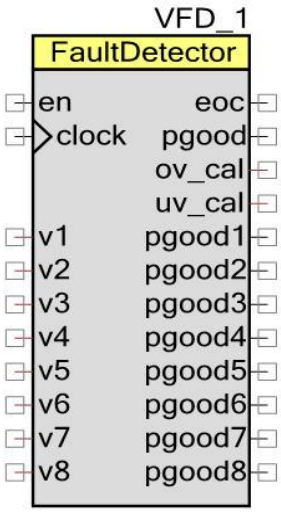
Voltage Fault Detector (VFD)

3.0



Features

- Able to monitor up to 32 voltage inputs
- User-defined over and under voltage limits
- Outputs a good/bad digital status output
- Programmable glitch filter length
- Operates entirely in hardware without any intervention from PSoC's CPU core resulting in known, fixed fault detection latency

General Description

The Voltage Fault Detector (VFD) component provides a simple way to monitor up to 32 voltage inputs against user-defined over and under voltage limits without using the ADC and without having to write any firmware. The component simply outputs a good/bad status result ("power good" or pgood[x]) for each voltage being monitored.

The component operates entirely in hardware without any intervention from PSoC's CPU core resulting in known, fixed fault detection latency.

When to Use a VFD

The VFD component can interface up to 32 voltage inputs and be responsible for determining the health of those voltages by comparing them to either a user-defined under-voltage (UV) threshold, over-voltage (OV) threshold, or both.

Input/Output Connections

This section describes the various input and output connections for the VFD component. A n asterisk (*) next to an I/O type in the following list indicates that the I/O may be hidden on the component symbol under the conditions listed in the description of that I/O.

Clock – Digital Input

The scanning time per rail depends on how much settling delay is necessary when multiplexing the voltage channels. The multiplexing frequency is determined by the settling delay count, the clock input frequency and the internal DMAs transfer time. The total settling delay is calculated by multiplying the settling delay count with the clock input period.

When using voltage DACs (VDACs) to generate OV and UV thresholds, the scanning time per voltage should also consider the maximum VDAC update rate. When the VDACs are configured for 0-1V range, the minimum scanning time should be 1 µs. When the VDACs are configured for 0-4V range, the minimum scanning time should be 1.6 µs.

When external references are selected, you can set the clock frequency and the settling delay count that meets the system requirements. In that case, the VDAC settling time does not need to be factored in, because the VDACs are not present. In this usage, the OV and/or UV thresholds will be common across the entire voltage set to be monitored, so the frequency is limited only by the analog voltage settling time and the maximum frequency of operation of the component's state machine.

In either case, since DMA is involved and needs to run to completion within the time window dictated by the multiplexing frequency selected, this component inherently dictates a minimum BUS_CLK frequency. The component minimum BUS_CLK:clock ratio for this component is 5:1.

Enable (en) – Digital Input

This synchronous active high signal stops the state machine controller. One purpose of this input is to support VDAC calibration. When disabling the component, the PGOOD outputs are kept at their current state.

Note De-asserting the en signal does not stop the component until it completes the current process cycle. Therefore, if the purpose of disabling the component is specifically to change the VDAC settings (for calibration purposes for example), sufficient time should be allowed to let the component run to a completion before an attempt to access the VDACs directly. This can be checked by calling VFD_IsPaused().

Over Voltage Reference (ov_ref) – Analog Input *

This analog input is exposed only when the Enable external reference parameter is selected. In this case, you provide an over voltage threshold that replaces the internal OV VDAC. This can come from a PSoC pin or through a separate instantiation of a VDAC, for example.

Under Voltage Reference (uv_ref) – Analog Input *

This analog input is exposed only when the Enable external reference parameter is selected. In this case, you provide an under voltage threshold that replaces the internal UV VDAC. This can come from a PSoC pin or through a separate instantiation of a VDAC, for example.

Voltages (v[x]) – Analog Input

These analog inputs are the voltages that this component needs to monitor. The number of terminals displayed depends on the number of voltages selected, up to a maximum of 32.

Multiplexed analog inputs (vin_muxed) – Analog Output *

This analog output reflects the analog signal that is currently being monitored (that is, connected to the comparator block). It displays when the Enable vin mux output option is selected.

End of Cycle (eoc) – Digital Output

This terminal pulses active high after every voltage input has been compared to its reference threshold(s). It indicates the end of one complete comparison cycle.

Power Good (pgood) – Digital Output

A single, active high signal indicating all voltages are within range

pgood[x] (individual) – Digital Output

An array of active-high signals, one for each voltage input indicating v[x] is within range.

Over Voltage VDAC (ov_cal) – Analog Output *

This analog output is exposed when the Enable external reference parameter is not selected. The purpose of this is to enable calibration of the OV VDAC. To properly support the calibration activity, the component needs to be disabled through an API call or by de-asserting the en terminal.

Under Voltage VDAC (uv_cal) – Analog Output *

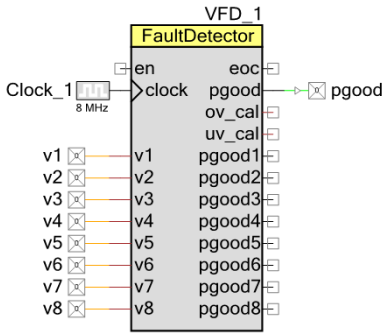
This analog output is exposed when the Enable external reference parameter is not selected. The purpose of this is to enable calibration of the UV VDAC. To properly support the calibration activity, the component needs to be disabled through an API call or by de-asserting the en terminal.

Schematic Macro Information

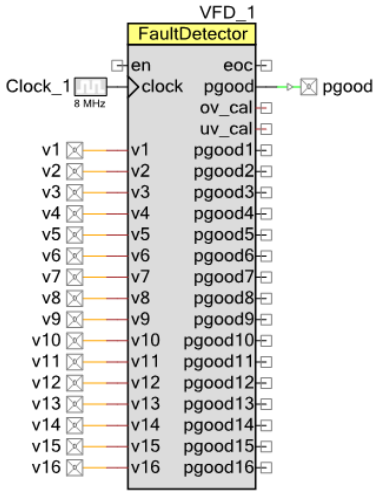
This section contains pertinent information for the schematic macros associated with the VFD.

Schematic Macro Name	Purpose	Description
Voltage Fault Detector – 8 Rails	Intended to be competitive with most off-the-shelf, 8 rail Power Supervisor ASSPs.	Supports 8 voltage inputs and the component is configured for OV/UV window comparison with internal DAC references. Clock input is set to 8MHz.
Voltage Fault Detector – 16 Rails	Intended to be competitive with newer off-the-shelf, 16 rail Power Supervisor ASSPs	Supports 16 voltage inputs and the component is configured for OV/UV window comparison with internal DAC references. Clock input is set to 8MHz.
Voltage Fault Detector – 32 Rails	Intended to demonstrate PSoC's ability to implement a full-featured Power Supervisor with support for more power converters than any other competitor.	Supports 32 voltage inputs and the component is configured for OV/UV window comparison with internal DAC references. Clock input is set to 8MHz.

