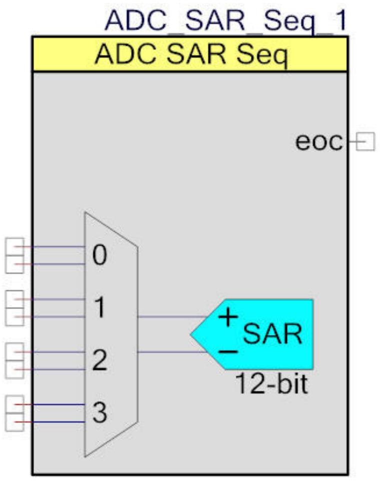
Sequencing Successive Approximation ADC (ADC_SAR_Seq)

1.0

Features



- Supports PSoC 5LP devices
- Selectable resolution (8, 10 or 12 bit) and sample rate (up to 1 Msps)
- Scans up to 64 single ended or 32 differential channels automatically, or just a single input

  Note The actual maximum number of input channels depends on the number of routable analog GPIOs that are available on a specific PSoC part and package.

General Description

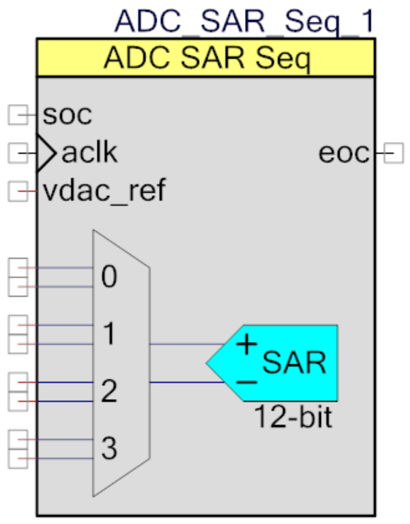
The Sequencing SAR ADC component enables makes it possible for you to configure and then use the different operational modes of the SAR ADC on PSoC 5LP. You also have schematic level and firmware level support for seamless use of the Sequencing SAR ADC in PSoC Creator designs and projects. You are able to configure multiple analog channels that are automatically scanned with the results placed in individual SRAM locations.

When to Use the ADC_SAR_Seq

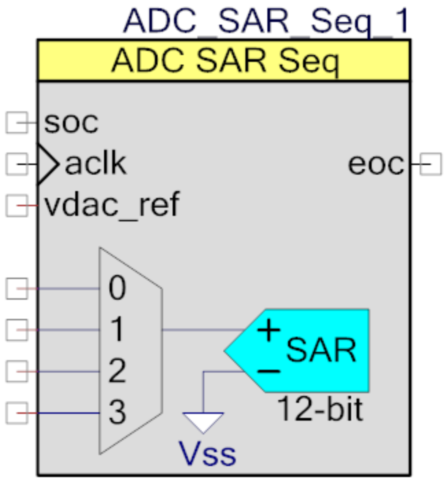
The Sequencing SAR ADC is commonly used in high sample rate systems where multiple channels must be sampled without CPU intervention until all channels are sampled.

Input/Output Connections

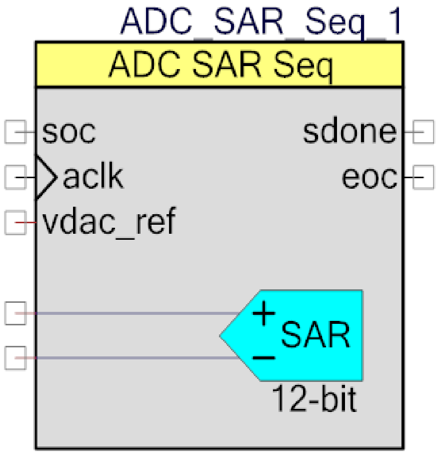
This section describes the input and output connections for the ADC_SAR_Seq. An asterisk (*) in the list of I/Os indicates that the I/O may be hidden on the symbol under the conditions listed in the description of that I/O.



Differential Inputs



Single-ended inputs



Differential inputs – 1 channel

(SDONE is visible)

+Input – Analog

This input is the positive analog signal input to the ADC_SAR_Seq. The conversion result is a function of the +Input signal minus the voltage reference. The voltage reference is either the Input signal or V_{SSA}.

