Application Report

SLAA941–March 2020

TUSS44x0 Software Development Guide

ABSTRACT

The objective of this guide is to explain the high-level software flow for platform development with the TUSS4440 and TUSS4470 devices using serial peripheral interface (SPI), general-purpose input/output (GPIO), and analog-to-digital converter (ADC) commands. This document contains a software development flow-chart, main executable routines, and source code examples. The example software is written in Energia for the MSP-EXP430F5529LP, but can be adapted to any TI LaunchPad™ development kit.

Contents

1 Introduction to High-Level Software Flow 2
2 Main Energia Sketch File 3
3 Energia Examples 8
4 References 10

List of Figures

1 High-Level Software Flowchart for TUSS44x0 Operation 2
2 Detailed Software Sequence Flowchart 3
3 MCU Initialization 3
4 TUSS44x0 Configuration 5
5 Read Ultrasonic Time-of-Flight 7
6 TUSS44x0_Ultrasonic GetDistance.ino Results 8
7 TUSS44x0_Ultrasonic GetDistance.ino Registers and Bit Fields 9
8 TUSS44x0_Ultrasonic VOUT_ADC_Processing.pde Processing GUI 9

Trademarks

1 Introduction to High-Level Software Flow

The TUSS44x0 device can only be operated as a slave device and must be paired with an external microcontroller unit (MCU), which acts as the master device. The master device is responsible for the initialization, configuration, and regular polling operation of the TUSS44x0 device. Figure 1 shows the high-level overview of the software flow for standard TUSS44x0 operation. There are three main components of the software flow:

- The main Energia sketch file
- The TUSS44x0 header and driver files
- The master controller header and driver files

After master controller initialization, the program of the main file loops the routine shown in Figure 1 to report the ultrasonic time-of-flight result based on the response from the TUSS44x0 device. For the purpose of this guide, the TUSS44x0-specific code is referenced from the TUSS44x0 Energia IDE library package containing the *GetDistance.ino*, *TUSS44x0_Ultrasonic.cpp* and *TUSS44x0_Ultrasonic.h* files. The master controller used in this example is the MSP430F5529 MCU on the MSP-EXP430F5529LP platform. The TUSS44x0 EVM hardware (BOOSTXL-TUSS44x0) is used to demonstrate the operation of the software.

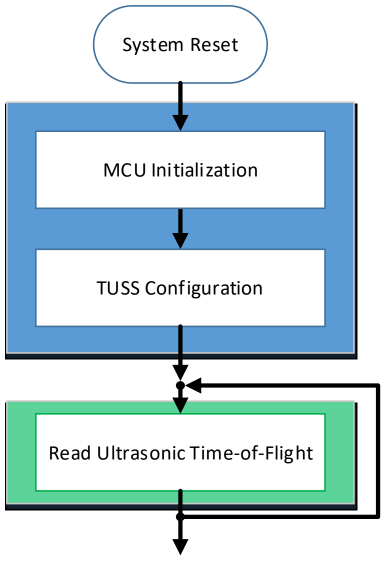


Figure 1. High-Level Software Flowchart for TUSS44x0 Operation

2 Main Energia Sketch File

The main file hosts the high-level code and routines required to initialize the master controller and TUSS44x0 device, and run TUSS44x0-specific operations. Figure 2 shows the software flowchart sequence and the mapping of device specific functions.

