Professional Drone, Hybrid Power Pack - Timebox 9

Team 2

5. maj 2019

Deltagere:

Stud. nr: 201602094	Navn: Søren Holm Korsgaard
Stud.nr.: 201607563	Navn: Jacob Gustafsson
Stud.nr.: 20084327	Navn: Simon Rasmussen
Stud.nr.: 201704483	Navn: Thomas Dueholm Jensen

Indhold

1 Strategy and planning (Jacob)

1.1 Strategy

I timebox 9 har vi planlagt at færdiggøre de resterende hardwaresystemer, så disse er testet og klar til implementering. Dette indebærer også, at vi skal have lagt os fast på et interface mellem ensretteren og batteriet. For PID-reguleringens vedkommende, er det planen at der skal simuleres i SimuLink.

1.2 Planning

1.2.1 PID-regulering

I denne timebox gennemgås tuning af PID-koefficienterne og næste timebox vil indeholde implementering af PID-reguleringen.

1.2.2 Ensretter

Ensretteren har gennemgået test i denne timebox. I næste timebox er vi klar til at analysere interface og tilslutte ensretteren til det endelige system.

1.2.3 Spændingsregulator

Spændingsregulatoren er endeligt testet i denne timebox. I timebox 10 skal der for spændingsregulatoren analyseres interface og tilsluttes til det endelige system.

1.2.4 Fuldt system

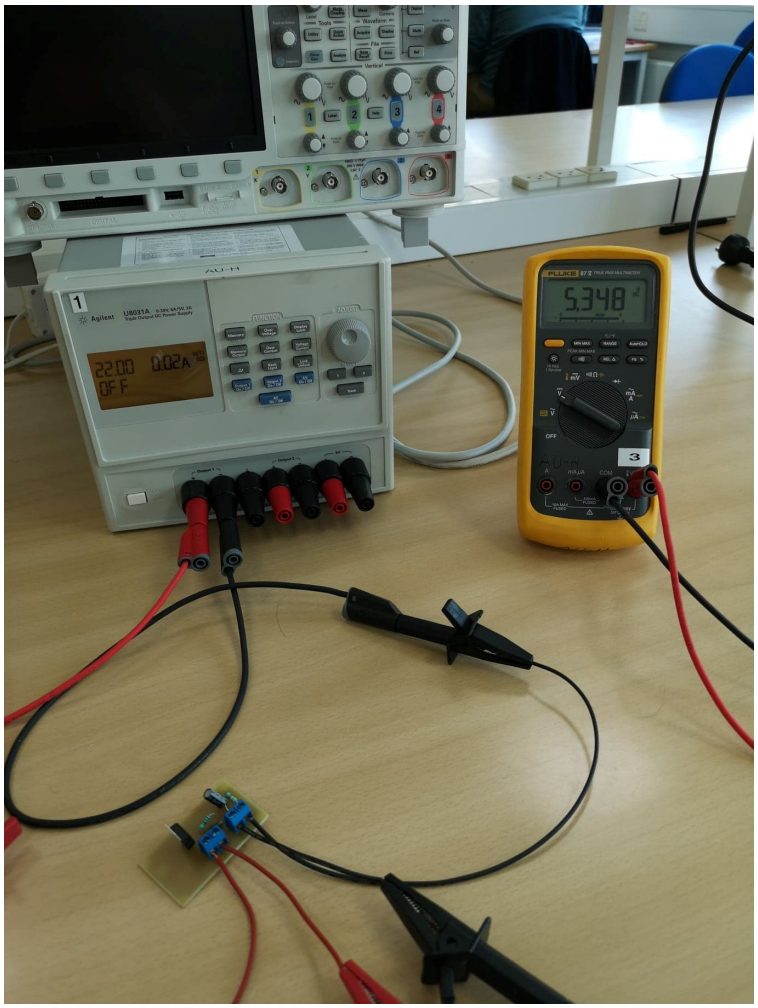
I timebox 10 tester vi det endelige system på hardwarebasis. Tester at alle interface fungerer og at de enkelte subsystemer fungerer med hinanden.

2 Spændingsregulator (Jacob)

I denne timebox er spændingsregulatoren blevet loddet op og testet.

