Propeller Game Console

GameBlade

Xander Bos, Imara van Dinten
Technische Informatica jaar 4
datum: 27-02-2013

Inhoudsopgave

Inhoudsopgave

1. Project omschrijving

Voor ons Business Unit project kregen wij de vrijheid om een projectvoorstel in te leveren. De eis hierbij is dat deze goed moet aansluiten bij onze studierichting en ook van een goed niveau moet zijn. Het project omhelst de ontwikkeling van een embedded game console. De bedoeling hierbij is dat een hobbyist deze kan gebruiken om mee te ontwikkelen. Voor dit project is onderzoek gedaan naar verschillende soorten embedded devices en implementatie mogelijkheden. Centraal hierbij staat dat een werkend prototype opgebouwd dient te zijn uit componenten die voor eenieder vrij toegankelijk zijn.
Dit DIY platform hebben we de naam GameBlade gegeven, een kleine woordspeling op het doel (games) en de naam van de microcontroller, Propeller.

2. Keuze prototype board

2.1 Propeller

De Propeller van Parallax Inc. biedt interessante mogelijkheden. De architectuur omhelst meerdere cores, genaamd cogs (zie appendix A), om software uit te kunnen voeren. Voor het gebruik van de Propeller heeft Parallax zelf een taal ontwikkeld, deze heet SPIN. SPIN is een zogenaamde geïnterpreteerde taal. Binnen het geheugen van de Propeller zit een vertaler die zorgt voor de uitvoer van instructies. SPIN wordt gecompileerd naar een tussenvorm, zogeheten bytecode. Deze bytecode wordt op zijn beurt gedurende runtime weer vertaald naar daadwerkelijke instructies voor de processor zelf.

Om dit gedrag tegen te gaan is gekozen voor een gecompileerde taal, namelijk C. De performance van gecompileerde binaries is hoger ten opzichte van een geïnterpreteerde taal, dit komt doordat er geen vertaalslagen meer gedaan hoeven te worden. Voor games zijn vertragingen door dergelijke vertalingen niet wenselijk, de algehele spelervaring kan hierdoor onder de maat zijn.
Wij maken hier gebruik van een C cross-compiler voor de Propeller die in een bèta-fase is. Er zitten dus mogelijk fouten in binaries die gecompileerd worden. Met performance in ons achterhoofd hebben wij dit als plausibel risico aangenomen.

Er is een groot voordeel dat de Propeller heeft ten opzichte van andere architecturen. Dit is dat er speciale registers in zitten met betrekking tot video-timing. Door het gebruik van deze registers is het genereren van een video-signaal (VGA, Composiet, o.i.d.) mogelijk met een minimum aan externe componenten. Verder bevat de ingebouwde ROM bitmap fonts, look-up tables voor logaritmen, anti-logaritmen en sinussen.

Deze lookup-tables zorgen dat het genereren van geluid puur binnen software gedaan kan worden. Elke simpele geluidsgolf is namelijk opgebouwd als een mogelijke functie van sinussen en logaritmen.
De Propeller kent niet alleen maar voordelen. Zo is voor sommige ontwikkelaars de programmeertaal SPIN een erg groot mysterie terwijl de taal bij anderen wel aan slaat. Dit is

niet zo’n erg groot punt omdat potentiële gebruikers een groot scala aan voorbeeldcode kunnen vinden in de Propeller Object Exchange.
Ook mist de Propeller specifieke I/O interfaces zoals SPI, I2C, UART etc. Deze worden in een software driver geïmplementeerd. Dit neemt schaarse resources (geheugen) in beslag. Bij het ontwerp is hier rekening mee gehouden. Interfaces die gebruikt worden verzorgen meerdere functionaliteiten.

2.2 Andere platforms

Er zijn enkele ontwikkel platforms die voor de gemiddelde hobbyist beschikbaar zijn. Hieronder valt ook de Arduino, een gebruikersvriendelijk board dat betaalbaar is. Echter zouden de meegeleverde libraries veel te veel overhead opleveren. Het geheel zou dus minimaal geïmplementeerd kunnen worden op een losse Arduino. Met een hoeveelheid randhardware zou dit een mogelijkheid zijn. Echter waren wij er van overtuigd dat een Arduino alsnog niet zou voldoen voor ons doel.

Een ander idee zou zijn om een ARM board te gebruiken. Echter is dit een architectuur die niet voor alle hobbyisten even toegankelijk is. Een gemiddeld ARM development board is namelijk meerdere malen duurder dan een Arduino of een Propeller development kit. De Raspberry Pi zou een goedkoper ARM board kunnen zijn, ware het niet dat de aansluitingen en I/O wat lastiger te benaderen zijn.

