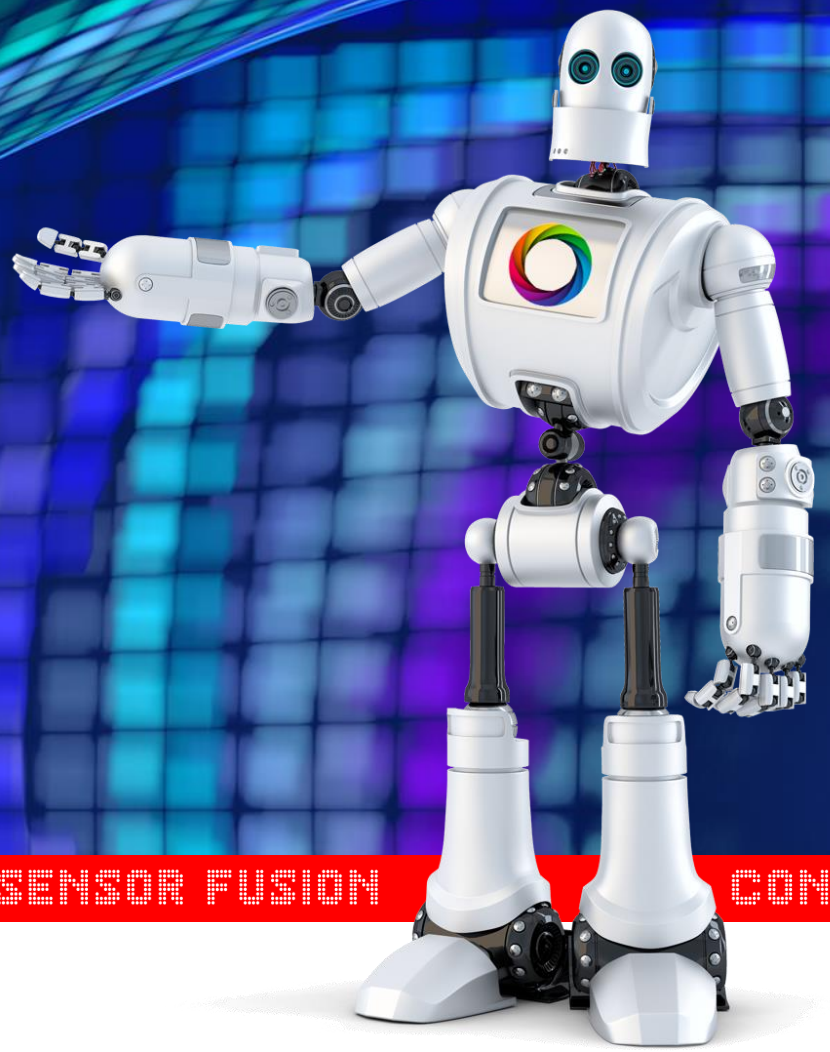
reVISION™

Responsive and Reconfigurable Vision Systems



MACHINE LEARNING | COMPUTER VISION | SENSOR FUSION | CONNECTIVITY

XILINX
ALL PROGRAMMABLE™

OpenCV on Zynq – Deep Dive : Embedded Vision Acceleration with xfOpenCV

정웅 부장, DSP Specialist

Mar 13, 2018

reVISION Stack

OpenVX Vision Acceleration | Caffe

Frameworks

OpenCV | SDSoC Environment | DNN, CNN, GoogLeNet, SSD, FCN …

Libraries and Tools




Development Kits

Agenda

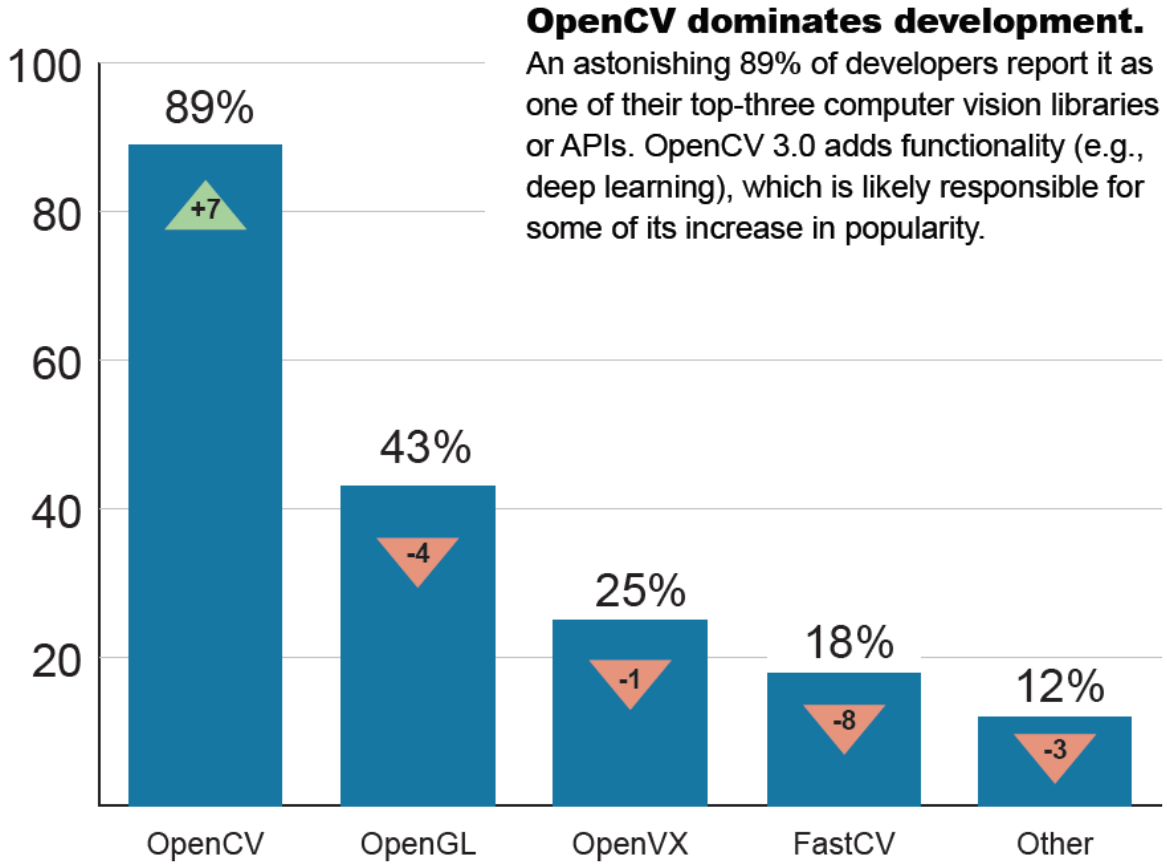
- Why Zynq SoCs for Traditional Computer Vision
- xfOpencv design flow quick review
- Revision Package installation & run example design
- xfOpencv Library API
- Lab1 – Soble filter
- Lab2 – Add dilation and erosion after Soble filter
- Summary

Why Opencv with Xilinx

Opencv Needs Acceleration in Embedded

- Typical ARM Cortex-A53

Typical Requirement	> 30 FPS
Harris Corner	2.4 FPS
Stereo Depth Map	2.1 FPS
Dense Optical Flow	0.1 FPS



OpenCV dominates development.
An astonishing 89% of developers report it as one of their top-three computer vision libraries or APIs. OpenCV 3.0 adds functionality (e.g., deep learning), which is likely responsible for some of its increase in popularity.

