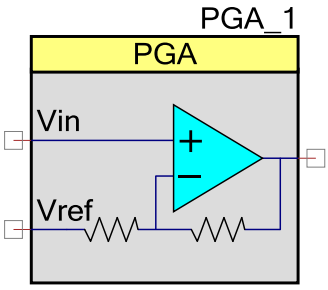
PSoC 4 Programmable Gain Amplifier (PGA_P4)

1.0



Features

- Eleven programmable gains between 1 and 32 in 3 dB steps
- Gain and power levels may be changed during runtime
- The internal feedback resistors saves up to two GPIOs
- Buffer high impedance sensor signals
- Amplify small signals to match ADC range
- Ability to operate in deep sleep low power mode

General Description

The PGA_P4 implements an opamp-based, non-inverting amplifier with user-programmable gain. The amplifier has high input impedance, wide bandwidth, and selectable input voltage reference. The gain can be between 1 (0 dB) and 32 (+30 dB) and can be configured at build time or during run time in firmware.

Two output modes (Internal only and Output to pin) are provided to drive internal or external signals, respectively. The Output to pin mode may drive both an external pin and an internal block such as the SAR ADC. You also have control of different power levels that provide a tradeoff between power and bandwidth.

When to Use a PGA_P4

Use the PGA_P4 when a signal needs to be buffered or amplified prior to being fed into the SAR ADC or another stage of the circuit.

