

# The Disk-O-Bot (floppydrive musik)

*Gör det omoderna modernt*

William Danielsson  
TE15TEK  
Anders Hölne  
Christoffer Lin

# Sammandrag

Kan inte skriva detta för en jag har skrivit resultatet och sammanfattningen. Det är iallafall så det står att man ska göra på mallen.

Projekt Disc-O-Bot är det arbetet jag har lagt större delen av mitt tredje år på. Det arbetet som ansågs som ett lätt projekt i början men resulterade i att bli min svåraste men samtidigt mest välförtjänta arbete hittills i min karriär som blivande Civilingenjör. Disc-O-Bot är en orkester som spelas helt av mekaniska floppydrives via en Arduino-Mega. Genom helt självskriven kod i Arduino IDE har jag lyckats manipulera floppydrives till att spela över 10 låtar i exakt rätt takt och ton. Projektet har varit som en bergodalbana, upp och ner. Jag gick från att aldrig ha skrivit en enda rad kod till att skriva scripts på flera tusen rader.

Detta projekt har gett mig kunskaper om Ellära, programmering, musikproducing, datateknik, problemlösning, mekanik, och sist men inte minst uthållighet. Med det sagt vill jag stolt säga att projekt Disc-O-Bot blev en succe och jag önskar dig läsaren en trevlig läsning om min resa genom detta projekt.

# Innehållsförteckning

SAMMANDRAG.....	X
1 INLEDNING.....	X
1.1 Bakgrund.....	X
1.2 Syfte.....	X
1.3 Frågeställningar.....	X
2 Metod och Material.....	X
2.1 Förberedelsen.....	X
2.2 Utmaningen.....	X
2.3 Arduino-uno.....	X
2.4 Programmeringen.....	X
2.5 Problemet.....	X
3 Resultat.....	X
4 Diskussion/Analys.....	X
5 Sammanfattning.....	X
Källförteckning/Litteraturförteckning.....	X

# 1 Inledning

## 1.1 Bakgrund

Gamla floppydiskar och kassetter var förr i tiden i fullt bruk och alla datorer hade en. Men när CD skivor kom och revolutionerade hela it branchen lades floppydrives på hyllan, av en bra anledning. De var helt enkelt värdelösa, de nya fräscha och moderna cd-skivorna var tunnare, mindre, lättare och lät betydligt mindre. Detta är ett exempel som inte bara förekommer inom floppydiskar utan hela tiden konstant. Gamla produkter och uppfinningar blir helt värdelösa när nya uppfinningar tillkommer.

Exempel på detta är kassettspelaren, VHS-band, bakelit telefonen och skrivmaskinen. Visst är det bra att utvecklingen går framåt, men varför överge den gamla produkten helt och hållet? Jag tycker istället att man ska vara kreativ och klura på vad man skulle kunna göra med dessa så kallade "värdelösa prylar". Varför inte kombinera dåtidens teknologi med nutidens? Det är vad jag har gjort iallafall.

Jag har skapat ett "band" av 4 stycken gamla 32" floppy drives som via en arduino-uno har kunnats omprogrammeras och manipulerats för att kunna spela noter på kommando i Arduinos egna utvecklingsmiljö *Arduino IDE*.

Idén att få floppy drives att spela musik är inget nytt, redan för 4 år sedan hade de första floppydrives programmeras om till att göra sina störande ljud till noter genom att ändra frekvensen på registrerings armen. Men jag trodde aldrig att jag skulle få chansen att pröva det själv. För bara några år sedan skapades den första musikspelande floppydriven och nu är det min tur att fortsätta denna trend.

## 1.2 Syfte

Detta projekt utförs för att upplysa samhället om att alla gamla uppfinningar inte nödvändigtvis måste vara värdelösa samt för att sätta en standard för hur det tekniska programmets undervisning kan förbättras genom att låta eleverna göra om tråkiga gamla uppfinningar till något nytt och intressant. Varför bara skriva tutorial-kod när man kan få eleverna att bli musikproducenter på ett intressant och välgrundat sätt?

När det gäller själva Disk-O-Boten är det mitt mål att helt själv utan någon som helst hjälp omprogrammera och ändra mina floppydrives för att sedan kunna med hjälp av arduino-uno få dem att spela musik som jag själv ska ha programmerat i Arduino-IDE.

Disk-O-Boten ska bestå av 2-4 floppy drives och eventuellt hårddiskar som trummor.

### 1.3 Frågeställningar

Jag valde att vara själv i detta projekt för att jag vill utmana mig själv och samtidigt göra något som jag verkligen tycker är roligt och intressant. Men såklart var jag medveten om att det skulle komma problem och hinder under detta projekts gång. För att kunna få floppydrives att spela musik måste man förstå hur en sådan fungerar och varför. Jag måste veta exakt vilken av alla 32 eller 64 pins (beror på vilken slags floppydrive jag får tag på) som gör vad och veta vilka mekanismer i maskinen jag måste löda om eller ta bort för att mitt mål ska uppnås.

Jag måste även få en grundläggande grund för hur man programmerar och förstå hur man överför kod i en dator till mekaniska rörelser i en maskin. När jag gick in i detta projekt hade jag aldrig skrivit en enda rad kod förut, vilket betydde att jag förstod att mycket tid skulle läggas ner på att försöka förstå mig på programmering.

Materialet är kanske ett av de större problemen. De delar som behövs för detta projekt är varken lätta att få tag på eller billiga. Floppydrives har slutat att produceras i de flesta länder vilket har nästintill gjort dem antika. Därför kommer det vara svårt att få tag på en fungerande Floppydrive.

Sedan krävs en spänningskälla som kan ge ut olika mängder av spänningar som kommer behövas till Floppydrivens olika delar.

Förutom det behövs förstås massvis av olika kablar samt en arduino bräda och breadboard.

## 2 Metod och Material

### 2.1 Förberedelsen

Det första steget i att förverkliga detta projekt var inspiration. Såklart hade jag redan gjort research och blivit inspirerad sen tidigare men då var det på grund av nöjet. Nu gör jag min research i vetenskapligt syfte för att veta till exempel vad andra har gjort så jag vet vad jag kan tänkas förvänta mig och även kolla lite övergripligt på hur dem har gått till väga.

Det andra steget var att införskaffa allt material och delar. Det jag behövde för att komplett kunna skapa detta projekt var.

