Freescale Semiconductor
Application Note

AN3233
Rev. 0, 2/2006

MAC7100 eMIOS Example Configurations

by: David McMenamin
MCD Applications, East Kilbride

1 Introduction

All devices in the Freescale MAC7100 family of microcontrollers feature an enhanced modular I/O subsystem (eMIOS).

This application note is intended to provide the reader with a better working knowledge of the eMIOS module and its features. This is achieved by presenting a selection of example configurations that cover a wide range of the eMIOS functionality, along with background information on the module. After reading this application note, you should have an understanding of how the eMIOS hardware operates, and the knowledge required to configure the eMIOS for use in an application.

This document is intended to complement the information contained within the MAC7100 reference manual (MAC7100RM). The reference manual and the examples used in this application note are available to download from www.freescale.com/mac7100.

Table of Contents

freescale semiconductor

1.1 Background

The eMIOS is an evolution of the previously implemented MIOS module that is present in other Freescale products such as the MPC500 family. The eMIOS module provides the ability to generate and measure a range of timed events.

The MAC7100 features a 16-channel implementation of the eMIOS. Each of the sixteen unified channels is identical and can operate independently of the other channels, providing either a timed input or output function. Unlike the MIOS channels, which supported only a fixed timer function, each eMIOS unified channel can operate in any of the supported timer functions. Each eMIOS channel is multiplexed with a GPIO pin on port F.

Each channel has its own counter that is used for reference when generating and measuring timed events. This counter is driven by a clock that can be scaled at both module and channel level. Channel counters are based on a resolution of sixteen bits. Three counter buses can be used to share a common time reference between the unified channels. Each channel has access to two of the counter buses. Figure 1 shows an overview of the MAC7100 eMIOS module.

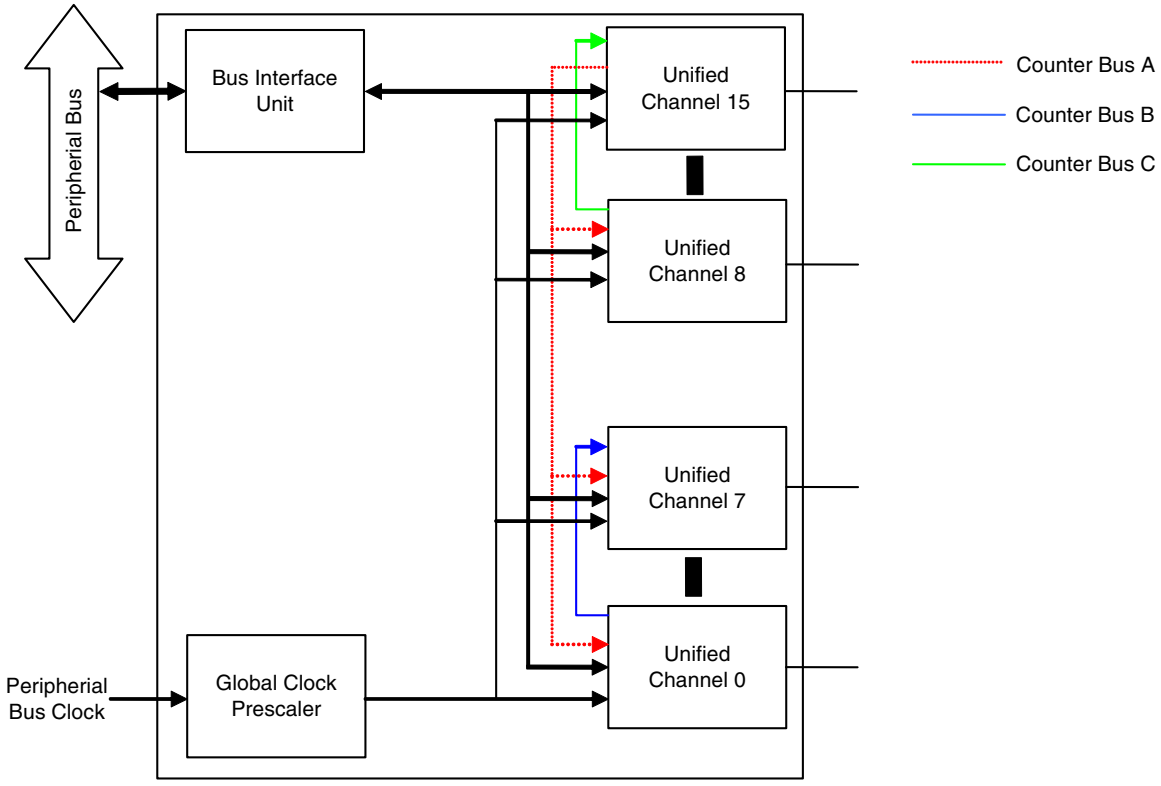


Figure 1. eMIOS Block Diagram

1.2 Supported Timer Functions

Each of the unified channels supports the range of timer functions detailed in Table 1.

Table 1. Unified channels Supported timer functions

Function		Description
GPIO	General Purpose Input/Output	Allows the output pin to be used as a GPIO pin while configured for peripheral mode within the port integration module (PIM). GPIO mode is also used when the channel is being reconfigured as it resets the internal registers.
Input Modes	Single Action Input Capture	Captures the time when the state at the input pin changes.
	Input Pulse Width Measurement	Measures the pulse width of the signal supplied at the channels input
	Input Period Measurement	Measures the period of the signal supplied at the channels input
	Pulse/Edge Counting	Measures the time taken for a specified number of pulses or edges to be accumulated
	Pulse/Edge Accumulation	Counts the number of pulses or edges that occur within a specified time window.
	Quadrature Decode	Count and direction, and Phase A Phase B modes are supported.
	Windowed Programmable Time Accumulation	Counts the high time of an input signal within a specified time window.
Output Modes	Single Action Output Compare	Changes the state of the pin output at a set period
	Double Action Output Compare	Toggles the state of the pin at two specified times.
	Output Pulse Width and Frequency Modulation	Generates a PWM output of which the frequency and pulse width are variable.
	Center Aligned Output Pulse Width Modulation	Generates a center aligned PWM output. Used for driving brushless DC motors. Offers better EMI when compared to edge aligned outputs.
	Output Pulse Width Modulation	Generates a PWM output.
Time Base Generation	Modulus Counter	Supports both up and up/down counts. This mode is used to drive a timer that can be shared on the counter bus.

Each channel can operate in any of the modes supported, although the channel can be configured in only one mode at any given time. It is possible to reconfigure a channel during run time, via GPIO mode. Examples of this are provided later in this document.

1.3 Unified Channel Hardware Overview

Each channel has its own internal 16-bit counter and two hardware comparators. Two A and B registers, A1/A2 and B1/B2 respectively, can compare and store values from either the internal counter or the value shared on the counter bus. Although the UC has two sets of A and B registers, A1/A2 and B1/B2, these registers are mapped to a single “Virtual” A and B register, which is automatically linked to the applicable A1/A2 or B1/B2 register based on the UC mode of operation. This is shown in Figure 2.

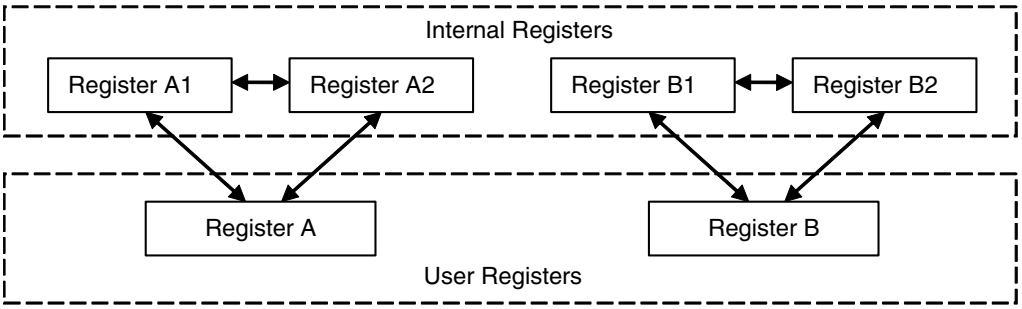


Figure 2. A and B Registers

Both the internal counter and the counter buses are 16-bit. The internal counter is clocked from the eMIOS modules prescaled clock, which can be divided down further within the unified channel. The clocking structure is discussed in more detail in Section 2.1, "Clocking".

In addition, each channel also provides the following functionality.

- A configurable input filter for noise and glitches.
- A state machine to control the hardware for the chosen timer function.
- A configurable active state for the channels output.

For more in-depth information on the architecture and features of the unified channel, refer to the eMIOS chapter within the MAC7100 reference manual.

1.4 eMIOS Principles of Operation

The generation of timed outputs in the eMIOS channels is based upon hardware comparisons between user defined values, stored in the A and B registers, and the chosen time reference: internal counter or the value shared via the counter bus. The user defined values are continually compared to the selected counter. When a match occurs, this can correspond to a change of state of an output signal. This concept is illustrated in Figure 3, which demonstrates how a simple pulse output can be generated, based on comparator matches.

