TÖV501M Hönnun tölva

Halldór Eldjárn - *hae28@hi.is*
Kristján Eldjárn Hjörleifsson - *keh4@hi.is*

SPILADÓS

Hópur 1: Drög að hönnun trommuheila

Háskóli Íslands

12. september 2014

Efnisyfirlit

1 Inngangur **1**

2 Lýsing **1**

2.1 Notendaviðmót 1

2.2 Trommuheili 2

2.2.1 Röðull (e. sequencer) 2

2.2.2 Hljóðfæri 2

2.2.3 Geymslueining 3

2.2.4 Hljóðblandari 3

3 Kostnaðaráætlun **4**

4 Verkþættir **4**

5 Frekari þróun **7**

Myndaskrá

1 Skýringarmynd af Spiladós 1

2 Hnitakerfi röðuls með ákveðnu mynstri 2

3 Bylgjugjafi 3

4 Low pass filter 3

5 ADSR-envelope 3

6 Tímaáætlun. 6

Töfluskrá

1 Kostnaðaráætlun 4

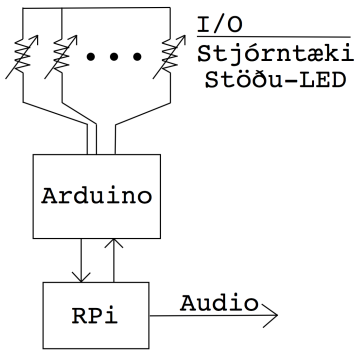
1 Inngangur

Við leggjum drög að hönnun og smíði trommuheila, sem heitir Spiladós. Trommuheili er hljóðfæri þar sem notandi getur búið til mismunandi trommumynstur (e. pattern) og stjórnað ýmsum eiginleikum hljóðanna sem úr trommuheilanum koma. Samskipti notanda við trommuheilann fara fram í gegn um stilliviðnám eða ákóðara (e. encoders) og þrýstihnappa.

Högun trommuheilans sjálfs byggir á einingamynstri og auðvelt er að útvíkka virkni hans með því að bæta við og skipta út einingum. Þannig verður hægt að velja og skrifa ný hljóðfæri í stafræna trommusettið, auk þess að geta breytt stillingum þeirra.

2 Lýsing

Trommuheilinn skiptist í grundvallaratriðum í tvo vélbúnaðarþætti; notendaviðmót og trommuheilann sjálfan. Líta má á þessa tvo þætti sem framenda annars vegar og bakenda hins vegar. Við skilgreinum skil milli þessara þátta sem gerir tækinu kleift að bregðast við breytingum á framenda með tilsvarandi gjörð á bakenda.



Mynd 1: Staða stilliviðnáma og takka er lesin af Arduino, sem svo sendir þær upplýsingar áfram til Raspberry Pi. Raspberry Pi vinnur úr þeim upplýsingum og þær geta haft áhrif á útgangshljóðið.

2.1 Notendaviðmót

Öll samskipti notanda og trommuheila eiga sér stað í gegn um samansafn stilliviðnáma eða ákóðara og þrýstihnappa. Staða þessara íhluta hverju sinni er lesin af Arduino[1] bretti, sem hefur það hlutverk að koma upplýsingunum á vel skilgreint, staðlað form og senda þær áfram á trommuheilann til þess að hann geti brugðist við aðgerðum notanda.

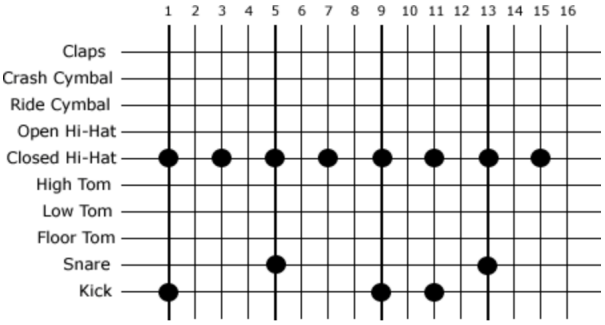
2.2 Trommuheili

Trommuheilinn sjálfur er forrit sem keyrir á Raspberry Pi[3] smátölvu. Tölvan tekur við skilaboðum frá framenda og sendir þau áfram til trommuheilans, sem svo vinnur úr þeim. Þetta forrit er í eðli sínu fjölbreytt og útvíkkanlegt. Við skilgreinum því annars vegar þá grundvallarvirkni sem þarf að vera til staðar til þess að tækið teljist tilbúið, og hins vegar leggjum við drög að því hvernig hægt væri að ná fram nýjum víddum í virkni með því að skrifa nýjar einingar fyrir trommuheilann. Trommuheilinn verður að öllum líkindum skrifaður í C++ við hönnun hans verður byggt á Tonic-bókasafninu[5] (e. library).