2.1 Testen

Spændingsregulatoren blev meget enkelt testet ved at tilslutte en strømforsyning som input og tilslutte et voltmeter, for at måle udgangsspændingen. Opstillingen kan ses herunder.



Figur 1: Testen

2.2 Resultater og konklusion

Som det fremgår af ovenstående billede, regulerer spændingsregulatoren fra 22 V til 5,3 V. Output spændingen er altså en smule højere end tilsigtet, men ikke mere end det kan fungere. Da setuppet blev bygget på breadboard var outputspændingen på 5,1 V. Forskellen ligger i, at der er en større intern modstand på breadboardet end på PCB-printet, hvilket der ikke er blevet taget højde for.

3 Aflæsning af RPM (Søren)

Implementering af kode til at aflæse rpm, der skal bruges til vores PID-regulering. Vi har taget udgangspunkt i aflæse et tacho signal som kommer fra vores tændspole. Dette signal lave en høj spænding ved hver omdr, og går lav igen.

For at afkode dette signal, bruges der input capture, der er implementeret i TPM periferenheden på kl25. Funktionen input capture, fungere ved at den tælle op til hver 'rising edge' og gemmer denne værdi. Og når vi kender værdien og frekvensen vi tæller op med, kan vi omregne dette til RPM.

Ved formlen: $\frac{60 \text{ sek} \cdot \text{frekvens}}{input\ capture\ v\_rdien} = RPM$.

Selve koden er vist her forneden, hvor vi bruger interrupts, til at opfange overflows og når vi får et rising edge, gemmer den værdi i CNV-registeret.

```
void init_read_rpm() {
        //set clock
        SIM->SCGC6 |= SIM_SCGC6_TPM1_MASK;
        SIM->SCGC5 |= SIM_SCGC5_PORTE_MASK;
        //set clock source to TPM
        SIM->SOPT2 |= (SIM_SOPT2_TPMSRC(1) | SIM_SOPT2_PLLFLLSEL_MASK);
        //load counter to max value 2^16
        TPM1->MOD = 0xffff;
        //set channel to input capture and enable intterupt
        TPM1->CONTROLS[TPM_CH].CnSC = TPM_CnSC_ELSA_MASK |
           TPM_CnSC_CHIE_MASK;
        //set pin to timer TPM
        PORTE->PCR[RPM_PIN] |= (3UL << 8);
        // Enable0 interrupts, with 128 prescaler. so count freq is 48MHz
           /128 = 375 kHz
        TPM1->SC = TPM_SC_CMOD(1) | TPM_SC_PS(7) | TPM_SC_TOIE_MASK;

        NVIC_SetPriority(TPM1_IRQn, 3);
        NVIC_ClearPendingIRQ(TPM1_IRQn);
        NVIC_EnableIRQ(TPM1_IRQn);

}
```

Her initialiseres vores TPM1 modul. Vores frekvens der bruges til at tælle op med, bliver nedskaleret, til 375 kHz. Dvs. imellem hver overflows går der 5,75 Hz, hvilket passer med vores måle område (17 - 167 Hz) vi arbejder i. Ved testen, simuleret vi et tastesignal med frekvensen 10 Hz, og kunne decode det til 600 RPM -+ 20. Da der var lidt støj på signalet.

```

#include "rpm_detect.h"
#define TPM_CH 0    // TPM channel for PTE 20
#define RPM_PIN 20  //PTE 20 is used to detect rpm

enum {
        wait, set_data
} state;
volatile int count;
void TPM1_IRQHandler() {
        static uint32_t overflows = 0;
        static uint32_t prev_count = 0;
        uint32_t timer_val;

        if (TPM1->STATUS & TPM_STATUS_TOF_MASK) {
                overflows++;
        }

        if (TPM1->STATUS & TPM_STATUS_CH0F_MASK) {
                timer_val = TPM1->CONTROLS[TPM_CH].CnV;
                timer_val |= (overflows << 16);
                count = timer_val - prev_count;
                prev_count = timer_val;
                state = set_data;
```

