



AARHUS UNIVERSITET

Aarhus University School of Engineering

SW2GFV - Experiment Motor Control

Gruppe 69

Jakob Damgård Dall

Studienummer: 201805013 AU-id: au616329

Mathias Martin Rahbek Markussen

Studienummer: 202107385 AU-id: au694672

Thomas Kaae

Studienummer: 202100350 AU-id: au698066

DATO

16. februar 2022

Indhold

1	Indledning	2
2	DC-motor	3
2.1	Formål	3
2.2	Forsøg 1: PWM styring af DC-motor	3
2.2.1	Resultater	7
2.2.2	Konklusion	7
2.3	Forsøg 2: DC-motor rotations retning	8
2.3.1	Konklusion	10
3	Stepper Motor	11
3.1	Formål	11
3.2	Forsøg 3: Stepper motor styring	11
3.2.1	Resultater	15
3.2.2	Konklusion	15
4	Konklusion	16
5	Litteratur	17

1 | Indledning

I denne journal bliver forskellen på to forskellige motorer, samt kontrollen af dem undersøgt. Den ene en DC-motor, og den anden en unipolær stepper-motor.
DC-motorens hastighed skal kontrolleres med et PWM-signal, og dens rotationsretning igennem en H-bro. Alle digitale signaler bliver sendt fra en PSoC.
Stepper-motoren skal ligeledes kontrolleres af PSoC'en, dog gennem fire MOSFET transistorer. Der ønskes en forklaring på forskellen mellem Full-step, Half-step og Wave-drive.

2 | DC-motor

2.1 Formål

Formålet med øvelserne, er at kontrollere en DC-motors rotationshastighed og rotationsretning. Til øvelsen gøres der brug af en DC-motor af mærket "IGARASHI-MOTORS".

DC-motorens hastighed bliver styret med et PWM-signal fra PSoC'en, der løber igennem en IRLZ24 MOSFET transistor. Da PSoC'en ikke kan levere nok strøm, bruges der i stedet en ekstern strømforsyning, der kan levere 9V og en strøm på 2,5A, hvilket er rigeligt til dette formål. Retningen bliver kontrolleret af en L298 H-Bridge. Alle datablade for de nævnte komponenter kan ses i vedhæftede bilag.

2.2 Forsøg 1: PWM styring af DC-motor

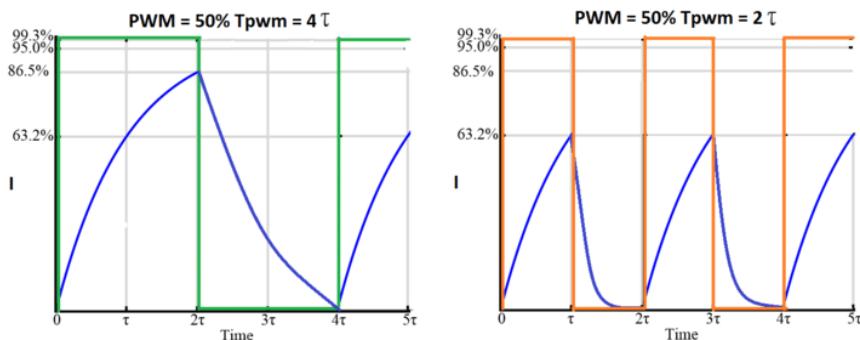
I dette forsøg ønskes det at kontrollere en DC-motors hastighed, gennem en UART, der er programmeret på en PSoC.

PWM-signal:

Motorens hastighed bliver kontrolleret med et PWM-signal.

Et PWM-signal fungerer ved at den gennemsnitlige spænding justeres, idet der tændes og slukkes for den tilførte belastning på motoren. Det har fordelen, at strømtabet er utrolig lavt, idet belastningen på motoren enten er tændt eller slukket, hvilket også resulterer i et mindsket effektabet på transistoren.

Kontrolling ved brug af PWM, er oplagt i forbindelse med en DC-motor, da motorens inertimoment, gør at de er forholdsvis sløve til at reagere på, at strømmen til dem tændes og slukkes, såfremt frekvensen, som det gøres ved, er hurtig nok.



