A Simple Glass-Breakage Detector Using an MSP430™ MCU

Robin Kammel, Kripasagar Venkat

MSP430 Applications

ABSTRACT

This application report describes a simple glass-breakage detector using the MSP430F2274 ultra-low-power microcontroller (MCU). The algorithm is based on the spectral analysis of a typical glass-breakage signal. The input signal spectrum, limited to a frequency of 20 kHz, is processed for a valid glass breakage. Various signal characteristics such as peak content, number of zero crossings, and frequency composition are analyzed. Real-time signal processing is achieved by implementing and using a low-order bireciprocal lattice wave digital filter (LWDF). A glass-breakage alert is indicated by an onboard buzzer and an LED. The entire setup is designed to operate at low power, allowing long battery life.

Related schematics and code source can be downloaded from www.ti.com/lit/zip/slaa351.

For more information, see A Simple Glass-Breakage Detector Reference Design and A Robust Glass-Breakage Detector Reference Design.

Contents

List of Figures

Trademarks

MSP430 is a trademark of Texas Instruments.
All other trademarks are the property of their respective owners.

1   Introduction

A glass-breakage detector can help ensure safety in buildings and homes. It is a simple mechanism to detect illegal entry through glass windows and doors. The detector analyzes acoustic signals produced during a glass breakage.

The frequency spectrum of the sounds produced during a glass breakage varies with the type of glass used in these doors and windows. This calls for a variety of solutions tailored to types of glass used. In this application report, typical glass-breakage sounds have been used as a tool to design the alert mechanism. These acoustic signals are analyzed after being captured by an onboard microphone. The steps that follow this capture are explained in detail in this report.

The MSP430F2274 is the microcontroller chosen to do this analysis.[2] Although the processor can operate up to 16 MHz, an active-mode frequency of 12 MHz is used.[1] Furthermore, the CPU operating frequency is changed between 8 MHz and 12 MHz to reduce power consumption. The required peripherals are enabled every 2 ms, only when the input signal needs to be captured, to ensure optimal power management. An optional anti-aliasing filter (AAF) in hardware is activated to ensure the signal spectrum is restricted to 20 kHz. The total power consumption of the system is approximately 80 µA with the AAF enabled and approximately 50 µA with the AAF disabled.

The entire hardware setup, software flow, and test setup are discussed in this report. The following sections provide a complete description of the hardware, software, and test setup. The complete details of this reference design are provided in A Simple Glass-Breakage Detector Reference Design.

2   Hardware Description

This section describes the glass-breakage detector board using the MSP430F2274 MCU.

2.1   Device Specifications

The MSP430F2274 is a 16-bit microcontroller (MCU) from the 2xx family of the ultra-low-power MSP430™ family of devices from Texas Instruments.[1][2] The supply voltage required for the microcontroller spans a broad range of 1.8 V to 3.6 V. The MCU operates at frequencies up to 16 MHz. The CPU has a 16-bit RISC architecture with a total of 51 instructions (27 core and 24 emulated). It supports a single-cycle shift and single-cycle add and subtract instructions. This enables efficient multiplication in the absence of a hardware multiplier.[4] The MCU also has an internal very-low-power low-frequency oscillator (VLO) that operates at 12 kHz at room temperature. This oscillator eliminates the need for an external onboard crystal for the operation of the device. However, an option has been provided on the board to use external crystal/resonators of up to 16 MHz. The MCU has two 16-bit timers (Timer_A and Timer_B), each with three capture/compare registers. An integrated 10-bit analog-to-digital converter (ADC10) supports conversion rates of up to 200 kilo-samples per second (ksps). The ADC10 can be configured to work with to on-chip operational amplifiers (OA0 and OA1) for analog input signal conditioning. The memory model supports up to 32KB of flash memory and 1KB of RAM in addition to 256 bytes of information memory. This device comes with four 8-bit I/O ports that can be used to control external devices. The current consumption of 0.7 µA during standby mode and active mode current of 250 µA at 1 MHz make this device an excellent choice for battery-powered applications.

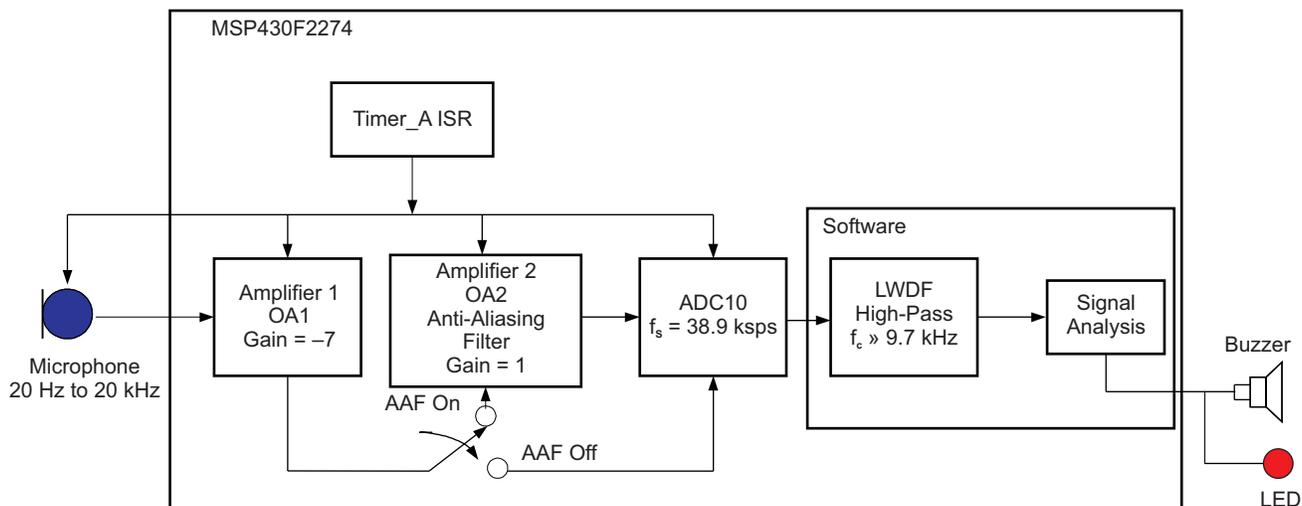Figure 1 shows the setup for the glass-breakage detector using this device.

Figure 1. Setup for Glass-Breakage Detector Using MSP430F2274

The microphone captures the analog input, and a buzzer or LED indicates detection of glass breakage. The op amps internal to the MSP430 MCU are connected to a few external passive components as part of the design of active analog filters.

2.2 Power Supply

The board is powered by two 1.5-V AAA batteries delivering a supply voltage of 3 V. For reliable operation at 12 MHz CPU frequency, the supply voltage (DV_{CC}) must be 2.7 V or higher. Low-battery detection is not implemented for this demonstration application but can be added using the internal ADC10.

2.3 Microphone

The input signal is captured by a Panasonic WM-61A microphone that has a frequency response between 20 Hz and 20 kHz and can operate at voltages up to 10 V. The microphone has a signal-to-noise ratio of approximately 60 dB. The microphone is turned on every 2 ms when the input signal needs to be captured. This on-and-off mechanism is controlled in software by port pin P4.0 of the MSP430 MCU, which provides the supply voltage of DV_{CC} to the microphone. The maximum current consumption during this capture is approximately 0.5 mA. The microphone is physically disabled by placing a jumper on header X3.

2.4 LED and Buzzer Alert

Glass-breakage detection is indicated by a buzzer or an LED. The buzzer used in this application is the Panasonic EFB-RL37C20. This buzzer can generate sound level outputs up to 106 dB at a frequency of 3.7 kHz. For high sound levels, a large voltage must be supplied to this buzzer. To obtain this high voltage, a DC/DC boost converter is used. The low-power TPS61040, which can provide an output voltage of 28 V when sourced by a 3-V voltage supply, is used. The current consumption during its operation is approximately 15 mA. This TPS61040 supports an enable signal that is controlled in software by port pin P3.7 of the MSP430 MCU to turn on only during a glass breakage. The buzzer circuit can also be disabled using the onboard jumper (JP3) that physically disconnects the power to the buzzer, so that indications can be given using only the LED.

