AN60594

PSoC® 3 and PSoC 5LP: Low Frequency FSK Modulation and Demodulation

Author: Todd Dust
Associated Project: Yes
Associated Part Family: CY8C3xxx,CY8C5xxx
Software Version: PSoC® Creator™2.1 SPI
Related Application Notes: AN52705, AN2336, AN2108, AN58827

Abstract

AN60594 describes how to create a frequency shift keying (FSK) modulator and demodulator with PSoC® 3 and PSoC 5LP, with digital 0s represented by 2100 Hz, and digital 1s, represented by 1300 Hz. The method described in this application note uses zero CPU, it is completely done in hardware. This application note covers only the physical layer implementation of an FSK transmitter and receiver; higher-level encoding techniques and physical modem connections are not discussed.

Contents

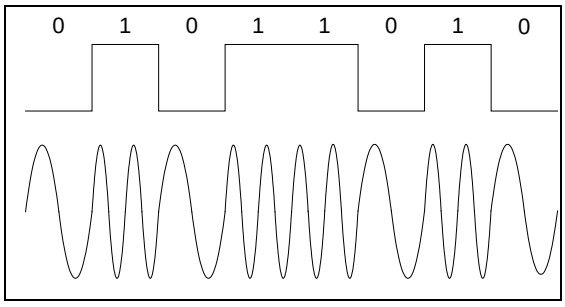
Introduction 1
FSK Modulator 2
  Transmit Results 2
  Generating Different Frequencies 4
FSK Demodulator 5
  Band-pass Filter 5
  Zero-Crossing Detector 6
  Correlator 6
  LPF 6
  Digitize 7
Receiving Results 8
Design Modifications 9
Attached Projects 9
Summary 9
About the Author 9
Appendix A 10
Document History 12

Introduction

In some applications, it is necessary to transmit digital data through analog signals. This is often due to bandwidth limitations in the transmission medium. There are numerous methods of transmitting digital data through analog signals; this application note specifically focuses on frequency shift keying (FSK) and how to implement an FSK modulator and demodulator in PSoC 3 and PSoC 5LP.

An FSK modulator converts digital data into two distinct frequencies; one frequency represents a digital 1 and the other represents a digital 0 (see Figure 1).

Figure 1. Example FSK Waveform



The FSK demodulator receives the transmitted analog signal and demodulates it to determine the correct sequence of 1s and 0s.

In this Application Note you will be shown how to configure a PSoC 3 and PSoC 5LP device to implement an FSK

modulator and demodulator. The example project included with this Application Note represents digital '0' with 2100 Hz and digital '1' with 1300 Hz. The example is not a complete modem; it does not include the phone line or other interfaces; it also does not include the data processing code.

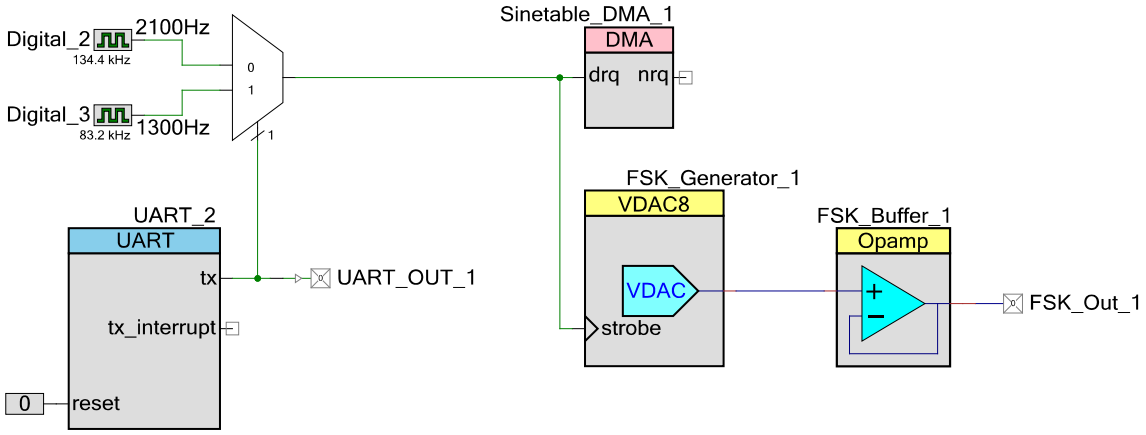
FSK Modulator

An FSK modulator converts digital data from a digital source such as a universal asynchronous receiver transmitter (UART) into the appropriate analog output signals. For this discussion, the analog output signal is a sine wave of either 1300 Hz or 2100 Hz, depending on the digital input. It is also important that the output be phase continuous, no discontinuities in the waveform

The included project creates sine wave with PSoC 3 and PSoC 5LP using a digital-to-analog converter (DAC) with the Direct Memory Access (DMA) periodically writing values to it from a sine table.

