Aarhus School of Engineering

Electronic Engineering

E4PRJ

Detaljeret Software Design

Author:
Nicolai Glud
Johnny Kristensen
Rasmus Lund-Jensen
Mick Holmark
Jacob Roesen

30. november 2012

Indholdsfortegnelse

Indledning 1

Dette dokument beskriver det detaljerede SW-design for BROS, som er fastlagt ud fra dokumenterne kravspecifikation og systemarkitektur.

1.0.1 Formål

Formålet med dokumentet er:

- At fastlægge systemets detaljerede softwarestruktur udfra kravene specificeret i kravsspecifikationen. Derudover beskrivelsen af softwarekomponenterne og deres grænseflader beskrevet i systemarkitektur-dokumentet.
- At fastlægge systemets softwareklasser og deres indbyrdes interaktioner.
- At beskrive de enkelte klassers vigtigste metoder.

1.0.2 Reference dokumentation

- Kravspecifikation for projektet.
- Systemarkitektur-dokument.

KI 2

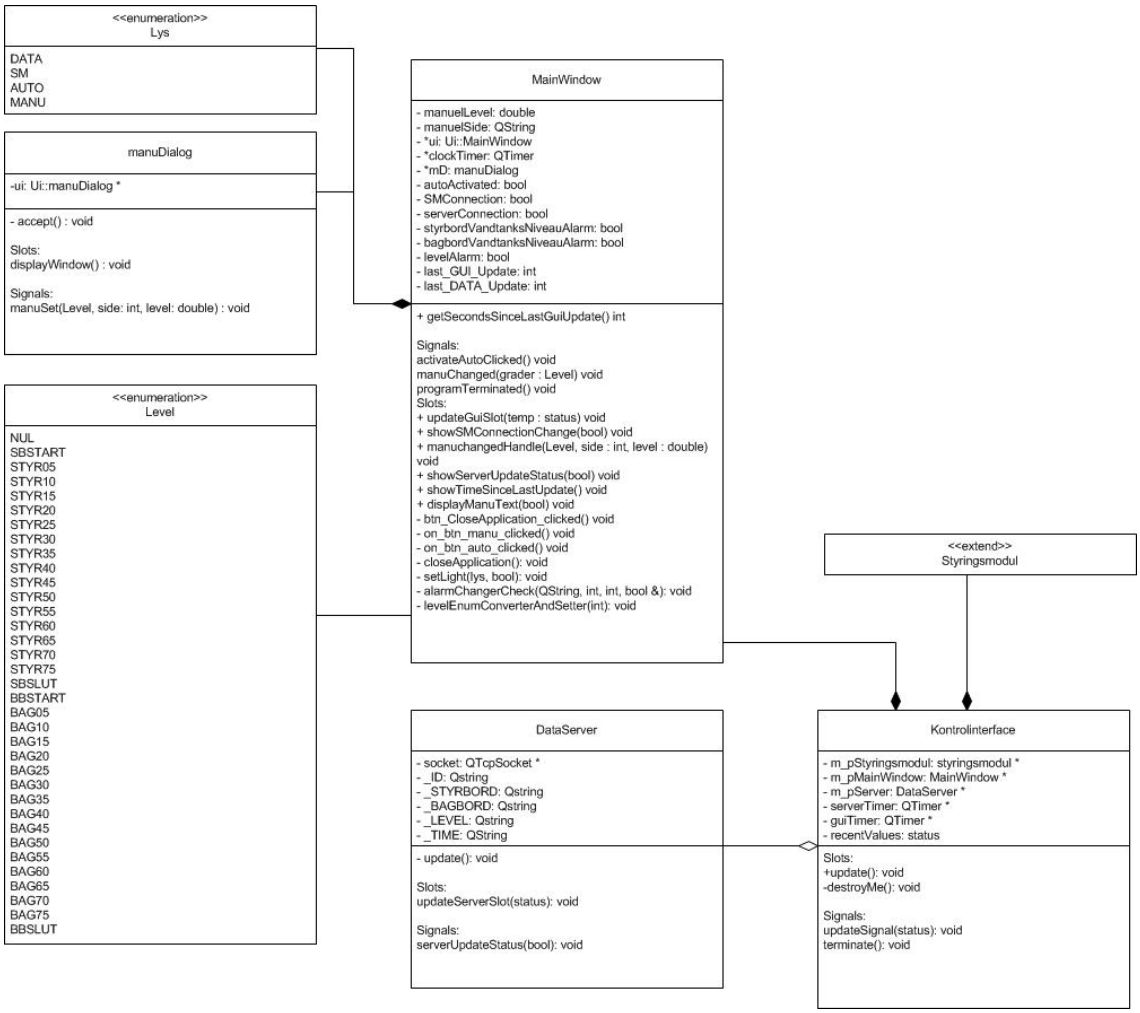
Nedenfor følger design af software til Kontrolinterfacet. Dette er lavet på baggrund af kravspecifikation og systemarkitektur.

