Freescale Semiconductor
Application Note

Document Number: AN3952
Rev. 0, 10/2009

Qorivva MPC560xP and MPC564xL Compatibility

Transition from MPC5604P to MPC5643L in QFP 144 packages

by: Oliver Bibel
München, Freescale EMEA

The high performance Qorivva MPC56xxx MCU architecture, which includes the MPC560xP family, now adds the MPC564xL family, supporting higher functional safety levels, significantly higher performance, and enhanced peripheral modules. This application note describes common properties of, and the differences between, the MPC560xP and MPC564xL families, plus steps that allow maximize reuse of software and hardware between the two families.

The MPC560xP and the MPC564xL MCUs are both part of the microcontroller family targeted for automotive chassis applications and designed to reach a high level of compatibility. They are based on the same skeleton architecture and share a wide set of identical peripheral modules.

The MPC564xL family of devices is targeted for SIL3 and ASIL-D applications and therefore, when compared to the MPC560xP family, has additional functional safety measures implemented. The MPC564xL implements an application-independent architecture to

Contents

reach its functional safety target. This application independence allows the MPC564xL to reach a high level of compatibility with other MCUs within the chassis family, such as the MPC560xP.

NOTE:

The MPC5604P comparison is done for MPC5604P cut 2.0 and higher only.

The compatibility comparison between the two families is based on two selected devices — the MPC5604P and the MPC5643L — which are both available in a 144-pin QFP package.

For this comparison the MPC5643L is operated in lockstep mode, which allows the application to access the sphere of replication as a single logical processing channel. Decoupled parallel mode (DPM) is not discussed in this document.

1 Block diagrams

Figure 1 shows a simplified version of the MPC5643L architecture on a block diagram level. It shows the dual core architecture and other replicated IP modules as well as the Redundancy Control and Checker Units (RCCU) from a hardware perspective. Please refer to reference document [1] for a more detailed block diagram of the MPC5643L architecture.

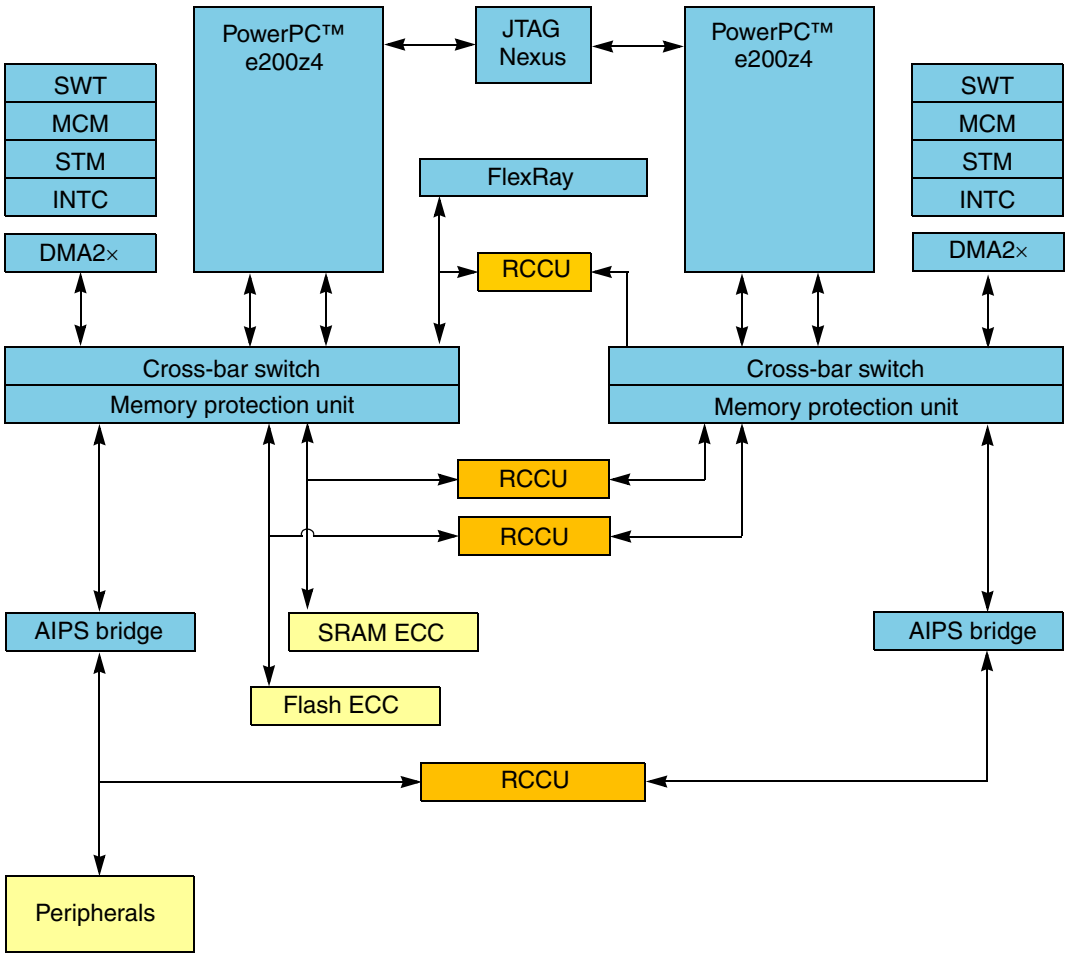

Figure 1. MPC5643L architecture block diagram hardware view

For this application note, comparing the MPC5604P and the MPC5643L in LSM, it is more relevant to compare the software view of both architectures. This is because it is the architecture that will be visible to the application software when the MPC5643L is operated in lockstep mode. This comparison is shown in Figure 2 and Figure 3 below.

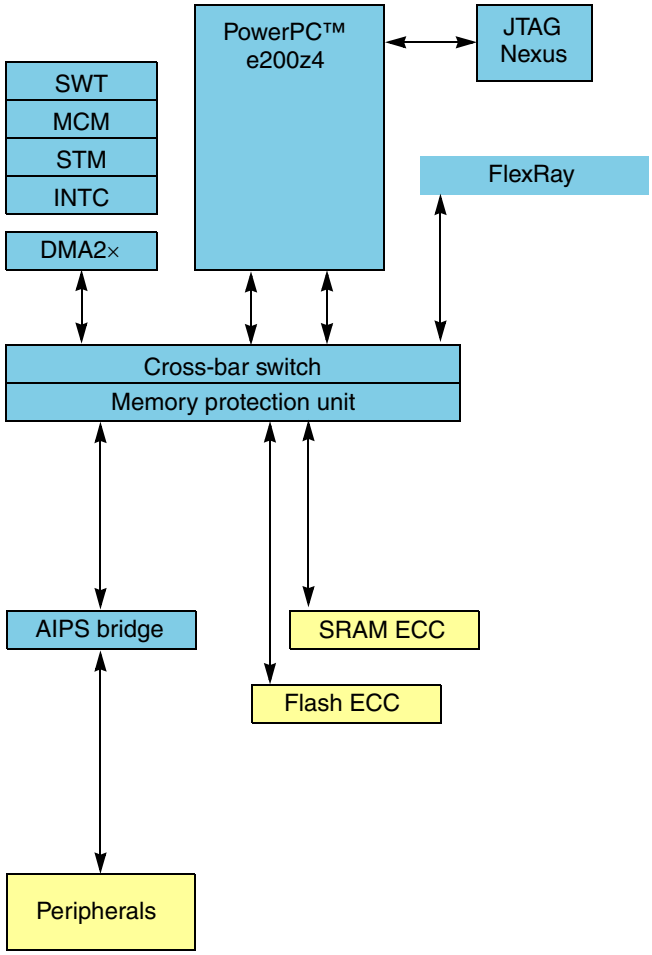

Figure 2. MPC5643L block diagram — software view in LSM

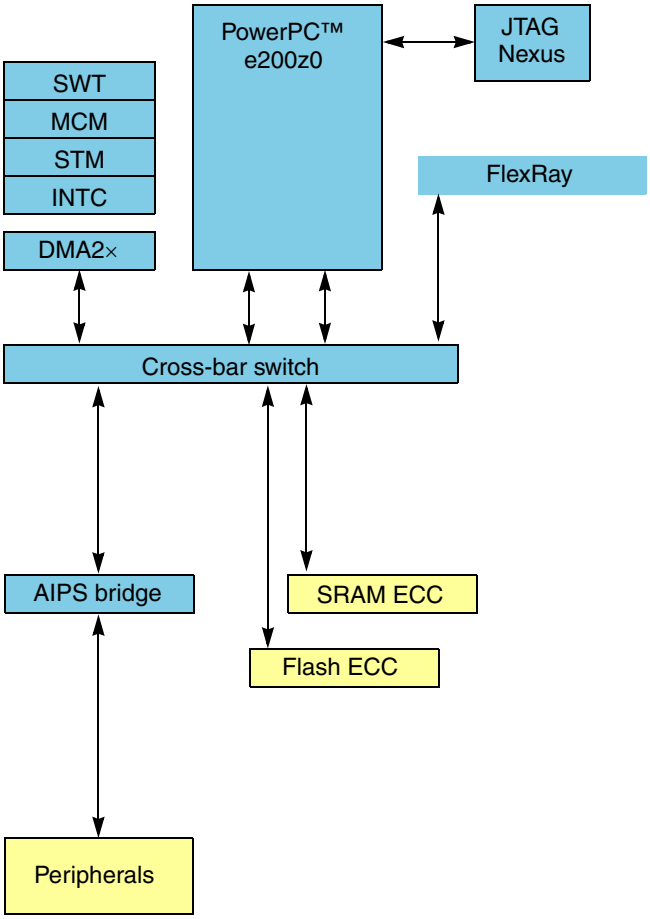

Figure 3. MPC5604P block diagram

The comparison of these two views shows that the MPC5643L sphere of replication in LSM is completely transparent for application software. In that mode, the MPC5643L can be treated and programmed as a single-core device with enhanced fault-detection capabilities.

2 Overview

Table 1 shows on overview of the feature set of the two products. A more detailed description of both the common features and of the differences is given in the next sections.