2.5 Interface to CC1100 or CC2500 Devices

The MSP430 MCU has proven to be a great solution for ultra-low-power applications. For wireless applications, the MSP430 MCU must interface to the an existing transceiver such as the TI devices CC1100 or CC2500. They form low-cost single-chip transceivers equipped with serial interfaces that can be used to directly communicate with the MSP430 MCU. On this board, an option is provided to interface to the CC1100 and CC2500 evaluation boards, and libraries are available for use with the MSP430 MCUs. These libraries are available for download and can be used to develop a wireless glass-breakage detector with the existing board.[7]

2.6 Operational Amplifiers (OAs)

The MSP430F2274 has two OAs that are configurable using software. The gain of these OAs can be set by internal resistor ladder settings. Depending on the choice to have an AAF, one or both of the OAs are used in this application. The first OA (OA0) is used as an inverting amplifier with a gain of 7. The output of OA0 is connected internally to one of the channels of ADC10 for further processing. If the AAF is needed, the output of OA0 is internally connected to OA1, which is configured as a unity-gain low-pass filter. The filter is a second-order Butterworth filter, realized through a Sallen-Key circuit with the 3-dB cutoff set at 19.2 kHz. For both of these OAs, the reference is maintained at the voltage $DV_{CC}/2$.

2.7 Internal Very-Low-Power Oscillator (VLO)

The MSP430F2274 has an internal VLO that typically operates at 12 kHz. With the VLO as a clock source, the MSP430 MCU can be operated in low-power mode 3 (LPM3). Using the VLO eliminates the need for an external low-frequency crystal. The VLO is the clock source to the internal timer, which is functional when the device is in standby or LPM3. Timer_A is responsible for the periodic wake up every 2 ms. Due to the temperature drift of the VLO frequency, this wake-up time could be different. An algorithm to consider the drift of the VLO and its impact on the wake up time of 2 ms is not implemented but can be added. For example, the factory-provided DCO calibration constants in the flash memory can be used to determine the VLO frequency and adjust the length of the wake-up interval accordingly.

2.8 JTAG Interface

The board has a standard 14-pin header for interface to programming tools through JTAG. A 2-wire interface known as the Spy-Bi-Wire (SBW) is used instead of the conventional 4-wire interface for the JTAG interface in this design. This 2-wire interface is supported only by some in the MSP430 2xx family of devices. The user must make this selection to program the MSP430F2274. For SBW communication, the USB FET debugger must be used.

2.9 Current Consumption

The board is sourced by the two 1.5-V AAA batteries, which have a capacity of approximately 800 mAh. This gives a battery life of up to 416 days at 80-µA current consumption and up to 666 days at 50-µA current consumption. These figures are with the peripherals switched off when not in operation. The current consumption can be further reduced if the wake up is increased from the present interval of 2 ms. However, a longer interval increases the chances of missing a sound event. Figure 2 through Figure 4 are graphical representations of the current consumption for this board.

Figure 2 shows the current consumption profile when there is no activity. The device goes into active mode 1 (CPU at 8 MHz), denoted by AM1, for 37.5 µs once every 2 ms.
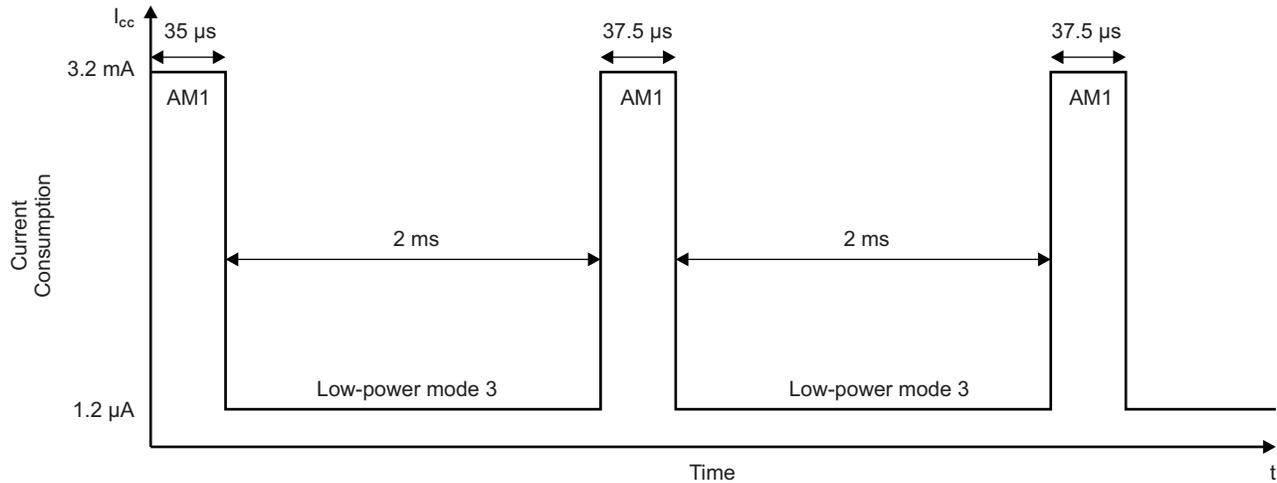

Figure 2. Current Consumption Profile During No Activity

Figure 3 shows the current consumption profile when there is activity detected on the microphone. The device goes into active mode 1 (CPU at 8 MHz) first for 37.5 µs and then, if the signal is a possible glass breakage, the CPU is configured to work at 12 MHz, denoted as AM2. Samples are accumulated for a period of 60 ms, and signal analysis (as described in Section 3.4) is done on every sample. If the signal is not a glass breakage, the device returns to LPM3 with Timer_A reconfigured.
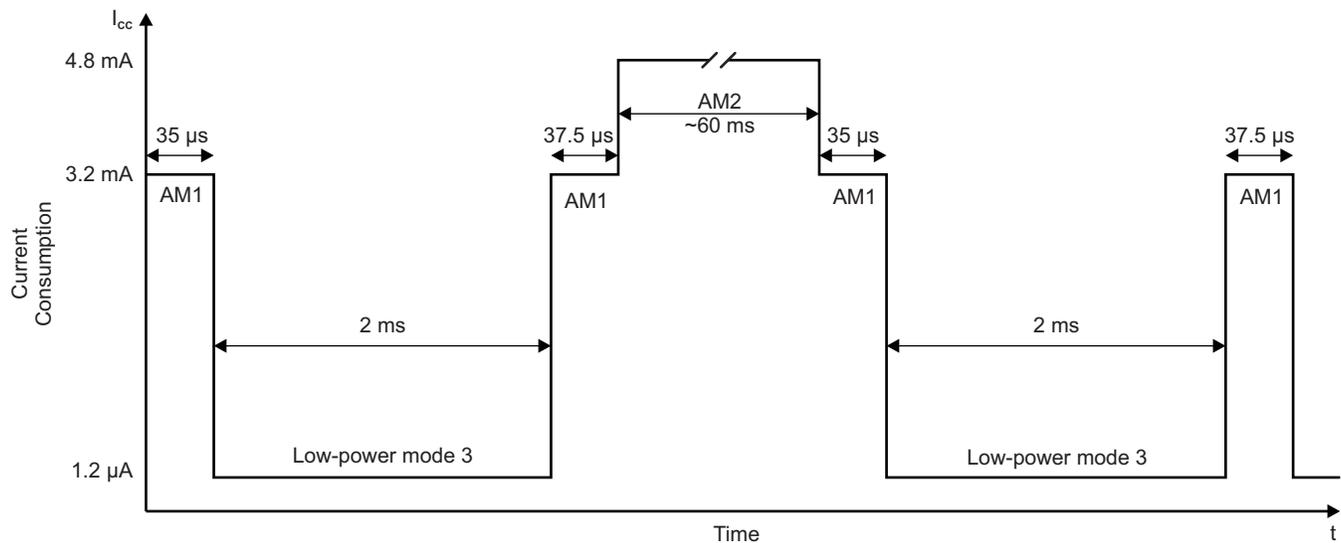

