMA
  - HW acceleration

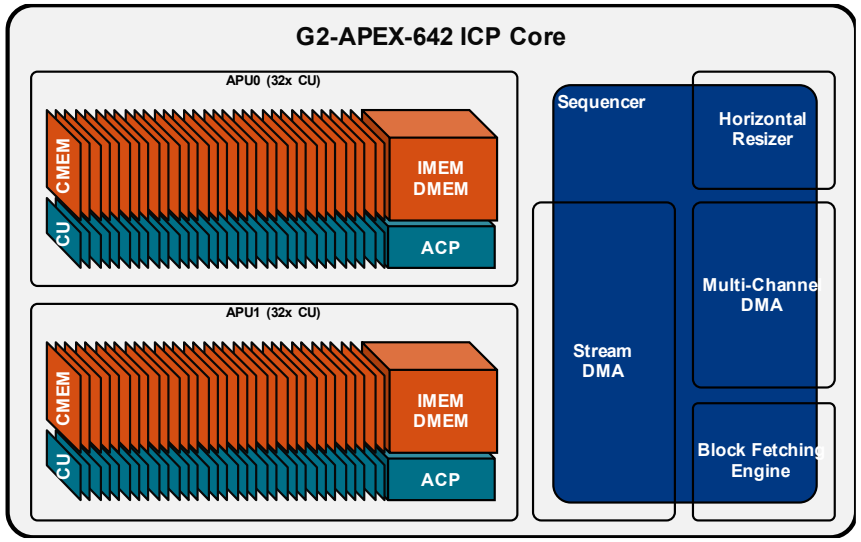




A New Class of Processor

- Image Cognition Processor
  - G2-APEX ICP Core (x2) present in S32V234
    - 64 CUs/APEX, 256KB CMEM/APEX (32 x 4KB)

- APEX ICP is different from CPU, DSP and GPU
  - ICP massively parallel, SIMD, local memory, advanced DMA, int
  - CPU/FPU, limited parallelism, cache memory, float/int [non-linear]
  - DSP, limited parallelism, cache memory, simple DMA, int, [1D]
  - GPU, massively parallel SIMD, mostly optimized for float, high power [float]
  - Other, massively parallel SIMD/MIMD, cache or shared memory, simple or no DMA, int [simpler 2D]



Enablement

- APEX-CV
- APEX-native

APEX-CV Overview

- APEX-CV is distributed as part of Vision SDK
  - Extensive Computer Vision libraries, customer driven OpenCV or OpenVX algorithms, optimized for the APEX cores for fast development of high performance Computer Vision algorithms.
  - Simple Host API implementation, no APEX coding required

- APEX-CV Base: Library of essential computer vision filters optimized for APEX cores
- APEX-CV Pro: Advanced feature detectors and algorithms optimized for APEX cores

- Accelerating customers and partners to market

APEX-CV Base (as in VSDK 1.2)

- Interpolation:
  - Bilinear Grayscale
  - Bicubic Grayscale
  - Linear Grayscale
- Image Filters:
  - Bilateral filter
  - Box filter
  - Census filter
  - Convolve filter
  - Dilate filter
  - Derivative filter
  - Erode filter
  - Gaussian filter
  - Median filter
  - Prewitt filter
  - Saturate Filter
  - Separable Filter
  - Sobel filter
  - Scharr filter
- Histogram
- Integral Image
- Arithmetic Operations:
  - Absolute difference
  - Accumulate
  - Accumulate squared
  - Accumulate weighted
  - Addition
  - Bitwise AND, NOT, OR, XOR
  - Count leading zero
  - Magnitude
  - Subtraction
  - Pixels wise multiplication
  - Min, Max
  - Threshold, ThresholdVX
- Color Conversions:
  - RGB565 to RGB888
  - RGB888 to RGB565
  - RGB888 to Y
  - RGB888 to YUV
  - RGB888 to Grey
  - Grey to RGB888
  - Channel extract RBG, RGBX
  - Channel Merge
  - Table Lookup

APEX-CV Pro (as in VSDK 1.2)

