Aarhus universitet

Indlejret Signalbehandling

6. Semester

ISB projekt

Gruppemedlemmer:
Søren Landgrebe
Stinus Lykke Skovgaard

Studienr:
201508295
201401682

1. maj 2018

Indhold

Figurer

1 Indledning

Under optagelse til et live madprogram, sker det ofte at værten skal bruge en blender/food-processer. Dette betyder at værten ikke kan kommunikere med seerne mens blenderen/food-processeren kører. Denne problematik vil Noise Suppression System (NSS) kunne afhjælpe. Gennem en digital signal prossesering (DSP), vil vi dæmpe støjsignalet fra en køkkenmaskine dynamisk i realtid. Systemet består af to mikrofoner, et placeres tæt på støjen, et andet tæt på værten. De to mikrofoner fungere som input til vores system (Blackfin), hvor procceseringen og filtreringen foregår. Efter proceseringen bliver produktet afspillet på en højtaler, som erstatning for højtaleren fra et tv-apperat. Et overblik over systemet kan ses på figur 1

Figur 1: Konceptbillede for NSS

Med udgangspunkt i brugerens behov vil der blive opstillet en række brugsscenarier, der beskriver brugerens interaktion med systemet. Disse scenarier vil sammen med en række veldefinerede krav og afgrænsninger, danne grundlaget for designet af alle dele af systemet.

2 Krav

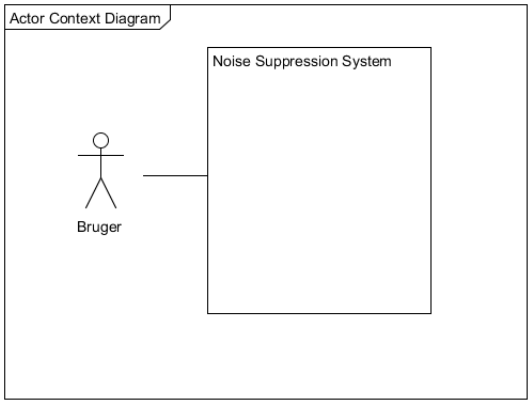
I dette afsnit beskrives kravene til hvilken funktionalitet systemet har.

I samarbejde med vejleder er der opstillet en række krav.

- Systemets skal kunne filtrere en støj fra et lydsignal, som indeholder et tale signal overlappet af et støjsignal.
- Brugeren skal manuelt kunne tænde og slukke for filtreringen.
- Lydsignalet skal feedes til en højtaler som afspiller lyden fra mikrofonen.

Kravene er bygget op gennem use cases, som beskriver systemets funktionelle krav, samt en liste over alle ikke-funktionelle krav for systemet. Først vises aktørbeskrivelserne med tilhørende aktørkontekst diagram. Herefter vises use cases for systemet. Der er udfærdiget to use cases der tilsammen beskriver funktionaliteten af systemet. I use case afsnittet vises der også et use case diagram med alle use cases og hvordan aktørerne interagerer med dem. Til sidst gives et kort overblik over ikke funktionelle krav.

3 Aktørbeskrivelse

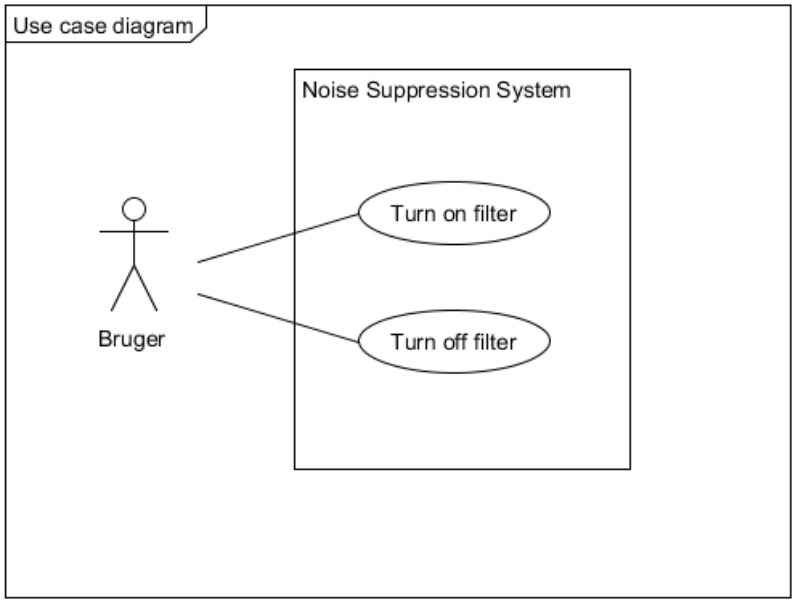


Figur 2: Aktør Kontekst diagram

På figur 2 ses aktør kontekst diagrammet som beskriver sammenhængen mellem aktøren og det system den interagere med. Aktøren er som følger:

Bruger: Den aktør der interagerer med systemet og vælger den ønskede funktionalitet

4 Use case beskrivelse



Figur 3: Usecase diagram

På figur 3 ses usecase diagrammet som beskriver sammenhængen mellem aktørerne og de forskellige funktionaliteter der findes for systemet.