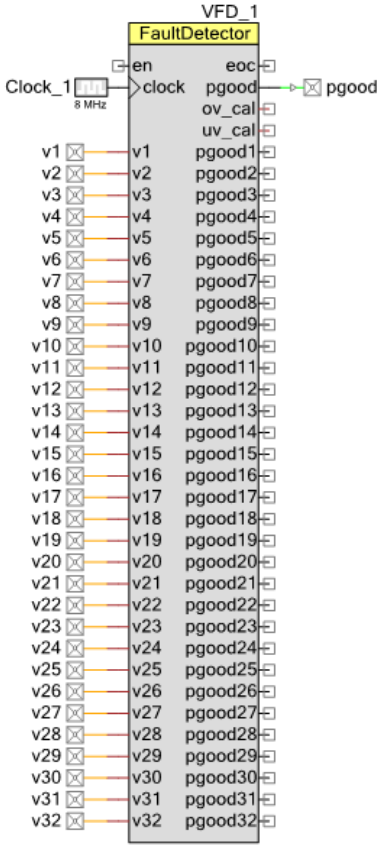
Voltage Fault Detector - 8 Rails



Voltage Fault Detector - 16 Rails



Voltage Fault Detector - 32 Rails



Resources

Digital

The VFD component uses the following digital resources:

Configuration [1] (# of voltages)	Resource Type					
	Datapath Cells	Macrocells	Status Cells	Control Cells	DMA Channels	Interrupts
8	2	37	5	3	5	–
16	2	57	7	3	5	–
24	2	75	9	3	5	–
32	2	93	11	3	5	–

Analog

The VFD component uses the following analog resources:

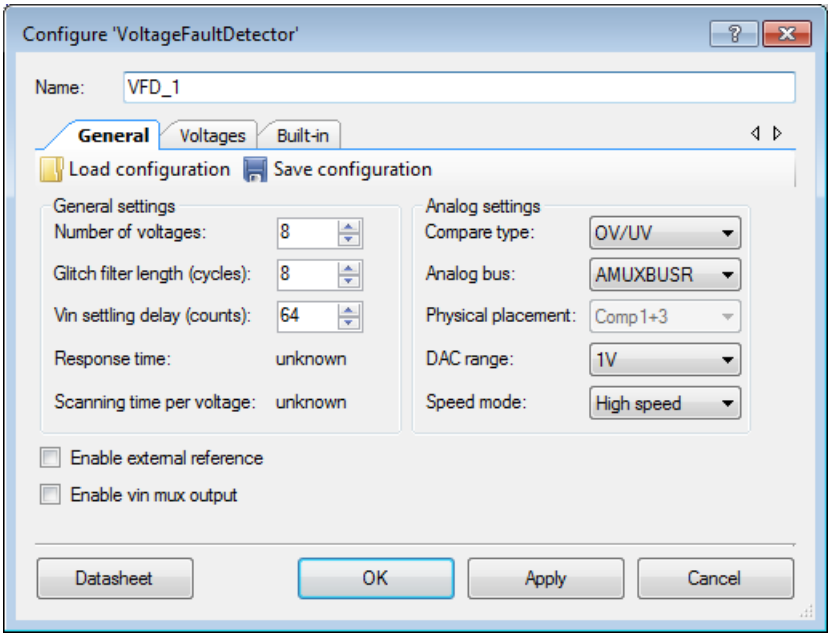
Configuration (Compare Type)	Resource Type	
	Comparators	VDACs [2]
OV only	1	1
UV only	1	1
OV and UV	2	2

1. In all configurations compare type is set to OV/UV and External reference is unchecked. This results in the maximum resource usage.
2. VDACs are not present if External reference parameter is selected.

Component Parameters

Drag a VFD component onto your design schematic and double-click it to open the Configure dialog. This dialog contains the following tabs with various parameters.

General Tab



Load configuration

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

Save configuration

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

Number of voltages

Number of input voltages to be monitored. Range = 1-32 (default = 8).

Glitch filter length

Specifies how many samples the input must be stable before being propagated to the pgood[x] output. The filtering is applied to both rising and falling edge transition. The absolute units depend on the reference clock input, settling delay count, and number of voltages.

Range = 1-255 (default = 8).

The absolute glitch length that will be filtered out is calculated as:

$$t_{PW} = X * (L-1) * (5 + t_{SETTLE}) * T_{CLOCK}$$

where X = number of voltages, L = filter length, t_{SETTLE} = vin settling delay, and T_{CLOCK} = period of the input clock.

Vin settling delay

Specifies the number of clock cycles between the analog inputs being connected to the comparators and the moment the comparator's outputs are sampled. This provides the time delay for the input signal to settle before being compared with fault thresholds. Options = 3-128 (default = 64).. Absolute units depend on the clock input and can be calculated as:

$$t_{SETTLE} = N * T_{CLOCK}$$

where N = entered vin settling delay value and T_{CLOCK} = period of the input clock.

Response time

The time delay between the voltage change on the input is being reflected on the pgood[x] output. That is, the time delay between the fault occurring on the input, and the VFD component responding by de-asserting the pgood[x], or when the voltage becomes in range on the input and the VFD component asserts the pgood[x] output.

Scanning time per voltage

The amount of time each voltage input is processed. This value represents the best case timing, assuming there are no delay cycles incurred by DMA.

Compare type

Pull-down list to select comparator type. Options = OV/UV, OV only, UV only (default = OV/UV).

Analog Bus

Pull-down list to select routing options. It specifies a global routing resource to implement the analog mux. Options = AMUXBUS, AMXUBUSR, AMXUBUSL, Unconstrained (default=AMUXBUSR). When the Unconstrained option is selected, the mux will be automatically routed by PSoC Creator using either an analog mux bus (AMUXBUS) or an analog global (AG) resource.

Some options will be grayed out depending on the Physical Placement parameter setting.

Physical Placement

Pull-down list to select placement options in the analog subsystem. The DAC selection will be tied to the comparator (Comp) selection. That is, DAC0 goes with Comp0, etc. Options = Comp0, Comp1, Comp2, Comp3, Comp0+2, Comp1+3 (default=Comp1+3).

Some options will be grayed out depending on the Compare type parameter setting.

DAC range

Pull-down list to select internal VDAC range. Options = 1V or 4V (default=1V). This pull-down list is grayed out if the Enable external reference parameter is checked. The advantage of the 0-1 V range is that the VDAC update rate is higher (1 MHz) compared to the 625 kHz of the 0-4 V range. This translates to a faster fault detection time, critical in many applications. In either case, all voltages to be monitored need to be scaled such that they fall within the VDAC max limits.

Speed mode

Pull-down list to select the speed mode of internal VDAC(s) and Comparator(s). If the Enable external reference option is selected, this setting is applied to the Comparator(s) only. Options = High speed, Low speed (default = High speed).

