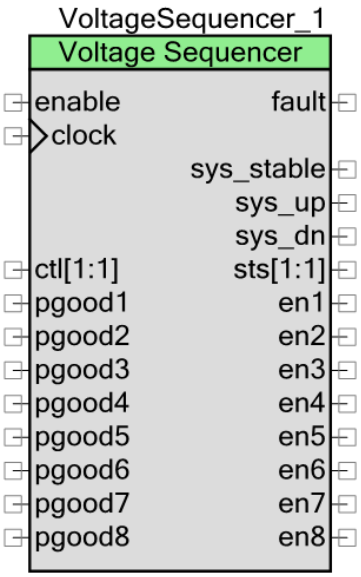
Voltage Sequencer

3.40



Features

- Supports sequencing and monitoring of up to 32 power converter rails
- Supports power converter circuits with logic-level enable inputs and logic-level power good (pgood) status outputs
- Autonomous (standalone) or host driven operation
- Sequence order, timing and inter-rail dependencies can be configured through an intuitive, easy-to-use graphical configuration GUI

General Description

The Voltage Sequencer component provides a simple way to define power-up and power-down sequencing of up to 32 power converters to meet user-defined system requirements. Once the sequencing requirements have been entered into the easy-to-use graphical configuration GUI, the component will automatically take care of the sequencing implementation without requiring any firmware development by the user.

When to Use a Voltage Sequencer

The Voltage Sequencer component should be used in any application that requires sequencing of multiple DC-DC power converters.

For sequencing-only applications, the component can be directly connected to the enable (en) and pgood pins of the DC-DC power converter circuits.

For more comprehensive power supervisor applications, the component can be connected to the Power Monitor or Voltage Fault Detector components in PSoC Creator design schematics. The APIs for these components have also been designed to simplify the firmware interaction between them. The Power Monitor and Voltage Fault Detector components are available in the Power Supervision category of the Cypress component catalog.

Input/Output Connections

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

Enable – Input

Edge triggered global enable input that can optionally be used to initiate a power up sequence or a power down sequence**.** The enable input must *toggle from low to high* to begin a power up sequence on all power converters. A falling edge on this input will force a power-down sequence on all power converters. Connect this terminal to a logic high level if hardware control of the component is not required.

Clock – Input

Timing source used by the component. The component clock frequency is the inverse of the desired fault response time. For example, a 20 MHz clock is required for 50 ns fault response time.

System Stable – Output

Active high output is asserted when all power converters have powered up successfully (all sequencer state machines are in the ON state) and have been running normally for a user-defined amount of time. The output is de-asserted at a sequencer state transition rate when at least one power converter leaves the ON state.

System Up – Output

Active high output is asserted when all power converters have powered up successfully (all sequencer state machines are in the ON state). The output changes at a sequencer state transition rate.

System Down – Output

Active high output is asserted when all power converters have powered down successfully (all sequencer state machines are in the OFF state). The output changes at a sequencer state transition rate.

Warn – Output *

Active high output is asserted when one or more power converters did not shut down within the user-specified time period. Once asserted, remains high until cleared by calling the GetWarnStatus() API function. This terminal is visible when you select the checkbox labeled **Enable TOFF_MAX warnings** on the Power Down tab of the Configure dialog.

Fault – Output

Active high output is asserted when a fault condition has been detected on one or more power converters. Once asserted, remains high until cleared by calling the GetFaultStatus() API function. Avoid connecting this terminal to an interrupt component since this component has a buried interrupt service routine that needs to respond to faults as soon as possible. The intended usage model for this terminal is driving other logic or pins.

Sequencer Control Inputs – Input *

General purpose inputs with user defined polarity that may be used to gate power-up sequencing state changes, to force partial or complete power-down sequencing or both. These terminals are visible when a non-zero value is entered into the **Number of control inputs** parameter on the General tab of the Configure dialog.

Sequencer Status Outputs – Output *

General purpose outputs with user defined polarity that can be asserted and de-asserted depending on the corresponding pgoods[x] inputs to indicate the sequencer's progress. The outputs change at a sequencer state transition rate. These terminals are visible when a non-zero value is entered into the **Number of status outputs** parameter on the General tab of the Configure dialog.

Power Converter Enables – Output

Active high power converter enable outputs. When asserted, these outputs enable the selected power converter so that it will begin regulating power to its output.

Power Converter Power Goods – Input

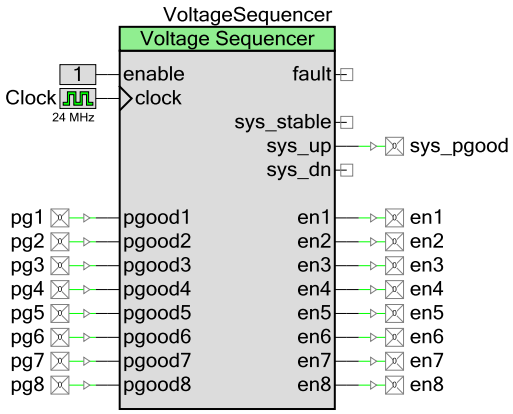
Active high power converter power good status inputs. These inputs may come directly from the power converter status output pins or be derived inside PSoC from ADC monitoring of power converter voltage outputs (using the **Power Monitor** component, for example) or over-voltage/under-voltage window comparator threshold detection (using the **Voltage Fault Detector** component, for example).

The pgood input terminals (pgood[x]) should never be tied high. Doing so will prevent the power converter from being able to re-sequence. If no power good monitoring is desired for one of the power converters, tie the enable terminal (en[x]) directly to the associated pgood[x] terminal.

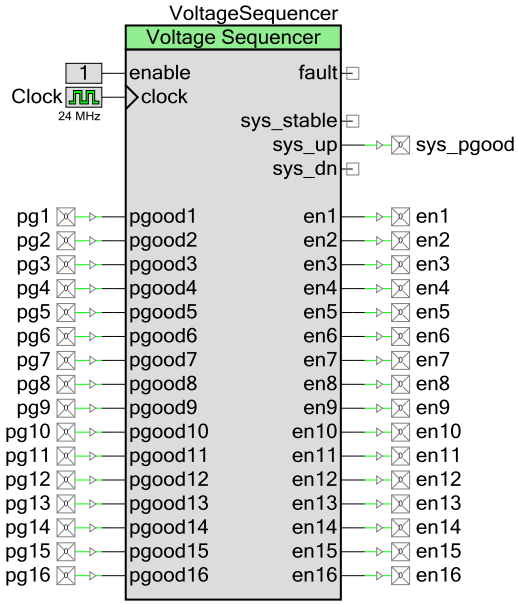
Schematic Macro Information

By default, the PSoC Creator Component Catalog provides three Schematic Macro implementations for the Voltage Sequencer component. These macros contain the Voltage Sequencer component already connected to digital pin components. The Schematic Macros use the Voltage Sequencer component configured for 8, 16, or 32 power converters as shown in the following diagrams.

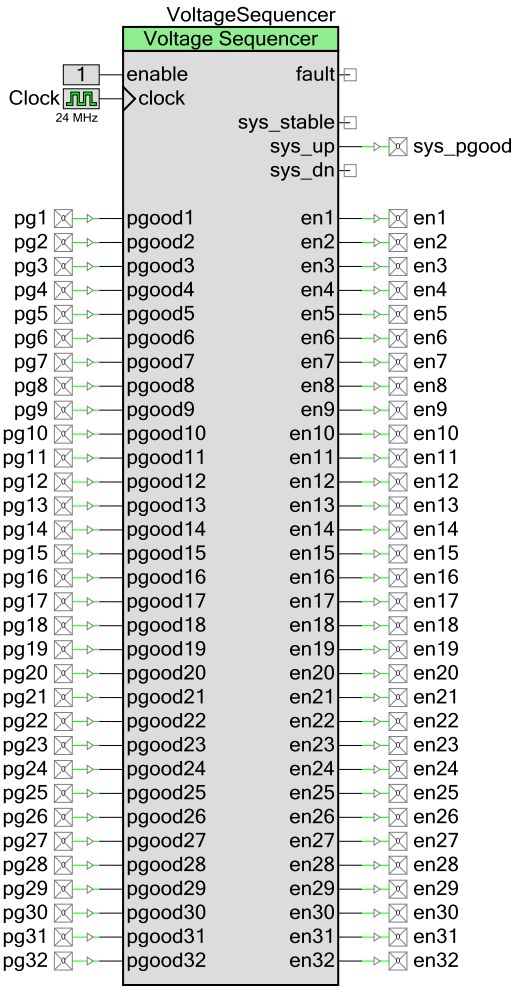
Voltage Sequencer – 8 Rails



Voltage Sequencer – 16 Rails



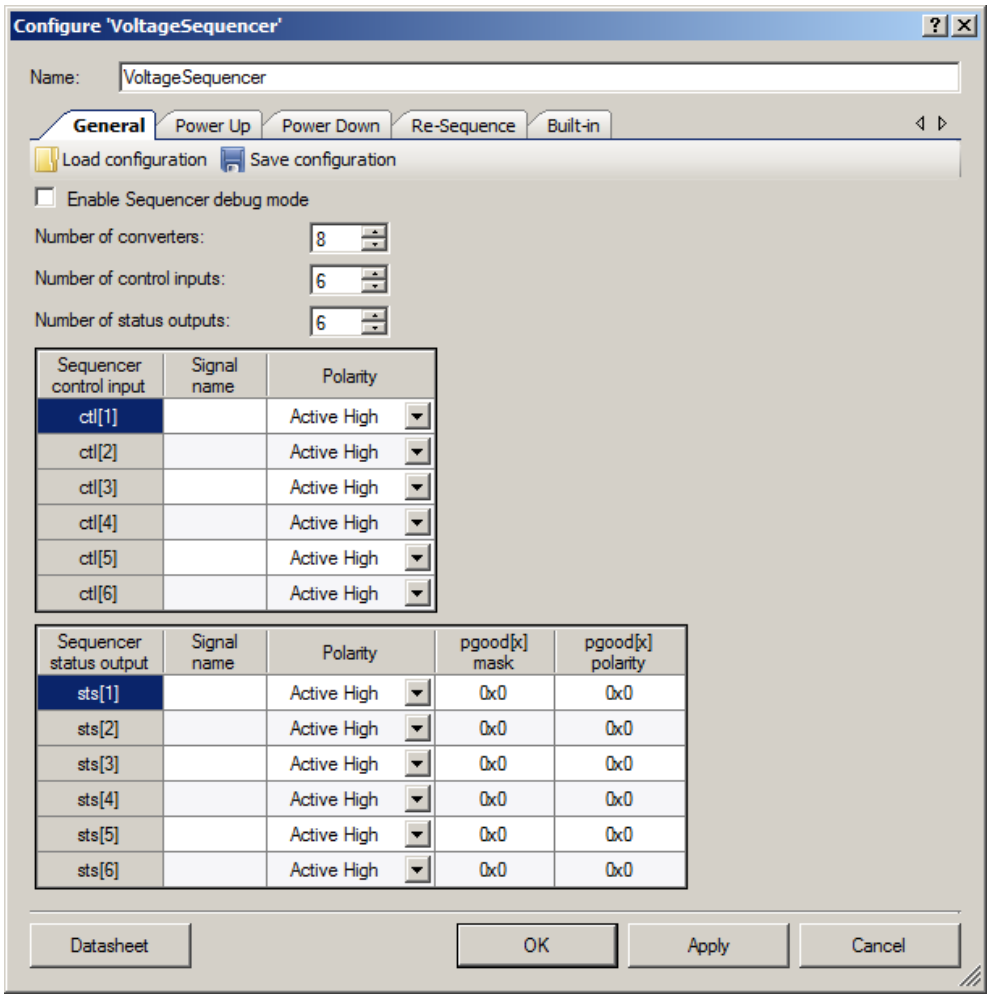
Voltage Sequencer – 32 Rails



Component Parameters

Drag a Voltage Sequencer component onto your design and double click it to open the Configure dialog. This dialog has three tabs to guide you through the process of setting up the Voltage Sequencer component.

General Tab



Load configuration

Restores all customizer settings, including tables, from an external file. Keyboard shortcut – [Ctrl] [L]

Save configuration

Stores all customizer settings, including tables, in an external file. Keyboard shortcut – [Ctrl] [S]

Enable sequencer debug mode

Globally enables or disables sequencer manual debug modes (play/pause/single-stepping) for a board bring-up. The debugging mode enables you to pause the sequencer in its current state preventing any further state transitions, tick timer updates and firmware fault handling. Execution can be resumed later under your control. In addition, a single-step mode enables you to slowly step through complex sequencing events. When single-step mode is invoked, the sequencer will play until there is a state change on any rail, at which time it will automatically pause. You may then elect to single-step again or resume normal sequencing by resuming play mode.

Options = Checked or un-checked. (Default = Un-Checked).

Number of converters

Number of power converters to sequence. Range=1-32. (Default=8).

Number of control inputs

Number of general purpose control inputs. Range=0-6. (Default=1).

Number of status outputs

Number of general purpose status outputs. Range=0-6. (Default=1).

ctl[x] Signal name

Text field, 16 characters, for annotation purposes only. Use it to enter a descriptive name of the control input. By default this field is empty and no value is required. It will be visible only when the **Number of control inputs** parameter is non-zero.

ctl[x] Polarity

Options=Active High or Active Low. It will be visible only when the **Number of control inputs** parameter is non-zero. (Default = Active High).

sts[x] Signal name

Text field, 16 characters, for annotation purposes only. Use it to enter a descriptive name of the status output. By default this field is empty and no value is required. It will be visible only when the **Number of status outputs** parameter is non-zero.

sts[x] Polarity

Options=Active High or Active Low. It will be visible only when the **Number of status outputs** parameter is non-zero. (Default = Active High).

pgood[x] mask

Hexadecimal encoding of which pgood[x] inputs participate in the logic equation for the sts[x] output where bit 0 corresponds to pgood[1] and bit 31 corresponds to pgood[32]. The encoding value will display 2, 4, 6 or 8 hex digits depending on the **Number of converters** parameter. The encoding for each bit is as follows:

1=pgood[x] participates

0=pgood[x] does not participate

The hexadecimal encoding can be entered manually, or the helper form can be used to select the participating pgood[x] inputs from the array, automatically generating the hexadecimal encoding for you. An example is shown below.

It will be visible only when the **Number of status outputs** parameter is non-zero.

