המכללה האקדמית להנדסה בכרמיאל
אורט בראודה

תכנות מעבדי DSP 31561

P10- Heartbeat Detection

מגישות:

בלום אלונה 316406198

ליש לטם 313359556

בהנחיית:

מר קרוין יצחק

הוגש בתאריך: 29.01.2023

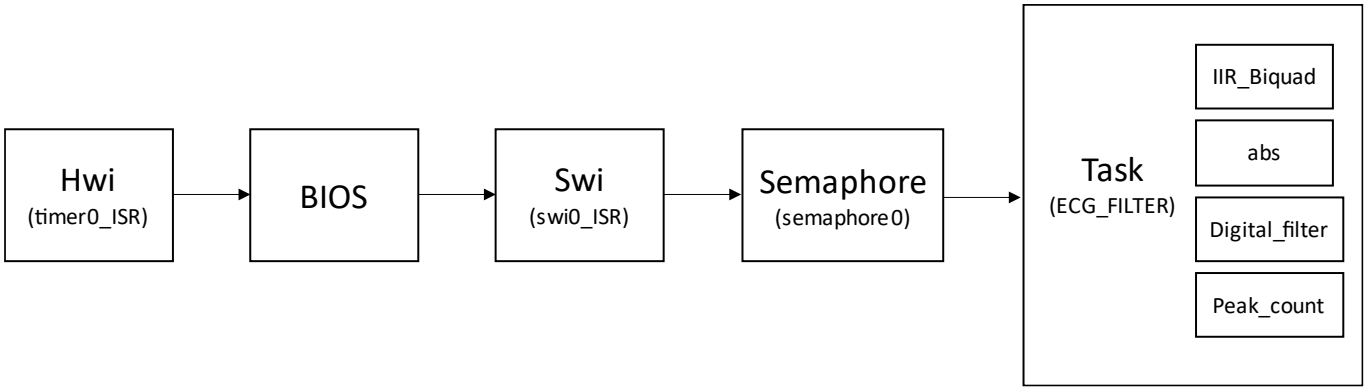
1 תוכן עניינים

2 דרישות הפרויקט

- ✓ השמת אות ECG נתון במערך בגודל 4012 ודגימתו בתדר דגימה של 2kHz
- ✓ שימוש ב- Timer אשר דוגם את אות ה-ECG בתדר הדגימה שנקבע ובאופן רציף וציקלי
- ✓ שימוש בתוכנת MATLAB לתכנון המסנן המתאים לסינון התדרים הלא רצויים ושיפור הזיהוי של הדופק. בהמלצת המרצה נבחר מסנן מסוג IIR
- ✓ זיהוי מרווח הזמן בין הדפקים הראשיים (בין S1 ל- S1) ובין דופק ראשי לדופק משני (בין S1 ל- S2) וחישוב קצב פעימות הלב על סמך מרווח הזמן בין הדפקים הראשיים
- ✓ הצגה על מסך ה- console את קצב פעימות הלב בפורמט bps. בנוסף, בחרנו להציג למסך גם את קצב הפעימות בפורמט bpm, את מרווחי הזמן בין הדפקים והאינדקסים שלהם במערך הדגימות.
- ✓ הצגת הגרפים של אות ה- ECG בשלבים השונים של העיבוד (האות המקורי, לאחר מעבר במסנן IIR, לאחר מעבר במיישר גל שלם ולאחר מעבר בגלאי מעטפת) במישור הזמן ובמישור התדר
- ✓ הצגת Execution graph, CPU load, Task load

3 דיאגרמת בלוקים של התוכנה

דיאגרמת הבלוקים בדגש על ארכיטקטורת multi-threading בזמן אמת:



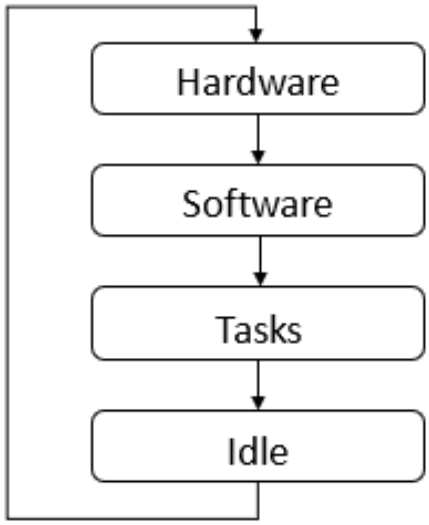
איור מס. 1 דיאגרמת הבלוקים של התוכנה

4 תרשים זרימה של המערכת

ה- Priority של ה- threads במערכת RTOS הינה בסדר הבא:

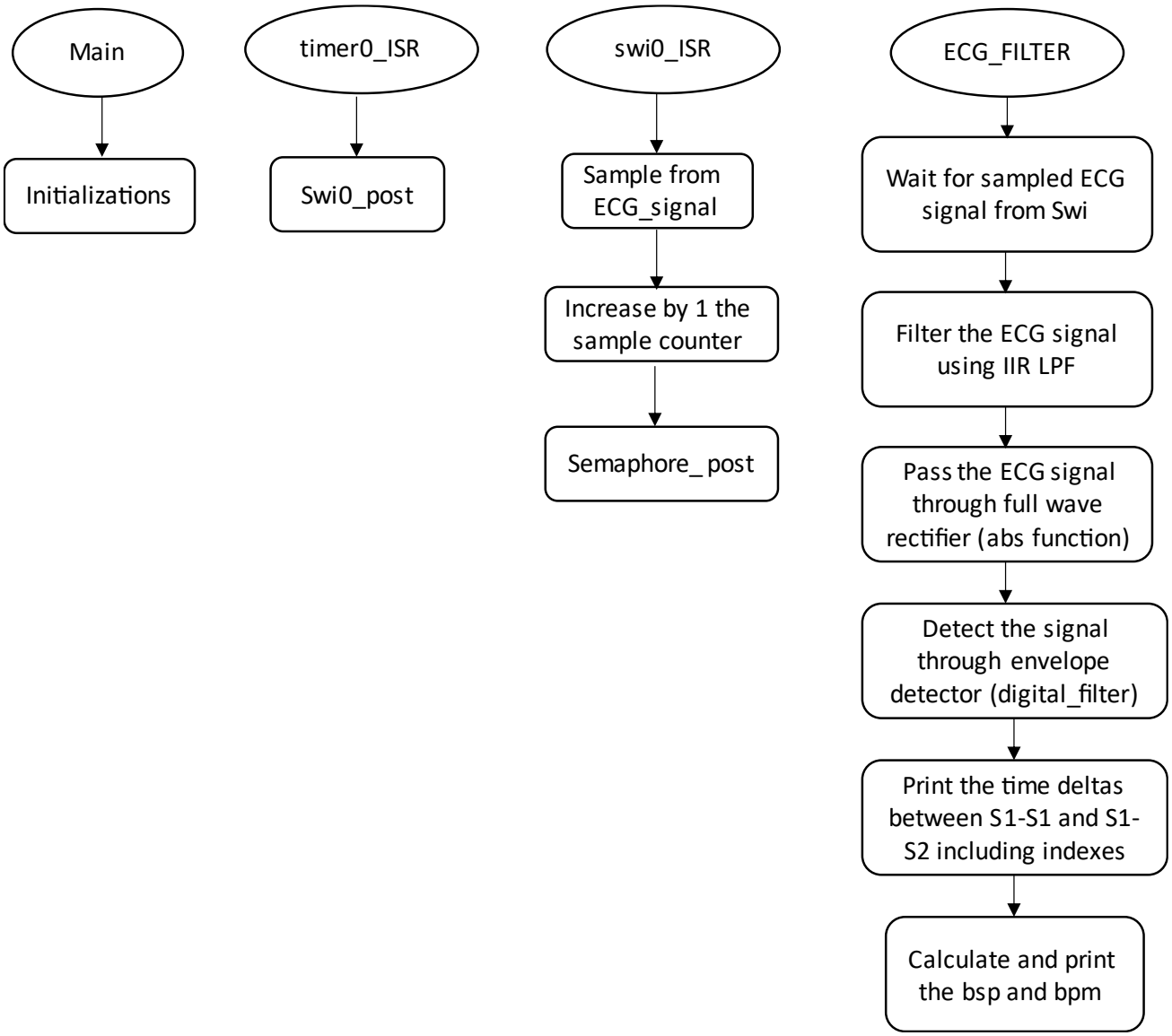
איור מס. 2 ה- priority של ה-threads

וסדר ביצוע הפעולות הינו:



איור מס. 3 סדר ביצוע ה-threads בכל הרצה

תרשים הזרימה של התוכנית:



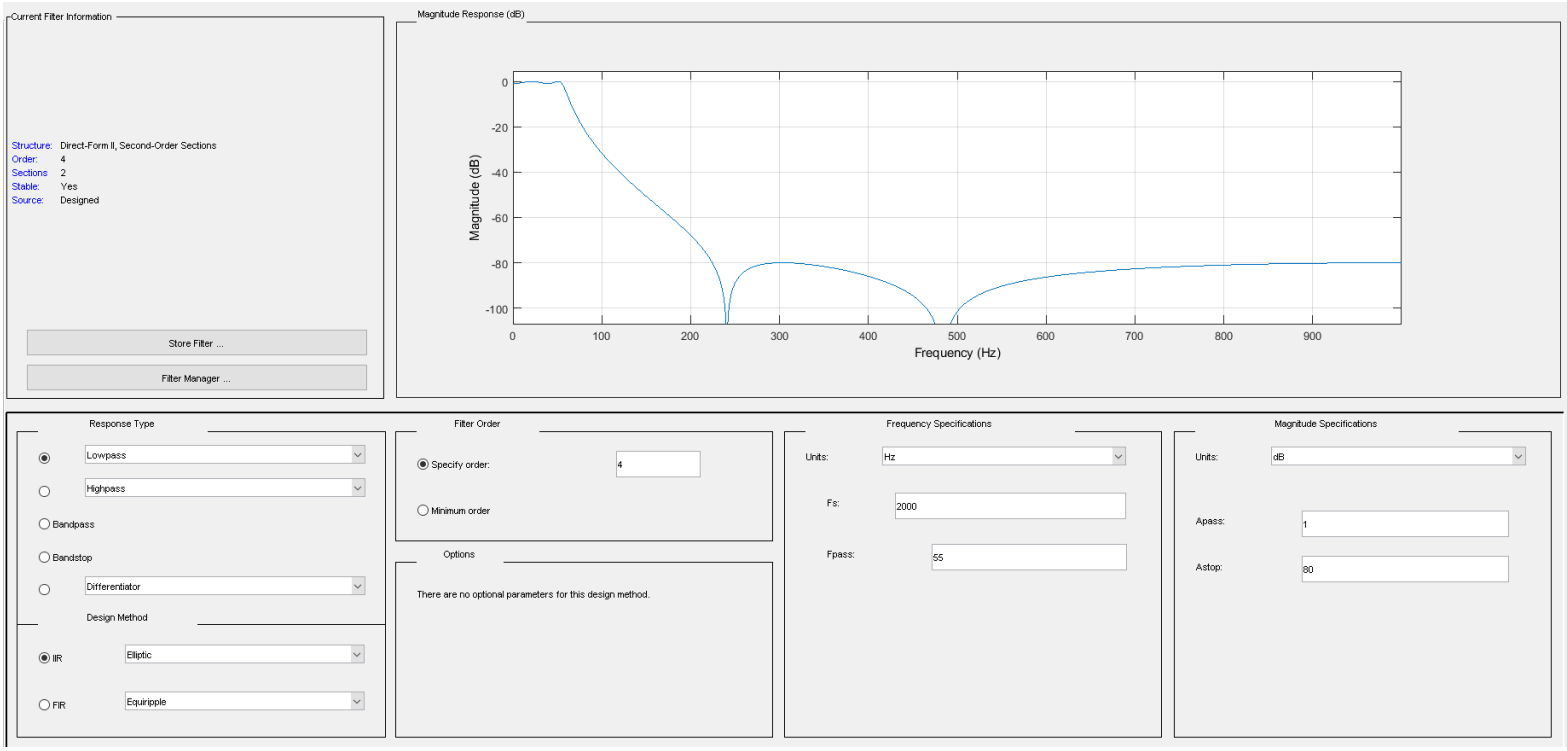
איור מס. 4 תרשים הזרימה של התוכנית

5 שיקולי תכנון

מטרת הפרויקט היא לחשב את קצב פעימות הלב של אדם ולהציג אותו בפורמט bps בהינתן אות ה ECG -שלו. ראשית, ביצענו את תכנון המסנן באמצעות אפליקציית Filter Designer ב- MATLAB. בחרנו במסנן מסוג IIR אליפטי ואת סדר המסנן בחרנו להיות 4 לסינון תדרים נמוכים, מתוך השיקולים שיוסברו בהמשך. עבור מימוש התוכנית, נעזרנו בתוכנית מתוך מעבדה מספר 6 בקורס (IIR Filter) וביצענו בתוכנית זו שינויים, למשל שינוי זמן המחזור של כל דגימה ב-timer0 ושינוי שעון ה-CPU ל-300MHz כפי שהיה בתבנית של פרק ג. בנוסף, הוספנו גם פונקציות שרלוונטיות למטרת הפרויקט אשר יוסברו בהמשך. שאר המשאבים נותרו ללא שינוי.

5.1 שיקולי התכנון עבור המסנן

להלן תצוגת המסנן הנבחר כפי שמופיעה ב- Filter Designer ב-MATLAB:



איור מס. 5 תכנון המסנן ב- Filter Designer ב- MATLAB

תכנון המסנן בוצע במספר שלבים:

1. על מנת להבין אילו תדרים עלינו לסנן, בשלב הראשון הסתכלנו על אות ה-ECG במישור התדר ובחנו את התדרים הרצויים. באמצעות ניסוי וטעייה מצאנו שהמסנן אמור להיות מסוג LPF עם תדר העברה של 55Hz על מנת לזהות באופן מיטבי את התדרים המתאימים עבור S1 ו- S2 עם מינימום רעשים.
2. קבענו את סוג המסנן להיות IIR מכיוון שהוא יעיל יותר מבחינה חישובית. ניתן ליישם אותו באמצעות מספר קטן של מקדמים לעומת ה-FIR מה שהופך אותו לחסכוני מבחינת חומרה ומשאבים חישוביים.
3. בחרנו את מסנן ה- IIR להיות אליפטי מכמה סיבות:
   א. ראינו שהוא מעניק לנו Transition band חד יותר לעומת הסוגים האחרים.
   ב. ראינו שיש לו יכולת גבוהה של הנחתה ב- Stop band.
   ג. ראינו שעבורו אנחנו מקבלות סינון טוב עם מעט מקדמים.

4. בחרנו את סדר המסנן להיות 4 (כלומר עם שתי דרגות) על מנת לקבל רמת סינון מספקת. תחילה ביצענו במערכת השמה של מסנן עם דרגה אחת וראינו שהתוצאות אינן טובות. לאחר מכן ביצענו תכנון חוזר של המסנן אך הפעם עם שתי דרגות וראינו שתוצאות השתפרו מאוד.

להלן המפרט של המסנן:

Filter type: IIR

Response: Lowpass

Number of sections: 2

Filter order: 4

Design method: elliptic

Passband frequency: 55Hz

Cutoff frequency (-3dB point): 57.811Hz

Half power frequency (-6dB point): 61.05Hz

Stopband frequency: 224.34Hz

Transition band width: 169.34Hz

Passband ripple: 1dB

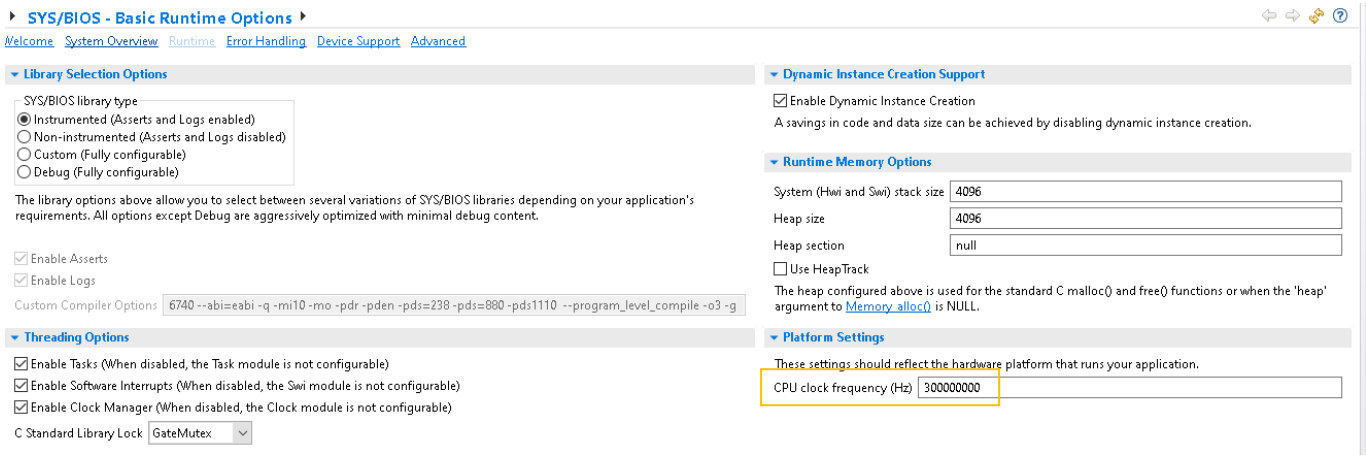
Attenuation Rate: $\frac{Gh[dB]-Gl[dB]}{\log_2(\frac{1}{224.34})} = \frac{80}{-7.81} = -10.24[dB/dec]$

5.2 שיקולי התכנון עבור התוכנית

5.2.1 תצורות

5.2.1.1 BIOS

יחידה זו משמשת לניהול threads בזמן אמת (תקשורת וסנכרון), תמיכה בציוד היקפי, טיפול בפסיקות וניהול זיכרון. ביחידה זו מגדירים את תדר שעון ה-CPU, שבחרנו להשאיר אותו ב-300MHz, כפי שהיה בתבנית שניתנה על ידי המרצה בתרגול של פרק ג.

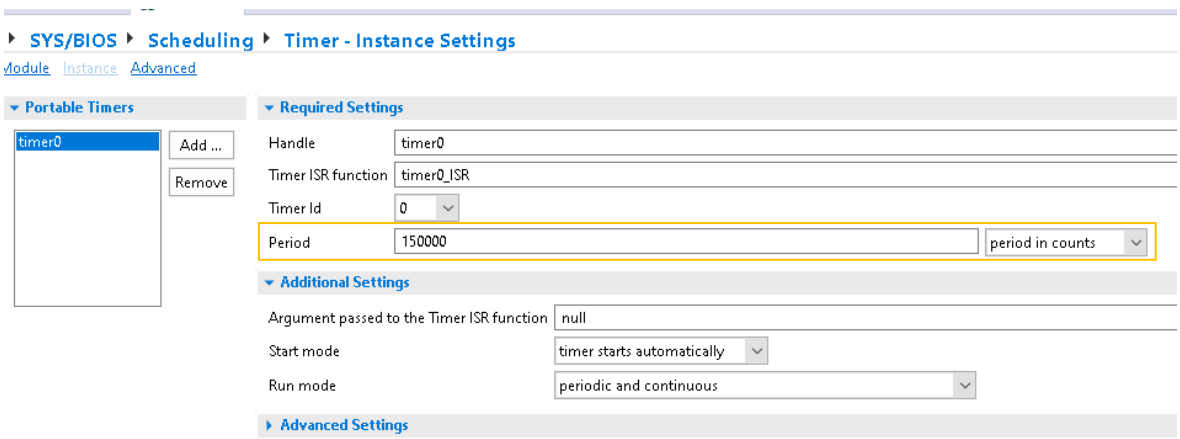


איור מס. 6 תצורת ה-BIOS

5.2.1.2 Timer

timer0 הינו מונה אשר ניתן להשתמש בו ליצירת פסיקות תקופתיות. המונה מופעל לפי תדר השעון של המערכת והינו בעל יכולת טעינה מחדש לאחר שהוא חוצה את הערך העליון שהוגדר לו. בפרויקט זה הוא משמש להפקת מרווחי זמן בין דגימה לדגימה, בהתאם לתדר דגימה של 2kHz (כלומר דגימה בכל 0.5msec). הגדרנו אותו לספור בפורמט של counts ולכן ערכו העליון הוא על פי הנוסחה:

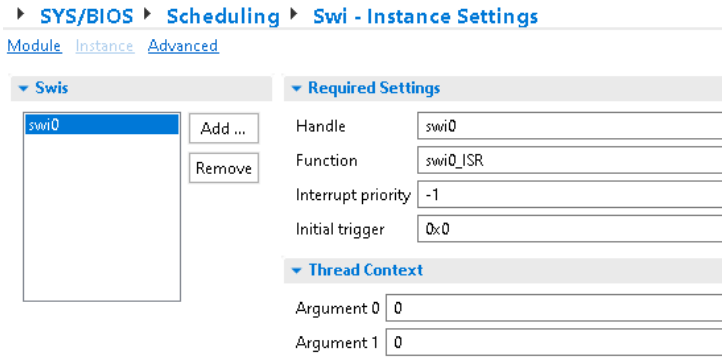
$$Timer0_{in\ counts} = \frac{CPU_{CLK}}{Fs} = \frac{300[MHz]}{2[kHz]} = 150{,}000\ counts$$



איור מס. 7 תצורת ה-Timer

5.2.1.3 Swi

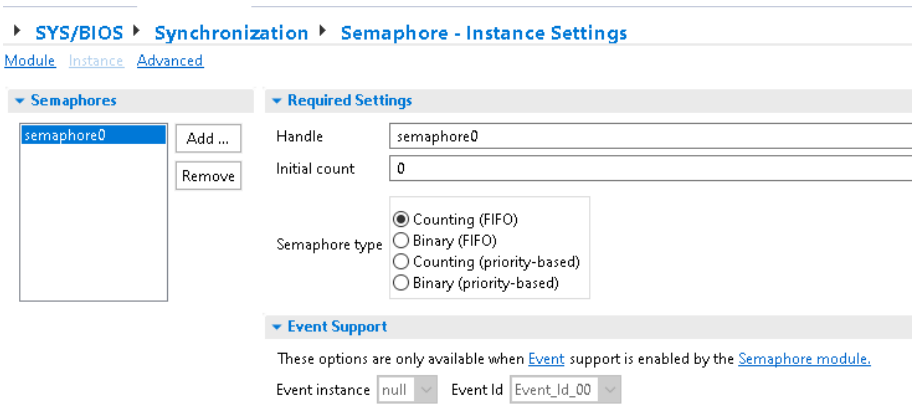
swi0 הינו פסיקת תוכנה עבור המעבד. משתמשים בו על מנת לעבור בין משימות, כך שמשימה בעלת עדיפות גבוהה יותר (מה שבתוך שגרת הפסיקה של swi0) מקדימה משימה בעלת עדיפות נמוכה יותר. במקרה שלנו, הקריאה ל- swi0 מתרחשת בתוך שגרת הפסיקה של timer0, כלומר בכל 0.5msec. בשגרת הפסיקה של swi0 אנו דוגמים ערך מתוך המערך ECG_signal לצורך עיבוד.



איור מס. 8 תצורת ה- Swi

5.2.1.4 Semaphore

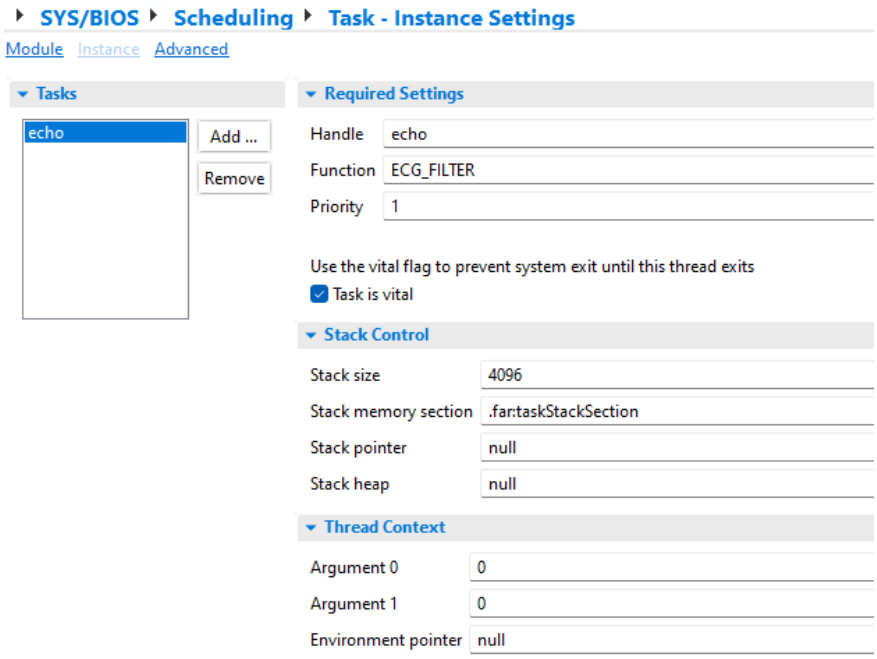
semaphore0 משמש לסנכרון תהליכים מרובים. הוא מאפשר ל- thread אחד לשלוט מתי threads אחרים יכולים לגשת למשאב משותף, תוך שהם ממשיכים בעבודתם בזמן ההמתנה. בכך הוא מבטיח שרק thread מסוים ישתמש במשאב משותף בכל רגע נתון. במקרה זה, ה- semaphore מתזמן את הקריאה ל- TASK: בסוף שגרת הפסיקה של swi0 ה- semaphore מאותת על משאב זמין (באמצעות הפקודה semaphore_post) בעוד שבפונקציה TASK ה-semaphore מאותת על המתנה למשאב (באמצעות הפקודה semaphore_pend).



איור מס. 9 תצורת ה- Semaphore

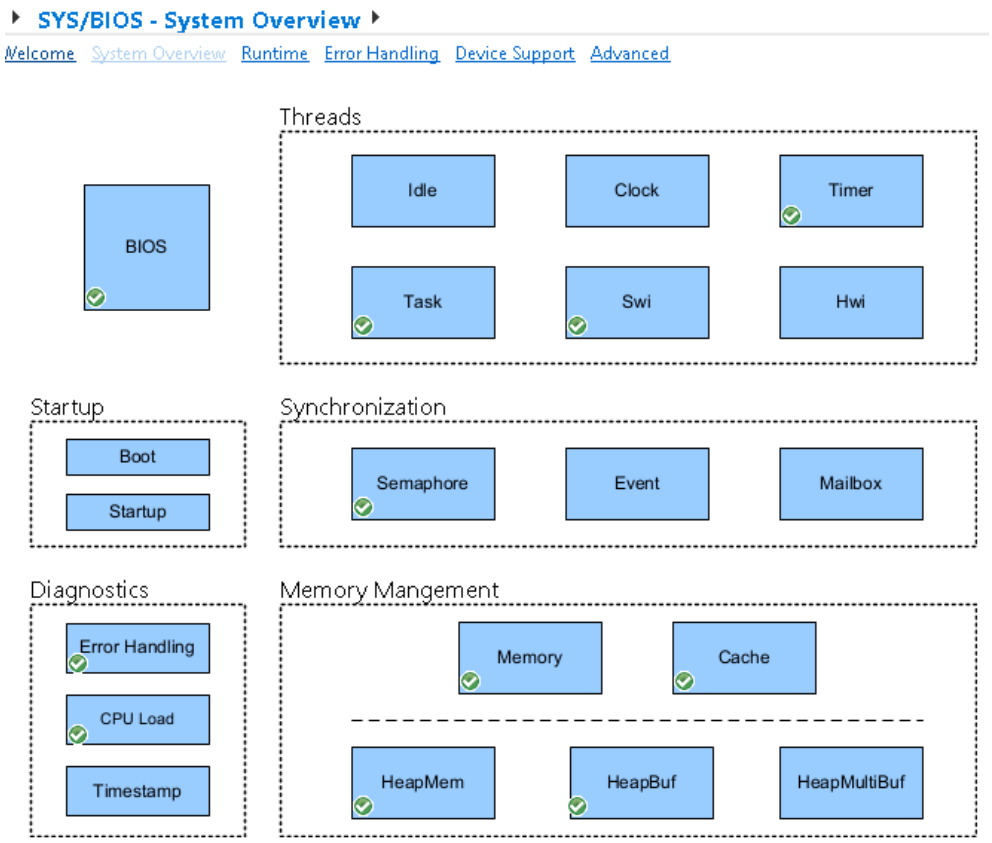
5.2.1.5 Task

זה יחידת העבודה שמבוצעת על ידי התוכנית. ניתן להשתמש בו כדי לארגן ולבצע פעולות בסדר מסוים או במרווחי זמן ספציפיים. במקרה שלנו הוא מעין הפונקציה הראשית בתוך לולאה אינסופית שקוראת לכל הפונקציות האחרות בתוכנית (מלבד הפסיקות). בין הפונקציות: IIR_Biquad, digital_filter, peak_counts.



איור מס. 10 תצורת ה- Task

System Overview 5.2.1.6



איור מס. 11 System Overview

5.2.2 פונקציות התוכנית

5.2.2.1 ECG-signal.h

זו קובץ header אשר מכיל מערך המייצג אות ECG לעיבוד. המערך נקרא ECG_signal בהתאם, והוא מכיל 4012 ערכים המייצגים את האותות החשמליים של הלב לאורך זמן. קובץ זה ניתן לנו מהמרצה ואין אנו מימשנו אותו, מלבד השמתו בתוכנית. בתוכנית הראשית אנו קוראים ממנו ערך בכל דגימה.

