Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.

AN76530

Psoc® 1 Automotive Ultrasonic Distance Measurement For Park-Assist Systems

Author: Valeriy Kyrynyuk, Ben Kropf, Balaji M V

Associated Project: Yes

Associated Part Family: CY8C24x23A, CY8C24x94, CY8C29x66, CY8C28xxx

Software Version: PSoC Designer™ 5.4

Related Application Notes: AN50987, AN2219

To get the latest version of this application note, or the associated project file, please visit http://www.cypress.com/go/AN76530.

AN76530 describes how to use PSoC® 1 devices to build a four-channel ultrasonic distance measurement system for ultrasonic parking-assistance applications (UPA). This application note explains how Cypress's PSoC is the best fit for UPA applications; it includes an example project and reference hardware design.

Contents

Introduction

Ultrasonic parking-assistance (UPA) systems are increasingly popular in cars because they enhance safety and driver convenience, especially in large cities. This application note shows you how to create an ultrasonic distance-measurement system using enclosed ultrasonic transducers, which are typical in automotive applications. Our implementation measures the distance between the ultrasonic transducers and any nearby objects. You can use the distance value locally within the system to control outputs or displays. You also can send the distance value, using a communication interface, to be used by another controller in the vehicle. Modern safety expectations and driver convenience, along with parking conditions and increased vehicle size, are driving the popularity of UPA systems.

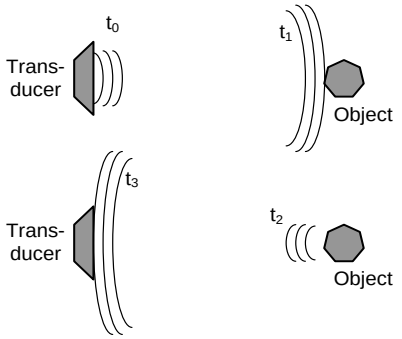
Parking Assistance

The main task of the distance-measurement system is to measure the distance of the objects that are within the beam angle of each ultrasonic transducer. Using the communication interface, the system must provide the distance measurement to the main MCU for further processing.

Ultrasonic Distance Measurement

Ultrasonic distance-measurement is based on the speed property of sound. The system transmits multiple sound waves that travel out into the air. These sound waves reflect off of any objects they impact and return back as an echo to the location from which they originated. The system detects these reflected sound waves (that is, echoes). The time between the transmission of the sound waves and the detection of the echo is measured, as shown in Figure 1. At time t0, the transducer creates the sound waves. At time t1, the sound waves impact an object. At time t2, the waves have reflected off of the object and are traveling back toward the transducer. At time t3, the echo has impacted the transducer, which detects these waves. The system subtracts t0 from t3 to calculate the total travel time of the sound.

Figure 1. Ultrasonic Distance Measurement



The sound's travel time is divided by the speed of the sound to calculate the total distance that the sound waves traveled. This distance is divided by two to calculate the distance of the object that caused the echo. These calculations are shown in Equation 1, where s is the distance between the transducer and the detected object, v is the speed of sound, and t is the measured time between sound wave transmission and detection of the echo.

The following "Equation 1" text uses the Equation Title style.

Equation 1 $$s = \frac{v \cdot t}{2}$$

The speed of the sound depends on several factors, including temperature, density, and air composition. The speed of the sound in a gas is given by Equation 2. In the equation, v is the speed of sound, γ is the adiabatic index, k is the Boltzmann constant, T is the absolute temperature in Kelvins, and m is the mass of a single molecule of the gas in kilograms.

Equation 2 $$v = \sqrt{\frac{\gamma \cdot k \cdot T}{m}}$$

Where, for dry air:

$k = 1.3806503 \times 10^{-23}\ m^2\ kg\ s^{-2}\ K^{-1}$

$\gamma = 1.4$

$m = 4.8097808 \times 10^{-26}\ kg$

Typically, Equation 2 is simplified into Equation 3, which takes only air temperature into account.

Equation 3 $$v = 20.046\sqrt{T}$$

The nominal speed of sound at 23 °C is nearly 345 m/s. In this application note, the influence of temperature on sound speed is not taken into account. Therefore, Equation 1 can be simplified into Equation 4.

Equation 4 $$s = \frac{345 \cdot t}{2}$$

However, the influence of temperature on the speed of sound can easily be included in the calculations if air temperature was measured. It would also be possible to measure other environmental properties (such as humidity) that would also affect the sound speed to give even more compensation to improve accuracy. However, the added system cost of measuring environmental properties to do compensation often is not worth the improvements in accuracy. The various compensation techniques are beyond the scope of this application note.

There are many other complicated aspects of sound wave properties. However, the scope of this application note does not cover those details. Instead, this application note is primarily focused on the analog and digital electrical circuit implementation of the system. Therefore, more complex properties of sound are not considered. Instead, we are applying a simplified model of sound wave travel.

Transducers

In automotive ultrasonic parking assistance applications, the most common type of ultrasonic transducer is enclosed. Enclosed transducers are hermetically sealed to avoid any contamination that would reduce reliability. Additionally, the transducers in these applications are transceivers. That is, they can convert electrical energy to sound energy and convert sound energy to electrical energy. Therefore, each transducer can create and detect ultrasonic sound.

Enclosed transducers are much more reliable and rugged. However, those qualities also force a tradeoff in performance. An enclosed sensor can be 10 to 20 times less effective at energy conversion than an equivalent open transducer. (The open type transducer design exposes the piezo bender bonded with a metal conical cone behind a protective screen.) This means it is more difficult to implement an ultrasonic distance measurement system when using enclosed, rather than open, transducers. When using enclosed transducers the system must have a significant amount of gain to achieve the system's requirements.

Ultrasonic transducers have these important properties:

- Ultrasonic resonant frequency (typically 40 kHz)
- Maximum applied voltage (typically 100 VPP)
- Beam pattern angle
- Ringing time (typically about 1 ms)
- Nominal resonant impedance
- Nominal capacitance

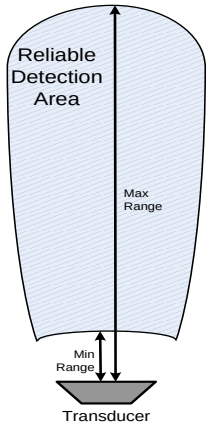
Maximum Range

The desired detection range for a system varies by application. The required maximum range could vary from 1.0 m to 10.0 m. The target maximum detection distance for the implementation in this application note is 2.5 m or greater.

- The maximum detection range mainly depends on the following factors:
- The amount of power used to drive the transducer
- The signal gain and signal-to-noise ratio (SNR) of the circuit used to measure the return signal
- The properties of the transducer itself.
- In general, if you need a large maximum range, then maximize the signal-driving power, the signal gain, and the SNR, and choose a transducer with lower ultrasonic resonant frequency (30 kHz to 50 kHz), good receiver sensitivity (>-70 dB), higher transmitting sound pressure level (>120 dB), maximum input voltage range (>120 Vpp)

Figure 2 shows the area in which a transducer can reliably detect an object. The maximum range is the edge of the area farther away from the transducer.

Figure 2. Object Detection Range



Minimum Range

An ultrasonic distance measurement system typically has a minimum range. Objects within this range cannot be reliably detected. A minimum range for a system can vary between 0 m and 0.3 m. In this application note, the target minimum range for the implementation is 0.2 m or less.

The minimum range is an undesirable aspect of the system caused by the use of transceiver transducers. A transceiver transducer transmits the sound and receives it. Therefore, there is a period after transmitting the sound during which the system must wait for the transducer to stop “ringing.” If the system immediately started waiting for the return signal, it would always measure a distance of zero, because it would immediately detect the ringing signal on the transducer. Therefore, the system must wait until the ringing on the transducer has sufficiently diminished for normal signal detection to begin. The waiting time is directly proportional to the minimum detection range distance.