(Default = 0)

pgood[x] polarity

Hexadecimal encoding of the polarity of the pgood[x] input that will be used in the logic equation for the sts[x] output where bit 0 corresponds to pgood[1] and bit 31 corresponds to pgood[32]. The encoding value will display 2, 4, 6 or 8 hex digits depending on the **Number of converters** parameter. The encoding for each bit is as follows:

1=use the true pgood[x] in the logic equation

0=use the inverted pgood[x] in the logic equation

The associated sts[x] is the logical AND of the pgood[x] inputs of the selected power converters. The hexadecimal encoding can be entered manually, or the helper form can be used to select the participating pgood[x] inputs from the array, automatically generating the hexadecimal encoding for you. An example is shown below.

It will be visible only when the **Number of status outputs** parameter is non-zero.

(Default = 0)

pgood[x] mask and pgood[x] polarity example

A system that supports sequencing on 8 rails with these parameters configured for the **sts[1]** output:

Number of converters = 8,

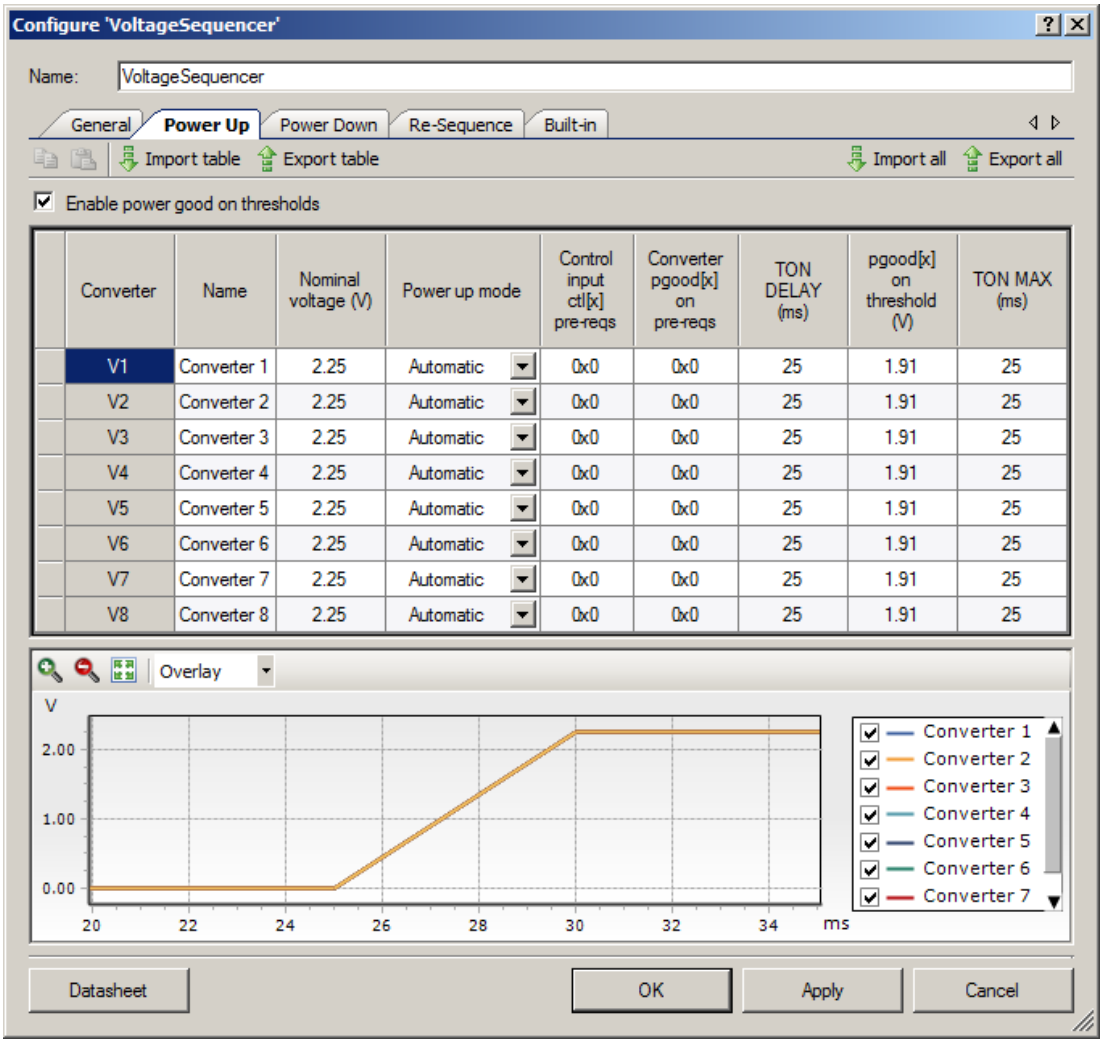
pgood[x] mask = 0xC5 (hexadecimal),

pgood[x] polarity = 0x81 (hexadecimal)

The logic equation for the sts[1] output becomes:

sts[1] = pgood[8] & !pgood[7] & !pgood[3] & pgood[1]

Power Up Tab



Import table

Imports data from file to table cells on active tab. Supports .csv file format. Keyboard shortcut – [Ctrl] [M]

Export table

Exports data from table cells active tab to file. Supports .csv file format. Keyboard shortcut – [Ctrl] [R].

Import all

Executes import functionality for all three tables. Keyboard shortcut – [Ctrl] [Alt] [M]

Export all

Executes export functionality for all three tables. Keyboard shortcut – [Ctrl] [Alt] [R].

Enable power good on thresholds

Globally enable or disable power good on threshold checking.

Options = Checked or un-checked. Enabling this option exposes the **pgood[x] on threshold** column (Default = Un-Checked). Note that this component has no capability to monitor the analog output voltages of the power converters to determine whether or not they are good. Checking this parameter simply provides hooks in the firmware APIs that enable the Voltage Sequencer component to interact with other components that do support analog voltage monitoring such as the **Power Monitor** component or the **Voltage Fault Detector** component.

Name

Text field, 16 characters, for annotation purposes only. Use it to enter a descriptive name of the power converter. By default this field is labeled **Converter x** (where x=1..**Number of converters**) and no value is required.

Nominal voltage (V)

Nominal converter output voltage. Annotation purposes only. Range=0.01–65.54.

Power up mode

Power up mode pull-down box. Options=Automatic or Manual. When “Manual” is chosen, the associated power converter will not begin power-up sequencing until instructed to do so either by calling the ForceOn()/ForceAllOn() firmware APIs or by toggling the enable input terminal from low to high. When “Automatic” mode is selected, neither of these conditions must be met in order to begin power-up sequencing. (Default = Automatic).

Control input ctl[x] pre-reqs

Hexadecimal encoding of which ctl[x] inputs are pre-requisite inputs required for the associated power converter to power-up. This control enables you to wait for one or more hardware signals to be asserted before powering-up the specified power converter. The encoding for each bit is as follows:

1=ctl[x] is a pre-requisite for power-up sequencing

0=ctl[x] is not a pre-requisite for power-up sequencing

The hexadecimal encoding can be entered manually, or the helper form can be used to select the participating ctl[x] inputs from the array, automatically generating the hexadecimal encoding for you. It will be visible only when the **Number of control inputs** parameter on the General tab is non-zero.

(Default = 0)

Converter pgood[x] on pre-reqs

Hexadecimal encoding of which pgood[x] inputs are pre-requisite inputs required for the associated power converter to power-up. This control enables you to wait for one or more rails to become good before powering-up the specified power converter. The encoding for each bit is as follows:

1=pgood[x] is a pre-requisite for power-up sequencing

0= pgood[x] is not a pre-requisite for power-up sequencing

The hexadecimal encoding can be entered manually, or the helper form can be used to select the participating pgood[x] inputs from the array, automatically generating the hexadecimal encoding for you.

(Default = 0)

TON delay (ms)

Turn on delay. The time between all sequencing pre-requisites being met and enabling the power converter. Units are ms. (Default = 25).

Number of converters	Step size (ms)	Range (s)
≤ 16 converters	0.25	0-16.384
> 16 converters	0.50	0-32.768

pgood[x] on threshold

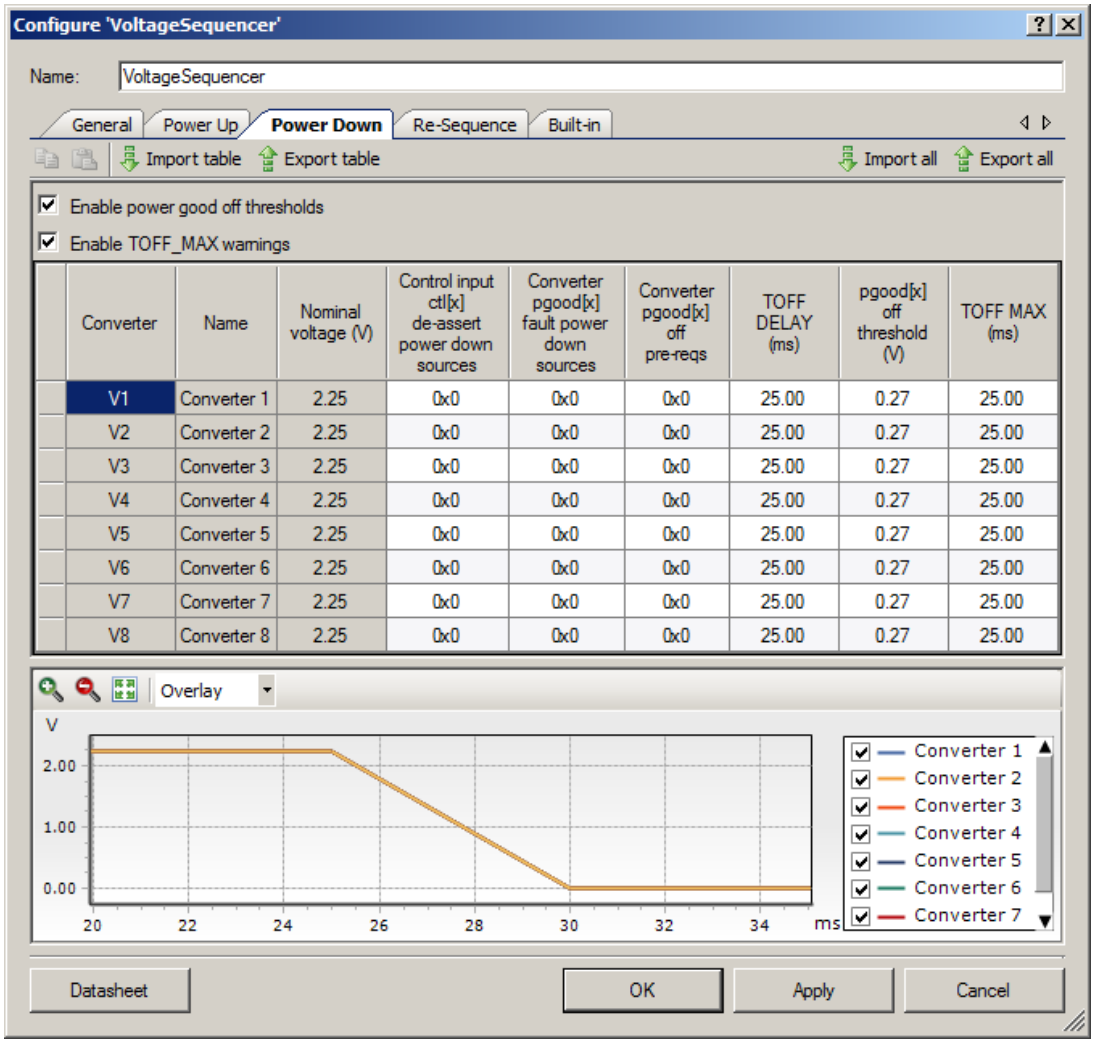
Minimum converter output voltage required to be considered good during power up sequencing. Range=0.01–65.54. Must be less than or equal to the **Nominal voltage** parameter for that converter. This column is only displayed when **Enable power good on thresholds** is checked. Note that this component has no capability to monitor the analog output voltages of the power converters to determine whether or not they are good. Checking this parameter simply provides hooks in the firmware APIs that enable the Voltage Sequencer component to interact with other components that do support analog voltage monitoring such as the **Power Monitor** component or the **Voltage Fault Detector** component.

TON_MAX (ms)

Maximum turn on delay. This is a power-up timeout parameter that specifies the maximum time allowable between enabling the power converter and the associated pgood[x] input being asserted. Units are ms. (Default = 25).

Number of converters	Step size (ms)	Range (s)
≤ 16 converters	0.25	0-16.384
> 16 converters	0.50	0-32.768

Power Down Tab



Enable power good off thresholds

Globally enable or disable power good off threshold checking. Note that this component has no capability to monitor the analog output voltages of the power converters to determine whether or not they are good. Checking this parameter simply provides hooks in the firmware APIs that enable the Voltage Sequencer component to interact with other components that do support analog voltage monitoring such as the **Power Monitor** component or the **Voltage Fault Detector** component.

Options = Checked or un-checked. Enabling this option exposes the pgood[x] off threshold column (Default = Un-Checked).

Enable TOFF_MAX warnings

Globally enable or disable warnings caused by TOFF_MAX_WARN_LIMIT timeouts

Options = Checked or un-checked. Enabling this option exposes the **warn** terminal on the symbol (Default = Un-Checked).

Name

Text field, 16 characters, for annotation purposes only. This is a display (not editable) brought forward from the Power Up tab.

Nominal voltage (V)

Nominal converter output voltage for annotation purposes only. This is a display (not editable) brought forward from the Power Up tab.

Control input ctl[x] de-assert power down sources

Hexadecimal encoding of which ctl[x] inputs will force the associated power converter to power-down when they are de-asserted. This control enables you to power-down the specified power converter by de-asserting one or more hardware signals. The encoding for each bit is as follows:

1=ctl[x] will force a power-down sequence when de-asserted

0= ctl[x] will not force a power-down sequence when de-asserted

The hexadecimal encoding can be entered manually, or the helper form can be used to select the participating ctl[x] inputs from the array, automatically generating the hexadecimal encoding for you. It will be visible only when the **Number of control inputs** parameter on the General tab is non-zero.

