Zeynepnur ŞAHİNEL
2305399
25.01.2022

EE447 TERM PROJECT PRELIMINARY REPORT
Audio Frequency Based Stepper Motor Driver

In this project, a stepper motor drive system based on the frequency of the applied audio signal. The driver system has three subsystems being audio sampling, frequency detection, user interface respectively. Figure 1 shows the overall system diagram.

Audio Sampling Subsystem:
In this subsystem, initially the audio signal is converted to the electrical signals through the microphone module. And this module is used as input to the ADC module in TM4C123. The ADC Module continuously samples the audio signal and stores these samples in an array consisting of 256 elements. For this purpose, ADC is read using SysTick Interrupt Handler.

Frequency Detection Subsystem:
This subsystem is responsible for the calculation of the frequency using 256 samples array. While the frequency detection FFT (Fast Fourier Transform)is used. To implement FFT, arm_cfft_q15 function is used. Equation 1 shows the DFT (discrete Fourier transform)

$$X_k = \sum_{n=0}^{N-1} x_n e^{\frac{-j2\pi kn}{N}}, \quad k = 0, 1, \ldots, N-1 \qquad (1)$$

FFT is a computational algorithm that diminishes the computing time and complexity. This transform takes N (N=256 for this case) element sequence input in the time domain, then it converts the sequence into frequency domain with a complex value. In this subsystem after taking FFT of the 256 sample array, complex magnitudes of the elements are computed. Then the index having the highest magnitude is chosen and given as an output to the same memory location as the input array.

User Interface Subsystem:
In this subsystem, output relations are determined such as according to the coming frequency magnitude value a led is lightened, stepper motor speed has changed and the results are displayed on. Here, at low frequencies Red, middle frequencies Green and the high frequencies Blue led is on. If no frequency is detected led is going to be turned off. Moreover, stepper motor directions are defined through the switches on the board. And at the beginning, threshold values can be set using a potentiometer or 4x4 keypad module.

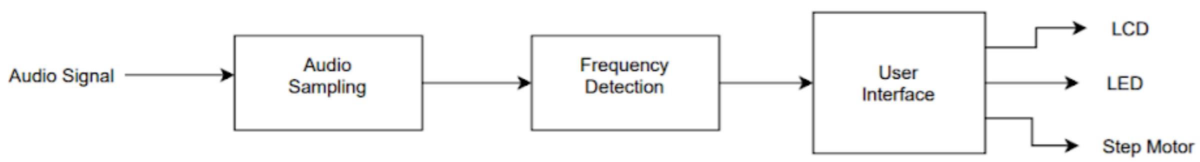

Figure 1. Overall System Block Diagram

Zeynepnur ŞAHİNEL
2305399 25.01.2022

```
;*************************Pseudo Code********************************
;0. Initialization subroutines for the GPIO Ports, ADC Module and setting thresholds
 Start   BL  Port_Init
         BL  Inıt_SysTicK;
         BL  ADC_Init
         BL  Thresholds ;Set thresholds for the frequency (amplitude, low and high freq.)

         LDR CMSIS_memory; for the sample array (ptr)
         LDR ADC_RIS         ; interrrupt address
         LDR ADC_FIFO        ; sample result address
         LDR ADC_PSSI        ; sample sequence initiate address
         LDR ADC_ISC         ;interrupt status clear

         ArrayCounter=0      ;for the magnitude and the highest element computation
;1. Wait for SysTick handler to store 256 readings in the array
Loop
 Sampling is done at the MySysTickHandler, not in the main part
/*Smpl  Initiate sampling by enabling proper sequencer
        Poll until sample complete (Check RIS bit 3)
        Clear the flag
        Load FIFO Reg value to the CMSIS memory
        CMSIS Memory+4; ptr=ptr+4
        B Smpl */
;2. Call the FFT subroutine
        BL  arm_cfft_q15
;3. Compute the complex magnitudes of elements of the frequency domain sequence
Loop1 load the first half-word and multiplied by itself (real part)
      load the second half-word and multiplied by itself (imaginary part)
      Then add two of them
      ArrayCounter++;
      if   ArrayCounter ==256
           ArrayCounter=0
           exit the Loop1
      B Loop1
;4. Find index and magnitude of the dominant (highest-magnitude) frequency component
      Load the first element as max
      Create pointer (ptr) to the memory
Loop2  Load the next data from memory
       If the next data > (ptr)
            Update max=the next data
       ArrayCounter++
       Ptr=Ptr+4;
       If ArrayCounter==256
            Update AraryCounter =0
            Load CMSIS_memory; ptr
            Update memory; max=(ptr)
            exit the Loop2
        Else B Loop2
;5. Convert that index to frequency
```

```
            Load frequency magnitude from CIMS_memory
;6. If that magnitude is larger than a threshold, turn on the corresponding LED and update
the stepper motor speed
            If (freq magnitude < Low Freq. Threshold)
                  RED led on
                  Increase the TAILR   ;to slower the stepper motor
            Else if (freq. mag>Low Freq. T. && mag< high mag)
                  GREEN led on
                  Increase less the TAILR
            Else
                  BLUE led on
                  Decrease  the TAILR
;7. If more than a second has elapsed after the last LCD update, show the frequency and
magnitude values on LCD
            B  Delay2Sec
            Send Info to the LCD
;8. Loop back to 1.
          B   Loop
```

