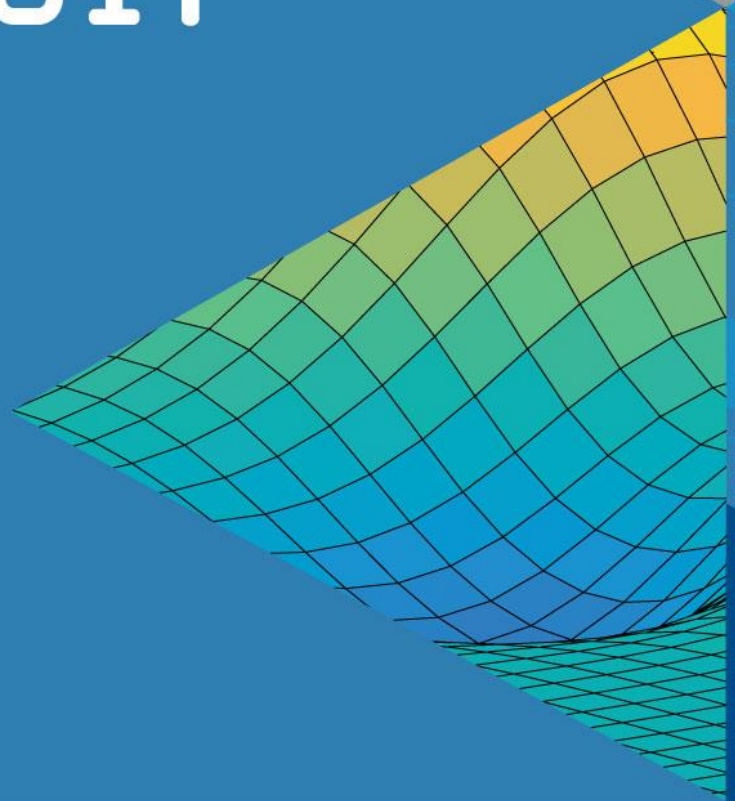
MATLAB EXPO 2017 KOREA

4월 27일, 서울

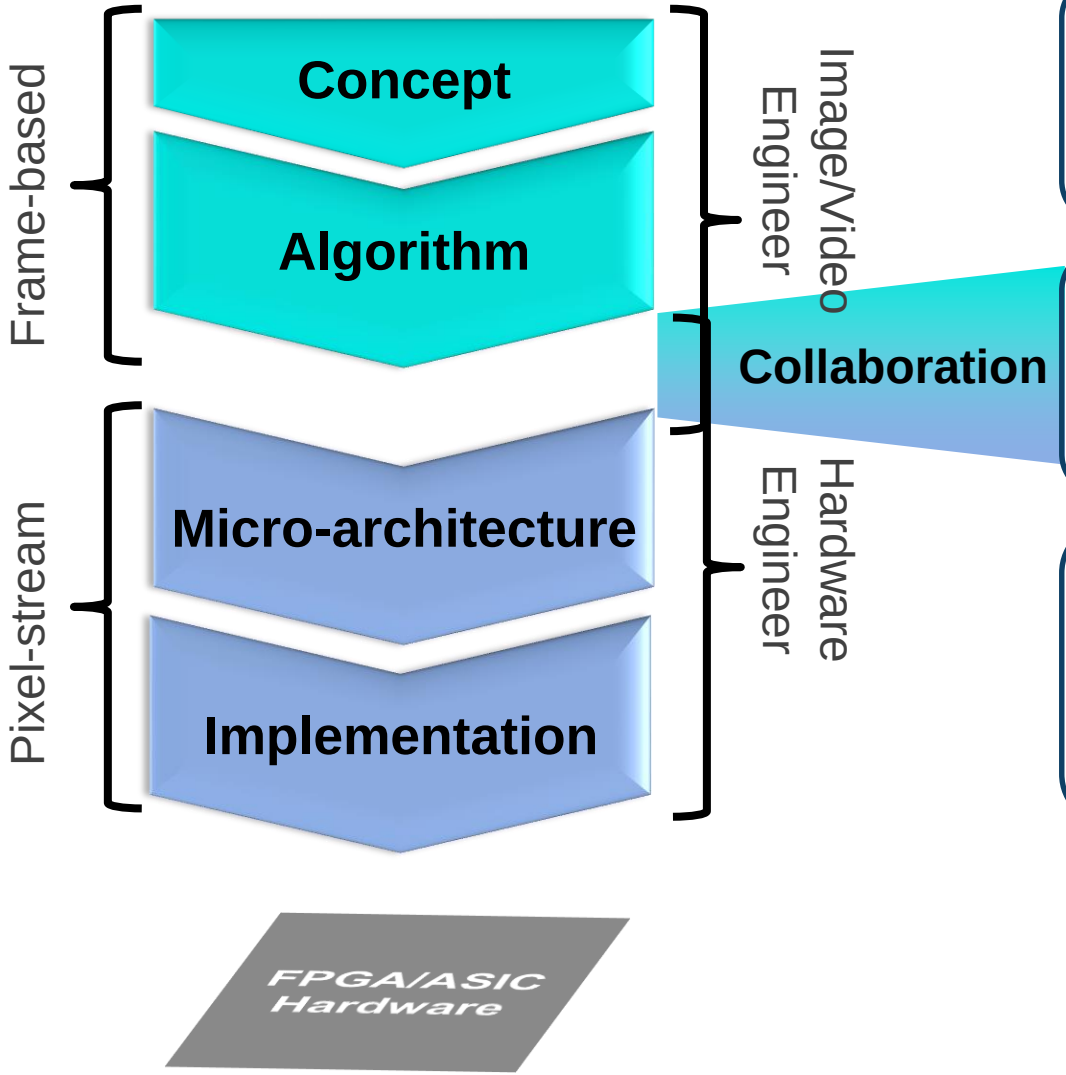
등록 하기 matlabexpo.co.kr



Designing and Targeting Video Processing Subsystems for Hardware

정승혁 과장
Senior Application Engineer
MathWorks Korea

Process : From Algorithm to Hardware

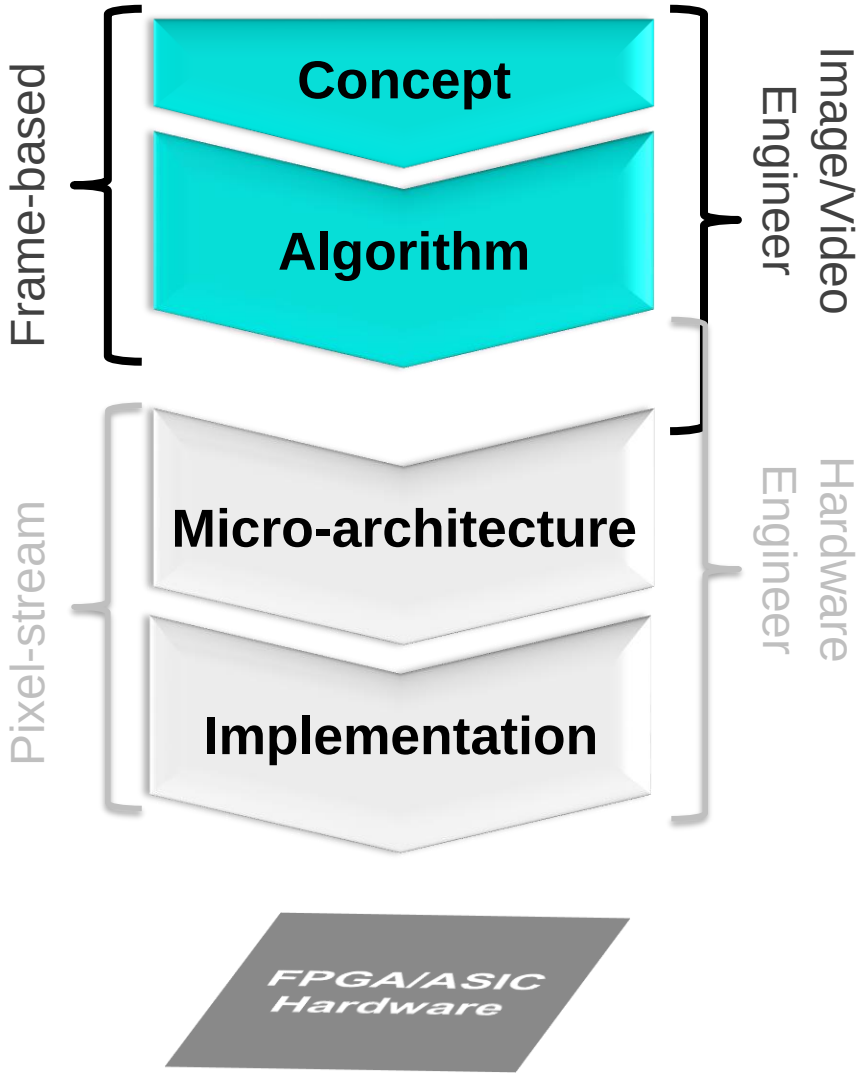


1. Concept & Algorithm
- Develop system-level algorithms
- Simulate, analyze, modify
- Partition hardware vs software implementation

2. Algorithm to Micro-architecture
- Convert to pixel-streaming algorithms
- Add hardware micro-architecture
- Convert data types to fixed-point

3. Micro-architecture to implementation
- Speed and area optimization
- HDL code generation
- Verification
- FPGA/ASIC implementation