(Default = 0)

Converter pgood[x] fault power down sources

Hexadecimal encoding of which pgood[x] inputs will force the associated power converter to power-down when they are de-asserted. This control enables you to power-down a rail in response to a fault on one or more other rails. Associating rails in this manner is referred to as a “fault group”. The encoding for each bit is as follows:

1=pgood[x] will force a power-down sequence when de-asserted

0= pgood[x] will not force a power-down sequence when de-asserted

The hexadecimal encoding can be entered manually, or the helper form can be used to select the participating pgood[x] inputs from the array, automatically generating the hexadecimal encoding for you.

(Default = 0)

Converter pgood[x] off pre-reqs

Hexadecimal encoding of which pgood[x] inputs are pre-requisite inputs required for the associated power converter to power-down. This control enables you to wait for one or more rails to be turned off before powering-down the specified power converter. The encoding for each bit is as follows:

1=pgood[x] is a pre-requisite for power-down sequencing

0= pgood[x] is not a pre-requisite for power-down sequencing

The hexadecimal encoding can be entered manually, or the helper form can be used to select the participating pgood[x] inputs from the array, automatically generating the hexadecimal encoding for you.

(Default = 0)

TOFF delay (ms)

Turn off delay. The time between initiating a power-down of the associated power converter and actually de-asserting the **en** output to the power converter. Units are ms. Set to 0 for immediate shutdown. (Default = 25).

Number of converters	Step size (ms)	Range (s)
≤ 16 converters	0.25	0-16.384
> 16 converters	0.50	0-32.768

pgood[x] off threshold (V)

The voltage level that the power converter output must drop to in order to be considered powered-off. Range=0.00–65.54. Must be <= **Nominal Voltage.** This column is only displayed when **Enable power good off thresholds** is checked. Note that this component has no capability to monitor the analog output voltages of the power converters to determine whether or not they are good. Checking this parameter simply provides hooks in the firmware APIs that enable the Voltage Sequencer component to interact with other components that do support analog voltage monitoring such as the **Power Monitor** component or the **Voltage Fault Detector** component.

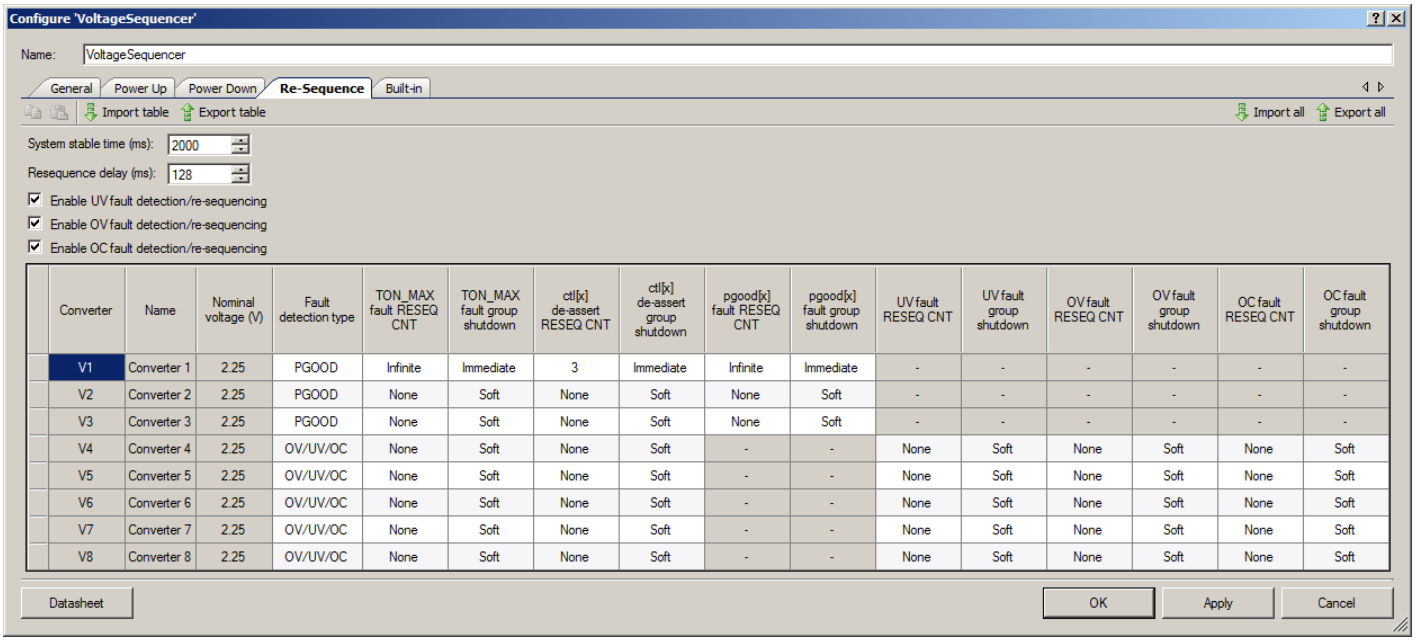
TOFF_MAX (ms)

Maximum turn off max delay. This is a power-down timeout parameter that specifies the maximum time allowable between disabling the power converter and the associated pgood[x] input being de-asserted. Units are ms. (Default = 25).

Number of converters	Step size (ms)	Range (s)
≤ 16 converters	0.25	0-16.384
> 16 converters	0.50	0-32.768

This column is only displayed when **Enable TOFF_MAX warnings** is checked.

Re-Sequence Tab



Converter	Name	Nominal voltage (V)	Fault detection type	TON_MAX fault RESEQ CNT	TON_MAX fault group shutdown	ctl[x] de-assert RESEQ CNT	ctl[x] de-assert group shutdown	pgood[x] fault RESEQ CNT	pgood[x] fault group shutdown	UV fault RESEQ CNT	UV fault group shutdown	OV fault RESEQ CNT	OV fault group shutdown	OC fault RESEQ CNT	OC fault group shutdown
V1	Converter 1	2.25	PGOOD	Infinite	Immediate	3	Immediate	Infinite	Immediate	-	-	-	-	-	-
V2	Converter 2	2.25	PGOOD	None	Soft	None	Soft	None	Soft	-	-	-	-	-	-
V3	Converter 3	2.25	PGOOD	None	Soft	None	Soft	None	Soft	-	-	-	-	-	-
V4	Converter 4	2.25	OV/UV/OC	None	Soft	None	Soft	-	-	None	Soft	None	Soft	None	Soft
V5	Converter 5	2.25	OV/UV/OC	None	Soft	None	Soft	-	-	None	Soft	None	Soft	None	Soft
V6	Converter 6	2.25	OV/UV/OC	None	Soft	None	Soft	-	-	None	Soft	None	Soft	None	Soft
V7	Converter 7	2.25	OV/UV/OC	None	Soft	None	Soft	-	-	None	Soft	None	Soft	None	Soft
V8	Converter 8	2.25	OV/UV/OC	None	Soft	None	Soft	-	-	None	Soft	None	Soft	None	Soft

System stable time (ms)

Number of ms that all power converters must remain in the ON state before the system is considered “stable”. When the system is stable, the sys_stable output terminal is asserted high. This parameter plays a key role when automatic re-sequencing is enabled. When the system is stable, the re-sequence counters are re-loaded with the user defined count values. On the other hand, if a fault occurs when the system is not stable, the re-sequence counters are decremented, indicating that a re-sequence attempt is in progress. 16-bit value, 8 ms resolution, 0-524 sec range. (Default = 2000).

Resequence delay (ms)

Global re-sequence delay for all power converter state machines. Controls the amount of time to wait between automatic re-sequence attempts. Units are steps of 8 ms. Range=0-65535 (0-534.28 s). (Default = 128).

Enable UV fault detection/re-sequencing

Checking this option gives you the ability to enter automatic re-sequencing parameters unique to under voltage fault conditions. Note that this component has no capability to monitor the analog output voltages of the power converters to determine whether or not they are good. Checking this parameter simply provides hooks in the firmware APIs that enable the Voltage Sequencer component to interact with other components that do support analog voltage monitoring such as the **Power Monitor** component or the **Voltage Fault Detector** component.

Options = Checked or unchecked. (Default = Unchecked).

Enable OV fault detection/re-sequencing

Checking this option gives you the ability to enter automatic re-sequencing parameters unique to over voltage fault conditions. Note that this component has no capability to monitor the analog output voltages of the power converters to determine whether or not an over-voltage fault has occurred. Checking this parameter simply provides hooks in the firmware APIs that enable the Voltage Sequencer component to interact with other components that do support analog voltage monitoring such as the **Power Monitor** component or the **Voltage Fault Detector** component.

Options = Checked or unchecked. (Default = Unchecked).

Enable OC fault detection/re-sequencing

Checking this option gives you the ability to enter automatic re-sequencing parameters unique to over current fault conditions. Note that this component has no capability to monitor the analog load currents of the power converters to determine whether or not an over-current fault has occurred. Checking this parameter simply provides hooks in the firmware APIs that enable the Voltage Sequencer component to interact with other components that do support analog current monitoring such as the **Power Monitor** component.

Options = Checked or unchecked. (Default = Unchecked).

Name

Text field, 16 characters, for annotation purposes only. This is a display (not editable) brought forward from the Power Up tab.

Nominal voltage (V)

Nominal converter output voltage for annotation purposes only. This is a display (not editable) brought forward from the **Power Up** tab.

Fault detection type

This parameter is used to specify which type of fault detection and re-sequencing is enabled for this power converter. The power good (pgood[x]) inputs can be connected directly to the pgood status output pin on the power converters or to the pgood signals generated internally by PSoC. The signal is generated via the Voltage Fault Detector component or the Power Monitor component, since each component has the capability to actively monitor the power converter analog output voltages and load currents.

Options = **PGOOD** or **OV/UV/OC** (Default = **OV/UV/OC**). Select **OV/UV/OC** if the analog voltage or current monitoring is supported for this power converter. Select **PGOOD** if the pgood[x] input is connected directly to the external pgood status output pin on the power converter.

This parameter is available when at least one of the **Enable…** check boxes is selected.

TON MAX fault RESEQ CNT

TON_MAX fault re-sequence count for this power converter. This parameter specifies how many times to attempt automatic re-sequencing of this rail (and all other rails in associated fault groups) when it experiences a power-up timeout condition. Options=None, 1-30, Infinite. (Default = Infinite)

TON MAX fault group shutdown

TON_MAX fault group shutdown response pull-down box. If this rail experiences a power-up timeout condition, it will be turned off immediately. However, this parameter specifies the power down timing of all rails in associated fault groups. The list of other rails that constitute the fault groups is controlled through the bitmasks entered into the **Converter pgood[x] fault power down sources** parameters on the Power Down tab.

Options=Soft or Immediate. When “Soft” is chosen, the power down delay time for all rails in associated fault groups is determined by the **TOFF delay** parameter set for those rails in the Power Down tab. (Default = Immediate).

ctl[x] de-assert RESEQ CNT

Ctl[x] fault re-sequence count for this power converter. This parameter specifies how many times to attempt automatic re-sequencing of this rail (and all other rails in associated fault groups) in response to de-asserting one or more ctl[x] inputs. The list of ctl[x] inputs that can cause a shutdown and re-sequence attempt on this rail is controlled through the bitmask entered into the **Control input ctl[x] de-assert** parameter for this rail on the Power Down tab. Options=None, 1-30, Infinite. (Default = Infinite).

This parameter is only available when the **Number of control inputs** parameter on the General tab is non-zero.

ctl[x] de-assert group shutdown

Ctl[x] fault group shutdown response pull-down box. This parameter specifies the power down timing of this rail (and all other rails in associated fault groups) in response to de-asserting one or more ctl[x] inputs. The list of ctl[x] inputs that can cause a shutdown and re-sequence attempt on this rail is controlled through the bitmask entered into the **Control input ctl[x] de-assert** parameter for this rail on the Power Down tab.

Options=Soft or Immediate. When “Soft” is chosen, the power down delay time for this rail is determined by its own **TOFF delay** parameter set in the Power Down tab. (Default = Immediate).

This parameter is only available when the **Number of control inputs** parameter on the General tab is non-zero.

pgood[x] de-assert RESEQ CNT

pgood[x] fault re-sequence count for this power converter. This parameter specifies how many times to attempt automatic re-sequencing of this rail (and all other rails in associated fault

groups) when it experiences a fault condition causing de-assertion of its own pgood[x] input. Options=None, 1-30, Infinite. (Default = Infinite)

This parameter is available when all of the **Enable…** check boxes are not selected or when the **Fault detection type** parameter is set to **PGOOD** for this power converter.

pgood[x] de-assert group shutdown

pgood[x] fault group shutdown response pull-down box. If this rail experiences a fault causing its own pgood[x] signal to be de-asserted, it will be turned off immediately. However, this parameter specifies the power down timing of all rails in associated fault groups. The list of other rails that constitute the fault groups is controlled through the bitmasks entered into the **Converter pgood[x] fault power down sources** parameters on the Power Down tab.

Options=Soft or Immediate. When “Soft” is chosen, the power down delay time for all rails in associated fault groups is determined by the **TOFF delay** parameter set for those rails in the Power Down tab. (Default = Immediate).

This parameter is available when all of the **Enable…** check boxes are not selected or when the **Fault detection type** parameter is set to **PGOOD** for this power converter.

UV fault RESEQ CNT

Under voltage fault re-sequence count for this power converter. This parameter specifies how many times to attempt automatic re-sequencing of this rail (and all other rails in associated fault groups) when it experiences an under voltage fault condition causing de-assertion of its own pgood[x] input. Options=None, 1-30, Infinite. (Default = Infinite)

This parameter is only available when the **Enable UV fault detection/re-sequencing** check box is selected and the **Fault detection type** parameter is set to **OV/UV/OC** for this power converter. Note that this component has no capability to monitor the analog output voltages of the power converters to determine whether or not they are good. Checking the **Enable UV fault re-sequencing** checkbox simply provides hooks in the firmware APIs that enable the Voltage Sequencer component to interact with other components that do support analog voltage monitoring such as the **Power Monitor** component or the **Voltage Fault Detector** component.

UV fault group shutdown

Under voltage fault group shutdown response pull-down box. If this rail experiences an under voltage fault causing its own pgood[x] signal to be de-asserted, it will be turned off immediately. However, this parameter specifies the power down timing of all rails in associated fault groups. The list of other rails that constitute the fault groups is controlled through the bitmasks entered into the **Converter pgood[x] fault power down sources** parameters on the Power Down tab.

