Introduction

In this application note we will describe how to create an ADC of up to 16-bits in a GreenPAK5 device which has an I2C interface useful for connecting to MCU's. The ADC architecture uses minimal GreenPAK resources, yet allows for easy input range adjustment via resistors.

ADC Architecture

The ADC is essentially comprised of an analog comparator and a Digital-to-Analog Converter (DAC). The comparator senses the input voltage vs. the DAC output voltage, and subsequently controls whether to increment or decrement the DAC input code, such that the DAC output converges to the input voltage. The resulting DAC input code becomes the ADC digital output code.

In our implementation, we create a DAC using a PWM controlled resistor network. We can easily create a precise digitally controlled PWM output using GreenPAK. The PWM when filtered becomes our analog voltage and thus serves as an effective DAC. A distinct advantage of this approach is that it is easy to set the voltages which correspond zero code and full scale (equivalently offset and gain) by simply adjusting resistor values. For example, a user wants to read zero code from a temperature sensor with zero degrees Celsius corresponding to 0.5V, and full scale code at 100 degrees corresponding to 0.7V. This is easily implemented by simply setting a few resistor values. By having the ADC range match the sensor range of interest, we make greatest use of the ADC resolution.

A design consideration for this architecture is that an internal PWM frequency needs to be much faster than the ADC update rate to prevent underdamped behavior of its control loop. Thus, this architecture is more appropriate for relatively slow sensing applications such as temperature sensors.

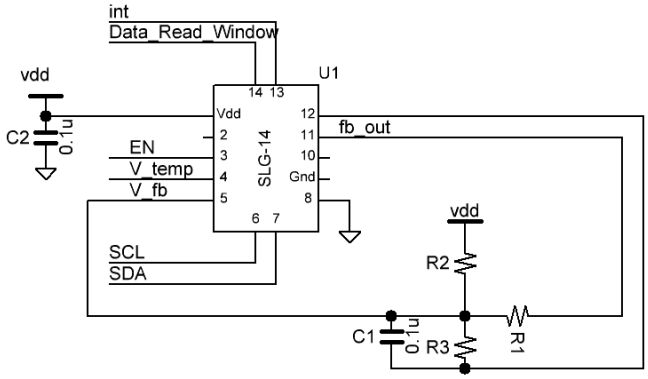


Figure 1. ADC Resistor Network

Implement: Resistor Network

An external resistor and capacitor network is used to convert a PWM into an analog voltage as shown in the circuit schematic in Figure 1. The values are calculated for maximum resolution where the DAC min and max voltage match the sensor's min and max voltage. To achieve this flexibility, we add resistors R2 and R3 in parallel to VDD and ground. Their values can be solved using the following equations.

$$V_{MAX} = V_{DD} * \frac{R3}{R3 + R2||R1}$$

$$V_{MIN} = V_{DD} * \frac{R3||R1}{R3||R1 + R2}$$

Note: the ACMP's negative input must not exceed 1.0V. GreenPAK has on-chip gain dividers that can be used to divide down the input voltage if needed.

Implement: ACMP

Configure the ACMP such that the positive input is a GPIO and the negative input is also a GPIO. The positive terminal shall connect to the sensor and the negative terminal to the resistor network.

The output of this ACMP controls the direction in which the PWM will increase or decrease. A HIGH output indicates the sensor is greater than the DAC. A LOW output indicates the sensor is less than the DAC.

Implement: PWM DAC

The PWM is generated internally by the GreenPAK. Its output is connected to R1. This PWM is unique because it is adjustable and constantly running.

First, take two counters of the same length and allow them to continuously run. Their outputs will become the Set and Reset signals to the PWM. The Set indicates when the PWM output goes HIGH. The Reset indicates when the PWM output goes LOW. Therefore, the relative timing between Set and Reset is equivalent to the PWM width.

To create the adjustable timing, the Set counter is adjusted by adding or subtracting clocks while the Reset counter is held constant. See Figure 2 for 3-bit LUT1 properties. When IN2 (Up/nDOWN) is logic 0, the output of this block will skip one clock. When IN2 is logic 1, the output of this block will add one clock.

Adding a clock shifts the Set earlier relative to Reset, increasing the PWM width. Skipping a clock would shift the Set later relative to the reset, decreasing the PWM width. As stated before, the choice to add or skip is controlled by the ACMP output.

