

ALL PROGRAMMABLE

ANY MEDIA

5G

4K/8K

ANY STANDARD

ANY MACHINE

ANY NETWORK

5G Wireless • Embedded Vision • Industrial IoT • Cloud Computing



Vision with Precision Webinar Series

Video Surveillance

정 웅 부장, DSP specialist, Xilinx
정 승혁 과장, MathWorks

Xilinx Vision with Precision Webinar Series

- Monitoring Things: **Surveillance**
- Perceiving Environment / Taking Action
 - *ADAS and the Road to Autonomous Vehicles*
 - *Drones & Other Vision Guided Robotics*
 - *Augmented, Virtual and Mixed Reality*



Machine Vision



Surveillance



Medical Imaging

Differentiate by Design



Agenda

- Embedded Vision Market Trends
- Surveillance Technology Trends
- Introducing The MathWorks
- The MathWorks Solutions
- Conclusion



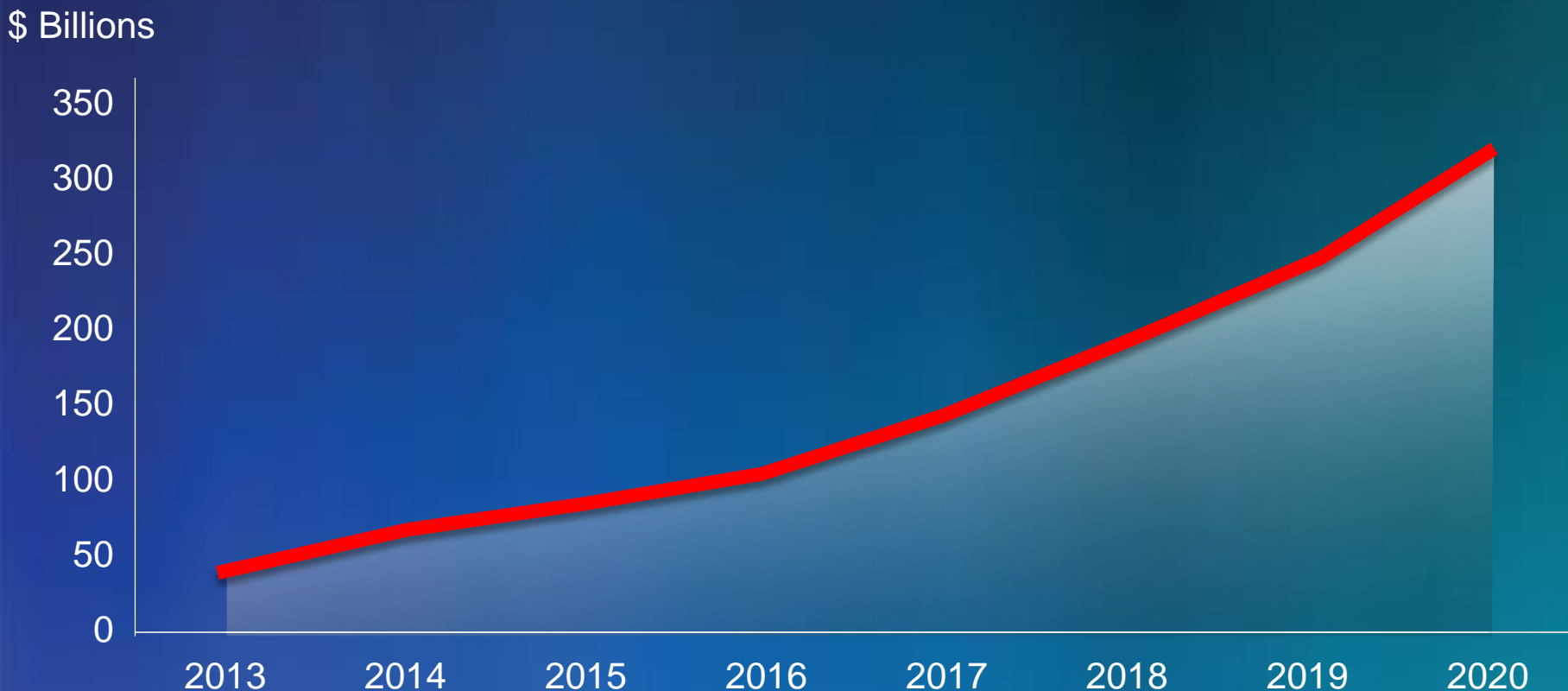
Enabling Embedded Vision



ADAS, Machine Vision, Surveillance, Drones, Medical, ProAV, Displays...

