

Migration from IBM 750FX to MPC7447A

by *Douglas Hamilton*
European Applications Engineering
Networking and Computing Systems Group
Freescale Semiconductor, Inc.

1 Scope and Definitions

The purpose of this application note is to facilitate migration from IBM's 750FX-based systems to Freescale's MPC7447A. It addresses the differences between the systems, explaining which features have changed and why, before discussing the impact on migration in terms of hardware and software.

Throughout this document the following references are used:

- 750FX—which applies to Freescale's MPC750, MPC740, MPC755, and MPC745 devices, as well as to IBM's 750FX devices. Any features specific to IBM's 750FX will be explicitly stated as such.
- MPC7447A—which applies to Freescale's MPC7450 family of products (MPC7450, MPC7451, MPC7441, MPC7455, MPC7445, MPC7457, MPC7447, and MPC7447A) except where otherwise stated. Because this document is to aid the migration from 750FX, which does not support L3 cache, the L3 cache features of the MPC745x devices are not mentioned.

Contents

1. Scope and Definitions	1
2. Feature Overview	2
3. 7447A Specific Features	12
4. Programming Model	16
5. Hardware Considerations	27
6. Revision History	30

2 Feature Overview

There are many differences between the 750FX and the MPC7447A devices, beyond the clear differences of the core complex. This section covers the differences between the cores and then other areas of interest including the cache configuration and system interfaces.

2.1 Cores

The key processing elements of the G3 core complex used in the 750FX are shown below in [Figure 1](#), and the G4 complex used in the 7447A is shown in [Figure 2](#).

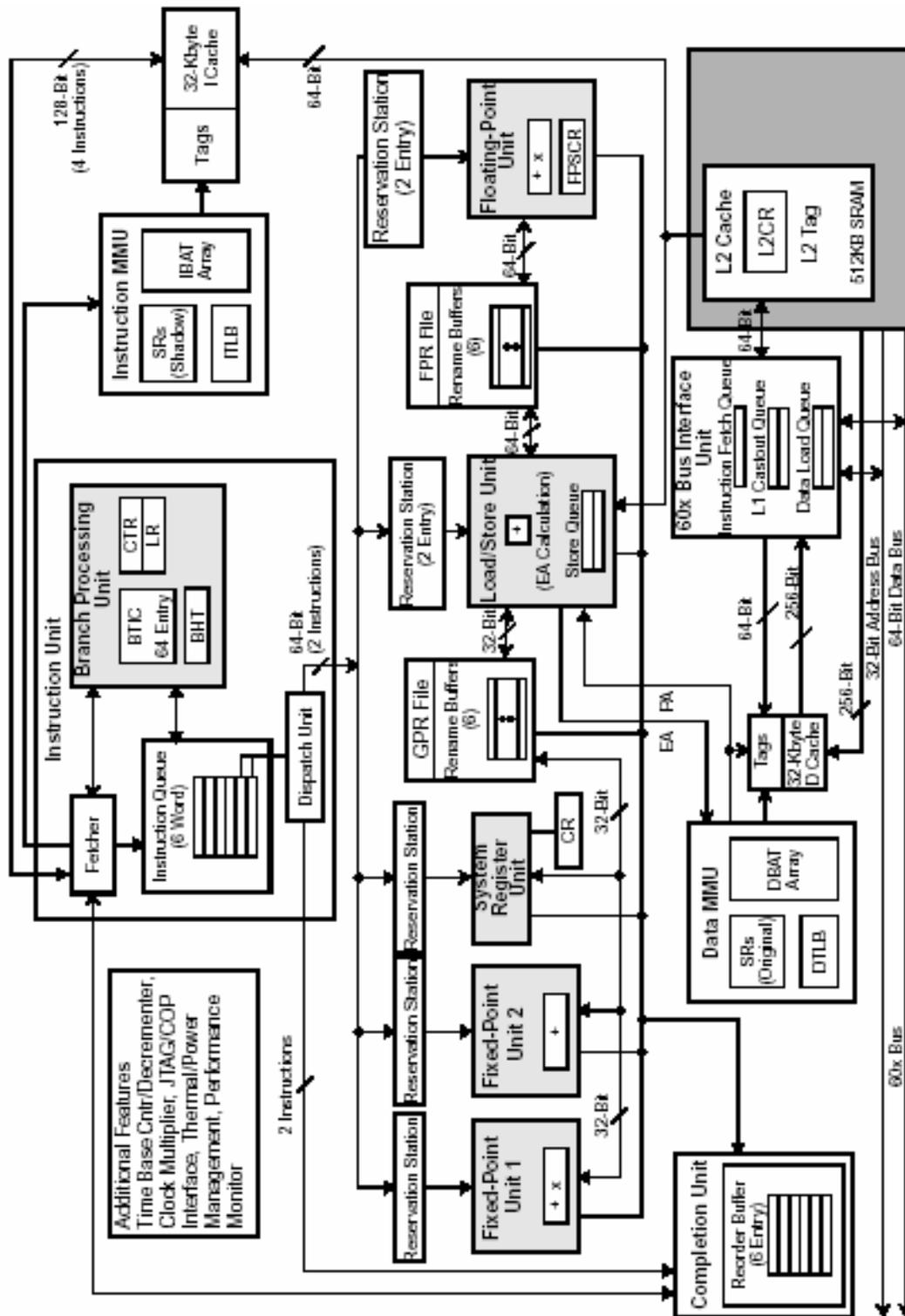
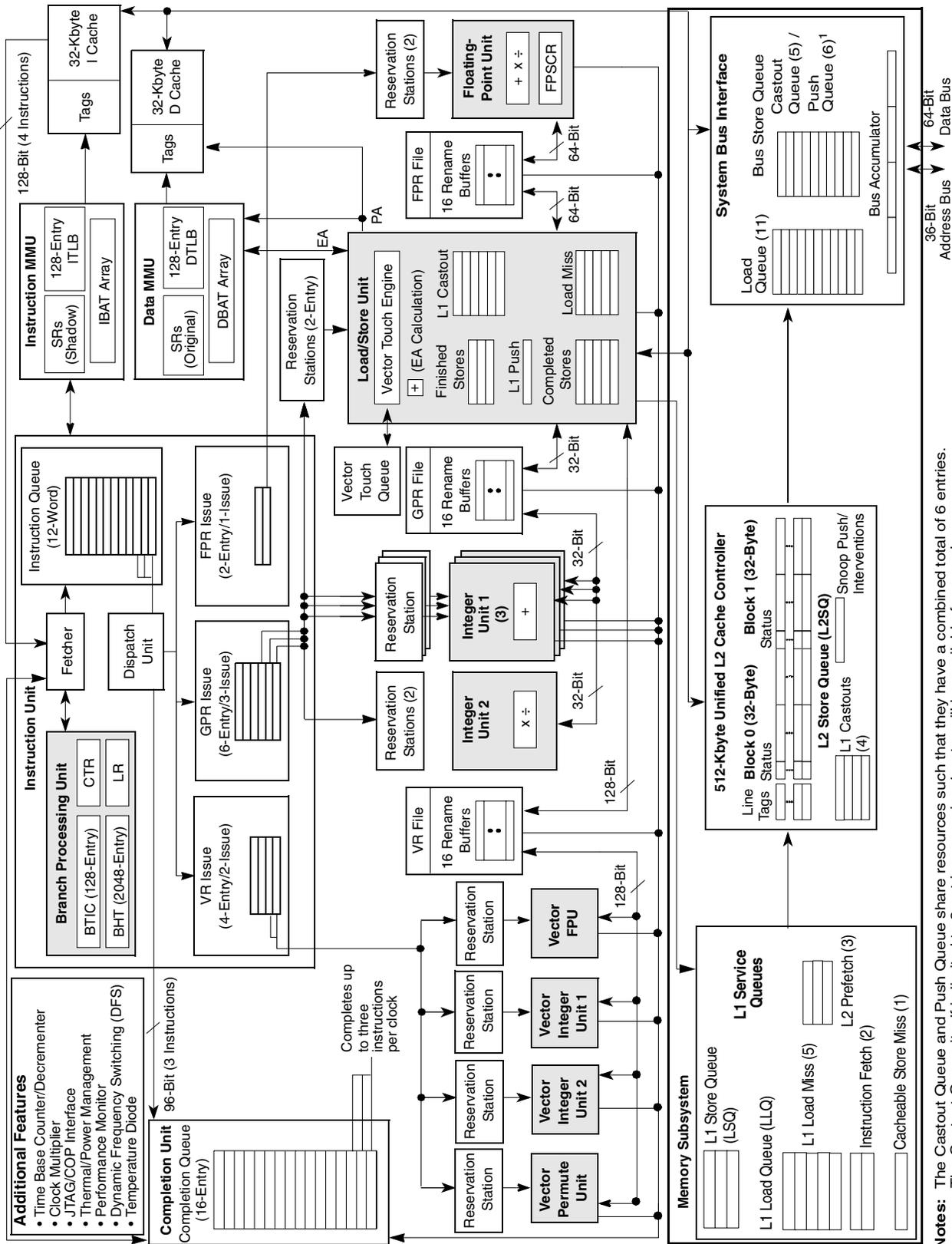


Figure 1. 750FX Core



Notes: The Castout Queue and Push Queue share resources such that they have a combined total of 6 entries. The Castout Queue itself is limited to 9 entries, ensuring 1 entry will be available for a push.

Figure 2. MPC7447A

2.1.1 Integer Units

IU1 and IU2 denote fixed point units 1 and 2 which are the complex and simple integer units respectively. The multiply and divide instructions of IU1 are multi-cycle, whereas all other operations are complete in a single cycle. Both the integers operate on 32 32-bit registers. [Table 1](#) shows the operations that each fixed unit can perform. Each unit consists of three units: adder/comparator, logical, and shift/rotate units. In addition to these standard units, FXU1 also has a multiply/divide unit.