Options=Soft or Immediate. When “Soft” is chosen, the power down delay time for all rails in associated fault groups is determined by the **TOFF delay** parameter set for those rails in the Power Down tab. (Default = Immediate).

This parameter is only available when the **Enable UV fault re-sequencing** check box is selected and the **Fault detection type** parameter is set to **OV/UV/OC** for this power converter.

OV fault RESEQ CNT

Over voltage fault re-sequence count for this power converter. This parameter specifies how many times to attempt automatic re-sequencing of this rail (and all other rails in associated fault groups) when it experiences an over voltage fault condition causing de-assertion of its own pgood[x] input. Options=None, 1-30, Infinite. (Default = Infinite)

This parameter is only available when the **Enable OV fault re-sequencing** check box is selected and the **Fault detection type** parameter is set to **OV/UV/OC** for this power converter. Note that this component has no capability to monitor the analog output voltages of the power converters to determine whether or not they are good. Checking the **Enable OV fault re-sequencing** checkbox simply provides hooks in the firmware APIs that enable the Voltage Sequencer component to interact with other components that do support analog voltage monitoring such as the **Power Monitor** component or the **Voltage Fault Detector** component.

OV fault group shutdown

Over voltage fault group shutdown response pull-down box. If this rail experiences an over voltage fault causing its own pgood[x] signal to be de-asserted, it will be turned off immediately. However, this parameter specifies the power down timing of all rails in associated fault groups. The list of other rails that constitute the fault groups is controlled through the bitmasks entered into the **Converter pgood[x] fault power down sources** parameters on the Power Down tab.

Options=Soft or Immediate. When “Soft” is chosen, the power down delay time for all rails in associated fault groups is determined by the **TOFF delay** parameter set for those rails in the Power Down tab. (Default = Immediate).

This parameter is only available when the **Enable OV fault re-sequencing** check box is selected and **Fault detection type** parameter is set to **OV/UV/OC** for this power converter.

OC fault RESEQ CNT

Over current fault re-sequence count for this power converter. This parameter specifies how many times to attempt automatic re-sequencing of this rail (and all other rails in associated fault groups) when it experiences an over current fault condition causing de-assertion of its own pgood[x] input. Options=None, 1-30, Infinite. (Default = Infinite)

This parameter is only available when the **Enable OC fault re-sequencing** check box is selected and the **Fault detection type** parameter is set to **OV/UV/OC** for this power converter. Note that this component has no capability to monitor the analog output voltages of the power converters to determine whether or not they are good. Checking the **Enable OC fault re-sequencing** checkbox simply provides hooks in the firmware APIs that enable the Voltage Sequencer component to interact with other components that do support analog load current monitoring such as the **Power Monitor** component.

OC fault group shutdown

Over current fault group shutdown response pull-down box. If this rail experiences an over current fault causing its own pgood[x] signal to be de-asserted, it will be turned off immediately. However, this parameter specifies the power down timing of all rails in associated fault groups. The list of other rails that constitute the fault groups is controlled through the bitmasks entered into the **Converter pgood[x] fault power down sources** parameters on the Power Down tab.

Options=Soft or Immediate. When “Soft” is chosen, the power down delay time for all rails in associated fault groups is determined by the **TOFF delay** parameter set for those rails in the Power Down tab. (Default = Immediate).

This parameter is only available when the **Enable OC fault re-sequencing** check box is selected and the **Fault detection type** parameter is set to **OV/UV/OC** for this power converter.

Application Programming Interface

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

By default, PSoC Creator assigns the instance name “ VoltageSequencer_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 “Sequencer”.

Control and Status Functions

Function	Description
Sequencer_Start()	Enables the component and places all power converter state machines into the appropriate state
Sequencer_Stop()	Disables the component
Sequencer_Init()	Initializes the component
Sequencer_Enable()	Enables the component
Sequencer_Pause()	Pauses the sequencer, preventing sequencer state machine state transitions
Sequencer_Play()	Resumes the sequencer if previously paused
Sequencer_SingleStep()	Puts the sequencer in single step mode
Sequencer_EnableCalibrationState()	Enables the sequencer calibration state, preventing sequencer state machine state transitions, and disables fault detection and processing
Sequencer_DisableCalibrationState()	Resumes the sequencer state machine state transitions and enables fault detection and processing

Function	Description
Sequencer_ForceOn()	Forces the selected power converter to power up
Sequencer_ForceAllOn()	Forces all power converters to power up
Sequencer_ForceOff()	Forces the selected power converter to power down either immediately or after the TOFF delay
Sequencer_ForceAllOff()	Forces all power converters to power down either immediately or after their TOFF delays
Sequencer_GetState()	Returns the current state machine state for the selected power converter
Sequencer_GetPgoodStatus()	Returns a bitmask that represents pgood[x] status for all power converters
Sequencer_GetFaultStatus()	Returns a bitmask that represents which power converters have experienced a fault that caused de-assertion of their pgood[x] inputs
Sequencer_GetCtlStatus()	Returns a bitmask that represents which ctl[x] inputs have caused one or more converters to shutdown
Sequencer_GetWarnStatus()	Returns a bitmask that represents which power converters have experienced a power down warning caused by exceeding the TOFF_MAX_WARN timeout
Sequencer_EnFaults()	Enables/disables assertion of the fault output terminal
Sequencer_EnWarnings()	Enables/disables assertion of the warn output terminal

void Sequencer_Start(void)

Description: Enables the component and places all power converter state machines into the appropriate state (OFF or PEND_ON). Calls the Init() API if the component has not been initialized before. Calls the Enable() API.

void Sequencer_Stop(void)

Description: Disables the component, preventing sequencer state machine state transitions, system timer updates and fault handling.

Side Effects: All output terminals are de-asserted.

void Sequencer_Init(void)

Description: Initializes the component. Parameter settings are initialized based on parameters entered into the various Configure dialog tabs.

void Sequencer_Enable(void)

Description: Enables the component. Enables sequencer state machine state transitions, system timer updates and fault handling.

void Sequencer_Pause(void)

Description: Pauses the sequencer, preventing sequencer state machine state transitions, system timer updates and fault handling.

Side Effects: All 3 sequencer interrupts are disabled.

void Sequencer_Play(void)

Description: Resumes the sequencer if previously paused. Re-enables sequencer state machine state transitions, system timer updates and fault handling.

Side Effects: All 3 sequencer interrupts are enabled.

void Sequencer_SingleStep(void)

Description: Puts the sequencer in single step mode. If the sequencer was paused, it will resume normal operation. The sequencer will then run until there is a state transition on any rail. At that time, the sequencer will be paused automatically until either the Play() API or the SingleStep() is called again.

Side Effects: All 3 sequencer interrupts are enabled until a state transition occurs on any rail. At that time, all 3 sequencer interrupts will be disabled again.

void Sequencer_EnableCalibrationState(void)

Description: Enables the sequencer calibration state, preventing sequencer state machine state transitions, system timer updates and fault handling. Stops the hardware fast shutdown block.

Side Effects: All three sequencer interrupts are disabled. Power converter enable outputs will hold their states while the sequencer is in calibration state. Faults will not be detected and faulty rails will not be shut down.

void Sequencer_DisableCalibrationState(void)

Description: Disables the sequencer calibration state. Re-enables sequencer state machine state transitions, system timer updates and fault handling. Enables the hardware fast shutdown block.

Side Effects: All three sequencer interrupts are enabled.

void Sequencer_ForceOn(uint8 converterNum)

Description: Forces the selected power converter to the PEND_ON state. All selected power up pre-requisite conditions must be satisfied for the power converter to turn on. The re-sequence counter for that converter's state machine is re-initialized.

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

void Sequencer_ForceAllOn(void)

Description: Forces all power converters to the PEND_ON state. All selected power up pre-requisite conditions must be satisfied for the power converter to turn on. The re-sequence counter for that converter’s state machines is re-initialized.

void Sequencer_ForceOff(uint8 converterNum, uint8 powerOffMode)

Description: Forces the selected power converter to power down either immediately or after the TOFF delay. All selected power down pre-requisite conditions must be satisfied for the power converter to turn off.

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

uint8 powerOffMode: Specifies the shutdown mode
Options: 0=immediate, 1=soft

void Sequencer_ForceAllOff(uint8 powerOffMode)

Description: Forces all power converters to power down either immediately or after their TOFF delays. All selected power down pre-requisite conditions must be satisfied for the power converter to turn off.

Parameters: uint8 powerOffMode: Specifies the shutdown mode
Options: 0=immediate, 1=soft

uint8 Sequencer_GetState(uint8 converterNum)

Description: Returns the current state machine state for the selected power converter.

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint8 state: Power converter state machine state

Encoding	State
0	OFF
1	PEND_ON
2	TON_DELAY
3	TON_MAX
4	ON
5	PEND_OFF
6	TOFF_DELAY
7	TOFF_MAX
8	PEND_RESEQ
9	TRESEQ_DELAY
10..255	Undefined

uint8/uint16/uint32 Sequencer_GetPgoodStatus(void)

Description: Returns a bitmask that represents pgood[x] status for all power converters.

Return Value: uint8/uint16/uint32 pgoodStatus. Depends on the number of converters
Power good status of power converters

Bit Field	Power Good Status
0	power converter 1 pgood status
1	power converter 2 pgood status
…	…
31	power converter 32 pgood status

uint8/uint16/uint32 Sequencer_GetFaultStatus(void)

Description: Returns a bitmask that represents which power converters have experienced a fault that caused de-assertion of their pgood[x] inputs. Bits are sticky until cleared by calling this API.

Return Value: uint8/uint16/uint32 faultStatus. Depends on the number of converters
Fault status of power converters

Bit Field	Fault Status
0	1=power converter 1 has/had a pgood fault
1	1=power converter 2 has/had a pgood fault
…	…
31	1=power converter 32 has/had a pgood fault

Side Effects: Calling this API de-asserts the fault output terminal

uint8 Sequencer_GetCtlStatus(void)

Description: Returns a bitmask that represents which ctl[x] inputs have caused one or more converters to shut down. Bits are sticky until cleared by calling this API.

Return Value: uint8 ctlStatus
Specifies which ctl[x] inputs have caused a shutdown

Bit Field	Control Pin Shutdown Mask
0	1=ctl[1] de-assertion caused a shutdown
1	1=ctl[2] de-assertion caused a shutdown
…	…
5	1=ctl[6] de-assertion caused a shutdown
7..6	Reserved. Set to zeroes

uint8/uint16/uint32 Sequencer_GetWarnStatus(void)

Description: Returns a bitmask that represents which power converters have experienced a power down warning caused by exceeding the TOFF_MAX_WARN timeout. Bits are sticky until cleared by calling this API.

Return Value: uint8/uint16/uint32 warnStatus. Depends on the number of converters
Warning status of power converters

Bit Field	Warning Status
0	1=power converter 1 has/had a warning
1	1=power converter 2 has/had a warning
…	…
31	1=power converter 32 has/had a warning

Side Effects: Calling this API de-asserts the warn output terminal

void Sequencer_EnFaults(uint8 faultEnable)

Description: Enables/disables assertion of the fault output terminal. Faults are still processed by the state machine and fault status is still available through the GetFaultStatus() API.

Parameters: uint8 faultEnable
Options: 0=disabled, 1=enabled
Enabled when the component is started

void Sequencer_EnWarnings(uint8 warnEnable)

Description: Enables/disables assertion of the warn output terminal. Warning status is still available through the GetWarningStatus() API.

Parameters: uint8 warnEnable
Options: 0=disabled, 1=enabled
Enabled when the component is started

Run-time Configuring Functions

Function	Description
Sequencer_SetStsPgoodMask()	Specifies which pgood[x] inputs participate in the generation of the specified general purpose sequencer status output
Sequencer_GetStsPgoodMask()	Returns which pgood[x] inputs participate in the generation of the specified general purpose sequencer status output
Sequencer_SetStsPgoodPolarity()	Configures the logic conditions that will cause the selected general purpose sequencer status output to be asserted

Function	Description
Sequencer_GetStsPgoodPolarity()	Returns the polarity of the pgood[x] inputs used in the AND expression for the selected general purpose sequencer status output
Sequencer_SetPgoodOnThreshold()	Sets the power good voltage threshold for power on detection
Sequencer_GetPgoodOnThreshold()	Returns the power good voltage threshold for power on detection
Sequencer_SetPowerUpMode()	Sets the power up default state for the selected power converter
Sequencer_GetPowerUpMode()	Returns the power up default state for the selected power converter
Sequencer_SetPgoodOnPrereq()	Determines which pgood[x] inputs are power up pre-requisites for the selected power converter state machine
Sequencer_GetPgoodOnPrereq()	Returns which pgood[x] inputs are power up pre-requisites for the selected power converter state machine
Sequencer_SetPgoodOffPrereq()	Determines which pgood[x] inputs are power down pre-requisites for the selected power converter state machine
Sequencer_GetPgoodOffPrereq()	Returns which pgood[x] inputs are power down pre-requisites for the selected power converter state machine
Sequencer_SetTonDelay()	Sets the TON delay parameter for the selected power converter
Sequencer_GetTonDelay()	Returns the TON delay parameter for the selected power converter
Sequencer_SetTonMax()	Sets the TON_MAX parameter for the selected power converter
Sequencer_GetTonMax()	Returns the TON_MAX parameter for the selected power converter
Sequencer_SetPgoodOffThreshold()	Sets the power good voltage threshold for power down detection
Sequencer_GetPgoodOffThreshold()	Returns the power good voltage threshold for power down detection
Sequencer_SetCtlPrereq()	Sets which ctl[x] input is a pre-requisite for a power converter
Sequencer_GetCtlPrereq()	Returns which ctl[x] input is a pre-requisite for a power converter
Sequencer_SetCtlShutdownMask()	Determines which ctl[x] inputs will cause the selected power converter to shutdown when de-asserted
Sequencer_GetCtlShutdownMask()	Returns which ctl[x] inputs will cause the selected power converter to shutdown when de-asserted
Sequencer_SetPgoodShutdownMask()	Determines which other pgood[x] inputs will shutdown the selected power converter when de-asserted
Sequencer_GetPgoodShutdownMask()	Returns which other pgood[x] inputs will shutdown the selected power converter when de-asserted
Sequencer_SetToffDelay()	Sets the TOFF delay parameter for the selected power converter
Sequencer_GetToffDelay()	Returns the TOFF delay parameter for the selected power converter
Sequencer_SetToffMax()	Sets the TOFF_MAX_DELAY parameter for the selected power converter

