Pre-silicon Software Development for the Freescale MCF5282

Freescale Semiconductor
Authors, Marlan Winter, Jeff Kehres and Ginger Hansel

Document Number: MCF5282WP
Rev. 0
11/2005

freescale semiconductor

Using an MCF5272 evaluation board and a real-time operating system, you can start developing software for the MCF5282 before the silicon is available.

CONTENTS

When 5-micron semiconductors were commonplace, there was no such thing as an application-specific microprocessor (MPU). Boards were designed around a general-purpose central processing unit (CPU) and peripherals were added as external integrated circuits (ICs). As long as the instruction set remained the same, these CPUs could easily be swapped without affecting software developers. This allowed engineers to write code for next generation systems without worrying about the availability of silicon for new CPUs.

Embedded systems today, however, employ sophisticated processors with on-chip peripheral sets tailored to specific applications. Witness the Freescale MCF5272, which has random access memory (RAM), Ethernet, two universal asynchronous receivers/transmitters (UARTs), timers, direct memory access (DMA) controllers, pulse width modulators (PWMs) and universal serial bus (USB) all on a single integrated piece of silicon. Although the MCF5272 carries the same processor core as most of the other microcontrollers (MCUs) in the Freescale ColdFire® product family, it has a different set of peripherals from its siblings. As a result, software developers are very reluctant to start application development without having a piece of silicon in their hands because each revision of an MCU creates a new and unique hardware platform for which application code has to be retargeted.

As new application-specific MCUs roll out, such as the MCF5282, the amount of integration per unit cost of a microcontroller is raised significantly over previous microcontrollers. Companies that adopt this device will enjoy an advantage in the market over companies that do not adopt this part. Assume, for example, there is a 400 percent markup over a project's bill of materials to the final price. Furthermore, assume that if adoption of the MCF5282 saves just $5 in a final electronics design, then the selling price can drop $20 or costs can drop $5, or you can split the difference. Either way, if a head start over the competition were achieved, it would be possible to add features, enjoy a cost benefit, enjoy a profit benefit and/or gain market share in that time period. In order to achieve this head start, the development cycle must start now before the MCF5282 is available. But how is this possible without real silicon?

1. An Out-of-the-Box Strategy

The first step in pre-silicon development is to create highly portable code. With a portable application written to run on silicon that exists today, it can be quickly and cheaply moved to next generation chips as soon as they become available. This objective can be met by introducing an abstraction layer to isolate the application software from the hardware. This abstraction layer is called an operating system, or in the embedded world, a real-time operating system (RTOS). While most people think of an RTOS as a method and discipline to create deterministic code, it is the portability of this code that has the most important business implications.

In writing an application for an RTOS, hardware is manipulated indirectly via a set of functions created by the RTOS vendor. The syntax of these functions is referred to as the application programming interface (API). As long as the API is held fixed, the implementation of these functions can be changed without affecting the original application code. This allows the RTOS vendor to create implementations tailored for many different microcontrollers while conforming to a common API. The RTOS effectively hides the hardware details from the software developer and decouples the application from the hardware on which it runs. With no architecture specific code in an application (it has all been moved to the RTOS), it is portable and can be freely moved among the chips supported by that RTOS.

From an application developer's perspective, an RTOS allows a program to be moved from hardware system to hardware system with very little modification because an RTOS vendor will have a consistent API across many sets of hardware. This prevents application code from being locked into a particular microcontroller architecture. Although the code is now locked into a specific RTOS architecture, it is still far more flexible in its reuse. Ultimately, some cost must be paid for hardware independence. An RTOS, however, is a very economic investment relative to engineering expenses for porting hardware-dependent applications to new microcontroller architectures. There are many other advantages to using an RTOS such as quality, time to market and more efficient team programming capabilities, but that is for a different discussion.

Although an RTOS vendor will most likely provide libraries to utilize common peripherals on the chips it supports, it will not be able to create drivers for every peripheral on every supported chip. Communication protocol stacks (sometimes sold separately by RTOS vendor) and third-party device drivers extend the abstraction layer of an RTOS by observing the same principles of indirect hardware manipulation. In situations where no device driver is commercially available, an application developer has two options: hire a contractor to write the missing driver or write it from scratch. Deciding which path to take depends of the developer's budget, experience level and starting point. If source code (provided under certain license agreements) is available that can be easily modified, the choice may be clear.

