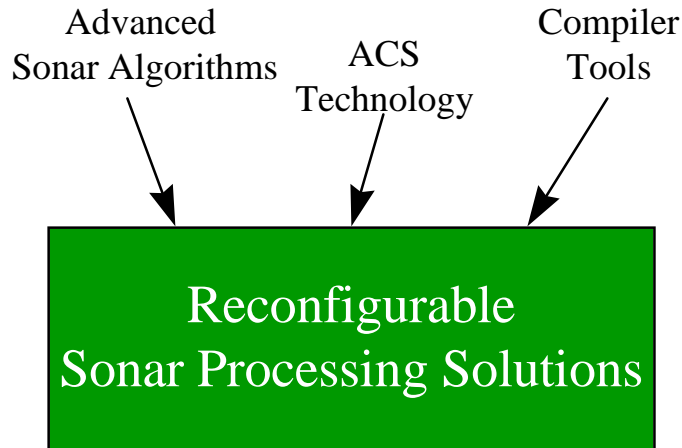

Sonar Processing and ACS: Feasibility Study Results

Brent Nelson
Brigham Young University

6 November 1997



ACS Technology for Sonar Processing



New Ideas:

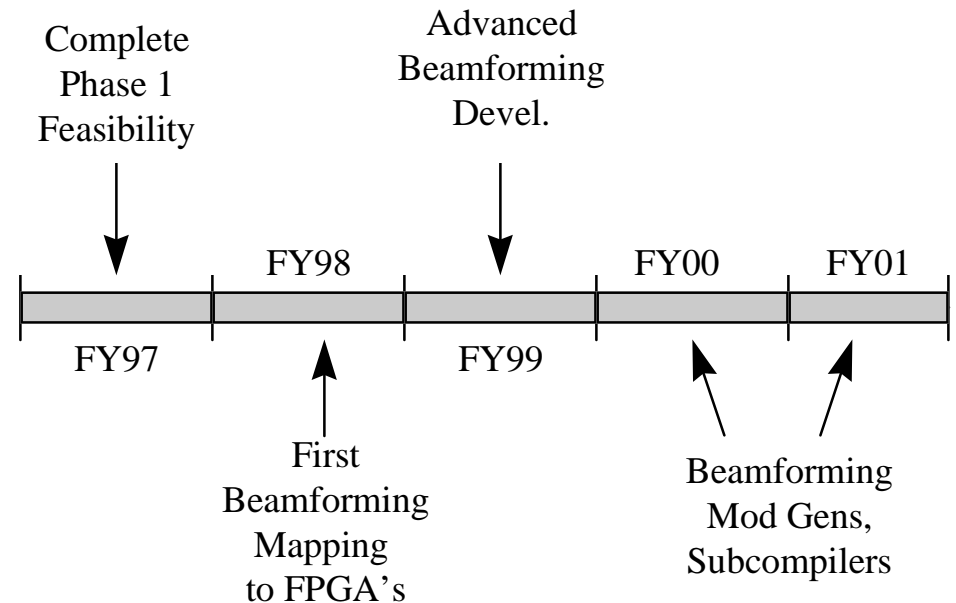
- Time-domain, frequency-domain, and matched field algorithms
- New ACS board architecture
- Automated compilation techniques

Impact:

- 5X to 10X cost/area/weight reduction
- Family of sonar processing architectures for ACS

Brigham Young University

Schedule:



Background - ACS For Sonar

- ◆ Application-driven focus at BYU
- ◆ Sonar:
 - Important challenge problem
- ◆ Investigation begun 12/96
 - in conjunction with Bob Bernecky (NUWC)
 - became part of SLAAC proposal