Function	Description
Sequencer_GetToffMax()	Returns the TOFF_MAX_DELAY parameter for the selected power converter
Sequencer_SetSysStableTime()	Sets the global System Stable parameter for all power converter state machines
Sequencer_GetSysStableTime()	Returns the global System Stable parameter for all power converter state machines
Sequencer_SetReseqDelay()	Sets the global Re-sequence Delay parameter for all power converter state machines
Sequencer_GetReseqDelay()	Returns the global Re-sequence Delay parameter for all power converter state machines
Sequencer_SetTonMaxReseqCnt()	Sets the re-sequence count for TON_MAX fault condition
Sequencer_GetTonMaxReseqCnt()	Returns the re-sequence count for TON_MAX fault conditions
Sequencer_SetTonMaxFaultResp()	Sets the shutdown mode for a fault group when a TON_MAX fault condition occurs on the selected master converter
Sequencer_GetTonMaxFaultResp()	Returns the shutdown mode for a fault group when a TON_MAX fault condition occurs on the selected master converter
Sequencer_SetCtlReseqCnt()	Sets the re-sequence count for fault conditions due to de-asserted ctl[x] inputs
Sequencer_GetCtlReseqCnt()	Returns the re-sequence count for fault conditions due to de-asserted ctl[x] inputs
Sequencer_SetCtlFaultResp()	Sets the shutdown mode for a fault group in response to fault conditions due to de-asserted ctl[x] inputs
Sequencer_GetCtlFaultResp()	Returns the shutdown mode for a fault group in response to fault conditions due to de-asserted ctl[x] inputs
Sequencer_SetFaultReseqSrc()	Sets the power converter fault re-sequence sources
Sequencer_GetFaultReseqSrc()	Returns the power converter fault re-sequence sources
Sequencer_SetPgoodReseqCnt()	Sets the re-sequence count for fault conditions due to de-asserted pgood[x] inputs
Sequencer_GetPgoodReseqCnt()	Returns the re-sequence count for fault conditions due to de-asserted pgood[x] inputs
Sequencer_SetPgoodFaultResp()	Sets the shutdown mode for a fault group due to de-asserted pgood[x] inputs
Sequencer_GetPgoodFaultResp()	Returns the shutdown mode a fault group due to de-asserted pgood[x] inputs
Sequencer_SetOvReseqCnt()	Sets the re-sequence count for over-voltage (OV) fault conditions
Sequencer_GetOvReseqCnt()	Returns the re-sequence count for over-voltage (OV) fault conditions

Function	Description
Sequencer_SetOvFaultResp()	Sets the shutdown mode for a fault group due to overvoltage (OV) fault conditions
Sequencer_GetOvFaultResp()	Returns the shutdown mode for a fault group due to overvoltage (OV) fault conditions
Sequencer_SetUvReseqCnt()	Sets the re-sequence count for under-voltage (UV) fault conditions
Sequencer_GetUvReseqCnt()	Returns the re-sequence count for under-voltage (UV) fault conditions
Sequencer_SetUvFaultResp()	Sets the shutdown mode for a fault group due to under-voltage (UV) fault conditions
Sequencer_GetUvFaultResp()	Returns the shutdown mode for a fault group due to under-voltage (UV) fault conditions
Sequencer_SetOcReseqCnt()	Sets the re-sequence count for over-current (OC) fault conditions
Sequencer_GetOcReseqCnt()	Returns the re-sequence count for over-current (OC) fault conditions
Sequencer_SetOcFaultResp()	Sets the shutdown mode for a fault group due to overcurrent (OC) fault conditions
Sequencer_GetOcFaultResp()	Returns the shutdown mode for a fault group due to overcurrent (OC) fault conditions
Sequencer_SetFaultMask()	Sets which power converters have fault detection enabled
Sequencer_GetFaultMask()	Returns which power converters have fault detection enabled
Sequencer_SetWarnMask()	Sets which power converters have warnings enabled
Sequencer_GetWarnMask()	Returns which power converters have warnings enabled

void Sequencer_SetStsPgoodMask(uint8 stsNum, uint8/uint16/uint32 stsPgoodMask)

Description: Specifies which pgood[x] inputs participate in the generation of the specified general purpose sequencer control output (sts[x])

Parameters: uint8 stsNum: Specifies the status output number. Valid range: 1-6

uint8/uint16/uint32 stsPgoodMask. Depends on the number of converters

Bit Field	Status Pgood Mask
0	1=Sts output depends on pgood[1]
1	1=Sts output depends on pgood[2]
…	…
31	1=Sts output depends on pgood[32]

uint8/uint16/uint32 Sequencer_GetStsPgoodMask(uint8 stsNum)

Description: Returns which pgood[x] inputs participate in the generation of the specified general purpose sequencer control output (sts[x])

Parameters: uint8 stsNum: Specifies the status output number. Valid range: 1-6

Return Value: uint8/uint16/uint32 stsPgoodMask. Depends on the number of converters

Bit Field	Status Pgood Mask
0	1=Sts output depends on pgood[1]
1	1=Sts output depends on pgood[2]
…	…
31	1=Sts output depends on pgood[32]

void Sequencer_SetStsPgoodPolarity(uint8 stsNum, uint8/uint16/uint32 pgoodPolarity)

Description: Configures the logic conditions that will cause the selected general purpose sequencer control output (sts[x]) to be asserted

Parameters: uint8 stsNum: Specifies the status output number. Valid range: 1-6

uint8/uint16/uint32 stsPgoodPolarity. Depends on the number of converters. Specifies the polarity of the pgood[x] inputs required to assert the specified sts[x] output

Bit Field	Status Polarity
0	0=pgood[1] must be low, 1=pgood[1] must be high
1	0=pgood[2] must be low, 1=pgood[2] must be high
…	…
31	0=pgood[32] must be low, 1=pgood[32] must be high

uint8/uint16/uint32 Sequencer_GetStsPgoodPolarity(uint8 stsNum)

Description: Returns the polarity of the pgood[x] inputs used in the AND expression for the selected general purpose sequencer control output (sts[x]).

Parameters: uint8 stsNum: Specifies the status output number. Valid range: 1-6

Return Value: uint8/uint16/uint32 stsPgoodPolarity. Depends on the number of converters. Polarity of the pgood[x] inputs required to assert the specified sts[x] output

Bit Field	Status Polarity
0	0=pgood[1] must be low, 1=pgood[1] must be high
1	0=pgood[2] must be low, 1=pgood[2] must be high
…	…
31	0=pgood[32] must be low, 1=pgood[32] must be high

void Sequencer_SetPgoodOnThreshold(uint8 converterNum, uint16 onThreshold)

Description: Sets the power good voltage threshold for power on detection

Parameters: uint8 ctlNum: Specifies the converter number. Valid range: 1-32

uint16 onThreshold: Specifies the power good power on threshold in mV
Valid range: 0-65535

uint16 Sequencer_GetPgoodOnThreshold(uint8 converterNum)

Description: Returns the power good voltage threshold for power on detection

Parameters: uint8 ctlNum: Specifies the converter number. Valid range: 1-32

Return Value: uint16 onThreshold: The power good power on threshold in mV
Valid range: 0-65535

void Sequencer_SetPowerUpMode(uint8 converterNum, uint8 powerUpMode)

Description: Sets the power up default state for the selected power converter

Parameters: uint8 converterNum: Specifies the power converter number.
Valid range: 1-32

uint8 powerUpMode: Specifies the power up mode for the selected power converter

Options: 0=manual, 1=automatic

uint8 Sequencer_GetPowerUpMode(uint8 converterNum)

Description: Returns the power up default state for the selected power converter

Parameters: uint8 converterNum: Specifies the power converter number.
Valid range: 1-32

Return Value: The power up mode for the selected power converter
Options: 0=manual, 1=automatic

void Sequencer_SetPgoodOnPrereq(uint8 converterNum, uint8/uint16/uint32 pgoodMask)

Description: Determines which pgood[x] inputs are power up pre-requisites for the selected power converter

Parameters: uint8 converterNum: Specifies the power converter number.
Valid range: 1-32

uint8/uint16/uint32 pgoodMask. Depends on the number of converters
Specifies which pgood[x] inputs are power up pre-requisites for the selected power converter

Bit Field	Power Good Power Up Pre-Requisite Mask
0	1=pgood[1] must be asserted
1	1=pgood[2] must be asserted
…	…
31	1=pgood[32] must be asserted

uint8/uint16/uint32 Sequencer_GetPgoodOnPrereq(uint8 converterNum)

Description: Returns which pgood[x] inputs are power up pre-requisites for the selected power converter

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint8/uint16/uint32 pgoodMask. Depends on the number of converters
Specifies which pgood[x] inputs are power up pre-requisites for the selected power converter

Bit Field	Power Good Power Up Pre-Requisite Mask
0	1=pgood[1] must be asserted
1	1=pgood[2] must be asserted
…	…
31	1=pgood[32] must be asserted

void Sequencer_SetPgoodOffPrereq(uint8 converterNum, uint8/uint16/uint32 pgoodMask)

Description: Determines which pgood[x] inputs are power down pre-requisites for the selected power converter

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

uint8/uint16/uint32 pgoodMask. Depends on the number of converters
Specifies which pgood[x] inputs are power down pre-requisites for the selected power converter

Bit Field	Power Good Power Down Pre-Requisite Mask
0	1=pgood[1] must be asserted
1	1=pgood[2] must be asserted
…	…
31	1=pgood[32] must be asserted

uint8/uint16/uint32 Sequencer_GetPgoodOffPrereq(uint8 converterNum)

Description: Returns which pgood[x] inputs are power down pre-requisites for the selected power converter

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint8/uint16/uint32 pgoodMask. Depends on the number of converters
Specifies which pgood[x] inputs are power down pre-requisites for the selected power converter

Bit Field	Power Good Power Down Pre-Requisite Mask
0	1=pgood[1] must be asserted
1	1=pgood[2] must be asserted
…	…
31	1=pgood[32] must be asserted

void Sequencer_SetTonDelay(uint8 converterNum, uint16 tonDelay)

Description: Sets the **TON** delay parameter for the selected power converter. Defined as the time between all power converter's pre-requisites becoming satisfied and the en[x] output being asserted

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

uint16 tonDelay

Number of converters	Units	Valid Range
≤ 16 converters	0.25 ms per LSB	0-65535 (0-16.384 s)
> 16 converters	0.50 ms per LSB	0-65535 (0-32.768 s)

uint16 Sequencer_GetTonDelay(uint8 converterNum)

Description: Returns the **TON** delay parameter for the selected power converter. Defined as the time between all power converter's pre-requisites becoming satisfied and the en[x] output being asserted

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint16 tonDelay

Number of converters	Units	Valid Range
≤ 16 converters	0.25 ms per LSB	0-65535 (0-16.384 s)
> 16 converters	0.50 ms per LSB	0-65535 (0-32.768 s)

void Sequencer_SetTonMax(uint8 converterNum, uint16 tonMax)

Description: Sets the TON_MAX timeout parameter for the selected power converter. Defined as the maximum time allowable between a power converter's en[x] being asserted and its pgood[x] being asserted. Failure to do so generates a fault condition

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

uint16 tonMax

Number of converters	Units	Valid Range
≤ 16 converters	0.25 ms per LSB	0-65535 (0-16.384 s)
> 16 converters	0.50 ms per LSB	0-65535 (0-32.768 s)

uint16 Sequencer_GetTonMax(uint8 converterNum)

Description: Returns the TON_MAX timeout parameter for the selected power converter. Defined as the maximum time allowable between a power converter's en[x] being asserted and its pgood[x] being asserted. Failure to do so generates a fault condition

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint16 tonMax

Number of converters	Units	Valid Range
≤ 16 converters	0.25 ms per LSB	0-65535 (0-16.384 s)
> 16 converters	0.50 ms per LSB	0-65535 (0-32.768 s)

void Sequencer_SetPgoodOffThreshold(uint8 converterNum, uint16 onThreshold)

Description: Sets the power good voltage threshold for power off detection

Parameters: uint8 converterNum: Specifies the converter number
Valid range: 1-32

uint16 offThreshold: Specifies the power good power off threshold in mV
Valid range: 0-65535

uint16 Sequencer_GetPgoodOffThreshold(uint8 converterNum)

Description: Returns the power good voltage threshold for power off detection

Parameters: uint8 converterNum: Specifies the converter number
Valid range: 1-32

Return Value: uint16 offThreshold: The power good power off threshold in mV
Valid range: 0-65535

void Sequencer_SetCtlPrereq (uint8 converterNum, uint8 ctlPinMask)

Description: Sets which ctl[x] input is a pre-requisite for the selected power converter

Parameters: uint8 converterNum: Specifies the converter number
Valid range: 1-32

uint8 ctlPinMask: Specifies which ctl[x] inputs are power up pre-requisites:

Bit Field	Control Pin Pre-requisite Mask
0	1=ctl[1] is a power up pre-requisite
1	1=ctl[2] is a power up pre-requisite
…	…
5	1=ctl[6] is a power up pre-requisite
7..6	Reserved. Set to zeroes

uint8 Sequencer_GetCtlPrereq (uint8 converterNum)

Description: Returns which ctl[x] input is a pre-requisite for the selected power converter

Parameters: uint8 converterNum: Specifies the converter number. Valid range: 1-32

Return Value: uint8 ctlPinMask: Specifies which ctl[x] inputs are power up pre-requisites:

Bit Field	Control Pin Pre-requisite Mask
0	1=ctl[1] is a power up pre-requisite
1	1=ctl[2] is a power up pre-requisite
…	…
5	1=ctl[6] is a power up pre-requisite
7..6	Reserved. Set to zeroes

void Sequencer_SetCtlShutdownMask(uint8 converterNum, uint8 ctlPinMask)

Description: Determines which ctl[x] inputs will cause the selected power converter to shutdown when de-asserted

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32
uint8 ctlPinMask: Specifies which ctl[x] inputs can cause a shutdown

Bit Field	Control Pin Pre-requisite Mask
0	1=ctl[1] de-assertion will shutdown the converter
1	1=ctl[2] de-assertion will shutdown the converter
…	…
5	1=ctl[6] de-assertion will shutdown the converter
7..6	Reserved. Set to zeroes

uint8 Sequencer_GetCtlShutdownMask(uint8 converterNum)

Description: Returns which ctl[x] inputs will cause the selected power converter to shutdown when de-asserted

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint8 ctlPinMask
Specifies which ctl[x] inputs can generate fault conditions

Bit Field	Control Pin Shutdown Mask
0	1=ctl[1] de-assertion will shutdown the converter
1	1=ctl[2] de-assertion will shutdown the converter
…	…
5	1=ctl[6] de-assertion will shutdown the converter
7..6	Reserved. Set to zeroes

void Sequencer_SetPgoodShutdownMask(uint8 converterNum, uint8/uint16/uint32 pgoodMask)

Description: Determines which converter's pgood[x] inputs will shutdown the selected power converter when de-asserted.