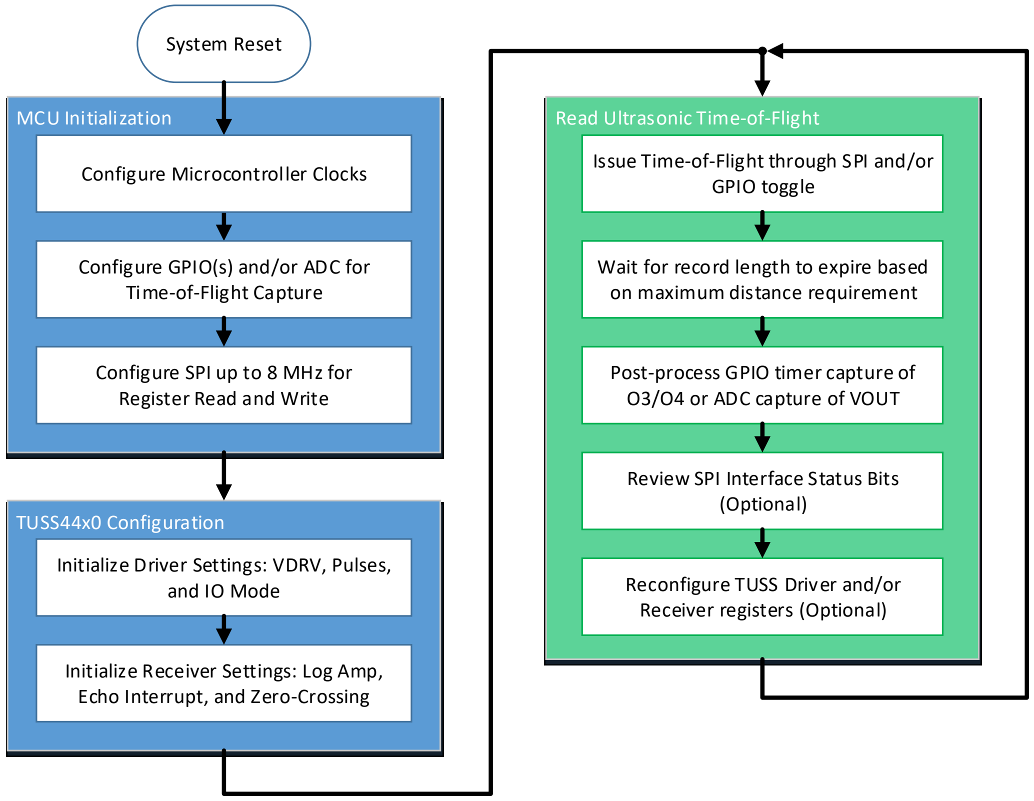

System Reset
MCU Initialization
Configure Microcontroller Clocks
Configure GPIO(s) and/or ADC for Time-of-Flight Capture
Configure SPI up to 8 MHz for Register Read and Write
TUSS44x0 Configuration
Initialize Driver Settings: VDRV, Pulses, and IO Mode
Initialize Receiver Settings: Log Amp, Echo Interrupt, and Zero-Crossing
Read Ultrasonic Time-of-Flight
Issue Time-of-Flight through SPI and/or GPIO toggle
Wait for record length to expire based on maximum distance requirement
Post-process GPIO timer capture of O3/O4 or ADC capture of VOUT
Review SPI Interface Status Bits (Optional)
Reconfigure TUSS Driver and/or Receiver registers (Optional)

Figure 2. Detailed Software Sequence Flowchart

2.1 MCU Initialization

The internal clocks, SPI ports, GPIOs, and ADC of the master controller must be configured prior to initialization of the TUSS44x0 registers.

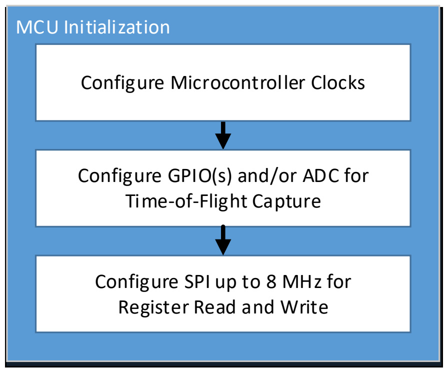

MCU Initialization
Configure Microcontroller Clocks
Configure GPIO(s) and/or ADC for Time-of-Flight Capture
Configure SPI up to 8 MHz for Register Read and Write

Figure 3. MCU Initialization

The SPI terminal of the master controller must be configured to meet the TUSS44x0 requirements for SDI, SDO, SCLK, and NCS pins. The SPI supports up to an 8 MHz clock, and is formatted in a 16 data bit frame with MSB first. The master SPI should match SPI MODE 1 (CPOL = 0 and CPHA = 1), such that the SDO is sampled on the falling edge of the SCLK pin, and the SDI is shifted out on the rising edge of the SCLK pin. The SPI logic level of the TUSS44x0 device is equal to the value of V_{VDD} that can be set to a value between 3.1 V and 5.5 V. This means traditional 3.3 V or 5.0 V logic level MCUs can interface with the TUSS44x0. Because the logic level is typically limited to a single voltage, and fixed for most systems, the default logic level is referenced to 3.3 V for compatibility with the MSP430F5529 microcontroller.

