Space Shooter

C Ingenjörsjobb 2

Johan Granath
TEIS2012
jg@int80h.se

16 januari 2013

Innehåll

1 Sammanfattning

Ett enkelt spel har tagits fram i C på en befintlig hårdvara. Det går ut på att spelaren kontrollerar ett skepp som ska skjuta ner attackerande aliens. Projektets konstruktion är modulbaserad för att kunna återanvända kod i framtida projekt. Kodstorleken på projektet är totalt 89388 bytes, eller ca 90kb.

2 Specifikation

Uppgiften är att konstruera ett spel i C på en hårdvara som kör NIOSII-CPU:n. Spelet kallas för 'Space shooter' och ska vara ett arkadliknande spel där spelaren är ett skepp som ska skjuta ner aliens som attackerar. Se figur 1.

Figur 1: Space shooter

2.1 C-kod

Det finns konstruktionskrav från kunden. De gäller hur C-koden skall vara disponerad.

1. Filhuvud i varje C-fil
2. Namn på variabler ska vara tydliga
3. Kontroll av gränsvärden skall ske
4. Rikligt med kommentarer

3 Kontrollfunktioner

Spelaren kontrollerar skeppet med KEY0-1, vilket förflyttar skeppet till höger respektive vänster. För att skjuta projektiler mot aliens så används KEY2.

3.1 Krav

- Aliens ska öka i hastighet i takt med att spelet fortlöper.
- Aliens ska kunna skjuta projektiler mot skeppet.
- Skeppet ska styras med KEY0-1, samt KEY2.
- Hälsa ska visas med en grön stapel i övre högra hörnet.
- Poäng ska visas med siffror i övre högra hörnet.
- Spelet fortlöper tills det att spelaren dör.
- Om en alien eller skeppet träffas med en projektil ska en explosion visas.
- Bakgrunden ska vara stjärnor som skrollas från ovankant till nederkant.

4 Hårdvara

Hårdvaran levereras av kunden, denna kan ej påverkas utan konstruktören är tvungen att förhålla sig till den vid utveckling av spelet. I korthet bygger hårdvaran på följande huvudkomponenter (ej komplett):

- NIOSII CPU
- VGA-kontroller
- Button 0-3 (KEY0-3)
- Gröna LEDs
- JTAG UART
- LCD display

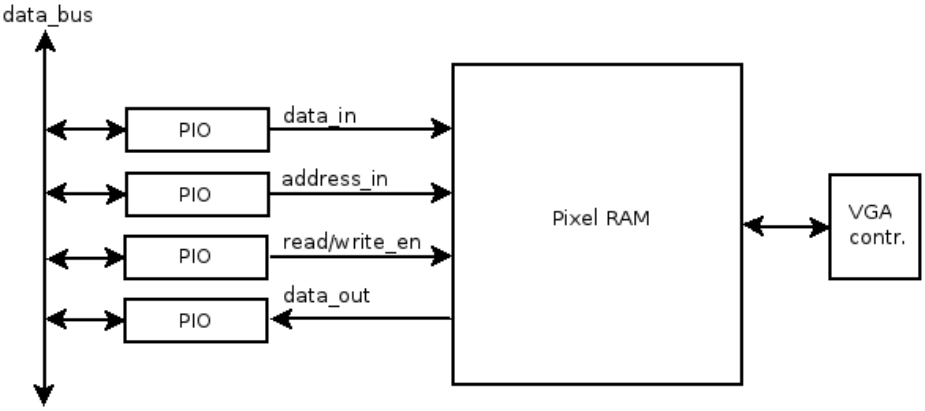
För att använda hårdvaran så tillhandahåller kunden en minneskarta över komponenternas minnespositioner. Se figur 2.