Figur 2.1: PWM med 50% duty cycle, ved to forskellige frekvenser [1]

Frekvens:

For at opnå en stabil drift af motoren, gøres der derfor brug af en frekvens på 50KHz. Dette er grundet en frekvens på under 20KHz, kan skabe problematikker, hvor motoren ikke nødvendigvis vil opnå en flydende rotation, da perioderne, hvor motoren ikke tilføres belastning, er for lange, hvorved motorens rotationshastighed, vil kunne nå at sænkes mærkbart, før næste puls af belastning påføres motoren.

Derudover vil en teoretisk frekvens på over 4,2MHz være for hurtig for MOSFET'en at reagere på. Den reelle maksimum frekvens er langt lavere, da der ved 4,2MHz nærmest ikke vil blive overført nogen energi.

Se figur 2.2 for beregning af MOSFET'ens teoretiske maksimum.

Spolen i motoren skaber også sine egne begrænsninger, langt tidligere end MOSFET'en, da en elektromotorisk kraft yder modstand mod strømmen, der flyder gennem den. Dette har den betydning, at stigningen ikke sker øjeblikkeligt, når der tændes for strømmen. Dette kan ses på kurven i 2.1, hvor den blå linje udgør strømmen, og den grønne og orange linje udgør de to forskellige frekvenser. Her kan det også ses, at højere frekvenser, giver strømmen korte tid til at stige.

$t_{d(on)}$	Turn-On Delay Time	-	-	40	ns
t_r	Rise Time	-	-	260	ns
$t_{d(off)}$	Turn-Off Delay Time	-	-	200	ns

Datablad for: IRLZ24

$$td_{on} := 40 \text{ ns} \quad td_{off} := 200 \text{ ns}$$

$$t := td_{on} + td_{off} = 240 \text{ ns}$$

$$f_{teoretisk_maks} := \frac{1}{t} = 4.167 \text{ MHz}$$

Figur 2.2: Udregning af teoretisk maksimal frekvens for IRLZ24

Komponenter:

Transistoren, der bruges til øvelsen, er en N-channel IRLZ24 MOSFET transistor. Transistoren kan trække op til 14A, og ved brug af PWM-signalen åbner og lukker den for tilførslen af spændingen til motoren, og justerer herved motorens hastighed.

MOSFET'en har en lav indre modstand, hvilket gør at den er ideel til at agere "switch" i forbindelse med PWM-signalen. Se figur 2.3.

$R_{DS(on)}$	Static Drain-to-Source On-Resistance	—	11	13.5	$\text{m}\Omega$	$V_{GS} = 10V, I_D = 31A$ ③
		—	—	20	$\text{m}\Omega$	$V_{GS} = 5.0V, I_D = 30A$ ③
		—	—	22.5	$\text{m}\Omega$	$V_{GS} = 4.5V, I_D = 15A$ ③

Figur 2.3: Indre modstand IRLZ24 MOSFET

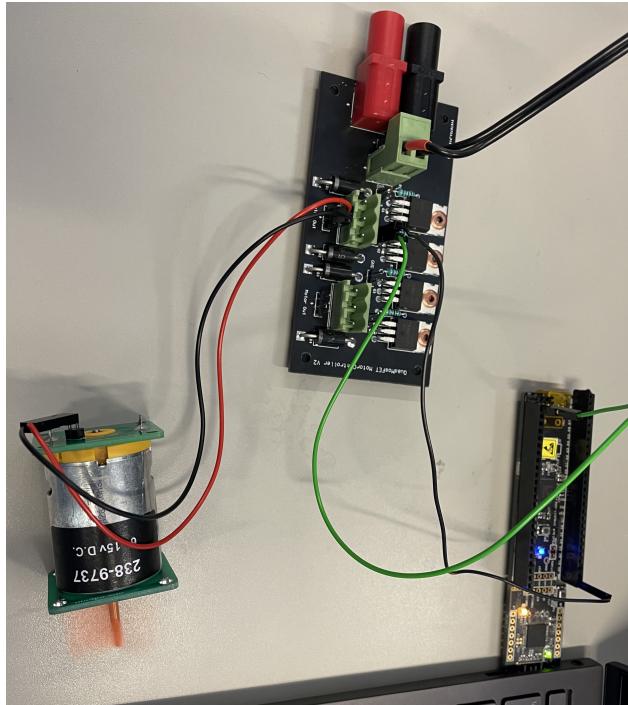
Nedenstående er et eksempel på den kode, der blev brugt til at sætte rotationshastigheden på DC-motoren. Realiseringen af forsøget på fumlebræt ses på figur 2.4, og opstillingen i MultiSim ses på figur 2.5.

```
void decreaseSpeed()
{
    UART_1_PutString("Decreasing speed\r\n");

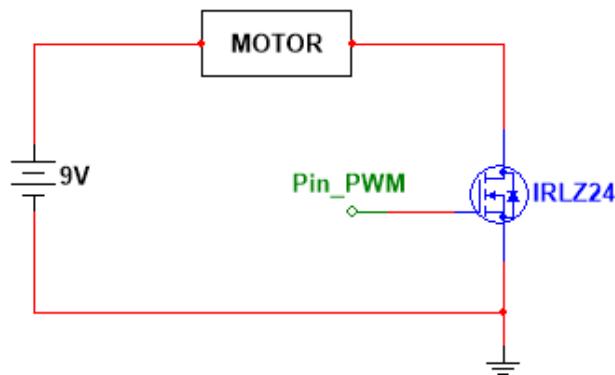
    if (CURR_PWM > MIN_PWM)
    {
        CURR_PWM -= 10;
        PWM_1_WriteCompare(CURR_PWM);
    }
}

void increaseSpeed()
{
    UART_1_PutString("Increasing speed\r\n");

    if (CURR_PWM < MAX_PWM)
    {
        CURR_PWM += 10;
        PWM_1_WriteCompare(CURR_PWM);
    }
}
```