להלן חלק מקטע הקוד:

```
#ifndef _ECG_SIGNAL_H
#define _ECG_SIGNAL_H

/*
  SAMPLES:  4012
  SAMPLERATE: 2000
*/
#define N 4012       //size array


float ECG_signal[N] = {-2307.58,-1684.78,73.4506 ,5224.59 ,3470.11 ,4951.74 ,4501.13 ,4811.73 ,3878.71 ,4284.36 ,-719.158,-371.91 ,-120.07 ,
                    -3623.17,-3923.36,-4115.15,-2300.49,-1590.68,-3776.94,-59.0414,-3384.14,-1140.64,-5774.39,-3964.53,-2615.89,-4892.96,
                    -3679.11,-4361.12,-7369.18,-6468.93,-4961.26,-9324.97,-5777.07,-4985.04,-6419.31,-8233.98,-3897.68,-5944.36,-5673.17,
                    555.797 ,1019.79 ,2650.2  ,4326    ,835.874 ,1782.48 ,245.028 ,5663.8  ,-579.575,3866.38 ,2150.56 ,-53.4769,1014.77 ,
                    127.761 ,-1360.48,1583.99 ,811.742 ,-662.335,3592.36 ,107.027 ,6502.59 ,6744.6  ,5495.97 ,2102.28 ,4776.59 ,4244.19 ,
                    1341.92 ,1687.16 ,-560.093,2226.79 ,-236.094,-2788.78,-877.103,653.325 ,-2583.59,2755.18 ,2447.73 ,-1923   ,2790.37 ,
                    2240.95 ,1732.91 ,-1702.79,3566.37 ,-991.141,237.729 ,1367.78 ,-2125.31,-4686.77,-405.844,-3290.72,-3610.71,-4914.76,
                    -5183.36,-3384.31,332.194 ,-5752.19,-2480.28,-5730.59,-3633.12,-537.017,-3011.51,-3948.55,-5078.59,-869.891,-3055.17,
                    1416.35 ,2257.34 ,4084.59 ,1165.29 ,738.05  ,591.442 ,-556.597,723.238 ,2477.3  ,4508.15 ,5896.99 ,2669.3  ,4566.55 ,
                    -56.1934,5557.98 ,6165.19 ,4575.94 ,1691.66 ,758.231 ,5369.97 ,2380.48 ,2118.07 ,5582.98 ,2653.5  ,4198.16 ,-1172.6 ,
                    1812.09 ,888.863 ,69.4418 ,3627.74 ,5496.19 ,4716.57 ,25.4324 ,4170.76 ,-219.583,-341.368,-1535.39,2491.93 ,3394.35 ,
                    527.276 ,-8.14026,-192.138,-1068.17,-2534.2 ,978.548 ,77.7074 ,1482.58 ,-2640.11,-5352.2 ,-1215.26,-4088.53,-6431.4 ,
                    -2052.44,-5784.95,-3577.88,-1356.41,-3779.49,-3348.51,-28.2329,1620.79 ,1706.36 ,2896.78 ,-1593.15,-2670.78,-1274.53,
                    -2928.79,-2539.97,185.711 ,599.105 ,-2175.38,-128.732,-43.5413,-421.551,1927.9  ,517.06  ,-1074.57,400.278 ,1991.34 ,
                    -1170.36,-368.751,773.309 ,3223.8  ,-2917.57,-3601.39,1512.37 ,-1776.33,-3259.56,-4433.6 ,-1327.31,-1287.29,-1312.9 ,
                    -2777.33,-4623.98,-541.656,-1948.47,-4381.94,-2646.72,-3486.73,294.377 ,-235.877,2496.78 ,-1911.59,2264.7  ,4535.57 ,
```

איור מס. 12 פונקצית הספריה ECG_signal

5.2.2.2 fdacoefs.h

זו קובץ header בו מגדירים מערכים דו מימדיים המכילים את המקדמים של המסנן IIR שתיכננו באמצעות תוכנת MATLAB. אלו הם הערכים המספריים המגדירים את המאפיינים של המסנן בהתאם לפרמטרים שהכנסנו בממשק ה-Filter Designer ב- MATLAB. שיקולי התכנון של המסנן מוסברים בהמשך.

להלן קטע הקוד:

```
/* General type conversion for MATLAB generated C-code  */
#include "tmwtypes.h"
/*
 * Expected path to tmwtypes.h
 * C:\Program Files\MATLAB\R2019a\extern\include\tmwtypes.h
 */
/*
 * Warning - Filter coefficients were truncated to fit specified data type.
 *   The resulting response may not match generated theoretical response.
 *   Use the Filter Design & Analysis Tool to design accurate
 *   single-precision filter coefficients.
 */
#define MWSPT_NSEC 5
const int NL[MWSPT_NSEC] = { 1,3,1,2,1 };
const real32_T NUM[MWSPT_NSEC][3] = {
  {    0.07501506805,              0,              0    },
  {                1,  -0.1038354784,              1     },
  {     0.002682692604,              0,              0     },
  {                1,   -1.454896331,              1   },
  {                1,              0,              0    }
};
const int DL[MWSPT_NSEC] = { 1,3,1,2,1 };
const real32_T DEN[MWSPT_NSEC][3] = {
  {                1,              0,              0   },
  {                1,   -1.880930066,   0.8890467286   },
  {                1,              0,              0   },
  {                1,   -1.925604343,   0.9543580413 },
  {                1,              0,              0   }
};
```

איור מס. 13 פונקצית הספריה fdacoefs

5.2.2.3 timer0_ISR ()

להלן קטע הקוד:

```
// Sampling ISR according to fs frequency (2kHz)
void timer0_ISR(void)
{
    Swi_post(swi0);     //SW interrupt post
}
```

איור מס. 14 שגרת הפסיקה של ה- Timer

5.2.2.4 swi0_ISR ()

בשגרת פסיקה זו מתבצעת דגימת ערך מהמערך ECG_signal בכל 0.5msec ושמירת הערך בתוך משתנה בשם sample. הגדרנו משתנה נוסף בשם sample_cnt אשר סופר כלפי מעלה בכל דגימה. בהמשך התוכנית ניתן לראות ש-sample_cnt מתאפס לאחר 2000 דגימות המייצגות שניה אחת ולאחר מכך חוזר לספור מחדש. בסוף הפסיקה מתבצעת קריאה ל-semaphore לאיתות על משאבים זמינים.

להלן קטע הקוד:

```
// Getting the ECG_signal sample
void swi0_ISR(void)
{
    // samples ECG signal
    sample = ECG_signal[idx];
    sample_cnt ++;

    Semaphore_post(semaphore0); //post semaphore
}
```

איור מס. 15 שגרת הפסיקה של Swi

5.2.2.5 ECG_FILTER

פונקציה זו היא ה-Task ומהווה את הפונקציה הראשית של התוכנית. היא קוראת לפונקציה IIR_Biquard על מנת לבצע סינון ראשוני של האות הנדגם, לאחר מכן עושה פעולת ערך מוחלט על האות שיצא מהמסנן (ערך מוחלט הינו תיאור למיישר גל שלם) ושומרת את הערך במערך abs_arr. בשלב הבא היא קוראת לפונקציה digital_filter שהיא גלאי מעטפת דיגיטלי. לאחר שהאות עובר את השלבים הנ"ל, היא קוראת לפונקציה peak_count אשר סופרת את מספר הפיקים (בתוך הפונקציה peak_count ישנה קריאה לפונקציה אשר מחשבת את הפרשי הזמן בין הפיקים ואת קצב פעימות הלב). בסוף התוכנית האינדקס של מערך הדגימות idx עולה ב-1 עד שהוא מגיע לסוף המערך (N דגימות) ומתאפס. יחד עם האיפוס של idx מתאפסים גם המשתנים השומרים את אינקדסי הפיקים ואת מונה הפיקים.

```
//Task
void ECG_FILTER(UArg arg0, UArg arg1)
{
    while(1)
    {
        //Pending semaphore
        Semaphore_pend(semaphore0, BIOS_WAIT_FOREVER);  //Wait for sample
        IIR_Biquad();                                   //Filtering the sample
        filtered_arr[idx] = abs( filtered_arr[idx]);      //Absolute array's values
        abs_arr[idx] =  filtered_arr[idx];

        // Envelope detector
        if (idx > 0)
            digital_filter();

        peak_count();

        if(idx == N)  // Zeroing indexes when the array is full
        {
            idx = 0;
            peak_idx = 0;
            prev_peak_idx = 0;
            second_prev_peak_idx = 0;
            peak_cnt = 0;
        }


        else
            idx++;
    }
}
```

איור מס. 16 פונקציית ECG_FILTER (Task)

5.2.2.6 IIR_Biquad()

פונקציה זו הינה מסנן IIR דיגיטלי מסוג LPF לסינון תדרים לא רצויים באות ה-ECG. פונקציית המסנן מותאמת למסנן שתוכנן ב- Filter Designer ב- MATLAB: היא מכילה שתי דרגות של משוב וגם משתמשת במקדמים שחושבו ב-MATLAB (מיובאים דרך קובץ ה-fdacoefs.h header). היא מקבלת בתור קלט את הערך הנדגם מהמערך ECG_signal ומוציאה בתום שתי הדרגות את הערך לאחר סינון. כאשר index מתאפס, גם ערכי הפונקציה של המסנן מתאפסים. הבסיס לקוד נלקח מתוך מצגת ההרצאה על IIR Filter. <u>הוספנו מערך IIR_filtered_arr על מנת לראות את האות היוצא מהמסנן.</u>

להלן קטע הקוד:

```
//IIR biquad filter-  Eliptic IIR with 5 coeffs
void IIR_Biquad()
{
    // Clean IIR every time the buffer is fulfilled
    if (idx == 0 )
    {
        d00 = 0;
        d01 = 0;
        d02 = 0;
        d00_1 = 0;
        d01_1 = 0;
        d02_1 = 0;
    }

    // Read the input sample from the signal buffer
    xn = sample;
    y0 = 0;
    prod1 = d02*DEN[1][2];
    prod2 = d01*DEN[1][1];
    d00 = xn - prod1 - prod2;
    prod3 = d01*NUM[1][1];
    prod4 = d02*NUM[1][2];
    prod5 = d00*NUM[1][0];
    y0 = prod3+prod4+prod5;
    d02 = d01;
    d01 = d00;

    // Read the input sample after the first stage
    xn_1 = y0*NUM[0][0];
    y0_1 = 0;
    prod1_1 = d02_1*DEN[3][2];
    prod2_1 = d01_1*DEN[3][1];
    d00 = xn_1 - prod1_1 - prod2_1;
    prod3_1 = d01_1*NUM[3][1];
    prod4_1 = d02_1*NUM[3][2];
    prod5_1 = d00_1*NUM[3][0];
    y0_1 = prod3_1+prod4_1+prod5_1;
    d02_1 = d01_1;
    d01_1 = d00;

    // The signal after the IIR filter
    IIR_filtered_arr[idx] = y0_1*NUM[2][0];
    filtered_arr[idx]= IIR_filtered_arr[idx];
}
```

איור מ 17 פונקצית IIR_Biquad

5.2.2.7 *digital_filter()*

פונקציה זו הינה גלאי מעטפת דיגיטלי. לאחר המעבר דרך מסנן ה-IIR וביצוע פעולת ערך מוחלט, האות הנדגם עובר דרך ה-digital filter. גלאי המעטפת הוא אלגוריתם המשמש לחילוץ המעטפת של אות מאופנן, שהיא מייצגת את המידע המקורי הכלול באות. בעולם האנלוגי, גלאי המעטפת מורכב מדיודה, נגד וקבל, כאשר הדיודה מאפשרת את הטעינה המהירה של הקבל והנגד מאפשר את ההתפרקות האיטית של הקבל. ביישום האלגוריתם בתוכנה, גלאי המעטפת פועל על ידי השוואת הערך המספרי הנוכחי עם הערך המספרי הקודם: אם הערך הנוכחי גדול מהערך הקודם אז הערך הנוכחי נשמר ואם הערך הנוכחי קטן מהערך הקודם, הערך הנוכחי אינו נשמר והוא יורד ב-delta מסוימת, כלומר יורד בשיפוע קבוע. ערך ה-delta נמצא בתוך משתנה הנקרא num שערכו 50. ערך זה נבחר להיות 50 מתוך איזון בין השיפוע בעליה לשיפוע בירידה של אות המידע.