The DMA can be configured to transfer individual bytes from the sine table to the DAC based on a trigger from a hardware signal (DRQ). The sine table is 64 elements wide, so the trigger must occur 64 times faster than the target frequencies. To generate sine waves of 1300 Hz and 2100 Hz, the DMA DRQ is triggered at 83.2 kHz or 134.4 kHz, respectively. This example uses two separate clocks to trigger the DMA transfer. A digital mux controls the input into the DRQ, as Figure 2 shows. That MUX is controlled by the digital source, in this case a UART. This can be changed to any other digital component. For more information on DMA, see the application note, AN52705.

Figure 2. FSK TX PSoC Creator™ Schematic



The associated project is configured to create the FSK waveforms based on the output of a UART. This program can be loaded into a CY8CKIT-001. The main loop can be updated to control the output of the UART to send the desired data. The FSK output can be observed on P0.1 and the UART output can be observed on P1.7. The project is configured to continuously send the string "Hello World".

Transmit Results

Now that we know how to create an FSK modulator with PSoC 3 and PSoC 5 LP, it is time to look at the quality of the FSK waveforms. Presented below are screen captures of the FSK outputs and their FFTs.

Figure 3. 2100 Hz Output

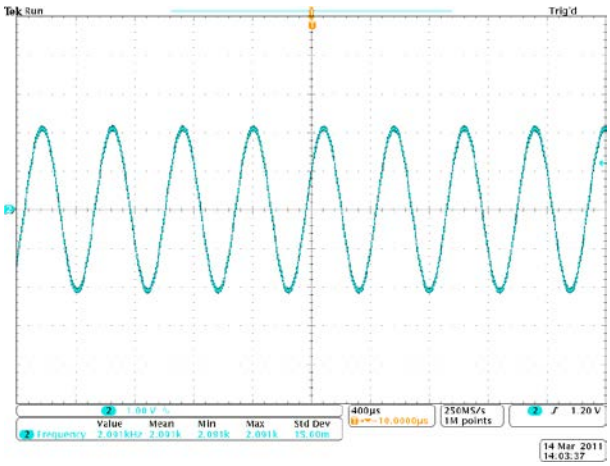
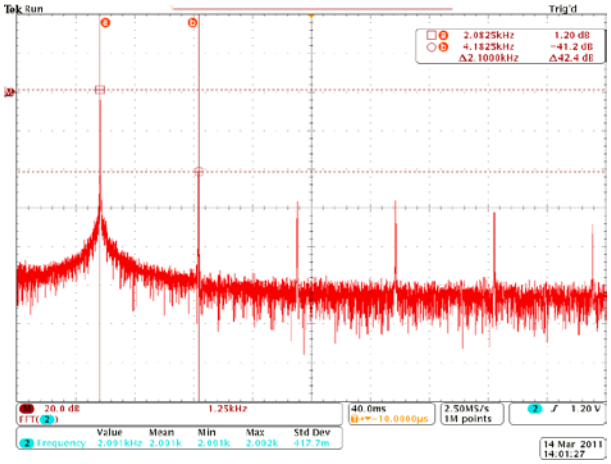
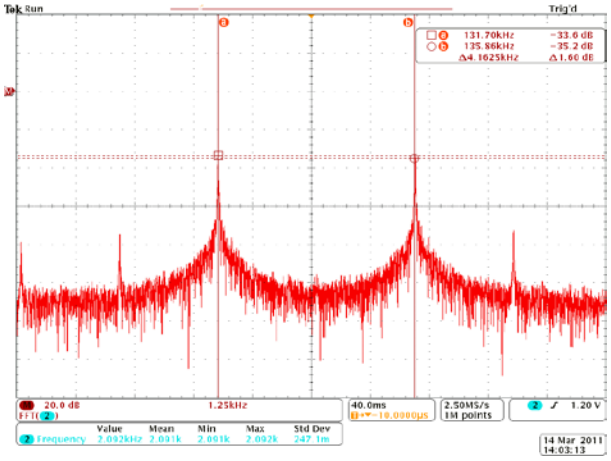


Figure 4. 2100 Hz FFT



The cursor window in the upper right of Figure 4 shows the fundamental frequency at 1.2 dB, and the second harmonic at –41.2 dB, which is approximately 42 dB down.

Figure 5. 2100 Hz Primary Alias



The cursor window in Figure 5 shows that the sample alias noise is located at approximately 131.7 kHz and 135.86 kHz, and has a magnitude of –33.6 dB and –35.2 dB, respectively.

