Unique Features of the 56F801x Family of Devices

Les Lewis

Contents

1. Introduction

The devices of the 56F801x family contain a number of differences and improvements that separate them from the rest of the 56800E series of devices. This application note identifies and explains these differences to help the customer get the most from the part.

2. Unified RAM

The 56F801x family of devices contains a different memory model than what is used in previous 56800E devices. In this family, the RAM architecture has been modified so it is available on both the Program and Data memory maps, creating a number of repercussions that a developer should consider.

1. RAM is a single memory block that appears in both Program and Data memory maps. Although the address is different in the two maps, it is the same physical memory.
2. The memory configuration file used by Metrowerks' CodeWarrior changes to accommodate the unified characteristics of the RAM. Dynamic (build-time) allocation of unified RAM between Program and Data spaces is shown elsewhere in application examples.

3. Clocks

Figure 3-1 shows the 56F801x's OCCS block diagram. Comparing this diagram to the equivalent diagram in the **56F8300 Peripheral User Manual** reveals several changes, most notably:

- External clock driver is supported, but not an external crystal
- The clock prescaler has been eliminated

- The PLL is essentially designed for a single frequency of operation, 192MHz
  - The PLL multiplier is fixed at 24x and is not variable, as in previous designs. With an 8MHz source clock, this results in a PLL operating frequency of 192MHz, which is divided by 6, to drive the core at 32MHz
- Only the PWM and Timer modules can be clocked at either the core clock rate or at 3x the core clock rate. The 3x clock provides a finer granularity of timing control for applications that need it.
  - The PWM clock source is selected by the PCR bit in the SIM_GPS register
  - The timer clock source is selected by the TCR bit in the SIM_GPS register
- ADC clocking options have been enhanced with the addition of standby mode

Note that there are two Postscaler blocks in **Figure 3-1**. These are both controlled by the PLLCOD field of the OCCS_DIVBY register, so they change modes together. Likewise, the ZSRC field of the OCCS_CTRL register controls two muxes such that they select the input/oscillator clock or they both select the PLL output clocks. The net effect is that HS_PERF_CLK is always 3x the SYS_CLK rate (which drives the core and the IPbus clock) for all selected core clock options.

Also note that the ADC Standby Clock is driven directly by the relaxation oscillator. If the ADC standby feature is used, the relaxation oscillator must be powered and operating. It was assumed during the design activity that the relaxation oscillator would be clocking the chip when the ADC standby feature is used.

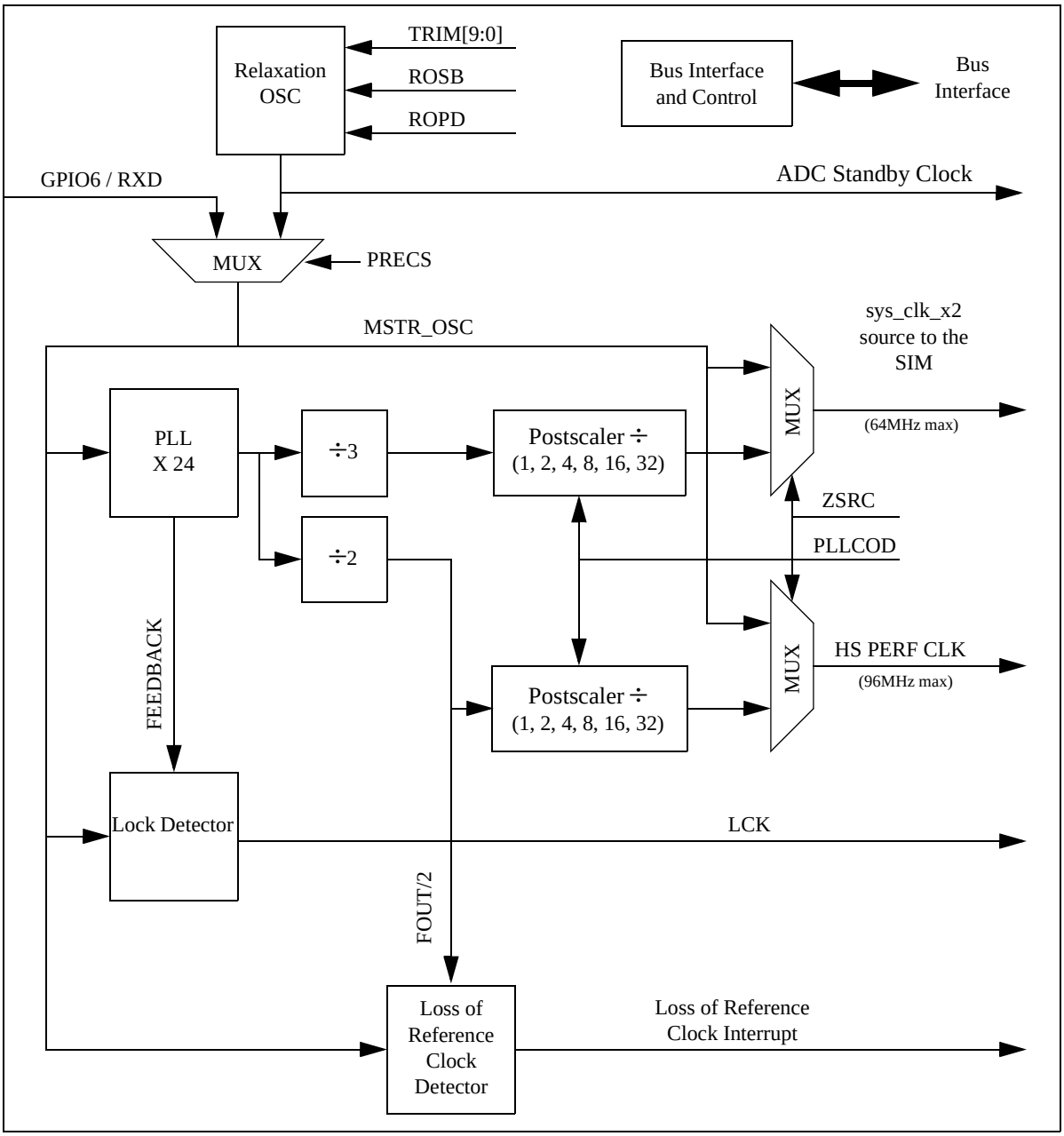

Figure 3-1. OCCS Block Diagram with Relaxation Oscillator

3.1 Internal Clock

In the **56F8000 Peripheral Reference Manual**, the description of the Relaxation Oscillator notes that the oscillator is calibrated at the factory so that the trim register can be set for correct 8MHz operation. This is true, but the calibration value cannot be maintained in the OCCS_OCTRL register when the device is powered off.

The calibration parameter is actually stored in the Flash memory and is available for application software to read from the FM_OPT1 register. The application software must copy this register trim value to the OCCS_OCTRL register if calibrated relaxation oscillator operation is to be achieved.

3.2 External Clocks

When using an external clock as the clock source of the chip, be aware of these considerations:

- The PLL is designed for 192MHz operation with a very narrow frequency range. Therefore, if the PLL is to be used, the input clock must be at 8MHz.
- If the input clock rate is other than 8MHz, then PLL operation is not possible. The chip will operate at 1/2 the input clock rate and the 3x HS_PERF_CLK operation of the timer and PWM are not available.
- When operating the chip from an external clock source, the ADC standby clock is not available unless the relaxation oscillator is also powered and operating. With this configuration, the ADC standby clock is asynchonous to the rest of the chip.

4. Muxing of Input/Output (I/O) Pins

Table 4-1 identifies which control bits must be set in order to configure the I/O function of each pin on the 56F801x devices. The default (reset) function of each pin is highlighted by shading that row in the table. The last two columns of the table identify which peripheral register bit controls whether the pin is an input or an output.