Depending on the I/O driver mode used by the TUSS44x0, either one or two MCU GPIOs must be configured as outputs to drive the IO1 and IO2 pins. When using IO_MODE1 or IO_MODE2, two GPIOs must be configured as outputs for both IO1 and IO2 pins. When using IO_MODE0 or IO_MODE3, only one GPIO must be configured as an output for pin IO2; pin IO1 is not used in these cases.

Depending on the type of data response from the TUSS44x0 that the MCU is to capture, one or two GPIOs must be configured as inputs, or one ADC input capture is to be enabled.

2.2 TUSS44x0 Configuration

Configuration of the TUSS44x0 registers settings requires a set of individual SPI write commands, but the microcontroller must calculate the SPI frame odd parity bit calculation to successfully perform any SPI read or write command.

2.2.1 TUSS44x0 Odd Parity Calculation

The parity bit ensures that the total number of 1-bits in the string is even or odd. Accordingly, there are two variants of parity bits: even parity bit and odd parity bit. In the case of even parity, for a given set of bits, the occurrences of bits whose value is 1 are counted. If that count is odd, the parity bit value is set to 1, making the total count of occurrences of 1s in the whole set (including the parity bit) an even number. If the count of 1s in a given set of bits is already even, the parity bit's value is 0. In the case of odd parity, the coding is reversed.

The frame odd parity bit value is required and used to ensure that data communicated between the master controller and the TUSS44x0 device has not been compromised or corrupted when transmitted or received. The odd parity bit value is generated by both the master and slave devices, and is included as bit 8 of the 16 bit frame. The odd-parity bit is always assumed to be a value of 0h prior to performing the odd parity bit calculation on the entire frame.

The master-to-slave (MCU to TUSS4470 over the SDI line) frame from MSbit to LSbit is: 1 RW bit, 6 bits for the register address, 1 ODD parity bit for entire SPI frame, 8 bits for data.

Odd Parity Bit Calculation Examples:

- To read BPF_CONFIG_1 (address 0x10), the 16-bit SPI frame to read this register will be 0xA100, where 0xA1 in binary is 10100001:
  - 1 = Read High / Write Low bit
  - 010000 = Register Address
  - 1 = ODD Parity bit

  In this case, there are only two bits set to '1' due to the read high bit and the register address, so the parity bit is set to '1' to create an odd number of '1' bits in the entire 16-bit frame.
- To write a value of 0x11 to BPF_CONFIG_1, the 16-bit MOSI SPI frame is 0x2011, where 0x20 in binary is 00100000:
  - 0 = Read High / Write Low bit
  - 010000 = Register Address
  - 0 = ODD Parity bit

  and 0x11 is the data byte. In this case there at already three (odd number) bits set to '1' due to the register address and the data byte, so the ODD parity bit remains '0'.

2.2.2 TUSS44x0 Settings

All TUSS44x0 register settings are stored in volatile memory; therefore, the registers will always default to the TI factory values at start-up. Due to the lack of EEPROM, it is not possible to permanently store any device settings on the TUSS44x0, so the microcontroller must configure the device after every power-up or power-cycle.

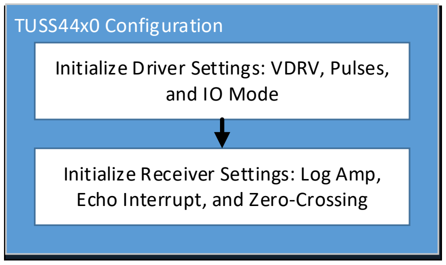

TUSS44x0 Configuration
Initialize Driver Settings: VDRV, Pulses, and IO Mode
Initialize Receiver Settings: Log Amp, Echo Interrupt, and Zero-Crossing

Figure 4. TUSS44x0 Configuration

The TUSS44x0 device only requires that the configuration registers be written to at least once upon start-up to properly enable the device to execute a time-of-flight. The registers to configure can be separated into two categories: driver settings and receiver settings.

2.2.2.1 Driver Settings

If the driver voltage is to be generated internally by the TUSS44x0, then the VDRV_HI_Z bit field must be set to 0h to enable VPWR to VDRV regulation. Ensure the equivalent voltage of VDRV_VOLTAGE_LEVEL bit field is less than or equal to the voltage at the VPWR pin. If VDRV_VOLTAGE_LEVEL is greater than the voltage at VPWR, the SPI Interface Status Bit will report an error that the VDRV power regulator has not reached the programmed voltage level. If VDRV is sourced externally, keep the VDRV_HI_Z bit field set to 1h to disable internal VPWR to VDRV regulation.