Figure 3. Current Consumption Profile During Activity on the Microphone

Figure 4 shows the current consumption profile when there is a glass breakage. This is a step further in comparison to Figure 3. If a decision is made in favor of a glass-breakage detection, the CPU continues in AM2, during which time the LED and buzzer are on for a period of three seconds. After this time, the device is reinitialized and returns to LPM3.
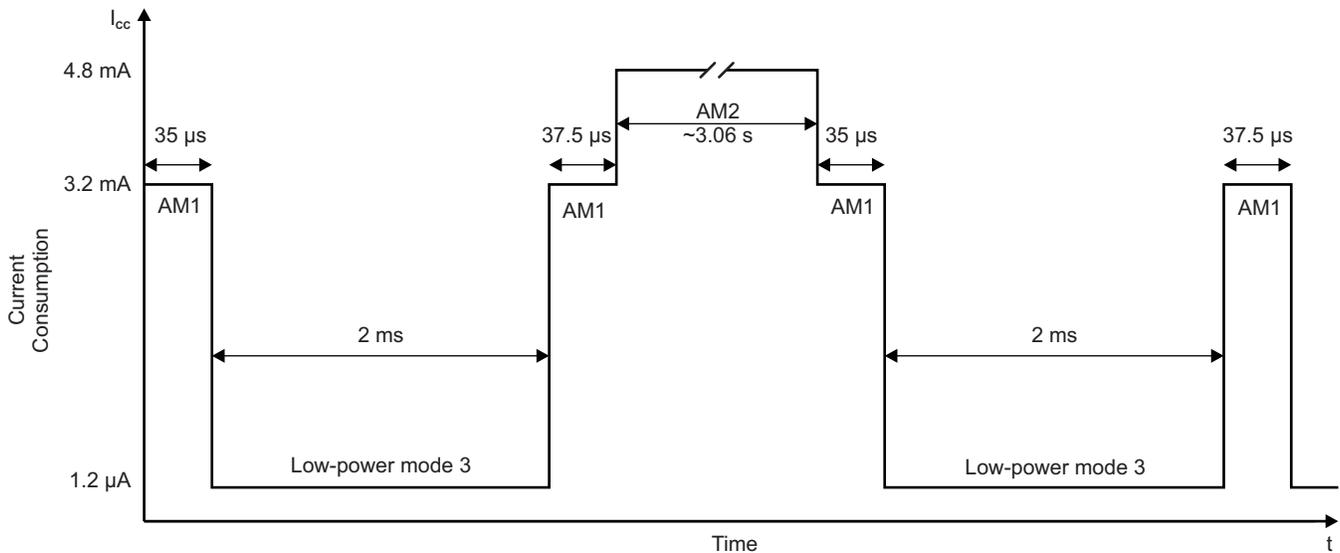

Figure 4. Current Consumption Profile During Glass-Breakage Detection

3 Software Description

This section discusses the flow and description of the software modules used in the glass-breakage detector using the MSP430F2274 MCU.

3.1 Initialization Routine

Figure 5 shows a high-level flow diagram of the software execution for initialization of the board. This step occurs at the beginning and after a glass-breakage detect has been issued.

This section of the program initializes all of the peripherals that are needed in the detection routine. The peripherals initialized by this program include Timer_A, CPU clock, ADC10, OAs, and port pins.

The first step is to disable the watchdog timer to avoid an unintended watchdog timer reset. The CPU clock is then set to 8 MHz, and port pins that correspond to the analog inputs are set accordingly. Unused pins are set to outputs, and their values set to high to avoid current consumption. A check on the AAF_select flag is made to enable or disable the AAF. A zero on this flag disables the AAF, and a nonzero value enables it. If this filter is disabled, the input to ADC10 is the output of OA0 and appears at channel A1 of the ADC10. With AAF enabled, the input to ADC10 is the output of OA1, which appears at channel A13 of ADC10. In both cases, the ADC10 is configured with its clock set at SMCLK/3 and for single-conversion mode. Independent of the choice made on the use of AAF, both OA0 and OA1 are configured but not enabled. As mentioned previously, OA0 is configured as an inverting amplifier with a gain of 7 and OA1 as a unity gain buffer. After these initializations are complete, the device enters LPM3, during which time all of the clocks except ACLK (chosen from VLO) are switched off. Timer_A, which is configured to generate an interrupt every 2 ms, wakes the device from this mode.
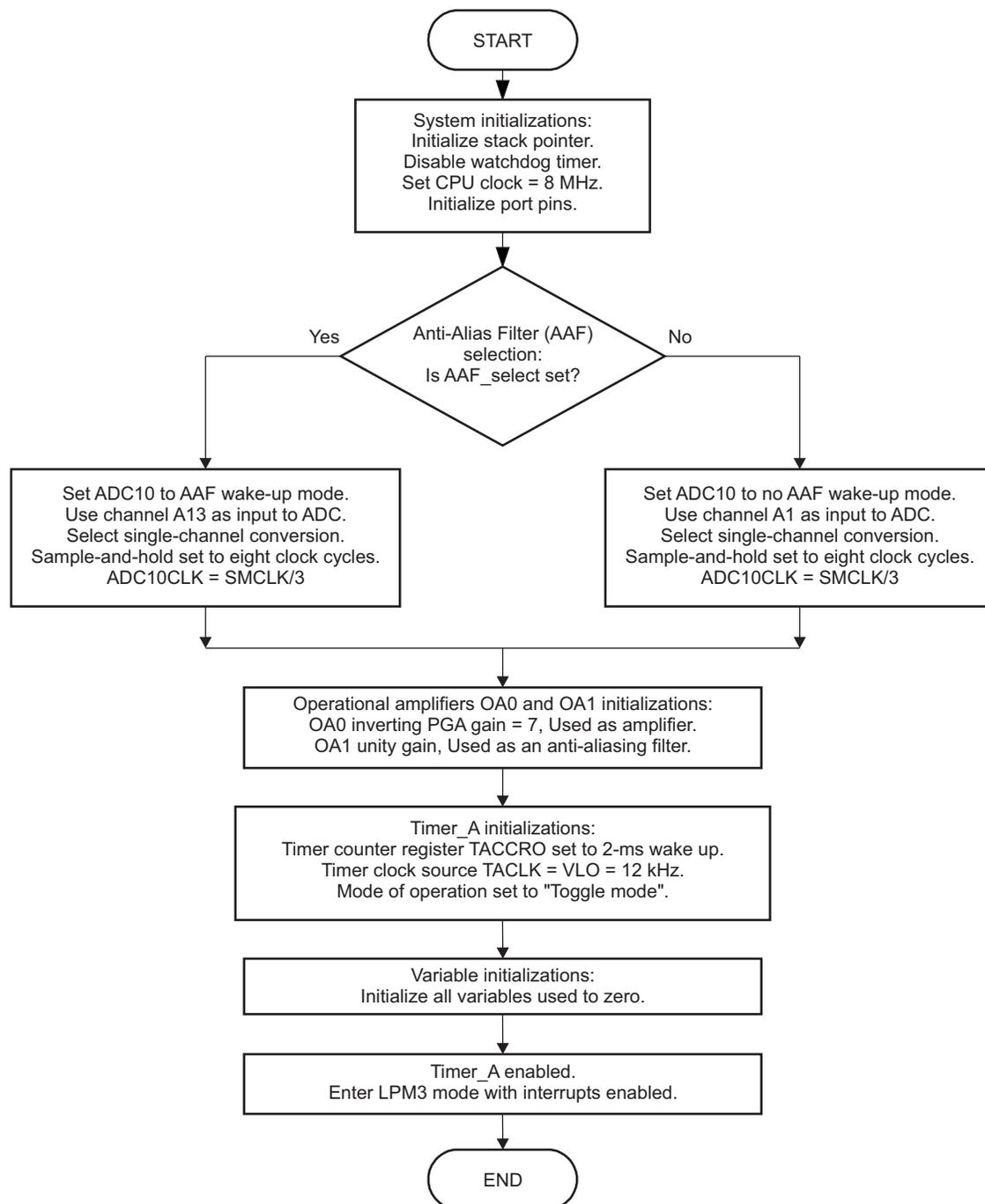
Figure 5. Flowchart for Initializations

