Agenda

• What is a Configurable Processor?
• Configurable Processors architectures
• Building custom peripherals
• Hardware/software trade-offs, algorithmic acceleration
• Comparing the alternatives
• Configurable Processor technical challenges
  – Communication
  – Software development environment
  – Debugging
• Summary
Terminology Used in this Session

• **Configurable Processor**
  - Embedded processor core
  - Programmable logic to build peripherals
  - Dedicated System Bus
  - On-chip memory

• **User-Definable Processor**
  - Parameterized or changeable processor hardware description
  - Typically synthesizable from VHDL/Verilog
  - Usually targeted to ASIC or system-on-a-chip
  - Examples: ARC, Tensilica, etc.
Trends Toward the Configurable Processor

- **Decreasing Time-to-Market**
  - Fast iterations
  - Fast in-system, real-time debugging
  - Fast component availability

- **More Adaptability**
  - During design and debug
  - In the field or in the application

- **Higher Performance**
  - Match the architecture to the problem
  - Fast response to real-time events
  - Parallel operations

- **Increased Differentiation**

- **Ever-improving Process Technology**
Toward a Configurable Processor

Increasing Functionality, Density, and Performance

1980
- Discrete Solutions
  - MCU
  - PAL
  - TTL
  - EPROM
  - RAM
  - Peripheral

1990
- MCU Derivatives and Programmable Logic
  - MCU Derivative
  - CPLD/FPGA
  - RAM
  - EPROM/FLASH
  - Peripheral

2000
- Configurable Processors
  - Configurable Processor
  - "System on a Chip"

"Roll-Your-Own" MCU in FPGA

"System on a Chip"
What Is a Configurable Processor?

- Industry-standard processor
- Dedicated bus
- Programmable Logic
  - Soft peripherals
  - User-defined functions
  - Hardware acceleration
- On-Chip Memory
## Configurable Processors

<table>
<thead>
<tr>
<th>Vendor/Family</th>
<th>Processor</th>
<th>Status</th>
<th>Dedicated Resources</th>
<th>Programmable Resources</th>
<th>Embedded Bus Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Triscend/E5</td>
<td>8032 &quot;Turbo&quot;</td>
<td>Sampling</td>
<td>2-channel DMA 8K-64K bytes RAM Hardware debug JTAG</td>
<td>Triscend Coarse-grained, bus oriented</td>
<td>8-bit Data 32-bit Address</td>
</tr>
<tr>
<td>Triscend</td>
<td>ARM 7TD MI</td>
<td>In Development</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Motorola/CORE+</td>
<td>ColdFire</td>
<td>Cancelled</td>
<td>2-channel DMA 3K bytes RAM DRAM controller Hardware debug</td>
<td>Motorola MPA Fine-grained</td>
<td>Multiple busses Unknown format</td>
</tr>
<tr>
<td>National/NAPA 1000</td>
<td>Compact-RISC</td>
<td>In Development</td>
<td>16K RAM 8x256 RAM Timer JTAG debugger</td>
<td>Concurrent Fine-grained</td>
<td></td>
</tr>
<tr>
<td>Siemens</td>
<td>TriCore</td>
<td>Plans Announced</td>
<td></td>
<td>Gatefield Fine-grained</td>
<td></td>
</tr>
<tr>
<td>Atmel</td>
<td>?</td>
<td>Plans Announced</td>
<td></td>
<td>Atmel AT40K Coarse-grained</td>
<td></td>
</tr>
<tr>
<td>SIDSA/FIPSOC</td>
<td>8031</td>
<td>Sampling?</td>
<td>Programmable analog</td>
<td>SIDSA Coarse-grained</td>
<td>None, Memory-mapped</td>
</tr>
</tbody>
</table>
Motorola CORE+

Clock
JTAG
8Kbyte Instruction Cache
8Kbyte SRAM
DMA Controller
Parallel Port
Timers
DUART
I²C Controller

System Bus Controller
Master Bus Interface
Slave Bus Interface
1Kbyte Embedded RAM
3Kbyte Embedded RAM

DRAM Controller
Chip Selects
Interrupt Controller
External Bus Interface
Chip Selects