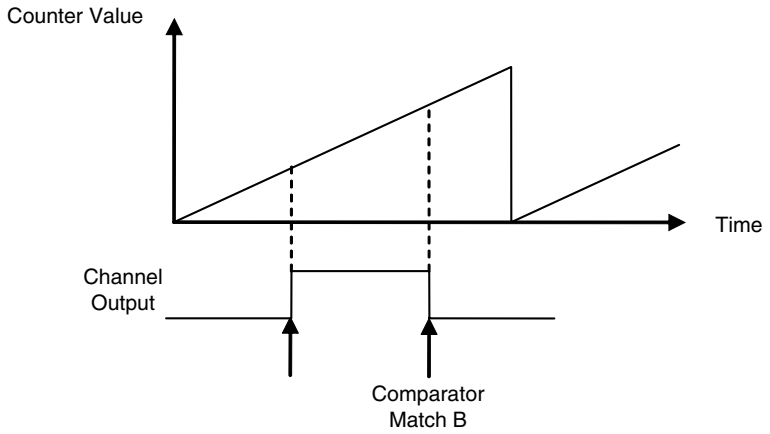


Figure 3. eMIOS Principle of Operation: Output Modes

To perform time measurements on input signals, the unified channel captures and stores the value of the counter when the appropriate change of state occurs at the input. The counter values are captured in the A and B registers. Figure 4 shows how the period of an input waveform can be measured using this technique.

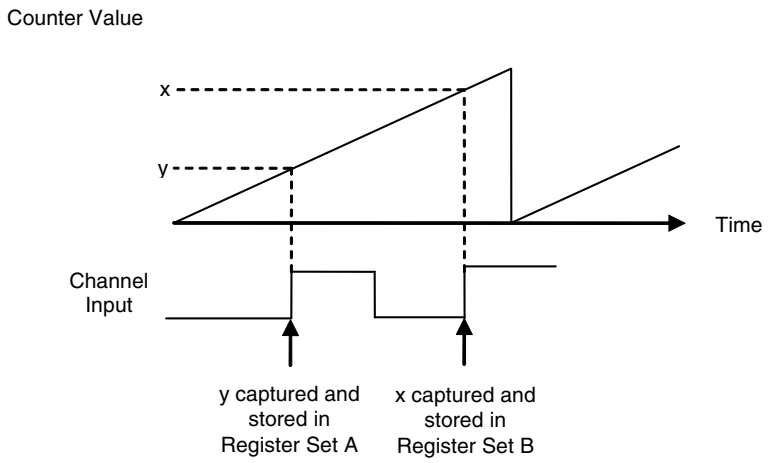


Figure 4. eMIOS Principle of Operation: Input Modes

The hardware can also increment its counter as input edges are detected. This allows for edge counting modes to be supported.

Using these principles and combinations of these principles enables the eMIOS unified channels to support a range of timing functions.

2 Time Base Generation and Sharing

2.1 Clocking

For modes that require a time base, a clock must be supplied to drive the channel's internal counter. The frequency of this clock establishes the speed at which the channels internal counter increments. This determines the frequency of signals that the channel can generate and perform measurements on.

The eMIOS module is clocked from the peripheral bus clock. This clock operates at half the speed of the system clock. A global prescaler within the eMIOS allows this clock to be divided before it is distributed to each of the unified channels. The clock can be divided by any integer value between 1 and 256 at this stage. Within the unified channel, the clock can be divided locally by the channel's clock prescaler, before being supplied to the counter hardware. The channel prescaler can divide the clock by 1, 2, 3 or 4. The flow of the system clock to the unified channel is shown in Figure 5.

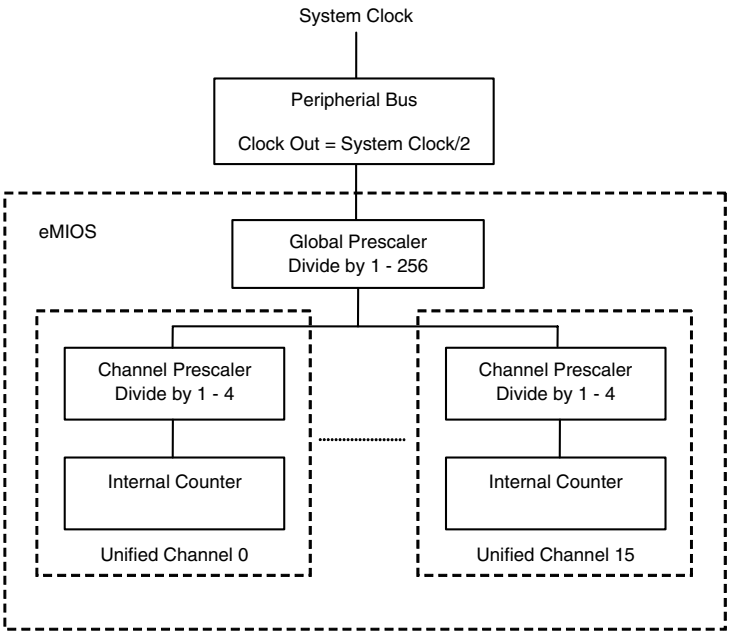


Figure 5. Clock Routing

NOTE

To enable the clock in the eMIOS module, the global time base enable bit must be set in the eMIOS module configuration register. To enable the clock at each channel, both the channel and global prescaler must also be enabled and configured, even if prescaler values of 1 are required. This is a common omission, when configuring the module, that often leads to the question, "Why is the eMIOS not working?"

2.2 Counter Buses

To provide a group of channels with a synchronized time base, there are three counter buses A, B and C, which allow a common counter to be shared among a number of channels. The counter buses and the channels they service are shown in Figure 1. The counter is generated by the drive channel operating in modulus up counter or up/down counter mode.

Table 2 details the counter bus drive channels and the channels that have access to each counter bus. The counter bus structure is also illustrated in Figure 1.

Table 2. MAC7100 Counter Bus Details

Counter Bus	Drive Channel	Channels with Bus Access
A	15	All
B	0	1–7
C	8	9–15

When one of the three counter buses is selected by a UC, the selected bus is then used as the comparator reference, instead of the UC internal counter. The internal counter does not mirror the value on the counter bus and, hence, is free to be used for other purposes — for example, in pulse/edge counting mode, the internal counter is used to store the number of edges, as the A and B registers are used to define the time window for the count. Timer functions like this, therefore, require a counter bus reference to operate. The clocking requirements and options for each of the timer functions are detailed in the next section.

Example configurations that demonstrate the use of counter buses are covered within the examples section of this applications note.

2.3 Clocking Requirements and Options

Table 3 provides details of the timing bus that must be selected in the channel configuration register to support each mode of operation. The timers must be set up by the user for the function to operate correctly.

Timer functions can require use of one of the following:

1. either the internal counter OR a selected counter bus, or
2. only a counter bus (internal counter is required for other purposes in these modes, i.e. counts pulses), or
3. only the internal counter.

Table 3. Bus Selection Requirements for each Mode of Operation

Mode / Bus	Counter Bus Required*	Internal Counter	Internal Counter OR counter Bus
GPIO (not time base required)			
Single Action Input Capture			X
Single Action Output Compare			X
Input Pulse Width Measurement			X
Input Period Measurement			X
Double Action Output Compare			X
Pulse/Edge Accumulation	X		
Pulse/Edge Counting	X		
Quadrature Decode	X		
Windowed Programmable Time Accumulation	X		
Modulus Up Counter		X	
Modulus Up / Down Counter		X	
Output pulse width and frequency modulation		X	
Center Aligned Output Pulse Width Modulation	x		
Output Pulse Width Modulation			X

Note: The counter bus must be selected in these modes as both the counter bus and the internal counter are required to support the timing function

3 Configuration Examples

The examples that follow in this section describe and explain how to configure the eMIOS for a range of its supported timer functions. The full source code for the examples is contained within the file AN3233SW.zip. All of the software examples use the standard MAC7100 header file. A system frequency of 40 MHz is assumed throughout the examples.

These examples provide a good reference for configuring the eMIOS for the other timing functions not covered by specific code in this application note. Users of the eMIOS should also refer to the MAC7100 microcontroller family mask set errata documentation on the Freescale website.

3.1 Example 1: Double Action Output Compare

This example covers the configuration of the eMIOS module unified channel 3 to generate a double action output compare signal, using the channels internal counter as reference. The output has a pulse width of 1 ms and the channel's flag is raised on the second comparator match.

3.1.1 Overview of Double Action Output Compare

Double action output compare (DAOC) allows a single pulse to be generated on the channel's output. The leading edge of the pulse is generated when a comparator match occurs between register A and the selected time reference. The trailing edge is generated when the comparator match occurs on register B. The timing diagram for this mode is shown in Figure 6, with outputs shown for the case of EDPOL = 1 and EDPOL = 0. Note how the output signal is effectively inverted, for each value of EDPOL.

