MSP430 Powered Low Cost Oscilloscope

Quinn Leydon
Rochester Institute of Technology
RIT
Rochester, New York

***Abstract*—Oscilloscopes are a necessary interment for debugging microcontrollers. As the use of microcontroller begins at a younger age, a less expensive alternative is needed. In this paper a device implementing an MSP430 and Capacitive touch sensor is designed and used as an alternative for a traditional oscilloscope.**

Keywords—MSP430; Capacitive Touch; Timer; UART; ADC; Oscilloscope

I. Introduction (*Heading 1*)

As PCB development decreases in price and increases in demand, there are more microelectronic devices produced than ever before. Individuals are introduced to their first microcontrollers at a younger age. This increases the demand for an affordable Oscilloscope to read the output of GPIO pins from microcontrollers. The cost of a new digital oscilloscope begins around $350 for an entry level model. This is far too expensive for most new users, especially as the age of introduction decreases. These oscilloscopes are far too complicated for a young user. The device proposed will use one MSP430 and one capacitive touch sensor to provide a basic understanding of an electrical output. It can be used to gain a basic understanding of the shape and frequency of an output as well as its respective voltage level.

II. Ease of Use

A. Capacitive Touch Sensor

The TI capacitive touch sensor has 8 display LEDs and 5 capacitive touch buttons. Where a large Oscilloscope would have a complicated interface, this device provides information in a simple way that a young user can understand. After attaching the output device to the proposed product, two selections must be made. One for sample speed and one for a linear vs logarithmic scale. The output is shown on the LED interface and can be read through UART communication

B. Maintaining the Integrity of the Specifications

To maintain the device, a user can access the code using Code Composer Studio, a free software provided by TI. To decrease the sensitivity of base values, the amount decreased from the initial measurement can be increased. The timer can be reconfigured to change the sample rate of the device. The voltage range of input is from 0V to 1V.

III. Proposed System

A. System Overview

Fig 1 shows the flow chart of the proposed mechanism. The user will first attach the input to pin 0.1. The user may then turn on the board and hit the central button to begin. The user may then choose up to sample at 100 Hz, or the down to sample at 200 Hz. After selection the user can chose between the left button do display a linear scale or the right button to display a logarithmic scale. The display uses both a physical led display and a digital output the number of LEDs over UART.

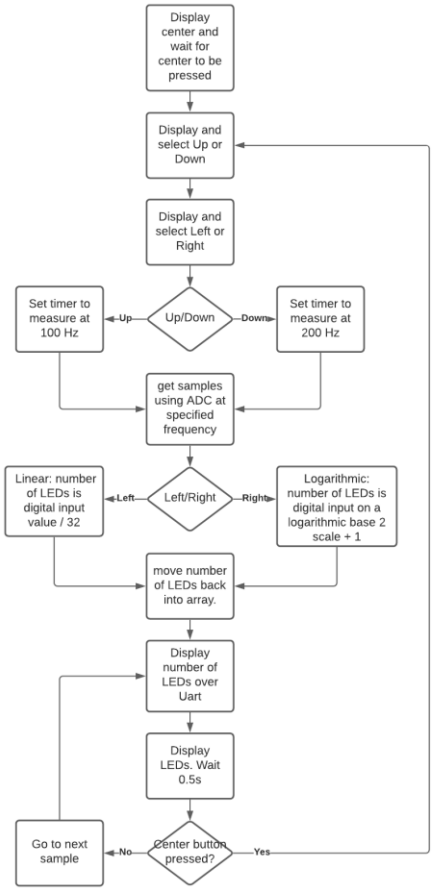

Fig. 1. Flow Chart of proposed design

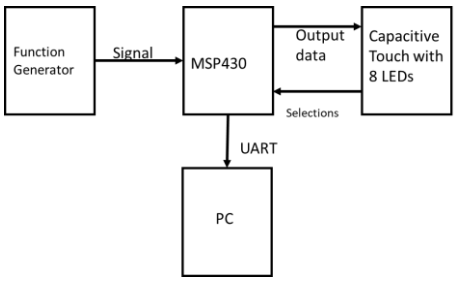

Fig. 2. Wiring Diagram

B. MSP430 LaunchPad

The MSP430 LaunchPad is an entry level group of microcontroler developed by Texas Instraments. The MSP430 used is a MSP-EXP430G2ET. This is a 20 pin launchpad. This includs a MSP430G2553 16MHz MCU with 16KB FLASH, 512B SRAM, 10-bit ADC, comparator, UART/SPI/I2C, Timer [1]. This is programed using Code Composer Studio, a free software provided by TI.