The Set output is inverted and connected to the nReset of DFF7. The Reset output is connected to the CLK of DFF7. DFF7 output is also inverted. When Set goes Low, the PWM signal goes HIGH. When Reset goes High, the PWM signal goes LOW. Figure 3 is a timing diagram showing how the PWM signal is toggled.

Implement: Update Period

Previously, we mentioned that the internal loop must be much slower than the external loop for best stability. Here it is implemented by CNT3, which counts 64 Reset signals before its output is high. Therefore, the Update rate is 64 times slower than the PWM frequency.

This signal is also used to update the ADC data, which is discussed next.

IN2	IN1	IN0	OUT
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

Figure 2. 3-bit LUT1 properties

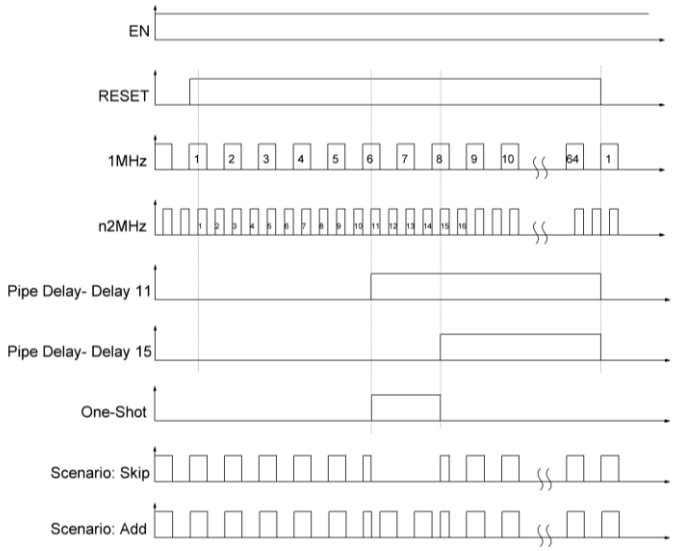


Figure 3. Timing Diagram of Clock Add and Clock Skip.

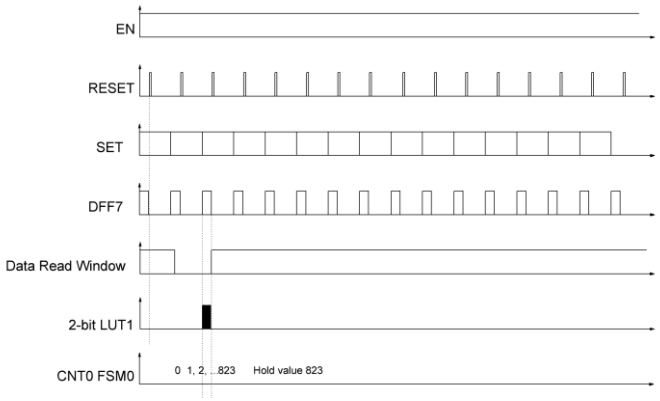


Figure 4. Timing Diagram of PWM Signal and Data_Read_Window

Implement: I2C Readability

In the GreenPAK5, we can use I2C to read counter data. This only applies to counters CNT0, CNT1, CNT5 and CNT6. Here we use the 16-bit CNT0 to store the PWM width. At the rising edge of Update, CNT0 begins filling up using the same clock source, only this time it is ANDed with the PWM output. Thus, CNT0 holds the digital representation of the PWM.

In order to read the data in CNT0, access location 0xEC and 0xEB which represent bits [15:8] and [7:0]. It is important to remember that this process is done once every Update cycle and data is incorrect during the loading processes. Data_Read_Window is an output signal, connected to nUpdate, which can be used as an indicator for when the counter has valid data.

Variants

The basic design is ratiometric with the power supply. This works great for resistive bridge sensors for example.

In non-ratiometric applications, we can use an external LDO, or, we can use a dual-rail GreenPAK such as the SLG46538 to implement an internal LDO. Dual Supply devices can be configured such that the second supply is a buck of the first supply, thereby creating a simple LDO. Note the output drive of such an LDO is limited by the GPIO output strength.

Another variant we can implement is a 2nd ADC channel on the same chip. This can be accomplished by reusing the Reset counter for both channels. However due to not having enough 16-bit counters, the two-channel version can only have up to 8-bits resolution. We have provided an example design where the two channel ADC's counter data is stored at 0xEB and 0xEE.