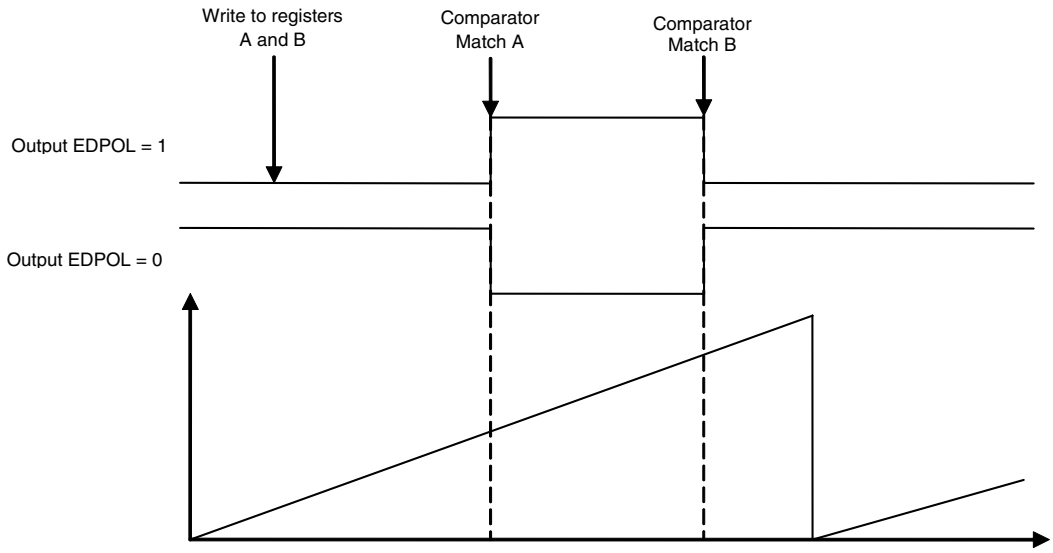


Figure 6. DAOC Overview

When using a channel in DAOC mode, the comparators are disabled until the required comparator match values have been written into registers A and B. The channel's flag can be raised on both comparator matches or on the second comparator match only.

3.1.2 Configuration Flow

The configuration flow followed and described for this example is shown in Figure 7.

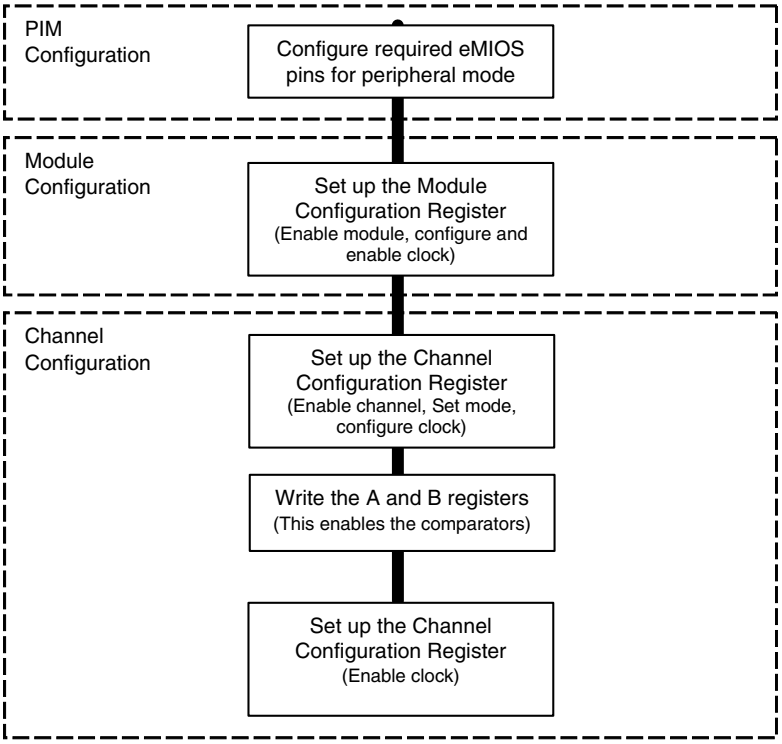


Figure 7. Example 1 Configuration Flow

3.1.3 PIM Configuration

All of the required eMIOS signals must be set for peripheral mode by setting the appropriate CONFIGn_x[MODE] bits in the PIM. For this example, port F, pin 3 must be configured for peripheral mode.

```
/* Configure PF3 for Peripheral Mode */

 PIM_PF_CONFIG(3) = 0x0080;
```

3.1.4 Module Configuration

Prior to using a channel to generate a DAOC signal, the module configuration register (MCR) must be configured to enable the eMIOS module (which is disabled from reset) and supply a suitable clock to the unified channels. Power saving and debug options for the eMIOS are also configured from within the module configuration register. However, these are outside the scope of this application note and, hence, are left in the default state in this example.

For this example, a clock of 4 MHz is supplied to the unified channels. Based on the 40 MHz system clock, this requires a global prescaler of 5 (GPRE = 4) to be set within the module configuration register. Figure 8 shows how the 4 MHz clock is generated from the system clock.

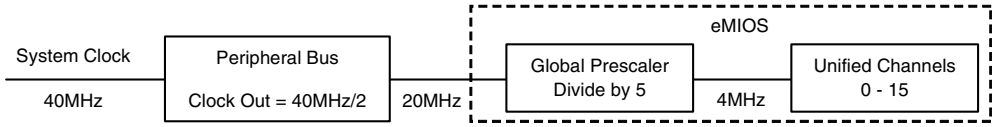


Figure 8. Clock Dividers

To correctly configure the eMIOS and enable the clock supply for the unified channel, the following options must be configured in the MCR.

- The module disable bit (MDIS) must be cleared to enable the module.
- The global time base enable bit must be set to turn on the clock supply to the unified channels.
- The global prescaler enable bit must be set. This is necessary even if the global prescaler is left in its default value.
- The global prescaler field must be set to the required value.

A single write to the MCR can configure it correctly for this example. The C code used to configure the register is:

```
/* Set up eMIOS Module Configuration Register*/

 EMIOS_MCR = (EMIOS_GTBE                    /* Global Time Base Enable */
             |EMIOS_GPREN                   /* Global Prescaler Enable */
             |EMIOS_GPRE(4));               /* Module Prescaler div 5  */
```

NOTE

For a global prescaler value of x, (x-1) must be written to the GPRE register.

When the MCR has been set up, the channels that are to be configured for use must be enabled in the disable channel register (UCDIS). From reset, all channels are disabled; to enable a channel, its corresponding bit must be cleared in the UCDIS register. Therefore, to enable channel 3 for use in this example, bit 3 must be cleared.

```
 EMIOS_UCDIS = 0xFFF7;                      /* Enable Channel 3        */
```

3.1.5 Channel Configuration

To configure channel 3 to generate a DAOC signal the channel control register (CCR) must be set up appropriately for DAOC mode; then registers A and B must be written to set the comparator match values and enable the comparators.

The rate at which the internal counter increments is determined by the clock used to drive the counter. The module has already been configured to supply a clock of 4 MHz to the unified channels. If the clock is divided by a channel prescaler of 1, the counter will increment every 250 ns over the range 0 – 0xFFFF. Therefore, it will take 16.38 ms for the counter to roll over. The pulse width of 1 ms required for this example thus takes up approximately 1/16 of the timer's range. This is suitable as it allows for flexibility

in the configuration, if it is used to generate other pulses in the same application. It allows the time at which the pulse starts to be altered and also allows pulses of varying width to be generated.

Before setting the CCR for DAOC, the channel must be configured in GPIO mode. This resets the channel's internal hardware.

NOTE

This step is necessary only if the channel has been used in another mode since the device was last reset.

For this example, the parameters in channel 3's configuration register are set as follows.

- Channel prescaler configured as 1.
- Internal counter selected as reference for the comparator.
- EDPOL bit set to ensure that the pulse is high and the idle state is low.
- Mode selected as double action output compare flag set on second match.

The code for this, including that used to reset the channel's hardware by putting that channel into GPIO, is as follows.

```
/* Prepare Channel 3 */
 EMIOS_CHC(3) = (EMIOS_UCPRE(0)              /* Channel Prescaler div 1         */
             |EMIOS_UCPREN                   /* Enable ch Prescaler             */
             |EMIOS_BSL(0x3)                 /* Select Internal Counter         */
             |EMIOS_MODE(0x1));              /* Set mode GP OUTPUT              */

/* Configure Channel 3 -Double Action Output Compare- Flag Set On Second Match */
 EMIOS_CHC(3) = (EMIOS_UCPRE(0)              /* Channel Prescaler div 1         */
             |EMIOS_BSL(0x3)                 /* Select Internal Counter         */
             |EMIOS_EDPOL                    /* Generate High Pulse             */
             |EMIOS_MODE(0x6));              /* Set mode                        */
```

At this point, the channel prescaler enable bit is not set. This puts the channel in DAOC mode, but the internal counter is not running — it remains at 0. This allows the A and B registers to be written to enable the comparators while the clock is at a known value. The advantage of this is that the time when the DAOC occurs can be determined relative to when the timer is started. It also prevents the internal counter from rolling over before the A and B registers have been written.