Note that a converter's own pgood[x] input is automatically a fault source for that converter whether or not the corresponding bit in the pgoodMask is set or not.

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32
uint8/uint16/uint32 pgoodMask. Depends on the number of converters
Specifies which pgood[x] inputs can cause a shutdown

Bit Field	Power Good Mask
0	1=pgood[1] de-assertion will shutdown the converter
1	1=pgood[2] de-assertion will shutdown the converter
…	…
31	1=pgood[32] de-assertion will shutdown the converter

uint8/uint16/uint32 Sequencer_GetPgoodShutdownMask (uint8 converterNum)

Description: Returns which converter's pgood[x] inputs will shutdown the selected power converter when de-asserted. Note that a converter's own pgood[x] input is automatically a fault source for that converter and the corresponding mask bit is not returned

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint8/uint16/uint32 pgoodMask. Depends on the number of converters
Specifies which pgood[x] inputs can cause a shutdown

Bit Field	Power Good Mask
0	1=pgood[1] de-assertion will shutdown the converter
1	1=pgood[2] de-assertion will shutdown the converter
…	…
31	1=pgood[32] de-assertion will shutdown the converter

void Sequencer_SetToffDelay(uint8 converterNum, uint16 toffDelay)

Description: Sets the TOFF delay parameter for the selected power converter. Defined as the time between making the decision to turn a power converter off and to actually de-asserting the en[x] output

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32
uint16 toffDelay

Number of converters	Units	Valid Range
≤ 16 converters	0.25 ms per LSB	0-65535 (0-16.384 s)
> 16 converters	0.50 ms per LSB	0-65535 (0-32.768 s)

uint16 Sequencer_GetToffDelay(uint8 converterNum)

Description: Returns the TOFF delay parameter for the selected power converter. Defined as the time between making the decision to turn a power converter off and to actually de-asserting the en[x] output

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint16 toffDelay

Number of converters	Units	Valid Range
≤ 16 converters	0.25 ms per LSB	0-65535 (0-16.384 s)
> 16 converters	0.50 ms per LSB	0-65535 (0-32.768 s)

void Sequencer_SetToffMax(uint8 converterNum, uint16 toffMax)

Description: Sets the TOFF_MAX_DELAY timeout parameter for the selected power converter. Defined as the maximum time allowable between a power converter's en[x] being de-asserted and power converter actually turning off. Failure to do so generates a warning condition

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32
uint16 toffMax

Number of converters	Units	Valid Range
≤ 16 converters	0.25 ms per LSB	0-65535 (0-16.384 s)
> 16 converters	0.50 ms per LSB	0-65535 (0-32.768 s)

uint16 Sequencer_GetToffMax(uint8 converterNum)

Description: Returns the TOFF_MAX_DELAY timeout parameter for the selected power converter. Defined as the maximum time allowable between a power converter's en[x] being de-asserted and power converter actually turning off. Failure to do so generates a warning condition

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint16 toffMax

Number of converters	Units	Valid Range
≤ 16 converters	0.25 ms per LSB	0-65535 (0-16.384 s)
> 16 converters	0.50 ms per LSB	0-65535 (0-32.768 s)

void Sequencer_SetSysStableTime(uint16 stableTime)

Description: Sets the global TRESEQ_DELAY parameter for all power converters. Defined as the time between making the decision to re-sequence and beginning a new power up sequence

Parameters: uint16 stableTime
units = 8 ms per LSB. Valid Range=0-65535 (0-534.28 s)

uint16 Sequencer_GetSysStableTime(void)

Description: Sets the global TRESEQ_DELAY parameter for all power converters. Defined as the time between making the decision to re-sequence and beginning a new power up sequence

Return Value: uint16 stableTime
units = 8 ms per LSB
Valid Range=0-65535 (0-534.28 s)

void Sequencer_SetReseqDelay(uint16 reseqDelay)

Description: Sets the global TRESEQ_DELAY parameter for all powers. Defined as the time between making the decision to re-sequence and beginning a new power up sequence

Parameters: uint16 reseqDelay
units = 8 ms per LSB
Valid Range=0-65535 (0-534.28 s)

uint16 Sequencer_GetReseqDelay(void)

Description: Returns the global TRESEQ_DELAY parameter for all power converters. Defined as the time between making the decision to re-sequence and beginning a new power up sequence

Return Value: uint16 reseqDelay
units = 8 ms per LSB
Valid Range=0-65535 (0-534.28 s)

void Sequencer_SetTonMaxReseqCnt(uint8 converterNum, uint8 ReseqCnt)

Description: Sets the re-sequence count for TON_MAX fault conditions

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32
uint8 reseqCnt
Options: 0=no re-sequencing, 31=infinite re-sequencing, 1-30=valid re-sequencing counts

uint8 Sequencer_GetTonMaxReseqCnt(uint8 converterNum)

Description: Returns the re-sequence count for TON_MAX fault conditions

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint8 reseqCnt
Options: 0=no re-sequencing, 31=infinite re-sequencing, 1-30=valid re-sequencing counts

void Sequencer_SetTonMaxFaultResp(uint8 converterNum, uint8 faultResponse)

Description: Sets the shutdown mode for all associated fault groups when a TON_MAX fault condition occurs on the selected power converter

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32
uint8 faultResponse: Specifies the shutdown mode for all associated fault groups
Options: 0=immediate, 1=soft

uint8 Sequencer_GetTonMaxFaultResp(uint8 converterNum)

Description: Returns the shutdown mode for all associated fault group when a TON_MAX fault condition occurs on the selected power converter

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint8 faultResponse: The shutdown mode for all associated fault groups
Options: 0=immediate, 1=soft

void Sequencer_SetCtlReseqCnt(uint8 converterNum, uint8 reseqCnt)

Description: Sets the re-sequence count for fault conditions due to de-asserted ctl[x] inputs

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32
uint8 reseqCnt
0=no re-sequencing, 31=infinite re-sequencing,
1-30=valid re-sequencing counts

uint8 Sequencer_GetCtlReseqCnt(uint8 converterNum)

Description: Returns the re-sequence count for fault conditions due to de-asserted ctl[x] inputs

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint8 reseqCnt
0=no re-sequencing, 31=infinite re-sequencing,
1-30=valid re-sequencing counts

void Sequencer_SetCtlFaultResp(uint8 converterNum, uint8 faultResponse)

Description: Sets the shutdown mode for the selected power converter and rails in associated fault groups in response to de-assertion of ctl[x] inputs

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32
uint8 faultResponse: Specifies the shutdown mode for all associated fault groups
Options: 0=immediate, 1=soft

uint8 Sequencer_GetCtlFaultResp(uint8 converterNum)

Description: Returns the shutdown mode for the selected power converter and rails in associated fault groups in response to de-assertion of ctl[x] inputs

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint8 faultResponse: The shutdown mode for all associated fault groups
Options: 0=immediate, 1=soft

void Sequencer_SetFaultReseqSrc(uint8 converterNum, uint8 reseqSrc)

Description: Sets the power converter fault re-sequence sources

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

uint8 reseqSrc

Bit Field	Re-Sequence Source
0	1=OV fault source enabled
1	1=UV fault source enabled
2	1=OC fault source enabled
7:3	Reserved

Side Effects: When reseqSrc is zero, power good (pgood) inputs become the fault re-sequence source.

uint8 Sequencer_GetFaultReseqSrc(uint8 converterNum)

Description: Returns the power converter fault re-sequence source

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint8 reseqSrc

Bit Field	Re-Sequence Source
0	1=OV fault source enabled
1	1=UV fault source enabled
2	1=OC fault source enabled
7:3	Reserved

void Sequencer_SetPgoodReseqCnt(uint8 converterNum, uint8 reseqCnt)

Description: Sets the re-sequence count for fault conditions due to a de-asserted pgood[x] input on the selected rail

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

uint8 reseqCnt
0=no re-sequencing, 31=infinite re-sequencing,
1-30=valid re-sequencing counts

uint8 Sequencer_GetPgoodReseqCnt(uint8 converterNum)

Description: Returns the re-sequence count for fault conditions due to a de-asserted pgood[x] input on the selected rail

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint8 reseqCnt
0=no re-sequencing, 31=infinite re-sequencing,
1-30=valid re-sequencing counts

void Sequencer_SetPgoodFaultResp(uint8 converterNum, uint8 faultResponse)

Description: Sets the shutdown mode for the selected power converter and rails in associated fault groups in response to de-assertion of the selected power converter's pgood[x] input

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32
uint8 faultResponse: Specifies the shutdown mode for all associated fault groups
Options: 0=immediate, 1=soft

uint8 Sequencer_GetPgoodFaultResp(uint8 converterNum)

Description: Returns the shutdown mode for the selected power converter and rails in associated fault groups in response to de-assertion of the selected power converter's pgood[x] input

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint8 faultResponse: The shutdown mode for all associated fault groups
Options: 0=immediate, 1=soft

void Sequencer_SetOvReseqCnt(uint8 converterNum, uint8 reseqCnt)

Description: Sets the re-sequence count for over-voltage (OV) fault conditions

Parameters: uint8 converterNum
Specifies the power converter number
Valid range: 1-32
uint8 reseqCnt
0=no re-sequencing, 31=infinite re-sequencing, 1-30=valid re-sequencing counts

uint8 Sequencer_GetOvReseqCnt(uint8 converterNum)

Description: Sets the re-sequence count for over-voltage (OV) fault conditions

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint8 reseqCnt
0=no re-sequencing, 31=infinite re-sequencing, 1-30=valid re-sequencing counts

void Sequencer_SetOvFaultResp(uint8 converterNum, uint8 faultResponse)

Description: Sets the shutdown mode for all associated fault groups due to over-voltage (OV) fault conditions on the selected power converter

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32
uint8 faultResponse: Specifies the shutdown mode for all associated fault groups
Options: 0=immediate, 1=soft

uint8 Sequencer_GetOvFaultResp(uint8 converterNum)

Description: Returns the shutdown mode for all associated fault groups due to over-voltage (OV) fault conditions on the selected power converter

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint8 faultResponse: The shutdown mode for all associated fault groups
Options: 0=immediate, 1=soft

void Sequencer_SetUvReseqCnt(uint8 converterNum, uint8 reseqCnt)

Description: Sets the re-sequence count for under-voltage (UV) fault conditions

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32
uint8 reseqCnt
0=no re-sequencing, 31=infinite re-sequencing, 1-30=valid re-sequencing counts

void Sequencer_SetUvFaultResp(uint8 converterNum, uint8 faultResponse)

Description: Sets the shutdown mode for all associated fault groups due to under-voltage (UV) fault conditions on the selected power converter

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32
uint8 faultResponse: Specifies the shutdown mode for all associated fault groups
Options: 0=immediate, 1=soft

uint8 Sequencer_GetUvFaultResp(uint8 converterNum)

Description: Returns the shutdown mode for all associated fault groups due to under-voltage (UV) fault conditions on the selected power converter

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint8 faultResponse: The shutdown mode for all associated fault groups
Options: 0=immediate, 1=soft

void Sequencer_SetOcReseqCnt(uint8 converterNum, uint8 reseqCnt)

Description: Sets the re-sequence count for over-current (OC) fault conditions

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32
uint8 reseqCnt
0=no re-sequencing, 31=infinite re-sequencing, 1-30=valid re-sequencing counts

uint8 Sequencer_GetOcReseqCnt(uint8 converterNum)

Description: Returns the re-sequence count for over-current (OC) fault conditions

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint8 reseqCnt
0=no re-sequencing, 31=infinite re-sequencing, 1-30=valid re-sequencing counts

void Sequencer_SetOcFaultResp(uint8 converterNum, uint8 faultResponse)

Description: Sets the shutdown mode for all associated fault groups due to over-current (OC) fault conditions on the selected power converter

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32
uint8 faultResponse: Specifies the shutdown mode for all associated fault groups
Options: 0=immediate, 1=soft

uint8 Sequencer_GetOcFaultResp(uint8 converterNum)

Description: Returns the shutdown mode for all associated fault groups due to over-current (OC) fault conditions on the selected power converter

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint8 faultResponse: The shutdown mode for all associated fault groups
Options: 0=immediate, 1=soft

void Sequencer_SetFaultMask(uint8/uint16/uint32 faultMask)

Description: Sets which power converters have fault detection enabled

Parameters: uint8/uint16/uint32 faultMask. Depends on the number of converters
All bits are set when the component is started

Bit Field	Fault Mask
0	1=enable fault detection for power converter 1
1	1=enable fault detection for power converter 2
…	…
31	1=enable fault detection for power converter 32

uint8/uint16/uint32Sequencer_GetFaultMask(void)

Description: Returns which power converters have fault detection enabled

Return Value: uint8/uint16/uint32 faultMask. Depends on the number of converters
Fault mask of power converters

Bit Field	Fault Mask
0	1=fault detection for power converter 1 is enabled
1	1=fault detection for power converter 2 is enabled
…	…
31	1=fault detection for power converter 32 is enabled

void Sequencer_SetWarnMask(uint8/uint16/uint32 warnMask)

Description: Sets which power converters have warnings enabled

Parameters: uint8/uint16/uint32 warnMask. Depends on the number of converters
All bits are cleared when the component is started

Bit Field	Warning Mask
0	1=enable warnings for power converter 1
1	1= enable warnings for power converter 2
…	…
31	1= enable warnings for power converter 32

uint8/uint16/uint32 Sequencer_GetWarnMask(void)

Description: Returns which power converters have warnings enabled

Return Value: uint8/uint16/uint32 warnMask. Depends on the number of converters
Warn mask of power converters

