A Framework for Run-Time Reconfigurable Computing

DANIEL LLAMOCCA

Electrical and Computer Engineering Department,

Oakland University

October, 24th, 2014

Outline

RECRLab
Reconfigurable Computing Research Laboratory

- Motivation
- General Approach
- Implementation Details
- Application Examples:
  - Pixel Processor
  - 2D FIR Filter

- Research Plans
- Teaching Plans
- Conclusions

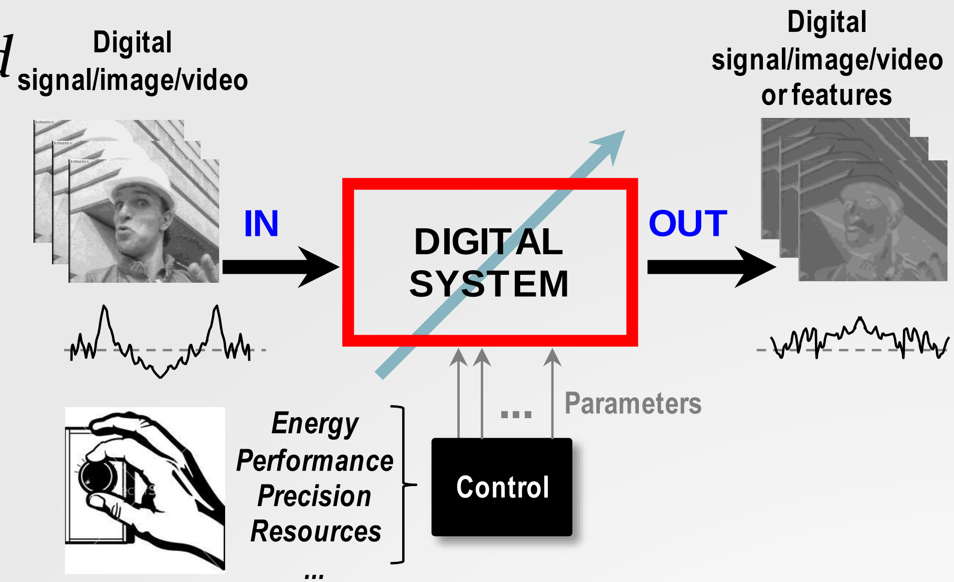
OAKLAND UNIVERSITY

Motivation

Digital systems can be characterized by a series of properties:

Energy, Performance, Precision, Resource Usage, Bandwidth, etc.

The controlling of these variables at run-time is defined as ***Dynamic Reconfigurable Computing Management****.*



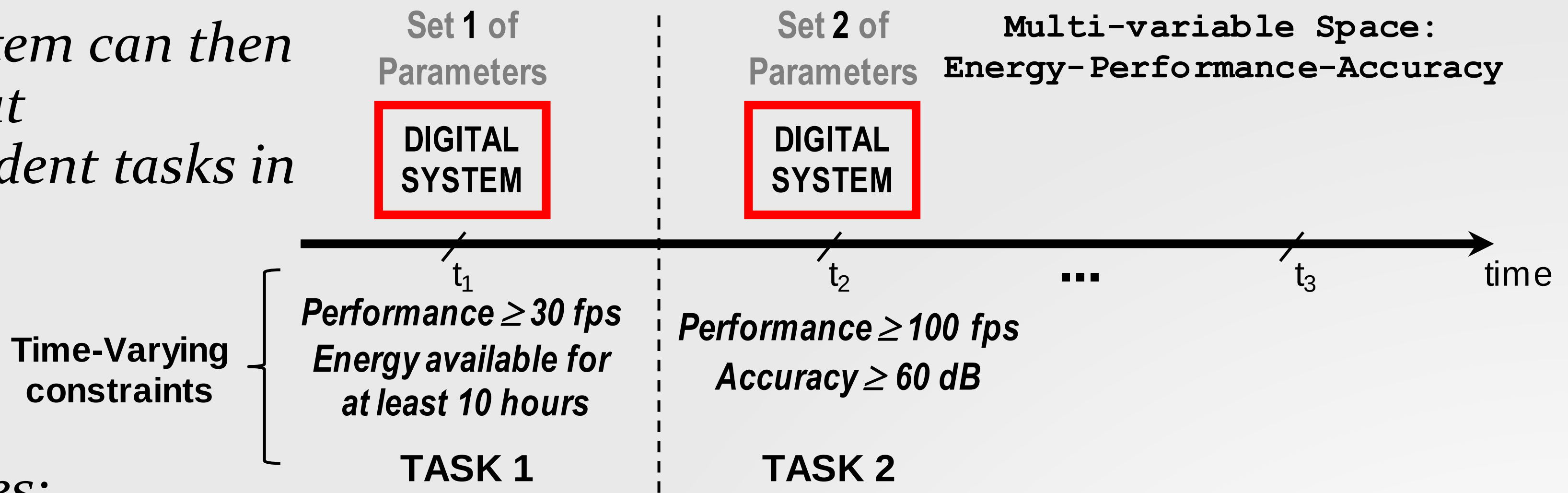
Dynamic Reconfigurable Computing Management *will enable us to deliver:*

- *A dynamically self-adaptive system (by dynamic allocation of computational resources and dynamic frequency control) that satisfies time-varying Multi-variable requirements (or constraints).*
- *Optimal hardware realizations: We want to investigate optimal solutions that can meet time-varying Multi-variable requirements .*

  For example, if the variables were Energy, Performance, and Accuracy, then the system should ***minimize energy consumption, and at the same time maximize performance and precision****, while satisfying the given multi-variable requirements.*

Motivation

The system can then carry out independent tasks in time:

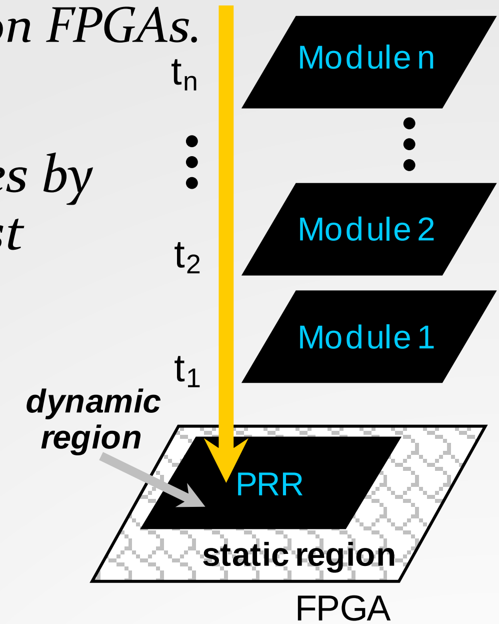


Examples:

- *Task 1: A video processing system is asked to deliver real time performance at 30 frames per second (fps) on limited battery life that will also need to operate for at least 10 hours. This is a* ***multi-objective optimization*** *problem. If solutions are found, pick the system realization that delivers the highest accuracy.*
- *Task 2: Now, suppose that we are asked to deliver performance at 100 frames per second (fps) at some minimum level of accuracy (60dB). In this case, we select the hardware realization with the lowest energy requirements while meeting the performance and accuracy constraints.*

Motivation

Dynamic Reconfigurable Computing Management can rely on: ***Dynamic Partial Reconfiguration*** *and* ***Dynamic Frequency Control*** *on FPGAs.*



Dynamic Partial Reconfiguration (DPR)

DPR technology enables the adaptation of hardware resources by modifying or switching off portions of the FPGA while the rest remains intact, continuing its operation.

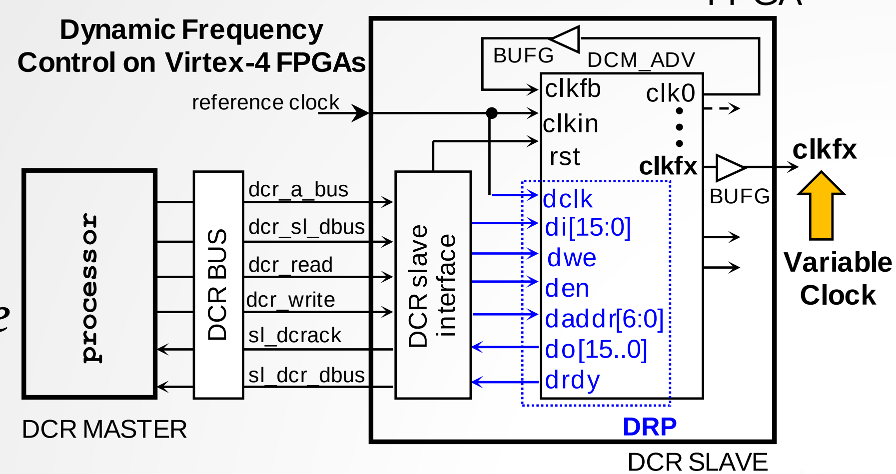
*A Partial Reconfiguration Region (**PRR**) is a region whose hardware configuration can be modified at run-time.*

Xilinx® devices: the PRR is dynamically reconfigured via the internal configuration access port (ICAP).

Dynamic Frequency Control

Digital Clock Managers (DCMs) inside FPGAs provide a wide range of clock management features.

The Dynamic Reconfiguration Port (DRP) of the DCM enables dynamic control of the frequency and phase. → We can dynamically adjust the frequency without reloading a new bitstream to the FPGA.



General approach (1/7)

For a given system, Dynamic Reconfigurable Computing Management is carried in the following manner:

1) *Definition of Objective Functions*
2) *Development of efficient cores*
3) *Parameterization of Hardware Cores*
4) *Multi-objective Pareto Optimization in the Multi-Variable Space*
5) *Dynamic management based on real-time multi-variable constraints*

General approach (2/7)

RECRLab Reconfigurable Computing Research Laboratory

1) ***Definition of objective functions:***

A wide range of quantities (e.g., energy, performance, precision, hardware usage) can considered as the objective functions of system parameters. These properties may have a slightly different definition depending on the application:

***Energy** can be measured as the total energy spent during the system operation, or the energy spent during an operation (e.g., energy per video frame). In some instances, measuring **Power** is more useful.*

***Performance** can be measured by: Megasamples per second, frames per second, Megabytes per second, etc.*

***Precision** can be measured by: numerical representation, or accuracy with respect to an idealized result (e.g., PSNR).*

*There can be objective functions that pertain to an specific application. For example, in image compression, the **bitrate** metric evaluates the compression efficiency of a hardware architecture (e.g. JPEG processor). Or **bandwidth** for communication networks.*

OAKLAND UNIVERSITY

General approach (3/7)

2) ***Development of efficient cores:*** *The hardware architectures should use techniques that:*

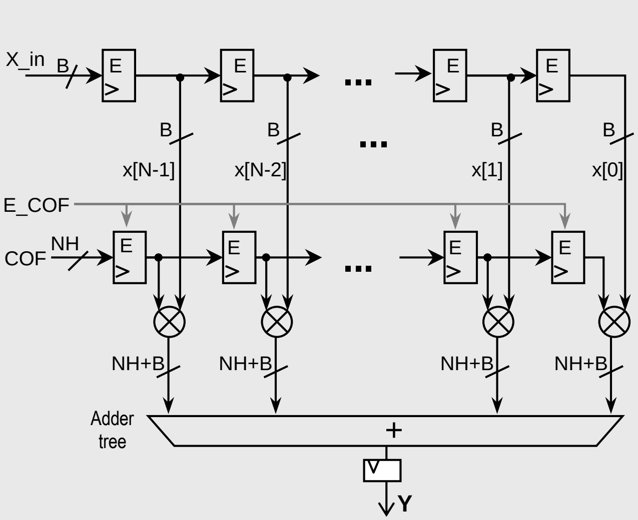
i) Minimize the amount of computational resources (e.g. LUT-based approaches, Distributed Arithmetic),

ii) Exploit parallelism and pipelining so as to obtain high performance architectures.

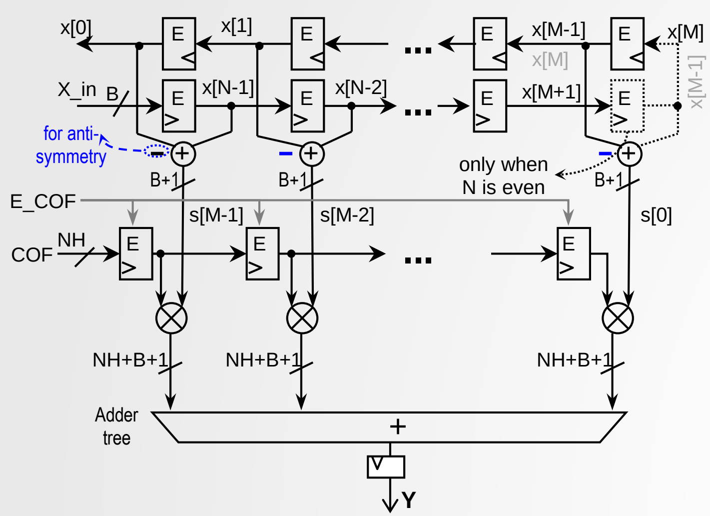
iii) Make intensive use of DPR and/or take advantage of DPR.