Select High speed for the fastest possible fault detection response time. Select Low speed to minimize power consumption. For additional information about speed settings of each of these components, please refer to the appropriate datasheet.

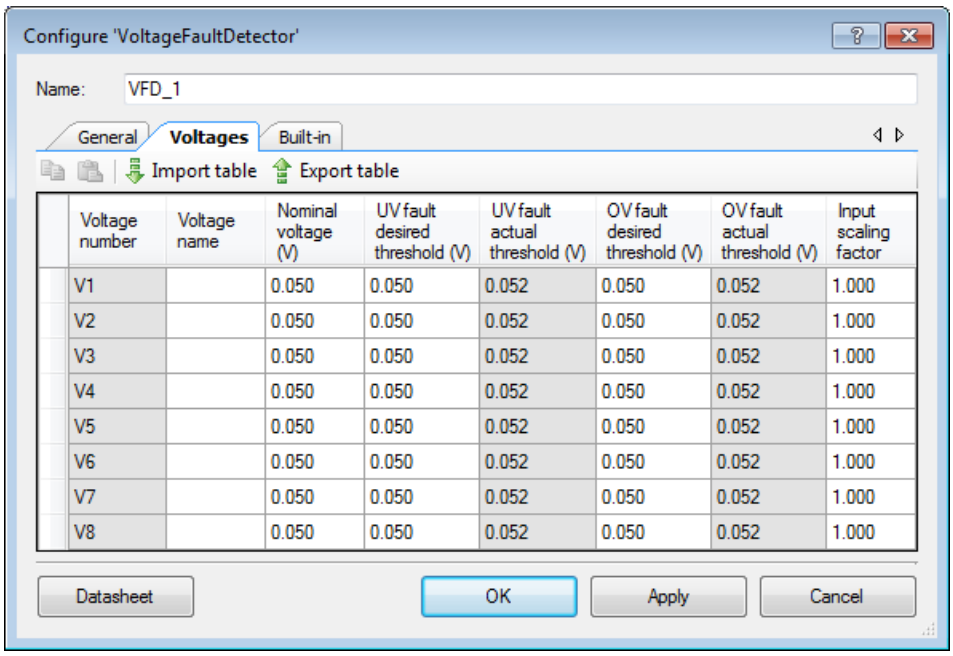
Enable external reference

Check box to enable or disable external references. If checked, the internal OV/UV DACs are removed. This also exposes the ov_ref / uv_ref inputs to provide the fault thresholds. In this usage, the OV and/or UV thresholds will be common across the entire voltage set to be monitored. (Default = unchecked).

Enable vin mux output

If checked, this option exposes the vin_muxed output terminal that is connected to the currently muxed voltage input to the comparator. This terminal is useful for debugging and evaluating the VFD component. The VFD component could be routed to a pin through a PGA along with the ov_cal and/or uv_cal outputs to confirm that the component is working properly and for characterization/validation purposes in the design.

Voltages Tab



Voltage number	Voltage name	Nominal voltage (V)	UV fault desired threshold (V)	UV fault actual threshold (V)	OV fault desired threshold (V)	OV fault actual threshold (V)	Input scaling factor
V1		0.050	0.050	0.052	0.050	0.052	1.000
V2		0.050	0.050	0.052	0.050	0.052	1.000
V3		0.050	0.050	0.052	0.050	0.052	1.000
V4		0.050	0.050	0.052	0.050	0.052	1.000
V5		0.050	0.050	0.052	0.050	0.052	1.000
V6		0.050	0.050	0.052	0.050	0.052	1.000
V7		0.050	0.050	0.052	0.050	0.052	1.000
V8		0.050	0.050	0.052	0.050	0.052	1.000

Import table

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

Export table

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

Voltage name

Text field, 16 characters. For annotation purposes only. By default this field is not populated; no value is required to successfully build the design.

Nominal voltage

Nominal voltage. For annotation purposes only. Range=0.001–65.535 V.

UV fault desired threshold

Desired under voltage fault threshold. Range=0.001–65.535 V.

UV fault actual threshold

Calculated under voltage fault threshold based on DAC range and Input scaling factor. Range=0.001–65.535 V.

OV fault desired threshold

Desired over voltage fault threshold. Range=0.001–65.535 V. When internal VDACs are used to set the OV/UV thresholds, the OV threshold for any given voltage must fall within the VDAC max limits. The general guideline is that all voltages should be scaled such that the nominal voltage is within 85% of the selected VDAC upper limit. This is assuming that the OV threshold is no more than 10% above nominal.

OV fault actual threshold

Calculated over voltage fault threshold based on DAC range and Input scaling factor. Range=0.001–65.535 V.

Input scaling factor

Input voltage scaling factor. Indicates the amount of attenuation applied to the converter output voltage before connecting to the PSoC. Range=0.001-1.000 (default 1.000).

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 " VFD_1" to the first instance of a component in a given design. You can rename it to any unique value that follows the syntactic rules for identifiers. The instance name becomes the prefix of every global function name, variable, and constant symbol. For readability, the instance name used in the following table is "VFD"

Functions

Function	Description
VFD_Start()	Starts the component operation.
VFD_Stop()	Stops the component.
VFD_Init()	Initializes the component.
VFD_Enable()	Enables hardware blocks.
VFD_GetOVUVFaultStatus()	Returns over/under voltagefault status of each voltage input (Applicable if Compare type is set to **OV/UV**).
VFD_GetOVFaultStatus()	Returns over voltagefault status of each voltage input (Applicable if Compare type is set to **OV**).
VFD_GetUVFaultStatus()	Returns under voltagefault status of each voltage input (Applicable if Compare type is set to **UV**).

Function	Description
VFD_SetUVFaultThreshold()	Sets the under voltage fault threshold for the specified voltage input.
VFD_GetUVFaultThreshold()	Returns the under voltage fault threshold for the specified voltage input.
VFD_SetOVFaultThreshold()	Sets the over voltage fault threshold for the specified voltage input.
VFD_GetOVFaultThreshold()	Returns the under voltage fault threshold for the specified voltage input.
VFD_SetUVGlitchFilterLength()	Sets the glitch filter length.
VFD_GetUVGlitchFilterLength()	Returns the glitch filter length.
VFD_SetUVDac()	Sets UV DAC value of each channel.
VFD_GetUVDac()	Gets UV DAC value for the specified voltage input.
VFD_SetOVDac()	Sets OV DAC value of each channel.
VFD_GetOVDac()	Gets OV DAC value for the specified voltage input.
VFD_Pause()	Pauses the state machine and fault detection logic.
VFD_IsPaused()	Checks to see if the component is paused.
VFD_Resume()	Resumes the control state machine and fault detection logic.
VFD_SetUVDacDirect()	Allows manual control of the UV VDAC value.
VFD_GetUVDacDirect()	Returns current UV VDAC.
VFD_SetOVDacDirect()	Allows manual control of the OV VDAC.
VFD_GetOVDacDirect()	Returns current OV VDAC.
VFD_ComparatorCal()	Runs a calibration routine.
VFD_SetSpeed()	Allows setting speed mode for the VDAC(s) (if Internal Reference option is enabled) and Comparator(s).

void VFD_Start(void)

Description: Calls the Init() API if the component has not been initialized before. Runs a calibration routine for comparators and then calls Enable() to begin the component operation.