Figure 6. 1300 Hz Output

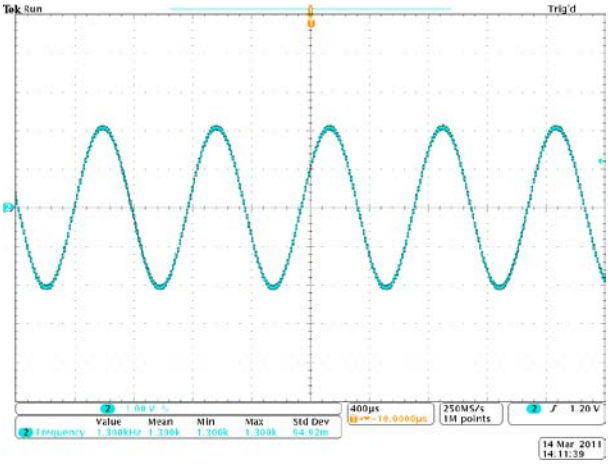
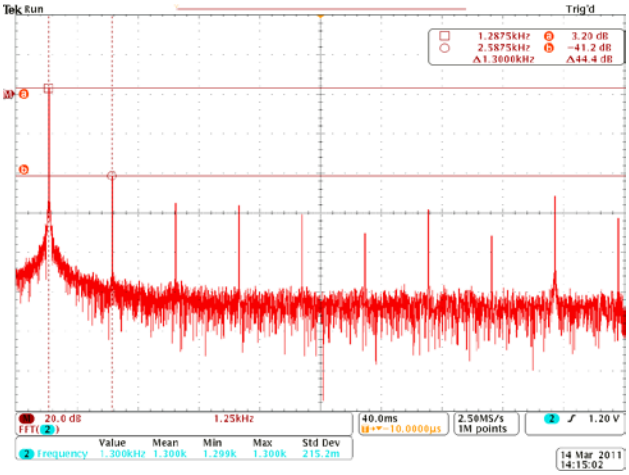
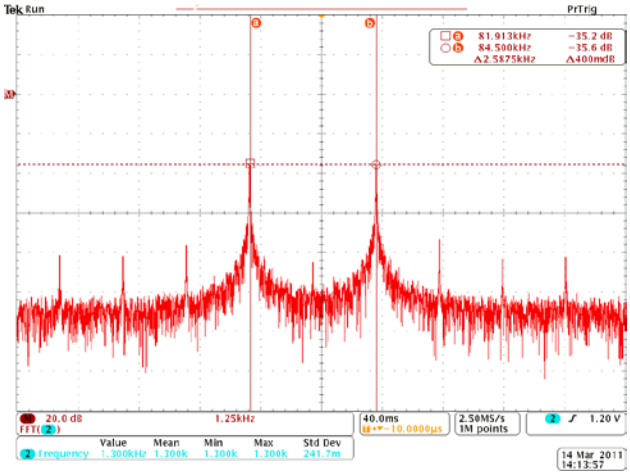


Figure 7. 1300 Hz FFT



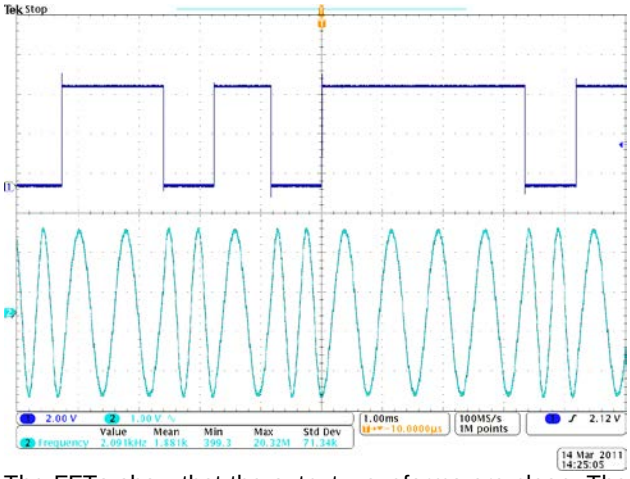
The cursor window in the upper right of Figure 7 shows the fundamental frequency at 3.2 dB, and the second harmonic at –41.2 dB, which is approximately 44 dB down.

Figure 8. 1300 Hz Primary Aliases



The cursor window in Figure 8 shows that the sample alias noise is located at approximately 81.9 kHz and 84.9 kHz, and has a magnitude of –35.2 dB and –35.6 dB, respectively.

Figure 9. Combined Output



The FFTs show that the output waveforms are clean. The second harmonic and higher are at least 40 dB below the fundamental. The combined waveform is also clean with no phase errors at bit transitions. With PSoC 3 and PSoC 5LP devices, it is easy to create an FSK modulator that requires no CPU interaction.

Generating Different Frequencies

For your application 1300 Hz and 2100 Hz may not be the frequencies that you need. This section describes how you can achieve different frequencies than those discussed above.

The FSK output frequency can be calculated by dividing the sample clock frequency by the size of the sine table, as shown in Equation 1.

$$OutputFreq = \frac{SampleClock}{N} \quad \text{Equation 1}$$

N is the number of elements in the sine table. The sample clock is the clock used to drive the DRQ of the DMA

To generate a digital 0 in the previous example, a 134.4 kHz sample clock was used with a 64-element sine table. Using Equation 1, the output frequency can be calculated as:

$$OutputFreq = \frac{134.4kHz}{64}$$
$$OutputFreq = 2100Hz$$

For a digital 1, an 83.2 kHz sample clock was used with the same 64-element sine table. Again, using Equation 1, the output frequency can be calculated:

$$OutputFreq = \frac{83.2kHz}{64}$$
$$OutputFreq = 1300Hz$$

To increase the output frequency, the sample clock can be increased, or the number of sine table elements can be decreased.