The cores should be described using Hardware Description Language (HDL) at the Register Transfer Level (RTL). The best effort must be made so that these cores remain portable across devices and vendors.

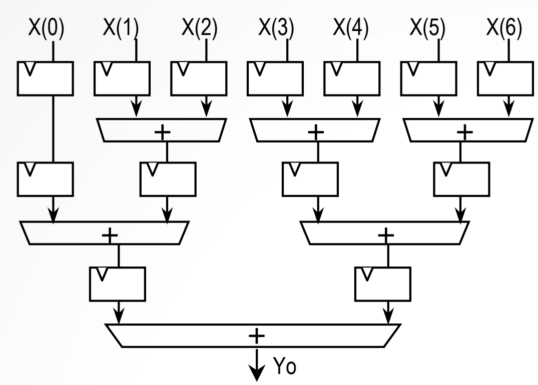
NON-SYMMETRIC FILTER



SYMMETRIC/ANTI-SYMMETRIC FILTER



PIPELINED ADDER TREE



General approach (4/7)

3) ***Parameterization of hardware cores:***

Fine control of the objective functions (e.g., energy, performance, accuracy) is greatly helped by realistic parameterization of the hardware cores (e.g., I/O bit-width, number of parallel cores).

Parameterized HDL code allows us to quickly create a set of hardware realizations by varying the parameters. Each realization comes with different values for the objective functions, which we ultimately control by varying the hardware parameters.

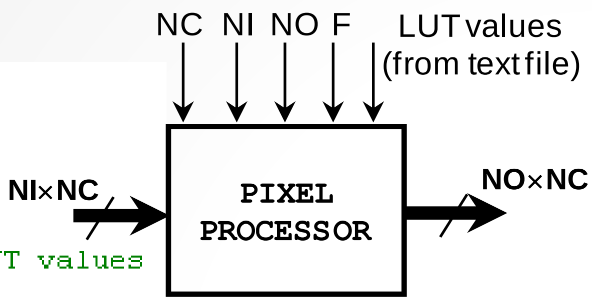
<u>Example</u>: Parameterization of the 'Pixel processor' architecture:

NC (number of cores), NI (number of input bits per pixel),

NO (number of output bits per pixel), F (function to be implemented),

LUT values (text file with LUT values)

```
entity pix_processor is
   generic ( NC: INTEGER:= 4; -- number of cores
             NI: INTEGER:= 8; -- number of input bits per pixel
             NO: INTEGER:= 8; -- number of output bits per pixel
             F: INTEGER:= 1; -- type of function (1..5)
             file_LUT: STRING:= "LUT_values.txt");  -- text file for LUT values
   port ( dyn_in: in std_logic_vector (NC*NI - 1 downto 0);
          dyn_out: out std_logic_vector (NC*NO - 1 downto 0));
end pix_processor;
```

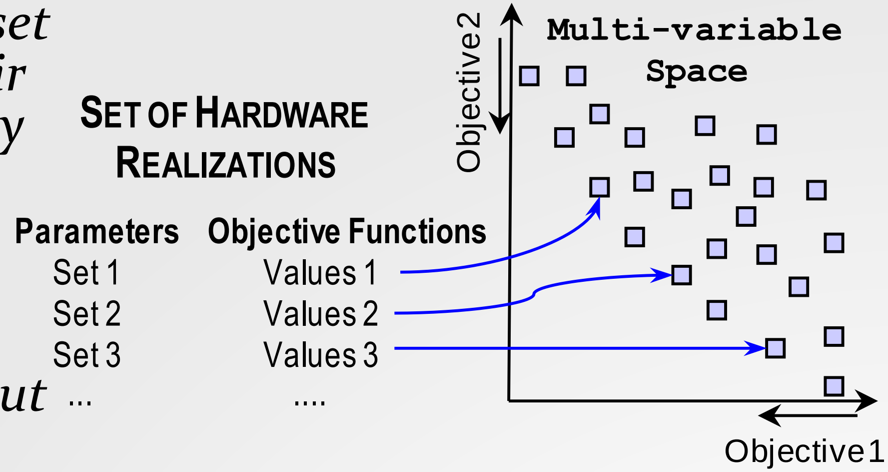


General approach (5/7)

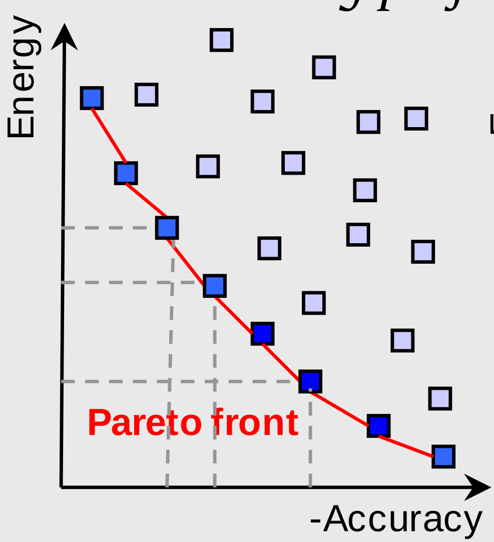
4) *Multi-objective Optimization in the Multi-Variable Space:*

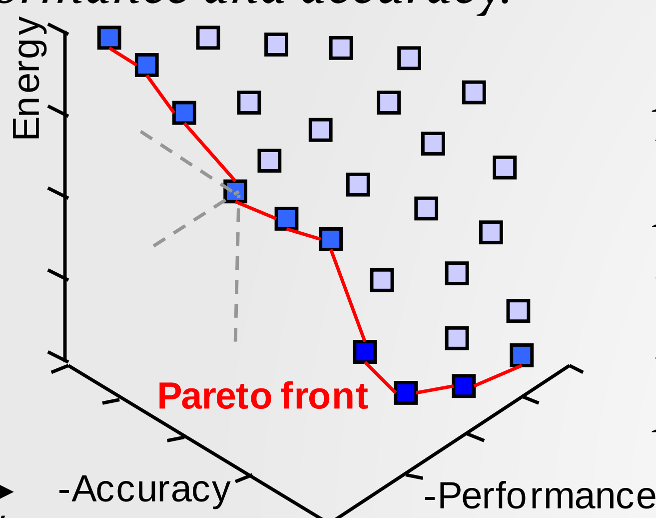
Multi-variable space: Represented by a set of hardware realizations along with their objective function values. We create it by varying the system parameters.

***Optimality**: A hardware realization is defined to be optimal in the multi-objective (Pareto) sense if it is not possible to improve on all criteria without deteriorating in at least one of them.*



<u>Example</u> 3-variable space: Energy-Performance-Accuracy Space: An optimal hardware realization is defined as one that minimizes energy, while maximizing performance and accuracy.





*The Energy-Performance-Accuracy space is shown along with the **Pareto- optimal** points.*

In some cases, we may want to explore a space of just 2 variables, e.g., the Energy-Accuracy space.

General approach (6/7)

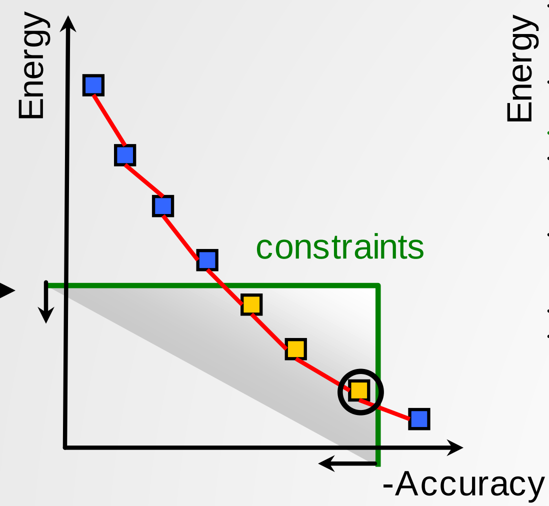
5) ***Dynamic management based on real-time multi-variable constraints:*** *Once the Pareto front has been extracted, we can cast optimization problems based on multi-variable constraints. We will use the Energy-Performance-Accuracy Space to explain this idea.*

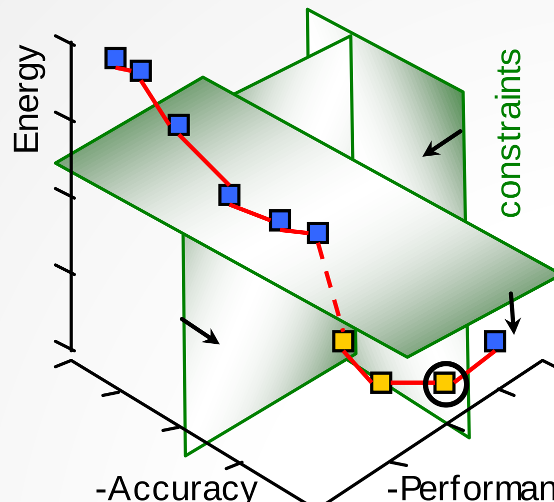
Example*: We are given constraints on the 3 variables. The feasible set is then represented by the* ***golden points****. We prioritize energy consumption, so we select the realization from the feasible set that also minimizes energy consumption. This can be cast as the following optimization problem:*

$$\underset{Ri}{minimize}\ Energy(Ri) \quad subject\ to: \quad \begin{matrix} Energy(Ri) \leq 2mJ \\ Accuracy(Ri) \geq 50dB \\ Performance(Ri) \geq 30fps \end{matrix}$$

Circled point: Realization from the feasible set that minimizes energy consumption.

If we ignore one variable (say, performance), we have a 2D optimization problem.

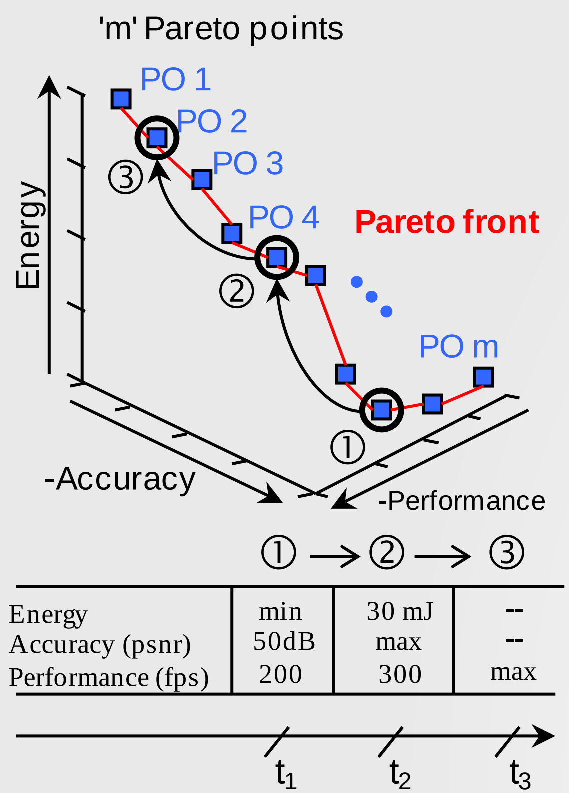




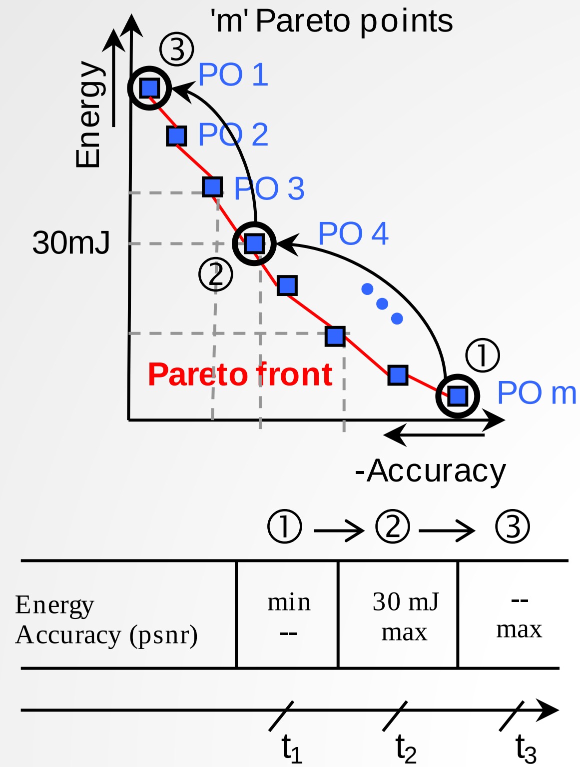
General approach (7/7)

RECRLab
Reconfigurable Computing Research Laboratory

5) ***Dynamic management based on real-time multi-variable constraints:*** *Now, we are given time-varying simultaneous constraints: different set of constraints are applied at different moments in time.*
The system receives stimuli in the form of multi-variable constraints and reconfigures itself via DPR and/or Dynamic Frequency Control to meet the multi-variable constraints. The figure shows examples with 3 and 2 simultaneous constraints.