Developing the System-Level Algorithm



<Concept & Algorithm>

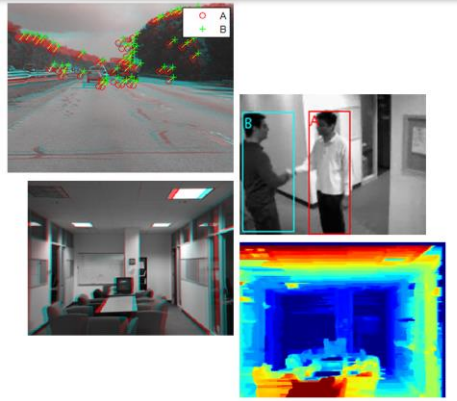
- Develop system-level algorithms
- Simulate, analyze, modify
- Partition hardware vs software implementation

Image Processing Toolbox™

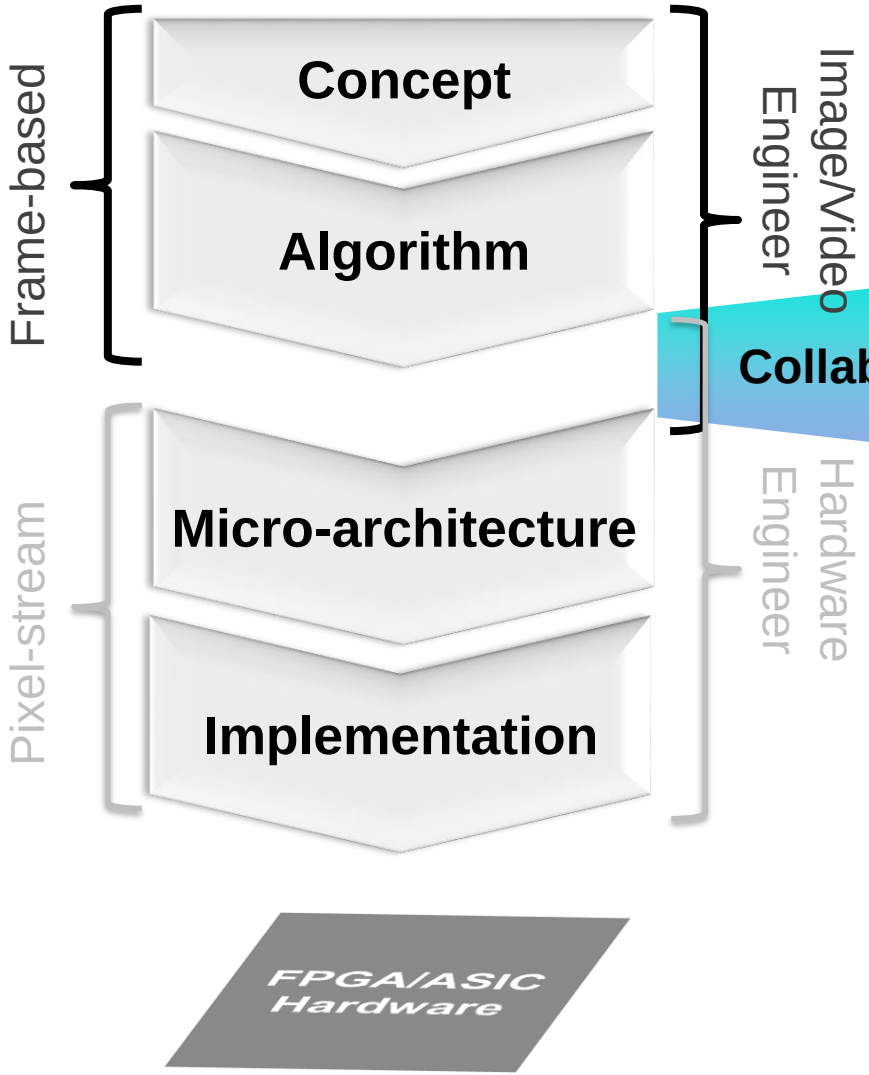
- Image display and exploration
- Image enhancement
- Morphological operations
- Image registration
- Geometric transformation
- ROI-based processing

Computer Vision System Toolbox™

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



From Frames to Pixels to Hardware



<Algorithm to Micro-architecture>

- Convert to pixel-streaming algorithms
- Add hardware micro-architecture
- Convert data types to fixed-point

Vision HDL Toolbox™

- Simulate hardware micro-architecture of algorithms
- Streaming pixel-based functions and blocks
- Convert between frames and pixels
- Standard and custom frame sizes
- Built-in line buffer management

Vision HDL Toolbox System Objects

Show: System Objects

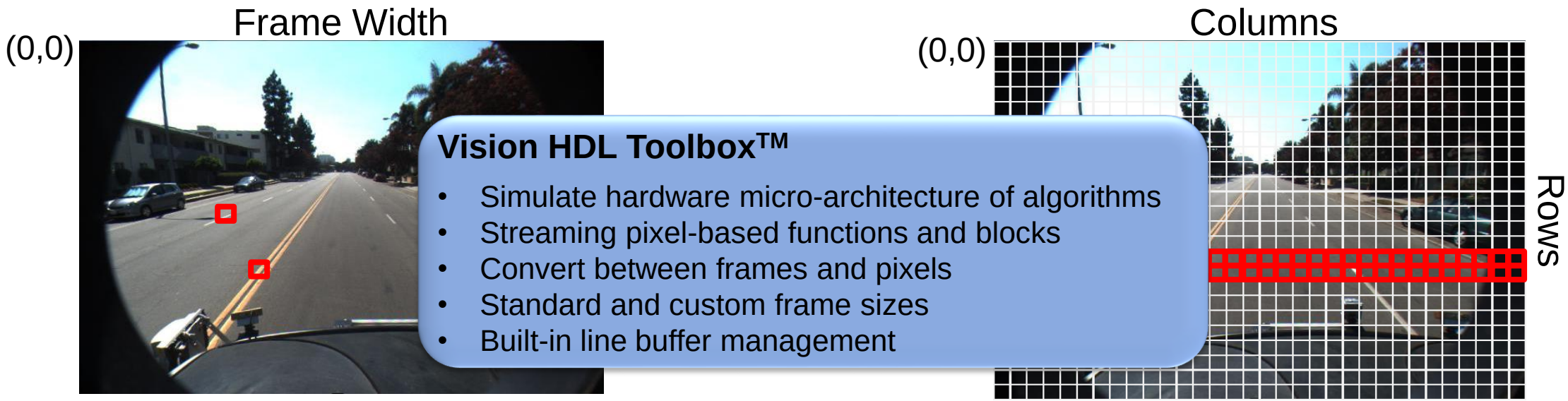
Video Formats and Interfaces

visionhdl.FrameToPixels	Convert frame-based video to pixel stream
visionhdl.PixelsToFrame	Convert pixel stream to frame-based video
visionhdl.MeasureTiming	Measure timing of pixel control structure input
visionhdl.PixelStreamAligner	Align two streams of pixel data
visionhdl.ROISelector	Select region of interest (ROI) from pixel stream

HDL-Optimized Algorithm Design

visionhdl.ChromaResampler	Downsample or upsample chrominance component
visionhdl.ColorSpaceConverter	Convert color information between color spaces
visionhdl.DemosaicInterpolator	Construct full RGB pixel data from Bayer pattern pixels
visionhdl.GammaCorrector	Apply or remove gamma correction
visionhdl.LookupTable	Map input pixel to output pixel using custom rule
visionhdl.EdgeDetector	Find edges of objects

Image Processing Algorithms: Frame-Based vs Streaming-Pixel



Vision HDL Toolbox™

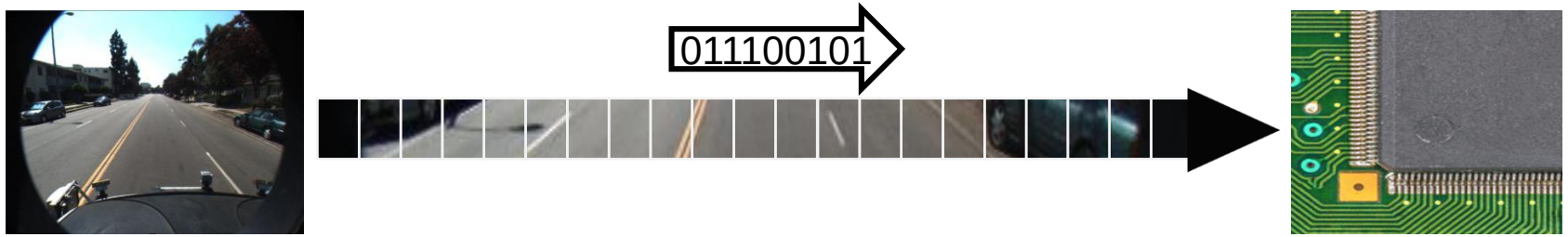
- Simulate hardware micro-architecture of algorithms
- Streaming pixel-based functions and blocks
- Convert between frames and pixels
- Standard and custom frame sizes
- Built-in line buffer management

Frame-Based

- Whole frame at a time
- Random access to any pixel via MATLAB/Simulink [X,Y] coordinate