The default number of burst pulses at the BURST_PULSE bit field is set to 0h, which allows for continuous pulsing. TI recommends to change this value to a specific number to prevent additional unintentional pulses that may damage the transducer. For applications that prioritize short ranging, using a low number of burst pulses such as 1 to 4 is recommended. Some transducers have a maximum number of burst pulses rating, which is typically 16 to 20.

Choose an IO_MODE based on the MCU's capabilities or user preference. TUSS44x0 has multiple modes to excite the transducer through the OUTA and OUTB pins. For each of the modes, the desired frequency of burst is supplied through an external clock or GPIO output on the IOx pins. This enables the user to supply a highly precise clock or GPIO output calibrated to the center frequency of transducer to enable the highest sound pressure level generation. These modes can be selected by the IO_MODE bit field in the DEV_CTRL_3 register. When IO_MODE = 0h, there is a SPI dependency to start a time-of-flight cycle. All other IO_MODEs only use the IOx pins to start a time-of-flight cycle.

The TUSS4440 current limit control block set by the XFMR_DRV_ILIM bit field drives current efficiently into the primary side of the transformer to achieve the maximum swing (set by voltage on the center tap and turn ratio of the transformer) on the secondary side. By default, the current limit is set to lowest value of 50 mA, which may be suitable for short range applications. To increase the maximum detectable range, increasing the current limit to a nominal 300 mA may be required for transducers with a larger driver voltage requirement.

2.2.2.2 Receiver Settings

The TUSS44x0 band-pass filter center frequency in the BPF_HPF_FREQ bit field must be set to equal the center frequency of the transducer. For more information, see the *Bandpass Filter Center Frequency Configuration* table in the *TUSS4470 Direct Drive Ultrasonic Sensor IC With Logarithmic Amplifier Data Sheet*. This value is centered to 40.64 kHz by default. The band-pass filter mode is only compatible up to 500 kHz. For transducers with a center frequency above 500 kHz and up to 1 MHz, use the high-pass filter mode by settings the BPF_BYPASS bit field to 1h to bypass the band-pass filter. Once in high-pass filter mode, set the value of BPF_HPF_FREQ to 1Fh for a high pass filter corner frequency of 400 kHz.

The logarithmic amplifier has two controls: slope and intercept. These settings will change how the log amp output responds as a function of input amplitude. In general, as the log amp slope increases, the log amp output gain increases, and as the log amp intercept decreases, the log amp output gain increases. By default, the log amp settings are set to nominal values. To override the nominal settings, set the LOGAMP_FRC bit field to 1h to enable the use of the LOGAMP_SLOPE_ADJ and LOGAMP_INT_ADJ bit fields. For more details on the log amp, see the *Logarithmic Amplifier for Ultrasonic Sensor Signal Conditioning*. The low-noise amplifier has a fixed gain, and can be set by the LNA_GAIN bit-field.

The TUSS44x0 is an analog front-end device, meaning the output is analog signal that has been pre amplified and filtered. More specifically, the TUSS44x0 output at the VOUT pin has been amplified by a low-noise amplifier, band-pass or high-pass filtered, and logarithmically amplified before becoming available as a demodulated signal to an external system. The VOUT signal can be captured by an ADC for additional post-processing by an MCU. The VOUT output can be scaled to a 3.3 V or 5.0 V referenced ADC by setting the VOUT_SCALE_SEL bit field to 0h or 1h, respectively.

The TUSS44x0 includes a built-in echo interrupt and zero-crossing intelligence in the event the VOUT signal can not be captured or does not need to be captured. The echo interrupt signal is available on the OUT4 pin that goes high when the signal on the VOUT pin crosses a threshold as defined by the ECHO_INT_THR_SEL bit field. This function is disabled at device power up, but can be enabled by setting the ECHO_INT_CMP_EN bit field. As long as the VOUT signal is higher than this threshold, the echo interrupt signal is held high. The signal goes low asynchronously when the VOUT signal drops below the programmed threshold. This signal can be used to interrupt a MCU when an object has been detected. The threshold value is also dependent on the setting of the VOUT_SCALE_SEL bit field. This echo interrupt features removes the need for an ADC if the post-processing is intended to perform the same function to determine the time-of-flight.