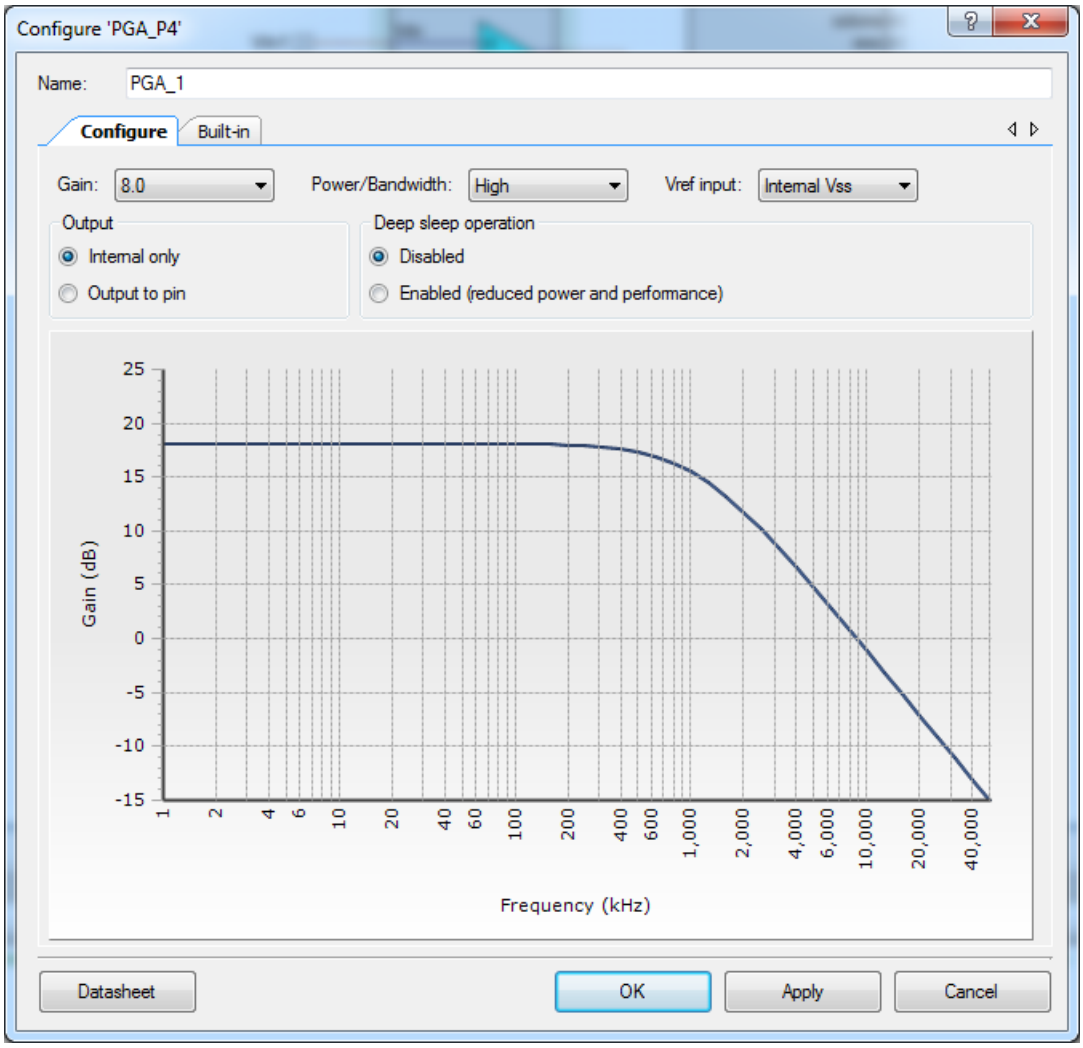
Input/Output Connections

This section describes the various input and output connections for the PGA_P4 component. An asterisk (*) in the following list indicates that it may not be shown on the component symbol for the conditions listed in the description of that I/O.

Terminal Name	I/O Type	Description
Vin	Analog Input	Voltage input signal terminal.
Vout	Analog Output	Vout is the output voltage signal terminal. Vout is a function of (Vin – Vref) times the specified Gain: $Vout = Vref + (Vin – Vref) \times Gain$
Vref*	Analog Input	Vref is the input terminal for a reference signal. The reference input can be connected to an external (to the component) reference or internal (to the component) VSS (ground). When the reference is connected externally, the routing resistance is added to the internal resistors, slightly decreasing the gain and increasing the gain tolerance.

Component Parameters

Drag a PGA_P4 component onto your design and double-click it to open the **Configure** dialog.



Parameter Name	Description
Gain	Selects default voltage gain. Gain selections are in +3dB steps. You can select the gain from the following set of allowed values: 1.0(default), 1.4, 2.0, 2.8, 4.0, 5.8, 8.0, 10.7, 16.0, 21.3, 32.0. The frequency response graph will reflect this setting.
Power / Bandwidth	Selects the power/bandwidth tradeoff. You can select following values: Low, Med, High (default). The frequency response graph will reflect this setting.

Parameter Name	Description
Vref Input	Selects how the Vref input is connected. Possible values: Internal Vss (default), External An external reference can only be driven by an opamp, PGA, or dedicated GPIO. That is, it is an external signal to the part. The component symbol will reflect this setting.
Output	Selects PGA output drive capable of either internal only connections or directly to the designated GPIO. Possible values: Internal only (default), Output to pin. The frequency response graph will reflect this setting.
Deep sleep operation (DSOp)	Enables or disables deep sleep operation (with reduced power and performance) Values: Disabled (default), Enabled. The frequency response graph will reflect this setting. When the “Output” option is set to “Internal only”, the Deep sleep operation will be disabled. **Note** If Deep sleep operation is enabled, then the power and performance is reduced during both active and deep sleep modes. It avoids any transient side effects and makes the performance stable during the whole lifetime.

Application Programming Interface

Application Programming Interface (API) routines allow you to configure the component using software. By default, PSoC Creator assigns the instance name **PGA_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 **PGA** .

General API

Description

General API functions are used for run-time configuration of the component during active power mode.

These include initializing, starting, stopping, reading from registers and writing to registers.

Functions

- void PGA_Init(void)
- void PGA_Enable(void)
- void PGA_Start(void)
- void PGA_Stop(void)
- void PGA_SetPower(uint32 powerLevel)
- void PGA_SetGain(uint32 gainLevel)
- void PGA_PumpControl(uint32 onOff)

Function Documentation

void PGA_Init (void)

Initializes component's parameters to the values set by user in the customizer of the component placed onto schematic. Usually it is called in PGA_Start().

void PGA_Enable (void)

Powers up amplifier (to default power level or restored after previous PGA_Stop() call). Usually it is called in PGA_Start().

void PGA_Start (void)

Enables the amplifier operation (calls PGA_Enable). Also on the first call (after the system reset) initializes all the component related registers with default values (calls PGA_Init).

