

מסנן מגראט 50/60 הרץ: הסרת רעשי קו מתח

מצגת זו עוסקת במסנני מגראט (Notch Filter) בתדר 50/60 הרץ. נדון בשימושים, ביצירנות, במגבלות ובשיקול העיצוב.
מנחים : יהונתן לוי ודניאל כהן

מהו מסנן מגראעת?

סוג של מסנן תדרים

מסנן המעביר את רוב התדרים, אך דוחה טווח ספציפי.



מיועד לסינון הפרעות

משמש להחלשת אותות לא רצויים סביב תדר מסוים.



תגוננות תדר חדה

מציג הנחתה חדה מאוד סביב תדר המגראעת הספציפי.





מדוע 60/50 הרץ?

חדרת רعش



רעש חזק של מיכשירים דרך קווי חשמל וشنאים.

תדר רשת החשמל



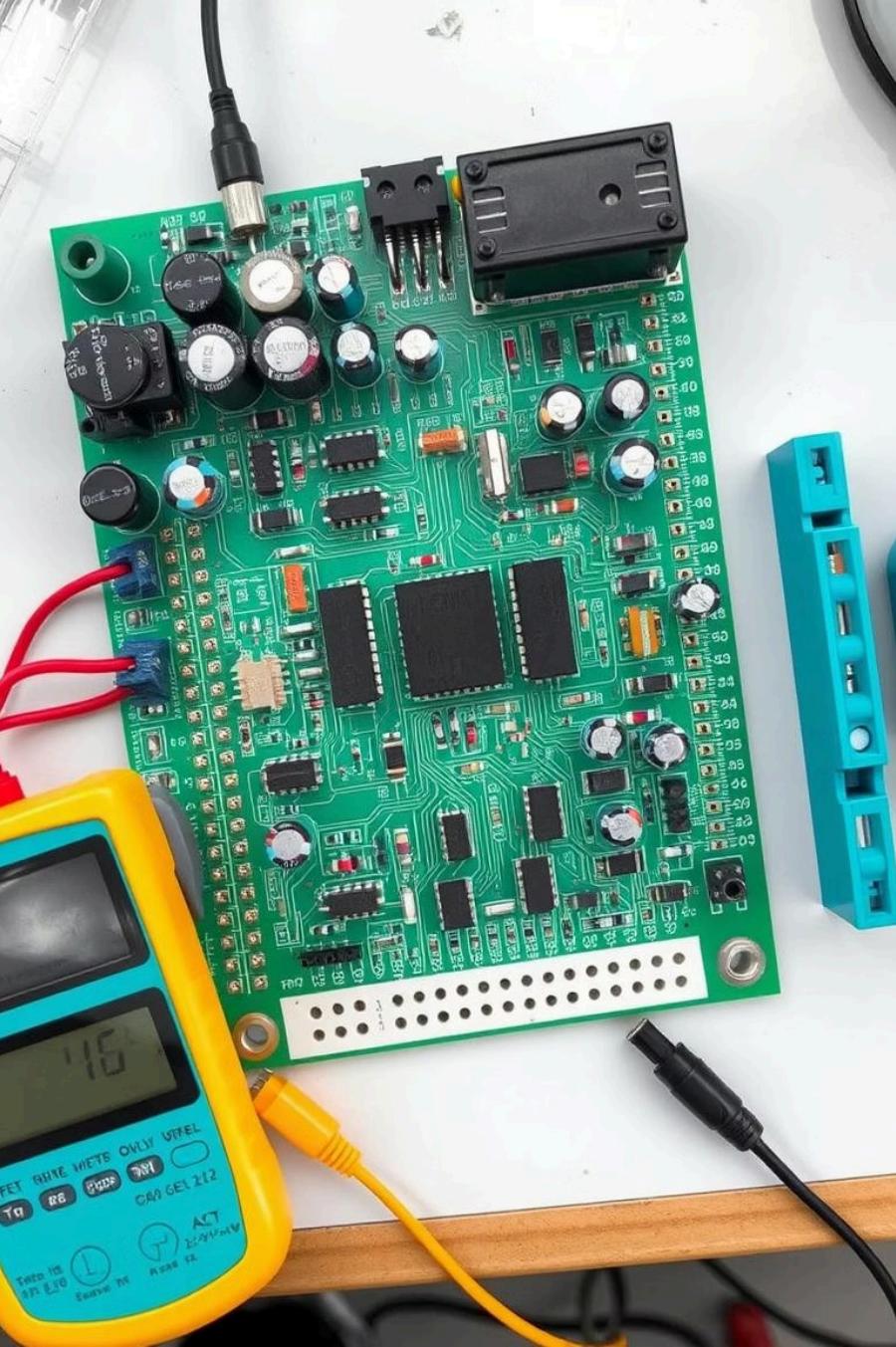
תדר סטנדרטי של 50 הרץ באירופה, 60 הרץ בצפון אמריקה.

השפעה על מדידות



ההפרעות משפיעות על מדידות רגניות, אודיו ומכשור רפואי.

שיקולי עיצוב



תדר מגעט מדויק

50 הרץ או 60 הרץ, בהתאם למיקום נאוגרפי.

