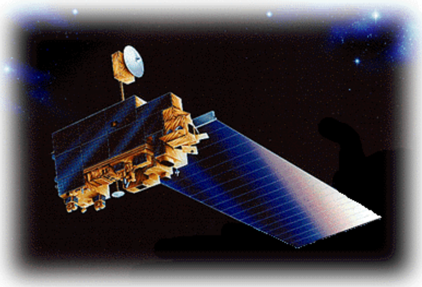
Adaptive Computing in NASA Multi-Spectral Image Processing

Mark L. Chang
Northwestern University
Evanston, IL
mchang@ece.nwu.edu

Scott A. Hauck
University of Washington
Seattle, WA
hauck@ee.washington.edu

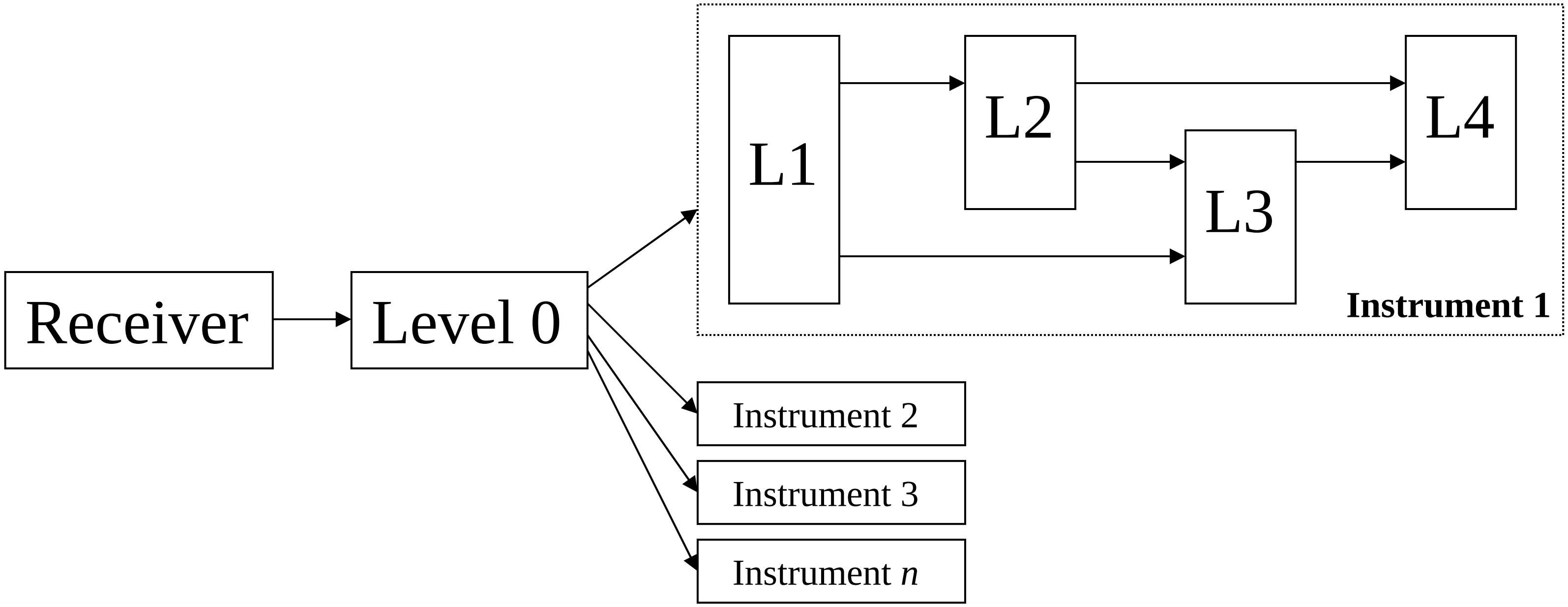
Background

- (1991) Initiative by NASA to study Earth as an environmental system—Earth Science Enterprise (ESE)
- (1999) Launch of the first Earth Observation System (EOS) satellite, *Terra*



The Data Flow

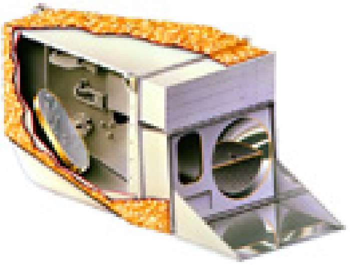
- **EOS divides telemetry processing into five levels with the following flow:**



