

**Watch-out!**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Mikrodatorprojekt – TSIU51 – Survival-spel – 2022-02-05**

**Grupp 5**

Emil Pihl  
Kebba Jeng  
Aidin Jamshidi  
Martin Castro Bildhjerd

**Innehållsförteckning**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**0.0 Spelomslag 1**

**0.1** **Projektinformation** **2**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**1.0** **Inledning** **5**

**1.1** **Bakgrund kring projektet** **5**

**1.2** **Grundlig beskrivning av spelet** **5**

**1.3** **Kravspecifikation** **6**

**1.4** **Blockschema** **7**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**2.0** **Översikt** **7**

**2.1** **Övergripande beskrivning av spelet** **8**

**2.2** **Komponenter** **8**

**2.3** **Organisering av projektarbete** **90**

**2.4** **Beskrivning av hårdvara & protokoll** **99**

**2.4.1** **DAvid-kort** **99**

**2.4.2** **Arduino Uno** **99**

**2.4.3** **DAmatrix** **99**

**2.4.4** **Piezoelektrisk högtalare** **99**

**2.4.5** **Spelknappar** **99**

**2.4.6** **LCD-display** **99**

**2.4.7** **TWI** **99**

**2.4.8** **SPI** **99**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**3.0** **Framgångar** **99**

**3.1** **Motgångar** **99**

**3.2** **Saker som skulle ha gjorts annorlunda** **99**

**3.3** **Diskussion** **99**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**4.0** **Referenser** **99**

**Figurförteckning**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Figur **0.0 – Watch-out! 0**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Figur **1.2 – Spelexempel 4**

Figur **1.3 – Förslag 7**

Figur **1.4 – Blockschema 8**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Figur **2.3 – Planner 33** Figur **2.4.1 – DAvid-kort 33** Figur **2.4.2 – Arduino Uno 34**

Figur **2.4.2 – I/O-portar 24**

Figur **2.4.30 – Pixel position 45**

Figur **2.4.31 – DAmatrix 45**

Figur **2.4.4 – Högtalare 34**

Figur **2.4.5 – Tryckknappar 45**

Figur **2.4.6 – New game 45**

Figur **2.4.7 – TWI 24**

Figur **2.4.8 – SPI 33**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Figur **2.3 – Kopplingsschema xxx**

Figur **2.3 – Kopplingsschema xxx**

Figur **2.3 – Kopplingsschema xxx**

Figur **2.3 – Kopplingsschema xxx**

Figur **2.3 – Kopplingsschema xxx**

Figur **2.3 – Kopplingsschema xxx**

Figur **2.3 – Kopplingsschema xxx**

**1.0 - Inledning**

Detta spel skapades som ett projekt för kursen mikrodatorprojekt TSIU51 som undervisas på Linköpings universitet för högskoleingenjör inom elektronik. Gruppen som skapade detta projekt består av medlemmarna Emil Pihl, Kebba Jeng, Aidin Jamshidi och Martin Castro Bildhjerd, som utgör grupp fem.

Watch-out! är ett spelprojekt skapat av grupp fem. Projektet är inspirerat av spelet som kan spelas på Googles webbläsare när internetåtkomst saknas. I spelet ska saker undvikas för att överleva och samla ihop poäng. Till spelet har olika rutiner skrivits åt hårdvaran för att skapa ett beroende till spelet genom saker som ljud, blinkande färger eller information på skärmarna.

**1.1 – Bakgrund kring projektet**

När projektet påbörjades spenderades mycket tid på att fundera ut hur spelet skulle fungera. Det som bestämdes kvickt var att spelet skulle både vara roligt och ha en viss svårighetsgrad. Genom inspiration från de symboliska och legendariska 90-tals spelen kunde gruppen konstatera att spel med perfekt användarupplevelse inte behöver omfatta invecklade funktioner eller avancerad grafik.

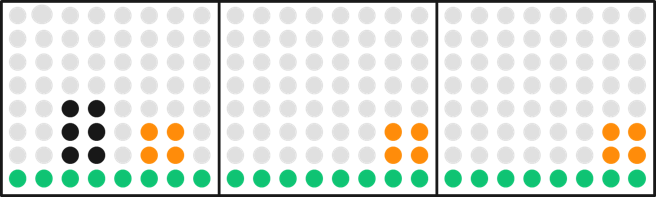
Ett spel som har lyckats fånga beroendeframkallande delar av ett spel är Tetris och många andra som inte går att nämna. Genom denna uppfattning kunde gruppen gå vidare i att bestämma vilket typ av spel som skulle skapas för projektet.

Gruppen bestämde sig tidigt att spelet skulle innehålla en karaktär som styrs av spelaren, där personens uppgift är att undvika objekt. Det diskuterades mycket kring ifall karaktären rör sig mot sidan eller ifall objekten skulle röra sig mot spelaren men beslutades i att karaktären skulle stå stilla och objekten rör sig. Där spelaren ges möjligheten att hoppa eller ducka för inkommande objekt.

**1.2 - Grundlig beskrivning av spelet**

Spelet är av en simpel natur, där spelaren ska undvika flygande objekt och stenar för att försöka att samla på poäng. Karaktären i spelet kommer visa sig på vänster sida av spelplan och från höger sida av spelplan dyker objekt upp som måste undvikas. Karaktären står stilla och objekt rör sig emot den för att ge illusionen av att karaktären rör sig.