Energy Accuracy (psnr) Performance (fps)	min 50dB 200	30 mJ max 300	-- -- max



Energy Accuracy (psnr)	min --	30 mJ max	-- max

OAKLAND UNIVERSITY

Implementation Details (1/2)

***Embedded FPGA system that supports Dynamic Partial Reconfiguration and Dynamic Frequency Control**:*

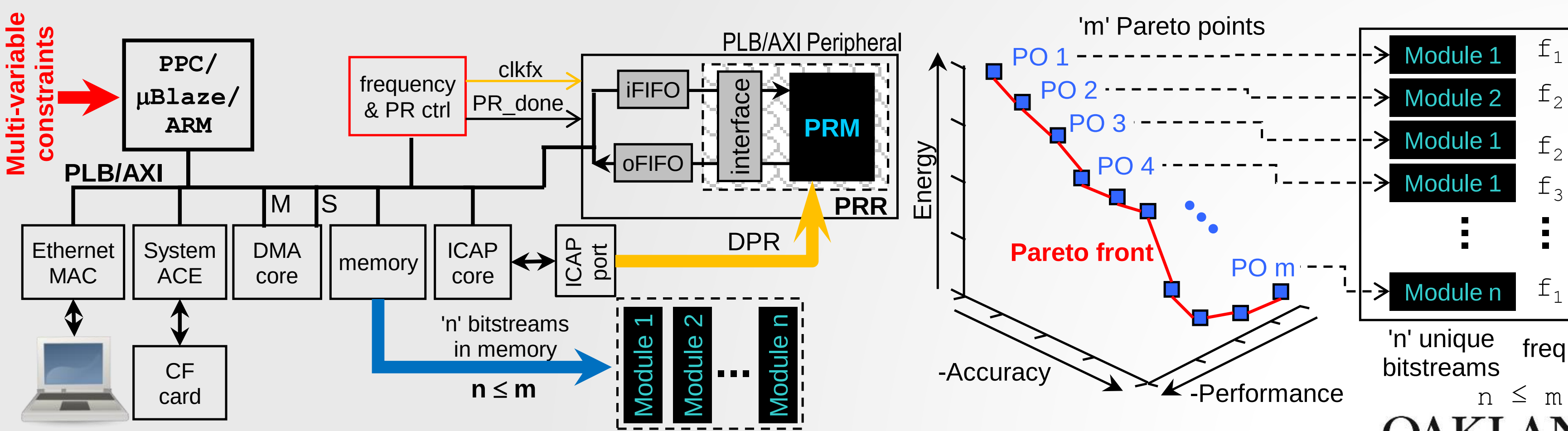
<u>Pareto-optimal point</u>: Represented by <bitstream, frequency of operation>

- Hardware realization that becomes active in the FPGA via Dynamic Partial Reconfiguration (DPR) and/or Dynamic Frequency Control.

If the system receives a multi-variable constraint:

- *It looks for a solution in the Pareto-optimal set: <bitstream*, freq*>*
- *It reconfigures the FPGA dynamic region(s) and /or frequency of operation, so as to meet the multi-variable constraints.*

***Example (one Dynamic Region)**: The PRM (Partial Reconfigurable Module) is a hardware core that performs an specific task and that can be modified at run-time.*



*PRR: Partial Reconfigurable Region

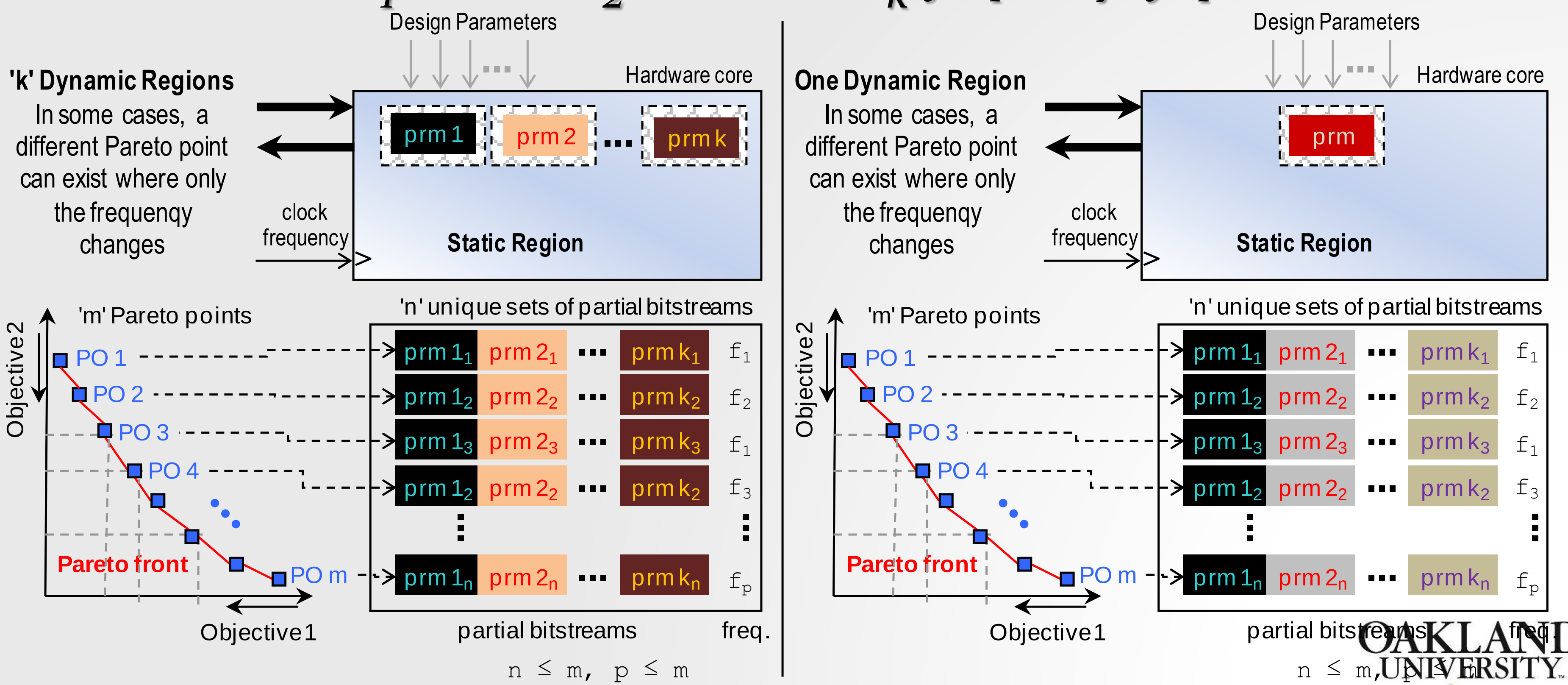
Implementation Details (2/2)

A Pareto point is represented by 'k' bistreams and the frequency of operation if:

- *The system has 'k' dynamic regions. The requested operation requires the dynamic regions to have a unique combination of bitstreams.*
- *The system has one dynamic region, but the requested operation requires reconfiguring the dynamic region 'k' times.*

Generalization: A Pareto point is represented by:

$$<bitstream_1, bitstream_2, \ldots, bitstream_k, \textit{frequency of operation}>$$



Digital signal, image, and video processing applications

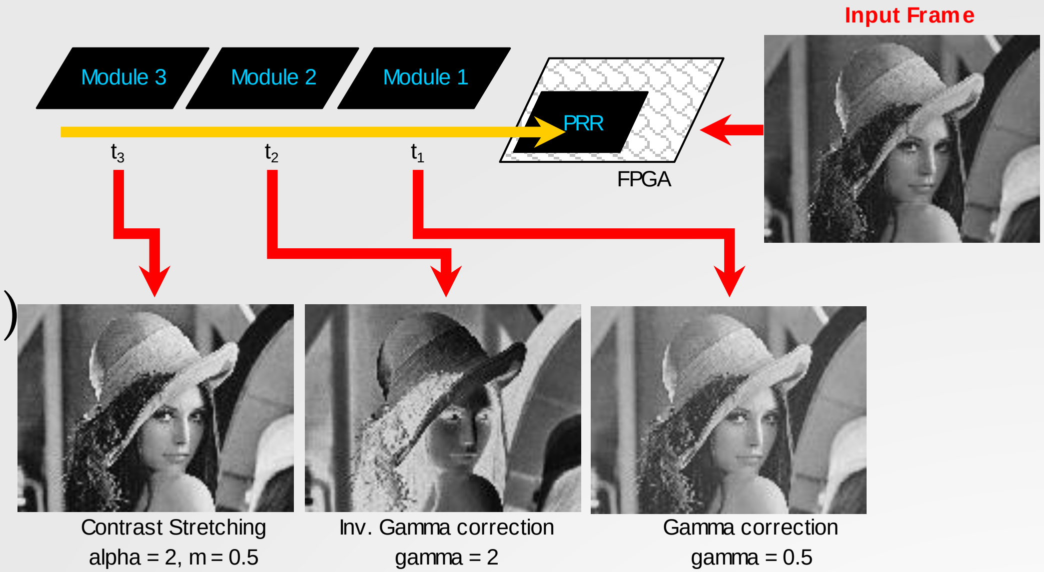
The following systems are discussed:

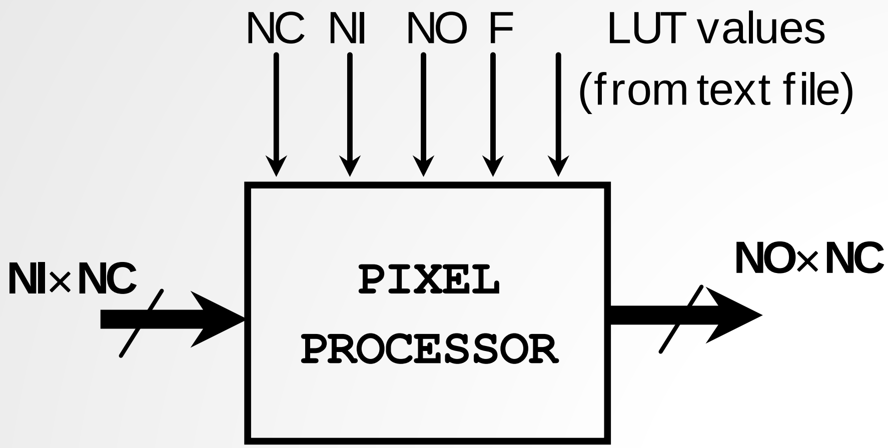
- *Pixel Processor and Dynamic PPA/EPA Management*

- *2D Separable FIR Filter Dynamic EPA Management.*

Pixel Processor (1/4)

- LUT-based architecture.
- Single-pixel operations (e.g., *gamma correction, Huffman encoding, histogram equalization, contrast stretching*) can be dynamically swapped. Parameter *F* modifies the function.
- In addition to dynamically modifying the input-output function, we allow for the modification of:
  - Input pixel bitwidth (*NI*)
  - Output pixel bitwidth (*NO*),
  - Number of parallel processing elements (*NC*)

