Delta Sigma ADC Datasheet DelSig V 1.50

Copyright © 2002-2013 Cypress Semiconductor Corporation. All Rights Reserved.

Resources	PSoC® Blocks				API Memory (Bytes)		Pins (per External I/O)
	Digital	Analog CT	Analog SC	Decimator Column	Flash	RAM	
CY8C29xxx, CYC8C24x94, CY7C64215, CY8CLED04/16, CY8CLED0xD, CY8CLED0xG, CY8CTST120, CY8CTMG120, CY8CTMA120, CY8C28x45, CY8CPLC20, CY8CLED16P01, CY8C28x43, CY8C28x52							
6, 1st-Order, 32	1	1	0	1	93	2	1
7.5, 1st-Order, 64	1	1	0	1	97	2	1
9, 1st-Order, 128	2	1	0	1	121	3	1
10.5, 1st-Order, 256	2	1	0	1	121	3	1
8, 2nd-Order, 32	1	2	0	1	108	2	1
10, 2nd-Order, 64	1	2	0	1	127	3	1
12, 2nd-Order, 128	2	2	0	1	130	3	1
14, 2nd-Order, 256	2	2	0	1	130	3	1

Note “DecimatorColumn" resource is available only for CY8C28x45 device.

See AN2239, ADC Selection Guide for other converters.

Features and Overview

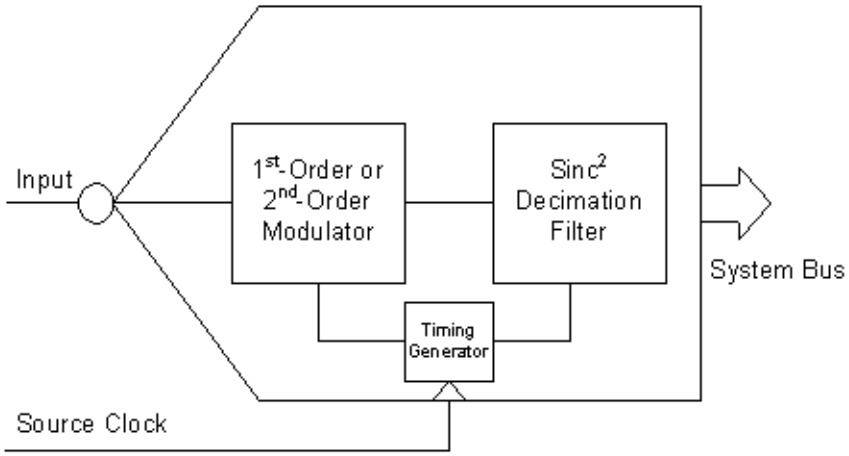
- 6-bit resolution with 32X oversampling to 14-bit resolution with 256X oversampling
- Data in unsigned or signed 2’s complement formats
- Maximum sample rates of 65,500 sps at 6 bit resolution, 7812 sps at 14-bit resolution
- Sinc^2 filter fully implemented in hardware reduces CPU overhead and anti-alias requirements
- First-order or second-order modulator, user selectable
- Input range defined by internal and external reference options
- Optional synchronized PWM Output

The DelSig is an integrating converter, requiring from 32 to 256 integration cycles to generate a single output sample. Changing multiplexed inputs, invalidates the first two samples following the change.

Review the Parameters section before placing a module.

As shown in the block diagram, the DelSig comprises three primary functions, a modulator, a Sinc^2 Decimation Filter, and a timing generator. Each component offers options that may be tailored to find the right balance between performance and resource use for a given application.

Figure 1. DelSig Block Diagram



Functional Description

The resolution and sample rate of the DelSig are determined by the modulator type, decimation rate, and column clock frequency. Selection of these parameters and the power setting also determine the linearity and offset voltage of the converter.

Figure 2. Sample Rate vs Decimation and Clock for 1st Order Modulator

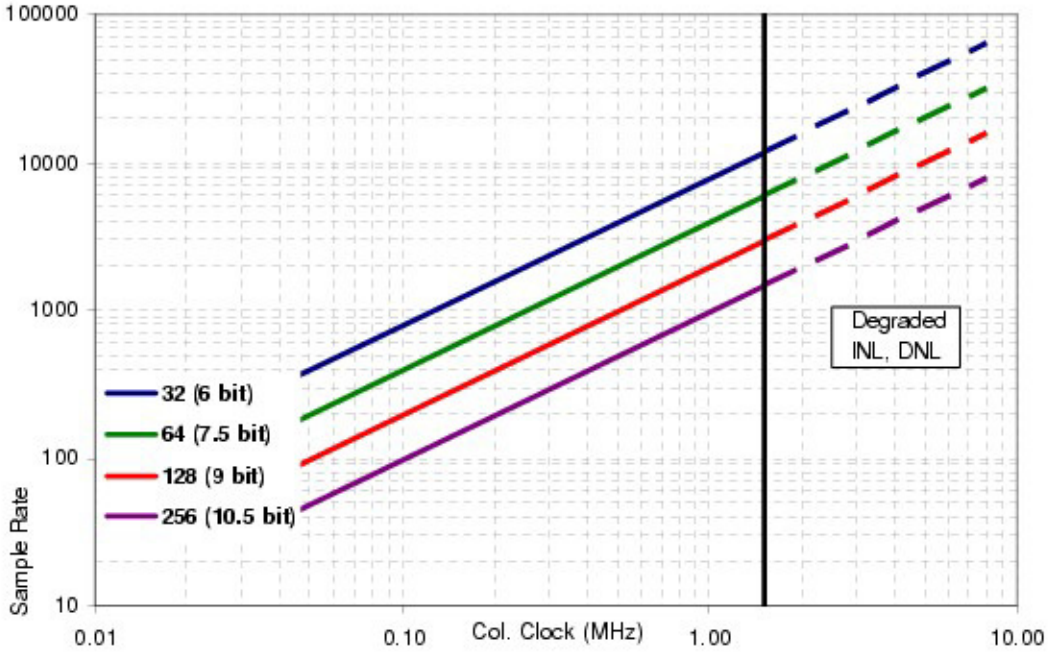
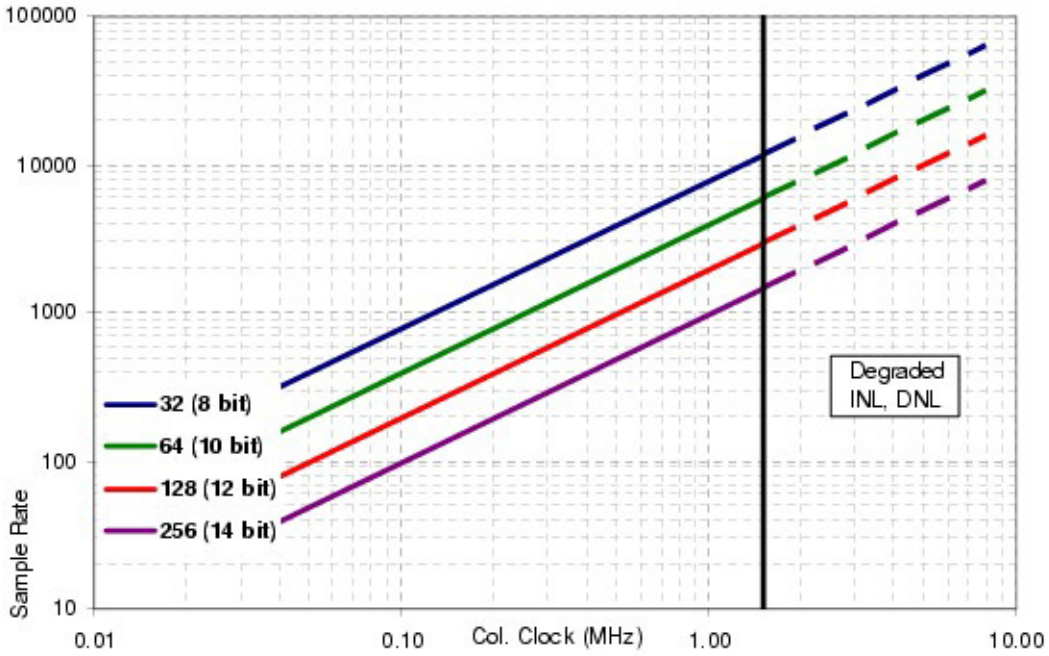