Stundvis behöver spelaren hoppa för att undvika objekt på backen och även ducka för att inte bli träffad av flygande objekt. Spelaren har bara möjlighet att hoppa eller ducka, därav används två knappar till spelet.

****

**Figur 1.2 - Spelexempel:** *På figuren syns ett exempel på hur ett objekt närmar sig karaktären som styrs av spelaren.*

**1.3 - Kravspecifikation**

I starten av projektet skall en kravspecifikation skrivas som innehåller viss information angående slutprodukten. Först bestämdes hur spelet ska utspela sig och därefter komponenter som kan användas till detta. När en grund hade blivit lagd så byggdes det ut på idén. Därefter gjordes en lista med ”Skall-krav:” och ”Bör-krav:”.

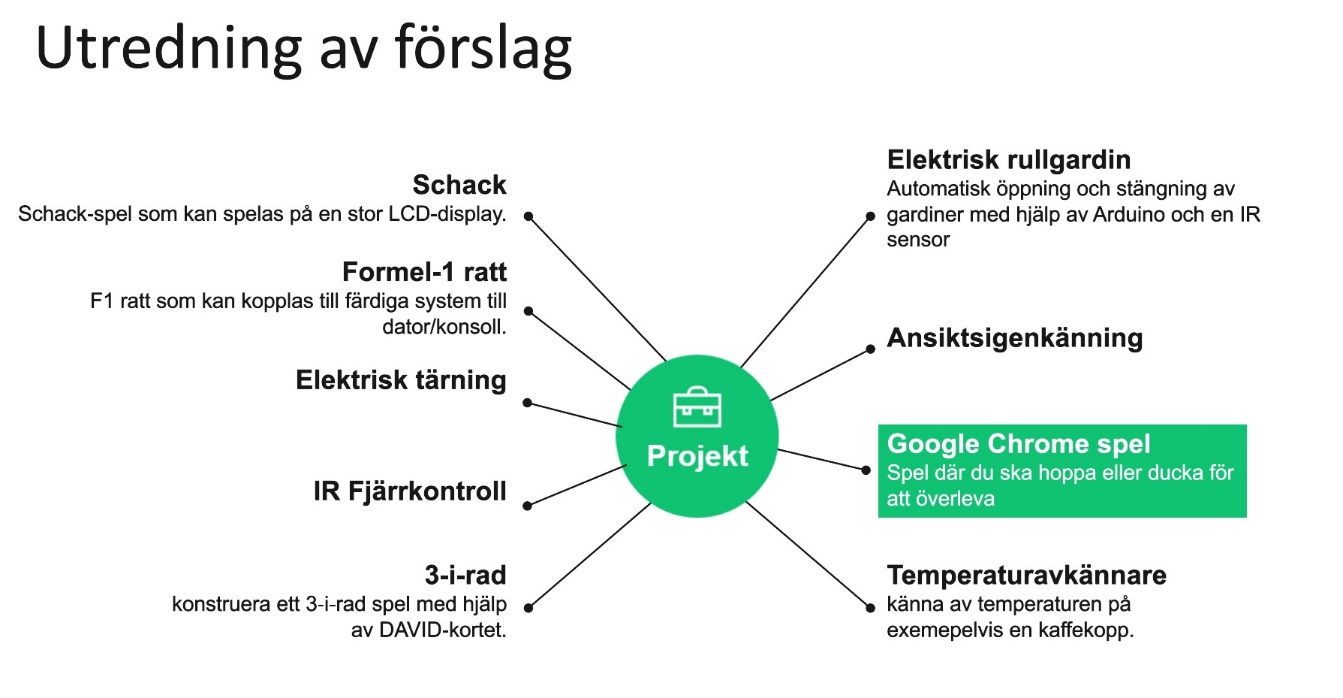
Efter mycket tankar och idéer kring hur spelet skulle fungera så skrevs allting upp på en tavla, vilket sedan gick över till en röstning. Under röstning togs varje idé upp och diskuterades ifall det skulle fungera och efter mycket tid fastställdes den. Resultatet kan ses nedan.

**Skall-krav:**

* Spelaren **skall** vara en figur som liknar en dinosaurie.
* Spelaren **skall** kunna hoppa med figuren.
* Spelaren **skall** röra sig åt höger automatiskt med figuren.
* Spelaren **skall** kunna samla poäng.
* Spelaren **skall** kunna förlora spelet genom att förlora alla hjärtan.
* Spelaren **skall** kunna spela med knappar på DAVID-kortet.
* Det **skall** låta när spelaren startar spelet.
* Det **skall** låta när spelaren förlorar.
* Det **skall** finnas en huvudmeny.
* Det **skall** finnas en poäng “bild” när spelaren förlorar.
* Det **skall** finnas två DAmatrix.