Table 1. MPC560xP and MPC564xL feature overview

Features	MPC5604P	MPC5643L
Platform architecture	Single core mode	Lockstep mode[1] or decoupled parallel mode
CPU core	e200 z0hn2	e200 z446n3
Core bus interface	Harvard (instruction, data)	
Signal processing unit (SPE)	—	SPE 1.1
Floating point unit (FPU)	—	EFP2

Table 1. MPC560xP and MPC564xL feature overview (continued)

Features	MPC5604P	MPC5643L
Nominal execution frequency	0–64 MHz	0–20 MHz
DMIPS/MHz (no in-lining)	> 1.24	> 2.06
Memory management unit (MMU)	—	16 entries
Instruction set	VLE	Classic PPC & VLE
Instruction cache	—	4 KB, 2-way with EDC
Data cache	—	—
AHB cross bar switch (AXBS)	4 masters and 3 slaves	
Memory protection unit (MPU)	—	16 regions
System RAM	40 KB	128 KB
Flash	Code flash 512 KB + 2×16 KB Data flash 4×16 + 1×16 KB RWW between code and data flash modules	1 MB + 2×16 KB Inter-partition RWW capable
Software watchdog timer (SWT)	Yes	
System timer modules (STM)	Yes	
AHB to IP Sky Blue bus bridge (AIPS)	Yes	
Interrupt controller (INTC)	Yes	
MCM	Yes	
PLL	2 FMPLL	
FlexRay	32 message buffers	64 message buffers; ECC/EDC on internal RAMs
PIT	1 × 4	1 × 4
CTU	Yes	
eTimer	2 modules × 6 channels	3 modules × 6 channels
FlexPWM	2 modules (4 × (2 + 1) ch) each	2 modules (4 × (2 + 1) ch) each[2]
LINFlex	2 (no DMA support)	2 (with DMA support)
DSPI	4 modules, up to 8 CS each	3 modules, up to 8 CS each
CAN/Safety port	1 × 32 message buffers[3]	2 × 32 message buffers
ADC	2 × 10 bit 2 × 11 external channels 2 × 1 internal channel 4 shared external channels	2 × 12 bit 2 × 9 external channels 2 × 3 internal channels 4 shared external channels
Sine wave generator (SWG)	—	Yes
Direct memory access (DMA)	16 channels	
DMA channel mux (DMACHMUX)	32 channels	
Clock output	1	

Table 1. MPC560xP and MPC564xL feature overview (continued)

Features	MPC5604P	MPC5643L
System integration unit light (SIUL)	Yes	
Wakeup unit (WAKEUP)	Yes	
System configuration and status module (SSCM)	Yes	
Cyclic redundancy checker (CRC)	1 module (2 contexts)	1 module (3 contexts)
Serial boot assist module (BAM)	Yes	
Redundancy control and checker unit (RCCU)	—	7
Fault collection (and control) unit	FCU	FCCU
Clock monitoring unit (CMU)	1	3
Power management unit (PMU)	external ballast	external ballast or internal ballast mode selectable
Low voltage detect 1.2 V (LVD12)	1	
High voltage detect 1.2 V (HVD12)	—	1
Low voltage detect 2.7 V (LVD27)	3	
Internal RC oscillator (IRCOSC)	16 MHz	
Main oscillator (XOSC)	4–40 MHz	
Junction temperature sensor	1	2
Nexus support	Class 2+	Class 3+

1 In this document it is assumed that the MPC5643L is operated in lockstep mode.

2 The second FlexPWM is not externally available on the LQFP 144 package.

3 Second CAN is implemented as safety port.

3 Platform

3.1 Core complex

Both products implement a 32-bit PowerPC Book E compliant CPU which is binary compatible for standard code. Code for the z0 and the z4 can be compiled by using the same compiler but using different compiler configurations.

The next sections provide an overview of the main differences and common features between the two cores. For more details please see the reference manuals of both cores (reference documents [3] and [4]).

3.1.1 Central processing unit

The differences and the common features listed in Table 2 are typically transparent for application software written in a high level language like “C.” They are either architectural implementations internal to the core only or are made transparent to software by the compiler.

Table 2. CPU

Feature	MPC5604P (z0h)	MPC5643L (z446n3)
32-bit PowerPC Book E compliant CPU	Single issue	Dual issue
General purpose register size	32-bit	64-bit
Implements VLE APU for reduced code footprint	Yes	
Implements Classic Power Architecture instruction set for highest performance	No	Yes
In-order execution and retirement	Yes	
Precise exception handling	Yes	
Branch processing unit with dedicated branch address calculation adder	Yes	
Branch target prefetching	8-entry BTB	
Endian support	Big endian	Big endian or little endian
Fully pipelined design	4-stage pipeline	5-stage pipeline
Bus interface port size for instruction fetching	32-bit	64-bit
Bus interface port size for load/store	32-bit	64-bit

3.1.2 Machine check APU

Compared to the implementation on the z0, the machine check APU implemented on the z4 core allows more exception situations to be recovered.

The source code that manages machine check exceptions for the z4 on the MPC5643L needs to be adjusted from the code used for the z0 on MPC5604P.

Please refer to reference documents [3] and [4] for more details.

3.1.3 Memory management unit (MMU)

The MPC5604P does not implement a memory management unit.

Use of the MMU implemented on the MPC5643L is configured via additional special purpose registers inside the core and via additional instructions available for the z4 core.

See reference document [4] for more details on MMU implementation on the MPC5643L.

3.1.4 Instruction cache

The MPC5604P does not implement an instruction cache.

Use of the instruction cache implemented on the MPC5643L is configured via additional special purpose registers inside the core and via configuration of the MMU.

See reference document [4] for more details on the instruction cache implementation on the MPC5643L.

3.1.5 Signal processing extension (SPE1.1)

The MPC5604P does not implement an SPE unit.

Use of the SPE implemented on the MPC5643L is configured via additional special purpose registers inside the core, and via additional instructions and additional exception vectors (IVOR) available for the z4 core.

See reference document [4] for more details on SPE implementation on the MPC5643L.

3.1.6 Embedded floating-point (EFP2)

The MPC5604P does not implement an EFP unit.

Use of the EFP implemented on the MPC5643L is configured via additional special purpose registers inside the core, and via additional instructions and additional exception vectors (IVOR) available for the z4 core.

See reference document [4] for more details on EFP implementation on the MPC5643L.

3.2 AHB cross bar switch (AXBS)

The XBAR multi-port crossbar switch supports simultaneous connections between master ports and slave ports. The crossbar supports a 32-bit address bus.

The crossbar allows concurrent transactions to occur from any master port to any slave port. If a slave port is simultaneously requested by more than one master port, arbitration logic selects the higher priority master and grants it ownership of the slave port. All other masters requesting that slave port are stalled until the higher priority master completes its transactions.

The AXBS implementation on the MPC5604P and on the MPC5643L is based on the same IP. However, the MPC5604P implements a lite version of the AXBS and the MPC5643L implements a configurable version of this IP. After being configured (for MPC5643L) this IP is transparent for application software.

Table 3 shows the comparison of the IP implementations on both products.

Table 3. AXBS

Feature	MPC5604P	MPC5643L
AXBS IP	Lite version	Configurable version
Memory map entry	—[1]	See RM [1]
Configurable features	—	Arbitration scheme Master priority levels
Support for # parallel transfers	3	
Data bus width	32-bit	64-bit

1 Not configurable; does not appear in the IP module memory map.

See reference document [1] for more details of the AXBS on MPC5643L

3.3 Memory protection unit (MPU)

The AMBA-AHB memory protection unit (MPU) provides hardware access control for all memory references generated in a device. Using pre-programmed region descriptors which define memory spaces and their associated access rights, the MPU concurrently monitors all system bus transactions and evaluates the appropriateness of each transfer. Memory references that have sufficient access control rights are allowed to complete, while references that are not mapped to any region descriptor or have insufficient rights are terminated with a protection error response.

The MPC5604P does not implement a MPU.

The default state of the MPU implemented on the MPC5643L is disabled. In that state the access from all bus masters is allowed, which is the same behavior as on the MPC5604P.

3.4 AHB to IP Sky Blue bridge (AIPS)

The AIPS is the interface between the Advanced High performance Bus (AHB) interface and on-chip IPS peripherals. IPS peripherals are modules that contain readable/writable control and status registers. The AHB master reads and writes these registers through the AIPS. The AIPS generates module enables, the module address, transfer attributes, byte enables, and write data. These elements then function as inputs to the IPS peripherals.

The AIPS implementation on the MPC5604P and on the MPC5643L is based on the same IP. However, the MPC5604P implements a lite version of the AIPS, and the MPC5643L implements a configurable version of this IP. After being configured (for the MPC5643L) this IP is transparent for application software. The AIPS-lite version implemented on the MPC5604P cannot be configured and therefore does not appear in the memory map of the MPC5604P.

The two major additional features provided by the AIPS implementation on the MPC5643L are listed here:

- AIPS is configurable per-module write-buffering support.
- AIPS is configurable per-module and per-master access protections.

3.5 Interrupt Controller (INTC)

The INTC provides priority-based preemptive scheduling of interrupt requests. This scheduling scheme is suitable for statically scheduled real-time systems. The INTC has two handshaking modes with the processor: software vector mode and hardware vector mode.

The INTC implementation on the MPC5604P and on the MPC5643L is based on the same IP.

The interrupt vector table defines the assignment of interrupt sources to interrupt vector numbers. The assignment used on the MPC5643L is compatible with the assignment used for the MPC5604P. Additional interrupt vectors on the MPC5643L are either appended to the end of the interrupt vector table of the MPC5604P or are placed at vector numbers marked as reserved for the MPC5604P. Interrupt vector numbers originated from modules originally implemented on the MPC5604P become reserved on the MPC5643L.

3.5.1 Software vector mode

In software vector mode, software — that is, the interrupt exception handler — must read a register in the INTC to obtain the vector associated with the interrupt request to the processor.

In software vector mode the INTC implementation on the MPC5643L and the MPC5604P is identical.

3.5.2 Hardware vector mode

In hardware vector mode, the hardware signals the interrupt vector from the INTC in conjunction with a processor that can use that vector. This hardware causes the first instruction that is executed in handling the interrupt request to the processor to be specific to that vector.

In hardware vector mode the start addresses for interrupt service routines are automatically calculated based on the interrupt vector number.

For the MPC5604P the address offset between two consecutive interrupt vectors is four bytes.

For the MPC5643L the address offset between two consecutive interrupt vectors is sixteen bytes.

3.6 Memory map

The memory maps for the MPC5604P and for the MPC5643L are compatible. Modules available on both products are located at the same start address.

- Flash start address: 0x0000_0000 (for different sectorization. please refer to Section 5, “NVM — flash and EEPROM emulation”)
- SRAM start address: 0x4000_0000
- AIPS peripheral blocks start addresses:
  - 0xC3F8_0000
  - 0xFFE00000
  - 0xFFF00000
  - 0xFFF80000

For more details please refer to the memory map section inside the reference manuals of both products (reference documents [1] and [2]).

