



# Reconfigurable Data Path Processor for Space Applications

Gregory W. Donohoe  
K. Joseph Hass  
University of New Mexico  
Institute of Advanced Microelectronics  
801 University Blvd., SE, Suite 206  
Albuquerque, NM 87106

[donohoe@mrc.unm.edu](mailto:donohoe@mrc.unm.edu)

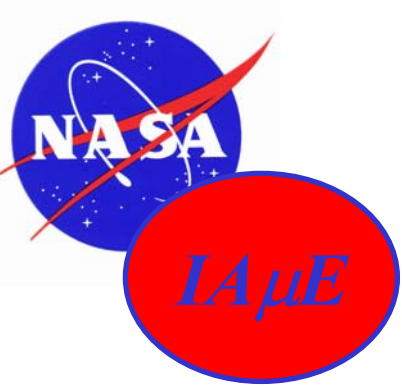
Stephen Bruder  
University of New Mexico  
New Mexico Institute of Mining and  
Technology  
Electrical Engineering Dept.  
Socorro, NM

[bruder@ee.nmt.edu](mailto:bruder@ee.nmt.edu)



# Overview

- Introduction
- Implementation Technology
- Architecture
- Algorithm implementation examples
- Conclusion



# RDPP Requirements

**Space Qualified**

Radiation tolerant

**Low power consumption**

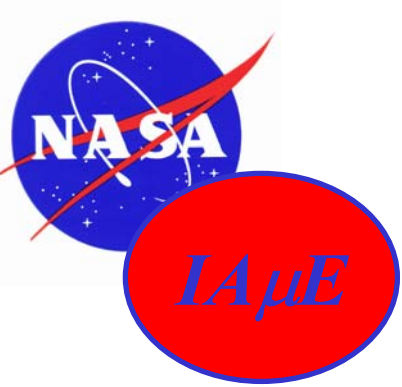
**Versatile**

Multiple classes of scientific algorithms

**High Performance**

Complex algorithms at high speed

**Designer Friendly**



# Development Approach

**Versatility**

*Architecture*

**Reconfigurable &  
Extensible**

**High Data  
Rate**

*Architecture*

Highly Parallel

**Low Power  
Consumption**  
*Implementation  
Technology*

ULP Design

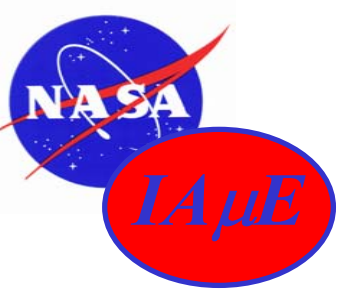
**Designer  
Friendly**

*Support  
Software*

Translators,  
Simulators

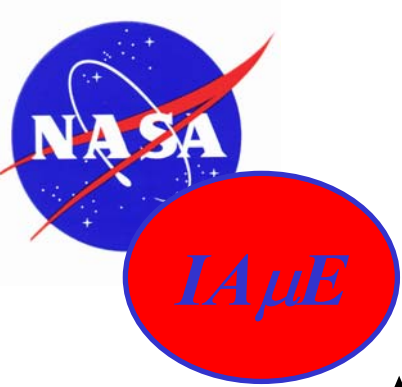
**Space  
Qualified**  
*Implementation  
Technology*

ULP-RT Library

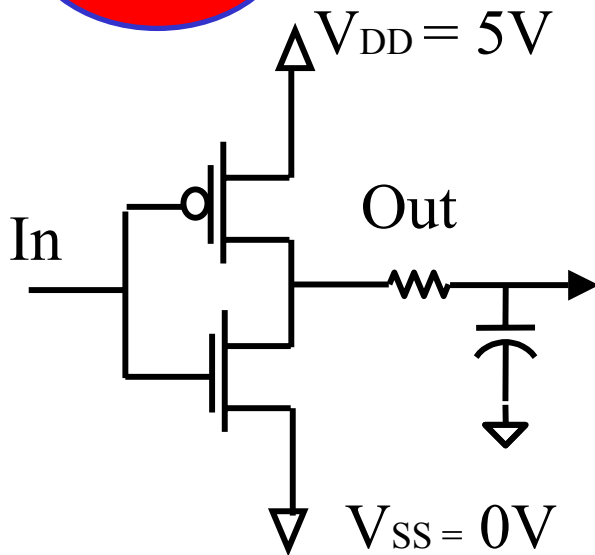


# Implementation Technology

Ultra-low-power, Radiation Tolerant  
CMOS



# CMOS Power Consumption



Dynamic (switching) power

$$P_D \propto fCV_{DD}^2$$

Short-circuit (crossbar) power

$$P_{SC} \propto (V_{DD} - 2V_t)^3$$

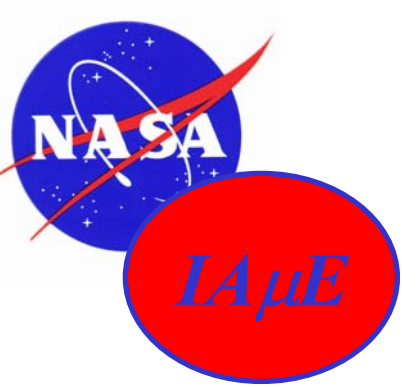
Static (leakage) power

$$P_S \propto \exp(Z)V_{DD}$$

$Z$  = physical & process parameters

$$\text{Total power: } P_T = P_D + P_{SC} + P_S$$

*With ULP technology, dynamic and short-circuit power are reduced by orders of magnitude. This alters the system design parameters.*



# CMOS Energy Consumption

*Dynamic:* function of switching events

*Static:* function of number of transistors in system

## **Conventional CMOS low-power design strategy**

Dynamic switching dominates; idle transistors consume no power.  
Design to keep circuits idle.

- Maximize functionality per switching event.
- Disable unused modules circuits (clock gating).
- Avoid large, parallel, redundant data paths.

## **ULP Design strategy**

Balance between static and dynamic energy consumption.

- Make maximum use of gates in the system.
- Maximize functionality through parallel data paths.
- Use conditional switching instead of branching.



# ULP/RT CMOS

CULPRiT: Ultra-low power CMOS

- Fortuitously, exhibits excellent total-dose radiation tolerance.

Radiation Tolerant Standard Cell Library

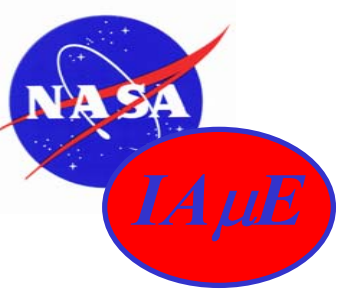
- Single Event Upset hardness through circuit design.
- Single Event Latchup hardness through layout techniques.

Program underway to combine the two technologies in AMI 0.35 $\mu$  CMOS process.