Rapid Growth of Vision Systems

Vision System Shipments



Source: Synopsys, consolidated from multiple sources

>200 Vision Customers Powered by Xilinx

POWERED BY XILINX

Camera



>30 Camera Brands

ADAS



23 Auto Makers, 85 Models

Industrial



>50 Equipment Manufacturers

Broadcast



8 Major Broadcasters

ProAV



>70 Equipment Makers

Drones



>5 Drone Companies

VR/AR



8 VR/AR Companies

Medical



>10 Medical Companies

Display

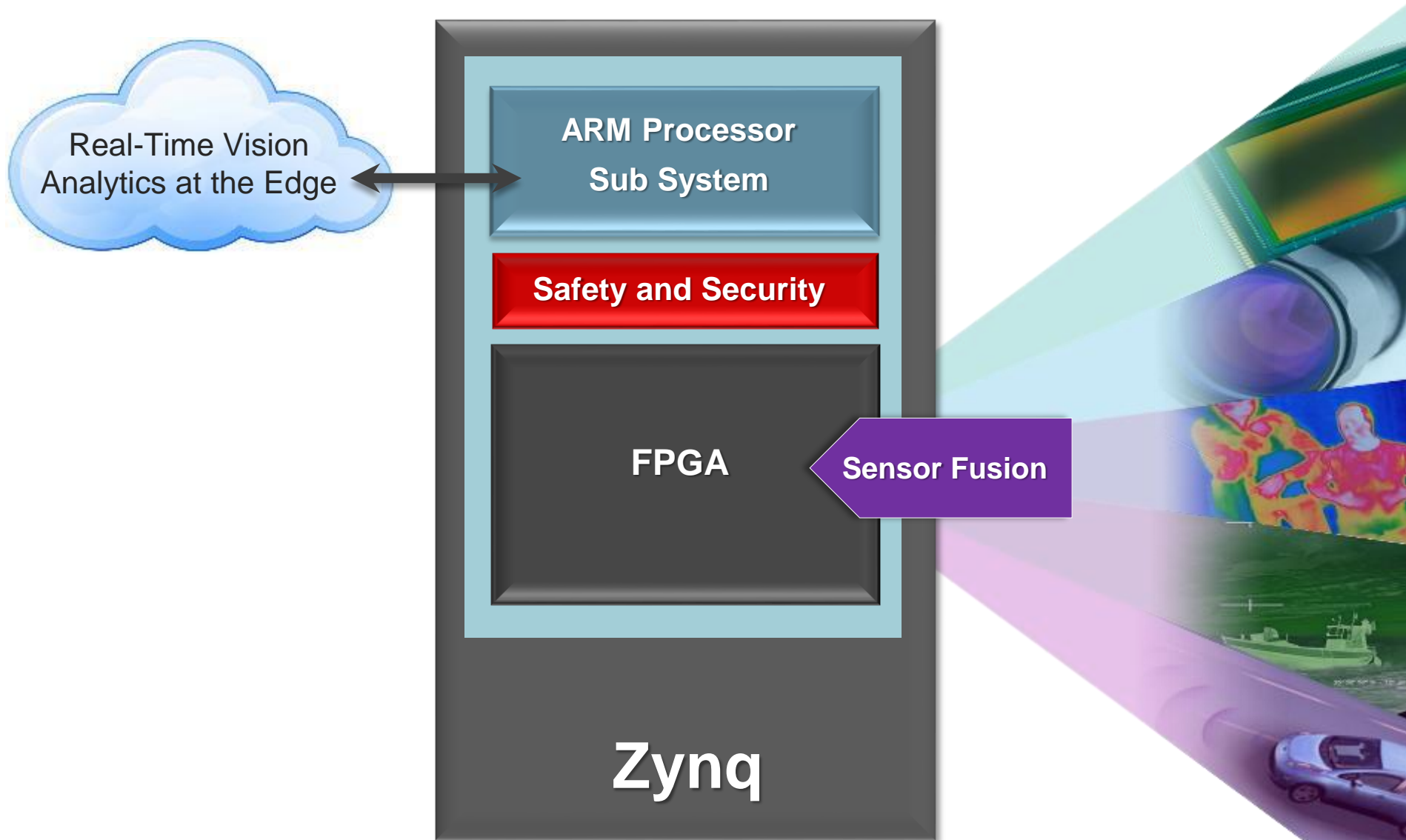


>30 Major Brands

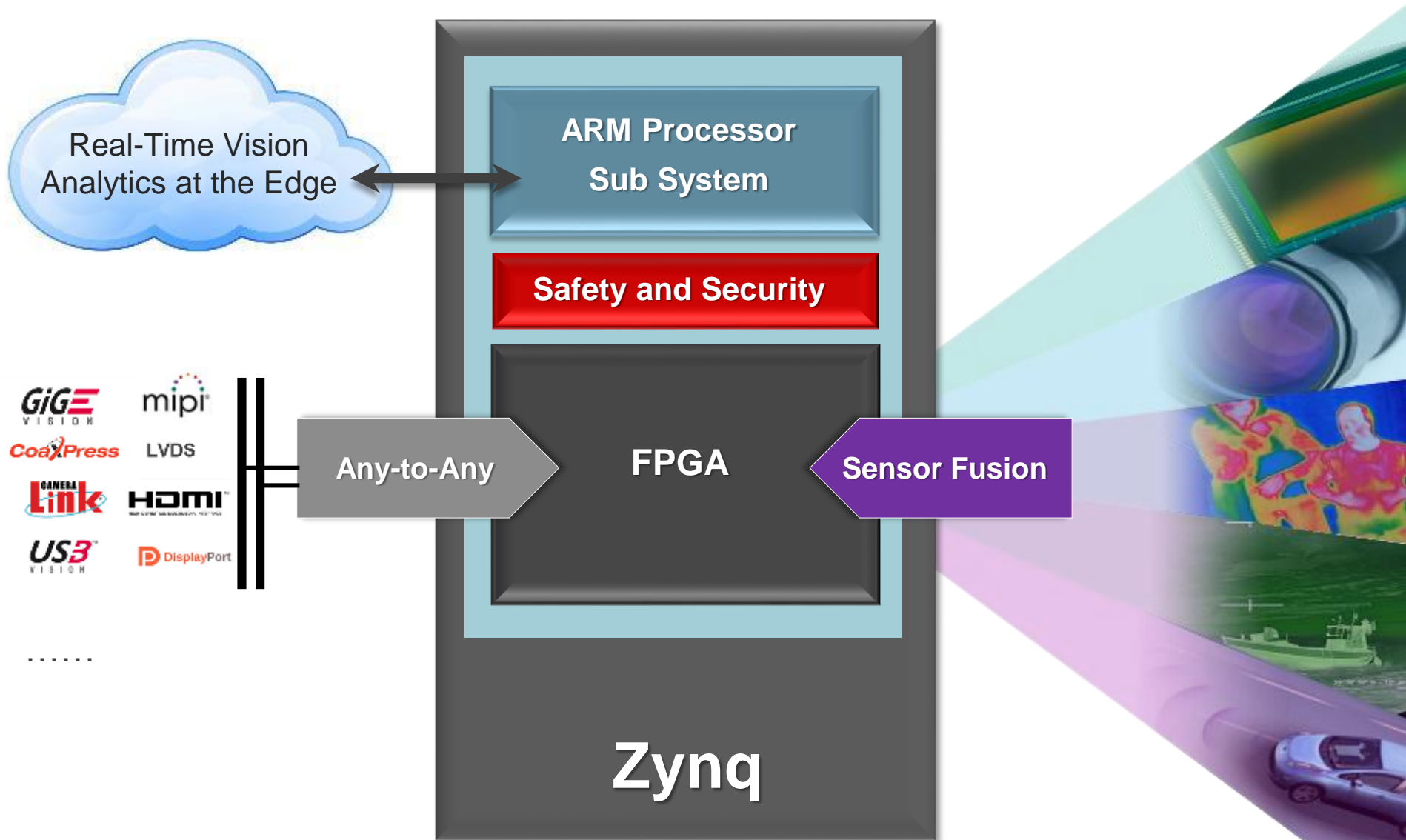
Best Platform for Embedded Vision



Best Platform for Embedded Vision



Best Platform for Embedded Vision





Vision and ADAS

5 Differentiating Advantages

"Vision with Precision"

- 1 Real-time Image Recognition and Analytics
- 2 All Programmable Platform Reuse
- 3 Scalable Sensor Fusion
- 4 Highest Performance/Watt
- 5 Only Single Chip Safety and Security



Surveillance

Enabling Smarter Surveillance Systems



- Multi-Sensor Fusion
- Real-Time Intelligence
- Compute at the Edge

Differentiating Advantages in Surveillance



Vision and ADAS 5 Differentiating Advantages "Vision with Precision"

- 1 Real-time Image Recognition and Analytics ✓
- 2 All Programmable Platform Reuse
- 3 Scalable Sensor Fusion ✓
- 4 Highest Performance/Watt ✓
- 5 Only Single Chip Safety and Security ✓

- Very high frame rate, recognition and analytics enabled through massive parallelism
- Scalable *sensor fusion* supports stereo to N vision pipelines + different sensor types
- Most computationally productive platform enabling highest performance per Watt
- ARM TrustZone & TRUST compliance for anti-tamper and information assurance

Markets and Applications



Source: <http://www.videosurveillance.com/apps/>

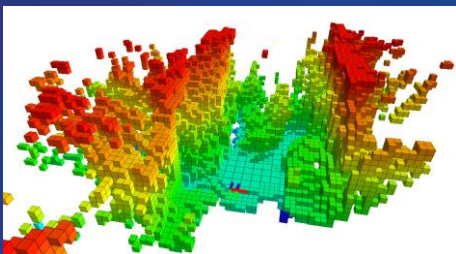
Remote Monitoring • ISR • Event Surveillance • Traffic Monitoring
Monitor Operations • Loss Prevention • Public Safety • Business Intelligence

TECHNOLOGY TRENDS IN SURVEILLANCE



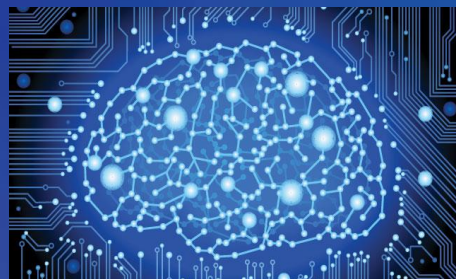
Multi Camera Vision

- Complete perspective with surround view
- Diverse sensor modalities provide enhanced vision
- Processing performance can now support dense fusion



Computer Vision (CV) Techniques

- OpenCV/OpenVX libraries increase productivity
- Optical Flow provides enhanced motion detection
- 3D/Stereo Vision enhances depth perception



Machine Learning Techniques → Building on CV

- Promises better recognition capability
- Object Detection & Classification thru Neural Networks
- Includes Convolutional, Deep and Recursive Neural Nets

The Machine Learning Dichotomy

Training



Photo: NVIDIA

- How the model is formed and developed
- Many approaches: DNN, CNN, RNN
- Low volume application requiring HPC
- DPfpu required to build models

Best Suited for GPGPUs

Inference

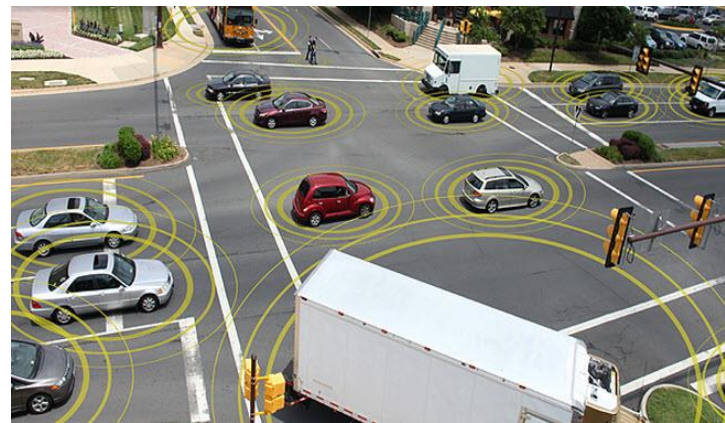


Photo: US DOT

- Requires efficient processing
- Does not need the precision of training
- High volume application targeting...
 - Automotive
 - VGR & Drones
 - Surveillance
 - Medical Imaging
- Fixed point math used to deploy models

