in1060 ukeoppgaver i Arduino, uke 3

1 Analoge signaler

1.1 Lese og skrive analoge signaler

Når vi jobber med digitale systemer forholder vi oss som regel til to tilstander. Av eller på (strøm eller ikke strøm), f.eks. er knappen trykket ned eller ikke? Dette kalles gjerne digitale signaler. I den fysiske verden er det derimot mer enn bare av eller på. Temperatur er for eksempel ikke bare "av" eller "på". Selv om Arduino er et digitalt system er det også mulig å lese og skrive analoge signaler ved hjelp av Arduinoens innebygde ADC (analog-to-digital converter).

1. Fra hvilke porter kan du lese analoge signaler og hvordan gjør du dette?
2. Fra hvilke porter kan du sende ut analoge signaler og hvordan gjør du dette?
3. Hva er PWM?

1.2 Seriell kommunikasjon

I Arduino-språket er finnes Serial. I denne referansen https://www.arduino.cc/reference/en/ står det om alle nøkkelordene i Arduino. Serial gjør det mulig for oss å kommunisere mellom Arduino og datamaskin, flere Arduino-kort, og annet med seriell komunikasjon.

1. hvilken pin kan brukes for å lese potensiometeret?
2. Koble opp et potensiometer og print ut tall i Serial monitor
3. Bruk map() slik at verdiene potensiometeret sender er mellom 0 og 100.

```
1       void setup(){
2       // dette skjer bare en gang
3       pinMode(/*pin*/, input);
4       Setial.begin(9600)
5       }
6
7       void loop(){
8
9 }
```

1.3 Tid og serial

Funksjonen millis() returnerer antall millisekunder som har gått siden Arduinoen startet. Dette kan være nyttig for en rekke bruksområder. Det krever derimot ofte litt ekstra planlegging for at koden skal fungere som man vil. Det kan derfor være lurt å øve seg litt på hvordan millis() fungerer.

Skriv et program som skriver "Det har gått fem sekunder!" til serial monitor hvert femte sekund ved å bruke millis().

1.4 Pulserende lys

Koble en LED til en av portene til Arduinoen som lar deg sende analoge signaler (merket med PWM eller) slik som i figuren til høyre.

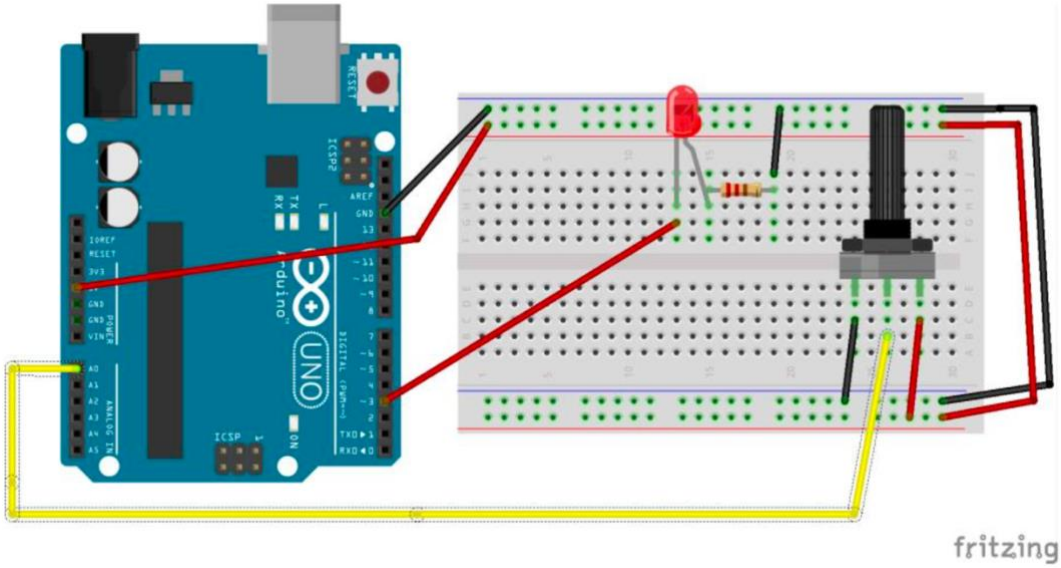
1. Bruk funksjonen analogWrite() til å sette lysstyrken til LED-pæren ved å sende med forskjellige verdier mellom 0 og 255. Prøv for eksempel med 20 og deretter 200 for å se hvordan lysstyrken endres.

