Preliminary

Gas Sensor Analog Front End Datasheet GasSensorAFE V 1.10

Copyright © 2012-2013 Cypress Semiconductor Corporation. All Rights Reserved.

This datasheet contains Preliminary information.

Resources	PSoC® Blocks			API Memory (Bytes)		Pins (per External I/O)
	Digital	Analog CT	Analog SC	Flash	RAM	
Supported Devices: CY8C29466, CY8C29566, CY8C29666, CY8C29866, CY8C28243, CY8C28433, CY8C28445, CY8C28452, CY8C28533, CY8C28545, CY8C28623, CY8C28643, CY8C28645, CY8C27143, CY8C27243, CY8C27443, CY8C27543, CY8C27643, CY8C24094, CY8C24794, CY8C24894, CY8C24994, CY8C24123A, CY8C24223A, CY8C24423A						
	0	2	1	427	0	4

Features and Overview

- Implements a three-lead electrochemical sensor with current output.
- Contains a trans-impedance amplifier (TIA) for current-to-voltage conversion.
- Maintains bias conditions needed by the cell
- Can be used to build a single-chip gas sensing solution in combination with appropriate ADC, SLCD, and CapSense User Modules

The Gas Sensor Analog Front End (AFE) User Module contains a bias circuit to maintain the sensor at a specific state and a TIA (current-to-voltage converter) to translate the sensor current, which is proportional to gas concentration, into a voltage that can be digitized.

Figure 1. GasSensor Hardware Block Diagram (PSoC 1)

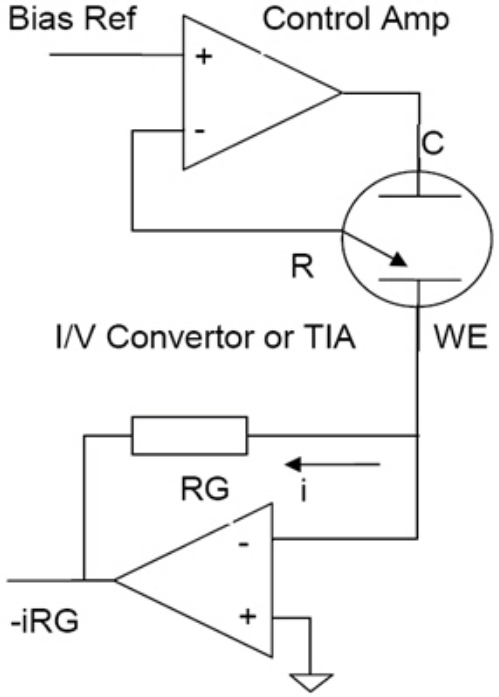
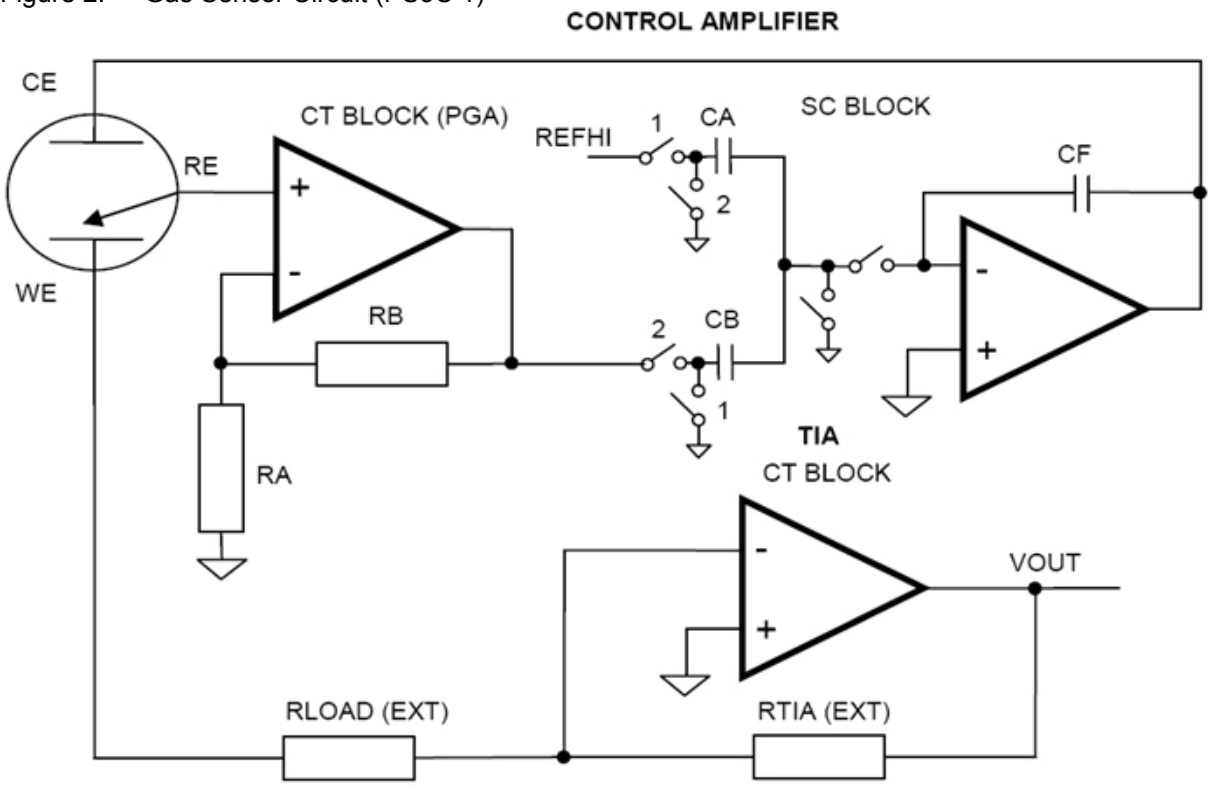