**Bör-krav:**

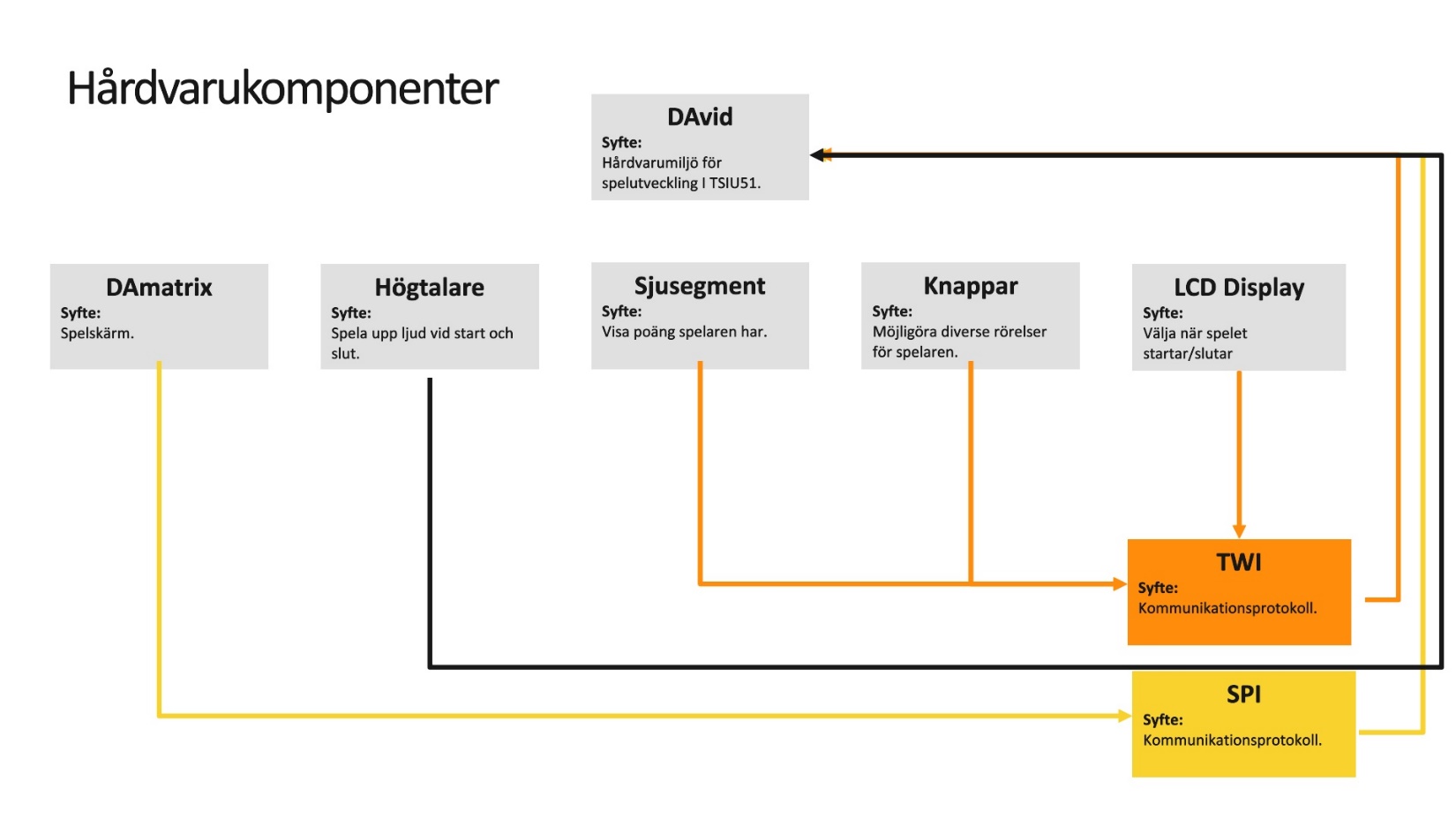
* Spelaren **bör** kunna ducka med figuren.
* Spelaren **bör** kunna samla poäng genom olika metoder.
* Spelaren **bör** kunna samla på sig upp till tre hjärtan.
* Spelaren **bör** kunna spela med joystick.
* Det **bör** finnas tre DAmatrix.
* Det **bör** finnas flygande föremål som spelare måste undvika.
* Det **bör** spelas upp ett ljud när man samlar på sig poäng
* Det **bör** spelas upp ett ljud när man samlar på sig ett hjärta.
* Det **bör** spelas upp ett ljud när förlorar ett hjärta.
* Det **bör** gå att starta om spelet från “förlora-bilden” utan att starta om kortet.
* Spelets hastighet **bör** öka för en ökad svårighetgrad.

****

**Figur 1.3 - Förslag:** *Exempel på projektidéer och det slutgiltiga markerat med grön färg.*

**1.4 - Blockschema**

Blockschemat nedan visar vilken hårdvara som används i projektet och även tillhörande rutiner till dessa hårdvaror. Vissa rutiner kunde användas mellan flera delar av hårdvaran medan viss hårdvara har helt egna rutiner. Anledningen till detta var för att inte skapa kod som är svårläslig, detta för att få ett strukturerat sätt att läsa koden för att förenkla förståendet kring hur allt binds samman.



**Figur 1.4 - Blockschema:** *Hårdvarukortet DAvid. Framtaget för projektkurser i Mikrodatorteknik. Hårdvarumiljön består utav diverse komponenter som LCD-display, LEDs, tryckknappar, med mera.*

**2.0 – Översikt**

Med tanke på hur många komponenter som skulle kunna förbättra användarupplevelsen både funktionellt och estetiskt, valde gruppen att ta en mellanväg som tänktes inte vara för överkomplicerad men som ändå ger en god användarupplevelse, likt dem klassiska 90-tals spelen.

Genom en hel del diskussion via dator och även på plats så bestämdes vilka delar av DAvid-kortet som skulle tänkas vara användbara för projektet. Skärmen bestämdes till tre DAmatrix-skärmar horisontellt placerade för att ge en överblick över spelplanen, för att kunna reagera på inkommande objekt. Detta för att spelaren ska få en maximal spelupplevelse och inte frustrerad då personer inte hinner reagera på objekt.