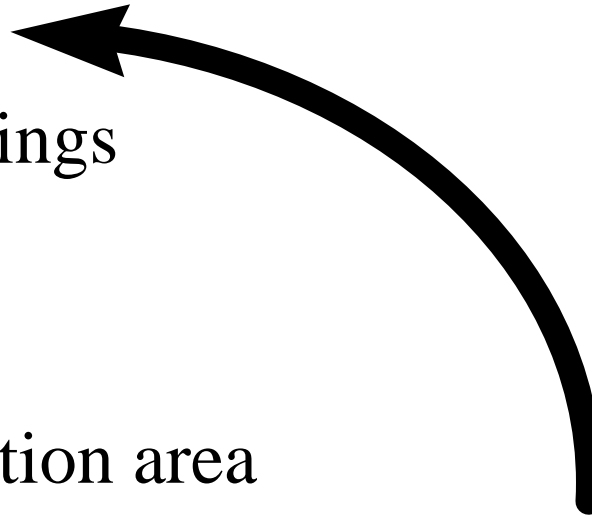
Sonar & SLAAC Baseline Effort

- ◆ BYU Focus is:

- CAD tools
- Algorithm mappings

- ◆ Sonar

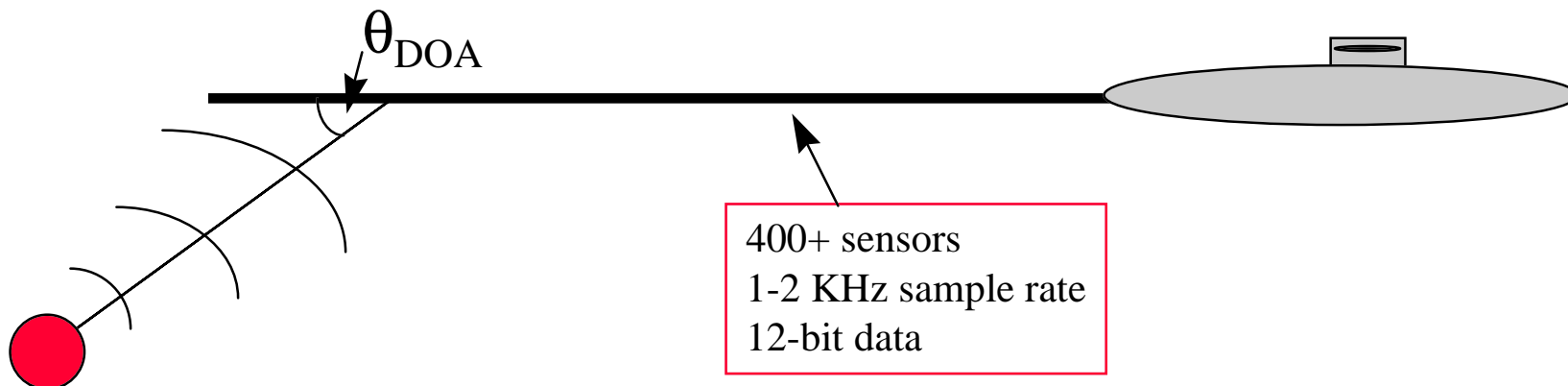
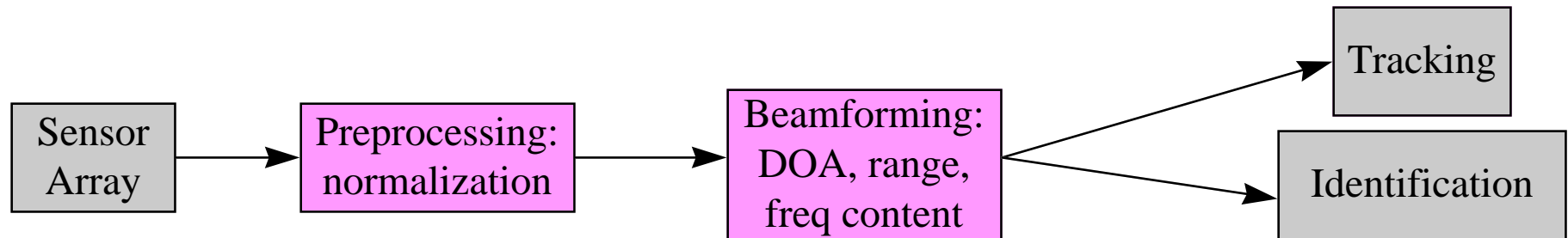
- Example application area
- Module generators and sub-compilers
- Small-scale laboratory demonstration of ACS



