Optimal Configuration of Combined GPP/DSP/FPGA Systems for Minimal SWAP

Presented by
John K. Antonio
University of Oklahoma

ACS Fall 1999 PI Meeting
Current Personnel

- John K. Antonio, Co-Principal Investigator
- Sudarshan K. Dhall, Co-Principal Investigator
- Jack West, Research Scholar
- Jeff Muehring, Research Scholar
- Hongping Li, Research Assistant, Ph.D. Student
- Sirirut Vanichayobon, Research Assistant, Ph.D. Student
- Seok-Hyun Ko, Research Assistant, M.S. Student
Past Personnel

• Tim Osmulski, Research Assistant, M.S. (graduated 5/98)
• Nikhil Gupta, Research Assistant, M.S. (graduated 8/98)
• Brian Veale, Research Assistant, M.S. (graduated 8/99)
Configuring Combined GPP/DSP/FPGA Systems for Minimal SWAP

Requirements
- Throughput
- SWAP

Applications
- SAR
- STAP

New Ideas
- Systematic determination of minimal SWAP configuration based on proven mathematical programming techniques
- Optimal configuration based on automatic “tuning” of system design parameters
  - number and types of cards used
  - data mapping and communication schemes
  - data format, degree of pipelining, clock rate
- Novel computing techniques based on characteristics of GPP/DSP/FPGA system

Impact
- Embedded Systems requirements for the 21st Century can be satisfied with the combined use of GPP, DSP, and FPGA technologies
- Demonstrate use of FPGA boards as co-processors for embedded multiprocessor GPP and DSP systems
- Demonstrate systematic approaches to optimally configure GPP/DSP/FPGA syst. for minimal SWAP for embedded applications

Schedule
- Develop optimal configuration techniques
- Implement and test optimal configurations on GPP/DSP/FPGA system
- Demonstrate advantages of combining technologies

Jun 97 Start
Jun 98
Jun 99
Jun 00
Dec 00 End

University of Oklahoma: John K. Antonio and Sudarshan K. Dhall
Outline

- System Overview
- Optimal Configuration for SAR
- FPGA Power Prediction Simulator
- Optimal Configuration for STAP
Equipment Status

Mercury

- 20 Slot Hybrid Chassis with SPARC 5V
- Solaris 2.5 with C Compiler
- MC/OS, Cross Assembler, Toolkit
- MPI-Pro for MC/OS
- 9U VME RACE Board
- 1 SHARC Daughtercard (2CNs, 8MB/CN, 3 SHARCs/CN) = 6 SHARCs
- 3 SHARC Daughtercards (2CNs, 16MB/CN, 3 SHARCs/CN) = 18 SHARCs
- 4 PowerPC Daughtercard (2CNs, 16MB/CN, 1 PPC/CN) = 8 PPCs
- RIN-T Input Card
- ROUT-T Output Card

Annapolis Micro Systems

- 4 PCI WILDONE Cards (Xilinx 4028/4036)
- 4 PCI WILDFORCE Array Card (5 Xilinx 4085s)
- Interfacing Cables

Software Support

- MPI/PRO™ (MPI Software Technology, Inc.)
- Synplify Synthesis Software (Synplicity, Inc.)
- Xilinx Foundation Software (Xilinx, Inc.)
Architecture of Prototype System

- **Data Source**:
  - PC
  - Data Sink

- **VME**
  - PE
  - CN

- **Mercury System**
  - DSP/GPP Subsystem
  - Reconfigurable Subsystem

- **Annapolis System**
  - Reconfigurable Subsystem

- **PCI**
  - 120 MB/sec

- **PC**
SAR Processing Flow

- Range Compression
- Azimuth Processing
- Data Transfer

Range → Data Transfer → Azimuth
Outline

- System Overview
- **Optimal Configuration for SAR**
- FPGA Power Prediction Simulator
- Optimal Configuration for STAP
Sectioned Convolution for SAR Azimuth Processing

Large Overlap/Section ratio $\Rightarrow$ Small azimuth memory, large number azimuth processors
Small Overlap/Section ratio $\Rightarrow$ Large azimuth memory, small number azimuth processors

Optimal DSP/GPP Configuration Formulation for SAR

Minimize:

\[ Z = N_X \Pi_{CN}(X_T) + N_Y \Pi_{CN}(Y_T) \]