For cases where we want to handle inputs exceeding the ADC range, we'd need to implement roll-over protection circuits. For example, when we reach 0% or 100% and we want to prevent the ADC code from rolling

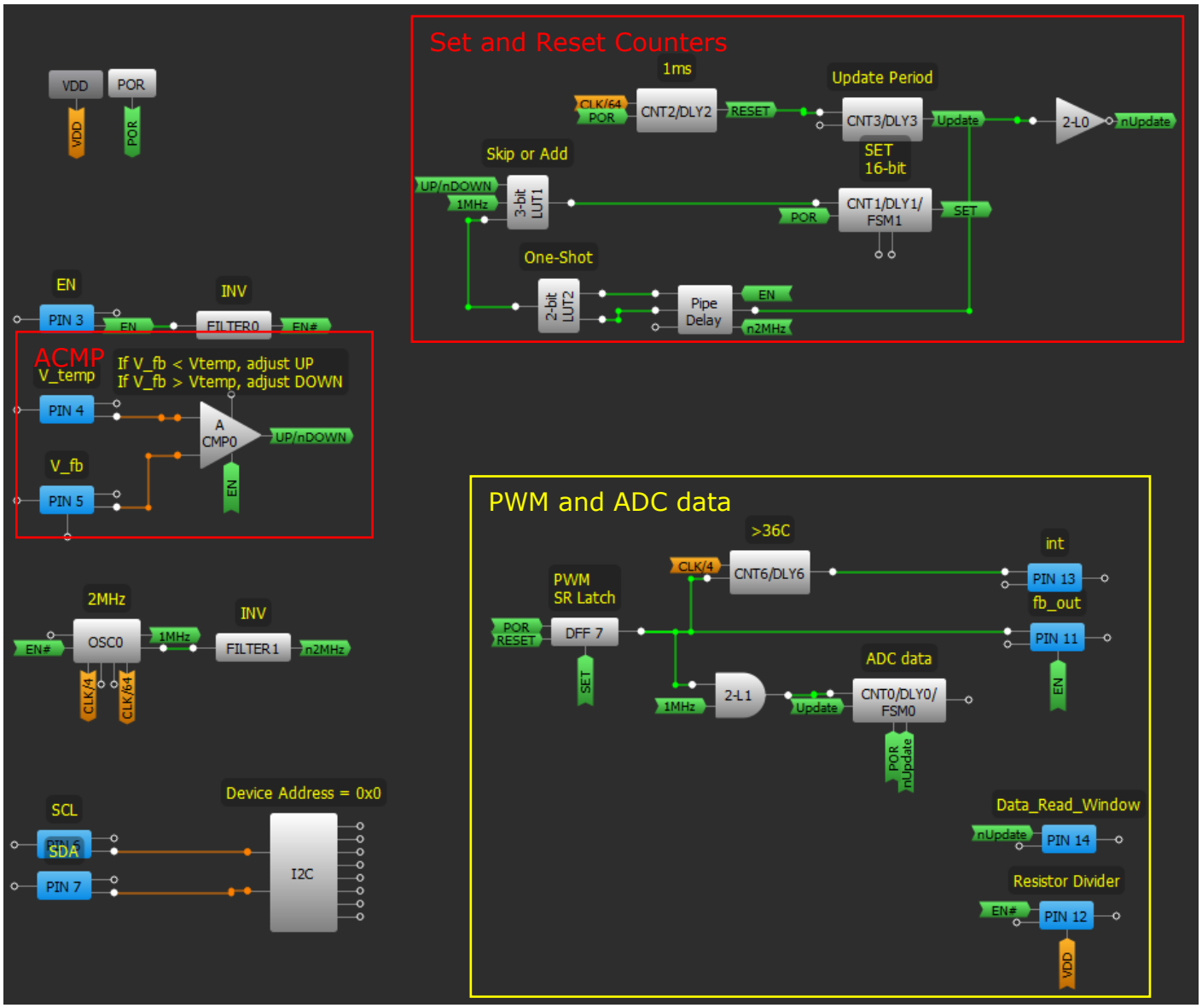
over. Additional counters can be used to set the minimum PWM and maximum PWM. Alternatively, the resistor dividers can be set such that the sensor output will never reach Vmin and Vmax.