Högtalaren används till att spela olika typer av små pip-ljud för att indikera olika stadier i spelet. När spelet startas spelas ett specifikt ljud som ger spelaren feedback om att spelet har startat, detta sker även när spelet återställs genom DAvid-kortet. När spelaren förlorar ska ett annat specifikt ljud spelas upp för att indikera att spelaren har förlorat sin omgång.

För att ge spelaren möjlighet till att röra sig ges knapparna på hårdvarukortet. Det blir ganska simpelt för spelaren att förstå kontrollerna då det endast blir två knappar för hoppa och ducka. För att hjälpa spelaren förstå

vad som sker i spelet används en LCD-display för att förmedla information till spelaren för att förklara ifall det är ett nytt spel eller avslutat spel. Anledning till detta då det går snabbt och lätt att informera spelaren om vad som sker.

En LCD-display finns även på kortet som används för att förmedla information som när spelaren har startat ett nytt spel eller ifall den har förlorat, genom att visa spelarens poäng. Vid omstart av ett nytt spel så ska skärmen varna någon sekund innan allt börjar röra sig. Förlorar spelaren ska det synas på skärmen med ett meddelande som sedan visar poängen som har samlats ihop.

På hårdvarukortet finns en RGB-LED slinga som blir använd för att ge spelaren feedback på vad som sker i spelet. Detta genom att lysa olika färger beroende på spelstadie. Detta genom att förmedla liv på slingan, eller ifall spelaren har förlorat byta färg till rött.

**2.1 - Övergripande beskrivning av spelet**

När spelaren startar eller omstartar spelet ska en melodi spelas för att visa att spelet snart startar. På LCD-displayen visas även information om vilka knappar som ska användas av spelaren för att hoppa eller ducka. När spelaren förlorar ska en annan typ av melodi spelas och även informationen på LCD-displayen ska ändras. Efter några sekunder ska poängen som har samlats ihop visas på skärmen.

När spelet påbörjas börjar poäng samlas ihop sakta men säkert. Efter ett tag börjar objekt dyka upp på DAmatrix-skärmen längst till höger som sakta börjar röra sig emot karaktären som spelaren styr. Är det ett flygande objekt måste spelaren ducka, eller ifall det är en sten måste spelaren i stället hoppa.

Desto längre spelaren klarar sig desto mer poäng samlas ihop, dock ökas svårighetsgraden på spelet med tiden. Detta för att skapa lite mer av ett beroende till spelet så att inte är liknande mönster under hela spelets gång. Anledningen till detta är för att kopiera det som gjorde de gamla spelen roliga.

**2.2 - Komponenter**

**1x** DAvid-kort **(Mikrokontroller)**

**1x** Arduino Uno **(Processor)**

**3x** DAmatrix 8x8 **(Skärm 8x24)**

**1x** Piezoelektrisk högtalare **(Högtalare)**

**2x** Spelknappar **(Joystick)**

**1x** LCD-display **(Skärm)**

**1x** 3D-printad stativ för DAmatrix **(Stativ)**

**2.3 - Organisering av projektarbete**

Organiseringen av arbetet gick till genom att ha två möten i veckan. Ett som var på söndagen genom Discord och ett annat möte som var på plats varje tisdag för att se hur projektarbetet låg till. Under varje möte diskuterades det kring hur varje medlem låg till med sitt arbete, eller ifall personen var klar. Ifall personen var klar med sitt arbete så påbörjades nästa arbete.

Något som användes under projektets gång var Microsoft planner där arbetet kunde planeras och sedan kunde tillägnas åt en medlem, därav visste personen vad som skulle göras näst.

En bild som visar text

Automatiskt genererad beskrivning

**Figur 2.3 – Planner:** *Olika uppgifter i Microsoft planner där de rankas under olika kategorier med specifika medlemmar tilldelade.*

**2.4 - Beskrivning av hårdvara & protokoll**

**2.4.1 - DAvid-kort**

DAvid-kortet är ett hårdvarukort utvecklat för kursen Mikrodatorprojekt (TSIU51) och detta kort används under projektets gång. Det är en hårdvarumiljö med diverse komponenter för att möjliggöra strukturerad mjukvaruutveckling, vars hårdvarumiljö lämpar sig för exempelvis spel. Kortet styrs av en Arduino Uno med processorn ATMega328p.