The Processing Problem

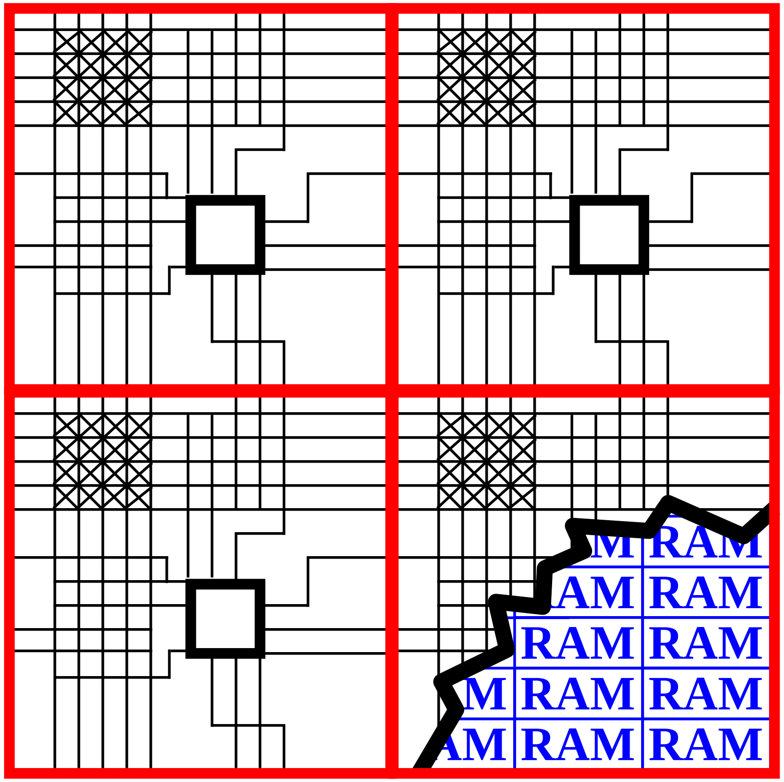
- **I/O intensive**
  - **Terra satellite generates ~918 Gbytes of data *per day***
  - **Current NASA-supported data holdings total ~125,000 Gbytes**
  - **MODIS instrument accounts for over half the daily data and processing load**

MODIS
Instrument



Why Adaptive Computing?

- Instrument dependent processing
- Data products involve many different algorithms
- Algorithms often change over the lifetime of the instrument



MATCH Compiler

- **Current mappings are done by hand**
  - **Hardware description languages (Verilog, VHDL)**
  - **C program interface to adaptive compute engine**
  - **Requires low-level understanding of the architecture**
- **MATCH == MATlab Compiler for Heterogeneous computing systems**
  - **MATLAB codes compiled to a configurable computing system *automatically***
    - **Embedded processors, DSPs, and FPGAs**
  - **Performance goals**
    - **Within a factor of 2-4 of the best manual approach**
    - **Optimize performance under resource constraints**

MATCH Compiler Framework

- Parse MATLAB programs into intermediate representation
- Build data and control dependence graph
- Identify scopes for fine-grain, medium grain, and coarse grain parallelism
- Map operations to multiple FPGAs, multiple embedded processors and multiple DSP processors
- Automatic parallelization, scheduling, and mapping

MATCH Testbed

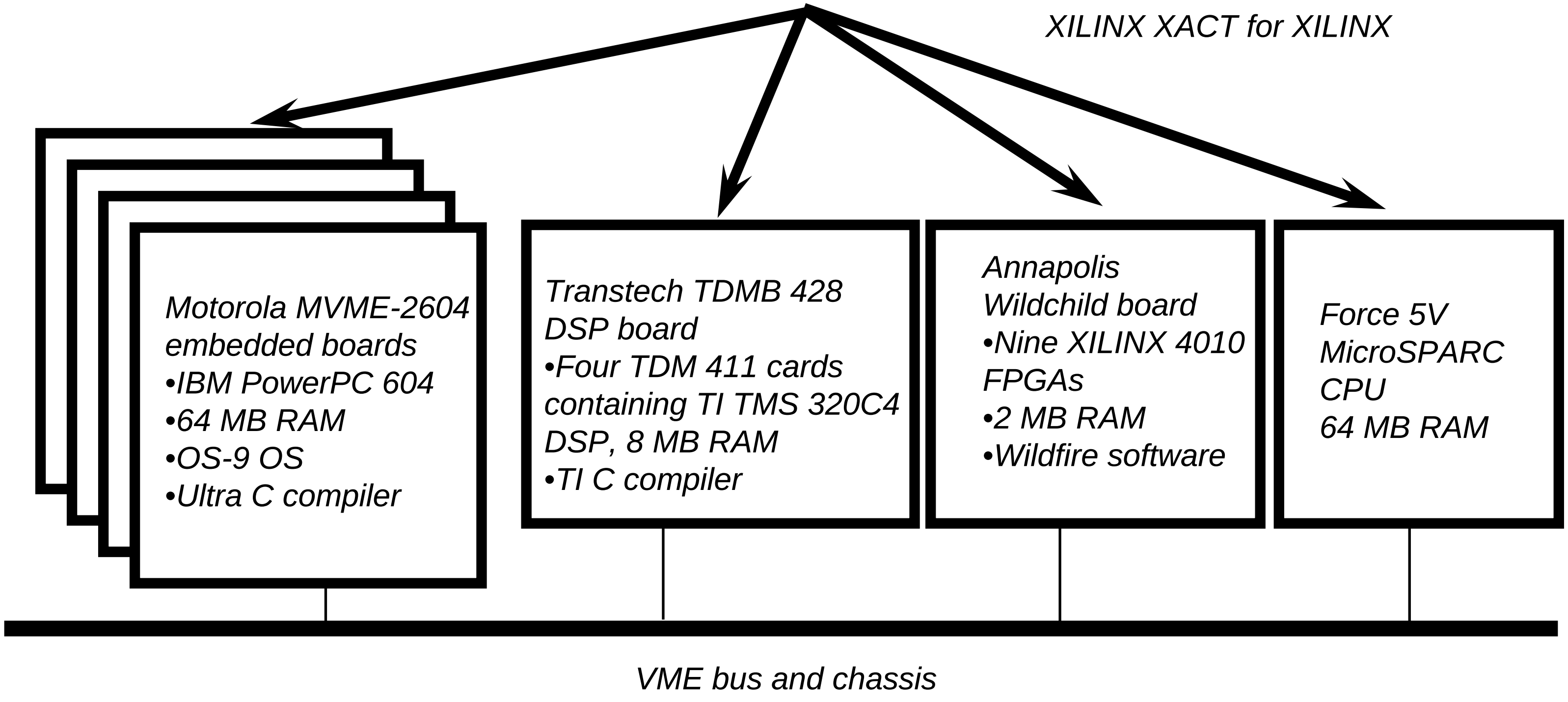
Development Environment:

SUN Solaris 2, HP HPUX and Windows

Ultra C/C++ for MVME

TI C for TMS320

XILINX XACT for XILINX

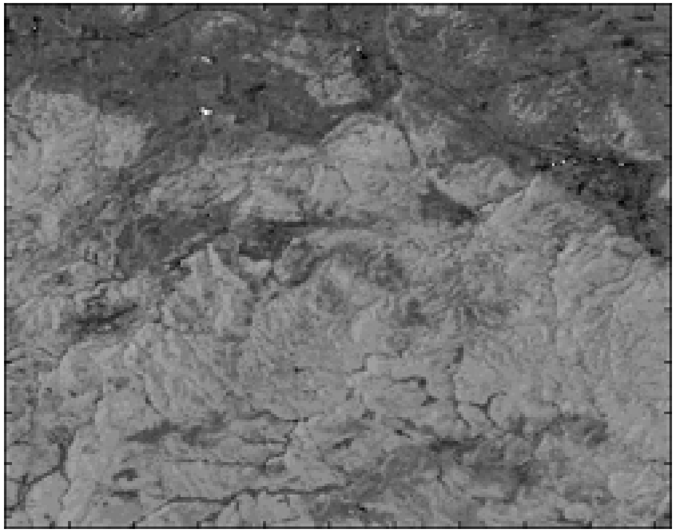


Motivation for MATCH

- NASA scientists prefer MATLAB
  - High-level language, good for prototyping and development
- NASA applications are well-suited to the MATCH project
  - Lots of image and signal processing applications
  - Same domain as users of embedded systems
  - High degree of data parallelism
  - Small degree of task parallelism
- NASA has an interest in adaptive technologies (ASDP)
- Will be a benchmark for the MATCH compiler