- APEX-CV Feature Detection Library
  - Block Matching
  - Binary Robust Independent Elementary Features (BRIEF)
  - Oriented Fast and Rotated BRIEF (ORB)
  - Canny Edge Detector
  - Good Features To Track / Harris Corner Detector
  - HOG Object Detector
  - Aggregated Channel Feature Based Pedestrian Detector
  - Hough Line Detector
  - FAST9
  - LBP Face Recognition
- APEX-CV Image Pyramid Library
  - Gaussian Image Pyramid / Multi-scale Gaussian Image Pyramid
  - Laplacian Image Pyramid
- Histogram Equalization
- APEX-CV Image Transform Library
  - Affine Transformation
  - Image Remap (Dewarping)
  - Image Resize
  - Tone Mapping
- APEX-CV Feature Tracking Library
  - L-K Tracker / Multi-Scale L-T Tracker

Lab 3A: Using APEX-CV Libraries

- Task
  - See printed instructions
- Learn
  - How to use APEX CV libraries into your application using example of Gaussian Filter
- DIY
  - Use different window size
  - Use sobel filter

APU Programming and APEX Core Framework (ACF)

APU Programming

- Allows customers to develop customs functions for APEX
- Simple C/C++ programming

ACF

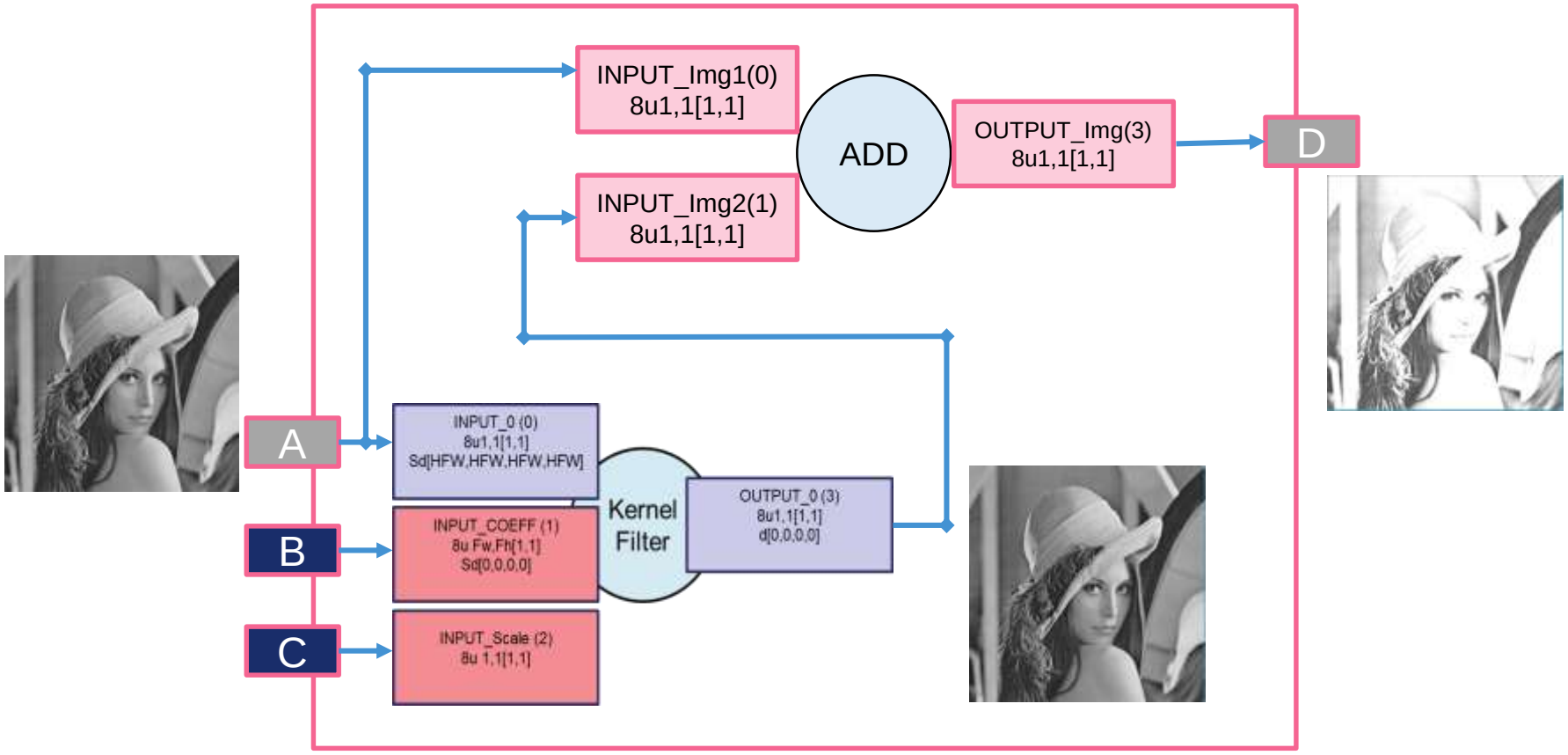
- Allows customers to use custom functions to develop complex algorithm
- Implement and run complex algorithm on APEX
- Visual graph tool

