Multiple processing schemes using TI MSP430

Long Nguyen[1]
[1] Department of Electrical and Microelectronic Engineering,
Rochester Institute of Technology, Rochester, NY 14623

Abstract

The TI MSP430 Launch Pad is a line of microcontrollers commonly used for low cost, low power consumption embedded applications. In this project, the MSP430 is used to perform multiple processing methods, and output through the universal asynchronous receiver-transmitter (UART). The user can choose between linear or logarithmic conversion and display speed by interacting with the MSP430 Capacitive Touch Booster pack.

I. Introduction

A. Hardware setup

The microcontroller used in this project is the TI MSP430 Launch Pad. For the code to work properly, jumpers P2.1, P2.3 and P2.5 must be removed and the UART TXD and RXD ports are connected.

To capture the user's touch input, the Capacitive Touch is attached to the MSP430 so that the logos on 2 devices are aligned. There are 5 sensors (Center, Left, Right, Up, Down) and 9 LEDs (Center LED and 8 surrounding LEDs).

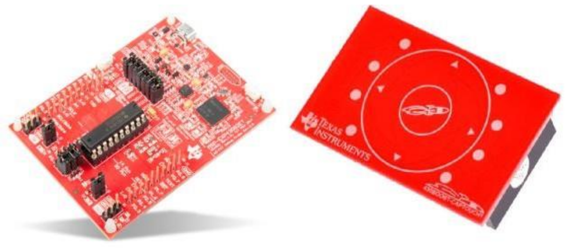

Figure 1: TI MSP430 and MSP430 Captouch Boost1 sensor

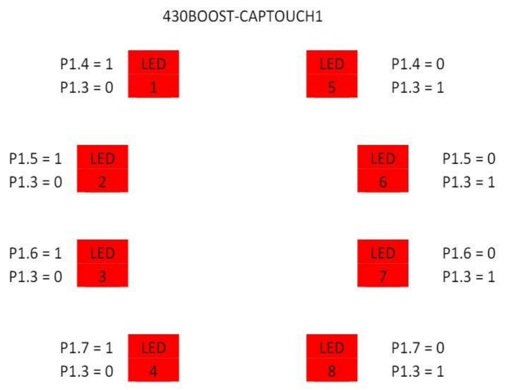

Figure 2: LEDs positions on the Capacitive Touch

B. Functionality

The program receives 20 numbers entered by the user through UART communication protocol. The data will then be processed by how the user interact with the Capacitive Touch Booster. Specifically, the user can choose between combination of up/down and left/right. Up and down sensors control the processed data display speed (0.2s/display for up, 0.5s/display for down). Left and right sensors control how data will be processed (linear for right, logarithm for left). After displaying all 20 processed number, it will repeat and ask for inputs. The program is implemented in a mix of Assembly and C language.

C. The linear and logarithmic conversion

In the last 10 bits of each number enter, linear conversion only considers the first 3 bits. The range of data will be equally divided by 8, represented by the number of LEDs turned on

10-bit sample	LEDs to turn on
000x xxxx xx	4
001x xxxx xx	4, 3
010x xxxx xx	4, 3, 2
011x xxxx xx	4, 3, 2, 1
100x xxxx xx	4, 3, 2, 1, 5
101x xxxx xx	4, 3, 2, 1, 5, 6
110x xxxx xx	4, 3, 2, 1, 5, 6, 7
111x xxxx xx	4, 3, 2, 1, 5, 6, 7, 8

Table I: LEDs turn-on order in linear conversion

Log conversion will perform the "leading 1" where the position of the first "1" bit in the first 8 bits determines LEDs output. When the first "1" is shifted left, the value is doubled. Thus, the logarithm conversion reacts to every double of data value.

