

AARHUS UNIVERSITET

ANVENDTE MICROCONTROLLER SYSTEMER

6. SEMESTER

AMS projekt

Gruppemedlemmer:

Søren Landgrebe

Philip Nygaard Scmhidt

Studienr:

201508295

201506381



28. maj 2018

Indhold

1 Indledning	3
2 Krav	4
3 Aktørbeskrivelse	4
4 Use case beskrivelse	5
4.0.1 UC 1 - ADD DEVICE	5
4.0.2 UC 2 - REMOVE DEVICE	5
4.0.3 UC 3 - LOCK ON / LOCK OFF / AUTO	5
5 Sekvensdiagram	6
6 Ikke-funktionelle krav	9
6.1 Problemrelateret krav	9
6.2 System og algoritme krav	9
6.3 Afledte krav	9
7 System beskrivelse	10
7.1 Blokbeskrivelse BA-TA	10
7.2 Internal Block Diagram for BA-TA	11
7.3 Entity Relationship Diagram for BA-TA	11
8 System Arkitektur	13
9 Bruger Grænse Flade	14
10 Graphic display driver	20
11 Display driver	21
12 Touch driver	24
13 Fordele og ulemper	25

Figurer

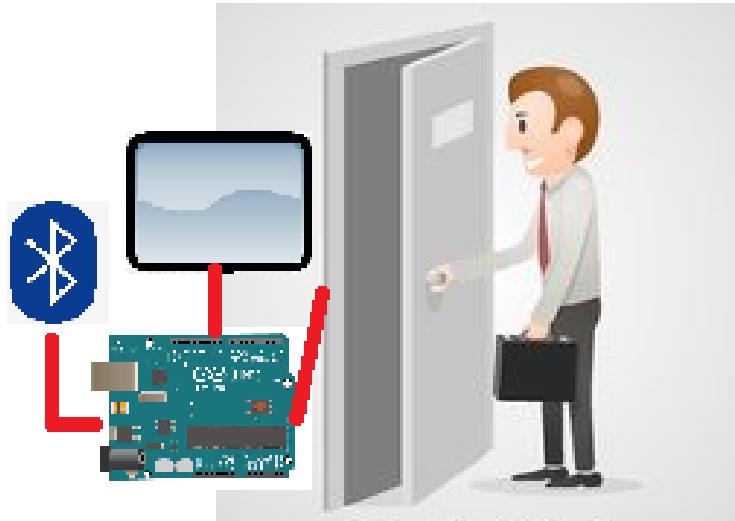
1	Konceptbillede	3
2	Aktør Kontekst diagram	4
3	Usecase diagram	5
4	Sekvensdiagram for use case 1	6
5	Sekvensdiagram for use case 2	7
6	Sekvensdiagram for use case 3	8
7	Bdd af BA-TA	10
8	Ibd af BA-TA	11
9	Entity Relationship Diagrams	12
10	welcome	14
11	Hovedmenu	15
12	Searching	15
13	Tilgængelige enheder	16
14	Valg af enhed	16
15	Fjern en enhed	17
16	Ingen enheder	17
17	Valgt fjernet en enhed	18
18	Låst op	18
19	Lås Automatisk	19
20	Bus Timing	21
21	The Dot Factory karakter	22

1 Indledning

Næsten alle danskere har nu til dags en smartphone med indbygget bluetooth modul, som man altid har med på sig når man forlader sit hjem. Dette vil vi gerne udnytte til at kunne gøre det nemmere for brugere, at kunne låse op og låse hoveddøren, som adskiller omverdenen fra ens dyrebare ejendele. Dette betyder at forbrugerne aldrig skal tænke mere på nøgler, da disse bliver overflødige. Dermed slipper brugeren for at skulle huske på dette, samt at skulle fumble med nøglerne når man kommer hjem med flere poser i hænderne fra dagens indkøbstur.

Denne prototype, BA-TA (Bluetooth Anti-Theft Alarm), vil derfor hhv. kunne låse op og låse hoveddøren, så snart systemet registrerer at husets ejer og dermed ejerens smartphone er i nærheden eller ej.

Hele systemet styres fra en samlet enhed, som kan integreres fra brugeren gennem en skærm og dertilhørende touch funktion.



Figur 1: Konceptbillede

Igennem brugergrænsefladen kan brugeren tilføje og fjerne enheder (personer) som skal kunne låse og låse op for døren.

Når alle godkendte enheder er uden for rækkevidde af systemets bluetooth scanner, låses døren, og så snart systemet ser en godkendt enhed indenfor rækkevidden bliver døren låst op.

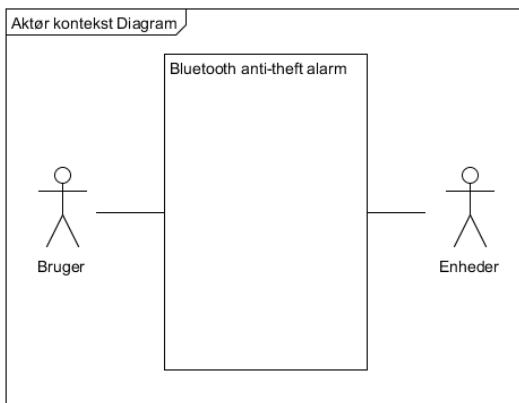
2 Krav

I dette afsnit beskrives kravene til hvilken funktionalitet systemet har.