APU programing: APEX Code Example

```
//  dst = srcImage0 + srcImage1
void add(vec16u* dst, vec08u* srcImage0, vec08u* srcImage1, int bw, int bh,
int inStrideW, int outStrideW)
{
    for (int by = 0; by<bh; ++by)  // y data row
    {
        for (int bx = 0; bx<bw; ++bx) // x in the blck_tile
        {
            vec16u a = (vec16u) srcImage0[bx];
            vec16u b = (vec16u) srcImage1[bx];
            dst[bx] = a + b;
        }
        dst += outStrideW;
        srcImage0 += inStrideW;
        srcImage1 += inStrideW;
    }
}
```

We call it – a Kernel

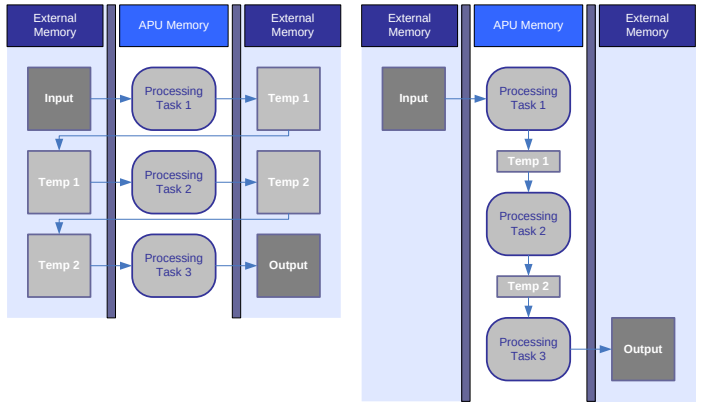
ACF: Kernels and Graph



Why ACF?

- At low level ACF creates a processing pipeline that incorporates the following steps:
  - Transfer data from external/host memory to APU memory
  - Process input data (residing in APU memory) with the APU processor and produce output data (also in APU memory)
  - Transfer output data from APU memory back to external/host memory

