Automated Bus Generation for Multiprocessor SoC Design

Kyeong Keol Ryu and Vincent J. Mooney III
School of Electrical and Computer Engineering
Georgia Institute of Technology, Atlanta, GA, USA

March 5, 2003
Outline

- Introduction
- Related Work
- Bus System Structure
- Bus Generation Tool and It’s User Options
- Bus Generation Sequence with an Example: BFBA
- Application Examples
- Experimental Environment
- Performance Evaluation
- Conclusion
Introduction

- System-on-a-Chip (SoC)
- Bus Synthesis Goal
  - Easy and Quick Design
  - Design Space Exploration
- Motivation
  - Automated Custom Bus Design
  - Automated Design Space Exploration across Performance Impacting Factors (e.g., type of bus architecture, PEs and programming style)
Related Work

- M. Gasteier et al., “Bus-Based Communication Synthesis on System-Level”
  - Automatic generation of communication topologies on system-level
- R.A. Bergamaschi et al., “Designing Systems-on-Chip using cores”
  - Assembling an SoC using IP blocks and their properties
  - A component-based approach to SoC system building
- TIMA Lab: Component-based Design, Wrapper Generation
- Our Work: Supporting Multiple and Heterogeneous Bus Architectures and Various Wrappers in a System
- SoC Bus Efforts in Industry
  - AMBA from ARM Ltd., CoreConnect from IBM Corp., SiliconBackplane from Sonics Inc. Wishbone from Silicore Corp., CoreFrame from Palmchip Corp.
Bus System Structure

Bus System

Bus Subsystem 1
- Arbiter
- Memory
- CPU_A
- Memory
- CPU_B
- Memory

Bus Subsystem 2
- BAN G
- BAN A
- BAN B
- BAN J

BusSyn and User Options

**User Input List**

1. **Bus System:**
   - Number of Bus Subsystems

2. **Bus Subsystem:**
   - For Each Bus Subsystem
     - Number of BANs:
     - Bus Type: GBAVI, GBAVIII, BFBA, Hybrid or SplitBA

3. **BAN Properties:**
   - For Each BAN:
     - CPU Type: MPC750, MPC755, MPC7410 or ARM9TDMI
     - Non-CPU Type: DCT or MPEG2 decoder
     - Number of Memories

4. **Memory Properties:**
   - For Each BAN:
     - For Each Memory:
       - Type: SRAM, DRAM, DPRAM or FIFO
       - Address bus width
       - Data bus width

5. **Bus Properties:**
   - For Each Bus Type
     - address bus width
     - data bus width
     - Bi-FIFO depth for BFBA and Hybrid
A Bus Generation Sequence with an Example, BFBA

1. Bus System:
   - # of Bus Subsystems: 1
   - Bus Subsystem:
     - # of BANs: 2
     - Bus Type: BFBA
   - BANs:
     - BAN 1:
       - CPU Type: MPC755
       - # of Memories: 1
     - BAN 2:
       - CPU Type: MPC755
       - # of Memories: 1

2. Memory Properties:
   - BAN 1:
     - Type: SRAM
     - Address Bus Width: 20
     - Data Bus Width: 64
   - BAN 2:
     - Type: SRAM
     - Address Bus Width: 20
     - Data Bus Width: 64

3. Bus Properties:
   - Data Bus Width: 64
   - Bi-FIFO Depth: 1024

User Input List

1. Bus System:
   /* Skipped */
   .up_dataout(dataout_up_2[FIFO_D_WIDTH-1:0]),
   .up_gen_int(gen_int_up_2),
   .up_isr0_cthi(isr0_cthi_up_2),
   .up_isr0_ctllo(isr0_ctllo_up_2),
   .dn_datain(datain_up_3[FIFO_D_WIDTH-1:0]),
   .reb_dn(reb_up_3),
   .web_dn(web_up_3),
   .fifo_area_dn(fifo_area_up_3)
); endmodule

module BusSystem(sysrstb, sysclk);
  input sysrstb;
  input sysclk;
  SubSys_bfba_ban4 SubSystem1(
    .sysrstb(sysrstb),
    .sysclk(sysclk)
  );
endmodule