With the counter being driven by a 4 MHz clock, it takes 4000 counts to produce a 1 ms output pulse. Therefore, register B must be configured with a match value that is 4000 greater than that of register A. If the pulse is generated 5 ms after the timer is started, then the rising edge of the pulse should occur after

20000 counts. Thus, register A = 20000 = 0x4E20 and register B = 24000 = 0x5DC0. The register setting is done using the following C instructions.

```
EMIOS_CHB(3) = 0x5DC0;                    /* Set Match B value                */
EMIOS_CHA(3) = 0x4E20;                    /* Set Match A Value                */
```

When the A and B registers have been written, the comparators are enabled. The order in which the A and B registers are written is irrelevant for this example. The internal counter can now be started by enabling the channel prescaler:

```
                                          /* Start Channel 3's internal counter */
 EMIOS_CHC(3) |= EMIOS_UCPREN;            /* Enable CH Prescaler              */
```

The DAOC pulse occurs 5 ms after setting the channel prescaler enable bit. The channel flag is raised on match B.

3.1.6 Channel Output

The output generated from this example is illustrated in Figure 9.

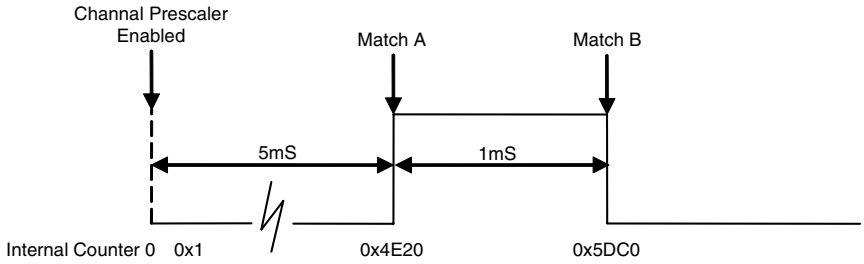


Figure 9. DAOC Example Output Signal

After the DAOC compare signal has been generated the channel output remains idle. The internal counter continues to run, but the comparators are disabled. Hence, no matches occur, and the output remains in its idle state.

To generate another DAOC signal registers A and B must be updated with new match values, or the originals if the same output is required again. It is not necessary to rewrite the channel control register, unless a different configuration is required.

This example uses the internal counter but it is also possible to generate a DAOC with the counter bus selected as the timer.

3.2 Example 2: Output Pulse Width and Frequency Modulation

In this example, channel 9 is configured to provide an output pulse width and frequency modulated signal (OPWFM). The channel is configured such that it is capable of generating outputs with 50% duty over the

frequency range 1 kHz – 500 kHz. This example focuses on the channel configuration and the updating of the parameters of the output signal at run time. The module configuration used in this example is the same as that used in example 1. Hence, it is not covered in detail in this example.

3.2.1 Overview of Output Pulse Width and Frequency Modulation

Output pulse width and frequency modulation (OPWFM) enables a channel to generate a PWM output, the pulse width and frequency of which can be altered at run time. To support the frequency being changed, this mode requires its own dedicated counter that can be reset when required. The channel's internal counter is therefore automatically used to support this mode.

Two OPWFM options are supported by the MAC7100 eMIOS — immediate update and next period update, each with the option to raise the flag on both comparator matches or on the second comparator match. In both cases, the duty is defined in register A and the period in register B. When the first match occurs (register A match), the output of the channel changes to the value of EDPOL. The second register match (register B) causes the output to change to the complement of EDPOL, and the internal counter is reset. The duty cycle output value is, therefore, the complement of EDPOL. The counter continues to run after it has been reset, and the comparators remain enabled, ensuring that matches continue to occur with the defined values.

To update the period and duty, new values must be written to registers A and B. In immediate update mode, the updates occur in the current cycle of the internal counter, provided the values have not been surpassed. In the next period update, the match does not occur with the new values until the internal counter resets and begins a new cycle. An example of OPWFM mode is shown in Figure 10 and Figure 11; both immediate and next period update are covered.

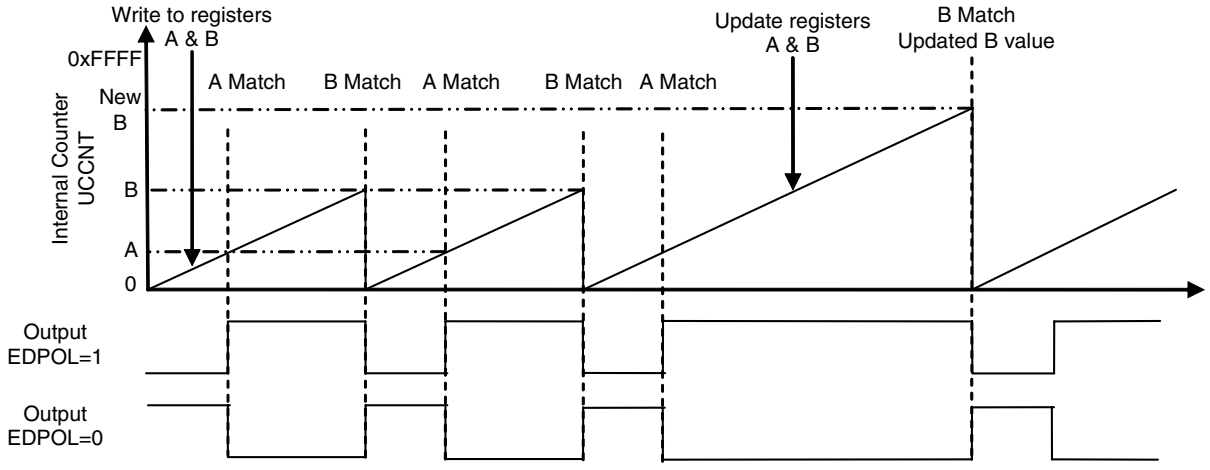


Figure 10. OPWFM Immediate Update

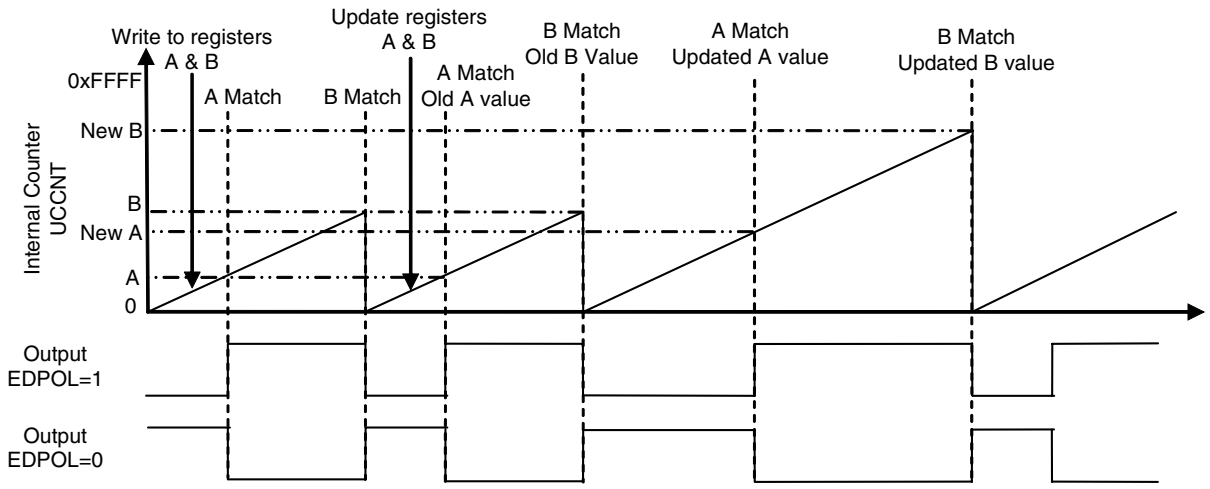


Figure 11. OPWFM Next Period Update

NOTE

When writing to the A and B registers, a value of 1 must be added to the desired value as the hardware subtracts one from the value in the A and B registers.

3.2.2 Configuration Flow

All of the steps required to configure the eMIOS module for this example are displayed in Figure 12.

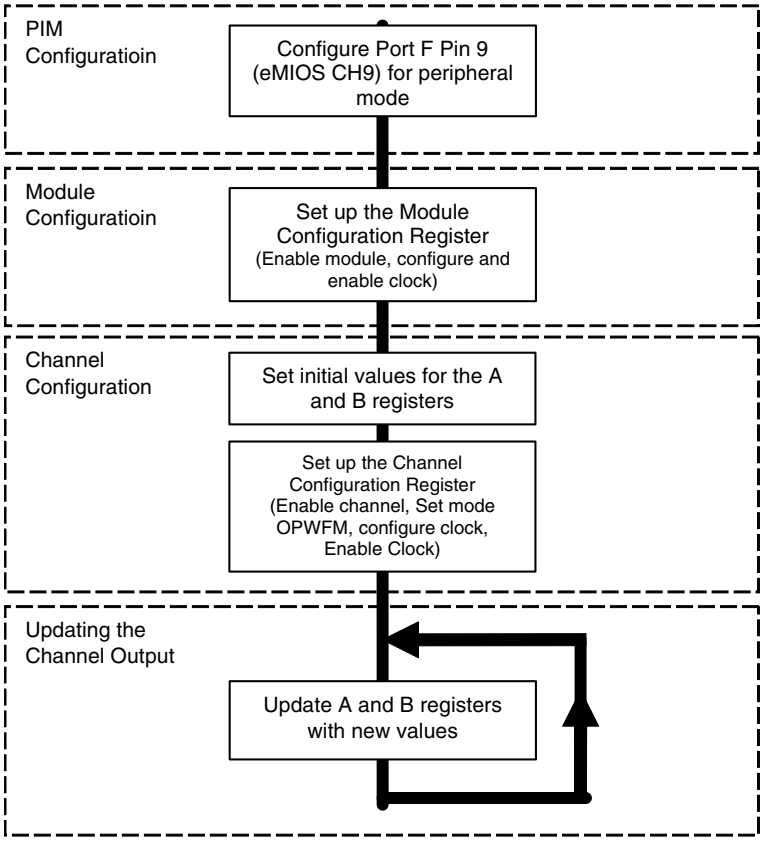


Figure 12. Example 2 Configuration Flow

3.2.3 Channel Configuration

Before writing the channel configuration register, initial values can be set for the A and B registers. When the channel configuration register is written to start the channel, and the values of A and B have already been defined, the specified output occurs within the first period of the internal counter. This prevents the counter overflow bit from being raised within the channel status register before an output has occurred.