2. Skriv et program som får LED-pæren til å "pulsere". Med andre ord, få LED-pæren til å fade inn og ut i en jevn rytme. Prøv med forskjellige delay-tider for å finne en passe hastighet på pulseringen.

3. Prøv til slutt å skrive et program som lar deg kontrollere lysstyrken med et potentiometer (Her er kretsen fra oppgaven om potensiometer nyttig). Her må du bruke både analogRead() for å lese fra potentiometeret og analogWrite() til å sette lysstyrken. Denne deloppgaven er uavhengig av forrige deloppgave.

4. Les beskrivelsene av komponenter Arduino Projects Book side 6 - 9. Kommer du på flere måter å kontrollere lysstyrken på? For eksempel ved å bruke noen av komponentene i startsettet?

5. Er det mulig å lage en krets som lar et potensiometer styre lysstyrken til en LED, uten å bruke Arduino?



1.5 RGB LED

I Arduinosettet følger det med en såkalt RGB LED. Denne er litt annerledes enn de andre.

1. Hva er spesielt med denne og hvorfor har den fire "føtter" i stedet for to?

2. Finn tre porter på Arduinoen som støtter analoge signaler (PWM), f.eks. 9-11, og koble hver av disse portene til hver sin fot på RGB-en. Om du trenger litt hjelp til å koble opp kretsen kan det lønne seg å se på oppgave 4 i Arduinoboka. Husk også å bruke resistorer (viktig!) og koble den fjerde foten til jord.

3. Når du har koblet opp kretsen: Prøv å skrive enkel testkode for å finne ut om RGB-en fungerer ordentlig. Prøv å sende forskjellige verdier mellom 0 og 255 til de forskjellige føttene med analogWrite() for å se at RGB-en endrer farge.

4. Det følger også med tre potensiometere i Arduino-settet. Bruk disse til å kontrollere fargen på RGB-en der hvert potensiometer styrer hver farge.

Hint: Verdiene du får inn ved å bruke analogRead() er mellom 0 og 1023, men største verdi du kan sende ut med analogWrite() er 255. Du må derfor skalere verdiene du får fra analogRead() slik at de er innenfor rekkevidden 0 til 255. Dette kan gjøres på flere måter, en av disse er funksjonen map() som du kan lese om her https://www.arduino.cc/en/Reference/Map

1.6 Konvertering av signaler

1. Kan en konvertere analoge signaler til digitale? Forklar eventuelt hvordan, tegn en slik krets, eller skriv en slik funksjon i Arduino-språket.

2. Kan en konvertere digitale signaler til analoge? Forklar eventuelt hvordan, tegn en slik krets, eller skriv en slik funksjon i arduino-språket.

2 Knappetrykk

2.1 Upålitelig knappetrykk

1. Hva gjør denne kombinasjon av krets og kode for upålitelig til å informere oss om knappen er trykket ned eller ikke?
2. Endre koden slik at systemet kan lese (på en pålitelig måte) om knappen blir trykket ned.
3. Endre kretsen slik at systemet kan lese (på en pålitelig måte) om knappen blir trykket ned, men bruk koden fra bildet i oppgaven.

Hint: Her kan det være veldig nyttig å lese:

- www.arduino.cc/en/Tutorial/InputPullupSerial
- For en litt bedre forklaring kan du se denne videoen: http://youtu.be/wxjerCHCEMg.

2.2 Knappetrykk og serial monitor

Koble opp en knapp til Arduinoen slik at du kan lytte på den. Skriv et program som skriver "Trykket!" til serial monitor når knappen trykkes. Når programmet kjører, åpne serial monitor på PC-en.

1. Hva skjer i serial monitor når du holder knappen inne?
2. Kan du forklare hvorfor dette skjer?

2.3 Knapp og blink

Koble til en LED til Arduinoen og programmer LED-pæren slik at den blinker med ett sekunds mellomrom ved hjelp av delay(). Koble deretter til en knapp og skriv "Trykket" til serial monitor hver gang knappen blir trykket.

