0.1 Analyse og optimering

Optimering af projektet har været i fokus, men er ikke vægtet i samme grad, som andre punkter. Dette viser sig i Quartus' analyse af projektet, hvor LE:Register-forholdet ikke er 1:1. Der er totalt 23.272 logiske elementer og 13.549 registre. Langt størstedelen af alle disse registre kommer fra de to *RamAccess*-moduler, som hver i sær har 5911 registre, hvorimod *TransferProtocol* har 59, mens *PlaySound* har 66.

Quartus vil desværre ikke vise hvad fmax er, hvilket er en skam, da dette fortæller hvor høj frekvens systemet maksimalt vil kunne køre med.

I bund og grund ville det heller ikke give alt for meget mening at gå ned og optimere meget i VHDL-koden idet vi har valgt at skrive store dele af vores projekt i C som langt fra er så hurtigt som hvis man gjorde det direkte i VHDL. Herunder er bl.a. også at vi vælger at bruge math-bibliotekets sinus-funktion hver gang en ny frekvens findes frem for at gemme det i en LUT. Se mere herom under **??**

0.2 Test og simulering

For at sikre, at VHDL-koden virker som ønsket, er koden testet med programmet ModelSim 10.1b, hvor hvert komponent er unittestet samt en integreringstest for 2 af komponenterne.

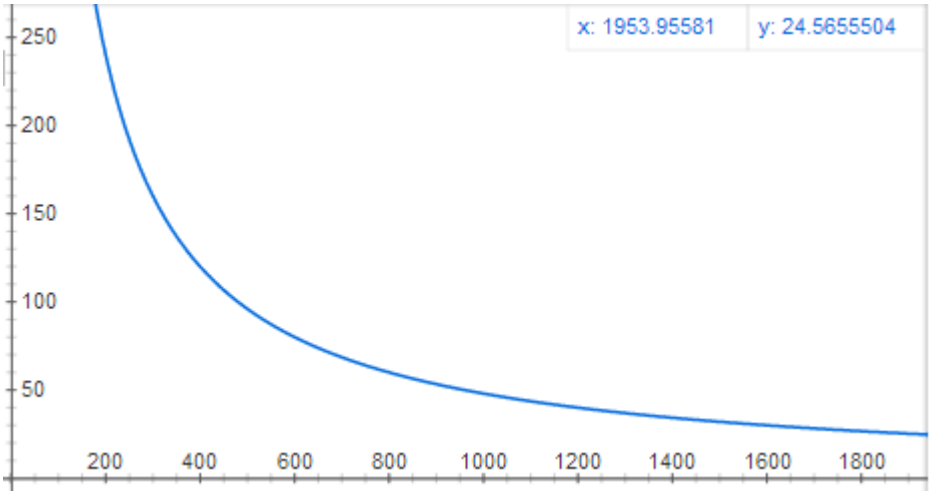
Koden er testet således, at alle linjer er blevet afprøvet, så der ikke er en linje der vil sende underlig data ud. Ligeledes er der testet flere scenarier, hvor forskellige værdier sendes til og fra de forskellige komponenter.

Grundet den mængde af data der sendes til ram-modulerne, for hver frekvens der ønskes afspillet, er der ikke testet nær det antal der sendes over, men blot enkelte, sekventielle værdier, således selve processen er vist funktionel. Hvis man ønskede at teste for, at alle værdier ville blive skrevet korrekt over vil det tage meget lang tid at verificere, bedst illustreret på figur 1.

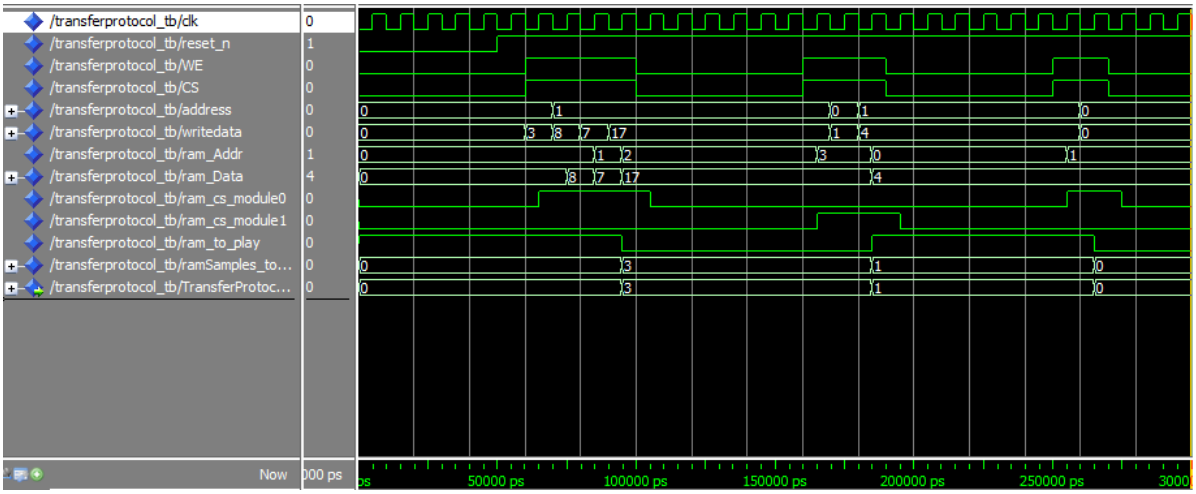
$$\text{antal samples} = \frac{48000 \text{ Hz}}{\text{ønsket frekvens}}$$