C. Capacitive Touch Sensor

The Capacitive Touch sensor 430Boost-Sense1 is sourced from Texas Instruments add designed for the MSP430 series LaunchPad used. Consisting of 5 capacitive inputs and 8 LEDs it connects directly to the GPIO pins on the surface of the MSP430. Fig. 2 shows the Sensor touch areas and LED placement. The sensor detects a change in the dielectric properties of the surrounding area. By setting the GPIO pins to oscillation mode, the board can read readings from the sensor. Since a finger is more conductive than open air, the sensor has more capacitance thus a larger period. The oscillations are measured during an acquisition period. These are used to determine the frequency and thus the change in dielectric of the sensor. This uses Timer 0 in capture mode. The LEDs are multiplexed meaning the right half of the board cannot be displayed at the same time as the left half of the board. By oscillating between the two sides at a high frequency, it appears that both sides of LEDs are turned on simultaneously.

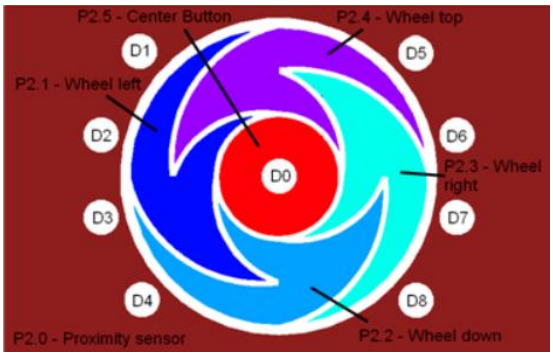

Fig. 3. Capacitive Touch Layout [2]

D. Timers

The MSP 430 can implement to timers, timer 0 and timer 1. For the proposed product both timers are identical. A timer in compare mode will capture the TAR register value at the moment the interrupt is generated. This is used like a stopwatch between two events. Compare mode generates an interrupt when the TAR value equals the value in TAXCCRX. This allows the timer to be used as a traditional timer. A timer in up mode will count to TAxCCR0 and reset to zero.

E. ADC

The MSP 430 has an onboard analog to digital converter, or an ADC. The ADC compares the input voltage to a reference voltage to produce a digital value. The ADC used is a 10bit ADC with a 1.5V reference voltage making the tolerance 1.46 mV.

F. UART

UART stands for Universal Asynchronous Receiver/Transmitter. This is an asynchronous communication protocol, so a synchronizing clock is not transmitted. Instead, a baud rate is assigned so that the receiver and transmitter communicate at the same frequency. The baud rate used is 9600, an industry standard. Since the clock within the microcontroller is not perfect, a scaling factor must be added to adjust the baud rate.

IV. Development

A. Touch Sensing and Selection

The project uses the capacitive touch sensor to start and determine both acquisition rate and conversion A. central series of function measures all 5 sensors and returns the information on which sensor was pressed is used to make these decisions. After the initial setup of the capacitive touch sensor, a baseline for all five sensors is collected. These values are reduced by two and saved as baseline values. The value can be decreased from two but may be tripped due to noise in the system. If the value is increased, it is less likely to have a false reading but may miss a light touch. For each consecutive measurement, if the value of read from the sensor is lower than the base value, that sensor is read as active. Each sensor is given its own unique ID which is saved to as a variable and returned to the caller. The center function will exit only if the center button is pressed. A function determines if the up button or down button is selected. It moves the ID of the selected button into an updown variable which will be read later. If this value is equal to that of up or down, the function exits, and the value is saved in updown. If the value is not up or down it receives five new values and checks again. The same process is repeated to determine if the left or right button was pressed and saved to leftright after up or down is selected. By using this method, the values are saved for later and the code will not skip a section if left is pressed during the up down sequence.

