



Peripheral Trigger Generator (PTG)

HIGHLIGHTS

This section of the manual contains the following major topics:

1.0	Introduction	2
2.0	Register Map.....	4
3.0	Step Commands and Format	14
4.0	Module Operation	19
5.0	Application Examples.....	32
6.0	Power-Saving Modes.....	39
7.0	Related Application Notes.....	40
8.0	Revision History	41

dsPIC33/PIC24 Family Reference Manual

Note: This family reference manual section is meant to serve as a complement to device data sheets. Depending on the device variant, this manual section may not apply to all dsPIC33/PIC24 devices.

Please consult the note at the beginning of the “**Peripheral Trigger Generator (PTG)**” chapter in the current device data sheet to check whether this document supports the device you are using.

Device data sheets and family reference manual sections are available for download from the Microchip Worldwide Web site at: <http://www.microchip.com>.

1.0 INTRODUCTION

The Peripheral Trigger Generator (PTG) module is a user-programmable sequencer, which is capable of generating complex trigger signal sequences to coordinate the operation of other peripherals.

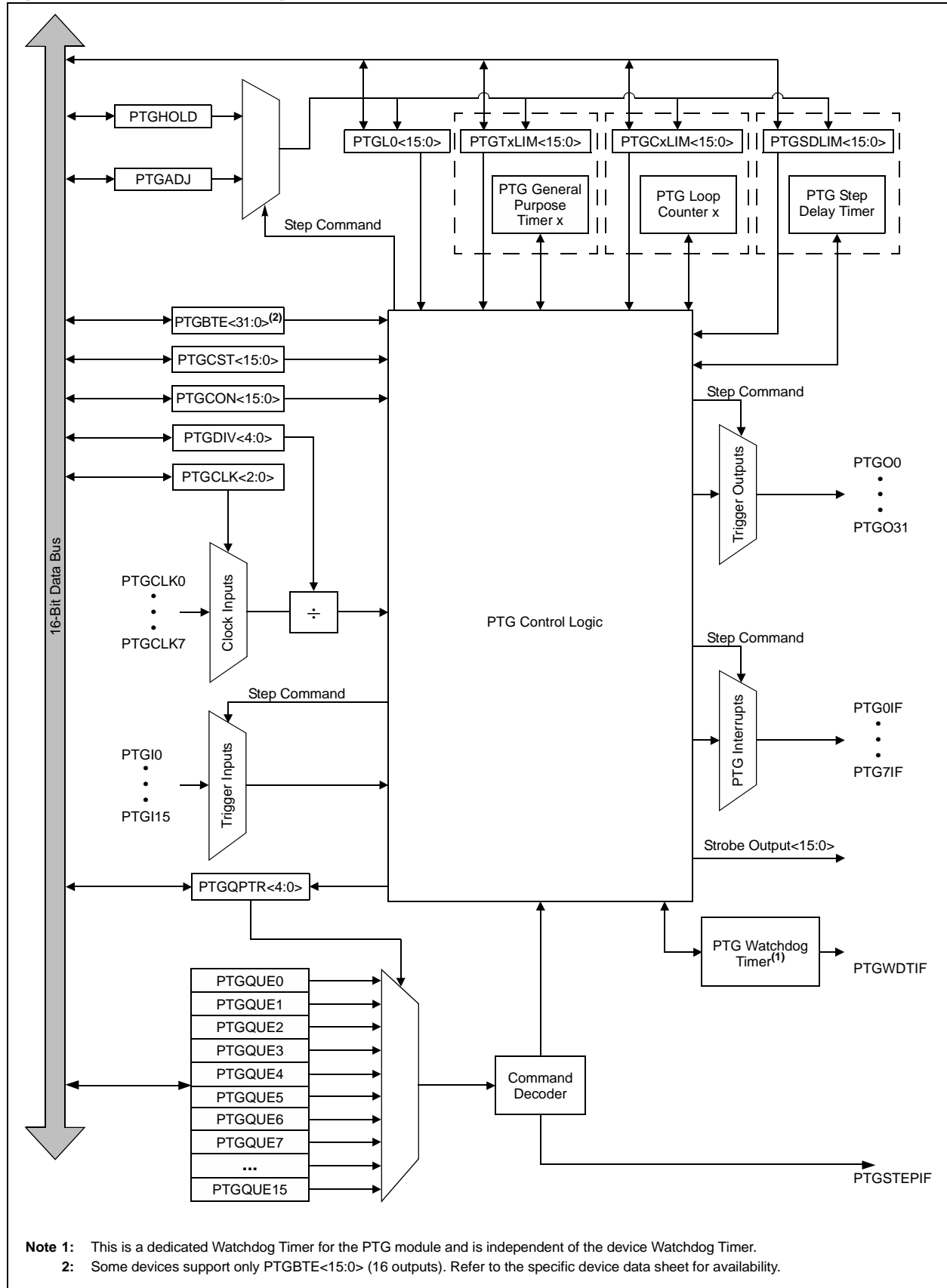
The PTG module is designed to interface with other modules, such as Analog-to-Digital Converter (ADC), output compare and PWM modules, timers and interrupt controllers.

1.1 Features

- Behavior is Step Command-Driven:
 - Step commands are 8 bits wide
- Commands are Stored in a Step Queue:
 - Queue depth is up to 32 entries
 - Programmable Step execution time (Step delay)
- Supports the Command Sequence Loop:
 - Can be nested one-level deep
 - Conditional or unconditional loop
 - Two 16-bit loop counters
- 16 Hardware Input Triggers:
 - Sensitive to either positive or negative edges, or a high or low level
- One Software Input Trigger
- Generates up to 32 Unique Output Trigger Signals
- Generates Two Types of Trigger Outputs:
 - Individual
 - Broadcast
- Strobed Output Port for Literal Data Values:
 - 5-bit literal write (literal part of a command)
 - 16-bit literal write (literal held in the PTGL0 register)
- Generates up to 10 Unique Interrupt Signals
- Two 16-Bit General Purpose Timers
- Flexible Self-Contained Watchdog Timer (WDT) to Set an Upper Limit to Trigger Wait Time
- Single Step Command Capability in Debug mode
- Selectable Clock (System, Pulse-Width Modulator (PWM) or ADC)
- Programmable Clock Divider

Peripheral Trigger Generator (PTG)

Figure 1-1: PTG Block Diagram



2.0 REGISTER MAP

Table 2-1: PTG Register Map

File Name	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PTGCST	PTGEN	r	PTGSIDL	PTGTOGL	—	PTGSWT	PTGSSEN	PTGIVIS	PTGSTRT	PTGWDTO	PTGBUSY	—	—	—	PTGITM<1:0>	
PTGCON	PTGCLK<2:0>			PTGDIV<4:0>					PTGPWD<3:0>				—	PTGWDT<2:0>		
PTGBTE	PTGBTE<15:0>															
PTGBTEH ⁽¹⁾	PTGBTE<31:16>															
PTGHOLD	PTGHOLD<15:0>															
PTGTOLIM	PTGTOLIM<15:0>															
PTGT1LIM	PTGT1LIM<15:0>															
PTGSDLIM	PTGSDLIM<15:0>															
PTGC0LIM	PTGC0LIM<15:0>															
PTGC1LIM	PTGC1LIM<15:0>															
PTGADJ	PTGADJ<15:0>															
PTGL0	PTGL0<15:0>															
PTGQPTR	—	—	—	—	—	—	—	—	—	—	—	PTGQPTR<4:0>				
PTGQUE _n ⁽²⁾	STEP2n+1<7:0> ^(3,4)								STEP2n<7:0> ^(3,4)							

Legend: — = unimplemented, read as '0'; r = reserved bit.

Note 1: This register is not available on all devices. Refer to the specific device data sheet for availability.

2: n = 0-15

3: These bits are read-only when the module is executing Step commands.

4: Refer to [Table 3-1](#) for the Step command encoding.

Peripheral Trigger Generator (PTG)

Register 2-1: PTGCST: PTG Control/Status Low Register

R/W-0	r-0	R/W-0	R/W-0	U-0	R/W-0, HC	R/W-0	R/W-0
PTGEN	—	PTGSIDL	PTGTOGL	—	PTGSWT ⁽²⁾	PTGSSEN ⁽³⁾	PTGIVIS
bit 15							bit 8

R/W-0, HC	R/W-0, HS	R/W-0, HS/HC	U-0	U-0	U-0	R/W-0	R/W-0
PTGSTRT	PTGWDTO	PTGBUSY ⁽⁴⁾	—	—	—	PTGITM1 ⁽¹⁾	PTGITM0 ⁽¹⁾
bit 7							bit 0

Legend:	HC = Hardware Clearable bit	U = Unimplemented bit, read as '0'
R = Readable bit	W = Writable bit	HS = Hardware Settable bit r = Reserved bit
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared x = Bit is unknown

bit 15	PTGEN: PTG Enable bit 1 = PTG is enabled 0 = PTG is disabled
bit 14	Reserved: Must be written as '0'
bit 13	PTGSIDL: PTG Freeze in Debug Mode bit 1 = Halts PTG operation when device is Idle 0 = PTG operation continues when device is Idle
bit 12	PTGTOGL: PTG Toggle Trigger Output bit 1 = Toggles state of TRIG output for each execution of PTGTRIG 0 = Generates a single TRIG pulse for each execution of PTGTRIG
bit 11	Unimplemented: Read as '0'
bit 10	PTGSWT: PTG Software Trigger bit ⁽²⁾ 1 = Toggles state of TRIG output for each execution of PTGTRIG 0 = Generates a single TRIG pulse for each execution of PTGTRIG
bit 9	PTGSSEN: PTG Single-Step bit ⁽³⁾ 1 = Enables single step when in Debug mode 0 = Disables single step
bit 8	PTGIVIS: PTG Counter/Timer Visibility bit 1 = Reading the PTGSDLIM, PTGCxLIM or PTGTxLIM registers returns the current values of their corresponding Counter/Timer registers (PTGSD, PTGCx and PTGTx) 0 = Reading the PTGSDLIM, PTGCxLIM or PTGTxLIM registers returns the value of these Limit registers
bit 7	PTGSTRT: PTG Start Sequencer bit 1 = Starts to sequentially execute the commands (Continuous mode) 0 = Stops executing the commands
bit 6	PTGWDTO: PTG Watchdog Timer Time-out Status bit 1 = PTG Watchdog Timer has timed out 0 = PTG Watchdog Timer has not timed out
bit 5	PTGBUSY: PTG State Machine Busy bit ⁽⁴⁾ 1 = PTG is running on the selected clock source; no SFR writes are allowed to PTGCLK<2:0> or PTGDIV<4:0> 0 = PTG state machine is not running
bit 4-2	Unimplemented: Read as '0'

- Note 1:** These bits apply to the PTGWHI and PTGWLO commands only.
- 2:** This bit is only used with the PTGCTRL Step command software trigger option.
- 3:** The PTGSSEN bit may only be written when in Debug mode.
- 4:** This bit is not available on all devices. Check the specific device data sheet for availability.

dsPIC33/PIC24 Family Reference Manual

Register 2-1: PTGCST: PTG Control/Status Low Register (Continued)

bit 1-0 **PTGITM<1:0>**: PTG Input Trigger Operation Selection⁽¹⁾

- 11 = Single level detect with Step delay not executed on exit of command, regardless of the PTGCTRL command (Mode 3)
- 10 = Single level detect with Step delay executed on exit of command (Mode 2)
- 01 = Continuous edge detect with Step delay not executed on exit of command, regardless of the PTGCTRL command (Mode 1)
- 00 = Continuous edge detect with Step delay executed on exit of command (Mode 0)

- Note 1:** These bits apply to the PTGWHI and PTGWLO commands only.
- 2:** This bit is only used with the PTGCTRL Step command software trigger option.
- 3:** The PTGSSEN bit may only be written when in Debug mode.
- 4:** This bit is not available on all devices. Check the specific device data sheet for availability.