3.7 Semaphore module (SEMA4)

The SEMA4 module provides the hardware support needed in multi-core systems for implementing semaphores and provides a simple mechanism to achieve lock/unlock operations via a single write access. These gating mechanisms are used by the software to serialize (and synchronize) writes to shared data and/or resources. This is done to prevent race conditions and to preserve memory coherency between different processes and processors.

This module is available on the MPC5643L in DP mode only. DP mode is not within the scope of this document.

3.8 Software watchdog timer (SWT)

The software watchdog timer (SWT) is a peripheral module that can prevent system lockup in situations such as software becoming trapped in a loop or a bus transaction failing to terminate. When enabled, the SWT requires periodic execution of a watchdog servicing operation. The servicing operation resets the timer to a specified timeout period. If this servicing action does not occur before the timer expires, the SWT generates an interrupt or hardware reset. The SWT can be configured to generate a reset or interrupt on an initial timeout; a reset is always generated on a second consecutive timeout.

The MPC5643L uses an enhanced version of the SWT implemented on the MPC5604P. The SWT implementation on the MPC5643L offers keyed service mode as an additional mode to trigger the SWT. In keyed service mode, two pseudo-random key values are used to service the watchdog instead of the fixed sequence used for this purpose on the MPC5604P.

The default configuration of the SWT on the MPC5643L uses the fixed sequence and is fully compatible with the MPC5604P.

In both products the IRCOSC clock output is used as the clock source for the SWT counter.

3.9 System timer module (STM)

The STM is a 32-bit timer designed to support system and application software timing functions that are commonly required. The STM includes a 32-bit up counter and four 32-bit compare channels, with a separate interrupt source for each channel. The counter is driven by the system clock divided by an 8-bit prescale value (1 to 256).

The STM implementation on the MPC5604P and on the MPC5643L is based on the same IP. There is no functional difference between the two implementations.

3.10 Miscellaneous control module (MCM)

The MCM provides several miscellaneous control and status functions for the SoC, including:

- Program-visible information about the platform configuration and revision levels
- Information on memory errors reported by error-correcting codes (ECC) of the ECC protection of the flash and the system SRAM
- ECC error injection control for the system SRAM

The MCM implementation on the MPC5604P and on the MPC5643L is based on the same IP.

As the MCM also provides different information about the SoC, the content of the corresponding configuration registers is different for the MPC5604P and the MPC5643L because, for example, the products use different versions of the e200 core.

As the SRAM ECC on the MPC5643L is extended to the local SRAM array address (see Section 4, “System SRAM”), the MCM registers handling the SRAM ECC error reporting and injection are extended to be able to manage this enhancement.

The MCM generates two different ECC-related interrupts for the INTC:

- For single-bit error correction notification

- For uncorrectable ECC error detection notification

On the MPC5604P these interrupt vectors are combined from:

- ECC information from the system SRAM
- Code
- Data flash

On the MPC5643L these interrupt vectors are combined from

- ECC information from the system SRAM
- Flash module

Please refer to the MCM documentation in reference documents [1] and [2] for more details.

3.11 Direct memory access (DMA)

The DMA is a second-generation platform module capable of performing complex data transfers with minimal intervention from a host processor via sixteen programmable channels. It is a highly programmable data transfer engine, which has been optimized to minimize the required intervention from the host processor. It is intended for use in applications where the data size to be transferred is statically known, and is not defined within the data packet itself.

The DMA implementation on the MPC5604P and on the MPC5643L is based on the same IP. There is no functional difference between the two implementations.

3.12 Direct memory access channel multiplexer (DMACHMUX)

The DMA channel mux allows the routing of a configurable amount of DMA sources (slots) to a configurable amount of DMA channels at the DMA engine.

The DMACHMUX implementation on the MPC5604P and on the MPC5643L is based on the same IP. However, on the MPC5643L the source slot assignment of the DMACHMUX to the peripherals supporting DMA requests is slightly different than the assignment used on the MPC5604P.

For more details, please refer to the DMA channel mux source slot assignment tables in reference documents [1] and [2].

4 System SRAM

The two products implement a different amount of system SRAM. The real system SRAM start address is identical for both products.[1]

All access types to the system SRAM available on the MPC5604P are supported in the same way on the MPC5643L.

The AHB bus system data width is twice as wide on the MPC5643L (64-bit) as it is on the MPC5604P (32-bit). The internal system SRAM architecture on both products is adjusted to the bus interface width.

1. Because an MMU is available on the MPC5643L, real addresses can be mapped to alternate locations within the virtual memory space.

In addition to the SRAM access types that are supported by the MPC5604P, the MPC5643L also supports double-word (64-bit) accesses for higher performance.

The ECC boundary for the system SRAM is at 32 bits for both products. On the MPC5604P the 7-bit ECC is built over 32 bits of data. On the MPC5643L the 7-bit ECC is built over a vector composed of 32 bits of data and the local address bits used for the SRAM array. This MPC5643L enhancement is transparent for application software when read and write accesses are being executed.

This enhancement does have an impact on ECC error injection and ECC error reporting managed by the MCM. For more details on ECC handling see Section 3.10, “Miscellaneous control module (MCM).”

5 NVM — flash and EEPROM emulation

The MPC5604P and the MPC5643L both implement nonvolatile memory with automotive qualified flash technology. On both products, E2PROM emulation with true read-while-write operation is supported within the flash subsystem. The register interface and programming model used for the flash subsystem implemented on both products is highly compatible, but not identical.

The two products implement different amounts of flash memory and different sectorization of the flash modules. The designs of the different flash modules implemented on the MPC5604P and the MPC5643L are adjusted to the different maximum system frequency of each product.

5.1 Flash array

Basic flash modify and configuration operations are compatible between the flash subsystem implemented on the MPC5604P and the one implemented on the MPC5643L. The flash system implemented on the MPC5643L supports faster access times compared to the one implemented on the MPC5604P. Therefore, when both products are operated at the same nominal frequency, fewer wait state cycles can be used on the MPC5643L.

The electrical specifications of the two flash modules used on the two products are different in terms of (for example) data retention, maximum modify/erase cycles, and erase time per sector over lifetime. The differences in electrical specification, together with the different sector size, need to be respected for E2PROM emulation drivers, so as not to exceed the maximum values for each flash module.

In general, the differences between the flash modules used on the two products should be made transparent for the application software by a low-level flash driver.

5.1.1 MPC5604P

The flash subsystem on the MPC5604P is built out of two flash modules:

- One flash module optimized for code storage, referred to as code flash
- One module optimized for data storage, referred to as data flash

Both modules are configured and controlled via two separated register slots in the MPC5604P memory map. One register slot is for code flash and the other is for the data flash module.

E2PROM emulation or read-while-write operation is possible for operations split between these two modules. The operation executed on the code flash module (reading or modifying) is independent from the operation executed on the data flash module (reading or modifying). For example, instruction fetching from the code flash module is possible concurrently with a content modification of the data flash module.

The code flash data array start address on the MPC5604P is 0x0000_0000.

The data flash data array start address on the MPC5604P is 0x0080_0000.

For more details on the MPC5604P flash module configuration and control refer to reference document [2].

5.1.2 MPC5643L

The flash subsystem on the MPC5643L is built out of one flash module split into several partitions. Each partition and each sector can be used for code storage or for data storage. The whole flash module is configured and controlled via one register slot in the MPC5643L memory map. This register slot is located at the same address within the memory map as the register slot used for configuration and control of the code flash module on the MPC5604P. The two register interfaces are very similar.

E2PROM emulation or read-while-write operation is possible for operations split between different partitions within the flash module. An operation executed on one partition (reading or modifying) is independent from an operation executed on an other partition (reading or modifying). For example, instruction fetching from one partition is possible concurrently with a content modification of another partition.

The flash data array start address on the MPC5643L is 0x0000_0000.

For more details on the MPC5643L flash module configuration and control refer to reference document [1].

6 Peripherals

In general, peripherals implemented on the MPC5604P and on the MPC5643L which have the same name are based on the same IP, and are therefore compatible to a very high degree. They have identical configuration interfaces and are located at the same real address within the memory map of the SoC. Interrupts are generated in an identical manner. For more details, please see the subsections in this section of this document.

For peripheral clock sources please refer to Section 8.6.1, “Clock source configuration,” and Section 8.6.2, “Clock selection and distribution.”

6.1 System integration unit lite (SIUL)

The SIUL:

- Provides control over the electrical configuration for the SoC pads (analog and digital)
- Provides General Purpose I/O (GPIO) functionality
- Provides external interrupts

- Controls I/O muxing

The SIUL implementation on the MPC5604P and on the MPC5643L is based on the same IP, and the register interface layout for the implementation on both products is identical. However, each product implements a slightly different set of peripherals, a different number of instantiations of peripherals, and a different total amount of available pins usable by the application. These differences have an impact on:

- I/O muxing implemented via the PCRs (Pad Configuration Registers) and the PSMIO (Pad Selection for Multiplexed Inputs) registers
- The number of available GPIOs
- Mapping the GPIOs to the GPIO related registers

As an example, in both products the PCR0 register will be located at the same address offset within the SIUL module and will implement the same functionality, but it will not necessarily control the same functions via the PA (Pad Output Assignment) bit field within the PCR register. The same is true for the PADSEL bit fields within the PSMIO registers.[1]

The GPIO-related registers of the SIUL are implemented in the same manner for both products. GPIO registers referring to GPIO[x] on the MPC5604P also control GPIO[x] on the MPC5643L. However, that does not necessarily mean that GPIO[x] is connected to the same pad and pin number of the package of each product.

6.2 Flexible pulse width module (FlexPWM)

Each FlexPWM module contains four PWM submodules, each of which can be set up to control a single half-bridge power stage. This PWM is capable of controlling most motor types:

- AC induction motors (ACIM)
- Permanent magnet AC motors (PMAC)
- Both brushless DC (BLDC) and brush DC motors (BDC)
- Switched reluctance (SRM) and variable reluctance motors (VRM)
- Stepper motors

The PWM can also be used to generate independent PWM waveforms on its channel outputs.

The FlexPWM implementation on the MPC5604P and on the MPC5643L is based on the same IP. There is no functional difference between the two implementations.

