Freescale Semiconductor
Application Note

Document Number: AN4088
Rev. 0, 10/2010

MPC5500/MPC5600 Nexus Support Overview

by: Randy Dees
Applications Engineering
Austin, Texas
USA

1 Introduction

The MPC5500 and MPC5600 devices conform to the IEEE-ISTO 5001™ debug standard. It was introduced to define a common debug interface across microcontroller (MCU) manufacturers. In addition to defining a standard method to output trace information, it also defined 4 classes of support that makes it easier to compare relative debug support level that a MCU supports. While the official title is "Standard for a Global Embedded Processor Debug Interface", it is commonly referred to as the Nexus debug Standard, or just Nexus. This application note describes the required and optional features that are supported by each of the Nexus classes and shows the support class for each of the devices in the MPC5500 and MPC5600 families.

Nexus supports multiple cores and Nexus clients on a single device. In some cases, different processing units within a MCU can support different classes of nexus support. In addition, there are multiple revisions of the IEEE-ISTO 5001™ standard, 1999, 2003, and 2010 versions. Some Nexus clients support the 5001-2003 standard and some support the 5001-2010 standard.[1]

Contents

1. The IEEE-ISTO 5001-2010 standard is expected to be released by the end of 2010.

2 MPC5500/MPC5600 Nexus architectural overview

The MPC5500 and MPC5600 devices implement a varying number of internal Nexus clients. Collectively, all of the JTAG and Nexus clients are known as the Nexus Development Interface (NDI). The typical interconnections between the different Nexus clients are shown in the following figures. Tools can communicate with each of these clients by selecting the client through the JTAG Controller (JTAGC) and the Nexus Port Controller (NPC).