Parameters: None

Return Value: None

Side Effects: None

void VFD_Stop(void)

Description: Stops the component. Stops DMA controller and resets TDs. Disconnects AMux channels.

Parameters: None

Return Value: None

Side Effects: pgood and pgood[x] outputs are de-asserted.

void VFD_Init(void)

Description: Initializes or restores default VFD configuration provided with the customizer. Initializes internal DMA channels. It is not necessary to call VFD_Init() because the VFD_Start() routine calls this function, which is the preferred method to begin component operation.

Parameters: None

Return Value: None

Side Effects: Disables all interrupts for the moment the counter enabling bits are set. Configures DMA transfer the first time this function is called. Does not reinitialize DMA on subsequent calls. Use VFD_Stop() to reset DMA to the initial state.

void VFD_Enable(void)

Description: Enables hardware blocks, the DMA channels and the control state machine. It is not necessary to call VFD_Init() because the VFD_Start() routine calls this function, which is the preferred method to begin the component operation.

Parameters: None

Return Value: None

Side Effects: Sets/restores the initial TD for the AMux DMA channel, resets glitch filter counters and the current voltage index. This function should not be called when the component is started.

void VFD_GetOVUVFaultStatus(uint32 * ovStatus, uint32 * uvStatus)

Description: Assigns over/under voltage fault status of each voltage input to its parameters. Bits are sticky and cleared by calling this API. Applicable only if Compare type is set to **OV/UV**.

Parameters: ovStatus: Over voltage status for all voltages. OV fault status is sticky.

Bit Field	OV fault status (sticky)
0	1=OV fault condition on voltage input 1
1	1=OV fault condition on voltage input 2
…	…
31	1=OV fault condition on voltage input 32

uvStatus: under voltage status for all voltages.

Bit Field	UV Fault Status
0	1=UV fault condition on voltage input 1
1	1=UV fault condition on voltage input 2
…	…
31	1=UV fault condition on voltage input 32

Return Value: None

Side Effects: Calling this API clears the fault condition source sticky bits. If the condition still persists then the bits will be set again after the response time delay.

void VFD_GetOVFaultStatus(uint32 * ovStatus)

Description: Assigns over voltage fault status of each voltage input to its parameter. Bits are sticky and cleared by calling this API. Applicable only if Compare type is set to **OV**.

Parameters: ovStatus: Over voltage fault status for all voltages.

Bit Field	OV Fault Status
0	1=OV fault condition on voltage input 1
1	1=OV fault condition on voltage input 2
…	…
31	1=OV fault condition on voltage input 32

Return Value: None

Side Effects: Calling this API clears the fault condition source sticky bits. If the condition still persists then the bits will be set again after the response time delay.

void VFD_GetUVFaultStatus(uint32 * uvStatus)

Description: Assigns under voltage fault status of each voltage input to its parameter. Applicable only if Compare type is set to **UV**.

Parameters: uvStatus: Under voltage fault status for all voltages.

Bit Field	UV Fault Status
0	1=UV fault condition on voltage input 1
1	1=UV fault condition on voltage input 2
…	…
31	1=UV fault condition on voltage input 32

Return Value: None

Side Effects: None

cystatus VFD_SetUVFaultThreshold(uint8 voltageNum, uint16 uvFaultThreshold)

Description: Sets the under voltage fault threshold for the specified voltage input. The uvFaultThreshold parameter is converted to a VDAC value and gets written to an SRAM buffer for use by the DMA controller that drives the UV DAC. This API does not apply when the Enable external reference option is selected.

Parameters: voltageNum: Specifies the voltage input number. Valid range: 1-32.

uvFaultThreshold: Specifies the under voltage fault threshold in mV. Valid range: 1-65,535.

Return Value: Returns the following:

Value	Description
CYRET_SUCCESS	Operation is successful.
CYRET_BAD_PARAM	The fault threshold exceeds DAC max value.

Side Effects: uvFaultThreshold value is rounded to fit the VDAC data register format. As a result, the actual threshold value may be different from uvFaultThreshold.

uint16 VFD_GetUVFaultThreshold(uint8 voltageNum)

Description: Returns the under voltage fault threshold for the specified voltage. This API does not apply when the Enable external reference option is selected.

Parameters: voltageNum: Specifies the voltage input number. Valid range: 1-32.

Return Value: The under voltage fault threshold in mV. Valid range: 1-65,535.

Side Effects: None

cystatus VFD_SetOVFaultThreshold(uint8 voltageNum, uint16 ovFaultThreshold)

Description: Sets the over voltage fault threshold for the specified voltage input. The ovFaultThreshold parameter is converted to a VDAC value and gets written to an SRAM buffer for use by the DMA controller that drives the OV DAC. This API does not apply when the Enable external reference option is selected.

Parameters: voltageNum: Specifies the voltage input number. Valid range: 1-32.

ovFaultThreshold: Specifies the over voltage fault threshold in mV. Valid range: 1-65,535.

Return Value: Returns the following:

Value	Description
CYRET_SUCCESS	Operation is successful.
CYRET_BAD_PARAM	The fault threshold exceeds DAC max value.

Side Effects: ovFaultThreshold value is rounded to fit the VDAC data register format. As a result, the actual threshold value may be different from ovFaultThreshold.

uint16 VFD_GetOVFaultThreshold(uint8 voltageNum)

Description: Returns the over voltage fault threshold for the specified voltage input. This API does not apply when the Enable external reference option is selected.

Parameters: voltageNum: Specifies the voltage input number. Valid range: 1-32.

Return Value: The over voltage fault threshold in mV. Valid range: 1-65,535.

Side Effects: None

void VFD_SetGlitchFilterLength(uint8 filterLength)

Description: Sets the glitch filter length.

Parameters: filterLength: Glitch filter length in scanning cycles. Absolute time units depend on input clock frequency. Valid range: 1-255.

Return Value: None

Side Effects: None

uint8 VFD_GetGlitchFilterLength(void)

Description: Returns the glitch filter length.

Parameters: None

Return Value: Glitch filter length in scanning cycles. Absolute time units depend on input clock frequency. Valid range: 1-255.

Side Effects: None

void VFD_SetUVDac(uint8 voltageNum, uint8 dacValue)

Description: Sets the UV DAC value for the specified voltage input. Calling this API does not change the UV VDAC setting immediately. Instead, the dacValue gets written to an SRAM buffer for use by the DMA controller that drives the UV DAC for the specified voltage input. This API does not apply when the Enable external reference option is selected.

