Document order number: MCORE/D

Architectual Brief

M•CORE™ microRISC Engine

M•CORE technology provides a high level of performance for embedded control. These innovative 32-bit microRISC cores are designed for high-performance, cost-sensitive embedded control applications, and optimized for minimal power consumption, making them an outstanding choice for many battery-operated, portable, and mobile products.

Features

- 32-bit load/store RISC architecture
- Fixed 16-bit instruction length
- 16 entry, 32-bit general-purpose register file
- Efficient four-stage execution pipeline, hidden from application software
- Single-cycle execution for most instructions, two-cycle branches and memory accesses
- Support for byte/halfword/word memory access
- Fast interrupt support, with 16-entry user-controlled alternate register file
- Vectored and autovectored interrupt support
- On-chip emulation support
- Full static design for minimal power consumption
- Supported by a comprehensive suite of third-party development tools

Overview

The M•CORE architecture is one of the most compact full 32-bit implementations available. The pipelined RISC execution unit uses 16-bit instructions to achieve maximum speed and code efficiency, while conserving on-chip memory resources. The instruction set is designed to support high-level language implementation. A non-intrusive, JTAG-compliant resident debugging system supports product development and in-situ testing. The M•CORE technology library also encompasses a full complement of on-chip peripheral modules designed specifically for embedded control applications.

Total power consumption is determined by all the system components, rather than the processor core alone. In particular, memory power consumption (both on-chip and external) is the dominant factor in total power consumption of the core plus memory subsystem. With this in mind, the M•CORE instruction set architecture trades absolute performance capability for reduced total energy consumption. This is accomplished while maintaining an acceptably high level of performance at a given clock frequency.

The streamlined execution engine uses many of the same performance enhancements and implementation techniques incorporated in desktop RISC processors. A strictly-defined load/store architecture minimizes control complexity. Use of a fixed, 16-bit instruction encoding

The M•CORE name and logotype, and the OnCE name are trademarks of Freescale Semiconductor, Inc.

Revision 2

lowers the memory bandwidth needed to sustain a high rate of instruction execution, and careful selection of the instruction set allows code density and the overall memory footprint of the M•CORE architecture to surpass those of CISC architectures.

These features reduce system energy consumption significantly, and the fully-static M•CORE design uses other techniques to reduce it even more. Dynamic clock management automatically powers down internal functions that are not in use, and the core incorporates three power conservation operating modes, which can be invoked via dedicated instructions.

M•CORE Processor

Figure 1 is a block diagram of the M•CORE processor.