רוחב פס מוגדר

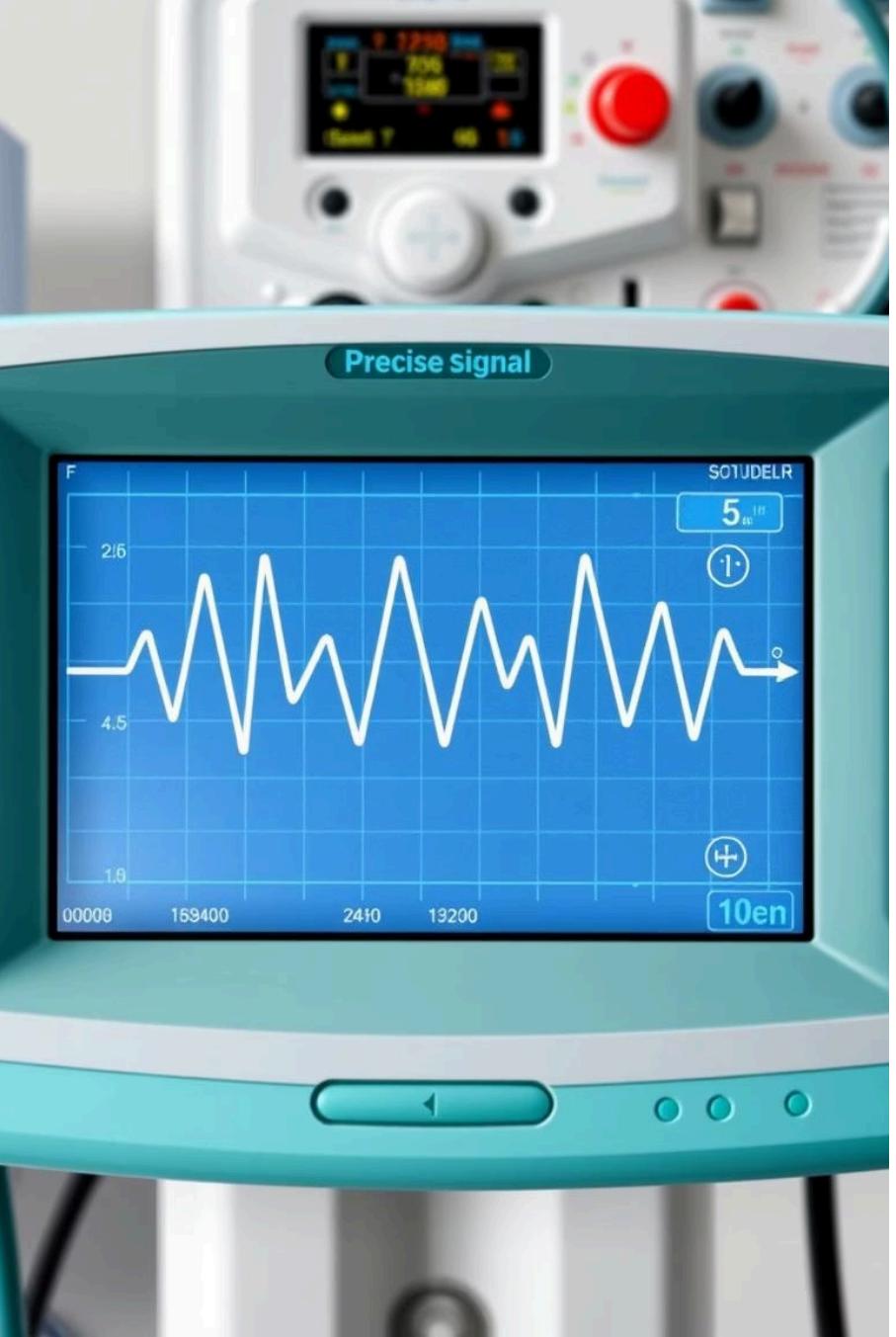
טווח התדרים המוחלט, צר להפרעות ממוקדות.

סדר המسانן

קובע את תלילות הדחיה, גובה יותר = חד יותר.

שיטת יישום

מעגלים אנלוגיים (מנברי-שרת) או עיבוד数 דיגיטלי (אלגוריתמים).



ישומים בעולם האמיתי

מכשור רפואי

אך"ג, EEG: שיפור
aicות האות ב-30%,
זיהוי פעילות מוחית
עדינה.

אודיומטר

הסרת זמזום מרשת
החשמל בהקלות,
שיפור יחס אות לרעש
ב-B20.

מכשירי בדיקה ומדידה

סוקטרומטרים,
אוסצילוסקופים: הגדלת
דיוק המדידה ב-2-5%.

מיחוש מסנן מגראעת דיגיטלי

מסנן מגראעת דיגיטלי מושם באמצעות אלגוריתמים. מקדמיו מחושבים במדוק, לרוב בעזרת MATLAB.

טהליק זה מאפשר שליטה אופטימלית ביצועי המסנן, כגון תדר מרכזי ורוחב פס. הגמישות הדיגיטל עלתה על אנלוגי.



