Opgave 4

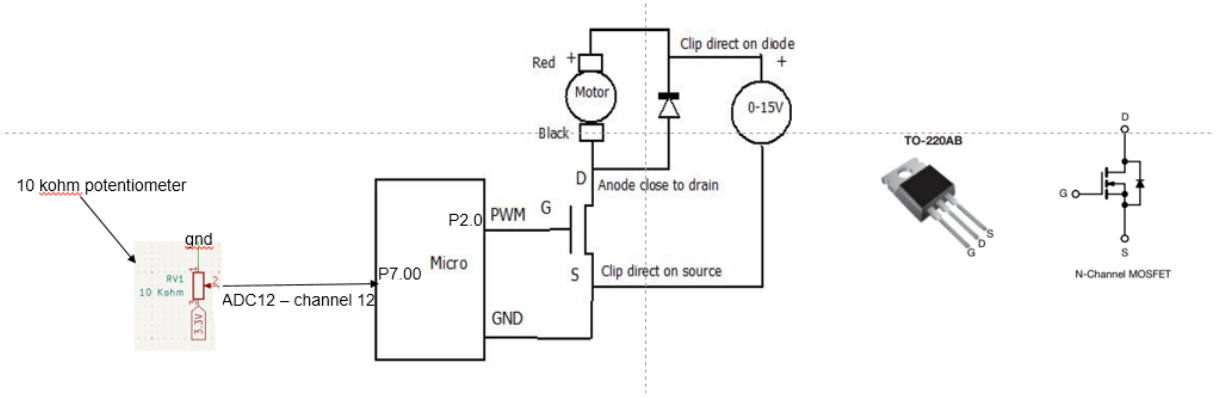
Motor driver and encoder

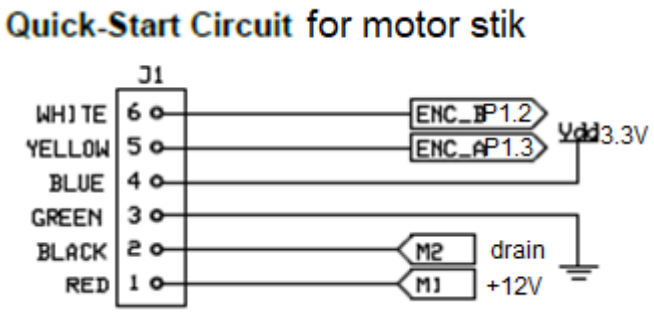
Forudsætningen for opgaven her er at du har udført opgave 3

Formålet er at styre en DC-motor vha. PWM med en frekvens på op til ca.10kHz

Formålet er at måle encoder signalerne fra motoren vha. af et oscilloskop

1. ADC12 samples skal styre pwm pulsebredden som i opgave 3 ad7. dvs. PWM signalet skal i denne opgave ud på P2.0 og analog signal skal ind på p7.0 fra et potentiometer. Styring af sampleraten skal ske som i opgave 3.





2. Hent databladet for motoren – under opgave 4
3. Hent databladet for MOSFET IRL530 under opgave 4
4. Forbind en MOS FET IRF530 som vist i diagrammet her til dit msp430 board og et potentiometer på 10 kohm til adc12 indgangen på P7.0. Gate på Mos FET skal forbindes til PWM udgangen på P2.0. En ekstern spændingsforsynings Plus forbindes til den røde motor-ledning på motoren – maks. spænding 15 V benyt f.eks. 12 V katoden (stregen) på dioden 1N4006 forbindes som vist (ikke til mikrocontrolleren!) - sort motor-ledning til drain på MOSFET. Tilslut MSP430 gnd til gnd som vist til MOSFET source og spændingskildens gnd.
5. Få en anden gruppe til at kontrollere forbindelserne før tændes for 0-15V spænding indstillet på 12V.
6. Opret et nyt projekt i Code composer og kopier adc12 koden inkl. ADC12 interrupt service rutine og PWM koden med Timer TA1 fra opgave 3 ind i et nyt projekt og foretag disse justeringer:
    a. Funktionen der er i opgave4 templaten for skalering af SMCLK til 25MHz skal benyttes til timer TA1. Og den skal kaldes fra main() før while(1).
    b. sæt ADC12 til 10 bits opløsning med `ADC12CTL2=ADC12RES_1;  //10 bit adc` i initialisering af adc12. Hvis adc12 converteren ikke har slået interrupt til på adc12, så skal du få det op at køre i denne opgave og hjælpen til koden findes i slide serie lektion 7. Hints (ADC12IE |= ADC12IE0;) skal der stå i initialisering af adc12, og der skal være en service rutine der starter med (#pragma vector = ADC12_VECTOR // interrupt service rutine for ADC) hints se slide serie lektion 7.