The amount of ringing delay time is inherent to transducers and does not vary widely from sensor to sensor. Therefore, you must either accept this ringing period or find a way to mitigate it. Advanced techniques for reducing the ringing period are beyond the scope of this application note.

Figure 2 shows the area in which a transducer can reliably detect an object. The minimum range is the edge of the area closest to the transducer.

When discussing or determining range performance, choose a reference object. The reference object must always be detected when it is inside of the shaded area depicted in Figure 2. The object may or may not be detected if it is outside of the shaded area or on the edge of the area. For the purpose of this application note, the reference object that must be detected at a range of between 0.2 m and 2.5 m is a piece of wood (60 × 30 × 1 cm) or any solid plane object of specified dimension.

System Implementation

Figure 3 shows a typical block diagram for an ultrasonic parking assistance system using a conventional microcontroller (MCU). The MCU is responsible for generating a 40 kHz signal burst. At the same time, a timer is started. The 40 kHz signal is converted into a 40 kHz sound wave at the ultrasonic transducer. The sound wave reflects off an object and is converted back into an electrical signal by the same transducer. The signal is amplified and filtered. The MCU has a comparator that detects the electrical signal created by the reflected sound waves. The comparator trips when the signal is detected, stopping the timer. The MCU's CPU is used to read or calculate the elapsed time and send this value somewhere else in the system using a communication interface.

The power amplifier converts the logic-level square wave generated by the MCU into a signal with much greater power. This conversion is necessary because the general-purpose input/output (GPIO) pins of typical MCUs cannot drive high-voltage or high-current signals. Therefore, a power amplifier circuit is needed to deliver maximum power to the transducer. Typically, the power amplifier circuit uses the battery voltage of the vehicle and voltage-boosting circuit to amplify the power driven to the load. When the transducer is driven with the high-power 40-kHz signal, this same signal will appear at the input of the signal chain. This very high voltage easily could damage the amplifier, filter, or MCU devices because it would exceed their maximum input voltage ratings. Therefore, you need a signal preconditioning circuit that can attenuate the high-voltage signal to lower voltage levels without damaging the rest of the devices in the signal chain. In addition, this preconditioning circuit must have high input impedance so that it does not attenuate the power delivered to the transducer. In effect, the signal preconditioning circuit serves as a high-impedance buffer or amplifier that tolerates large input voltages and clamps its output voltage to safe levels.

As seen in Figure 3, many external devices and circuits are needed in addition to the MCU. Most of these circuits are analog devices.

Figure 4 shows the same ultrasonic parking assistance system, but it uses a PSoC device instead of a typical MCU. Note that many of the analog circuits and components have been integrated into the PSoC device.

Figure 3. Typical Ultrasonic Parking Assistance System

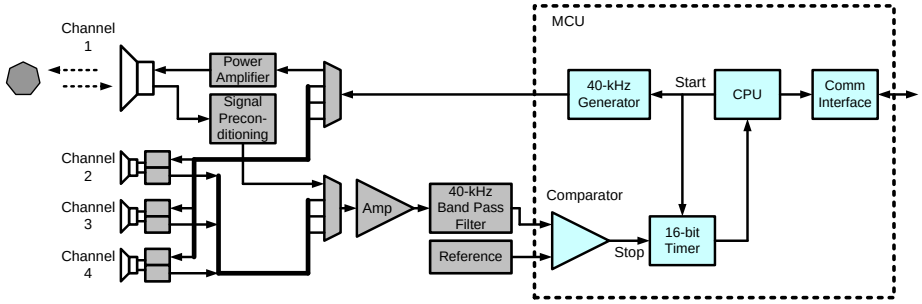
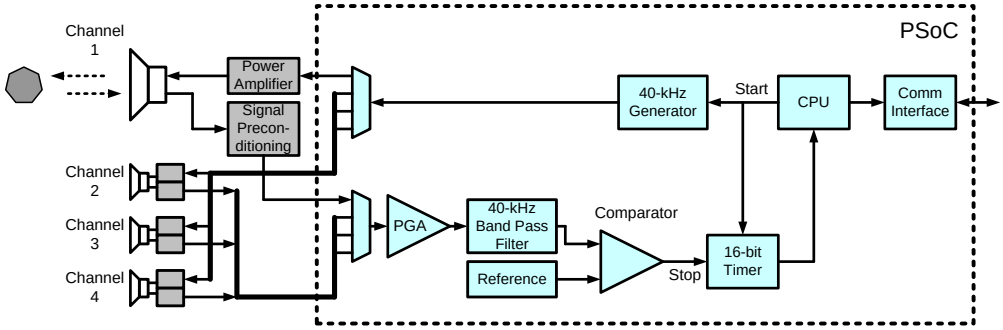


Figure 4. PSoC-Based Ultrasonic Parking Assistance System



The PSoC can internally implement the digital demultiplexer, the analog mux, the voltage amplifier, the band-pass filter, and the reference. However, the power amplifier and signal preconditioning circuits cannot be integrated into the PSoC (or any other typical MCU) because of the high-voltage signals with which they interface.

External Power Amplifier Circuit

Design the power amplifier circuit based on the transducer you use and the application requirements. The primary requirements of the power amplifier are listed below:

- It must be controlled by a signal from the PSoC device.
- It must operate at the resonant frequency of the transducer. That is, the amplifier should be tuned for an operation frequency of 40 kHz.
- It must not drive the transducer with too much voltage. Specifically, do not exceed the transducer's maximum voltage specification.

Figure 5 shows an example schematic of an effective power amplifier circuit. Transistor Q1 is used as a switch to drive the transformer primary with the 40-kHz signal burst. The transformer multiplies its input voltage (VBat) by its turns-ratio (N) to drive the transducer with a high-voltage signal. In this case, the transducer must be able to withstand a peak voltage of the maximum VBat voltage multiplied by N. Resistor R3 helps reduce ringing on the transducer when it is not being driven. Resistor R1 and capacitor C1 provide a power supply bypass circuit, to reduce power supply noise and spurious oscillations. Note that these components are not part of the power amplifier, and only one instance of them is needed.

Note that the transformer must be specially chosen, designed, or tuned to have resonance frequency of 40 kHz, and it must be matched with the transducer being used.

Figure 5. Example Power Amplifier Circuit

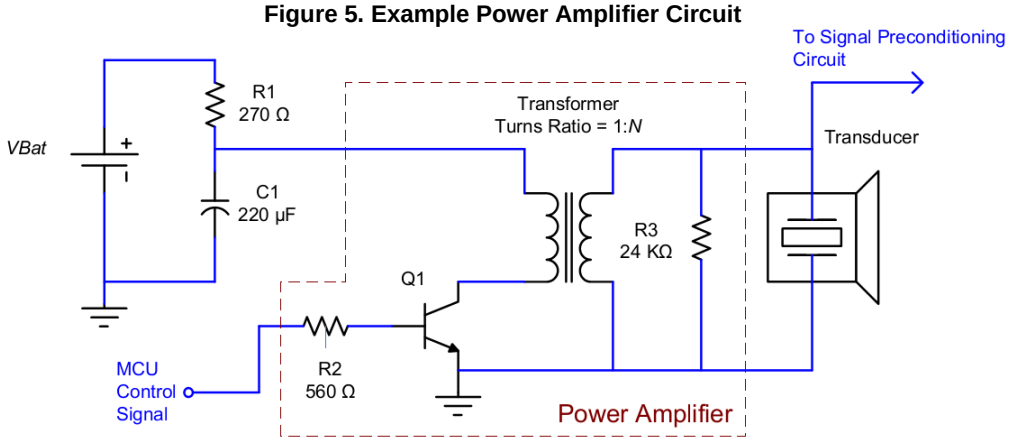


Figure 5 shows only an example circuit. Other topologies also are possible, such as inductive boost, inductive flyback, or capacitive charge pump topologies. Also, for each circuit, the amplifier can generally be designed to be driven single-ended or differentially. Driving the circuit differentially allows the effective power to be doubled. However, doing that leads to a more complex circuit and higher component cost.