–Input – Analog *

When shown, this optional input is the negative analog signal (or reference) input to the ADC_SAR_Seq. The conversion result is a function of +Input minus –Input. You see this pin when you set the **Input Range** parameter to one of the differential modes.

vdac_ref – Input *

The VDAC reference (vdac_ref) is an optional pin. You see it if you select **Vssa to VDAC*2 (Single Ended)** or **0.0 +/- VDAC (Differential)** input range; otherwise, this I/O is hidden.

> **Note** You can only connect this pin to a VDAC component output. Do not connect it to any other signal.

soc – Input *

The start of conversion (soc) is an optional pin. You see it if you select the **Hardware** t**rigger** sample mode. A rising edge on this input starts an ADC conversion. Synchronize this signal to the ADC_SAR_Seq clock. If you set the **Sample Mode** parameter to **Free-running** or **Software trigger** this I/O is hidden.

aclk – Input *

You see this optional pin if you set the **Clock Source** parameter to **External**; otherwise, the pin is hidden. This clock determines the conversion rate as a function of the conversion method and resolution.

eoc – Output

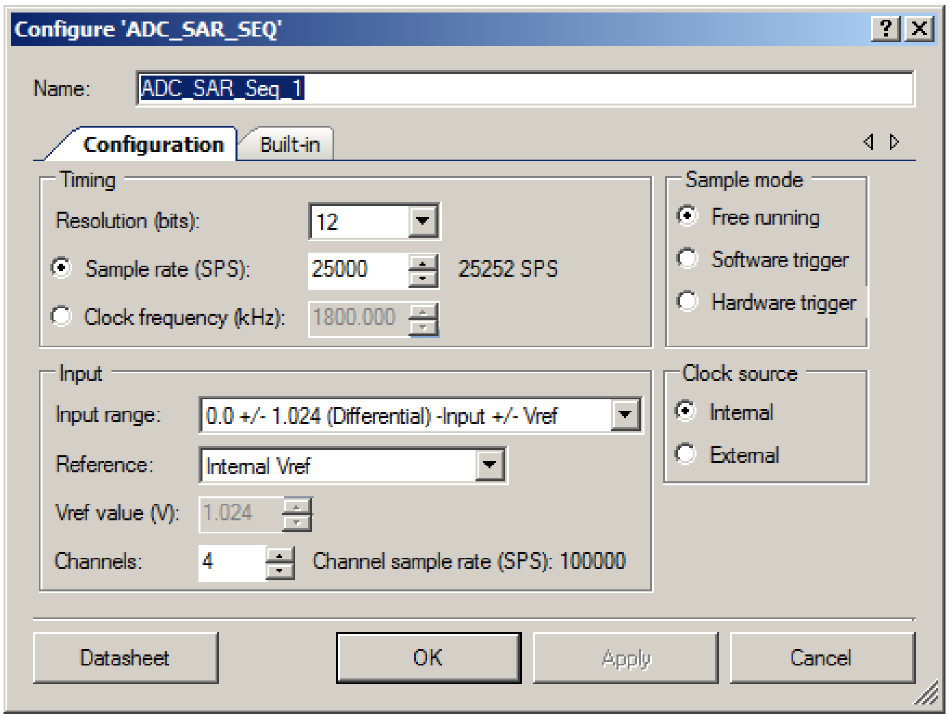
A rising edge on the end of conversion (eoc) output means that one conversion cycle is complete. At this moment, conversion results for all channels are ready to be read from the SRAM. An internal interrupt is also connected to this signal, or you may connect your own interrupt.

eos – Output

A rising edge on the end of sampling (eos) output means that sampling is complete. This is a direct connection from EOS output of the ADC SAR component. This signal is only visible when the component is configured for one channel operation.

Component Parameters

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



Name: ADC_SAR_Seq_1

Timing
Resolution (bits): 12
(●) Sample rate (SPS): 25000 25252 SPS
( ) Clock frequency (kHz): 1800.000

Sample mode
(●) Free running
( ) Software trigger
( ) Hardware trigger

Input
Input range: 0.0 +/- 1.024 (Differential) -Input +/- Vref
Reference: Internal Vref
Vref value (V): 1.024
Channels: 4 Channel sample rate (SPS): 100000

Clock source
(●) Internal
( ) External

The ADC_SAR_Seq has these parameters. The option shown in bold is the default.

Modes

Resolution

Sets the resolution of the ADC_SAR_Seq.