Table 1. Fixed Point Unit Operations

	IU1	IU2
Add, shift, logical functions	Yes	Yes
Multiply/divide	Yes	No

Like the 750FX, the MPC7447A has one complex integer unit with the same functionality as IU1. However, it has three simple integer units like the IU2 instead of one. A good compiler can take advantage of these features; when presented with a combination of instructions with multi-cycle latencies that would tie up two of the integer units, the remaining units can start executing, thus preventing stalling. Also, the MPC7447A has sixteen GPR-rename buffers to support the 16-entry completion queue, as compared to the 6-entry completion queue for the 750FX. The floating point can also source rename buffers as a source operand without waiting for the value to be committed and retrieved from GPR.

2.1.2 Floating-point Units

The 750FX floating-point unit has 32 64-bit registers for single-precision and double-precision IEEE-754 standards. Different operations have various latencies associated with them due to the three-stage pipeline with multiply, add, and normalize stages. The latency/throughput varies, from 3/1 clock cycles for single multiply-add, increasing to 4/1 clocks for double-multiply and double-multiply-add, because two cycles are required in the multiply unit.

The MPC7447A floating-point unit meets the same standards for IEEE-754 precision and, in addition, has an increased pipeline depth of five stages to allow even double-precision calculations to have a one-cycle throughput. Although the latency is increased, the overall throughput is better for the majority of double-precision calculations. The floating point can also source rename buffers as a source operand without waiting for the value to be committed and retrieved from FPR.

2.1.3 Instruction Queues

The instruction queue in the 750FX can hold up to six instructions. While the instruction queue depth allows, the instruction fetcher retrieves up to four instructions maximum per clock. Two instructions can be dispatched simultaneously to fixed or floating-point units, the branch processing unit, and the load/store unit, to execute in a four-stage pipeline containing fetch, dispatch, execute, and complete stages.

The MPC7447A offers a 12-slot instruction queue with a maximum of four fetches per cycle and can dispatch up to three instructions per cycle to any of the 11 instruction units: the branch processing unit, the four integer units, the floating-point unit, the four 128-bit (Altivec) vector units, and the load/store unit.

2.1.4 Branch Processing Unit

The branch processing unit found in the 750FX can process one branch while resolving two speculative branches per cycle. It uses a 512-entry-deep BHT for dynamic branch prediction to produce four possible outcomes: not taken, strongly not taken, taken, and strongly taken. It incorporates a 64-entry BTIC to reduce branch delay slots. It incorporates a 64-entry BTIC which reduces idle cycles caused by branch instructions while new instructions from the branch target are fetched from the instruction cache. A hit in the BTIC usually allows the first two instructions at the branch target to be supplied on the next clock cycle, giving enough time for the next two instructions to be fetched from the instruction cache without any idle cycles in the instruction stream.

In contrast, the MPC7447A processes one branch per cycle like the 750FX but can resolve three speculative branches per cycle. The increased BHT with 2048 entries offers the same four predication states but with the advantage of the larger size. In addition to these four predication states, the BHT, using `HID0[BHTCLR]`, can be cleared to a fifth state: weakly not taken. The BTIC is twice the size of the 750FX, providing 128 entries arranged as 32 sets, using a four-way, set-associative arrangement.

2.1.5 Completion Unit

In the 750FX, the completion unit works with the dispatch unit to track dispatched instructions and retire them, in order, to the completion queue. Up to two instructions can be retired per clock cycle. The instruction is removed from the completion queue if the rename buffers have been freed and results have been written into processor registers such as GPRs, FPRs, LR, and CTR.

For the MPC7447A, due to deeper pipelines, we can have up to 16 instructions at some stage of pipeline processing and retire a maximum of three instructions per clock to one of the 16 completion queue slots.

2.2 Pipeline Comparison

The difference in pipeline depths between the 750FX and the MPC7447A is significant. The 750FX pipeline is four stages, including instruction fetch, dispatch/decode, execute, and complete/write-back. The pipeline diagram for the 750FX is shown in [Figure 3](#). Note that the 750FX has a maximum depth of six stages when using the floating-point unit.

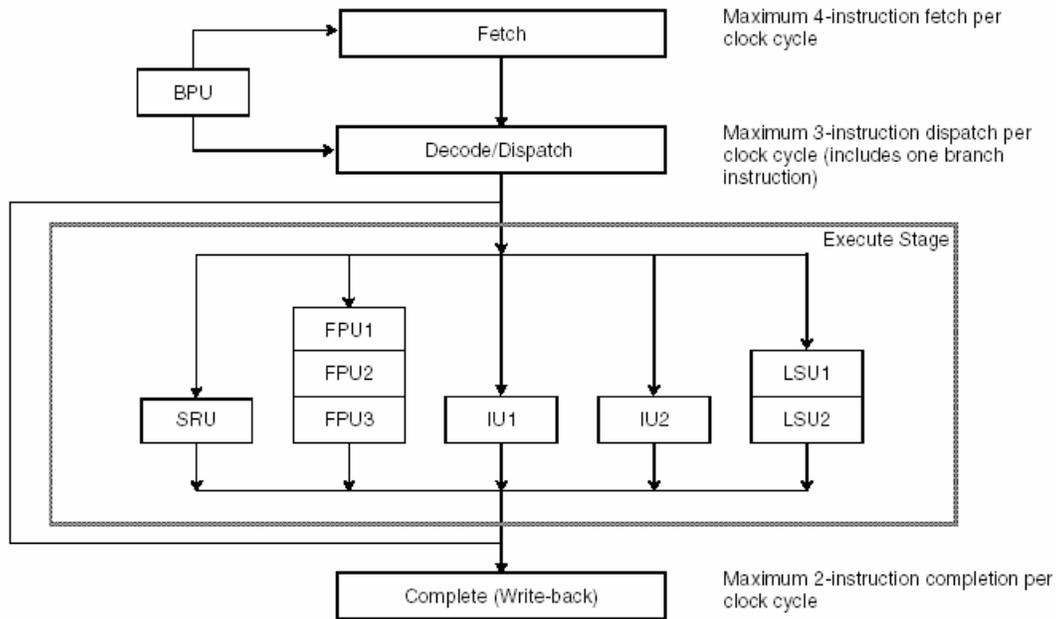


Figure 3. – IBM750FX Pipeline Diagram

If branch prediction does not work well for a particular application, then having such a short pipeline is advantageous due to a fairly small pipeline flushing penalty. However, branch prediction and modern compilers can prevent frequent pipeline flushes, so the completion rate of two instruction retirements per clock becomes more of a performance bottleneck.

With a minimum depth of seven stages, the MPC7447A pipeline, shown in [Figure 4](#), boasts efficient use of its additional hardware resources by dispatching three instructions per cycle to its execution units as well as the ability to retire three instructions per cycle. The extra pipeline depth allows the MPC7447A to run at a higher frequency (up to 1.5GHz). The deeper pipeline is more efficient; reducing the latency of many instructions, including several floating-point and complex integer instructions. Compilers can take advantage of the extended pipeline to potentially reach a target maximum of sixteen instructions are in flight at any one time.

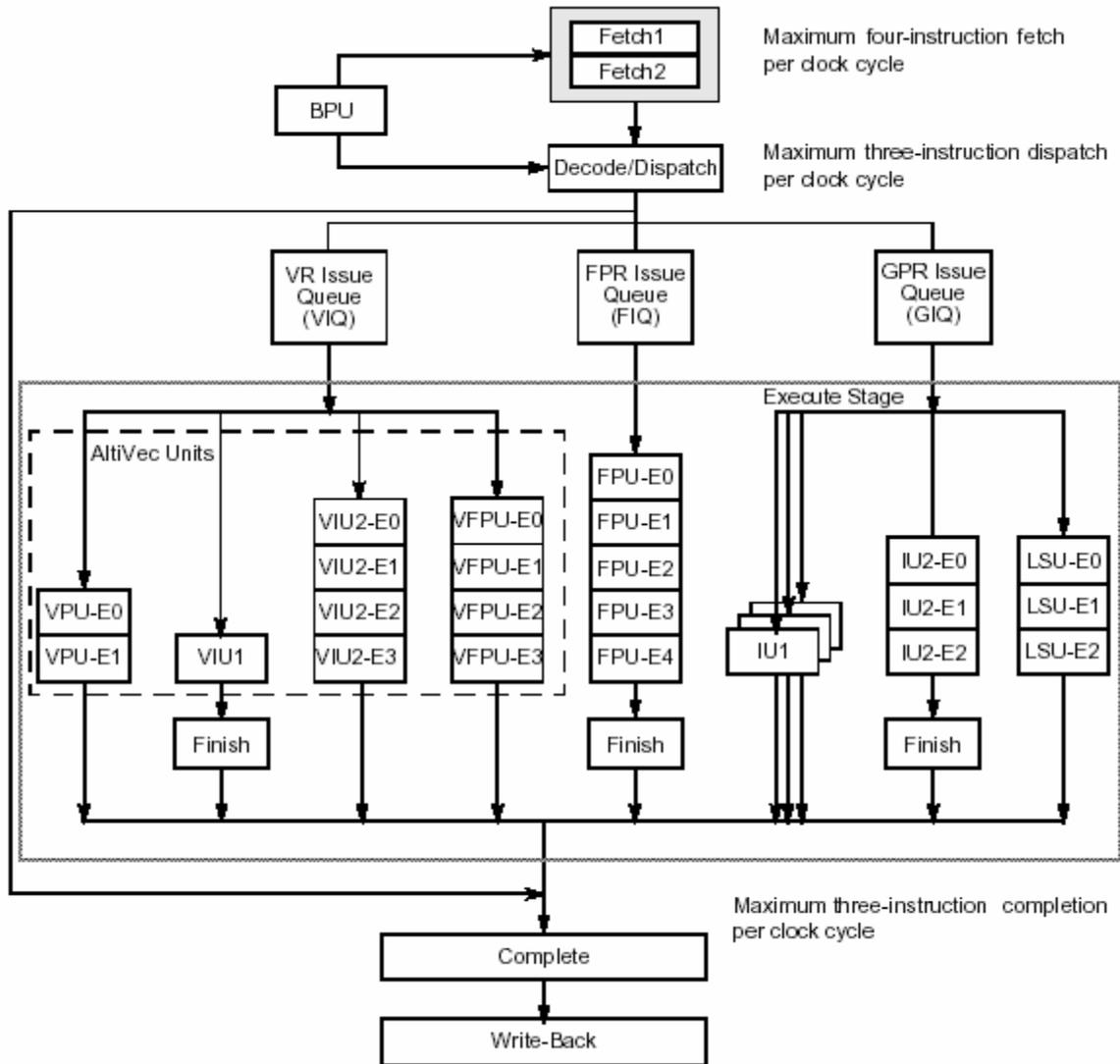


Figure 4. – MPC7447A Pipeline Diagram

2.3 L1 and L2 Cache

Table 2 summarizes the differences between the 750FX and the MPC7447A inL1 and L2 cache configuration.