External Signal Preconditioning Circuit

The primary requirements of the signal preconditioning circuit follow:

- It must attenuate any high-voltage signals to a level that will not damage the PSoC device.
- It must have a high enough impedance so that it does not attenuate too much the power delivered to the transducer.
- It must bias the signal to be referenced to the analog reference voltage provided by the PSoC device.

It must not attenuate the small signals that appear on the transducer during the measurement phase. Figure 6 shows an example of an effective signal preconditioning circuit.

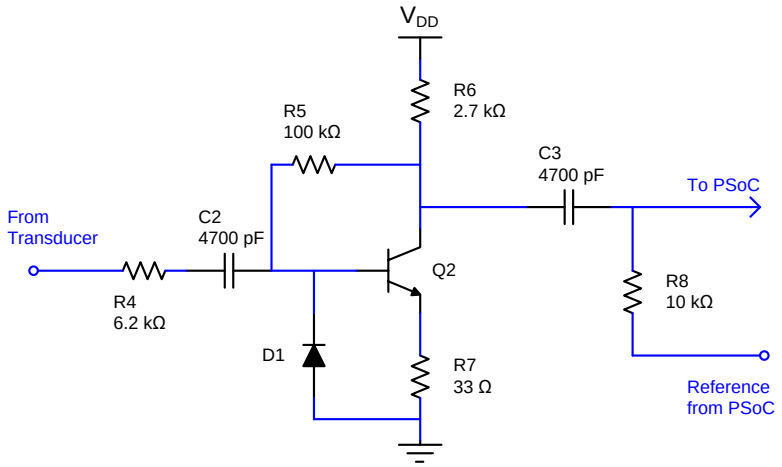
During the signal burst phase, when the transducer is being driven, a very large voltage signal is driven on the transducer. This signal is about 100 VP-P, and it is centered at 0 V. During the small-signal measurement phase, the signal can be very small (around 1 mVP-P), and it is also centered at 0 V.

This circuit is basically a common-emitter BJT amplifier with an RC high-pass filter at the input and the output. The RC high-pass filters are used to level-shift the signals to the desired DC level.

Diode D1 is needed to protect transistor Q2. Because the high-voltage signal is centered at 0 V, the base-to-emitter junction of the transistor would be reverse-biased with a very large voltage when the transducer signal is negative. This would cause the transistor to be destroyed because NPN BJT transistors cannot tolerate large base-to-emitter reverse voltages. Therefore, diode D1 is used to clamp the base voltage of the transistor to 0 V.

Resistor R4 is a key component. During the high-voltage signal burst phase, this resistor increases the input impedance of this circuit. If it is too small, then too much current will flow in and out of this circuit, and not as much power will be delivered to the transducer. During the small-signal measurement phase, this resistor acts as an output impedance of the transducer. Therefore, if it is made too large, the signal will be attenuated too much and it will be difficult to measure. Therefore, you need to balance two factors: providing high enough input impedance to the circuit during the burst phase and low enough output impedance to the transducer during the measurement phase.

Figure 6. Example Signal Preconditioning Circuit



Capacitor C2 combines with R4 to level-shift the signal to the bias voltage of the amplifier. Resistor R5 is used to bias transistor Q2 in its saturation region. Resistors R6 and R7 are used to control the gain of the amplification stage. Finally, capacitor C3 and resistor R8 are used to level-shift the signal again to be biased to the analog reference provided by the PSoC.

Diode D1 clamps to 0 V the negative half of the high-voltage signal during the burst phase. The positive half of the signal is clamped to the VDD power supply because the output of the BJT amplifier cannot rise above its supply voltage. Therefore, the voltage at the pin of the PSoC will never be below 0 V or above the VDD supply voltage. As a result, that pin is protected from high-voltage signals.

All of the components that are shown in Figure 6 make up the signal preconditioning circuit. Therefore, one instance of this circuit must exist for each transducer channel.

Internal Circuits

Figure 4 shows a number of blocks integrated inside the PSoC, with the implementation of each described below.

40 kHz Generator

The 40 kHz generator is easy to implement in a PSoC device by using a Timer8, Counter8, or PWM8 User Module (UM). Each user module can generate a 40 kHz square wave output with a 50-percent duty cycle. Each of these UMs uses one digital block resource in the PSoC device. Therefore, the 40 kHz generator uses one digital block when it is operational.

Digital Signal De-multiplexer

The digital signal de-multiplexer is also easy to implement. Because the digital interconnect routing in PSoC devices is flexible, it allows the output signal of a digital block to be routed to any pin. A signal route can be set up in the Chip View for the project. However, changing the signal routing during application runtime requires the registers that control the digital interconnect to be written. In this case, a Digital Block Output Register (DxBxxOU) and the Row Digital Interconnect Row Output Registers (RDIxROx) must be modified by application code to change the signal routing at runtime. Please refer to PSoC® TRM to learn more about these registers.

Analog Signal Multiplexer

The analog input multiplexer also is easy to implement. PSoC devices have dedicated analog input multiplexers. In general, at least eight analog signals can be muxed. In general, the analog input muxes are always connected to the Port0 GPIO pins of the PSoC. In this implementation, the Analog Input Select Register (AMX_IN) is used to change the analog muxes at runtime.

Programmable-Gain Amplifier (PGA)

The PGA is implemented using a PGA User Module in the PSoC. This user module uses a continuous-time analog block resource to implement an amplifier that has a programmable gain. For this application, try to have as much gain as possible. Therefore, the PGA User Module is always set with its maximum gain of 48.

40-kHz Band-Pass Filter

The 40-kHz band-pass filter is implemented with a BPF2 UM. This UM uses two switched-capacitor analog blocks in the PSoC device to implement a two-pole band-pass filter. The BPF2 UM has a wizard to help you set the center frequency to 40.3 kHz, the bandwidth to 5 kHz, and the gain to 2.5 (or 8 dB).

The band-pass filter is a key component of the analog signal chain. It provides extra gain to the signal, filters out noise above and below the nominal ultrasonic frequency, and eliminates the DC offset voltage of the signal caused by the high-gain PGA.

Reference

PSoC devices have on-chip analog references that can be used inside the device and driven out of the device for use by the external circuitry. For this application, the analog reference used is the VDD voltage divided by two. This voltage is used as the “analog ground” (AGND) inside the device. All analog signals get referenced to this voltage. This AGND reference is also driven out of the device through an analog output buffer, which allows the external analog signal to be biased to the on-chip analog reference. Please refer to “Reference from PSoC” in Figure 6.

Having an internal analog reference is beneficial because it requires fewer external components, less system current consumption in many cases, and better performance than a passive external reference circuit.

Comparator

The comparator in the system is used to compare the amplified and filtered signal with a voltage based on the internal reference. When the signal amplitude crosses the comparator’s reference voltage, the comparator output asserts, which causes the timer to stop, and an interrupt is generated.

The comparator is implemented with a COMP UM that occupies a continuous-time analog block in the PSoC device. This analog block has a resistor ladder used to generate a configurable voltage reference for the comparator. Therefore, it is easy to reconfigure the comparator’s reference voltage at compile-time or runtime to make the system either more or less sensitive.

16-Bit Timer

The 16-bit timer in the system measures the time it takes for the return echo signal to be detected. More than eight bits of resolution is recommended so that the distance to the object can have a high resolution. The timer implemented is an 8-bit hardware timer with an accompanying software counter implemented in the timer’s interrupt service routine (ISR). The 8-bit hardware timer and the 8-bit software counter yield a total resolution of 16 bits. During the signal measurement phase, the hardware timer runs freely. Each time a terminal count event occurs, an interrupt is generated and the ISR is executed. In this ISR, the software counter is incremented once. When the comparator interrupt occurs, the hardware timer is triggered to capture its current value. Then, the ISR of the comparator’s interrupt stops the hardware timer, reads the captured 8-bit hardware timer value and the 8-bit software counter value, and combines them to form a 16-bit value of time.

