1

MSP-430 Signal Sampling Scale

Brandon Zeng
Rochester Institute of Technology
EEEE-420 Embedded Systems Design

***Abstract*— In signal processing, sampling is when a discrete signal is extracted from a continuous signal. The MSP430G2553 is the microcontroller that is used for receiving, sampling, and converting data. Using the CAPTOUCH board attached to the microcontroller allows for user and output interface. The software implementation of this design creates the converted samples by communicating to the microcontroller through the UART and touchpad, functions written on CCS and displaying it through the UART and CAPTOUCH board.**

I. INTRODUCTION

Digital Signal Processing (DSP) takes continuous real world signals and converts it to discrete signals which can be analyzed and manipulated. These signals that have been converted can be used for different purposes such as quicker and easier computation.

Typically, analog to digital converters are used to transmute analog signals to digital signals. In these converters, there are different resolutions that can correspond with each designed device. The resolution determines the quantization error of the sampled signal and how well the signal to noise ratio is for the ADC.

The signal sampling scale written in this project focuses on two conversions that have 8-bit resolutions: linear and logarithmic. The software implemented in pair with the hardware allows for the signals to be obtained by the microcontroller, converted and then outputted back out to the user visually through the UART and a LED display.

The MSP-430's GPIO channels for the UART input and output, hardware clocks and assembly and C code is used to implement the functions and systems required for the conversions. For this project, the user is prompted to start the program, choose the display rate and the conversion scale. Pin 1 is used as a I/O for the data received and transmitted through the UART. The data is run through the designed program which converts it into the specified conversion scale. The output of the conversion is then transmitted from the microcontroller to the UART and the captouch board for display.

1

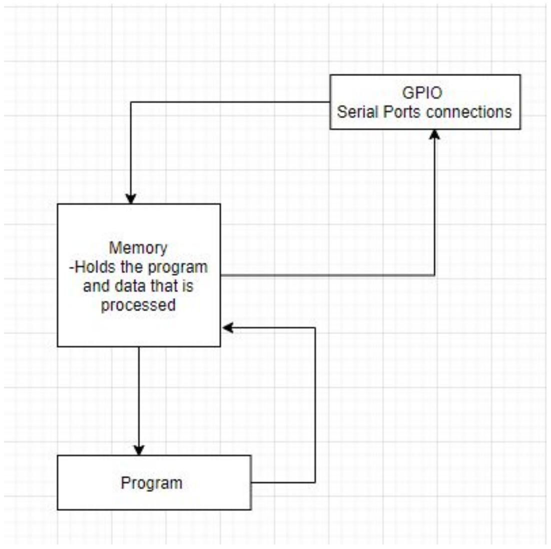

Figure 1: Simple diagram of ADC converter

II. Program Design

The design of the program begins by prompting the user to touch the center of the captouch board. This leads to the next prompt which is for the user to touch either the top or bottom of the captouch sensors. The top sensor will tell the program to display the converted samples at a rate of 0.2 seconds while the bottom sensor will display samples at a rate of 0.5 seconds. The user will then be prompted to touch the left or right sensors. The left sensor will convert the samples on a linear scale, and the right sensor will convert the samples on a logarithmic scale.

After giving the program all the conversion specifications, the user will be able to communicate to the MSP-430 using the UART. Up to 20 samples can be inputted into the terminal which is received and converted and transmitted back to the user via the terminal and touchpad board.

The converted samples will be converted using the linear or logarithmic algorithm and each converted sample will be displayed on the terminal. The value of the converted sample will correspond directly to how many LED's will turn on on the captouch board.

