SLAAC
Systems Level Applications of Adaptive Computing

DARPA/ITO Adaptive Computing Systems PI Meeting
Santa Fe, New Mexico
September 28 - 30, 1998

Presented by:
Bob Parker
Deputy Director,
Information Sciences Institute
**SLAAC Objectives**

- Define a *system-level* open, distributed heterogeneous adaptive computing architecture
- Design, develop and evolve scalable reference platforms implementing the adaptive systems architecture
- Validate the approach by deploying reference platforms in multiple defense application domains
  - SAR ATR
  - Sonar Beamforming
  - IR ATR
  - Others
System Level Applications of Adaptive Computing

Utilizing Three Phases of Adaptive Computing Components
Large Current Generation FPGAs
Rapid Reconfigurable and/or Fine Grain FPGAs
Hybrid FPGAs

Integrating Multiple Constituent Technologies
Gigabit/Sec Networking
Modular Adaptive Compute Modules
Network Based Runtime Control Software
Algorithm Analysis/CompilationTools

Developing Reference Platforms
Flight Worthy Deployable System
Low Cost Researchers Kit

Lab Demo of an ACS implemented SAR ATR algorithm
First Generation of Reference Platforms
Embedded SAR ATR Demo of ACS HW (Clear, 1Mpixel/s, 6TT)

Embedded SAR ATR Demo (CC&D, 1Mpixel/s, 6TT)
Embedded SAR ATR Demo (CC&D, 10Mpixel/s, 6TT)

‘97 ‘98 ‘99 ‘00 ‘01

Significant reduction in power, weight, volume, and cost for several challenging DoD embedded applications
• Demonstrate 500x reduction in system volume on SAR/ATR ACS challenge
• Demonstrate First forward looking, 50,000 beam towed array on ACS sonar challenge

Team Members: USC/ISI (Lead), BYU, UCLA, Sandia National Labs, LANL, LM-GES
SLAAC Affiliates
SLAAC - Scalable Performance for Adaptive Computing Systems

• SLAAC combines the high performance computing elements of adaptive computing with scalable multicomputer networking
Taxonomy of SLAAC Applications

- Memory Access Rate
  - Sonar Beamforming
    - (14.4 G reads/s)
  - ECMA
    - (< 0.1s reconfiguration time)
- Reconfiguration Rate
- Computation Rate
  - SAR/ATR
    - (12 TeraOPS)
SLAAC’s Contributions to Application Performance

● Integrating new devices into the SLAAC system
  ■ New devices provide improved computation (e.g. Colt), memory access (e.g. Virtex), and reconfiguration (e.g. CSRC) rates

● Provides network scalability through 2-level multicomputing
  ■ Increases computation and number of memory ports available

● Parallel host library
  ■ Ability to manage scalability and reconfiguration
### SLAAC Team/Customer Interactions

**FY98**
- **SAR ATR (Sandia)**: FY97: 1Mpixel/s, 6 TC, CC&D 0%
- **Beamforming (NUWC)**
- **IR ATR (NVL)**
- **Wideband/HS (LANL)**
- **ECMA (LM/GES)**

**FY99**
- **Beamforming (NUWC)**
- **IR ATR (NVL)**
- **Wideband/HS (LANL)**
- **ECMA (LM/GES)**
- **SLAAC Team/Customer Interactions**
  - Lab Demo on SLAAC1
  - Bellatrix2 Demo
  - Concept Demo of ECMA on ACS
  - CSEDS Demo w/ AN/SPY-1 radar signal processor

**FY00**
- **Beamforming (NUWC)**
- **IR ATR (NVL)**
- **Wideband/HS (LANL)**
- **ECMA (LM/GES)**
- **SLAAC Team/Customer Interactions**
  - Lab Demo on SLAAC1‘
  - Bellatrix2 Demo
  - Concept Demo of ECMA on ACS
  - CSEDS Demo w/ AN/SPY-1 radar signal processor

**FY01**
- **Beamforming (NUWC)**
- **IR ATR (NVL)**
- **Wideband/HS (LANL)**
- **ECMA (LM/GES)**
- **SLAAC Team/Customer Interactions**
  - Lab Demo on SLAAC1‘
  - Bellatrix2 Demo
  - Concept Demo of ECMA on ACS
  - CSEDS Demo w/ AN/SPY-1 radar signal processor