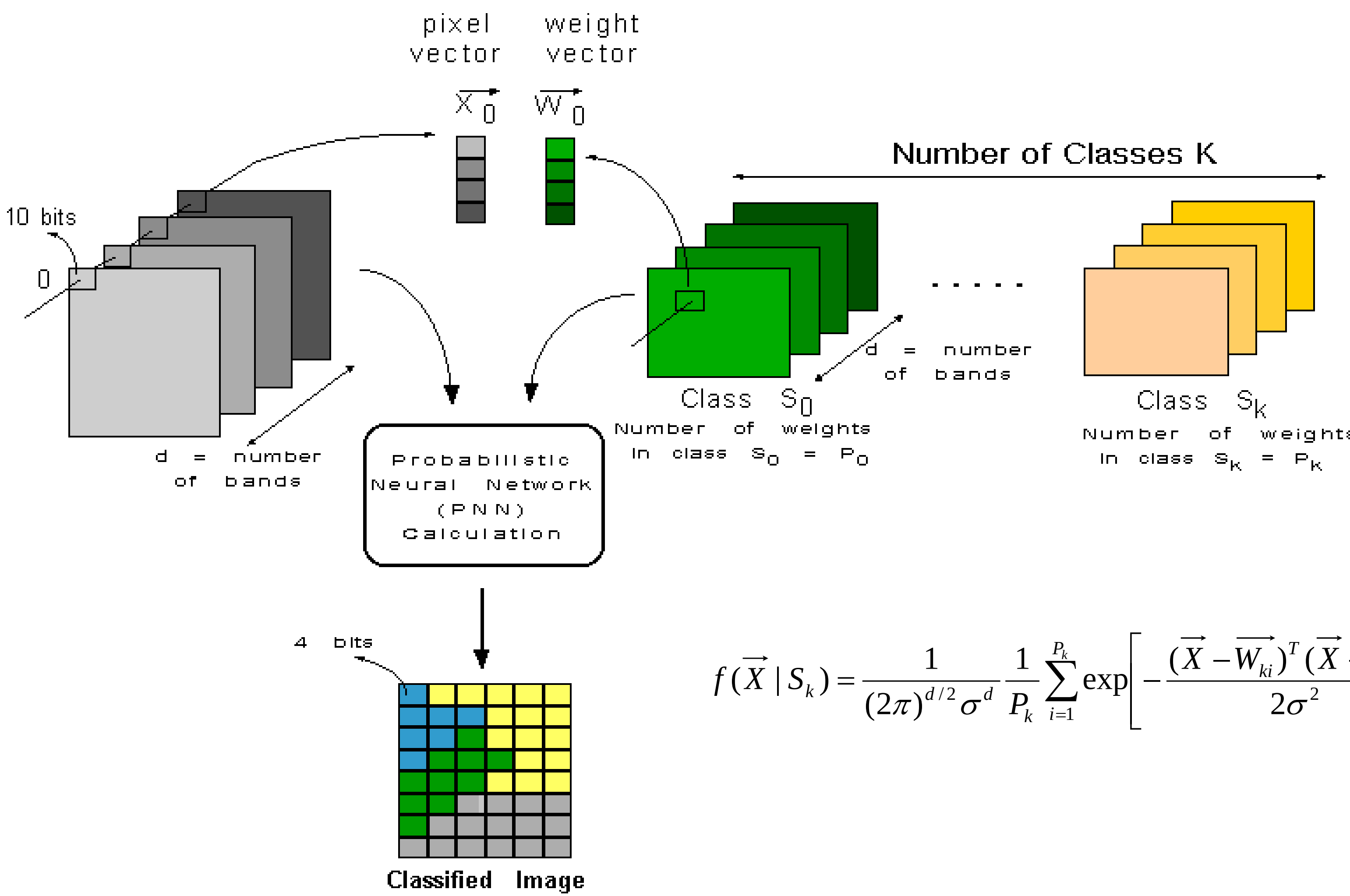
Multi-spectral Image Classification

- **Want to classify a multi-spectral image in order to make it more useful for analysis by humans**
  - **Used to determine type of terrain being represented**
  - **Similar to data compression**
  - **Similar to clustering analysis**



```
Pixel[000][000] = Forest
Pixel[123][123] = Urban
Pixel[255][212] = Tundra
Pixel[410][230] = Water

etc…
```

Multi-Spectral Classification



$$f(\vec{X} \mid S_k) = \frac{1}{(2\pi)^{d/2}\sigma^d}\frac{1}{P_k}\sum_{i=1}^{P_k}\exp\left[-\frac{(\vec{X}-\vec{W}_{ki})^T(\vec{X}-\vec{W}_{ki})}{2\sigma^2}\right]$$

MATLAB Iterative

```
for p=1:rows*cols
   % load pixel to process
   pixel = data( (p-1)*bands+1:p*bands );

   class_total = zeros(classes,1);
   class_sum   = zeros(classes,1);

   % class loop
   for c=1:classes

      class_total(c) = 0;
      class_sum(c) = 0;

      % weight loop
      for w=1:bands:pattern_size(c)*bands-bands
         weight = class(c,w:w+bands-1);
         class_sum(c) = exp( -(k2(c)*sum( (pixel-weight').^2 ))) + class_sum(c);
      end

      class_total(c) = class_sum(c) * k1(c);
   end
   results(p) = find( class_total == max( class_total ) )-1;
end
```

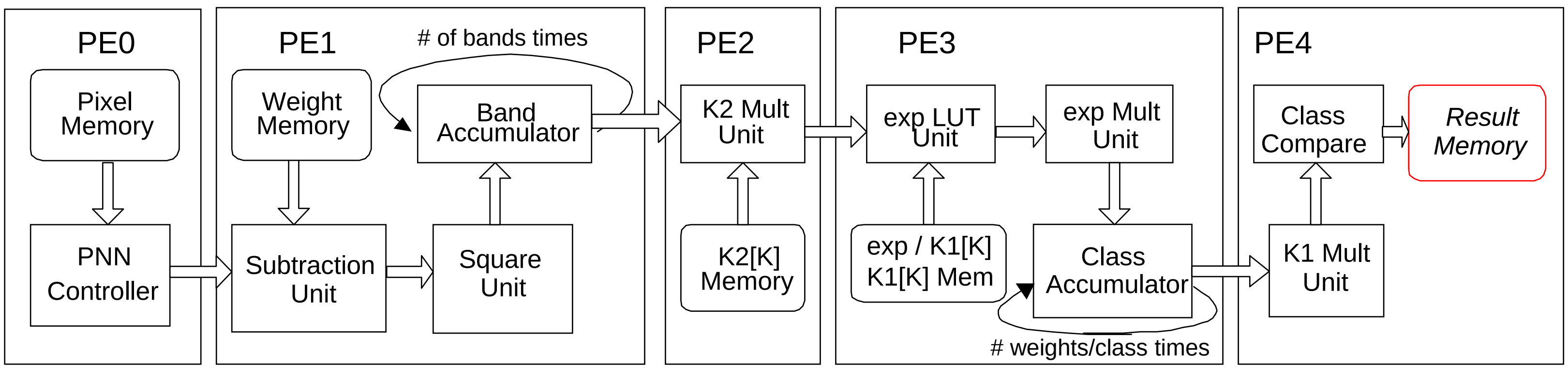
$$f(\vec{X} \mid S_k) = \frac{1}{(2\pi)^{d/2}\sigma^d} \frac{1}{P_k} \sum_{i=1}^{P_k} \exp\left[-\frac{(\vec{X} - \vec{W}_{ki})^T (\vec{X} - \vec{W}_{ki})}{2\sigma^2} \right]$$