Slave Descriptor	Address Range	Size	Attributes/PIN
Terasic_IrDA_Rec_0	0x001010E0 - 0x001010E3	4	IR enheten
sysid	0x001010D8 - 0x001010DF	8	-
jtag_uart_0	0x001010D0 - 0x001010D7	8	Printf, scanf
Expansion_in	0x001010C0 - 0x001010CF	16	GPIO[7..0]
Expansion_out	0x001010B0 - 0x001010BF	16	GPIO[17..10]
lcd_0	0x001010A0 - 0x001010AF	16	LCD display
VGA_data_r	0x00101090 - 0x0010109F	16	VGA
VGA_Control_w	0x00101080 - 0x0010108F	16	VGA
VGA_data_w	0x00101070 - 0x0010107F	16	VGA
VGA_adress_w	0x00101060 - 0x0010106F	16	VGA
PERIPH_SYS_BUTTON_PIO	0x00101050 - 0x0010105F	16	Button 0-3
PERIPH_SYS_LEDS_PIO	0x00101040 - 0x0010104F	16	De längst till höger
PERIPH_HIGH_RES_TIMER	0x00101020 - 0x0010103F	32	timer
PERIPH_SYS_CLK_TIMER	0x00101000 - 0x0010101F	32	timer
onchip_memory2_0	0x00080000 - 0x000CFFFF	327680	memory

Figur 2: Minneskarta

Speciellt viktigt för konstruktionen är VGA-komponenten som har ett specifikt sätt att interagera mot. VGA-enheten är konstruerad som ett RAM-minne. Varje enskild pixel är addresserbar och både läs- och skrivbar. Skärmen har 320 pixlar i bredd och 239 pixlar i höjd. Totalt 76480 individuellt adresserbara pixlar. Varje address kan ha ett värde som motsvarar 3 bitar (RGB). Det ger oss en upplösning på 320x239x3.

Arkitekturen ska hanteras som ett vanligt RAM. Det innebär att det finns data-, address- och kontrollbus tillgängligt. Se figur 3. För att interagera med VGA-enheten ska ett strikt protokoll följas.



Figur 3: VGA-enhet

För att läsa och skriva till Pixel-RAM används Alteras funktioner:

```
IOWR_ALTERA_AVALON_PIO_DATA()
IORD_ALTERA_AVALON_PIO_DATA()
```

För att detta skall fungera krävs att systemet känner till basaddressen för de respektive PIO-enheterna. Se tabell 1.

Tabell 1: Basaddresser

Basaddress (namn)	Förklaring	Storlek (bitar)
VGA_ADRESS_W_BASE	Pixel RAM	17
VGA_DATA_W_BASE	Pixel data (RGB)	3
VGA_CONTROL_W_BASE	Read/Write Enable	1
VGA_DATA_R_BASE	Pixel data (RGB)	3

4.1 Skrivcykel

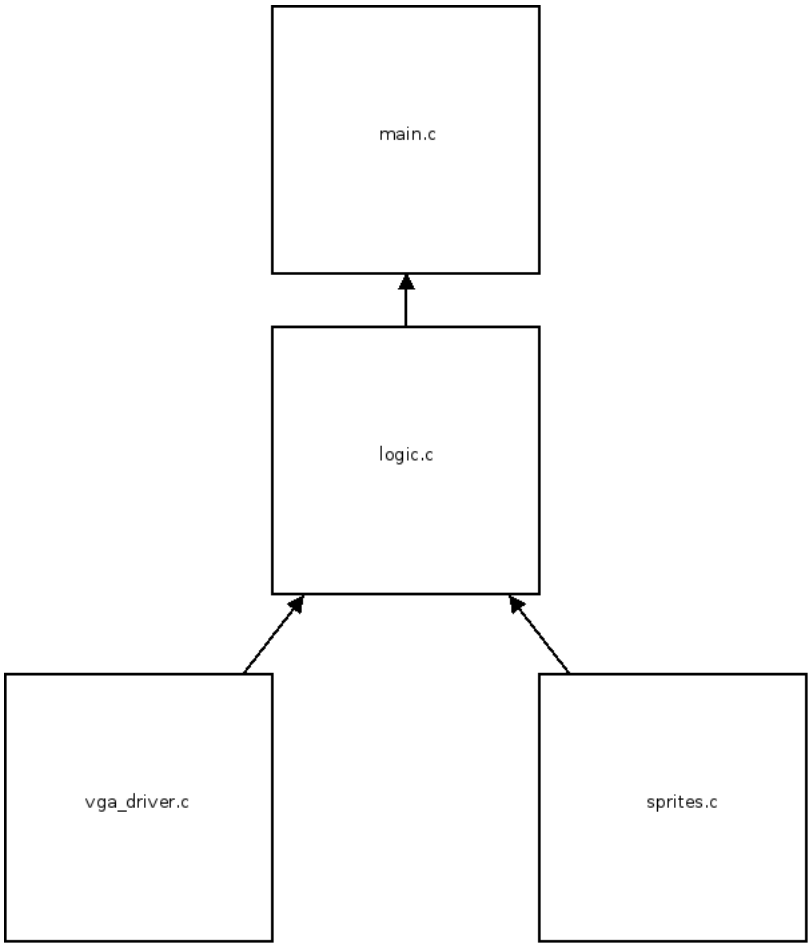
1. Tilldela data_in (RGB)
2. Tilldela address_in (Position)
3. R/W sätts till 1
4. R/W sätts till 0