<table>
<thead>
<tr>
<th>FY98</th>
<th>FY99</th>
<th>FY00</th>
<th>FY01</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SAR ATR (Sandia)</strong></td>
<td>FY97: 1Mpixel/s, 6 TC, CC&amp;D 0%</td>
<td>2Mpixel/s, 6 TC, CC&amp;D 30%</td>
<td>10Mpixel/s, 6 TC, CC&amp;D 30%</td>
</tr>
<tr>
<td><strong>Beamforming (NUWC)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>IR ATR (NVL)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Wideband/HS (LANL)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>ECMA (LM/GES)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**ISI**
- **Hardware**
  - SLAAC1 V.0 V.1
  - SLAAC2 V.0 V.1 V.2
  - System Layer V.0 V.1 V.2
  - Control Layer V.0 V.1 V.2
  - SLAAC1 Node Layer V.0 V.1
  - SLAAC2 Node Layer V.0 V.1
  - Wildforce Node Layer V.0 V.1

**BYU**
- **Runtime Control Library**
  - Beamforming algorithm development
  - 1st mapping to FPGA platform
  - Advanced BF algorithm development

**Sandia**
- **Beamforming algorithm development**
- **FOA Superquant Alg. Development**
  - V.0 V.1 V.2

**UCLA**
- **R0 Algorithm complete**
- **R0 Algorithm optimizations**

**LM/GES**
- **Specify ECMA Algorithms**
- **Algorithm Mapping ACS Concept demo Integration & Test**
- **FOA Alg Review 1 Review 2**

**SLAAC Team/Customer Interactions**
- **ECMA**
  - LM/GES Integration & Test
  - LM/GES Concept demo Integration & Test

**Research Objectives**
- **R0: ACS R0-R5<100ms**
  - AP Hill Site Demo
  - Bellatrix2 Demo
  - Concept Demo of ECMA on ACS
  - CSEDS Demo w/ AN/SPY-1 radar signal processor

**LM/GES**
- **Bellatrix2 Demo**
- **Concept Demo of ECMA on ACS**
- **CSEDS Demo w/ AN/SPY-1 radar signal processor**

**ISI**
- **SLAAC1 Board Optimizations**
- **SLAAC2 Board Optimizations**
- **System & Control Layer Optimizations**
- **Node Layer Optimizations**

**By-Products**
- **Advanced BF algorithm development**
- **Advanced FOA algorithm development**

**SLAAC Team/Customer Interactions**
- **FOA Superquant Alg. Development**
  - V.0 V.1 V.2

**ISI**
- **SLAAC1 Node Layer V.0 V.1**
- **SLAAC2 Node Layer V.0 V.1 V.2**
- **Wildforce Node Layer V.0 V.1**

**LM/GES**
- **Specify ECMA Algorithms**
- **Algorithm Mapping ACS Concept demo Integration & Test**
SLAAC Architecture
**SLAAC Programming Model**

- **System programmer uses** message passing to communicate between distributed modules.
- **Module developer uses** FIFOs and virtual channels to supply data to design.
- **The SLAAC runtime control library provides** both of these interfaces.
**Runtime Control Library**

- **System Layer (SL)**
  - Is a parallel MPI library directly called by the programmer. Manages all communication in the distributed system.

- **Control Layer (CL)**
  - Provides functionality common to all ACS boards. Manages node state.

- **Node Layer (NL)**
  - Provides low-level hardware control functionality for a specific board type.
Developers View of SLAAC

- **ISI Provides**
  - SLAAC1 board
  - SLAAC System Layer API
    - C parallel libraries
    - Standard compiler
    - Debugger
  - SLAAC VHDL Simulation Model
    - Synopsys compatible
    - Simulation and synthesis libraries

- **Application developers**
  - use SLAAC VHDL simulation to develop applications
  - provide programming environment as required
  - provides PC, OS, MYRINET
  - develop drivers/libraries for new boards
Research Reference Platform

- **RRP is network of ACS-accelerated workstations.**
  - Inexpensive readily available platform for ACS development.
  - Tracks performance advances in workstations and cluster computing.
- **ACS hardware is PCI based.**
- **OS is NT or Unix; ISI Runtime Control library**
- **Network is simple Ethernet or high speed such as Myrinet**

- Supports broad range of compute and I/O resources
  ⇒ Deployed at: VT, ISI
RRP Test Platform

- **Sixteen PCs**
  - P2-300
  - 256 MB DRAM
  - Sixteen WF4/4062XL
- **100baseT Network**
  - 24-port 100baseT Switch
- **Myricom Network**
  - 16-port Myricom Switch
- **MPI/GM/LANai**
Deployable Reference Platform