2.0.3 Modulets ansvar

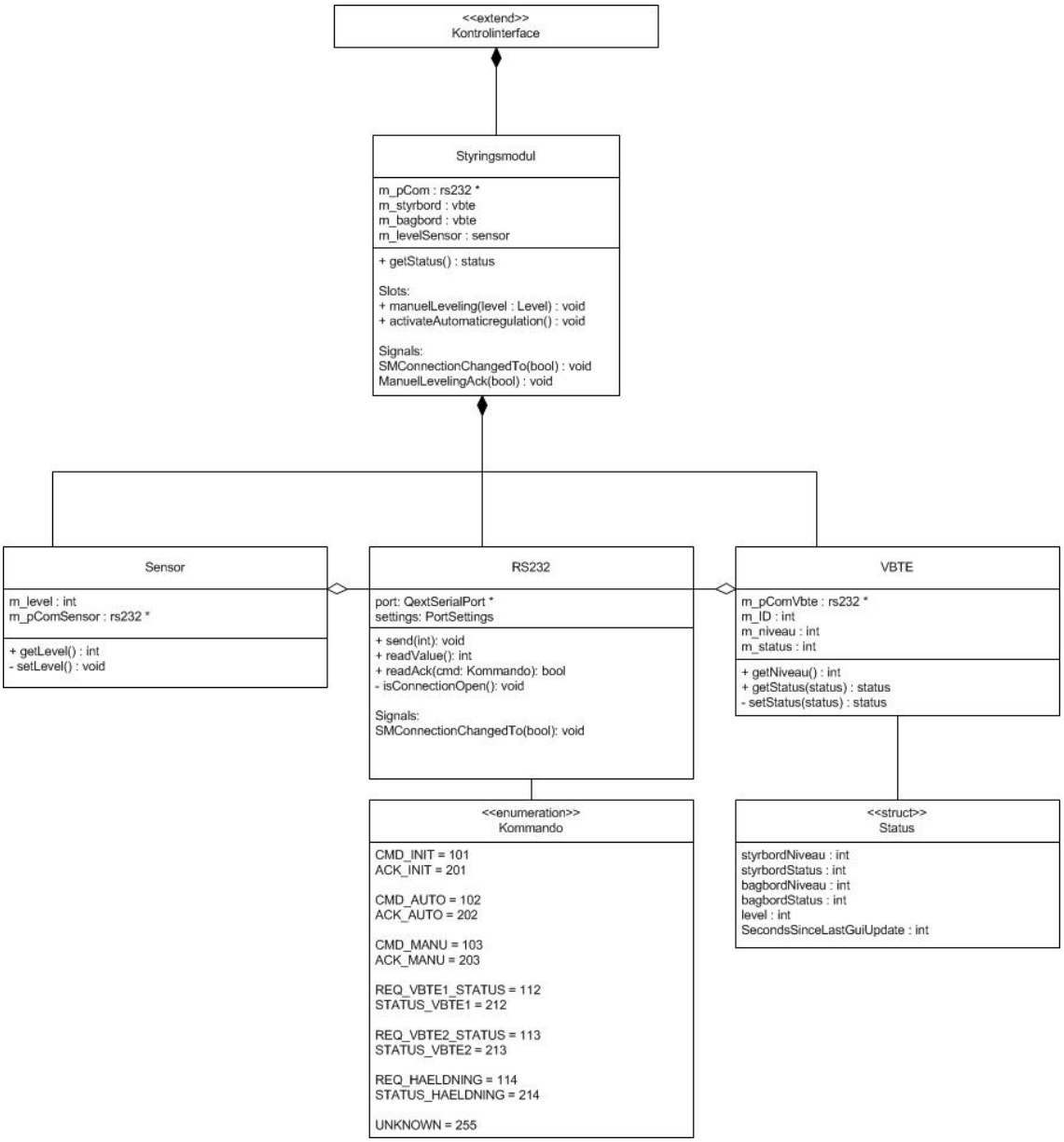
Kontrolinterfacet er brugerens primære kontaktflade til systemet. Programmet indeholder en brugergrænseflade der opfylder kravene i Kravspecifikationen. Her kan der også ses en prototype på den grafiske brugergrænseflade. Kontrolinterfacet står for at modtage inputs fra brugeren. Disse inputs sendes som kommandoer til Styringsmodulet. Det er også herfra at Kontrolinterfacet modtager de værdier, som sidenhen vises på den grafiske brugergrænseflade. Kontrolinterfacet står også for kommunikationen til den eksterne database. Her sendes en række parametre om skibet og dets status.