*RDPP will initially be designed for standard AMI 0.35 $\mu$  CMOS, with upgrade path to ULP/RT technology.*



# Architecture



# Architecture Development

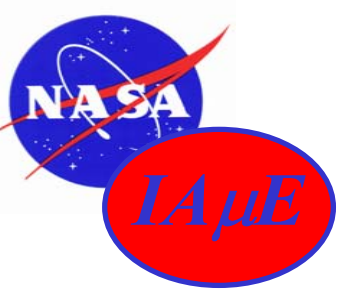
## 1. Define Computational Model

## 2. Define Architectural Components

- Data representation
- Processing elements
- Internal storage
- Interconnects
- Data flow and control strategy
- External interfaces

## 3. Verify architecture on sample problems

- Paper-and-Pencil simulation
- Hardware Design Language, Computer Simulation



# Computational Model

Synchronous Data flow



# Importance of Computational Model

“Applications are built on a model of computation, whether the designer is aware of this or not.”

-- Edward Lee, U.C. Berkeley

## Turing-Church

Random-access memory  
No time, just sequence  
Component = subroutine

## Embedded Real-Time

Limited memory  
Execution time & latency  
Interrupt driven  
Component = thread

## Dataflow

Data-driven  
Systolic execution  
Streaming data (DSP)

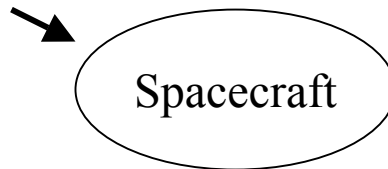
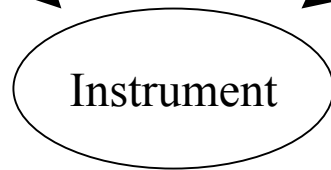
A. Scientist/Signal  
Processor  
*Matlab, “C”*

B. Instrument  
Designer  
*Hardware,  
Dataflow*

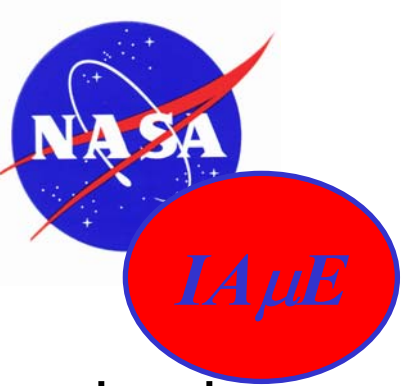
## Reconfigurable

**May combine elements  
of different models**

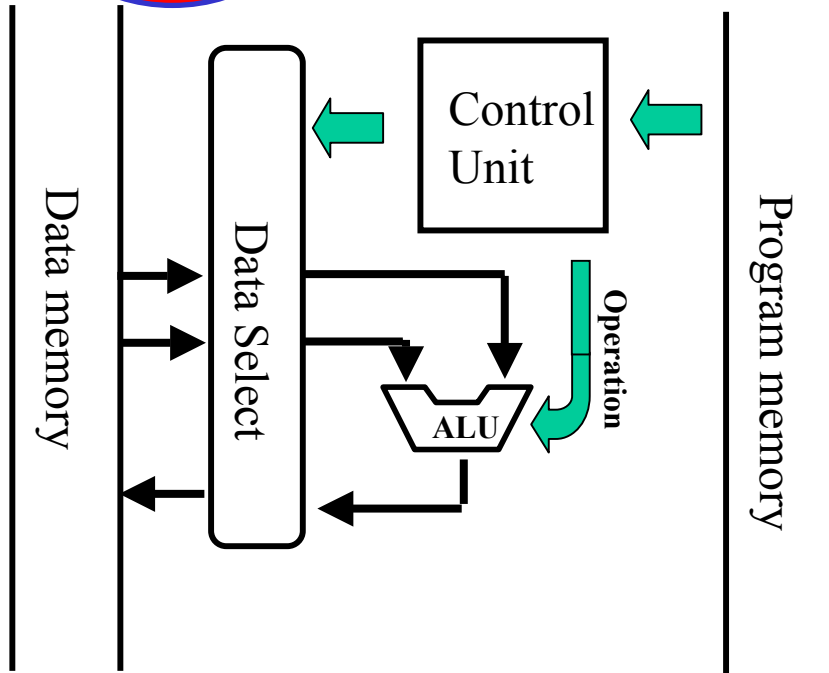
C. Spacecraft  
Integrator



Scientist wants to compile Matlab workspace into an embedded system.  
*Problem:* different models of computation.



# Sequential Computational Model



## Assumptions

- Infinite memory
- Randomly addressable
- Sequential operation:

Fetch  
Decode  
Execute  
Store

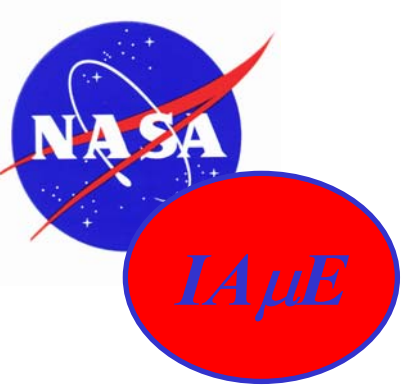
## Advantages

- Extremely versatile
- Execution stream agility
- Best choice for random-access applications, e.g. control.

## Disadvantage

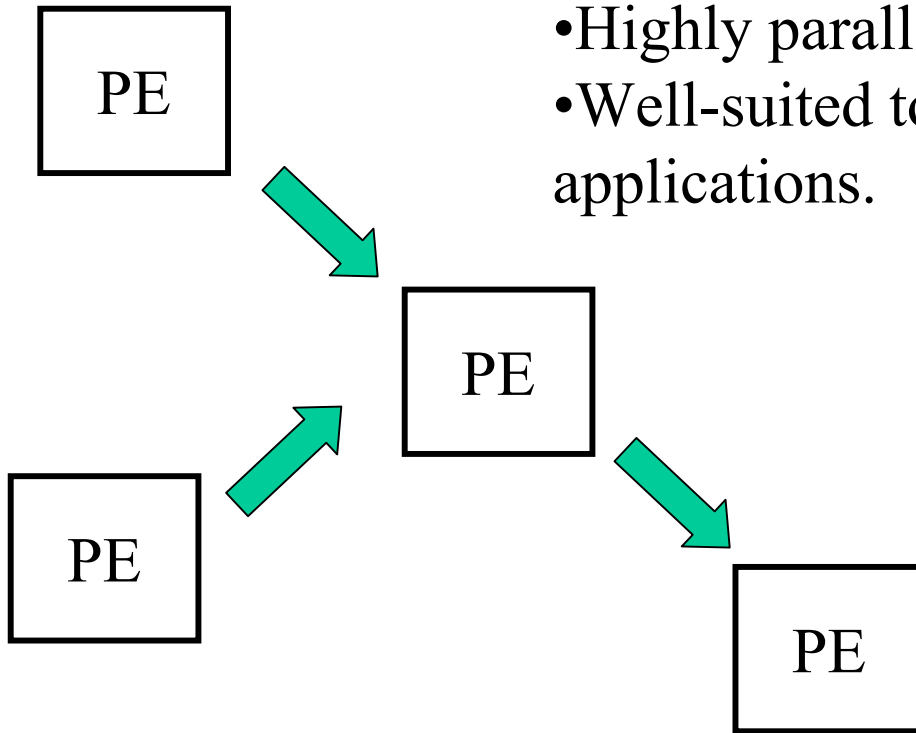
*Major performance bottleneck for data intensive processing.*

Programming languages:  
C, C++, Matlab



# Computational Models: Synchronous Dataflow

- Control and data flow through network.
- Highly parallel.
- Well-suited to embedded, data-intensive applications.