Source: Embedded Vision Alliance,
Embedded Vision Developer Survey, January 2017

Zynq Offer Superior Performance, Latency

<10

ms latency

Real Time Applications Latency

Xilinx Benchmark

42x

Frames/sec/watt

Computer Vision

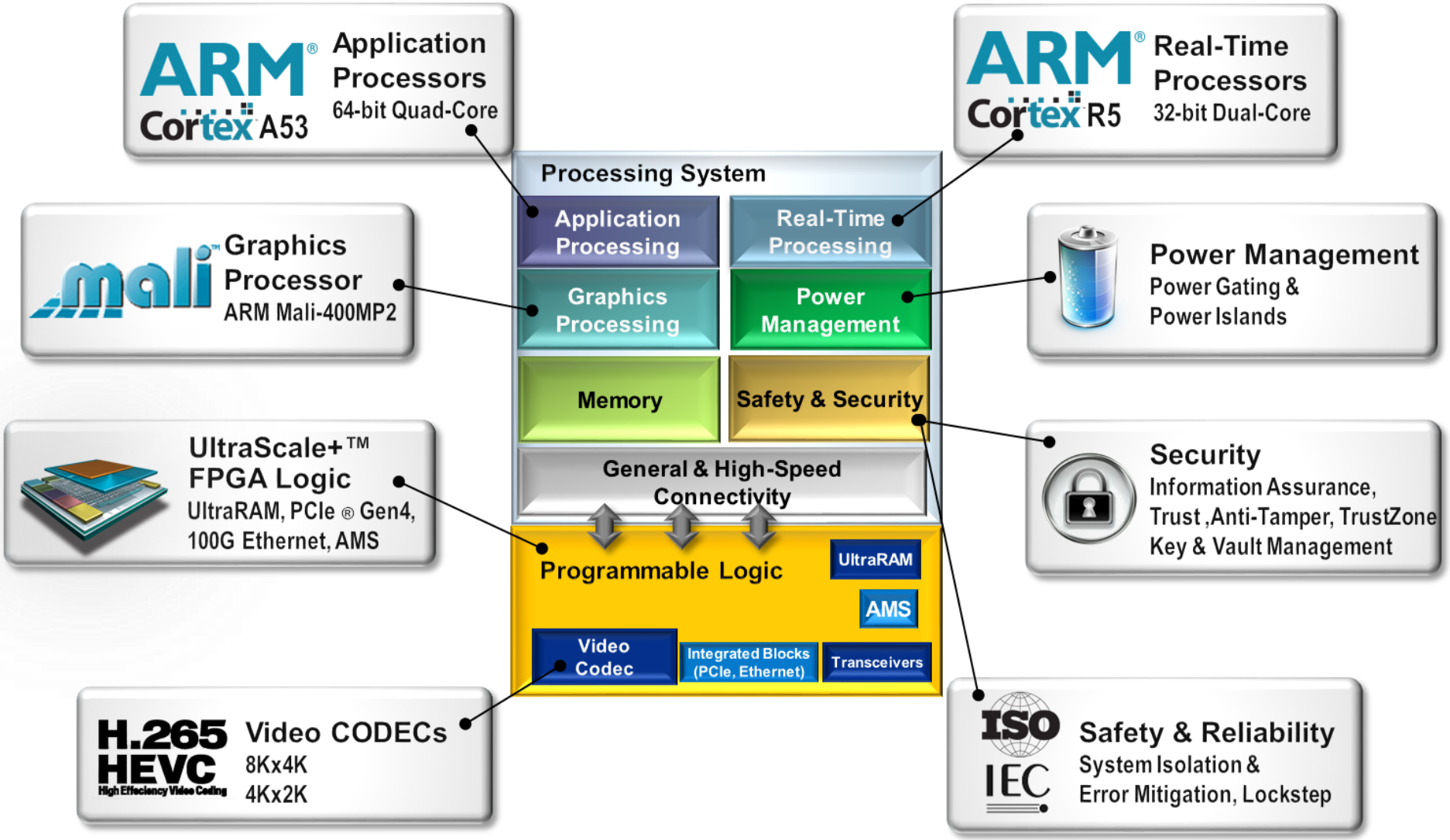
Xilinx Benchmark

cv:: StereoLBM @1080p		Xilinx ZU9	Xilinx ZU5	eGPU*
	Frames/s	**700**	**296**	43
	Power (W)	**4.8**	**3.3**	7.9
	Frames/s/watt	**145.8**	**89.7**	5.4

cv:: LK Dense Optical Flow @720p		Xilinx ZU9	Xilinx ZU5	eGPU*
	Frames/s	**170**	**73**	7
	Power (W)	**4.8**	**3.3**	7.9
	Frames/s/watt	**35.4**	**22.1**	0.9

- eGPU = nVidia Tegra X1 using VisionWorks for StereoLBM and Opencv4Tegra for OpticalFlow
- All benchmarks utilize as much resources as possible on GPU (~99%) and programmable logic (~70%)

Zynq Offers the Most Efficient cv Acceleration



Zynq Offer Superior Performance, Latency

<10

ms latency

Real Time Applications Latency

Xilinx Benchmark

42x

Frames/sec/watt

Computer Vision

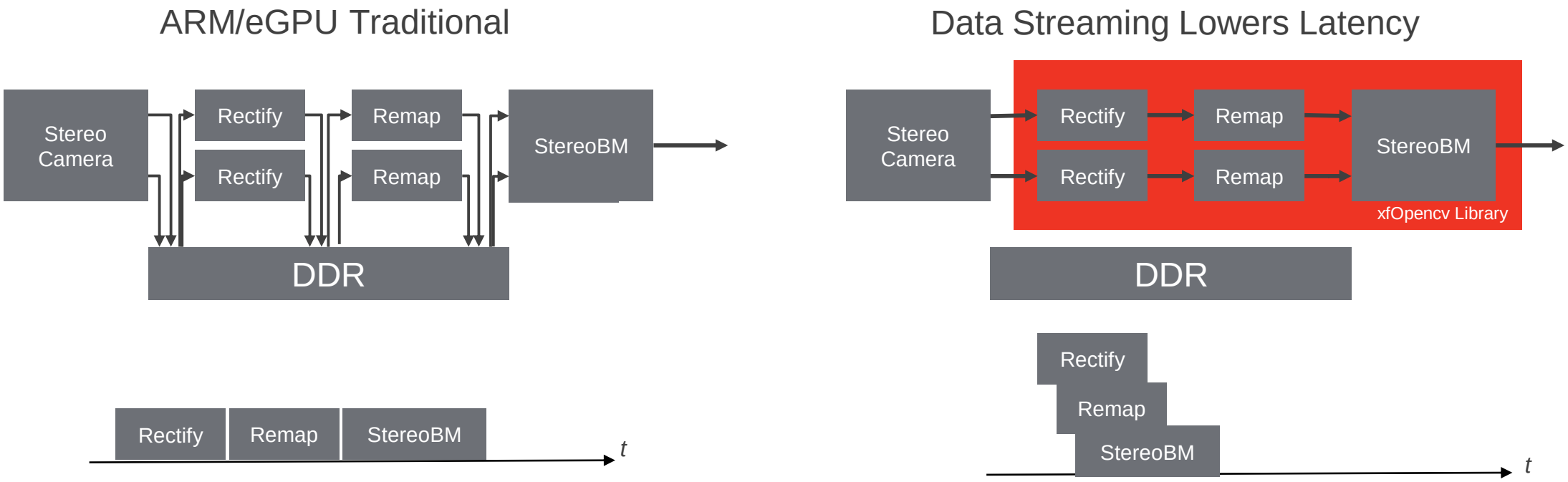
Xilinx Benchmark

cv:: StereoLBM @1080p		Xilinx ZU9	Xilinx ZU5	eGPU*
	Frames/s	**700**	**296**	43
	Power (W)	**4.8**	**3.3**	7.9
	Frames/s/watt	**145.8**	**89.7**	5.4

cv:: LK Dense Optical Flow @720p		Xilinx ZU9	Xilinx ZU5	eGPU*
	Frames/s	**170**	**73**	7
	Power (W)	**4.8**	**3.3**	7.9
	Frames/s/watt	**35.4**	**22.1**	0.9