Peripheral Trigger Generator (PTG)

Register 2-2: PTGCONH: PTG Control/Status High Register

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGCLK<2:0>			PTGDIV<4:0>				
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
PTGPWD<3:0>				—	PTGWDT<2:0>		
bit 7							bit 0

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-13 **PTGCLK<2:0>**: PTG Module Clock Source Selection bits
Refer to the specific device data sheet for clock source selection.

bit 12-8 **PTGDIV<4:0>**: PTG Module Clock Prescaler (Divider) bits
11111 = Divide-by-32
11110 = Divide-by-31
•
•
•
00001 = Divide-by-2
00000 = Divide-by-1

bit 7-4 **PTGPWD<3:0>**: PTG Trigger Output Pulse-Width (in PTG clock cycles) bits
1111 = All trigger outputs are 16 PTG clock cycles wide
1110 = All trigger outputs are 15 PTG clock cycles wide
•
•
•
0001 = All trigger outputs are 2 PTG clock cycles wide
0000 = All trigger outputs are 1 PTG clock cycle wide

bit 3 **Unimplemented**: Read as '0'

bit 2-0 **PTGWDT<2:0>**: PTG Watchdog Timer Time-out Selection bits
111 = Watchdog Timer will time-out after 512 PTG clocks
110 = Watchdog Timer will time-out after 256 PTG clocks
101 = Watchdog Timer will time-out after 128 PTG clocks
100 = Watchdog Timer will time-out after 64 PTG clocks
011 = Watchdog Timer will time-out after 32 PTG clocks
010 = Watchdog Timer will time-out after 16 PTG clocks
001 = Watchdog Timer will time-out after 8 PTG clocks
000 = Watchdog Timer is disabled

dsPIC33/PIC24 Family Reference Manual

Register 2-3: PTGBTE: PTG Broadcast Trigger Enable Low Register^(1,2)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGBTE<15:8>							
bit 15				bit 8			

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGBTE<7:0>							
bit 7				bit 0			

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-0 **PTGBTE<15:0>**: PTG Broadcast Trigger Enable bits

1 = Generates trigger when the broadcast command is executed

0 = Does not generate trigger when the broadcast command is executed

Note 1: These bits are read-only when the module is executing Step commands.

Note 2: See the specific device data sheet for availability and assignments of these bits.

Register 2-4: PTGBTEH: PTG Broadcast Trigger Enable High Register^(1,2,3)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGBTE<31:24>							
bit 15				bit 8			

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGBTE<23:16>							
bit 7				bit 0			

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-0 **PTGBTE<31:16>**: PTG Broadcast Trigger Enable bits

1 = Generates trigger when the broadcast command is executed

0 = Does not generate trigger when the broadcast command is executed

Note 1: This register is not available on all devices. Refer to the specific device data sheet for availability.

Note 2: These bits are read-only when the module is executing Step commands.

Note 3: See the specific device data sheet for availability and assignments of these bits.

Peripheral Trigger Generator (PTG)

Register 2-5: PTGHOLD: PTG Hold Register⁽¹⁾

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGHOLD<15:8>							
bit 15				bit 8			

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGHOLD<7:0>							
bit 7				bit 0			

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-0 **PTGHOLD<15:0>**: PTG General Purpose Hold Register bits

This register holds the user-supplied data to be copied to the PTGTxLIM, PTGCxLIM, PTGSDLIM or PTGL0 register using the PTGCOPY command.

Note 1: These bits are read-only when the module is executing Step commands.

Register 2-6: PTGT0LIM: PTG Timer0 Limit Register⁽¹⁾

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGT0LIM<15:8>							
bit 15				bit 8			

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGT0LIM<7:0>							
bit 7				bit 0			

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-0 **PTGT0LIM<15:0>**: PTG Timer0 Limit Register bits

General purpose Timer0 Limit register.

Note 1: These bits are read-only when the module is executing Step commands.

dsPIC33/PIC24 Family Reference Manual

Register 2-7: PTGT1LIM: PTG Timer1 Limit Register⁽¹⁾

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGT1LIM<15:8>							
bit 15				bit 8			

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGT1LIM<7:0>							
bit 7				bit 0			

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-0 **PTGT1LIM<15:0>**: PTG Timer1 Limit Register bits
General purpose Timer1 Limit register.

Note 1: These bits are read-only when the module is executing Step commands.

Register 2-8: PTGSDLIM: PTG Step Delay Limit Register⁽¹⁾

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGSDLIM<15:8>							
bit 15				bit 8			

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGSDLIM<7:0>							
bit 7				bit 0			

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-0 **PTGSDLIM<15:0>**: PTG Step Delay Limit Register bits
This register holds a PTG Step delay value representing the number of additional PTG clocks between the start of a Step command and the completion of a Step command.

Note 1: These bits are read-only when the module is executing Step commands.

Peripheral Trigger Generator (PTG)

Register 2-9: PTGC0LIM: PTG Counter 0 Limit Register⁽¹⁾

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGC0LIM<15:8>							
bit 15				bit 8			

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGC0LIM<7:0>							
bit 7				bit 0			

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-0 **PTGC0LIM<15:0>**: PTG Counter 0 Limit Register bits

This register is used to specify the loop count for the PTGJMPC0 Step command or as a Limit register for the General Purpose Counter 0.

Note 1: These bits are read-only when the module is executing Step commands.

Register 2-10: PTGC1LIM: PTG Counter 1 Limit Register⁽¹⁾

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGC1LIM<15:8>							
bit 15				bit 8			

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGC1LIM<7:0>							
bit 7				bit 0			

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-0 **PTGC1LIM<15:0>**: PTG Counter 1 Limit Register bits

This register is used to specify the loop count for the PTGJMPC1 Step command or as a Limit register for the General Purpose Counter 1.

Note 1: These bits are read only when the module is executing Step commands.

dsPIC33/PIC24 Family Reference Manual

Register 2-11: PTGADJ: PTG Adjust Register⁽¹⁾

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGADJ<15:8>							
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGADJ<7:0>							
bit 7							bit 0

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-0

PTGADJ<15:0>: PTG Adjust Register bits

This register holds the user-supplied data to be added to the PTGTxLIM, PTGCxLIM, PTGSDLIM or PTGL0 register using the PTGADD command.

Note 1: These bits are read-only when the module is executing Step commands.

Register 2-12: PTGL0: PTG Literal 0 Register^(1,2)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGL0<15:8>							
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTGL0<7:0>							
bit 7							bit 0

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-0

PTGL0<15:0>: PTG Literal 0 Register bits

This register holds a 16-bit value to be written to the AD1CHS0 register using the PTGCTRL command.

Note 1: These bits are read-only when the module is executing Step commands.

2: The PTG strobe output is typically connected to the ADC Channel Select register. This allows the PTG to directly control ADC channel switching. See the specific device data sheet for connections of the PTG output.

Peripheral Trigger Generator (PTG)

Register 2-13: PTGQPTR: PTG Step Queue Pointer Register⁽¹⁾

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8
U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	PTGQPTR<4:0>				
bit 7							bit 0

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-5 **Unimplemented:** Read as '0'

bit 4-0 **PTGQPTR<4:0>:** PTG Step Queue Pointer Register bits

This register points to the currently active Step command in the Step queue.

Note 1: These bits are read-only when the module is executing Step commands.

Register 2-14: PTGQUn: PTG Step Queue Pointer n Register (n = 0-15)^(1,2)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STEP2n+1<7:0>							
bit 15							bit 8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STEP2n<7:0>							
bit 7							bit 0

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

bit 15-8 **STEP2n+1<7:0>:** PTG Command 4n+1 bits

A queue location for storage of the STEP2n+1 command byte, where 'n' is from PTGQUn.

bit 7-0 **STEP2n<7:0>:** PTG Command 4n bits

A queue location for storage of the STEP2n command byte, where 'n' equals the odd numbered Step Queue Pointers.

Note 1: These bits are read-only when the module is executing Step commands.

2: Refer to [Table 3-1](#) for the Step command encoding.

dsPIC33/PIC24 Family Reference Manual

3.0 STEP COMMANDS AND FORMAT

The PTG operates using eleven 8-bit Step commands to perform higher level tasks. There are four types of Step commands:

- Input Event Control
- Control Functions
- Flow Control
- Output Generation

Combinations and sequences of these commands can be used to make decisions and take action without CPU intervention.

3.1 Input Event Control

There are two input event control commands, `PTGWHI` and `PTGWLO`, that wait for a high or low edge on one of the 16 `PTGlx` inputs. These commands are used in conjunction with a 4-bit `OPTION` field that specifies which `PTGlx` is used. Once the specified input transitions in the intended direction, the Step queue is incremented and the next Step command is evaluated. See [Section 4.7.2 “Wait for Trigger Input”](#) for additional information.

3.2 Control Functions

There are three commands for control functions, `PTGCTRL`, `PTGADD` and `PTGCOPY`. The `PTGCTRL` command controls the operation of the delay timers, software triggers and the strobe output. The `PTGADD` command is used to add the contents of the `PTGHOLD` register to other PTG registers, including the counters, timers, Step Delay and Literal register. The `PTGCOPY` command is used to copy the contents of the `PTGHOLD` register to other PTG registers, similar to the `PTGADD` command.

3.3 Flow Control

Flow control is accomplished using the 3 jump commands, `PTGJMP`, `PTGJMPC0` and `PTGJMPC1`. These jump commands are 3 bits to allow a larger 5-bit `OPTION` to match that of the Queue Pointer, `PTGQPTR`. The `PTGJMP` command simply jumps to the specified queue location, whereas the `PTGJMPCx` commands are conditional jumps based on the comparison of the counters to the PTG Counter Limit registers (`PTGCxLIM`).

3.4 Output Generation

Output generation is achieved using the `PTGTRIG`, `PTGIRQ` and `PTGSTRB` commands. `PTGTRIG` is used to select and generate an output trigger (`PTGOx`). `PTGTRIG` also uses a 5-bit `OPTION` field to support the 32 `PTGOs`' selections. The `PTGIRQ` command is used to generate an interrupt request with the `OPTION` field specifying the interrupt (`PTGxIF`). See [Section 4.9 “PTG Module Outputs”](#) for additional information on triggers and interrupts. The `PTGSTRB` command is used to generate a strobe output, which outputs 16 bits of data to another peripheral. See [Section 4.10 “Strobe Output”](#) for additional information.

Peripheral Trigger Generator (PTG)

Table 3-1 provides an overview of the PTG commands and Table 3-2 elaborates on the options available for each command. Example 3-1, Example 3-2 and Example 3-3 provide C code examples for command definitions and their options. Later examples in this FRM refer back to these examples.

Table 3-1: PTG Step Command Format and Description

Step Command Byte			
		STEPx<7:0>	
CMD<3:0>		OPTION<3:0>	
bit 7	bit 4 bit 3		bit 0

bit 7-4	Step Command	CMD<3:0>	Command Description
	PTGCTRL	0000	Execute the control command as described by the OPTION<3:0> bits.
	PTGADD	0001	Add contents of the PTGADJ register to the target register as described by the OPTION<3:0> bits.
	PTGCOPY		Copy contents of the PTGHOLD register to the target register as described by the OPTION<3:0> bits.
	PTGSTRB	001x	Copy the values contained in the bits, CMD<0>:OPTION<3:0>, to the strobe output bits<4:0>.
	PTGWHI	0100	Wait for a low-to-high edge input from a selected PTG trigger input as described by the OPTION<3:0> bits.
	PTGWLO	0101	Wait for a high-to-low edge input from a selected PTG trigger input as described by the OPTION<3:0> bits.
	—	0110	Reserved; do not use. ⁽¹⁾
	PTGIRQ	0111	Generate an individual interrupt request as described by the OPTION<3:0> bits.
	PTGTRIG	100x	Generate an individual trigger output as described by the 5-bit field of CMD<0>:OPTION<3:0>.
	PTGJMP	101x	Copy the values contained in the bits, CMD<0>:OPTION<3:0>, to the PTGQPTR register and jump to that Step queue.
	PTGJMPC0	110x	PTGC0 = PTGC0LIM: Increment the PTGQPTR register.
			PTGC0 ≠ PTGC0LIM: Increment Counter 0 (PTGC0) and copy the values contained in the bits, CMD<0>:OPTION<3:0>, to the PTGQPTR register, and jump to that Step queue.
	PTGJMPC1	111x	PTGC1 = PTGC1LIM: Increment the PTGQPTR register.
			PTGC1 ≠ PTGC1LIM: Increment Counter 1 (PTGC1) and copy the values contained in the bits, CMD<0>:OPTION<3:0>, to the PTGQPTR register, and jump to that Step queue.

Note 1: All reserved commands or options will execute, but they do not have any affect (i.e., they execute as a NOP instruction).