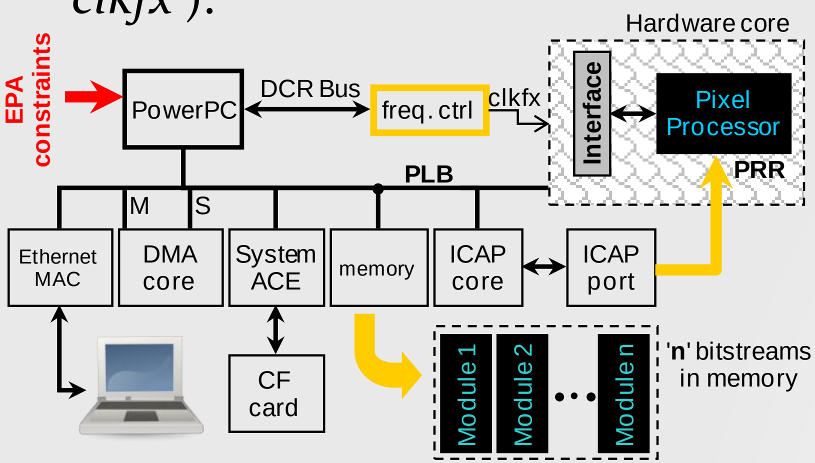




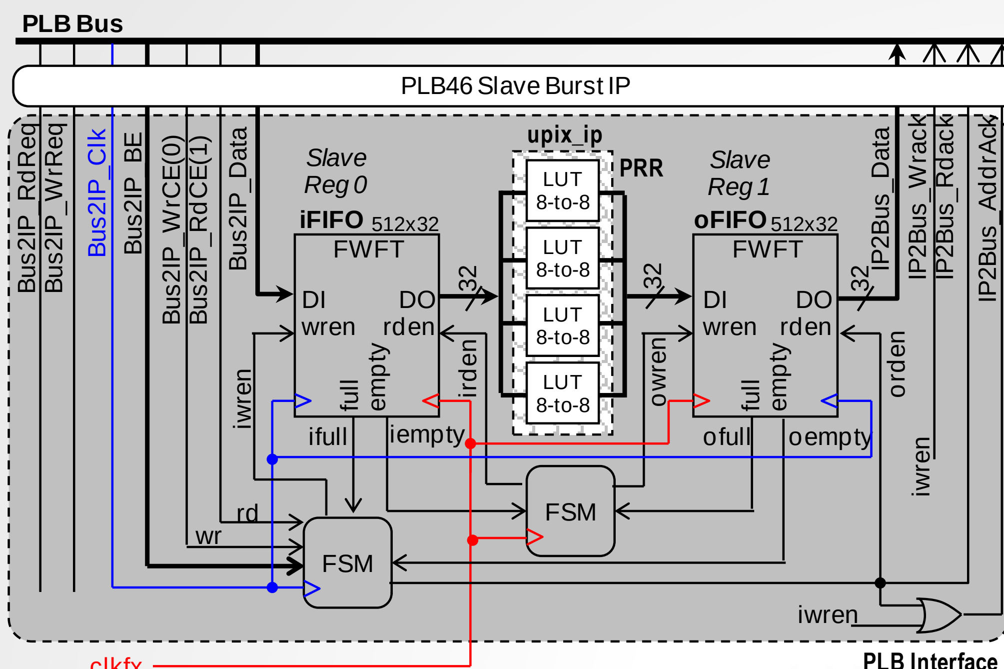
* VHDL IP core available at: dllamocca.org

Pixel Processor (2/4)

- **Embedded System**:
- One Dynamic Region. Pareto point represented by: <bitstream, frequency>.
- Pixel processor interface: PLB (Processor Local Bus) slave burst interface. The figure shows a PRR with NC=4, NI=NO=8.
- The system dynamically reconfigures: NC, NI, NO, FUNCTION, under the following constraints: NI×NC≤32, and NO×NC ≤ 32 (because of the 32-bit PLB)
- Five '*clkfx*' frequencies allowed: 100.00, 66.66, 50.0, 40.00, and 33.33 MHz.
- FIFOs required to properly isolate different clock regions (PLB clk= 100 MHz and '*clkfx*').



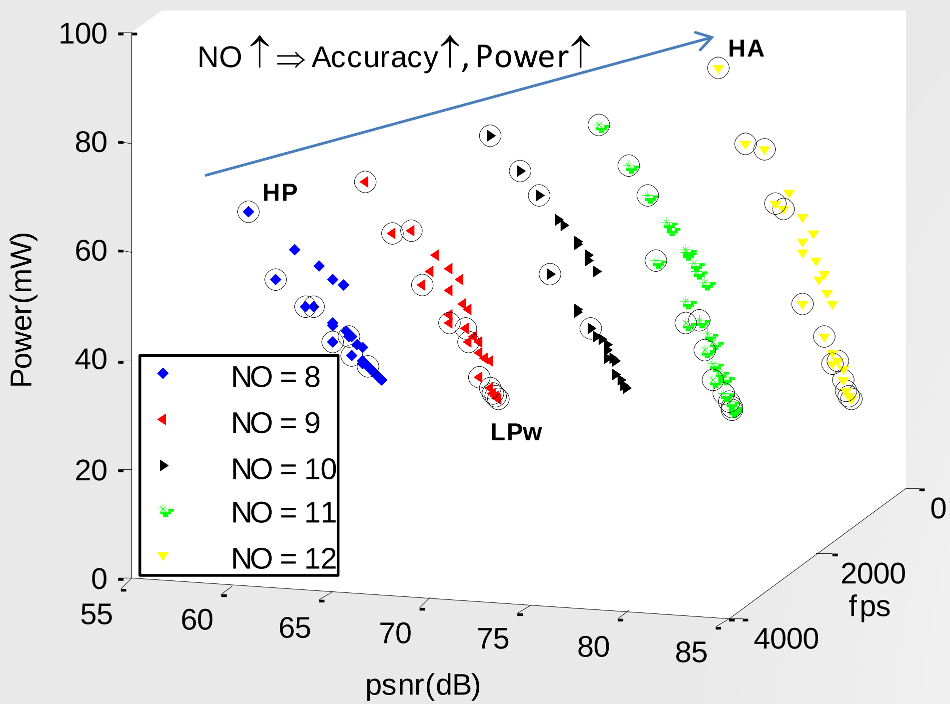
Implemented on **a ML405 Development Board**
Device: XC4VFX20-11FF672

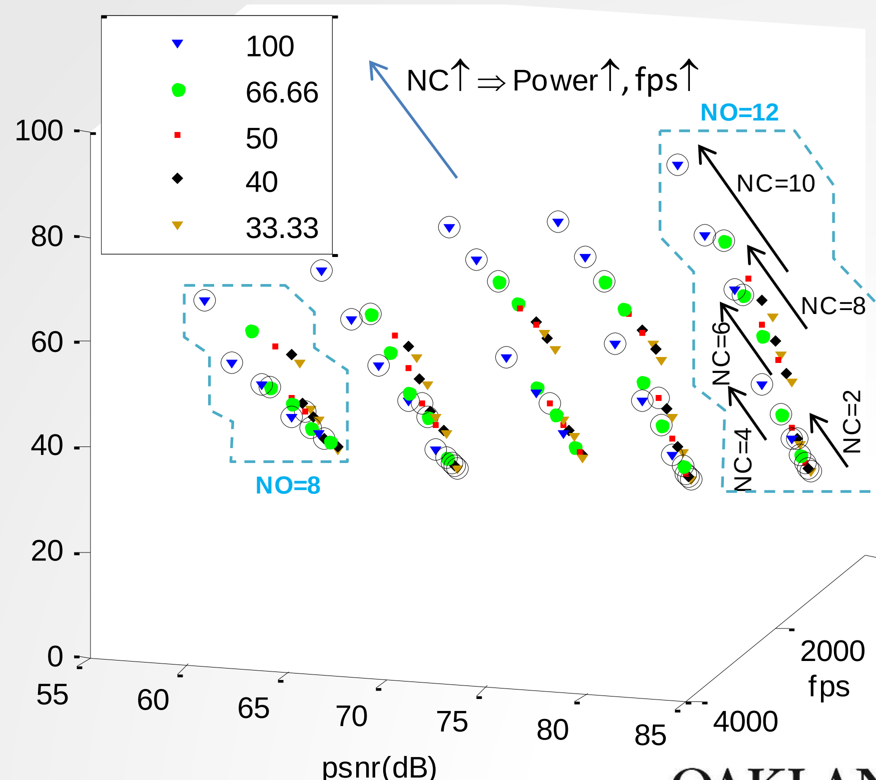


Pixel Processor (3/4)

Multi-objective optimization of the Power-Performance-Accuracy space:

- Function: *log(1 + I) + Full Histogram stretching*. Image: Lena (VGA: 640x480).
- 8-bit input image (*NI=8* fixed). Pareto points are clustered as a function of NO (# of output bits). A similar trend occurs with NC (# of cores) (not shown)
- Left side shows how power and performance depend on frequency.

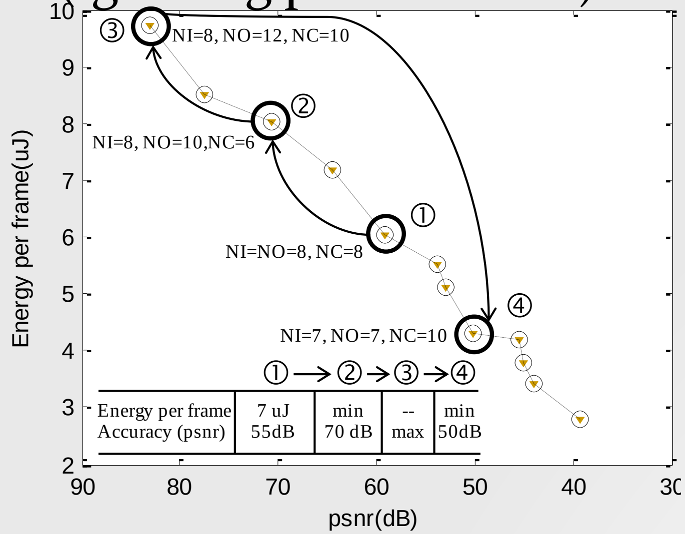


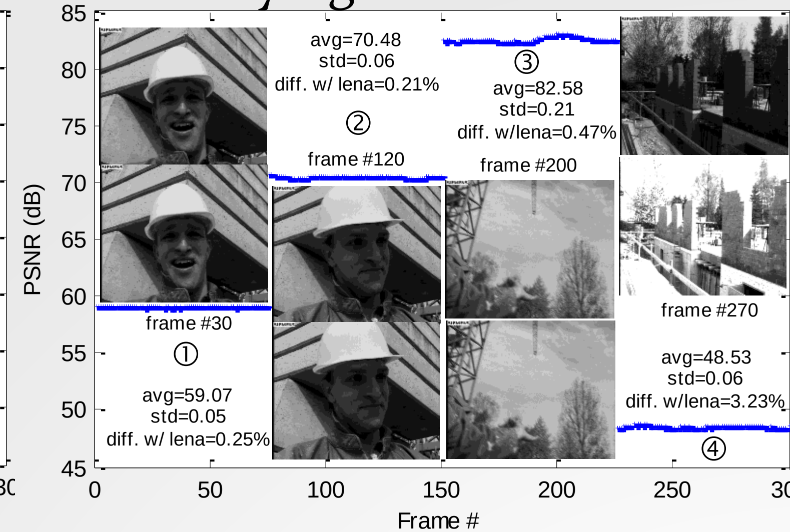


Pixel Processor (4/4)

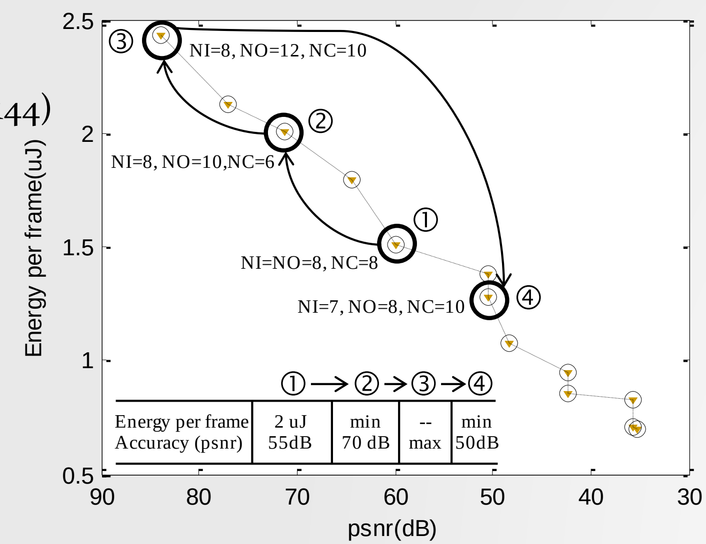
- **Dynamic Energy-Performance-Accuracy management**: We show two examples on 2D (ignoring performance) with time-varying constraints:

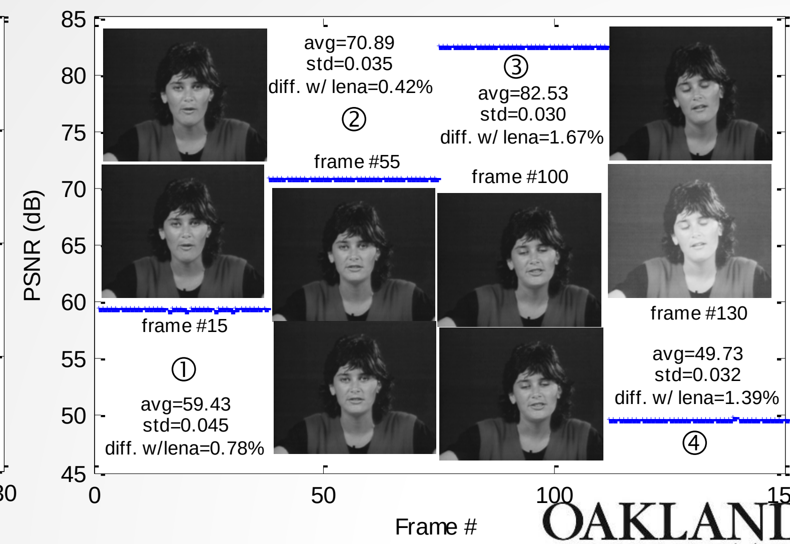
Video sequence: *foreman* (CIF: 352x288)
Function:
Gamma correction:
$I^{\gamma}, \gamma = 1/0.45$





Video sequence: *missamerica*(QCIF: 176x144)
Function:
Nonlinear contrast Stretching: $1/(1 + m/I^{\alpha})$, $m = 0.5$, $\alpha=2$





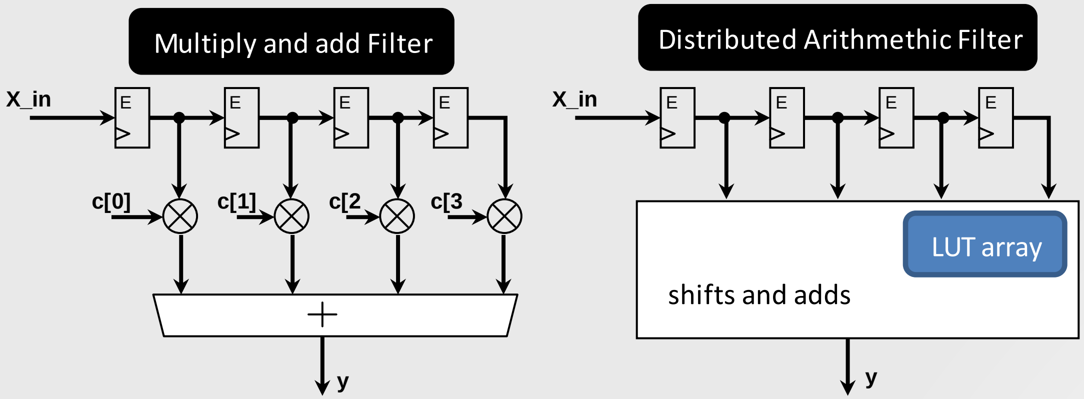
* Published in IEEE Transactions on Circuits and Systems for Video Technology

2D Separable FIR Filter (1/7)

RECRLab
Reconfigurable Computing Research Laboratory

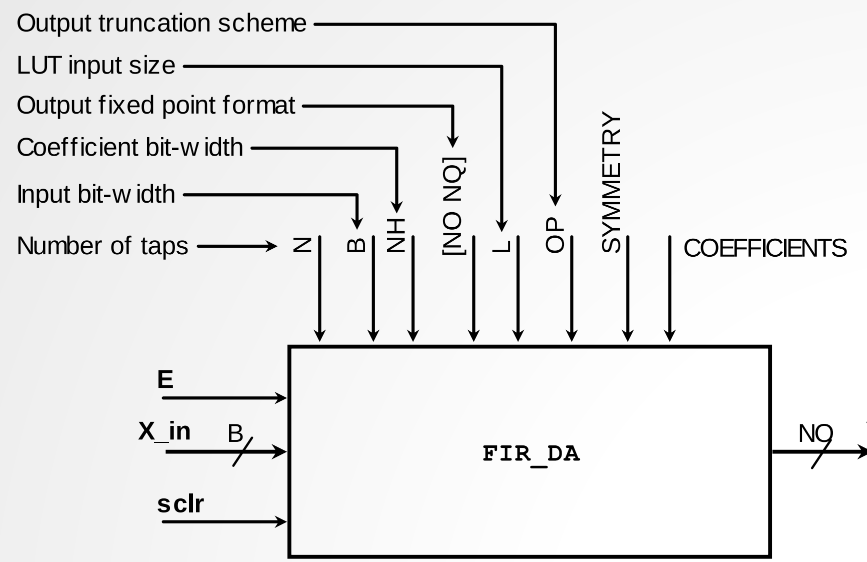
1D FIR Filter implementation

TWO TYPES OF IMPLEMENTATIONS



Distributed Arithmetic (DA) approach is more efficient since it is a LUT-based approach that turns the multiplications into shifts and adds. But it requires the coefficients to be constant.

- **Efficient implementation of a 1D FIR Filter via DPR**: Dynamic Partial Reconfiguration turns the fixed-coefficient DA filter into a variable-coefficient DA filter, at the expense of partial reconfiguration time overhead.
- **Parameterization of the VHDL-coded FIR filter core:**



* VHDL IP core available at: dllamocca.org

OAKLAND UNIVERSITY

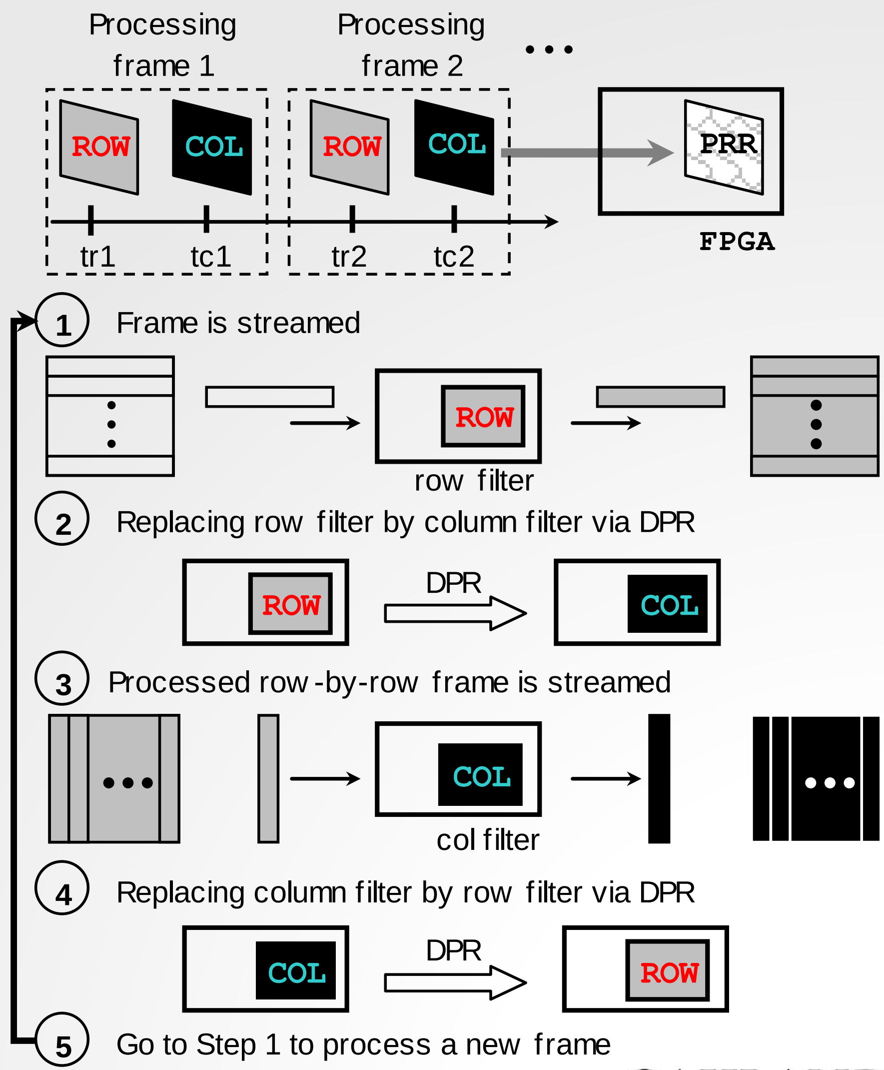
2D Separable FIR Filter (2/7)

2D Separable Filter Implementation:

- Separable FIR filters allow for efficient implementations by means of two 1D FIR Filters.
- The reconfiguration rate is constant (twice per frame).
- Cyclic Dynamic reconfiguration of two 1-D filters (usually full-filter reconfiguration):
  - Implement row filter
  - Replace by column filter
  - Implement column filter
  - Replace by row filter

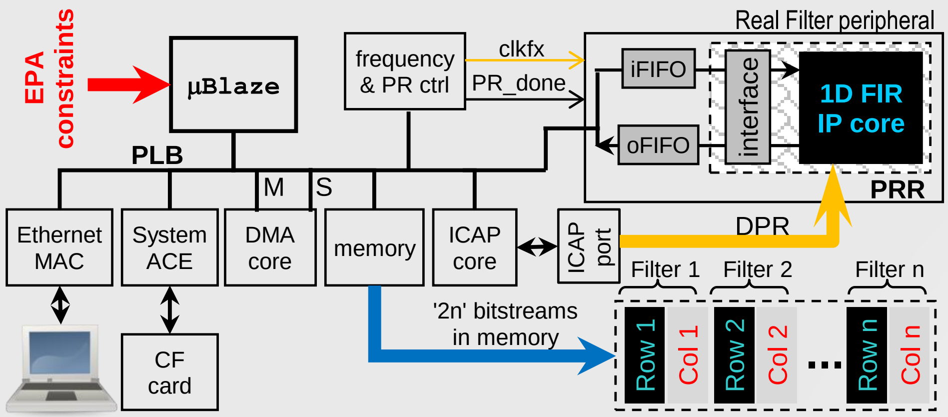
  ...

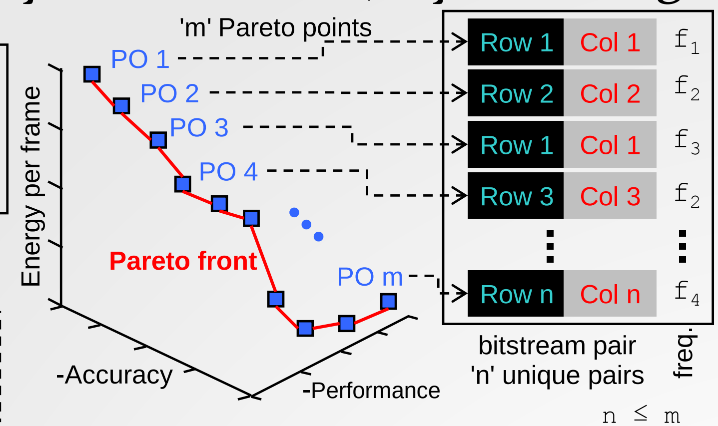
* A comparison of this 2D FIR Filter and a GPU implementation for different number of coefficients was published 2011 IEEE Field Programmable Logic Conference (FPL'2011)



