

**Kauno technologijos universitetas**

Elektros ir elektronikos fakultetas

**FFT**

5 Laboratorinis darbas

|  |
| --- |
|  |
| **Žygimantas Marma, EMEI-2 gr.**  Studentas |
|  |
| **Doc. Šarūnas Kilius**  Dėstytojas |
|  |

**Kaunas, 2023**

**Kaunas, 2023**

Turinys

[1. Programos analizė 4](#_Toc151845491)

[1.1. Programos veikimo algoritmas 4](#_Toc151845492)

[2. Filtrų impulsinės charakteristikos 6](#_Toc151845493)

[3. Audio efektai 7](#_Toc151845494)

[4. Išvados 8](#_Toc151845495)

[5. Priedai 9](#_Toc151845496)

**Darbo tikslas**: Susipažinti su FFT filtravimo metodika.

**Laboratorinio darbo uždaviniai:**

1. Sugeneruoti poli-harmoninį signalą. Modifikuoti FFT filtravimo funkciją taip, kad:  
a. Būtų filtruojamas vienas pasirinkto dažnio signalas.  
b. Būtų filtruojami visi skirtingų dažnių signalai, paliekant didžiausios amplitudės signalą

2. Išmatuoti FFT filtro greitaveiką esant skirtingiems signalo masyvo dydžiams bei skirtingiems kompiliatoriaus optimizavimo lygiams.  
3. Sugeneruoti kintančio dažnio (Chirp) signalą bei apskaičiuoti jo dažnių spektrą.

# Poliharmoniins signalas

## Programos veikimo algoritmas

Filtravimo testavimui buvo sugeneruotas signalas, kurio dedamosios yra 1kHz, 3kHz ir 5kHz, kaip pavaizduota programiniu kodu toliau.

**1 lentelė.** Sugeneruotas signalas

|  |
| --- |
| /\*\*\* Test Signal generation \*\*\*/  **for**(**int** index = 0; index < SAMPLES\_QTY; index++)  {  //Test signal 1000 Hz + 3000 Hz + 5000 Hz  InputArray[index] = ( 20000\***sin**((FREQUENCY/SAMPLING\_FREQ)\*2\*PI\*index)  + 10000\***sin**(3\*(FREQUENCY/SAMPLING\_FREQ)\*2\*PI\*index)  + 30000\***sin**(5\*(FREQUENCY/SAMPLING\_FREQ)\*2\*PI\*index)  );  } |

1 pav. Signalo dažnių amplitudės

Kaip galima matyti 1 pav. signalą sudaro xxxxx nes tai atitinka yyy indeksuss blaha

FFT sudarytas is kosinoso todel ir gaunasi -90 laipsniu

## Pasirinkto dažnio filtravimas

a. Būtų filtruojamas vienas pasirinkto dažnio signalas.

2 lentelė.

|  |
| --- |
|  |

## Maksimalios amplidudės signalo filtravimas

Toliau FFT filtravim funkcija buvo pakeista kad butu filtruojami visi signalai išskyrus didžiausios amplitudes. Tam reikėjo rasti masyvo maksimuma su arm\_max\_f32() funkcija. Toliau pateiktas maksimalios reikšmės FFT filtro funkcija.

3 lentelė. Funkcija skirta pašalinti visas signalo dedamasias išskyrus didžiausios amplitudės

|  |
| --- |
| **void** FFT\_filtering(**void**)  {  float32\_t FFTMaxValue;  uint32\_t MaxIndex;  //rFFT initialization. Supported FFT Lengths are 32, 64, 128, 256, 512, 1024, 2048, 4096.  arm\_rfft\_fast\_init\_f32(&rFFT\_S, SAMPLES\_QTY);  //rFFT process. 0 = RFFT, 1 = RIFFT  arm\_rfft\_fast\_f32(&rFFT\_S, InputArray, FFTOutputArray, 0);//direct FFT calculation  //Calculate the Magnitude  arm\_cmplx\_mag\_f32(FFTOutputArray, FFTMagnitude, SAMPLES\_QTY/2);  //Find max value  arm\_max\_f32(&FFTMagnitude, SAMPLES\_QTY/2, &FFTMaxValue, &MaxIndex);  //Removing frequencies  **for**(**int** i=0; i<SAMPLES\_QTY; i++)  {  **if**(i != (MaxIndex\*2)+1)  FFTOutputArray[i] = 0;  }  //Calculate the Magnitude  arm\_cmplx\_mag\_f32(FFTOutputArray, FFTMagnitude, SAMPLES\_QTY/2);  //rFFT process. 0 = RFFT, 1 = RIFFT  arm\_rfft\_fast\_f32(&rFFT\_S, FFTOutputArray, OutputSignalArray, 1); //inverse FFT calculation  } |

2 pav. Signalas prieš filtravima (kairėje) ir signalas po filtravimo paliekant didžiausios amplitudes dedamają (dešinėje).

