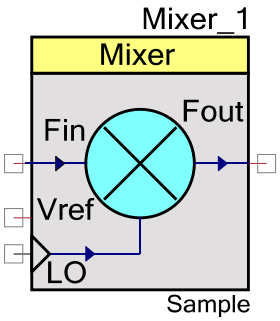
Mixer
1.91

Features



- Single-ended mixer
- Continuous-time up mixing:
  - Input frequencies up to 500 kHz
  - Sample clock up to 1 MHz
- Discrete-time, sample-and-hold down mixing:
  - Input frequencies up to 14 MHz
  - Sample clock up to 4 MHz
- Adjustable power settings
- Selectable reference voltage

General Description

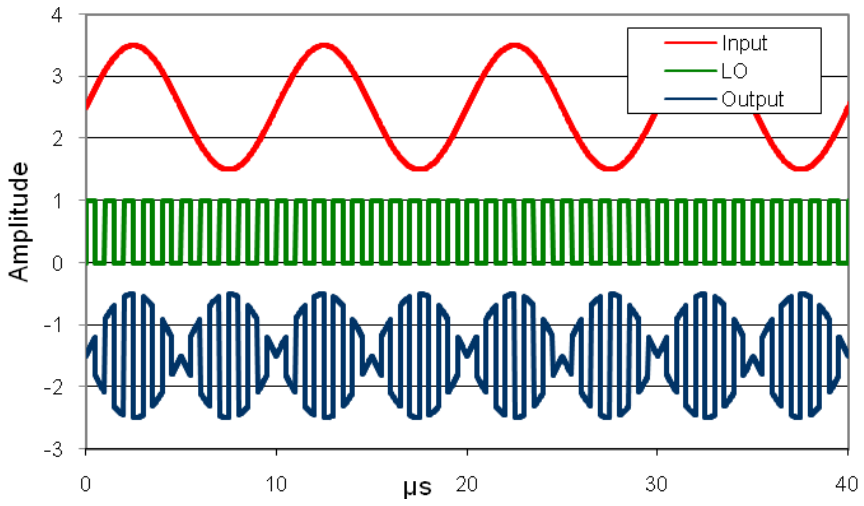
The Mixer component provides a single-ended modulator. The Mixer component can be used for frequency conversion of an input signal using a fixed Local Oscillator (LO) signal as the sampling clock. The manipulations of signal frequencies that a mixer performs can be used to move signals between frequency bands or to encode and decode signals. A mixer can be used to convert signal power at one frequency into power at another frequency to make signal processing easier, typically shifting higher frequencies to baseband. The mixer output is best used by filtering the desired signal harmonics using an off-chip filter. Alternatively, the output can be used to drive an on-chip ADC through internal routing. The component offers two configurations:

- Up mixer, continuous-time balance mixer, operates as a switching multiplier
- Down mixer, discrete-time, sample-and-hold mixer

The component accepts two signals at different frequencies as inputs and outputs a mixture of signals at multiple frequencies, including the sum and difference of the input signal and the local oscillator signal. Typically, the unwanted frequency components in the output signal are removed by filtering. A few examples illustrate the operation of the mixer in different modes.

Up Mixer: LO frequency greater than signal frequency

Shown with 100-kHz sine wave input, modulated by a 1.0-MHz Local Oscillator

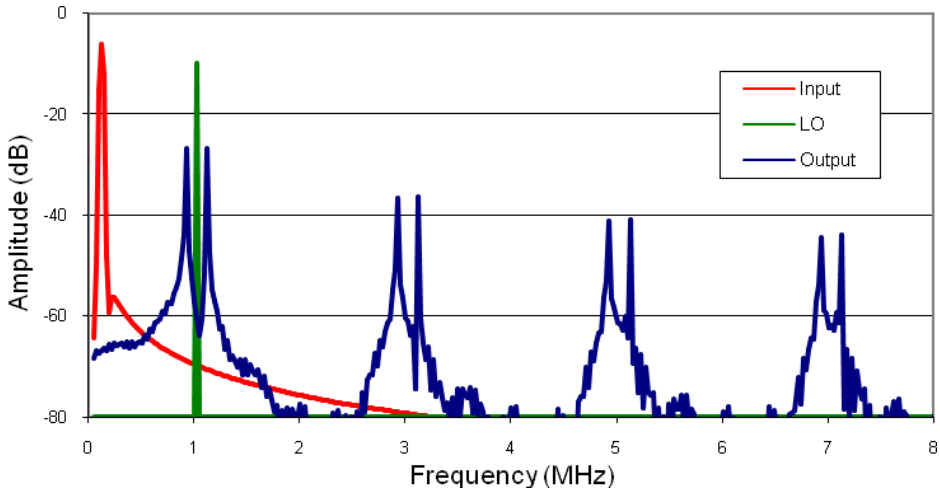


The Up mixer is a multiplier. For signal frequency at F_{SIG} and clock at F_{LO} it generates a modulated signal that is the product of the input and the LO. Because the LO is a square wave, with all of its expected harmonics, the output has the form

$$F_{MOD}(t) = \sin(2\pi F_{SIG}) \sum_{n=odd} \frac{1}{n} \sin(2\pi F_{LO} t)$$

$$F_{MOD}(t) = \frac{1}{2} \sum [\cos(2\pi(nF_{LO} - F_{SIG})) - \cos(2\pi(nF_{LO} + F_{SIG}))]$$