A zero-crossing signal is output at the OUT3 pin, which can be used to validate the frequency of the received echo signal to provide robustness against interference from other signals. This zero-crossing signal is derived from the raw amplified input signal from a particular stage as it is being demodulated in the log amp block. This function is disabled at device power up, but can be enabled by setting the ZC_CMP_EN bit field. When enabled, the ZC_CMP_STG_SEL bit field is used to select which log amp gain stage is used to generate the zero crossing signal while the ZC_CMP_HYST bits control the hysteresis of the zero-crossing comparator. The stage selection to see the OUT3 pin toggling depends on the strength of signal received by the log amp and has to be configured depending on the application. For large amplitude of input signal, a lower stage of the log amp should be selected. For lower amplitude of input signal, a higher stage should be selected. To avoid switching noise generated by the toggling of the zero-crossing comparato,r when the ZC_EN_ECHO_INT bit is set, the zero-crossing output will only be enabled while the echo interrupt signal is high.

2.3 Read Ultrasonic Time-of-Flight

After the TUSS44x0 registers have been updated, the system can now continuously run a time-of-flight command and extract the resulting measurement data from the VOUT, O3, and O4 pins to compute and convert time-of-flight to distance or speed-of-sound. The SPI Interface Status Bits can be reviewed between each time-of-flight measurement to determine if the driver block is behaving as expected. The TUSS44x0 can be updated dynamically, so if any of the device settings are changed between time-of-flight cycles, the MCU is able to do so.

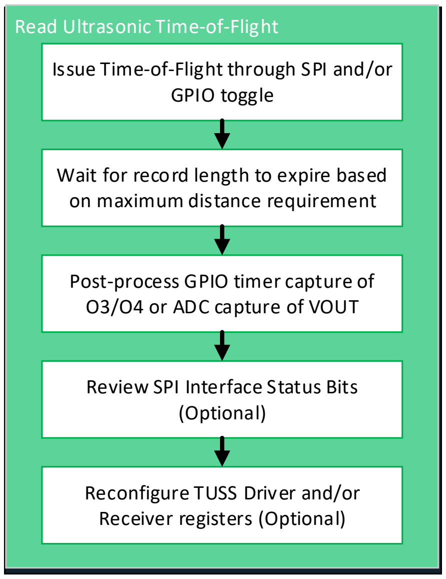

Read Ultrasonic Time-of-Flight
Issue Time-of-Flight through SPI and/or GPIO toggle
Wait for record length to expire based on maximum distance requirement
Post-process GPIO timer capture of O3/O4 or ADC capture of VOUT
Review SPI Interface Status Bits (Optional)
Reconfigure TUSS Driver and/or Receiver registers (Optional)

Figure 5. Read Ultrasonic Time-of-Flight

2.3.1 Run Time-of-Flight

The MCU initiates a time-of-flight command by sending the appropriate SPI write and GPIO/clock toggle of the IOx pins. The MCU is able to force burst pulses at any given time because there is no preset record length in the TUSS44x0. The MCU should wait a user-defined amount of time before attempting a subsequent time-of-flight command. The delay time between time-of-flight commands is crucial to allow the ultrasonic sound wave travel and return to the sensor because the speed of sound is much slower than the majority of modern MCUs. If the MCU attempts to read the ADC, O3, and/or O4 results too quickly or immediately after the time-of-flight burst is initiated, there will be limited or no information available. Depending on the transmission medium, the speed of sound varies. For example, the speed of sound through air at room temperature is approximately 343 m/s, while the speed of sound through sea water at room temperature is 1531 m/s.

2.3.2 Post-Process Data

If the VOUT output is captured using an ADC, the MCU can apply a custom threshold time and level map to extract the return echoes time-of-flight, peak amplitude, and width. If the O3 zero-crossing pin toggles are captured using a GPIO configured as an input, the zero-crossing frequency and count can be determined. If the O4 echo interrupt pin toggles are captured using a GPIO configured as an input, the time-of-flight and echo width can be calculated for multiple objects.

2.3.3 Convert Time-of-Flight to Distance

To convert time-of-flight to distance, use **velocity = distance/time**. More specific to ultrasonic time-of-flight to distance, use $\textbf{velocity}_{\text{sound}} = \textbf{distance}_{\text{to\_object}} / \textbf{time}_{\text{of\_flight\_one\_way}}$ for one-way measurements, and $\textbf{velocity}_{\text{sound}} = \textbf{distance}_{\text{to\_object}} / (\textbf{time}_{\text{of\_flight\_round\_trip}} / 2)$ for round-trip measurements. Note that the speed of sound varies as a function of transmission medium density and temperature, so TI recommends to use an external temperature sensor to measure ambient temperature to calibrate the speed of sound variable when converting time-of-flight to distance.