I samarbejde med vejleder er der opstillet en række krav.

- Systemet skal kunne finde de 4 sterkeste bluetooth signaler.
- Systemet skal kunne køre uafbrudt
- Systemet skal scanne efter enheder hvert 5. sekund

3 Aktørbeskrivelse

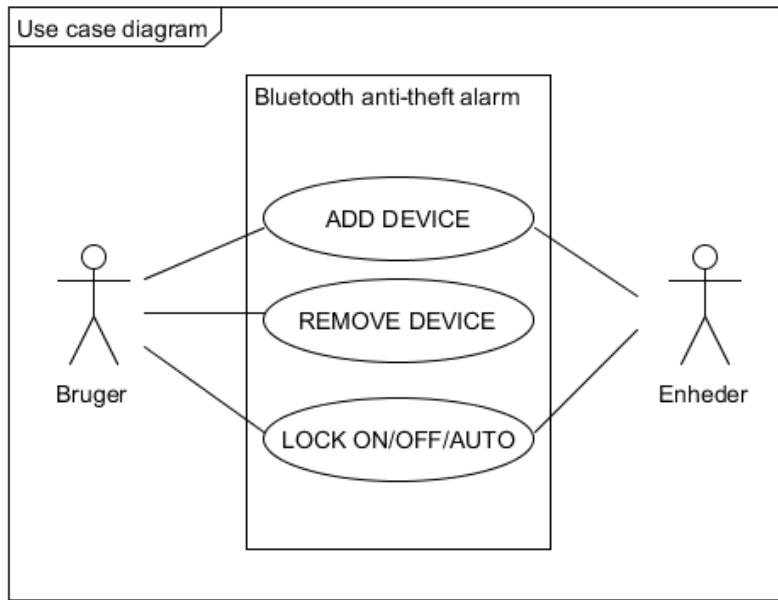


Figur 2: Aktør Kontekst diagram

På figur 2 ses aktør kontekst diagrammet som beskriver sammenhængen mellem aktørene og det system de interagere med. Aktørene er som følger:

- **Bruger:** Aktøren der interagerer med systemet og dets funktionalitet
- **BA-TA:** Bluetooth Anti-Theft Alarm systemet
- **Enheder:** Op til 4 Bluetooth enheder som er registreret af BA-TA

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 - ADD DEVICE

- Brugeren trykker på ”ENTER” når pointeren er ved ”ADD DEVICE”.
- Systemet scanner efter 4 stærkeste Bluetooth signaler
- Brugeren kan vælge imellem de registrerede bluetooth-enheder, som skal tilføjes til listen over de godkendte enheder.

4.0.2 UC 2 - REMOVE DEVICE

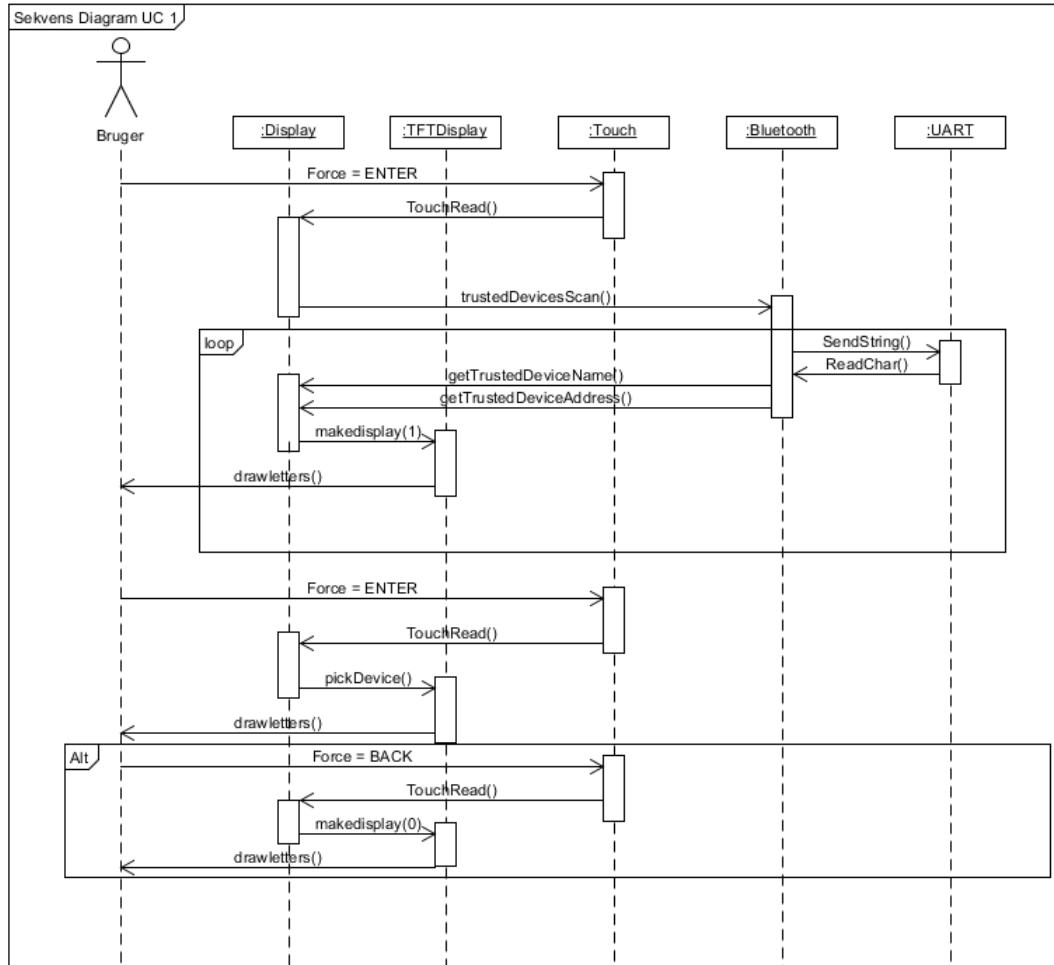
- Brugeren trykker på ”ENTER” når pointeren er ved ”REMOVE DEVICE”.
- Systemet præsenterer listen over de godkendte enheder
- Brugeren kan vælge imellem de godkendte bluetooth-enheder, som kan fjernes fra listen.

