
02321 Hardware/Software Programming

3 ugers projekt

Udarbejdet af:

Gruppe nr. 1



s093485 - Christensen, Anders Jan



s093478 - Hansen, Mathias



s000536 - Vandall Zimsen, Jakob



s072657 - Tang Khoa Nguyen, Nikolai



s010164 - Mitri Emil Kjær Rebeiz,
Sami

Timeregnskab

| Dato | Deltager | Design | Impl. | Test | Dok. | Andet | I alt | Kommentarer |
|--------|----------|-------------|-------------|--------------|-----------|---------|-------|--|
| 1-3-11 | Mathias | 2 | 4 | | | | 6 | Færdiggørelse af wiring af rat som input device. |
| 1-3-11 | Anders | 2 | 4 | | | | 6 | Block-diagram + memory |
| 1-3-11 | Nikolai | 2 | 3 | | | | 5 | VGA |
| 1-3-11 | Jakob | 2 | 4 | | | | 6 | VGA |
| 1-3-11 | Sami | 2 | 3 | | | | 5 | |
| 1-4-11 | Mathias | | 6 | | 1 | 1 | 8 | Interface mellem rat og fpga færdigt |
| 1-4-11 | Anders | | | | | | 0 | Syg |
| 1-4-11 | Nikolai | | 6 | | | | 6 | VGA |
| 1-4-11 | Sami | | | | | | 0 | Syg |
| 1-4-11 | Jakob | | 8 | | | | 8 | Vga og Rat |
| 1-5-11 | Mathias | 2 | 3 | | 2 | 2 | 9 | Memory |
| 1-5-11 | Anders | 3 | 6 | | | | 9 | Memory + vga |
| 1-5-11 | Nikolai | 2 | 2 | | 3 | | 7 | Portmapping + Dokumentation |
| 1-5-11 | Jakob | 3 | 6 | | | | 9 | Vga |
| 1-5-11 | Sami | 1 | 2 | | 4 | 2 | 9 | Memory og rapport |
| 1-6-11 | Mathias | 4 | 4 | | | | 8 | memory sammen med |
| | | | | Gruppe nr. 1 | | | | lc3 cpu |
| 1-6-11 | Anders | s093485 - 4 | s072657 - 4 | s093478 - | s010164 - | s000536 | 8 | memory sammen |

Akkumuleret timeregnskab

| Mathias | Anders | Nikolai | Jakob | Sami | Total |
|---------|--------|---------|-------|------|-------|
| 119 | 100 | 81 | 114 | 63 | 477 |

Rollefordeling**Analyse/Design**

Lorem ipsum: ?

LC3 Implementering

VGA samt ROMs til VGA: Primært udarbejdet af Jakob & Anders

Hukommelse: Primært udarbejdet af Anders & Jakob

Rat I/O: Primært udarbejdet af Mathias

UART: Primært udarbejdet af Anders & Jakob

Sprites: Primært udarbejdet af Anders & Jakob & Mathias

Spil implementering

Spillogik(Instruktioner til LC3): Primært udarbejdet af Mathias

Hardware(Signaler og objekter): Primært udarbejdet af Jakob & Anders

Test

Lorem ipsum: ?

Dokumentation

Lorem ipsum: ?

Indholdsfortegnelse

| | |
|--------------------------------|------------|
| 1 Opgave formulering. | 1 |
| 2 Design | 4 |
| 2.1 Design valg | 5 |
| 2.1.1 Wrapper | 5 |
| 2.1.2 Padding | 5 |
| 2.1.3 Tri State Buffer | 6 |
| 2.1.4 Chip Selector | 6 |
| 2.2 Ram | 6 |
| 2.3 VGA | 7 |
| 2.4 Rat | 7 |
| 2.5 DIO4 I/O | 7 |
| 2.6 UART. | 7 |
| 2.7 Grafik. | 8 |
| 2.8 LC3 Instruktioner | 8 |
| 2.9 Highscore database. | 8 |
| 3 Diskussion. | 10 |
| 4 Videreudvikling | 11 |
| 4.1 Hvad vi har nået. | 11 |
| 5 Konklusion | 12 |
| 5.1 ET UDKAST!!! | 12 |
| Kildehenvisninger. | 12 |
| Bilag | A-1 |