For this example. the maximum output frequency required is 500 kHz with a 50% duty cycle. The channel is supplied with a 4 MHz clock from the module. To generate a 500 kHz 50% duty output requires a minimum clock of 2 MHz to be driving the internal counter. When writing the A and B registers to achieve the maximum frequency (B match occurring at the minimal value) with a 50% duty cycle in OPWFM mode A must be written as 1 and B as 3. This is the only special case situation where values other than the expected are required to generate the desired output. A maximum frequency of 666.7 kHz could be achieved with the 2 MHz clock, but only with a 75% duty cycle. For demonstration purposes, this example sets the channel clock prescaler to generate the minimum required clock of 2 MHz. The resultant clocking is displayed in Figure 13.

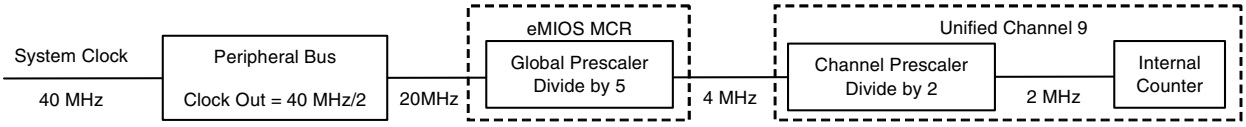


Figure 13. Example 2 Clock Structure

To start the channel output on the first counter bus value with the maximum frequency, this example writes to the A and B registers before the channel is configured.

```
/* Setup A and B for Maximum frequency @ 50% Duty Cycle */
 EMIOS_CHB(9) = 0x3;                          /* Set Match B value       */
 EMIOS_CHA(9) = 0x1;                          /* Set Match A Value       */
```

After the initial match values have been set, the channel is configured. In this example, OPWFM next period update flag set on second match mode is selected.

```
/* Configure Channel 9-OPWFM- Next period update Flag Set On Second Match */
 EMIOS_CHC(9) = (EMIOS_UCPRE(1)               /* Channel Prescaler div 2 */
            |EMIOS_UCPREN                     /* Enable ch Prescaler     */
            |EMIOS_EDPOL                      /* Set Duty Cycle as low   */
            |EMIOS_MODE(0x19));               /* Set mode OPWFM FSSM     */
```

The bus select field is left in its default state, as the internal counter is automatically used by the channel when the mode is selected as OPWFM.

3.2.4 Channel Output

As the prescaler is set and enabled along with the mode in the single write to the channel configuration register, the channel's internal counter starts after the write to the register is complete. With the values of A and B already preset, the output shown in Figure 14 is generated continuously. The output remains the same until the duty and period are updated or the channel is stopped for another reason.

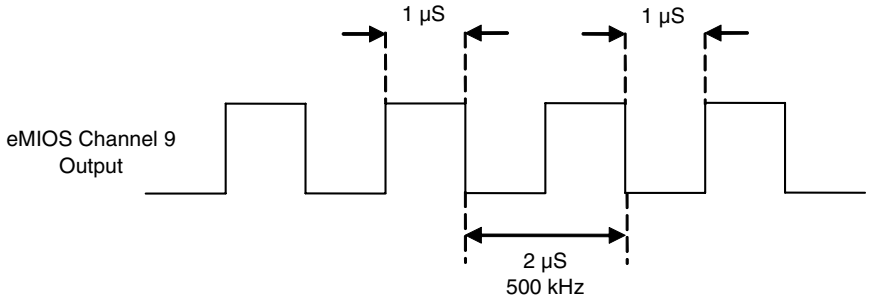


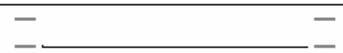
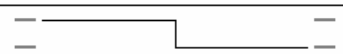
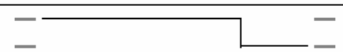
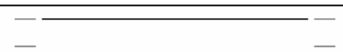
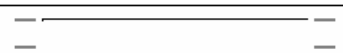
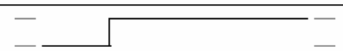
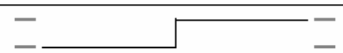
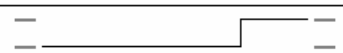
Figure 14. OPWFM Example Output Signal

3.2.5 Changing the Period and the Duty

Updating register B changes the period, whereas updating register A changes the duty cycle. Note that updates to A and B are independent.

A set of example OPWFM output waveforms is provided in Table 4. These examples show how the relationship between the values in register A and register B effects the duty cycle. The examples are based upon a fixed frequency (register B remains constant) and a variable duty. From the 2 MHz clock supplied to the internal counter, a period of 1000 decimal (register B = 1001) results in an output signal of 2 kHz.

Table 4. Example OPWFM Output Waveforms

EDPOL	Duty Cycle	A (decimal)	B (decimal)	Waveform
0 (active high output)	0%	1000	1000	
	25%	250	1000	
	50%	500	1000	
	75%	750	1000	
	100%	0	1000	
1 (active low output)	0%	1000	1000	
	25%	250	1000	
	50%	500	1000	
	75%	750	1000	
	100%	0	1000	

3.3 Example 3: Output Pulse Width Modulation Using a Counter Bus

UC0 is configured to drive counter bus B. A second channel UC5 is then configured to generate an OPWM output that uses counter bus B as its reference.

To generate an up counter to drive the counter bus, channel 0 is configured in modulus up counter mode, internal clock. Channel 5 is initially configured to generate the maximum possible output frequency from a 40 MHz clock, i.e. 10 MHz.

3.3.1 Overview of Modulus Up Counter

Modulus up counter mode is used to generate a counter that is shared on the counter bus. This can then be used, by all channels that have access to that counter bus, as the time reference for the channels' comparators. Modulus up counter mode can also be used as a general purpose timer. The clock for the modulus counter can be taken from the internal clock or from an external clock, provided via the channel's port pin.

Register B is not used when a channel is in modulus up counter mode. Register A is used to define the maximum count value. The modulus counter increments its internal counter at a rate determined by the

frequency of the clock, until it reaches the maximum count value. When the maximum count value is realized, the channel's flag is asserted and the counter resets and continues to count up. Register A can be updated at any time to provide a new match value. Figure 15 shows an example of modulus counter operation.

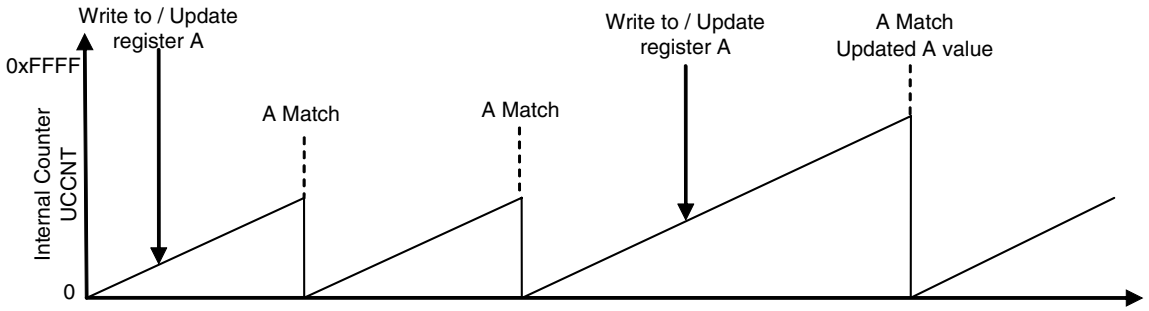


Figure 15. Modulus Up Counter

3.3.2 Overview of Output Pulse Width Modulation

Output pulse width modulation (OPWM) signals can be generated with the channel's internal counter or a modulus up counter, shared via the counter bus, as reference. The frequency of the output is determined by the time taken for the reference counter (internal counter or external counter bus counter) to roll over. The edges of the output pulse occur at the times specified in registers A and B.

When a match occurs between register A and the selected time base, the output is set to the value of EDPOL. The output is then set to the complement of EDPOL, whenever a match occurs on comparator B. As with OPWFM mode, both next period and immediate update options are available. An example of immediate update mode is shown in Figure 16.