4.0.3 UC 3 - LOCK ON / LOCK OFF / AUTO

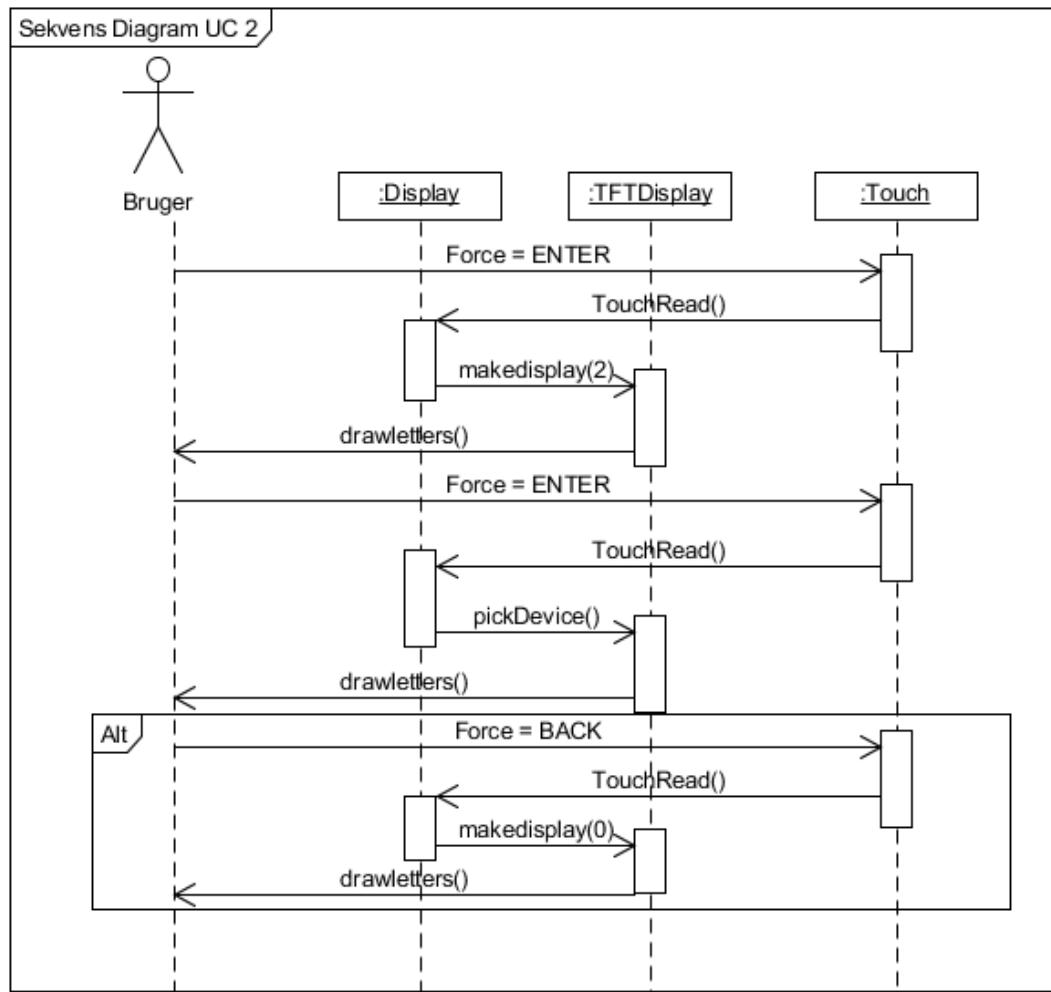
- Brugeren trykker på ”ENTER” når pointeren er ved ”LOCK ON/”LOCK OFF/”LOCK AUTO”.
- Systemet skifter state i rækkefølgen 1 - LOCK ON 2 - LOCK OFF 3 - LOCK AUTO.

