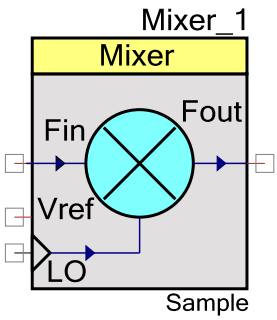
# Mixer
1.50

## Features




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

## General Description

The Mixer component provides a single-ended, non-precision mixer. The component offers two configurations:

- continuous time mode, used for multiply or up mixing
- discrete time, sample and hold mode, used for sampled or down mixing.

The component accepts as inputs two signals at different frequencies and presents at the output a mixture of signals at multiple frequencies, including the sum and difference of the input signal and the local oscillator signal. Typically, the undesired frequency components in the output signal are removed by filtering.

### When to use a Mixer

The Mixer component can be used for frequency conversion of a given input signal using a fixed local oscillator signal as the sampling clock.

The manipulations of signal frequencies performed by a mixer 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, whether in hardware or software.

The mixer output can be used by filtering the desired signal harmonics using an off-chip filter or the output can be used to drive an on-chip ADC through internal routing.

**PRELIMINARY**

# Input/Output Connections

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

## Fin – Analog

$F_{in}$ is the input signal terminal. The $F_{in}$ signal is mixed with the local oscillator clock signal to generate the $F_{out}$ signal. $F_{in}$ 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 $F_{in}$ signal to generate the $F_{out}$ signal. For the Multiply Mixer mode, the LO clock signal must have a duty cycle of 50%.

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 may be one of the PSoC internal reference sources, an internal VDAC value, or an external signal.

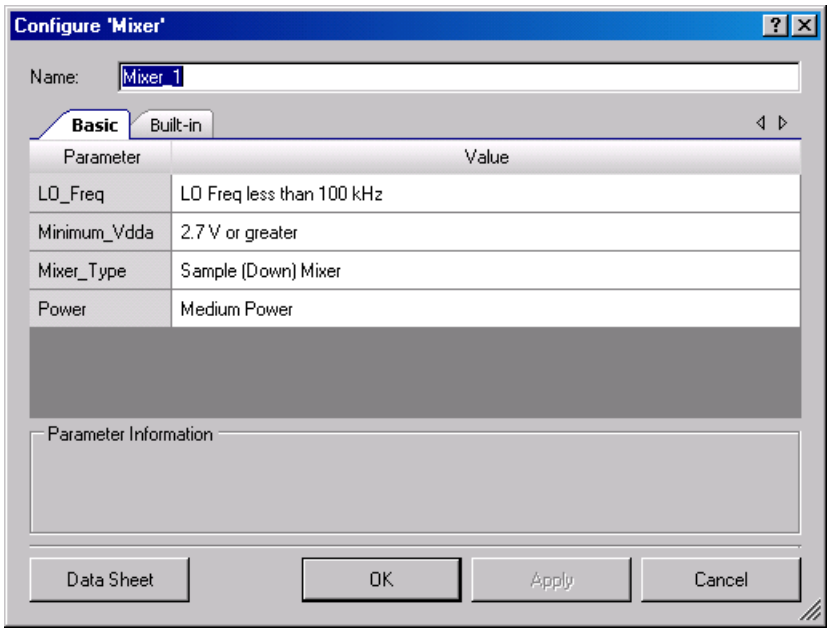
## Fout – Analog

$F_{out}$ is the output signal terminal. The $F_{out}$ signal is the resultant signal of the mixing operation of the $F_{in}$ and LO signals.

# Component Parameters

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

**Figure 1: Configure Mixer Dialog**




## LO_Freq

This parameter is used to determine the appropriate values for the input and feedback resistance of the mixer op-amp circuit. The parameter options are:

- LO Freq less than 100 kHz – For local oscillator frequencies less than 100 kHz, 40 kΩ is used for the input and feedback resistance.
- LO Freq 100 kHz or greater – For local oscillator frequencies of 100 kHz or higher, the resistance values are set to 20 kΩ.

## Minimum Vdda

This parameter is determined by the minimum analog supply voltage expected for the PSoC in the design. The parameter can be set to one of two values:

- 2.7 V or greater (default)
- Less than 2.7 V

For an analog supply voltage below 2.7 V, the amplifier makes use of an internal boost circuit. The component implementation uses an additional 10 MHz clock to drive the boost circuit for the amplifier block.

### 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.

## Placement

There are no placement specific options.

## Resources

The mixer uses one SC/CT block.