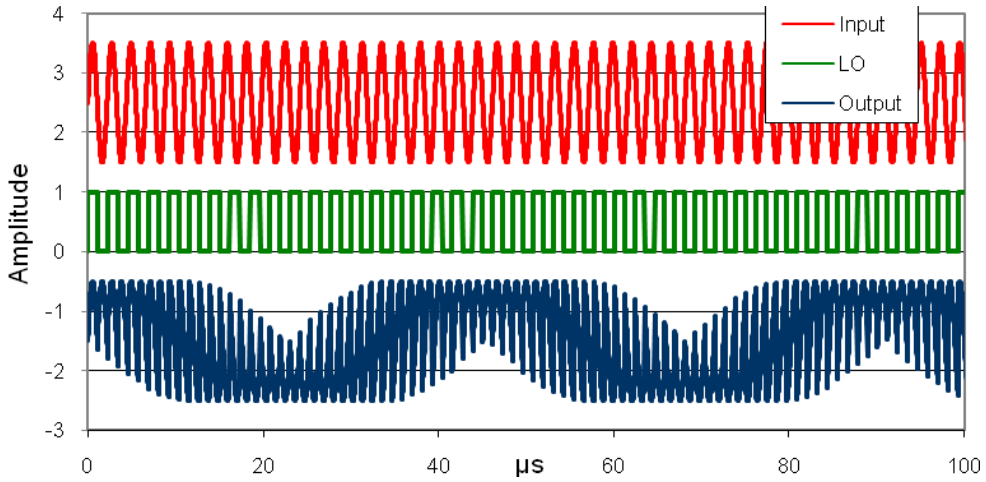
In this case, the intended output is at $F_{LO} + F_{SIG}$ and $F_{LO} - F_{SIG}$, as shown in the FFT below.



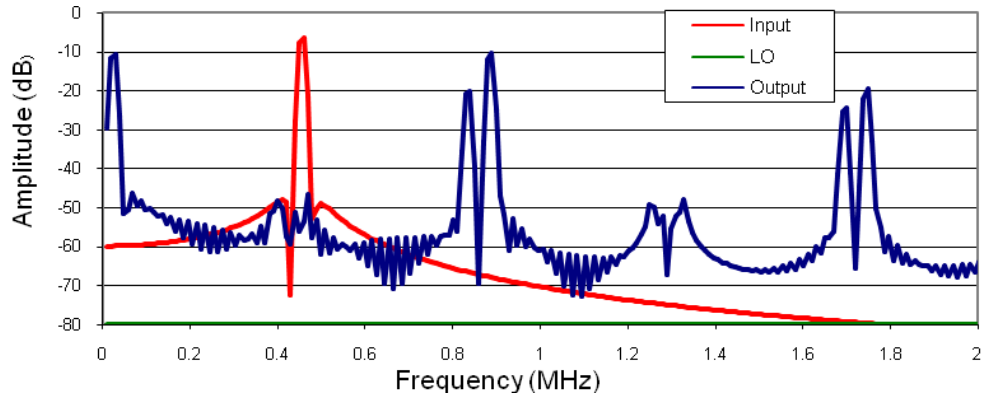
If a specific sideband, for example, $F_{LO} + F_{SIG}$, is required, the unwanted sideband can be filtered out with active RC filters using the on-board opamps. You can also use the Filter component after digitizing the Mixer output waveform.

Up Mixer: LO frequency less than signal frequency

Shown with input frequency of 455 kHz and LO of 430 kHz to yield a nominal output at 25 kHz. The underlying sine wave at 25 kHz is apparent, but not obvious because the sum of 455 and 430 kHz appears at the same level.