The MPC5604P implements one FlexPWM — the MPC5643L implements two. The second FlexPWM is not externally available on the LQFP 144 package. The first FlexPWM on the MPC5643L is located at the same place in the memory map as the FlexPWM implemented on the MPC5604P, is integrated into the cross triggering scheme in the same manner, and generates the same interrupts.

All FlexPWM instantiations on both products support hardware-initiated DMA requests and DMA transfers.

1. The target is to maintain for both products, as best as possible, the same mapping between the location of these registers within the SIUL and the I/O muxing that they control. For more details please refer to reference documents [1] and [2].

The second FlexPWM on the MPC5643L is integrated into the cross-triggering scheme implemented by the CTU. Please refer to reference document [1] for more details on the integration of the second FlexPWM into the cross-triggering scheme on the MPC5643L.

Both FlexPWM instantiations on the MPC5643L are operated from the same time base (clock source).

6.3 Enhanced timer module (eTimer)

Each eTimer module contains six identical counter/timer channels. Each 16-bit counter/timer channel contains:

- Prescaler
- Counter
- Load register
- Hold register
- Two queued capture registers
- Two compare registers
- Two compare preload registers
- Four control registers

One watchdog timer function is also provided, but only on the first eTimer instantiation on each product.

The eTimer implementation on the MPC5604P and on the MPC5643L is based on the same IP. There is no functional difference between the two implementations.

The MPC5604P implements two eTimer modules with six channels each. The MPC5643L implements three eTimer modules with six channels each. The first two eTimer modules on the MPC5643L are located at the same places in the memory map as the two implemented on the MPC5604P, are integrated into the cross-triggering scheme in the same manner, and generate the same interrupts.

All eTimer instantiations on both products support hardware-initiated DMA requests and DMA transfers.

The second eTimer instantiation on the MPC5643L supports additional triggering capabilities for the two ADCs that are not supported on MPC5604P.

The third eTimer module on the MPC5643L is integrated into the cross-triggering scheme implemented by the CTU. Please refer to reference document [1] for more details on additional cross-triggering features offered by the enhanced cross-triggering integration of the second and third eTimer modules on the MPC5643L.

All three eTimer instantiations on the MPC5643L are operated from the same time base (clock source).

6.4 Sine wave generator (SWG)

The SWG is a mixed analog/digital hardware IP which generates an analog, high quality sinusoidal voltage signal. It can be programmed with the desired oscillation frequency and amplitude voltage.

The MPC5604P does not implement an SWG module.

The analog signal output of the SWG on the MPC5643L is muxed to a GPIO pin. Please see reference document [1] for more details on the MPC5643L pin muxing.

6.5 Analog to digital converter (ADC)

To put it simply, each ADC converter is built of an analog ADC block that performs the analog-to-digital conversion and a digital interface to configure and control the operation of the analog unit. The digital portion (bus interface unit) of the ADC represents the user interface for application code.

The MPC5604P and the MPC5643L each implement two identical instantiations of each ADC.

The implementations on the MPC5604P and on the MPC5643L share the same IP for the digital interface of the ADC.[1] The register interface is compatible between the MPC5604P and the MPC5643L. Because the MPC5604P implements an ADC (analog portion) with a resolution of ten bits and the MPC5643L implements an ADC with a resolution of twelve bits, the conversion result registers are different in size for the two products. The conversion result registers are located at the same offset for both products and the conversion result is right-justified in both cases.

Each ADC on both products can select one of sixteen analog input channels for conversion. Each product offers four channels which are shared between the two ADCs. These shared channels are connected to package pins to allow analog signal conversion from external sources.

On the MPC5604P one channel of each ADC is connected to a signal source internal to the SoC (Temperature Sensor and internal 1.2 V rail).

On the MPC5643L three channels of each ADC are connected to signal sources internal to the SoC (Temperature Sensor 0, Temperature Sensor 1, BandGap 0, and BandGap 1). The sine wave generator output is also connected to an internal channel of each converter.

Based on the concept described above, a lower number of ADC channels is available at the MPC5643L package pins when compared to external ADC channels available on the MPC5604P. The same ADC channel number is not necessarily mapped to the same package pin on each product. Please see reference documents [1] and [2] for more details on ADC channel mapping to internal and external signal sources.

On both products the ADC clock source can be selected via the corresponding registers in the reset and mode control modules. See reference documents [1] and [2] for more details. The digital interface of the ADC provides an optional divider (ADCLKSEL) that defines the clock frequency of the reference clock used for conversion and sampling by the analog block of the ADC. This bit field is extended for the MPC5643L to allow a finer granularity of the settings for conversion and sampling timing. On both products, if a different clock frequency for the analog block of the ADC is used, then the settings for the conversion and the sampling timing need to be adjusted accordingly.

Due to the higher functional safety level and the functional safety concept applied to the MPC5643L, the analog ADC block used on the MPC5643L has self-checking capabilities implemented which are configured and controlled via the digital interface. These self-checking capabilities are not present on the analog block used for the MPC5604P ADC. Please see reference document [1] for more details on the

1. This statement is true for MPC5604P cut 2.0 and higher. It is not correct for MPC5604P cut 1.x, as this derivative of the MPC5604P is using an older ADC digital interface.

register bits that control the ADC built-in self-checking capabilities and their impact on the conversion sequencing.

6.6 Cross-triggering unit (CTU)

The CTU implements the cross-triggering capability between the motor-control-related peripherals like eTimer, FlexPWM, ADC, and external events. The CTU receives triggering information from motor-control-related peripherals and generates conversion commands to the ADCs synchronous to the received triggering information.

The CTU implementation on the MPC5604P and on the MPC5643L is based on the same IP. The two implementations are compatible.

The MPC5643L implementation of the CTU is slightly enhanced. See reference document [1] for more details.

6.7 WakeUp

The WakeUp unit is used to configure the function of the external NMI.

The WakeUp implementation on the MPC5604P and on the MPC5643L is based on the same IP. There is no functional difference between the two implementations.

The NMI signal on the MPC5604P is directly mapped to the machine check exception of the core.

The NMI signal on the MPC5643L is combined with other NMI sources (which can be generated by the FCCU) before it is mapped to the machine check exception of the core. Please refer also to Section 3.1.2, "Machine check APU," for machine check differences between the two cores used on the MPC5604P and on the MPC5643L.

6.8 Periodic interrupt timer (PIT)

The PIT block implements four timers which can be used for DMA triggering and general purpose interrupts.

The PIT implementation on the MPC5604P and on the MPC5643L is based on the same IP. There is no functional difference between the two implementations.

The assignment of the PIT channels to the DMA channel mux channels is identical for both products.

6.9 Cyclic redundancy checker unit (CRCU)

The CRCU is dedicated to the computation of CRC off-loading the CPU. Several contexts are supported allowing the calculation of the CRC of multiple data streams concurrently. Each context has a separate CRC computation engine. Bit-swap and bit-inversion operations can be applied on the final CRC signature. Each context can be configured with one of two hard-wired polynomials, normally used for most of the standard communication protocols. The data stream supports multiple data width (byte/half-word/word) formats.

The CRCU implementation on the MPC5604P and on the MPC5643L is based on the same IP. The only functional difference between the two implementations is the number of contexts supported by each product. The CRCU implemented on the MPC5604P supports two contexts; the CRCU implemented on the MPC5643L supports three contexts.

6.10 Deserial serial peripheral interface (DSPI)

The DSPI block provides a synchronous serial bus for communication between an MCU and an external peripheral device.

The DSPI implementation on the MPC5604P and on the MPC5643L is based on the same IP.

The MPC5604P implements four instantiations of the DSPI block; the MPC5643L implements three instantiations. The first three instantiations are located at the same location in the memory map for each product.

The DSPI instantiations on the MPC5604P support eight CTAR registers per DSPI module; the DSPI instantiations on the MPC5643L support four CTAR registers per DSPI module.

For supported number of chip select (CS) lines per DSPI module, please see the pinout table and the pin-muxing documentation in reference documents [1] and [2].

The instantiations on each product generate the same type of interrupt requests and all support hardware-initiated DMA requests.

6.11 Flexible controller area network (FlexCAN)

The FlexCAN module is a communication controller implementing the CAN protocol according to the CAN 2.0B protocol specification.

The FlexCAN implementation on the MPC5604P and on the MPC5643L is based on the same IP. Both products implement two instantiations of the FlexCAN IP. On the MPC5604P the two FlexCAN instantiations are called FlexCAN 0 and SafetyPort. On the MPC5643L they are referred to as FlexCAN 0 and FlexCAN 1. The first instantiation (FlexCAN 0) is located at the same address in the memory map of each product. The second instantiation of the FlexCAN IP is located at different addresses in the memory map of each product. However, the register interface is identical for all FlexCAN IP instantiations on both products. The interrupts generated are identical for each instantiation. For the interrupt mapping between the FlexCAN modules and the INTC, please refer to the documentation in reference documents [1] and [2].

Both products allow the different FlexCAN (SafetyPort) instantiations to run with different combinations of clock frequency and clock source. Please see the clocking chapter in reference documents [1] and [2] for more details. Various clock frequency and clock source options are supported on both products that allow FlexCAN IP instantiations to operate with the same clock frequency and clock source selection, but require slightly different configuration settings inside the Clock Generation Module (clock divider settings, clock selector settings) and/or inside the FlexCAN IP (CLKSEL).

6.12 FlexRay

The FlexRay implementation on the MPC5604P and on the MPC5643L is based on the same IP. However, the FlexRay version implemented on the MPC5643L is improved in terms of functional safety features compared to the version implemented on the MPC5604P. This list shows a brief overview of the improvements due to the enhanced functional safety features which have been implemented on the MPC5643L:

- ECC added for local FlexRay SRAM memories

  Disabled per default and therefore backwards-compatible with implementation on MPC5604P.
- ECC error injection and reporting added

  Disabled per default and therefore backwards-compatible with implementation on MPC5604P.
- ECC fault interrupt added

  Disabled per default and therefore backwards-compatible with implementation on MPC5604P.
- Message Buffer Configuration data needs to be written using 16-bit accesses due to ECC availability. Register content has not changed compared to MPC5604P. MPC5604P also already allows 16-bit access to these 16-bit registers.
- The MPC5604P supports a soft reset command — this command is not available on the MPC5643L.

7 SoC-wide functional safety elements

7.1 FCU & FCCU

The MPC5604P implements an FCU module (Fault Collection Unit) whereas the MPC5643L implements an FCCU (Fault Collection and Control Unit). Both modules reside at the same start address in each product. The FCCU is a very enhanced version of the FCU. In contrast to the FCU which can only collect SoC faults to signal them on Error Out pins, the FCCU on the MPC5643L can also generate interrupt requests, as well as request safe mode transitions and internal resets for the SoC.