4.0.1 UC 1 - Turn on filter

1. Mål: LMS filteret aktiveres
2. Initiering: Bruger
3. Aktøre: bruger
4. Prækondition: Systemet har strøm og er ikke tændt
5. Postkondition: Filteret er aktivt
6. Brugeren trykker på SW1 og filteret aktiveres

4.0.2 UC 2 - Turn off filter

7. Mål: LMS filteret deaktiveres
8. Initiering: Bruger
9. Aktøre: bruger
10. Prækondition: Systemet har strøm og er tændt

11. Postkondition: Filteret er ikke aktivt
12. Brugeren trykker på SW1 og filteret deaktiveres

5 Ikke-funktionelle krav

Kravene er delt op i tre underkategorier. Krav der relaterer til problemet. krav der relaterer til DSP platform og algoritme. Til sidst er der en kategori der beskriver kravene til systemet på baggrund af de to første kategorier.

5.1 Problemrelateret krav

1. R1: Systemet skal have 2 mikrofoner og 1 højtaler
2. R2: Filteret skal gøre brug af LMS algoritmen
3. R3: Systemet skal kunne processerer lyd i frekvensbåndet 50-20000Hz.
4. R4: Systemet skal kunne dæmpe uønkset støj 30dB.
5. R5: Systemet skal kunne dæmpe støj uden at dæmpe ønsket lydsignal.
6. R6: Systemet burde have en latency under 30ms.
7. R7: Systemet burde have et dynamikområde på min 96dB

5.2 System og algoritme krav

8. R8: Filter algoritmen skal implementeres med fixed point
9. R9: Filteret skal max bruge 10kByte memory
10. R10: Filteret skal implementeres på Blackfin BF533
11. R11: Filteret må max benytte 98% DSP load

5.3 Afledte krav

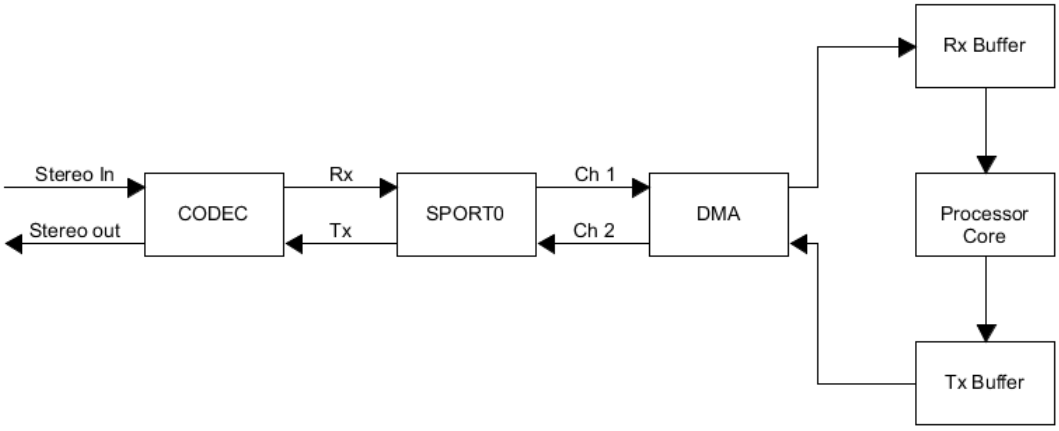
12. DR1: DSP systemet skal kunne håndtere en samplingsrate på min 44100kHz(På baggrund af krav R3)
13. DR2: Filteret skal implementeres med 1.15 fixed point.(På baggrund af krav R7 og R8. Dette giver et dynamikområde på 96dB)
14. DR3: Filter latency må max forsinkes 1280 samples(På baggrund af R6. (1/44100)*1280=30ms)
15. Filteret må max bruge 13333 cycles af DSP processering for hver sample.

6 Acceptest

Til hver use case er der lavet en tilhørende accepttest, der tjekker om use casen bliver gennemført korrekt. Udover de tre accepttests er der oprettet en accepttest af de ikke-funktionelle krav til systemet. Alle accepttests kan ses i kravspecifikations dokumentation.

7 Struktur

I dette projekt er der taget et valg udfra læringsmålene om at vi bruger Blackfin platformen til at udføre projektetes funktionalitet. Da en blackfin processor er bygget op af mange funktionalle blokke, vil der i det kommende afsnit laves et overblik over de hardware blokke som bliver brugt i dette projekt.