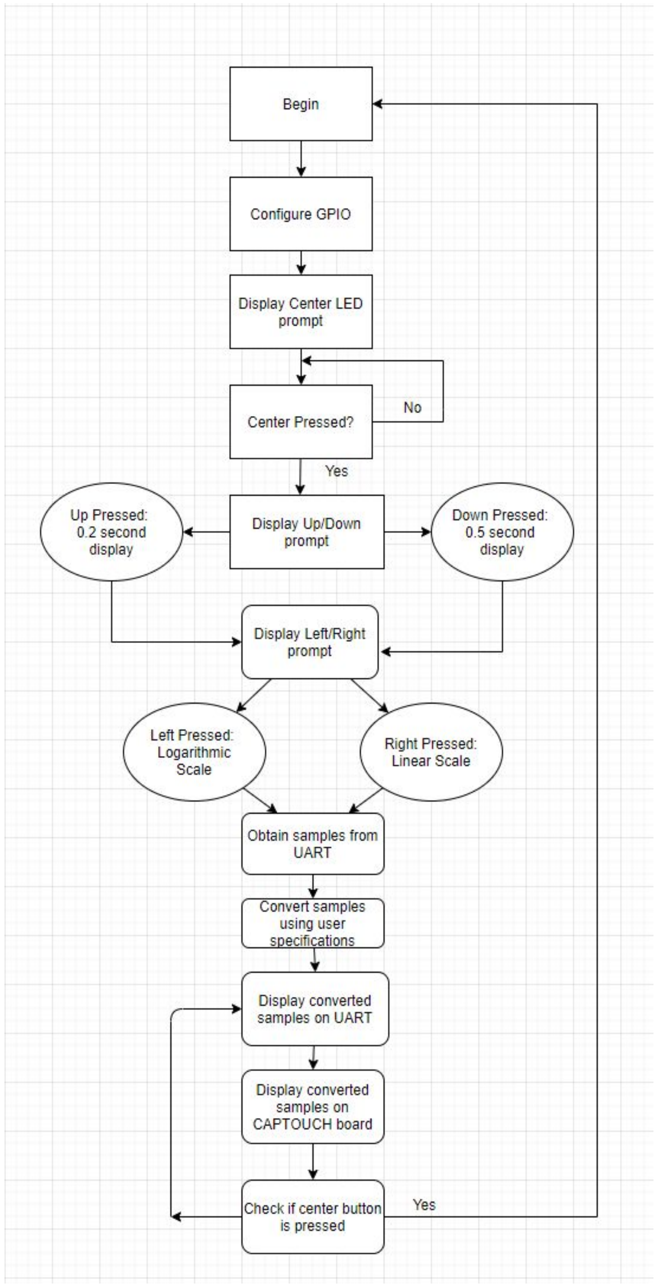
Begin
Configure GPIO
Display Center LED prompt
Center Pressed?
No
Yes
Up Pressed: 0.2 second display
Display Up/Down prompt
Down Pressed: 0.5 second display
Display Left/Right prompt
Left Pressed: Logarithmic Scale
Right Pressed: Linear Scale
Obtain samples from UART
Convert samples using user specifications
Display converted samples on UART
Display converted samples on CAPTOUCH board
Check if center button is pressed
Yes

Figure 2: Flowchart of the program design

II. Conversion Algorithm Design

Algorithm for linear conversion:

- Take the 3 most significant bits of the 10-bit sample
- Given a 10-bit sample, shift the sample 7 times to the right to mask the sample.
- This leaves the 3 most significant bits as the only bits left.
- These 3 bits represent the converted value and how many LEDs will turn on in the display.
- This is a linear conversion because we look at a 3-bit binary number that goes from 0 to 7.

Algorithm for logarithmic conversion:

- Look at the most significant bit of the 10-bit sample
- If the MSB of the sample is not 1, then shift left until the MSB is 1.
- The maximum amount of shifts is 8 because the last two bits of the sample are don't care conditions.
- Taking the amount of shifts subtracted from 8 is the converted value of the log conversion.
- This is a log conversion because everytime the sample is shifted left, the sample is increased by a power of 2.

III. Setup of Hardware

Pin 1 in the setup loop is set as outputs. For the LEDs only pins 1.3 to 1.7 are used to control the output of 8 LEDs so pins 1.1 and 1.2 can be used for the UART.

The UART controls communication of data from the terminal and sending it to the microcontroller for the program to run. Pins 1.1 and 1.2 of the MSP-430 were set as the TX/RX of the microcontroller to the terminal.

The MSP-430 has multiple hardware timers, but TimerA is used in this project. TimerA is configured to up mode where the TimerA register (TAR) will count up to TACCR0 and trigger an interrupt.

IV. Setup of Embedded Software

The MSP-430 uses embedded C and assembly to develop the program. The main method of the entire program is in C and calls functions written in assembly. When the board is powered up and the program starts, the MSP-430 sets up pin 1 to its proper direction for outputs and specific pins for the UART communication. It also measures the base value of all 5 sensors on the captouch board and stores it in an array for later use.

The program is configured to begin when the center is touched. The program detects if a sensor is pressed by taking a measured_latest value. If this value is lower than the measured_base value then we know that a sensor is pressed.