Floppy drives 32" eller 64"

Disketter (rätt storlek för Floppy driven)

Strömkälla (som ska kunna ge ut 4 och 12 volt samtidigt.)

Kablar

Arduino-Uno

USB-Mini USB kabel

Arduino-IDE (mjukvara)

Efter över en månad med att sakta men säkert få tag på dessa delar var det äntligen dags att sätta igång. Jag kom in i detta steget med mer eller mindre noll förberedelse eller kunskaper. Men efter X antal google sökningar och mycket testande hade jag förstått mig på hur jag skulle gå till väga.

### 2.2 Utmaningen

En Floppy Drive styrs med hjälp av 32 "pins" som är små metall pinnar som sticker ut på baksidan plus 4 till som är till för ström och jord. Alla dessa 32 har olika syfte och utför olika arbeten. När det skickas en ström till en av dessa pins exekveras dens specifika arbete. Det kan vara till exempel flytta läsarmen ett steg eller byta håll på läser armen eller ännu mer simpelt som t.ex lys en lampa eller stanna motorn.

De "pins" som jag var intresserad av var byt håll och rör motorn ett steg. Hela konceptet med floppydrive musik rör sig kring dessa två funktioner. Floppydriven ger ifrån sig ett ljud varje gång den rör sig och beroende på vilken frekvens (antal steg per sekund i detta fall) den har avgör ifall det blir en ljus eller mörk ton. Därför ifall man själv kan bestämma med vilken hastighet och under vilken tid den ska röra sig så kan man skapa massvis med olika noter.

Efter cirka 3 dagar av intensivt klurande lyckades jag få den att starta (lampan på driven lyser grönt) och koppla rätt pins för att den ska röra sig ett steg och byta håll.

Floppy Driven fungerar med impulser av ström. 1 impuls=1 steg vilket betyder att man inte kan skicka in ett konstant flöde. För att testa att allt var kopplat rätt fick jag helt enkelt snabbt nudda två separata kablar mot varandra flera gånger för att få den att röra sig ett steg per nuddning.

### 2.3 Arduino-Uno

Jag hade nu nått ett av mina huvudmål vilket var att kunna manipulera och kontrollera Floppy driven som jag ville. Men det var aldrig tänkt att göras manuellt. Planen var att använda Arduino för att få min dator att kommunicera med Floppydriven. Men det var lättare sagt än gjort. Jag hade aldrig arbetat med en sådan förut och kände mig helt vilse. Många Youtube tutorials krävdes men de gav mig ingen större inblick. Inte för ens Mitt Universitet hade sin årliga "Raspberry Jam" som var just en kort undervisning i hur man jobbar med den typen av elektronik. Efter endast 2 timmar hade jag blivit upplyst och förstod nu hur det fungerade. 2 dagar senare hade jag skapat en lyckad kommunikation mellan min dator och floppy driven. Nu äntligen skulle det roliga börja. Programmeringen eller som jag såg det som, musikskapandet.

### 2.4 Programmeringen

Jag trodde aldrig att detta skulle bli det mest tidskrävande och frustrerande delen i detta projekt men det var det verkligen. Att göra kod till mekaniska rörelser som sen ska efterlikna musik var minst sagt en enkel uppgift. Planen från början var att koda i programmet Python men efter att jag började lära mig enkla kommandon i Arduino IDE för att förstå hur Arduino-Uno fungerar märkte jag att jag lika gärna kunde programmera i deras egna anpassade programmeringsspråk.

Arduino-IDE är en utvecklingsmiljö som använder ett eget programspråk som kan efterliknas med C++ som är specifikt framtaget för att göra det enklare att koda sin Arduino. Den del av programmeringen som jag använde mig av handlade för det mesta av "digitalWrite" som är ett kommando för att berätta för datorn ifall något ska vara på (HIGH) eller av (LOW). Med detta kunde jag berätta för arduino att den skulle skicka strömmar till floppy driven genom att snabbt byta mellan HIGH och LOW vilket skickar pulser av strömmar som floppydriven behöver för att röra sig. HIGH och LOW kommandot används också för att ändra riktningen på läsaren. Floppydrivens ljud uppkommer varje gång den rör sig ett steg som sagt förut men läsaren kan endast röra sig 80 steg innan den krockar med väggen. Därför behövde jag kunna byta håll på läsaren för att se till att den aldrig krockade med väggen. Detta gjordes helt för hand i början, alltså att jag fick skriva in några rader kod lite då och då för att byta håll på läsaren när jag ansåg att den var på väg att krocka.

Det tog mig över en vecka innan jag hade lyckats göra mitt första "beat" och därmed rent teoretiskt sätt blev klar med projektet. Jag hade nu lyckats skapa musik från en floppydrive. Men jag ville fortsätta och lära mig mer och utveckla nya metoder och kunskaper. Det första jag gjorde var helt enkelt att skriva ungefär 5 rader kod för att få arduinon att spela en ton och sen skriva 5 rader kod till för att få den att spela samma ton fast ljusare (högre frekvens) och sen helt enkelt loopa den vilket skapade ett beat.

Men jag var inte nöjd här. jag fortsätta att skapa nya prototyper för att göra det lättare att skapa musik utan att behöva skriva överdrivet mycket kod och lyckades även koppla in mer än 1 floppy drive helt utan problem. Efter över 2 månader och över 5 egengjorda metoder att skriva kom jag äntligen på den ultimata metoden.

Den består av 10 "premade" noter som alla består av 10 rader kod som mer eller mindre flyttar läsarmen ett X antal steg och beroende på vilken ton man väljer 1-10 så kommer läsarmen att göra detta steg olika fort 1 är det snabbaste och 10 det långsammaste. Detta ger mig ett ganska stort spann av toner som jag snabbt kan läggas in i min "Loop" som det kallas. Man kan sen även snabbt välja hur lång en viss ton ska vara genom att skriva in de antal steg som man vill att läsarmen ska utföra den tonen.

Min plan då var att manuellt koda in dessa noter i rätt ordning och med paus imellan för att försöka få dem att spela början av final countdown av Europe. Jag tänkte helt enkelt lyssna och gissa på ett ungefär vilken av de 10 noterna jag skulle använda och med vilken "delay" jag skulle ha emellan noterna för att få det att låta så nära som möjligt. Detta krävdes mycket tid förstås och jag kom aldrig nära originalet men man kunde höra att det var final countdown som spelades och det fick jag helt enkelt vara nöjd med.