MATLAB Vectorized

```
% reshape data
weights = reshape(class',bands,pattern_size(1),classes);

for p=1:rows*cols
   % load pixel to process
   pixel = data( (p-1)*bands+1:p*bands);

   % reshape pixel
   pixels = reshape(pixel(:,ones(1,patterns)),
                    bands,pattern_size(1),classes);

   % do calculation
   vec_res = k1(1).*sum(exp( -(k2(1).*sum((weights-pixels).^2)) ));
   vec_ans = find(vec_res==max(vec_res))-1;
   results(p) = vec_ans;
end
```

$$f(\vec{X} \mid S_k) = \frac{1}{(2\pi)^{d/2}\sigma^d} \frac{1}{P_k} \sum_{i=1}^{P_k} \exp\left[-\frac{(\vec{X} - \vec{W_{ki}})^T (\vec{X} - \vec{W_{ki}})}{2\sigma^2} \right]$$

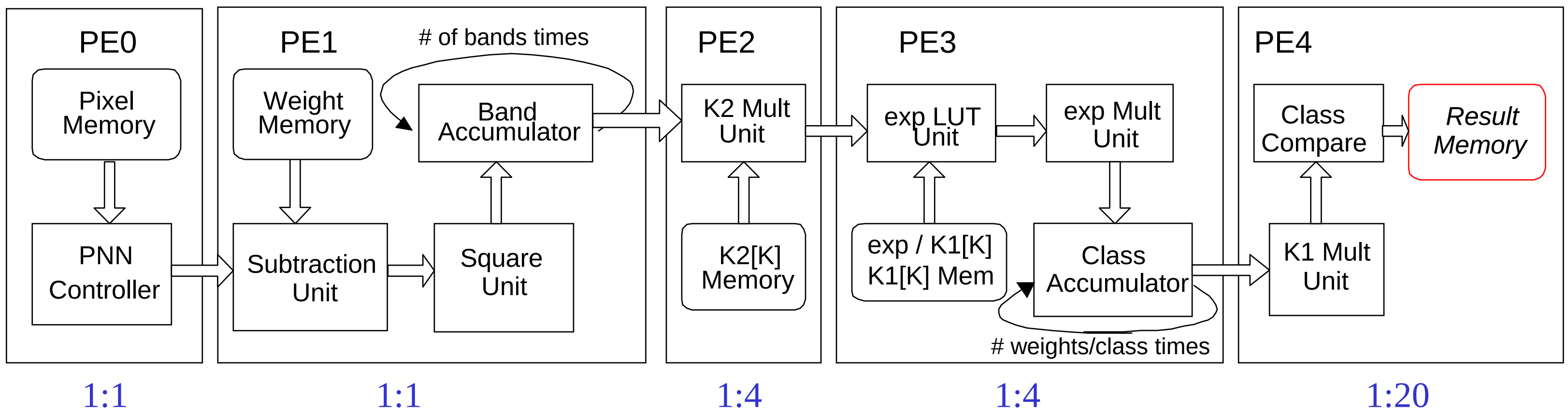
Initial FPGA Mapping



5% 67% 85% 82% 82%

$$f(\vec{X} \mid S_k) = \frac{1}{(2\pi)^{d/2}\sigma^d} \frac{1}{P_k} \sum_{i=1}^{P_k} \exp\left[-\frac{(\vec{X} - \vec{W}_{ki})^T (\vec{X} - \vec{W}_{ki})}{2\sigma^2} \right]$$

