High Performance Embedded Computing Software Initiative (HPEC-SI)

Jeremy Kepner* (kepner@ll.mit.edu) MIT Lincoln Laboratory, Lexington, MA 02420

Abstract

The High Performance Embedded Computing Software Initiative (see www.hpec-si.org) is addressing the military need to advance the state of embedded software development tools, libraries, and methodologies to retain the nation’s military technology advantage in increasingly software-based systems. Key accomplishment include completion of the first demonstration and the development of the Parallel VSIPL++ standard. Currently the HPEC-SI effort is on track towards its goal of changing the state-of-the-practice in programming DoD HPEC SIP systems.

1 Introduction

The High Performance Embedded Computing Software Initiative (HPEC-SI) involves a partnership of industry, academia, and government organizations to foster software technology insertion demonstrations, to advance the development of existing standards, and to promote a unified computation/communication embedded software standard. The goal of the initiative is software portability: to enable “write-once/run-anywhere/run-any-size” for applications of high performance embedded computing (see [7, 4, 10, 8, 9, 18, 12]).

This paper gives a brief overview of the HPEC-SI program objectives, technical objectives and program plans. Detailed progress of the demonstration, development and applied research activities that are taking place within the HPEC-SI can be found in the HPEC2002[15, 20, 27], GOMAC2002[26, 5, 11, 21, 23], GOMAC2003[28, 6, 14, 17, 22], and other conferences[16, 13].

2 Program Objectives

HPEC-SI is organized around demonstrations, standards development and applied research. Each of these activities is overseen by a Working Group. The demonstrations team Prime contractors with FFRDC or academic partners to use currently defined standards, evaluate their performance, and report on how well their needs are being met. The first demonstration was with the Common Imagery Processor (CIP) and successfully showed the use of MPI communication standard ([1]) and the VSIPL computation standard ([2]) to achieve portability (while preserving performance) across shared servers and distributed memory embedded systems. The Development Working Group is extending the VSIPL standard to include parallel object-oriented software practices already prototyped by the research community. This effort is tightly coupled with military demonstrations, and provides the next generation of standards with direct feedback from the military user base. The Applied Research Working Group is also taking a longer term view to assess the potential impact of a variety of emerging technologies such as: fault tolerance and dynamic scheduling, self-optimization, and next generation high productivity languages.

3 Technical Objectives

The HPEC-SI program uses three principal metrics to measure the progress of its efforts:

- Portability (reduction in lines-of-code to change portSCALE to new system);
- Productivity (reduction in overall lines-of-code);
- Performance (computation and communication benchmarks).

Traditionally, it has always been possible to improve in two of the above areas while sacrificing the third. HPEC-SI aims to improve quantitatively in all three areas.

*This work is sponsored by the High Performance Computing Modernization Office, under Air Force Contract F19628-00-C-0002. Opinions, interpretations, conclusions and recommendations are those of the author and are not necessarily endorsed by the United States Government.
<table>
<thead>
<tr>
<th>1. REPORT DATE</th>
<th>2. REPORT TYPE</th>
<th>3. DATES COVERED</th>
</tr>
</thead>
<tbody>
<tr>
<td>20 AUG 2004</td>
<td>N/A</td>
<td>-</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>4. TITLE AND SUBTITLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>High Performance Embedded Computing Software Initiative (HPEC-SI)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>6. AUTHOR(S)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIT Lincoln Laboratory, Lexington, MA 02420</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>12. DISTRIBUTION/AVAILABILITY STATEMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approved for public release, distribution unlimited</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>13. SUPPLEMENTARY NOTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>See also ADM001694, HPEC-6-Vol 1 ESC-TR-2003-081; High Performance Embedded Computing (HPEC) Workshop (7th)., The original document contains color images.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>16. SECURITY CLASSIFICATION OF:</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. REPORT unclassified</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>17. LIMITATION OF ABSTRACT</th>
<th>18. NUMBER OF PAGES</th>
<th>19a. NAME OF RESPONSIBLE PERSON</th>
</tr>
</thead>
<tbody>
<tr>
<td>UU</td>
<td>34</td>
<td></td>
</tr>
</tbody>
</table>
HPEC-SI expects to achieve at least a 3x reduction in the number code changes necessary to port an application across computing platforms. This improvement will primarily be achieved through the use and enhancement of open software standards (MPI and VSIPL) that will insulate applications from the details of the underlying hardware. An equivalent reduction in code changes will also be seen when porting from one size of platform to another. This will be achieved by the development of a unified computation and computation standard (Parallel VSIPL) which will allow applications to be moved from a computer with N processors to a computer with M processors with minimal code changes.