## **2.5 Problemet**

Men det fanns ett problem, ett problem som jag trodde var omöjligt att lösa och som fick mig att skrota allt jag hade gjort tidigare och börja om från början.

Mitt mål var att få flera floppydrives att spela olika "beats" samtidigt oberoende av varandra, men när jag kopplade in den andra floppydriven märkte jag att den först spelade alla noter på ena floppydriven och sen på den andra. Det tog mig lång tid innan jag förstod mig på problemet och jag övervägde till och med att ge upp på att skapa en låt och helt enkelt bara visa att man kan göra musik med floppydrives.

Problemet var att jag inte förstod hur datorer fungerar. Man kan tro att en dator gör flera tusen saker samtidigt, till exempel vi säger att du har google öppet samtidigt som du skriver i microsoft Word och sen kanske du har ett spel på i bakgrunden



också. Då kan man tro att datorn göra alla dessa kommandon samtidigt, men detta är helt fel. En dator kan bara göra en sak i taget men den gör dessa saker flera miljoner gånger per sekund vilket skapar illusionen av att saker händer samtidigt.

Jag använde mig extremt mycket av "delay" kommandot i min kod som var till för att berätta hur länge den skulle vänta innan den ska skicka ut nästa impuls av ström, detta var mer eller mindre kärnan av min kod. Men det jag inte fattade var att när man skriver "delay" så säger man åt arduinon att inte göra någonting för visst länge. Detta betyder att den stannar hela koden och inget annat kan hända så länge delay kommandot är i bruk. Jag blev helt enkelt tvungen att hitta ett annat sätt att göra pauser emellan noterna och stegen.

Svaret var kommandot "currentmillis" detta skapar en klocka som räknar i millisekunder, med hjälp av det kunde jag skriva saker som "gör den här saken när det har gått så här många millisekunder". Detta kanske låter simpelt men det tog en riktigt lång tid koden blev 10 gånger mer komplicerad och längre, men det fungerade och det var huvudsaken.

## 2.6 den revolutionerande idén

Målet hade alltid varit att lyssna och för hand skriva koden för att efterlikna en låt så bra som möjligt. Nu kunde jag äntligen göra det med mitt nya sätt att skriva kod på utan att använda "delay". Men jag råkade snubbla över ett nytt begrepp jag aldrig hade hört talas om förut MIDI. Jag vet inte vad som fick mig att börja läsa om vad MIDI var men jag är glad att jag gjorde det för det ändrade hela projektet helt och hållet. MIDI står för Musical Instrument Digital Interface mer eller mindre är detta ett sätt att lagra riktig musik i datafiler. Man kan till exempel spela noter på ett piano och omvandla dem till binärkod som datorn sedan kan läsa och använda för att lagra eller spela upp i musikprogram.

Där hade jag lösningen, det där var precis vad jag behövde. Ifall jag skulle kunna få min Arduino att läsa MIDI filer skulle jag kunna spela vilken låt som helst med exakt rätt timing och not utan att behöva lyssna själv och gissa på hur låten ska låta.

Med detta ingick förstås många problem och hinder som jag fick grubbla på extremt länge och säkert 100 timmar lades ner från det här steget till det färdiga resultatet.

Jag skulle kunna gå in på djupet på alla problem som tillkom men då skulle rapporten bli 100 sidor och få hälften av läsarna att somna.

Men de största problemen som tillkom var.

- Första hur en MIDI text fungerar och fatta hur dem jobbar.
- Göra så att arduinon kan läsa en text och utföra arbete utefter den texten.
- Omvandla om MIDI värdena till något som Arduinon kan förstå sig på

- Omvandla värden till rörelser i floppydriverna
- Göra om DRAM minnet till FLASH minne för att spara på data.
- Införskaffa en Arduino-MEGA för Arduino-Uno hade för lite minne
- Införa två egengjorda formler för att omvandla värdena till noter som faktiskt låter bra i floppydriven
- skriva en automatiskt script som själv byter håll på läsarmen varje 80 steg så att jag aldrig behöver oroa mig om att den ska krocka i väggen.

Lite ytligt skulle jag kunna berätta processen från att jag upptäckte MIDI till den färdiga produkten.

Jag började med att söka på nätet för att hitta färdiga MIDI filer för låtar som jag tror skulle låta bra på floppydrives, den första jag valde var förstås final countdown. MIDI filer består bara av binärkod t.ex 010100101. Det är med andra ord oläsligt för en människa så jag laddade ner ett program som heter MIDIEditor som kan öppna MIDI filer och till och med spela upp dem i programmet med konstgjorda instrument. Detta program hjälpte mig mycket på vägen därför att jag kan gå in och ändra eller ta bort instrument eller melodier för att bättre anpassa låten till mina floppydrives. När allt är klart i MIDIEditor går jag in på en sida som kallas för FlashMusicGames där jag kan skanna in min MIDIfil och göra om den till text.

En liten rad text kan se ut såhär

```
32687 On ch=1 n=50 v=95
239 Off ch=1 n=50 v=0
0 On ch=1 n=57 v=95
241 Off ch=1 n=57 v=0
0 On ch=1 n=50 v=95
240 Off ch=1 n=50 v=0
```

Bättre än såhär blir det inte, det är dem här värdena som jag kommer att använda i min kod. Jag fick lägga ner tid för att läsa upp om MIDI formatet och förstå vad den här texten betyder.

Varje rad är en separat not. Längst till vänster är tiden tills den noten ska spelas i ticks. ticks är ett tidsformat som är baserat på vad låtens BPM är (beats per minute) vilket betyder att 1000 ticks kan vara olika långt från låt till låt. ch=1 är vilket instrument som spelas, detta är för att veta vad som ska spela vad. n=57 står för not 57 som är en premade not. Not spannet går från 0-127 där 127 är den ljusaste tonen. v=0 står för velocity och är ljudvolymen i det här fallet som också kan sträcka sig från 0-127 där 127 är det högsta.

Om man kollar noga kan man se att på velocity så är varannan not på 0. Det är för att i midi så tar den aldrig pauser utan istället spelar tysta noter för att få det som att se ut som en paus. Det var det jag blev tvungen att göra med.