שימוש הקוד ב-Arduino

```
1 #include <DueTimer.h>
2
3 const int dacPin = DAC1; // DAC1 = A13
4
5 const float pi = 3.14159;
6 const float sawFreq = 1.0; // תדר שוו מסור
7 const float sineFreq = 50.0; // תדר רעש
8 const int dacMax = 4095; // 12-bit
9
10 volatile float currentTime = 0.0; // מילוי כלובלי
11 const float dt = 0.001; // 1ms = 1kHz
12
13 void generateSignal() {
14     currentTime += dt;
15
16     // אונת שוו מסור
17     float saw = fmod(currentTime * sawFreq, 1.0);
18
19     // רעש סינוס
20     float noise = 0.05 * sin(2 * pi * sineFreq * currentTime);
21
22     // מיברג אונת
23     float signal = saw + noise;
24 }
```

```
25     // מילוי כלובלי והמרה ל-DAC
26     signal = constrain(signal, 0.0, 1.0);
27     int dacValue = signal * dacMax;
28     analogWrite(dacPin, dacValue);
29 }
30
31 void setup() {
32     analogWriteResolution(12); //設置 12 bit
33
34     // תקע הפעלה טריזמה 3 שיקרא את
35     Timer3.attachInterrupt(generateSignal);
36     Timer3.start(1000); // 1000 1 = 1kHz
37 }
38
39 void loop() {
40
41 }
```