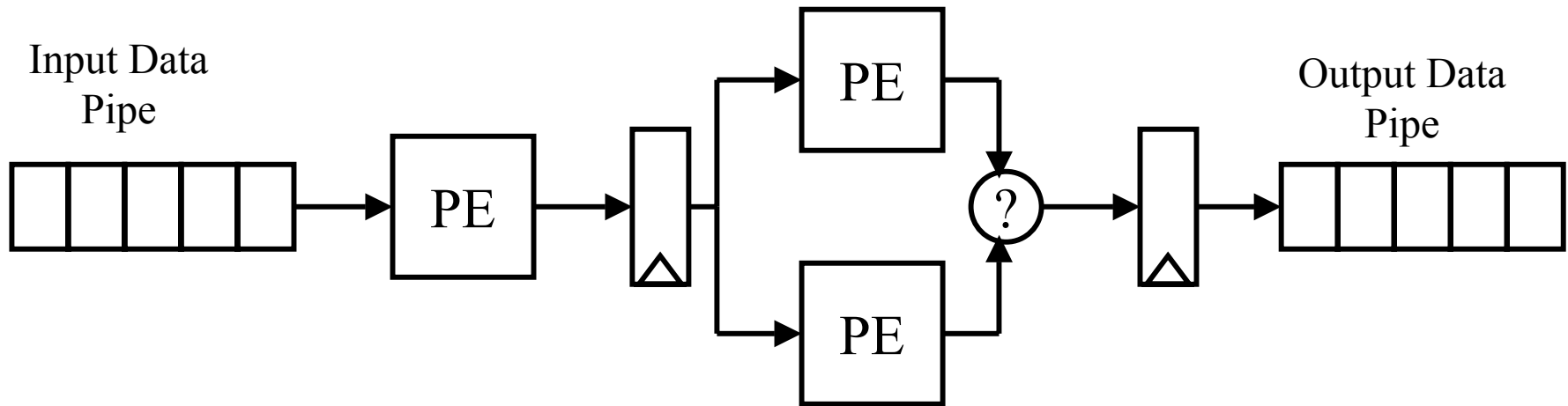


*Programming languages*

- Khoros
- Simulink

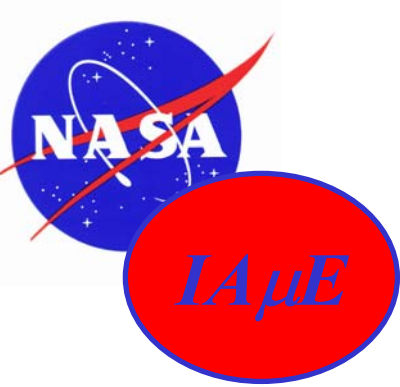


# Synchronous Data Flow Example



***Agile data flow* instead of agile control flow.**

- Conventional processor branches by executing one instruction stream or another.
- Data flow machine executes both simultaneously, but only one result is passed on downstream.



# RDPP Will Implement a Synchronous Data Flow under Control of a Sequential Machine

## Assumptions

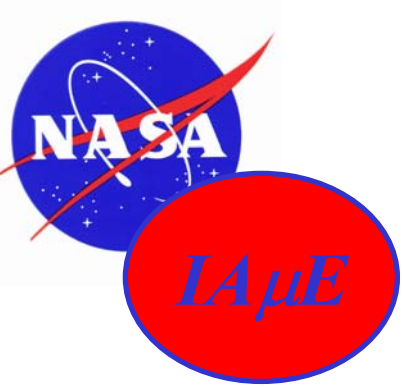
- Pipeline processor acts as an accelerator, or co-processor, to a host CPU.
- Host CPU can address, read and write individual registers within Processing Elements.
- Host CPU establishes interconnects.
- Host CPU can set up a pipeline, run it on a cycle-by-cycle basis, then tear it down.
- There can be more than one independent pipeline.

## Kinds of Data Objects

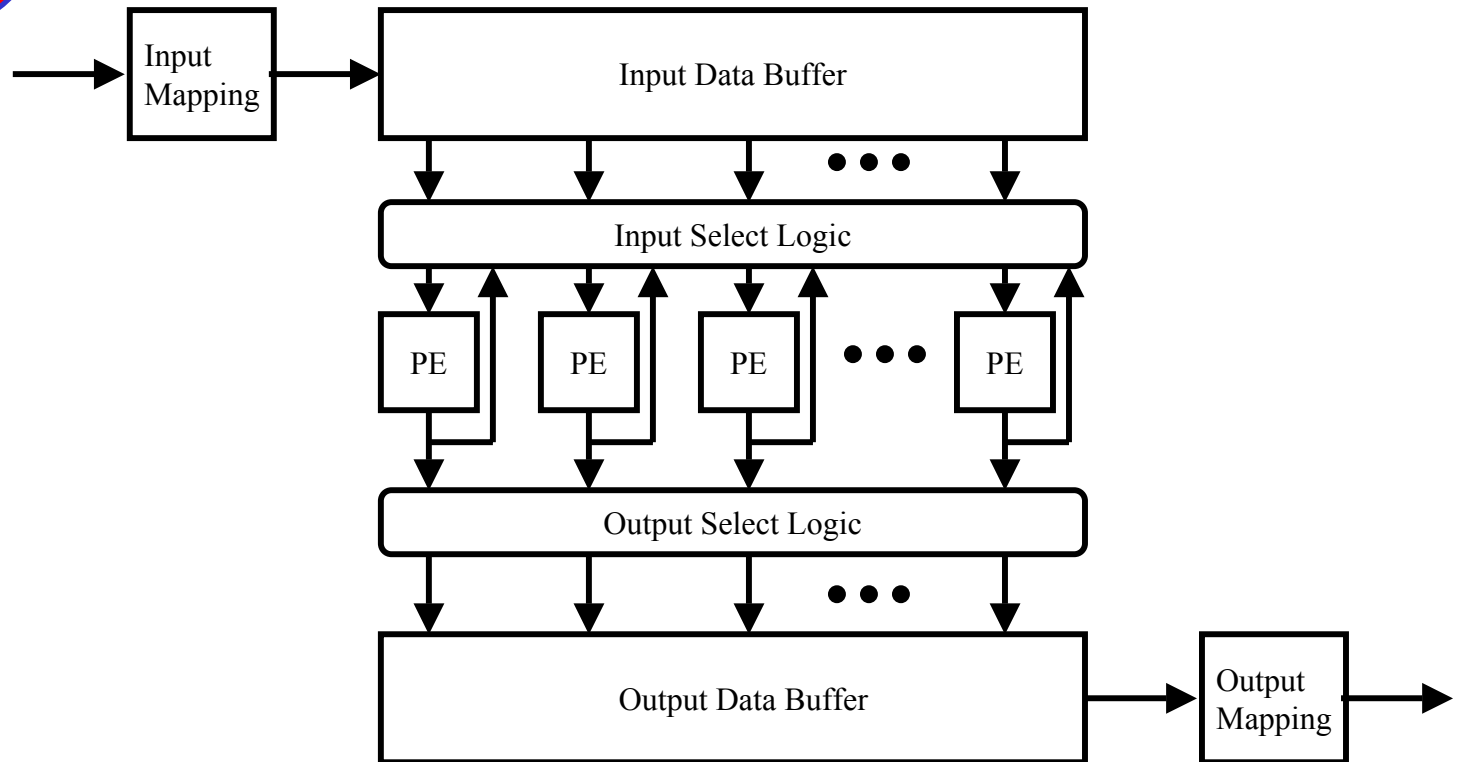
- Image and signal data
- Image coordinates (addresses)
- Status flags (e.g., “done”)
- State information: counts (Hough transform), region labels (connected component analysis)