Part 1: Feasibility Study

- ◆ Determine suitability of ACS for sonar processing

Passive Sonar



Beamforming & ACS

◆ Data

- modest input data rates
- *data explosion*

◆ Arithmetic

- multiply-accumulates
- 12-24 bit
- complex numbers

Beamforming & ACS 2

◆ Memory

- memory access-limited
 - » performance \propto #-Memories

◆ Algorithms

- massively parallel/pipelined
- small PE's
- CORDIC
- sorting/searching

Feasibility Study Methodology

- ◆ Paper study
 - FLOPs
 - I/O bandwidth
 - memory architectures
 - board architectures
- ◆ VHDL coding, simulation, synthesis
- ◆ Download & execution

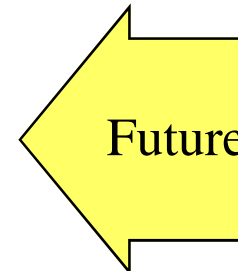
Algorithms Investigated

- ◆ Time-delay
- ◆ Frequency-domain

Non-adaptive

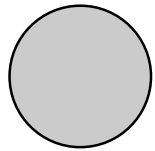


- ◆ Matched field
- ◆ Adaptive versions of above



ACS Mappings - Sensor Arrays

- ◆ Spherical Sensor Array (960 sensors)



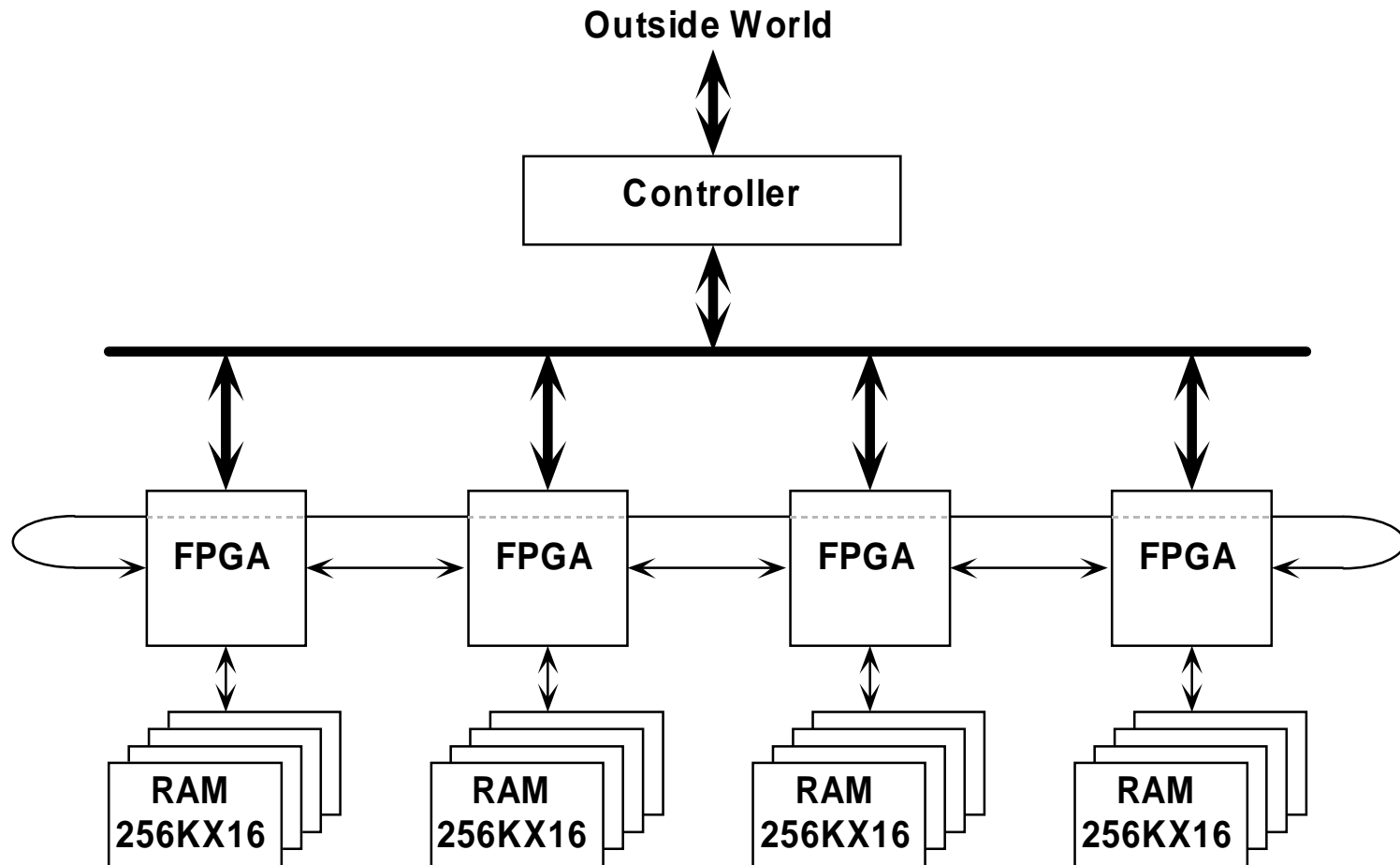
- 600 beams
- $f_s = 25\text{KHz}$, 12-bit samples

