Low Power Comparator Datasheet CMPLP V 1.0

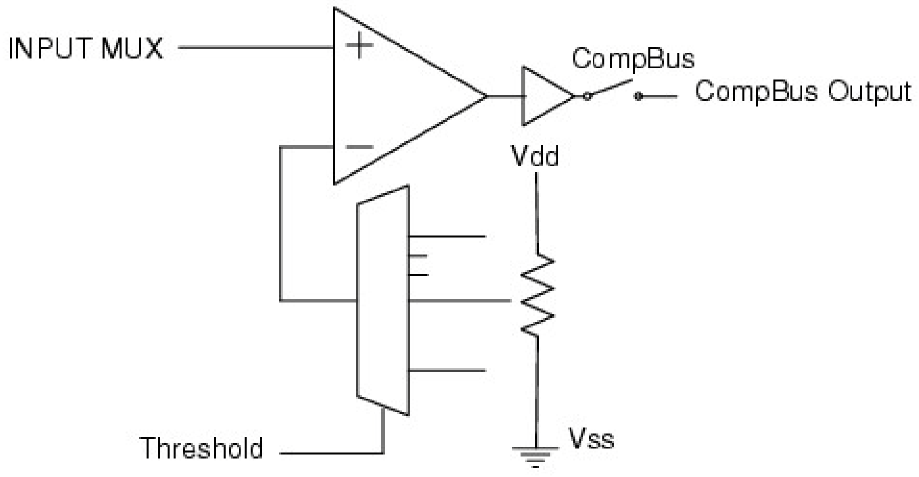
Resources	PSoC® Blocks			API Memory (Bytes)		Pins (per External I/O)
	Digital	Analog CT	Analog SC	flash	RAM	
CY8C29/27/24/22xxx, CY8C23x33, CY8CLED04/08/16, CY8CLED0xD, CY8CLED0xG, CY8CTST120, CY8CTMG120, CY8CTMA120, CY8C28x45, CY8CPLC20, CY8CLED16P01, CY8C28x43, CY8C28x52						
	0	1	0	73	0	1

Features and Overview

- Programmable threshold
- Direct connection to digital PSoC block and interrupt

The CmpLP User Module provides a comparison of an input from the input column MUX against a programmable reference threshold. Its only output is the column comparator bus.

Figure 1. CmpLP Block Diagram



Functional Description

The comparator is based on a low power comparator which is configured in parallel with the opamp in the continuous time block. When the CmpLP is used, the opamp is disabled. The positive input is connected to the input multiplexer. The negative input is connected to the tap of a resistive divider between Vdd and Vss.

The reference value of the comparator is determined by the following equation:

Equation 1

$$V_{Reference} = Vdd \cdot Threshold$$

The output polarity follows the inputs (that is, a positive input greater than the negative input to the comparator results in a positive output logic level).

The comparator logic output can be switched onto the CompBus to drive the enable inputs of digital blocks, the interrupt controller, and a register that can be read by the CPU.

Placement

The COMP block maps onto any of the continuous time PSoC blocks in the device. However, if the COMP AnalogBus output and the CompBus output are enabled onto their respective buses, care must be exercised to ensure that no other user module tries to drive the same buses.

Parameters and Resources

CompBus

The COMP block comparator output may be routed to the input bus of the digital PSoC blocks or to an interrupt. The CompBus must be enabled to make any of these connections. The CompBus parameter must be enabled to enable interrupts.

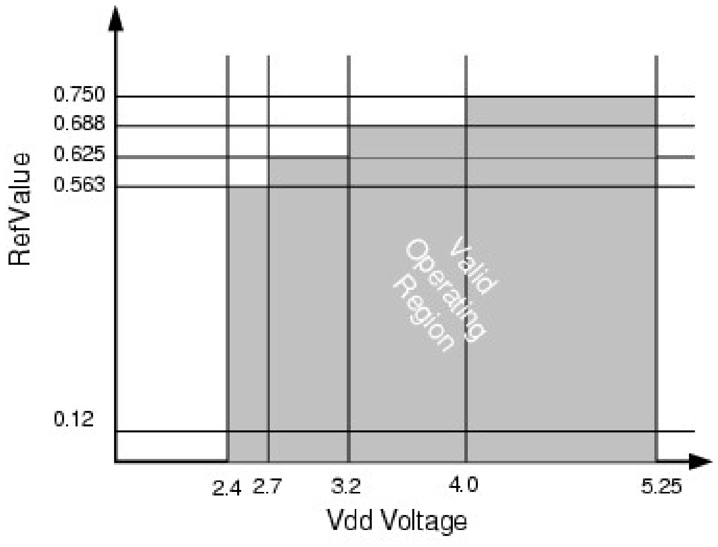
Note By default, comparator signals routed to the CompBus only generate interrupts when latched in on SysClk rising edges. If the PSoC is asleep, then its SysClk is not running, and comparator interrupts will never post. This latching may be bypassed using the CLDIS bits in the CMP_CR1 register, causing comparator interrupts to post during sleep.