Første test er af komponentet *TransportProtocol*, der står for overførelsen af data fra microcontrolleren til ram-modulerne. På figur 2 ses et wave-udsnit af *TransportProtocol*'s testbench:

På figur 2 foretages der først et reset af komponentet (0-50 ns), hvorefter der skrives



Figur 1: Mængden af data der skal overføres (y-akse) for en frekvens (x-akse)



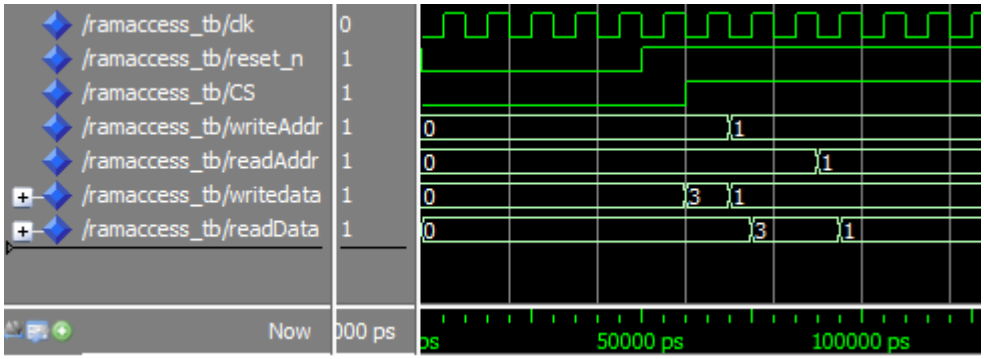
Figur 2: Waveforms for *TransportProtocol*

hvor mange samples der kommer (60-70 ns), værdierne skrives med 10 ns mellemrum (70-100 ns), hvilket også er vist i appendix **??**. Ved 95 ns bliver den sidste værdi overført og det angives hvor mange samples der er overført. Alt dette skrives til *ram_cs_module0*.

Efter 160 ns aktiveres *CS* igen og der skrives én ny værdi til ram-modulet. Denne skrives til *ram_cs_module1*, hvor *ram_cs_module0* er deaktiveret.

Der testes ligeledes også når der ikke sendes værdier over – altså hvor der ikke skal afspilles noget.

Komponentet *RamAccess* lagrer de overførte værdier.



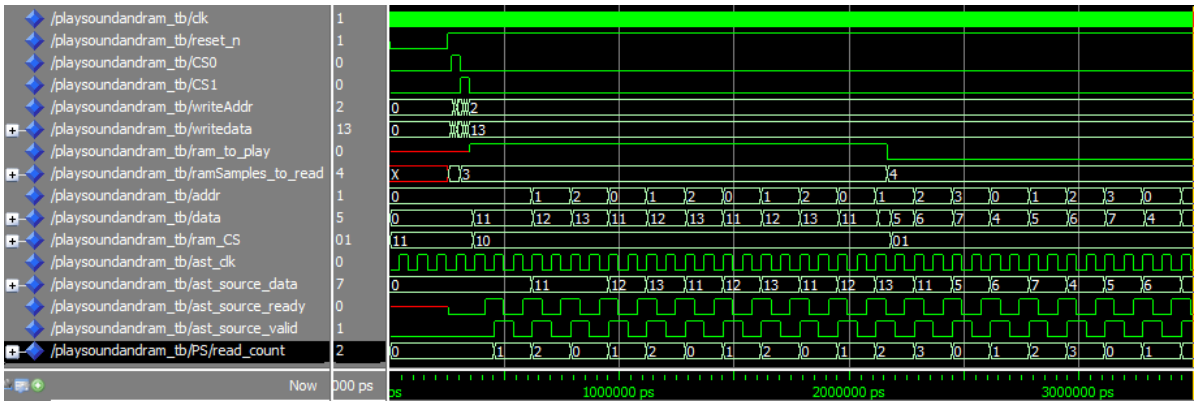
Figur 3: Waveforms for *RamAccess*

På figur 3 foretages der først et reset på 50 ns, chipselect aktiveres og der skrives til adresse 0 med værdien 3 (60-70 ns). Herefter skrives til adresse 1 med værdien 1 (70-90 ns) og fordi *readAddress* ikke ændres, læses værdien fra adresse 0 ud (75-95 ns). Dette vises også i **??**. Læsningsprocessen gentages fra 95 ns og frem.
Dette viser, at *RamAccess* umiddelbart virker som det skal.

Komponentet *PlaySound* er en smule mere omfattende og det testes sammen med *RamAccess* for at opnå det bedste resultat.

Den viste sekvens på figur 4 er på 3,5 microsekunder. Først foretages et reset, for at nulstille alle værdier (0-250 ns), hvorefter der skrives nogle dummy-værdier ind i de to ram-moduler (270-350 ns). Herefter vælges læsning fra ram-modul1 (250-620 ns) og værdierne læses og genlæses indtil ram-modul0 vælges (620-2.170 ns). Det nye antal af værdier der skal læses bliver herefter læst ind (2.170-3.500 ns).

Overstående test viser, at der kan læses fra *RamAccess* når der ligger data deri, samt den kun afspiller det den bliver bedt om igen og igen.



Figur 4: Waveforms for *Playsound* og *RamAccess*