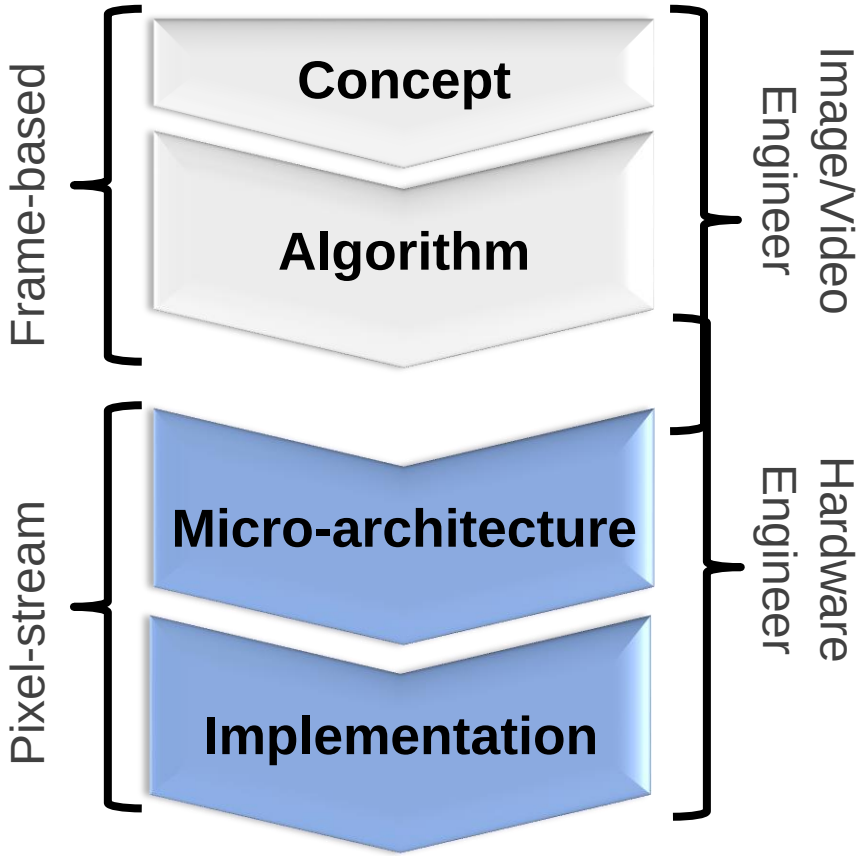
Streaming-Pixel

- Across rows, row-by-row
- Region of interest (ROI) stored in a multi-line buffer



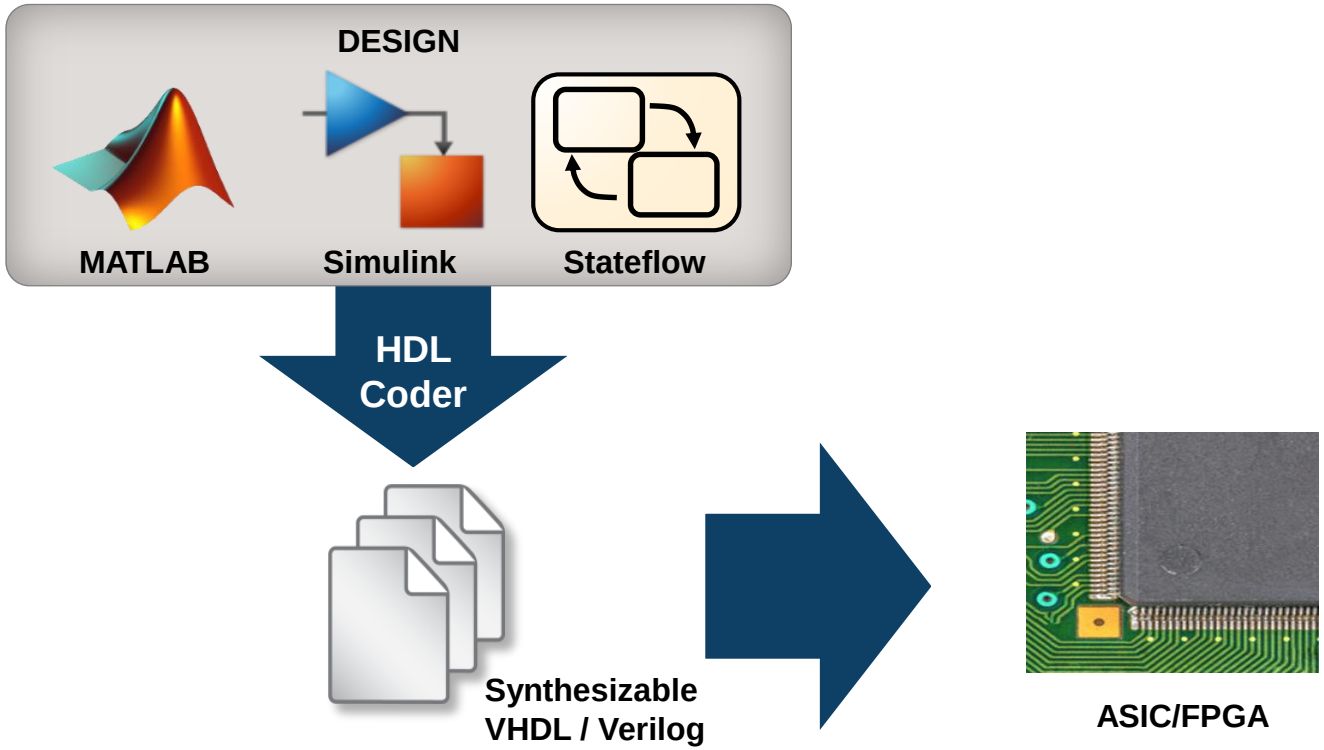
From Model to HDL Code

Automatically generate synthesizable HDL from system-level design



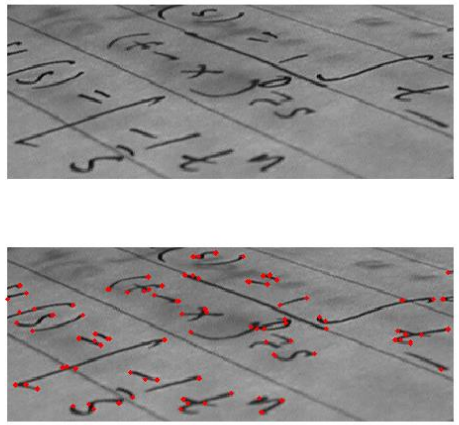
Micro-architecture to implementation

- Speed and area optimization
- HDL code generation
- Verification
- FPGA/ASIC implementation



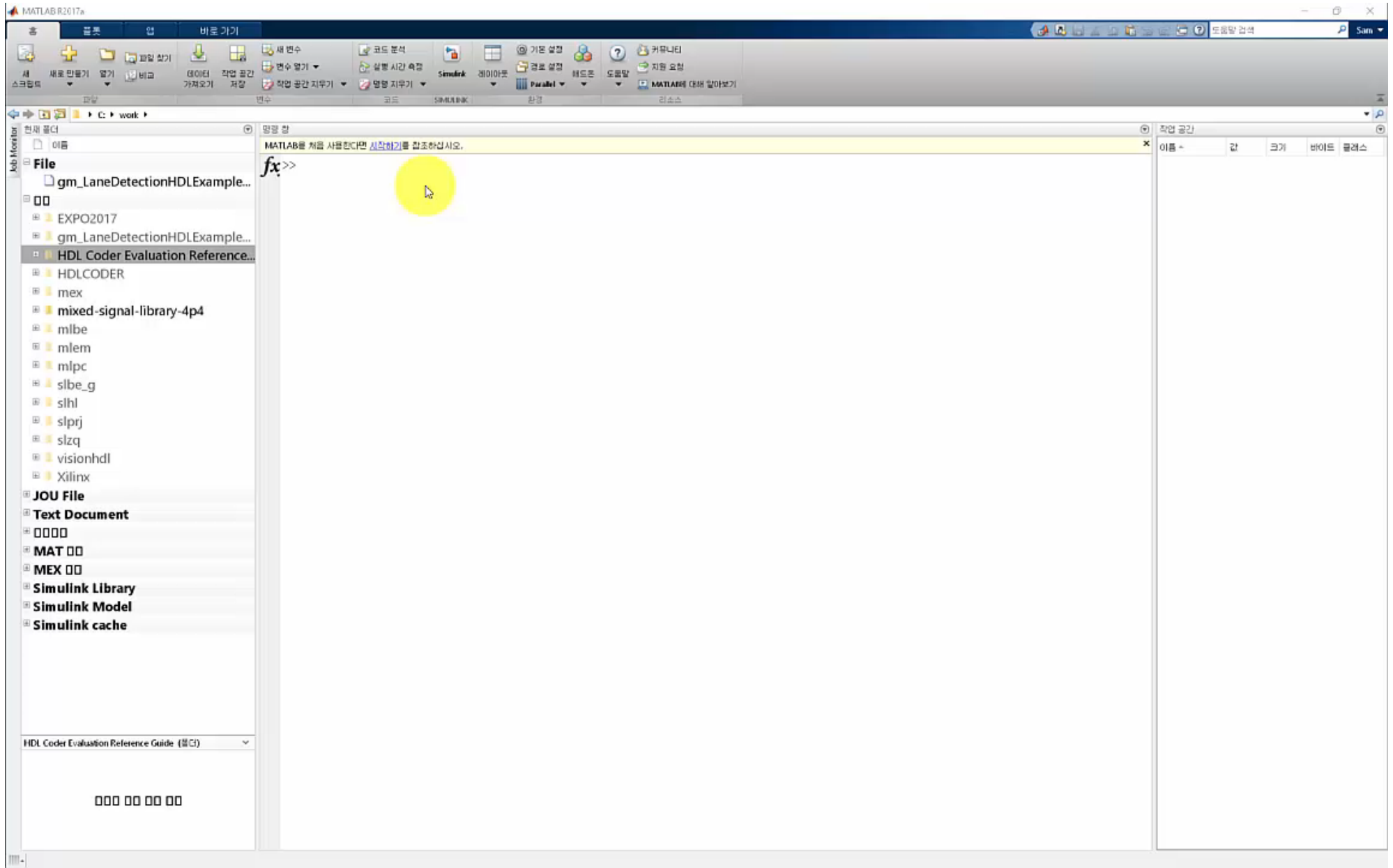
Corner Detection Algorithm

- A corner can be defined as the intersection of two edges
- An approach used within computer vision systems to extract certain kinds of features and infer the contents of an image
- Corner detection is frequently used in motion detection, image registration, video tracking, image mosaicing, panorama stitching, 3D modelling and object recognition