Table 2. – L1, L2 Cache Configurations

	750FX	MPC7447A
L1		
Size, configuration	32K-byte instruction, 32-Kbyte data, eight-way, set associative	32-Kbyte instruction, 32-Kbyte data, eight-way, set associative
Memory Coherency	MEI (data only)	MESI (data only)

Table 2. – L1, L2 Cache Configurations (continued)

Locking	Completely	By way
Replacement policy	Pseudo-least-recently used (PLRU)	Pseudo-least-recently used (PLRU)
Per page/block write configuration	Write-back or write-through (data)	Write-back or write-through (data)
L2		
Size, configuration	512-Kbyte, two-way, set-associative Two 32-byte blocks/line	512-Kbyte, eight-way, set-associative (MPC7447A) Two 32-byte blocks/line
Memory coherency	MEI	MESI
Locking	Yes, independently by way	Completely
Replacement policy	Cast out replacement only	3-bit counter or pseudo random
Parity	8 bits/64 bytes on tags	8 bits/64 bytes on tags and data

2.4 MMU

Figure 5 shows the standard PowerPC MMU translation method. The presence of TLBs and page-table search logic are optional although both the IBM 750CX and Freescale 7447A implement them.

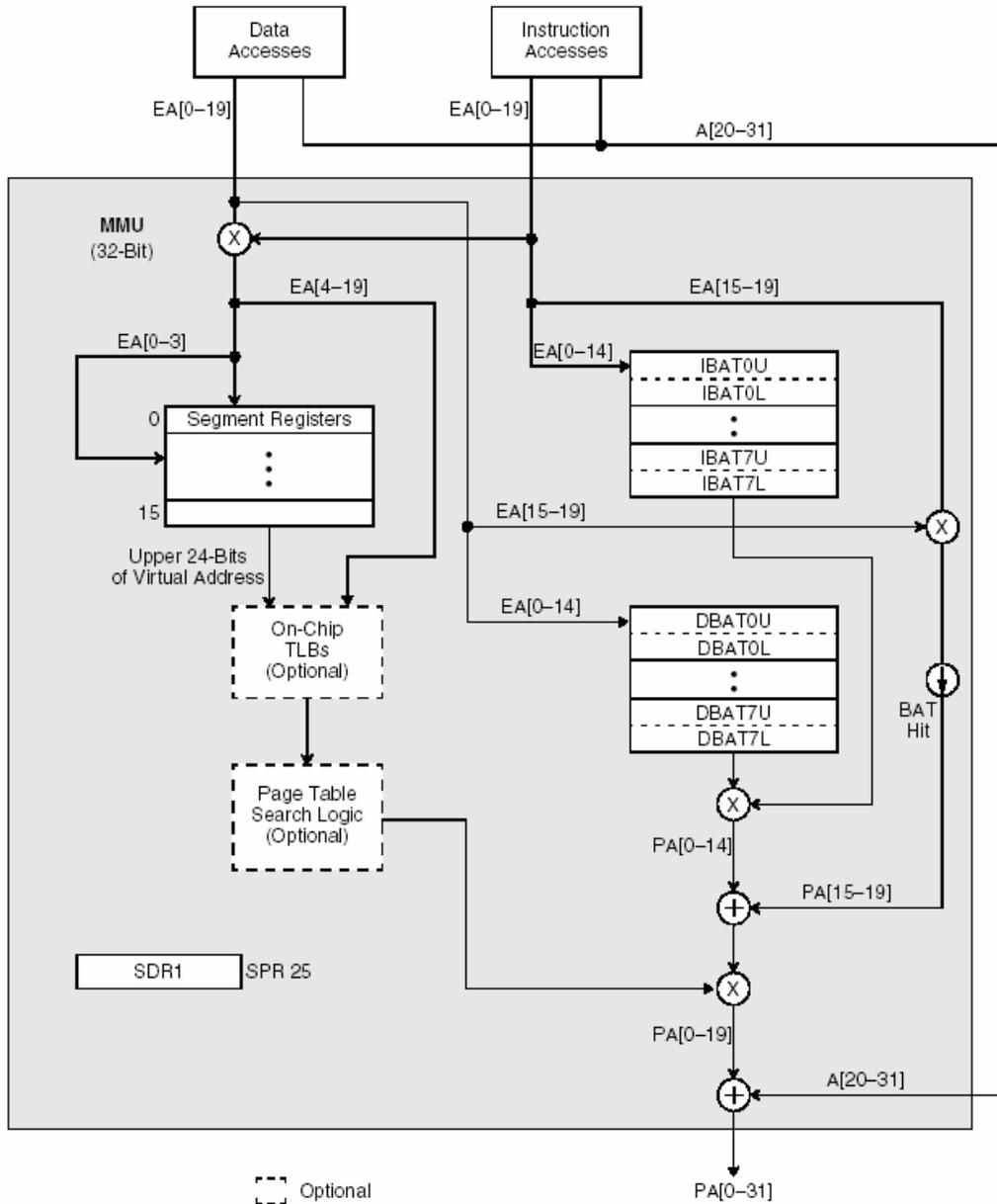


Figure 5. Effective-to-Physical Mapping

Both the 750FX and the MPC7447A offer the following features:

- 128-entry, two-way, set-associative instruction TLB and data TLB
- Eight data BAT and eight instruction BAT pairs
- Translation for 4-Kbyte page size and 256-Mbyte segment size

- Block sizes from 128-Kbyte to 256-Mbyte (up to 4 Gbytes for MPC7447A)

The other significant difference is the fact that the MPC7447A can support 36-bit physical addressing by enabling HID0[XAEN], thus allowing the increased 64-Gbyte memory space. The extended block size of greater than 256-Mbyte is enabled by asserting HID0[XBSEN] and by using the BAT n U[XBL] field in the upper BAT registers to select larger blocks up to 4 Gbytes. The increased area of memory that can be mapped per BAT means that the programmer does not have to use multiple BATs to map multiple sequential 256-Mbyte blocks on the MPC7447A.

An added feature on the MPC7447A is software support for page-table searching to offer a custom page-table entry and searching operation.

2.5 System Interface

Both the 750FX and MPC7447A support the 60x bus protocol and the MPC7447A also supports the MPX bus protocol which is a more efficient protocol based on the 60x implementation. [Table 3](#) highlights the differences between the 750FX and MPC7447A 60x support.

Table 3. – 60x Bus Features

750FX 60x Features	MPC7447A 60x Features
32-bit addressing with 4 bits odd parity	36-bit addressing with 5 bits odd parity
64-bit data bus with 8 bits odd parity, 32-bit data bus support	64-bit data bus with 8 bits odd parity
Three-state MEI cache coherency protocol	Four-state MESI cache coherency protocol
L1 and L2 snooping support for cache coherency	L1 and L2 snooping support for cache coherency
Address-only broadcast instruction support	Address-only broadcast instruction support
Address pipelining	Address pipelining
Support for up to two outstanding transactions (one instruction/one data, or two data)	Support for up to 16 outstanding transactions
200Mhz maximum bus speed	167Mhz maximum bus speed

In addition, the MPC7447A supports MPX bus mode, offering up to 16 out-of-order transactions, data streaming, and data intervention for MP systems. These features make the system bus operation much more efficient, thus increasing the effective bandwidth available in the system. The advantages of the MPX bus can be found in [Section 3.3, “MPX Mode.”](#)

2.6 Thermal Assist Unit

The thermal assist unit (TAU) used in the 750FX provides a means of monitoring the junction temperature, offering an advantage over case- or cabinet-temperature readings, given that the die temperature would be very different.

It can operate on a one- or two-threshold system, whereby the threshold values are programmed into one or two of the TAU’s four special purpose registers. When the temperature reaches one of these thresholds,

an interrupt is generated, allowing software to take appropriate action to reduce the temperature accordingly.

Instead of the TAU, the MPC7447A incorporates a temperature diode that connects to an external temperature monitor device. These devices are widely available from vendors such as Analog Devices, Maxim, and National Semiconductor. Using the negative temperature coefficient of the diode at a constant current, the monitor device can determine the junction temperature. Figure 6 shows how the monitoring device can be connected directly to the anode and cathode of temperature diode on the MPC7447A. The monitor chip is also connected via the 60x or MPX bus to a bridge chip/system controller, which then communicates with the monitor chip itself using I²C. This second connection allows thresholding values to be defined so that the monitor chip can generate interrupts via the bridge chip in a similar manner to the TAU in the 750FX.

