The Adventures of O

ARDUINO UNO SPILLSYSTEM MED DS1307 KLOKKE, 1.8” TFT SKJERM, SD-KORT LAGRING, HÅNDKONTROLLER, OG LYD

Torstein Alvern | PG5500 - Embedded Systems | 26.11.2017

Introduksjon

Jeg ønsket å starte på oppgaven så fort vi hadde fått den, men grunnet komplikasjoner med skolen og innleveringer i andre fag som mer tidskritiske ble denne utsatt til siste uken. Jeg hadde originalt et ønske om å bruke sensorer som den ultrasoniske dybdesensoren som spill-input, men valgte å gå vekk fra dette senere i utviklingen. Jeg baserte meg i stor grad på koblingsskjemaet fra min andre innlevering, og det gikk raskt å komme i gang, og å få noe opp på skjermen.

Jeg ønsket å gå for noe helhetlig, og valgte derfor å lage et spillsystem med mer enn ett spill, og en tittel som gjør at alt føles som en naturlig pakke. Basert på at figuren man beveger rundt er karakteren 'o' og dette var spill av typen «hold deg i live lengst mulig», så virket tittelen på systemet «The Adventures of O» både kul og passende.

Tittelen på spillene i systemet er basert på hvordan man spiller de respektive spillene og burde være ganske rett frem. Jeg gikk gjennom litt forskjellige titler på tavla for å komme frem til et noen jeg likte godt.

README.md inneholder informasjon om hvordan man spiller spillet.

Valg av komponenter

Jeg startet med utgangspunkt i oppgaveteksten som ved denne oppgaven krevde bruk av en Arduino Uno, en TFT-skjerm og SD-kort. Utover dette valgte jeg å legge til RTC klokken slik at jeg kunne bruke denne for å øke poengsummen og lage et spill hvor poengene er basert på hvor lenge man holder seg i live, som er et godt utgangspunkt for et spill man ønsker å spille flere ganger da man stadig kan utfordre sin egen, eller venners beste poengsum.

Jeg koblet også til en buzzer for å gi spillsystemet lyd og gjøre spillopplevelsen bedre. Denne ble satt inn uten motstand grunnet at den spiller av nokså lav lyd, og jeg puttet en liten klebrig papirbit over den for å forsterke lyden ytterligere.

Jeg ønsker å ha en håndkontroller for spillsystemet, og originalt bestod denne av en styrespak med knapp på et separat breadboard sammen med en ultrasonisk sensor, men da dette ble svært ustabilt grunnet oppkoblinger med dårlige ledninger valgte jeg å gå for noe enklere, og forholdt meg kun til styrespaken med knapp. Dette ble også en langt mer behagelig håndkontroller, og føles intuitiv å bruke.

Oppkobling og Fritzing

Oppkoblingen gikk veldig greit og veldig rett frem med unntak av joysticken. Jeg baserte meg på i utgangspunktet på koblingsskjemaet mitt fra andre innlevering, hvor jeg hadde koblet opp TFT panelet og RTC klokken. Da trengte jeg kun å legge til buzzer, og joysticken, som ikke lot seg virke med pull-down på knappens motstand, og jeg ble nødt å sette 10KΩ motstanden til 5V istedenfor jord for at det skulle virke med både de to analoge signalene og den digitale knappen. Jeg hadde også vært litt dristig og prøvd meg på å forlenge ledninger, som gjorde at kontakten ble nokså ustabil, og jeg måtte finne på noe annet.

Da jeg skulle koble opp SD-kortet kom jeg over et problem som viste seg å være en defekt på breadboardet, som var grunnlaget for at jeg valgte å putte TFT-skjermen på et separat breadboard. Det gjorde også at jeg kunne peke skjermen litt dit jeg ville for en mer behagelig spillopplevelse, og var litt av motivasjonen for å koble på nytt som nevnt i neste avsnitt.

Da jeg var sikker på at jeg hadde alle komponentene på plass og slik jeg ønsket koblet jeg alt sammen på nytt, men denne gangen med ledningene mer sortert så det var langt enklere å ha oversikt over hvor de forskjellige ledningene går, og hvilke ledninger som hører sammen.

I Fritzing fikk jeg koblet opp det meste, men kunne ikke finne komponentene for skjermen, joysticken, 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 derfor tilstede i .png bildet, men ikke i .fzz filen). Jeg skrev også et lite notat under Joystick-komponenten jeg brukte som viser hvordan komponenten er koblet, siden dette ikke stemmer med hva tilkoblingene heter på bildet.