2. Hardware Nuts and Bolts

The second step in pre-silicon development is to emulate the peripheral set of the nonexistent chip as best as possible. Using temporary external ICs and software routines, a closely related MCU can be adapted into a platform for developing code targeted for new silicon. This platform, although not perfect, can allow a significant amount of an application to be written as long as the limitations of such an approach are acknowledged.

In the case of the new MCF5282, Freescale is advocating the use of the MCF5272 as an interim development solution until MCF5282 silicon arrives. To completely understand the differences between these two chips, please reference the Freescale application note AN2394/D from the MCF5272 or MCF5282 Web site. This application note describes peripherals only found on the MCF5282, peripherals only found on the MCF5272, and minor variations between peripherals found on both chips. A summary of the new features found on the MCF5282 but not on the MCF5272 is as follows:

- Phased-Locked Loop (PLL)
- FlexCAN (CAN = Controller Area Network)
- Inter-Integrated Circuit (I^2C) serial communication bus
- General Purpose Timer (GPT)
- Periodic Interrupt Timer (PIT)
- Flash memory
- Queued Analog-to-Digital Converter (QADC)

Of these new features, FlexCAN, I^2C, and QDAC can be emulated when using the MCF5272.

First, to emulate the FlexCAN peripheral, Freescale has created a reference design (RDM5272C3CAN) for a CAN daughter card that plugs into the MCF5272C3 evaluation board (EVB). Schematics, an application note (AN2320/D), and software drivers for this card can be obtained from the “Reference Designs and Systems” section on the MCF5272 Web site.

Next, the I^2C peripheral that exists on the MCF5282 can be simulated using a software routine on the MCF5272. Freescale is currently developing this free software library and will post the code on the MCF5272 Web site as soon as it has been completed.

Finally, the QADC can be approximated using an external IC from either Texas Instruments or Maxim. The Texas Instruments part (TLV1548) is a 10-bit analog-to-digital (ADC) converter which can be connected to the queued serial peripheral interface (QSPI) on the MCF5272. The Maxim part (MAX155) is an 8-bit, multi-channel ADC.

3. The CodeWarrior™ Advantage

Using the process outlined in this white paper, software developers can start writing applications for MCF5282 today using a real-time operating system and an MCF5272 evaluation board. Moving the development cycle earlier in time translates into collecting revenues earlier. The time-value of money means more interest is collected on earlier revenue versus later revenue making early development for the MCF5282 even more profitable.

Assisting developers with the implementation details of pre-silicon development is a core competency of Freescale. As an innovative tools provider, Freescale creates products like CodeWarrior Development Tools for ColdFire Embedded Systems—an award winning integrated development environment (IDE) for building embedded ColdFire based applications. Through partnerships with ARC International and Quadros Systems, Freescale is able to offer two complete RTOS solutions for ColdFire that feature total integration with CodeWarrior. Freescale is a key partner in the upcoming RTOS tools and silicon bundles created specifically for the MCF5282.

A customer adopting CodeWarrior development tools can take advantage of a multi-tool, multi-processor, multi-language RTOS-agnostic environment. The APIs for Freescale's tools are available to all interested parties, which means that kernel-awareness can be added by anyone to any RTOS-commercial, open-source or proprietary. Freescale works closely with its preferred RTOS suppliers to ensure compatibility and integration with CodeWarrior development tools right out of the box. Adopting the CodeWarrior IDE means that writing code, compiling, debugging and testing can all be handled in a consistent manner regardless of RTOS, target processor or programming language. Freescale is committed to making sure that customers' risks are mitigated within every processor family by creating an environment that makes it is easy to port applications to other targets.

Freescale is a company dedicated to always being available to assist engineers and programmers with every step of the development cycle. With tech support personnel only a phone call away, field application engineers located throughout the world, and a world-class staff of embedded systems experts offering numerous consulting and design services, Freescale is committed to being an active partner with each and every customer. Visit www.freescale.com to learn more about our solutions for embedded systems.

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

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 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

303-675-2140

Fax: 303-675-2150

LDCForFreescaleSemiconductor

@hibbertgroup.com

Document Number: MCF5282WP
Rev. 0
11/2005

freescale semiconductor