Please refer to reference documents [1] and [2] for more details on the fault input assignment and to [1] for more details on the fault reaction capabilities of the FCCU on the MPC5643L.

On the MPC5604P, the protocol used for the fault signalling via the Error Out pins can be configured via software only. On the MPC5643L this protocol can also be configured via user option data in the flash memory.

7.2 Self-test control unit (STCU)

The STCU is a completely programmable hardware IP that controls the self-test procedure executed during the system's reset sequence. This IP is one of the fundamental mandatory blocks in the safety devices that guarantees the required safety integrity level (SIL) before the application starts up. It is able to manage by hardware the device's logic built-in self-test (LBIST) targeting physical defects in the digital logic and to manage the SRAM/ROM built-in self-test (MBIST) targeting the physical defects in the memory array. The STCU is configured via configuration data stored in the nonvolatile memory. Self-test parameters

(scheduling activity, LBIST setup, and fault reaction configuration) are loaded from flash memory during the SoC's reset sequence.

The MPC5604P does not implement a self-test control unit.

The STCU implemented on the MPC5643L is configured once during the SoC reset phase via the corresponding configuration data in flash memory. See reference document [1] for more details on flash location and full documentation of the configuration options. For functional safety applications, the self-test results collected by the STCU during the reset sequence shall be checked once against the expected results, by the application software after system startup. For subsequent application operation the application software does not need to access the STCU anymore.

7.2.1 MBIST

The MPC5604P does not implement memory built-in self-tests.

See Section 7.2, "Self-test control unit (STCU)," for more details.

7.2.2 LBIST

The MPC5604P does not implement logic built-in self-tests.

See Section 7.2, "Self-test control unit (STCU)," for more details.

7.3 RCCU

The MPC5604P does not implement a Redundancy Control and Checker Unit.

The RCCUs implemented on the MPC5643L do not have any configuration interface. They check all valid output signals leaving the SoR for equality in every clock cycle in the LockStep mode of operation. Faults (mismatches found when two corresponding output signals of the SoR are compared) are signalled to the reset generation module and the FCCU module on the MPC5643L. RCCU fault injection can be controlled via the FCCU.

8 System support functions

8.1 Voltage Regulator

The voltage regulator implementation for the MPC5643L is derived from the implementation used for the MPC5604P and enhanced by the possibility to also operate the VREG with an internal ballast transistor. Both implementations use a compatible register interface for VREG configuration. The MPC5643L register interface, which controls and configures the enhanced VREG, is extended. It is located at the same start address within the MPC5643L memory map as the one on the MPC5604P.

8.1.1 LVDs/HVD

Please refer to Section 8.6.3, "Reset," for the details of the LVDs and HVD.

8.2 Boot assist module (BAM)

The BAM is a re-locatable block of read-only memory mapped into the SoC memory space. It implements code that provides serial boot support on the SoC. The BAM code is automatically executed if an alternate boot mode (FAB, force alternate boot mode pin) is selected during the device reset sequence.

Please refer to reference document [1] and [2] for more details on boot mode configuration and boot configuration pins.

8.3 Censorship protection

The censorship protection strategy is identical on both products.

8.4 System status and configuration module (SSCM)

The SSCM module includes these application-visible features in LSM mode:

- System configuration and status
- Device mode and security status
- DMA status
- Debug status port enable and selection
- Bus and peripheral abort enable/disable

The SSCM implementation on the MPC5604P and on the MPC5643L is based on the same IP. The register interface is placed at the same start address in each of the two products. The register layout is compatible between the implementation on the MPC5604P and on the MPC5643L. Due to the different flash modules used in the MPC5604P and the MPC5643L, as well as the two modes of operation (LSM and DPM) implemented on the MPC5643L, there are additional registers available on the MPC5643L SSCM implementation. These additional registers are used to configure, control, and reflect the current status of the additional functions.

Please refer to reference documents [1] and [2] for more details on additional registers present on the MPC5643L SSCM implementation.

8.5 Device Configuration

Selected device configurations are required to be already available before the SoC is released from reset. These device configurations are stored at dedicated memory locations in flash memory.

The MPC5643L supports the same user option settings as the MPC5604P and provides additional configuration options, due to the enhanced feature set for:

- LockStep mode/decoupled parallel mode selection
- Built-in self-test configuration
- FCCU error out protocol configuration

8.6 MagicCarpet

From a programming point of view, the MagicCarpet's functionality can be broken down into four groups:

- Mode control
- Clock control
- Reset control
- Power control

The programmable registers for each of these functions are contained in separate MagicCarpet modules: MC_ME, MC_CGM, MC_RGM, and MC_PCU, respectively. When configuring an SoC's operating modes (in other words, defining the SoC's behavior during the desired operating modes), the programmer must treat all four module's registers as a single and consistent entity.

The MagicCarpet implementation on the MPC5604P and on the MPC5643L is based on the same IP. The MagicCarpet implementation used on the MPC5643L is slightly enhanced compared to the version used on the MPC5604P. The register interface start address of the different submodules of the MagicCarpet is identical for both products. The register interface of the different submodules is compatible between the MagicCarpet version implemented on the MPC5604P and the one used on the MPC5643L. For details of the differences see reference documents [1] and [2].

8.6.1 Clock source configuration

8.6.1.1 XOSC

The XOSC register interface is located at the same start address and provides the same functionality on both products.

8.6.1.2 IRCOSC

The IRCOSC register interface is located at the same start address and provides the same functionality on both products.

8.6.1.3 FMPLL

The FMPLL register interfaces are located at the same start addresses and provide the same functionality on both products.

8.6.1.4 Clock monitoring unit (CMU)

The CMU serves two purposes. The main task is to permanently supervise the integrity of the various System-on-Chip's clock sources, for example the crystal oscillator or the FMPLL. If the FMPLL output frequency leaves an upper or lower frequency boundary or if the crystal oscillator fails, it can detect and forward these kind of events to a clock management unit and/or a fault collection unit, and then generate interrupts when enabled. It can also monitor an external crystal oscillator clock. The second task of the CMU is to provide a frequency meter, which allows measurement of the frequency of one clock source versus a reference clock.

The CMU register interface for the CMUs on both products is identical. However, the MPC5604P implements two CMU instantiations and the MPC5643L implements three CMU instantiations. Also, the CMUs implemented on the MPC5604P and the MPC5643L SoC monitor different clock signals.

Please refer to reference documents [1] and [2] for more details on the SoC clock signals monitored by the CMU instantiations on both products.

8.6.2 Clock selection and distribution

Because the MPC5643L SoC is more complex than the MPC5604P SoC, more resources of the Clock Generation Modules are used. For example, the MC_CGM provides the system-clock and auxiliary-clock selector resources, as well as the corresponding configuration registers for the clock selection and distribution on the SoC. On the MPC5643L more clock selector resources are used than on the MPC5604P, and the clock source assignment and clock distribution are similar but not identical.

This list enumerates the main differences in terms of the MC_CGM resource usage and the clock distribution on the MPC5643L:

- Clock source for either FMPLL selectable between XOSC or IRCOSC clock
- Reduced number of clock source inputs for system clock selector
- Different clock source assignment for auxiliary clock selectors
- Support for reduced bus interface clock for selected peripherals (PERI0_Divider)
- Configuration of selected peripheral clocks mapped to other auxiliary clock selector registers

Please refer to reference documents [1] and [2] for more details on the mapping of MC_CGM resources to clock sources on both products.

8.6.3 Reset

The MC_RGM submodule of the MagicCarpet manages the various reset sources on the SoC and generates reset signals for the SoC's different sections. On each product, the MC_RGM manages different classes of reset sources — power-on reset, external reset, destructive resets, and functional resets — which can be originated by different sources outside of the SoC, as well as from IP modules within the SoC.

On the MPC5643L the assignment of the different reset sources to one of the classes mentioned above was adjusted to its enhanced functional safety concept, as compared to the MPC5604P.

The updated assignment of reset sources to the reset classes on the MPC5643L is designed to meet these targets:

- All under-voltage or over-voltage conditions assert a destructive reset.
- Non-voltage related reset sources cause a long or a short functional reset.
- Error-out signalling during functional reset is supported.
- Default configuration for the bidirectional reset pin is such that functional resets do not assert the reset pin.

Please refer to reference documents [1] and [2] for more details on the mapping of reset sources to the MC_RGM.

8.7 JTAG

Both products support a IEEE 1149.1 compliant JTAG interface.

The MPC5604P provides a 4-pin interface with TDI, TDO, TCK, and TMS.

The MPC5643L provides the same 4-pin interface as on the MPC5604P, but due to its enhanced functional safety concept a fifth pin, called JCOMP, is provided. The JCOMP provides a dedicated reset pin for the JTAG interface.

8.8 Nexus

Both products support Nexus debugging. The MPC5604P supports Nexus Class 2+ and the MPC5643L supports Nexus Class 3+. Nexus Class 3+ is a superset of the Class 2+ feature set. Both products support four MDO pins on the Nexus auxiliary port in the 144 QFP package. On both products, all pins of the Nexus auxiliary interface except the MDO[0] pin are muxed with alternate functions.

The MPC5604P supports a maximum clock frequency of 64 MHz at the MCKO pin, which is equal to the maximum core frequency.

The MPC5643L supports a maximum clock frequency of 60 MHz at the MCKO pin, which is equal to one half of the maximum core frequency.

To provide a higher bandwidth on the Nexus auxiliary port interface, the MPC5643L supports twelve MDO signals in the BGA package.