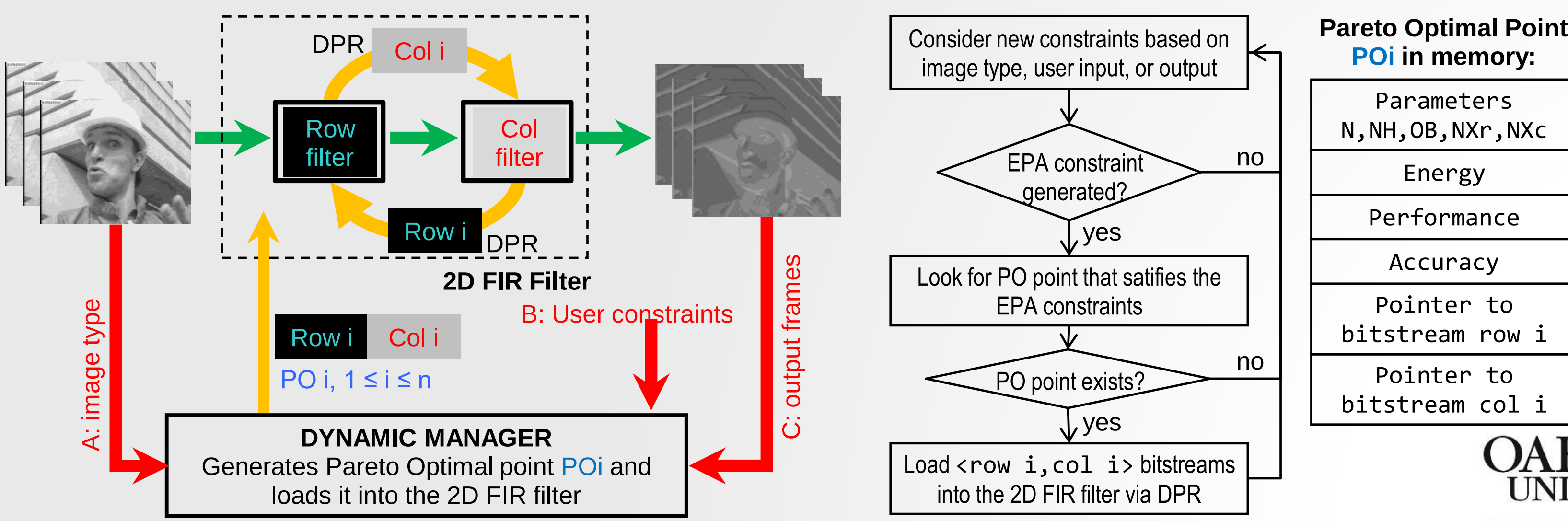
2D Separable FIR Filter (3/7)

- **Embedded System**:
- PLB Interface: The interface is inside the PRR (Partial Reconfigurable Region), so that we can dynamically modify the I/O bitwidth.
- Each 2D filter realization is represented by 2 bitstreams (1 dynamic region)





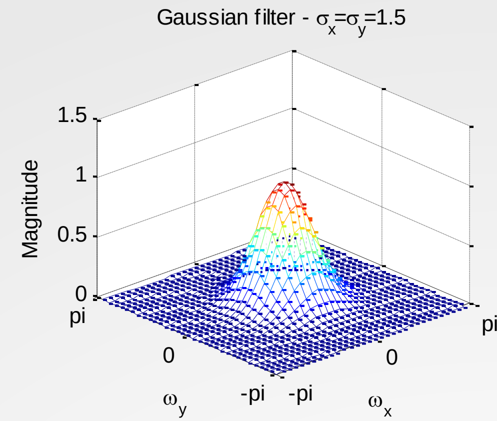
Implemented on **a ML605 Development Board**



2D Separable FIR Filter (4/7)

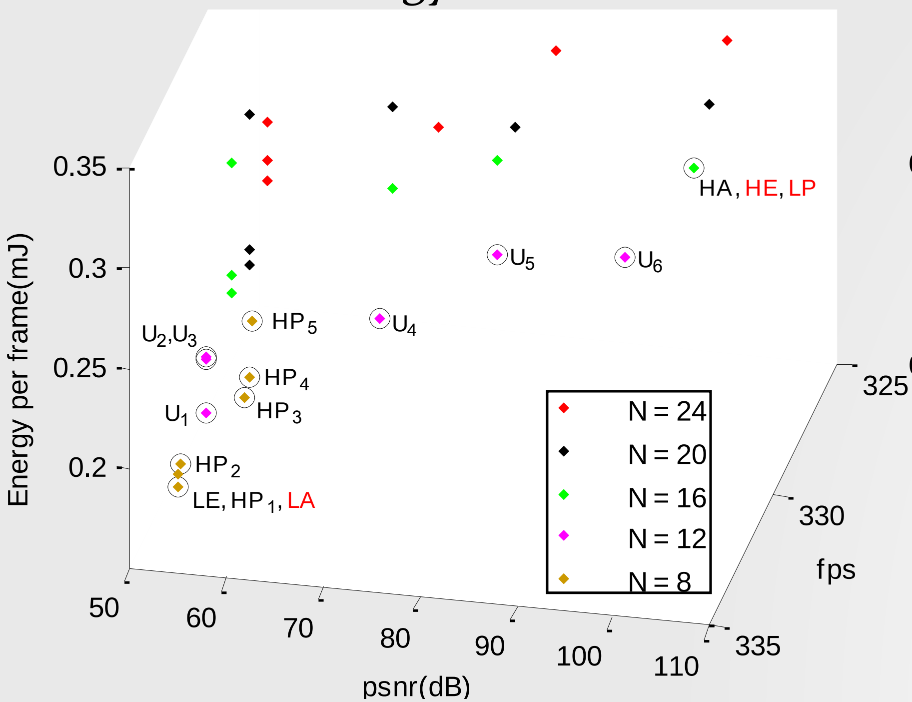
Multi-objective optimization of the Energy-Performance-Accuracy space:

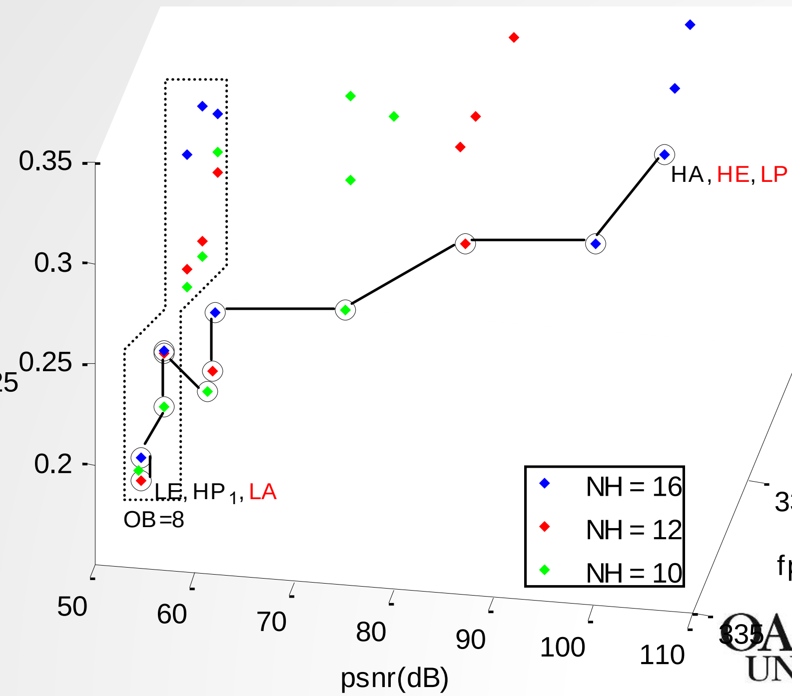
2D Filter Parameters: N (# of coefficients), NH (# of bits per coefficients), OB (# of bits per output pixel),



Results:

Low-pass Gaussian Filter, σx=σy=1.5 (symmetric coefficients).
Image: Lena (CIF: 352x288).
HA: highest-accuracy. HP: highest-performance,
LE: lowest energy



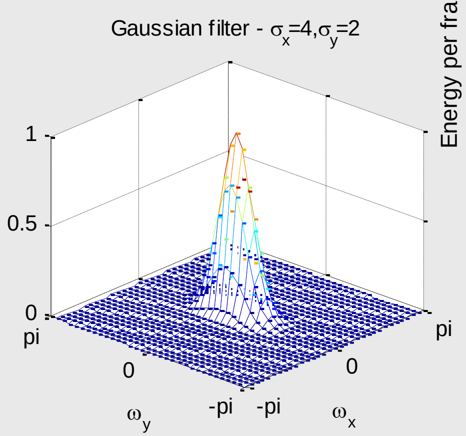


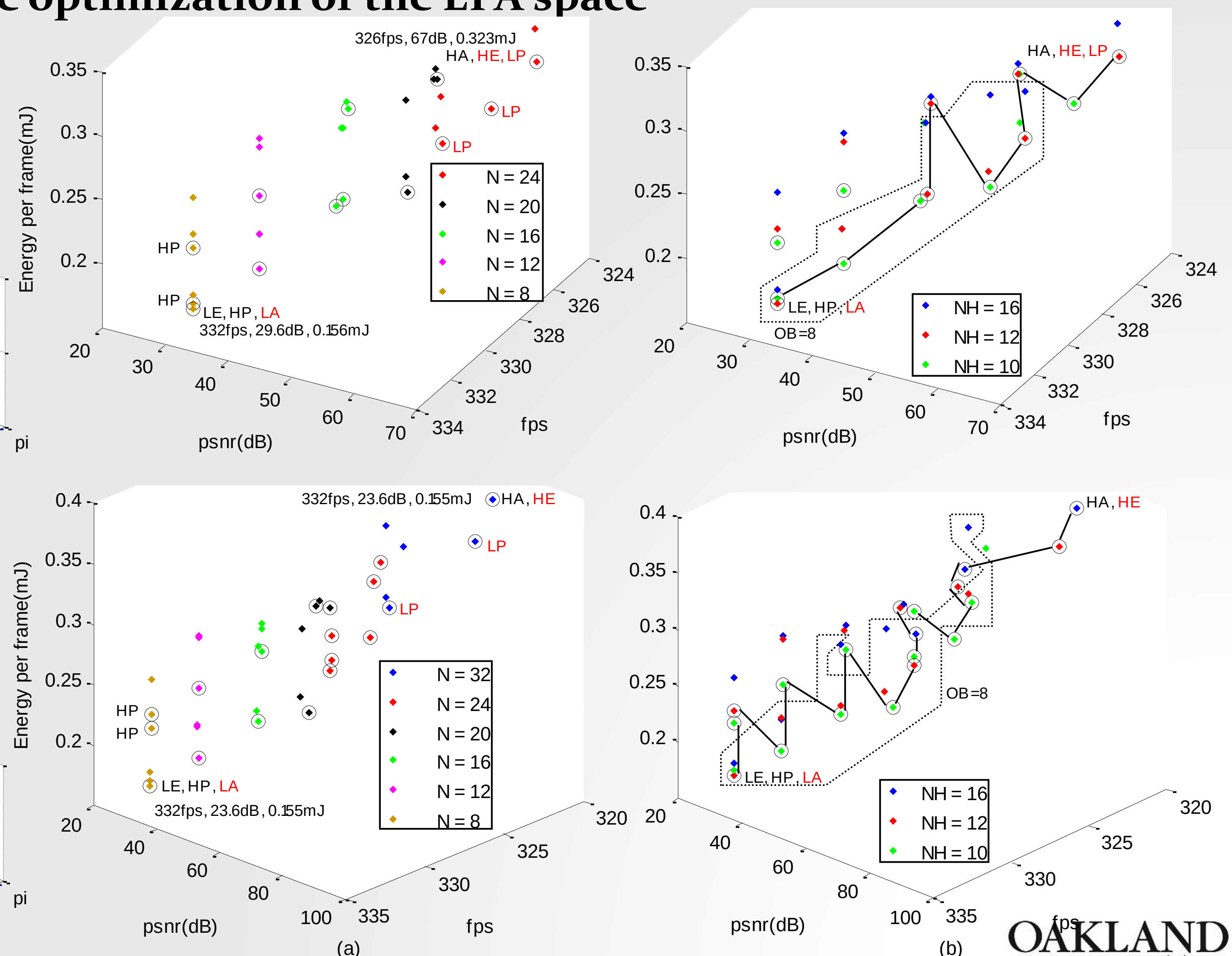
2D Separable FIR Filter (5/7)

RECRLab
Reconfigurable Computing Research Laboratory

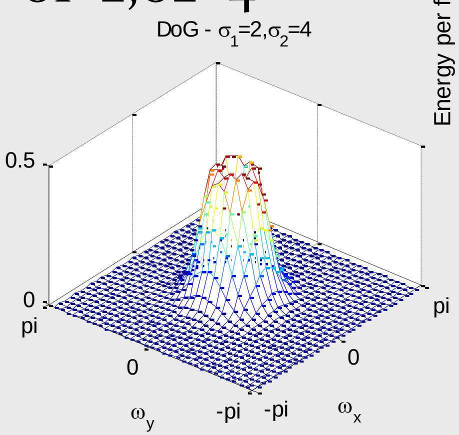
Multi-objective optimization of the EPA space