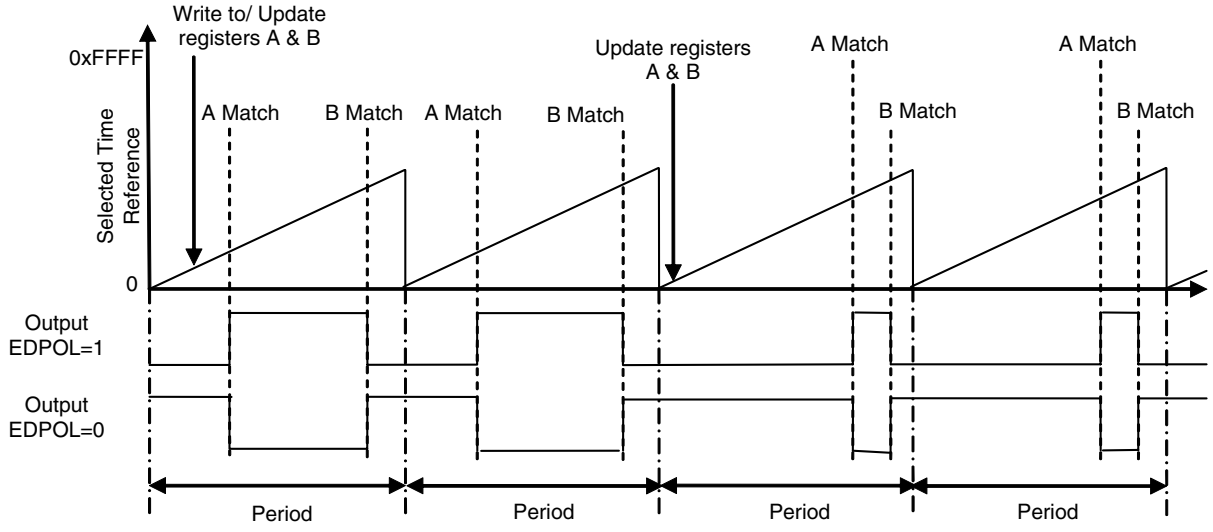


Figure 16. OPWM Immediate Update

3.3.3 Configuration Flow

The steps taken to configure the module for this example are shown in Figure 17.

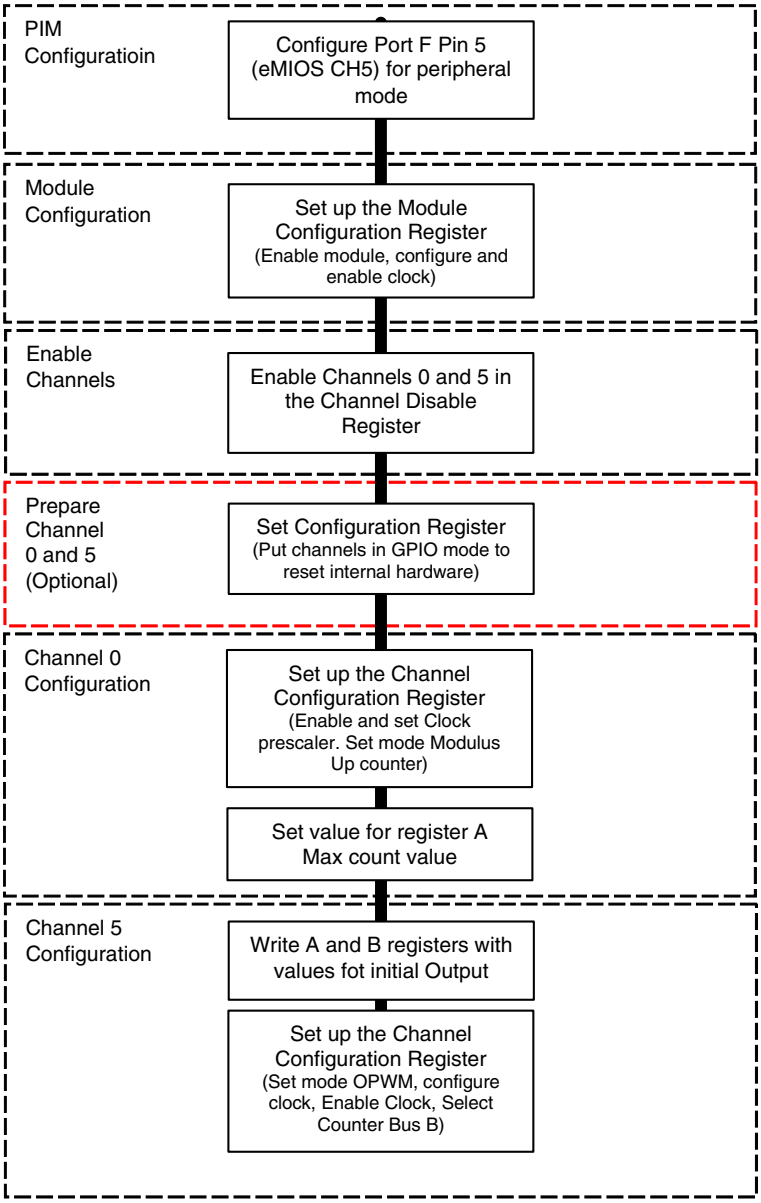


Figure 17. Example 3 Configuration Flow

3.3.4 PIM Configuration

All that is necessary is to configure PF5 for peripheral mode. Channel 0 does not drive a timed output on the pin when in modulus up counter, internal clock, mode. Therefore, it is not necessary to connect the output from channel 0 to the output pin. The pin for the channel is free to be configured and used as a GPIO from within the PIM.

```
/* Configure PF5 for Peripheral Mode */

PIM_PF_CONFIG(5) = 0x0080;
```

3.3.5 Module Configuration

To make it possible to generate the maximum output frequency, the channel must be supplied with the maximum clock speed. To achieve this, the clock prescalers must be configured to divide the clock by the lowest possible prescaler value.

The module configuration register is configured to enable the clock, enable the clock prescaler, and set the prescaler to 1.

```
/* Setup eMIOS Module Configuration Register */

 EMIOS_MCR = (EMIOS_GTBE                        /* Global Time Base Enable */
             |EMIOS_GPREN                       /* Global Prescaler Enable */
             |EMIOS_GPRE(0));                   /* Module Prescaler div 1  */
```

With the prescaler set to 1, the 20 MHz peripheral clock will be routed to all channels.

3.3.6 Channel Configuration

Both of the channels to be used must be enabled before the channel configuration register is written. Channels 0 and 5 are enabled in the channel disable register by clearing their corresponding bit in the register.

```
EMIOS_UCDIS = 0xFFDE;                           /* Enable Channel 0 & 5    */
```

When the channels have been enabled, the channel control registers can be written to configure the channel for the required timer function. Writing to a channel configuration register, or A or B register, will work only if the channel is enabled. Writing to a disabled channel will fail.

To prepare the channels, they are configured for GPIO mode before being configured for the desired timer function. If this is the first time the channels have been used since the device was reset, then this step is not necessary.

```
/* Prepare Channel 0 */

 EMIOS_CHC(0) = (EMIOS_UCPRE(0)                 /* Channel Prescaler div 1 */
               |EMIOS_UCPREN                    /* Enable ch Prescaler     */
               |EMIOS_BSL(0x3)                  /* Select Internal Counter */
               |EMIOS_MODE(0x1));               /* Set mode GP OUTPUT      */


/* Prepare Channel 5 */

 EMIOS_CHC(5) = (EMIOS_UCPRE(0)                 /* Channel Prescaler div 1 */
```

```
                |EMIOS_UCPREN                    /* Enable ch Prescaler     */
                |EMIOS_BSL(0x3)                 /* Select Internal Counter */
                |EMIOS_MODE(0x1));              /* Set mode GP OUTPUT      */
```

As the OPWM output from channel 5 is dependent on the modulus up counter being supplied by channel 0, channel 0 is configured first. Again, to generate the maximum output frequency from the module, the modulus up counter must be driven by the fastest clock possible. Thus the channel prescaler must also be configured to divide the clock by the lowest possible prescaler, which is 1. This results in a 20 MHz clock driving the internal counter of channel 0. Modulus up counter mode, generated from the internal clock, is selected.

```
/* Configure Channel 0 - Modulus Up Counter */
 EMIOS_CHC(0) = (EMIOS_UCPRE(0)          /* Channel Prescaler div 1 */
                |EMIOS_UCPREN                    /* Enable ch Prescaler     */
                |EMIOS_MODE(0x10));              /* MOD UP CNT INTCLK       */
```

To enable channel 5 to generate the maximum output frequency, which is ¼ of the system clock, the modulus up counter is set to count up to 1. This value provides minimal flexibility for other channels that wish to use this counter bus as reference, but it is required to generate the maximum frequency as it is the lowest value that can be selected as maximum.

```
EMIOS_CHA(0) = 0x1;                              /* Set Match A Value       */
```

Counter bus B is driven automatically by channel 0. The value being shared on the counter bus can be monitored in the counter register of channel 0.