Figurer

| | |
|---|---|
| 1.1 Simpelt mock-up over systemet | 2 |
| 1.2 Videreudviklet mock-up over systemet | 2 |
| 2.1 Design system. | 5 |
| 2.2 Screenshot af highscore applikationen | 8 |

Kapitel 1

Opgave formulering

I dette 3 ugers projekt skal der sammensættes en komplet computer, baseret på von Neumann arkitekturen ud fra en udleveret implementeret LC3 CPU, computeren implementeres på et Virtex-II Pro FPGA board.

LC3 Cpu'en er leveret til os som en black box, hvortil vi skal tilslutte de andre nødvendige komponenter for at have en komplet kørende computer; memory, vga samt et rat påsat en A/D-converter som I/O til LC3'en. Et eksternt stykke hardware i form af en skærm er også nødvendig.

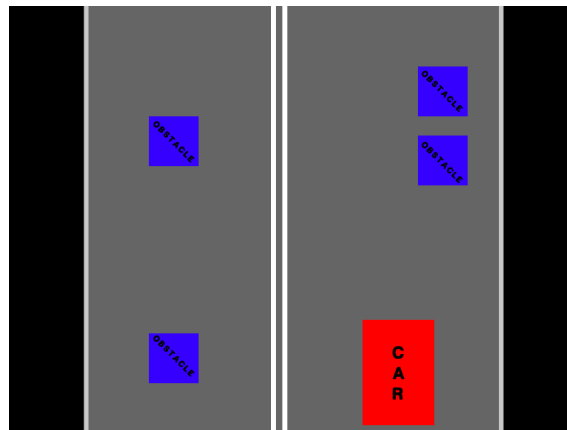
Såfremt det bliver nødvendigt, kan der være tale om at dele af spillet implementeres på en almindelig PC og med data sendt over seriel forbindelse til LC3 computeren. Derefter udvikles et spil baseret på den implementerede LC3 computer. Det er valgt at udvikle et bil spil, hvor bilen skal styres af et fysisk rat eller tastatur. Målet med spillet er at undgå forhindringer i form af andre biler, kasser og lignende som indsættes på kørebanen tilfældigt til forskellige tilfældigt tidspunkt.

På figur 1.1 på den følgende side ses et simpelt mock-up over hvordan spillet kunne se ud, hvor der er indtegnet en simpel baggrund, en bil man styrer samt forhindringer man skal undgå.

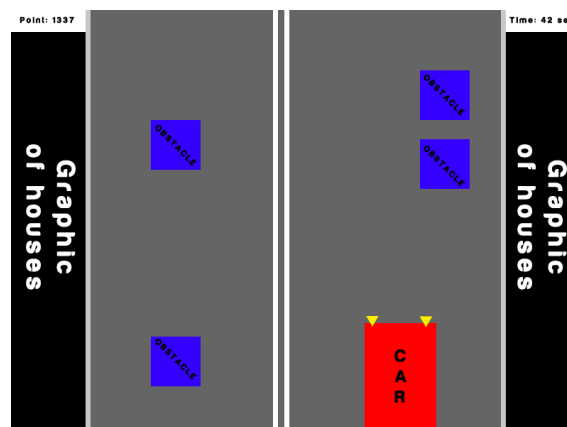
Afhængigt af tiden op til aflevering, påtænkes der at udvide projektet med en eller flere ekstra tilføjelser. Disse tilføjelser kunne f.eks. være at indsætte flere *forskellige* forhindringer på vejen, benytte *forskellige* baggrunde, have flere *forskellige* baner, lave bedre grafik eller at implementere lyd.

På figur 1.2 på næste side ses et tænkt videreudviklet mock-up over spillet, hvor points og tid også er vist.