3.2   Timer_A

In this application, Timer_A is used to generate the periodic wake up every 2 ms. The timer clock is sourced from the on-chip VLO, which operates at 12 kHz at room temperature. The timer counter TACCR0 is set to a value to generate an interrupt every 2 ms at the VLO clock. The input signal measurement for a glass-breakage detect is done in the Timer_A interrupt service routine (ISR). The program flow is shown in Figure 6. The external microphone is first enabled, and then OA0 and/or OA1 is enabled, depending on the choice made to use the AAF. ADC10 is then enabled to convert the incoming signal either at channel A13 or channel A1. For completed conversions, the digitized samples are first converted to a bipolar format by subtracting a value equal to 520, which is approximately $DV_{CC}/2$. The resulting value is compared to the thresholds –40 and +90, which distinguish spurious noisy inputs from a true sound event. In the absence of any sound event, all the peripherals are switched off with Timer_A reconfigured to work in LPM3 mode. On the other hand, if the sound levels are significant, the CPU clock is increased from 8 MHz to 12 MHz. ADC10 is reconfigured for repeated single-channel conversions at a sampling rate approximately of 39 ksps. Program control is passed on to the ADC10 ISR, which performs the signal analysis of the incoming signal to detect a glass breakage.
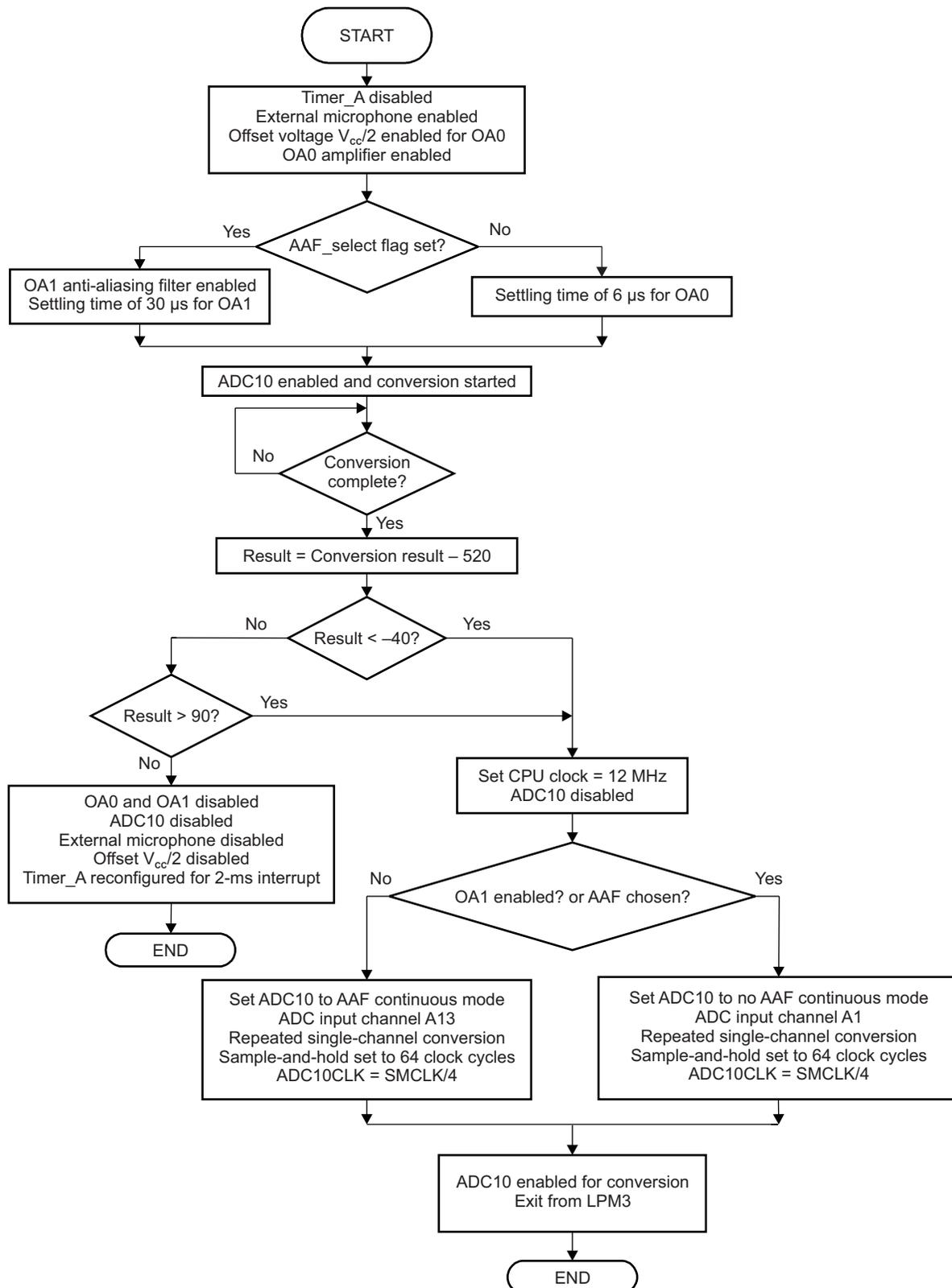
Figure 6. Flowchart of Timer_A ISR Functionality

3.3 ADC10

The MSP430F2274 has an integrated 10-bit analog-to-digital converter (ADC) capable of sampling rates of up to 200 ksps using the internal reference. ADC10 is turned on only at the beginning of the Timer_A ISR, to ensure low current consumption. In continuous conversion mode, the sampling frequency (f_s) is set at a rate of 38.96 ksps at a typical CPU frequency of 12 MHz. The input channel to ADC10 is connected to different OAs, depending on the choice made to have the AAF. Once an active conversion is complete, the ADC10 is switched off. The ADC10 ISR is used to perform the signal analysis and determine a glass breakage. For real-time operation, the entire processing must be complete before the arrival of the next sample. This number of available CPU cycles between successive sampling instants is approximately 300.

The ADC10 ISR is active only after a valid sound event is detected. The complete signal analysis is done in this ISR. Each converted sample is converted to a bipolar signal by subtracting 512. Figure 7 shows the software flowchart in this ISR.
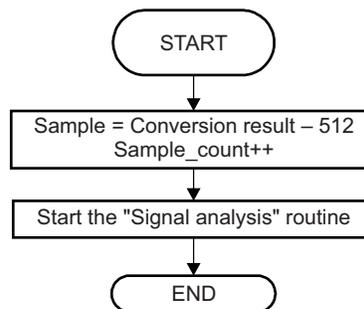

Figure 7. Flowchart of ADC10 ISR Functionality

The signal analysis routine forms a major part of this ISR and is dealt with in detail in Section 3.4. ADC10 is active until 60 ms of the incoming signal is passed through a stage of preliminary processing. Further processing of this data does not require the ADC10 to remain active, and it is switched off.

3.4 Signal Analysis

This portion of the software implements the methodologies used to detect a glass breakage. The signal analysis is split into two parts. The first part of the signal analysis is done every sample and is completed before the arrival of the next sample. The second part of the signal analysis is done on 60 ms of the incoming data that has passed through the first stage of the signal analysis. This is not done on a sample-by-sample basis, but over 2336 samples that correspond to 60 ms of the incoming sample. The duration of 60 ms is not based on any requirement of the algorithm but was chosen as a convenient number for efficient processing.