Figur 2.4: Realisering af forsøg 1 med DC-motor



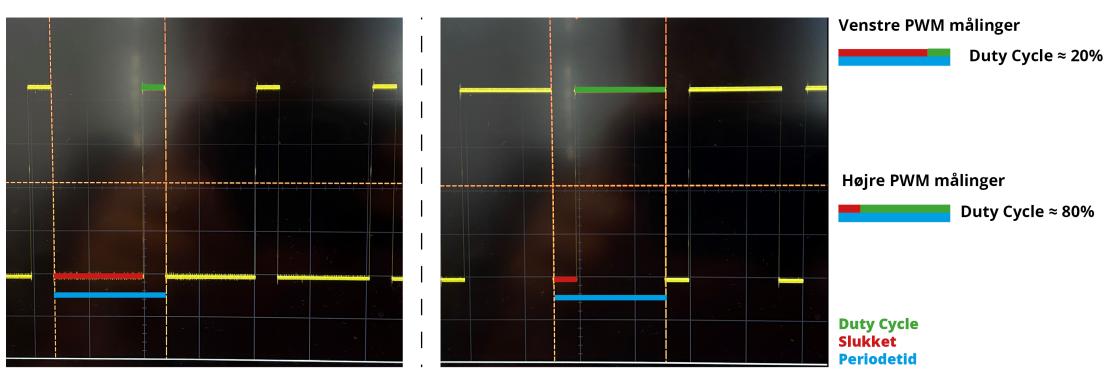
Figur 2.5: DC-motor kredsløb i MultiSim, hvor den digitale udgang kommer fra PSoCen

2.2.1 Resultater

Motoren har reageret som forventet, hvor der auditivt og visuelt er bekræftet acceleration og deceleration, ved kontrollering fra PSoC'en.

Teorien ift. for lav frekvens, er ligeledes auditivt og visuelt bekræftet, hvor motoren pulserer ved forsøg med frekvens på 200Hz.

Ved at trække ledninger fra PCB'et ned på et fumlebræt, er der blevet sat et oscilloskop til kredsløbet, som har givet følgende målinger på figur 2.6.



