ASHLING

THE DEVELOPMENT SYSTEMS COMPANY

NEXUS 5001 FORUM

A PROGRAM OF THE IEEE INDUSTRY STANDARDS AND TECHNOLOGY ORGANIZATION

IEEE-ISTO 5001™-1999, The Nexus 5001 Forum™ Standard providing the Gateway to the Embedded Systems of the Future.

DOC:GEPDIS_PAPER.RTF

Revision History

Version	**Date**	**Comments**
1.0	10th Jan 2000	Initial, based on CMN NEXUS white paper
1.1	13th Jan 2000	Minor edits, removed NEXUS/GEPDIS references

Authors

This paper was written by Hugh O'Keeffe. Hugh can be contacted as follows:

Hugh.OKeeffe@ashling.com
Ashling Microsystems Limited
National Technological Park
Limerick
Ireland
http://www.ashling.com
Tel:+353-61-334466
Fax:+353-61-334477

Hugh is the R&D Manager with Ashling Microsystems Ltd. based in the National Technological Park in Limerick Ireland. Hugh has a B. Eng. in Electronic Engineering from the University of Limerick. Hugh has extensive experience in both embedded systems design and the development of debug tools for embedded system designers. Hugh's team recently completed work on the first phase of the Ashling TriCore Development Toolset, this resulted in the worlds first In-Circuit Emulator and Source-level C/C++ debugger for the Infineon Technologies TriCore Architecture.

References

1. "*IEEE-ISTO 5001™-1999, the Nexus 5001 Forum™ Standard for a Global Embedded Processor Debug Interface*". Available at www.ieee-isto.org/Nexus5001.
2. *"The Evolution of Powertrain Microcontrollers and its impact on Development Processes and Tools"* by Gary Miller (Motorola), Kevin Hall (Hewlett Packard), Wayne Willis (Hewlett Packard) and Wilfried Pless (Hewlett Packard). Available at www.nexus-standard.org.
3. *"A New Development Tool Interface Paradigm"* by Ron Stence (Motorola) and Kevin Hall (Hewlett Packard). Presented at Embedded Systems Conference, San Jose, California, September 1999.

Table of Contents

1. Introduction and Summary

IEEE-ISTO 5001™-1999, the Nexus 5001 Forum™ Standard, previously known as GEPDISC (Global Embedded Processor Debug Interface Standard Consortium) or NEXUS has been created to provide a standard debug interface for embedded control applications. This debug interface can be used to connect tools to an embedded system to facilitate any of the following:

- Run-time control (debugging)
- Code execution trace capture and Data access trace capture
- Calibration (Data access on-the-fly)
- Logic Analysis
- Rapid-prototyping

The standard builds on the experience of a wide range of semiconductor vendors, development system companies and embedded systems design engineers (see http://www.ieee-isto.org/Nexus5001/membership.html for a full list of Nexus 5001 Forum™ members). The Nexus 5001 Forum™, which developed the standard, began work in 1998 with the aim of defining a common set of microcontroller on-chip debug features, protocols and interfaces for the development and debugging tools that are used by embedded systems developers.

Although the Nexus 5001 Forum™ originally targeted the requirements of the automotive industry, they has produced a scalable standard that addresses the needs of high-performance microcontrollers in all industry segments. Scalability allows tool vendors like Ashling to produce a family of tools with a range of performance levels and cost that can answer the needs of development project teams right through the design and verification cycle.

2. Why On-Chip debugging?

On-chip debugging support logic is required in today's (and tomorrow's) microprocessors because without such support logic it would be impossible or prohibitively expensive to produce debugging tools capable of supporting increasing clock speeds. Some of the main reasons for this are:

- The use of on-chip memories for program storage make it difficult (or impossible) for external tools, for example, Logic Analysers or In-Circuit Emulators (ICEs) to determine the actual instruction being currently executed as there is no external or off-chip visibility of the program address bus.
- Very high microprocessors frequencies mean that an ICE cannot implement breakpoints (or other functions) as the available timing budget is simply too short to ensure that ‘break-before-execute’ breakpoints are possible.
- Deep instruction pipelines, multiple-issue RISC architectures and on-chip caches can make it very difficult to determine what instructions were fetched and actually executed.
- The introduction of application-specific or customer-specific Systems-On-Silicon (SOCs), some with multiple processors, means that a uniform, reliable debugging interface is essential to reduce the task of redesigning the In-Circuit Emulator for each individual SOC design.

- Product development times are shortening and engineering teams are under more and more pressure to deliver quicker. Many of today’s products have a ‘window’ of sales opportunity in the market, missing this window due to development slippage can mean that the market for that product has gone!. Debug and Verification is a significant part of the development cycle, hence, an effective debug mechanism is vital.

