RECRLab
Reconfigurable Computing Research Laboratory

Fly-Inspired Edge Detection: Architecture and Reconfigurable Embedded Implementation

Oluwakemi Adabonyan, Daniel Llamocca, and Brian Dean

Electrical and Computer Engineering Department,

Oakland University

August 8th, 2018

OAKLAND UNIVERSITY

Outline

- *Introduction*
- *Fly-Inspired Edge Detection*
- *Fixed-Point Hardware Architecture*
- *Run-Time Reconfigurable Embedded System*
- *Results*

Introduction

- ***The common housefly*** *can perform complicated image processing tasks such as object recognition, motion detection, obstacle tracking and avoidance.*

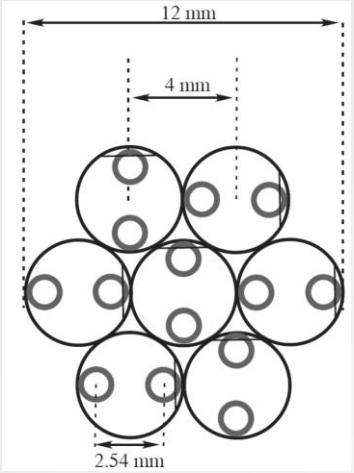
- ***Fly-Inspired edge detection*** *can outperform or supplement traditional image processing systems: The integration of the unique abilities of the fly-inspired* ***optical front-end*** *with existing imaging technologies can produce efficient machine vision systems ranging from unmanned aerial and ground vehicles to structural monitoring.*

- ***Large data processing requirements:*** *Dedicated hardware implementations are desired. To optimize resources, we use fixed-point arithmetic.*

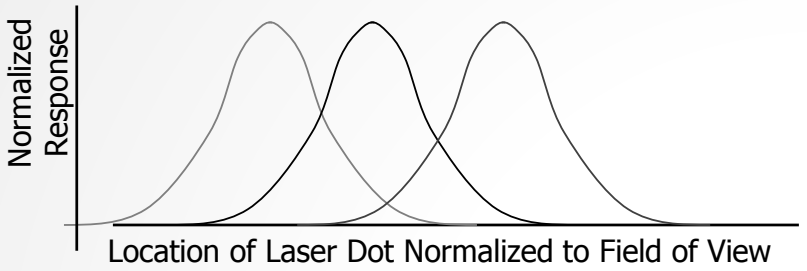
Fly-Inspired Edge Detection

Vision Sensor *(Optical Front-end). Plano-convex lens (12 mm diameter and 12 mm focal length) placed above 7 photodiodes arranged in a hexagonal pattern.*

- *Hexagonal pattern approximates fly-eye optical arrangement.*
- *A single photoreceptor response to a sweeping laser beam resembles a Gaussian signal. Photoreceptors' response to an object in their Field of View (FOV) is a set of overlapping Gaussians.*

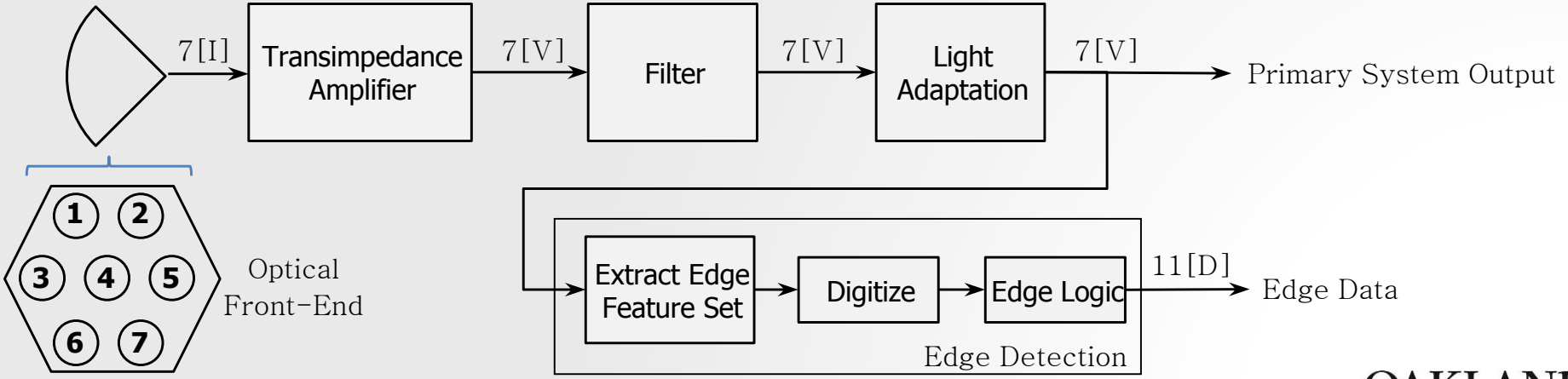


- ***Example****: Response from 3 photoreceptors to an object moving across their FOV. Overlap represents area in space where different photoreceptors FOV overlap. Motion is detected when 2 neighboring photoreceptors detect a concurrent change in response.*