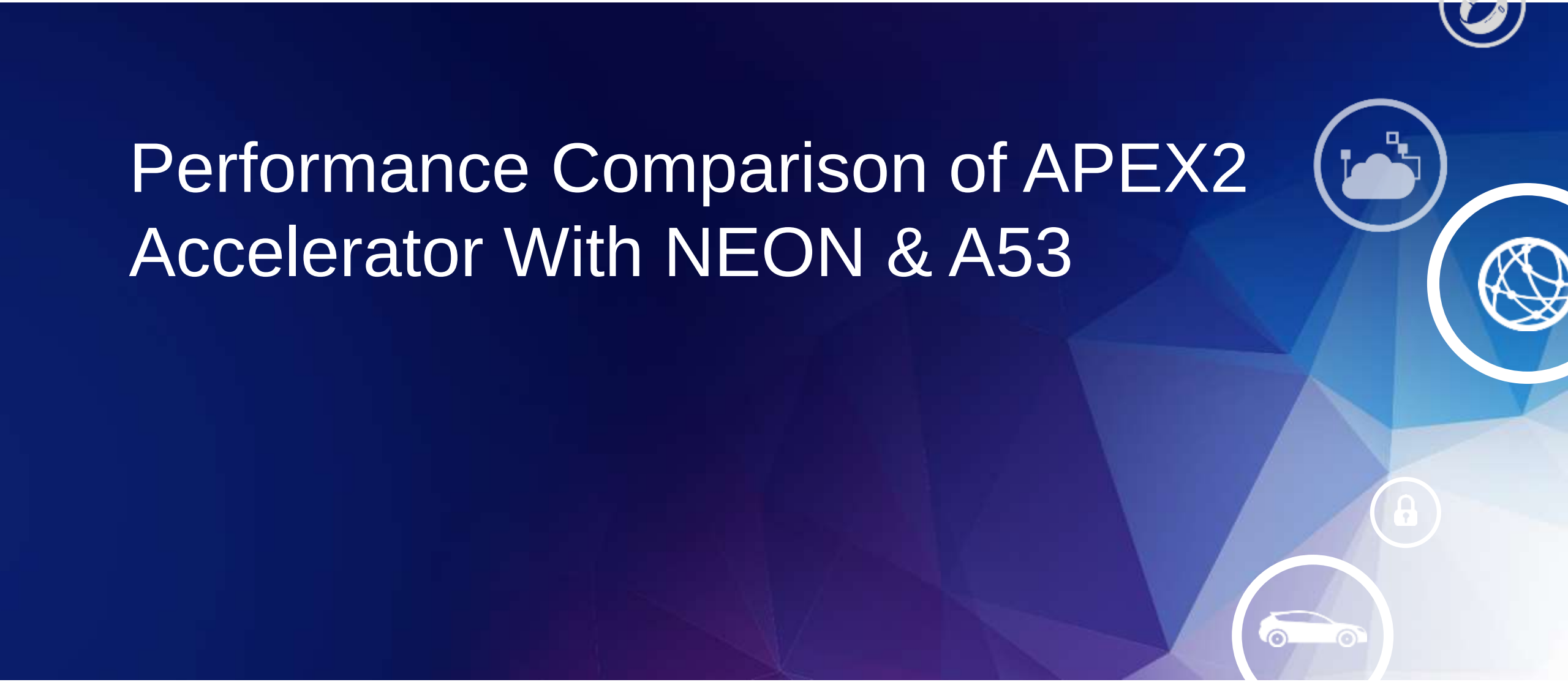
- Minimizing the cost associated with data transfers is accomplished by:
  - Pipelining *data transfers* with *processing* to ‘hide’ the cost of moving data to and from host and APU memory.
  - Combining multiple processing tasks into a single process, allowing the framework to take advantage of data locality and local intermediate results



Demo: APEX Programming Using Visual Graph Tool

- Task
  - See printed instructions
- Learn
  - How to use visual graph tool to make your vision algorithm that runs on APEX core
- DIY
  - Up-sample image 4 times

Performance Comparison of APEX2 Accelerator With NEON & A53



Lab 4: Compare Performance – APEX vs A53 & APEX vs Neon

- Task
  - See printed instructions

- Learn
  - Compare performance of APEX2 engine compared to A53 cores and neon for the same task
  - Understand value of APEX2 accelerator for vision processing tasks

Compile Vision SDK Demos

Demo: Compile and Run Vision SDK Demos

- Download standalone Vision SDK
- Go to *…\VisionSDK_S32V2_RTM_1_2_0\s32v234_sdk*
- Open *runShell*
- Set *Environment Variables*
  - Environment variables for ARM compilers are already set
  - Set environment variables for APU compilers
    - *export APU_COMP=nxp*
    - *export APU_TOOLS=/c/NXP/APU_Compiler_v1.0/*
    - *export PATH=/c/NXP/APU_Compiler_v1.0/bin/:$PATH*
- Go to the build folder of the demo you want to build
  - E.g. for feature tracking demo, go to *…/VisionSDK_S32V2_RTM_1_2_0/s32v234_sdk/demos/apps/feature_tracking/build-v234ce-gnu-linux-d*
- Issue *make allsub* command
  - This builds demo project along with all dependent libraries
- Copy .elf file to target
- Also copy configuration files to target
  - E.g. for feature tracking demo copy ***data*** folder from *../VisionSDK_S32V2_RTM_1_2_0/s32v234_sdk/demos* to target

Refer to Quick Start Guide for more instructions

Lab Instructions

Agenda

- Lab 1: Getting Familiar With EVB, VSDK and Tools
- Lab 2: ISP Processing Using Visual Graph Tool
- Lab 3A: Using APEX-CV Libraries
- Lab 4: Compare Performance – APEX vs A53 & APEX vs Neon