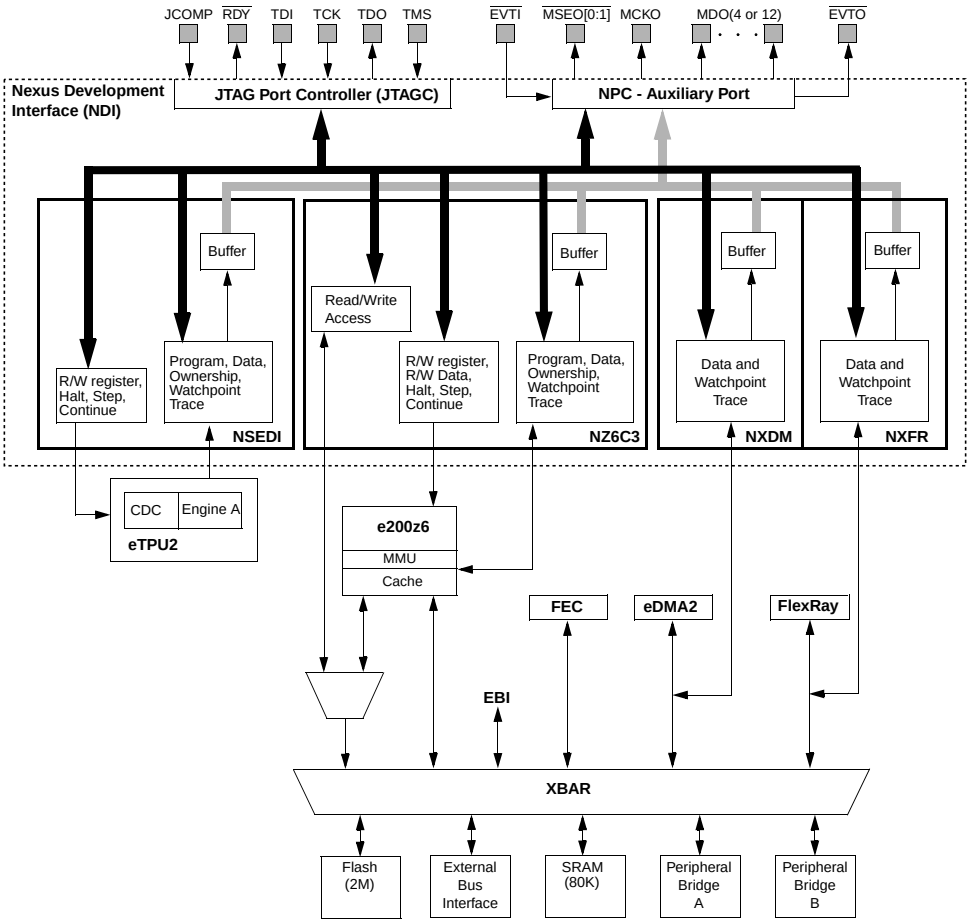


Figure 1. Typical MPC5500 Nexus Development Interface (MPC5567 shown)

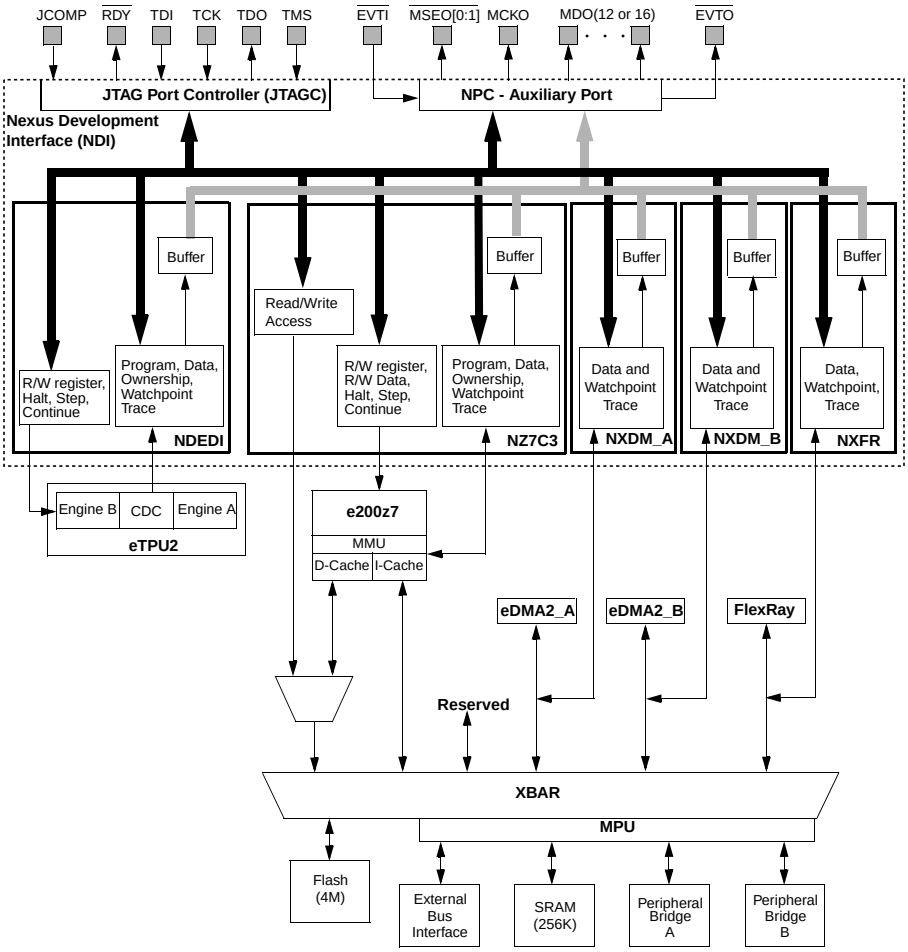


Figure 2. Typical MPC5600 Nexus Development Interface (MPC567xF shown)

The Nexus clients can consist of many types, but the typical clients are:

- e200z*x* core - In the MPC5500 and MPC5600 devices, the primary processing cores are based on Power Architecture®. Current available e200z*x* cores are the e200z0, e200z1, e200z3, e200z4, e200z6, e200z6 with VLE, and the e200z7. Some devices implement multiple cores, some as true dual (homogeneous) cores and some as a primary core and a secondary core. The secondary core is a smaller core intended for off-loading the primary core. The Nexus client is usually named for the core that it supports plus the Nexus class that the client supports. For example, the Nexus client for the e200z4 that supports Class 3 functionality is called the NZ4C3.