Figure 2. Gas Sensor Circuit (PSoC 1)



Functional Description

The Gas Sensor AFE User Module is made up of two main sections as shown in Figure 1 and Figure 2:

- A control amplifier
- A trans-impedance amplifier

The gain resistor for the TIA (RTIA) and load resistor for the sensor (RLOAD) are external components.

Control Amplifier

The control amplifier maintains the sensor's reference electrode (RE) at the Bias Reference Voltage using one analog continuous time block (CT BLOCK (PGA)) and one analog switched capacitor block (SC BLOCK).

The voltage at RE is buffered via the CT BLOCK (PGA) and fed to the CB arm of the SC BLOCK. REFHI drives the CA arm of the SC BLOCK. The opamp in the SC BLOCK servos its output to maintain its inverting input at the same level as the non-inverting input, which is at AGND. Therefore, the node at which CA and CB are connected is also maintained at AGND. Since the switch phases of CA and CB are opposite, the voltages across CA and CB subtract.

$$(REFHI \times CA) - (PGA_{OUT} \times CB) = 0$$

Solving for PGA_OUT we have,

$$PGA_{OUT} = \frac{REFHI \times CA}{CB}$$

We also know,

$$PGA_{OUT} = VRE \times PGA_{GAIN}$$

Substituting Equation 3 for PGA_OUT in Equation 2 and solving for VRE we get,

$$VRE = \frac{REFHI \times CA}{CB \times PGA_{GAIN}}$$

TIA

The TIA maintains the working electrode, WE, at ground potential through the resistor RG, and provides a voltage output proportional to the current through WE.

DC and AC Electrical Characteristics

Table 1. 5.0V PGA AC Electrical Characteristics

Parameter	Description	Min	Typ	Max	Units	Conditions and Notes
I_{RE}	Input leakage current through the RE pin	–	–	1	nA	
I_{CE}	Minimum sink/ source current through the CE pin	1	–	–	mA	
$V_{OS\ RW}$	Offset voltage between WE and RE pin	–	–	10	mV	
V_{RW}	RE-WE bias voltage error	–	–	2	% of FS	

Placement

The Gas Sensor AFE User Module uses three analog PSoC blocks, two continuous time (CT), and one switched capacitor (SC). The Gas Sensor AFE also uses two output buffers and the band gap reference generator.

Parameters and Resources

Bias Voltage

This parameter specifies the voltage, in millivolts, that the control amplifier maintains between the RE and WE terminals.

TIA_Output

The TIAMP block output can be routed through an analog output bus. Setting the TIA_Output parameter to Disable, the default value, restricts the set of possible connections to the local network. If TIA_Output is enabled on to the analog bus, make sure that no other user module drives this same bus.

SCA_Output

The SCAMP block output can be routed through an analog output bus. Setting the SCA _Output parameter to Disable, the default value, restricts the set of possible connections to the local network. If SCA _Output is enabled on to the analog bus, make sure that no other user module drives this same bus.

Application Programming Interface

The Application Programming Interface (API) routines are provided as part of the user module to allow the designer to deal with the module at a higher level. This section specifies the interface to each function together with related constants provided by the “include" files.