להלן קטע הקוד:

```
/* Create constant negative slope using in envelope detector
(capacitor discharge through a resistor) */
void digital_filter()
{
    if(filtered_arr[idx] < filtered_arr[idx - 1])
        filtered_arr[idx] = filtered_arr[idx - 1] - num;
}
```

איור מס. 18 פונקצית digital_filter

5.2.2.8 *peak_count()*

פונקציה זו סופרת את כמות ה-peaks שהתגלו ושומרת את האינדקסים של ה-peak הנוכחי וה-peak הקודם. היא מגלה peak ברגע שהיא מקבלת ערך הגדול מערך סף מסוים (Threshold), שהוגדר להיות 6480. ערך זה נקבע על סמך בחינה בין הערכים שאנו רוצים לקבל (S1 ו- S2) לערכים שאנו רוצים לסנן. בנוסף, הפונקציה דואגת שה-peak יספר פעם אחת בלבד בכך שהיא בודקת האם הערך הנוכחי גדול מערך ה- Threshold.

להלן קטע הקוד:

```
// Count s1 and s2 peaks
void peak_count()
{
    // Detect rising and falling of every peak
    if (filtered_arr[idx] >= Threshold && flag == 0 )
    {
        peak_cnt++;                 // Increase the current index of the peak
        prev_peak_idx = peak_idx;   // Save the previous peak index
        peak_idx = idx;             // Update the current peak index
        flag = 1;                   // Make sure the peak is measured only once
    }
    else if (filtered_arr[idx] < Threshold)
        flag = 0;

    if (sample_cnt == 2000) //2000 counts in every second for fs = 2[kHz]
        sample_cnt = 0;     // After crossing one second, reset the counter

    time_peaks();           // Call the time measure function
}
```

איור מס. 19 פונקצית peak_count

5.2.2.9 *time_peaks()*

פונקציה זו סופרת את ההפרש בין הדפקים הראשיים (S1-S1) ובין דופק ראשי לדופק משני (S1-S2). ההפרש בין שני דפקים ראשיים נמדד כאשר ה-peak הנוכחי הוא אי-זוגי וישנם 3 peaks לפחות (כלומר כאשר ישנם שני דפקים ראשיים לפחות) ומרווח זמן בין דופק ראשי לדופק משני נמדד כאשר ה-peak הנוכחי הוא זוגי והוא 2 לפחות (כלומר יש דופק ראשי ודופק משני). כאשר מחשבים את הזמן בין S1-S2 שומרים את הערך של S1 (second_prev_peak) על מנת שניתן יהיה לחשב את המרווח בין S1 הנוכחי ל-S1 הקודם.

לאחר שמחשבים את הזמן בין S1-S1 (s1_delta) מחשבים את קצב פעימות הלב ב-bps בכך שמחלקים את הערך 2000 (כמות התאים במערך המגדיר שניה) ב-s1_delta (כמות התאים במערך המגדיר את מרווח הזמן בין ה-peaks). מנה זו מתארת כמה זמני מחזור של S1-S1 נכנסים בשניה אחת, כלומר את קצב פעימות הלב לשניה. בנוסף, חישבנו את קצב פעימות הלב לדקה באמצעות הכפלת ה-bps פי 60.

את קצב פעימות הלב, מרווחי הזמן ואינדקסי הזמן אנו מדפיסים למסך.

להלן קטע הקוד:

```
// Calculate the delta time between two peak (s1-s1 and s1-s2)
void time_peaks()
{
    prev_s1_delta  = s1_delta;
    prev_s12_delta = s12_delta;

    if (peak_cnt % 2 && peak_cnt > 2) // Calculate the delta time for primary peaks (S1-S1)
    {
        s1_delta = peak_idx - second_prev_peak_idx; //peak_idx minus peak_idx-2
        bps = (float)(2000.0/s1_delta);             // Calculate the heart rate in beat per second format
        if (s1_delta != prev_s1_delta)
        {
          System_printf ("previous s1 peak index = %d current s1 peak index = %d\n", second_prev_peak_idx, peak_idx);
          System_printf("s1_delta = %d [0.5msec]\n", s1_delta);
          System_printf ("bps = %f bpm = %f\n", bps, bps*60);
          System_flush();
        }
    }
    else if (((peak_cnt % 2) == 0) && peak_cnt > 1) // Calculate the delta time for secondary peaks (S1-S2)
    {
        second_prev_peak_idx = prev_peak_idx; // Save the peak_idx-2 index
        s12_delta = peak_idx - prev_peak_idx; //peak_idx minus peak_idx-1
        if (s12_delta != prev_s12_delta)
        {
          System_printf ("s1 peak index = %d s2 peak index = %d\n", second_prev_peak_idx, peak_idx);
          System_printf("s12_delta = %d [0.5msec]\n", s12_delta);
          System_flush();
        }
    }
}
```

איור מס. 20 פונקצית time_peaks

<u>על מנת להשתמש בפונקציה System_printf כולל האפשרות להדפיס מספרים שהם float, יש להוסיף את שורות הקוד הבאות בקובץ ECG.cfg:</u>

```
var System = xdc.useModule('xdc.runtime.System');
var SysStd = xdc.useModule('xdc.runtime.SysStd');
System.SupportProxy = SysStd;
System.extendedFormats = "%f";
```

איור מס. 21 שורות הקוד הנחוצות להדפסה עבור System_printf

6 תוצאות ההרצה

הגדרות עבור הצגת הגרפים במישור הזמן:

עבור האות המקורי נציג את המערך ECG_signal:

Property	Value
Data Properties	
Acquisition Buffer Size	4012
Dsp Data Type	32 bit floating point
Index Increment	1
Q_Value	0
Sampling Rate Hz	2000
Start Address	ECG_signal
Display Properties	
Auto Scale	☑ true
Axis Display	☑ true
Data Plot Style	Line
Display Data Size	4012
Grid Style	No Grid
Magnitude Display Scale	Linear
Time Display Unit	sample
Use Dc Value For Graph	☐ false

עבור האות שנראה אחרי המסנן IIR נציג את המערך IIR_filtered_arr:

Graph Properties

Property	Value
Data Properties	
Acquisition Buffer Size	4012
Dsp Data Type	32 bit floating point
Index Increment	1
Q_Value	0
Sampling Rate Hz	2000
Start Address	IIR_filtered_arr
Display Properties	
Auto Scale	☑ true
Axis Display	☑ true
Data Plot Style	Line
Display Data Size	4012
Grid Style	No Grid
Magnitude Display Scale	Linear
Time Display Unit	sample
Use Dc Value For Graph	☐ false

עבור האות שנראה אחרי המעבר במיישר הגל השלם (ערך מוחלט) נציג את המערך abs_arr:

Graph Properties

Property	Value
Data Properties	
Acquisition Buffer Size	4012
Dsp Data Type	32 bit floating point
Index Increment	1
Q_Value	0
Sampling Rate Hz	2000
Start Address	abs_arr
Display Properties	
Auto Scale	☑ true
Axis Display	☑ true
Data Plot Style	Line
Display Data Size	4012
Grid Style	No Grid
Magnitude Display Scale	Linear
Time Display Unit	sample
Use Dc Value For Graph	☐ false

עבור האות שנראה אחרי המעבר בגלאי המעטפת נציג את המערך filtered_arr:

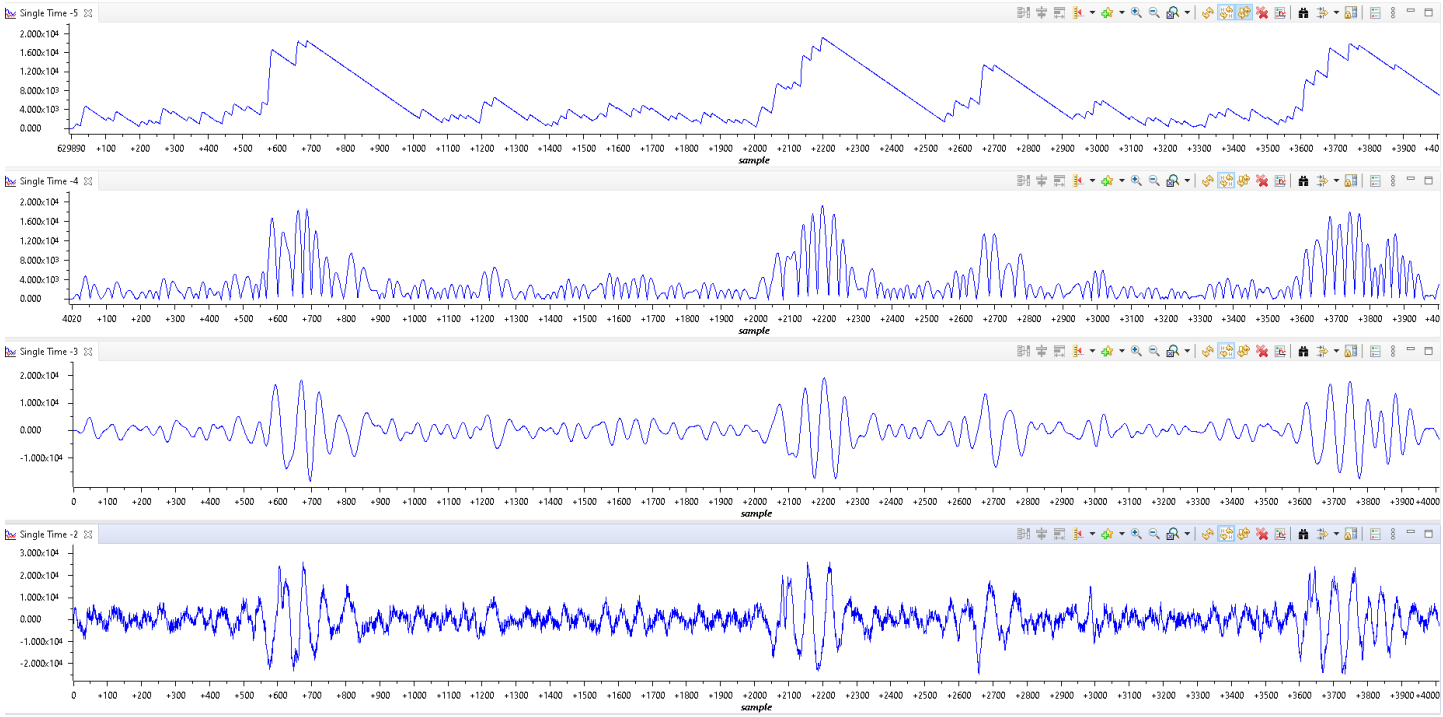
Graph Properties

Property	Value
Data Properties	
Acquisition Buffer Size	4012
Dsp Data Type	32 bit floating point
Index Increment	1
Q_Value	0
Sampling Rate Hz	2000
Start Address	filtered_arr
Display Properties	
Auto Scale	☑ true
Axis Display	☑ true
Data Plot Style	Line
Display Data Size	4012
Grid Style	No Grid
Magnitude Display Scale	Linear
Time Display Unit	sample
Use Dc Value For Graph	☐ false