Figur 2.6: Duty Cycle måling ved circa 20% og 80%

2.2.2 Konklusion

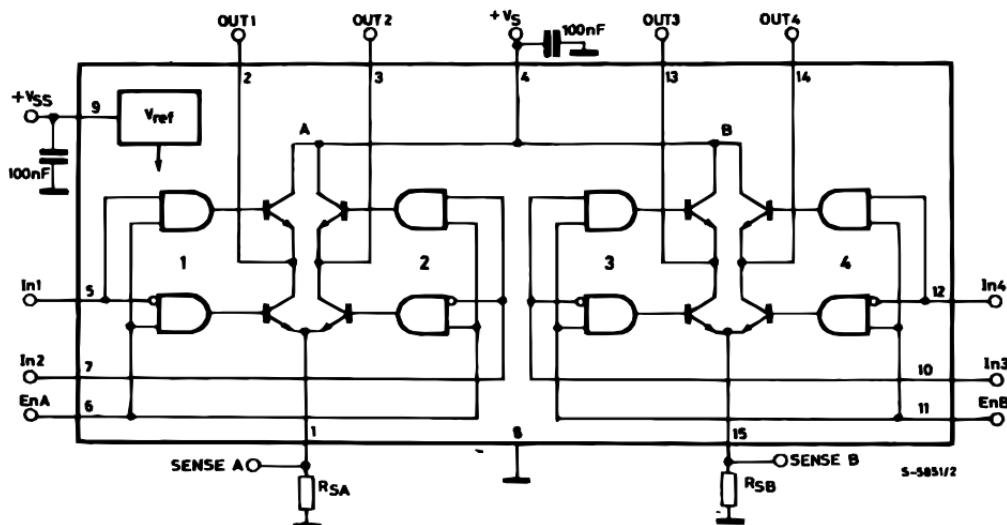
Det ønskedes at kunne styre DC-motoren gennem et MOSFET driver kredsløb, hvilket succesfuldt er blevet gjort. Brugeren har med sit tastatur kunne styre hastigheden på motoren, og med software blev det muligt at sætte en begrænsning på den maksimale og minimale hastighed.

2.3 Forsøg 2: DC-motor rotations retning

I dette forsøg ønskes det at få DC-motorens rotationsretning til at skifte, og samtidig kontrollere dens hastighed, som set i forsøg 1.

H-bro:

H-broen, der bruges til øvelsen, er en L298 DUAL FULL-BRIDGE DRIVER. H-broen bruges til at styre DC-motorens rotationsretning. Måden hvorpå en H-bro fungerer, er ved at skifte polariteten for en given spænding. Skift af polariteten, ændrer dermed retningen som DC-motoren roterer, da belastningen nu har skiftet til den modsatte retning.



Figur 2.7: Block diagram af H-Bro L298. Se bilag

Hastigheden på motoren reguleres på samme måde som før, hvor den modtager et PWM-signal fra PSoC'en der nu sendes hen til EnA. For at ændre på retningen bruges signalerne In1 og In2, der bliver skrevet til fra PSoC'en. Som set på figur 2.7 er det muligt at kontrollere to motorer, men i dette forsøg bliver der kun gjort brug af den ene.

Nedenstående ses et udsnit af koden, der kontrollerer motorens rotations retning.

```

void driveForwards()
{
    UART_1_PutString("Set direction: forwards\r\n");
    Pin1_Write(1);
    Pin2_Write(0);
}

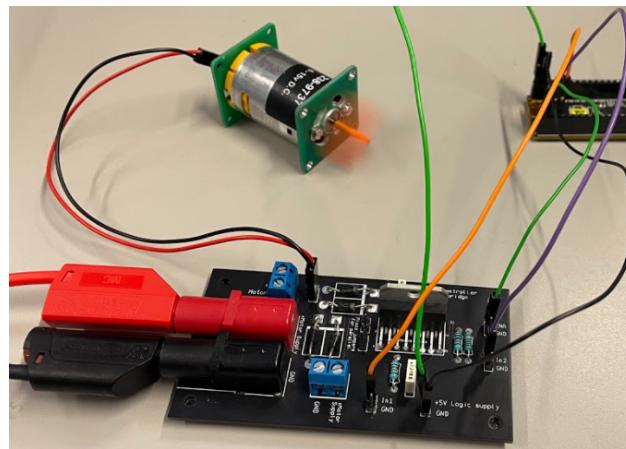
void driveBackwards()
{
    UART_1_PutString("Set direction: backwards\r\n");
    Pin1_Write(0);
    Pin2_Write(1);
}

```

De respektive funktioner bliver kaldt via UARTEn, og sætter de enkelte bit højt og lavt

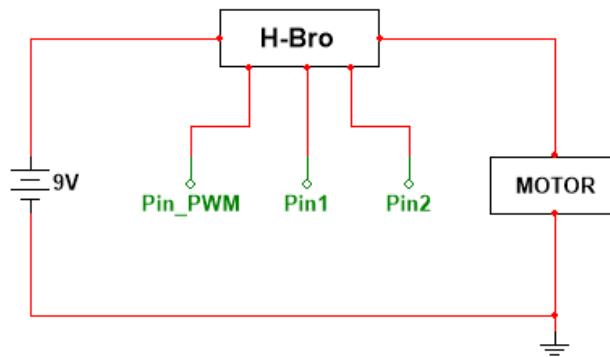
Opsætningen af forsøget ses på figur 2.8, og et diagram fra multisim på figur 2.9.