5 Sekvensdiagram

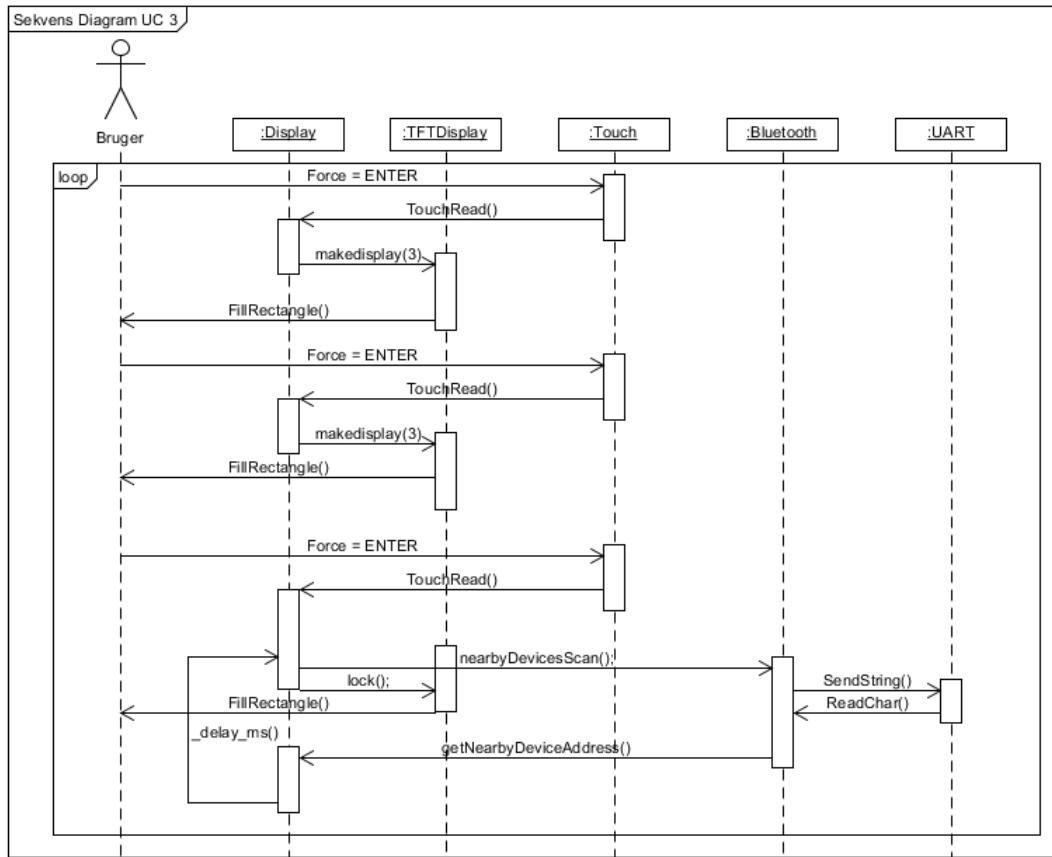
I dette afsnit gives et overblik over funktionaliteterne ud fra use cases, som er præsenteret i foregående afsnit.



Figur 4: Sekvensdiagram for use case 1



Figur 5: Sekvensdiagram for use case 2



Figur 6: Sekvensdiagram for use case 3

6 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.

6.1 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 80dB

6.2 System og algoritme krav

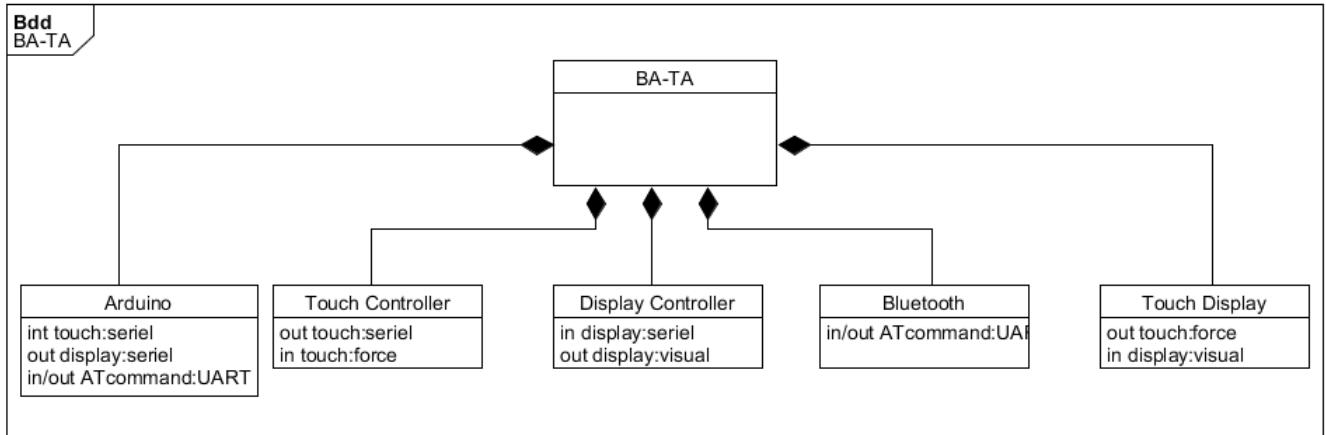
8. R8: Filter algoritmen skal implementeres med fixed point
9. R9: Filteret skal max bruge 10kByte memory
10. R10: Filteret skal implementeres på Blackfin BF533
11. R11: Filteret må max benytte 98% DSP load

6.3 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. DR4: Filteret må max bruge 13333 cycles af DSP processering for hver sample. (På baggrund af R10, R11 og DR1 $(600MHz/44.1kHz)*98\%$)

7 System beskrivelse

I dette afsnit dannes et overblik over systemets blokke og de interne forbinder. Derudover gives der et overblik over funktionerne som er implementeret i projektet. På figur 7 ses et overordnet Bdd for projektet, hvor de interne forbindelser forklares på figur 8.