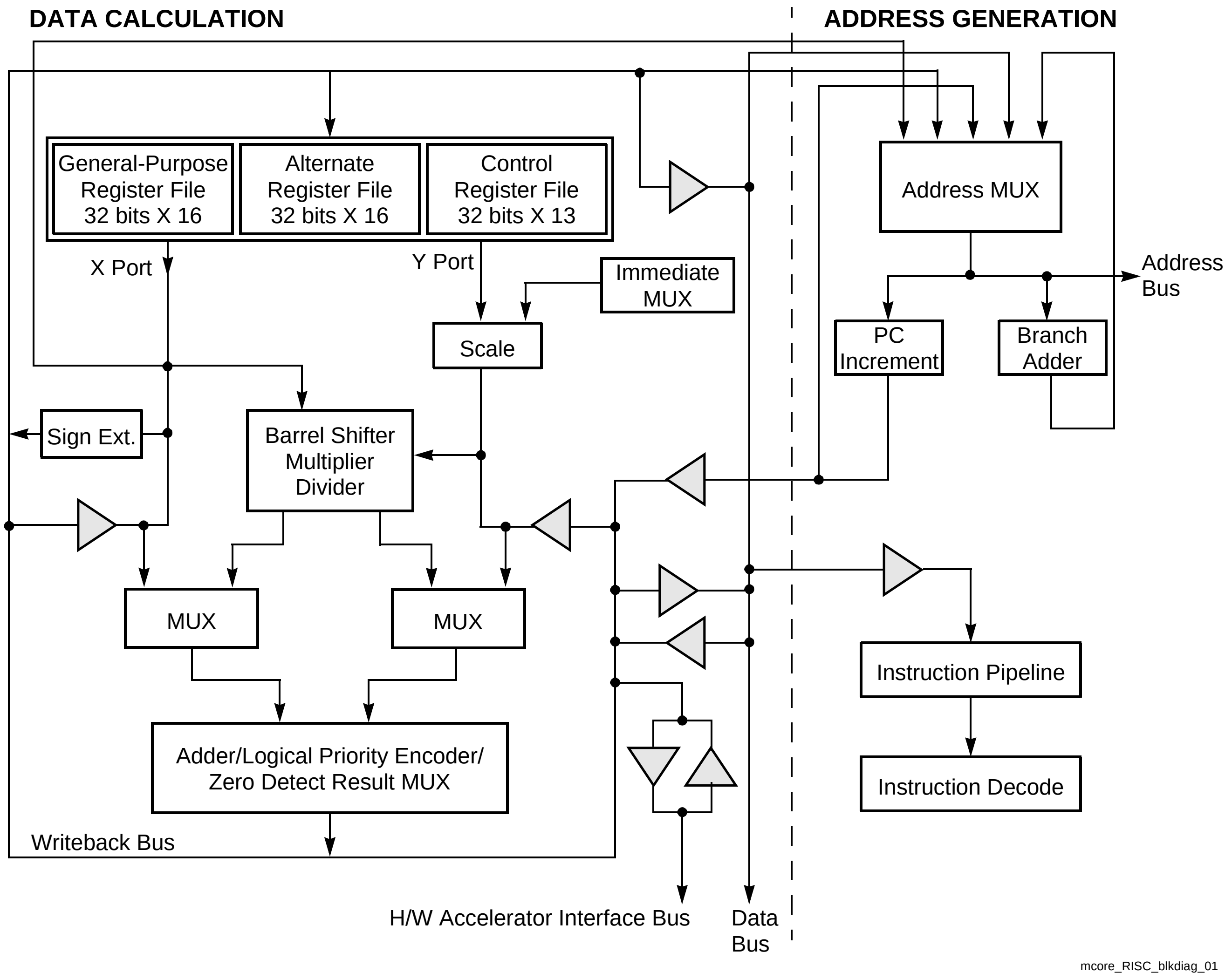
DATA CALCULATION
ADDRESS GENERATION
General-Purpose Register File 32 bits X 16
Alternate Register File 32 bits X 16
Control Register File 32 bits X 13
Address MUX
X Port
Y Port
Immediate MUX
Address Bus
Scale
PC Increment
Branch Adder
Sign Ext.
Barrel Shifter Multiplier Divider
MUX
MUX
Instruction Pipeline
Adder/Logical Priority Encoder/ Zero Detect Result MUX
Instruction Decode
Writeback Bus
H/W Accelerator Interface Bus
Data Bus
mcore_RISC_blkdiag_01

Figure 1 M•CORE Processor Block Diagram

The processor utilizes a four-stage pipeline for instruction execution. The instruction fetch, instruction decode/register file read, execute, and register file writeback stages operate in an overlapped fashion, allowing single clock instruction execution for most instructions.

The execution unit consists of a 32-bit arithmetic/logic unit, a 32-bit barrel shifter, a find-first-one unit, result feed-forward hardware, and miscellaneous support hardware for multiplication, division, and multiple-register loads and stores.

Arithmetic and logical operations are executed in a single cycle. Multiplication is implemented with a 2-bit per clock, overlapped-scan, modified Booth algorithm with early-out capability, to reduce execution time for operations with small multipliers. Divide is implemented with a 1-bit per clock early-in algorithm. The find-first-one unit operates in a single clock cycle.

The program counter unit incorporates a dedicated branch address adder to minimize delays during change of flow operations. Branch target addresses are calculated in parallel with branch instruction decode. Taken branches and jumps require only two clocks; branches which are not taken execute in a single clock.

Memory load and store operations are provided for 8-bit (byte), 16-bit (halfword), and 32-bit (word) data, with automatic zero extension of byte and halfword load data. These instructions can execute in as few as two clock cycles. Load and store multiple register instructions allow low overhead context save and restore operations. These instructions can execute in (N+1) clock cycles, where N is the number of registers to transfer.

A condition/code carry (C) bit is provided for condition testing and for use in implementing arithmetic and logical operations with operands/results greater than 32 bits. The C bit is typically set by explicit test/comparison operations, not as a side-effect of normal instruction operation. Exceptions to this rule occur for specialized operations where it is desirable to combine condition setting with actual computation.

The processor supports both normal and fast interrupt requests. Fast interrupts take precedence over normal interrupts. Both types have dedicated exception shadow registers. For service requests of either kind, a 7-bit interrupt vector can be supplied when the request is made, or automatic vector generation can be requested by means of an autovector control signal.