Fly-Inspired Edge Detection

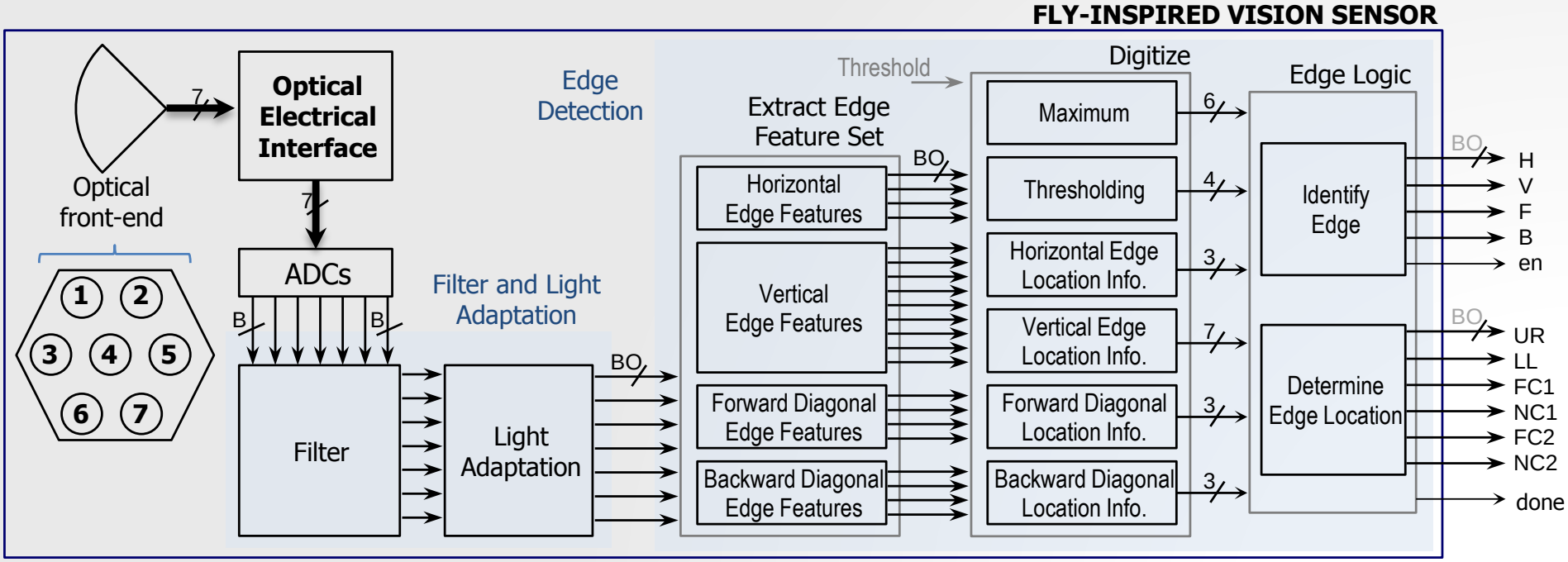
- ***Analog System:*** *Original system fully tested*. Our work (fully digital system) is based on this system.*
- ***Components:***
  - *Biomimetic Front:* 7 photo elements in a hexagonal pattern.
  - *Trans impedance:* Converts current to voltage (7 signals) to match the environment.
  - *Filter:* Removes interference due to indoor lightning.
  - *Light Adaptation:* Removes effects caused by ambient light.
  - *Edge Detection:* Produces 11 binary signals with information about the presence, orientation, and location of an edge.



* B. K. Dean, *"Light Adaptation and Applications for a Fly Eye Vision Sensor"*, Ph.D. Dissertation, University of Wyoming, 2012.