Table 4-1. External Pin Function Configuration

Pin	Selected Function	Register Settings for Selected Function		Register Settings for....	
		GPIO_PEREN	SIM_GPS	Input[1,2]	Output[1, 2]
GPIOA0	GPIO	bit 0 = 0	N/A	DDIR = 0	DDIR = 1
	PWM0	bit 0 = 1			
GPIOA1	GPIO	bit 1 = 0	N/A	DDIR = 0	DDIR = 1
	PWM1	bit 1 = 1			
GPIOA2	GPIO	bit 2 = 0	N/A	DDIR = 0	DDIR = 1
	PWM2	bit 2 = 1			
GPIOA3	GPIO	bit 3 = 0	N/A	DDIR = 0	DDIR = 1
	PWM3	bit 3 = 1			
GPIOA4	GPIO	bit 4 = 0	N/A	DDIR = 0	DDIR = 1
	PWM4	bit 4 = 1	CFG_A4 bit = 0, 1		
	FAULT1		CFG_A4 bit = 2		
	T2		CFG_A4 bit = 3	TMR2_SCTRL bit OEN = 0	TMR2_SCTRL bit OEN = 1

Table 4-1. External Pin Function Configuration (Continued)

Pin	Selected Function	Register Settings for Selected Function		Register Settings for....	
		GPIO_PEREN	SIM_GPS	Input[1,2]	Output[1, 2]
GPIOA5	GPIO	bit 5 = 0	N/A	DDIR = 0	DDIR = 1
	PWM5	bit 5 = 1	CFG_A5 bit = 0, 1		
	FAULT2		CFG_A5 bit = 2		
	T3		CFG_A5 bit = 3	TMR3_SCTRL bit OEN = 0	TMR3_SCTRL bit OEN = 1
GPIOA6	GPIO	bit 6 = 0	N/A	DDIR = 0	DDIR = 1
	FAULT0	bit 6 = 1			
GPIOA7	GPIO	bit 7 = 0	N/A	DDIR = 0	DDIR = 1
	RESET_B	bit 7 = 1			
GPIOB0	GPIO	bit 0 = 0	N/A	DDIR = 0	DDIR = 1
	SCLK	bit 0 = 1	CFG_B0 bit = 0	SPI_SCTRL bit SPMSTR = 0	SPI_SCTRL bit SPMSTR = 1
	SCL		CFG_B0 bit = 1	Bi-Directional Signal	
GPIOB1	GPIO	bit 1 = 0	N/A	DDIR = 0	DDIR = 1
	SS_B	bit 1 = 1	CFG_B1 bit = 0	SPI_SCTRL bit SPMSTR = 1	
	SDA		CFG_B1 bit = 1	Bi-Directional Signal	
GPIOB2	GPIO	bit 2 = 0	N/A	DDIR = 0	DDIR = 1
	MISO	bit 2 = 1	CFG_B2 bit = 0	SPI_SCTRL bit SPMSTR = 1	SPI_SCTRL bit SPMSTR = 0
	T2		CFG_B2 bit = 1	TMR2_SCTRL bit OEN = 0	TMR2_SCTRL bit OEN = 1
GPIOB3	GPIO	bit 3 = 0	N/A	DDIR = 0	DDIR = 1
	MOSI	bit 3 = 1	CFG_B3 bit = 0	SPI_SCTRL bit SPMSTR = 0	SPI_SCTRL bit SPMSTR = 1
	T3		CFG_B3 bit = 1	TMR3_SCTRL bit OEN = 0	TMR3_SCTRL bit OEN = 1

Table 4-1. External Pin Function Configuration (Continued)

Pin	Selected Function	Register Settings for Selected Function		Register Settings for....	
		GPIO_PEREN	SIM_GPS	Input[1,2]	Output[1, 2]
GPIOB4	GPIO	bit 4 = 0	N/A	DDIR = 0	DDIR = 1
	T0	bit 4 = 1	CFG_B4 bit = 0	TMR0_SCTRL bit OEN = 0	TMR0_SCTRL bit OEN = 1
	CLKO		CFG_B4 bit = 1		
GPIOB5	GPIO	bit 5 = 0	N/A	DDIR = 0	DDIR = 1
	T1	bit 5 = 1	CFG_B5 bit = 0	TMR1_SCTRL bit OEN = 0	TMR1_SCTRL bit OEN = 1
	FAULT3		CFG_B5 bit = 1		
GPIOB6	GPIO	bit 6 = 0	N/A	DDIR = 0	DDIR = 1
	CLKIN		N/A (PRECS bit in OCCS_CTRL = 1)	DDIR = 0	
	RXD	bit 6 = 1	CFG_B6 bit = 0		
	SDA		CFG_B6 bit = 1	Bi-Directional Signal	
GPIOB7	GPIO	bit 7 = 0	N/A	DDIR = 0	DDIR = 1
	TXD	bit 7 = 1	CFG_B7 bit = 0		
	SCL		CFG_B7 bit = 1	Bi-Directional Signal	
GPIOC0	GPIO	bit 0 = 0	N/A	DDIR = 0	DDIR = 1
	ANA0	bit 0 = 1			
GPIOC1	GPIO	bit 1 = 0	N/A	DDIR = 0	DDIR = 1
	ANA1	bit 1 = 1			
GPIOC2	GPIO	bit 2 = 0	N/A	DDIR = 0	DDIR = 1
	ANA2	bit 2 = 1	N/A **(ADC_VREF** bit 15 (SEL_VREFH) = 0)		
	VREFH		N/A **ADC_VREF** bit 15 (SEL_VREFH) = 1		
GPIOC3	GPIO	bit 3 = 0	N/A	DDIR = 0	DDIR = 1
	ANA3	bit 3 = 1			
GPIOC4	GPIO	bit 4 = 0	N/A	DDIR = 0	DDIR = 1
	ANB0	bit 4 = 1			
GPIOC5	GPIO	bit 5 = 0	N/A	DDIR = 0	DDIR = 1
	ANB1	bit 5 = 1			

Table 4-1. External Pin Function Configuration (Continued)

Pin	Selected Function	Register Settings for Selected Function		Register Settings for....	
		GPIO_PEREN	SIM_GPS	Input[1,2]	Output[1, 2]
GPIOC6	GPIO	bit 6 = 0	N/A	DDIR = 0	DDIR = 1
	ANB2	bit 6 = 1	N/A **ADC_VREF** bit 14 (SEL_VREFLO) = 0		
	VREFLO		N/A **ADC_VREF** bit 14 (SEL_VREFLO) = 1		
GPIOC7	GPIO	bit 7 = 0	N/A	DDIR = 0	DDIR = 1
	ANB3	bit 7 = 1			
GPIOD0	GPIO	bit 0= 0	N/A	DDIR = 0	DDIR = 1
	TDI	bit 0 = 1			
GPIOD1	GPIO	bit 1= 0	N/A	DDIR = 0	DDIR = 1
	TDO	bit 1 = 1			
GPIOD2	GPIO	bit 2= 0	N/A	DDIR = 0	DDIR = 1
	TCK	bit 2 = 1			
GPIOD3	GPIO	bit 3= 0	N/A	DDIR = 0	DDIR = 1
	TMS	bit 3 = 1			

1 Assumes the peripheral device clocks are enabled in the SIM_PCE register and the peripheral has been enabled.
2 Black shading indicates that the Input/Output configuration is not possible.

5. SCI with LIN Slave Mode

A new feature of the SCI is the LIN Slave mode.

The LIN mode will allow the Slave SCI to auto baud to the Master's baud rate as long as the original Slave rate is within 15% of the baud rate used by the Master. The LIN protocol is thus supported. Its use is mainly in automotive applications as a lower-cost alternative to the CAN bus.

In Asynchronous protocols, the detection of the start bit and the data bits requires an accurate match of the transmit and receive baud rates. The LIN mode allows for the slave to adjust its baud rate to be more accurate in relationship to the actual data being received by the slave.

In order to provide more reliable communication, the LIN protocol has the master device prepend messages with a leading break and synchronization character. The synchronization character is not to be confused with the ASCII SYNC character, but consists of alternating ones and zeros. The slave receiver must be set to within 15% of the correct baud rate that the master is using.

In LIN mode, the Slave receiver will wait for a break signal (spacing line of 10 bit times or more) and then look for the sync character. If it does not get a valid sync character, it will reenter the search for break state. Once a valid sync character is received following a break character, the baud rate will be adjusted for more reliable communication. Other aspects of the protocol must be implemented in software.

6. SPI Input/Output Issues

6.1 Master Mode

During the SPI peripheral's master mode operation, the GPIOB1 pin is normally configured for GPIO operation, allowing the pin to drive the $\overline{SS}$ input of another device. In a multi-slave environment, additional GPIO pins will normally be used to select the other slave SPI devices as needed.

When the GPIOB1 pin is configured as a GPIO pin, the $\overline{SS}$ input to the SPI will be driven high internally.

6.2 Slave Mode

During the SPI peripheral's slave mode operation, the GPIOB1 pin should be configured for SPI ($\overline{SS}$) operation to allow for standard slave select control.

7. ADC Changes

The 56F801x ADC has several enhancements over the 56F8300 family's design. These enhancements are documented in the **56F8000 Peripheral Reference Manual** and are summarized here:

- The two A/D converters can be triggered independently, so the converter can perform:
  - A single sequential scan of up to eight samples, using any input for any sample
  - A dual parallel scan of up to four samples, selecting any of the ANA*n* inputs for simultaneous sampling with any of the ANB*n* inputs. In this mode, the A/D converters function with identical timing.
  - A dual parallel scan where the two converters function with independent timing. See **Section 7.1** for a discussion of potential timing issues.
- Added Standby operation for an additional power saving mode
- Added auto standby operation for reduced start-up latency compared to auto power-down operation

7.1 Parallel Independent Scan Clocking

During normal ADC operation, when a START/Trigger event occurs, the ADC clock is re-timed to start on the next IPBus clock and all subsequent ADC timing is then derived from the ADC clock. When operating in non-simultaneous parallel scan mode (SIMULT = 0), this is still the case for the first converter to be activated (assuming both converters were idle).

When one of the A/D converters is already active and the second converter is then triggered, this ADC clock re-timing cannot happen. In this case, the second conversion will not actually start until the first edge of the ADC clock. Thus, there is some uncertainty concerning the exact timing the conversions, relative to the second trigger event. This uncertainty is dependent on how much slower the ADC clock is than the IPBus clock, which is determined by the DIV field of the ADC_CTRL2 register.