# Proposed Architecture



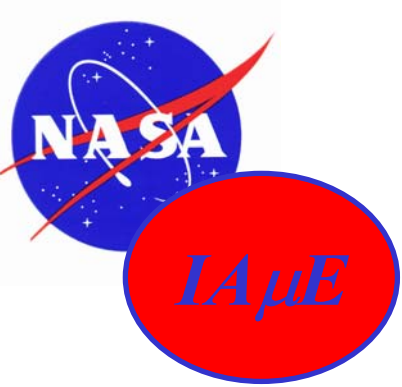
# Proposed RDPP Architecture



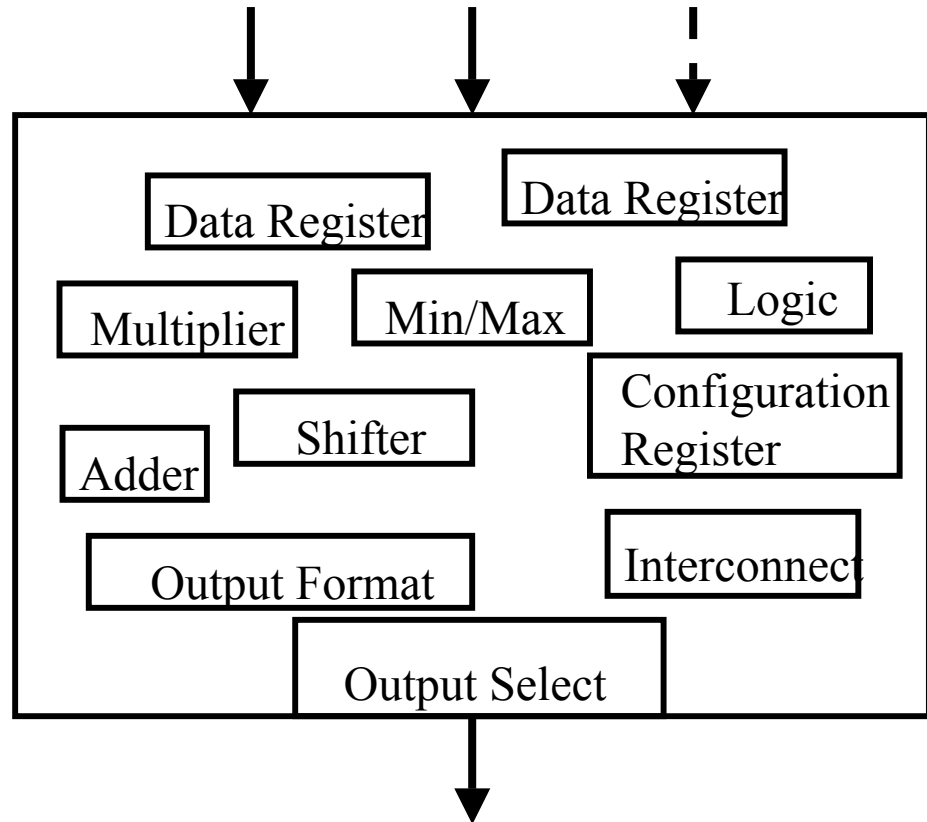


# Architectural Elements

- Dynamically controlled, reconfigurable, synchronous data flow pipelines.
- 24 bit (S/I/F) data format.
- Multiplexer-based flow control in pipeline.
- Configurable processing elements.
- 1024 x 3 x 24 bit Input Data Buffer
- Crossbar Interconnect
- Output buffer TBD

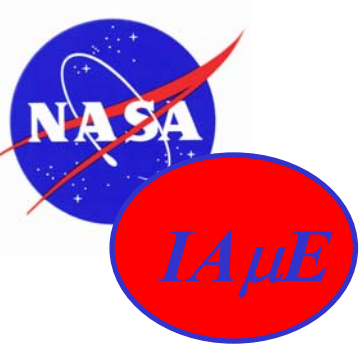


# Contents of a Processing Element



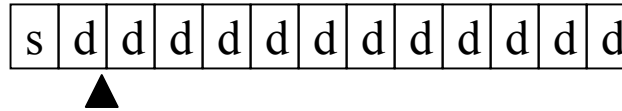
## Issues:

- Maximize component re-use.
- Maintain throughput.
- Operate on all 4 data objects.



# Data Representation and Normalization

Fixed Point:



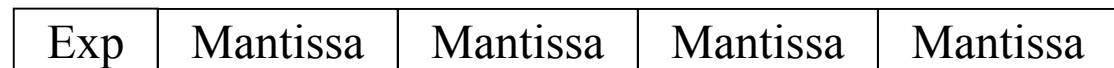
- Simple hardware
- Maximum precision within limited dynamic range
- Design-time normalization; maximum user control

Floating Point:

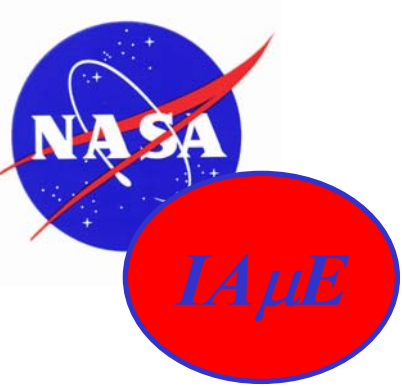


- Run-time normalization
- Essentially sequential operations (shift and test)
- Maximum user convenience
- Very complex hardware

Block Floating Point:



- Block-normalization
- Share the control hardware



# Design-time Scaling

- If processing elements use floating point, we severely limit the number of PE's available.
- Floating point applicable to a small number of algorithms.

**Challenge:** *the convenience of floating point, with the implementation efficiency of fixed point.*

## Sign/Integer/Fraction (S/I/F) Format

S = Number of sign bits

I = Number of integer bits

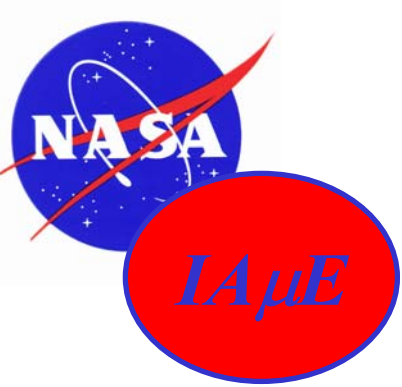
F = Number of fraction bits

Allows a fixed word to be interpreted in different ways.