Vores mål og krav er, at implementere et minimum af funktionalitet, som beskrevet i den opgave case som ligger til grund for projektet, inden for den givne tidshorisont.



Figur 1.1: Simpelt mock-up over systemet



Figur 1.2: Videreudviklet mock-up over systemet

Ikraft af vores roller som udviklere, stiler vi naturligvis efter at lave det bedst kørende simulator spil på den bedst kørende LC3 computer. Tiden er dog en afgørende faktor for, hvor meget vi når at implementere ud over vores minimum krav. Vi påtænker at kode hardwaren i VHDL og spillogik i C.

Desuden er det ønskeligt, at bruge viden fra andre af semesterets kurser til projektet, herunder f.eks. implementering af en database til at gemme spiller navne og highscores i spillet, med henvisning til kurset 02344 OOAD og databaser.

Kapitel 2

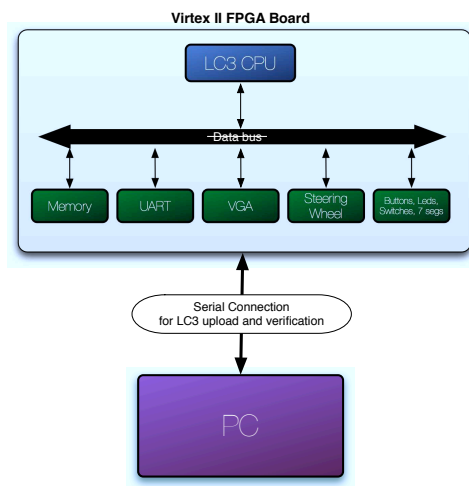
Design

| | |
|-------------------------------|----------|
| 2.1 Design valg. | 5 |
| 2.1.1 Wrapper | 5 |
| 2.1.2 Padding | 5 |
| 2.1.3 Tri State Buffer | 6 |
| 2.1.4 Chip Selector | 6 |
| 2.2 Ram | 6 |
| 2.3 VGA | 7 |
| 2.4 Rat. | 7 |
| 2.5 DIO4 I/O | 7 |
| 2.6 UART | 7 |
| 2.7 Grafik | 8 |
| 2.8 LC3 Instruktioner | 8 |
| 2.9 Highscore database | 8 |

Vores system som overordnet er beskrevet på figur 2.1 på modstående side, er opbygget af et FPGA board der implementerer en computer med en black box LC3 CPU. Til computeren er der af eksterne I/O komponenter tilknyttet en skærm, et I/O board¹, til Digilent boardet er tilsluttet et rat som essentielt er en variabel modstand. Desuden benyttes der to-vejs serial kommunikation til en PC til upload af bruger-programmer til LC3 computeren samt verificering af disse.

Spillogikken er skrevet på PC i C kode, derefter skal indholdet overføres til FPGA board vha. en serial forbindelse. Indholdet af spillet gemmes i block ram på FPGA. En VGA skærm viser output og er opdateret løbende, for at man kan se og følge med i hvad der sker og dermed kan styre spille. Spillet kan styres direkte fra FPGA board ved hjælp af en styrings enhed, denne er et rat der blevet implementeret.

¹Digilent DIO4™I/O board



Figur 2.1: Design system

LC-3 system design består af forskellige komponenter som er sat sammen. De forskellige komponenter: CPU, rat I/O, Memory (RAM og ROM), UART, VGA og DIO-4 board.

Til dette projekt har det været nødvendigt at tilkoble en VGA skærm og en A/D converter.

Design valg

2.1

2.1.1 Wrapper

Til LC3 systemet valgtes det at bruge wrappers til de elementer der både har et input og et output, dette gjorde det nemmere at samle padding og tristate buffer. Ved at lave en wrapper kunne der sendes signaler videre ned i LC3 systemet fra bussen.