Figur 7: Bdd af BA-TA

***Skal vi nøjes med et mere simpelt BDD hvor der eksempelvis ikke står nogen til Arduinoen og de andre?

7.1 Blokbeskrivelse BA-TA

I det følgende vil komme en beskrivelse af de enkelte blokke i BA-TA og deres interne funktionalitet.

Arduino

Arduino'en, mega2560, initialiserer og styrer alt funktionaliteten som indgår i systemet. Arduino blokken står for at initialisere Touch Display, Touch Controller og Display Controller, som bruges til at kunne modtage signaler fra Touch-delen, samt at sende signaler til Display'et og dermed få repræsenteret noget til brugeren. Yderligere bruges Arduinoen til at kommunikere med Bluetooth-blokken.

Touch Controller

Touch Controller står for at modtage værdier fra Touch Display, samt at sende den modtagede værdi videre til Arduino'en.

Display Controller

Display Controller modtager værdier fra Arduino'en og sende værdien videre til Touch Display.

Bluetooth

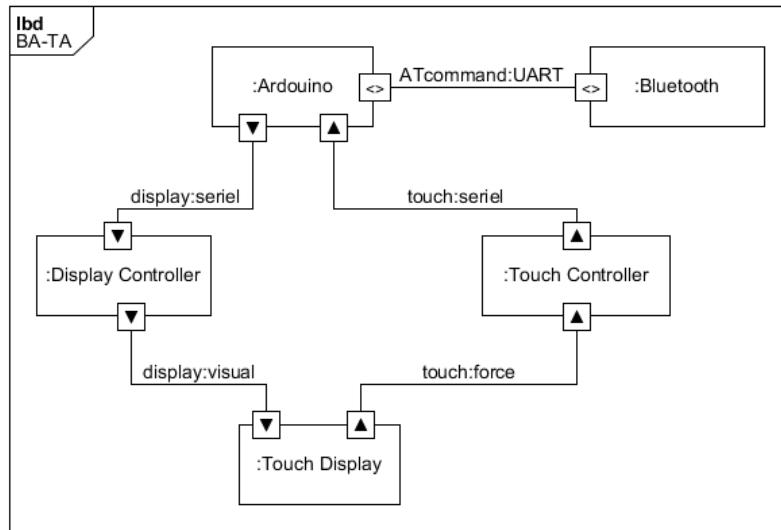
Bluetooth modul, HC-05, som kommunikkerer med Arduino'en vha. AT-kommandoer.

Touch Display

Touch Display fungerer som brugergrænseflade og er dermed måden brugeren kan interagere med systemet.

7.2 Internal Block Diagram for BA-TA

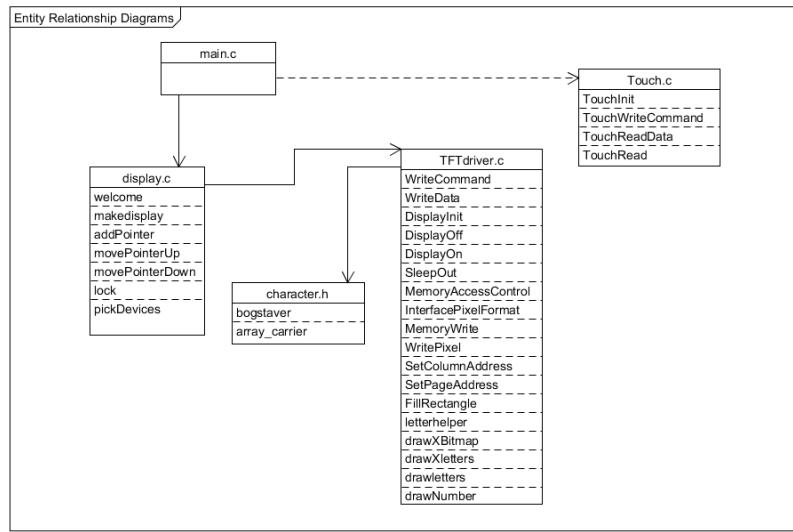
På figur 8 ses et overordnet IBD over selve systemet, som er bygget på Bdd'et.



Figur 8: Ibd af BA-TA

7.3 Entity Relationship Diagram for BA-TA

Herefter gives et overblik på figur 9 over koden til driverne som er skrevet til dette projekt.
*** Skal opdateres når vi er færdige med alt kode



Figur 9: Entity Relationship Diagrams

8 System Arkitektur

9 Bruger Grænse Flade

I dette afsnit gives der et overblik over brugergrænsefladen og brugerens mulighed for interaktion med denne. Afsnittet er bygget op af billeder, hvor hvert billede viser brugergrænsefladens visuelle struktur.