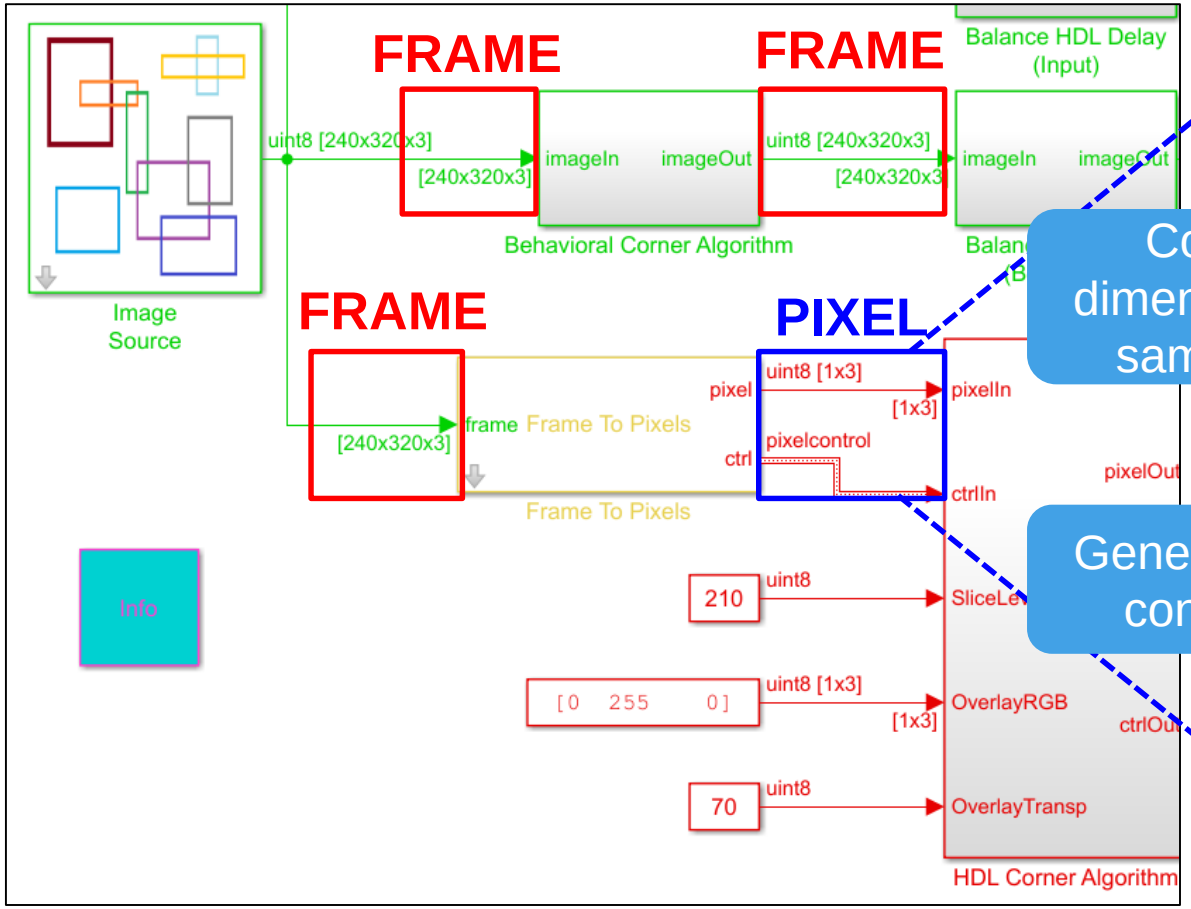


Method	Merit
Harris corner detection *(by Harris & Stephens)*	Accurate results
Minimum eigenvalue *(by Shi & Tomasi)*	Fastest computation
Local intensity comparison *(Features from Accelerated Segment Test, FAST by Rosten & Drummond)*	Trade-off between accuracy and computation