The 8-bit hardware timer is implemented in the PSoC device using a Timer8 UM.

Central Processing Unit (CPU)

The CPU manages the hardware components in the system and acts as a state machine for executing the measurement phase of the system.

Communication Interface

The communication interface used in the example project is I2C. This is implemented using the EzI2Cs UM.

Although I2C is used in the example project, the communication interface could be any interface supported by the PSoC device. For example, the PSoC device could instead use LIN, SPI, UART, or some proprietary interface.

Because the communication interface does not directly relate to the ultrasonic application, it is not discussed in detail in this application note. However, you must ensure that the nature of the communication interface does not interfere with the distance measurements. For example, do not use a communication interface that needs many ISRs or very long ones that could occur at any time. The reason is that the ISRs could interfere with the distance measurement interrupts and corrupt the distance measurement results.

Application Firmware

This section discusses the firmware in the associated example project used to implement the ultrasonic distance measurements.

The code in the example project has many comments, and the code can be inspected to see how each of the processes described in this document is accomplished.

Note Download AN76530.zip from the web page of this Application note for example project.

Distance Measurement Firmware

The application firmware for doing basic ultrasonic distance measurements is the primary goal of this application note. The distance measurement firmware is contained in the MeasureDistance() function in the example project. This function can be called to execute one distance measurement. The function returns an unsigned 16-bit value that corresponds to the time that passes before an ultrasonic echo signal is detected.

Figure 7 shows the sequence of actions that execute one distance measurement using one ultrasonic transducer. The figure offers a description and discussion of the processes.

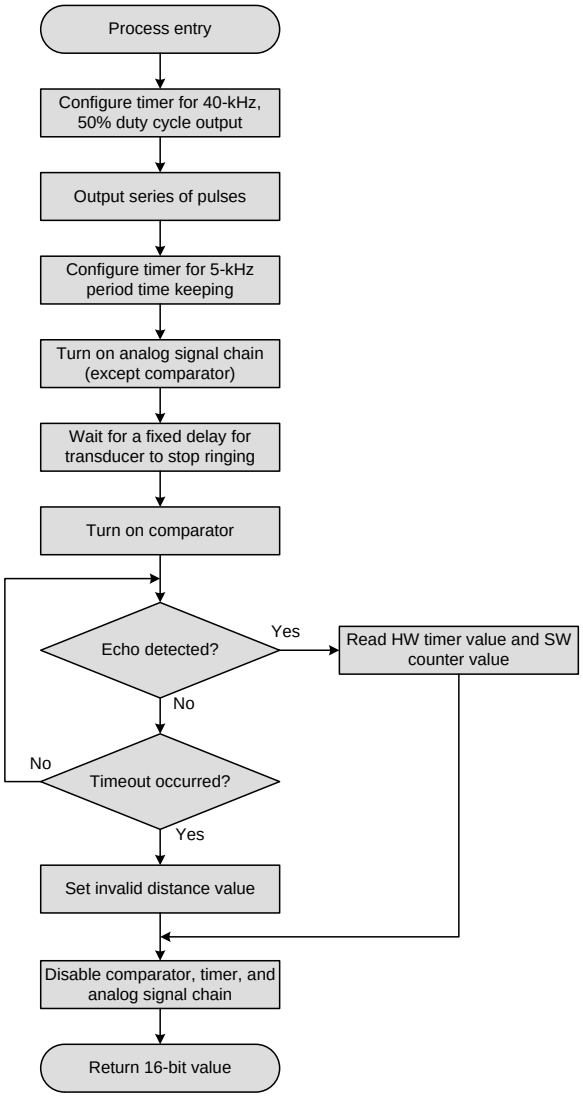
Measurement Initialization

- The measurement initialization part of the firmware is made up of the following processes from Figure 7:
- Configure timer for 40 kHz, 50-percent duty-cycle output
- Output series of pulses
- Configure timer for 5 kHz period time keeping
- Turn on analog signal chain (except comparator)
- Wait for fixed delay for transducer to stop ringing
- Turn on comparator

Note that the entire distance measurement process should never take more that 20 ms to complete. A maximum distance value of 3.0 m and a speed of sound value of 345 m/s substituted into Equation 1 cause the solution for the time to be 17.4 ms. Similarly, there is always a minimum amount of time the process will take. So, it is clear that the amount of time spent executing the `MeasureDistance()` function can vary widely.

The timing of all processes in Figure 7 is achieved using an 8-bit time-keeping variable, called PeriodCount. This variable is incremented in the terminal count ISR of the timer. Therefore, it is keeping track of how much time has elapsed, where the time base is determined by the period of the timer. This variable is used throughout the distance measurement code to keep track of how much time has elapsed during the distance measurement.

Figure 7. Distance Measurement Firmware Flowchart

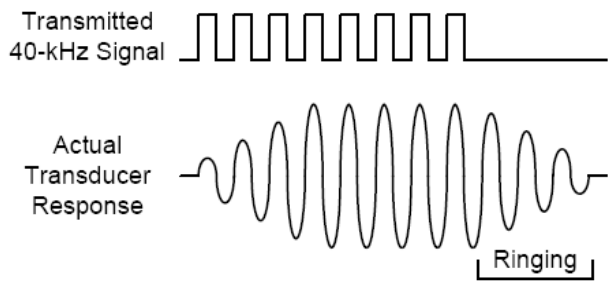


The first step in the measurement initialization phase is for the 40 kHz signals to be transmitted. These signals from the PSoC device pass through the power amplifier and are driven onto the transducer itself. To accomplish that, the timer is configured for a 40 kHz output frequency and a 50 percent duty cycle. It is then turned on and allowed to run until it has transmitted the proper number of output pulses. The PeriodCount variable is used to keep track of how many pulses have been transmitted.

Choosing the number of transmitted pulses is an important system design choice. The transducer circuit will not immediately output the full amount of sound with the very first pulse. Instead, capacitances in the circuit must be charged. Therefore, multiple pulses must be used. Generally, use just enough pulses for the transducer to reach its maximum sound output. If too few pulses are used, the transducer will never reach its maximum output level. Therefore, the range of the entire system will be reduced. The internal parallel capacitance of the transducer and the external parallel resistance are the primary factors that determine how long it takes for the transducer to reach its maximum level of output.

See Figure 8 for an example response of a transducer to a transmitted signal. Observe that in the figure, the transducer has not reached its maximum output until the fourth output pulse has completed (note that Figure 8 is only an example, and a real system would likely have different timing).

Figure 8. Transducer Response Time



Conversely, do not use more pulses than necessary because it will cause the minimum range of the system to increase. For example, assume it takes 16 pulses for the transducer to reach its maximum output level. Also, assume that it takes 1 ms for the transducer to stop ringing after it is not driven. This means that 1.4 ms will elapse before echo detection can begin (the added 0.4 ms is time it takes to transmit sixteen 40 kHz pulses). If echo detection cannot begin before 1.4 ms, then it means the minimum range would be 24.2 cm (using Equation 1). If too many pulses are transmitted (for example, 20), then it means the total amount of time before echo detection starts is now 1.5 ms. Consequently, the minimum range is now larger at 25.8 cm. So, do not transmit more pulses than necessary because doing so will increase the minimum range.

Observe in Figure 8 that there are four extra pulses transmitted; eight pulses were transmitted, though only four were needed for the transducer to reach its maximum output.

In summary, an ideal number of transmitted pulses exists for any application. Determine the number by deciding if minimum range or maximum range should be optimized. Also, analyze the properties of the transducer and the circuit to determine the time constant of the circuit in order to choose the optimal number of pulses.

In the example project, the number of transmitted pulses is configured by the I²C interface. Therefore, the number of transmitted pulses in the example project is not fixed, and it can be configured dynamically at runtime. The `PingPulses` member of the `I2cBuffer` structure is used for this purpose.