HPEC-SI expects to achieve a 3x reduction in the total number of lines of code necessary to implement an application. This productivity improvement will be primarily be through the use of higher level object oriented languages (e.g. C++) as well as a unified computation and communication library which will abstract away many of code intensive details of writing a parallel program.

HPEC-SI expects to achieve a 1.5x increase in performance over existing approaches on some computation and communication benchmarks. This is primarily due to an increased level of abstraction which allows the increased use of “early binding” in the application, in the library and in the compiler. [Early binding is the process of building data structures in advance that increase performance at runtime.]

4 Summary

The current achievements of HPEC-SI include the successful utilization of the Vector Signal and Image Processing Library (VSIPL) and the Message Passing Interface to demonstrate a tactical synthetic aperture radar (SAR) code running without modifications and at high performance on parallel embedded, server and cluster systems. HPEC-SI is also creating the first parallel object oriented computation standard by adding these extensions to the VSIPL standard. The parallel VSIPL++ standard will allow high performance parallel signal and image processing applications to take advantage of the increased productivity offered by object oriented program as well as the performance advantages found using advanced expression template technology. The draft object oriented specification and reference code are both available on the HPEC-SI website and are being tested by a variety of early adopters. Finally, HPEC-SI is evaluating advanced software technologies such as fault tolerance and the use of higher level languages to determine which aspects are ready for future standardization. Combined, all of these efforts are successfully changing the state-of-the-practice in programming DoD HPEC SIP systems. Critical to this effort has been the availability of a wide variety of HPCMO systems (Mercury, Sky, SGI, Compaq, IBM, Linux, and FPGA) that has allowed the testing and demonstration of advanced software technologies for DoD signal and image processing applications.

References


High Performance Embedded Computing Software Initiative (HPEC-SI)

Dr. Jeremy Kepner / Lincoln Laboratory

This work is sponsored by the Department of Defense under Air Force Contract F19628-00-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the author and are not necessarily endorsed by the United States Government.
Outline

- Introduction
- Goals
- Program Structure
- Demonstration
- Development
- Applied Research
- Future Challenges
- Summary
Overview - High Performance Embedded Computing (HPEC) Initiative

**DARPA**

Applied Research

Development

Demonstration

**HPEC Software Initiative**

Common Imagery Processor (CIP)

Shared memory server

Embedded multi-processor

Challenge: Transition advanced software technology and practices into major defense acquisition programs

**Enhanced Tactical Radar Correlator (ETRAC)**
Why Is DoD Concerned with Embedded Software?

- COTS acquisition practices have shifted the burden from “point design” hardware to “point design” software.
- Software costs for embedded systems could be reduced by one-third with improved programming models, methodologies, and standards.

Source: “HPEC Market Study” March 2001

Estimated DoD expenditures for embedded signal and image processing hardware and software ($B)
Issues with Current HPEC Development
Inadequacy of Software Practices & Standards

• High Performance Embedded Computing pervasive through DoD applications
  – Airborne Radar Insertion program
    85% software rewrite for each hardware platform
  – Missile common processor
    Processor board costs < $100k
    Software development costs > $100M
  – Torpedo upgrade
    Two software re-writes required after changes in hardware design

Today – Embedded Software Is:
• Not portable
• Not scalable
• Difficult to develop
• Expensive to maintain
Evolution of Software Support Towards “Write Once, Run Anywhere/Anysize”

- Application software has traditionally been tied to the hardware

- Many acquisition programs are developing stove-piped middleware “standards”

- Open software standards can provide portability, performance, and productivity benefits

- Support “Write Once, Run Anywhere/Anysize”
Quantitative Goals & Impact

Program Goals

• Develop and integrate software technologies for embedded parallel systems to address portability, productivity, and performance
• Engage acquisition community to promote technology insertion
• Deliver quantifiable benefits

Portability: reduction in lines-of-code to change port/scale to new system
Productivity: reduction in overall lines-of-code
Performance: computation and communication benchmarks
Organization