Example 3-1: PTG Command Definitions

/* PTG Commands */	
#define PTGCTRL(x)	((0x0 << 4) ((x) & 0x0F))
#define PTGADD(x)	((0x2 << 3) ((x) & 0x07))
#define PTGCOPY(x)	((0x3 << 3) ((x) & 0x07))
#define PTGSTRB(x)	((0x1 << 5) ((x) & 0x1F))
#define PTGWHI(x)	((0x4 << 4) ((x) & 0x0F))
#define PTGWLO(x)	((0x5 << 4) ((x) & 0x0F))
#define PTGIRQ(x)	((0x7 << 4) ((x) & 0x0F))
#define PTGTRIG(x)	((0x4 << 5) ((x) & 0x1F))
#define PTGJMP(x)	((0x5 << 5) ((x) & 0x1F))
#define PTGJMPC0(x)	((0x6 << 5) ((x) & 0x1F))
#define PTGJMPC1(x)	((0x7 << 5) ((x) & 0x1F))

dsPIC33/PIC24 Family Reference Manual

Table 3-2: PTG Command Options

bit 3-0	Step Command	OPTION<3:0>	Command Description
	PTGCTRL ⁽¹⁾	0000	NOP
		0001	Reserved; do not use.
		0010	Disable Step delay timer (PTGSD).
		0011	Reserved; do not use.
		0100	Reserved; do not use.
		0101	Reserved; do not use.
		0110	Enable Step delay timer (PTGSD).
		0111	Reserved; do not use.
		1000	Start and wait for the PTGT0 to match the PTGT0LIM register.
		1001	Start and wait for the PTGT1 to match the PTGT1LIM register.
		1010	Wait for the software trigger (level, PTGSWT = 1).
		1011	Wait for the software trigger (positive edge, PTGSWT = 0 to 1).
		1100	Copy the PTGC0LIM register contents to the strobe output.
		1101	Copy the PTGC1LIM register contents to the strobe output.
		1110	Copy the PTGL0 register contents to the strobe output.
		1111	Generate the triggers indicated in the PTGBTE register.
	PTGADD ⁽¹⁾	0000	Add the PTGADJ register contents to the PTGC0LIM register.
		0001	Add the PTGADJ register contents to the PTGC1LIM register.
		0010	Add the PTGADJ register contents to the PTGT0LIM register.
		0011	Add the PTGADJ register contents to the PTGT1LIM register.
		0100	Add the PTGADJ register contents to the PTGSDLIM register.
		0101	Add the PTGADJ register contents to the PTGL0 register.
		0110	Reserved; do not use.
		0111	Reserved; do not use.
	PTGCOPY ⁽¹⁾	1000	Copy the PTGHOLD register contents to the PTGC0LIM register.
		1001	Copy the PTGHOLD register contents to the PTGC1LIM register.
		1010	Copy the PTGHOLD register contents to the PTGT0LIM register.
		1011	Copy the PTGHOLD register contents to the PTGT1LIM register.
		1100	Copy the PTGHOLD register contents to the PTGSDLIM register.
		1101	Copy the PTGHOLD register contents to the PTGL0 register.
		1110	Reserved; do not use.
		1111	Reserved; do not use.

Note 1: All reserved commands or options will execute, but they do not have any affect (i.e., they execute as a NOP instruction).

Peripheral Trigger Generator (PTG)

Table 3-2: PTG Command Options (Continued)

bit 3-0	Step Command	OPTION<3:0>	Option Description
	PTGWHI ⁽¹⁾ or PTGWLO ⁽¹⁾	0000	PTGI0 (see specific device data sheet for input assignments).
		• • •	• • •
		1111	PTGI15 (see specific device data sheet for input assignments).
	PTGIRQ ⁽¹⁾	0000	Generate PTG Interrupt 0 (see specific device data sheet for interrupt assignments).
		• • •	• • •
		0111	Generate PTG Interrupt 7 (see specific device data sheet for interrupt assignments).
		1000	Reserved; do not use.
		• • •	• • •
		1111	Reserved; do not use.
	PTGTRIG	00000	PTGO0 (see specific device data sheet for assignments).
		00001	PTGO1 (see specific device data sheet for assignments).
		• • •	• • •
		11110	PTGO30 (see specific device data sheet for assignments).
		11111	PTGO31 (see specific device data sheet for assignments).

Note 1: All reserved commands or options will execute, but they do not have any affect (i.e., they execute as a NOP instruction).

dsPIC33/PIC24 Family Reference Manual

Example 3-2: PTGCTRL Options

```
/ Used with PTGCTRL command
typedef enum
{
    stepDelayDisable      = 0x2,
    stepDelayEnable       = 0x6,
    t0Wait                 = 0x8,
    t1Wait                 = 0x9,
    softTriggerLevelWait  = 0xA,
    softTriggerEdgeWait   = 0xB,
    c0Strobe               = 0xC,
    c1Strobe               = 0xD,
    l0Strobe               = 0xE,
    triggerGenerate        = 0xF,
} CTRL_T;
```

Example 3-3: Options for PTGADD and PTGCOPY Commands

```
/ Used with PTGADD and PTGCOPY commands
typedef enum
{
    c0Limit   = 0x0,
    c1Limit   = 0x1,
    t0Limit   = 0x2,
    t1Limit   = 0x3,
    stepDelay = 0x4,
    literal0  = 0x5,
} ADD_COPY_T;
```

Peripheral Trigger Generator (PTG)

4.0 MODULE OPERATION

4.1 PTG Description

The PTG module is a user-programmable sequencer for generating complex peripheral trigger sequences. The PTG module provides the ability to schedule complex peripheral operations, which would be difficult or impossible to achieve via a software solution.

The user writes 8-bit commands, called Step commands, to the PTG Queue registers (PTGQUE0-PTGQUE15). Each 8-bit Step command is made up of a command code and an option field. [Table 3-1](#) shows the format and encoding of a Step command. Based on the commands, the PTG can interact with other peripherals, such as the PWM, ADC, MCCP, input capture and output compare. See the device-specific data sheet for availability of peripherals.

4.2 PTG Clock Selection

The PTG module has multiple clock options and has a selectable prescaler, which divides the PTG clock input from 1 to 32.

4.2.1 CLOCK SOURCE SELECTION

The [PTGCLK<2:0>](#) bits (PTGCON<15:13>) specify the clock source for the PTG clock generation logic. These clock sources are device-specific. Refer to the specific device data sheet for available sources.

4.2.2 CLOCK PRESCALER SELECTION

The [PTGDIV<4:0>](#) bits (PTGCON<12:8>) specify the prescaler value for the PTG clock generation logic. These bits can be written only when the PTG module is disabled (PTGEN = 0).

Note: Any attempt to write to the PTGDIV<4:0> bits or the PTGCLK<2:0> bits while PTGEN = 1 will have no effect.

4.2.3 MODULE ENABLE DELAY

Once the PTG module is enabled (PTGEN = 1), there is a delay before the PTG starts to execute commands. This delay is expressed in [Equation 4-1](#). The PTG clock period is the effective clock after the prescaler.

Equation 4-1:

$$TDLYEN = 4 \cdot PTG \text{ Clock Period (Maximum)}$$

The user must ensure that no control bits are modified during the delay. Also, no external triggers are asserted prior to the PTG state machine commencing execution; otherwise, the triggers will be missed.

4.3 Basic Operation

The user loads the Step commands (8-bit values) into the PTG Queue registers. The commands define a sequence of events for generating the trigger output signals to the peripherals. The Step commands can also be used to generate the interrupt requests to the processor.

The PTG module is enabled and clocked when the PTGEN bit (PTGCST<15>) = 1. While the PTGSTRT bit (PTGCST<7>) = 0, the PTG module is in the Halt state.

- Note 1:** The control registers cannot be modified when the PTG module is in the Halt state.
- 2:** The PTG module must be enabled (PTGEN = 1) prior to attempting to set the PTGSTRT bit.
- 3:** The user should not attempt to set the PTGEN and PTGSTRT bits within the same data write cycle.

Subsequently, setting PTGSTRT = 1 will enable the module for Continuous mode execution of the Step command queue. The PTG sequencer will start to read the Step queue at the address held in the Queue Pointer (PTGQPTR). Each command byte is read, decoded and executed sequentially. The minimum duration of any Step command is one PTG clock as explained in [Section 4.2 “PTG Clock Selection”](#).

Step commands will execute sequentially until any of the following occurs:

- A PTGJMP, PTGJMPC0 or PTGJMPC1 (flow change) Step command is executed.
- The user clears the PTGSTRT bit, stopping the PTG sequencer. No further Step commands are read/decoded and execution halts.
- The internal Watchdog Timer overflows, clearing the PTGSTRT bit and stopping the PTG sequencer. No further Step commands are read/decoded and execution halts.
- The PTG module is disabled (PTGEN = 0).

The Step commands can also be made to wait on a condition, such as an input trigger edge, a software trigger or a timer match, before continuing execution. For more information, refer to [Section 4.11 “Stopping the Sequencer”](#).

Peripheral Trigger Generator (PTG)

4.4 Control Register Access

When the PTG module is enabled (PTGEN = 1), writes are inhibited to all control registers with the exception of the PTGCST register, which may be read and written to as normal.

When the PTG module is enabled (PTGEN = 1), reads can be performed from any control register at any time; however, the data read (control register or associated timer/counter value) will depend upon the state of the PTGIVIS bit (PTGCST<8>).

Note: Only some registers are affected by the state of the PTGIVIS bit. Refer to the PTGIVIS bit description.

When the PTG module is disabled (PTGEN = 0), all control registers can be read and written to as normal. The PTGIVIS bit (PTGCST<8>) has no effect when PTGEN = 0, because all timers/counters will be cleared to zero; however, the values in the Limit register will stay same.

4.5 Step Queue Pointer

The PTG Step Queue Pointer register (PTGQPTR) holds the pointer to the PTG queue. The pointer is loaded from the PTGQPTR register when PTGEN = 1 and the PTGSTRT bit is set, and addresses the currently active Step command in the Step queue. The PTGQPTR register is cleared when the module is disabled (PTGEN = 0) or is in the Reset state. This is a one-time event that occurs on transition of the PTGEN bit from '1' to '0'.

- Note 1:** The PTGQPTR register is not cleared when the PTG module is halted (PTGSTRT = 0).
- 2:** The PTGQPTR register is cleared when the PTGEN bit is cleared, but any value can be written to the register irrespective of the state of the PTGEN bit.

The user can read the PTGQPTR register at any time. In the Disabled state (PTGEN = 0), a read returns the contents of the PTGQPTR register. In the Idle or Active state (PTGEN = 1), a read returns the contents of the pointer.

The PTGQPTR register is typically incremented during the first cycle of each command. The exceptions to this rule are:

- If the PTGJMP command is executed: The Step Queue Pointer is loaded with the target queue address
- If the PTGJMPCx command is executed and PTGCx is not zero: The Step Queue Pointer is loaded with the target queue address

4.6 Command Looping Control

Two 16-bit loop counters are provided (PTGC0 and PTGC1) that can be used by the PTGJMPCx command as a block loop counter or delay generator.

Each loop counter consists of an incrementing counter (PTGCx) and a Limit register (PTGCxLIM). The Limit register value can be changed by writing directly to the register (when the module is disabled) or by the PTG sequencer (when the module is enabled). The data read from the Limit register depends upon the state of the PTG Counter/Timer Visibility bit (PTGIVIS). The counters are cleared when the module is in the Reset state or when the PTG module is disabled (PTGEN = 0).

4.6.1 USING THE LOOP COUNTER AS A BLOCK LOOP COUNTER

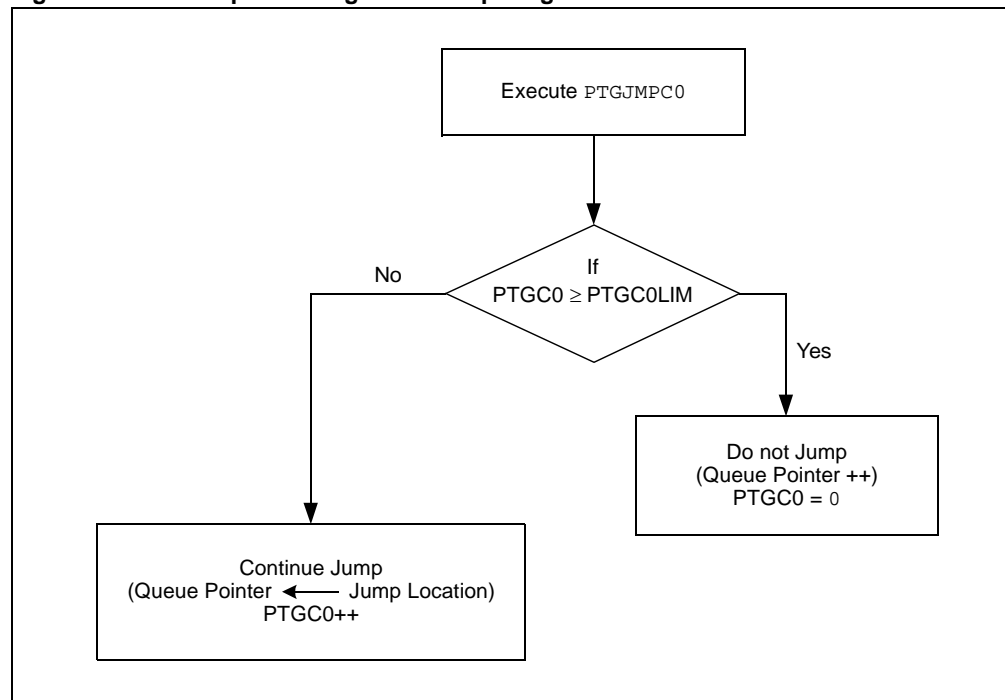
The `PTGJMPCx` (Jump Conditional) command uses one of the loop counters to keep track of the number of times the `PTGJMPCx` command is executed, and can therefore, be used to create code block loops. This is useful in applications where a sequence of peripheral events needs to be repeated several times. The `PTGJMPCx` command allows the user to create code loops and use fewer Step commands.