Resolution	Description
12	Sets resolution to 12 bits
10	Sets resolution to 10 bits
8	Sets resolution to 8 bits

Conversion Rate

This parameter sets the ADC conversion rate. The conversion time is the inverse of the conversion rate. Enter the conversion rate in samples per second. Converting one sample takes 18 component clock cycles.

Clock Frequency

This text box is a read-only (always grayed out) area that displays the required clock rate for the selected operating conditions: resolution and conversion rate. It is updated when either or both of these conditions change. Clock frequency can be anywhere between 1 MHz and 18 MHz. The duty cycle should be 50 percent. Make the minimum pulse width greater than 25.5 ns. PSoC Creator generates an error during the build process if the clock does not fall within these limits. In there is an error, change the Master Clock in the Design-Wide Resources clock editor.

Sample Mode

This parameter determines how the ADC operates.

SampleMode	Description
Free Running	ADC_SAR_Seq runs continuously
Software trigger	Calling of the ADC_SAR_Seq_StartConvert() starts a single cycle of conversion for all channels
Hardware trigger	A rising-edge pulse on the SOC pin starts a single cycle of conversion for all channels

Clock Source

This parameter allows you to select either a clock that is internal to the ADC_SAR_Seq module or an external clock.

ADC_Clock	Description
Internal	Use the internal clock of the ADC_SAR_Seq
External	Use an external clock. The clock source can be analog, digital, or generated by another component.

Input

Input Range

This parameter configures the ADC for a given input range. The analog signals connected to the PSoC must be between V_{SSA} and V_{DDA} regardless of the input range settings.

InputRange	Description
0.0 to 2.048V (Single Ended) **0 to Vref*2**	When using the internal reference (1.024 V), the usable input range is 0.0 to 2.048 V. The ADC is configured to operate in a single-ended input mode with –Input connected internally to Vrefhi_out. If you are using an external reference voltage, the usable input range is 0.0 to Vref*2.

InputRange	Description
Vssa to Vdda (Single Ended)	This mode uses the V_{DDA}/2 reference; the usable input range covers the full analog supply voltage. The ADC is put in a single-ended input mode with –Input connected internally to Vrefhi_out. If you are using an external reference voltage, the usable input range is 0.0 to Vref*2.
Vssa to VDAC*2 (Single Ended)	This mode uses the VDAC reference, which should be connected to the vdac_ref pin. The usable input range is Vssa to VDAC*2 volts. The ADC is configured to operate in a single-ended input mode with –Input connected internally to Vrefhi_out.
0.0 ± 1.024V (Differential) –Input ± Vref	This mode is configured for differential inputs. When using the internal reference (1.024 V), the input range is –Input ± 1.024 V. For example, if –Input is connected to 2.048 V, the usable input range is 2.048 ± 1.024 V or 1.024 to 3.072 V. For systems in which both single-ended and differential signals are scanned, connect –Input to Vssa when scanning a single-ended input. You can use an external reference to provide a wider operating range. You can calculate the usable input range with the same equation, –Input ± Vref.
0.0 ± Vdda (Differential) –Input ± Vdda	This mode is configured for differential inputs and is ratiometric with the supply voltage. The input range is –Input ± Vdda. For systems in which both single-ended and differential signals are scanned, connect –Input to Vssa when scanning a single-ended input. If you are using an external reference voltage, the usable input range is –Input ± Vref.
0.0 ± Vdda/2 (Differential) –Input ± Vdda/2	This mode is configured for differential inputs and is ratiometric to the supply voltage. The input range is –Input ± Vdda/2. For systems in which both single-ended and differential signals are scanned, connect –Input to Vssa when scanning a single-ended input. If you are using an external reference voltage, the usable input range is –Input ± Vref.
0.0 ± VDAC (Differential) –Input ± VDAC	This mode is configured for differential inputs and uses the VDAC reference, which should be connected to the vdac_ref pin. The input range is –Input ±VDAC. For systems in which both single-ended and differential signals are scanned, connect –Input to Vssa when scanning a single-ended input.

Reference

This parameter selects the switches for reference configuration for the ADC_SAR.

Reference	Description
Internal Vref	Uses the internal reference. The maximum sampling rate allowed with this option is 100,000 sps. Use the **Internal Vref, bypassed** option for higher rates.
Internal Vref, bypassed	Uses the internal reference; you must place a bypass capacitor on pin P0[2]* for SAR1 or on pin P0[4]* for SAR0.
External Vref	Uses an external reference on pin P0[2] for SAR1 or on pin P0[4] for SAR0.