När jag skriver min kod till mina floppydrives så använder jag vilken not den ska spela och i hur länge, så note och ticks värdena kommer jag behöva. Men floppydrives kan bara spela i en volym så volymvärdet är inte lika viktigt förutom ifall den ska spela en tyst not.

Jag kopierade alla värdena och la in dem i ett Openoffice kalkyldokument och möblerade om värdena så att de skulle kunna bli kopierade in i koden.

I koden har jag sedan fixat så kallade arrayer vars jobb är att läsa in data. Med den data som läggs in kan jag få floppydriven att spela en viss channel/instrument beroende på vilken av MIDI värdena jag la in. Det finns såklart saker som måste omvandlas, till exempel så jobbar MIDI i ticks medan Arduino jobbar i millisekunder. Därför var jag tvungen att skriva ett block kod som omvandlar ticks till millisekunder beroende på i vilken BPM låten spelas som har nämnts ovan.

Förutom det kan MIDI spela upp noter från 0-127 medan floppydrivens noter bara låter bra mellan 8-20 vilket betyder att jag måste konvertera noterna för att de ska låta bra. Jag gick så långt till att utveckla en formel som jag döpte till floppys formel som är just till för att omvandla MIDI noter till noter som min floppydrive kan spela.

Sen var det också de tysta noterna. Jag fick se till så ifall velocity i MIDI datat var 0 så ska floppydriven inte spela någonting under den tiden som just den notens velocity var 0.

Allt detta många intensiva dagar av kodande och klurande men jag fick till det tillslut. Tills att jag märkte att kopiera in 3000 värden per floppydrive tog mycket plats på minnet. Ännu värre är att dessa värden automatiskt lagras i Ram minnet som endast kan innehålla 2472 bytes och varje värde tar upp 2 bytes.

Jag lyckades komma runt detta hinder genom mycket research att säga åt Arduinon att lagra dem värdena i Flashminnet istället som kunde hålla ändå upp till 30 000 bytes. Men även det här var för lite ifall jag ville göra komplicerad musik med flera drives. Jag valde därför att investera i den dyrare men bättre Arduino-MEGA som har 247 000 000 bytes i Flash minnet, nästan 10 gånger minnet av Arduino-uno som jag hade använt tills nu. Den nya Arduino-MEGA har även flera digitala-pins med andra ord portar som behövs för att koppla kablarna från arduinon till floppydriverna. Med Arduino-Mega kunde jag nu därför ha fler floppydrives inkopplade samtidigt ifall jag skulle vilja det.

Vid det här stadiet var alla problem lösta. Allt jag behövde göra nu för att få en fungerade floppydrive låt är att

1. hitta en låt
2. ladda ner midi filen
3. konfigurera och anpass midi filen i MidiEditor
4. omvandla midi datat till text
5. ommöblera midi texten så att den kan kopieras in i koden
6. kopiera texten in i koden
7. ändra konstanter och värden beroende på låtens BPM och not spann med floppysformel
8. verifiera koden
9. kör programmet och njut

Med allt detta klart kan man nu hyffsat lätt skapa beats och melodier. Men det finns ett sista dilemma kvar.

Arduino-IDE är skapat på ett sånt sätt att det inte går att göra flera saker samtidigt vilket betyder att jag inte kan få två separata floppy drives att spela olika saker samtidigt vilket kraftigt minimerar mina möjligheter när det kommer till musik.

Ifall jag valde att programmera i Python kanske inte detta problem hade uppstått men det var ingenting jag visste i förväg. Detta resulterar med att det inte går att göra ett band med floppydrives som alla spelar olika rytmer som var mitt mål från början.

### 3 Resultat

Jag skulle själv inte säga att jag är klar med mitt projekt, jag har fortfarande fler beats att göra och eventuellt en kort början på en låt. Därför väljer jag att inte skriva denna delen nu.

Den resulterande produkten Disc-O-Bot är en hel orkester av 10 32" floppydrives som alla helt oberoende av varandra kan producera ljud som kan bli melodier. 10 låtar har skapats som alla är spelbara med dessa floppydrives. Dessa 10 låtar är

Haddaway- What is love 1993

Aha - Take on me 1985

Europe - Final countdown 1986

pirates of the caribbean - He's a pirate 2003

Gigi D'Agostino - L'Amour Toujours 1999

Game of Thrones Theme 2011

Star Wars - Imperial March 1980

Electric Light Orchestra - In the Hall of the Mountain King 1973

Darude - Sandstorm 2000

Awolnation - Sail 2011

Koden som är skriven i Arduino IDE är helt och hållet skriven av mig själv med mina egna erfarenheter som jag har fått under arbetets gång. Disc-O-Bot'en drivs av en multivolt spänningskälla och kommunicerar med floppydrivsken via en Arduino-MEGA 2560.

# Diskussion

Jag vill börja med att berätta att jag många gånger under detta projekt velat kasta in handduken, men tack vare att jag fortsatte och var engagerad så lyckades jag uppnå ett resultat som jag inte ens hade kunnat föreställa mig för 6 månader sedan då detta projekt startade. När det gäller själva resultatet är det viktigt att förstå att jag har skapat musik från en apparat vars uppgift aldrig var tänkt att skapa musik. Vilket betyder att en låt som spelas upp aldrig kommer att låta lika bra som originalet. Det är helt mekaniska vilket innebär att det inte är gjort för att låta bra eller rent. Annars hade det redan varit ett instrument vid det här laget. Det är det själva faktum att den kan spela låtar i huvudtaget som är den stora bedriften.

Vid den tiden jag skriver denna rapport så har jag endast 4 floppydrives men tanken är att jag ska införskaffa fler inför presentationen. Anledningen varför 4 drives inte är tillräckligt är för att det inte låter tillräckligt högt och för att alla låter med samma volym vilket betyder att ifall två floppydrives spelar olika melodier i en låt så är det lätt att den ena överröstar den andra. Detta problem kan lösas ifall jag har fler floppydrives som spelar själva melodin och några mindre som spelar t.ex bas eller trummor.

Alla låtar låter inte heller lika bra på floppydrives, floppydrives är dåliga på att spela långa eller alldeles för ljusa noter. Därför är de låtarna som jag har valt de som jag anser är bäst anpassad. Vilket är t.ex simpel och lätt igenkänd melodi, korta och helst mörka noter.