10-bit sample	LEDs to turn on
0000 0000 xx	none
0000 0001 xx	4
0000 001x xx	4, 3
0000 01xx xx	4, 3, 2
0000 1xxx xx	4, 3, 2, 1
0001 xxxx xx	4, 3, 2, 1, 5
001x xxxx xx	4, 3, 2, 1, 5, 6
01xx xxxx xx	4, 3, 2, 1, 5, 6, 7
1xxx xxxx xx	4, 3, 2, 1, 5, 6, 7, 8

Table II: LEDs turn-on order in logarithmic conversion

II. SOFTWARE DESIGN

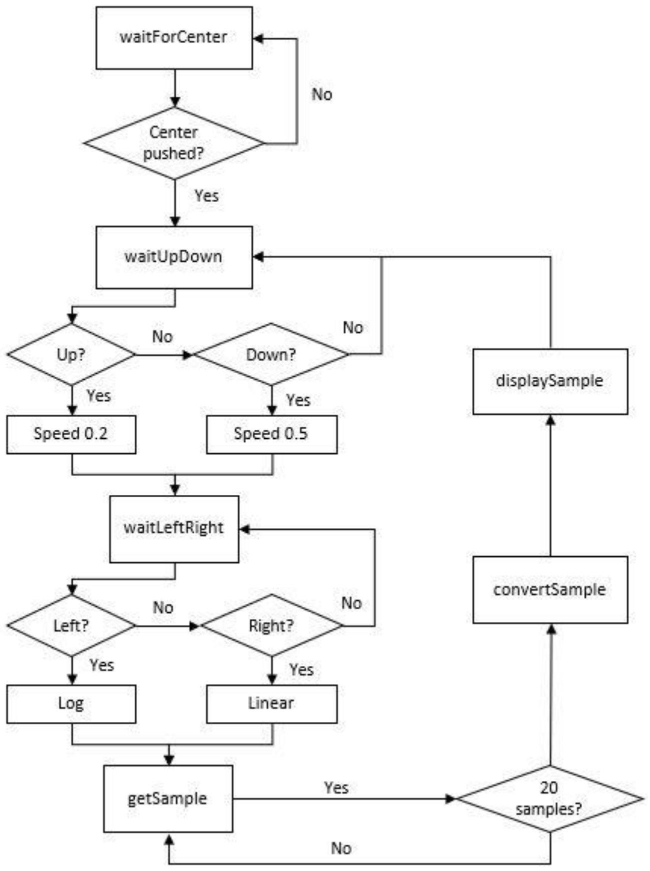

Figure 3: Overview of the algorithm

The main function call is written in C, calling subsequent functions depicted by the block diagram. The functions *waitForCenter()* *,waitforUpDown()*, *waitForLeftRight()*, *convertSamples()*, and *displaySamples()* are written in Assembly.

The *getSample()* will receive 20 numbers in hexadecimal after the "$" is entered. If the character is invalid, it will restart the sequence.

Subsequently, *convertSamples()* is called. In linear conversion, since each number is 2 bytes – 16 bits data. To select the first 3 bits of the last 10 bits, we rotate entire data 7 times. Then, clear all other bits and store the remained value (0-7) as processed data. In logarithmic conversion, similarly, we rotate left 6 times to bring last 10 bits front. Then, count the number of additional times we have to rotate left until the MSB is a "1". Store the count as the processed data.

The *displaySamples()* takes all processed data and turn on LEDs accordingly. The LEDs turn on from LED 4 (bottom left), going clockwise to LED 8 (bottom right). The display time between 2 consecutive processed data is determined by a hardware delay. Hardware delay is produced by timing the 2 interrupts from MSP430 internal timer in UP mode.

III. RESULT

A. Test data and expected results

The functionality of the project is tested using 4 sets of test datas, which are sinusoidal with different frequencies. The results from the program matches with the expected result tables for linear and logarithmic conversion

Sample #	Data 1 (HEX)	Data 2 (HEX)	Data 3 (HEX)	Data 4 (HEX)
001	$171	$112	$0BD	$07E
002	0ED	05B	00B	005
003	07F	004	03A	0E6
004	030	021	134	288
005	006	0AC	28A	3CD
006	004	185	3A2	3D5
007	02B	279	3FE	29C
008	077	352	375	0F8
009	0E2	3DE	244	008
010	164	3FB	0F5	071
011	1F2	3A5	01D	1EB
012	281	2EE	01E	373
013	306	200	0F7	3FD
014	377	113	247	329
015	3C9	05B	377	18A
016	3F7	004	3FE	03B
017	3FC	021	3A0	022
018	3DA	0AB	287	150
019	391	184	132	2F6
020	328	279	038	3F3