Table 1. All possible e200z*x* Core Nexus Clients

Core	Nexus Class Level Supported	Nexus Client	Full Client Name	Nexus Specification Implementation
e200z0	Class 1 only	—	No Nexus client	—
e200z0	Class 2+	NZ0C2	Nexus e200z0 Class 2+	IEEE-ISTO 5001™-2003
e200z0h	Class 2+	NZ0C2	Nexus e200z0h Class 2+	IEEE-ISTO 5001™-2003
e200z0	Class 3+	NZ0C3	Nexus e200z0 Class 3+	IEEE-ISTO 5001™-2010
e200z1	Class 2+	NZ1C2	Nexus e200z1Class 2+	IEEE-ISTO 5001™-2003
e200z335	Class 2+	NZ3C2	Nexus e200z3 Class 2+	IEEE-ISTO 5001™-2003
e200z3	Class 3+	NZ3C3	Nexus e200z3 Class 3+	IEEE-ISTO 5001™-2003

Core	Nexus Class Level Supported	Nexus Client	Full Client Name	Nexus Specification Implementation
e200z446/e200z448	Class 3+	NZ4C3	Nexus e200z4 Class 3+	IEEE-ISTO 5001™-2010
e200z6	Class 3+	NZ6C3	Nexus e200z6 Class 3+	IEEE-ISTO 5001™-2003
e200z6 with VLE	Class 3+	NZ6C3	Nexus e200z6 Class 3+	IEEE-ISTO 5001™-2003
e200z759[1]	Class 3+	NZ7C3	Nexus e200z7 Class 3+	IEEE-ISTO 5001™-2010
e200z760[1]	Class 3+	NZ7C3	Nexus e200z7 Class 3+	IEEE-ISTO 5001™-2010

1. Generically, both the e200z759 and the e200z760 are referred to as e200z7 to indicate both cores.

- eTPU - The Enhanced Timing Processor (eTPU or eTPU2) is a task-specific computing engine that is designed primarily for dealing with timing functions and includes some mathematics capabilities (including multiplication and division). The eTPU Nexus client is implemented as either a dual eTPU engine (Nexus Dual eTPU Development Interface - NDEDI) or a single eTPU engine client (Nexus Single eTPU Development Interface - NSEDI). In addition to the eTPU engine interface, the NDEDI/NSEDI interfaces with a secondary Nexus client, the Coherent Data-Parameter Controller (CDC). The CDC supports the shared memory and logic between the dual eTPU engines. On NDEDI devices, the eTPU Nexus interface appears to have three sub-clients (the two eTPU engines and the CDC). On NSEDI devices there are two (one engine and the CDC). All of the eTPU clients support the IEEE-ISTO 5001-2003 version of the Nexus standard.

Table 2. eTPU Nexus Clients

Implementation	Nexus Class Level Supported	Nexus Client	Full Client Name
Single eTPU engine	Class 1 only	NSEDI[1]	Nexus single eTPU Development Interface
Single eTPU engine	Class 3	NSEDI	Nexus single eTPU Development Interface
Dual eTPU engine	Class 3	NDEDI	Nexus Dual eTPU Development Interface

1. All of the development features for the eTPU are accessed through the eTPU Nexus client. However, some implementations do not support instruction or data trace (no Class 2 or above support).