WP490

Best Suited for FPGA / FPGA SoCs



Caffe

Application Development



SDSoC
Environment

DNN
CNN

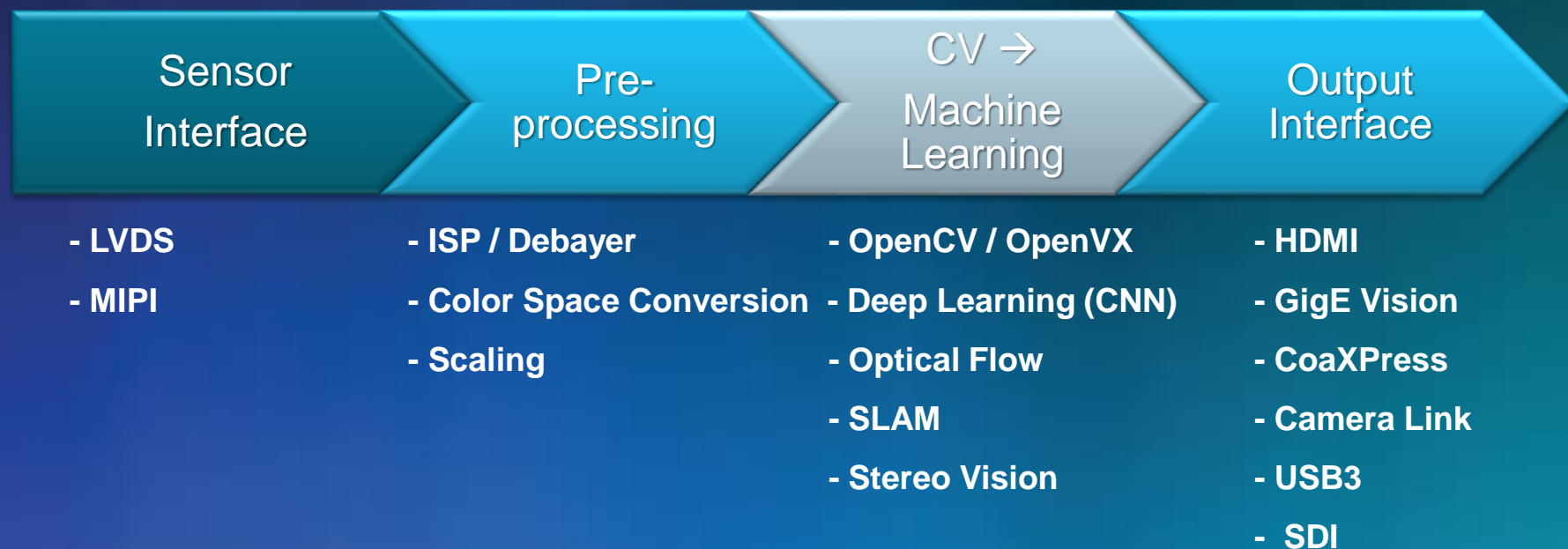
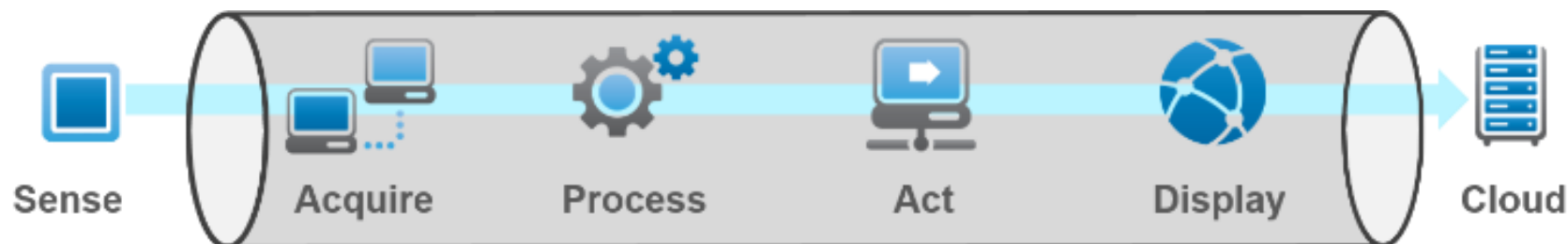
GoogLeNet
SSD
FCN ...

Algorithm Development

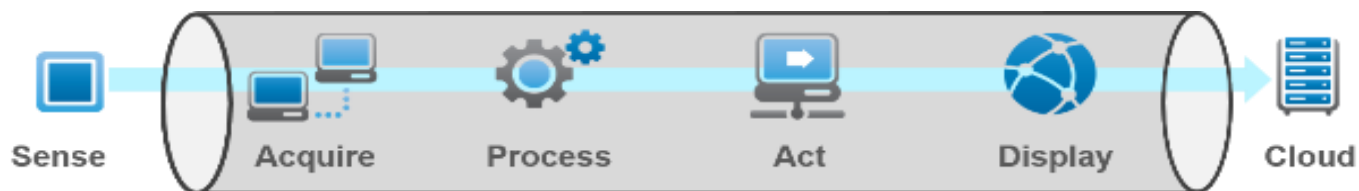


Platform Development








Typical Image Pipeline



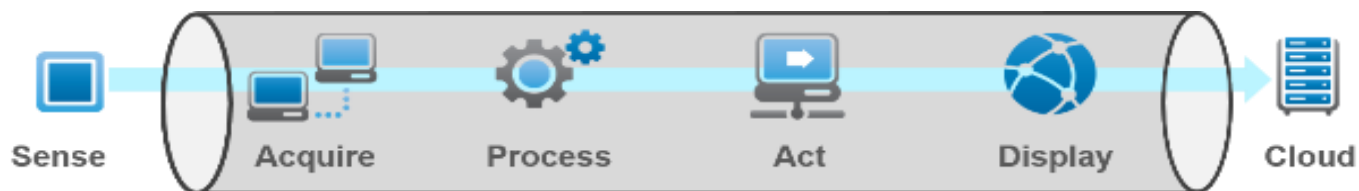
The Xilinx Embedded Vision Ecosystem



EMBEDDED SYSTEMS ECOSYSTEM FOR VIDEO / VISION

Sensor Processing	Video Processing	Analytics CV / xNN	Codecs	Connect
 	  	  	   	   

The Xilinx Embedded Vision Ecosystem



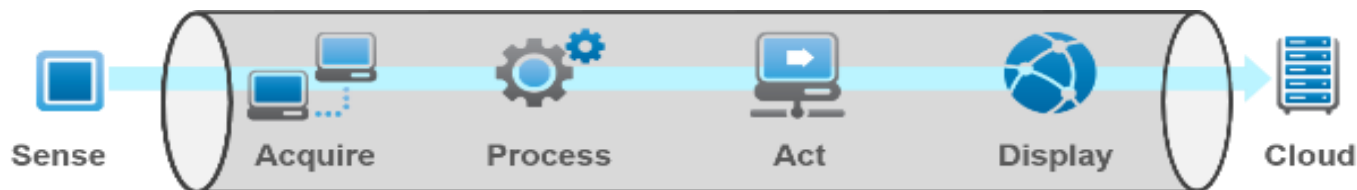
EMBEDDED SYSTEMS ECOSYSTEM FOR VIDEO / VISION

Sensor Processing	Video Processing	Analytics CV / xNN	Codecs	Connect
 	  	  	   	   

DESIGN ENABLEMENT



The Xilinx Embedded Vision Ecosystem



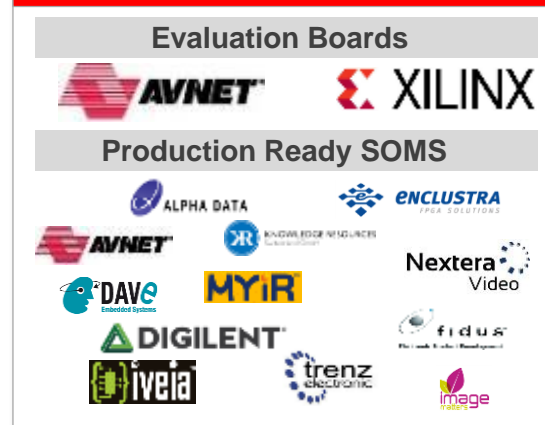
EMBEDDED SYSTEMS ECOSYSTEM FOR VIDEO / VISION