Each time the `PTGJMPCx` command is executed, the corresponding internal loop counter is compared to its limit value. If the loop counter has not reached the limit value, the jump location is loaded into the `PTGQPTR` register and the loop counter is incremented by 1. The next command will be fetched from the new queue location. If the counter has reached the limit value, the sequencer proceeds to the next command (i.e., increments the Queue Pointer). While preparing for the next `PTGJMPCx` command loop execution, the corresponding loop counter is cleared (see Figure 4-1).

Note: The loop counter value can be modified (via the `PTGADD` or `PTGCOPY` command) prior to execution of the first iteration of the command loop.

The provision for two separate loop counters and associated `PTGJMPCx` commands allows for the nested loops to be supported (one-level deep). There are no restrictions with regard to which `PTGJMPCx` command resides in the inner or outer loops.

Figure 4-1: Implementing Block Loop Diagram



Peripheral Trigger Generator (PTG)

4.7 Sequencer Operation

All commands are executed in a single cycle, except for the flow change commands and the commands that are waiting for an external input.

4.7.1 STEP COMMAND DURATION

By default, each Step command executes in one PTG clock period. There are several methods to slow the execution of the Step commands:

- Wait for a trigger input
- Wait for a GP timer (PTGTxLIM)
- Insert a delay loop using the PTGJMP and PTGJMPCx commands
- Enable and insert a Step delay (PTGSDLIM) after execution of each command

4.7.2 WAIT FOR TRIGGER INPUT

The PTG module can support up to 16 independent trigger inputs. The PTG inputs, PTGI0 through PTGI15, are device-specific; refer to the device data sheet for availability. The user can specify a Step command that waits for a positive or negative edge, or a high or low level, of the selected input signal to occur. The operating mode is selected by the PTGITM<1:0> bits in the PTGCST register.

The PTGWHI command looks for a positive edge or high state to occur on the selected trigger input. The PTGWLO command looks for a negative edge or low state to occur on the selected trigger input. The PTG repeats the trigger input command (i.e., effectively waits) until the selected signal becomes valid before continuing the Step command execution.

The minimum execution time to wait for a trigger is one PTG clock. There is no limit for the PTG wait for a trigger input other than that enforced by the Watchdog Timer. Refer to [Section 4.8 “PTG Watchdog Timer”](#) for more information.

The PTG module supports four input trigger command operating modes (Mode 0-Mode 3), which are selected by the PTGITM<1:0> bits in the PTGCST register.

Note:	If the Step delay is disabled, Mode 0 and Mode 1 are equivalent in operation, and Mode 2 and Mode 3 are equivalent in operation.
--------------	--

4.7.2.1 Mode 0: PTGITM<1:0> = 0x00 (Continuous Edge Detect with Step Delay at Exit)

In this mode, the selected trigger input is continuously tested starting immediately after the PTGWHI or PTGWLO command is executed. When the trigger edge is detected, the command execution completes.

If the Step delay counter is enabled, the Step delay will be inserted (once) after the valid edge is detected and after the command execution.

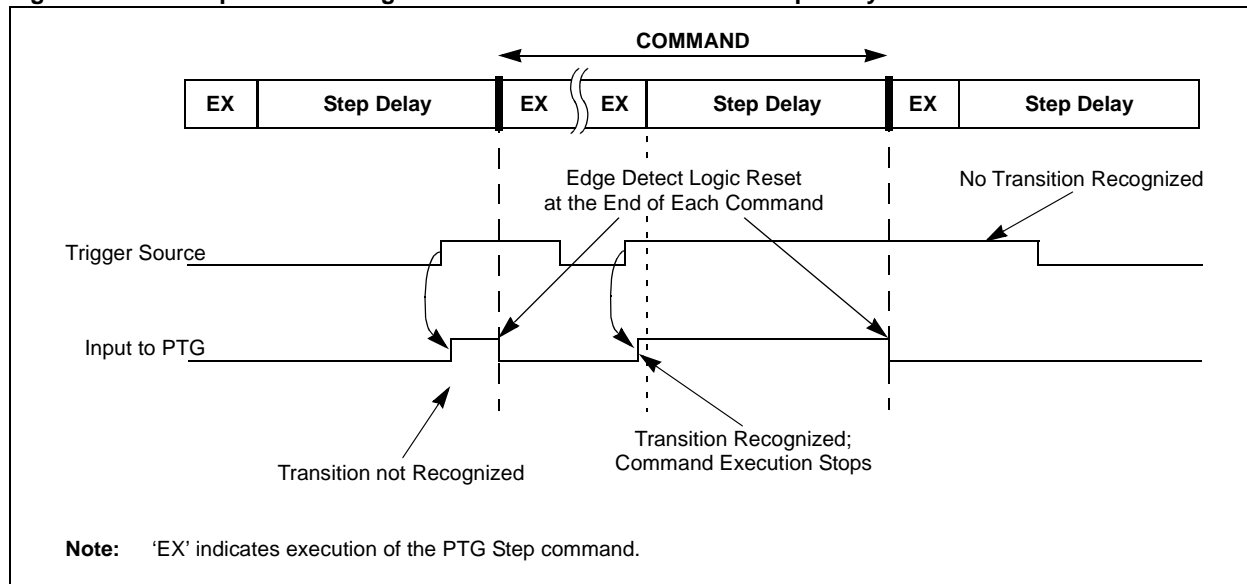
If the Step delay counter is not enabled, the command will complete after the valid edge is detected and execution of the subsequent command will commence immediately.

Note:	The edge detect logic is reset after the command execution is complete (i.e., prior to any Step delay associated with the command). For the edge to be detected, the edge should occur during the PTGWHI or PTGWLO command execution, or during the Step delay of the prior command.
--------------	--

dsPIC33/PIC24 Family Reference Manual

Figure 4-2 shows an example timing diagram of Mode 0 operation.

Figure 4-2: Operation of Edge-Sensitive Command with Exit Step Delay



4.7.2.2 Mode 1: PTGITM<1:0> = 0x01 (Continuous Edge Detect without Step Delay at Exit)

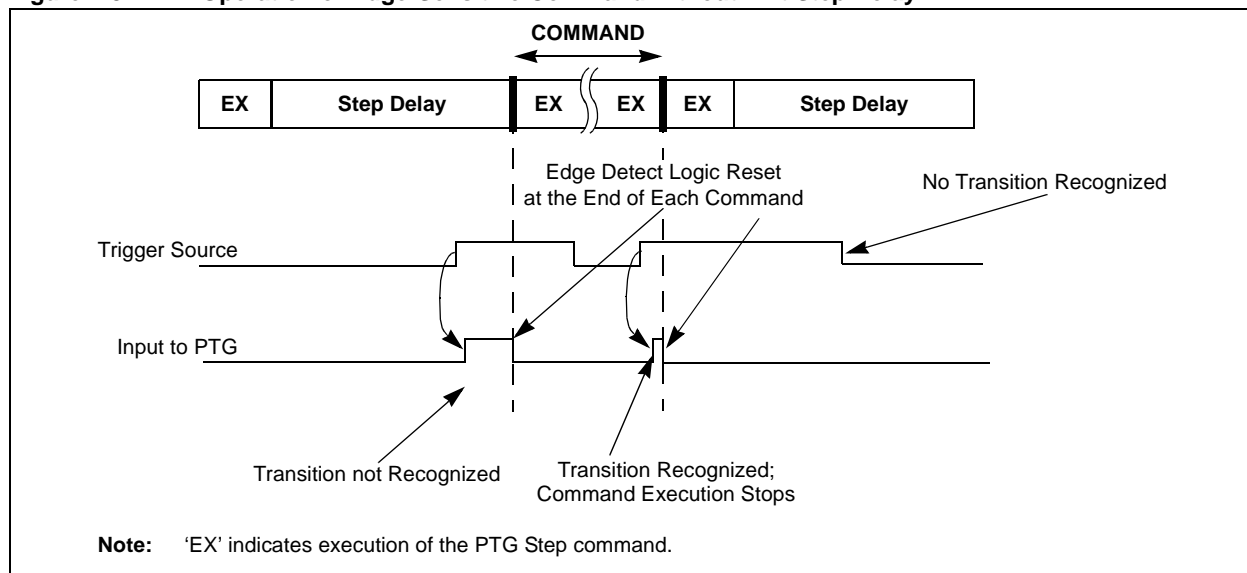
In this mode, the selected trigger input is continuously tested starting immediately after the PTGWHI or PTGWLO command is executed. When the trigger edge is detected, the command execution completes.

Regardless of whether the Step delay counter is enabled or disabled, the Step delay will not be inserted after the command execution has completed.

Note: The edge detect logic is reset after the command execution completes. To be detected, the edge may therefore occur during the PTGWHI or PTGWLO command execution, or during the Step delay of the prior command.

Figure 4-3 shows an example timing diagram of Mode 1 operation.

Figure 4-3: Operation of Edge-Sensitive Command without Exit Step Delay



Peripheral Trigger Generator (PTG)

4.7.2.3 Mode 2: PTGITM<1:0> = 0x10 (Sampled Level Detect with Step Delay at Exit)

In this mode, the selected trigger input is sample tested for a valid level immediately after the PTGWHI or PTGWLO command is executed; the trigger input is tested (once per PTG clock).

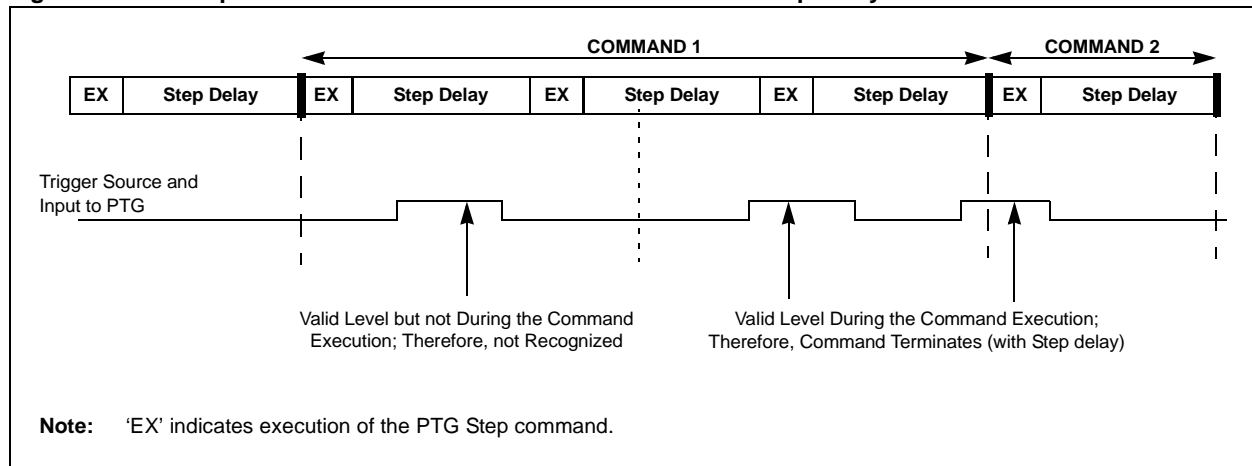
If the trigger does not occur, and the Step delay is enabled, the command waits for the Step delay to expire before testing the trigger input again. When the trigger occurs, the command execution completes and the Step delay is reinserted.

If the trigger does not occur and the Step delay is disabled, the command immediately tests the trigger input again during the next PTG clock cycle. When the trigger occurs, the command execution completes and execution of the subsequent command will commence immediately.

- Note 1:** As this operating mode is level-sensitive, if the input trigger level is true at the start of execution of the PTGWHI or PTGWLO command, the input test will be instantly satisfied.
- 2:** The input is not latched, therefore, it must be valid when the command executes in order to be recognized.

Figure 4-4 shows an example timing diagram of Mode 2 operation.

Figure 4-4: Operation of Level-Sensitive Command with Exit Step Delay



4.7.2.4 Mode 3: PTGITM<1:0> = 0x11
(Sampled Level Detect without Step Delay at Exit)

In this mode, the selected trigger input is sample tested for a valid level immediately after the PTGWHI or PTGWLO command is executed; the trigger input is tested (once per PTG clock).

If the trigger does not occur and the Step delay is enabled, the command waits for the Step delay to expire before testing the trigger input again. When the trigger is found to be true, the command execution completes and execution of the subsequent command will commence immediately. The Step delay is not inserted.