3.4.1 First Stage of Processing

The first stage of processing is done every sample once a sound event has been detected. Figure 8 shows a summary of the operations for this processing.
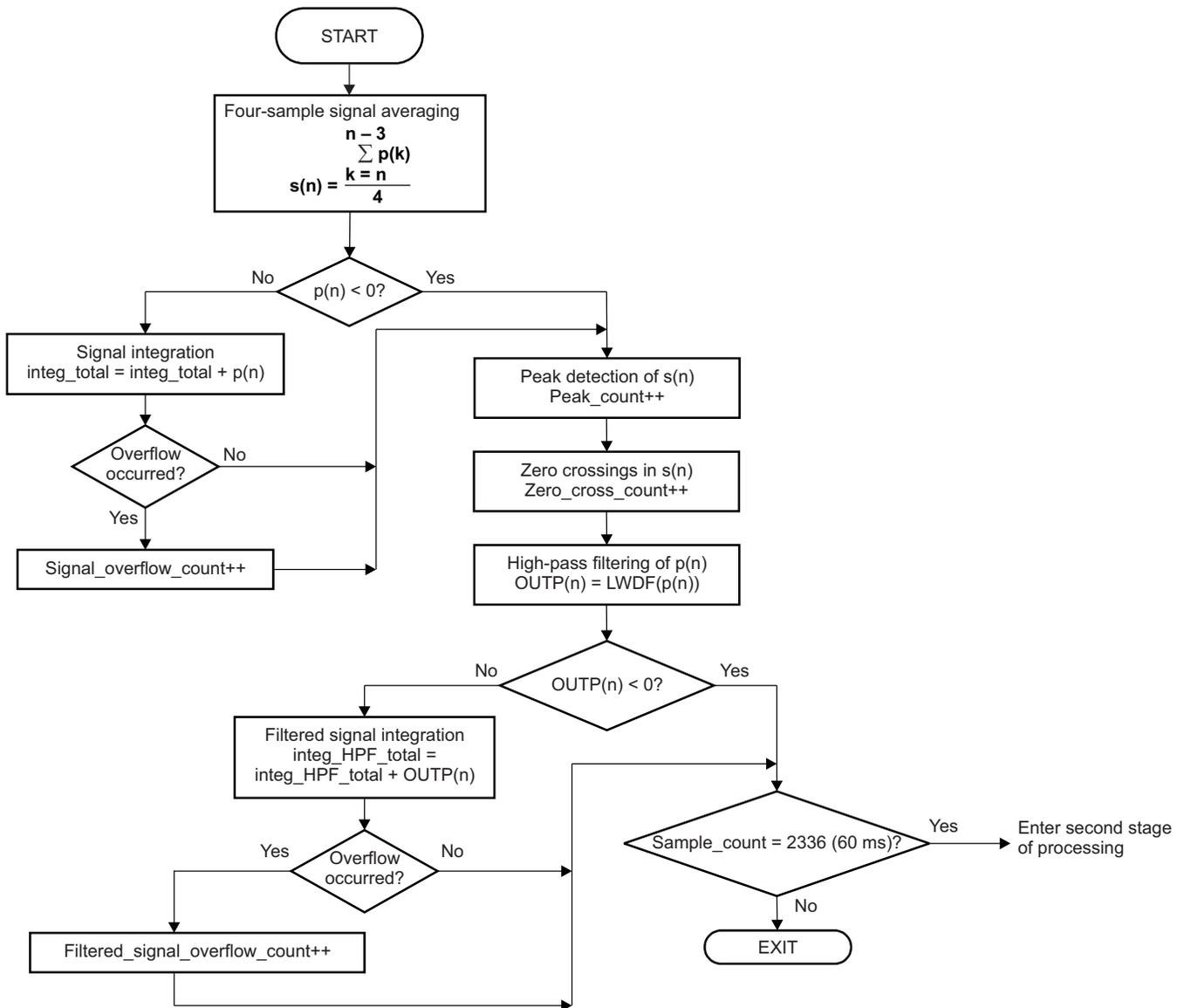
Figure 8. First Stage Signal Analysis Flowchart

3.4.1.1   *Signal Averaging, Peak Detection, and Zero Crossings*

To reduce noise in the incoming signal, a signal averaging is done. The averaged signal is obtained using a simple accumulate function of the current input sample and the previous three samples. This process is needed only to obtain accurate results during peak detection and zero crossings. An integration of the positive nonaveraged signal is done to facilitate the second stage of processing. During this integration, a check for overflow is done and kept count to accommodate a 32-bit result. The number of zero crossings of the averaged signal is done as part of the algorithm used to detect a glass breakage. Figure 9 shows the signal averaging, peak detection, and the zero crossings of the averaged signal, respectively.
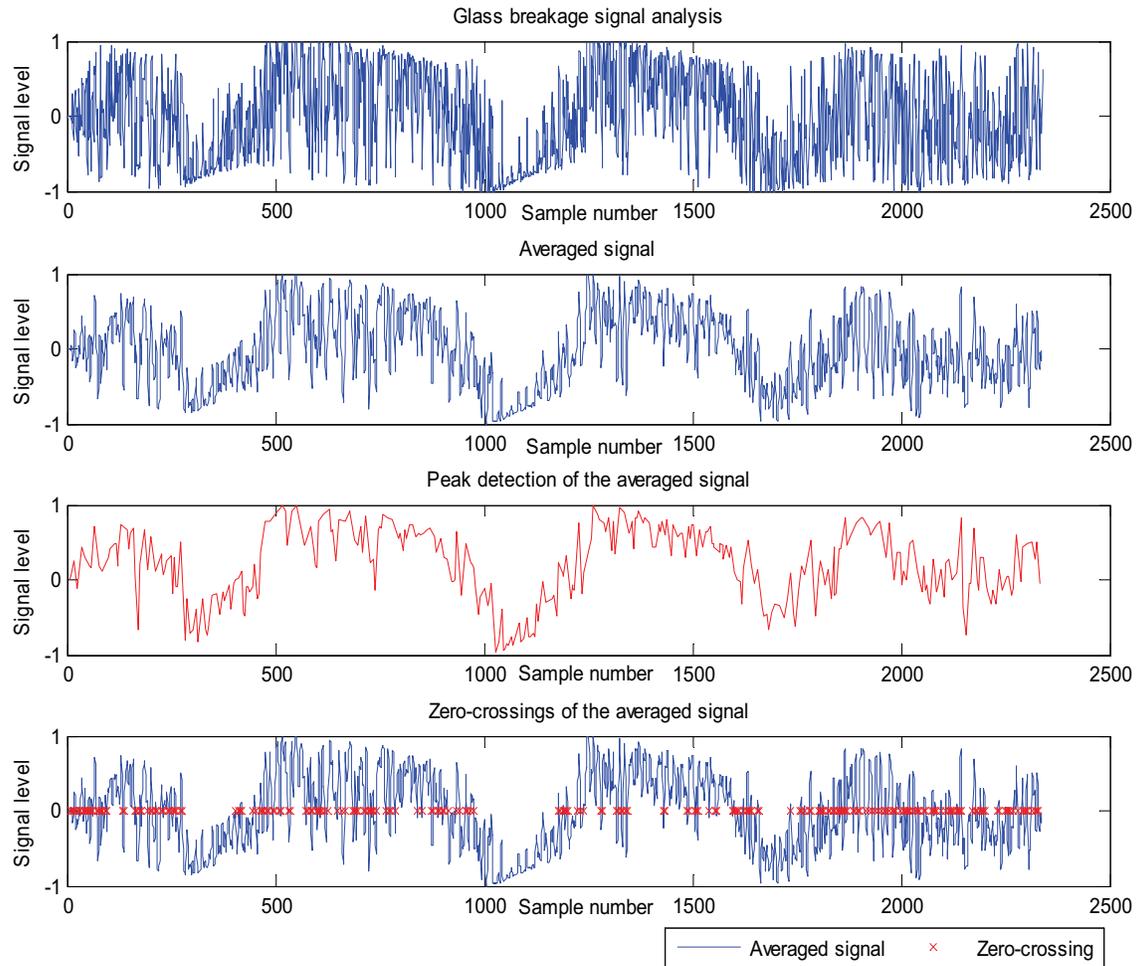