2.2.1 Röðull (e. sequencer)

Röðull stjórnar því í hvaða röð trommuheilinn spilar og hvaða hljóð hann spilar hverju sinni. Hentugt er að líta á röðulinn sem hnitakerfi með stakrænan tíma, τ, á X-ás og hljóðfæri á Y-ás. Eigi ákveðið hljóðfæri trommuheilans að spila hljóð á ákveðnum tímapunkti er settur hnútur í hnitakerfið á þann tíma í þá línu sem svarar til hljóðfærisins.

Þegar ákveðið mörg τ eru liðin, yfirleitt veldi af tveimur, setjum við $\tau = 0$ og mynstrið endurtekur sig.



Mynd 2: Röðull (e. sequencer). τ er á Y-ásnum og táknar stað í tíma.

Röðullinn er eðli sínu samkvæmt tímaháð (e. time critical) eining. Hann þarf að keyra óháð öðrum þáttum kerfisins og nauðsynlegt er að ávallt líði jafnlangur tími milli hverrar tímaeiningar á hnitakerfi hans. Eðlilegast er að keyra þessa einingu á sérþræði, óháðum þeim sem sér um skilin við notendaviðmótið. Sökum þessarar sterku tímatengingar er röðullinn stærsti óvissuþáttur kerfisins sem heildar.

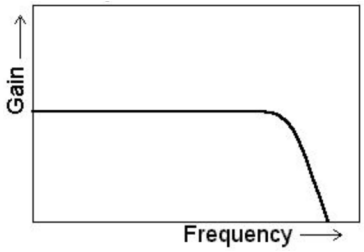
2.2.2 Hljóðfæri

Trommuheilinn þarf að útfæra eitt eða fleiri hljóðfæri sem hægt er að stjórna með röðli. Hljóðfærin eru skrifuð sem einingar, þannig að auðvelt sé að skipta út hljóðfærum og skrifa ný, ef þau sem þegar eru til staðar duga ekki til.

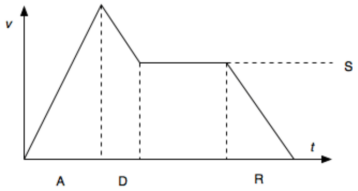
Hvert hljóðfæri samanstendur af einum eða fleiri bylgjugjöfum (e. oscillator), síu sem hleypir

í gegn um sig ákveðnum tíðnisviðum (e. low-pass filter[2], high-pass filter, band-pass filter) og umslagi[6] (e. envelope) sem stjórnar amk styrk hljóðsins.

Mynd 3: Grunnurinn að hljóðfærinu er bylgjugjafi, sem gæti gefið frá sér sínus, sagartannar eða púlsbylgju eftir því sem við á.



Mynd 4: Low-pass filter dempar þann hluta tíðnisviðsins sem er fyrir ofan cutoff-ið .



Mynd 5: ADSR-envelope (Attack, Decay, Sustain, Release). Við myndum etv. nota einfaldara envelope.

Hvert hljóðfæri mun útfæra stærðir sem stillanlegar eru gegn um notendaviðmót og hafa áhrif á mótun hljóðsins.

2.2.3 Geymslueining

Trommuheilinn þarf að útfæra einingu sem gerir notanda kleift að vista bæði þau mynstur sem hann hefur búið til í röðli, sem og stillingar hans fyrir hljóðfærin. Eins og gefur að skilja þarf því einnig að vera hægt að hlaða vistuðum stillingum inn í trommuheilann.

2.2.4 Hljóðblandari

Hlutverk hljóðblandara (e. mixer) er að stjórna styrk hvers hljóðfæris um sig, og stöðu þess í vinstri-hægri-rýminu.