Conclusion

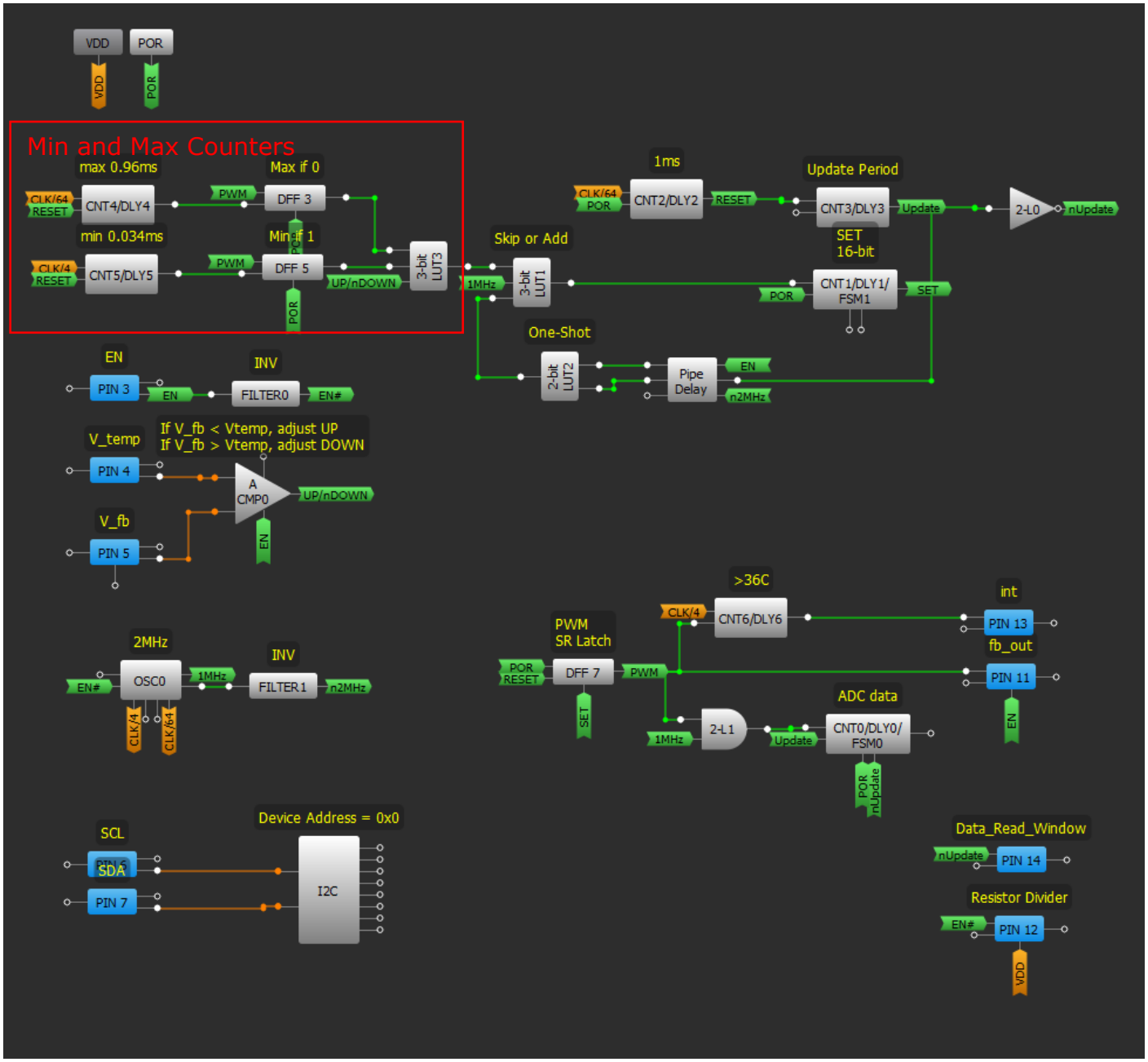
Based on the results, we were able to achieve a maximum 16-bit ADC with +/- 1 or 2 LSB of jitter. The design example is set to 10-bits for faster convergence. A 16-bit ADC using this design is quite slow, as the PWM DAC updates by linear steps, and thus would only be used with a slow-moving signal. The bit width is easily adjusted by adjusting the PWM counters.