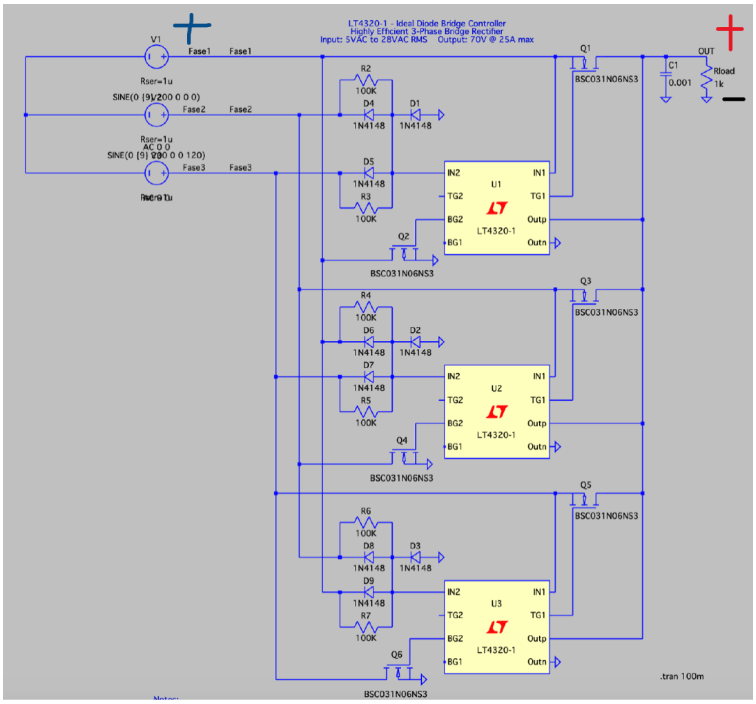
```
25          }
26          // reset all flags
27          TPM1->STATUS |= TPM_STATUS_TOF_MASK | TPM_STATUS_CHOF_MASK
28                          | TPM_STATUS_CH1F_MASK;
29  }
```

Her ses vores handler, hvori RPM bliver beregnet ud fra værdierne fra TPM1 perifer enhed. Og videre i koden kan man detektere om der er kommet ny RPM-data, ved bruge enum 'state'. Da det tager minimum en omgang på krumtappen, før den har beregnet en ny værdi.

4 Funktionalitetstest af aktiv ensretter. (Thomas)

For at teste funktionalitet af det færdig-loddede PCB print til den aktive ensretter, besluttede vi os for at lave en måling af V_{in}, fase 1, vs. V_{out}. Ved at sammenligne niveauet på kredsløbets udgangsspænding med niveauet på indgangen, vil det være muligt at se det samlede spændingstab gennem kredsløbet. Dette spændingstab skal være under 0.7 volt, for at kredsløbet er en forbedring i forhold til en almindelig ensretter bestående af en diodebro. Målingen foretages med et tilgængeligt oscilloskop fra El-lab i AU Herning, hvor data fra målingen eksporteres til en .csv fil. Med .csv filen er muligt at behandle signalerne i programmet, Matlab.

På nedenstående figur ses et diagram over kredsløbet til den aktive ensretter for at vise, hvor måleproberne er placeret under målingen.



Figur 2: Blå: Probe 1 (Fase 1). Rød: Probe 2 (V_{out})

Teststanden fra timebox 7 benyttes til at generere en 3 faset spænding, som sendes ind i den aktive ensretter. Der er benyttet en loadmodstand på 1 kΩ, ligesom der er monteret en kondensator på 1000 μF parallelt med loadmodstanden for at udglatte outputspændingen.

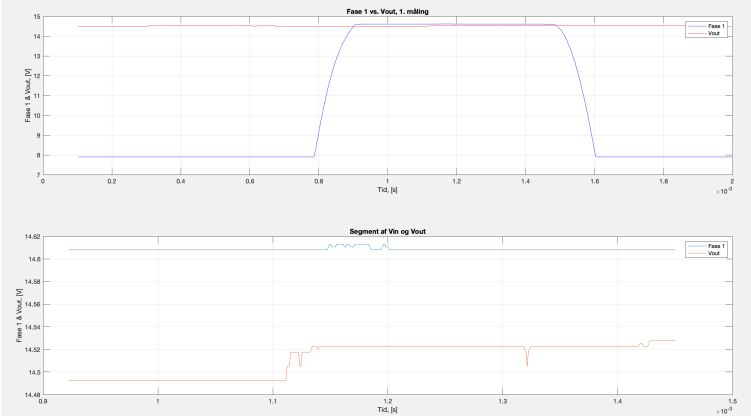
Der blev foretaget 2 målingen. Én hvor generatoren producerede ca. 14.5 V (peak), og én hvor der produceres ca. 10 V (peak). Efter den fysiske måling blev data importeret i Matlab, hvor signalerne blev midlet med et medianfilter. Herefter plottes signalerne for at udfinde det segment, hvor fasen er større end outputtet.