Figure 9. Glass-Breakage Signal Analysis in Time Domain

3.4.1.2 High-Pass Filtering

The non-averaged signal is then passed through a high-pass filter with a cutoff set at one-fourth the sampling frequency (9.7 kHz). This is done to remove the low-frequency signals to decide on a glass breakage. The lattice wave digital filter (LWDF) is used to do this filtering. The LWDF is well suited for microcontrollers without a hardware multiplier.[3] The LWDF exhibits excellent stability properties over nonlinear conditions and has a good dynamic range in its coefficients. The bireciprocal filter structure of seventh order is chosen in this application. The LWDF gives one output sample every sample period, and this output is also integrated to facilitate the second stage of processing. Similarly, during this integration, the overflow is considered, and a 32-bit result accommodated. Figure 10 shows the signal flow diagram of the seventh-order bireciprocal LWDF. During the signal analysis, high-pass filtering is done using the LWDF.[5]
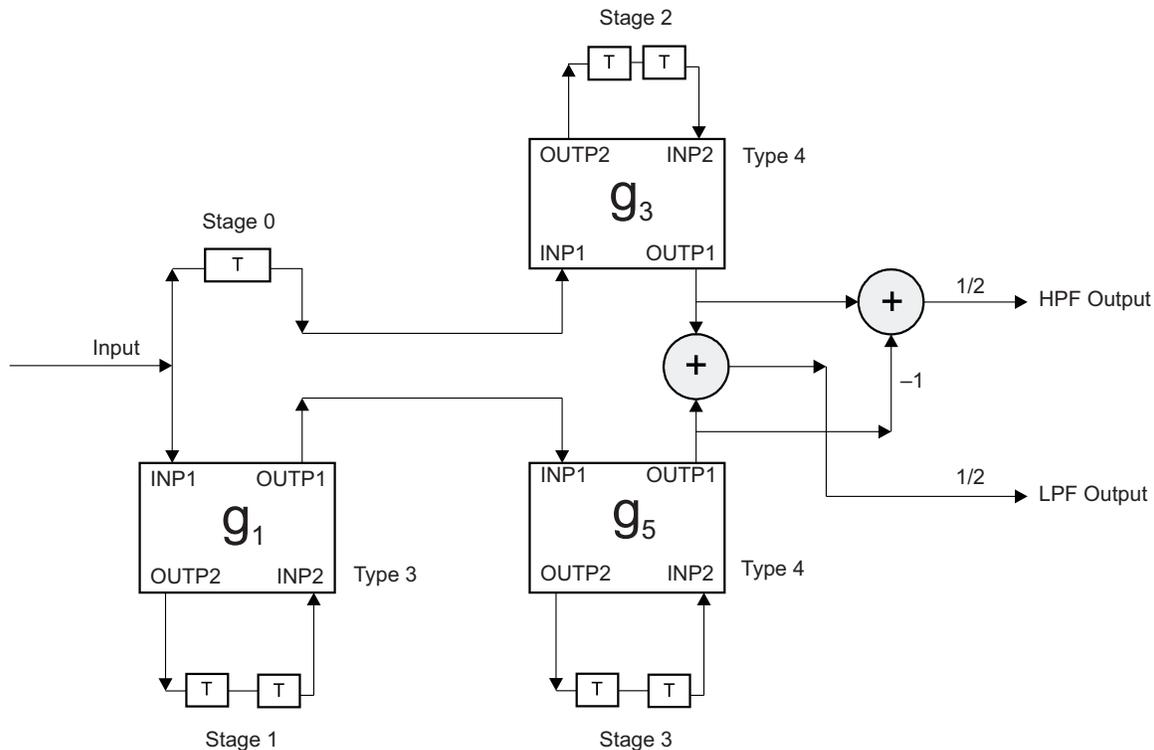

Figure 10. Signal Flow Diagram of a Seventh-Order Bireciprocal LWDF

The LWDF structure is able to simultaneously give the complimentary output to the HPF with just one extra instruction cycle. The entire filtering operation is done in only 153 instruction cycles. The filter specifications and the resulting coefficients are:

Filter specifications:

> Filter response type = High pass
>
> Sampling frequency = 38.960 kHz
>
> 3-dB cutoff frequency = 9.74 kHz
>
> Stop-band attenuation = 44 dB
>
> Filter type = Elliptical
>
> Filter structure = Bireciprocal
>
> Filter order = 7
>
> Filter coefficients = −0.109375, −0.375, and −0.75

Figure 11 shows the frequency response of this LWDF with its complimentary counterpart.
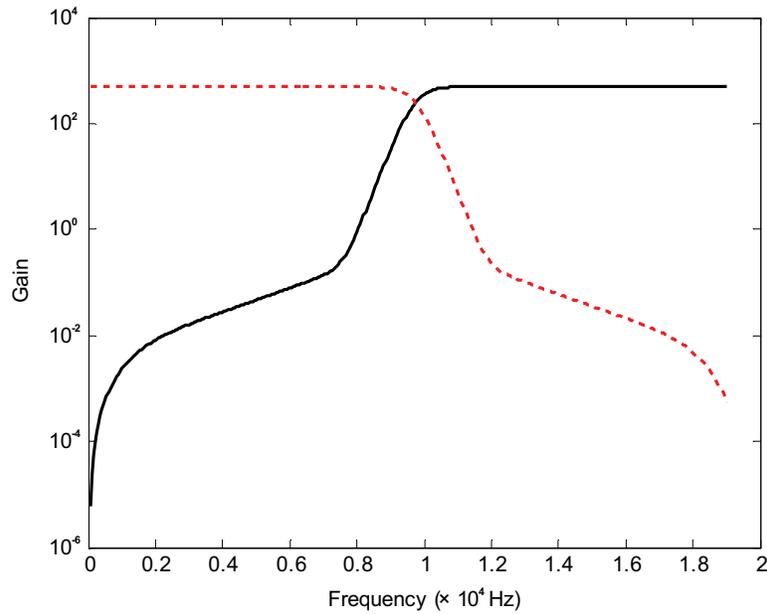
Figure 11. Frequency Response of Seventh-Order Bireciprocal LWDF

3.4.2   Second Stage of Processing

The second stage of processing does not require a real-time operation. It is done only after 60 ms of the input signal has passed through the first stage of processing. Some of the parameters evaluated during the first stage in conjunction to some new processing are used to complete the process of a glass-breakage detect. Figure 12 shows a flowchart that summarizes the operations.

START

Disable ADC10, Timer_A, OA0, OA1

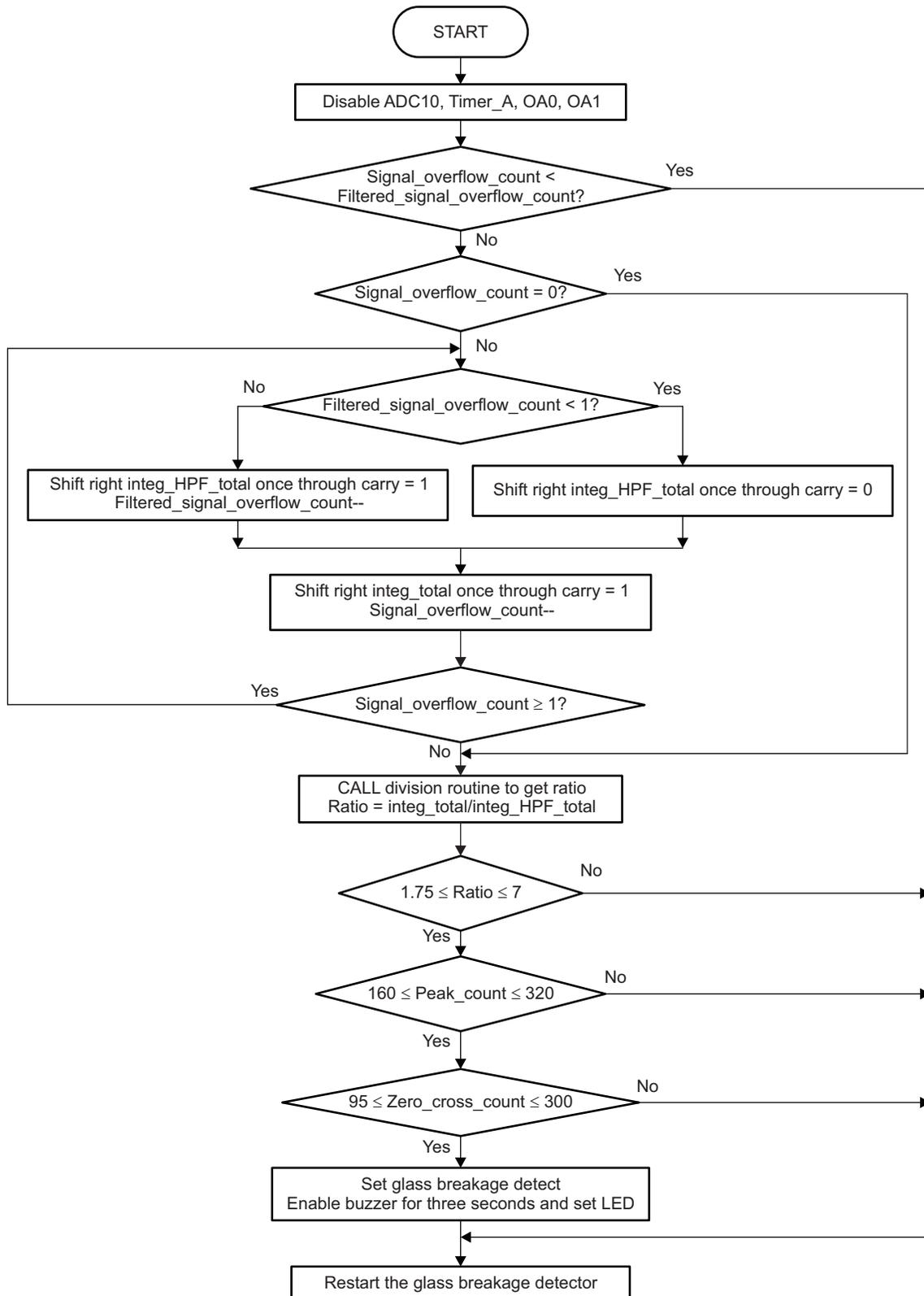Signal_overflow_count < Filtered_signal_overflow_count? — Yes

No

Signal_overflow_count = 0? — Yes

No

Filtered_signal_overflow_count < 1?