Sensor Processing	Video Processing	Analytics CV / xNN	Codecs	Connect
 	  	  	   	   

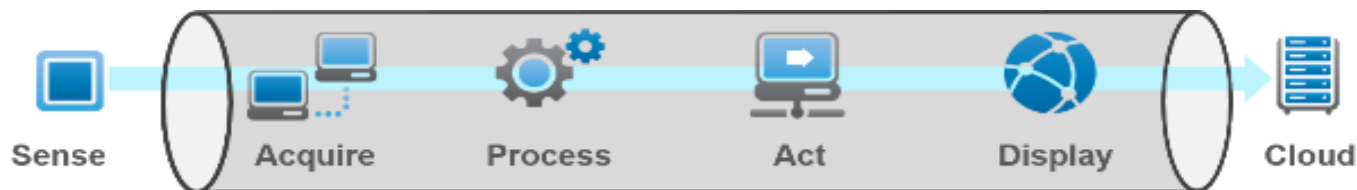
DESIGN ENABLEMENT



MODULES & BOARDS



The Xilinx Embedded Vision Ecosystem



EMBEDDED SYSTEMS ECOSYSTEM FOR VIDEO / VISION

Sensor Processing	Video Processing	Analytics CV / xNN	Codecs	Connect

DESIGN ENABLEMENT



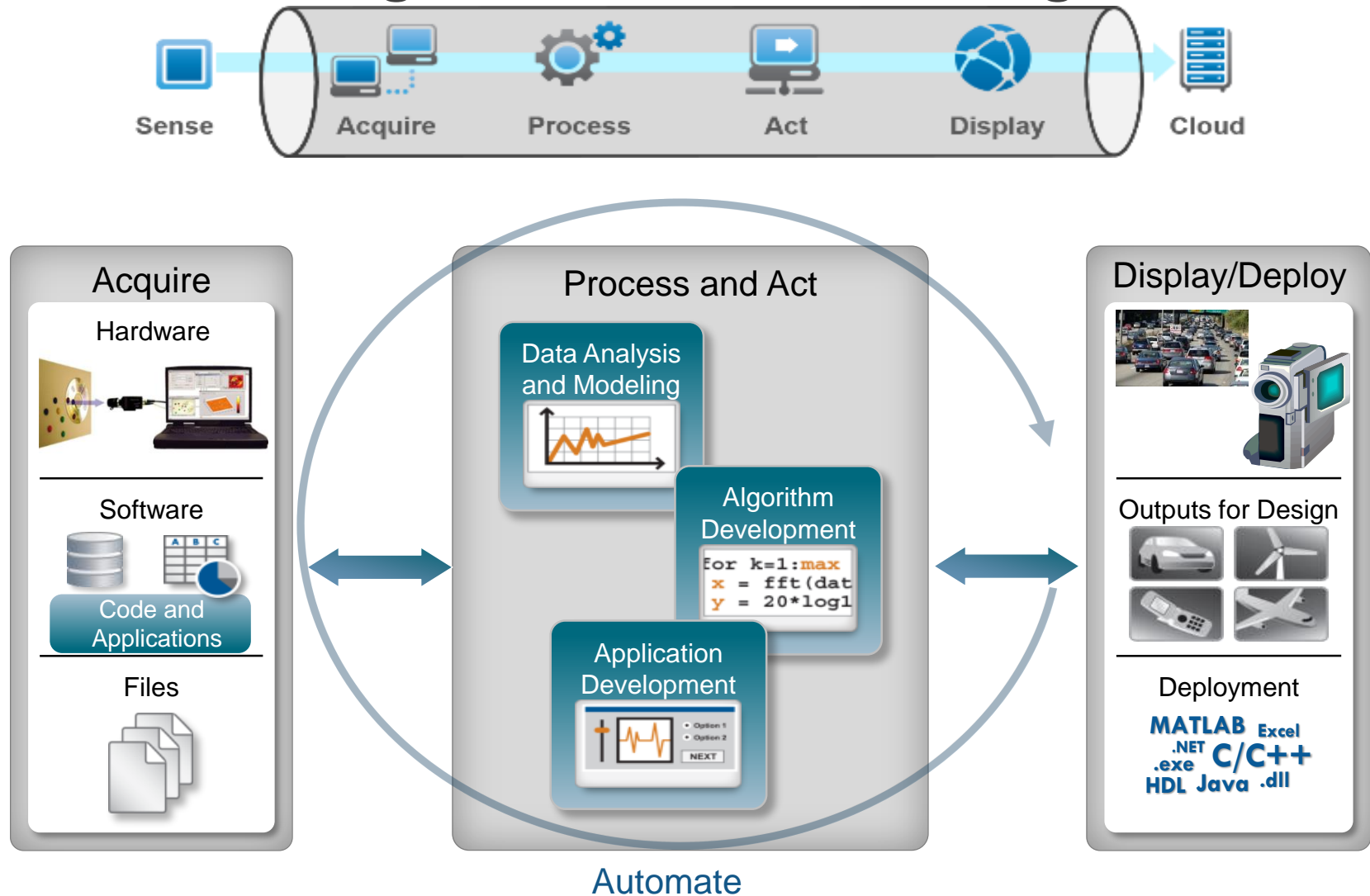
MODULES & BOARDS



DESIGN SERVICES



MATLAB® Image and Video Processing Workflow



MATLAB® Image and Video Processing Workflow

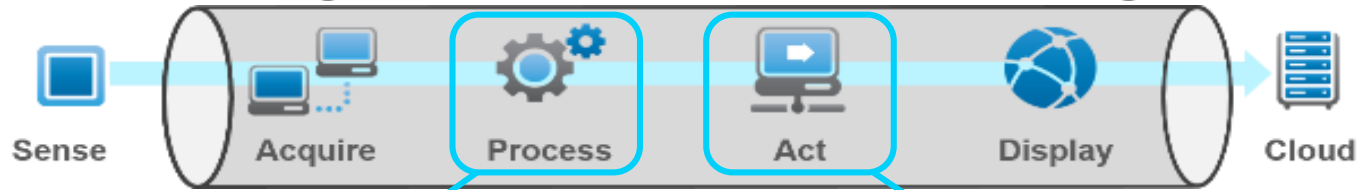
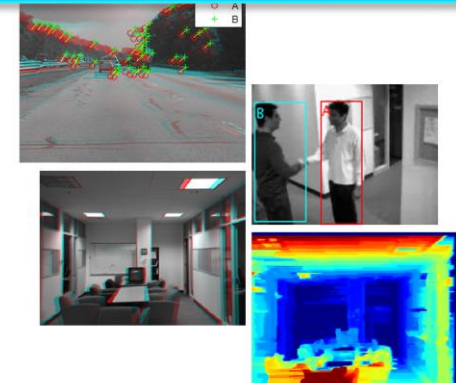
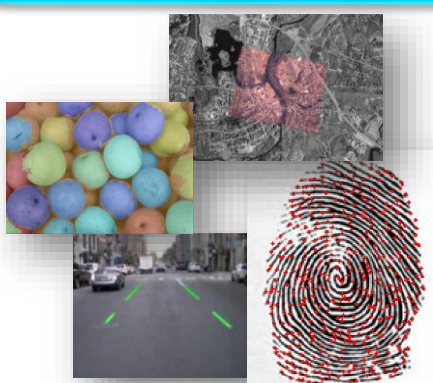