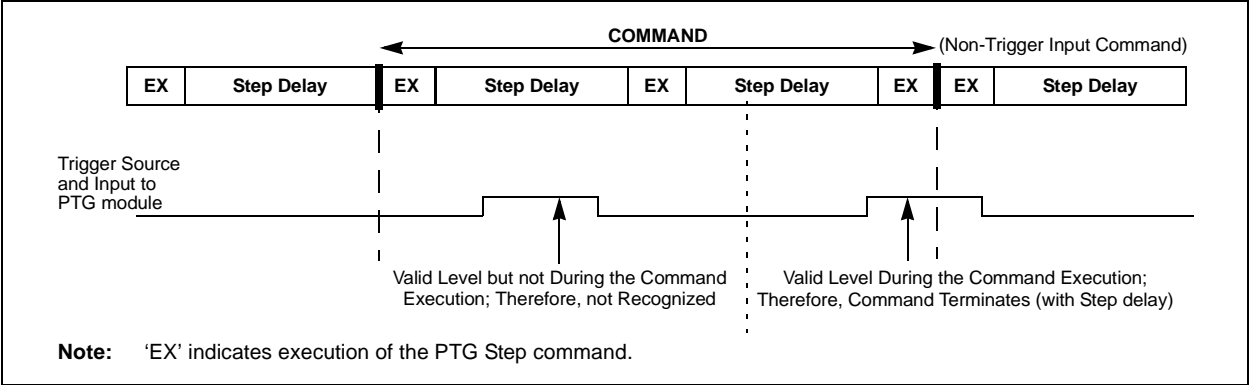
If the trigger does not occur and the Step delay is disabled, the command immediately tests the trigger input again during the next PTG clock cycle. When the trigger occurs, the command execution completes and execution of the subsequent command will commence immediately.

- Note 1:** As this operating mode is level-sensitive, if the input trigger level is true at the start of execution of the PTGWHI or PTGWLO command, the input test will be instantly satisfied.

2: The input is not latched, therefore, it must be valid when the command executes in order to be recognized.

Figure 4-5 shows an example timing diagram of Mode 3 operation.

Figure 4-5: Operation of Level-Sensitive Command without Exit Step Delay



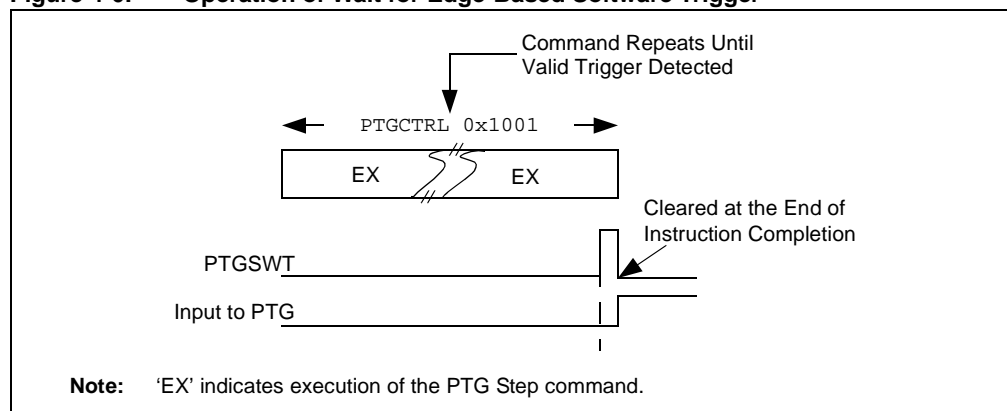
Peripheral Trigger Generator (PTG)

4.7.3 WAIT FOR SOFTWARE TRIGGER

The user can set either a PTGCTRL 0x1011 (edge-triggered) or PTGCTRL 0x1010 (level-triggered) command to wait for a software generated trigger. This trigger is generated by setting the PTGSWT bit (PTGCST<10>).

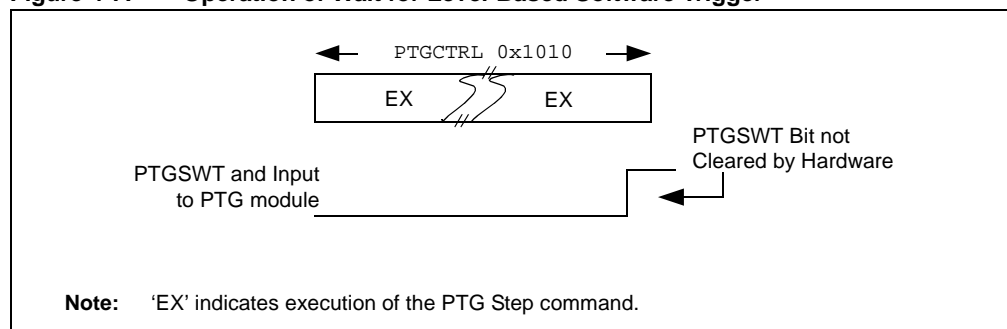
The PTGCTRL 0x1011 command is sensitive only to the PTGSWT bit transition from '0' to '1'. This transition must occur during the command execution; otherwise, the command will continue to wait. The PTGSWT bit is automatically cleared by hardware on completion of the PTGCTRL 0x1011 command execution, initializing the bit for the next software trigger command. Figure 4-6 explains the operation of the wait for edge-based software trigger.

Figure 4-6: Operation of Wait for Edge-Based Software Trigger



The PTGCTRL 0x1010 command is sensitive to the level of the PTGSWT bit. This command waits until PTGSWT = 1. It will complete immediately if PTGSWT = 1 upon entry to the command. The PTGSWT bit is not automatically cleared by the PTGCTRL 0x1010 command. If desired, the PTGSWT bit can be cleared by the user application on completion of the PTGCTRL 0x1010 command execution. Figure 4-7 explains the operation of the wait for the level-based software trigger.

Figure 4-7: Operation of Wait for Level-Based Software Trigger



The PTGSWT bit, along with a PTG Step command, generates an interrupt that allows the user to coordinate activity between the PTG module and the application software.

Note: The level-sensitive software trigger (PTGCTRL 0x1010) is not available on all devices. For details, refer to the specific device data sheet.

4.7.4 WAIT FOR GP TIMER

The PTG has two internal dedicated 16-bit General Purpose (GP) timers (PTGT0 and PTGT1), which can be used by the sequencer to wait for a specified period. The Step commands are available for loading, modifying or initializing the GP timers.

Each GP timer consists of an incrementing timer (PTGTx) and a Limit register (PTGTxLIM). The Limit register value can be changed by a CPU write (when the module is disabled) or by the PTG sequencer (when the module is enabled). Data read from the Limit register depends upon the state of the PTG Counter/Timer Visibility bit (PTGIVIS).

When running, the timers increment on the rising edge of the PTG clock, which is defined in the PTGCST register. The user can specify a wait operation using a GP timer by executing the appropriate PTGCTRL 0x1000 or PTGCTRL 0x1001 command (wait for selected GP timer).

When waiting for the GP timer, the command will wait until the value of the timer (Timer0 or Timer1) reaches its respective limit value (PTGT0LIM or PTGT1LIM). On reaching the limit value, the Step command execution completes and the next command will start. The timer is also cleared for its next use. All timers are cleared when the device is in the Reset state or when the PTG module is disabled (PTGEN = 0).

4.7.5 STEP COMMAND DELAY

The Step Delay Timer (SDLY) is a convenient method to make each Step command consume a specific amount of time. Normally, the user specifies a Step delay equal to the duration of a peripheral function, such as the ADC conversion time. The Step delay enables the user to generate the trigger output signals at a controlled rate, thereby avoiding overload on the target peripheral.

The PTGSDLIM register defines the additional time duration of each Step command in terms of PTG clocks.

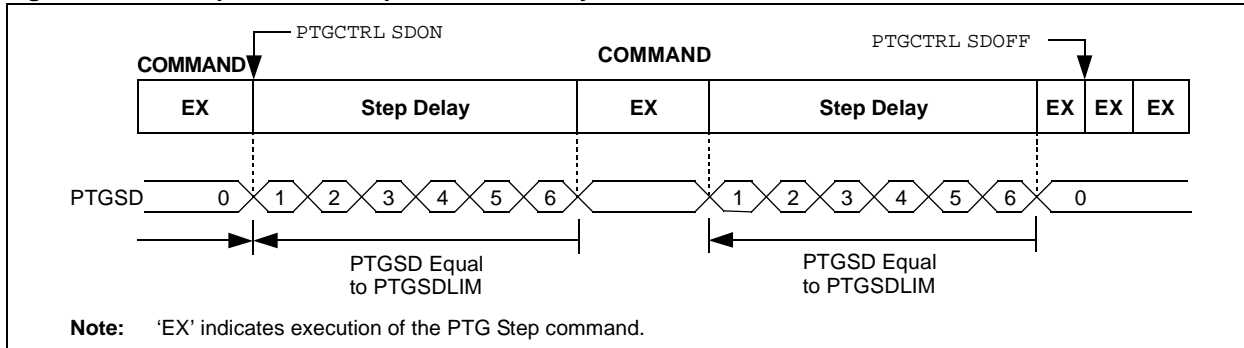
By default, the SDLY is disabled. The user can enable and disable the SDLY via the PTGCTRL 0x0110 or PTGCTRL 0x0010 command, which can be placed in the Step queue.

When operating, the SDLY increments at the PTG clock rate defined in the PTGCST register. The PTGSDLIM register value is referred to as the Step delay timer limit value. The Step delay is inserted after each command is executed, so that all Step commands are stalled until the PTGSD timer reaches its limit value. On reaching the limit value, the command execution completes and the next command starts execution. The timer is also cleared during execution of each command, so that it is ready for the next command execution.

Note: The PTGSDLIM register value of '0x0000' does not insert the additional PTG clocks when the Step delay timer is enabled. The PTGSDLIM register value of '0x0001' inserts a single PTG Step delay (1 PTG clock) into every subsequent instruction after the Step delay timer is enabled.

The trigger sources for the edge-sensitive commands (PTGCTRL 0x1011 and PTGWHI/PTGWLO when operated in Edge-Sensitive mode) have an additional hardware, external to the sequencer, to recognize the appropriate edge transition. The hardware is reset at the end of each command to maintain the edge-sensitive nature of these input triggers. If an additional valid edge occurs during a Step delay that has been inserted after the Step command has executed, it will not be recognized by any subsequent Step command. Figure 4-8 explains the operation of the Step command delay.

Figure 4-8: Operation of Step Command Delay



Peripheral Trigger Generator (PTG)

4.8 PTG Watchdog Timer

A Watchdog Timer (WDT) is required as the PTG can wait indefinitely for an external event when executing the following commands:

- Wait for hardware trigger positive edge or high state (PTGWHI)
- Wait for hardware trigger negative edge or low state (PTGWLO)

The WDT is enabled and it starts counting when the command starts to execute. It is disabled when the command completes execution and prior to any Step delay insertion. All other commands execute with the predefined cycle counts.

Note: The PTG Watchdog Timer is not required during execution of the PTGCTRL 0x1011 or PTGCTRL 0x1010 command. It is assumed that correct operation of the device will be monitored through other means.

4.8.1 OPERATION OVERVIEW

If an expected event fails to arrive before the WDT time-out period expires, the PTG module:

1. Aborts the (failing) command underway.
2. Halts the sequencer (PTGSTRT = 0).
3. Sets PTGWDTO = 1.
4. Issues a Watchdog Timer error interrupt to the processor.

The user can either use the Watchdog Timer error interrupt or periodically poll the PTGWDTO bit (PTGCST<6>) to determine that a WDT event has occurred.

4.8.2 CONFIGURATION

The WDT is configured by setting the PTGWDT<2:0> bits (PTGCON<2:0>). The WDT counts the PTG clocks as defined by the PTGCLK<2:0> and PTGDIV<4:0> bits in the PTGCON register. For more information, refer to [Section 4.2 “PTG Clock Selection”](#). The WDT time-out count value is selected by using the PTGWDT<2:0> bits and is disabled when PTGWDT<2:0> = 0x000.

Note 1: The WDT is disabled prior to insertion of any Step delay; therefore, the user does not need to account for the Step delay when calculating a suitable WDT time-out value.

- 2: Some bits within the PTGCON register are read-only when PTGSTRT = 1 (Sequencer executing commands). Refer to [Register 2-2](#).

4.8.3 WATCHDOG TIMER EVENT RECOVERY

If a WDT event occurs, the user has the option to take the necessary action to identify and fix the problem, and then continue the Step command sequence, or can simply restart the sequence.

To clear the PTGWDTO bit and to restart the PTG Sequencer from the start of the Step queue, disable (PTGEN = 0) and re-enable (PTGEN = 1) the PTG module, and then restart execution (PTGSTRT = 1).

Alternatively, as the Sequencer is only halted (not reset), the user has the option to examine the PTGQPTR register to identify which Step command was the source of the problem and can then take a corrective action. The offending command is aborted prior to the PTGQPTR update. Therefore, it will still address the failing command after the WDT event. After the PTGWDTO bit is cleared, the Step queue can be restarted at the same command by setting PTGSTRT = 1.

Note: The user should clear the PTGWDTO bit after a WDT event. Failing to clear the bit will not interfere with the subsequent module operation, but it will not be possible for the bit to poll any future WDT events.