Global Variables

The PGA_initVar variable is used to indicate initial configuration of this component. The variable is initialized to zero (0u) and set to one (1u) at the first time PGA_Start() is called. This allows to enable the component without re-initialization in all subsequent calls of the PGA_Start() routine.

void PGA_Stop (void)

Powers down the amplifier.

void PGA_SetPower (uint32 *powerLevel*)

Sets the power level of amplifier.

Parameters:

powerLevel	The power level setting of amplifier. Possible values:

	• PGA_LOW - The lowest power consumption. • PGA_MED - The middle setting. • PGA_HIGH - The highest amplifier bandwidth.

void PGA_SetGain (uint32 *gainLevel*)

Sets values of the input and feedback resistors to set a specific gain of the amplifier.

Parameters:

gainLevel	The gain setting of amplifier. Possible values: • PGA_GAIN_1 - gain 1.0. • PGA_GAIN_1_4 - gain 1.4. • PGA_GAIN_2 - gain 2.0. • PGA_GAIN_2_8 - gain 2.8. • PGA_GAIN_4 - gain 4.0. • PGA_GAIN_5_8 - gain 5.8. • PGA_GAIN_8 - gain 8.0. • PGA_GAIN_10_7 - gain 10.7. • PGA_GAIN_16 - gain 16.0 • PGA_GAIN_21_3 - gain 21.3. • PGA_GAIN_32 - gain 32.0.

void PGA_PumpControl (uint32 *onOff*)

Allows the user to turn the amplifier's boost pump on or off. By Default the PGA_Init() function turns the pump on.

Parameters:

onOff	The boost pump setting. Possible values: • PGA_BOOST_OFF - Turn off the pump. • PGA_BOOST_ON - Turn on the pump.

Low Power Functions

Description

Low Power Functions perform the necessary configurations to the component to prepare it for entering low power modes.

These API functions should be used if the "Deep sleep operation" is disabled and intent is to put the chip into low power mode, then to continue the component operation when it comes back to active power mode.

Functions

- void PGA_Sleep(void)
- void PGA_Wakeup(void)
- void PGA_SaveConfig(void)
- void PGA_RestoreConfig(void)

Function Documentation

void PGA_Sleep (void)

When the "Deep sleep operation" is disabled then the function disables component's operation and saves its configuration. Should be called just prior to entering sleep. When the "Deep sleep operation" is enabled then the function does nothing and the component continues to operate during Deep Sleep state.

void PGA_Wakeup (void)

When the "Deep sleep operation" is disabled then the function enables block's operation and restores its configuration. Should be called just after awaking from sleep. When the "Deep sleep operation" is enabled then the function does nothing and the component continues to operate during Active state.

void PGA_SaveConfig (void)

Empty function. Included for consistency with other components.

void PGA_RestoreConfig (void)

Empty function. Included for consistency with other components.

Global Variables

Description

Global variables used in the component.

The following global variables are used in the component.

Variables

- uint8 PGA_initVar

Variable Documentation

uint8 PGA_initVar

Describes the init state of the component

API Constants

Description

Component API functions are designed to work with pre-defined enumeration values.

These values should be used with the API functions that reference them.

Modules

- Group_powerLevel
  Definitions for PGA_SetPower() function.
- Group_boostPump
  Definitions for PGA_PumpControl() function.
- Group_gain

Definitions for PGA_SetGain() function.

Group_powerLevel

Description

Definitions for PGA_SetPower() function.

Macros

- #define **PGA_LOW** ((uint32)PGA__LOW)
- #define **PGA_MED** ((uint32)PGA__MED)
- #define **PGA_HIGH** ((uint32)PGA__HIGH)

Group_boostPump

Description

Definitions for PGA_PumpControl() function.

Macros

- #define **PGA_BOOST_ON** ((uint32)1u)
- #define **PGA_BOOST_OFF** ((uint32)0u)

Group_gain

Description

Definitions for PGA_SetGain() function.

