Embedded Eksamen.

Årsak:

I denne oppgaven var jeg først veldig i tvil om hvordan jeg skulle gå frem da jeg ikke var helt "gira" på at 80% av oppgaven skulle være "spill programmering". Når jeg først bestemte meg å gå for et mer interaktivt spill ble entusiasmen og tankene rundt det mye høyere.

Jeg var innom flere muligheter, men endte til slutt opp med å gå for det tradisjonelle «Wire game».

Selv om det er lett i etterkant, så klarte jeg ikke i begynnelsen helt å se hvordan jeg skulle få wiren til å registrere at den var borti holderen og hvordan jeg skulle koble dette sammen i en pakke med skjerm og lyd. Dette gjorde at jeg så muligheten til å øve på modulene fra studie, men også elektronikk og kobling på en litt "frihånd" måte, da var valget 100% bestemt.

Spillets funksjon:

Det starter med at buzzeren spiller en melodi mens skjermen viser et bilde fra Commander Keen. Når spiller trykker ok går spillet over til staten «IDLE» og en start meny vises med 3 valg, starte spillet, velge vanskelighetsgrad eller se på high-score listen.

- ➔ Start game
  - Buzzeren spiller melodi.
  - Skjermen viser nedteller.
  - Spiller fører holderen langs wiren og om den berører wiren kommer det lyd fra buzzeren og spiller mister ett liv.
  - Spiller er tom for liv -> spilles ny melodi og «Game over» vises på skjermen.
  - Spiller kommer i mål -> spilles ny melodi og skjermen viser tiden spiller har brukt og gir spiller valget om å lagre i high-score listen eller starte på ny.
  - Spiller velger å lagre score -> da vises "AAA" og spiller kan scrolle igjennom bokstaver til han har valgt sine initialer og ved siste bokstav lagres scoren i listen på riktig "palle plass" om den er bedre enn den nåværende sisteplassen.
- ➔ Difficulty
  - Spiller får valg mellom easy, medium og hard.
  - Easy = 3 liv, Medium = 2 liv, Hard = 1 liv.
- ➔ High Score
  - Ny melodi spilles av.
  - High score listen blir hentet fra EPROM til RTC'en i oppstart og oppdatert både i koden og på RTC'en om spiller velger å lagre score.

Problemstillinger:

KODE

Jeg fikk tilbakemelding om å sjekke ut pattern «State machine» i forrige arbeidskrav, jeg innså da at jeg "på en måte" hadde brukt dette, men på en veldig stygg og uoversiktlig måte, så jeg bestemte meg får å fokusere på å strukturere dette fra begynnelsen. Dette ble jeg veldig fornøyd med og gjorde det mye lettere og oversiktlig når jeg skulle bygge på koden.

Når jeg var "ferdig" ønsket jeg å dele prosjektet opp i flere filer slik at main ikke inneholdt alt og da blir koden mer oversiktlig. Jeg laget en ny branch og testet å legge ut til h og cpp filer «tft», «rtc», «tunes» og «highscore», men etter en liten dag med «multiple definitions» meldinger måtte jeg bruke tiden på andre ting. Globals/extern var vanskeligere jeg trodde.

THE problem

Den uten tvil største tid-slukeren for meg i dette prosjektet var å få lyden som jeg ville og henger veldig godt sammen med å forklare de aller fleste problem stillingene jeg gikk igjennom.

Oppsumering (forkortet):

- ➔ PCM biliotek som tok inn array av binære verdier
  - Formaterte lyd i riktig sample rate og unsigned 8 bit format.
  - Brukte alt for mye av minne på arduino så kunne ikke bruke det.
- ➔ TMRpcm bibliotek som kunne spille av lyd asynkront med "hoved tråd" fra minnekort.
  - Formaterte lyd på riktig måte og lastet inn som fil på SD kortet i skjermen.
  - Skjerm ble pixelert
    - Strøm problem? (nei)
    - Fant ut at TMR lib brukte en timer på pin 10 som jeg allerede brukte og da jeg endret dette i config filen samt satte på StopPlayback(),som stopper musikken og free'er filen som ble brukt for å spille, etter hver gang jeg endret «State». Funket skjermen og lyden.
- ➔ Koblet til RTC
  - Kode fungerer, men inkonsistente feil oppstår.
  - Ut-kommentere fuksjonen som kalibrerer rtc'en og alt funker som det skal.
  - Skal nå bruke EPROM i klokken til å lagre high-score listen og da blir lyden borte igjen.
  - Denne gangen kan jeg ikke ut-kommentere noe da dette er kritisk for at koden skal virke som jeg vil.
  - Her stuper jeg inn, med hode først, i en verden med å forstå timere, activt minne forbruk og noen helt utrolig inkonsistente feil.
  - Jeg begynner å vurdere å gå over til ESP 32, men er ikke helt klar enda.
  - Jeg finner ut at arduino har 3 timere , 2 på 8 bit og 1 på 16.
  - Bruker TMR biblioteket samme timer som RTC eller WIRE ?
  - Jeg finner ut at TMR biblioteket bruker timer 1 som er 16 bit