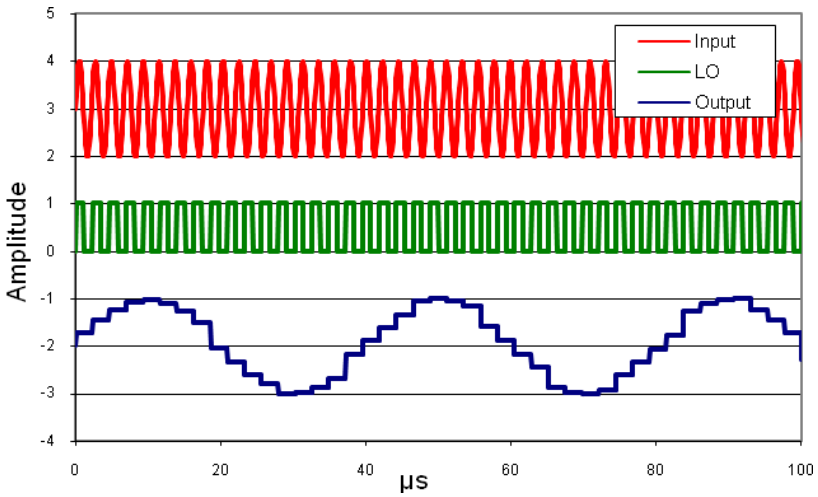


The FFT for these waveforms clearly shows the input at 455 kHz, the output difference frequency at the intended 25 kHz, and the sum output of the signal and first LO harmonic at 885 kHz. Just below the first harmonic's sum output is a term at 3 × F_{LO} – F_{SIG} or 835 kHz. The pattern repeats with the term at 5 × F_{LO} – F_{SIG} just below 3 × F_{LO} + F_{SIG}. The miscellaneous “stuff” between these well-known spectral lines is a function of the FFT, the windowing calculation process, and the sin(x)/x nature of the sampling process. The look of these signals can change depending on the type of spectrum analyzer used (swept spectrum versus FFT).



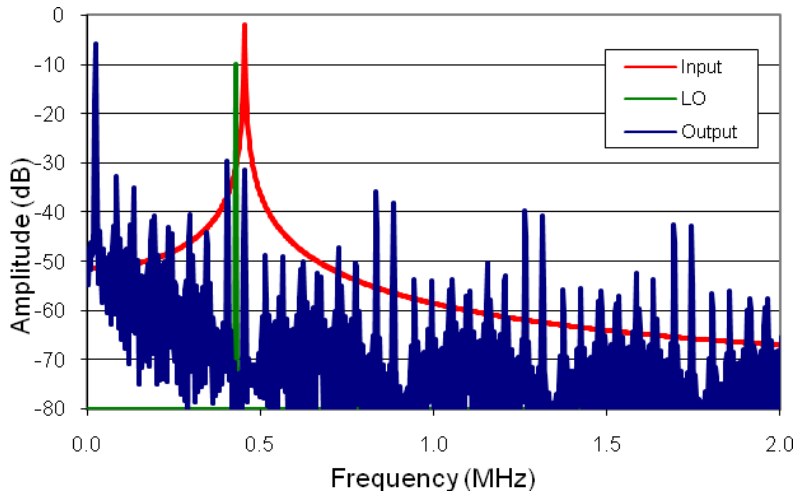
Down Mixer: LO near F_{SIG}

When the LO frequency is near the signal frequency, a sampling mixer offers advantages over a multiplying mixer. The time domain plot shows less higher-frequency content for harmonics of the mixer. The steps at the sampling rate of the LO are readily apparent. The LO can be either above or below the signal frequency, but the frequency distribution for LO > F_{SIG} will be inverted compared to the frequency distribution for LO < F_{SIG}.



The mixing products are sin(x)/x related, so that when the sampling frequency (LO) is close to the signal frequency, 'x' is close to π. These terms are quite different from the 1/n harmonic characteristic of the multiplying mixer. The harmonic content generated is substantially lower, which means that higher-order terms are more easily filtered and eliminated.

The difference frequency between the signal and LO shows clearly in the FFT. Mix products near the signal frequency are somewhat higher than the multiplying mixer, but all higher-order harmonic terms are substantial.

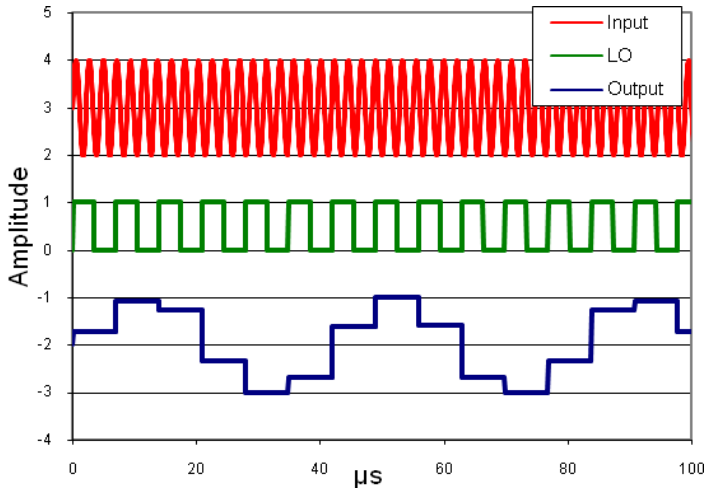


When the LO frequency is above $F_{SIG} \div 2$ or below $F_{SIG} \times 1.5$, substantial mix products appear and the mixer loses its utility, it hardly separates the desired difference frequency from the mix products.