9 Acronyms and abbreviations

Table 4. Terms and acronyms

Term	Description
ADC	Analog-Digital Converter
AHB	Advanced High-performance Bus
AIPS	AHB to IP Sky Blue
AMBA	Advanced Microcontroller Bus Architecture
APU	Auxiliary Processing Unit. Application specific coprocessor visible to the instruction set.
AXBS	AHB Cross Bar Switch
BAM	Boot Assist Module
Big Endian	A computer architecture in which, within a given multi-byte numeric representation, the most significant byte has the lowest address.
BIST	Built-in Self-Test. Use of embedded test structures within a device to apply vectors and/or evaluate responses from embedded logic and memory arrays.
BIU	Bus Interface Unit
Breakpoint	An event that, when detected, forces the machine to branch to a breakpoint exception routine.
Burst	A bus transfer which has more than one beat of data associated with it.
CAN	Controller Area Network (Flexcan)

Table 4. Terms and acronyms (continued)

Term	Description
CMU	Clock Monitoring Unit
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
CTU	Cross Triggering Unit
Data Breakpoint	Processor is halted at an appropriate instruction boundary after a trigger is set at a data valid time. The trigger is set when the data address and/or value matches a preselected address and/or value.
DMA	Direct Memory Access
DMA2x	Direct Memory Access (Version 2x)
DPM	Decoupled Parallel Mode (Non LockStep)
DRM	Data Read Message: external visibility of data reads to internal memory mapped resources.
DSPI	Deserial Serial Peripheral Interface
ECC	Error Correction Code
EDC	Error Detection Code
Epilog	Final code of an exception handler after servicing the exception.
eTimer	Enhanced Timer
FCCU	Fault Collection and Control Unit
Field	A number of bits that represent a single piece of information.
Flag Bit	Sticky bit which records an interrupt event.
FlexCAN	Flexible Controller Area Network
FlexPWM	Flexible Pulse Width Modulation
FM	Frequency Modulation
FMPLL	Frequency Modulation Phase Locked Loop
Global	Scope of a signal, valid across entire chip.
IEEE-ISTO 5001	Standard for a Global Embedded Processor Debug Interface. Also known as Nexus.
INTC	Interrupt Controller
Interrupt	Cessation by a processor of the execution an ISR or task to execute an ISR.
Interrupt Event	Event in peripheral that necessitates processor intervention.
Interrupt Exception Handler	Code containing the epilog and prolog that calls the ISR specific to the interrupt event that caused the interrupt request to the processor.
Interrupt Request	Signal indicating that a flag bit is asserted and an interrupt event needs to be serviced. A peripheral interrupt request is from a flag bit that was set by an interrupt event in a peripheral. A software settable interrupt request is from a flag bit that was set by software. An interrupt request to the processor indicates that a peripheral or software settable interrupt request is asserted.
IP	Intellectual Property. Refers to reusable blocks.
IRCOSC	Integrated RC Oscillator
ISR	Interrupt Service Routine

Table 4. Terms and acronyms (continued)

Term	Description
JTAG	Joint Test Action Group. Group that developed the IEEE1149.1 standard.
LIFO	Last In First Out
LSM	LockStep Mode
MC	MagicCarpet (Mode, Clock, Reset, and Power)
MCM	Miscellaneous Control Module
MCU	Micro Controller Unit
MMU	Memory Management Unit. Translation Lookaside Buffer which controls translation, protection, and storage attributes of the JPC563M60 memory space.
MPU	Memory Protection Unit
Mux	Multiplexer
Nexus	Forum that wrote the IEEE-ISTO 5001 standard. The standard can also be referred to as the Nexus standard.
Packet	A distinct piece of information contained within a message. Messages may contain one or more packets.
PIT	Periodic Interrupt Timer
PMU	Power Management Unit
Preempt	Suspend execution of an ISR or task to execute another ISR or task.
Priority	Determines if an ISR or task can preempt another ISR or task. It is sometimes referred to in literature as the Priority Level.
Prolog	Initial code of an exception handler before servicing the exception.
PWM	Pulse Width Modulation
RTOS	Real Time Operating System
SIU	System Integration Unit
SoC	System-on-Chip. Refers to the methodology of IP reuse. Also used to describe an MCU that is implemented using an SoC approach. Used interchangeably with MCU.
SoR	Sphere of Replication
SPE	Signal Processing Engine
SPE APU	Signal Processing Extension APU. Vector coprocessor for accelerating DSP algorithms.
SSCM	System Status and Configuration Module
STCU	Self-Test Control Unit
STM	System Timer Module
SWG	Sine Wave Generator
SWT	Software Watchdog Timer

Table 4. Terms and acronyms (continued)

Term	Description
Task	Software routine not initiated by an interrupt request. Tasks usually execute under the control of the RTOS.
VLE	Variable Length Encoding
Word	A word consists of four bytes or 32 bits.
XOSC	Oscillator for external quartz

10 Reference documents

1. Freescale document MPC5643LRM, *MPC5643L Microcontroller Reference Manual,* Rev. 3, 07/2009
2. Freescale document MPC560XPRM, *MPC5604P Microcontroller Reference Manual,* Rev. 2, 11/2008
3. Freescale document e200z0CORERM, *e200z0 Power Architecture™ Core Reference Manual,* Rev. 0, 4/2008
4. Freescale document e200z4RM, *e200z446n3 Power Architecture™ Core Reference Manual,* Rev. 0, 03/2009

11 Revision history

Table 5. Revision history table

Revision number	Revision date	Author	Description of changes
0, Draft A	10th October 2008	O. Bibel	Initial release
0	18 September 2009	G. Emerson	Amended initial release
No change[1]	04 April 2012	S. Holcombe	• Front page: Add SafeAssure branding. • Title and first page: Add Qorivva branding. • Back page: Apply new back page format.

[1] No substantive changes were made to the content of this document; therefore the revision number was not incremented.

Appendix A Pin-out differences in LQPF 144 packages

The MPC5604P to MPC5643L devices in 144 LQFP packages have a significant degree of pin compatability. Significant differences include:

- The MPC5643L has a second core. Some ADC and digital I/O pins on the MPC5604P device were re-allocated as power supply pins on the MPC5643L device to satisfy current density restrictions.
- The MPC5643L has a JCOMP pin which the MPC5604P does not have.
- The MPC5643L has no DSPI3 peripheral.
- The MPC5643L has two pins dedicated to FCCU outputs.

Table 7 shows the differences between the MPC5643L and MPC5604P in the 144 LQFP package.

Table 6. Color key for differences table

	function added to MPC5643L or moved from its pin position on MPC5604P
	same
	function missing on MPC5643L or moved from its pin position on MPC5604P
	different

Table 7. Differences between MPC5643L and MPC5604P in 144 LQFP package

Pin	MPC5643L DS rev 2							MPC5604P DS rev 4						
1	NMI							NMI						
2	A[6]	siul_GPIO[6]	dspi1_SCK	siul_EIRQ[6]				dspi1 SCK	siul_GPIO[6]	A[6]	siul_EIRQ[6]			
3	D[1]	siul_GPIO[49]	etimer1_ETC[2]	ctu0_EXT_TGR	flexray_CA_RX			flexray0 CA RX	siul_GPIO[49]	etimer1 ETC[2]	ctu0 EXT TRG	D[1]		
4	F[4]	siul_GPIO[84]	npc_wrapper_MDO[3]					nexus0 MDO[3]	siul_GPIO[84]	F[4]				
5	F[5]	siul_GPIO[85]	npc_wrapper_MDO[2]					nexus0 MDO[2]	siul_GPIO[85]	F[5]				
6	VDD_HV_IO0_0							VDD_HV_IO0						
7	VSS_HV_IO0_0							VSS_HV_IO0						
8	F[6]	siul_GPIO[86]	npc_wrapper_MDO[1]					nexus0 MDO[1]	siul_GPIO[86]	F[6]				
9	MDO0							nexus0 MDO 0						
10	A[7]	siul_GPIO[7]	dspi1_SOUT	siul_EIRQ[7]				dspi1 SOUT	siul_GPIO[7]	A[7]	siul_EIRQ[7]			
11	C[4]	siul_GPIO[36]	dspi0_CS0	flexpwm0_X[1]	sscm_DEBUG[4]	siul_EIRQ[22]		sscm DEBUG[4]	siul_GPIO[36]	dspi0 CS0	flexpwm0 X[1]	C[4]	siul_EIRQ[22]	

Table 7. Differences between MPC5643L and MPC5604P in 144 LQFP package (continued)

Pin	MPC5643L DS rev 2							MPC5604P DS rev 4						
12	A[8]	siul_GPIO [8]	dspi1_SIN	siul_EIRQ [8]				dspi1 SIN	siul_GPIO [8]	A[8]	siul_EIRQ [8]			
13	C[5]	siul_GPIO [37]	dspi0_SCK	sscm_DEBUG[5]	flexpwm0_FAULT[3]	siul_EIRQ [23]		sscm DEBUG[5]	siul_GPIO [37]	dspi0 SCK	flexpwm0 FAULT[3]	C[5]	siul_EIRQ [23]	
14	A[5]	siul_GPIO [5]	dspi1_CS0	etimer1_ETC[5]	dspi0_CS7	siul_EIRQ [5]		dspi1 CS0	siul_GPIO [5]	etimer1 ETC[5]	dspi0 CS7	A[5]	siul_EIRQ [5]	
15	C[7]	siul_GPIO [39]	flexpwm0_A[1]	sscm_DEBUG[7]	dspi0_SIN			sscm DEBUG[7]	siul_GPIO [39]	dspi0 SIN	flexpwm0 A[1]	C[7]		
16	VDD_HV_REG_0							dspi0 CS1	siul_GPIO [35]	etimer1 ETC[4]	lin1 TXD	C[3]	siul_EIRQ [21]	
17	VSS_LV_COR0_0							VSS_LV_COR0						
18	VDD_LV_COR0_0							VDD_LV_COR0						
19	F[7]	siul_GPIO [87]	npc_wrapper_MCKO					nexus0 MCKO	siul_GPIO [87]	F[7]				
20	F[8]	siul_GPIO [88]	npc_wrapper_MSEO[1]					nexus0 MSEO1	siul_GPIO [88]	F[8]				
21	VDD_HV_IO0_1							VDD_HV_IO1						
22	VSS_HV_IO0_1							VSS_HV_IO1						
23	F[9]	siul_GPIO [89]	npc_wrapper_MSEO[0]					nexus0 MSEO0	siul_GPIO [89]	F[9]				
24	F[10]	siul_GPIO [90]	npc_wrapper_EVTO					nexus0 EVTO	siul_GPIO [90]	F[10]				
25	F[11]	siul_GPIO [91]	leo_sor0_EVTI					nexus0 EVTI	siul_GPIO [91]	F[11]				
26	D[9]	siul_GPIO [57]	flexpwm0_X[0]	lin1_TXD				flexpwm0 X[0]	siul_GPIO [57]	lin1 TXD	D[9]			

Table 7. Differences between MPC5643L and MPC5604P in 144 LQFP package (continued)