איור מס.22 הגדרות עבור הצגת הגרפים במישור הזמן

האיור הבא מתאר את אות ה-ECG במישור הזמן בשלבי העיבוד השונים (מלמטה למעלה)

1. אות ה-ECG המקורי שניתן על ידי המרצה
2. אות ה-ECG לאחר מעבר במסנן IIR (הפונקציה IIR_Biquad)
3. אות ה-ECG משלב 2 לאחר מעבר במיישר גל שלם (פונקציית ערך מוחלט)
4. אות ה-ECG משלב 3 לאחר מעבר בגלאי מעטפת דיגיטלי (הפונקציה digital_filter)



איור מס. 23 אות ה-ECG בשלבי העיבוד השונים במישור הזמן

הגדרות עבור הצגת הגרפים במישור התדר:

עבור האות המקורי נציג את המערך ECG_signal:

Property	Value
Data Properties	
Acquisition Buffer Size	4012
Dsp Data Type	32 bit floating point
Index Increment	1
Q_Value	0
Sampling Rate Hz	2000
Signal Type	Real
Start Address	ECG_signal
Display Properties	
Auto Scale	☑ true
Axis Display	☑ true
Data Plot Style	Line
Frequency Display Unit	Hz
Grid Style	No Grid
Magnitude Display Scale	Linear
FFT	
FFT Frame Size	2048
FFT Order	11
FFT Window Function	Hamming

עבור האות שנראה אחרי המסנן IIR נציג את המערך IIR_filtered_arr:

Graph Properties

Property	Value
Data Properties	
Acquisition Buffer Size	4012
Dsp Data Type	32 bit floating point
Index Increment	1
Q_Value	0
Sampling Rate Hz	2000
Signal Type	Real
Start Address	IIR_filtered_arr
Display Properties	
Auto Scale	☑ true
Axis Display	☑ true
Data Plot Style	Line
Frequency Display Unit	Hz
Grid Style	No Grid
Magnitude Display Scale	Linear
FFT	
FFT Frame Size	2048
FFT Order	11
FFT Window Function	Hamming

עבור האות שנראה אחרי המעבר במיישר הגל השלם (ערך מוחלט) נציג את המערך abs_arr:

Graph Properties

Property	Value
Data Properties	
Acquisition Buffer Size	4012
Dsp Data Type	32 bit floating point
Index Increment	1
Q_Value	0
Sampling Rate Hz	2000
Signal Type	Real
Start Address	abs_arr
Display Properties	
Auto Scale	☑ true
Axis Display	☑ true
Data Plot Style	Line
Frequency Display Unit	Hz
Grid Style	No Grid
Magnitude Display Scale	Linear
FFT	
FFT Frame Size	2048
FFT Order	11
FFT Window Function	Hamming

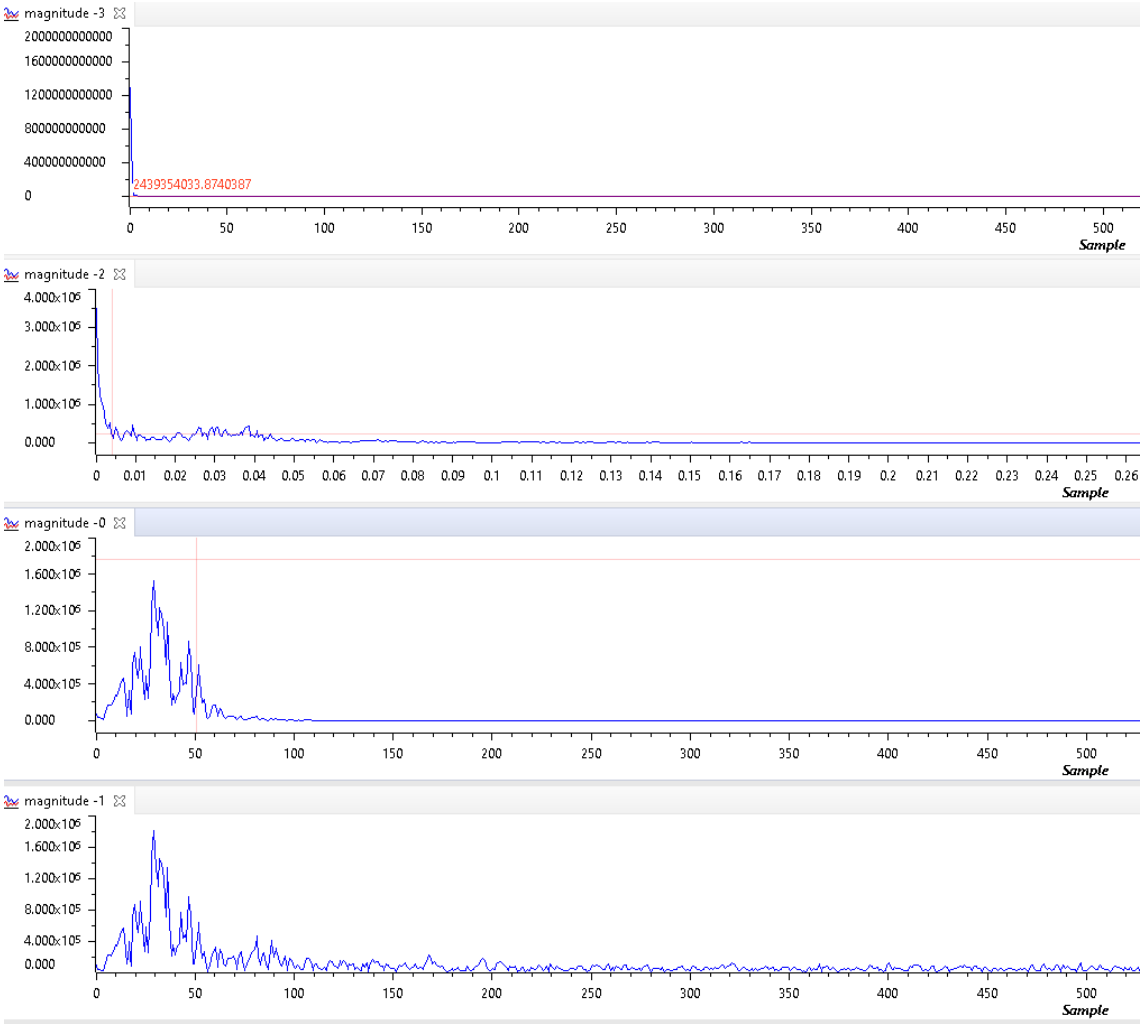
עבור האות שנראה אחרי המעבר בגלאי המעטפת נציג את המערך filtered_arr:

Graph Properties

Property	Value
Data Properties	
Acquisition Buffer Size	4012
Dsp Data Type	32 bit floating point
Index Increment	1
Q_Value	0
Sampling Rate Hz	2000
Signal Type	Real
Start Address	filtered_arr
Display Properties	
Auto Scale	☑ true
Axis Display	☑ true
Data Plot Style	Line
Frequency Display Unit	Hz
Grid Style	No Grid
Magnitude Display Scale	Linear
FFT	
FFT Frame Size	2048
FFT Order	11
FFT Window Function	Hamming

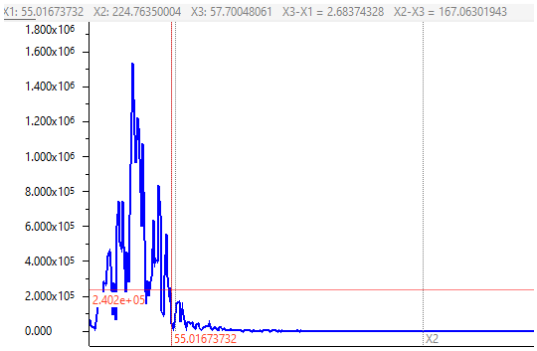
איור מס. 24 הגדרות עבור הצגת הגרפים במישור התדר

האיור הבא מתאר את אות ה-ECG במישור התדר בשלבי העיבוד השונים (מלמטה למעלה):



איור מס. 25 אות ה-ECG בשלבי העיבוד השונים במישור התדר

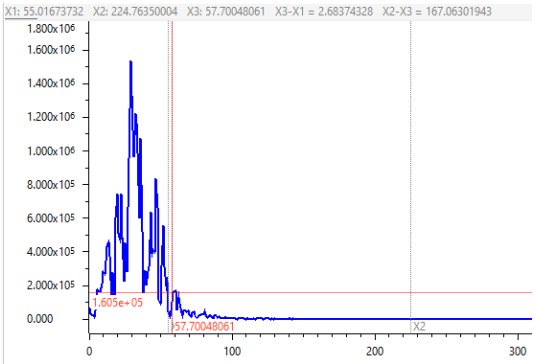
עבור ה- passband נקבל:



איור מס.26 passband בהשוואה ל-MATLAB

הערכים עבור ה-passband הם בקירוב: X=55.017Hz, Y=2.402E+5

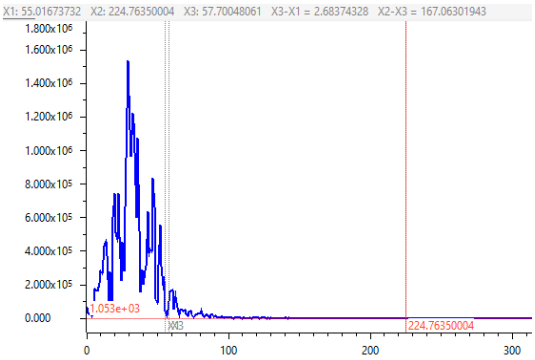
עבור ה- cutoff frequency נקבל:



איור מס. 27 cutoff frequency בהשוואה ל-MATLAB

הערכים עבור ה-cutoff frequency הם בקירוב: X=57.7Hz, Y=1.6E+5

עבור ה-stopband נקבל:



איור מס. 28 stopband בהשוואה ל-MATLAB

הערכים עבור ה-stopband הם בקירוב: X=224.764, Y=1.053E+3

נסכם את התוצאות ונקבל:

Passband frequency: 55.017Hz

Cutoff frequency (-3dB point): 57.7Hz

Stopband frequency: 224.764Hz

Transition width = 167.063Hz

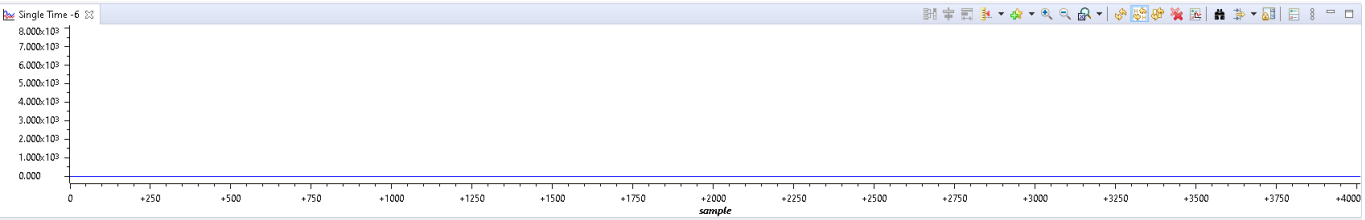
Attenuation Rate: $\frac{20\log\left(\frac{2.402E+5}{1.053e+3}\right)}{\log_2\left(\frac{1}{224.764}\right)} = \frac{47.163}{-7.812} = -6.037[dB/dec]$

ניתן לומר שהתוצאות עבור המסנן המעשי דומות לתוצאות עבור המסנן התיאורטי שתוכנן ב- MATLAB, מלבד שקצב ההנחתה הוא מעט קטן יותר משל המסנן בתיאוריה. יש לקחת בחשבון שהמדידה של ערכי המסנן המעשי נעשתה באופן ידני על סמך העין האנושית בעוד שערכי המסנן ב-MATLAB חושבו באופן אוטומטי על ידי ה- MATLAB. בנוסף, גרף ה- FFT

Magnitude ב- CCS הוא אינו סטטי אלא מעדכן במהלך ההרצה ויכול להיות שעבור זמן אחר היינו מקבלות תוצאות מעט שונות. באופן כללי ניתן לומר שהמסנן המעשי עמד בדרישות המצופות.

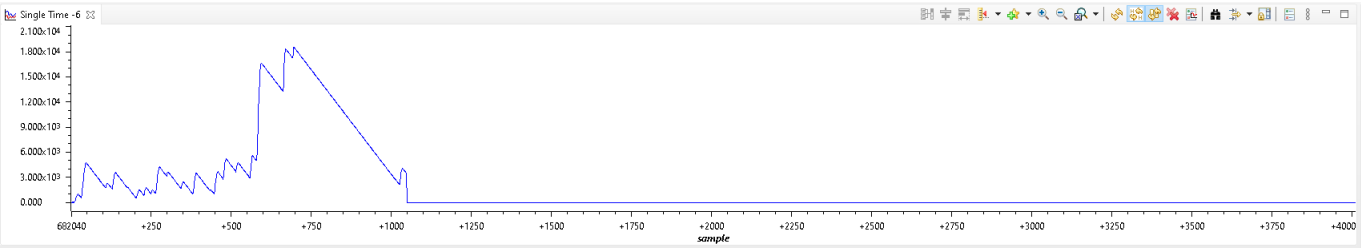
האיורים הבאים מתארים את שלבי זיהוי מרווחי הזמן בין S1-S1 ו-S1-S2 וזיהוי קצב פעימות הלב:

שלב 1- המערכת עוד לא זיהתה peaks



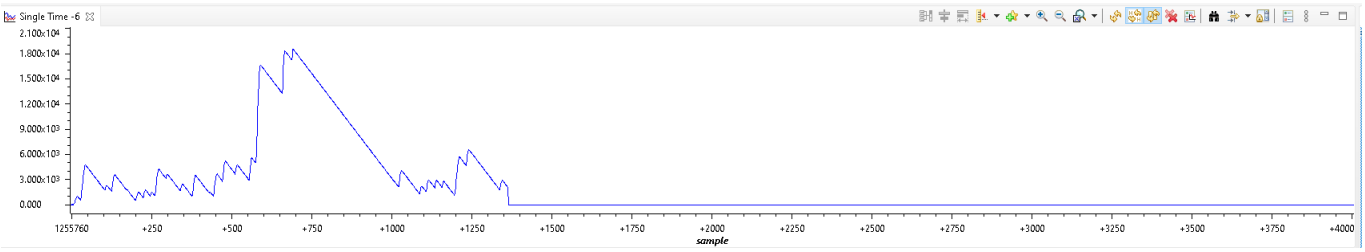
איור מס.29 שלב זיהוי 1- אין Peaks

שלב 2- המערכת זיהתה את S1 ושומרת את האינדקס שלו, אבל עדיין לא ביצעה חישובים עבורו.



איור מס. 30 שלב זיהוי 2- זוהה S1

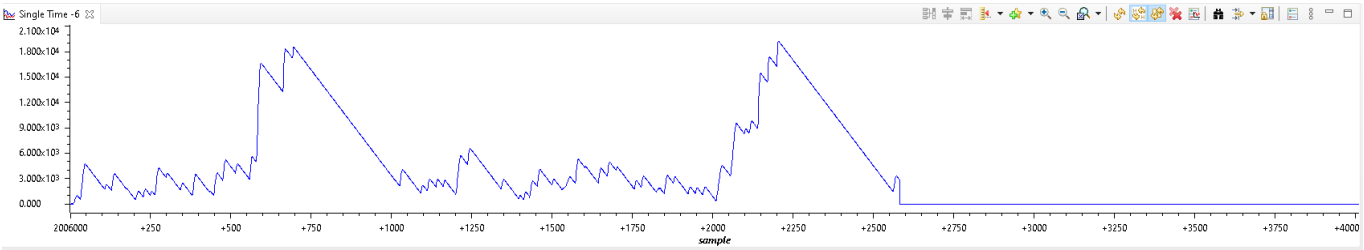
שלב 3- המערכת זיהתה את S1 ו- S2 וביצעה חישוב של מרווח הזמן ביניהם. ניתן לראות בפלט את האינדקסים של S1 ו-S2 ואת ההפרש ביניהם.




```
Console
ECG:CIO
s1 peak index = 582 s2 peak index = 1244
s12_delta = 662 [0.5msec]
```

איור מס. 31 שלב זיהוי 3- זוהה זוג S1-S2 וחושב מרווח הזמן ביניהם

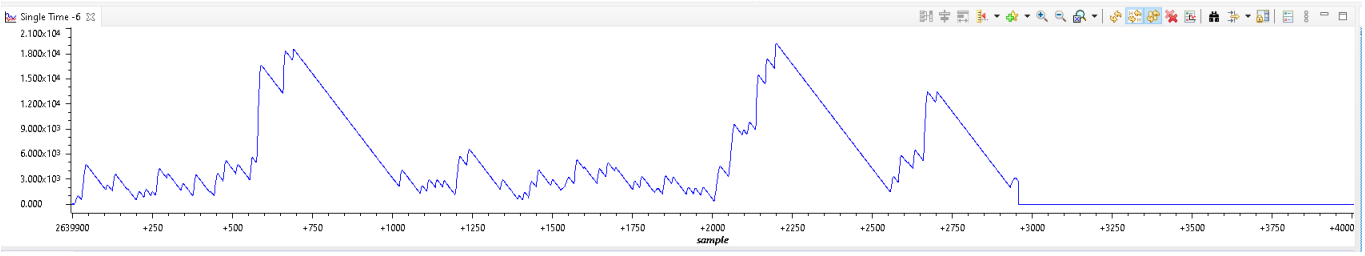
שלב 4- המערכת זיהתה שני S1, ביצעה חישוב של ההפרש ביניהם ואת קצב פעימות הלב (ב-bps ו- bpm). ניתן לראות בפלט את האינדקסים של שני ה- S1, את ההפרש ביניהם ואת קצב פעימות הלב ב-bps ו-bpm.




```
Console
ECG:CIO
s1 peak index = 582 s2 peak index = 1244
s12_delta = 662 [0.5msec]
previous s1 peak index = 582 current s1 peak index = 2064
s1_delta = 1482 [0.5msec]
bps = 1.3495 bpm = 80.9716
```

איור מס.32 שלב זיהוי 4- זוהה זוג S1-S1 וחושב מרווח הזמן ביניהם וקצב פעימות הלב

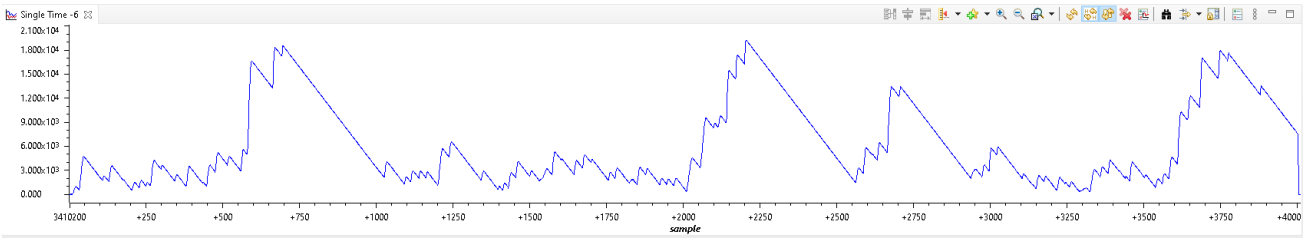
שלב 5- המערכת זיהתה זוג חדש של S1-S2 וביצעה חישוב של ההפרש ביניהם. החישוב עבור S1-S1 טרם התעדכן כי אין S1 חדש. ניתן לראות בפלט את האינדקסים של הזוג S1-S2 החדש ואת ההפרש ביניהם.




```
Console
ECG:CIO
s1 peak index = 582 s2 peak index = 1244
s12_delta = 662 [0.5msec]
previous s1 peak index = 582 current s1 peak index = 2064
s1_delta = 1482 [0.5msec]
bps = 1.3495 bpm = 80.9716
s1 peak index = 2064 s2 peak index = 2666
s12_delta = 602 [0.5msec]
```