שימוש הקוד ב-EasyPIC

```

// ===== TFT הגדרות לממד =====
char TFT_DataPort at LATE;                                // פורט TFT הנתונים לממד
sbit TFT_RST at LATD7_bit;                               // לממד Reset פין
sbit TFT_BLED at LATD2_bit;                             // תאורת רקם לממד
sbit TFT_RS at LATD9_bit;                               // Register Select
sbit TFT_CS at LATD10_bit;                            // Chip Select
sbit TFT_RD at LATD5_bit;                               // Read Enable
sbit TFT_WR at LATD4_bit;                               // Write Enable

10 char TFT_DataPort_Direction at TRISE;                // הגדרת כיוון לפורט
sbit TFT_RST_Direction at TRISD7_bit;                  // // Reset כיוון פין
sbit TFT_BLED_Direction at TRISD2_bit;                  // // תאורת רקם כיוון פין
sbit TFT_RS_Direction at TRISD9_bit;                  // // Register Select כיוון פין
sbit TFT_CS_Direction at TRISD10_bit;                 // // Chip Select כיוון פין
sbit TFT_RD_Direction at TRISD5_bit;                  // // Read כיוון פין
sbit TFT_WR_Direction at TRISD4_bit;                  // // Write כיוון פין

// ----- הגדרות כלליות -----
#define FS 1000.0                                         // (Hz) חדר דגימה
#define BUFFER_SIZE 3                                     // גודל הבuffer לפילטר

// ----- מקדמים Notch -----
const float notch_a0 = 0.9722;                           // a0 מקדם
const float notch_a1 = -1.8479;                          // a1 מקדם
const float notch_a2 = 0.9722;                           // a2 מקדם
const float notch_b1 = -1.8479;                          // b1 מקדם
const float notch_b2 = 0.9445;                           // b2 מקדם

// ----- משתנים גלובליים -----
30 float xBuf[BUFFER_SIZE] = {0};                         // ADC באפף לקולט
float yBuf[BUFFER_SIZE] = {0};                           // באפף לפלט מהפילטר
int index = 0;                                            // אינדקס בריינץ באפף

40 float filteredValue = 0;                                // ערך הפילטר האחרון
unsigned int adcRaw = 0;                                 //ADC דוגימה גולמית מה-
volatile int newSampleReady = 0;                          // דגל לזמןנות דגימה

unsigned int xPos = 0;                                   // צייר בפס 0 מיקום
int prevYRaw = 150;                                    // קודם לאות הגולמי Y
int prevYfilt = 200;                                    // קודם לאות המסונן Y

// ----- TFT -----
void PrepareTFT(){
    TFT_BLED = 1;                                       // הדלקת תאורה אחוריית TFT
    TFT_Init_ILI9341_8bit(320, 240);                  // אבחזול גודל מסך 240x320
    TFT_Fill_Screen(CL_YELLOW);                        // מילוי רקם בעקבות
    TFT_Set_Pen(CL_BLUE, 1);                           // ש ברירת מחדל כחול
}

50 // ----- אתחול ADC0 -----
void PrepareADCChannel0(){
    AD1PCFG = 0xFFFF;                                  // הכנה אנלוגית RBO הגדרת
    JTAGEN_bit = 0;                                    // JTAG ביטול
    TRISB0_bit = 1;                                    // קלט RBO הגדרת ADC
    ADC1_Init();                                      // אתחולADC
    Delay_ms(100);                                    // זמן התיעבות
}

// ----- אתחול טירום -----
60 void InitTimer2_3(){                                 // הפעלת טירום 2 עם קדם-מחזור 1:1
    T2CON = 0x8008;                                  // טירום 3 כבוי
    T3CON = 0x0;                                    // איפוס מונח טירום 2