Figure 3. Sample Rate vs Decimation and Clock for 2nd Order Modulator

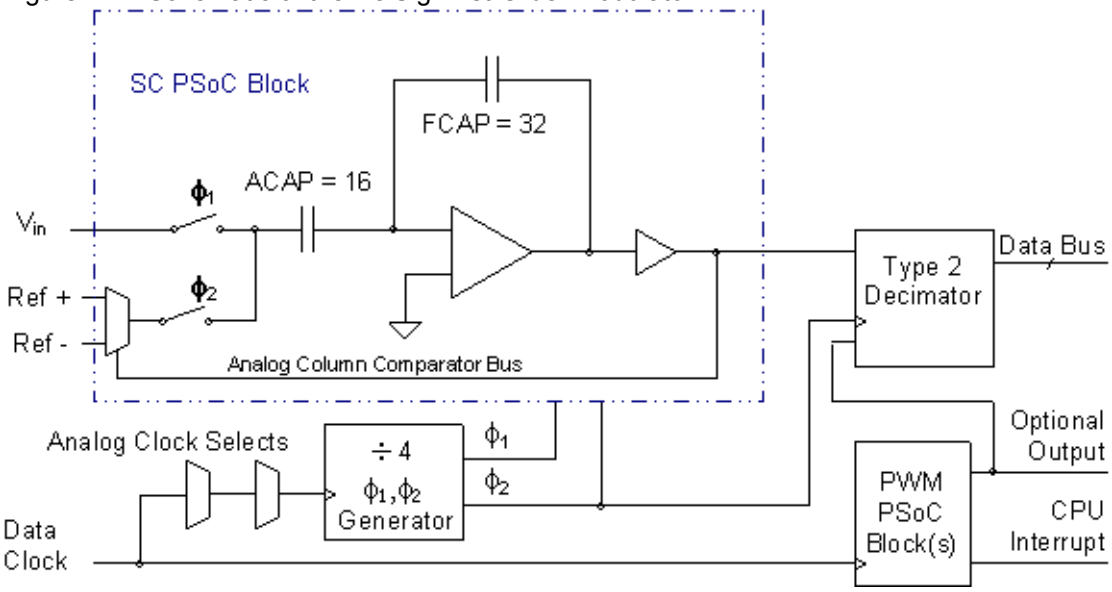


Modulator

The modulator is a 1-bit oversampling integrator/comparator circuit that represents the input voltage as the density of a serial bit stream of 1's and 0's. The modulator output is reduced to the final sample rate by the low-pass decimation filter that converts multiple 1-bit samples into samples of higher resolution. In general, higher decimation rates (that is, higher oversample rates) produce higher resolution results but other factors, such as the order of the modulator, also matter.

A key benefit of delta-sigma converters is the "noise shaping" provided by the modulator. Normally, the quantization noise inherent in sampling a signal is evenly distributed in frequency between "DC" and one-half the sample frequency or Nyquist frequency. Simply put, the delta-sigma modulator shifts some of the quantization noise from the lower frequencies into higher frequencies that are later attenuated by the decimation filter. A second-order modulator that requires two switched-capacitor analog PSoC blocks does a better job of noise shaping than the first-order modulator that only requires one analog PSoC block. At the highest decimation rate of 256X, a second-order modulator accounts for a 3.5-bit increase in the effective resolution compared to a first-order modulator.

Figure 4. Schematic of the DelSig First-Order Modulator



The analog block is configured as an integrator. The output polarity of the comparator configures reference multiplexer so the reference voltage is either added or subtracted from the input and placed in the integrator. This reference control attempts to pull the integrator output back towards zero. The single-bit comparator output is also fed into the decimator $sinc^2$ filter.

Note that the 1-bit oversample rate is determined by the divide-by-four generator that produces the Φ1 and Φ2 s that control the switched-capacitor (SC) PSoC block. The output rate is determined by dividing the data by 4 to get the 1-bit over sample rate and further dividing by the decimation rate to get the final sample rate.