Figure 2. Pseudo Code

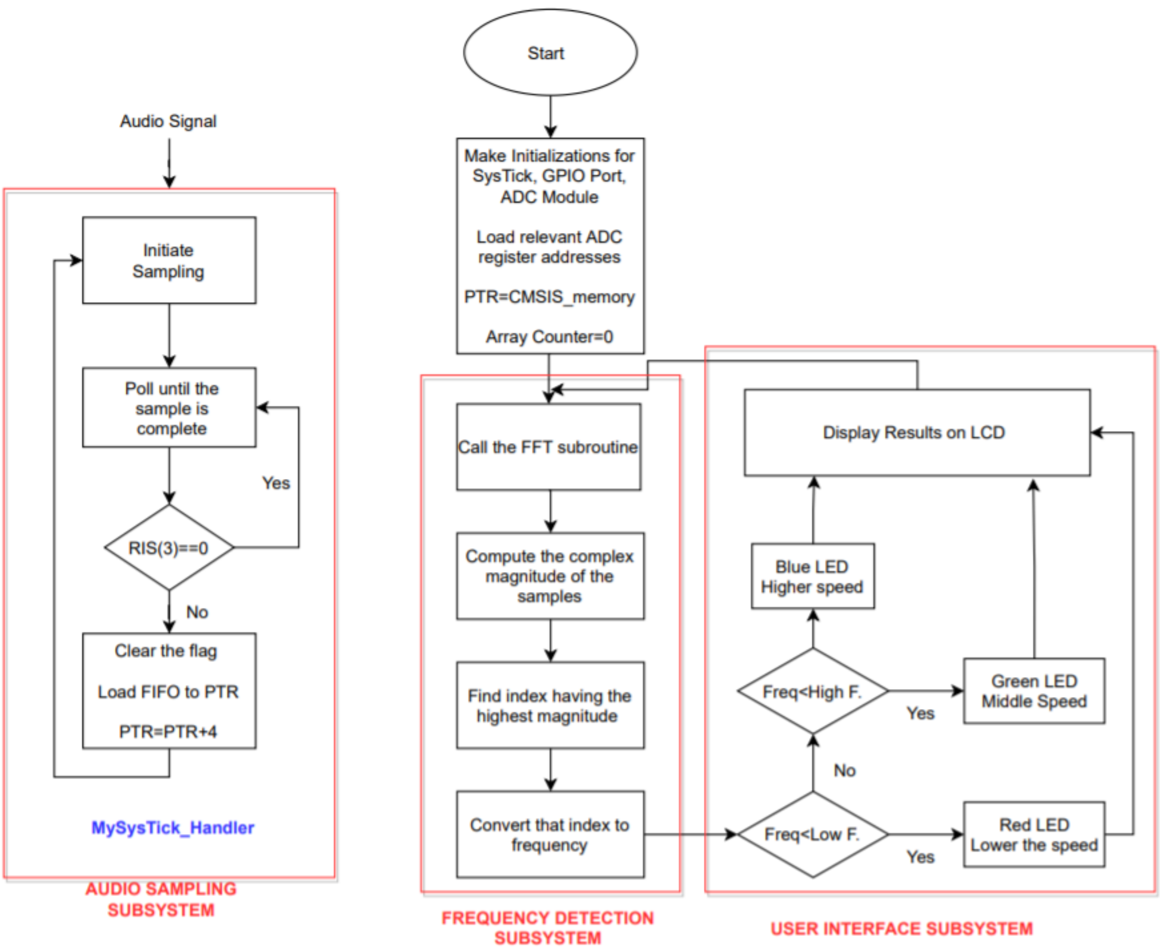

Figure 3. Flow Chart of the Overall System