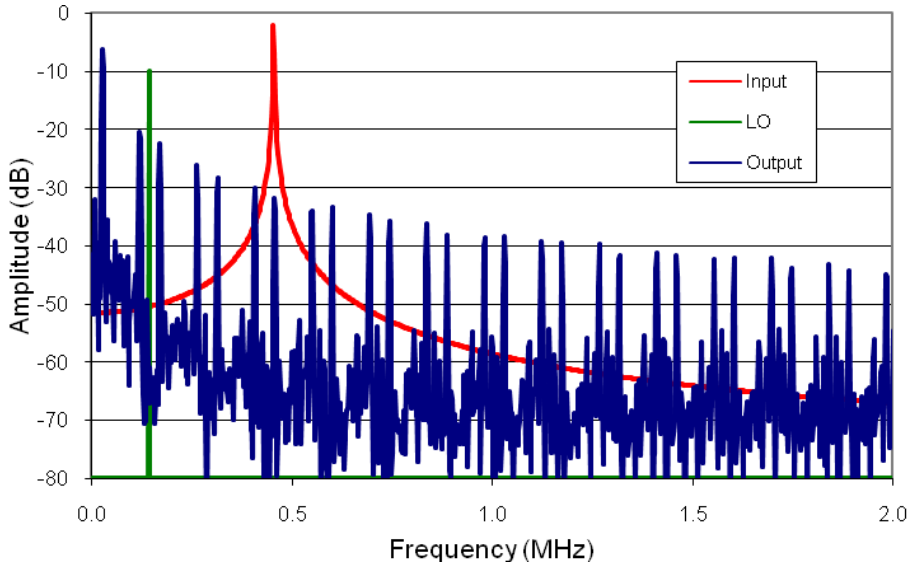
Down Mixer: LO frequency less than $F_{SIG}/2$

This is referred to as a subsampling mixer. When the LO frequency is less than half of the signal frequency, the primary output is at $F_{SIG} - n \times F_{LO}$ where 'n' is the largest integer such that $n \times F_{LO}$ is less than F_{SIG}. The waveforms for F_{SIG} = 455 kHz and F_{LO} = 143.3 kHz (= 430 kHz ÷ 3) show

that the primary output frequency is 25 kHz. The waveform is “coarser” than one with a higher frequency LO, but the output frequency is the same as if the signal was sampled at a higher rate.



The advantage of the subsampling mixer is in the range of the allowed input signal frequency. It is common to subsample by a factor of 4, so that a 13.57-MHz frequency can be sampled at 3.2 MHz to yield a primary output frequency of 770 kHz.



Over sampling mixers (for example, F_{SIG} = 455 kHz and LO = 820 kHz) result in mixer products similar to those of the multiplying mixer. These products may be more difficult to filter out of the desired waveform.

Input/Output Connections

This section describes the input and output connections for the Mixer component. An asterisk (*) in the list of I/Os indicates that the I/O may be hidden on the symbol under the conditions listed in the description of that I/O.

Fin – Analog

Fin is the input signal terminal. The Fin signal is mixed with the local oscillator clock signal to generate the Fout signal. Fin frequency is limited as follows:

- Multiply (Up) mixer Fin < 500 kHz
- Sample (Down) mixer Fin < 14 MHz

LO – Digital

LO is the local oscillator signal terminal. This signal serves as the sampling clock for the mixer. The LO signal is mixed with the Fin signal to generate the Fout signal. For Multiply Mixer mode, the LO clock signal must have a duty cycle of 50 percent.

LO frequency is limited as follows:

- Multiply (Up) mixer LO < 1 MHz
- Sample (Down) mixer LO < 4 MHz

Vref – Analog

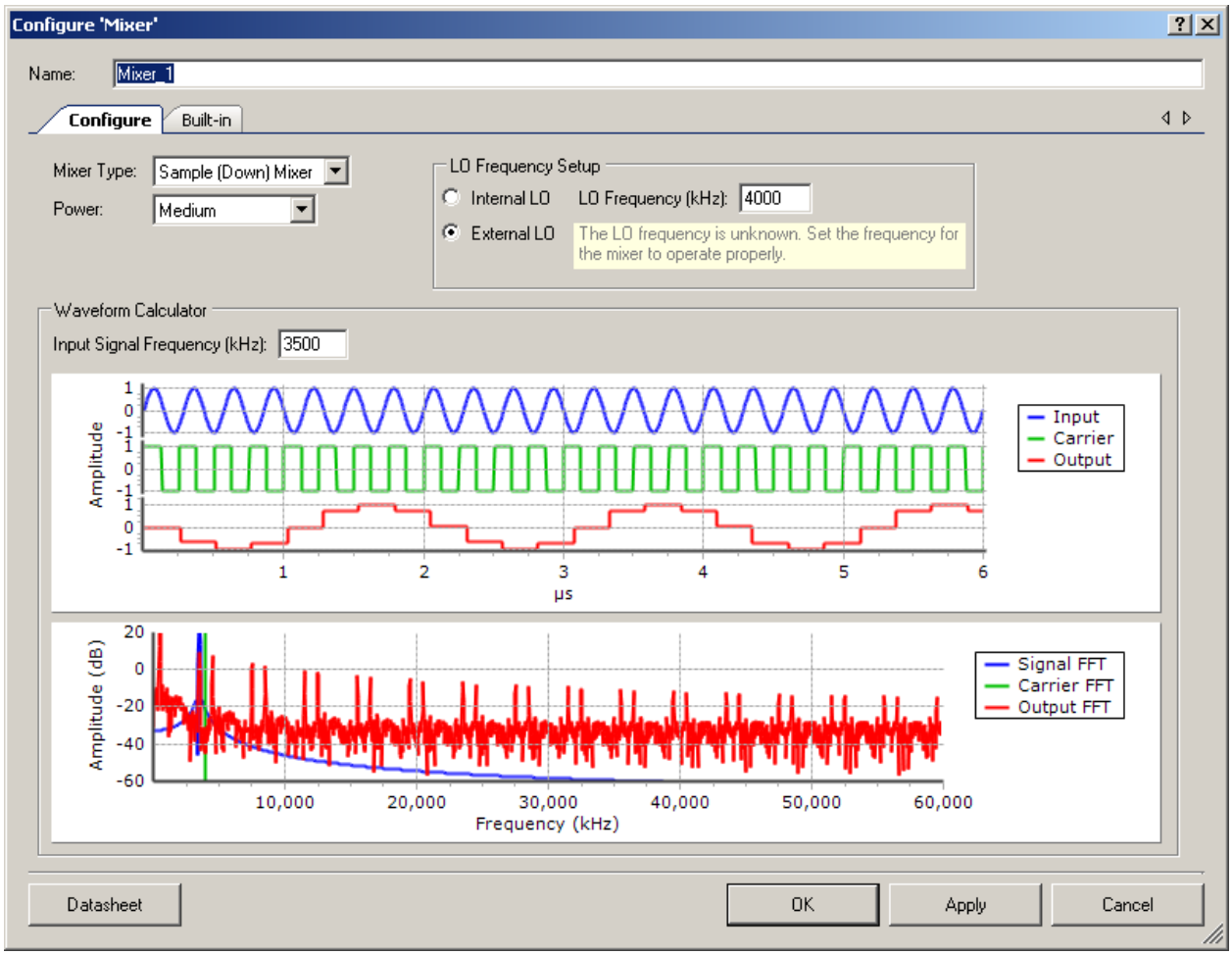
Vref is the input terminal for a reference voltage. The reference voltage can be one of the PSoC internal reference sources, an internal VDAC value, or an external signal.

Fout – Analog

Fout is the output signal terminal. The Fout signal is the resultant signal of the mixing operation of the Fin and LO signals.

Component Parameters

Drag a Mixer component onto your design and double-click it to open the **Configure** dialog.



Mixer Type

This parameter determines the configured mode of the mixer SC/CT block. The component supports two mixer modes: **Multiply (Up) Mixer** and **Sample (Down) Mixer**.

Power

This sets the initial drive power of the mixer. The power determines the speed with which the mixer reacts to changes in the input signal. There are four power settings: **Minimum**, **Low**, **Medium** (default), and **High**. A **Low** power setting results in the slowest response time and a **High** power setting results in the fastest response time.

LO Frequency Setup

The Mixer can be connected to a clock source external to the component (**External LO**) or may configure its own clock (**Internal LO**). If the LO is external, you must supply a 50-percent duty cycle for Up mixers (Down mixers do not have this requirement). If the LO is internal, the component derives the desired clock frequency with a 50-percent duty cycle for Up mixers. This impacts the clock divider calculation. When changing a Mixer from Up to Down, or vice versa, you may need to change the clock parameters to keep the Mixer operating properly in the Up mode.

LO Frequency

This parameter sets the clock frequency when **LO Frequency Setup** is set to **Internal LO**. In the Up mode, the terminating resistances in the mixer have values that are switched depending on operating frequency to optimize performance. Lower **LO Frequency** values allow you to use higher internal resistance values, resulting in slightly better modulator performance.

When **LO Source** is set to **External LO**, you must set the frequency in your external clock (whether clock resource or digital block source).

Application Programming Interface

Application Programming Interface (API) routines allow you to configure the component using software. The following table lists each routine and provides a brief functional description. The subsequent sections cover each function in more detail.