Første display man møder er velkommen. Her går Arduino'en igang med at initialisere de forskellige moduler, således de er klar til brug.



Figur 10: welcome

Næste display er hovedmenuen for systemet, hvor der er tre forskellige muligheder, som ses på

figur 11. Yderligere ses der 4 trykknapper, hvor brugeren kan interagere med systemet. Med "Enter" vælger brugeren en funktion. "Back" gør brugeren i stand til at gå tilbage til hovedmenuen, hvis man forinden har trykket "ENTER" ved "ADD DEVICE" eller "REMOVE DEVICE". "Up" og "Down" flytter pilen, hhv. op og ned. Herudover ses der en farvet boks øverst i højre hjørne, som indikerer låsens tilstand. Denne lås viser grøn ved låst op og rød ved låst. Øverst til venstre på skærmen vises projektets logo.



Figur 11: Hovedmenu

Hvis brugeren trykker på ”ADD DEVICE” søger BA-TA efter de 4 størkeste Bluetooth signaler, som er indenfor rækkevidde.



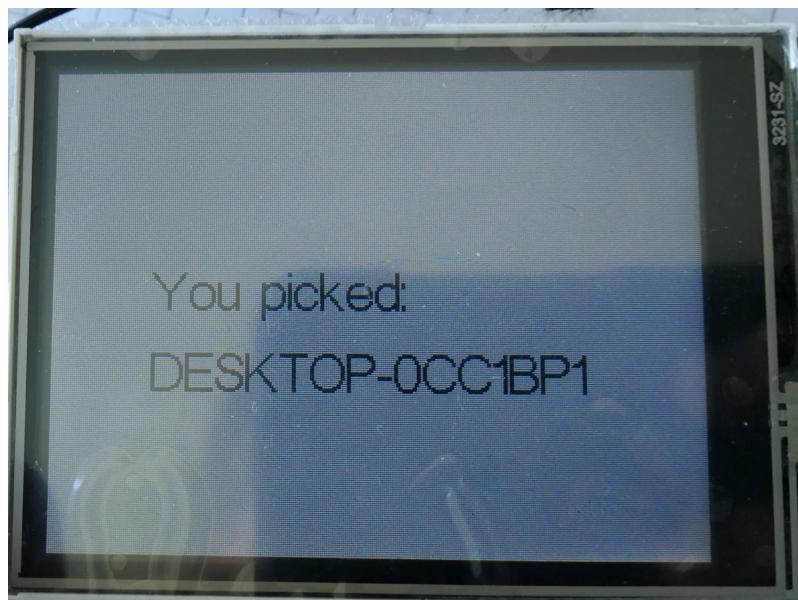
Figur 12: Searching

Herefter vises de fundne Bluetooth-enheder, som systemets Bluetooth-modul har fundet. Dette ses på figur 13.



Figur 13: Tilgængelige enheder

Hvis der vælges en enhed ved et tryk på ”Enter”, så vises navnet på den valgte enhed. Herefter går skærmens tilstand tilbage til hovedmenuen.

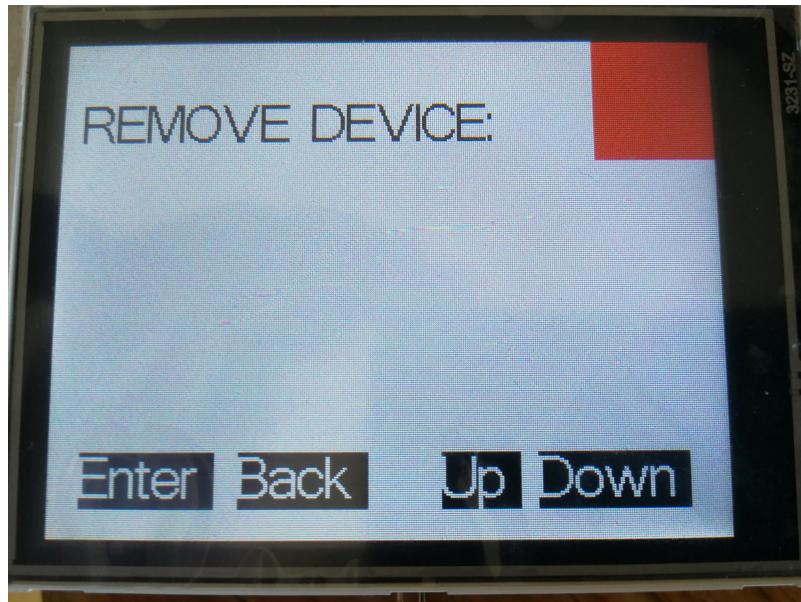


Figur 14: Valg af enhed

Hvis brugeren herefter vælger funktionen ”REMOVE DEVICE” fra figur 11, åbnes der en liste over de godkendte Bluetooth-enheder, der er gemt. Dette ses på 15.. Hvis der på forhånd ikke er nogle gemte godkendte Bluetooth-enheder og listen dermed er tom, så vil displayet også være tomt, som vist på figur 16.