Improving the Mapping

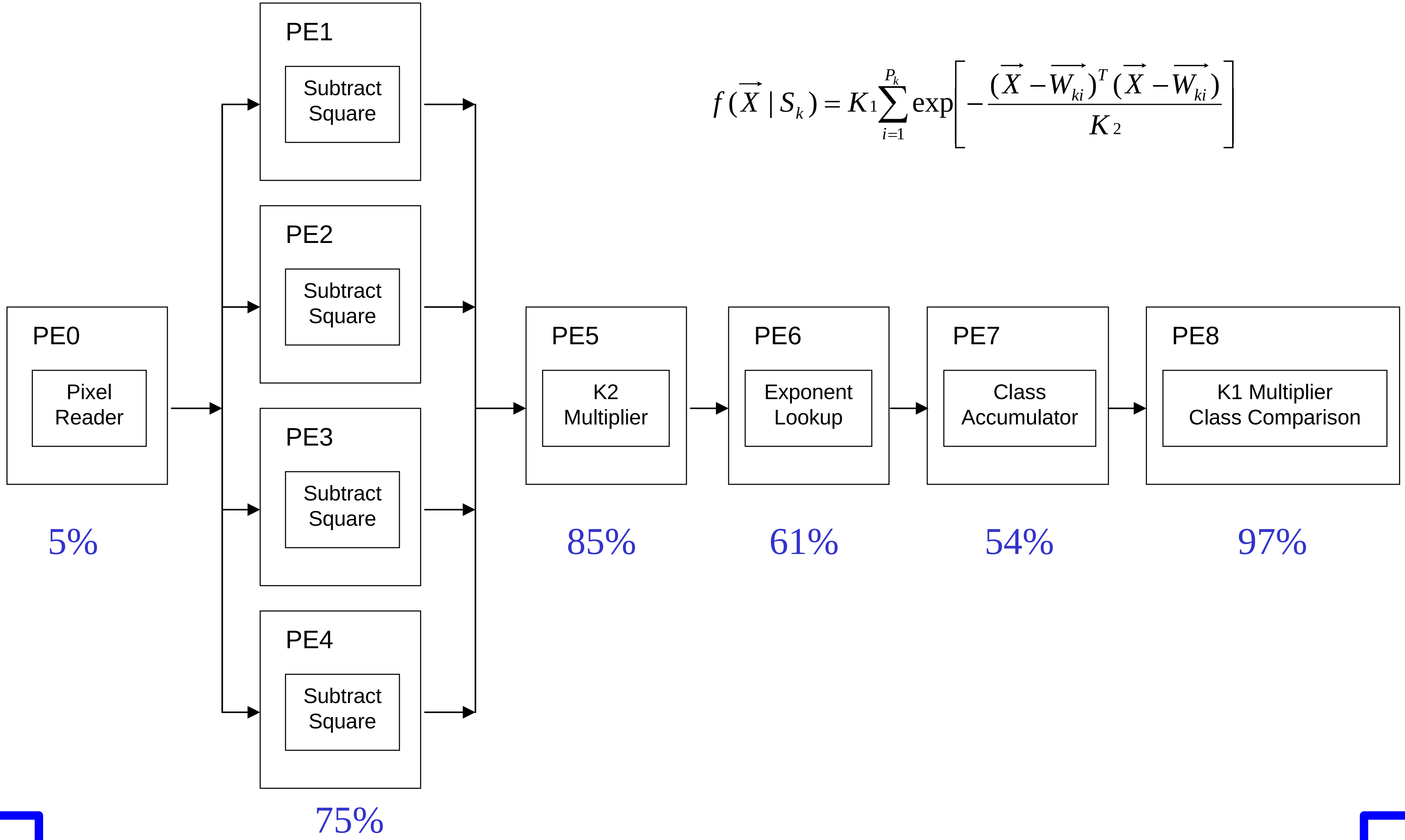


- **Improve speed of PNN**
  - **Utilize all eight processing elements**
  - **Time-multiplex low-rate functions**
  - **Vary precision of multipliers/lookups**

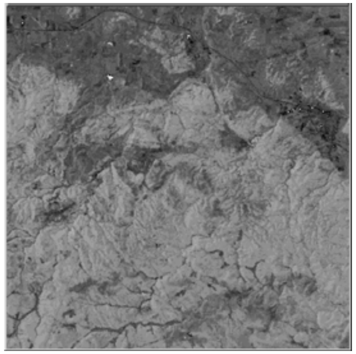
$$f(\vec{X} \mid S_k) = K_1 \sum_{i=1}^{P_k} \exp\left[-\frac{(\vec{X} - \vec{W}_{ki})^T (\vec{X} - \vec{W}_{ki})}{K_2} \right]$$

Optimized Mapping

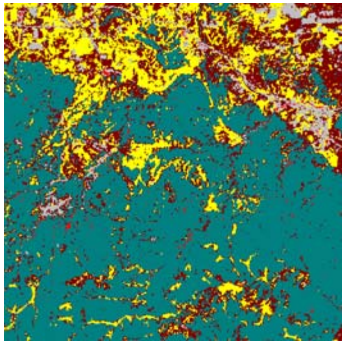
$$f(\vec{X} \mid S_k) = K_1 \sum_{i=1}^{P_k} \exp\left[-\frac{(\vec{X} - \vec{W}_{ki})^T (\vec{X} - \vec{W}_{ki})}{K_2} \right]$$