2.0.4 Klassediagram

Nedenfor ses klassediagrammet for Kontrolinterfacet. Bemærk at klassediagrammet er delt op i to. Skæringsstedet er mellem Kontrolinterface-klassen og Styringsmodul-klassen og er markeret med «extend».



Figur 2.1. På figuren ses klassediagrammet for KI - Kontrolinterface-delen



Figur 2.2. På figuren ses klassediagrammet for KI - Styringsmodul-delen

2.0.5 Globale variabler

Variabel	Beskrivelse
BurstLengthVal	Denne variabel er anvendt til at håndtere antallet af perioder burstet bliver sendt med.
WaitBurstVar	Bliver brugt til nonblocking delay til SendBurst funktionen.
BurstTimerVal	Holder på Timerens værdi når et burst er sendt.
DistanceTimerVal	Holder værdien på timeren når et burst er modtaget.
CalcDistFlag	Bliver sat når et burst er modtaget så en afstand kan blive beregnet.

2.0.6 Valve

Ansvar

Denne header har til ansvar at styre ventilerne ud fra "state-variablen modtaget fra I2C_handle. Headeren benytter PSoC-API'et til kontrol registre..

Funktionsbeskrivelser

void ChangeState(uint8 state);

Beskrivelse:	Funktionen anvender API'et fra I2C blokken i PSoC miljøet. Med disse tjekker den om der er fyldt nyt i bufferen og aflæse dette. Herfer kalder den funktionen I2C_decode(); til at afkode beskeden fra SM. Herefter klargøres readbufferen til evt. at sende vandniveau tilbage.
Parametre:	uint8* WriteBuffer uint8* ReadBuffer uint8 BufferSize
Returværdi:	uint8 State

2.0.7 Dist

Ansvar

Denne header har til ansvar at sende burst, beregne afstanden samt at omregne afstanden til procent.

Funktionsbeskrivelser

void SendBurst(void);