# Greitaveikos tyrimas

Kaip priklauso nuo masyvo dydžio ir optimizacijos laipsnio

Galiausiai buvo atlikta laiko analizė, palyginti ST FIR filtro greitaveiką su paprasta vidurkio funkcija pritaikant slenkančio vidurkio filtrą.

**4 lentelė.** Programos fragmentas ištestuoti filtrų greitaveiką

|  |
| --- |
| **void** **countInit**(**void**) {  // Configure the DWT cycle counter  **if** (!(CoreDebug->DEMCR & CoreDebug\_DEMCR\_TRCENA\_Msk)) {  CoreDebug->DEMCR |= CoreDebug\_DEMCR\_TRCENA\_Msk;  DWT->CYCCNT = 0;  }  }  //Variables  uint32\_t count\_tic = 0;  uint32\_t count\_us = 0;  **int** run\_times = 0;  ....  **while** (1)  {  HAL\_GPIO\_TogglePin(LD4\_GPIO\_Port, LD4\_Pin);  HAL\_Delay(1);  /\* USER CODE END WHILE \*/  MX\_USB\_HOST\_Process();  run\_times++;  **if**(run\_times> 1000)  {  count\_tic = DWT->CYCCNT;  count\_us = count\_tic / (SystemCoreClock/1000000U); //time in us  } |

**5 lentelė.** Rezultatų palyginimas

|  |  |  |
| --- | --- | --- |
| **Filtro tipas** | **Laikas, µs** | **Pokytis, %** |
| Vidurkio funkcija | 2’692’593 | - |
| ST *arm\_fir\_f32()* api | 3’413’260 | +26,7 % |

# Kintančio dažnio (*Chirp*) signalo analizė

Funkcija sugeneruoti Chrip:

**6 lentelė** Chrip generavimo funkcija

|  |
| --- |
| float32\_t **generateChirp**(float32\_t w1, float32\_t w2, float32\_t A, float32\_t M, float32\_t time)  {  float32\_t res;  res=A\***cos**(w1\*time+(w2-w1)\*time\*time/(2\*M));  **return** res;  } |

Gautas daznis

# Išvados

1. Laboratorinio darbo metu buvo sugeneruotas poli-harmoninis signalas, jam atlikta FFT analizė nustatant signalo dažnines dedamasias ir jų amplitudes.
2. Darbe buvo atliktas signalo filtravimas naudojant FFT metodiką, sukurtas vienas filtras signale palikti tikne didesnio nei 1kHz dedamassias ir kitas filtras signale palikti tik didžiausios amplitudės dažnio dedamąja.

Darbe atlikta analizė, kurioje tiriama kaip keičiasi FFT filtro greitaveika esant kitokiems optimizavimo lygiams ir nustatant skirtingus signalo masyvo dydžius.

Gauti rezultatai parode, kad xxxx

1. Galiausiai buvo sukurta funkcija generuoti Chirp signalą ir jam atlikta analizė

# Priedai

**7 lentelė.** MATLAB programa sukurtų filtrų koeficientų konvertavimui į C kodą

|  |
| --- |
| % FIR testas  % Filtro koeficientai saugomi faile    clear;  % HD - this is a generated MATLAB filter function  % Hd = FIR\_filter\_30ord; % example  Hd = lowpassFinal;  %  ylabel('Amplitude');    % Display only positive frequencies  axis([0 Fs/2 0 max(P\_y)]);  end |

**8 lentelė** pagrindinis laboratorinio darbo programinis kodas

|  |
| --- |
| #define SAMPLING\_FREQ 48000.0   //BSP\_audio\_out sampling frequency  //--- FIR related PD ---  #define TEST\_LENGTH\_SAMPLES 1024  //Test signal length                    //Turi dalintis is BLOCK\_SIZE! |