For example: If the time-of-flight for a round-trip measurement is 30ms, and the transmission medium is air at room temperature, then distance is calculated as shown in Equation 1.

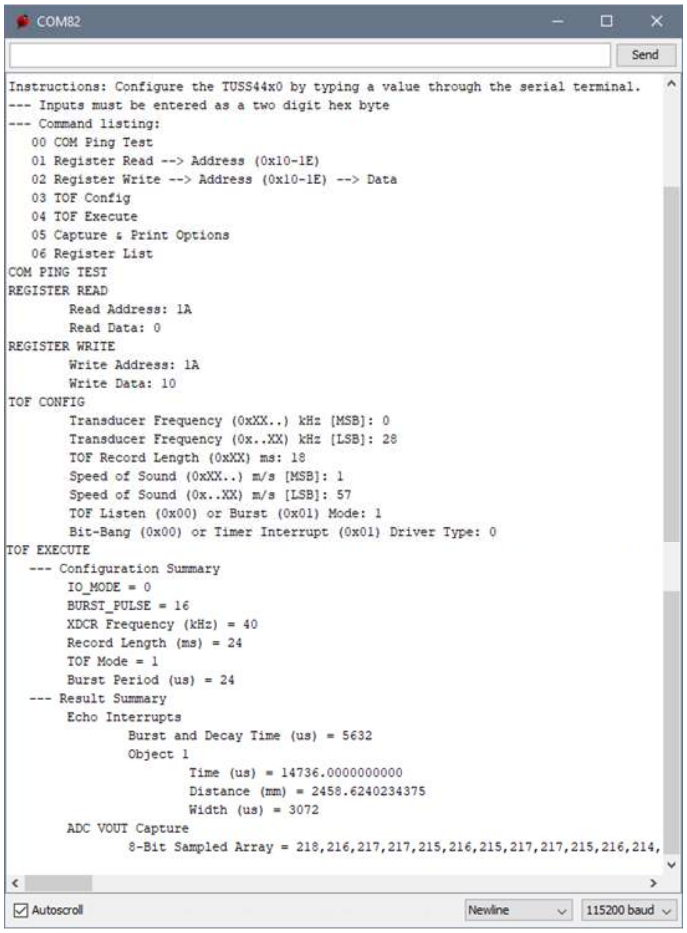
$$distance_{to\_object} = velocity_{sound} \times (time_{of\_flight\_round\_trip} / 2) = 343\ m/s \times (30\ ms / 2) = 5.145\ m \quad (1)$$

3 Energia Examples

The example TUSS44x0 program is written for the Energia IDE and is intended for cross-platform LaunchPad evaluation. The *GetDistance.ino* project requires user input through a COM serial terminal, and the *VOUT_ADC_Capture.ino* automatically runs and prints the VOUT ADC captured data in a Processing sketch display.