Table III: Four testing datasets

Sample #	Data 1 (linear/ log)	Data 2 (linear/ log)	Data 3 (linear/ log)	Data 4 (linear/ log)
001	3/7	3/7	2/6	1/5
002	2/6	1/5	1/2	1/1
003	1/5	1/1	1/4	2/6
004	1/4	1/4	3/7	6/8
005	1/1	2/6	6/8	8/8
006	1/1	4/7	8/8	8/8
007	1/4	5/8	8/8	6/8
008	1/5	7/8	7/8	2/8
009	2/6	8/8	5/8	1/2
010	3/7	8/8	2/6	1/5
011	4/7	8/8	1/3	4/7
012	6/8	6/8	1/3	7/8
013	7/8	5/8	2/6	8/8
014	7/8	3/7	5/8	7/8
015	8/8	1/5	7/8	4/7
016	8/8	1/1	7/8	4/4
017	8/8	1/4	8/8	1/4
018	8/8	2/6	6/8	3/7
019	8/8	4/7	3/7	6/8
020	7/8	5/8	1/4	8/8

Table IV: Expected results for linear/log conversion

The results consist of 2 parts. The data conversion will be printed out the terminal, and LEDs are turned on in sequence. We can visually observe the effect of pressing Up/Down/Left/Right sensors from the LEDs sequence and the speed of the display. After the display, the program prompts user for sensors presses and input. If the center sensor is pressed during the display, the program successfully loops back and prompt for Up/Down sensors press.

B. *Result plots*

The testing data and its processed result can be plotted and viewed as waveform.