Graphical user interface, diagram

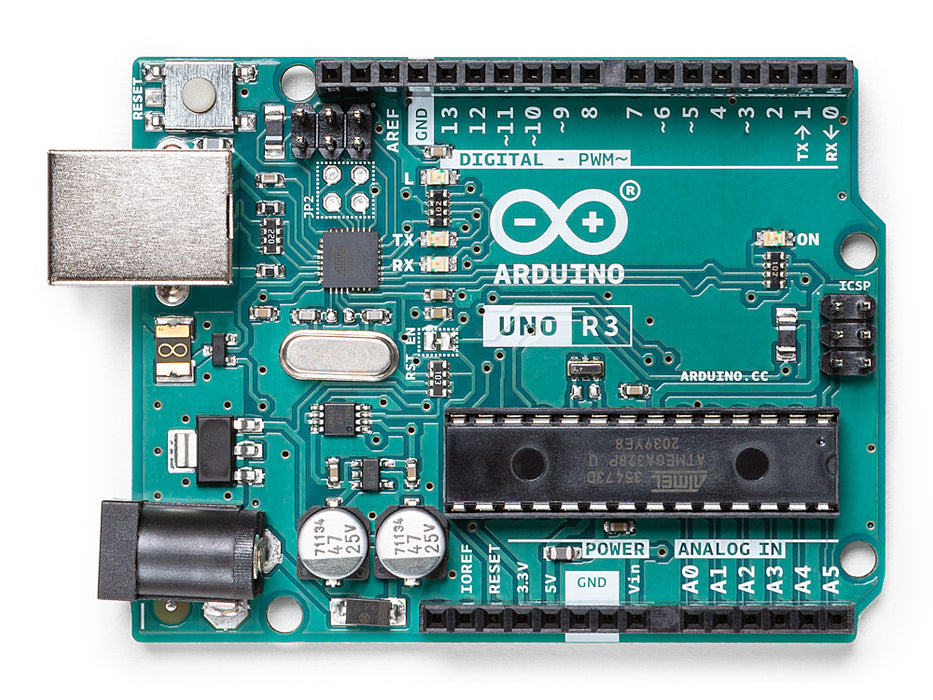
Description automatically generated

**Figur 2.4.1 – DAvid-kort:** *Hårdvarukortet DAvid. Framtaget för projekt i kursen Mikrodatorprojekt. Hårdvarumiljön består utav diverse komponenter som LCD-display, LEDs, tryckknappar, med mera.*

**2.4.2 - Arduino Uno**

Arduino Uno är ett mikrokontrollerkort[X] baserat på processorn ATMega328p[X]. En mikrokontroll är en mindre dator med CPU, arbetsminne, och programminne integrerat komplett med stödfunktioner på en enda interagerad elektronisk krets.

Mikronkontrollerkortet är utrustat med digitala och analoga I/O[X] pinnar som kan användas för att kommunicera med andra komponenter och kretsar.



**Figur 2.4.2 – Arduino Uno:** *Överblick av mikrokontrollerkortet Arduino Uno.*

ATMega328p är en mikrokontroll från Atmel. ATMega328p har ett 32 kbyte minne och en 16 MHz klockfrekvens, detta gör att mikrokontrollerna kan utföra upp till 16 miljoner instruktioner per sekund [x].

Det finns ett flertal I/O portar på ATMega328p, detta ger möjlighet att ansluta extern hårdvara som inte finns i DAvid-kortets hårdvarumiljö, exempelvis DAmatrix [x]

A picture containing table

Description automatically generated

**Figur 2.4.2 – I/O-portar:** Överblick på tillgängliga I/O portar på Arduino Uno.

**2.4.3 - DAmatrix**

DAmatrix är ett block som består av 64 lysdioder (LED) i en matris (8x8). Genom att använda tre DAmatrix så kan en större (8x24) spelplan skapas.

I detta projekt används tre DAmatrix för att få en spelplan på 8x24.

DAmatrix drivs med det seriella gränssnittet SPI [X] via Arduino UNO:s I/O pinnar.

DAmatrix är byggt på ett sätt så att

* Dioder i varje rad i matrisen hänger ihop i sina anoder.
* Dioder av samma sort och kolumn har sina katoder sammankopplade.

En bild som visar bord

Automatiskt genererad beskrivning

**Figur 2.4.30 – Pixel position:** *Exempelbild på hur dioder kan tändas separat på DAmatrix genom beskrivningen ovan.*

Med den infon tillhands så behöver alltså raden tillförses med positiv spänning, samtidigt som diodens katod jordas [x] för att tända en diod.

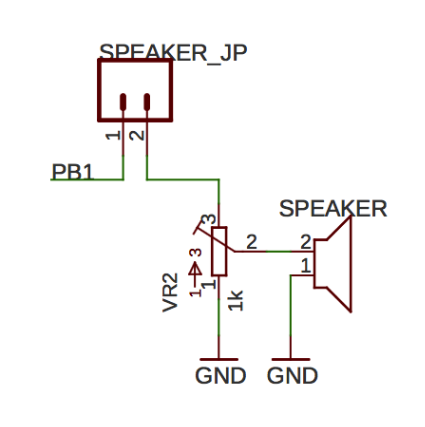
A picture containing text, receipt

Description automatically generated

**Figur 2.4.31 – DAmatrix:** *Kopplingsschema för DAmatrix. Bilden är omvänd för att förenkla tydning av dioder.*

**2.4.4 - Piezoelektrisk högtalare**

På hårdvarukortet DAvid finns det en piezoelektrisk högtalare som kan användas. Högst effektivitet uppnås vid frekvenser runt 3000–4000 Hz. Högtalaren är passiv, därav behövs matningsspänning vilket gör att högtalaren bara behöver en låg och en hög puls för att låta.



**Figur 2.4.4 – Högtalare:** *Kopplingsschema för piezoelektriska högtalaren.*

Genom att skapa en rutin som skickar en puls sådan att högtalaren varierar från låg till hög så kan ett ljud skapas. Genom rutiner som skickar frekvenser i varierande storlek kan simpla melodier eller ljud skapas. Via en potentiometer kan ljudstyrkan ändras.

**2.4.5 – Spelknappar**

I projektet används tryckknappar. Trycknapparna används för att utlösa [se över namn] de animationer i spelet som resulterar i att spelaren undviker hinder.