B. Acquiring Samples

A function to retrieve the samples first reads the up down value and determines what speed to the set the timer. Timer 1 is placed in up mode, with a division of one, cleared and has

the TAIE interrupt enabled. The division must be cleared as it is set to 8 later in the code. Timer 1 is used for timing and timer 0 is used for receiving samples from the capacitive touch sensor. While this does not impact the function, it will impact the display samples loop. Timer 1 is used for consistency. Since TAIE is used, an interrupt will run every time the timer is reset. The timer will reset every time it counts to TA1CCR0 is reached. Since SMCLK is used, which has a frequency of 10 kHz, or a period of 100us. This this clock input was chosen because it can be used for both sample acquisition and displaying samples. The ISR that is triggered clears the flag and sets a new flag which is used to determine if the ISR was executed.

If up was previously picked in the sequence, then the values are acquired at 100Hz. TA1CCR0 is set to 10,000 or 0x2710. If down was selected, values are acquired at 200Hz. TA1CCR0 is set to 5,000 or 0x1388

The acquire function resets the timer and interrupts. It then waits for the interrupt to be triggered. After the interrupt is triggered it stops the timer and disables its interrupt. This results in the acquisition time to be a few microseconds longer than expected when accounting for delay from the rest of the process, but it stopped complications of the ISR running during the ADC process. The ADC process is then set to begin. Once a value is returned interrupts are disabled and the configuration registers are cleared. This stops a new value from being measured before the value can be saved into the data array. Fig 3 shows a sine wave acquired at 100Hz. Fig 4 shows a sine wave acquired at 200Hz.

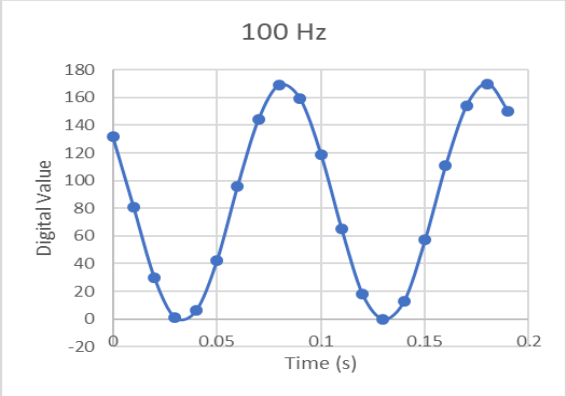

Fig. 4. 100 Hz Acquisition of a 10Hz Sine Wave

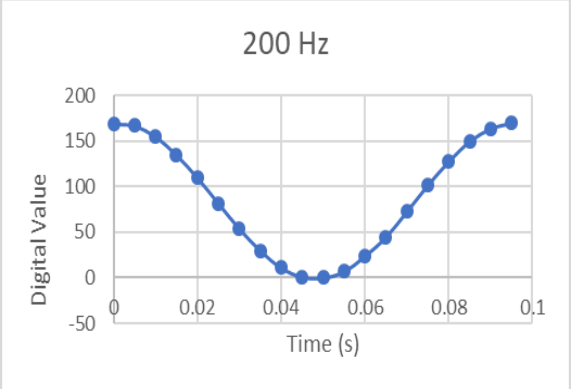

Fig. 5. 200 Hz acquisition of a 10Hz Sine Wave

C. *Convert samples*

Convert samples determines if left or right is pressed. This decision determines the number of LEDs which will be displayed. The algorithm does not consider the least two bits, so they are removed before conversion. To remove these bits the 10-bit ADC value is arithmetic moved right twice, then bit cleared with 0xFF00 to remove any carry in that may have occurred. If the left button was pressed, the samples will be converted to a linear scale where the data is divided by 32 and that value is rounded up. Since division is not a simple process of a microcontroller, a series of compare and jump if less than statements determine the number of LEDs to be displayed. A more complex algorithm could be used; however, this algorithm was simpler to debug. The number of LEDs is moved into a different array. This costs data but allows the user to see the hard data if they look in Code Composer Studio. If right is selected the same process occurs but the value being compared begins with 2^0 and ends with 2^7, meaning 0 will display zero LEDS and any value greater than 2^7 will display 8 LEDs. This is the equivalent to the function

$$Log_2() + 1 \quad (1)$$

These functions could be implemented using a short algorithm. The series of compare statements were used because they are direct and easier to debug. It also explicitly states its function and simplifies the display routine.