c. Opsætning af PWM ændres så 1024 (10 bit opløsning) sættes som topværdi i TA1CCR0. Hvilken frekvens opnås når SMCLK er 20 MHz – regn det ud Beregn PWM frekvensen med denne formel og sæt f source til 20 MHz:

$$f_{PWM} = \frac{f\ clock - timer}{2 * TAxCCR0} = \frac{fsource}{IDx} \times \frac{1}{2 * TAxCCR0}$$

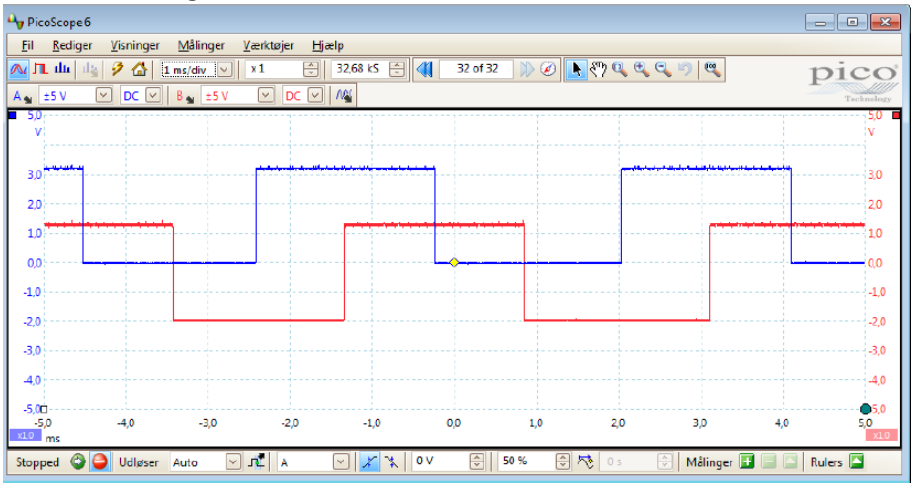
d. I main() funktionen under en betingelse if (TA1CTL & TAIFG) sættes resultatet fra adc konverteren ind i TA1CCR1 registeret (som i opgave 3 del7) , så når du drejer på potentiometeret skal du være i stand til at styre hastigheden på motoren. I princippet som du gjorde med i opgave 3 del 7 med dc spændingen.

7. Kompiler projektet og prøv det af i MSP430.
   a. Forbind et potentiometer til P7.0 som vist på figuren på siden ovenfor og et oscilloskop til P2.0. Kontrollér når der er skruet helt op på potentiometer så spændingen på midter-benet er 3.3 V (dvs. den digital værdi er 1023) at duty cycle på PWM på P2.0 er 100% og at den kan skrues ned på 0 – når spændingen ud fra midter benet er 0V.

8. Afbryd forbindelsen til mikrocontrolleren fra PC og Forbind kredsløbet som vist ovenfor til motoren vha. breadboardet
9. Kør programmet og kontroller, at motorens hastighed kan styres af potentiometeret indstilling. Tjek PWM-outputtet på oscilloskopet.
10. I journalen/rapporten for opgave4:
    a. Gør redefor hvordan PWM frekvensen hænger sammen med top-værdien (TA1CCR0) for timeren TA1 og hvilken frekvens der kan opnås hvis topværdien sættes til 1024 – beregn og mål PWM frekvensen med oscilloskop på P2.0.
    b. Hvorfor er dioden nødvendig?
    c. Hvorfor kan motoren ikke tilsluttes direkte til mikrocontrollerens PWM-udgang - det vil sige hvorfor den nødvendige MOSFET?
    d. Hvilken spænding på MOSFET-porten får den til at tænde (se datablad)? Er denne spænding passende, når den drives fra mikrocontrollerens PWM-udgang?
    e. Hvad er drain-source modstanden for MOSFET'en, når den er tændt (se datablad)?
    f. Hvad er den maksimalt tilladte drain-strøm og drain-spænding for MOSFET'en (se datablad)? Er vi inden for disse grænser, når vi kører motoren?
    g. Hvor hurtigt kan MOSFET tænde og slukke (se datablad)? Og hvordan hænger det sammen med PWM frekvensen på P2.0 signalet der går ind på Gate