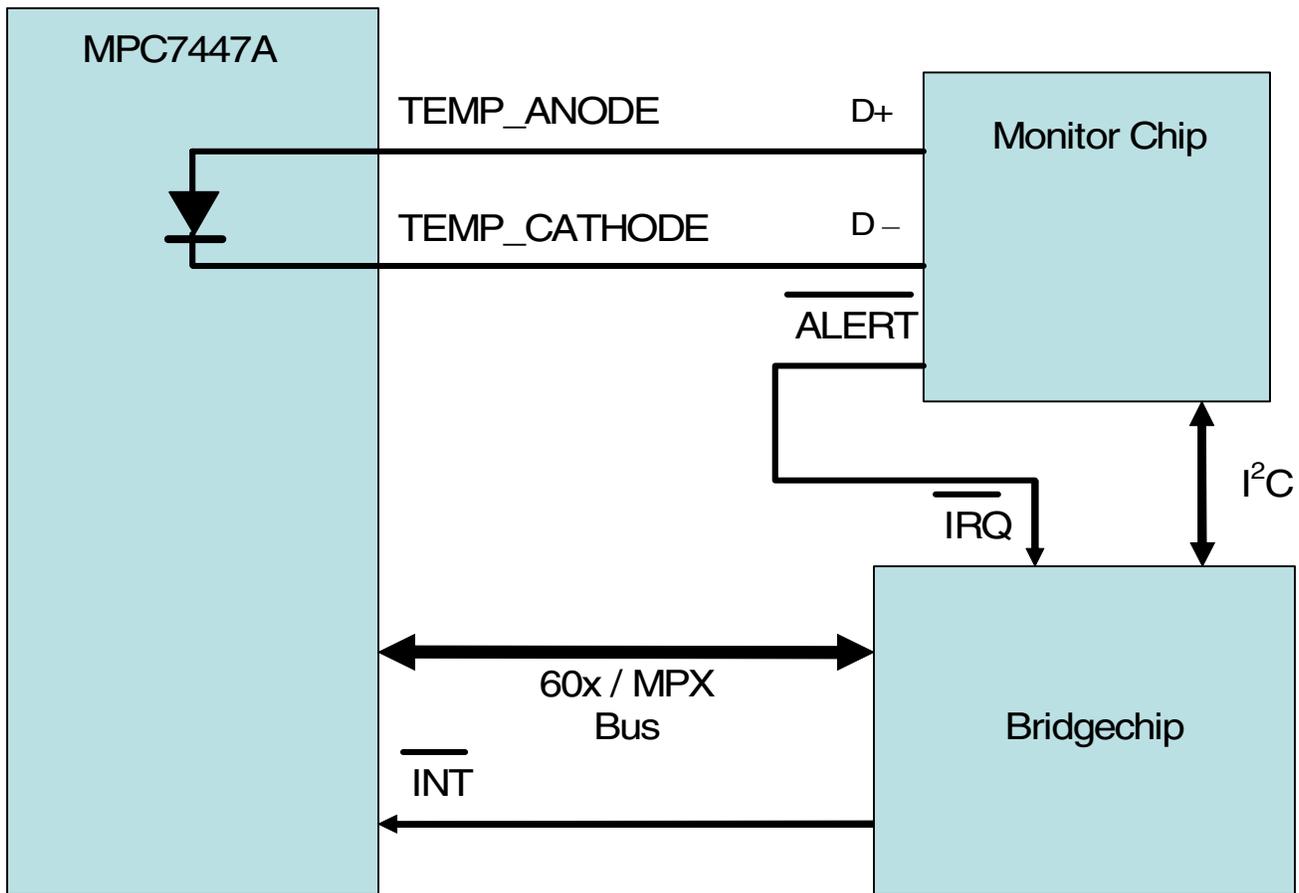


Figure 6. Temperature Monitoring Device Connection

3 7447A Specific Features

This section briefly introduces some major features of MPC7447A devices that are not available on the 750FX and explains how they can offer significant performance improvements.

3.1 AltiVec

Perhaps the most notable difference between the 750FX and MPC7447A is that of AltiVec. It is a SIMD (single-instruction, multiple-data) extension of the PowerPC architecture in terms of both instructions and hardware. It is available on all G4 devices and can offer significant performance improvements on applications that benefit from SIMD programming, such as graphics, certain DSP algorithms, and many computationally-intensive vector applications.

The following are key features of AltiVec:

- 162 new powerful arithmetic and conditional instructions for intra- and inter-element (that is, parallelism) support
- Four operands per instructions: three sources and one destination
- Pipelined execution units to give the following:
 - One cycle latency for simple and permute operations
 - 3-4 cycle latency for compound/complex operations
- No penalty for issuing AltiVec/Integer instruction mix

The new instructions allow vector/SIMD operations on 128-bit-wide vector registers (VRs), through any of the four AltiVec execution units: permute, simple, complex, and float, which have two-, one-, four-, and four-stage pipes respectively. These 128-bit VRs can be used as a single, 128-bit quantity, but also to provide varying levels of parallelism to give a maximum of 16 operations per instruction on 8-bit quantities, or to put into a more comparable format four 32-bit integer-based operations per instruction. These different levels of parallelism, 16x8-bit, 8x16-bit, or 4x32-bit, can be seen in [Figure 7](#).

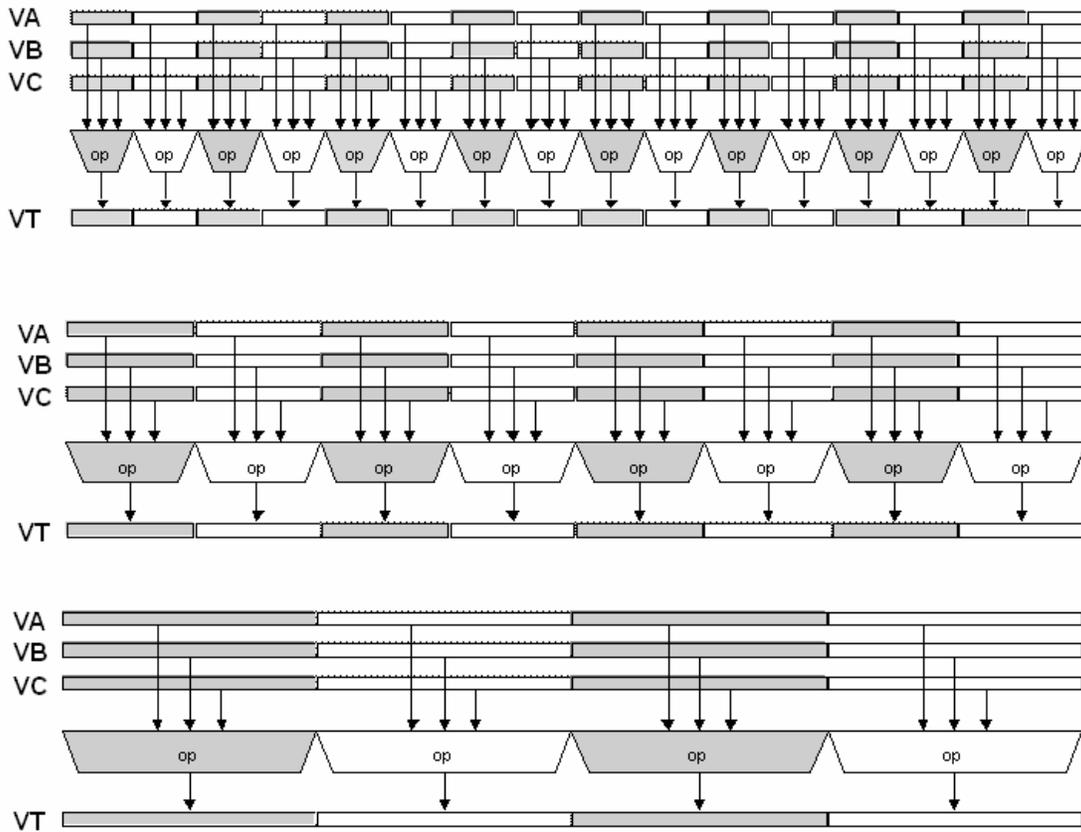


Figure 7. – AltiVec Degrees of Parallelism

3.2 MESI vs. MEI Comparison

Another important difference to note is the difference between the MEI cache coherency features on the 750FX and the enhanced MESI capability of the MPC7447A. These protocols are used as a coherency mechanism in SMP (symmetric multi-processing) configurations to indicate the relationship between 32-byte blocks stored in cache and their corresponding blocks in main memory. In an SMP system, some or all of the main memory is shared. Therefore, it is important to find the most efficient method of maintaining coherency across the caches and memory of the CPUs.

MEI refers to the cache-coherency states available in the 750FX:

- M (modified)—This block is modified with respect to main memory.
- E (exclusive)—This block is valid and only present in this CPU’s cache.
- I (invalid)—This block is invalid with respect to main memory.

It is best to illustrate the MEI protocol operation by way of an example. In a dual processor SMP system using 750FX processors we can refer to the processors and the CPU1 and CPU2 operating on a shared area of memory. If CPU1 loads a cache line from this area of main memory, it is marked as ‘exclusive’ with the assumption that the user has flushed cache on both CPUs. If, however, CPU2 snoops the read request from CPU1 and already has a ‘modified’ in its cache, then it changes its MEI status to ‘invalid’ and pushes the block into main memory, causing CPU1 to wait for and then read the latest version of the data. Then if

CPU2 tries to read the data again, it must read it from main memory; and, to make the situation worse, CPU1 may have since modified the data in its cache. If CPU1 has modified the data, then CPU2 has to wait for CPU1 to write its data back to memory for the CPU2 to access.

The extra bandwidth used and time wasted in waiting for each CPU to write its cache block back to memory for the other CPU to access is a very inefficient use of the bus. To help combat this problem, the MPC7447A supports the MPX bus, which extends the 60x functionality with some efficiency improvements as discussed in the next section. The main method used to improve performance on MPC7447A was to incorporate the MESI protocol which includes the new shared state, 'S'. This block exists in multiple caches and is consistent with main memory. That is, it is read-only.