Fine-Grained FPGA Array

CANCELLED!
National NAPA

System Port

National CompactRISC CPU

Bus Interface Unit

Peripheral Devices

ToggleBus Transceiver

Reconfigurable Pipeline Controller

Pipeline Memory Array

Scratchpad Memory Array

Adaptive Logic Processor
(fine-grain FPGA architecture)

Configurable I/O

External Memory Interface
Triscend E5 Configurable Processor

- Power Control
- Clock and Crystal Oscillator Control
- Power-On Reset
- Bus Arbiter
- Memory Interface Unit
- Address Mappers
- Two-channel DMA Controller
- JTAG Interface
- 8032 "Turbo" Microcontroller
- Configurable System Logic (CSL) Matrix
- Configurable System Interconnect (CSI) bus
- Byte-wide System RAM
- Hardware Breakpoint Unit
- Selectors
- Data Bus
- Address Bus
- CSI Socket
Configurable Processor Applications

- Custom Peripheral Set
  - Practically any digital function
  - Matched specifically to the application
  - Derivative on demand

- Hardware Acceleration
  - Algorithms in hardware
  - Handling odd-size math
  - Faster real-time response
  - Multiple operations in parallel
  - Bit manipulation
Custom Peripherals
(Hardware/Software Trade-Offs)

• **Software Solution** (µs to ms)
  – Slow peripherals (serial ports, etc.)
  – Limited by CPU performance (Scenix, Teragen)
  – Easy to modify
  – Cheap, re-use existing silicon

• **Hardware Solution** (ns to µs)
  – Standard derivative (no differentiation)
  – CPU + ASIC/FPGA (difficult to modify)
  – Configurable Processor (easy to modify)
  – Additional silicon/cost
Example: SPI Interface

• Find a processor derivative that matches your requirements
  – It has SPI, but does it have everything else you need?
  – Availability? Software support?
• Implement your peripheral in software (ex. Scenix)
• Build your peripheral in an external ASIC or FPGA
• Use a SPI soft peripheral in your configurable processor
# Design Techniques

<table>
<thead>
<tr>
<th>Peripherals in Software</th>
<th>Peripherals in Hardware</th>
</tr>
</thead>
<tbody>
<tr>
<td>• ‘C’ language</td>
<td>• Schematic capture</td>
</tr>
<tr>
<td>• Assembly</td>
<td>• VHDL/Verilog entry</td>
</tr>
<tr>
<td>• Instruction-set</td>
<td>• Digital logic simulator</td>
</tr>
<tr>
<td>simulator</td>
<td>• Soft macros available</td>
</tr>
<tr>
<td>• Function library</td>
<td></td>
</tr>
</tbody>
</table>
Hardware Acceleration Example

• Calculate the instantaneous average of four 8-bit values

\[ Z = \frac{(A + B + C + D)}{4} \]

• Issues
  – Concurrency (I/O, processing requirements)
  – Handling overflow (accumulator width)
  – Performance (processing time)
Two Solutions

• Processor Solution
  - Move PortA to Accumulator
  - Add PortB to Accumulator
  - Add PortC to Accumulator
  - Add PortD to Accumulator
  - Shift Accumulator Right Twice (divide by four)
  - Move Accumulator to RegZ

More instances require additional time

• Logic Solution
  - A + B + C + D ÷4 → Z

More instances require additional logic
## Comparing the Alternatives

<table>
<thead>
<tr>
<th>Solution</th>
<th>Device Cost</th>
<th>Development Time/Cost</th>
<th>Issues</th>
<th>When to Use It</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor Derivative</td>
<td>$1 - $15</td>
<td>Quick/ Low</td>
<td>Availability, software support, differentiation</td>
<td>Lowest cost, if your application fits</td>
</tr>
<tr>
<td>System-on-a-Chip</td>
<td>$5 - $50 + development cost</td>
<td>Long/ High</td>
<td>Acquiring cores, verification, NRE, vendor selection</td>
<td>Volume, complexity, performance justify it.</td>
</tr>
<tr>
<td>Fast Processor</td>
<td>$5 - $50</td>
<td>Moderate/ Low</td>
<td>Creating ‘soft’ peripherals</td>
<td>If it fits and it’s fast enough, use it!</td>
</tr>
<tr>
<td>CPU + ASIC/FPGA</td>
<td>$10 - $100</td>
<td>Moderate/ Moderate</td>
<td>Multi-chip solution, inter-chip communication, debugging support, multiple CAE tools</td>
<td>For applications where a configurable processor does not yet exist.</td>
</tr>
<tr>
<td>Configurable Processor</td>
<td>$8 - $80</td>
<td>Moderate/ Moderate</td>
<td>New technology</td>
<td>Fast time to market, complete embedded system</td>
</tr>
</tbody>
</table>
Configurable Processor Technical Challenges