Lab 1: Getting Familiar With EVB, VSDK and Tools

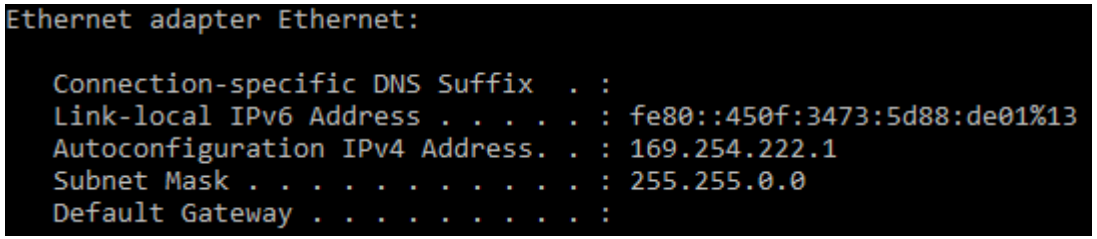
- Part 1: We will boot Linux on the EVB and run one of the examples on the sd-card
- Make sure sd-card is loaded with BSP that supports VSDK and is inserted in side the EVB.
- Connect the USB cable (usb2serial port) to your laptop and wait for windows to tell you the drivers have been installed.
- Check what COM port is used (this is shown when the drivers are installed or in the device manager from windows)
- Open teraterm, select ‘serial’ as connection type, select com port and click on OK
- Go to Setup>> Serial port…, select baudrate of 115200 and click on OK
- Turn on the EVB & wait for Linux to boot. If you see nothing, Linux may have already booted. press enter to see login prompt
- Use ‘root’ as user
- (optional)Set up password:
  - root@s32v234evb:~# passwd
  - (enter the password of your choice)
- Now run the following application:
  - root@s32v234evb:~# vsdk/isp_sonyimx224_csi_dcu.elf (stop this program with ctrl+c)
  - You can also run other demos like….
    - root@s32v234evb:~# vsdk/apex_isp_face_detection_cv.elf
    - root@s32v234evb:~# vsdk/apex_isp_fast9.elf

Lab 1: Getting Familiar With EVB, VSDK and Tools

Part 2: Ethernet setup and SSH connection using S32DS

1. In console type: ifconfig
   - If Ethernet doesn't have ip address assigned, follow steps below to establish static connection
     1. Open command prompt from your Windows pc and type **ipconfig**

        Look for ip address and subnet mask under **Ethernet adapter Ethernet**



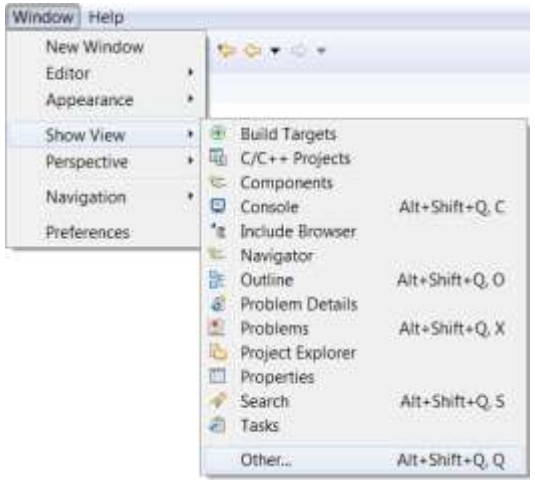
     2. Using terminal window issue following command to S32V234
        - Use ip address and subnet mask according to your network settings as shown above and below

          ifconfig eth0 169.254.185.210

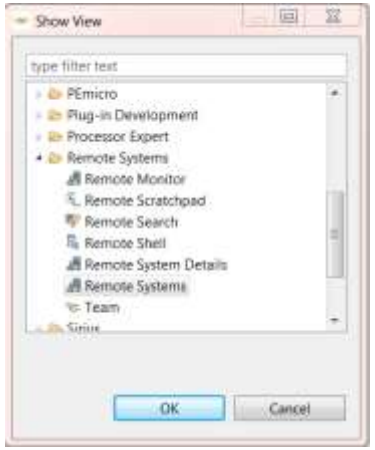
          ifconfig eth0 netmask 255.255.0.0
     3. Ping the connections