    TMR2 = 0;                                       // איפוס מונח טירום 3
    TMR3 = 0;                                       // (111) רמת מדיפות 7
    T3IP0_bit = 1;                                 // איפוס דגל פסיקה
    T3IP1_bit = 1;                                 // הפעלת פסיקה
    T3IP2_bit = 1;                                 // איפוס דגל פסיקה
    T3IF_bit = 0;                                    // הפעלת פסיקה
    T3IE_bit = 1;                                 // איפוס דגל הפסיקה

70 PR2 = 13568;                                    // טרולו PBCLK
PR3 = 12;                                         // סטם להשגת 1
}

// ----- פסיקה טירום: דגימה וסינון -----
80 void Timer2_3Interrupt() iv IVT_TIMER_3 illevel 7 ics ICS_SRS {
    float x;                                         // איפוס דגל הפסיקה
    float y;                                         // איפוס דגל הפסיקה
    T3IF_bit = 0;                                 // איפוס דגל הפסיקה

    adcRaw = ADC1_Get_Sample(0);                    // ADC דוגימה מה-
    x = (float)adcRaw;                            //ADC דוגימתה ל- float

    // ----- פונקציית פילטר -----
    y = notch_a0 * x +                            // שמיירת הדגימה
        notch_a1 * xBuf[(index + BUFFER_SIZE - 1) % BUFFER_SIZE] + // שמירת הפלט
        notch_a2 * xBuf[(index + BUFFER_SIZE - 2) % BUFFER_SIZE] - // שמירת הפלט האחרוני
        notch_b1 * yBuf[(index + BUFFER_SIZE - 1) % BUFFER_SIZE] - // סכום באינדקס דיבנג באפף
        notch_b2 * yBuf[(index + BUFFER_SIZE - 2) % BUFFER_SIZE]; // איתות שיש דגימה חדשה

90     xBuf[index] = x;                           // שמירת הדגימה
    yBuf[index] = y;                           // שמירת הפלט
    filteredValue = y;                          // שמירת הפלט האחרוני

    index = (index + 1) % BUFFER_SIZE;           // סכום באינדקס דיבנג באפף
    newSampleReady = 1;                          // איתות שיש דגימה חדשה
}

// ----- פונקציית MAIN -----
100 void main()
{
    PrepareTFT();                                // אתחול המסך
    PrepareADCChannel0();                         // אתחולADC

    ADC1_Get_Sample(0);                          // דגימה ראשונית
    InitTimer2_3();                             // אתחול טירום לפסיקה
    EnableInterrupts();                         // הפעלת פסיקות

    while (1){
        if (newSampleReady) {                     // אם יש דגימה חדשה
            int yRaw, yFilt;                   // איפוס דגל
            newSampleReady = 0;

            yRaw = 150 - (adcRaw >> 4);      // לעיור האות המסונן (כחול)
            yFilt = 200 - ((int)filteredValue >> 4); // המרת גם לאות המסונן

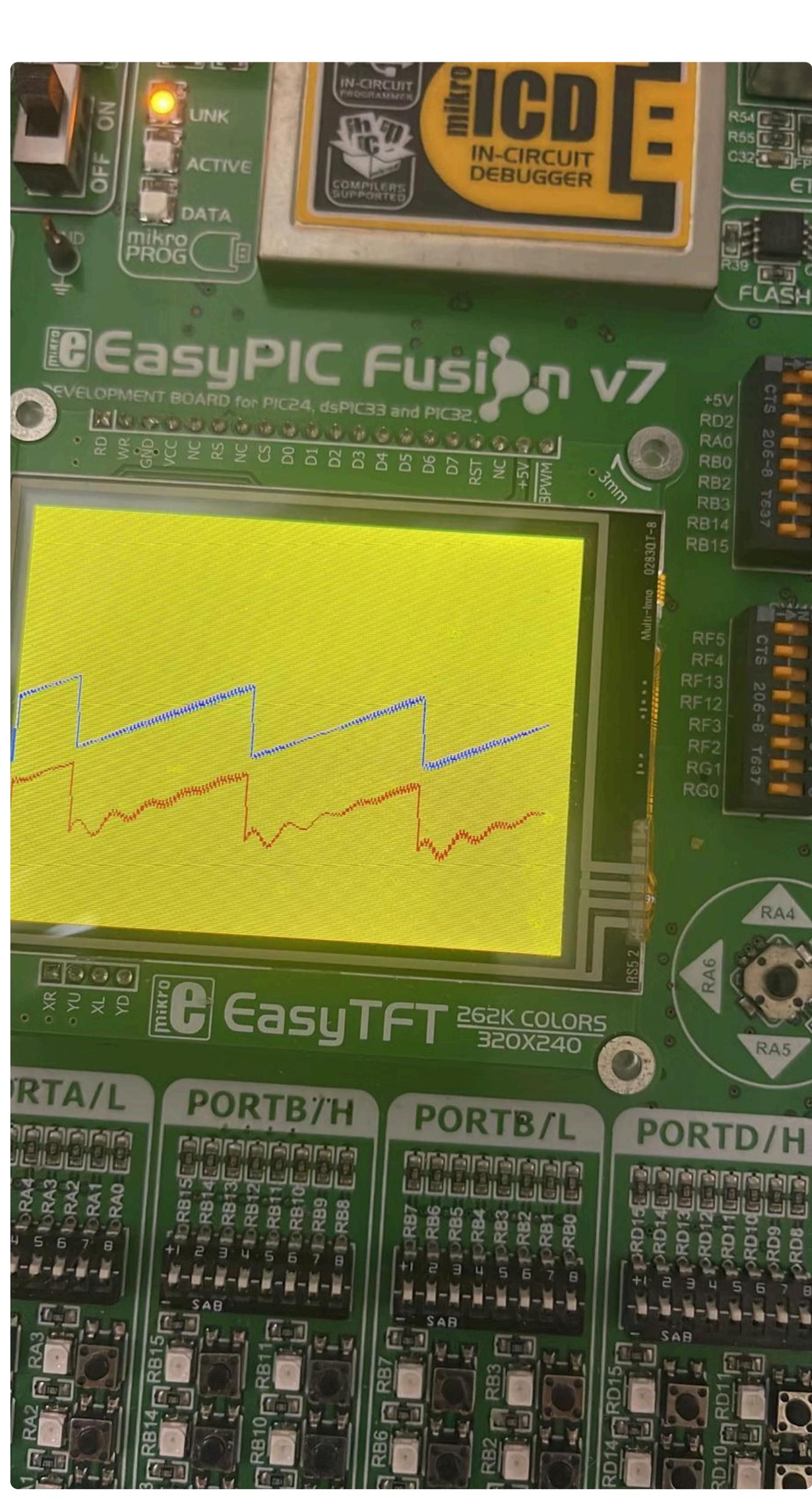
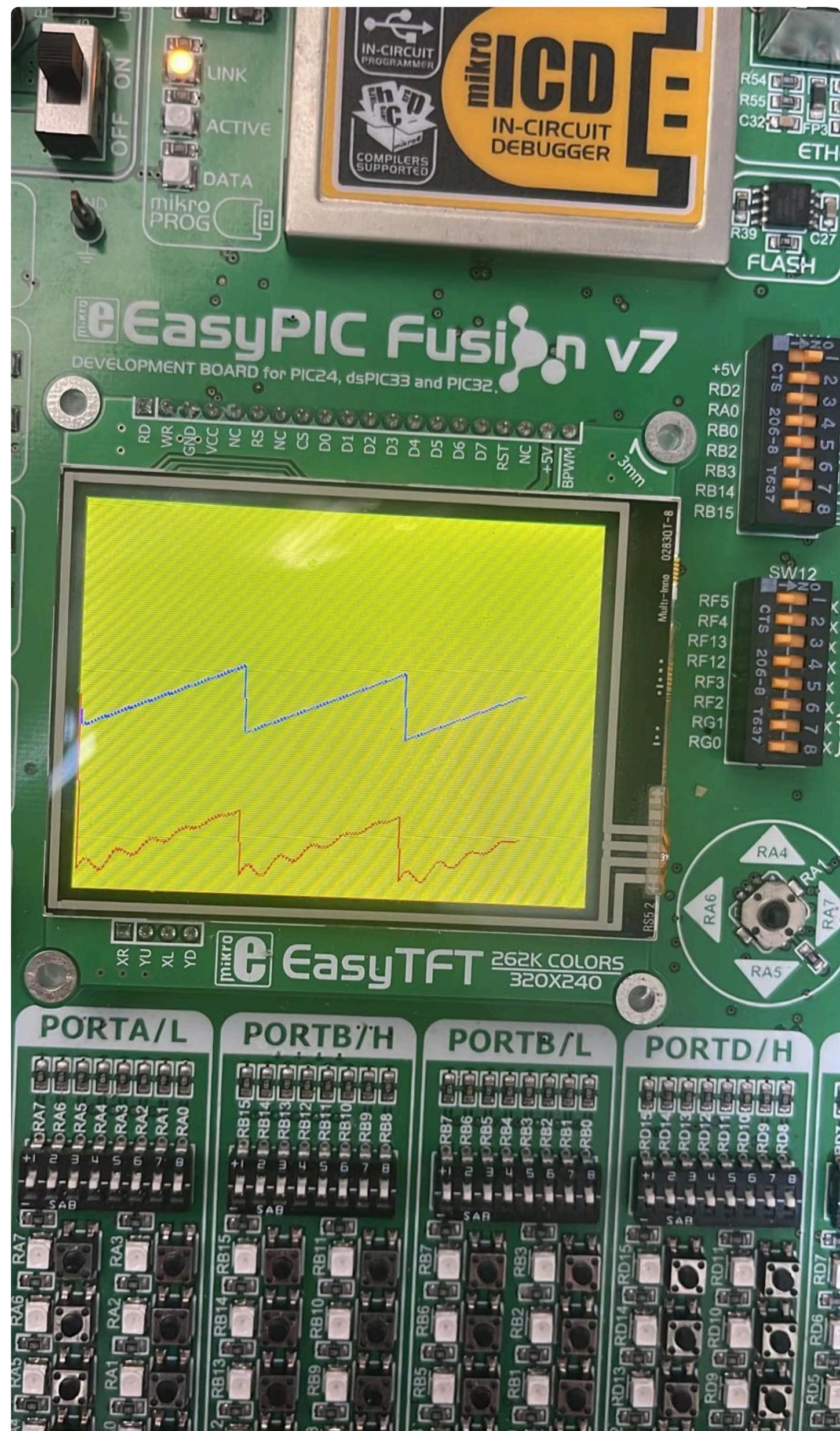
            if (xPos > 0) {                  // צייר האות המסונן (כחול)
                TFT_Set_Pen(CL_BLUE, 1);       // TFT_Set_Pen(CL_RED, 1);
                TFT_Line(xPos - 1, prevYRaw, xPos, yRaw); // TFT_Line(xPos - 1, prevYFilt, xPos, yFilt);
                prevYRaw = yRaw;
                prevYFilt = yFilt;
            }

            xPos++;                           // בפס X קידום
            if (xPos >= 320) {               // אם הגיע לסוף המסך
                xPos = 1;                      // איפוס X
                prevYRaw = 120;                // איפוס Y
                prevYFilt = 120;               // איפוס Y
                TFT_Fill_Screen(CL_YELLOW);   // ניקוי רקם
            }
        }
    }
}

// ----- צייר האות הגולמי (אילומ) -----
130
131
132
133
134
135
136
137