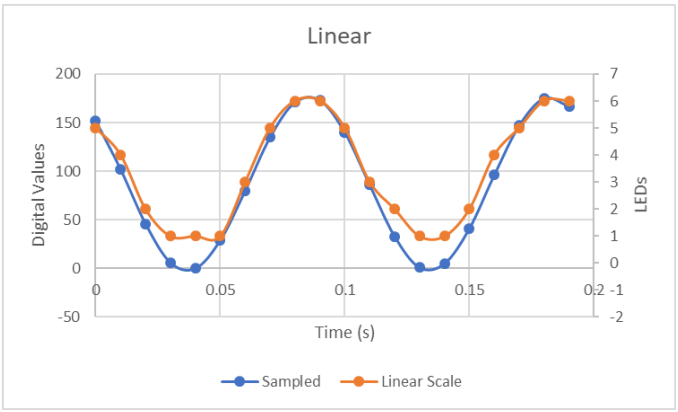

Fig. 6. Linear Display of a 10Hz Sine Wave

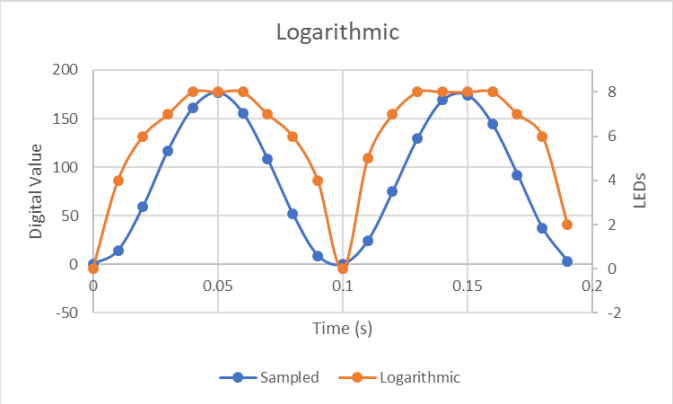

Fig. 7. Logarithmic Display of a 10Hz Sine Wave

D. Display Samples

The display samples routine displays the number of LEDs through UART and displays them on the board for 0.5s. First Timer 1 must be set to have a division of 8 and a TA1CCR0 vale of 0xF424. This is 500,000 divided by 8. It uses SMCLK, a divider of 8, up mode, and is cleared. This uses CCIE which triggers its interrupt when TA1CCR0 is reached. Like the get samples function, the timer is only active when it is in the timing loop. This stops the code from entering the ISR trap unnecessarily but decreases the accuracy of the 0.5s time. Function uses a series of compare and jump if equals to move to the correct LED function. This displays the number of LEDs to be displayed over UART, and then sets a return and new line character so the next data entry will be on its own line. It then enters a loop that waits for the ISR to be run. In that loop the left side of the board has its LEDs turned on then the right side turned on. This occurs at a high enough frequency that both sides appear to be constantly on. After the loop is executed the timer is stopped, interrupts disable, and LEDs are turned off. This loops through all the data and repeats.

If the center button is pressed during this routine, the system should restart at up down. To implement this the check for center function occurs while waiting for the timer ISR. This makes the center touch incredibly quick. A downside of having the check for center in the loop is that it has a long delay time so one half of the LEDs is far brighter than the other. By implementing a software delay for the other half of the board, both sides of LEDs can be held at equal brightness.

V. Conclusion

A low-cost oscilloscope is designed, implementing a Texas Instruments MSP430 and Capacitive Touch Board. The capacitive touch board provides a simple user interface designed so that even the younger users can understand. With a production cost below $50, this device is far more affordable than competitive entry level Oscilloscopes. Through the uses of a free software, Code Composer Studio, the user can fully customize the device. A user can select rate of acquisition, linear or logarithmic conversion of data, and see the values through an LED display.

References

[1] Texas Instruments. “MSP-EXP430G2ET.” *MSP-EXP430G2ET Development Kit | TI.com*, 2020, www.ti.com/tool/MSP-EXP430G2ET?keyMatch=MSP-EXP430G2ET.

[2] Texas Instruments (2011) *Users Guid: 30BOOST-SENSE1- Capacitive Touch Booster Pack for the LaunchPad. Number SLAU337B.* Retreived from https://www.ti.com/lit/ug/slau337b/slau337b.pdf?ts=1606064460285&ref_url=https%253A%252F%252Fwww.google.com%252F.

[3] Texas Instraments (2004*) User’s Guide: MSP430x2xxFamily*. Number SLAU144J Retreived from https://www.ti.com/lit/ug/slau144j/slau144j.pdf.

[4] Barrios, Carlos (2020) *EEEE – 420 – Lab 11* .