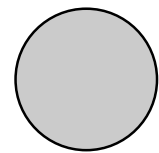
- ◆ TB-29 Towed Array (416 sensors)



- 416 sensors per aperture
- 13,000 beams
- $f_s = 1562.5\text{Hz}$, 12-bit samples

SLAAC4 - Target Platform





Time-Delay BF - Spherical Array

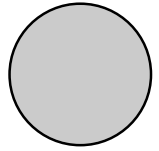
$$\text{beamResponse} = \sum_{s=0}^{\text{numSensors}} \text{shade}_s \times \text{delayedSensorSamples}$$

- 600 beams in 40μsec requires $4.8\text{B} \frac{\text{MACs}}{\text{sec}}$
- 96 PE's @ 50MHz (no communication)
- 4 PE's / FPGA & 4 FPGA's / board \Rightarrow 10 boards = 160 PE's
- 1,280 cycles + 336 cycles < 2,000 cycle budget

↑
Time for 160 PE's
to form 600 beams
(4 sets of 320 cycles)

↑
Time to load
new sensor data

↑
40μs/20ns clock cycle

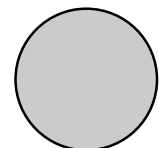


Spherical Time-Delay BF

- ◆ Much **symmetry** in problem

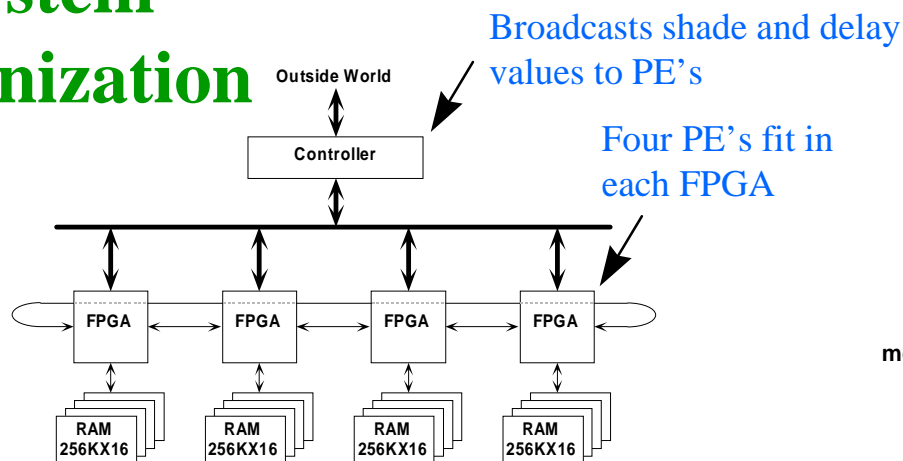
- multiple beams computed at once
- common shading coefficients and delay values
 - » SIMD operation
 - » use broadcast of this shared data

```
repeat {
  for (b=0;b<numBeams;b++) {
    power = 0;
    for (a=0;a<apertureSize;a++) {
      receive_broadcast_data(shade, delay);           /* Get shade and delay */
      datum = lookupData(delay, s);                 /* The only memory access */
      power = power + datum * shade;                /* The actual computation */
    }
    outputBeamResponse(power);
  }
} /* repeat forever */
```