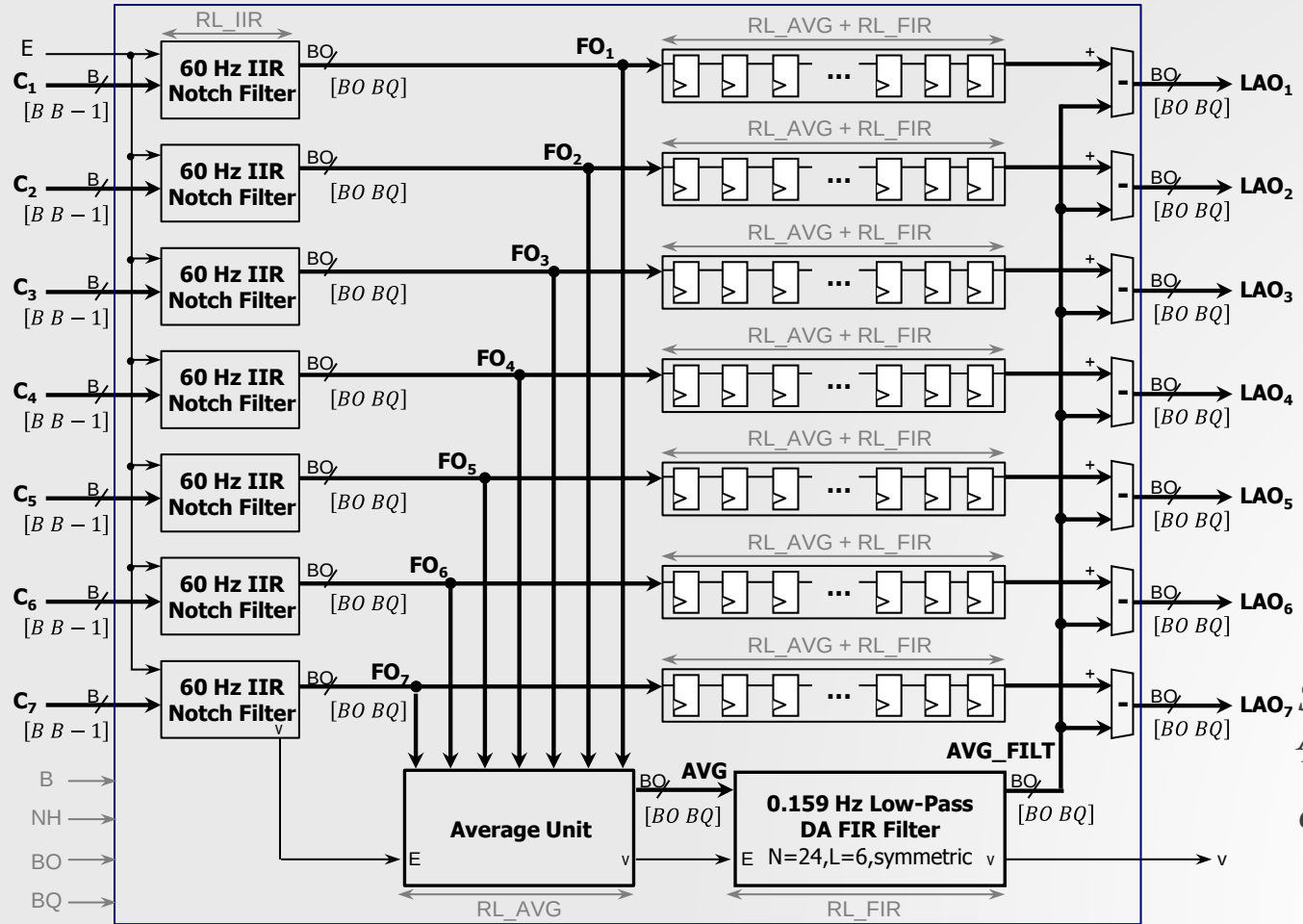
Hardware Architecture

- ***Digital System for Fly-Inspired Edge Detection***
  - *Optical Electrical Interface: Trans-impedance amplifier. By using ADCs, the 7 analog outputs are converted into digital signals.*
  - *Filter and Light Adaptation Stage: Fully Pipelined and Scalable Hardware Implementation.*
  - *Edge Detection: It extracts information from the scene for edge detection: orientation and location.*



Hardware Architecture

- ***Filter and Light Adaptation***: *Fully-pipelined, parameterized VHDL architecture* that uses fixed-point arithmetic:*
  - *Input: [B B-1]. Output/intermediate signals: [BO BQ].*
  - *Design Parameters: B,BO,BQ,NH (# of bits of filters' coefficients)*



Architecture:
7 IIR Filters,
1 average unit,
1 FIR filter,
7 subtractors,
7 chain registers (synchronization registers for pipelining)

* D. Llamocca, B. K. Dean, *"A Scalable Pipelined Architecture for Biomimetic Vision Sensors", in FPL'2015 conference.*

OAKLAND UNIVERSITY

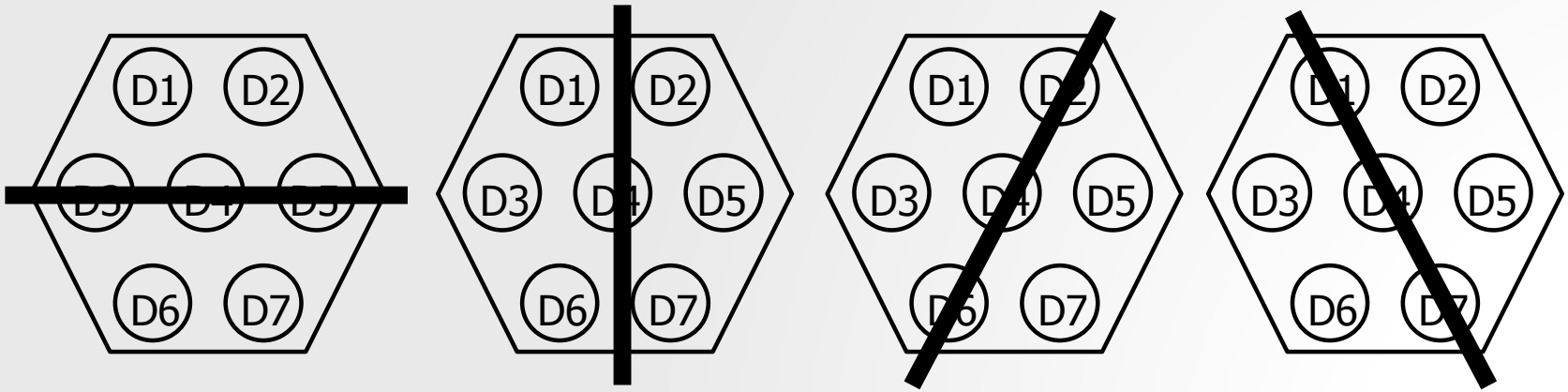
Fly-Inspired Edge Detection

Fly-Inspired Edge Detection: After the sensor outputs are passed through the Filter and Light Adaptation circuit, we can identify the orientation of an edge and determine its location.

- **Edge Orientation:**
  - Based on Madsen's algorithm. 4 orientations of interest. The equations yield the magnitude of the intensity of each orientation.