* The use of an external bypass capacitor is recommended if the internal noise caused by digital switching exceeds an application's analog performance requirements. To use this option, configure either port pin P0[2] or P0[4] as an analog HI-Z pin and connect an external capacitor with a value between 0.01 µF and 10 µF.

Note The same internal reference is used for ADC_SAR and for ADC_DelSig components. If both types of the ADC need to work with an internal reference simultaneously, use the **Internal Vref, bypassed** option for the best performance.

Voltage Reference

The voltage reference is used for the ADC count to voltage conversion functions that is discussed in the Application Programming Interface section. This parameter is read-only when using the internal reference. When using an external reference, change this value to match the external reference voltage.

- When selecting input range **Vssa to Vdda**, **-Input +/- Vdda**, or **-Input +/- Vdda/2**, the value is derived from the V_{DDA} setting in System tab of the Design Wide Resources (DWR).
- When selecting the input range **Vssa to VDAC*2** or **–Input +/- VDAC**, type the VDAC supply voltage value.

Note The input range and reference voltage is limited by the V_{DDA} voltage.

Application Programming Interface

Application Programming Interface (API) routines allow you to configure the component using software. This table lists and describes the interface to each function. The following sections discuss each function in more detail.

By default, PSoC Creator assigns the instance name " ADC_SAR_Seq _1" to the first instance of a component in a given design. You can rename the instance 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. For readability, the instance name used in the following table is "ADC_SAR_Seq".

Function	Description
ADC_SAR_Seq _Start()	Powers up the ADC_SAR_Seq and resets all states
ADC_SAR_Seq _Stop()	Stops ADC_SAR_Seq conversions and reduces the power to the minimum
ADC_SAR_Seq _SetResolution()	Sets the resolution of the ADC_SAR_Seq
ADC_SAR_Seq _StartConvert()	Starts conversions
ADC_SAR_Seq _StopConvert()	Stops conversions
ADC_SAR_Seq _IRQ_Enable()	An internal IRQ is connected to the eoc. This API enables the internal ISR
ADC_SAR_Seq _IRQ_Disable()	An internal IRQ is connected to the eoc. This API disables the internal ISR
ADC_SAR_Seq _IsEndConversion()	Returns a nonzero value if conversion is complete
ADC_SAR_Seq _GetAdcResult()	Returns a signed 16-bit conversion result available in the ADC SAR Data Register not the result buffer

Function	Description
ADC_SAR_Seq _GetResult16()	Returns a signed16-bit conversion result for specified channel
ADC_SAR_Seq _SetOffset()	Sets the offset of the ADC_SAR_Seq
ADC_SAR_Seq _SetGain()	Sets the ADC_SAR_Seq gain in counts per volt
ADC_SAR_Seq _CountsTo_Volts()	Converts ADC_SAR_Seq counts to floating-point volts
ADC_SAR_Seq _CountsTo_mVolts()	Converts ADC_SAR_Seq counts to millivolts
ADC_SAR_Seq _CountsTo_uVolts()	Converts ADC_SAR_Seq counts to microvolts
ADC_SAR_Seq _Sleep()	Stops ADC_SAR_Seq operation and saves the user configuration
ADC_SAR_Seq _Wakeup()	Restores and enables the user configuration
ADC_SAR_Seq _Init()	Initializes the default configuration provided with the customizer
ADC_SAR_Seq _Enable()	Enables the clock and power for the ADC_SAR_Seq
ADC_SAR_Seq _SaveConfig()	Saves the current user configuration
ADC_SAR_Seq _RestoreConfig()	Restores the user configuration

Global Variables

Variable	Description
ADC_SAR_Seq _initVar	This variable indicates whether the ADC has been initialized. The variable is initialized to 0 and set to 1 the first time ADC_SAR_Seq _Start() is called. This allows the component to restart without reinitialization after the first call to the ADC_SAR_Seq _Start() routine. If reinitialization of the component is required, then the ADC_SAR_Seq _Init() function can be called before the ADC_SAR_Seq _Start() or ADC_SAR_Seq _Enable() functions.
ADC_SAR_finalArray	This array contains valid conversion results for all channels each time after EOC pulse has been generated and EOC status is set. **Note** When using the values from this array directly (not using ADC_SAR_Seq_GetResult16() API function), the fact should be taken into account, that channels are scanned in a reverse order, so the conversion result for the last channel will be placed at ADC_SAR_finalArray[0]. Do not use this array directly in any of Differential modes because the ADC_SAR_offset is not taken into account. Use ADC_SAR_Seq_GetResult16(), where it is handled. If used directly, you must manually handle the ADC_SAR_offset.