- Design-time software converts high-level representation to SIF format.
- Translates data operations into series of RDPP primitives:  
Multiply, Add, And, Clip, Shift, etc.



# Implementation Examples



# Some Challenge Problems

Convolution (e.g., image enhancement).

Nonlinear filters: min, max, median, speckle reduction.

Sensor readout correction and calibration.

Bad pixel replacement.

Object detection and tracking.

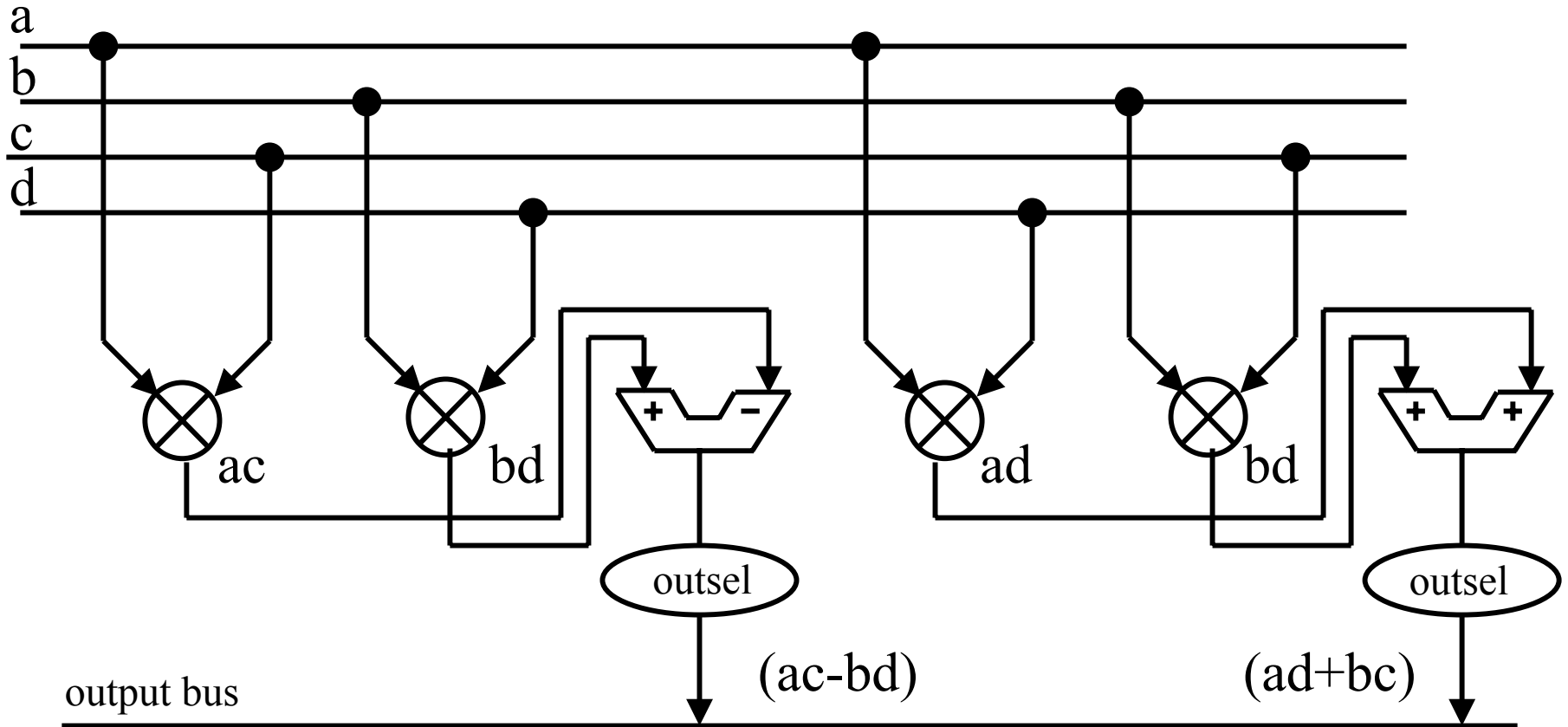
Sensor data conversion: Fourier Transform Hyperspectral  
Imager

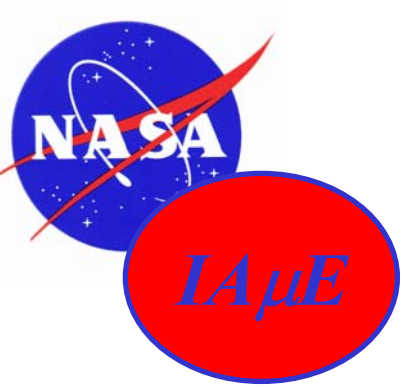
*Must be extensible to large data sets.*