4.9 PTG Module Outputs

The PTG module can generate trigger, interrupt and strobed data outputs by execution of specific Step commands.

4.9.1 TRIGGER OUTPUTS

The PTG module can generate up to 32 unique trigger output signals. There are two types of trigger output functions:

- Individual
- Broadcast

The PTG module can generate an individual output trigger on any one of the trigger outputs using the `PTGTRIG` command. The trigger outputs are device-specific. Refer to the specific device data sheet for availability.

The individual trigger outputs are typically used to trigger individual ADC input conversion operations, but can be assigned to any function, including general purpose I/O ports. When the PTG module is used with a compatible peripheral, such as the ADC module, the individual trigger output signals of the PTG are individually assigned to specific analog input conversion controllers within the ADC module.

The broadcast trigger output feature is specified by the `PTGBTE/PTGBTEH` registers. Each bit in the `PTGBTE/PTGBTEH` registers corresponds to an associated individual trigger output. If a bit is set in the `PTGBTE/PTGBTEH` registers, and a broadcast trigger Step command (`PTGCTRL 0x1111`) is executed, the corresponding individual trigger output is asserted. The broadcast trigger output enables the user to simultaneously generate a large number of trigger outputs with a single Step command.

4.9.2 INTERRUPT OUTPUTS

The PTG module can generate up to 16 unique interrupt request signals. These signals are useful for interacting with an application software to create more complex functions.

The PTG module can generate an individual interrupt pulse by using the `PTGIRQ` command.

4.10 Strobe Output

The strobe output of the PTG module can be used to output data from the PTG module. Typically, this output is connected to the ADC Channel Selection register, allowing the PTG to loop through ADC channels. The device-specific data sheet will indicate how the PTG strobe output is connected to other peripherals.

The `PTGCTRL 0x1110` command writes the `PTGL0` register contents to the strobe output. The `PTGL0` register can be modified by using the `PTGADD` and `PTGCOPY` commands.

The `PTGCTRL 0x1100` command writes the `PTGC0` register contents to the strobe output. The `PTGCTRL 0x1101` command writes the `PTGC1` register contents to the strobe output.

Peripheral Trigger Generator (PTG)

4.10.1 OUTPUT TIMING

All triggers, interrupts and data strobe outputs are internally asserted by the PTG state machine when the corresponding Step command execution starts (i.e., before any additional time specified by the Step delay timer) on the rising edge of the PTG execution clock.

Note: If a command has triggered the pulse-width delay counter, the counter is synchronously reset with respect to the PTG clock, terminating the pulse (subject to a minimum pulse width of 1 PTG clock cycle).

In Pulse mode (PTGTOGL = 0), the width of the trigger output signals is determined by the PTGPWD<3:0> bits (PTGCON<7:4>), and can be any value between 1 and 16 PTG clock cycles. The default value is 1 PTG clock cycle.

Refer to [Section 4.10.1.2 “TRIG Negation When PTGTOGL = 1”](#) when operating in Toggle mode (PTGTOGL = 1).

When globally controlled by the PTGCTRL 0x1111 broadcast trigger command, the TRIG output pulse width is determined by the PTGPWD<3:0> bits (PTGCON<7:4>), and can be any value between 1 and 16 PTG clock cycles. The default value is 1 PTG clock cycle.

Note: The trigger generated by using the PTGCTRL 0x1111 broadcast trigger command can only operate in Pulse mode (i.e., PTGTOGL = ‘don’t care’).

4.10.1.1 TRIG Negation When PTGTOGL = 0

If generating an individual trigger output, and when the PTGTOGL bit (PTGCST<12>) = 0, or if generating a broadcast trigger output, the TRIG output(s) pulse width is determined by the PTGPWD<3:0> bits.

4.10.1.2 TRIG Negation When PTGTOGL = 1

If generating an individual trigger output and when the PTGTOGL bit (PTGCST<12>) = 1, the TRIG outputs will remain set until the PTGTRIG command is executed again. On start of the PTGTRIG command execution, the TRIG outputs are toggled at the beginning of the PTG execution clock.

Note: The PTGTOGL bit has no effect on the operation of the PTGCTRL 0x1111 multiple trigger (broadcast) generation command. The exception cases are as follows:

- The pulse width of all broadcast triggers is always determined by the PTGPWD<3:0> bits.
- If a target trigger output is already in the logic ‘1’ state (because PTGTOGL is active), the PTGCTRL 0x1111 command will have no effect and the trigger output will remain at logic ‘1’.

4.11 Stopping the Sequencer

When the PTG module is disabled (PTGEN = 0), the PTG clocks are disabled (except the trigger pulse counter), the Sequencer stops execution and the module enters its lowest power state. The PTGSTRT, PTGSWT, PTGWDTO and PTGQPTR<4:0> bits are all reset. All other bits and registers are not modified. All of the control registers can be read or written when PTGEN = 0.

When the PTGEN bit is cleared, a command that is underway is immediately aborted if the command is waiting for any of the following actions:

- An input from another source
- A timer match
- The Step delay to expire (for more information, refer to [Section 4.7.5 “Step Command Delay”](#))

All other commands are allowed to complete before the PTG module is disabled.

When the PTG module is halted, all of the control registers remain in their present state. The PTG module can be halted by the user by clearing the PTGSTRT bit, or in the event of a Watchdog Timer time-out, which also clears the PTGSTRT bit. Refer to [Section 4.8 “PTG Watchdog Timer”](#) for more information.

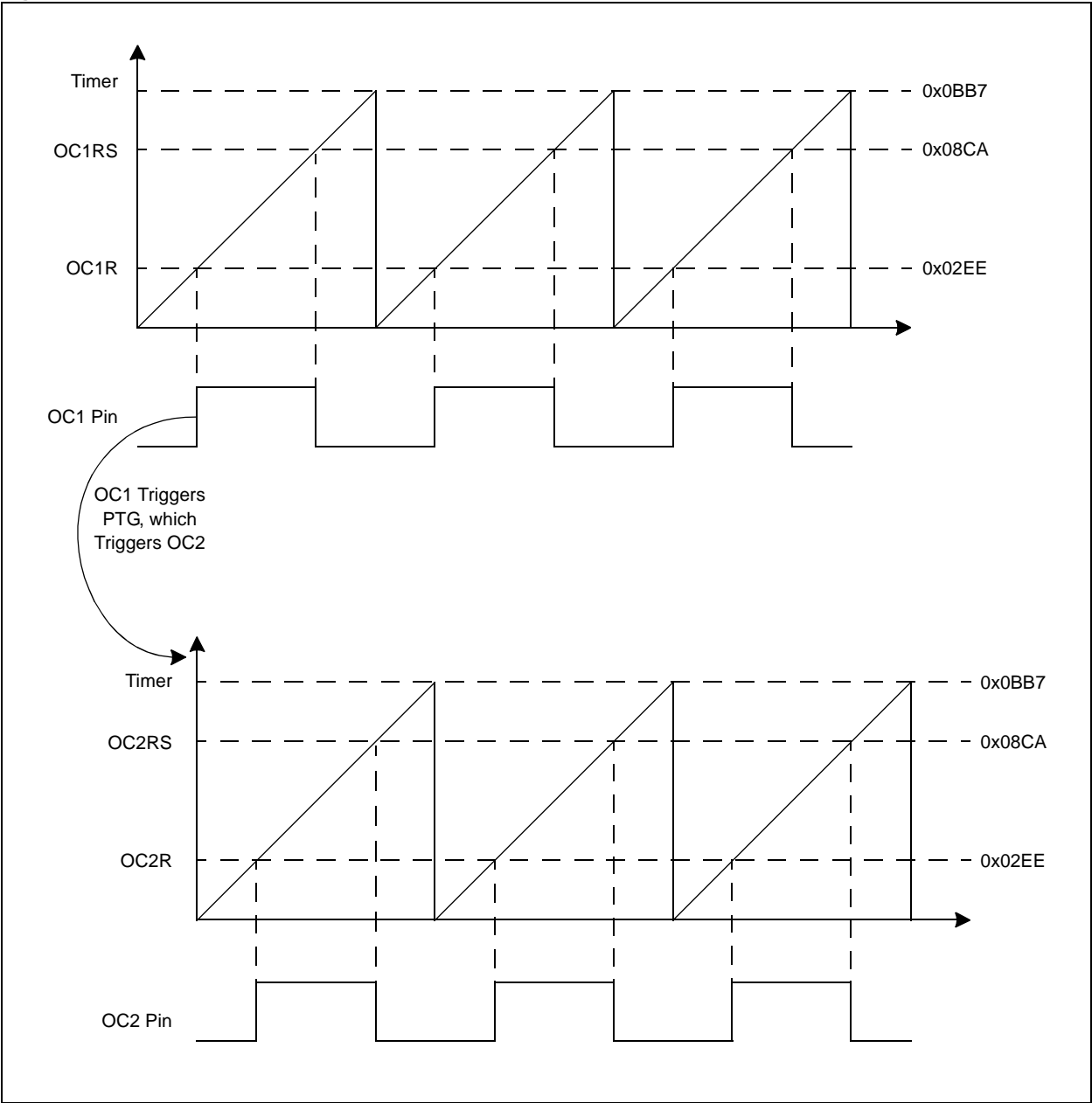
5.0 APPLICATION EXAMPLES

5.1 Generating Phase-Shifted Waveforms

Figure 5-1 shows an application example where the user needs to generate phase-shifted waveforms. In this example, Output Compare 1 uses Timer2 as the synchronization source to generate a pulse. The rising edge of the pulse is the trigger input to the PTG module. When the trigger is sensed, the PTG module triggers Output Compare 2, which uses the PTG module as the synchronization source to generate a phase-shifted waveform.

Note: This application example is only applicable to devices that have output compare and Timer2 peripherals. Refer to the device-specific data sheet for availability.

Figure 5-1: Phase-Shifted Waveform Example Application



Peripheral Trigger Generator (PTG)

Example 5-1 shows code for generating a phase-shifted waveform.

Example 5-1: Generating Phase-Shifted Waveforms

```
#include <xc.h>

_FOSCSEL(FNOSC_FRC);
_FOSC(FCKSM_CSECMD & POSCMD_XT & OSCIOFNC_OFF & IOL1WAY_OFF);
_FWDT(FWDTEN_OFF);
_FPOR(ALT12C1_ON & ALT12C2_ON);
_FICD(ICS_PGD2 & JTAGEN_OFF);

void Init_Timer(void);
void Init_Ptg(void);
void Init_PPS(void);
void Init_OC1(void);
void Init_OC2(void);

int main(void)
{
    // Configure the device PLL to obtain 60 MIPS operation. The crystal frequency is 8 MHz.
    // Divide 8 MHz by 2, multiply by 60 and divide by 2. This results in Fosc of 120 MHz.
    // The CPU clock frequency is Fcy = Fosc/2 = 60 MHz.
    PLLFBD = 58;                /* M = 30 */
    CLKDIVbits.PLLPOST = 0;      /* N1 = 2 */
    CLKDIVbits.PLLPRE = 0;      /* N2 = 2 */
    OSCTUN = 0;

    // Initiate Clock Switch to Primary
    // Oscillator with PLL (NOSC= 0x3)
    __builtin_write_OSCCONH(0x03);
    __builtin_write_OSCCONL(0x01);
    while (OSCCONbits.COSC != 0x3);
    while (_LOCK == 0);          /* Wait for PLL lock at 60 MIPS */

    Init_Timer();
    Init_Ptg();
    Init_PPS();
    Init_OC1();
    Init_OC2();

    PTGCSTbits.PTGEN = 1;        /* Enable the PTG */
    PTGCSTbits.PTGSTRT = 1;      /* Start the PTG */
    T2CONbits.TON = 1;           /* Start the timer */

    while(1);
}

void Init_Timer( void )
{
    // Initialize and enable Timer2
    T2CON = 0x0000;              /* Timer reset */
    TMR2 = 0x0000;              /* Clear timer register */
    PR2 = 0x0BB7;               /* Load the period value */
}

void Init_Ptg( void )
{
    PTGCST = 0; // Clear the control/status register
    PTGCON = 0; // Clear the control register

    /* Program the command sequence */
    _STEP0 = PTGWHI(0x7);        /* Wait for OC1 input trigger event */
    _STEP1 = PTGTRIG(0x1);       /* Trigger PTG02 (trig/sync for OC2) */
    _STEP2 = PTGJMP(0x0);        /* Jump to _STEP0 */
}
```

See [Example 3-1](#) for PTG command definitions. See [Example 3-2](#) and [Example 3-3](#) for PTGCTRL, PTGADD and PTGCOPY command options.