Bit Field	Warning Mask
0	1=warnings for power converter 1 are enabled
1	1=warnings for power converter 2 are enabled
…	…
31	1=warnings for power converter 32 are enabled

uint8 Sequencer_GetUvReseqCnt(uint8 converterNum)

Description: Returns the re-sequence count for under-voltage (UV) fault conditions

Parameters: uint8 converterNum: Specifies the power converter number
Valid range: 1-32

Return Value: uint8 reseqCnt
0=no re-sequencing, 31=infinite re-sequencing, 1-30=valid re-sequencing counts

Global Variables

Variable	Description
Sequencer_initVar	Indicates whether the Voltage Sequencer has been initialized
Sequencer_ctlShutdownMaskList[]	Defines which ctl[x] inputs will cause shutdown for each converter
Sequencer_stsPgoodMaskList[]	Defines which pgood[x] inputs are used to generate each sts[x] output
Sequencer_stsPgoodPolarityList[]	Defines the logic conditions for generation of each sts[x] output
Sequencer_pgoodOnThresholdList[]	Defines power good voltage threshold for power on detection
Sequencer_initState[]	Defines the power up default state for each converter
Sequencer_ctlPrereqList[]	Defines which ctl[x] inputs are power up pre-requisites for each converter
Sequencer_pgoodPrereqList[]	Defines which pgood[x] inputs are power up pre-requisites for each converter
Sequencer_tonDelayList[]	Defines TON_DELAY parameter for each power converter
Sequencer_tonMaxDelayList[]	Defines TON_MAX_DELAY parameter for each power converter
Sequencer_pgoodOffThresholdList[]	Defines power good voltage threshold for power off detection
Sequencer_pgoodShutdownMaskList[]	Defines which pgood[x] inputs will cause shutdown for each converter
Sequencer_toffDelayList[]	Defines TOFF_DELAY parameter for each power converter

Variable	Description
Sequencer_toffMaxDelayList[]	Defines TOFF_MAX_DELAY parameter for each power converter
Sequencer_sysStableTime	System Stable Time parameter
Sequencer_globalReseqDelay	Global TRESEQ_DELAY parameter
Sequencer_tonMaxFaultReseqCfg[]	Defines the re-sequence configuration for TON_MAX fault conditions
Sequencer_ctlFaultReseqCfg []	Defines the re-sequence configuration for CTL fault conditions
Sequencer_faultReseqSrcList[]	Defines the power converter fault re-sequence sources
Sequencer_pgoodFaultReseqCfg[]	Defines the re-sequence configuration for pgood fault conditions
Sequencer_ovFaultReseqCfg[]	Defines the re-sequence configuration for OV fault conditions
Sequencer_uvFaultReseqCfg[]	Defines the re-sequence configuration for UV fault conditions
Sequencer_ocFaultReseqCfg[]	Defines the re-sequence configuration for OC fault conditions
Sequencer_faultEnable	Enable/disable assertion of the fault output terminal
Sequencer_faultMask	Defines which power converters have fault detection enabled
Sequencer_faultStatus	Bit mask containing the pgood fault status for all power converters
Sequencer_pgStatus	Bit mask containing the pgood active status for all power converters
Sequencer_warnEnable	Enable/disable assertion of the warn output terminal
Sequencer_warnStatus	Bit mask containing TOFF_MAX_WARN warning status for all power converters
Sequencer_warnMask	Defines which power converters have warnings enabled
Sequencer_ctlStatus	Bit mask containing which ctl[x] inputs have caused a shutdown
Sequencer_operatingMode	Specifies sequencer current operating mode (pause or play).

API Constants

Name	Description
NUMBER_OF_CONVERTERS	Number of converters to sequence
NUMBER_OF_CTL_INPUTS	Number of sequencer control inputs
NUMBER_OF_STS_OUTPUTS	Number of sequencer status outputs
INFINITE_RESEQUENCING	Fixed value = 31 (from PMBus specification)

Sample Firmware Source Code

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

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

MISRA Compliance

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

- 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 Voltage Sequencer component has the following specific deviations:

MISRA-C:2004 Rule	Rule Class (Required/ Advisory)	Rule Description	Description of Deviation(s)
3.4	R	All uses of the #pragma directive shall be documented.	The #pragma directive is required to ensure that the C51 compiler produces efficient code for sequencer state machine interrupt handler.
11.4	A	A cast should not be performed between a pointer to object type and a different pointer to object type.	The casts between a pointer to object type and a different pointer to object type is used to reduce the execution time of processing 16-bit and 32-bit variables on 8051 CPU. The cast is also performed to read/write a 16-bit or 32-bit value from/to a set of 8-bit hardware registers.
19.7	A	A function should be used in preference to a function-like macro.	Deviated since function-like macros are used to allow more efficient code.

This component has the following embedded components: Interrupt and Clock. Refer to the corresponding component datasheet for information on their MISRA compliance and specific deviations.

API Memory Usage

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

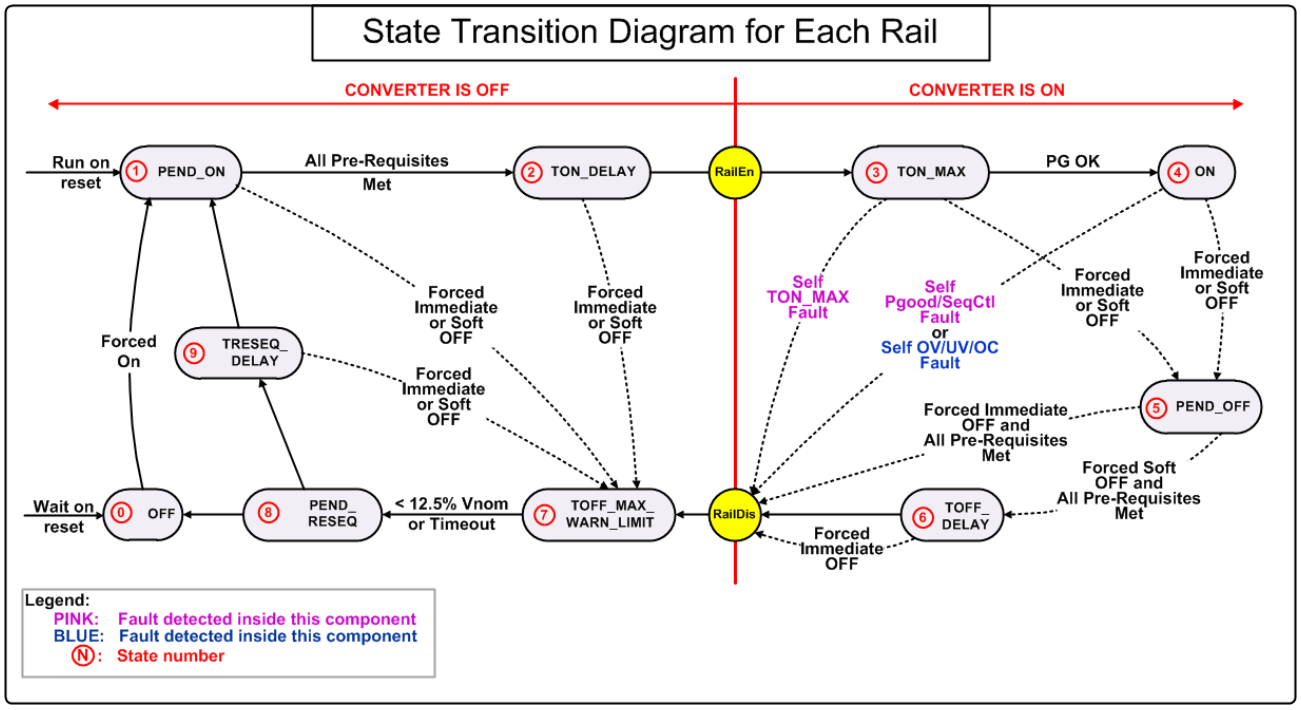
The measurements have been done with associated compiler configured in Release mode with optimization set for Size. For a specific design the map file generated by the compiler can be analyzed to determine the memory usage.

Configuration	PSoC 3 (Keil_PK51)		PSoC 5LP (GCC)	
	Flash KB	SRAM KB	Flash KB	SRAM KB
8 Converters	6.5	0.27	5.2	0.30
16 Converters	8.0	0.59	5.9	0.64
24 Converters	9.7	1.09	6.4	1.18
32 Converters	10.2	1.41	6.8	1.51

Functional Description

Firmware Sequencer State Machine

To support complex event-based sequencing, management of each power converter is done through an independent firmware state machine that drives the enable output (en[x]) for the associated power converter. Each power converter has its own state machine. The state transition flow is shown in the diagram below.



When the component is initialized by calling the Start() API, (after a 'power on' reset for example), all state machines for all of the power converters begin in either the OFF state or the PEND_ON state, under user control. The state machine for each power converter then transitions to a new state depending on how the user defines the sequencing conditions. Power converter fault conditions also drive the associated state machine to a new state, as defined by the user. In the diagram above, the two identified fault response transitions (highlighted in pink and blue color) refer to faults that have occurred on this power converter. At any given point, any of the state machines can be in any one of the defined states.

State machine transitions for every power converter are always handled in the Sequencer State Machine ISR that is invoked every 250 µs (when the number of converters is 16 or less) or 500 µs (when the number of converters is greater than 16). The component has a built-in tick timer clock source, which is automatically configured to produce the appropriate time reference for this ISR. Anytime a fault occurs, the Fault Handler ISR is invoked. The Fault Handler ISR is responsible for time critical activities such as disabling the faulted power converter immediately. It also sets a fault flag that is recognized the next time the Sequencer State Machine ISR is

invoked. The Sequencer State Machine ISR then takes care of non-time critical fault handling activities such as state machine transitions.

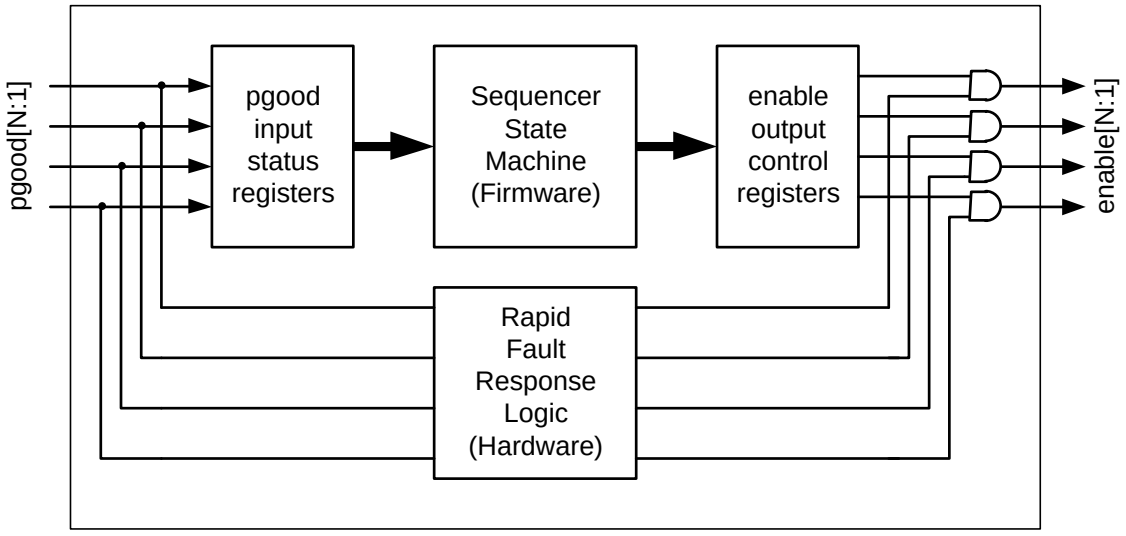
In most real-world applications, power converters have a relationship to each other – they are not truly independent. This may occur when multiple power converters supply power to a single chip or a group of chips. In that case, when one power converter fails, the other power converters must be shutdown also. Another example is that there may be a hardware enforced relationship between two or more power converters. For example, the output of one power converter may be the power supply input of another power converter. In that case, when the primary power converter faults and is be shutdown, it is required to shut down the secondary power converter also because it loses power anyway.

To support these use cases, fault conditions on one power converter state machine must be able to influence state transitions of the state machines for other power converters. To address this requirement, the concept of a Fault Group is introduced. If the user specifies that a fault on one power converter must force a shutdown on one or more operational power converters, then the operational power converters are referred to as the faulty power converter's Fault Group.

The Fault Group can be configured to shut down immediately or go through a soft shutdown process with user-configurable delays. When there is a hardware enforced relationship between power converters, the Fault Group that draws power from the faulty power converter must be set for immediate shutdown to ensure fault conditions are not generated by the Fault Group that is powering down.

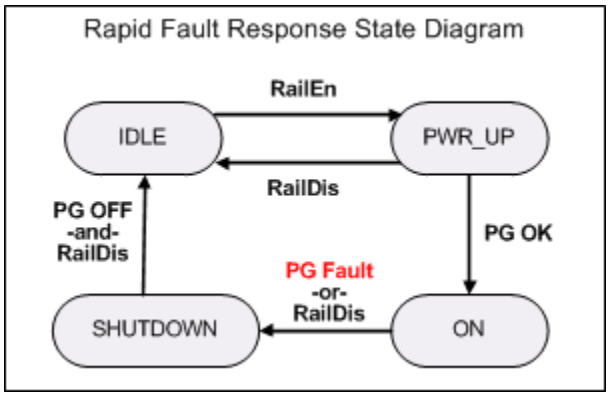
Hardware Rapid Fault Response Logic

To support high-speed fault response shutdown, a hardware block is placed in parallel with the firmware state machine as shown in the diagram below.



The intention is to let firmware control the enable output pins during power up sequencing and during intended power down sequencing. The hardware will de-assert the enable output pins that go to the power converters when a pgood input is de-asserted indicating a fault. In this way, the hardware has the ability to override the firmware in case of emergency.

The Rapid fault response logic is implemented as a simple Verilog state machine shown in the following diagram.