DEL B. Encoder og motor

11. Tilslut motor-encoderens positive spændings indgang (blå) til mikrocontrollerens 3,3V (ikke den eksterne forsyning!) og ground (grøn) til mikrocontrollerens GND.
12. Tilslut oscilloskopprober til hver af encoder-udgangene (hvid og gul) vha. små ledninger ned i stikket, og bemærk frekvenser og faser for de to signaler på oscilloskopet. Prøv variere motorhastigheden fra 0 og op vha. potentiometer tilsluttet ADC12 P7.0 – jfr. de første punkter
13. Du burde se noget som dette:



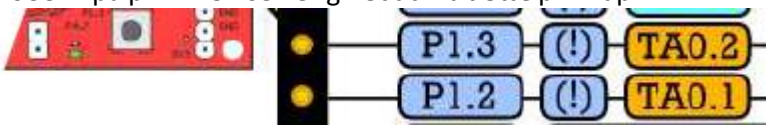
14. Vend nu motorforbindelserne (rød og sort). Bemærk motorens rotationsretning og observere encoder signalerne på oscilloskopets.
15. Alternativt - Du kan også afbryde spændingen til motoren (rødt eller sort) og dreje encoderen med hånden og observere encoder signalerne på scopet
16. Dokumentation: Vis oscilloskopbilleder for de to motorpolaritets tilfælde. Kommenter resultaterne forklar. Og svar på disse spørgsmål.

    a. Forklar hvordan encoderen kan give information om motorens hastighed og mdrejningsretning.
    b. Er encoderens udgange helt periodiske (stabile på oscilloskopet) eller varierer de lidt (det kaldes jitter hvis de står og svejer frem og tilbage)?

    Forsættelse følger lektion 11

17. **Opgave 4 del 2.**
    I lektion 11 – forberedelsen er den nødvendige kode gennemgået.

    Formålet er at sætte msp430 op til at måle frekvensen af motor-encoder signalerne vha. timerA0 sat op i capture mode.
    Og formålet er at måle den maksimale frekvens på encoderen ved 12 V på motor med oscilloskop og vha. msp430F5529.
18. Jfr. lektion 11 Capture fungerer ved at hver gang der sker en ændring på en Capture indgangspind vil TimerA0's tæller register (TA0R) værdi blive kopieret over i det til pinden tilhørende capture register TA0CCRx– her benyttes pin P1.2 med capture register TA0CCR1 og capture register TA0CCR2 på pin P1.3. se her givet ud fra dette pinmap