Lab 1: Getting Familiar With EVB, VSDK and Tools

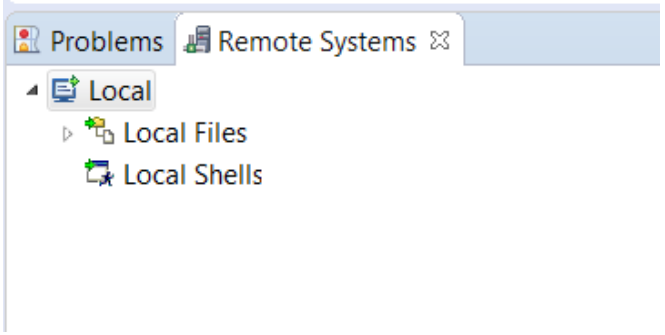
2. Go to S32DS and Window > Show View > Other…



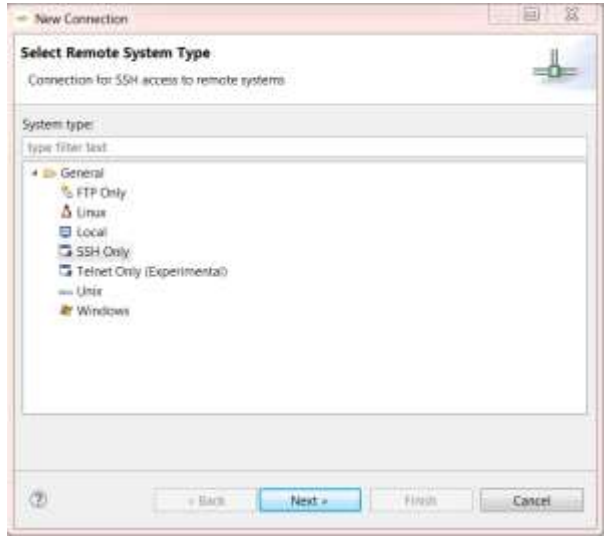
3. Select Remote Systems > Remote Systems



4. You can see following window at the bottom of your screen

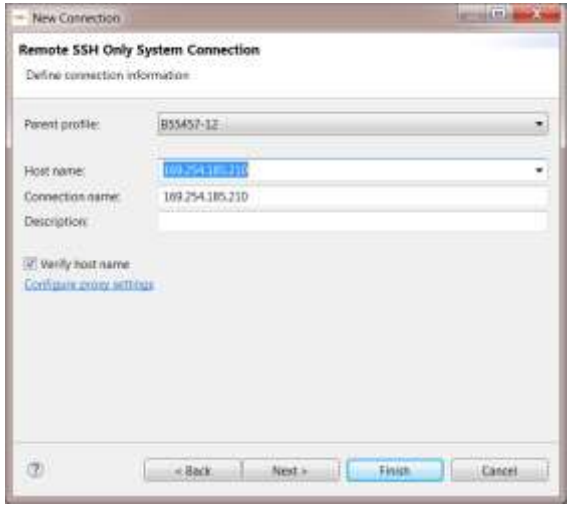


4. Click on  icon on the right side of the window to create new SSH connection and select **SSH Only**

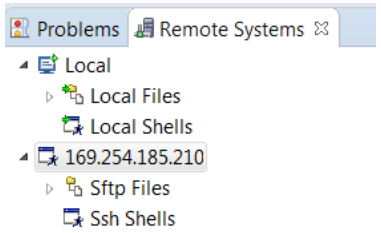


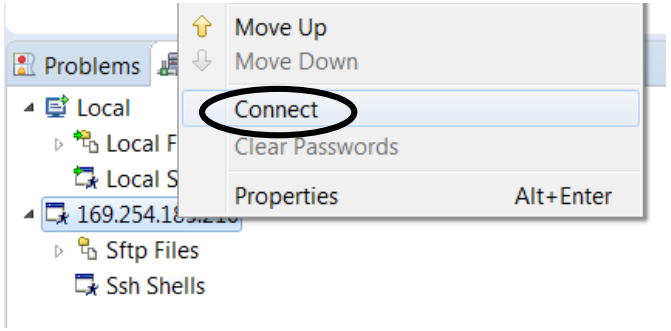
Lab 1: Getting Familiar With EVB, VSDK and Tools