2.1.2 Padding

Padding er en metode hvorved længden af en bitstreng ændres. Da det var nødvendigt at ændre bus adresse længden i f.eks. UART'en fra en længde på 16 bit ned til 8 bit, da det er den længde UART forventer. Dette gøres ved at når UART wrapperen modtager et

signal sender den kun de 8 mindst betydende bit videre ned til UART'en at arbejde med, og når et signal modtages fra UART'en paddes der med 0'er foran, så der bliver sendt et signal tilbage til bussen der er konstrueret således '00000000' og UART signal. Derved ender vi med et signal på 16 bit igen, som er den længde bussen arbejder med.

2.1.3 Tri State Buffer

En tri state buffer fungerer således at der bliver sendt et enable signal der et andet signale angiver om der må skrives eller læses fra et element, og derudover sørger for at hvis det ikke er det element der skal bruge signalet, kun vil modtage Z'er der i et signal er en tom værdi. På denne måde er det muligt at styre hvilke elementer der bliver skrevet og læst fra på en given clock cycle.

2.1.4 Chip Selector

Chip selector gør det muligt at lave en kontrol af hvilke adresse der bliver skrevet til, og igennem et I/O adresse register, er det blevet bestemt hvilke værdi der har med hvilke signaler at gøre. Ved at tjekke på denne adresse kan man derved bestemme om et signal til tri state bufferen skal være 1 eller 0.

Ram

2.2

LC3 complete system RAM implementering. Til projektet valgtes det at bruge single-port synchronous RAM. Der blev valgte single-port RAM fordi i første omgang var der ikke brug for at skrive til eller læse fra 2 adresser samtidig, som dual-port RAM ville have givet mulighed for. Synchronous tillader at bruge block rammen på FPGA boardet, og ikke CLBs rammen, der er forbeholdt til logic og asynchronous RAM. Til dette projekt var der ingen grund til at bruge asynchronous, da alle udregningerne ikke behøver at blive behandlet sammen clock cycle. Implementeringen af RAM, skete ved at følge bogen og derefter justere længden af adressen så det passede til den størrelse af det designede program der blev kørt på FPGA'en havde. Ved at reducere størrelsen på rammen, sørgede dette for at syntetiseringen af LC3 systemet ikke tog unødvendig langt tid. Der blev dog løbende ændret på adressen længden af hukommelsen, da C kode blev længere, og krævede en større hukommelse.

Til design af rammen blev der brugt følgende design decisions. Der blev lagt en wrapper omkring rammen for at gøre det nemmere at bruge et andet design decisions nemlig padding. Derudover er benyttet en tri state buffer til at afgøre om der skulle læses eller skrives til rammen.

VGA

2.3

Lorem ipsum dolor...

Rat

2.4

Det er valgt at benytte et rat som input, rattet er et gammelt playstation/pc rat som er skilt ad for at bruge rattets egenskaber og komponenter. Når rettet drejes påvirkes en variabel modstand som giver et analogt signal om rattets position, signalet læses af en 12 bit A/D converter² og kan på den måde fortolkes af LC3 processoren, når rattet er drejet helt til venstre er outputtet fra A/D converteren 0 mens at outputtet er 4096 når rattet er drejet helt til højre, A/D converteren er forbundet til et interface board fra Digilent³

Der er desuden implementeret andre dele af rattets input og output funktionaliteter; en knap på rattet er forbundet til FPGA boardet igennem samme interface board, knappen er dog ikke i brug i det endelige projekt. Endvidere er to vibratorer fra rattet også styret af FPGA boardet således at det kan gives feedback når man kører ind i en forhindring, styringen af disse er udviklet i samarbejde med nogle diplom elektro studerende⁴.

DIO4 I/O

2.5

Lorem ipsum dolor...

UART

2.6

UART er en forkortelse for Universal asynchronous reciever and transmitter. UART blev designet ud fra eksemplet i bogen, UART'en var nødvendig for at kunne uploade c kode på LC3 systemet. Implementeringen af denne var kompliceret, men nødvendig da den blev benyttet til test, om kommunikation mellem RAM og den black box CPU der var stillet til rådighed. Til UART'en var design decicions at pakke den ind i wrapper for at gøre padding nemmere. Der blev benyttet en chip selector til at bestemme hvilke af vores I/O signaler der skulle benyttes.