- eGPU = nVidia Tegra X1 using VisionWorks for StereoLBM and Opencv4Tegra for OpticalFlow
- All benchmarks utilize as much resources as possible on GPU (~99%) and programmable logic (~70%)

Why So Good? Efficient Window-based Streaming cv increases Performance



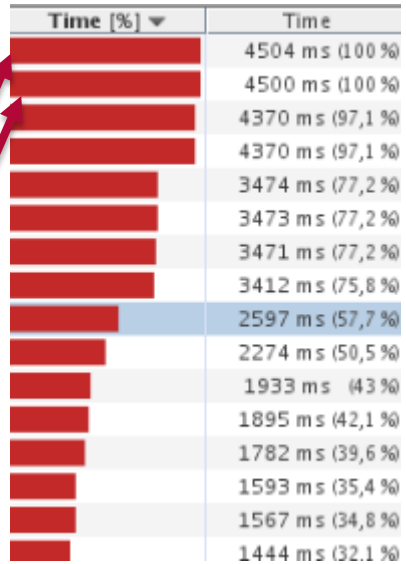
- xfOpencv directly infers pipelining functions from one to the next, avoiding frame buffers and external memory

xfOpencv design flow quick review

Opencv Support with Automatic HW Acceleration

1. Cross-compile Opencv application to Zynq (ARM A9/A53)
2. Profile and identify bottleneck functions
3. Minimal changes to the code and set functions to hardware.
   Compile using SDSoC
4. Run on a Zynq board

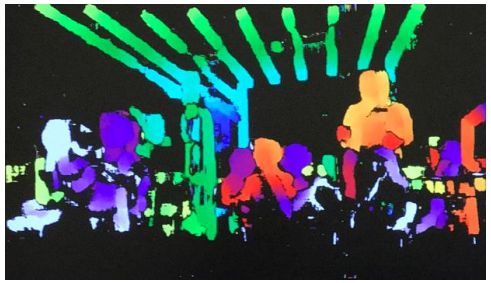

```
main(){
cv::imread(A);
cv::stereoRectify(A,B,C,D);
cv::stereoLBM(C,D,out);
cv::imshow(out);
}
```



HW functions

Name	Clock Frequency (MHz)
stereoRectify	300
stereoLBM	300

```
main(){
cv::imread(A);
xF:stereoRectify<line>(A,B,C,D);
xF:stereoLBM<win,n_disp>(C,D,out);
cv::imshow(out);
}
```



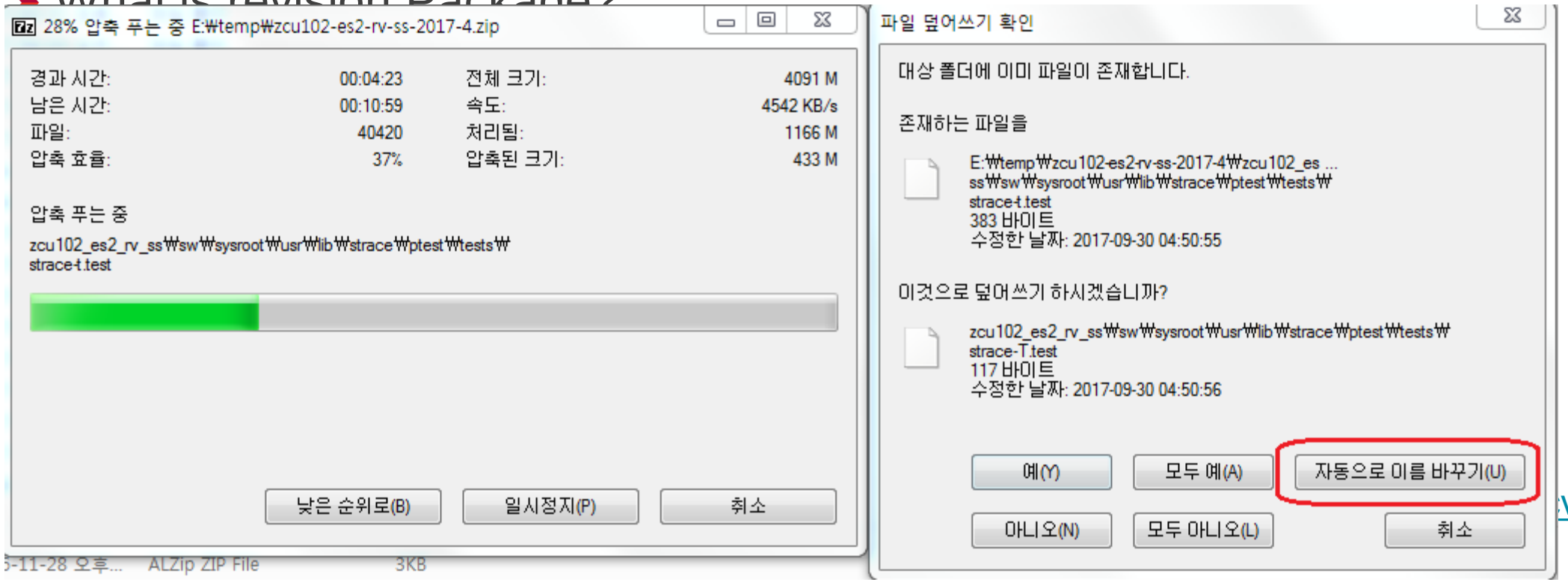
xfOpencv: 50+ Most Needed Opencv Functions

Basic Functionality	Geometric Transforms	Image Processing and Filters	Feature Detection and Classifiers	3D Reconstruction	Motion Analysis and Tracking
Absolute difference	Scale/Resize	Box	Canny edge detection	StereoLBM	Mean Shift Tracking (MST)
Accumulate	StereoRectify	Gaussian	Fast corner		LK Dense Optical Flow
Accumulate squared	Warp Affine	Median	SVM (binary)		
Accumulate weighted	Warp Perspective	Sobel	Harris corner		
Arithmetic addition	Remap	Custom convolution	Histogram of Oriented Gradients (HOG)		
Arithmetic subtraction		Equalize Histogram			
Bitwise: AND, OR, XOR, NOT		Dilate			
Pixel-wise multiplication		Erode			
Channel combine		Bilateral			
Channel extract		OTSU Thresholding			
Color convert		Thresholding			
Convert bit depth		Image pyramid			
Table lookup		Color Detection			
		Integral image			
		Gradient Magnitude			
		Histogram			
		Gradient Phase			
		Min/Max Location			
		Mean & Standard Deviation			

Revision Package installation & run example design

Revision Package

- What is revision Package?



