Prosjekt del 1

Mikrokontrollerkretser

- Opprett gjerne et eget GIT repository for prosjekt del 1.

Om prosjekt del 1

Nødvendig SW for første del av prosjektet er Microchip Studio og Autodesk Eagle (eller lignende). Første del skal gi en introduksjon til mikrokontrollerkretser, enkel digital I/O og PWM.

Dag 1 (og evt. 2)

Oppgave 0 - Installere Autodesk Eagle og Arduino IDE

Autodesk Eagle og Arduino IDE skal installeres på lab-maskinen (eller egen PC).

- Velg gratisversjon av Eagle. Denne har noen begrensninger rundt størrelse på kretskort etc., men dette er uproblematisk med tanke på bruken her. https://www.autodesk.com/products/eagle/free-download
- Det kan være nødvendig å registrere bruker hos Autodesk for å fullføre installasjonen.
- Arduino IDE er tilgjengelig her: https://www.arduino.cc/en/Main/Software

Oppgave 1 - Strømforsyning

1. Velg 3.3V eller 5V lineær spenningsregulator. Husk at strømforsyningen skal kunne drive både LED og mikrokontroller. Sett dere inn i de viktigste parametrene i databladet (her må dere se på hva som skal kobles opp videre ut i oppgaven for å finne et tilnærmet effektbehov).
2. Tegn opp strømforsyningskretsen i Eagle. Hva er anbefalt innspenning og maksimal utstrøm for denne konfigurasjonen?
3. Sjekk koblingsskjemaet vha. faglærer eller studass. Sørg for å få koblet opp kretsen.
4. Bruk lab-strømforsyning for å gi kretsen en fornuftig innspenning.
5. Mål utspenningen med multimeter. Hva er denne? Er dette som forventet?

Oppgave 2 - Tegning av mikrokontrollerkrets m/ programmeringsheader

ATmega168 mikrokontroller skal kobles opp i en relativt minimal konfigurasjon (få komponenter).

1. Tegn kretsskjema for ATmega168 m/spenningsregulatoren valgt i forrige oppgave. Bruk http://www.atmel.com/Images/Atmel-2521-AVR-Hardware-Design-Considerations_ApplicationNote_AVR042.pdf som utgangspunkt for å komme frem til nødvendige komponenter og komponentverdier.

2. Mikrokontrolleren skal kobles slik at den kan programmeres med ISP ("Atmel AVR-språk" for SPI-basert programmering). Oppkobling av programmerer er beskrevet her: http://www.atmel.com/Images/Atmel-42330-Atmel-ICE_UserGuide.pdf . Tegn på en pin-header i kretsskjemaet hvor programmeringspinnene fra mikrokontrolleren er tilgjengelige i rett rekkefølge.

3. Ta vare på kretsskjemaet -- husk at alt på lab-maskinene har en variert brukergruppe, og dine filer kan forsvinne ved reinstallasjon/tanking av maskiner.

Oppgave 3 - Et enkelt "hello world-program"

1. Start et nytt C++-prosjekt i Atmel Studio. Sjekk at eksempel-programmet lar seg kompilere. Beskriv eventuelle valg som tas underveis.

2. Basert på eksempelprogrammet, lag et program for å sette en mikrokontrollerpinne som utgang, og skriv en lav verdi til denne. Sjekk at programmet fremdeles kompilerer.

3. Bruk gjerne versjonskontroll og sjekk inn denne versjonen.

Oppgave 4 - Lysdiode (LED)

En lysdiode (LED) skal kobles til utgangen på den lineære spenningsregulatoren fra oppgave 1. Denne skal fungere som en "power-on"-indikator.

1. FØR dette kobles skal det tegnes et modifisert kretsskjema som viser hvordan LED-en er planlagt oppkoblet. Lysdioder brenner opp (eller blir skadet i større eller mindre grad) dersom maksimal foroverstrøm overstiges.

2. Utfør nødvendige beregninger og velg verdi for strømbegrensende seriemotstand.

3. LED-en kobles opp basert på kretsskjemaet og utfør funksjonstest.

Oppgave 5 - Oppkobling av krets og programmering av mikrokontroller

1. Kretsen fra oppgave 2 skal kobles opp.
2. LED-en fra oppgave 4 skal flyttes slik at mikrokontrolleren kan styre denne med utgangspinnen valgt i oppgave 3. Husk at strømbegrensende seriemotstand fremdeles må være med.

   Bør LED-en kobles slik at mikrokontrolleren sourcer eller sinker strøm? Hva betyr dette, og hvorfor er ditt valg rett? Bruk gjerne referanse til datablad. Kretsskjemaet skal oppdateres for å reflektere valget.
3. Etter at LED og programmeringsgrensesnitt er koblet opp basert på SPI-figuren her: http://www.atmel.com/Images/Atmel-42330-Atmel-ICE_UserGuide.pdf så skal oppkoblingen kontrolleres av studass eller faglærer.
4. Sørg for at powersupply er av. Koble til Atmel ICE USB-programmerer og debugger. Én student kobler, de andre kontrollerer. Sjekk med faglærer eller studass før dere slår på strøm.
5. Velg "Device Programming" i Atmel Studio. Sjekk at korrekt device er valgt. Forsøk å lese ut device signature. Er denne korrekt?