²En analog til digital converter konverterer et analogt signal til et digitalt signal således at FPGA boardet kan læse fra ikke-digitale komponenter.

³Digilent FX2 MIB

⁴Lasse Møller, s093440 og Emil Møller, s08310 hhv. 3. semester diplom elektro studerende og 5. semester diplom elektro studrende.

Grafik

2.7

Til projektet efter minimum implementeringen var udførst, blev grafikken til spillet udvidet. Dette skete ved at der blevet tegnet nogle png filer i photoshop, der derefter blev konverteret til et hex array ved hjælp af værktøjer⁵

LC3 Instruktions

2.8

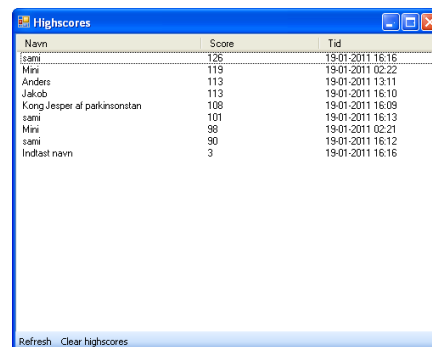
Lorem ipsum dolor...

Highscore database

2.9

Som proof-of-concept er der udviklet en applikation til PC'en som kommunikerer med LC3 Computeren over en serial forbindelse for at lagre highscores i en database, applikationen er skrevet i C#.

Applikationen opretter en tråd hvis formål er at poole for data på serial forbindelsen, af den årsag er det heller ikke muligt at køre andre programmer samtidig som bruger serial forbindelsen. Når en linje modtages med scoren oprettes et "Game Over"vindue hvor det er muligt at skrive sig på highscore listen, spillet er imens på pause indtil karakteren 's' (for start) sendes tilbage over serial forbindelsen af highscore applikationen når highscoren er gemt hvorefter spillet nulstiller og starter forfra.



| Navn | Score | Tid |
|------------------------------|-------|------------------|
| sami | 126 | 19-01-2011 16:16 |
| Mini | 119 | 19-01-2011 02:22 |
| Anders | 113 | 19-01-2011 13:11 |
| Jakob | 113 | 19-01-2011 16:10 |
| Kong Jesper af parkinsonstan | 108 | 19-01-2011 16:09 |
| sami | 101 | 19-01-2011 16:13 |
| Mini | 98 | 19-01-2011 02:21 |
| sami | 90 | 19-01-2011 16:12 |
| Indtast navn | 3 | 19-01-2011 16:16 |

Refresh Clear highscores

Applikationen benytter en database til at lagre highscores, det er valgt at benytte en sqlite⁶ database da denne danner baggrund for en simpel og hurtig database gemt i en enkelt fil i modsætning til alternativer som f.eks. MySQL som benytter kræver en database server.

Figur 2.2: Screenshot af highscore applikationen

En simpel tabel er oprettet til at indeholde highscore listen, bemærk at submitted_at angiver tiden for oprettelsen af scoren i unix tid⁷ formatet.

⁵matisen.dk/dtu/02321

⁶For yderligere information om SQLite databaser se <http://sqlite.org>

⁷Unix tid er angivet i antal sekunder siden d. 1. Januar 1970

```
1 CREATE TABLE highscore (  
2   name varchar(200) ,  
3   score int ,  
4   submitted_at int  
5 );
```

Kapitel 3

Diskussion

Under 3 uger arbejdet med projekt, har vi lært og oplevet mange nye ting. Det tales om både gode og svære tilgang.