6. Set IP address of SBC-S32V234



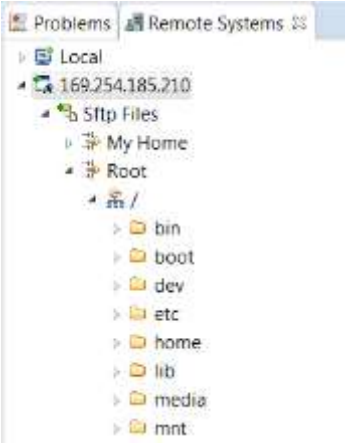
7. You can now see new connection. Now, right click on connection and click on **Connect**.





8. Write user ID and password(if any) to connect to SBC-S32V234. Click “Yes” if any prompt shows up.

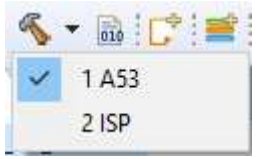
9. You can now browse file system of SBC-S32V234



Lab 2: ISP Processing Using Visual Graph Tool

1. Go to File > New > S32DS Project from example
2. Select S32DS Vision SDK Examples>ISP > isp_sonyimx224_csi_dcu
3. Go to C/C++ prospective

   Click on “Switch to C/C++ Development” option on the screen *or* Go to Windows > Perspective > Open Perspective > Others and select C/C++
4. Compile the project, for **A53**



5. Copy Binaries > s32ds_isp_sonyimx224_csi_dcu_linux.elf file

   and paste it to your SSH connection > Sftp Files > My Home

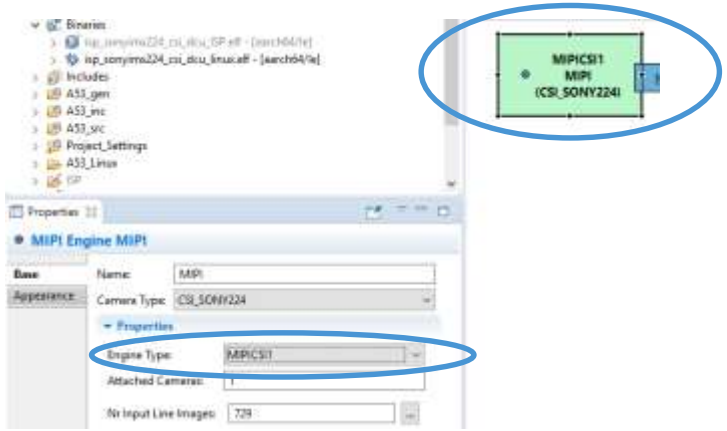
   - while switching prospective you may find “Remote Systems” tab disappeared. Open the new “Remote Systems” tab.
6. Go to terminal window and run following commands

   root@s32v234evb:~# chmod 777 s32s_isp_sonyimx224_csi_dcu_linux.elf (to assign permissions)

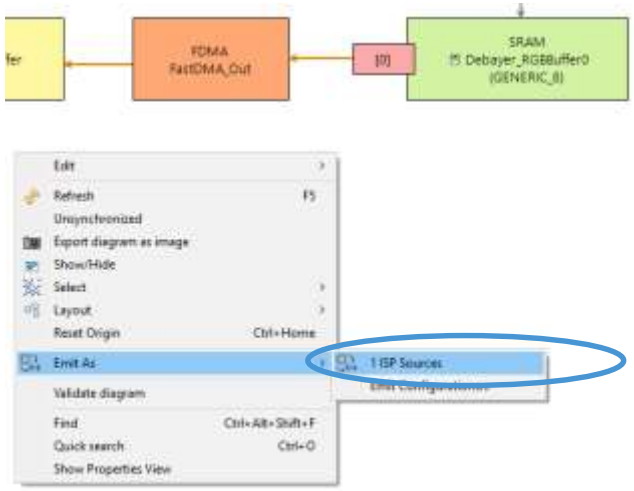
   root@s32v234evb:~# ./s32s_isp_sonyimx224_csi_dcu_linux.elf

Lab 2: ISP Processing Using Visual Graph Tool

7. DIY:
   - Open isp_sonyimx224_csi_dcu_**graph** project
   - Go to Vision prospective  (right-top corner of the window)
   - Select MIPICSI0 block and change Engine Type to MIPICSI1