Technical Advisory Board
- Dr. Rich Linderman AFRL
- Dr. Richard Games MITRE
- Mr. John Grosh OSD
- Mr. Bob Graybill DARPA/ITO
- Dr. Keith Bromley SPAWAR
- Dr. Mark Richards GTRI
- Dr. Jeremy Kepner MIT/LL

Executive Committee
- Dr. Charles Holland PADUSD(S+T)

Government Lead
- Dr. Rich Linderman AFRL

Demonstration
- Dr. Keith Bromley SPAWAR
- Dr. Richard Games MITRE
- Dr. Jeremy Kepner, MIT/LL
- Mr. Brian Sroka MITRE
- Mr. Ron Williams MITRE

Development
- Dr. James Lebak MIT/LL
- Dr. Mark Richards GTRI
- Mr. Dan Campbell GTRI
- Mr. Ken Cain MERCURY
- Mr. Randy Judd SPAWAR

Applied Research
- Mr. Bob Bond MIT/LL
- Mr. Ken Flowers MERCURY
- Dr. Spaanenburg PENTUM
- Mr. Dennis Cottel SPAWAR
- Capt. Bergmann AFRL
- Dr. Tony Skjellum MPISoft

Advanced Research
- Mr. Bob Graybill DARPA

- Partnership with ODUSD(S&T), Government Labs, FFRDCs, Universities, Contractors, Vendors and DoD programs
- Over 100 participants from over 20 organizations
HPEC-SI Capability Phases

- Phase 1
  - Applied Research: Unified Comp/Comm Lib
  - Development: Object-Oriented Standards
  - Demonstration: Existing Standards
  - Demonstrate insertions into fielded systems (CIP)
    - Demonstrate 3x portability

- Phase 2
  - Applied Research: Fault tolerance
  - Development: Parallel VSIPL++
  - Demonstration: Object-Oriented Standards
  - High-level code abstraction (AEGIS)
    - Reduce code size 3x

- Phase 3
  - Applied Research: Hybrid Architectures
  - Development: Fault tolerance
  - Demonstration: Unified Comp/Comm Lib
  - Parallel VSIPL++

Unified embedded computation/communication standard
- Demonstrate scalability
Outline

- Introduction
- Demonstration
- Development
- Applied Research
- Future Challenges
- Summary
- Common Imagery Processor
- AEGIS BMD (planned)
Common Imagery Processor (CIP) is a cross-service component

Sample list of CIP modes:
- U-2 (ASARS-2, SYERS)
- F/A-18 ATARS (EO/IR/APG-73)
- LO HAE UAV (EO, SAR)
- System Manager

* CIP picture courtesy of Northrop Grumman Corporation
Common Imagery Processor
- Demonstration Overview -

- Demonstrate standards-based platform-independent CIP processing (ASARS-2)
- Assess performance of current COTS portability standards (MPI, VSIPL)
- Validate SW development productivity of emerging Data Reorganization Interface
- MITRE and Northrop Grumman

Single code base optimized for all high performance architectures provides future flexibility
Software Ports

Embedded Multicomputers
- CSPI - 500MHz PPC7410 (vendor loan)
- Mercury - 500MHz PPC7410 (vendor loan)
- Sky - 333MHz PPC7400 (vendor loan)
- Sky - 500MHz PPC7410 (vendor loan)

Mainstream Servers
- HP/COMPAQ ES40LP - 833-MHz Alpha ev6 (CIP hardware)
- HP/COMPAQ ES40 - 500-MHz Alpha ev6 (CIP hardware)
- SGI Origin 2000 - 250MHz R10k (CIP hardware)
- SGI Origin 3800 - 400MHz R12k (ARL MSRC)
- IBM 1.3GHz Power 4 (ARL MSRC)
- Generic LINUX Cluster
Portability: SLOC Comparison

![Bar Chart showing SLOCs and VSIPL SLOCs comparison]

- **~1% Increase**
- **~5% Increase**

- **Sequential**
- **VSIPL**
- **Shared Memory VSIPL**
- **DRI VSIPL**

**SLOCs** vs. **VSIPL SLOCs**
Application can now exploit many more processors, embedded processors (3x form factor advantage) and Linux clusters (3x cost advantage)
Form Factor Improvements