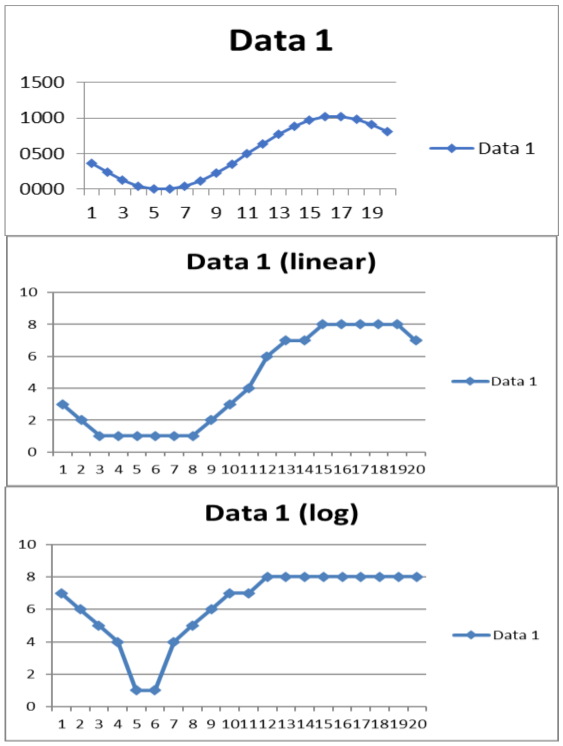

Figure 4: Waveform for Data 1

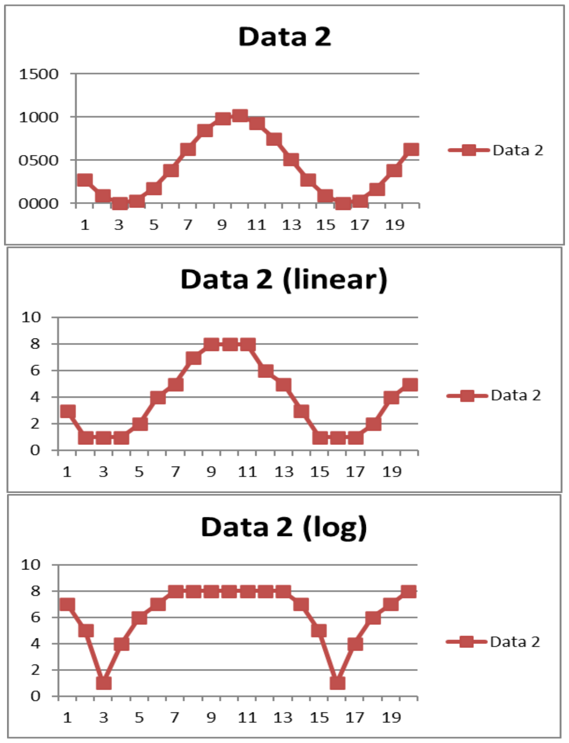

Figure 5: Waveform for Data 2

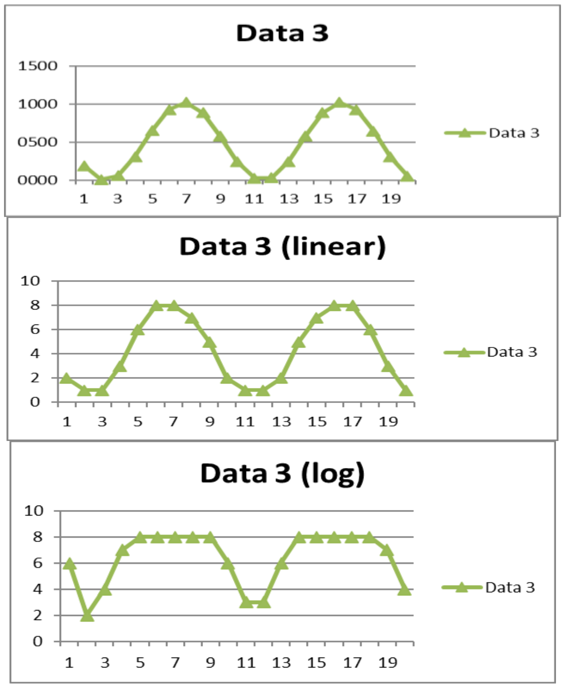

Figure 8: Waveform for Data 3

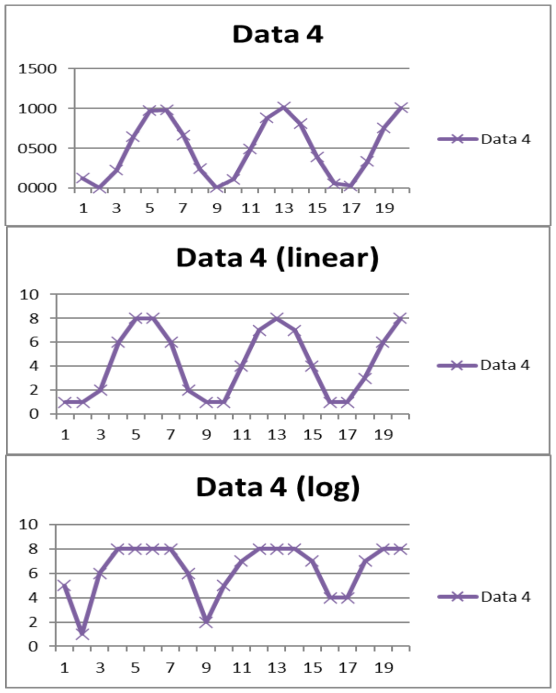

Figure 6: Waveform for Data 4

IV. CONCLUSION

From the plots, the logarithmic conversion has difference spacing than linear conversion: data points cluster more towards the higher end value. This is reasonable since the number must increase more to double. The project has demonstrated the use of UART communication and GPIO to external devices. With these two functions of a microcontroller, we can develop the program to handle more computation and output to a LCD screen.

V. REFERENCE

[1] Kate Gleason College of Engineering, *EEEE-420 Lab 11_HomeVersion*, Spring 2021, Rochester Institute of Technology, lab handout

[2] Texas Instruments, *MSP430x2xxx family user's guide,* July 2013