To decrease the output frequency, the sample clock can be decreased, or the number of sine table elements can be increased.

The question that often arises at this point is how fast can I get my FSK output with this method? This method only works with low frequencies; it will not work for MHz signals. Below is a discussion on how to determine the maximum output frequency.

A VDAC configured in the 0-4.048 V range has a maximum sample rate of 250 ksps; using a 64-bit sine table, the maximum output frequency that can be achieved is 3.9 kHz. A VDAC configured in the 0-1.024 V range has a maximum sample rate of 1000 ksps; using a 64-bit sine table, the maximum output frequency that can be achieved is 15.625 kHz.

If an IDAC is used instead of a VDAC, the maximum sample rate is 8000 ksps. This allows for a maximum output frequency of 125 kHz with a 64-element sine table.

If an IDAC is used, the current needs to be routed to an external resistor to convert the current to a voltage. See Figure 10.

Figure 10. Converting an IDAC to a VDAC

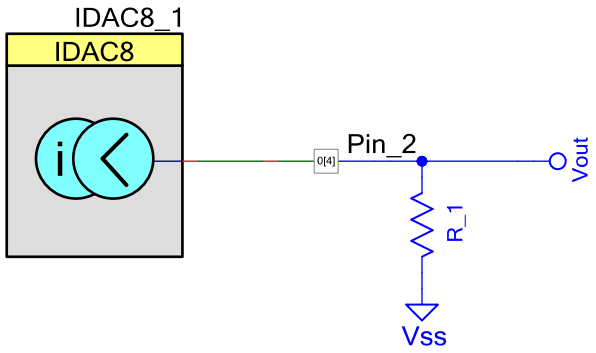


Table 1below gives an idea of the maximum output frequencies for different sized sine tables.

Table 1. Sine Table Size versus. Max Output Frequency

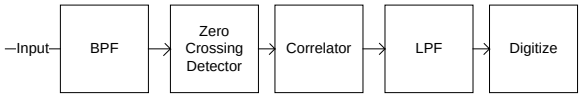
Sine Table Elements	IDAC	1V VDAC	4V VDAC
256	31.25 kHz	3.9 kHz	976 Hz
128	62.5 kHz	7.8 kHz	1.95 kHz
64	125 kHz	15.6 kHz	3.9 kHz
32	250 kHz	31.25 kHz	7.8 kHz
16	500 kHz	62.5 kHz	15.6 kHz

Decreasing the number of elements in the sine table has the disadvantage of increasing the harmonic and sampling noise. When designing an FSK modulator, the tradeoffs between speed and noise need to be considered. The FFTs on the previous page give a good approximation of the noise present in this system for a 64 element sine table.

FSK Demodulator

The FSK demodulator takes the analog input signal and demodulates it into the appropriate sequence of digital 1s and 0s. The block diagram for the FSK demodulator is shown in Figure 11.

Figure 11. Demodulator Block Diagram



Band-pass Filter

The band-pass filter (BPF) is designed to allow 1300 Hz and 2100 Hz signals to pass through with little attenuation (around 1 dB), and everything else to be attenuated. The purpose of this filter is to remove any out-of-band noise, thus making the demodulation processes easier.

This filter was designed using standard filter design techniques. It has a center frequency close to 1500 Hz. The bandwidth is set to 1570 Hz to achieve less than 1 dB of attenuation at the modulating frequencies. This filter is designed for a gain of 1 V/V. The circuit for the BPF is shown in Figure 12; derivation of component values is a standard design problem, see Appendix A for more detail on how the filter was designed or if you need to design a filter for different frequencies.

Figure 12. BPF Circuit

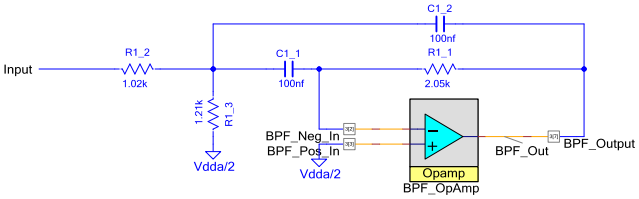
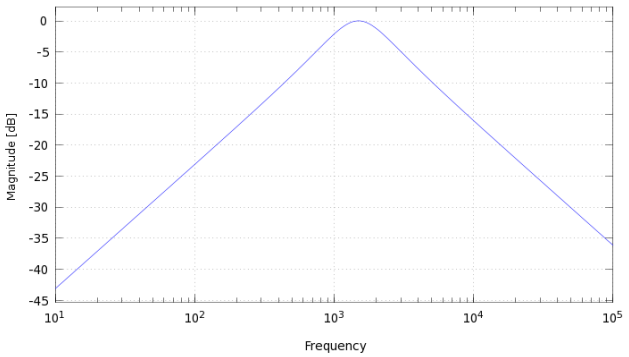


Figure 13. BPF Bode Plot

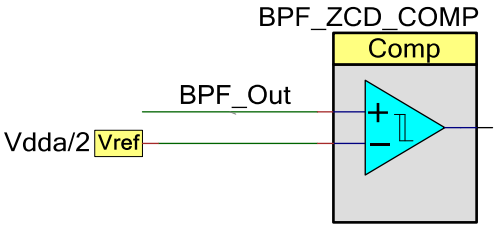


This filter is meant as an example to the designer. The filter should be appropriately modified for the target design. Signaling frequencies, signal amplitude, and out-of-band noise all need to be considered when designing this front-end BPF. The topology in can be modified for increased gain, as well as different center frequencies and bandwidths. If more out-of-band attenuation is needed, a higher-order filter must be constructed out of multiple PSoC opamps.