19. Opret et nyt projekt i Visual studio code for at skrive den kode der kan fange encoder pul ser. Timer A0 sættes op til at køre med ACLK som clock med frekvens på 32768 Hz –der benyttes ikke nogen pre-scaler på timer clock – hvad vil tælleren tælle til mellem hvert rising-edge på input af P1.2 og P1.3?
20. Sæt pind P2.2 og P2.3 op som udgang og toggle (hver anden gang 0V hver anden gang 3.3V P2OUT^=BIT2; P2OUT^=BIT3;) de to udgange for hver gang programmet kommer ind i service rutinen for Capture-register der fanger pulsens rising-edge på henholdsvis på P1.2 til TA0CCR1 og P1.3 til TA0CCR2.
21. Programmet der kan fange pulser vha. Capture på TA0 skal testet, så du ved at frekvensen på pulser, der fanges, måles rigtigt. Det skal testes vha. en signalgenerator (se video om hvordan den betjenes). Inden generatoren forbindes sættes den op til 500 Hz og en spæn ding på 3.3V spids til spids – juster DC-offset så pulsen går fra 0 V til 3.3V mål på firkant signalet med oscilloskop så det stemmer at frekvensen er 500Hz og spændingen går fra 0 til 3.3V.
22. Forbind to prober fra oscilloskopet til henholdsvis P2.2 og P2.3 – mål frekvensen på de 2 kanaler og sæt 2 andre prober på indgangssignalerne til P1.2 og P1.3 – sammenhold de to sæt af målinger på indgang P1.2 med P2.0 og P1.3 med P2.2– Passer de målte frekvenser på P2.2 og P2.3 med indgangsfrekvensen der måles Dokumenter med oscilloskop billede(er) og begrund/beregn - hvorfor det passer.
23. Nu oprettes et nyt projekt i Visual studio code – hvor alt koden fra 4.1 –del1 tages ind og det nye capture kode der er testet under punkt 24 lægges ind også – sørg for at der er initialiserings-funktion til hver enhed (ADC12, Timer1A, TimerA0 – endnu bedre opret moduler(adc12.c, adc12.h, timerA0.c, timerA0.h .. osv.)
24. Capture delen jfr. pkt 24 er nu verificeret og vi ved at der måles rigtigt. Nu forbindes motor encoder-udgangene: for B-encode (hvid ledning) til P1.2, der sender signal til capture register TA0CCR1 og for A- encoder (gul ledning) til P1.3, der sender signal til capture register TA0CCR2. – og igen måles som under pkt. 24 – dokumenter hvad maksimum frekvensen for encoderne er når motoren får 12 V og der er 100%PWM ind på gate – mål frekvenserne med oscilloskop på

henholdsvis P1.2 og P1.3 og – noter ned og dokumenter vha. oscilloskop billede værdierne og benyt de målte frekvenser på P2.3 og P2.2 til at udregne tælle-værdien der er mellem to capture interrupts– husk clock frekvensen er 32768 Hz. Jfr. pkt. 21.

25. Prøv at få capture til at fange både opadgående og nedadgående kant på encoder pulserne ved i capture-compare control register 1 TA0CCTL1 at ændre bit CM_1 til CM_3 for (P1.2) og i capture-compare control-register TA0CCTL2 ændres CM_1 til CM_3. Benyt de målte frekvenser på P2.0 og P2.2 til at udregne, hvilken værdi er der mellem to capture interrupts stadig med timer-clock frekvens 32768 Hz. Hvad er maksimal encoder frekvens så? Hvor godt stemmer det med at frekvensen på P2.3 og P2.2 er dobbelt så stor

Opgave 4 del 3.

Formålet er at benytte erfaringerne fra opgave 4 del 1 og del 2 til at sætte et samlet projekt op der kan sætte hastighed for motoren vha. en spænding fra et potentiometer ind på ADC12 og sample den og at benytte to kanaler i TA0 til capture af encoder A og B pulsernes kanter (rising - og Falling edge) til at holde konstant hastighed med spændingsvariationer på forsyningen og belastning af motoren.

I lektion 12 – og på video gennemgås det reguleringstekniske og hvad der skal ændres for at fange både opadgående og nedadgående kanter på de to encodere og derved få fuldt udbytte af 48 pulser per omdrejning