2.3 Conclusie

De Propeller biedt meer verwerkingskracht dan andere alternatieven. Het paradigma dat gebruikt wordt bij ontwikkeling is ook anders aangezien er geen stack is en geen interrupts. Het allergrootste voordeel is dat er een Round Robin scheduler in de architectuur verwerkt is.

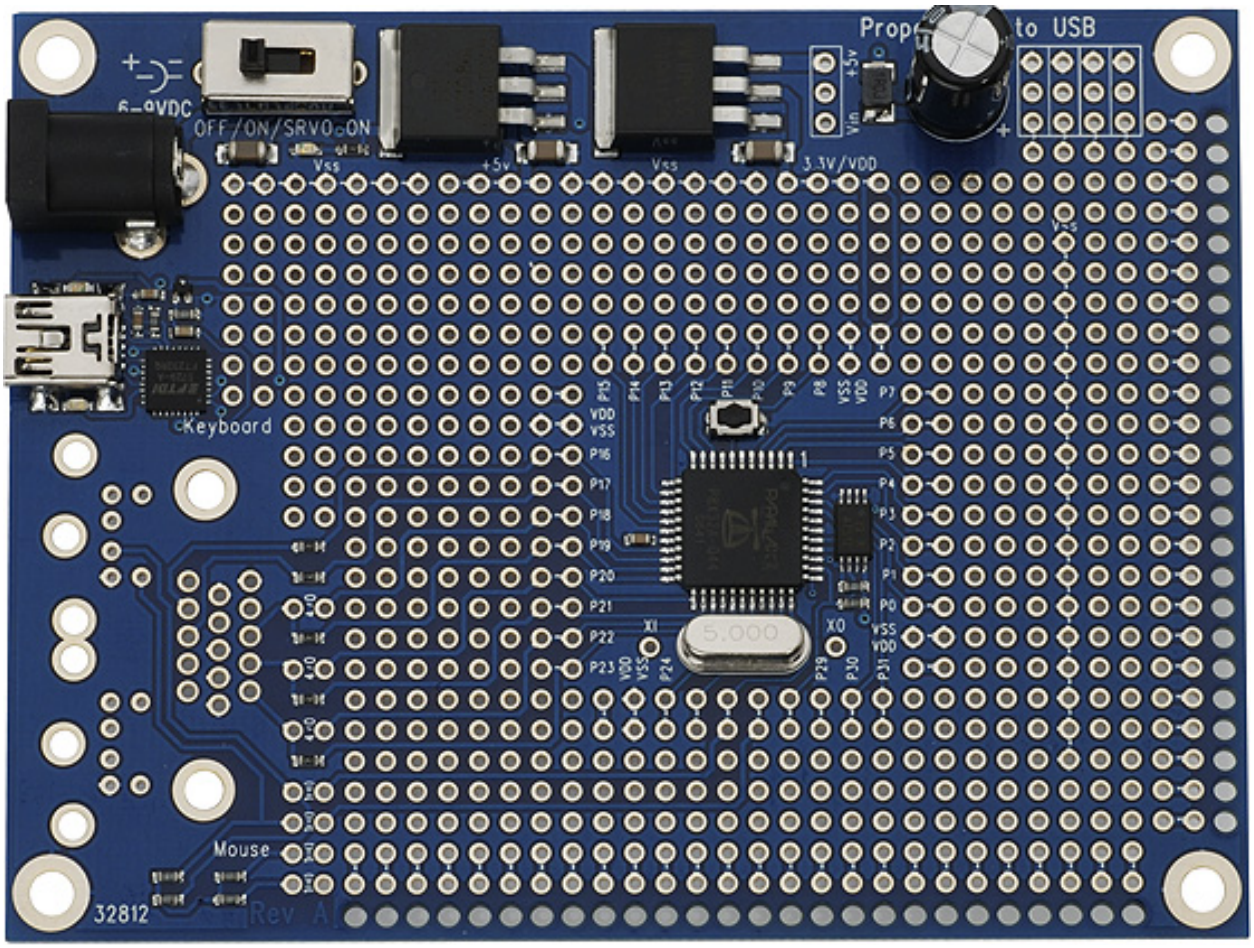
Onze keuze is op de propeller gevallen vanwege het feit dat er een grote userbase is. Er is geen beperking tot een enkele taal bij de ontwikkeling van software, dat wil zeggen, SPIN, Propeller Assembly en C kunnen door elkaar heen gebruikt worden. Daarnaast biedt de architectuur voor een groot gedeelte meer voordelen dan nadelen.