Equation 1

$$SampleRate = \frac{DataClockFrequency}{4 \times DecimationRate} \text{ samples per second}$$

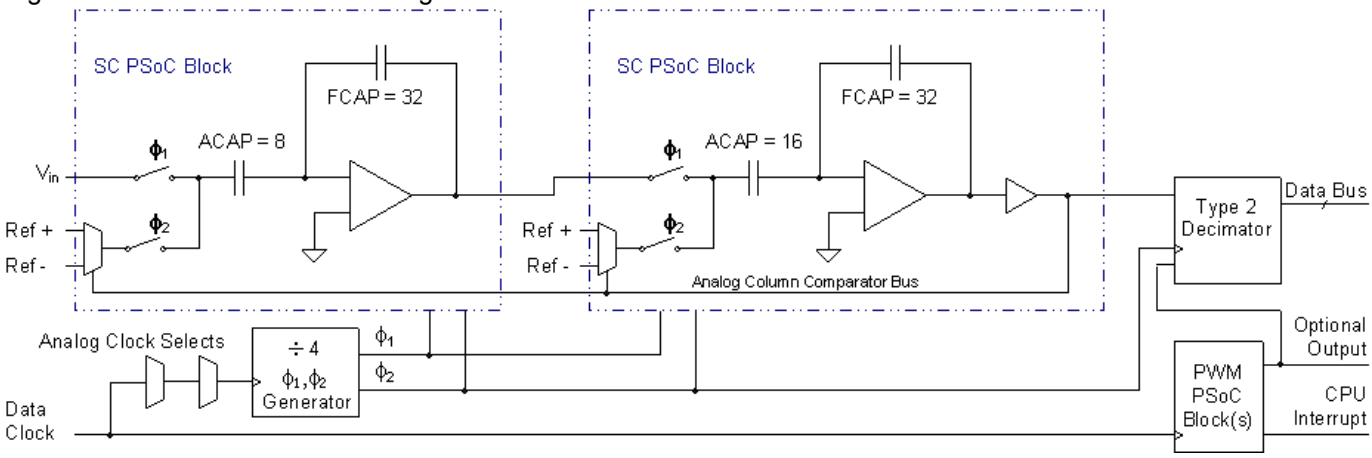
The highest data clock frequency that can be used is given in the specification tables, below. For a data clock of 8MHz, and a decimation rate of 256, the sample rate is:

Equation 2

$$8 \times 10^6 / (4 \times 256) = 7812.5 sps$$

A second-order modulator is constructed by feeding the analog output of a first-order modulator into a similar PSoC block and modifying the feedback arrangement so that the 1-bit comparator output of the second block back into both blocks as illustrated in Figure 5.

Figure 5. Schematic of the DelSig Second-Order Modulator



Because the analog comparator busses run vertically in the columns of the analog PSoC block array, the blocks of a second order modulator must be positioned one above the other.

The range of the DelSig is established by $\pm V_{Ref}$, where V_{Ref} is set by the user in the Global Resources window of PSoC Designer. For fixed scale, V_{Ref} is set to $\pm V_{Bandgap}$ or, for the CY8C29/27/24/22/21xxxfamily of PSoC Devices, ±1.6 $V_{Bandgap}$. For adjustable scale, V_{Ref} is set to ±Port 2[6]. For supply ratiometric scale, V_{Ref} is set to $\pm V_{DD}/2$. The complete list of options is given in the following table:

Table 1. Input Voltage Ranges for the RefMux Global Parameter Setting

RefMux Setting	V_{DD} = 5 V	V_{DD} = 3.3 V
(V_{DD}/2) ± BandGap	1.2 < V_{in} < 3.8	0.35 < V_{in} < 2.95
(V_{DD}/2) ± (V_{DD}/2)	0 < V_{in} < 5	0 < V_{in} < 3.3
BandGap ± BandGap	0 < V_{in} < 2.6	0 < V_{in} < 2.6
(1.6*BandGap) ± (1.6*BandGap)	0 < V_{in} < 4.16	NA
(2*BandGap) ± BandGap	1.3 < V_{in} < 3.9	NA
(2*BandGap) ± P2[6]	(2.6 - $V_{P2[6]}$) < V_{in} < (2.6 + $V_{P2[6]}$)	NA
P2[4] ± BandGap	($V_{P2[4]}$ - 1.3) < V_{in} < ($V_{P2[4]}$ + 1.3)	($V_{P2[4]}$ - 1.3) < V_{in} < ($V_{P2[4]}$ + 1.3)
P2[4] ± P2[6]	($V_{P2[4]}$-$V_{P2[6]}$) < V_{in} < ($V_{P2[4]}$+$V_{P2[6]}$)	($V_{P2[4]}$-$V_{P2[6]}$) < V_{in} < ($V_{P2[4]}$+$V_{P2[6]}$)

Sinc^2 Decimation Filter

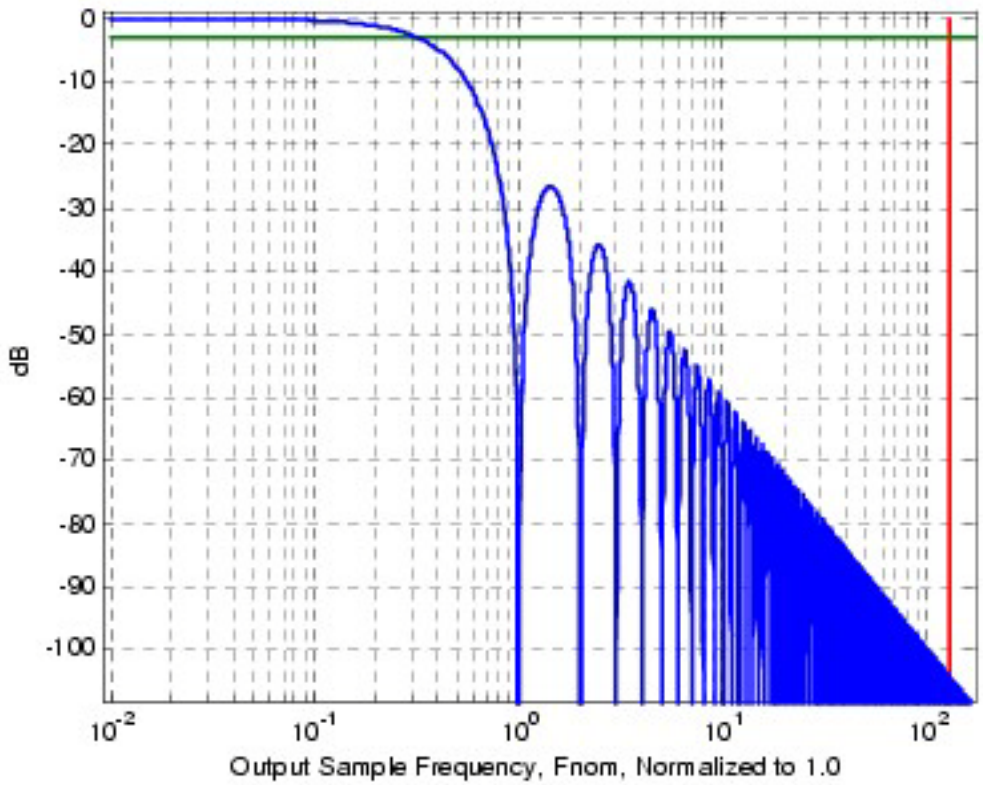
The response of the decimation filter is given by the following z-domain relation:

Equation 3

$$H(z) = \left[\frac{1 - z^{-n}}{1 - z^{-1}}\right]^2, \text{ where n is the decimation level.}$$

The frequency domain transfer function plotted in Figure 6 normalizes the frequency so the output sample rate, F_{nom}, equals 1.0. The -3 dB point occurs just above $0.318 \times F_{nom}$ and zeros of the function occur at each integer multiple of F_{nom}. Since the 1-bit sample rate is 32 to 256 higher than the nominal output rate, the Nyquist limit is 4 to 7 octaves above F_{nom}, significantly reducing the requirements for an anti-alias filter. The 1-bit Nyquist frequency for a decimation rate of 256 is shown by the heavy vertical line at the right of the graph. Though higher decimation rates are possible, they contribute little additional benefit because of the noise floor of the device. In the case of the 14-bit topology, a second-order modulator with a decimation rate of 256, the resolution is limited by the signal-to-noise ratio. To obtain repeatable 14-bit resolution in the measurement of DC or slow-moving signals, it is necessary to average multiple output samples or apply more sophisticated signal processing techniques.

