

Alarm klokke

med ds1307 klokke og 1.8” tft skjerm

Torstein Alvern | PG5500 - Embedded Systems | 18.10.2017

# Introduksjon

Jeg startet å planlegge oppgaven da vi fikk en “sneak-peak” på oppgaven i timen. Tenkte meg ut hva jeg så for meg å få gjort, og hvilke komponenter jeg trengte/ønsket å koble til. I timen uken etter koblet vi opp TFT skjermen, og DS1307 klokka, og jeg hadde dermed de to komponentene ferdig koblet når timen var ferdig og jeg dro hjem for å jobbe med oppgaven.

# Valg av komponenter

Jeg ønsket å holde meg tils å få komponenter som mulig utover de fikk fikk beskjed om å bruke, men ikke så få at det blir problematisk å bruke alarmklokken. Jeg endte med to digitale knapper, og et potensiometer som analog input enhet. Ved å ha to knapper kan man velge om man skal stille alarmens time eller minutt først, som er en fordel gitt hvordan potensiometeret brukes til å stille tiden, og (hvertfall så lenge det er på breadboard stadiet) er nokså unøyaktig til tider.  
  
Jeg trengte også en buzzer så alarmen kan bruke lyd for å trekke oppmerksomhet, som er en fordel da den skal kunne vekke brukeren. (Skal den reelt kunne vekke brukeren og ikke bare vise funksjonen vil den nok trenge å være litt kraftigere).

Jeg valgte også å holde meg til Arduino Uno fremfor Particle Photon? da jeg allerede hadde koblet opp hovedkomponentene på Uno innen jeg kom til å tenke over valgmuligheten.

# Oppkobling og Fritzing

Oppkoblingen gikk veldig greit og veldig rett frem. Det vanskeligste ble gjort i timen da vi koblet opp TFT panelet med aksellerometer og DS1307 separat. Utover disse koblet jeg opp mine valgte komponenter veldig enkelt og greit. Brukte 10KΩ motstander til bryterne, og hadde tidligere en motstand koblet til buzzeren, men fjernet denne for å øke volumet.

I Fritzing fikk jeg koblet opp det meste, men kunne ikke finne komponentene for skjermen og klokka. Fant derfor en DS1307 som har tilkoblinger med samme navn som jeg har brukt, så jeg kunne vise koblingen slik den er tiltenkt, og limte inn TFT panelet ved hjelp av MS Paint. (TFT panelet er defor tilstede i .png bildet, men ikke i .fzz filen)

# Kode/Utfordringer

Jeg skrev mesteparten av koden fra bunnen av. Jeg brukte bibliotekene vi brukte i timen for TFT skjermen og for ds1307 klokka. (Adafruit\_GFX, Adafruit\_ST7735, SPI, Wire, RTCLib). Jeg hentet også pitches.h fra eksempelene for å kunne spille melodier.

Jeg startet med å slå sammen kodeeksemplene fra timen for skjermen og klokka slik at jeg kunne få dato og tidspunkt opp på skjermen. For å få tekst opp på skjermen brukte jeg graphicstest eksempelet for å se hvordan den håndterte tekst. Deretter utvidet jeg til å legge til en alarm med hardkodet tidspunkt som jeg implementerte funksjonalitet for slik at den kunne vise «alarm» på skjermen. Deretter kunne jeg utvide med funksjonalitet for å justere tidspunktet for alarmen ved hjelp av knapper og potensiometer. Når dette virket, koblet jeg til buzzer og lagde noen fine melodier til oppstart og selve alarmen. Jeg fant senere ut at jeg heller enn å ha en forhåndsdefinert melodi, ønsket å ha en melodi som forandrer seg for hver gang den spilles av, for å gjøre alarmen mer uforutsigbar, og slik at den fortsetter å ta oppmerksomhet når vanlige melodier hadde blitt gamle/bakgrunnsstøy.

Jeg implementerte så en snooze funksjon slik at man kan både skru alarmen av og utsette den i 5 minutter når alarmen går av. Jeg refaktorerte også flere ganger for å gjøre koden mer lesbar, og for å enklere finne bugs/feil jeg fikk rettet opp.

Fordi potensiometeret ikke er veldig nøyaktig, og verdiene flimrer frem og tilbake implementerte jeg en smoothing funksjon fra smoothing eksempelet for å gjøre det litt jevnere. Det å bruke gjennomsnittet over flere lesinger gir mer delay, så jeg prøvde å finne en balanse mellom hvor treigt det ble og hvor mye det løste flimringen.

# Referanser

* Adafruit\_GFX
* Adafruit\_ST7735
* SPI
* Wire
* RTCLib

(Uferdig) – Trenger å bli sett på og fikset