Parameters: voltageNum: Specifies the voltage input number. Valid range: 1-32.

dacValue: Specifies the value to be written to the UV VDAC. Valid range: 1-255.

Return Value: None

Side Effects: None

uint8 VFD_GetUVDac(uint8 voltageNum)

Description: Returns the dacValue currently being used by the DMA controller that drives the UV DAC value for the specified voltage input. This API does not apply when the Enable external reference option is selected.

Parameters: voltageNum: Specifies the voltage input number. Valid range: 1-32.

Return Value: UV DAC value for the specified voltage input.

Side Effects: None

void VFD_SetOVDac(uint8 voltageNum, uint8 dacValue)

Description: Calling this API does not change the OV VDAC setting immediately. Instead, the dacValue gets written to an SRAM buffer for use by the DMA controller that drives the OV DAC for the specified voltage input. This API does not apply when the Enable external reference option is selected.

Parameters: voltageNum: Specifies the voltage input number. Valid range: 1-32.

dacValue: Specifies the value to be written to the OV VDAC. Valid range: 1-255.

Return Value: None

Side Effects: None

uint8 VFD_GetOVDac(uint8 voltageNum)

Description: Returns the dacValue currently being used by the DMA controller that drives the OV DAC value for the specified voltage input. This API does not apply when the Enable external reference option is selected.

Parameters: voltageNum: Specifies the voltage input number. Valid range: 1-32.

Return Value: OV DAC value for the specified voltage input.

Side Effects: None

void VFD_Pause(void)

Description: Pauses the controller state machine. The current PGOOD states are kept when the component is paused.

Note that calling this API does not stop the component until it completes the current process cycle. Therefore, if the purpose of calling this API is specifically to change the VDAC settings (for calibration purposes for example), sufficient time should be allowed to let the component run to a completion before an attempt to access the VDACs directly. This can be checked by calling VFD_IsPaused().

Parameters: None

Return Value: None

Side Effects: Stops the fault detection state machine. Does not stop the DMA controller immediately.

bool VFD_IsPaused(void)

Description: Checks to see if the component is paused.

Parameters: None

Return Value: True if the component is paused.

Side Effects: None

void VFD_Resume(void)

Description: Enables the clock to the comparator controller state machine.

Parameters: None

Return Value: None

Side Effects: Restarts the fault detection logic.

void VFD_SetUVDacDirect(uint8 dacValue)

Description: Allows manual control of the UV VDAC value. The dacValue is written directly to the UV VDAC component. Useful for UV VDAC calibration. Note that if the VFD component is running when this API is called, the state machine controller will override the UV VDAC value set by this API call. Call the Pause API to stop the state machine controller if manual UV VDAC control is desired. This API does not apply when the Enable external reference option is selected.

Parameters: dacValue: value to be written directly to the UV VDAC. Valid range: 1-255.

Return Value: None

Side Effects: Calling this API may cause the comparator to trigger a fault condition. To prevent this, call the VFD_Pause() API prior to calling this API.

uint8 VFD_GetUVDacDirect(void)

Description: Returns current UV VDAC. The returned dacValue is read directly from the UV VDAC component. Useful for UV VDAC calibration. Note: if this API is called while the component is running, it isn't possible to know which voltage input the returned UV VDAC value is associated with. Call the Pause API to stop the state machine controller if manual UV VDAC control is desired. This API does not apply when the Enable external reference option is selected.

Parameters: None

Return Value: Current UV VDAC value.

Side Effects: None

void VFD_SetOVDacDirect(uint8 dacValue)

Description: Allows manual control of the OV VDAC value. The dacValue is written directly to the OV VDAC component. Useful for OV VDAC calibration. Note that if the VFD component is running when this API is called, the state machine controller will override the OV VDAC value set by this API call. Call the Pause API to stop the state machine controller if manual OV VDAC control is desired. This API does not apply when the Enable external reference option is selected.

Parameters: uint8 dacValue: value to be written directly to the OV VDAC. Valid range: 1-255.

Return Value: None

Side Effects: Calling this API may cause the comparator to trigger a fault condition. To prevent this, call the VFD_Pause() API prior to calling this API.

uint8 VFD_GetOVDacDirect(void)

Description: Returns current OV VDAC. The returned dacValue is read directly from the VDAC component. This is useful for OV VDAC calibration.

Note If this API is called while the component is running, it is impossible to know which voltage input the returned OV VDAC value is associated with. Call the Pause API to stop the state machine controller if manual UV VDAC control is desired. This API does not apply when the Enable external reference option is selected.

Parameters: None

Return Value: Current OV VDAC value.

Side Effects: None

void VFD_ComparatorCal(uint8 compType)

Description: Runs a calibration routine that measures the selected comparator's offset voltage by shorting its inputs together. It corrects for it by writing to the CMP block's trim register.

A reference voltage in the range at which the comparator will be used must be applied to the negative input of the comparator while the offset calibration is performed.

When Enable external reference is selected, the negative comparator inputs are connected to ov_ref/uv_ref inputs respectively. Therefore, the reference voltage must be provided externally to the VFD component.

When internal reference is selected, the negative comparator inputs are connected to internal OV/UV DACs. In this case, call the SetOVDacDirect() / SetUVDacDirect() function to provide the reference voltage.

Parameters: uint8 compType: Specifies the comparator to calibrate. Valid values: VFD_OV, VFD_UV.

Return Value: None

Side Effects: Calling this API may cause the comparator to trigger a fault condition. To prevent this, call the VFD_Pause() API prior to calling this API.

void VFD_SetSpeed(uint8 speedMode)

Description: Allows setting speed mode for the VDAC(s) (if Internal Reference option is enabled) and Comparator(s).

Parameters: uint8 speedMode: Specifies speed mode of the internal VDAC(s) and comparator(s).

Valid values: VFD_HIGH_SPEED, VFD_LOW_SPEED

Return Value: None

Side Effects: None

Global Variables

Variable	Description
VFD_initVar	The initVar variable is used to indicate initial configuration of this component. The variable is initialized to zero and set to 1 the first time VFD_Start() is called. This allows for component initialization without reinitialization in all subsequent calls to the VFD_Start() routine.
VFD_initUVFaultThreshold[]	Array which contains initial UV fault threshold multiplied by scaling factor for all voltages.
VFD_initOVFaultThreshold[]	Array which contains initial OV fault threshold multiplied by scaling factor for all voltages.
VFD_VoltageScale[]	Array which contains scaling factor values for all voltages.

API Constants

Name	Description
VFD_ NUMBER_OF_VOLTAGES	Number of voltages to be monitored. Range=1-32.
VFD_OV / VFD_UV / VFD_OV_UV	Comparator type. Also used by VFD_ComparatorCal() API to specify the comparator to calibrate.
VFD_HIGH_SPEED / VFD_LOW_SPEED	Speed mode of the internal VDACs and comparators for VFD_SetSpeed() routine.
VFD_EXT_VREF / VFD_INT_VREF	Reference type.