Programming Model

Figure 2 shows the M•CORE programming model. The model is defined differently for supervisor and user privilege modes. By convention, in both modes R15 serves as the link register for subroutine calls. R0 is typically used as stack pointer.

The user programming model consists of 16 general-purpose 32-bit registers (R[15:0]), the 32-bit PC and the C bit. The C bit is implemented as bit 0 of the PSR, and is the only portion of the PSR accessible in the user model.

The supervisor programming model consists of the user model plus 16 additional 32-bit general-purpose registers (R[15:0]’, or the alternate file), the entire PSR, and a set of status/control registers (CR[12:0]). Setting the S bit in the PSR enables supervisor mode operation.

The alternate file allows very low overhead context switching for real-time event handling. While the alternate file is enabled, general-purpose operands are accessed from it.

The vector base register (VBR) determines the base address of the exception vector table.

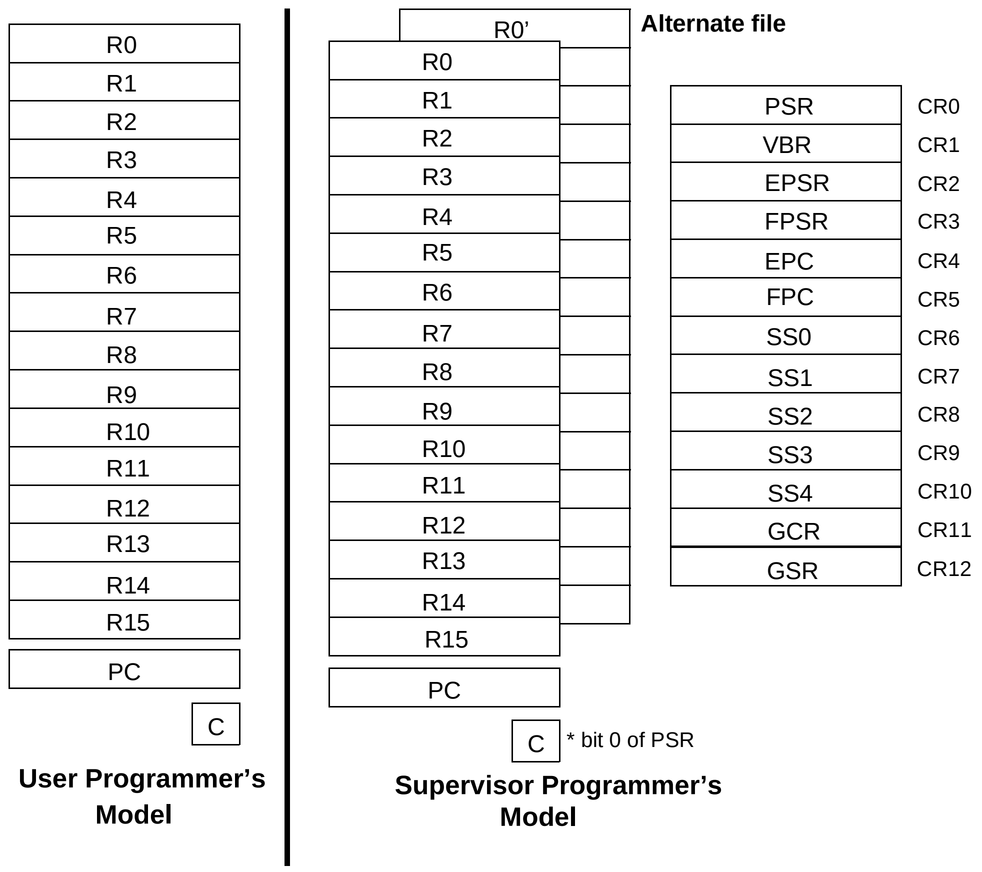

Figure 2 Programming Model

Exception shadow registers EPC and EPSR are used to save the state of the PSR and the program counter when an exception occurs. Shadow registers FPC and FPSR are used to minimize context switching overhead for fast interrupts.

Scratch registers (SS[4:0]) are used to handle exception events.

The global control (GCR) and status (GSR) registers can be used for a variety of system monitoring tasks.

Signal Function

The M•CORE architecture provides a comprehensive set of bus, status, and control signals.

Figure 3 shows the functional grouping of M•CORE signals.