Zero-Crossing Detector

After the waveforms are filtered, it is important to digitize them for use in the correlator. This is done by constructing a simple zero-crossing detector. This detector can be created with a comparator set to trip at VDD/2.

Figure 14: Zero-Cross Detector



Correlator

The theory for the correlator is available in application note AN2336. The basic function of the correlator is to delay the digital signal out of the zero-crossing detector, and then XOR that signal with the non-delayed signal. The end result is that one frequency ends up with a high duty cycle, while the other frequency has a low duty cycle at the output of the XOR gate. Figure 15 and Figure 16 are scope captures at the output of the XOR gate for the two different frequencies.

Figure 15. Correlator Output at 2100 Hz

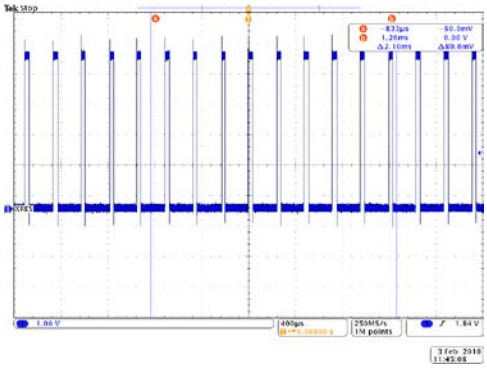


Figure 16. Correlator Output at 1300 Hz

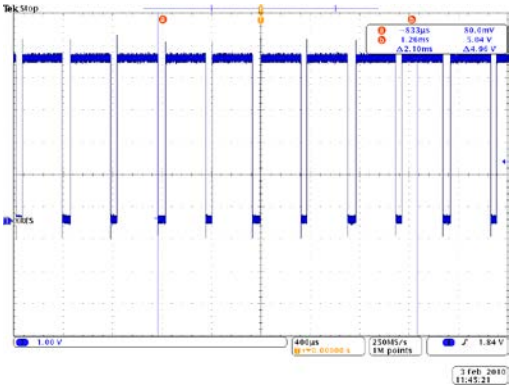
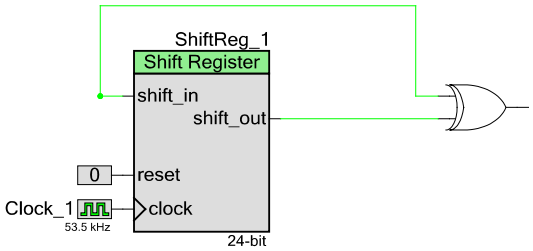


Figure 15 and Figure 16 show that the output of the correlator has a high duty cycle at 1300 Hz, while the output has a low duty cycle at 2100 Hz.

From AN2336, it is calculated that a delay of approximately 450 µs is needed. This delay is constructed out of a shift register; see Figure 17. The shift register is configured for 24 bits and the input clock is 53.5 kHz or 18.7 µs; 18.7 µs multiplied by 24 is approximately 448 µs.

Figure 17. PSoC 3 Correlator



If signal frequencies other than 2100 Hz and 1300 Hz are used, the delay must be adjusted accordingly. The delay can be calculated by using Equation 2. For a detailed derivation of Equation 2, see AN2336.

$$Diff = \cos(2\pi f_L d) - \cos(2\pi f_H d) \qquad \text{Equation 2}$$

f_L is the lower frequency, f_H is the high frequency, and d is the delay in seconds. You can plug this equation into Excel to find the maximum difference.

As a practical matter, the delay with the largest possible difference simplifies the filtering and detection process.

LPF

The best way to remove the glitches in Figure 15 and Figure 16 is to low-pass filter the data out of the correlator. For this design, a 3-pole filter with a cut-off at 1.1 kHz is designed. The LPF is designed so that the rise time is sufficiently fast for the signaling constraints. For example, at 1200 baud, the bit time is approximately 800 µs. Thus,

the rise time of the LPF must be faster to avoid errors. This filter has a rise time of approximately 450 µs.