void ADC_SAR_Seq_Start(void)

Description:	This is the preferred method to begin component operation. ADC_SAR_Seq _Start() sets the initVar variable, calls the ADC_SAR_Seq _Init() function, and then calls the ADC_SAR_Seq _Enable() function.
Parameters:	None
Return Value:	None
Side Effects:	If the initVar variable is already set, this function only calls the ADC_SAR_Seq _Enable() function.

void ADC_SAR_Seq_Stop(void)

Description:	Stops ADC_SAR_Seq conversions and reduces the power to the minimum. **Note** This API does not power down the ADC, but reduces power to a minimum level. This device has a defect that causes connections to several analog resources to be unreliable when the device is not powered. The unreliability manifests itself in silent failures (for example, unpredictable bad results from analog components) when the component using that resource is stopped.
Parameters:	None
Return Value:	None
Side Effects:	None

void ADC_SAR_Seq_SetResolution(uint8 resolution)

Description: Sets the resolution for the GetResult16() and GetAdcResult() APIs.

Parameters: uint8 resolution: Resolution setting

Parameters Name	Value	Description
ADC_SAR_Seq_ADC__BITS_12	12	Sets resolution to 12 bits.
ADC_SAR_Seq_ADC__BITS_10	10	Sets resolution to 10 bits.
ADC_SAR_Seq_ADC__BITS_8	8	Sets resolution to 8 bits.

Return Value: None

Side Effects: You cannot change the ADC resolution during a conversion cycle. The recommended best practice is to stop conversions with ADC_SAR_Seq_StopConvert(), change the resolution, then restart the conversions with ADC_SAR_Seq_StartConvert().

If you decide not to stop conversions before calling this API, use ADC_SAR_Seq_IsEndConversion() to wait until conversion is complete before changing the resolution.

If you call ADC_SAR_Seq_SetResolution() during a conversion, the resolution does not change until the current conversion is complete. Data will not be available in the new resolution for another 6 + "New Resolution(in bits)" clock cycles. You may need add a delay of this number of clock cycles after ADC_SAR_Seq_SetResolution() is called before data is valid again.

Affects ADC_SAR_Seq_CountsTo_Volts(),ADC_SAR_Seq_CountsTo_mVolts(), and ADC_SAR_Seq_CountsTo_uVolts() by calculating the correct conversion between ADC_SAR_Seq counts and the applied input voltage. Calculation depends on resolution, input range, and voltage reference.

void ADC_SAR_Seq_StartConvert(void)

Description: This forces the ADC to initiate a conversion. In free-running mode, the ADC_SAR_Seq runs continuously. In software trigger mode, the function also acts as a software version of the SOC and every conversion must be triggered by ADC_SAR_Seq_StartConvert(). In Hardware trigger mode this function is unavailable.

Parameters: None

Return Value: None

Side Effects: Calling ADC_SAR_Seq_StartConvert() disables the external SOC pin.

void ADC_SAR_Seq_StopConvert(void)

Description:	This forces the ADC_SAR_Seq to stop conversions. If a conversion is currently executing, that conversion completes, but no further conversions happen. This only applies to free-running mode.
Parameters:	None
Return Value:	None
Side Effects:	In free-running and software trigger mode, this function sets a software version of the SOC to low level and switches the SOC source to hardware SOC input (Hardware trigger).

void ADC_SAR_Seq_IRQ_Enable(void)

Description:	This enables interrupts to occur at the end of a conversion. Global interrupts must also be enabled for the ADC interrupts to occur. To enable global interrupts, call the enable global interrupt macro "`CyGlobalIntEnable;`" in your *main.c* file before enabling any interrupts.
Parameters:	None
Return Value:	None
Side Effects:	Enables interrupts to occur. Reading the result clears the interrupt.

void ADC_SAR_Seq_IRQ_Disable(void)

Description:	Disables interrupts at the end of a conversion.
Parameters:	None
Return Value:	None
Side Effects:	None

uint8 ADC_SAR_Seq_IsEndConversion(uint8 retMode)

Description: Immediately returns the status of the conversion or does not return (blocking) until the conversion completes, depending on the retMode parameter.