Figure 6. Sinc^2 Decimation Filter Magnitude Response, with -3dB point and Nyquist Frequency



Unlike the earlier DELSIG8 and DELSIG11, this user module implements both the numerator and denominator of the transfer function entirely in hardware. This requires the improved “Type 2" decimator first introduced in the CY8C29x6x devices. It is used for both the first and second-order modulator topologies. The decimator implements the denominator of the transfer function by a double integrator operating at the 1-bit sample rate. The numerator is implemented by a double differentiator (second difference operator) that runs at the nominal output sample rate. The CPU overhead and interrupt latency consumed by the DelSig User Module is limited to the approximately 80 cycles or less required to retrieve the sample data from the decimator registers in I/O space. The Type 2 decimator natively produces an unsigned value ranging from 0 to 2^n-1 for an n-bit converter. The interrupt service routine can be configured to convert this into a 2’s complement value ranging from -2^{n-1} to $+2^{n-1}$-1.

Timing Generator and Requirements

The divide-by-four clock generator that supplies the Φ1 and Φ2 clocks to the analog modulator also provides a bit-clock to the decimator. The decimation factor corresponding output sample rate is determined by a word clock. The word clock is generated by one or two digital PSoC blocks configured as a PWM. The PWM output is automatically routed to the decimator and may be used externally. Data in the decimator is retrieved via an interrupt service routine that performs any necessary formatting and places the value in RAM. The interrupt occurs when the PWM reaches its terminal count and the PWM’s output produces a falling edge. The choice of a PWM for this function is purely arbitrary; a timer or counter could have been used as well. However, the PWM provides a variable duty cycle that can be used to advantage in some applications.

Because the bit clock produced by the analog clock generator is not generally available (except directly to the decimator), its divide-by-four operation must be accounted for by the PWM. Thus, the period of the PWM must be 4 times the decimation rate. Thus, for a decimation rate of 64, the period is set to 4 times 64 or 256. This requires an 8-bit PWM that can be implemented in a single digital PSoC block. Higher decimation rates up to the limit of 256 require two digital PSoC blocks. Though higher decimation rates are possible, their benefit is limited by the device noise floor.

If the PWM output is routed to other user modules or through a pin to an external sink, the duty cycle may be set at configuration time (see Parameters section) and modified at run-time (see Application Programming Interface section). The duty cycle should be limited to values from 1 to 1 less than the period.

Equation 4

$$1 \le CompareValue \le (4 \times DecimationRate) - 1$$

The resulting duty cycle may be computed from the following relation.

Equation 5

$$DutyCycle(\%) = 100 \times \frac{CompareValue}{4 \times DecimationRate}$$

For example, when the decimation rate is 256, a 50% duty cycle requires a CompareValue of 511. Note that setting the CompareValue should to zero produces a 0% duty cycle. Although the converter will continue to run, no word clock is generated. If the duty cycle is restored to a non-zero value, the decimation filter results will be invalid for the next two samples. Although it takes slightly longer to call the API DelSig_Stop function, this will consume less power then setting the duty cycle to zero.

Note **IMPORTANT:** When placing this module, it is imperative that it is configured with the same data clock for both the analog column clock and any digital blocks. Failure to do so will cause it to operate incorrectly.

DC and AC Electrical Characteristics

The following values are indicative of expected performance and based on initial characterization data. Unless otherwise specified below TA = 25 °C, VDD= 5.0 V, Power HIGH, Opamp Bias LOW, output referenced to 2.5 V external Analog Ground on P2[4] with 1.25 external VRef on P2[6].

Table 2. 5.0 V Second-Order Modulator DC and AC Electrical Characteristics

Parameter	Typical	Limit	Units	Conditions and Notes
Input				
Input Voltage Range	---	V_{SS} to V_{DD}	V	RefMux = $V_{DD}/2 \pm V_{DD}/2$
Input Capacitance	3	---	pF	Includes I/O pin.
Input Impedance	1/(C*clk)	---	Ω	
Effective Resolution Decimate by 32 Decimate by 64 Decimate by 128 Decimate by 256	---	 8 10 12 14	Bits	
Sample Rate Decimate by 32 Decimate by 64 Decimate by 128 Decimate by 256	---	 62,500 31,250 15625 7812	sps	Data Clock 8 MHz
DC Accuracy				
DNL Decimate by 32 Decimate by 64 Decimate by 128 Decimate by 256	 <1 <1 <1 0.6	---	LSB	Source Clock 1.5 MHz
Offset Error	13	---	mV	
Gain Error	2		% FSR	Including Reference Gain Error
Data Clock	---	0.032 to 8.0	MHz	Input to digital blocks and analog column clock

Table 3. 5.0 V First-Order Modulator DC and AC Electrical Characteristics

Parameter	Typical	Limit	Units	Conditions and Notes
Input				
Input Voltage Range	---	V_{SS} to V_{DD}	V	Ref Mux = $V_{DD}/2 \pm V_{DD}/2$
Input Capacitance	3	---	pF	Includes I/O pin.
Input Impedance	1/(C*clk)	---	Ω	
Effective Resolution Decimate by 32 Decimate by 64 Decimate by 128 Decimate by 256	---	 6 7.5 9 10.5	Bits	
Sample Rate Decimate by 32 Decimate by 64 Decimate by 128 Decimate by 256	---	 62,500 31,250 15625 7812	sps	Data Clock 8 MHz
DC Accuracy				
DNL Decimate by 32 Decimate by 64 Decimate by 128 Decimate by 256	 0.25 <1 <1 <1	---	LSB	Data Clock 1.5 MHz
Offset Error	5	---	mV	
Gain Error	3		% FSR	Including Reference Gain Error
Data Clock	---	0.032 to 8.0	MHz	Input to digital blocks and analog column clock

The following values are indicative of expected performance and based on initial characterization data. Unless otherwise specified below, TA = 25 °C, VDD = 3.3 V, Power HIGH, Op-Amp Bias LOW, output referenced to 1.64 V external Analog Ground on P2[4] with 1.25 external VRef on P2[6].