```

תוצאות על נבי מסר ה-TFT :



מימוש הקוד : TINKERCAD

```
1 // Sampling parameters                                     קצב הדגימה (500 דגימות לשנייה)
2 const int SAMPLE_RATE = 500;                            משך הזמן בין שתי דגימות (בשניות)
3 const float TIME_STEP = 1.0 / SAMPLE_RATE;             משך הזמן בין דגימות במייקרודשניות
4 const unsigned long SAMPLE_INTERVAL_US = 1000000 / SAMPLE_RATE; // הזמן האחרון שבו בוצעה דגימה
5 unsigned long lastMicros = 0;                           הזמן האחרון שvu בוצעה דגימה

6
7 // Signal parameters                                     תדר של גל שנ מסור (1 הרץ)
8 const float SAW_FREQ = 1.0;                            זהה לתחדש של רעש (סינוס) - 50
9 const float NOISE_FREQ = 50.0;                         אמפליטודה של גל שנ מסור
10 const float SAW_AMPLITUDE = 50.0;                      אמפליטודה של הרעש (גל הסינוס)
11 const float NOISE_AMPLITUDE = 10.0;                     אמפליטודה של הרעש (גל הסינוס)

12
13 float currentTime = 0;                                מעתנה שמייצג את הזמן הנוכחי (מצטבר)

14
15 // Notch filter coefficients for 50Hz                  של פילטר b0 מקדם
16 const float coef0 = 0.9710;                            של פילטר b1 מקדם
17 const float coef1 = -1.5716;                           של פילטר b2 מקדם
18 const float coef2 = 0.9710;                            של פילטר a1 מקדם
19 const float coefA1 = -1.5716;                          של פילטר a2 מקדם
20 const float coefA2 = 0.9420;                           גודל הבאför המוגלי (צריך 3 דגימות לפחות מסדר שני)

21
22 // Ring buffer                                         זיכרונו לדגימות קודמות של פלט ה필טר
23 #define HISTORY_LEN 3                                 גודל ההיסטוריה (3 דגימות לפחות לפילטר מסדר שני)
24 float inputHistory[HISTORY_LEN] = {0};                זיכרונו לדגימות קודמות של פלט הפלט
25 float outputHistory[HISTORY_LEN] = {0};               מוגלי Ring Buffer
26 int ringIndex = 0;                                    מוגלי לכמה דגימות שעברו
27 int sampleIndex = 0;                                  מעתנה לבדיקת שלב החימום (עד שהבאför מלא)
28 int warmup = 0;                                      הפלט האחרון של ה필טר
29 float filteredOutput = 0;                            הפלט הנוכחי של הפלט

30
```

```
31 // על קלט אחד פונקציה שמריצה את פילטר זה // אם נדיין לא מילאנו את הבאför, אל מסנו
32 float runNotchFilter(float inputValue) {               פושט תח้อน את הקלט בבאför
33     if (warmup < HISTORY_LEN) {                      ותחזיר אפס (או פלט מסוון עדין)
34         inputHistory[warmup++] = inputValue;          אינדקס למייקום הנוכחי
35         return 0;                                     אינדקס לדגימה אחת אחורה
36     }                                                 אינדקס לשתי דגימות אחורה
37
38 // חישוב האינדקסים בבאför (באיו מוגלי)           מסדר שני (IIR) חישוב הפלט של הפילטר
39 int idx0 = (ringIndex + 0) % HISTORY_LEN;           filteredOutput = coef0 * inputHistory[idx2] + // b0 * x[n]
40 int idx1 = (ringIndex + 1) % HISTORY_LEN;           coef1 * inputHistory[idx1] + // b1 * x[n-1]
41 int idx2 = (ringIndex + 2) % HISTORY_LEN;           coef2 * inputHistory[idx0] - // b2 * x[n-2]
42
43 //                                                      coefA1 * outputHistory[idx2] - // a1 * y[n-1]
44 //                                                      coefA2 * outputHistory[idx1]; // a2 * y[n-2]
45
46
47
48
49
50     inputHistory[ringIndex] = inputValue;            עדכון הקלט בבאför
51     outputHistory[idx0] = filteredOutput;           טמירת הפלט החדש בבאför הפלטים
52     ringIndex = (ringIndex + 1) % HISTORY_LEN;       עדכון המייקום המוגלי הבא
53     sampleIndex++;                                ספירת דגימה נוספת
54
55     return filteredOutput;                         הגדלת הפלט מסוון
56 }
57
58 void setup() {
59     Serial.begin(1200);                            התחלת תקשורת טורית בקצב 1200 ביט לשנייה
60 }
61
```

```
62 void loop() {                                       קבלת הזמן הנוכחי במייקרודשניות
63     unsigned long now = micros();                  אם עבר ממספר זמן מאז הדגימה האחרונות
64     if (now - lastMicros >= SAMPLE_INTERVAL_US) { // עדכון זמן הדגימה האחרונות
65         lastMicros = now;
66
67         // 1Hz ואמפליטודה גל שנ מסור בתדר
68         float sawComponent = 2 * SAW_AMPLITUDE * (currentTime * SAW_FREQ - floor(0.5 + currentTime * SAW_FREQ));
69
70         // 50Hz ואמפליטודה NOISE_FREQ יצירת גל סינוס בתדר
71         float sineComponent = NOISE_AMPLITUDE * sin(2 * PI * NOISE_FREQ * currentTime);
72
73         // משלב שני הרכיבים: אוט שנ מסור עם רעש של 50Hz
74         float rawSignal = sawComponent + sineComponent;
75
76         // הרצת הפילטר על האות
77         float cleanedSignal = runNotchFilter(rawSignal);
78
79         // -.rawSignal,cleanedSignal
80         Serial.print(rawSignal);                    הדפסת האות לפני הסינון //
81         Serial.print(",");                        פיסיק בփרדה //
82         Serial.println(cleanedSignal);            הדפסת האות אחרי סינון //
83
84         currentTime += TIME_STEP;                 עדכון הזמן לדגימה הבא //
85     }
86 }
```