Parameters: uint8 retMode: Check conversion return mode. This table describes the available options.

Options	Description
ADC_SAR_SEQ_RETURN_STATUS	Immediately returns the status. If the value returned is zero, the conversion is incomplete, and this function should be retried until a nonzero result is returned.
ADC_SAR_SEQ_WAIT_FOR_RESULT	Does not return a result until the ADC_SAR_Seq conversion is complete.

Return Value: uint8: If a nonzero value is returned, the last conversion is complete. If the returned value is zero, the ADC_SAR_Seq is still calculating the last result.

Side Effects: This function reads the end of conversion status, which is cleared on read.

int16 ADC_SAR_Seq_GetAdcResult(void)

Description: Gets the data available in the SAR DATA register, not the results buffer.

Parameters: None

Return Value: int8: The LSB of the last ADC conversion.

Side Effects: Converts the ADC counts to the 2’s complement form.

int16 ADC_SAR_Seq_GetResult16(uint16 chan)

Description: Returns the conversion result for channel “chan”.

Parameters: None

Return Value: int16: Returns converted data as a signed 16-bit integer

Side Effects: Converts the ADC counts to the 2’s complement form.

void ADC_SAR_Seq_SetOffset(int32 offset)

Description:	Sets the ADC_SAR_Seq offset, which is used by ADC_SAR_Seq_CountsTo_Volts(),ADC_SAR_Seq _CountsTo_mVolts(), and ADC_SAR_Seq _CountsTo_uVolts(), to subtract the offset from the given reading before calculating the voltage conversion.
Parameters:	int32 offset: This value is measured when the inputs are shorted or connected to the same input voltage.
Return Value:	None
Side Effects:	Affects ADC_SAR_Seq _CountsTo_Volts(),ADC_SAR_Seq _CountsTo_mVolts(), and ADC_SAR_Seq _CountsTo_uVolts() by subtracting the given offset.

void ADC_SAR_Seq_SetGain(int32 adcGain)

Description:	Sets the ADC_SAR_Seq gain in counts per volt for the voltage conversion functions that follow. This value is set by default by the reference and input range settings. It should only be used to further calibrate the ADC_SAR_Seq with a known input or if the ADC_SAR_Seq is using an external reference.
Parameters:	int32 adcGain: ADC_SAR_Seq gain in counts per volt
Return Value:	None
Side Effects:	Affects ADC_CountsTo_Volts(), ADC_CountsTo_mVolts(), ADC_CountsTo_uVolts() by supplying the correct conversion between ADC counts and the applied input voltage.

float32 ADC_SAR_Seq_CountsTo_Volts(int16 adcCounts)

Description:	Converts the ADC_SAR_Seq output to volts as a floating-point number. For example, if the ADC_SAR_Seq measured 0.534 volts, the return value would be 0.534. The calculation of voltage depends on the value of the voltage reference. When the Vref is based on Vdda, the value used for Vdda is set for the project in the System tab of the Design Wide Resources (DWR).
Parameters:	int16 adcCounts: Result from the ADC_SAR_Seq conversion
Return Value:	float32: Result in volts
Side Effects:	None

int32 ADC_SAR_Seq_CountsTo_mVolts(int16 adcCounts)

Description:	Converts the ADC_SAR_Seq output to millivolts as a 16-bit integer. For example, if the ADC_SAR_Seq measured 0.534 volts, the return value would be 534. The calculation of voltage depends on the value of the voltage reference. When the Vref is based on Vdda, the value used for Vdda is set for the project in the System tab of the Design Wide Resources (DWR).
Parameters:	int16 adcCounts: Result from the ADC_SAR_Seq conversion
Return Value:	int32: Result in mV
Side Effects:	None

int32 ADC_SAR_Seq_CountsTo_uVolts(int16 adcCounts)

Description:	Converts the ADC_SAR_Seq output to microvolts as a 32-bit integer. For example, if the ADC_SAR_Seq measured 0.534 volts, the return value would be 534000. The calculation of voltage depends on the value of the voltage reference. When the Vref is based on Vdda, the value used for Vdda is set for the project in the System tab of the Design Wide Resources (DWR).
Parameters:	int16 adcCounts: Result from the ADC conversion
Return Value:	int32: Result in µV
Side Effects:	None

void ADC_SAR_Seq_Sleep(void)