Table 4. 3.3 V Second-Order Modulator DC and AC Electrical Characteristics

Parameter	Typical	Limit	Units	Conditions and Notes
Input				
Input Voltage Range	---	V_{SS} to V_{DD}	V	Ref Mux = $V_{DD}/2 \pm V_{DD}/2$
Input Capacitance	3	---	pF	Includes I/O pin.
Input Impedance	1/(C*clk)	---	Ω	
Effective Resolution Decimate by 32 Decimate by 64 Decimate by 128 Decimate by 256	---	 8 10 12 14	Bits	
Sample Rate Decimate by 32 Decimate by 64 Decimate by 128 Decimate by 256	---	 62,500 31,250 15625 7812	sps	Data Clock 8 MHz
DC Accuracy				
DNL Decimate by 32 Decimate by 64 Decimate by 128 Decimate by 256	 <1 <1 <1 0.5	---	LSB	Data Clock 1.5 MHz
Offset Error	13	---	mV	
Gain Error	2		% FSR	Including Reference Gain Error
Data Clock	---	0.032 to 8.0	MHz	Input to digital blocks and analog column clock

Table 5. 3.3 V First-Order Modulator DC and AC Electrical Characteristics

Parameter	Typical	Limit	Units	Conditions and Notes
Input				
Input Voltage Range	---	V_{SS} to V_{DD}	V	RefMux = $V_{DD}/2 \pm V_{DD}/2$
Input Capacitance	3	---	pF	Includes I/O pin.
Input Impedance	1/(C*clk)	---	Ω	
Effective Resolution Decimate by 32 Decimate by 64 Decimate by 128 Decimate by 256	---	 6 7.5 9 10.5	Bits	
Sample Rate Decimate by 32 Decimate by 64 Decimate by 128 Decimate by 256	---	 62,500 31,250 15625 7812	sps	Data Clock 8 MHz
DC Accuracy				
DNL Decimate by 32 Decimate by 64 Decimate by 128 Decimate by 256	 0.25 <1 <1 <1	---	LSB	Data Clock 1.5 MHz
Offset Error	5	---	mV	
Gain Error	3		% FSR	Including Reference Gain Error
Data Clock	---	0.032 to 8.0	MHz	Input to digital blocks and analog column clock

Placement

When the DelSig User Module is selected in the tool bar or by double-clicking its icon in the selector view, a selection window opens that provides guidance in selecting the appropriate topology. The topology may be changed at any later time by right-clicking on the user module in the placement view and choosing “User Module Selection Options..." from the context menu.

The first-order modulator design requires two PSoC blocks, one digital and one analog. The digital block, named “PWM" may be placed in any digital PSoC Block. Similarly, the analog block, “ADC" may be placed in any switched capacitor PSoC block.

The second-order modulator design uses two switched capacitor PSoC blocks, ADC1, and ADC2. Because the analog comparator bus that connects them runs vertically in each column of the analog array, the switched capacitor PSoC blocks must be placed vertically, one above the other. The digital block or blocks may be placed anywhere in the digital rows.

Note **IMPORTANT:** When placing this module, it is imperative that it is configured with the same data clock for both the analog column clock and any digital blocks. Failure to do so will cause it to operate incorrectly.

Although there are many placements possible for the analog and digital blocks, the DelSig also uses the PSoC device's only hardware decimation filter. The decimator is automatically allocated when the analog blocks are placed; no additional action is necessary. Because of this, only one instance of the DelSig User Module may be placed in a given configuration. With dynamic reconfiguration it is possible to load (activate) more than one configuration at a time and there is no check performed that would prevent two DelSig User Modules from operating at the same time. If this occurs, both instances may appear to work; however, only the instance most recently loaded will control the decimation filter. Both interrupts may still operate, possibly interfering.

Parameters and Resources

After a DelSig instance is placed, four parameters must be configured for proper operation: the Input Signal Multiplexer selection, the Clock Phase, the Data Clock, and the Polling selection

DataFormat

This parameter may take the values of Unsigned or Signed (default). Unsigned data will take values from zero to 2^n-1 for n-bits of resolution. Signed data will range in value from -2^{n-1} to $+2^{n-1}$-1

Data Clock

The Data Clock determines the sample rate (see Equation 1). This clock goes to both PSoC blocks of the first-order modulator design and to all three PSoC blocks of the second-order designs.

Note **IMPORTANT:** When placing this module, it is imperative that it is configured with the same data clock for both the analog column clock and any digital blocks. Failure to do so will cause it to operate incorrectly.

The PWM data clock is configured by selecting a source such as VC3 or a row input from the User Module Parameters grid in the device editor. Selecting the Analog Column clock requires configuring one or two multiplexors in the placer view. If the data clock is derived from a digital PSoC block, one of the two AnalogClock_Select multiplexors must be configured to select that block. The Analog_ColumnClock mux above the modulator column must be configured to select either the appropriate AnalogClock_Select multiplexor or VC1 or VC2.

The PWM provides an interrupt every 4*n counts of the data clock, where n is the decimation rate. On each interrupt, the data is extracted from the decimator and made available through the various API "GetData" functions. The sample rate is thus defined as:

Equation 6

$$SampleRate = \frac{DataClock}{4 \times DecimationRate}$$

Clock Phase

The selection of the Clock Phase is used to synchronize the output of one analog PSoC block to the input of another. The switched capacitor analog PSoC blocks use a two-phase clock ($\Phi1$, $\Phi2$) to acquire and transfer signals. Normally, the input to the DelSig is sampled on $\Phi1$. A problem arises in that many of the user modules auto-zero their output during $\Phi1$ and only provide a valid output during $\Phi2$. If such a module's output is fed to the DelSig's input, the DelSig will sample an indeterminate value. The Clock Phase selection allows the phases to be swapped, so that the input signal is acquired during $\Phi2$.

PosInput

This parameter determines the signal source for singe-ended inputs, or the non-inverting input for differential inputs.

NegInput and NegInputGain

NegInput selects the source for the inverting input of a differential signal pair. When a single-ended input is used, this parameter may be set to any legal value. It is disconnected from the converter by setting the NegInputGain parameter to “Disconnected" (zero gain).