Det är även viktigt att ta med sig att allt det här gjordes av en helt oerfaren nyligen myndig 18 årig teknikare med noll förberedelser. Jag hade ingen av de kunskaperna som krävs i detta projekt. Jag visste ingenting om mekanik eller ström lära på den nivån som krävdes, jag hade aldrig skrivit en enda rad kod och jag hade aldrig hållit på med musikproducerande förut.

För att sätta det i perspektiv kan jag meddela att den första kända personen som gjorde detta var Paweł Zdrożniak en student en polskt student på AGH University of Science and Technology och den andra välkända personen var en PHD högskolestudent i datorteknik där av han till och med erkände att han bara hade kopierat koden från internet (länk finns i innehållsförteckningen).

Jag har blivit så fast vid detta projekt att jag är helt säker att jag kommer fortsätta att vidareutveckla Disc-O-Bot genom att utöka antalet floppydiskar, förbättra och simplificera koden samt skapa nya låtar. Jag ångrar inte för en sekund att jag gick solo och valde detta ovanliga men kreativa projektet.

## Sammanfattning

Projekt Disc-O-Bot bevisar verkligen hur det finns mer än ett sätt att återvinna på. Varför förstöra när man kan modifiera. Projekt Disc-O-Bot är också ett bevis på att elever ska få vara kreativa när det gäller gymnasiearbetet, varför bara sitta och göra research på globala uppvärmningen därför att en lärare sa åt en att göra det när man istället kan få göra något man tycker är roligt som samtidigt testar ens tekniska egenskaper. Det är väl ändå det gymnasiearbetet är till för? Ett sista test på att man faktiskt har lärt sig något under dessa tre år. Man måste inte lösa ett storskaligt problem, det räcker med att man har kul.

## Källförteckning

[https://www.youtube.com/watch?v=yHJOz\\_y9rZE&t=1s](https://www.youtube.com/watch?v=yHJOz_y9rZE&t=1s)

[http://pinouts.ru/HD/InternalDisk\\_pinout.shtml](http://pinouts.ru/HD/InternalDisk_pinout.shtml)



Final Countdown vocals midi values.ods - OpenOffice Calc

Arkiv Redigera Visa Infoga Format Verktyg Data Fönster Hjälp

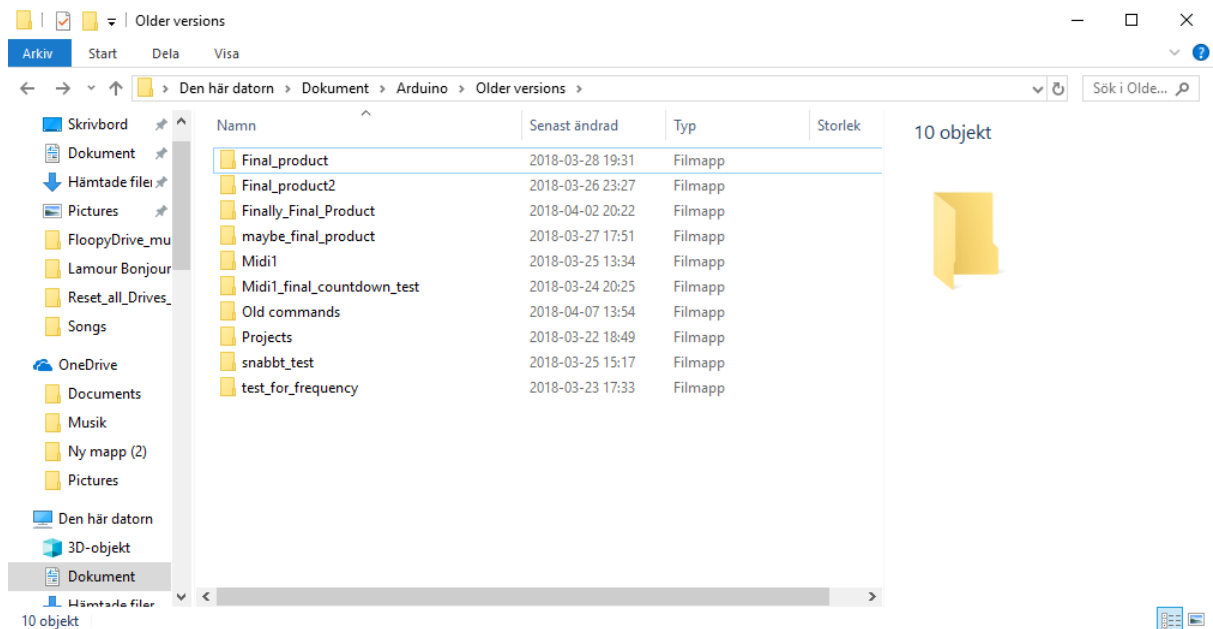
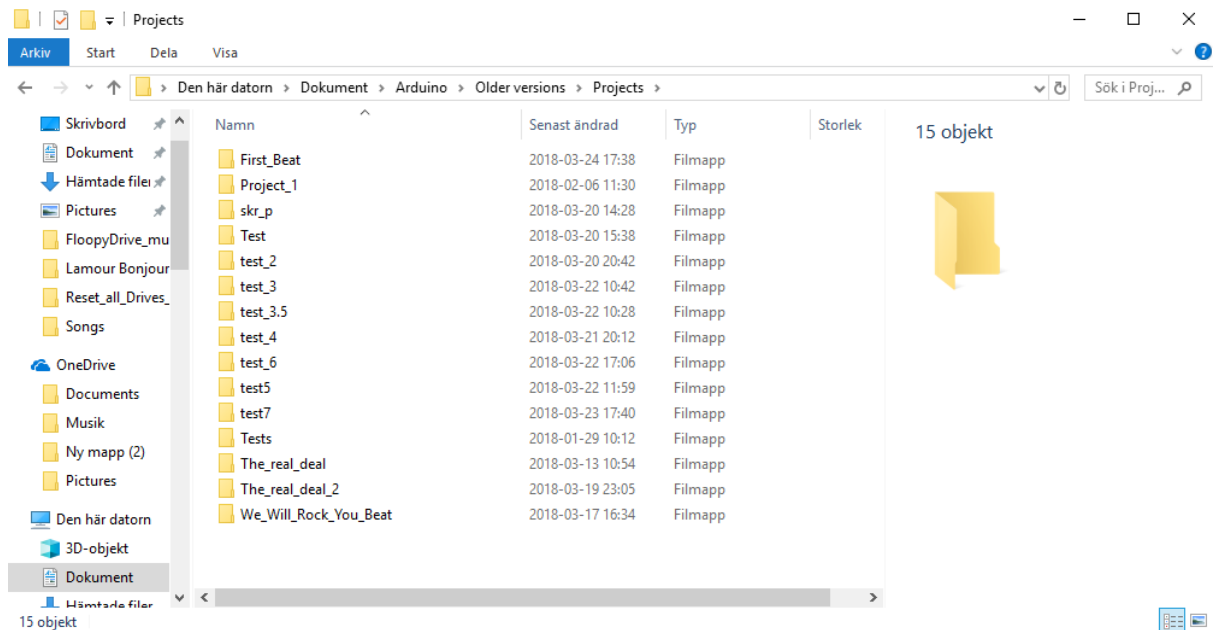
Arial 10 F K U