By default, PSoC Creator assigns the instance name “Mixer_1” to the first instance of the component in a given design. You can rename it to any unique value that follows the syntactic rules for identifiers. The instance name becomes the prefix of every global function name, variable, and constant symbol associated with the component. For readability, the instance name used in the following table is “Mixer.”

Function	Description
Mixer_Start()	Powers up the Mixer.
Mixer_Stop()	Powers down the Mixer.
Mixer_SetPower()	Sets drive power to one of four levels.
Mixer_Sleep()	Stops and saves the user configuration.
Mixer_Wakeup()	Restores and enables the user configuration.
Mixer_Init()	Initializes or restores default Mixer configuration.
Mixer_Enable()	Enables the Mixer.
Mixer_SaveConfig()	Empty function. Provided for future use.
Mixer_RestoreConfig()	Empty function. Provided for future use.

Global Variables

Variable	Description
Mixer_initVar	Indicates whether the Mixer has been initialized. The variable is initialized to 0 and set to 1 the first time Mixer_Start() is called. This allows the component to restart without reinitialization after the first call to the Mixer_Start() routine. If reinitialization of the component is required, then the Mixer_Init() function can be called before the Mixer_Start() or Mixer_Enable() function.

void Mixer_Start(void)

Description: Performs all of the required initialization for the component and enables power to the block. The first time the routine is executed, the input and feedback resistance values are configured for the operating mode selected in the design. When called to restart the mixer following a Mixer_Stop() call, the current component parameter settings are retained.

Parameters: None

Return Value: None

Side Effects: None

void Mixer_Stop(void)

Description: Turns off the Mixer block.

Parameters: None

Return Value: None

Side Effects: Does not affect mixer type or power settings

void Mixer_SetPower(uint8 power)

Description: Sets the drive power to one of four settings; minimum, low, medium, or high.

Parameters: uint8 power: See the following table for valid power settings.

Power Setting	Notes
Mixer_MINPOWER	Lowest active power and slowest reaction time
Mixer_LOWPOWER	Low power and speed
Mixer_MEDPOWER	Medium power and speed
Mixer_HIGHPOWER	Highest active power and fastest reaction time

Return Value: None

Side Effects: None

void Mixer_Sleep(void)

Description: This is the preferred API to prepare the component for sleep. The Mixer_Sleep() API saves the current component state. Then it calls the Mixer_Stop() function and calls Mixer_SaveConfig() to save the hardware configuration.

Call the Mixer_Sleep() function before calling the CyPmSleep() or the CyPmHibernate() function. Refer to the PSoC Creator *System Reference Guide* for more information about power-management functions.

Parameters: None

Return Value: None

Side Effects: None

void Mixer_Wakeup(void)

Description: This is the preferred API to restore the component to the state when Mixer_Sleep() was called. The Mixer_Wakeup() function calls the Mixer_RestoreConfig() function to restore the configuration. If the component was enabled before the Mixer_Sleep() function was called, the Mixer_Wakeup() function will also re-enable the component.

Parameters: None

Return Value: None

Side Effects: Calling the Mixer_Wakeup() function without first calling the Mixer_Sleep() or Mixer_SaveConfig() function may produce unexpected behavior.

void Mixer_Init(void)

Description: Initializes or restores the component according to the customizer Configure dialog settings. It is not necessary to call Mixer_Init() because the Mixer_Start() API calls this function and is the preferred method to begin component operation.

Parameters: None

Return Value: None

Side Effects: All registers will be set to values according to the customizer Configure dialog.

void Mixer_Enable(void)

Description: Activates the hardware and begins component operation. It is not necessary to call Mixer_Enable() because the Mixer_Start() API calls this function, which is the preferred method to begin component operation.

Parameters: None

Return Value: None

Side Effects: None

void Mixer_SaveConfig(void)

Description: Empty function. Provided for future use.

Parameters: None

Return Value: None

Side Effects: None

void Mixer_RestoreConfig(void)

Description: Empty function. Provided for future use.

Parameters: None

Return Value: None

Side Effects: None

Sample Firmware Source Code

PSoC Creator provides many example projects that include schematics and example code in the Find Example Project dialog. For component-specific examples, open the dialog from the Component Catalog or an instance of the component in a schematic. For general examples, open the dialog from the Start Page or **File** menu. As needed, use the **Filter Options** in the dialog to narrow the list of projects available to select.

Refer to the “Find Example Project” topic in the PSoC Creator Help for more information.