This uncertainty will also affect the second converter's powering up if power savings mode has been activated. The PUDELAY time will not start until the first ADC clock occurs.

7.2 Timer Triggers for ADC Sync Inputs

With the addition of the second SYNC trigger in the ADC, there are now two timer channels that can trigger ADC scan events. Timer 3 output drives SYNC0 and Timer 2 output drives SYNC1.

7.3 V_{REF} Selection

The V_{REFH} and V_{REFLO} pins are now multiplexed with two of the ADC inputs. This means that there is a choice between having two fewer inputs with greater conversion accuracy or having the full complement of inputs with reduced conversion accuracy; see **Table 7-1**. For information on conversion accuracy in either configuration, see the Data Sheet for the device being implemented.

This configuration is controlled by the SEL_VREFH and SEL_VREFLO bits in the ADC_VREF register.

Table 7-1. Input Pins

Device	Internal V_{REF}	External V_{REF}
8013	6	4
8014	8	6

8. Timer Input/Output

The Quad Timer's documentation indicates that there are four external inputs that the timer can access. The System Integration Module (SIM) and GPIO configuration control how (when) these inputs are actually connected to external signals. As an example, **Figure 8-1** shows the various muxes and control associated with Counter #3 input control, which is the most complicated of all of the input controls. **Table 8-1** through **Table 8-3** show the specific controls used for each of the four inputs to select what signal is actually routed to the timer inputs.

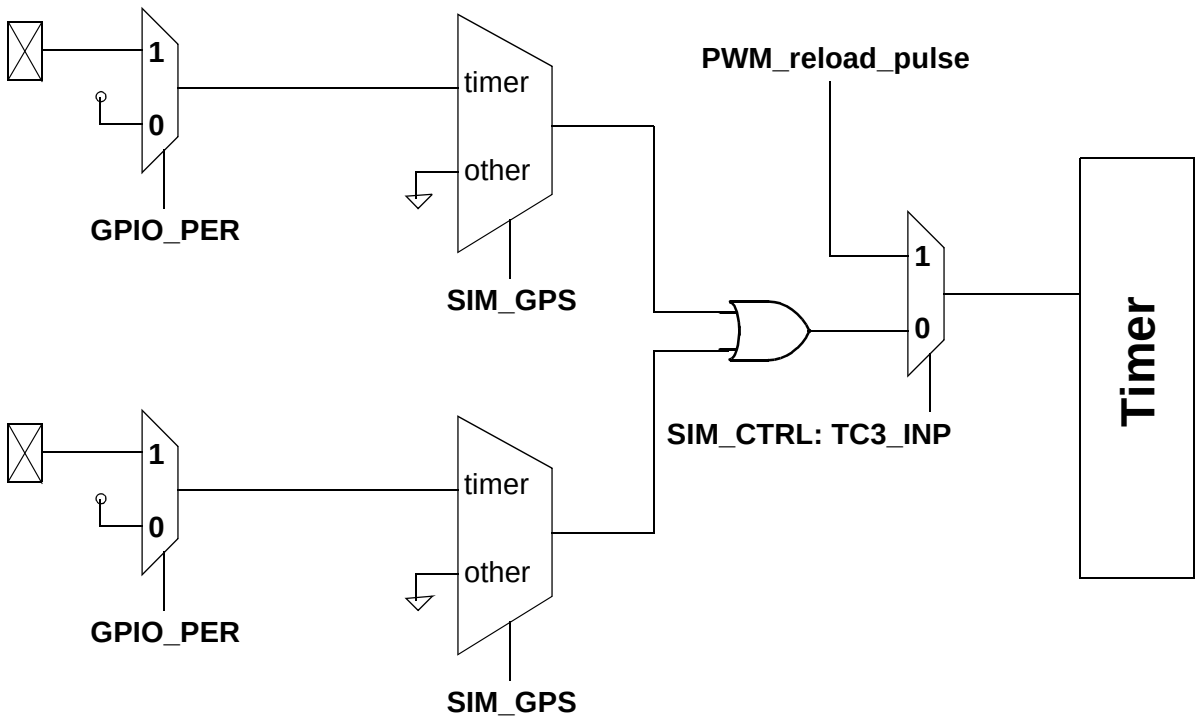

Figure 8-1. Timer Counter #3 Input Control Muxes

Table 8-1. Timer Inputs 0 and 1

Timer Input (TMR_CTRL: PCS or SCS)	GPIO_PER Control Bit	SIM_GPS	Actual Timer Input
#0	B4 = 0	CFG_B4 (selects T0) = 0	1
		CFG_B4 (selects CLKO) = 1	0
	B4 = 1	CFG_B4 (selects T0) = 0	T0_in (from pad)
		CFG_B4 (selects CLKO) = 1	0
#1	B4 = 0	CFG_B5 (selects T1) = 0	1
		CFG_B5 (selects FAULT3) = 1	0
	B4 = 1	CFG_B5 (selects T1) = 0	T1_out (to pad)
		CFG_B5 (selects FAULT3) = 1	0

Table 8-2. Timer Inputs 2

Timer Input (TMR_CTRL selection)	GPIOA_PEREN (Bit 4)	GPIOB_PEREN (Bit 2)	SIM_GPS CFG_A4	SIM_GPS CFG_B2	Actual Timer Input
#2	0	0	(T2) 3	(T2) 1	1
				(MISO) 0	1
			(Other)	(T2) 1	1
				(MISO) 0	0
		1	(T2) 3	(T2) 1	1
				(MISO) 0	1
			(Other)	(T2) 1	T2_in (from GPIOB2 pad)
				(MISO) 0	0
	1	0	(T2) 3	(T2) 1	1
				(MISO) 0	T2_out (to GPIOA4 pad)
			(Other)	(T2) 1	1
				(MISO) 0	0
		1	(T2) 3	(T2) 1	T2_in (from GPIOA4 pad) ORed with T2_in (from GPIOB2 pad)
				(MISO) 0	T2_in (from GPIOA4 pad)
			(Other)	(T2) 1	T2_in (from GPIOB2 pad)
				(MISO) 0	0

Table 8-3. Timer Inputs 3

Timer Input (TMR_CTRL selection)	TC3_INP in SIM_CTRL	GPIOA_PEREN (Bit 4)	GPIOB_PEREN (Bit 2)	SIM_GPS CFG_A5	SIM_GPS CFG_B3	Actual Timer Input
#3	0	0	0	(T3) 3	(T3) 1	1
					(MOSI) 0	1
				(Other)	(T3) 1	1
					(MOSI) 0	0
			1	(T3) 3	(T3) 1	1
					(MOSI) 0	1
				(Other) 3	(T3) 1	T3_in (from GPIOB3 pad)
					(MOSI) 0	0
		1	0	(T3) 3	(T3) 1	1
					(MOSI) 0	T3_out (to GPIOA5 pad)
				(Other)	(T3) 1	1
					(MOSI) 0	0
			1	(T3) 3	(T3) 1	T3_in (from GPIOA5 pad) ORed with T3_in (from GPIOB5 pad)
					(MOSI) 0	T3_in (from GPIOA5 pad)
				(Other)	(T3) 1	T3_in (from GPIOB5 pad)
					(MOSI) 0	0
	1	N/A	N/A	N/A	N/A	PWM reload_sync signal

9. PWM Control

The block diagram for the 56F801x devices is shown in **Figure 9-1**. In this figure, it is evident that several different PWM control options are available and the user must make choices concerning PWM pin functions. From the figure, the choice of I/O functions (PWMn outputs vs. FAULTn inputs) should be fairly obvious. (See **Table 4-1**.)The selection of how the PWM is controlled is not so obvious and will be the topic of the remainder of this section. For each operating mode discussed, the block diagram will be repeated and the appropriate control path will be highlighted to aid in understanding. The highlighted orange and magenta lines in subsequent figures identify the inputs and outputs that are active for the mode of operation being discussed.