1. Trykk på knappen mange ganger etter hverandre og følg med i serial monitor. Hvorfor blir bare et fåtall av knappetrykkene registrert?
2. Tilpass koden: Løs oppgaven ved bruk av millis() i stedet for delay().
3. Ut fra observasjonene fra oppgavene ovenfor: I hvilke situasjoner må man bruke millis() i stedet for delay()?
4. Burde man alltid bruke millis()?

2.3.1 Debounce

Av og til må vi ta hensyn hvor ofte og hvor raskt funksjonen loop() blir kjørt. Dette blir tydelig når en jobber med å registrere knappetrykk: Hvordan kan vi vite om det er et nytt knappetrykk hver gang vi leser at knappen er trykket ned? å takle dette problemet kalles for "debounce".
https://www.arduino.cc/en/Tutorial/Debounce

Debounce er ikke nødvendig for å få alle systemer til å fungere, men det er en typisk feil som kan føre til at systemet ikke fungerer slik det skal. SOS signalsystemet fra uke 1 trengte ikke debounce for å virke, men hvis f.eks. antall knappetrykk spiller en rolle må du ta hensyn til dette.

1. Skriv kode for å registrere knappetrykk der du tar hensyn til debounce ved bruk av delay(). Skriv antall knappetrykk til serial monitor når knappen trykkes.
2. Tilpass koden slik at du bruker millis() i stedet for delay.
3. *Ekstra*: Tilpass koden slik at en kan holde inne knappen i ubestemt tid uten at det tolkes som flere trykk.

3 Funksjoner og modularisering

3.1 Modularisering av "Digital Hourglass"

Se på koden til oppgave 8 (Digital Hourglass) i Arduinoboken. Selv om programmet har relativt lite funksjonalitet ser vi at det fort kan bli mye kode i loop(). Dette gjør at det kan bli et rotete program som er vanskelig å lese. Ser du noen måte å modularisere denne koden på for å gjøre den mer oversiktlig?

Hint: prøv å identifiser deler av koden som kan uttrykkes som et funksjonskall. For eksempel på formen "knappTrykket()" eller "blink()".

3.2 Modularisering

Modularisering er veldig viktig i emnet in1060. Med ryddig kode blir det raskt og enkelt for deg selv og andre på gruppen å endre koden når systemet skal endre seg.

Modulariser dette programmet slik at det bruker funksjonen **blink(int lengdeimillis)**

```
1   void setup() {
2   pinMode(2, INPUT_PULLUP);
3   pinMode(4, OUTPUT);
4   Serial.begin(9600); 5 }
6
7   void loop() {
8   if(digitalRead(2)){
9   digitalWrite(4, HIGH);
10  delay(300);
11  digitalWrite(4, LOW);
12  delay(300);
13  digitalWrite(4, HIGH);
14  delay(300);
15  digitalWrite(4, LOW);
16  delay(300);
17  digitalWrite(4, HIGH);
18  delay(300);
19  Serial.print("S");
20  digitalWrite(4, LOW);
21  delay(300);
22  digitalWrite(4, HIGH);
23  delay(800);
24  digitalWrite(4, LOW);
25  delay(800);
26  digitalWrite(4, HIGH);
27  delay(800);
28  digitalWrite(4, LOW);
29  delay(800);
30  digitalWrite(4, HIGH);
31  delay(800);
32  digitalWrite(4, LOW);
33  delay(800);
34  Serial.print("O");
35  digitalWrite(4, HIGH);
36  delay(300);
37  digitalWrite(4, LOW);
38  delay(300);
39  digitalWrite(4, HIGH);
40  delay(300);
41  digitalWrite(4, LOW);
42  delay(300);
```

```
43    digitalWrite(4, HIGH);
44    delay(300);
45    digitalWrite(4, LOW);
46    delay(300);
47    Serial.println(S);
48    }
49    }
```

4 Skjold Shields
I arduino-økosystemet er det mulig å utvide funksjonen av kortet med såkalte *skjold*. Disse kan man lage selv eller kjøpe på nett.

Finn et skjold for å spille av mp3 og beskriv hvordan det kan brukes.

5 nøtter
5.1 Om sikker overføring av data mellom to kort
Ben Eater om overføring av data: https://www.youtube.com/watch?v=eq5YpKHXJDM

Bruk to Arduinoer og en LCD-display for overføring av tekst mellom to Arduinoer.