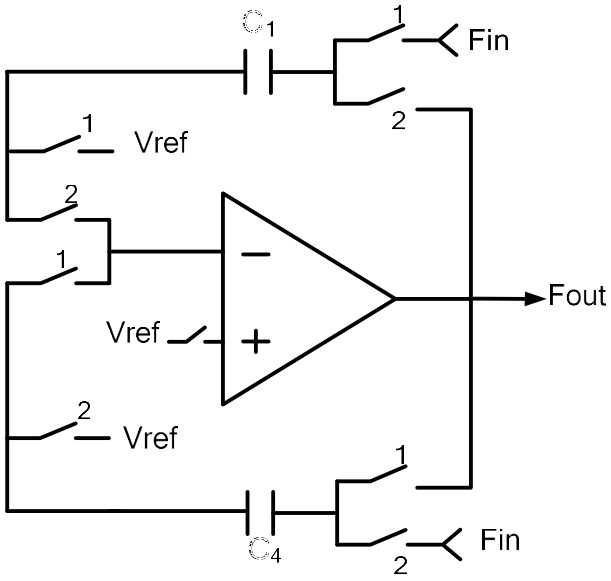
Functional Description

Mixer functionality is implemented using the PSoC SC/CT block. The discrete-time Down mixer is implemented using the switched-capacitor mode. The multiplying (Up) mixer uses the continuous-time block mode.

Discrete Time Down Mixer

The schematic for the internal configuration of the discrete time mixer is shown in Figure 1.

Figure 1. Discrete-Time Sample-and-Hold Mixer Schematic



The non-return-to-zero sample and hold is achieved by switching the integrating capacitor between two capacitors. In Figure 1, either C_1 or C_4 can always be sampling the input signal while the other is being integrated across the amplifier. The Fin signal is sampled at a rate less than the F_{IN} signal frequency. The mixer component is configured such that F_{OUT} is integrated with a new value on the rising edge of the input clock.

For LO sample clock frequencies greater than half of the F_{IN} signal frequency, the output is the difference between the input and LO frequencies plus aliasing components. When the sample clock frequency is less than half of the F_{IN} signal frequency, the output is the difference between the input and the largest integer multiple of the LO frequency that is less than the F_{IN} signal frequency.

For a given input carrier frequency, F_{IN}, a sample LO clock frequency, F_{CLK}, can be chosen to provide the desired output frequency, F_{OUT}, for the system.

Provided that F_{CLK} is less than 4 MHz, and F_{IN} is less than 14 MHz:

If $\frac{2N-1}{2}F_{CLK} < F_{IN} < N \times F_{CLK}$, then $F_{OUT} = N \times F_{CLK} - F_{IN}$ Equation 1

If $N \times F_{CLK} < F_{IN} < \frac{2N+1}{2}F_{CLK}$, then $F_{OUT} = F_{IN} - N \times F_{CLK}$ Equation 2

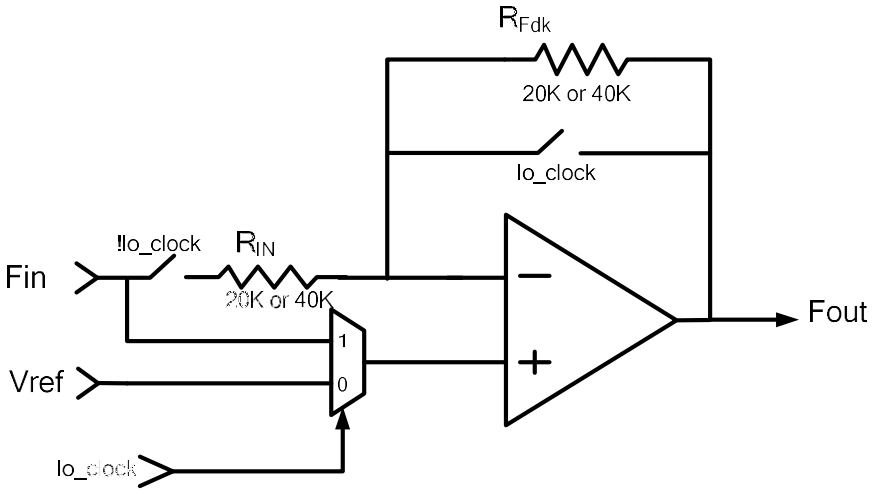
Equation 1 and Equation 2 can be summarized as:

$F_{OUT} = \text{abs}(N \times F_{CLK} - F_{IN})$ Equation 3

Continuous-Time Up Mixer

The schematic for the internal configuration of the continuous-time mixer is shown in Figure 2.

Figure 2. Continuous-Time Mixer Configuration Schematic



In this mode, the opamp is configured as a PGA that uses the LO input signal to toggle between an inverting PGA gain of 1 and a non-inverting unity gain buffer. The output signal includes frequency components at $F_{CLK} \pm F_{IN}$ plus terms at odd harmonics of the LO frequency plus and minus the input signal frequency: $3 \times F_{CLK} \pm F_{IN}$, $5 \times F_{CLK} \pm F_{IN}$, $7 \times F_{CLK} \pm F_{IN}$, and so on.

$$F_{OUT} = N \times F_{CLK} \pm F_{IN} \quad \text{with N holding odd values} \quad \text{Equation 4}$$

Frequency Planning

Proper frequency planning is required to achieve the desired F_{OUT}. The clocks must be carefully controlled in the design-wide resources.