- Data bus width: 64
- Bus System Generation
- Bus Subsystem Generation
- Bus Access Node (BAN) Integration
- Module Extraction from Library & Required Module Generation
- User Input
Bus System Examples I
Bus System Examples II
Application Examples and Experimental Environment

- OFDM Transmitter, a Wireless Application
- MPEG2 Decoder
- Database Example, a Multi-thread Example

Note: VCS and Design Compiler from Synopsys, Seamless CVE and Xray from Mentor Graphics and GCC from GNU
OFDM Transmitter

<table>
<thead>
<tr>
<th>Case</th>
<th>Bus System</th>
<th>Throughput [Mbps]</th>
<th>Software Programming Style</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BFBA</td>
<td>2.6504</td>
<td>PPA</td>
</tr>
<tr>
<td>2</td>
<td>GBAVI</td>
<td>2.1087</td>
<td>PPA</td>
</tr>
<tr>
<td>3</td>
<td>GBAVIII</td>
<td>4.5599</td>
<td>FPA</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>2.2567</td>
<td>PPA</td>
</tr>
<tr>
<td>5</td>
<td>Hybrid</td>
<td>4.5599</td>
<td>FPA</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>2.6504</td>
<td>PPA</td>
</tr>
<tr>
<td>7</td>
<td>SplitBA</td>
<td>5.1132</td>
<td>FPA</td>
</tr>
<tr>
<td>8</td>
<td>GGBA</td>
<td>4.3913</td>
<td>FPA</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>2.1880</td>
<td>PPA</td>
</tr>
</tbody>
</table>

Note: 1. PPA: Pipelined Parallel Algorithm, FPA: Functional Parallel Algorithm  
2. Data: 2048 complex samples and 32 guard complex samples per packet  
3. All Bus Systems run on four PowerPCs support instruction and data cache operations

- SplitBA outperforms GGBA by 16.44%
- Bus Systems using a shared memory (e.g., GGBA) requires more memory arbitration time than in Bus System having separate memory (e.g., GBAVIII)
Performance Evaluation II

- **MPEG2 Decoder**

<table>
<thead>
<tr>
<th>Case</th>
<th>Bus System</th>
<th>Throughput [Mbps]</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>BFBA</td>
<td>0.8594</td>
</tr>
<tr>
<td>11</td>
<td>GBAVI</td>
<td>0.8271</td>
</tr>
<tr>
<td>12</td>
<td>GBAVIII</td>
<td>1.1444</td>
</tr>
<tr>
<td>13</td>
<td>Hybrid</td>
<td>1.1650</td>
</tr>
<tr>
<td>14</td>
<td>CCBA</td>
<td>1.0083</td>
</tr>
</tbody>
</table>

[Note] 1. Picture size: 16 x 16
2. All Bus Systems run on four PowerPCs have Functional Parallel Algorithm

- Hybrid shows the best in performance (15.54% against CCBA)

- **Database**

<table>
<thead>
<tr>
<th>Case</th>
<th>Bus System</th>
<th>Execution Time [ns]</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>GGBA</td>
<td>2,241,100</td>
</tr>
<tr>
<td>16</td>
<td>SplitBA</td>
<td>1,317,804</td>
</tr>
</tbody>
</table>

[Note] 1. Each Bus System is composed of 1 server and 4 clients.
2. The server has 1 task, and the each client has 10 tasks.
3. Each task accesses 100 data to or from the shared memory.

- SplitBA outperforms GGBA by 41% reduction in time
Summary and Conclusion

- Bus System Structure for Bus System Generation
- Bus System Generation Tool: BusSyn
- BusSyn: How to Generate BANs, Bus Subsystems and Bus System
- Performance Evaluation:
  - SplitBA Shows 16.44% Improvement and 41% Reduction in Time When Compared to GGBA
  - Hybrid Outperforms CCBA by 15.54% in MPEG2 Decoder
- Methodology Benefit