- xfOpencv library
  - Download from Github, https://github.com/Xilinx/xfopencv

Getting Started

Prerequisites

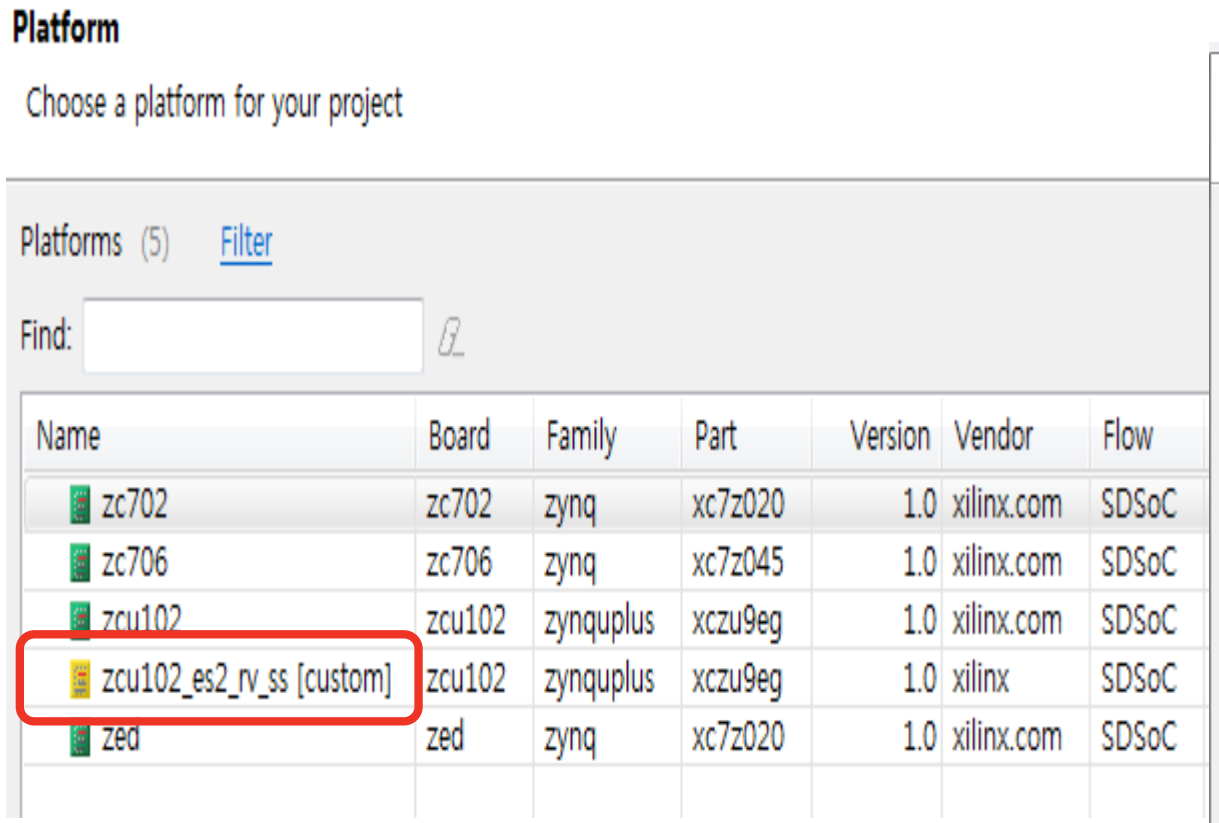
- Install SDx
- Download zcu102_[es2]rv_ss
- Before launching SDx

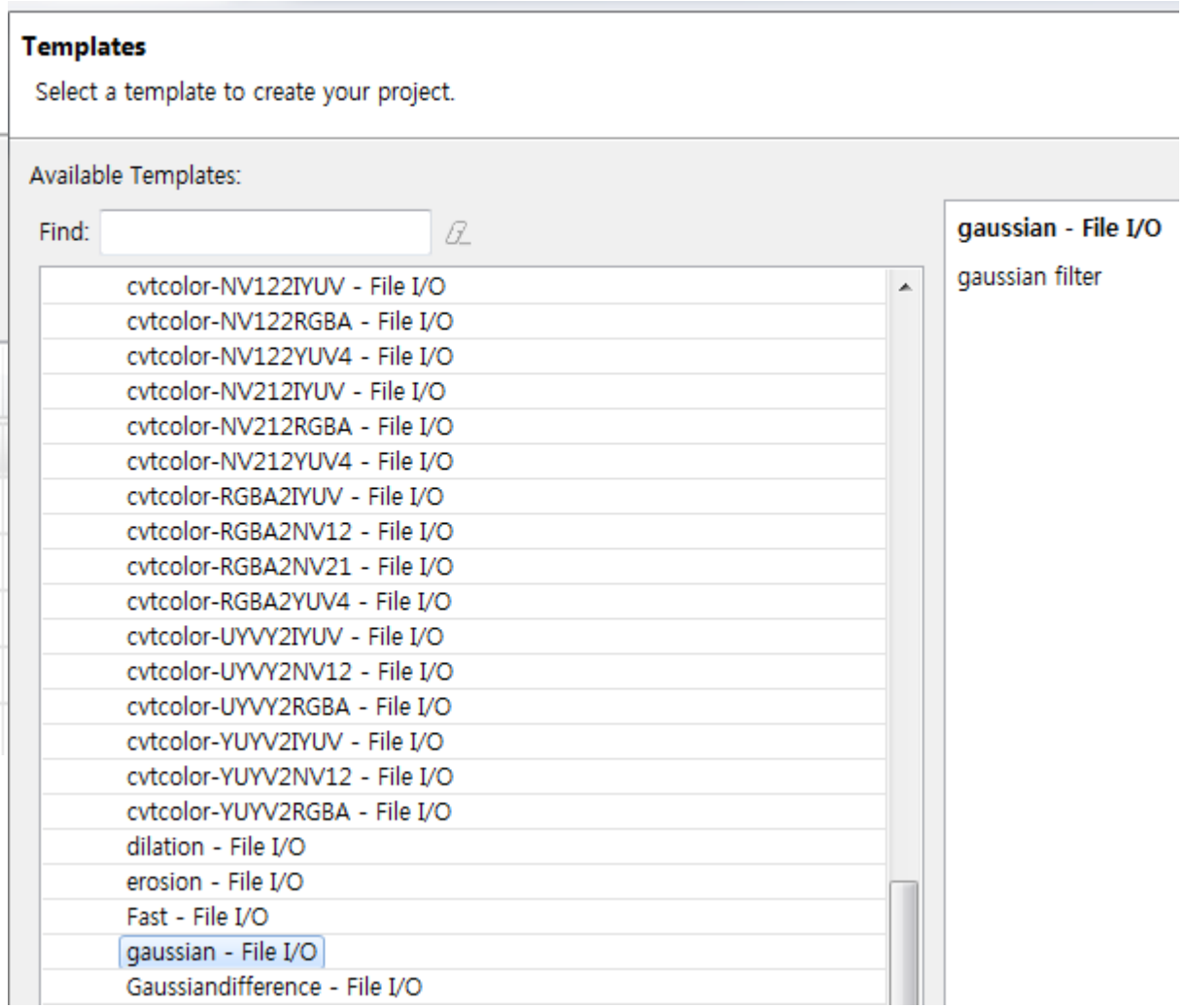
  - **IMPORTANT:** Before starting SDx, set the **SYSROOT environment variable** to point to the Linux root file system delivered with the EV platform, for example:

```
Linux:
export SYSROOT=<local>/zcu102_[es2_]rv_ss/sw/sysroot

Windows:
Start->Control Panel->System->Advanced->Environment Variables
Create environment variable SYSROOT with value <local>\zcu102_[es2_]rv_ss\sw\sysroot
```

- The very first time you do this for a new workspace, you must hit Add Custom Platform..., and browse to ./zcu102_[es2_]rv_ss, hit OK, note that "zcu102_[es2_]rv_ss (custom)" now appears in the Name column.