Subject to:

\[ P_r \leq N_X X_r + N_Y Y_r \]

\[ P_a(S_a) \leq N_X X_a + N_Y Y_a \]

\[ M_{CN}(X_T) \geq X_r \frac{M_r}{P_r} + X_a \frac{M_a(S_a)}{P_a(S_a)} \]

\[ M_{CN}(Y_T) \geq Y_r \frac{M_r}{P_r} + Y_a \frac{M_a(S_a)}{P_a(S_a)} \]

\[ X_r + X_a \leq P_{CN}(X_T) \]

\[ Y_r + Y_a \leq P_{CN}(Y_T) \]

\[ F_a = 2^k \geq S_a + K_a, \quad k = 1, 2, \ldots \]

\[ N_X, N_Y, X_r, X_a, Y_r, Y_a \geq 0, S_a \geq 1 \]
Minimum Power
Optimal CN Configurations

![Diagram showing optimal CN configurations with resolution on the x-axis and velocity on the y-axis.}]
Comparison of Two FPGA FIR Filter Designs

Serial-Multiply/Parallel Add

- Ease of routing
- Poor modularity

Parallel-Multiply/Serial Add

- Poor routing
- Good modularity
Comparison of Two FPGA FIR Filter Designs

- Both designs implemented using fixed-point complex data (16-bit fixed-point real and imaginary components)
- Both designs make use of constant coefficient multipliers (from core generator)
- Four tap serial-multiply/parallel-add filter fit onto one 4036xla part
- Three tap parallel-multiply/serial-add filter fit onto one 4036xla part (insufficient routing resources for four taps)
- Four tap parallel-multiply/serial-add filter implemented across two parts on one board (one 4036 and one 4013)
Including FPGAs in the SAR Optimization Formulation

- Power estimates must be determined for a range of kernel sizes for both filter designs

- Hybrid designs may exist for multi-chip implementations that yield desired features of both modularity and routability

- Binary optimization variable defines whether entry-FPGA or DSP/GPP subsystems perform range compression

- Real optimization variable defines fraction of azimuth processing divided among GPP/DSP and exit-FPGA subsystems
Outline

• System Overview
• Optimal Configuration for SAR
• **FPGA Power Prediction Simulator**
• Optimal Configuration for STAP
Power Dissipation in CMOS

- **Leakage Current**
- **Dynamic Capacitance Charging Current**
- **Transient Current**

Most important for CMOS
Dependant on clock frequency
Dependant on signal activity

\[ P_{avg} = \frac{1}{2} CV^2 fA \]
$p(s)$: the probability that signal $s$ attains a logical value of true at any given clock cycle.

\[ p(clock) = 0.50 \quad p(x_1) = 0.88 \quad p(x_2) = 0.29 \quad p(x_3) = 0.69 \]

$A(s)$: the probability that signal $s$ transitions at any given clock cycle.

\[ A(clock) = 1.0 \quad A(x_1) = 0.10 \quad A(x_2) = 0.17 \quad A(x_3) = 0.27 \]
Signal Design

Local Signal
- Symbolic Probability
- Numeric Probability
- Numeric Activity

Remote Signal
- Signal Reference
- Manhattan Distance

CLB

Signal Reference
Manhattan Distance
Routing Example
Basic Approach to Calibration

- $N \times N$ array of CLBs (configurable logic blocks)
- $S$ denotes the set of all internal signals for a configuration and $S_i$ denote all signals of length $i$
- $A_i$ denotes the sum of activities for all signals of length $i$
- $2N + 1$ distinct capacitances ($C$) dependent on signal length

$$P_{avg} = \frac{1}{2} V^2 f \sum_{s \in S} C_d(s) A_s$$

$$P_{avg} = \frac{1}{2} V^2 f \left( C_0 \sum_{s \in S_0} A_s + C_1 \sum_{s \in S_1} A_s + \cdots + C_{2N} \sum_{s \in S_{2N}} A_s \right)$$
Basic Approach to Calibration