איור מס. 33 זוהה זוג S1-S2 חדש וחושב מרווח הזמן ביניהם

שלב 6- המערכת זיהתה זוג חדש של S1-S1, ביצעה חישוב של ההפרש ביניהם ואת קצב פעימות הלב החדש. החישוב עבור S1-S2 לא התעדכן כי אין S2 חדש.



```
Console
ECG:CIO
s1 peak index = 582 s2 peak index = 1244
s12_delta = 662 [0.5msec]
previous s1 peak index = 582 current s1 peak index = 2064
s1_delta = 1482 [0.5msec]
bps = 1.3495 bpm = 80.9716
s1 peak index = 2064 s2 peak index = 2666
s12_delta = 602 [0.5msec]
previous s1 peak index = 2064 current s1 peak index = 3613
s1_delta = 1549 [0.5msec]
bps = 1.2911 bpm = 77.4693
```

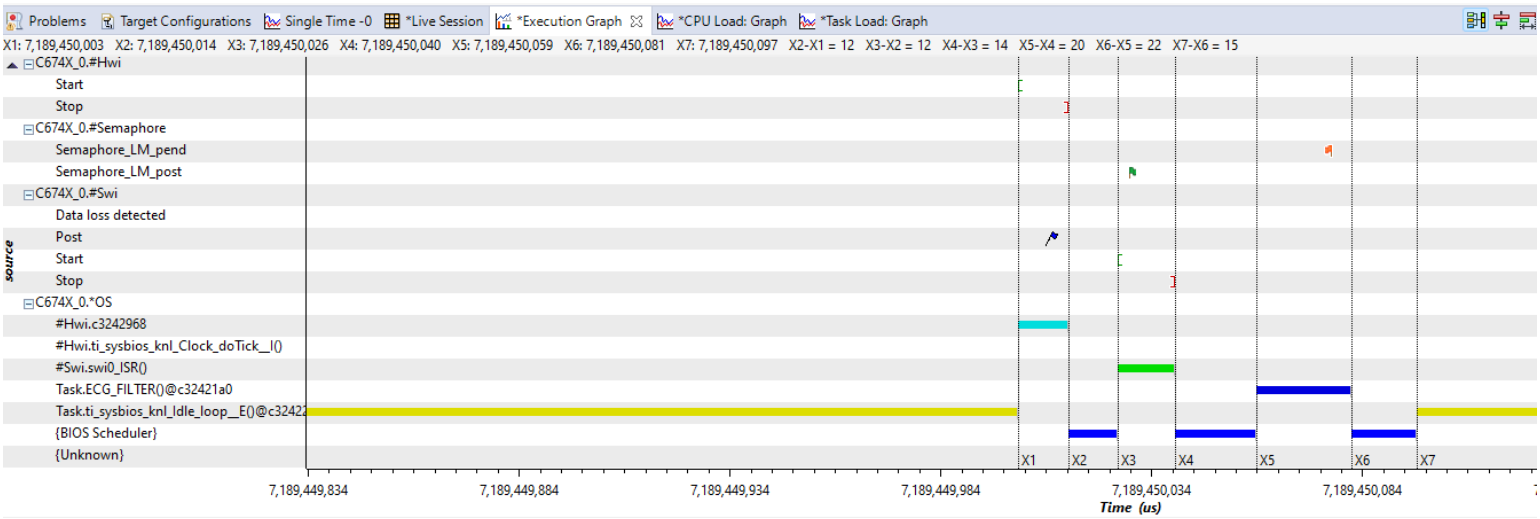
איור מס. 34 זוהה זוג S1-S1 חדש וחושב מרווח הזמן ביניהם וקצב פעימות הלב

7 Execution graph, Task and CPU load

7.1 Execution graph

גרף הביצוע מציג באופן חזותי את סדר ביצוע הפקודות על ידי מחשב ומראה כמה זמן לוקח כל חלק בתוכנית. משתמשים בו על מנת למצוא צוואר בקבוק ולספק תובנות מדוע פעולות מסוימות נמשכות יותר זמן מאחרות לשם ייעול ביצועי התוכנה.
לשם בדיקה אמינה של הביצועים, הצבנו breakpoint בשלב שבו אינדקס מערך הדגימות מגיע לערך האחרון.

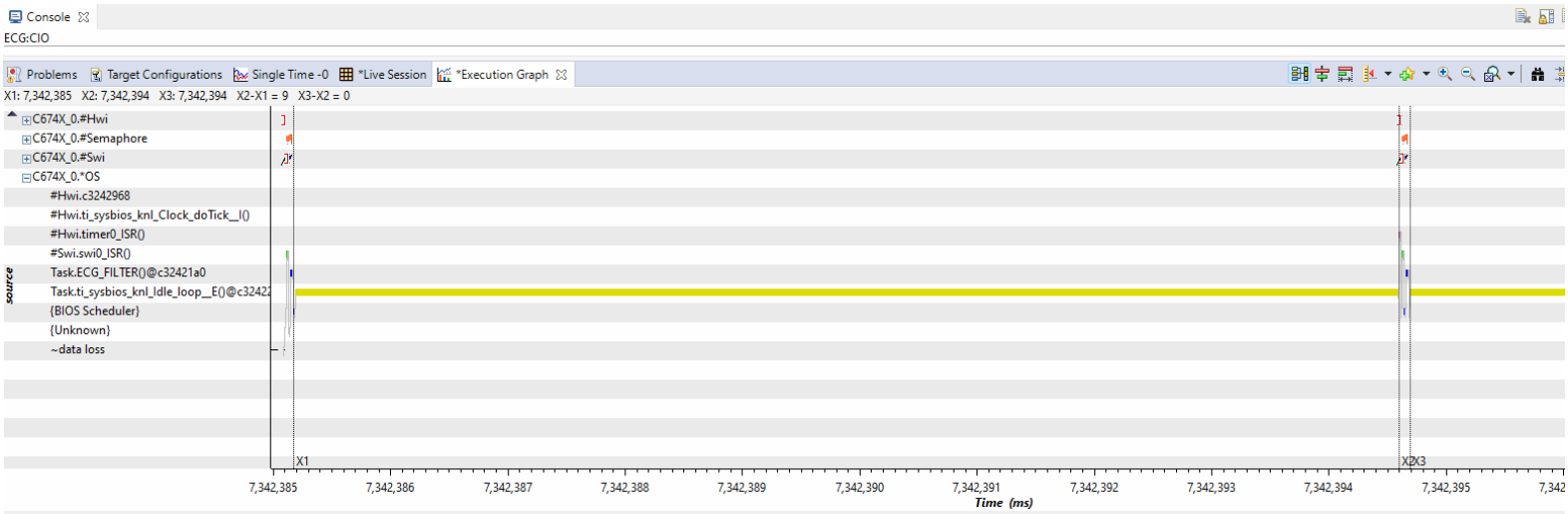
להלן גרף הביצוע של התוכנית שלנו:



איור מס. 35 Execution graph

ניתן לראות שהפונקציה הארוכה ביותר היא ה- Task. מכיוון שרוב התוכנית מתבצעת ב- Task, הגיוני שהיא הפונקציה הארוכה ביותר והיא נמשכת 22µsec. שגרות הפסיקה הינן קצרות מאוד, כאשר הפסיקת ה- Hwi נמשכת 12 µsec ופסיקת ה- Swi נמשכת 14µsec. פונקציית Task שהיא הארוכה ביותר בתוכנית אך עדיין קצרה מבחינת זמן ושגרות פסיקה קצרות מאוד מעידות על תוכנית יעילה מבחינת ביצועים. בנוסף, ניתן לראות שבסיום מחזור אחד של הרצה התוכנית חוזרת למצב Idle.

להלן הגרף לאחר שתי הרצות:



איור מס. 36 Execution graph בתום שתי הרצות

לאחר הרצה נוספת מדדנו את משך הזמן של ה-idle וראינו שהוא נמשך כ-9msec. הוא ארוך משמעותית מה-threads האחרים שמשכם ביחד 95µsec. בגלל ההבדלים הגדולים בסדרי הגודל, התוכנה עיגלה את פרק הזמן של ה-95µsec ל-0. פרק זמן ארוך מאוד של Idle לעומת threads אחרים מעיד על ביצועים טובים של התוכנית.

7.2 Task load

גרף ה- Task load (גרף עומס המשימות) מציג באופן חזותי את עומס העבודה של התוכנית.

להלן גרף עומס המשימות של התוכנית שלנו:

איור מס. 37 Task load

ניתן לראות שהעומס הגדול ביותר מתקיים כאשר התוכנית נמצאת במצב idle (~%99.16) ושאר ה-threads בשברי האחוז הנותרים. אחוז Idle גבוה ואחוז פסיקות נמוך בגרף העומס מעיד על תוכנית יעילה מבחינת ביצועים.

7.3 CPU load

איור מס. 37 CPU load

ניתן לראות שניצול המעבד הינו ~1.12%, כלומר המעבד נמצא בשימוש בפרק זמן קצר מאוד, דבר המעיד על ביצועים טובים של המערכת.

8 בעיות הנדסיות ופתרונן

בעיה	פתרון
בהתחלה חשבנו לתכנן מסנן IIR עם section יחיד, אך ראינו שהתוצאות שהתקבלו עדיין רועשות	תיכננו מסנן חדש עם 2 sections וראינו שהתוצאות השתפרו מאוד.
לא הצלחנו לבצע מדידה של ה- Execution graphs, CPU and Task load. גם כאשר הצלחנו לקבל גרפים הם היו נראים לא הגיוניים.	הוסבר לנו שעלינו להציב breakpoint בנקודה מסוימת, רצוי בנקודה שזה אינדקס המערך מגיע לערכו המקסימלי. לאחר ההצבה הצלחנו לקבל את התוצאות הרצויות.
דגימת דפיקות הלב שקיבלנו לא נמדד באיכות טובה וקיים בה פולס S2 שנדגם בעוצמה נמוכה מידי ולכן היה קשה לתפוס אותו בספירת ה-peaks	הורדנו את סף הThreshold על ידי ניסוי וטעייה עד אשר מצאנו ערך בו המערכת שלנו מצליחה לספור את הפולס ללא ספירה של רעש.