RefValue

The RefValue is programmable in steps shown in the table below relative to Vdd. When operating from a 5.00V supply, this sets reference values of 0.105V to 3.75V.

Note that not all RefValues are allowed for all PSoC Vdd voltages. See the figure below for reference. Setting the wrong values can cause the comparator to function incorrectly.

Figure 2. Allowed RefValue versus Vdd Voltage



RefValue	Alowed Vdd range, V	
	Min	Max
0.750 Vdd	4.0	5.25
0.688 Vdd	3.2	5.25
0.625 Vdd	2.7	5.25
0.563 Vdd	2.4	5.25
0.500 Vdd	2.4	5.25
0.438 Vdd	2.4	5.25
0.375 Vdd	2.4	5.25
0.313 Vdd	2.4	5.25
0.250 Vdd	2.4	5.25
0.188 Vdd	2.4	5.25
0.125 Vdd	2.4	5.25
0.063 Vdd	2.4	5.25
0.042 Vdd	2.4	5.25
0.021 Vdd	2.4	5.25

Application Programming Interface

The Application Programming Interface (API) routines are provided as part of the user module to allow 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.

Note

In this, as in all user module APIs, the values of the A and X register may be altered by calling an API function. It is the responsibility of the calling function to preserve the values of A and X before the call if those values are required after the call. This “registers are volatile" policy was selected for efficiency reasons and has been in force since version 1.0 of PSoC Designer. The C compiler automatically takes care of this requirement. Assembly language programmers must ensure their code observes the policy, too. Though some user module API functions may leave A and X unchanged, there is no guarantee they will do so in the future.

For Large Memory Model devices, it is also the caller's responsibility to preserve any value in the CUR_PP, IDX_PP, MVR_PP, and MVW_PP registers. Even though some of these registers may not be modified now, there is no guarantee that will remain the case in future releases.

CmpLP_Start

Description:

Performs all required initialization for this user module. The comparator output is driven.

C Prototype:

```
void  CmpLP_Start(void)
```

Assembler:

```
lcall  CmpLP_Start
```

Parameters:

None

Return Value:

None

Side Effects:

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

CmpLP_Stop

Description:

Powers the user module off. The outputs are not driven.

C Prototype:

```
void  CmpLP_Stop(void)
```

Assembler:

```
lcall  CmpLP_Stop
```

Parameters:

None

Return Value:

None

Side Effects:

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

CmpLP_EnableInt

Description:

Enables interrupt mode operation.

C Prototype:

```
void  CmpLP_EnableInt(void)
```

Assembler:

```
lcall  CmpLP_EnableInt
```

Parameters:

None

Return Value:

None

Side Effects:

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

Note To achieve LPCMP interrupt generation during sleep, the comparator bus signal must be set to unlatched using the CLDIS bits in the CMP_CR1 register.

CmpLP_DisableInt

Description:

Disables interrupt mode operation.

C Prototype:

```
void  CmpLP_DisableInt(void)
```

Assembler:

```
lcall  CmpLP_DisableInt
```

Parameters:

None

Return Value:

None

Side Effects:

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

CmpLP_ClearInt

Description:

Clears a posted interrupt.

C Prototype:

```
void  CmpLP_ClearInt(void)
```

Assembler:

```
lcall  CmpLP_ClearInt
```

Parameters:

None

Return Value:

None

Side Effects:

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

CmpLP_PollInt

Description:

This function polls for a posted interrupt.

C Prototype:

```
BYTE  CmpLP_PollInt(void)
```

Assembler:

```
lcall  CmpLP_PollInt
mov    [bIntState], A
```

Parameters:

None

Return Value:

Returns a zero in the Accumulator if no posted interrupt is detected and a nonzero value when a posted interrupt is detected.

Side Effects:

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

CmpLP_PollOutput

Description:

This function checks the status of the comparator outs stored in CMP_CR0.

C Prototype:

```
BYTE  CmpLP_PollOutput(void)
```

Assembler:

```
lcall  CmpLP_PollOutput
mov    [bComparatorState], A
```

Parameters:

None

Return Value:

Returns a zero in the Accumulator when the comparator output is low. Returns a nonzero value otherwise.