The addition of this state reduces the wasted time and bandwidth associated with MEI coherency and requires an additional 60x/MPX signal called $\overline{\text{SHD}}$. If we look at the previous example, it is easy to see the benefits of the MESI over MEI. If CPU1 tried to read a block of main memory to its cache, CPU2 would snoop the transaction as before but this time assert the $\overline{\text{SHD}}$ signal to tell CPU1 that it also has a cached copy of this block, too. CPU1 would load the block into its cache with shared status, and CPU2 would change its cache entry to shared from exclusive, allowing both CPUs to access the data quickly from cache, provided they are only reading it.

3.3 MPX Mode

The MPX bus protocol is based on the 60x bus protocol. It also includes several additional features that allow it to provide higher memory bandwidth than the 60x bus and more efficient utilization of the system bus in a multiprocessing environment.

Memory accesses that use the MPX bus protocol are separated into address and data tenures. Each tenure has three phases: arbitration, transfer, and termination. The separation of the address and data tenures allows advanced bus techniques—such as split-bus transactions, enveloped transactions, and pipelining—to be implemented at the system level in multiprocessor systems. The MPX bus protocol also supports address-only transactions.

One benefit that MPX has over 60x is the fact that the MPX does not require an idle cycle between tenures. To illustrate the importance of this difference, consider the following example:

- 100Mhz 60x bus:
Transfer rate = (32 bytes / 5 clock cycles) * 100MHz = 640 Mbyte/s
- 100Mhz MPX bus:
Transfer rate = (32 bytes / 4 clock cycles) * 100MHz = 800 Mbytes/s

Also, given the higher bus speeds of 167MHz available on the 7447A, the transfer rate is scaled accordingly to give significant increase to 1336 Mbytes/s, which compares favorably to the 750GX 1280 Mbytes/s maximum with its 200MHz 60x bus.

The MPX bus mode's support for data intervention and full data streaming for burst reads and writes is realized through the addition of two new signals: $\overline{\text{HIT}}$ and $\overline{\text{DRDY}}$.

The $\overline{\text{HIT}}$ signal is a point-to-point signal output from the processor or local bus slave to the system arbiter. This signal indicates a valid snoop response in the address retry window (the cycle after an $\overline{\text{ACK}}$ that indicates that the MPC7447A will supply intervention data). Intervention occurs when the MPC7447A has

the data that has been requested by another master's bus transaction in its L1 or L2. Instead of asserting $\overline{\text{ARTRY}}$ and flushing the data to memory, the MPC7447A may assert $\overline{\text{HIT}}$ to indicate that it can supply the data directly to the other master. This external intervention functionality is disabled by $\text{MSSCR0}[\text{EIDIS}]$. The $\overline{\text{DRDY}}$ signal is also used by the MPX bus protocol to implement data intervention in the case of a cache hit. The $\overline{\text{SHDI}}$ signal operates in conjunction with the $\overline{\text{SHD0}}$ signal to indicate that a cached item is shared.

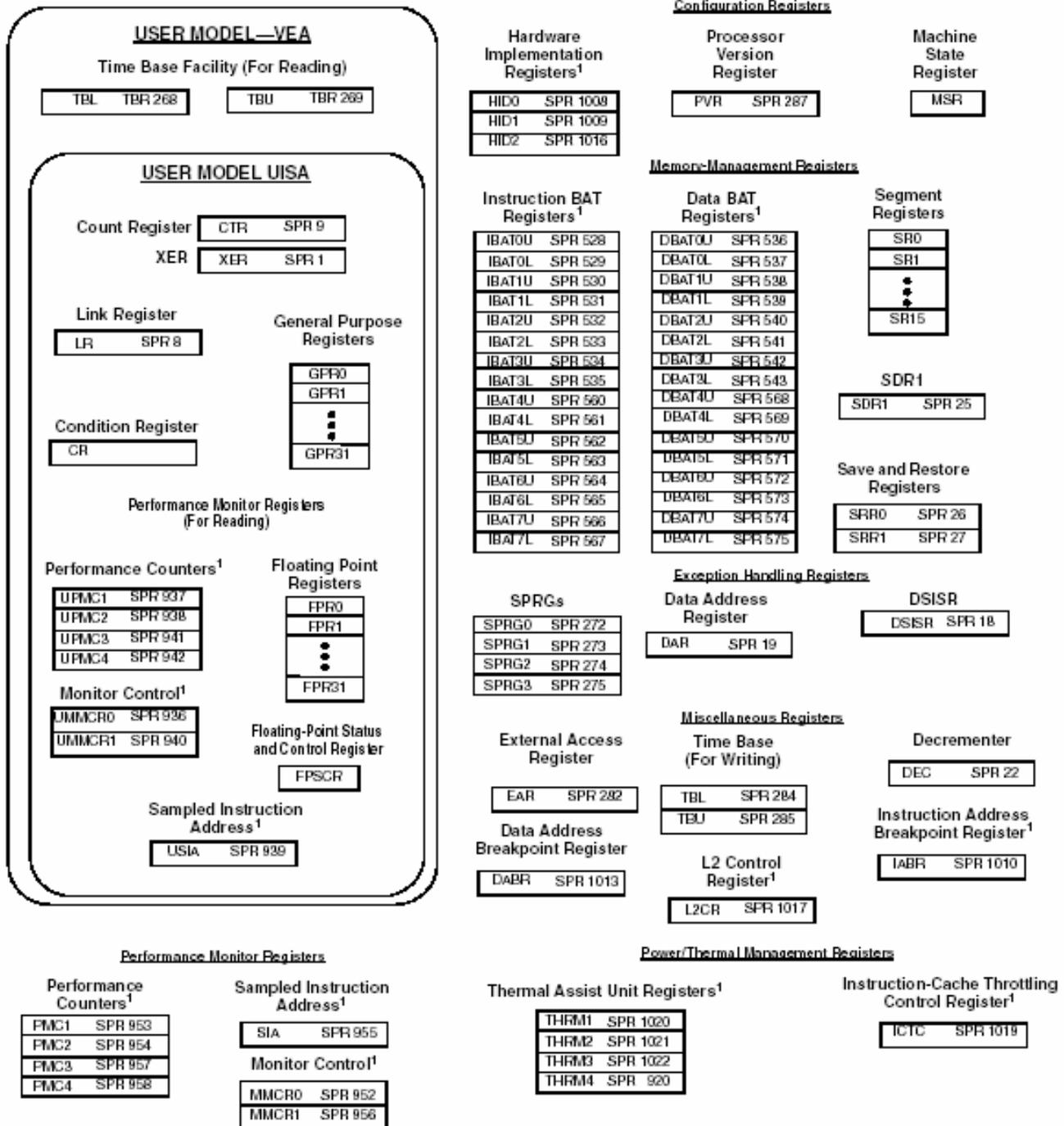
MPX mode offers one final improvement to the 60x with support for out-of-order transactions. As mentioned previously, the MPC7447A supports up to 16 outstanding transactions, as compared to the two supported by the 750FX. This means that the MPC7447A has increased efficiency with its deeper pipeline of transactions. A further improvement that is specific only to MPX mode is the fact that these transactions can be out of order, allowing lower-latency devices to return data as soon as they are ready without waiting for higher-latency devices to return data first just because their transaction was first. This is achieved using the four DTI (Data Transaction Index) signals on the MPX bus. The DTI signals act as a pointer to the outstanding transactions queue and indicate which of the outstanding transactions is to be serviced by subsequent data tenure.

4 Programming Model

Both the IBM 750FX and MPC7447A have to support the PowerPC standard architecture in order to retain compatibility in user mode. Recompilation is not necessary for the IBM 750FX user code to execute properly on the MPC7447A. However, in supervisor mode, there are many differences between device-dependent registers; even though some of the names are the same, the fields are often changed in name and/or bit position. There are also additional registers in different PowerPC implementations to support additional features. This section maps the supervisor-level registers between 750FX and MPC7447A and points out any additional or device-specific features.

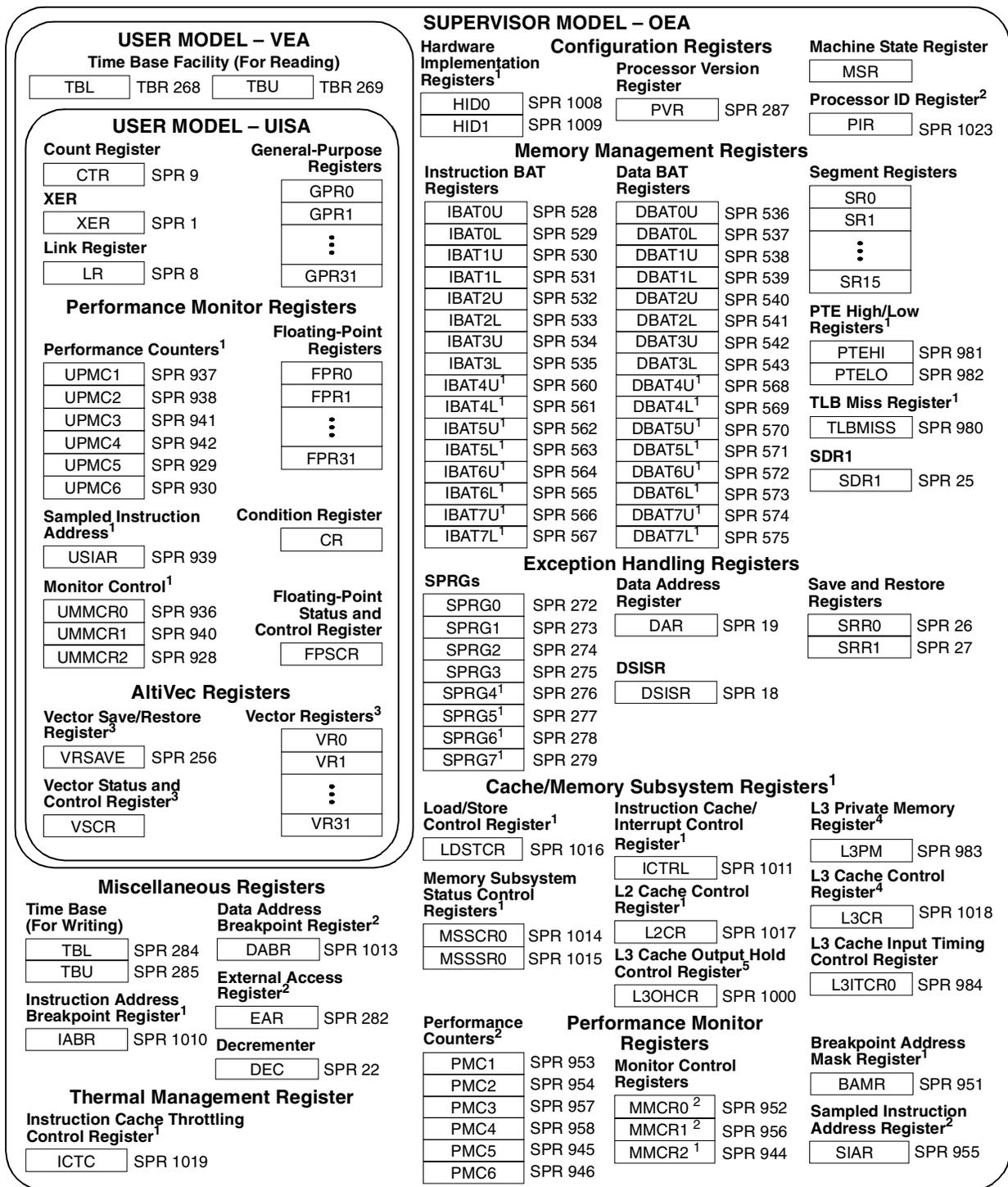
The diagrams in [Figure 8](#) and [Figure 9](#) show the 750FX and MPC7447A programming models respectively.