- Bus Trace Clients - Another trace client available is a Nexus client that traces data accesses made through a crossbar bus port. Usually, these have been implemented on the master side of the crossbar switch for the eDMA module and the FlexRay communication interface. This allows master accesses made by these bus masters to peripherals or memory accessed though the crossbar switch bus (XBAR). Freescale has chosen to implement these interfaces on the master side of the crossbar for most devices, but this module can also be instantiated on the slave side of the crossbar switch on future devices, allowing trace of all accesses into a particular slave port of the crossbar switch bus. Bus trace clients are referenced by the the type of interface they support. For example, the Nexus eDMA Master XBAR client is referred to as the NXDM; the FlexRay Master XBAR trace client is the NXFR. One client that has been defined for use on the slave side of the XBAR is a Nexus trace client that traces access to a SRAM (Nexus XBAR Slave Port SRAM trace module - NXSS). All current versions of the XBAR Bus Monitor support the IEEE-ISTO 5001™-2003 version of the Nexus standard.

Table 3. Crossbar Bus Nexus Clients

Bus type	Nexus Class Level Supported	Nexus Client	Full Client Name
XBAR Master Side eDMA Bus Monitor	Class 3 features	NXDM	Nexus XBAR Master eDMA Trace Client
XBAR Master Side FlexRay Bus Monitor	Class 3 features	NXFR	Nexus XBAR Master FlexRay Trace Client
XBAR Slave Side SRAM Bus Monitor	Class 3 features	NXSS	Nexus XBAR Slave SRAM Trace Client

There are three versions of the IEEE-ISTO 5001 standard, 5001™-1999, 5001™-2003, and 5001™-2010. The 5001-1999 and the 5001-2003 are completely upwards compatible with each other, however, while the intent was to remain upward compatible from the previous versions of the standard, the 2010 version of the standard includes some new features that required changes

to some messages and additional bits added to registers. Most of the new message fields are optional and therefore backwards compatible, but there are some message fields that did change. Also, in some cases causes bit fields to move to different registers to support expansion of other bit fields within registers. See the device or core Reference Manuals for more information on each individual Nexus client implementation.

3 Nexus Class Definitions

To understand the level of debug support for a device, the Nexus standard divides the debug features into four classes. The four feature classes are Class 1 Basic Run Control, Class 2 Dynamic Debug (Instruction) trace, Class 3 Data Trace, and Class 4 Advanced Debug features. In the Nexus 5001 standard, some features are optional but most features are required. In addition, the standard defines a minimum set of resources required for some features.

In general, when features from a higher class are included in a lower Class device, (but not the full higher Class features), Freescale refers to the extra features as a "+" feature. For example, a Class 2+ devices includes all Class 2 features plus some features from either Class 3 or Class 4; however, all of the Class 3 or 4 features are not supported. Freescale implements the Class 3 features of Read/Write Access, (access to memory in real-time while the core is executing) on all MPC5500/MPC5600 devices that support Nexus. The Class 4 feature of enabling and disabling of trace via a watchpoint is included on all devices that support trace.

An overview of the Nexus classes is summarized in the table below.

Table 4. Nexus Feature/Class Overview

Class	Type	Required Features	Optional Features
1	Static debug (Basic Run Control)	Read/write registers and memory	—
		Start and stop program execution, including single instruction execution	
		Control entry and exit debug mode from both reset and user modes	
		Set breakpoint and watchpoints	
		Read Nexus device identification	
2	Dynamic Debug	Real Time Ownership Trace	Port replacement
		Real Time Program Trace	
		Watchpoint Messaging	
3	Data Collection	Data Trace of memory/peripheral writes	Data Trace of memory/peripheral reads
		Dynamic memory read and write	Data Acquisition

Class	Type	Required Features	Optional Features
4	Advanced Debug	Complex triggering control of trace	Start/stop memory substitution on watchpoint
		Memory Substitution	

3.1 Nexus Class 1 - Basic Run Time Control

Nexus Class 1 provides the minimal debug capability and includes primarily stop mode debug features (static debug), but some dynamic debug features are required. Most features require that the device is stopped, either at a breakpoint or immediately after reset. Required capabilities include the ability to start and stop the target system, set breakpoints, (at least 2 are required), reset the target system, read and write memory, and execute single instructions. All of these features are also supported in the higher classes (Class 2, Class 3, and Class 4). The table below summarizes the required static debug features for Class 1 devices.