3. The IEEE-ISTO 5001™-1999 feature set

The IEEE-ISTO 5001™-1999 feature set is based on today’s best on-chip debug implementations, the goal was to create the best possible debug feature set while minimising the required pin-count and die area. IEEE-ISTO 5001™-1999 is designed to be processor and architecture independent and to support multi-core or multi-processor designs. Physically, IEEE-ISTO 5001™-1999 defines a standard set of connectors for connecting the debug tool to the target or system under test. Logically, data is transferred using a packet based protocol. This protocol can be IEEE JTAG 1149.1 based or for high speed systems an auxiliary port can be used which supports full duplex, higher bandwidth transfers.

This section summarises the IEEE-ISTO 5001™-1999 feature set.

3.1 Run-time control

This is the most basic feature and is standard on all on-chip debug implementations. It allows the debug tool to start and stop the processor, to modify registers and to single-step (execute a single assembly instruction).

3.2 Memory Access

IEEE-ISTO 5001™-1999 supports memory access while the processor is running (“on-the-fly”). This allows the debug tool to read and write memory while the processor is running without impacting or intruding on the currently executing instructions. On-the-fly access is a very powerful feature and is a requirement for debugging truly real-time systems were it is not possible to halt the system under test.

3.3 Breakpoints

IEEE-ISTO 5001™-1999 provides support for breakpoints which allow the program to be halted when a specified event (breakpoint) has occurred. The event can be specified as code execution at a specified address or as a data access (read or write) to a specified address with a specified value (e.g. break when 0x55 is written to address 0x123456; break when 0xAA is read from 0xFFFF00). IEEE-ISTO 5001™-1999 breakpoints are similar to the hardware breakpoints found in other processor architectures and can be set in flash or ROM based memory. Watchpoints are a similar concept, however, when a watchpoint occurs a message is sent to the debug tool (as opposed to halting the processor).

3.4 Instruction or Program Trace

IEEE-ISTO 5001™-1999 uses the Branch-Trace technique to compress the information needed to trace program execution. Executed code address information is emitted via the IEEE-ISTO 5001™-1999 port at branch or exception instructions only; the debug tool interpolates the program trace for sequential (non-branch) instructions from a local image of code memory contents. This allows full reconstruction of the program flow by the debug tool.

3.5 Data Trace

This feature allows the debug tool to track real-time data accesses to memory locations. The trace can be qualified be specifying a specific range (start and stop address) and a specific access type (read or write).

3.6 Ownership Trace

The Ownership Trace feature allows a real-time operating system (RTOS) to identify the currently executing process or task to the debug tool. The RTOS simply writes to a predefined IEEE-ISTO 5001™-1999 register when switching tasks, this write forces an Ownership Trace message to be emitted from the IEEE-ISTO 5001™-1999 port. The message will contain an ID which identifies the Task or Process to the debug tool.

3.7 Memory Substitution and Port Replacement

This feature allows internal memory or port accesses to be implemented over the auxiliary IEEE-ISTO 5001™-1999 port. For example, this feature can be used to implement ROM patching, that is instead of reading on-chip ROM, the instruction will be fetched from the debug tool via the auxiliary port. Port replacement is useful when relatively slow I/O pins have a secondary function, for example, a IEEE-ISTO 5001™-1999 port function, it allows the debug tool to 'emulate' their primary functionality.

3.8 Data Acquisition

This feature was added to support rapid prototyping, it allows the rapid transfer of arbitrarily large amounts of data via the auxiliary port to the debug tools. Its uses a more efficient protocol than that used in Data Trace.

3.9 Software API

This is a low-level Application Programming Interface (API) that 'hides' the target specifics such as host connection (e.g. emulator) mechanism and processor specific IEEE-ISTO 5001™-1999 register details. This API is produced jointly by the tool and semiconductor vendor.

4. The IEEE-ISTO 5001™-1999 compliance classes

IEEE-ISTO 5001™-1999 is a scalable standard, there are currently four classes of compliance to the standard, ranging from the basic (JTAG-only) Class 1 up to Class 4.

4.1 Class 1

This class supports Run Time Control (run, stop, memory upload/download when the processor is halted, breakpoints, read or set registers) using the JTAG interface. Communications are half duplex only and bandwidth is limited. Trace is not supported.

4.2 Class 2

This class adds ownership trace and program trace and allows the auxiliary debugging port to be shared with "slow" I/O port pins. Ownership trace allows current task or current process trace for systems based on real-time kernels or operating-systems.