   - Now, emit the graph configuration by right clicking on the graph screen and clicking on ISP Source



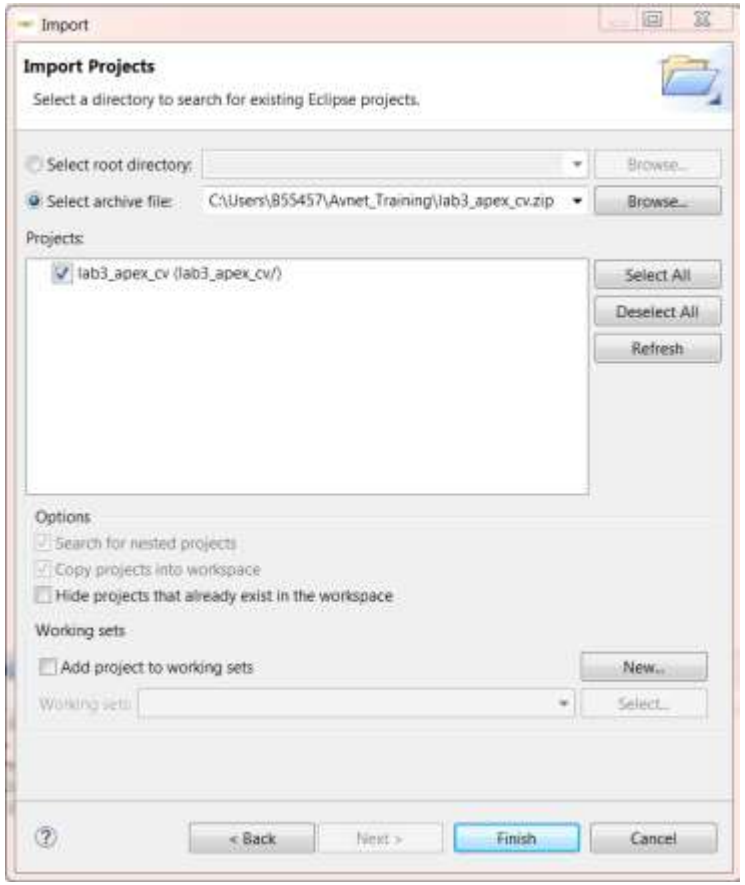
   - Mount camera on MIPI-B port on the SBC.
   - Repeat steps 4 to 6 (Found errors! During compilation.. See next page)

Lab 2: ISP Processing Using Visual Graph Tool

8. Errors!!!
   - Change names of followings in A53_sec>>main.cpp
     - *gpGraph_mipi_simple* to *gpGraph*
     - *&gGraphMetadata_mipi_simple* to *&gGraphMetadata*
     - *FDMA_IX_FastDMA_Out_MIPI_SIMPLE* to *FDMA_IX_FastDMA_Out*
   - *Include the following line in A53_gen>>inc>>mipi.simple_c.h*
     - *extern char sequencer_mipi_simple_srec[ ];*

Lab 3A: Using APEX-CV Libraries

1. Import project “**lab3_apex_cv**”
   1. Go to File > Import.. > General > Existing Projects into Workspace
   2. Go to “**Select achieve file”** option and “**Browse..**” to correct path
   3. Select the project and hit Finish
2. Build the project
3. Copy both **lab3_apex_cv.elf** and **in_grey_256x256.png** to S32V234 EVB using SSH connection
5. Assign the execute permission to .elf file and execute the file
6. Refresh the remote systems console
7. Open **gaussion.png** file from S32V234 file system to see results



Lab 4: Compare Performance – APEX vs A53 & APEX vs Neon

- In this lab we will do gaussian blurring operation on 1280x720 size picture
  - Filter size = 3x3
  - APEX uses APEX-CV
  - Neon is uses assembly code
  - A53 uses OpenCV
- Import project lab4_apex_neon_a53
- Compile the Project
- Copy both lab3_apex_neon_a53.elf and
  - grey.png to S32V234 EVB
  - using SSH connection
- Assign the execute permission to .elf file and
  - execute the file
- Compare run times of all three engines and realize value of APEX engine

SECURE CONNECTIONS
FOR A SMARTER WORLD

www.nxp.com