Table 5. Class 1 (and higher) Required Static Debug Features

Development Feature	Description	Required (R) or Optional (O) Feature	Class 1	Class 2	Class 3	Class 4[1]
Register read/write	Ability to read and write user registers in debug (stopped) mode.	R	√[2]	√	√	√
Memory read/write	Ability to read and write user memory in debug (stopped) mode.	R	√	√	√	√
Debug entry from reset	Enter debug mode from reset.	R	√	√	√	√
Debug entry from running	Enter debug mode from user mode.	R	√	√	√	√
Exit to run mode	Exit a debug mode to a user mode.	R	√	√	√	√
Single step	Single step instruction in user mode and re-enter debug mode.	R	√	√	√	√
Breakpoint	Stop program execution on instruction/data breakpoint and enter debug mode (a minimum of 2 breakpoints are required).	R	√	√	√	√

1. Most Class 4 features are not supported on the MPC5500 or MPC5600 devices.
2. √ indicates that the feature is implemented on all devices.

The table below shows the required dynamic debug features for a Class 1 device.

Table 6. Class 1 (and higher) Dynamic Debug Features

Development Feature	Description	Required (R) or Optional (O) Feature	Class 1	Class 2	Class 3	Class 4[1]
Breakpoints/Watchpoints	Ability to set breakpoints and watchpoints	O	√[2]	√	√	√

Development Feature	Description	Required (R) or Optional (O) Feature	Class 1	Class 2	Class 3	Class 4[1]
Device Identification Message[3]	Access to a device identification either through a JTAG ID register or an automatic message	R	√	√	√	√

1. Most Class 4 features are not supported on the MPC5500 or MPC5600 devices.
2. Since Class 1 does not require a Nexus Auxiliary Output port, watchpoints are signaled by asserting the Event Out ($\overline{\text{EVTO}}$) signal.
3. The Device ID message is optional on JTAG based devices.

3.2 Nexus Class 2 - Dynamic Debug (Instruction Trace)

Nexus Class 2 requires that all Class 1 features be supported and adds support for Watchpoints Messages, Ownership Trace messages, and Program Trace messages. An Auxiliary output port is required to transmit this data outside of the device. All Class 2 or higher cores and modules support all features of Class 2.

NOTE

The Nexus standard allows trace data to be stored in an internal memory space and read via the JTAG interface. This option is currently not supported in the MPC5500/MPC5600 devices.

Table 7. Class 2 (and higher) Dynamic Debug Features

Development Feature	Description	Required (R) or Optional (O) Feature	Class 1	Class 2	Class 3	Class 4[1]
Watchpoint Messages	Capability to signal the occurrence of watchpoint with a signal ($\overline{\text{EVTO}}$) and/or a watchpoint (an address was accessed, either from execution or by a load/store access).	R[2]	—	√	√	√
Ownership Trace Messages	Capability to monitor the process identifier in real-time with a message.	R[2]	—	√	√	√
Program Trace Messages	Monitor program flow in real-time.	R[2]	—	√	√	√
Port Replacement	Low Speed Input/Output port replacement and High Speed IO port sharing	O	—	Not Supported[3]	Not Supported[3]	Not Supported[3]

1. Most Class 4 features are not supported on the MPC5500 or MPC5600 devices.
2. Required for Class 2 and above.
3. This optional feature is not implemented.

3.3 Nexus Class 3 - Data Trace

Nexus Class 3 requires all Class 2 features and adds the capability of performing data trace of writes by the MCU core. Tracing of reads is an optional features that is not required, but is supported on current devices and modules that support Nexus Class 3 features.