Gruppen har valt att använda trycknappar över [informellt, ändra] exempelvis joysticks då det ur ett programmeringsperspektiv är lättare att addressera.

Trycknapparna är bunda till en specifik port. Porten är i sitt standbyläge satt som låg[x]. För att addressera knapparna behövs således en positiv insignal. Gruppen har med den informationen tillhanda skrivit en kod med instruktioner som kollar status på porten som knapparna är bundna till.

**En bild som visar text, enhet, recept

Automatiskt genererad beskrivning**

**Figur 2.4.5 – Tryckknappar:** *Kopplingsschema för tryckknapparna. I detta fall används tryckknapp L och R.*

**p**

**2.4.6 - LCD-display**

För projektet så behövdes en metod för att förmedla spelaren ifall omgången var förlorad eller ifall en ny omgång höll på att starta. Detta kunde LCD-displayen användas till genom att förmedla information så spelaren vet vad som sker. Detta gjordes genom två rutiner som separat användes sig av TWI protokollet för att visa text på skärmen.

****

**Figur 2.4.6 – New game:** *Skärmen som visas för spelaren innan spelet startar.*

**2.4.7 – TWI**

Two-Wire Interface, förkortat TWI är ett synkront seriellt kommunikationsprotokoll. Ett vanligt användnings område är i hårdvara där tillverkningskostnad prioriteras över prestanda.

TWI ger hårdvara, system, och processer ett kommunikationsgränsnitt med en buss som består av två stycken ledningar, Serial Data (SDA), och Serial Clock (SCL).

SDA är dataledningen, där överförs data mellan en Master och Slave. SCL är klockledningen

TWI kommunicerar i duplex med en master-slave arikitektur, det vill säga att kommunikation sker samtidigt i två riktningar simultant. Master-slave arkitekturen möjligör en assymetrisk kommunikation där en en enhet eller process styr och kontrollerar andra enheter och processer.

För att överföra data och kommunicera mellan noder så behöver protokollet konfigureras. Detta görs genom att modifiera binära bitar i TWI:s olika register.

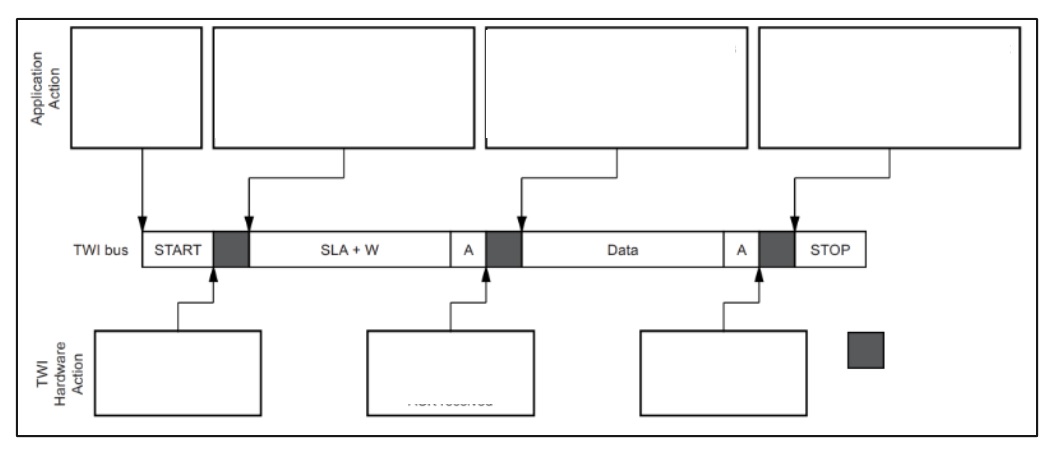
I detta projekt konfigurerades följande register:

* TWI Bit Rate Register, detta register kontrollerar klockledningens period.
* TWI Control Register, detta register kontrollerar TWI operationer, exempelvis avbrott. Registret används även för att generera START, STOP, och ACK pulser.
* TWI Data Register, registret kan sättas i olika lägen. I överförningsläge skickar den data. I mottagningsläge, sparas datan som mottagits.

En TWI dataöverföring består av

* Ett start vilkor
* En address och information om Read/Write samt Slave Acknowledge
* Data
* Ett stopp vilkor

Figuren nedan illustrerar TWI processen.



**Figur 2.4.7 – TWI:** *Exempel på hur information skickas genom TWI-protokollet.*

**2.4.8 - SPI**

Serial Peripheral Interface, förkortat SPI är ett synkront seriellt kommunikationsprotokoll som huvudsakligen används för kort-distans kommunikation i inbyggda system.

SPI kommunicerar i full-duplex med en master-slave arkitektur, det vill säga att kommunikation sker samtidigt i två eller fler riktningar. Master-slave arkitekturen möjligör en assymetrisk kommunikation där en en enhet eller process styr och kontrollerar andra enheter och processer.

SPI förbinds av en buss som består av fyra gemensamma ledningar. Serial Clock Output (SCLK), Master Out Slave In (MOSI), Master In Slave Out (MISO), och Slave Select (SS).

SCLK är som namnet indikerar klockledningen. Data som skickas genom MOSI och MISO synkroniseras till klockan som huvudnoden genererar.

MOSI och MISO är dataledningarna. MOSI överför data från huvudnoden till subnoden, MISO överför data från subnoden till huvudnoden. Protokollet kan göra detta simultant.