Demo : CornerDetectionHDLExample

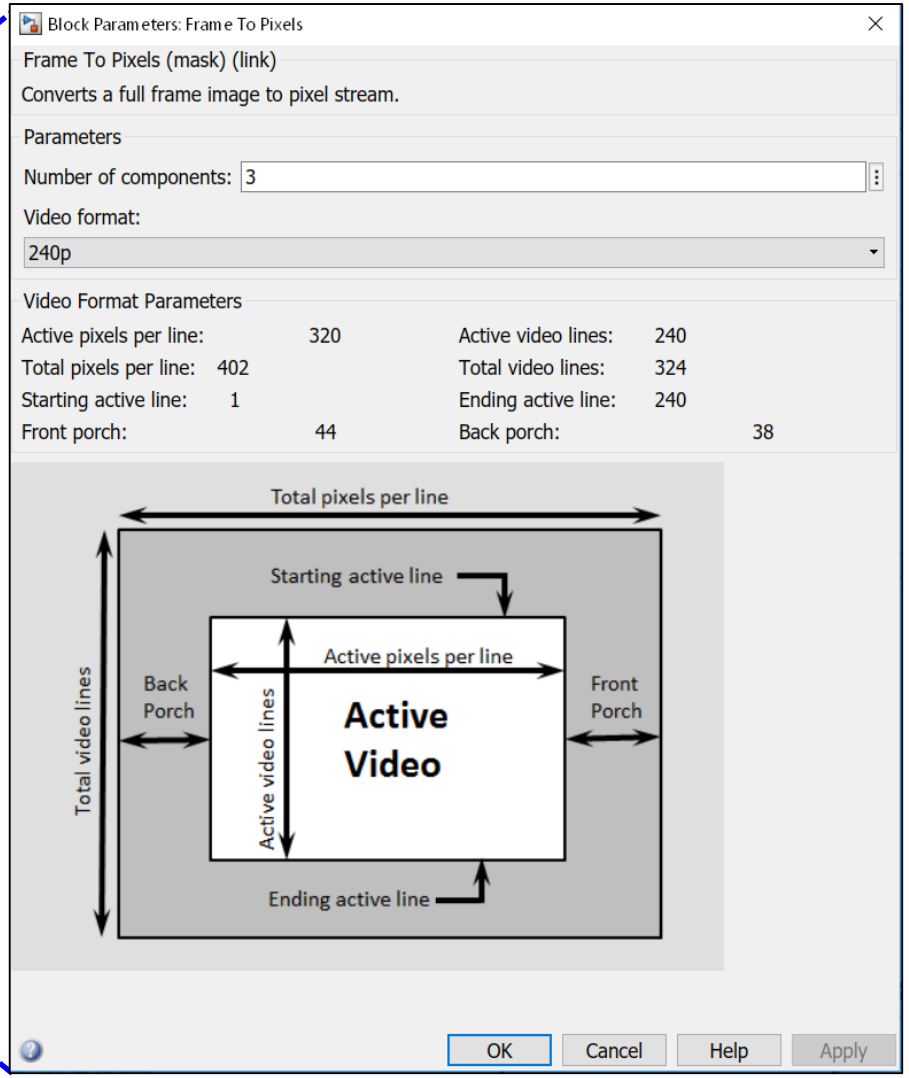


From frame-based to streaming-pixel



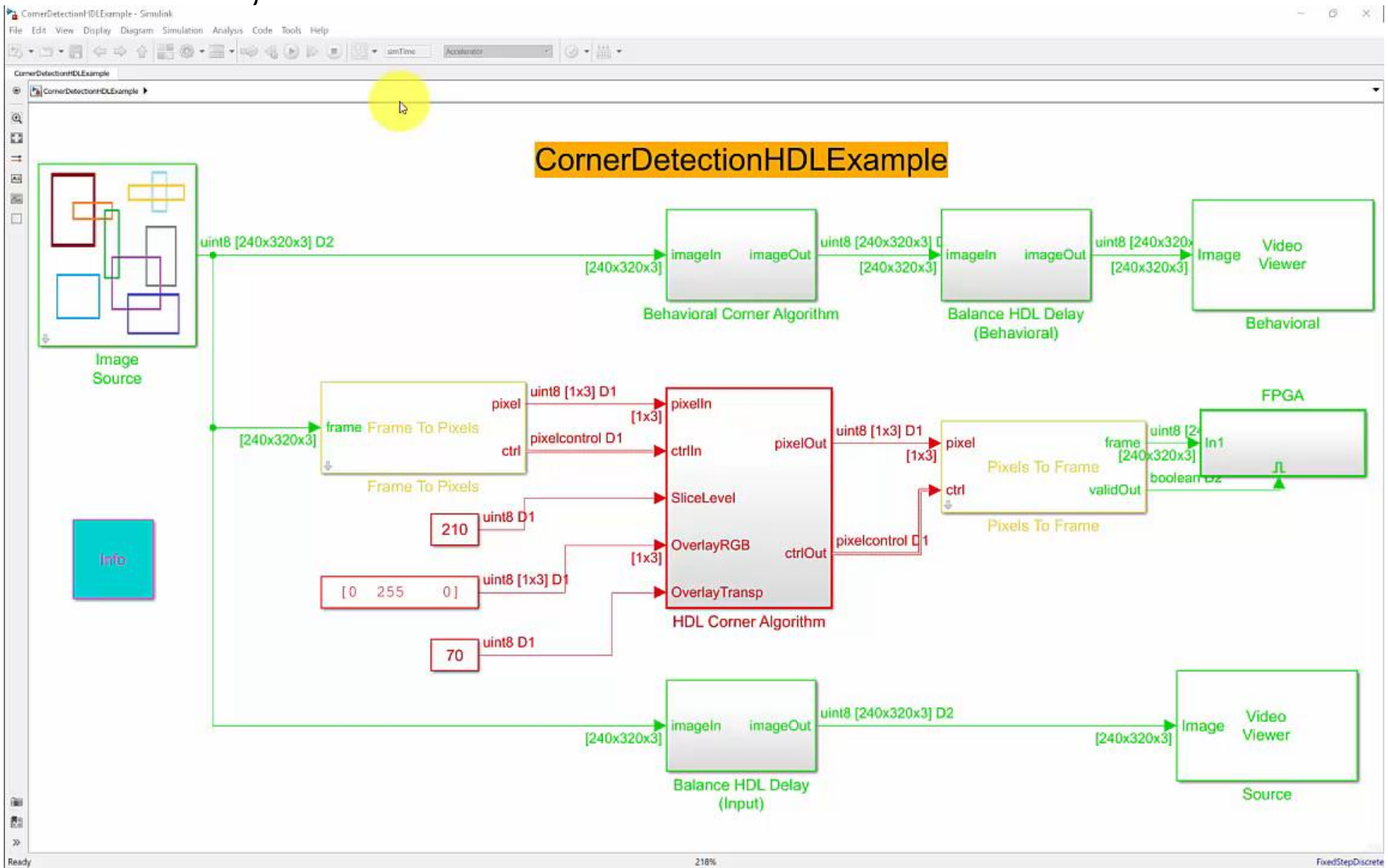
Converts dimensions and sample rate

Generates pixel control bus



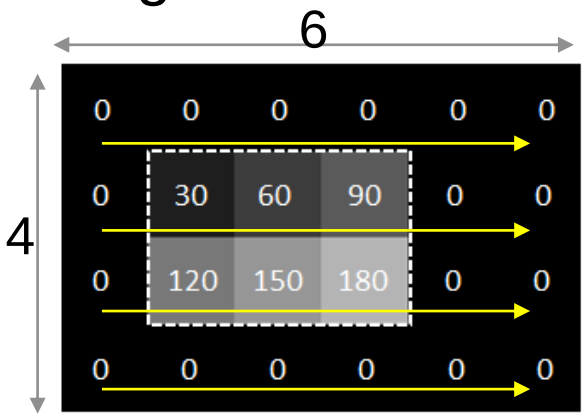
Useful blocks for streaming-pixel conversion

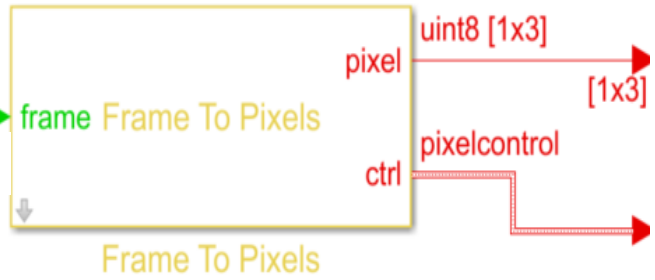
- Frame to Pixel, Pixel to Frame

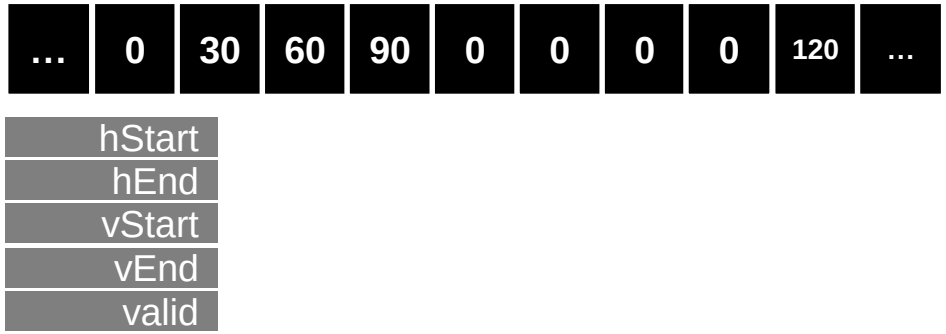


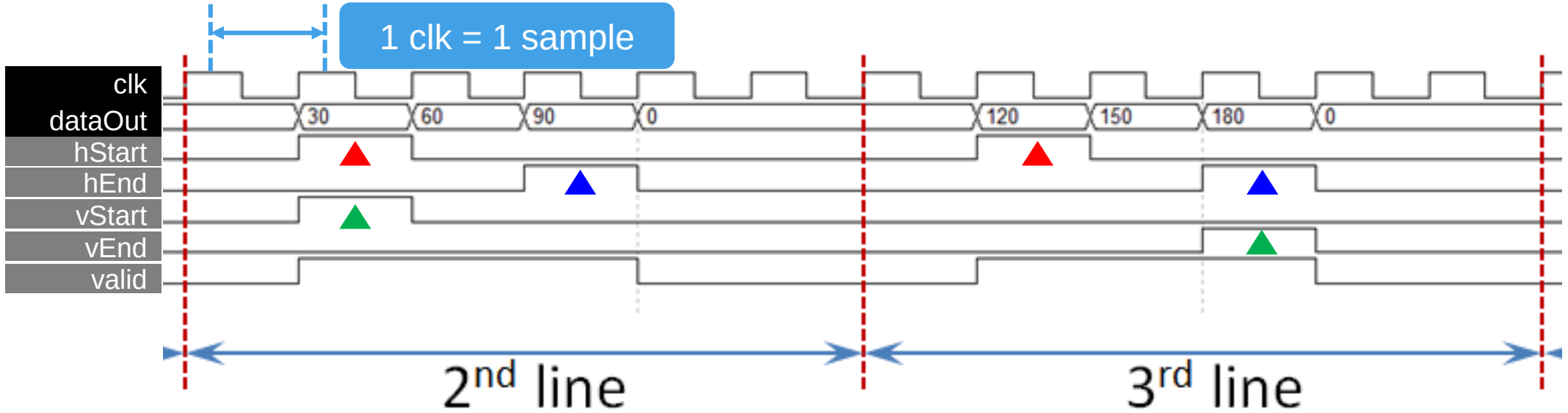
Managing the Pixel Stream from Active Video Sources

- Vertical and Horizontal Blanking Intervals
- Algorithm needs to handle sync signals