Description:	This is the preferred routine to prepare the component for sleep. The ADC_SAR_Seq _Sleep() routine saves the current component state. Then it calls the ADC_SAR_Seq _Stop() function and calls ADC_SAR_Seq _SaveConfig() to save the hardware configuration. Call the ADC_SAR_Seq _Sleep() function before calling the CyPmSleep() or the CyPmHibernate() function. See the PSoC Creator *System Reference Guide* for more information about power-management functions.
Parameters:	None
Return Value:	None
Side Effects:	None

void ADC_SAR_Seq_Wakeup(void)

Description: This is the preferred routine to restore the component to the state when ADC_SAR_Seq _Sleep() was called. The ADC_SAR_Seq _Wakeup() function calls the ADC_SAR_Seq _RestoreConfig() function to restore the configuration. If the component was enabled before the ADC_SAR_Seq _Sleep() function was called, the ADC_SAR_Seq _Wakeup() function also re-enables the component.

Parameters: None

Return Value: None

Side Effects: Calling the ADC_SAR_Seq _Wakeup() function without first calling the ADC_Sleep() or ADC_SAR_Seq _SaveConfig() function can produce unexpected behavior.

void ADC_SAR_Seq_Init(void)

Description: Initializes or restores the component according to the customizer Configure dialog settings. It is not necessary to call ADC_SAR_Seq _Init() because the ADC_SAR_Seq _Start() routine 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 ADC_SAR_Seq_Enable(void)

Description: Activates the hardware and begins component operation. The higher power is set automatically depending on clock speed. The ADC_SAR_Seq _SetPower() API description contains the relation of the power from the clock rate. It is not necessary to call ADC_SAR_Seq _Enable() because the ADC_SAR_Seq _Start() routine calls this function, which is the preferred method to begin component operation.

Parameters: None

Return Value: None

Side Effects: None

void ADC_SAR_Seq_SaveConfig(void)

Description:	This function saves the component configuration and nonretention registers. It also saves the current component parameter values, as defined in the Configure dialog or as modified by the appropriate APIs. This function is called by the ADC_SAR_Seq _Sleep() function.
Parameters:	None
Return Value:	None
Side Effects:	All ADC_SAR_Seq configuration registers are retained. This function does not have an implementation and is meant for future use. It is shown here so that the APIs are consistent across components.

void ADC_SAR_Seq_RestoreConfig(void)

Description:	This function restores the component configuration and nonretention registers. It also restores the component parameter values to what they were before calling the ADC_SAR_Seq _Sleep() function.
Parameters:	None
Return Value:	None
Side Effects:	Calling this function without first calling the ADC_SAR_Seq _Sleep() or ADC_SAR_Seq _SaveConfig() function can produce unexpected behavior. This function does not have an implementation and is meant for future use. It is provided here so that the APIs are consistent across components.

MISRA Compliance

This section describes the MISRA-C:2004 compliance and deviations for the component. There are two types of deviations:

- project deviations – deviations that are applicable for all PSoC Creator components
- specific deviations – deviations that are applicable only for this component

This section provides information on component-specific deviations. Project deviations are described in the MISRA Compliance section of the *System Reference Guide* along with information on the MISRA compliance verification environment.