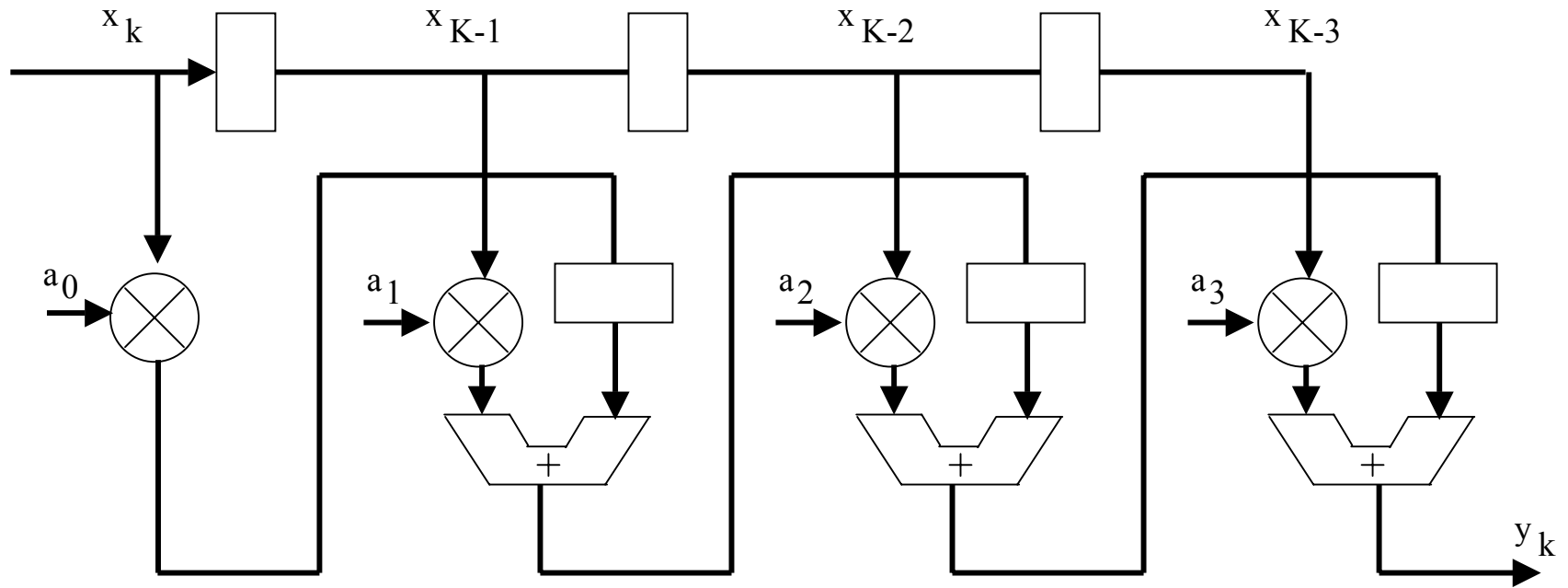
# Complex Multiply

$$(a + jb)(c + jd) = (ac - bd) + j(ad + bc)$$





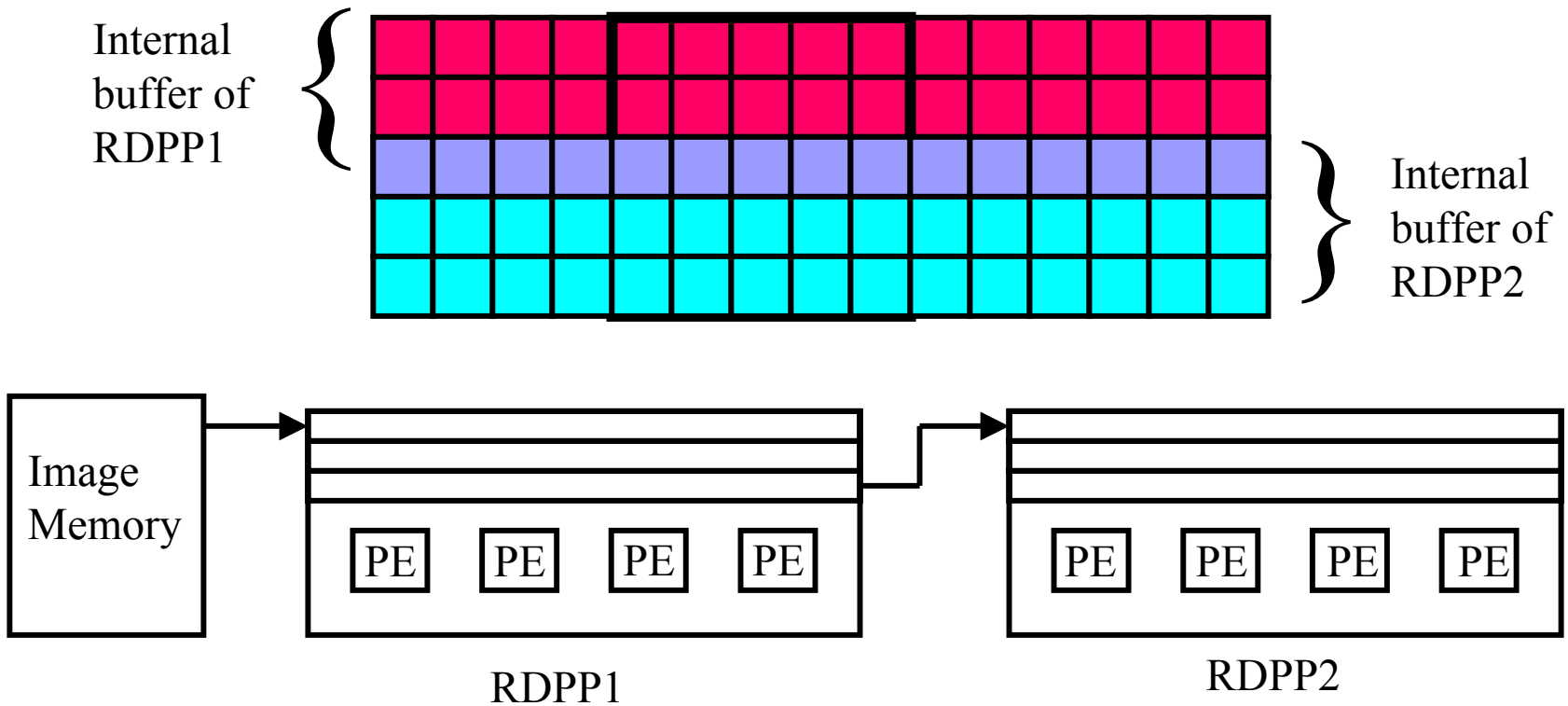
# Convolution



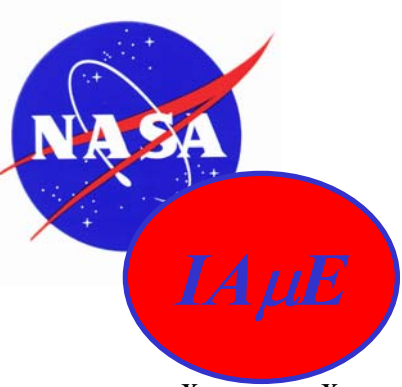
A natural application for systolic pipelined processing.