SUPERVISOR MODEL—OEA



1. These are processor-specific registers. They might not be supported by other PowerPC processors.

Figure 8. – 750FX Registers



¹ MPC7445-, MPC7447-, MPC7455-, and MPC7457-specific register may not be supported on other processors that implement the PowerPC architecture.
² Register defined as optional in the PowerPC architecture.
³ Register defined by the AltiVec technology.
⁴ MPC7455- and MPC7457-specific register.
⁵ MPC7457-specific register.

Figure 9. – MPC7447A Register Set

4.1 Differences in HID0 and HID1

Although the 750FX and the MPC7447A have both the HID0 and the HID1 registers defined in their implementations, however their bit settings can differ. Table 4 summarizes these differences and shows the mapping of fields between devices.

Table 4. 750FX HID0 to MPC7447A Mapping

Function	750FX	MPC7447A
Enable MCP	HID0[EMCP]	HID1[EMCP]
Disable 60x bus address and data parity generation	HID0[DBP]	N/A ¹
Enable 60x bus address parity checking	HID0[EBA]	HID1[EBA]
Enable 60x bus data parity checking	HID0[EBD]	HID1[EBA]
Disable precharge of $\overline{\text{ARTRY}}$	HID0[PAR]	HID1[PAR]
Doze mode enable	HID0[DOZE]	N/A ²
Nap mode enable enable	HID0[NAP]	HID0[NAP]
Sleep mode enable enable	HID0[SLEEP]	HID0[SLEEP]
Dynamic power management enable	HID0[DPM]	HID0[DPM]
Read instruction segment register	HID0[RISEG]	N/A ³
Miss-under-miss enable	HID0[MUM]	N/A ⁴
Not a hard reset	HID0[NHR]	HID0[NHR]
Instruction cache enable	HID0[ICE]	HID0[ICE]
Data cache enable	HID0[DCE]	HID0[DCE]
Instruction cache lock	HID0[ILOCK]	HID0[ILOCK]
Data cache lock	HID0[DLOCK]	HID0[DLOCK]
Instruction cache flush invalidate	HID0[ICFI]	HID0[ICFI]
Data cache flush invalidate	HID0[DCFI]	HID0[DCFI]
Speculative data and instruction cache disable	HID0[SPD]	HID0[SPD]
Enable M bit on bus for instruction fetches (M from WIM states)	HID0[IFEM]	HID0[23] ⁵
Store gathering enable	HID0[SGE]	HID0[SGE]
Data cache flush assist	HID0[DCFA]	HID0[25] ⁶
BTIC enable	HID0[BTIC]	HID0[BTIC]
Address broadcast enable	HID0[ABE]	HID1[ABE] ⁷

Table 4. 750FX HID0 to MPC7447A Mapping (continued)

Branch history table enable	HID0[BHT]	HID0[BHT]
No-op the data cache touch instructions	HID0[NOOPTI]	HID0[NOOPTI]

- ¹ Not available in MPC7447A implementation.
- ² Not required on MPC7447A due to processor-system handshake protocol system explained in Power Management.
- ³ Not implemented. For test only on the 750FX.
- ⁴ Always enabled in MPC7447A implementation. The 750FX supports two outstanding data misses, and the MPC7447A supports five outstanding data misses.
- ⁵ Reserved. Used for IFEM in earlier processors but is also used for extended BAT block size enable.
- ⁶ Reserved. Defined as DCFA on earlier processors.
- ⁷ Must be enabled in multiprocessing systems. HID1[SYNCBE] enables address broadcast for sync and eiio instructions.

4.2 Power Management

Although they are very similar there are a couple of differences between the 750FX and MPC7447A power management functionality. This section only mentions the differences. Features like instruction cache throttling to slow the instruction dispatch rate are the same in both implementations. Both implementations support the four states: full power, doze, nap and sleep.

From [Table 4](#) above, note that there is no HID0[DOZE] bit for the MPC7447A. This is because the MPC7447A enters doze mode when requested by the processor-system protocol. The processor can transition to doze mode from the following:

1. Full power if HID0[NAP] or HID0[SLEEP] and MSR[POW] are asserted and the core is idle.
2. Nap if the system negates \sim QACK to signal a snoop operation is outstanding.

It can transition from doze mode to the following:

1. Full power following one of many possible interrupts: external, \sim SMI interrupt, \sim SRESET, \sim HRESET, machine check or decremter interrupt.
 2. Nap if the system asserts \sim QACK with HID0[NAP] and MSR[POW] set.
- or
3. Sleep if system asserts \sim QACK with HID0[SLEEP] and MSR[POW] set.

Additionally, the MPC7447A has a deep sleep mode which can offer further power savings from sleep mode, turning off the PLL by setting PLL_CFG to 0xF, thus allowing the SYSCLK source to be disabled.

For further explanation on standard power management features between both implementations please refer to the *MPC7450 RISC Microprocessor Family Reference Manual*.

4.2.1 PLL Configuration

HID1 primarily holds PLL configuration and other control bits in both the 750FX and the MPC7447A, although there are a couple of differences as shown below due to the dual PLL in the 750FX vs. the dynamic frequency selection (DFS) in the MPC7447A but not in other current MPC7450 family devices.

For this reason there is not a direct mapping between the two. The concept behind both schemes is to save power through reducing the core clock rate when full rate is not required.

4.2.1.1 Dual PLL Configuration

The 750FX has dual PLL, allowing the frequency to be selected from PLL0 or PLL1, where the transition is controlled through software. A change in clock frequency will take three cycles to complete.

Due to the presence of dual PLL, a change in frequency involves changing a few parameters in sequence. An example of this is changing the source from PLL0 to PLL1 as shown below:

1. Configure PLL1 to produce the desired clock frequency, by setting HID1[PR1] and HID1[PC1] to the appropriate values. Note that it is necessary to wait until PLL1 locks.
2. Set HID1[PS] to select PLL1 as the processor clock source.
3. After three cycles, PLL1 becomes the source and the HID1 status fields are updated.

Table 5 shows the fields in HID1 required to configure and change between the two PLLs.

Table 5. – 750FX HID1/Dual PLL Settings

Function	750FX
PLL external configuration, PLL_CFG[0-4] (Read only)	HID1[PCE]
PLL external range configuration (Read only)	HID1[PRE]
PLL status/selection	HID1[PSTAT1]
Enable external clock, CLKOUT	HID1[ECLK]
Internal clock to output, CLKOUT, selection	HID1[9-11] ¹
PLL0 internal configuration select	HID1[PIO]
PLL select	HID1[PS]
PLL0 configuration	HID1[PC0]
PLL0 range select	HID1[PR0]
PLL1 configuration	HID1[PC1]
PLL1 range select	HID1[PR1]

Note: 1. 000 - Factory use, 001 - PLL0 core clock (freq/2), 010 - Factory use, 011 – PLL1 core clock (freq/2), 100 - Factory use, 101 - Core clock (freq/2)

The PLL range is configured according to the frequency ranges shown in Table 6.

Table 6. – PLL Range Configuration

PLL_RNG[0:1]	PLL Frequency Range
00 (default)	600 MHz–900 MHz
01 (fast)	900 MHz–1.0 GHz
10 (slow)	500 MHz–600 MHz
11 (reserved)	Reserved

4.2.1.2 DFS Configuration

The configuration of DFS is comparatively simple, given that it does not use dual PLL. DFS allows the core clock frequency to be halved.

To illustrate the simplicity of the DFS features:

1. The frequency is switched completely “on the fly.”
2. This change occurs in only one clock cycle.
3. It requires zero idle time or operations before or during the transition.

Considering the following equation:

$$P = (C \times V^2 \times f) + P_{DS}$$

Where: P = core power consumption

C = effective capacitance (approx. as a constant)

V = core voltage, V_{DD}

f = core frequency, f_{CORE}

P_{DS} = deep sleep mode power consumption

Excluding deep sleep mode power consumption, which is a minimum fixed power cost for an inactive core, the dynamic power consumption of the device is halved when DFS is applied. Note that static (leakage) power is not affected by DFS, so the power consumption with DFS enabled is not exactly 50% of the full-power consumption. For more information, see [Section 4.2.1.2, “DFS Configuration.”](#) This provides a significant advantage in supporting dynamic processing requirements in power-sensitive applications.