Macros

- #define **PGA_GAIN_1** ((uint32)PGA__GAIN_1)
- #define **PGA_GAIN_1_4** ((uint32)PGA__GAIN_1_4)
- #define **PGA_GAIN_2** ((uint32)PGA__GAIN_2)
- #define **PGA_GAIN_2_8** ((uint32)PGA__GAIN_2_8)
- #define **PGA_GAIN_4** ((uint32)PGA__GAIN_4)
- #define **PGA_GAIN_5_8** ((uint32)PGA__GAIN_5_8)
- #define **PGA_GAIN_8** ((uint32)PGA__GAIN_8)
- #define **PGA_GAIN_10_7** ((uint32)PGA__GAIN_10_7)
- #define **PGA_GAIN_16** ((uint32)PGA__GAIN_16)
- #define **PGA_GAIN_21_3** ((uint32)PGA__GAIN_21_3)
- #define **PGA_GAIN_32** ((uint32)PGA__GAIN_32)

Code Examples and Application Notes

This section lists the projects that demonstrate the use of the component.

Code Examples

PSoC Creator provides access to code examples in the Code Example 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 "Code Example" topic in the PSoC Creator Help for more information.

There are also numerous code examples that include schematics and example code available online at the Cypress Code Examples web page. Examples that use this component include:

- CE204024 : PgaAdjustableGain

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 an 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.

PSoC 4 (GCC)

Configuration	PSoC Analog Coprocessor	
	Flash Bytes	SRAM Bytes
Vref_input - External, Deep Sleep Operation - Enabled	428	8
Vref_input - Internal Vss, Deep Sleep Operation - Enabled	444	8
Vref_input – External, Deep Sleep Operation - Disabled	492	8
Vref_input – Internal Vss, Deep Sleep Operation - Disabled	508	8

Functional Description

The Programmable Gain Amplifier uses one of the amplifiers and resistor blocks found in the CTB (see Definitions) to create a user adjustable gain non-inverting amplifier. Two PGAs can be created from a single CTB block. You can find details about this block in the applicable device datasheet and TRM, available on the Cypress website. The gain is selected by adjusting two resistors, Ra and Rb (see Figure 2. PGA Schematic). Actually Ra and Rb are parts of the resistor ladder with total resistance of 192 kOhms and multiple tap points (connection point between Ra and Rb on schematic) to select different gain values.

The block has a programmable compensation capacitor in parallel with the feedback resistor, Rb. The value of the capacitor is configured automatically (by the component API code) for each gain and power level to achieve guaranteed stability. Reassigning resistor values (directly in registers) without selecting the appropriate feedback capacitor value can result in PGA instability or reduced performance (response bandwidth). Cypress strongly recommends that you use the provided APIs for gain and power changes.

The bandwidth of the PGA is determined by Gain, Power and Output setting. Because of compensation capacitor and stability requirements, the bandwidth may be somewhat reduced from the absolute maximum expected from the opamp's open loop gain-bandwidth.

The input and output voltages and current ranges are limited (as well as all other opamps; see DC and AC Electrical Characteristics section for details). Exceeding the valid ranges may cause a signal distortion in both voltage and time domains.

The CTB pump does not have an internal oscillator and needs (if it is turned on) the system clock with frequency of 24 MHz.

Definitions

- **PGA** – Programmable Gain Amplifier
- **CTB** – Continuous Time Block
- **DSOp** – Deep sleep operation

Block Diagram and Configuration

Figure 1. PGA Configurations

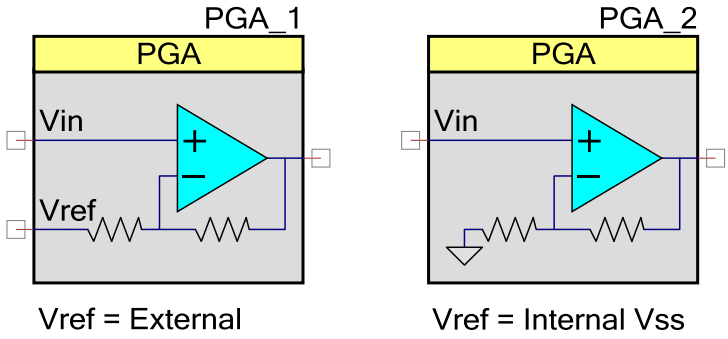
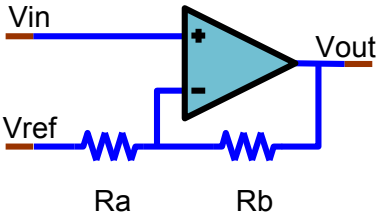


Figure 2. PGA Schematic



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 PGA_P4 component does not have any specific deviations.