Table 8. Class 3 (and higher) Dynamic Debug Features

Development Feature	Description	Required (R) or Optional (O) Feature	Class 1	Class 2	Class 3	Class 4[1]
Data Trace (Writes only)	Monitor data writes while process runs in real-time	R[2]	—	—	√	√
Read/Write Access (RWA)	Read and write memory locations without stopping the core CPU.	R[2]	—	Supported[3]	√	√
Data Trace (reads)	Monitor data reads while the CPU runs in real-time.	O	—	—	Supported[4]	Supported[4]
Data Acquisition	Transmit data values for acquisition by the tool	O	—	—	Supported[5]	Supported[5]

1. Most Class 4 features are not supported on the MPC5500 or MPC5600 devices.
2. Required to be supported for Class 3 and above.
3. The Nexus Read/Write Access feature is supported by all Freescale Automotive devices that support Class 2 and above.
4. This is an optional feature that is implemented.
5. This is an optional Nexus feature that is supported on devices that use either the e200z4 or the e200z7 core.

3.4 Nexus Class 4 - Advanced Features

Nexus Class 4 requires all Class 3 features and adds some very advanced features such as memory substitution and the capability of enabling or disabling trace upon a watchpoint occurrence. The ability to start or stop trace, (either program or data trace depending on the class device), based on a watchpoint is supported on all MPC5500 and MPC5600 devices that support Class 2 or Class 3.

Table 9. Class 4 (and higher) Dynamic Debug Features

Development Feature	Description	Required (R) or Optional (O) Feature	Class 1	Class 2	Class 3	Class 4[1]
Memory Substitution	Allows program execution directly from the Nexus port upon either a reset or an exception.	R[2]	—	—	—	√
Start/stop trace on watchpoint occurrence	Supports the ability to start or stop ownership, program, or data trace upon a watchpoint occurrence.	R[2]	—	Supported[3]	Supported[3]	√[4]
Start/stop memory substitution based on watchpoint occurrence	Ability to start or stop memory substitution upon a watchpoint occurrence.	O	—	—	—	O

1. Most Class 4 features are not supported on the MPC5500 or MPC5600 devices.
2. Required to support Class 4.
3. This feature is not required or optional, but is implemented.
4. This is a required Class 4 feature, but is supported on devices that support either Class 2 or Class 3 features.

4 Nexus class 1 JTAG-only operation

For Nexus class 1 (run control only) operation, the JTAG pins can be used for standard, stopped mode debug operations. In this mode, Nexus does not have to be enabled and the auxiliary port pins can be ignored by the debugger. Since there is no JTAG standard connector pinout definition, Freescale recommends that class-1-only debuggers use the same connector options as full

Nexus debuggers. This allows customers to avoid putting two separate connectors on boards for Nexus versus JTAG-only debug tools. However, this document includes a definition for a 14-pin BERG style connector (2 × 7, 0.1" pin centers) for extremely low-cost applications that cannot use one of the full Nexus connectors.

NOTE

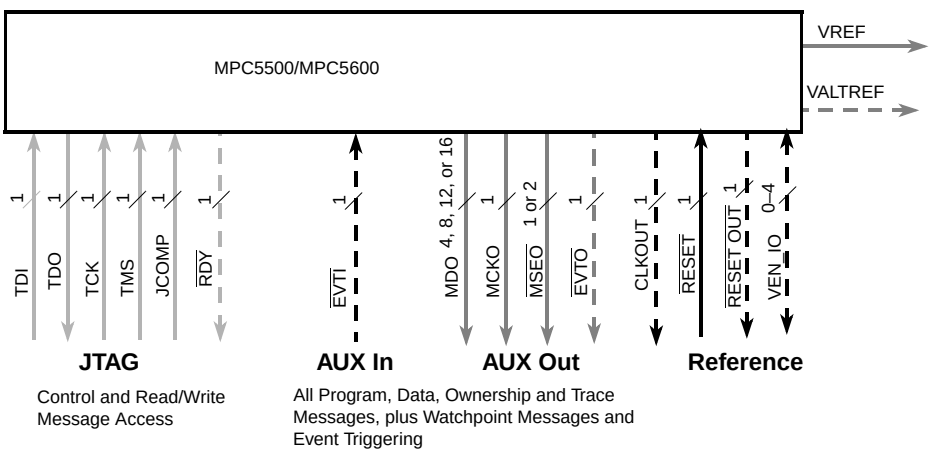
The Nexus class 3 feature read/write access is accessible through a JTAG connection. Other Class 3 features are not available unless a full Nexus connector is used.