[Table 7](#) shows the bits corresponding with DFS mode.

Table 7. - MPC7447A HID1/DFS Settings

Feature	MPC7447A
DFS divide-by-two enable	HID1[DFS2]
PLL configuration, PLL_CFG[0-4] (Read only)	HID1[PC0-PC4]

4.3 Cache / Memory Subsystem Configuration

The MPC7447A implements two registers named memory subsystem status and control registers, MSSSR0 and MSSCR0, that do not exist in the 750FX. Some of the functions in these extra registers are held in another 750FX register. [Table 8](#) summarizes this relationship.

Table 8. – 750FX Mapping to MPC7447A MSSSR0, MSSCR0 Registers

Function	750FX	MPC7447A
Address bus parity error	SRR1[AP]	MSSSR0[APE]
Data bus parity error	SRR1[DP]	MSSSR0[DPE]
Bus transfer error acknowledge	SRR1[TEA]	MSSSR0[TEA]

In addition, MSSCR0 stores more configuration data. This configuration relates to features not available in the 750FX, including L3 cache parameters for MPC745x devices. It MSSCR0 also defines the number of outstanding bus transactions, MSSCR0[DTQ], and intervention for MPX mode, MSSCR0[EIDIS].

4.4 Differences in L1 and L2 cache configuration

Due to the differences in programming models, the L1 and L2 cache configuration and status bits are located in different registers in the MPC7447A than in the 750FX.

In the 750FX, the HID2 is used for L1 and L2 cache parity error settings and status, but this register is not present in the MPC7447A. Table 9 shows which register bits in the MPC7447A give the same functionality as the HID2. These functions are spread across SSR1 (which is present in the 750FX, but its bits are reserved), MSSSR0, the instruction cache and interrupt control register (ICTRL, which is not present in the 750FX), and L2CR.

Table 9. – 750FX HID2 to MPC7447A Mapping

Function	750FX	MPC7447A
Isolate guarded requests on the bus. This bit prevents pipelining of guarded requests with any other requests on the bus	HID2[IGRE]	N/A ¹
L2 cache low voltage enable and is only available.	HID2[L2LVE]	N/A ¹
Force instruction-cache bad parity	HID2[FICBP]	N/A ¹
Force instruction-tag bad parity	HID2[FITBP]	N/A ¹
Force data-cache bad parity	HID2[FDCBP]	N/A ¹
Force data-tag bad parity	HID2[FDTBP]	N/A ¹
Force L2-tag bad parity	HID2[FL2TBP]	N/A ¹
L1 instruction-cache/instruction-tag parity error status/mask	HID2[ICPS]	SRR1[1]
L1 data-cache/data-tag parity error status/mask	HID2[DCPS]	SRR1[2]
L2 tag parity error status/mask	HID2[L2PS]	MSSSR0[L2TAG] – Tag error (MSSSR0[L2DAT]) – Data error
Enable L1 instruction-cache/instruction-tag parity checking	HID2[ICPE]	ICTRL[EICE] (ICTRL[EICP]) ²
Enable L1 data-cache/data-tag parity checking	HID2[DCPE]	ICTRL[EDEC]
Enable L2 parity checking	HID2[L2PE]	L2CR[L2PE] ³

¹ Not available in MPC7447A implementation.

² When the EICP bit is set, the parity of any instructions fetched from the L1 instruction cache are checked. Any errors found are reported as instruction cache parity errors in SRR1. If EICE is also set, these instruction cache errors cause a machine check or checkstop. If either EICP or EICE is cleared, instruction cache parity is ignored. Note that when parity checking and error reporting are both enabled, errors are reported even on speculative fetches that are never actually executed. Correct instruction cache parity is always loaded into the L1 instruction cache regardless of whether checking is enabled or not.

³ Enables tag and data parity checking.

Table 10 shows the mapping of the 750FX’s L2CR to the MPC7447A.

Table 10. – 750FX L2CR to MPC7447A Mapping

Function	750FX	MPC7447A
L2 cache enable	L2CR[L2E]	L2CR[L2E]
L2 double bit checkstop enable	L2CR[L2CE]	N/A ¹
L2 data only	L2CR[L2DO]	L2CR[DO]
L2 global invalidate	L2CR[L2I]	L2CR[L2I]
L2 write through	L2CR[L2WT]	N/A ¹
L2 test support	L2CR[L2TS]	N/A ¹
L2 cache way locking	L2CR[L2LOCK0], L2CR[L2LOCK1] ²	L2CR[D0] and L2CR[IO]
Snoop hit in locked line checkstop enable	L2CR[SHEE]	N/A ¹
Snoop hit in locked line error	L2CR[SHEER]	N/A ¹
L2 instruction only	L2CR[IO]	L2CR[IO]
L2 global invalidate progress bit	L2CR[IP]	N/A ¹

¹ Not available in MPC7447A implementation.

² 750FX can lock each of the two ways independently, MPC7447A locks completely or not at all.

4.4.1 MPC7450 Extended Capabilities

The MPC7447A also offers the choice of the first or second replacement algorithm, L2CR[L2REP], and an L2 hardware flush feature, L2CR[L2HWF], which the 750FX does not.

An L2 feature supported by the MPC7447A family but not by the 750FX is L2 prefetching. This can offer an improvement in performance by loading the second block of a cache line after a cache miss on the line, because the second block may be required in the near future even if it is not required right now. The MPC7447A family capitalizes of this concept, known as spatial locality, using up to three hardware prefetch engines.

The L2 prefetching feature can be enabled by setting the L2 prefetch enable bit in the memory configuration subsystem register, MSSCR0[L2PFE], provided that the L2 cache is enabled and not configured as data or instruction only.

4.4.2 L1 and L2 Cache Locking

The MPC7447A contains a load/store control register that configures L1 data cache locking by way. The LDSTCR is not present in the 750FX because L1 way locking is not supported. It can be configured on the MPC7447A using the eight bits in LDSTCR[DCWL], indicating which way(s) to lock. There are equivalent bits in the instruction cache and interrupt control register (ICTRL). The eight bits of ICTRL[ICWL] indicates which way(s) to lock.

The 750FX has the ability to lock L2 cache by way using L2CR[LOCK_n] bits. And L2CR[DO] or L2CR[IO] will setup the L2 as data only or instruction only, respectively. The MPC7447A does not support locking the L2 by way, but the whole cache can be locked by setting both L2CR[DO] and L2CR[IO]. L2CR[DO] and L2CR[IO] individually behave identically as the 750FX.

4.5 Memory Management Registers

Because the 750FX does not have the ability to resolve page-table entries in software, it has no need for PTEHI, PTELO, and TLBMISS registers known as SPR 981, 982, and 980 respectively.

The TLBMISS register is automatically loaded when software searching is enabled (HID0[STEN] = 1) and a TLB miss exception occurs. Its contents are used by the TLB miss exception handlers (the software table search routines) to start the search process.

The PTEHI and PTELO registers are used by the **tlbld** and **tlbli** instructions to create a TLB entry. When software table searching is enabled and a TLB miss exception occurs, the bits of the page table entry (PTE) for this access are located by software and saved in the PTE registers.

A full explanation of software page table searching can be found in the *MPC7450 RISC Microprocessor Family Reference Manual*.

4.6 Performance Monitor

Although it is optional, both implementations support the performance monitor features. Performance monitors give the user software the ability to monitor and count specific events including processor clocks, L1 and L2 cache misses, types of instructions dispatched, and branch prediction statistics, among others. The count of these events can be used to trigger an exception.

The performance monitor has three key objectives:

1. To increase system performance with efficient software, especially in a multiprocessing system—Memory hierarchy behavior can be monitored and studied in order to develop algorithms that schedule tasks (and perhaps partition them) and that structure and distribute data optimally.
2. To characterize processors—Some environments may not be easily characterized by a benchmark or trace.
3. To help system developers bring up and debug their systems.

The MPC7447A contains two additional performance counters (PMC5 and PMC6), a breakpoint address mask register (BAM), and an extra monitor control register (MMCR2). This section looks at any differences in the common registers and the purpose of the extra MPC7447A registers. The MPC7447A offers the extra registers to monitor more events, including AltiVec based events that the 750FX obviously does not have to support. Full listings of PMC events available in each implementation can be found in the *IBM PowerPC 750FX RISC Microprocessor User's Manual* and in the *MPC7450 RISC Microprocessor Family Reference Manual*.

Each implementation provides read-only registers in user mode for PMC and MMCR registers with the prefix U. For example: UPMC1, UMMCR1 and so on.

4.6.1 Monitor Mode Control Registers

The mapping between the MMCR0 and MMCR1 is very similar but not identical. [Table 11](#) and [Table 12](#) show this mapping for the 750FX MMCR0 and MMCR1 respectively.