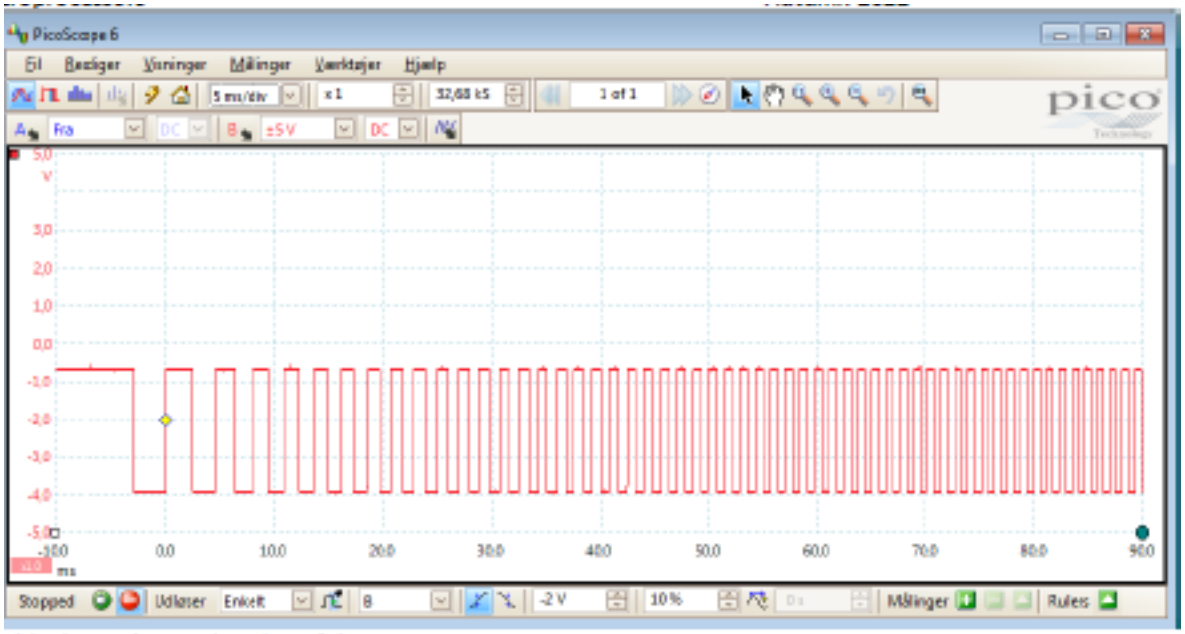
26. Der skal arbejdes videre med programmet fra del 2.
27. Capture skal være opsat til at capture på både op og nedgående kanter, som i opgave 4 del 2 Og i service rutinen for timer compare match TA0 skal værdierne i henholdsvis TA0CCR1 (for B-encoder) og tilsvarende for TA0CCR2 gemmes vha. i følgende algoritme program snippet:

```
captured_value1 = captured_value1+ TA0CCR1-last1;

last1=TA0CCR1;
i++;
if (i==2){     // efter 3 interrupts så er  talt antal timer clock
 freq1=32768/captured_value1;   //er frekvensen på encoder B
   captured_value1=0;
              i=0;
              t_flag1=1;
          }
```

Dette sættes ind under case 0x02 – jfr slide serie lektion 11
da der er 3 flanker på encoder signalet der definerer en periodetid og dermed en encoder frekvens, derfor i=2

28. Det er nyttigt at estimere motorens tidskonstant. Indstil potentiometer til 'fuld hastighed', men afbryd potentiometeret fra 3,3V-forsyningen (så pwm cyklus skal være 0). Indstil oscilloskopet til at udløse enkeltskud på P1.2 (encoder-output). Derefter tilslut potentiometeret til 3,3V-forsyningen igen (så duty cycle pludselig skulle stige fra 0 til 100 %). Du skulle se noget som dette på oscilloskopet (kun P1.2 pin-signalet er vist her) (tidsskalaen kan være anderledes):



This shows the acceleration of the motor

29. Dokumentation: Vis et skærmbillede af oscilloskopet, forklar sporet set på oscilloskopet og lav et groft skøn over motorens tidskonstant. [Tip: For et system med en enkelt tidskonstant nås 63% af den asymptotiske værdi i periode på en tidskonstant. Estimer den asymptotiske frekvens, og estimer derefter den tid, det tager fra starten til at nå 63 % af denne værdi.] Hvis det driller så vendt med det!

30. **Forøg tidskonstanten på systemet**
    Det har vist sig nødvendigt af stabilitetsmæssige årsager at øge tidskonstanten for systemet (den årsag synes at være (a) "jitteren" eller variationen i tiden mellem indkoderimpulser og (b) relativt lang latenstid ved måling af motorhastigheden. Her vil vi gøre dette ved at bremse motorens reaktion på ændringer i encoder frekvensen i arbejdscyklussen input. Det vil sige, at vi tilføjer en slags digitalt lavpasfilter. Se senere i opgaven – eksperimenter