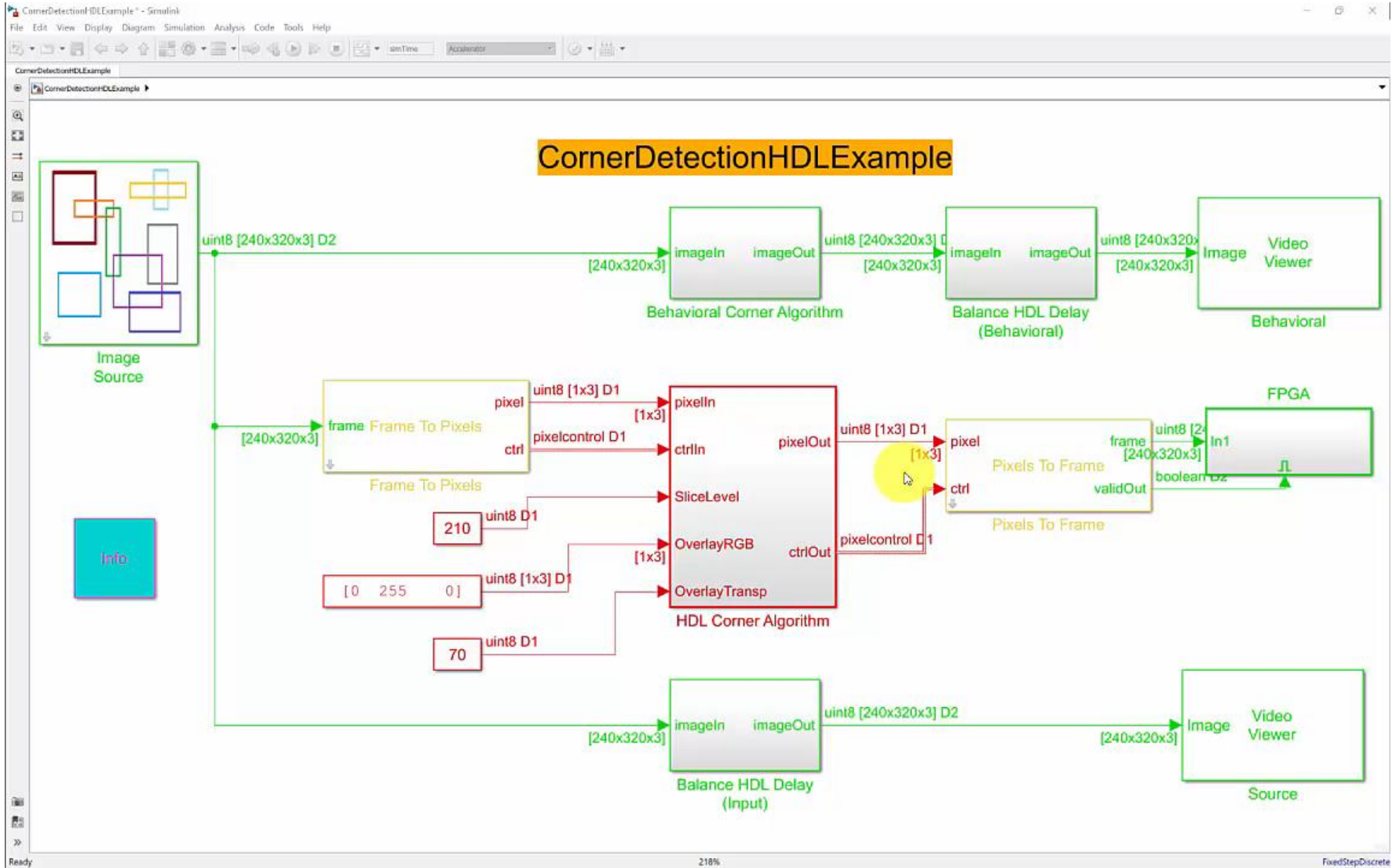




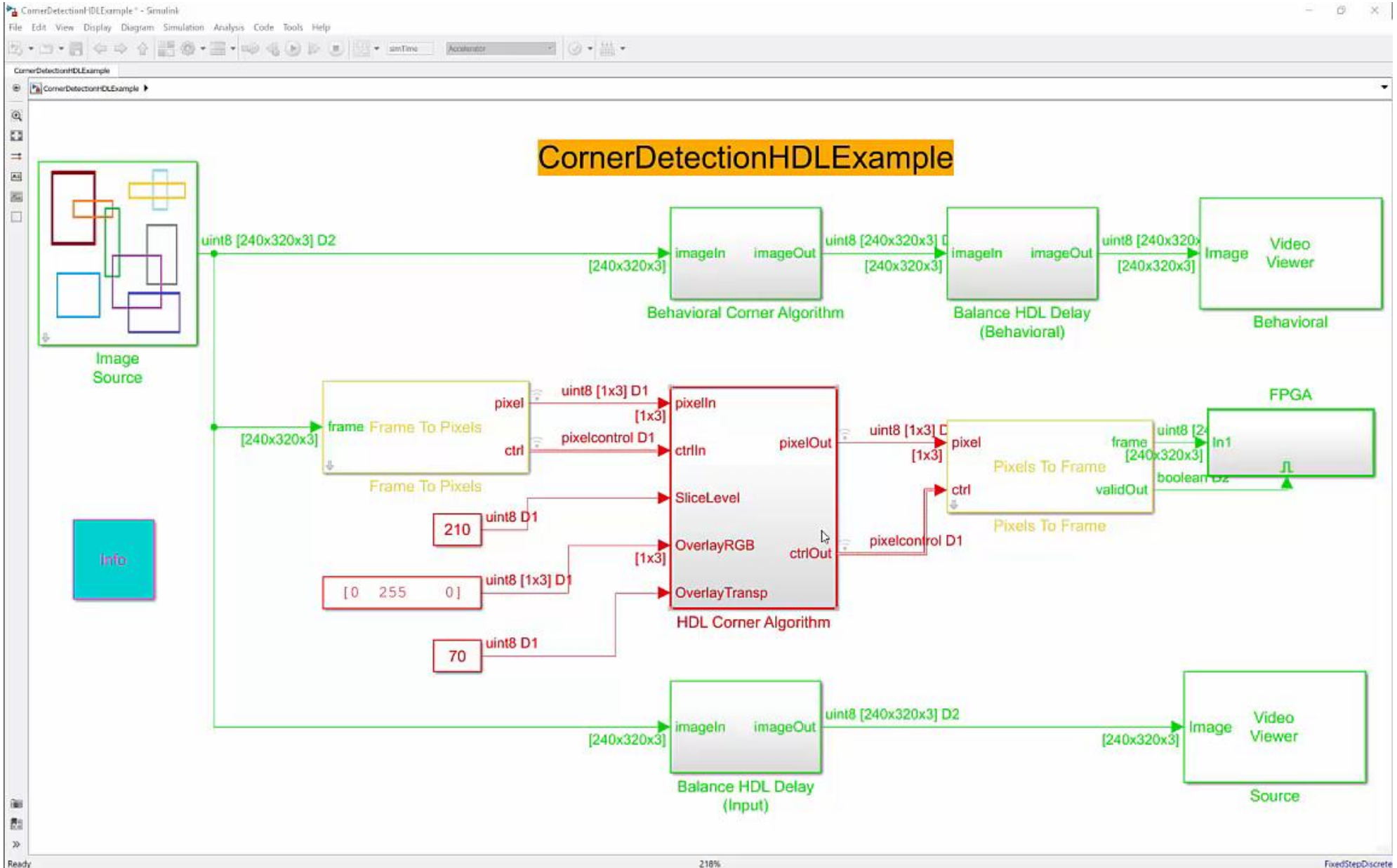


Probing Signal Values using Logic Analyzer

- Simply inspect and compare signal data in model (DSP System Toolbox™ is Required)