Nedenstående figurer (3-5) viser Matlab scriptet, der blev benyttet til databehandlingen, samt plots af de to målinger.

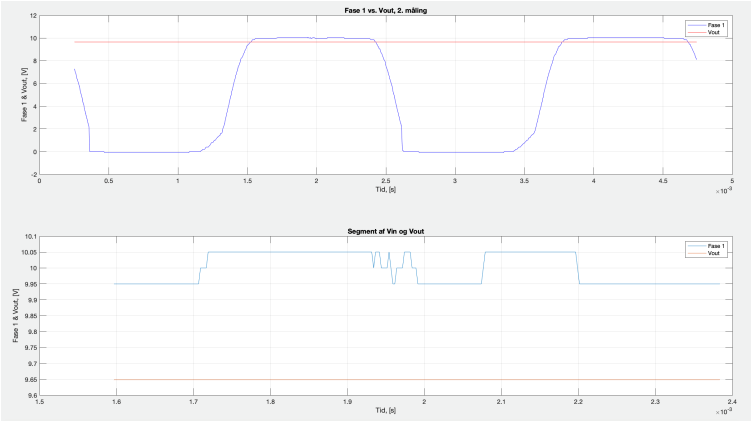
```
% Inputspænding fra .csv fil (Volt [V])
vin = data1.deltavoutfase1.Volt1;
% Outputspænding fra .csv fil (Volt [V])
vout = data1.deltavoutfase1.Volt;
% Antal samples i signalerne
N1 = length(vin);
n1 = 0:N1-1;
% Varighed af signalerne
Tdur1 = t1(2000)-t1(1);
% Samplevarighed
Ts1 = Tdur1/N1;
% Samplingfrekvens
fs1 = 1/Ts1;
% Midling af input og output
vin_mid = medfilt1(vin,200);
vout_mid = medfilt1(vout,200);
% Max værdi for vin og vout
vin_mid_max = max(vin_mid);
vout_mid_max = max(vout_mid);

% Plot af hele sigalet
figure
subplot(211)
```

```
figure
subplot(211)
plot(n1*Ts1,vin_mid,'b'), hold on
plot(n1*Ts1,vout_mid,'r')
xlabel('Tid, [s]'), ylabel('Fase 1 & Vout, [V]')
legend('Fase 1','Vout'), grid
title('Fase 1 vs. Vout, 1. måling')
% Udvælgelse af segmenter, hvor Fase 1 > Vout
vin_seg = vin_mid(924:1452);
vout_seg = vout_mid(924:1452);
% Plot af segmenter
subplot(212)
plot(n1(924:1452)*Ts1,vin_seg), hold on
plot(n1(924:1452)*Ts1,vout_seg )
xlabel('Tid, [s]'), ylabel('Fase 1 & Vout, [V]')
legend('Fase 1','Vout'), grid
title('Segment af Vin og Vout')
% Udregning af spændingstabet, (Vin - Vout)
v_delta1 = mean(vin_seg)-mean(vout_seg);
```

Figur 3: Matlab script.



Figur 4: Første måling (V_{in} = 14.613 [V], V_{out} = 14.528 [V])