Image Processing Toolbox™

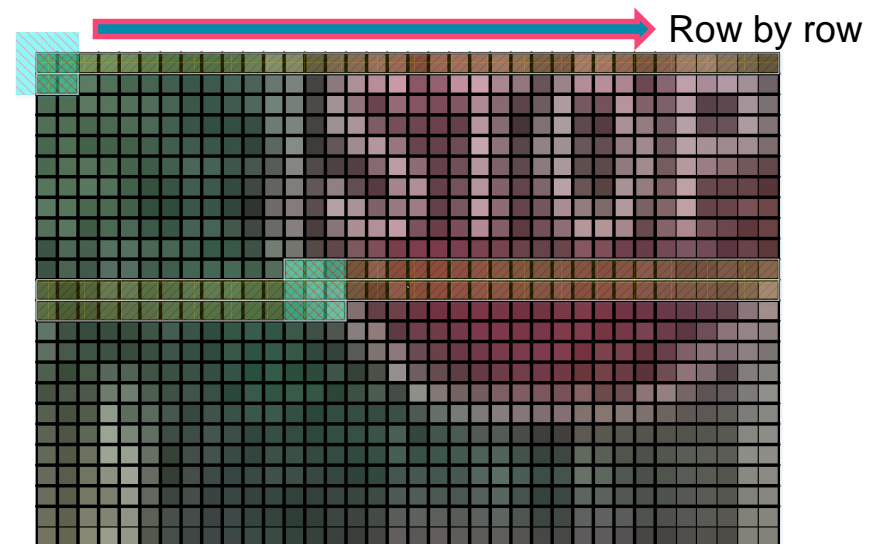
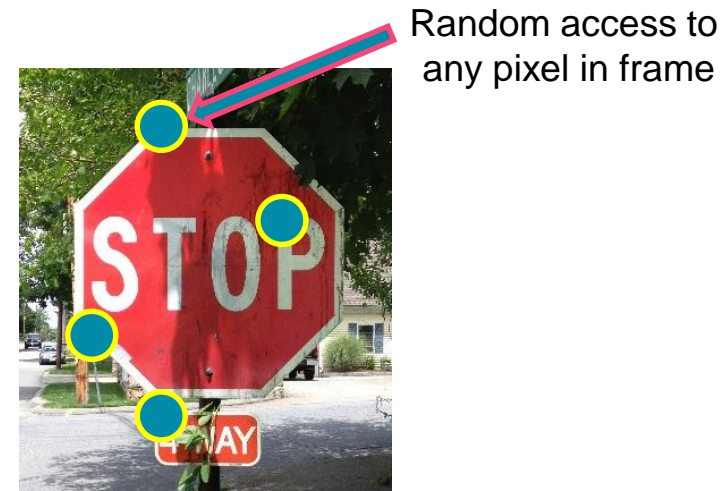
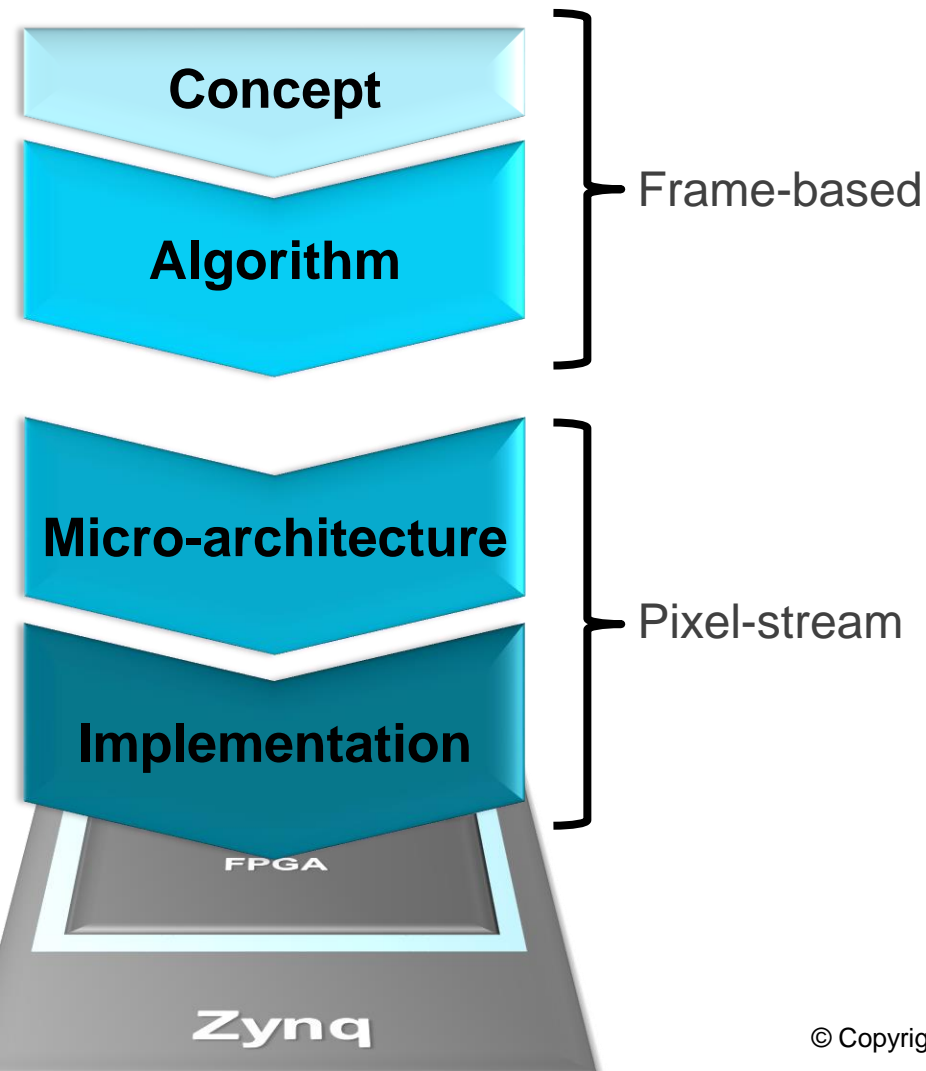
- Image display and exploration
- Image enhancement
- Image analysis
- Morphological operations
- Image registration
- Geometric transformation
- ROI-based processing

Computer Vision System Toolbox™

- Feature detection, extraction and matching
- Feature-based registration
- Object detection and tracking
- Stereo vision
- Video processing
- Motion estimation

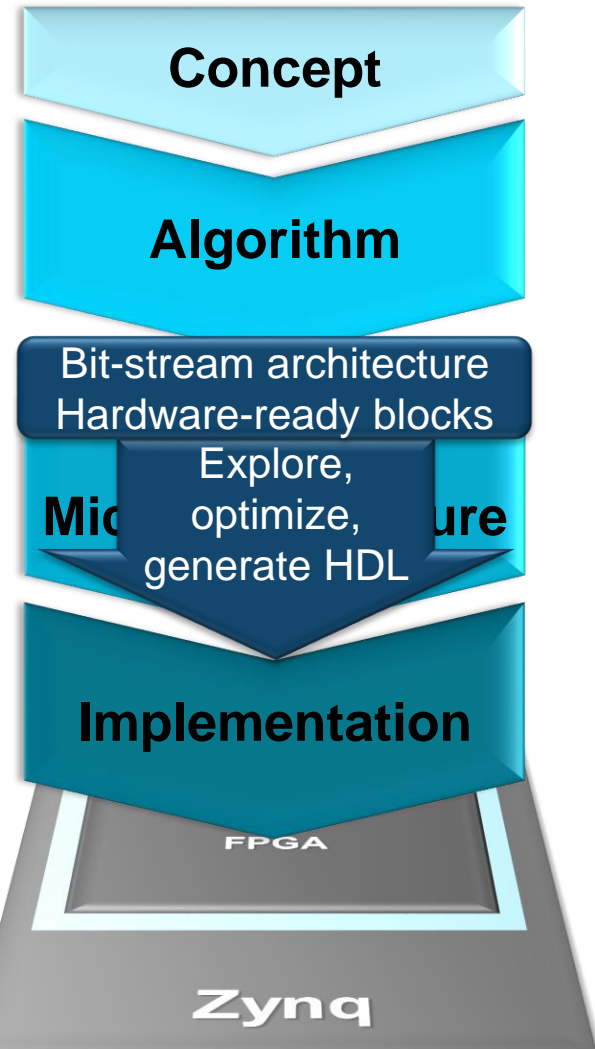


Frame vs pixel-stream processing



Vision HDL Toolbox™

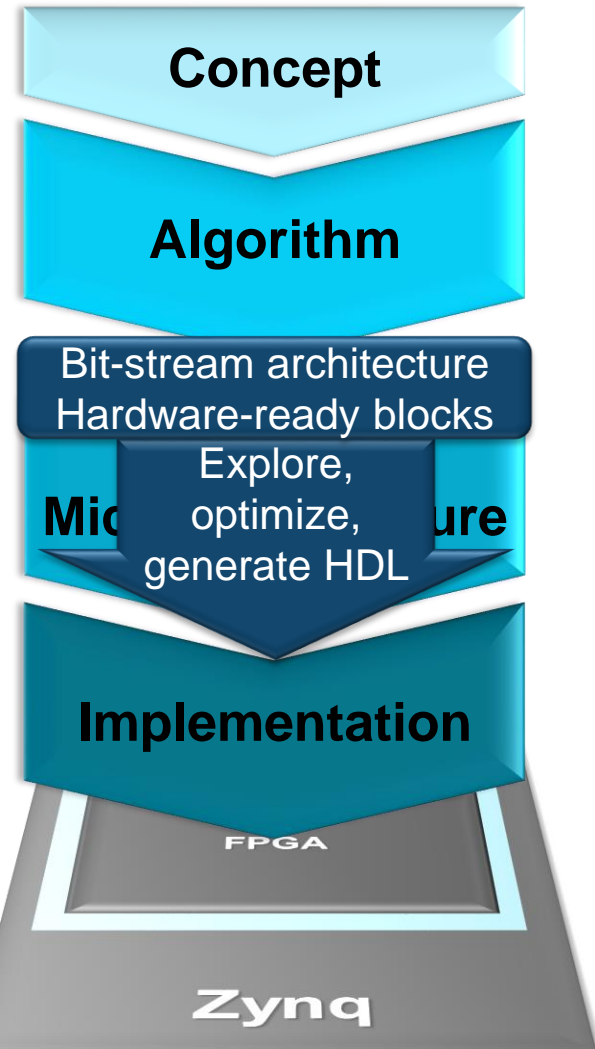
Hardware-ready streaming vision algorithms