Appendix-A

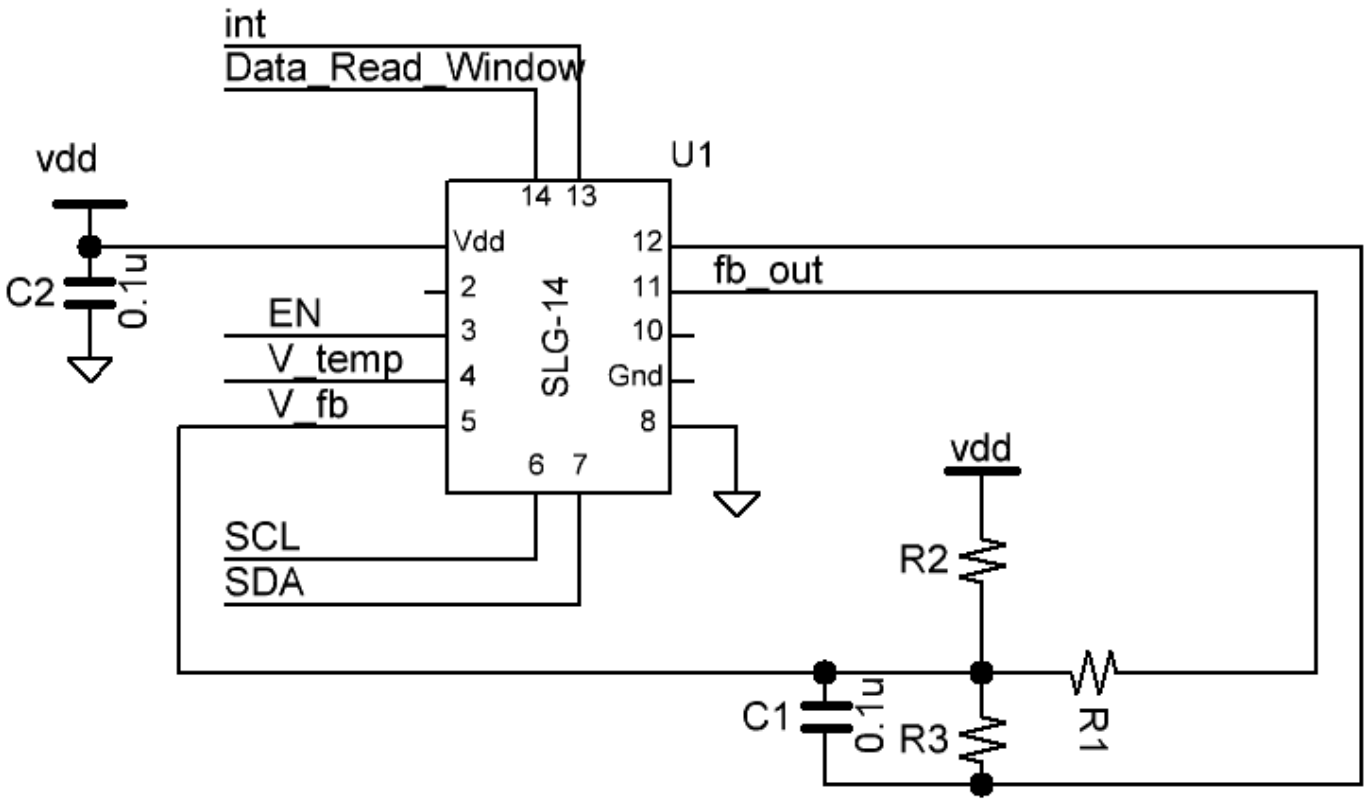
1-channel ADC in a SLG46536