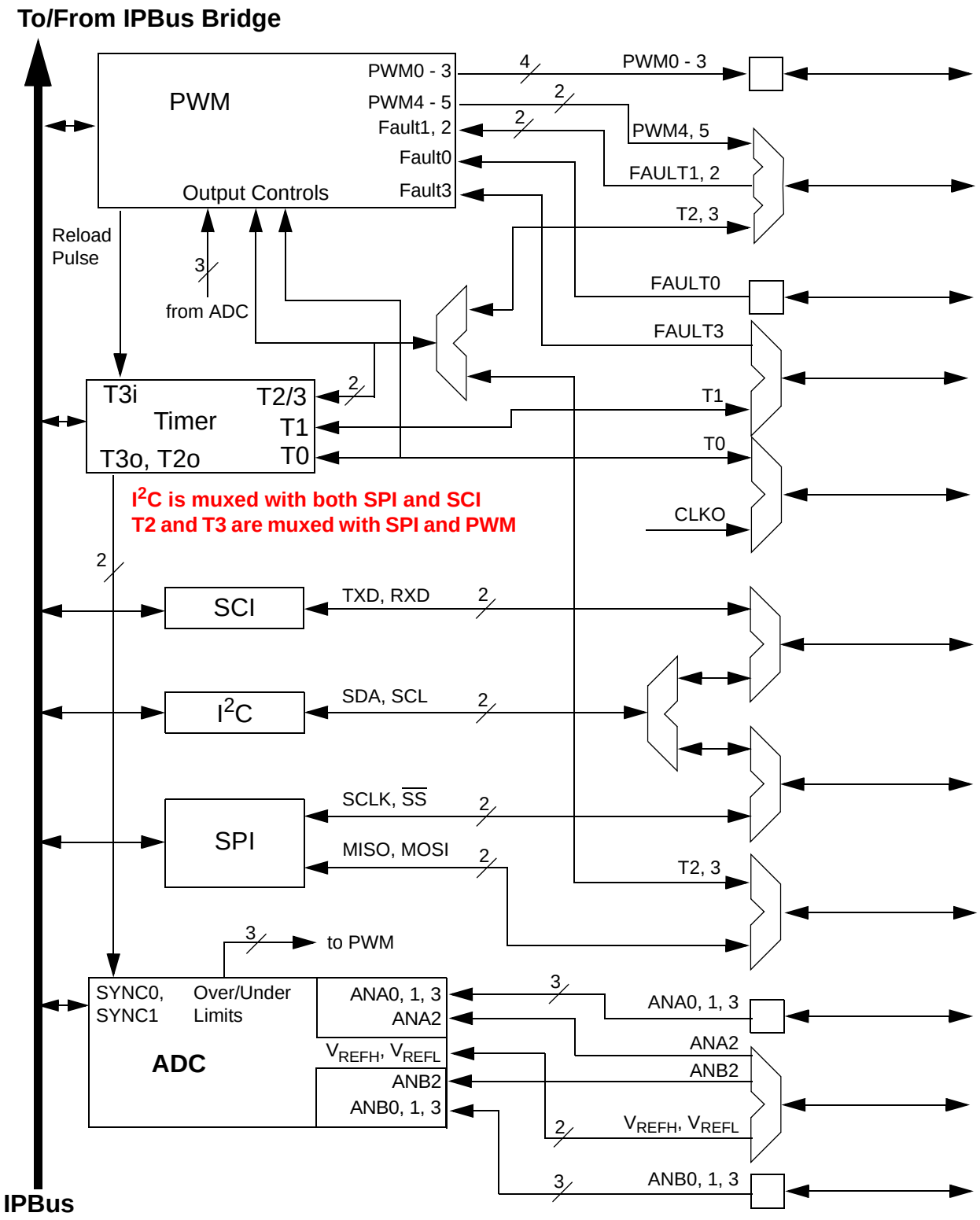

Figure 9-1. Muxing of Peripheral I/O Pin-out

The first change to note in the PWM module is that two new registers were added, as shown in **Table 9-1**.

In previous designs, there was only a single control for dead time insertion. Starting with the 56F801x devices, this control has been extended to allow for independent timing control of rising-edge dead time and falling-edge dead time. Thus, the old PMDEADTM register is now two registers; PWM_DTIM0 and PWM_DTIM1.

The PMSRC register was added to increase the flexibility of the peripheral and allow several new ways to control the PWM.

Table 9-1. PWM Registers - 56F801x and Possible Future Devices

Address (PWM_BASE +)		Register Name	Description
New Offset	Legacy Offset		
$14	New	SCTRL	PWM Source Control Register
$13	$12	ICCTRL	PWM Internal Correction Control Register
$12	$11	PORT	PWM Port Register
$11	$10	CCTRL	PWM Channel Control Register
$10	$0F	CNFG	PWM Configuration Register
$0F	$0E	DMAP2	PWM Disable Mapping Register Two
$0E	$0D	DMAP1	PWM Disable Mapping Register One
$0D	Redefined	DTIM1	PWM Dead Time Register 1
$0C	$0C	DTIM0	PWM Dead Time Register 0
$0B	$0B	VAL5	PWM Value Register 5
$0A	$0A	VAL4	PWM Value Register 4
$09	$09	VAL3	PWM Value Register 3
$08	$08	VAL2	PWM Value Register 2
$07	$07	VAL1	PWM Value Register 1
$06	$06	VAL0	PWM Value Register 0
$05	$05	CMOD	PWM Counter Modulo Register
$04	$04	CNTR	PWM Counter Register
$03	$03	OUT	PWM Output Control Register
$02	$02	FLTACK	PWM Fault Status Acknowledge
$01	$01	FCTRL	PWM Fault Control Register
$00	$00	CTRL	PWM Control Register

9.1 Traditional PWM Operation

The highlighted orange and magenta lines in **Figure 9-2** identify the inputs and outputs that are normally associated with PWM operation. The PWM internal controls are used to control the timing of the PMW output signals (PWM0 - 3 and PWM4 - 6). The PWM fault inputs (FAULT0 - 3) provide fault disabling of the PWM outputs.

Note that two of the outputs (PWM4, 5) are shared with the fault inputs (FAULT1, 2), so a choice must be made concerning the function of these two pins.

In **Figure 9-2**, note that the PWM load interrupt (SYNC output) signal is used as an input to Timer 3. The timer can then be used to introduce a controllable delay before generating its output signal. The timer output then triggers the ADC, with the SYNC0 ADC input. For more information on this interaction, please see **Table 8-3** and the **56F8000 Peripheral Reference Manual** for clarification of all three of these peripherals.