Using reVISION Samples on the reVISION Platform





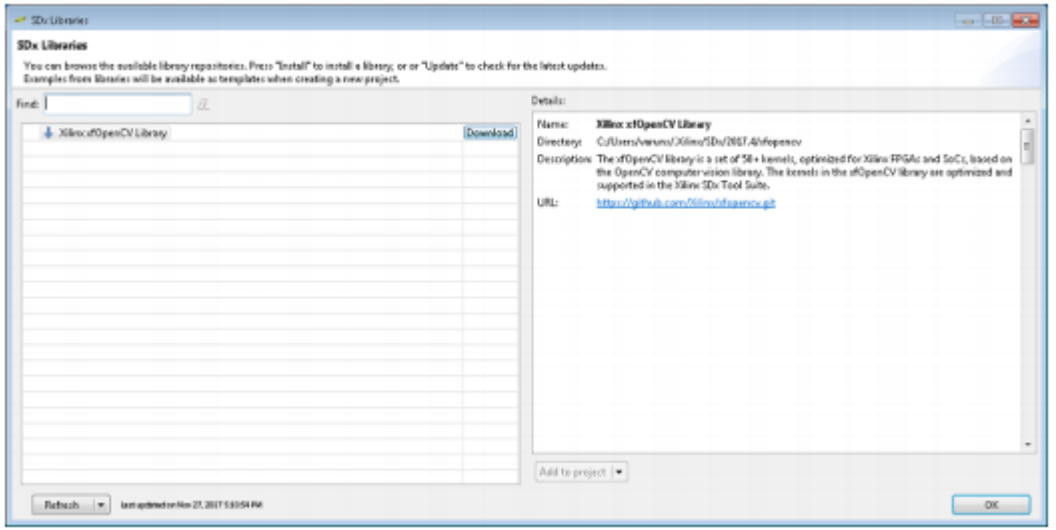
- Select revision platform
- Choose example design

Using the xfOpencv library

- Include header file in source code

Folder	Details
`include`	Contains the header files required by the library.
`include/common`	Contains the common library infrastructure headers, such as types specific to the library.
`include/core`	Contains the core library functionality headers, such as the `math` functions.
`include/features`	Contains the feature extraction kernel function definitions. For example, `Harris`.
`include/imgproc`	Contains all the kernel function definitions, except the ones available in the `features` folder.
`examples`	Contains the sample test bench code to facilitate running unit tests. The `examples/` folder contains the folders with algorithm names. Each algorithm folder contains host files, `.json` file, and `data` folder. For more details on how to use the xfOpenCV library, see xfOpenCV Kernel on the reVISION Platform.

- Include whole libraries in SDx project creation



Using the xfOpencv Library on a non-reVISION Platform

- Generate empty project
- Add required libs ➔ recommend libs in revision package
- Add sources (coding or importing)
- Setup Build settings
  - Add directories
  - Add libs(linker)
  - Add lib search paths(linker)
- Define HW functions
- Build
- Runs SD_Card image (project/Debug(Release)/sd_card)

xfOpencv Library API

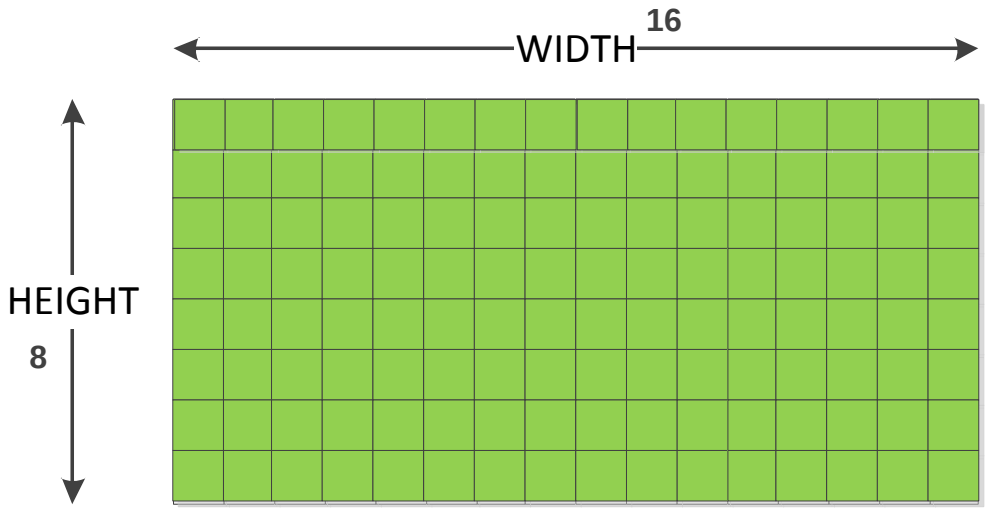
Xf::Mat Image Container Class – UG1233

```
/* Template class of Mat */
template<int T, int ROWS, int COLS, int NPC>
class Mat {
public:
    unsigned char allocatedFlag; //flag to mark memory allocation in this class
    int rows, cols,size;                // actual image size

#ifndef __SYNTHESIS__
    XF_TNAME(T,NPC) *data;
#else
    XF_TNAME(T,NPC) data[ROWS*(COLS>> (XF_BITSHIFT(NPC)))];
#endif
    Mat();                          // default constructor

    Mat(int _rows, int _cols);
    Mat(int _size, int _rows, int _cols);
    Mat(int _rows, int _cols, void *_data);
    ~Mat();

    // copy constructor
    Mat(const Mat& src)
        {

    //Assignment operator
    Mat& operator=(const Mat& src)
    {

    void init(int _rows, int _cols);
    void copyTo(void* fromData);
    unsigned char* copyFrom();

    template<int DST_T>
    void convertTo(Mat<DST_T,ROWS, COLS, NPC> &dst, int otype, double alpha=1, double beta=0);


};
```

Xf::Mat Class Parameter Descriptions – UG1233

Table 4: **xf::Mat Class Parameter Descriptions**

Parameter	Description
rows	The number of rows in the image or height of the image.
cols	The number of columns in the image or width of the image.
size	The number of words stored in the data member. The value is calculated using `rows*cols/(number of pixels packed per word).`
*data	The pointer to the words that store the pixels of the image.



Xf::Mat Member Function Descriptions – UG1233

Table 5: **xf::Mat Member Function Descriptions**

Member Functions	Description
Mat()	This default constructor initializes the Mat object sizes, using the template parameters ROWS and COLS.
Mat(int _rows, int _cols)	This constructor initializes the Mat object using arguments _rows and _cols.
Mat(const xf::Mat &_src)	This constructor helps clone a Mat object to another. New memory will be allocated for the newly created constructor.
Mat(int _rows, int _cols, void *_data)	This constructor initializes the Mat object using arguments _rows, _cols, and _data. The *data member of the Mat object points to the memory allocated for _data argument, when this constructor is used. No new memory is allocated for the *data member.
convertTo(Mat<DST_T,ROWS, COLS, NPC> &dst, int otype, double alpha=1, double beta=0)	Refer to xf::convertTo
copyTo(* fromData)	Copies the data from Data pointer into physically contiguous memory allocated inside the constructor.
copyFrom()	Returns the pointer to the first location of the *data member.
~Mat()	This is a default destructor of the Mat object.