Low-pass Gaussian Filter: σx=4,σy=2





Diff. of Gaussians (DoG) filter: σ1=2,σ2=4



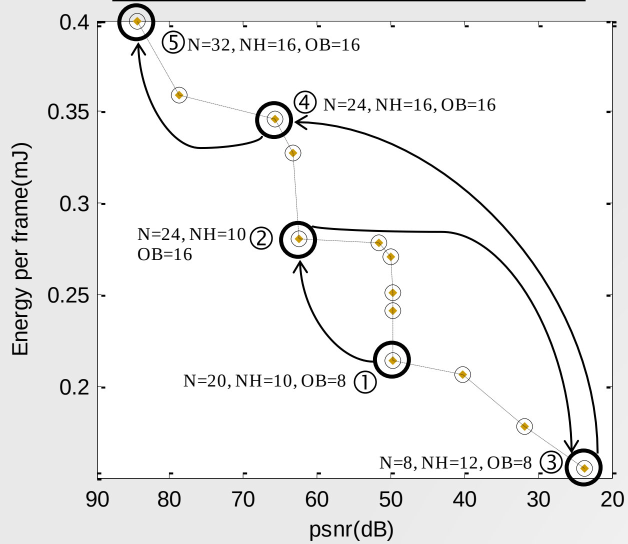
OAKLAND UNIVERSITY

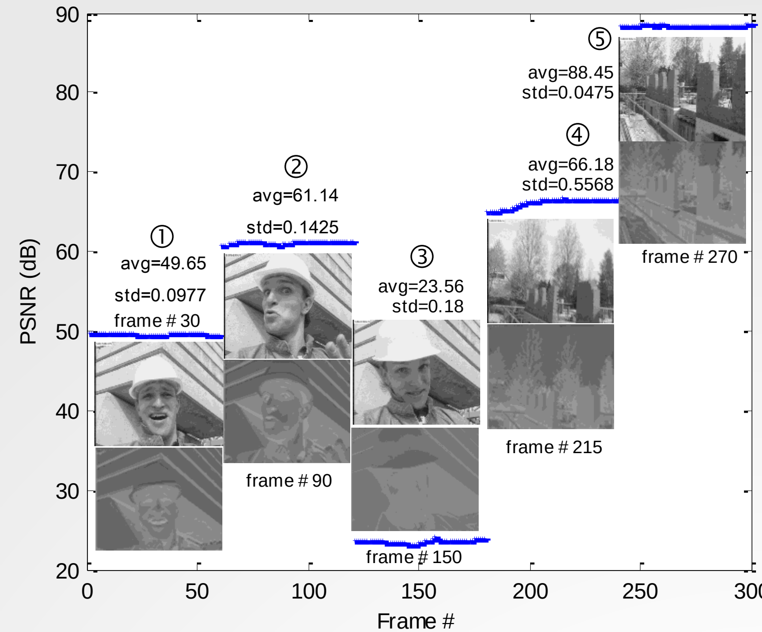
2D Separable FIR Filter (6/7)

Dynamic EPA Management (1st example): Applied on the Pareto front of the DoG filter .

Video: foreman'(CIF: 352x288)

	① →	② →	③ →	④ →	⑤
Energy per frame Accuracy (psnr)	0.3 mJ 45dB	0.3mJ max	min --	min 65dB	-- max



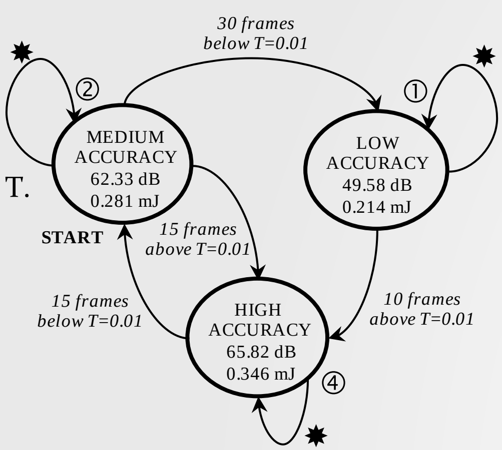


Dynamic management based on user constraints:
Time-Varying constraints are Provided at run-time by the user

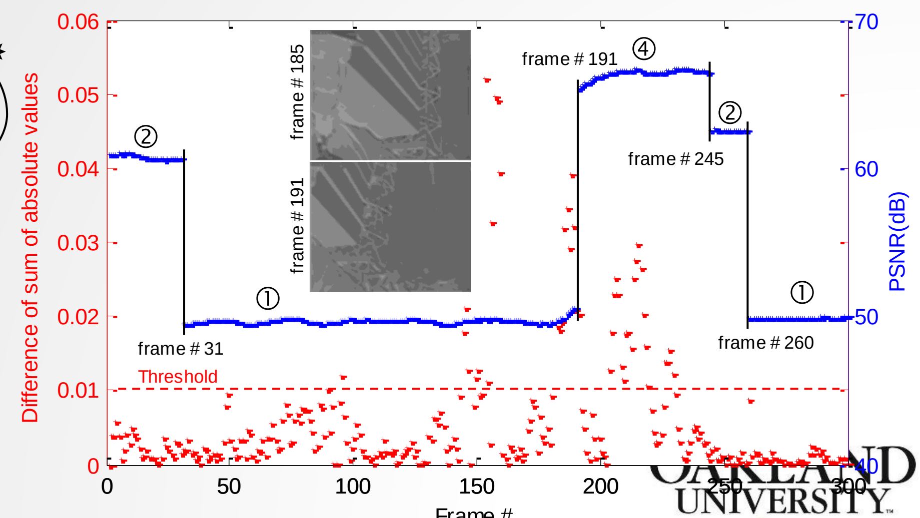
Dynamic management based on output frames:
Metric: Percentage change (T) of the filter output (between consecutive frames). Large scenes are associated with large values of T

Right side: State diagram.
Low accuracy: no significant variation in T.
Medium accuracy: some variation in T.
High accuracy: large variation in T.

Left side: Detection of Scene changes, especially Frames 185 to 191.



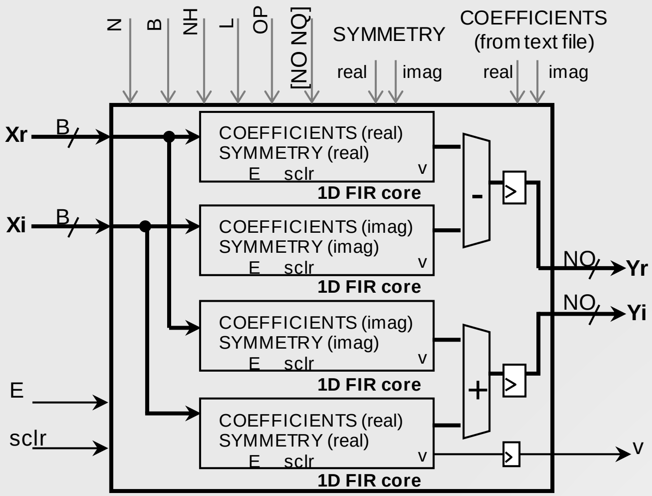
✸ If the conditions do not hold, stay in the current state

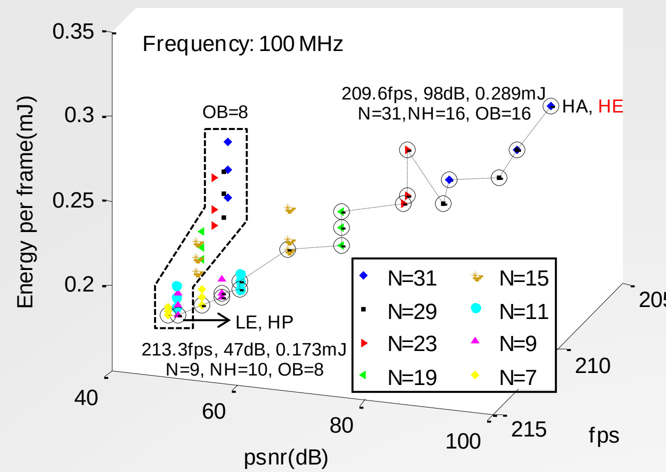


* to appear in ACM Transactions on Reconfigurable Technology and Systems

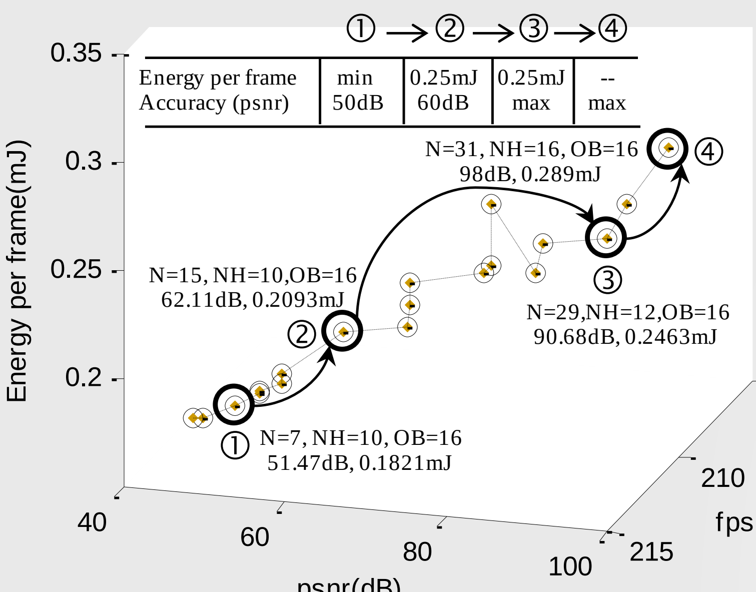
2D Separable Complex FIR Filter (7/7)

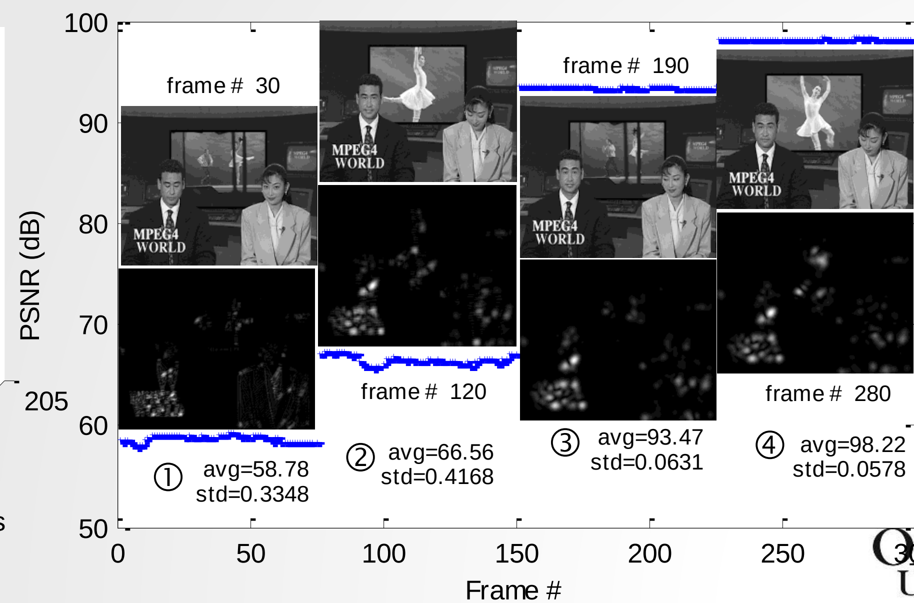
Here, the input image is complex and the filter coefficients are complex. These types of filters are very useful in AM-FM decompositions for image analysis applications.



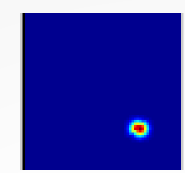


Results: Gabor complex filter, video sequence: news (CIF: 352x288),.
* Published in 2012 IEEE International Conference on Field Programmable Logic and Applications





2D Gabor at 45°



Research Areas (1/5)

RECRLab
Reconfigurable Computing Research Laboratory

RECONFIGURABLE COMPUTING:

- **Run-Time Reconfigurable Architectures under Time-Varying Constraints**
  - Automatic generation of time-varying constraints.
  - Self-aware Computing and Self-adaptive techniques.
  - Design Space Exploration for large multi-objective spaces.