Example Projects, Code Examples, and Application Notes

Example Projects

PSoC Creator provides access to example projects 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.

Code Examples

There are numerous code example projects that include schematics and example code available online at the Cypress Code Examples web page. Code Example CE95398 - Voltage Fault Detector with PSoC 3/5LP demonstrates the use of VFD.

Application Notes

Cypress provides a number of application notes describing how PSoC can be integrated into your design. You can access the Cypress Application Notes search web page (www.cypress.com/appnotes). AN93529 - Introduction to Power Supervision with PSoC 5LP demonstrates how to configure a fully-featured Power Supervision solution using the PSoC Power Supervision Tool. Note this application note is not posted publically. Please contact Cypress technical support to obtain this document.

Deprecated code

The VFD component contains deprecated code that is not recommended for use but is kept to preserve backward compatibility with the existing designs. The deprecated code is located under the following comment header in the component source files:

```
/***************************************
* The following code is DEPRECATED and
* should not be used in new projects.
***************************************/
```

Follow instructions in the following table for how to update your design.

What is deprecated	Reason for deprecation	How to handle it
VFD_External_Reference	Wrong naming conventions.	Use VFD_EXT_VREF
VFD_CompareType	Wrong naming conventions.	Use VFD_COMP_TYPE
VFD_DAC_VOL_DIVIDER	Wrong naming conventions.	Use VFD_mV_LSb
VFD_Dac_Range	Wrong naming conventions.	Use VFD_DAC_RANGE
VFD_OV_ONLY	Redundant. Consolidated to VFD_OV.	Use VFD_OV
VFD_UV_ONLY	Redundant. Consolidated to VFD_UV.	Use VFD_UV

API Memory Usage

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

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

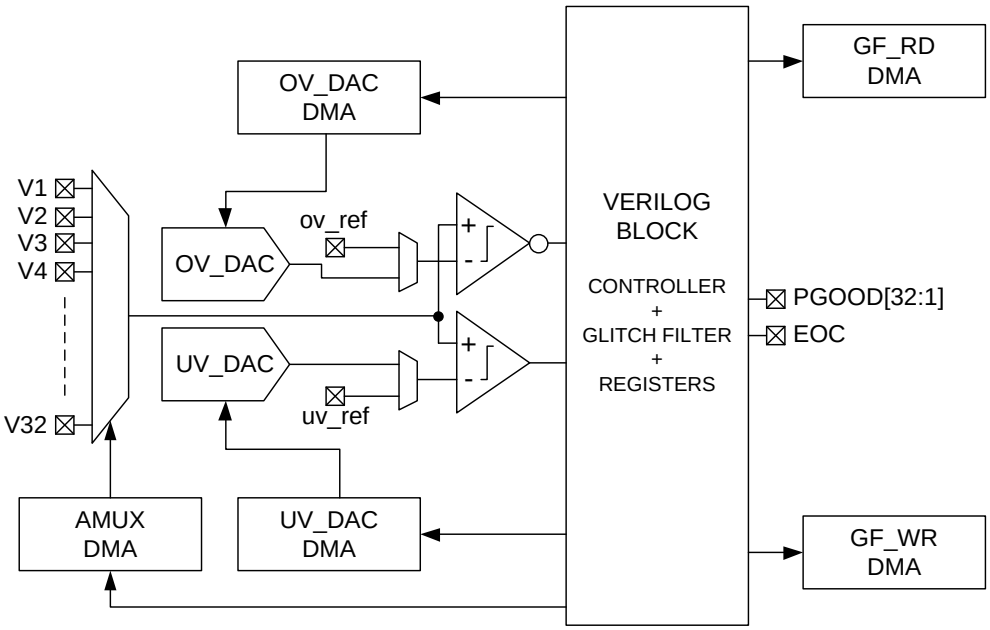
Configuration (number of voltages)	PSoC 3 (Keil_PK51)		PSoC 5LP (GCC)	
	Flash Bytes	SRAM Bytes	Flash Bytes	SRAM Bytes
8	3053	60	2892	57
16	3113	100	2980	97
24	3173	140	3052	137
32	3232	180	3160	177

Functional Description

Block Diagram and Configuration

Figure 1 shows an example representation of the Voltage Fault Detector implementation supporting both OV and UV fault detection with VDACs used to set fault detection thresholds.

Figure 1. Voltage Fault Detector Block Diagram



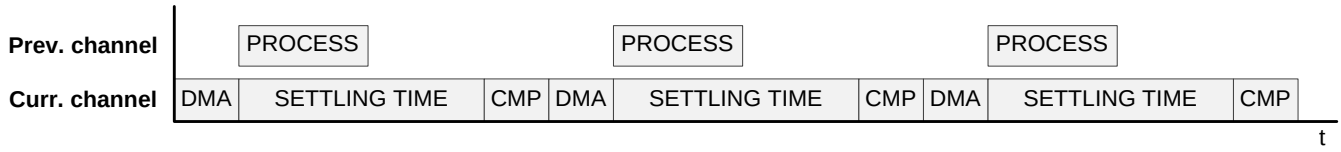
The DMA is used to set the DACs to the particular fault threshold and to connect the next analog input voltage to the window comparator. A state machine triggers the DMA and samples the comparator outputs at the desired time. Likewise, the DMA is used to read the previous context of the glitch filter for the current input from SRAM and store the updated glitch filter result back to SRAM.

The “Glitch Filter” is a counter with a programmable period (the Glitch Filter length). The filtering is applied to both rising and falling edge transitions of comparator outputs. Its main purpose is to prevent temporary input voltage glitches to be reflected on pgood output. It gets reset to zero anytime the current input sample and associated pgood output are the same and increments when they differ. Once it hits the user-specified glitch filter length, the glitch filter output propagates the current comparators state.

The Verilog block implements a state machine controller, glitch filter and status registers that retain the OV/UV/PGOOD status of each input voltage.

The processing is pipelined to ensure maximum possible performance. That is, during the time that the VDACs are settling to their new OV/UV limits, the post processing of the previous input is performed. This is graphically shown in Figure 2.