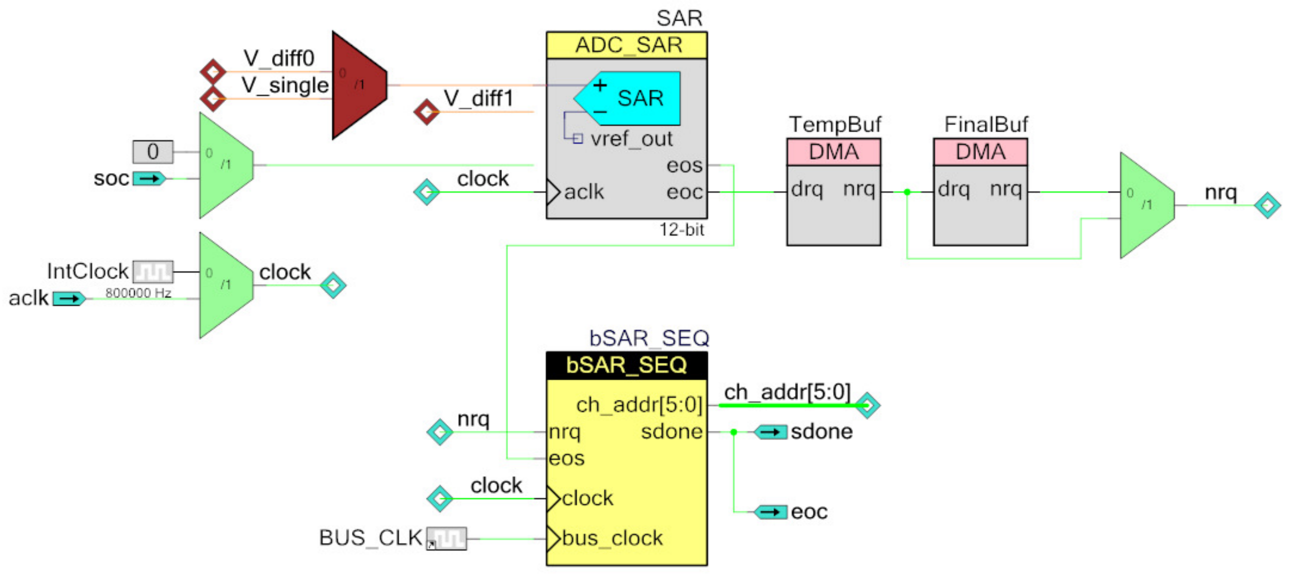
The ADC_SAR_Seq component has these specific deviations:

MISRA-C: 2004 Rule	Rule Class (Required/ Advisory)	Rule Description	Description of Deviation(s)
10.1	R	The value of an expression of integer type shall not be implicitly converted to a different underlying type if: a) it is not a conversion to a wider integer type of the same signedness, or b) the expression is complex, or c) the expression is not constant and is a function argument, or d) the expression is not constant and is a return expression.	The DMA component provides general integer type definitions.
13.2	A	Tests of a value against zero should be made explicit, unless the operand is effectively Boolean.	The DMA component gives general integer type definitions, which are ORed together to provide correct function argument.
17.4	R	Array indexing is the only allowed form of pointer arithmetic.	The DMA structure access uses pointer arithmetic with array indexing.

This component has the following embedded components: ADC_SAR, DMA, interrupt, Hardware AMUX. Refer to the corresponding component datasheets for information on their MISRA compliance and specific deviations.

Functional Description

This block diagram shows how input analog signals from the hardware AMUX are sampled and converted by the ADC SAR component. The first DMA channel moves the results one at a time from the SAR ADC to the temporary RAM buffer. When a complete set of data for all channels is collected in the temporary RAM buffer, the contents of the temporary RAM buffer are transferred to the results (SRAM) buffer in one burst. Using a temporary buffer allows all data to be collected so that the results buffer are updated once per scan, so that you have the necessary time to do one scan sequence to move the last results.



Registers

Status Register

ADC_SAR_SEQ_STATUS_REG

Bits	7	6	5	4	3	2	1	0
Value	Unused							EOC

- EOC – End of Conversion. This register is set when one cycle of conversion for all channels is complete.

Resources

The ADC_SAR_Seq uses SAR ADC component, which in turn is placed on a fixed-block SAR in the silicon.

Configuration	Resource Type					
	Datapath Cells	Macrocells	Status Cells	Control/ Counter7 Cells	DMA Channels	Interrupts
Default	-	45	1	2	2	1

API Memory Usage

The component memory usage varies significantly depending on the compiler, device, number of APIs used and component configuration. This table gives the memory usage for all APIs available in the default component configuration.

The measurements were done with the associated compiler configured in release mode with optimization set for size. For a specific design, analyze the map file generated by the compiler to determine the memory usage.

Configuration	PSoC 5LP (GCC)	
	Flash Bytes	SRAM Bytes
Default	1410	37

DC and AC Electrical Characteristics

These values indicate expected performance and are based on initial characterization data. Unless otherwise specified, operating conditions are:

- Fclk = 1-18 MHz
- **Note** The desired sample rate is guaranteed only if bus clock frequency value is at least twice as large as the component clock frequency.Because of STA violations bus clock should not exceed 48 MHz.
- Input range = $\pm V_{REF}$
- Bypass capacitor of 10 µF

See the appropriate section of the ADC SAR component datasheet for more details.

Component Changes

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

Version	Description of Changes	Reason for Changes / Impact
1.0.a	Updated AC and DC Electrical Characteristics section with the maximum bus clock value	
1.0	First component release	