Beskrivelse:	tis
Parametre:	hund
Returværdi:	henning

2.0.8 Eventuelle Sekvensdiagrammer og state machines

hab hab

Database 3

3.1 Klassediagrammer

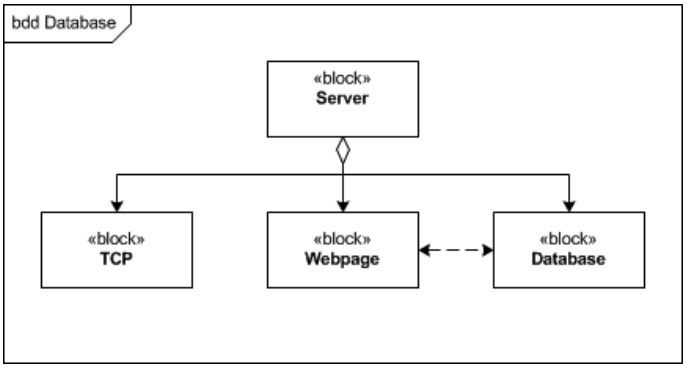
3.1.1 Server

3.1.2 TCP server

3.1.3 Wep-side

3.2 Design

3.2.1 Server



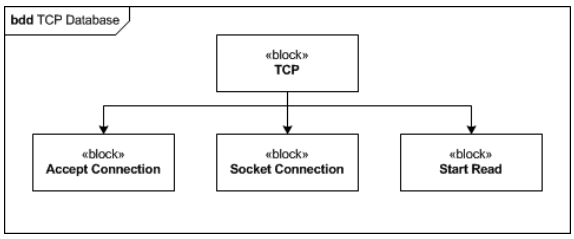
Figur 3.1. BDD server

Databasen er en server som har de 3 underblokke TCP, Database og Web-page som illustreret på figur 3.1

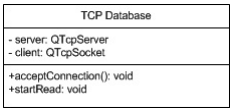
TCP er en dataforbindelse (Transmission Control Protocol). Denne protokol er benyttes til at sende data fra KI til serveren. Severen vil modtage data og lagere dette i en midlertidig backup fil. TCP-forbindelsen er kodet i C++. For TCP-forbindelsen benyttes TCP - protocollen som tilbyder sikker data overførelse fra BROS. Databasen er en mySQL database som frit kan downloades og installeres på en Linux, Windows eller Mac. Man skal installere en server del og en client del. Server delen er den del der gør det muligt at håndtere og lagre data. Denne ligger under client delen og er nødvendig for at client kan fungere. mySQL client gør det muligt at en bruger kan tilgår og læse fra databsen uden at gør ændringer i denne. Dette benytte i web interfacet. mySQL kan tilgås direkte fra terminalen og giver mulighed for forskellige opsætninger af databaser og tabeller samt forskellige bruger rettigheder. **mulighede med mySQL** Web-pagen er udviklet i php som giver gode muligheder for kommunikation til og fra mySQL databasen. Web-pagen

er implementeret ved hjælp af en apache server som er en web server. Web-pagen har en general information omkring BROS og et login til at tilgå databasen. Ved at anvende et web-interface gives der mulighe for at data kan aflæses fra andre pladser end fra den direkte server. Ved at kende ip eller host navn er det muligt at tilgå web-siden. Den web baserede database loader mySQL databasen og fremviser denne grafisk for brugeren.

3.2.2 TCP server



Figur 3.2. BDD TCP server(ikke færdig)



Figur 3.3. UML Databaser

TCP protocollen er en af kerneprotokollerne på nutidens internet. Gennem TCP kan forskellige værtsmaskine igennem f.eks. internet ethernet og trådet forbindelse oprette forbindelse til hinanden og udveksle datepakker. Protokollen giver programmelt på værtsmaksine nogle vitale garantier for at disse datapakker afsendes og modtages ved:

- Stabilitet: En pakke der går tabt bliver forsøgt afsendt igen
- Ordnet levering: En pakke kommer frem til modtageren i samme rækkefølge som de blev afsendt

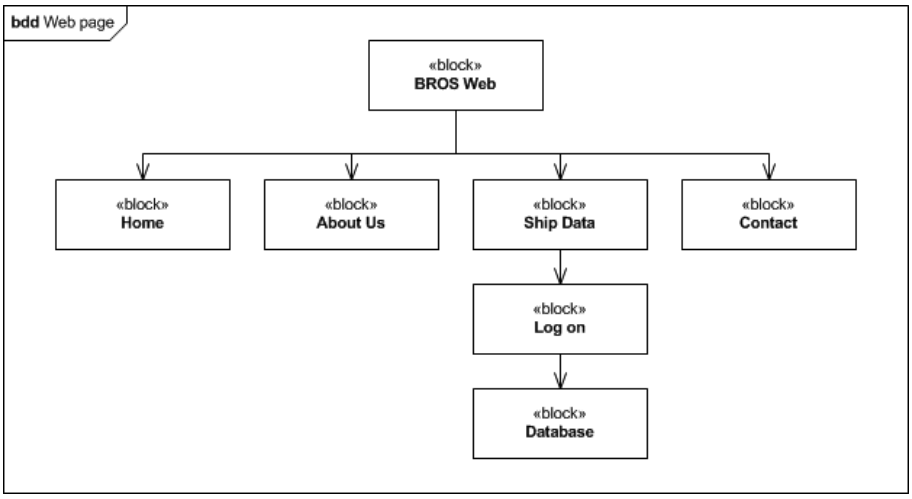
Der ud over benytter TCP sig af forskellige port numre. Forskellige portnumrer gør dte muligt etablere flere forskellige datstrømme til og fra samme værtsmaskine.