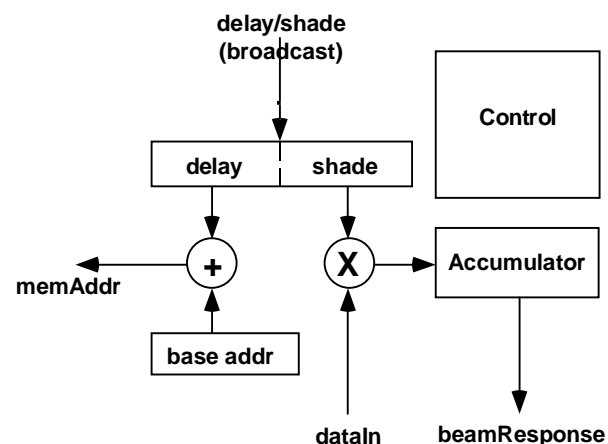


Spherical Time-Delay BF

System Organization



PE Architecture

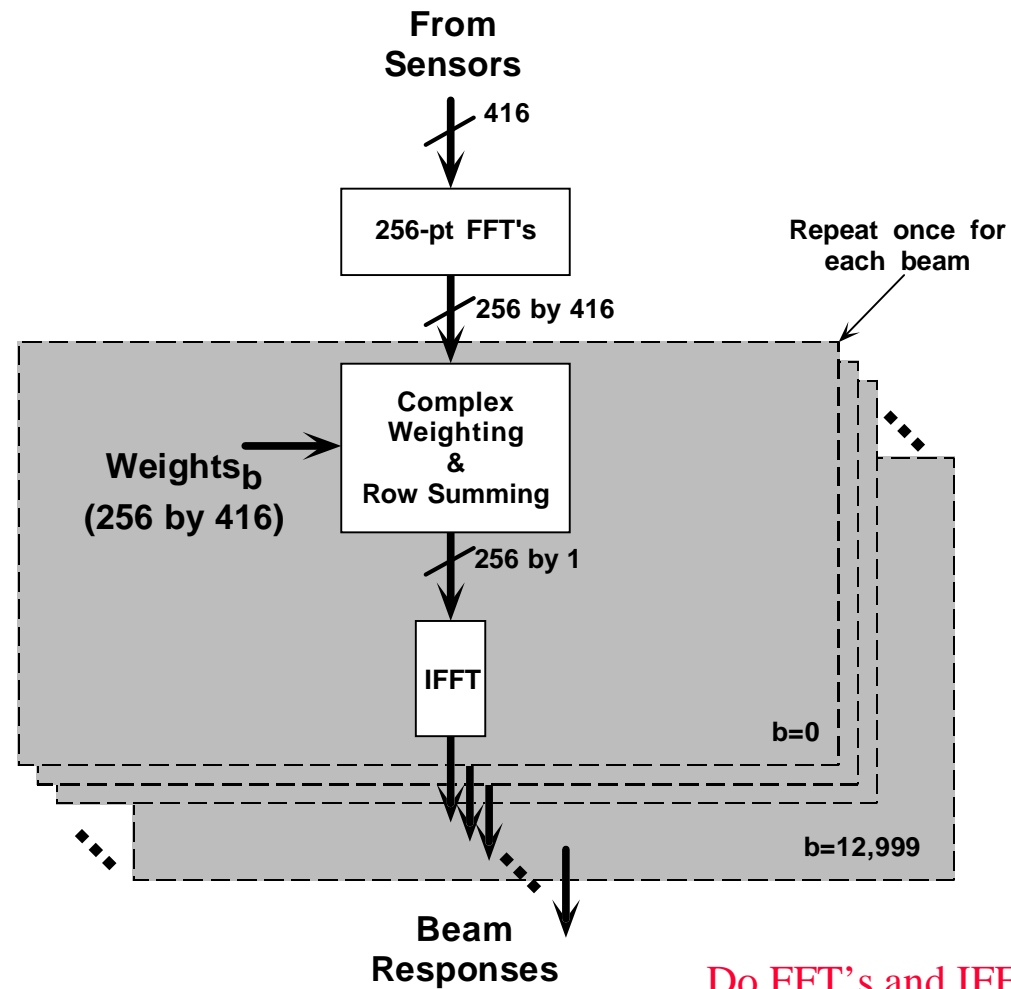


- Each PE does MACs @ 50MHz rate
- Each board computes 64 beams

Time-Delay BF - Towed Array

- ◆ Limitation here is memory - not computing
 - no symmetry - no broadcast
 - » Reduce data to 8-bits
 - » Process 400 sensors' data
- ◆ 2 PE's/FPGA
- ◆ 13,000 beams requires 8.5B MACs/sec
- ◆ Problem requires 22 boards