dsPIC33/PIC24 Family Reference Manual

Example 5-1: Generating Phase-Shifted Waveforms (Continued)

```
void Init_PPS(void)
{
    _RP39R = 0x10;           // Set up the PPS for OC1
    _RP40R = 0x11;           // Set up the PPS for OC2
}

void Init_OC1(void)
{
    OC1R = 0x02EE;           // Initialize the compare register
    OC1RS = 0x08CA;          // Initialize the secondary compare register

    // Initialize Output Compare Module
    OC1CON1 = 0x0;           // Clear all control bits
    OC1CON2 = 0x0;           // Clear all control bits
    OC1CON1bits.OCTSEL = 0x7; // Select peripheral clock as clock source
    OC1CON2bits.SYNCSEL = 0xC; // Select Timer2 as sync source
    OC1CON1bits.OCM = 0x5;    // Double compare continuous pulse mode
}

void Init_OC2(void)
{
    OC2R = 0x02EE;           // Initialize the compare register
    OC2RS = 0x08CA;          // Initialize the secondary compare register

    // Initialize Output Compare Module
    OC2CON1 = 0x0;           // Clear all control bits
    OC2CON2 = 0x0;           // Clear all control bits
    OC2CON1bits.OCTSEL = 0x7; // Select peripheral clock as clock source
    OC2CON2bits.SYNCSEL = 0xA; // Select PTG as sync source
    OC2CON1bits.OCM = 0x5;    // Double compare continuous pulse mode
}
```

Peripheral Trigger Generator (PTG)

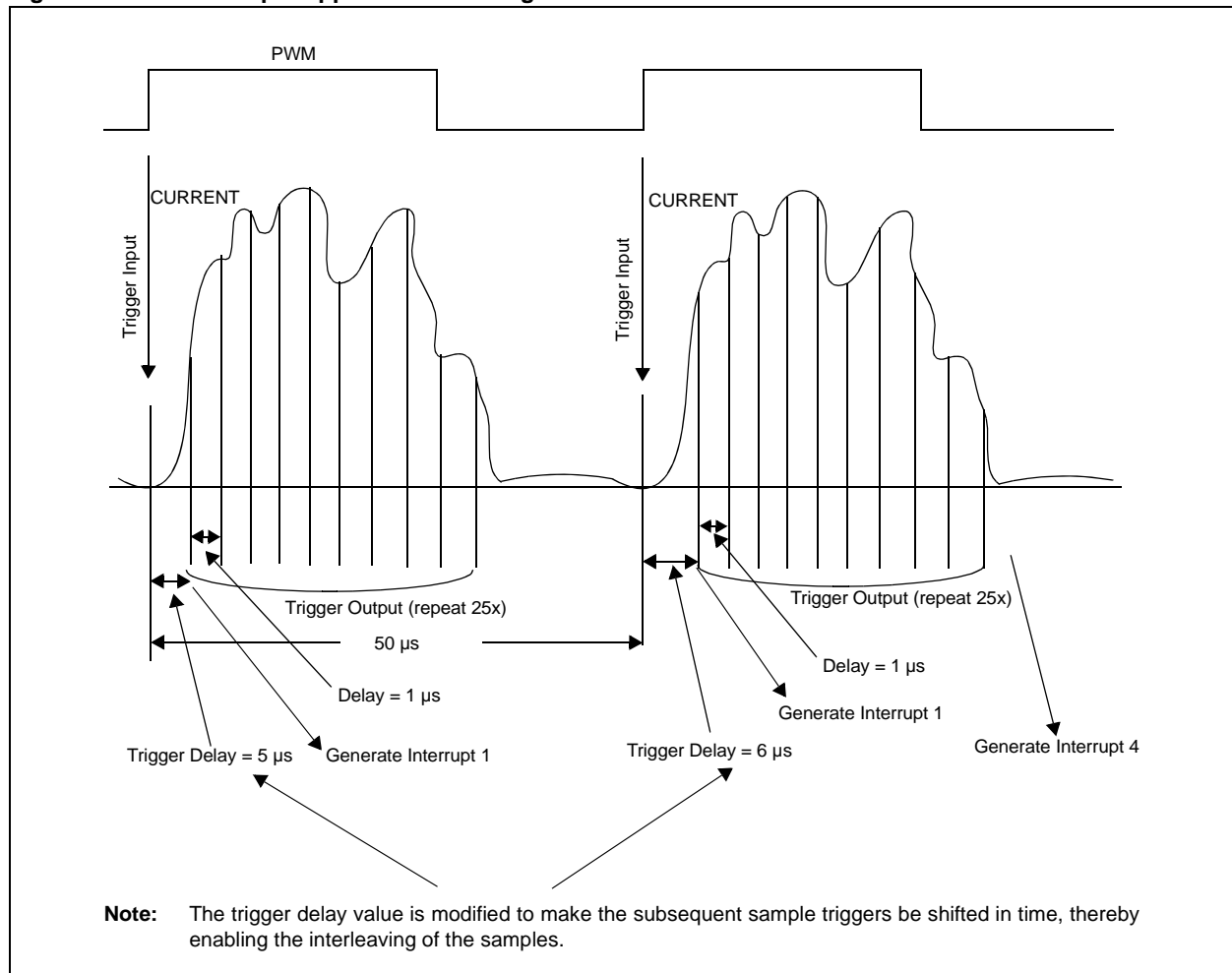
5.2 Interleaving Samples Over Multiple Cycles

Figure 5-2 shows the waveforms of an application where the user needs to accurately measure the power in a system where the current load is highly dependent on temperature, voltage and user application. The current waveforms vary widely per user usage, but over a few PWM cycles, the waveforms are relatively stable.

The aim of this example is to collect many current and/or voltage readings over several PWM cycles in an interleaved manner. The data is stored in the memory during acquisition and is later processed (integrated) to yield an accurate power value.

This example shows a situation where it would not be practical or possible for software to accurately schedule the ADC samples.

Figure 5-2: Example Application – Average Power Calculation



5.3 Interleaved Sampling Step Command Program

This section describes the Step command programming for implementing the timing sequence shown in [Figure 5-2](#).

The following assumptions are made:

1. Trigger Input 1 is connected to the PWM signal. The rising edge of the PWM signal starts the sequence.
2. Output Trigger 3 is connected to the ADC module. This signal gives command to the ADC module to begin a sample and conversion process.
3. Interrupt 1 is used to indicate to the processor that a subsequence has started (provides status).
4. Interrupt 4 is used to indicate to the processor that the entire sequence has completed.
5. The ADC clock is selected as the PTG clock source.
6. The ADC clock is 14 MHz.
7. The initial trigger delay is 5 μ s.
8. The second trigger delay is 6 μ s.
9. In each PWM cycle, the ADC will be triggered 25 times.
10. The basic sequence is executed twice.

Initialize the following control registers:

- PTGT0LIM = 0x0046 (5 μ s x 14 clocks/ μ s)
- PTGT1LIM = 0x000B ([1 μ s x 14 clocks/ μ s] – 3 Step Clocks)
- PTGC0LIM = 0x0018 (total of 25 inner loop iterations)
- PTGC1LIM = 0x0001 (total of two outer loop iterations)
- PTGHOLD = 0x0046 (5 μ s x 14 clocks/ μ s)
- PTGADJ = 0x000E (1 μ s x 14 clocks/ μ s)
- PTGSDLIM = 0x0000 (no Step delay)
- PTGBTE = 0x0000 (no broadcast triggers)
- PTGQPTR = 0x0000 (start of Step queue)
- PTGCST = 0x8200 (after PTGQPTR is initialized)

Example 5-2: Step Commands in PTGQUEn

```
void PTG_InterleavedSamplingQueue(void)
{
    // Outer loop
    _STEP0 = PTGWHI(0x1);           // Wait for positive edge trigger 1
    _STEP1 = PTGCTRL(t0Wait);       // Start PTGT0, wait for time out
    _STEP2 = PTGIRQ(0x1);           // Generate IRQ 1
    // Inner loop
    _STEP3 = PTGTRIG(0x3);          // Generate output trigger 3
    _STEP4 = PTGCTRL(t1Wait);       // Start PTGT1, wait for time out
    _STEP5 = PTGJMPC0(0x3);         // Go to STEP3 if PTGC0 != PTGC0LIM, increment PTGC0
    // End inner loop
    _STEP6 = PTGADD(t0Limit);        // Add PTGADJ to PTGT0LIM
    _STEP7 = PTGJMPC1(0x0);
    // End outer loop

    _STEP8 = PTGIRQ(0x4);           // Generate IRQ 4
    _STEP9 = PTGCOPY(0x8);          // Copy PTGHOLD to PTGT0LIM (restore original value)
    _STEP10 = PTGJMP(0x0);          // Jump to start of queue
}
```

See [Example 3-1](#) for PTG command definitions. See [Example 3-2](#) and [Example 3-3](#) for PTGCTRL, PTGADD and PTGCOPY command options.

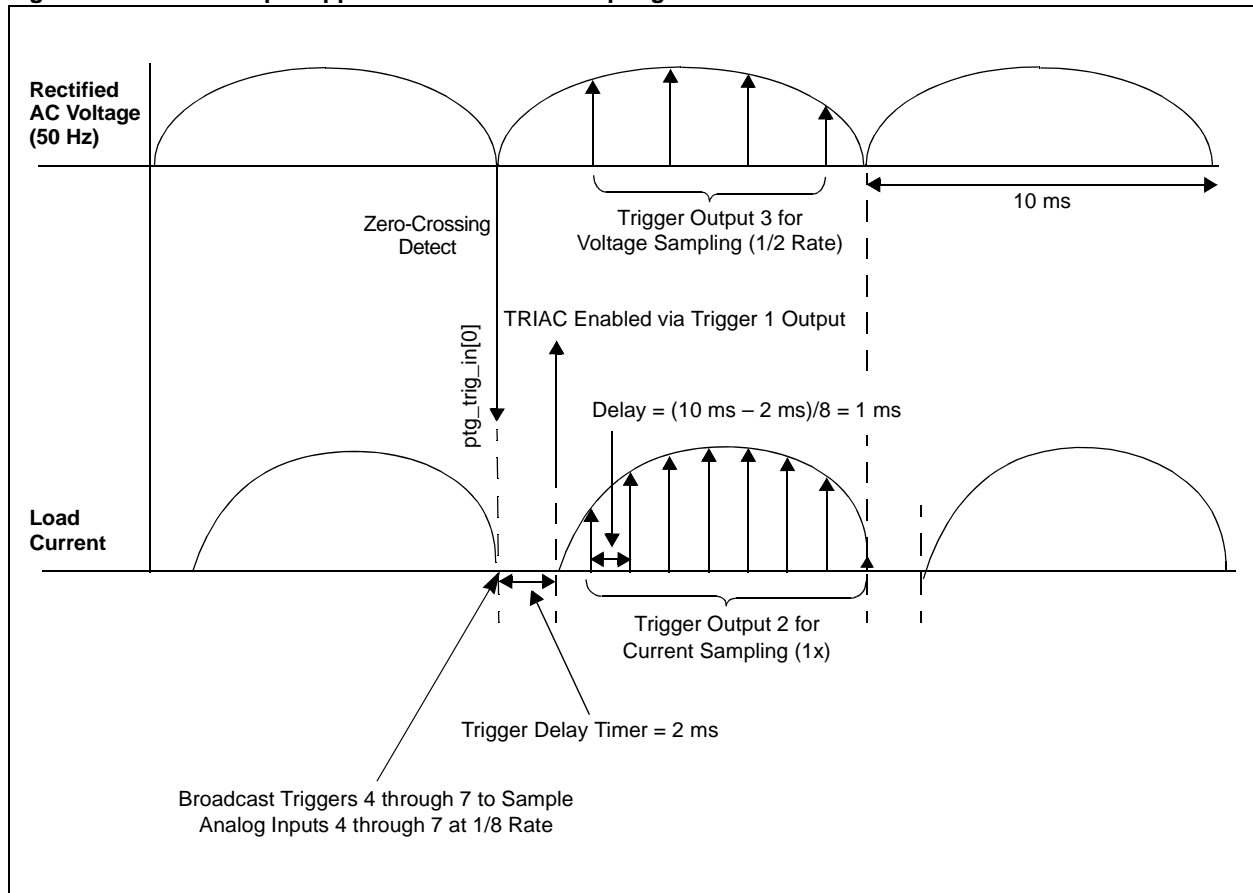
Peripheral Trigger Generator (PTG)

5.4 Sampling at Multiple Rates

Figure 5-3 shows an application example wherein the aim is to sample an ADC input at a fast rate (1x rate), a second analog input at a slower rate (1/2 rate) and Analog Inputs 4 through 7 at a 1/8 rate. The example is of a motor control application using a Silicon Controlled Rectifier (SCR) that triggers at a specified time after the AC line zero crossing.

While this example uses the simple binary sampling rates, the PTG module can generate a very wide range of sample ratios to meet the requirements of an application.

Figure 5-3: Example Application – Ratioed Sampling



5.5 Ratioed Sampling Step Command Program

This section describes the Step command programming for implementing the timing sequence shown in [Figure 5-3](#).