- Convert frame-to-pixel, pixel-to-frame
 - Built-in frame sizes
 - Generates control signal bus
- Popular image processing FPGA building blocks
 - Conversion, filters, morphology, edge detection, statistics, etc
 - Built-in line buffer management
- Optimized for FPGA hardware
 - Generate HDL with HDL Coder

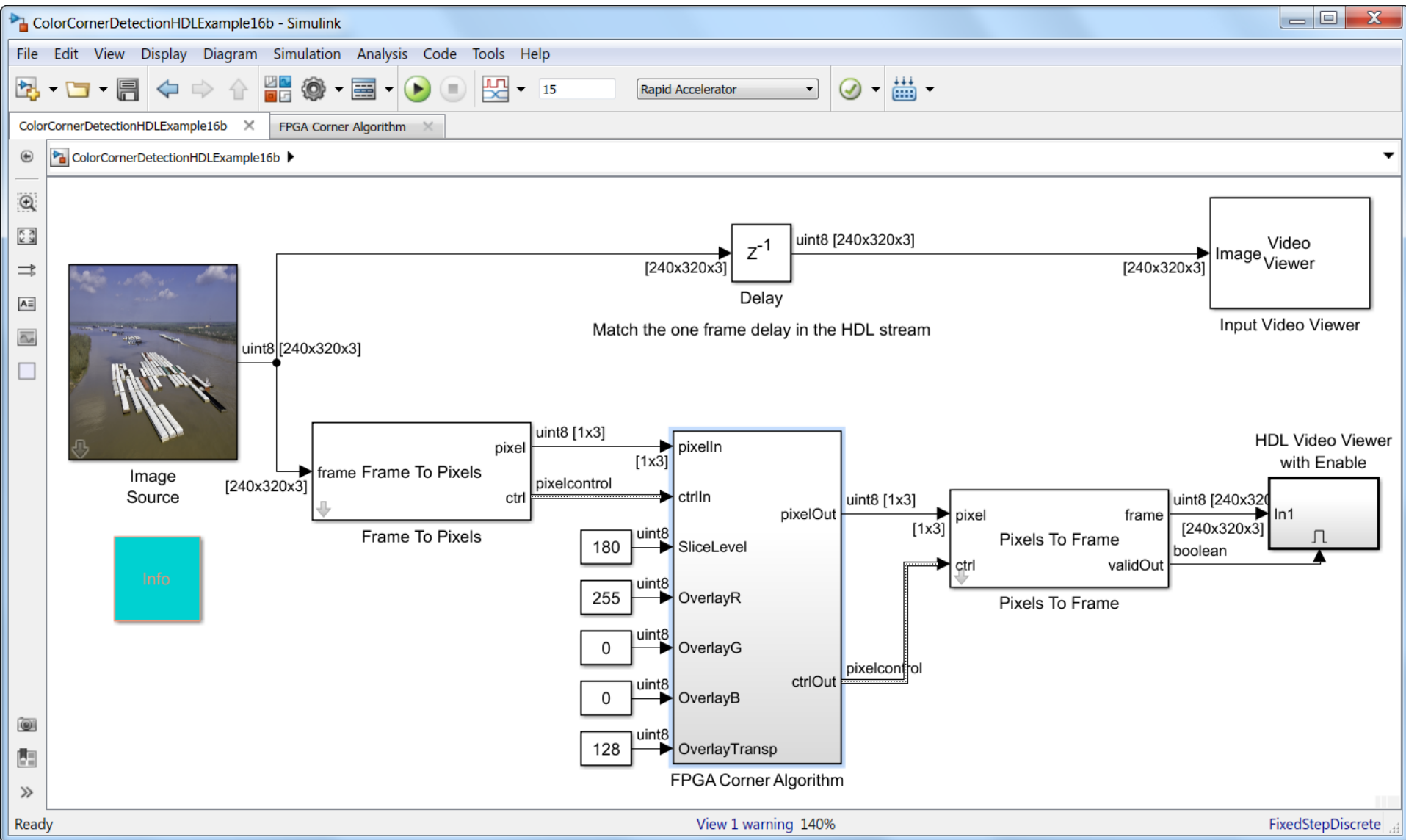
HDL Coder™

Automatically generate synthesizable HDL



- Explore micro-architectures
- High-level target-independent optimization
- Eliminate manual coding errors
- Connected to Simulink model for full traceability
- Readable, rule-compliant code
- Quickly adapt to changes

Example: Corner Detection for Barge Surveillance



Example: Frame-to-pixel gateway blocks

The image shows a Simulink workspace and a 'Block Parameters: Frame To Pixels' dialog box.

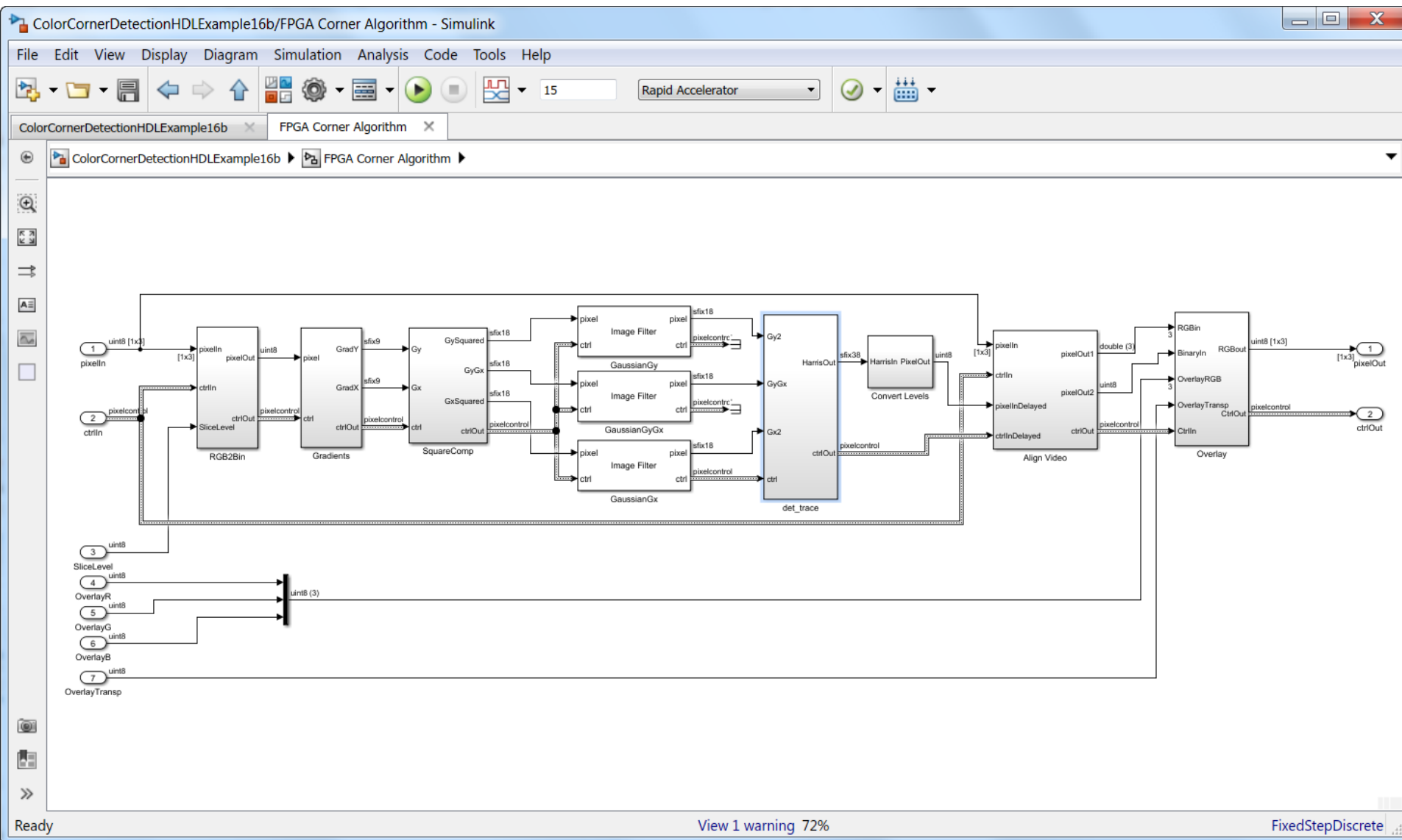
Simulink Workspace:

- Model: ColorCornerDetectionHDLExample16b - Simulink
- Block: Image Source (output: uint8 [240x320x3])
- Block: frame Frame To Pixels (input: uint8 [240x320x3], output: pixel [240x320x3])
- Block: Frame To Pixels (input: pixel [240x320x3], output: pixel [240x320x3])
- Block: Video Viewer (input: pixel [240x320x3])
- Block: HDL Video Viewer with Enable (input: pixel [240x320x3])

Block Parameters: Frame To Pixels (mask) (link):

- Converts a full frame image to pixel stream.
- Parameters:
 - Number of components: 3
 - Video format: 240p (selected)
- Diagram illustrating the video format parameters:
 - Active pixels per line
 - Active video lines
 - Ending active line
 - Back Porch
 - Front Porch
 - Total video lines

Example: FPGA design



Example: Vision HDL Toolbox blocks

The screenshot shows the Simulink environment for the 'ColorCornerDetectionHDLExample16b/FPGA Corner Algorithm'. The block diagram includes the following components and connections:

- Inputs:** A 'pixelIn' input (uint8 [1x3]) is connected to a 'pixelIn' block.
- Processing:** The signal passes through a 'GradY' block (sfix9), a 'Gy' block (sfix18), and a 'GySquared' block (sfix18).
- Filtering:** The signal enters the 'Image Filter' block (sfix18) within the 'GaussianGy' block. The 'Image Filter' block has inputs for 'pixel' and 'ctrl', and outputs for 'pixel' and 'ctrl'.
- Conversion:** The output of the 'Image Filter' block is connected to a 'Gy2' block (sfix18).
- Alignment:** The signal then passes through a 'Convert Levels' block (sfix38) and an 'Align Video' block (sfix18).
- Output:** The final output is connected to an 'Overlay' block (uint8 [1x3]), which produces the 'pixelOut' (1) and 'ctrlOut' (2) signals.

The 'Block Parameters: GaussianGy' dialog is open, showing the 'Data Types' tab. The parameters are:

- Image Filter:** Performs two-dimensional FIR filtering of the input image using the specified filter coefficients.
- Filter coefficients:** `fspecial('gaussian',[5,5],1.5)`
- Padding method:** Constant
- Padding value:** 0
- Line buffer size:** 2048

The status bar at the bottom indicates 'Ready', 'View 1 warning 72%', and 'FixedStepDiscrete'.

Example: MATLAB function block

The screenshot displays the Xilinx Vivado IDE interface, specifically the Simulink environment. The main window shows a block diagram of the 'ColorCornerDetectionHDLExample16b/FPGA Corner Algorithm'. The diagram includes several blocks: 'RGB2Bin', 'Gradients', 'SquareComp', and 'Overlay_ML'. The 'Overlay_ML' block is highlighted, showing its internal structure and inputs/outputs. The inputs to 'Overlay_ML' are: 'RGBin' (double (3)), 'BinaryIn' (uint8), 'OverlayRGB' (uint8 (3)), 'OverlayTransp' (uint8), and 'CtrlIn' (pixelcontrol). The output is 'RGBout' (uint8). The 'Overlay_ML' block is connected to the 'SquareComp' block. The 'SquareComp' block has inputs 'Gy' and 'Gx' (both sfix18) and outputs 'GySquared' and 'GxSquared' (both sfix18). The 'Gradients' block has inputs 'GradY' and 'GradX' (both sfix9) and outputs 'Gy' and 'Gx' (both sfix18). The 'RGB2Bin' block has inputs 'pixelIn' (uint8 [1x3]) and 'ctrlIn' (uint8) and outputs 'pixelOut' (uint8) and 'ctrlOut' (uint8). The 'Overlay_ML' block is also connected to a 'CtrlIn' block (uint8 (3)). The 'Overlay_ML' block is also connected to a 'CtrlIn' block (uint8 (3)).

The 'Overlay_ML' block is a MATLAB function block. The code inside the block is as follows:

```
function RGBout = fcn(RGBin, BinaryIn, OverlayRGB, OverlayTransp)
%#codegen
N=uint8(255);
N_fi=fi(255,0,16,8);

% set up persistent variables to create registers on the output mults
persistent Rout Bout Gout;

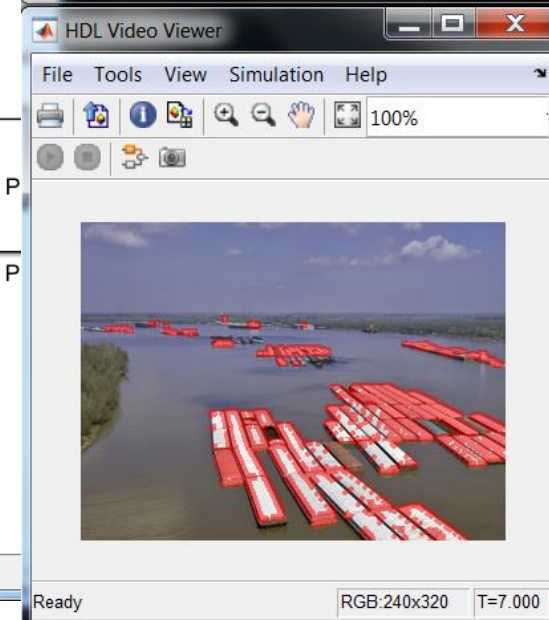
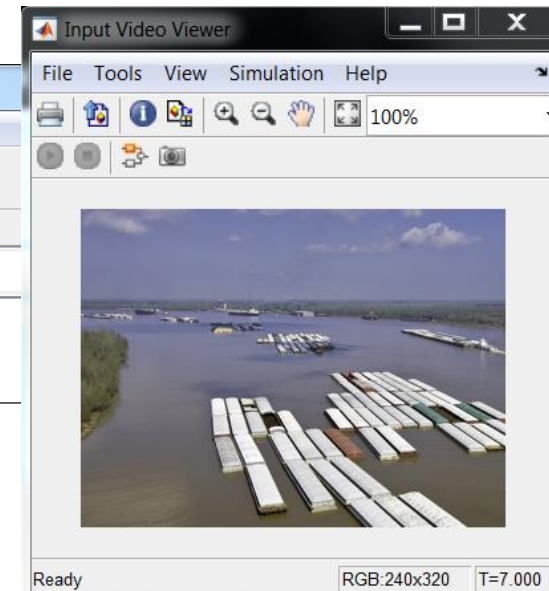
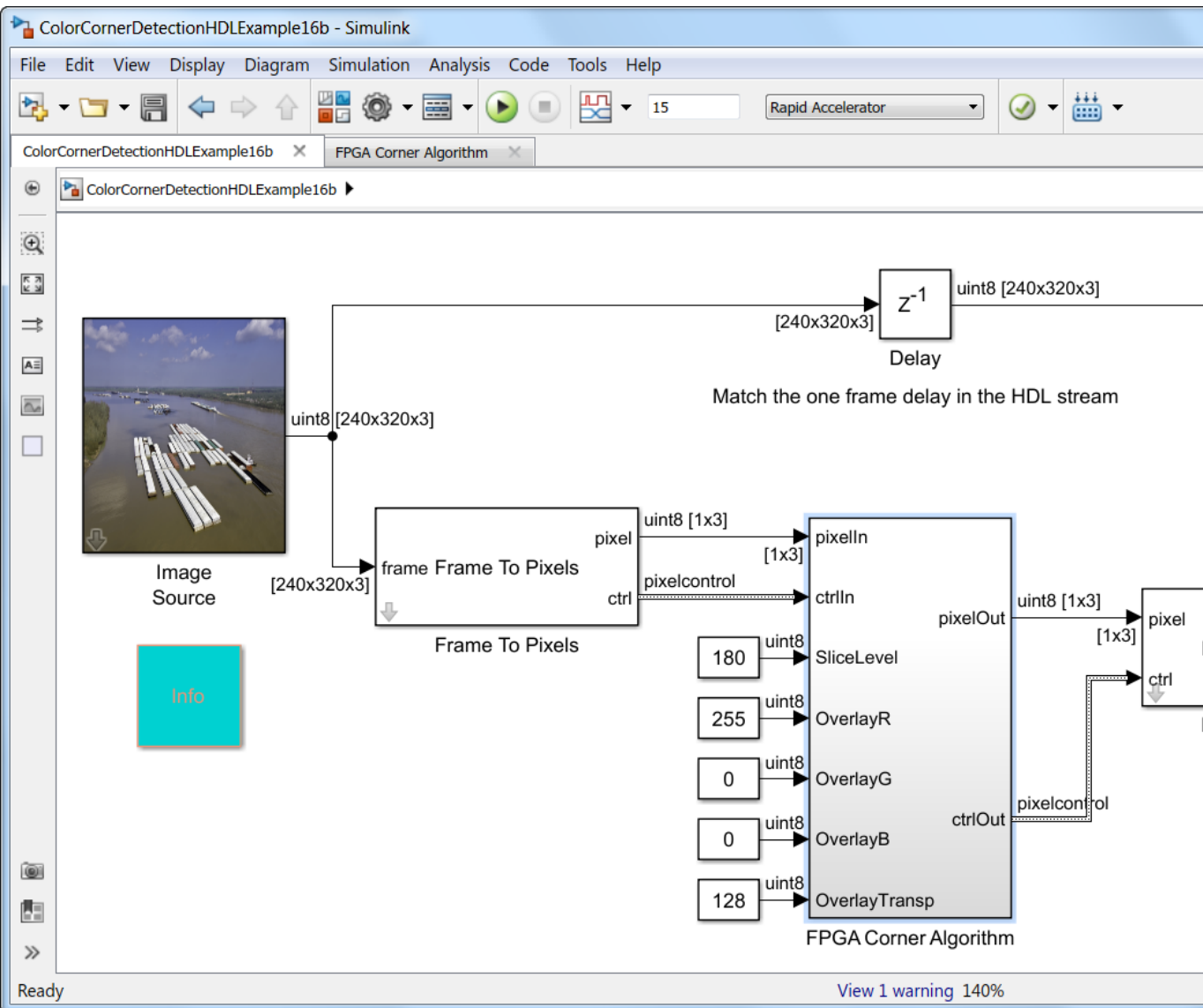
if BinaryIn == 0
    transp=fi(OverlayTransp,0,8,0);
else
    transp=fi(0,0,8,0);
end

norm=fi(1/N_fi,0,8,8);
transp_normalized=fi(transp*norm,0,8,8);

% convert inputs from int to fixed-point for normalization
RGBin1_fi=fi(RGBin(1),0,8,0);
RGBin2_fi=fi(RGBin(2),0,8,0);
RGBin3_fi=fi(RGBin(3),0,8,0);
OverlayRGB1_fi=fi(OverlayRGB(1),0,8,0);
OverlayRGB2_fi=fi(OverlayRGB(2),0,8,0);
OverlayRGB3_fi=fi(OverlayRGB(3),0,8,0);

% first stage of multipliers, add one pipeline stage
% coder.hdl.pipeline - create registers without affecting behavior
RGB1_normalized=coder.hdl.pipeline(fi(RGBin1_fi*norm,0,8,8),1);
```