Selve programmet er kodet op over socket programmering. Under opstart initialiseres socket, ip og porte. Når disse er succesfuldt initialiseret afvendter TCP serveren at KI connecter. Efter connection modtager TCP serveren datapakken fra KI og gemmer denne i en textfil kaldt ship.txt. Denne fil bruges bruges som en midlertidig sikkerhed for at data fra KI sikkert bliver inført i mySQL databasen.

sequuens diagram

3.2.3 Web-side

For at web siden kan køre kræves der at der på serveren er installeret en web - server. Der er på denne server installeret apache som web - server.
Ved opstart af serveren bliver denne automatisk startet og start siden er **BROS**. Web siden fungere som bruger interfacet for havne terminalen. Web siden er opbygget som et



Figur 3.4. BDD Web-page BROS

dynamisk web page og kodet i php. Dette sikre minmal loading time ved hjælp af ajax. Alle styles på siden er styret af css. Siden har 4 forskellige under sider Home, About Us, Ship Data og Contact. Ved klik på Ship Data vil man blive bedt om at taste sit password som sikre at kun autoriserede personer får adgang til systemet.
Siden der håndtere skibs data starter med at connecte til mySQL databasen om ikke der kan connectes til databasen vil der blive udskrevet en error og siden vil igen forsøge at conencte til databasen. Efter connection til databasen vil den gemte fil fra TCP-serveren blive loadet ind i mySQL databasen og filen vil blive slettet. efter loading vil siden loade alle data i mySQL databasen og vise denne for brugeren. Data der kan vises for brugeren er:

- Skibs ID
- Højre tanks vandniveau
- Venstre tanks vandniveau
- Hældningsniveau
- Forbindelse til KI

Siden checker hvert 5 sekund om der er nu data. Hved ny data vil denne blvie placeret øverst på siden. Når brugeren er færdig med at benytte BROS databasen kan brugeren trykke log of i øverste højre hjørne.

3.3 Metodebeskrivelse

3.3.1 TCP KI

Void msg(string, string, string, string, string);
Håndtere data der skal sendes via tcp.
Void socket();
Opretter socket forbindelse for ctp client.
Void connection();
Kalder ip adresse og portnummer for TCP server. Overføre data.

3.3.2 TCP database

Void socketConnect();
Void msgServer();

SM 4

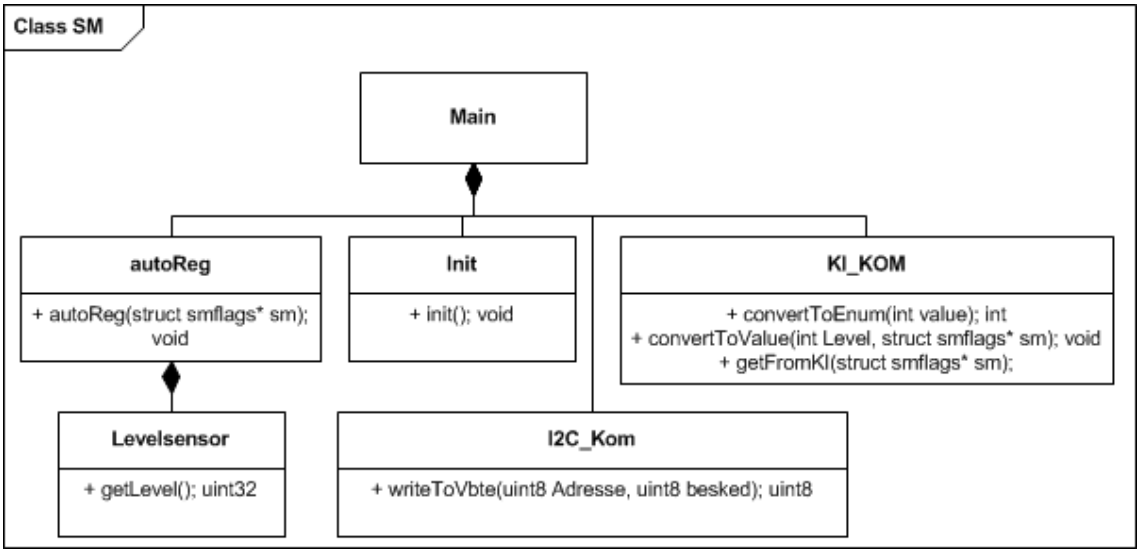
Dette afsnit beskriver designet af styringsmodulet, SM.