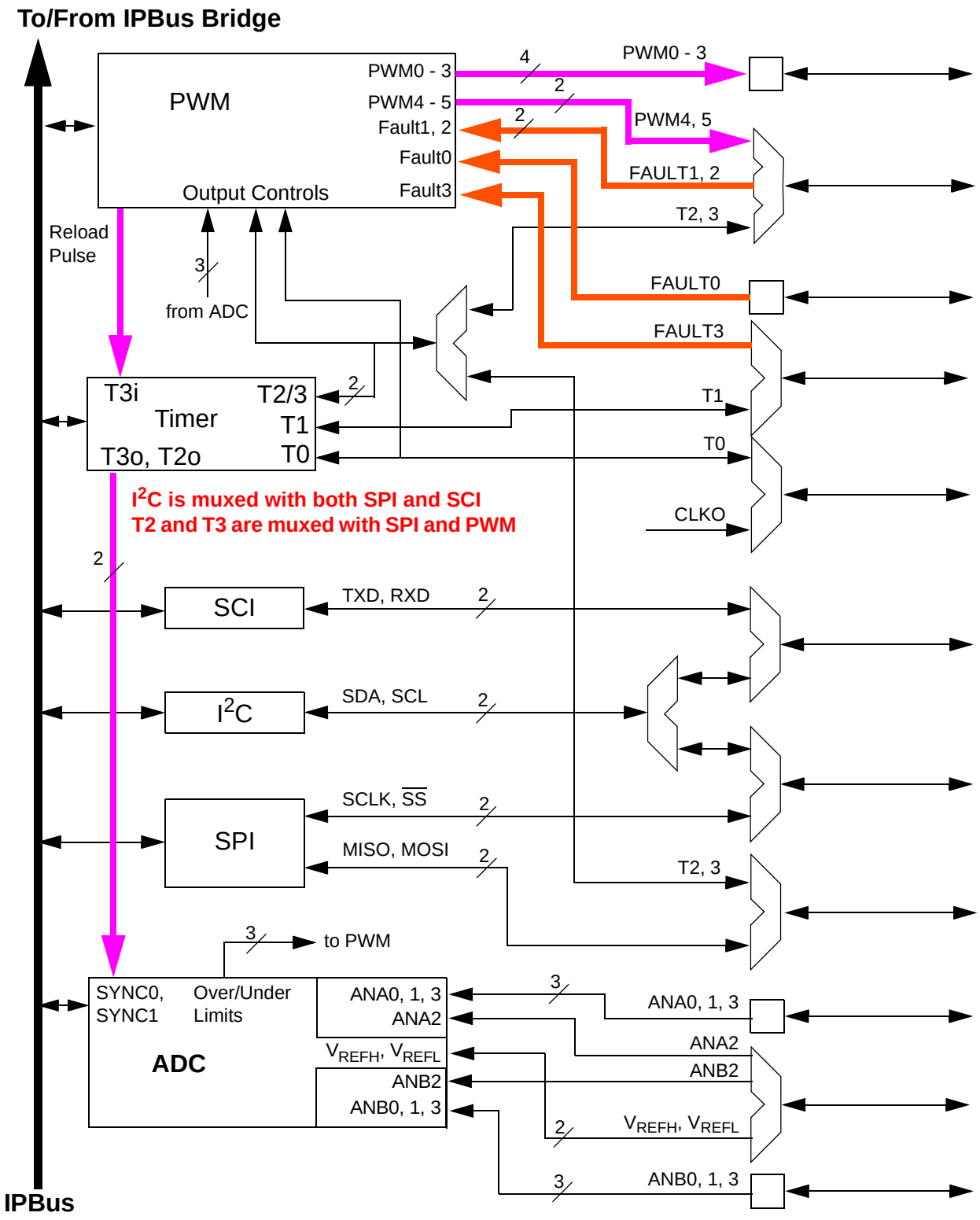

Figure 9-2. Traditional PWM Operation

Table 9-2. Setting Up for Traditional PWM Control to Drive ADC Measurements

Peripheral / Register Configuration	Initialization Steps
GPIO	The default function of all of the I/O pins is GPIO, so all desired PWM signals must have the GPIOA_PEREN register configured to select peripheral control of the pin. Before this is done, the SIM_GPS register should be configured to select the desired PWM function for the GPIOA4 and GPIOA5 pins.
SIM_PCE	Enables clocks to the ADC, TMR and PWM peripherals.
TC3_INP in SIM_CTRL	TC3_INP must be set to 1 to enable the PWM_RELOAD_PULSE signal to drive the T3 input.
Timer 3	The Quad Timer is typically configured for Triggered-Count mode. 1. Configure Timer 3 (or Timer 2, depending on the ADC SYNC to be driven) so that one of the IP_bus clocks is used as the primary count source and the T3 input is used as the secondary input 2. Set the output mode for “Set OFLAG on compare, clear on secondary source input edge.” This output signal will drive the SYNC0 (1) input of the ADC. 3. TMR_CTRL: Length = 1; OnCE = 1. 4. Set the compare registers to provide the appropriate delay for your application
ADC	Assuming SYNC0 is used to trigger the ADC: 1. Set STOP0 in the ADC_CTRL1 register 2. Configure SMODE[2:0] for Triggered Sequential Mode (Triggered Parallel mode may also be used) 3. Set the DIV and other control fields in ADC_CTRL2 as appropriate for your application 4. Set the ADC_CLIST1 register so that the appropriate inputs are sampled during the ADC scan 5. Set the ADC_SDIS register 6. Set the ADC_OFFST*n* registers as appropriate for your application 7. Enable power savings features of the ADC, if desired 8. Set STOP0 = 0
PWM	1. Configure the PWM for normal control of your application. This will depend a great deal on the application you are implementing and so is not discussed in detail here. 2. Leave the PWM_SCTRL register in its reset configuration. 3. Enable the PWM to start the system operating

9.2 ADC Control of PWM Outputs

ADC inputs can be used to control the complementary PWM outputs, as shown in **Figure 9-3**. The actual control signals within the chip block diagram are shown in **Figure 9-5**. When operating in this mode, the ADC continuously monitors the selected input(s). The converted digital representation of the input is then compared against the high/low limit register values and control signals are sent to the PWM module to drive its outputs. The ADC can generate interrupts when the threshold limits are reached, allowing the user software to be aware of the control taking place.

Please note:

- ADC's sample list determines which ADC input is used for each PWM pair:
  - Sample 0 -> PWM 0 - 1
  - Sample 1 -> PWM 2 - 3
  - Sample 2 -> PWM 4 - 5
- If V_{IN} > High Limit, PWM is turned off
  If V_{IN} < Low Limit, PWM is turned on

This logic is independent of the sticky bits in the ADC's ADLSTAT register.

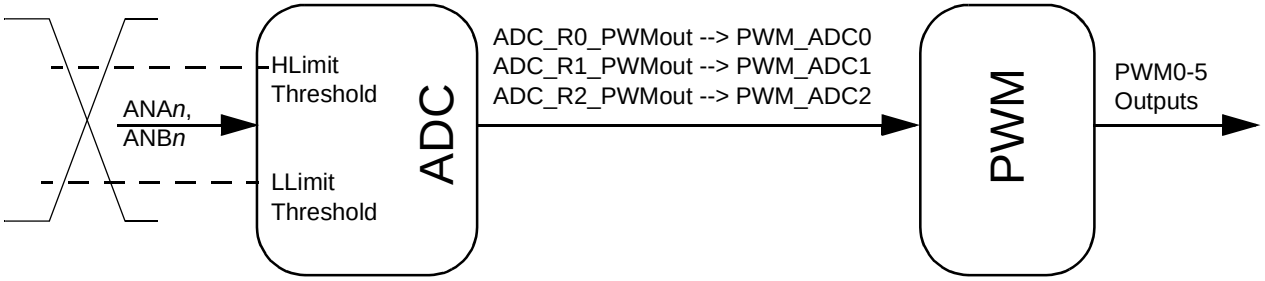

Figure 9-3. ADC Inputs Driving PWM Outlets

This mode of operation is entered when OUTCTL*n* bits of the PMOUT are 1 and the PMSRC register is programmed so the SRC0 - 2 fields contain 01 (shown in **Figure 9-4**). With the register setting shown, PWM0/1 will be controlled by the ADC input selected for ADC_RSLT0, while the input selected for ADC_RSLT1 controls PWM2/3 and the input selected for ADC_RSLT2 values control PWM4/5 outputs.

PWM_BASE+$14	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	CINV 5	CINV 4	CINV 3	CINV 2	CINV 1	CINV 0	SRC2			SRC1			SRC0	
Write																
Contents	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1

Figure 9-4. PWM Source Control Register (SCTRL)

When properly configured, the user has complete control over the signal level trigger point and the applied hysteresis in the control signal used to drive the PWM outputs.

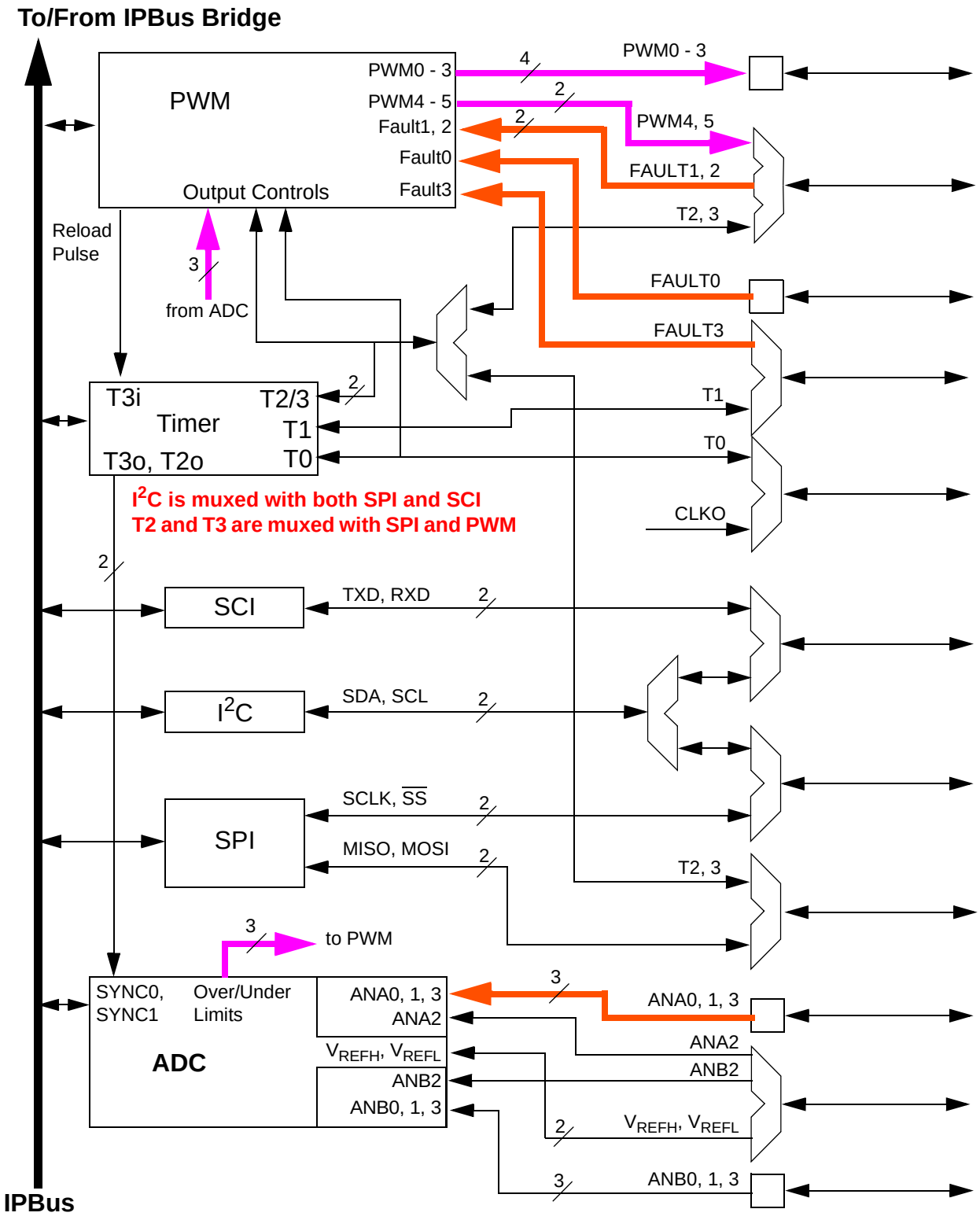

Figure 9-5. ADC Control of PWM Outputs

Table 9-3. Setting Up for ADC Control of PWM Outputs[1]