    Nu skal ønsket hastighed Xd (som frekvens) og fejl i hastighed Xe beregnes i programmet inde I main() funktionen under en betingelse if (TA1CTL & TAIFG) hvor pulsviden opdateret dvs. i TA1CCR1 registeret. Som input giver vi adc12 en dc værdi Vin fra et potentiometer med midterbenet forbundet til P7.0. Det antages at frekvensen på encoder er proportional med rotationen og med spændingens gennemsnitsværdi over motoren. Mål Vin med voltmeter på P7.0 og udregn den digital værdi `adc_value` målt med adc12 - jfr . opgave 3. det kan så sættes ind i formlen her
    `Xd=(adc_value* freqMax)/1023.0;` // den ønskede hastighed gives som en ønsket encoder frekvens skaleret med adc_value i forhold til maks. værdi. freqMax er encoder frekvens målt med ren 12 v dc over motoren svarende til 100% PWM duty cycle
    og den ønskede frekvens svarende til n rotationer / sekund

31. Når der i timer TA0 capture service rutinen er regnet en frekvens ud sættes et flag, t_flag1=1 og i main benyttes den så til at regne fejlen ud i frekvens som Xe= Xd- encoder-frekvens
32. Nu ønskes der lavet beregninger og eksperimenter med gains – sæt en oscilloskops-probe på PWM output på P2.0 og når det går i selvsving/ser moduleret ud så er G for høj. Demand Xd sættes ved at måle dc spænding på potentiometer midterpind Vin og benyt tallet i formlen ovenfor!

Chosen			Measured/observed			Calculated	
Loop gain (G)	Drive voltage	Demand speed (frekvens) (`Xd`)	Duty cycle (%)	Stable/ unstable?	Actual speed (`Frekvens`)	Actual speed/ Demand speed	G/(1+G)

10	12						
20	12						
50	12						
5	12						
2	12						
…	…						

Forklar i dokumentationen hvad vi ser i tabellen og giv argumenter for resultaterne.

Nu skal der eksperimenteres med at ændre spændingen på de 12 V
Regulering af hastighed mod ændring i spændingen:

33. Kør koden og noter værdierne af Xd, målt frekvens på P1.2 med oscilloskop og målt PWM duty_cycle med oscilloskop.
34. Skift motor-spændingen med f.eks. 20% (maks. 12V) til V2 og noter %-ændringen i hastigheden.
35. Eksperimenter med forskellige værdier af loop gain G. Start med følgende foreslåede værdier.
    Bemærk: Gc er sløjfeforstærkningen G korrigeret for tabet i G på grund af reduceret drivspænding. Når motor spændingen reduceres, skal sløjfeforstærkningen G korrigeres med det samme forhold, så Gc = G V2/V1.
    Udfyld denne tabel:

Chosen			Measured/observed			Calculated			
Loop gain	Drive voltage	Demand speed (frekvens) (Xd)	Duty cycle	Stable/ unstable?	Actual speed (frekvens)	% Change in speed (frekvens) (when drive V changed)	Actual speed/ Demand speed	G/(1+G)	$G_c/(1+G_c)$
10	12					-			
10	10					?			
20	12					-			
20	10					?			
50	12					-			
50	10					?			
5	12					-			
5	10					?			
2	12					-			
2	10					?			
…	…								

36. Forklar overstående tabel og argumenter for resultaterne

37. Hvis der er ustabilitet prøv så at gøre noget i stil med nedenstående for frekvensen freq1 eller freq2 – ex. freq1 = freq_prev + slew*(freq1 - freq_prev);

38. **Dokumentation i form af en Rapport**
    Forside med billeder af jer og navne på gruppemedlemmer og gruppe nummer:
    Indholdsfortegnelse med side numre og 4 hovedafsnit i alt:
    Et afsnit for hver opgave del – med svar på alle spørgsmål, beregninger og forklaring til figurer

Og et afsnit med 4 oversigt over funktioner i programmet med forklaring af hvad de gør/udfører
Interrupt diagram med forklaring af hvordan koden kommunikerer med hinanden vha. service rutinerne.
Alt koden vedlægges rapporten i bilag med fortløbende side numre.
Rapportens dokumentationsværdi indgår i den samlede bedømmelse

39. evt nødvendigt for at minimere jitter
Sæt en 74HC14 kreds (inverter med smittrigger – datablad på learn under opgave 4) på bread-boardet forbind to invertere efter hinanden og lad encoder pulserne gå igennem 2 af disse inden de forbindes til henholdsvis P1.2 og P1.3.