

DANMARKS TEKNISKE UNIVERSITET

EMBEDDED SYSTEMS 02131

Assignment 1

Gruppemedlemmer:

Cebrail ERDOGAN s113414

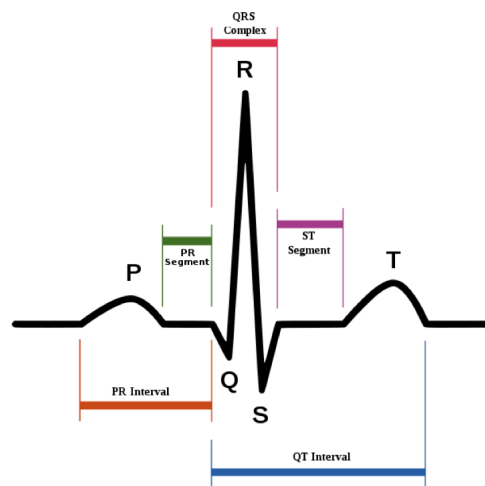
Christian Mathias Rohde KIÆR s123812

Jonathan Binner BECKTOR s123094

30. september 2013

INTRODUKTION

Vi har fået stillet til opgave at implementere Pan-Thompkins QRS detection algorithm. Algoritmens formål er at analysere et ECG signal og QRS komplekset. QRS komplekset giver det elektriske billede af ventriklernes sammentrækning. Et eksempel på et QRS kompleks ses på figur 1.



Figur 1: QRS Kompleks

Vores mål med algoritmen er derfor at give en vurdering af R peaks i QRS komplekset. Dette vil kunne alarmere patienter om eventuelle hjerteproblemer. Vores algoritme skal derfor kunne registrere følgende:

- Intensiteten af R-peaks.
- Varigheden af R-peaks.
- Tiden siden sidste R-peak.
- Advare hvis intensiteten af R-peaket er under en vis grænse eller hvis intervallet mellem R-peaks er ustabil.

INDHOLD

1	Kravspecifikationer	3
1.1	Funktioner	3
1.1.1	Data erhvervelse	3
1.1.2	Filtrering	3
1.2	Varierende del??	4
1.3	Bruger output??	4
2	Programdesign	4
2.1	Data indlæsning	4
2.2	Data analyse	4
2.3	Data output	5
3	Implementering	5
3.1	Filtre	5
3.1.1	Low-Pass, High-pass, Derivative og Moving window implementation . .	5
3.1.2	Squared	5
4	Resultater	5
5	Diskussion	5
6	Konklusion	5
7	Bilag	6
7.1	HelloWorld.c	6
7.2	ReadFile.c	6

1 KRAVSPESIFIKATIONER

Vi vil her beskrive kravene for programmet,

1.1 FUNKTIONER

1.1.1 DATA ERHVERVELSE

Vores program skal kunne simulere en ECG måling. Der skal derfor indlæses et datasæt, som repræsentere data fra en rigtigt patient, i realtime for så nøjagtigt som muligt at simulere en ECG måling.

1.1.2 FILTRERING

Da algoritmen kræver at datapunkterne behandles med stor præcision, anvendes 5 forskellige algoritmer til at filtrere den data vi har fået. De forskellige algoritmer gør alle noget forskelligt og meget specifikt for datapunkterne. Low-Pass filtrer som i navnet lader de lave frekvenser komme forbi og blokerer de alt for høje. Highpass gør det modsatte. Derivative forstærker de signaler vi er interesseret i. Square filtrer tildeles for at vise større forskel mellem datapunkterne og for at gøre alle datapunkterne positive. Tilsidst bruges Moving window implementation filtret for at gøre signalet mere "blødt" altså at fjerne ekstra støj for at forbedre visualiseringen af R-peaks. De 5 filtre skal anvendes i følgende orden:

- Low-pass filter
- High-pass filter
- Derivative
- Squaring
- Moving window integration

1.2 VARIERENDE DEL??

1.3 BRUGER OUTPUT??

2 PROGRAMDESIGN

STRUKTUR

Vi har taget udgangspunkt i de udleverede program filer til senoren, og efterfølgende oprettet klasser for alle filtre og QRS-registrings delene af programmet. Vi har en main klasse der kører selve programmet, og derfra kalder de nødvendige funktioner.

Filtrene er oprettet med navn, og funktionerne i det pågældende filter bliver derfra kaldt (eks. er low-pass filteret oprettet som lowPass.c). R-peak registrering forgår i QRS.c, hvorfra også udregningerne af intensiteten, varigheden og tiden siden sidste R-peak foregår.

Opdelingen er sket på denne måde, da vi synes en klasse for hver filter gav mest overskuelighed. Det er meget let derfra at gå ind og rette på en enkelt del af et givent filter, uden det giver problemer i resten af programmet. At holde QRS funktionerne adskilt fra resten af programmet giver samme fordele.

2.1 DATA INDLÆSNING

Da programmet skal forestille at være en simulation af en ECG måling, ved brug af Pan-Thompsons QRS detection algorithm, skal programmet kunne læse data fra en fil. Til dette har vi videreudviklet på sensor.c filen, og implenteret funktionen getNextData(). Dette vil give os muligheden for at hente data fra en fil, og bruge dette datasæt til analyse af QRS detection algoritmen. Sensor.c bruges derefter i main.c og henter alle vores signal data ind til analysen.

2.2 DATA ANALYSE

Analyseringen af data sker, når hver

2.3 DATA OUTPUT

3 IMPLEMENTERING

3.1 FILTRE

De forskellige filtre er belvet implementeret i hver sin klasse. Her er en gennemngang af nogle af dem da nogle af dem er meget ens hvor det eneste der gør dem forskellige er den matematiske formel.

3.1.1 LOW-PASS, HIGH-PASS, DERIVATIVE OG MOVING WINDOW IMPLEMENTATION

Her initialiseres 2 arrays en til resultaterne og en til dataen. Størrelsen af arrayet afhænger af hvor langt tilbage formlen siger vi skal have resultater fra. Fx. skal vi have x_{n-12} i Low-Pass filtret, vi skal altså have et array der gemmer x værdierne til og med x_{n-12} . Det bliver gjort med en forløkke som sætter den positon næste lig den forrige. Hvorefter vi så returner resultatet. Derudover har vi også tilføjet en reset metode hvis man skal bruge filtrene igen. Den sætter bare alle positionerne i arraysne til nul.

3.1.2 SQUARED

Squared filtret ligger også i sin egen klasse. her får den input fra Derivative og returner inputtet ganget sig selv.

4 RESULTATER

5 DISKUSSION

6 KONKLUSION

7 BILAG

7.1 HELLOWORLD.C

```
#include <stdio.h>

int main(int argc, const char * argv[])
{
    printf("Hello, World!\n");
    return 0;
}
```

7.2 READFILE.C

```
#include <stdio.h>

int main(int argc, const char * argv[])
{
    static const char filename[] = "/users/christiankiaer/Downloads/ECG.txt";
    FILE *file = fopen ( filename, "r" );

    int maxReading;
    int i1;
    while (fscanf(file,"%i",&i1) != EOF)
    {
        if (i1 > maxReading)
        {
            maxReading = i1;
        }
    }

    printf("The highest EKG reading is: %i", maxReading);
    return 0;
}
```