Figur 15: Fjern en enhed



Figur 16: Ingen eneheder

Såfremt der er godkendte Bluetooth-enheder på listen, kan brugeren vælge at slette en bestemt enhed. Dette sker ved at trykke "Enter" ved enheden, som vælges ved hjælp af "Up" og "Down" og indikeres af pilen. Displayet viser på figur 17 en besked om hvilken enhed brugeren har valgt at slette.



Figur 17: Valgt fjernet en enhed

Hvis brugeren vælger den sidste mulighed/state i hovedmenuen, "LOCK OFF", låses systemet op, teksten skifter og låsen skifter farve. Dette ses på 18.



Figur 18: Låst op

Hvis der endnu engang trykkes på den sidste state i hovedmenuen, ændres teksten, og låsen skifter, alt efter BA-TA's funktionalitet. Hvis der trykkes endnu engang på sidste state, ændres låsen til default, hvilket er låst.



Figur 19: Lås Automatisk

10 Graphic display driver

Som brugergrænseflade i dette projekt bruges et ITDB02, som er valgt da der tidligere er arbejdet med netop dette produkt. Dertil kommer ILI9341 som driver til selve display'et.

Driver softwaren til hele displayet, er delt op i flere forskellige cpp filer, dette er gjort for at gøre koden mere overskuelig, og gøre funktionaliteten mere effektiv. Herunder forsøges at gøre et overblik over de forskellige cpp filer og deres integeren.

—Skriv noget når det hele er færdig her

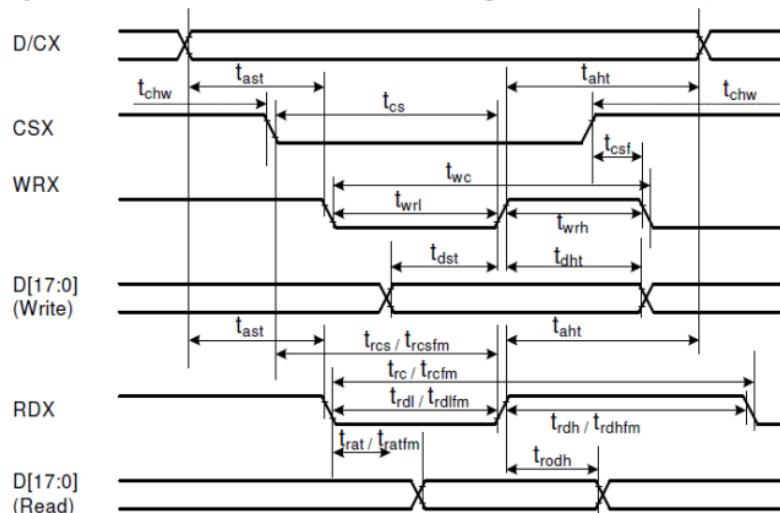
11 Display driver

Igennem øvelser til undervisningen, er der blevet opbygget en driver til et grafisk display. Igennem processen til dette projekt er der blevet modifieret og arbejdet videre på selvsamme driver. Cpp filen ligger under TFTdriver, som er bygget op af flere forskellige funktioner. I dette afsnit vil der gives et overblik over de mest essentielle funktioner. For at få en forståelse af selve opbygningen af driveren, henvises til datasheet for controlleren. [ILI9341](#)

For at opbygge en basisforståelse for driveren, vil der først blive forklaret `DisplayInit()`. Først sætter vi de porte vi skal bruge fra Arduino til hhv. indgange og udgange. Vi har dog valgt ikke at have tilbagemeldinger, og derfor er der ikke initialiseret nogle indgange. Herefter sætter vi RESET, CS, WRX, RDX høje ift figur 20. Der bliver kørt en reset (tjek lige koden med timing, burde være kortere), for at resette displayet, og bagefter sendt fire kommandoer, som kan findes i command list i databladet. Sleep Out, Display On, Pixel format set = 16 bit og Memory Access Control (BGR = 1) (Tjek i henning hvad dette gør).

18.3 AC Characteristics

18.3.1 Display Parallel 18/16/9/8-bit Interface Timing Characteristics (8080- I system)



Figur 20: Bus Timing

Herunder vil de essentielle funktioner deles op i hvert sin tabel. Flere af funktionerne gør brug af både `Writecommand()` og `Writedata()`, som er opbygget udfra Bus Timing figur 20. Derudover bruger flere af funktionerne `SetColumnAddress()` og `SetPageAddress()`, som er bygget op fra datasheet 8.2.20. og 8.2.21.:

Included Font:

character.h
Karakter størrelse: 24*24 pixels
Antal karakter: 95



Figur 21: The Dot Factory karakter

Funktions:

FillRectangle(StartX,StartY, Width, Height, Red, Green, Blue)
Funktion som laver en firkant med den valgte baggrundsfarve
Paramentre: StartX: Startværdi på x-aksen StartY: Startværdi på y-aksen Width: Bredden på firkanten Height: Højde på firkanten Red: farve(værdi) Green: farve(værdi) Blue: farve(værdi)

letterhelper(numberletter, startx, starty, Red, Green, Blue)
Funktion som bruger input fra Include filen character, til at bestemme hver karakter bredde og længde
Parametre: numberletter: Bestemmer hvilken karakter der skal hentes Startx: Startværdi på x-aksen Starty: Startværdi på y-aksen Red: farve(værdi) Green: farve(værdi) Blue: farve(værdi)

drawXBitmap(bitmap[],length,count,startx,starty, letter, modulus,Red, Green, Blue)