The waitForUpDown applies this same technique to check which sensor is pressed. A 1 will be stored in a register for 0.2s and 2 for 0.5s.

The waitForLeftRight also follows this technique which stores a 1 for the log conversion and 2 for a linear conversion. These values will be used later on to guide the program to make the proper conversions and display rate corresponding to the user's actions.

An interrupt is triggered after TAR counts up 0x6B6C for 0.2s or 0xFFFF for 0.5s. In the setup, interrupts are enabled and the timer will trigger an interrupt at the TACCR0 values. When an interrupt is triggered, it'll go into the interrupt and set a SW_flag high and turn off the clock. This flag will let the program know that an interrupt is triggered and return to wherever the program was left off. The clock is also disabled so that it doesn't count up to TACCR0 again and trigger an interrupt before transmitting the conversion to the terminal and then displaying the LEDs. If the timer isn't turned off, the program will continuously trigger interrupts before the display is finished and we will fall into a ISR trap.

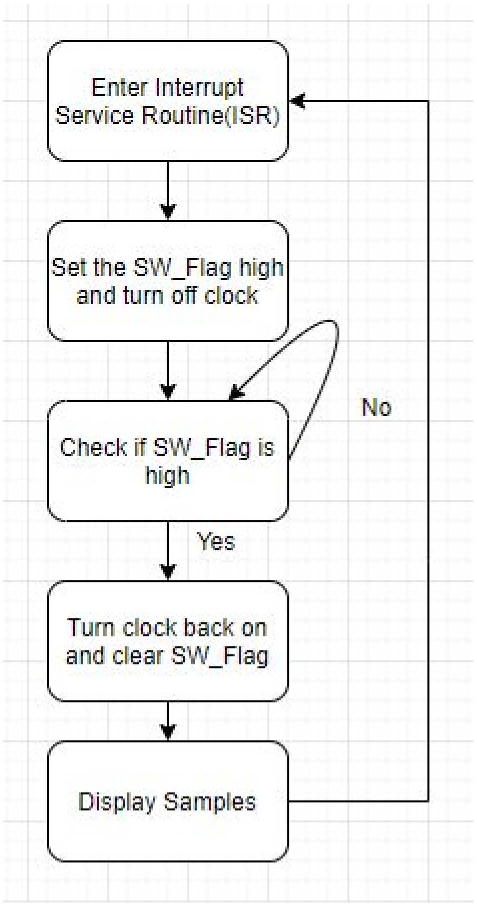
Enter Interrupt Service Routine(ISR)
Set the SW_Flag high and turn off clock
Check if SW_Flag is high
No
Yes
Turn clock back on and clear SW_Flag
Display Samples

Figure 3: Interrupt Service Routine

The display loop first checks if the user pressed the top or bottom sensor. This is done by comparing the value stored in the register earlier. The conversion method is also checked by comparing the corresponding register that has the value stored. Then the program constantly checks if the SW_Flag is high. This flag will never go high until the interrupt is triggered which sets it high. Once it goes high, the program will clear the SW_Flag, transmit the converted sample to the terminal for display, display the number of LEDs and then turn the clock back on. This allows for the next cycle of samples to begin displaying after the clock reaches the selected display rate. This is ran until all 20 samples are displayed and then will continuously run all 20 samples from the beginning again. After each sample is displayed, the program checks if the middle sensor is pressed. If it is pressed by the user, the program will immediately jump to the beginning of the main method which restarts the whole thing.

V. Results

The program was tested by inputting any group of 20 samples. The program I worked on was unable to display the logarithmic samples on the captouch board because it falls into an ISR trap. This may be caused due to the reasons stated in the previous section where the clock is triggering an interrupt quicker than the display. This causes my program to constantly trigger interrupts before it can even display the converted value. The linear conversion works very well, where the converted values are displayed on the touchpad at the display rate that is chosen.

VI. Conclusion

A signal sampling scale is created by implementing two different conversion algorithms on the MSP-430. Although the program is not yet finished, the output of the linear scale conversion works well. With the embedded C and assembly code, the program can work in a quick and efficient way to provide the converted signals and display them on a visual interface.

REFERENCES

[1] C. Barrios EEEE-420 Lab 5 CAPTOUCH Pinout
[2] C. Barrios EEEE-420 Lab 11 Conversion Chart
[3] Texas Instruments MSP430x2xx User Guide