• For the $j$-th design/data set combination:
  let $P_j$ denote the measured power
  let $A_{j,k}$ denote the aggregate activity of all signals
  of length $k$

\[
\frac{1}{2} V^2 f \begin{pmatrix}
A_{0,0} & A_{0,1} & \cdots & A_{0,2N} \\
A_{1,0} & A_{1,1} & \cdots & A_{1,2N} \\
\vdots & \vdots & \ddots & \vdots \\
A_{2N,0} & A_{2N,1} & \cdots & A_{2N,2N}
\end{pmatrix}
\begin{pmatrix}
C_0 \\
C_1 \\
\vdots \\
C_{2N}
\end{pmatrix} =
\begin{pmatrix}
P_0 \\
P_1 \\
\vdots \\
P_{2N}
\end{pmatrix}
\]

• For each design/data set combination, the simulator
  provides the values for one row of the above matrix
• Given $2N + 1$ measured values for $P_j$, the unknown
  capacitance values are then determined. This is how the
  simulator is calibrated.
Outline

- System Overview
- Optimal Configuration for SAR
- FPGA Power Prediction Simulator
- Optimal Configuration for STAP
Pulses
Data Cube
Doppler Filter
Channels
Range
Pulses
Input Data
Data Cube
Pulse Compress
Rotate
Doppler Filter
Channels
Range
Pulses
Data Cube
QR Decomposition
Beamform
Weights
Steering Vectors
Channels
Range
Beam Outputs
Pulses
STAP PROCESSING FLOW
Pulse Compression Partitioning
with range dimension whole.

Doppler Filtering Partitioning
with pulses dimension whole.
RACE NETWORK INTERCONNECT
FAT-TREE TOPOLOGY
1. 40Mhz clock, 32 bit data paths, 2048 byte circuit-switched packets.

2. Contention resolved using priorities.
   a. User-programmable message priority
   b. Hardware priority assigned at each crossbar along a path (based on complex connection rules)

3. A packet with higher priority preempts (suspends) a lower priority packet (active or inactive) to gain control of a crossbar port.
### STANDARD CROSSBAR PRIORITY ARBITRATION ALGORITHM TABLE

<table>
<thead>
<tr>
<th>Hardware Priority</th>
<th>Transaction Status</th>
<th>Active</th>
<th>Not Yet Active</th>
<th>Port E Involved</th>
<th>Port E Not Involved</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Entry Port</td>
<td>Exit Port</td>
<td>Entry Port</td>
<td>Exit Port</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>F</td>
<td>A,B,C,D,E</td>
<td>F</td>
<td>A,B,C,D,E</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>E</td>
<td>F</td>
<td>E</td>
<td>F</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>A,B,C,D</td>
<td>F</td>
<td>A,B,C,D</td>
<td>F</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>E</td>
<td>A,B,C,D</td>
<td>E</td>
<td>A,B,C,D</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>*A,B,C,D</td>
<td>*A,B,C,D,E</td>
<td>A,B,C,D*</td>
<td>A,B,C,D*</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>-</td>
<td>-</td>
<td>A,B,C,D</td>
<td>E</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

* - Peer Kill Rules Apply
Process Set - Phase 1
(CN:12, R:200, P:22, C:16, Routing:F)

Communication Phase 1
PROCESS SET
SIMULATED PERFORMANCE
Communication Phase 2

Process Set - Phase 2
(CN:12, R:200, P:22, C:16, Routing:F)
Communication Phase 1

Message Traffic - Phase 1
(CN:16, X:12, Y:4, R:400, P:22, C:16, Routing:EF)

Count

Time (ms)

CN Traffic

CE Traffic
Message Traffic - Phase 2

Communication Phase 2

Message Traffic - Phase 2
(CN:16, X:12, Y:4, R:400, P:22, C:16, Routing:EF)

Count

Time (ms)
STAP Mapping/Scheduling Optimization

STAP Data Cube

Mercury RACE®
(Configured with 1..P CNs)

Network Simulator
(Estimate Overall Communication Time)

Genetic Algorithm
(Determine Optimal Schedule)

Select # CNs (P)
(P=Allocated Compute Nodes)

Select Fixed or Random Mapping

Minimize Mapping
(Use Objective Function)

OPTIMIZE

Adjust Allocated P
Including FPGAs in the STAP Optimization Formulation

- Utilize filter optimizations from SAR formulation for pulse compression phase of STAP

- FPGA optimization parameters for weight calculations
  - degree of pipelining
  - data format (e.g., integer vs. floating point)
  - basic architecture (e.g., accumulator vs. adder)
Array-Based Integer Multiplier