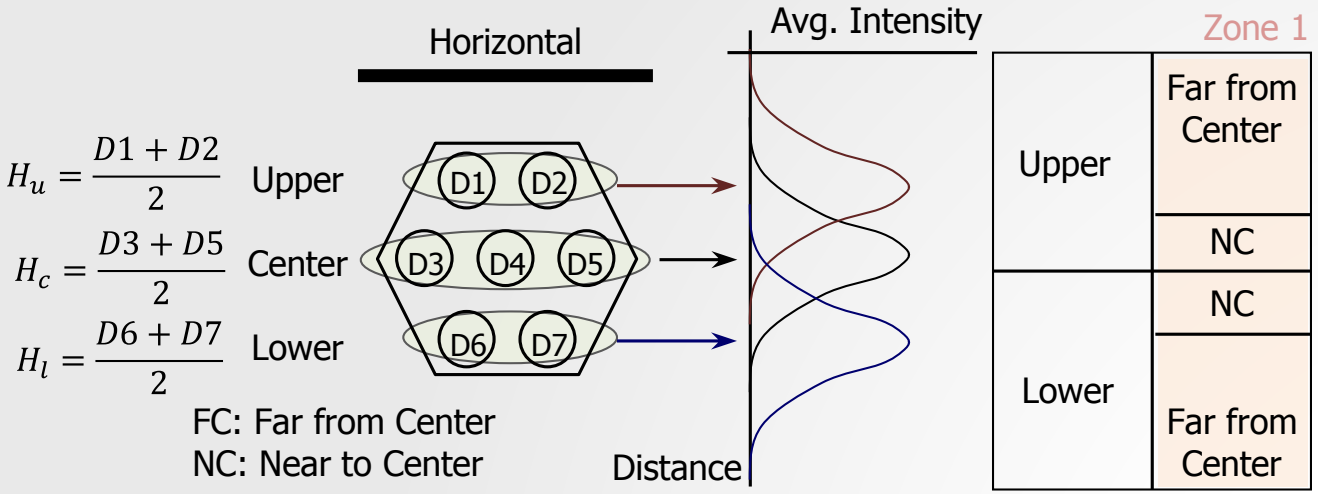
$$|H_d| = \left|\frac{D1+D2}{2} - \frac{D6+D7}{2}\right|, |V_d| = \left|\frac{D1+D3+D6}{2} - \frac{D2+D5+D7}{2}\right|$$

$$|F_d| = \left|\frac{D1+D3}{2} - \frac{D5D7}{2}\right|, |B_d| = \left|\frac{D2+D5}{2} - \frac{D3+D6}{2}\right|$$



Fly-inspired Edge Detection

- **Edge Location**:
  - For Horizontal Edge Location: 3 average light intensities are computed: upper (H_u), center (H_c), and lower (H_l). Plotted together, they produce an overlapping Gaussian profile. By carefully comparing each curve to the others, the Field of View (FOV) can be divided into 4 sections and the sensor can identify which section the edge is within: Upper FC, Lower FC, Upper NC, Lower NC.



  - Similarly, for the Forward Diagonal Edge and Back Diagonal Edge, we can divide the FOV into 4 sections.
    Vertical Edge: Besides Zone 1 for FC, NC, a more refined Zone 2 for FC, NC exists → FOV is divided into 8 sections.