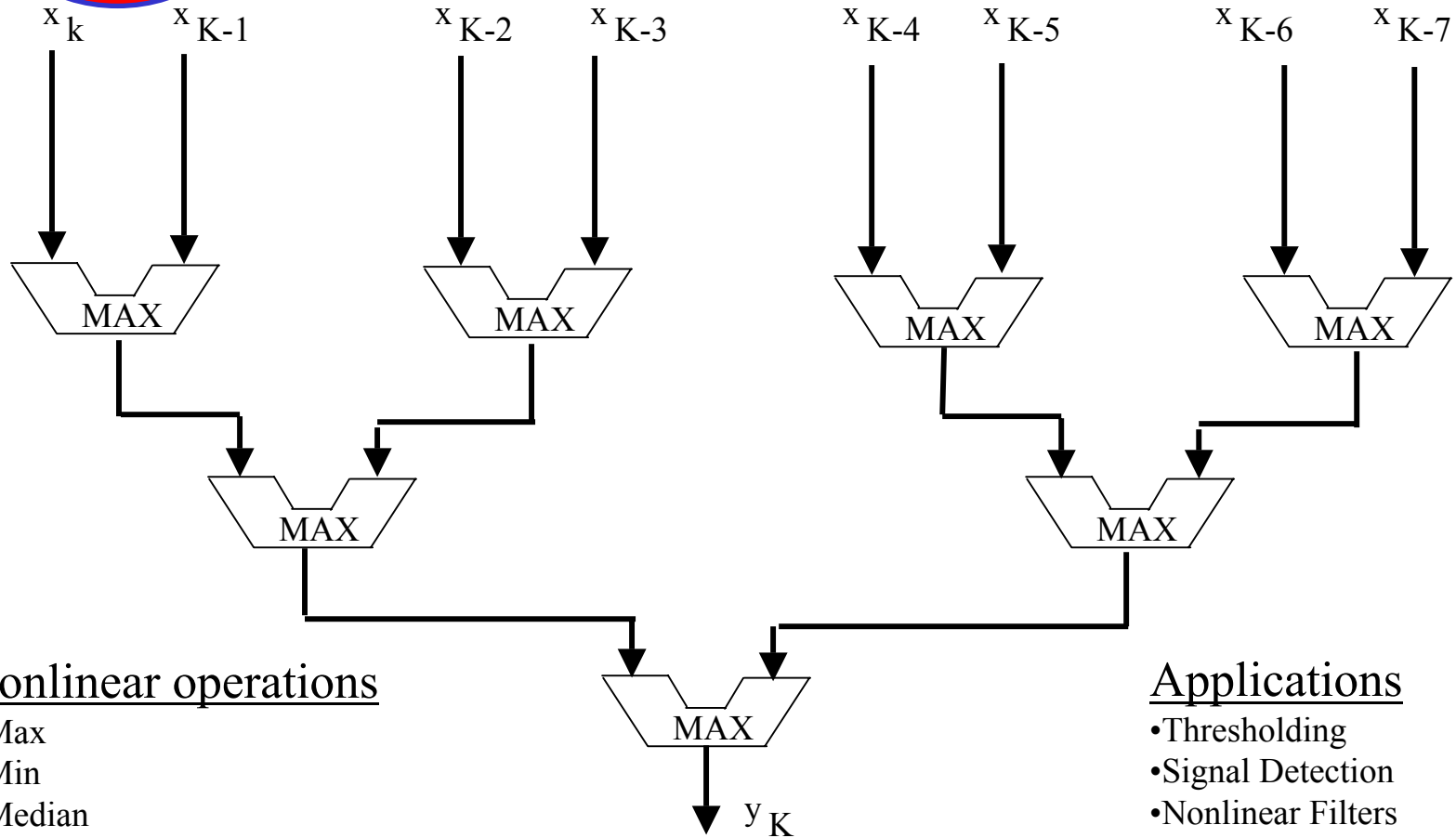
# 5x5 Convolution with 3 Line Buffers



- Partition convolution into two RDPP chips.
- One pixel overlap in the internal line buffers.