### Current Configuration
- **IOP**: 6U VME chassis (9 slots potentially available)
- **IFP**: HP/COMPAQ ES40LP

### Possible Configuration
- **IOP**: 6U VME chassis
- **IFP1**
- **IFP2**
- **6U VME**
- **IOP** could support 2 G4 IFPs
  - form factor reduction (x2)
- **6U VME** can support 5 G4 IFPs
  - processing capability increase (x2.5)
HPEC-SI Goals
1st Demo Achievements

Portability: **zero** code changes required
Productivity: DRI code 6x smaller vs MPI (est*)
Performance: 2x reduced cost or form factor

**Achieved Goal 3x Portability**
**Achieved* Goal 3x Productivity**
**Performance Goal 1.5x Achieved**
Outline

- Introduction
- Demonstration
- Development
  - Object Oriented (VSIPL++)
  - Parallel (||VSIPL++)
- Applied Research
- Future Challenges
- Summary
Emergence of Component Standards

HPEC Initiative - Builds on completed research and existing standards and libraries

Definitions
VSIPL = Vector, Signal, and Image Processing Library
||VSIPL++ = Parallel Object Oriented VSIPL
MPI = Message-passing interface
MPI/RT = MPI real-time
DRI = Data Re-org Interface
CORBA = Common Object Request Broker Architecture
HP-CORBA = High Performance CORBA
**BLAS zherk Routine**

- **BLAS** = Basic Linear Algebra Subprograms
- Hermitian matrix $M$: $\text{conj}(M) = M^t$
- $\text{zherk}$ performs a rank-$k$ update of Hermitian matrix $C$:
  \[
  C \leftarrow \alpha \ast A \ast \text{conj}(A)^t + \beta \ast C
  \]

- **VSIPL code**
  ```cpp
  A = vsip_cmcreate_d(10,15,VSIP_ROW,MEM_NONE);
  C = vsip_cmcreate_d(10,10,VSIP_ROW,MEM_NONE);
  tmp = vsip_cmcreate_d(10,10,VSIP_ROW,MEM_NONE);
  vsip_cmprodh_d(A,A,tmp); /* A*conj(A)^t */
  vsip_rscmmul_d(alpha,tmp,tmp); /* alpha*A*conj(A)^t */
  vsip_rscmmul_d(beta,C,C); /* beta*C */
  vsip_cmadd_d(tmp,C,C); /* alpha*A*conj(A)^t + beta*C */
  vsip_cblockdestroy(vsip_cmdestroy_d(tmp));
  vsip_cblockdestroy(vsip_cmdestroy_d(C));
  vsip_cblockdestroy(vsip_cmdestroy_d(A));
  ```

- **VSIPL++ code (also parallel)**
  ```cpp
  Matrix<complex<double> > A(10,15);
  Matrix<complex<double> > C(10,10);
  C = alpha * prodh(A,A) + beta * C;
  ```

**Sonar Example**
- **K-W Beamformer**
- Converted C VSIPL to VSIPL++
- 2.5x less SLOCs
PVL PowerPC AltiVec Experiments

Results

- Hand coded loop achieves good performance, but is problem specific and low level
- Optimized VSIPPL performs well for simple expressions, worse for more complex expressions
- PETE style array operators perform almost as well as the hand-coded loop and are general, can be composed, and are high-level

Software Technology

<table>
<thead>
<tr>
<th></th>
<th>AltiVec loop</th>
<th>VSIPPL (vendor optimized)</th>
<th>PETE with AltiVec</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>C</strong></td>
<td>- C</td>
<td>- C</td>
<td>- C++</td>
</tr>
<tr>
<td><strong>For loop</strong></td>
<td>- For loop</td>
<td>- Altivec aware VSIPRO Core Lite (<a href="http://www.mpi-softtech.com">www.mpi-softtech.com</a>)</td>
<td>- PETE operators</td>
</tr>
<tr>
<td><strong>Direct use of AltiVec extensions</strong></td>
<td>- Direct use of AltiVec extensions</td>
<td>- No multiply-add</td>
<td>- Indirect use of AltiVec extensions</td>
</tr>
<tr>
<td><strong>Assumes unit stride</strong></td>
<td>- Assumes unit stride</td>
<td>- Cannot assume unit stride</td>
<td>- Assumes unit stride</td>
</tr>
<tr>
<td><strong>Assumes vector alignment</strong></td>
<td>- Assumes vector alignment</td>
<td>- Cannot assume vector alignment</td>
<td>- Assumes vector alignment</td>
</tr>
</tbody>
</table>
Parallel Pipeline Mapping