- Prøver å finne ut om de andre bibliotekene bruker denne, men klarer ikke se at de gjør det.
- Jeg finner ut at jeg i config kan endre TMR biblioteket til å bruke timer 2 istedenfor timer 1.
- Kan dette funke, antagelig med dårligere lyd, men tross alt så har jeg jo formatert lydfilen i unsigned 8 bit format?
- Jeg prøver dette, men funker dårlig.
- Kan det være aktivt forbruk av minne som går over tilgjengelig minne?
- Med tanke på at problemet oppsto også i begynnelsen når jeg bare hadde kalibrering av klokken og mye ledig minne så var jeg litt i tvil på om dette, i hvert fall alene, var problemet.
- Etter utrolig mye av en blanding av å rive i håret og banke hode i bordet, veggen, taket og gulvet, for å prøve å forstå noe som antageligvis er over "my paygrade", måtte jeg til slutt innse at jeg får prøve noe nytt.

➔ Over til ESP 32

- Dette gikk overaskende fint og jeg fikk samtidig ryddet opp i oppkoblingen drastisk.
- Som jeg fryktet virket ikke TMRpcm bilioteket på ESP32 ☹️ .
- Jeg søkte etter bibliotek som kunne gjøre det samme, spille av wav fil i et visst format fra SD kort asynkront i med ESP32.
- Etter en stund måtte jeg bite i det sure eplet og "skrote" hele fremgangsmåten min med lyd.
- Jeg fant et nytt bibliotek som kunne spille av lyd med hjelp av et annet lydformat, RTTTL.
- Dette er et tekst format som bruker utrolig mye mindre minne for å lagre/overføre melodier og jeg kunne da lagre det i koden.
- Dette var overaskende greit å bruke og konverteringen i koden min fra TMRpcm til NonBlockingRTTTL gikk veldig bra.
- Selv om det var surt å gi opp mitt "theme" som var Commander Keen (DOS) så ble jeg fornøyd med resultatet.
- Jeg prøvde meg frem med motstand og fant ut at en 68 Ohms var perfekt for å få den beste lyden i buzzeren.

➔ SD KORT

- Siden jeg gikk over til å lagre tonene i koden, brukte jeg ikke sd kortet til noe lenger og valgte da at et bilde kunne være fint med tanke på tiden jeg har og samtidig fikk sett hvordan dette kunne gjøres.
- Jeg fikk da også litt av viljen min med tema, da jeg fant en .bmp fil av Commander keen! 😊
- Jeg reduserte størrelsen og endret til 24 bits farge format, la til en ny «state» og alt funket som det skulle. Dette var ganske mye "bibliotek magi" for å være ærlig, men jeg ble fornøyd med resultatet.

REFERANSER

- ➔ Button (debounce effect)
  - https://roboticsbackend.com/arduino-push-button-tutorial/
- ➔ TFT
  - https://learn.adafruit.com/adafruit-1-44-color-tft-with-micro-sd-socket/
  - https://learn.adafruit.com/adafruit-1-14-240x135-color-tft-breakout/pinouts
- ➔ RTC
  - Minne Forelesning Dag 8 side 83.
- ➔ BUZZER
  - rtttl_demo example code from "NonBlockingRTTTL by Antoine Beauchamp"
  - https://github.com/end2endzone/NonBlockingRTTTL
- ➔ SD
  - Adafruit ImageReader Library <BreakoutST7789> eksempel kode.
  - https://learn.adafruit.com/adafruit-gfx-graphics-library/loading-images
- ➔ LYD
  - http://arcadetones.emuunlim.com/
- ➔ Bilde
  - https://int10h.org/blog/2016/05/dopefish-goes-ntsc-commander-keen-4/

BIBLIOTEKER

- ➔ <Arduino.h>
- ➔ <Adafruit_ImageReader.h>
- ➔ <Adafruit_ST7789.h>
- ➔ <RtcDS3231.h>
- ➔ <Wire.h>
- ➔ <NonBlockingRtttl.h>
- ➔ <SdFat.h>
- ➔ <SPI.h>

Brainstorming notater:

Jeg trykker på en knapp for å starte spillet, skjermen viser animasjon og ender opp med å velge vanskelighetsgrad. Skjermen viser animasjon «Ready, set go», timer starter og spillet begynner.

Er jeg borti wiren kommer det lyd fra buzzeren + shock animasjon på skjermen og spillet er over. (alternativt lagre som poeng)

"Game over" animasjon vises og jeg får spørsmål om å lagre scoren min.

Skjermen viser nå «high-score» tabellen og spiller «play game» lyd.

- ➔ Wire game
  - TFT
    - Show time spent.
    - Time left?
    - High score table
  - Difficulty
    - Moving wire with servo or motor.
  - Sound
    - Beep sound when touching the wire.
    - StartGame sound
    - Game over sound
    - New record sound
  - Handle
    - Battery.
    - IR sender sending signal, can it then be wireless handle and manageable with 1 MC?.
    - 3d printed handle?