NOTE

It is recommended that all the full Nexus signals be made available (somewhere) on production boards to facilitate debugging new boards and analysis of errors in software, even on boards that have restricted space and provide a JTAG-only connection. If all of the Nexus signals are available on the production board, an adapter could be built to provide a Nexus connection on boards that do not have a complete footprint for one of the standard Nexus connectors. Likewise, the JTAG connector does not have to be populated on production boards, and could even use a smaller connector footprint. An adapter could then be used to provide standard debug connections.

5 Nexus signals for the MPC5500/MPC5600 family

The figure below shows a block diagram of the MPC5500/MPC5600 family Nexus signals. The pins are divided between the JTAG/OnCE port, the auxiliary input port, the auxiliary output port, and reference signals. The $\overline{\text{RDY}}$ pin is grouped with the JTAG pins, but it is used for Nexus block transfers done through the JTAG port (read/write messaging).



Note: Dashed signals are optional in the Nexus standard. Tool_IO pins do not have any connectivity to the MCU.

Figure 3. Pin interface for an MPC5500/MPC5600 Nexus combined JTAG/AUX port

The auxiliary output port can be implemented with four, eight, twelve, or sixteen Message Data Output signals (MDO) and either one or two Message Start/End Output signals ($\overline{\text{MSEO}}$).

Some of the auxiliary output port functions on the MPC5500/MPC5600 family are shared with other pin functions.

6 Nexus class support summary

The table below is a summary of all of the Nexus Class supported by each of the cores and and other Nexus clients that are implemented on each device in the MPC5500 and MPC5600 families of devices. All of the modules support the IEEE-ISTO 5001-2003™ standard unless otherwise noted. Some cores support the IEEE-ISTO 5001-2010™ version of the standard and some message formats and registers have minor changes.

Table 10. MPC5500 and MPC5600 Nexus Class Summary

Device	Package	Voltage	Port Width	Number of MSEO signals	JCOMP Implemented	Nexus Client Module					
						Core 0	Core 1	eTPU	eDMA	FlexRay	SRAM
						NZxCx	NZxCx	NSEDI/ NDEDI[1]	NXDM	NXFR	NXSS
MPC5510	All	3.3V[4]/5V	4/8	1	Yes	e200z1 Class 2+[2]	e200z0 Class 2+[2]	—	—	—	—
MPC5534	All	3.3V	4/12[3]	2	Yes	e200z3 Class 3+	—	Class 3+	—	—	—
MPC5553	All	3.3V	4/12	2	Yes	e200z6 Class 3+	—	Class 3+	Class 3+	—	—
MPC5554	All	3.3V	4/12	2	Yes	e200z6 Class 3+	—	Class 3+	Class 3+	—	—
MPC5561	All	3.3V	4/12	2	Yes	e200z6 Class 3+	—	-	Class 3+	—	—
MPC5565	All	3.3V	4/12	2	Yes	e200z6 Class 3+	—	Class 3+	Class 3+	—	—
MPC5566	All	3.3V	4/12	2	Yes	e200z6 Class 3+	—	Class 3+	Class 3+	—	—
MPC5567	All	3.3V	4/12	2	Yes	e200z6 Class 3+	—	Class 3+	Class 3+	—	—
MPC560xB/C	All except 208 MAP-BGA	3.3V[4]/5V	0	1	No	e200z0 Class 1+	—	—	—	—	—
	208 MAP-BGA	3.3V[4]/5V	4	1	No	e200z0 Class 2+	—	—	—	—	—
MPC560xP	All	3.3V[4]/5V	4	2	No	e200z0 Class 2+	—	—	—	—	—
MPC560xS	All except 176 LQFP and 208 MAP-BGA	3.3V[4]/5V	0	1	No	e200z0 Class 2+[5]	—	—	—	—	—
	176 LQFP[6], 208 MAP-BGA	3.3V[4]/5V	4	1	No	e200z0 Class 2+	—	—	—	—	—