• Communication between the processor and programmable logic functions
• Maintaining a standard development flow
• Debugging a system with both processor and programmable logic
Communication between the Processor and Programmable Logic

- Distributing the data and address bus to the programmable logic
- Decoding/controlling bus transactions
- Multi-master bus support
- Register intimacy
- Debugging support
One Solution: CSI Bus Socket
(Configurable System Interconnect)

- Distributes address and data to CSL matrix
  - Full access to data and address bus
  - Dedicated address decoding
  - Predictable, synchronous timing
  - Forward compatible with future configurable processors
  - Wait-state and breakpoint control
  - Contention-free bussing

8032 "Turbo" Microcontroller

2-Channel DMA Controller

Hardware Breakpoint Unit

Configurable System Logic (CSL) Matrix

CSI Socket Interface

Side-band Signals

8032 "Turbo" Microcontroller

Configurable System Interconnect (CSI) Bus

Data Write

Data Read

Address

Selectors

Bus Clock

DMA Request/Acknowledge

Wait-State Control

Breakpoint Control

2-Channel DMA Controller
Decoding Bus Transactions

CSI Selector Style

Fast address decoding
- Any address range
- Access type
  - Code
  - Data
- Exported Special Function Registers (SFR)
- Three modes
  - Selector
  - Chip Select
  - DMA Control Register
- Up to 200 selectors in a single device
Connecting the Two Development Worlds

**Hardware Development**
- Design and “soft” module libraries
- Passing register addresses to compiler/assembler
- Vendor place and route software
- Device programming support
- System-wide in-system debugging support

**Software Development**
- Compiler/assembler support
- Function libraries
- Instruction-set simulator
- System-wide in-system debugging support
Preserving Existing Tool Flow

Configurable Processor Tools

1. System Configuration
2. Program Development
3. Program Debug
4. System Test and Debug
5. Device Programming

MCU Development Tools

Header File
Object File

Soft Module Library
Source Code Library

Designer’s standard tool flow
System on a Chip Flow

- 3rd Party EDA Tools
  - Capture or Synthesis
  - Functional Simulation

- Configurable Processor Development Software
  - System Configuration
  - System Test and Debug
  - Device Programming

- MCU Development Tools
  - Program Development
  - Instruction Simulation
  - In-System Debug

Designer’s standard tool flow

Source Code Library

Netlist capture or synthesis, functional simulation, system configuration, system test and debug, device programming, program development, instruction simulation, in-system debug.
Example System: FastChip Software

“Soft” Module Library

Dedicated Resources

“Soft” peripherals dragged into CSL matrix

Resources Used Indicators

Example System: FastChip Software

“Soft” Module Library

Dedicated Resources

“Soft” peripherals dragged into CSL matrix

Resources Used Indicators
Real-Time, In-System Debugging

- Difficult in most ASIC or system-on-a-chip designs
  - Must rely on simulation before completing design
- Most debuggers only support the processor
  - Monitor bus activity
  - Monitor processor registers
  - Break on event and single-step
- Additional debugging desired for programmable logic functions
  - Monitor the state of logic and flip-flops in “soft” peripherals
  - Monitor or force a breakpoint from programmable logic
Configurable Processor Debugging Capabilities

Access to all address mapped and other key processor resources

Commands from 3rd party debuggers translated to JTAG instructions

Breakpoint unit snoops the internal bus, providing complex runtime control features

All sequential and combinatorial logic nodes have complete observability
Summary

• **Configurable Processors** offer benefits in embedded system design:
  – Faster time to market than ASIC/SOC designs
  – Higher performance compared to most processors
  – Higher product differentiation

• **Configurable processors** are a new class of single-chip programmable devices designed for embedded systems applications
  – Industry-standard processor
  – Dedicated, high-performance internal bus
  – Programmable logic, connected to internal bus
  – On-chip, high-density memory