Table 11. – 750FX MMCR0 to MPC7447A

Function	750FX	MPC7447A
Disable counting unconditionally	MMCR0[DIS]	MMCR0[FC]
Disable counting while in supervisor mode	MMCR0[DP]	MMCR0[FCS]
Disable counting while in user mode	MMCR0[DU]	MMCR0[FCP]
Disable counting while MSR[PM] is set	MMCR0[DMS]	MMCR0[FCM1] ¹
Disable counting while MSR[PM] is zero	MMCR0[DMR]	MMCR0[FCM1] ¹
Enable performance monitor interrupt signaling	MMCR0[ENINT]	MMCR0[PMXE]
Disable counting of PMC _n when a performance monitor interrupt is signalled	MMCR0[DISCOUNT]	MMCR0[FCECE] ²
64 bit time base transition selector	MMCR0[RTCSELECT]	MMCR0[TBSEL]
Enable interrupt when RTCSELECT defined bit transitions off/on	MMCR0[INTONBITTRANS]	MMCR0[TBEE]
Threshold value, 0-63, which can be varied to get to characterize the events occurring above the threshold	MMCR0[THRESHOLD]	MMCR0[THRESHOLD]
Enable interrupt due to do PMC1 overflow	MMCR0[PMC1INTCONTROL]	MMCR0[PMC1CE]
Enable interrupts due to PMC _n overflow	MMCR0[PMCINTCONTROL]	MMCR0[PMC _n CE] ³
Trigger counting of PMC2-4 after PMC1 overflows or after a interrupt is signaled	MMCR0[PMCTRIGGER]	MMCR0[TRIGGER] ⁴
PMC1 event selector, 128 events	MMCR0[PMC1SELECT]	MMCR0[PMC1SEL]
PMC2 event selector, 64 events	MMCR0[PMC2SELECT]	MMCR0[PMC2SEL]

¹ MSR[PM] on the 750FX corresponds to MSR[PMM] on the MPC7447A.

² For all PMCs, not just PMC_n.

³ Enable overflow interrupts on PMC1-4 for 750FX and PMC1-6 for MPC7447A.

⁴ Trigger counting of PMC2-6 for MPC7447A.

Table 12. – 750FX MMCR1 to MPC7447A

Function	750FX	MPC7447A
PMC3 event selector, 32 events	MMCR1[PMC3SELECT]	MMCR1[PMC3SEL]
PMC4 event selector, 32 events	MMCR1[PMC4SELECT]	MMCR1[PMC4SEL]
PMC5 event selector, 32 events	N/A ¹	MMCR1[PMC5SEL]
PMC6 event selector, 64 events	N/A ¹	MMCR1[PMC6SEL]

¹ PMC5 and PMC6 not present in 750FX.

As mentioned previously, the MPC7447A also has an MMCR2 with a one-bit field, MMCR2[THRESHMULT]. This can be used to extend the range of the MMCR0[THRESHOLD] by multiplying by 2 if set at 0, or by 32 if set at 1.

The MPC7447A also has a breakpoint address mask register, BAMR, that is used as a mask for debug purposes to compare to IABR[0:29] when PMC1 is set to monitor event 42. This event monitors for IABR hits specifically by checking that they match BAMR. A match is defined as: $IABR[0:29] \& BAMR[0:29] = instruction_addr[0:29] \& BAMR[0:29]$

5 Hardware Considerations

5.1 Pinout Comparison

Because there is no footprint/pinout compatibility, the easiest way to compare the 750FX and MPC7447A pins is to look at the different pins on the 750FX that do not exist on the MPC7447A and then to look at the pins present on the MPC7447A but not on the 750FX.

5.1.1 750FX Uncommon Pins

Table 13 shows the signal name, pin number, and a description of the signal.

Table 13. – 750FX Additional Signals

Signal Name	Pin Number	Active	I/O	Description
A1Vdd	Y15			PLL0 supply voltage
A2Vdd	Y16			PLL1 supply voltage
\overline{AACK}	A8	Low	Input	
\overline{ABB}	Y6	Low	I/O	Address bus busy
\overline{AGND}	Y14			Ground for PLL
\overline{DBB}	U7	Low	I/O	Data bus busy
\overline{DBDIS}	A10	Low	I	Data bus disable
\overline{DBWO}	A6	Low	I	Data bus write only
\overline{DRTRY}	W3	Low	I	Data retry
PLL_RNG	W15, U14	High	I	Specifies PLL range ¹
\overline{RSRV}	Y4	Low	O	Internal reservation coherency bit
$\overline{TLBISYNC}$	W11	Low	I	TLB invalidate synchronize

¹ As in Table 6, PLL Range Configuration

5.1.2 MPC7447A Uncommon Pins

Table 14 shows the signal name, pin number and a description of the signal.

Table 14. – MPC7447A Additional Signals

Signal Name	Pin Number	Active	I/O	Description
AVdd	A8			PLL supply voltage
$\overline{\text{BMODE0}}$	G9	Low	I	Bus mode select 0
$\overline{\text{BMODE1}}$	F8	Low	I	Bus mode select 1
$\overline{\text{DRDY}}$	R3	Low	O	Data ready output signal to system arbiter
DTI[0:3]	G1, K1, P1, N1	High	I	Data transfer index (for outstanding bus transactions)
EXT_QUAL	A11	High	I	Extension qualifier
GND_SENSE	G12, N13			Internally connected to GND allowing an external device to know core ground level.
$\overline{\text{HIT}}$	B2	Low	O	MPX support for cache to cache transfers and local bus slaves.
OVDD_SENSE	E18, G18			Supply voltage connection for system interface
$\overline{\text{PMON_IN}}$	D9	Low	I	Transitions counted by PMC1, event 7
$\overline{\text{PMON_OUT}}$	A9	Low	O	Asserted when any performance monitor threshold or condition occurs regardless of whether exceptions are enabled or not
$\overline{\text{SHD}}[0:1]$	E4, H5	Low	I/O	Assertion indicates processor contains data from the snooped address. Second SHD signal required for MPX bus mode.
TEMP_ANODE	N18			Anode from internal temperature diode
TEMP_CATHODE	N19			Cathode from internal temperature diode
TEST[0:3]	A12, B6, B10, E10		I	For internal factory test. Should be pulled up to OVdd for normal operation.
TEST[4]	D10		I	For internal factory test. Should be pulled down to GND.
VDD_SENSE	G13, N12			Internally connected to OVdd allowing an external device to know I/O voltage level. (Were OVdd in earlier MPC74xx implementations)

5.2 60x Signal Differences

One of the significant differences between the 750FX and the MPC7447A is the fact that the MPC7447A does not support 3.3V I/O. It only supports 1.8V and 2.5V, as shown in [Table 15](#).

Table 15. Supported I/O Voltages

Voltage Level	750FX	MPC7447A
1.8V	BVSEL=0, L1TSTCLK=1	BVSEL=0
2.5V	BVSEL=1, L1TSTCLK=1	BVSEL=1
3.3V	BVSEL=1, L1TSTCLK=0	N/A

[Table 16](#) shows some of the differences in 60x signals between the 750FX and MPC7447A. The 750FX contains some optional 60x signals that are not implemented in the MPC7447A. All other 60x signals are the same.

Table 16. – 60x Signal Differences

Signal Description	750FX	MPC7447A
60x bus mode select	Default	$\overline{\text{BMODE0}}=\text{OVdd}$ $\overline{\text{BMODE1}}=\text{OVdd}$
Address bus	A[0:31]	A[0:35] ¹
Address parity	AP[0:3]	AP[1:4] ²
Address parity error	$\sim\text{APE}$	N/A
Address bus busy	$\overline{\text{ABB}}$ (input/output)	N/A
Transaction burst	$\overline{\text{TBST}}$ (input/output)	$\overline{\text{TBST}}$ (output)
Cache inhibited	$\overline{\text{CI}}$ (output)	$\overline{\text{CI}}$ (output)
Write through	$\overline{\text{WT}}$ (output)	$\overline{\text{WT}}$ (output)
Data bus busy	$\overline{\text{DBB}}$ (input/output)	N/A
Data bus write only	$\overline{\text{DBWO}}$	N/A
Data bus disable	$\overline{\text{DBDIS}}$	N/A
Data parity error	$\overline{\text{DPE}}$	N/A
Data retry	$\overline{\text{DRTRY}}$	N/A
Reservation	$\overline{\text{RSRV}}$	N/A
TLB invalidate synchronize	$\overline{\text{TLBISYNC}}$	N/A

¹ Use A[4-35] for 32-bit addressing, with A[0-3] pulled down if not in use.

² In 32-bit mode, AP[0] should be pulled up. In 36-bit mode, use AP[0:4] as follows:

AP[0] contains odd parity for A[0:3].

AP[1] contains odd parity for A[4:11].

AP[2] contains odd parity for A[12:19].

AP[3] contains odd parity for A[20:27].

AP[4] contains odd parity for A[28:35].

Revision History

In the MPC7447A, $\overline{\text{BMODE1}}$ is sampled after $\overline{\text{HRESET}}$ is negated to set the processor ID in $\text{MSSCR0}[\text{ID}]$. The value of the processor ID is important in a multiprocessor system, in which one would want to define one processor with the value 0, by negating $\overline{\text{BMODE1}}$, and make that processor responsible for booting and configuring other processors and system logic. Other processors would have $\overline{\text{BMODE1}}$ tied high to differentiate. In this case, the processor 0 could also configure the other processors' processor ID register, PIR, with unique values within the system.

Another important point is the fact the MPC7447A supports up to 16 pipelined transactions configured by $\text{MSSCR}[\text{DTQ}]$. Note that the $\overline{\text{DBWO}}$ signal is not present on 7447A because 60x does not support out-of-order transactions. Therefore, in this mode, the data transaction index pins, $\text{DTI}[0:3]$, should be pulled low.

6 Revision History

Table 17 gives a revision history for this application note.

Table 17. Revision History

Revision	Release Date	Substantive Change(s)
1	6/22/2005	Minor editing
0	5/9/2005	Initial release

THIS PAGE INTENTIONALLY LEFT BLANK

How to Reach Us:

Home Page:

www.freescale.com

email:

support@freescale.com

USA/Europe or Locations Not Listed:

Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
(800) 521-6274
480-768-2130
support@freescale.com

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)
support@freescale.com

Japan:

Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku
Tokyo 153-0064, Japan
0120 191014
+81 2666 8080
support.japan@freescale.com

Asia/Pacific:

Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate,
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:

Freescale Semiconductor
Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
(800) 441-2447
303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. The described product is a PowerPC microprocessor. The PowerPC name is a trademark of IBM Corp. and used under license. All other product or service names are the property of their respective owners.

© Freescale Semiconductor, Inc. 2005.