Peripheral / Register Configuration	Initialization Steps
GPIO	The default function of all I/O pins is GPIO, so all desired PWM signals must have the GPIOA_PEREN register configured to select peripheral control of the pin. Before this is done, the SIM_GPS register should be configured to select the desired PWM function for the GPIOA4 and GPIOA5 pins.
SIM_PCE	Enables clocks to the ADC and PWM peripherals
ADC	1. Set STOP0 in the ADC_CTRL1 register 2. Enable Low Limit and High Limit interrupts, if desired, in the ADC_CTRL1 register. Configure the inputs for single-ended or differential conversations, as required. Configure SMODE[2:0] for Loop Sequential mode (Loop Parallel mode may also be used). 3. Set the DIV and other control fields in ADC_CTRL2 as appropriate for your application 4. Set the ADC_CLIST1 register so that SAMPLE0, SAMPLE1 and SAMPLE2 obtain samples from the desired control input signals. The selected input samples will be stored in the ADC_RSLT0, ADC_RSLT1, and ADC_RSLT2 registers, respectively. The other SAMPLE*n* fields may also be set if additional inputs are to be measured. 5. Set the ADC_SDIS register to allow at least three samples to be taken during each scan. A value of 0x00F8 can be used. 6. Set the ADC_LOLIM0 - 2 registers for your lower threshold value. Set the ADC_HILIM0 - 2 registers for your upper threshold limit value. 7. Set the ADC_OFFST0 - 2 registers as appropriate for your application (assumed to be 0x0000)
PWM	1. Configure the PWM for normal complementary operation. This will depend a great deal on the application you are implementing and so is not discussed in detail here. 2. Set the appropriate OUTCTLn bits in the PWM_OUT register to 1 3. Set the PWM_SCTRL register as shown in **Figure 9-4**.
Final Step	Set STOP0 = 0 and then set START0 = 1 in the ADC_CTRL1 register to start the ADC operation

1 This example assumes you want individual control of each pair of PWM outputs.

9.3 Timer Drives ADC Samples for PWM Control

Section 9.2 shows ADC control of the PWM outputs, with the ADC operating at its maximum sampling rate. If the ADC sampling does not need to operate at full rate, a timer can be used to provide periodic sampling at any user-defined rate. This is illustrated in **Figure 9-6** and **Figure 9-7**.

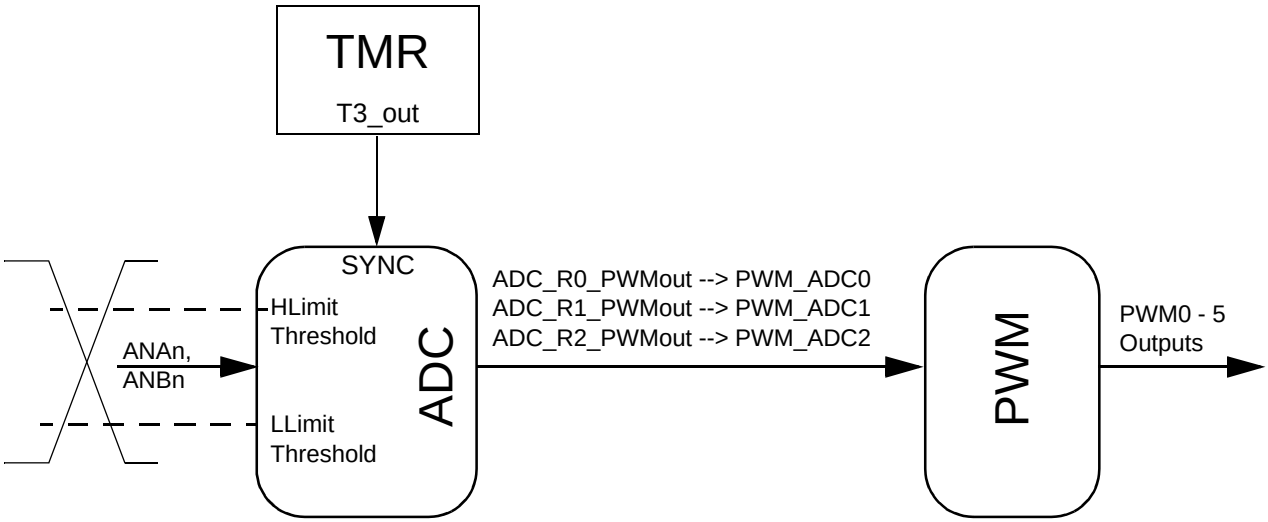

Figure 9-6. ADC Inputs Driving PWM Outputs

Table 9-4. Setting Up for Timer Driven, ADC Control of PWM Outputs[1]

Peripheral / Register Configuration	Initialization Steps
GPIO	The default function of all I/O pins is GPIO, so all desired PWM signals must have the GPIOA_PEREN register configured to select peripheral control of the pin. Before this is done, the SIM_GPS register should be configured to select the desired PWM function for the GPIOA4 and GPIOA5 pins.
SIM_PCE	Enable clocks to the TMR, ADC and PWM peripherals
ADC	Configure the ADC as indicated in **Table 9-3**, except use Triggered Sequential mode instead of Loop Sequential mode
PWM	1. Configure the PWM for normal complementary operation 2. Set the appropriate OUTCTL*n* bits in the PWM_OUT register to 1 3. Set the PWM_SCTRL register as shown in **Figure 9-4** 4. Set additional registers to cover swap, mask, dead time and fault settings as appropriate for your application
Timer	Configure Timer 3 for Count mode, such that a rising edge of OFLAG occurs at the desired rate Once the timer is started, the system should begin operating

1 This example assumes you want individual control of each pair of PWM outputs.

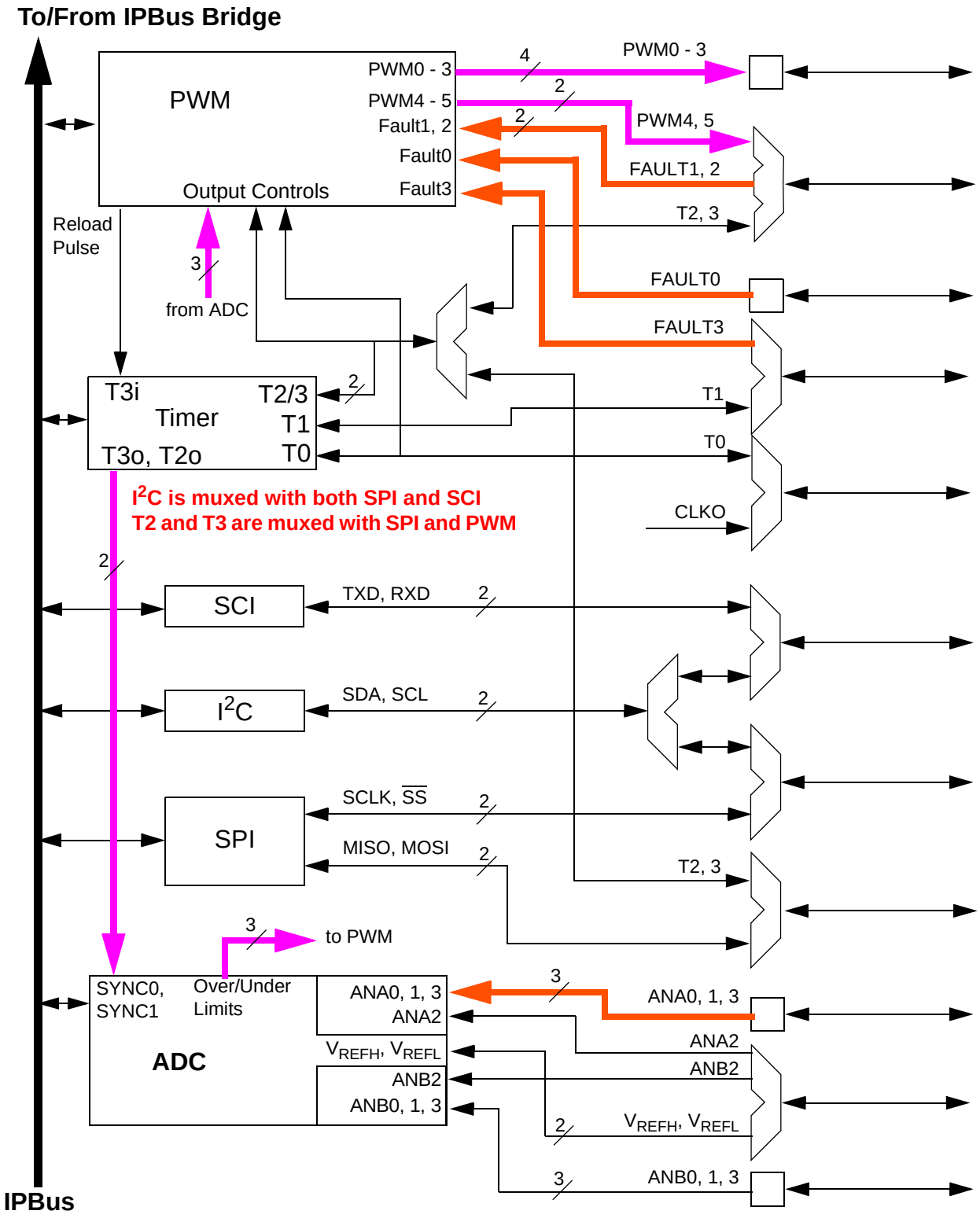

Figure 9-7. Timer Drives ADC Samples for ADC Control of PWM Outputs

9.4 Timer Control of PWM Outputs

Traditional PWM operation uses a single register (PWM_CMOD) to control the period of the control signals for each of the PWM outputs. Once the period is set, the PWM duty cycle is set by writing to the appropriate VALUE register (PWM_VAL0 - 5). This works very well for 3-phase motor control, but is somewhat restrictive in other applications.