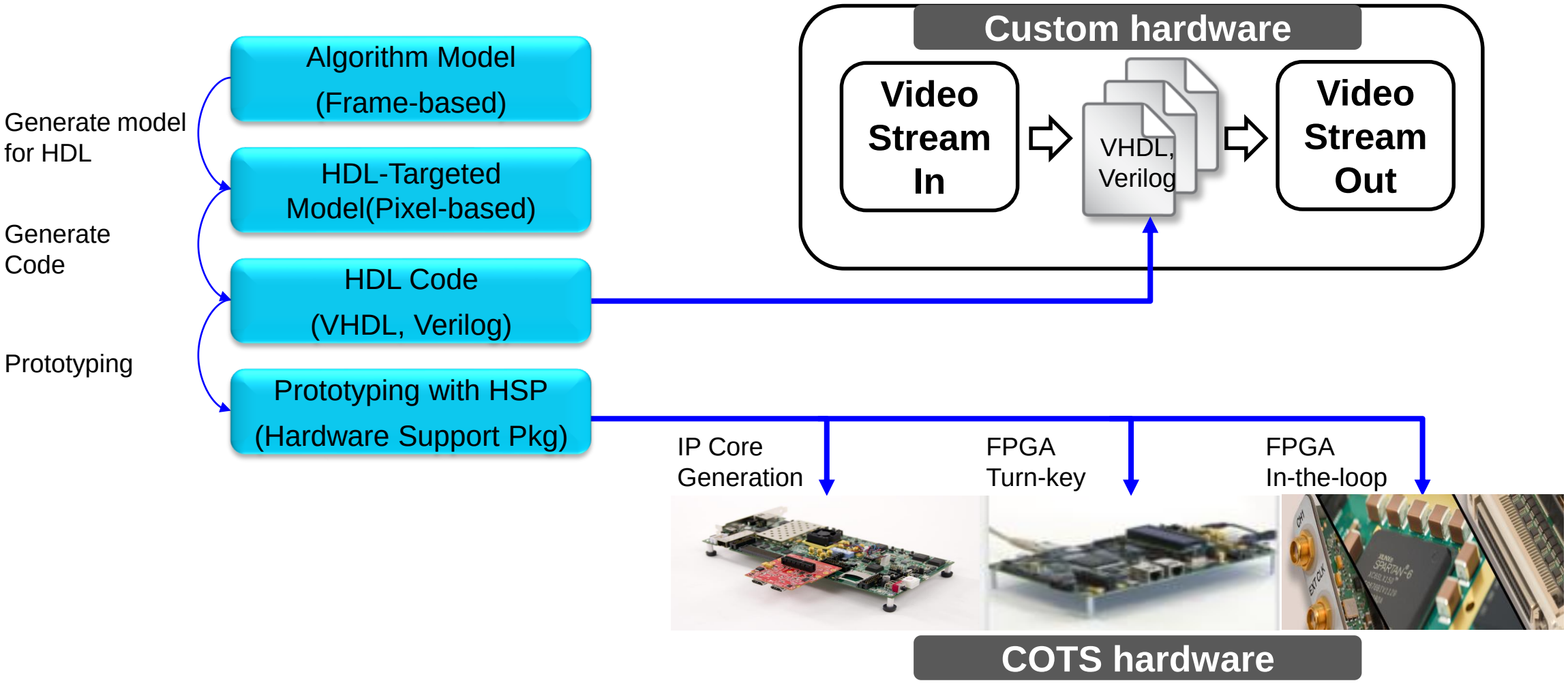


HDL Code Generation for Hardware

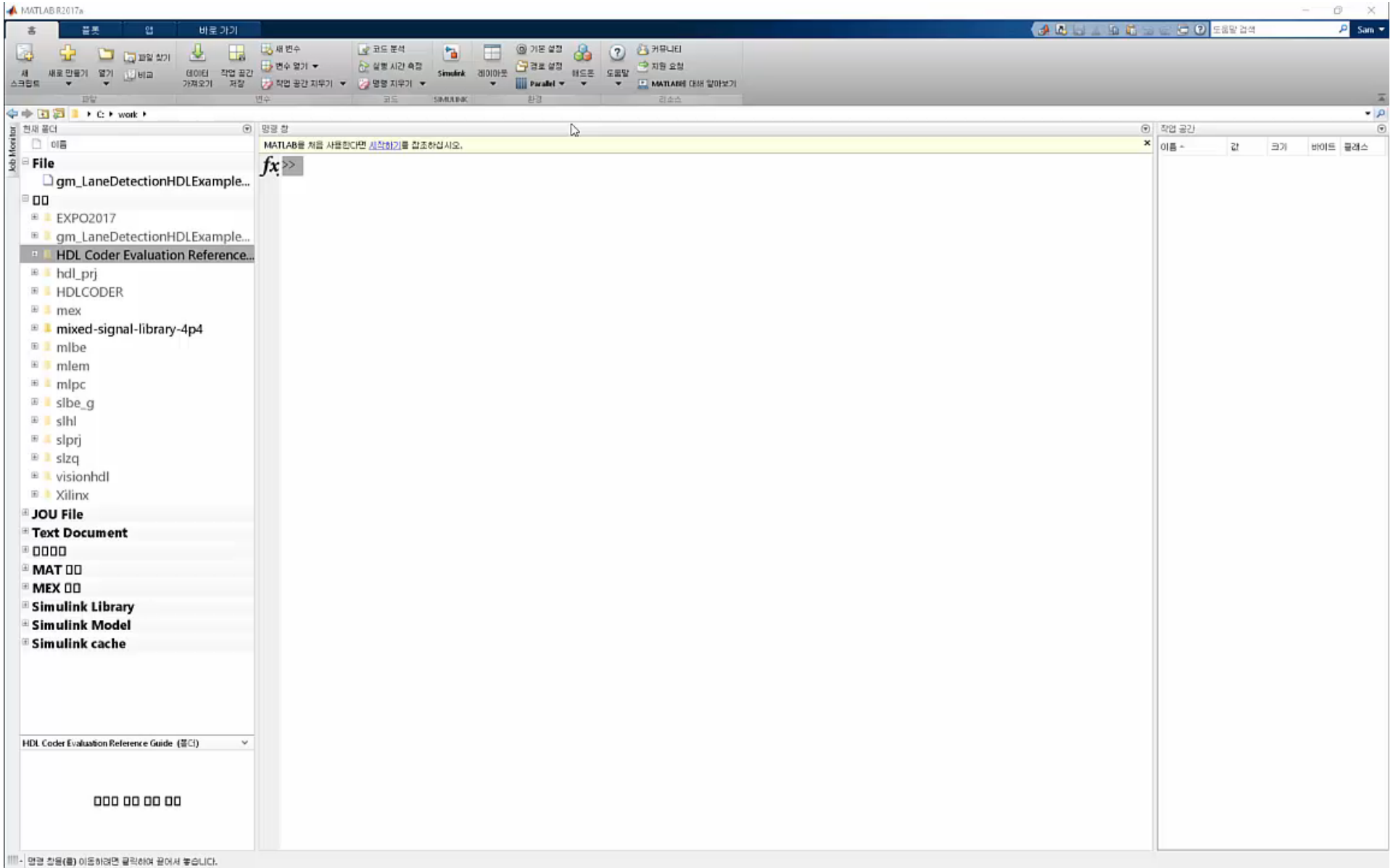


Deployment on Hardware

- Merge generated HDL code to User code
- Use Hardware Support Package for rapid prototyping



Install Hardware Support Package (HSP)



Summary : Workflow from Algorithm to Hardware

Generate model for HDL

Generate Code

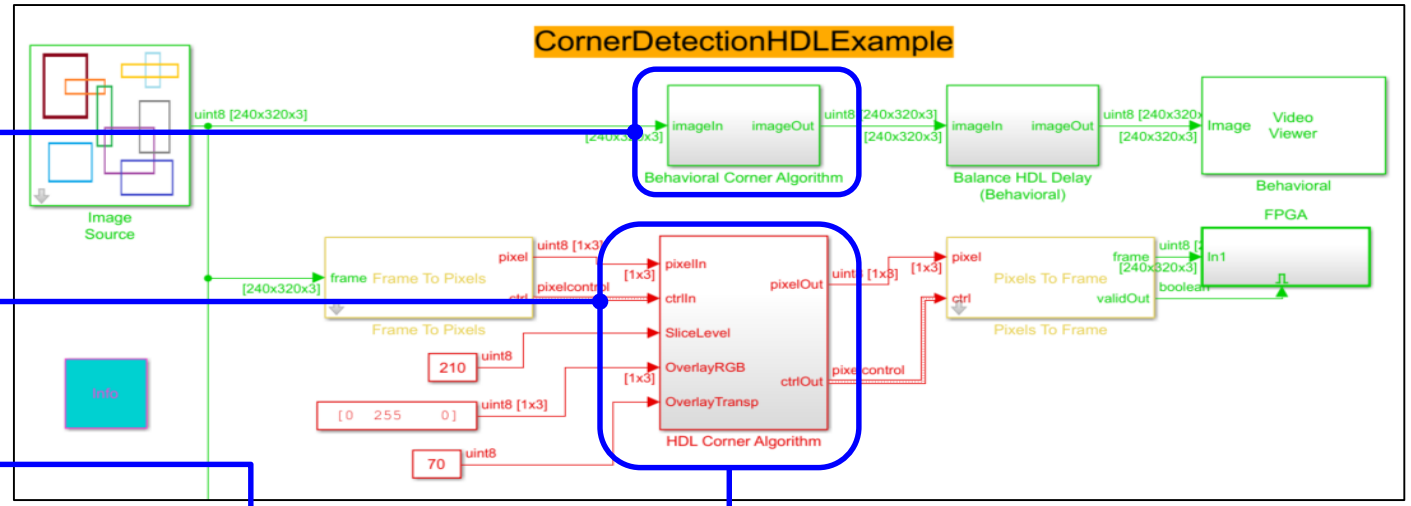
Prototyping

Algorithm Model (Frame-based)

HDL-Targeted Model(Pixel-based)

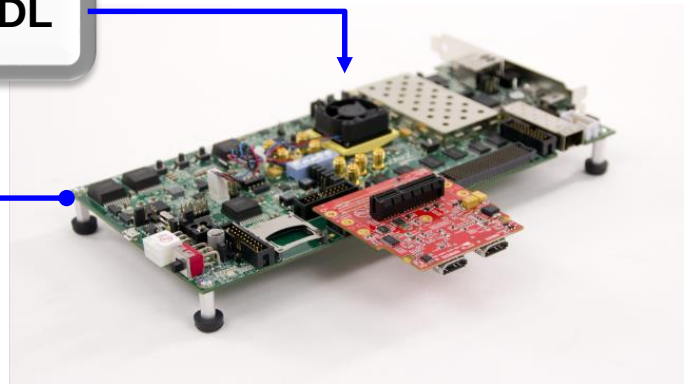
HDL Code (VHDL, Verilog)

Prototyping with HSP (**H**ardware **S**upport **P**kg)