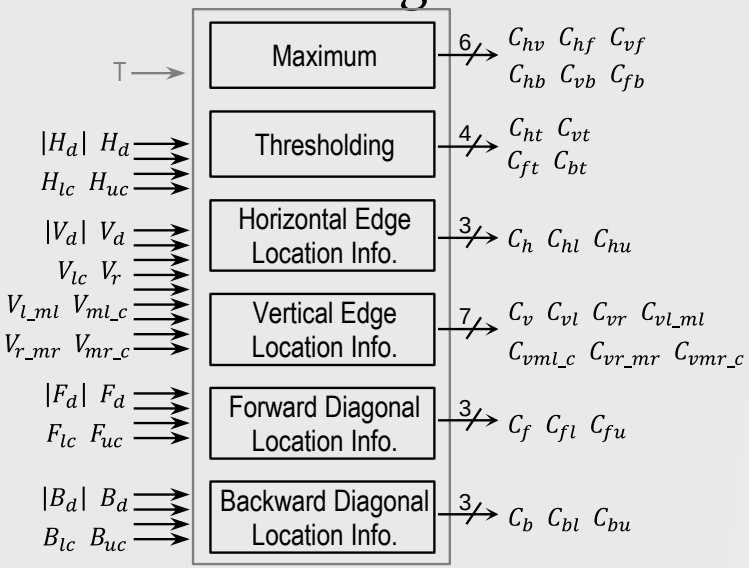
Hardware Architecture

- **Edge Detection:** *Extract Edge Feature Set*
  - Average light intensities are computed for each edge:
    - Horizontal Edge: H_u, H_l, H_c.
    - Forward Diagonal Edge: F_d, F_l, H_c. Backward Diagonal Edge: B_d, B_l, B_c.
    - Vertical Edge: left: V_l, middle left: V_{ml}, center: V_c, middle right: V_{mr}, right: V_r.
  - Horizontal Edge: Location is refined by computing:
    - $H_{ul} = H_u - H_l$. If $H_{ul} < 0$, edge above the center point of the sensor' FOV.
    - $H_{uc} = H_u - H_c$. If $H_{uc} < 0$, edge above center point and far from center point.
    - $H_{lc} = H_l - H_c$. If $H_{lc} < 0$, edge below center point and far from center point.
  - Similar features are computed for the other edges, this results in:
    - 4 horizontal edge features: H_d=H_{ul}, H_{lc}, H_{uc}, $|H_d|$
    - 4 forward diagonal edge features: F_d, F_{lc}, F_{uc}, $|F_d|$
    - 4 backward diagonal edge features: B_d, B_{lc}, B_{uc}, $|B_d|$
    - 8 vertical edge features: V_d, V_{lc}, V_r, V_{l_ml}, V_{ml_c}, V_{r_mr}, V_{mr_c}, $|V_d|$

    4 edge orientation and 16 edge location signals
  - *Hardware*: The average light intensities and the derived signals are implemented using simple arithmetic units: addition, subtraction, absolute value.

Hardware Architecture

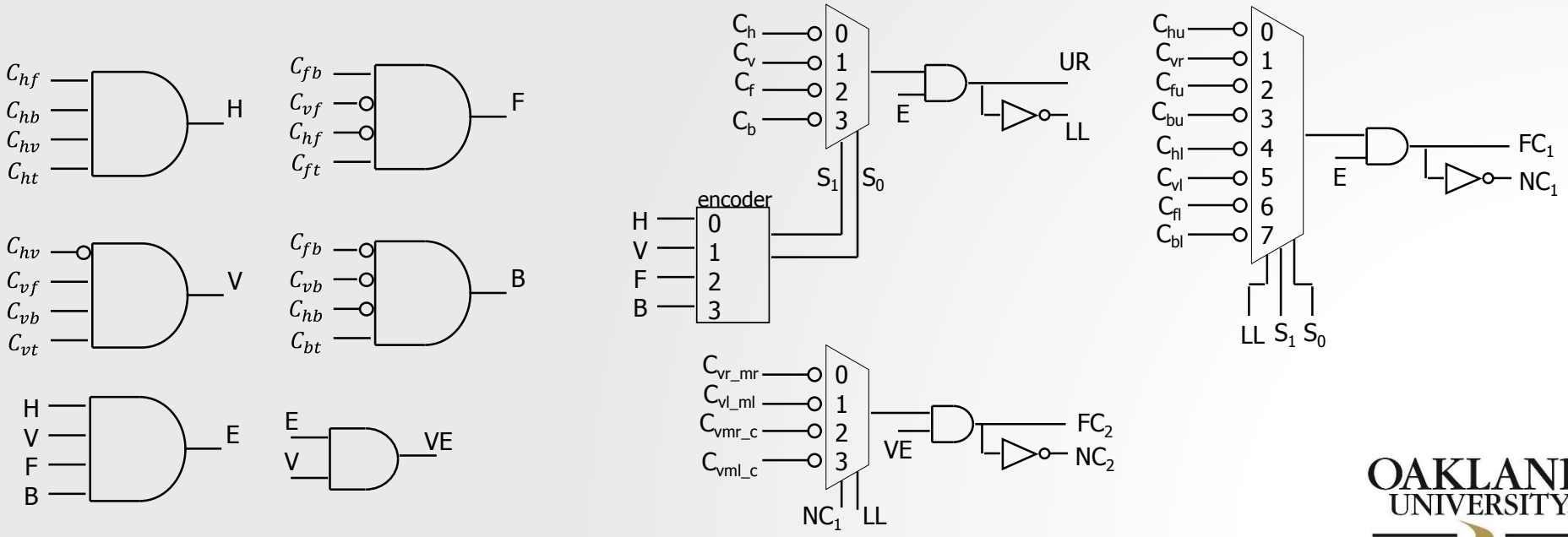
- **Edge Detection:** *Digitize*
  - It performs comparisons to produce 26 binary values for edge orientation and location. Inputs: orientation signals ($|H_d|, |V_d|, |F_d|, |B_d|$) and location signals (the other 16 signals).
    - Max.: Compares orientation signals against each other, it generates 6 signals.
    - Thresholding: sets the threshold T for detecting an edge.
    - Edge Location Information: Compares the 16 edge location signals with 0.
  - Hardware: Set of signed comparators with a constant (T or 0) and with a signal.



MAXIMUM		THRESHOLD		EDGE LOCATION INFO			
Comparison	Output	Comparison	Output	Comparison	Output	Comparison	Output
$\|H_d\| > \|V_d\|$	$C_{hv} = 1$	$\|H_d\| > T$	$C_{ht} = 1$	Horizontal: $H_d > 0$	$C_h = 1$	$V_d > 0$	$C_h = 1$
$\|H_d\| > \|F_d\|$	$C_{hf} = 1$	$\|V_d\| > T$	$C_{vt} = 1$	$H_{lc} > 0$	$C_{hl} = 1$	$V_{lc} > 0$	$C_{vl} = 1$
$\|V_d\| > \|F_d\|$	$C_{vf} = 1$	$\|F_d\| > T$	$C_{ft} = 1$	$H_{uc} > 0$	$C_{hu} = 1$	$V_{rc} > 0$	$C_{vr} = 1$
$\|H_d\| > \|B_d\|$	$C_{hb} = 1$	$\|B_d\| > T$	$C_{bt} = 1$	$F_d > 0$	$C_f = 1$	$V_{l\_ml} > 0$	$C_{vl\_ml} = 1$
$\|V_d\| > \|B_d\|$	$C_{vb} = 1$			Forward Diagonal: $F_{lc} > 0$	$C_{fl} = 1$	$V_{ml\_c} > 0$	$C_{vml\_c} = 1$
$\|F_d\| > \|B_d\|$	$C_{fb} = 1$			$F_{uc} > 0$	$C_{fu} = 1$	$V_{r\_mr} > 0$	$C_{vr\_mr} = 1$
				$B_d > 0$	$C_b = 1$	$V_{mr\_c} > 0$	$C_{vmr\_c} = 1$
				Backward Diagonal: $B_{lc} > 0$	$C_{bl} = 1$	Vertical	
				$B_{uc} > 0$	$C_{bu} = 1$		