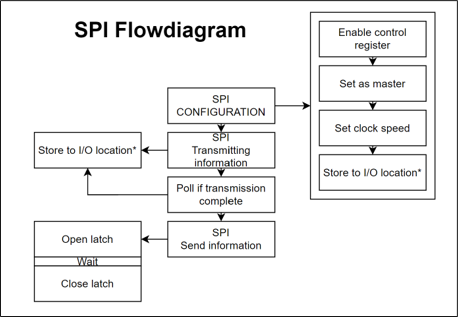
SS berättar vad för enhet eller proccess som är subnod, när flera subnoder används så används individuella slave selects.

För att överföra data och kommunicera mellan noder så behöver protokollet konfigureras. Detta görs genom att modifier binara bitar i SPI:s kontrollregister (SPCR), därefter så behöver riktningen också anges, det vill säga vart informationen ska skickas till.

I projektet används SPI för att kommunicera med spelplanet som består av 3 styckna Damatrix.

För att driva LED matriserna skickas färginformation, samt katod och anodinformation med SPI.

Flödesdiagrammet nedan illustrerar kodflödet för SPI.



**Figur 2.4.8 – SPI:** *Flödesdiagram för SPI protokollet.*

**3.0 - Framgångar**

Gruppen var duktig med att planera och hålla sig till planering och sitta tillsammans för projektarbetet. Gruppen kunde på grund av denna anledning diskutera mycket med varandra kring hur rutiner och olika komponenter skulle fungera. Många frågor kring varandras kod för att förstå hur allt fungerar och även för personen som skrev rutinen att se ifall det fanns några fel.

Två viktiga rutiner som behövdes för att få spelet att fungera med de komponenter som behövdes var TWI och SPI. Dessa två rutiner var viktiga för att få komponenter att svara och även för att kunna skicka information till skärmen. När gruppen satt och diskuterade hur dessa rutiner skulle fungera så gick det snabbt framåt och inte tog stor del av projektets tid.

När rutinerna TWI och SPI var klara kunde rutiner för komponenterna som skulle användas till spelet börja skrivas. Även denna del gick väldigt fort då det mesta flöt på när gruppen samarbetade. Gruppen brukade hålla sig till två eller fyra personer för att effektivt kunna arbeta.

**3.1 – Motgångar**

När väl kursen mikrodatorprojekt startade så insåg inte gruppen hur stort projektet skulle bli i slutändan. Detta orsakade att tiden inte räckte till för att få spelet till hur det var tänkt från början, då idéerna var många. När väl slutet av projektet började närma sig insåg gruppen hur mycket arbete som fortfarande var kvar och därav fick sitta många timmar mot slutet.

De flesta rutiner och liknande gick fort att skriva och göra klart, dock insåg inte gruppen hur svårt det egentligen skulle bli med att skriva ett videominne för spelet, där information kring karaktär, golv och objekt ska sparas. Det skulle visa sig vara ett av de svåraste hindren för projektet där nästan all tid spenderades på att få det att fungera korrekt.

**3.2 - Saker som skulle ha gjorts annorlunda**

När projektet väl hade blivit bestämt och rutinerna för komponenter och protokoll skulle skrivas så delades gruppen upp, där varje medlem satt och jobbade med något separat. Detta var något som gruppen trodde skulle fungera bra men i stället skapade problem. När problem uppstod för en medlem så fanns det ingen att diskutera med. Detta orsakade att man blev kodblind för sina egna misstag.

När väl gruppen mot slutet av projekttiden började sitta tillsammans för de sista rutinerna märktes det kvickt hur mycket mer effektivt gruppen jobbade. Detta då gruppen alltid började ifrågasätta varandra när problem uppstod, som att kod inte agerade så som planerat. När frågorna ställdes börjades problemen dyka upp och koden kunde skrivas om för att få resultaten som förväntades.

Något som gruppen gjorde mycket i början var att rita på tavla för att förstå hur saker skulle fungera. När gruppen hade delats upp så skedde detta inte på samma skala då varje medlem satt ensam med sin del. När gruppen började att problemlösa tillsammans igen så ritades det mycket mer och detta hjälpte med många problem. Koden som skulle skrivas kunde i princip ritas upp på tavla.

**3.3 – Diskussion**

**3.3.1 - Resi  
  
Referenser och information  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**4.0 – Referenser**

**How to Design a Mobile Game So Addictive It’s Almost Irresponsible**

<https://blog.proto.io/how-to-design-a-mobile-game-so-addictive-its-almost-irresponsible/>

**What makes tetris so incredibly addictive**

<https://tetris.com/article/44/what-makes-tetris-so-incredibly-addictive>

**Atmel AVR 8-bit Instruction Set**

<https://content.instructables.com/ORIG/FIH/E7GU/I301K81S/FIHE7GUI301K81S.pdf>  
  
**Datorteknik – Hyfsa kod** - Michael Josefsson  
[https://teams.microsoft.com/\_?tenantId=913f18ec-7f26-4c5f-a816-784fe9a58edd#/pdf/](https://teams.microsoft.com/_?tenantId=913f18ec-7f26-4c5f-a816-784fe9a58edd#/pdf/viewer/teams/https:~2F~2Fliuonline.sharepoint.com~2Fsites~2FLisam_TSIU51_2022VT_SP~2FDelade%20dokument~2FGeneral~2FKursdokument~2F7_Hyfsakod.pdf?threadId=19:P7uRHZyBcChbuIiSdQhbOrWzM1xBRpg9u6L_WzCjAB01@thread.tacv2&baseUrl=https:~2F~2Fliuonline.sharepoint.com~2Fsites~2FLisam_TSIU51_2022VT_SP&fileId=fd6da490-af26-4fd6-a670-6e4e32fcaf5d&ctx=files&rootContext=items_view&viewerAction=view)