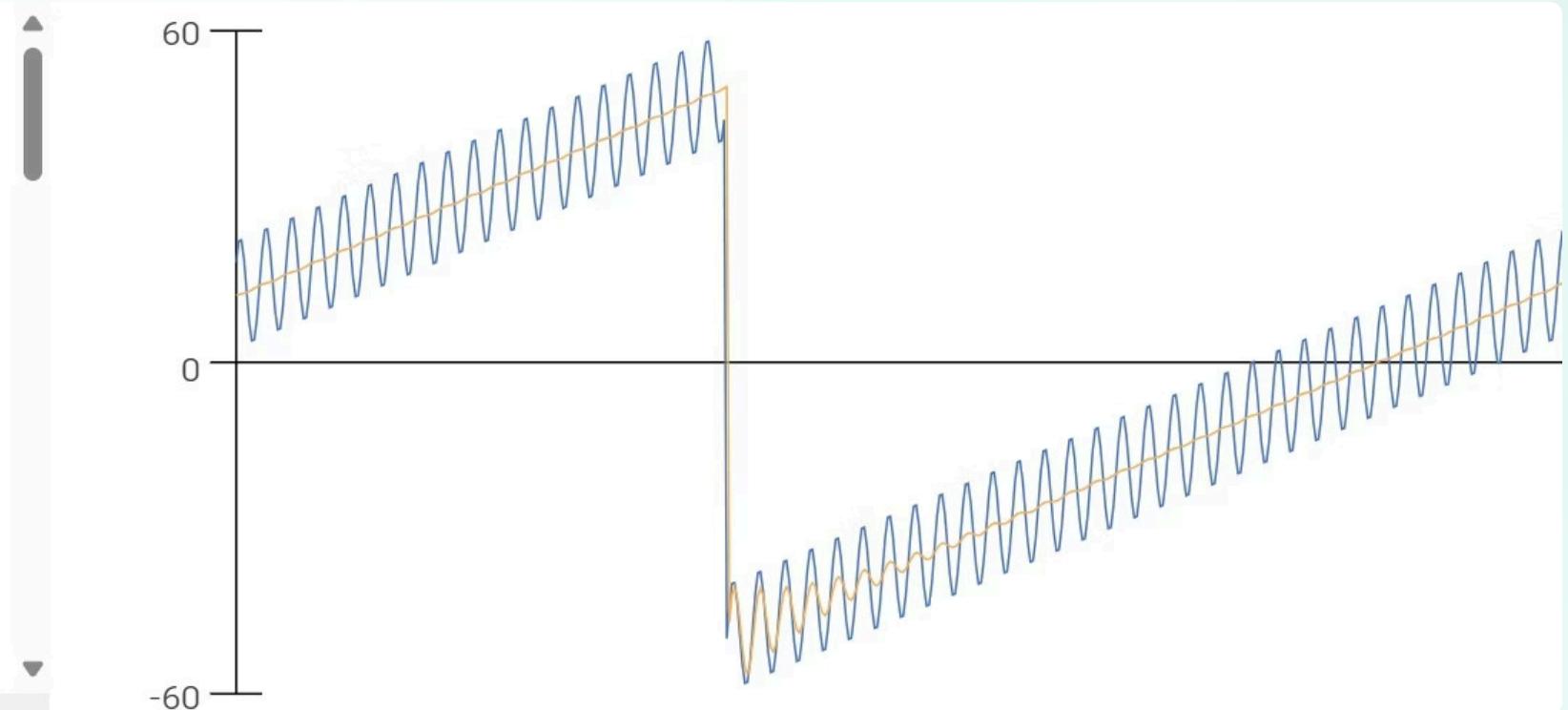
פתרונות הקוז ב-TINKERCAD

LINK TO TINKERCAD : <https://www.tinkercad.com/things/3kqYFaN4q0Z/editel?returnTo=%2Fdashboard>