3.1 *GetDistance.ino - COM Terminal Input Example*

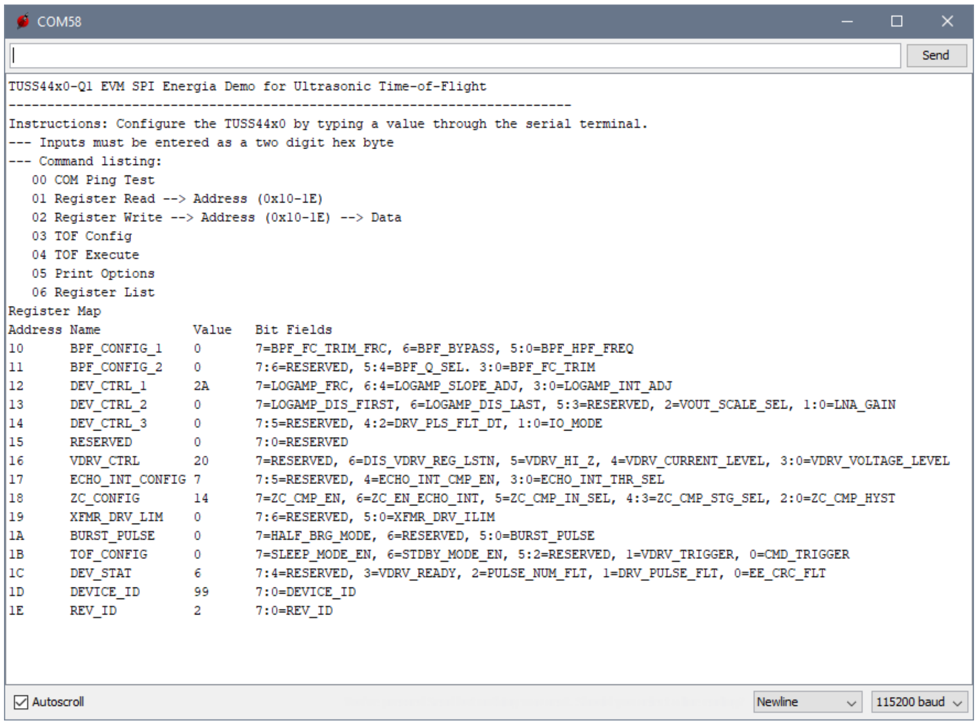
Before the Energia sketch is run, a serial COM terminal should be opened to see the instructions and available command listing for the GetDistance.ino sketch. A two digit hex input with a newline terminator should always be entered for every request from the sketch. For example, to run a COM Ping Test, send "00" through the COM terminal. This sketch allows you to perform a single register read or write (commands 01 and 02), configure the time-of-flight driver and record settings of the MCU (command 03), run a time-of-fight command to print the time-of-flight results of the VOUT ADC capture and/or the O4 Echo Interrupt (command 04), and choose what results to capture and display (command 05).



```
Instructions: Configure the TUSS44x0 by typing a value through the serial terminal.
--- Inputs must be entered as a two digit hex byte
--- Command listing:
   00 COM Ping Test
   01 Register Read --> Address (0x10-1E)
   02 Register Write --> Address (0x10-1E) --> Data
   03 TOF Config
   04 TOF Execute
   05 Capture & Print Options
   06 Register List
COM PING TEST
REGISTER READ
        Read Address: 1A
        Read Data: 0
REGISTER WRITE
        Write Address: 1A
        Write Data: 10
TOF CONFIG
        Transducer Frequency (0xXX..) kHz [MSB]: 0
        Transducer Frequency (0x..XX) kHz [LSB]: 28
        TOF Record Length (0xXX) ms: 18
        Speed of Sound (0xXX..) m/s [MSB]: 1
        Speed of Sound (0x..XX) m/s [LSB]: 57
        TOF Listen (0x00) or Burst (0x01) Mode: 1
        Bit-Bang (0x00) or Timer Interrupt (0x01) Driver Type: 0
TOF EXECUTE
   --- Configuration Summary
        IO_MODE = 0
        BURST_PULSE = 16
        XDCR Frequency (kHz) = 40
        Record Length (ms) = 24
        TOF Mode = 1
        Burst Period (us) = 24
   --- Result Summary
        Echo Interrupts
                Burst and Decay Time (us) = 5632
                Object 1
                        Time (us) = 14736.0000000000
                        Distance (mm) = 2458.6240234375
                        Width (us) = 3072
        ADC VOUT Capture
                8-Bit Sampled Array = 218,216,217,217,215,216,215,217,217,215,216,214,
```