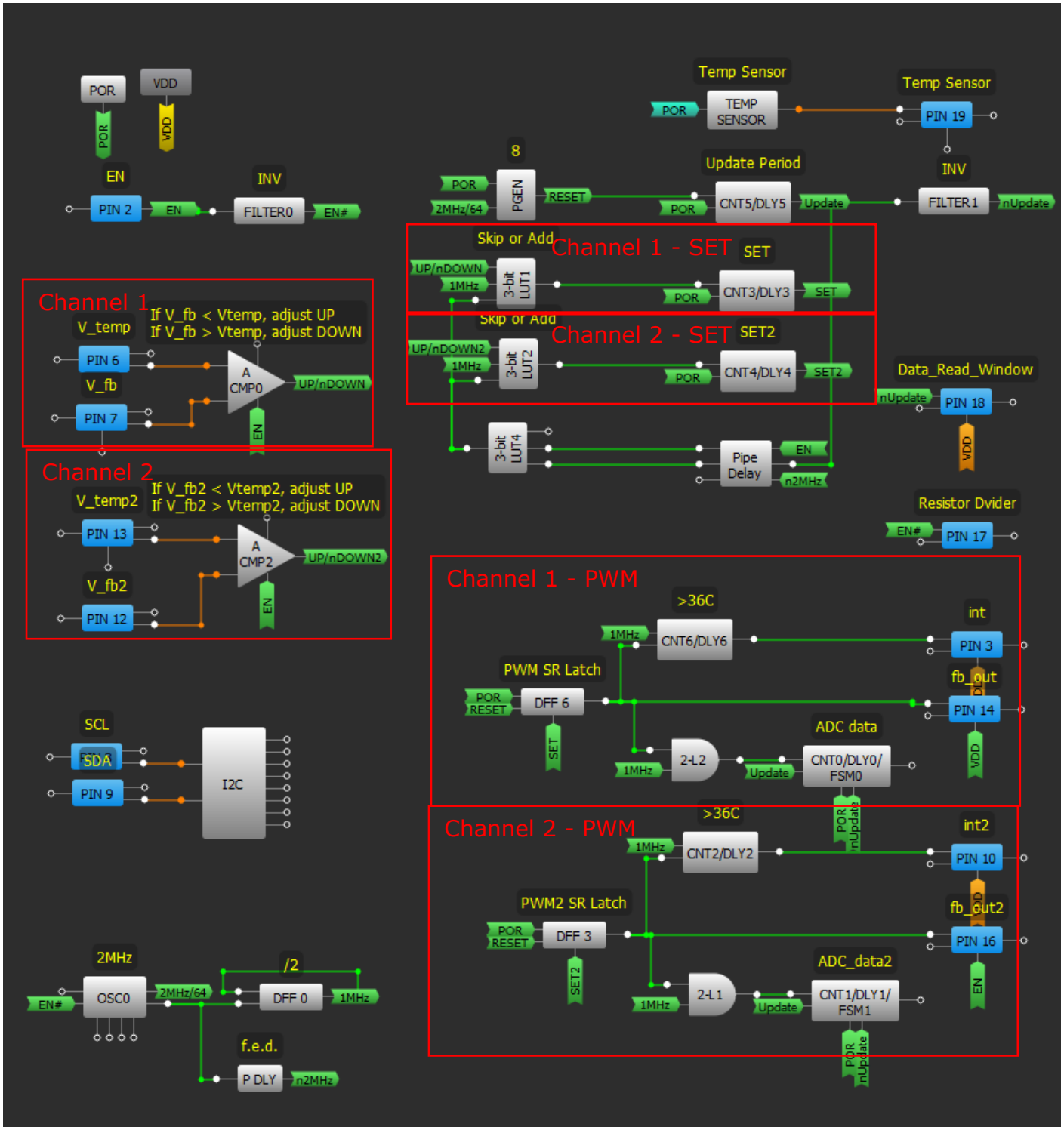
1-channel ADC with min/max limiters in SLG46536