# Nonlinear Operations

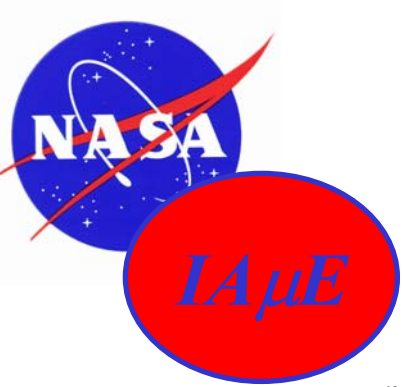


## Nonlinear operations

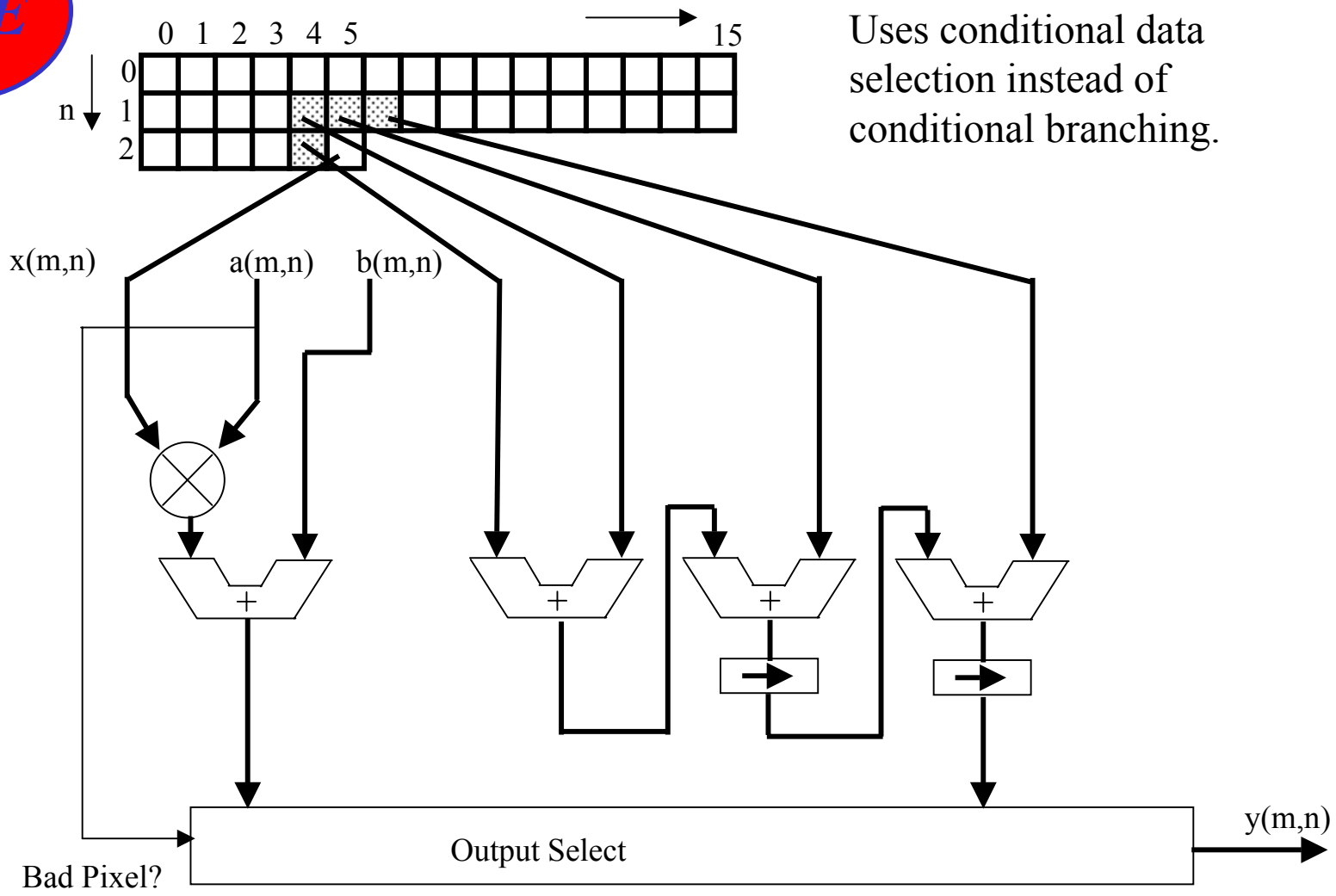
- Max
- Min
- Median
- Sort

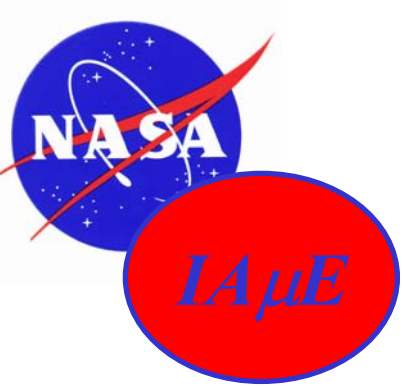
## Applications

- Thresholding
- Signal Detection
- Nonlinear Filters
- Mathematical Morphology

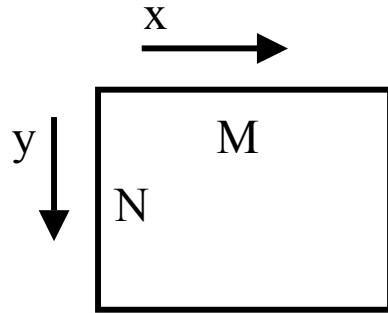


# Sensor Readout Correction





# Nested Loop Counter



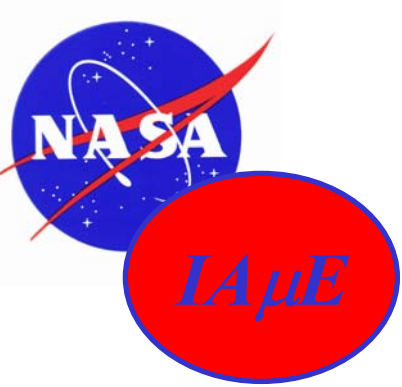
Useful to keep track of image coordinates for correlation & tracking

C-pseudocode:

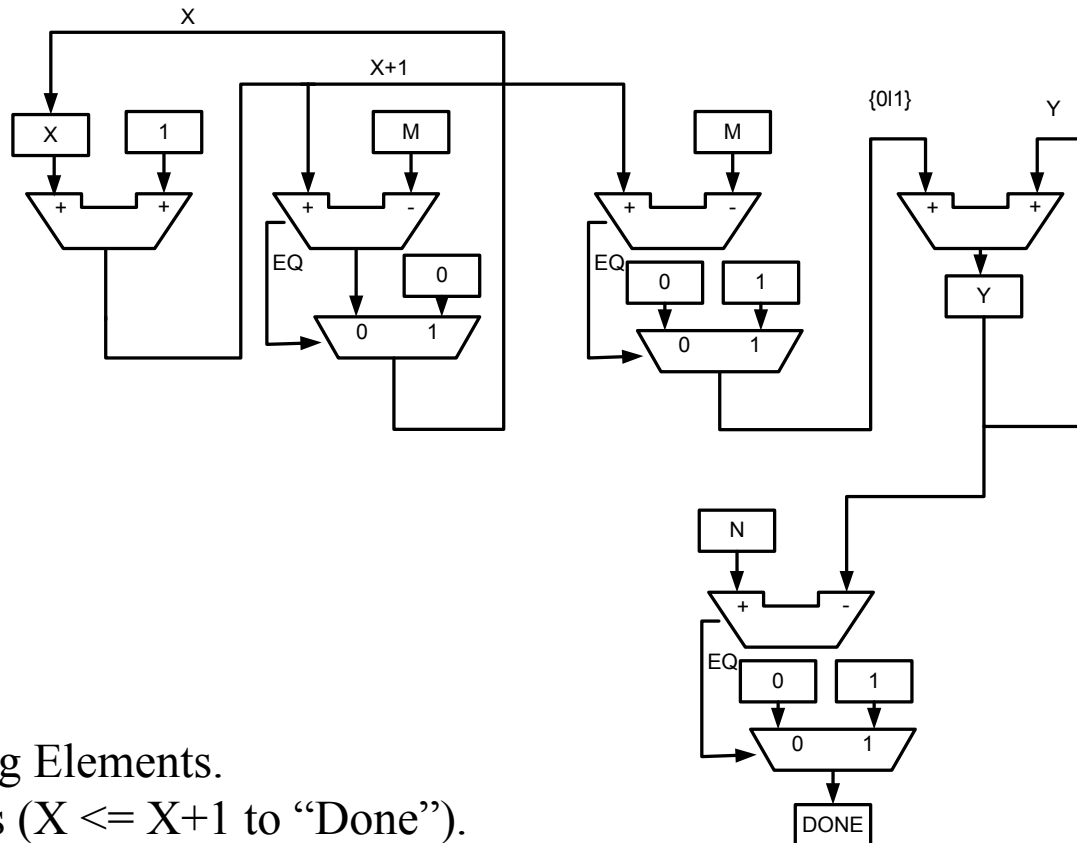
```
int x=0;
int y=0;
while (y < N) {
    while (x < M) {
        // do something
        x = x+1;
    }
    y = y+1;
}
```

RDPP data flow notation  
(from VHDL)

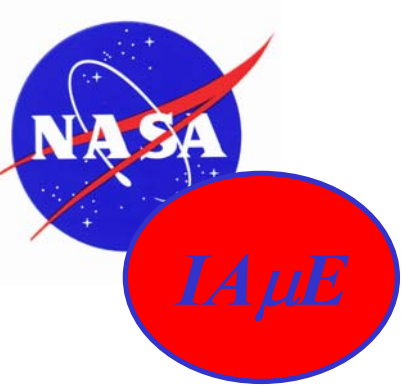
```
define false = 0, true = 1;
begin init;
    x <= 0; y <= 0;
    done <= false;
end init;
begin pipe; // three concurrent "ifs"
    if x+1 = M then
        x <= 0;
    else
        x <= x+1;
    end if;
    if x+1 = M then
        y <= y+1;
    else
        y <= y + 0;
    end if;
    if y = N then
        done <= true
    else
        done <= false;
    end if;
end pipe;
```



# Pipelined Loop Counter



Requires 5 Processing Elements.  
 Max delay = 3 cycles ( $X \leq X+1$  to "Done").  
 Test for  $X=M$  is replicated, with different output.



# Dynamic Range Example: FTHSI

Fourier Transform Hyper-Spectral Image data conversion: Inverse DFT (symmetric form)

$$f(x) = \frac{1}{\sqrt{N}} \sum_{u=0}^{N-1} F(u) e^{\frac{j2\pi ux}{N}}$$

N=number of spectral bands (~256)

Worst case:  $f(0) = \frac{1}{\sqrt{N}} \sum_{u=0}^{N-1} F(u)$

Assume 12-bit data,  $F(u) = F(u)_{\max} = 2^{12}$

$$f(0) = \left( \frac{2^8}{2^4} \right) 2^{12} = 2^{16}$$

16 bits are sufficient.

24 bits would be comfortable.



# Software Development



# RDPP Design Software

Simulator software being developed for  
Sign/Integer/Fraction data format

- Written in C++ with Standard Template Library.
- Foundation for two principal RDPP software tools:  
RDPP Design Entry  
RDPP Simulator
- Required to evaluate effectiveness of RDPP architecture on challenge problems.
- Will simplify RDPP software design.



# Conclusion

- The RDPP is targeted to an ultra-low-power, radiation-tolerant CMOS process.
- ULP CMOS permits highly parallel architecture with little cost in power consumption.
- Aimed at high-data-rate, streaming applications aboard spacecraft.
- Synchronous data flow model for highly parallel operation.
- Dynamically create and execute data flow pipelines.
- Conditional data switching instead of conditional branching.
- Currently beginning creating VHDL models.