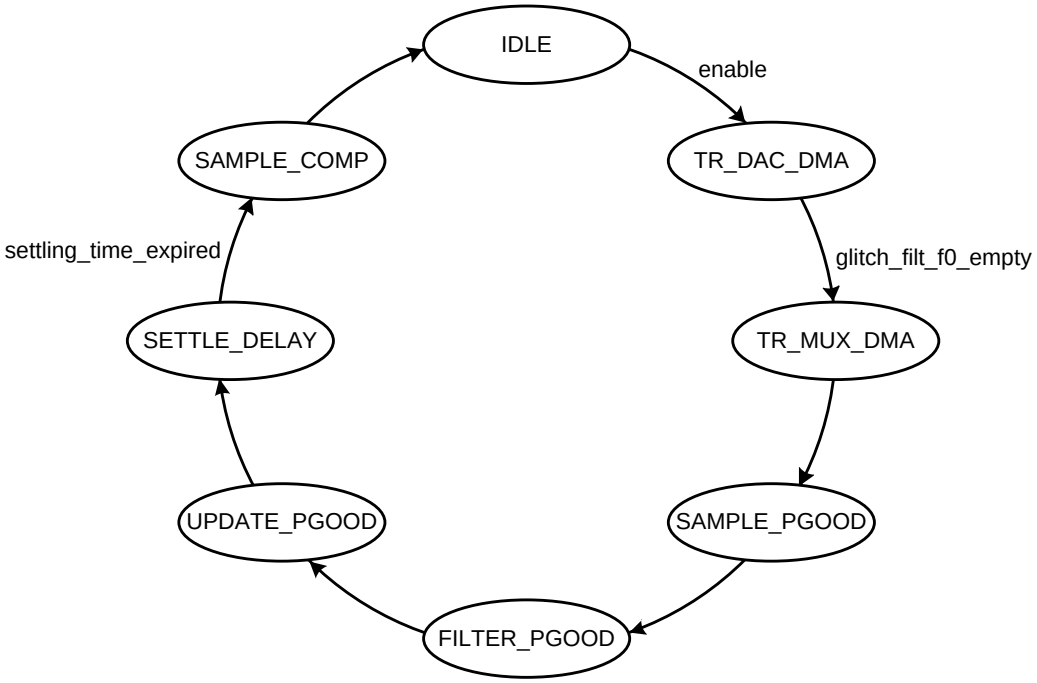
Figure 2. Sampling and Processing Pipeline



When internal VDACs are used to set the OV/UV thresholds, you can select between the 0-1V range and the 0-4V range. The advantage of the 0-1V range is that the VDAC update rate is higher (1 MHz) compared to the 625 kHz of the 0-4V range. This translates to a faster fault detection time, critical in many applications. In either case, all voltages to be monitored need to be scaled such that they fall within the selected VDAC range in the extreme case. That is, the OV threshold for any given voltage must fall within the VDAC max limits. The general guideline is that all voltages should be scaled such that the nominal voltage is within 85% of the selected VDAC upper limit. This is assuming that the OV threshold is no more than 10% above nominal.

The state machine has 8 states as shown in Figure 3.

Figure 3. VFD FSM State Diagram



The operational flow is:

IDLE:	Wait for enable signal.
TR_DAC_DMA:	Trigger DMA to move DACs to fault detection thresholds associated with next input channel.
TR_MUX_DMA:	Trigger DMA to connect next analog input voltage to the window comparator.
SAMPLE_PGOOD:	Connect sampled comparators from the previous input along with corresponding pgood state to glitch filter.
FILTER_PGOOD:	Glitch filtering. If the comparators outputs match the pgood state, reset glitch filter counter to zero. If differ, increment count. If count value is equal to user specified glitch filter length, reload the counter and assign new value to glitch filter output.
UPDATE_PGOOD:	Update pgood output and trigger DMA to store glitch filter count for the previous channel in SRAM. Trigger DMA to load glitch filter count for the next input channel from SRAM.
SETTLE_DELAY:	Wait for the input signal connected to the comparators to settle down before the comparators outputs are sampled. On receiving DMA completion status from MUX DMA, the state machine starts settling delay counter and waits until it expires.
SAMPLE_COMP:	The comparator outputs for the current input channel are sampled.

Clock Selection

Refer to clock input description of Input/Output Connections section for more information.

Placement

The choice of analog routing channel to use is set to AMUXBUSR by default, but the user can override this to maximize routing efficiency.

Industry Standards

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 VFD component has the following specific deviations:

Rule	Rule Class	Rule Description	Description of Deviation(s)
10.1	R	The value of an expression of integer type shall not be implicitly converted to a different underlying type under some circumstances.	Applies for PSoC 3 only. A value of unsigned integer is implicitly converted to integer type when passed as a 'len' parameter to memcpy() c51 library routine. There is no unintended side effect.
18.4	R	Unions shall not be used.	Deviated for constructing an efficient implementation.
19.12	R	There shall be at most one occurrence of the # or ## preprocessor operators in a single macro definition.	Deviated to simplify the code and provide more compact representation of similar functionalities.
19.13	A	The # and ## preprocessor operators should not be used.	
19.7	A	A function should be used in preference to a function-like macro.	Deviated for more efficient code.

This component has the following embedded components: AMux, DMA, VDAC8, Comp and Clock. Refer to the corresponding component datasheets for information on their MISRA compliance and specific deviations.

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 Specifications

The VFD component's DC characteristics are related to the VDAC and Comparator DC characteristics. Refer to the appropriate datasheet for more information.