Funktion som står for at skrive til hver bit, med værdier fra letterhelper()

Paramentre:

bitmap[]: Henter en byte fra character.h

length: Fortæller funktionen, hvor bred karakteren der skal skrives er

count: hvor mange bytes funktionen skal køre igennem for at lave hele karakteren

Startx: Startværdi på x-aksen

Starty: Startværdi på y-aksen

Red: farve(værdi)

Green: farve(værdi)

Blue: farve(værdi)

Note: Funktionen sletter gamle karakterer, da en ny smallere karakter end forrige stadig vil forblive på displayet

drawletters(str[],startx, starty,Red, Green, Blue)

Funktion som modtager en en streng, og konverterer ascii værdien til den rette værdi iif character.h

Paramentre:

str[]: Modtager en streng

Startx: Startværdi på x-aksen

Starty: Startværdi på y-aksen

Red: farve(værdi)

Green: farve(værdi)

Blue: farve(værdi)

drawNumber(number,startx, starty,Red, Green, Blue)

Funktion som modtager en en integer, og konverterer ascii værdien til den rette værdi iif character.h

Paramentre:

number: Modtager et interger tal

Startx: Startværdi på x-aksen

Starty: Startværdi på y-aksen

Red: farve(værdi)

Green: farve(værdi)

Blue: farve(værdi)

setBaggroundPixel(int red, int green, int blue)

Funktion som sætter baggrundsfarven af tekst og tal

Paramentre:

Red: farve(værdi)

Green: farve(værdi)

Blue: farve(værdi)

12 Touch driver

Da projektet skulle bruge integering af en bruger, er der valgt at inkudere en touch driver som gør brug af **XPT2046** Touch Screen Controller, som allerede var en del af **ITDB02** Arduino MEGA shield, som bliver også brugt i Display driveren.

Driveren har tre funktioner, hvor Init() sætter de forskellige porte til hhv indgange og udgange, og derefter sætter de respektive ben til enten høj og lav. Funktionen pulse() står for at lave en puls på clk benet som er opsat ift. Figur 15 i **XPT2046** datasheet. Herunder vil der laves en tabel over den sidste funktion.

TouchRead(xy)
Funktion som står for at læse værdien fra brugerinputtet.
Paramentre:
xy: Styrer om retur værdien skal være for x eller y
Retur: Værdien for enten x eller y.

13 Fordeler og ulemper

Da projektets omfang og opbygning har været meget frit, er der også blevet lavet nogle valgt og fravælg i process fasen. Dette vil vi give et overblik over i dette afsnit, forklare fordele og ulmepor ved hvert modul vi har valgt, og driverne dertil. Herunder gives et overblik over de emner gruppen mener har været mest essentielle ift fordele og ulemper af de forekellige moduler.

TFT Display

Da omfanget for projektet inkluderede et display, gjorde gruppen et valg om at bruge TJC-9341-032 som display, og **ILI9341** som display controller. Dette **ILI9341** er valgt på baggrund af, gruppen har arbejdet med dette display modul tidligere, derudover var dette også tilgængeligt. Til selve display'et er der valgt kun at kunne skrive til display'et og ikke læse fra det. Fordelen ved kun at sende data til displayet, er at det gør opsætningen meget nemmere for udvikleren.

Fordelen ved at initialisere driveren til at kunne modtage data fra displayet, ville være at programstrukturen, kunne spørge skærmen hvilket display, der var på skærmen nu. Dette vil sikre at programstrukturen altid ved hvilket frame der vises på displayet. Ulempen ved den måde gruppen har initialiseret driveren på, er at microcontroller skal holde styr på hvilken frame, der vises på skærmen. Dette gør det mere udfordrende for udvikleren, og derved gør at koden kommer til at fylde mere. Hvis driveren skulle laves på en memory kritisk microcontroller, ville det være en fordel at tilføje skærmen at kunne læse hvilket frame der står på skærmen.

Vi har ikke brugt meget tid på at undersøge alternative display. Der blev valgt at et Alphanumeric display, som tidligere var bearbejdet, ikke ville opfylde de krav vi havde til displayet, og da controlleren shielded ITDB02, havde en Touch controller del, blev Alphanumeric display'et valgt fra.

Gør brug af globale variabler

Igennem de forskellige drivere gøres brug af globale variabler, dette er gjort da for at simplificere udviklingen af driverene. Denne metode at programmere på er dog ikke god program skik, og hvis gruppen havde afsat mere tid til at udvikling, ville det første være at erstatte disse globale variabler, med get og set metoder.

Bluetooth

Litteratur

- [1] Gan and Kuo.
Embedded Signal Processing with the Micro Signal Architecture, Chapter 4.4.1
John Wiley 1st Ed. 2007.
- [2] Gan and Kuo.
Embedded Signal Processing with the Micro Signal Architecture, Chapter 7.2.2.1
John Wiley 1st Ed. 2007.