The theory of operation is as follows:

- At power on reset, the hardware state machine is in the IDLE state. The enable output is de-asserted.
- When the firmware state machine asserts the firmware driven enable signal, the hardware state machine transitions to the PWR_UP state and the enable output is asserted.
- If the firmware driven enable signal is de-asserted for some reason, the hardware state machine goes back to the IDLE state. If, on the other hand, the pgood input is asserted, then the hardware state machine transitions to the ON state.
- In the ON state, if the pgood input is de-asserted, the enable signal is de-asserted on the next clock cycle as the hardware state machine transitions to the SHUTDOWN state. This provides the rapid fault response capability. If the firmware controlled enable signal is de-asserted, the same transition occurs. That ensures that firmware can shutdown the power converter if an intended shutdown is required.
- In the SHUTDOWN state, the hardware state machine waits until firmware driven enable signal and pgood are both de-asserted. This allows the firmware state machine to catch up with the hardware state machine. When that occurs, the hardware state machine returns to the IDLE state.

The pgood should never be tied high. Doing so will prevent the power converter from being able to re-sequence. If the hardware state machine transitions to the SHUTDOWN state due to a fault condition, it will not exit from that state until the pgood input is asserted.

Interacting with Other Power Supervision Components

When developing a complete power supervisor solution, the Voltage Sequencer component is not used in isolation. The power good (pgood[x]) inputs to the component can be generated internally to PSoC through the use of the **Voltage Fault Detector** component or the **Power Monitor** component as they have the capability to actively monitor the power converter analog output voltages and load currents. When a fault is detected by either of those components and one or more pgood[x] inputs to the Voltage Sequencer get de-asserted, the Voltage Sequencer can be configured to respond in different ways depending on the type of fault. Therefore, the Voltage Sequencer component needs to be able to interact with those components, which is achieved by calling the APIs which those components provide. Similarly, since the Voltage Sequencer component is responsible for turning power converters on and off, interaction with the **Trim Margin** component is required if it is instantiated in the design.

To simplify this interaction, the API files that are automatically generated by PSoC Creator use the callback mechanism to execute the user provided code. Refer to the PSoC Creator Help and *Component Author Guide* for the more details.

All of these callback functions can be found in the *Sequencer_INT.c* API source code file, located in the **Generated_Source/PSoC3/ Sequencer** or **Generated_Source/PSoC5/Sequencer** folder (depending on the device family targeted) in the **Workspace Explorer** window. Every function call has comments on when and why to call the APIs of the other power supervision components. Note that the folder name examples assume that the instance name of the Voltage Sequencer component is set to **Sequencer**.

In order to add code to the macro callback present in the component's generated source files, perform the following:

- Define a macro to signal the presence of a callback (in *cyapicallbacks.h*). This will "uncomment" the function call from the component's source code.
- Write the function/macro declaration (in *cyapicallbacks.h*). This will make this function visible by all the project files.
- Write the function implementation (in any user file).

Here is a summary of the callbacks and what action should be taken to have the power supervision components interact cohesively:

Declarations and Variables

Use *cyapicallbacks.h* file to include header files from components whose APIs are called from within the *Sequencer_INT.c* file. References to global variables can also be defined in this file.

For example, if interaction with the **Power Monitor** component is required, include *PowerMonitor.h* in the *cyapicallbacks.h* file.

Interactions Related to Power Good Threshold Settings

During normal operation, when all power converter rails are up and stable, each rail can have an under-voltage (UV) and/or over-voltage (OV) threshold that define the acceptable power converter output voltage limits. As long as the converter output voltage remains in that range, the converter is considered good and its corresponding pgood[x] status output is asserted true.

Both the Power Monitor component and the Voltage Fault Detector component support user-defined OV and UV thresholds for each rail, and there are associated firmware APIs that enable users to change those thresholds at run time. When the Voltage Sequencer is turning a powered-down converter on, it may be desirable to increase the UV threshold above the nominal level to give the converter sufficient time to ramp up to a stable voltage level. Once the converter is stable, the nominal UV threshold can be restored.

To support this feature, select the **Enable power good on thresholds** check box on the **Power Up** tab of the Configure dialog of the Voltage Sequencer component. Doing so exposes a new column of parameters that enable users to define the UV threshold for each converter to use during power up.

Similarly, when the Voltage Sequencer turns off a power converter, it may be desirable to confirm that the power converter output voltage truly does decay down to a safe level before taking further action such as re-sequencing. This can be achieved by lowering the UV threshold to a level much lower than the nominal operation threshold forcing the power good output to remain asserted until the power converter voltage decays to a safe level.

To support this feature, select the **Enable power good off** thresholds check box on the **Power Down** tab of the Configure dialog of the Voltage Sequencer component. Doing so exposes a new column of parameters that enable users to define the UV threshold for each converter to use during power down.

In the **SeqStateMachineISR** interrupt service routine, there are several state machine transitions where the user-provided callback functions are invoked to call Power Monitor or Voltage Fault Detector component APIs to make the UV threshold change in order to support the usage model described above.

The state transitions can be summarized as follows:

- Moving from the **TON_DELAY** state to the **TON_MAX** state

  For these transitions, change the UV threshold from the nominal value to the **pgood[x] on threshold** value that was specified in the **Power Up** tab.

- Any transition to the **TOFF_MAX_WARN_LIMIT** state (there are 6).

  For these transitions, change the UV threshold from the nominal value to the **pgood[x] off threshold** value that was specified in the **Power Down** tab

- Moving from the **TON_MAX** to the **ON** state

  For these transitions, change the UV threshold back to its nominal value (for normal fault detection operation)

The Power Monitor component and the Voltage Fault Detector component both have APIs named SetUVFaultThreshold that is used to change the UV threshold. Be sure to use the correct prefix in front of those API names to match the instance name of the component that is providing the pgood[x] inputs to the Voltage Sequencer.

Interactions Related to Fault Source Detection

The **Re-Sequence** tab of the Voltage Sequencer component enables users to specify different automatic re-sequencing behavior depending on the type of fault. This can be enabled by checking any of these checkboxes:

- Enable UV fault re-sequencing
- Enable OV fault re-sequencing
- Enable OC (over-current) fault re-sequencing

Doing so exposes additional parameter columns to enable users to specify the re-sequencing behavior independently for each fault type.

In the **FaultHandlerISR** interrupt service routine, there is one call to user-defined function where users can call APIs in the Power Monitor or Voltage Fault Detector component to determine the fault source on an individual converter basis. Based on that information, the Voltage Sequencer can take appropriate action.

The Voltage Fault Detector component can be configured to support OV and UV fault detection, OV only fault detection or UV only fault detection. Depending on how that component has been configured, call the GetOVUVFaultStatus, GetOVFaultStatus or GetUVFaultStatus APIs to retrieve the fault condition detected on each rail. Note that the Voltage Fault Detector component cannot detect OC faults. The Power Monitor component also provides GetOVFaultStatus and GetUVFaultStatus APIs. If the component has also been configured to support optional power converter load current measurements, then the GetOCFaultStatus API is also available. Call the appropriate API as required to get the re-sequencing functionality needed. Be sure to use the right prefix in front of those API names to match the instance name of the component that is providing the pgood[x] inputs to the Voltage Sequencer.

Interactions Related to Trimming and Margining

Depending on whether a power converter is turned on or off, a different duty cycle is generated on the pulse-width-modulated (PWM) outputs of the Trim Margin component. For detailed information on the operation of that component, please refer to its datasheet. In the **SeqStateMachineISR** interrupt service routine, there are three state machine transitions that require a duty cycle change in the Trim Margin component. The transitions are:

- Moving from the **OFF** state or the **TRESEQ_DELAY** state to the **PEND_ON** state.

  Call the **SetPreRun** API to prepare the converter for imminent power on.

- Moving from the **TON_DELAY** state to the **TON_MAX** state.

  Call the **SetNominal** API to allow the converter to run at nominal voltage output.

Be sure to use the right prefix in front of those API names to match the instance name of the Trim Margin component used in the design.

Registers

The Voltage Sequencer component has several control and status registers that are used by the firmware APIs to control operation and monitor status. None of these registers are accessible directly by user firmware.

Resources

The Voltage Sequencer component is almost entirely firmware based. The component utilizes the following resources.

Configuration	Resource Type					
	Datapath Cells	Macrocells	Status Cells	Control Cells	DMA Channels	Interrupts
8 Converters	–	26	2	5	–	3
16 Converters	–	44	3	7	–	3
24 Converters	–	62	4	9	–	3
32 Converters	–	80	5	11	–	3

DC and AC Electrical Characteristics

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

DC and AC Characteristics

Parameter	Description	Min	Typ	Max	Unit
f_{CLOCK}	Component Clock frequency	–	–	66	MHz
f_{BUS_CLK}	Min Bus Clock Frequency				
	8 Converters	20	–	–	MHz
	16 Converters	30	–	–	MHz
	17-32 Converters	40	–	–	MHz
t_{FAULT_RESP}	Fault response time	$1/f_{CLOCK}$	–	–	ns

Parameter	Description	Min	Typ	Max	Unit
t_{RESEQ_DELAY}	Programmable re-sequence delay	0	–	524	s
N_{RESEQ}	Number of restart attempts	0	–	30 [1]	
Sequencer configured for up to 16 power converters					
$t_{TRANSITION}$	Sequencer state transition time	–	250	275	µs
$f_{TRANSITION}$	Sequencer state transition rate	–	4	–	kHz
t_{ON_DELAY}	Programmable power-on delay	0	–	16.384	s
t_{OFF_DELAY}	Programmable power-off delay	0	–	16.384	s
$t_{ADJUSTMENT}$	Adjustable timer resolution	–	250	–	µs
Sequencer configured for 17-32 power converters					
$t_{TRANSITION}$	Sequencer state transition time	–	500	550	µs
$f_{TRANSITION}$	Sequencer state transition rate	–	2	–	kHz
t_{ON_DELAY}	Programmable power-on delay	0	–	32.768	s
t_{OFF_DELAY}	Programmable power-off delay	0	–	32.768	s
$t_{ADJUSTMENT}$	Adjustable timer resolution	–	500	–	µs

Component Changes

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

Version	Description of Changes	Reason for Changes / Impact
3.40.b	Clarified the threshold usage in Interactions Related to Power Good Threshold Settings.	Documentation correction.
3.40.a	Updated Interacting with Other Power Supervision Components with information on macro callbacks.	Added the callback mechanism to execute user provided code from component API functions.
	Enabled component to be nested into other components.	Support for hierarchical component design.
3.40	Corrected a defect that caused the component to start unintentional re-sequence of a power converter if a fault occurred during it is powering down.	A power converter may have unintentionally resequenced during powering down.

1. A setting of 31 is a special case and gives ∞ attempts.

Version	Description of Changes	Reason for Changes / Impact
	Corrected the component changes made in PSoC Creator 3.0 SP1. For further information, see Knowledge Base Article KBA94159 (www.cypress.com/go/kba94159). Removed Component Errata section.	Correction of the Component Errata item.
	Corrected MISRA rule 19.4 deviation.	
	Datasheet edits and updates.	Reflect all changes made in v3.40.
3.30.a	Edited datasheet to add Component Errata section.	Document that the component was changed, but there is no impact to designs.
3.30	Added the capability for the sequencer to enable/disable the calibration state. While the calibration state is enabled, the sequencer state machine is paused, the enable outputs hold their states and faulty converters are not shutdown. Added the EnableCalibrationState() and DisableCalibrationState() APIs to support this capability.	When the calibration is performed for the Power Monitor or the Voltage Fault Detector component they do not monitor the power converter health and may trigger an erroneous pgood faults to the Voltage Sequencer component. The component will ignore these faults while the calibration state is enabled.
	Added GetPgoodStatus() API that returns active pgood status for all power converters.	Improvements made based on usability feedback.
	Added "Converter fault detection type" parameter on the Re-Sequence tab that allows selecting what type of fault detection and re-sequencing is enabled for each power converter.	
	Added a PEND_OFF state to the sequencer state machine to provide an indication that a power converter is not turned off because its power off pre-requisites are not met.	
	Converter pgood[x] off pre-requisites must be met in the case of intentional shutdown request or due to a fault group shutdown if the fault group is configured to go through a soft shutdown process.	Converter pgood[x] off pre-requisites were applied only in the case of intentional shutdown request and were ignored in the case of shutdown due to a fault condition.
	Updated MISRA Compliance section.	The component has specific deviations described.
	Datasheet edits and updates.	Reflect all changes made in v3.30.
3.20.a	Edited datasheet to remove references to PSoC 5.	PSoC 5 has been replaced by the PSoC 5LP.
3.20	Added custom user code section in the TON_MAX state when the TON_MAX timer expires.	Improvements made based on usability feedback.
	Added a debug capability for the sequencer to assist with board/system bring up. When enabled, provides Pause(), Play() and SingleStep() APIs to step through complex sequencing events under user control.	

Version	Description of Changes	Reason for Changes / Impact
	Added “Converter pgood[x] off pre-reqs” parameter on the Power Down tab that ensures that converters power down sequentially.	
	Simplified presentation of Control Input parameters.	
	Datasheet edits and updates.	Reflect all changes made in v3.20
3.10	Added MISRA Compliance section.	The component was not verified for MISRA compliance.
	Updated for compatibility with PSoC Creator v2.2	
3.0	Added SetCtlPrereq() and GetCtlPrereq() APIs. Removed SetCtlPolarity() and GetCtlPolarity() APIs.	Clarifications to state machine operation and APIs.
	Give users the capability to enable/disable PGOOD ON/OFF thresholds. Simplify power up conditions in the customizer. Removed SetEnPinPrereq(), GetEnPinPrereq(), SetOnCmdPrereq(), GetOnCmdPrereq() APIs and replace with SetPowerUpMode() and GetPowerUpMode().	Improvements made based on usability feedback.
	Datasheet edits and updates.	Reflect all changes made in v3.0
2.0	Complete redesign. Voltage Sequencer v2.0 is NOT AT ALL backwards compatible with previous versions.	Redesigned sequencing engine, to support time-based sequencing as well as more convoluted and complicated event-based sequencing.

Version	Description of Changes
1.50	Updated for compatibility with PSoC Creator v2.0 Re-classified as “Concept” component Corrected intermittent sequencing time delay error
1.40	Status register clocking scheme changed in Verilog file to improve timing performance Corrected error detecting pgood failure of previously enabled rails during up sequencing (error was introduced in v1.30)
1.30	Symbol colors and size updated, resource utilization updated, references to power management APIs removed
1.20	Updated to support PSoC 5
1.10	Updated for PSoC Creator 1.0 Beta 5 compatibility
1.0	First release

© Cypress Semiconductor Corporation, 2016. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC (“Cypress”). This document, including any software or firmware included or referenced in this document (“Software”), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress’s patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage (“Unintended Uses”). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.