After all of the pulses are transmitted, there must be a delay before echo detection can begin. This delay allows the ringing signal on the transducer to drop to a low enough level. For an example of ringing, see Figure 8. Even though the transmitted pulses stop, there is still a period during which the transducer is still outputting sound because it is ringing. If the echo detection would start immediately, then a false echo would always be detected immediately because of the ringing signal on the transducer.

The amount of delay needed depends on a number of factors. Some transducer manufacturers specify a maximum ringing period. In that case, choosing the delay time is straightforward. If this information is not available, determine the ringing period by circuit analysis, simulation, or characterization. The delay for ringing is typically about 1 ms.

Implementing the delay period (and the time keeping for the rest of the distance measurement) is done by reconfiguring the timer for a lower frequency (5 kHz in the example project) and resetting the `PeriodCount` variable to zero. The lower frequency of the timer is necessary because generating interrupts at a 40-kHz is too much for the processor to handle if other tasks need to be done. Additionally, it would provide a time measurement resolution that is too high. The 5-kHz frequency of the timer provides a good time resolution and does not generate too many interrupts. Resetting the `PeriodCount` variable to zero at this time causes the end of the transmitted pulses to be the starting point of the distance measurement.

The ringing delay in the example project is configured using the I²C interface. The `WaitPulses` member of the `I2cBuffer` structure is used for this. Therefore, the amount of ringing delay time is configured at runtime.

During the ringing delay, the analog signal chain (except the comparator) is started up. This includes the PGA, the band pass filter, and the reference. The comparator is not started at the same time because the analog circuits need an opportunity to settle to their steady-state conditions. The analog voltages are likely not valid right after starting the analog circuits.

After the ringing delay ends, the comparator is started and its interrupt is enabled. This action marks the end of the measurement initialization phase and the beginning of the echo detection phase.

Echo Detection

The echo detection part of the firmware is made up of the following processes from Figure 7.

- Echo detected?
- Read HW timer value and SW counter value
- Timeout occurred?
- Set invalid distance value

The ISR for the comparator's interrupt does two simple things. It asserts a flag variable (called `CompleteFlag`) and it stops the hardware timer. The comparator will trip and generate an interrupt as soon as an echo signal with sufficient magnitude travels through the analog signal chain and arrives at the comparator. The trip of the comparator causes the ISR to execute, which will stop the timer from measuring time and assert the flag that lets the mainline code know that the echo was detected.

The code in this section repeatedly checks the `CompleteFlag` variable to see if the echo has been detected. If the echo has been detected (the `CompleteFlag` is asserted), then the `PulseCount` variable and the timer counter register are both read and concatenated into a 16-bit value. This value indicates how much time has elapsed, where every least significant bit (LSb) of the value is equal to 833 ns of time. 833 ns is the clock period of the 8-bit timer. This can be used in Equation 1 to determine that each LSb of the value is equal to 0.144 mm of distance. This distance resolution is likely much higher than needed for most applications. Therefore, lower bits of the 16-bit value can be ignored as desired.

In many cases, there will be no objects in the transducer's range, or the objects will not be large enough to be detected. In this case, an echo signal will never be detected. Therefore, it is also necessary to check for a timeout condition. Since echoes are not expected to ever be detected after 20 ms, then a 20 ms timeout period is a good choice. If 20 ms have elapsed and the CompleteFlag variable never asserted, then it means a timeout has occurred. In this case, the distance value is set to an invalid value (0x7FFF).

Before an echo detection or a timeout has occurred, it may be advantageous for the system to dynamically update the gain of the analog signal chain (including the comparator) during the echo detection phase. The reason is that the farther the sound waves travel, the lower the magnitude gets. This means that an echo that bounces off an object at 0.5 m (1.0 m of total sound travel distance) will have a much higher magnitude when it arrives back at the transducer than an echo that bounces off an object at 2.5 m (5.0 m of total sound travel distance). Therefore, the highest gain is needed near the end of the echo detection phase, and the gain near the beginning does not need to be as high.

PSoC devices are very well-suited for this technique of dynamically increasing the gain during the echo detection phase. The gain of the PGA, the band-pass filter, or the threshold of the comparator all can be adjusted. Therefore, there are many flexible options for doing this technique.

Disabling the Channel

The last process in the distance measurement firmware is to shut down the various modules used by the distance measurement code. This is an important step because the analog modules use a significant amount of current. Therefore, they should shut down when not in use so that current is not wasted. Before exiting the `MeasureDistance()` function, all of the user modules used for the distance measurement process are stopped.

After all of the modules have been stopped, the function exits and returns the 16-bit value that corresponds to the measured distance of any objects (or an invalid value if no objects were detected).

Main Application Firmware

The application code in *main.c* is simple. Most of the complex code in the example project is contained with the distance measurement application code in the `MeasureDistance()` function.

The sequence of actions executed in `main()` of the example project is shown in **Error! Reference source not ound.**.

Initialization

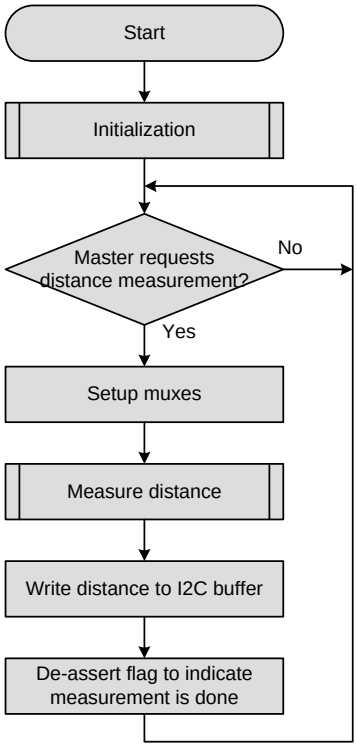
The initialization code of the system is contained in the `Node_Init()` function. The initialization code consists of the following tasks:

- Cycling through each mux/demux setting, which sets the transmit signals low
- Configuring the comparator to be completely asynchronous with no output latching
- Initializing and starting the I^2C interface
- Configuring the test mux in an analog CT block to drive the AGND voltage to the analog output bus to be used as the external reference

Main Loop

After initialization of the various systems in the device, the main loop of the firmware is executed. The firmware continually checks a flag that gets asserted by the I^2C master in the system. If this flag is asserted, it indicates to the PSoC device that it should do a distance measurement on an ultrasonic channel. The `Ctrl` member of the `I2cBuffer` structure is used for two purposes. Bit 0 of `Ctrl` is used as the flag that gets asserted to indicate that the PSoC should do a measurement. Bit 1 and bit 2 of `Ctrl` form a bit field that selects the channel on which the measurement should be done.

Figure 9. Main Application Firmware Flowchart



After the PSoC device detects that the flag in `Ctrl` has been asserted, it calls a function to set the muxes properly based on the channel selection bit field in `Ctrl`. The code that sets the muxes is found in the `SetMuxes()` function. This function properly sets the RDIxROx registers, which control the digital output path; the AMX_IN register, which controls the analog input path; and the proper DBxxxOU register, which controls the output path of the timer. You must set up all of these registers properly to ensure that the transmitted digital signal goes to the proper pin and that the received analog signal is input to the proper pin.

After all of the muxes are set up properly, then the distance measurement can be done by calling the `MeasureDistance()` function. The code in this function is described in the Distance Measurement Firmware.

After the 16-bit value has been returned from the `MeasureDistance()` function, then it is written to the `TimeArray` member of the `I2cBuffer` structure. After this, the flag in `Ctrl` is de-asserted to indicate to the master that the distance measurement has finished and the distance value is available in the I^2C buffer to be read. After this step, the whole process is repeated.