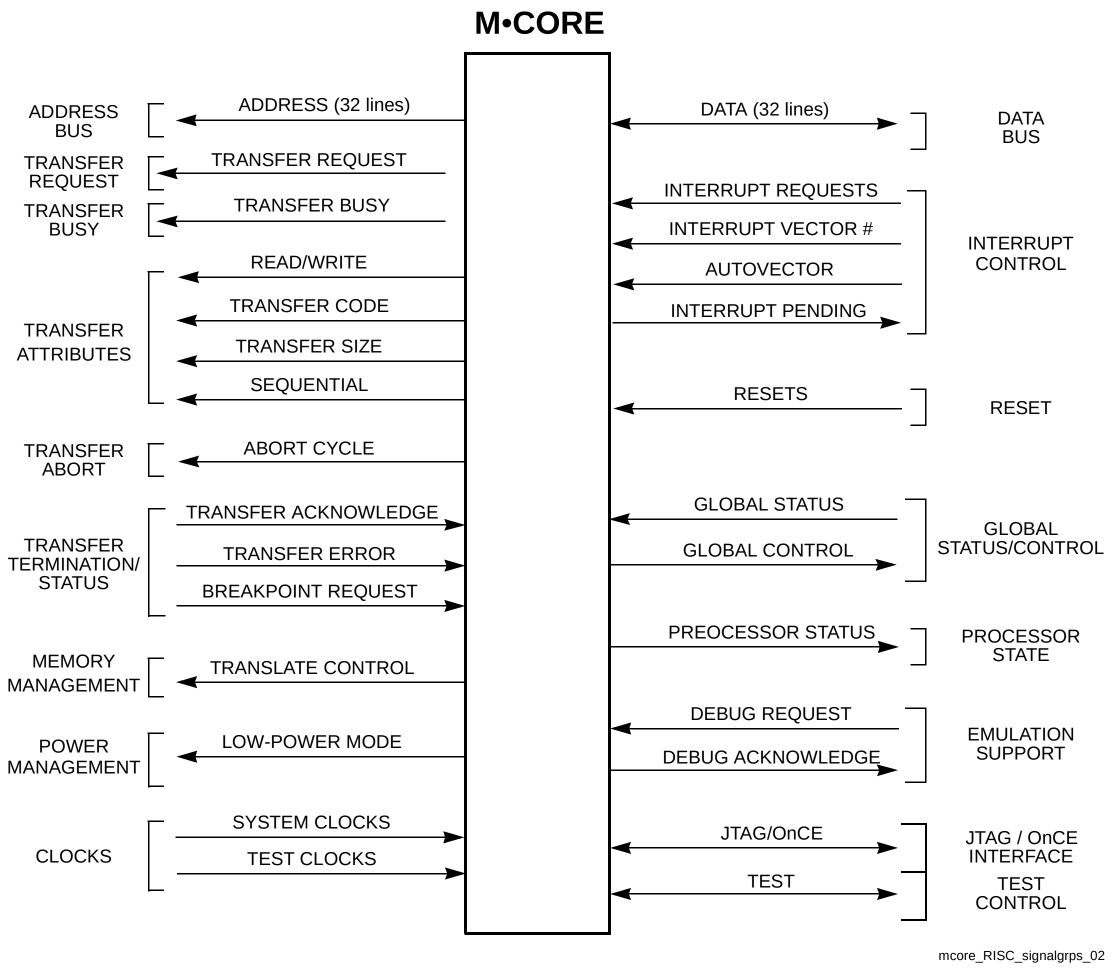

Figure 3 M•CORE Signal Groups

Code Density

The M•CORE engine minimizes memory system overhead by using 16-bit instruction encoding. This choice significantly lowers the memory bandwidth needed to sustain a high rate of instruction execution. Careful selection of instructions allows for a compact data structure and a small overall memory footprint. M•CORE supports 8, 16 and 32-bit data transfers, but is optimized for 16-bit off-chip memory. This allows a design to use less expensive and smaller memories, decreasing overall system cost, and also uses less memory on-chip in more integrated solutions.

Power Management

M•CORE's industry-leading design maximizes power efficiency. Both static and dynamic power-enhancing features are included in the architecture and implementation.

The fixed, 16-bit instruction set supports efficient access to both internal and external memory. Sixteen-bit instruction mapping also provides a compact memory image, which minimizes the number of accesses to both internal and external memory.