It is now possible to use the PWM timers to control one (or more) complimentary pair(s) of outputs while using a timer from the Quad Timer to control other PWM output pairs. This example will discuss using the Quad Timer to control the complimentary pairs of PWM outputs without using the PWM counters. **Figure 9-8** shows the timer driving the PWM module.

Table 9-5 identifies which timer channels can drive the various PWM outputs. Note that timer #1 is not used directly for motor control functions. **Table 9-6** shows the steps involved in setting up this control.

Table 9-5. Timer Channels to PWM Complimentary Pair Control

Timer Output	PWM Channel
#0	Drives PWM0 / PWM1 pair This timer can also be used to control PWM2/PWM3 and PWM4/PWM5 outputs
#2	Drives PWM2 / PWM3
#3	Drives PWM4 / PWM5

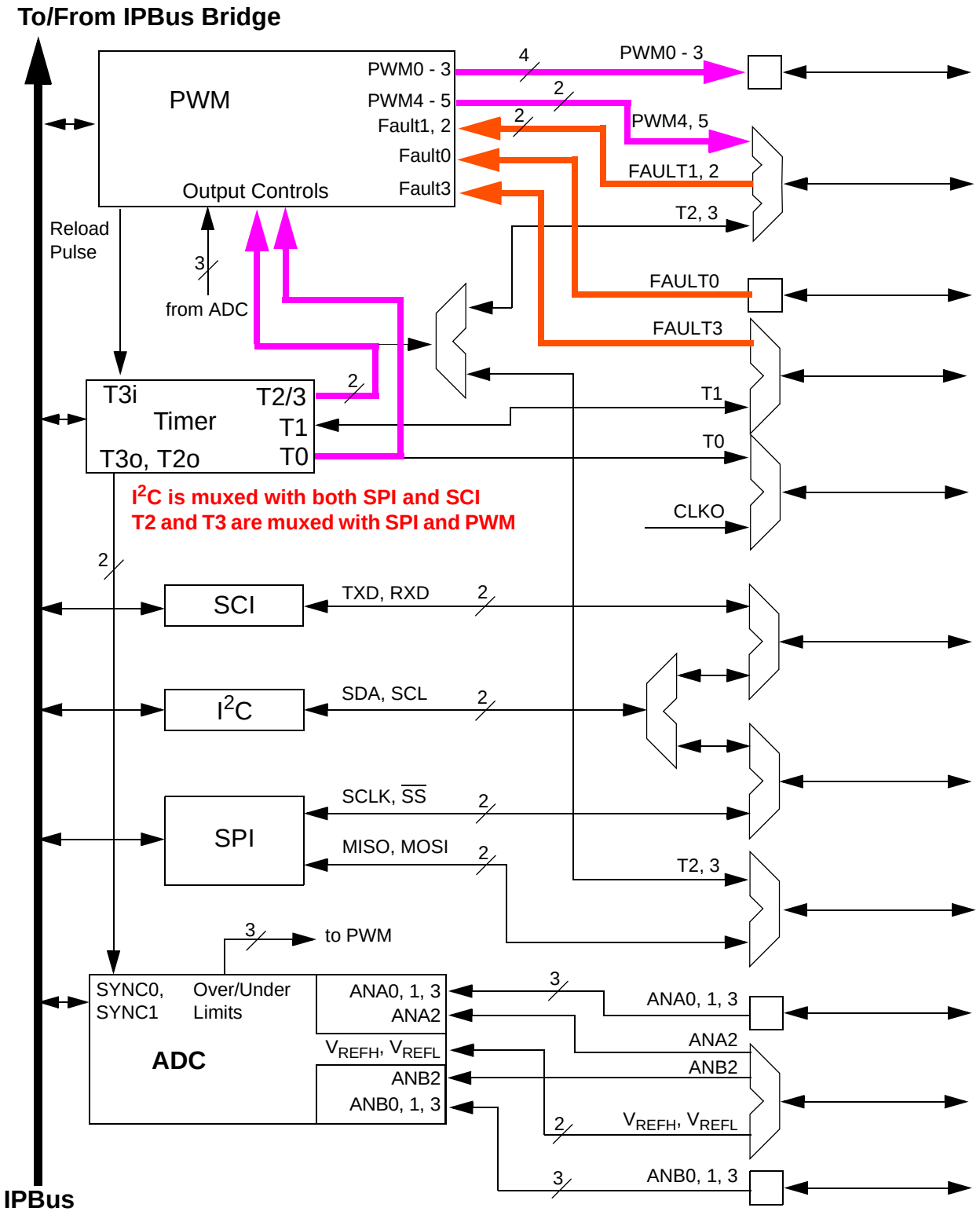

Figure 9-8. Timer Control of PWM Outputs

Table 9-6. Setting Up for Timer Control of PWM Outputs

Peripheral / Register Configuration	Initialization Steps
GPIO	The default function of all I/O pins is GPIO, so all desired PWM signals must have the GPIOA_PEREN register configured to select peripheral control of the pin. Before this is done, the SIM_GPS register should be configured to select the desired PWM function for the GPIOA4 and GPIOA5 pins.
SIM_PCE	Enable clocks to the TMR and PWM peripherals
PWM	1. Configure the PWM for normal complementary operation 2. Set the PWM_SCTRL register as shown in **Figure 9-9** if all three PWM output pairs are to be controlled independently. **Figure 9-10** shows the PWM_SCTRL register setting if all PWM output pairs are to be controlled by Timer #0. Other combinations are also possible. 3. Set additional registers to cover swap, mask, deadtime and fault settings as appropriate for your application
Timer	Configure timers for either fixed-frequency or variable-frequency PWM mode. Variable-frequency mode is more flexible and, therefore, may be a better choice. See the Quad Timer section in the **56F8000 Peripheral Reference Manual** to learn more about how to set this up.

PWM_BASE+$14	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	CINV5	CINV4	CINV3	CINV2	CINV1	CINV0	SRC2			SRC1			SRC0	
Write																
Contents	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1

Figure 9-9. PWM Source Control Register (SCTRL)
Timers T0, T2 and T3 Control PWM Outputs

PWM_BASE+$14	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	CINV5	CINV4	CINV3	CINV2	CINV1	CINV0	SRC2			SRC1			SRC0	
Write																
Contents	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	1

Figure 9-10. PWM Source Control Register (SCTRL)
Timer T0 Controls All PWM Outputs

9.5 External Pin Control of PWM Outputs

For some applications, it is desirable to provide PWM output controls based on an external control. The 56F801x devices provide this flexibility. When configured for external control, the PWM outputs can still provide dead time insertion with the swap and mask features of the PWM module.

Figure 9-12 shows the active control signals in green. Note that these signals are taken from the timer inputs and therefore they go through the same set of muxes. See **Table 8-1** through **Table 8-3** for details of how to enable these signals for access to the PWM control inputs.

It is not possible to have these controls interrupt the processor, except indirectly via timer interrupts (see **Section 9.6**).