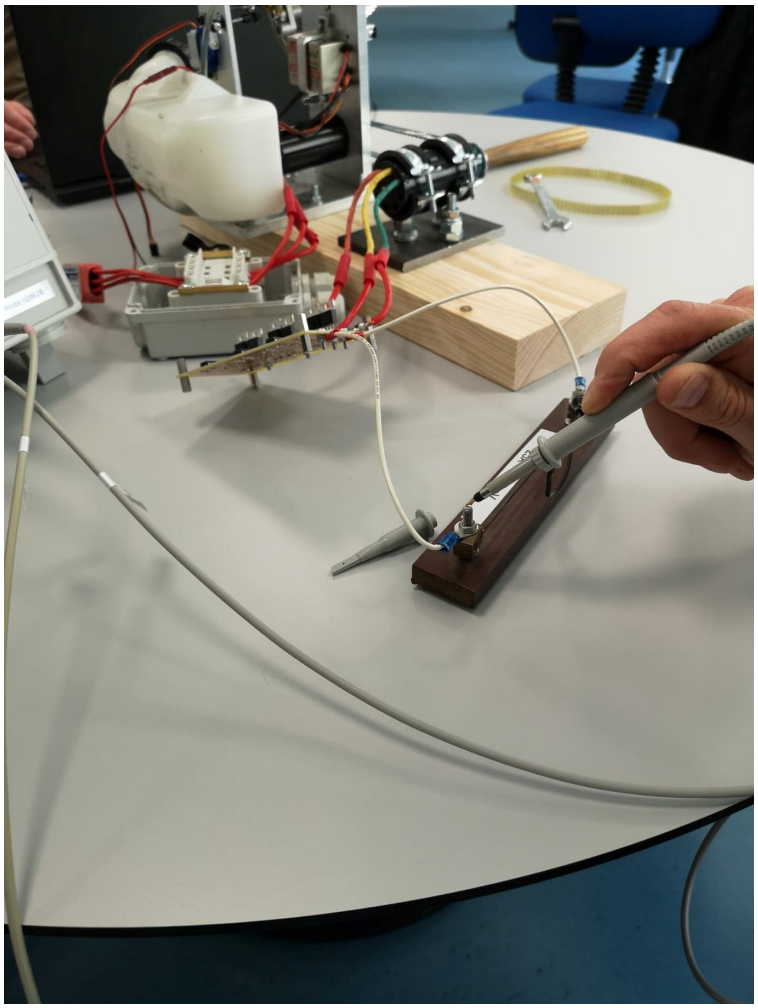
Figur 5: Anden måling (V_{in} = 10.050 [V], V_{out} = 9.648 [V])

Som det ses ud fra begge plots i figur 4 og 5, er der et spændingstab på udgangen af ensretteren i forhold til indgangen. Spændingstabene for målingerne er i Matlab udregnet til:

1. måling: Vdelta1 = 0.0967 volt
2. måling: Vdelta2 = 0.3497 volt

4.1 Større load

For at teste om ensretter kredsløbet kan leve op til de store strømme (80 A peak), som det færdige HPP system teoretisk set kan komme til at trække, er der anskaffet en effektmodstand på ca. 12 mΩ. Denne effektmodstand kan håndtere op til 25 A. Kredsløbet kan ikke testes fuldt ud med denne modstand, men det vil give en indikation af, hvordan kredsløbet reagerer ved en større belastning.



Figur 6: Foto af testopstilling med load modstand på ca. 12 mΩ.

Effektmodstanden er designet således, at der løber 25 A gennem den, hvis der er et spændingsfald på 300 mV over modstanden.

Teststanden fra timebox 7 blev igen benyttet til at genere inputtet til ensretteren.

1. måling: Generatoren blev drevet således, der løb ca. 25 A gennem load modstanden.

Efter få sekunder kunne vi konstatere, at de 6 MOSFETS på printet blev meget varme, hvorfor testen blev stoppet.

De 6 MOSFETS på printet er af typen:

- Fairchild, FDP61N20, N-channel MOSFET[1]
- Drain-current: 61 A
- Drain-source spænding: 200 V
- RDS(ON) = 0.041 Ω

[1] http://213.114.131.21/_pdf/FD/FDP61N20.pdf

- VGS = 10 V
- Thermal Resistance, J-A, $R_{\theta iA} = 62.5°C/W$

Ud fra datasheet'et kan det ses, at den interne modstand i disse MOSFETS er på 41 mΩ. Det er denne modstand, der forårsager spændingstabet fra faserne til outputtet.

Hvis det antages, at der i værste fald er et spændingsfald på de tidligere beregnede 0.3497 V fra source til drain på hver af de 6 MOSFETS, vil dette resultere i et effekttab på

$$P = \frac{0.3497^2 V}{0.041\Omega} = 2.87W \tag{1}$$

Og dette vil aflede en stigning i temperaturen i de 6 MOSFETS på:

$$T = P \cdot R_{\theta jA} = 2.87W \cdot 62.5\frac{°C}{W} = 179.2°C \tag{2}$$

Ud fra datasheet'et til FDP61N20 ses det, at disse MOSFETS maksimalt kan håndtere 150 $°C$.