Xf::Mat Template Parameter Descriptions – UG1233

Table 6: **xf::Mat Template Parameter Descriptions**

Parameters	Description
TYPE	Type of the pixel data. For example, XF_8UC1 stands for 8-bit unsigned and one channel pixel. More types can be found in `include/common/xf_params.h`.
HEIGHT	Maximum height of an image.
WIDTH	Maximum width of an image.
NPC	The number of pixels to be packed per word. For instance, XF_NPPC1 for 1 pixel per word; and XF_NPPC8 for 8 pixels per word.

- Pixel Type

Table 8: **xf::Mat Class - Available Data Types**

Option	Number of bits per Pixel	Unsigned/ Signed/ Float Type	Number of Channels
XF_8UC1	8	Unsigned	1
XF_16UC1	16	Unsigned	1
XF_16SC1	16	Signed	1
XF_32UC1	32	Unsigned	1
XF_32FC1	32	Float	1
XF_32SC1	32	Signed	1
XF_8UC2	8	Unsigned	2
XF_8UC4	8	Unsigned	4
XF_2UC1	2	Unsigned	1

Xf::Mat Template Parameter for Parallelism – UG1233

Pixel-level Parallelism

Table 7: **Options Available for Specifying the Level of Parallelism**

Option	Description
XF_NPPC1	Process 1 pixel per clock cycle
XF_NPPC2	Process 2 pixels per clock cycle
XF_NPPC8	Process 8 pixels per clock cycle

XF_NPPC4 가능

Macro to Work with Parallelism

- The `XF_NPIXPERCYCLE(flags)` macro resolves to the number of pixels processed per cycle.
  - `XF_NPIXPERCYCLE(XF_NPPC1)` resolves to 1
  - `XF_NPIXPERCYCLE(XF_NPPC2)` resolves to 2
  - `XF_NPIXPERCYCLE(XF_NPPC8)` resolves to 8
- The `XF_BITSHIFT(flags)` macro resolves to the number of times to shift the image size to right to arrive at the final data transfer size for parallel processing.
  - `XF_BITSHIFT(XF_NPPC1)` resolves to 0
  - `XF_BITSHIFT(XF_NPPC2)` resolves to 1
  - `XF_BITSHIFT(XF_NPPC8)` resolves to 3

Additional Utility Functions for Software – UG1233

- ***xf::imread***
- ***xf::imwrite***
- ***xf::absDiff***
- ***xf::convertTo***

Lab1 – Soble filter

Overall Code Flow

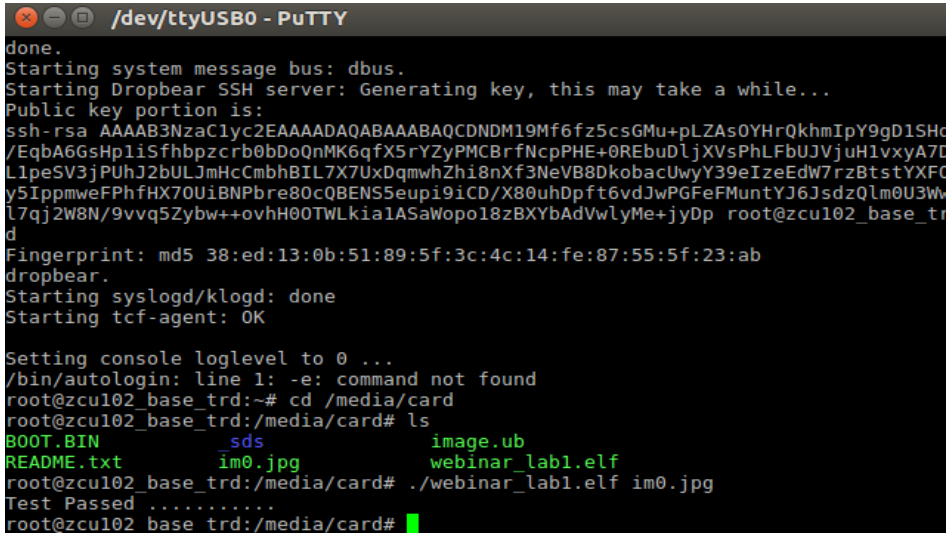
```
#include <stdio.h>
#include <stdlib.h>
#include "xf_headers.h"
#include "xf_sobel_config.h"

int main(int argc, char** argv){
    if(argc != 2)
    {
    char in[100];
    //  Standard OpenCV Rerference ////////////////////////
    // image object define
    {
    // image read
    {
    // Parameter setup
    {
    // create memory for output image
    {
    // CV function call
    {
    // Output image write
    {
    // Xilinx OpenCV ////////////////////////////////////
    // image object define
    {
    // image read
    {
    // Xilinx OpenCV function call
    {
    // Output image write
    {
    // Check correctness ////////////////////////////////
    {
    return return_value;
}
```

- Object Definition
  - cv::Mat() / xf::Mat<T>()
- Image Read
  - cv::imread() / xf::imread()
- Parameter Setup
- cv function call
  - cv::Sobel() / xf::Sobel<T>()
- Output image Write
  - cv::imwrite() / xf::imwrite()