The **stop**, **doze**, and **wait** instructions allow designers to reduce system power consumption by invoking power-saving operating modes. The functionality of these modes is not determined by the core itself, but is implemented in an on-chip power management module, thus providing added flexibility to the designer.

The M•CORE processor also provides output signals associated with the execution of each power-conservation instruction. These signals can be monitored by external logic to control operation of the core, as well as the rest of the system.

The M•CORE has a compact die area of 2.2 mm sq., in 0.36 (L effective) micron CMOS technology. The logic and routing capacitance have been minimized. Gated clocks play a major role in minimizing unnecessary or spurious bus transitions in the data path portion of the design.

Debug Interface

The M•CORE architecture supports OnCE™ (on-chip emulation) circuitry that communicates via a standard JTAG interface. OnCE provides a non-intrusive means of interacting with the M•CORE processor core and on-chip peripherals, thus facilitating hardware/software development. Internal status and control registers are accessible via the OnCE serial scan chain. Special circuits and dedicated pins are provided to avoid sacrificing any user-accessible on-chip resources during debugging.

Target Applications

The M•CORE architecture represents a new level of performance for embedded applications — an innovative, ultra-low power microRISC core designed to power a new generation of portable and mobile applications. It provides full 32-bit performance out of 16-bit memory in a compact, low-power design, offering superior solutions for applications where battery life and system costs are as important as MIPS.

M•CORE microcontrollers deliver a high-performance, low-power solution ideally suited for battery-powered and wireless applications such as digital phones, pagers, and electronic personal assistants. M•CORE controllers are also well-suited to applications where total system cost and extended temperature range are key factors, including automotive safety products such as anti-lock braking and airbag systems.

Road Map

Building on the initial success of the base M•CORE design in the hand-held communications and automotive markets, new versions of the core with bus arbitration logic, memory management logic, enhanced numeric processing capability and floating-point capability are under development. Each new version will be upwardly code compatible with previous designs, and fully supported by a wide variety of standardized development tools.

Development Tools and Evaluation Systems

A solid selection of M•CORE development tools is available from Freescale and from established third-party vendors. Tools include highly optimized compilers, instruction-set simulators, debuggers, target monitors, software/hardware co-verification tools, real-time operating systems, and evaluation boards, as shown in **Table 1**.

To assure high-quality development tools, the M•CORE technology center has developed an application binary interface standard that defines requirements for creating M•CORE toolchain components. Adherence to the standard guarantees interoperability with other ABI compliant tools, and allows developers to evaluate and select tools based on performance, rather than compatibility.

To provide additional support and protection, M•CORE development tools are fully tested to ensure "customer-ready" validation before release.

Table 1 M•CORE Development Tools

Company Name	Web Site
Compilers/Debuggers	
Cosmic	www.std.com/cosmic
Diab Data	www.ddi.com
Green Hills	www.ghs.com
HIWARE	www.hiware.com
Metrowerks	www.metrowerks.com
Freescale GNU	www.freescale.com/mcore
Software Development Systems (SDS)	www.sdsi.com
RTOS	
Accelerated Technology, Inc.	www.atinucleus.com
Embedded System Products	www.esphou.com
Integrated Systems	www.isi.com
Microware	www.microware.com
Microtec	www.mentorg.com
Freescale RTEK	www.freescale.com/rtek
Precise Software Technologies	www.psti.com
Wind River Systems	www.wrs.com
Instruction Set Simulators	
Green Hills	www.ghs.com
HIWARE	www.hiware.com
Software Development Systems (SDS)	www.sdsi.com
Software/Hardware Verification	
HIWARE	www.hiware.com
Mentor Graphics	www.mentorg.com
Summit	www.summit-design.com
ViewLogic/Eagle Design	www.viewlogic.com
Logic Analyzers	
Hewlett-Packard (Processor Probe, Logic Analyzer)	www.hp.com
iSystem Emulator)	www.isystem.com
Tektronix (Logic Analyzer)	www.tek.com
Development Boards	
MMCEVB1200	www.freescale.com/mcore
MMCCMB1200	www.freescale.com/mcore
MMCFPGA1200	www.freescale.com/mcore
MMCLAB01	www.freescale.com/mcore

More Information

For more information on the M•CORE architecture and embedded system components that use M•CORE technology, please contact your local sales office or M•CORE Market Development at (512) 342-6974.

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
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 or +81 3 5437 9125
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
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

freescale™ semiconductor

MCORE/D