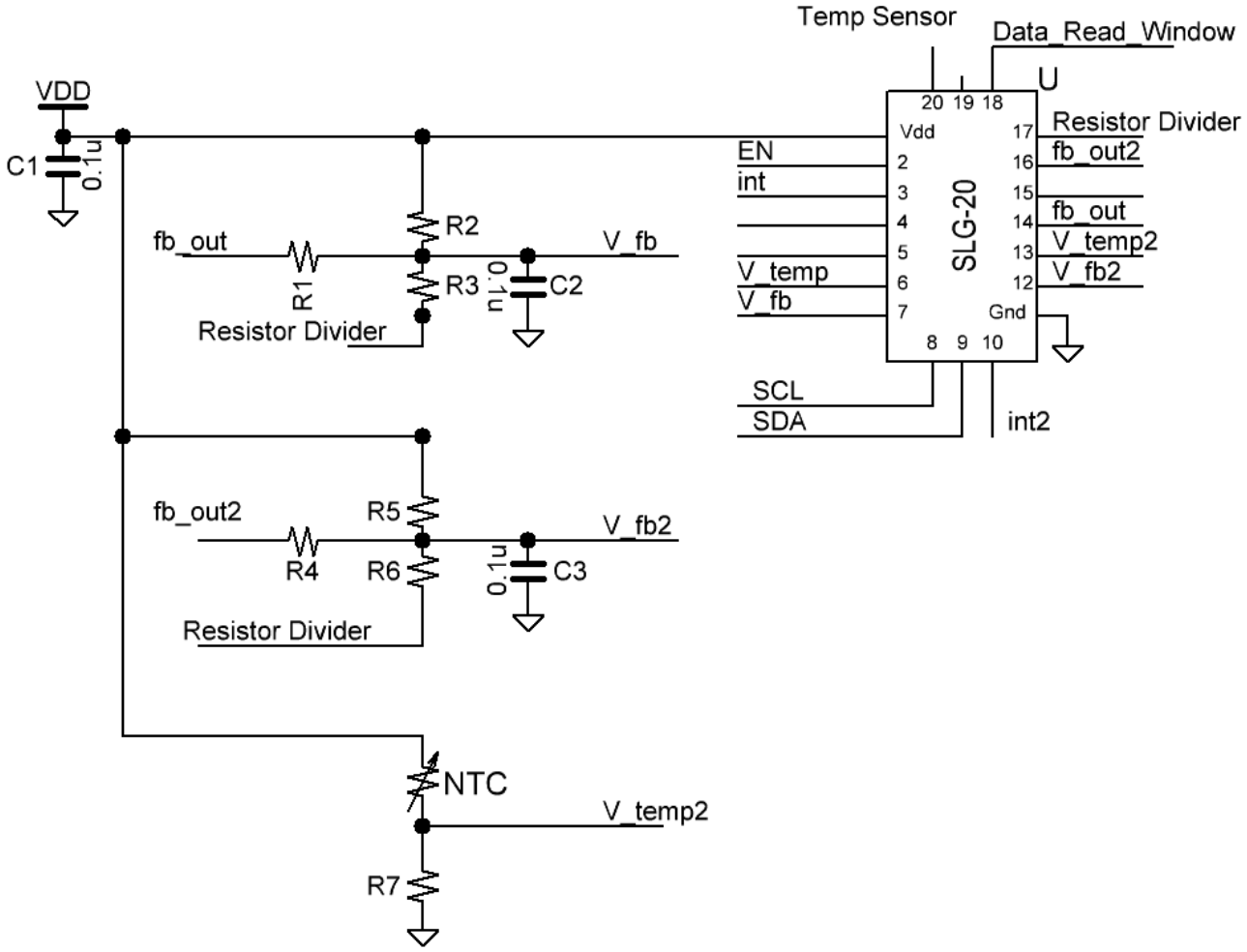
Application Circuit for 1-channel ADC in SLG46536