Example: Simulate



Example: Generate HDL

The screenshot displays the Simulink environment for the 'ColorCornerDetectionHDExample16b' model. The 'FPGA Corner Algorithm' subsystem is selected, and the 'HDL Code' menu item is highlighted. The 'HDL Workflow Advisor' option is also highlighted in a blue box.

The diagram shows an 'Image Source' block connected to a 'Frame To Pixels' block. The output of 'Frame To Pixels' is a 'uint8 [240x320x3]' signal, which is then connected to an 'HDL Video Viewer with Enable' block. The 'HDL Video Viewer with Enable' block has an 'In1' input and a 'validOut' output. The 'validOut' output is connected to an 'Image Video Viewer' block.

The 'HDL Code' menu is open, showing options such as 'Check Subsystem Compatibility', 'Generate HDL for Subsystem', 'HDL Coder Properties ...', 'HDL Block Properties', 'HDL Workflow Advisor', and 'Navigate to Code'. The 'HDL Workflow Advisor' option is highlighted in a blue box.

Example: HDL Workflow Advisor

The screenshot shows the HDL Workflow Advisor application window. The title bar reads "HDL Workflow Advisor - ColorCornerDetectionHDLExample16b/FPGA Corner Algorithm". The menu bar includes File, Edit, Run, and Help. Below the menu bar is a "Find:" field with a search icon and navigation arrows. The left sidebar displays a tree view of the workflow tasks:

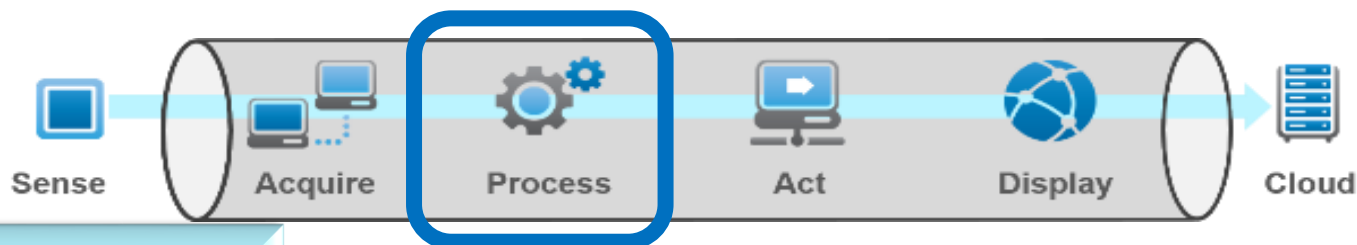
- HDL Workflow Advisor
 - 1. Set Target
 - ^1.1. Set Target Device and Synthesis Tool**
 - ^1.2. Set Target Interface
 - 2. Prepare Model For HDL Code Generation
 - 2.1. Check Global Settings
 - ^2.2. Check Algebraic Loops
 - ^2.3. Check Block Compatibility
 - ^2.4. Check Sample Times
 - 3. HDL Code Generation
 - 3.1. Set Code Generation Options
 - 3.1.1. Set Basic Options
 - 3.1.2. Set Advanced Options
 - 3.1.3. Set Optimization Options
 - ^3.2. Generate RTL Code and IP Core

The main panel displays the details for the selected task, "1.1. Set Target Device and Synthesis Tool". It includes a sub-header "Analysis (^Triggers Update Diagram)" and a description "Set Target Device and Synthesis Tool for HDL code generation". Below this is the "Input Parameters" section with the following fields:

- Target workflow: IP Core Generation
- Target platform: Generic Xilinx Platform (with a "Launch Board Manager" button)
- Synthesis tool: Xilinx Vivado (with a "Tool version: 2014.4" field and a "Refresh" button)
- Family: Zynq
- Device: xc7z045
- Package: fbg676
- Speed: -1
- Project folder: hdl_prj (with a "Browse..." button)

Below the input parameters is a "Run This Task" button. The "Result:" section shows a status icon and the text "Not Run". A large text area below this contains the instruction "Click Run This Task." At the bottom right of the window are "Help" and "Apply" buttons.

MATLAB and Simulink Image and Video Processing



Concept

Algorithm

Bit-stream architecture
Hardware-ready blocks

Explore,
optimize,
generate HDL

Implementation

FPGA

Zynq

Image
Processing
Toolbox

Computer
Vision
System
Toolbox

MATLAB
Simulink

Vision HDL Toolbox

HDL Coder

Xilinx Vivado

Connected workflow from algorithm to FPGA fabric

Conclusion

- The Embedded Vision market is growing fast
- Xilinx is the best platform for Embedded Vision for...
 - Any-to-Any Connectivity
 - Sensor Fusion
 - Real-Time Analytics at the Edge
- Multi Camera Vision, OpenCV and Machine Learning key trends
- MathWorks & Xilinx Make Embedded Vision Development Easy
 - HDL Coder for Zynq is a fast track to realize real-time vision systems
 - SIMULINK + Vivado System Edition (System Generator) quicken system-level design
 - Vision HDL Toolbox accelerates your development with commonly needed algorithms

Email: embedded-vision@xilinx.com for this presentation