4.1 Klassens ansvar

Styringsmodulet har til ansvar at holde styr på levelsensoren og værdierne fra VBTE. Den kommunikere med KI og VBTE med indbyggede API'er fra Cypress PSoC 5 biblioteker.

4.2 Klassediagram

Nedenfor ses klassediagrammet for SM. Bemærk at koden dog er i C men for overblikket er der lavet klassediagram.



Figur 4.1. På figuren ses klassediagrammet for SM

4.3 Funktioner

bla bla

4.4 Variabler

Variabel	Beskrivelse
autoflag	Denne variable er et flag der holder styr på automatisk regulering .
manuflag	Et flag til at holde styr på manuel regulering.
levelVal	En variable med vores level værdi.
VBTE1Niveau og VBTE2Niveau	Holder styr på vandniveauet i ballasttanke i %.
VBTE1Status og VBTE2Status	Holder styr op tilgængelighed for VBTE1 og 2.
vinkelVal	Indeholder værdien for manuel regulering.

Alle variabler er indkapslet i en struct navngivet "smflags".

4.5 Funktionsbeskrivelser

4.5.1 Init

Ansvar

Denne header har til ansvar at sørge for alle komponenter opretter og initieret. void init(void);

Beskrivelse:	Funktionen anvender API'et fra Cypress componenter og står for at initiere og starte vores PSoC hardware. Den sætter også et register tilhørende vores Accelerometer.
Parametre:	ingen
Returværdi:	ingen

4.5.2 Levelsensor

Ansvar

Denne header har til ansvar at hente levelværdien ind fra vores accelerometer. uint32 getLevel(void);

Beskrivelse:	Funktionen anvender API'et fra Cypress componenter og venter på at vores ADC henter convertere det analoge signal. Funktionskald for ADC ses i PSoC databladet.
Parametre:	ingen
Returværdi:	uint32 levelVal

4.5.3 autoReg

Ansvar

Denne header har til ansvar at void autoReg(struct smflags* sm);

Beskrivelse:	ingen
Parametre:	struct smflags* sm
Returværdi:	ingen

4.5.4 I2C_Kom

Ansvar

Denne header har til ansvar at uint8 writeToVbte(uint8 Adresse, uint8 besked);

Beskrivelse:	ingen
Parametre:	uint8 Adresse
	uint8 besked
Returværdi:	uint8 VbteNiveau

4.5.5 KI_KOM

Ansvar

Denne header har til ansvar at .
void convertToEnum(int value);

Beskrivelse:	ingen
Parametre:	int value
Returværdi:	ingen

void convertToValue(int Level, struct smflags* sm);

Beskrivelse:	ingen
Parametre:	int Level,
	struct smflags* sm
Returværdi:	ingen

void getFromKI(struct smflags* sm);

Beskrivelse:	ingen
Parametre:	struct smflags* sm
Returværdi:	ingen

4.6 Eventuelle Sekvensdiagrammer og state machines

Måske kommer de senere?