Note

In this, as in all user module APIs, the values of the A and X register may be altered by calling an API function. It is the responsibility of the calling function to preserve the values of A and X prior to the call if those values are required after the call. This “registers are volatile" policy was selected for efficiency reasons and has been in force since version 1.0 of PSoC Designer. The C compiler automatically takes care of this requirement. Assembly language programmers must ensure their code observes the policy,

too. Though some user module API function may leave A and X unchanged, there is no guarantee they will do so in the future.

For Large Memory Model devices, it is also the caller's responsibility to preserve any value in the CUR_PP, IDX_PP, MVR_PP, and MVW_PP registers. Even though some of these registers may not be modified now, there is no guarantee that will remain the case in future releases.

GasSensorAFE_Start

Description:

Configures the Analog CT and SC blocks to function as a Gas Sensor AFE.

C Prototype:

```
void  GasSensorAFE_Start(BYTE bPowerSetting)
```

Assembly:

```
lcall  GasSensorAFE_Start
```

Parameters:

bPowerSetting: Gas AFE operating power.

Symbolic Name	Value
GasSensorAFE_OFF	0
GasSensorAFE_LOWPOWER	1
GasSensorAFE_MEDPOWER	2
GasSensorAFE_HIGHPOWER	3

Return Value:

None

GasSensorAFE_SetPower

Description:

Changes the power setting of the GasSensorAFE.

C Prototype:

```
void  GasSensorAFE_SetPower(BYTE bPowerSetting)
```

Assembly:

```
lcall  GasSensorAFE_SetPower
```

Parameters:

bPowerSetting: Gas AFE operating power.

Symbolic Name	Value
GasSensorAFE_OFF	0
GasSensorAFE_LOWPOWER	1
GasSensorAFE_MEDPOWER	2
GasSensorAFE_HIGHPOWER	3

Return Value:

None

GasSensorAFE_SetBias

Description:

Sets the bias voltage between the RE and WE pins.

C Prototype:

```
void  GasSensorAFE_SetBias (int iBias, int iExtRef)
```

Assembler:

```
lcall  GasSensorAFE_SetBias
```

Parameters:

iBias: Bias voltage in millivolts, 0–999.

iExtRef: The external reference voltage in millivolts, 0–2500, if external reference is enabled in the global parameters, otherwise this parameter is ignored.

Return Value:

None

GasSensorAFE_Stop

Description:

Stops the Gas Sensor AFE by turning off the power to the Analog CT and SC blocks.

C Prototype:

```
void  GasSensorAFE_Stop
```

Assembler:

```
lcall  GasSensorAFE_Stop
```

Parameters:

None

Return Value:

None

Sample Firmware Source Code

The sample code in C is as follows:

```
//-------------------------------------------------------------------------
// C main line
//-------------------------------------------------------------------------

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

void main(void)
{
    // Start the GasSensorAFE UM in the medium power mode
    GasSensorAFE_Start(GasSensorAFE_MEDPOWER);
    // Set the bias voltage between RE and WE to 300 mV and set the external reference
    //(P2[6]) to 1000 mV.
    // If reference is internal, iExtRef parameter is ignored.
    GasSensorAFE_SetBias(300, 1000);

    while (1);
}
```

The equivalent code in assembly is:

```
;-------------------------------------------------------------------------
; Assembly main line
;-------------------------------------------------------------------------
include "m8c.inc"       ; part specific constants and macros
include "memory.inc"    ; Constants & macros for SMM/LMM and Compiler
include "PSoCAPI.inc"   ; PSoC API definitions for all User Modules

export _main

_main:
    ;Start the GasSensorAFE UM in the medium power modes
    mov   A, GasSensorAFE_MEDPOWER
    call  GasSensorAFE_Start
    ;Set the bias voltage between RE and WE to 300(0x012Ch) mV and set the external
    ;reference (P2[6]) to 1000(0x03E8h) mV.
    ;If reference is internal, iExtRef parameter is ignored.
    mov   A, 0x03
    push  A
    mov   A, 0xE8
    push  A
    mov   A, 0x01
    push  A
    mov   A, 0x2C
    push  A
    call  GasSensorAFE_SetBias

.terminate:
    jmp .terminate
```

Version History

Version	Originator	Description
1.0	KUK	Initial version
1.10	MYKZ	1. Added code generation script to read Ref Mux Global parameter and initialize the SC block settings properly. 2. Moved UM to Legacy state as this is no longer recommended for new designs.

Note PSoC Designer 5.1 introduces a Version History in all user module datasheets. This section documents high level descriptions of the differences between the current and previous user module versions.

Copyright © 2012-2013 Cypress Semiconductor Corporation. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

PSoC Designer™ and Programmable System-on-Chip™ are trademarks and PSoC® is a registered trademark of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

Any Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.