- **DRPs are embedded implementations of RRP architecture.**
  - ACS cards converted from PCI to PMC.
  - Source-code compatible with RRP.
  - Carriers provide increasing levels of compute density.
- **Simplest carrier is a cluster of single-board computers.**

- **ACS hardware is PMC-based.**
- **OS is VxWorks; ISI Runtime Control Library.**
- **Network is SAN.**
Deployable Reference Platform
JSTARS ATR Processor

- **PowerPC Multicomputer**
  - 13 Commercial Motorola VMEbus CPU boards
    - 200Mhz 603e PowerPC per board
    - 5.2 GFLOPS Peak
  - Commercial Myrinet High Speed Communications
    - 1.28Gbits/sec full duplex
    - Cross point topology

- **SHARC Multicomputer**
  - 4 Sanders HPSC processor boards
    - 8 33Mhz Analog Devices SHARC DSP processors per board
    - 3.2 GFLOPS Peak
  - Myrinet High Speed Communications
Source-Code Compatibility

- Scalable source-code compatibility for these platforms will simplify the transition from the research lab to the field.

Recompile to retarget and scale
SLAAC1 Board Architecture

- Full-sized 64-bit PCI card.
- Two Xilinx XC40150XV compute FPGAs and one XC4085 control FPGA.
- Ten 256Kx18 ZBT SSRAMs.
- 64-bit data-paths throughout.
- 100MHz. clock
Systolic Mode

- **Two QC64 mezzanine boards provide 300MB/sec systolic data path.**
  - X1, X2 can be used for systolic processing in-band
  - SLAAC1 board can down-sample data for host processing
Proposed
SLAAC2 PMC Board Architecture

- DUAL 64-bit PMC
- Four Xilinx XC40150XV compute FPGAs and one XC4085 control FPGA
- Eighteen 256Kx18 ZBT SSRAMs
- 64-bit data-paths throughout
- 100MHz clock
Network Carrier Alternatives
Two Slot Carrier

- COTS 6U VME Single Board Computer Host
- COTS PMC Myrinet Networking

- ACS technology as dual-wide PMC on second slot carrier
- No support for Myrinet-P0
**Network Carrier Alternatives**

**Single Slot Carrier**

- **COTS Network-Integrated**
  **6U VME Single Board Computer Host**
  - ACS technology as dual-wide PMC
  - Support for Myrinet-P0 interconnect

![Diagram of network carrier alternatives with labels for PPC, DRAM, PCI, LANai, SLAAC2, X1, X2, X3, X4, X0, and 6U VME connections.](image-url)
**SLAAC 1 Board Package**

- **Hardware**
  - SLAAC1 PCI board

- **Hardware Development Environment**
  - VHDL simulation of SLAAC1 board
  - Synthesis support files

- **Host Development Environment**
  - SLAAC1 Node Layer library
  - System Layer and Control Layer common to all hardware.

- **Host prototyping and debugging tools** (interactive debugger, performance monitors, boardscope, etc.)
SLAAC Board Schedule

- VHDL simulation of SLAAC1 board and constraints file for application timing analysis.
- Small prototype run (3-5) for smoke test, timing analysis, and firmware/software design.
- First run (12+) of SLAAC1 PCI board delivered with drivers and node library.
- First run of SLAAC2 PMC board delivered.
# Customer Applications Menu

<table>
<thead>
<tr>
<th>Tasks &amp; Technologies</th>
<th>Applications</th>
<th>Beamforming</th>
<th>SAR/ATR</th>
<th>Hyperspectral</th>
<th>IR/ATR</th>
<th>ECMA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(LANL)</td>
<td>(NUWC)</td>
<td>(Sandia)</td>
<td>(LANL)</td>
<td>(NVL)</td>
<td>(LM/GE)</td>
</tr>
<tr>
<td><strong>Form Factors</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCI: SLAAC1</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>(2 compute FPGAs)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PMC: SLAAC2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(4 compute FPGAs)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td><strong>Drivers</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VxWorks</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>LINUX</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>NT</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Algorithm Development/Mapping</strong></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td><strong>Runtime Control Library</strong> (system, control, node)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
ACS Module Developer Provides ACS Challenge User

- **Hardware configuration file for a given problem**
- **Compiler/Code with documentation**
  - to recreate exact configuration file (to make sure the compiler works in the ACS users environment)
  - so ACS user can create different configurations with current supported compiler parameters
  - so the ACS user can extend the compilers current capability if needed (grad students are not around forever, and neither will SLAAC)
- **Code with documentation on how to configure and run the design**