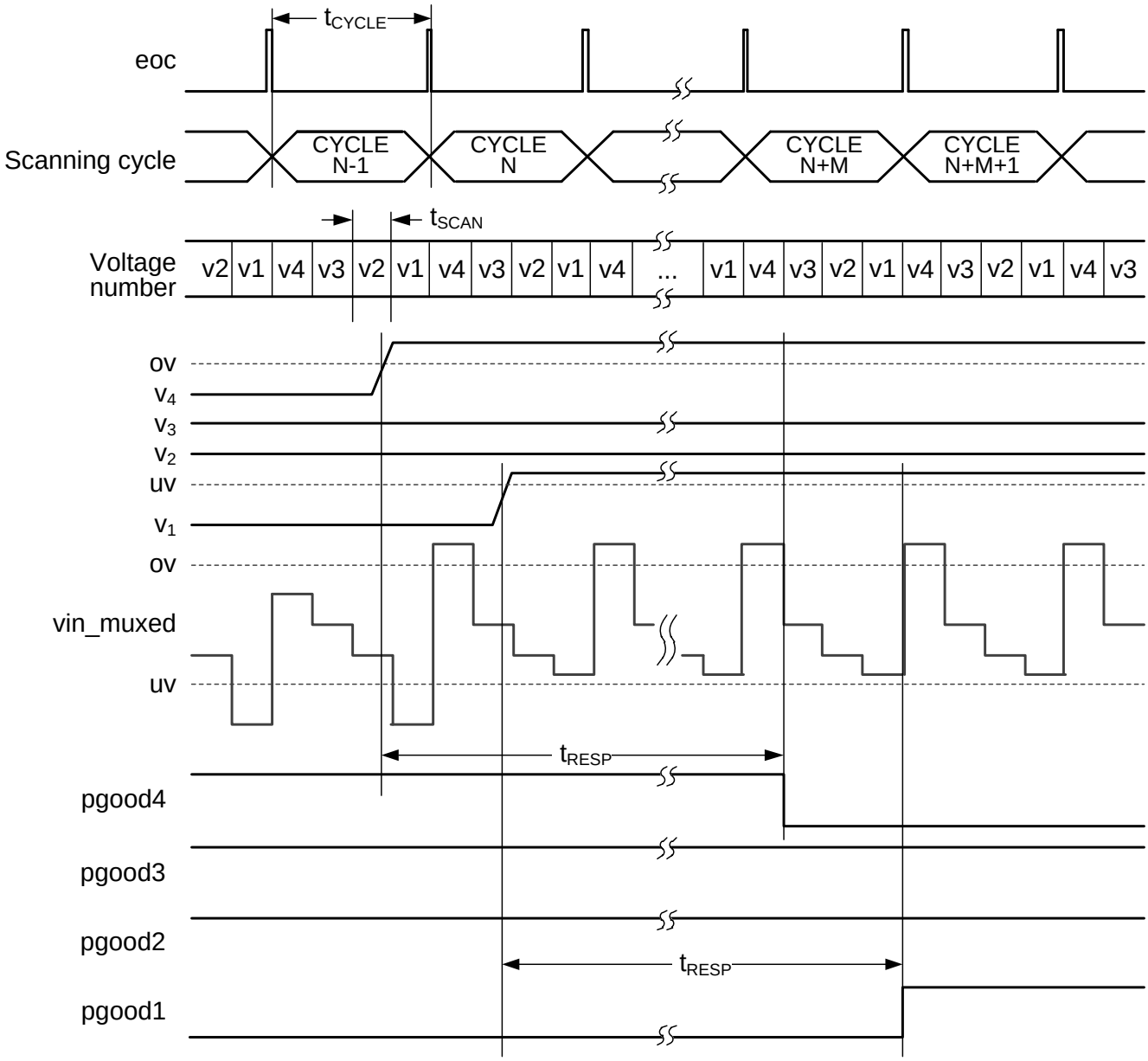
AC Specifications

Parameter	Description		Min	Typ	Max	Units
f_{CLOCK}	Operating frequency	Internal DAC, 1V range	–	–	8	MHz
		Internal DAC, 4V range	–	–	5	MHz
		External reference	–	–	8	MHz

Parameter	Description		Min	Typ	Max	Units
f_{BUS_CLK}	BUS_CLK frequency [3]		$5 \times f_{CLOCK}$	–	–	MHz
t_{SETTLE}	Delay time for input voltage to settle		3	–	128	$1/f_{CLOCK}$
t_{SCAN}	Scanning time per voltage		$5 + t_{SETTLE}$	–	–	$1/f_{CLOCK}$
t_{CYCLE}	Scanning cycle time for X voltages		–	X	–	t_{SCAN}
t_{RESP}	Response time [4]		$(L-1) \times X+1$	–	$L \times X$	t_{SCAN}
t_{CMP_RESP}	Comparator response time [5]	Fast speed	–	75	110	ns
		Slow speed	–	155	200	ns

3. Since DMA is involved and needs to run to completion within the time window dictated by the scanning time, this component inherently dictates a minimum BUS_CLK frequency. The minimum BUS_CLK:clock ratio for this component is 5:1.
4. L - filter length; X – number of voltages.
5. Taken from the comparator datasheet.

Figure 4. Operation Timing Diagram (Common OV/UV; Number of voltages = 4)



Component Changes

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

Version	Description of Changes	Reason for Changes / Impact
3.0.b	Minor datasheet edits.	
3.0.a	Enabled component to be nested into other components.	Support for hierarchical component design.
3.0	Corrected erratum ID 152757 (The control logic responsible for selecting input voltages and configuring the OV and UV thresholds can become out of synchronization when the component is paused and then resumed.)	The component could become out of synchronization for both the usage of the VFD_Pause() and VFD_Resume() functions as well as changing the value provided on the EN terminal of the component.
	Corrected erratum ID 159310 (Fault status of the Voltage Fault Detector is determined by reading a PGOOD and Over Voltage (OV) status register. The bits in the OV status register are sticky and the APIs responsible for returning fault status incorrectly clear the sticky bits. As a result, this can cause an issue to manifest in one of two scenarios: If a UV fault occurs on a rail that was preceded by an OV fault, the GetOVUVFaultStatus() function may incorrectly report an OV fault and fail to report the UV fault. When using the GetOVUVFaultStatus() or GetOVFaultStatus() APIs, the fault data returned may contain stale fault data, indicating an OV fault, for a rail that did not experience that fault condition.)	UV fault condition could be incorrectly reported as OV fault condition.
	Corrected erratum ID 159749 (The General tab in the Voltage Fault Detector customizer displays a fault response time next to the Glitch Filter Length selection. The time calculated only includes the delay associated with the glitch filter and doesn't include the delay for the final scan that will detect the fault.)	The Fault Response Time did not account the delay for the final scan that detected the fault. Thus, the value displayed for the Fault Response Time was inaccurate.
	Improved timing model. Reduced scanning time per rail to 1us when internal DACs range is set to 1 V.	Improved timing model to increase multiplexing frequency. The input clock has to be set to 8x the desired multiplexing frequency. The maximum clock speed is 8MHz. Recommended BUS_CLK to clock ratio is 5:1 to achieve desired scanning time.
	Improved the glitch filter.	Applies the glitch filter in the rising and falling edge of the internal comparators output.
	Updated the Customizer.	Organize how the configuration setting, splitting in different groups. Add new columns to show the actual OV and UV threshold values. Added a new parameter to set the settling delay.

Version	Description of Changes	Reason for Changes / Impact
	Consolidated the OV/UV Glitch Filter Length APIs.	The Glitch Filter Length cannot be individually configured for OV/UV fault condition.
	Datasheets edits and corrections. Note The errata section was removed from this datasheet.	To reflect changes made in v3.0.
2.30	Corrected the component changes made in PSoC Creator 3.0 SP1.	Correction of the Component Errata item – Cypress ID 191257.
2.20.b	Corrected validation of the OV and UV fault threshold parameters in the Configure dialog.	The component allowed out of range OV and UV fault threshold values to be entered in the Configure dialog.
	Datasheet updates and corrections.	
2.20.a	Added Component Errata section	Document known problems in the component.
	Reduced maximum clock speed from 12 MHz to 4 MHz.	Usage in systems has shown that 4 MHz is the maximum reliable operating speed for the component clock.
2.20	Added vin_muxed output. Added VFD_SetSpeed() API.	Updated requirements for the component. Fixed number of DMA channels used in the implementation.
2.10	Component schematic is updated with the latest version of all components.	
	Added MISRA Compliance section.	The component was not verified for MISRA compliance.
2.0	Added support for PSoC 5LP devices. Added GetOVUVFaultStatus() API. Changed GetOVFaultStatus() and GetOVFaultStatus() API description. Updated screenshot of the General Tab of the component's customizer. Added description of the Speed Mode Parameter. Added 'AMUXBUS' option for "Analog Bus". Added the following global arrays to the 'Global Variables' section: VFD_initOVFaultThreshold[], VFD_initUVFaultThreshold[], VFD_VoltageScale[]. Added description of the Software Analog MUX functionality. Resource usage table is updated. Minor datasheet edits.	GetOVFaultStatus() and GetOVFaultStatus() API are incompatible with v1_0.
1.0	First release