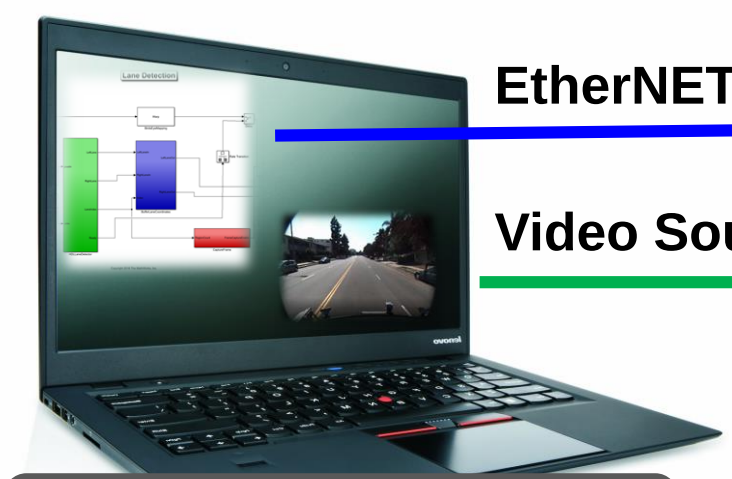
HDL

Fast Prototyping



Target Hardware

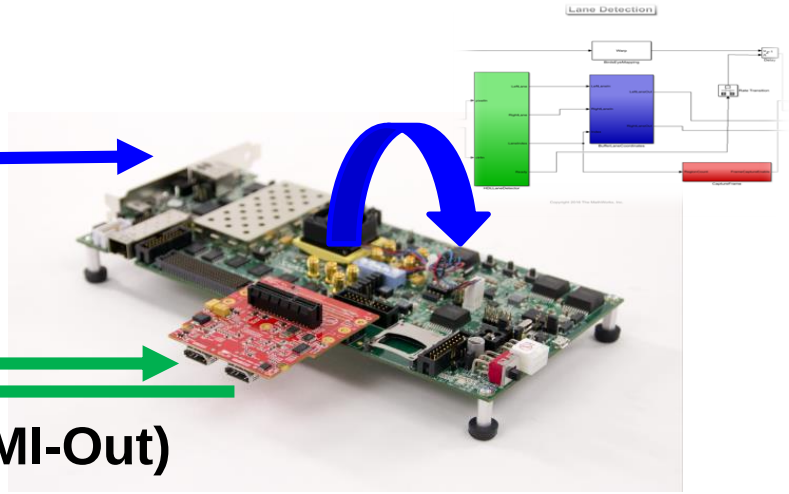
Lane Detection System Demo



EtherNET

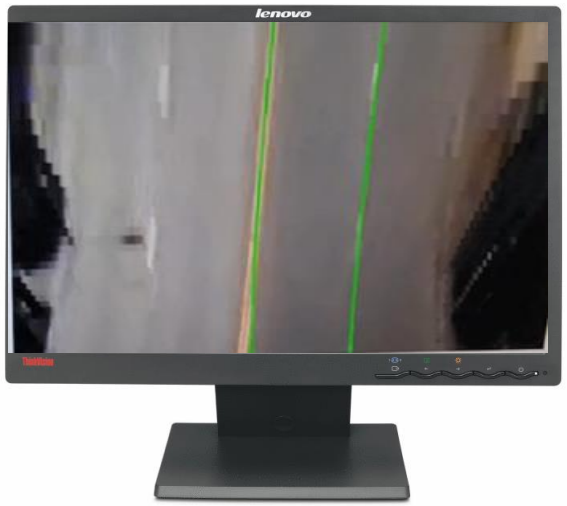
Video Source(HDMI-In)

Processed Video(HDMI-Out)



Video Source Simulink

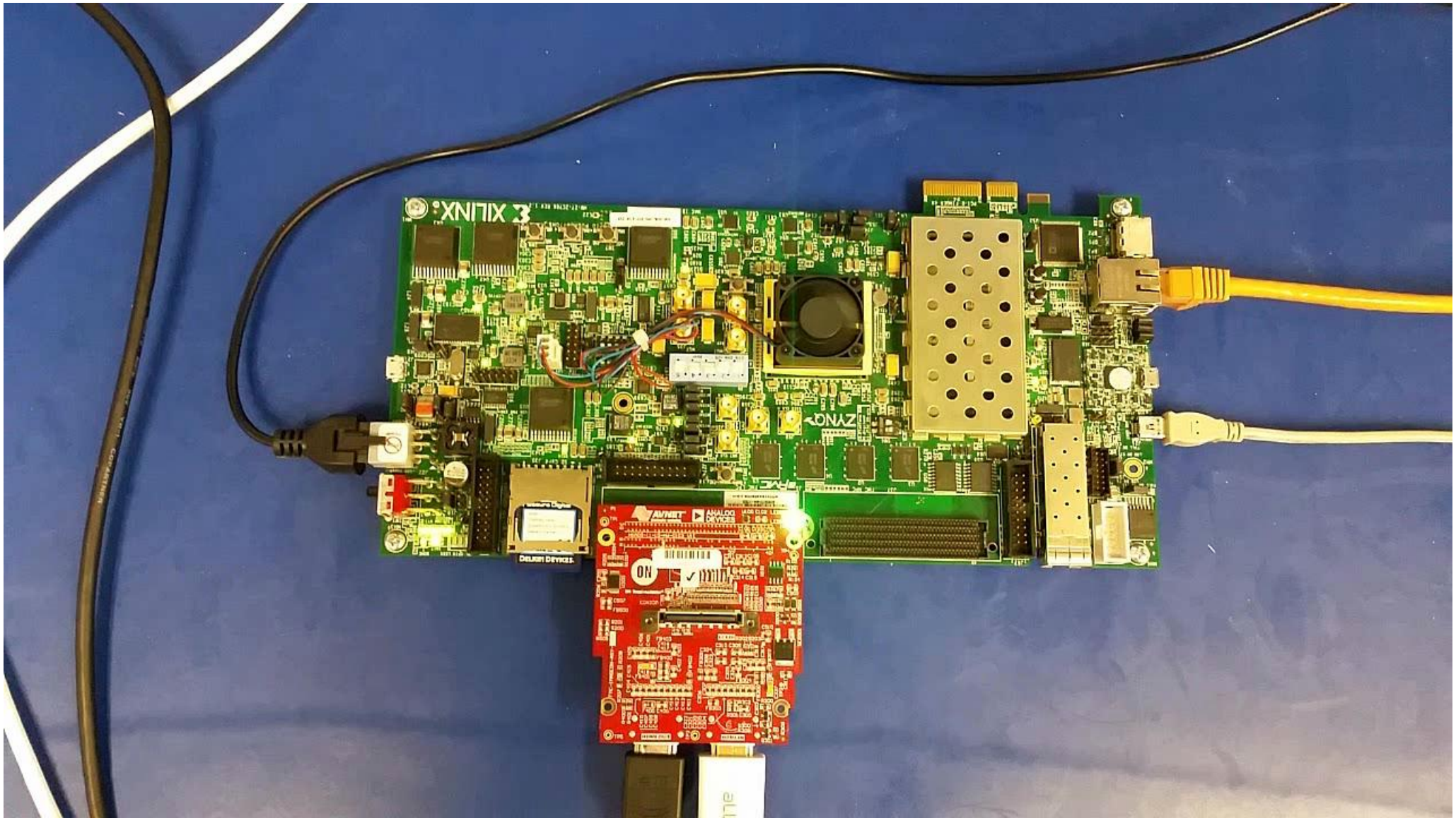
Hardware Target



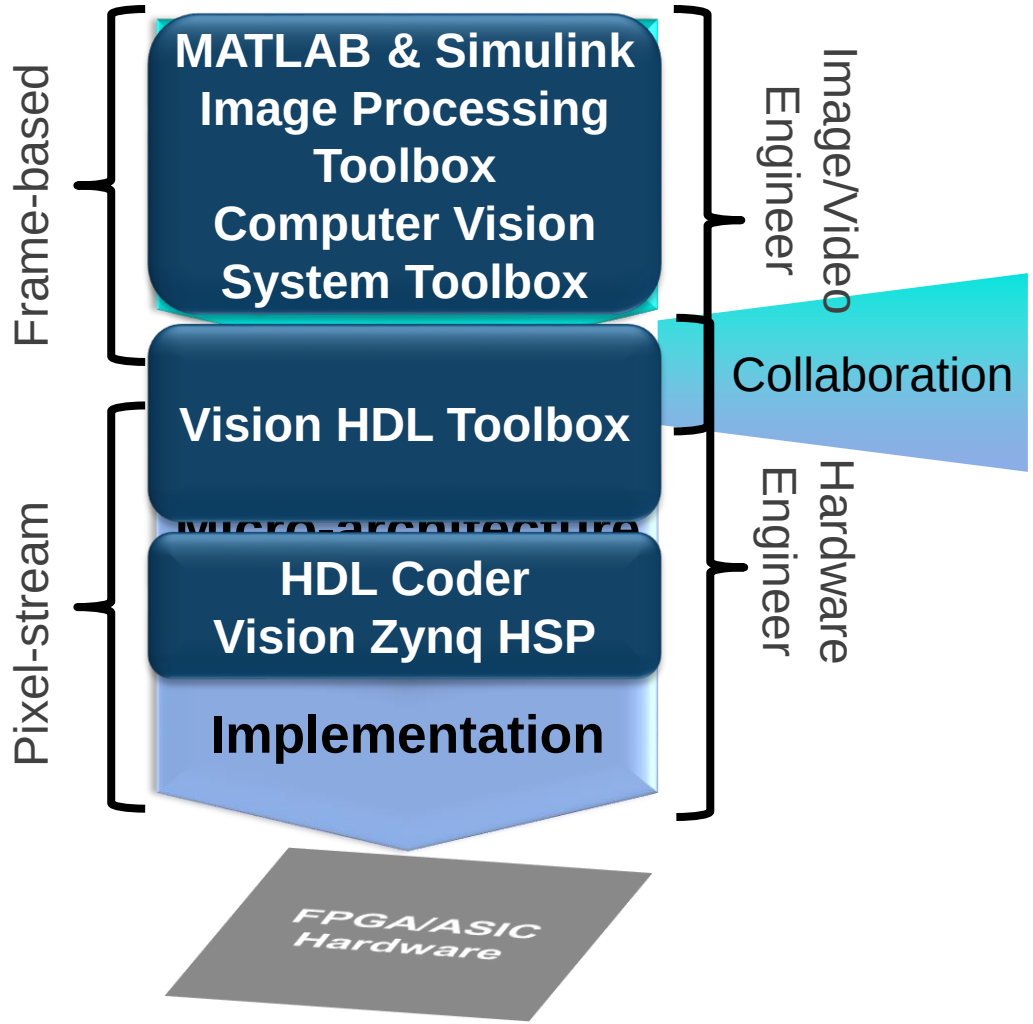
Display

Lane Detection System in Action

Computer Vision System Toolbox™ Support Package for Xilinx® Zynq®-Based Hardware



Workflow From Frames to Pixels to Hardware



- New application innovation happens at the system-level
  - Implemented across software and hardware
- Successful implementation requires collaboration
- Connected workflow to FPGA/ASIC hardware delivers:
  - Broader micro-architecture exploration
  - Agility to make changes, simulate, generate code
  - Continuous verification