Pin	MPC5643L DS rev 2							MPC5604P DS rev 4						
27	VDD_HV_OSC0							VDD_HV_OSC						
28	VSS_HV_OSC0							VSS_HV_OSC						
29	OSCOUT							XTAL						
30	OSCIN							EXTAL						
31	RESET							RESET_B						
32	D[8]	siul_GPIO [56]	dspi1_CS 2	etimer1_ETC[4]	dspi0_CS 5	flexpwm0 _FAULT[3]		dspi1 CS2	siul_GPIO [56]	flexpwm0 FAULT[3]	dspi0 CS5	D[8]		
33	D[5]	siul_GPIO [53]	dspi0_CS 3	flexpwm0 _FAULT[2]				dspi0 CS3	siul_GPIO [53]	fcu0 F[0]	dspi3 SOUT	D[5]		
34	D[6]	siul_GPIO [54]	dspi0_CS 2	flexpwm0 _X[3]	flexpwm0 _FAULT[1]			dspi0 CS2	siul_GPIO [54]	dspi3 SCK	flexpwm0 FAULT[1]	D[6]		
35	VSS_LV_PLL0_PLL 1							VSS_LV_PLL						
36	VDD_LV_PLL0_PLL 1							VDD_LV_PLL						
37	D[7]	siul_GPIO [55]	dspi1_CS 3	dspi0_CS 4	SWG ANAOUT			dspi1 CS3	siul_GPIO [55]	fcu0 F[1]	dspi3 SIN	dspi0 CS4	D[7]	
38	fccu0_F[0]							fcu0 F[0]	siul_GPIO [96]	G[0]	siul EIRQ[30]			
39	VDD_LV_COR0_4							adc0 AN[4]	siul_GPIO [65]	E[1]				
40	VSS_LV_COR0_4							adc0 AN[6]	siul_GPIO [67]	E[3]				
41	C[1]	siul_GPIO [33]	adc0 AN[2]					adc0 AN[2]	siul_GPIO [33]	C[1]				
42	E[4]	siul_GPIO [68]	adc0 AN[7]					adc0 AN[7]	siul_GPIO [68]	E[4]				
43	B[7]	siul_GPIO [23]	lin0_RXD	adc0 AN[0]				adc0 AN[0]	siul_GPIO [23]	lin0 RXD	B[7]			

Table 7. Differences between MPC5643L and MPC5604P in 144 LQFP package (continued)

Pin	MPC5643L DS rev 2							MPC5604P DS rev 4						
44	E[5]	siul_GPIO [69]	adc0 AN[8]					adc0 AN[8]	siul_GPIO [69]	E[5]				
45	C[2]	siul_GPIO [34]	adc0 AN[3]					adc0 AN[3]	siul_GPIO [34]	C[2]				
46	E[6]	siul_GPIO [70]	adc0 AN[4]					adc0 AN[9]	siul_GPIO [70]	E[6]				
47	B[8]	siul_GPIO [24]	etimer0_ETC[5]	adc0 AN[1]				adc0 AN[1]	siul_GPIO [24]	etimer0 ETC[5]	B[8]			
48	E[7]	siul_GPIO [71]	adc0 AN[6]					adc0 AN[10]	siul_GPIO [71]	E[7]				
49	E[2]	siul_GPIO [66]	adc0 AN[5]					adc0 AN[5]	siul_GPIO [66]	E[2]				
50	VDD_HV_ADR0							VDD_HV_AD0						
51	VSS_HV_ADR0							VSS_HV_AD0						
52	B[9]	siul_GPIO [25]	adc0-adc1 AN[11]					adc0-adc1 AN[11]	siul_GPIO [25]	B[9]				
53	B[10]	siul_GPIO [26]	adc0-adc1 AN[12]					adc0-adc1 AN[12]	siul_GPIO [26]	B[10]				
54	B[11]	siul_GPIO [27]	adc0-adc1 AN[13]					adc0-adc1 AN[13]	siul_GPIO [27]	B[11]				
55	B[12]	siul_GPIO [28]	adc0-adc1 AN[14]					adc0-adc1 AN[14]	siul_GPIO [28]	B[12]				
56	VDD_HV_ADR1							VDD_HV_AD1						
57	VSS_HV_ADR1							VSS_HV_AD1						
58	VDD_HV_ADV0_ADV1							adc1 AN[4]	siul_GPIO [63]	D[15]				

Table 7. Differences between MPC5643L and MPC5604P in 144 LQFP package (continued)

Pin	MPC5643L DS rev 2							MPC5604P DS rev 4						
59	VSS_HV_ADV0_ADV1							adc1 AN[6]	siul_GPIO [72]	E[8]				
60	B[13]	siul_GPIO [29]	lin1_RXD	adc1 AN[0]				adc1 AN[0]	siul_GPIO [29]	lin1 RXD	B[13]			
61	E[9]	siul_GPIO [73]	adc1 AN[7]					adc1 AN[7]	siul_GPIO [73]	E[9]				
62	B[15]	siul_GPIO [31]	siul_EIRQ [20]	adc1 AN[2]				adc1 AN[2]	siul_GPIO [31]	B[15]	siul_EIRQ [20]			
63	E[10]	siul_GPIO [74]	adc1 AN[8]					adc1 AN[8]	siul_GPIO [74]	E[10]				
64	B[14]	siul_GPIO [30]	etimer0_ETC[4]	siul_EIRQ [19]	adc1 AN[1]			adc1 AN[1]	siul_GPIO [30]	etimer0 ETC[4]	B[14]	siul_EIRQ [19]		
65	E[11]	siul_GPIO [75]	adc1 AN[4]					adc1 AN[9]	siul_GPIO [75]	E[11]				
66	C[0]	siul_GPIO [32]	adc1 AN[3]					adc1 AN[3]	siul_GPIO [32]	C[0]				
67	E[12]	siul_GPIO [76]	adc1 AN[6]					adc1 AN[10]	siul_GPIO [76]	E[12]				
68	E[0]	siul_GPIO [64]	adc1 AN[5]					adc1 AN[5]	siul_GPIO [64]	E[0]				
69	BCTRL							BCTRL						
70	VDD_LV_REGCOR0							VDD_LV_REGCOR						
71	VSS_LV_REGCOR0							VSS_LV_REGCOR						
72	VDD_HV_PMU							VDD_HV_REG						
73	A[0]	siul_GPIO [0]	etimer0_ETC[0]	dspi2_SCK	siul_EIRQ [0]			A[0]	siul_GPIO [0]	etimer0 ETC[0]	dspi2 SCK	fcu0 F[0]	siul EIRQ[0]	

Table 7. Differences between MPC5643L and MPC5604P in 144 LQFP package (continued)

Pin	MPC5643L DS rev 2							MPC5604P DS rev 4						
74	A[1]	siul_GPIO [1]	etimer0_ETC[1]	dspi2_SOUT	siul_EIRQ [1			A[1]	siul_GPIO [1]	etimer0 ETC[1]	dspi2 SOUT	fcu0 F[1]	DEBUG[7]	siul EIRQ[1]
75	G[11]	siul_GPIO [107]	flexray_DBG3	flexpwm0_FAULT[3]				G[11]	siul_GPIO [107]	flexpwm0 FAULT[3]				
76	D[10]	siul_GPIO [58]	flexpwm0_A[0]	etimer0_ETC[0]				D[10]	siul_GPIO [58]	flexpwm0 A[0]	dspi3 CS0			
77	G[10]	siul_GPIO [106]	flexray_DBG2	dspi2_CS3	flexpwm0_FAULT[2]			G[10]	siul_GPIO [106]	flexpwm0 FAULT[2]				
78	D[11]	siul_GPIO [59]	flexpwm0_B[0]	etimer0_ETC[1]				D[11]	siul_GPIO [59]	flexpwm0 B[0]	dspi3 CS1	dspi3 SCK		
79	G[9]	siul_GPIO [105]	flexray_DBG1	dspi1_CS1	flexpwm0_FAULT[1]	siul_EIRQ [29]		G[9]	siul_GPIO [105]	flexpwm0 FAULT[1]				
80	C[11]	siul_GPIO [43]	etimer0_ETC[4]	dspi2_CS2				C[11]	siul_GPIO [43]	etimer0 ETC[4]	dspi2 CS2	dspi3 CS0		
81	G[8]	siul_GPIO [104]	flexray_DBG0	dspi0_CS1	flexpwm0_FAULT[0]	siul_EIRQ [21]		G[8]	siul_GPIO [104]	flexpwm0 FAULT[0]				
82	C[12]	siul_GPIO [44]	etimer0_ETC[5]	dspi2_CS3				C[12]	siul_GPIO [44]	etimer0 ETC[5]	dspi2 CS3	dspi3 CS1		
83	G[7]	siul_GPIO [103]	flexpwm0_B[3]					G[7]	siul_GPIO [103]	flexpwm0 B[3]				
84	A[2]	siul_GPIO [2]	etimer0_ETC[2]	flexpwm0_A[3]	dspi2_SIN	mc_rgm_ABS[0]	siul_EIRQ [2]	A[2]	siul_GPIO [2]	etimer0 ETC[2]	dspi2 SIN	flexpwm0 A[3]	mc_rgm_ABS[0]	siul EIRQ[2]
85	G[5]	siul_GPIO [101]	flexpwm0_X[3]	dspi2_CS3				G[5]	siul_GPIO [101]	flexpwm0 X[3]				
86	B[5]	siul_GPIO [21]	jtagc_TDI					B[5]		jtag0 TDI				
87	TMS							jtag0 TMS						
88	TCK							jtag0 TCK						
89	B[4]	siul_GPIO [20]	jtagc_TDO					B[4]		jtag0 TDO				
90	VSS_HV_IO0_2							VSS_HV_IO2						

Table 7. Differences between MPC5643L and MPC5604P in 144 LQFP package (continued)