Side Effects:

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

CmpLP_ChangeThreshold

Description:

Sets the threshold value for the comparator. Note that not all thresholds are allowed for all Vdd voltages. See the Threshold parameter description for reference. Setting the wrong values can cause the comparator to function incorrectly.

C Prototype:

```
void  CmpLP_ChangeThreshold(BYTE bThreshold)
```

Assembler:

```
mov   A, bThreshold
lcall  CmpLP_ChangeThreshold
```

Parameters:

bThreshold: Symbolic names are provided in C and assembly and their associated values are given in the following table. The values provide a comparator threshold at Threshold*Vdd.

Symbolic Name	Value	Reference at Vdd = 2.7V	Reference at Vdd = 3.3V	Reference at Vdd = 5.0V
CmpLP_REF0_750	B0h	Not Allowed	Not Allowed	3.750
CmpLP_REF0_688	A0h	Not Allowed	2.270	3.440
CmpLP_REF0_625	90h	1.688	2.063	3.125
CmpLP_REF0_563	80h	1.520	1.858	2.815
CmpLP_REF0_500	70h	1.350	1.650	2.500
CmpLP_REF0_438	60h	1.183	1.446	2.190
CmpLP_REF0_375	50h	1.013	1.238	1.875
CmpLP_REF0_313	40h	0.845	1.033	1.565
CmpLP_REF0_250	30h	0.675	0.825	1.250
CmpLP_REF0_188	20h	0.508	0.620	0.940
CmpLP_REF0_125	10h	0.338	0.413	0.625
CmpLP_REF0_063	00h	0.170	0.208	0.315
CmpLP_REF0_042	14h	0.113	0.139	0.210
CmpLP_REF0_021	04h	0.057	0.069	0.105

Return Value:

None

Side Effects:

Comparator output is driven. This may result in false readings from the comparator while the Threshold is being changed. The A and X registers may be altered by this function.

Sample Firmware Source Code

This sample code creates a comparator with the threshold set to 0.500, overriding the threshold value set in the PSoC Designer Device Editor.

```
;;-----------------------------------------------------------------
;; Sample Code for the CmpLP
;; Turn on power and set Threshold to 0.5Vdd.
;;-----------------------------------------------------------------

export _main

include "m8c.inc"
include "CmpLP.inc"

_main:

mov  A, CmpLP_REF0_500           ; specify comparator Threshold
call CmpLP_ChangeThreshold       ; update Threshold

call CmpLP_Start                 ; and turn Comparator on
; Place user code here
ret
```

The same project written in C is as follows.

```
//------------------------------------------------------------------------
//  Sample C Code for the CmpLP
//  Turn on power and set Threshold to 0.5Vdd.
//
//------------------------------------------------------------------------
#include <m8c.h>        // Part specific constants and macros
#include "PSoCAPI.h"    // PSoC API definitions for all User Modules

void main(void)
{
   CmpLP_ChangeThreshold(CmpLP_REF0_500);   // Set Comparator Threshold
   CmpLP_Start();                           // Turn it on
    // Place user code here
}
```

Configuration Registers

The basic topology of the comparator sets most of the bits in the register configuration for the analog CT block that is used. Specific inputs available for comparison and reference are determined by user module placement.

Table 1. Block COMP, Register: CR0

Bit	7	6	5	4	3	2	1	0
Value	Threshold				1	1	1	0

Threshold chosen sets the reference value as a percentage of Vdd and is set in PSoC Designer. It is modified by calling the ChangeThreshold entry point in the API.

Table 2. Block COMP, Register: CR1

Bit	7	6	5	4	3	2	1	0
Value	0	Comp Bus	1	0	0	0	0	1

CompBus determines whether the COMP PSoC block drives the comparator bus. The value of this bit is determined by the choice made in Device Editor subsystem.

Table 3. Block COMP, Register: CR2

Bit	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	0	0	0

Table 4. Block COMP, Register: CR3

Bit	7	6	5	4	3	2	1	0
Value	0	0	0	0	LPCMPEN	0	0	EXGAIN

LPCMPEN: Set to enable low power comparator mode. It is set to 0 after placement and set to 1 by the CmpLP_Start API routine. EXGAIN: This bit is set automatically when either the 0.042 or 0.021 compare values are selected.

Note To achieve LPCMP interrupt generation during sleep, the comparator bus signal must be set to unlatched using the CLDIS bits in the CMP_CR1 register.

Version History

Version	Originator	Description
1.0	DHA	Initial version

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.