Opsætning af forsøg:



Figur 2.8: Realisering af forsøg 2 med DC-motor

9



Figur 2.9: DC-motor kredsløb med H-bro i MultiSim, hvor de digitale udgange kommer fra PSoC'en

2.3.1 Konklusion

Motoren reagerede som forventet, og det var muligt at ændre retning, starte og stoppe motoren, og justere dens rotationshastighed.

Dette kunne bekræftes visuelt og auditivt.

3 | Stepper Motor

3.1 Formål

Formålet med dette forsøg er at styre en SP2575M0206-A stepper-motor.

Motoren styres af et MOSFET driver kredsløb, hvor der gøres brug af et udleveret PCB.

Kredsløbet gør brug af fire IRLZ24 MOSFETs, hvor der ved DC-motor forsøget, kun blev gjort brug af en. Hver en MOSFET bliver kontrolleret (åbnet og lukket) af output pins fra PSoC'en, der samtidigt laver et fælles stel på PCB'en.

I forsøget skal der anvendes både half-step, full-step og Wave-drive, hvor hastigheden og retningen ændres.

3.2 Forsøg 3: Stepper motor styring

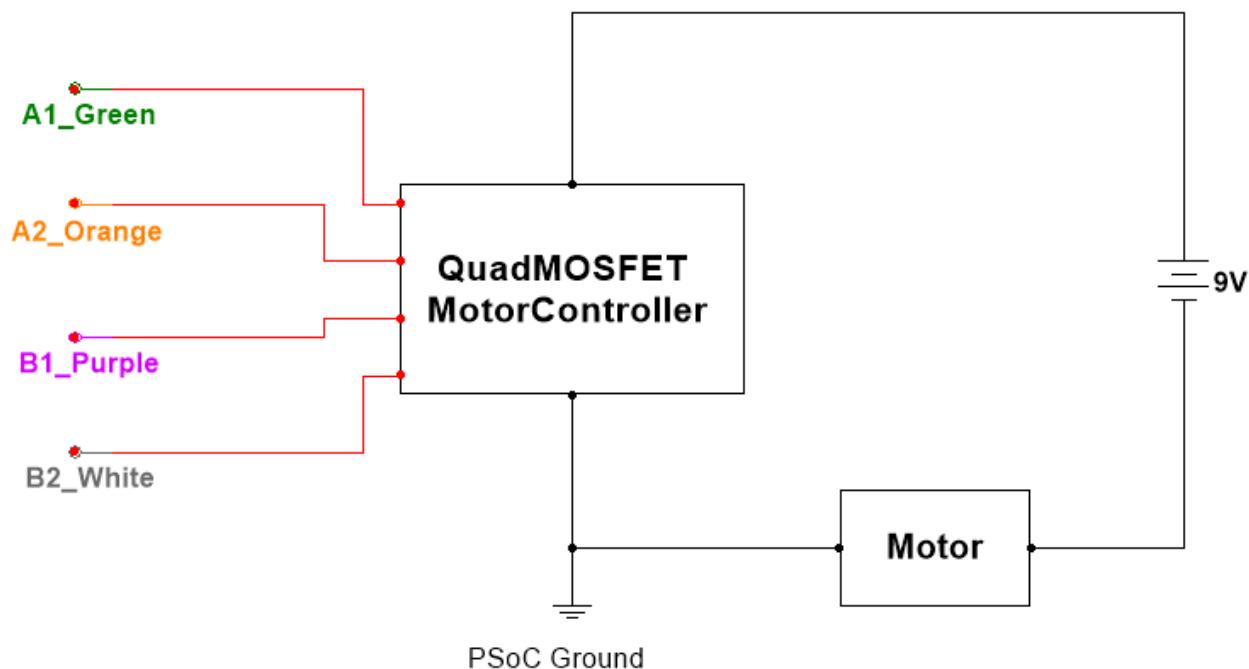
For at skifte tilstand for step-motoren, bruges PSoC'ens UART. Dette gør det muligt at sende kommandoer ved brug af forskellige knapper fra tastaturet.