No — Yes

Shift right integ_HPF_total once through carry = 1
Filtered_signal_overflow_count--

Shift right integ_HPF_total once through carry = 0

Shift right integ_total once through carry = 1
Signal_overflow_count--

Signal_overflow_count $\geq$ 1? — Yes

No

CALL division routine to get ratio
Ratio = integ_total/integ_HPF_total

$1.75 \leq$ Ratio ≤ 7 — No

Yes

$160 \leq$ Peak_count ≤ 320 — No

Yes

$95 \leq$ Zero_cross_count ≤ 300 — No

Yes

Set glass breakage detect
Enable buzzer for three seconds and set LED

Restart the glass breakage detector

Figure 12. Second Stage Signal Analysis Flowchart

3.4.2.1 Frequency Composition Ratio

During the first stage of processing, the incoming signal was accumulated for the entire period of 60 ms. This is termed "Signal integration" in Figure 8. Also, the filtered signal (i.e., the output of the LWDF) was correspondingly accumulated. This was termed "Filtered signal integration" in Figure 8. A ratio of these accumulated values compared to a fixed threshold is done in the detection of a glass breakage. To obtain a ratio of these runtime values, integer-integer division must be performed. An optimized 16-bit integer division routine has been implemented to do this.[6] The routine is separated into two parts: one that calculates the integer part of the quotient and one that calculates the fractional. To calculate the approximate fractional part, the divisor is simply divided by four and the result subtracted from the remainder of the integer. A counter is used to store how many times one-fourth of the divider fits into the remainder, thus calculating an approximation of the fractional part in increments of 0.25. If the ratio of the "Signal integration" to the "Filtered Signal integration" fails to fall in the range of 1.75 to 7, a false flag is set, halting further processing. These fixed threshold values vary with different types of glass, and it is left to the end users to set these thresholds based on the spectrum analysis of their glass-breakage signal.

3.4.2.2 Peak and Zero-Crossing Count

During the first stage of processing, the number of peaks and zero crossings of the samples were counted. These values are now used to detect a glass breakage. This comparison is done if the ratio condition is satisfied. A false flag is set if the number of peaks falls outside the range of 160 to 320 and if the number of zero crossings falls outside the range of 95 to 300. These fixed values again depend on the type of glass used.

3.4.2.3 Glass-Breakage Detect

If none of the false flags are set, then a glass-breakage detect is issued. This immediately enables the buzzer and the LED for indication. The buzzer is turned on for a period of three seconds and then turned off. The device then goes back into its wake-up mode settings and is ready for the next detection.

4 Hardware Schematic

This section shows the hardware design schematic for the glass-breakage detector board. Figure 13 shows the schematic of this board. The zip file at www.ti.com/lit/zip/slaa351 has all of the necessary files to reproduce the hardware and software designs.
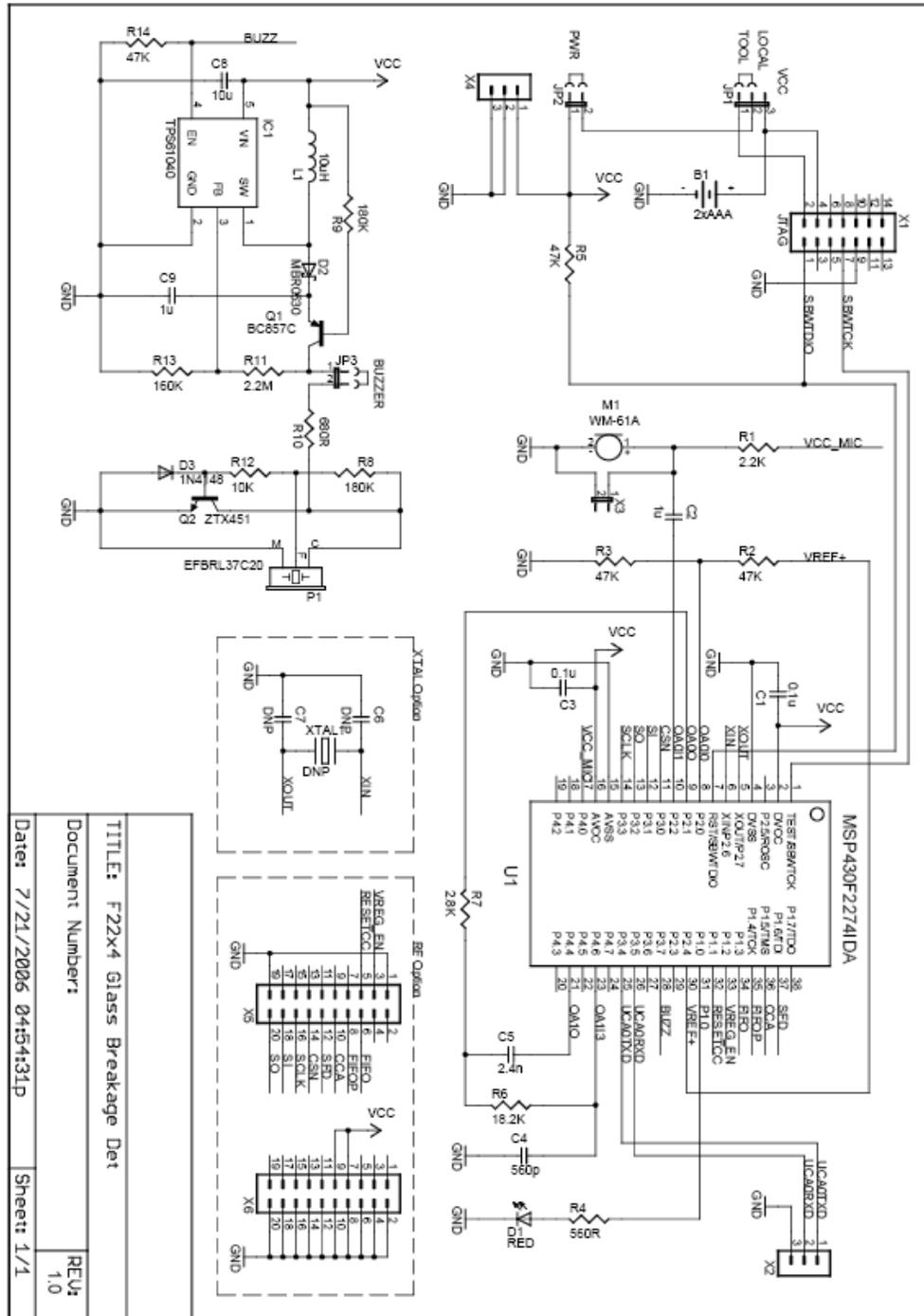