The maximum frequency can be generated with a 50% duty cycle only. This is a result of setting the reference counter to the minimal possible value, limiting the values at which a match can be made. For this case, the match must be made on the following:

```
/* Setup A and B for Maximum frequency @ 50% Duty Cycle */
 EMIOS_CHB(5) = 0x0;                             /* Set Match B value       */
 EMIOS_CHA(5) = 0x1;                             /* Set Match A Value       */
```

NOTE

Note that these values are outside the expected values. This is a special case required only for generation of the maximum output frequency. With register A and B set for channel 5 writing, the configuration register causes the output to commence. When writing the configuration register counter bus B must be selected as the timer reference. As the channel's internal clock is not used for reference, it is not necessary to set or enable the internal prescaler.

```
/* Configure Channel 5 - OPWM */
 EMIOS_CHC(5) = (EMIOS_BSL(0x1)                  /* Select Counter Bus B     */
                |EMIOS_EDPOL                     /* Duty Period Is Low       */
                |EMIOS_MODE(0x20));              /* Set mode OPWFM FSSM      */
```

3.3.7 Channel Output

The output that will be generated from channel 5 is shown in Figure 18. This output remains the same until the A and B registers are updated.

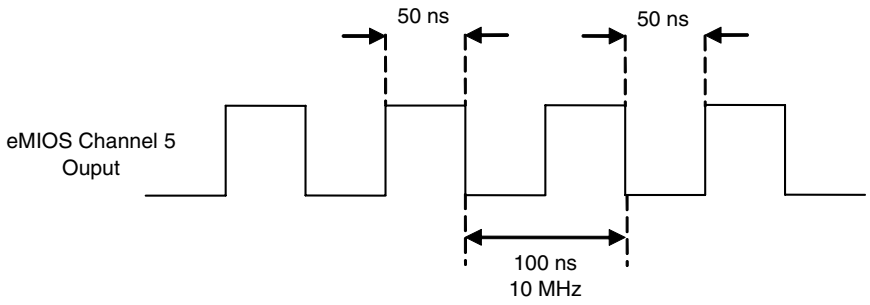


Figure 18. Example 3 OPWM Output Signal

3.3.8 Updating the Output

To enable channel 5 to generate waveforms of other frequencies and periods, the time reference must be changed from the minimal period to another value that provides a wider range of values, at which matches can be specified to occur.

With the same module and channel configuration, changing register A of channel 0 will vary the frequency of the output. Table 5 shows a range of values for channel 0, register A, and the frequencies that these will generate from channel 5. Changing the value in register A will also affect other channels using this counter bus.

Table 5. Alternate Frequencies for Example 3

Channel 0 Register A (Decimal)	Channel 5 OPWM Frequency
20	1 MHz
250	80 KHz
1000	20 KHz
5000	4 kHz
40000	500 Hz

Updating register A and B with new values allows the times at which the matching and trailing edges of the pulse occurs to be varied. The values specified must always be within the range of the value being shared on the counter bus i.e. equal to or less than the value in channel 0 Register A for this example.

When using immediate update mode, care must be taken when updating the A and B registers of the OPWM channel. Please refer to Section 3.2.5, “Changing the Period and the Duty”; the guidelines for performing register updates also apply to OPWM mode.

3.4 Example 4: Pulse Width and Period Measurement

This example is a continuation of example 3. The output that was generated in example 3 is used as an input signal in this example. An additional eMIOS channel is configured to perform a pulse width measurement and then a period measurement on the OPWM output signal from channel 5.

Channel 3 is configured to perform the period and pulse width measurement on the output from channel 5. The parameters of channel 5 and channel 0 are configured to generate a different output signal from that used in example 3, but the channel and module configuration remain the same.

Channels 3 and 5 must be physically connected for this example. On the MAC7100 EVB, this can be done by connecting a jumper across the header of port pins F3 and F5.

3.4.1 Overview of Pulse Width Measure

When configured for pulse width measure, the unified channel captures, from the counter bus, the times at which the rising and trailing edges of the pulse occur. The time of the leading edge is captured in register B, and the trailing edge in register A. This is illustrated in Figure 19. The counter bus selected as reference must be operating as a modulus up counter at a frequency suited to the range of the input signal.

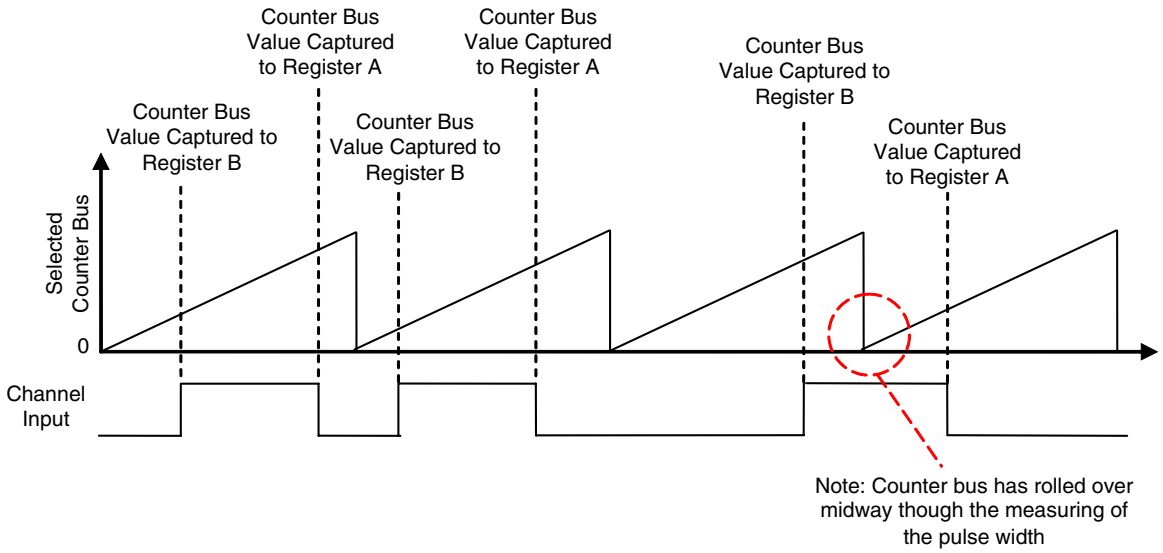


Figure 19. Pulse Width Measure

When the time, at which each edge of the pulse occurs, has been captured and results are available, the channel's flag is raised. At this point, registers A and B can be read and the pulse width can be calculated as the difference between the values in registers A and B.

If the counter bus has rolled over between the leading and trailing edges of the pulse, as shown with the third pulse in Figure 18, then the value contained in register A will be less than the value in register B. If this occurs, the pulse width can still be calculated, as the time at which the counter bus rolled over is known. To ensure that the correct result can always be obtained, the maximum pulse width of the input signal must be less than the period of the counter bus selected as reference.

The channel can be configured to measure a pulse as either high or low, by specifying the polarity of the leading edge. This is done with the EDPOL bit.

3.4.2 Overview of Period Measurement

Period measurement works in a similar way to pulse width measurement, but in this case, the UC looks for A and B matches on the same input pulse edge. The times at which two consecutive edges of the same type, rising or falling, occur are captured. The type of edge that is captured is determined by the value of the EDPOL bit. When both the edges have been captured, the channel raises its flag to signal that the period measure is complete. Reading register B returns the time at which the first edge occurred and register A the second. The period can be calculated from the difference in these values. As with pulse width measure, if the period has been captured and the counter bus has rolled over between the edges, care must be taken to acknowledge this when calculating the period.

The counter bus being used as the time reference must have a roll-over period that is greater than the maximum period of the input signal. This will ensure that the correct period can be calculated, and that no false results are generated. If the counter bus has rolled over between the capturing of the edges of the period, then the value captured in register B will be less than the value captured in register A. When this

case occurs, the period can be calculated by adding the value in register B to the counter bus roll over period, minus the value in register A.

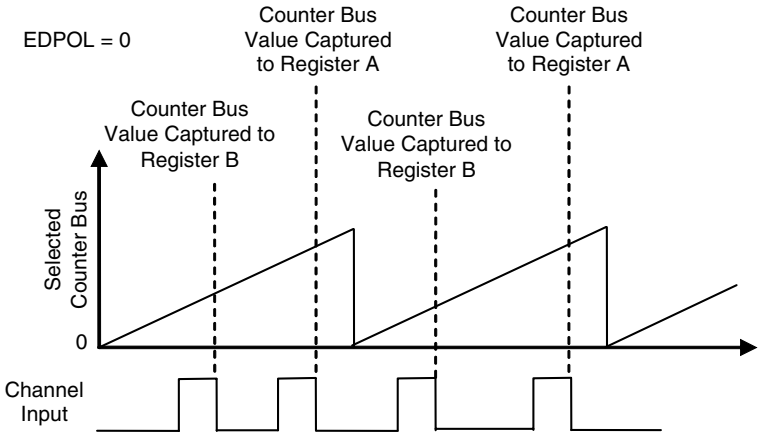


Figure 20. Period Measure

3.4.3 Configuration Flow

The configuration flow in Figure 21 shows the stages that are followed in addition to those in example 3 to configure channel 3 to perform a pulse width measurement then a period measurement on the output PWM signal. This flow assumes that the extra channels are enabled and their port pins configured along with the others; this diagram excludes this stage.