Programmet gør brug af flere state machines, der hjælper med at håndtere de forskellige input. Udsnit af koden ses nedenunder, hvor et interrupt fra UART'en kalder funktionen handleByteReceived().

```
void handleByteReceived(uint8_t byteReceived)
{
    switch(byteReceived)
    {
        case '1' :
        {
            driveForwards();
        }
        break;

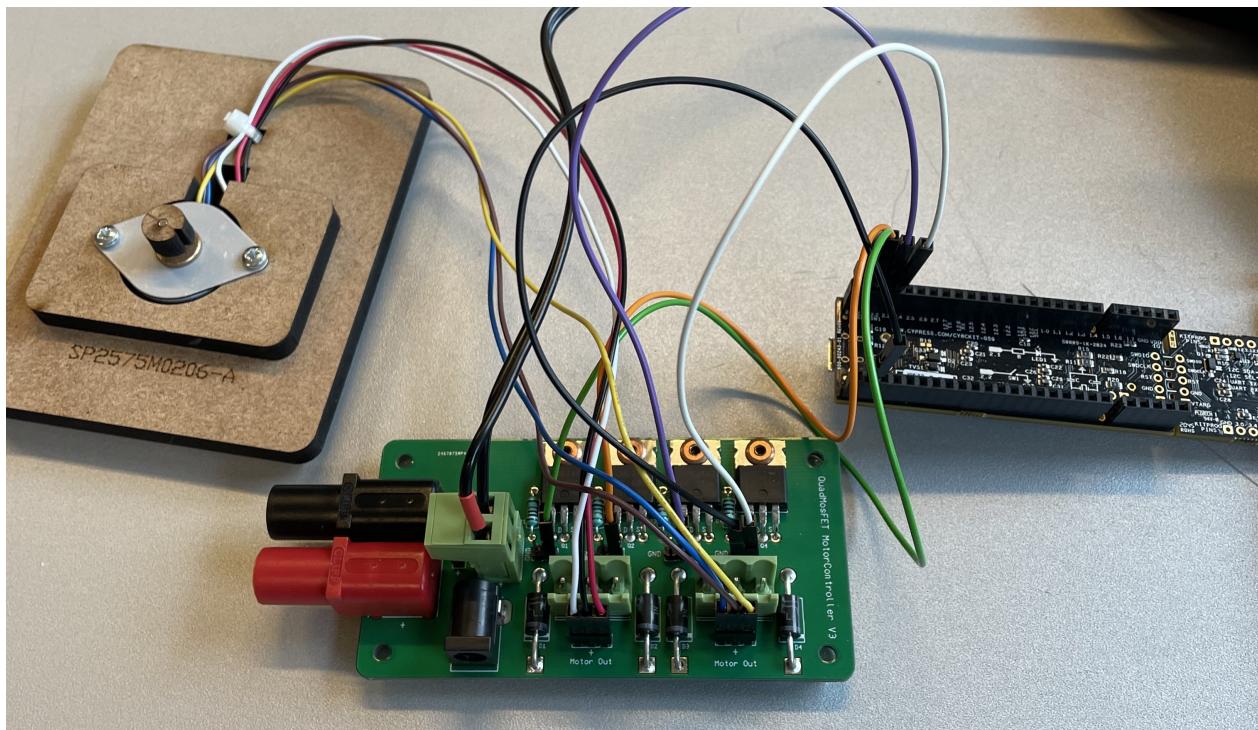
        default :
        {
            // nothing
        }
        break;
    }
}
```

En opstilling i MultiSim af kredsløbet ses på figur 3.1, hvor det realiserede kredsløb ses på figur 3.2.



Figur 3.1: Stepper-motor kredsløb i MultiSim, hvor de digitale udgangene kommer fra PSoC'en

12



Figur 3.2: Stepper-motor kredsløb realiseret

For at kontrollere stepper-motoren tændes og slukkes de individuelle gates i MOSFET'ene, som vender polariteten på magneterne inde i motoren, og derved skaber rotation. I tabel 3.1 ses full-step drive. I MultiSim diagrammet (Figur 3.1), svarer signalnavnene til følgende ift. full-step sekvensen:



Step	A	A/	B	B/
1	+	-	+	-
2	-	+	+	-
3	-	+	-	+
4	+	-	-	+

Tabel 3.1: Full-step motor sekvens

Step	A	A/	B	B/
1	+	-	-	-
2	-	-	+	-
3	-	+	-	-
4	-	-	-	+

Tabel 3.2: Wave-drive motor sekvens

Step	A	A/	B	B/
1	+	-	+	-
2	-	-	+	-
3	-	+	+	-
4	-	+	-	-
5	-	+	-	+
6	-	-	-	+
7	+	-	-	+
8	+	-	+	-

Tabel 3.3: Half-step motor sekvens

Som der ses i tabel 3.1, er der ved full-step tændt to magneter ad gangen. Idet der er to magneter tændt hele tiden, har den en stor trækraft.