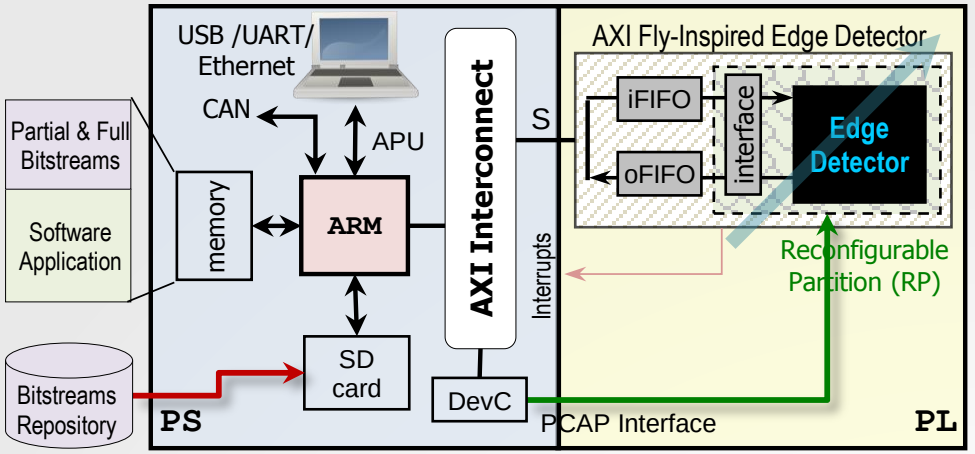
Hardware Architecture

- **Edge Detection:** *Edge Logic (11 output bits)*
  - It determines: edge presence (E), orientation (H,V,F,B), and location (UR, FC1, FC2). Far from Center: Zone 1 (H,V,F,B). Zone 2 (only V).
  - UR: Upper/Right. It determines whether an edge is on the upper or lower half. Vertical edge: right or left half. $LL = \overline{UR}$: Lower/Left.
  - Location can be refined:
    - FC1: Far from Center – Zone 1. NC1: Near from Center – Zone 1 = $\overline{FC_1}$
    - Only for Vertical Edge: FC2: Far from Center – Zone 2, $NC_2 = \overline{FC_2}$.



Embedded System

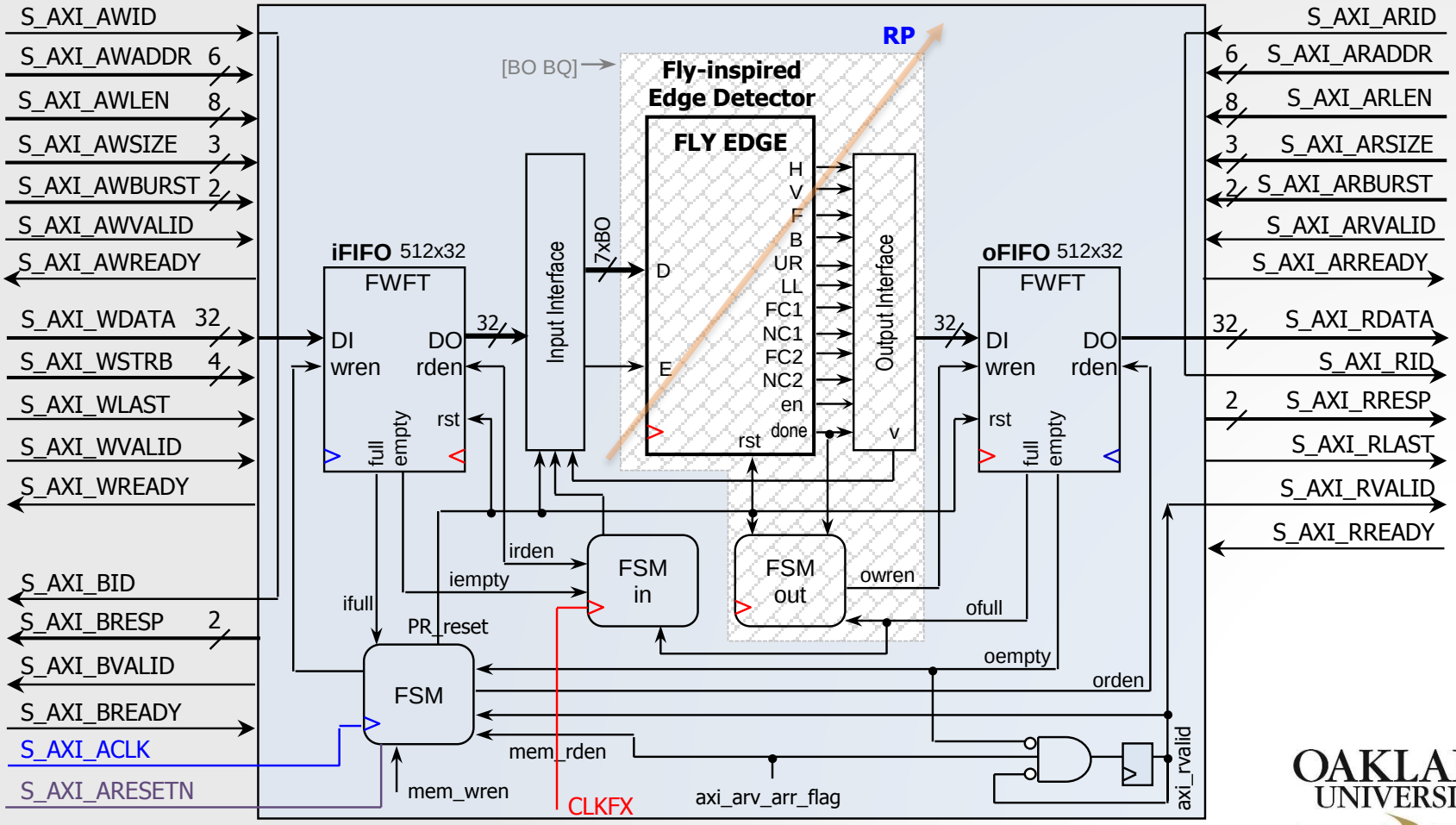
- For real-time hardware validation, the Edge Detector was placed in a Run-Time Reconfigurable embedded system.
- It was implemented on a Programmable System-on-Chip that integrates:
  - Processing System (PS): dual-core ARM® processor and common peripherals
  - Programmable Logic (PL): run-time reconfigurable fabric (FPGA).
- The PS feeds data to/extracts data from our Edge Detector via an AXI4-Full Interface.