4.3 Class 3

This class adds data write trace and memory read/write on-the-fly without halting execution. Data read/write tracing, sharing of the Auxiliary port with High Speed I/O ports such as the address/data bus, and support for data acquisition (visibility of related data parameters stored in internal resources, typically related calibration variables) may also be optionally part of Class 3 compliance.

4.4 Class 4

The class adds memory substitution (fetching or reading data over the IEEE-ISTO 5001™-1999 auxiliary port) and allows tracing to be triggered by a watchpoint. Triggering memory substitution on a watchpoint is an optional feature of Class 4 compliance.

The main characteristics of each Class are summarised in Table 1.

IEEE-ISTO 5001™-1999 Class	**Class 1**	**Class 2**	**Class 3**	**Class 4**
Trace features	Trace not supported	Adds ownership trace and program trace via Auxiliary ports	Adds data write trace and read/write memory on the fly via Auxiliary ports	Allows tracing to be triggered by a watchpoint via Auxiliary ports
Debug communication method	Half-duplex communication (Limited bandwidth)	Communication may be full duplex using Auxiliary port (higher bandwidth)	Communication may be full duplex using Auxiliary port (higher bandwidth)	Communication may be full duplex using Auxiliary port (higher bandwidth)
Run-time control	Supports run time control features using JTAG interface	Supports run time control features using JTAG interface or Auxiliary port	Supports run time control features using JTAG interface or Auxiliary port	Supports run time control features using JTAG interface or Auxiliary port
Auxiliary Port Implementation	No Auxiliary Port	Allows Port sharing; the Auxiliary port may be shared with slow IO port pins (e.g. static Config pins that are latched at reset time).	Allows Port sharing with high speed I/O ports.	Allows Port sharing with high speed I/O ports.
Data acquisition	Not supported	Not supported	Supports data acquisition	Supports data acquisition
Memory Substitution	Not supported	Not supported	Not supported	Supports memory substitution (fetching or reading data over the IEEE-ISTO 5001™-1999 auxiliary port) Triggering memory substitution on a watchpoint is an optional feature of Class 4 compliance

Table 1: IEEE-ISTO 5001™-1999 Class Characteristics

Note that 'optionally' in the context for the IEEE-ISTO 5001™-1999 specification means that a processor of a specific class does not have to support the optional features to be compliant with that class.

5. IEEE-ISTO 5001™-1999. What's currently happening ?

IEEE-ISTO 5001™-1999 is now an official IEEE Industry Standards and Technology organisation (IEEE-ISTO) standard, see their website www.ieee-isto.org for more details. Formed in January 1999, the IEEE-ISTO is a not-for-profit corporation offering standards-related industry groups an innovative and flexible operational forum and support services. The IEEE-ISTO facilitates the activities that support the implementation and acceptance of standards in the marketplace. The standard **IEEE-ISTO 5001™-1999, the Nexus 5001 Forum™ Standard for a Global Embedded Processor Debug Interface** is now available for download from the Forum's website at www.ieee-isto.org/Nexus5001.

Ashling Microsystems, as a member of the Nexus 5001 Forum™, brings to the initiative its recent successful implementation of the On Chip Debug System (OCDS) built into Infineon Technologies' new TriCore Microcontroller/DSP. Through close co-operation with Infineon, Ashling was the first vendor to market a range of In-Circuit Emulators and development tools based on Infineon's OCDS system. Ashling's Vitra and Ultra emulators were the first products to successfully implement a real-time program execution trace. As full members of the Nexus 5001 Forum™, Ashling and Infineon have contributed extensively to the development of the standard to its current status.

Right now, Ashling's 32-bit emulator design team is developing a full-function In-Circuit Emulator and Source-Level Debugger for microprocessors that supports the IEEE-ISTO 5001™-1999 standard, based on Ashling's "Vitra" emulator architecture and "PathFinder" source-level debugger. Ashling's engineering team are working closely with *all* of the Semiconductor Vendors in the Nexus 5001 Forum™ to ensure that the new Ashling 'Universal Emulator' supports their IEEE-ISTO 5001™-1999 compliant processor architectures.

6. Conclusions

The IEEE-ISTO 5001™-1999 standard offers many benefits to the industry, including the Semiconductor Vendors, the Development Tool Vendors and the End Users. Silicon vendors now have a standard debug port design that can be used across different architectures, in addition this debug port will be known and easily supported by tool vendors. Tool vendors will now be able to design universal tools that can work with silicon from different vendors thus increasing the market for their tools. Finally, end users can now be assured of a standard, scalable toolset that covers all their needs and is available with first silicon.