Component Debug Window

PSoC Creator allows you to view debug information about components in your design. Each component window lists the memory and registers for the instance. For detailed hardware registers descriptions, refer to the appropriate device technical reference manual.

To open the Component Debug window:

1. Make sure the debugger is running or in break mode.
2. Choose **Windows > Components…** from the **Debug** menu.
3. In the Component Window Selector dialog, select the component instances to view and click **OK**.

The selected Component Debug window(s) will open within the debugger framework. Refer to the "Component Debug Window" topic in the PSoC Creator Help for more information.

Resources

The PGA_P4 uses the following device resources:

- One half of the CTB:
  - One CTB Operational Amplifier
  - One CTB resistor ladder

DC and AC Electrical Characteristics

The following values are based on characterization data. Specifications are valid for –40 °C ≤ T_A ≤ 85 °C and T_J ≤ 100 °C except where noted. Unless otherwise specified in the following tables, all Typical values are for T_A = 25 °C, V_{DDA} = 5.0 V, Power = High, output referenced to analog ground, V_{SSA}.

Note Final characterization data for PSoC 4000S, PSoC 4100S and PSoC Analog Coprocessor devices is not available at this time. Once the data is available, the component datasheet will be updated on the Cypress web site.

DC Electrical Characteristics

Parameter	Description	Conditions	Min	Typ	Max	Units
Vin	Input Voltage Range	Pump is turned on	Vssa		Vdda-0.2, <5	V
		Pump is turned off	Vssa		Vdda-1.5	
Vos	Input offset voltage				1	mV

Parameter	Description	Conditions	Min	Typ	Max	Units
TCVos	Temperature coefficient, input offset voltage	Power mode = High, Gain = 1		TBD	TBD	uV/C
IDD_Hi	Power = High			TBD	TBD	uA
IDD_Med	Power = Medium			TBD	TBD	uA
IDD_Low	Power = Low			TBD	TBD	uA
Iout_Max_High	Power = High	Supply >= 2.7 volts			10	mA
Iout_Max_Med	Power = Medium	Supply >= 2.7 volts			10	mA
Iout_Max_Low	Power = Low	Supply >= 2.7 volts			4	mA
Iout_Max_High	Power = High	Supply < 2.7 volts			TBD	mA
Iout_Max_Med	Power = Medium	Supply < 2.7 volts			TBD	mA
Iout_Max_Low	Power = Low	Supply < 2.7 volts			TBD	mA
VOUT_1	Power = High, load = 10mA		0.5		Vdda-0.5	V
VOUT_2	Power = High, load = 1mA		0.2		Vdda-0.2	V
VOUT_3	Power = Med, load = 1mA		0.2		Vdda-0.2	V
VOUT_4	Power = Low, load = 0.1mA		0.2		Vdda-0.2	V
VOS	Offset voltage, Power = High		-	+/-0.5	1	mV
VOS	Offset voltage, Power = Medium		-	+/- 1	-	mV
VOS	Offset voltage, Power = Low		-	+/- 2	-	mV
VOS_DR	Offset voltage drift Power = High		-10	+/-3	10	uV/C
VOS_DR	Offset voltage drift Power = Medium		-	+/-10	-	uV/C
VOS_DR	Offset voltage drift Power = Medium		-	+/-10	-	uV/C
PERF.TOTAL_RES		Total resistor string resistance		192K		Ohms
Gain Error						
Ge1	Gain accuracy, Gain = 1	Just Vos	-		0	+/-%
Ge1.4	Gain accuracy, Gain = 1.4		-		0.88	+/-%
Ge2	Gain accuracy, Gain = 2		-		0.93	+/-%
Ge2.8	Gain accuracy, Gain = 2.8		-		1.09	+/-%
Ge4	Gain accuracy, Gain = 4		-		1.36	+/-%
Ge5.8	Gain accuracy, Gain = 5.8		-		1.78	+/-%