Figure 18. Low-Pass Filter

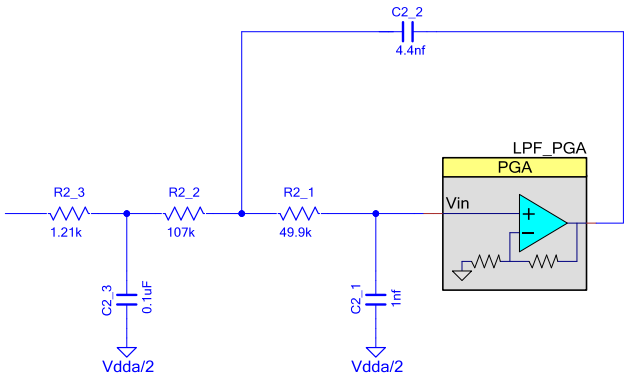
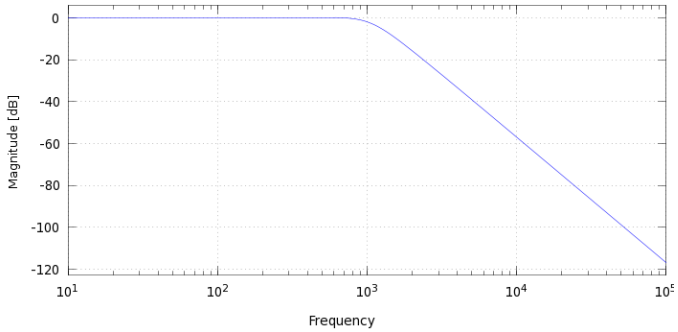


Figure 19. LPF Bode Plot



The LPF is constructed out of external components and a PSoC PGA configured in unity gain mode. The PGA is used to save one of the valuable opamps for other uses in your application

The LPF must be designed so that it sufficiently removes all glitches, but with a rise time sufficiently shorter than the bit time, to avoid bit errors.

Digitize

The output of the LPF is then run through a comparator to digitize the signal. This digitized signal can then be fed into a UART or other digital circuitry. In some designs, it may be necessary to add hysteresis to this comparator. This becomes necessary in systems with faster baud rates and higher signal noise. The PSoC comparator has a built-in 10 mV of hysteresis. AN2108 describes how to achieve wider hysteresis