4.2 Konklusion

Ud fra funktionalitettesten af den aktive ensretter, hvor inputspænding vs. outputspænding plottes og spændingsfaldet beregnes i Matlab, kan det konkluderes, at kredsløbet lever op til kravet om ikke at have et spændingsfald på udgangen på mere end 0.7 volt i forhold til indgangen. Det kan derfor konstateres, at den er væsentlig mere effektiv end en ensretter bestående af en diodebro.

Spændingstabet ved anden måling af funktionalitetstesten er dog relativ høj i forhold den første måling (0.3497 V vs. 0.0967 V), hvilket forekommer besynderligt. Ideelt set burde spændingstabet være det samme ved samtlige inputspændinger, så forskellen kan muligvis forklares ud fra en uregelmæssighed på én eller flere af de tre faser fra generatoren, der producerer inputsignalet til kredsløbet.

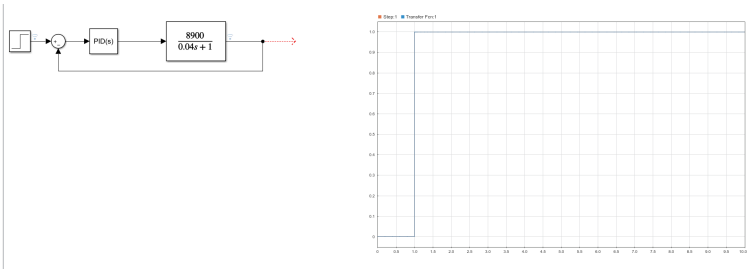
Derudover kan der konkluderes, at valget af MOSFET til realiseringen af ensretter-printet skal ændres. De nuværende MOSFETS kan ikke leve op til varmeafledningen, hvorfor der skal vælges andre MOSFETs med en lavere intern modstand og evt. påmonteres køleplader til disse. Dette vil blive testet og redegjort for i næste timebox. Den endelige test af max belastning forventes ligeledes gennemført i timebox 10.

5 Tuning af PID-koefficienter

I sidste timebox fandtes overføringsfunktionen

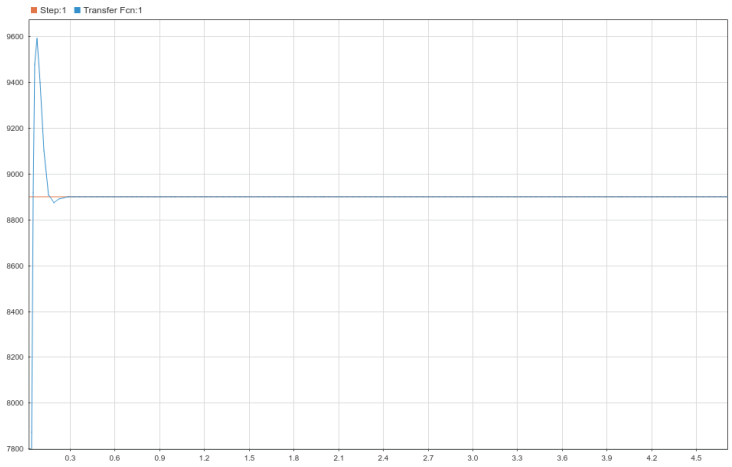
$$G(s) = \frac{8900}{0,04s + 1} \tag{3}$$

I Simulink oprettes følgende:



Figur 7: Simulink - diagram

Herefter laves autotuning i PID-modulet. Der findes koefficienter svarende til $P = 0,00015$, $I = 0,00751$, $D = -2.1832$ og følgende respons:



Figur 8: Simulink - diagram 2

I næste timebox implementeres og justeres PID-reguleringen.

6 Deployment (Alle)

Hermed godkender kunderne, Morten Oppbrud Jakobsen og Jan Møller Nielsen, ovenstående i timebox 9.
Mandag den 6/5-2019

Morten Opprud Jakobsen	Jan Møller Nielsen