Nexus Based
Multi-Core Debug

Dr. Neal Stollon
MIPS Technologies / First Silicon Solutions
neals@fs2.com

Rich Collins
Freescale Semiconductor
rich.collins@freescale.com
Impact of Debug Strategy

Yes! but... half of what???
Prototype silicon to production release is 6-9 months.

Debug cost can get even bigger for new sets of debug problems for multi-core designs

Much of cost of debug driven by architecture limitations and debug learning curve
But what if SoC internals are more visible, accessible?
Debug implementations become standardized?
SoC Debug Evolution

Embedded Debug Complexity keeps increasing
- Gates increase geometrically - Pins increase linearly
- Significant debug difficulties for leading architectures
- More complex debug needs better Instrumentation
What is Nexus?

• Nexus is a standard for industry level instrumentation with real time interface for embedded system debug
  • “Nexus” is synonymous with the IEEE-ISTO 5001 std.
  • Supported by over 20 Member Companies
  • Specification freely available at http://www.nexus5001.org

• Nexus is a toolbox of defined ways for processor debug:
  • Defines JTAG and Auxiliary bus debug interfaces
  • Defines simple packet based messaging protocol
  • Defines standard set of debug appliation registers
  • Defines 4 implementation classes for standardizing increasingly complex debug implementations
Nexus Forum Philosophy

• Nexus Forum is industry wide consortium supporting Nexus standard with
  – Aggregate Best Practices
  – Proven functionality
  – Re-use technology where possible

• Maximum functionality for minimum pins and silicon

• Reducing pins while providing acceptable visibility

• Minimal impact to performance while running
  – Create visibility for deeply embedded information

• Allow for vendor differentiation and extensions
  – Enable vendor specific extensions without interfering with standard functionality
Nexus Forum Membership
Nexus Technology CPU Core List

• Range of Supported Cores
  – ARM7
  – ARM9
  – PowerPC MPC500
  – Book-e e200z6 PowerPC
  – M-CORE
  – Super10
  – CompactRISC CR16C
  – DSP - StarCore
  – Video processor
  – eZ80
  – Time Processor (eTPU)
  – AHB (ARM AMBA bus)
Nexus architecture defines a combination of:

- JTAG extensions
- Auxiliary Port signals
- debug message protocol
- debug registers and data
FS2 Multi-Core Debug (MED)

Key interfaces: Scalable and modular SoC instrumentation framework
- Capture of many types of trace and other debug information
- System event detection, triggering, trace and analysis
  - System level view - Synchronized multi-core trigger & control
  - In depth core debug - execution/ data trace for each core
  - Ability to run in conjunction with bus level instruments
MED Nexus Environment

- Core A (Nexus compliant) subsystem
- Core B (Non-Nexus) subsystem
- Nexus Port
- Core A to Nexus Interface
- Core B to Nexus Translation
- Embedded Nexus Bus Trace
- Nexus Port Control
- Nexus Probe (Off Chip)
- Combined Nexus Stream

- System Run Control, trigger in/out signals
- Inst/Data/Addr trace
- Compression/Formatting

- AMBA AHB / OCP
- JTAG
- MED Ctrl & Bus Monitor SW
- Core A Debugger SW
- Core B Debugger SW

- Bus Monitor Gasket
- Inst/Data/Addr trace
Nexus Classes of Features

Class 1: Basic Run Control
- Static Debug
  - r/w regs. & mem.
  - start/stop processor
  - hw/sw breakpoints
- Watchpoint Msg
- Ownership Trace Msg

Class 2: Instruction Trace
- Program Trace Msgs
- Read/Write Access
- Data Trace Msgs

Class 3: Read/Write Access
- Memory Substitution

Class 4: Memory and Port Substitution
- Port Replacement
Nexus Compliance Classifications 1,2

• **Class1:**
  - Read/ write user registers in debug mode
  - Read/ write user memory in debug mode
  - Enter a debug mode from reset
  - Enter a debug mode from user mode
  - Single step instructions in user mode and re-enter debug mode
  - Stop program execution on instruction/ data breakpoint and enter debug mode (minimum 2 breakpoints)
  - Ability to set breakpoints or watchpoints
  - Device identification
  - Ability to send out an event occurrence when watchpoint matches

• **Class2:**
  - All Class1 features plus …
  - Monitor process ownership while process runs in real-time
  - Monitor program flow while processor runs in real-time
Nexus Compliance Classifications 3, 4

- **Class3:**
  - All Class2 features plus …
  - Monitor data writes while processor runs in real-time
  - Read/write memory locations while processor runs in real-time
- **Class4:**
  - All Class3 features plus …
  - Program execution from Nexus port (memory substitution)
  - Ability to start traces upon watchpoint occurrence
- **Optional Features**
  - Ability to start memory substitution upon watchpoint occurrence
  - Monitor data reads while processor runs in real-time
  - Port Replacement and Port Sharing
  - Transmit data values for acquisition by tool (Data Acquisition)
- **An interface can be Class1, Class2, Class3, or Class4 compliant**
Nexus Messages consist of a 6-bit TCODE (Transfer Code) followed by a variable number packets.