Table 9-7. Setting Up for External Control of PWM Outputs

Peripheral / Register Configuration	Initialization Steps
GPIO	The default function of all I/O pins is GPIO, so all desired PWM signals must have the GPIOA_PEREN register configured to select peripheral control of the pin. Before this is done, the SIM_GPS register should be configured to select the desired PWM function for the GPIOA4 and GPIOA5 pins. The GPIO pins that are intended for PWM control must be configured for GPIO control and not for peripheral control (the PEREN register bit must be set to 0). Note that the DDIR control bit must be set to 0 to configure the pin as an input.
SIM_PCE	Enable clocks to the PWM peripheral
PWM	1. Configure the PWM for normal complementary operation 2. Set the PWM_SCTRL register as shown in **Figure 9-11** if all three PWM output pairs are to be controlled independently. Other combinations are also possible. 3. Set additional registers to cover swap, mask, deadtime and fault settings as appropriate for your application

PWM_BASE+$14	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0	0	CINV 5	CINV 4	CINV 3	CINV 2	CINV 1	CINV 0	SRC2			SRC1			SRC0	
Write																
Contents	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	0

Figure 9-11. PWM Source Control Register (SCTRL) External Pins Control PWM Outputs

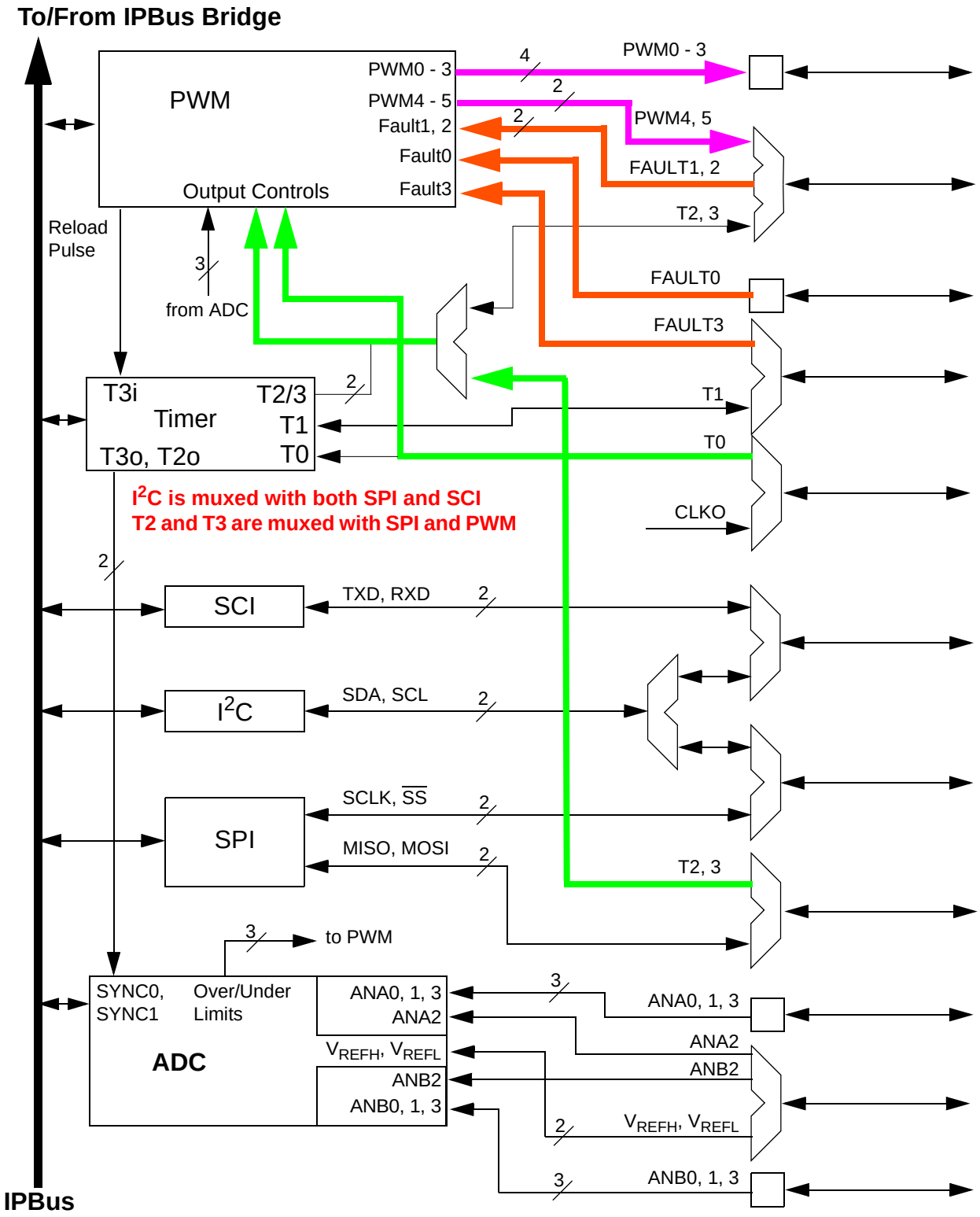

Figure 9-12. External Pin Control of PWM Outputs

9.6 External Pin Control of PWM Outputs with ADC Sample Triggering

The example of **Section 9.5** can be extended to provide processor interrupts after an arbitrary delay. This delayed interrupt can also be used to trigger an ADC sampling scan. This is shown in **Figure 9-13**, but will not be expanded upon here. The previous example explains how to set this up, if desired.

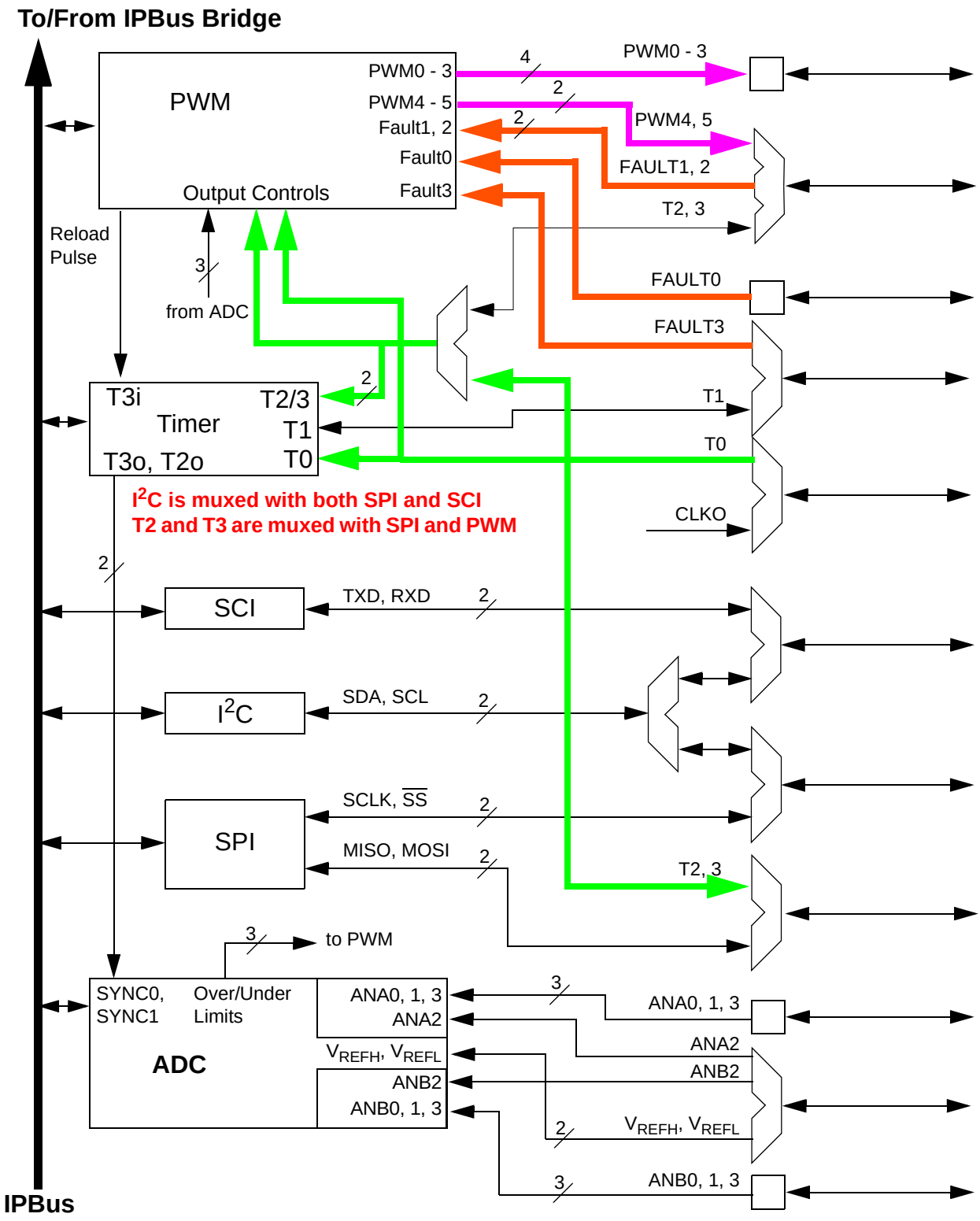

Figure 9-13. External Pin Control of PWM Outputs Trigger-Delayed ADC Scan

10. Power Savings Features

There are a number of power savings features that have been incorporated into the 56F801x family of devices for power-conscious applications. This will not be extensively covered in this application note, since that information may be found in AN3104. Instead, the following list offers a high-level view of the incorporated features for further reader study. The reader should review the indicated sections of the Data Sheet for the specific device being implemented for additional information.

- The clocks to each peripheral can be turned off, if the peripheral is not needed; see the SIM_PCE register
- The OCCS can be reconfigured as needed to provide lower frequency clocking options
- The internal relaxation oscillator can operate at a standby frequency of 400Kz instead of 8MHz
- In some circumstances, the on-chip regulator can be placed in a reduced power mode
- The ADC can operate in one of five power management modes

11. References

1. *56F8300 Peripheral User Manual*,MC56F8300UM
2. *56F8000 Peripheral Reference Manual*,MC56F8000RM

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

AN3102

Rev. 0
9/2005