## 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 |
|---|---|
| void Mixer_Init(void) | Initializes or restores default Mixer configuration. |
| void Mixer_Enable(void) | Enables the Mixer. |
| void Mixer_Start(void) | Power up the Mixer. |
| void Mixer_Stop(void) | Power down the Mixer. |
| void Mixer_SetPower(uint8 power) | Set drive power to one of four levels. |
| void Mixer_Sleep(void) | Stops and saves the user configuration. |
| void Mixer_Wakeup(void) | Restores and enables the user configuration. |

| Function | Description |
|---|---|
| void Mixer_SaveConfig(void) | Empty function. Provided for future usage. |
| void Mixer_RestoreConfig(void) | Empty function. Provided for future usage. |

## 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.<br>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_Init(void)

**Description:** Initializes or restores default Mixer configuration.

**Parameters:** None

**Return Value:** None

**Side Effects:** All registers will be reset to their initial values. This will re-initialize the component.

## void Mixer_Enable(void)

**Description:** Enables the Mixer.

**Parameters:** None

**Return Value:** None

**Side Effects:** None

## 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:** Turn 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:** Stops the component operation. Saves the configuration registers and the component enable state. Should be called just prior to entering sleep.

**Parameters:** None

**Return Value:** None

**Side Effects:** None

## void Mixer_Wakeup(void)

**Description:** Restores the component enable state and configuration registers. Should be called just after awaking from sleep.

**Parameters:** None

**Return Value:** None

**Side Effects:** None

### void Mixer_SaveConfig(void)

**Description:** Empty function. Provided for future usage.

**Parameters:** None

**Return Value:** None

**Side Effects:** None

### void Mixer_RestoreConfig(void)

**Description:** Empty function. Provided for future usage.

**Parameters:** None

**Return Value:** None

**Side Effects:** None

## Sample Firmware Source Code

The following is a C language example demonstrating the basic functionality of the mixer component. This example assumes the component has been placed in a design with the default name "Mixer_1."

**Note** If you renamed your component you must also edit the example code as appropriate to match the component name you specified.

### External Clock Oscillator

If the mixer component will be used with an external local oscillator signal, and the parameter settings are configured during the project design phase, only a call to the associated Mixer Start() routine is required to use this component.

```
#include <device.h>

void main()
{
    Mixer_1_Start();
}
```

### LO Generated Internally

If the LO signal is generated internal to the PSoC, APIs to configure and start the LO clock must also be called. The power setting for the Mixer_1 component can also be configured at run time.

```
#include <device.h>
#include "lo_clk.h"

void main()
```

```
{
    /* Setup Local Oscillator Clock */
    lo_clk_Enable();
    lo_clk_SetMode(CYCLK_DUTY);

    /* API Calls for Mixer Instance */
    Mixer_1_Start();
    Mixer_1_SetPower(Mixer_1_HIGHPOWER);

    while (1)
    {

    }
}
```

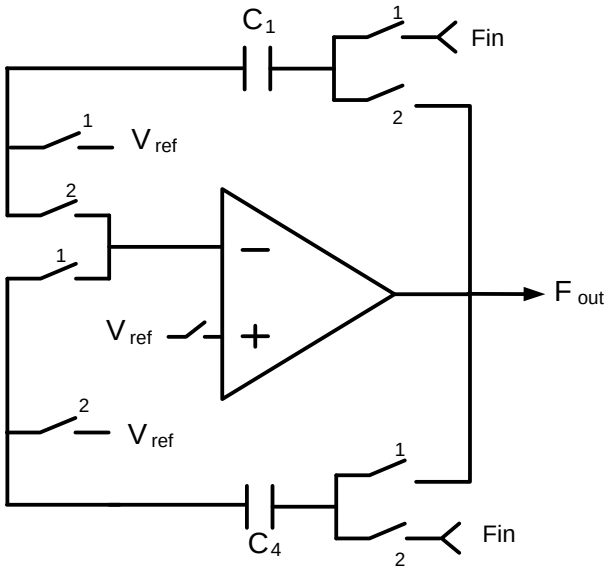
# Functional Description

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 the following figure:

**Figure 2 Discrete Time Sample & Hold Mixer Schematic**




The non-return-to-zero sample and hold is achieved by switching the integrating capacitor between two capacitors. In Figure 2 above, either C1 or C4 can always be sampling the input signal while the other is being integrated across the amplifier. The $F_{in}$ 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 4MHz, and $F_{in}$ is less than 14MHz:

**If** $\frac{2N+1}{2}F_{clk} < F_{in} < NF_{clk}$, **then** $F_{out} = NF_{clk} - F_{in}$ **Equation 1**

**If** $NF_{clk} < F_{in} < \frac{2N+1}{2}F_{clk}$, **then** $F_{out} = F_{in} - NF_{clk}$ **Equation 2**

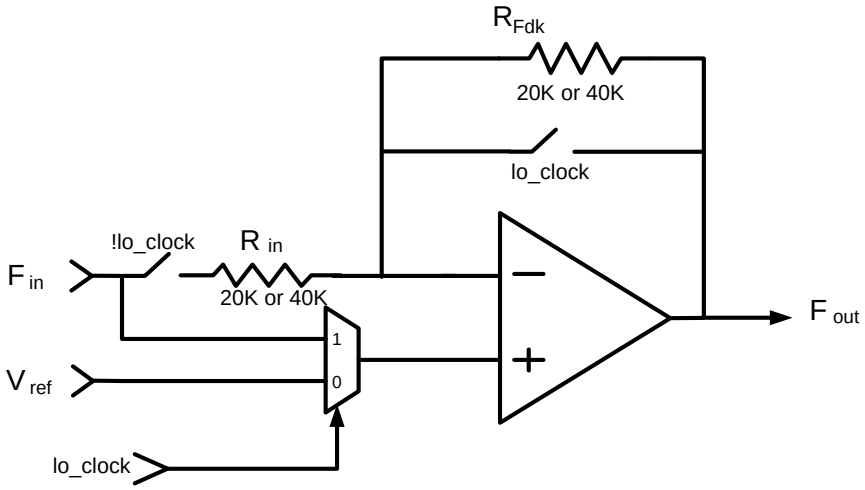
Equation 1 and Equation 2 can be summarized as:

$F_{out} = |F_{in} - NF_{clk}|$ **Equation 3**

## Continuous Time Up Mixer

The schematic for the internal configuration of the continuous time mixer is shown below:

**Figure 3: Continuous Time Mixer Configuration Schematic**




In this mode the op-amp 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: $3*F_{clk} \pm F_{in}$, $5*F_{clk} \pm F_{in}$, $7*F_{clk} \pm F_{in}$ etc.

$$F_{out} = N * F_{clk} \pm F_{in}$$ **with N holding odd values** **Equation 4**

### Frequency Planning

Note that proper frequency planning is required to achieve the desired $F_{out}$. As a minimum requirement, the Nyquist criteria must be met for the desired $F_{out}$.

$$F_{clk} > 2 * F_{out}$$ **Equation 5**

# DC and AC Electrical Characteristics

The following values are indicative of expected performance and based on initial characterization data. Unless otherwise specified in the tables below, all $T_A = 25^{\circ}C$, $V_{dd} = 5.0$ V, Power HIGH, Opamp bias LOW, output referenced to 1.024 V.

**Note** Characteristic data table will be updated following silicon characterization.

## DC Electrical Characteristics

| Parameter | Typical | Min | Max | Units | Conditions and Notes |
|---|---|---|---|---|---|
| Offset Voltage | | | | | |
| Input Offset Voltage | | 1.3 | 10 | mV | |
| Input Current (linear)<br>Rfb = 20 kΩ | | | +/- 120 | uA | Vdda = 5.0 V<br>Vref = Vdda/2<br>Linearity +/1 1% |
| Input Current (linear)<br>Rfb = 40 kΩ | | | +/- 2.5 | uA | Vdda = 5.0 V<br>Vref = Vdda/2<br>Linearity +/1 1% |
| Operating Current | | | | | |
| Off | | 0 | 0.1 | uA | Output tri-stated |
| Minimum Power | | 80 | 100 | uA | |
| Low Power | | | | | |
| Medium Power | | | | | |
| High Power | | 400 | 500 | uA | |

## AC Electrical Characteristics

| Parameter | Typical | Min | Max | Units | Conditions and Notes |
|---|---|---|---|---|---|
| AC Electrical Characteristics | | | | | |
| Slew Rate (20% to 80%) | | | | | |
| Minimum Power | | | | V/uS | |
| Low Power | | | | V/uS | |
| Medium Power | | | | V/uS | |
| High Power | | | | V/uS | |
| CT Up Mixer | | | | | |
| Sample Frequency | | | 1.0 | MHz | |
| Signal Frequency | | | 500 | kHz | |
| SC Down Mixer | | | | | |
| Sample Frequency | | | 4.0 | MHz | |
| Signal Frequency | | | 14 | MHz | |
| Noise | | | | | |
| Minimum Power | | | | nV/√Hz | |
| Low Power | | | | nV/√Hz | |
| Medium Power | | | | nV/√Hz | |
| High Power | | | | nV/√Hz | |
| CMRR | 90 | 60 | | | at 1.0 kHz, 1.0 V headroom |
| PSRR | | 69 | | dB | at 100 kHz, 1.0 V headroom |

# Component Changes

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

| Version | Description of Changes | Reason for Changes / Impact |
|---|---|---|
| 1.50.b | Minor datasheet edits. | |
| 1.50.a | Minor datasheet edits. | |
| 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. |

**PRELIMINARY**