All of the actual I^2C communication is handled in the background in ISRs of the I^2C module's interrupt. For more details on I^2C communication, see application note AN50987 or the datasheet for the EzI2Cs User Module (which is the UM used in the example project). Ultrasonic transducer frequency other than 40 KHz can also be used, but this will require the following:

- Change in frequency generator in MCU
- Designing the power amplifier circuit based on selected transducer
- Change in design of Band pass filter in MCU as per selected frequency

Summary

We described the hardware and firmware design details of a UPA distance measurement application based on PSoC 1 devices. In so doing, we explored major system parameters and design tradeoffs. This application note is a good starting point if you are developing an ultrasonic distance measurement system that uses enclosed, ultrasonic transceiver sensors.

About the Author

Name:	Valeriy Kyrynyuk, Ben Kropf, Balaji M V
Title:	Systems Engr Principal, Applications Engineer Staff, Applications Engineer Sr

Appendix A: Schematic of the Hardware

Note Please download AN76530.zip from the web page of this Application note for Schematic and Gerber files.

Figure 10. PSoC and Interface Section Schematic

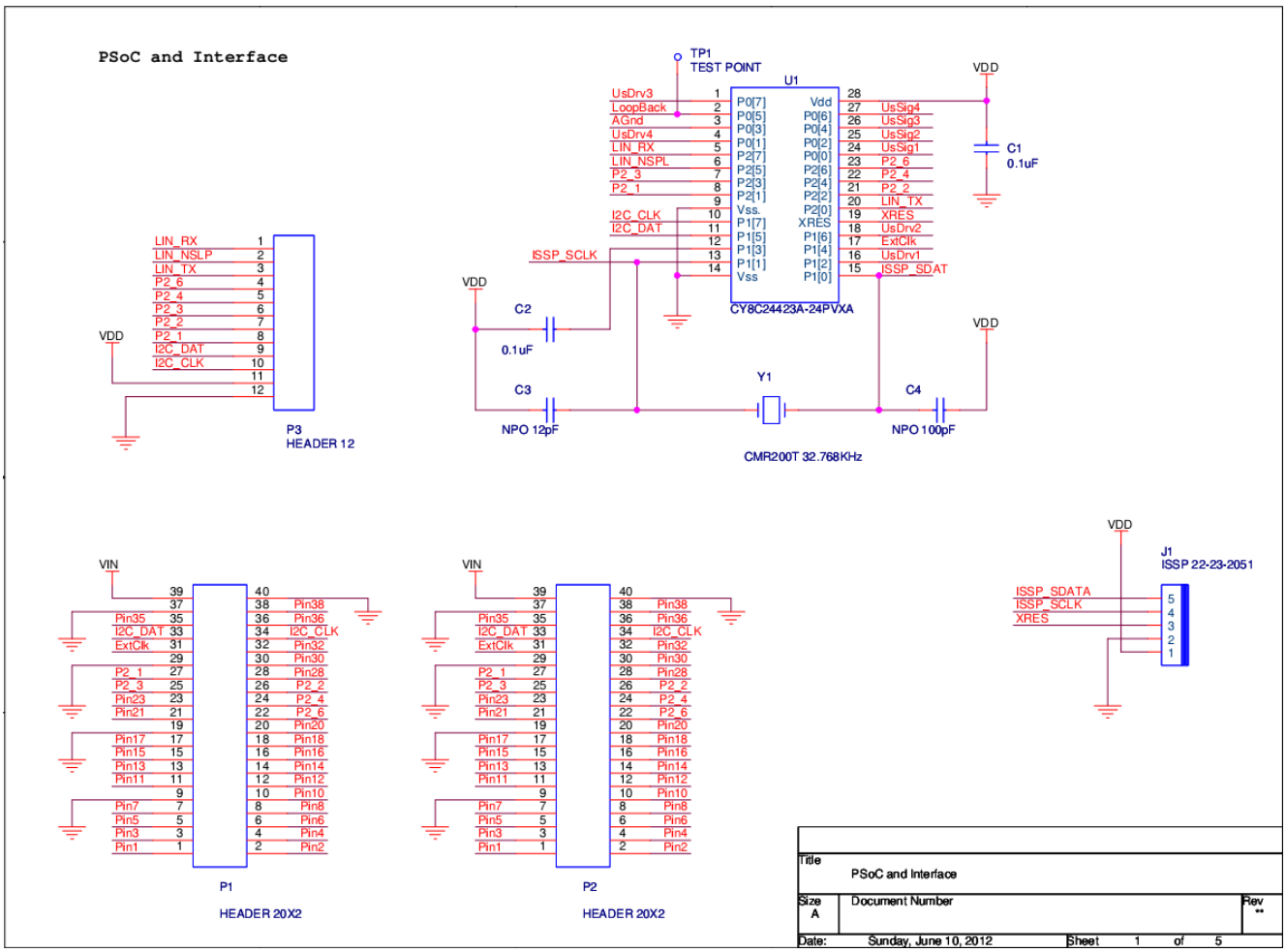


Figure 11. Power Supply Section Schematic

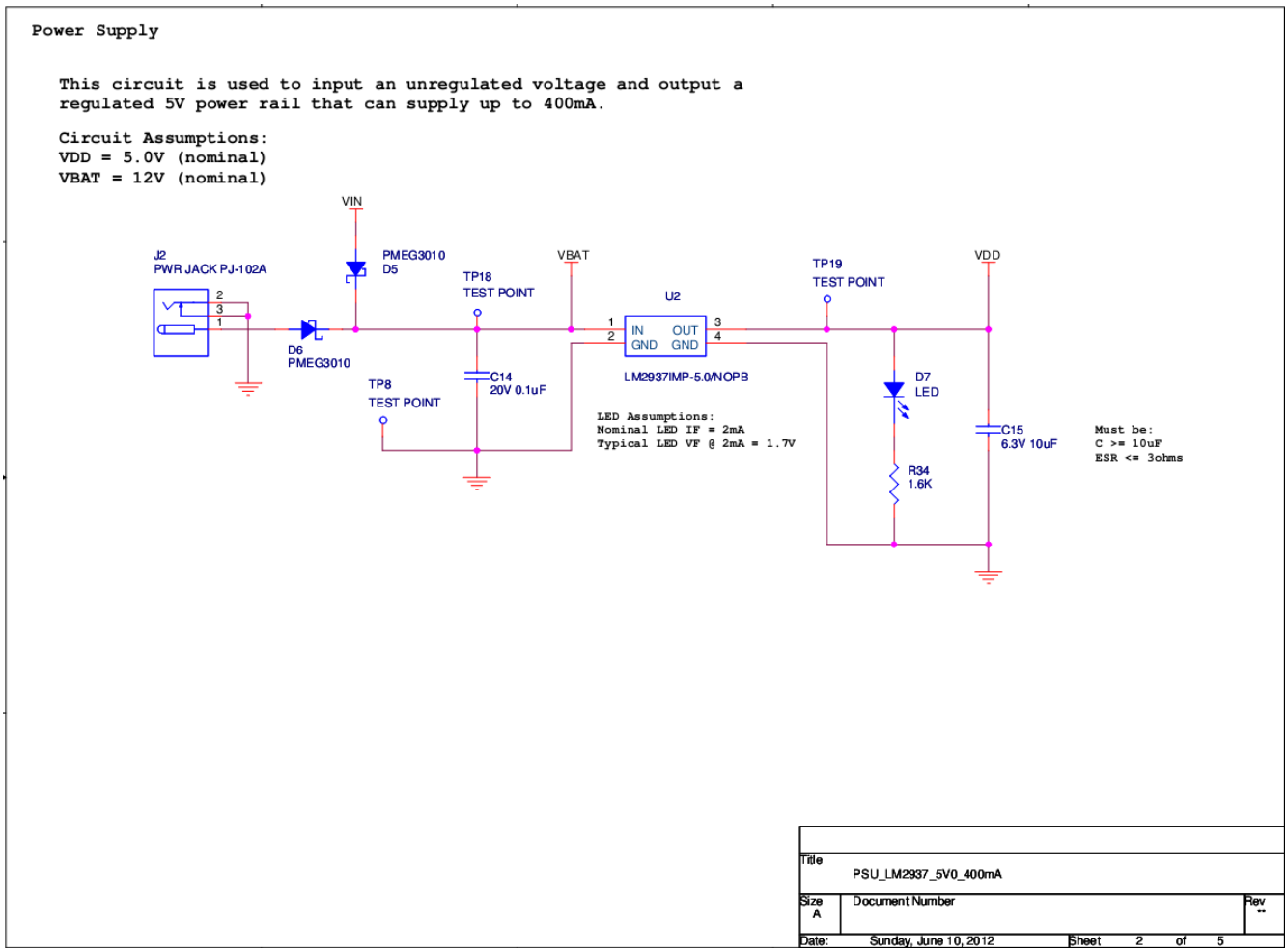


Figure 12. Power Amplifier Section Schematic - Part1

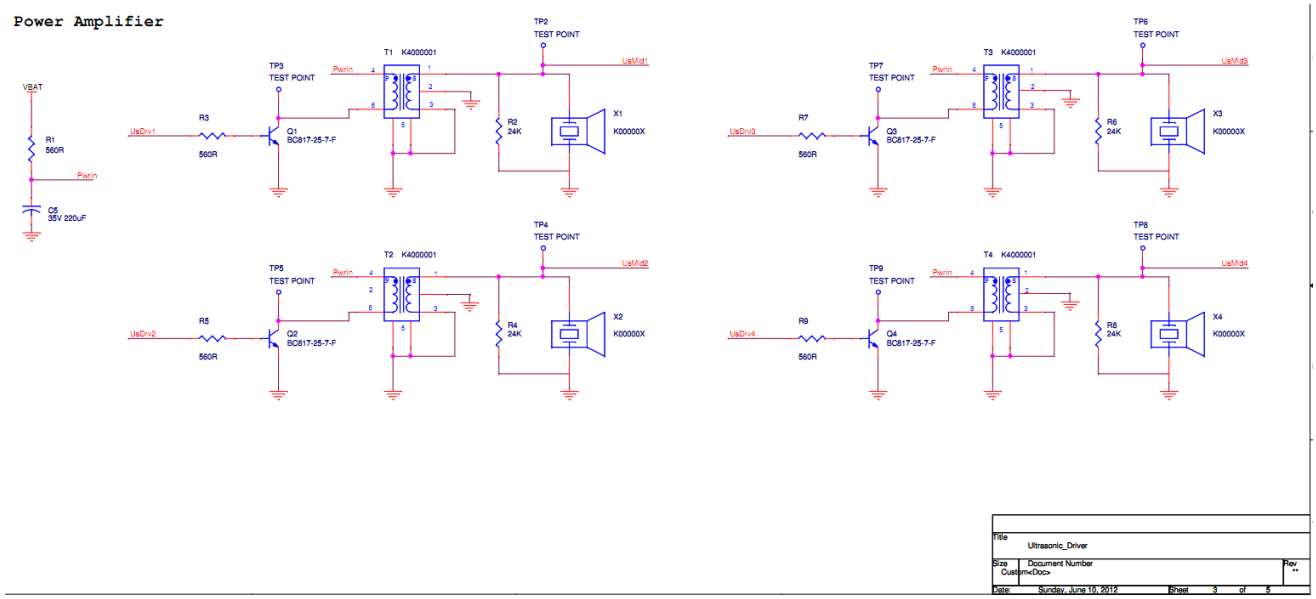


Figure 13. Power Amplifier Section Schematic - Part 2

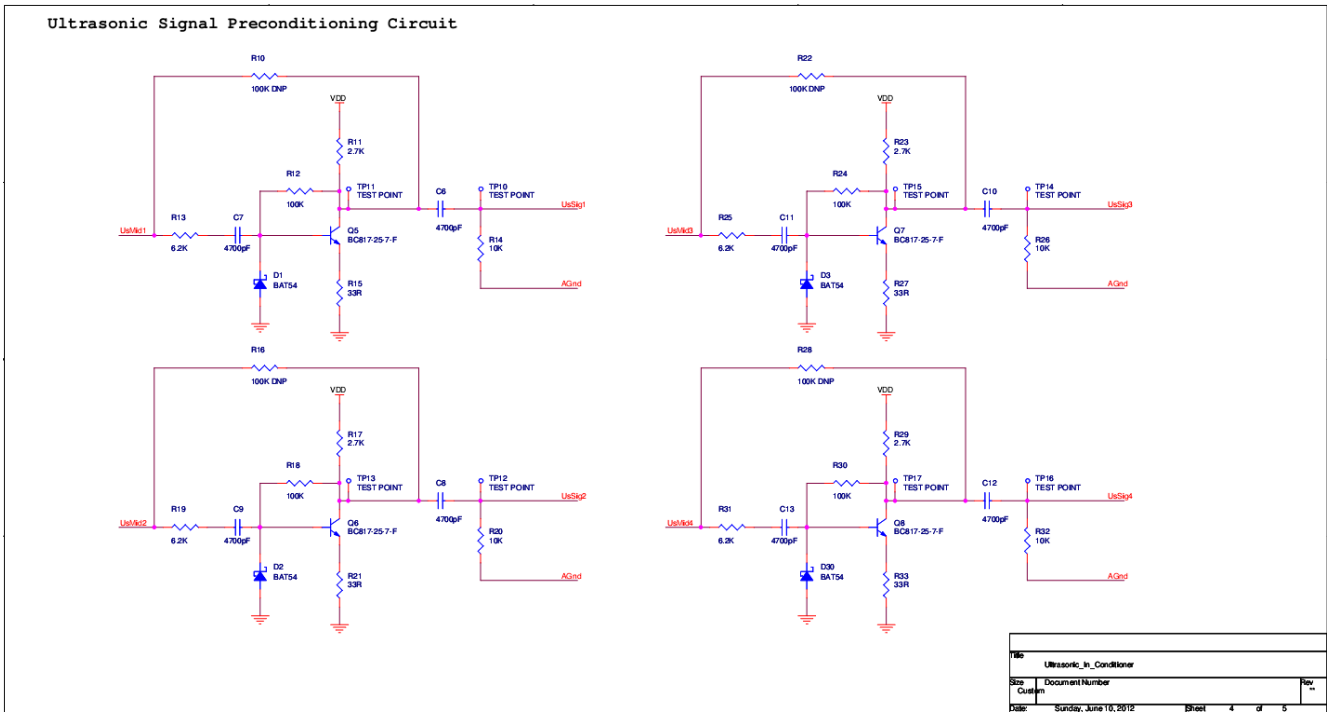
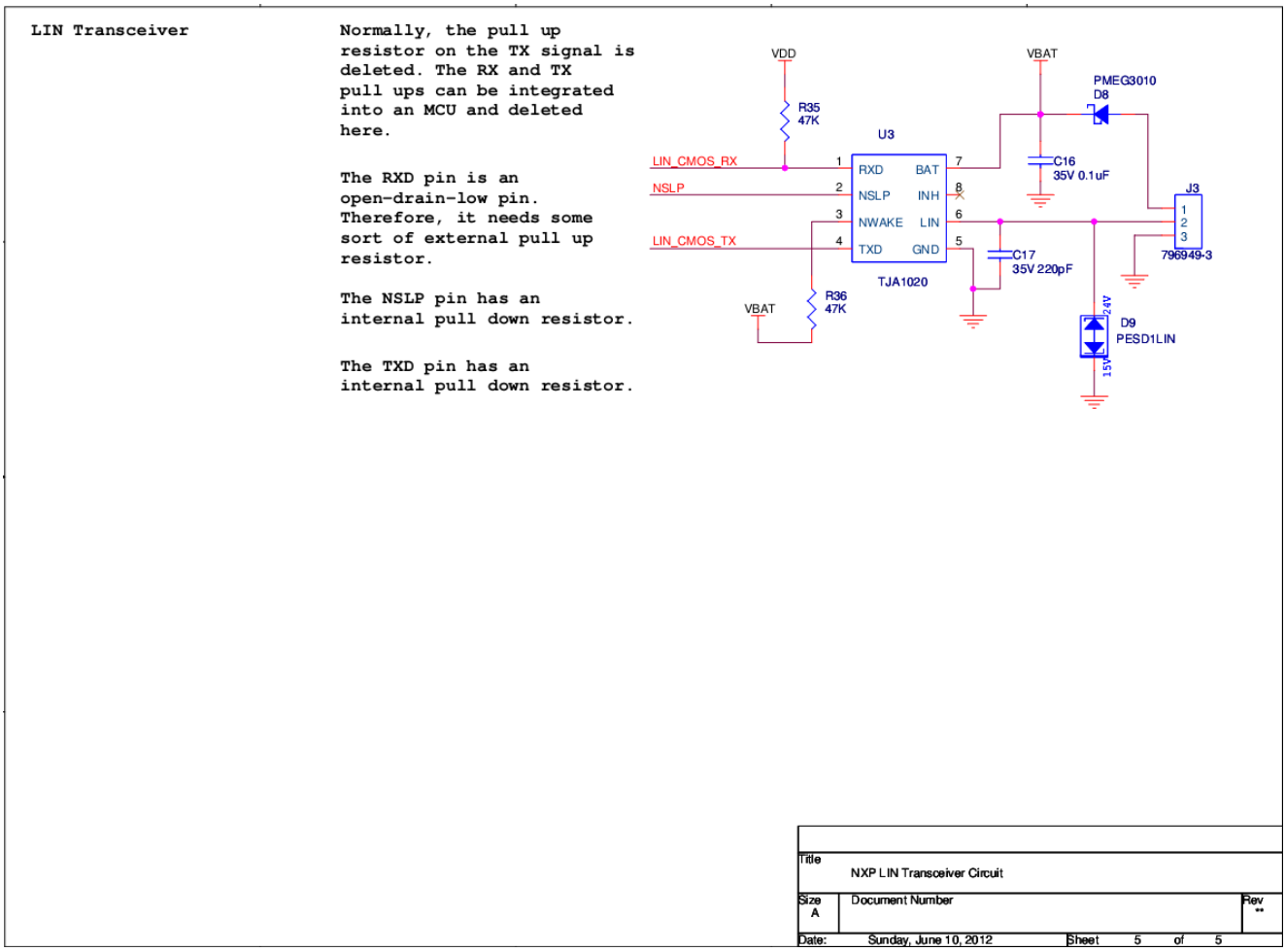
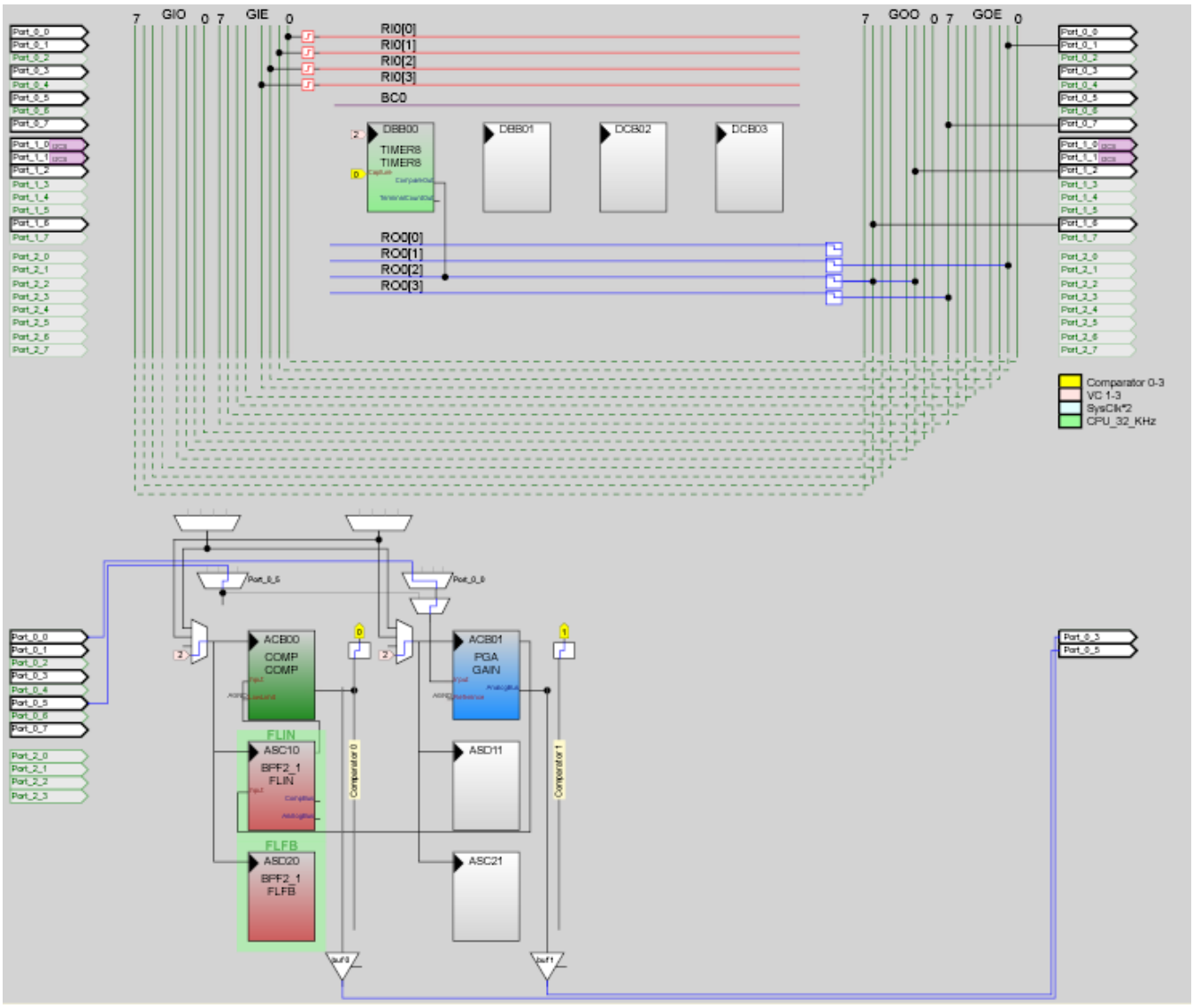


Figure 14. Lin Transceiver Section Schematic



Appendix B: Example Project: User Module Placement

Figure 15. User Module Placement in PSoC Designer



Note Please download AN76530.zip from the web page of this Application Note for example project.

Document History

Document Title: AN76530 - PSoC® 1 Automotive Ultrasonic Distance Measurement for Park-Assist Systems

Document Number: 001-76530

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	3608615	BALM	06/19/2012	New Spec.
*A	3889184	KUK	01/22/2013	AN title changed from PSoC® 1 Automotive Ultrasonic Distance Measurement to PSoC® 1 Automotive Ultrasonic Distance Measurement for Park Assist Systems; added CY8C28xxx part family reference; Added AN2219 reference for related application notes. Provided characteristics of transducer in Maximum range section. Added note "Note: Please download AN76530.zip from the web page of this Application note for example project" under Application Firmware Section. Added User module placement in Appendix B.
*B	4413579	KUK	06/19/2014	Updated Project to PSoC Designer 5.4. Updated Schematics to reflect transformer pin numbers.
*C	4669781	KUK	02/24/2015	Corrected typo in distance calculation, in Echo Detection. Added “833 ns is the clock period of the 8-bit timer” in Echo Detection. Add information on using ultra sonic transducer of different frequency in Main Loop.
*D	5774474	AESATMP8	06/15/2017	Updated logo and Copyright.

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

ARM® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6

Cypress Developer Community

Forums | WICED IOT Forums | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

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