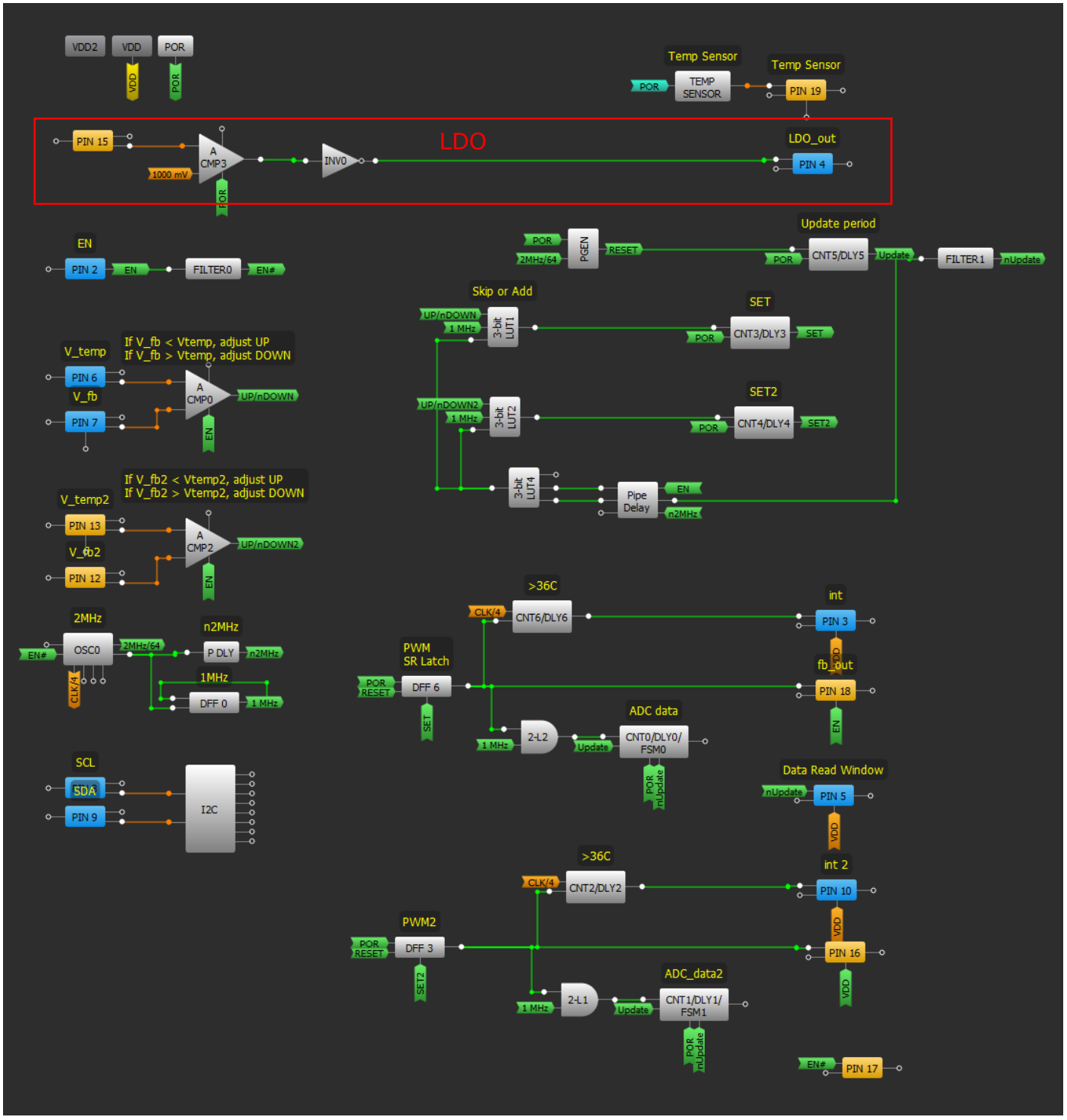
2-channel ADC in SLG46533



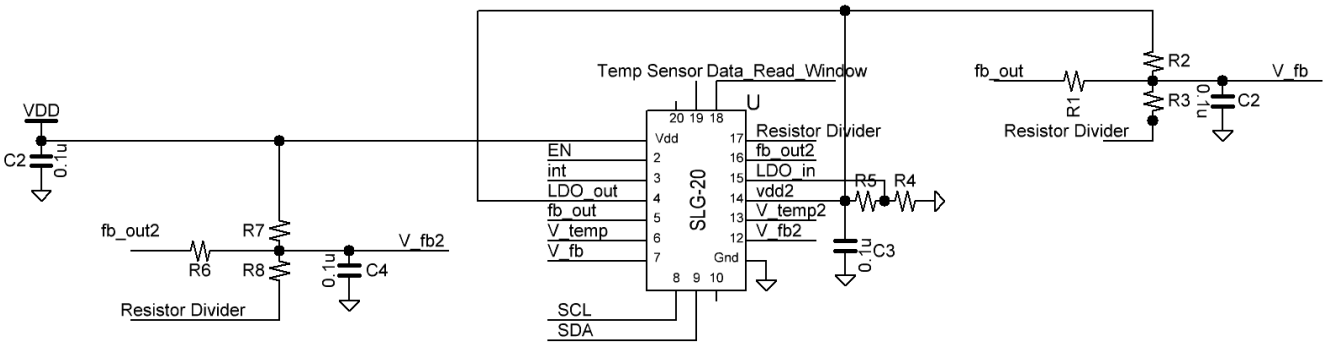
Application Circuit for 2-channel ADC in SLG46533



Non-ratiometric 2-channel ADC in SLG46538



Application Circuit for Non-ratiometric 2-channel ADC in SLG46538



IMPORTANT NOTICE AND DISCLAIMER

RENESAS ELECTRONICS CORPORATION AND ITS SUBSIDIARIES (“RENESAS”) PROVIDES TECHNICAL SPECIFICATIONS AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD-PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for developers who are designing with Renesas products. You are solely responsible for (1) selecting the appropriate products for your application, (2) designing, validating, and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. Renesas grants you permission to use these resources only to develop an application that uses Renesas products. Other reproduction or use of these resources is strictly prohibited. No license is granted to any other Renesas intellectual property or to any third-party intellectual property. Renesas disclaims responsibility for, and you will fully indemnify Renesas and its representatives against, any claims, damages, costs, losses, or liabilities arising from your use of these resources. Renesas' products are provided only subject to Renesas' Terms and Conditions of Sale or other applicable terms agreed to in writing. No use of any Renesas resources expands or otherwise alters any applicable warranties or warranty disclaimers for these products.

(Disclaimer Rev.1.01)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit www.renesas.com/contact-us/.

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.