Signal Processing Algorithm

Filter
\[ X_{OUT} = FIR(X_{IN}) \]

Beamform
\[ X_{OUT} = w \times X_{IN} \]

Detect
\[ X_{OUT} = |X_{IN}| > c \]

Mapping

Parallel Computer

- Data Parallel within stages
- Task/Pipeline Parallel across stages
Scalable Approach

#include <Vector.h>
#include <AddPvl.h>

void addVectors(aMap, bMap, cMap) {
    Vector< Complex<Float> > a('a', aMap, LENGTH);
    Vector< Complex<Float> > b('b', bMap, LENGTH);
    Vector< Complex<Float> > c('c', cMap, LENGTH);
    b = 1;
    c = 2;
    a=b+c;
}

Single Processor Mapping
A = B + C

Multi Processor Mapping
A = B + C

Lincoln Parallel Vector Library (PVL)
• Single processor and multi-processor code are the same
• Maps can be changed without changing software
• High level code is compact
Outline

- Introduction
- Demonstration
- Development
- Applied Research
- Future Challenges
- Summary

- Fault Tolerance
- Parallel Specification
- Hybrid Architectures (see SBR)
Dynamic Mapping for Fault Tolerance

- Switching processors is accomplished by switching maps
- No change to algorithm required
- Developing requirements for ||VSIPL++
Parallel Specification

Clutter Calculation (Linux Cluster)

- Matlab is the main specification language for signal processing
- pMatlab allows parallel specifications using same mapping constructs being developed for ||VSIPL++
Outline

• Introduction
• Demonstration
• Development
• Applied Research
• Future Challenges
• Summary
Optimal Mapping of Complex Algorithms

Application

- **Input:** $X_{IN}$
- **Low Pass Filter:** $X_{IN}$, FIR1, FIR2, $X_{OUT}$
  - $W_1$, $W_2$
- **Beamform:** $X_{IN}$, mult, $X_{OUT}$
  - $W_3$
- **Matched Filter:** $X_{IN}$, FFT, IFFT, $X_{OUT}$
  - $W_4$

Different Optimal Maps

- Workstation
- Intel Cluster
- PowerPC Cluster
- Embedded Board
- Embedded Multi-computer

Hardware

- **PowerPC Cluster**
- **Embedded Multi-computer**

**Note:**

- Need to automate process of mapping algorithm to hardware
HPEC-SI Future Challenges

End of 5 Year Plan

Phase 5

Applied Research: Higher Languages (Java?)

Development: Self-optimization

Demonstration: Hybrid Architectures

Phase 4

Applied Research: PCA/Self-optimization

Development: Hybrid Architectures

Demonstration: Fault tolerance

Phase 3

Applied Research: Hybrid Architectures

Development: Fault tolerance

Demonstration: Unified Comp/Comm Lib

Unified Comp/Comm Standard
• Demonstrate scalability

Fault Tolerant VSIPL

Demonstrate Fault Tolerance
• Increased reliability

Hybrid VSIPL

Portability across architectures
• RISC/FPGA Transparency

FT VSIPL

Parallel VSIPL++

www.hpec-si.org
Summary

- HPEC-SI Program on track toward changing software practice in DoD HPEC Signal and Image Processing
  - Outside funding obtained for DoD program specific activities (on top of core HPEC-SI effort)
  - 1st Demo completed; 2nd selected
  - Worlds first parallel, object oriented standard
  - Applied research into task/pipeline parallelism; fault tolerance; parallel specification

- Keys to success
  - Program Office Support: 5 Year Time horizon better match to DoD program development
  - Quantitative goals for portability, productivity and performance
  - Engineering community support
Web Links

High Performance Embedded Computing Workshop
http://www.ll.mit.edu/HPEC

High Performance Embedded Computing Software Initiative
http://www.hpec-si.org/

Vector, Signal, and Image Processing Library
http://www.vsipl.org/

MPI Software Technologies, Inc.
http://www.mpi-softtech.com/

Data Reorganization Initiative
http://www.data-re.org/

CodeSourcery, LLC
http://www.codesourcery.com/

MatlabMPI
http://www.ll.mit.edu/MatlabMPI