NegInputGain adjusts the gain of the inverting input (see NegInput parameter, previous paragraph) relative to the non-inverting input. For a single-ended input, this parameter should take the value “Disconnected. For differential inputs the NegInputGain can be set to 1.000. If desired, the gain applied to the inverting input can also be adjusted in 1/16-th increments between 0.0625 and 1.9375 relative to the non-inverting input.

PWM Output

Synchronized access DelSig converters employ a Pulse-Width Modulator function to provide a time base for the hardware $sinc^2$ filter (decimator unit) and to provide an interrupt for extracting samples. The output of this PWM can optionally be routed onto the row output busses by setting this parameter to the name of the desired bus. Output samples are produced by the interrupt service routine when the PWM reaches its terminal count and the PWM Output transitions from high to low.

PulseWidth

This parameter establishes the initial duty cycle of the PWM output. It can be configured in the device editor to any value from 1 up to (4*n)-1 where n is the Decimation rate. It can be set to zero using the API. This produces a 0% duty cycle, but it deactivates processing of the output samples but leaves the converter running. Other values result in non-zero duty cycles as described in Equation 4 and Equation 5.

Interrupt Generation Control

There is an additional parameter that becomes available when the **Enable interrupt generation control** check box in PSoC Designer is checked. This is available under **Project > Settings > Chip Editor**. Interrupt Generation Control is important when multiple overlays are used with interrupts shared by multiple user modules across overlays:

IntDispatchMode

The IntDispatchMode parameter is used to specify how an interrupt request is handled for interrupts shared by multiple user modules existing in the same block but in different overlays. Selecting “ActiveStatus" causes firmware to test which overlay is active before servicing the shared interrupt request. This test occurs every time the shared interrupt is requested. This adds latency and also produces a nondeterministic procedure of servicing shared interrupt requests, but does not require any RAM. Selecting “OffsetPreCalc" causes firmware to calculate the source of a shared interrupt request only when an overlay is initially loaded. This calculation decreases interrupt latency and produces a deterministic procedure for servicing shared interrupt requests, but at the expense of a byte of RAM.

Application Programming Interface

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

Each time a user module is placed, it is assigned an instance name. By default, PSoC Designer assigns the DelSig_1 to the first instance of this user module in a given project. It can be changed to any unique value that follows the syntactic rules for identifiers. The assigned instance name becomes the prefix of every global function name, variable and constant symbol. In the following descriptions the instance name has been shortened to DelSig for simplicity.

DelSig_Start

Description:

Performs all required initialization for this user module and sets the power level for the switched capacitor PSoC block

C Prototype:

```
void  DelSig_Start (BYTE bPowerSetting)
```

Assembly:

```
mov   A, bPowerSetting
lcall  DelSig_Start
```

Parameters:

bPowerSetting:One byte that specifies the power level. Following reset and configuration, the analog PSoC block assigned to DelSig is powered down. Symbolic names provided in C and assembly, and their associated values are given in the following table.

Symbolic Name	Value
DelSig_OFF	0
DelSig_LOWPOWER	1
DelSig_MEDPOWER	2
DelSig_HIGHPOWER	3

Return Value:

None

Side Effects:

You can modify the A and X registers by this or future implementations of this function. The same is true for all RAM page pointer registers in the Large Memory Model (CY8C29xxx). When necessary, it is the calling function's responsibility to preserve the values across calls to fastcall16 functions.

DelSig_Stop

Description:

Sets the power level to the switched capacitor PSoC block to OFF.

C Prototype:

```
void  DelSig_Stop (void)
```

Assembly:

```
lcall  DelSig_Stop
```

Parameters:

None

Return Value:

None

Side Effects:

You can modify the A and X registers by this or future implementations of this function. The same is true for all RAM page pointer registers in the Large Memory Model (CY8C29xxx). When necessary, it is the calling function's responsibility to preserve the values across calls to fastcall16 functions.

DelSig_SetPower

Description:

Sets the power level for the switched capacitor PSoC block.

C Prototype:

```
void  DelSig_SetPower (BYTE bPowerSetting)
```

Assembly:

```
mov   A, bPowerSetting
lcall  DelSig_SetPower
```

Parameters:

bPowerSetting: Same as the bPowerSetting parameter used for the Start entry point.

Return Value:

None

Side Effects:

You can modify the A and X registers by this or future implementations of this function. The same is true for all RAM page pointer registers in the Large Memory Model (CY8C29xxx). When necessary, it is the calling function's responsibility to preserve the values across calls to fastcall16 functions.

DelSig_StartAD

Description:

Enables the timer and the integrator.

C Prototype:

```
void  DelSig_StartAD (void)
```

Assembly

```
lcall  DelSig_StartAD
```

Parameters:

None

Return Value:

None

Side Effects:

You can modify the A and X registers by this or future implementations of this function. The same is true for all RAM page pointer registers in the Large Memory Model (CY8C29xxx). When necessary, it is the calling function's responsibility to preserve the values across calls to fastcall16 functions.

DelSig_StopAD

Description:

Disables the timer and resets the integrator.

C Prototype:

```
void  DelSig_StopAD (void)
```

Assembly:

```
lcall  DelSig_StopAD
```

Parameters:

None

Return Value:

None

Side Effects:

You can modify the A and X registers by this or future implementations of this function. The same is true for all RAM page pointer registers in the Large Memory Model (CY8C29xxx). When necessary, it is the calling function's responsibility to preserve the values across calls to fastcall16 functions.

DelSig_fIsDataAvailable

Description:

Checks the availability of sampled data.

C Prototype:

```
BYTE  DelSig_fIsDataAvailable(void)
```

Assembly:

```
lcall  DelSig_fIsDataAvailable
cmp   A, 0
jz    .DataNotAvailable
```

Parameters:

None

Return Value:

Returns a non-zero value if data has been converted and is ready to read.

Side Effects:

You can modify the A and X registers by this or future implementations of this function. The same is true for all RAM page pointer registers in the Large Memory Model (CY8C29xxx). When necessary, it is the calling function's responsibility to preserve the values across calls to fastcall16 functions. Currently, only the CUR_PP page pointer register is modified.

DelSig_cGetData

DelSig_iGetData

Description:

Returns converted data as a signed CHAR or INT. Note that the user module DataFormat parameter determines the underlying representation. Calling a signed format function does not change the value of the data when the underlying representation is unsigned. DataFormatDelSig_fIsDataAvailable() may be called to verify that the data sample is ready.

C Prototypes:

```
CHAR DelSig_cGetData(void)      // use for 8-bit resolution or lower
INT  DelSig_iGetData(void)      // use for 9-bit resolution or higher
```

Assembly:

```
lcall  DelSig_cGetData           ; Result will be in A
- or -
lcall  DelSig_iGetData           ; LSB will be in A, MSB in X upon return
```

Parameters:

None

Return Value:

Returns the converted data sample in 8-bit or 16-bit 2’s complement format.