Resources

The Mixer component uses one SC/CT analog block.

API Memory Usage

The component memory usage varies significantly, depending on the compiler, device, number of APIs used and component configuration. The following table provides the memory usage for all APIs available in the given component configuration.
The measurements have been done with the associated compiler configured in Release mode with optimization set for Size. For a specific design the map file generated by the compiler can be analyzed to determine the memory usage.

Configuration	PSoC 3 (Keil_PK51)		PSoC 5 (GCC)		PSoC 5LP (GCC)	
	Flash Bytes	SRAM Bytes	Flash Bytes	SRAM Bytes	Flash Bytes	SRAM Bytes
Default	178	2	296	12	236	5

DC and AC Electrical Characteristics for PSoC 3

Specifications are valid for –40 °C ≤ T_A ≤ 85 °C and T_J ≤ 100 °C, except where noted.
Specifications are valid for 1.71 V to 5.5 V, except where noted. Typical values are for T_A = 25°C

DC Specifications

Parameter	Description	Conditions	Min	Typ	Max	Units
V_{OS}	Input offset voltage		–	–	10	mV
	Quiescent current		–	0.9	2	mA
G	Gain		-	0	-	dB

AC Specifications

Parameter	Description	Conditions	Min	Typ	Max	Units
F_{LO}	Local oscillator frequency	Down mixer mode	–	–	4	MHz
F_{IN}	Input signal frequency	Down mixer mode	–	–	14	MHz
F_{LO}	Local oscillator frequency	Up mixer mode	–	–	1	MHz
F_{IN}	Input signal frequency	Up mixer mode	–	–	1	MHz
SR	Slew rate		3	–	–	V/µs

DC and AC Electrical Characteristics for PSoC 5

Specifications are valid for –40 °C ≤ T_A ≤ 85 °C and T_J ≤ 100 °C, except where noted.
Specifications are valid for 2.7 V to 5.5 V, except where noted. Typical values are for T_A = 25°C

DC Specifications

Parameter	Description	Conditions	Min	Typ	Max	Units
V_{OS}	Input offset voltage		–	–	26	mV
	Quiescent current		–	0.9	2	mA
G	Gain		–	0	–	dB

AC Specifications

Parameter	Description	Conditions	Min	Typ	Max	Units
F_{LO}	Local oscillator frequency	Down mixer mode	–	–	4	MHz
F_{IN}	Input signal frequency	Down mixer mode	–	–	14	MHz
F_{LO}	Local oscillator frequency	Up mixer mode	–	–	1	MHz
F_{IN}	Input signal frequency	Up mixer mode	–	–	1	MHz
SR	Slew rate		3	–	–	V/µs

Component Changes

This section lists the major changes in the component from the previous version.

Version	Description of Changes	Reason for Changes / Impact
1.91	For low voltage VDDA operation uses a boost clock shared by all the SC/CT based components.	Reduces the number of analog clocks required in the system for boost clocks. With this change a single boost clock is shared instead of using a separate clock for each SC/CT based component.
1.90	Added all APIs with the CYREENTRANT keyword when they are included in the .cyre file.	This change is required to eliminate compiler warnings for functions that are not reentrant used in a safe way: protected from concurrent calls by flags or Critical Sections.
	Added PSoC 5LP support.	
1.80	Updated customizer to avoid overlapping of configure window labels and controls across different DPI settings.	Labels and text were overlapped in the old version of Mixer.
	Updated customizer to prevent the user from entering invalid values for LO and Signal frequencies	Customizer was not responding to user entries when invalid frequencies are provided for LO and Signal frequencies.
1.70	Mixer_Stop() API changes for PSoC 5.	Change required to prevent the component from impacting unrelated analog signals when stopped, when used with PSoC 5.
	Mixer Response GUI implementation	To guide the user to an understanding of Mixer performance.
	Added PSoC 5 characterization data to datasheet	

1.60	Removed VDDA parameter from component customizer	VDDA setting in the component is redundant and unnecessary for multiple components. The parameter was removed and the component queries the global setting for minimum VDDA in the DWR and automatically enables the pump when necessary.
	Added a GUI Configuration Editor	Previous configuration window did not provide enough information for ease of use.
	LO - local oscillator is enabled correctly	The local oscillator was not being enabled correctly in previous versions of the component.
	Added characterization data to datasheet	
	Minor datasheet edits and updates	
1.50	Added Sleep/Wakeup and Init/Enable APIs.	To support low power modes, as well as to provide common interfaces to separate control of initialization and enabling of most components.
	Updated Symbol and Configure dialog.	To comply with corporate standards.

© Cypress Semiconductor Corporation, 2012. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

PSoC® is a registered trademark, and PSoC Creator™ and Programmable System-on-Chip™ are trademarks of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

Any Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.