3 Kostnaðaráætlun

Þróun		Framleiðsla	
Íhlutur	Verð	Íhlutur	Verð
Raspberry Pi Model B	kr. 7500,-	Raspberry Pi Model B	kr. 7500,-
4GB SD-kort	kr. 700,-	4GB SD-kort	kr. 500,-
Arduino Uno	kr. 3500,-	ATmega328	kr. 350,-
Stilliviðnám, 8 stk	kr. 2400,-	Stilliviðnám, 8 stk	kr. 2000,-
Takkar[4], 16 stk	kr. 2500,-	Takkar, 16 stk	kr. 2000,-
Kassi og tilfallandi	kr. 3000,-	Kassi og tilfallandi	kr. 1500,-
Alls	kr. 19600,-	**Alls**	kr. 13850,-

Tafla 1: Kostnaðaráætlun

Ef hægt væri að finna lausn þar sem ekki þyrfti Raspberry Pi, heldur væri hægt að nota eingöngu örgjörvann, væri hægt að lækka kostnað á íhlutum í framleiðsluútgáfu um uþb. kr. 7000,-

4 Verkþættir

Sprettur 1

1. Koma á áreiðanlegum samskiptum milli Arduino-brettis og Raspberry Pi.
2. Compile-a Tonic á Raspberry Pi og fá það til þess að keyra áreiðanlega.
3. Láta Arduino lesa stöðu stilliviðnáma og senda hana á Raspberry Pi. Láta Arduino senda Raspberry Pi skilaboð þegar þrýst er á hnappa viðmóts.
4. Skrifa eitt hljóðfæri. Skrif fyrsta hljóðfæris mun líklega taka mun lengri tíma en skrif annarra hljóðfæra.
5. Skrifa röðul.

Sprettur 2

1. Smíða notendaviðmót.
2. Tengja notendaviðmót við hljóðfæri.
3. Tengja notendaviðmót við röðul.
4. Skrifa fleiri hljóðfæri. Ætla má að skrif hvers hljóðfæris taki mun styttri tíma en skrif fyrsta hljóðfæris.

Sprettur 3

1. Skrifa módúlu sem getur vistað og hlaðið inn stillingum hljóðfæra.
2. Skrifa módúlu sem getur vistað og hlaðið inn mynstrum og stillingum röðuls.
3. Frágangur, refactoring, skrifa fleiri hljóðfæri ef tími vinnst til og það þykir þurfa.
4. Frágangur trommuheila í hentugar neytendaumbúðir.

Mynd 6: Tímaáætlun.

5 Frekari þróun

Þegar sú virkni sem lýst er í skjali þessu hefur verið uppfyllt, má útvíkka virkni trommuheilans á marga vegu. Til að mynda er hægt að skrifa fleiri hljóðfæri í C++, sem hægt væri að skipta út fyrir þau sem þegar eru til staðar.

Enn fremur væri hægt að smíða rásir fyrir hliðræn hljóðfæri og tengja þau við trommuheilann að auki við þau stafrænu hljóðfæri sem þegar eru til staðar. Þá væru stillingar þeirra lesnar inn á Raspberry Pi í gegn um stilliviðnámin í notendaviðnáminu. Þar æru þær geymdar, en einnig sendar aftur til baka í Arduino-kubbinn sem myndi þýða þær aftur á hliðrænt form og nota til þess að stjórna stilliviðnámum í hliðrænu rásunum.

Með þessu móti mætti geyma stillingar fyrir hliðrænu hljóðfærin á sama hátt og þær yrðu geymdar fyrir stafrænu hljóðfærin. Þegar spila ætti á hliðrænt hljóðfæri væri sendur gikkur (e. trigger) inn í rásina sem hefði í för með sér að umslag hennar yrði virkjað.

Heimildir

[1] Arduino Uno, Sótt 11.09.'14
http://arduino.cc/en/Main/arduinoBoardUno

[2] Low pass filter (LPF), Sótt 11.09.'14
https://ccrma.stanford.edu/ jos/fp/Simplest_ Lowpass_ Filter.html

[3] Raspberry Pi model B, Sótt 11.09.'14
http://www.raspberrypi.org/products/modelb/

[4] Takkadriver; PCB-borð fyrir takka, Sótt 10.10.'14
http://www.adafruit.com/products/1616

[5] Tonic, Sótt 09.09.'14
https://github.com/TonicAudio/Tonic

[6] Umslag (e. envelope), Sótt 11.09.'14
http://en.wikipedia.org/wiki/Synthesizer#ADSR_ envelope