Device	Package	Voltage	Port Width	Number of MSEO signals	JCOMP Implemented	Nexus Client Module					
						Core 0	Core 1	eTPU	eDMA	FlexRay	SRAM
						NZxCx	NZxCx	NSEDI/ NDEDI[1]	NXDM	NXFR	NXSS
MPC563xM	144 LQFP, 176 LQFP, 208 MAP-BGA	3.3V	4[7]	2	Yes	e200z335 Class 2+	—	Class 1	—	—	—
MPC564xA	All	3.3V	4/12[8]	2	Yes	e200z448 Class 3+[9]	—	Class 1			—
MPC564xL	144 LQFP	3.3V	4	2	Yes	e200z446 Class 3+[9,10]	—	—	—	—	—
	257 MAP-BGA	3.3V	4/12								
MPC5668E/G	208 MAP-BGA	3.3V	0	0	Yes	e200z6 Class 3+[5]	e200z0 Class 2+[5]	—	—	—	—
	256 MAP-BGA	3.3V	12	2	Yes	e200z6 Class 3+	e200z0 Class 2+	—	—	—	—
MPC567xF	All	3.3V	12/16	2	Yes	e200z7 Class 3+[9,10]	—	Class 3+	Class 3+ (2)	Class 3+	—

1. Device with two eTPU modules use the Nexus Dual eTPU Interface. Devices with a single eTPU engine use the Nexus Single eTPU Interface.
2. Only one core can be traced at a time.
3. Only 4 MDO signals are available in the 208 MAPBGA package.
4. Based on the IO voltage of the device. Generally, a 5 volt supply is used by most devices, therefore most systems will have a 5V JTAG/Nexus interface.
5. While the module supports trace, the package does not include an Auxiliary Output Port to transmit the information.
6. Nexus functions are shared with alternate functions.
7. 12 MDO signals are available in the 496 CSP package.
8. Significant IO is removed if 12-bit MDO is enabled in some packages.
9. The e200z4 and e200z7 cores support the 2010 version of the Nexus 5001 standard.
10. Data Acquisition supported.

7 References

For more information on Nexus and the Nexus implementation on the MPC5500 and MPC5600 families of devices, see the device reference manual and the additional documents listed in the table below.

Table 11. Nexus References

Document	Title	Location/Availability
e200z0CORERM	e200z0 Power Architecture Core Reference Manual	Freescale web site http://www.freescale.com
e200z1RM	e200z1 Power Architecture Core Reference Manual	
e200z3coreRM	e200z3 Power Architecture Core Reference Manual	
e200z4RM	e200z4 Power Architecture Core Reference Manual	
e200z6RM	e200z6 PowerPC Core Reference Manual	
e200z7RM	e200z760 Power Architecture Core Reference Manual	
AN2614	MPC553x, MPC555x, and MPC556x Family Nexus Interface Connector	
AN3968	Nexus Interface Connector for the MPC5674F	
AN3970	MPC5500 and MPC5600 Nexus/JTAG Client Access Overview	
AN4030	Using the Development Tool Semaphore Module on the MPC564xA	
AN4088	Nexus Overview and Nexus Support for the MPC5500 and MPC5600 Families	
IEEE-ISTO 5001-1999	The Nexus 5001 Forum™ Standard for a Global Embedded Processor Debug Interface, Version 1	IEEE-ISTO Nexus web site http://www.nexus5001.org
IEEE-ISTO 5001-2003	The Nexus 5001 Forum™ Standard for a Global Embedded Processor Debug Interface, Version 2.0	
IEEE-ISTO 5001-2010	The Nexus 5001 Forum™ Standard for a Global Embedded Processor Debug Interface, Version 3	Not yet released
IEEE Std 1149.1-1990	IEEE Standard Test Access Port and Boundary-Scan Architecture	IEEE web site http://www.ieee.org

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
1-800-441-2447 or +1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: AN4088
Rev. 0, 10/2010