Frequency-Domain BF for TB-29



- 13,000 beams
- 256 freq bins

Do FFT's and IFFT's using a DSP!

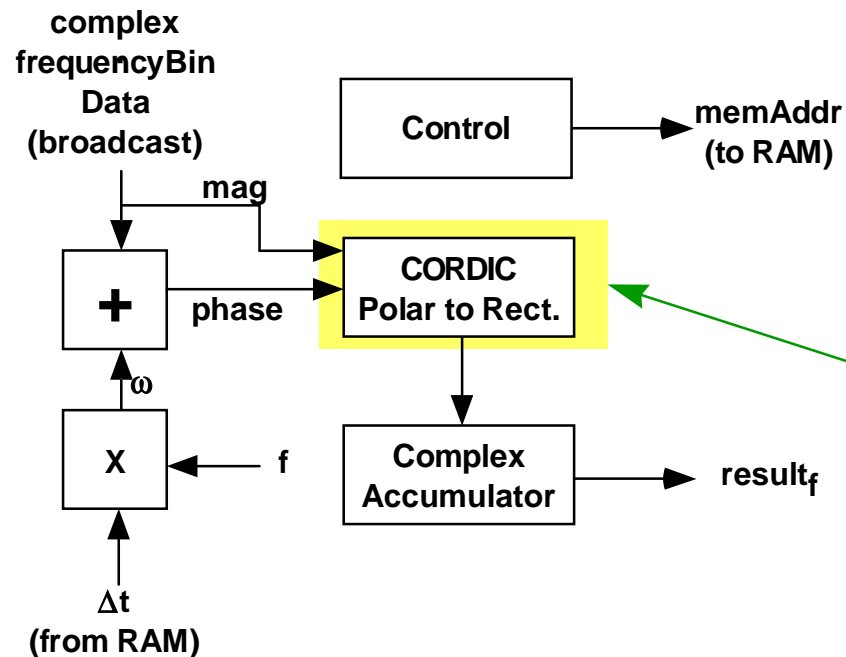
Frequency-Domain BF

- 13,000 beams in 0.164 sec requires $8.4B \frac{CMACs}{sec}$
- 192 PE's can do this
- 4 PE's / FPGA & 4 FPGA's / board \Rightarrow 12 boards
- Broadcast freq bin data to all PE's

Complex MACs

- ★ Idea: use Mag/Phase instead of Re/Im
 - phase shift is an ADD
 - use CORDIC for conversion back to Re/Im