The following assumptions are made:

- Trigger Input 0 is connected to the zero-crossing detect. The rising edge of the zero-crossing detect signal starts the sequence
- The trigger delay from Trigger Input 0 to the generation of Trigger Output 1 is 2 ms
- Trigger Output 1 enables the SCR in the application circuit
- Trigger Output 2 is connected to the ADC to trigger sampling of the current measurement at 1 ms intervals
- Trigger Output 3 is connected to the ADC to trigger sampling of the supply voltage measurement at 2 ms intervals
- Trigger Outputs 4, 5, 6 and 7 are connected to the ADC to sample other data values once per cycle
- The ADC clock is selected as the PTG clock source
- The ADC clock is 14 MHz

Initialize the following control registers:

- PTGT0LIM = 0x6D60 (2 ms x 14 clocks/ μ s)
- PTGT1LIM = 0x36B0 (1 ms x 14 clocks/ μ s)
- PTGC0LIM = 0x0018 (total of 25 inner loop iterations)
- PTGC1LIM = 0x0001 (total of two outer loop iterations)
- PTGHOLD = 0x0000 (not used)
- PTGADJ = 0x0000 (not used)
- PTGSDLIM = 0x0000 (no Step delay)
- PTGBTE = 0x00F0 (enable Broadcast Triggers 4-7)
- PTGQPTR = 0x0000 (start of Step queue)
- PTGCST = 0x8200 (after PTGQPTR is initialized)

Peripheral Trigger Generator (PTG)

6.0 POWER-SAVING MODES

The PTG module supports three power-saving modes:

- Disabled: The PTG module is not clocked in this mode
- Idle: The processor core and selected peripherals are shut down
- Sleep: The entire device is shut down

6.1 Disabled Mode

When PTGEN = 1, the module is considered in an active mode and is fully powered and functional. When PTGEN = 0, the module is turned off. The PTG clock portions of the circuit are disabled for maximum current savings. Only the control registers remain functional for reading and writing to allow the software to change the module's operational mode. The module sequencer is reset.

6.2 Idle Mode

To continue full module operation while the PTG module is in Idle mode, the PTGSIDL bit must be cleared prior to entry into Idle mode. If PTGSIDL = 1, the module will behave the same way in Idle mode as it does in Sleep mode.

6.3 Sleep Mode

If the PTG module enters Sleep mode while the module is enabled (PTGEN = 1), the module will be suspended in its current state until clock execution resumes. This situation should be avoided as it might result in unexpected operation. It is recommended that all peripherals be shut down in an orderly manner prior to entering Sleep mode.

dsPIC33/PIC24 Family Reference Manual

7.0 RELATED APPLICATION NOTES

This section lists application notes that are related to this section of the manual. These application notes may not be written specifically for the dsPIC33/PIC24 product families, but the concepts are pertinent and could be used with modification and possible limitations. The current application notes related to the Peripheral Trigger Generator (PTG) module include the following:

Title	Application Note #
No application notes at this point of time.	N/A

Note: Please visit the Microchip web site (www.microchip.com) for additional Application Notes and code examples for the dsPIC33/PIC24 families of devices.

Peripheral Trigger Generator (PTG)

8.0 REVISION HISTORY

Revision A (September 2011)

This is the initial released version of this document.

Revision B (August 2017)

- Sections:
 - Moved register map to the front of the document to replace the “**Status and Control Registers**” section.
 - Updated [Section 1.0 “Introduction”](#), [Section 3.0 “Step Commands and Format”](#), [Section 4.0 “Module Operation”](#), [Section 4.1 “PTG Description”](#), [Section 4.2 “PTG Clock Selection”](#), [Section 4.2.3 “Module Enable Delay”](#), [Section 4.2.1 “Clock Source Selection”](#), [Section 4.3 “Basic Operation”](#), [Section 4.4 “Control Register Access”](#), [Section 4.6 “Command Looping Control”](#), [Section 4.7 “Sequencer Operation”](#), [Section 4.7.4 “Wait for GP Timer”](#), [Section 4.8 “PTG Watchdog Timer”](#), [Section 4.10 “Strobe Output”](#), [Section 4.10.1 “Output Timing”](#), [Section 4.10.1.2 “TRIG Negation When PTGTOGL = 1”](#), [Section 4.11 “Stopping the Sequencer”](#) and [Section 5.1 “Generating Phase-Shifted Waveforms”](#).
 - Removed **Section 4.8 “Step Commands”**.
- Figures:
 - Updated [Figure 1-1](#).
 - Added [Figure 4-1](#).
 - Removed **Figure 4-9: “GP Timer”** and **Figure 4-11: “Literal ‘0’”**.
- Examples:
 - Updated [Example 5-2](#).
 - Removed **Example 32-1: “Generating Phase-Shifted Waveforms”**.
- Tables:
 - Updated [Table 2-1](#), [Table 3-1](#) and [Table 3-2](#).
- Registers:
 - Updated [Register 2-1](#), [Register 2-2](#), [Register 2-3](#), [Register 2-4](#) and [Register 2-14](#).

Minor grammatical corrections throughout the document.

dsPIC33/PIC24 Family Reference Manual

NOTES:

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELoQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
== ISO/TS 16949 ==

Trademarks

The Microchip name and logo, the Microchip logo, AnyRate, AVR, AVR logo, AVR Freaks, BeaconThings, BitCloud, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, Helder, JukeBlox, KEELoQ, KEELoQ logo, Klear, LANCheck, LINK MD, maxStylus, maxTouch, MediaLB, megaAVR, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, Prochip Designer, QTouch, RightTouch, SAM-BA, SpyNIC, SST, SST Logo, SuperFlash, tinyAVR, UNI/O, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

ClockWorks, The Embedded Control Solutions Company, EtherSynch, Hyper Speed Control, HyperLight Load, IntelliMOS, mTouch, Precision Edge, and Quiet-Wire are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, BodyCom, chipKIT, chipKIT logo, CodeGuard, CryptoAuthentication, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, EtherGREEN, In-Circuit Serial Programming, ICSP, Inter-Chip Connectivity, JitterBlocker, KlearNet, KlearNet logo, Mindi, MiWi, motorBench, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICKit, PICtail, PureSilicon, QMatrix, RightTouch logo, REAL ICE, Ripple Blocker, SAM-ICE, Serial Quad I/O, SMART-I.S., SQI, SuperSwitcher, SuperSwitcher II, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

Silicon Storage Technology is a registered trademark of Microchip Technology Inc. in other countries.

GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are property of their respective companies.

© 2011-2017, Microchip Technology Incorporated, All Rights Reserved.

ISBN: 978-1-5224-1994-5

Worldwide Sales and Service

AMERICAS

Corporate Office
 2355 West Chandler Blvd.
 Chandler, AZ 85224-6199
 Tel: 480-792-7200
 Fax: 480-792-7277
 Technical Support:
<http://www.microchip.com/support>
 Web Address:
www.microchip.com

Atlanta
 Duluth, GA
 Tel: 678-957-9614
 Fax: 678-957-1455

Austin, TX
 Tel: 512-257-3370

Boston
 Westborough, MA
 Tel: 774-760-0087
 Fax: 774-760-0088

Chicago
 Itasca, IL
 Tel: 630-285-0071
 Fax: 630-285-0075

Dallas
 Addison, TX
 Tel: 972-818-7423
 Fax: 972-818-2924

Detroit
 Novi, MI
 Tel: 248-848-4000

Houston, TX
 Tel: 281-894-5983

Indianapolis
 Noblesville, IN
 Tel: 317-773-8323
 Fax: 317-773-5453
 Tel: 317-536-2380

Los Angeles
 Mission Viejo, CA
 Tel: 949-462-9523
 Fax: 949-462-9608
 Tel: 951-273-7800

Raleigh, NC
 Tel: 919-844-7510

New York, NY
 Tel: 631-435-6000

San Jose, CA
 Tel: 408-735-9110
 Tel: 408-436-4270

Canada - Toronto
 Tel: 905-695-1980
 Fax: 905-695-2078

ASIA/PACIFIC

Asia Pacific Office
 Suites 3707-14, 37th Floor
 Tower 6, The Gateway
 Harbour City, Kowloon

Hong Kong
 Tel: 852-2943-5100
 Fax: 852-2401-3431

Australia - Sydney
 Tel: 61-2-9868-6733
 Fax: 61-2-9868-6755

China - Beijing
 Tel: 86-10-8569-7000
 Fax: 86-10-8528-2104

China - Chengdu
 Tel: 86-28-8665-5511
 Fax: 86-28-8665-7889

China - Chongqing
 Tel: 86-23-8980-9588
 Fax: 86-23-8980-9500

China - Dongguan
 Tel: 86-769-8702-9880

China - Guangzhou
 Tel: 86-20-8755-8029

China - Hangzhou
 Tel: 86-571-8792-8115
 Fax: 86-571-8792-8116

China - Hong Kong SAR
 Tel: 852-2943-5100
 Fax: 852-2401-3431

China - Nanjing
 Tel: 86-25-8473-2460
 Fax: 86-25-8473-2470

China - Qingdao
 Tel: 86-532-8502-7355
 Fax: 86-532-8502-7205

China - Shanghai
 Tel: 86-21-3326-8000
 Fax: 86-21-3326-8021

China - Shenyang
 Tel: 86-24-2334-2829
 Fax: 86-24-2334-2393

China - Shenzhen
 Tel: 86-755-8864-2200
 Fax: 86-755-8203-1760

China - Wuhan
 Tel: 86-27-5980-5300
 Fax: 86-27-5980-5118

China - Xian
 Tel: 86-29-8833-7252
 Fax: 86-29-8833-7256

ASIA/PACIFIC

China - Xiamen
 Tel: 86-592-2388138
 Fax: 86-592-2388130

China - Zhuhai
 Tel: 86-756-3210040
 Fax: 86-756-3210049

India - Bangalore
 Tel: 91-80-3090-4444
 Fax: 91-80-3090-4123

India - New Delhi
 Tel: 91-11-4160-8631
 Fax: 91-11-4160-8632

India - Pune
 Tel: 91-20-3019-1500

Japan - Osaka
 Tel: 81-6-6152-7160
 Fax: 81-6-6152-9310

Japan - Tokyo
 Tel: 81-3-6880-3770
 Fax: 81-3-6880-3771

Korea - Daegu
 Tel: 82-53-744-4301
 Fax: 82-53-744-4302

Korea - Seoul
 Tel: 82-2-554-7200
 Fax: 82-2-558-5932 or
 82-2-558-5934

Malaysia - Kuala Lumpur
 Tel: 60-3-6201-9857
 Fax: 60-3-6201-9859

Malaysia - Penang
 Tel: 60-4-227-8870
 Fax: 60-4-227-4068

Philippines - Manila
 Tel: 63-2-634-9065
 Fax: 63-2-634-9069

Singapore
 Tel: 65-6334-8870
 Fax: 65-6334-8850

Taiwan - Hsin Chu
 Tel: 886-3-5778-366
 Fax: 886-3-5770-955

Taiwan - Kaohsiung
 Tel: 886-7-213-7830

Taiwan - Taipei
 Tel: 886-2-2508-8600
 Fax: 886-2-2508-0102

Thailand - Bangkok
 Tel: 66-2-694-1351
 Fax: 66-2-694-1350

EUROPE

Austria - Wels
 Tel: 43-7242-2244-39
 Fax: 43-7242-2244-393

Denmark - Copenhagen
 Tel: 45-4450-2828
 Fax: 45-4485-2829

Finland - Espoo
 Tel: 358-9-4520-820

France - Paris
 Tel: 33-1-69-53-63-20
 Fax: 33-1-69-30-90-79

France - Saint Cloud
 Tel: 33-1-30-60-70-00

Germany - Garching
 Tel: 49-8931-9700

Germany - Haan
 Tel: 49-2129-3766400

Germany - Heilbronn
 Tel: 49-7131-67-3636

Germany - Karlsruhe
 Tel: 49-721-625370

Germany - Munich
 Tel: 49-89-627-144-0
 Fax: 49-89-627-144-44

Germany - Rosenheim
 Tel: 49-8031-354-560

Israel - Ra'anana
 Tel: 972-9-744-7705

Italy - Milan
 Tel: 39-0331-742611
 Fax: 39-0331-466781

Italy - Padova
 Tel: 39-049-7625286

Netherlands - Drunen
 Tel: 31-416-690399
 Fax: 31-416-690340

Norway - Trondheim
 Tel: 47-7289-7561

Poland - Warsaw
 Tel: 48-22-3325737

Romania - Bucharest
 Tel: 40-21-407-87-50

Spain - Madrid
 Tel: 34-91-708-08-90
 Fax: 34-91-708-08-91

Sweden - Gothenberg
 Tel: 46-31-704-60-40

Sweden - Stockholm
 Tel: 46-8-5090-4654

UK - Wokingham
 Tel: 44-118-921-5800
 Fax: 44-118-921-5820