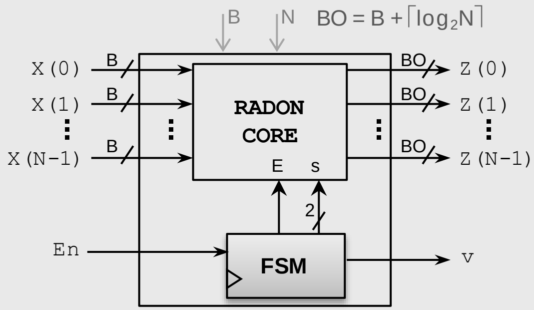
- **Advanced Topics on Computer Architecture**
  - Fully-pipelined architectures for: Signal, Image, and Video Processing: Discrete Cosine Transforms, 1D/2D Filterbanks.
  - Non-standard numerical representations: Trade-offs between double floating point precision and fixed-point precision.
  - Specialized architectures for CORDIC (trigonometric, linear, and hyperbolic functions), square root, fast division, multiplication. Use of non-standard numerical representations.
  - LUT-based design

OAKLAND UNIVERSITY

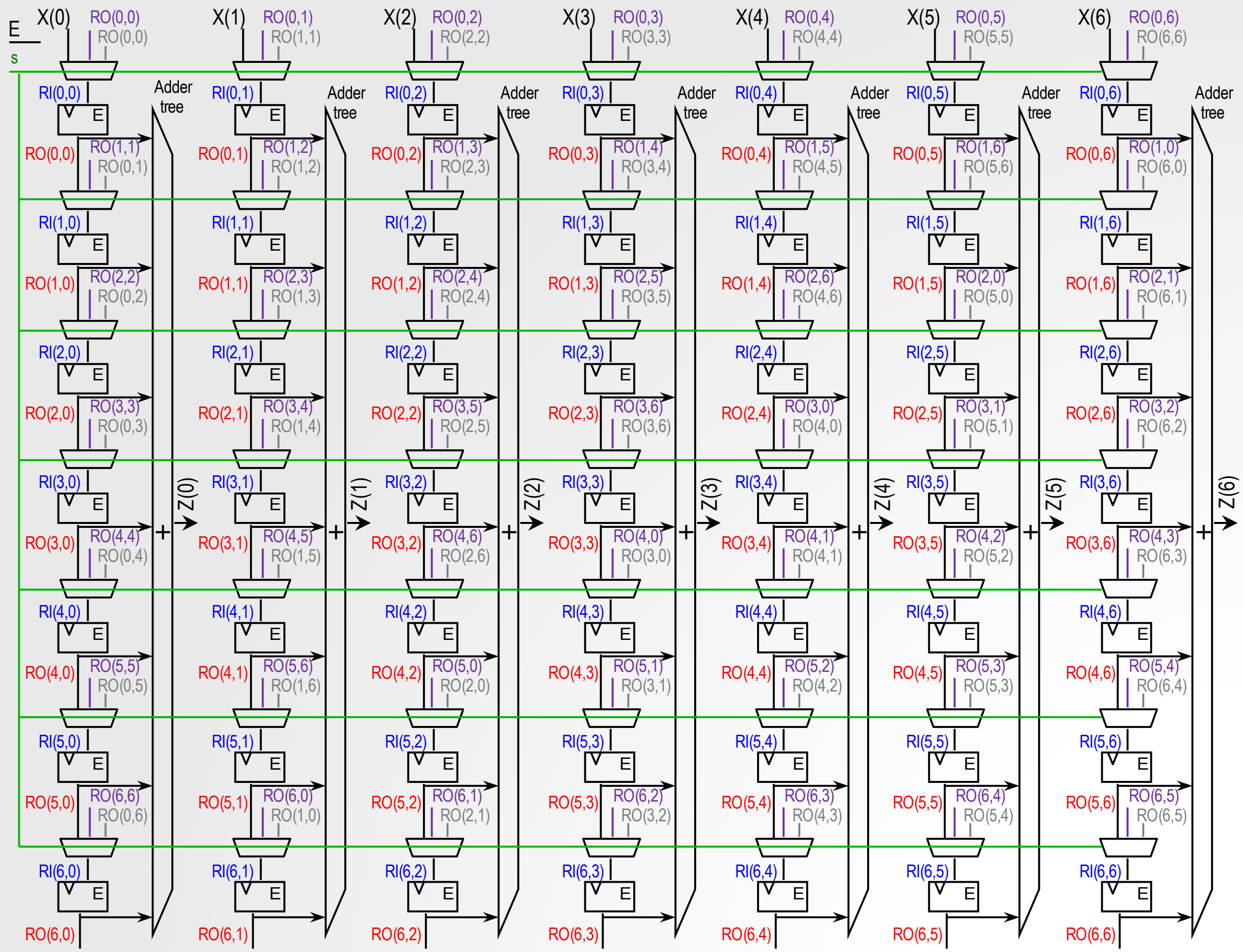
Research Areas (2/5)

❖ Example: Radon Transform:

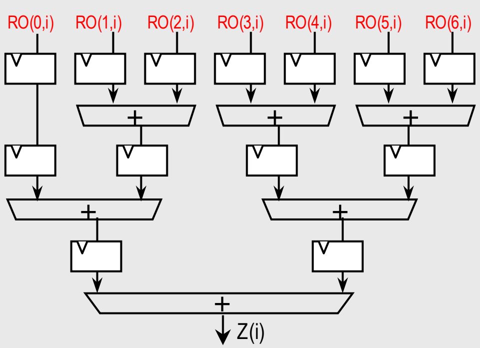
- 7x7 Radon Transform core. Presented in *SSIAI'2014 and ICIP'2014 conferences*



(a)



(b)



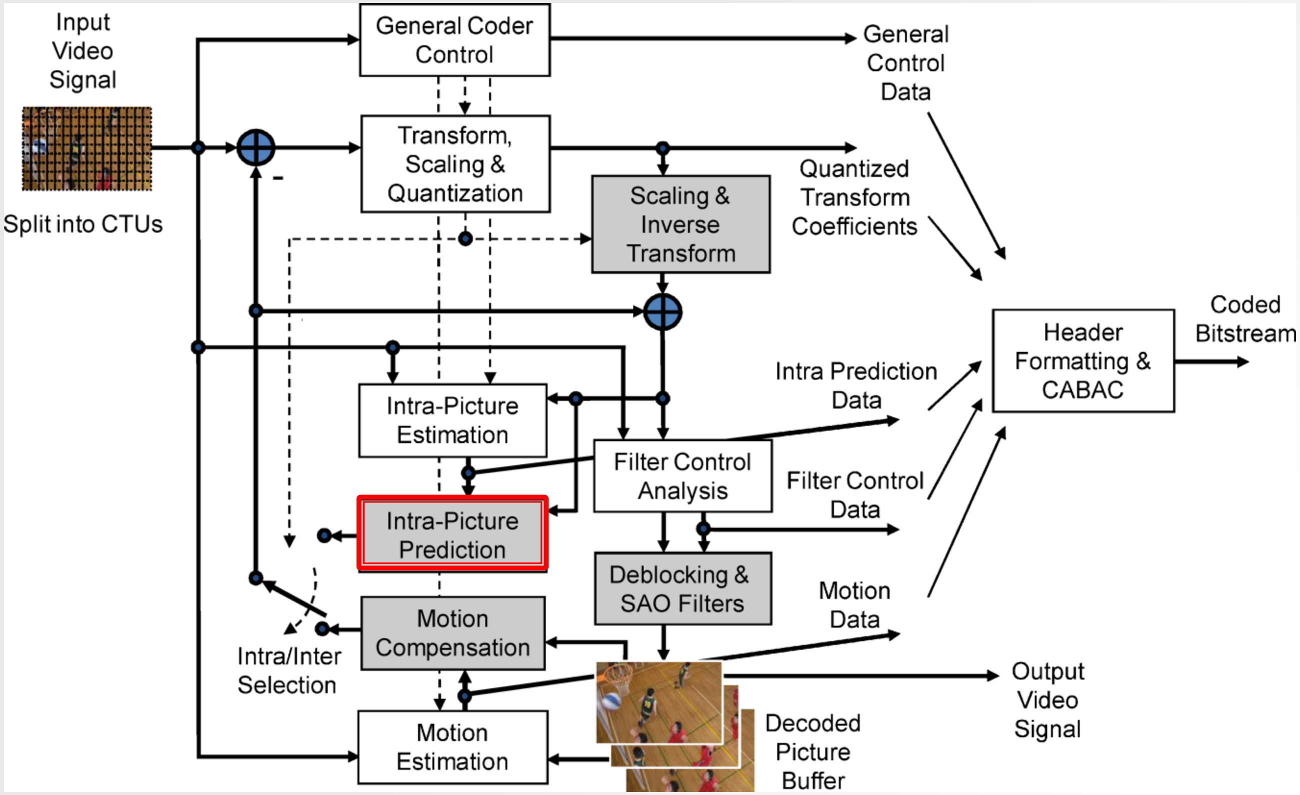
(c)

Research Areas (3/5)

RECRLab
Reconfigurable Computing Research Laboratory

❖ **Example: HEVC (High Efficiency Video Coding Standard)**

- ***HEVC Intra Prediction Core:*** Presented in the *SSIAI'2014 conference*.
- The HEVC Intra prediction core computes the following modes: Angular (33), planar (1), and DC (1).
- Current work: Implementation of HEVC Transform, Scaling, and Quantization.
- Future work: Self-reconfigurable hardware implementation of HEVC Encoder



Sullivan, G., et al, "Overview of the High Efficiency Video Coding (HEVC) Standard", IEEE TCSVT, V.22, No. 12, Dec. 2012

Research Areas (4/5)

❖ FPGA-based Embedded Systems

- Applications: automotive, networking, bioengineering, software-defined radio, communications, video compression, video filtering.
- Software-Hardware Co-Design.
- Dealing with different processors: ARM, MicroBlaze (Xilinx), Nios (Altera), OpenRISC (Open-source).
- Development of embedded interfaces for a variety of buses: Advanced eXtensible Interface (AXI, Xilinx), Avalon switch fabric (ALTERA), Wishbone (open source), etc.
- External communication interfaces (SpaceWire, CAN, Ethernet).
- Current work: Open-Source embedded system that supports DPR via Wishbone.

❖ Run-Time Reconfiguration on FPGAs

✓ Objective: Develop high-speed dynamic reconfiguration controllers:

- Support for standard buses: AXI, Wishbone.
- Hardware and software techniques the provide dramatic increases in reconfiguration speed.
- Dynamic Frequency Control on FGPAs

Research Areas (5/5)

- **Specialized techniques on FPGAs**
  - Low Power techniques
  - Advanced coding mechanism for efficient parameterization using VHDL and Verilog HDL.
  - Test-bench generation
  - Crossing clock domains

GPU PROGRAMMING:

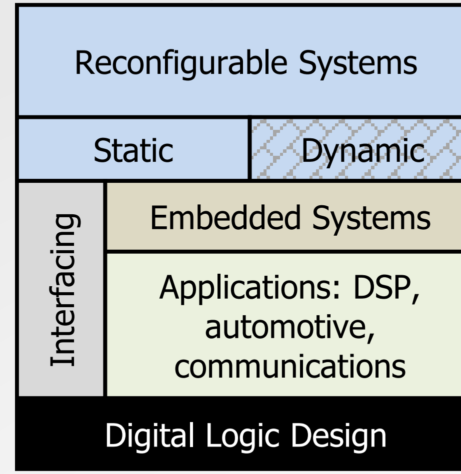
- High performance implementation of Digital Signal, Image, and Video Processing Algorithms.
- Integration with multi-threaded implementations on CPUs
- Comparisons with FPGA implementations.

EMBEDDED SYSTEMS

- Microcontroller-based System
- Efficient architectures for embedded interfaces
- Embedded design using System-on-Chip that incorporates analog, programmable logic, memory, and microcontroller (e.g. Cypress devices)

Teaching Plans

- **ECE378: Digital Logic and Microprocessor Design (Winter 2015)**
  - Digital System Design
  - Microprocessor Design in VHDL
  - Digital Synthesis with VHDL
  - Parameterized VHDL coding



- **ECE495/595: Reconfigurable Computing (Fall 2015)**
  - Hardware/Software co-design on FPGAs
  - Self-Reconfigurable systems: Partial Reconfiguration
  - Advanced topics in Computer Arithmetic
  - Applications in:
    - ✓ Digital Signal, Image, and Video Processing
    - ✓ Communication interfaces (SpaceWire, CAN, Ethernet)

Conclusions

RECRLab
Reconfigurable Computing Research Laboratory

- *A framework was presented for Dynamic Management of Optimization of Run-Time Reconfigurable Architectures. The examples presented (Pixel Processor, 2D FIR Filter) were successfully tested on several standard video sequences.*

- *The results suggest that the general framework can be applied to a variety of digital systems. This framework will lead to exciting new methods. As an example, consider the automatic generation of time-varying constraints. For example: detection of a scene triggers a requirement for increased accuracy, a scene remaining still triggers a requirement for a decrease in energy consumption.*

- *The presented work opens up* ***new exciting interdisciplinary research opportunities*** *(e.g.: automotive applications, design space exploration, automatic constraints generation, pervasive healthcare applications, low power applications).*

OAKLAND UNIVERSITY