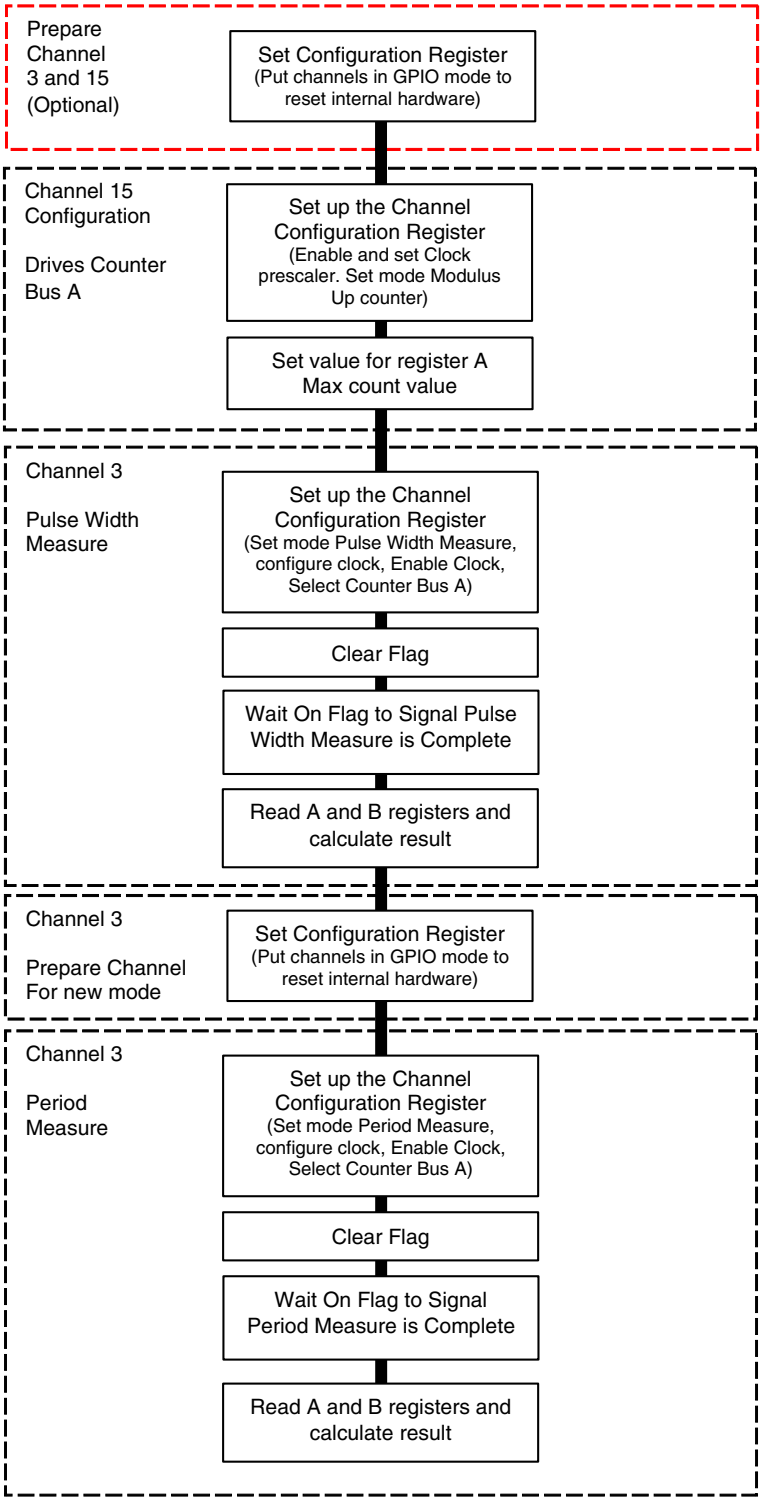


Figure 21. Example 4 Configuration Flow

3.4.4 Input signal

The input signal for channel 3 is generated from channel 5, and based upon counter bus B. For this example, the modulus up counter (supplied by channel 0) is configured to count up to 500. This results in a frequency of 40 kHz, using the same prescalers as example 3.

The OPWM leading and trailing edges are configured to occur at 237 and 123. The input signal is shown in Figure 22.

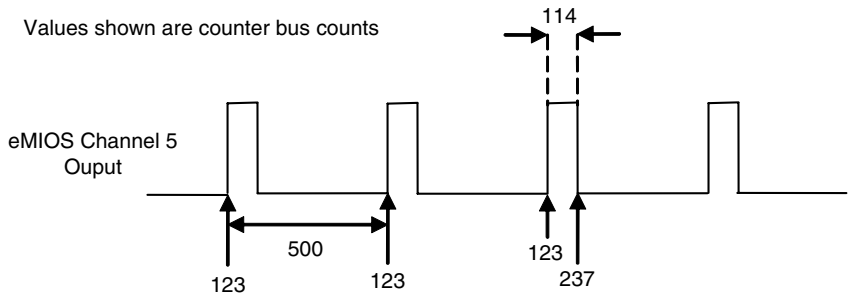


Figure 22. OPWM Input Signal for Period/Pulse Measure

3.4.5 Performing the Pulse Width and Period Measurements

To provide a time reference for channel 3, channel 15 is configured to drive a modulus up counter on counter bus A. The channel 15 prescaler is configured to be 1. This matches the values of the channel prescaler used to generate the OPWM. This allows the pulse width and period measurement results to be compared directly with the values specified for the input signal. The counter bus is configured to roll over at the maximum value of 0xFFFF. A lower value could be used, but this maximum value reduces the chance of the pulse/period occurring during the period when the counter bus value resets.

```
/* Configure Channel 15 - Modulus Up Counter - Drives Counter Bus A */

 EMIOS_CHC(15) =(EMIOS_UCPRE(0)                  /* Channel Prescaler div 1 */
                |EMIOS_UCPREN                    /* Enable ch Prescaler     */
                |EMIOS_MODE(0x10));              /* MOD UP CNT INTCLK       */


 EMIOS_CHA(15) = 0xFFFF;                         /* Set Match A Value       */
```

To perform the pulse width measurement, channel 3 is configured for pulse width measurement mode. Counter bus A is selected as the time base, by default. The channel is configured to trigger initially on the rising edge. This results in the high pulse being recorded.

```
/* Setup PF3 Pulse width measurement */

 EMIOS_CHC(3) = (EMIOS_EDPOL                     /* Rising Edge trigger     */
                |EMIOS_MODE(0x04));              /* Pulse width measure     */
```

Once configured, the flag is cleared by writing a 1 to the flag bit within the status register of channel 3.

```
EMIOS_CHS(3) = EMIOS_FLAG;                          /* Clear CH3 flag          */
```

The code then waits for the flag to be raised again to signal that the pulse width measurement has completed. The A and B registers are then read, and the pulse width is calculated. Clearing the flag and waiting for it to be asserted again, before reading the results, ensures that the latest information on the input is being collected.

```
/* Wait for flag (pulse width measurement complete)                          */
 while((EMIOS_GFLAG & EMIOS_F3)!= EMIOS_F3);


/* Calculate result of Pulse width measure */
 Pulse_Width = (EMIOS_CHA(3) - EMIOS_CHB(3));
```

The channel is then configured as a general purpose input, to reset the internal hardware of the channel before it is configured for period measure. This can be done simply by clearing the channel control register.

```
/* Set eMIOS Channel 3 into GPInput Mode BEFORE new mode */
 EMIOS_CHC(3) = 0;
```

The period measure can now be performed in the same manner as the pulse width measure. The channel is configured to measure the period between rising edges; this is done by setting the EDPOL bit.

```
/* Setup PF3 for Period measurement */
 EMIOS_CHC(3) = (EMIOS_EDPOL                    /* Rising to Rising Edge   */
                |EMIOS_MODE(0x05));             /* Pulse width measure     */


 EMIOS_CHS(3) = EMIOS_FLAG;                     /* Clear CH3 flag          */


/* Wait for flag (Period measurement complete) */
 while((EMIOS_GFLAG & EMIOS_F3)!= EMIOS_F3);


/* Calculate result of Period measure */
 Period = (EMIOS_CHA(3) - EMIOS_CHB(3));
```

When the code is run, the Period and Pulse_Width variables should return results that approximately match the values specified for the input signal. If the results are incorrect, then it may be that the matches occurred between two counter bus periods. This result calculation method does not take this into account. To make allowances for this, an if statement can be used to check that the value in register A is greater than that in

register B. If this is the case, then the counter has rolled over, and this must be taken into account when calculating the result.

As this code is time critical, it is recommended that the user use a breakpoint at the end of the example program, rather than stepping through the code.

4 Conclusion

Through the series of example configurations and overview of the eMIOS hardware in this application note, you should have a better working knowledge of the eMIOS hardware and be able to configure the eMIOS for use within an application. The example configurations covered and provided with this applications note should provide a good reference for this purpose.

For more information on the Freescale MAC7100 family, please visit www.freescale.com/mac7100.

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

AN3233
Rev. 0, 2/2006

freescale™
semiconductor