Kode/Utfordringer

Jeg skrev mesteparten av koden fra bunnen av, men deler av koden jeg startet med var skrevet til forrige innlevering. Jeg brukte bibliotekene vi ble introdusert til i timen for TFT skjermen, SD-kortet, og for ds1307 klokka. (Adafruit_GFX, Adafruit_ST7735, SPI, Wire, RTCLib, SD). Jeg hentet også pitches.h fra eksempelene for å kunne spille melodier.

Fordi jeg kunne hente kode fra forrige innlevering, kom jeg raskt i gang med å få skjermen til å vise tekst. Jeg satte deretter opp en state machine for å kunne bytte mellom en hovedmeny og spillene mine. Jeg gikk så i gang med å få første spillet til å virke, og skrev

metoder for å få karakteren til å bevege seg, og for å få et objekt til å «falle fra taket», og metoder for å sjekke om spilleren traff dette objektet. Da spillet var spillbart la jeg inn muligheten til å kunne pause, og gjenbrukte logikk fra hovedmenyen for å kunne gjøre valg i denne menyen.

Jeg tok så å la inn animasjoner for oppstart, for første spillet, og for andre spillet. Deretter brukte jeg litt tid på å gjøre menyene penere, velge forskjellige farger, endre tekststørrelser, og legge på avrundede firkanter. Da dette var lagt inn hadde jeg de fleste tekst-strengene jeg trengte i spillsystemet, og flyttet disse til Flash minne (PROGMEM) for å spare plass på SRAM.

Deretter ønsket jeg å legge til muligheten til å lagre poengsummen fra spillet, og skrev metoder for håndtering av lagring. Jeg prøvde meg også på en egen high-score skjerm, som leste av verdiene på SD kortet, sorterte de, slettet de dårligere verdiene, og viste frem de beste poengene for hvert spill. Etter å ha brukt noen timer på dette valgte jeg å holde meg til kun lagring av poengsum da dette raskt ble veldig stort dersom det skulle være «bra nok» for den spillopplevelsen jeg hadde satt opp for resten av systemet. Det å ha med denne lese-og-sortere-funksjonen gjorde også alt av SD-kort håndtering langt mer ustabilt, som blant annet gjorde at jeg i ganske stor grad fikk feilmeldinger ved lagring, og fikk en del andre interessante visuelle bugs hvor minnehåndteringen gikk litt skeis, og det å øke stabiliteten til lagringen var en medvirkende faktor for å fjerne denne funksjonen.

Det andre spillet jeg la inn kunne gjenbruke store deler av koden direkte fra det første spillet, og det tok dermed svært lite ekstra plass. Det jeg trengte å lage på nytt kunne baseres i stor grad på det jeg hadde fra før, med noen justeringer. Jeg tok så å la inn et litt mer avansert poengsystem enn bare å telle sekundene. Jeg gjorde poengsystemet slik at det oppdaterte hvert sekund, men hvor mange poeng man tjente var basert på hvor risikabelt man spilte. Dette gir gode insentiver til å gjøre spillet vanskeligere for seg selv, og gjør at man har mer lyst til å spille igjen da man kan utforske andre taktikker og spillestiler for å øke poengsummen sin.

Da jeg hadde fått begge spillene til å virke valgte jeg å legge til lyd da dette øker spillopplevelsen. Jeg skrev en liten melodi for hovedmenyen, som kunne gå i loop her, og en kort melodi som spiller av når man taper et av spillene. Jeg holdt meg til lavere frekvenser for å bruke mindre ressurser i form av mindre variabeltyper.

GitHub ble brukt som versjonskontrollsystem, og tillot meg å jobbe fra stasjonær PC da jeg var hjemme og fra laptop da jeg ikke var det. Det tillot meg også å teste ting på egne branches uten å være redd for at det var et blindspor.

https://github.com/TorsteinA/EmbeddedSystems_Exam (repoet er privat da jeg har hørt rykter om at public repo kan regnes som juks. Ta kontakt med meg dersom du ønsker tilgang)

Referanser

- Adafruit_GFX: https://github.com/adafruit/Adafruit-GFX-Library
- Adafruit_ST7735: https://github.com/adafruit/Adafruit-ST7735-Library
- SPI: https://www.arduino.cc/en/pmwiki.php?n=Reference/SPI
- Wire: https://www.arduino.cc/en/Reference/Wire
- RTCLib: https://github.com/adafruit/RTClib
- SD: https://github.com/adafruit/SD