Parameter	Description	Conditions	Min	Typ	Max	Units
Ge8	Gain accuracy, Gain = 8		-		2.27	+/-%
Ge10.7	Gain accuracy, Gain 10.7		-		2.84	+/-%
Ge16	Gain accuracy, Gain = 16		-		3.97	+/-%
Ge21.3	Gain accuracy, Gain = 21.3		-		5.05	+/-%
Ge32	Gain accuracy, Gain = 32		-		7.15	+/-%
Deep Sleep mode	For Deep Sleep mode, only the external driver can be enabled, therefore all specs are for external driver.					
Vos_DR	Offset voltage drift			TBD	TBD	uV/C
Vout_DS	Output Voltage			TBD	TBD	V
Vcm_DS	Common Mode Voltage			TBD	TBD	V
IDD_HIGH	Current, Power = High			TBD	TBD	uA
IDD_MED	Current, Power = Med			TBD	TBD	uA
IDD_LOW	Current, Power = Low			TBD	TBD	uA
VOS_HIGH	Offset voltage, Power = High			TBD	TBD	mV
VOS_MED	Offset voltage, Power = Med			TBD	TBD	mV
VOS_LOW	Offset voltage, Power = Low			TBD	TBD	mV
IOUT_HIGH	Output current, Power = High, 0.5 V to VDDA-0.5 V			TBD	TBD	mA
IOUT_MED	Output current, Power = High, 0.5 V to VDDA-0.5 V			TBD	TBD	mA
IOUT_LOW	Output current, Power = High, 0.5 V to VDDA-0.5 V			TBD	TBD	mA

AC Electrical Characteristics

Parameter	Description	Conditions	Min	Typ	Max	Units
CMRR	DC, Vddd = 3.6V		70	80		dB
PSRR	At 1 kHz, 100 mV ripple, Vddd = 3.6V		70	85		dB
GBW_High	Power = High, Internal driver		8			MHz
GBW_Med	Power = Med, Internal driver		3			MHz
GBW_Low	Power = Low, Internal driver		1			MHz
Noise						
VN1	Input buffered, 1 Hz – GHz, Power = High			TBD	TBD	uVrms

Parameter	Description	Conditions	Min	Typ	Max	Units
VN2	Input buffered, 1 kHz, Power = High			TBD	TBD	uVrms
VN3	Input buffered, 1 0kHz , Power = High			TBD	TBD	uVrms
VN4	Input buffered, 100kHz, Power = High			TBD	TBD	uVrms
CLOAD	Stable up to a maximum load. Performance specs at 50pF.				125	pF
Slew_rate	Cload = 50 pF, Power = High, VDDA ≥ 2.7 V	Gain <= 4	6			V/usec
T_op_wake	From disable to enable, no external RC dominating			TBD	TBD	usec
Deep Sleep mode	For Deep Sleep mode, only the external driver can be enabled, therefore all specs are for external driver.					
GBW_HIGH	GBW, Power = High	External driver, in Deep Sleep mode	1			MHz
GBW_MED	GBW, Power = Med			TBD	TBD	MHz
GBW_LOW	GBW, Power = Low			TBD	TBD	MHz

Component Changes

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

Version	Description of Changes	Reason for Changes / Impact
1.0.c	Minor datasheet edits	Specified Slew_rate conditions apply to gains of four or less.
1.0.b	Minor datasheet edits.	
1.0.a	Edited datasheet.	Final characterization data for PSoC 4000S, PSoC 4100S and PSoC Analog Coprocessor devices is not available at this time. Once the data is available, the component datasheet will be updated on the Cypress web site.
1.0	Initial version	New component

© Cypress Semiconductor Corporation, 2016-2018. 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.

PRELIMINARY