   Power må være på før programmering kan utføres. Grønn LED på Atmel ICE indikerer OK spenning på target.
6. Gå til memory og utfør programmering. Beskriv hva som skjer. Hvilken filtype leses av programmereren, og hvor havner innholdet i denne?
7. Lag en enkel skjematisk oversikt over programmeringsoppsettet (ikke Eagle, men "boksologi"). Beskriv hva som er target og hva som er host.
8. Hvorfor sier vi at det utføres krysskompilering når vi kompilerer i Atmel Studio for AVR-mikrokontrollere?

Oppgave 6 - Soft-blink

Mikrokontrollerprogrammet skal utvides til å blinke pulserende og "mykt" ved hjelp av pulsbreddemodulasjon (PWM).

1. Hva sier databladet til LED-en om maksimal strøm ved pulsbreddemodulasjon? Er denne lik, mindre eller større enn maksimal konstant strøm?

2. Velg PWM-frekvens og begrunn valget. Test valget ved å forsøke f.eks. 50% og 100% duty-cycle. Sjekk responsen. Bruk gjerne et oscilloskop. Stemmer målt frekvens overens med valgt frekvens? Og eventuelt hvorfor ikke?

3. LED-en skal blinke mest mulig mykt og jevnt fra 0 til 100% lysstyrke. Beskriv løsningen og resultatet.

   Inneholder databladet informasjon om lysutbytte som funksjon av strøm? Er sammenhengen lineær?

   Responderer menneskets øyne lineært på lys?

Husk å ta vare på både kretsskjema(er) og kildekode!
Husk push til ekstern server dersom git er i bruk.

Valgfritt

Oppgave 7 - Avbrudd (interrupt)

I denne oppgaven skal avbrudd (interrupt) benyttes for å respondere på knappetrykk.

1. Velg en pinne på mikrokontrolleren som skal benyttes til input.
2. Utvid kretsskjemaet med en knapp. Knappen kobles slik at pinnen valgt i punkt 2 jordes når knappen er trykket ned.
3. Hva skjer med input-pinnen når knappen ikke er trykket og mikrokontrolleren ikke driver denne? Hvordan kan dette løses?
4. Koble opp kretsen og lag et lite program som leser knappens verdi og setter ut denne på lysdioden. Lysdioden skal lyse når knappen er trykket ned. Dette gjøres uten bruk av interrupt. Test programmet på mikrokontrolleren. Gå videre til neste punkt når dette fungerer.
5. Nå skal programmet skrives om til å være interrupt-basert, og fungere litt annerledes. Hver gang knappen trykkes skal lysdioden bytte tilstand (lyse / ikke lyse). Bruk pin change interrupts til dette.

   For å få aktivert pin change interrupts for en gitt pinne, må Pin Change Interrupt Control Registeret og ett av Pin Change Mask-Registerene stilles inn korrekt.

   Videre er det nødvendig å lage selve ISR-funksjonen som kalles av interruptet, i tillegg til at interrupts må slås på globalt. Se dokumentasjonen på avr-libc-websiden.

Merk at pin change interrupts vil trigge på BÅDE stigende og synkende flanke. Dette må håndteres.

6. Toggle KAN fungere ok etter at forrige trinn er fullført. Men direkte bruk av interrupts på en knapp/bryter på denne måten er strengt tatt ikke å anbefale. Brytere, knapper og releer har ofte prell/bounce i større eller mindre grad. Det vil si at ett trykk fort kan føre til flere interrupt raskt etterhverandre. Finn en løsning for å unngå dette.

   Tips: Sjekk enten at det har gått en minstetid mellom knappetrykkene, eller les knappen med en fast rate. F.eks. ved å bytte ut pin change interrupt med et timer-basert interrupt. Med timer-basert løsning kan samplingsrate for knappen enkelt velges.

   Det er også mulig å lage et analogt filter i tilknytning til knappen for å fjerne prell/bounce-effekter, men dette er ikke ansett som en del av oppgaven. Vurdér dette hvis dere har god tid etter oppgave 9.

Oppgave 8 - Analog-til-digital-konvertering

Et tre-pinners potensiometer kan benyttes som justerbar spenningsdeler.

1. Utvid kretsen ved å koble et 10kOhm potensiometer slik at spenningen inn på en valgfri ADC-pinne kan justeres mellom 0V og VCC.

2. Husk at AREF og eventuell AVCC må kobles til! Dette skal også være synlig i kretsskjemaet. Hva er funksjonen til AREF?

3. Koble opp kretsen. Sjekk gjerne med multimeter at den analoge verdien inn på valgt pinne faktisk lar seg justere opp og ned.

4. Lag et program som leser den analoge verdien fra potensiometeret og benytter den avleste verdien til å sette lysstyrken på LED-en.

   Dette krever bruk av dimmeprogrammet fra Oppgave 6. Dersom dere ikke kom i mål med Oppgave 6, så skal dere i stedet sjekke om avlest spenning er større enn 2V, og aktivere LED-en dersom dette er tilfellet, og slukke den hvis spenningen er lavere.

   Tips:
   - Før bruk av ADC-en er det nødvendig å første aktivere ADC og (eventuelt) sette prescale. Sjekk hva prescale gjør! Se ADCSRA-registeret i databladet. Dette kan gjøres ved oppstart.
   - Når ADC-en skal benyttes for å lese en analog verdi, så må først ADMUX settes til rett kanal. Deretter må konvertering initieres, og det må ventes frem til ADIF-flagget i ADCSRA-registeret er satt. Når ADIF er satt, så er konverteringen ferdig, og den avleste analoge verdien er tilgjengelig i digital form i ADC-registeret.