B260

	A	B	C	D	E	F	G	H	I
1	12184,	ch=1	n		66,			127,	
2	61,	ch=1	n		66,			0,	
3	7,	ch=1	n		66,			127,	
4	87,	ch=1	n		66,			0,	
5	0,	ch=1	n		68,			127,	
6	92,	ch=1	n		68,			0,	
7	0,	ch=1	n		69,			127,	
8	56,	ch=1	n		69,			0,	
9	2,	ch=1	n		66,			127,	
10	94,	ch=1	n		66,			0,	
11	5,	ch=1	n		66,			127,	
12	130,	ch=1	n		66,			0,	
13	433,	ch=1	n		66,			127,	
14	59,	ch=1	n		66,			0,	
15	3,	ch=1	n		66,			127,	
16	94,	ch=1	n		68,			127,	
17	5,	ch=1	n		66,			0,	
18	84,	ch=1	n		68,			0,	
19	2,	ch=1	n		67,			127,	
20	130,	ch=1	n		67,			0,	
21	6,	ch=1	n		66,			127,	
22	179,	ch=1	n		66,			0,	
23	394,	ch=1	n		66,			127,	
24	61,	ch=1	n		66,			0,	
25	7,	ch=1	n		66,			127,	
26	69,	ch=1	n		66,			0,	
27	9,	ch=1	n		68,			127,	
28	90,	ch=1	n		68,			0,	
29	1,	ch=1	n		69,			127,	
30	54,	ch=1	n		68,			127,	
31	3,	ch=1	n		69,			0,	
32	116,	ch=1	n		68,			0,	
33	5,	ch=1	n		64,			127,	
34	121,	ch=1	n		61,			127,	
35	11,	ch=1	n		64,			0,	
36	111,	ch=1	n		61,			0,	
37	313,	ch=1	n		69,			127,	
38	44,	ch=1	n		69,			0,	
39	18,	ch=1	n		68,			127,	
40	80,	ch=1	n		68,			0,	
41	6,	ch=1	n		66,			127,	
42	82,	ch=1	n		66,			0,	
43	9,	ch=1	n		64,			127,	
44	127,	ch=1	n		64,			0,	
45	6,	ch=1	n		64,			127,	
46	136,	ch=1	n		64,			0,	
47	435,	ch=1	n		66,			127,	
48	64,	ch=1	n		66,			0,	
49	7,	ch=1	n		66,			127,	

Ark1 Ark2 Ark3

Ark 1 / 3 Standard



## FloppyDrive\_music\_template

```
currentMillispasted5 = millis();
currentMillis6 = millis();
currentMillispasted6 = millis();

if ( pgm_read_word_near(&(velocityCh1[indexCh1])) > 0) {on1 = 1;} else {on1 = 0;};

Length1=(pgm_read_word_near(&(ticksDeltaCh1[indexCh1]))*4.2735);
convertednote1 = (74-pgm_read_word_near(&(notesCh1[indexCh1]))); // konstanten ska alltid vara den högsta noten som spelas
note1 = (convertednote1 * 0.6)+8; //ändra konstanten via floopysformel för att ändra spannet på noter

if ( pgm_read_word_near(&(velocityCh2[indexCh2])) > 0) {on2 = 1;} else {on2 = 0;};

Length2 =(pgm_read_word_near(&(ticksDeltaCh2[indexCh2]))*4.2735);
convertednote2 = (74-pgm_read_word_near(&(notesCh2[indexCh2]))); // konstanten ska alltid vara den högsta noten som spelas
note2 = (convertednote2 * 0.6)+8; //ändra konstanten via floopysformel för att ändra spannet på noter

if ( pgm_read_word_near(&(velocityCh3[indexCh3])) > 0) {on3 = 1;} else {on3 = 0;};

Length3 =(pgm_read_word_near(&(ticksDeltaCh3[indexCh3]))*4.2735);
convertednote3 = (74-pgm_read_word_near(&(notesCh3[indexCh3]))); // konstanten ska alltid vara den högsta noten som spelas
note3 = (convertednote3 * 0.6)+8; //ändra konstanten via floopysformel för att ändra spannet på noter

if ( pgm_read_word_near(&(velocityCh4[indexCh4])) > 0) {on4 = 1;} else {on4 = 0;};

Length4 =(pgm_read_word_near(&(ticksDeltaCh4[indexCh4]))*4.2735);
convertednote4 = (74-pgm_read_word_near(&(notesCh4[indexCh4]))); // konstanten ska alltid vara den högsta noten som spelas
note4 = (convertednote4 * 0.6)+8; //ändra konstanten via floopysformel för att ändra spannet på noter

if ( pgm_read_word_near(&(velocityCh5[indexCh5])) > 0) {on5 = 1;} else {on5 = 0;};

Length5 =(pgm_read_word_near(&(ticksDeltaCh5[indexCh5]))*4.2735);
convertednote5 = (74-pgm_read_word_near(&(notesCh5[indexCh5]))); // konstanten ska alltid vara den högsta noten som spelas
note5 = (convertednote5 * 0.6)+8; //ändra konstanten via floopysformel för att ändra spannet på noter

if ( pgm_read_word_near(&(velocityCh6[indexCh6])) > 0) {on6 = 1;} else {on6 = 0;};

Length6 =(pgm_read_word_near(&(ticksDeltaCh6[indexCh6]))*4.2735);
convertednote6 = (74-pgm_read_word_near(&(notesCh6[indexCh6]))); // konstanten ska alltid vara den högsta noten som spelas
note6 = (convertednote6 * 0.6)+8; //ändra konstanten via floopysformel för att ändra spannet på noter
```