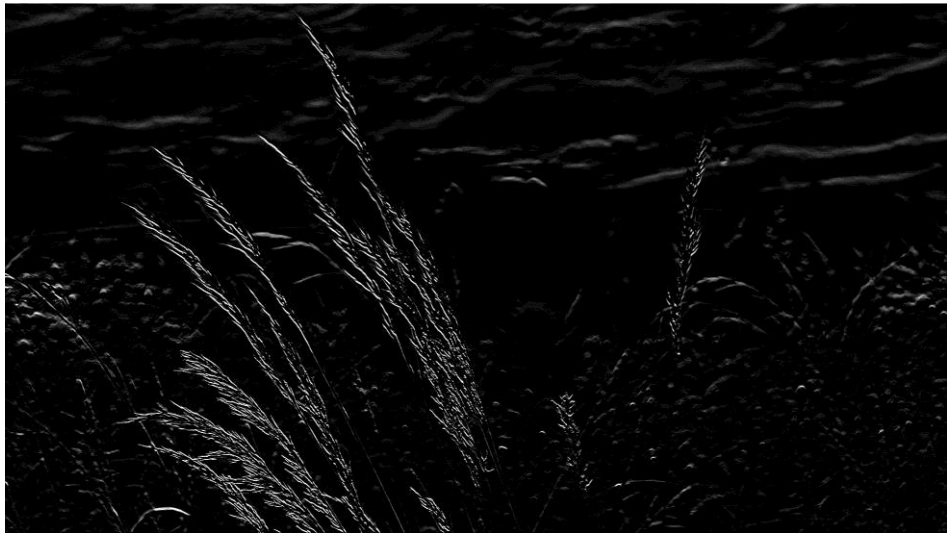
Outputs

Input Image

X-gradient output



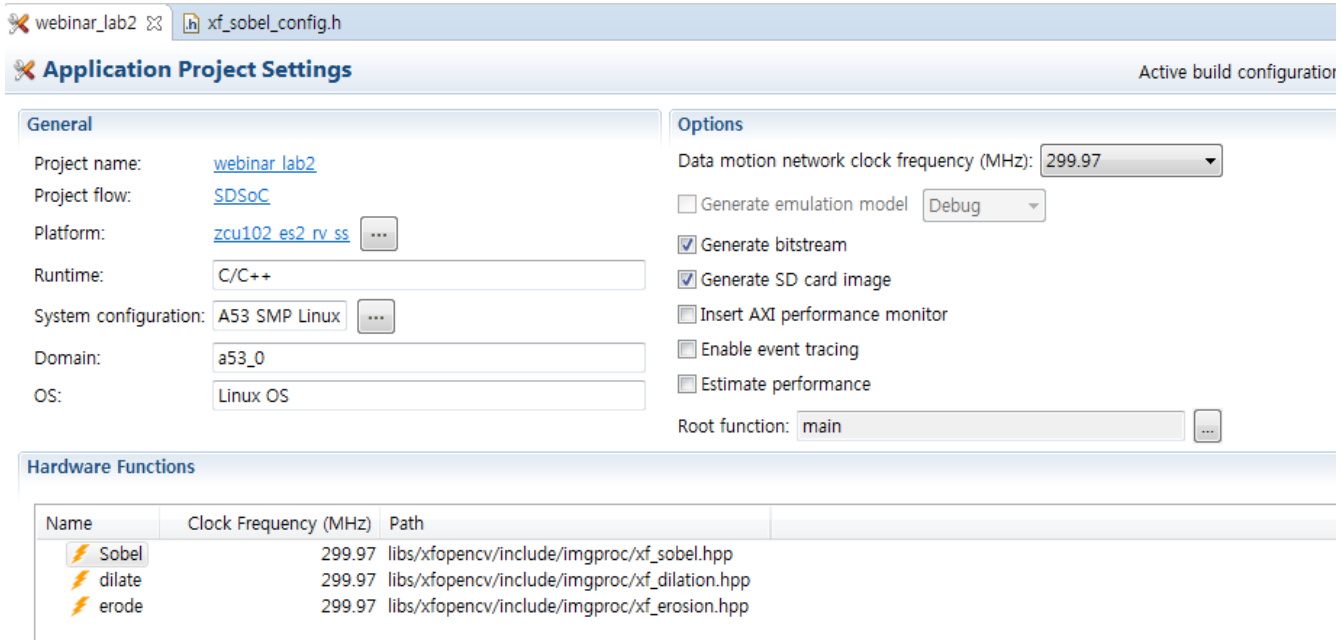
Run in HW



Y-gradient output

Lab2 – Add dilation and erosion after Soble filter

Assign xfOpenCV functions in HW

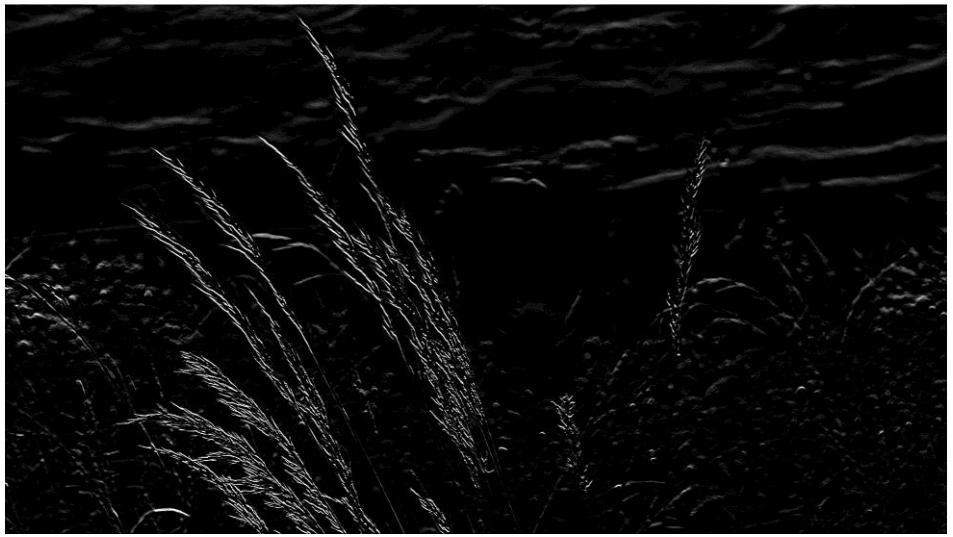



```
2  Copyright (c) 2016, Xilinx, Inc.
0
1  #include "xf_sobel_config.h"
2
3  void sobel_accel(xf::Mat<IN_TYPE, HEIGHT, WIDTH, NPC1> &_src,xf::Mat<OUT_TYPE, HEIGHT, WIDTH, NPC1> &_dstg
4  {
5
6      xf::Sobel<XF_BORDER_CONSTANT,FILTER_WIDTH,IN_TYPE,OUT_TYPE,HEIGHT, WIDTH,NPC1>(_src, _dstgx,_dstgy);
7      xf::dilate<XF_BORDER_CONSTANT,OUT_TYPE,HEIGHT, WIDTH,NPC1>(_dstgx, _dilation);
8      xf::erode<XF_BORDER_CONSTANT,OUT_TYPE,HEIGHT, WIDTH,NPC1>(_dstgy, _erosion);
9
0  }
```