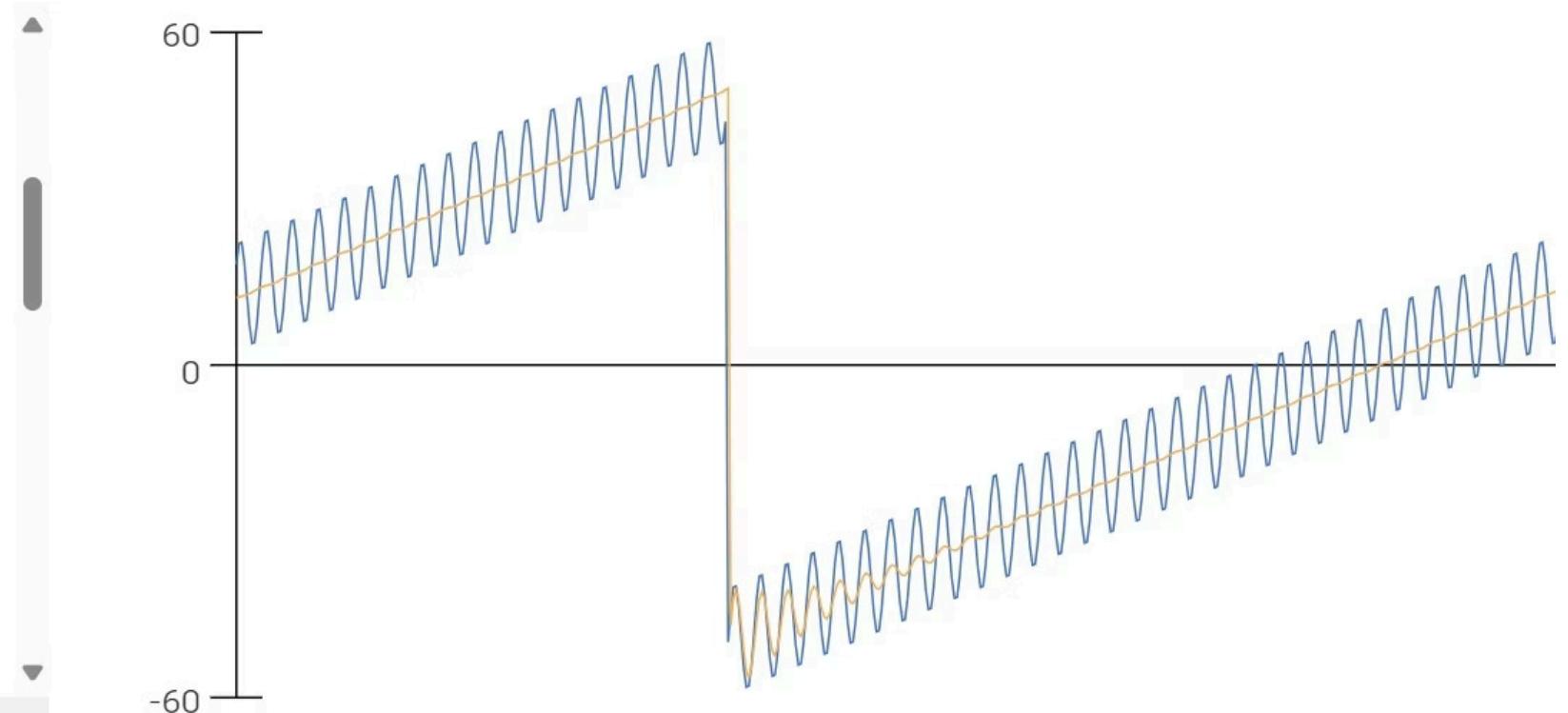
לו כחול - אות לא מסוכן

לו כתום - אות לאחר סינון 50 HZ

2.02,-2.12
4.06,-1.89
7.90,-1.71
8.12,-1.58
4.69,-1.47
-0.98,-1.34
-6.66,-1.17
-10.11,-0.95
-9.92,-0.68
-6.09,-0.40
-0.02,-0.12
6.06,0.11
9.90,0.29
10.12,0.42
6.69,0.53
1.02,0.66
-4.66,0.83
-8.11,1.05
-7.02,1.22



-7.92,1.32
-4.09,1.61
1.98,1.88
8.06,2.11
11.90,2.29
12.12,2.42
8.69,2.53
3.02,2.66
-2.66,2.83
-6.11,3.05
-5.92,3.32
-2.09,3.61
3.98,3.88
10.06,4.11
13.90,4.29
14.12,4.42
10.69,4.53
5.02,4.66
-0.66,4.83



מגבלות

עיותאות

יכול לעוות את האות אם רוחב הפס רחב מדי.

עיות פאה משפיע על 5-10% מהאותות הסמכים.

רכיבים מדויקים

מסננים אנלוגיים דורשים רכיבים מדויקים ורגניים לטמפרטורה.

יצירת פאה באותות ליד תדר המגראט מהווה אתגר.



סיכון

מסנני מגענות 60/50 הרץ הם כלי חיוני ובעל ערך רב. הם מסייעים בהסרת רעשி קוו מתח במגנון רחב של יישומים. התקדמות עתידית צפוייה להביא מסננים חכמים ואדפטיביים יותר.