Results

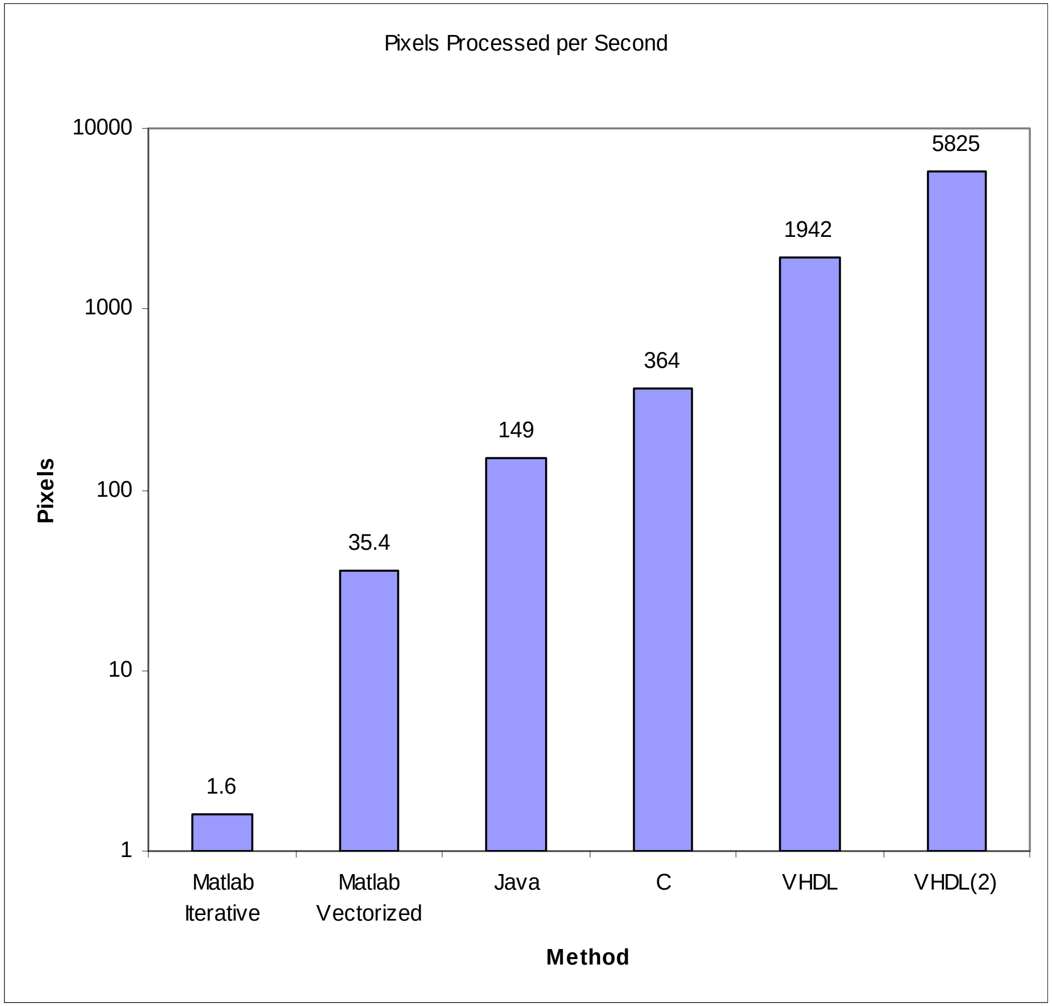


Raw Image Data



Processed Image

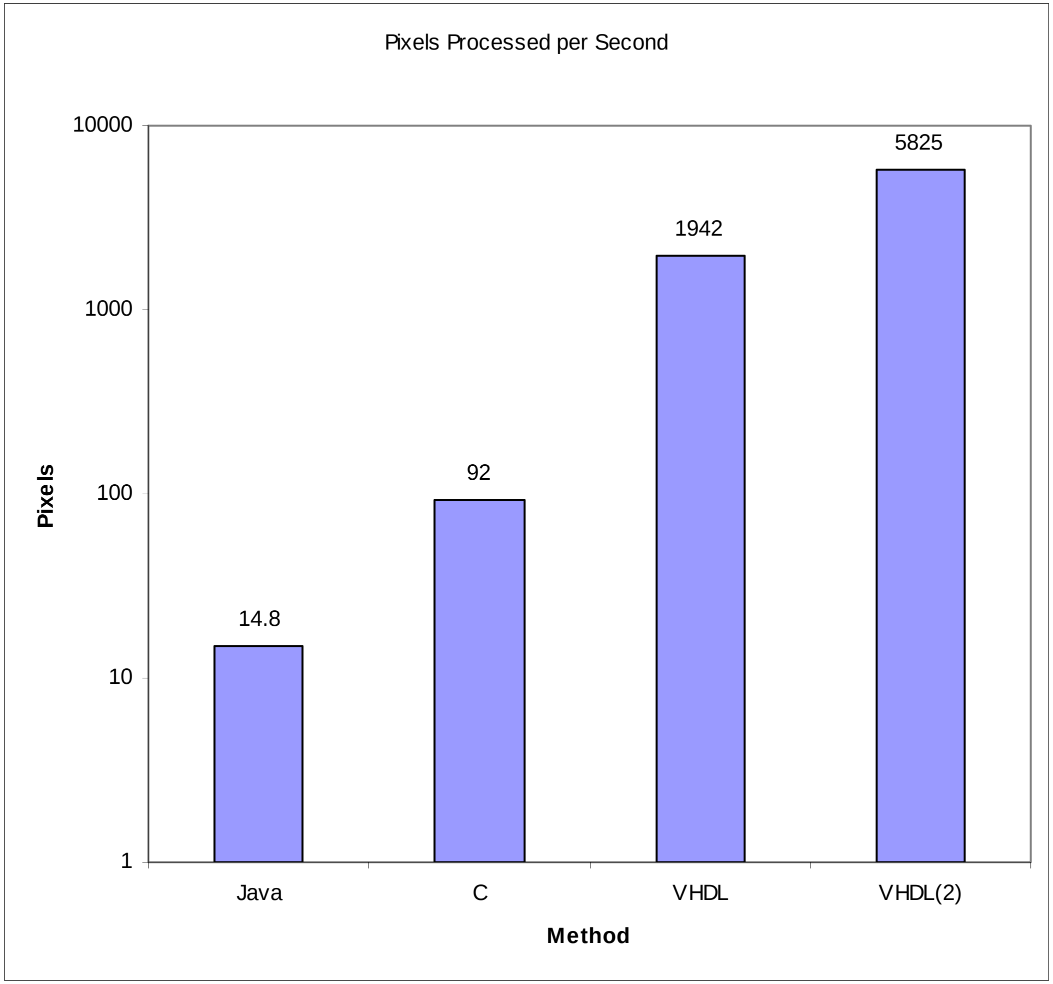
Reference: HP C180 Workstation



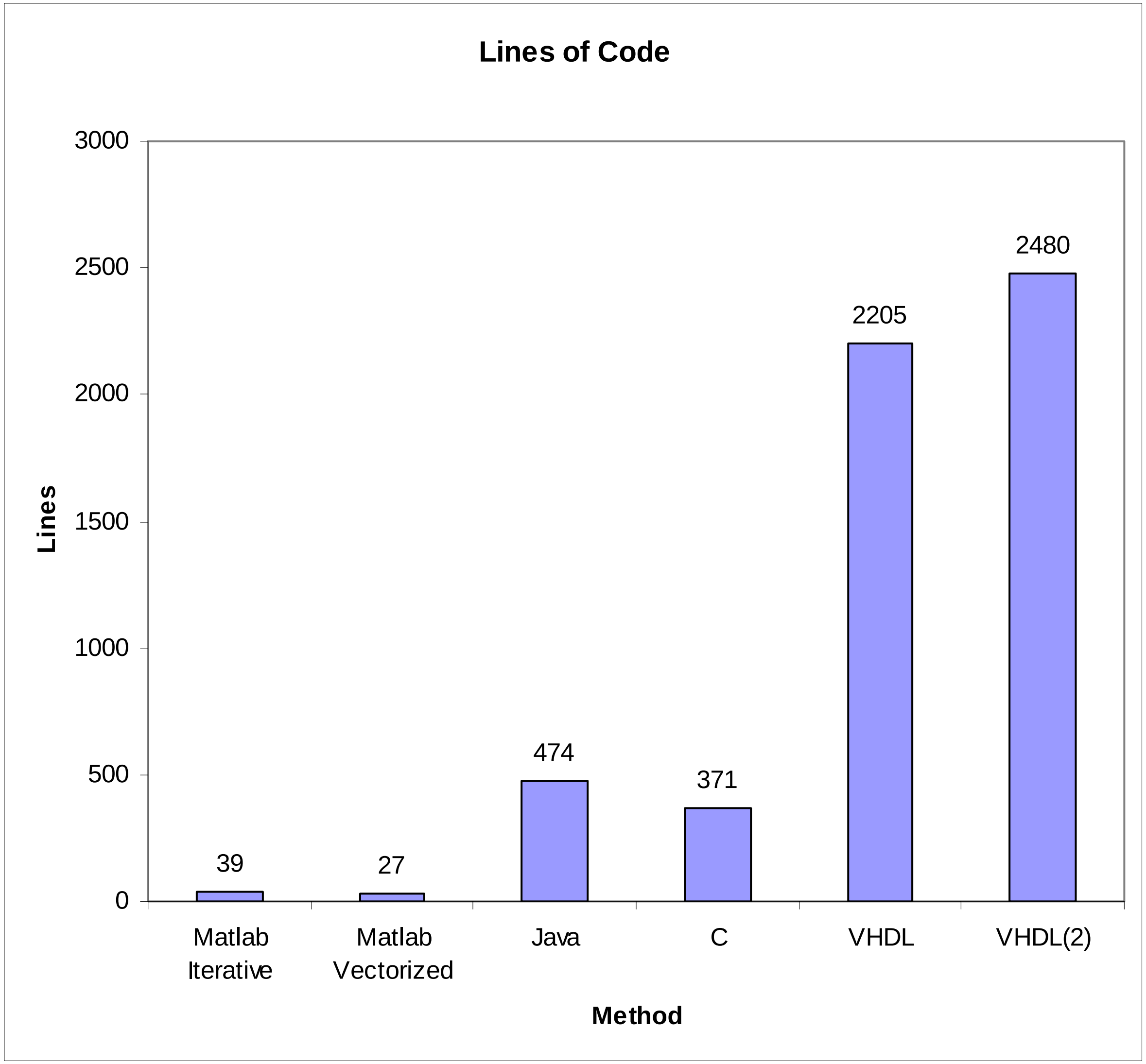
Results (Cont'd)

Reference:
MATCH Testbed

Force 5V
MicroSPARC CPU
64 MB RAM



Results (Cont'd)



Conclusions

- **NASA is interested in adaptive computing**
- **NASA has many candidate applications**
  - **High processing loads and I/O requirements**
  - **Applications are well-suited for acceleration using adaptive computing**
  - **Scientists will want to write in MATLAB rather than C+VHDL**
- **Good benchmarks for the MATCH compiler**
- **Will help identify functions and procedures necessary for real-world applications**