3. Ontwerp Game Console

3.1 Gekozen hardware

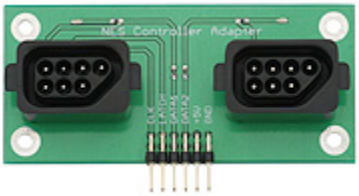
De hardware die gekozen is voor dit project is als volgt.

- Propeller Proto Board USB, een van de makkelijk te gebruiken Propeller ontwikkelborden. Op het bord zit alles dat benodigd is om de Propeller te programmeren, er is dus enkel nog een USB kabel en een externe voeding nodig. Linksonder in de afbeelding zijn nog enkele aansluitpunten te zien. Deze kunnen voorzien worden van een VGA en toetsenbord/muis adapter die los besteld kan worden bij Parallax.



- Nintendo Entertainment System controllers, voor de aansturing maakt het niet uit welke van deze controllers aangesloten wordt. Deze controllers zijn redelijk goed verkrijgbaar bij retro game zaken (zo ook via webwinkels).

- Aansluiting voor de controllers. Aangezien het lastig was aan losse controller poorten te komen hebben we besloten deze te gebruiken. Het grote voordeel ligt hem in de eenvoud bij het prototypen.



- SD card interface, deze is gekozen vanwege het feit dat een SD kaart genoeg ruimte biedt. Tevens zijn SD kaarten door de jaren heen steeds goedkoper geworden en zijn ze ook ruim verkrijgbaar.

3.2 Extra opties

- SRAM, In eerste instantie was het de bedoeling om meteen extra SRAM erbij te implementeren, echter kregen wij te maken met niet te plaatsen bugs. In verband met tijdnood waren wij noodgedwongen deze extra optie voorlopig erbuiten te laten.
- Het verwerken van de SD kaart benaderingen zou kunnen worden gedelegeerd aan een andere microcontroller, bijvoorbeeld een ATMega8 van Atmel.
- Geluid, Momenteel is er nog geen optie om geluid te kunnen afspelen. Voor een eventueel volgend prototype zouden wij dit wel graag toe willen voegen. Geluid draagt voor een groot gedeelte bij aan een betere game ervaring.
- Ondersteuning voor meer en/of andere invoermogelijkheden

4. Werking Game Console

4.1 Invoer

De game controllers bestaan eigenlijk uit 8 knoppen. Deze worden serieel uitgelezen door middel van een schuifregister. Dit type schuift alle bits een voor een naar buiten richting de microcontroller. Voor elke clock cycle die gemaakt wordt, wordt er 1 bit naar buiten geschoven. Voor het uitlezen van de twee controllers zoals aangesloten zijn dus slechts 16 cycles nodig, dit is dus in een zeer kort tijdsbestek mogelijk.

Mogelijkheden zijn nog aanwezig om via wat aanpassingen een standaard PS/2 toetsenbord of muis aan GameBlade te hangen. De externe aansluitingen bestaan, echter zijn de benodigde weerstanden niet op het bord geplaatst.

4.2 Bootloader

Een bootloader is een stuk software bedoeld als opstap om gewenste code in te laden en uit te voeren. Elke computer heeft op z'n minst een of twee dergelijke mechanismen.
Binnen GameBlade bestaat er een zogenaamde multi-tier loader. De Propeller heeft intern een loader die een stuk geheugen op het bord leest en de code daarin uitvoert. Per ons ontwerp zit hier ook een bootloader in, wat ook wel een Second Program Loader genoemd kan worden.

Dit stuk software leest de index van de SD kaart en zoekt bepaalde soorten bestanden er tussen uit. Deze bestanden kennen de extensie PEX en zijn zogenaamde Propeller Executables.
Alle gevonden bestanden worden vervolgens in een lijst getoond op het scherm. Momenteel kent deze een limiet van 24 gevonden bestanden. De gebruiker dan met behulp van een gamecontroller een bestand selecteren en er voor kiezen deze te laden.

De bedoeling is dat de Propeller deze PEX inlaadt in geheugen en uit voert. Momenteel zijn er nog enkele problemen met de bootloader. Af en toe worden PEX files niet correct ingeladen. Deze fouten hebben we nog niet kunnen traceren naar een vaste oorzaak.

4.3 Game

Als test hebben we een van de simpelere games uitgewerkt. De game die het geworden was is “steen, papier, schaar”. Het doel van ons project was niet het maken van een game, deze game is puur een proof-of-concept.

Wij willen verder met dit project en kijken wat de opties zijn om alleen de code van de game in te hoeven laden. Dat wil zeggen dat er binnen vastgestelde cores vaste drivers ingeladen zouden moeten worden. De gamecode zou dan voor een gedeelte invoerafhandeling zijn en voor een groot gedeelte messenger spelen voor een VGA driver of een audio driver.

Appendix A