Side Effects:

You can modify the A and X registers by this or future implementations of this function. The same is true for all RAM page pointer registers in the Large Memory Model (CY8C29xxx). When necessary, it is the calling function's responsibility to preserve the values across calls to fastcall16 functions. Currently, only the CUR_PP page pointer register is modified.

DelSig_bGetData

DelSig_wGetData

Description:

Returns converted data as an unsigned BYTE or WORD. Note that the user module DataFormat parameter determines the underlying representation. Calling an unsigned format function does not change the value of the data when the underlying representation is signed. DataFormatDelSig_fIsDataAvailable() may be called to verify that the data sample is ready.

C Prototypes:

```
BYTE DelSig_bGetData(void)      // use for 8-bit resolution or lower
WORD DelSig_wGetData(void)      // use for 9-bit resolution or higher
```

Assembly:

```
lcall  DelSig_bGetData           ; Result will be in A
- or -
lcall  DelSig_wGetData           ; LSB will be in A, MSB in X upon return
```

Parameters:

None

Return Value:

Returns the converted data sample in 8-bit or 16-bit 2’s complement format according to the function.

Side Effects:

You can modify the A and X registers by this or future implementations of this function. The same is true for all RAM page pointer registers in the Large Memory Model (CY8C29xxx). When necessary, it is the calling function's responsibility to preserve the values across calls to fastcall16 functions. Currently, only the CUR_PP page pointer register is modified.

DelSig_ClearFlag

Description:

Resets the data available flag.

C Prototype:

```
void  DelSig_ClearFlag(void)
```

Assembly:

```
lcall  DelSig_ClearFlag
```

Parameters:

None

Return Value:

None

Side Effect:

You can modify the A and X registers by this or future implementations of this function. The same is true for all RAM page pointer registers in the Large Memory Model (CY8C29xxx). When necessary, it is the calling function's responsibility to preserve the values across calls to fastcall16 functions. Currently, only the CUR_PP page pointer register is modified.

DelSig_cGetDataClearFlag

DelSig_iGetDataClearFlag

Description:

Returns converted data as a signed CHAR or INT and resets the data available flag. Note that the user module DataFormat parameter determines the underlying representation. Calling an unsigned format function does not change the value of the data when the underlying representation is signed. DataFormatDelSig_fIsDataAvailable() may be called to verify that the data sample is ready.

C Prototype:

```
CHAR DelSig_cGetDataClearFlag(void)   // use for 8-bit resolution or lower
INT  DelSig_iGetDataClearFlag(void)   // use for 9-bit resolution or higher
```

Assembly:

```
lcall  DelSig_cGetDataClearFlag   ; Result will be in A
- or -
lcall  DelSig_iGetDataClearFlag   ; LSB will be in A, MSB in X upon return
```

Parameters:

None

Return Value:

Returns the converted data sample in 8-bit 2’s complement format.

Side Effects:

You can modify the A and X registers by this or future implementations of this function. The same is true for all RAM page pointer registers in the Large Memory Model (CY8C29xxx). When necessary, it is the calling function's responsibility to preserve the values across calls to fastcall16 functions. Currently, only the CUR_PP page pointer register is modified.

DelSig_bGetDataClearFlag

DelSig_wGetDataClearFlag

Description:

Returns converted data as an unsigned BYTE or WORD and resets the data available flag. Note that the user module DataFormat parameter determines the underlying representation. Calling an unsigned format function does not change the value of the data when the underlying representation is signed. DataFormatDelSig_fIsDataAvailable() may be called to verify that the data sample is ready.

C Prototype:

```
BYTE DelSig_bGetDataClearFlag(void)   // use for 8-bit resolution or lower
WORD DelSig_wGetDataClearFlag(void)   // use for 9-bit resolution or higher
```

Assembly:

```
lcall  DelSig_bGetDataClearFlag   ; Result will be in A
- or -
lcall  DelSig_wGetDataClearFlag   ; LSB will be in A, MSB in X upon return
```

Parameters:

None

Return Value:

Returns the converted data sample in 8-bit 2’s complement format.

Side Effects:

You can modify the A and X registers by this or future implementations of this function. The same is true for all RAM page pointer registers in the Large Memory Model (CY8C29xxx). When necessary, it is the calling function's responsibility to preserve the values across calls to fastcall16 functions. Currently, only the CUR_PP page pointer register is modified.

DelSig_WritePulseWidth

Description:

Sets the PWM compare register, DR2, to the value of the actual parameter.The resulting duty cycle is given by Equation 5. Proper operation requires a duty cycle greater than 0% and less than 100%. Zero and 100% duty cycles cause the ADC to stop producing samples. Appropriate pulse width values for each decimation rate are given in the following table.

Decimation Rate	Min	Max	Type	Duty-Cycle Range
32	1	127	BYTE	0.8 - 99.2%
64	1	255	BYTE	0.4 - 99.6%
128	1	511	WORD	0.2 - 99.8%
256	1	1023	WORD	0.1 - 99.9%

C Prototype:

```
void DelSig_WritePulseWidth(BYTE)   // applies to decimate by  32 or  64
- or -
void DelSig_WritePulseWidth(WORD)   // applies to decimate by 128 or 256
```

Assembly:

```
mov   A, [bPulseWidth]                // use for decimate by  32 or  64
lcall  DelSig_WritePulseWidth   ;
- or -
mov   A, [wPulseWidth+1]              // use for decimate by 128 or 256
mov   X, [wPulseWidth]                // MSB in X, LSB in A
lcall  DelSig_WritePulseWidth
```

Parameters:

8-bit unsigned bPulseWidth or16-bit unsigned wPulseWidth

Return Value:

None.

Side Effects:

You can modify the A and X registers by this or future implementations of this function. The same is true for all RAM page pointer registers in the Large Memory Model. When necessary, it is the calling function's responsibility to preserve the values across calls to fastcall16 functions.

Sample Firmware Source Code

This example repeats the previous scenario but uses API functions rather than direct references to the global variables.