- Target device: Xilinx® XC7020 Zynq-7000 All Programmable SoC.
- It was tested on a ZED Development Board.
- AXI Pressure Estimator Peripheral: It includes our design. It is located in the PL, and it runs at 50 MHz.

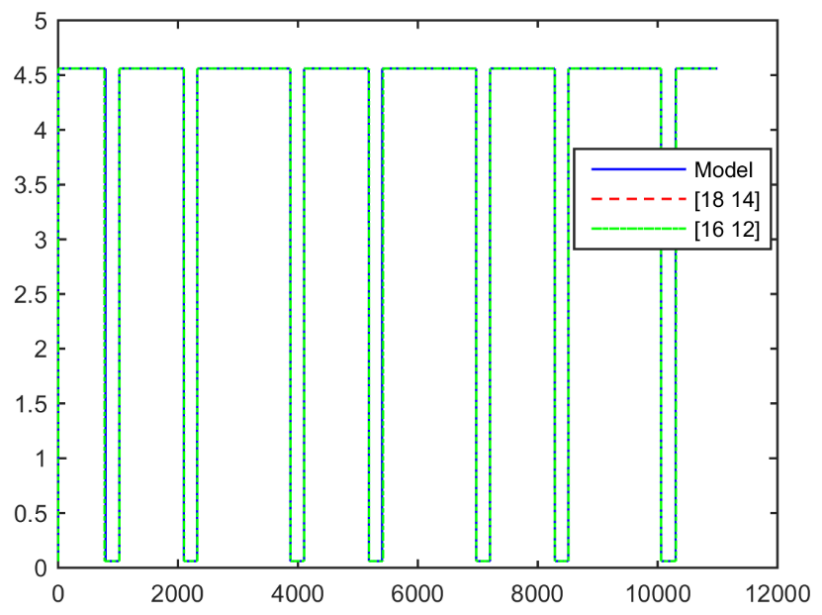
Embedded System

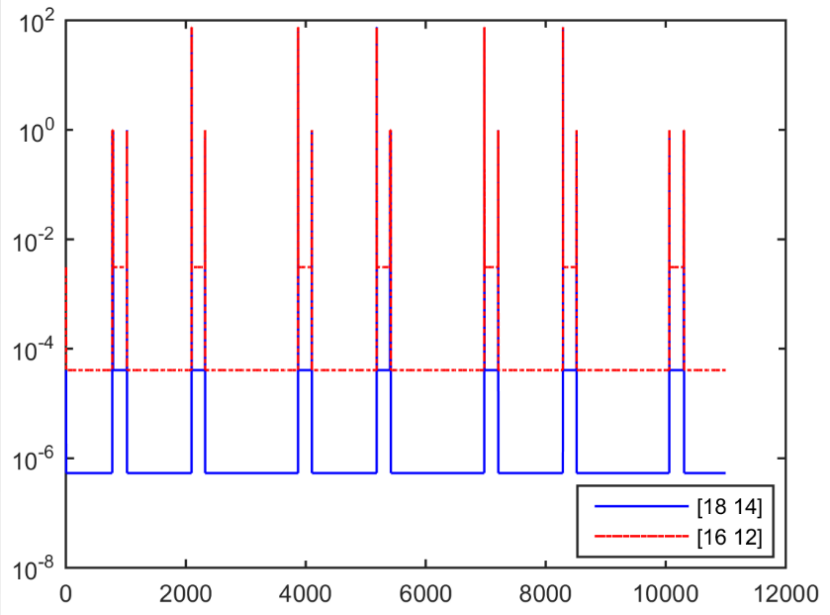
- Fly-Inspired Edge Detector Peripheral: our design + a 32-bit AXI4-Full Slave Interface (2 FIFOs, control, and extra logic).
- With this configuration, we feed the data sets and then retrieve the results via the AXI4-Full Interface.