Figur 4: Struktur NSS

Igennem processen af vores funktionalitet, sendes lydsignalet gennem en codec1836, som har til opgave at sende inputtet gennem en ADC, hvorefter den sender det digitale signal videre til SPORT0, som står for at forbinde de interne blokke, igennem dette projekt bruger vi den interne clock, hvor frekvensen kan beregnes ud fra formlen:

$$\text{SPORTx\_TCLK frequency} = \frac{\text{SCLK frequency}}{2(\text{SPORTx\_TCLKDIV} + 1)}$$

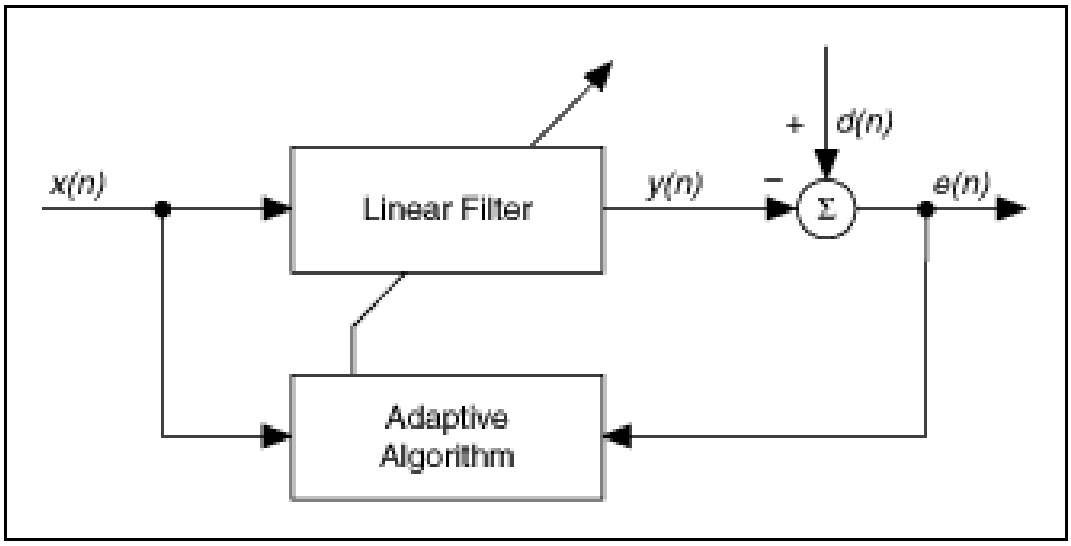
$$\text{SPORTx\_RCLK frequency} = \frac{\text{SCLK frequency}}{2(\text{SPORTx\_RCLKDIV} + 1)}$$

Figur 5: Formel for clock frekvens

Systemet fører herefter signalet til DMA'en som står for at placerer dataen i de rigtige registre i memory. Herefter sker selve proceceringen af dataen, hvor vores funktionalitet er implementeret. Det filtrede og bearbejde signal sendes herefter den modsatte vej og kommer igennem alle blokke igen og ender som et analogt signal på udgangen.

8 Teori

I procceseringsfasen af projektet, har vi valgt at bruge et adaptivt filter - LMS (Least Mean square) algoritme. Et adaptivt filter består af 2 funktionelle blokke, hvor den ene blok fungerer som et filter, fra dynamiske kofficienter som bliver opdateret LMS blokken. Filteret trækker herefter de beregnede filter fra det samlede lydsignal.



Figur 6: LMS adaptive filter

På figur 6 ses et overblik over det adative system, hvor x(n) er støjsignalet, y(n) er det filtrede støjsignal, med koefficienter som opdateres fra blokken "Adaptive Algoritm". d(n) er det ønskede signal inklusiv det støjende signal. e(n) er forskellen mellem d(n) og y(n) og derved støjen fratrukket fra det samlede signal af ønsket og støj.

Det digitale filter bliver beregnet ud fra formlen:

$$y(n) = \sum_{l=0}^{L-1} W(n) * x(n-1) \tag{1}$$

Hvor W(n) er den værdi, som dynamisk opdateres. Dette sker ved at vi beregner den næste værdi ud fra formlen:

$$W(n+1) = W(n) - \mu * X(n) * e(n) \tag{2}$$

Hvor W er den nye koefficient til filteret, X(n) er input signalet, og µ er en faktor, som bestemmer hastigheden af filteret, samt styrer infaldstiden. Hvis µ er lav bliver filteret langtsommere, mens settling time stiger jo højere vi kommer. Typisk må denne værdi ikke overstige 1.
Dette giver os et endeligt udtryk som stemmer overens med figur 6, ift summeringspunktet:

$$e(n) = d(n) - y(n) \tag{3}$$