12-Bit Programmable Interval Timer Datasheet PITIMER12 V 1.1

Resources	API Memory (Bytes)		Pins (per External I/O)
	flash	RAM	
CY7C639/638/633/601/602xx, CYRF69xx3	57 Bytes	0	None

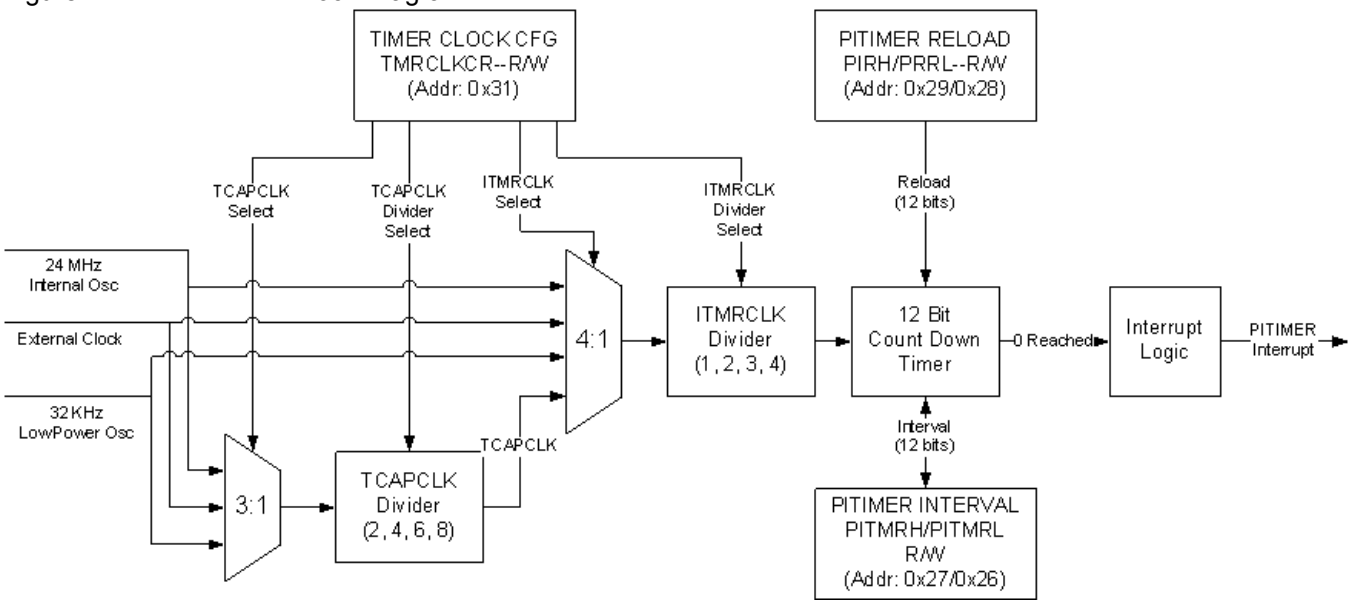
Features and Overview

- 12-bit programmable interval timer
- Source clock rates up to 24 MHz
- Automatic reload of period on terminal count

Functional Description

The 12-Bit Programmable Interval Timer User Module provides convenient access to the Programmable Interval Timer feature of the enCoRe II.

Figure 1. PITIMER12 Block Diagram



DC and AC Electrical Characteristics

See the device datasheet for your enCoRe II device for the electrical characteristics of the Programmable Interval Timer feature of the enCoRe II.

Placement

The PITIMER12 User Module occupies the ITMRCLK block of the enCoRe II device. Alternate placement is not available.

Parameters and Resources

PI-TIMER Source

Choose one of the available timer sources for the PI-Timer:

- Internal (24 MHz) timer
- External clock source
- Low power (32 kHz) oscillator
- TCAP clock divider

PI-TIMER Divider

Choose to divide the input clock by 1, 2, 3, or 4.

Application Programming Interface

The Application Programming Interface (API) routines are provided as part of the user module to enable you to deal with the module at a higher level. This section specifies the interface to each function together with related constants provided by the "include" files.

The API programming routines provided for PITIMER12 are given in this section.

User code can be added to the interrupt service routine found in PITIMER12INT.ASM in the project library directory.

PITIMER12_Start

Description:

Starts the PITIMER12 operation by loading the timer reload register. The Count register is decremented on the next clock cycle.