Packet Types:
- **Variable**: Variable length packets, minimum length is 1.
- **Vendor-Fixed**: Fixed length field length is defined by chip vendor (by Nexus standard).
- **Vendor-Variable**: Variable length field, can be vendor defined as 0 length for unneeded fields.

Messages can be Sync or Non-sync (synchronized):
- Sync messages include full address
- Non-sync only include relative address change

Each message contains a source packet (vendor-fixed):
- Indication of source for each message (multi-Nexus)

Each message contains an optional Timestamp packet.
## Typical Nexus Message Format

<table>
<thead>
<tr>
<th>Program Trace - Indirect Branch Message</th>
<th>Direction: from target</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Minimum Packet Size (bits)</strong></td>
<td><strong>Packet Name</strong></td>
</tr>
<tr>
<td>0</td>
<td>TSTAMP</td>
</tr>
<tr>
<td>1</td>
<td>U-ADDR</td>
</tr>
<tr>
<td>1</td>
<td>I-CNT</td>
</tr>
<tr>
<td>0</td>
<td>B-TYPE</td>
</tr>
<tr>
<td>0</td>
<td>SRC</td>
</tr>
<tr>
<td>6</td>
<td>TCODE</td>
</tr>
</tbody>
</table>
Example System using Nexus5001 Debug Port

- I/O block can be a simple mux or complex shared Nexus block to transmit concurrent events information
- Nexus registers for each core would be memory mapped or accessed from JTAG
Multi-Client Debug Arch. - StarCore DSP

SC1400 DSP Platform

- StarCore DSP Core
- EOnCE
- MMU
- DPU
  - (Core Performance Profiling)
  - (Ownership Trace)
  - (Data Trace)
- Nexus Registered Interface (NRI)
  - Nexus Port
  - Nexus Control (NPC)
  - 4K Nexus Trace Buffer (NTB)
- Program Trace & Watchpoints
- Bus I/F
- Q2SB
- SkyBlue Interface
- JTAG RunCtrl
- JTAG Port
- AHB Bus
- Time stamp
- Nexus Port
Multi-Client Debug Arch. - PowerPC
Nexus Benefits

**MPU Vendors**
- Simplifies tools support
- Customer understanding of tools strategy
- Design re-use reduces time & cost
- Leverage best in-class tools
- Easier, faster check-out of tools on new architectures
- Ability to add features in standard method
- Same MCU in development & production
- Trace without the bus (pin overhead)

**Tool Vendors**
- Reduced development cost
- Rapid migration to new architectures
- Standard functions
- Opportunity to differentiate tools
- Opportunity to address customer requirements
- High performance – lower cost tools

**Users**
- Learning cycles – tools and architecture(s)
- Quicker time to market
- Development tool reuse
- Proven capabilities - multiple architectures
- Single small foot print interface
- Non-stop debugging, triggering & trace
Ongoing Nexus Development

Nexus Forum is addressing moving target of next generation processor and systems:

• Enhancements to existing 2003 std. definitions:
  – Defining bus monitor functions as alternative to processor functions
  – Better support for alternative processor data sizes (i.e. DSP)
  – Inclusion of branch instruction in I-CNT field
  – Minimal pin count interface (4 pins or less)

• Higher performance processor support:
  – High Speed (Gb/s+) I/O definition
  – Multi-threaded processor support

• Improved tool hardware interfaces (rapid prototyping, debug, calibration)

• Software API
  – Enhancing / Upgrading debug centric API

• Cooperation with related interest groups (MIPI, OCP-IP, …)
Mission Statement:
To define a global, open, embedded processor development interface standard for embedded control applications.

- RTOS Support and Analysis
- Real Time Debugging
- Hardware in the Loop
- Program Tuning
- Logic Analysis
- Run Control
- Prototyping

Development Tool Standard Focus
- Low Level API
- Mechanical Interconnect
- Silicon

For More information
www.nexus5001.org