Pin	MPC5643L DS rev 2							MPC5604P DS rev 4						
91	VDD_HV_IO0_2							VDD_HV_IO2						
92	A[3]	siul_GPIO[3]	etimer0_ETC[3]	dspi2_CS0	flexpwm0_B[3]	mc_rgm_ABS[2]	siul_EIRQ[3]	A[3]	siul_GPIO[3]	etimer0 ETC[3]	dspi2 CS0	flexpwm0 B[3]	mc_rgm_ABS[2]	siul EIRQ[3]
93	VDD_LV_COR0_1_FLA0							VDD_LV_COR1						
94	VSS_LV_COR0_1_FLA0							VSS_LV_COR1						
95	VDD_HV_REG_1							D[13]	siul_GPIO[61]	flexpwm0 A[1]	dspi3 CS2	dspi3 SOUT		
96	VSS_HV_FLA0							VSS_HV_FL						
97	VDD_HV_FLA0							VDD_HV_FL						
98	G[6]	siul_GPIO[102]	flexpwm0_A[3]					G[6]	siul_GPIO[102]	flexpwm0 A[3]				
99	D[12]	siul_GPIO[60]	flexpwm0_X[1]	lin1_RXD				D[12]	siul_GPIO[60]	flexpwm0 X[1]	lin1 RXD			
100	G[4]	siul_GPIO[100]	flexpwm0_B[2]	etimer0_ETC[5]				G[4]	siul_GPIO[100]	flexpwm0 B[2]				
101	C[13]	siul_GPIO[45]	etimer1_ETC[1]	ctu0_EXT_IN	flexpwm0_EXT_SYNC			C[13]	siul_GPIO[45]	etimer1 ETC[1]	ctu0 EXT IN	flexpwm0 ext. sync		
102	G[2]	siul_GPIO[98]	flexpwm0_X[2]	dspi1_CS1				G[2]	siul_GPIO[98]	flexpwm0 X[2]				
103	C[14]	siul_GPIO[46]	etimer1_ETC[2]	ctu0_EXT_TGR				C[14]	siul_GPIO[46]	etimer1 ETC[2]	ctu0 EXT TGR			
104	G[3]	siul_GPIO[99]	flexpwm0_A[2]	etimer0_ETC[4]				G[3]	siul_GPIO[99]	flexpwm0 A[2]				
105	D[14]	siul_GPIO[62]	flexpwm0_B[1]	etimer0_ETC[3]				D[14]	siul_GPIO[62]	flexpwm0 B[1]	dspi3 CS3	dspi3 SIN		

Table 7. Differences between MPC5643L and MPC5604P in 144 LQFP package (continued)

Pin	MPC5643L DS rev 2							MPC5604P DS rev 4						
106	F[12]	siul_GPIO [92]	etimer1_ETC[3]	siul_EIRQ [30]				F[12]	siul_GPIO [92]	etimer1 ETC[3]				
107	VSS							VPP TEST						
108	A[4]	siul_GPIO [4]	etimer1_ETC[0]	dspi2_CS1	etimer0_ETC[4]	mc_rgm_FAB	siul_EIRQ [4]	A[4]	siul_GPIO [4]	etimer1 ETC[0]	dspi2 CS1	etimer0 ETC[4]	mc_rgm_FAB	siul_EIRQ [4]
109	B[0]	siul_GPIO [16]	can0_TXD	etimer1_ETC[2]	sscm_DEBUG[0]	siul_EIRQ [15]		B[0]	siul_GPIO [16]	can0 TXD	etimer1 ETC[2]	sscm DEBUG[0]	siul_EIRQ [15]	
110	B[1]	siul_GPIO [17]	etimer1_ETC[3]	sscm_DEBUG[1]	can0_RXD	can1_RXD	siul_EIRQ [16]	B[1]	siul_GPIO [17]	can0 RXD	etimer1 ETC[3]	sscm DEBUG[1]	siul_EIRQ [16]	
111	C[10]	siul_GPIO [42]	dspi2_CS2	flexpwm0_A[3]	flexpwm0_FAULT[1]			C[10]	siul_GPIO [42]	dspi2 CS2	flexpwm0 FAULT[1]	flexpwm0 A[3]		
112	F[13]	siul_GPIO [93]	etimer1_ETC[4]	siul_EIRQ [31]				F[13]	siul_GPIO [93]	etimer1 ETC[4]				
113	F[15]	siul_GPIO [95]	lin1_RXD					F[15]	siul_GPIO [95]	lin1 RXD				
114	B[2]	siul_GPIO [18]	lin0_TXD	sscm_DEBUG[2]	siul_EIRQ [17]			B[2]	siul_GPIO [18]	lin0 TXD	sscm DEBUG[2]	siul_EIRQ [17]		
115	F[14]	siul_GPIO [94]	lin1_TXD					F[14]	siul_GPIO [94]	lin1 TXD				
116	B[3]	siul_GPIO [19]	sscm_DEBUG[3]	lin0_RXD				B[3]	siul_GPIO [19]	lin0 RXD	sscm DEBUG[3]			
117	E[13]	siul_GPIO [77]	etimer0_ETC[5]	dspi2_CS3	siul_EIRQ [25]			E[13]	siul_GPIO [77]	dspi3 SCK	siul_EIRQ [25]			
118	A[10]	siul_GPIO [10]	dspi2_CS0	flexpwm0_B[0]	flexpwm0_X[2]	siul_EIRQ [9]		A[10]	siul_GPIO [10]	dspi2 CS0	flexpwm0 B[0]	flexpwm0 X[2]	siul_EIRQ [9]	
119	E[14]	siul_GPIO [78]	etimer1_ETC[5]	siul_EIRQ [26]				E[14]	siul_GPIO [78]	dspi3 SOUT	siul_EIRQ [26]			
120	A[11]	siul_GPIO [11]	dspi2_SCK	flexpwm0_A[0]	flexpwm0_A[2]	siul_EIRQ [10]		A[11]	siul_GPIO [11]	dspi2 SCK	flexpwm0 A[0]	flexpwm0 A[2]	siul_EIRQ [10]	
121	E[15]	siul_GPIO [79]	dspi0_CS1	siul_EIRQ [27]				E[15]	siul_GPIO [79]	dspi3 SIN	siul_EIRQ [27]			

Table 7. Differences between MPC5643L and MPC5604P in 144 LQFP package (continued)

Pin	MPC5643L DS rev 2							MPC5604P DS rev 4						
122	A[12]	siul_GPIO [12]	dspi2_SOUT	flexpwm0_A[2]	flexpwm0_B[2]	siul_EIRQ [11]		A[12]	siul_GPIO [12]	dspi2 SOUT	flexpwm0 A[2]	flexpwm0 B[2]	siul_EIRQ [11]	
123	JCOMP							C[9]	siul_GPIO [41]	dspi2 CS3	flexpwm0 FAULT[2]	flexpwm0 X[3]		
124	C[15]	siul_GPIO [47]	flexray_CA_TR_EN	etimer1_ETC[0]	flexpwm0_A[1]	ctu0_EXT_IN	flexpwm0_EXT_SYNC	C[15]	siul_GPIO [47]	flexray0 CA TR EN	etimer1 ETC[0]	flexpwm0 A[1]	ctu0 EXT IN	flexpwm0 ext. sync
125	D[0]	siul_GPIO [48]	flexray_CA_TX	etimer1_ETC[1]	flexpwm0_B[1]			D[0]	siul_GPIO [48]	flexray0 CA TX	etimer1 ETC[1]	flexpwm0 B[1]		
126	VDD_HV_IO0_3							VDD_HV_IO3						
127	VSS_HV_IO0_3							VSS_HV_IO3						
128	D[3]	siul_GPIO [51]	flexray_CB_TX	etimer1_ETC[4]	flexpwm0_A[3]			D[3]	siul_GPIO [51]	flexray0 CB TX	etimer1 ETC[4]	flexpwm0 A[3]		
129	D[4]	siul_GPIO [52]	flexray_CB_TR_EN	etimer1_ETC[5]	flexpwm0_B[3]			D[4]	siul_GPIO [52]	flexray0 CB TR EN	etimer1 ETC[5]	flexpwm0 B[3]		
130	VDD_HV_REG_2							C[8]	siul_GPIO [40]	dspi1 CS1	flexpwm0 FAULT[2]	dspi0 CS6		
131	VDD_LV_COR0_2							VDD_LV_COR2						
132	VSS_LV_COR0_2							VSS_LV_COR2						
133	F[0]	siul_GPIO [80]	flexpwm0_A[1]	etimer0_ETC[2]	siul_EIRQ [28]			F[0]	siul_GPIO [80]	flexray0 DBG0	dspi3 CS3	siul_EIRQ [28]		
134	A[9]	siul_GPIO [9]	dspi2_CS1	flexpwm0_B[3]	flexpwm0_FAULT[0]			A[9]	siul_GPIO [9]	dspi2 CS1	flexpwm0 FAULT[0]	flexpwm0 B[3]		
135	VDD_LV_COR0_5							F[1]	siul_GPIO [81]	flexray0 DBG1	dspi3 CS2	siul EIRQ[29]		
136	A[13]	siul_GPIO [13]	flexpwm0_B[2]	dspi2_SIN	flexpwm0_FAULT[0]	siul_EIRQ [12]		A[13]	siul_GPIO [13]	dspi2 SIN	flexpwm0 B[2]	flexpwm0 FAULT[0]	siul_EIRQ [12]	
137	VSS_LV_COR0_5							F[2]	siul_GPIO [82]	flexray0 DBG2	dspi3 CS1			

Table 7. Differences between MPC5643L and MPC5604P in 144 LQFP package (continued)

Pin	MPC5643L DS rev 2							MPC5604P DS rev 4						
138	B[6]	siul_GPIO [22]	mc_cgl_clk_out	dspi2_CS 2	siul_EIRQ [18]			B[6]	siul_GPIO [22]	CLKOUT	dspi2 CS2	siul_EIRQ [18]		
139	F[3]	siul_GPIO [83]	dspi0_CS 6					F[3]	siul_GPIO [83]	flexray0 DBG3	dspi3 CS0			
140	D[2]	siul_GPIO [50]	etimer1_E TC[3]	flexpwm0 _X[3]	flexray_C B_RX			D[2]	siul_GPIO [50]	flexray0 CB RX	etimer1 ETC[3]	flexpwm0 X[3]		
141	fccu0_F[1]							G[1]	siul_GPIO [97]	fcu0 F[1]	siul EIRQ[31]			
142	C[6]	siul_GPIO [38]	dspi0_SO UT	flexpwm0 _B[1]	sscm_DE BUG[6]	siul_EIRQ [24]		C[6]	siul_GPIO [38]	dspi0 SOUT	flexpwm0 B[1]	sscm DEBUG[6]	siul_EIRQ [24]	
143	A[14]	siul_GPIO [14]	can1_TX D	etimer1_E TC[4]	siul_EIRQ [13]			A[14]	siul_GPIO [14]	safetyport 0 TXD	etimer1 ETC[4]	siul_EIRQ [13]		
144	A[15]	siul_GPIO [15]	etimer1_E TC[5]	can1_RX D	can0_RX D	siul_EIRQ [14]		A[15]	siul_GPIO [15]	safetyport 0 RXD	etimer1 ETC[5]	siul_EIRQ [14]		

THIS PAGE IS INTENTIONALLY BLANK

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Document Number: AN3952
Rev. 0
10/2009