**Datablad\_ATMega328.pdf** - Michael Josefsson

[https://teams.microsoft.com/\_?tenantId=913f18ec-7f26-4c5f-a816-784fe9a58edd#/pdf/](https://teams.microsoft.com/_?tenantId=913f18ec-7f26-4c5f-a816-784fe9a58edd#/pdf/viewer/teams/https:~2F~2Fliuonline.sharepoint.com~2Fsites~2FLisam_TSIU51_2022VT_SP~2FDelade%20dokument~2FGeneral~2FKursdokument~2FDatablad_ATMega328.pdf?threadId=19:P7uRHZyBcChbuIiSdQhbOrWzM1xBRpg9u6L_WzCjAB01@thread.tacv2&baseUrl=https:~2F~2Fliuonline.sharepoint.com~2Fsites~2FLisam_TSIU51_2022VT_SP&fileId=7a6138c2-2fed-4147-a6a6-114fbe064fe1&ctx=files&rootContext=items_view&viewerAction=view)  
  
**David\_hardvarubeskrivning.pdf** - Michael Josefsson  
[https://teams.microsoft.com/\_?tenantId=913f18ec-7f26-4c5f-a816-784fe9a58edd#/pdf/](https://teams.microsoft.com/_?tenantId=913f18ec-7f26-4c5f-a816-784fe9a58edd#/pdf/viewer/teams/https:~2F~2Fliuonline.sharepoint.com~2Fsites~2FLisam_TSIU51_2022VT_SP~2FDelade%20dokument~2FGeneral~2FDavid_hardvarubeskrivning.pdf?threadId=19:P7uRHZyBcChbuIiSdQhbOrWzM1xBRpg9u6L_WzCjAB01@thread.tacv2&baseUrl=https:~2F~2Fliuonline.sharepoint.com~2Fsites~2FLisam_TSIU51_2022VT_SP&fileId=c67453f6-fca2-42bb-a980-79ae964d047e&ctx=files&rootContext=items_view&viewerAction=view)  
  
**ATMega328p.pdf**  
[https://teams.microsoft.com/\_?tenantId=913f18ec-7f26-4c5f-a816-784fe9a58edd#/pdf/](https://teams.microsoft.com/_?tenantId=913f18ec-7f26-4c5f-a816-784fe9a58edd#/pdf/viewer/teams/https:~2F~2Fliuonline.sharepoint.com~2Fsites~2FLisam_TSIU51_2022VT_SP~2FDelade%20dokument~2FGeneral~2FDAvid%20datasheets~2FATMega328.pdf?threadId=19:P7uRHZyBcChbuIiSdQhbOrWzM1xBRpg9u6L_WzCjAB01@thread.tacv2&baseUrl=https:~2F~2Fliuonline.sharepoint.com~2Fsites~2FLisam_TSIU51_2022VT_SP&fileId=26114ee2-758b-4f6c-ac9c-3da051ae7362&ctx=files&rootContext=items_view&viewerAction=view)

**Drivning av LED-matrisen – DAmatrix** - Michael Josefsson

[https://teams.microsoft.com/\_?tenantId=913f18ec-7f26-4c5f-a816-784fe9a58edd#/pdf/](https://teams.microsoft.com/_?tenantId=913f18ec-7f26-4c5f-a816-784fe9a58edd#/pdf/viewer/teams/https:~2F~2Fliuonline.sharepoint.com~2Fsites~2FLisam_TSIU51_2022VT_SP~2FDelade%20dokument~2FGeneral~2FDAvid%20datasheets~2FDAmatrix%20-%208x8%20RGBLED.pdf?threadId=19:P7uRHZyBcChbuIiSdQhbOrWzM1xBRpg9u6L_WzCjAB01@thread.tacv2&baseUrl=https:~2F~2Fliuonline.sharepoint.com~2Fsites~2FLisam_TSIU51_2022VT_SP&fileId=4cd0d761-b6f6-49ac-b648-70cc5db9ba9e&ctx=files&rootContext=items_view&viewerAction=view)

**Remote 8-bit I/O expander for I2C-bus with interrupt**

[https://teams.microsoft.com/\_?tenantId=913f18ec-7f26-4c5f-a816-784fe9a58edd#/pdf/](https://teams.microsoft.com/_?tenantId=913f18ec-7f26-4c5f-a816-784fe9a58edd#/pdf/viewer/teams/https:~2F~2Fliuonline.sharepoint.com~2Fsites~2FLisam_TSIU51_2022VT_SP~2FDelade%20dokument~2FGeneral~2FDAvid%20datasheets~2FPCF8574%20-%20IOexpander.pdf?threadId=19:P7uRHZyBcChbuIiSdQhbOrWzM1xBRpg9u6L_WzCjAB01@thread.tacv2&baseUrl=https:~2F~2Fliuonline.sharepoint.com~2Fsites~2FLisam_TSIU51_2022VT_SP&fileId=dd42040c-4616-45dc-ba89-f52fd8b6649e&ctx=files&rootContext=items_view&viewerAction=view)