I tabel 3.2 er der gjort brug af Wave-drive, hvor der modsat full-step kun er en magnet tændt ad gangen. Den har derfor ikke samme kraft, som en stepper-motor i full-step har.

Sidst, i tabel 3.3, ses der sekvensen for en rotation i half-step. Half-step er en kombination af både full-step og Wave-drive, da den skiftevist tænder og slukker en, og herefter to magneter. Ved denne tilstand er der derfor en større præcision ift. de to førhen nævnte. Det kan dog antages, at maksimalhastigheden i denne forbindelse må blive lavere, da der nu skal udføres flere steps end tidligere.

I opgaveformuleringen efterspørges der et program, der får stepper-motoren til at tage en enkelt omdrejning om sin egen akse. For at udregne antallet af skridt, ses der i motorens datasheet (Se bilag), at den har en step angle på 7.5.

En rotation er 360 grader, hvilket giver en gentagelse af sekvenserne for full-step og Wave-drive på $\frac{360}{7.5} = 48$ ($\frac{\text{Rotation}}{\text{Step-angle}}$).

En rotation i full-step består af 4 sekvenser, og antallet af gentagelser bliver derfor $\frac{48}{4} = 12$. Et eksempel på koden ses nedenunder.

```

void driveForwardsFull() {
    A1_Green_Write(1);
    A2_Orange_Write(0);
    B1_Purple_Write(1);
    B2_White_Write(0);
    CyDelay(speedDelay);

    A1_Green_Write(0);
    A2_Orange_Write(1);
    B1_Purple_Write(1);
    B2_White_Write(0);
    CyDelay(speedDelay);

    A1_Green_Write(0);
    A2_Orange_Write(1);
    B1_Purple_Write(0);
    B2_White_Write(1);
    CyDelay(speedDelay);
}

void runOnceForward() {
    for (uint32_t i = 0; i < 12; i++){
        switch(state) {

            // Full step
            case 1 :
            {
                driveForwardsFull();
            }
            break;

            default :
            {
                //Nothing
            }
            break;
        }
    }
    direction = 0; // Stop
}

```

Laves der en rotation for half-step er det dog anerledes, da den består af en kombination af både full-step og Wave-drive. Det giver en gentagelse, der er dobbelt så stor som de andre to.

3.2.1 Resultater

Alle resultater har været visuel bekræftelse ift. forsøget formål, og motoren opførte sig som forventet.

3.2.2 Konklusion

Det har været muligt at starte og stoppe motoren, og samtidigt er motorens hastighed og retning blevet kontrolleret. Alt kontrol af motoren foregik ved brug af en UART, der visuelt opstillede motorens funktioner, og skabte forbindelse mellem brugerens input og motorens software.

Både half-step, full-step og Wave-drive blev succesfuldt implementeret, og det blev i den forbindelse konkludered, at da full-step gør brug af to stators på en gang, produceres der en større trækraft end ved half-step og Wave-drive.

Sammenlignes step motoren med DC-motoren, er der nogle klare forskelle.

Med step-motoren er det muligt at dreje hen til meget præcise punkter. Den stopper, starter og ændrer hurtigt retning, og den beholder sit drejningsmoment så længe den får tilført strøm, også selvom motoren ikke bevæger sig. Den kan dog miste sit kendskab til sin position, da den naturligt ikke modtager feedback, og dens maksimale hastighed kan ikke måle sig med DC-motoren, som er bedst til høje hastigheder.

4 | Konklusion

Alt i alt har det været succesfulde forsøg.

Forsøg 1, 2 og 3 er alle blevet gennemført, og de forventede resultater har også været dem, der er blevet observeret.

Forsøgene har givet en klarhed om forskellen på DC- og stepper-motoren, hvilket er blevet forklaret i afsnit 3.2.2.

Ligeledes er der dannet et større kendskab til, hvordan DC-motor og den unipolære stepper-motor kontrolleres.

5 | Litteratur

- [1] P. Microdrives. Ab-022: Pwm frequency for linear motion control. [Online]. Available: <https://www.precisionmicrodrives.com/ab-022>