 CSA 0

 CSA 1

 CSA 2

 CSA 3

 CSA 4

 CSA 5

 CSA 6

 CSA 7

 CSA 8

 CSA 9

 Propagate Adder

 $b_{11A} \ b_{10A} \ b_{9A} \ b_{8A} \ b_{7A} \ b_{6A} \ b_{5A} \ b_{4A} \ b_{3A} \ b_{2A} \ b_{1A} \ b_{0A}$
• The Wild-One system runs at a maximum speed of 50MHz
• The 4036xla has more routing resources than the 4028ex
• Table shows maximum achieved clock rate as a function of the number of pipelined stages employed

<table>
<thead>
<tr>
<th># pipelined stages</th>
<th>Max.Speed (Mhz)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4028ex</td>
</tr>
<tr>
<td>1</td>
<td>14</td>
</tr>
<tr>
<td>2</td>
<td>19</td>
</tr>
<tr>
<td>3</td>
<td>21</td>
</tr>
<tr>
<td>4</td>
<td>22</td>
</tr>
<tr>
<td>5</td>
<td>29</td>
</tr>
<tr>
<td>6</td>
<td>39</td>
</tr>
<tr>
<td>7</td>
<td>22</td>
</tr>
<tr>
<td>8</td>
<td>33</td>
</tr>
</tbody>
</table>
Floating Point Multiplier

12 bit Array-Based Multiplier

If the msb = 1 take the bits msb-1…msb-11
If the msb = 0 take the bits msb-2…msb-11

If underflow = 1, set exponent = 0
If overflow = 1, set exponent = 15 (representing infinity)

If the msb = 1 take the bits msb-1…msb-11
If the msb = 0 take the bits msb-2…msb-11

excess-7 adder

1 bit 4 bits 11 bits
Floating Point Adder

Choose Exponent

Align Mantissas

Add/Subtract Mantissas

Normalize Mantissa and Adjust Exponent

Check for Absolute Zero and Infinity and Add Phantom Bit

Registers

Compare Exponents by Subtraction

difference

pos./neg.

Registers

Registers

Read exponent mantissa sign
Inner Product Co-processor Designs

Multiply-Add Scheme

Multiply-Accumulate Scheme
## Performance

<table>
<thead>
<tr>
<th>Co-Processor Type</th>
<th>Max. Speed</th>
<th># of CLBs</th>
<th># of Flip-Flops</th>
<th># of 3-Input LUTs</th>
<th># of 4-Input LUTs</th>
<th>Equivalent Gate Count</th>
<th>Simulated Consumption*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Int. Multiply-Accumulate</td>
<td>50MHz</td>
<td>622</td>
<td>720</td>
<td>180</td>
<td>794</td>
<td>10076</td>
<td>N/A</td>
</tr>
<tr>
<td>Int. Multiply-Add</td>
<td>43MHz</td>
<td>1013</td>
<td>1148</td>
<td>423</td>
<td>1421</td>
<td>16809</td>
<td>415</td>
</tr>
<tr>
<td>F.P. Multiply-Accumulate</td>
<td>38MHz</td>
<td>437</td>
<td>414</td>
<td>154</td>
<td>742</td>
<td>8072</td>
<td>454</td>
</tr>
<tr>
<td>F.P. Multiply-Add</td>
<td>34MHz</td>
<td>716</td>
<td>654</td>
<td>254</td>
<td>1082</td>
<td>11766</td>
<td>390</td>
</tr>
</tbody>
</table>

### Notes:

1. Integer co-processors implemented with 16-bit integer multipliers and 32-bit integer adders
2. The simulated power consumption calculated from power simulator based on simplified (non-calibrated) constants:

\[
* \text{Simulated Consumption} = \left( 1 \sum_{s \in S_0} A_s + 2 \sum_{s \in S_1} A_s + \cdots + 2N \sum_{s \in S_{2N-1}} A_s + (2N+1) \sum_{s \in S_{2N}} A_s \right)
\]
Work to be Completed

- Interfacing of FPGA and GPP/DSP Subsystems
- Implement Parallel SAR Algorithm on GPP/DSP System
- Integrate FPGA FIR Filters for Range and Azimuth Processing for SAR
- Implement Parallel STAP Algorithm for GPP/DSP System
- Integrate FPGA FIR Filters for Range Processing for STAP
- Implement FPGA-based Linear Equation Solver
- Integrate FPGA-based Linear Equation Solver with STAP