VBTE 5

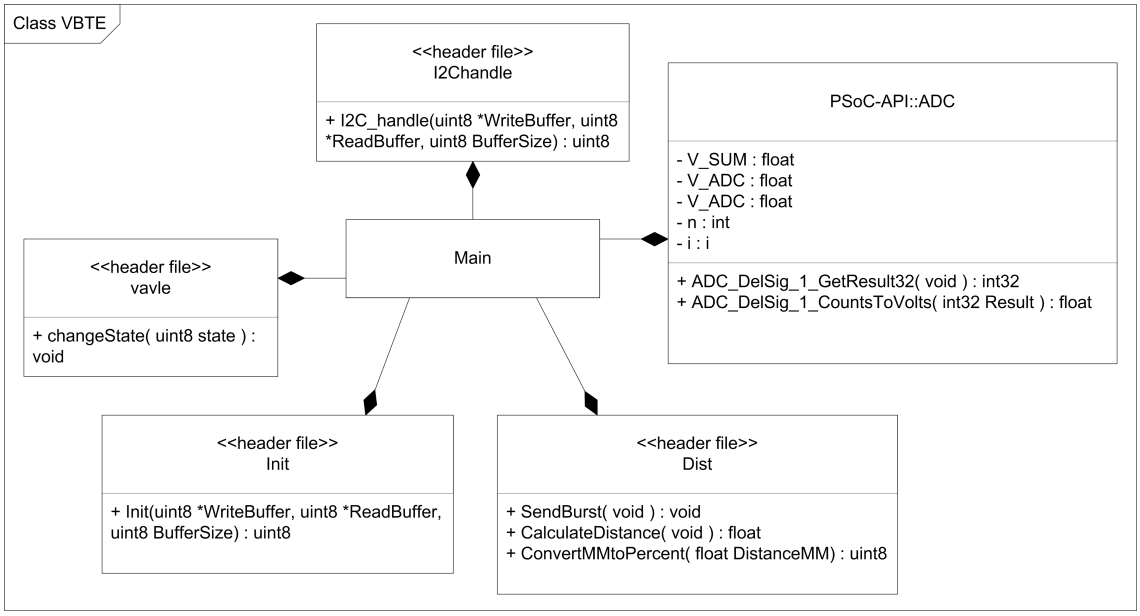
Nedenfor følger design af software til VBTE. Dette er lavet på baggrund af kravspecifikation og systemarkitektur. Bemærk der i dette design dokument blandt andet ikke er beskrevet mixer, pga osv. da deres eneste funktion er "Start();".

5.0.1 Klassens ansvar

Som beskrevet i systemarkitektur står VBTE'en for at måle vandniveauet i ballasttankene samt at lukke vand ind eller ud af ballasttankene. Hertil er der også en kommunikation med SM modulet indeholende instruktioner.

5.0.2 Klassediagram

Nedenfor ses klassediagrammet for VBTE. Bemærk at koden dog er i C men for overblikket er der lavet klassediagram.



Figur 5.1. På figuren ses klassediagrammet for VBTE **Billedet skal opdateres**

5.0.3 Globale variabler

Variabel	Beskrivelse
BurstLengthVal	Denne variabel er anvendt til at håndtere antallet af perioder burstet bliver sendt med.
WaitBurstVar	Bliver brugt til nonblocking delay til SendBurst funktionen.
BurstTimerVal	Holder på Timerens værdi når et burst er sendt.
DistanceTimerVal	Holder værdien på timeren når et burst er modtaget.
CalcDistFlag	Bliver sat når et burst er modtaget så en afstand kan blive beregnet.

5.0.4 Valve

Ansvar

Denne header har til ansvar at styre ventilerne ud fra "state-variablen modtaget fra I2C_handle. Headeren benytter PSoC-API'et til kontrol registre..

Funktionsbeskrivelser

void ChangeState(uint8 state);

Beskrivelse:	Funktionen anvender API'et fra I2C blokken i PSoC miljøet. Med disse tjekker den om der er fyldt nyt i bufferen og aflæse dette. Herfer kalder den funktionen I2C_decode(); til at afkode beskeden fra SM. Herefter klargøres readbufferen til evt. at sende vandniveau tilbage.
Parametre:	uint8* WriteBuffer uint8* ReadBuffer uint8 BufferSize
Returværdi:	uint8 State

5.0.5 Dist

Ansvar

Denne header har til ansvar at sende burst, beregne afstanden samt at omregne afstanden til procent.

Funktionsbeskrivelser

void SendBurst(void);

Beskrivelse:	tis
Parametre:	hund
Returværdi:	henning

5.0.6 Eventuelle Sekvensdiagrammer og state machines

hab hab