De gode: • Vi får implementeret vores advanced/udvidelse version af spillet, og det virker som forventning. Bilens form og farver kan tegnes efter vores vilje, den kan styres fra raten. • Point statistik for hver spiller kan vises på skærm vha. en lokale database. Dette er en ekstra ting som vi har implementeret ved brug af viden fra kurset OOAD Database. •

De svære: • Det tager meget længere tid at implementere hele computeren system og få dem op at køre end vi regner med i starten. Hardware/Vhld delen tog os 2 uger at blive færdig med. • Især UART og MEM var en udfordring for os at implementere, der ligger masse "tricks" bag i. • Efter implementering af hver komponent, har vi det svært at få dem til at "snakke" sammen med LC-3 CPU, der opstod fejl i forskellige steder i vores program, som tog os ekstrem lang tid at finde rundt og rette dem. • For at nå vores tidsplanlægning om deadline for arbejde er vi nødt til at bruge weekenderne på laboratoriet.

Kapitel 4

Videreudvikling

4.1 Hvad vi har nået.. 11

Hvad vi har nået..

4.1

Vi har nået baseret på start vanskeligheder men føler os tilfredse med det produkt vi har udviklet. Vi har komponenter der virker og et kørende/næsten kørende produkt og hvis vi havde haft mere tid ville vi have bygget videre på (grafikken, spillet generelt etc). Vi tænker at en version 2 af dette spil kunne være med (bedre grafik, flere baner, større DB til yderligere data der skal gemmes, bedre statistik over f.eks. hændelser i spillet så som antal "sejre" kontra antal spillede spil.. etc.) En videre udvikling kunne også være (bedre boards, bedre komponenter, måske noget med spil over netværk, etc)

Kapitel 5

Konklusion

| | |
|----------------------------|----|
| 5.1 ET UDKAST!!! | 12 |
|----------------------------|----|

ET UDKAST!!!

5.1

Vi har med dette projekt fået en bedre forståelse for sammenspillet mellem hardware og software, hvordan de nødvendige komponenter i LC3'en bruges i samspil med et stykke software - her vores simulator spil. Vi har udviklet os fremadrettet mht. at forstå hvad der egentligt sker på komponent niveau når man bruger en computer, der kører noget software. Vi føler os bedre til at kunne abstrahere på forskellige niveauer mht. at "dykke ned" på/i komponent niveau og få et sådant til at fungere, i sammenspil med et stykke software (her vores simulator spil) og bevæge os op og ned mellem de forskellige abstraktions lag iht. low-level/high-level udvikling. Det har været meget interessant at lave/kode de respektive komponenter (så som memory, VGA.. etc) vi har brugt og derefter få dem til at "snakke" sammen med vores spil. Det har været et meget interessant projekt at gå i krig med, det har givet en meget bedre forståelse for hvad der egentligt sker "inden i" en computer, når man som bruger "bare" sidder foran skærmen og bruger den til div. ting. Vores viden er blevet bredere efter dette projekt. Dog er tidsfaktoren en afgørende faktor for udviklingen af produktet kontra ønskede opnåede mål, vi forestiller os at man ude i erhvervslivet har mere tid til udviklingen, men vi er generelt tilfredse med det vi har nået på den givne tid.. :)

Bilag

I/O Registers

.1

| Address | Description |
|---------|---------------------------------|
| xFE00 | Stdin Status Register |
| xFE02 | Stdin Data Register |
| xFE04 | Stdout Status Register |
| xFE06 | Stdout Data Register |
| xFE0A | Switches Data Register |
| xFE0E | Buttons Data Register |
| xFE12 | 7SegDisplay Data Register |
| xFE16 | Leds Data Register |
| xFE18 | Steering Wheel Status Register |
| xFE1A | Steering Wheel Data Register |
| xFE1C | Car VGA X position |
| xFE1E | Obstacle 1 VGA X position |
| xFE20 | Obstacle 1 VGA Y position |
| xFE22 | Obstacle 2 VGA X position |
| xFE24 | Obstacle 2 VGA Y position |
| xFE26 | Obstacle 3 VGA X position |
| xFE28 | Obstacle 3 VGA Y position |
| xFE2A | VGA Refresh Tick register |
| xFE2C | Vibrator register |
| xFE30 | VGA Road movement data register |

Kildetekst

.2

Indholdsfortegnelse

Figurer