Frequency-Domain BF PE



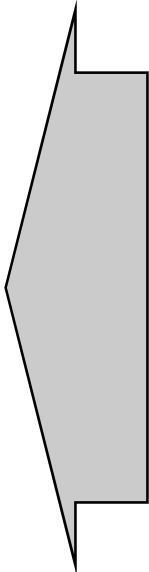
CORDIC is size of 16X16 MPY

Part 2: ACS & DSP's

- ◆ DSP's were *made* for beamforming
- ◆ How does ACS measure up?
- ◆ What can we learn about ACS from a comparison?
- ◆ How scalable is the ACS solution?

Time-Delay: ACS & DSP's

- ◆ Spherical beamformer
 - 1 FPGA == 12 SHARC's
- ◆ Linear array beamformer
 - 1 FPGA == 6 SHARC's



Time-delay
computations

Based on completed
mappings to hardware
and detailed results
analysis.

DSP & ACS - Hardware Features

◆ Memory

- DSP
 - » 2 RAM banks
 - » up to 512KB total
- ACS
 - » 4 external RAMs
 - » 512KB each
 - » more parallel I/O

◆ Arithmetic Unit

- DSP
 - » single-cycle MAC
- ACS
 - » single-cycle MAC if pipelined
 - » other pipelines possible
 - ◆ CORDIC
 - ◆ address generation

DSP & ACS - Hardware Features 2

◆ Multiprocessing

- DSP
 - » 1 PE per SHARC
 - » 6 SHARCs per board
 - » shared bus architecture
 - » shared external memory
 - » MYRINET capability
- ACS
 - » 2-4 PE's per FPGA
 - » flexible architecture
 - » MYRINET capability

◆ Address Generation

- DSP brittle
 - » 3 cycle inner loop for beamforming
- ACS flexible
 - » zero-overhead looping

◆ Summary

- 6-12X ACS advantage *per chip*

ACS Beamforming Scalability

- ◆ Time-delay beamformers are I/O pin-limited, not logic-limited
 - more pins
 - higher I/O clock rate
- ◆ Frequency-domain beamformers
 - currently balanced
 - I/O pin-limited in future

FY 97 Feasibility Study Results

- ◆ Time-delay and frequency-domain beamforming + ACS = **good match**
- ◆ Matched field is promising
- ◆ Quantified performance, I/O requirements, FPGA count, RAM count, board count
- ◆ SLAAC4 board proposal
- ◆ Detailed DSP & ACS comparisons

FY98 Plans

- ◆ CAD

- module generators

- ◆ Algorithms

- adaptive

- ◆ Lab Demos

- first beamforming->FPGA mapping



Rest of BF Deliverables

- ◆ 1998

- “Mapping of beamforming algorithm to an FPGA platform”

- ◆ 1999

- “Demonstration of advanced beamforming and sonar application development (i.e. floating point, advanced adaptive algorithm, etc)
- “... Beamforming/sonar module generators”

- ◆ 2000

- “... Beamforming/sonar subcompilers”

Part 3: Beamforming Option

- ◆ Extend beamforming work
 - Collaboration with NUWC
 - ACS-friendly algorithms
 - Requirements specification
 - Prototype system development
 - Leverage SLAAC reference platform



Option Task #1

- ◆ Extend algorithm feasibility study
 - arrays-to-displays challenge
 - adaptive bf
 - matched field processing
 - data normalization
 - results visualization

Option Task #2

- ◆ System-level demo issues
 - heterogeneous hardware organization
 - configuration
 - diagnostics
 - run-time software
 - validation
 - I/O interfacing

Option Task #3

- ◆ Prototype design
- ◆ Laboratory end-to-end demo
- ◆ Support sea-test

Interactions/Responsibilities

◆ NUWC

- synthetic data
- proposed algorithms
- system integration on submarine
- display system

◆ NUWC + BYU

- hardware organization
- I/O requirements

◆ BYU

- algorithm mapping
- validation of mappings
- end-to-end lab demo

◆ ISI

- reference platform