4.2 Läscykel

1. Tilldela address_in
2. Läs från data_out

5 Mjukvara

Hela projektet är utvecklat i C. En modulär design har använts för att hålla nere komplexiteten samt för att öka möjligheten att återanvända moduler i andra projekt. Se figur 4 för en översikt av strukturen.



Figur 4: SW-struktur

Konstruktionen kretsar kring tre delmoduler som sedan knyts samman i top-level modulen main.c. Dessa delmoduler är logic.c, vga_driver.c och sprites.c.

5.1 main.c

Detta är topmodulen som knyter samman projektet. Här anropas de olika funktionerna från logic.c-modulen. Denna modul innehåller main()-funktionen och en ändlös loop (for(;;)) som loopar genom hela spelet.

5.2 logic.c

Kärnan i spellogiken. Det är här som alla funktioner som anropas av main()-funktionen finns. I korthet så är det följande funktioner.

- update_keys() - uppdaterar KEY0-3.
- update_starfield() - uppdaterar parallax-scroll.
- update_game_over() - kontrollerar tillstånd för game over.
- update_enamies() - uppdaterar aliens.
- update_ship() - uppdaterar skeppet.
- update_bullets() - uppdaterar skeppets projektiler.
- update_enemy_bullets - uppdaterar alien projektiler.
- update_explosions() - uppdaterar explosioner.
- delay() - delay för varje iteration av updates.

5.3 vga_driver.c

Drivrutiner för VGA. Här finns funktioner för att skriva en pixel, linjer och cirklar på skärmen.

5.4 sprites.c

Här finns sprites definierade och även hjälpfunktioner som skriver ut dem på skärmen.

5.5 Kodstorlek

Ingen optimering gällande storlek på konstruktionen har gjorts.

Tabell 2: Storlek

Minnestyp	Storlek
.text	77412
.data	7048
.bss	4928
total	89388

6 Verifiering och Validering

Konstruktionen ska testas genom att verifiera och validera olika scenarion. Dessa finns beskrivna i tabell 3.

6.1 Testprotokoll

Tabell 3: Testprotokoll

case	beskrivning	OK	verifiering	validering
case_1	KEY0 intryckt	Skeppet rör sig åt höger	OK/ej OK	OK/ej OK
case_2	KEY1 intryckt	Skeppet rör sig åt vänster	OK/ej OK	OK/ej OK
case_3	KEY2 intryckt	Skeppet skjuter en projektil	OK/ej OK	OK/ej OK

6.2 Validering

Konstruktionen validerades på Alteras DE2-115 kort med inkopplad VGA-skärm. Se figur 5 för en skärmdump.

Figur 5: Space shooter

Det finns en film som visar en spelsession. Den finns på länken: http://www.youtube.com/watch?v=f_EYaiPG5vc

I projektinlämningen finns två filer som gör att kunden kan testa konstruktionen.

- trn_teis_c_time_limited.sof - hårdvara för programmering av kortet.

- space.elf - C-kod för nedladdning till kortet.

A Testprotokoll

Tabell 4: Testprotokoll

case	beskrivning	OK	verifiering	validering
case_1	KEY0 intryckt	Skeppet rör sig åt höger	OK	OK
case_2	KEY1 intryckt	Skeppet rör sig åt vänster	OK	OK
case_3	KEY2 intryckt	Skeppet skjuter en projektil	OK	OK

Verifieringen genomfördes med en debugger.