C Prototype:

```
void  PITIMER12_Start(WORD wReload);
```

Assembly:

```
reload:equ        1024
movx, <reload
mova, >reload
lcall  PITIMER12_Start
```

Parameters:

wReload: 12-bit reload value

Return Value:

None

Side Effects:

The A and X registers may be altered by this function.

PITIMER12_Stop

Description:

Stops the PITIMER12 operation. This function also disables the timer interrupt.

C Prototype:

```
void  PITIMER12_Stop(void);
```

Assembly:

```
lcall  PITIMER12_Stop
```

Parameters:

None

Return Value:

None

Side Effects:

The A and X registers may be altered by this function.

PITIMER12_EnableInt

Description:

Enables the interrupt mode operation. However, note that the global interrupts must also be enabled before the interrupts are actually serviced.

C Prototype:

```
void  PITIMER12_EnableInt(void);
```

Assembly:

```
lcall  PITIMER12_EnableInt
```

Parameters:

None

Return Value:

None

Side Effects:

This routine modifies the appropriate interrupt enable register in IO space. The A and X registers may be altered by this function.

PITIMER12_DisableInt

Description:

Disables the interrupt mode operation.

C Prototype:

```
void  PITIMER12_DisableInt(void);
```

Assembly:

```
lcall  PITIMER12_DisableInt
```

Parameters:

None

Return Value:

None

Side Effects:

This routine modifies the appropriate interrupt enable register in I/O space. The A and X registers may be altered by this function.

PITIMER12_SetReload

Description:

Sets the PITIMER12 Reload Registers.

C Prototype:

```
void  PITIMER12_SetReload(WORD wReload);
```

Assembly:

```
reload:equ        1024
movx, <reload
mova, >reload
lcall  PITIMER12_SetReload
```

Parameters:

wReload is the 12-bit reload value

Return Value:

None

Side Effects:

The A and X registers may be altered by this function.

PITIMER12_GetReload

Description:

Reads and returns the current PITIMER12 Reload Register values.

C Prototype:

```
WORD  PITIMER12_GetReload(void);
```

Assembly:

```
lcall  PITIMER12_GetReload
```

Parameters:

None

Return Value:

None

Side Effects:

The A and X registers may be altered by this function.

PITIMER12_GetInterval

Description:

Reads and returns the current PITIMER12 Interval Register values. Note: To avoid function call overhead, some applications may want to access the PITMRH/PITIMRL registers directly according the to device datasheet.

C Prototype:

```
WORD  PITIMER12_GetInterval(void);
```

Assembly:

```
lcall  PITIMER12_GetInterval
```

Parameters:

None

Return Value:

The current 12-bit interval value

Side Effects:

The A and X registers may be altered by this function.

Sample Firmware Source Code

The following code is an example of a using the PITIMER12 User Module. The code sample in assembly language is:

```
include "m8c.inc"           ; part specific constants and macros
include "memory.inc"    ; Constants & macros for SMM/LMM and Compiler
include "PSoCAPI.inc"   ; PSoC API definitions for all User Modules

RELOAD_VALUE:  equ 4095; Set timer interval to 4095 input cycles of the input clock

export _main

_main:

    M8C_EnableGInt                      ; Enable Global Interrupts
    mov   X, <RELOAD_VALUE      ;Load LSB of RELOAD_VALUE into X register
    mov   A, >RELOAD_VALUE      ;Load MSB of RELOAD_VALUE into A register
    lcall PITIMER12_Start             ; Start the user module operation
    lcall PITIMER12_EnableInt     ; Enable user module interrupts


.terminate:
    jmp .terminate
```

The same code written in C is:

```
#include <m8c.h>        // part specific constants and macros
#include "PSoCAPI.h"    // PSoC API definitions for all User Modules

#define RELOAD_VALUE 4095 // Set timer interval to 4095 input cycles of the input clock

void main(void)
{
M8C_EnableGInt ;                                    // Enable Global Interrupts
PITIMER12_Start(RELOAD_VALUE);    // Start the user module operation
PITIMER12_EnableInt();                          // Enable user module interrupts
}
```

Version History

Version	Originator	Description
1.1	DHA	Added Version History
1.1.b	DHA	Added Sample Firmware Source Code section to this user module datasheet.

Note PSoC Designer 5.1 introduces a Version History in all user module datasheets. This section documents high level descriptions of the differences between the current and previous user module versions.