FloppyDrive\_music\_template

```
unsigned long Length6 = 0;
```

```
unsigned long previousDriveInote05=0;
```

```
void setup ()
```

```
{
```

```
  Serial.begin(9600);
```

```
  pinMode(DRIVE1,OUTPUT);
```

```
  pinMode(SWITCH1,OUTPUT);
```

```
  pinMode(DRIVE2,OUTPUT);
```

```
  pinMode(SWITCH2,OUTPUT);
```

```
  pinMode(DRIVE3,OUTPUT);
```

```
  pinMode(SWITCH3,OUTPUT);
```

```
  pinMode(DRIVE4,OUTPUT);
```

```
  pinMode(SWITCH4,OUTPUT);
```

```
  pinMode(DRIVE5,OUTPUT);
```

```
  pinMode(SWITCH5,OUTPUT);
```

```
  pinMode(DRIVE6,OUTPUT);
```

```
  pinMode(SWITCH6,OUTPUT);
```

```
}
```

```
void loop()
```

```
{
```

```
currentMillis1 = millis();
```

```
currentMillispasted1 = millis();
```

```
currentMillis2 = millis();
```

```
currentMillispasted2 = millis();
```

```
currentMillis3 = millis();
```

```
currentMillispasted3 = millis();
```

```
currentMillis4 = millis();
```