Results

- ***Experimental Setup***: 11000 sets of seven samples
  - 2 Hardware configurations: Fixed-point formats [18 14] and [16 12].
  - To illustrate hardware accuracy, we multiplied the binary values by a constant and added an offset to emulate analog values. Hardware verification: via hardware simulation and real-time embedded testing. *Metric used*: relative error: (HW–MATLAB)/MATLAB



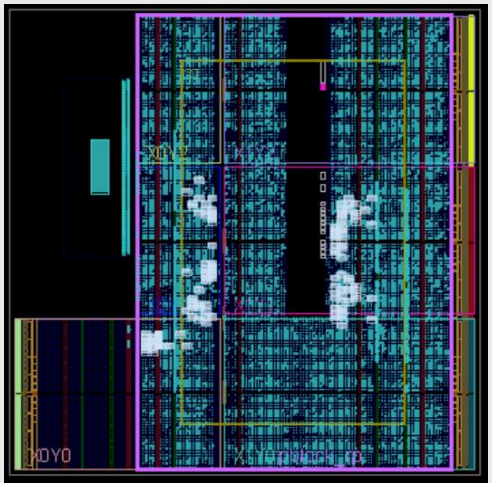


- We show results for Horizontal Edge Detector (Bit H). Note that the curves are pretty close. This occurs for all the bits.

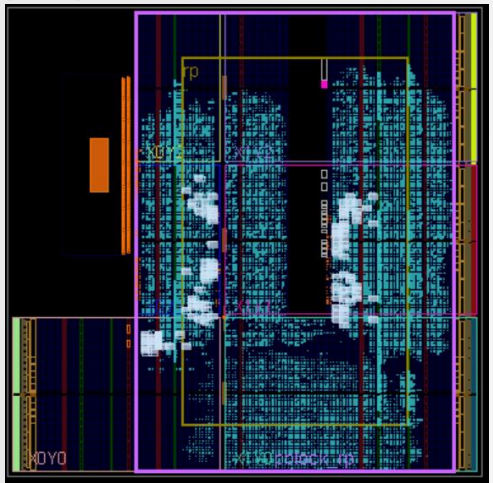
Embedded System

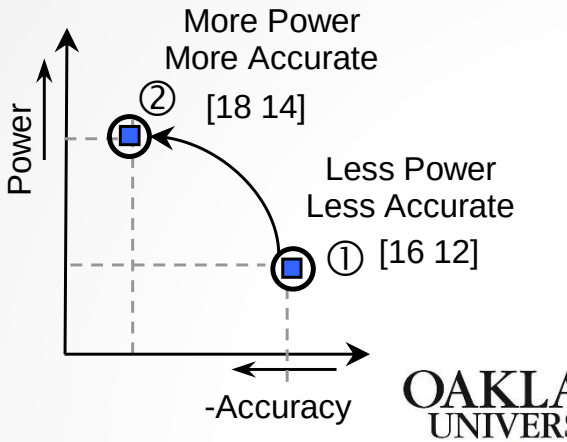
- Hardware resources: (only hardware core)
  - [18 14] design: 19264 LUTs, 48393 FFs, 79 DSPs.
  - [16 12] design: 16157 LUTs, 32987 FFs, 79 DSPs.
- Reconfigurable Partition (RP):
  - RP size: 88×150 Slices (device: Zynq-7000). Partial bitstream size: 3285 KB.
  - By loading partial bitstreams, the designs [18 14] and [16 12] can be swapped at run-time in order to adjust accuracy and power consumption.
    - [16 12] design: 36% of the LUTs and FFs available in the RP
    - [18 14] design: 52% of the LUTs and 58% of FFs available in the RP.
  - Reconfiguration speed (PCAP): 128 MB/s. Reconfiguration Time: 25.66 ms.

[18 14]



[16 12]





Embedded System

Execution Time:

- Hardware Processing cycles: 8 + 2(BO+14).
  Operating frequency: 50 MHz.
  - [18 14]: Output data computed in 72 cycles.
  - [16 12]: Output data computed in 68 cycles.

- Embedded System Processing Time: It includes reading the entire input data set (11000 samples), processing them, and storing the output results on the SD card.
  - [18 14]: It takes 33.7 ms.
  - [16 12]: It takes 30.1 ms.

Conclusions

- A hardware design was completed for fly-inspired edge detection. Verification was carried out via time-accurate simulation.

- The hardware design was placed in a reconfigurable embedded system. Real-time testing was carried out using *11000* sets of data. The system correctly detected the edges, their orientation and location at a fast enough rate to keep up with scene changes.

- The run-time alterable embedded system was successfully tested with 2 configurations that vary in their numerical representation. This allows for a trade-off between power and accuracy.