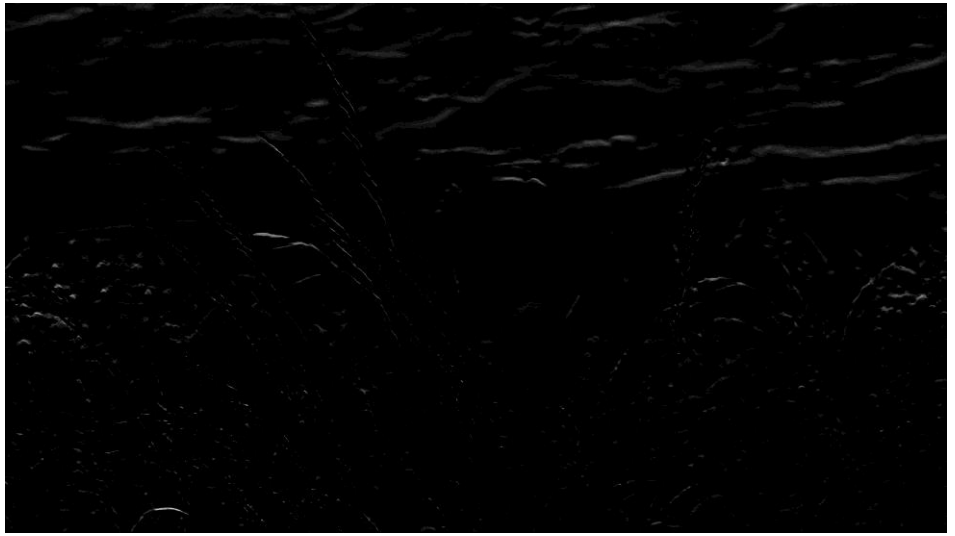
Outputs

X-gradient output

Dilation output

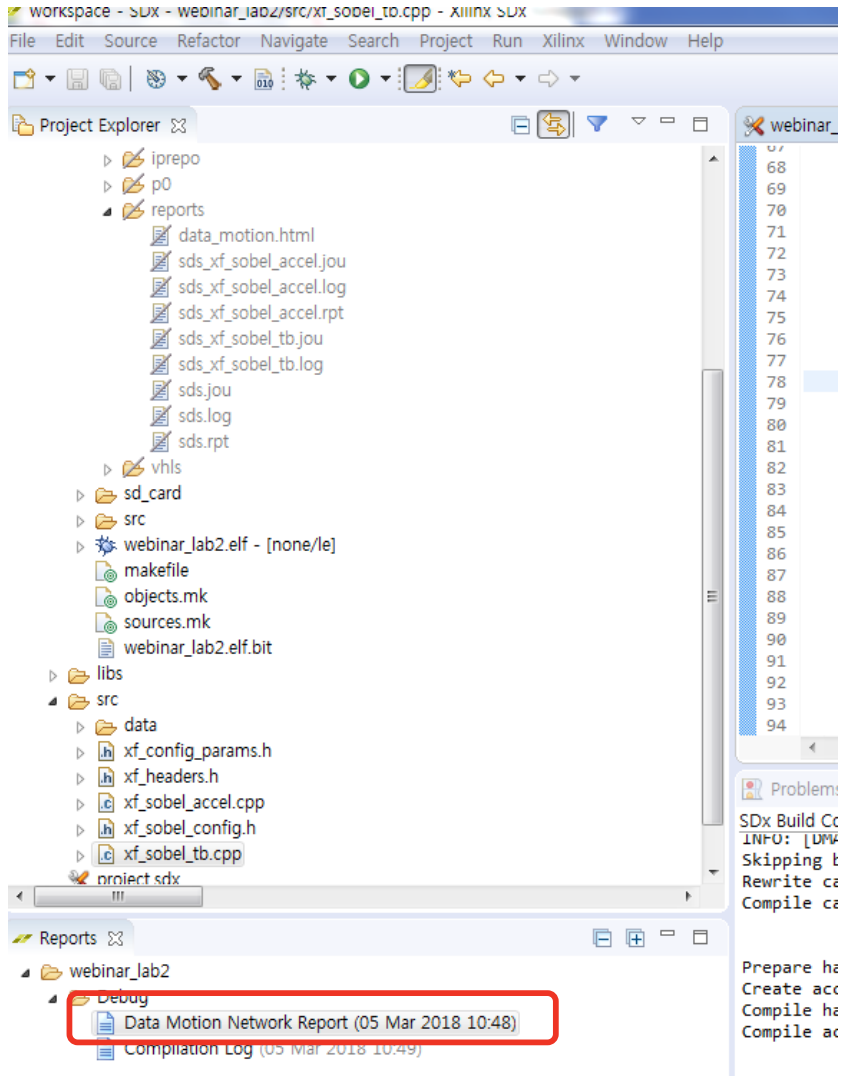


Y-gradient output



Erosion output

Data Motion Network



Data Motion Network

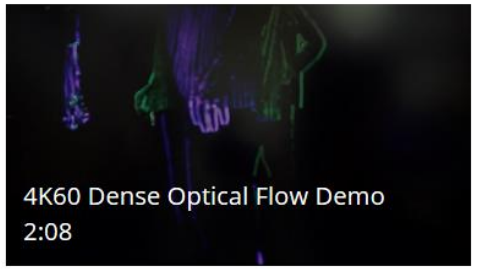
Accelerator	Argument	IP Port	Direction	Declared Size(bytes)	Pragmas	Connection
w0_xf_Sobel_1	_src_mat	p_src_mat_allocatedFlag	IN	1		axi_interconnect_hpm1_M08_AXI:AXILITE:0xC
		p_src_mat_rows	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x10
		p_src_mat_cols	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x14
		p_src_mat_size	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x18
		p_src_mat_data_V	IN	2073600*1	• length:(_src_mat.size) • mem_attribute:NON_CACHEABLE.PHYSICAL_CONTIGUOUS	zynq_ultra_ps_e_0_S_AXI_HP2_FPD:AXIDMA_SG
	_dst_matx	p_dst_matx_allocatedFlag	IN	1		axi_interconnect_hpm1_M08_AXI:AXILITE:0x1C
		p_dst_matx_rows	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x20
		p_dst_matx_cols	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x24
		p_dst_matx_size	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x28
		p_dst_matx_data_V	OUT	2073600*1	• length:(_dst_matx.size) • mem_attribute:NON_CACHEABLE.PHYSICAL_CONTIGUOUS	w1_xf_dilate_1:p_src_mat_data_V
	_dst_maty	p_dst_maty_allocatedFlag	IN	1		axi_interconnect_hpm1_M08_AXI:AXILITE:0x2C
		p_dst_maty_rows	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x30
		p_dst_maty_cols	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x34
		p_dst_maty_size	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x38
		p_dst_maty_data_V	OUT	2073600*1	• length:(_dst_maty.size) • mem_attribute:NON_CACHEABLE.PHYSICAL_CONTIGUOUS	w2_xf_erode_1:p_src_mat_data_V
w1_xf_dilate_1	_src_mat	p_src_mat_allocatedFlag	IN	1		axi_interconnect_hpm1_M08_AXI:AXILITE:0xC
		p_src_mat_rows	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x10
		p_src_mat_cols	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x14
		p_src_mat_size	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x18
		p_src_mat_data_V	IN	2073600*1	• length:(_src_mat.size) • mem_attribute:NON_CACHEABLE.PHYSICAL_CONTIGUOUS	w0_xf_Sobel_1:p_dst_matx_data_V
	_dst_mat	p_dst_mat_allocatedFlag	IN	1		axi_interconnect_hpm1_M08_AXI:AXILITE:0x1C
		p_dst_mat_rows	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x20
		p_dst_mat_cols	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x24
		p_dst_mat_size	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x28
		p_dst_mat_data_V	OUT	2073600*1	• length:(_dst_mat.size) • mem_attribute:NON_CACHEABLE.PHYSICAL_CONTIGUOUS	zynq_ultra_ps_e_0_S_AXI_HP0_FPD:AXIDMA_SG
w2_xf_erode_1	_src_mat	p_src_mat_allocatedFlag	IN	1		axi_interconnect_hpm1_M08_AXI:AXILITE:0xC
		p_src_mat_rows	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x10
		p_src_mat_cols	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x14
		p_src_mat_size	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x18
		p_src_mat_data_V	IN	2073600*1	• length:(_src_mat.size) • mem_attribute:NON_CACHEABLE.PHYSICAL_CONTIGUOUS	w0_xf_Sobel_1:p_dst_maty_data_V
	_dst_mat	p_dst_mat_allocatedFlag	IN	1		axi_interconnect_hpm1_M08_AXI:AXILITE:0x1C
		p_dst_mat_rows	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x20
		p_dst_mat_cols	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x24
		p_dst_mat_size	IN	4		axi_interconnect_hpm1_M08_AXI:AXILITE:0x28
		p_dst_mat_data_V	OUT	2073600*1	• length:(_dst_mat.size) • mem_attribute:NON_CACHEABLE.PHYSICAL_CONTIGUOUS	zynq_ultra_ps_e_0_S_AXI_HP2_FPD:AXIDMA_SG

Summary

Summary

- Zynq SoCs offer superior performance and lower latency compared to other SoC offerings
- reVISION stack on SDSoC introduces familiar software environment with pre-optimized Opencv libraries
- Available NOW
- Visit the reVISION developer zone https://www.xilinx.com/products/design-tools/embedded-vision-zone.html

Featured Videos



View More Videos

Q&A