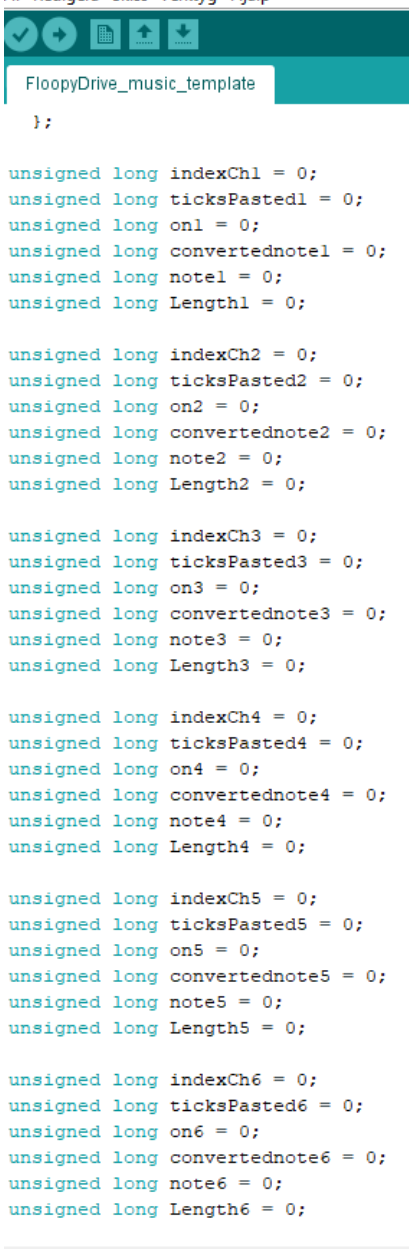
```
currentMillispasted4 = millis();
```

```
currentMillis5 = millis();
```

```
currentMillispasted5 = millis();
```

```
currentMillis6 = millis();
```

```
currentMillispasted6 = millis();
```



```
};

unsigned long indexCh1 = 0;
unsigned long ticksPasted1 = 0;
unsigned long on1 = 0;
unsigned long convertednote1 = 0;
unsigned long note1 = 0;
unsigned long Length1 = 0;

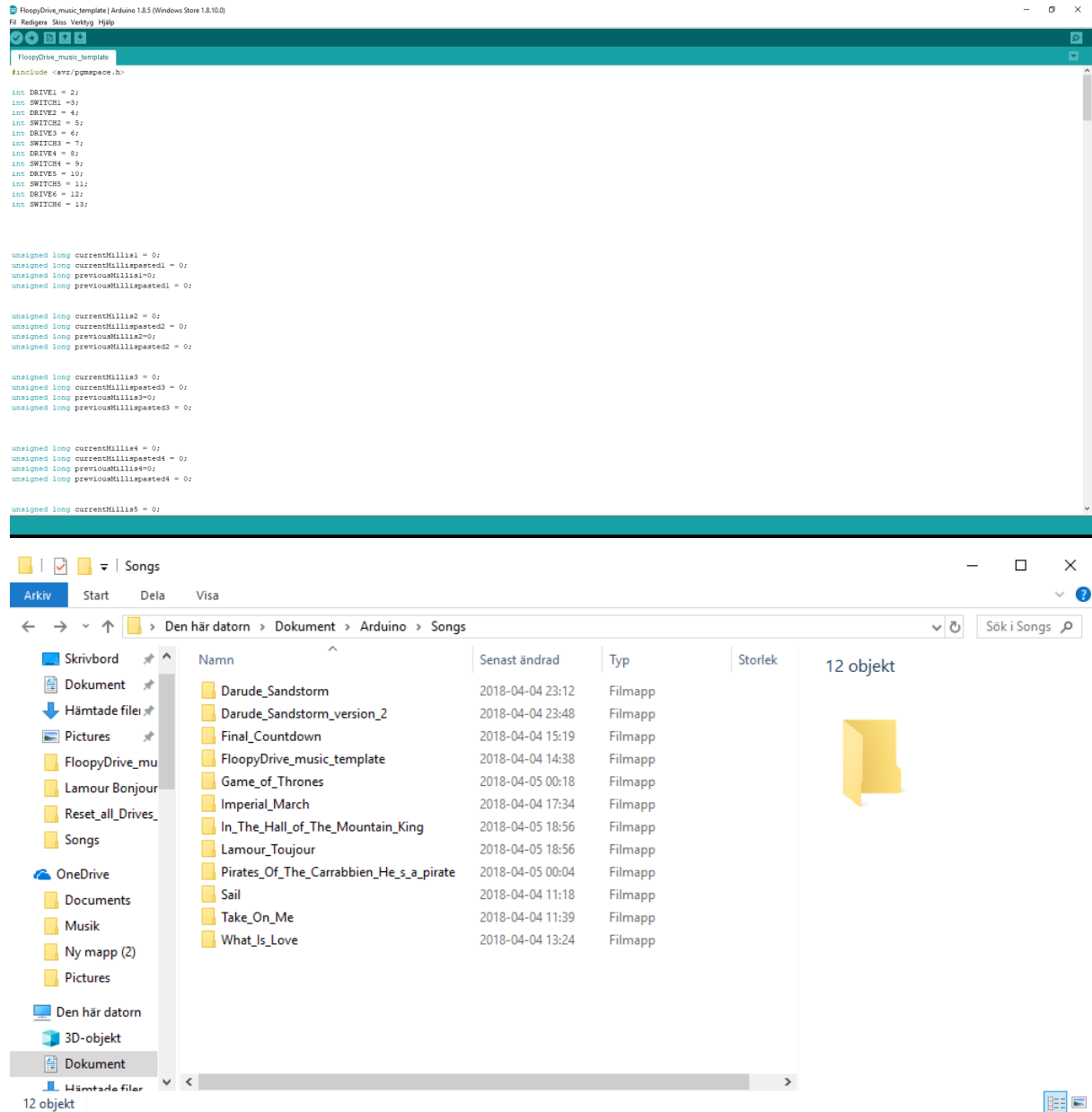
unsigned long indexCh2 = 0;
unsigned long ticksPasted2 = 0;
unsigned long on2 = 0;
unsigned long convertednote2 = 0;
unsigned long note2 = 0;
unsigned long Length2 = 0;

unsigned long indexCh3 = 0;
unsigned long ticksPasted3 = 0;
unsigned long on3 = 0;
unsigned long convertednote3 = 0;
unsigned long note3 = 0;
unsigned long Length3 = 0;

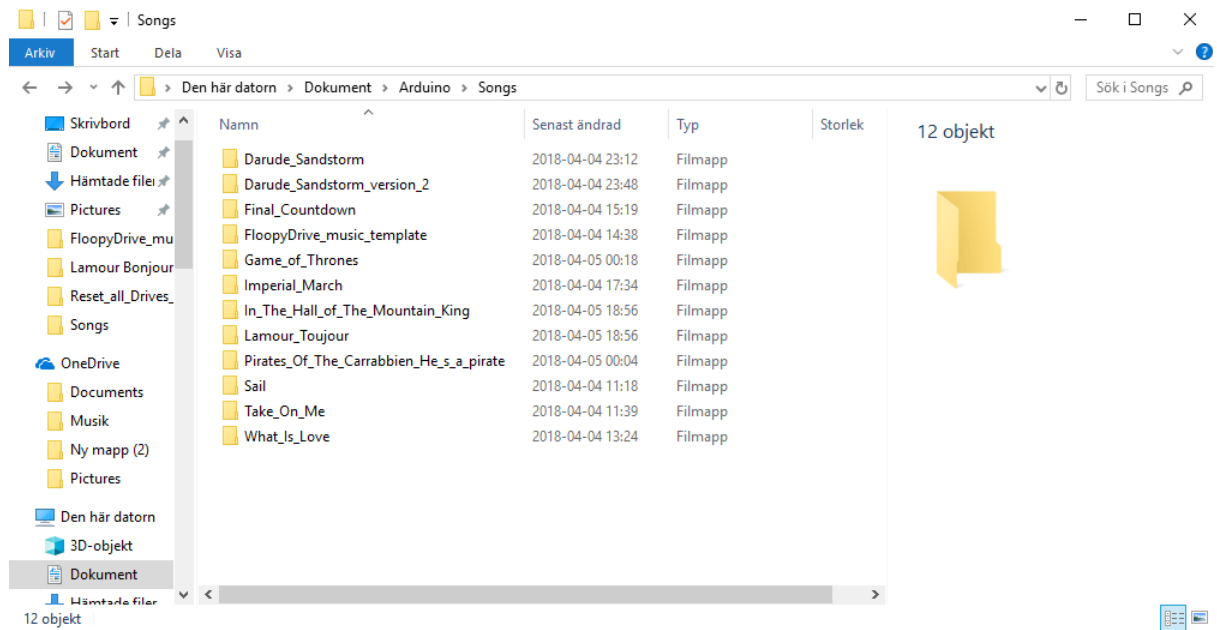
unsigned long indexCh4 = 0;
unsigned long ticksPasted4 = 0;
unsigned long on4 = 0;
unsigned long convertednote4 = 0;
unsigned long note4 = 0;
unsigned long Length4 = 0;

unsigned long indexCh5 = 0;
unsigned long ticksPasted5 = 0;
unsigned long on5 = 0;
unsigned long convertednote5 = 0;
unsigned long note5 = 0;
unsigned long Length5 = 0;

unsigned long indexCh6 = 0;
unsigned long ticksPasted6 = 0;
unsigned long on6 = 0;
unsigned long convertednote6 = 0;
unsigned long note6 = 0;
unsigned long Length6 = 0;
```



[illegible]





FloopyDrive\_music\_template | Arduino 1.8.5 (Windows Store 1.8.10.0)

Fil Redigera Skiss Verktyg Hjälp



FloopyDrive\_music\_template

```
// DRIVE1
const unsigned int notesCh1[] PROGMEM = {1000,

1000, // tyst not, ska alltid vara sist
};
const unsigned int velocityCh1[] PROGMEM = {0,

0, // tyst not, ska alltid vara sist
};
const unsigned int ticksDeltaCh1[] PROGMEM = {

65000, // tyst not, ska alltid vara sist
};

//DRIVE 2
const unsigned int notesCh2[] PROGMEM = {1000,

1000, // tyst not, ska alltid vara sist
};
const unsigned int velocityCh2[] PROGMEM = {0,

0, // tyst not, ska alltid vara sist
};
const unsigned int ticksDeltaCh2[] PROGMEM = {

65000, // tyst not, ska alltid vara sist
};
// DRIVE3
const unsigned int notesCh3[] PROGMEM = {1000,

1000, // tyst not, ska alltid vara sist
};
const unsigned int velocityCh3[] PROGMEM = {0,

0, // tyst not, ska alltid vara sist
};
const unsigned int ticksDeltaCh3[] PROGMEM = {

65000, // tyst not, ska alltid vara sist
};
// DRIVE4
const unsigned int notesCh4[] PROGMEM = {1000, // kan vara vilken not som helst
```