Receiving Results

Figure 20. Putting It All Together

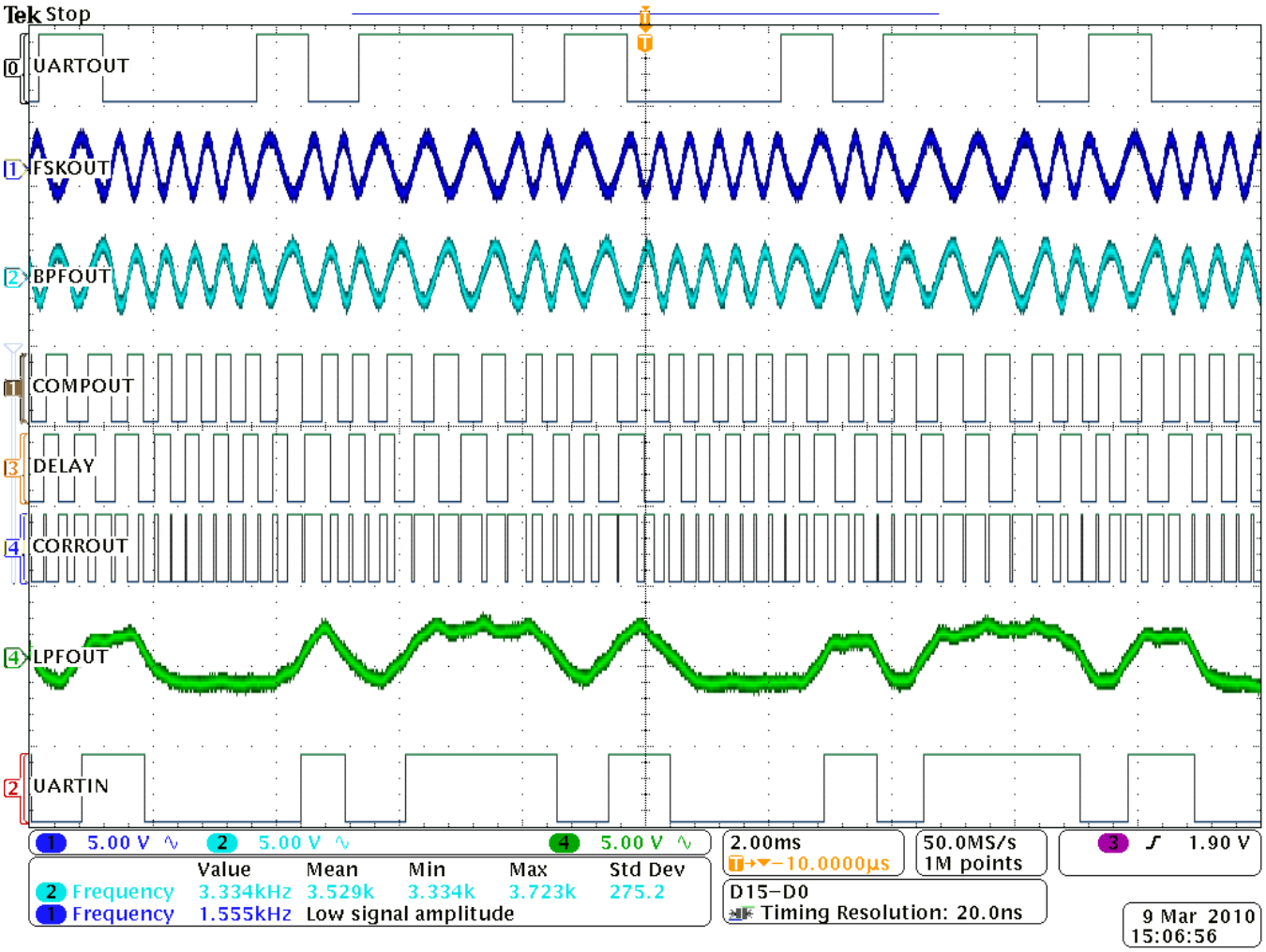


Figure 20 shows the signals received:

- The top signal (UARTOUT) is the output of the TX UART.
- The second signal (FSKOUT) is the analog output from the FSK modulator described earlier in this application note.
- The third signal (BPFOUT) is the output of the BPF; 2100 Hz has slightly more attenuation because of resistor tolerance issues.
- The fourth signal (COMPOUT) is the output of the first zero-crossing detector circuit.
- The fifth signal (DELAY) is the output of the shift register, the delayed signal.
- The sixth signal (CORROUT) is the XOR oF the COMPOUT and DELAY signals; notice again that with one input frequency the correlator output has a high duty cycle, while at the other frequency it has a low duty cycle.
- The seventh output (LPFOUT) is that of the LPF.
- The eighth signal (UARTIN) is the final output that goes to the receive UART. Notice that the eighth signal is essentially a delayed version of the first signal (UARTOUT).

Design Modifications

A bare essential FSK modulator and demodulator design has been shown. There are numerous opportunities for design refinements, including increasing gain, improving filter selectivity, and adapting to other operating frequencies. The design principles presented are easily extended to these other applications.

Attached Projects

There are two projects attached to this application note: FSK_RX and FSK_TX. They are located in the same workspace. The FSK_TX project transmits the string “Hello World”. The FSK_RX project receives that string and displays it on the LCD of one of the Cypress Standard Development boards (CY8CKIT-001, 030, or 050). The FSK output of FSK_TX can be found on P0.1.

The resistors and capacitors, described above for the BPF and LPF, must be added to the development board for the FSK_RX project. Follow the schematics shown in this document or in the project.

Attach the FSK output on FSK_TX board (P0.1) to the input of the BPF on the FSK_RX board. In addition, attach the grounds of the two development boards together.

The projects attached to this application note are designed for both PSoC 3 and PSoC 5LP. The default device is PSoC 3 (CY8C3866AXI-040). To change to another device, including PSoC 5LP devices, go to **Project** > **Device Selector** and select the device. For further information, see the PSoC Creator Help article, Device Selector.

Summary

This application note explains how to create an FSK modulator and demodulator in PSoC 3 and PSoC 5LP without any CPU interaction. Attached to this application note are projects that implement what is described. The appropriate external components must be provided for the projects to work.

With a proper understanding of the material presented in this application note, you can now implement FSK in your design.

The Advantage to using a PSoC device is that the CPU was not used, so it can be used to accomplish other tasks for your application. Also there are still numerous hardware resources left over for your application.

About the Author

Name:	Todd Dust
Title:	Applications Engineer
Background:	BSEE, Seattle Pacific University
Contact:	tdu@cypress.com

Appendix A

In the main body of the Application Note, we did some hand waving when it came to deriving the requirements for the Band Pass Filter. As designing the BPF is a more advanced topic and not the main focus of the Application Note, the discussion on the BPF was moved to this Appendix.

Remember that the signaling frequencies are 1300 Hz and 2100 Hz. The goal of this filter is to have very little attenuation at these frequencies, ~ -1 dB if possible, and as much attenuation on any out of band signals. The filter was designed following the steps below.

The first step is to find the corner frequency of a Low Pass Filter (LPF) with a –1dB point at 800Hz, this number will give us the bandwidth of the BPF filter. The 800Hz is computed by taking the difference of the 2100Hz and 1300Hz

Consider the Transfer Function of a 1st order LPF

$$\frac{V_{out}}{V_{in}} = \frac{1}{\sqrt{1+\left(\frac{\omega}{\omega_c}\right)^2}}$$

We wanted a gain of -1dB so insert that into the equation.

$$-1dB = \frac{1}{\sqrt{1+\left(\frac{\omega}{\omega_c}\right)^2}}$$

Next, we can convert -1dB into units we can work with.

$$-1 = 20Log\left(\frac{1}{\sqrt{1+\left(\frac{\omega}{\omega_c}\right)^2}}\right)$$

From here we can do a lot of algebra and reduce this equation so that we can determine ω_c. After all the Algebra and re-arranging the equation looks like the following:

$$\omega_c = \frac{\omega}{\sqrt{\left(\frac{1}{10^{\frac{-1}{20}}}\right)^2 - 1}}$$

Now we plug in 800 for ω

$$\omega_c = \frac{800 * 2\pi}{\sqrt{\left(\frac{1}{10^{\frac{-1}{20}}}\right)^2 - 1}}$$

$$\omega_c = 9878.306$$

$$BW = 1572.2\ Hz$$

Now we know that the Bandwidth of the filter will be 1572.2 Hz. The next step is to determine the center frequency of the filter. To do this we need to compute the following equation

$$f_c = \sqrt{f_u * f_l}$$

Where fc is the center frequency, f_u is the upper frequency and f_l is the lower frequency.