Here is an assembly language example:

```
;-------------------------------------------------------------------------------
; Assembly main line
;-------------------------------------------------------------------------------

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
include "DelSig.inc"

export _main

_main:
    M8C_EnableGInt ; enable global interrupts
    mov A, DelSig_HIGHPOWER ; Establish power setting...
    call DelSig_Start ; and initialize
    call DelSig_StartAD ; Commence sampling process

    mov A,PGA_HIGHPOWER ; Set Power level of PGA
    call PGA_Start ; Initialize PGA
    call LCD_Start ; Initialize LCD)

mainloop:
    call DelSig_fIsDataAvailable ; Retrieve the status byte
    cmp A, 0 ; check is data available
    jz mainloop ; spin lock until(data is Available)
    mov A, 00h ; Set cursor position at row=0
    mov X, 01h ; set cursor position to col=1
    call LCD_Position
    call DelSig_iGetDataClearFlag ; fastcall convention puts data in X, A
    call LCD_PrHexInt ; Call a routine to display the A and X registers contents on LCD
    jmp mainloop
```

The equivalent code in C is:

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

void ProcessSample( int iSample )
{
    ; // (Do something useful with the data)
}

void main(void)
{
   M8C_EnableGInt;
   DelSig_Start( DelSig_HIGHPOWER );
   DelSig_StartAD();
   while (1) {
      if ( DelSig_fIsDataAvailable() ) {
```

```
            ProcessSample( DelSig_iGetDataClearFlag() );
        }
    }
}
```

Configuration Registers

Analog Registers, 1st-Order Modulator

Table 6. Registers used by the “ADC" Analog Switched Capacitor PSoC Block

Register	7	6	5	4	3	2	1	0
CR0	1	0	0	1	0	0	0	0
CR1	PosInput			InvertingGain				
CR2	0	1	0	0	0	0	0	0
CR3	1	1	0	0	NegInput		Power	

PosInput selects the single-ended input signal or the non-inverting input of a differential input signal. NegInput selects the inverting input of a differential input. The inverting input is disconnected when ever the InvertingGain field is set to zero. Power is set by the DelSig_Start and DelSig_SetPower API functions.

Analog Registers, 2nd-Order Modulator

Table 7. Registers used by the “ADC1" and “ADC2" Analog Switched Capacitor PSoC Block

Register	7	6	5	4	3	2	1	0
ADC1CR0	1	0	0	0	1	0	0	0
ADC1CR1	PosInput			InvertingGain				
ADC1CR2	0	1	0	0	0	0	0	0
ADC1CR3	1	1	0	0	NegInput		Power	
ADC2CR0	1	0	0	1	0	0	0	0
ADC2CR1	LinkToADC1			0	0	0	0	0
ADC2CR2	0	1	0	0	0	0	0	0
ADC2CR3	1	1	0	0	0	0	Power	

PosInput selects the single-ended input signal or the non-inverting input of a differential input signal. NegInput selects the inverting input of a differential input. The inverting input is disconnected when ever the InvertingGain field is set to zero. LinktoADC1 is determined by block placement and connects the output of the ADC1 block to the “A" input capacitor of the ADC2 PSoC block. Power is set by the DelSig_Start and DelSig_SetPower API functions.

Digital PSoC Block Registers, Decimate by 32 and 64

Table 8. Registers used by the PWM Digital PSoC Block

Register	7	6	5	4	3	2	1	0
Function	0	0	1	0	0	0	0	0
Input	0	0	0	1	Clock			
Output	0	0	0	0	0	0	0	0
DR0	PWM Down Count Value (Never Accessed by the API)							
DR1	1	1	1	1	1	1	1	1
DR2	PWM Pulse-Width Value							
CR0	0	0	1	0	0	0	0	Enable

The PWM is a digital PSoC block configured with a period value equal to 1 less than four times the decimation rate (i.e. for a decimation rate of 64, the value is 0xFF = 255 = (64*4) - 1). At the interrupt on terminal count, the decimator is read and the data-ready flag is set. Clock selects the input clock from one of 16 sources. This parameter is set in the Device Editor. *Note, the source chosen must also be used to control the analog clock for the column in with the ADC block resides.* Enable turns the PWM on when set. It is modified and controlled by the DelSig StartAD and StopAD API functions.

Digital PSoC Block Registers, Decimate by 128 and 256

Table 9. Registers used by the PWM Digital PSoC Block

Register	7	6	5	4	3	2	1	0
LSB FN	0	0	1	0	0	0	0	0
LSB IN	0	0	0	1	Clock			
LSB OUT	0	0	0	0	0	0	0	0
LSB DR0	PWM Down Count Value (Never Accessed by the API)							
LSB DR1	1	1	1	1	1	1	1	1
LSB DR2	PWM Pulse-Width Value							
LSB CR0	0	0	1	0	0	0	0	0
MSB FN	0	0	1	0	0	0	0	0
MSB IN	0	0	0	1	0	0	1	1
MSB OUT	0	0	0	0	0	0	0	0
MSB DR0	PWM Down Count Value (Never Accessed by the API)							
MSB DR1	1	1	1	1	1	1	1	1
MSB DR2	PWM Pulse-Width Value							
MSB CR0	0	0	1	0	0	0	0	Enable

The PWM is a digital PSoC block configured with a period value equal to 1 less than four times the decimation rate (i.e. for a decimation rate of 128, the value is 0x01FF = 511 = (128*4) - 1). At the interrupt on terminal count, the decimator is read and the data-ready flag is set. Clock selects the input clock from one of 16 sources. This parameter is set in the Device Editor. *Note, the source chosen must also be used to control the analog clock for the column in with the ADC block resides.* Enable turns the PWM on when set. It is modified and controlled by the DelSig StartAD and StopAD API functions.

Decimator Control Registers

Table 10. Decimation Control Registers

Bit	7	6	5	4	3	2	1	0
DEC_CR0	0	0	0	0	0	DCol		DCLKSEL
DEC_CR1	0	1	0	0	0	DCLKSEL		
DEC_CR2	1	0	Shift		1	DecimationRate		
DEC_DH	High Byte Output of Decimator							
DEC_DL	Low Byte Output of Decimator							

The decimator is dedicated hardware used to implement a Sinc2 filter. It consists of three control registers and two data output registers. DCol selects which column comparator is connected. DCLKSEL selects which digital block is used to control the decimator timing. Both parameters are set in Device Editor. Shift, in DEC_CR2, is set according to the decimation rate, also specified in DEC_CR2, to minimize the data aligned that must be accomplished in software.

Version History

Version	Originator	Description
1.2	DHA	Added a DRC to check if: a. The source clock is different in digital and analog resources. b. The ADC Clock is higher than the CPU Clock.
1.30	DHA	Restored VC3 as the source for data clock.
1.40	DHA	Fixed interrupt service routine to prevent data corruption.
1.50	MYKZ	1. Added design rules check for the situation when the ADC clock is faster than 8 MHz. 2. Corrected the method of clearing posted interrupts.

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.

Copyright © 2002-2013 Cypress Semiconductor Corporation. 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 Designer™ and Programmable System-on-Chip™ are trademarks and PSoC® is a registered trademark 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.