Figure 13. MSP430F2274 Glass-Breakage Detector Board Schematic

5   Test Setup

This section describes the test setup for the verification of this glass-breakage detector. Various glass-breakage test files were obtained from the sound effects vendor sounddogs.com. Some of the tests were performed with increased sound level of the original sound files to ensure that all sound files have nearly the same sound pressure level. A specified sound level is measured at a certain distance from the center of the membrane of the main box while a particular sine test tone is replayed. Figure 14 through Figure 16 show the measurement setup.
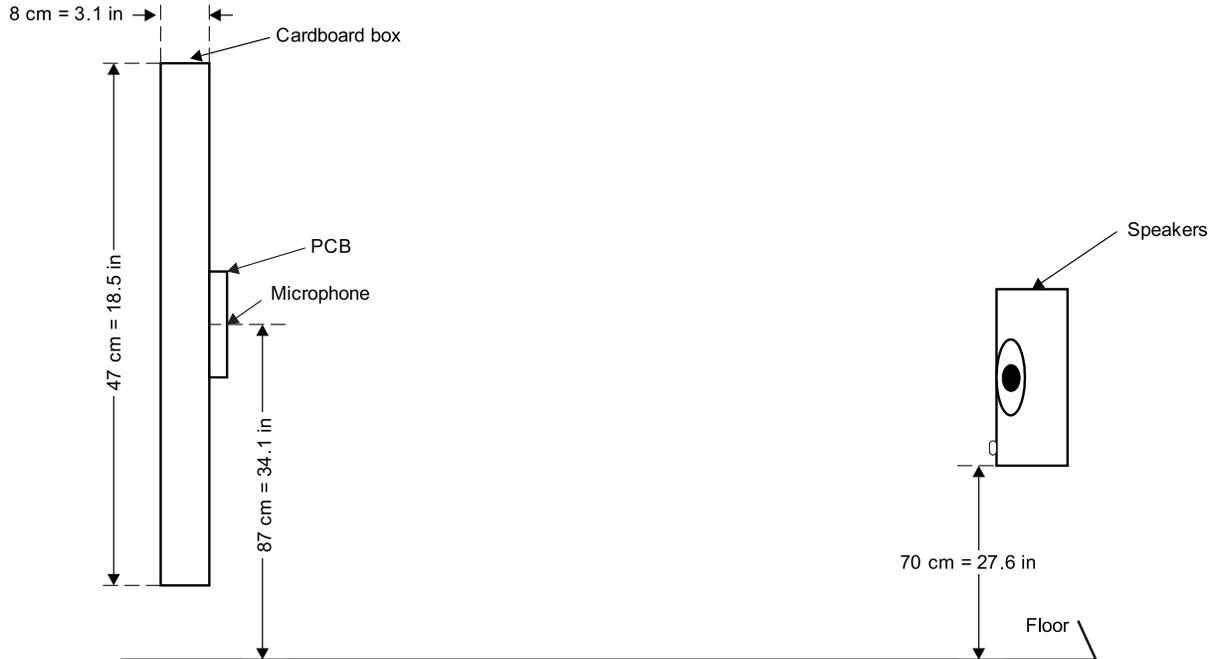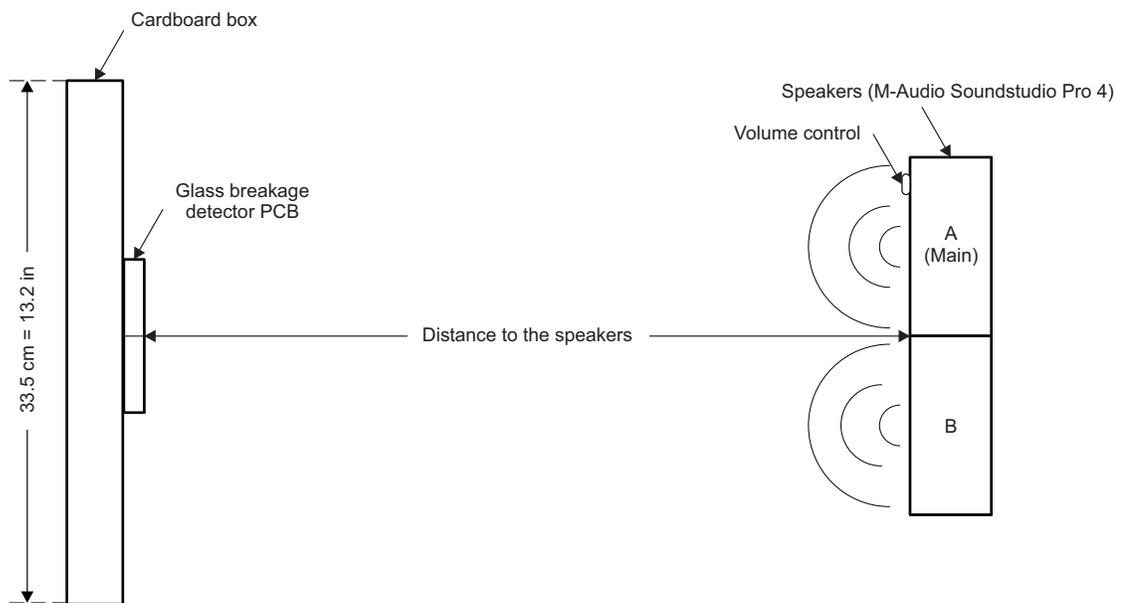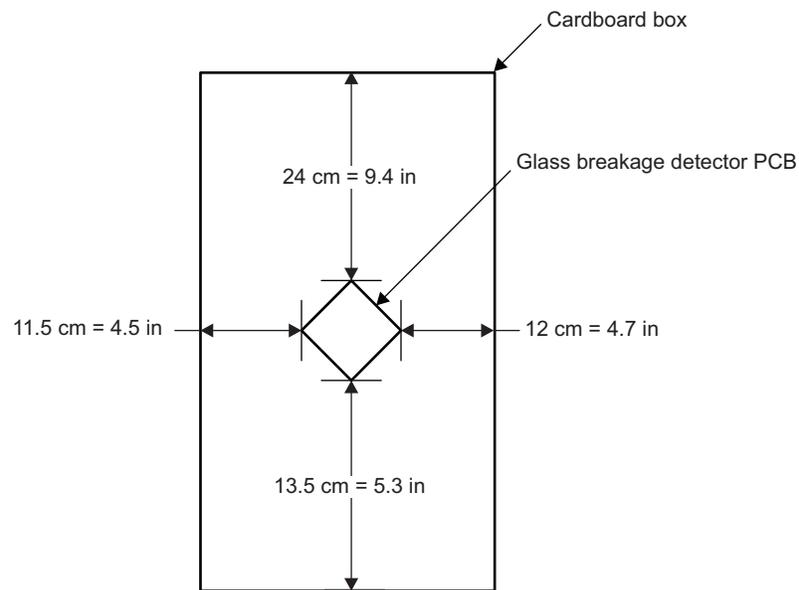

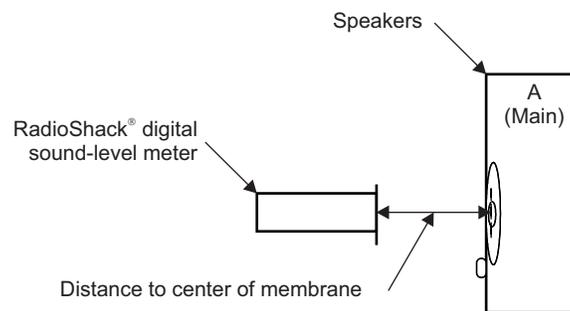
Figure 14. Test Setup (Side View)

Figure 15. Test Setup (Top View)

Figure 16. Test Setup (Front View)

Figure 17. Test Setup (Sound-Level Measurement)

6 References

1. MSP430x2xx Family User Guide
2. MSP430x22x2, MSP430x22x4 Mixed-Signal Microcontrollers
3. Kaiser, Ulrich, "Wave Digital Filtering for TI's Sensor Signal Processor MSP430", *Texas Instruments Technical Journal*, Vol. 11, Nr. 6, 1994 S.65-83
4. Venkat, Kripasagar, Efficient Multiplication and Division Using MSP430
5. Venkat, Kripasagar, Wave Digital Filtering Using the MSP430
6. V. Carl Hamacher, Zvonko G. Vranesic, Safwat G. Zaky, *Computer Organization*, Third Edition, McGraw Hill Publication, 1990
7. Quiring, Keith, MSP430 Interface to the CC1100/2500 Code Library

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from April 27, 2007 to November 9, 2018   **Page**

- Editorial and formatting changes throughout document 1
- Added links to related reference designs 1

IMPORTANT NOTICE AND DISCLAIMER

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

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI's products are provided subject to TI's Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI's provision of these resources does not expand or otherwise alter TI's applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.