We already calculated the BW to be 1572.2 Hz, thus we need to now determine the upper and lower frequencies of the Band. The upper and lower frequencies are the –3dB points of the filter. In order to find these frequencies you take the 800 Hz difference between 2100 and 1300, and subtract that from the BW of 1572.2 Hz, this gives us 772.2Hz, divide this by two and we get 386.1 Hz to add to either side of the filter.

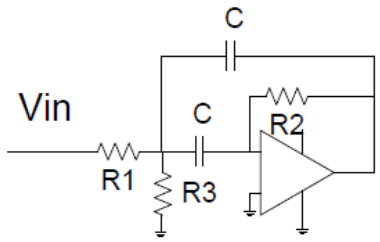
$$f_u = 2100 + 386.1 = 2486.1$$
$$f_l = 1300 - 386.1 = 913.9$$

$$f_c = \sqrt{2486.1 * 913.9} = 1507.3\ Hz$$

Now we know the BW of the filter, and the Center Frequency. The Q of the filter can be calculated with the following equation

$$Q = \frac{f_c}{BW} = \frac{1507.3}{1572.2} = .96$$

I chose a Multi-Feedback(MFB) BPF, the generic implementation of a MFB BPF looks like the following:



Below are the equations for the resistor values

$$R_2 = \frac{Q}{\pi * C * f_c}$$

$$R_1 = \frac{R_2}{-2 * A_m}$$

$$R_3 = \frac{-A_m * R_1}{2 * Q^2 + A_m}$$

$$A_m = -1$$

Now we just need to pick a value for C. For this project I just picked a value that I could easily find in the lab, which was 100*nf*. C needs to be chosen such that the values for the resistors are realistic and aren't too big or too small. Too big will mean more noise, too small will mean more power.

Using 100 *nf* and all of the other information the resistors were calculated to be the following values

$$R_2 = 2027.7\Omega$$
$$R_1 = 1013.9\Omega$$
$$R_3 = 1202.4\Omega$$

If you need to use different frequencies for your design, then follow the steps above but change the frequency values

Document History

Document Title: PSoC® 3 and PSoC 5LP: Low Frequency FSK Modulation and Demodulation - AN60594

Document Number: 001-60594

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	2900900	TDU	03/29/2010	New application note
*A	2991607	TDU	07/22/2010	Updated software version and fixed branding discrepancies
*B	3017205	TDU	08/27/2010	Added the Device Selection section
*C	3208213	TDU	03/28/2011	Changed title of document Changed transmit to use 8-bit VDAC Added information on how to modify transmit frequencies Added information on how to modify receive frequencies Updated projects
*D	3347563	TDU	08/17/2011	Updated project files.
*E	3470298	TDU	12/21/2011	Updated Template Updated Software Version to PSoC Creator® 2.0.
*F	3819270	RRSH	11/22/2012	Updated for PSoC 5LP.
*G	3922897	TDU	03/06/2013	Updated Document Title to read as “PSoC® 3 and PSoC 5LP: Low Frequency FSK Modulation and Demodulation - AN60594”. Updated Abstract. Updated Introduction. Renamed “FSK Transmitter” as “FSK Modulator” and updated the same section. Renamed “FSK Receiver” as “FSK Demodulator” and updated the same section. Updated Receiving Results. Updated Design Modifications. Updated Attached Projects. Updated Summary. Added Appendix A.

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

Automotive	cypress.com/go/automotive
Clocks & Buffers	cypress.com/go/clocks
Interface	cypress.com/go/interface
Lighting & Power Control	cypress.com/go/powerpsoc cypress.com/go/plc
Memory	cypress.com/go/memory
PSoC	cypress.com/go/psoc
Touch Sensing	cypress.com/go/touch
USB Controllers	cypress.com/go/usb
Wireless/RF	cypress.com/go/wireless

PSoC® Solutions

psoc.cypress.com/solutions

PSoC 1 | PSoC 3 | PSoC 5LP

Cypress Developer Community

Community | Forums | Blogs | Video | Training

Technical Support

cypress.com/go/support

PSoC is a registered trademark of Cypress Semiconductor Corp. PSoC Creator is a trademark of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are the property of their respective owners.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

Phone : 408-943-2600
Fax : 408-943-4730
Website : www.cypress.com

© Cypress Semiconductor Corporation, 2010-2013. 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.

This 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.