Figure 6. TUSS44x0_Ultrasonic GetDistance.ino Results

The complete TUSS44x0 register listing and values can also be printed for review (command 06).



```
TUSS44x0-Q1 EVM SPI Energia Demo for Ultrasonic Time-of-Flight
--------------------------------------------------------------------------
Instructions: Configure the TUSS44x0 by typing a value through the serial terminal.
--- Inputs must be entered as a two digit hex byte
--- Command listing:
   00 COM Ping Test
   01 Register Read --> Address (0x10-1E)
   02 Register Write --> Address (0x10-1E) --> Data
   03 TOF Config
   04 TOF Execute
   05 Print Options
   06 Register List
Register Map
Address Name            Value   Bit Fields
10      BPF_CONFIG_1    0       7=BPF_FC_TRIM_FRC, 6=BPF_BYPASS, 5:0=BPF_HPF_FREQ
11      BPF_CONFIG_2    0       7:6=RESERVED, 5:4=BPF_Q_SEL. 3:0=BPF_FC_TRIM
12      DEV_CTRL_1      2A      7=LOGAMP_FRC, 6:4=LOGAMP_SLOPE_ADJ, 3:0=LOGAMP_INT_ADJ
13      DEV_CTRL_2      0       7=LOGAMP_DIS_FIRST, 6=LOGAMP_DIS_LAST, 5:3=RESERVED, 2=VOUT_SCALE_SEL, 1:0=LNA_GAIN
14      DEV_CTRL_3      0       7:5=RESERVED, 4:2=DRV_PLS_FLT_DT, 1:0=IO_MODE
15      RESERVED        0       7:0=RESERVED
16      VDRV_CTRL       20      7=RESERVED, 6=DIS_VDRV_REG_LSTN, 5=VDRV_HI_Z, 4=VDRV_CURRENT_LEVEL, 3:0=VDRV_VOLTAGE_LEVEL
17      ECHO_INT_CONFIG 7       7:5=RESERVED, 4=ECHO_INT_CMP_EN, 3:0=ECHO_INT_THR_SEL
18      ZC_CONFIG       14      7=ZC_CMP_EN, 6=ZC_EN_ECHO_INT, 5=ZC_CMP_IN_SEL, 4:3=ZC_CMP_STG_SEL, 2:0=ZC_CMP_HYST
19      XFMR_DRV_LIM    0       7:6=RESERVED, 5:0=XFMR_DRV_ILIM
1A      BURST_PULSE     0       7=HALF_BRG_MODE, 6=RESERVED, 5:0=BURST_PULSE
1B      TOF_CONFIG      0       7=SLEEP_MODE_EN, 6=STDBY_MODE_EN, 5:2=RESERVED, 1=VDRV_TRIGGER, 0=CMD_TRIGGER
1C      DEV_STAT        6       7:4=RESERVED, 3=VDRV_READY, 2=PULSE_NUM_FLT, 1=DRV_PULSE_FLT, 0=EE_CRC_FLT
1D      DEVICE_ID       99      7:0=DEVICE_ID
1E      REV_ID          2       7:0=REV_ID
```

Figure 7. TUSS44x0_Ultrasonic GetDistance.ino Registers and Bit Fields

3.2 *VOUT_ADC_Capture.ino - Standalone Example*

For a demonstration that does not require any user-input, the Energia VOUT_ADC_Capture.ino runs to allow the TUSS44x0 settings to be initialized by hard-coded values, then continuously performs time-of-flight measurements to capture the VOUT output with an ADC input to report the echo envelope data points over a serial port. The serial port can be accessed by the Processing ADC_VOUT_Capture.pde sketch to visually depict the echo envelope over time per time-of-flight measurement.

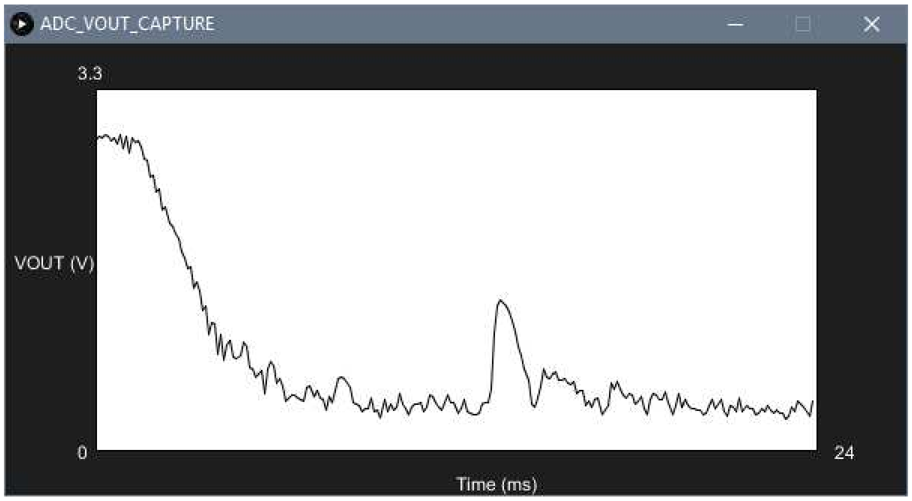


Figure 8. TUSS44x0_Ultrasonic VOUT_ADC_Processing.pde Processing GUI

4 References

- Texas Instruments: *TUSS44x0 Energia Library and Code Example*
- Texas Instruments: *TUSS4470 Direct Drive Ultrasonic Sensor IC With Logarithmic Amplifier Data Sheet*
- Texas Instruments: *Logarithmic Amplifier for Ultrasonic Sensor Signal Conditioning*